Beacon-based algorithms for geometric routing

DOI:
10.1007/978-3-642-40104-6_14

Document status and date:
Published: 01/01/2013

Document Version:
Accepted manuscript including changes made at the peer-review stage

Beacon-Based Algorithms for Geometric Routing [⋆]

Michael Biro[⋆⋆], Justin Iwerks[⋆⋆], Irina Kostitsyna[⋆⋆⋆], and Joseph S. B. Mitchell[⋆⋆]

Stony Brook University

Abstract. We consider beacons, an analog of geographical greedy routing, motivated by sensor network applications. A beacon b is a point object that can be activated to create a 'magnetic pull' towards itself everywhere in a polygonal domain P. We explore the properties of beacons and their effect on points in polygons, as well as demonstrate polynomial-time algorithms to compute a variety of structures defined by the action of beacons on P. We establish a polynomial-time algorithm for routing from a point s to a point t using a discrete set of candidate beacons, as well as a 2-approximation and a PTAS for routing between beacons placed without restriction in P.

1 Introduction

We consider a model of *beacon-based routing* that generalizes geographical greedy routing in sensor networks. In geographical routing [1, 2], each node is given a Euclidean coordinate and a message is transmitted to the neighbor whose Euclidean distance to the destination is a minimum. When the distribution of sensors is very dense, i.e. close to infinity, the route a message takes under geographical routing will follow a straight line towards the destination, or, when the message hits the network boundary, may follow a boundary edge to greedily minimize the distance to the destination. This is precisely the model of beacon-based routing in this paper (see also [3–6]), where the destination is a beacon. In this context, we demonstrate algorithms to compute all nodes that can transmit a message to a given beacon, to compute all nodes that a given node can transmit to, as well as algorithms to compute an optimal, or nearly-optimal, sequence of beacon locations to transmit messages between two given nodes in the network.

Other routing schemes in sensor networks that are related to beacons include a family of routing methods that use *landmarks*, as in Fang et al., Fonseca et al., and Nguyen et al., [7–9]. In this type of routing, a collection of nodes, called landmarks, first transmit throughout the entire network so that each node

[⋆] This research was supported by the National Science Foundation (CCF-1018388).

[⋆⋆] Department of Applied Mathematics and Statistics, Stony Brook University, {`mbiro, jiwerks, jsbm`}`@ams.stonybrook.edu`

[⋆⋆⋆] Department of Computer Science, Stony Brook University, `ikost@cs.stonybrook.edu` can record its distance to each landmark. Then, in order to route towards a destination, a function based on the distance vector to the landmarks is used for selecting the neighbor to which to transmit the message next. The paper containing a model most similar to ours is the one adopted in Nguyen et al., [9]. In their paper, the message is routed directly towards a single landmark until the current node is at an equal distance away from the landmark as the destination. At that point another landmark is selected. The paper shows that by choosing the landmarks carefully, the message path's length is within a constant factor of the shortest path.

In our model, a beacon can occupy a point location on the interior or the boundary of P, ∂P. When a beacon is *activated*, we imagine that an object starting at a point $p \in P$ moves along a straight line toward b until it either reaches b or makes contact with ∂P. If contact is made with ∂P, the object will follow along ∂P as long as its straight-line distance to b decreases monotonically. Following the path determined by the beacon, the object may alternate between moving in a straight-line path toward b on the interior of P and following along ∂P. If there is no infinitesimal movement that an object at p can make so that its distance to b (strictly) decreases, we say that the object is 'stuck' and has reached a local minimum or *dead point* on ∂P (see Figure 3). If an object starting at p eventually reaches b we say that b *attracts* p. Two points are *routed* if there is a sequence of beacons that can be activated and then deactivated, one at a time and in order, so that an object beginning at a starting point s would visit each beacon in the sequence after it is activated and terminate at a destination point t, which we will always require to be a beacon itself.

