First publ. in: Proceedings of the 11th International Conference on Information Visualization (IV ´07), 2007, pp. 263-270

Geographic Routing on Improved Coordinates

Ulrik Brandes, Daniel Fleischer
Department of Computer & Information Science, University of Konstanz
Ulrik.Brandes@uni-konstanz.de, Daniel.Fleischer@uni-konstanz.de

Abstract

We consider routing methods for networks when geographic positions of nodes are available. Instead of using the original geographic coordinates, however, we precompute virtual coordinates using barycentric layout. Combined with simple geometric routing rules, this greatly reduces the lengths of routes and outperforms algorithms working on the original coordinates. Along with experimental results we proof properties such as guaranteed message delivery and worst-case optimality. Our methods apply to static networks in which short routes are important, but memory for full routing tables is not available and the one-time-precomputation is affordable.

1. Introduction

Routing in a communication network $G = (V, E)$ denotes the task of sending a message from a source node $s \in V$ to a target node $t \in V$. When no direct connection is available this means to forward the message on a path from s to t using intermediate nodes. While in wired networks the specific path is usually determined by routers, in *wireless networks* each node has to decide how to forward the message. Specific for geographic routing is the existence of geographical positions of the nodes. It is assumed that each node v knows the position of t, its own position and the positions of all its neighbors. This information can be used for the search of an s-t-path. Among the simplest routing algorithms are, e.g., Greedy Routing (see, e.g. [7]) and Compass Routing [4]. Greedy Routing always forwards the message from a node v to its neighbor w closest to t. Since w must be strictly closer to t than v, the method can get stuck in a *dead-lock*. Compass Routing chooses the neighbor with least (absolute) *deviation angle*, but again message delivery to t is not guaranteed. The first geographic routing algorithm with guaranteed delivery was Compass Routing II [4]. The currently most efficient algorithm is GOAFR+[5], which we therefore use as a reference. Our methods employ a mixture of Greedy and Compass Routing running not on the original geographic coordinates, but instead on precomputed *virtual coordinates*. This greatly reduces the lengths of routes and guarantees message delivery. Our choice of virtual coordinates is motivated by a theorem of W. Tutte [8] on *barycentric layouts*. The improvement of route lengths is mainly due to the following observation. The number of dead-locks is reduced significantly in planar barycentric layouts, since each face is convex. Hence, Greedy Routing, which heuristically delivers short routes, leads to t more often (see, e.g. [6]). Our routing methods – presented in detail in Section 3 – with a brief summary of their properties are:

- BR (Barycentric Routing): simple routing rules, good in practice for a certain density range of networks, guaranteed delivery
- GBR (Greedy BR): very short routes for all densities, outperforming routing methods on geographic coordinates, guaranteed delivery
- AGBR (Adaptive GBR): fixes worst-cases of GBR, guaranteed delivery
- AGBFR (AGB Face Routing): less consuming precomputation, delivery guaranteed after iteration (1) yields a planar embedding

Since proofs of theoretical results are omitted from this extended abstract because of space restrictions, we refer to the full paper for all details. Surveys on geographic routing and virtual coordinates can be found in [10] and [1].

2. Preliminaries

We consider wireless networks modeled as *unit disk graphs* $G = (V, E)$, whose nodes are embedded in the Euclidean plane $\mathbb{R}^2$ and two nodes are adjacent iff their distance is at most 1. We assume that G is connected and there are no two nodes at the very same position. *Geographic routing* uses these positions to route a message from a source node s to a target node t under the assumptions that the coordinates of t, the coordinates of all neighbors of

