Department of Computer Science
Faculty of Mathematics, Physics and Informatics
Comenius University in Bratislava

Broadcast in radio networks

(Master's thesis)

Lukáš Poláček

Advisor: Doc. RNDr. Rastislav Královič, PhD.

Bratislava, 2009

Čestne prehlasujem, že som túto diplomovú prácu vypracoval samostatne s použitím citovaných zdrojov.

.....

Abstract

We consider deterministic broadcasting in radio networks, where nodes can transmit information. A signal from a node can reach other nodes, but this relation does not have to be symmetric – if node u can reach v, node v does not have to be able to reach u. Communication is synchronous and if two or more neighbours of a node transmit to the node in one round, collision occurs and the node hears nothing.

Finding optimal broadcasting time for a network is known to be NP-hard. We introduce three algorithms with different complexities that find optimal broadcasting time for a network. The second part of the thesis focuses on approximation algorithms in geometric radio networks. We present a global algorithm and a distributed broadcasting algorithm where nodes have knowledge of all nodes within a certain distance.

Keywords: Broadcast, radio networks, distributed algorithms, approximation algorithms

Contents

List of Figures

Chapter 1

Introduction

Broadcasting is one of the most important network communication primitives. One node of the network, called the *source*, has to transmit a message to all other nodes. Remote nodes are informed via intermediate nodes, along directed paths in the network. One of the basic performance measures of a broadcasting scheme is the total time, i.e., the number of rounds it uses to inform all the nodes of the network. The aim of this thesis is to develop algorithms that find optimal broadcast scheme and approximation algorithms that find a broadcast scheme with bounded broadcasting time.

1.1 Terminology and model description

In the entire thesis we will use the model that was used e.g. in [DP07], [IKP08] and we will also use some definitions from [SW06]. In [DP07] and [IKP08] authors fixed a set of ranges $\{r_1, \ldots, r_\rho\}$ in the model of a geometric radio network. We consider the ranges of a network to be a property of a radio network, instead of the model.

Throughout the thesis, if a node u is within a node v's transmission range, we say that u is *reachable* from v. We say that v is an *in-neighbour* of u and u is an *out-neighbour* of v. These relations can be modelled by a directed graph, where nodes are vertices. There is an edge (u, v) if v is reachable from u. The network modelled by a general directed graph is called a *radio network*.

In the case of an approximately flat region without large obstacles, nodes that can be reached from u are those within a circle of radius r centered at u, and the positive real r, called the *range* of u, depends on the power of the transmitter located at u. These networks are called *geometric radio networks* (GRN). An example of such a network can be seen on figure 1.1.

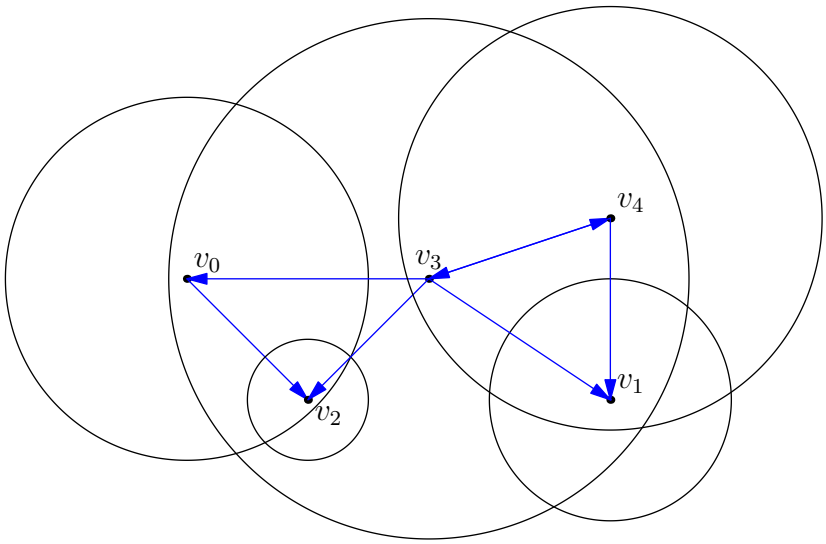

Figure 1.1: Geometric radio network

Nodes send messages in synchronous *rounds*. All clocks in nodes show the same time at any given time. In every round every node acts either as a *transmitter* or as a *receiver*. A node gets a message in a given round, if and only if, it acts as a receiver and exactly one of its neighbours transmits in this round. The message received in this case is the one that was transmitted. If at least two neighbours of a receiving node u transmit simultaneously in a given round, none of the messages is received by u in this round. In this case we say that a *collision* occurred at u.

An algorithm that can operate on the entire network is called *global*. On the other hand, in *distributed* algorithm, every node runs its own algorithm. For a fixed real $s \geq 0$, called the *knowledge radius*, we assume that each node in GRN knows the part of the network within the circle of radius s centered at it, i.e., it knows the positions, labels and ranges of all nodes at distance at most s.

1.2 Previous work and our results

It was already proved in [SH97] that establishing optimal broadcasting time is NP-hard problem even in geometric radio networks. Obvious lower bound for broadcasting time is the maximum distance of a node from the source, which is called the *eccentricity* of the source. In the entire thesis the eccentricity will be denoted by number D. In [ABNLP91] authors proved that there

exist a family of N-node networks of radius 2, such that any deterministic broadcasting algorithm requires time $\Omega(\log^2 N)$. It follows that the lower bound for broadcasting time is $\Omega(D + \log^2 N)$. For symmetric networks, deterministic algorithm to produce broadcasting scheme with broadcasting time $O(D + \log^2 N)$ was published in [KDPA07].

Various other scenarios have been studied. In [BDP97, CGG$^+$02, DP07] authors studied the impact of the zero knowledge on the broadcast time, i.e. they studied the situation where nodes know only their own label and in case the network is GRN, also its absolute position. Graphs that require broadcasting time $\Omega(N \log N)$ have been showed in [BDP97, CGG$^+$02], but these graphs are not GRN. Algorithm with broadcasting time $O(N(\frac{r_{max}}{r_{min}})^4)$ for GRN has been published in [DP07].

Another interesting topic is the recently introduced framework of network algorithms with advice. The first paper which studied this framework in radio networks was [IKP08]. For arbitrary radio networks they showed a trade-off between the size of advice and the time of deterministic broadcasting, by presenting a broadcasting algorithm whose time is inverse-proportional to the size of advice. More precisely, for any $q \in O(N)$ they showed an oracle which gives advice of size q to the nodes of a network, and an algorithm using this advice, which performs broadcasting in time $O(\frac{ND}{q} \log^3 N)$.

This thesis is the first work that studies exact broadcasting algorithms. In the second chapter we introduce three algorithms. The first one works in $O(M3^N)$ time and $O(N2^N)$ space, where M is the number of edges in the graph of the network. The second algorithm works in $2^{N^2/2+O(N)}$ time and $O(N^2)$ space. Finally, the third algorithm works in time $2^{N^2/2K+O(N)}$ and $O(N^K)$ space for a constant $K \geq 2$. This algorithm is an extension of the second algorithm parametrized by a constant K.

The third chapter examines approximation algorithms in GRN. Broadcasting in GRN was considered, e.g., in [RS94, SH97, DP07]. In [SH97] the authors proved that scheduling optimal broadcasting is NP-hard even when restricted to such graphs, and gave an $O(N \log N)$ algorithm to schedule an optimal broadcast when nodes are situated on a line. In [RS94] broadcasting was considered with nodes randomly placed on a line. Our work is based mainly on the broadcasting algorithm published in [DP07], whose broadcasting time is $O(D(\frac{r_{max}}{r_{min}})^4)$ where D is the eccentricity of the source node, r_{max} is the maximal range and r_{min} is the minimal range of the nodes. Authors fix numbers r_{max} and r_{min} in the model of GRN, thus they treat these numbers as constants. They claim that the broadcasting time is $O(D)$, but if r_{max} and r_{min} are not constants, the broadcasting time is $O(D(\frac{r_{max}}{r_{min}})^4)$.

We present a global algorithm, which finds broadcast scheme with broadcasting time $O(D(\frac{r_{max}}{r_{min}})^3)$. The second algorithm is a distributed algorithm,

where knowledge radius is $r_{max} + r_{min}$, i.e. nodes know the labels, positions and ranges of all nodes within range $r_{min} + r_{max}$. The broadcasting time of this algorithm is 8 times longer than the broadcasting time of the previous algorithm, which is also $O(D(\frac{r_{max}}{r_{min}})^3)$.

