Resolving Performance Anomaly using ARF-aware TCP

Seehwan Yoo, Tae-Kyung Kim and Chuck Yoo

Department of Computer Science and Engineering
Korea University

Abstract. In this study, we propose ARF-aware TCP that resolves the performance anomaly in 802.11 WLAN networks. Performance anomaly is a network symptom that fairness among the nodes is broken when multiple nodes in the same channel have different link rates. Recent studies on the performance anomaly focus on QoS mechanism at MAC layer. However, MAC layer approach has drawbacks such as framing overhead or side-effects at transport protocol. ARF-aware TCP successfully provides a fair-share of wireless links in an easier way. By adjusting the congestion window size during a RTT period, we can get high fairness and enhanced throughput. The fairness index in our cases increases from 0.63, 0.74 to 0.99, 0.99 respectively. In addition, it improves the performance of sessions up to 30% by resolving the performance anomaly.

1 Introduction

In WLAN network, it has been reported that performance anomaly occurs when nodes in the same channel have different operating rates[1]. If the nodes have different sending rates, then the channel utilization of each node becomes different. A slow node consumes the wireless channel longer than a faster node. Performance anomaly is a network problem that the fairness of channel utilization among the nodes is broken because of the link rate adaptation mechanism. When the performance anomaly happens, the performance of a faster node is seriously degraded by the neighbor nodes that operate at slower rate regardless of the current operating rate. In performance anomaly scenario, the higher rate node and the lower rate node have the same throughput in the end.

WLAN ARF (Automatic Rate Fallback) mechanism changes the network sending rate as the link status fluctuates[2]. Namely, ARF is not designed for the link fairness but designed to adapt to link status. It changes the channel coding scheme so that redundancy in channel coding hides the wireless channel error. However, ARF makes difficult to keep the fairness among the nodes in the channel. When a slower rate node grabs the channel, it takes longer to transmit the same-length frame than a higher rate node. Although a slower node is more resilient against the wireless channel error, it breaks the fairness. Because WLAN keeps the fairness only based on the medium access probability, a slower rate node consumes more channel time than the other nodes.

The performance anomaly occurs because of this WLAN ARF function. Namely, channel time is unfairly distributed because the slower rate node grabs more channel time than the higher rate node. To resolve the performance anomaly, channel utilization among the nodes should be equally distributed. Recent studies[7, 6, 5, 3, 4] on performance anomaly focus on resolving the anomaly using QoS adaptation at MAC layer.

This paper proposes ARF-aware TCP that is an adaptation protocol for resolving the performance anomaly. Existing study on performance anomaly focuses on MAC layer approach because ARF is a MAC layer function and it can be resolved at MAC layer using framing modification or adjusting a parameter to achieve service differentiation. However, it also has drawbacks as follows: 1) MAC layer QoS adaptation would incur unexpected side-effect on end-to-end rate control protocols such as TCP-Vegas. When ARF is applied, transmission delay is changed, and it affects RTT of the session. When RTT is increased, transport protocol estimates that network is congested, and it starts congestion control algorithm. This results in performance degradation from unnecessary congestion control. 2) MAC layer QoS adaptation is hard to be implemented because it is normally implemented as a firmware. On the other hand, ARF-aware TCP resolves performance anomaly by just modifying a variable to adjust the sending window size.

ARF-aware TCP achieves high fairness among the sessions easily and enhances the performance without side-effects.

2 Related Work

The performance anomaly[1] was analyzed by Heusse. In the study, the performance anomaly happens when there are multiple nodes, on the same wireless channel, that are operating at different rates. The authors have shown that when the performance anomaly happens, throughputs of all the nodes are equalized regardless of own link rate. Namely, the higher rate node should get smaller link utilization and performance is degraded and the throughput drops as much as that of the lower rate node.

802.11e performance extension standard has a TXOP option[8]. Using TXOP option, higher rate node combines multiple frames into one, and sends it once when it grabs the wireless channel. To resolve the unfair network utilization, TXOP option intentionally gives more time to the higher rate node. It makes a fair-share of network resources in terms of link utilization.

