Energy Optimal Routing in Radio Networks Using Geometric Data Structures *

René Beier, Edgar Ramos, Peter Sanders, and Naveen Sivadasan
Max-Planck-Institut für Informatik,
Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany.
E-mail: [rbeier,ramos,sanders,ns]@mpi-sb.mpg.de
WWW: http://www.mpi-sb.mpg.de/~sanders

January 21, 2002

Abstract

Given the current position of n sites in a radio network, we discuss the problem of finding routes between pairs of sites such that the energy consumption for this communication is minimized. Though this can be done using Dijkstra's algorithm on the complete graph $G = (V, V \times V)$ in time $\mathcal{O}(n^2)$, it is less clear how to do it in sub-quadratic or even near linear time. We present such algorithms for the important case where the transmission cost between two sites is the square of their Euclidean distance plus possibly a constant offset. We give an $\mathcal{O}(n \log n)$ time algorithm for the case where offsets can be neglected. This algorithms also works for super-quadratic cost functions and limited transmission ranges. We give an $\mathcal{O}(kn \log n)$ time algorithm that permits offsets and finds an optimal path with at most k hops, and an $\mathcal{O}(n^{1+\epsilon})$ time algorithm for the case of an unrestricted number of hops. The algorithms are based on geometric data structures ranging from simple 2-dimensional Delaunay triangulations to more sophisticated proximity data structures that exploit the special structure of the problem.

1 Introduction

Networks between a number of sites can be set up without additional infrastructure using radio communication between the sites. Such an approach has

*This work was partially supported by DFG grant SA 933/1-1 and the Future and Emerging Technologies programme of the EU under contract number IST-1999-14186 (ALCOM-FT).

recently gained considerable interest. We discuss the fundamental problem of finding good routes between pairs of sites.

Since the sites have often only limited energy reserves from batteries or power from solar panels, a prime optimization criterion is the energy consumption. We model this as follows. When two sites, say u and v, communicate directly, the sender node u needs a transmission energy $C_u + f(|uv|^2)$. The *cost offset* C_u accounts for distance independent energy consumption like the energy consumption of the signal processing during sending and receiving. $|uv|^2$ denotes the square of the Euclidean distance of u and v. $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ is a function where $f(a+b) \geq f(a) + f(b)$. For example, this encompasses the frequently used model where $f(x) = x^{\rho/2}$ for some power $\rho \geq 2$ and also allows limited range. f can also be any convex function with $f(0) = 0$. For most results we assume the particularly important case $f =$id that describes energy requirements of free space communication.

In addition to energy consumption, another expensive resource is bandwidth[1] — transmitting a data packet using range r inhibits[2] communication in an area of size $\Theta(r^2)$. Hence, our quadratic cost measure can also be viewed as an estimate for the degree to which a route hinders other connections from using the same frequencies. A third cost measure is the reliability of a connection. The more nodes are involved, the higher is the probability of a failure. This cost measure is indirectly captured by the offset C_u.

We model the network as a complete geometric graph where the nodes of the graphs corresponds to the sites of the network and an edge from node u to v in the graph has weight $|uv|^2 + C_u$. In principle, an optimal path can be easily found in time $\mathcal{O}(n^2)$ using Dijkstra's shortest path algorithm in this graph. The subject of this paper is to exploit the geometric structure of the problem to find energy optimal routes in time closer to $\mathcal{O}(n)$ than $\mathcal{O}(n^2)$.

1.1 Our results

In this paper we consider different variations of the problem depending on the assumptions on the offset costs as well as on the restrictions on the number of hops. In section 2, we consider the case in which all offset costs are zero but the number of hops can be arbitrary. For this case, we observe that it suffices to find the shortest path in the Delaunay triangulation of the nodes of the graph. This immediately yields an algorithm with running time $\mathcal{O}(n \log n)$ and space $\mathcal{O}(n)$. Intuitively speaking, in the zero offset case, routing is easy because it is optimal to communicate only between near neighbors. This property completely breaks

