Long Range Contacts in Overlay Networks with Unbalanced Node Distribution

Filipe Araújo
Luís Rodrigues

DI–FCUL TR–04–8

July 22, 2004

Departamento de Informática
Faculdade de Ciências da Universidade de Lisboa
Campo Grande, 1749–016 Lisboa
Portugal

Technical reports are available at `http://www.di.fc.ul.pt/tech-reports`. The files are stored in PDF, with the report number as filename. Alternatively, reports are available by post from the above address.

Long Range Contacts in Overlay Networks with Unbalanced Node Distribution

Filipe Araújo
Universidade de Lisboa
filipius@di.fc.ul.pt

Luís Rodrigues
Universidade de Lisboa
ler@di.fc.ul.pt

July 22, 2004

Abstract

A fundamental aspect in the design of overlay networks is the path length/node degree trade-off. Previous research has shown that it is possible to achieve logarithmic path lengths for logarithmic or even constant node degree. While nearby contacts, with nodes that have close identifiers, ensure a connected lattice of nodes, short path lengths demand for the use of long range contacts. In this respect, previous work exhibits limitations in scenarios where node distribution is unbalanced: either short path length properties do not hold or may require node degree and/or signaling to grow with respect to the virtual identification space instead of the number of nodes (which is usually several order of magnitudes smaller).

This paper proposes and evaluates a new mechanism to establish long range contacts in unbalanced overlay networks. This mechanism does not need any kind of manual configuration to adapt to different network configurations and is oblivious to the virtual identification space. Experimental comparison with previous work suggests that our mechanism achieves logarithmic path lengths with respect to the number of nodes, but regardless of node distribution in the virtual identification space.

1 Introduction

An overlay network is a network operated on top of another underlying network (e.g., IP), but organized under an independent logic. For this reason overlay networks are also deemed "logical" or "virtual" networks. In recent years, a significant body of research has been devoted to the study of overlay networks capable of implementing distributed hash tables (DHTs) in a simple and efficient way. A fundamental efficiency metric is the path length/node degree trade-off. Path lengths are measured in terms of hops traveled by a message in the overlay network, while the degree of a node is the number of its overlay neighbors. These parameters are tightly connected and cannot be varied independently, as any n-node network meets its fundamental limits in the inequality $d^{h+1}/(d-1) \geq n-1$, because $d^{h+1}/(d-1)$ is an upper bound for the number of nodes within h hops for a given maximum node degree d. In fact, it can be seen in [1] that for $O(1)$ node degree, expected path lengths can be, at best, $O(\log n)$, while for $O(\log n)$ node degree, expected path lengths cannot be shorter than $O(\log n/\log\log n)$.

When the distribution of nodes in the address space is uniform, there are many overlay networks that achieve results that are either near-optimal or even optimal with respect to the path length/node degree trade-off (see Table 1). Often, in these overlay networks, it is possible to distinguish between two different types of contacts: "nearby" contacts, forming a kind of connected lattice between nodes that have close virtual identifications, and "long range contacts" (LRCs) between nodes that have "distant" virtual node identifications. While the former type of contacts may be important in certain overlays to ensure connectedness and routing convergence, short path lengths actually depend on the latter type of contacts. In fact, it is the capability to "jump" over many closer nodes in a single hop that makes it possible to achieve short path lengths. Designers

Table 1: Comparison between *expected* performance of several peer-to-peer systems

P2P system	Node degree	Path lengths
Small-worlds [4]	$O(1)$	$O(\log^2 n)$
Chord [2], Pastry [5], Tapestry [6], eCAN [3]	$O(\log n)$	$O(\log n)$
Koorde [1] (config. 1), D2B [7], Viceroy [8]	$O(1)$	$O(\log n)$
Koorde [1] (config. 2)	$O(\log n)$	$O(\log n / \log\log n)$
CAN [9]	$O(d)$	$O(dn^{1/d})$

may opt to cover the entire identification space with LRCs, thus coupling the number of LRCs and/or signaling to the size of that space, which may be several order of magnitudes greater than the number of nodes (e.g., Chord [2] [1]) or they may use a sparser cover of the identification space (e.g., typical configurations of expressways Content-Addressable Networks, eCAN [3]), which will not ensure logarithmic path lengths if distribution of nodes is not sparse and homogeneous.

