J Supercomput (2013) 63:407–430
DOI 10.1007/s11227-008-0244-7

Distributed scheduling algorithms for channel access in TDMA wireless mesh networks

Hongju Cheng · Naixue Xiong · Larence T. Yang · Young-Sik Jeong

Published online: 22 October 2008

Abstract In this paper, we have considered the distributed scheduling problem for channel access in TDMA wireless mesh networks. The problem is to assign time-slot(s) for nodes to access the channels, and it is guaranteed that nodes can communicate with all their one-hop neighbors in the assigned time-slot(s). And the objective is to minimize the cycle length, i.e., the total number of different time-slots in one scheduling cycle. In single-channel ad hoc networks, the best known result for this problem is proved to be K^2 in arbitrary graphs (IEEE Trans Comput C-36(6):729–737, 1987) and $25K$ in unit disk graphs (IEEE/ACM Trans Netw pp 166–177, 1993) with K as the maximum node degree. There are multiple channels in wireless mesh networks, and different nodes can use different control channels to reduce congestion on the control channels. In this paper, we have considered two scheduling models for wireless mesh networks. The first model is that each node has two radios, and the scheduling is simultaneously done on the two radios. We have proved that the upper bound of the cycle length in arbitrary graphs can be $2K$. The second model is that the time-slots are scheduled for the nodes regardless of the number of radios on them. In this case, we have proved that the upper bound can be $(4K - 2)$. We

H. Cheng (✉)
Department of Computer Science, Fuzhou University, Fuzhou, People's Republic of China
e-mail: cscheng@fzu.edu.cn

N. Xiong
Department of Computer Science, Georgia State University, Atlanta, USA
e-mail: nxiong@cs.gsu.edu

L.T. Yang
Department of Computer Science, St. Francis Xavier University, Antigonish, Canada
e-mail: ltyang@stfx.ca

Y.-S. Jeong
Department of Computer Engineering, Wonkwang University, Iksan, South Korea
e-mail: ysjeong@wku.ac.kr

also have proposed greedy algorithms with different criterion. The basic idea of these algorithms is to organize the conflicting nodes by special criterion, such as node identification, node degree, the number of conflicting neighbors, etc. And a node cannot be assigned to a time-slot(s) until all neighbor nodes, which have higher criterion and might conflict with the current node, are assigned time-slot(s) already. All these algorithms are fully distributed and easy to realize. Simulations are also done to verify the performance of these algorithms.

Keywords Scheduling algorithms · Time Division Multiple Access (TDMA) · Wireless mesh networks · Distributed algorithms · Channel access.

1 Introduction

The wireless mesh network [1] is a new wireless broadband technology drawing increasing attention these days. It is deployed in the last mile to extend or enhance Internet connectivity for mobile clients located at the edge of wired networks. Different from the traditional single-channel ad hoc network, the wireless mesh network generally includes several orthogonal channels which can be provided by IEEE 802.11 a/b/g. The mesh node is generally equipped with multiple radios, and each radio can communicate with different neighbors simultaneously on different orthogonal channels. In this way, the bandwidth and system throughput is improved significantly compared with the single-channel network [21]. Another important characteristic of the wireless mesh network is its ad hoc mode. The mesh nodes make decisions with limited local information. The correctness and efficiency of these local decisions depend heavily on the message exchange in the networks [21, 22]. This means that an efficient and fully distributed mechanism to control packet exchange is necessary in the wireless mesh network. Consequently, distributed access scheduling on the channels is of great importance.

The wireless mesh network is designed to support wireless broadband access and various scenarios such as voice and video meeting, broadband home networking, building automation, etc. The Quality-of-Service (QoS) support is one key issue in these applications. QoS support affects not only in the process of the resource reservation and relaxation, but also in the process of local information exchange. For example, if a node wants to send control packets to neighbors, it is reasonably required that the transmission process will be correctly done in a guaranteed delay bound. The contention-based access scheduling, such as the IEEE 802.11 DCF, cannot provide such a guarantee because of its randomly contention and back-track mechanism. Furthermore, the contention-based solution only runs well in small-size network. With increasing size, the performance decreases to zero. All these show that the contention-based solution is not suitable for the wireless mesh networks.

This leads to a determined scheduling for channel access. Each node is assigned one/several chances (time-slots in a TDMA system) to access the channels, and the broadcasting in the time-slots is guaranteed to be received correctly by neighbor(s). The access delay on the channels is limited to the length of the scheduling cycle. Note that a single time-slot can be reused by two nodes if they do not interfere with each

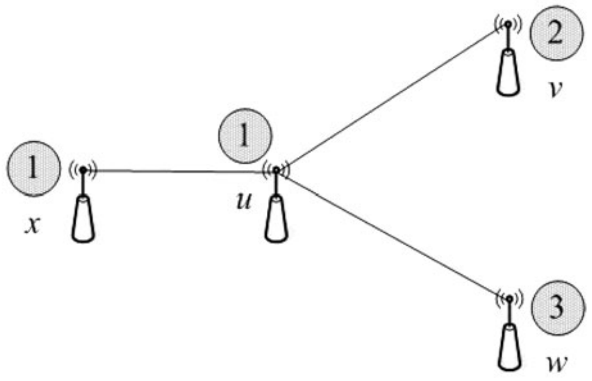


Fig. 1 Control packet exchange mechanism in the radio-oriented model

other. Then the objective of the access scheduling problem is to minimize the length of the scheduling cycle.

When all nodes in the network use a common control channel to transmit the control packets, the length of the scheduling cycle might be too long since all nodes have to contend on this common channel [7, 16]. The mesh network exploits multiple channels in the system and the access scheduling shall also benefit from this characteristic. In order to reduce the congestion on the control channel, a separate control channel can be introduced to each node in the network [12]. The separate control channel for one node is called the *fixed channel*. In case two neighbor nodes have different fixed channels, they shall switch to the fixed channel of the target, and the target shall listen on its own fixed channel at the same time. Switching a radio from one channel to another further incurs some delay, which is in the range of a few hundred microseconds [15] to a few milliseconds [5] in the current situation. However, with the development of the technology, the delay will be minimized and within a tolerable range.

It is expected that the separate control channel model can reduce congestion on the control channels while providing reduced scheduling cycle for channel access. However, in this paper, we do not discuss how to select the control channel for each node. We just focus on the scheduling problem for channel access in the mesh networks. Our theoretical analysis shows that a wonderful control channel assignment will reduce the length of the scheduling cycle compared with the single-channel solution. However, the scheduling for channel access depends heavily on the system model of the mesh networks. In this paper, we consider two typical models [12, 16].

The first model is called the *radio-oriented model,* in which each node has two radios: the first one working on the fixed channel and called the *fixed radio*, and another one which can switch among all the other available channels and called the *switchable* radio [12]. If two neighbor nodes have different fixed channels, they shall switch their fixed radio to the fixed channel of the neighbor, and send control packets through it. However, if they have the same fixed channels, they shall use the fixed radio working on the fixed channel to send the control packets. Note that a node will always listen on its fixed channel (as well as on the fixed radio).

Figure 1 provides an example to illustrate the mechanism in the radio-oriented model. The numbers in the gray circles are the number of fixed channels of the node.

The fixed channels of u, v, w, and x are channels 1, 2, 3, and 1, accordingly. u and v, u and w, and u and x are within transmission range of each other. u can turn its switchable radio to the fixed channel of v, i.e., 2, and send the control packet to v. However, u has to use its fixed radio (working on the fixed channel 1) to send to x because they have the same fixed channel.

The second system model is called the *node-oriented model,* in which time-slots are assigned to the node, regardless of the number of radios on it. This model is popular and reasonable because people seldom care how many radios one node owns or which channels the radios work on. The node-oriented model is introduced here to describe the case time-slots are assigned to nodes regardless the radios on it. Each node also has a separate control channel called the *fixed channel*. The exchange mechanism of control packets in this model is similar to the case in the radio-oriented model, i.e., when a node wants to send control packets to another, it switches to the fixed channel of the target and sends on it, and at the same time-slot the target shall listen on its own fixed channel.

