Fair Scheduling with Bottleneck Consideration in Wireless Ad-hoc Networks

Xinran Wu, Clement Yuen
Department of Computer Science

Yan Gao
Department of Mechanical and Industrial Engineering
University of Toronto
Toronto, ON M5S 1A4

Hong Wu, Baochun Li
Department of Electrical and Computer Engineering

Abstract —Most research work in the area of wireless ad-hoc networks attempts to balance the trade-off between *fairness* and *channel utilization*. In this paper, we first propose a topology-independent methodology to predict maximum achievable channel utilization under fairness constraint by two performance bounds. Based on the notion of *bottlenecks* introduced in prediction, we design a centralized and improved fair scheduling algorithm for wireless ad-hoc networks. We capture traffic load characteristics by using a proposed parameter that represents the "contending power" of nodes in the weighted flow contention graph. Finally, we demonstrate the effectiveness of our proposed algorithm through both provable analysis and simulations, and discuss natural derivations of a fully distributed algorithm using our bottleneck-based analytic model.

I. INTRODUCTION

In recent research, various resource management algorithms and protocols for mobile networking environments [1-3] are proposed to devise effective management schemes to support Quality-of-Service (QoS) in capacity-constrained and highly dynamic wireless networks. Typical proposals include QoS-oriented MAC layer design, packet scheduling, and admission control schemes, where *fair distribution of bandwidth* and *maximization of resource utilization* have been identified as two important design goals [2-4]. However, as identified by Luo et al. in their recent work [5-7], achieving both fairness and maximization of channel utilization is particularly challenging in wireless ad-hoc networks. They have proposed various distributed schemes that seek to maximize the aggregate throughput with a basic fairness guarantee. Although these are effective solutions, it is not clear *exactly what levels of fairness or throughput* they are able to achieve before simulating the algorithms.

In this paper, our major contributions are the following. First, we propose a novel prediction methodology, based on lower and upper bound analysis, to reveal the maximum achievable throughput under the strict notion of fairness for any network topology. Such predictions provide essential guidelines during the design of new fairness-aware protocols. Second, along with our prediction methodology, we present a key observation with respect to *bottleneck* considerations in multi-hop wireless networks. Such bottlenecks should receive full attention during analysis and scheduling. Finally, from such observations, we propose a new QoS parameter, based only on local flow weights and topology information, to integrate the degree of contention among flows into our fairness mode. With the parameter we design a centralized packet scheduling algorithm that achieves optimal channel utilization and fairness for each flow. The fact that only local state information is used promotes a fully distributed version of the scheduling algorithm.

The rest of this paper is organized as follows. In Section II, we present our system model and analysis on throughput prediction. Section III formalizes the notion of contending power of a flow based on bottleneck considerations, and presents our packet scheduling algorithm. We show our simulation results in section IV. Section V concludes the paper with discussions regarding a possible distributed implementation.

II. SYSTEM MODEL AND THROUGHPUT PREDICTION

In shared-medium multi-hop wireless networks, fair scheduling amounts to unbiased scheduling of spatially contending flows. Based on widely accepted definitions of fairness[1], various scheduling disciplines from wireline networks have been adapted in the multi-hop wireless domain [5-8]. On the other hand, non-contending flows that are spatially far apart could potentially be scheduled together, leading to effective channel utilization. A common strategy to arbitrate the conflicts between the two inherently incompatible design goals has been to maximize channel utilization under a certain fairness constraint. Taking this strategy into account, we propose our throughput prediction methodology based on *Weighted Flow Contention Graphs*.

A. Weighted Flow Contention Graph

A *flow contention graph* (or *flow graph*) represents spatial contention relationships among contending flows. Vertices

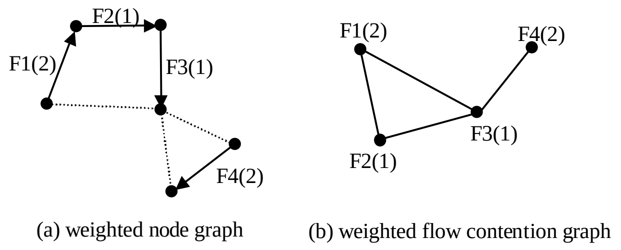

Fig. 1. A simple topology and its weighted flow contention graph

[1] A well-known definition is that of *max-min* fair share, which is not fully compatible with the notion used in this work, as illustrated in section II.C.2.

are mapped to backlogged flows represented by edges in the network node graph. An edge in the flow graph connects two vertices whenever the represented flows are within a two-hop distance. Thus the resulting undirected flow graph precisely illustrates the location dependency of spatial contention. Fig. 1 shows a conversion from a node graph to the corresponding flow contention graph.

