Concurrent Geometric Multicasting

Jordan Adamek, Mikhail Nesterenko, James Scott Robinson, Sébastien Tixeuil

▶ **To cite this version:**

Jordan Adamek, Mikhail Nesterenko, James Scott Robinson, Sébastien Tixeuil. Concurrent Geometric Multicasting. [Research Report] UPMC Sorbonne Universités. 2017. hal-01540744

HAL Id: hal-01540744

https://hal.sorbonne-universite.fr/hal-01540744

Submitted on 16 Jun 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Concurrent Geometric Multicasting

Jordan Adamek* Mikhail Nesterenko* James Scott Robinson*
Sébastien Tixeuil*

*Kent State University
*UPMC Sorbonne Universités & IUF

Abstract

We present $\mathcal{MCFR}$, a multicasting concurrent face routing algorithm that uses geometric routing to deliver a message from source to multiple targets. We describe the algorithm's operation, prove it correct, estimate its performance bounds and evaluate its performance using simulation. Our estimate shows that $\mathcal{MCFR}$ is the first geometric multicast routing algorithm whose message delivery latency is independent of network size and only proportional to the distance between the source and the targets. Our simulation indicates that $\mathcal{MCFR}$ has significantly better reliability than existing algorithms.

1 Introduction

Geometric routing. Geometric routing is transmitting a message from the sender to the targets on the basis of the geometric locations of the nodes. Geometric routing may offer a number of attractive features: it does not require nodes to maintain routing information beyond its immediate neighborhood; it can be stateless when no information is retained at the node as it forwards the message; its message size can be kept constant as each message carries limited amount of data, independent of network size.

Unicasting face routing. In *unicasting*, a single source sends a message to a single target whose coordinates are known to the source. The simplest form of unicast geometric routing is greedy. In *greedy routing* [10], each intermediate node forwards the message to its neighbor that is the closest to the target. Pure greedy routing fails if the message encounters *local minimum*: a node that does not have neighbors closer to the target. A *sequential* geometric routing algorithm, such as the classic GFG/GPSR [3, 15], routes a single message in the *greedy mode* until a local minimum is encountered. The algorithm then switches to *recovery mode* which involves traversing the faces of a planar subgraph of the original communication graph. Specifically, the algorithm traverses the faces that intersect the line that connects the source and the target.

There are several disadvantages of sequential face traversal of GFG and similar algorithms. A sequential algorithm cannot predict which traversal direction results in shorter

distance. Hence, in the worst case, the latency of message delivery of such algorithms is proportional to the network size. Furthermore, such algorithms have low reliability: a single transmission fault results in complete message delivery failure. Last, sequential algorithms are unable to determine if the target is disconnected from the source, possibly resulting in the message remaining in the network arbitrarily long. A *concurrent* unicast algorithm CFR [6] sends a pair of messages to traverse the source-target line concurrently. This naturally selects the shortest face traversal direction. Moreover, the second message provides greater robustness in case of message loss. The meeting of these messages signifies the end of face traversal and automatically determines whether the target is connected to the source. These advantages are offset by greater message cost.

Multicasting. Geometric multicasting requires the source to transmit the same message to several targets. The source knows the coordinates of the targets. Unicasting to each target separately may be inefficient if several targets are located near each other since multiple redundant messages are sent to similar locations. Efficient multicasting optimizes the routes the messages take so that a single message is routed to multiple targets as long as possible. LGS [5] computes a minimum Euclidean length spanning tree rooted in the source and containing all the targets. LGS then uses geometric unicast to transmit the messages along the edges of this tree. More sophisticated schemes of PBM [19] and GMP [23] compute Euclidean Steiner tree[1] at every intermediate node and then unicast the message along this tree. This scheme improves efficiency since the total length of Steiner tree edges may only be half that of the minimum spanning tree [14]. In both GMP and PBM, the message is geometric unicast towards the node, real or virtual, that roots the subtree of the target destinations. As the message approaches a virtual node, it may be advantageous to *split* the message and route separate messages towards different destination groups. GMP is shown to be more efficient than PBM in route selection and Steiner tree computation [23]. Overall, all these multicasting algorithms use sequential face routing to recover from a local minimum. Hence, they are prone to inefficient route selection, message loss and inability to detect network disconnection.

Our contribution. In this paper, we present $\mathcal{MCFR}$: the first concurrent multicasting face routing algorithm. The algorithm computes Steiner tree of the targets and then concurrently routes the message around all the faces that intersect this Steiner tree. We prove it correct, provide asymptotic measures of its latency and message complexity and then evaluate its performance through simulation. We show that, unlike sequential multicasting algorithms, the latency of $\mathcal{MCFR}$ is independent of network size. It only depends on the distance between the source and the target. Our simulation shows that due to concurrency, the delivery ratio of $\mathcal{MCFR}$ significantly exceeds that of sequential algorithms.

[1] A Euclidean Steiner tree is a minimum length spanning tree with virtual *Steiner* nodes that connects the source and the destinations.

2 Notation and Definitions

Wireless network, message and node limitations. A *wireless network* $G = (N, E)$ is represented as a graph where N is a set of nodes that are devices capable of exchanging messages wirelessly, while E is a set of edges connecting the nodes if the two adjacent nodes can send messages directly. Two such nodes are *neighbors*. The communication is bi-directional and the graph is undirected. Every node has unique planar coordinates that *embeds* the graph into the geometric plane. When it is clear from the context, we refer to a graph's embedding as just graph.

