LMD: a local minimum driven and self-organized method to obtain locators

Yonggong Wang, Gaogang Xie
Institute of Computing Technology
Chinese Academy of Sciences
Beijing, P.R.China
{wangyonggong,xie}@ict.ac.cn

Mohamed-Ali Kaafar
INRIA Rhône-Alpes
Grenoble, France
mohamed-ali.kaafar@inria.fr

Steve Uhlig
Queen Mary, University of London
London, UK
steve@eecs.qmul.ac.uk

***Abstract*—The scalability of routing architectures for large networks is one of the biggest challenges that the Internet faces today. Greedy routing, in which each node is assigned a locator used as a distance metric, recently received increased attention from researchers and is considered as a potential solution for scalable routing. In this paper, we propose LMD — a Local Minimum Driven method to compute the topology-based locator. Contrary to previous work, our algorithm employs a quasi-greedy and self-organized embedding method, which outperforms similar decentralized algorithms up to 20% in success rate. To eliminate the negative effect of the "quasi" greedy property — transfer routes longer than the shortest routes, we introduce a two-stage routing strategy, which combines the greedy routing with source routing. The greedy routing path discovered and compressed in the first stage is then used by the following source-routing stage. Through extensive evaluations, based on synthetic topologies as well as on a snapshot of the real Internet AS topology, we show that LMD guarantees 100% delivery rate on large networks with a very low stretch.**

I. Introduction

The information-centric networking (ICN) paradigm is one of the new trends in future Internet research [1], where a "*name*" is the primary identifier, instead of the IP address. However, the well-known Rekhter's Law: "*Addressing can follow topology or topology can follow addressing. Choose one.*"[2] does not hold in ICN, exposing their routing system to even greater scalability issues.

Distributing "*names*" rather than IP addresses will make the situation much worse than in today's Internet. For example, [3] reports that a name-based routing table carrying only the top-level domains as prefixes would have to hold 2×10^8 routes, to be compared to the only 4×10^4 in today's Internet. Arguably, we need topology-based identifiers, also called locators [4], as an intermediate layer on top of which more scalable routing systems can be built in ICN. In recent work [5], [6], [7], [8], greedy routing has been identified as a potential solution to answer the scalability needs. In greedy routing, the "*name*" is first translated into a locator by a mapping service comparable to the current DNS service. Then, the packets are greedily routed according to the locator.

Greedy routing in large networks has already been studied in the past [9]. However, it was not until Kleinberg's seminal work [10] that is started to be considered as an attractive routing solution. In [10], each node is assigned a synthetic coordinate, called locator (in our study denoting its virtual location), and it can route greedily by selecting a neighbor that is the closest to the destination.

Unlike traditional routing protocols that require routers to maintain the next hops for every destination, greedy routing only relies on local information: the locators of its neighbors. This localized routing strategy has two advantages: 1) the size of the routing tables is significantly reduced, from the number of nodes in network to the number of neighbors; 2) the router only needs to process the update messages originated from its neighbors, which is a very desirable feature for large networks. However, greedy routing suffers from the "local minimum problem", which sometimes prevents the routing from reaching the destination.

In this paper, we present a greedy routing protocol to compute locators, called LMD (Local Minimum Driven). Unlike most previous work that depend on the global knowledge of the topology, our method is self-organized and configuration-free. Experiments show that LMD can achieve better success rate, in another word—less local minimum, comparing with existing methods up to 20%. LMD, honestly, can not eliminate local minima completely. To overcome this issue, we enhance the greedy routing and not only to resolve the local minimum issue, but we also compress the length of the greedy forwarding path.

The rest of this paper is organized as follow. The LMD (Local Minimum Driven) method and the enhanced greedy routing algorithm are detailed in Section II. Section III evaluates LMD. In Section IV, we discuss related work. Section V concludes the paper.