1.3 Useful definitions

Let $a_1, \ldots, a_N$ and $b_1, \ldots, b_N$ be two integer sequences. We say that $a_1, \ldots, a_N$ is *lexicographically smaller* than $b_1, \ldots, b_N$ if there exists j such that $a_j < b_j$ and $a_i = b_i$ for all $i < j$.

Let $A = \{a_1, \ldots, a_N\}$ be a finite set, where $a_1 < a_2 < \cdots < a_N$ and let $X = \{x_1, \ldots, x_k\} \subseteq A; x_1 < \cdots < x_k$ and $Y = \{y_1, \ldots, y_l\} \subseteq A; y_1 < \cdots < y_l$ be subsets of A. Let $s_1, \ldots, s_N$ and $t_1, \ldots, t_N$ be integer sequences such that $s_i = [a_i \in X]$ and $t_i = [a_i \in Y]$. Here $[P]$ is Iverson's bracket that was used e.g. in [GKP94]. If P is true then $[P] = 1$ and $[P] = 0$ otherwise. We say that X is *lexicographically smaller* than Y if sequence $s_1, \ldots, s_N$ is lexicographically smaller than $t_1, \ldots, t_N$.

Chapter 2

Exact Algorithms

In this chapter we will discuss algorithms that find optimal broadcasting scheme for a network. It was already proved in [SH97] that finding optimal broadcasting scheme is NP-hard even for geometric radio networks. We will present three algorithms with different complexities. All three work for arbitrary radio networks. The first one finds optimal broadcasting scheme in $O(M3^N)$ time and $O(N2^N)$ space, where N is the number of nodes and M is the number of edges. The second one finds optimal broadcasting scheme in $2^{N^2/2+O(N)}$ time and $O(N^2)$ space. Both algorithms work similarly, but the main difference is that the second one caches values. The third algorithm is a mix of both ideas. It caches the most time consuming procedure calls, which results in time complexity $2^{N^2/2K+O(N)}$ and memory complexity $O(N^K)$ for a fixed constant $K \geq 2$. We see that the performance is scalable—the more values we cache the faster is the algorithm.

In this chapter we will use the following notation. Denote by $G = (V, E), |V| = N, |E| = M$ the graph of the radio network and denote by s the source node. Since we assume that all nodes can eventually hear the message, the graph should be connected, hence the number of edges is at least $N - 1$, thus $N = O(M)$ and $O(M + N) = O(M)$. Denote by G_T the set of nodes in $V \setminus T$ that have exactly one neighbour in T.

Lemma 2.0.1. *There exists an algorithm with time complexity $O(M)$ and with memory complexity $O(N)$ that finds the set G_X for a given graph G and set X (the size of the input is not included in the memory complexity).*

Proof. For every node in $V \setminus X$ we need to calculate the number of neighbours in X. At the beginning we set the number of neighbours for every node to 0. We iterate through the set of edges and when we encounter an edge (u, v) such that $u \in X, v \in V \setminus X$ we increase the number of neighbours of v. At the end of the algorithm we select all nodes that have only one neighbour.

The slowest part is the iteration through the set of edges, which takes $O(M)$ operations. We need to store the number of the neighbour for each node, hence the memory complexity is $O(N)$. □

2.1 Exponential algorithm

This algorithm uses *dynamic programming* technique. Let $V, |V| = N$ is the set of nodes of the network and s is the source. For each $S \subseteq V$ we will compute the time of the optimal broadcasting scheme. Denote this number by $p(S)$. Then

$$p(\{s\}) = 0$$

$$p(S) = \min(\{\ p(Y) + 1 \mid X \subseteq Y \subset S \wedge G_X \cup Y = S\ \} \cup \{\ \infty\ \}) \qquad (2.1)$$

The second equation describes one step of the broadcasting scheme. Suppose that all nodes in Y know the source message. We choose some nodes from the set Y and these nodes transmit a message. Denote the set of these nodes by X. The set G_X contains all nodes in V that have exactly one neighbour in X, i.e. these are the nodes that will hear the message when all nodes in X transmit at once. Therefore in the next step nodes $Y \cup G_X$ will all be informed and the optimal broadcast time for $Y \cup G_X$ is at most $p(Y) + 1$.

Straightforward implementation of equations (2.1) can lead to an algorithm with time complexity $O(M4^N)$. Let $\mathcal{P}(V)$ be the power set of V and let $A_1, A_2, \ldots, A_{2^N}$ be lexicographical ordering of the set $\mathcal{P}(V)$. Algorithm 2.1.1 is an implementation with time complexity $O(M3^N)$.

Algorithm 2.1.1 Exponential algorithm

Require: $G = (V, E), s \in V$
Ensure: $(\forall S \subseteq V) r(S) = p(S)$
1: **for all** $Y \in \mathcal{P}(V)$ **do**
2: $\quad r(Y) \leftarrow \infty$
3: **end for**
4: $r(\{s\}) \leftarrow 0$
5: $r(\emptyset) \leftarrow 0$
6: **for** $i \leftarrow 1$ to 2^N **do**
7: $\quad$ **for all** $X \in \mathcal{P}(A_i)$ **do**
8: $\quad\quad r(A_i \cup G_X) \leftarrow \min\{r(A_i \cup G_X), r(A_i) + 1\}$
9: $\quad$ **end for**
10: **end for**
11: **return** $r(V)$

Theorem 2.1.1. *Algorithm 2.1.1 is correct.*

Proof. We will prove by induction that when i is equal to j, $r(A_l) = p(A_l)$ for all $l \leq j$. For $i = 1$ the statement holds, because $p(\emptyset) = \infty$ and $r(\infty)$ will also be equal to ∞. Suppose that the statement holds for every number that is at most j. Let the optimal broadcasting scheme for A_{j+1} be the sequence $\{s\} = B_1, B_2, \ldots, B_k$, where B_l denotes the set of nodes that transmit in the l-th step. By the $(k-1)$-th step, nodes $C = \{s\} \cup \bigcup_{l=1}^{k-1} G_{B_l}$ are informed, thus the optimal broadcast time for A_{j+1} is $p(C)+1$ and $A_{j+1} = C \cup G_{B_k}$. Since C is a subset of A_{j+1}, it is lexicographically smaller than A_{j+1}. Therefore $C = A_c$ for $c \leq j$. By the induction hypothesis, $r(C)$ was equal to $p(C)$ when $i = c$, hence we performed the operation $r(C \cup G_{B_k}) \leftarrow \min\{r(C \cup G_{B_k}), r(C)+1\}$ before i reached $j+1$. Therefore $r(A_{j+1}) = p(A_{j+1})$. □

Theorem 2.1.2. *Time complexity of algorithm 2.1.1 is $O(M3^N)$ and memory complexity is $O(N2^N)$.*

Proof. Since every subset of V can be encoded as a bit sequence of length N, we assume that access to array p is in time N. The first 3 lines perform $O(N2^N)$ operations. Line 6 iterates through every subset and then line 7 iterates through every subset of this subset. Set of size k has 2^k subsets. There are $\binom{N}{k}$ subsets of V of size k, hence the number of executions of line 8 is

$$\sum_{k=0}^{N} \binom{N}{k} 2^k = (1+2)^N = 3^N$$

In line 8 we need to compute the set G_X, which can be done according to lemma 2.0.1 in $O(M)$ time. Hence the time complexity of this line is $O(M)$ and overall complexity is $O(M3^N)$. We need to store the whole array p in the memory and we need N bits to store each set, therefore the memory complexity is $O(N2^N)$. □

2.1.1 C++ implementation

Implementation of line 7 can be very easy in real programming languages. As we mentioned earlier, we will store every subset T of $V = \{v_1, \ldots, v_N\}$ as a bit sequence such that i-th bit is 1 if and only if $v_i \in T$. Thus we need N bits to store every subset. In this C++ implementation we use unsigned 32-bit numbers to store sets, which limits the usage of the program to $N \leq 32$. Let t be a bit sequence corresponding to set T. Then this for-cycle iterates through all subsets of T:

```
for (unsigned j = t; j > 0; j = (j - 1) & t)
```

The cycle starts with the set T represented by bit sequence t. The set following after set j is determined by the operation `(j - 1) & t`. Let $J = \{v_{u_1}, \ldots, v_{u_k}\}$ is the set represented by bit sequence j. Sequence j contains ones only at positions $u_1, u_2, \ldots, u_k$. The positions are listed from the least significant bit to the most significant one. What does `(j - 1) & t` do? The bit sequence `j - 1` has ones at positions $1, 2, \ldots, u_1 - 1, u_2, u_3, \ldots, u_k$, i.e. the least significant one at position u_1 was changed to zero and all lower bits were set to one. This bit sequence corresponds to the lexicographically smaller subset of V nearest to J. Operation `&` is the same as an intersection of two sets, hence the set represented by `(j - 1) & t` is the lexicographically smaller subset of T nearest to J. Thus the for-cycle iterates through all subsets of T in lexicographical order from the biggest to the smallest.