Planarity, face traversal, Steiner tree. A graph embedding is *planar* if the graph edges intersect only at vertices. For short, we call this planar embedding of a graph, a *planar graph*. A *connected planar subgraph* is a subset of vertices and their induced edges such that the resultant graph is planar and connected. Finding a maximum planar subgraph of a general graph is NP-hard [18]. However, for certain graphs, the task may be solved efficiently. A graph is *unit-disk* if a pair of its vertices u and v are neighbors if and only if the Euclidean distance between them is no more than 1. Such a graph approximates a wireless network. In such a graph, a connected planar subgraph may be found by local computation at every node using Relative Neighborhood or Gabriel Graph [3, 12, 15, 21]. *Face* is a region of the plane such any two points of the region may be connected by a continuous curve that that does not intersect the edges of the graph. A planar embedding of a finite graph divides the plane into a finite set of faces. The areas of all but one of the faces are finite. The finite area faces are *internal*. The infinite face is *external*. For example, in Figure 2, the graph has three internal faces: F, G and H and the external face.

Consider node u and its neighbors v and w. Node w is *next-right* after v, if it is the next neighbor of u after v clockwise; it is *next-left* after v, if it is next to it counter-clockwise. For example, in Figure 2, f is next-right neighbor of i after s. Observe that if w is next-left after v, then v next-right after w. Node u, its two neighbors v and w and the two incident edges form *angle* $\angle vuw$. An angle *intersects* a segment of a line if at least one of its edges lies on or intersects this segment. For example, $\angle sif$ intersects segment sx. Note that we limit angle intersection to the fixed-size graph edges, not the infinite half-rays of a classic geometric angle. In a planar graph, to traverse a face, messages are routed using right- or left-hand-rule. In the *right-hand-rule*, if a node receives a message, it forwards the message to the neighbor that is next-right after the sender. In the *left-hand-rule*, the message is forwarded to the next-left neighbor. For example, in Figure 2, if i receives a right-hand-rule traversal message from s, then it forwards it to f. Two messages are *mates* if they are traversing the same face in the opposite directions. A single node is able to detect mates if the sender of each message is the receiver of the other and the traversal direction of the two messages is opposite. A *Euclidean minimum Steiner tree* connects a selected set of nodes on a plane, possibly with added *virtual nodes*, by a graph of minimum total length. The problem of computing such tree is NP-hard [13]. GeoSteiner is the most successful algorithm that computes the exact solution in reasonable time [22]. Efficient polynomial approximations are also available [1, 20]. For the rest of the paper, we refer to the Euclidean minimum Steiner tree as just Steiner tree and ignore the fact that it is being approximated.

Message and node memory constraints. To help with routing, a message carries routing information. Only *constant size messages* are allowed. This means that the message may carry only a fixed number of node coordinates and related information. This fixed number is independent of the network size. This limitation, for example, precludes a routing algorithm from requesting the message to carry its entire route. Each message always carries the immediate sender, the node that transmitting this message, and immediate receiver, the node the message is being sent to. Each node stores the coordinates of its neighbors. No other information, either temporarily or permanently, can be stored by the node. This limitation precludes nodes from maintaining extensive routing tables of the network or storing state information between message transmissions.

Steps, computations, fairness, multicasting. Every node has a *send queue* SQ that collects messages to be sent. A message is transmitted by taking it from the sender's send queue, transferring it to the receiver and processing it according to the routing algorithm. In the theoretical discussion about the algorithm, we assume that this transferal and processing is done in a single atomic *step*. The *atomicity* of the step means that it may not overlap with the steps of this or any other nodes. In the simulation section, this assumption is lifted. *Computation* is a sequence of atomic steps that starts in an initial state of the algorithm. A computation is *fair* if each message, in the send queue of every node, is either transmitted or removed from this queue. That is, a message may not "get stuck" in a send queue forever. We consider only fair computations. A computation with a finite number of steps is itself *finite*. A routing algorithm is *terminating* if its every computation is finite. A terminating routing algorithm never leaves messages circulating in the network indefinitely. A *multicasting routing algorithm* ensures a message is delivered from the *source* to the set of *targets*. The source knows the coordinates of the targets and these coordinates fit in single message. For example, in Figure 2, source s may need to send a messages to targets b, d and k. To aid in navigation, a multicasting algorithm may compute a Steiner tree that includes source, targets and virtual nodes x and y.

3 Algorithm Description

The pseudocode of algorithm $\mathcal{MCFR}$ is shown in Figure 1. It operates as follows. The sender computes the Steiner tree T of targets and itself. For each angle that intersects T, it sends a pair of mates with right R and left L traversal directions. Every message carries the encoding of T. For simplicity, we assume that source itself is never a target.

Once some node n receives a message, it checks if there is its mate in its send queue SQ. If the mate is found, both messages are removed and further processing stops as this completes the traversal of a face. If there is no mate, n checks if it is the target and delivers the message. After the delivery, message processing continues. Specifically, the message is forwarded along the face that it traverses. A node is *juncture* if at least one of its angles intersects T. If n is a juncture node then n *splits* the message by injecting a pair of mates in every angle that intersects T. Observe that the source node is always a juncture.

