Generalized Broadcast Scheduling in Duty-Cycle Multi-Hop Wireless Networks

Yueming Duan  Shouling Ji  Zhipeng Cai
Department of Computer Science
Georgia State University
Atlanta, Georgia, 30303, USA
Email: {yduan1, sji, zcai}@cs.gsu.edu

Abstract

Network wide broadcast is a fundamental and widely-used operation in multi-hop wireless networks. Through one-to-all broadcast, a source node disseminates a message to all the other nodes in the network. In duty-cycle scenarios, the nodes alternate between active and sleep states, which results in the increasing of broadcast latency. In this paper, we investigate the Minimum Latency Broadcast Scheduling problem in Duty-Cycle (MLBSDC problem) multi-hop wireless networks. Our goal is to provide a generalized framework for the MLBSDC problem. We mathematically remodel this problem and propose a vector iteration algorithm, in which a greedy strategy is employed.

We also conduct extensive simulations to show the efficiency of the broadcast schedule returned by our algorithm under various network configurations. Compared with the results of the current best algorithm, our greedy algorithm achieves significant improvement both in broadcast latency and the total number of transmissions.

1 Introduction

Wireless networks usually consist of nodes with limited transmission ranges, thus multi-hop broadcast is one of the most fundamental and important communication modes. Multi-hop broadcast is employed mainly for network-wide commands dissemination.

The multi-hop broadcast has been well studied in always-on wireless networks [5-11]. Unfortunately, a node in duty-cycle wireless networks usually requires transmitting multiple times to inform all its neighbor nodes in different time slots, which results in not only more broadcast latency, but also more transmissions. Therefore, most of the previously proposed broadcast scheduling algorithms are less effective in duty-cycle scenarios.

Recently, some problems have been made to solve the multi-hop broadcast in duty-cycle wireless networks, e.g., [1-3]. Currently, the best approximation ratio for the MLBSDC problem under the Unit Disk Graph (UDG) model is 17/7 [3], where 7 denotes the number of time slots in a scheduling period, and the periodically wake up pattern is pre-defined. The OTAB in [3] schedules a broadcast to avoid collisions through D2-color, which cannot deal with the interference problem. It essentially employs a layer-by-layer approach, and numerous time slots are wasted in a schedule.

In this paper, our objective is to provide a generalized framework to solve the MLBSDC problem. We consider the broadcast scheduling in an arbitrary directed graph, and both the duty cycle length and wake up pattern of each node are arbitrary. Moreover, our broadcast scheduling is considered under interference environment. In interference-free broadcast schedule, a node u receives the message only if exactly one node v broadcast the message, where u is in the transmission range of v, and any other node v' cannot broadcast the message if node u is within the interference range of v'. It is worth mentioning that the interference range of a node is much larger than its transmission range [11]. However, this factor has not been taken into account in [2], [3].

Our contributions can be summarized as follows. Firstly, we introduce a more generalized model to characterize the essentials of broadcast scheduling in duty-cycle wireless networks under interference environment. Secondly, we illustrate how to employ our greedy algorithm to solve the MLBSDC problem with the collision-free purpose [3] in a general directed graph. For collision-free broadcast scheduling, our algorithm has an approximation ratio of 1/17, where Δ is the maximum out degree of nodes in a graph, and |T| is the number of time slots in a scheduling period. Finally, we conduct extensive simulations to evaluate the performance of our algorithm under different network configurations. The simulation results demonstrate that our algorithm has significant improvements on both broadcast latency and the total number of transmissions compared with OTAB [3]. Moreover, the simulation results show that the broadcast latency of schedules obtained by our algorithm-
m are very close to the lower bounds, indicating that schedules produced by our algorithm are near optimal.

2 Related Work

As a fundamental and important service, multi-hop broadcast has been extensively studied in literatures [1-11]. Some previous works [6-9] study the minimum latency broadcast scheduling (MLBS) problem under the Unit Disk Graph (UDG) model, in which the transmission radius of every node is assumed to be the same, and the communication link is bidirectional and usually determined by the distance between each pair of nodes. Interference-free broadcast scheduling also has been studied in [10-12]. However, there are very few works for the MLBS problem taking a sleep schedule into consideration. In [1], the multi-hop broadcast problem was modeled as a shortest-path dynamical programming problem. Hong et al. [2] proved the NP-hardness of sleeping schedule-aware minimum latency broadcast problem in duty-cycle scenarios and provided a heuristic algorithm ELAC with constant approximation ratio of 24|T| + 1, where |T| is the number of time slots in a scheduling period. Later, Jiao et al. [3] proposed OTAB to further improve the constant approximation ratio to 17|T|.