When flows have unequal rights to channel resources, flow weights are often associated to represent their relative share. In our analysis we consider positive integer weights[2] $w = \{w_1, \ldots, w_n\}$ to be associated with the n vertices of the flow graph G, resulting in a *weighted flow contention graph* (G, w) for the topology. It should be noted that multi-hop flows are being modeled as *multi*-single-hop flows in our formulation. This can be understood as a per-hop behavior of packet scheduling.

B. Channel Reuse Index

We define the *average throughput* or *transmission rate*, u, for a system of flows in the multi-hop wireless network as

$$u = \frac{\textit{no. of packets transmitted in the network}}{\textit{transmission time}}. \qquad (1)$$

For simplicity, transmission is assumed to occur in discrete time slots. For fixed packet length the average throughput can be seen to be proportional to the number of packets transmitted per time slot. And if u_{sch} denotes the average throughput attained by a scheduling discipline *sched*, and u_{no_reuse} denotes the average throughput without channel reuse (e.g., from a strict fair queuing discipline), we define a *channel reuse index* (*CRI*) for the discipline to be

$$CRI = u_{sch} / u_{no\_reuse}. \qquad (2)$$

CRI can be seen to precisely measure the performance boost of a scheduling discipline with channel reuse considerations, in terms of its channel utilization.

C. Throughput Prediction

We study two common graph-theoretic techniques to predict maximal throughput in a generic multi-hop wireless network. Both techniques take fairness constraint into account when striving for maximal throughput. In the forthcoming discussions, mutually contending flows in G share a single channel of capacity C.

1) Weighted Graph Coloring

Graph or vertex coloring in graph theory finds widespread applications in many day-to-day scheduling problems, such as timetabling, register allocation and frequency assignment. Most of them have to do with avoidance of scheduling conflicts. In our context, we need to schedule weighted flows in a multi-hop network by segregating them into multiple non-contending sets, thereby exploiting channel reuse. The ultimate goal is to come up with a partitioning strategy that results in a minimal number of non-contending sets. This can be conveniently formulated as a minimum weighted graph coloring problem: Suppose (G, w) denotes a weighted flow graph where $G = (V_G, E_G)$. A proper k-coloring of (G, w) is an assignment to each vertex $i \in V_G$ a set of colors $S_i \in \{1, 2, \ldots, k\}$, such that $|S_i| = w_i$ and $S_i \cap S_j = \emptyset$ for any adjacent vertices i, j. We determine the minimum number of colors, also known as the *weighted chromatic number* $\chi_w(G)$ of the flow graph, for which a proper k-coloring exists.

A linear programming approach is, however more commonly used to formulate the generalized version of the problem: Suppose L denotes the set of all stable sets of a weighted graph (G, w). Find positive integer y_S for each $S \in L$ to solve

$$\min \sum\nolimits_{S \in L} y_S \qquad \textit{subject to}$$
$$\sum\nolimits_{S \in L \wedge i \in S} y_S \geq w_i \qquad \forall i \in V. \qquad (3)$$

Assuming a packet can be transmitted in unit time slot, by minimum coloring techniques we could then deliver all packets within one scheduling cycle in only $\chi_w(G)$ time slots. The throughput gain based on this approach, or in other words, its channel reuse index is therefore given by

$$CRI_{col} = \sum\nolimits_i w_i / \chi_w(G). \qquad (4)$$

From the optimality of minimum coloring, we contend that *CRI_{col} sets out a feasible lower bound on maximum channel reuse or throughput prediction.*

2) Maximum Weighted Clique (Bottleneck Analysis)