Let us consider an example of $\mathcal{MCFR}$ operation shown in Figure 2. Source node s

```
node s
compute T    /* T is Steiner tree */
foreach ∠asb that intersects T do
    add L(s,T,a) to SQ
    add R(s,T,b) to SQ

node n
if receive L(s,T,a) then
    if R(s,T,a) ∈ SQ then
        /* found mate */
        discard R(s,T,a) from SQ
    else
        if n ∈ T then
            deliver L(s,T,a) to n
        /* let b be the next left after a */
        add L(s,T,b) to SQ
        if ∠anb intersects T then
            /* split message */
            foreach ∠cnd ≠ ∠anb that intersects T do
                /* let d be the next left after c */
                add R(s,T,c) to SQ
                add L(s,T,d) to SQ
if receive R(s,T,a) then
    /* handle similar to L(s,T,a) */
```

Figure 1: $\mathcal{MCFR}$ pseudocode.

computes the Steiner tree T, determines that $\angle isa$ intersects T and sends a left-hand-rule L traversing message to a, while sending R to i. This injects a pair of mates into face F. When a receives a message, it forwards it to its next-left left after s neighbor b. When i receives a message, it determines that i is a juncture node and the angle that intersects T is $\angle fij$. Node i injects a pair of mates by sending an L to f and an R to j. Meanwhile, i also forwards the originally received message to f. Node f injects mates into faces H and G and forwards the original message to b. Node b injects messages into face G and forwards the message to a. However, then b receives a message from a. Once b receives a message from a, it inspects its SQ, finds its mate and destroys both messages. This completes the traversal of face F. The operation of $\mathcal{MCFR}$ continues until all faces that intersect T are traversed.

4 Correctness Proof and Efficiency Bounds

Correctness proof. Let us introduce some notation to aid in the correctness discussion. A node is *segment-visited*, or just *visited*, with respect to a particular face if it was visited during the traversal of this face. It is *unvisited* otherwise. A *visited segment* of a face is a sequence of neighbor nodes that are visited. A *segment-border* of a visited segment, with respect to a particular face, is a visited node with a neighbor that is either (i) unvisited or (ii) carrying a message for the border node. A non-border visited node is *segment-internal*. Note that an edge in a planar graph is adjacent to two faces. Thus, a node may be visited in one face but unvisited in another. Two faces are *adjacent* if they share a common juncture node. Two faces F and G are *juncture connected* if there exists a sequence of adjacent faces

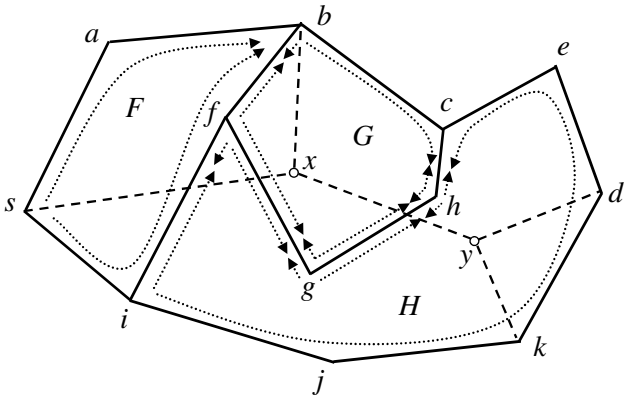


Figure 2: $\mathcal{MCFR}$ operation example.

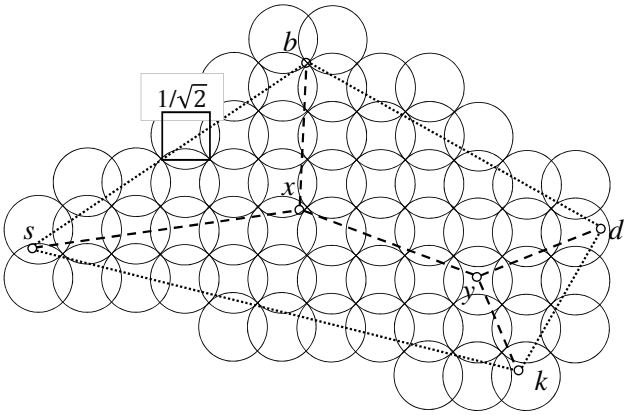


Figure 3: Covering the source-target Steiner hull with unit circles. Illustration for the proof of Theorem 3.