2 Properties of beacons

We examine the effect of beacons among obstacles in the plane. Our terminology describes the attracted components to be moving objects, however, autonomous robots, message delivery, or geographic routing interpretations are equally valid. We begin with some definitions describing the structures and behavioral properties of beacons in polygons.

We define a *beacon* b as an transmitter-like object that is placed at a point in a polygon P and can be *activated* to effect a pull on objects in P. When b is activated, objects in P move to greedily minimize their Euclidean distance to b, while being constrained to remain interior to P. A beacon b *attracts* a point p if, under the action of b, an object starting at p moves so that its Euclidean distance to b eventually decreases to 0. In this case, we also say that p is attracted to b.

Using these definitions, we may want to determine the set of points that are attracted to a beacon b, called the *attraction region of* b, $A(b)$ or determine the set of beacons that attract a point p, called the *inverse attraction region of* p, $IA(p)$ (See Figure 1). If a beacon b is activated, objects at the points that are attracted to b will reach b, but objects at the points not attracted to b will reach a local minimum with respect to distance to b in P and remain there under the influence of b. We are interested in determining the classification of the points of

P based on the final position of the points under the action of b. These questions motivate the following definitions.

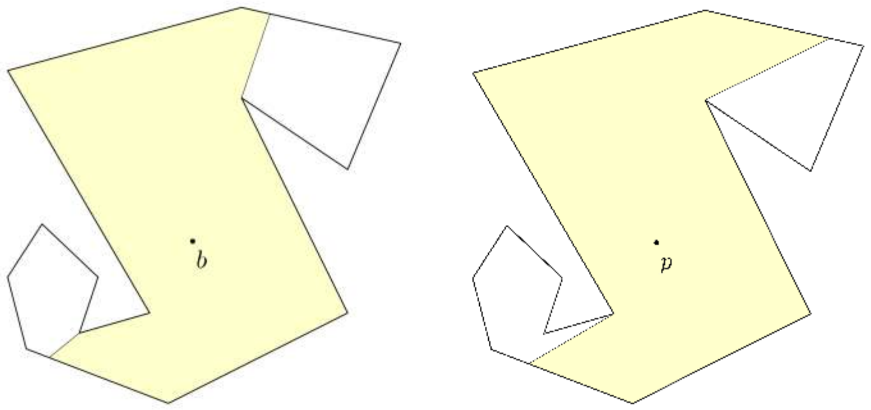

Fig. 1. (Left) The attraction region of a beacon b. (Right) The inverse attraction region of a point p.

We say a point d in P is a *dead point* with respect to a beacon b if d is not b, and an object at d remains stationary under the influence of the beacon b. That is, d is a point such that the Euclidean distance from b to d is a non-zero local minimum inside of P. For a given beacon b in a polygon P, let $D(b)$ be the set of dead points with respect to b in P.

Then, for each dead point of a beacon b in a polygon P, $d \in D(b)$, define the *dead region* of d with respect to b, $DR_b(d)$, to be the set of points of P that reach d if the beacon at b is activated. Since d is at a local minimum with respect to Euclidean distance to b, if a point reaches d it can never leave d under the action of b.

We can bound the number of dead points a given beacon may have in a polygon P.

Theorem 1. *Let P be a simple polygon on n vertices. If $D(b)$ is the set of dead points with respect to b, then $0 \leq |D(b)| \leq n-3$. Similarly, let P be a polygon with n vertices and h holes. If $D(b)$ is the set of dead points with respect to b, then $0 \leq |D(b)| \leq n-h-3$. Furthermore, these bounds are tight.*

Proof. For simple P: If P is convex, then there are no dead points, satisfying the lower bound. Since $b \in P$, at least 3 edges of P must have a point visible to b, which implies that these three edges cannot have any dead points. Since no edge can have more than 1 dead point, this implies that the upper bound of $n-3$ cannot be exceeded. An example achieving the upper bound is shown in Figure 2.