This paper is motivated by the observation that there are circumstances where a homogeneous distribution of nodes cannot be assumed. Consider for example "landmark ordering" used in CAN [9] to improve the overlay network with topological information. Here, every node of the overlay network selects an identification based on its distance in terms of round-trip-time (RTT) to a well-known set of IP machines, called "landmarks". Yet another example are location-aware applications requiring a DHT service from a set of nodes scattered in space defining a bi-dimensional network. Such a scenario requires nodes to use real physical locations, thus precluding balancing from being uniform, in general.

Therefore, we propose a new mechanism to establish and maintain LRCs, called "hop level", capable of extracting excellent performance with resources which are a logarithmic function of the *effective* number of nodes, *regardless* of node identification space. This mechanism ignores any form of virtual or physical distance between nodes and only takes into account the distance in terms of hops. As a consequence, hop level mechanism is oblivious to the way nodes are distributed in the name space. Furthermore, it automatically adapts the number and positions of the LRCs to the characteristics of the underlying network without requiring any sort of manual or pre-defined configuration. When compared to current logarithmic/logarithmic LRC schemes, hop level achieves similar results in balanced scenarios and offers better performance with unbalanced distributions. To experimentally evaluate our solution we have used a location-aware peer-to-peer system [10].

The remainder of the paper is organized as follows: Section 2 states the problem we are solving. Section 3 overviews previous work. Our long range contact mechanism is described and evaluated, respectively in Sections 4 and 5. Section 6 concludes the paper and points directions for future work.

2 Problem Statement

Throughout this paper we will consider a scenario where routing convergence is ensured by some kind of bi-dimensional lattice that uses short range contacts to connect nodes with close identifications. Meshes formed by a bi-dimensional CAN, a Delaunay triangulation [10], or a structure made of squares as in [4] serve as examples of lattices. However, we emphasize that nodes are not required to be evenly spread in space. Moreover, although we are considering a world made of bi-dimensional surfaces, we believe that our algorithm can be also adopted in multidimensional or even unidimensional scenarios.

Furthermore, we will consider the use of a routing scheme where i) the preprocessing algorithm can only collect information of $O(1)$ nearby peers and $O(\log n)$ distant peers per node and ii) the

[1] Although Chord may discard repeated LRCs, creation of such LRC is nevertheless attempted.

routing algorithm will select, among the forwarding node's contacts (either short or long range), the one which is closest to destination in terms of Euclidean distances [2].

Given this scenario, our goal is to design a mechanism that creates and maintains a set of LRCs at each node with the following properties: *i*) each node should only create and store $O(\log n)$ LRCs, where n is the number of nodes in the system and does not bear any relation to the virtual (or physical) identification space. *ii*) guaranteed routing convergence and expected $O(\log n)$ path lengths *despite* non-uniform node distribution. The reader may notice that these properties are required to build efficient overlay networks even when the population of the address space is sparse and unbalanced.

Before presenting our hop level mechanism, we will overview previous research in the topic of overlay networks to capture the relevant features that should be owned by efficient sets of LRCs.

3 Related Work

There is a huge body of work related with overlay networks. In most of them, separation between nearby and long-range contacts is only implicit and in some cases non-existent as there is no underlying lattice. For instance, Chord [2], Viceroy [8] and Koorde [1] use a ring as the underlying lattice. They only differ in the way they organize their LRCs. In Pastry [5], the existence of an underlying lattice is not so explicit, but we could consider the entire set of individual node's leaf sets as a lattice between nearby nodes. Unlike these overlay networks, neither Tapestry [6], nor D2B [7] use an underlying lattice to route messages. Contrary to the previous examples, CAN [9] has no LRCs but only short range contacts. For this reason, it has longer path lengths. To overcome this problem, Xu and Zhang [3] proposed a mechanism called "expressway CAN" that augments basic CAN with LRCs. Although their work is fairly complex and includes many subtleties to account for topological information in the selection of the LRCs, the main idea of their work is captured in the algorithm eCAN-like that we use in Section 5 for comparison purposes. It is worthwhile mentioning that to achieve short path lengths, these overlay networks assume a homogeneous distribution of nodes. Some, like Chord might resist to a disadvantageous distribution, but at the cost of trying to populate the entire node identification space with LRCs. Table 1 resumes expected performance of the overlay networks mentioned here. Unlike most other work, SkpiNet [11] was built from scratch to cope with the unbalanced use of identification space. However, identification space of a SkipNet is unidimensional and extension to the more complex two-dimensional case, as we address here, is not trivial. In its most obvious application, SkipNet identifies nodes by using their domain name as a prefix, to allow queries or location of resources to remain within scope of some organization sharing a common domain name, like "someexistingname.com".