In wireless ad-hoc network, localities of intense spatial contention, or *bottlenecks,* should be identified and honored when predicting maximum throughput under the fairness constraint. In this context we are particularly interested in severe bottlenecks, identified as maximal weighted cliques in a given weighted flow graph. Consider the simple flow graph of Fig. 1. A feasible fair scheduling scheme would allocate 1/2, 1/4 and 1/4 of the channel capacity to the bottleneck flows F1-F3, while F4 would not be given a larger share than F1[3] (Fig. 2). A proper prediction of maximum throughput of

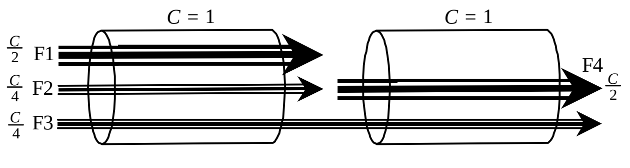

Fig. 2. Flow contention situation for flow graph of Fig. 1: F1, F2 and F3 fully consume one channel; F3 and F4 partially consume a different channel. Resource allocation, however, has complied with the fairness constraint.

[2] Generalization to non-integer weights is possible by means of proper formulation. For example, this requirement can be relaxed in the linear programming formulation of weighted graph coloring.

[3] This is in contrast with max-min fairness, where in this scenario flow 4 will be allocated a share of 3/4.

the network is thus 3/2 of the single channel capacity. However, a careless prediction without bottleneck consideration would have claimed 1/3 and 2/3 of the capacity for F3 and F4, leading to an unrealistic conclusion that F1-3 combined consumes 4/3 of the single channel capacity.

Formally, suppose u_0 is the throughput per unit weight in the weighted flow graph $G = (G, w)$. Observing the capacity constraint for each identified weighted clique, we write down a system of inequalities

$$\sum_{i\in\Omega_j} u_0 w_i \le C \qquad \Omega_j \in \Omega_G, \tag{5}$$

where Ω_G is a set of identified cliques of G. To ensure feasibility in (5), the throughput per unit weight must satisfy

$$u_0 \le C \Big/ \max\left(\sum_{i\in\Omega_j} w_i\right) \qquad \Omega_j \in \Omega_G. \tag{6}$$

The denominator of (6) is naturally the *weighted clique number* $\omega_w(G)$ of the weighted flow graph, frequently used as a lower bound for $\chi_w(G)$. Hence, an upper bound on the channel reuse index easily follows:

$$CRI_{clq} = u_0 \sum_i w_i \big/ C = \sum_i w_i \big/ \omega_w(G). \tag{7}$$

Since capacity constraints can never be violated, we contend that CRI_{clq} *sets out a theoretical upper bound on maximum channel reuse or throughput prediction.*

With the two bounds for prediction in place, we discuss their significance. First, in cases where $\chi_w(G) = \omega_w(G)$, we have the tightest bounds; otherwise other theories are also known to obtain a tighter lower bound for CRI_{col}[4]. Second, we expect scheduling disciplines claimed to deliver optimal throughput under fairness constraint to observe the two bounds. This expectation, however, may just be too optimistic in our opinion. Third, we also note that the theoretical bound of CRI_{clq} is maximal on condition that the fairness constraint is honored. There is absolutely no reason why an exceeding *CRI* cannot be realized through flow starvation.

III. Centralized Fair Scheduling with Bottleneck Considerations

In this section, we describe a centralized scheduling algorithm that takes our bottleneck notion into consideration for multi-hop networks. Recall that bottlenecks are localities where special attention is required. We would simply prefer to pay such attention when designing a scheduling discipline. *In particular, such scheduling discipline needs to give priority services to flows belonging to a bottleneck locality.* In order to differentiate between the severities of bottlenecks to which flows belong so as to assign the appropriate priorities, we devise a metric known as the *contending power* of flow for the purpose.

[4] When the bounds are equal, the weighted flow graph is said to be *pluperfect*. Techniques to obtain tighter lower bound include the use of fractional coloring, where the integer requirements on color "weights" y_S are relaxed.

