HDLBR: A name-independent compact routing scheme for power-law networks

Mingdong Tang a,⁎, Guoqiang Zhang b,c,1, Tao Lin d, Jianxun Liu a

⁎ Key Lab of Knowledge Processing and Networked Manufacturing, Hunan University of Science and Technology, Xiangtan, China
b School of Computer Science and Technology, Nanjing Normal University, Nanjing, China
c Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
d High Performance Network Lab, Institute of Acoustics, Chinese Academy of Sciences, Beijing, China

Abstract

Compact routing intends to achieve a good tradeoff between routing path length and storage overhead, and is recently considered as a main alternative to overcome the fundamental scaling limitations of the Internet routing system. It is generally believed that specialized compact routing schemes for peculiar network topologies have better average performance than universal ones, and name-independent schemes are more flexible due to their natural support of the Locator/ID split principle. In this paper, we propose the highest degree landmark based routing scheme (HDLBR), which is the first specialized name-independent compact routing scheme for Internet-like power-law networks. HDLBR optimizes routing performance by selecting a few nodes with high degrees as the landmarks and making these nodes to be the entrances for mapping topologically agnostic node names to topology-aware node addresses. Simulation results show that HDLBR has very small average routing table size, in the order of O(n1/2) bits per node on both synthesized power-law graphs and the real AS graphs. Meanwhile, the average stretch of the HDLBR scheme is comparable with the base Abraham scheme using random coloring, and is only slightly outperformed by the customized Abraham scheme which also takes into account the degree heterogeneity of power-law networks.

Keywords: Compact routing, Power-law, Name-independent, Stretch, Routing

1. Introduction

Routing is the basic functionality of communication networks. Performance of a routing scheme is often evaluated by three metrics: path length experienced by packets, space needed to store the auxiliary routing information, and size of the packet headers. Traditional shortest path routing guarantees that paths are shortest between any node pairs, but requires each node to store routing information to all other nodes, which means that the number of routing entries will grow linearly as a function of the network size. Consequently, this approach is by no means scalable, which has already been evidenced by the inter-domain BGP routing protocol, a protocol that combines the shortest path routing and policy based routing. In recent years, the routing tables of the default zone (DFZ) BGP routers increase exponentially, severely stress the routers, and heavily restrict further growth of the Internet [1–5].

Devising scalable routing infrastructure is thus necessary to mitigate the challenge faced by present Internet routing. Compact routing is recently considered as a main alternative to overcome the fundamental scaling limitations of the Internet routing system. The basic idea of compact routing is to balance between routing table size and path length. A compact routing scheme guarantees that the routing table imposed on each node is o(n), and in compensation, relaxes the shortest path requirement. The quality of the path is quantitatively measured by stretch. The most general stretch representation is in the form of a two tuple (x, β) [9–11,32], which means for any i and j, the following inequality holds:

\[ d^x(i,j) \leq xd(i,j) + \beta. \]

where \( d^x(i,j) \) is the path length between node i and j taken by the routing scheme R, and \( d(i,j) \) is the shortest path length between these two nodes. When \( \beta = 0 \), this representation reverts to the multiplicative stretch x. When \( x = 1 \), R is said to have additive stretch \( \beta \). Unlike stretch, which measures the worst-case upper bound path quality, the average stretch measures the average path quality of a routing scheme R, which is defined as:

\[ \text{averagesretch} = \frac{\sum_{x=1}^{n} d^x(i,j)}{n(n-1)}. \]

where n is the number of nodes.

Compact routing schemes on generic network topologies (also termed as universal schemes) [8,12–14] and specific networks such as trees [17,19], planar graphs [21], growth bounded graphs [18] and graphs with low doubling dimension [22,20] have...
Attracted wide research attentions. Presently, the most optimized universal compact routing schemes, either name-dependent or name-independent, can restrict the routing table size on each node to $O(n^{1/2})$ bits, where $O(n)$ is a short notation for $O(n \log n)$ for some constant $c$, and meanwhile, achieve multiplicative stretch of 3, i.e., the length of the path between any node pair taken by the routing is no more than 3 times the length of the shortest path between the node pair.

Although multiplicative stretch of 3 has been proven to be the optimal worst case lower bound for all compact routing schemes on generic networks, it is generally believed that routing schemes specialized for peculiar network topologies can achieve better performance than universal ones [17–22,34,42]. One pervasive topological property of large-scale real networks revealed in the past decade is the power-law degree distribution. As a result, compact routing on power-law networks have attracted increasingly more attentions in recent years [31–34,40,42].

However, so far, all these specialized routing schemes for power-law graphs are name-dependent, which means node names encode topologically sensitive information useful for routing and hence should be renamed upon topology changes. In contrast name-independent schemes do not encode topology-dependent information in the node names, hence are more flexible in coping with topology changes. Motivated by these, we propose a name-independent compact routing scheme that is specialized for Internet-like power-law networks. More specifically, our main contributions are:

1. We propose a specialized name-independent compact routing scheme (named HDLBR) for Internet-like power-law networks;
2. We give analytical evaluations for the stretch and routing table size of HDLBR;
3. We show by simulation that HDLBR produces average routing table size on each node in the order of $O(n^{1/2})$ bits, which is smaller than the two realizations of Abraham schemes mentioned in our paper. The average stretch produced by HDLBR is well below 1.2 for the power-law networks used in our simulation, comparable to the base Abraham scheme, and about 10% larger than the customized Abraham scheme. Hence, HDLBR achieves good tradeoff between average routing table size and average stretch on power-law graphs.

