Proceedings of the International Multiconference on Computer Science and Information Technology pp. 879–884
ISBN 978-83-60810-27-9
ISSN 1896-7094

Fast Construction of Broadcast Scheduling and Gossiping in Dynamic Ad Hoc Networks

Krzysztof Krzywdziński
Faculty of Mathematics and Computer Science
Adam Mickiewicz University, 60–769 Poznań, Poland
Email: kkrzywd@amu.edu.pl

Abstract—**This paper studies the minimum latency broadcast schedule (MLBS) problem in ad hoc networks represented by unit disk graphs. In our approach we use an algorithm, which does not need BFS tree. We introduce a construction, which does not depend on a source, can be found in constant number of synchronous rounds, uses only short messages and produces broadcast schedule with latency at most 258 times optimum. The advantage of our construction over known algorithms is its ability to adapt fast to the changes in the network, such as adding, moving or deleting vertices (even during the broadcasting).**

We also study the minimum-latency gossiping (all-to-all broadcast) problem in unit disk graphs. Our algorithm is the best result for gossiping in unit disk graph in unbounded-size messages model.

Since our construction of broadcast scheduling does not depend on the source, it may be also used to solve other problems concerning broadcasting in unit disk graphs, such as single source multiple message broadcasting and multi channel broadcast scheduling.

I. Introduction

DUE TO a wide range of applications, such as military surveillance, emergency disaster relief or environmental monitoring, solving problems considering the communication in wireless ad hoc networks (such as sensor networks or cell phone networks) became an important issue. Generally the wireless ad hoc network is modeled by an undirected connected graph G, where the set of vertices $V(G)$ represents the set of computational units constituting the network (for example sensors, cellphones) and $E(G)$ contains unordered pairs of distinct vertices, such that $(v,w) \in E(G)$ if and only if the transmissions of the computational unit represented by the vertex v can directly reach the computational unit represented by the vertex w and vice versa (the reachability of transmissions is assumed to be a symmetric relation). For simplicity in the following considerations frequently we will say vertex v instead of the computational unit represented by the vertex v. Also if $(v,w) \in E(G)$ we will say that the vertices v and w are neighbours in G.

Usually it is assumed that communication in the wireless ad hoc network is synchronous and consists of a sequence of communication steps. Moreover in a wireless ad hoc network, a message transmitted by a vertex is always sent to all of its neighbours. In each step, a vertex v either transmits or listens. If v transmits, then the transmitted message reaches each of its neighbours by the end of the step. However, a vertex w adjacent to v successfully receives this message if and only if in this step w is listening and v is the only transmitting vertex among the neighbours of w. If vertex w is a neigbour of more than one transmitting vertex, then due to the interference, w does not receive any message in this step.

The classical problem of the information dissemination in wireless networks is the broadcasting problem. In the broadcasting problem the aim is to disseminate a particular message from a distinguished source vertex to all other vertices in the network. Due to the interference threat the brute force algorithm may not be the effective tool to solve this problem. The classical approach is to construct the broadcast schedule in order to minimize the adverse influence of interferences. The broadcast schedule of latency l is the sequence of subsets $(U_1, U_2 \dots U_l)$ of $V(G)$ satisfying

(1) $U_1 = \{s\}$;
(2) $U_i \subseteq \bigcup_{j=1}^{i-1} Inf(U_j)$ for each $2 \le i \le l$;
(3) $V(G) \setminus \{s\} \subseteq \bigcup_{j=1}^{l} Inf(U_j)$,

where for any $U \subseteq V$, $Inf(U)$ is the set of vertices in $V(G) \setminus U$ each of which has exactly one neighbour in U. By the definition we have that, given the broadcast schedule $(U_1, U_2 \dots U_l)$, we may disseminate information from s into the whole network in l synchronous rounds. Namely, in order to do so, in the i-th round the information should be transmitted only by vertices from U_i.

In this paper we consider the problem of minimum latency broadcast schedule (MLBS), i.e., the problem of minimization of the time needed to complete the task of dissemination of the information from a single source. Therefore we seek a fast algorithm which constructs the broadcast schedule with the minimum latency.

