Minimizing Interference for the Highway Model in Wireless Ad-Hoc and Sensor Networks*

Haisheng Tan[1], Tiancheng Lou[2], Francis C.M. Lau[1], Yuexuan Wang[2], and Shiteng Chen[2]

[1] Department of Computer Science, The University of Hong Kong, Pokfulam, Hong Kong, China
[2] Institute for Theoretical Computer Science, Tsinghua University, Beijing, 100084, China

Abstract. Finding a low-interference connected topology is one of the fundamental problems in wireless ad-hoc and sensor networks. The receiver-centric interference on a node is the number of other nodes whose transmission ranges cover the node. The problem of reducing interference through adjusting the nodes' transmission ranges in a connected network can be formulated as that of connecting the nodes by a spanning tree while minimizing interference. In this paper, we study minimization of the average interference and the maximum interference for the high-way model, where all the nodes are arbitrarily distributed on a line. Two exact algorithms are proposed. One constructs the optimal topology that minimizes the average interference among all the nodes in polynomial time, $O(n^3\Delta^3)$, where n is the number of nodes and Δ is the maximum node degree. The other algorithm constructs the optimal topology that minimizes the maximum interference in sub-exponential time, $O(n^3\Delta^{O(k)})$, where $k = O(\sqrt{\Delta})$ is the minimum maximum interference.

Keywords: wireless ad-hoc and sensor networks, interference minimization, topology control, combinatorial optimization, dynamic programming.

1 Introduction

Wireless ad-hoc and sensor networks consist of a set of nodes deployed across a region of interest. Each node has limited processing ability and is equipped with a wireless radio for communication. Compared with traditional wired networks, they do not have a fixed infrastructure. The nodes can adjust their transmission powers to achieve their desired transmission ranges which then form a multi-hop network. Wireless ad-hoc and sensor networks have many applications in real life such as environmental monitoring, intrusion detection, and health care. It is regarded as one of the most popular networking paradigms.

Due to the environments in which they are typically deployed, wireless nodes can only use relatively weak batteries. Energy is therefore at a premium which

* The work presented in this paper was supported in part by Hong Kong RGC-GRF grants (714009E), the National Basic Research Program of China Grant Nos. 2007CB807900, 2007CB807901 and the National Natural Science Foundation of China Grant No. 61073174.

I. Černá et al. (Eds.): SOFSEM 2011, LNCS 6543, pp. 520–532, 2011.

however is critical for the network's lifetime. One direction for energy conservation is to reduce interference which occurs when communication between two nodes is interfered by another concurrent transmission nearby. Different models have been defined to depict the phenomenon [3,4,5,6,7]. Paper [14] proposed a stable and realistic interference model, called *the receiver-centric model*, where the interference on a node v is the number of other nodes whose transmission ranges cover v (Figure 1). In this paper, unless specified, the receiver-centric model is assumed.

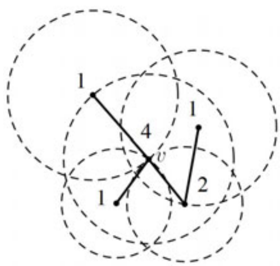

Fig. 1. The receiver-centric interference: the numbers are interference on each node

Topology control refers to selecting only a subset of the available communication links for data transmission, which has been widely used to construct networks with specific properties such as planarity, bounded node degree, the spanner property and low interference [13,1,10,8]. Researchers are not only interested in minimizing the average interference on the nodes, but also the maximum interference, because the maximum interference is closely related to the time when the first node runs out of energy, which could mean a halt of the entire network's operation. The problem of minimizing the maximum interference while preserving connectivity in two-dimensional networks has been proved to be NP-complete [2]. Authors of [9] proposed an algorithm that could bound the maximum interference by $O(\sqrt{\Delta})$ using the $\varepsilon - net$ theory in computational geometry. Here, n is the number of nodes and Δ is the maximum node degree in the topology when each node is set to the maximum transmission range and connected to all the other nodes in its range (If all the nodes have the same maximum transmission range, the topology is actually a unit-disk-graph). For minimizing average interference in 2D networks, paper [12] developed an asymptotically optimal algorithm with an approximation ratio of $O(logn)$. Researchers are also interested in the interference problem in 1D networks as there are also many application scenarios for 1D networks, such as bridges and tunnels. For minimizing the maximum interference on the exponential chain, authors in [14,15] proposed an asymptotically optimal algorithm and proved a tight lower bound of $\Omega(\sqrt{\Delta})$. Here *the exponential chain* means the nodes are distributed on a 1D line with the distances growing exponentially. Furthermore, for the general case, in which the nodes are arbitrarily distributed on a line, the so called *highway model*, they bounded the minimum maximum interference by $O(\sqrt{\Delta})$ and presented an approximation with ratio of $O(\sqrt[4]{\Delta})$.