A. The Flow Contending Power

We define the *flow contending power* P_i for a flow i as

$$P_i = \max\left(\sum_{k\in\Omega_j} w_k\right) \qquad i \in \Omega_j, \Omega_j \in \Omega_G. \tag{8}$$

Intuitively, it measures indirectly the level of contention a flow perceives in its neighborhood. By comparing the contending powers of respective flows, we can identify locations of bottlenecks and assign priorities accordingly. Under such notion of contending power, we do not intend to single out at all times the flow that experiences the most contention, but only the one from a particular subset currently under scheduling consideration. We claim that even in a fully distributed environment, a node can "learn" its P_i by exchanging topology and weight information with only the nodes in its neighborhood.

Example 1: Consider the flow graph of Fig. 3, Table I reflects the contention power for two different weight sets.

B. Considerations in a Centralized Scheduling Algorithm

While bottleneck consideration is a valuable methodology for prediction, it does not constitute sufficiency in scheduling decisions per se. In our case, it is only being used as a supplementary tool within a fair scheduling discipline that provides basic fairness. We adopt Start-time Fair Queuing (SFQ) [9] to assign two tags to each arriving packet: a start tag and a finish tag. Specifically, a packet with sequence number n of flow i arriving at time $A(t_{i,n})$ is assigned a start tag $s_{i,n}$ and a finish tag $f_{i,n}$, defined as follows:

$$s_{i,n} = \max\{V(A(t_{i,n})), f_{i,n-1}\}; \qquad f_{i,n} = s_{i,n} + L_p \big/ w_i, \tag{9}$$

where L_p denotes the packet size in bits. The virtual time $V(t)$ of the scheduler at time t is set to be the start tag of the packet currently being served by the scheduler at time t.

In order to enable spatial reuse, we have to locally swap the transmission orders so that non-contending flows can transmit at the same time. We would also like to perform our bottleneck analysis to only a subset of flows to ensure long-term fairness. This motivates a "scheduling window" setting at any time instant, in which the scheduler tries to efficiently schedule up to η packets within the scheduling boundary. New flow packets enter the scheduling window based on the ascending order of their start tags. Basically, the scheduling window should always be filled up to full capacity to expedite delivery of packets.

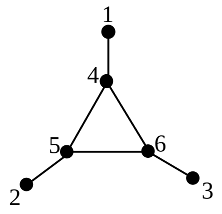

Fig. 3. Flow graph

TABLE I
WEIGHTS AND CONTENTION POWER FOR FIG. 3.

Flow	w_i	P_i	w_i	P_i
1	1	2	1	2
2	1	2	2	4
3	1	2	3	6
4	1	3	1	6
5	1	3	2	6
6	1	3	3	6

In summary, the following scheduling rules are enforced within the scheduling window:

Rule 1: Bottleneck Consideration. The packet from a flow f carrying the highest contending power P_f is always given the priority for transmission.

Rule 2: Start Tag Usage. For the packets from flows carrying the same maximum contending power, the one with the smallest start tag from flow f is given the priority to transmit.

Rule 3: Maximal Independent Flow Set. To optimize network utilization, the maximal set of packets that are not contending with flow f is selected to transmit simultaneously.

Rule 4: Secondary Usage of Contending Power. If there are several such flow sets in Rule 3, we compute the total contending power for each set and select the highest one to transmit with flow f. Among multiple sets with the highest contending power, the one with the largest cardinality will be selected to transmit with flow f. Further ties are broken arbitrarily.

C. *The Centralized Scheduling Algorithm*

The algorithm comprises of five steps:

Step 1: Compute the contending power P_i, pre-compute start tag and finish tag for each flow in the flow graph.

Step 2: Pre-fill the scheduling window with packets from the scheduling queue in ascending order of start tags.

Step 3: Within the scheduling window, apply Rule 1 to grant transmission priority to the flow f with the largest P_f. Apply Rule 2 when necessary.

Step 4: Apply Rule 3 to select the appropriate non-contending flow set containing P_f. Apply Rule 4 accordingly to seek additional resolution. Transmit the resulting flow set simultaneously with P_f.

Step 5: Refill packets into the scheduling window from the scheduling queue. Repeat steps 3, 4, and 5.