Based on the two different models, we have proposed fully distributed algorithms to solve the scheduling problem to provide channel access. Our algorithms have the following advantages: (1) A determined scheduling which improves system performance by eliminating all conflicts and retransmissions in the network; (2) Guaranteed access delay with upper bound as the length of the scheduling cycle; (3) High utilization with load balance on the different fixed channels and reduced access delay compared with the single channel model.

The rest of this paper is organized as following. In Sect. 2, we discuss related work. In Sect. 3, the problem formulation is shown. In Sect. 4, we propose algorithms based on the radio-oriented model. In Sect. 5, we introduce the algorithms based on the node-oriented model. The related discussions are presented in Sect. 6. Simulation results are presented in Sect. 7, and in Sect. 8 are the concluding remarks.

2 Related work

So far as we know, this is the first paper on the access scheduling problem in multichannel mesh networks. The known related works were all done in single channel wireless networks. Most of the current results are based on the assumption that the interference range is equal to the transmission range, which was proved not true.

Chlamtac and Pinter [7] have proposed a greedy algorithm for the access scheduling problem in single channel networks. The basic idea is that each node selects a time-slot for broadcasting with 2-hop local information. The number of time-slots is upper-bounded by $\min(n, K^2 + 1)$ for arbitrary graphs, where n is the node number and K is the maximum node degree. The main problem is that the interference range is assumed to be equal to the transmission range. However, recent results [23] have shown that the interference range is generally about 2–3 times the transmission range when the distance between the transmitter and receiver is close to the transmission range, which means nodes may potentially interfere even though they are 2-hops away.

Some algorithms make access scheduling on the channel with in-band signaling, such as the strategies of phase negotiation [8] and RTS/CTS handshakes [20, 24].

The in-band signaling in these algorithms is simple and easy to realize, but it cannot eliminate the potential interference completely because the interference range can be larger than the transmission range, and thus collision may occur at some time-slots, and finally the system performance is reduced. Another problem is that these methods spend some band resources on conflicts. Further, deadlock may happen in the phase negotiation method because nodes know nothing about the local topology.

A number of topology-independent scheduling algorithms have been proposed [4, 11, 19]. The basic idea of topology-independent scheduling is for a node to transmit in several time-slots in one scheduling cycle, so that there is a unique time-slot on which a node can communicate with any one-hop neighbor. The benefit of topology-independent scheduling is that the scheduling result is independent of the topology of the network, and thus very suitable for dynamic scenarios where the topology changes frequently. The limitations are that the length of the scheduling cycle is generally very long and the result always depends on the network size.

The broadcast scheduling problem was studied in [6, 13, 16, 18]. Most works are centralized algorithms [6, 13, 16]. Ramanathan and Lloyd [18] have proposed several on-line algorithms. The main idea is that time-slots are scheduled according to a series of events: the nodes' joining and leaving the network. The on-line algorithms need a centralized node to deal with these events, and thus they are, in fact, centralized. There has been no fully distributed algorithm for the broadcast scheduling problem until now.

Another problem that is similar to the channel access scheduling problem is the code assignment in CDMA systems, if we take the codes as the time slots in TDMA systems [2, 3, 9, 10, 14, 17]. A pair-wise code-assignment algorithm is proposed to assign codes to edges [9]. Simulation results show that the proposed scheme requires fewer codes than transmitter-based code assignment, while maintaining throughput performance in sparse networks. Code assignment for hidden terminal interference avoidance in multi-hop packet radio networks is studied in [2, 14]. Optimal algorithms are proposed for ring and binary tree topologies [2] while optimal algorithms are proposed for bus, hex, and grid topologies [14]. Heuristics are proposed for general network topologies in both works. A centralized code assignment is proposed in [10]. The main idea is to first assign codes for the node with maximum node degree and its neighbors, and then order the nodes by decreasing criterion of the node degree, and then assign codes to nodes one by one. A distributed code assignment algorithm with object-oriented software simulations is presented in [10]. The algorithm is based on the saturation-degree coloring scheme by using different depth information of the neighborhood when a local decision must be made.

3 Problem formulation

Assume there are totally Q channels in the network, numbered from 1 to Q. A node is equipped with two omni-directional radios. There is a preassigned control channel (called the *fixed channel*) for each node in the network. Different nodes may have different fixed channels. The topology of the network can be represented by a graph $G = (V, E)$ where V is the set of nodes and E is the set of edges. There is an edge

(u, v) between node u and v if they can communicate with each other, or their Euclidean distance is no larger than the transmission range. We assume all nodes have the same transmission range in the rest of this paper.

The number of edges on the shortest path $u = v_0, v_1, \ldots, v_h = v$ is defined as the distance between nodes u and v. The set of nodes whose distance to node u is no more than variable h is defined as the h-hop neighbors of node u, i.e., N_u^h; especially N_u^1 denotes the set of one-hop neighbors of node u in the network.

Most current works assume that the interference range is equal to the transmission range. However, recent results [23] have shown that this assumption is not true. It is proved that the interference range is generally larger than the transmission range, that is, about 2–3 times, when the distance between the transmitter and receiver is close to the transmission range. Thus, the previous assumption is not accurate and conflicts may occur. Here, we propose an H-hop interference model to describe this large interference range problem.

The H-hop model has defined the case in which a transmission is *received* successfully by its intended receiver. Supposing node u transmits over channel k to node v, this transmission is successfully received by node v if no other node $w \neq u$ within H-hop from v is transmitting over channel k simultaneously. The H-hop interference model has described the situation where a zone (within H hops from the receiver) is specified, to prevent a nearby node from transmitting on the same channel at the same time. The parameter H is a constant, depending on the system signal-to-interference ratio required for correct reception.

The access scheduling problem assigns time-slots to each node in the network for broadcasting. The correctness of the scheduling depends heavily on the system model. In the radio-oriented model, each node has one fixed radio working on the fixed channel (the control channel), and another radio that can switch among the other channels. In this model, the scheduling is carried out simultaneously on the two radios for each node. Assume that a node, say u, is scheduled one time-slot which can be used to send control packets to another node, say v. The following conditions must be satisfied to guarantee this schedule is feasible.

$C1$-1. u should transmit on v's fixed channel with the fixed radio if their fixed channels are the same, and with the switchable radio if different.
$C1$-2. Transmission and reception between u and v must occur in the same time-slot.
$C1$-3. One time-slot of the same radio can not be used for transmission and reception simultaneously.
$C1$-4. u can not transmit or receive on another channel in the same time-slot.
$C1$-5. Nodes in N_v^H except u can not transmit on channel k in the same time-slot.

Conditions $C1$-2, $C1$-3, and $C1$-4 explicitly show that the radio works in the half-duplex mode. Condition $C1$-5 guarantees the correctness of v's reception.

The access scheduling problem for channel access with the radio-oriented model can be formulated as: Given a graph $G = (V, E)$ and the fixed channel of each node in V, assign time-slots on the fixed and switchable radios to each node in V so that the node can transmit the control packet to any one-hop neighbor in one scheduling cycle while satisfying the conditions $C1$-1–$C1$-5. The objective is to minimize the length of the cycle.

The conditions are a little different with the node-oriented model in which time-slots are assigned to nodes regardless of the number of radios on them. Assume one node, say u, is scheduled to send control packets to another node, say v. The following conditions must be satisfied to guarantee this schedule is feasible.

$C2$-1. u should transmit on v's fixed channel.
$C2$-2. Transmission and reception between u and v must occur in the same time-slot.
$C2$-3. One time-slot cannot be used for transmission and reception simultaneously.
$C2$-4. u cannot transmit or receive on another channel in the same time-slot.
$C2$-5. Nodes in N_v^H except u can not transmit on channel k in the same time-slot.

The access scheduling problem for channel access with the node-oriented model can be formulated as: Given a graph $G = (V, E)$ and *fixed*(u) for each $u \in V$, assign time-slots to nodes in V so that nodes can transmit the control packet to any one-hop neighbor in the scheduling cycle. The objective is also to minimize the cycle length.

