Principles of Locality-Aware Networks for Locating Nearest Copies of Data

Ittai Abraham* and Dahlia Malkhi†

Introduction

Building self-maintaining overlay networks for locating information in a manner that exhibits locality-awareness is crucial for the viability of large internets. It means that costs are proportional to the actual distance of interacting parties, and in many cases, that load may be contained locally. This paper presents a step-by-step decomposition of several locality-aware networks, that support distributed content-based location services. It explains their common principles and their variations with simple and clear intuition on analysis. Section 2 describes a novel technique for robustifying locality-aware overlays.

Problem statement. This paper considers the problem of forming a self-organizing, self-maintaining overlay network that locates objects (possibly replicated) placed in arbitrary network locations. Recent studies of scalable content exchange networks, e.g., [6], indicate that up to 80% of Internet searches could be satisfied by local hosts within one's own organization. Therefore, in order for the network to remain viable, it is crucial to consider locality awareness from the outset when designing scalable, decentralized network tools.

More formally, consider that the network constitutes a metric space, with a cost function $c(x, y)$ denoting the "distance" from x to y. Let $s = x_0, x_1, ..., x_k = t$ be the path taken by the search from a source node s to the object residing on a target node t. The main design goal to achieve is constant *stretch*. Namely, that the ratio $\frac{c(x_0,x_1)+...+c(x_{k-1},x_k)}{c(s,t)}$ is bounded by a (small) constant. Another important goal is to keep node degree low, so as to prevent costly reconfigurations when nodes join and depart. Thus, trivial solutions that connect all nodes to each other are inherently precluded.

Bounded-stretch solutions The problem of forming overlay routing networks was considered by several recent works in the context of networks that are searchable for content. Many of the prevalent overlay networks were formed for routing search queries in peer-to-peer applications, and exhibit **no** locality awareness.

There are several known schemes that provide locality awareness, including [12, 13, 15, 9, 3]. All of these solutions borrow heavily from the PRR scheme [12], yet they vary significantly in their assumptions and properties. Some of these solutions are designed for a uniform density space [9]. Others work for a class of metrics space whose growth rate is bounded both from above and from below [12, 13, 15, 4], while others yet cope with an upper bound only on the growth rate [3]. There is also variability in the guarantee provided on the stretch: In [9], there is no bound on stretch (except the network diameter). In [12], the stretch is an expected constant, a rather large one which depends on the growth bound. And in [3], the stretch can be set arbitrarily small $(1 + \varepsilon)$. Diversity is manifested also in the node degree of the schemes.

A step-by-step deconstruction. This paper offers a deconstruction of the principles that underlie these locality-aware schemes step by step, and indicate how and where they differ. It demonstrates the principles of locality awareness in a simplistic, yet reasonable (see [4]) network model, namely, a network with *power law* latencies. In our belief, the simplicity and the intuitive analysis may lead to improved practical deployments of locality-aware schemes.

For clarity, our exposition describes the design of

*The Hebrew University of Jerusalem, Israel. ittaia@cs.huji.ac.il

†The Hebrew University of Jerusalem, Israel. dalia@cs.huji.ac.il

an N-node network. It should be clear however, that this network design is intended to be self-maintaining and incremental. In particular, it readily allows nodes to arrive and depart with no centralized control whatsoever.

Some additional issues, such as dynamic maintenance, are provided in an accompanying technical report [1].

1 Locality-aware solutions

Preliminaries. The set of nodes within distance r from x is denoted $N(x,r)$. We assume a network model with *power law latencies*, $|N(x,r)| = \Gamma r^2$, for some known constant Γ. For convenience, we define neighborhoods $A_k(x) = N(x,2^k)$, and the radius $a_k = 2^k$. Thus, we have that $|A_k(x)| = \Gamma 4^k$.

For the purpose of forming a routing structure among nodes, nodes need to have addresses and links. We refer to a *routing entity* of a node as a router, and say that the node *hosts* the router. Thus, each node u hosts an assembly of routers.