3 Preliminaries

3.1 Network Model and Assumptions

We consider a static wireless network including n wireless nodes in this paper. Let \( V = \{s_1, s_2, \ldots, s_n\} \) denotes the set of all the nodes. In duty-cycle scenarios, all the nodes in a network alternate between active and sleep states. To simplify the analysis, we assume the time is slotted and all time slots have the same duration \( \tau \). Once a node wakes up, it stays in the active state for one time slot. A node \( s_i \) can forward a message once it receives the broadcast message, while it is only allowed to receive messages at its active time slots, and it cannot forward and receive the message at the same time slot. For generality, we assume that nodes independently determine their active/sleep schedules in advance similar to [2] and [3]. To be specific, we define a cluster of functions \( f(t) : s_i \rightarrow \{0, 1\} \) \( (1 \leq i \leq n) \), to indicate the active or sleep states of node \( s_i \) at time slot \( t \). \( f_i(t) = 1 \) if \( s_i \) is active at time slot \( t \), otherwise \( f_i(t) = 0 \).

Let \( E = \{(s_i, s_j) \mid \text{dist}(s_i, s_j) \leq r_{T_i}\} \), where \( \text{dist}(s_i, s_j) \) is the Euclidean distance of node \( s_i \) and \( s_j \), and \( r_{T_i} \) is the transmission radius of node \( s_i \). Based on the above assumptions, we model a duty-cycle multi-hop wireless network as a time dependent graph \( G^t = (V, E^t) \), where \( E^t \) is the set of communication links at time slot \( t \), i.e., \( E^t = \{(s_i, s_j) \mid (s_i, s_j) \in E \text{ and } f_i(t) = 1\} \).

3.2 Interference Model

In this paper, the broadcast scheduling is considered under the protocol interference model, in which each node \( s_i \) is associated with two radii: the transmission radius \( r_{T_i} \) and the interference radius \( r_I \), where \( r_I \leq r_{T_i} \). Interference ratio \( \alpha \) is defined as the ratio of interference radius and transmission radius. The transmission range of a node \( s_i \) is modeled as a disk centered at \( s_i \) with radius \( r_{T_i} \), and the interference range of \( s_i \) is a disk with radius \( r_I \), centered at \( s_i \) as well.

Let \( TR_i = \{ s_j \mid \text{dist}(s_i, s_j) \leq r_I \} \), i.e., \( TR_i \) is the set of nodes which can transmit message to node \( s_i \); \( I_i = \{ s_j \mid \text{dist}(s_i, s_j) \leq r_I \} \), which is the set of nodes interfere with node \( s_i \). Node \( s_i \) successfully receives a message only if exactly one node \( s_j \in TR_i \) broadcast this message and any other nodes in \( I_i \) do not broadcast any message at the same time slot.

3.3 Problem Formulation

Considering a predefined source node \( s \in V \) disseminates a message to all the other nodes. The broadcast latency is defined as the total time slots such that all the nodes in \( V \) successfully receive the broadcast message.

A valid broadcast schedule \( S = (S^1, S^2, \ldots, S^t) \) satisfies the following requirements: 1) \( S^t \subseteq V \) representing the set of nodes scheduled to forward the message at time slot \( t \), where \( i \in N \) and \( 1 \leq i \leq t \); 2) a node is only allowed to receive a message at its active time slots, and it cannot forward and receive the message at the same time; 3) any node \( s_i \) cannot be scheduled to forward the message until it receives the broadcast message; 4) at the end of time slot \( t \), all the nodes in \( V \) successfully receive the message from the source node \( s \).

Denote \( S \) as the space of valid schedules. The MLBSDC problem is defined as follows. Given a general directed graph \( G = (V, E) \), the sleep-active function set \( \{f_i(t)\mid i \in N, 1 \leq i \leq |V|\} \) and a source node \( s \in V \), find a valid broadcast schedule \( S \in S \) such that the broadcast latency is minimized.

Obviously, the MLBSDC problem is NP-hard, which can be easily proven by reduction from the conventional MLBS problem [5]. The MLBS problem is essentially a special case of the MLBSDC problem.