Consider the special case where all nodes in the network have the same fixed channel, and a single time-slot cannot be assigned to two nodes if they are within $(H+1)$-hop. If we view the time-slot as the color, the scheduling problem can be induced to a graph coloring problem as follows: construct graph $G' = (V', E')$, in which $V' = V$ and $(u, v) \in E'$ if and only if the distance between u and v on graph G is no more than $(H+1)$. The graph coloring problem is proved to be NP-complete even in some special graphs. This means that the access scheduling problem on the control channels is also NP-complete. Heuristic algorithms are needed to solve this in polynomial time.

4 Algorithms for the radio-oriented model

4.1 The centralized One Neighbor per Cycle Algorithm 1 (ONPC1)

For each node u in the network, the One Neighbor per Cycle Algorithm 1 (ONPC1) assigns *one* time-slot on the fixed radio and *one* time-slot on the switchable radio in the scheduling cycle. The broadcasting of the control packet at the assigned time-slot on the fixed radio can be correctly received by all one-hop neighbors with the same fixed channel as node u. Node u can switch its switchable radio in the assigned time-slot to the fixed channel of any neighbor whose channel is different, and send the control packet to that neighbor. The strategy is similar to the problem of collision-free scheduling in a single-channel radio network [4, 6, 7, 13, 16, 18].

The ONPC1 guarantees that each node has the right to speak/broadcast to any one-hop neighbor in one scheduling cycle. However, the algorithm does not explicitly indicate the channel it switches to at the assigned time-slot on the switchable radio. This means that all potential receivers listen in the same time-slot.

In Fig. 1, assume that 2 is the number of assigned time-slot for u on the fixed radio, and 1 on the switchable radio. Obviously, u can broadcast control packets to x at time-slot 2 on the fixed radio. Now consider the switchable radio on u. Note that v and w have different fixed channels which are distinct from that of u. With Condition $C1$–4, u can send to either v or w at time-slot 2 on the switchable radio, but it cannot

Algorithm for Centralized One Neighbor per Cycle 1 (ONPC1)

```
Input: topology $G = (V, E)$ and $fixed(u), u \in V$.
Output: the assigned time-slots on the fixed and switchable radios for each node $u, u \in V$.
1. Arrange the nodes by a special criterion into sequence $\{1, 2, \ldots, n\}$.
2. Select the first unassigned node in the sequence.
3. If at least one neighbor has the same fixed channel as the selected node, select the
   least number of time-slots that can be used to transmit to all such neighbor nodes
   and does not break $C1$-1–$C1$-5.
4. If at least one neighbor has a different fixed channel from the selected node, select
   the least number of time-slot that can be reused to transmit to ALL such neighbor
   nodes and does not break $C1$-1–$C1$-5.
5. If all nodes are assigned, stop. Otherwise, return to step 2.
```

Fig. 2 The centralized ONPC1

send to them simultaneously in one scheduling cycle. Since v and w do not know which channel that node u will switch to in time-slot 2, they have to listen on their fixed channel in this time-slot simultaneously.

For convenience, we first introduce the centralized algorithm. The basic idea of the ONPC1 is as follows. Arrange the nodes by a special criterion into sequence $\{1, 2, \ldots, n\}$ where n is the node number. The criterion shall be unique for each node and it can be chosen as the node identification, node degree, the number of 2-hop neighbors, or the number of all potential interference neighbors, etc. The selection of the criterion will be discussed later. Then assign time-slots consequently to the fixed and switchable radios of the nodes. The process continues until all nodes are scheduled. The assigned time-slot on the fixed or switchable radio is the least time-slot unused by its $(H + 1)$-hop neighbors. Pseudo code for ONPC1 is listed in Fig. 2.

Let n be the number of nodes in the network, and K be the maximum node degree. The following theorems prove the upper bound of the length of the cycle.

Theorem 1 *For arbitrary graphs, the ONPC1 guarantees an upper bound on the total number of time-slots used as*:

$$\min\left(n, K + 1 + K(K-1)\frac{(K-1)^H - 1}{K-2}\right).$$

Proof Suppose the greedy algorithm is going to assign a timeslot for node u. With conditions $C1$-1–$C1$-5, the assignment result that might interfere with the assignment of node u is the assignment of nodes within $(H + 1)$-hop from node u. The number of nodes that are one-hop away from node u is no more than K, and the number of nodes that are 2-hops away is no more than $K(K - 1), \ldots,$ the number of nodes that are $(H + 1)$-hop away is no more than $K(K - 1)^H$. The total number of potential interference nodes is:

$$K + K(K-1) + \cdots + K(K-1)^H = K + (K-1)\frac{(K-1)^H - 1}{K-2}.$$

This means that the number of conflicting time-slots is no more than

$$\min\left(n-1, K+K(K-1)\frac{(K-1)^H-1}{K-2}\right).$$

Note the algorithm always assigns the least unsigned time-slot, the number of time-slot assigned on either the fixed radio or the switchable radio for node u is no more than

$$\min\left(n, K+1+K(K-1)\frac{(K-1)^H-1}{K-2}\right).$$

□

Lemma 2 *For arbitrary graphs, the ONPC*1 *guarantees an upper bound on the total number of time-slots used as* $\min(n, K^2+1)$ *in case that* $H=1$.

Proof The conclusion can be easily induced with Theorem 1 with $H=1$. □

Theorem 3 *The time complexity of ONPC*1 *is* $O(n^2)$.

Proof The time complexity of the ordering is $O(n\log n)$. In the following assignment process, the number for node selection is n and the assignment $2n$. The number of comparison is at most n when selecting one time-slot for one assignment. So, the full time complexity of the algorithm is $O(n\log n)+O(2n\cdot n)=O(n^2)$. □

4.2 The centralized All Neighbors per Cycle Algorithm 1 (ANPC1)

In the ONPC1, only one time-slot is assigned on the switchable radio of each node. However, the All Neighbors per Cycle Algorithm 1 (ANPC1) assigns several time-slots on the switchable radio so that the node can transmit the control packet to all one-hop neighbors in one scheduling cycle. The assignment results not only indicate the time-slot, but also the channel that the switchable radio shall switch to in this time-slot. Accordingly, the receivers are also indicated (the one-hop neighbors with the identical fixed channel).

Let *fixed*(u) be the fixed channel of u, and *NFixed*(u) be the set of the fixed channels of the one-hop neighbors of u. The basic idea of the ANPC1 is as follows. Arrange the nodes by a special criterion into sequence $\{1,2,\dots,n\}$. The criterion shall be unique for each node also and it can be chosen as the node identification, node degree, the number of 2-hop neighbors, or the number of all potential interference neighbors, etc. Then select the first node u in the sequence and assign one time-slot on the fixed radio and ensure that the broadcasting at it on channel *fixed*(u) can be correctly received by all its one-hop neighbors whose fixed channel is *fixed*(u). For each channel $k\neq$ *fixed*(u) and $k\in NFixed(u)$, assign one time-slot on the switchable radio and ensure that the broadcasting on it using channel k can be correctly received by all one-hop neighbors with fixed channel as k. The process continues until all nodes are assigned. Pseudo code for ANPC1 is listed in Fig. 3.