[1] 120 MHz of UMTS frequency range in Germany cost about $50 \cdot 10^9$ Euro

[2] In channel based systems like GSM, inhibition basically means blocking the channel for other concurrent communication. In CDMA systems like UMTS the inhibition is more subtle but nevertheless present.

down if the offsets are present. Nevertheless, it turns out that the dual of the Delaunay triangulation — the Voronoi-diagram — is still useful if we want to find routes with very few hops. In Section 3 we give a very simple algorithm that finds optimal 2-hop paths for uniform offset costs in time $\mathcal{O}(\log n)$ after a preprocessing time of $\mathcal{O}(n \log n)$ and space $\mathcal{O}(n)$. Using the dynamic programming principle this can easily be extended to an $\mathcal{O}(n \log n)$ algorithm for optimal paths with at most four hops. We then give a more general algorithm for finding optimal routes with up to k hops and non-uniform offset costs with running time $\mathcal{O}(kn \log n)$ and $\mathcal{O}(n)$ space by reducing it into a special case of 3D nearest neighbor problem and solving it efficiently. All the above algorithms are quite practical. Finally section 4 gives a more theoretical solution for the general case of non-uniform costs and arbitrary number of hopes with running time and space $\mathcal{O}(n^{1+\epsilon})$ for any constant $\epsilon > 0$. The solution is based on 3D closest bichromatic pair queries. We give ideas for handling further modeling issues including obstacles, limited range of nodes, dynamic network changes, sub-linear time approximation etc. in Sections 5 and 6.

1.2 Related Work

Energy efficient communication in radio network is a widely studied problem. Besides simple constant cost models [5] the most common simple model is our model with $f(|uv|^2) = |uv|^\rho$ for some constant $\rho \geq 2$ [8, 6].[3] Rabaey et al. [4] mention sensor and monitoring networks as a natural application for radio networks where positions of stations are known because they are also needed on the application level. Energy-efficient communication by solving network optimization problems in a graph with transmission power as edge weights is an intensively studied approach. However, so far more complex problems like broadcasting and multi-cast routing have been studied (e.g. [11, 10]) whereas the quadratic complexity of solving the shortest path problem in a complete graph has not been challenged.

An intensive area of research is distributed routing in dynamic situations without complete information. Unfortunately, the worst case performance of such systems is not encouraging. Therefore we believe that research should be driven from both sides, namely : making pessimistic distributed models more powerful and making static models with complete information more dynamic. We view our work as a contribution in the latter direction because faster routing means that a centralized scheduler can handle more routing requests.

Although we are mainly applying computational geometry to radio communication, one could also take the more abstract point of view that we are interested in solving fundamental network problems in implicitly defined (ge-

[3]Powers $\rho < 2$ are only relevant for nowadays rather special cases like short wave transmission with reflections on the ionosphere.

ometric) graphs. There are results in this direction on spanning trees [7] and matchings [9]. The equally fundamental problem of finding shortest paths has not been considered so far, perhaps because the question is trivial for a cost measure that is linear in the Euclidean distance.

2 Zero Offsets

Theorem 1 *Consider a complete geometric graph G defined by n nodes in a plane with edge weights $f(|uv|^2)$ where $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ has the property $f(x+y) \geq f(x) + f(y)$. The single source shortest path problem for G can be solved in time $\mathcal{O}(n \log n)$.*

PROOF: In Lemma 1 we show that we can restrict the shortest path computation to the subgraph defined by a Delaunay triangulation of the sites. A Delaunay triangulation can be computed in time $\mathcal{O}(n \log n)$ and it contains only $\mathcal{O}(n)$ edges [7]. Hence running Dijkstra's shortest path algorithm on this subgraph takes time $\mathcal{O}(n \log n)$. Thus, the overall running time is $\mathcal{O}(n \log n)$. ■

Lemma 1 *Consider any Delaunay triangulation[4] D of V, any source node s and any target node t. There is a shortest path from s to t in the complete graph G using only edges from D.*