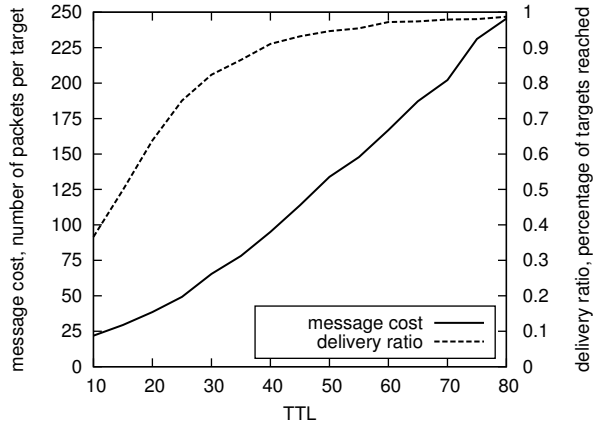
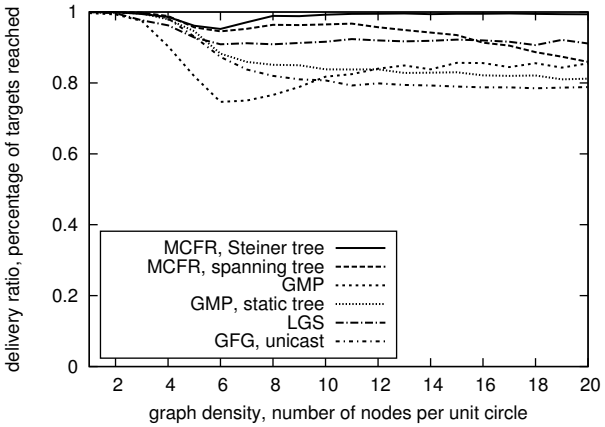
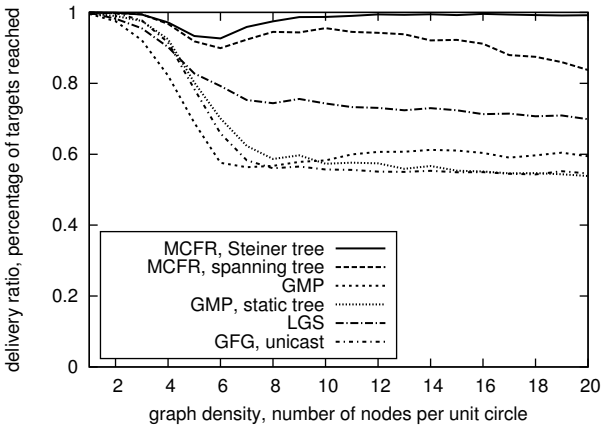


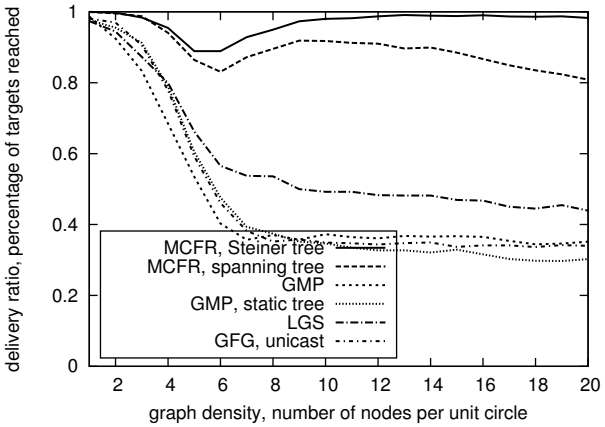
Figure 4: Time to live evaluation for MCFR, Steiner Tree, 15 dBm.



(a) 15 dBm



(b) 7 dBm



(c) 0 dBm

Figure 5: Delivery ratio.

that starts in F and ends in G. Observe that by the design of the algorithm, once the juncture is visited, it splits the message in every angle that intersects T. That is, a juncture is visited in every adjacent face at once.

Lemma 1 *In $\mathcal{MCFR}$, for every face F with a visited segment, the segment border node has a message to send across to the unvisited neighbor. A segment-internal node never holds such a message.*

Proof: The proof is by induction on the nodes of a particular face F. A visited segment is created in F when a juncture node is visited. This juncture may be the source or another node splitting the message when it is visited in an adjacent face. Once the visited segment is created, it contains a single border node with two messages sent in the opposite directions. This is our base case.

Let us consider a computation of $\mathcal{MCFR}$ where every visited segment of every face is as stated in the conditions of the lemma. First, let us consider a message transmission by node u adjacent to face F. By the induction hypothesis, u may only be a border node. The message recipient v may be an unvisited node or a visited node that has a message for u. Let us consider the unvisited case first. Once unvisited node v receives a message from u, v becomes a new border node with this message while v becomes segment internal without a message. Hence the conditions of the lemma hold. If v is visited, then by definition of the border node, it holds a mate for the message that v transmits. Moreover, by the condition of the lemma, v is also a border node of an adjacent segment. Once v receives the message from u, both mates are destroyed and both u and v become segment-internal nodes. This also preservers the condition of the lemma.

Let us now discuss the transmission of a message by a node u that is not adjacent to F. The only way that it may affect F is if v is a juncture adjacent to F. However, by the design of the algorithm, the juncture is instantly visited in every adjacent face. That is, when v receives a message transmission, it is not visited. Once it receives a message, it creates a new visited segment in F with a single border node v and appropriate outgoing messages.

That is, regardless of the message transmissions, the conditions of the lemma are preserved. □

Lemma 2 *In $\mathcal{MCFR}$, if a face has a visited segment, every node adjacent to this face is eventually visited and none holds messages.*

Proof: If a face with a visited segment contains an unvisited node, then, at least one such unvisited node is adjacent to a border of a visited segment. Due to Lemma 1, this border node has a message to be sent to the unvisited adjacent node. Since we only consider fair computations of the routing algorithm, this message is eventually transmitted. Once the message is sent, the adjacent node becomes visited. This process continues until all nodes of the face are visited. Once all nodes are visited they become internal and, according Lemma 1, do not hold message. Hence the lemma. □

Lemma 3 *In $\mathcal{MCFR}$, if a face intersects the Steiner tree T, then every node adjacent to this face is eventually visited.*

Proof: Consider the face that contains the source node s. The algorithm starts by creating a single-node visited segment there. According to Lemma 2, every node in this face is eventually visited. This includes all junctures adjacent to this face. Repeated application of Lemma 2 proves this lemma. □

Proposition 1 *If a target node is connected to the source node, then this node lies on a face that is juncture connected to the source node face.*

The below theorem follows from Proposition 1 and Lemma 3.