In this paper, we study minimization of the average and the maximum interference for the highway model. Two exact algorithms are proposed. One is to construct the optimal connected topology with minimum average interference in $O(n^3 \Delta^3)$ time. The other constructs the connected topology with minimum maximum interference. Here, minimizing the maximum interference is related to the second open problem proposed in [11], but we restrict the model to one dimension and add a constraint on the maximum transmission range of the nodes. Our algorithm runs in sub-exponential time, $O(n^3 \Delta^{O(k)})$, where $k = O(\sqrt{\Delta})$ is the minimum maximum interference. We can see when Δ is small, which means a low maximum node degree, our algorithm is fast. To our knowledge, the former algorithm is the first polynomial-time algorithm for minimizing the average interference and the latter is the first sub-exponential-time algorithm for minimizing the maximum for the highway model.

The rest of the paper is organized as follows. In Section 2, we give the formal definitions of the interference model and the problem. Section 3 describes the no-cross property and gives the algorithm to minimize the average interference for the highway model. Section 4 describes how to minimize the maximum interference. Section 5 concludes the paper and points out some open problems and possible future work.

2 Models and Problem Definitions

We assume a wireless ad-hoc and sensor network in which the nodes are stationary after deployment in a region. If at some point they need to be moved, we can re-run the proposed algorithms using the new coordinates. The maximum transmission radius of the nodes is denoted as r_{max}. Each node can self-adjusts its transmission radius from 0 to r_{max} in a continuous manner. There are no obstacles to block the communications. Therefore, the maximum transmission range of a node v will be the disk centered at v with radius r_{max}. For the highway model, we assume r_{max} is not shorter than the farthest distance between two consecutive nodes, or else it is not possible to construct connected topology.

The network is modeled as an undirected graph $G = (V, E)$, where V is the set of nodes and E is the set of communication links. For the highway model, the n nodes in $V = \{v_0, v_1, ..., v_{n-1}\}$ are arbitrarily deployed along a line from left to right. We can view the line as an x-axis, and $v_0 = 0$. Then, each node u is denoted as its x-coordinate. An edge $(u, v) \in E$ exists only if both their transmission radii, r_u and r_v, are not shorter than their Euclidean distance $|u - v|$. Therefore, in G, the transmission radius of a node is equal to the distance to its farthest neighbor (Two nodes are neighbors means there is an edge incident on them.). In addition, we introduce the following terms. For a segment $\overline{v_s v_t}$ on the line, where $s \leq t$, the nodes *located on* $\overline{v_s v_t}$ are $\{v_s, v_{s+1}, ..., v_{t-1}, v_t\}$; the nodes outside $\overline{v_s v_t}$ are the other nodes that are not on it; the nodes *inside* $\overline{v_s v_t}$ are $\{v_{s+1}, v_{s+2}, ..., v_{t-1}\}$.

The receiver-centric interference model is adopted. The interference of a node v, denoted as $RI(v)$, is defined as the number of other nodes whose transmission ranges can cover v:

$$RI(v) = |\{u|u \in V/\{v\}, |u - v| \leq r_u\}|. \tag{1}$$

The average node interference in G, $RI_{avg}(G)$, can be defined as:

$$RI_{avg}(G) = \frac{\sum_{v \in V} RI(v)}{|V|}. \tag{2}$$

The maximum node interference, $RI_{max}(G)$, can be defined as:

$$RI_{max}(G) = max_{v \in V}\ RI(v). \tag{3}$$

Besides minimizing interference, we also need to preserve the network connectivity. Therefore, the optimal topology with the minimum interference should be a spanning tree on V. Therefore, our problems can be defined as:

Given n nodes arbitrarily distributed on a 1D line, construct a spanning tree, $G = (V, E)$, to connect all the nodes with edges no longer than r_{max}. The minimization of the average interference problem is to construct a spanning tree that minimizes $RI_{avg}(G)$, and the minimization of the maximum interference problem is to construct a spanning tree that minimizes $RI_{max}(G)$.

3 Minimizing the Average Interference

3.1 No-Cross Property

For a spanning tree $G = (V, E)$ constructed on the nodes along a line, we can draw all the edges on one side of the line. A *cross* means there are two edges that share at least a common point excluding their endpoints (Figure 2(a)). By

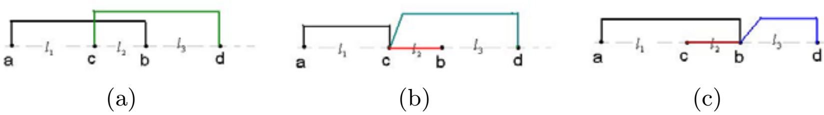

Fig. 2. a, b, c and d are four nodes distributed on a line, where $l_1 = c - a$, $l_2 = b - c$, and $l_3 = d - b$, and (a, b) and (c, d) are two edges: (a) (a,b) and (c, d) have a cross; (b) the cross removed when $l_1 \leq l_2 + l_3$; (c) the cross removed when $l_1 > l_2 + l_3$

adding and deleting edges, we show below that a cross can be removed without increasing interference on any nodes while preserving the network connectivity.

