on Communications

**VOL. E99-B NO. 3
MARCH 2016**

The usage of this PDF file must comply with the IEICE Provisions on Copyright.

The author(s) can distribute this PDF file for research and educational (nonprofit) purposes only.

Distribution by anyone other than the author(s) is prohibited.

A PUBLICATION OF THE COMMUNICATIONS SOCIETY

**The Institute of Electronics, Information and Communication Engineers
Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3chome, Minato-ku, TOKYO, 105-0011 JAPAN**

PAPER

Efficient Geometric Routing in Large-Scale Complex Networks with Low-Cost Node Design

Sahel SAHHAF^{†a)}, *Member*, Wouter TAVERNIER[†], Didier COLLE[†], Mario PICKAVET[†],
and Piet DEMEESTER[†], *Nonmembers*

SUMMARY The growth of the size of the routing tables limits the scalability of the conventional IP routing. As scalable routing schemes for large-scale networks are highly demanded, this paper proposes and evaluates an efficient geometric routing scheme and related low-cost node design applicable to large-scale networks. The approach guarantees that greedy forwarding on derived coordinates will result in successful packet delivery to every destination in the network by relying on coordinates deduced from a spanning tree of the network. The efficiency of the proposed scheme is measured in terms of routing quality (stretch) and size of the coordinates. The cost of the proposed router is quantified in terms of area complexity of the hardware design and all the evaluations involve comparison with a state-of-the-art approach with virtual coordinates in the hyperbolic plane. Extensive simulations assess the proposal in large topologies consisting of up to 100K nodes. Experiments show that the scheme has stretch properties comparable to geometric routing in the hyperbolic plane, while enabling a more efficient hardware design, and scaling considerably better in terms of storage requirements for coordinate representation. These attractive properties make the scheme promising for routing in large networks.

key words: geometric routing, greedy forwarding, greedy embedding, spanning tree, large-scale topology, scale-free networks

1. Introduction

Geometric routing has been proposed in the literature as an alternative to traditional lookup-based routing schemes to solve their scalability issue in terms of memory consumption. In this routing, only local information is required to find the next hop of a packet. Although geometric routing was initially designed for ad hoc and wireless sensor networks (WSNs) [1], the applicability of this routing paradigm to wired networks has gained increasing interest and has been investigated as well [2]. However, the latter require different design approaches because unlike WSNs, many of the inhomogeneously wired networks are modeled as scale-free networks*. Geometric routing is being applied in different areas. Concretely, Content-Centric Networking (CCN) is one example which can benefit significantly from geometric routing. CCN is known to work in many environments from high-speed data centers to resource constrained sensors. It is also used in Internet of Things (IoT) scenarios as was investigated in [3]. Using this routing, efficient and scalable content-based forwarding is possible as was suc-

cessfully demonstrated in our previous works [4], [5].

The idea of geometric routing is to attach coordinates to nodes in a network. An obvious choice is to attach GPS coordinates to the routers of a communication network [1]. Every node in the network is aware of its own coordinate and the coordinates of its neighbors. Upon arrival of a packet, the distance between every neighbor and the packet's destination is calculated and the neighbor with maximal decrease in the distance is selected as the next hop. Following this distance-decreasing policy, the packet can eventually reach the intended destination. This scheme is referred to as greedy forwarding/routing because in every step, the neighbor which maximally decreases the distance towards the destination is selected. In [6] authors used the same idea but instead of physical location of the nodes, virtual coordinates are assigned to them. In this routing, each node stores only the coordinates of its neighbors which is more memory-efficient than lookup-based routing. The cost of this efficiency is more computation complexity in determining the next hop (distance calculation instead of a lookup). Therefore, this routing replaces the lookups with more computation in forwarding. Figure 1(a) depicts an example of greedy forwarding (from 'd' to 'a') in a Euclidean space.

In greedy forwarding, packets might get stuck in a local minimum (void). This means that the current node is the closest node to the destination among all of its neighbors. Figure 1(b) depicts an example of a local minimum. Greedy embeddings have been proposed to solve this issue

Fig. 1 Example of greedy forwarding in Euclidean space is given in (a) and (b) depicts an example of local minimum in greedy forwarding.

*A scale-free network is a network whose degree distribution follows a power law, at least asymptotically. That is, the fraction $P(k)$ of nodes in the network having k connections to other nodes goes for large values of k as $P(k) \approx c.k^{-\gamma}$. Wikipedia, https://en.wikipedia.org/wiki/Scale-free_network

Manuscript received June 23, 2015.

Manuscript revised September 25, 2015.

[†]The authors are with the Internet-Based Communication Networks and Services (IBCN) research group of the Department of Information Technology (INTEC), Ghent University — iMinds, Belgium.

a) E-mail: sahel.sahhaf@intec.ugent.be
DOI: 10.1587/transcom.2015EBP3268

[7]. A greedy embedding for a given graph $G(V, E)$ into a metric space X , is a function from $V(G)$ to X such that for all graph nodes $s \neq t$, s has a neighbor u which decreases the distance toward t in metric space X . Several works proposed greedy embeddings in different metric spaces such as hyperbolic plane [8], [9] and multidimensional Euclidean spaces [10], [11]. Greedy forwarding based on these embeddings never gives rise to local minima and thus 100% successful delivery of the packets is guaranteed. Therefore, in geometric routing, a structured address space is required to guarantee successful packet delivery. However, such dependency on a structured space results in: i) possible deviation from shortest paths and ii) non-local changes with regard to the location of topological changes.