Theorem 1 *Algorithm $\mathcal{MCFR}$ guarantees termination and delivery of the message from the source to all targets connected to the source.*

Efficiency bounds. *Latency* of an algorithm is the shortest path that the message may take to reach the target. In multicasting, the latency is the longest such path among all targets. For latency estimation, following Kuhn et al [16], we assume that the network graph G is bounded degree since such graph can be efficiently obtained in a unit-disk graph by computing a connected-dominating set of G. The latency of a unicast concurrent face routing algorithm is established to be $O(t^2)$ where t is the distance between the source and the target [6, Theorem 2].

Theorem 2 *$\mathcal{MCFR}$ latency is in $O(d^2)$ where d is the Steiner tree diameter.*

Proof: Let d be the diameter of the Steiner tree and m is the number of multicast targets. A Euclidean Steiner tree has at most $m-2$ virtual nodes [14]. In the worst case, the multicast message in $\mathcal{MCFR}$ has to sequentially reach all nodes in the Steiner tree. That is, the total latency is $O((2m-2)d^2)$ which is $O(md^2)$. The theorem's claim follows if the number of targets is constant. □

Message cost of an algorithm is the total number of messages expended in delivering it to the targets. In the worst case, $\mathcal{MCFR}$ traverses every face of the graph. An edge is adjacent to two faces, hence $\mathcal{MCFR}$ may send up to $2|E|$ messages where E is the number of edges in the graph. However, for most graphs, $\mathcal{MCFR}$ is a lot more efficient. To give a more realistic message cost estimate of $\mathcal{MCFR}$ we make several assumptions about the network graphs.

The graph is *face smooth* if there are two constants c_1 and c_2 that are independent of network parameters such that (i) for each face $\rho^2 < c_1 a$ where ρ is the perimeter of the face, and a its area, and (ii) for any two points in the graph, $a_s < c_2 \frac{\pi d^2}{4}$ where a_s is the area of all internal faces that intersect the line between these two points, and d is the Euclidean distance between them. For an internal face, the area computation is straightforward; for the external face, an area of an arbitrary figure enclosing the graph, for example a convex hull, is considered. The first assumption places limits on how "ragged" the perimeter of the face may be, while the second limits how "uneven" the faces may be in size by assuming that the area of all intersecting faces is included in a certain disk whose diameter is related to the distance

between two devices. These assumptions hold for most realistic wireless communication graphs such as unit-disk graphs. *Steiner hull* is the convex hull that contains the nodes of the Steiner tree. It is known that virtual nodes are internal to the Steiner hull [14].

Theorem 3 *For face smooth graphs, the message cost for $\mathcal{MCFR}$ is in $O(|H| + \sqrt{|G|})$, where $|H|$ is the area of the Steiner hull and $|A|$ is the area of the complete graph G.*

Proof: We completely cover the Steiner hull H with unit-disks. See Figure 3 for illustration. In this arrangement, each unit disk covers a square with side length of $\frac{1}{\sqrt{2}}$. Since k is the maximum node degree, the number of nodes in each unit disk is no more than k. Hence, the number of nodes inside H is: $k\frac{|H|}{(\frac{1}{\sqrt{2}})^2} = 2k|H|$.

Since there are at most k neighbors, each node may be adjacent to at most k edges. A message may be sent across each edge at most twice. Hence, the number messages to be sent inside $|H|$ is $4k^2|H|$ which is in $O(|H|)$.

Let us know estimate the number of messages it takes to traverse all the faces that intersect H. Any convex polygon can be inscribed into a rectangle whose area at most twice the size of that of the polygon [17]. This means that H can be inscribed into a rectangle whose area is at most $2|H|$. The perimeter of this rectangle is $4\sqrt{2|H|}$. We assume that the graph is face-smooth. Combining the two face smoothness conditions we obtain that the sum of the perimeters of all the internal faces that intersect H has this relation:

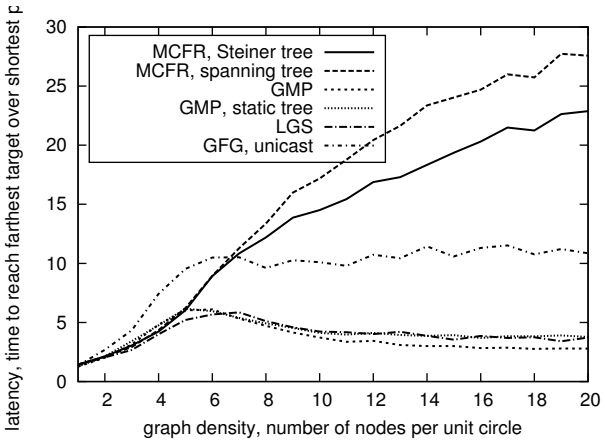
$$p_s^2 = c_1 c_2 \frac{\pi(4\sqrt{2|H|})^2}{4} = c_1 c_2 \pi 4\sqrt{2}|H|,$$

which means that p_s is in $O(\sqrt{|H|})$. Similarly, the perimeter of the external face is in $O(\sqrt{|G|})$. Since each face is traversed at most once, the total number of messages used to traverse internal faces that intersect H as well as the external face is in $O(\sqrt{|H|} + \sqrt{|A|})$. Combined with the number of messages needed to traverse faces inside the Steiner hull we get $O(|H| + \sqrt{|H|} + \sqrt{|G|}) = O(|H| + \sqrt{|G|})$. □