II. LMD: A Local Minimum Driven Locator Calculation

A. Algorithm design philosophy

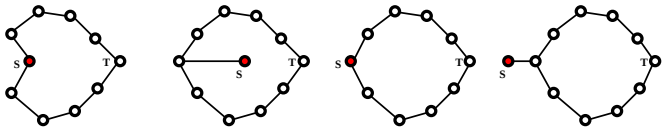
Greedy routing is typically made of two components: (1) the greedy embedding (computing the locator) and (2) the greedy forwarding algorithm (using the locator to forward packets). Each component's success depends on the other's completeness and complexity. In other words, if the greedy embedding process is well designed, the greedy forwarding process should be trivial. Otherwise, forwarding must com-

pensate for the shortcomings of the embedding, to guarantee packet delivery.

In LMD, we chose a simple greedy embedding algorithm in combination with a complex forwarding algorithm. Despite the complexity of a self-organized and configuration-free greedy embedding, we believe that it is important to make progress in "autonomic networking" [11], [12]. To compensate for the lack of guarantees in packet delivery, we rely on a mechanism to compress the routing path. The two corresponding components of LMD, the local minimum driven method (LMD embedding) and the corresponding enhanced greedy routing (LMD forwarding), are detailed in the following two subsections.

B. Local minimum driven method

Local minima are typically due to the shortcomings of locators. The rigorous greedy embedding is defined as a mapping $f : V(G) \rightarrow (X, \rho)$ such that $\forall s \neq t \in V(G)$ it holds that s has a neighbor w with $\rho(f(w), f(t)) < \rho(f(s), f(t))$,where graph $G = (V, E)$. Similarly, the local minimum is defined as the node s which neighbors are farther to t than themselves. For example, in Fig. 1(a), packets to the node t might be stuck at node s, which is called a local minimum node. While in Fig. 1(b), a better greedy embedding is constructed, where no local minimum exists. Calculating the embedding locators for a small graph is straightforward. In large networks on the other hand, it can be computationally challenging. Inspired by



(a) Locators with local minimum (b) Locators without local minimum

Fig. 1. Local minima in locators.

the difference between two sets of locators in Fig.1, we found that slightly moving the node stuck in the local minimum, s, away from the destination node t, will make the embedding. We note that, though both s and t are "correctly" embedded, while the neighbors of s may lead the local minimum at s. We prefer, in LMD, to move the local minimum node s to satisfy the self-organized constraint in our design principle.

Furthermore, we require that the locators of neighboring nodes be close to each other. This is quite reasonable in light of the definition of the hidden metric space [13]: *The smaller the distance between two nodes in the hidden metric space, the more likely they are connected*. Given that we only allow to use local information, a spring is created between any pair of neighbors. The pull force between connected nodes will help to prevent the locator space from inflating infinitely. The attraction forces between neighbors can be described as follows:

$$F = \frac{1}{\log d}$$

, where d is the metric distance between neighbors.

The complete LMD embedding algorithm is described in Fig. 2, which is executed once a packet is stuck at a local minima node v. The step factor c is a tuning parameter in the LMD algorithm. If c is too large, the correcting process may overshoot with infinite oscillations. A too small value will make the time to leave the local optimum longer. In our experiments, we use a constant value of 0.1 for c. At

Input: $v, L_v, t, L_t, N_v, L_{N_v}, c$
Output: L_v, L_{N_v}
$\backslash\backslash v$ is the local minimum node.
$\backslash\backslash t$ is the target node.
$\backslash\backslash N_i$ is the neighbors of node i.
$\backslash\backslash L_i$ is the locator of node i.
$L_v \leftarrow L_v + c(L_v - L_t)$
$\backslash\backslash$move local minimum node v away from target node t.
for node n,$n \in N_v$ **do**
$F \leftarrow \frac{1}{\sum_{nn \in N_n} \log(L_{nn} - L_n)}$
$L_n \leftarrow L_n + c\frac{F}{|N_n|^2}$
$\backslash\backslash$the correctness at the local minimum also benefit its neighbors
end for

Fig. 2. Local minimum driven algorithm.

the very beginning, each node will choose a random vector as its initial locator. Fig. 3 illustrates how LMD constructs the locators gradually in a 10×10 grid network. In this example, all packets are sent randomly between sources and destinations. The locators are expressed in a 2-dimensional Euclidean metric space. Even though the success rate of greedy routing in the final graph is very high, i.e., 99%, there are still about 1% of the packets that stuck at the local minima. To deal with the local minima due to the LMD embedding, we present an enhanced greedy routing (LMD forwarding) in the following subsection.