v and its own position are known to each node v. The path along which a message is routed is called *message path*. A mapping $p : V \longrightarrow \mathbb{R}^2$ is called *layout* or *embedding*. Let $\mathbb{R}^2$ be equipped with the common notions of open and closed sets. Each layout p of a graph G naturally corresponds to a closed set $\mathcal{G} \subset \mathbb{R}^2$ (we do not distinguish between G and $\mathcal{G}$), where the edges of G are drawn as straight lines between its incident nodes (not including them). We call an embedding p planar if its straight-line drawing $\mathcal{G}$ is planar. Instead of using the original geographic coordinates that will be denoted $\widehat{x}_v, \widehat{y}_v$, we compute *virtual coordinates* x_v, y_v and z_v during a precomputation phase of our routing methods. The Euclidean distance of two nodes v, w in original coordinates is $\widehat{d}(v, w)$, in virtual coordinates $d(v, w)$. (For distances in virtual coordinates the z-coordinate is always neglected.) The distance $d(e, t)$ of an edge is defined to be the infimum of distances $d(p_e, t)$ for all points $p_e \in e$. The angle $\varphi_t(v)$ is called *direction angle* to the target node t, where $\varphi_t(s)$ is defined to be 0 and for each further node v on a message path $\varphi_t(v)$ is increased or decreased according to the angle at t to the predecessor on the path. Thus, $\varphi_t(v)$ can have arbitrary real values and even have different values by multiples of 2π if the message path surrounds t and hits v again. The angle $\psi_t(v, w)$ is called *deviation angle* and denotes the angle at v from t to w and is defined to take values in $]-\pi, \pi]$. Note that all nodes w on the (open) right hand side of the (infinite) line $\overrightarrow{vt}$ have *negative* deviation angle. During the precomputation phase the restricted Gabriel Graph G_{GG} will be used. This is a planar embedded graph that is connected if the original unit disk graph is connected. Each node $v \in G_{GG}$ maintains an *ordered list* (counter-clockwise) of its neighbors. To pass a message *right hand rule* denotes to pass the message to the *next neighbor* in this list after the sender, if a sender exists. Otherwise, a direction will be given, and to pass a message right hand rule *directed to* $\overrightarrow{vw}$ denotes to take the next neighbor after $\overrightarrow{vw}$. Sometimes *virtual edges* are added to G_{GG}. These are edges that possibly do not exist in G and have to be realized as paths. However, by construction *virtual neighbors* can easily be reached by sending the message right hand rule or left hand rule, which has to be specified for each virtual edge. When adding a virtual edge, the two incident nodes simply update their lists. Since virtual edges are always inserted within a face of G_{GG} the planarity of G_{GG} is never destroyed (although the straight-line embedding of G_{GG} may then contain a crossing). For ease of simplicity we call a graph that is a subdivision (replacing an edge by edge-node-edge) of a 3-connected graph also 3-connected. Theorem 1 also holds for this class of graphs.

3. Barycentric Routing

In this section we introduce our routing methods BR, GBR, AGBR and AGBFR working on virtual coordinates computed during the precomputation phase described in Section 3.1. We need the following basics. With the definitions given in Section 2 the theorem of W. Tutte can be formulated as follows.

Theorem 1 ([8]) *Fixing the nodes of a face of a planar embedded, 3-connected graph onto the corners of a convex polygon C and setting the remaining nodes to the barycenter of their neighbors, yields a planar embedding.*

We call the layout obtained from Theorem 1 *barycentric layout* p_C. After fixing the nodes of C it is unique (for every connected graph G) and can be computed using the *Laplacian matrix* L of the given graph $G = (V, E)$. The layout $p_C = (x_v, y_v)_{v \in V}$ is then given by the unique solutions of $Lx = 0$ and $Ly = 0$, where the positions of the nodes of C are fixed and its corresponding lines in the equation systems deleted, see, e.g. [2]. These equations can iteratively be solved by the following Jacobi-iteration for all nodes $v \in V \setminus C$, see, e.g. [3].

$$x_v \leftarrow \sum_{w \in N(v)} \frac{x_w}{\deg(v)} \quad \text{and} \quad y_v \leftarrow \sum_{w \in N(v)} \frac{y_w}{\deg(v)} \qquad (1)$$

Note that iteration (1) only needs communication between adjacent nodes. An equivalent way to define p_C is the following, see, e.g. [2].

$$p_C = \arg\min_p \left\{ \sum_{v,w \in V} d(v, w)^2 \ : \ \text{nodes of } C \text{ fixed} \right\} \qquad (2)$$

Note that this formulation directly implies that all nodes $v \in V$ are within the closed set delimited by C.

3.1. Precomputation Phase

The precomputation phase for the following routing rules is divided into four steps that once have to be executed to obtain the desired virtual coordinates $(x_v, y_v, z_v)_{v \in V}$. After this precomputation phase messages can easily (with simple Routing Rules 1 to 4) and on very short paths in practice be routed (see Figure 1).