4 Broadcast Scheduling Algorithm

4.1 Vector-Iteration Algorithm

Before presenting the broadcast scheduling algorithm, we first introduce the following concepts to facilitate observing the essential of our broadcast scheduling.
Adjacency Matrix: Define an \( n \times n \) adjacency matrix \( K \), in which the element \( K_{ij} = 1 \) if \((s_i, s_j) \in E\), otherwise \( K_{ij} = 0 \). This adjacency matrix defines all the communication links with the assumption that all the nodes are in the active state.

Node-State Matrix: Define an \( n \times n \) node-state matrix \( W_t \), which indicates the states of all the nodes in a network at each time slot \( t \), where \( W_{ij}^t = f_j(t) \) and \( W_{ij}^t = 0 \), if \( i \neq j \).

Message-Holding Vector: Let \( V^t = (v_1^t, v_2^t, \ldots, v_n^t) \) to indicate whether a node \( s_i \) has already received the broadcast message at the end of time slot \( t \). To be specific, \( v_i^t = 1 \) if at the end of time slot \( t \), node \( s_i \) has already received the broadcast message, otherwise, \( v_i^t = 0 \). We assume initially the node \( s_1 \) has a message to broadcast, i.e., \( V^0 = (1, 0, \ldots, 0) \).

Broadcast-Set Vector: Let \( x^t = (x_1^t, x_2^t, \ldots, x_n^t) \) to indicate whether a node \( s_i \) is scheduled to broadcast the message at time slot \( t \). Node \( s_i \) is scheduled at time slot \( t \) if \( x_i^t = 1 \), otherwise \( x_i^t = 0 \).

Based on the above definitions, we have the following recursive function:

\[
V^t = V^{t-1} \lor x^t KW^t \quad (1)
\]

\[
V^0 = (1, 0, \ldots, 0) \quad (2)
\]

where \( t \geq 1, V^{t-1} \lor x^t \geq 0 \), and \( \lor \) is defined as component or.

According to the definition of message holding vector \( V^t \), the left side of the recursive equation represents all the nodes who have already received the broadcast message at the end of time slot \( t \). Considering the right side, let \( M^t = KW^t \), then \( E^t \) is determined by \( M^t \), i.e., \( E^t = \{(s_i, s_j) \mid M^t_{ij} = 1\} \), which is the available transmission links at time slot \( t \). \( x^t \) indicates the set of nodes being scheduled to broadcast message at time slot \( t \), then \( x^t M^t \) is exact the set of the receiving nodes at time slot \( t \). Thus, Equation (1) holds.

The original MLBSDC problem is equivalent to finding a sequence of Broadcast-Set Vectors \( X = (x^1, x^2, \ldots, x^t) \) to make \( V^t = 1 \), with the objective of minimizing \( t \). Since The MLBSDC problem is NP-Hard, instead of finding the optimal \( X^t \) to minimize \( t \), we solve this problem by heuristically constructing \( x^t \) at each time slot \( i \). To simplify the construction of \( x^t \), we introduce candidate broadcasting and receiving sets as follows.

Candidate Broadcasting Set (CBS): At the beginning of each time slot \( t \), only the nodes who have already received the message can broadcast the message. Let \( CBS^t = \{s_i \mid V^{t-1}_{i} = 1\} \), then for any valid schedule \( S = (S^1, S^2, \ldots, S^t), S^k \subseteq CBS^k \), where \( 1 \leq k \leq t \).

Candidate Receiving Set (CRS): Let \( Z^t = V^{t-1}KW^t \), and \( CRS^t = \{s_i \mid Z^t_{i} = 1\} \). Then, any possible set of receiving nodes is a subset of \( CRS^t \). Since \( CRS^t \) is derived based on the assumption that all the nodes in \( CBS^t \) are scheduled to broadcast the message.

Both \( CBS^t \) and \( CRS^t \) can be further pruned. On the one hand, it is unnecessary for the nodes who have already received the message at the beginning of time slot \( t \) to receive the message again, thus \( CRS^t \) can be further pruned as \( CRS^t = CRS^t - CBS^t \). On the other hand, if a node \( s_i \) in \( CBS^t \) cannot find any neighbor node in \( CRS^t \), it can be safely removed from \( CBS^t \), i.e., \( CBS^t = CBS^t - \{s_i \mid \forall s_j \in CRS^t((s_i, s_j) \notin E')\} \).