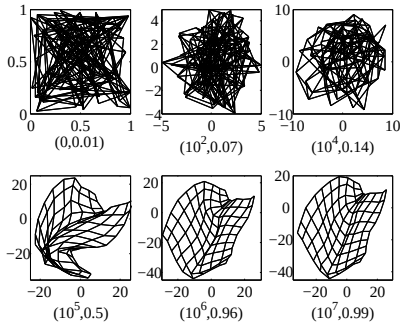


Fig. 3. LMD in a 10×10 grid network. The number of packets since the beginning of simulation and the corresponding success rate of greedy routing are shown in the brackets below the subgraphs.

C. Enhanced greedy routing

In a standard greedy forwarding algorithm, two conditions are used to select the neighbor to which packets are forwarded:

1) it is closest to the destination among all the neighbors and, 2) it is closer to the destination than the current node. When among the possible neighbors, none satisfies condition two, the forwarding process is stuck in a local minimum and fails.

As we mentioned in the previous subsection, LMD embedding cannot generate the perfect greedy locators, i.e., distances might not be monotonously decreasing on the path of a packet. Therefore, we must drop the second condition in LMD: packets are always forwarded to the closest neighbor towards the destination. However, this condition relaxation in turn can creates forwarding loops, which in standard greedy forwarding is avoided by design. To resolve the loop, all the visited nodes are recorded in the packet and will be avoided by subsequent forwarding. If all the neighbors of a given node have been visited, a random neighbor is chosen as next hop to break ties.

Although a 100% successful delivery rate can be ultimately guaranteed through random walks, it will be at the expense of a much longer path than the actual shortest path [14]. In order to reduce the greedy path length, we propose and implement a distributed path compressing method in LMD. Rather than forwarding all the packets greedily as suggested in previous work [15], [6], [16], [17], we design a two-stage routing scheme. During the first stage, routing discovery, the enhanced greedy forwarding is performed. In the second stage, data packets are routed on the "compressed" route obtained from the routing discovery stage. In LMD, the path learned in the first stage can be compressed thanks the following two rules:

- If a node occurred in the path more than once, the edges between the occurrences are removed.
- If a node and its neighbor both occurred in a path, the edges connecting the node and its neighbor in the path should be replaced by the direct edge.

The asymmetric routing means that "source-destination" and "destination-source" paths are not always equal during a topology discovery. We further compress the path by exploiting the asymmetry in the paths. Fig. 4 illustrates the asymmetry in routing discovery and the corresponding path compression. The high-level steps of our two-stage LMD routing scheme are illustrated in Fig. 5. When the source node has packets to send to a destination node, a routing discovery packet will be sent and forwarded to the destination node through the enhanced greedy forwarding. The routing discovery packet records all the visited edges in its payload. The first path compression rule can be applied by the destination node locally. However, since the destination node only maintains a local topology view, it does not know the nodes included in the path towards such destination, the second path compression rule must be applied in the backward path. The probe packets indeed help in overcoming the local minima as well as reduce the path stretch thanks to the route discovery and path compression.

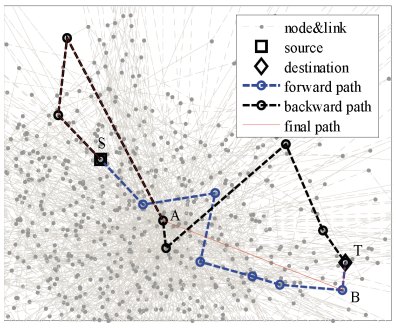


Fig. 4. Both-way path compression: The forward path (7 hops) denoted by the blue dashed line and the backward path (7 hops) denoted by the black dashed line are connected by edge AB. The final path after compression, denoted by the red solid line, has only 5 hops.

