This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2010 proceedings.

A Probabilistic Approach for Improving TCP Fairness across Multiple Contending WLANs

Ming Zhang S. M. Iftekharul Alam Shigang Chen
Computer & Information Science & Engineering
University of Florida, Gainesville, FL 32611, USA

Jianwei Liu
Electronic & Information Engineering
Beihang University, Beijing, 100191, China

***Abstract*—Contention among multiple nearby WLANs in urban areas may cause severe TCP unfairness, where some TCP flows can achieve very high throughput at the expense of starving others. This unfairness results from the fact that different physical nodes conveying TCP flows at a wireless bottleneck may have different channel observations and consequently they may provide inconsistent feedbacks to the TCP sources. Existing solutions to this problem try to synchronize channel observations of contending nodes by exchanging control messages among them. They rely on the assumption that these nodes are within each other's transmission range, which however may not always hold. In this paper, we propose a new approach, called *Wireless Probabilistic Drop* (WPD), to improve TCP fairness without requiring direct communication among nodes. In WPD, when a node detects congestion, it probabilistically chooses to either drop some packets to resolve the congestion, or aggressively spread the congestion signal to other contending nodes. Each node makes the choice with a probability that is proportional to its flow rate. Henceforth, high-rate flows tend to perform rate reduction more often, and low-rate flows are more likely to increase their flow rates. Eventually, all flows passing the bottleneck are expected to get a fair share of the channel bandwidth. Extensive simulations in ns-2 demonstrate that WPD can significantly improve fairness among TCP flows across multiple contending WLANs.**

I. Introduction

IEEE 802.11 Wireless LANs (WLANs) have been densely deployed in many urban areas [1]. A typical WLAN consists of one access point and several wireless clients. With a limited number of non-overlapping channels in 802.11 wireless networks, multiple overlapping WLANs will inevitably contend for the same channel. TCP is the dominating transport layer protocol used by many applications over WLANs [2], [3]. People surf websites, chat by using instant messengers and download files through TCP connections. However, it has been observed that TCP may demonstrate a severe fairness problem among contending WLANs, where TCP flows in some WLANs may achieve very high throughput at the expense of others in nearby WLANs [4], [5].

Based on network traffic conditions, TCP dynamically adjusts the source sending rate by performing AIMD (Additive Increase Multiplicative Decrease) on its congestion window. AIMD works well in wired networks, where a bottleneck router detects congestion and notifies the TCP sources of the congestion. However, it may not always work well in wireless networks because TCP flows may pass through different physical nodes that contend in a bottleneck channel. These nodes may have different channel observations and different capabilities of obtaining the channel for transmissions [6], [7]. Consequently, when only some nodes (but not all) detect channel congestion, the TCP flows that pass those nodes will receive congestion feedback but the flows passing other nodes will not. For TCP to achieve fairness, congestion feedback must be consistent across all flows that contend in the bottleneck. Inconsistent feedback to the flow sources will render TCP ineffective in its rate control among contending flows.

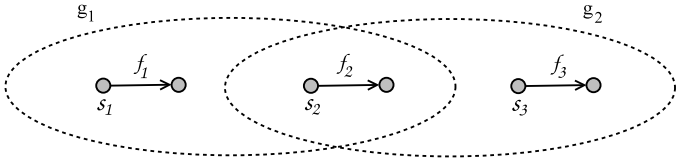

Fig. 1. Three partially overlapping WLANs form two contention groups g_1 and g_2. s_1, s_2 and s_3 are three access points. A TCP flow from a server on the Internet passes through an access point to a wireless client in each WLAN. Note that only the wireless part of a flow (from an access point to a wireless client) is drawn in the figure.

Fig. 1 shows an example of three partially overlapping WLANs, where s_1, s_2 and s_3 are access points. A TCP flow from a server on the Internet passes through an access point to a wireless client in each WLAN. Note that we only draw the wireless part of each TCP flow (from the access point to the wireless client) in all figures throughout the paper. Suppose s_1 and s_2 contend and they form a contention group g_1. Nodes s_2 and s_3 also contend, and they form another contention group g_2. But s_1 and s_3 do not contend; they can transmit simultaneously. Because s_2 contends with both s_1 and s_3, it senses a busier channel and is less capable of accessing the channel due to heavier contention. Hence, when the channel is saturated, as its queue builds up and exceeds a threshold, s_2 is likely to detect the congestion first. At the time when s_2 detects congestion, s_1 and s_3 may still have small queues since they are more capable of sending out their packets due to less contention (as we observe consistently in our ns-2 simulations). If s_2 immediately drops packets to inform the source of f_2 to reduce its sending rate, the congestion will be resolved while the flows passing s_1 and s_3 are still performing additive increase. In this case, f_1 and f_3 will seize the channel bandwidth given up by s_2. Hence, the TCP rate adaptation actually promotes unfairness among the rates of f_1, f_2 and f_3 because it penalizes the flow that experiences heavier contention (and thus performs