Algorithm for Centralized All Neighbors per Cycle 1 (APNC1)

```
Input: topology $G = (V, E)$ and fixed($u$), $u \in V$.
Output: the assigned time-slots on the fixed and switchable radios for each node $u, u \in V$.
1. Arrange the nodes by a special criterion into sequence $\{1, 2, \ldots, n\}$ and mark all
   nodes as UNASSIGNED.
2. Select the first unassigned node $u$ and calculate its NFixed($u$).
3. Select one channel number $k$ from NFixed($u$).
4. If the selected channel number $k$ is the same as fixed($u$), select the least number
   of time-slots that can be used by node $u$ to transmit on the fixed radio to all
   neighbors whose fixed radio is $k$, and the assignment satisfies $C1$-1–$C1$-5.
5. If the selected channel number $k$ is different from fixed($u$), select the least number
   of time-slots that can be used by node $u$ to transmit on the switchable radio to all
   neighbors whose fixed radio is $k$, and the assignment satisfies $C1$-1 $\sim$ $C1$-5.
6. Remove $k$ from NFixed($u$) and return to step 3.
7. If NFixed($u$) is empty, change $u$'s state to ASSIGNED.
8. If all nodes are assigned, stop. Otherwise, return to step 2.
```

Fig. 3 Centralized APNC1

In Fig. 1 u, v, and w have distinct fixed channels. With the ANPC1, u needs two time-slots on the switchable radio. Assuming that the scheduling results are time-slots 1 and 2, and the relevant channels are 2 and 3, then u can switch its switchable radio to channel 2 at time-slot 1 and send the control packet to v; and channel 3 at time-slot 2 to w. In this case, v only needs to listen in time-slot 1, and w in time-slot 2.

Intuitively, the ANPC1 needs more time-slots than the ONPC1 for any node in the network and it shall result in a longer length of scheduling cycle. However, both theoretical and simulation show that ANPC1 has wonderful performance compared with the ONPC1. The main reason may be that the channel number is not explicitly indicated in the assigned time-slot on the switchable radio in ONPC1. With $C1$-3 and $C1$-5, this time-slot cannot be reused by all potential receivers. The total number of time-slots required way increase because of this.

The following theorems prove the upper bound of the length of the scheduling cycle.

Theorem 4 *For arbitrary graphs, the ANPC*1 *guarantees an upper bound on the total number of time-slots used as*

$$\min\left(n, K + 1 + M(K-1)\frac{(K-1)^H - 1}{K-2}\right),$$

where M is the maximum number of neighbors of some nodes whose fixed channels are the same and $1 \le M \le K$.

Proof Assume that we are to assign one time-slot on channel k for node u. The assignment can be either on the fixed radio or the switchable radio of node u. Let us consider the two cases separately.

Case (A): $k = \mathit{fixed}(u)$. In this case, to satisfy $C1$-1 the time-slot is assigned to node u on the fixed radio. The receivers are the one-hop neighbors whose fixed channel is k. Let S be the set of such nodes and m be the size of $S, 0 < m \le M$. Now calculate the number of time-slots that cannot be assigned to node u on the fixed radio.

To satisfy $C1$-2–$C1$-4, the time-slots used for reception on the fixed radio by node u cannot be reused. The reception is from the one-hop neighbors, so the number is K. The time-slots used for transmission on the fixed radio by node u cannot be reused either. Since the fixed radio transmits only on the fixed/control channel, the number of such time-slots is 0. Furthermore, the time-slots used for transmission on the fixed radio by nodes in set S cannot be reused. Note that nodes in S have the same fixed channel as node u. These transmission time-slots are also the reception time-slots of node u, which have been calculated already.

To satisfy $C1$-5, the transmission time-slots on channel k used by nodes whose distance to nodes in S is no more than H, cannot be reused. The number is:

$$m(K-1) + m(K-1)^2 \cdots + m(K-1)^H = m(K-1)\frac{(K-1)^H - 1}{K-2}.$$

Note that only one time-slot is assigned to each node on any given channel. So, the total number of time-slots that cannot be reused by node u for transmission is:

$$K + 0 + 0 + m(K-1)\frac{(K-1)^H - 1}{K-2} = K + m(K-1)\frac{(K-1)^H - 1}{K-2}.$$

When $m = M$, the length of the cycle is maximized and the theorem stands.

Case (B): $k \neq \mathit{fixed}(u)$. In this case, to satisfy $C1$-1 the time-slot is assigned on the switchable radio of node u. The proof process is similar to case A. □

Lemma 5 *For arbitrary graphs, the ANPC1 guarantees an upper bound on the total number of time-slots used as* $\min(n, 2K)$ *if* $H = 1$ *and all neighbors have different fixed channels for any node.*

Proof If all neighbors have different fixed channels for any node, $M = 1$. The lemma can be induced with Theorem 1 with $H = 1$ and $M = 1$. □

Theorem 6 *The time complexity of ANPC1 is* $O(Kn^2)$.

Proof The time complexity of the ordering is $O(n \log n)$. In the following assignment process, the number of node selection is n and the assignment is at most Kn. The number of comparison is at most n when selecting one time-slot for one assignment. So, the full time complexity of the algorithm is $O(n \log n) + O(Kn \cdot n) = O(Kn^2)$. □

4.3 Distributed versions

With the H-hop interference model, the time-slot assignment for one node will influence the assignment for nodes at most $(H + 1)$-hop away. This also means that in distributed situations two nodes can assign time-slots simultaneously if their distance is more than $(H + 1)$. However, if their distance is no more than $(H + 1)$, there shall be a sequential method to assign time-slots for them one by one to avoid interference.

The basic idea of the distributed algorithms is as follows. Each node first collects the criterion information of nodes within $(H + 1)$-hop away, assuming that the criterion value is unique for each node in the network. If nodes find that their criterion is the maximal compared with the $(H + 1)$-hop neighbors that are still unassigned, they can assign time-slots for themselves. Then the assignment results are broadcast to $(H + 1)$-hop away, and nodes that have lower criterion have chances to assign time-slots for them after receiving ASSIGNMEN messages. This process continues until all nodes are scheduled.

Accordingly, each node maintains a *NeighborTable* which has four fields, *criterion-value*, *fixed-channel*, *state,* and *assign-results*. The *criterion-value* denotes the value of the criterion parameter; the *fixed-channel* is the number of the fixed channel; the *state* describes it is assigned or not, and *assign-results* are the scheduling results.

The distributed versions of the ONPC1 and ANPC1 are listed in Figs. 4 and 5. There are two processes in the algorithms. One is the Initialize-Process and another is

Algorithm for Distributed One Neighbor per Cycle 1 (ONPC1)

```
Initialize-Process( )
1. Initialize its own state as UNASSIGNED, and broadcast a HELLO message
   $(H + 1)$-hops away with the state, fixed-channel and criterion value included in
   the message.
2. Build the neighbor table NeighborTable with incoming HELLO messages.

Assign-Process()
1. Compare the criterion with the nodes in the NeighborTable whose state is UNAS-
   SIGNED.
2. If some neighbors have larger criterion value, wait until receiving an ASSIGN-
   MENT message originated from another node, and then modify the state and
   assign-result for this originating node in the NeighborTable, and return to step 1.
3. If one neighbor has the same fixed channel as the selected node, select the least
   time-slot that can be used to transmit to all such neighbor nodes and satisfy
   $C1$-1–$C1$-5.
4. If one neighbor has a fixed channel different from the selected node, select the
   least time-slot that can be used to transmit to all such neighbor nodes and satisfy
   $C1$-1–$C1$-5.
5. Change the state as ASSIGNED; send an ASSIGNMENT message to $(H + 1)$-
   hops away.
```

Fig. 4 The distributed ONPC1

the Assign-Process. In the Initialize-Process, the node initializes its state as UNASSIGNED, collects its *criterion-value* and *fixed-channel,* and broadcasts a HELLO message to $(H+1)$-hop away. At the end of this process, each node will finally build the *NeighborTable*. In Assign-Process, the node compares its criterion value with those in *NeighborTable* whose state is marked as "UNASSIGNED." If the criterion value is the maximal, the node assigns time-slots for itself according to *assign-results* in *NeighborTable,* and broadcasts an ASSIGNMENT message to $(H+1)$-hop away.

The differences between the distributed ONPC1 and ANPC1 are that in ANPC1 each node also builds the set of fixed channels of its one-hop neighbors, i.e., *NFixed*, and assigns time-slots for each channel in *NFixed* to ensure that it can send control packets to all one-hop neighbors in one scheduling cycle.

Theorem 7 *Each node transmits no more than* $\frac{2K^{H+1}-2}{K-1}$ *messages with* $ONPC1$ *and* $ANPC1$.