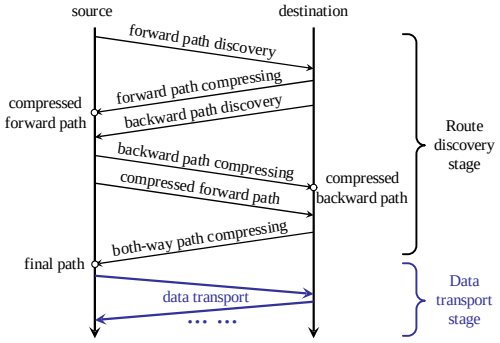


Fig. 5. High-level steps of LMD routing.

III. Evaluation

A. Experiment settings

As discussed in the previous sections, LMD does not make assumptions about the network topology structure. In this paper we focus on scale-free networks, because they have been identified throughout multiple natural and engineering areas, including the Internet [18]. The scale-free networks in our study are generated by the Barabási-Albert (BA) model[19], in which each node is added to the network with m connections at each step. The probability of attaching to an existing node of degree k is proportional to k. This model yields a power-law degree distribution with exponent γ between 2 and 3. The default parameters of the network topology used in our experiments is $N = 1,000, m = 2, k = 4, \gamma = 2.5$ unless otherwise specified.

Since the locator is corrected gradually by the packets stuck at local minimum, the traffic distribution, or called traffic model, is important in the LMD evaluation. Thus, traffic following Zipf-law and uniformly distributed traffic are used to evaluate the performance of LMD. Unless otherwise specified, the source nodes and the target nodes of the packets are randomly and uniformly chosen in our evaluation.

B. LMD on synthetic scale-free networks

As the quality of greedy embedding which can be evaluated by the success rate is the basis of greedy routing, first, we compare the embedding performance of LMD with three existing decentralized embedding methods in Fig. 6, where [16]

and [20] are two recent work constructing greedy embedding by self-organized methods, which are labeled as AVE and FPC separately. Vivaldi[21] is a well-known work in network coordinates which can be seen as an isometric embedding. Considering the fact that a perfect isometric embedding is also a guaranteed greedy embedding (but not the converse), it is chosen as a candidate greedy embedding algorithm, where the packet delay between nodes in original paper is replaced by the hops of shortest path. We can see that LMD outperform [16] and [20] up to 20% with different dimensions. Though all the pair-wise distance are known in [21], our method still get better success rate. Besides success rate, convergent rate

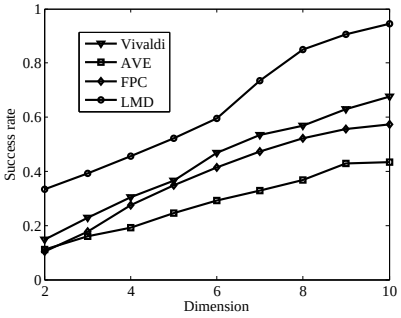


Fig. 6. The comparison of LMD with decentralized embedding methods. There are $1,000$ nodes in this scale-free network ($\gamma = 2.5$).

is another important index for greedy embedding in dynamic network environment. Fig. 7 illustrates the convergency of LMD with different failure ratio, where the failed nodes are those with wrong locators but can execute LMD embedding properly. Furthermore, this ratio can also be seen as the ratio of new added nodes at some time. In this figure, all nodes are stay in the convergency state during the first $1,000$ packets. Then, we reset 1% to 30% nodes' locators to random values. We can see that after about $3,000$ packets per node, the average success rate recovers to the normal level in all scenarios. Considering the increasing bandwidth of current network equipments, LMD embedding is quite robust with network churn.