Algorithm 1: Vector-Iteration Algorithm

\[\begin{array}{ll}
\text{Algorithm } 1: \text{ Vector-Iteration Algorithm} \\
\text{input: } \mathbf{V}^0, K, \{f_j(t) \mid i \in N, 1 \leq i \leq n\} \\
\text{output: A Valid Broadcast Schedule } S = (S^0, S^1, \ldots, S^t) \\
1 \quad t \leftarrow 0; \\
2 \quad \text{while } \mathbf{V}^t \neq (1, 1, \ldots, 1) \text{ do} \\
3 \quad \quad \quad \text{for } i \leftarrow 1 \text{ to } n \text{ do} \\
4 \quad \quad \quad \quad \text{if } j \leftarrow 1 \text{ to } n \text{ do} \\
5 \quad \quad \quad \quad \quad \text{if } i = j \text{ and } f_j(t) = 1 \text{ then } W_{ij}^t \leftarrow 1; \\
6 \quad \quad \quad \quad \quad \text{else } W_{ij}^t = 0; \\
7 \quad \quad \quad \quad \quad \text{M}^t \leftarrow K W^t; \\
8 \quad \quad \quad \quad \quad \text{V}^t \leftarrow \text{V}^{t-1} \text{M}^t; \\
9 \quad \quad \quad \quad \quad \text{E}^t \leftarrow \{(s_i, s_j) \mid M^t_{ij} = 1\}; \\
10 \quad \quad \quad \quad \quad \text{CBS}^t \leftarrow \{s_i \mid V^{t-1}_{i} = 1\}; \\
11 \quad \quad \quad \quad \quad \text{CRS}^t \leftarrow \{s_i \mid Z^t_{i} = 1\} - CBS^t; \\
12 \quad \quad \quad \quad \quad \text{CBS}^t \leftarrow \text{CBS}^t - \{s_i \mid \forall s_j \in CRS^t((s_i, s_j) \notin E')\}; \\
13 \quad \quad \quad \quad \quad \text{E}^t \leftarrow \{(s_i, s_j) \mid s_i \in CBS^t, s_j \in CRS^t \text{ and } (s_i, s_j) \in E'\}; \\
14 \quad \quad \quad \quad \quad \text{G}^t \leftarrow (CBS^t, CRS^t, E^t); \\
15 \quad \quad \quad \quad \quad \text{S}^t \leftarrow \text{Greedy Compatible Broadcast-Set Construction Algorithm (G);} \\
16 \quad \quad \quad \quad \text{for } i \leftarrow 1 \text{ to } n \text{ do} \\
17 \quad \quad \quad \quad \quad \text{if } s_i \in S^t \text{ then } x^t_i \leftarrow 1; \\
18 \quad \quad \quad \quad \quad \text{else } x^t_i \leftarrow 0; \\
19 \quad \quad \quad \quad \quad \mathbf{x}^t \leftarrow (x^t_1, x^t_2, \ldots, x^t_n); \\
20 \quad \quad \quad \quad \quad \text{V}^t = \text{V}^{t-1} \lor \text{x}^t M^t; \\
21 \quad \quad \quad \quad \text{return } S; \\
\end{array}\]

Ideally, if all the nodes in \( CBS^t \) are simultaneously scheduled to broadcast the message without any interference at each time slot \( t \), then the schedule (\( CBS^1, CBS^2, \ldots, CBS^t \)) is obviously an optimal minimum latency broadcast schedule. Unfortunately, the nodes in \( \{s_i \mid s_j \in CBS^t((s_j, s_j) \in E') \} \) usually cannot successfully receive the message simultaneously at time slot \( r \) because of communication interference. Thus, we have to schedule a subset of nodes in \( CBS^t \) to broadcast the message, and how to choose the broadcast set to avoid communication interference will be discussed in the next subsection.

Based on the above discussion, the pseudo code of Vector-Iteration Algorithm (VIA) is summarized as in Algorithm 1, in which the subprocess Compatible Broadcast-Set Construction Algorithm will be discussed in the following subsection.
4.2 Compatible Broadcast-Set Construction

In this subsection, we discuss how to determine the set of nodes scheduled to broadcast the message at each time slot. Instead of assigning time slots to broadcast the message for each node as previous works [2] [3] did, we select nodes to broadcast the message such that as many nodes as possible can interference-freely receive the message at each time slot. Note that the selection space of a broadcast set at each time slot is $O(2^n)$, and different selection schemes usually result in different broadcast schedules.