For general graphs the best approximation algorithms construct the broadcast schedules with latency: $O(R\Delta)$ [14], $O(R\log^2(n/R))$ [13], $O(R\log n + \log^2 n)$ [4], $R + O(\sqrt{R}\log^2 n)$ [18], $O(R + \log^6 n)$ [15], $R + O(\log^3 n)$ [17], $R + O(\log^3 n/\log\log n)$[6] and $O(R + \log^2 n)$ [16].

Here by R we mean the radius of the graph (the maximum distance between the source s and the vertices from $V(G)$) and $n = |V(G)|$.

The MLBS problem in unit disk graphs has been considered in [8], [5],[7],[20] and [11].

In [8] Gąsieniec, Kowalski, Lingas and Wahlen show that $R + \Omega(\log(n - R))$ rounds are required to accomplish broadcasting in UDG. In [5] Dessmark and Pelc presented a broadcast schedule of latency at most $2400R$. In [7] Gandhi, Parthasarathy and Mishra claim the NP-hardness of MLBS in unit disk graphs and construct broadcast schedule of latency at most $648R$. In [20] Scott, Huang, Wan, Jia and Du improve latency to $51R$, $24R$, and $R+O(\sqrt{R}\log^{1.5} R)$. In [11] Huang, Wan, Jia, Du and Shang propose three different algorithms which produce broadcast schedules with latency at most $24R+23$, $16R+15$ and $R+O(\log R)$. However implementation of these algorithms in distributed model depend on the source and is based on BFS algorithm, therefore using these algorithms $\Omega(R)$ rounds to build the broadcast structure are needed. This is not efficient enough in many applications.

Here we present HEXAGONSBROADCASTING, which is the first distributed algorithm solving MLBS problem without using BFS tree. In fact HEXAGONSBROADCASTING needs only $O(1)$ synchronous rounds to prepare the vertices for the transmission. Our algorithm has latency $258R$. The structure constructed by HEXAGONSBROADCASTING does not depend on the source, therefore the additional advantage of our algorithm is that the source can be easily changed or even chosen randomly. Previous algorithms do not take into account possible changes in network during propagation of information. If a vertex is delete, they fail to deliver a message to a part of graph, despite the fact that graph might be still connected. In our algorithm bad impact of the movement, addition or deletion of the vertex in the network during the broadcasting may be rapidly diminished (Subsection III-A).

Finally the HEXAGONSBROADCASTING approach can be applied to solve real problems such as gossiping, multiple messages and multi channel broadcast schedule.

Our paper is organized as follows. In Section II we present basic definitions and facts. In Section III we introduce the auxiliary algorithms SELECTBROADCASTNODES and BROADCASTSETS and the main algorithm HEXAGONSBROADCASTING. In Section IV we present the algorithm GOSSIPINGINUDG and in Section V other two important generalizations of MLBS are presented.

II. PRELIMINARIES

The natural theoretical model for wireless ad hoc network is a unit disk graph (UDG). By definition, the set of vertices of UDG is a set of points on the plane and two vertices of UDG, v and w, are connected by an edge if and only if they are at the distance at most one in Euclidean norm ($\|v,w\| \leq 1$). UDG vertices represent computational units and edges represent communication links between them.

Our algorithms will work on unit disk graphs, in distributed, synchronous, message-passing model of computations. We assume that local clocks of computational units (vertices of UDG) can be synchronized i.e., we assume that we perform computations in rounds (model LOCAL defined in [19]). In each round a computational unit represented by a vertex of UDG can send, receive messages from its neighbours, and can perform some local computations. Moreover we assume that every computational unit is either equipped with the Global Positioning System (GPS), or knows own position on the plane by other sources.

In the algorithm HEXAGONSBROADCASTING we will use a partition of UDG based on grid tiling of the plane into hexagons of side equal $1/2$ (see Figure 1). Since each vertex of UDG knows its position on the plane, it also knows the hexagon of the grid to which it belongs. We define spanning subgraph G' of G where $V(G') = V(G)$ and $vw \in E(G')$ if and only if v and w lie in the same hexagon. We define $Hex(G)$ to be the set of all connected components of its spanning subgraph G' (notice that each hexagon lies inside a circle of radius $\frac{1}{2}$ so each $K \in Hex(G)$ is a clique).

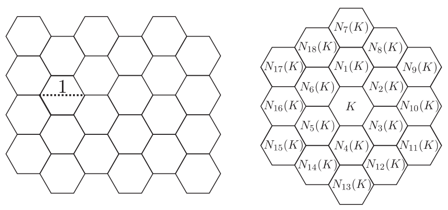

Fig. 1. Division into hexagons and numeration of the 18 neighbouring hexagons of the hexagon.