Proof During the Initialize-Process, each node sends one message to its neighbors, and also forwards messages from neighbors within $(H+1)$-hop away. The total number of messages transmitted in this process is no more than

$$1+K+K^2+\cdots+K^H = \frac{K^{H+1}-1}{K-1}.$$

```
Algorithm for Distributed All Neighbors per Cycle 1 (ANPC1)
Initialize-Process( )
1. Initialize its own state as UNASSIGNED, and broadcast a HELLO message
   $(H+1)$ hops away with the state, fixed-channel and the value of criterion.
2. Build the neighbor table NeighborTable and NFixed with the incoming HELLO
   messages.

Assign-Process()
1. Compare the criterion with the nodes in the NeighborTable whose state is UNAS-
   SIGNED.
2. If some neighbors have larger criterion value, wait until receiving an ASSIGN-
   MENT message originating from another node, and then modify the state and
   assign-result for this originating node in the NeighborTable, then return to step 1.
3. Select one channel number $k$ from $NFixed(u)$.
4. If $k = fixed(u)$, select the least time-slot that can be used by node $u$ to transmit on
   the fixed radio to all neighbors whose fixed channel is $k$, and satisfy $C1$-1–$C1$-5.
5. If $k \neq fixed(u)$, select the least time-slot that can be used by node $u$ to transmit
   on the switchable radio to all neighbors whose fixed radio is $k$, and satisfy $C1$-1–
   $C1$-5.
6. Remove $k$ from $NFixed(u)$ and return to step 3.
7. If $NFixed(u)$ is empty, change $u$'s state as ASSIGNED, and broadcast an AS-
   SIGNMENT message out to $(H+1)$ hops away.
```

Fig. 5 The distributed ANPC1

The Assign-Process is similar, and so the theorem stands. □

Theorem 8 *The computation time complexity of the distributed ONPC*1 *and ANPC*1 *algorithms is* $O(n)$.

Proof By noticing that each node sends out a constant number of messages and a constant number of steps are performed per message, and the theorem stands. □

5 Algorithms for the node-oriented Model

In the node-oriented model, time-slots that are scheduled for channel access are assigned to the nodes regardless of how many radios they have, which makes the algorithms a little different from those in the radio-oriented model. Here, we also introduce two distributed algorithms for the scheduling problem, namely One Neighbor per Cycle 2 (ONPC2) and All Neighbors per Cycle 2 (ANPC2). The basic idea of the two algorithms is similar to that of ONPC1 and ANPC1. However, in this model, time-slots are scheduled to nodes while in the ONPC1 and ANPC1 time-slots are scheduled on different radios for each node in the network.

5.1 The distributed One Neighbor per Cycle Algorithm 2 (ONPC2)

The One Neighbor per Cycle 2 (ONPC2) algorithm assigns one slot for each node, such that the node is able to transmit control packets to any one-hop neighbors collision-free in every scheduling cycle. This guarantees that each node reserves the right to transmit a control packet in every cycle. The scheduling also means that all one-hop neighbors shall listen in the same time-slot on their own fixed channel.

In Fig. 1, assume that 2 is the assigned time-slot for u. In time-slot 2, u can switch to channel 1 and send the control packet to x in one scheduling cycle; u also can switch to channel 2 (or 3) and send to v (or w) in another cycle. But u can not speak to both v and x in one cycle because v and x have different fixed channels. Note that v, w, and x have to listen in time-slot 2 simultaneously in this case.

The basic idea of the distributed ONPC2 is that each node first collects the criterion information of nodes within $(H+1)$-hop away. If a node finds that its criterion value is maximal compared with the $(H+1)$-hop neighbors which are still unassigned, it can assign time-slots for itself. The results are broadcast to $(H+1)$-hop away, and then nodes that have lower criterion have chances to assign time-slots for themselves.

Accordingly, each node also maintains a *NeighborTable* with three fields, which are the same as in ONPC1. The pseudo code for the algorithm is shown in Fig. 6.

Theorem 9 *For arbitrary graphs, the ONPC*2 *guarantees an upper bound on the total number of time-slots used as*

$$\min\left(n, K+1+K(K-1)\frac{(K-1)^H-1}{K-2}\right).$$

Proof The proof process is similar to Theorem 1. □

```
Algorithm for Distributed One Neighbor per Cycle 2 (ONPC2)

Initialize-Process( )
1. Initialize its own state as UNASSIGNED, and broadcast a HELLO message
   ($H + 1$)-hops away, with its state, fixed channel and criterion value.
2. Build the neighbor table NeighborTable with incoming HELLO messages.

Assign-Process()
1. Compare the criterion with the nodes in the NeighborTable whose state is UNAS-
   SIGNED.
2. Compare with nodes with state as UNASSIGNED in NeighborTable. If some
   neighbors have larger criterion value, wait until receiving an ASSIGNMENT
   message originating from another node; modify the state and assign-result for
   this originating node in the NeighborTable, and return to step 1.
3. Select the least number of time-slot that can be used to transmit to all one-hop
   neighbor nodes, and which satisfies conditions $C2$-1–$C2$-5.
4. Change state as ASSIGNED. And send an ASSIGNMENT message to ($H + 1$)-
   hops away.
```

Fig. 6 The distributed ONPC2

Lemma 10 *For arbitrary graphs, the ONPC*2 *guarantees an upper bound on the total number of time-slots used as* $\min(n, K^2 + 1)$ *in the case that* $H = 1$.

Proof The conclusion can be easily induced with Theorem 9 with $H = 1$. □

Theorem 11 *The time complexity of ONPC*2 *is* $O(n^2)$.

Proof The proof process is similar to Theorem 3. □

Theorem 12 *Each node transmits no more than* $\frac{2K^{H+1}-2}{K-1}$ *messages with ONPC*2.

Proof During the Initialize-Process, each node sends once its list of neighbors, and also forwards messages from neighbors within ($H + 1$)-hop away. The total number of messages transmitted in this process is no more than

$$1 + K + K^2 + \cdots + K^H = \frac{K^{H+1} - 1}{K - 1}.$$

The Assign-Process is similar and the theorem stands. □

Theorem 13 *The computation time complexity of ONPC*2 *is* $O(n)$.

Proof By noticing that each node sends out a constant number of messages and a constant number of steps is performed per message. □

```
Algorithm for Distributed All Neighbors per Cycle 2 (ANPC2)
Initialize-Process( )
1. Initialize its own state as UNASSIGNED, and broadcast a HELLO message
   ($H + 1$) hops away with its state and criterion.
2. Build the neighbor table NeighborTable and NFixed with the incoming HELLO
   messages.

Assign-Process()
1. Compare the criterion with the nodes in the NeighborTable whose state is UNAS-
   SIGNED.
2. Compare with nodes with state as UNASSIGNED in NeighborTable. If some
   neighbors have larger criterion value, wait until receiving an ASSIGNMENT
   message originating from another node; modify the state and assign-result for
   this originating node in the NeighborTable, the return to step 1.
3. For each channel $k$ in NFixed($u$), select the least number of time-slot that can
   be used by node $u$ to transmit to all neighbors with fixed channel $k$, and which
   satisfy $C2$-1–$C2$-5.
4. Change $u$'s state as ASSIGNED. Send an ASSIGNMENT message to ($H + 1$)-
   hops away.
```

Fig. 7 The distributed ANPC2

5.2 The distributed All Neighbors per Cycle Algorithm 2 (ANPC2)

Similar to ANPC1, the All Neighbors per Cycle 2 (ANPC2) algorithm assigns several time-slots to one node so that this node can transmit the control packet to all one-hop neighbors in one scheduling cycle. The assignment results not only indicate the time-slot in the cycle, but also the channel that the node should switch to, and accordingly the receivers (the one-hop neighbors with the identical fixed channel).

The basic idea of ANPC2 is as follows. Each node first collects the criterion information of nodes within $(H + 1)$-hop away. If a node finds that the criterion value is the maximal compared with the $(H+1)$-hop neighbors which are still unassigned, assign one time-slot for each channel k in *NFixed*(u) and ensure that the transmission of control packets at it on channel k can be correctly received by all the one-hop neighbors with fixed channel as k. This process continues until all channels in *NFixed*(u) are assigned. The final results are broadcasted to $(H + 1)$-hop away and now nodes with lower criterion have chances to assign time-slots. Pseudo code is listed in Fig. 7.

