Packet Aggregation based Back-pressure Scheduling in Multi-hop Wireless Networks*

Gaurish Deuskar, Parth H. Pathak, Rudra Dutta
Department of Computer Science, North Carolina State University, Raleigh, NC, USA.
Email: gaurishdeuskar@gmail.com, phpathak@ncsu.edu, dutta@csc.ncsu.edu

***Abstract*—The back-pressure based scheduling policy originally proposed by Tassiulas et al. in [1] has shown the potential of solving many fairness and network utilization related problems of wireless multi-hop networks. Recently, the scheduling policy has been adapted in random medium access protocols such as CSMA/CA using prioritization of MAC layer transmissions. Here, MAC priorities are used to provide differentiated services to nodes depending on their queue backlogs. Even though these schemes work well in experiments to emulate back-pressure scheduling, they perform poorly with realistic Internet-type traffic where there is a large variation in packet sizes. In this paper, we propose packet aggregation based back-pressure scheduling which aggressively increases the rates at which back-logged queues are served. Different from other aggregation schemes, the presented scheme utilizes the back-pressure principles for determining when and how much aggregation is performed. We show that this results into increased service rates of back-logged queues which in turn results into high network throughput and utilization. We verify our scheme using simulations and testbed experiments, and show that it achieves significant performance improvements as compared to the original scheme.**

I. Introduction

The back-pressure scheduling/routing policy first proposed by Tassiulas et al. [1] has recently shown a great potential for solving a number of issues in wireless multi-hop networks. The central idea of back-pressure scheduling policy is that contention among the links should be resolved by scheduling the link which has the largest product of queue differential backlog between its endpoints and transmission rate at which the link can be served. In a perfectly time-slotted medium access mechanism such as TDMA, this will result into optimal throughput of flows while guaranteeing queue stability (ingress traffic to a queue never exceeds its egress traffic). The utility maximization framework initially proposed in [2] shows that injection rates of flows should be chosen such that aggregate utility of the flows is maximized. Here the utility of flow represents a desirable effect on the network achieved by a particular rate of the flow. It was shown in [3]–[6] that back-pressure scheduling and utility based rate control together can solve the global problem of network utility maximization.

The fundamental challenge with back-pressure framework is that solution of the underlying scheduling strategy is NP-hard [7]. Also, since it was proposed for a centralized, synchronized and time-slotted system, a distributed implementation which can achieve even a closer approximation is very difficult to develop. Recently, [8], [9] have attempted to incorporate back-pressure based scheduling in random medium access protocols such as CSMA/CA. These protocols try to approximate the performance of the ideal back-pressure scheduler by prioritizing the frame transmissions according to differential backlogs of queues. Here, every node in the network maintains a per destination queue (PDQ) and the packets destined to a particular destination are stored in the PDQ of that destination until further forwarding decisions are made. Now, nodes share their PDQ information with their neighbors, and this information is utilized by every node to calculate differential backlogs of its PDQs. The differential backlog of a PDQ at a node is equal to the size of the PDQ minus the size of the PDQ of its upstream neighbor towards the destination. To emulate the back-pressure scheduling, packets of the PDQ which has the highest differential backlog (highest back-pressure) in the neighborhood are given the higher chances for transmission. This way, the likelihood that packets are transmitted from a particular PDQ at a node is proportional to its differential backlog compared to the differential backlogs of PDQs of all nodes in the neighborhood. This prioritization quickly moves the traffic from long back-logged queues to shorter queues achieving an improved throughput and a better overall stability of queues.