Theorem 1. *For a spanning tree connecting the nodes on a line with crosses, there is always another spanning tree to remove the crosses without increasing interference on any node.*

Proof. We prove this theorem by illustrating how to remove a cross. Without loss of generality, we handle the cross in Figure 2(a). Note that there can be other nodes distributed at any other places on the line and the four nodes need not be consecutive. For the case $l_1 \leq l_2 + l_3$, we remove the cross by replacing the edge (a, b) with (a, c) and adding (c, b) (Figure 2(b)). Firstly, we check whether the newly added edges, (a, c) and (c, b), are valid which means their lengths do not exceed r_{max}. Since $|a - c| = l_1 < l_1 + l_2 = |a - b|$ and (a, b) is valid, (a, c) is also valid. Similarly, (c, b) is also valid. Secondly, there are 3 nodes, a, b and c, whose edges are changed. We check whether the changes potentially make them interfere with any new nodes. For a, one of its longer edges (a, b) is replaced with a shorter one (a, c), so a cannot interfere with more nodes in the new topology. A similar conclusion can be arrived at for b. As for the node c, we add a new edge (a, c) of length l_1 and (b, c) of length l_2. However, in both topologies, c already has an edge (c, d) of length $l_2 + l_3$. Since $l_2 + l_3 > l_2$ and $l_2 + l_3 \geq l_1$, the new edges will not make c interfere with any new nodes. Therefore, the topology in Figure 2(b) would not add to the interference on any nodes. Thirdly, since there are still paths to connect the nodes a, b and the nodes c, d, the new topology is connected as long as the topology in Figure 2(a) is connected. Further, since deleting an edge will not increase any interference, we can destroy any cycles in the new topology by deleting edges to form a spanning tree. Therefore, for the case $l_1 \leq l_2 + l_3$, we can remove the cross to construct a new spanning tree without adding to the interference on any nodes. Similarly, we can prove that the above is also true when $l_1 > l_2 + l_3$ as illustrated in Figure 2(c), and the theorem is proved. □

According to the no-cross property, if there is already an edge (v_s, v_t), all the nodes inside the segment $\overline{v_s v_t}$ can be only adjacent to nodes located on the segment, but not to any other nodes on the line. (Two nodes are adjacent means they are neighbors.) However, it does not mean that interference of the nodes inside the segment is independent of the nodes outside. The nodes inside $\overline{v_s v_t}$ can interfere with the ones outside, and vice versa. This gives an important clue for us to design algorithms to minimize the average or the maximum interference which are described in the following sections.

3.2 Algorithms to Minimize the Average Interference

General Ideas. Based on the no-cross property, in the optimal spanning tree with minimum average interference, the nodes can be separated into segments. The nodes inside each segment are only adjacent to the other nodes on the same segment. However, as mentioned above, interference of the nodes inside a segment is still independent of the outside. Therefore, we do not compute the total interference by summing up the interference on each individual node, but the interference created by each node. Here, interference created by a node v with transmission radius r_v, $CI(v, r_v)$, is defined as the number of other nodes covered by the transmission range of v:

$$CI(v, r_v) = |\{u | u \in V/\{v\}, |u - v| \leq r_v\}|, \tag{4}$$

so that $\sum_{v \in V} CI(v, r_v) = \sum_{v \in V} RI(v)$. $CI(v, r_v)$ is only influenced by r_v, which is determined by the neighbors of v, and the locations of the other nodes. If all the nodes inside $\overline{v_s v_t}$ can only be adjacent to the nodes on it, the total interference created by the inside nodes will be independent of the topology of the outside nodes; and vice versa. Moreover, to compute the optimal spanning tree, we need to determine 1) how to divide the line into segments and 2) how to connect the nodes on each segment. Therefore, we can construct the optimal spanning tree based on dynamic programming as follows.

Algorithms. Two auxiliary functions are defined. The function $F(s,t)$[1], where $s < t$, is to compute the topology on $\overline{v_s v_t}$ so that the total interference created by the nodes inside $\overline{v_s v_t}$ is minimized with the following conditions satisfied:

1) the transmission radius of v_s is r_{v_s}.
2) the transmission radius of v_t is r_{v_t}.
3) all the nodes inside $\overline{v_s v_t}$ can be only adjacent to the ones on the segment $\overline{v_s v_t}$.
4) each node inside $\overline{v_s v_t}$ has a path either to v_s or to v_t.

The function $G(s,t)$, where $s < t$, is to compute the topology on $\overline{v_s v_t}$ so that the total interference created by the nodes inside $\overline{v_s v_t}$ is minimized with the following conditions satisfied:
$1), 2), 3)$ are the same as the first three conditions of $F(s,t)$.
4) all the nodes on $\overline{v_s v_t}$ are connected to each other directly or by nodes on $\overline{v_s v_t}$.

Both the functions F and G return the minimum total interference created by the nodes inside $\overline{v_s v_t}$. If $+\infty$ is returned, it means there is no such a topology to satisfy all the conditions. Comparing the fourth conditions, for function F, to achieve connectivity among all the nodes, we actually assume there is already a path from v_s to v_t before adding any edges to the nodes inside $\overline{v_s v_t}$. For G, there is no such a path.