2. Related work

Since 1990s, several name-dependent compact routing schemes are proposed for generic networks [6–8,12], among which the Cowen scheme [8] and the TZ scheme [12] are two prominently referenced ones. Cowen scheme is the first universal compact routing scheme that reduces the multiplicative stretch to 3, and produces sublinear routing table size whose upper bound is $O(n^{2/3})$ bits. TZ scheme improves the Cowen scheme's routing table size upper bound to $O(n^{1/2})$ bits. Both these two schemes are landmark-based and differ only in how landmarks are selected. Cowen scheme adopts a greedy algorithm to find the dominating set of the graph, and obtains the landmark set based on this dominating set, whereas the TZ scheme relies on random sampling techniques for landmark selection. Since Gavoille and Gengler [15] showed that no universal routing scheme with multiplicative stretch strictly less than 3 can produce $o(n)$ routing table size, and Thorup and Zwick [16] proved that any routing scheme with multiplicative stretch strictly below 5 cannot guarantee space smaller than $O(n^{1/2})$, it thus follows that the TZ scheme is nearly optimal as far as the stretch and routing table size are concerned.

A key limitation of name-dependent schemes is that nodes need to be reassigned whenever the topology changes. In contrast, name-independent schemes are more flexible in handling network dynamics. Arias [13] and Abraham [14] proposed two representative universal name-independent schemes. The Arias scheme's multiplicative stretch is 5, and the routing table size upper bound is $O(n^{1/2})$ bits. Whereas the Abraham scheme is the first nearly-optimal universal name-independent scheme that provides the best trade-off between routing path stretch, memory space, and header size, whose multiplicative stretch and routing table size upper bound are 3 and $O(n^{1/2})$ bits respectively. Abraham scheme uses $\sqrt{n}$ colors to color the nodes by a random coloring algorithm, ensuring that the nearest $O(\sqrt{n})$ nodes of any given node cover all the colors. The set of these nearest nodes is defined as the vicinity ball of the given node. By storing a complete copy of the name-to-address mapping within each node's vicinity ball, Abraham scheme achieves good balance between the path overhead incurred by the address lookup and the storage overhead of the mapping tables.

All the above schemes, either name-dependent or independent, are universal in the sense that they are designed for generic networks. However, recent studies revealed that most real networks, for example, the Internet, WWW and unstructured P2P networks, can all be categorized as power-law networks [23–30], i.e., the degree distribution $p(k)$ satisfies $p(k) \sim k^{-\gamma}$, where $\gamma$ is called the power-law exponent, typically between 2 and 3. It has been shown that on power-law networks, high degree nodes play the hub role for network connectivity and routing [35,36], most nodes are directly connected to these nodes or separated by only a few hops, and the path between these high degree nodes are typically very short [37]. So recently, compact routing schemes on power-law networks have attracted much attention. Krioukov [31] first evaluated the performance of the TZ scheme on Internet-like power-law graphs, and found that the average performance is far better than the worst case theoretical results, e.g., the average multiplicative stretch is only 1.1. Brady and Cowen [32] proposed a compact routing scheme with stretch $(1, d)$ and logarithmic routing table scaling, i.e., $O(\log^2 n)$. Simulation results showed that $d$ and $e$ can both take small values on power-law networks, and the average stretch is lower than the TZ scheme. In Ref. [34], the random power-law graph model is utilized to give an elegant analytical performance evaluation of the proposed name-dependent routing scheme adapted from the TZ scheme. Other specialized compact routing schemes on power-law networks can be found in recent works [33,41,40,42]. All these work showed that these schemes can achieve better space-stretch tradeoff than universal ones.

However, all the specialized routing schemes mentioned above are name-dependent. There are a few schemes that can be categorized to specialized name-independent schemes, but they are for trees [19], growth bounded networks [18] and networks with low doubling dimension [22,20]. Our proposed HDLBR in this paper is the first specialized name-independent routing scheme for power-law networks, which achieves good tradeoff between the average routing table size and average stretch on power-law graphs.

3. Assumptions and notation

This section defines the assumptions and notation used in this paper, e.g., network, routing scheme and compact routing.

A network is abstracted as a graph $G = (V, E)$, where $V$ is the set of nodes, and $E$ is the set of edges. We denote $|V| = n$ and $|E| = m$. Denote $d(u, v)$ as the shortest path length between node $u$ and $v$ in the network. In this paper, only unweighted graphs are considered.

A routing scheme $R$ is defined as a distributed algorithm on graph $G$ such that each node can base on the message header
and its routing table to correctly forward the message from an arbitrary source node to an arbitrary target node. A routing scheme is usually composed of two parts: the preprocessing procedure and the forwarding procedure. Denote \( d \text{out}(u, v) \) to be the path length from node \( u \) to \( v \) taken by routing scheme \( R \).

The quality of a routing scheme is primarily measured by three metrics: the message header length, the storage space of the distributed data structures established by the preprocessing procedure, typically referring to the size of routing table, and the routing path quality, typically measured by stretch. Other metrics include routing table convergence time and communication cost. In this paper, we only consider routing on static topologies, hence the convergence time and communication cost are out of our consideration.

Rigorously speaking, a routing scheme can be called compact only if: (a) it produces logarithmic addresses and header sizes, more formally, in the order of \( \Theta(\log n) \) bits for some constant \( c \); (b) the routing table size grows sublinearly with the network size, i.e., in the order of \( o(n) \); and (c) the stretch is bounded by a constant.