In Fig. 1, u has three neighbors, i.e., v, w, and x with different fixed channels. Since the ANPC2 assigns time-slots for all neighbors in one scheduling cycle as mentioned above, u needs three time-slots in the cycle. Assume the scheduling results for u are time-slots 1, 2, and 3, and the relevant channels are 1, 2, and 3. Then in one cycle u can switch to channel 1 in time-slot 1 and send control packet to x. In the same cycle, u can also switch to channel 2 in time-slot 2 and send control packet to v; and to channel 3 in time-slot 3 to send the packet to w. In this case, x only needs to listen in time-slot 1; while v listens in time-slot 2 and v in time-slot 3.

Theorem 14 *For arbitrary graphs, the ANPC2 guarantees an upper bound on the total number of time-slots used as*

$$\left(n, 2K+1+M\left[(K-1)\frac{(K-1)^H-1}{K-2}+K-2\right]\right)$$

for the cycle length, where M is the maximum number of one-hop neighbors whose fixed channels are the same, $1 \le M \le K$.

Proof Suppose currently node u is selected and the algorithm is going to assign a timeslot for transmission from u to one of u's neighbors with fixed channel as k. Let S be the set of such nodes and m be the size of $S, 0 < m \le M$. Now calculate the number of time-slots that cannot be assigned to node u.

To satisfy $C2$-1 and $C2$-4, time-slots that have been assigned for transmissions from u to u's neighbors cannot be reused for this transmission. The number of such time-slots is at most $(K-m)$. Time-slots that have been assigned for reception of group S from other neighbors can not be reused for this transmission. The number of time-slots is at most $m(K-1)$. To satisfy $C2$-2 and $C2$-3, time-slots that have been assigned to group S for transmission cannot be reused. The number of such timeslots is at most mK. At the same time, time-slots that have been assigned to u for reception cannot be reused. Note that the reception of node u is also the transmission of its one-hop neighbor and the number is partially included above. The number of the rest time-slots is $(K-m)$(time-slots for the transmissions of neighbors not in S). To satisfy $C2$-5, time-slots that have been assigned to group v's neighbors except u for transmissions on the fixed channel of v can not be reused. The number of such timeslots is at most

$$m(K-1)+m(K-1)^2\cdots+m(K-1)^H = m(K-1)\frac{(K-1)^H-1}{K-2}.$$

Note that timeslots that have been assigned for reception by group S are also included in the above. So, the total number of timeslots that are NOT available are:

$$\begin{aligned} &K-m+mK+K-m+m(K-1)\frac{(K-1)^H-1}{K-2} \\ &= 2K+m\left[(K-1)\frac{(K-1)^H-1}{K-2}+K-2\right]. \end{aligned}$$

Therefore, timeslots as

$$\min\left(n, 2K+1+m\left[(K-1)\frac{(K-1)^H-1}{K-2}+K-2\right]\right)$$

can be assigned for the transmission from u on channel k to group S.

When $m = M$, the length of the cycle is maximized and the theorem stands. □

Lemma 15 *For arbitrary graphs, the ANPC2 guarantees an upper bound on the total number of time-slots used as* $\min(n, 4K-2)$ *for the cycle length if $H = 1$ and every two neighbors have different fixed channels for any node.*

Proof If every two neighbors have different fixed channels for any node, this means that $M = 1$. The lemma can be induced with Theorem 3 with $H = 1$ and $M = 1$. □

Theorem 16 *Each node transmits no more than* $\frac{2K^{H+1}-2}{K-1}$ *messages during the ANPC2 algorithm.*

Proof During the Initialize-Process, each node sends once its list of neighbors, and also forwards messages from neighbors within $(H + 1)$-hop away. The total number of messages transmitted in this process is no more than

$$1 + K + K^2 + \cdots + K^H = \frac{K^{H+1} - 1}{K - 1}.$$

The Assign-Process is similar and the theorem stands. □

Theorem 17 *The computation time complexity of the distributed ANPC2 algorithms is* $O(n)$.

Proof By noticing that each node sends out a constant number of messages and a constant number of steps is performed per message. □

6 Discussions

6.1 The criterion selection

In the algorithms mentioned above, a special criterion parameter is necessary to organize the nodes in the network to maintain the distributed manner while avoiding the cases that two nodes in the interference range of each others are assigned simultaneously. The criterion can be any parameter that is unique to nodes in the network, such as the station nick name/identification, node degree, the number of $(H + 1)$-hop neighbors, random data or a combination of these parameters. As we can observe from the analysis, the selection of this criterion does not influence the result of the upper bound for the cycle length. However, a good criterion can reduce the length in the general cases. We will illustrate the difference in the simulation to illustrate its importance.

There is still another problem that needs consideration. After the criterion is selected, the nodes can be organized into sequence either in decreasing or increasing order of this criterion. As mentioned above, the order (decreasing or increasing) does not change the theoretical analysis. Intuitively, a node with more neighbor nodes may potentially increase the cycle length, and it should be assigned time-slots ahead of others. This idea is similar to the well-known graph coloring problem in which nodes are generally organized and colored in decreasing order of node degree. Our solution follows this intuition, and nodes are sequenced in a decreasing order of the criterion.

6.2 The dynamic topology

The dynamic topology originates from the movement of the mobile nodes in the traditional ad hoc network. However, the mesh nodes are always placed at fixed positions, such as on the roof of a building and they are relatively stable and not mobile in most scenarios. So, the dynamic topology discussed regarding the mesh networks is the cases that nodes leave or join the network.

When a node joins the network, it may change the distances between its two direct neighbors and may introduce conflicts among its neighbors. A previously conflict-free schedule may be turned into a conflicting schedule when the new member is considered. A functional correction scheduling algorithm must avoid these conflicts.

The leaving of one node can change the distances between its direct neighbors, too, but it never introduces conflicts into the schedule. However, the length of the schedule can be reduced if possible. The scheduling algorithm also needs to deal with such situations to improve network performance.

Here, we show how to deal with the two cases with the distributed ANPC1 as an example. The idea for the ANPC2, ONPC1, and ONPC2 are also similar.

When a node, say u, joins the network, node u will first set its state as UNASSIGNED. Then it collects the one-hop neighbor information and broadcasts a JOIN message. Nodes that have received the JOIN message will increase their node degree. The JOIN message is also forwarded to H-hop away, so that all nodes within $(H + 1)$-hop from node u will know about this new member. Nodes that have received this JOIN message will change their state to UNASSIGNED, and insert a new item into the *NeighborTable*. The scheduling algorithm is carried out by all nodes with state as UNASSIGNED after this process is finished.

When a node has detected that a neighbor node has left the network, it will decrease its node degree and broadcast a LEAVE message to H-hop away. The nodes that have created or received the LEAVE message will set their state as UNASSIGNED and remove the relevant item from the *NeighborTable*. The algorithm is carried out after this process is finished.

6.3 The implement of these algorithms

The implement of the access scheduling on the control channels can be done on the Media Access Control (MAC) sublayer of the Data Link Layer in OSI reference model. It can be realized with hardware or software programming to determine the length of the scheduling cycle and the access time-slots of the node. The building of the scheduling process can be viewed as a sequence of events as nodes joining the network. Generally, the new nodes well set its state as UNASSIGNED and listen for messages from neighbor nodes during a predefined time. If the new node does not receive any message, it knows it is the only node in the networks, and assign the first available time-slot on the scheduling cycle. The assignment information is broadcasted out frequently if there is no message for transmission. However, if the new nodes have received messages from its neighbor(s), the scheduling procedure and related message exchange are similar to the mechanism in the previous section.