An efficient geometric routing is required especially in large networks where scalability is the main concern. Such an efficient routing scheme should facilitate computation in forwarding plane to enable low-cost design while minimizing required memory in network nodes and act rapidly upon network changes with short convergence time.

Our contribution. In this paper, we investigate a simple and powerful greedy embedding. In this embedding, instead of relying on complex geometry, network nodes coordinates are derived from a spanning tree of the network. This decreases the overall overhead/complexity of the scheme compared to approaches based on hyperbolic [8], [9] or high-dimensional Euclidean spaces [10], [11]. The less complex the approach, the more likely it would be used in practice. The main goal of this paper is to illustrate that a geometric router can be implemented in a very low-cost way without sacrificing efficiency of the routing. We propose an efficient low-cost circuit to greedy forward the packets based on the calculated tree-coordinates. The cost of the proposed forwarder is measured in terms of area complexity of the hardware design. Routing efficiency is measured in terms of routing quality (deviation from the shortest path length) and the required storage for coordinate representation. The efficiency of the routing is compared with a state-of-the-art embedding in the hyperbolic plane proposed in [8]. Evaluation results indicate that the proposed scheme can be implemented through a very efficient low-cost design without degrading the routing performance in terms of stretch[†] while scaling significantly better in terms of coordinate precision compared to the hyperbolic embedding. Additionally, in [12], we have evaluated the convergence behavior of this scheme which shows good performance in terms of convergence time upon changes in the network topology. These make the scheme very suitable candidate for routing in large-scale networks.

Importantly, this paper is different from the existing work in the sense that it explores: i) the experimental execution of the mentioned embeddings on large-scale networks with respect to the resulting stretch and required coordinate precision, and ii) a novel low-cost hardware design

of a greedy forwarder. To the best of our knowledge, no research has been conducted so far focusing on such design. We evaluate these topics and show that our scheme outperforms the hyperbolic-embedding in several aspects. The advantage of using the proposed embedding is apparent in the hardware implementation. The proposed scheme in this paper is applicable to a wide range of topologies, however this paper targets scale-free complex networks in general. A wide range of social, biological, technological and communication systems can be described as complex networks. Scale-free networks are one of the well-known and much studied classes of complex networks in which the nodes connectivities (nodes degrees) follow a power-law distribution. Such networks expand continuously by the addition of new nodes and new nodes attach preferentially to already well-connected nodes. Technological networks such as the router-level graphs and domain-level graphs in the Internet, peer-to-peer networks and the World Wide Web belong to this category of the networks. Graphs that represent such networks typically have thousands to millions of nodes and links. Therefore, all our evaluations are performed on large-scale scale-free topologies. This should enable better conclusions regarding the applicability of the proposed scheme on a wide range of real systems.

The rest of the paper is organized as follows. Section 2 explains the related work. In Sect. 3, the tree-based embedding is explained in detail. Section 4 describes the hardware design of a greedy forwarder based on the embedding. Section 5 includes the experimental results and Sect. 6 discusses existing challenges and future plans. Finally Sect. 7 concludes the paper.

2. Related Work

Although many proposed approaches for geometric routing target different types of networks (e.g. unit-disk graph^{††}) rather than scale-free topologies, it is beneficial to investigate the existing approaches used to avoid local minima to guarantee packet delivery. In the following we explain the existing techniques however, not all of them are applicable to scale-free topologies as they do not possess the desired graph properties.

Two groups of techniques have been proposed to solve the local minimum issue in geometric routing: i) face routing techniques and ii) greedy embeddings. In face routing techniques, the void is bypassed by routing around this area and greedy forwarding is resumed from the moment a closer node than the local minimum is reached [13]. However, such techniques suffer from several issues: i) the graph should be planar/planarized and ii) planarizing might not be feasible in every graph.

In greedy embeddings, nodes are embedded in a metric space in such a way that for every node there is always a distance decreasing neighbor toward every destina-

[†]The ratio between the length of the path produced by greedy forwarding and shortest path length.

^{††}A graph $G = (V, E)$ is a unit-disk graph when $\forall u, v \in V : v \in N(u) \Leftrightarrow \delta(u, v) \leq 1$ in case G is embedded into a Euclidean space, with δ the Euclidean distance.

tion in the network. There are several works which propose greedy embeddings in different metric spaces. Works such as [8] and [9] proposed embeddings in hyperbolic space. In [14] authors proposed another embedding in the hyperbolic plane which requires $O(\log n)$ bits (with n being the network size). [10] proposed an embedding in a Euclidean $O(\log^2 n)$ -dimensional space and a poly-logarithmic dimension embedding into Euclidean space was proposed in [11].