Rigorously speaking, a routing scheme can be called compact only if: (a) it produces logarithmic addresses and header sizes, more formally, in the order of \( \Theta(\log n) \) bits for some constant \( c \); (b) the routing table size grows sublinearly with the network size, i.e., in the order of \( o(n) \); and (c) the stretch is bounded by a constant.

4. HDLBR

According to Ref. [5], a name-independent compact routing scheme can be typically decomposed into two building blocks: (1) a name-dependent compact routing scheme operating underneath the name-independent one, and (b) dictionary tables forming an efficiently-distributed database containing information necessary to translate nodes' topology-independent flat identifiers to topology-dependent locators. The routing procedure then should first translate the target name into the target address, and then forward the packet to the target node by this topology-dependent address.

Compared with name-dependent routing schemes, name-independent ones are more complex and more resource consuming. It introduces an additional step of node address lookup, which can increase both the routing path length and storage overhead. Generally speaking, here is a tradeoff issue. In one extreme, if each node stores the whole dictionary table, then name-to-address translation can be done locally, but it incurs \( \Omega(n) \) storage overhead on each node; in the other extreme, if the dictionary table is uniformly partitioned into \( n \) slices and stored in \( n \) nodes in a distributed manner, then the storage overhead is negligible, but the name-to-address translation may incur long path overhead. Neither of the above two extremes is desirable. The translation should neither impose high storage overhead on any node, nor significantly exacerbate the routing path length.

Guided by the above idea, we introduce our design of the HDLBR. HDLBR builds upon a name-dependent routing scheme called TZ, which is a modification of the TZ scheme [12].

4.1. TZ: a name-dependent compact routing scheme

Similar to the TZ scheme, TZ is also a landmark based routing scheme. The basic idea of landmark based routing is to select a few nodes from the network as landmarks. The routing table of each node only stores routing entries for all the landmarks as well as nodes in its neighborhood. As a result, if the target node could be found in the source node's routing entries, then the packet can be delivered along the shortest path, otherwise, the packet is first forwarded to the nearest landmark of the target, and then be delivered to the target by the landmark. Unlike the TZ scheme which randomly selects landmarks, our proposed TZ scheme chooses high degree nodes as landmarks.

Let \( L \subseteq V \) represent the landmark set. For any \( u \in V \), define \( u \)'s ball \( B(u) \) to be \( B(u) = \{ v \in V | d(u, v) < d(u, L) \} \), where \( d(u, L) \) is the minimum path length among all the shortest paths from \( u \) to all the landmarks in \( L \), and define \( u \)'s cluster \( C(u) \) to be \( C(u) = \{ v \in V | u \in B(v) \} \). Table 1 illustrates these definitions.

As said before, a routing scheme consists of two procedures: the preprocessing procedure and the packet forwarding procedure. The former setups the data structures, e.g., assigning node addresses and building routing tables, whereas the latter forwards the packet according to the packet header and the routing table.

4.1.1. Preprocessing

The preprocessing procedure consists of the following steps:

1. **landmark selection**: sort the nodes in decreasing order of their degrees, and choose the first \( n^x \) nodes as the landmarks, where \( 0 < x < 1 \) is a tunable parameter;
2. **address assignment**: for each non-landmark \( u \in V - L \), assign \( u \)'s address \( e_{\text{addr}}(u) \) as its address, denoted as \( \text{Addr}_{\text{sys}}(u) \), where \( L(u) \) is the name of the nearest landmark to \( u \), and \( e_{\text{addr}}(u) \) is the port identifier at \( L(u) \) that lies on the shortest path from \( L(u) \) to \( u \); for each landmark, its address is identical to its name.
3. **routing table setup**: for each non-landmark \( u \in V - L \), its routing table is \( \text{Table}_{\text{TZ}}(u) = \{ (v, e_{\text{addr}}(v)) | v \in L \cup C(u) \cup B(u) \} \). In other words, \( u \) stores the routing entries to nodes in \( L, C(u), \) and \( B(u) \), where \( e_{\text{addr}}(v) \) is port identifier at \( u \) that lies on the shortest path from \( u \) to \( v \); for each landmark \( l \in L \), its routing table only contains the routing information to other landmarks, i.e., \( \text{Table}_{\text{TZ}}(l) = \{ (v, e_{\text{addr}}(v)) | v \in L \setminus \{l\} \} \).

For example, in Fig. 1, the routing tables of \( u \) and \( v \) contain routing entries for \( \{a, b, c, d, t\} \) and \( \{a, b, c, d, s, t\} \) respectively. Compared with the TZ scheme, TZ adds the additional routing entries of \( B(u) \) for any non-landmark node \( u \), as which we will show later decreases the stretch of HDLBR.

4.1.2. Packet forwarding

The packet forwarding procedure of \( \text{TZ} \) is similar to TZ. Assuming a packet with a destination address \( (v, L(v), e_{\text{addr}}(v)) \) arrives at node \( u \), \( u \) performs the following checking:

1. if \( u = v \), then the packet reaches its destination;
2. otherwise, if \( u = L(v) \), then forward the packet using the local port \( e_{\text{addr}}(v) \);
3. otherwise, if \( \text{Table}_{\text{TZ}}(u) \) contains the routing entry for \( v \), then forward the packet using the local port \( e_{\text{addr}}(v) \);
4. otherwise, find the routing entry of \( L(v) \) in \( \text{Table}_{\text{TZ}}(u) \), and forward the packet towards \( v \)'s nearest landmark \( L(v) \) using the local port \( e_{\text{addr}}(L(v)) \).