Given graph $G(V_b, V_i, E_{br})$, where $V_b$ and $V_i$ denote the nodes in the candidate broadcast set and candidate receiving set respectively, and $V_b \cap V_i = \emptyset$. $E_{br} = \{(s_i, s_j) \mid s_i \in V_b, s_j \in V_i, \text{dis}(s_i, s_j) \leq r_T\}$. Let $g(s_i) = \{s_j \mid \exists(s_i, s_j) \in E_{br}\}$. We say node $s_i$ is compatible to node $s_j$ if and only if for all $s_k \in g(s_i), \text{dis}(s_i, s_k) > r_T$. A is a compatible broadcast set if and only if for all $s_i$ and $s_j$ are compatible to each other. The objective of the CBSC problem is to find an optimal compatible broadcast set $V^* \subseteq V_b$ to maximize $|\bigcup_{s \in V^*} g(s)|$, where $|A|$ is the cardinality of set $A$. Obviously, for all $s_i \in V^*, |g(s_i)| \geq 1$, otherwise, $s_i$ can be simply removed from $V_b$ without affecting the optimal broadcast set $V^*$.

The BCSC problem is obviously NP-Hard, which can be proven by reduction from the conventional maximum independent set problem in a general graph. The NP-hardness of finding a maximum independent set in a general graph has already been proven [12]. Since the Broadcast-Set Construction problem is NP-hard, we employ a greedy approach again to solve this problem. The CBSC problem is similar to the set-cover problem, except that the selected subsets have to be compatible, which makes the CBSC problem essentially more challenging than the set-cover problem. However, we can follow the same idea of the set-cover problem, greedily choosing nodes round-by-round. The pseudo code is shown in Algorithm 2, and the time complexity is $O(\Delta |V_b| \min(|V_b|, |V_i|))$, where $\Delta$ is the maximum out degree of nodes in the candidate broadcast set $V_b$.

4.3 Collision-free Broadcast Scheduling

In this subsection, we consider the collision-free MLBSDC problem in [3]. If two neighbors $v_1$ and $v_2$ of $u$ within $u$’s transmission range broadcast at a same time slot, we say a collision occurs at $u$ and it fails to receive any message. The collision-free MLBS aims to find a valid broadcast schedule such that all the nodes collision-freely receive the message with the minimum number of time slots.

A multi-hop wireless network is modeled as a UDG $G = (V, E)$ in [3]. According to the definition of $G$, the Adjacency Matrix $K$ can be easily constructed as following: arbitrarily assign an order for the nodes in $V$, named $(s_1, s_2, \ldots, s_{|V|})$ with only one condition that $s_1$ is the source node; then let $K_{ij} = 1$ if $\text{dis}(s_i, s_j) \leq r_T$, otherwise, $K_{ij} = 0$. OTAB [3] considers all the nodes in a network adopting the same scheduling period $T$, which is further divided into $\lfloor T \rfloor$ time slots. Each node $s_i \in V$ randomly and independently chooses one active time slot $T_i$, where $0 \leq T_i \leq \lfloor T \rfloor - 1$. Equivalently, let $f_i(t) = 1$ if $t \% |T| = T_i$, otherwise $f_i(t) = 0$.

Now we have equivalently constructed all the necessary inputs, and we can produce a collision-free broadcast schedule for the MLBSDC problem after running our VIA with the above constructed inputs.

4.4 Analysis

Assume node $s_i$ wakes up at least one time every $T_i$ time slots, where $1 \leq i \leq n$. Let $T = \sum_{i=1}^{n} T_i$, then the upper bound of the broadcast latency of the schedule returned by VIA is $O(T)$. Thus our algorithm stops in finite time. To be specific, the time complexity of VIA is $O(\Delta T n^2)$. For the correctness of our algorithm, we have the following lemma, which can be easily proved the lemma according to the definition of the valid schedule. Here we omit the proof due to limit space.

**Lemma 1** The Vector-Iteration Algorithm returns a valid broadcast schedule.

For collision-free MLBSDC in a general directed graph with homogeneous periodic wake up pattern, we have the following theorem.
**Theorem 1** VIA returns a schedule with approximation ratio of $\Delta |T|$, where $\Delta$ is the maximum out degree of the nodes in a graph, and $|T|$ is the number of the time slots in a schedule period.