Step 1 computes the restricted Gabriel Graph G_{GG} and determines *tree children*, i.e. nodes that would be deleted if successively removing degree-one-nodes from G_{GG}.

Step 2 determines the nodes of the outer face of G_{GG} (i.e. the *perimeter nodes*) and sets them equidistantly on a circle enclosing G_{GG} to form the convex polygon C.

Step 3 establishes 2-connectedness of G_{GG} by adding *virtual edges* that do not destroy planarity of G_{GG}.

Step 4 establishes 3-connectedness (in the sense given in Section 2) of G_{GG}, such that Theorem 1 can be applied.

Algorithm 1: BR step 1 (determine tree nodes).

```
compute restricted Gabriel Graph $G_{GG}$ and declare all nodes as *non-tree nodes*
foreach $v$ with exactly one non-tree node neighbor do
    denote $v$ a *tree child*
repeat the last step (at most) $n$ rounds
foreach non-tree child $v$ with tree children do
    denote $v$ a *tree root*
foreach non-tree child $v$ do $z_v \leftarrow 0$
foreach tree root $v$ with children $v_1, \dots, v_k$ do
    send message $(z, z') := (i, i+1)$ to $v_i$
foreach tree child $v$ receiving $(z, z')$ do
    set $z_v \leftarrow z$ and send message
    $\big(z+(z'-z)\cdot i/(k+1), z+(z'-z)\cdot(i+1)/(k+1)\big)$
    to its children $v_i, 1 \le i \le k$
repeat the last step (at most) $n$ rounds
```

Algorithm 2: BR step 2 (set perimeter nodes).

```
let $M$ denote all nodes $v$ with minimum $\widehat{y}_v$ among their neighbors
foreach $v \in M$ do
    pass message $(\widehat{x}_v, \widehat{y}_v, c)$ containing the coordinates $\widehat{x}_v, \widehat{y}_v$ and a counter $c = 1$ *right hand rule* directed straight down
foreach $v \in V$ do
    if $v$ received message $(\widehat{x}, \widehat{y}, c)$ and $\widehat{y} \le \widehat{y}_v$ then
        set $c_v \leftarrow c$
        pass message $(\widehat{x}, \widehat{y}, c+1)$ right hand rule
    if $v$ received message $(\widehat{x}, \widehat{y}, c)$ and $\widehat{x} = \widehat{x}_v, \widehat{y} = \widehat{y}_v$ then
        pass message $(x_r, y_r, r, c)$ right hand rule directed straight down, where $x_r, y_r$ denotes the center of a circle of radius $r$ that encloses all nodes on the outer face (let therefore minimum and maximum $\widehat{x}, \widehat{y}$ values have been collected on the walk around the outer face)
        set $x_v \leftarrow x_r, y_v \leftarrow y_r - r$
    if $v$ received message $(x_r, y_r, r, c)$ (and $x_v, y_v$ are not yet set) then
        pass message $(x_r, y_r, r, c)$ right hand rule
        set $x_v \leftarrow x_r + r\sin(2\pi c_v/c)$
        set $y_v \leftarrow y_r - r\cos(2\pi c_v/c)$
repeat the last step at most $2n$ rounds
```

Starting from degree-one-nodes Algorithm 1 detects all tree children in G_{GG} and assigns values z_v that induce a prefix ordering of each tree. This allows easy addressing by Routing Rule 1. Note that a precomputation of an upper bound of the diameter of G_{GG} can replace the n-rounds repetitions, where n is the number of nodes in G. Let the next algorithm steps work only on non-tree nodes and tree roots, i.e. assume there exist no tree children. Algorithm 1 is mainly used to decrease the number of virtual edges inserted in the next three steps.

Algorithm 2 has to be extended as follows if the outer face may contain cut nodes. If a message $(\widehat{x}_v, \widehat{y}_v, c)$ initiated from node v with coordinates $\widehat{x}_v, \widehat{y}_v$ reaches a node w more than once (which can only happen if w is a cut node), w does not set its c_w value, but inserts a *virtual edge* between its predecessor and its successor on the outer face and then passes the message $(\widehat{x}_v, \widehat{y}_v, c)$ right hand rule (neglecting virtual edges). Note that the inserted virtual edges may temporarily destroy the planarity of the embedding, but they do not affect planarity of the barycentric layout after completion.