For arbitrary P: If P is convex and the holes are triangles oriented so they lack dead points, then there are no dead points, satisfying the lower bound. Let H be the set of holes, and let n_i be the number of vertices of the ith hole. First, examine the polygon, $P - H$, with all the holes removed. Polygon $P - H$ has at most $n - \sum_i n_i$ vertices, and since $b \in P$, $P - H$ can have at most $n - \sum_i n_i - 3$ dead points. Now examine a hole $i \in H$, with n_i vertices. Take a planar arrangement consisting of only b and i and note that at least one edge of i must be visible to b, so it does not contribute a dead point. Therefore, i can contribute at most $n_i - 1$ dead points, and all holes together contribute at most $\sum_i (n_i - 1) = (\sum_i n_i) - h$. Adding the two contributions yields $n - \sum_i n_i - 3 + \sum_i n_i - h = n - h - 3$ dead points. An example achieving the upper bound is shown in Figure 2. □

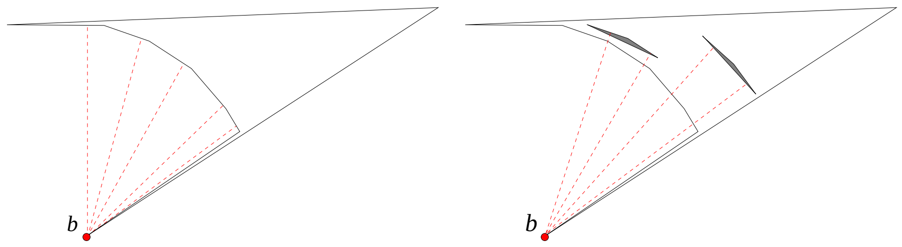

Fig. 2. (Left) A simple polygon with $n = 8$ vertices and $n - 3 = 5$ dead points. (Right) A polygon with $n = 14$ vertices, $h = 2$ holes and $n - h - 3 = 14 - 2 - 3 = 9$ dead points. The four additional dead points are shown.

Lemma 1. *The set of dead regions, $D(b)$, along with the attraction region of b, $A(b)$, forms a partition of the polygon P.*

Proof. We see that every point must eventually either reach b or be forced to stop at a dead point, so these sets cover P. We remove any ambiguity about the movement of a point on a *reflex* vertex (having internal angle greater than π) by assuming it always falls to the left of $\overrightarrow{bp}$. Then, every point follows a unique path induced by the beacon b, as the rules for all possible positions are fixed. Therefore a point cannot end up at two different dead points d_1 and d_2, and so the dead regions and attraction region subdivide the polygon into disjoint regions. Since each point is in a region, this is a partition of the polygon P. □

Using local criteria (see [6] for details), we can determine special *cut vertices*, *split vertices*, and *split edges* that are vital in determining the boundary edges of the partition of a polygon under the action of a beacon. Cut vertices are reflex vertices of the polygon such that the ray emanating from the vertex oriented away from b lies interior to the polygon. There are three classes of cut vertices, depicted in Figure 4, corresponding to the different ways the ray may lie in the polygon. Furthermore, we examine the different cut vertices for situations where

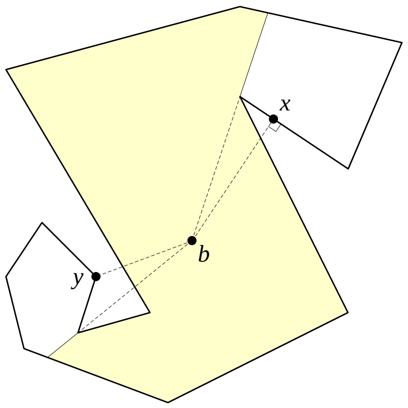

Fig. 3. The partition of P with respect to b. Highlighted is the attraction region $A(b)$ of beacon b; x and y are dead points with respect to b.

the vertex acts as a separator, i.e., where the edges are angled so that points on the left of the ray from b slide away from points on the right of the ray and vice versa. These special vertices are the split vertices, and the line segment from them to the first intersection of the ray with the polygon is called the split edge of that split vertex. The far vertex of a split edge is called the *ray vertex* of that split edge (split vertex, respectively). Using these special classes of vertices, we may classify the boundary of the partition of a simple polygon P into an attraction region and dead regions.