PROOF: Among all shortest paths from s to t in G consider a path $P = \langle s = v_0, v_1, \ldots, v_k = t \rangle$ that maximizes the number of hops k. We claim that P contains only edges of the Delaunay triangulation D. Assume the contrary and let (xy) denote an edge on P that is not in D. Consider the disk C of smallest radius with x and y on its boundary. It is well known that C must contain a third point $z \in V$ (possibly on its boundary). Using elementary geometry one can see that $|xy|^2 \geq |xz|^2 + |zy|^2$ and therefore

$$f(|xy|^2) \geq f(|xz|^2 + |zy|^2) \geq f(|xz|^2) + f(|zy|^2) \ .$$

Hence, replacing the edge (xy) by edges (xz) and (zy) yields another shortest path with a larger number of hops. This contradicts the assumption that P is a shortest path with a maximum number of edges. ■

3 Bounded Number of Hops

There are several reasons why one might prefer routes that traverse only a small number of edges (hops) in the communication graph. If there are more

[4]Only if there are no four co-circular points, the Delaunay triangulations would be unique.

hops, latency might get too large, reliability becomes an issue and distance independent energy consumption in the intermediate nodes may become too large. Therefore, we might be interested in the energy minimal path with the additional constraint that at most k hops should be used. Section 3.1 presents a simple algorithm for the case of two-hop paths with uniform offsets based on 2-dimensional closest point queries and extends this approach to paths with three and four-hops. Then Section 3.2 gives a generalization for k-hop paths with non-uniform offsets that needs more sophisticated and less standard data structures.

3.1 Two, Three, or Four Hops

Theorem 2 *In a complete geometric graph defined by n points in the plane with edge weights $C + |uv|^2$, queries for optimal 2-hop paths can be answered in time $\mathcal{O}(\log n)$ after preprocessing time $\mathcal{O}(n \log n)$ using $\mathcal{O}(n)$ space.*

The basic idea is the observation that two-hop routing can be reduced to a 2-dimensional nearest neighbor query with respect to the mid point of the straight line segment joining the sender and receiver. Three and four hop routing can be reduced to $\mathcal{O}(n)$ two-hop routing problems (using dynamic programming) yielding a very simple $\mathcal{O}(n \log n)$ time algorithm. For detailed proofs refer to the full paper.

For edge weights $C + f(|xz|^2)$ one can use similar geometric data structures as for the case $\rho = 2$ to find optimal routes efficiently. We are not aware of worst-case run-time guarantees but conjecture that such guarantees might be possible for approximately optimal solutions.

3.2 k Hops

For any $p \in P$, let $\mu_k(p)$ denote the cost of the shortest path from the source node s to p using at most k hops. In this section we describe a dynamic programming algorithm that determines $\mu_k(t)$ in time $\mathcal{O}(kn \log n)$ for any $k \leq n-1$. Clearly, $\mu_1(p) = C_s + |sp|^2$. For $i > 1$ the function μ_i can be computed recursively as follows:

$$\mu_{i+1}(p) = \min\{\mu_i(p), \min_{q \in P}\{\mu_i(q) + C_q + |qp|^2\}\}. \tag{1}$$

A trivial implementation of this formula yields an algorithm with running time $\mathcal{O}(kn^2)$: We start with μ_1 and iteratively determine μ_i for increasing i where i is ranging from 2 to k. In each of the $k-1$ phases, equation 1 is applied for all $p \in P$. We can exploit the geometric structure of the problem to speed up the computation.

Theorem 3 *In a complete geometric graph defined by n points in the plane with edge weights $C_u + |uv|^2$, the single source shortest path problem restricted to paths with at most k-hops can be solved in time $\mathcal{O}(kn \log n)$ using space $\mathcal{O}(n)$.*

PROOF: Applying Lemma 2 and 4, $\mu_{i+1}(p)$ can be computed from μ_i for all $p \in P$ in time $\mathcal{O}(n \log n)$. This corresponds to one phase of the algorithm. For all $k-1$ phases, a total of $\mathcal{O}(kn \log n)$ time is sufficient. ■