For a node v, the set of its potential neighbors, $N(v)$, are the nodes covered by v's maximum transmission range:

$$N(v) = \{u | u \in V/\{v\}, |u - v| \leq r_{max}\}. \tag{5}$$

Recall that the transmission radius of v is the distance to its farthest neighbors. So, the set of its potential transmission radii, $R(v)$, is

$$R(v) = \{|u - v| | u \in N(v)\}, \tag{6}$$

and $|R(v)| \leq |N(v)| \leq \Delta$. If v can only be adjacent to a subset nodes S, its potential neighbors $N(v,S)$ and its potential transmission radii $R(v,S)$ are $N(v,S) = N(v) \bigcap S$ and $R(v,S) = \{|u - v| | u \in N(v,S)\}$ respectively. To compute the functions F and G, we calculate and store each $CI(v, r_v)$ in an $n \times \Delta$ array. For $F(s,t)$, the boundary condition is there are no nodes inside $\overline{v_s v_t}$. For the other cases, to satisfy the condition 4), there must be at least one node v_m inside $\overline{v_s v_t}$ that is adjacent to v_p, where $v_p = v_s$ or v_t. Without loss of generality, we set $v_p = v_s$. Since v_s and v_m are connected as well as v_s and v_t in the assumption, there is already a path from v_m to v_t. Therefore, $F(s,t)$ consist of three parts, $F(s,m)$, $F(m,t)$ and $CI(m, r_{v_m})$. We can enumerate v_m and r_{v_m}, so F can be computed in Algorithm 1. In line 1, we first check the boundary

[1] For conciseness, we use $F(s,t)$ to stand for $F(v_s, v_t, s, t, r_{v_s}, r_{v_t})$, and $G(s,t)$ to stand for $G(v_s, v_t, s, t, r_{v_s}, r_{v_t})$.

condition. The set S is defined to store the nodes on $\overline{v_s v_t}$ in line 2. Lines 3–10 are to compute the minimum interference created by the nodes inside $\overline{v_s v_t}$ recursively with the four conditions satisfied. As v_m can only be adjacent to the nodes on $\overline{v_s v_t}$, its potential transmission radii are defined as $R(v_m, S)$ in line 4. In line 7 we assume adding an edge (v_m, v_p), and line 8 is to compute $F(s,t)$.

Algorithm 1. Compute $F(s,t)$

1. **if** $s+1=t$ **then** **return** $F=0$ /* the boundary condition*/
2. $F=+\infty$ $S=\{v_s, v_{s+1}, ..., v_t\}$
3. **for each** $v_m \in S/\{v_s, v_t\}$ **do**
4. $\quad R(v_m, S)=\{|u-v_m||u \in N(v_m) \bigcap S\}$
5. $\quad$ **for each** $v_p \in \{v_s, v_t\}$ **do**
6. $\quad\quad$ **for each** $r_{v_m} \in R(v_m, S)$ **do**
7. $\quad\quad\quad$ **if** $|v_p - v_m| \leq min(r_{v_m}, r_{v_p})$ **then** /* assume adding an edge (v_m, v_p) */
8. $\quad\quad\quad$ $F=min(F, F(s,m)+F(m,t)+CI(m, r_{v_m}))$
9. **return** F

As for the function $G(s,t)$, in order to satisfy condition 4), there are two choices. One is that v_s is directly connected to v_t, such that $G(s,t)=F(s,t)$. The other is v_s and v_t are connected by some other nodes inside $\overline{v_s v_t}$. Then, there must be at least one node v_m inside $\overline{v_s v_t}$ which is adjacent to v_s, and $G(s,t)$ can consist of $F(s,m)$, $G(m,t)$, and $CI(m, r_{v_m})$. Similar to Algorithm 1, $G(s,t)$ can be computed in Algorithm 2.

Algorithm 2. Compute $G(s,t)$

$G=+\infty$
if $|v_s - v_t| \leq min(r_{v_s}, r_{v_t})$ **then** /* assume adding an edge (v_s, v_t) */
$\quad G=F(s,t)$
$S=\{v_s, v_{s+1}, ..., v_t\}$
for each $v_m \in S/\{v_s, v_t\}$ **do**
$\quad R(v_m, S)=\{|u-v_m||u \in N(v_m) \bigcap S\}$
$\quad$ **for each** $r_{v_m} \in R(v_m, S)$ **do**
$\quad\quad$ **if** $|v_s - v_m| \leq min(r_{v_m}, r_{v_s})$ **then** /* assume adding an edge (v_s, v_m) */
$\quad\quad$ $G=min(G, F(s,m)+G(m,t)+CI(m, r_{v_m}))$
return G

With F and G, the minimum average interference can be computed in Algorithm 3 by calling $G(0, n-1)$.

Algorithm 3. Compute the minimum average interference

$total=+\infty$
for each $r_{v_0} \in R(v_0)$ **do**
$\quad$ **for each** $r_{v_{n-1}} \in R(v_{n-1})$ **do**
$\quad total=min(total,$
$\quad\quad CI(v_0, r_{v_0})+CI(v_{n-1}, r_{v_{n-1}})+G(0, n-1))$
return $\frac{total}{n}$

When computing the minimum average interference, we record v_m and r_{v_m} for each function $G(s,t)$, and v_p, v_m and r_{v_m} for each function $F(s,t)$. Through tracing backwards, we can construct a connected topology of $n-1$ edges with the minimum average interference, which is the optimal spanning tree. For conciseness, we omit the traceback function here. The correctness of the above algorithms are verified through comparing our results with the outputs generated by the brute-force search, which runs slowly in the exponential time $O(n^{\Delta})$. Figure 3 gives an example of an optimal spanning tree for the 6-node exponential chain. In our method, time is mainly spent on computing functions F and G. Since the number of possible transmission radii of a node can not exceed Δ, the time complexity to compute the optimal spanning tree with minimum average interference is $O(n^3\Delta^3)$.

