A Novel Link Scheduler for Multi Tx/Rx Wireless Mesh Networks

Jemie Effendy, Sieteng Soh
Department of Computing
Curtin University, WA, Australia
Email: {jemie.effendy@student., s.soh@}curtin.edu.au

Kwan-Wu Chin, He Wang, Luyao Wang
School of Electrical, Computer and Telecommunications Engineering
University of Wollongong, NSW, Australia
Email: kwanwu@uow.edu.au,{hw407, lw233}@uowmail.edu.au

***Abstract*—This paper considers the problem of deriving a Time Division Multiple Access (TDMA) link schedule for a single-channel, Multi-Transmit or Receive (MTR) Wireless Mesh Network (WMN). This problem is significant because a short schedule or superframe means the WMN has a higher network capacity. We first show that an existing solution, called Directed Edge Coloring (DEC), to channel assignment developed for WMNs with multiple transmit *and* receive capability is isomorphic to a TDMA schedule. This thus allows us to develop an efficient algorithm called DEC-MTR that derives a TDMA schedule or superframe for use in single-channel MTR WMNs. In addition, we propose a method to increase the number of links activated in each slot. We compare DEC-MTR against four state-of-the-art schedulers: DEC, Algo-1, HWF and MDF. Experiment results show that DEC-MTR produces equal superframe lengths with up to 23% higher capacity as compared to DEC, and up to 50% shorter superframe lengths as compared to state-of-the-art schedulers such as Algo-1, HWF, and MDF with up to 58% higher capacity.**

***Index Terms*—Directed Edge Color, Link Scheduler, Time Division Multiple Access, Wireless Mesh Network.**

I. Introduction

Wireless Mesh Networks (WMNs) have developed rapidly in recent years [1]. An interesting advance is to equip routers with multi transmit *or* receive (MTR) capability; see works such as [2] [3] and [4]. A key enabler of MTR is multiple-input multiple-output (MIMO) technology [5], where spatial multiplexing allows multiple transmissions/receptions over different antenna elements. Consequently, unlike conventional WMNs, those based on MIMO afford nodes multiple transmissions/receptions within k hops. These transmissions or receptions, however, must be coordinated by a link scheduler. Indeed, the network capacity of a MTR WMN is dependent on an efficient link scheduler. In particular, one that derives a schedule or superframe with the minimum number of slots. That is, we seek the minimal number of slots that repeat periodically such that all links are activated according to their demand. Figure 1 shows an example transmission/reception schedule for an MTR operating over a *single* frequency. We see that in slot-1, see Figure 1(a), node-2 transmits to all its neighbors simultaneously. In the subsequent slot, see Figure 1(b), it becomes a receiver. A key constraint is that nodes cannot transmit *and* receive at the same time. Thus, nodes operate their links in a half-duplex manner; see Figure 1(c).

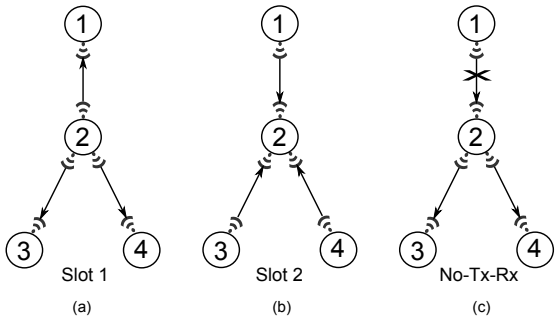


Fig. 1: Example of MTR transmissions

Interestingly, in [6], the authors removed the half-duplex constraint by assigning multiple channels to nodes. Their system requires nodes to have multiple off-the-shelf IEEE 802.11 radios. They showed that a node is able to transmit *and* receive to/from different neighbors at the same time as long as no incoming links share the same channel as outgoing links; see Figure 2. In this full duplex MTR WMN, the key problem is to determine the minimum number of colors such that incoming and outgoing links of each node have a different color. Note that the problem is similar to the well-known edge coloring problem but with the key difference that two or more incoming/outgoing links can share the *same* color. Although the system created in [6] allows for full duplex communications, it is not suitable for MIMO-based WMNs operating over a single frequency. Moreover, its applicability is limited in large scale WMNs given the small number of orthogonal channels in IEEE 802.11 systems.