978-1-4244-5638-3/10/$26.00 ©2010 IEEE

multiplicative decrease more frequently).

To solve this problem, Xu *et al.* [4] proposed NRED (Neighborhood Random Early Detection). The basic idea is to make sure that all contending nodes observe the same channel condition. For instance, in Fig. 1, s_1, s_2 and s_3 periodically synchronize their channel observations through exchanging control messages. Therefore, they can detect channel congestion simultaneously and then drop packets to notify the TCP sources to properly perform rate reduction, which leads to enhanced fairness. However, due to the disparity among transmission/interference/carrier-sensing ranges [8], nodes that are not able to directly communicate may still contend in the same channel. In Fig. 1, if s_1, s_2 and s_3 are outside of each other's transmission range, NRED will fail because these nodes cannot synchronize their channel observations and provide consistent feedbacks to the TCP sources.

In this paper, we propose a new solution, WPD (Wireless Probabilistic Drop), for improving TCP fairness *without requiring any means of direct communication among contending nodes*. It is a fully distributed solution that does not require any modification to the operational protocols of TCP and 802.11 DCF (except for certain MAC parameter changes during congestion). We use the example in Fig. 1 to illustrate the basic idea. Suppose s_2 detects congestion first. Instead of dropping packets immediately, our solution requires s_2 to make a probabilistic choice between two states: *resolution* or *signaling*. In the resolution state, s_2 will drop packets to resolve the congestion. In the signaling state, s_2 will not drop packets, but instead it will signal other nodes about the congestion by aggressively pushing more packets into the channel. As s_2 grabs more channel bandwidth, s_1 and s_3 will observe that their queues build up and eventually pass the threshold. Once they detect congestion, they will perform similar operations. In our solution, each node that detects congestion makes a choice between *resolution* and *signaling* states; the probability for choosing the resolution state is proportional to the node's *channel occupancy* (i.e., the fraction of time during which the node occupies the channel for its transmission). Hence, if s_2 has a smaller channel occupancy, it will have a higher probability to choose the signaling state, while s_1 and s_3 are more likely to choose the other state and perform packet dropping. Consequently, f_2 will increase its sending rate more often while f_1 and f_3 perform rate reduction, which shrinks the gap between the rate of f_2 and the rates of other flows. Under such dynamics, our simulations demonstrate that all contending flows receive fair shares of the channel bandwidth over the long run in Fig. 1 as well as in other more complex scenarios.

The rest of the paper is organized as follows. Section II gives the network model. Section III proposes our WPD protocol. Section IV presents the simulation results. Section V discusses the related work. Section VI draws the conclusion.

II. Network Model

We consider a common scenario where multiple WLANs coexist in an area. A typical WLAN consists of an access point and multiple wireless clients. We assume IEEE 802.11 a/b/g DCF or other DCF-like protocols at the MAC layer. Each WLAN selects one channel (a sub-band of frequency range) for data transmissions between the access point and clients. With only a limited number of non-overlapping channels in 802.11 wireless networks, a WLAN may contend in the same channel with nearby WLANs.

We consider TCP flows (or TCP connections) between wireless clients and Internet servers. Each flow passes an Internet wired path and a final wireless link across a WLAN. TCP performs well for congestion on the wired path, but not for the wireless link, which is the focus in this paper. It is well known that packet drops due to wireless transmission failures can interfere with TCP congestion signaling. This problem has been studied extensively and can be largely solved by increasing the number of retransmission attempts. This paper investigates a less-studied issue, i.e., how inconsistent channel observations by different contending nodes will cause the TCP rate adaptation to fail in achieving its fairness objective (see Section I) and how to solve this problem.