Kim et al. presents a novel mechanism to resolve the performance anomaly[3]. Proposed scheme adjusts the parameter in the MAC layer, Contention Window (CW). The authors insist that service differentiation can be achieved by simply adjusting the CW parameter. Because CW decides the probability to access the wireless channel, it increases or decreases the CW as the operating link rate. The scheme resolves performance anomaly by service differentiation. Namely, the link utilization is differentiated and distributed as the link rate, and utilization is kept fairly.

Yoo et al. proposed a scheme to eliminate the performance anomaly[4]; the authors insist that performance anomaly can be eliminated by adjusting the frame size. In the study, they proved that the aggregated throughput becomes higher when it is resolved.

Sadeghi proposed an opportunistic media access mechanism for multi-rate WLAN[5]. In the paper, the authors proposed OAR(Opportunistic Auto Rate) protocol, which improves performance in multi-rate WLAN networks. In the protocol, the sender opportunistically transmits multiple back-to-back data packets whenever the channel is good. The authors insist that OAR achieves a significant throughput gain because the channel coherence times typically exceed multiple packet transmission times for both mobile and non-mobile users.

Godfrey revealed inefficiency in multi-rate WLAN network[6]. The authors proved that multi-rate WLAN DCF function can reach to undesirable Nash equilibrium that makes total network performance degraded. Their idea has proved in analytic method as well as simulation. In another paper[7], he presented that time-based fairness can improve the performance in multi-rate WLAN network. In the study, the authors focus on the impact of rate adaptation on AP based WLAN network. They proposed a new traffic regulator, TBR(Time-Based Regulator), which runs on the AP and works with MAC protocols. TBR presents a good performance in terms of fairness as well as total throughputs. In addition, the authors insist that TBR can cooperate with the existing MAC protocols.

3 ARF-aware TCP: Resolving Performance Anomaly using transport layer control

The main idea of ARF-aware TCP is that transport protocol, which is responsible for end-to-end rate adaptation, controls the sending rate considering the fair link utilization. That is, ARF function at the MAC layer notifies the higher layer protocol, TCP, to adapt the available bandwidth estimation. If we reduce the congestion window size of a TCP session, then the packets in the network will be decreased, and channel utilization will also be reduced. Namely, TCP is aware of the ARF function at MAC layer, and controls the maximum sending rate to achieve fair network utilization.

If MAC layer adopts QoS mechanisms, higher layer protocols would experience unexpected performance degradation. For example, TXOP option of 802.11e can affect the RTT of the session. TXOP option aggregate multiple packets into one jumbo packet and send it at once. This affects the RTT experienced at transport layer because the transmission delay is affected by the link rate and frame size. Moreover, it affects the RTT variance because the packet error rate becomes large. Note that larger packets are more probable to have more bit errors and requires longer time to recover.

MAC layer QoS mechanisms perform fine-grained access control; however, they do not consider the long term behavior estimation such as available bandwidth estimation. On the other hand, rate adaptation at transport protocol is performed for the larger scale network adaptation. Transport protocol finds a

proper sending rate based on bandwidth estimation, and it controls network congestion using distributed algorithm(AIMD). Because the transport protocol performs an end-to-end adaptation, it finds the bandwidth of the bottleneck link and adjust the sending rate to the link. Because ARF function can change the available bandwidth, the bandwidth estimation at transport layer should be modified. Otherwise, transport protocol would keep increasing the sending rate although the available bandwidth is reduced significantly.

Bandwidth estimation at transport layer is performed based on the bottleneck link. Consequently, the estimation mechanism should be modified properly when ARF is activated because it can change the bottleneck link. The transport protocol should find another bottleneck link with its available bandwidth, which requires an amount of time.

In addition, ARF-aware TCP is more flexible than the MAC layer QoS adaptation mechanisms because MAC layer protocols are difficult to be modified. In many cases, it is implemented in firmware, and software modification is limited. On the other hand, TCP can handle rate adaptation by simply modifying some variables. At the sender side, it works very simply. We adapt the congestion window size as the current link rate changes. For example, if the link rate is changed from 11Mbps to 1Mbps, then the congestion window size variable is scaled down to keep the bandwidth estimation history.

