Performance Improvement of Topology-Transparent Broadcast Scheduling in Mobile Ad Hoc Networks

Yiming Liu, *Student Member, IEEE*, Victor O. K. Li, *Fellow, IEEE*, Ka-Cheong Leung, *Member, IEEE*, and Lin Zhang, *Member, IEEE*

***Abstract*—Broadcasting is a key function in mobile ad hoc networks. Topology-dependent scheduling algorithms depend on the detailed network connectivity information and cannot adapt to dynamic topological changes in mobile networks. To overcome this limitation, topology-transparent broadcast scheduling algorithms have been proposed. Due to the large overhead of implementing the acknowledgement mechanism in broadcast communications and to guarantee that there is at least one collision-free transmission in each frame, most existing topology-transparent broadcast scheduling algorithms require a node to transmit the same packet repeatedly at multiple slots during one frame. This is very inefficient. In this paper, we propose an efficient topology-transparent broadcast scheduling algorithm. First, instead of transmitting the same packet repeatedly during one frame, each node collects transmission codes of its two-hop neighbors and transmits several different packets during one frame. Second, noting that many unassigned slots are not utilized, we propose several methods to utilize the unassigned slots efficiently in a collision-free and traffic-adaptive manner. Thus, our algorithm provides a guaranteed throughput and achieves a much better average throughput. The analytical and simulation results show that our proposed algorithm outperforms other existing topology-transparent broadcast algorithms dramatically.**

***Index Terms*—Ad hoc networks, broadcast, topology-transparent scheduling.**

I. Introduction

SCHEDULING medium access in mobile wireless ad hoc networks is challenging because of node mobility and the limited availability and variability of wireless bandwidth. Many transmission scheduling algorithms have been proposed to maximize the spatial reuse and minimize the time-division multiple-access (TDMA) frame length. In the conventional TDMA networks, each node is assigned a unique time slot to transmit. This works well when the connectivity information among the nodes is known and when the number of nodes in the network is not large [5]. In mobile ad hoc networks, the number of nodes is much larger than the number of neighbors of a node. Thus, the system performance can be greatly improved by applying spatial reuse. Previous approaches in topology-dependent scheduling [5], [7], [9], [19], [21] require each node to maintain accurate network connectivity information. This is impractical in mobile ad hoc networks, which are characterized by highly dynamic topological changes. An alternative approach is topology-transparent scheduling [3], [4], [12], in which each node is assigned multiple time slots to transmit in each frame, with the guarantee that at least one time slot will be successful. However, since the acknowledgement mechanism cannot be efficiently implemented in broadcast communications,[1] most existing topology-transparent broadcast scheduling methods require a node to transmit the same packet repeatedly at all of its assigned time slots during one frame time. In addition, many unassigned time slots in each frame are not utilized at all. Thus, they can only provide a guaranteed minimum throughput with relatively low network utilization.

In this paper, we propose an efficient topology-transparent broadcast scheduling algorithm with a different design strategy. The main contributions of our work are as follows.

1) First, unlike the existing topology-transparent broadcast scheduling algorithms that transmit one packet repeatedly during one frame time, in our algorithm, each node can transmit multiple packets during one frame time according to the collected information of its two-hop neighbors. Each node collects such information as the identification number (ID), transmission schedule, and priority of each of its two-hop neighbors. The details of the information collection will be introduced and discussed in Section II-B. Thus, our algorithm can also provide a guaranteed minimum throughput.
2) Second, observing the fact that the codes assigned to each node are highly sparse (i.e., many unassigned slots are not utilized at all), we propose several methods to utilize the unassigned slots in a collision-free and traffic-adaptive manner. We utilize the unassigned slots while ensuring that the transmissions in the assigned slots are not affected. Thus, our proposed broadcast scheduling algorithm provides a guaranteed throughput by utilizing the assigned slots and achieves a much better average throughput, compared with existing algorithms, by utilizing the unassigned slots. Moreover, our algorithm can provide a guaranteed and smaller delay than the

Manuscript received October 14, 2013; revised January 18, 2014; accepted March 8, 2014. Date of publication March 24, 2014; date of current version November 6, 2014. This work was supported by the Research Grants Council of Hong Kong under Grant HKU 714310E. The review of this paper was coordinated by Dr. Y. Ji.

Y. Liu and L. Zhang are with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China (e-mail: liu-ym05@mails.tsinghua.edu.cn; linzhang@tsinghua.edu.cn).

V. O. K. Li and K.-C. Leung are with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Kowloon, Hong Kong (e-mail: vli@eee.hku.hk; kcleung@ieee.org).

Digital Object Identifier 10.1109/TVT.2014.2313272

[1]The acknowledgements from different intended receivers may collide at the transmitter, leading to the failure of the acknowledgement mechanism.

existing algorithms.[2] This will be discussed in detail in Sections III and IV.

3) Third, we study the performance of the proposed algorithm analytically and conduct extensive simulations to show that our algorithm outperforms the other existing topology-transparent broadcast algorithms dramatically. Moreover, we investigate the effect of inaccuracies in the estimation of the maximum node degree on the performance of the proposed algorithm.

Note that although information (codes) on two-hop neighbors is collected in our topology-transparent algorithm, as in topology-dependent scheduling algorithms [5], [7], [9], [19], [21], some fundamental differences exist between these two algorithms. In our algorithm, each node determines its transmission pattern according to its assigned code, which remains the same when topology changes. Collecting the transmission codes of two-hop neighbors helps broadcast more packets within one frame time. However, in a topology-dependent scheduling algorithm, each node collects information on its two-hop neighbors, negotiates with its neighbors, and finally determines its transmission slots. When the topology changes, each node has to collect, compute, negotiate, and determine its transmission slots again, resulting in inefficiency and possible failure of such algorithms.

A. Related Work

Many medium access control (MAC) protocols have been developed for ad hoc networks. The contention-based approaches, such as carrier sense multiple access, cannot provide deterministic delay and throughput bounds. The primary drawback surfaces at high load, when these approaches spend most of their time resolving collisions. When the network load is very high, the throughput approaches zero, resulting in an unstable network. It is also shown that contention-based approaches suffer from serious instability and unfairness issues in multihop ad hoc networks [27], which is why most networks offering throughput and delay guarantees, such as the tactical networks Enhanced Position Location Reporting System [13] and Joint Tactical Radio System [22], implement deterministic allocation-based protocols, such as TDMA.

Related work on allocation-based protocols can be categorized into two different groups, namely, topology-dependent and topology-transparent, based on whether detailed network connectivity information is required. Existing topology-dependent approaches focus on finding a minimum-length conflict-free schedule based on the detailed network topology. This problem is proved to be NP-complete [1], [5]. In addition, recomputation and information exchanges are required to maintain accurate network topology information and to distribute the new schedules when the network topology changes. Thus, the robustness and effectiveness of these topology-dependent scheduling algorithms are undermined in large highly dynamic wireless mobile ad hoc networks.

To overcome the aforementioned limitations, topology-transparent scheduling algorithms have been proposed. Chalamatac and Farago [4] developed a topology-transparent algorithm that guarantees at least one collision-free time slot in each frame time, but the performance is even worse than the conventional TDMA in some cases. Ju and Li [12] proposed another algorithm to maximize the minimum guaranteed throughput. Other work on the design and analysis of topology-transparent scheduling algorithms can be found in [20], [24], and [26]. However, only unicast communication is considered. Cai *et al.* [3] proposed a broadcast scheduling algorithm, i.e., modified Galois field design (MGD), which sends the same message multiple times during one frame time to guarantee exactly one successful broadcast transmission per frame. The throughput is relatively small, since the maximum number of transmissions is one in a frame time. Sun *et al.* [25] designed an acknowledgement-based scheduling protocol for multicast and broadcast and obtained improved expected throughput. Unfortunately, the overhead introduced by acknowledgements dramatically increases with increasing frame length, thus degrading the performance, particularly when the total number of nodes and the maximum number of neighbors of a node are large. Farnoud and Valaee [8] applied positive orthogonal codes to design a reliable broadcast algorithm for safety messages in vehicular networks; however, it focused on a specific application and network topology (1-D roads). The algorithm requires each node to be location aware, and the performance metric is the success probability when the traffic load is not heavy. In [16], a probabilistic topology-transparent broadcast-scheduling algorithm was proposed, and fewer time slots are used to broadcast the same packet repeatedly. That is, it sacrifices some reliability to improve the average throughput. However, neither algorithm in [16] and [18] can guarantee at least one conflict-free transmission slot per frame time. Another topology-transparent broadcast-scheduling algorithm with guaranteed throughput was proposed in [15], in which erasure coding is applied to improve the throughput. However, many unassigned slots are not utilized at all.

The remainder of this paper is organized as follows. We introduce our system model in Section II. Our proposed algorithm is presented in detail in Sections III and IV. We propose to utilize the assigned slots efficiently and obtain the optimal data frame structure in Section III. In Section IV, we propose several methods to utilize the unassigned slots in a collision-free and traffic-adaptive manner to further improve the throughput of our algorithm. The performance of our algorithm is analytically studied in Section V. Simulations are conducted to evaluate the performance of our proposed algorithm in Section VI. The effect of inaccuracies in the estimation of the maximum node degree on the performance of the proposed algorithm is also investigated. Section VII concludes this paper with suggestions for further research.