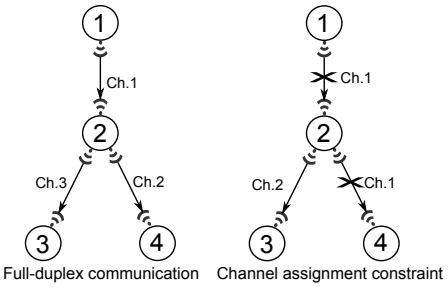


Fig. 2: Example of channel assignment

A key innovation in [6] is a novel channel assignment

algorithm called Directed Edge Coloring (DEC). Briefly, DEC first determines the chromatic number, denoted as ω, of an undirected graph G. Here, the chromatic number of a graph G is the smallest number of colors needed to color all the vertices of G so that no two neighboring vertices share the same color. For example, for the topology in Figure 3, the chromatic number is four, and the assigned *node* color is shown in the square bracket next to each node. Then, DEC finds the minimum *edge color* number ϖ that satisfies $\binom{\varpi}{\lfloor \varpi/2 \rfloor} \geq \omega$. After that for each node with the same node color, DEC assigns them an edge color set of size $\lfloor \frac{\varpi}{2} \rfloor$ *edge*, see round brackets in Figure 3. By definition, the condition $\binom{\varpi}{\lfloor \frac{\varpi}{2} \rfloor} \geq \omega$ ensures there are sufficient *edge* color sets that can be assigned to each node color. In Figure 3, we have $\binom{4}{\lfloor \frac{4}{2} \rfloor}$ = 6 *edge* color sets, each with $\lfloor \frac{4}{2} \rfloor = 2$ colors or channels. We see that node A and C have the same *node* color of 1. They are then assigned the *edge* color set $(1, 2)$. Finally, for each *directed* edge, DEC assigns the channel that is in the *edge* color set of the transmit end but not in the *edge* color set of the receive end. For example, edge AB is assigned channel 2 and edge BA is assigned channel 3. We see that the resulting coloring ensures incoming and outgoing links of a node do not share a color/channel. The main advantage of DEC is that it is simple, does not involve expensive computation and the coloring result is optimal. However, as mentioned, DEC is designed for multi-channel WMNs and is not suitable for *single* frequency, MIMO-based MTR WMNs.

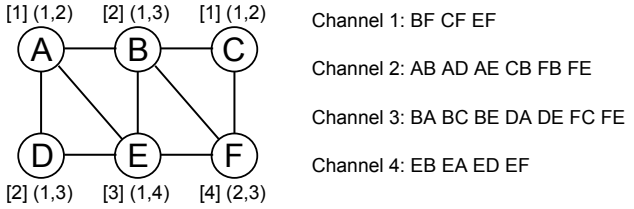


Fig. 3: An example of DEC.

In this paper, we show that one can transform the solution computed by DEC into a TDMA schedule that is suitable for a MIMO-based, MTR WMN operating over a *single* channel. In addition, our solution, called DEC-MTR, improves the transformed schedule by assigning color sets to nodes using a novel heuristic, as opposed to randomly, that activates the maximal number of links in each time slot. We remark that the said transformation may be possible for other wireless systems. In this respect, this paper is the first to demonstrate that a channel assignment algorithm can be exploited to construct a TDMA schedule. Indeed, exploring this connection is an interesting future research direction. Our experiment results show that when all links have a weight of one, i.e., they require at least one slot in the final schedule, DEC-MTR produces schedules with up to 50% shorter superframe lengths as compared to existing algorithms such as Algo-1 [7], HWF and MDF [8], and up to 15% higher network capacity as compared to DEC while achieving the optimal superframe length. When links have a higher weight, DEC-MTR increases the network capacity by up to 58% as compared to the methods in [9] and [8], and it outperforms DEC by up to 2% in terms of superframe length and by up to 10% with respect to network capacity.

The remainder of this paper is organized as follows. Section II reviews prior works. Section III presents the network model. Our solution is outlined in Section IV and experiment results are shown in Section V. We conclude in Section VI.

II. Related Works

In this section, we will only review directly relevant link scheduling algorithms for MTR WMNs. For other Medium Access Control (MAC) or link scheduling protocols, we refer the reader to [10].

Raman et al. [3] propose a link scheduling algorithm called 2P to maximize the capacity of MTR WMNs. 2P switches nodes between two states: SynTx and SynRx. When a node is in the SynTx state, it transmits on all links and all its neighbors must be in the SynRx state. Thus the topology must be bipartite. Chin et al. [7] propose an efficient algorithm, called Algo-1, which does not require the topology to be bipartite. Algo-1 approximates the MAX-CUT problem [11] to determine the maximal number of link activations in each time slot whereby links have a weight of one. Algo-1 generates a MAX-CUT every other slot. Nodes that have transmitted in one slot become receivers in the next slot. Later, in [9], the authors generalize Algo-1 to consider links with weight more than one. As confirmed by Dai et al. [8], Algo-1 generates schedules with longer superframe lengths, i.e., the total number of slots. A short superframe is advantageous to reduce the average end-to-end path delay as well as average response time [8]. Hence, the authors of [8] proposed two heuristic algorithms, called Heavy-Weight First (HWF) and Maximum-Degree First (MDF). HWF and MDF first generate a conflict graph, which in turn, is used to generate the Maximum Independent Set [11] to obtain links that can be activated in each time slot. Bao et al. [4] proposed a distributed link scheduler called Receiver-Oriented Multiple Access (ROMA). ROMA derives a random channel access schedule for each node using a hash function, current time, link weight and node ID. However, the aim of ROMA and our work is different as we aim to maximize network capacity. In [2], the authors propose a distributed version of 2P called JazzyMAC. JazzyMAC addresses the following limitations of 2P: (i) operates only over a bipartite network topology, and (ii) inability to adapt to dynamic traffic loads. These limitations are addressed by assigning each link a token. A node is allowed to transmit once it holds the token for all incident links.