The idea of using different MAC layer priorities for PDQs with different back-pressure works well in imitating the ideal back-pressure scheduling over CSMA/CA. Even though this enables a distributed implementation, it does not guarantee that only the frames from maximum back-logged queue are being transmitted at any given time. This is because MAC priorities themselves are implemented by modifying the size of the contention window from which the back-off times are randomly chosen. A higher MAC priority frame uses a shorter random back-off time before being transmitted, and it is possible that frames from queue with lower differential backlog get transmitted before frames from queue with higher differential backlog. This problem of not serving the most back-logged queue with the highest possible rate is further aggravated by the fact that most of the packets in last mile networks such as wireless mesh networks are very small in size. [10]–[12] show that application layer traffic and TCP acknowledgments account for more than 70 to 80 percent of total traffic where packet sizes are lesser than 100 bytes. With such small packet sizes, time required for medium access and header/trailer overheads adversely affect the overall performance. In case of prioritized frame transmissions, this further reduces the chances that the most back-logged queue is obtaining the maximum possible medium access. Even if the packets are being transmitted from that queue, additional time required for medium access before transmitting every small

*This work is supported by the U.S. Army Research Office (ARO) under grant W911NF-08-1-0105 managed by NCSU Secure Open Systems Initiative (SOSI). The contents of this paper do not necessarily reflect the position or the policies of the U.S. Government.

packet accumulates to a large waste of available air time. Both these factors reduce the rates at which back-logged queues are served which in turn reduces the throughput and network utilization.

In this paper, we propose utilizing packet aggregation in the back-pressure framework. The objective is to increase the throughput while maintaining the inherent properties and benefits of back-pressure scheduler. The central idea is to take advantage of packet aggregation for improving the rates at which the back-logged queues are served. Different from other packet aggregation schemes ([12]–[17]), the presented strategy utilizes the back-pressure invariants to guide when and how much aggregation is performed. Specifically, when a queue has the highest differential backlog in the neighborhood, instead of transmitting all its head of line packets one by one, the proposed scheme aggregates the first few packets in a large MAC frame which is then transmitted with the highest MAC priority. The number of packets that are aggregated depends on maximum allowable MAC frame size and the differential backlog of the queue. The proposed scheme attempts to reduce the size of the back-logged queues more aggressively which results into their faster service rates. Since the back-pressure policy is utilized to perform the aggregation, the presented scheme preserves the network utility and fairness advantages of the back-pressure policy while yielding an improved throughput performance. This leads to a closer approximation to the optimal solution of back-pressure problem (formally described in section II). Apart from throughput improvements, aggregation based back-pressure solution reduces the end-to-end delay of packets dramatically which is especially important in delay-constrained applications. We test our scheme using OPNET simulations and testbed experimentation to verify its performance. It is observed that our scheme outperforms the most current implementation of back-pressure scheduler which does not utilize packet aggregation.

The rest of the paper is organized as follows. Section II formally describes the back-pressure framework along with utility maximization. Section III presents the aggregation based back-pressure scheduling strategy in details. The numerical results of simulation and experimentation are presented in section IV and we conclude the findings in section V.

II. The Back-pressure Framework

We first provide a brief overview of the back-pressure framework. The back-pressure framework consists of two parts: a link scheduling strategy based on back-pressure and a rate control module for network utility maximization.

Back-pressure scheduling: Back-pressure based link scheduling policy was originally proposed by Tassiulas et al. in [1]. Let's represent the network under consideration using a graph $G = (V, E)$. Let f denote a traffic flow from source node $s(f)$ to destination node $d(f)$ and F be the set of all flows currently active in the network. Let $l(u, v) \in E$ denote a link between node u and v, and $\gamma_{l(u,v)}$ be the transmission rate of link $l(u, v)$. Transmission rates of all links in E are presented by $\Gamma = \{\gamma_{l(u,v)}, l(u, v) \in E\}$. Let χ be the set of all possible combinations of rates at which links can operate.

Let us assume only for now that the transmission time is divided into equal sized time slots. Every node $u \in V$ maintains a separate queue for destinations of all flows in F. These queues at each node are also known as per destination queues (PDQs). Packets received by u for a flow f destined to $d(f)$ is stored in the queue $Q_u^{d(f)}$ until further forwarding decisions are made. Let $|Q_u^{d(f)}(t)|$ denote the size of the PDQ maintained at node u for destination $d(f)$ at time t. Every node shares its PDQ length information with all its neighbors at the beginning of time slot t. Every node then calculates its differential backlog as compared to its neighbors for every flow destination. That is, a node u calculates $D_{l(u,v)}^{d(f)}(t) = |Q_u^{d(f)}(t)| - |Q_v^{d(f)}(t)|$ for all $l(u, v) \in E$ and all $f \in F$. Now, for every link $l(u, v) \in E$, let

$$\Delta_{l(u,v)}(t) = \max_{f \in F} \left(D_{l(u,v)}^{d(f)}(t) \right) \tag{1}$$