Algorithm 3: BR step 3 (establish 2-connectedness).

```
repeat
    foreach $v \in V$ do
        if $v$ is not a perimeter node then
            apply iteration (1)
        if $v$ both had neighbors $u \in U$ with $d(v,u) < \varepsilon$ and neighbors $w \in W$ with $d(v,w) \ge \varepsilon$ for more than $\kappa_1$ rounds then
            insert *virtual edge* between succeeding neighbors $u \in U, w \in W$ if this edge not already exists
until stop criterion is fulfilled
```

Algorithm 4: BR step 3 (establish 3-connectedness).

```
repeat
    foreach $v \in V$ do
        if $v$ is not a perimeter node then
            apply iteration (1)
        if $v$ had neighbors $u \in U, |U| \ge 2$, all within a sector $S$ of angle $\vartheta$, for more than $\kappa_2$ rounds then
            insert *virtual edge* between $u \in U$ and $w^*$, the first node that starting from a neighbor $w \notin U$ succeeding (preceeding) $u$ right hand rule (left hand rule), is outside the sector $S$ rotated by $\pi$ if such a node $w^*$ exists
until stop criterion is fulfilled
```

Parameters ε, κ_1 and ϑ, κ_2 for the next step, together with an appropriate stop criterion have to be adapted depending on the specific scenario. Choosing ε too small and the stop criterion too weak may result in a non-detection of a cut node. Choosing ε rather too great (we used 10^{-2} in our tests) is the better choice, since unnecessarily inserted edges do not affect the remaining algorithm. When choosing a weak stop criterion it is helpful to extend the following Routing Rules 1 to 4 such that degree-2-nodes simply pass the message to their *other* neighbor. Algorithm 3 terminates (i.e. it stops inserting virtual edges) because of the

Euler Formula for planar graphs. In fact, when choosing ε sufficiently small, virtual edges are inserted unnecessarily (i.e. if G is already 2-connected) only if the edge is part of a component G_1 that is separated from G by a cut pair.

Lemma 1 *Consider the barycentric layout p_C of a connected planar graph G. G is 2-connected if Algorithm 3 does not insert a virtual edge.*

Lemma 2 *Consider the barycentric layout p_C of a 2-connected planar graph G. The layout p_C is planar if Algorithm 4 does not insert a virtual edge.*

Algorithm 4 terminates because each inserted virtual edge decreases the number of cut pairs (after termination only cut pairs $v, w \in C$ may remain). Thus, the barycentric layout finally becomes a planar embedding that allows no more insertion of virtual edges.

The insertion of virtual edges can cause a delay of position information during the communication of iteration (1), if the inserted edges do not exist in the original graph G, i.e. a node v that has a virtual neighbor w at a distance k only receives its position after k rounds instead of one round.

Lemma 3 *Iteration* (1) *also converges to p_C with delayed position information.*

3.2. BR Rule

We now present our routing rules. These assume that virtual coordinates, a barycentric layout p_C, have been precomputed. The following simple rule is sufficient for guaranteed message delivery and already delivers good results for graphs of certain densities (see Figure 1).

Routing Rule 1 *If $d(v,t) = 0$ pass message to w with maximum $z_w \leq z_t$.*
If $d(v,t) > 0$ and $z_v > 0$ pass message to the neighbor w with minimum z_w.
If $d(v,t) > 0$ and $z_v = 0$ pass message to w with minimum angle $\psi_t(v,w) \geq 0$.

3.3. GBR Rule

For dense graphs BR suffers from its restriction to the Gabriel Graph. GBR prevents this drawback. Whenever possible GBR advances in greedy mode to t using the original graph G instead of G_{GG}. This leads to an algorithm with overall good performance on the whole density spectrum, outperforming routing algorithms on geographic coordinates (see Figure 1) .