There are several proposed embeddings based on one or multiple spanning trees of the network [8], [15]–[17]. These works are interesting in the sense that they guarantee the packet delivery using a simple structure such as a tree. Some of them consider simple labeling/numbering of nodes instead of coordinates in a complex geometry. While [8] embeds a network spanning tree in the hyperbolic plane, [15] and [16] use a labeling of the nodes with the path from the tree root to each node. In [17] a different embedding compared to [15], [16] with poly-logarithmic memory complexity was proposed. In their work, the good performance in terms of stretch is achieved at the cost of higher complexity in the forwarding process and relying on multiple trees. In a recent work [18], enabling load balancing in geometric routing using multiple trees was proposed. The effect of using multiple spanning trees on network resiliency was investigated in [19]. In general, using multiple trees improves the geometric routing performance. This good performance is at the cost of: i) construction of multiple trees ii) increased storage requirements of the packet headers (as a node location is represented by multiple coordinates) and iii) increased computational complexity of the forwarding plane (as the next hop is determined considering multiple coordinates).

Through experimental results we show that despite the simplicity of the embedding based on a single tree, a good performance in terms of stretch and coordinates bit precision can be achieved. We then propose a circuit to implement a greedy forwarder based on this embedding. The simplicity of the embedding and forwarding result in a low-cost hardware design. Such an embedding with the low-cost hardware and stretch close to 1 is a very promising scheme for large-scale networks.

3. Greedy Tree-Based Geometric Routing (GTR)

GTR is composed of two components: i) tree-based greedy embedding and ii) greedy forwarding. In the embedding scheme, a spanning tree of the network is constructed and nodes coordinates are derived from this tree. These coordinates are then used by the forwarding component to forward the packets to the intended destinations. In the concept of geometric routing, to send a packet from node s to d , s should know the coordinate of d . Therefore, a mapping system to bind node identifiers to node coordinates is required. Such a mapping system can be based on a DNS-like mechanism. There are several proposals in the literature for the mapping systems [20] focusing on different aspects such as scalability, resiliency, security and end-host mobility. How-

Fig. 2 The tree-based embedding is depicted in (a) and (b) illustrates an example of greedy forwarding.

ever, the detail of such mapping system is out of the scope of the paper and we assume there exists a mapping system in the network.

3.1 Tree-Based Greedy Embedding

Instead of embedding network nodes to a complex metric space (e.g. hyperbolic), this paper evaluates the use of tree-coordinates. This will reduce the computational overhead and complexity of the overall scheme. As we will see in Sect. 5, this will not penalize the resulting performance. The approach relies on construction of a spanning tree of the network.

Given a spanning tree (see Fig. 2(a)), every node can be assigned a Coordinate Set (CS). The number of coordinates in the sets is determined by the depth of the tree. These sets can be derived using the following steps:

1. The root node sets all of its coordinates to zero: $(0, \dots, 0)$, and each node numbers its children from 1 to d (d is the number of the node's children in the tree).
2. A node calculates the CSs of its children by putting the number assigned to each child in place of the first zero coordinate in its own CS, e.g. $(1, 0, \dots, 0)$ for the first child of the root node, and $(d, 0, \dots, 0)$ for the last.

In [15] and [16] similar coordinates in ad hoc and sensor networks were used. However, the routing in [15] only used the tree edges and referred to as tree routing. They discussed several possibilities such as using multiple trees and shortcuts[†] to further improve the efficiency of their routing.

These coordinates reflect the location of the nodes in the tree. Therefore, the depth and the maximum degree of the tree determine the size of the CSs. Each coordinate in a CS should have enough bits to be able to show a value equal to the maximum degree of the tree. Experimental results in Sect. 5 confirm that given a suitable tree, the coordinate size differs significantly from the characterization of $O(n)$ bit required, made in [16] (n is the network size).

As investigated in [21], the choice of the spanning tree may impact the performance of the geometric routing. For this reason, we focus on a tree which has minimal depth with a root node which has large(st) degree in order to minimize memory requirements for coordinate representation and enable shorter paths. This can be done by selecting the node

[†]Links which are not in the spanning tree.

with maximum degree as the root of the tree and constructing a Breadth-First-Search (BFS) tree from that node. As our focus is on scale-free networks, such a tree construction method does not generate very deep branches because the average distance between nodes in scale-free networks is very short. In [12], we proposed a distributed algorithm to implement such a tree. In this algorithm, each node of the network has a unique identifier (ID) which is composed of two parts: i) the degree of the node and ii) a unique number to be used as tie breaker. These two parts are concatenated in such a way that a node with higher degree has a higher ID as well. Each node in the network initiates spanning tree generation considering itself as the root of the tree. As we need to generate a single spanning tree, all except one tree is suppressed based on a rule. Using the nodes IDs, the tree initiated by the root with the highest ID remains. Additionally, the level of the nodes in the selected tree is taken into account to enable BFS tree construction. This is done by sending ANNOUNCEMENT messages which include two parameters to indicate: (i) the ID of the root node initiating the tree and (ii) the level of the node receiving the message upon joining the tree. When an ANNOUNCEMENT(newroot,newlevel) from j arrives at i :

- if (newroot > myroot) or (newroot = myroot and newlevel < mylevel): node i should suppress its current tree/location in the tree due to its lower priority. It updates its data structures and joins j 's subtree with newroot as the root and newlevel as the location in the tree. It then announces the new states to its neighbors except j .