Back-pressure scheduling suggests that Γ at time t should be chosen such that –

$$\Gamma(t) = \max_{\Gamma \in \chi} \sum_{l(u,v) \in E} \left(\gamma_{l(u,v)} \Delta_{l(u,v)}(t) \right) \tag{2}$$

It was proved in [1] that a routing/scheduling policy that can achieve a solution of Equ. 2 is throughput optimal. This means that it stabilizes the queues at every node while supporting the largest possible capacity region. A capacity region (C) of a network is defined as the set of all flow rates which are supportable by the network. Due to link interference constraints, the above mentioned problem is proven to be NP-hard in wireless networks. Also, its distributed implementation imposes many more challenges in design of a routing/scheduling scheme for wireless networks.

Rate control: The back-pressure scheduler determines the transmission rates at which packets are served at links so that the queue sizes at nodes remain bounded while maximizing the achievable throughput. The other part of the back-pressure framework performs the rate control of the flows. The flow controller determines the rate at which flows can inject the packets in the network. In back-pressure framework, the flow controller determines the input rates of flows based on the state of queues at each intermediate node. The PDQ information is utilized by the flow controller to adjust the flow input rates in a way that a desirable network-wide objective is optimized. In a seminal work, [2] showed that such flow/congestion control can be viewed as primal-dual algorithm for the solution of network utility maximization problem. This was further elaborated in [8] in context of the back-pressure framework which we describe next.

Suppose that each flow f from $s(f)$ to $d(f)$ has a utility function associated with it. This utility function $U_f(x_f)$ is a function of the rate x_f of the flow f. Let us represent input rates of all flows using $\psi = \{x_f, f \in F\}$. The utility maximization problem suggests that the flow rates should be chosen such that their aggregate utility is maximized, that is

–

$$\max_{\psi \in C} \sum_{f \in F} U_f(x_f) \tag{3}$$

A flow controller that can maximize the aggregate utility was presented in [8]. It suggests that in each time slot t, source $s(f)$ of a flow f injects a packet in the network if and only if

$$U'_f(x_f(t)) - \beta Q^{s(f)}_{d(f)}(t) > 0, \tag{4}$$

where U'_f is the first derivative of utility function of the flow and β is a small constant. The above condition interprets to the fact that a packet should be injected into the network by a flow only if the eventual utility benefit of the insertion is larger than a constant times the size of source node PDQ. That is when intermediate nodes of a flow are sufficiently back-logged, the source node pushes more packets into the flow at a slower rate. The back-pressure scheduler ensures that queue backlog status of intermediate nodes is reflected back at the source node which then performs the rate control to accordingly change the flow input rate.

This way, the back-pressure based link scheduling controls which links should be transmitting at what rate, and the rate control module manages the rates at which the packets are injected by the flows. Both together can solve the network-wide utility maximization problem to yield a throughout-optimal solution.

III. Back-pressure and Packet Aggregation

In this section, we provide the details of aggregation based back-pressure scheduling. First, we show how back-pressure scheduling is implemented using MAC priorities and then we go on to describing the aggregation algorithm.

As described before, in back-pressure scheduling every node maintains a PDQ for every flow passing through the node. Note that we assume a fixed routing policy in this work where routes are calculated in advance for every flow and packets are forwarded on these routes only. This is different from an ideal back-pressure strategy where the routing is adaptive, and forwarding decisions are made on packet by packet basis. In the ideal scheme, every neighbor of a node depending on its size of PDQ is a potential candidate for forwarding the packet [18]. Since our objective is to explore packet aggregation with back-pressure strategy, we restrict our focus to fixed routing and leave the adaptive routing extension to future work.