II. System Model

A. Network Model

A mobile ad hoc network can be represented by a graph $G(V, E)$. V is the set of all network nodes, and E is the set of all edges. If node v is within the interference range of node u, an edge denoted by (u, v) is in E. We assume that if $(u, v) \in E$,

[2]Delay is defined as the time experienced by each packet between when it leaves the buffer and when it is successfully transmitted.

then $(v, u) \in E$. To simplify the calculation, but without loss of generality, we assume that the transmission range of a node is equal to its interference range. Let $N_1(u)$ and $N_2(u)$ denote the sets of one-hop neighbors and two-hop neighbors of node u, respectively. Thus, $N_1(u) \subset N_2(u)$. The degree of node u, i.e., $D(u)$, is defined as the number of one-hop neighbors of node u, i.e., $D(u) = |N_1(u)|$. $N_1(u, m)$, where $m = 1, 2, \ldots, D(u)$ indicates the mth neighbor of node u. The maximum degree $D_{\max}$ is defined as $D_{\max} = \max_{u \in V} D(u)$. We assume that $D_{\max}$ is much smaller than the number of nodes N and remains constant while the network topology changes. The maintenance and benefits of this kind of degree-bounded topology have been discussed in [10] and [11]. In practical networks, empirical data and statistics can be used to estimate $D_{\max}$. In addition, $D_{\max}$ may be pessimistically estimated based on network parameters and/or empirical data, thus ensuring that the actual number of interfering neighbors does not exceed the estimate.

We focus on TDMA networks. As discussed in previous work on topology-transparent scheduling, we assume that the transmission channel is error free and that a reception failure is only due to packet collisions. The transmission from node u to node v succeeds when 1) node v is not transmitting and 2) other nodes in v's interference range are not transmitting.

As in previous topology-transparent scheduling algorithms, we assume that the network is quasi-mobile. That is, the network connectivity remains the same during one frame time.

In broadcast scenarios, the acknowledgements from different intended receivers may collide at the transmitter, leading to the failure of the acknowledgement mechanism. This is the most challenging issue in broadcast communication and leads to the low throughput of previous topology-transparent broadcast scheduling algorithms.

B. Frame Structure

In our algorithm, time is divided into equal-sized frames. Each frame is further divided into three parts. The first two parts are intended for control packets and called control frames (CF1 and CF2). The last part is meant for data packets and called data frame (DF). In practice, the length of DF should be much larger than that of CF1 and CF2 to reduce the overhead.

Fig. 1 shows the detailed frame structure. CF1 and CF2 are each divided into q_c subframes, each of which consists of p_c synchronized minislots. The data frame is divided into q_d subframes, each of which consists of p_d synchronized minislots. We set $q_c \leq p_c$ and $q_d \leq p_d$, according to [3], [4], and [12]. Synchronization can be achieved by Global Positioning System.

Each node v is assigned a unique polynomial with degree k mod p_c, $f_v(x) = \sum_{i=0}^{k} a_i x^i (\mathrm{mod} p_c)$, where $v \in V$, as its time slot assignment function (TSAF). Node v transmits in time slot $f_v(i)$ in subframe i, where $i \in \{0, 1, 2, \ldots, q_d - 1\}$ for each data frame, and $i \in \{0, 1, 2, \ldots, q_c - 1\}$ for each control frame. We know $(f_v(i))$ as the time slot location vector (TSLV) for node v [12]. These slots are also known as the assigned slots of node v, whereas the other slots are known as unassigned slots. To ensure that the TSAFs of each node in the control and data frames are the same, we set $p_c = p_d = p$.

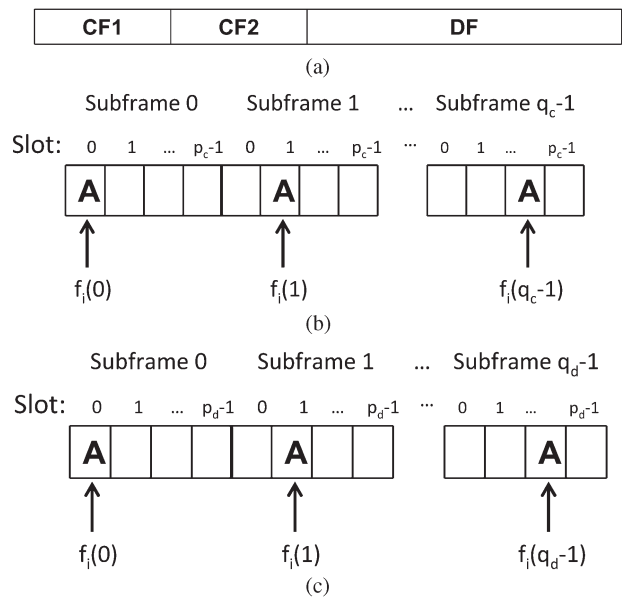


Fig. 1. Frame structure. (a) Structure of the whole frame. (b) Structure of control frame (CF1 and CF2). (c) Structure of data frame (DF).

For each node, CF1 is used to distribute its information and to collect the information of its neighbors. CF2 is used to distribute the information of its one-hop neighbors collected in CF1 and to collect the transmitted information, which is received by its neighbors in CF1. Thus, each node can collect the information of all its two-hop neighbors after CF1 and CF2 and transmit the data packets according to the collected information, as specified in the following section. The information of each node includes its own identity number (ID), TSAF, segment, and priority. The priority of each node is used to determine how to utilize the unassigned slots and will be discussed in Section IV. The segment of each node indicates the unassigned slots in the subset of subframes it can utilize. If the segment of a node is set to 0, it can possibly access the unassigned slots in all subframes. Our algorithm can provide heterogenous quality of service (QoS) for different nodes by properly configuring the respective segments. This will be discussed in detail in Section IV-C.

Consider a single-channel TDMA network $G(V, E)$ with N mobile nodes and the maximum node degree $D_{\max}$. To guarantee that each node receives the TSAFs of all its two-hop neighbors after CF2, two basic constraints must be satisfied, as discussed in [3], [4], and [12]. Therefore, we have

$$p_c^{k+1} \geq N \tag{1}$$

$$q_c \geq k D_{\max} + 1. \tag{2}$$

To minimize the overhead introduced by CF1 and CF2, we use the same method discussed in [3] to minimize the frame length of CF1 and CF2, stated as follows.

Let k_0 be the root of the following equation:

$$N^{\frac{1}{x+1}} = x D_{\max} + 1. \tag{3}$$

Then

$$\lceil k_0 \rceil \quad \text{or} \quad \lfloor k_0 \rfloor = \arg\min_k L_c(k) \tag{4}$$

where $L_c(k)$ is the frame length (in number of control slots) of each of CF1 and CF2.

Let p_1 be the minimum prime or prime power greater than or equal to $\lceil k_0 \rceil D_{\max} + 1$ and p_2 the minimum prime or prime power greater than or equal to $N^{1/\lfloor k_0 \rfloor + 1}$. Then, the minimum frame length of each of CF1 and CF2 is $\min\{p_1(\lceil k_0 \rceil D_{\max} + 1), p_2(\lfloor k_0 \rfloor D_{\max} + 1)\}$.

Based on the given discussions, we obtain the TSAFs and TSLVs of our proposed algorithm as follows.

1) Obtain k_0 and determine the optimal k that yields the minimum length of each of CF1 and CF2 according to (5) and (6).
2) Determine p_c and q_c from k. If $k = \lceil k_0 \rceil$, $p_c = p_1$, and $q_c = \lceil k_0 \rceil D_{\max} + 1$. If $k = \lfloor k_0 \rfloor$, $p_c = p_2$, and $q_c = \lfloor k_0 \rfloor D_{\max} + 1$. Hence, we have p_c^{k+1} available polynomials over $GF(p)$.
3) Distribute the polynomials to nodes. Each node i has a unique polynomial $f_i(x)$ as its TSAF, calculates its TSLV, and transmits the control packets according to its TSLV.

C. Overhead Analysis

Compared with the existing topology-transparent scheduling algorithms, some overhead, namely, CF1 and CF2, is introduced in the proposed algorithm to distribute and collect the information of two-hop neighbors of each node. Each node transmits its ID, priority, segment, and TSAF in CF1. Note that each TSAF is a degree-k polynomial, and each of its coefficients can be represented in one byte (each coefficient is no larger than 255). Hence, each TSAF can be represented in $(k+1)$ bytes. The ID, segment, and priority can be represented in two bytes each. In CF2, each node broadcasts all (up to $D_{\max}$) the information from its one-hop neighbors. Hence, we need at most $(k+7)D_{\max}$ bytes to broadcast the information. Each slot in CF1 is designed to accommodate the transmission of one packet of $(k+7)$ bytes and a guard time. Each slot in CF2 is designed to accommodate the transmission of one packet of $(k+7)D_{\max}$ bytes and a guard time. Recall that the frame length of CF1 and CF2 is $p_c q_c$ and that the frame length of DF is $p_d q_d$. To simplify the calculation, we neglect the guard time. The overhead introduced in the proposed algorithm is as follows:

$$\beta = \frac{(k+7)(D_{\max}+1)p_c q_c}{L p_d q_d + (k+7)(D_{\max}+1)p_c q_c} \tag{5}$$

where L is the length of the payload packet in bytes. Taking $N = 256$, $D_{\max} = 8$, and $L = 1024$ as an example, $k = 1$, $p_c = p_d = 17$, $q_c = 9$, $q_d = 17$,[3] and thus, $\beta = 3.6\%$.