Perhaps the works that are closer in spirit to this paper are those of Kleinberg [4], Barrière *et al.* [12] and of Xu and Zhang [3] as they all address an explicit separation between an underlying lattice (often bi-dimensional) and a set of LRCs created to ensure short path lengths. The work of Kleinberg [4] models the small world phenomenon as a lattice of squares with $O(1)$ contacts, where nodes select their single LRC according to a random process based on the distance to their peers raised to the power $-r$. The interesting conclusion of this work is that power $r = 2$ represents the correct balance between the geographical information implicit in the LRCs and their ability to forward messages to large distances. Other overlay networks, with $O(\log n)$ contacts, implicitly follow a similar principle and keep a nearly constant number of LRCs for exponentially larger disjoint areas of the virtual space around the identification of each node. Hence, our work will inherit from the idea of distributing a nearly constant number of LRCs between disjoint groups of nodes of exponentially different sizes. Crucially, the fundamental difference we introduce, is the decoupling between the number of surrounding nodes and the size of the surrounding area, as these may not coincide.

[2]There is no loss of generality in assuming Euclidean distances, as other metrics could also be used if more appropriate to the structure of the lattice, e.g., Manhattan distance or unidimensional virtual identification distance.

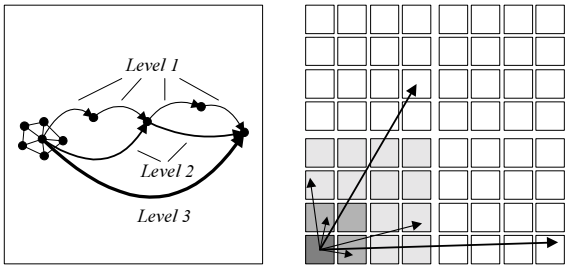

Figure 1: LRC mechanisms: a) Hop level, b) eCAN-like

4 Hop Level LRCs Mechanism

We now describe our proposal to build LRCs in unbalanced and sparse overlays. The goal of the *hop level* mechanism is to prevent messages from doing more than a predefined number of hops, say b hops. To achieve that goal, LRCs are established automatically whenever a message is detected to perform b consecutive hops. Consider, for instance, that some node F is forwarding a message m from S to N_1. Assume that F realizes that this will be the b-th hop of the message. In this case, F triggers the creation of a LRC from S to N_1, denoted $S \xrightarrow{1} N_1$, by sending a control message to S. The process is repeated, this time from N_1 onwards: if after b hops, message m reaches N_2, N_1 will create a LRC to N_2, $N_1 \xrightarrow{1} N_2$, and so on. Let us call these LRCs, *level-1 LRCs*. If the message path is very long, it may happen that a sequence of b *level-1 LRCs* occurs, for instance: $S \xrightarrow{1} N_1$, $N_1 \xrightarrow{1} N_2$, ..., $N_{b-1} \xrightarrow{1} N_b$. In this case, a new LRC from S directly to N_b should be created. This new LRC, $S \xrightarrow{2} N_b$, is one level above of the previous ones. This mechanism should be applied recursively for all levels. Hence, a LRC of level-l jumps over b^l hops. Figure 1a) illustrates our mechanism.

When some forwarding node uses a LRC of level-l to send a message, the actions it must perform depend not only on level-l, but also on the level of the LRC used by the previous hop node, say level-p. Consider that $l > p$. In this case, the previous level-p hop can no longer be joined with subsequent level-p hops to form a level-$(p+1)$ LRC because only sequences of *consecutive* hops of the same level must be accounted for. Therefore, if $l > p$, information regarding previous hops must be invalidated. Now consider that message m is going to be sent along its b-th consecutive hop of level-l to node N. In this case, forwarding node F sends a control message to the node that initiated the sequence of level-l, prompting it to create a LRC of level-$(l+1)$ to node N. Since this signifies that b hops of level-l are worth 1 hop of level-$(l+1)$, node F must set the number of hops of level-l to 0 and increment the number of hops of level$(l+1)$ by 1. Should this substituting hop become the b-th hop of level-$(l+1)$, the same process is repeated for level-$(l+1)$, and so on, until a level with fewer than b hops is reached.