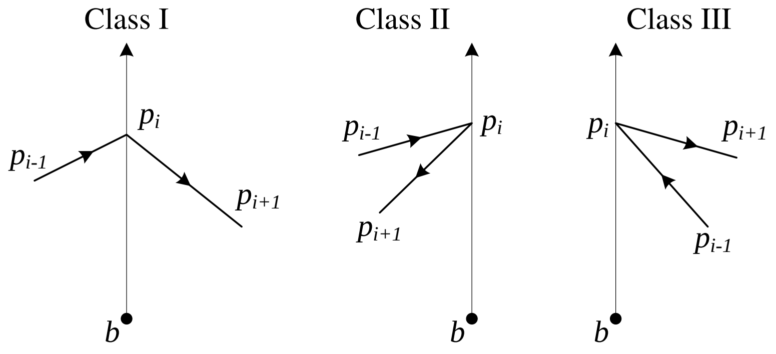

Fig. 4. Three classes of cut vertices

Theorem 2. *If P is a simple polygon, and $e = \overline{p_i q_i}$ is a split edge of b, then e is the boundary between two regions of the partition of P with respect to b. If P*

has holes, then e may lie entirely interior to a region of the partition, but cannot intersect more than one region of the partition due to b.

Proof. In the case where P is simple, e is a diagonal of the polygon $P + q_i$ and therefore splits it into two pieces, P_L and P_R. The convention mentioned in Lemma 1 means that the points on e move to the left and so are part of P_L. Since e is parallel to $\overline{bp_i}$, the direct unconstrained action of b can never pull a point from P_L to P_R or vice versa. Therefore, the only possible way for a point to move from one side of e to the other is to move unconstrained until reaching ∂P and then slide along an edge. In order to slide along an edge across e, it must pass p_i, and therefore must slide along p_ip_{i+1}, or $p_{i-1}p_i$, depending on whether it started in P_L or P_R. Since p_i is a split vertex, then regardless of which class cut vertex it is, due to the angles defined for a split vertex, a point on edge p_ip_{i+1} or $p_{i-1}p_i$ is pulled away from p_i, and can never reach it. Therefore, since the polygon is simple, the points on one side of e cannot end in the same location as the points on the right side and so are in different regions. Furthermore, e is the boundary of at most two regions as the unconstrained points all travel parallel to their ray from b and end at the same point, meaning that a given ray lies in the same region. Therefore, they are in different regions and e is their boundary.

If P has holes, then the same argument holds, except for the fact that some split edges may have points arbitrarily close on either side that end up in the same location due to points sliding around holes. This corresponds to the split edge lying entirely inside a given region. □

Conversely, we can classify the boundary edges of the partition of P with respect to b.

Theorem 3. *If a curve is a boundary edge of a region in the partition of a polygon P defined by a beacon b then it is either a part of the boundary of P or a split edge of b.*

Proof. Take a boundary component c of a region that is not part of an edge of P. If some length of c is not parallel to the ray from b, then the unconstrained attraction from b will pull points across c, implying that the two sides share a dead region. Therefore, c must be a straight segment parallel to the ray from b. Now, c must intersect the polygon at two locations, say s_1 and s_2, with s_1 closer to b. All points on c slide down c to s_1 under the influence of b. If s_1 is on the interior of an edge, there are two cases. If s_1 is a dead point, then points on both sides of the edge of s_1 slide to s_1, implying that c is in the interior of the dead region of s_1, contradiction. If s_1 is not a dead point, then it will slide along the edge, either left or right. In both cases, points from both sides of c end at the same dead point, so c is not on a boundary. Therefore, s_1 is a vertex. We see that the conditions that force all points from one side of c to a different region than all points on the other side of c are exactly the conditions that make c a split vertex.

