Locality-Aware Network Solutions
(A survey)

Dahlia Malkhi
The Hebrew University of Jerusalem

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 survey paper describes 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.

Problem statement. This survey 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., [24], 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, given are $n$ nodes and a relative distance function $c$ between pairs of nodes. There are several known variants of searching/routing problems in this settings:

Labeled routing: Route from any node to any other node using node-names designated by the designer of the routing scheme. This enables to present any node in the network with a packet destined by a virtual node name, determined according to the routing algorithm. The nodes should forward the packet among them until it reaches its destination.

Name-independent routing: Similar to labeled routing, the goal is to route from any node to any other. The difference is that the target node is named by its real network name, over which the routing protocol designer has no control. More specifically, in this problem nodes in the network may be presented with a packet destined to another node’s original name, and the scheme should forward it to the target node.

Resource location: In addition to $n$ nodes, the network contains $m$ objects residing on nodes arbitrarily and independently of the objects names or contents. There may be more than a single replica of each object. The goal of lookup here is to locate the nearest node containing a desired object. In a resource location scheme, each node may be presented with a packet destined at a particular object, and the routing scheme needs to find the closest node containing it.
Of course, any of these problems is trivialized by global flooding of the network. Hence, the solutions we seek must be communication-efficient. We desire that the routing path from source to destination have length proportional to the actual distance between them. More precisely, let $x$ be the origin point and $y$ be the target node. Let $x = x_1, x_2, \ldots, x_k = y$ be the nodes traversed by the routing or resource lookup scheme. Then the stretch is defined to be the ratio $\frac{c(x_1, x_2) + \ldots + c(x_{k-1}, x_k)}{c(x,y)}$.

We aim at algorithms with optimal worst case stretch, namely, as close to stretch-1 as desired.

On the other hand, these problems are also trivially solved with full information, i.e., if every node contains up-to-date data on the location and best routes toward all other nodes/objects. Such global information is costly not only in terms of storage, but also in terms of maintainance in face of dynamism. Our focus is on compact solutions, is which only partial information may be kept by each node. Driven by the scale and dynamism of today’s networks, we strive to allow each node to keep low, e.g., logarithmic, amount of data in terms of $n$. Though this is our ultimate goal, we consider any solution which requires nodes to maintain only $o(n)$ information compact.

1 Compact routing on Growth Bounded Networks

The first model for which this problem is solvable structures the network model as a metric space, and assumes that the densities of nodes in different parts of the network are not terribly different. More formally, there is a bound on the rate of density growth from one neighborhood, to a larger, encompassing one. There are indeed experiments, e.g., [23] that indicate that the Internet may be a power-law network, in which the growth bound is simple.

Until recently, the main known result is the seminal work of Plaxton et al. in [44], on which a number of systems are based, e.g., Tapestry [56] and Pastry [48]. The PRR method guarantees expected constant stretch. The actual constant is rather large, and is really mostly of theoretical interest. However, the approximate deployments of this algorithm in Tapestry [56] and Pastry [48] report encouraging performance results. The LAND scheme [5] improves on the PRR scheme, achieving a guaranteed low stretch, as close to optimal as desired. It does so with essentially the same node degree as the PRR scheme.

In the remainder of this section, we explain the principles that underlie the PRR-like methods, and a comparison of their properties.

1.1 Principles of locality-aware schemes for finding nearest copies of objects

The family of solutions considered in the section are designed for a restricted class of metric spaces whose density rate change is bounded: For all nodes $x$ in the network, and any radius $r$, a growth-rate bound $\Delta$ requires that the number of nodes inside the ball $B(x, 2r)$ (whose center is $x$ and has radius $2r$) is at most $\Delta |B(x, r)|$; a shrink rate bound $\delta$ indicates that $\delta |B(x, r)| \leq |B(x, 2r)| \leq \Delta |B(x, r)|$. Solutions for this class of networks, e.g., [44, 48, 56, 5], achieve a highly compact solution, in which each node stores $O(\log^2 n)$ routing information, and headers are of size $O(\log n)$.

All of these solutions borrow heavily from the PRR scheme [44], yet they vary significantly in their assumptions and properties. Some work for a class of metrics space whose growth rate is bounded both from above and from below [44, 48, 56, 23], while others yet cope with an upper bound only on the growth rate [5]. There is also variability in the guarantee provided on the stretch: In [44], the stretch is an expected constant, a rather large one which depends on the growth bound. And in [5], 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. We now offer a deconstruction of the principles that underlie these locality-aware schemes step by step, and indicate how and where they differ. We demonstrate the principles of locality awareness in a simplistic, yet reasonable (see [23]) 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 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.