Eventually all nodes receive the ANNOUNCEMENT message indicating the root node with the highest ID and abandon the lower priority trees. Note that the tree-based embedding and the spanning tree generation can be performed simultaneously [12]. The messages that are exchanged between network nodes to generate the spanning tree can include the calculated CSs. This means that once node i sends an ANNOUNCEMENT message to its neighbor j , it can calculate the corresponding CS (see Sect. 3.1) for j and add it to the same message. Upon arrival of this message, if j decides to change its parent to i , it should also update its CS to the one announced in the message. This way, once the tree is generated the coordinates are also calculated.

Each node announces its CS and ID to its neighbors (using ADVERTISEMENT messages) and nodes store the neighbors CS in a table to enable greedy forwarding.

3.2 Greedy Forwarding Based on Coordinate Sets (CS)

Once every node has its deduced CS using the above procedure, packets can be forwarded towards neighbors which (maximally) reduce the distance towards the CS of the destination node mentioned in a received packet. This process is different from tree routing because the shortcut links can also be used. In this context, we propose to use tree-distance

as metric. This refers to the hop count on the tree between two nodes and can be calculated as follows:

1. The closest common ancestor to both nodes is found.
2. The hop count of each node to the ancestor is counted.
3. The sum of these two hop counts determines the tree-distance between them.

This distance can be calculated easily using the assigned CSs to the network nodes. Consider CSs (2,1) and (2,2) in Fig. 2(a). Their longest common prefix (2) determines the CS of the closest common ancestor which is (2,0). The number of non-zero coordinates after the common prefix in each CS determines the hop count of each node to the common ancestor. Both CSs have only one non-zero coordinate after the common (2). Thus, the tree-distance between them is 2. Note that in case of no common prefix, the closest common ancestor is the root (0,0) and all non-zero coordinates of each CS should be counted.

Figure 2(b) depicts an example of greedy forwarding (from $S=(2,2)$ to $D=(1,1)$) based on the derived CSs. Each node is aware of the CSs of its neighbors. In each node the tree-distance between every neighbor and D is calculated and the one with maximum decrease in the distance is selected as the next hop. In node S , the tree-distances (td) are as follows: $td((0,0),(1,1))=2$, $td((2,0),(1,1))=3$ and $td((2,1),(1,1))=4$. Therefore, (0,0) is selected as next hop. Following this greedy forwarding leads to the path depicted by arrows. This path is 1 hop longer than the shortest path $((2,2) \rightarrow (2,1) \rightarrow (1,1))$ and one hop shorter than the path produced by tree routing $((2,2) \rightarrow (2,0) \rightarrow (0,0) \rightarrow (1,0) \rightarrow (1,1))$.

3.3 Delivery Guarantee

In this section, we explain that the tree-based embedding is a greedy embedding. Based on [8], if $H \subseteq G$ is a subgraph containing all vertices of G , then every greedy embedding of H is also a greedy embedding of G . In order to verify that the tree-based embedding is a greedy embedding, it suffices to show that for every path $s = s_0, s_1, \dots, s_k = t$ in the tree, the inequality $td(s_0, t) > td(s_1, t)$ is satisfied (with $td(i, j)$ being the tree-distance between i and j). This is always satisfied, because there is exactly one path between every two nodes in a spanning tree of a network. Therefore, the hop count of the second node along the path (s_1) to the destination (t) is exactly one hop less than the hop count between the first node (s_0) and the destination (t). Thus distance can always be decreased by 1 by following the tree. Whether shortcuts exist or not does not change this fact, they only help getting closer to the destination “faster”.

4. Hardware Design of a Greedy Forwarder

As geometric routing is a recent paradigm, emerging in the last decade, no research has focused on the design of a circuit in hardware supporting greedy forwarding of the packets yet. In order to close the gap between geometric routing

Fig. 3 The tree-distance calculator circuit.

in theory and its applicability in practice, we propose a hardware design of a greedy forwarder.

The simple structure of the proposed coordinates and the distance function in the forwarding process (explained in Sect. 3) enables designing of a low-cost greedy forwarder. The cost of this design was quantified in terms of area complexity of the hardware and was compared to a state-of-the-art approach with virtual coordinates in the hyperbolic plane which is reported in Sect. 5. The main objective of these evaluations is to illustrate that a geometric router can be implemented in a very low-cost way without sacrificing efficiency of the routing.

A greedy forwarder, independent of the type of coordinates and the distance function, is composed of two major components: i) a distance calculator and ii) a comparator. The distance calculator calculates the distance between every neighbor and the destination of a packet and the comparator selects the minimum value among them. We propose a circuit for the tree-distance calculator. For the sake of clarity, the circuit for only one neighbor is explained which can be easily extended for more neighbors. In this circuit, given two CSs (the CS of the destination 'D' and a neighbor 'N') the tree-distance between them is calculated.

Figure 3 depicts the proposed design for a tree-distance calculator. The idea in this circuit is to find the first uncommon coordinate in the two CSs and start counting the non-zero coordinates from that location in each CS. As illustrated in this figure, the input CSs are serial. The reason is twofold: i) the packets arrive sequentially and thus the destination CS included in the packet arrives sequentially as well (one bit after the other) and ii) compared to parallel designs, bit-serial designs result in a huge reduction in the required hardware. To serialize the neighbor CS, it is stored in a shift register. In Fig. 3 the values at different stages of the architecture are marked for sample CSs including 3 coordinates each composed of 3 bits.

