1-1-2012

A novel scheduler for concurrent Tx/Rx wireless mesh networks with weighted links

Kwan-Wu Chin
University of Wollongong, kwanwu@uow.edu.au

Sieteng Soh
Curtin University Of Technolgoy, s.soh@curtin.edu.au

Chen Meng
University of Wollongong, cm488@uowmail.edu.au

A novel scheduler for concurrent Tx/Rx wireless mesh networks with weighted links

Abstract

This paper considers the NP-hard problem of scheduling weighted links in concurrent transmit/receive wireless mesh networks. The problem generalizes existing works to links with weight wij ≥ 1. We propose an O(|V|2) algorithm, where V is the set of routers, that is orders of magnitude faster than computationally intensive approaches that use the well-known Goemans-Williamson (GWA)'s maximum cut algorithm and also brute-force. Our algorithm generates schedules, on average, with at most 3% and 9% fewer links than the GWA and brute-force approaches respectively.

Keywords

novel, scheduler, weighted, concurrent, links, tx, rx, wireless, mesh, networks

Disciplines

Physical Sciences and Mathematics

Publication Details

K. Chin, S. Soh & C. Meng, "A novel scheduler for concurrent Tx/Rx wireless mesh networks with weighted links," IEEE Communications Letters, vol. 16, (2) pp. 246-248, 2012.

A Novel Scheduler for Concurrent Tx/Rx Wireless Mesh Networks with Weighted Links

Kwan-Wu Chin, Sieteng Soh and Chen Meng

***Abstract*—This paper considers the NP-hard problem of scheduling weighted links in concurrent transmit/receive wireless mesh networks such that the resulting superframe length is minimized. We propose an algorithm that is orders of magnitude faster than approaches based on the well-known Goemans-Williamson's maximum cut algorithm and also brute-force. In fact, our algorithm has a time complexity of $O(|V|^2)$, where V is the set of routers, and has comparable superframe lengths and generates no more than 9% and 3% fewer links as compared to the brute-force and Goemans-Williamson approach respectively.**

***Index Terms*—Scheduler, Weighted Links, Multiple Transmit/Receive, Wireless Mesh Networks**

I. Introduction

A key development in Wireless Mesh Networks (WMNs) is equipping routers with multiple radios and connecting each one to a directional antenna [5][4][7]. A similar approach is installing WiFi arrays on each router [3]. As a result, a router with N radios is capable of transmitting *or* receiving N distinct packets simultaneously. Notably, in [3], Kakumanu et al. demonstrated a router capable of transmitting close to 600 Mbps using 15 Wi-Fi radios, and the authors of [5] and [4] have successfully deployed WMNs with concurrent transmit/receive (Tx/Rx) capability in a number of developing countries. In particular, these low cost, IEEE 802.11-based WMNs span tens of kilometers to provide critical services, e.g., health, to rural villages.

Link scheduling is a fundamental problem in these WMNs. Specifically, a scheduler is responsible for deriving a superframe that maximizes network capacity whilst adhering to the following constraints:

1) A node can transmit on all links or receive on all links; a node is not allowed to receive on some links and transmit on the remaining links at the same time.
2) Each node is only allowed to form a maximum of k concurrent links, where k corresponds to the number of antenna elements.
3) Each link e is activated at least w_e slots in a given superframe.

The first constraint is a manifestation of side lobes, e.g., 8 dBi side lobes [4], and high transmission power, e.g., 400mW [5] – both of which cause high packet loss when a node transmit *and* receive on more than one radio simultaneously.

K-W Chin and C. Meng are with the School of Electrical, Computers, and Telecommunications Engineering, University of Wollongong, NSW, Australia. Email: {kwanwu,cm488}@uow.edu.au

S. Soh is with the Dept. of Computing, Curtin University of Technology,Perth, WA, Australia. Email:S.Soh@curtin.edu.au

To illustrate the link scheduling problem, consider the "2boxes" topology shown in Figure 1, and the links scheduled in each time slot, where $k = 5$. The resulting schedule is then realized using the 2P Medium Access Control (MAC) protocol [5]. As we can see, the resulting schedule adheres to all constraints. In addition, unlike other WMNs, we see multiple point-to-multipoint or multipoint-to-point transmissions. Also, nodes that are transmitters in slot i becomes receiver in slot $i + 1$. Intuitively, the problem can be solved using graph coloring. However, as we showed in [1], graph coloring yields a superframe length of eight slots for this topology. In addition, we showed that obtaining the optimal link schedule is equivalent to solving the NP-hard, max-cut problem. Interestingly, in time slot 3 and 4, it is possible to establish *opportunistic* links. Specifically, we can increase network capacity further by including links e_{AE}, e_{BD}, e_{BF} and e_{CE} in time slot 3, and conversely, links e_{EA}, e_{DB}, e_{FB} and e_{EC} in time slot 4.