Due to fixed routing, every node has a unique upstream neighbor for every flow passing through it. Now, the differential backlog at a node u for flow f can be presented as $D^{d(f)}_u(t) = |Q^{d(f)}_u(t)| - |Q^{d(f)}_v(t)|$, where v is next hop neighbor of node u for flow f. Each node maintains following information ([8], [9]) in order to correctly execute the joint back-pressure and aggregation scheme –

1) Per destination Queue (PDQ) – a separate queue for each destination of flows passing through the node. Every packet (generated at the node or received from downstream neighbor) is stored in the PDQ of corresponding destination until further forwarding decisions are made.
2) Urgency Weight – every PDQ has an urgency weight associated with it. The urgency weight is the differential backlog which equals to the backlog of the PDQ subtracted by the backlog of the PDQ on the next hop neighbor towards the destination.
3) Urgency Weight State (UW state) – along with maintaining PDQs and their corresponding urgency weights, every node knows the PDQ ID, node ID and urgency weight of the PDQ which has the maximum urgency weight in the neighborhood. The same is also maintained for the minimum urgency weight PDQ of the neighborhood. This information determines a node's UW state with respect to its neighbors. To implement this, every node shares its PDQ information with its neighbors by using separate messages or piggybacking techniques.
4) Source List – source node of every flow which generates packets first stores the packets in a per destination source list. Once the rate control is performed then only the packets are added to the corresponding PDQ at the node. Different from the PDQs which are maintained at all nodes, per flow source lists are only maintained at the source nodes of the flows.

In a back-pressure scheduling implementation without aggregation ([8], [9]), when a node sends out a frame from a PDQ, it first determines its MAC layer priority. This is described in *Determine-MAC-priority* function of Algorithm 1. The MAC priority is essentially determined by comparing the urgency weight of the PDQ to the UW state of the node. In essence, this compares the differential backlog of the PDQ with the differential backlog of other PDQs in the neighborhood and assigns it a MAC priority based on it. Packets of the PDQ which has the highest urgency weight in the neighborhood are assigned the highest MAC layer priority for faster transmissions. Here, it is assumed that MAC layer is able to provide differentiated levels of service. In our experiments, 4 MAC layer priorities are considered starting from 0 to 3. Packet sent out with higher priority has higher chances of accessing the channel as compared to the packets sent out at lower priorities. CSMA/CA MACs like 802.11e provides such differentiated service levels which we also use in our simulations. This prioritization of transmissions tries to serve the longer queues with higher service rates which is the ultimate objective as described in Equ. 2.

Packet aggregation: The service rates of back-logged queues can be further increased if more and more packets are transmitted from the queues with higher urgency weights. Note that the above implementation of back-pressure strategy without aggregation tries to maximize the chances that medium is occupied by the packets transmitted from the longer queues, but it does not guarantee to do so. Since a large number of packets are very small in size and medium access has to be initiated before every transmission, it is not guaranteed that the packets of the longest queue in a neighborhood receives the highest possible service rate from the medium. Instead if the packets of the longest queues are aggregated before their medium access and eventual transmission, more

packets can be sent out from the backlogged queue in each medium access. This eliminates the additional time required for medium access by every small packet and increases the total air time utilization by longer back-logged queues. This increases the service rates of back-logged queues which in turn maximizes the objective function presented in Equ. 2. It is obvious that back-pressure with aggregation is also an approximation of the optimal solution to Equ. 2 but is closes the gap further towards the optimal solution by improving on any currently available scheme.

Algorithm 1 Packet aggregation + back-pressure scheduling

maxAggSize ←Maximum allowable size of a MAC frame;
size ← 0,
q ←PDQ with the highest urgency weight,
repeat
  Dequeue the HOL packet p from PDQ q,
  Add p to the packet aggregate,
  size := *size* + sizeof(p),
until *size* $\leq$ *maxAggSize* **AND** PDQ q has the highest urgency weight in the neighborhood
Determine MAC priority of the aggregated packet using Determine-MAC-priority(q),
Transmit the packet with the MAC priority,
Update the size and urgency weight of PDQ q.