The *flow rate* of a TCP connection is defined as the number of packets that the wireless client successfully receives per second. The *sending rate* is defined as the number of packets that the source of a TCP flow (e.g., a server on the Internet) sends out per second. When a TCP flow crosses a wireless link, the *channel occupancy* of the wireless node that forwards the packets is defined as the fraction of time that the node occupies the channel for its data delivery (i.e., DATA/ACK exchanges).

Both access points and wireless clients are referred to as *nodes*. Two nodes *contend* if one's data transmission can make the other sense a busy channel or corrupt the other's data reception. A group of mutually contending nodes forms a *contention group*. A node may participate in multiple contention groups. We use the concept of contention group only to simplify the presentation of the paper; the operations in our solution do not need to know the actual contention relationship among wireless nodes. In the discussion of a wireless node carrying a TCP connection, we often use the *flow rate of the wireless node* to refer to the flow rate of the TCP connection that the node carries.

III. Wireless Probabilistic Drop (WPD)

In this section, we first explain the basic idea behind our solution WPD (Wireless Probabilistic Drop), then introduce several rate control techniques, and finally describe how they work together to enhance TCP fairness.

A. Basic Idea

Each wireless node monitors the size of the local MAC-layer queue and measures its channel occupancy. A node detects channel congestion when its queue size exceeds a certain threshold[1]. If the node that detects congestion drops packets immediately, it may cause unfairness because the contending nodes in the same saturated channel may not yet

[1]The same method is used in [7], whose focus is however on MAC flows, instead of TCP flows.

detect congestion and hence they may not drop packets. To solve this problem, instead of dropping packets immediately, we make the node to transit from its *normal* state to either *resolution* or *signaling* state for a period of time T (such as 1 second in our simulations).

In the resolution state, the node emulates RED (Random Early Detection) [9] by probabilistically dropping packets to resolve congestion. It causes the TCP source to observe triple duplicate ACKs and consequently reduce the sending rate by shrinking the congestion window. In the signaling state, instead of dropping packets, the node aggressively competes for channel access by modifying its MAC parameters, such as reducing the minimum contention window. This is called the *aggressive mode*. The past research has shown that reducing the minimum contention window is very effective for enhancing a node's ability to occupy the channel [10]. As this node enters the aggressive mode for a period of time and consumes more channel bandwidth, other contending nodes send out fewer packets. They will observe their queues build up and exceed the threshold. In this way, the congestion signal is spread to these nodes without explicit communication, causing them to transit to either resolution or signaling state.

The probability for a node to choose the resolution state is proportional to the node's current channel occupancy. Hence, nodes with low channel occupancies tend to select the signaling state and grab more channel bandwidth, while nodes with high channel occupancies are more likely to select the resolution state that causes the traversing TCP flows to reduce rates, which helps achieving fairness. It may happen that the contending nodes all choose the signaling state. In this case, some nodes will observe queue overflows, and the standard taildrop is performed to resolve congestion.

Below we give more detailed description for various techniques that are employed by WPD.

B. Periodical Measurement of State Information

Each wireless node periodically measures its average queue size $\bar{q}$ and average channel occupancy $\bar{u}$ to learn the channel condition. It also measures the average rate $\bar{r}$ of each TCP flow that it carries. At the end of every measurement period m (such as 0.1 second), the new value of $\bar{q}$ is computed as $\bar{q} := (1-w) \times \bar{q} + w \times q$, where $:=$ is the assignment operator, q is the current queue size, and $0 < w < 1$, which is the parameter for weighted moving average. Similarly, $\bar{u} := (1-w) \times \bar{u} + w \times u$, where u is the current channel occupancy, and $\bar{r} := (1-w) \times \bar{r} + w \times (pks/m)$, where pks is the number of packets successfully delivered by the wireless node during the measurement period.

C. Aggressive Mode

In the 802.11 DCF, after a node successfully transmits a packet, it randomly picks a number from [0, CW_{min}] as the value of the backoff timer for its next packet. CW_{min} is known as the *minimum contention window*, which is the initial window size for the exponential backoff algorithm. The default value of CW_{min} is 31. In WPD, a node in the signaling state will enter the aggressive mode by temporarily reducing CW_{min} to a small value (such as 3 in our simulations) for a period of time. With a reduced value of CW_{min}, the node is more capable of obtaining the channel than other contending nodes. As this node aggressively occupies the channel, less bandwidth is left for others, whose queue lengths are then forced up. Hence, the aggressive mode is used by a node that has detected congestion to spread the congestion signal to other contending nodes.