Full C++ implementation of algorithm 2.1.1 follows:

```
#include <iostream>
#include <bitset>
#include <vector>
using namespace std;

const int size = 32;
void decrease(int &a, int b)
{
    a = min(a, b);
}
int main()
{
    int n, m, s;
    cin >> n >> m >> s;
    bitset<size> g[size];
    for (int i = 0; i < m; i++)
    {
        int a, b;
        cin >> a >> b;
        a--; b--;
        g[a][b] = true;
    }

    unsigned count = (1u << n) - 1;
    vector<int> p(count, n + 1);
    p[1 << (s - 1)] = 0;
    for (unsigned k = 0; k <= count; k++) if (p[k] < n + 1)
        for (unsigned j = k; j > 0; j = (j - 1) & k)
        {
```

```
            bitset<size> b(j), q;
            for (int i = 0; i < n; i++)
            {
                int num = 0;
                for (int l = 0; l < n && num < 2; l++)
                    if (b[l] && g[l][i])
                        num++;
                if (num == 1)
                    q[i] = true;
            }
            decrease(p[q.to_ulong() | k], p[k] + 1);
        }
    cout << p[count] << endl;
}
```

2.2 Backtracking algorithm

We will use backtracking to calculate the result with minimum memory used. Let $Y \subseteq V$ be a set of nodes that have already heard the source message and $q(Y)$ be the shortest time to inform the rest of the network $V \setminus Y$. Procedure 2.2.1 calculates value $q(Y)$ for a set Y and algorithm 2.2.2 calculates optimal broadcast time for a radio network.

Procedure 2.2.1 $OptimalTime(Y)$

Require: $G = (V, E), Y \subseteq V$
Ensure: $v = q(Y)$
1: **if** $Y = V$ **then**
2: $v \leftarrow 0$
3: **else**
4: $v \leftarrow \infty$
5: **for all** $X \subseteq Y$ **do**
6: **if** $G_X \neq \emptyset$ **then** {to avoid infinite loop}
7: $v \leftarrow \min\{v, OptimalTime(Y \cup G_X) + 1\}$
8: **end if**
9: **end for**
10: **end if**
11: **return** v

Theorem 2.2.1. *Algorithm 2.2.2 calculates the optimal broadcast time for a radio network.*

Algorithm 2.2.2

Require: $G = (V, E), s \in V$
Ensure: Return value is $q(\{s\})$
1: **return** $OptimalTime(\{s\})$

Proof. We will start by proving that the procedure 2.2.1 is correct. If $Y = V$, all nodes are informed, thus the time required to inform the rest of the network is zero. Otherwise we will calculate the result by trying every possible transmission. If all nodes in X transmit, nodes in G_X will get the message, because all other nodes have either zero or at least two neighbours in X, i.e. they either hear nothing or collision occurs. Hence the value $q(Y)$ is the minimum of all values $q(Y \cup G_X) + 1$ for all $X \subseteq Y$ and the procedure is correct.

The algorithm for optimal broadcast time just calls this procedure with argument $\{s\}$, since initially only the source is informed. Since the procedure returns $q(\{s\})$, the algorithm is correct. □

Theorem 2.2.2. *Time complexity of algorithm 2.2.2 is* $2^{N^2/2+O(N)}$ *and memory complexity is* $O(N^2)$.

Proof. Recursive computation can be modelled as a tree where nodes are sets. The root is the set $\{s\}$, because we start the algorithm by calling $OptimalTime(\{s\})$. Children of a node have always greater size than their parent, because in the algorithm we check for condition $G_X \neq \emptyset$. Let $f(i)$ be the maximum number of operations needed to compute $OptimalTime(Y)$ where $|Y| = i$. Since sets of smaller size are parents of the sets of bigger size, we have $f(i) \geq f(i + 1)$ for $0 \leq i < N$. Since there are $2^{|Y|}$ subsets of Y, we will call $OptimalTime(Y \cup G_X)$ at most $2^{|Y|} = 2^i$ times during the computation of $OptimalTime(Y)$. Before each procedure call we need to calculate G_X in time $O(M)$, thus in the worst case we have

$$f(i) = 2^i(f(i + 1) + cM) \tag{2.2}$$

for some positive constant c.

The number of operations performed while computing $OptimalTime(V)$ is N, therefore we have $f(N) = N$. This implies

$$\begin{aligned} f(1) =& 2^1(2^2(2^3(\dots(N2^{N-1} + cM) + \dots) + cM) + cM) \\ =& cM \sum_{j=0}^{N-2} 2^{\sum_{i=1}^{j} i} + N2^{N(N-1)/2} \end{aligned}$$

Since

$$\sum_{j=0}^{N-2} 2^{\sum_{i=1}^{j} i} \leq \sum_{i=1}^{(N-1)(N-2)/2} 2^i = 2^{(N-1)(N-2)/2+1} - 2$$

we can bound $f(i)$ by

$$\begin{aligned} f(i) \leq & cM2^{1+(N-1)(N-2)/2} + N2^{N(N-1)/2} \\ = & 2^{(N-1)(N-2)/2}(2cM + N2^N) \end{aligned} \tag{2.3}$$

Since $2cM \leq 2cN^2 = O(2^N)$, the time complexity of the algorithm is $O(N2^{(N-2)(N-1)/2}2^N) = O(2^{N(N-1)/2+\log_2 N}) = 2^{N^2/2+O(N)}$. In each call of the procedure we use $O(N)$ memory to calculate the set G_X and the depth of the recursion is at most N, thus the overall memory complexity is $O(N^2)$. □

2.3 Algorithm with cache

We have introduced two algorithms for optimal broadcasting scheme in this chapter. The first one has time complexity $O(M3^N)$ and memory complexity $O(N2^N)$. The second one has time complexity $2^{N^2/2+O(N)}$ and memory complexity $O(N^2)$. The main difference between these algorithms is that the second one does not cache any results, while the first caches all results. In this section we will introduce a modification of the second algorithm that will cache some results, which will lead to better time complexity but worse memory complexity. A big advantage of this algorithm is that the complexity is scalable.

We will modify procedure 2.2.1, such that it will remember, which sets G_X it already used in the procedure call $OptimalTime(Y \cup G_X)$. As we mentioned in the discussion about complexity of algorithm 2.2.2, the time needed to compute value $OptimalTime(S)$ in the worst case for a small set is greater than the time needed for a bigger set. Therefore we will remember the smallest sets G_X. The number of the remembered sets will be determined by a fixed constant $K \geq 2$. More precisely, we will remember that we already computed value $OptimalTime(Y \cup G_X)$, if and only if $|G_X| < K$. Since $G_X \subseteq V \setminus Y$ and $|V| = N$, the size of the cache will be

$$S(N - |Y|, K) = \sum_{i=1}^{K-1} \binom{N - |Y|}{i}$$

Since $S(N - |Y|, K) = \Theta(2^{N-|Y|})$ for $K > (N - |Y|)/2$, it is better to use algorithm 2.1.1 in the case when $K > N/2$.

For small Y the size of the cache can be larger than $2^{|Y|}$ and it can happen that the cached values will not be used, thus the performance will be worse than the performance without the cache. Hence we will not use the cache if $S(N-|Y|, K-1)$ is bigger than $2^{|Y|}$.