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 \( \Gamma \). 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.

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 \( u.r \) 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 [38, 23] 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 [5]. 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 $1 \leq k \leq M$ 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 [5]. In all other algorithms, e.g., [44, 48, 56], 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 [44], 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 [5], 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 $x_1 \ldots x_k$ 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 triangle 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 [5]. The designs in [44, 48, 56] 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 Compact routing on Euclidean Metrics

The second network model for which we have locality-aware solutions is a geometric space. This model is very promising for two reasons. First, in certain networks (e.g., mobile networks) it is reasonable to assume that devices have geo-positioning devices. Secondly, there are several recent works that successfully embed Internet hosts in a 3-dimensional Euclidean space, such that the distance between nodes reflects fairly accurately their real distance in the network.

Embracing this model is a promising direction, which opens new exciting opportunities. By utilizing geometric coordinates, not only can the distances between nodes be predicted accurately and efficiently, but in addition, the properties of the Euclidean space may be used. Most importantly, Euclidean spaces have a ‘sense of direction’ which allows to perform distance-preserving routing while maintaining a very low number of links. Thus, we believe that this may dramatically improve the design of overlay networks for various real-life systems.

An essential requirement for such networks is a service that can establish communication sessions between (potentially mobile) nodes whose location is unknown. That is, we need a distributed algorithm that allows any source node \( s \) to route messages to any destination node \( t \), simply by knowing \( t \)'s network identifier and without the need to know \( t \)'s current location.

In the remainder of this section, after stating the problem more formally, we review results in this domain, and introduce two recent works of ours:

- A locality aware object location mechanism called LLS with logarithmic storage overhead by Abraham et al. in [2]. In LLS, costs are distance proportional both in the case of a location query, and in the case of an object move.

Geometric networks We study the problem of designing a communication network and a compact routing scheme for two dimensional Euclidean metrics \(^1\). Given is a set \( V \) of \( n \) nodes situated on a two-dimensional plane. Each node \( v \in V \) is defined by its coordinates \( \langle x(v), y(v) \rangle \). For any two nodes \( u, v \in V \), let \( \|uv\| \) denote the standard \( L_2 \)-norm distance, let \( D \) be the normalized diameter \( \frac{\max_{u,v}\|uv\|}{\min_{u,v}\|uv\|} \). The problem of compact routing on Euclidean metrics is a combined problem of locating target objects by names, and of designing a routing network and a routing scheme on top of the network.

2.1 Locality-aware location services

In the geometric network model each node knows its location on the Euclidean plane. Consider the following natural question; A source node \( s \) wants to initiate a communication session with a

---

\(^1\)We present the two dimensional case for clarity, all constructions can be easily extended to higher dimensional Euclidean metrics.
A survey of known solutions. A standard technique for name services in wireless cellular networks (e.g., ISA-4 [1], GSM MAP [41]) employs a home location register (HLR) for each mobile host. A publish algorithm stores the whereabouts of a node at its home location. The lookup first routes to the home location, and from there to the current destination of the node. Since the home may be further away from the source than the target, HLR are not locality aware by our definition. In addition, in a scalable settings nodes do not have global knowledge of the identities of other nodes in the network, and hence, defining a mapping that designated a home location is itself challenging.

An attractive solution for finding the home location is suggested in the context of sensor networks in the Geographic Hash Table (GHT) of [47]. In their approach, a home location is defined as a virtual coordinate rather than a node. They enhance the underlying routing to reach the closest node to the virtual point. A similar concept is employed in the GeoQuorums of [17], where geometric coordinates determine the location of home servers. In GeoQuorums, these focal point coordinates define geographic areas that must be inhabited by at least one server at any time. Still, the drawback of the home-based approach is that the cost of the lookup and of the publish may be arbitrarily high compared to the optimal path between source and destination.

In order to provide for better scalability and alleviate the problem of reaching specific location servers, several works suggest to replicate home location servers using quorum systems for availability and load balancing. Among these, the works of [45, 31, 26, 27] have no locality awareness. Other quorum based location services addressed locality in a partial way.