Routing Rule 2 *Pass message to a G-neighbor w with minimum distance $d(w,t) < d(v,t)$ if such a neighbor exists. Otherwise, pass the message according to Rule 1 until reaching a node w with $d(w,t) < d(v,t)$ and then again apply Rule 2.*

3.4. AGBR Rules

AGBR needs an additional step 0 during the precomputation phase for virtual coordinates to prove Theorem 5 that is analogous to a theorem concerning the message path lengths of GOAFR+ [5]. Hence, AGBR fixes the unbounded message paths of GBR in the worst-case (Theorem 4), while still being good in practice. In fact, in our tests (Figure 1) we used AGBR instead of GBR, but we chose parameter ρ sufficiently great, such that AGBR always runs in GBR mode. Step 0 computes a *backbone graph G_{BG}* (see, e.g. [9]), i.e. a subgraph of the original unit disk graph G forming a dominating set of G. The remaining steps apply to the backbone graph and routing between the backbone graph G_{BG} and G is straight-forward.

Routing Rule 3 *Pass message according to Rule 2 if it will thereby be passed to a node w with $\widehat{d}(w,t) \leq \rho\widehat{d}(s,t)$. Otherwise, walk right hand rule along the outer face of the graph G_ρ formed by all nodes u with $\widehat{d}(u,t) \leq \rho\widehat{d}(s,t)$ until returning to v. If t was not surrounded by this walk set $\rho \leftarrow 2\rho$ and again apply Rule 3. Otherwise, set angle $\varphi^* \leftarrow \varphi_t(v)$ and $v^* \leftarrow v$ and apply Rule 4.*

Routing Rule 4 *Pass message according to Rule 1 if it will thereby be passed to a node w with $\widehat{d}(w,t) \leq \rho\widehat{d}(s,t)$. Otherwise, walk right hand rule around the outer face of G_ρ until a node w with $\varphi_t(w) \geq \varphi^*, w \neq v^*$ is reached. Set $\varphi^* \leftarrow \varphi_t(w), v^* \leftarrow w$ and apply Rule 4.*

3.5. AGBFR Rule

AGBFR uses the same rules as AGBR with some modifications to Rules 1 and 4. The good performance of AGBFR is only due to improved greedy success rates, i.e. the ratio of delivered messages only using greedy mode (see Figure 2).

Routing Rule 1a *If $d(v,t) = 0$ pass message to w with maximum $z_w \leq z_t$.*
If $d(v,t) > 0$ and $z_v > 0$ pass message to the neighbor w with minimum z_w.
If $d(v,t) > 0$ and $z_v = 0$ walk right hand rule (directed to $\overrightarrow{vt}$) along the face at v until returning to v. For the edge e with minimum $d(e,t) < d(v,t)$ continue walking right hand rule (directed to $\overrightarrow{p_e t}$ for any point $p_e \in E$) along the face at e.

Routing Rule 4a *Apply Rule 2 restricted to G_ρ.*

4. Theoretical Results

In this section we consider provable properties of the routing methods just introduced. Experimental results follow in the Appendix. Routing Rule 1 directly implies the following lemma for tree nodes.

Lemma 4 *Routing rule 1 guarantees message delivery if s and t are in the same tree. Otherwise the message is passed from s to its root node if s is a tree node. If t is a tree node the message will be passed from its root node to t.*

Theorem 2 *BR, GBR and AGBR guarantee message delivery.*

Theorem 3 *AGBFR guarantees message delivery after iteration (1) yields a planar embedding (respecting the cyclic order of neighbors from G_{GG}).*

Theorem 4 *There exists a series of unit disk graphs with a designated target node t, such that the expected quotient of the length of the BR-path (and GBR-path) between a random source node s and t divided by the length of a shortest s-t-path is unbounded.*

Theorem 5 *The length of an AGBR-path is in $\mathcal{O}(\ell^2)$, where ℓ is the length of a shortest s-t-path.*

The following and previous theorem together deliver worst-case optimality of AGBR.

Theorem 6 *There exists a series of unit disk graphs with a designated target node t, such that the expected length of an AGBR-path between a random source node s and t is in $\Omega(\ell^2)$, where ℓ is the length of a shortest s-t-path.*

5. Conclusion

We presented four routing methods for networks, where geographic positions of the nodes are available. All methods make use of virtual coordinates obtained during a one-time-precomputation phase. Using a barycentric layout as virtual coordinates allows simple routing rules with guaranteed delivery, and very short routes in practice, outperforming algorithms working only on geometric coordinates. Attempts to reduce the consuming precomputation phase are made with AGBFR. Nevertheless, if guaranteed message delivery on short routes is obligatory, our methods clearly apply to static, long-time-living networks, where the precomputation is affordable and where short routes are of great importance, but memory for full routing tables is not available. Considering AGBFR as a heuristical routing method on the other hand might also make it a candidate for dynamic networks, where some few iterations (even only local) are used to update the barycentric layout, since in general small modifications in the network G_{GG} induce small modifications in its barycentric layout.