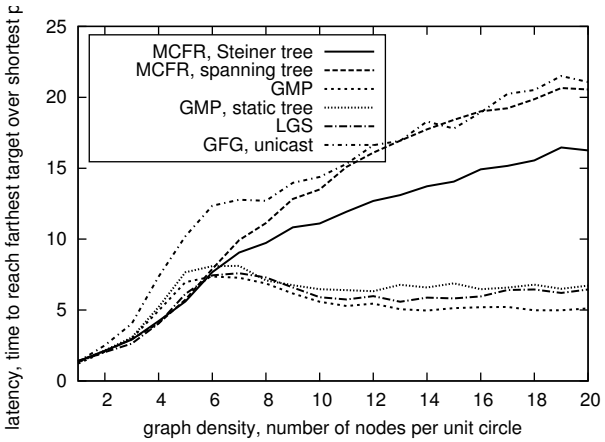
To summarize, Theorem 2 shows that the latency of message delivery of $\mathcal{MCFR}$ does not depend on the overall network size, just on the distance between the source and the targets; Theorem 3 shows that the total number of messages sent by $\mathcal{MCFR}$ depends on the locations of the targets with respect to the source and the length of the external face of the graph.

5 Simulation

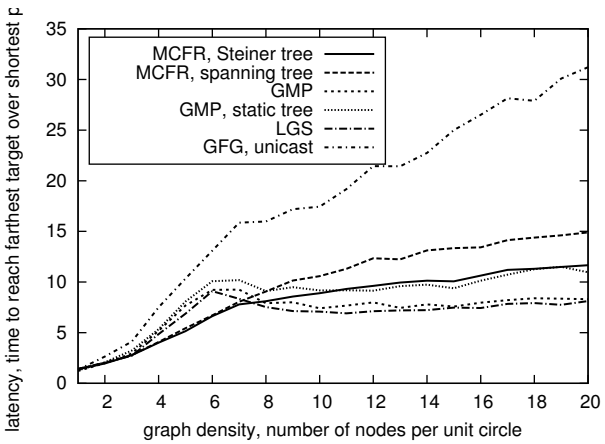
Setup. To evaluate the performance of our algorithms, we implement them in WSNet [2, 8, 9, 11] wireless sensor network simulator. The simulated MAC layer is IEEE 802.15.4 with 866 MHz frequency band and BPSK modulation. The ratio model is freespace propagation with constant path loss and rayleigh fading [2]. In our geometric simulation setup, we recreate and extend the setup of the classic geometric unicast algorithm study of Kuhn et al [16].



(a) 15 dBm

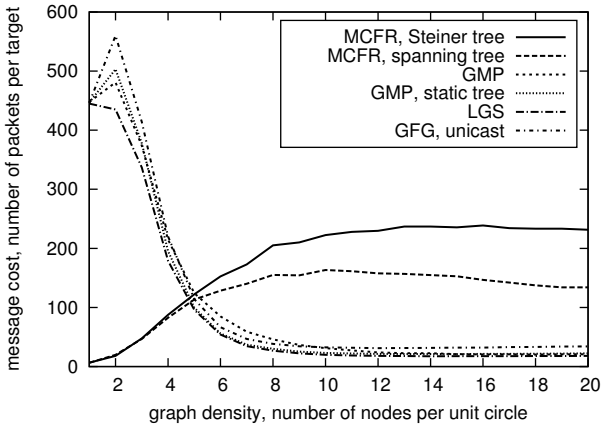


(b) 7 dBm

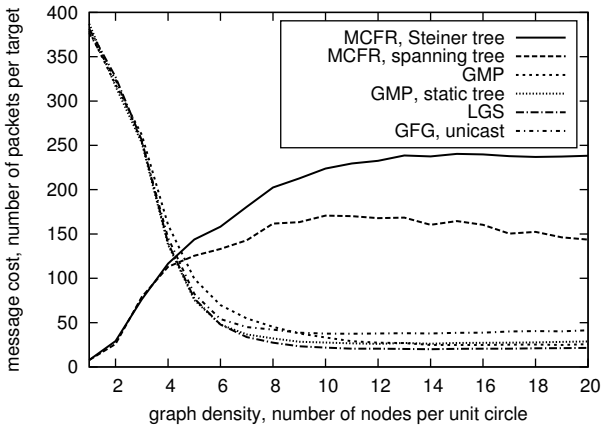


(c) 0 dBm

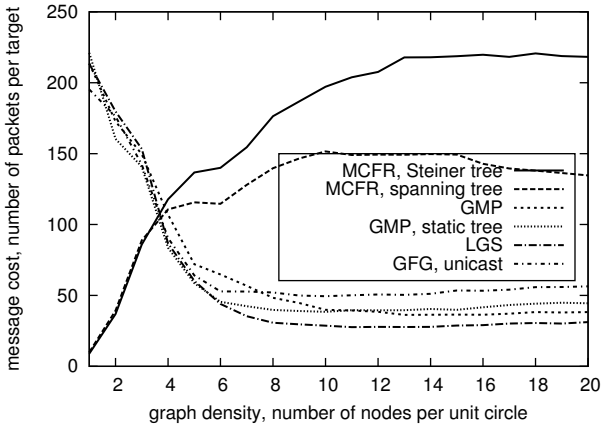
Figure 6: Latency.