Procedure 2.3.1 *OptimalTime*(Y)

Require: $G = (V, E), Y \subseteq V$
Ensure: $v = q(Y)$
1: **if** $Y = V$ **then**
2: $v \leftarrow 0$
3: **else**
4: $v \leftarrow \infty$
5: $c \leftarrow [\sum_{i=1}^{K-1} \binom{N-|Y|}{i} \leq 2^{|Y|}]$
6: **for all** $X \subseteq Y$ **do**
7: **if** $G_X \neq \emptyset$ **then** {to avoid infinite loop}
8: **if** $c \wedge |G_X| < K$ **then**
9: **if** G_X is not in the cache **then**
10: $v \leftarrow \min\{v, OptimalTime(Y \cup G_X) + 1\}$
11: insert G_X into the cache
12: **end if**
13: **else**
14: $v \leftarrow \min\{v, OptimalTime(Y \cup G_X) + 1\}$
15: **end if**
16: **end if**
17: **end for**
18: **end if**
19: **return** v

Algorithm 2.3.2

Require: $G = (V, E), s \in V$
Ensure: Return value is $q(\{s\})$
1: **return** $OptimalTime(\{s\})$

Theorem 2.3.1. *Procedure 2.3.1 returns value* $q(Y)$.

Proof. Since this procedure differs from procedure 2.2.1 only in the use of the cache, we need to show that after this change the procedure remains correct. During the procedure call $OptimalTime(Y)$ we will do the operation $v \leftarrow \min\{v, OptimalTime(Y \cup G_X) + 1\}$ only if G_X is not yet in the cache.

The repeated use of the value $OptimalTime(Y \cup G_X) + 1$ has no effect on v, therefore we only need to do this operation once. Hence the procedure is correct. □

Corollary 2.3.1. *Algorithm 2.3.2 is correct.*

Lemma 2.3.1. *Let A be a set $\{a_1, \ldots, a_N\}$ where $a_1 < a_2 < \cdots < a_N$. Let B be a lexicographical ordering of all subsets of A of size at most L. Then there is an algorithm with time complexity $O(L^2)$ that for a given subset of A of size at most L calculates the position in the ordering B.*

Proof. Let $C = \{a_{i_1}, \ldots, a_{i_k}\}$ be a subset of A. Let $c_1, \ldots, c_N$ be a bit sequence corresponding to set C, i.e. $c_m \Leftrightarrow a_m \in C$ or, using Iverson's bracket, $c_m = [a_m \in C]$. By the definition of lexicographical ordering of sets, to calculate the position of C in the ordering B we need to calculate the number of bit sequences of length N that contain at most L ones and are lexicographically smaller than $c_1, \ldots, c_N$. A bit sequence $d_1, \ldots, d_N$ is smaller than $c_1, \ldots, c_N$ if there is j such that $d_j < c_j$ and $d_m = c_m$ for all m less than j. If $c_j = 0$ then there is no d_j such that $d_j < c_j$, therefore the only interesting case is when $c_j = 1$ and $d_j = 0$. Let us count the number of such sequences $d_1, \ldots, d_N$. Since the first $j - 1$ values are the same as sequence $c_1, \ldots, c_{j-1}$ and $d_j = 0$, the only values that are not fixed are at positions beyond position j.

By the definition of $C = \{a_{i_1}, \ldots, a_{i_k}\}$, the number j is equal to some i_l. Thus we can place at most $L - l$ ones at positions $j + 1, j + 2, \ldots, N$. The number of ways how this can be done is

$$\sum_{m=0}^{L-l} \binom{N - i_l}{m}$$

To count the final result we should sum the above result through all $l \in \{1, \ldots, k\}$, thus we get

$$\sum_{l=1}^{k} \sum_{m=0}^{L-l} \binom{N - i_l}{m}$$

We assume that binomial coefficients are already computed and stored in a two-dimensional array. Hence the time complexity to compute the above sum is $k \cdot L \leq L^2 = O(L^2)$, thus the time complexity of the whole algorithm is $O(L^2)$. □

Theorem 2.3.2. *The time complexity of algorithm 2.3.2 is $2^{N^2/2K+O(N)}$ and the memory complexity is $O(N^K)$.*

Proof. According to [GKP94] there is no closed form for the partial sum of a row of a Pascal's triangle. But we can bound the size of the cache by using the simple inequality

$$\binom{N-|Y|}{j} \leq (N-|Y|)^j \leq N^j$$

and we get

$$S(N-|Y|,K) = \sum_{i=1}^{K-1} \binom{N-|Y|}{j} \leq \sum_{i=1}^{K-1} N^j = O(N^{K-1})$$

We will store the cache as a boolean array of length $S(N-|Y|,K)$, where the i-th bit determines if the i-th set is in the cache. The number of the set is the position of the set in the lexicographical ordering of all subsets of $V \setminus Y$ of size at most $K-1$. The depth of the recursion is at most N, hence the overall memory complexity is $O(N^K)$.

Let $f(i)$ be the worst case time complexity of $OptimalTime(Y)$ where $|Y| = i$. We will use the cache only if $\sum_{j=1}^{K-1} \binom{N-i}{j} \leq 2^i$ and according to lemma 2.3.1 the time required to calculate the position of a set G_X in the cache is $O(K^2) = O(1)$, therefore the performance of this algorithm will be in the worst case slower than the algorithm 2.2.2 only by a constant factor. Hence we can use equation (2.2) to bound the time complexity for all i:

$$f(i) \leq 2^i(f(i+1) + cM)$$

Let $T(N,K)$ be the first time when we use the cache, i.e. $T(N,K)$ is equal to the smallest h such that $\sum_{j=1}^{K-1} \binom{N-i}{j} \leq 2^h$. For all $i \geq T(N,K)$ we can bound $f(i)$ with another inequality. Since we use the cache and $f(j) \geq f(j+1)$, the worst case for $|Y| = i$ looks as follows. Every value in the cache will be used exactly once and the remaining procedure calls will be $OptimalTime(Y \cup G_X)$ where $|G_X| = K$. This implies

$$\begin{aligned} f(i) &= \sum_{j=1}^{K-1} \binom{N-i}{j} f(i+j) + \left(2^i - \sum_{j=1}^{K-1} \binom{N-i}{j}\right) f(i+K) + 2^i cM \\ &= \sum_{j=1}^{K-1} \binom{N-i}{j} (f(i+j) - f(i+K)) + 2^i f(i+K) + 2^i cM \end{aligned} \tag{2.4}$$

The first part of the equation corresponds to the cached values. We cache all G_X such that $|G_X| < K$, therefore the number of operations is at most $\sum_{j=1}^{K-1} \binom{N-i}{j} f(i+j)$. The remaining part of the 2^i procedure calls will in the

worst case be for G_X of size K, which is at most $(2^i - \sum_{j=1}^{K-1} \binom{N-i}{j}) f(i+K)$ operations. Finally, before each of the 2^i procedure calls we have to find the set G_X, which takes at most $O(M)$ operations.

Since in most cases $\sum_{j=1}^{K-1} \binom{N-i}{j} \ll 2^i$, we will try to simplify $f(i)$:

$$\hat{f}(i) = \begin{cases} 2^i \hat{f}(i+K), & i \leq N-1 \\ 1, & i \geq N \end{cases}$$

This simple recurrence leads to $\hat{f}(i) = 2^{\sum_{j=0}^{\lfloor (N-i)/K \rfloor} i+jK}$. The sum $\sum_{j=0}^{\lfloor (N-i)/K \rfloor} i+jK$ is a sum of an arithmetic progression. Let $a_1, a_2, \ldots, a_b$ be an arithmetic progression. Then the sum of this progression is $b \cdot (a_1 + a_b)/2$, i.e. the product of the length and the average of the first and the last number. The length of progression $i, i+K, \ldots, i+\lfloor (N-i)/K \rfloor K$ can be bounded by $\frac{N-i+K}{K}$ and the average of the first and last number can be bounded by $(N+i)/2$. Hence

$$\sum_{j=0}^{\lfloor (N-i)/K \rfloor} i + jK \leq (N+i)(N-i+K)/2K$$

whence we get

$$\hat{f}(i) \leq 2^{(N+i)(N-i+K)/2K}$$

We will use this result and write $f(i)$ as a product of two functions

$$f(i) = 2^{(N+i)(N-i+K)/2K} g(i)$$

and try to estimate $g(i)$. Since $f(i) = 0$ for $i > N$ and $f(N) = N$, we have $g(i) = 0$ for $i > N$ and $g(N) = N2^{-}N$. By (2.4) we have

$$\begin{aligned} &2^{(N+i)(N-i+K)/2K} g(i) = \\ &\sum_{j=1}^{K-1} \binom{N-i}{j} \left(g(i+j) 2^{(N+i+j)(N-i-j+K)/2K} - g(i+K) 2^{(N+i+K)(N-i)/2K} \right) \\ &\qquad + 2^{i+(N+i+K)(N-i)/2K} g(i+K) + 2^i cM \end{aligned} \quad (2.5)$$

for $1 \leq i \leq N-1$. Dividing both sides of the equation with $2^{(N+i)(N-i+K)/2K}$ yields

$$\begin{aligned} g(i) = \sum_{j=1}^{K-1} \binom{N-i}{j} &\left(\frac{g(i+j)}{2^{j(j+2i-K)/2K}} - \frac{g(i+K)}{2^i} \right) \\ &+ g(i+K) + cM 2^{-(N+i+K)(N-i)/2K} \end{aligned} \quad (2.6)$$

We will now prove that for $i \geq K^2 \log N$ the cache will be used. Since

$$2^i \geq 2^{K^2 \log N} = N^{K^2} > N^K \geq S(N, K)$$

the cache will be used. Hence for $i \geq K^2 \log N$ we can use equations (2.4) and (2.5).