To the best of our knowledge, no work has considered constructing a TDMA schedule, where all links operate over the same frequency, from the colors/channels assigned in a Multi-Channels Multi-Radios (MC-MR) WMN. In fact, single and multiple frequencies/radios works have proceeded in parallel. In the former, as we showed earlier, a key aim is to minimize the resulting superframe length. On the other hand, channel assignment research is focused on minimizing the number of

interfering links sharing a channel; i.e., they seek a solution that minimizes the number of contending links. This is also true for works that assume nodes are equipped with directional antennas; see [12]. Other design goals include connectivity, stability, fault tolerance, throughput/latency, fairness, and routing [13]. The channel assignment problem is particularly challenging given the limited number of orthogonal channels, and hence, it is important to use the minimum number of colors. We note that existing channel assignment methods that ensure full duplex communication, assuming sufficient number of channels, will assign a unique channel to each radio on a given node. However, as we pointed out in Section I, the nodes in MTR WMNs are able to receive/transmit simultaneously on multiple radios. Consequently, each incoming/outgoing link is not required to have a unique channel. In fact, doing so leads to unnecessarily long superframe lengths.

III. Preliminaries

We consider a TDMA-based MTR WMN as a multi-graph $G(V, E)$, where V corresponds to the set of nodes/routers and E represents the set of directed links/edges. Let w_{ij} indicate the weight of edge e_{ij}, meaning the link between node i and j is to be activated at least $w_{ij} \geq 1$ time slots. We remark that in a multi-graph, the weight of an edge is represented as multiple parallel links between two nodes, meaning E is a multi-set with w_{ij} copies of directed edge e_{ij}. We assume time is divided into time slots, and each slot contains a set of directed edges. A superframe S is defined as a grouping of slots, i.e., $S = (S_1, S_2, \ldots, S_{|S|})$, where S_c denotes the set of links activated in time slot c. Note, S is determined once and repeated periodically. Hence, a short superframe ensures a link is activated frequently, and thereby ensuring an MTR WMN has a high network capacity. Let ω be the chromatic number of G, and $V_\alpha \subseteq V$ be a set of vertices in G that use the color α, where $\alpha \in \{1, 2, \ldots, \omega\}$. Thus, if $\alpha, \beta \in \{1, 2, \ldots, \omega\}$ and $\alpha \neq \beta$, then $V_\alpha \cap V_\beta = \emptyset$. This also implies $|V_1| + |V_2| + \cdots + |V_\omega| = |V|$. Let $\mathcal{V}$ be a collection of vertices grouped according to their assigned color, i.e., $\mathcal{V} = \{V_1, V_2, \ldots, V_\omega\}$.

The problem at hand is to design a link scheduler for a single channel, MIMO-based MTR WMN that 1) minimizes $|S|$, and 2) ensures the network capacity $\Omega = \frac{\sum_{S_z \in S} |S_z|}{|S|}$ is maximized. Moreover, we require that a) each link e_{ij} is activated at least w_{ij} times, and b) because all links operate over the same frequency, a node cannot transmit to and receive from neighbors simultaneously.

IV. Deriving a TDMA Schedule

This section first shows how to transform the coloring solution from DEC into a TDMA schedule before outlining our novel link scheduler called DEC-MTR.

As noted in Section II, channel assignment schemes aim to minimize the number of contending links operating in the same channel by assigning interfering links to different channels. This approach is similar to deriving a TDMA schedule whereby interfering links occupy different time slots. This means one can assign links with a similar color/channel to the same slot. In other words, we convert 'channels' into 'slots'. Referring to Figure 3, and the assigned channels, we can derive a TDMA schedule for a *single* channel MTR WMN by simply scheduling links assigned with channel 1, 2, 3, 4 to slot 1, 2, 3, 4, respectively. Observe that in each slot, nodes operate all links in a half-duplex manner; i.e., the set of links assigned to each slot adhere to what we call the *no-tx-rx* constraint. We note that given the finite number of orthogonal channels, current channel assignment schemes, such as DEC, will always seek the minimal number of channels, which implies the resulting TDMA superframe length will also be minimal. At this point, we remind the reader that it is feasible to have two or more incoming or outgoing links share the same channel or time slot.

We are now ready to present DEC-MTR. The key idea is to use DEC [6] to construct a channel assignment solution which we then convert into a TDMA schedule. In addition, we propose a novel method to maximize the number of activated links in each time slot. To aid our presentation of DEC-MTR, we need a few notations and definitions. Let ϖ be the smallest integer that satisfies the inequality $\binom{\varpi}{\lfloor \frac{\varpi}{2} \rfloor} \geq \omega$. Denote cs_m to be the m-th slot or edge color set of size $\lfloor \frac{\varpi}{2} \rfloor$, where m is an index in the range $[1, \binom{\varpi}{\lfloor \frac{\varpi}{2} \rfloor}]$. Denote $\mathcal{C}$ to be a set containing all slot sets, i.e., $\mathcal{C} = \{cs_m \mid m \in [1, \binom{\varpi}{\lfloor \frac{\varpi}{2} \rfloor}]\}$.