One of the early locality-aware location services that employs a hierarchy of partitions is provided for the general problem of object location in graphs ([12]). Their solution does not make use of geometric coordinates, nor address dynamism. Consequently, their solution is not easily adaptable to dynamic settings. In addition, their locality factors are somewhat large (polylogarithmic).

The approach taken by several works, e.g., in [51, 42, 13, 52], for quorum construction makes use of the planar structure of the network. It defines a write quorum for updating location information of a node as a column of some choice trajectory, and potentially some choice thickness. Similarly, a read quorum for querying location information is a row (of a choice trajectory and thickness).
Figure 1: Example of source s and destination t that are arbitrarily close to each other, but the smallest square that contains both of them is arbitrarily large

Trajectories are determined such that in average density networks, read and write quorums are likely to intersect. This method has good average case locality for lookup, but its publish cost is always the full diameter of the network, thus not proportional to the size of the movement. In addition, there are extreme cases in which read and write quorums might not intersect.

The position based multi-zone routing method of Amouris et al. [6] stores location information about each node in geometrically increasing discs, each disc referencing the smaller disc that contains the node. When a node moves a distance $2^i$, it broadcasts an update about the change to an area of radius $2^{i+1}$. Thus, both lookup and publish have locality awareness. The drawback of the scheme is that within a $2^i$ zone, location update is flooded to all nodes. This implies that each node in the network needs to maintain information (albeit not accurate) about every other node.

One of the pioneering works on efficient and scalable location services is by Li et al. in [37]. Similarly to the multi-zone method of [6], GLS utilizes a hierarchy of exponentially decreasing sets of regions (GLS uses squares rather than discs) that cover the plane. Every node belongs to only $\log M$ squares (were $M$ is the diameter of the network). Using ingenious techniques drawn from the consistent hashing approach [29], every node has a designated hashed location server within each square, thus distributing the load of location services across the network. The path taken by a GLS lookup operation is bounded inside the minimal square that contains both the source and the destination.

Yet GLS does not achieve either of our goals, and supports neither locality aware lookup, nor locality aware publish. There are several reasons for that, none of which is trivial to fix. First, the minimal square containing a particular source and destination may be arbitrarily large (see Figure 1). As a consequence, even in average case networks, the cost of lookup is not constant bounded.

Second, within each square GLS routes to a location server in order to find the target. In extreme network conditions, ad hoc routing to the location server, even if close by, could be much more costly than routing to the destination. Thus, in the worst case lookup cost could be arbitrarily higher than the optimal.

Third, their scheme makes little effort to proactively handle updates and out of date information. This problem arises when, for instance, a node crosses a grid boundary line. The authors state that a remaining open question is improving the handling of node mobility.
A totally different approach focusing on worst case analysis is discussed in the Conclusion section of [35]. The authors describe an algorithm that we name the Iterative Bounded Flooding (IBF) algorithm. This algorithm runs in phases beginning with phase 1 and incrementing the phase by one until the destination is found. At phase $i$, the algorithm floods the network to all nodes whose minimal cost path from the source is at most $2^i$. IBF is asymptotically worst case optimal. Specifically, if the minimal cost of the path from source to destination is $d$ then IBF guarantees to reach the destination in cost $O(d^2)$. The main drawback of this approach is that its average cost is also $\Omega(d^2)$.

**New Results.** We recently present in [2] a Locality aware Location Service named LLS. Our location service is the first location service which is both worst case optimal and average case efficient. For worst case networks, our lookup is asymptotically optimal, incurring a lookup cost of $O(d^2)$ for a source and a destination whose minimal cost path has length $d$.

For average case networks in which ad hoc routing costs $\Delta$ times the distance from source to destination, LLS achieves in addition an average case linear cost over the distance, $O(d)$. Thus, in average case networks, where the routing cost is proportional to the distance from source to destination, lookup in LLS costs only a constant factor more than the distance between the source and the destination.

Our scheme is also the first to provide guaranteed average case efficient publishing. That is, our service ensures that the expected cost of updating the data structures due to a node’s movement is bounded as a function of the distance of the movement. Specifically, when a node moves distance $d$, the average cost of publishing its new location is $O(d \log d)$.