D. Probabilistic Dropping

In the resolution state, a node resolves congestion by dropping packets, which informs the TCP sources to reduce their sending rates. We adopt a probabilistic dropping algorithm that emulates the RED [9] in wired networks: A base probability p_b is first computed as $p_{max} \times \bar{u}$, where p_{max} is a predefined maximum packet dropping probability. Then the node drops each arrival packet with a probability $p_a := p_b/(1 - count \times p_b)$, where $count$ is the number of packets that have been forwarded since the previous packet drop. Hence, the dropping probability is an increasing function of the node's channel occupancy. A node with higher channel occupancy will drop more packets in its resolution state, which helps improve fairness among the contending nodes.

E. Minimum Rate Assurance

To avoid starvation, it is desirable to ensure that each TCP connection has a minimum rate r_{min}, even for one that is carried by a wireless node under the heaviest contention. As an optimization, when a wireless node detects that the average flow rate $\bar{r}$ is below r_{min}, it enters the aggressive mode until $\bar{r}$ reaches r_{min}. Ensuring the minimum flow rate has a positive impact in WPD because it assures that a node in the signaling state will have enough arrival packets to occupy the channel in order to spread the congestion signal. Without enough arrival packets, the node would have to inject fake packets to occupy the channel, which wastes bandwidth.

F. Adaptive Intermittent Release

When multiple wireless nodes contend in the same channel, if all of them continuously compete for channel access, collision can happen frequently. We design a method called *adaptive intermittent release* to interrupt the pattern of continuous channel access. A node stores the received packets in a network-layer queue and releases the packets to a MAC-layer queue for transmission. Based on the channel condition, we control the time t_{ips} (called Inter-Packet Spacing) between two consecutive releases. After the MAC layer transmits all its packets, it may have to wait for the network layer to release the next one. We observe that, at the congestion time, such wait helps reduce collisions.

The adaptation of t_{ips} is described as follows: When a node performs probabilistic dropping in its resolution state, it performs multiplicative increase on t_{ips} such that $t_{ips} := (1 + \alpha) \times t_{ips}$ after each measurement period m until the congestion is resolved (i.e., the average queue length falls below the threshold), where α is the factor of

multiplicative increase. In all other cases, the node performs additive decrease on t_{ips} such that $t_{ips} := t_{ips} - \beta$, where β is a constant value. If t_{ips} becomes less than $50\mu s$, we set it to be $50\mu s$. The rationale behind the above adaptation is to keep t_{ips} small unless congestion occurs. When that happens, we increase the inter-packet spacing for nodes in the resolution state. As we will see in the next subsection, nodes with higher channel occupancies are more likely to be in this state. Their larger inter-packet spacings make it easier for other nodes with lower channel occupancies to access the channel during congestion.

When adaptive intermittent release is used, the queue length $\bar{q}$ is measured based on the combined length of the network-layer queue and the MAC-layer queue.

G. WPD Protocol

We now assemble the above rate control techniques to form WPD. Each node can be in one of three states: *normal*, *resolution* and *signaling*.

In the normal state, a node releases packets from the network layer to the MAC layer through adaptive intermittent release. After each measurement period m, the node computes $\bar{q}$, $\bar{u}$ and $\bar{r}$. When $\bar{q}$ is below a threshold H, it additively decreases t_{ips}. Once $\bar{q}$ exceeds H, it draws a random number rd from $[0, 1]$. If $rd \leq \bar{u}$, the node transits to the resolution state to perform probabilistic dropping; otherwise, if $rd > \bar{u}$, it transits to the signaling state to enter the aggressive mode. It is clear that the probability for the resolution state is proportional to the channel occupancy.

In the resolution state, a node performs probabilistic dropping to inform the TCP source to reduce its sending rate. In the mean time, it performs multiplicative increase on t_{ips} to give other contending nodes more chance of accessing the channel. After a period of time T, the node transits to the normal state.

In the signaling state, a node uses the aggressive mode to consume more channel bandwidth in order to spread the congestion signal to other nodes. After a period of time T, the node transits to the normal state. When the minimum rate assurance is implemented, a node may also temporarily enter the aggressive mode to bring up its flow rate if it is too low.

The pseudo code for the operations of WPD is given below.