Referring to Algorithm 1, DEC-MTR first uses any existing coloring algorithms, e.g., [6], to generate the chromatic index ω of graph G. We thus have the color of each vertex. Next, it constructs the set V_α, which contains the nodes with color α; see line $1-2$. DEC-MTR then computes ϖ; see line 3. After that, DEC-MTR generates the collection $\mathcal{C}$; see line 4. DEC-MTR then calls $CS2Vi()$ to assign a $cs_m \in \mathcal{C}$ to each $V_\alpha \in \mathcal{V}$; see line 5. Finally, DEC-MTR assigns each edge e_{ij} to a time slot c as follows. Assume $i \in V_\alpha$ and $j \in V_\beta$. The respective slot set of V_α and V_β is cs_m and cs_n. Then *FindSlot()* returns a c that satisfies $c \in cs_m \wedge c \notin cs_n$. Lastly, in line 8, DEC-MTR assign e_{ij} to slot S_c. By iterating through all edges, we thus have the superframe $S = (S_1, S_2, \ldots, S_\varpi)$. In other words, DEC-MTR generates a schedule with superframe length ϖ;

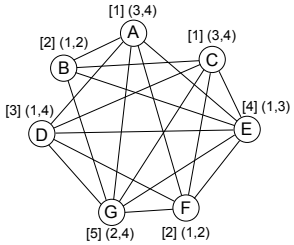
Algorithm 1: DEC-MTR

input : Multi-graph $G(V, E)$
output: link schedule $S = (S_1, S_2, \ldots, S_\varpi)$

1 $\omega \leftarrow$ `Color(`$G(V, E)$`)`
2 $\mathcal{V} \leftarrow \{V_1, V_2, \ldots, V_\omega\}$
3 Find the smallest ϖ such that $\binom{\varpi}{\lfloor \frac{\varpi}{2} \rfloor} \geq \omega$
4 $\mathcal{C} \leftarrow$ `GenSlotSets(`ϖ`)`
5 $\{(cs_m, V_\alpha)\} \leftarrow$ `CS2Vi(`$\mathcal{V}$`,`$\mathcal{C}$`,`ω`,`ϖ`)`
6 **for** *each* $e_{ij} \in E$ **do**
7 | c $\leftarrow$`FindSlot()`
8 | `AssignSlot(`*c*, e_{ij}`)`
9 **end**

As an example, consider the MTR WMN in Figure 4. DEC-MTR first calculates its chromatic index, which in this case is

$\omega = 5$. We thus have $\mathcal{V} = \{V_1 = \{A, C\}, V_2 = \{B, F\}, V_3 = \{D\}, V_4 = \{E\}, V_5 = \{G\}\}$; see Algorithm 1, line $1 - 2$. In this example, line 3 will yield $\varpi = 4$. DEC-MTR then generates $\mathcal{C} = \{cs_1 = \{1, 2\}, cs_2 = \{1, 3\}, cs_3 = \{1, 4\}, cs_4 = \{2, 3\}, cs_5 = \{2, 4\}, cs_6 = \{3, 4\}\}$; see Algorithm 1, line 4. DEC-MTR then calls $CS2Vi()$ to assign these slot sets to nodes. Assume $CS2Vi()$ assigns $cs_6, cs_1, cs_3, cs_2, cs_5$ to node sets V_1, V_2, V_3, V_4, V_5, respectively; see Algorithm 1, line 5. Then DEC-MTR allocates each edge into the slot c that is in cs_m but not in cs_n. For example, link AE is assigned to slot 4, and EA is assigned to slot 1.



Slot 1	Slot 2	Slot 3	Slot 4
BA EA	BA FE	AB CF	AB DE
BC EC	BC GA	AD CG	AE DF
BG EG	BE GC	AF EB	AF GB
DA FA	FA GD	AG ED	CB GE
DC FC	FC GE	CB EF	CE GF
DG FG	FD	CD EG	CF

Fig. 4: Example of DEC-MTR

A key problem with DEC is that it assigns slot sets randomly. We found this to be inefficient and yields sub-optimal network capacity. In particular, we found the resulting schedule is not optimized for *opportunistic* links. Specifically, in order to further increase network capacity, one can activate each link e_{ij} in more than w_{ij} time slots, called opportunistic activations [7] when possible. For example, in the example schedule shown in Figure 4, assume link AB has a weight of one and it is activated in both slot 3 and 4, meaning in slot 4, link AB is an opportunistic link. In other words, link AB is included with the sole purpose to increase the number of links activated in slot 4. To this end, DEC-MTR assigns slot sets heuristically with the aim to create more opportunistic links, and thus increasing the number of link activations in each slot, which in turn leads to a higher network capacity. In fact, the main novelty of DEC-MTR is the function $CS2Vi()$; see Algorithm 2.