Determine-MAC-priority(q)
  numLevels ← Number of MAC priority levels,
  max ←Maximum urgency weight in the neighborhood,
  min ←Minimum urgency weight in the neighborhood,
  urg ←Urgency weight of the PDQ q,
  macPrioLevel := $\frac{urg-min}{max-min} * numLevels$,
return *macPrioLevel*

Now we describe how exactly packet aggregation is performed while using back-pressure invariants as guidelines. Aggregation based PDQ scheduler at every node de-queues the head of line packets from PDQ with the highest urgency weight. The node then performs the check whether the PDQ is in fact the highest urgency weight PDQ in the entire neighborhood. If so, it de-queues more packets from the PDQ. It continues to do so until the PDQ retains the highest urgency weight in the neighborhood or until sum of the size of all the de-queued packets is less than the maximum allowable MAC frame size. All the de-queued IP packets are then bundled into a large MAC frame of aggregated packets. The MAC frame is then assigned a MAC priority depending on the current UW state of the node. Since the PDQ from which the packets were de-queued had the highest urgency weight in the neighborhood, it is also likely that the MAC priority of the frame of aggregated packets will be the highest too. Because of aggregation the total time taken to transmit these set of packets will be reduced as compared to their individual transmissions. This increases the service rate of the PDQ with highest urgency weight in the neighborhood yielding a closer approximation of Equ. 2. The complete procedure of aggregation and back-pressure scheduling is described in Algorithm 1.

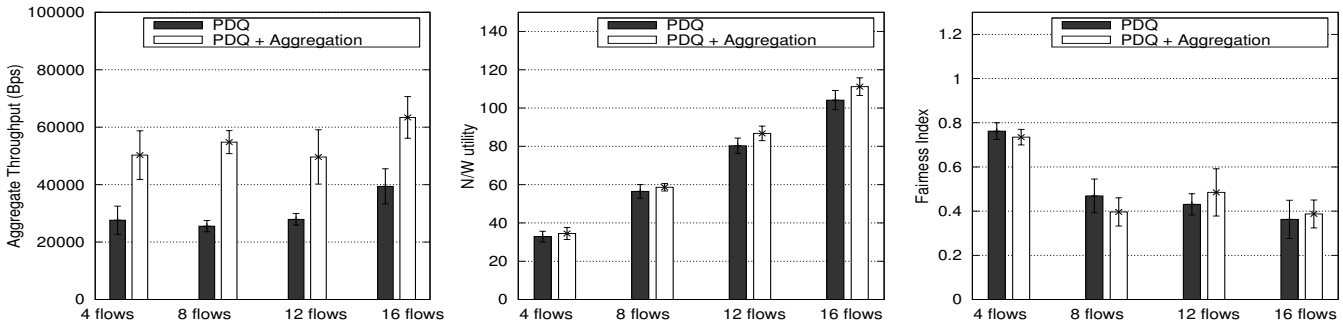
Rate control: While the back-pressure scheduling and aggregation is performed at every node of the network, source node of every flow performs the rate control. The function of this rate control policy is to determine at what rate the packets should be injected in the network. As described in section II, the objective of such a flow control is to maximize the utility (or benefit) associated with each flow. Here, whether a packet is injected into a flow or not depends on the utility function of the flow and the size of PDQ at the source node for the flow. Specifically, a flow f injects a packet into the network as long as $U'(x_f) > \beta Q^{s(f)}_{d(f)}(t)$, where $U'(x_f)$ is the first derivative of the utility function U_f. We use $U_f(x_f) = log(x_f)$ as the utility function as described in [8]. β is a small constant whose value is set to 10^{-6}.

This means that the source node $s(f)$ of a flow f first checks whether $U'(x_f) > \beta Q^{s(f)}_{d(f)}(t)$ condition holds. If the condition holds true, the source node inserts the packet in its PDQ for flow f. On the other hand, if the condition does not hold true, the source node inserts the packets in the source list of the flow. The source list acts as a tentative storage where packets are buffered until rate control permits them to be added in corresponding PDQ. When back-pressure scheduler transmits packets from the PDQ, packets are added in the PDQ from the source list using the rate control condition. At all times, any addition or removal from PDQ is followed by recalculation of urgency weight of the PDQ. If the source node PDQ reflects that intermediate nodes are sufficiently back-logged, it slows down the rate at which the packets are added from the source list to the PDQ.