We illustrate our mechanism with some concrete examples in a network where b is set to 2. Consider the following message path ($\rightarrow$ represents a hop where no LRC is used): $N_0 \rightarrow N_1 \rightarrow N_2 \rightarrow N_3 \rightarrow N_4 \rightarrow N_5 \rightarrow N_6 \rightarrow N_7 \rightarrow N_8$. In this case, the following LRCs would be created: $N_0 \xrightarrow{1} N_2$, $N_2 \xrightarrow{1} N_4$, $N_4 \xrightarrow{1} N_6$, $N_6 \xrightarrow{1} N_8$, $N_0 \xrightarrow{2} N_4$, $N_4 \xrightarrow{2} N_8$, and $N_0 \xrightarrow{3} N_8$, for a grand total of seven LRCs. Consider now the following message path: $N_0 \xrightarrow{3} N_8 \xrightarrow{1} N_{10} \xrightarrow{1} N_{12} \xrightarrow{2} N_{16}$. In this case, the following additional LRCs would be created: $N_8 \xrightarrow{2} N_{12}$, $N_8 \xrightarrow{3} N_{16}$, and $N_0 \xrightarrow{4} N_{16}$. On the other hand, a message path such as $N_0 \xrightarrow{1} N_2 \xrightarrow{4} N_{18} \xrightarrow{1} N_{20}$ would not trigger the creation of any additional LRC.

Interestingly, our mechanism does not require the definition of *a priori* limits to the number of LRCs, as it naturally ceases to create more LRCs, as performance evolves from the $O(n^{1/2})$ to $O(\log n)$. In fact, the number of LRCs per level should never grow beyond a constant threshold

that reflects saturation of a perimeter by LRCs (for instance, if only 10 or 12 nodes exist at 2 hops of distance, there will be typically only 6 or so LRCs of level 1 by node). Hence, one of the fundamental characteristics of the hop level mechanism is that it does not need any kind of manual configuration as it naturally adapts to different network conditions.

We now describe our hop level LRC mechanism in a more precise way. Our scheme requires messages to carry some control information and an algorithm to be executed at every node, whenever a message is routed. We describe here a version of hop level that requires a minimum of two variables per level l to be carried in each message m: the number of hops, $nh_m[l]$, and the node that may receive a new LRC of that level, $s_m[l]$ (we later discuss how to reduce the signaling cost). Whenever level counter $nh_m[l-1]$ reaches the limit b, a new LRC, starting at $s_m[l]$ should be created. Additionally, messages must carry the level p_m of the LRC used by the previous hop to reach F, and an indication of the highest level of the array that contains valid information, max_m. Each node F, when forwarding the message m to N executes Algorithm 1. The variable l captures the level of the LRC used by F to reach the next hop N. If F is the source of the message, $F = S$, it is necessary to set previous level $p_m \leftarrow \perp$. In this case, the execution of the algorithm will initialize $max_m \leftarrow l+1$, $s_m[max] \leftarrow S$ and $nh_m[max-1] \leftarrow 0$.

Algorithm 1 Hop Level mechanism (executed at node F when forwarding message m to node N)

{Control information carried in message m:}
{max_m — highest valid level; }
{p_m — level of LRC used to reach F; }
{$\forall k \in [0, max_m] : nh_m[k], s_m[k]$ — resp., number of hops and first node of level-k;}

$l \leftarrow$ level of LRC from F to N ($F \xrightarrow{l} N$)
if $p_m = \perp$ **or** $p_m < l$ **then**
  $max_m \leftarrow l+1$; $lim \leftarrow max_m$
else
  $lim \leftarrow p_m$
end if
for all $k \in \{l, \ldots, lim-1\}$ **do**
  $s_m[k+1] \leftarrow F$; $nh_m[k] \leftarrow 0$
end for
$nh_m[l] \leftarrow nh_m[l] + 1$
while $nh_m[l] \geq b$ **do**
  $nh_m[l] = 0$
  send control message to $s_m[l+1]$ to create LRC $s_m[l+1] \xrightarrow{l+1} N$
  $l \leftarrow l+1$; $nh_m[l] \leftarrow nh_m[l] + 1$
  if $max_m == l$ **then**
    $max_m \leftarrow max_m + 1$; $s_m[max_m] \leftarrow s_m[max_m - 1]$; $nh_m[max_m - 1] \leftarrow 0$
  end if
end while

Signaling Cost The hop level mechanism presented in Algorithm 1 requires $O(\log n \times (\log b + \log N))$ bits in each message to store the arrays $nh[k]$ and $s[k]$, where n is the effective number of nodes and N is the size of the virtual identification space. This can be reduced $O(\log n \times \log b)$ by message by making nodes store back pointers to previous hops. This optimization is discussed in the Appendix.