For example, in Fig. 1, when sending a packet from \( s \) to \( t \), the forwarding path would be \( s \rightarrow w \rightarrow b \rightarrow u \rightarrow t \) according to the above forwarding algorithm, but the shortest path between \( s \) and \( t \) is \( s \rightarrow v \rightarrow t \).

4.2. HDLBR

In the following, we describe the design of HDLBR in detail, which builds upon the TZ scheme.

**Table 1** Balls and clusters of the nodes \( s, t, u, v, w \) in the network specified by Fig. 1.

<table>
<thead>
<tr>
<th>( s )</th>
<th>( t )</th>
<th>( u )</th>
<th>( v )</th>
<th>( w )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( B(\cdot) )</td>
<td>( \emptyset )</td>
<td>( {u, v} )</td>
<td>( \emptyset )</td>
<td>( {s, t} )</td>
</tr>
<tr>
<td>( C(\cdot) )</td>
<td>( {v} )</td>
<td>( {v} )</td>
<td>( {t} )</td>
<td>( \emptyset )</td>
</tr>
</tbody>
</table>
4.2. Preprocessing

The preprocessing procedure of HDLBR is as follows:

1. The landmark selection step is similar to TZ; i.e., choosing the highest degree nodes as landmarks. In HDLBR, we set the parameter \( x = 1/2 \), i.e., \( n^{1/2} \) nodes with the highest degrees compose the landmark set \( L \).
2. The address assignment and routing table setup steps are the same as TZ. We use the same notations \( \text{Address}_{TZ}(v) \) and \( \text{Table}_{TZ}(u) \) to represent \( u \)'s address and routing table respectively.
3. Name-to-address dictionary table construction: distribute the dictionary table to all landmarks, and each node can lookup the address of any other node.

The key aspect of the preprocessing procedure is how to construct the name-to-address dictionary table. Similar to Ref. [14], we shall assume a balanced hash function \( H \) that maps the name of each node \( v \) to a landmark identified as \( H(v) \) in such a way that at most \( O(\sqrt{n \log n}) \) names map to the same landmark. Each node \( v \) can capitalize on this hash function to independently determine the landmark on which to store the mapping entry \( (v, \text{Address}_{TZ}(v)) \). Since each node maintains all the landmarks, if we order the landmarks by their names, then we can use the numbers 0, 1, 2, \ldots, \( n^{1/2} - 1 \) to identify the landmarks. Consequently, the output of the hash function \( H \) could be the numbers in \( \{0, 1, \ldots, n^{1/2} - 1 \} \). If we restrict the use of the numbers of 0, 1, 2, \ldots, \( n^{1/2} - 1 \) to identify the nodes, then the hash function \( H \) would be quite simple: assuming the node name is encoded in \( \log n \) bits, then we can extract \( \frac{\log n}{2} \) bits from the name as the output, e.g., the lowest \( \frac{\log n}{2} \) bits. Otherwise, if nodes are named by arbitrary strings, then we can borrow the idea from Ref. [14] to design the hash function \( H \), which is further described in the following.

For simplicity, assuming \( n^{1/2} \) is a power of 2, i.e., on the form of \( 2^k \), then \( \log n/2 \) is an integer. First, we use a standard universal hash function to map the node names to the numbers in 1, 2, \ldots, \( n^{1/2} \). With high probability, this mapping will not cause collision. Let \( q = \log n/2 \), then we need to map the 5q bits long short name to \( q \) bits long address. Here we borrow the idea proposed in Ref. [38]. For \( i = 1, 2, 3, 4 \), let \( T_i \) be a random table containing \( 2^i = n^{1/2} \) entries, that maps \( q \) bits to \( iq \) bits. Then for a 5q bits short name \( x \), the hash function works as follows. Initially, set \( x_4 = x \). For \( i = 4, 3, 2, 1 \), repeat the following two steps: (1) set \( y_i \) to be the most insignificant \( q \) bits of \( x_i \) and \( z_i \) to be the remaining \( iq \) bits of \( x_i \), and (2) set \( x_{i-1} = T_i[y_i]z_i \). In the end, \( x_0 \) only contains \( \log n/2 \) bits, which can be output as the landmark identifier. The above procedure is guaranteed to finish in constant time, and it can be proven that with high probability the number of nodes mapped to each landmark is bounded by \( O(n^{1/2}) \) [14].

4.2.2. Packet forwarding

The forwarding procedure of HDLBR is described as follows. Assuming a source node \( u \) needs to send a packet to a target node \( v \), then:

1. if \( v \) is in \( u \)'s routing table, \( u \) can forward the packet to \( v \) along the shortest path;
2. otherwise, \( u \) computes the identifier of the landmark \( H(v) \) on which \( v \)'s address is stored;
3. \( u \) sends the packet to \( H(v) \). According to the routing table construction policy, \( H(v) \) is in \( u \)'s routing table, so that the packet can reach \( H(v) \) along the shortest path. Notice that in this process, if any intermediate node \( w \) contains the routing information for \( v \), then \( w \) will forward the packet to \( v \) along the shortest path. Hence, it is possible that the packet may be delivered by a shortcut that bypasses \( H(v) \);
4. when the packet arrives at \( H(v) \), \( H(v) \) obtains \( v \)'s address \( (v, \text{Address}_{TZ}(v)) \), uses it to rewrite or encapsulate the packet header, and then forwards the packet to node \( v \) according to the packet forwarding algorithm of the TZ scheme.

5. Analytical performance analysis

In this section, we provide an analytic performance evaluation of HDLBR in terms of the packet header length, stretch and routing table size. When performing the evaluation, we borrowed the approach used in Ref. [34] (i.e., analyzing the performance based on the random power-law network model) and we relied our analysis on some proved results from the same paper.