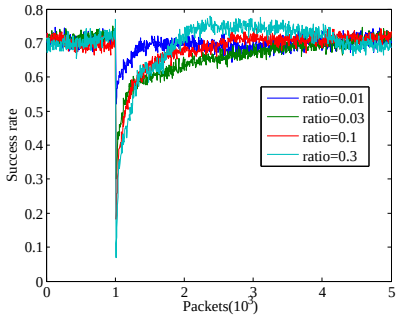
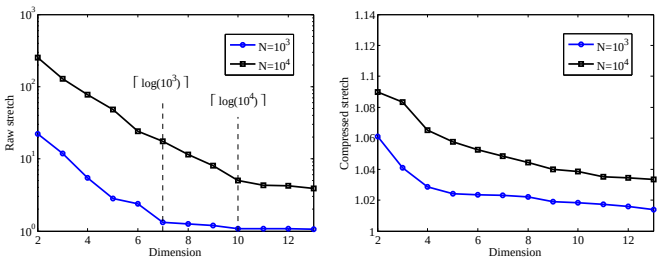


Fig. 7. The convergency of LMD with different failure ratio. Some parts of nodes lose their locators after send 1,000 packets, and reconstruct the greedy embedding in the following packets delivery. There are 1,000 nodes in this scale-free network ($\gamma = 2.5$). The embedding space is 7-dimension space.

Fig. 8 illustrates how the dimension of the metric space affects the performance of LMD forwarding. The raw stretch is the ratio of the greedy forwarding path length in the routing discovery stage to the theoretical shortest path length. The compressed stretch is the ratio of the final data path length to the shortest path length. Higher dimensions are expected to give better routing performance. This comes at a cost of a larger locator, as well as more overhead in locator computation and packet forwarding. As expected, we observe on Fig. 8(a) that the path stretch gets close to 1 when the dimensionality reaches $log(N)$. This is consistent with previous work [22] proving $log(N)$ to be the lower bound of a no-stretch greedy embedding, where N is the number of nodes in network. The benefits of higher dimensions is limited, which our results confirm. The compressed stretch on Fig. 8(b) illustrates how well path compression allows the path stretch to get close to 1, even for low dimensionality.



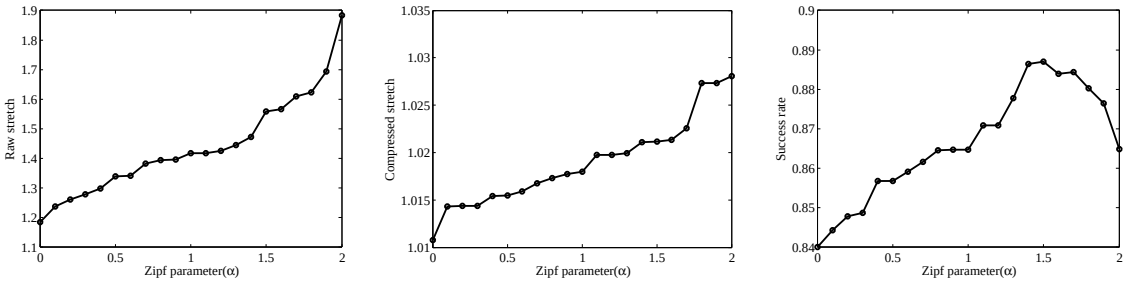
(a) Raw stretch (b) Compressed stretch

Fig. 8. LMD forwarding in Euclidean space with different dimensions. There are 1,000 nodes in this scale-free network ($\gamma = 2.5$)

In most real networks, traffic is not uniformly distributed among all nodes. To simulate more realistic traffic [23], a Zipf distribution is used for the traffic towards target nodes, where the probability of the top t visited node is proportion to $1/t^{\alpha}$. Note that a Zipf exponent of 0 corresponds to a uniform distribution, while the higher the value of α, the stronger the bias towards a limited number of destinations. We observe on Fig. 9(a) that the stretch increases with α. Indeed, the less uniform traffic will prevent routing discovery from improving the length of the compressed paths, see Fig. 9(b). The success rate of LMD also depends on the uniformity of the traffic across the source-destination pairs. For example, Fig. 9(c) shows that depending on the value of the exponent α, the success rate varies. With heavy-tailed distributions such as Zipf and large vakues of α, the limited number of targeted destinations may not help LMD to build good locators.