Therefore the boundary edges of regions in the attraction arrangement are exactly the edges of P, dead edges, or edges of the form (p_k, q_i) or (q_i, q_j) for some pair of adjacent ray vertices q_i, q_j. □

These theorems form the idea for the attraction-region algorithms for points given in the next section. We first find the split vertices of the polygon with respect to the beacon b, then propagate the split edges to find the ray vertices. In simple polygons, this immediately gives the attraction partition of the polygon P with respect to a point beacon b. For polygons with holes, some of the split edges may not be relevant, so we then walk along the boundary of each region, deleting edges seen twice, as they are interior edges. Then, the attraction arrangement is exactly a decomposition of P into polygons each of which either contains a single dead point or, in the case of the attraction region, b. The arrangement therefore consists of the dead regions and the attraction region.

In the following, we give some additional global properties of the attraction region of a point in a polygon P. Specifically, properties of connectedness, convexity, simplicity, and complexity are given for both simple polygons and polygons with holes.

Proposition 1. *Given a beacon b, $b \in A(b)$. Furthermore, the visibility polygon of b, $V(b)$, is a subset of $A(b)$.*

Proof. Each point in the visibility polygon moves to greedily minimize its Euclidean distance to b. Since the straight line path connecting them to b lies in the polygon, that is the path they take, and they are attracted to b. There are also examples where equality is achieved. □

Proposition 2. *The attraction region of a beacon b in a polygon P is connected.*

Proof. Take two arbitrary points in $A(b)$, say p_1 and p_2. Then, b attracts both p_1 and p_2, as well as the entirety of the paths each take under the action of b. The concatenation of these two paths gives a path from p_1 to b to p_2, showing that $A(b)$ is connected. □

Theorem 4. *The attraction region of a beacon b in a simple polygon P is convex with respect to P. This is not necessarily the case if P has holes.*

Proof. A subset R of a polygon P is convex with respect to P if, for any two points p_1 and p_2 in R, either p_1 is not visible to p_2, or the line segment $\overline{p_1 p_2}$ lies entirely in R. See Figure 5. Take a line segment that does not intersect ∂P with endpoints p_1 and p_2 that are both attracted to b. Suppose there exists a point on the segment, p_3, that is not attracted to b. Since p_3 is not attracted to b, p_3 lies outside $A(b)$ and so there exists a split edge that separates p_3 from b. Since the line segment is convex, that split edge must also separate one of p_1 or p_2 from b, and that is a contradiction, as both p_1 and p_2 are attracted to b. In polygons with holes, the line segment may be intersected by many split edges, instead of at most 2 as in simple polygons. Therefore, there can be points on the line segment that are not attracted to b. □

Corollary 1. *The attraction region of a beacon b in a simple polygon P is simple, i.e. it has no holes.*

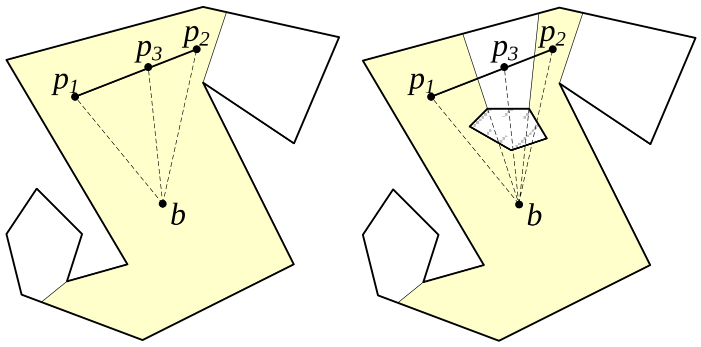

Fig. 5. A beacon's attraction region is convex with respect to a simple polygon and is not necessarily convex with respect to a polygon with holes.