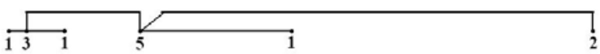

Fig. 3. The spanning tree for the 6-node exponential chain with minimum average interference $\frac{13}{6}$: the numbers next to each node is interference it creates

4 Minimizing the Maximum Interference

4.1 General Ideas

For the n nodes, $V=\{v_1,v_2,...,v_{n-1}\}$, the minimum maximum node interference in all the possible spanning trees is denoted as k, where $k\le\Delta\le n-1$ since all the nodes have the same maximum transmission radius r_{max}. In this section, we first design an algorithm to check whether there is a spanning tree with the maximum interference no larger than k set from 1 to $n-1$. After computing k, we can construct the optimal tree with such a maximum interference by traceback.

For a segment $\overline{v_sv_t}$, even when the nodes inside are not allowed to be adjacent to the ones outside, they still interfere with the outside nodes. We record all the interference from the nodes on $\overline{v_sv_t}$ to the outside nodes as a set $C(v_s,v_t,k)$, where $s\le k$. Each element $c(v_s,v_t,k)\in C(v_s,v_t,k)$, called *a skeleton* of the topologies on $\overline{v_sv_t}$, stores the following nodes and their transmission radii:

1) if $s>0$ and $t<n-1$: the nodes on $\overline{v_sv_t}$ that interfere with v_{s-1} or v_{t+1};
2) if $s=0$ and $t<n-1$: the nodes on $\overline{v_sv_t}$ that interfere with v_{t+1};
3) if $s>0$ and $t=n-1$: the nodes on $\overline{v_sv_t}$ that interfere with v_{s-1};
4) if $s=0$ and $t=n-1$: meaningless.

Specifically, $C(v,v,k)$ has $|R(v)|$ elements that store the node v and its potential transmission radii in $R(v)$. Since there must be no more than k nodes on $\overline{v_sv_t}$ that interfere with the left or the right nodes outside respectively, we call a skeleton $c(v_s,v_t,k)$ *valid* if and only if there are no more than k nodes in it that interfere with the first node left or right to $\overline{v_sv_t}$ respectively. Figure 4 gives an example of a valid skeleton $c(v_s,v_t,3)$ and two different topologies built according to the skeleton on $\overline{v_sv_t}$, where only v_s and v_{s+2} interfere with v_{s-1}, and only

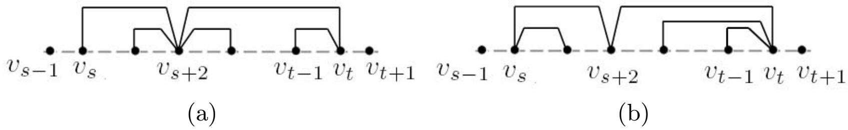

Fig. 4. The skeleton $c(v_s, v_t, 3) = \{(v_s, r_{v_s} = |v_s - v_{s+2}|), (v_{s+2}, r_{v_{s+2}} = |v_t - v_{s+2}|), (v_t, r_{v_t} = |v_t - v_{s+2}|)\}$ on the segment $\overline{v_s v_t}$. (a) and (b) are two possible topologies computed according to $c(v_s, v_t, 3)$

v_t interferes with v_{t+1}. Note that a valid skeleton does not guarantee that the maximum interference in the whole topology would not exceed the maximum, such as $RI(v_{s+2}) = 4 > 3$ in Figure 4(a).

Further, given $c(v_0, v_s, k)$, $c(v_s, v_t, k)$ and $c(v_t, v_{n-1}, k)$, to compute the topology on $\overline{v_s v_t}$, the following two requirements need to be satisfied: 1) together with the interference from nodes in $c(v_s, v_t, k)$, each node outside $\overline{v_s v_t}$ can not be interfered with more than k nodes; and 2), together with interference from nodes in $c(v_0, v_s, k)$ and $c(v_t, v_{n-1}, k)$, each node on $\overline{v_s v_t}$ can not be interfered with more than k nodes. Considering the mutual interference among the nodes on or outside each segment, we can design an algorithm to check whether there is a spanning tree with maximum interference no greater than k by dynamic programming as follows.