IV. Numerical Evaluation

In this section, we present the numerical results obtained using simulation and testbed experimentation. First, we simulate the aggregation based back-pressure scheduling in OPNET [19] and evaluate its performance under various different performance metrics. Next, we provide details of testbed implementation and related numerical results of observed performance.

As it is obvious that the above mentioned back-pressure framework requires every node to share their PDQ information with every node in its reach. To implement this, we piggyback the PDQ information on every packet that is exchanged between a node and its neighbors. Specifically, every packet carries an additional UW header which contains the following information: 1) PDQ ID, urgency weight and backlog (in bytes) of the PDQ from which the packet was transmitted, 2) ID and urgency weights of maximum and minimum urgency weight PDQs of the node from which the packet was transmitted. The receiver uses this information from every neighbor to calculate its own UW state. Then the UW state is utilized to calculate the MAC layer priority of an outgoing frame from that node. We use 4 priority levels for MAC transmissions as provided by 802.11e.



(a) Aggregate network throughput (b) Network utility (c) Fairness

Fig. 1: Comparison of network throughput, network utility and flow throughput fairness in back-pressure scheduling with and without aggregation

The aggregation based back-pressure scheduler and rate control were implemented by modifying current 802.11 implementation of OPNET. We first consider a 50 node topology where nodes are uniform randomly placed in the network area. Variable number of UDP flows are initiated between random pairs of source and destination. The rate control is performed by modifying how a UDP flow inserts data at the source node from its source list to its corresponding PDQ. To understand the impact of aggregation, we emulate realistic Internet-type traffic scenario where packet sizes vary substantially [20]. Traffic for every flow is generated randomly in packet sizes of 40, 60, 80, 100, 200, 576 and 1400 bytes with their weights being 40, 15, 10, 5, 5, 5 and 20 percent respectively.

There are various different performance metrics of interest for evaluation of a back-pressure based strategy. We compare our aggregation based scheduling/rate control to a back-pressure scheduling/rate control scheme which does not perform any aggregation. Such a scheme was originally proposed by [8]. One obvious metric of performance evaluation is aggregate throughput which is simply the sum of throughput of all flows in the network. Two other metrics – network utility and throughput fairness are central to back-pressure based policies. Since the back-pressure framework was originally designed to maximize the network resource utilization while maintaining the fairness among flows, we use both the metrics in our evaluation. Network utility is measured as the sum of logarithm of flow throughput rates ($\sum_{f \in F} log(x_f)$). For throughput fairness of flows, we use Jain's index [21] which is defines as –

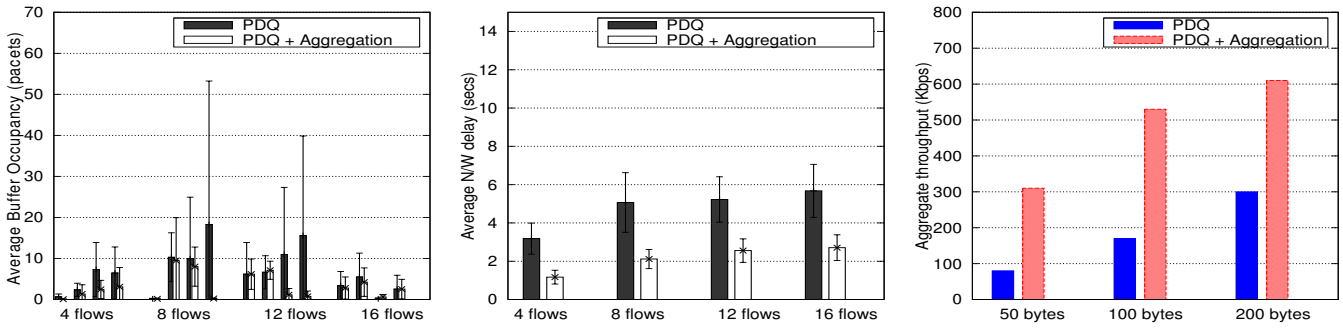
$$\text{Fairness index} = \frac{(\sum_{f=1}^{m} x_f)^2}{m \sum_{f=1}^{m} {x_f}^2} \tag{5}$$

where m is total number of flows in the network and x_f is the throughput of every flow.