By controlling the congestion window size, we have the identical result with the adaptation at the MAC layer. If the transport layer does not transmit packets, then the link layer protocol should wait. If we increase the cwnd size, the TCP sender would transmit at higher speed, and will utilize more wireless channel. On the other hand, if we decrease the cwnd size, then the sender would decrease the sending rate, and the channel utilization will be decreased, in turn.

We introduce a ratio variable k, which is an indicator of ratio between the current link rate and the maximum link rate.

Detailed algorithm of ARF-aware TCP is as follows:

$$cwnd := CongestionWindow,$$
$$k := [maximumlinkrate/currentlinkrate].$$

When a WLAN link rate changes, arf_cwnd, which is a new congestion window in arf mode is calculated.

$$\text{arf\_cwnd} := \lfloor \tfrac{cwnd}{k} \rfloor, \quad \text{if} \quad \lfloor \tfrac{cwnd}{k} \rfloor \geq 1,$$
$$\text{arf\_cwnd} := 1, \quad \text{otherwise.}$$

ARF-aware TCP enters the arf mode when link rate is changed from the highest speed $11Mbps$. When the ARF at MAC layer changes the link rate from the highest rate, it notifies ARF-aware TCP, and changes the mode from normal node to arf mode. TCP uses arf_cwnd when in the arf mode, and also saves cwnd.

At this time, new congestion window(arf_cwnd) is used to adjust outgoing traffic. When the link rate is recovered to the highest rate, then the saved cwnd

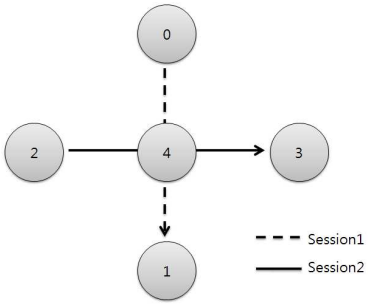

Fig. 1. Simulation Topology

value is used again. When the link rate is recovered to the highest rate, then ARF-aware TCP changes mode from the arf mode to the normal mode.

ARF-aware TCP adjusts the sending rate from arf_cwnd. Our assumption is that the WLAN link became an bottleneck in the path. If the bottleneck was not the WLAN, congestion window does not need to be shrunken as much as the link rate changes. In the case, congestion window is smaller than a proper value. However, the AIMD algorithm finds an available bandwidth with increasing the congestion window size, and the TCP will adapts the sending rate in time.

Our mechanism is easy to be deployed because it does not require global information. That is, every node controls its sending rate in distributed manner. Because each node knows its own link rate, TCP sender can get notification from its own MAC layer. To notify the TCP, a little change at MAC layer is required.

In general, there are two options for implementing the ARF notification. 1) Device driver can directly notify the link rate change. Because ARF is implemented as a device driver optimization option, it does not require firm-ware level modification. On the other hand, standardized DCF algorithm is usually implemented in firmware and hard to be modified. 2) Link rate can be known at TCP indirectly using another protocols such as SNMP(Simple Network Management Protocol). In this case, no modification is required. However, it has an overhead to check the link rate periodically.

In this study, we assume that the link modification can be performed easily and efficiently in terms of modifying the WLAN device driver and having less overhead, respectively.

4 Simulation Result

To validate our study, we performed extensive simulation using NS-2. At first, we present whether ARF-aware TCP can resolve the performance anomaly problem. Figure 1 presents our simulation topology. Node0 and Node1 use FTP applications using TCP. Both nodes begin sessions at 1 second. Initial link rate is 11Mbps. At 20 second, Node0 changes link rate from 11Mbps to 1Mbps. In real network environment, link rate is gradually decreased as the operating mode,

but we changed it for the sake of simplicity. Finally, TCP newReno is used for comparison with Delayed ACK option.

ARF-aware TCP successfully achieves high fairness in performance anomaly scenario. On the other hand, in performance anomaly scenario, TCP new-reno sessions suffer from the performance anomaly. As shown in performance anomaly study[1], both sessions have the identical throughput in the end. We can compare the fairness between the sessions via Jain's fairness index. Jain's fairness index[9] is defined as follow:

$$FairIndex = \frac{(\sum_n^{i=1} x_i)^2}{n \cdot \sum_n^{i=1} {x_i}^2}. \tag{1}$$

We can define x_i as follow:

$$x_i = \frac{R_i}{T_i} \tag{2}$$

, where R_i is a measured throughput of session i, and T_i is an optimal throughput for session i. In addition, we can define optimal throughput T_i as follow:

$$T_i = \sum_{L_r} \frac{L_r \cdot (\text{Time duration of operating rate at} L_r)}{(\text{Total simulation time})} \tag{3}$$

, where operating link rate $L_r \in \{\text{possible operating mode}\}$.