5.1. Packet header length

The packet header length of HDLBR is bounded by \( O(\log n) \) bits, which arises from the following observations. For any node \( v \), its address is \( (v, L(v), \text{Address}_{TZ}(v)) \). Since the node name and port identifier can both be represented using \( \log n \) bits, so the address length is \( O(\log n) \). A packet of the HDLBR scheme includes either the target name or target address in its header, which implies \( O(\log n) \) bits packet header length.

5.2. Stretch

Unlike the multiplicative stretch widely used in the analysis of universal compact routing schemes, here we use the general stretch representation \((\alpha, \beta)\) for our analysis.

The stretch of the HDLBR scheme is formally stated in the following theorem.

**Theorem 1.** The stretch of HDLBR is \((2, 2d)\), where \( d = \max_{u,v} \forall v \in Ld(u,v) \) is the maximum distance between any two landmarks.

**Proof.** Suppose a source node \( u \) needs to send a packet to a target node \( v \). If \( v \in L \cup C(u) \cup B(u) \), then the packet can be routed to \( v \) along the shortest path by using \( v \)'s node name. Otherwise, \( u \) has to first use the hash function \( H \) to compute the landmark \( H(v) \) that is responsible for storing \((v, \text{Address}_{TZ}(v))\), and then routes the packet to \( H(v) \). \( H(v) \) then rewrites the packet header using \( v \)'s address \( \text{Address}_{TZ}(v) \), and forwards the packet to the target node according to the packet forwarding algorithm of TZ. This procedure can be depicted by Fig. 2.
Hence, the path length between $u$ and $v$ satisfies:
\[
d^\text{HDLBR}(u, v) \leq d(u, H(v)) + d(H(v), L(v)) + d(L(v), v)
\]
According to the triangular inequality, $d(u, H(v)) \leq d(u, L(u)) + d(L(u), H(v))$, so
\[
d^\text{HDLBR}(u, v) \leq d(u, L(u)) + d(L(u), H(v)) + d(H(v), L(v)) + d(L(v), v)
\]
Since $v$ is not in Table $(u)$, which means $\{v\} \not\subseteq B(u) \cup C(u)$. According to the definitions of $B(u)$ and $C(u)$, we have $d(u, v) \geq d(u, L(u))$ and $d(u, v) \geq d(v, L(v))$, so
\[
d^\text{HDLBR}(u, v) \leq 2d(u, v) + d(L(u), H(v)) + d(H(v), L(v))
\]
\[
\leq 2d(u, v) + 2d
\]
\[
\square
\]

Hence, one critical problem is what the $d$ will be on a typical power-law network. In the following, we use the random power-law graph theory proposed by Chung [36] and Lu [35] to show that with high probability $d$ is bounded by a constant in power-law graphs.

**Definition 1.** Let $G(w)$ represent the ensemble of all random graphs satisfying the expected degree sequence $w = (w_1, w_2, \ldots, w_n)$, where $w_i$ is the expected degree of node $i$. The probability that there is an edge connecting two nodes $i$ and $j$ is given by $p(w_i, w_j) = k^{-\gamma}$. If $w = (w_1, w_2, \ldots, w_n)$ follows power-law distribution, i.e., $p(w_i = k) \propto k^{-\gamma}$, then $G \in G(w)$ is called a random power-law graph [35,36].

Let $S^c_{\text{G(w)}}$ denote the set of nodes whose expected degrees are no less than $k$, then the following property holds for the subgraph induced by $S^c_{\text{G(w)}}$:

**Lemma 1.** Suppose $G \in G(w)$ is a power-law graph instance with the power-law exponent $\gamma$ satisfying $2 < \gamma < 3$. Given $t \geq n^0$ ($0 < \delta < \frac{1}{\gamma^2}$), then the diameter of the subgraph induced by $S^c_{\text{G(w)}}$ has probability $1 - n^\delta$ not exceeding $(1 + o(1)) \frac{\log n}{(3 - \gamma) \log t}$.

**Proof.** Readers can refer to Ref. [36] Claim 4.1 for this Lemma. \(\square\)

**Corollary 1.** Let $G$ be an instance of $G(w)$, and the landmark set $L$ to be composed of the $n^2(0 < x < 1)$ nodes with the highest expected degrees, then $d$ has probability $1 - n^\delta$ to be bounded by a constant.

**Proof.** Assuming the degree distribution of $G(w)$ follows $P(k) \propto k^{\gamma-2}$, then the cumulative degree distribution is $P(z \geq k) = c k^{1-\gamma}$, where $c$ could be a constant. Denote the smallest expected node degree in $L$ as $\tau$. According to the cumulative expected degree distribution, we can determine the probability that a node has a degree larger than $x$ is $c x^{1-\gamma} = n^\delta/n$, so:
\[
c t^{1-\gamma} = n^\delta/n \Rightarrow \log c - (\gamma - 1) \log \tau \approx (x - 1) \log n \Rightarrow \log \tau \approx (1 - x) \log n/(\gamma - 1) + \log c/(\gamma - 1) \Rightarrow \tau \approx c_1 n^{(1-x)/(\gamma-1)}
\]
where $c_1$ is a constant depending on $\gamma$ and $c$.