Fig. 1a shows the aggregate throughput achieved with both back-pressure and aggregation based back-pressure schemes. The aggregation based strategy improves the throughput significantly mainly because the backlogged queues are served at faster rates when aggregation is utilized. Without aggregation, every packet of backlogged queues requires additional time for medium access. Also, overhead of MAC layer headers for each frame without aggregation also accounts for a large wastage of bandwidth. Both these issues are well addressed by the utilizing aggregation along with back-pressure policy.

As we described before, the back-pressure strategy was mainly devised to address low network utilization and unfairness issues in multi-hop wireless networks. Even though aggregation with back-pressure scheduling increases the network throughput, it is necessary to verify that it does not do so by penalizing network utilization or fairness. Fig. 1b shows the network utility of back-pressure scheme with and without aggregation. It can be observed that in fact aggregation can achieve same or sometimes more network utilization than back-pressure scheme without aggregation. This is mainly due to higher service rates of back-logged queues which in turn allows rate control to insert more and more packets in the flow, yielding an improved utilization. Similarly, Fig. 1c shows the fairness index calculated for both schemes. It is observed that aggregation also increases the throughput fairness in most cases. This is attributed to the fact that aggregation increases the air time utilization of back-logged queues more aggressively which results into improved balance among flow rates.

Another important performance metric related to back-pressure strategies is the average buffer occupancy of nodes. Since back-pressure framework guarantees a throughput optimal solution while stabilizing the queue, average buffer occupancy displays how fast a given scheme transfers traffic from highly back-logged queues to lesser back-logged queues. Fig. 2a compares average buffer occupancy of four randomly chosen nodes in both schemes. It shows that aggregation based policy aggressively attempts to reduce the size of back-logged queues which in turn reduces the overall buffer occupancy. Lastly, we compare the average per packet delay in both the schemes. This is especially an important metric because many real-time, time-sensitive traffic flows in practice have tight delay constraints. The back-pressure policy without aggregation suffers from higher per packet delay mainly because of the time overhead of medium access before transmitting every small or large packet. With aggregation, average per packet medium access time at every node along its path is reduced drastically as shown in Fig. 2b. We have confirmed the above presented results for other node placements such as random



(a) Average buffer occupancy (b) Average network delay (c) Aggregate throughput on Linux testbed

Fig. 2: Comparison of back-pressure scheduling with and without aggregation (a-b) simulation (c) testbed experiments

and clustered, and also for different flow input rates. We could not include these results here due to space limitations.

We have implemented aggregation based back-pressure scheme on our wireless mesh testbed. We modify Linux kernel network stack to implement per-destination queuing and back-pressure scheduling. Every node was equipped with a wireless radio which is operated using MadWiFi [22] device driver. To enable MAC prioritization, we used modified MadWiFi driver provided by [23] group. We use a 4 node topology for experimentation where node A initiates two flows destined to node C and Node D. An intermediate node B is utilized by node A to forwards packets of both flows which yields two routing paths (A – B – C) and (A – B – D). All four nodes where in the same contention neighborhood. We use three different packet sizes of 50, 100 and 200 bytes. The throughput results of the experiments are presented in Fig. 2c. The aggregation based back-pressure scheduling outperforms the scheme without aggregation by achieving at least double aggregate throughput.

V. Conclusions

In this work, we presented an aggregation based back-pressure scheduling policy for multi-hop wireless networks. We found that aggregation can be used intelligently to further increase the rates at which back-logged queues are served. Since aggregation is performed using back-pressure principles, the presented scheme achieves higher throughput and delay performance while preserving the network utility and fairness gains of back-pressure scheduling. Simulation and testbed experiments confirm the feasibility of implementation and performance improvements of the scheme.