Wireless Probabilistic Drop

1. at the end of each measurement period m
2. $\quad \bar{q} := (1 - w) \times \bar{q} + w \times q$
3. $\quad \bar{u} := (1 - w) \times \bar{u} + w \times u$
4. $\quad \bar{r} := (1 - w) \times \bar{r} + w \times (pks/m)$
5. **if** $\bar{q} > H$ **then**
6. $\quad rd_1 := random(0, 1)$
7. $\quad$ **if** $rd_1 > \bar{u}$ **then**
8. $\quad\quad$ **during** a period T **do**
9. $\quad\quad\quad CW_{min} := 3$ for aggressive mode
10. $\quad\quad$ **end during**
11. $\quad\quad$ restore CW_{min} to the original value
12. $\quad$ **else**
13. $\quad\quad t_{ips} := (1 + \alpha) \times t_{ips}$
14. $\quad\quad$ **during** a period T **do**
15. $\quad\quad\quad p_b := p_{max} \times \bar{u}$
16. $\quad\quad\quad count := 0$
17. $\quad\quad\quad$ **for each** arrival packet **do**
18. $\quad\quad\quad\quad count := count + 1$
19. $\quad\quad\quad\quad p_a := p_b/(1 - count \times p_b)$
20. $\quad\quad\quad\quad$ **if** $p_a < 0$ **then** $p_a = 1$
21. $\quad\quad\quad\quad rd_2 := random(0, 1)$
22. $\quad\quad\quad\quad$ **if** $rd_2 < p_a$ **then**
23. $\quad\quad\quad\quad\quad$ drop the arrival packet
24. $\quad\quad\quad\quad\quad count := 0$
25. $\quad\quad\quad$ **end for**
26. $\quad\quad$ **end during**
27. **else**
28. $\quad t_{ips} := t_{ips} - \beta$
29. $\quad$ **if** $t_{ips} < 50\mu s$ **then** $t_{ips} := 50\mu s$
30. $\quad$ **if** $\bar{r} < r_{min}$ **then**
31. $\quad\quad CW_{min} := 3$ for aggressive mode
32. $\quad$ **else** restore CW_{min} to the original value

IV. Simulation

We evaluate the proposed solution WPD by simulations.

A. Simulation Setup

All simulations are performed in ns-2 [11]. We use TCP NewReno at the transport layer. The size of a TCP data packet is 1,000 bytes. FTP is used to generate each TCP flow. We use IEEE 802.11 DCF at the MAC layer. RTS/CTS is turned off by default due to its high overhead, which is today's common practice. For the transmission range and the carrier-sensing range, we use ns-2 default values, which are 250m and 550m, respectively. The interference range is equal to the length of a wireless link times 1.78, which is also the default setting in ns-2. The transmission rate is 11Mbps. We compare WPD with DropTail (which drops packets when the queue is overflowed) and NRED [4], whose parameters are chosen based on the original paper. We set the parameters for WPD as follows: the rate measurement period m is 0.1 second, the threshold H is 5 packets, the state period T is 1 second, the minimum rate $rate_{min}$ is 25 packets per second, the maximum drop probability p_{max} is 0.03, the weighting factor w is 0.20, the α is 2 and the β is $50\mu s$. Each simulation is executed for 150 seconds, and the average TCP rates are reported.

B. Fairness Index

We use the theoretically-computed rates under the *proportional fairness* model [12] as the benchmarks for comparison. Proportional fairness strives to balance between the fairness requirement and the overall network throughput. Besides these benchmark flow rates, we also compute an overall fairness index, which is the summation of a utility function, $\sum_{f \in F} \ln r_f$, where r_f is the rate of a TCP flow f and F is the set of flows. Our computed benchmark rates maximize this index. Among the fairness solutions under comparison, one that achieves a higher value of the fairness index has better performance in terms of proportional fairness.