Let $K \in Hex(G)$ be any clique. Obviously K may have communication links only with vertices belonging to its 18 neighbouring hexagons (see Figure 1). We denote them by $N_1(K), N_2(K), \ldots, N_{18}(K)$ as on Figure 1 (the index of the clique depends on its position in the hexagon grid). Notice that, for example, $N_2(N_5(K)) = K$ and $N_{11}(N_{17}(K)) = K$. Therefore for any $k \in \{1,\ldots,18\}$ we define a value $\bar{k}$ to be the number such that $N_{\bar{k}}(N_k(K)) = K$ (for example $\bar{5} = 2$, $\bar{17} = 11$). In addition we set

$$N(K) = \\ \{K' \in Hex(G) : K' \neq K, \exists_{v \in K, v' \in K'} vv' \in E(G)\}.$$

For comparison, by $N(v)$ we denote the set of neighbours of the vertex v.

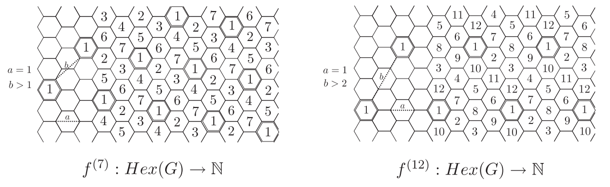
$a=1$
$b>1$
a
b
$f^{(7)} : Hex(G) \to \mathbb{N}$
$a=1$
$b>2$
a
b
$f^{(12)} : Hex(G) \to \mathbb{N}$

Fig. 2. Concept of $f^{(7)}$ and $f^{(12)}$ functions.

Finally we define two functions $f^{(7)} : Hex(G) \to \{1,\ldots,7\}$ and $f^{(12)} : Hex(G) \to \{1,\ldots,12\}$ (see Figure 2). Notice that in the labeling with the numbers $\{1,\ldots,7\}$ any two vertices from different hexagons with the same label are at distance greater than 1. Similarly in the labeling with the numbers $\{1,\ldots,12\}$ any two vertices from different hexagons with the same label are at distance geater than 2. This implies two simple facts.

Fact 1. *Any two vertices from different cliques $K_1, K_2 \in Hex(G)$ such that $f^{(7)}(K_1) = f^{(7)}(K_2)$ are not connected by an edge in G.*

Fact 2. *Any two vertices from different cliques $K_1, K_2 \in Hex(G)$ such that $f^{(12)}(K_1) = f^{(12)}(K_2)$ do not have a common neighbour in G.*

III. Main Algorithm

The main algorithm selects a small number of vertices, which we call broadcast vertices. In fact the algorithm assigns to each vertex two sets $F_{out}(\cdot)$ and $F_{in}(\cdot)$ and *broadcast* vertices are those for which $F_{out}(\cdot)$ and $F_{in}(\cdot)$ are nonempty. The first function F_{out} is used when $K \in Hex(G)$ sends information to cliques from the neighbouring hexagons. The second one F_{in} is used in dissemination of the message inside the clique. The auxiliary algorithm SelectBroadcastNodes selects broadcast vertices.

```
SelectBroadcastNodes
Input: Unit disk graph $G$
Output: Functions $F_{in} : V(G) \to \mathcal{P}(\{1,\dots,18\})$, $F_{out} : V(G) \to \mathcal{P}(\{1,\dots,12\})$, where $\mathcal{P}(A)$ denotes the set of all subsets of $A$.
  1) For every $v \in V(G)$ set $F_{in}(v) \equiv \emptyset$ and $F_{out}(v) \equiv \emptyset$.
  2) For $i = 1$ to 7 do : For $k = 1$ to 18 do
     For every $K \in Hex(G)$ such that $f^{(7)}(K) = i$ and $f^{(7)}(N_k(K)) > f^{(7)}(K)$ parallel do
       a) Let $V_k(K) = \{v \in V(N_k(K)) : N(v) \cap K \neq \emptyset\}$
       b) If $V_k(K) \neq \emptyset$ then
          i) Select a vertex $v_k \in V_k(K)$ and set
               $F_{in}(v_k) := F_{in}(v_k) \cup \{\bar{k}\};$
               $F_{out}(v_k) := F_{out}(v_k) \cup \{f^{(12)}(K)\}.$
          ii) Select a vertex $w \in N(v_k) \cap K$ and set
               $F_{in}(w) := F_{in}(w) \cup \{k\};$
               $F_{out}(w) := F_{out}(w) \cup \{f^{(12)}(N_k(K))\}.$
       c) We call $v_k$ and $w$ coupled broadcast vertices.
```