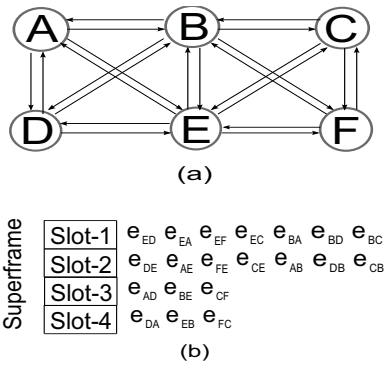

Fig. 1. An example WMN. (a) the 2boxes topology, (b) link schedule, not including opportunistic links.

In a previous work [1], and also others [5][4], all links have a weight of $w_i = 1$, and hence, they assume uniform traffic distribution. That is, they did not consider links with varying amount of traffic, meaning each link will require different number of time slots in order to have a service rate that commensurates with the corresponding link load. This fact thus motivates us to propose a scheduler that considers weighted links. In fact, the scheduler outlined in Section II generalizes the one proposed in [1].

II. Weighted Scheduler

We model a WMN with weighted links as a multi-graph $G(V, E)$, where V and E correspond to the set of nodes/routers and directed links respectively. The number of edges connecting node i and j is denoted by its weight w_{ij}.

Specifically, E is a multi-set where $w_{ij} = k$ corresponds to k copies of directed edge e_{ij} in E. The problem is then to derive a minimal superframe, where each slot denotes one transmission, that ensures all links in E are scheduled at least once. Note, we say *at least* because opportunistic links may be added into a slot. As mentioned, our problem can be cast as the well known NP-complete MAX-CUT problem. That is, if $S \subseteq V$ and $\overline{S} = V \setminus S$, then the problem in each time slot is to maximize the weight of the cut $(S : \overline{S})$,

$$w(S : \overline{S}) = \sum_{i \in S, j \in \overline{S}} w_{ij} \quad (1)$$

In our case, all link weights w_{ij} are set to one because a link is only activated once in a given slot. Hence, the weight of the cut is simply the maximum number of edges.

Algo-1 shows our proposed greedy scheduler. It outputs S and $\overline{S}$ for time slot i, and subsequently, for $i + 1$ – recall that transmitters in time slot i becomes receiver in $i + 1$. The function $Gselect(.)$ returns a node $v \in V$ with $\Delta_v > 0$ and also maximizes,

$$\Delta_v = \sum_{v_j \in N_2(v)} w_{i,j} - \sum_{v_j \in N_1(v)} w_{j,i} \quad (2)$$

where $N_1(v)$ denotes the set of nodes in S that have out-link to v, and $N_2(v)$ is the set of nodes in $V \setminus S$ that have in-link from v. In words, for a given node v, the term on the right denotes the number of links incident on v that originated from nodes in S, whilst the term on the left represents the total links originating from node v to neighbors not in S. $Gselect(.)$ then returns the node v that has the highest number of links to nodes in $V \setminus S$ less any links from S. Note, if there are more than one node with equal value, $Gselect(.)$ returns the node with the smallest $\sum_{v_j \in N_1(v)} w_{j,i}$ value; nodes with equal value are treated arbitrarily. It can be shown that Algo-1 has a time complexity of $O(|V|^2)$ as line 2–9 of Algo-1 and $Gselect(.)$ consider up to $O(|V|)$ nodes.

We will now show how Algo-1 determine the schedule for the 2boxes topology, with $w_{AD} = w_{BE} = w_{CF} = 2$. Figure 2(a) shows the Δ value for each node. Initially, the $\sum_{v_j \in N_1(v)} w_{j,i}$ term of Eqn. 2 is zero for all nodes; e.g., $\Delta_A = (2 + 1 + 1) - 0 = 4$; Algo-1 proceeds as follows:

- *Line 2*: Referring to Figure 2(a), $Gselect(.)$ will choose node B as it has the highest Δ value.
- *Line 3 – 9*: $S = \{B\}$ and $\overline{S} = \{A, C, D, E, F\}$.
- *Line 10*: $V = V \setminus \{B\}$.

At this point, the Δ value of all nodes, except for node-B, are updated as per Equ. 2; see Figure 2(b). For example, for node E we have $\Delta_E = (1 + 1 + 1 + 1) - 2$, which corresponds to the link it has to node D, A, C and F minus the two incoming links from node-B.