5 Evaluation

5.1 Experiment Settings

In this section we experimentally evaluate hop level with $b = 2$ and eCAN-like through simulation, using the Delaunay triangulation built by a location-aware peer-to-peer system called GeoPeer [10] as the underlying network. To do this we send a large number of messages between arbitrary pairs of nodes, under the following variables conditions: (i) 100, 500, 1000, 5000, 10000 and 50000

nodes and (*ii*) balanced distribution of nodes, vs. unbalanced distribution. To unbalance node distribution we used a truncated Gaussian bivariate distribution with standard deviations of 1.0, 0.1 and 0.01 in a $[0,1]\times[0,1]$ square [3]. To route the messages we used the greedy routing algorithm, as it has good performance and requires no extension to use LRCs. Furthermore, it agrees to the conditions of Section 2. Hence, next hop is always the neighbor (connected by a short or long range contact) closest to destination. To let hop level LRC scheme converge, and depending on the network size, we made up to $200,000,000$ routing operations and only used the final 3000 paths in the evaluation of path lengths.

5.2 eCAN-like Mechanism

To offer some comparative measurement, we run our scheme against a benchmark mechanism called "eCAN-like". This benchmark results from an adaptation of the eCAN [3] logarithmic/logarithmic node degree/path length mechanism (whose applications most closely resemble those of our own algorithm). We must emphasize that the resulting mechanism, "eCAN-like", is a simplified version of the complete eCAN solution, that only captures the fundamental impact of the expressways in routing, and does not attempt to reproduce other features of eCAN (such as the mechanisms that provide support for complex interaction schemes like publish/subscribe). In spite of these simplifications, we believe that our implementation of expressways mimics the eCAN LRC mechanism with enough accuracy to allow a fair comparison.

The idea in eCAN-like is to make a first level division of the entire space in four big squares. Each node keeps LRC to the two neighboring squares. Then, the four big squares are further divided in other four smaller squares. This time, nodes inside squares have a total number of four LRC (above, below, right and left). This process is repeated for as many levels as wanted. Figure 1b) illustrates the eCAN-like LRC scheme. In our context, we fixed the number of levels to 8, in a total of 30 LRCs. It should be noticed that, since the center of a given square will probably not correspond to any existing node, the actual LRC will be the node responsible for that center point.

5.3 Analysis of Path Lengths

Figure 2 plots path lengths as a function of the number of nodes for balanced and most unbalanced scenarios (in a logarithmic scale). Since a logarithmic growth must appear as a straight line, it is clear that eCAN-like is no longer logarithmic when nodes follow a Gaussian distribution with standard deviation 0.01. Actually, such a scenario with a strong concentration near the center is not only possible, but also very likely, for instance, in a location-aware network, near the center of a densely populated urban area. The reason for the bad behavior of the eCAN-like mechanism is easily explainable: density of LRCs is no longer enough near the center and routing to nearby nodes will tend to become linear with the number of hops in the lattice, instead of logarithmic. Increasing the levels of the LRCs would solve the problem, unless, of course, density near the center was also increased. The only solution to this limitation seems to be using a brute-force approach with a number of LRCs logarithmic to the space granularity (i.e., to the size of the virtual identification space). Superior results of hop level mechanism, in the balanced scenario, are due to the higher number of LRCs used per level. These results, as well as results obtained for the remaining Gaussian curves, (omitted to conserve space), suggest that our scheme is nearly location-independent.

[3] To do this, we used the Box-Muller transformation and a translation to transform a two-dimensional continuous uniform distribution in the square $[0,1]\times[0,1]$ to a Gaussian bivariate distribution centered at $(0.5, 0.5)$ with the standard deviations mentioned. Truncation came from elimination of all points outside the $[0,1]\times[0,1]$ square. Finally, the square was transformed to a side of 500.