We first explain the upper part of the circuit. The inputs are fed into a XOR gate which detects the different bits in the two CSs. By feeding the output of the XOR to an OR gate together with a feedback from a flip flop (F.F), we get a signal in the output of the F.F.1 which is set to '1' when the first uncommon bit in the two CSs is detected and it remains '1' until the last bit of the CS. As the coordinates should be counted and not the bits in each CS, the output of the F.F.1 is

fed to an AND gate together with a pattern which determines the end of each coordinate by a '1' bit. This pattern can simply be generated by a conventional modulo- K up counter (K is the number of bits in a coordinate). Such a counter starts counting from 0 and is reset when it reaches K . The outputs of F.Fs producing a logic '1' when the number in the counter is K are fed to an AND gate whose output is tied to the reset pins of the F.Fs in the counter. Using another AND gate with its inputs tied to the outputs of F.Fs producing a logic '1' when the number in the counter is $K - 1$, the desired pattern is generated as this AND's output.

The lower part of the circuit guarantees that only non-zero coordinates in each CS are counted. Note that in the tree-based embedding, as each node numbers its children starting from 1, it is impossible to have a zero coordinate between non-zero ones in a CS. Therefore, including such a logic is sufficient to count the hop counts correctly. The design of this part is dependent on the number of bits in a coordinate. Consider CS of the neighbor 'N' in Fig. 3. This CS should be delayed $K - 1$ times using $K - 1$ F.Fs (K is the number of bits in a coordinate). The output of each F.F is the input of the next one (see F.F.2 and F.F.3). The outputs of F.Fs and the original CS 'N' are combined. This can be done by a K -way OR gate or $(K - 1)$ 2-input OR gates (see OR2 and OR3). If there is at least one bit equal to '1' in each coordinate of 'N', the output of OR3 is '1' at the location of the last bit of that coordinate. These locations are highlighted in the inputs of the OR gates (OR2 and OR3). The same logic is repeated for CS of the destination 'D'. Applying the output of OR3 (OR5) and the output of AND1 to AND2 (AND3) results in a signal which is '1' only at the end of each non-zero coordinate and only if the first uncommon coordinate is detected. We refer to the outputs of AND2 and AND3 as count-pulse1 and count-pulse2 respectively. Counting the number of '1's in each count-pulse and calculating the sum of these two numbers determine the tree-distance between the two CSs. Instead of using two counters and an adder, we propose a counter which is capable of parallel counting. Given the two count-pulses if both are '0' no counting should take place, if only one of them is '1' the counter should count 1 and if both are '1', the counter should count 2. Using AND4, we determine where both count-pulses are '1' and using XOR2, we determine where only one of them is '1'. Feeding the outputs of

Fig. 4 The 3-bit counter circuit.

AND4 and XOR2 to the inputs of the proposed counter, the hop counts of both CSs are counted.

Figure 4 depicts the circuit of a 3-bit counter which is a modification of a conventional counter based on D-flip flops. Although asynchronous counters are less complex and require fewer components, we selected the synchronous design as it is faster and more reliable. In this counter an OR gate is added to the second stage of the counter. In case ‘in1’ is ‘1’ the counter counts 1 and if ‘in2’ is ‘1’, it counts 2. In our design ‘in1’ and ‘in2’ of the counter are not ‘1’ at the same time.

5. Performance Evaluation and Analysis

GTR was evaluated in a custom simulation environment developed in Python/C++ on a set of network topologies defined by the Barabasi-Albert (B-A) model [22] which generates random scale-free networks. These topologies were evaluated in different scales.

We evaluated GTR in terms of stretch, coordinate size and area complexity of the proposed hardware design. In the experiments, the selected spanning tree for embedding is a BFS tree rooted at the maximum degree node.

In order to avoid that we compare GTR only to ‘legacy’ routing mechanisms (i.e. shortest path routing), we compare it to a state-of-the-art geometric routing scheme in order to make fair comparisons and be able to estimate performance difference of routing schemes of the same class. We selected hyperbolic-embedding of [8] for comparison for several reasons: i) it is based on a single spanning tree (other works such as [16], [17] benefit from multiple trees in the network) ii) it is applicable to any connected topology and provides 100% successful delivery (no need for planarization and face routing techniques) and iii) it results in good performance in terms of stretch (given a suitable tree). We want to explore if the performance is the result of complicated hyperbolic structure/computations or a simple tree labeling with simple distance calculation/forwarding can be equally (more) efficient.

5.1 Routing Stretch Evaluation

Routing stretch is defined as the ratio between the length of the path produced by greedy forwarding and the length of the shortest path for the same source-destination pair. We evaluated GTR performance in terms of stretch to determine how the path length is affected using this scheme. We compared the stretch of GTR with: i) the stretch of greedy for-

Fig. 5 Average stretch in B-A networks.

Fig. 6 Stretch percentile of GTR in B-A networks.

warding based on the hyperbolic-coordinates [8] and ii) the stretch of tree routing [15] when shortcuts were not used.

In this experiment, we calculated the stretch for every pair of nodes in the network. The average stretch for each topology is calculated and this is repeated for 10 B-A topologies and the average over all of them is reported. As we see in Fig. 5, the stretch of GTR is very similar to the routing based on hyperbolic-embedding and also outperforms tree routing. In Fig. 6, we report the stretch percentile of GTR on large-scale B-A topologies using Boxplot. Boxplots are used to represent the stretch values through their quartiles. The values on top of the graph are average stretch. The results show that the tree-based embedding scales very well with the increase in the number of nodes and the average stretch remains close to 1. In almost all topologies, up to 75 percentile of the stretch is 1.25 and the maximum stretch is 4.