- *Line 2*: At this stage, $Gselect(.)$ is able to choose A or C because their $\sum_{v_j \in N_1(v)} w_{j,i}$ term is smaller than that of node-E; viz. 2 versus 1. In this case, we choose them arbitrarily. Assume it's node-A.
- *Line 3 – 3*: $S = \{B, A\}$ and $\overline{S} = \overline{S} - \{A\}$.
- *line 5 – 9*: $\overline{S} = \{C, D, E, F\}$.
- *Line 10*: $V = V \setminus \{A\}$.

input : G(V, E)
output: Schedule represented as (S, $\overline{S}$)

```
1  S = S̄ = {};
2  while (V ≠ {} AND v_i=GSelect(V)) do
3  |   S = S ∪ {v_i};
4  |   S̄ = S̄ \ {v_i};
5  |   foreach v_j ∈ V AND v_j ≠ v_i do
6  |   |   if e_ij ∈ E then
7  |   |   |   S̄ = S̄ ∪ {v_j};
8  |   |   end
9  |   end
10 |   V = V \ {v_i};
11 end
   /* Remove scheduled links                    */
12 foreach v_i ∈ S AND v_j ∈ S̄ do
13 |   if e_ij ∈ E then
14 |   |   E = E \ {e_ij}
15 |   end
16 end
```

Algorithm 1: Weighted greedy scheduler. The output consists of nodes in S and $\overline{S}$, where nodes in S transmit to those in $\overline{S}$, and vice-versa, in time slot i and $i + 1$ respectively.

In the next iteration, $Gselect(.)$ considers the new Δ value of each node; see Figure 2(c), and Algo-1 proceeds one last time as follows:

- *Line 2*: $Gselect(.)$ chooses node-C.
- *Line 3 – 3*: $S = \{B, A, C\}$ and $\overline{S} = \overline{S} - \{C\}$.
- *line 5 – 9*: $\overline{S} = \{D, E, F\}$.
- *Line 10*: $V = V \setminus \{C\}$.

After node C is included in S, we have $\Delta_D = -2$, $\Delta_E = -2$, and $\Delta_F = -1$. As all values are negative, $Gselect(.)$ returns false, which causes Algo-1 to return $S = \{B, A, C\}$ and $\overline{S}\{D, E, F\}$. The nodes in S are then set to transmit in time slot-1, and those in $\overline{S}$ become transmitters in time slot-2. Lastly, as per line 12 – 16, Algo-1 removes links connecting nodes in S and $\overline{S}$, and vice-versa from E. The resulting topology is shown in Figure 2(d). The above steps are then repeated to yield $S = \{B, D, F\}$ and $\overline{S} = \{A, E, C\}$, where the corresponding links are scheduled in time slot 3 and 4 respectively. We like to note that in our implementation, in each slot, we also any opportunistic links that do not violate the concurrent transmit *or* receive constraint.

III. Simulation Methodology

Our experiments are conducted in Matlab, where we used MatGraph [6] to generate graphs with 20 nodes. Each node has degree ranging from two to nine. Links have a weight ranging from one to five. Our results are an average of 10 simulation runs, each with a different topology and new set of link weights. In each experiment, after deriving the schedule of a given topology, we compute the (i) *Superframe Length*, which corresponds to the number of slots required to activate each link as per its weight, and (ii) *Number of activated*

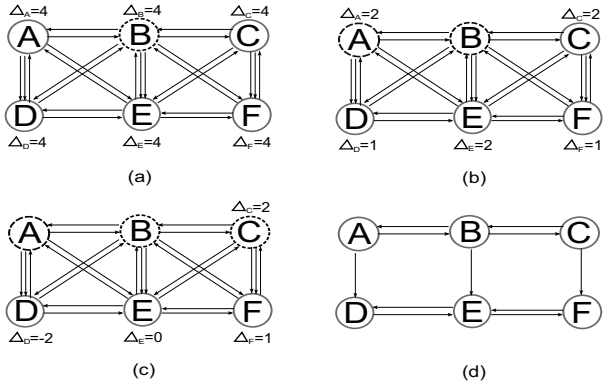

Fig. 2. Subfigures (a)–(c) correspond to scheduling steps for slot 1 and 2. Sub-figure (d) denotes the remaining links to be scheduled in slot 3 and 4. Nodes with dotted line have been included in S.

links, including *opportunistic*) in each slot, and (iii) *End-to-End Delay*, where we performed 100 random transmissions between different source-destination pairs over their respective shortest path. However, in this paper, due to space limitation, we only report on the superframe length and activated links.

We compare Algo-1 to two sets of approaches. The first is graph coloring, where we used the "optimal" algorithm included in Matgraph's toolkit. The second set approaches comprise of algorithms that derive the maximum cut of a graph, and includes (a) Vazirani's approximation [9], (b) Goemans-Williamson Algorithm (GWA)[2], as implemented in [8], and (c) brute force, which yields the maximum cut. It's important to note that these approaches are not designed for scheduling links in concurrent Tx/Rx WMNs. To this end, we extend them as follows. Derive a graph $G(V', E')$ from $G(V, E)$ by setting $V' = V$, and copying *one* link from E into E' that connects node $v_i \rightarrow v_j$, and vice-versa, and removing said links from E. Schedule $G(V', E')$ using one of the said approach; e.g., use GWA to derive S, and $\overline{S}$. We then repeat the process until E is empty.