We will prove by induction that $g(i) \leq 2^{2N-i}$ for $K^2 \log N \leq i \leq N$ and sufficiently large N. We have $g(N) = 2^{-N} \leq 2^N$, hence the basis of the induction holds. Now assume that this statement holds for every number that is at least $i + 1$. By (2.6), it follows that

$$\begin{aligned} g(i) \leq & \sum_{j=1}^{K-1} \binom{N-i}{j} \frac{2^{2N-i-j}}{2^{j(j+2i-K)/2K}} + 2^{2N-i-K} + cM2^{-(N+i+K)(N-i)/2K} \\ = & 2^{2N-i} \left(\sum_{j=1}^{K-1} \binom{N-i}{j} \frac{1}{2^{j(j+2i+K)/2K}} + \frac{1}{2^K} + \frac{cM}{2^{(N+i+K)(N-i)/2K+2N-i}} \right) \end{aligned} \tag{2.7}$$

Since $1 \leq j$ and $\binom{N-i}{j} \leq N^j$, we get

$$\begin{aligned} \sum_{j=1}^{K-1} \binom{N-i}{j} \frac{1}{2^{j(j+2i+K)/2K}} &\leq \frac{1}{2^{(2i+K)/2K}} \sum_{j=1}^{K-1} \frac{N^j}{2^{j/2K}} \\ &= \frac{1}{2^{(2i+K)/2K}} \sum_{j=1}^{K-1} \left(\frac{N}{2^{1/2K}} \right)^j \\ &\leq \frac{1}{2^{(2i+K)/2K}} \left(\frac{N}{2^{1/2K}} \right)^K \\ &= \frac{N^K}{2^{(2i+2K)/2K}} \end{aligned}$$

Since $i \geq K^2 \log_2 N$, it follows that

$$\frac{N^K}{2^{(2i+2K)/2K}} \leq \frac{N^K}{2^{K \log_2 N + 1}} = \frac{N^K}{2N^K} = \frac{1}{2}$$

which implies

$$\sum_{j=1}^{K-1} \binom{N-i}{j} \frac{1}{2^{j(j+2i+K)/2K}} \leq \frac{1}{2} \tag{2.8}$$

Finally, we will bound the last part of the equation (2.7). We have

$$\begin{aligned}\frac{cM}{2^{(N+i+K)(N-i)/2K+2N-i}} &\leq \frac{N^2}{2^{(N+i+K)(N-i)/2K+2N-i}} \\ &= \frac{2^{d+2\log_2 N}}{2^{(N^2+5KN-i^2-3iK)/2K}} \\ &= \frac{1}{2^{(N^2+5KN-i^2-3iK-4K\log_2 N-2dK)/2K}}\end{aligned} \tag{2.9}$$

Since

$$\begin{aligned}N^2 + 5KN - i^2 - 3iK - 4K\log_2 N - 2dK \\ = N(N+5K) - i(i+3K) - 4K\log_2 N - 2dK \\ = N(N+3K) - i(i+3K) + 2KN - 4K\log_2 N - 2dK\end{aligned}$$

and $i < N$, we have

$$N(N+3K) - i(i+3K) + 2KN - 4K\log_2 N - 2dK > 2KN - 4K\log_2 N - 2dK$$

K is a constant, hence

$$(2KN - 4K\log_2 N - 2dK)/2K \geq 2$$

for sufficiently large N. Thus

$$\frac{cM}{2^{(N+i+K)(N-i)/2K+2N-i}} \leq \frac{1}{2^{(N^2+5KN-i^2-3iK-4K\log_2 N-2dK)/2K}} \leq \frac{1}{2^2} \tag{2.10}$$

Finally, by (2.7), (2.8), (2.10) and $K \leq 2$, we get

$$g(i) \leq 2^{2N-i}\left(\frac{1}{2} + \frac{1}{2^K} + \frac{1}{4}\right) \leq 2^{2N-i}$$

Now we can also bound $f(i)$ for $K^2\log_2 N \leq i \leq N$ and sufficiently large N:

$$\begin{aligned}f(i) = g(i)2^{(N+i)(N-i+K)/2K} \\ \leq 2^{(N+i)(N-i+K)/2K+2N-i} = 2^{(N^2+5KN-i^2-iK)/2K}\end{aligned}$$

For $i < K^2\log_2 N$ we can use bounds from 2.2.1, because the performance of the algorithm using cache will not be worse than the performance of the algorithm that does not use the cache. Let $L = K^2\log_2 N$. By (2.2), we have

$$\begin{aligned}f(i) &= 2^i(2^{i+1}(\ldots(f(L) + cM) + \ldots) + cM) \\ &= cM\sum_{j=i}^{L} 2^{\sum_{k=i}^{j} k} + f(L)2^{\sum_{k=i}^{L} k}\end{aligned}$$

The complexity of the whole algorithm is bounded by $f(1)$, because we start the search with the set $\{s\}$:

$$\begin{aligned} f(1) =& cM \sum_{j=1}^{L} 2^{j(j-1)/2} + f(L)2^{(L)(L-1)/2} \\ =& (cM + f(L))2^{L(L-1)/2+1} \end{aligned}$$

Since $cM = O(f(L))$, we get

$$\begin{aligned} f(1) =& O(2^{(N^2+5KN-L^2-LK)/2K})2^{L(L-1)/2+1} \\ =& O(2^{(N^2+5KN+KL^2-2LK-L^2+2K)/2K}) \end{aligned}$$

Since K is a constant and $L = O(\log_2 N)$, we get

$$f(1) = 2^{(N^2+5KN)/2K+O(\log N)} = 2^{N^2/2K+O(N)}$$

It follows that the time complexity of the algorithm is $2^{N^2/2K+O(N)}$. □

Chapter 3

Approximation algorithms in GRN

In this chapter we consider broadcast schemes, that are not optimal. We present a global polynomial algorithm that produces broadcast scheme with broadcasting time $O(D(\frac{r_{max}}{r_{min}})^3)$ and also a distributed algorithm, that performs broadcast in the same time $O(D(\frac{r_{max}}{r_{min}})^3)$. In the second algorithm we assume that all nodes know positions, labels and ranges of all nodes within distance $r_{max} + r_{min}$. Since the optimal time is at least D, our algorithms have approximation ratio $O((\frac{r_{max}}{r_{min}})^3)$.

Lemma 3.0.2. *Let P be a partition of a plane into a mesh of unit squares. Let C be a circle with radius r arbitrarily placed in the plane. Then the number of unit squares in partition P that are fully inside C is at least $\pi r^2 - 8r - 6$.*

Proof. All unit squares in partition P can be divided into 3 sets G_1, G_2 and G_3. Denote by S the centre of the circle C. In the first set G_1 are the squares that do not have common point with disk C, i.e. every point in these squares is farther from S than r. The second set G_2 consists of squares that intersect with circle C and are not fully inside C, i.e. points of a square in G_2 are both inside and outside C. Finally, the third set G_3 consists of squares that are fully inside C.

Let v be the number of times C intersects with vertical edges and let h be the number of times C intersects with horizontal edges. Each edge belongs to exactly two squares, hence the total number of intersection points is $2(h+v)$. This number is the "total number of intersections" of C with the squares in G_2. It is easy to see that each corner intersection is counted twice, because two edges intersect in the corner. All other intersections are counted once.