D. *Algorithmic Properties*

1) *Fairness Guarantee*

In our design, we use SFQ to achieve the basic fairness. In attaining channel reuse, we swap service order of the queuing packets. In order to guarantee long-term inter-flow fairness, we adopt the scheduling window mechanism to constrain potential unfairness due to channel reuse by only rescheduling queuing packets within the scheduling boundary. Even within the scheduling window, the notion of fairness is not totally abandoned. We note that the minimum start tag mechanism is still being adopted to resolve selection conflict between two candidate flows with maximum contending power. In addition, we claim that stricter short-term fairness can easily be achieved by simply adding a counter to each of the packets in the scheduling window, keeping track of its sojourn time. We then give a packet the highest priority when its counter exceeds a time bound ξ, so that short-term unfairness is effectively bounded by $f(\xi)$.

2) *Maximal Throughput*

Our algorithm has an edge over others that consider channel reuse in that we pay more attention to the highly congested areas in the topology. We always select the bottleneck flow within the scheduling window to realize channel reuse. We compare contending powers of the independent flow sets *containing* the bottleneck flow to identify the maximal non-contending one. This strategy allows efficient channel utilization while staying in line with our design tenet that localities of high contention should always be honored.

3) *Between Fairness and Maximal Throughput*

Based upon the aforementioned properties, we argue that our algorithm can find a balance spot between the two seemingly incompatible design goals: fairness and maximal throughput.

IV. SIMULATIONS

In this section, we perform simulations to compare our algorithm with one based on dynamic graph coloring [6]. At each time interval, the dynamic graph coloring approach adopts an adaptive or greedy algorithm to select flows for transmission, in an attempt to reach some local optimum solution. In this respect it approximates the global optimum in the end. Simulations show that in a centralized environment, our approach based on bottleneck considerations can achieve as good channel utilization as graph coloring approach in most of the cases.

Simulation is based on a multi-hop wireless network environment, with the following assumptions: (1) *No overhead considerations.* The similarity between the algorithms allows for this simplifying assumption. (2) *Discrete time slots and fully contending packet arrival pattern.* The physical channel capacity is assumed to be one packet per time slot. Packet is per-flow queuing and all flows are sufficiently backlogged during the course of simulation, showing the busy traffic scenario. (3) *Ignore the length effect of look ahead window and allow for warm up procedure.*

In Scenario 1 (Fig.3. and Table I), we study the cases of identical and different flow weights. Bottleneck analysis immediately reveals the locality 4-5-6 to be the highly congested part of the network, analogous to a heavily loaded subnet. The total run time is 10000 time slots, and the output is tabulated in Table II by different weight sets.

TABLE II: SIMULATION RESULTS
(1) GRAPH COLORING APPROACH; (2) OUR APPROACH

w_i	$u_{i\,(1)}$	$u_{i\,(2)}$	w_i	$u_{i\,(1)}$	$u_{i\,(2)}$
1	3333	3333	1	1666	1666
1	3334	3333	2	3333	3333
1	3334	3333	3	5001	5000
1	3333	3334	1	1666	1667
1	3333	3334	2	3334	3334
1	3333	3334	3	5001	5000
CRI	2.0000	2.0001	*CRI*	2.0001	2.0000

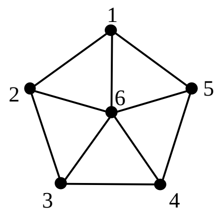

Fig. 4. Flow graph for scenario 2

TABLE III
WEIGHTS AND CONTENTION POWER FOR SCENARIO 2

Flow	w_i	P_i	w_i	P_i
1	1	3	1	6
2	1	3	2	8
3	1	3	3	8
4	1	3	1	7
5	1	3	2	6
6	1	3	3	8

TABLE IV: SIMULATION RESULTS
(1) GRAPH COLORING APPROACH; (2) OUR APPROACH

w_i	$u_{i\,(1)}$	$u_{i\,(2)}$	w_i	$u_{i\,(1)}$	$u_{i\,(2)}$
1	2857	2857	1	1250	1250
1	2857	2857	2	2500	2500
1	2857	2857	3	3750	3750
1	2857	2857	1	1250	1250
1	2857	2857	2	2500	2500
1	2857	2857	3	3750	3750
CRI	1.7142	1.7142	*CRI*	1.5000	1.5000

From the results, we note not only do both algorithms achieve long-term fairness (i.e. throughput proportional to flow weights), our approach is also as favorable as the graph coloring approach in terms of throughput. From Section II, the upper and lower bounds for this scenario are identical: $CRI_{col} = CRI_{clq} = 2$, meaning that both scheduling algorithms achieve the optimum.