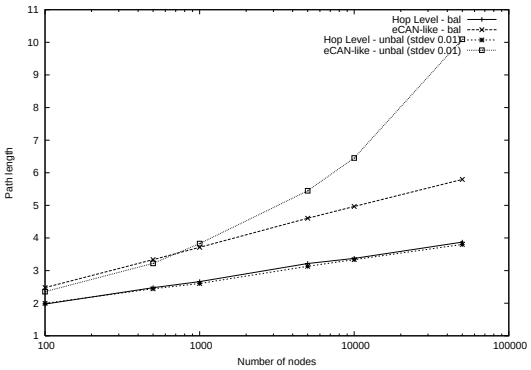

Figure 2: Comparison of path lengths

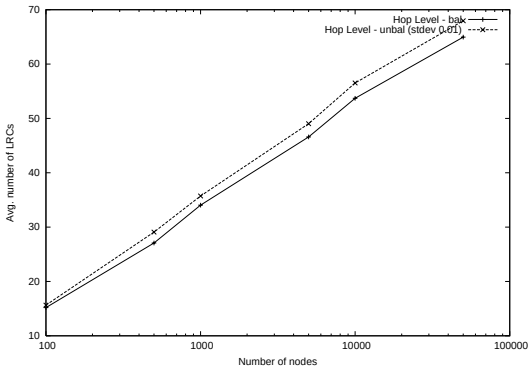

Figure 3: Average number of LRCs

5.4 Analysis of Created LRCs

A first fundamental aspect that can be observed in Figure 3 is that, for the network sizes we tested, the number of LRCs per node seems to grow logarithmically with respect to network size, despite not having any pre-configured limit. This is not surprising, since, as pointed out in Section 4, if the distribution of LRCs for a given level is reasonable, they should be near the perimeter of a circle (defined in terms of hops) around the node. The figure also shows that, for the most unbalanced scenario, slightly more LRCs are created, probably as a consequence of a worse distribution of LRCs around those perimeters. We believe that the routing algorithm is the cause of this effect, because it is the only source introducing dependence to location, as it is not aware of distances in terms of hops, but in terms of virtual space identification.

Figure 4 shows how many LRCs exist on the entire network and the average length of those LRCs for each hop level (50000 nodes). Growth of the number of LRCs per level is quite moderate, especially after the first two levels. Moreover, we could also observe a slight decrease of the average number of LRCs in the lower levels, from the 10000 to the 50000-node networks. These facts help to explain the logarithmic growth shown in Figure 3. Furthermore, the exponential-like growth of the distances of the LRCs, in the balanced scenario, allows to conclude that the level of the LRC strongly reflects its distance in terms of hops to origin. It should be noticed that the decrease in the number of LRCs, as well as the irregular behavior of the distances in the upper levels, are a consequence of the border effects introduced by the square space. Finally, the distribution of LRCs among the nodes looks like a normal curve. For instance, in the 50000-node balanced network 99.6% of the nodes have between 20 and 100 LRCs with a minimum of 1 and a maximum of 163 (not shown).

Figure 5 shows the total number of LRCs in the network as a function of the number of messages (balanced scenario with 50000 nodes). We could observe a similar curve for all network sizes, with a very fast growth at first, progressively slowing down. Our data clearly shows that second derivative of this growth function is negative, everywhere.

5.5 Discussion

Based on the previous discussion we believe that our hop level mechanism has the following interesting features:

- Hop level mechanism seems to achieve logarithmic path lengths, with a logarithmic number of LRCs, both with respect to the number of nodes (but not the identification space).
- Hop level mechanism is decoupled from node identification space (except in what concerns addresses) and therefore does not try to cover entire virtual identification space with LRCs (i.e., fewer bits dedicated to create LRCs).
- Node distribution has very little impact on hop level mechanism.