The problem of finding the point $q \in P$ that minimizes the expression $\mu_i(q) + C_q + |pq|^2$ on the right hand side of equation 1 can be reduced to a 3D nearest neighbor query. Consider the points P embedded into the plane $z = 0$ in 3-dimensional space. Thus a point $p \in P$ has the form $p = (x_p, y_p, 0) \in \mathbb{R}^3$. We code the cost of a i-hop path from s to p as well as the offset cost C_p using the third dimension: Define $f_i : P \to \mathbb{R}^3; f_i(x_p, y_p, 0) := (x_p, y_p, \sqrt{\mu_i(p) + C_p})$. Let $f_i(P) := \{f_i(p) : p \in P\}$. Since the vector from p to $f_i(p)$ is orthogonal to the plane $z = 0$ we have for each pair $p, q \in P$:

$$|f_i(q)p|^2 = |qp|^2 + \left(\sqrt{\mu_i(q) + C_q}\right)^2 = \mu_i(q) + C_q + |qp|^2 .$$

Let $p \in P$ be fixed and choose $q \in P$ such that $f_i(q)$ is closest to p among the points in $f_i(P)$. Then q minimizes the expression $\mu_i(q) + C_q + |pq|^2$. Using equation 1 we can compute $\mu_{i+1}(p)$ by answering one nearest neighbor query for p. This justifies the following lemma:

Lemma 2 *The problem of computing $\mu_{i+1}(p)$ reduces to an instance of the nearest neighbor problem with sites $f_i(P)$ and query point $p = (p_x, p_y, 0)$.*

A standard approach for solving the nearest neighbor problem uses Voronoi diagrams together with a point location algorithm. The worst case combinatorial complexity of the Voronoi diagram $\mathcal{V}(S)$ for a 3D point set S with n elements is $\Theta(n^2)$. Therefore, a straightforward application does not improve the asymptotic running time of our algorithm. Note however that for our purposes all query points lie in the plane $e := (z = 0)$. Hence it suffices to compute the subdivision of plane e that is induced by the Voronoi diagram $\mathcal{V}(f_i(P))$. Let $\mathcal{V}_e$ denote this subdivision of e which is very similar to a 2D Voronoi diagram except that there might be less than n cells (faces). In the following we view $\mathcal{V}_e$ as the intersection of $\mathcal{V}(f_i(P))$ and plane e. The worst case combinatorial complexity of $\mathcal{V}_e$ is only linear in the number of points in P.This can easily be verified using Euler's formula for planar graphs.

In the following we give an algorithm for computing $\mathcal{V}_e(S)$ in time $\mathcal{O}(n \log n)$. We use the well known relationship between Voronoi diagrams and the intersection of upper half spaces [2]. Given a set S of n points in $\mathbb{R}^3$ we map each

point $p = (p_x, p_y, p_z) \in S$ to the hyper-plane h_p in $\mathbb{R}^4$ which is the graph of the 3-variate linear function

$$h_p(x, y, z) = 2p_x x + 2p_y y + 2p_z z - (p_x^2 + p_y^2 + p_z^2).$$

A given query point $q = (q_x, q_y, q_z)$ is closest to $p \in S$ (among all points in S) if and only if

$$p = \arg\max_{s \in S} h_s(q_x, q_y, q_z).$$

This can be interpreted geometrically in 4-dimensional space with dimensions x, y, z and v as the height[5]: Consider the half spaces bounded from below (with respect to v) by the hyper-planes $H(S) := \{h(s) : s \in S)\}$. The upper envelope $\mathcal{UE}(H(S))$ of $H(S)$ is defined as the boundary of the intersection of these half spaces which is the graph of the 3-variate function $f(x, y, z) = \max_{s \in S} h_s(x, y, z)$. Hence $p \in S$ is a nearest neighbor to query point q if and only if $f(q) = h_p(q)$. In this case h_p is a supporting hyper-plane that contributes a facet to $\mathcal{UE}(H(S))$. Consequently, the orthogonal projection of $\mathcal{UE}(H(S))$ to the hyper-plane $v = 0$ is the Voronoi diagram $\mathcal{V}(S)$. For our purposes we need to intersect the Voronoi diagram of the 3D point set $f_i(P)$ with the hyper-plane $(z = 0)$. The worst case complexity of the resulting subdivision $\mathcal{V}_e$ and $\mathcal{V}(f_i(P))$ differ by a factor of n. To avoid computing unnecessary information, we schedule the transition to the subspace $z = 0$ before computing the upper envelope and obtain the following algorithm:

Algorithm 1

1. Compute set of hyper-planes $H = H(f_i(P))$.
2. Intersect each hyper-plane $h \in H$ with the hyper-plane $z = 0$ to get a set of 2D hyper-planes H_z.
3. Compute the upper envelope $\mathcal{UE}(H_z)$.
4. Project $\mathcal{UE}(H_z)$ onto the xy-plane $(v = z = 0)$.

Lemma 3 *Algorithm 1 computes $\mathcal{V}_e(f_i(P))$ correctly and has running time $\mathcal{O}(n \log n)$.*

PROOF: Let $q = (q_x, q_y, 0)$ be any query point on the xy-plane lying in the Voronoi cell of some site $p \in f_i(P) = S$ (i.e, p is a nearest neighbor for q). So we have $h_p(q) = \max_{s \in S} h_s(q)$. Note that for any $s \in S$ the 2D hyper-plane $h_s^z \in H_z$ is the graph of a 2-variate $h_s^z(x, y) = h_s(x, y, 0)$. Hence, for q, $h_p(q) = h_p^z(q_x, q_y) = \max_{s \in S}\{h_s^z(q_x, q_y)\}$. This means that h_p^z contributes a facet

[5]So we can use the notion of above and below.

to $\mathcal{UE}(H_z)$. Moreover, in the projection of the $\mathcal{UE}(H_z)$ to the xy-plane, q is part of the Voronoi region that corresponds to site p.

Since the worst case combinatorial complexity of $\mathcal{UE}(H_z)$ is $\mathcal{O}(n)$, it can be computed in time $\mathcal{O}(n \log n)$ [2]. It is easy to verify that all other steps of algorithm 1 can be done in linear time. ■

Having computed $\mathcal{V}_e(f_i(P))$, we can make use of a point location algorithm to answer 2D nearest neighbor queries efficiently.

Lemma 4 *Let P be a set of n points in $\mathbb{R}^3$. Allowing $\mathcal{O}(n \log n)$ preprocessing time, the nearest neighbor problem for sites P and query points lying in the xy-plane can be answered in time $\mathcal{O}(\log n)$.*

4 The General Case

In this section we consider the case of computing the shortest path without any restrictions in the number of hops used. Since all edge weights are non-negative, this corresponds to computing $\mu_{n-1}(t)$ for target node t. Using the results of the last section one could solve the problem in time $\mathcal{O}(n^2 \log n)$ but this would be even worse than applying Dijkstra's algorithm to the complete graph. Therefore, we now develop a different algorithm that is similar to Dijkstra's algorithm in that it settles nodes in order of the distance from the source. But edges are never explicitly relaxed. Instead a powerful geometric data structure replaces both edge relaxation and the priority queue data structure of Dijkstra's algorithm.

Theorem 4 *In a complete geometric graph defined by n points in the plane with edge weights $C_u + |uv|^2$, the single source shortest path problem can be solved in $\mathcal{O}(n^{1+\epsilon})$ time and space for any constant $\epsilon > 0$.*

Let $\mu(v) = \mu_{n-1}(v)$ denote the length of a shortest path from source s to $v \in V$. During the execution of the algorithm each node is either reached (red) or unreached (blue). Initially all nodes except the source node are blue. Nodes change their color from blue to red in order of increasing $\mu()$ values. When a node becomes red, its $\mu()$ value is known. Let R and B denote the set of red and blue nodes respectively. In each iteration, we determine a blue node with smallest distance to the source:

$$v = \arg \min_{b \in B} \min_{r \in R} \{\mu(r) + |rb|^2 + C_r\} \tag{2}$$

This can be modeled as a closest bichromatic point problem using one additional dimension. We define

$$f : R \longrightarrow \mathbb{R}^3; \qquad f(r_x, r_y, 0) = (r_x, r_y, \sqrt{\mu(r) + C_r}).$$

For any red-blue pair $(r, b) \in R \times B$, we have $|f(r)b|^2 = |rb|^2 + \sqrt{\mu(r) + C_r}^2$. Let $f(R) = \{f(r) : r \in R\}$. Then the closest bichromatic pair between the sets $f(R)$ and B gives the solution to Equation 2. Algorithm 2 states the pseudo code for the general case.