In scenario 2 (Fig. 4 and Table III), the upper and lower bounds are not tight for the case of identical weight (lower bound $CRI_{col} = 1.5$ and upper bound $CRI_{clq} = 2$). Table IV shows the throughput result. Our approach still compares equally well with the dynamic graph coloring approach. The *CRI* obtained for the unweighted topology is 1.7142, which falls within the range of the lower and upper bounds. In this respect, it shows that our approach based on bottleneck observation is promising. In addition, it also proves the correctness of the bounds prediction.

V. DISCUSSIONS AND CONCLUDING REMARKS

Simulations show that even in an ideal centralized environment, our approach leads to very desirable results, as optimal as the dynamic graph coloring approach [6]. Dynamic graph coloring relies on global information from the topology, rendering it hard to implement in a distributed fashion. In bottleneck approach, flow contending power (P_i) calculations require only knowledge of local topology information (via broadcasts among the neighboring nodes). Thus it easily motivates a fully distributed scheme suitable for running in an ad-hoc environment. We also believe that a fully distributed algorithm based on bottleneck approach will generate better throughput under strict fairness than other currently available distributed algorithms.

We outline a possible distributed algorithm based on bottleneck observations as follows. Each node (flow) broadcasts its own weight information to all neighboring nodes. This allows nodes (flows) sufficient topology and weight information to compute their own contending powers. Nodes also include their contending powers (P_i) in packet headers, along with other information such as weight (w_i) and start tag (s_i). To schedule flows for transmission, a look-up table with entries (w_i, s_i, P_i, m_i), where $m_i = S(w_i,s_i,P_i)$ represents a selection metric, is built in each node by overhearing packets from contending flows in the neighborhood. At any instant the flow in the table with the largest m_i has access to the channel. Part of our future work is to implement such a distributed algorithm and prove our arguments through simulations.

The goal of this paper is to fully study the notion of fairness in a shared-medium, multi-hop ad-hoc wireless network. Given a network topology, we are able to predict the maximum achievable throughput under the notion of fairness with our model based on bottleneck observation. Our model works well in a centralized fair scheduling discipline, and we also believe it constitutes solid theoretical and analytical grounds for a fully distributed implementation. We have demonstrated the effectiveness of our proposed algorithm through both provable analysis and simulations.

REFERENCES

[1] C. Chang, J. Chang, K. Chen, and M. You, “Guaranteed quality-of-service wireless access to ATM”, *IEEE Journal of Selected Areas in Communications*, 1997.

[2] J. Ju and V.O.K. Li, “An optimal topology-transparent scheduling method in multi-hop packet radio networks”, *IEEE Trans. Networking*, 6(3), June 1998.

[3] S. Lu, V. Bharghavan and R.Srikant, “Fair scheduling in wireless packet networks”, *IEEE Trans. Networking*, August1999.

[4] I. Chlamtac and A. Lerner, “Fair algorithms for maximal link activation in multi-hop radio networks”, *IEEE Trans. Communications*, 35(7), July 1987.

[5] H. Luo, S. Lu and V. Bharghavan, “A new model for packet scheduling in multi-hop wireless networks”, *ACM MOBICOM’00*, August 2000.

[6] H. Luo, S. Lu, “A topology-independent fair queuing model in ad hoc wireless networks”, *IEEE ICNP’00*, Nov. 2000.

[7] H. Luo, P. Medvedev, J. Cheng, S. Lu. “A self-coordinating approach to distributed fair queuing in ad hoc wireless networks”, *IEEE INFOCOM’01*, April 2001.

[8] T. Nandagopal, T. Kim, X. Gao and V. Bharghavan, “Achieving MAC layer fairness in wireless packet networks”. *MOBICOM 2000*, Boston, MA, August 2000.

[9] P. Gerla, H. Vin, and H. Cheng, “Start-time fair queuing: a scheduling algorithm for integrated services packet switching networks”, *ACM Journal on Wireless Networking*, December 1995.