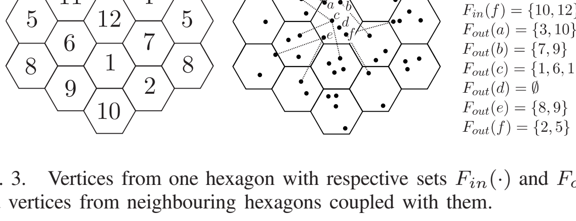

Fig. 3. Vertices from one hexagon with respective sets $F_{in}(\cdot)$ and $F_{out}(\cdot)$ and vertices from neighbouring hexagons coupled with them.

Notice that the algorithm SelectBroadcastNodes assigns exactly one pair of coupled broadcast vertices connected by an edge to each pair of the cliques $\{K, K'\}$ connected by at least one edge.

Basing on functions $F_{in} : V(G) \to \mathcal{P}(\{1,\dots,18\})$, $F_{out} : V(G) \to \mathcal{P}(\{1,\dots,12\})$ we construct sets $U_{out}(\cdot,\cdot)$ and $U_{in}(\cdot,\cdot)$ containing broadcast vertices, which will be used to construct the broadcast schedule.

The vertices in $U_{out}(i,k)$ send information from hexagons K with $f^{(12)} = i$ to hexagons K' with $f^{(12)}(K') = k$ and the vertices $U_{in}(i,1),\dots,U_{in}(i,18)$ are used in propagating information inside hexagons K with $f^{(7)} = i$.

```
BroadcastSets
Input: Unit disk graph $G$, Functions $F_{in} : V(G) \to \mathcal{P}(\{1,\dots,18\})$, $F_{out} : V(G) \to \mathcal{P}(\{1,\dots,12\})$
Output: Sets $U_{out}(i,k)$ and $U_{in}(i,k)$
  1) Let $K_v$ denote clique $K \in Hex(G)$ such that $v \in K$.
  2) For $i = 1$ to 12 : For $k = 1$ to 12 do
       $U_{out}(i,k) :=$
            $\Big\{v \in V : k \in F_{out}(v) \text{ and } f^{(12)}(K_v) = i\Big\}$
  3) For $i = 1$ to 7 : For $k = 1$ to 18 do
       $U_{in}(i,k) :=$
            $\Big\{v \in V : k \in F_{in}(v) \text{ and } f^{(7)}(K_v) = i\Big\}$
```

Note that for all $1 \le t \le 12$ the set $U_{out}(t,t)$ is empty. Hence the number of construceted nonempty sets is at most $11 \times 12 + 7 \times 18 = 258$ and in the main algorithm sets $U_{out}(t,t)$ may be omitted.

Lemma 1. *Let $1 \le i \le 12$, $1 \le k \le 12$ and $A \subseteq U_{out}(i,k)$. If $B \subseteq V(G)$ is the set of all vertices coupled with vertices from A and contained in cliques on which the value of $f^{(12)}$ equals k, then $B \subseteq Inf(A)$.*

Proof: Set $1 \le i \le 12$, $1 \le k \le 12$. Let $v_1 \dots v_t \in K$ be vertices from $U_{out}(i,k)$. For each vertex v_j the set $F_{out}(v_j)$ contains k if and only if v_j is coupled with at least one vertex from a neighbouring clique K' such that $f^{(12)}(K') = k$. Denote by W the set of vertices coupled with vertices from $\{v_1 \dots v_t\}$. Each vertex from W is contained in distinct clique (since to each pair of cliques at most one coupled pair of broadcast vertices is attributed). Combining Fact 2 with observation that on those distinct cliques the value of the function $f^{(12)}$ is k we have that no pair of vertices from the set W has a common neighbour in G, in particular no common neigbour in $\{v_1 \dots v_t\}$. Therefore, if $v_1 \dots v_t$ send information simultaneously into the network, then the vertices coupled with them (i.e vertices from W) receive it without interference. Now consider the vertices k_1, k_2 from $U_{out}(i,k)$ contained in two cliques K_1, K_2 (recall that, by the definition of U_{out}, $f^{(12)}(K_1) = f^{(12)}(K_2) = i$). By the Fact 2 vertices k_1 and k_2 do not have a common neighbour. Concluding, if vertices from the set $U_{out}(i,k)$ send information into the network, then the vertices coupled with them must get it without interference. ■