We now show how $CS2Vi()$, see Algorithm 2, generates an assignment (cs_m, V_α) that maximizes the number of opportunistic links. Firstly, in line-1, $Selecs()$ constructs a set $\mathcal{C}'$ that contains ω slot sets $cs_m \in \mathcal{C}$ where each slot c in $\mathcal{C}'$ appears at most $\lceil \frac{\omega}{2} \rceil$ times. Then $CS2Vi()$ enumerates all slots that appear in $\mathcal{C}'$ and collects them in set $\mathcal{R}$; see line 2. Intuitively, as these slots have a high frequency of occurences, they are more likely to be assigned to links, meaning they will have a high number of links. The next steps, lines 3-11, aim to increase their occurences even more, if possible. For each slot $c \in \mathcal{R}$ that is used less than $\lceil \frac{\omega}{2} \rceil$ times, $CS2Vi()$ finds all color sets cs_m that contains slot c. Then for each cs_m, $CS2Vi()$ finds a disjoint set cs_n, i.e., $cs_m \cap cs_n = \emptyset$, and places the slot set pair (cs_m, cs_n) into set Q. Next, $CS2Vi()$ adds c to set cs_n if it does not create a superset of any other slot sets; see line $3 - 11$.

The next step is to assign each slot set pair (cs_m, cs_n) in Q to unassigned vertex sets $V_i \in \mathcal{V}$. The idea here is to greedily assign a slot set pair to $V_i, V_j \in \mathcal{V}$ with the maximum number of edges between them. This thus maximizes the number of links that are included in the slots of cs_m or cs_n. This idea is realized as follows; see lines 12-24. If both cs_m and cs_n are unassigned, $CS2Vi()$ randomly picks an unassigned node set V_α. However, if one of them has been assigned, $CS2Vi()$ finds the node set V_α assigned with cs_m or cs_n. Otherwise, if both slot sets in a given pair have been assigned, $CS2Vi()$ moves to the next pair. With a V_α in hand, $CS2Vi()$ calls *FindSet()* to find another vertex set V_β that contains nodes with the maximum incoming links from the vertices in V_α. $CS2Vi()$ then assigns the slot pair (cs_m, cs_n) to the V_α and V_β in question respectively; see line $12 - 24$. Lastly, for each remaining unassigned V_α, line 25 assigns it any unused slot set in $\mathcal{C}'$. $CS2VI()$ returns a set of assignments (cs_m, V_α) for use in DEC-MTR to produce schedule S.

Algorithm 2: CS2Vi

input : $\mathcal{V}$, $\mathcal{C}$, ω, ϖ
output: A set of assignments (cs_m, V_α)

1 $\mathcal{C}' \leftarrow$ `Selcs(`$\omega, \mathcal{C}$`)`
2 $\mathcal{R} \leftarrow$`Enumslot(`$\mathcal{C}'$`)`
3 **for** *each* $c \in \mathcal{R}$ *that appears in* $\mathcal{C}'$ *for less than* $\lceil \frac{\omega}{2} \rceil$ **do**
4 **for** *each* $cs_m \in \mathcal{C}'$ *where* $c \in cs_m$ **do**
5 Select $cs_n \in \mathcal{C}'$ such that $cs_m \cap cs_n = \emptyset$
6 $Q \leftarrow Q \cup (cs_m, cs_n)$
7 **if** $cs_n \cup c$ *is not a superset of any other slot set* **then**
8 $cs_n \leftarrow cs_n \cup c$
9 **end**
10 **end**
11 **end**
`// Assign` (cs_m, cs_n) `to` $V_\alpha, V_\beta \in \mathcal{V}$
12 **for** *each pair* $(cs_m, cs_n) \in Q$ **do**
13 **if** *both* $cs_m \wedge cs_n$ *unassigned* **then**
14 Randomly select an unassigned $V_\alpha \in \mathcal{V}$
15 **else**
16 **if** $cs_m \vee cs_n$ *is not assigned* **then**
17 Find the V_α assigned with cs_m or cs_n
18 **else**
19 Move to the next pair in Q
20 **end**
21 **end**
22 $V_\beta \leftarrow$`FindSet(`$\mathcal{V}$`)`
23 Generate (cs_m, V_α) and (cs_n, V_β)
24 **end**
25 Randomly assign each unallocated V_α with a remaining slot set $cs_m \in \mathcal{C}'$

We now show how $CS2Vi()$ assigns each $cs_m \in \mathcal{C}$ to one node set $V_\alpha \in \mathcal{V}$ in the topology shown in Figure 4. $CS2Vi()$ first selects $\omega = 5$ different slot sets from $\mathcal{C}$ where each slot appears at most $\lceil \frac{\omega}{2} \rceil = 3$ times. Thus we have $\mathcal{C}' = \{cs_1, cs_2, cs_3, cs_5, cs_6\}$ with $\mathcal{R} = \{1, 2, 3, 4\}$; see