7 Simulation results

The simulations are built in a 1000 m $\times$ 1000 m plane area. The mesh nodes are placed randomly in the area and the node number n is 50 or 150. Each node has the same transmission range, which varies from 130 m to 250 m to simulate different networks with different connectivity. When the node position and the transmission range are given, there is an edge between two nodes if their distance is no more than the transmission range, and thus the graph is built. Each mesh node has one fixed radio and one switchable radio. The IEEE 802.11a/b have provided 12/3 orthogonal channels and the number of available channels Q in the network is assumed to be 12 or 3 in the simulation. The fixed channel for each node is randomly selected among all available channels, since the fixed channel assignment is not the focus of this paper. Readers are guided to [12] for the assignment algorithm for the fixed channel in the network. For a given number of nodes and transmission range, 1,000 scenarios are built and the relevant graphs are all connected. The final results are the average of 1,000 scenarios.

The length of the scheduling cycle Δ is related to the topology of the network. For easy comparison and analysis, a new parameter *ratio* is introduced here as:

$$ratio = \frac{\Delta}{K}$$

in which K represents the maximum degree of nodes in the network.

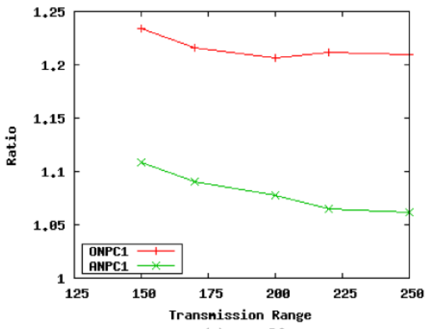
Since the basic ideas of ANPC2 and ONPC2 are similar to those of ANPC1 and ONPC1, the simulation results are omitted here, and we focus on the performance comparison of ANPC1 and ONPC1.

We simulate the ONPC1 and ANPC1 on the same topology with the same fixed channel assignment. In all these scenarios, the criterion chosen is the node degree. Since most of the current works are based on the assumption that $H = 1$, we test both cases $H = 1$ and $H = 2$ in the simulation. Figure 8 shows the results with $H = 1$. It can be seen that the performance of the ANPC1 is better than the ONPC1 in all scenarios.

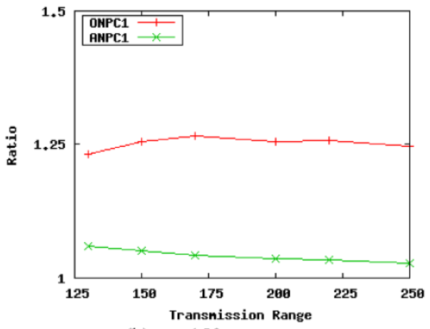
Although the length of the scheduling cycle is upper-bounded by the maximum number of $(H + 1)$-hop neighbors in the theoretical analysis, we can observe from the simulation results that the value of *ratio* is rather close to 1. This means that the average length of the scheduling cycle is close to the maximum node degree in the network. The results imply that a proper way to reduce the length of the cycle is to control the maximum node degree in the network through topology control methods, and thus the performance of the network can be improved significantly.

Figure 9 shows the results in the case that $H = 2$. The performance of the ANPC1 is better than the ONPC1 in all cases here, too. The length of the scheduling cycle increases significantly compared with the case $H = 1$, and this will weaken the system performance. A compromise may be necessary to balance the QoS requirement and system utilization in the mesh networks, which should be studied further.

To illustrate the influence of criterion on the scheduling algorithms, we have carried out the simulations with the selected criterion, node identification, node degree, and the number of $(H + 1)$-hop neighbors, accordingly. Figures 10 and 11 show the

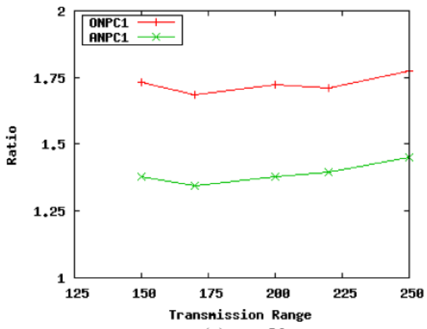


(a) $n = 50$.

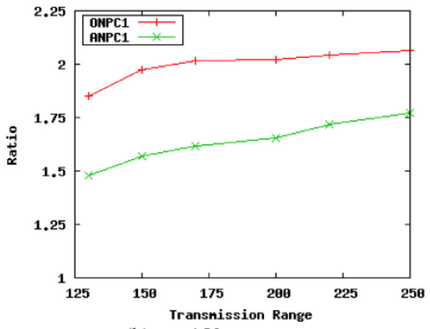


(b) $n = 150$.

Fig. 8 $H = 1$, $Q = 12$, and the node degree is the criterion

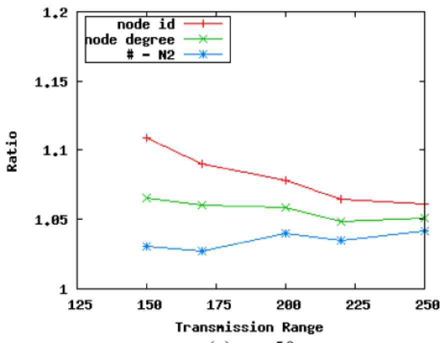


(a) $n = 50$.

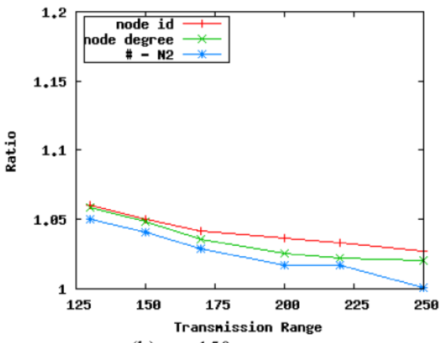


(b) $n = 150$.

Fig. 9 $H = 2$, $Q = 12$, and the node degree is the criterion.



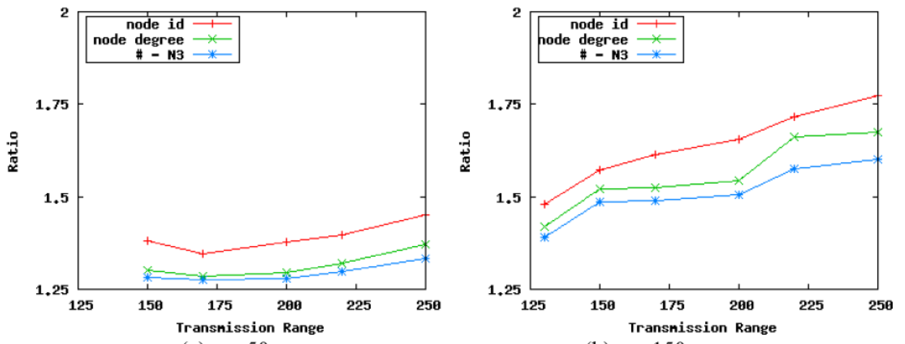
(a) $n = 50$.



(b) $n = 150$.

Fig. 10 $H = 1$, $Q = 12$, for the algorithm ANPC1.

final results in the cases that $H = 1$ and $H = 2$ with ANPC1. In these figures, the criterion of $(H + 1)$-hop neighbors is denoted as $\# - \text{N2}$ in the case that $H = 1$, and $\# - \text{N3}$ in the case that $H = 2$. As we can observe from these figures, the performance



(a) $n = 50$. (b) $n = 150$.

Fig. 11 $H = 2$, $Q = 12$, for the algorithm ANPC1

is the best in all cases as the criterion is selected as the number of $(H + 1)$-hop neighbors, while node degree is a little better than node identification. The node degree and $(H + 1)$-hop neighbor information are easy to obtain by limited local information exchange, and so all these parameters are feasible for use in scheduling channel access in wireless mesh networks.