Fairness index ranges from zero to one. If all the nodes perfectly share the network fairly, then the fairness index becomes one. T_i defines the optimal throughput when there is only one session which uses ARF.

Table 1. Comparison of ARF-aware TCP and TCP newReno (simple topology)

	ARF-aware TCP	TCP newReno
Throughput of Session1 (11Mbps→ 1Mbps at 20 sec.)	486 kbps	616 kbps
Throughput of Session2 (11Mbps)	998 kbps	595 kbps
Fairness Index	0.926377	0.746335
Aggregated throughput	1484 kbps	1211 kbps

Because ARF-aware TCP resolves performance anomaly, ARF-aware TCP achieves much higher fairness value as presented in Table 1.

Besides the fairness index, aggregated throughput is also enhanced in ARF-aware TCP. The reason of aggregated throughput enhancement is that the higher rate node can utilize the channel more than the lower rate node.

The session throughput variation in performance anomaly scenario is presented in Figure 2. The graph shows transmitted number of packets per second. In ARF-aware TCP, throughput of Session1 drops at 20 second because the sender adjusts the congestion window as arf_cwnd. In this scenario, ratio k is

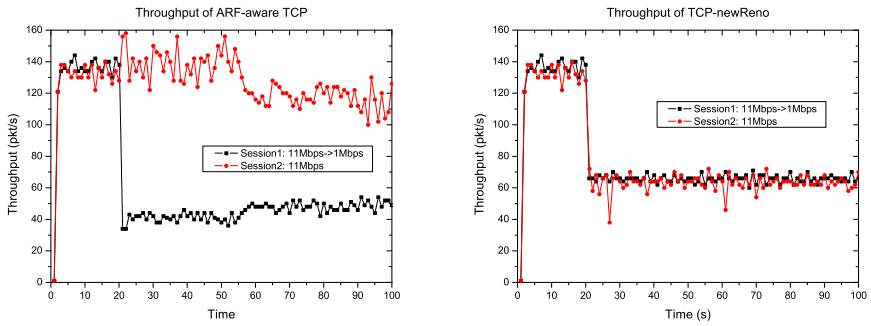

Fig. 2. Throughput(pkt/s) variations of ARF-aware TCP sessions and TCP newReno sessions

11, and the congestion window size is decreased to cwnd11. On the other hand, Session2 keeps sending rate as before, and utilizes the channel as before.

On the contrary to ARF-aware TCP, TCP newReno suffers from performance anomaly as presented in Figure 2. Both sessions utilize the channel time at equal proportion, and Session2 is also affected by the Session1 regardless of its own link rate.

Because performance anomaly is resolved, the aggregated throughput of sessions is increased. We can compare aggregated throughput through the table 1. ARF-aware TCP present slightly higher throughput than the Reno sessions. The performance gain from resolving performance anomaly is about 20% in this scenario.

To present that the original TCP newReno performs fairly, we measured throughput of two TCP newReno sessions without performance anomaly scenario. Namely, we did not change the link rate of Session1 in Fair-share scenario (remains 11Mbps). TCP newReno originally achieves high fairness as shown in Table 2. However, the fairness is broken when performance anomaly happens.

Table 2. Comparison of TCP newReno in the anomaly and fair-share scenario

Performance of TCP newReno	Fair-share scenario	Performance anomaly scenario
Throughput of Session1	1070 kbps	616 kbps
Throughput of Session2	1031 kbps	595 kbps
Fairness Index	0.999656	0.746335
Aggregated throughput	2101 kbps	1211 kbps

Because original TCP newReno keeps fairness, ARF-aware TCP also keeps fairness when both sessions reduce the link rate. We measured throughput result when both sessions reduce the link rate from 11Mbps to 1Mbps at 20 sec. ARF-