Theorem 5. *The partition of a polygon P with respect to a beacon b has boundary complexity $O(n)$. Specifically, $A(b)$ has complexity $O(n)$. Furthermore, there are cases with $\Omega(n)$ complexity. See Figure 2.*

Proof. P has n vertices and there are at most $n-3$ additional ray vertices added to make P'. Split edges cannot cross, and so there are at most $2n-3$ vertices in the partition of P. Therefore, the partition has complexity $O(n)$. Since $A(b)$ is a subset of the partition, it must also have at most $O(n)$ complexity. □

3 Algorithms for Computing Attraction Regions

Recall the definition of the attraction region of a point: The *attraction region*, $A(b)$, of a beacon b in a polygon P is defined as the set of points p in P that are attracted by b.

We can compute the attraction region, $A(b)$, by various methods; details can be found in [6]. One method is based on a simple rotational sweep, taking time $O(n \log n)$, and space $O(n)$, in simple polygons and polygons with holes. Another method is based on preprocessing P for ray-shooting queries, resulting in time $O(n \log n)$ in a simple polygon and $O(\sqrt{h}n \log n)$ in a polygon with h holes. Alternatively, the ray shooting approach can be done in time $O(n)$ in a simple polygon and $O(nh)$ in a polygon with h holes by using a triangulation to propagate only the split edges relevant to the attraction region, exploiting the connectedness and convexity properties of the attraction region.

Our overall most efficient method of computing $A(b)$ yields a running-time bound of $O(T(n) + n)$, where $T(n)$ is the time to triangulate the given polygon ($O(n)$ in simple polygons, and $O(n+h \log^{1+\epsilon} h)$, respectively for simple polygons and polygons with holes [10,11]). This is (nearly) optimal time, yet the difficulty of implementation may make the prior algorithms more suitable for practical use.

This algorithm computes the necessary split edges by finding a *radial trapezoidization* of the polygon P emanating from the beacon b. This trapezoidization may be computed from a triangulation of P in linear time, by a modification of the result on parallel trapezoidization by Fournier and Montuna [12]. See also the thesis of Mouawad [13] for a description of the modification from parallel to radial trapezoidization. Once the trapezoidization is found, the split edges with respect to b may be found quickly, and after a linear amount of additional work as described above, we compute the full partition of P with respect to b in $O(T(n) + n)$ time and $O(n)$ space.

Theorem 6. *The partition of a polygon P with respect to a beacon b can be computed in $O(n)$ time and space if P is simple, and $O(n + h \log^{1+\epsilon} h)$ time, $O(n)$ space if P has holes.*

4 Algorithms for Computing Inverse Attraction Regions

Recall the definition of the inverse attraction region of a point or region: The *inverse attraction region*, $IA(p)$, of a point p in a polygon P is defined as the set of beacon locations b in P that attract p. Similarly, the *inverse attraction region* of a subset R, $IA(R)$, in a polygon P is defined as the set of points p in P such that a beacon b at p attracts at least one point of R.

In this section we discuss the computation of the inverse attraction region of a point p in a polygon P, as well as the inverse attraction region of a subset of P. Note that inverse attraction regions, unlike attraction regions, may have $\Omega(n)$ connected components ($\Omega(n^2)$ components in polygons with holes) and the components may be free-floating in the interior of the polygon, with boundary edges defined by non-local conditions. This makes their computation more difficult, and we resort to a decomposition approach that determines an arrangement that contains the inverse attraction, then test each face of the arrangement for attraction, using the algorithms in the preceding section. (see [6] for details).

4.1 Algorithm for the inverse attraction region of a point

The algorithm begins by constructing an arrangement $\mathcal{A}_p$ made from taking the arrangement of lines defined by each edge of the polygon and the lines through each reflex vertex that are perpendicular to the edges incident on the reflex vertex. Then, using the properties of split vertices, we can prove the following results.

Lemma 2. *If b_1 and b_2 are two points in a face F of the arrangement $\mathcal{A}_p$ and p_i is a split vertex relative to b_1, then p_i is a split vertex relative to b_2.*

This allows us to show that the faces of the arrangement are constant with respect to attracting p.