### 2.2 Geometric ad hoc routing.

Geometric routing was also studied in the context of mobile ad hoc networks (MANETs) that are enhanced with self-positioning devices such as GPSs. The model here is somewhat different than ours. It assumes that each node has a certain transmission range, and is linked directly to all nodes within this range (Unit Disk Graph). The first routing algorithm to guarantee delivery is *face routing*, due to Kranakis et al. [34]. However, face routing has no bound on the ratio between the cost of route and the minimal cost path. Both Bose et al. [14] (CGF) and Karp and Kung [30] (GPSR) propose an algorithm that combines greedy routing with face routing. In the MANET model, these algorithms guarantee delivery and for average case networks have expected cost $O(d)$ between a source $s$ and a destination $t$, where $d$ is the cost of minimal-cost path between $s$ and $t$ on the unit disk graph. The first algorithm that gives worst case guarantees is by Kuhn et al. [35]. They present a scheme in which, if the minimal cost path has cost $d$, then delivery with cost $O(d^2)$ is guaranteed, which is asymptotically optimal. In a follow up paper [36], they combine their bounded face routing with greedy routing to achieve a scheme that is both worst case asymptotically optimal and average case efficient. Due to the MANET model, all of the above algorithms have worst case diameter $\Omega(n)$.

As for the general problem of routing in the plane, it has two components.

- **Network design.** Every node $u$ must choose a set of out going neighbors. This induces a directed graph $H$ on the set $V$.

- **Routing scheme.** A routing scheme, $RS$, on the graph $H$. The scheme allows any source node $s$ that knows a target $t \in V$ and its coordinates $(x(t), y(t))$, to route from $s$ to $t$. 


The problem of compact routing using geometric coordinates has been considered in a number of previous works. By simply linking each node to its immediate neighbor in every angle, say $\theta$, one obtains a constant stretch $\theta$-spanner as in [32], whose degree is constant but the number of hops may be $\Omega(n)$.

Combining constant stretch and low degree with low diameter is not trivial, and progress has been slow. Recent work by Hassin and Peleg [28] presents important progress. Their algorithm achieves the following measures. The stretch is $1+\varepsilon$, memory is $O(\log D)$, and diameter is $O(\log D)$.

Our recent work [2] improves their solution by bringing down the node degree (memory) to a constant. Hassin and Peleg present an alternate construction that has $O(\log n)$ diameter instead of $O(\log D)$. This construction has better asymptotic diameter when $D = \Omega(2^n)$, but in such a case simply storing the coordinates of a node requires $\Omega(n)$ bits which makes the whole scheme non-compact.

Applicability. Using geometric coordinates in general internets is made relevant not only by the ubiquity of GPS devices, but also by several recent techniques that embed internet nodes in a coordinate space. One of the pioneering mechanisms to predict network latency is based on the work of Ng and Zhang [43]. They embed the Internet latencies into a virtual geometric space (e.g., 3-D Euclidean) and characterize the position of any node with coordinates. The computed distances are used to predict the actual network distances. Following [43] other schemes have been developed to improve the embedding of internet hosts into virtual geometric spaces, e.g., [25], [55], [16], and [49].

Geometric routing is also relevant to an on-going effort in designing geometric routing networks for peer-to-peer (p2p) applications, based on routing in “small worlds” [33]. The goal in this domain is for a dynamic set of nodes to jointly implement a shared data structure, such as a hash table. In order to realize a shared structure distributively, operations on data are routed among the nodes in order to dispatch where the data resides. The p2p works consider nodes dispersed uniformly on a Euclidean space (either real or virtual) of one or two dimensions and route in a distance-preserving manner. Chord [50] uses $O(\log n)$ links per node, and achieves an expected diameter $O(\log n)$. Symphony [40] uses a Kleinberg-like link distribution, achieving an expected diameter of $O(\log^2 n/k)$ with $k$ links per node. The same complexity was achieved in [8] with a different Kleinberg-style randomized p2p network. Viceroy [39] achieves $O(\log n)$ diameter with 5 links.

Using our recent network design [2] with nodes that are dispersed uniformly on a uni-dimensional space matches the best complexity measures so far, i.e., of Viceroy. In this respect, our work extends all known p2p overlay network constructions into an arbitrary density space, while preserving locality, constant node degree, and logarithmic diameter.