Table 3. ARF-aware TCP in low-rate fair-share scenario

Stability of ARF-aware TCP	ARF-aware TCP	TCP newReno
Throughput of Session1 (11Mbps→ 1Mbps at 20 sec.)	407 kbps	411 kbps
Throughput of Session2 (11Mbps→ 1Mbps at 20 sec.)	392 kbps	383 kbps
Fairness Index	0.999648	0.998758
Aggregated throughput	799 kbps	794 kbps

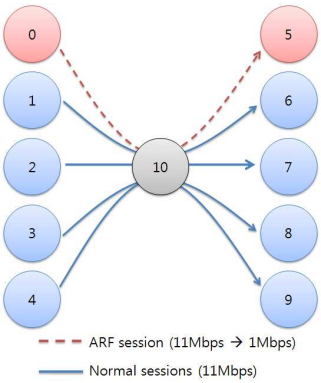

Fig. 3. Simulation topology of multi-session scenario

Table 4. Comparison of ARF-aware TCP and TCP newReno in multi-session scenario

	ARF-aware TCP	TCP newReno
Throughput of Session1 (11Mbps→ 1Mbps at 20 sec.)	127 kbps	341 kbps
Throughput of Session2(11Mbps)	444 kbps	320 kbps
Throughput of Session3(11Mbps)	430 kbps	305 kbps
Throughput of Session4(11Mbps)	485 kbps	298 kbps
Throughput of Session5(11Mbps)	512 kbps	286 kbps
Fairness Index	0.996125	0.626977
Aggregated throughput	1998 kbps	1550 kbps

aware TCP performs stably when both sessions reduce the link rate as presented in Table 3.

To confirm that ARF-aware TCP performs well in multi-session scenario, we increased the total number of sessions. In this case, we modified the network topology as shown in Figure 3. To simplify the simulation, we changed link rate from 11Mbps to 1Mbps directly. Yet, it does not lose generality because the operation mode can be set individually. In our best result, throughput gain is higher than traditional Reno sessions about 30%.

The result is presented in Table 4. ARF-aware TCP achieves much higher fairness index as well as aggregated throughput.

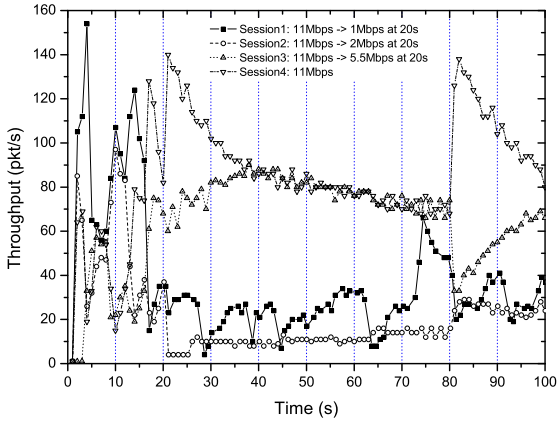

Fig. 4. Throughput of ARF-aware TCP in multi-session multi-rate scenario

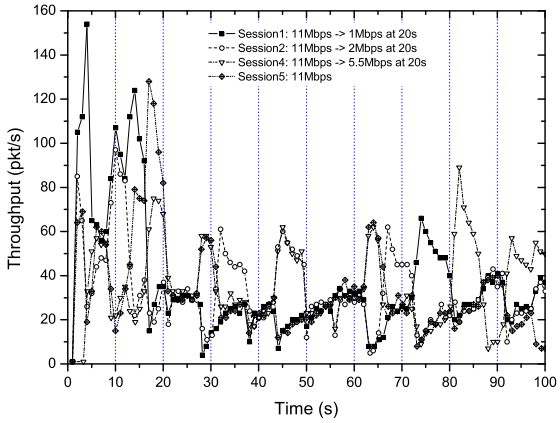

Fig. 5. Throughput of TCP-newReno in multi-session multi-rate scenario

For more realistic case, we performed simulation for diverse rates. All the five nodes changes to different operation rates (1Mbps, 2Mbps, 5.5Mbps, 11Mbps). That is, Node0 changes from 11Mbps to 1Mbps, Node1 from 11Mbps to 2Mbps, Node2 from 11Mbps to 5.5Mbps, so forth. Note that the ratio variable k is rounded. The throughput result confirms our study that ARF-aware TCP achieves