(a) 15 dBm



(b) 7 dBm



(c) 0 dBm

Figure 7: Message cost.

Specifically, we simulate 1000×1000 meters field. The unit is 100 meters. The field is populated by nodes placed uniformly at random to achieve a specific network density. For a fixed density, the number of nodes is calculated as follows. The total number of nodes n is equal to the area of the field divided by the area of the unit circle and multiplied by the required density d. That is $n = d\frac{1000\times1000}{\pi 100^2}$. To compute the network topology we use a unit-disk graph, then compute a Gabriel subgraph over it. The topology is calculated offline. We evaluate our algorithms' performance at three power levels: 15 dBm, 7 dBm and 0 dBm. The weaker the signal, the less reliable message transmissions are.

Experiment is a single delivery of a message from a particular source to a particular set of targets. In other words, it is a single complete computation of an algorithm. For each experiment, we generate a new random graph with randomly selected source and targets. The number of targets is selected to be 5% of the total number of nodes. For each specific data point we conduct 1000 experiments and compute the average value.

We evaluate the performance of our algorithms according to three metrics. *Delivery ratio* is the number of targets that receive the message divided by the total number of targets. Delivery ratio determines the reliability of the algorithm. *Latency* is the the time it takes the algorithm to deliver the message to the target that is geometrically furthest away from the source divided by the time it takes to unicast this message to this target using an optimum route. Latency determines the algorithm's speed of message delivery. *Message cost* is the number of message transmissions divided by the number of targets. Since a message transmission is a radio broadcast, message broadcast to several neighbors is counted as a single transmission. The message cost counts all message transmissions regardless of delivery success. Raw latency and message cost do not take into account delivery success. For example, an algorithm that delivers only to the nearest target may have low latency and message cost. To offset that, we divide the latency and message cost by the delivery ratio.

Algorithms. We simulate both sequential and concurrent multicasting algorithms. For sequential algorithms, as a baseline, we simulate the trivial algorithm that uses a unicast GFG to separately deliver the message to each individual target. We simulate LGS. We simulate GMP as presented in the original paper [23] as well as a variant of the algorithm where the Steiner tree is computed once at the source and is not recomputed by every intermediate node. For concurrent multicasting, we simulate two versions of MCFR. In the first version, MCFR navigates over Steiner tree; in the second, it routes over minimum Euclidean spanning tree. The trees are computed by the source.

In realistic environments, both concurrent and sequential algorithms have termination issues. If messages are lost, MCFR packets may not find mates. In this case, the messages may traverse graph faces indefinitely. On the other hand, a sequential algorithm does not detect if a target is disconnected. Again, a message for such disconnected target never reaches its destination. To force termination, we introduce time to live (TTL) for each message. A message is discarded after its TTL expires. To determine optimal TTL, we vary TTL then compute message cost and delivery ratio for MCFR with Steiner Tree and 15 dBm signal strength. The results are shown in Figure 4. The TTL of 55 hops seems to produce the best performance. For the rest of the experiments all algorithms have the TTL of 55.

Results and analysis. The simulation results for delivery ratio, latency and message cost of the simulated algorithms are shown in Figures 5, 6 and 7 respectively. Let us discuss the results. The reliability of MCFR exceeds that of sequential multicasting algorithms. The gap widens as signal strength lowers and message loss increases. For 0 dBm, the delivery ratio of MCFR Steiner Tree never drops below 90% while the delivery ratio of most of the sequential algorithms goes below 40%. Interestingly, the reliability of simple unicasting to all targets exceeds that of GMP and LGS. This is due to the difference in their operation. To optimize message cost, GMP and LGS combine the messages to multiple targets as long as possible. However, a loss of such combined message results in failed delivery to all targets.

The latency of concurrent algorithms is higher than that of sequential algorithms. This is due to the greater number of messages contending for radio channel access. However, with lower signal strength and greater message loss, the gap narrows. For 0 dBm, latency of concurrent and sequential algorithms is similar. The higher delivery ratio of concurrent algorithms match relative speed of delivery of the sequential algorithms. The only exception is the unicast GFG whose latency continues to be high.

Concurrent and sequential algorithms exhibit different message cost dynamics. At lower densities greater number of targets is disconnected. This forces sequential algorithms to send aimless messages to wander around the network until their TTL expires. Concurrent algorithms do not have this issue. As the network density grows, the relative frugality of the sequential algorithms gives them message cost advantage over concurrent algorithms.

6 Conclusion

Algorithm $\mathcal{MCFR}$ presented in this paper provides good practical reliability and asymptotic latency that depends only on the Steiner tree diameter. As future work, we suggest the following. To further improve latency, apply this algorithm to minimum diameter Steiner trees [7]. Evaluate fine-grained energy consumption of the algorihtms, for example using the approach of Bramas et al [4], as it may impact the survivability of strategic parts of the network.