At most 4 squares in G_2 can have exactly one intersection point with circle C (see figure 3.1). There are also squares with only one intersection point in the corner, but as we said earlier, we will count the corner intersection as two

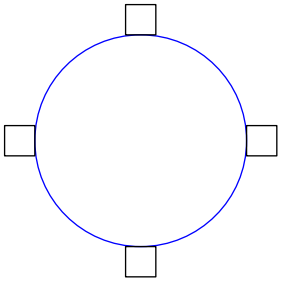

Figure 3.1: Four squares in the mesh that have just one common point with a circle.

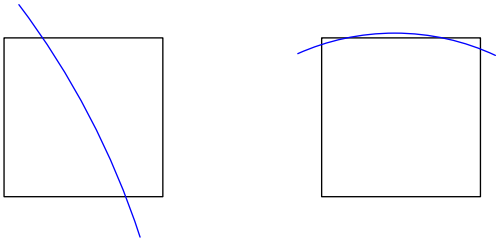

Figure 3.2: Two examples of an intersection of circle and square.

intersection points. All other squares in G_2 intersect with circle C at least two times (see figure 3.2). Hence the total number of intersection points of all squares in G_2 with the circle C is at least $2(|G_2| - 4) + 4$.

$$2(|G_2| - 4) + 4 \leq 2(h + v)$$
$$|G_2| - 2 \leq h + v \tag{3.1}$$

The horizontal and vertical edges of the squares in P form parallel lines one unit away from each other. The circle C intersects each line at most two times and there are $\lfloor 2r + 1 \rfloor$ horizontal and vertical lines within radius r of the circle C (see figure 3.3). Thus we get another pair of inequalities:

$$h \leq 2\lfloor 2r + 1 \rfloor \tag{3.2}$$

$$v \leq 2\lfloor 2r + 1 \rfloor \tag{3.3}$$

Finally, from inequalities 3.1, 3.2 and 3.3 we get:

$$|G_2| - 2 \leq 4\lfloor 2r + 1 \rfloor \leq 8r + 4$$
$$|G_2| \leq 8r + 6 \tag{3.4}$$

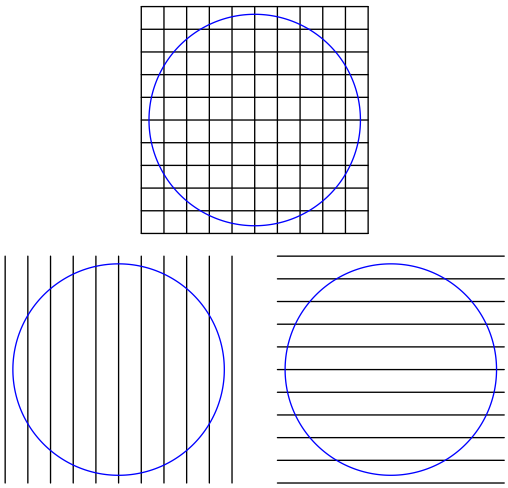

Figure 3.3: Circle intersects with both vertical and horizontal lines.

Let A be the set of points that are inside C but not inside one of the squares in G_2. The area of C is πr^2 and from (3.4) we get

$$|A| = \pi r^2 - |G_2| \geq \pi r^2 - 8r - 6$$

By the definition of A, all points of squares from G_3 form set A. Thus we have

$$|G_3| \geq \pi r^2 - 8r - 6$$

□

Lemma 3.0.3. *Let P be a partition of a plane into a mesh of unit squares. Let S be a square in the partition P and let C is the set of points that are at most r units away from S, i.e. the closest point from S is at most r units away. Then the number of squares in partition P that have at least one common point with C is at most $\pi r^2 + 12r + 9$.*

Proof. We will divide all squares in partition P into 3 groups G_1, G_2 and G_3. In the first group G_1 are the squares that do not have any common point with C, i.e. every point in these squares is farther from S than r. The second group G_2 consists of squares whose points are both inside and outside C. Finally, the third group G_3 consists of squares that are fully inside C.

The size of G_3 can be bounded by the area of C. Each square in G_3 has to be fully inside C, hence

$$|G_3| \leq |C| \tag{3.5}$$

The shape of C can be seen on figure 3.4. We see that C is a union of two rectangles $B_1B_2B_5B_6$, $B_3B_4B_7B_8$ and four quarter circles. These four quarter

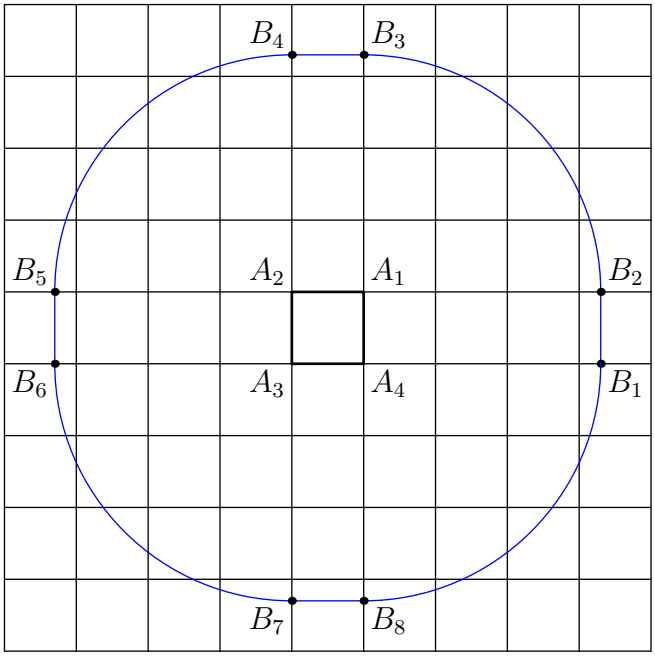

Figure 3.4: The set of points which are at most r units away from a unit square.

circles have centres in points A_1, A_2, A_3 and A_4 and their union forms a full disk. Union of rectangles $B_1B_2B_5B_6, B_3B_4B_7B_8$ has area $4r+1$, because the two rectangles have size $(2r+1) \times 1$ and their intersection is a unit square. Thus the area of C is $\pi r^2 + 4r + 1$ and by (3.5) we have

$$|G_3| \leq \pi r^2 + 4r + 1 \tag{3.6}$$

To bound the size of G_2 we will use arguments from lemma 3.0.2. Since the four quarter circles with centres in A_1, A_2, A_3, A_4 form together a circle, by inequality 3.4 the number of squares that intersects with these four quarter circles is at most $8r + 4$. There are four other squares that belong to the set G_2. These four squares are those that intersect with lines $B_1B_2, B_3B_4, B_5B_6, B_7B_8$. Hence

$$|G_2| \leq 8r + 4 + 4 = 8r + 8$$

and finally by (3.6), it follows that

$$|G_2| + |G_3| \leq \pi r^2 + 12r + 9 \tag{3.7}$$

□

3.1 Global polynomial approximation algorithm

In this section we will present a global polynomial algorithm that finds broadcast scheme with broadcast time $O(D(\frac{r_{max}}{r_{min}})^3)$ for a geometric radio network, where D is the eccentricity of the source, r_{max} is the maximal range and r_{min} is the minimal range of the nodes.

Theorem 3.1.1. *Let G is a geometric radio network with source eccentricity D. There exists a global polynomial algorithm which finds broadcast scheme with broadcast time $O(Dx^3)$, where $x = r_{max}/r_{min}$, r_{max} is the maximal range and r_{min} is the minimal range of all nodes in G.*

Proof. We will use the following construction: partition the plane into a mesh of squares of side $r_{min}/\sqrt{2}$, called *tiles*. There are many such partitions, use any of them. Each node now belongs to at least one tile. There are some nodes lying on the edges of the tiles and these nodes belong to two or more tiles. Nodes lying on vertical edges will belong to the right tile and nodes lying on horizontal edges will belong to the upper tile. Nodes lying on both types of edges, thus lying in the corner of a square, will belong to the upper right tile.

Observe that the points that are farthest away from each other in the tile are r_{min} units away from each other, hence every two nodes that are in the same tile can communicate with each other. We will number tiles in the partition such that tiles with the same number will be at least $2r_{max}$ units away from each other. We can number the tiles using $A(x)^2$ numbers where

$$A(x) = A\left(\frac{r_{max}}{r_{min}}\right) = \left\lceil \frac{2\sqrt{2}r_{max}}{r_{min}} + 1 \right\rceil = \lceil 2\sqrt{2}x + 1 \rceil$$

We will fill the $A(x)\times A(x)$ square with numbers from the set $\{0, 1, \ldots, A(x)^2 - 1\}$ as shown on figure 3.5 and repeat this pattern throughout the plane. The example on figure 3.5 uses numbers $0, 1, \ldots, 24$, hence in this case $\lceil 2\sqrt{2}x + 1 \rceil = 5$.