Algorithm 2

$B \leftarrow V \backslash \{s\}$, $R \leftarrow \{s\}$, $\mu[s] \leftarrow 0$;
while $(B \neq \emptyset)$
  Find closest bichromatic pair (r, b) between sets $f(R)$ and B.
  $R \leftarrow R \cup \{b\}$, $B \leftarrow B \backslash \{b\}$, $\mu[b] \leftarrow |f(r)b|^2$;

Correctness can be shown by induction with the following invariants: For any red-blue pair (r, b) we have $\mu(r) \leq \mu(b)$ and the $\mu()$-values of the red nodes are determined correctly. When a point p is discovered then Equation 2 ensures that all the remaining blue points have a $\mu()$-value at least as large as $\mu(p)$. Since edge weights are strictly positive, there cannot be a blue point on a shortest $s \rightarrow p$ path and hence $\mu(p)$ is computed correctly. The dynamic bichromatic closest points problems we are solving have the special property that all blue points are lying in the xy-plane. This enables us to speed up the computation compared to the general 3D dynamic closest pair problem.

Corollary 1 *In a complete geometric graph defined by n points in the plane with edge weights $C_u + |uv|^2$, the single source shortest path problem can be reduced to answer a sequence of $n - 1$ closest bichromatic neighbor queries for dynamically changing point sets $R, B \in \mathbb{R}^3$ of cumulative size n where all points in B are lying in the xy-plane.*

An algorithm for solving the dynamic bichromatic closest pair problem is given by Eppstein [3]. It is based on bichromatic ordered nearest neighbor paths. The running time only depends on the time to answer two kinds of nearest neighbor queries for dynamically changing sets of sites. The first type asks for the nearest blue point for a given red point. In the second type, the roles of the red and blue sets are interchanged. In our application, the blue set is basically a 2D point set. So we distinguish two types of nearest neighbor queries for a set of sites S and query point q:

1. Sites are restricted to the xy-plane ($S \subset \mathbb{R}^2$, $q \in \mathbb{R}^3$).

2. Queries are restricted to the xy-plane ($S \subset \mathbb{R}^3$, $q \in \mathbb{R}^2$).

We show that these special types of the 3D problem can be regarded as 2D nearest neighbor problems. Consider the first type. Given a query point $q = (q_x, q_y, q_z)$ we simply find the nearest neighbor for $q' = (q_x, q_y, 0)$. It is easy to verify that this is also a solution for q.

The second type of nearest neighbor queries was already considered in the last subsection for a static set of sites. There we used the relation between Voronoi diagrams and the upper envelope of a certain set of hyper-planes. Here again we can exploit this relation. Instead of precomputing the upper envelope, we determine for each individual query point $q = (q_x, q_y, 0, 0)$ the first hyper-plane that is intersected by the ray emanating from the point $(q_x, q_y, 0, +\infty)$ in $-v$ direction (see [1]). Since all query points lie in xy-plane, the corresponding query rays have z=0. Hence the problem complexity reduces by one dimension and we only need to maintain a set of 2D planes in $\mathbb{R}^3$ (with dimensions x, y, v). Agarwal and Matoušek [1] describe algorithms for dynamic ray shooting problems.