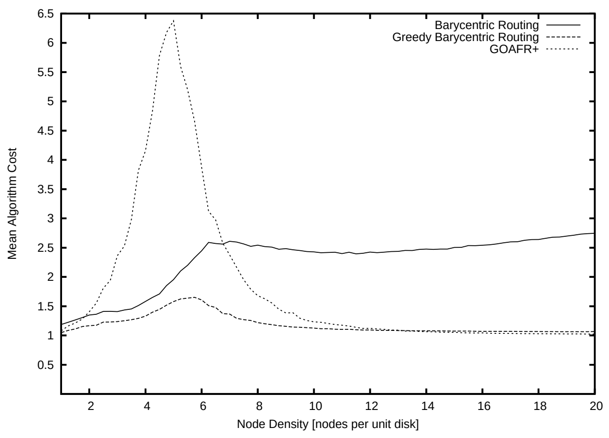

Figure 1. The mean algorithm costs of BR, GBR and GOAFR+ are compared on random unit disk graphs with 1000 nodes.

References

[1] D. Fleischer and C. Pich. Positioning and virtual coordinates. *In Wagner, D. and Wattenhofer, R., Eds.: Algorithms for Sensor and Ad Hoc Networks, Springer LNCS*, to appear.

[2] C. Godsil and G. Royle. *Algebraic Graph Theory*. Springer, 2001.

[3] W. Hackbusch. *Iterative Solution of Large Sparse Systems of Equations*. Springer, 1994.

[4] E. Kranakis, H. Singh, and J. Urrutia. Compass routing on geometric networks. *Proc. 11 th Canadian Conference on Computational Geometry*, 1999.

[5] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. Geometric ad-hoc routing: Of theory and practice. *Proc. Twenty-Second ACM Symposium on Principles of Distributed Computing*, 2003.

[6] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. *Proc. ACM MobiCom Conference*, 2003.

[7] H. Takagi and L. Kleinrock. Optimal transmission ranges for randomly distributed packet radio networks. *IEEE Transactions on Communication*, 32(3):246–257, 1984.

[8] W. Tutte. How to draw a graph. *Proc. London Math. Soc.*, 13(3):743–768, 1963.

[9] Y. Wang and X. Li. Geometric spanners for wireless ad hoc networks. *Proc. 22nd IEEE International Conf. Distributed Computing Systems*, 2002.

[10] A. Zollinger. Geographic routing. *In Wagner, D. and Wattenhofer, R., Eds.: Algorithms for Sensor and Ad Hoc Networks, Springer LNCS*, to appear.

A. Experimental Results

The *mean algorithm cost* is defined as the quotient of the length of the message path divided by the length of a shortest path. Figure 1 shows the results of tests on random unit