C. Case Study: A Base Scenario

We first perform a case study on a base scenario in Fig. 1, where three access points s_1, s_2 and s_3 form two contention groups even though they are placed outside of each other's transmission range. We observe that unfairness arises among

TABLE I
COMPARING THE FLOW RATES (IN PACKETS/SEC) OF THE FLOWS IN THE NETWORK OF FIG. 1 UNDER DROPTAIL, NRED AND WPD.

	f_1	f_2	f_3
DropTail	433.34	1.60	430.58
NRED	379.97	17.48	377.00
aggressive mode + probabilistic dropping	379.76	35.83	378.75
aggressive mode + probabilistic dropping + minimum rate assurance	339.43	74.75	343.20
aggressive mode + probabilistic dropping + minimum rate assurance + adaptive intermittent release (WPD)	252.34	126.46	260.54
Optimal Proportional Fairness	288.67	144.33	288.67

TABLE II
COMPARING THE FAIRNESS INDEX IN TERMS OF PROPORTIONAL FAIRNESS IN THE NETWORK OF FIG. 1 UNDER DROPTAIL, NRED AND WPD.

DropTail	NRED	WPD	Optimal
12.61	14.73	15.93	16.30

the three TCP flows, f_1, f_2 and f_3. As shown in Table I, when DropTail is used as the queue management scheme, f_1 and f_3 can both obtain flow rates above 430 packets per second, whereas f_2 is almost starved. In this case, without the ability for the access points to explicitly synchronize their channel observations (due to being outside of each other's transmission range), NRED makes little improvement. For WPD, we incrementally deploy the techniques proposed in Section III to demonstrate their individual impact on the performance. We first adopt the aggressive mode (Section III-C) and probabilistic dropping (Section III-D). Table I shows that the flow rate of f_2 is increased to 35.83 packets per second. After the minimum rate assurance (Section III-E) is incorporated, the flow rate of f_2 is increased to 74.75 packets per second, which demonstrates its positive impact on the performance of WPD. Finally, we apply the adaptive intermittent release (Section III-F), which further improves fairness. The flow rate of f_2 is increased to 126.46 packets per second. The final results of WPD are close to the theoretical flow rates under the proportional fairness model that are shown in the last row of the table.

We compare WPD with DropTail and NRED in terms of their fairness indices in Table II, which shows that WPD has the highest index value, close to the optimal index value that is achievable under the proportional fairness model.

D. Scalability Study: Three Contention Groups

Next we evaluate the performance of WPD on a scenario with more than two contention groups. Fig. 2 has six TCP flows belonging to three contention groups. The simulation results in Table III show that f_3 has very low throughput values under DropTail and NRED. WPD improves it to 75.30 packets per second. It is worth noting that comparing with DropTail and NRED, WPD also achieves better fairness among the other five flows.

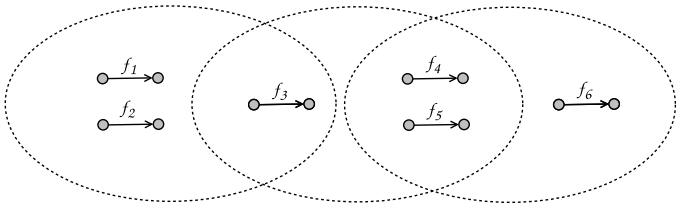

Fig. 2. Six WLANs form three contention groups and each WLAN contains one TCP flow.

TABLE III
COMPARING THE FLOW RATES (IN PACKETS/SEC) OF THE FLOWS IN THE NETWORK OF FIG. 2 UNDER DROPTAIL, NRED AND WPD.

flow	DropTail	NRED	WPD
f_1	221.65	203.17	153.03
f_2	234.10	196.89	189.98
f_3	7.50	22.29	75.30
f_4	205.91	154.11	136.07
f_5	205.28	154.71	135.42
f_6	55.58	114.38	145.17

E. A Street Scenario

Fig. 3 shows a more complicated scenario, where twenty-four TCP flows are carried by WLANs randomly deployed along two crossing streets. The relative positions of the nodes are drawn in the figure. The length of each wireless link is 150m. The contention relationship among the flows, which is much more complicated than those in the previous scenarios, is automatically determined by ns-2.

Table IV shows that, under DropTail, six TCP flows are starved (less than 10 packets per second) and five TCP flows have low flow rates (less than 40 packets per second). NRED performs better than DropTail. But it still has three starved TCP flows and three low-rate flows. Under WPD, all flows can achieve decent throughput values. We believe that this simulation demonstrates the effectiveness of WPD in complex settings.