Lemma 2. *Let $1 \le i \le 7$, $1 \le k \le 18$. Suppose that vertices from the set $A \subseteq U_{in}(i,k)$ send information into the network* *simultaneously. Then for every $K \in Hex(G)$ such that $K \cap A \neq \emptyset$ we have that $K \subseteq Inf(A)$.*

Proof: Set $1 \leq i \leq 7$, $1 \leq k \leq 18$. Note that, for each broadcast vertex v, the set $F_{in}(v)$ contains k if and only if v is coupled with a vertex w from $N_k(K)$. So the set $V(K \cap U_{in}(i,k))$ has at most one element. Moreover any two distinct verices $v, v' \in U_{in}(i,k)$ are contained in cliques on which the function $f^{(7)}$ has value i. Therefore, from Fact 1 if any vertices from $U_{in}(i,k)$ sends simultaneously information into the network, then the vertices from cliques in which those vertices are contained, receive information without interference. ∎

Now we are ready to introduce the main algorithm of propagating the information from the source $s \in V(G)$.

```
HEXAGONSBROADCASTING
Input: Unit disk graph $G$ and a vertex $s \in V(G)$
Output: Dissemiantion of the information from the source vertex $s$ to $V(G)$.
1) Run SELECTBROADCASTNODES algorithm on a graph $G$.
2) Run BROADCASTSETS algorithm on $G$.
3) Vertex $s$ transmits information to all his neighbours.
4) For $t = 1$ to $R$ do
   a) For $i = 1$ to 12; For $k = 1$ to 12; $k \neq i$ do
      - vertices from $U_{out}(i,k)$ which have received information transmit it in parallel
   b) For $i = 1$ to 7 : For $k = 1$ to 18 do
      - vertices from $U_{in}(i,k)$ which have received information transmit it in parallel
```

Such approach to the problem enables us to construct a broadcast schedule in constant number of rounds, since the algorithm SELECTBROADCASTNODES obviously takes constant number of synchronous rounds and also building sets U_{out} and U_{in} takes constant time.

Theorem 3. *Let R be the radius of unit disk graph G. Latency of the broadcast schedule $U_1, U_2, U_3, \ldots$ used by the algorithm* HEXAGONSBROADCASTING *is at most $258R + 1$.*

Proof: Let $U_1, U_2, U_3, \ldots$ be broadcast schedule used by HEXAGONSBROADCASTING algorithm and $K \in Hex(G)$ be such that $s \in K$. In the first step the vertex s sends information to all the vertices from K (i.e. $K \subseteq Inf(U_1)$). Then, by Lemma 1, in the next $11 \times 12 = 132$ steps broadcast vertices contained in K send information to all vertices coupled with them in cliques from $N(K)$. By Lemma 2, in the next 7×18 steps broadcast vertices propagate information inside the cliques from $N(K)$. Therefore, surely, $\bigcup_{l=1}^{258+1} Inf(U_l)$ contains all vertices from K and all vertices from cliques $N(K)$. Therefore all neighbours of vertex s in G receive information in the first $258 + 1$ steps. Now, we may repeat the reasoning replacing the broadcast vertices from clique K by broadcast vertices contained in cliques from $N(K)$. Therefore in the next 258 steps all vertices at distance 2 from s will receive the message (i.e. all vertices at distance two from s are contained in $\bigcup_{l=1}^{2\times 258+1} Inf(U_l)$). So by the same reasoning we get that all vertices at distance at most R from s are contained in $\bigcup_{l=1}^{258R+1} Inf(U_l)$. ∎

A. Modification of the network during the broadcasting

The presented algorithm can be adjusted to construct the broadcast scheduling in the dynamic network in which appearance or deletion of vertices is possible. More precisely, we assume that at any time one of the following events may happen:

1) A vertex v, such that $G \cup \{v\}$ is connected, can be added.
2) A vertex $v \in V(G) \setminus \{s\}$, such that $G \setminus \{v\}$ is connected, can be deleted.
3) A vertex $v \in V(G)$ can change position on the plane (can be moved), if after this operation G is still connected.