As shown in (5), the overhead introduced in our algorithm is independent of the total number of nodes in the network, i.e., N, where $N \leq 2^{16}$. Thus, the overhead of our proposed algorithm does not increase with the network size.

[3]The derivation of q_d will be given in the following section.

III. Proposed Algorithm

A. Algorithm Description

The utilization of assigned slots of the proposed topology-transparent broadcast-scheduling algorithm is described as follows.

1) Initially, each node i calculates its TSLV based on its assigned TSAF, i.e., $f_i(x)$.
2) In CF1, each node i broadcasts its ID (identification number), segment, priority, and TSAF $f_i(x)$ according to its TSLV and stores the IDs, priorities, and TSAFs received from its neighbors, i.e., $f_j(x)$, where $j \in N_1(i)$.
3) In CF2, each node i broadcasts the $D(i)$ TSAFs, i.e., $f_j(x)$, where $j \in N_1(i)$, IDs, segments, and priorities received in CF1 as a packet according to its TSLV, and stores the IDs, segments, priorities, and TSAFs received from its neighbors. Hence, each node i knows the ID, segment, priority, and TSAF $f_j(x)$ of node j, where $j \in N_2(i)$.
4) At the beginning of each data frame, set a flag vector $Flag_i = (Flag_{i,N_1(i,1)}, \ldots, Flag_{i,N_1(i,D(i))})$ for each node i. $Flag_{i,N_1(i,m)} = 1$, where $m = 1, \ldots, D(i)$, indicates that the mth neighbor of node i has received the broadcast packet from node i. Otherwise, $Flag_{i,N_1(i,m)} = 0$. Initially, set $Flag_i$ to be a zero vector.
5) In subframe n, where $n = 0, 1, \ldots, q_d - 1$, each node i checks $Flag_i$. If $g(Flag_i) = D(i)$, where $g(Flag_i)$ is defined as the number of "1"s in a vector $Flag_i$, transmit the next packet queued in node i in this subframe and set $Flag_i$ to be a zero vector; otherwise, transmit the current packet in this subframe. Then, each node i checks its transmission slot in this subframe, i.e., $f_i(n)$. If $f_i(n) \notin \{f_j(n) : j \in N_1(N_1(i,m)) \cup (N_1(i,m) \setminus \{i\})\}$, where $m = 1, \ldots, D(i)$, set $Flag_{i,N_1(i,m)} = 1$; otherwise, do nothing.

B. Correctness of the Proposed Algorithm

First, due to the frame structure of CF1 and CF2, each node can successfully broadcast one packet to its neighbors during both CF1 and CF2. Hence, each node can successfully receive and store the TSAFs of its one-hop neighbor after CF1. Moreover, each node i knows the number of its neighbors, i.e., $D(i)$, after CF1. Then, each node broadcasts the TSAFs received in CF1 during CF2. As a result, each node knows the TSAFs of all of its two-hop neighbors. As we assume that the network is quasi-mobile, the TSAFs collected in CF1 and CF2 are accurate for the corresponding data frame. In the data frame, at the beginning of each subframe, each node i checks its flag vector $Flag_i$. $g(Flag_i) = D(i)$ implies that all the neighbors of node i have received the current packet successfully. If so, the next queued packet will be broadcast in this subframe. Otherwise, each node checks whether its mth neighbor, where $m = 1, 2, \ldots, D(i)$, can receive the current packet in this subframe by comparing its own TSAF with the TSAFs of its mth neighbor and the neighbors of its mth neighbor, which are collected in CF1 and CF2. If so, set $Flag_{i,N_1(i,m)} = 1$. This is repeated until $g(Flag_i) = D(i)$. Thus, according to our

algorithm, each node knows whether and when a broadcast packet is received by all its neighbors.

In the following, we show that the aforementioned utilization of the assigned slots can provide a guaranteed delay. In our algorithm, packets are repeatedly broadcasted in the assigned slots of each node in a more intelligent manner than the previous topology-transparent algorithms [3]. Based on the information collected in CF1 and CF2, each node knows when to stop the repeated broadcast of the current packet and to start the broadcast of the next packet, rather than simply broadcasting a packet repeatedly in the whole frame. In the worst case, a packet can be successfully broadcasted during one frame similar to the algorithm in [3]. The maximum delay is one frame. This leads to our conclusion that the aforementioned utilization of the assigned slots can provide a guaranteed delay. We will show that the utilization of the unassigned slots does not alter this conclusion later in Section IV.

C. Optimal Data Frame Structure

It has been shown that the optimal frame structure for broadcast and unicast traffic is different and that the throughput of topology-transparent broadcast algorithms is much smaller than that of topology-transparent unicast algorithms [2], [11]. This is mainly due to the fact that the acknowledgement mechanism cannot be efficiently implemented in broadcast communication, and to eliminate the acknowledgement, each node has to transmit the same packet repeatedly in one frame time. Based on the TSAFs of the two-hop neighbors of each node collected in CF1 and CF2, each node can transmit more than one broadcast packet in one frame. We have obtained the control frame structure in Section II-B. Now, we study the average throughput of assigned slots of our algorithm and obtain the optimal data frame structure as follows.

Recall that $q_d \leq p_d$. We have Theorem 1 as follows.

Theorem 1: Given p_d and the TSAFs of the two-hop neighbors of each node, the optimal throughput for broadcast traffic is achieved when $q_d = p_d$.

Proof: Suppose that node u broadcasts a packet to all its neighbors. Let A_i^{uv} be the event that node v, which is a neighbor of node u, has received the packet from node u in subframe $i-1$ for the first time, where $i = 1, 2, \ldots, D_{\max}+1$. We assume the worst case here. There are up to $D_{\max}$ interfering nodes, and all are transmitting.

It is proved in [11] that the maximum degree of the polynomials, i.e., k, is one for most cases. Thus, without loss of generality, we use $k = 1$ in the following analysis for simplicity. That is, there are p_d^2 TSLVs in total. Let N^i be the number of ways for a given TSLV of node u, i.e., $TSLV_u$, to select $D_{\max}$ other TSLVs, the union of which intersects $TSLV_u$ in the first $i-1$ subframes (subframes 0 to $(i-2)$) and does not intersect $TSLV_u$ in subframe $i-1$. There are $\binom{p_d^2-1}{D_{\max}}$ ways to select $D_{\max}$ TSLVs from the remaining p_d^2-1 TSLVs. Thus, the probability that A_i^{uv} happens, where $i = 1, 2, \ldots, D_{\max}+1$, is as follows:

$$\Pr\left(A_i^{uv}\right) = \frac{N^i}{\binom{p_d^2-1}{D_{\max}}}. \tag{6}$$

Consider the TSAF of node u, i.e., $TSAF_u$. We categorize the remaining p_d^2-1 TSAFs into p_d+1 different subsets F_i $(i = 0, 1, \ldots, p_d)$ according to their coincidences with $TSAF_u$. We define the coincidence of any two polynomials as the root of the difference of these two polynomials. That is, if $f_u(j) - f_v(j) = 0$, j is the coincidence of $f_u(x)$ and $f_v(x)$. The TSAFs in F_i $(i = 0, 1, \ldots, p_d-1)$ have the coincidence i with $TSAF_u$, and the TSAFs in F_{p_d} have no coincidence with $TSAF_u$. Note that a TSAF over $GF(p_d)$ is uniformly distributed over $\{0, 1, 2, \ldots, p_d-1\}$ [6]. Thus, $|F_i| = p_d - 1$, where $i = 0, 1, \ldots, p_d$. The detailed verification is as follows. Consider an arbitrary degree-1 polynomial $f(x) = ax + b \pmod{p_d}$, where a, $b \in \{0, 1, \ldots, p_d-1\}$. Keeping the slope of $f(x)$, i.e., a, invariant and varying b, we get a sequence of polynomials $g_i(x) = ax + b_i \pmod{p_d}$, where $b_i \in \{0, \ldots, b-1, b+1, \ldots, p_d-1\}$ that have no coincidences with $f(x)$. The number of terms in the sequence of polynomials is $p_d - 1$. That is, $|F_{p_d}| = p_d - 1$. Similarly, consider $f(x) = ax + b \pmod{p_d}$ and an arbitrary integer x_0, where a, b, $x_0 \in \{0, 1, \ldots, p_d-1\}$. Fixing the point $(x_0, f(x_0))$ and varying a_i, where $a_i \in \{0, \ldots, a-1, a+1, \ldots, p_d-1\}$, we obtain a sequence of lines (polynomials) passing the point $(x_0, f(x_0))$ besides $f(x)$, the number of which is $p_d - 1$. Thus, $|F_i| = p_d - 1$, where $i = 0, 1, \ldots, p_d - 1$.

Given i (where $i = 0, 1, \ldots, D_{\max}$), we classify TSAFs other than $TSAF_u$ into two different groups.

- *Group 1*: The number of TSAFs that have the coincidence i with $TSAF_u$ is $p_d - 1$.
- *Group 2*: The number of TSAFs that do not have the coincidence i with $TSAF_u$ is $p_d(p_d-1)$.