C. LMD on Internet AS topology

Scalable routing in large network has been widely studied. We choose to compared LMD to two other centralized routing schemes, PIE [17] and TZ [24](the comparing with decentralized algorithms has been shown in Fig. 6). PIE is a recently proposed method to extract multi-level trees to embed a topology. We also compare LMD to PIE with single global tree, because its routing stretch is theoretically equal to many other single tree based methods, e.g., [15], [25]. TZ [24] is



(a) Raw stretch (b) Compressed stretch (c) Success rate

Fig. 9. The performance of LMD changes with bias traffic. The source nodes are uniformly distributed, and the distribution of target nodes follows the Zipf law. There are 1,000 nodes in this scale-free network($\gamma = 2.5$)

an optimal version of the classic compact routing [26] for scale-free networks. We rely on a recent snapshot of the AS level topology [27] of the Internet for our comparison. This topology was collected in November 2011 and contains $39,973$ ASes and $251,630$ edges. The main parameters of each routing method are the following:

- TZ: The core size is 200, similar to the square of the AS number, as required by the routing protocol.
- PIE1: Only one global tree rooted at the highest degree node.
- PIE12: The embedding contains 12 levels of tree hierarchy, which approximates to $log(N)$, N being the number of ASes. $log(N)$ is the preferred level in PIE.
- LMD12: LMD with 12 dimensions locator, whose space overhead is similar to PIE1.
- LMD144: LMD with 144 dimensions locator, whose space overhead is similar to PIE12.

Fig. 10 compares the CDF of the path stretch of the different methods. As expected, the stretch of PIE1 and LMD12 is worse due to their low overhead and dimensionality. Despite that TZ is the only method that guarantees a maximum stretch of 3, the CDF of the path stretch of TZ is worse than for PIE12 and LMD144. Indeed, the average stretch of both PIE12 and LMD 144 are slightly above 1, at 1.03. While PIE and LMD have similar performance in path stretch, they are very different: PIE focuses on guaranteeing the success rate, while LMD ensures that no global coordination is necessary, i.e., it is a self-organized protocol.

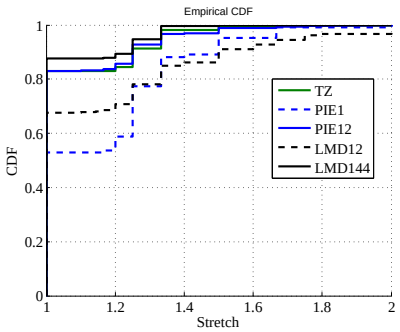


Fig. 10. The comparison of LMD with other scalable routing methods.

IV. Related work

Greedy routing, also called geometric routing or geographic routing, has been studied for over a decade, especially in ad-hoc wireless networks [28]. The two most popular approaches in this area, i.e., planar graph and UDG (Unit Disk Graph), either do not work with arbitrary networks, or are costly or unrealistic. Recent work [13], [15], [29], [25], [30], [16], [17] questioned the design rules for greedy routing in arbitrary graphs. Most of the early work can be grouped into three categories: routing on planar or "quasi" planar graphs, routing on spanning trees, and routing on general graphs.

A. Greedy routing on planar or"quasi" planar graph

Almost all the greedy routing algorithms in wireless networks belong to this category because of a widely used hypothesis: a link between any two nodes is considered if the two nodes are located in each other's transmission range (or radius), which makes planar graphs an appropriate approximation for wireless networks. This assumption makes sense in wireless environments, but unfortunately does not match networks such as the Internet.

GSpring [31] adjusts the node's coordinates by simulating a system of springs and repulsive forces. [31] defined the concept of the ownership region of each node (analogous to a Voronoi diagram), and used the conflict region to adjust coordinates iteratively. Another well-known routing algorithm in this category is NoGeo in [32], which also creates synthetic coordinates through an iterative relaxation algorithm. BVR [33] does not require a planar network. Even though it performs well in wireless environments, it experiences poor performance for low degree networks, such as Internet.