References

[1] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," dec 1990, pp. 2130 –2132 vol.4.

[2] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," in *Journal of the Operational Research Society*, vol. 49, 1998. [Online]. Available: http://citeseer.ist.psu.edu/kelly98rate.html

[3] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle, "Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, 2006, pp. 1 –13.

[4] A. Eryilmaz and R. Srikant, "Joint congestion control, routing, and mac for stability and fairness in wireless networks," *Selected Areas in Communications, IEEE Journal on*, vol. 24, no. 8, pp. 1514 –1524, 2006.

[5] X. Lin and N. Shroff, "Joint rate control and scheduling in multihop wireless networks," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 2, 2004, pp. 1484 – 1489 Vol.2.

[6] M. Neely, E. Modiano, and C.-P. Li, "Fairness and optimal stochastic control for heterogeneous networks," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 3, 2005, pp. 1723 – 1734 vol. 3.

[7] M. J. N. Georgiadis Leonidas and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," in *Foundations and Trends in Networking*, 2006.

[8] U. Akyol, M. Andrews, P. Gupta, J. Hobby, I. Saniee, and A. Stolyar, "Joint scheduling and congestion control in mobile ad-hoc networks," april 2008, pp. 619 –627.

[9] A. Warrier, S. Janakiraman, S. Ha, and I. Rhee, "Diffq: Practical differential backlog congestion control for wireless networks," april 2009, pp. 262 –270.

[10] D. Tang and M. Baker, "Analysis of a local-area wireless network," in *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2000, pp. 1–10.

[11] C. Na, J. Chen, and T. Rappaport, "Measured traffic statistics and throughput of ieee 802.11b public wlan hotspots with three different applications," *Wireless Communications, IEEE Transactions on*, vol. 5, no. 11, pp. 3296 –3305, november 2006.

[12] A. Jain, M. Gruteser, M. Neufeld, and D. Grunwald, "Benefits of packet aggregation in ad-hoc wireless network," Tech. Rep., 2003.

[13] D. Kliazovich and F. Granelli, "Packet concatenation at the ip level for performance enhancement in wireless local area networks," *Wirel. Netw.*, vol. 14, no. 4, pp. 519–529, 2008.

[14] R. Raghavendra, A. Jardosh, E. Belding, and H. Zheng, "Ipac: Ip-based adaptive packet concatenation for multihop wireless networks," 29 2006-nov. 1 2006, pp. 2147 –2153.

[15] S. Ganguly, V. Navda, K. Kim, A. Kashyap, D. Niculescu, R. Izmailov, S. Hong, and S. Das, "Performance optimizations for deploying voip services in mesh networks," *Selected Areas in Communications, IEEE Journal on*, vol. 24, no. 11, pp. 2147 –2158, nov. 2006.

[16] R. Riggio, D. Miorandi, F. De Pellegrini, F. Granelli, and I. Chlamtac, "A traffic aggregation and differentiation scheme for enhanced qos in ieee 802.11-based wireless mesh networks," *Comput. Commun.*, vol. 31, no. 7, pp. 1290–1300, 2008.

[17] H. Zhai and Y. Fang, "A distributed packet concatenation scheme for sensor and ad hoc networks," oct. 2005, pp. 1443 –1449 Vol. 3.

[18] L. Ying, S. S., R. A., and S. Liu, "On combining shortest-path and back-pressure routing over multihop wireless networks," *Networking, IEEE/ACM Transactions on*, vol. PP, no. 99, p. 1, 2010.

[19] http://www.opnet.com.

[20] K. Thompson, G. J. Miller, and R. Wilder, "Wide-area internet traffic patterns and characteristics," *IEEE Network*, vol. 11, pp. 10–23, 1997.

[21] D.-M. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Computer Networks and ISDN Systems*, vol. 17, no. 1, pp. 1 – 14, 1989.

[22] http://madwifi-project.org/.

[23] http://hop.cs.umass.edu/.