Theorem 7. *If b_1 and b_2 are two points in a face F of the arrangement $\mathcal{A}_p$ and $p \in A(b_1)$, then $p \in A(b_2)$.*

Therefore, we can test a candidate point from each face of the constructed arrangement, using the algorithms discussed in the previous section, and determine which faces make up the inverse attraction region. Walking through the arrangement allows us to update the attraction regions quickly, so this algorithm runs in $O(n^2)$ time.

Theorem 8. *The inverse attraction region of a point p in a polygon P can be computed in $O(n^2)$ time.*

4.2 Algorithm for the inverse attraction region of a region R

The following algorithm is a modification of the preceding algorithm, and works to compute the inverse attraction region of a polygonal region R with $|R| = m$. It uses the same decomposition idea, but with a slightly more refined arrangement, $\mathcal{A}_R$, made of the lines defined by each edge of the polygon, the lines through each reflex vertex perpendicular to the edges incident to the reflex vertex, and the lines from each vertex of R through each reflex vertex of P.

This allows us to show that the faces of the arrangement $\mathcal{A}_R$ are constant with respect to attracting a point from R.

Theorem 9. *If b_1 and b_2 are two points in a face F of the arrangement $\mathcal{A}_R$ and $R \cap A(b_1) \neq \emptyset$, then $R \cap A(b_2) \neq \emptyset$.*

Therefore, we can test a candidate point from each face of the constructed arrangement, using the attraction region algorithms from the previous section, and determine which faces make up the attraction region of R. Walking through the arrangement allows us to update the attraction regions quickly, so this algorithm runs in $O(m^2n^2)$ time.

Theorem 10. *The inverse attraction region of a region R with $|R| = m$, in a polygon P, can be computed in $O(m^2n^2)$ time.*

Later, we will use this algorithm for computing the inverse attraction region of a triangle, which takes $O(n^2)$ time.

Corollary 2. *The inverse attraction region of a triangle in a polygon P can be computed in $O(n^2)$ time.*

5 Beacon Routing

We are interested in finding a *minimum beacon path* between two points s, t, in a polygon P. A minimum beacon path from s to t is the smallest possible collection of points $b_1, b_2, \ldots, b_k$ in P with the property that b_1 attracts s, b_{i+1} attracts b_i for $i = 1, \ldots, k-1$, and t attracts b_k. Specifically, in this section, we solve the minimum beacon path problem for a special case where we are given a set of m candidate beacon locations, and we also approximate the solution in the general case.

5.1 Algorithm for minimum beacon routing with candidate beacons

If we are given a collection of m candidate beacon locations, the minimum beacon path algorithm constructs a digraph G whose vertices are the candidate locations and which has the edge $\overrightarrow{(u,v)}$ if $u \in A(v)$. The minimum beacon path is then given by the shortest $s - t$ path in G.

Theorem 11. *A minimum beacon path from s to t, chosen from a set of m candidate locations in a polygon P, can be found in time $O(mn + m^2)$ for simple polygons, and $O(m(n + h \log^{1+\epsilon} h + m \log h))$ for polygons with holes.*

Proof. Correctness follows from the one-to-one correspondence between $s - t$ beacon paths and $s - t$ paths in the graph G.

For simple polygons, the main contributor to the running-time is constructing G, where for each of the $m + 2$ point in C, we spend $O(n)$ computing the attraction region, and spend $O(n + m)$ determining which edges to include in G. This yields a total running-time of $O(m(n+m)) = O(nm + m^2)$. Computing the triangulation, the point location, and the shortest-path algorithm are all dominated by this running-time, so the total running-time is $O(nm + m^2)$.