The broadcast scheme works as follows. Every node will be either in the state *informed* or *uninformed*. Initially, all nodes except the source node are uninformed, i.e. they did not hear the source message yet. The tile is *informed* if all nodes in the tile are informed. The broadcast scheme will consist of phases that will be divided into rounds. Phases are numbered from 0 to $A(x)^2 - 1$. In the i-th phase only nodes in tiles numbered i can transmit. Additionally, only tiles that contain at least one informed node can transmit. After phase number $A(x)^2 - 1$ ends, phase 0 starts again. This will happen $D + 1$ times, thus the total number of phases is $(D + 1)A(x)^2$.

0	5	10	15	20	0	5	10	15	20	0
4	9	14	19	24	4	9	14	19	24	4
3	8	13	18	23	3	8	13	18	23	3
2	7	12	17	22	2	7	12	17	22	2
1	6	11	16	21	1	6	11	16	21	1
0	5	10	15	20	0	5	10	15	20	0
4	9	14	19	24	4	9	14	19	24	4
3	8	13	18	23	3	8	13	18	23	3
2	7	12	17	22	2	7	12	17	22	2
1	6	11	16	21	1	6	11	16	21	1
0	5	10	15	20	0	5	10	15	20	0

Figure 3.5: Partition of the plane with numbering of the tiles

Let T is a tile that can transmit in the current phase, i.e. the number of this tile is the same as the number of the current phase. In the first round the node in T with the smallest label that is informed transmits the source message. Since the length of the side of the tile is $r_{min}/\sqrt{2}$, all nodes in the tile T will hear the message, thus they enter the informed state (unless they had not entered the informed state earlier).

In the second round the node $\hat{t}$ in the tile T with the lowest label amongst the nodes in T with the largest range transmits the source message. Let F be the set of tiles that were fully covered by this transmission, i.e. all nodes in the tiles in F heard the message.

Let H is the set of tiles, such that they are reachable from tile T, i.e. for each tile S in H there exists $u \in S$ and $v \in T$ such that u is reachable from v. In the remaining rounds we will reach all tiles in H that have not been covered by the transmission in round 2. These tiles are in the set $H \setminus F = \{h_1, \ldots, h_m\}$.

By the definition of H, for each tile h_i there exists a node $t_i \in T$ such that at least one node in h_i is reachable from t_i. In the next m rounds, nodes $t_1, t_2, \ldots, t_m$ will transmit the source message.

Let R be the largest range of a node in T. By lemma 3.0.3, the number of tiles reachable from T is at most $r^2 + 12r + 1$ where $r = \frac{R}{r_1/\sqrt{2}}$, hence

$$|H| \leq \pi r^2 + 12r + 11$$

By lemma 3.0.2, the number of tiles fully covered by broadcast of node $\hat{t}$ is

at least $r^2 - 8r - 4$, hence

$$|F| \geq \pi r^2 - 8r - 4$$

It follows that

$$|H \setminus F| \leq \pi r^2 + 12r + 11 - \pi r^2 + 8r + 4 = 20r + 15 =$$
$$= 20\left(\frac{R}{r_{min}/\sqrt{2}}\right) + 15 \leq 20\left(\frac{r_{max}\sqrt{2}}{r_{min}}\right) + 15 = 20\sqrt{2}x + 15 \qquad (3.8)$$

$|H \setminus F|$ is an integer, hence $|H \setminus F| \leq \lfloor 20\sqrt{2}x \rfloor + 15$. There are two rounds before these $|H \setminus F|$ rounds, hence the number of rounds needed to reach at least one one in all tiles reachable from T is at most $\lfloor 20\sqrt{2}x \rfloor + 17$. The overall broadcast time of the broadcast scheme is $(D+1)A(x)^2(\lfloor 20\sqrt{2}x \rfloor + 17)$, which is $O(Dx^3)$.

The algorithm to generate the broadcast scheme performs only operations working in polynomial time: select a node with the lowest label from a given set of nodes, partition the plane into tiles, etc. These polynomial operations are performed polynomial number of times – they are performed once, for tiles or pairs of tiles that contain at least one node. Thus the overall complexity is polynomial in the number of nodes.

To end the proof of the lemma we need to prove that the broadcast scheme is correct, i.e. all nodes will eventually get the source message. We start by proving that there will be no interference. Suppose the contrary. Let there be a node u such that at least two his in-neighbours transmit. Denote these two nodes by v, w (if there are more than two in-neighbours transmitting, take any pair). The distance from v and w to u is at most r_{max}. By triangle inequality, the distance from v to w is at most $2r_{max}$. If both v and w are transmitting, the numbers of their tiles have to be the same, because only tiles with the same number can transmit in the same phase. The distance of two tiles with the same number is more than $(A(x) - 1)r_{min}/\sqrt{2}$, which is equal to

$$(A(x) - 1)\frac{r_{min}}{\sqrt{2}} = (\lceil 2\sqrt{2}x + 1 \rceil - 1)\frac{r_{min}}{\sqrt{2}} =$$
$$= \left\lceil 2\sqrt{2}\frac{r_{max}}{r_{min}} \right\rceil \frac{r_{min}}{\sqrt{2}} \geq \left(\frac{2\sqrt{2}r_{max}}{r_{min}}\right)\frac{r_{min}}{\sqrt{2}} = 2r_{max} \qquad (3.9)$$

We have a contrary, because the distance of v and w is at most $2r_{max}$. Thus there will be no interference during the execution of the broadcast scheme.

Let L_i be the set of nodes that are at most i hops away from the source node. We will prove by induction that after $i \cdot A(x)^2 + j$ phases and 1 round,

all nodes in L_i with tile number j are informed. At the beginning, only the source node has the source message, thus for $i = 0$ the statement holds. Suppose that the statement holds for i. By the broadcast scheme, in the rounds $2, 3, \ldots, \lfloor 20\sqrt{2}x \rfloor + 17$ all tiles that are within reachability distance of a tile will contain at least one informed node. Hence at the end of phase $(i+1)A(X)^2$, all tiles within reachability distance of L_i will contain at least one informed node. In the first round of each of the next $A(x)^2$ phases, all these tiles will become informed, because they contain at least one informed node. Thus all nodes in L_{i+1} will be informed.

The broadcast ends after $(D+1)A(x)^2$ phases, hence all nodes in L_D will eventually get the source message. By the definition of D, L_D is the set of all nodes in the network, thus the broadcast scheme is correct. □

3.2 Large-knowledge algorithm

In this section we assume that all nodes know labels, positions and ranges of all nodes within range $r_{max}+r_{min}$. We will introduce a distributed algorithm, with broadcast time $O(Dx^3)$, where $x = \frac{r_{max}}{r_{min}}$. It will be a modification of the algorithm from previous section. We cannot repeat the construction from the previous algorithm where the node with the lowest label that received the source message in the previous $A(x)^2$ phases transmits the source message in the first round of the phase. This is not possible, because nodes in the tile do not know, which nodes heard the source message. In this modified algorithm, node with the lowest label which heard the source message does not wait for the phase with the right number to begin, but immediately informs all nodes in the tile. This modification increases the broadcast time by a factor of 8.

Theorem 3.2.1. *Let C be a geometric radio network, where all nodes know labels, positions and ranges of all nodes within range $r_{max} + r_{min}$. Then there exists a distributed deterministic broadcasting algorithm that performs broadcasting in $O(D(\frac{r_{max}}{r_{min}})^3)$*

Proof. Every node will execute the same algorithm. We will partition the plane into a mesh of squares of side $r_{min}/\sqrt{2}$, called *tiles* such that the edges of the squares are parallel to x-axis and y-axis and one square has a corner at position $(0, 0)$. Each node now belongs to at least one tile. There are some nodes lying on the edges of the tiles and these nodes belong to two or more tiles. Nodes lying on vertical edges will belong to the right tile and nodes lying on horizontal edges will belong to the upper tile. Nodes lying on both types of edges, thus lying in the corner of a square, will belong to the upper right tile.