4.2 Algorithms

First of all, we define a function $Merge(c(v_{p_1}, v_{p_2}, k), c(v_{p_2+1}, v_{p_3}, k), ...,$ $c(v_{p_{m-1}}, v_{p_m}, k))$, where $0 \le p_1 \le p_2 \le ... \le p_m \le n-1$, to merge the skeletons on the consecutive segments and return $c(v_{p_1}, v_{p_m}, k)$. The method is to check every node in the skeletons whether to interfere with the first node left or right to $\overline{v_{p_1} v_{p_m}}$. Note that after merging, the new skeleton $c(v_{p_1}, v_{p_m}, k)$ may not be valid. Similar to compute the average interference, here we define two auxiliary boolean functions. The function *boolean* $F^*(s, t, k)$[2], where $s < t$, is to check whether there is a topology on $\overline{v_s v_t}$ that satisfies the following conditions simultaneously:

1) the transmission radius of v_s is r_{v_s}.
2) the transmission radius of v_t is r_{v_t}.
3) all the nodes inside $\overline{v_s v_t}$ can be only adjacent to the ones on $\overline{v_s v_t}$.
4) the skeleton for $\overline{v_{s+1} v_{t-1}}$ is $c(v_{s+1}, v_{t-1}, k)$.
5) $RI(v) \le k$, for each v inside $\overline{v_{s+1} v_{t-1}}$.
6) each node inside $\overline{v_s v_t}$ have a path either to v_s or to v_t.

Similarly, the function *boolean* $G^*(s, t, k)$, where $s < t$, is to check whether there is a topology on $\overline{v_s v_t}$ that satisfies the following conditions simultaneously: $1), 2), 3), 4)$ and 5) are the same as the first five conditions for $F^*(s, t, k)$. 6) all the nodes on $\overline{v_s v_t}$ are connected to each other directly or by nodes on $\overline{v_s v_t}$.

[2] For conciseness, we use $F^*(s, t, k)$ to stand for $F^*(v_s, v_t, s, t, r_{v_s}, r_{v_t}, c(v_0, v_s, k),$ $c(v_{s+1}, v_{t-1}, k), c(v_t, v_{n-1}, k))$, and $G^*(s, t, k)$ to stand for $G^*(v_s, v_t, s, t, r_{v_s}, r_{v_t},$ $c(v_0, v_s, k), c(v_{s+1}, v_{t-1}, k), c(v_t, v_{n-1}, k))$.

For $F^*(s,t,k)$, we still assume that there has been a path from v_s to v_t before adding any edges to the nodes inside $\overline{v_s v_t}$. When $s < t-1$, to satisfy the condition 6), there must be a node v_m inside $\overline{v_s v_t}$ that is adjacent to v_p, where $v_p = v_s$ or v_t. Therefore, there is a path from v_s to v_m as well as from v_m to v_t. With ensuring $RI(v_m) \leq k$, $F^*(s,t,k)$ is divided to check $F^*(s,m,k)$ and $F^*(m,t,k)$. So it can be computed in Algorithm 4. Lines 3–15 are to compute $F^*(s,t,k)$ recursively. In line 7, we assume adding an edge (v_m, v_p). Line 8 and 9 enumerate the possible skeletons on $\overline{v_{s+1} v_{m-1}}$ and $\overline{v_{m+1} v_{t-1}}$, and line 10 is to ensure the condition 4) is satisfied. In line 11, $c(v_0, v_m, k) = Merge(c(v_0, v_s, k), c(v_{s+1}, v_{m-1}, k), c(v_m, v_m, k))$ and $c(v_m, v_{n-1}, k) = Merge(c(v_m, v_m, k), c(v_{m+1}, v_{t-1}, k), c(v_t, v_{n-1}, k))$, and line 12 is to check their validity. All the three components for $F^*(s,m,k)$ are checked in line 13, and line 16 returns the value of $F^*(s,t,k)$.

Algorithm 4. Compute *boolean* $F^*(s,t,k)$

1. **if** $s+1=t$ **then** **return** $F^* = true$ /* the boundary condition*/
2. $S = \{v_s, v_{s+1}, ..., v_t\}$
3. **for each** $v_m \in S/\{v_s, v_t\}$ **do**
4. $R(v_m, S) = \{|u - v_m| \,|\, u \in N(v_m) \bigcap S\}$
5. **for each** $v_p \in \{v_s, v_t\}$ **do**
6. **for each** $r_{v_m} \in R(v_m, S)$ **do**
7. **if** $|v_p - v_m| \leq min(r_{v_m}, r_{v_p})$ **then** /* assume adding an edge (v_m, v_p) */
8. **for each** $c(v_{s+1}, v_{m-1}, k) \in C(v_{s+1}, v_{m-1}, k)$ **do**
9. **for each** $c(v_{m+1}, v_{t-1}, k) \in C(v_{m+1}, v_{t-1}, k)$ **do**
10. **if** $Merge(c(v_{s+1}, v_{m-1}, k), c(v_m, v_m, k), c(v_{m+1}, v_{t-1}, k)) = c(v_{s+1}, v_{t-1}, k)$ **then**
11. compute $c(v_0, v_m, k)$ and $c(v_m, v_{n-1}, k)$ by merging
12. **if** $c(v_0, v_m, k)$ is valid **&&** $c(v_m, v_{n-1}, k)$ is valid
13. **&&** no more than k nodes in $c(v_0, v_s, k)$, $c(v_{s+1}, v_{m-1}, k)$, $c(v_{m+1}, v_{t-1}, k)$ and $c(v_t, v_{n-1}, k)$ that interfere with v_m
14. **&&** $F^*(s,m,k)$ **&&** $F^*(m,t,k)$ **then**
15. **return** $F^* = true$
16. **return** $F^* = false$ /* no topology on $\overline{v_s v_t}$ to satisfy the 6 conditions*/

To compute $G^*(s,t,k)$, we actually assume there is no path from v_s to v_t before adding edges to the nodes inside $\overline{v_s v_t}$. In order to satisfy the condition 6), there must be a node v_m inside $\overline{v_s v_t}$ that is adjacent to v_s. Similarly, with ensuring $RI(v_m) \leq k$, $G^*(s,t,k)$ is divided to check $F^*(s,m,k)$ and $G^*(m,t,k)$. Algorithm 5 gives a detailed description on how to compute $G^*(s,t,k)$.