5.2 Coordinate Size Evaluation

We calculated the required number of bits for tree-based coordinate representation and compared it with the minimal floating point bit precision for hyperbolic-coordinate which results in a valid greedy embedding. As a hyperbolic-coordinate consists of 2 floating points, the total size of a coordinate is twice the minimal bit precision.

Figure 7 depicts the obtained results for B-A networks of varying scale between 50 nodes and 20K nodes, and the required number of bits for representing the coordinates, for both tree and hyperbolic coordinates. The difference in the bit requirement in the two embeddings is significant and the tree-based coordinates use the address space more efficiently. The results show that given a suitable spanning

Fig. 7 Required number of bits in coordinates in B-A networks.

tree as basis for the embedding, that in scale-free topologies, the required space-complexity for storing tree-based coordinates is significantly more scalable than the hyperbolic-coordinates.

5.3 Area Complexity Evaluation

We evaluated the performance of the proposed circuit in terms of required silicon area and compared it with a hyperbolic-distance calculator. With this comparison, we determine how much chip area is saved in our design.

The hyperbolic distance function in the Poincare Disk (a model in the hyperbolic geometry used in [8]) is defined by $d(u, v) = \text{arccosh}(1 + \delta(u, v))$ with $\delta(u, v) = \frac{\|u-v\|^2}{(1-\|u\|^2)(1-\|v\|^2)}$, in which $\|\cdot\|$ refers to the Euclidean norm. This function is composed of arccosh , multiplications/divisions and additions/subtractions. As in greedy forwarding, we do not need to find the exact distance between points, we can simplify the distance function. Also implementing complex mathematical functions such as arccosh and division consumes a large silicon area. Because arccosh is a monotonic function, we can avoid this calculation as it would not change the resulting comparison. The divisions can be removed for the same reason. As the coordinate of the destination is the common factor in every distance calculation, the division to this coordinate can be removed. The division to the second factor can be replaced by a multiplication to the inverse of that factor, but this value should be stored for every neighbor. With these simplifications, there are still 3 multiplications and 3 additions required for distance calculation. We consider these modules in the calculation of the silicon area of the hyperbolic-distance calculator. Note that for comparison, we considered a sequential design for this calculator.

The required silicon areas of the designs are derived from their basic logic modules. To evaluate the area complexity of our design against the hyperbolic-distance calculator, the area of the logic cells are normalized with an AND gate from the same standard cell library. This benchmarking method has been widely adopted in the literature. The normalized areas of different logic cells are reported in Table 1 based on [23]. The required area of both designs in different scales from 1K to 20K nodes are reported in Table 2. For the adders in the hyperbolic-based design, we used the area of a full adder and a flip flop (i.e., a sequential full adder), and the

Table 1 Logic modules normalized area.

Element	AND	OR	XOR	D-FF	Full adder
Area	1	1	1.99	4.24	5.24

Table 2 Normalized area(μm^2) of the tree (T) and hyperbolic (H) distance calculator for different scale B-A networks.

	1K	5K	10K	15K	20K
T	128.97	144.35	155.83	155.83	155.83
H	18773.6	21517.6	21517.6	21517.6	21517.6

area of the multipliers are calculated based on [24]. The forwarder based on the tree-based embedding is significantly more area-efficient than hyperbolic-based design. The evaluations in this section indicate that a geometric router can be implemented in a very low-cost way without sacrificing efficiency of the routing.

6. Challenges and Future Work

One of the challenging tasks in geometric routing is dealing with dynamics in the topology. In [12] we observed the convergence behavior of the geometric routing scheme in case of topology changes which was dependent on the location of the changes in the topology. As the embeddings are based on the connectivity of a spanning tree, a single failure in the tree might change the coordinates of many nodes. However, the results in [12] indicated that although the number of affected nodes might be high, the corresponding convergence is quite fast in networks with 1000 nodes. The number of affected nodes in very large networks (o(100K) nodes) could be quite high and thus, the convergence time might be very large as well. As a solution, protection techniques can be used to avoid changes in the coordinates as we successfully demonstrated in [25] and [26]. A clever combination of the protection schemes and coordinate re-calculation technique can lead to good performance in terms of stretch and convergence time [12]. Another solution to improve the convergence behavior of geometric routing in large-scale networks involves using multiple trees as we proposed in [19]. Minimizing the overlap of the constructed trees, there is a high chance that at least one tree remains unaffected upon failures in the network and greedy forwarding remains successful without any coordinate re-calculation. However, the complexity of the scheme may slightly increase.

Another challenge is to provide a scalable mapping system to bind node identifiers (e.g. existing IP addresses) to node locators (e.g., coordinates). Although there exist several proposals in the literature, there are still some open issues towards a scalable mapping system with fast convergence [20].

In the proposed scheme, although shortcut links are used to forward the packets to their intended destination, it is still possible that some traffic is routed towards the root of the tree in order to reach the destination. This might lead to congestion in the root node or the links/nodes close to that. In order to avoid such an issue, we investigated the usage of

multiple trees to enable load balancing in geometric routing [18]. Using multiple trees, in every step, the neighbor with less loaded link is selected as the next hop. This way the congestion in different parts of the network is avoided.