8 Conclusions

In this paper, we have studied the scheduling problem for channel access in TDMA wireless mesh networks. The problem is to assign time-slots for each node in the network to access the channels, so that is guaranteed to be able to each node broadcast the control packet to any one-hop neighbor in one scheduling cycle. The objective is to minimize the length of the scheduling cycle. In this paper, we have originally proposed an H-hop interference mode to describe the case in which the interference range is larger than the transmission range. Then we have shown the NP-completeness of the scheduling problem. We also describe two special models for the mesh networks to employ the advantage of multiple control channels, the radio-based model, and the node-based model. In the radio-based model, the time-slots are assigned on two radios, and in the node-based model time-slots are assigned to nodes. Then we have proposed two heuristic algorithms for the two models, namely One Neighbor per Cycle 1 and 2 (ONPC1 and ONPC2), All Neighbors per Cycle 1 and 2 (ANPC1 and ANPC2). We also prove that the ANPC1 guarantees an upper bound on the total number of time-slots used as $\min(n, 2K)$ in some special cases for the radio-based model, and $\min(n, 4K - 2)$ for the node-based model. The fully distributed versions of the algorithms are given in this paper, also. The simplicity and locality make these algorithms rather suitable for wireless mesh networks. Simulation results also show that the performance of ANPC1 is much better than ONPC1.

References

1. Akyildiz I, Wang X, Wang W (2005) Wireless mesh networks: a survey. Comput Netw 47:445–487

2. Bertossi A, Bonuccelli M (1995) Code assignment for hidden terminal interference avoidance in multihop packet radio networks. IEEE/ACM Trans Netw 3(4):441–449
3. Battiti R, Bertossi A, Brunato M (2000) Distributed code assignment in multihop radio networks: object-oriented software simulations. In: IEEE SoftCOM'00, Rijeka, Croatia, October 2000
4. Cai Z, Lu M, Georghiades C (2003) Topology-transparent time division multiple access broadcast scheduling in multihop packet radio networks. IEEE Trans Veh 52(4):970–984
5. Chandra R, Bahl P (2004) MultiNet: connecting to multiple IEEE 802.11 networks using a single wireless card. In: IEEE INFOCOM, 2004
6. Chen J, Ting P, Lin C, Chen J (2004) A novel broadcast scheduling strategy using factor graphs and sum-product algorithm. In: GLOBECOM'04, vol 6, 2004, pp 4048–4053
7. Chlamtac I, Pinter S (1987) Distributed nodes organization algorithm for channel access in a multihop dynamic radio network. IEEE Trans Comput C-36(6):729–737
8. Cidon I, Sidi M (1989) Distributed assignment algorithm for multihop packet radio networks. IEEE Trans Comput 38(10):1353–1361
9. Hu L (1993) Distributed code assignments for CDMA packet radio networks. IEEE/ACM Trans Netw 1(6):668–676
10. Hung K, Yum T (1990) An efficient code assignment algorithm for multihop spread spectrum packet radio networks. In: IEEE GLOBECOM'90, vol 1, 1990, pp 271–274
11. Ju J, Li V (1999) TDMA scheduling design of multihop packet radio networks based on Latin squares. IEEE J Sel Areas Commun 17(8):1345–1352
12. Kyasanur P, Chereddi C, Vaidya N (2006) Net-X: system extensions for supporting multiple channels, multiple radios, and other radio capabilities. Technical Report, Department of Computer Science, University of Illinois at Urbana-Champaign
13. Lloyd E (2002) Broadcast scheduling for TDMA in wireless multi-Hop networks. In: Stojmenovic I (ed) Handbook of wireless networks and mobile computing. Wiley, New York, pp 347–370
14. Makansi T (1987) Transmitter-oriented code assignment for multihop packet radio networks. IEEE Trans Commun C-35(12):1379–1382
15. Maxim 2.4 GHz 802.11b Zero-IF Transceivers. http://pdfserv.maxim-ic.com/en/ds/MAX2820-MAX2821.pdf
16. Ngo C, Li V (2003) Centralized broadcast scheduling in packet radio networks via genetic-fix algorithms. IEEE Trans Commun 51:1439–1441
17. Ramanathan S (1999) A unified framework and algorithm for channel assignment in wireless networks. Wirel Netw 81–94
18. Ramanathan S, Lloyd E (1993) Scheduling algorithms for multi-hop radio Nnetworks. IEEE/ACM Trans Netw 166–177
19. Su Y, Su S, Li J (2004) Topology-transparent link activation scheduling schemes for multihop CDMA Ad Hoc networks. In: IEEE GLOBECOM'04, vol 6, 2004, pp 3563–3567
20. Tang Z, Garcia-Luna-Aceves J (1999) A protocol for topology-dependent transmission scheduling in wireless networks. IEEE Commun Netw Conf 3:1333–1337
21. Xiong N, Defago X, Jia X, Yang Y, He Y (2006) Design and analysis of a self-tuning proportional and integral controller for active queue management routers to support TCP flows. In: Proc IEEE Infocomm 2006, Barcelona, Spain, April 23–29, 2006
22. Xiong N, Yang L, Cao J, Yang Y, He Y (2008) PIDNN: an efficient and distributed flow control approach for multicast networks. ACM Trans Autonom Adaptive Syst, Special Issue on Adaptive Learning in Autonomic Communication, 2008
23. Xu K, Gerla M, Bae S (2002) How effective is the IEEE 802.11 RTS/CTS handshake in Ad Hoc networks. In: IEEE GLOBECOM'02, Taipei, Taiwan, Nov 2002, pp 72–76
24. Zhu C, Corson M (2001) A five-phase reservation protocol (FPRP) for mobile Ad Hoc networks. Wirel Netw 7:371–384

Hongju Cheng received the B.E. and M.E. degrees in EE from Wuhan University of Hydraulic and Electric Engineering, in 1997 and 2000, respectively, and the Ph.D. in Computer Science from Wuhan University in 2007. Since 2007, he has been with the Department of Computer Science, Fuzhou University, Fuzhou, China. His interests include mobile ad hoc networks, wireless sensor networks, and wireless mesh networks. Dr. Cheng has published about 10 papers in international journals and conferences.

Naixue Xiong is a research scientist in the Department of Computer Science at the Georgia State University, Atlanta, USA. His research interests include Communication Protocols, Network Architecture and Design, and Optimization Theory. Until now, he published about 90 research articles (including over 30 international journal articles). Some of his works were published in IEEE or ACM transactions and IEEE INFOCOM. He has been a Publicity Chair, PC member, and OC member of over 50 international conferences, and as a reviewer of over 30 international journals.

Larence T. Yang is a professor at the Department of Computer Science, St Francis Xavier University, Canada. His research includes high performance computing and networking, embedded systems, ubiquitous/pervasive computing and intelligence, autonomic and trusted computing, and computational science and engineering.

He has published approximately 240 papers in refereed journals, conference proceedings, and book chapters in these areas. He has been involved in more than 100 conferences and workshops as the program/general conference chair and more than 200 conference and workshops as a program committee member. He served as the vice-chair of IEEE Technical Committee of Supercomputing Applications (TCSA) until 2004, currently is in the executive committee of IEEE Technical Committee of Scalable Computing (TCSC), and of IEEE Technical Committee of Self-Organization and Cybernetics for Informatics, and of IFIP Working Group 10.2 on Embedded Systems. He is also the co-chair of IEEE Task force on Intelligent Ubiquitous Computing. In addition, he is the editor-in-chief of 10 international journals and a few book series. He is serving as an editor for 14 international journals. He has been acting as the author or editor/co-editor of 30 books from Kluwer, Springer, Nova Science, American Scientific Publishers, and John Wiley & Sons. He has received 3 Best Paper Awards including the IEEE 20th International Conference on Advanced Information Networking and Applications (AINA-06); Distinguished Achievement Award, 2005; Distinguished Contribution Award, 2004; Outstanding Achievement Award, 2002; Canada Foundation for Innovation Award, 2003; University Research/Publication/Teaching Award 00-02/02-04/04-06.

Young-Sik Jeong received the B.S. degree in mathematics and the M.S. and Ph.D. degrees in computer science and engineering from Korea University, Seoul, Korea in 1987, 1989, 1993, respectively. Since 1993, he has been with the Department of Computer Engineering, Wonkwang University, Iksan, Korea, where he is a professor, working in the areas of Grid Computing and Semantic Grid, Distributed and Mobile computing, and LBS middleware.