By calling $F^*(s,t,k)$ and $G^*(s,t,k)$, we design the main function, $FindMinMax(V)$, to find the minimum maximum interference k. From 1 to $n-1$. We check and return k immediately when a spanning tree with the maximum interference of k is found. Algorithm 6 illustrates how to compute the function $FindMinMax(V)$ by calling $G^*(0, n-1, k)$, where we only consider the cases when $|V| = n > 2$.

After computing $FindMinMax(V)$, we can do traceback and construct the optimal spanning tree with the minimum maximum interference by adding exactly $n-1$ edges. To save space, we omit the traceback function here.

Algorithm 5. Compute *boolean* $G^*(s,t,k)$

if $|v_s - v_t| \leq min(r_{v_s}, r_{v_t})$ **&&** $F^*(s,t,k) = true$ **then**
  return $G^* = true$ /* assume adding an edge (v_s, v_t) */
$S = \{v_s, v_{s+1}, ..., v_t\}$
for each $v_m \in S/\{v_s, v_t\}$ **do**
$R(v_m, S) = \{|u - v_m| | u \in N(v_m) \bigcap S\}$
  for each $r_{v_m} \in R(v_m, S)$ **do**
    if $|v_s - v_m| \leq min(r_{v_m}, r_{v_s})$ **then** /* assume adding an edge (v_s, v_m) */
      for each $c(v_{s+1}, v_{m-1}, k) \in C(v_{s+1}, v_{m-1}, k)$ **do**
        for each $c(v_{m+1}, v_{t-1}, k) \in C(v_{m+1}, v_{t-1}, k)$ **do**
          if $Merge(c(v_{s+1}, v_{m-1}, k), c(v_m, v_m, k), c(v_{m+1}, v_{t-1}, k)) = c(v_{s+1}, v_{t-1}, k)$ **then**
            compute $c(v_0, v_m, k)$ and $c(v_m, v_{n-1}, k)$ by merging
            if $c(v_0, v_m, k)$ is valid **&&** $c(v_m, v_{n-1}, k)$ is valid
            && no more than k nodes in $c(v_0, v_s, k)$, $c(v_{s+1}, v_{m-1}, k)$, $c(v_{m+1}, v_{t-1}, k)$ and $c(v_t, v_{n-1}, k)$ that interfere with v_m
            && $F^*(s, m, k)$ **&&** $G^*(m, t, k)$ **then**
              return $G^* = true$
return $G^* = false$ /* no topology on $\overline{v_s v_t}$ to satisfy the 6 conditions*/

Algorithm 6. Compute $FindMinMax(V)$, and return the minimum maximum interference

$k = 1$
while $k \leq n - 1$ **do**
  for each $r_{v_0} \in R(v_0)$ **do**
    for each $r_{v_{n-1}} \in R(v_{n-1})$ **do**
      for each $c(v_1, v_{n-2}, k) \in C(v_1, v_{n-2}, k)$
        if no more than k nodes in $c(v_1, v_{n-2}, k)$ and v_{n-1} that interfere with v_0
        && no more than k nodes in $c(v_1, v_{n-2}, k)$ and v_0 that interfere with
        v_{n-1} **&&** $G^*(0, n-1, k)$ **then**
          return k
  k=k+1
End While
return $+\infty$ / *no connected topology on V with the constraint of r_{max} */

4.3 Analysis

Correctness. The method has been verified through comparing our optimal topologies with the outputs generated by the brute-force search running in time $O(n^{\Delta})$. Moreover, our algorithms can find all the topologies of minimum maximum interference without crosses. For example, we can find all the 17 optimal topologies without a cross for the 6-node chain (Figure 5).

Time Complexity. Firstly, we analyze the size of the set $C(v_s, v_t, k)$ when $s > 0$ and $t < n-1$. The nodes in each element $c(v_s, v_t, k)$ can be divided as the node sets CL and CR which contain the nodes on $\overline{v_s v_t}$ that interfere with v_{s-1} and v_{t+1} respectively. As the maximum interference is k, we get $|CL| \leq k \leq \Delta$. Each node has different transmission radii of Δ as many as possible. Thus, the number of combinations for the nodes in CL and their transmission radii is