Algorithm 2, line $1-2$. Note that slots $1, 2, 3$ and 4 appear $3, 2, 2, 3$ times respectively. As slot 2 and 3 appear less than three times, $CS2Vi()$ then seeks opportunities to add slot 2 and 3 to other slot sets to generate more opportunistic links. Starting with slot 2, $CS2Vi()$ first selects $cs_1 \in \mathcal{C}'$ because it contains slot 2. Then $CS2Vi()$ selects cs_6 because $cs_1 \cap cs_6 = \emptyset$, and places (cs_1, cs_6) into set Q. As adding slot 2 to cs_6 makes it a superset of cs_5, $CS2Vi()$ will not add slot 2 to cs_6. Next $CS2Vi()$ selects the slot set cs_2 that contains slot 3 and cs_5 where $cs_2 \cap cs_5 = \emptyset$. $CS2Vi()$ will also not add slot 3 to cs_5 because adding slot 3 to cs_5 creates a superset of cs_6. Thus $CS2Vi()$ produces $Q = \{(cs_1, cs_6), (cs_2, cs_5)\}$; see Algorithm 2, line $3-11$. Next, for pair (cs_1, cs_6), $CS2Vi()$ randomly selects $V_\alpha = V_2 = \{B, F\}$ and finds the corresponding set $V_\beta = V_1 = \{A, C\}$ that contains nodes with the maximum incoming links from the vertices in V_2. Thus, slots in cs_1 are assigned to links in V_2 and those in cs_6 are assigned to V_1. For pair (cs_2, cs_5), $CS2Vi()$ assigns cs_2 to V_4 and cs_5 to V_5; see Algorithm 2, line $12-24$. $CS2Vi()$ assigns the remaining set cs_3 to V_3. Using the set of assignments (cs_m, V_α) produced by $CS2Vi()$, DEC-MTR generates the link schedule shown in Figure 4 with a superframe length of 4 slots.

In the remainder of this section we outline the running time of $CS2Vi()$ and DEC-MTR. Note, due to space constraint, we will only focus on the most expensive steps.

Proposition 1. *CS2Vi() has a running time of* $\mathcal{O}((\Delta + 1)^2|V|^2)$*, where* Δ *is the maximum node degree.*

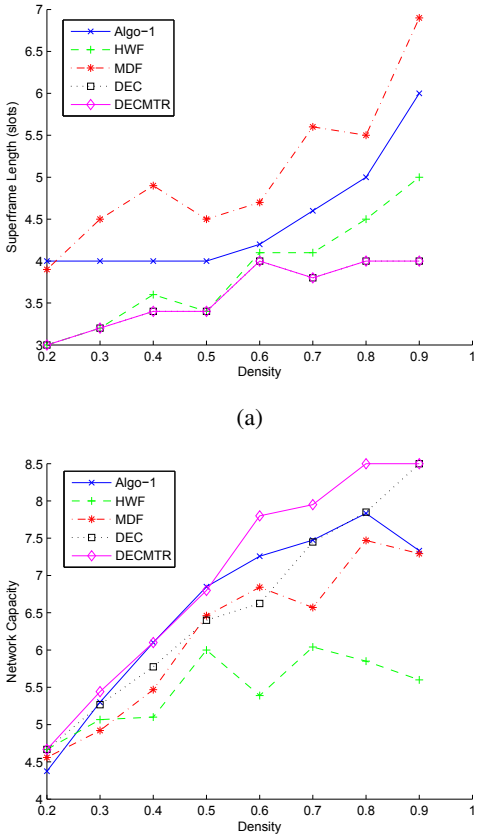
Proof. The most expensive steps can be shown to be Lines 12-24. The size of Q is bounded by $\frac{\omega}{2}$, which in turn is no more than $\Delta+1$. The function $FindSet()$ iterates through the collection $\mathcal{V}$, which has ω subsets and is upper bounded by $\mathcal{O}(\Delta + 1)$. For each subset, $FindSet()$ may need to iterate up to $\mathcal{O}(|E|)$ links to determine the number of incoming links from V_α. As $|E|$ is bounded by $|V|^2$, we thus have CS2Vi()'s stated running time. □

Proposition 2. *DEC-MTR's time complexity is* $\mathcal{O}(|V|^2(\Delta + 1)^2)$.