IV. Results

Figure 3 and 4 show the average sueprframe length and number of activated links per superframe respectively. We see that in both cases, Algo-1 ranks third in terms of performance. However, we like to point out that although GWA has better performance, it has significantly higher running time. In fact, Algo-1 runs 80% to 92% faster than GWA, and only yields superframes with one or two slots more than that of GWA. In addition, Algo-1 enables 50% more links that graph coloring, only generate 2% to 9% and 3% fewer links than brute force and GWA respectively. Moreover, from Figure 3, we see that Algo-1 is capable of generating superframe lengths that are equal or 1-2 slots longer than brute force and GWA.

V. Conclusion

This paper has presented an efficient algorithm for scheduling weighted links in concurrent Tx/Rx WMNs, and advantageously, it has comparable performance, in terms of superframe length and number of activated links, to more computational intensive methods. Moreover, unlike [5], it works on general topologies.

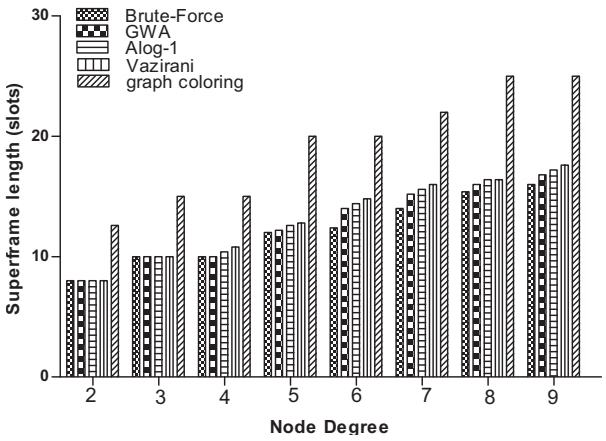

Fig. 3. Average superframe length versus node degree.

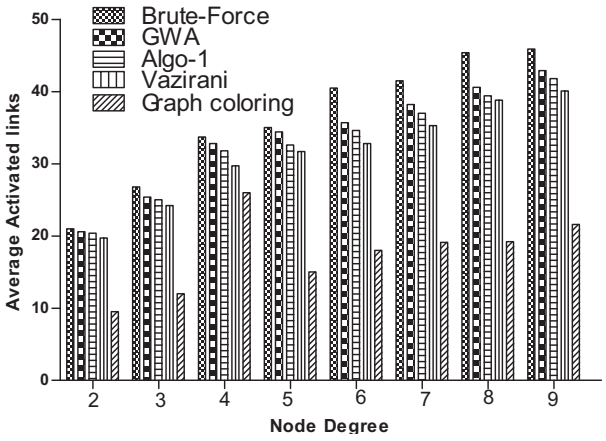

Fig. 4. Average number of active links in each slot versus node degree.

References

[1] K.-W. Chin, S. Soh, and C. Meng. A novel spatial TDMA scheduler for concurrent transmit/receive wireless mesh networks. In *The 24th IEEE Conference on Advanced Information and Networking Applications (IEEE AINA)*, Perth, Australia, Apr. 2010.

[2] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995.

[3] S. Kakumanu and R. Sivakumar. Glia: A practical solution for effective high data rate WiFi-arrays. In *ACM Mobicom*, China, Sept. 2009.

[4] R. Patra, S. Nedevschi, S. Surana, A. Sheth, L. Subramanian, and E. Brewer. WiLDNet: design and implementation of high performance WiFi distance networks. In *USENIX NSDI*, MA, USA, Apr. 2007.

[5] B. Raman and K. Chebrolu. Design and evaluation of a new MAC for long-distance 802.11 mesh networks. In *ACM MOBICOM*, Cologne, Germany, Aug. 2005.

[6] E. R. Scheinerman. Matgraph 1.7. http://www.ams.jhu.edu/ ers/matgraph.

[7] S. Surana, R. Patra, S. Devschi, M. Ramos, L. Subramanian, y. Bendavid, and E. Brewer. Beyond pilots: Keeping rural wireless networks alive. In *USENIX NSDI*, San Francisco, CA, USA, Apr. 2008.

[8] K.-C. Toh, M. J. Todd, and R. H. Tutuncu. SDPT3: A matlab software package for semidefinite programming. *Optimization Methods and Software*, 1:545–581, 1999. http://www.math.nus.edu.sg/~mattohkc/sdpt3.html.

[9] V. V. Vazirani. *Approximation Algorithms*. Springer Verlag, 2004.