B. Greedy routing on spanning tree

Greedy routing on spanning trees can be considered as special case of planar graphs. We review this category separately because of the large body of recent literature [25], [30], [17], [34], [6], [35], most of which fixing aspects of the seminal [15]. Computing the spanning tree of the whole network is a common step for this class of algorithms. In [15], a strictly greedy embedding algorithm is proposed such that any connected finite graph can be embedded in a two-dimension hyperbolic metric space. [25] removed the need

for global knowledge in [15] (the tree's maximum degree) and presented an online greedy embedding. Furthermore, [25] proposed a Gravity-Pressure routing method to deal with the greedy routing failure caused by network dynamics. The size of the coordinates was improved by [34], from $O(n)$ bits to $O(\log n)$ bits. [6] presents a greedy routing with bounded stretch by constructing a constant-stretch tree, while in [35] authors propose a way of approximately projecting the greedy embedding into an $O(\log n)$-dimension metric space where each coordinate only needs $O(\log n)$ bits. Recently, [17] extracted several trees with different locality levels and combined them together to perform a greedy routing.

Although the spanning tree protocol, a precondition for all the algorithms mentioned above, has been widely used in real networks for many years [36], it is not trivial to make it work in large networks for two reasons: 1) Choosing the root node is challenging; 2) The robustness of the the routing protocol might be questioned as the graph is cut into trees.

C. Greedy routing on general graphs

The origin of greedy navigability in scale-free network is explored in [13]. A hidden metric space is defined for the first time to explain the connectivity of a given network: *The smaller the distance between two nodes in the hidden metric space, the more likely they are connected*. [37] claims to propose a self-organized greedy routing method, but their scheme only guarantees the delivery to a sink node, and no complete point-to-point routing is evaluated. [30] approaches the problem from the opposite direction: the hyperbolic space is fixed and a graph is constructed in it. Our pervious work FPC[20] is a force-directed method for greedy embedding, LMD in this work get better embedding quality with similar requirements.

LMD algorithm belongs to this 3^{rd} category. As opposed to the previous algorithms in this category that completely depend on the hidden metric space, we present a two-stage routing scheme that combines greedy routing with source routing, which yields a high success rate and reasonably low stretch values on general networks.

V. Conclusion

In this paper, we have presented LMD — a self-organized and configuration-free greedy routing scheme. The key idea in our design is using the local minimum information in greedy routing to correct the locators in the network. Additionally, LMD combines a quasi-greedy but robust embedding method, with an enhanced greedy routing. Experiments on an Internet topology shows that LMD can achieve a high success rate on large networks with low stretch.

References

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A Survey of Information-Centric Networking," *IEEE Communication Magazine*, 2012.

[2] D. Meyer, L. Zhang, and K. Fall, "Report from the IAB Workshop on Routing and Addressing," RFC 4984, 2007.

[3] A. Narayanan and D. Oran, "NDN and IP Routing: Can It Scale?" http://trac.tools.ietf.org/.

[4] D. Meyer, "The Locator Identifier Separation Protocol (LISP)," *The Internet Protocol Journal*, 2008.

[5] P. Rodrigues and J. L. Martins, "Greedy Routing in the Internet: Is it a Solution?" in *Proc. of CRC*, 2010.

[6] R. Flury, S. Pemmaraju, and R. Wattenhofer, "Greedy routing with bounded stretch," in *IEEE INFOCOM*, 2009.

[7] M. Boguna and D. Krioukov, "Navigating ultrasmall worlds in ultrashort time," *Phys. Rev. Lett.*, 2009.

[8] "Greedy Forwarding on the NDN Testbed," http://www.caida.org/research/routing/greedy_forwarding_ndn/.

[9] G. Finn, "Routing and addressing problems in large metropolitan-scale internetworks," USC Tech. Rep. ISI/RR-87-180, 1987.

[10] J. Kleinberg, "Navigating ultrasmall worlds in ultrashort time," *Nature*, 2000.

[11] J. Kephart and D. Chess, "The vision of autonomic computing," *IEEE Computer*, 2003.