TABLE IV
COMPARING THE FLOW RATES (IN PACKETS/SEC) OF THE FLOWS IN THE NETWORK OF FIG. 3 UNDER DROPTAIL, NRED AND WPD.

flow	DropTail	NRED	WPD	flow	DropTail	NRED	WPD
f_1	216.4	235.4	163.9	f_{13}	0.8	3.5	45.6
f_2	215.9	174.4	84.2	f_{14}	191.5	181.4	162.6
f_3	1.1	6.8	163.9	f_{15}	242.3	226.7	181.1
f_4	205.2	189.5	165.0	f_{16}	0.4	0.8	52.9
f_5	218.6	205.6	99.8	f_{17}	217.4	207.3	194.5
f_6	9.8	25.5	106.4	f_{18}	214.2	209.5	167.4
f_7	10.2	91.9	89.4	f_{19}	388.9	198.9	125.9
f_8	23.5	95.4	124.4	f_{20}	17.0	117.0	168.0
f_9	382.3	183.0	144.6	f_{21}	402.9	237.4	155.1
f_{10}	7.1	38.2	88.2	f_{22}	4.0	26.1	84.9
f_{11}	30.6	104.0	147.7	f_{23}	26.2	93.6	146.7
f_{12}	397.7	286.3	218.6	f_{24}	403.2	301.5	217.4

978-1-4244-5638-3/10/$26.00 ©2010 IEEE

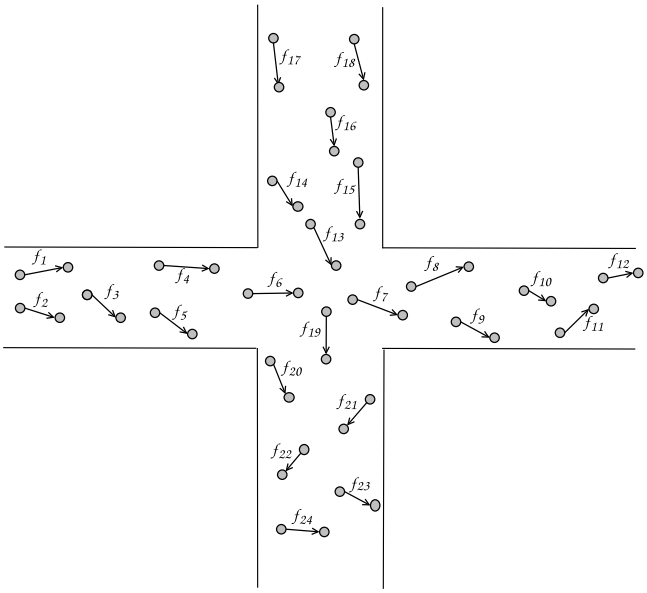

Fig. 3. Twenty four WLANs are randomly deployed along two crossing streets and each WLAN contains one TCP flow.

V. Related Work

Ergin *et al.* [3] verified the TCP performance degradation by testbed traces in an unplanned deployment of WLANs. Some work has been proposed to mitigate the interference between two nearby WLANs by transceiver parameter optimization [13], channel assignment [14] and association control [15]. However, they may cause channel underutilization or incur long delays. Moreover, they cannot solve the TCP fairness problem in a dense deployment.

Yang *et al.* [5] proposed a non-work-conserving scheduling for improving TCP fairness among flows crossing wireless ad hoc networks and wired networks. The basic idea is to set a timer to control the speed of sending packets to the MAC layer. The length of the timer is determined by the flow rate. This approach is simple, but it significantly downgrades the overall throughput.

Another line of research is to address TCP upstream/downstream fairness problem in WLANs [16], [17], where upstream flows may gain much more channel bandwidth than downstream flows. All these solutions focus on the TCP unfairness within a single WLAN.

There is a large body of work on fairness issues over multihop wireless networks [18], [19], [20]. However, the TCP unfairness among nearby WLANs studied in this paper has different emphasis. In a multihop network, a flow can carry certain information for the nodes along its routing path [18], [19]. Contrarily, two WLANs may contend in the same channel but they have no means of direct communication.

VI. Conclusion

In this paper, we propose WPD for achieving TCP fairness among multiple contending WLANs. It is a fully distributed solution only based on local information. It can work seamlessly with current TCP and MAC standards. It implicitly spreads congestion information to all contending flows in a bottleneck without requiring direct communications. Extensive simulations show that it can significantly improve TCP fairness in various scenarios when other existing solutions fail.

VII. Acknowledgments

This work was supported in part by the US National Science Foundation under grant CNS-0644033 and grant CPS-0931969.