3 Compact Routing on General graphs

The above solutions treat certain restricted network models. There are several limitations with the solutions known solutions so far. First, there are many situations in the Internet that break the triangle inequality. Second, it is rather silly to assume that nodes at the very endpoints such as

\footnote{
It is only fair to note that the importance of Kleinberg’s work lies completely outside the area of network design, and its focus is on modelling real-life small-world networks. Our work does not attempt to add to the understanding of social networks.
}
users over low-capacity ADSL links, communicate directly with one another. To the contrary, it would be beneficial, at least when considering the last mile, to consider the network is a partial graph. Third, networks of logarithmic path length are sensitive to node corruption: A linear fraction may hurt virtually all communication paths among pair of honest participants, and thus mount a DoS attack (potentially invisibly). In order to tolerate malicious nodes, there needs to be a higher redundancy in communication paths, and they must be short.

From all of the above reasons, we move in this section to model the network as a general, weighted undirected (partial) graph. We briefly survey the known solutions for general graphs.

For the three variants of routing/search problems above, the known lower bound in effect for general graphs is that of Gavoille and Gengler [21], indicating at least stretch-3 when each node has memory $o(n)$. For comprehensive surveys on compact routing and compact network data structures, see [20, 22].

Optimal stretch-3 compact schemes for labeled routing are already known. The first stretch 3 scheme was given by Cowen [15] with $\tilde{O}(n^{2/3})$ memory. Later, Thorup and Zwick [53, 54] improved the memory bound to only $O(\sqrt{n})$ bits. They also gave an elegant generalization of their scheme, achieving stretch $4k - 5$ (and even $2k - 1$ with handshaking) using only $O(n^{1/k})$ bits. Additionally, there exist various labeled routing schemes suitable only for certain restricted forms of graphs. For example, routing in a tree is explored, e.g., in [18, 54], achieving optimal routing. This routing requires $\tilde{O}(1)$ bits for local tables and $\tilde{O}(1)$ bits for headers, and this is tight [19].

As for name independent routing, progress has been much slower. Initial results in [10] provide non-compact name independent routing with $\tilde{O}(n^{3/2})$ total memory. Awerbuch and Peleg [11] were the first to show that constant-stretch is possible to achieve with $o(n)$ memory per node, albeit with a large constant. Recently, Arias et al. significantly reduced the stretch factor in [7], providing stretch-5 with $\tilde{O}(\sqrt{n})$ memory per node.

In [3], Abraham et al. close the gap and achieve stretch 3 for general graphs also with $\tilde{O}(\sqrt{n})$ memory per node. Besides improving Arias et al. [7] stretch from 5 to 3, these results answer affirmatively the challenge of optimal name independent routing that was open since the initial statement of the problem in 1989 [10]. Surprisingly, the results show that allowing the designer to label the nodes does not improve the stretch factor compared to the task when node labels are predetermined by an adversary.

It is worth noting why name independence is desirable. In many cases it is unacceptable to have the node names precomputed in a centralized manner by the network designer. One practical motivation for name independence is given by Awerbuch et al. in [9]. They indicate that name dependent solutions are less appropriate for dynamic settings, in which nodes may join and depart the network and must be assigned long-term identifiers independent of their network links.

Another salient motivation arises from p2p search tools, and the construction given by Karger et al. [29] for scalable object location services. In their consistent hashing approach, object names are hashed to yield an arbitrary string, and a reference to the object needs to be kept at the node that has the corresponding name. An overlay routing infrastructure is used in this setting for locating objects based on their hashed names. This type of solution called a Distributed Hash Table (DHT) is the focus of enormous recent interest [46, 50, 48, 56]. In this setting, there is no relationship between hashed object identifiers and their reference node’s location. For load balancing, the space of hashed identifiers must be independent of the network topology.

We may build a DHT which builds upon the underlying name-independent routing overlay, and

\[ 3 \text{ The notation } \tilde{O}() \text{ indicates the same complexity as } O(), \text{ respectively, up to polylogarithmic factors.} \]
enjoys the characteristics of the routing scheme. And with our recent results, we obtain a DHT scheme that keeps $\tilde{O}(\sqrt{n})$ routing information per node, and finds reference information in the DHT with stretch 3. In networks that form a metric space, searching for information takes only two hops.

More specifically, let $A$ denote the universe of potential objects. Let $H()$ be a hash function that uniformly maps object names to the range $[1..n]$. For every $1 \leq i \leq n$, node $i$ is responsible for storing reference information on all objects whose names hash to the value $i$. In order to query or to update information about an object $a \in A$, e.g., when it is created, removed, searched or moved, a message with the relevant information is sent addressed to $H(a)$. This message is routed using name-independent routing scheme to the node whose identity is $H(a)$. 
References