Each router $u.r$ has an identifier denoted $u.r.id$, and a level $u.r.level$. Identifiers are chosen uniformly at random. The radix for identifiers is selected for convenience to be 4. This is done so that a neighborhood of radius 2^k shall contain in expectation constant number of routers with a particular length-k identifier. Indeed, the probability of a finding a router with a specific level and a particular prefix of length k is $1/4^k$. According to our density assumption, a neighborhood of diameter 2^k has $\Gamma 4^k$ nodes. Hence, such a neighborhood contains in expectation Γ routers matching a length-k prefix.

Assume a network of N nodes, and let $M = \log_4 N$. Identifier strings are composed of M digits. The level is a number between 1 and M. A level-k router has links allowing it to 'fix' its k'th identifier digit. Routers are interconnected in a butterfly-like network, such that level-k routers are linked only to level-$(k+1)$ and level-$(k-1)$ routers.

Let d be a k-digit identifier. Denote $d[j]$ as the prefix of the j most-significant digits, and denote d_j as the j'th digit of d. A concatenation of two strings d, d' is denoted by $d||d'$.

Step 1: Geometric routing. The first step builds *geometric routing*, whose characteristic is that the routing steps toward a target increase geometrically in distance. This is achieved by having large flexibility in the choice of links at the beginning of a route, and narrowing it down as the route progresses. More specifically, each router r of level k has four neighbor links, denoted $L(b)$, $b \in \{0..3\}$. Each one of the links $L(b)$ is selected as the closest node within $C_b(r)$, where $C_b(r) = \{u \in V \mid \exists s, u.s.id[k] = v.id[k-1]||b, u.s.level = k+1\}$. The link $L(b)$ 'fixes' the k'th bit to b, namely, it connects to the closest node that has a level-$(k+1)$ router whose identifier matches $v.R_k[k-1]||b$.

Geometric routing alone yields a cost which is proportional to the network diameter. The designs in [9, 4] make use of it to bound their routing costs by the network diameter.

Step 2: Shadow routers. The next step is unique to the design of LAND in [3]. Its goal is to turn the expectation of geometric routing into a worst-case guarantee. This is done while increasing node degree only by a constant expected factor. The technique to achieve this is for nodes to *emulate* links that are missing in their close vicinity as *shadow nodes*. In this way, the choice of links **enforces** a distance upper bound on each stage of the route, rather than probabilistically maintaining it. If no suitable endpoint is found for a particular link, it is emulated by a shadow node.

The idea of bounding the distance of links is very simple: If a link does not exist within a certain desired distance, it is *emulated* as a shadow router. More precisely, for any level $1 \le k \le M$ let r be a level-k router hosted by node v (this could itself be a shadow router, as described below). For $b \in [0..3]$, if $C_b(v)$ contains no node within distance 2^k, then node v emulates a level-$(k+1)$ *shadow router* s that acts as the $v.r.L(b)$ endpoint. Router s's id is $s.id = v.r.id[k-1]||b$ and its level is $(k+1)$.

Since a shadow router also requires its own neighbor links, it may be that the j'th neighbor link of a shadow router s does not exist in $C_j(s)$ within distance 2^{k+1}. In such a case v also emulates a shadow router that acts as the $s.L(j)$ endpoint.

Emulation continues recursively until all links of all the shadow routers emulated by v are found (or until the limit of M levels is reached).

With shadow routers, we have a deterministic bound of 2^k on the k'th hop of a path, and a bound of $\sum_{i=1..k} 2^i = 2^{k+1}$ on the total distance of a k-hop path.

A different concern we have now is that a node might need to emulate many shadow routers, thus increasing the node degree. Using a standard argument

on branching processes, we may obtain that hosting show routers increases a nodes degree only by an expected constant factor.