Let C_j (where $j = 0, 1, \ldots, i-2$) be the set of events in which none of the chosen $D_{\max}$ TSAFs from Group 2 has the coincidence j with $TSAF_u$. Note that the number of TSAFs that have the coincidence j (where $j = 0, 1, \ldots, i-2$) with $TSAF_u$ is $p_d - 1$, and we choose $D_{\max}$ TSAFs from Group 2. Thus, the cardinality of the intersection of any m sets from D_j, where $j = 0, 1, \ldots, i-2$, is $\binom{p_d^2-1-(m+1)(p_d-1)}{D_{\max}}$. N^i is equal to the cardinality of the complementary set of $\bigcup_{j=0}^{i-2} C_j$, which is denoted by $\complement$. Note that there are $\binom{p_d(p_d-1)}{D_{\max}}$ ways to select $D_{\max}$ codewords from Group 2. Thus

$$\begin{aligned} N^i &= |\complement| \\ &= \binom{p_d(p_d-1)}{D_{\max}} - \left|\bigcup_{j=0}^{i-2} C_j\right|. \end{aligned} \tag{7}$$

Applying the inclusion–exclusion principle, we can obtain N^i, $i = 2, \ldots, D_{\max}+1$, as (8) and $N^1 = \binom{p_d(p_d-1)}{D_{\max}}$.

$$\begin{aligned} N^i = \binom{p_d(p_d-1)}{D_{\max}} &- \sum_{m=1}^{i-1} (-1)^{m-1} \binom{i-1}{m} \\ &\times \binom{p_d^2-1-(m+1)(p_d-1)}{D_{\max}}. \end{aligned} \tag{8}$$

We assume that all (up to $D_{\max}$) transmissions corresponding to a single broadcast communication are independent. Let μ

be the number of times a sender broadcasts the same message so that all its (up to $D_{\max}$) neighbors receive it successfully. Since nodes u and v are arbitrarily chosen, deleting the superscript uv for clarity, we have:

$$E[\mu] = \Pr(A_1)^{D_{\max}} + \sum_{i=2}^{D_{\max}+1} i \sum_{r=1}^{D_{\max}} \binom{D_{\max}}{r} \Pr(A_i)^r \times \Pr\left(\bigcup_{j=1}^{i-1} A_j\right)^{D_{\max}-r}$$

$$= \Pr(A_1)^{D_{\max}} + \sum_{i=2}^{D_{\max}+1} i \sum_{r=1}^{D_{\max}} \binom{D_{\max}}{r} \Pr(A_i)^r \times \left[\sum_{j=1}^{i-1} \Pr(A_j)\right]^{D_{\max}-r}. \quad (9)$$

The expected number of successful broadcast messages delivered by a node per frame is:

$$T = \frac{q_d}{E[\mu]}. \quad (10)$$

The expected throughput of the proposed algorithm can be computed as follows:

$$E[G_{\text{assigned}}] = \frac{T}{p_d q_d} = \frac{1}{p_d E[\mu]}. \quad (11)$$

Thus, the throughput of broadcast traffic does not depend on the number of subframes. Noting that the overhead decreases when q_d increases, as shown in (5), we prove Theorem 1. ■

IV. Utilization of Unassigned Slots

Note that the codes assigned to each node in our algorithm are highly sparse. That is, each node only chooses q_d out of $p_d q_d$ time slots to transmit within one frame time to support guaranteed throughput. The remaining $(p_d - 1)q_d$ time slots are not utilized at all, resulting in a relatively low average throughput. An RTS (Request to Send)-based and CTS (Clear to Send)-based ALOHA-type method has been proposed in [18] and [28] to utilize the unassigned slots to improve the throughput. A collision-free utilization of the unassigned slots was proposed in [14]. However, there are two main drawbacks. First, all these algorithms are designed only for unicast communications. Second, all these algorithms are designed under the heavy-traffic condition. Thus, none of these algorithms are adaptive to the network traffic. To overcome these problems, we develop several methods based on the collected codes of two-hop neighbors to utilize the unassigned slots in a collision-free manner, thus allowing nodes to access both the assigned and unassigned slots efficiently and improving the average throughput. More importantly, our methods are traffic adaptive.

In our algorithm, each node learns the indexes of the time slots that can possibly be utilized according to the collected TSAFs of its two-hop neighbors after CF1 and CF2. Let $PU(u)$ be the set of time slots that can possibly be utilized by node u (where $u = 1, 2, \ldots, N$) in the current frame. Time slot j in subframe i is in $PU(u)$ if and only if $j \notin \bigcup_{m \in \{u\} \cup N_2(u)} \{f_m(i)\}$. That is, $PU(u) = \{(i,j) | 0 \le i \le q_d - 1, 0 \le j \le p_d - 1, j \notin \bigcup_{m \in \{u\} \cup N_2(u)} \{f_m(i)\}\}$. The detailed process is described in Algorithm 1. Meanwhile, each node collects the priorities of its two-hop neighbors after CF1 and CF2. The nodes with the highest priority within their two-hop neighbors are granted to utilize the time slots in their PUs.

Algorithm 1 $FindAvailableUnassignedSlots(u)$

Input: The set of two-hop neighbors of node u, $N_2(u)$; TSAFs of the nodes in $N_2(u)$, $\{f_n(x) | n \in N_2(u)\}$.
Output: The set of unassigned slots that can be possibly utilized by node u, $PU(u)$.

```
1: Initialize PU(u) = ∅
2: for Subframe = 0 to q_d − 1 do
3:     for Slot = 0 to p_d − 1 do
4:         for n = 1 to |N_2(u)| do
5:             if Slot ∉ {f_n(Subframe)} ∪ {f_u(Subframe)} then
6:                 PU(u) ← PU(u) ∪ {Slot + Subframe × p_d}
7:             end if
8:         end for
9:     end for
10: end for
11: return PU(u)
```

Note that the guaranteed minimum throughput can still be achieved, even in a high-traffic-load environment and in dense networks. This is because the transmissions in the time slots belonging to $PU(u)$ do not interfere any transmissions of the two-hop neighbors of u in their assigned slots. Thus, we conclude that, for our algorithm, the transmissions in the unassigned slots do not affect the transmissions in the assigned slots of other nodes, implying that the use of the unassigned slots does not lead to a violation of the provisioning of a guaranteed delay. Thus, we can conclude that our algorithm can provide a guaranteed throughput and delay.

If node u is granted access to the time slots in $PU(u)$, the transmissions in these slots are collision free, since no nodes among its two-hop neighbors transmit in these slots. In the following, we introduce three ways to utilize the unassigned time slots by using different approaches to generate priorities, namely, uniform utilization (UU), traffic-based utilization (TU), and heterogeneous TU (HTU). UU is designed under the heavy-traffic condition. However, TU and HTU are traffic adaptive.

In the beginning of each frame, each node broadcasts and collects information of its two-hop neighbors, as discussed in Sections II and III. The analysis of the overhead is given in Section II-C. In the following, we show that our algorithm has a polynomial time complexity. As shown in Algorithm I, in all three loops, there are $p_d q_d |N_2(u)|$ iterations (lines 5–7 in Algorithm I), where $|N_2(u)|$ is the cardinality of $N_2(u)$.

For each iteration, a node determines whether the current time slot is an assigned slot of another node of its two-hop neighbors. As discussed in Sections II and III, $p_d = q_d = \max(N^{1/k+1}, D_{\max} + 1)$. We can obtain that $p_d = q_d = O(D_{\max} \log N / \log D_{\max})$. The detailed proof can be found in [4]. Moreover, since each node has at most $D_{\max}^2$ two-hop neighbors,[4] $|N_2(u)| \leq D_{\max}^2$. Thus, the number of iterations in all three loops in Algorithm I is $p_d q_d |N_2(u)| = O(D_{\max}^4 \log^2 N / \log^2 D_{\max})$. Thus, the polynomial running time of our algorithm follows, due to the facts that the iteration can be completed by doing elementary arithmetic operations over Galois field $GF(p_d)$ and computed in polynomial time and that the number of iterations executed by the algorithm is polynomial in N and $D_{\max}$.

A. Uniform Utilization of Unassigned Slots

Let M be a permutation of the vector $(1, 2, \ldots, N)$, which is known by all the N nodes *a priori*. The virtual ID (VID) of node i, where $i = 1, 2, \ldots, N$, in frame t can be computed as follows:

$$VID(i, t) = \mathrm{mod}\left((M(i) + t), N\right). \tag{12}$$

After CF1 and CF2, each node collects the IDs and calculates the VIDs of its two-hop neighbors.[5] The nodes with the highest VID within their two-hop neighbors are granted access to the time slots in their PUs. It is obvious that the priority (VID) of each node is unique in every frame and uniformly distributed as time evolves. Thus, each node utilizes the unassigned time slots in a uniform way.

B. Traffic-based Utilization of Unassigned Slots

In the previous discussion, we assume that all homogenous nodes are backlogged. In reality, the traffic load at each node may vary over time. Therefore, we propose to use the TU of unassigned slots to investigate whether our algorithm is adaptive to varying traffic load at each node. We simply use the number of buffered packets in each node to represent its traffic load, irrespective of the underlying traffic model. In Section VI-B, we use a specific traffic model and perform simulation to study the performance of our algorithm with TU. The impact of traffic patterns on the performance of scheduling algorithms can be found in [23] and references therein. Each node can be granted access to the unassigned time slots according to its number of buffered packets. Intuitively, the nodes with more packets queued in the buffer should be provided more transmission opportunities. In the TU of unassigned slots, we use the number of buffered packets in each node as its priority. The nodes with the highest priorities (numbers of packets) within their two-hop neighbors are granted access to the time slots in their PUs. If the priorities of multiple nodes within two-hop neighbors are equal, the node with the highest VID is granted access to the time slots in its PU.