For polygons with holes, the running time is increased, as for each of the $m+2$ points in C, we spend $O(n + h \log^{1+\epsilon} h)$ computing the attraction regions, and then spend $O(n + m \log h)$ to locate the candidates in the triangles. Again, the triangulation, point-location, and shortest-path algorithms are dominated by this running-time, so the total is $O(m(n + h \log^{1+\epsilon} h + m \log h))$ □

5.2 Approximation algorithm for minimum beacon routing

We also approximate the minimum beacon path by finding the inverse attraction regions of the set of triangles in a triangulation, then building a digraph G whose vertices correspond to triangles (along with s and t) and which has edge $\overrightarrow{(u,v)}$ if $u \cap IA(v) \neq \emptyset$. Then, a minimum s, t path in G corresponds to a sequence of triangles where a point in each triangle is attracted by a point in its successor triangle in the path. These points may not be the same, but a single additional beacon per triangle allows us to link the sequence of points together to yield a 2-approximation to the minimum beacon path.

Theorem 12. *A 2-approximation for the minimum beacons path from s to t can be found in time $O(n^3)$. This procedure can be iterated to achieve a polynomial-time approximation scheme for minimum beacon paths.*

Proof. Since the minimum beacon $s - t$ path is at least as long as the minimum path in G, and we use at most two beacons in our beacon path for each beacon in the minimum path in G, we have at most twice as many beacons as necessary. The running time is dominated by the computing of the inverse attraction region of triangles. By Corollary 2, this takes $O(n^2)$ per triangle, so $O(n^3)$ total. We can then find the attracted/attracting points by walking through the path starting from t, computing attraction regions for the points already determined. The minimum beacon path has length at most $O(n)$, [3], and so we spend at most $O(n^2)$, or $O(n^2 + nh \log^{1+\epsilon} h)$, to do so.

By increasing the number of iterations, (i.e $IA(IA(\ldots(IA(a)))) = IA^k(a)$) we modify the above algorithm to find a beacon path that has at most $k+1$ beacons for every k beacons in the minimum beacon path, yielding a PTAS. $\square$

References

1. P. Bose, P. Morin, I. Stojmenović, and J. Urrutia, "Routing with guaranteed delivery in ad hoc wireless networks," *Wireless Networks*, vol. 7, no. 6, pp. 609–616, 2001.
2. B. Karp and H. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *Proc. 6th International Conference on Mobile Computing and Networking.* ACM, 2000, pp. 243–254.
3. M. Biro, J. Gao, J. Iwerks, I. Kostitsyna, and J. Mitchell, "Beacon-based routing and coverage," in *21st Fall Workshop on Computational Geometry*, 2011.
4. ——, "Beacon-based structures in polygonal domains," in *CG:YRF 2012, Abstracts of the 1st Computational Geometry: Young Researchers Forum*, 2012.
5. J. Iwerks, "Combinatorics and complexity in geometric visibility problems," Dissertation, Stony Brook University, 2012.
6. M. Biro, "Beacon-based routing and guarding," Dissertation, Stony Brook University, 2013.
7. Q. Fang, J. Gao, L. Guibas, V. de Silva, and L. Zhang, "Glider: Gradient landmark-based distributed routing for sensor networks," in *INFOCOM 2005. 24th IEEE International Conference on Computer Communications*, vol. 1. IEEE, 2005, pp. 339–350.
8. 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 sensornets," in *NSDI 2005. Proc. 2nd Symposium on Networked Systems Design & Implementation-Volume 2.* USENIX Association, 2005, pp. 329–342.
9. A. Nguyen, N. Milosavljevic, Q. Fang, J. Gao, and L. Guibas, "Landmark selection and greedy landmark-descent routing for sensor networks," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications.* IEEE, 2007, pp. 661–669.
10. B. Chazelle, "Triangulating a simple polygon in linear time," *Discrete & Computational Geometry*, vol. 6, no. 1, pp. 485–524, 1991.
11. R. Bar-Yehuda and B. Chazelle, "Triangulating disjoint jordan chains," *International Journal of Computational Geometry and Applications*, vol. 4, no. 4, pp. 475–481, 1994.
12. A. Fournier and D. Y. Montuno, "Triangulating simple polygons and equivalent problems," *ACM Transactions on Graphics (TOG)*, vol. 3, no. 2, pp. 153–174, 1984.
13. N. Mouawad, "Minimal obscuring sets," Master's thesis, McGill University, 1990.