References

[1] RSA Press Release, "Annual RSA Wireless Security Survey: With Encrypted Wi-Fi Vulnerable, How Companies and Individuals are Risking their Assets and Reputations," *http://www.rsa.com/press_release.aspx?id=9725*, October 2008.

[2] X. Chen, H. Zhai, J. Wang, and Y. Fang, "A Survey on Improving TCP Performance over Wireless Networks," *Resource Management in Wireless Networking*, vol. 16, pp. 657–695, 2005.

[3] M. A. Ergin, K. Ramachandran, and M. Gruteser, "An Experimental Study of Inter-cell Interference Effects on System Performance in Unplanned Wireless LAN Deployments," *ACM Computer Networks*, vol. 52, no. 14, pp. 2728–2744, October 2008.

[4] K. Xu, M. Gerla, L. Qi, and Y. Shu, "Enhancing TCP Fairness in Ad Hoc Wireless Networks Using Neighborhood RED," In *Proc. of ACM MobiCom*, September 2003.

[5] L. Yang, W. Seah, and Q. Yin, "Improving Fairness among TCP Flows Crossing Wireless Ad Hoc and Wired Networks," In *Proc. of ACM MOBIHOC*, June 2003.

[6] M. Garetto, T. Salonidis, and E. W. Knightly, "Modeling Per-flow Throughput and Capturing Starvation in CSMA Multi-hop Wireless Networks," In *Proc. of IEEE INFOCOM*, April 2006.

[7] Y. Jian and S. Chen, "Can CSMA/CA Networks Be Made Fair?," In *Proc. of ACM MobiCom*, September 2008.

[8] K. Xu, M. Gerla, and S. Bae, "How Effective is the IEEE 802.11 RTS/CTS Handshake in Ad Hoc Networks?," In *Proc. of IEEE GLOBECOM*, November 2002.

[9] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, August 1993.

[10] M. Zhang, S. Chen, and Y. Jian, "MAC-layer Time Fairness across Multiple Wireless LANs," In *Proc. of IEEE INFOCOM*, March 2010.

[11] "The Network Simulator - ns-2," *http://www.isi.edu/nsnam/ns/*.

[12] F. Kelly, A. Maulloo, and D. Tan, "Rate Control in Communication Networks: Shadow Prices, Proportional Fairness and Stability," *Journal of the Operational Research*, vol. 49, no. 3, pp. 237–252, 1998.

[13] A. Akella, G. Judd, S. Seshan, and P. Steenkiste, "Self-management in Chaotic Wireless Deployments," In *Proc. of ACM MobiCom*, August 2005.

[14] A. Mishra, V. Shrivastava, D. Agarwal, S. Banerjee, and S. Ganguly, "Distributed Channel Management in Uncoordinated Wireless Environments," In *Proc. of ACM MobiCom*, September 2006.

[15] Y. Bejerano, S.-J. Han, and L. E. Li, "Fairness and Load Balancing in Wireless LANs Using Association Control," In *Proc. of ACM MobiCom*, September 2004.

[16] E. C. Park, D. Y. Kim, H. Kim, and C. H. Choi, "A Cross-Layer Approach for Per-Station Fairness in TCP over WLANs," *IEEE Transactions on Mobile Computing*, vol. 7, no. 7, pp. 898–911, July 2008.

[17] B. A. H. S. Abeysekera, T. Matsuda, and T. Takine, "Dynamic Contention Window Control Mechanism to Achieve Fairness between Uplink and Downlink Flows in IEEE 802.11 Wireless LANs," *IEEE Transactions on Wireless Communications*, vol. 7, no. 9, pp. 3517–3525, September 2008.

[18] S. Rangwala, A. Jindal, K. Y. Jang, K. Psounis, and R. GovindanL, "Understanding Congestion Control in Multi-hop Wireless Mesh Networks," In *Proc. of ACM MobiCom*, September 2008.

[19] H. Zhai, X. Chen, and Y. Fang, "Improving Transport Layer Performance in Multihop Ad Hoc Networks by Exploiting MAC Layer Information," *IEEE Transactions on Wireless Communications*, vol. 6, no. 5, pp. 1692–1701, May 2007.

[20] K. Nahm, A. Helmy, and C. C. J. Kuo, "Cross-Layer Interaction of TCP and Ad Hoc Routing Protocols in Multihop IEEE 802.11 Networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 4, pp. 458–469, April 2008.