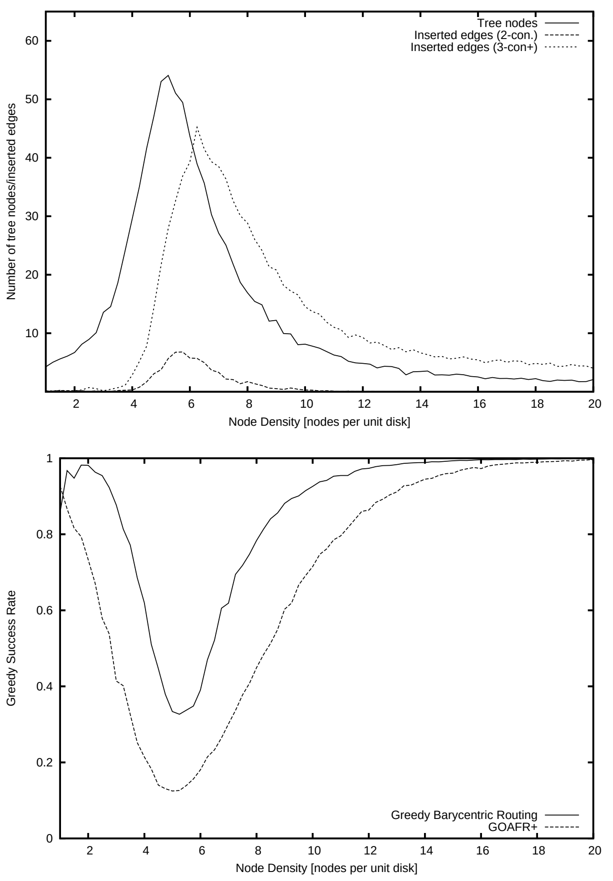

Figure 2. The number of tree nodes and the number of virtual edges that were inserted is displayed. Furthermore, success rates of greedy routing for GBR and GOAFR+.

disk graphs with 1000 nodes. For each node density from 1.0 to 20.0 in steps of 0.25 we created 100 networks, and routed messages between 1000 s-t-pairs. Figure 2 shows the improvement of the greedy success rate, i.e. the ratio of messages delivered only in greedy mode, due to convex faces in barycentric layout that make dead-locks unprobable. Furthermore, the numbers of tree nodes and inserted virtual edges are shown. Figure 3 finally shows mean algorithm costs of AGBFR with various maximum numbers of applications of iteration (1), restricted to the cases when delivery was successful. The delivery ratio of successful AGBFR-routes is also given in Figure 3.

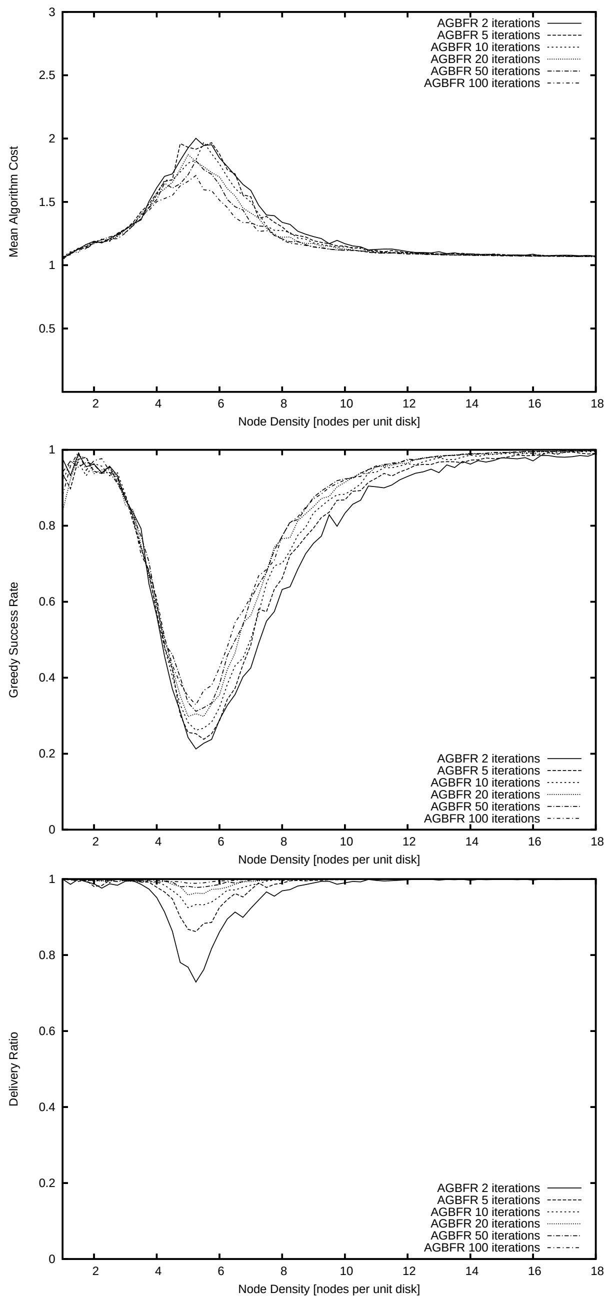

Figure 3. Mean algorithm costs and greedy success rates of AGBFR at maximum number of iterations 2,5,10,20,50,100 with successful delivery. The delivery ratio is displayed below.