$L$ can be treated as $S^c_{\text{G(w)}}$ of $G(w)$. According to Lemma 1, the diameter of the subgraph induced by $L$ has probability $1 - n^\delta$ not exceeding $(1 + o(1)) \frac{\log n}{(3 - \gamma) \log \log n}$. Since $\gamma$, $c$ and $x$ can all take constant values, this expression is bounded by a constant as $n \to \infty$. According to the definition of $d$, $d$ is no greater than the diameter of the subgraph induced by $L$, hence $d$ is bounded by a constant with probability $1 - n^\delta$. \(\square\)

### 5.3. Routing table size

The storage overhead incurred by HDLBR can be analyzed separately for landmark and non-landmark nodes. For each landmark $L$, its storage overhead includes two parts: (1) the routing information to all other landmarks, i.e., $\{(v, e_i(v)) | v \in L - \{l\}\}$, and (2) the name-to-address mapping, i.e., $\{(v, Address_{\tau}(v)) | v \in V \land H(v) = l\}$. When the landmark size is in $O(n^{1/2})$, the number of name-to-address mapping entries stored on each landmark is $O(n^{1/2})$, hence the total storage overhead measured in terms of the number of entries of each landmark does not exceed $O(n^{1/2})$. Since each address can be represented in $O(\log n)$ bits, the routing table size of landmark nodes is $O(n^{1/2})$ bits. This also indicates that routing table size is balanced for landmark nodes.

In the following, we analyze the routing table size for non-landmark nodes. For each non-landmark node $nu$, its routing table is $Table(u) = Table_{\tau}(u) = \{(v, e_i(v)) | v \in L \cup C(u) \cup B(u)\}$. Since non-landmark nodes need not store the name-to-address mapping, their routing table sizes critically depend on the size of $B(u)$ and $C(u)$, again, we analyze the size of $B(u)$ and $C(u)$ by means of the power-law random graph theory.

**Lemma 2.** Let $\beta = \frac{\gamma - 2}{\gamma - 1} + \xi$, $\beta' = \frac{1}{\gamma - 1}$, and $\alpha = \beta' (\gamma - 2) + 2\gamma - 3\xi$, where $\xi$ is any positive real number that satisfies $n^{\frac{2\gamma - 3\xi}{\gamma - 1}} \geq 2(\gamma - 1)$ in $n$ (apparently, when $n$ is sufficiently large, $\xi$ could be very small). Let $G$ be an instance of $G(w)$, then for all $u \in V(G)$, the following property holds with probability at least $1 - 3n^{-2}$: $|B(u)| = |\{u' \in V(G) : d(u, u') < d(u, S^c_{\text{G(w)}})\}| = O(n^\alpha)$, where $\alpha = \frac{(1 - \gamma)\xi - 2}{\gamma - 1} + \xi$, $\xi$ is sufficiently small.

**Proof.** Please refer to Ref. [34], Lemma 7, for the proof of this Lemma. \(\square\)

**Corollary 2.** Let $G$ be an instance of $G(w)$, $L$ be the landmark set consisting of the $n^2$ nodes with the highest expected degrees, then for any $u \in V(G)$, the probability that $|B(u)| = O(n^\alpha)$ is at least $1 - 3n^{-2}$, where $\alpha = \frac{(1 - \gamma)\xi - 2}{\gamma - 1} + \xi$, $\xi$ is sufficiently small.

**Proof.** According to Corollary 1, the least expected degree $\tau = c_1 n^{(1-x)/(\gamma-1)}$, which can be rewritten as $\tau = n^{\frac{1-x}{\gamma-1}}$. Since $\xi$ is sufficiently small, according to Lemma 2, we know that $\alpha = \frac{(1-x)}{\gamma-1}(\gamma - 2) + 2\gamma - 3\xi$, i.e., $\alpha = \frac{(1-x)}{\gamma-1} + \xi$, where $\xi = (\gamma - 2)\epsilon + 2\gamma - 3\xi$. Since $\epsilon$ and $\xi$ can both take sufficiently small values, so can $\xi$. \(\square\)
For any \( u \in V(G) \), Corollary 2 gives the upper bound of \(|B(u)|\). However, it is impossible to analyze the bound of \(|C(u)|\) for each \( u \) in this way. We however, could provide the upper bound of the average size of \(|C(u)|\).

**Lemma 3.** For any arbitrary graph \( G \), \( \sum_{u \in V(G)} |B(u)| = \sum_{u \in V(G)} |C(u)| \), i.e., \(|C(u)|\) and \(|B(u)|\) have the same average size.

**Proof.** According to the definitions of \(|C(u)|\) and \(|B(u)|\), it follows that for any \( v \in B(u) \), we have \( u \in C(v) \), and vice versa. Since these \( u, v \) occur pairwise, so Lemma 3 holds. \( \square \)

Based on the above analysis, we have the following Theorem.

**Theorem 2.** Let \( G \) be an instance of \( G(w) \), and let \( L \) be the landmark set containing the \( n^2 \) nodes with the highest expected degrees, then for any non-landmark node \( u \in V(G) \), with probability at least \( 1 - 3n^{-2} \), the average routing table size is \(|Table(u)| = O(n^2 + n^3)\) bits, where \( \alpha = \frac{(1-\gamma)}{2} + \xi \) and \( \xi \) is sufficiently small. Here \( \bar{X} \) denotes the average of \( X \).

**Proof.** Since \(|L| = O(n^2)\), \(|B(u)| = O(n^3)\) and \(|C(u)| = |B(u)| = O(n^3)\), so it is easy to see that the average number of routing entries is \( O(n^2 + n^3)\). Since each routing entry can be represented in \( O(\log n) \) bits, it thus follows that the average routing table size is \( O(n^2 + n^3) \). \( \square \)

**Corollary 3.** For any \( \gamma \) satisfying \( 2 < \gamma < 3 \), the average routing table size of \(|Table(u)|\) on non-landmark nodes in the HDLBR scheme can be bounded by \( O(n^{1/2}) \) bits with probability at least \( 1 - 3n^{-2} \).