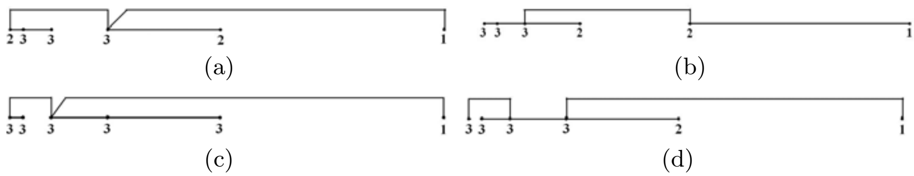

Fig. 5. 4 different optimal spanning trees for the 6-node exponential chain with the minimum maximum interference

$$\binom{\Delta}{0} + \binom{\Delta}{1} \times \Delta + ... + \binom{\Delta}{k} \times \Delta^k = O(\Delta^{2k}). \tag{7}$$

A similar result can be obtained for CR. Therefore, the size of $C(v_s, v_t, k)$ is $O(\Delta^{4k})$. Further, the total amount of the functions $F^*(s,t,k)$ and $G^*(s,t,k)$ is $O(n^2\Delta^{O(k)})$. For each function, the computing time is $O(n\Delta^{O(k)})$. As there are no functions being repeatedly computed, the time to finish $FindMinMax(V)$ will be $O(n^3\Delta^{O(k)})$. To construct the optimal spanning tree, the main time is to compute k by $FindMinMax(V)$. Thus, the time complexity to construct the spanning tree with the minimum maximum interference is $O(n^3\Delta^{O(k)})$. Since $\Delta \leq n-1$ and $k = O(\sqrt{\Delta})$ [14], the time is sub-exponential. However, when Δ is small, which means a low maximum node degree, our algorithm is fast.

Space Complexity. The space is mainly for storing the functions F^* and G^* as well as the sets $C(v_s, v_t, k)$. Therefore, the space complexity is $O(n^2\Delta^{O(k)})$.

5 Conclusion

In this paper, we study the problem to minimize the receiver-centric interference for the highway model. Based on the no-cross property and using dynamic programming, the first polynomial-time exact algorithm for constructing a connected topology with minimum average interference is proposed. We give also the first sub-exponential-time exact algorithm for constructing a connected topology while minimizing the maximum interference. The question of whether it is NP-hard to minimize the maximum interference for the highway model is still open. Related open problems include how to design an approximation with a ratio better than $O(\sqrt[4]{\Delta})$ for the highway model, how to design efficient approximations to minimize the maximum interference in 2D networks, and how to combine the interference minimization with other network properties.

References

1. Banner, R., Orda, A.: Multi-Objective Topology Control in Wireless Networks. In: INFOCOM 2008, pp. 448–456 (2008)
2. Buchin, K.: Minimizing the Maximum Interference is Hard, arXiv: 0802.2134v1 (2008)

3. Burkhart, M., von Rickenbach, P., Wattenhofer, R., Zollinger, A.: Does Topology Control Reduce Interference? In: MobiHoc 2004, pp. 9–19 (2004)
4. Moaveni-Nejad, K., Li, X.-Y.: Low-interference topology control for wireless ad hoc networks. Int. J. of Ad Hoc & Sensor Wireless Networks 1, 41–64 (2005)
5. Wu, K.-D., Liao, W.: On Constructing Low Interference Topology in Multihop Wireless Networks. Int. J. of Sensor Networks 2, 321–330 (2007)
6. Johansson, T., Carr-Motyčková, L.: Reducing Interference in Ad Hoc Networks Through Topology Control. In: DIALM-POMC 2005, pp. 17–23 (2005)
7. Benkert, M., Gudmundsson, J., Haverkort, H., Wolff, A.: Constructing minimum-interference networks. Comp. Geometry 40(3), 179–194 (2008)
8. Damian, M., Javali, N.: Distributed construction of bounded-degree low-interference spanners of low weight. In: MobiHoc 2008, pp. 101–110 (2008)
9. Halldórsson, M., Tokuyama, T.: Minimizing interference of a wireless ad hoc network in a plane. In: Nikoletseas, S.E., Rolim, J.D.P. (eds.) ALGOSENSORS 2006. LNCS, vol. 4240, pp. 71–82. Springer, Heidelberg (2006)
10. Li, X.Y., Song, W.Z., Wang, W.: A Unified Energy Efficient Topology for Unicast and Broadcast. In: MOBICOM 2005, pp. 1–15 (2005)
11. Locher, T., von Rickenbach, P., Wattenhofer, R.: Sensor networks continue to puzzle: Selected open problems. In: Rao, S., Chatterjee, M., Jayanti, P., Murthy, C.S.R., Saha, S.K. (eds.) ICDCN 2008. LNCS, vol. 4904, pp. 25–38. Springer, Heidelberg (2008)
12. Moscibroda, T., Wattenhofer, R.: Minimizing interference in ad hoc and sensor networks. In: DIALM-POMC 2005, pp. 24–33 (2005)
13. Santi, P.: Topology Control in Wireless Ad Hoc and Sensor Networks. Wiley, Chichester (2005)
14. von Rickenbach, P., Schmid, S., Wattenhofer, R., Zollinger, A.: A robust interference model for wireless ad hoc networks. In: WMAN 2005 (2005)
15. von Rickenbach, P., Wattenhofer, R., Zollinger, A.: Algorithmic Models of Interference in Wireless Ad Hoc and Sensor Networks. IEEE/ACM Trans. on Net (TON) 17(1), 172–185 (2009)