[4]The number of two-hop neighbors is much less than $D_{\max}^2$.

[5]No priority information is needed to broadcast in CF1 and CF2 in UU, since VID, which can be computed based on ID, is used as priority.

C. Heterogeneous Traffic-Based Utilization of Unassigned Slots

HTU of unassigned slots can be implemented in a heterogeneous network, in which different classes of nodes have different quality of service (QoS) requirements. Without loss of generality, we assume that two classes of nodes exist in the network, namely, Class V_1 with a higher QoS requirement and Class V_2 with a lower QoS requirement. The calculation of the priority of each node is the same as that in TU. The difference is as follows.

All time slots are equally divided into two subsets, namely, Subset 1 and Subset 2. Subset 1 includes the time slots in the subframes with odd indexes, and Subset 2 includes the time slots in the subframes with even indexes. The segments of the nodes in Class V_1 are set to 0, implying that each node in V_1 can utilize the unassigned slots in both subsets if it is granted access. However, the segments of the nodes in Class V_2 are set to 1, implying that each node in V_1 can utilize the unassigned slots in Subset 1 if it is granted access.

Each node generates and broadcasts its priority in CF1 and CF2. After CF1 and CF2, each node collects the IDs, segments, and priorities of its two-hop neighbors. An arbitrary node u in Class 1 is granted access to the time slots in $PU(u)$, if its priority is the highest within its two-hop neighbors. If not, it can also utilize the time slots in $PU(u) \cap$ Subset 2 as long as its priority is the highest within its two-hop neighbors with $segment = 0$. An arbitrary node v in Class 2 with the highest priority within its two-hop neighbors is granted to utilize the time slots in $PU(v) \cap$ Subset 1. If the priorities of multiple nodes within two-hop neighbors are equal, the node with the highest VID is granted access to the time slots. This guarantees that the transmissions in the time slots in the PUs of the granted nodes are all collision free. Thus, nodes are granted access to the unassigned time slots according to the class to which belong in a heterogeneous way. Although we only consider two classes of nodes with different QoS requirements in the aforementioned discussion, we can easily generalize it to more classes of nodes by dividing the time slots in one frame into more subsets and setting the segment of each node properly.

V. Analytical Results

A. Throughput Analysis

We study the average saturated throughput of the proposed algorithm. The average saturated throughput is defined as the average number of packets successfully received per time slot per node, under a heavy-traffic condition with all nodes backlogged. In the proof of Theorem 1, we obtain the average saturated throughput of the assigned slots for broadcast traffic, as shown in (9) and (11). Intuitively, the average throughput of unassigned slots depends on the number of nodes in the contention set. A larger number of nodes in the contention set leads to more contending nodes, less unassigned slots to be utilized, and, thus, smaller average throughput. In the following, we analyze the saturated throughput of the unassigned slots.

Consider an arbitrary node u with n two-hop neighbors. Node u cannot utilize the slots, which are the assigned slots of its n two-hop neighbors and node u itself. Recall that the

maximum degree of the polynomials, i.e., k, is one for most cases [11]. Thus, without loss of generality, we use $k = 1$ in the following analysis for simplicity. That is, there are p_d^2 TSLVs in total. Consider an arbitrary subframe j, where $0 \le j \le q_d - 1$. Let M^l denote the number of ways to select $n+1$ out of p_d^2 TSLVs such that the assigned slots of these $n+1$ nodes are exactly l specific slots in subframe j, where $l = 1, 2, \ldots, p_d$. Note that there are $\binom{p_d^2}{n+1}$ ways to select $n+1$ out of p_d^2 TSLVs and $\binom{p_d}{l}$ ways to select l out of p_d slots in one subframe. Thus, the probability that $p_d - l$ slots can be utilized by node u, i.e., $P(l)$, is as follows:

$$P(l) = \frac{M^l \binom{p_d}{l}}{\binom{p_d^2}{n+1}}. \tag{13}$$

As discussed in Section IV, the priority of each node changes in a uniform manner as time evolves. Thus, a node with n contention nodes has the highest priority and can utilize the unassigned slots once every $n+1$ frames. We obtain the average throughput of unassigned slots, conditioned on n, i.e., $E[G_{\text{unassigned}}|n]$, as follows:

$$E[G_{\text{unassigned}}|n] = \frac{\sum_{l=1}^{p}(p-l)P(l)}{(n+1)p}. \tag{14}$$

Thus, we have:

$$E[G_{\text{unassigned}}] = \sum_{m=1}^{N-1} E[G_{\text{unassigned}}|n=m]\,\Pr(n=m). \tag{15}$$

However, we can hardly obtain the distribution of n. We approximate the average throughput of unassigned slots as (16) (which is found to be quite accurate when compared with the simulation results as exhibited in Section VI). Thus,

$$E[G_{\text{unassigned}}] \approx E[G_{\text{unassigned}}|\bar{n}] \tag{16}$$

where $\bar{n}$ is the average number of two-hop neighbors of a node. We will discuss how to calculate $\bar{n}$ in the following section.

The calculation of M^l, where $l = 1, 2, \ldots, p_d$, is given in Appendix.

Thus, taking into consideration the overhead discussed in (5), the average effective saturated throughput, i.e., $E[G_e]$, is given as follows:

$$E[G_e] = \left(E[G_{\text{assigned}}] + E[G_{\text{unassigned}}]\right)(1-\beta) \tag{17}$$

where β, $E[G_{\text{assigned}}]$, and $E[G_{\text{unassigned}}]$ are given in (5), (11), and (16), respectively.

VI. Performance Evaluation

Here, we compare the performance of our proposed scheduling algorithm with the MGD algorithm proposed by Cai *et al.* [3], the algorithm in [25] (referred to as Sun's algorithm), and the erasure-coding-based broadcast algorithm in [15] (referred to as EC algorithm). We also compare our algorithm with the topology-dependent (coloring) scheduling algorithm proposed in [19].[6] We refer to it as the coloring algorithm. This coloring algorithm is a centralized algorithm. Several distributed versions of this algorithm have been proposed and shown to have similar performance to this centralized algorithms [21]. However, it takes several minutes to obtain the schedules [9], [21], which is too long in mobile networks, as the topology would have changed before the computation ends. Thus, the centralized coloring algorithm in [19] is not really applicable to our scenario; however, it may be considered as a benchmark of topology-dependent (coloring) algorithms. Moreover, we conduct all simulations using MATLAB.

A. Simulation Setup

We conduct computer simulation on the geometric model for the average performance of our algorithm. In the geometric model, all N nodes are uniformly distributed in a region of 1000 m $\times$ 1000 m initially. Each node moves according to the Gauss–Markov mobility model, which has been shown to be more realistic than the widely used random waypoint model [2]. In the Gauss–Markov mobility model, time is discrete so that the movement of a node from a time instance to the next time instance is determined by parameters θ and v. The tuning parameter θ is used to present different levels of randomness in the Gauss–Markov model. We set $\theta = 0.5$ (where Brownian motion is obtained by setting θ to zero, and the linear motion is obtained by setting θ to 1). The speed v follows a Gaussian distribution, the mean and standard deviation of which are 0.9 ms^{-1} and 0.5 ms^{-1} [29]. The detailed description of the Gauss–Markov mobility model can be found in [2] and [29]. Moreover, we consider that the 1000 m $*$ 1000 m region wraps around like a torus; hence, there are no edge effects. Given $D_{\max}$, we set the interference range of each node R_I such that the probability that the number of interfering neighbors of an arbitrary node exceeding $D_{\max}$, which is $\sum_{i=D_{\max}+1}^{N-1}\binom{N-1}{i}(\pi R_I^2/A)^i(1-(\pi R_I^2/A))^{N-1-i}$, is smaller than 0.05. For example, $R_I = 87$ m if $(N, D_{\max}) = (256, 10)$. If there exist more than $D_{\max}$ nodes in the interference range of a node, the nodes other than $D_{\max}$ randomly selected interfering nodes are assumed to be noninterfering. This guarantees that the maximum node degree is $D_{\max}$. We can calculate $\bar{n}$ as $\bar{n} = N\pi(\alpha R_I)^2/A$. In the simulations, we find that a good estimation of n can be achieved by setting $\alpha = 1.4$.

We apply the optimal k, p_c, q_c, and $p_d = q_d = p$ derived in Sections II and III. The packet length of the payload is set to 1024 bytes, i.e., $L = 1024$. The overhead is calculated according to (5). For each result, we run each simulation for 100 frame times, unless stated otherwise. The 95% confidence interval is computed for each data point, as shown in the figures.

Unlike our proposed algorithm and the other algorithms to which we refer, Sun's algorithm is acknowledgement based. To make a fair comparison, we must consider the overhead introduced by employing acknowledgements in Sun's algorithm. We adopt the same parameters used in [25]. In Sun's algorithm, a time slot is divided into two segments, namely,

[6]The best heuristic, i.e., Progressive Minimum Neighbors First, has been applied in coloring.