**Proof.** Since \( 2 < \gamma < 3 \), \( \alpha = \frac{(1-\gamma)}{2} + \xi \leq (1-x)/2 + \xi \), so \(|Table(u)| = O(n^2 + n^{(1-\gamma)/2})\). When \( x = 1/2 \), we have \(|Table(u)|\) be bounded by \( O(n^{1/2}) \) with probability at least \( 1 - 3n^{-2} \). \( \square \)

### 6. Simulation

In order to provide an insight into the real performance of our proposed scheme on power-law networks, we perform simulations on both the synthesized graphs generated by the random power-law graph (RPLG) model and the real AS-level Internet topology. We developed the simulation based on the JUNG graph package [44]. DIGG [45,32] is used to generate the random power-law graphs. We use different power-law exponent \( \gamma \) and network size \( n \) to generate different RPLG graphs, where \( \gamma \) takes the values in \( \{2.1,2.2,2.3,2.4,2.5\} \), and \( n \) takes the values in \( \{5000,6000,\ldots,15000\} \). For each pair of \( \gamma \) and \( n \), we generate 10 RPLG graphs, and the result presented in this paper is the average over the 10 graphs. The three AS-level Internet graphs we use are from the data provided by CAIDA's skitter project [39] and the routerviews project [46]. The basic topological properties of the three AS graphs are: \( n_1=9204, m_1=28595, \gamma_1 = 2.25 \), and \( n_2=17861, m_2=39637, \gamma_2 = 2.3 \), \( n_3=33559, m_3=75787, \gamma_3 = 2.3 \). We tested the average routing table size and average stretch of HDLBR over these networks. Also, we compared HDLBR to the Abraham scheme. In HDLBR, we explicitly set the number of landmarks to be \(|L| = n^{1/2} \). For the purpose of comparison, we have two realizations of the Abraham scheme. One is called the base Abraham scheme, which is based on random coloring and assignment of landmarks. Given the number

---

Fig. 3. Maximum distance between any two landmarks with HDLBR on random power-law graphs.

Fig. 4. Average stretch as a function of the power-law exponent \( \gamma \) (n = 10,000).

Fig. 5. Comparison of average stretches of the HDLBR scheme and Abraham schemes.
Table 2
Comparison of the average stretch and routing table size between the HDLBR scheme and the Abraham scheme on three AS-level Internet topologies.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Stretch</td>
<td>rt-size</td>
<td>Stretch</td>
</tr>
<tr>
<td>HDLBR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Base Abraham</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c = 0.5</td>
<td>1.61</td>
<td>1286</td>
<td>1.56</td>
</tr>
<tr>
<td>c = 0.4</td>
<td>1.55</td>
<td>819</td>
<td>1.48</td>
</tr>
<tr>
<td>c = 0.3</td>
<td>1.51</td>
<td>1345</td>
<td>1.43</td>
</tr>
<tr>
<td>Customized Abraham</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c = 0.5</td>
<td>1.28</td>
<td>1287</td>
<td>1.18</td>
</tr>
<tr>
<td>c = 0.4</td>
<td>1.25</td>
<td>1114</td>
<td>1.16</td>
</tr>
<tr>
<td>c = 0.3</td>
<td>1.22</td>
<td>1637</td>
<td>1.15</td>
</tr>
</tbody>
</table>

Fig. 6. Comparison of the average routing table sizes of the Abraham and HDLBR schemes on random power-law graphs ($\gamma = 2.3$).

Fig. 7. Average and maximum routing table size for both landmark and non-landmark nodes with the HDLBR scheme on random power-law graphs ($\gamma = 2.3$). Here, AVT RT and MAX RT stands for average and maximum routing table size respectively.

of colors $N^c$, the base Abraham scheme randomly colors the nodes and chooses a color set as the landmark. The other is called customized Abraham scheme, which takes into account the topological property in landmark construction. Specifically, the customized Abraham scheme works as follows: given the number of colors $N^c$, the $N^c$ nodes with highest degrees are first colored with a given color (say color one) and added to the landmark set, then the remaining nodes are randomly colored with $N^c$ colors and the nodes with the given color (say color one) are added to the landmark set. In order to keep the vicinity ball $B(u)$ as small as possible so that the routing table size of Abraham is not very large, we made slight improvement in the construction of $B(u)$ in both the base and customized Abraham Scheme. We use breadth-first search to construct $B(u)$. Instead of letting $B(u)$ contain $\lceil 4\sqrt{r \log(n)} \rceil$ closest nodes, we stop the search process as soon as the resulting ball covers all colors. As a result, the number of routing entries for our implementation of Abraham is expected to be smaller than the theoretical result of the original Abraham paper.

6.1. Stretch

Fig. 3 reports the maximum distance $d$ between any two landmarks on random power-law graphs by HDLBR scheme for $\gamma = 2.1–2.5$. We see that $d$ is almost insensitive to the network size (the fluctuation arises from the difference of average degrees in different sized RPLG graphs), which is in accordance with our analytical result. When $\gamma = 2.1$, the average value of $d$ is slightly larger than 5. As $\gamma$ increases, $d$ increases slightly as well. When $\gamma = 2.5$, the average value of $d$ goes around 12. This could be explained as: as $\gamma$ increases, the average degree of high degree landmarks and the connection density between them both decrease, so the maximum distance between them increases. We also perform simulation on the AS-level Internet topologies. The maximum distances between any two landmarks produced by HDLBR scheme for three AS graphs are 2, 5 and 4 respectively, quite small regarding its network size.