Table 5. Multi-session, multi-rate scenario

	ARF-aware TCP	TCP newReno
Throughput of Session1 (11Mbps→ 1Mbps at 20 sec.)	127 kbps	290 kbps
Throughput of Session2 (11Mbps→ 2Mbps at 20 sec.)	163 kbps	262 kbps
Throughput of Session3 (11Mbps→ 5.5Mbps at 20 sec.)	277 kbps	256 kbps
Throughput of Session4 (11Mbps)	670 kbps	260 kbps
Fairness Index	0.94693	0.831825
Aggregated throughput	1237 kbps	1068 kbps

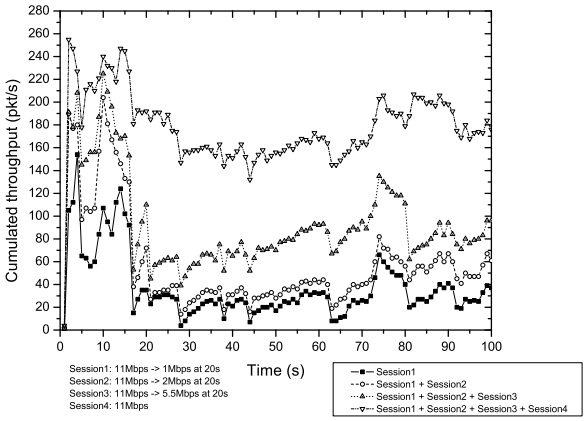

Fig. 6. Aggregated throughput of ARF-aware TCP in multi-session multi-rate scenario

high fairness as well as enhanced aggregated throughput as presented in Table 5.

We can observe the network utilization among the nodes by the throughput variation, presented in Figure 4.

ARF-aware TCP shares network bandwidth as the link rate as presented in Figure 4; however, the TCP newReno shares network bandwidth very inconsistently (Figure 5). Moreover, the cumulated throughput in Figure 6 and presents an interesting result that ARF-aware TCP keeps network utilization even in the performance anomaly scenario. On the other hand, the cumulated throughput drops very seriously in TCP newReno case as the Figure 7.

When the ARF-aware TCP recovers from the arf mode to normal mode, it boosts the sending rate and recovers the original rate very quickly. Consequently, it keeps fairness very stable when compared with the TCP newReno. The fol-

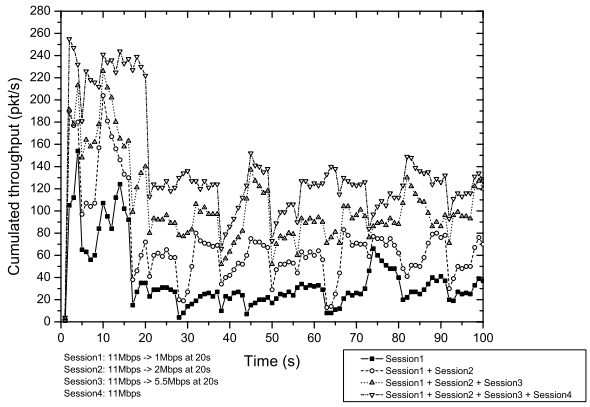

Fig. 7. Aggregated throughput of TCP newReno in multi-session multi-rate scenario

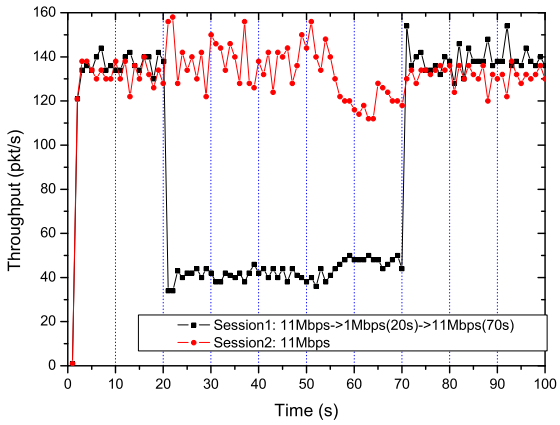

Fig. 8. Throughput(pkt/s) variation of ARF-aware TCP in recovery scenario

lowing simulation is based on the topology presented in Figure 1. Node0 changes link rate from 11Mbps to 1Mbps at 20 second, and recovers back to 11Mbps at 70 second.