Proof. Line-3 of DEC-MTR takes $\mathcal{O}((\Delta + 1)log(\Delta + 1))$, assuming binary search. Line-4 takes $\mathcal{O}(\lfloor \frac{\omega}{2} \rfloor + \binom{\omega}{\lfloor \frac{\omega}{2} \rfloor})$ using the method in [14]. From [6], $\binom{\omega}{\lfloor \frac{\omega}{2} \rfloor} \leq 2log(\omega)$. Given that $\omega \leq \Delta + 1$, line-4 takes $\mathcal{O}(log(\Delta + 1))$. From Proposition 1, we know line-5 takes $\mathcal{O}((\Delta+1)^2|V|^2)$. Lastly, lines 6-9 take at most $\mathcal{O}(|E|(\Delta + 1)^2)$. Using the bound for $|E|$, we thus have $\mathcal{O}(|V|^2(\Delta+1)^2)$. We see that lines 5, and 6-9 have the worst running time, which prove our claim. □

V. Evaluation

In this section, we evaluate the performance of DEC-MTR against Algo-1 [7] [9], MDF [8], HWF [8] and DEC [6] in terms of superframe length and network capacity. We have implemented all algorithms in Python and conducted three different experiments. For each experiment, we use Python's "network" library package to generate ten random networks. For each schedule produced by each of the five algorithms, we compute its superframe length (in terms of slots) and network capacity; i.e., average number of links activated in each slot, including opportunistic links for DEC, MDF, HWF and Algo-1. Note that each randomly generated network contains at least $|V|$ links and we consider only connected networks.



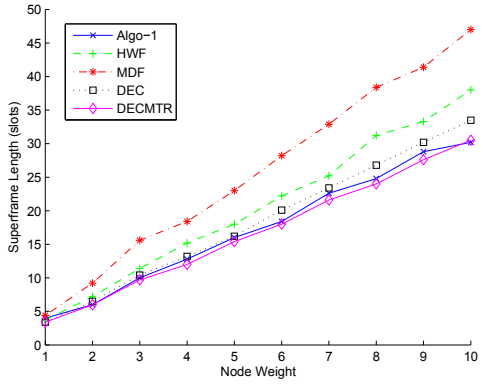
(a)

(b)

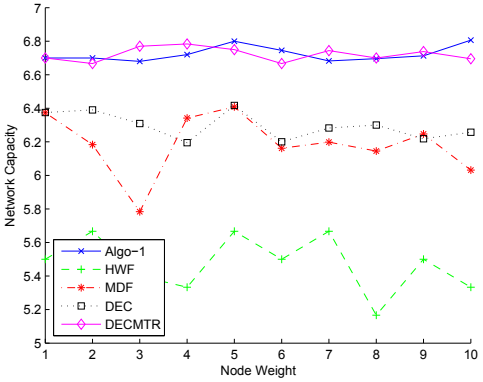
Fig. 5: Performance under different node densities

We first study network density. Here, network density is defined as $\delta = \frac{|E|}{|V| \times (|V|-1)}$, meaning a network with $\delta = 1$ is fully connected. Figure 5 shows the average result from 10 runs on WMNs with six nodes, where all links have a weight of one. From Figure 5(a), we see DEC-MTR and DEC has the same superframe length. On average, this is 33%, 20%, and 39% shorter than those produced by Algo-1, HWF, and MDF respectively. The results show DEC-MTR has better performance over Algo-1, MDF and HWF. In terms of capacity, see Figure 5(b), DEC-MTR outperforms DEC by up to 15%. It improves network capacity by up to 10%, 21%, and 40% as compared to Algo-1, MDF, and HWF, respectively. Notice that for $\delta = 0.5$, Algo-1 produces a schedule with a slightly better capacity, but with longer superframes.

Next we study the impact of link weight. We evaluate the performance of DEC-MTR and all other approaches on WMNs with 6-nodes and network density $\delta = 0.5$. We vary the

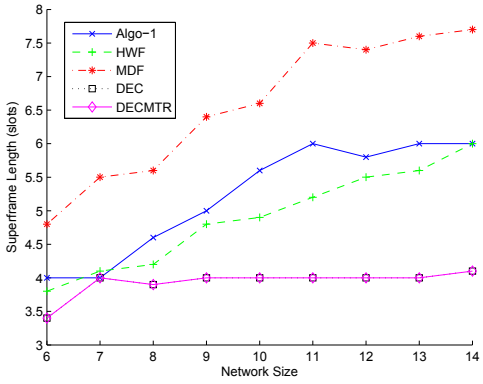


(a)

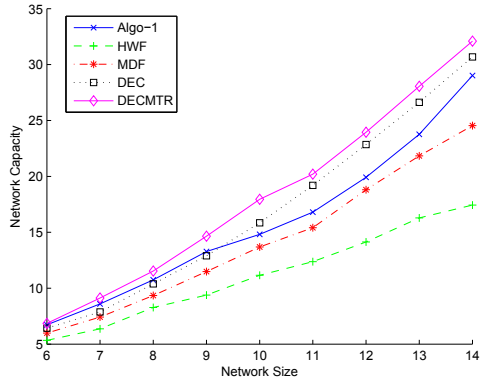


(b)

Fig. 6: Performance under different link weights



(a)



(b)

Fig. 7: Performance under different network sizes

weight of each link from 1 to 10. For DEC, we produce the schedule by repeating the algorithm w_{ij} times where w_{ij} is the link weight. As shown in Figure 6(a), DEC-MTR outperforms DEC with up to 2% shorter superframe length. Notice that DEC-MTR significantly outperforms Algo-1, HWF, and MDF. Specifically, it reduced the superframe length by up to 35%, 50%, and 50% respectively. In terms of capacity, as shown in Figure 6(b) DEC-MTR produces up to 10% higher capacity as compared to DEC, and up to 7%, 40%, and 30% as compared to Algo-1, HWF, and MDF respectively. Notice that for node weight of 2, Algo-1 produces the best capacity, while also produces the same superframe length as DEC-MTR.