Fig. 4 illustrates how the average stretch varies with our HDLBR scheme when the power-law exponent increases. We see that as the power-law exponent $\gamma$ increases, the average stretch also increases slightly. When $\gamma$ increases, the degree distribution becomes more balanced, and the connection between landmarks becomes more loose. Since HDLBR stores the mapping information on landmarks, so looser connections between landmarks lead to more path overhead spent on the address lookup and routing steps, resulting in larger average stretch. This observation also demonstrates that HDLBR is more suitable for highly heterogenous power-law networks.

Fig. 5 compares the average stretch between the HDLBR scheme and several realizations of the Abraham scheme. We observe that HDLBR has comparable average stretch with the base Abraham scheme. If the Abraham scheme is customized for the power-law graph, it becomes superior to HDLBR in terms of the average stretch. In addition, we observe that the number of colors influences the average stretch of the Abraham scheme. When the number of colors decreases (so the landmark size increases), the average stretch of the Abraham scheme also decreases. Even with the base Abraham scheme, when $c = 0.3$, the average stretch becomes smaller than that of HDLBR scheme.

Table 2 gives the average stretch of the HDLBR scheme and the aforementioned realizations of the Abraham scheme on three
AS-level Internet topologies. Similar to the synthesized graphs, we observe that the HDLBR scheme has comparable or smaller average stretches than the base Abraham scheme, but is slightly outperformed by the customized Abraham scheme.

6.2. Routing table size

In the following, routing table size is measured by the number of routing entries. Fig. 6 compares the average routing table sizes of HDLBR and Abraham on random power-law graphs ($\gamma = 2.3$). It can be seen that the average routing table size generated by HDLBR is only a little bit larger than $|L|$, much smaller than the Abraham scheme. On a random power-law graph with 10,000 nodes, the average routing table size is only 119 in the HDLBR scheme, whereas it is over 900 for our Abraham scheme realizations. Fig. 7 also plots the average and maximum routing table size for landmark and non-landmark nodes separately with HDLBR scheme. Our theoretical analysis shows that routing table size for landmark nodes are balanced, however, to this end, we still cannot mathematically show whether routing table size on non-landmark nodes are balanced. The maximum routing table size for non-landmark nodes in Fig. 7 gives us some hints that roughly, the maximum routing table size will not exceed twice the average routing table size, which is proved to be bound by $O(n^{1/2})$. Also, we see that the number of colors influences the average routing table size in the Abraham scheme. For both the base and customized Abraham schemes, the minimum routing table size is achieved at $c = 0.4$ in our settings. Table 2 also compares the average routing table size of the HDLBR scheme and the Abraham scheme on real AS-level topologies. It is obvious that the HDLBR have much smaller routing table size than either the base or customized implementations of the Abraham scheme. The reason why HDLBR allows very small routing table is because on power-law networks, each $u \in V(G)$ has high probability to be directly connected to high degree nodes or separated from high degree nodes with only a few hops, so the sizes of $B(u)$ and $C(u)$ can be very small. Indeed, we observed that in the AS graph with 17861 nodes, the average ball size is less than one, indicating the ball size of a significant number of nodes are zero. Whereas in the Abraham scheme, each ball should at least contain all colors, hence its ball size is at least in the order of $O(n^{1/2})$.

7. Conclusion

Compact routing is a good candidate to fundamentally address the scalability issue of the routing infrastructure. In this paper, we proposed a name-independent compact routing scheme, named HDLBR, that is specialized for power-law networks. By designating the $n^{1/2}$ highest degree nodes as landmarks and distributing the dictionary table used for name-to-address translation uniformly across the landmarks, we show that on random power-law network model: (1) the stretch of this proposed scheme is $(2, \Delta)$, where $\Delta$ is the maximum distance between any two landmarks, typically very small for power-law networks, and (2) the routing table size on landmark nodes is balanced and bounded by $O(n^{1/2})$ bits, and for non-landmark nodes on power-law networks with $2 < \gamma < 3$, the routing table size is bounded by $O(n^{1/2})$ with high probability. Simulation results on both the synthesized random power-law networks and the AS graphs corroborate our analytical evaluation.

We noticed that recent proposals for Internet routing architecture strongly recommend Locator/ID separation (LIS) [43] to support the scalability, mobility and security requirements of future Internet. Essentially, name-independent routing schemes implement the idea of LIS [5]. However, despite its natural support of the LIS idea, all the performance evaluation of HDLBR is undertaken in static network environment. Understanding how it behaves in the dynamic environment is critical for developing a practically scalable routing protocol, which could be the future research direction.

Acknowledgements

M. Tang is supported by the National Natural Science Foundation of China under Grant No. 61100054, Hunan Provincial Natural Science Foundation of China under Grant No. 11J6064 and Scientific Research Fund of Hunan Educational Committee under Grant No. 11B048. G. Zhang is supported by the National Natural Science Foundation of China under Grant No. 6110178 and the Startup Foundation of Nanjing Normal University under Grant No. 20111119XQGQ0248. J. Liu is supported by the National Natural Science Foundation of China under Grant No. 61272063, Program for New Century Excellent Talents in University under Grant No. NCET-10-0140 and Excellent Youth Foundation of Hunan Scientific Committee under Grant No. 11JJ1011.

References