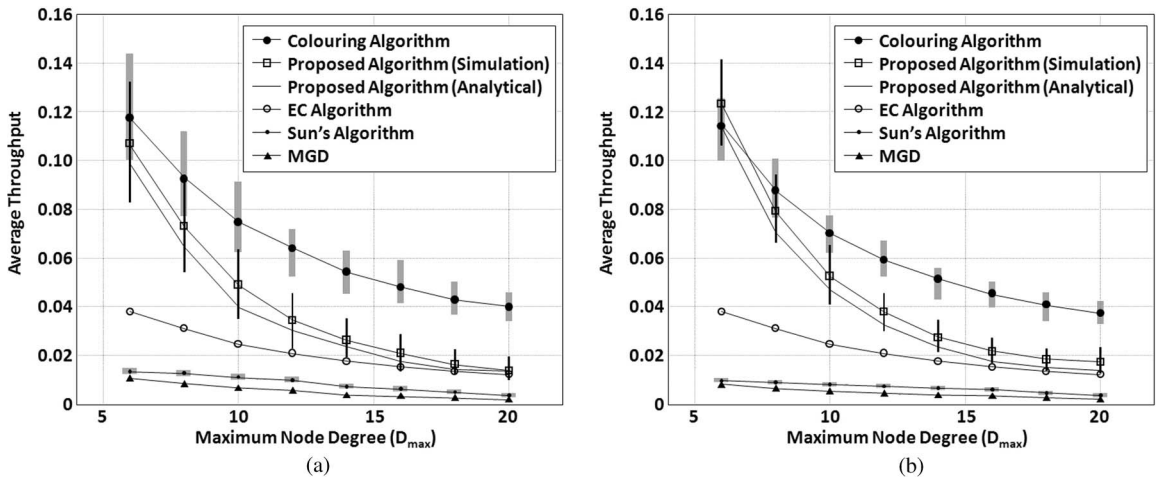


Fig. 2. Effect of $D_{\max}$ on the performance. (a) $N = 128$. (b) $N = 256$.

the data segment and the acknowledgement segment. The acknowledgement segment is further divided into pq minislots for acknowledgement transmission. Each of these minislots is used to accommodate the transmission of a four-byte acknowledgement. Thus, the overhead introduced by employing acknowledgement is $4p_cq_c/L + 4p_cq_c$, where L is the packet length of the payload. In the simulations, we also consider the overhead of our algorithm due to CF1 and CF2. The overhead introduced by our algorithm is discussed in Section II-C and can be expressed as (5).

B. Simulation Results

We first evaluate the average saturated throughput of our algorithm and study the other two traffic-adaptive utilizations of unassigned slots later, namely, TU and HTU.

1) Effect of $D_{\max}$ on the Performance: Given that $N = 128$ and $N = 256$, we investigate the performance of our algorithm with different $D_{\max}$ settings from 6 to 20. A larger value of $D_{\max}$ indicates that the network is denser and that there are more possible conflicts. As shown in Fig. 2, we can see that the average throughput of our algorithm is better than that of other topology-transparent scheduling algorithms. Moreover, the performance of our algorithm is comparable with that of the coloring algorithm, particularly when $D_{\max}$ is not large because there exists only one node transmitting among its two-hop neighbors in the coloring algorithm to avoid the costly recomputation of the updated schedules when the destinations of nodes change as time evolves. However, in our algorithm, each node can dynamically utilize its unassigned slots as long as it does not interfere with or is not interfered by other nodes in its contention set, regardless of the changes of the destinations of the nodes. We can also observe that the superiority of our algorithm, compared with the EC algorithm, becomes smaller with increasing $D_{\max}$. This is because the performance of the EC algorithm only depends on $D_{\max}$. However, the performance of our algorithm depends on the number of two-hop neighbors. Thus, the performance of the EC algorithm decreases slower than that of our algorithm with increasing $D_{\max}$. Moreover, we can observe that the performance of our proposed algorithm almost remains the same with increasing N, implying that the performance of our algorithm is insensitive to the number of nodes in the network.

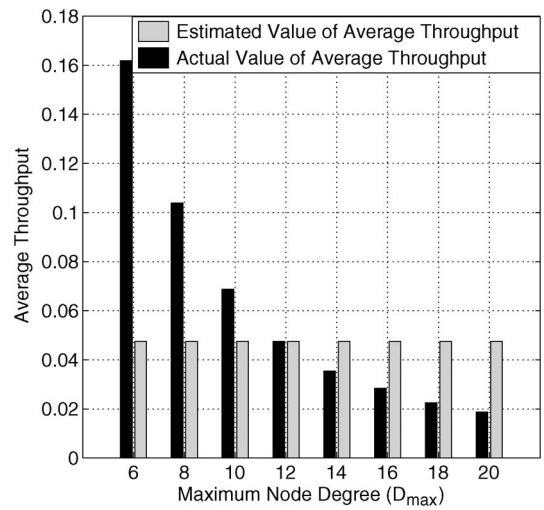


Fig. 3. Effect of inaccuracies in the estimation of $D_{\max}$ on the throughput.

It is shown in Fig. 2 that the simulation results match our analytical results closely. We can also observe that the throughput obtained by simulation is slightly higher than that obtained analytically. This is due to the fact that the number of neighbors for any node is not larger than $D_{\max}$ so that the throughput obtained analytically, assuming that the number of neighbors is $D_{\max}$, is indeed a lower bound of the average throughput.

2) Effect of Inaccuracies in the Estimation of $D_{\max}$ on the Performance: In Fig. 3, we investigate the effect of inaccuracies in the estimation of $D_{\max}$ on the throughput. Given that the network parameters $(N, D_{\max})$ is (256, 12), we then find $p = p_d = q_d = 17$, according to the discussion in Section III. By varying the actual maximum node degree from 6 to 20, we can observe in Fig. 3 that the actual value of the average throughput is smaller (larger) than the estimated value of the average throughput when the actual maximum node degree is larger (smaller) than the design value of $D_{\max}$. The actual value of the average throughput decreases with increasing actual maximum node degree. The difference between the actual and estimated values of the average throughput is relatively small (about 25% derivation) when the actual maximum node degree is close

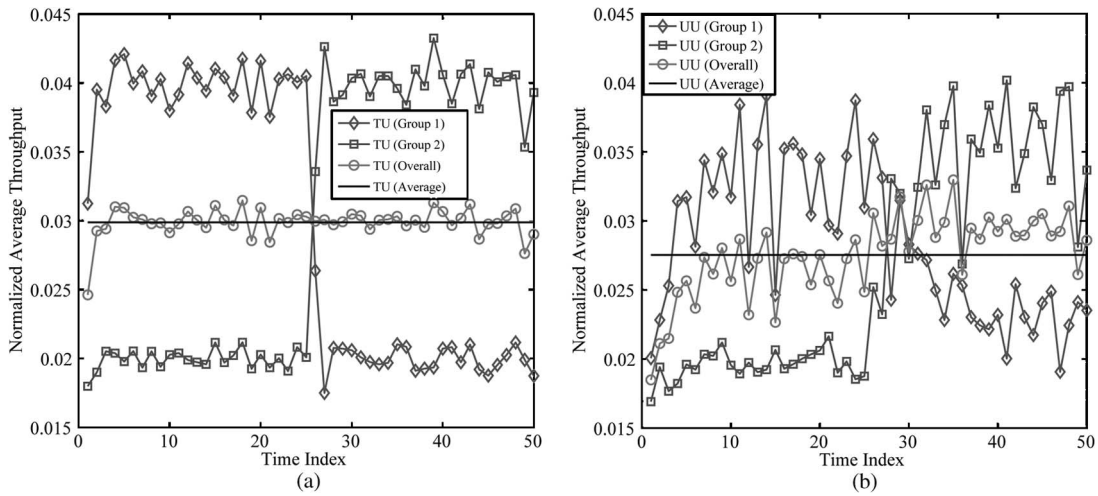


Fig. 4. Comparison of throughput of our algorithm with TU and UU. (a) Average throughput of our algorithm with TU. (b) Average throughput of our algorithm with UU.

to the design value ($D_{\max} = 12$). As the difference between the actual $D_{\max}$ and the design value of $D_{\max}$ increases, the penalty on the network performance increases (when the actual $D_{\max}$ is larger than the design value). When the actual value of $D_{\max}$ is 18 (50% larger than the design value), the actual value of the average throughput is about half as large as the estimated value. Hence, our proposed algorithm is topology adaptive in the sense that the average throughput gracefully degrades when the actual value of $D_{\max}$ is larger than the estimated value. In other words, the network throughput is lower than that obtained analytically when $D_{\max}$ is underestimated. When $D_{\max}$ is accurately estimated, the throughput is the same as the analytical results obtained based on the designed parameters. When $D_{\max}$ is overestimated, the throughput of our algorithm is even better than the analytical results obtained based on the designed parameters. This validates our statement in Section II-A that $D_{\max}$ should always be pessimistically estimated to ensure that the actual number of interfering neighbors does not exceed the estimate.

3) Performance of Our Algorithm With TU: Given that $N = 128$ and $D_{\max} = 8$, we evaluate the performance of our algorithm with TU. The frame structure can be determined according to the network parameters as $p_c = 13$, $q_c = 9$, and $p = p_d = q_d = 13$. The simulation lasts 1000 frame times. It is assumed that two groups of nodes exist in the network, namely, Group 1 and Group 2. The number of nodes in each group is $N/2 = 64$. The packet arrivals of each node follow different Poisson distributions as follows. The arrival rate of each node in Group 1 is 2λ from frame 1 to frame 500 and λ from frame 501 to frame 1000. The arrival rate of each node in Group 2 is λ from frame 1 to frame 500 and 2λ from frame 501 to frame 1000. In our simulation, $\lambda = 0.02$ packets per time slot. In Fig. 4, each point with time index i stands for the throughput averaged from frame $1 + 20(i-1)$ to frame $20i$. We can observe that with TU, the throughput of nodes in Group 1 (Group 2) is adaptive to their traffic arrivals, and the overall throughput is stable. Conversely, the throughput of nodes with UU is not adaptive to their arrivals at all. Moreover, the average throughput of our algorithm with TU is better than that of UU.