Finally, we study the impact of network sizes ranging from 6 to 14. Links have a weight of one and the network density is 0.5. Referring to Figure 7(a), DEC-MTR and DEC produce the same superframe length irrespective of the network size. They outperform Algo-1, HWF, and MDF with up to 32%, 33%, 45% shorter superframe lengths, respectively. Further, as shown in Figure 7(b), DEC-MTR has up to 23% higher capacity as compared to DEC. DEC-MTR also significantly outperforms Algo-1, HWF, and MDF, recording a higher average capacity of 12%, 58%, and 33%, respectively.

VI. Conclusion

This paper has presented a link scheduler, called DEC-MTR, that exploits a novel observation to create a TDMA schedule for single channel MTR WMNs. It produces a schedule with a higher network capacity and shorter superframe lengths as compared to the following state-of-the-art algorithms: Algo-1 [7] [9], HWF [8], MDF [12], and DEC [6]. More importantly, when links have a weight of one, DEC-MTR produces a schedule with the shortest superframe length. In particular, DEC [6] has been proven to use the optimal number of channels. As a future work, we plan to develop a distributed algorithm version of DEC and DEC-MTR.

References

[1] P. H. Pathak and R. C. A. C. A. C. Dutta, "A survey of network design problems and joint design approaches in wireless mesh networks," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 3, pp. 396–428, 2011.

[2] S. Nedevschi, R. K. Patra, S. Surana, S. Ratnasamy, L. Subramanian, and E. Brewer, "An Adaptive, High performance MAC for Long-Distance Multihop Wireless Networks," in *ACM Mobicom*, San Francisco, USA, Sept. 2008.

[3] B. Raman and K. Chebrolu, "Design and Evaluation of a New MAC Protocol for Long-Distance 802.11 Mesh Networks," in *ACM Mobicom*, Cologne, Germany, Aug. 2005.

[4] L. Bao and J. Garcia-Luna-Aceves, "Receiver-Oriented Multiple Access in Ad Hoc Networks with Directional Antennas," *Wireless Networks*, vol. 11, pp. 67–79, 2005.

[5] S. Chu and X. Wang, "Opportunistic and cooperative spatial multiplexing in MIMO ad hoc networks," *IEEE/ACM ToN*, vol. 18, no. 5, pp. 1610–1623, 2010.
[6] P. Dutta, S. Jaiswal, D. Panigrahi, and R. Rastogi, "A New Channel Assignment Mechanism for Rural Wireless Mesh Networks," in *IEEE INFOCOM*, Phoenix, USA, Apr. 2008.
[7] K.-W. Chin, S. Soh, and C. Meng, "Novel Scheduling Algorithms for Concurrent Transmit/Receive Wireless Mesh Networks," *Computer Networks*, vol. 56, pp. 1200–1214, 2012.
[8] H.-N. Dai, S. C. Liew, and L. Fu, "Link Scheduling in Multi-Transmit-Receive Wireless Networks," in *IEEE 36th Conference on Local Computer Networks (LCN)*, Bonn, Germany, Oct. 2011.
[9] K.-W. Chin, S. Soh, and C. Meng, "A Novel Scheduler for Concurrent Tx/Rx Wireless Mesh Networks with Weighted Links," *IEEE Communications Letters*, vol. 16, pp. 246–248, 2012.
[10] O. Bazan and M. Jaseemuddin, "A Survey on MAC Protocols for Wireless Ad Hoc Networks with Beamforming Antennas," *IEEE Communications Surveys & Tutorials*, vol. 14, no. 2, pp. 216–239, 2012.
[11] V. V. Vazirani, *Approximation algorithms*. Springer, 2001.
[12] H.-N. Dai, K.-W. Ng, and M.-Y. Wu, "Channel allocation in wireless networks with directional antennas," *Journal of Sensor and Actuator Networks*, vol. 2, no. 2, pp. 213–234, 2013.
[13] W. Si, S. Selvakennedy, and A. Y. Zomaya, "An overview of channel assignment methods for multi-radio multi-channel wireless mesh networks," *Journal of Parallel and Distributed Computing*, vol. 70, pp. 505–524, 2010.
[14] P. Eades and B. McKay, "An algorithm for generating subsets of fixed size with a strong minimal change property," *Information Processing Letters*, vol. 19, pp. 131–133, 1984.