Let s be a source and a be some constant. The following broadcasting algorithm broadcast information in the dynamic network.

```
DYNAMICBROADCASTING
1) Run HEXAGONSBROADCASTING and break step 4 of HEXAGONSBROADCASTING after $a$ iterations.
2) Go to step 1.
```

Since procedures SELECTBROADCASTNODES and BROADCASTSETS takes one round then using those algorithms it is possible to refresh rapidly the broadcast structure using the algorithm DYNAMICBROADCASTING. Therefore the algorithm DYNAMICBROADCASTING is an efficient tool to propagate information in dynamic network. If in some round the modifications (add, delete and move events) stop, then in the next $258R + (1/a)R + O(a)$ rounds all vertices receive information from the source. In reality the information will reach all the vertices even faster and it is possible, that the information will reach all vertices even if the modifications happen all the time.

In all known distributed algorithms (which are based on BFS tree) building the broadcast structure takes $O(R)$ time. Thus the analogous modification of those algorithm will not give comparable results. Namely, even if in some synchronous round the modifications (add, delete and move events) stop, then it would take at least $O(R^2/a + a)$ time to propagate the information from the source.

IV. GOSSIPING

In the gossiping problem each vertex in the network is initially given a message and the objective is to design a minimum-latency schedule such that each vertex distributes its message to all other vertices. We assume that during distribution of the messages interferences may occur.

There are two important models of the problem regarding whether or not we can combine two or more messages as a single message: unit-size and unbounded-size models. In unit-size trivial lower bound for latency is $V(G) + D - 1$ (D is a diameter of the graph G) and in unbounded-size model lower bound is $\Delta(G) + D - 1$ ($\Delta(G)$ is a maximum degree of a graph G).

For general graphs gossiping problem in unbounded-size model was investigated in: [1], [2], [3], [9], [10]. The known algorithms construct broadcast schedules which are approximations of the optimal one with approximation factor larger than any constant. For unit disk graphs in [7] the authors present constant approximation for gossiping. Although in the paper the exact ratio of approximation was not given it may be estimated to be at least $1944(D+|V(G)|)$. The best known algorithm for unit-size messages gossiping in unit disk graph is given in [12]. The authors present algorithm for gossiping with latency $27(|V(G)|+D-1)$. We should add that all known algorithms are based on the BFS tree therefore the time to build gossiping schedule is $\Omega(D)$.

We present here the algorithm for gossiping in unit disk graph in unbounded-size messages model which have latency $7\Delta + 258D$. The algorithm GOSSIPINGINUDG builds gossiping structure in constant time thus it can be adapted to become insensitive to adding, deleting or moving vertices (see Subsection III-A).

```
GOSSIPINGINUDG
1) Set function $F_{out}$ and $F_{in}$ using algorithm SELECT-BROADCASTNODES.
2) For each $K \in Hex(G)$ and vertices $v_1, v_2, v_3 \ldots = V(K)$ define $c(v_i) = 7i + f^{(7)}(K)$ for all $1 \leq i \leq |V(K)|$.
3) Let $v$ be a broadcast vertex. Define
       $c_2(v) = \max\{c(w) : \exists_u u \in N(v) \text{ and } w \in N(u)\}$
   (maximal value of the function $c$ in 2-neighbourhood of $v$).
4) Set all broadcast vertices (vertices which have $F_{out} \neq \emptyset$ and $F_{in} \neq \emptyset$ ) as inactive.
5) Parallel do:
   a) Each inactive vertex $v$ send his initial message in $c(v)$-th synchronous round.
   b) Every broadcast vertex $v'$ is activated in $c_2(v') + 1$ synchronous round and then sends all combined messages in rounds defined by point 4 of HEXAGONSBROADCASTING.
```

Theorem 4. *The algorithm* GOSSIPINGINUDG *delivers all messages to all vertices in* $7\Delta + 258D$ *synchronous rounds.*

Proof: Let $v \in V(K)$ and $v' \in V(K')$ be two distinct vertices such that $c(v) = c(v')$. From definition of the function c we have $f^{(7)}(K) = f^{(7)}(K')$ and $K \neq K'$. Therefore it follows from Fact 1 that in Step 5a no vertex from K will receive the initial message from v' while all vertices from K will receive the initial message from v. In step 5b broadcast vertices send messages after all vertices from their 2-neighbourhood (vertices at distance 2) have sent the initial message. Therefore there is no interference produced by broadcast vertices and the vertices sending the initial messages.