Lemma 5 *Given a convex polytope in $\mathbb{R}^3$ described as the intersection of n half-spaces, one can process it in time $\mathcal{O}(n^{1+\epsilon})$ into a data structure of size $\mathcal{O}(n^{1+\epsilon})$, so that the first point of the polytope boundary hit by a query ray can be determined in $\mathcal{O}\big(\log^5 n\big)$ time. The data structure can be maintained dynamically in amortized time $\mathcal{O}(n^{\epsilon})$ per insert/delete operation.*

Combined with the result of Eppstein [3] for maintaining the bichromatic closest pair, this proves Theorem 4. We showed how to solve the problem by answering a sequence of closest bichromatic point queries for dynamic sets R and B. These queries have more special properties than we have exploited:

1. We never insert points into set B.
2. We never delete points from set R.

Our application is even more specific when we view a deletion followed by an insertion as one operation, which is then the only operation we need: A node turns its color from blue to red and is lifted up in the third dimension keeping its x- and y-coordinates. This might be a starting points for finding more practical algorithms.

5 Generalizations

In the full paper, we explain how the following generalizations can be achieved: Even if the transmitters direct their power into a particular direction, the energy consumption is still proportional to the square of the distance to the receiver as long as the transmitter is only a constant factor larger than the wave length of the signal. The two-hop routing algorithm from Theorem 2 can be generalized to situations where the set of points changes dynamically with somewhat worse running times. The four hop routing algorithm from section 3.1 can refined so that it will run in sublinear time for many realistic inputs (not counting preprocessing).

Computing minimum spanning trees is even more robust than computing shortest paths. The results from [7] for linear cost functions generalize to all edges costs that are monotonous functions of the Euclidean distance. Although minimum spanning trees do *not* yield optimum broadcast trees they do yield a 12-approximation [10]. Perhaps more importantly, minimum spanning trees can be used for optimal *reduction*, i.e., distributed evaluation of an expression that uses an associative operator. We explain how minimum spanning trees can be computed in case of nonuniform offsets C_u.

6 Discussion

We have proposed efficient algorithms for energy and bandwidth efficient communication in a model designed for simplicity and resemblance to the actual physical conditions. The problems of communication in real radio networks are vastly more complicated and probably do not allow similarly clean solutions. However, we believe that the algorithmic principles developed may nevertheless remain useful. In the full paper we discuss some complications not grasped by our model and explain how they may be tackled without compromising our basic approach. For example, even if cost functions are *anisotropic*, i.e. do not only depend on the distance between two points but also on refraction, reflection and absorption by obstacles, we usually have so little information that we have to use an isotropic simplification of the cost function.

References

[1] P. D. Agarwal and J. Matousek. Ray shooting and parametric search. In N. Alon, editor, *Proceedings of the 24th Annual ACM Symposium on the Theory of Computing*, pages 517–526, Victoria, B.C., Canada, May 1992. ACM Press.

[2] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry Algorithms and Applications*. Springer-Verlag, Berlin Heidelberg, 2., rev. ed. edition, 2000.

[3] D. Eppstein. Dynamic euclidean minimum spanning trees and extrema of binary functions. *Disc. Comp. Geom.*, 13:111–122, 1995.

[4] J. M. Rabaey et al. Picoradio supports ad hoc ultra-low power wireless networking. *IEEE Computer Magazine*, pages 42–48, July 2000.

[5] K. Nakano, S. Olariu, and A. Y. Zomaya. Energy-efficient permutation routing in radio networks. *IEEE Transactions on Parallel and Distributed Systems*, 12(6):544–557, 2001.

[6] D. Patel. Energy in ad-hoc networking for the picoradio. Master's thesis, UC Berkeley, 2000.

[7] F. P. Preparata and M. I. Shamos. *Computational Geometry*. Springer, 1985.

[8] T. S. Rappaport. *Wireless Communication*. Prentice Hall, 1996.

[9] K. R. Varadarajan and P. K. Agarwal. Approximation algorithms for bipartite and non-bipartite matching in the plane. In *SODA*, pages 805–814, 1999.

[10] Peng-Jun Wan, G. Calinescu, Xiang-Yang Li, and Ophir Frieder. Minimum-energy broadcast routing in static ad hoc wireless networks. In *IEEE Infocom*, 2001.

[11] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides. On the construction of energy-efficient broadcast and multicast trees in wireless networks. In *IEEE Infocom*, volume 2, pages 585–594. IEEE, 2000.