[12] "Autonomic Network Architecture," http://www.ana-project.org.

[13] M. Boguna, D. Krioukov, and kc claffy, "Navigability of complex networks," *Nature Physics*, 2009.

[14] H. Tian, H. Shen, and T. Matsuzawa, "Random walk routing for wireless sensor networks," in *PDCAT*, 2005.

[15] R. Kleinberg, "Geographic routing using hyperbolic space," in *IEEE INFOCOM*, 2007.

[16] Z. Zhuo, S. Cai, Z. Fu, and W. Wang, "Self-organized emergence of navigability on small-world networks," *New Journal of Physics*, 2011.

[17] J. Herzen, C. Westphal, and P. Thiran, "Scalable Routing Easy as PIE: a Practical Isometric Embedding Protocol," in *ICNP*, 2011.

[18] A. Barabasi, "Scale-free networks: A decade and beyond," *Science*, 2009.

[19] A. Barabasi and R. Albert, "Emergence of scaling in random networks," *Science*, 1999.

[20] Y. Wang, G. Xie, and M. Kaafar, "Fpc: A self-organized greedy routing in scale-free networks," in *ISCC*, 2012.

[21] F. K. R. M. F. Dabek, R. Cox, "Vivaldi: A decentralized network coordinate system," in *ACM SIGCOMM*, 2004.

[22] P. Maymounkov, "Greedy embeddings, trees and Euclidian vs. Lobachevsky geometry," Technical Report, available at http://pdos.csail.mit.edu/petar/pubs.html, 2006.

[23] D. Alderson, H. Chang, M. Roughan, S. Uhlig, and W. Willinger, "The many facets of internet topology and traffic," *Networks and Heterogeneous Media*, 2006.

[24] W. Chen, C. Sommer, S. Teng, and Y. Wang, "Compact routing in power-law graphs," in *Proceedings of the 23rd international conference on Distributed computing*, 2009.

[25] A. Cvetkovski and M. Crovella, "Hyperbolic embedding and routing for dynamic graphs," in *IEEE INFOCOM*, 2009.

[26] M. Thorup and U. Zwick, "Compact routing schemes," in *Proc. of SPAA*, 2001.

[27] "Internet Topology Collection," http://irl.cs.ucla.edu/topology/.

[28] D. Chen and P. K. Varshney, "A survey of void handling techniques for geographic routing in wireless networks," *IEEE Commun. Surveys Tuts.*, 2007.

[29] R. Flury and R. Wattenhofer, "Randomized 3D Geographic Routing," in *IEEE INFOCOM*, 2008.

[30] F. Papadopoulos, D. Krioukov, and A. Vahdat, "Greedy forwarding in dynamic scale-free networks embedded in hyperbolic metric spaces," in *IEEE INFOCOM*, 2010.

[31] B. Leong, B. Liskov, and R. Morris, "Greedy virtual coordinates for geographic routing," in *ICNP*, 2007.

[32] A. Rao, S. Ratnasamy, and C. Papadimitriou, "Geographic routing without location information," in *ACM MOBICOM*, 2003.

[33] R. Fonseca, S. Ratnasamy, J. Zhao, C. T. Ee, D. Culler, S. Shenker, and I. Stoica, "Beacon-vector routing: Scalable point-to-point routing in wireless sensor networks," in *ACM NSDI*, 2005.

[34] D. Eppstein and M. Goodrich, "Succinct greedy graph drawing in the hyperbolic plane," in *Proc. of Graph drawing*, 2009.

[35] C. Westphal and G. Pei, "Scalable routing via greedy embedding," in *Proc. of Infocom Mini-Conference*, 2009.

[36] "ANSI/IEEE Standard 802.1D," LAN/MAN Standards Committee of the IEEE Computer Society, 1990.

[37] T. Watteyne, I. Augé-Blum, M. Dohler, S. Ubéda, and D. Barthel, "Centroid virtual coordinates - a novel near-shortest path routing paradigm," *Computer Networks*, 2009.