We will number the tiles such that every two tiles are at least $4r_{max}$ unit away from each other. We can do this using $B(x)^2$ numbers where

$$B(x) = \lceil 4\sqrt{2}x + 1 \rceil = \left\lceil \frac{4\sqrt{2}r_{max}}{r_{min}} + 1 \right\rceil =$$

We will fill the $B(x) \times B(x)$ square with numbers from the set $\{0, 1, \ldots, B(x)^2 - 1\}$ as shown on figure 3.5 and repeat this pattern throughout the plane. This construction will ensure that all tiles with the same number will be at least $4r_{max}$ units away from each other.

The broadcast scheme will consists of phases. We start with phase number 0, continue with number 1 and so on. After we reach phase number $B(x)^2-1$, we start with 0 again. This will happen D times, hence the total number of phases is $DB(x)^2$.

Each phase will consist of $\lfloor 40\sqrt{2}x \rfloor + 31$ rounds. In phase number j, we will start the transmission from informed tiles numbered j. Let T be an informed tile with number j. The nodes will choose a node $\hat{t}$ with the lowest label in T with the largest range. This is possible, since every node in T has knowledge about all nodes in T. Node $\hat{t}$ transmits the source message. Let F be the set of tiles that are fully covered by this transmission. All tiles that are in F enter the informed state, unless they did not enter it before. Nodes that are in tiles that are not in F are not allowed to enter informed state.

Let H be the set of tiles, such that they are reachable from tile T, i.e. for each tile S in H there exists $u \in S$ and $v \in T$ such that u is reachable from v. In the remaining rounds we will reach all tiles in H that have not been covered by the transmission in the first round. These tiles are in the set $H \setminus F = \{h_1, \ldots, h_m\}$.

Let $t_i \in T$ and $v_i \in h_i$ be nodes such that v_i is reachable from t_i and the pair (t_i, v_i) is lexicographically smallest. By the definition of h_i, such pair will exist. Since nodes have enough information about all nodes within range $r_{max} + r_{min}$, every node in T will choose the same pair (t_i, v_i). In round $2i$ node t_i transmits the source message. In round $2i + 1$ node v_i transmits the source message and tile h_i enters informed state, unless it did not enter it before.

We will now bound the size of the set $H \setminus F$. Let R be the largest range of a node in T. By lemma 3.0.3, the number of tiles reachable from T is at most $r^2 + 12r + 1$ where $r = \frac{R}{r_1/\sqrt{2}}$, hence

$$|H| \leq \pi r^2 + 12r + 11$$

By lemma 3.0.2, the number of tiles fully covered by transmission in round

1 is at least $r^2 - 8r - 4$, hence

$$|F| \geq \pi r^2 - 8r - 4$$

It follows that

$$|H \setminus F| \leq \pi r^2 + 12r + 11 - \pi r^2 + 8r + 4 = 20r + 15 =$$
$$= 20\left(\frac{R}{r_{min}/\sqrt{2}}\right) + 15 \leq 20\left(\frac{r_{max}\sqrt{2}}{r_{min}}\right) + 15 = 20\sqrt{2}x + 15 \qquad (3.10)$$

We need to perform $2|H \setminus F| + 1$ rounds, hence the maximum number of rounds is $40\sqrt{2}x + 31$. As there are $DB(x)^2$ phases, the algorithm consists of $DB(x)^2(40\sqrt{2}x + 31)$ rounds, which is $O(Dx^3)$ rounds.

Now we will prove that the broadcast scheme is correct. We start by proving that there will be no interference. Let j be the number of a phase. In rounds $1, 2, 4, 6, \ldots$ only nodes in tiles with number j transmit. Since their distance is at least $4r_{max}$, there will be no interference in these rounds. In rounds $3, 5, 7, \ldots$ nodes that are at most r_{max} units away from tiles with number j will transmit. Hence the distance of any two transmitting nodes is at least $4r_{max} - 2r_{max} = 2r_{max}$ units. Thus there will be no interference.

Let L_i be the set of nodes that are at most i hops away from the source. By the beginning of the algorithm, only the source node is informed. In $40\sqrt{2}x + 31$ rounds of a phase number j all tiles reachable from all informed tiles with number j become informed. With this information it is easy to prove by induction that by the phase number $iB(x)^2$, all nodes in L_i are informed. Hence by the phase $DB(x)^2$, all nodes in the network will be in the informed state and the algorithm is correct. □

Chapter 4

Summary

Broadcasting in radio networks has been studied in many papers in the recent years. It was proved in [SH97] that establishing optimal broadcast scheme is NP-hard, even in geometric radio networks. This thesis is the first one to study various algorithms for optimal broadcast scheme. In the second chapter we have introduced three algorithms. The first one works in time $O(M3^N)$ and memory $O(N2^N)$. The second algorithm works in $2^{N^2/2+O(N)}$ time and $O(N^2)$ space. Finally, the third algorithm works in time $2^{N^2/2K+O(N)}$ and $O(N^K)$ space for a constant $K \geq 2$. This algorithm is an extension of the second algorithm parametrized by a constant K.

Since establishing optimal broadcast time is known to be NP-hard, mainly approximation algorithms have been studied. Obvious lower bound for broadcast time is the eccentricity of the source D. Additionally, in [ABNLP91] authors proved that there exists a family of N-node networks of radius 2, such that any deterministic broadcasting algorithm requires time $\Omega(\log^2 N)$. For symmetric networks, deterministic algorithm to produce broadcasting scheme with broadcasting time $O(D + \log^2 N)$ was published in [KDPA07].

Our work was based mainly on the broadcasting algorithm published in [DP07]. Authors claimed that broadcasting time is $O(D)$, but they treated the ration $\frac{r_{max}}{r_{min}}$ as a constant, where r_{max} is the maximal range and r_{min} is the minimal range of the nodes. The broadcasting time of the algorithm is actually $O(D(\frac{r_{max}}{r_{min}})^4)$.

We have presented a global algorithm, which finds broadcast scheme with broadcasting time $O(D(\frac{r_{max}}{r_{min}})^3)$. The second algorithm in the third chapter was a distributed algorithm, where nodes know the labels, positions and ranges of all nodes within range $r_{min} + r_{max}$. The broadcasting time of this algorithm is also $O(D(\frac{r_{max}}{r_{min}})^3)$. It remains an open question, if this can be reduced both in global and distributed setting further to $O(D(\frac{r_{max}}{r_{min}})^2)$ or even less.

Bibliography

[ABNLP91] Noga Alon, Amotz Bar-Noy, Nathan Linial, and David Peleg. A lower bound for radio broadcast. *J. Comput. Syst. Sci.*, 43(2):290–298, 1991.

[BDP97] Danilo Bruschi and Massimiliano Del Pinto. Lower bounds for the broadcast problem in mobile radio networks. *Distrib. Comput.*, 10(3):129–135, 1997.

[CGG+02] Bogdan S. Chlebus, Leszek Gasieniec, Alan Gibbons, Andrzej Pelc, and Wojciech Rytter. Deterministic broadcasting in ad hoc radio networks. *Distrib. Comput.*, 15(1):27–38, 2002.

[DP07] Anders Dessmark and Andrzej Pelc. Broadcasting in geometric radio networks. *J. of Discrete Algorithms*, 5(1):187–201, 2007.

[GKP94] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics: A Foundation for Computer Science*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1994.

[IKP08] David Ilcinkas, Dariusz R. Kowalski, and Andrzej Pelc. Fast radio broadcasting with advice. In *SIROCCO '08: Proceedings of the 15th international colloquium on Structural Information and Communication Complexity*, pages 291–305, Berlin, Heidelberg, 2008. Springer-Verlag.

[KDPA07] Kowalski, Dariusz, Pelc, and Andrzej. Optimal deterministic broadcasting in known topology radio networks. *Distributed Computing*, 19(3):185–195, January 2007.

[RS94] Krishnamurthi Ravishankar and Suresh Singh. Broadcasting on [0,l]. In *Proceedings of the international workshop on Broadcasting and gossiping 1990*, pages 299–319, New York, NY, USA, 1994. Elsevier North-Holland, Inc.

[SH97] Arunabha Sen and Mark L. Huson. A new model for scheduling packet radio networks. *Wirel. Netw.*, 3(1):71–82, 1997.

[SW06] Stefan Schmid and Roger Wattenhofer. Algorithmic Models for Sensor Networks. In *14th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS), Island of Rhodes, Greece*, April 2006.