**Proof**: Assume the optimal broadcast schedule is $S^* = (S_0^*, S_1^*, \ldots, S_{opt}^*)$, where $S_i^*$ contains the nodes being scheduled to broadcast at time slot $i$ in the optimal schedule, and the minimum broadcast latency is $opt + 1$. Let $U_k^n = \bigcup_{i=0}^n S_i^*$, and $R_n^u = \bigcup_{k=0}^{|T|} g(S_k)$. We finish the proof of this theorem by proving the following lemma.

**Lemma 2** For $\forall n \in N$ and $0 \leq n \leq opt$, all the nodes in $R_n^u$ receive the message before time slot $\Delta |T|(n + 1)$.

**Proof Sketch**: We prove this lemma by strong mathematical induction. In the induction step, when $0 \leq k < opt$, suppose all the nodes in $R_k^n$ receive the broadcast message before time slot $\Delta |T|(k + 1)$, it is enough to finish the proof by showing that all the nodes in $R_{k+1}^n$ receive the broadcast message before time slot $\Delta |T|(k+2)$. Since any pair of the nodes in $S_{k+1}^*$ should not collide with each other, we can analyze the nodes in $S_{k+1}^*$ independently and show that any node in $S_{k+1}^*$ will receive the message before time slot $|T|(k + 2)$.

## 5 Performance Evaluation

We conduct extensive simulations to evaluate the performance of VIA, and the results are compared with that of OTAB [3], which has the best approximation ratio for the MLBSDC problem under the UDG model. Both the broadcast latency and the total number of the transmissions are considered. For fairness, the simulation setting is the same as that of [3], we randomly deploy all the nodes in a rectangle area of 200m x 200m. The impact of network size and transmission radius on the performance of the algorithms are evaluated. The network size ranges from 200 to 1000 with an interval of 200 and the range of the transmission radius varies from 20m to 60m.

Similar to OTAB in [3], we conduct all the simulations with two parameters fixed and the other one varied. For each setting, the algorithms are run on 200 randomly generated graph topologies, and the source node is randomly chosen as well. The performances of the algorithms are evaluated according to the average results. In Fig. 1-2, TLB is the trivial lower bound of the broadcast latency. We derive this lower bound through ignoring all the interference in VIA. Thus, the real optimal broadcast latency lies above TLB. OTAB-P is the result related to the total number of the transmissions, in which a prune technology is employed to reduce the total number of the transmissions in [3].

### 5.1 Impact of Network Size

We first study the impact of network size on the performance of OTAB and VIA. In this group of simulations, the transmission radius is fixed to 30m, and the duty cycle length $|T|$ is fixed to 20. As shown in Fig. 1(a), the broadcast latency of OTAB is far from optimal, while VIA is near optimal. The reason is that OTAB essentially schedules the broadcast in a layer-by-layer pattern, in which the spatial parallelism of broadcast is not sufficiently considered, while VIA greedily schedules nodes in each iteration. To show the broadcast latency with the increasing of network size more clearly, we extract the broadcast latency of VIA and TLB into Fig. 1(b). When the network size increases, the broadcast latency of both VIA and TLB decrease. Since the deployment area is fixed, the node density grows with the increasing of the number of nodes. Intuitively, in higher-node-density networks, the connectivity of duty-cycled wireless networks is better, which results in smaller broadcast delay. From Fig. 1(c), we can observe that the total number of transmissions of VIA is less than that of OTAB and OTAB-P as well. The improvement is more obvious with the increasing of node density.

### 5.2 Impact of Transmission Radius

In this group of simulations, we evaluate the performances of VIA and OTAB with different transmission radii. The network size and the duty cycle length $|T|$ are fixed to 400 and 20, respectively. From Fig. 2 we can see that VIA outperforms OTAB significantly. The reason is generally the same as the above analysis. In Fig. 2(b), the broadcast latency of VIA deviates further from that of TLB with the increasing of the transmission radius. With the deployment area fixed, the interference is more serious with the increasing of the transmission radius, while TLB is derived without considering interference, and TLB deviates further away from the actual optimal broadcast latency. As a result, the broadcast latency delay scheduled by VIA is still close to the optimal one.

## 6 Conclusions

This paper investigates the minimum latency broadcast scheduling problem for duty-cycled wireless networks. An generalized greedy algorithm VIA is proposed for this problem. The correctness of the broadcast schedule returned by VIA is proven. Extensive simulation results are shown to verify the efficiency of VIA, which indicate that the broadcast latency of the schedules returned by VIA is close to the optimal ones.
Acknowledgment

This work is partly supported by the NSF under grant No.CNS-1152001.

References