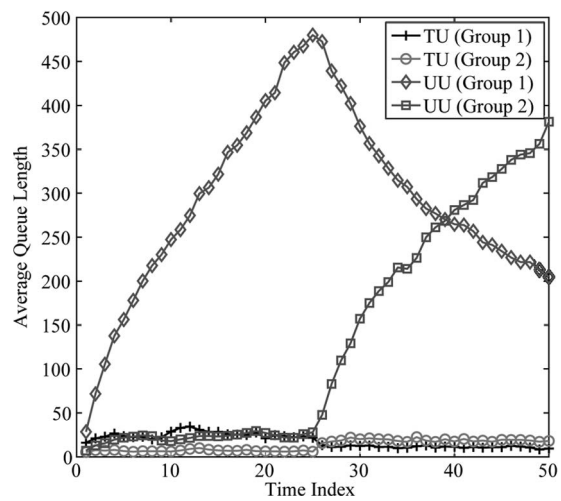


Fig. 5. Comparison of average queue length of our algorithm with TU and UU.

The average queue length in our algorithm with TU and UU is shown in Fig. 5.[7] We can see that the average queue length in our algorithm with TU is very small and much smaller than those of UU, which implies that our algorithm with TU is traffic adaptive. This can be explained by the fact that with TU, the nodes with more packets have larger probabilities to utilize the unassigned slots.

4) Performance of Our Algorithm With HTU: HTU of unassigned slots can be applied for the provisioning of different QoS requirements for different classes of nodes in heterogeneous networks. Suppose that $D_{\max} = 8$ and $N = 128$ nodes are equally divided to two classes, namely, Class V_1 with a higher QoS requirement and Class V_2 with a lower QoS requirement. The arrival rate of each node λ varies from 0.01 to 0.20. For each data point plotted, we perform 1000 simulation runs. In Fig. 6(a), we can observe that the throughputs of the nodes in two classes are almost the same when λ is small. However, the throughputs of the nodes in Class V_1 are larger than those of

[7]The average queue length is defined as the expected number of waiting packets of all nodes in the system.

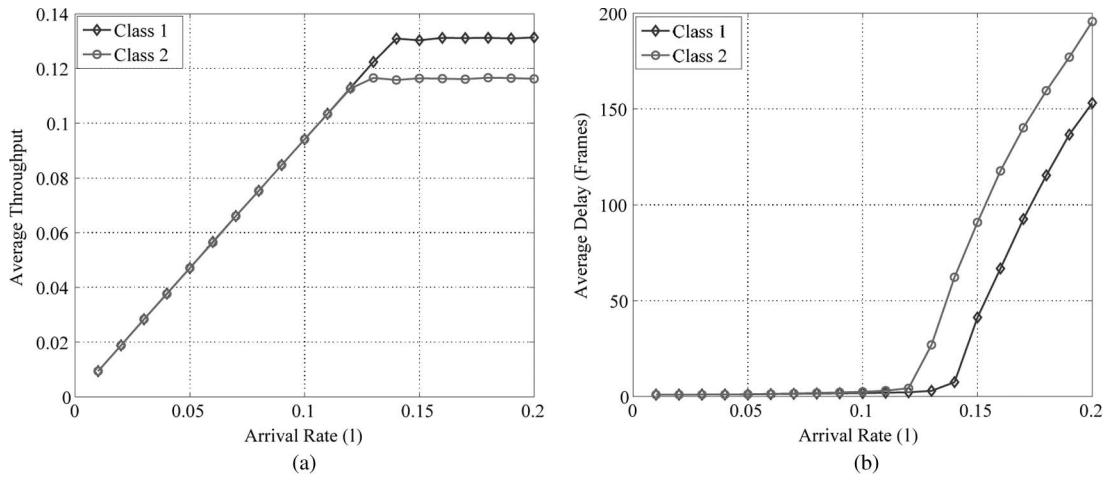


Fig. 6. Performance of our algorithm with HTU. (a) Average throughput. (b) Average delay.

the nodes in Class V_2 with increasing λ. Moreover, as shown in Fig. 6(b), the average delays of the nodes in Class V_1 are smaller than those of the nodes in Class V_2, even when λ is in the region in which the average throughputs of the nodes in two class are the same.[8] Thus, our algorithm with HTU can provide heterogeneous performance for different classes of nodes.

VII. Conclusion and Future Research

In this paper, we have proposed an efficient distributed topology-transparent scheduling algorithm for broadcasting in wireless ad hoc networks. First, instead of repeatedly transmitting the same packet during one frame time, each node collects transmission codes of its two-hop neighbors with little overhead and can transmit more than one packet during one frame time. Second, we propose several methods to utilize the unassigned slots efficiently in a collision-free and traffic-adaptive manner. Hence, our proposed algorithm achieves a much better average throughput. We have conducted analytical and simulation studies to show that the performance of our proposed algorithm outperforms other existing topology-transparent algorithms dramatically.

In this paper, we assumed that the transmission channel is error free. However, channel error is an important characteristic of wireless network, and we plan to study topology-transparent algorithms accounting for channel errors in the future.

Appendix
Calculation of M^l

Given an arbitrary subframe j, we categorize p_d^2 TSAFs into p_d different subsets FF_h $(h = 0, 1, \ldots, p_d - 1)$ according to their function values in subframe j. The function values of TSAFs in FF_h $(h = 0, 1, \ldots, p_d - 1)$ are h. Note that a TSAF over $GF(p_d)$ is uniformly distributed over $\{0, 1, 2, \ldots, p_d - 1\}$ [6]. Thus, $|FF_h| = p_d$, where $h = 0, 1, \ldots, p_d - 1$. The detailed verification is similar to that in the proof of Theorem 1.

[8]Here, we consider the average queueing delay at a node. This is different from the frame latency discussed in Sections III and IV. The queueing delay is defined as the waiting time experienced by each packet before it leaves the buffer.

Let A_{p_i} (where $i = 1, 2, \ldots, l$) be the set of events in which none of the chosen $n + 1$ TSAFs has the function value p_i in subframe j, where $0 \le p_i \le p_d - 1$. Note that the number of TSAFs that have the function value p_i (where $i = 1, 2, \ldots, l$) is lp_d, and we choose $n + 1$ TSAFs from those TSAFs, the function values of which in subframe j are not p_i (where $i = 1, 2, \ldots, l$). Thus, the cardinality of the intersection of any m sets from A_{p_i}, where $i = 1, 2, \ldots, l$, is $\binom{(l-m)p_d}{n+1}$. M^l is equal to the cardinality of the complementary set of $\bigcup_{i=1}^{l} A_{p_i}$. There are $\binom{lp_d}{n+1}$ ways to select $n + 1$ codewords from those TSAFs, the function values of which in subframe j are not p_i (where $i = 1, 2, \ldots, l$). We define a function $F(x, y)$ equal to $\binom{x}{y}$ if $x \ge y$ and zero, otherwise. Thus

$$M^l = F(lp_d, n + 1) - \left| \bigcup_{i=1}^{l} A_{p_i} \right| \tag{18}$$

where $l = 1, 2, \ldots, p_d$.

Applying the inclusion–exclusion principle, we can obtain the following results:

$$M^l = F(lp_d, n + 1) - \sum_{m=1}^{l} (-1)^{m-1} \binom{l}{m} F\left((l - m)p_d, n + 1\right) \tag{19}$$

where $l = 2, \ldots, p_d$, and $M^1 = F(lp_d, n + 1)$.

References

[1] E. Arikan, "Some complexity results about packet radio networks," *IEEE Trans. Inf. Theory*, vol. IT-30, no. 4, pp. 681–685, Jul. 1984.

[2] J. Ariyakhajorn, P. Wannawilai, and C. Sathitwiriyawong, "A comparative study of random waypoint and Gauss–Markov mobility models in the performance evaluation of MANET," in *Proc. IEEE ISCIT*, Sep. 2006, pp. 894–899.

[3] Z. Cai, M. Lu, and C. Georghiades, "Topology-transparent time division multiple access broadcast scheduling in multihop packet radio networks," *IEEE Trans. Veh. Technol.*, vol. 52, no. 4, pp. 970–984, Jul. 2003.

[4] I. Chalamatac and A. Farago, "Making transmission schedules immune to topology changes in multi-hop packet radio networks," *IEEE/ACM Trans. Netw.*, vol. 2, no. 1, pp. 23–29, Feb. 1994.

[5] A. M. Chou and V. O. K. Li, "Fair Spatial TDMA channel access protocols for multihop radio networks," in *Proc. IEEE INFOCOM*, Apr. 1991, pp. 1064–1073.

[6] D. Cohen, "Uniform distribution of polynomial over finite fields," *J. London Math. Soc.*, vol. s2-6, no. 1, pp. 93–102, Dec. 1972.