We use at most 7Δ synchronous rounds to send initial messages in Step 5a (there are 7 classes of hexagons and vertices from each hexagon form a clique) The number of rounds needed to complete 5b equals to the lattency of the schedule constructed by HEXAGONSBROADCASTING. Namely after the vertices from 2-neighbourhood have sent their initial message, all broadcast vertices send information according to HEXAGONSBROADCASTING. Properties of this broadcasting follow from Theorem 3. Therefore 5b takes at most $258D$ synchronous rounds.

Concluding the algorithm GOSSIPINGINUDG have latency at most $7\Delta + 258D$. ■

V. GENERALIZATIONS OF MLBS

A. Single Source Multiple Messages

Ghandi et al. in [7] introduce the problem of the single sources multiple messages broadcasting. They assume that the source may send multiple messages in different intervals of time and define latency as number of synchronous rounds in which all messages are received by all vertices. Let M be the number of the messages to be sent. In [7] latency is at least $194(M-1) + 7128(R-2)$.

We present algorithm which is a modification of the HEXAGONSBROADCASTING algorithm and construct Single Source Multiple Messages Broadcasting schedule of latency $518M+258R$. We only have to use the following modification of the points 3 and 4 of the algorithm HEXAGONSBROADCASTING.

```
3. Repeat $M + \lceil R/2 \rceil$ times
 (a) If $s$ contains a new message, then send it
 (b) Repeat two times
     – For $i = 1$ to 12; For $k = 1$ to 12; $k \neq i$ do
         - vertices from $U_{out}(i,k)$, which have received information and have not sent it before, send the information
     – For $i = 1$ to 7 : For $k = 1$ to 18 do
         - vertices from $U_{in}(i,k)$, which have received information and have not sent it before, send the information
```

Theorem 5. *If in the modified algorithm* HEXAGONSBROADCASTING *the source sends messages every* 259 *synchronous rounds then every vertex in network receive all the messages in at most* $518M + 258R$ *rounds, where* M *is the number of messages.*

Proof: Divide the broadcast schedule used by HEXAGONSBROADCASTING (omitting the first round) into intervals of 258 rounds. Notice that, if the broadcast vertex receive information in the t-th interval, then in the $t+1$-th interval the information is sent to all its neighbours. Therefore this vertex may send a new information in the $t+2$-th interval. The lack of bad impact of the interference follows from the proof of Theorem 3. Thus the gap of two intervals plus one round (to give time for s to send the message) is enough to broadcast several messages. ■

B. Multi Channel Broadcast Scheduling

In some applications the vertices can use n different channels and if two neighbours of the vertex v send information using different channels then vertex v receives it without interference. This is idea of multi channel broadcast scheduling. To the best of our knowledge this problem is considered for the first time.

Notice that in the previous considerations, if for any pair of indices $(i_1, k_1) \neq (i_2, k_2)$, vertices from sets $U_{out}(i_1, k_1)$ and $U_{out}(i_2, k_2)$ use different channels, then they may send information simultaneously in all the above mentioned algorithms without interference. The same thing concerns the sets $U_{in}(i_1, k_1)$ and $U_{in}(i_2, k_2)$ ($(i_1, k_1) \neq (i_2, k_2)$). Therefore, if we use 132 channels ($11 \times 12 \leq 132$, $7 \times 18 \leq 132$) and attribute to each set of type U_{out} different channel and the same thing with the sets U_{in}, then we may rewrite all the presented algorithms and reduce significantly the time needed to disseminate information. If we define the latency in the analogous manner as in the classical definition, then for example the multi channel broadcast schedule used in the algorithm HEXAGONSBROADCASTING has latency $2R$.

VI. CONCLUSION

In this paper we have studied the MLBS problem in ad hoc networks which are represented by unit disk graphs. We have introduced a construction which produces broadcast schedule with latency at most 258 times optimum. The advantage of our construction over known algorithms is its ability to adapt fast to the changes in the network. We have also studied the minimum-latency gossiping, single source multiple message broadcasting and multi channel broadcast scheduling problem in unit disk graphs.