Shadow emulation of nodes is employed in LAND [3]. In all other algorithms, e.g., [12, 13, 15], a node's out-degree is a priori set so that the stretch bound holds with high probability (but is not guaranteed). Hence, there is a subtle tradeoff between guaranteed out-degree and guaranteed stretch. We believe that it is better to design networks whose outliers are in terms of out-degree than in terms of stretch. Additionally, fixing a deterministic upper bound on link distances results in a simpler analysis than working with links whose *expected* distance is bounded.

Step 3: Publish links. The final step in our deconstruction describes how to bring down routing costs from being proportional to the network diameter (which could be rather large) to being related directly to the actual distance of the target. This is done via a technique suggested by Plaxton et al. in [12], that makes use of short-cut links that increase the node degree by a constant factor. With a careful choice of the short-cut links, as suggested by Abraham et al. in [3], this guarantees an optimal stretch.

The technique that guarantees a constant stretch is to 'publish' references to an object in a slightly bigger neighborhood than the regular links distance. The intuition on how to determine the size of the enlarged publishing-neighborhood is as follows. The route that locates *obj* on t from s starts with the source s, and hops through nodes $x_1 \dots x_k$ until a *reference* to *obj* is found on x_k. The length of the route from s to x_k is bounded by a_{k+1}. The distance from x_k to t is bounded (by the triange inequality) by $a_{k+1} + c(s,t)$. In order to achieve a stretch bound close to 1, we should therefore guarantee that a reference to *obj* is found on x_k, where a_k is proportional to $\varepsilon c(s,t)$. This will yield a total route distance proportional to $(1+\varepsilon)c(s,t)$.

Therefore, by selecting the range of publish links from to cover x_k, the stretch of any search path is bounded by $1+\varepsilon$. The total number of outgoing links per node increases only by an expected constant factor.

The increased neighborhood for publishing provides a tradeoff between out-degree and stretch. Setting it large, so as to provide an optimal stretch bound, is unique to the design of LAND [3]. The designs in [12, 13, 15] fix the size of publish neighborhoods indepedently of the network density growth. This yields a stretch bound that depends on the density growth rate of the network.

2 Solutions that are both Locality aware and Robust

Pervious lookup solutions achieved either fault tolerance [7, 11, 14] or provably good locality properties [12, 3] but not both. In this section we sketch a lookup network that has, with high probability, low stretch even in the presence of a failure model, where all nodes may have a constant probability of failure.

In terms of fault tolerance, the main drawback of PRR [12] like networks is in their *routing flexibility*. In [5] it is shown that while hypercube and ring geometries have about $(\log n)!$ different routes from a given source to a given target, PRR like networks have only one! Thus the basic architecture of [12, 15, 8, 13, 3] is fragile and must be augmented with some form of robustness.

The first overlay network that has both a provable low latency for paths and a high fault tolerance was presented by the authors in FTLAND [2]. FTLAND achieves the combination of these two properties by augmenting the basic LAND architecture of Abraham et al. [3] with novel, locality-aware fault tolerance techniques. The techniques are based on the goal of dramatically increasing the routing flexibility to $(\log n)^{\log n}$ while still maintaining a provably good proximity selection mechanism.

Technical approach. In order to have fault tolerance, a node must increase the number of outgoing links it may use for routing. Doing so naively, e.g., as in [10, 11, 7], by simply replicating each link to $\log n$ suitable destinations instead of one, compromises locality. More specifically, in PRR-like networks, hops have geometrically increasing distances. It is imperative that the i'th hop has distance at most a^i (where a denotes a base that is typically a parameter of the construction). However, if the closest link happens to be down and a replacement link is used, there is no guarantee on the distance, and locality is lost.