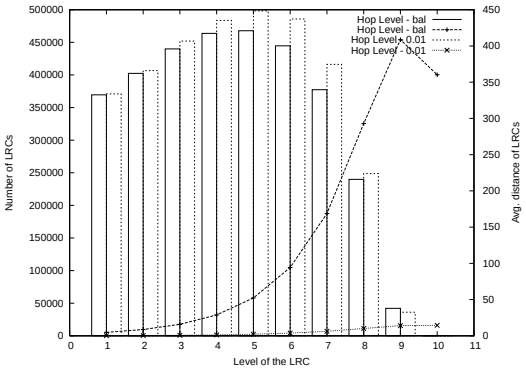

Figure 4: LRCs per level

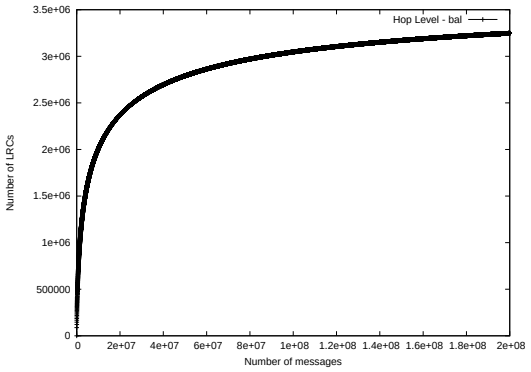

Figure 5: Growth in the number of LRCs

One aspect that deserves attention is the relatively slow bootstrap of the hop level mechanism. To obviate this problem, we propose two alternatives, presented in the Appendix, capable of creating LRCs more quickly.

6 Conclusions and Future Work

The contribution of this paper is a new mechanism, called "hop level", that manages Long Range Contacts (LRCs) in an overlay network. Unlike prior work that structures LRCs according to the virtual identification space occupied by nodes, hop level mechanism is oblivious to the virtual identification space. As a consequence, fundamental properties such as path length, node degree and signaling required, are relative to the number of nodes, which is typically several order of magnitudes smaller than the full size of the address space. Experimental results suggest that our mechanism effectively achieves logarithmic path length with logarithmic node degree, requiring logarithmic number of messages per node, all relative to the effective number of nodes. Furthermore, in all our experiments, hop level LRCs perform equally well for highly unbalanced node distributions. Although studied in the context of GeoPeer, this LRC mechanism is general, and should be applicable to many other underlying lattices with similar results.

As future work, we intend to study mechanisms to improve the quality of stored LRCs, by periodically adjusting the levels of LRCs according to network conditions, by selectively deleting less efficient contacts (for instance, LRCs to nodes that have failed and never recover) or by considering topological information.

Acknowledgements The authors are thankful to Patrick Eugster and Rachid Guerraoui for their comments on earlier versions of this paper.

References

[1] F. Kaashoek and D. R. Karger, "Koorde: A simple degree-optimal distributed hash table," 2003.

[2] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, "Chord: A scalable Peer-To-Peer lookup service for internet applications," in *ACM SIGCOMM*, San Diego, August 2001.

[3] Z. Xu and Z. Zhang, "Building low-maintenance expressways for p2p systems," HP, Tech. Rep. HPL-2002-41, 2002.

[4] J. Kleinberg, "The Small-World Phenomenon: An Algorithmic Perspective," in *Proceedings of the 32nd ACM Symposium on Theory of Computing*, 2000.

[5] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," *Lecture Notes in Computer Science*, vol. 2218, pp. 329–350, 2001.

[6] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing," UC Berkeley, Tech. Rep. UCB/CSD-01-1141, Apr. 2001.

[7] P. Fraigniaud and P. Gauron, "The content-addressable network D2B," LRI, Univ. Paris-Sud, France, Tech. Rep. 1349, Jan 2003.

[8] D. Malkhi, M. Naor, and D. Ratajczak, "Viceroy: A scalable and dynamic emulation of the butterfly," in *Twenty-First ACM Symposium on Principles of Distributed Computing (PODC 2002)*, Monterey, California, July 2002.

[9] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, "A scalable content-addressable network," in *Conference on applications, technologies, architectures, and protocols for computer communications*. ACM Press, 2001, pp. 161–172.

[10] F. Araújo and L. Rodrigues, "Geopeer: A location-aware peer-to-peer system," Department of Informatics, University of Lisbon, DI/FCUL TR 03–31, December 2003. [Online]. Available: http://www.di.fc.ul.pt/tech-reports/03-31.pdf

[11] N. J. A. Harvey, M. B. Jones, S. Saroiu, M. Theimer, and A. Wolman, "Skipnet: A scalable overlay network with practical locality properties," in *Fourth USENIX Symposium on Internet Technologies and Systems (USITS '03)*, Seattle, WA., March 2003.

[12] L. Barrière, P. Fraigniaud, E. Kranakis, and D. Krizanc, "Efficient routing in networks with long range contacts (extended abstract)," in *15th Internation Conference on Distributed Computing*, ser. Lecture Notes in Computer Science, J. Welch, Ed., no. LNCS 2180. Lisbon, Portugal: Springer, October 2001.