REFERENCES

[1] M. Chrobak, L. Gasieniec, and W. Rytter. Fast broadcasting and gossiping in radio networks. In *FOCS '00: Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, page 575, Washington, DC, USA, 2000. IEEE Computer Society.

[2] Marek Chrobak, Leszek Gasieniec, and Wojciech Rytter. A randomized algorithm for gossiping in radio networks. In *COCOON '01: Proceedings of the 7th Annual International Conference on Computing and Combinatorics*, pages 483–492, London, UK, 2001. Springer-Verlag.

[3] Marek Chrobak, Leszek Gasieniec, and Wojciech Rytter. Fast broadcasting and gossiping in radio networks. *J. Algorithms*, 43(2):177–189, 2002.

[4] A. D. Kowalski, A. Pelc. Centralized deterministic broadcasting in undirected multi-hop radio network. *In Random-Approx*, page 171182, 2004.

[5] Anders Dessmark and Andrzej Pelc. Tradeoffs between knowledge and time of communication in geometric radio networks. In *SPAA '01: Proceedings of the thirteenth annual ACM symposium on Parallel algorithms and architectures*, pages 59–66, New York, NY, USA, 2001. ACM.

[6] F. Manne F. Cicalese and Q. Xin. Faster centralized communication in radio networks. *LNCS*, 4288:339–348, 2006.

[7] Rajiv Gandhi, Srinivasan Parthasarathy, and Arunesh Mishra. Minimizing broadcast latency and redundancy in ad hoc networks. In *MobiHoc '03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 222–232, New York, NY, USA, 2003. ACM.

[8] Leszek Gąsieniec, Dariusz R. Kowalski, Andrzej Lingas, and Martin Wahlen. Efficient broadcasting in known geometric radio networks with non-uniform ranges. In *DISC '08: Proceedings of the 22nd international symposium on Distributed Computing*, pages 274–288, Berlin, Heidelberg, 2008. Springer-Verlag.

[9] Leszek Gąsieniec and Andrzej Lingas. On adaptive deterministic gossiping in ad hoc radio networks. *Inf. Process. Lett.*, 83(2):89–93, 2002.

[10] Leszek Gąsieniec and Igor Potapov. Gossiping with unit messages in known radio networks. In *TCS '02: Proceedings of the IFIP 17th World Computer Congress - TC1 Stream / 2nd IFIP International Conference on Theoretical Computer Science*, pages 193–205, Deventer, The Netherlands, The Netherlands, 2002. Kluwer, B.V.

[11] Huang, P. J. Wan, X. Jia, H. Du, and W. Shang. Minimum-latency broadcast scheduling in wireless ad hoc networks. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 733–739, 2007.

[12] Scott C.-H. Huang, Hongwei Du, and E-K. Park. Minimum-latency gossiping in multi-hop wireless networks. In *MobiHoc '08: Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, pages 323–330, New York, NY, USA, 2008. ACM.

[13] O. Weinstein I. Chlamtac. The wave expansion approach to broadcasting in multihop radio networks. *IEEE Trans. on Communications*, 39:26–433, 1991.

[14] S. Kutten I. Chlamtac. On broadcasting in radio networks - problem analysis and protocol design. *IEEE Transactions on Communications*, 33:12401246, 1985.

[15] Y. Mansour I. Gaber. Centralized broadcast in multihop radio networks. *Journal of Algorithms*, 46:120, 2003.

[16] D. Kowalski and A. Pelc. Optimal deterministic broadcasting in known topology radio networks. *Distributed Computing*, 19(3:185–195, 2007.

[17] D.Peleg L.Gasieniec and Q.Xin. Faster communication in known topology radio networks. In *Proceedings of The 24th Annual ACM Symposium on Principles of Distributed Computing*, 2005.

[18] G. Kortsarz M. Elkin. Improved broadcast schedule for radio networks. In *Symposium on Discrete Algorithms (SODA)*, 2005, 222-231.

[19] David Peleg. *Distributed computing: a locality-sensitive approach*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

[20] Xiaohua Jia Scott C. H. Huang, Peng-Jun Wan and Hongwei Du. Low-latency broadcast scheduling in ad hoc networks. *Lecture Notes in Computer Science*, 4138:527–538, 2006.