Finally, the scheme should be investigated thoroughly in terms of security vulnerability and possible solutions. One of the main vulnerability in the proposed scheme is that nodes might claim false degrees and the frequent appearance/disappearance of such nodes leads to frequent global coordinate re-calculation if the announced degree is higher than the degree of the current root node. Using protection mechanisms [25], [26] is one way to avoid global coordinate re-calculations. However, as explained in [12], there is a trade-off between communication cost and stretch/node reachability when protection is used. This means that when there is a change in the network, the pre-determined backup paths are used which causes no communication cost but the stretch and node reachability are negatively affected. A possible solution to improve this performance is to have (partial/global) coordinate re-calculation periodically, upon reaching a certain threshold or based on some condition in the network. The second approach involves using authentication mechanisms. This way we avoid that any malicious node becomes the root of the tree and triggers the global coordinate re-calculation and only authenticated nodes can be the root node. As future work, we will investigate possible approaches to solve the security vulnerability of the scheme.

7. Conclusion

We investigated a simple but powerful approach to calculate virtual coordinates for network nodes in geometric routing. Greedy forwarding based on these coordinates guarantees 100% successful delivery of the packets to their destination. The coordinates are based on a spanning tree of the network. We proposed a novel low-cost circuit to greedy forward the packets based on the deduced tree-coordinates. We evaluated the tree-based embedding thoroughly in terms of stretch, coordinates bit requirement, and silicon area complexity of the proposed hardware design and compared it with an existing hyperbolic-embedding. The results showed that good routing performance can be achieved without complex coordinate and distance calculations, with an efficient hardware node design and more scalable memory requirements for coordinate representation. To validate the feasibility of the scheme on large networks, some of the simulation experiments were performed on topologies of up to $o(100K)$ nodes. These results bring geometric routing based on tree-coordinates to the pole position as a memory-scalable scheme for routing in large-scale complex networks with low stretch and low (hardware) complexity.

Acknowledgment

This work is partly funded by the European Commission through the EULER project (grant 258307), part of the Future Internet Research and Experimentation (FIRE) objec-

tive of the Seventh Framework Programme (FP7).

References

- [1] B. Karp and H.T. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," *Proc. 6th Annual International Conference on Mobile Computing and Networking, MobiCom'00*, pp.243–254, 2000.
- [2] M. Boguñá, F. Papadopoulos, and D. Krioukov, "Sustaining the Internet with hyperbolic mapping," *Nature Communications*, vol.1, no.6, pp.1–8, 2010.
- [3] E. Baccelli, C. Mehlis, O. Hahm, T.C. Schmidt, and M. Wählisch, "Information centric networking in the IoT: Experiments with NDN in the wild," *arXiv preprint arXiv:1406.6608*, 2014.
- [4] W. Tavernier, S. Sahhaf, D. Colle, M. Pickavet, and P. Demeester, "Towards content-centric geometric routing," *2014 IEEE 21st Symposium on Communications and Vehicular Technology in the Benelux (SCVT)*, pp.133–138, 2014.
- [5] S. Sahhaf, D. Papadimitriou, W. Tavernier, D. Colle, and M. Pickavet, "Experimentation of geometric information routing on content locators," *2014 IEEE 22nd International Conference on Network Protocols*, pp.518–524, 2014.
- [6] R. Fonseca, S. Ratnasamy, J. Zhao, C. Ee, D. Culler, S. Shenker, and I. Stoica, "Beacon vector routing: Scalable point-to-point routing in wireless sensor networks," *Proc. 2nd Conference on Symposium on Networked Systems Design & Implementation*, vol.2, pp.329–342, USENIX Association, 2005.
- [7] C.H. Papadimitriou and D. Ratajczak, "On a conjecture related to geometric routing," *Theoretical Computer Science*, vol.344, no.1, pp.3–14, 2005.
- [8] R. Kleinberg, "Geographic routing using hyperbolic space," *IEEE INFOCOM 2007, 26th IEEE International Conference on Computer Communications*, pp.1902–1909, 2007.
- [9] A. Cvetkovski and M. Crovella, "Hyperbolic embedding and routing for dynamic graphs," *IEEE INFOCOM 2009, The 28th Conference on Computer Communications*, pp.1647–1655, 2009.
- [10] R. Flury, S.V. Pemmaraju, and R. Wattenhofer, "Greedy routing with bounded stretch," *IEEE INFOCOM 2009, The 28th Conference on Computer Communications*, pp.1737–1745, 2009.
- [11] C. Westphal and G. Pei, "Scalable routing via greedy embedding," *IEEE INFOCOM 2009, The 28th Conference on Computer Communications*, pp.2826–2830, 2009.
- [12] S. Sahhaf, W. Tavernier, D. Colle, M. Pickavet, and P. Demeester, "Experimental validation of resilient tree-based greedy geometric routing," *Computer Networks*, vol.82, pp.156–171, 2015.
- [13] A. Maghsoudlou, M. St-Hilaire, and T. Kunz, "A survey on geographic routing protocols for mobile ad hoc networks," *Systems and Computer Engineering*, Technical Report SCE-11-03, Carleton University, 2011.
- [14] D. Eppstein and M.T. Goodrich, "Succinct greedy geometric routing using hyperbolic geometry," *IEEE Trans. Comput.*, vol.60, no.11, pp.1571–1580, 2011.
- [15] E. Chávez, N. Mitton, and H. Tejada, "Routing in wireless networks with position trees," *Ad-Hoc, Mobile, and Wireless Networks*, pp.32–45, Springer, 2007.
- [16] M. Tang, H. Chen, G. Zhang, and J. Yang, "Tree cover based geographic routing with guaranteed delivery," *2010 IEEE International Conference on Communications*, pp.1–5, 2010.
- [17] J. Herzen, C. Westphal, and P. Thiran, "Scalable routing easy as PIE: A practical isometric embedding protocol," *2011 19th IEEE International Conference on Network Protocols*, pp.49–58, 2011.
- [18] R. Houthoofd, S. Sahhaf, W. Tavernier, F. De Turck, D. Colle, and M. Pickavet, "Robust geometric forest routing with tunable load balancing," *Proc. 2015 IEEE Conference on Computer Communications (INFOCOM)*, pp.1382–1390, 2015.
- [19] R. Houthoofd, S. Sahhaf, W. Tavernier, F. De Turck, D. Colle, and M. Pickavet, "Fault-tolerant greedy forest routing for complex net-