Appendix

A Decreasing the Signaling Cost

The hop level mechanism presented in the paper requires $O(\log n \times (\log b + \log N))$ bits in each message to store the arrays $nh_m[k]$ and $s_m[k]$, where n is the effective number of nodes and N is the size of the virtual identification space. This can be reduced by making nodes store back pointers to previous hops, instead of using the array $s_m[k]$. Back pointers will only require a limited amount of memory at nodes, as they can be cleaned periodically. This reduces the size needed to store the arrays to $O(\log n \times \log b)$ by message. Since addresses carried in the messages need $O(\log N)$ bits, this is an acceptable cost for practical uses.

To enable reconstruction of the complete information, *i*) nodes should store information on which levels they are endpoints for LRCs and *ii*) messages must carry a bit indicating whether some LRC should be created (set by the first node for which $nh_m[l] \geq b$, for some l). When message arrives at destination, if this bit is set, a special message to create LRCs must travel backward using the back pointers. Now, each node of the reverse path appends its own address and level of the new LRC to the message if itself should be the endpoint of a new LRC for some (previous) node [4] (cost of $O(\log N)$ per LRC). We do not detail on how do nodes recognize that they are the origin of a new LRC, because it is a trivial procedure. Hence, total cost of messages to create LRC may be reduced to $O(\log N)$ by LRC by hop, which is similar to other overlay networks (in fact, this is the only source of coupling to the virtual identification space that can not be eliminated when creating a new LRC). Difference is that hop level mechanism requires fewer such messages.

B Decreasing the Stabilization Time

In many overlay networks, entering nodes are required to create their LRCs as soon as possible to reach a stable state. Unlike these, our hop level mechanism defers creation of LRCs and takes advantage of ordinary messages between nodes to create those LRCs. Such a policy has a number of advantages, like better selection of LRCs (specially in unbalanced networks) and better load spreading, even if many nodes enter the network at once, as nodes will not immediately send a bunch of distant messages when they start. On the other hand, network will take more time to stabilize. To tackle this problem, we propose two alternatives that can complement our scheme.

The first alternative follows the approach of creating all LRCs at bootstrap, instead of deferring this creation. Hence, joining nodes start by sending a number of messages in different directions, approximately traveling in a straight line to "explore" their neighborhood. Then, similarly to the hop level mechanism, each b hops in a level fire the creation of a LRC in the level above. This approach would enable a fast setup, possibly achieving good path length and node degree, but, unlike the hop level mechanism, its adaptation to varying network topologies would be more difficult.

The second alternative we propose is a variation of the hop level mechanism, called "Fast Hop Level". In this version of the algorithm, to speed up network bootstrap, almost every $1/b$ hops of an ordinary message might result in the creation of a new LRC. To do that, forwarding node F never resets values from the array $nh_m[l]$, thus breaking the invariant of requiring b hops from the same level to be consecutive. Instead, when a LRC of level l is used, $nh_m[l]$ is incremented. If $nh_m[l] = b$, the same procedure as in hop level mechanism is adopted. Further, node F marks itself as the originator of level $s_m[l+1]$ from level l up to the topmost level for which $nh_m[k] = 0$ (k must not exceed the highest valid level). The rationale for this is that if $nh_m[k] = 0$, a number multiple of b hops was done at level k and node F can be considered the originator of a next, to be formed, LRC of level $k+1$. For instance, assume that node F is going to use a LRC of

[4] To simplify presentation we assumed that, unlike previous algorithm, it is the endpoint of the LRC that recognizes that situation.

level 3 to forward a message. If $nh_m[3] = 0$, then F marks $s_m[4] = F$ as itself might become the originator of a LRC of level 4. Furthermore, if $nh_m[4] = 0$, it might also become the originator of a LRC of level 5 and so on. As soon as there is some level k for which $nh_m[k] > 0$ node F might not become the originator of any LRC from $k + 1$ up to the topmost valid level. It is easy to see that level numeration in this version of the hop level mechanism is very coarse. This is a consequence of trading LRC distribution quality by bootstrap speed. However, we believe that impact in performance is not very significant, as preliminary evaluations of fast hop level show that it typically obtains shorter path lengths than eCAN-like mechanism, even after short up times.