References

[1] Sanjeev Arora. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *Journal of the ACM (JACM)*, 45(5):753–782, 1998.

[2] Elyes Ben Hamida, Guillaume Chelius, and Jean-Marie Gorce. On the complexity of an accurate and precise performance evaluation of wireless networks using simulations. In *Proceedings of the 11th international symposium on Modeling, analysis and simulation of wireless and mobile systems*, pages 395–402. ACM, 2008.

[3] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *The Journal of Mobile Communication, Computation and Information*, 7(6):48–55, 2001.

[4] Quentin Bramas and Sébastien Tixeuil. Benchmarking energy-centric broadcast protocols in wireless sensor networks. In Parosh Aziz Abdulla and Carole Delporte-Gallet, editors, *Networked Systems - 4th International Conference, NETYS 2016, Marrakech, Morocco, May 18-20, 2016, Revised Selected Papers*, volume 9944 of *Lecture Notes in Computer Science*, pages 87–101. Springer, 2016.

[5] Kai Chen and Klara Nahrstedt. Effective location-guided tree construction algorithms for small group multicast in MANET. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Society (INFOCOM-02)*, volume 3 of *Proceedings IEEE INFOCOM 2002*, pages 1180–1189, Piscataway, NJ, USA, June 23–27 2002. IEEE Computer Society.

[6] Thomas Clouser, Mark Miyashita, and Mikhail Nesterenko. Concurrent face traversal for efficient geometric routing. *Journal of Parallel and Distributed Computing*, 72(5):627–636, 2012.

[7] Wei Ding and Ke Qiu. Algorithms for the minimum diameter terminal steiner tree problem. *Journal of Combinatorial Optimization*, 28(4):837–853, 2014.

[8] Tony Ducrocq, Michaël Hauspie, and Nathalie Mitton. Geographic routing with partial position information. In Octavian Postolache, Marten van Sinderen, Falah H. Ali, and César Benavente-Peces, editors, *SENSORNETS 2014 - Proceedings of the 3rd International Conference on Sensor Networks, Lisbon, Portugal, 7 - 9 January, 2014*, pages 165–172. SciTePress, 2014.

[9] Tony Ducrocq, Michaël Hauspie, Nathalie Mitton, and Sara Pizzi. On the impact of network topology on wireless sensor networks performances: Illustration with geographic routing. In Leonard Barolli, Kin Fun Li, Tomoya Enokido, Fatos Xhafa, and Makoto Takizawa, editors, *28th International Conference on Advanced Information Networking and Applications Workshops, AINA 2014 Workshops, Victoria, BC, Canada, May 13-16, 2014*, pages 719–724. IEEE Computer Society, 2014.

[10] G.G. Finn. Routing and addressing problems in large metropolitan-scale internetworks. Technical Report ISI/RR-87-180, March 1987.

[11] Antoine Fraboulet, Guillaume Chelius, and Eric Fleury. Worldsens: development and prototyping tools for application specific wireless sensors networks. In *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*, pages 176–185. IEEE, 2007.

[12] K Ruben Gabriel and Robert R Sokal. A new statistical approach to geographic variation analysis. *Systematic Biology*, 18(3):259–278, 1969.

[13] Michael R Gary and David S Johnson. Computers and intractability: A guide to the theory of np-completeness, 1979.

[14] F.K. Hwang, D.S. Richards, and P. Winter. *The Steiner Tree Problem*, volume 53 of *Annals of Discrete Mathematics*. North-Holland: Elsevier, 1992.

[15] B. Karp and H.T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobille Computing and Networking (MobiCom)*, pages 243–254. ACM Press, August 2000.

[16] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. Geometric ad-hoc routing: Of theory and practice. *22nd ACM Symposium on the Principles of Distributed Computing (PODC)*, July 2003.

[17] Marek Lassak. Approximation of convex bodies by rectangles. *Geometriae Dedicata*, 47(1):111–117, 1993.

[18] P. C. Liu and R. C. Geldmacher. On the deletion of nonplanar edges of a graph. In *Proc. of the 10th Southeastern Conference on Combinatorics, Graph Theory, and Computing*, pages 727–738, 1979.

[19] Martin Mauve, Holger Füßler, Jörg Widmer, and Thomas Lang. Position-based multicast routing for mobile ad-hoc networks. *Mobile Computing and Communications Review*, 7(3):53–55, 2003.

[20] Gabriel Robins and Alexander Zelikovsky. Improved steiner tree approximation in graphs. In *SODA*, pages 770–779. Citeseer, 2000.

[21] Godfried T Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern recognition*, 12(4):261–268, 1980.

[22] David M Warme, Pawel Winter, and Martin Zachariasen. Exact solutions to large-scale plane steiner tree problems. In *Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms*, pages 979–980. Society for Industrial and Applied Mathematics, 1999.

[23] Shibo Wu and K. Selcuk Candan. GMP: Distributed geographic multicast routing in wireless sensor networks. In *26th IEEE International Conference on Distributed Computing Systems (26th ICDCS'06)*, page 49, Lisboa, Portugal, July 2006. IEEE Computer Society.