In FTLAND, every node hosts $O(\log n)$ routers (instead of one) at each level. As before, routers are interconnected in a butterfly-style, where level-k routers have outgoing links only to level-$(k+1)$ routers. However, each router has w.h.p. $O(\log n)$ outgoing links (instead of an expected constant) for each desired destination. Since each node in the net-

work has $\log n$ routers at each level, whose identifiers are independently and uniformly selected, a router finds all $O(\log n)$ replicated destinations at a distance no greater than the distance to the closest router in the LAND scheme. Hence, locality is preserved when using any of these links. The total number of links increases by a poly-log factor (for each of the $O(logn)$ levels there are $O(\log n)$ routers in place of one, each of which has $O(\log n)$ replicated links w.h.p.).

Routing over the $(\log n)^{\log n}$ possibilities is done deterministically, with no backtracking. At each hop, one live link is followed, and with high probability, it can lead to the target.

Dealing with failures can be done in a very lazy manner, since the network can maintain a successful, locality-aware service in face of a linear fraction of unavailabilities. This property is crucial for coping well with churn, as a sustained quality of service is guaranteed through transitions. It also serves well to cope with transient disconnections and temporary failures, since there is no need for the network to reconfigure itself in response to small changes.

In [2], the following is proven:

Theorem 1 *There exists a scheme that requires $O(\log^3 n)$ links and w.h.p routes on paths of stretch $1+\varepsilon$ even if every node in the network has a constant probability of failure.*

References

[1] I. Abraham and D. Malkhi. Principles of locality-aware networks for locating nearest copies of data. Technical Report Leibnitz Center TR 2003-84, School of Computer Science and Engineering, The Hebrew University, 2003.

[2] I. Abraham and D. Malkhi. A robust low stretch lookup network. Technical Report Leibnitz Center TR 2003, School of Computer Science and Engineering, The Hebrew University, 2004.

[3] I. Abraham, D. Malkhi, and O. Dobzinski. LAND: Stretch $(1+\varepsilon)$ locality aware networks for DHTs. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA04)*, 2004.

[4] A. Goal, H. Zhang, and R. Govindan. Incrementally improving lookup latency in distributed hash table systems. In *ACM Sigmetrics*, 2003.

[5] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica. The impact of DHT routing geometry on resilience and proximity. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 381–394. ACM Press, 2003.

[6] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 314–329. ACM Press, 2003.

[7] K. Hildrum and J. Kubiatowicz. Asymptotically efficient approaches to fault-tolerance in peer-to-peer networks. In *Proceedings of the 17th International Symposium on DIStributed Computing (DISC 2003)*, 2003.

[8] K. Hildrum, J. D. Kubiatowicz, S. Rao, and B. Y. Zhao. Distributed object location in a dynamic network. In *Proceedings of the Fourteenth ACM Symposium on Parallel Algorithms and Architectures*, pages 41–52, Aug 2002.

[9] X. Li and C. G. Plaxton. On name resolution in peer-to-peer networks. In *Proceedings of the 2nd ACM Worskhop on Principles of Mobile Commerce (POMC)*, pages 82–89, October 2002.

[10] D. Malkhi, M. Naor, and D. Ratajczak. Viceroy: A scalable and dynamic emulation of the butterfly. In *Proceedings of the 21st ACM Symposium on Principles of Distributed Computing (PODC '02)*, pages 183–192, 2002.

[11] M. Naor and U. Wieder. A simple fault tolerant distributed hash table. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03)*, 2003.

[12] C. Plaxton, R. Rajaraman, and A. Richa. Accessing nearby copies of replicated objects in a distributed environment. In *Proceedings of the Ninth Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA 97)*, pages 311–320, 1997.

[13] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 329–350, 2001.

[14] J. Saia, A. Fiat, S. Gribble, A. R. Karlin, and S. Saroiu. Dynamically fault-tolerant content addressable networks. In *Proceedings of the First International Workshop on Peer-to-Peer Systems*, 2002.

[15] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. Kubiatowicz. Tapestry: A resilient global-scale overlay for service deployment. *IEEE Journal on Selected Areas in Communications*, 2003.