[7] A. Ephremides and T. V. Truong, "Scheduling broadcast in multihop radio networks," *IEEE Trans. Commun.*, vol. 38, no. 4, pp. 456–460, Apr. 1990.
[8] F. Farnoud and S. Valaee, "Reliable broadcast of safety messages in vehicular ad hoc networks," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 226–234.
[9] S. Gandham, M. Dawande, and R. Prakash, "Link scheduling in wireless sensor networks: Distributed edge-coloring revisited," *J. Parallel Distrib. Comput.*, vol. 68, no. 8, pp. 1122–1134, Aug. 2008.
[10] T. Hou and V. O. K. Li, "Transmission range control in multihop packet radio networks," *IEEE Trans. Commun.*, vol. COM-34, no. 1, pp. 38–44, Jan. 1986.
[11] L. Hu, "A novel topology control for multihop packet radio networks," in *Proc. IEEE INFOCOM*, Apr. 1991, pp. 1084–1093.
[12] J. H. Ju and V. O. K. Li, "An optimal topology-transparent scheduling method in multhop packet radio networks," *IEEE/ACM Trans. Netw.*, vol. 6, no. 3, pp. 298–306, Jun. 1998.
[13] J. Kivett and R. Cook, "Enhancing PLRS with user-to-user data capability," in *Proc. IEEE PLANS*, Apr. 1986, pp. 154–161.
[14] Y. Liu, V. O. K. Li, K.-C. Leung, and L. Zhang, "Is topology-transparent scheduling really inefficient?" *IEEE Wireless Commun. Lett.*, vol. 2, no. 6, pp. 659–662, Dec. 2013.
[15] Y. Liu, V. O. K. Li, K.-C. Leung, and L. Zhang, "Topology-transparent broadcast scheduling with erasure coding in wireless networks," *IEEE Commun. Lett.*, vol. 17, no. 8, pp. 1660–1663, Aug. 2013.
[16] Y. Liu, V. O. K. Li, K.-C. Leung, and L. Zhang, "Topology-transparent distributed multicast and broadcast scheduling in mobile ad hoc networks," in *Proc. IEEE VTC-Spring*, May 2012, pp. 1–5.
[17] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam, The Netherlands: North Holland, 1977.
[18] K. Oikonomou and I. Stavrakakis, "Analysis of a probabilistic topology-unaware TDMA MAC policy for ad hoc networks," *IEEE J. Sel. Areas Commun.*, vol. 22, no. 7, pp. 1286–1300, Sep. 2004.
[19] S. Ramanathan, "A unified framework and algorithm for channel assignment in wireless networks," *Wireless Netw.*, vol. 5, no. 2, pp. 81–94, Mar. 1999.
[20] C. H. Rentel and T. Kunz, "Bounds and parameter optimization of medium access control coding for wireless ad hoc and sensor networks," *Ad Hoc Netw.*, vol. 10, no. 1, pp. 128–143, Jan. 2012.
[21] I. Rhee, A. Warrier, J. Min, and L. Xu, "DRAND: Distributed randomized TDMA scheduling for wireless ad hoc networks," *IEEE Trans. Mobile Comput.*, vol. 8, no. 10, pp. 1384–1396, Oct. 2009.
[22] D. R. Stephens, B. Salisbury, and K. Richardson, "JTRS infrastructure architecture and standards," in *Proc. IEEE MILCOM*, Oct. 2006, pp. 1–5.
[23] D. Stiliadis and A. Varma, "Latency-rate servers: A general model for analysis of traffic scheduling algorithms," *IEEE/ACM Trans. Netw.*, vol. 6, no. 5, pp. 611–624, Oct. 1998.
[24] Y. S. Su, S. L. Su, and J. S. Li, "Topology-independent link activation scheduling schemes for mobile CDMA ad hoc networks," *IEEE Trans. Mobile Comput.*, vol. 7, no. 5, pp. 599–616, May 2008.
[25] Q. Sun, V. O. K. Li, and K.-C. Leung, "Topology-transparent distributed scheduling in multi-hop wireless networks," in *Proc. IEEE GLOBECOM*, Nov./Dec. 2008, pp. 1–5.
[26] V. R. Syrotiuk, C. J. Colbourn, and S. Yellamraju, "Rateless forward error correction for topology-transparent scheduling," *IEEE/ACM Trans. Netw.*, vol. 16, no. 2, pp. 464–472, Apr. 2008.
[27] S. Xu and T. Saadawi, "Does the IEEE 802.11 MAC protocol work well in multihop wireless ad hoc networks?" *IEEE Commun. Mag.*, vol. 39, no. 6, pp. 130–137, Jun. 2001.
[28] G. Zhang, J. Li, W. Zhang, and L. Zhou, "Topology-transparent schedule with reservation and carrier sense for multihop ad hoc networks," *IEEE Commun. Lett.*, vol. 10, no. 4, pp. 314–316, Apr. 2006.
[29] E. Zola and F. Barcelo-Arroyo, "Impact of mobility models on the cell residence time in WLAN networks," in *Proc. IEEE SARNOFF*, Apr. 2009, pp. 1–5.

Yiming Liu (S'14) received the B.Eng. degree in electronic engineering from Tsinghua University, Beijing, China, in 2009, where he is currently working toward the Ph.D. degree in electronic engineering.

His research interest is in the area of wireless communications and networking, specifically in topology-transparent scheduling and cross-layer design.

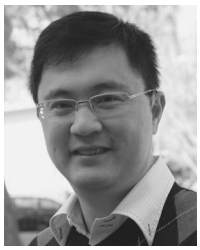
Victor O. K. Li (F'92) received the B.S., M.S., E.E., and D.Sc. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 1977, 1979, 1980, and 1981, respectively.

He is a Chair Professor of information engineering and the Head of the Department of Electrical and Electronic Engineering with the University of Hong Kong (HKU), Pokfulam, Hong Kong. He has also served as an Associate Dean of Engineering and the Managing Director of Versitech Ltd., which is the technology transfer and commercial arm of HKU. He served on the board of China.com Ltd. and now serves on the board of Sunevision Holdings Ltd. and Anxin-China Holdings Ltd., which are listed on the Hong Kong Stock Exchange. Previously, he was a Professor of electrical engineering with the University of Southern California (USC), Los Angeles, CA, USA, and the Director of the USC Communication Sciences Institute. Sought by government, industry, and academic organizations, he has lectured and consulted extensively around the world.

Dr. Li has received numerous awards, including the PRC Ministry of Education Changjiang Chair Professorship at Tsinghua University, Beijing, China; the U.K. Royal Academy of Engineering Senior Visiting Fellowship in Communications; the Croucher Foundation Senior Research Fellowship; and the Order of the Bronze Bauhinia Star, Government of the Hong Kong Special Administrative Region, China. He is a Registered Professional Engineer and a Fellow of the Hong Kong Academy of Engineering Sciences, the IAE, and the HKIE.

Ka-Cheong Leung (S'95–M'01) received the B.Eng. degree in computer science from the Hong Kong University of Science and Technology, Hong Kong, in 1994 and the M.Sc. degree in electrical engineering (computer networks) and the Ph.D. degree in computer engineering from the University of Southern California, Los Angeles, CA, USA, in 1997 and 2000, respectively.

From 2001 to 2002, he was a Senior Research Engineer with the Nokia Research Center, Nokia Inc., Irving, TX, USA. Between 2002 and 2005, he was an Assistant Professor with the Department of Computer Science, Texas Tech University, Lubbock, TX, USA. Since June 2005, he has been with the University of Hong Kong, Pokfulam, Hong Kong, where he is currently an Assistant Professor with the Department of Electrical and Electronic Engineering. His research interests include transport layer protocol design, wireless packet scheduling, congestion control, routing, and vehicle-to-grid.



Lin Zhang (M'08) received the B.Sc., M.Sc., and Ph.D. degrees from Tsinghua University, Beijing, China, in 1998, 2001, and 2006, respectively.

He is currently an Associate Professor with Tsinghua University, where, for the past few years, he has been teaching selected topics in communication networks (40230992) and information theory (70230063) to senior undergraduate and graduate students, and a Visiting Associate Professor with Stanford University, Stanford, CA, USA. In 2006, he led a 2008 Beijing Olympic Stadium (the "Bird's Nest") structural security surveillance project, which deployed more than 400 wireless temperature and tension sensors across the stadium's steel support structure and dome. The system adopted a flexible spectrum sensing and adaptive multihop routing algorithm to overcome strong radio interference and long-distance transmission channel fading and played a critical role in the construction of the stadium. Since then, he has implemented wireless sensor networks in a wide range of application scenarios, including underground mine security, precision agriculture, and industrial monitoring. Since 2008, he has been working in close association with CISCO to develop a Metropolitan Area Sensing and Operating Network (MASON). MASON provides a smart-city and intelligent-urbanization sensor network system for metropolitan areas. MASON has attracted the interest of several large-sized Chinese cities, including Beijing, Shenzhen, Tianjing, and Chengdu. Recently, he has also led two National Science Foundation of China projects, three National High-Tech Developing (863) projects, and more than ten research projects that are primarily funded by private industry in the area of wireless sensor networks. He is a coauthor of more than 40 peer-reviewed technical papers and has five U.S. or Chinese patent applications. His current research focuses on wireless sensor networks, distributed data processing, and information theory.

Dr. Zhang and his team received the IEEE/ACM SenSys 2010 Best Demo Award. In 2004 and 2010, he received Excellent Teacher Awards from Tsinghua University.