We can observe the quick recovery from the performance anomaly through the throughput graph shown in Figure 8. Session2 keeps high sending rate in the anomaly period, and Session1 successfully recovers from the performance anomaly and shares the link with Session2.

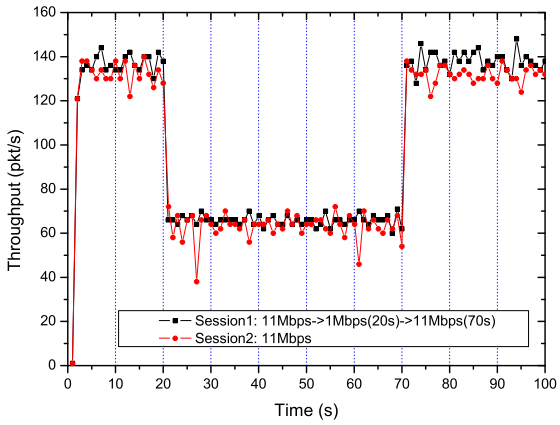

Fig. 9. Throughput(pkt/s) variation of TCP newReno in recovery scenario

TCP newReno presents performance anomaly during the anomaly period. However, it also successfully recovers from the anomaly as shown in Figure 9.

Proposing ARF-aware TCP resolves performance and achieves much higher fairness among sessions. It also naturally enhances aggregated throughput of sessions. In addition, it works stable because it keeps fairness in the case that both nodes are in arf mode. Finally, it presents much higher fairness and throughput in more practical scenarios that multiple nodes are operating at different rates at the same time.

5 Conclusion

We present ARF-aware TCP that resolves performance anomaly in the transport layer, which can largely enhance the fairness in multi-rate WLAN networks. ARF-aware TCP resolves performance anomaly by adjusting the congestion window size. It can be easily deployed because MAC layer modification is not required. ARF-aware TCP enhances not only the fairness but also the aggregated throughput. In our simulation, ARF-aware TCP presents a good performance in the performance anomaly scenarios. Fairness index among ARF-aware TCP sessions always achieves higher than 0.9, while TCP newReno achieves 0.74 or 0.62 in cases. ARF-aware TCP achieves high fairness in multi-session, and multi-session multi-rate scenario, also. In addition to fairness, aggregated throughput is enhanced by 30% in our best case.

References

1. Heusse, M., et al. *Performance anomaly of 802.11b. in INFOCOM 2003*, In proceedings of the twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies, IEEE, 2003.
2. Lacage, M., Manshaei, M.H. and Turletti, T., *IEEE 802.11 Rate Adaptation: A Practical Approach*, In proceedings of the 7th ACM international Symposium on Modeling, Analysis and Simulation of wireless and mobile systems, 2004, 126-134.
3. Hyogon Kim, Sangki Yun, Inhye Kang and Saewoong Bahk, *Resolving 802.11 performance anomalies through QoS differentiation*, Communications Letters, IEEE, 2005, 9(7), 655-657.
4. See-hwan Yoo, Jin-Hee Choi, Jae-Hyun Hwang and Chuck Yoo, *Eliminating the Performance Anomaly of 802.11b*, Lecture Notes in Computer Science, 2005, 3421/2005, 1055-1062.
5. Sadeghi, B., Kanodia, V., Sabharwal, A. and Knightly, E., *OAR: An Opportunistic Auto-Rate Media Access Protocol for Ad-Hoc Networks*, Wireless Networks, 2005, 11(1), 39-53.
6. Godfrey Tan and John Guttag, *The 802.11 MAC protocol leads to inefficient equilibria*, In proceedings of the INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies, 2005, vol.1, 1-11.
7. Godfrey Tan and John Guttag, *Time-based fairness improves performance in multi-rate WLANs*, In proceedings of the USENIX Annual Technical Conference 2004 on USENIX Annual Technical Conference, 2004, 269-282.
8. IEEE 802.11e-2005, in Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements, IEEE, 2005.
9. Jain, R., Chiu, D.M., and Hawe, W., *A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Systems*, DEC Research Report TR-301, 1984.