- works,” Proc. 2014 6th International Workshop on Reliable Networks Design and Modeling (RNDM), pp.1–8, 2014.
- [20] M. Hoefling, M. Menth, and M. Hartmann, “A survey of mapping systems for locator/identifier split Internet routing,” *IEEE Communications Surveys & Tutorials*, vol.15, no.4, pp.1842–1858, 2013.
 - [21] A. Cvetkovski and M. Crovella, “On the choice of a spanning tree for greedy embedding of network graphs,” *Networking Science*, vol.3, no.1-4, pp.2–12, 2013.
 - [22] A. Barabási and R. Albert, “Emergence of scaling in random networks,” *Science*, vol.286, no.5439, pp.509–512, 1999.
 - [23] “Standard cell library,” www.zhuhai.gov.cn/image200105186428.pdf
 - [24] M.R. Meher, C.C. Jong, and C.-H. Chang, “A high bit rate serial-serial multiplier with on-the-fly accumulation by asynchronous counters,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol.19, no.10, pp.1733–1745, 2011.
 - [25] S. Sahhaf, W. Tavernier, D. Colle, M. Pickavet, and P. Demeester, “Link failure recovery technique for greedy routing in the hyperbolic plane,” *Comput. Commun.*, vol.36, no.6, pp.698–707, 2013.
 - [26] S. Sahhaf, W. Tavernier, D. Colle, M. Pickavet, and P. Demeester, “Single failure resiliency in greedy routing,” 2013 9th IEEE International Conference on the Design of Reliable Communication Networks (DRCN), pp.306–313, 2013.

Sahel Sahhaf received a M.Sc. degree in Information Technology from Sharif University of Technology (SUT), Tehran, Iran in 2010. Since then she joined the Internet Based Communication Networks and Services (IBCN) research group at Ghent University — iMinds as a Ph.D. student. Her research focuses on novel network architectures for next generation Internet (Greedy router architecture).

Wouter Tavernier received a M.Sc. in Computer Science in 2002 from Ghent University (Belgium) and a B.Sc. degree in Philosophy in 2003 from K.U.Leuven (Belgium). After a two-year period as business analyst at Accenture Belgium, he joined the Internet Based Communication Networks and services group of Ghent University in 2005 to research future internet-topics including resiliency of (Carrier) Ethernet and IP networks, geometric routing and the application of (machine) learning techniques in the context of routing and switching. This research has lead to a Ph.D. degree in Computer Science Engineering from Ghent University (Belgium) in 2012. Current research focuses on Software-Defined Networks and Service Function Chaining in the context of national and European research projects such as UNIFY.

Didier Colle received a M.Sc. degree in Electrotechnical engineering (option: communications) from Ghent University in 1997. Since then, he has been working at the same university as researcher in the department of Information Technology (INTEC). He is part of the research group Internet Based Communication Networks and Services (IBCN) headed by prof. Piet Demeester. His research lead to a Ph.D. degree in February 2002. He became professor at Ghent University in October 2011. Currently, he is co-responsible for the research cluster on network modeling, design and evaluation and is coordinating the research on fixed internet architectures and optical networks.

Mario Pickavet is professor at Ghent University where he is teaching courses on discrete mathematics, broadband networks and network modeling. He is leading research tracks on fixed internet architectures and optical networks, green ICT and design of network algorithms. He has published about 300 international publications, both in journals (IEEE JSAC, IEEE Comm. Mag., Journal of Lightwave Technology, Proceedings of the IEEE, ...) and in proceedings of conferences. He is co-author of the book ‘Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS’.

Piet Demeester is professor in the Faculty of Engineering at Ghent University. He is head of the research group “Internet Based Communication Networks and Services” (IBCN) that is part of the Department of Information Technology (INTEC) of Ghent University. He is also leading the Future Internet (Networks, Media and Service) Department of the Interdisciplinary Institute for Broadband Technology (iMinds). He is Fellow of the IEEE.