A Trace-Based Approach for IEEE 802.11 Rate Adaptation in Vehicular Ad-Hoc Networks

Juan M. Navarro
Graduate Student
Network Research Laboratory
Department of Computer Science
University of California at Los Angeles
(773)339-4160
jmnavarr@cs.ucla.edu

ABSTRACT

The fundamental challenge behind rate adaptation is that IEEE 802.11 wireless networks present a strong variety of channel conditions, including random channel errors and mobility-induced channel attenuation. Rate adaptation in Vehicular Ad-hoc Networks (VANETs) faces the following key challenges: (1) due to channel fading and mobility at vehicular speeds, the transmission rate must adapt quickly in order to be effective, and (2) the transmission rate must be able to estimate the link quality with few or no packets transmitted due to infrequent or bursty transmission. The goal of rate adaptation is to estimate the channel quality of a wireless network and adjust the transmission rate accordingly. While critical to system performance, rate adaptation is unspecified in the IEEE 802.11 family of standards. As such, a variety of rate adaptation algorithms for both indoor and mobile scenarios have been proposed. In this work, we present a testbed to objectively compare different rate adaptation algorithms for VANETs. Experimental data is gathered in a real vehicular testbed for three different enviornments, and entered into the Qualnet Network Simulator for performance evaluation. Our results illustrate tradeoffs between different rate adaptation strategies.

General Terms

Algorithms, Measurement, Performance, Design, Reliability

Keywords

Rate Adaptation, VANET, IEEE 802.11

1. INTRODUCTION

The IEEE 802.11 family of standards allows the use of several different rates for data transmission, which enable the use of rate adaptation techniques. Several rate adaptation schemes have been proposed and implemented for 802.11 networks for VANET scenarios. However, a fair and thorough performance evaluation of the available algorithms for VANET applications has been absent. Rapidly fluctuating channel conditions has made it virtually impossible to present different rate adaptation algorithms with the same experimental data in order to perform rate selection. In this work, we obtain the Signal-to-Noise-Ratio along with GPS coordinates for an experimental vehicular scenario, and enter such information into the Qualnet Network Simulator, where we implement various rate adaptation algorithms. This approach allows us to decide whether a data frame will be received by a node on the network, and moved from the physical layer to the MAC layer, where rate selection can then be performed by the rate adaptation algorithm.

2. RELATED WORK

Several rate adaptation algorithms have been developed for IEEE 802.11, including AARF [2], ARF [9], CARA [10], MiSer [11], RBAR [12], and BARA [13]. This section presents an overview of rate adaptation algorithms that have been developed for IEEE 802.11.

2.1 AMRR

The Adaptive Multi Rate Retry (AMRR) Algorithm [2] takes advantage of hardware support for different transmission rates. This feature allows a packet to be sent at successively lower transmission rates in the event of transmission failure. Thus, after transmission fails for a number of times at a given selected rate, the hardware automatically tries to transmit at a successively lower rate. This allows AMRR to cope with short-term variance in link quality, without altering the overall transmission rate as frequently. AMRR uses binary exponential backoff to adapt the interval between rate changes, and uses a probe packet mechanism similar to SampleRate to select overall transmission rates.

2.2 CARS

CARS [4] is a rate adaptation algorithm tailored to vehicular applications. In addition to transmission statistics, CARS uses contextual information to change transmission rates. This information consists of data about the environment which can affect packet delivery probability, such as position speed, and acceleration of the vehicle, distance to neighboring vehicles, time of day, traffic density, and so on. The algorithm estimates packet error as a weighted decision function between context information and Exponential Weighted Moving Average (EWMA) of past transmission statistics. CARS uses a learning model to determine the impact of context information to packet loss probability. Evaluation in a vehicular network scenario shows that CARS is able to quickly adapt to dynamic environments, and outperform other rate adaptation algorithms in this context [4].

2.3 MINSTREL

Minstrel [6] is a rate control algorithm that relies only on measured performance as its criteria for changing rates and adapt to changes in the environment. It takes a probabilistic approach to rate control, and tries to select the rate most likely to result in highest throughput, based on past transmission statistics. For each of the possible transmission rates, Minstrel keeps a record of successful and unsuccessful transmissions. It then defines the probability of successful transmission at any given rate as the ratio between the number of successful transmissions and the total number of transmissions attempted at that rate. When selecting a transmission rate, Minstrel selects the rate in which its measure of throughput is maximized, where throughput is defined as:

$$throughput_{rate} = \frac{P(success)_{rate} \times transmitted\ \ bytes_{rate}}{time\ \ to\ \ transmit\ \ one\ \ packet_{rate}}$$

The throughput maximization procedure is executed at a fixed rate (10 times per second). The algorithm keeps track of which rate has the best expected throughput, the second best throughput, and the highest probability of success. In order to place more importance on recent transmission statistics, Minstrel uses an Exponential Weighted Moving Average (EWMA) to calculate probabilities and rates. Thus, Minstrel can adapt quickly to environment changes, while being robust to spurious interference. Minstrel is implemented and distributed with the MadWifi package, but has not been extensively evaluated by its authors.

2.4 ONOE

Onoe [7] introduces a concept of “credit score” for different transmission rates. Onoe starts by setting an intermediate rate (e.g. 24 Mbps in 802.11g), and a credit score of zero. At every estimation window (1 second), the algorithm evaluates the success/failure rates for the last transmitted packets. If the failure rate was less than a given threshold (e.g. 10%), Onoe increases the credit for that rate. Likewise, if the failure rate was more than the threshold, credit is decreased. Whenever the credit reaches the value of 10, Onoe increases the rate. Likewise, whenever credit reaches 0, the algorithm decreases the rate. Whenever rates change, Onoe resets the credit value to 0. This mechanism makes Onoe change rates conservatively, favoring lower transmission rates. This also makes Onoe robust to individual packet failures. Onoe is implemented and distributed with the MadWifi package, and has been evaluated in [8].

2.5 SAMPLE

The Sample Rate [1] algorithm analyses past transmission statistics: it records successive failures in packet transmissions and total transmission times for each peer. Rates are decreased whenever a threshold of successive transmission failures is reached. Probe packets are used to “test” the link: the protocol periodically sends packets at a rate higher than the current rate. If the transmission is successful, Sample Rate considers that the link conditions are favorable, and increases the rate accordingly. When increasing the rate, the algorithm selects the highest rate which has not failed more than a given number of transmissions. If no such rate exists, the algorithm selects the rate which has the lowest average transmission time.

3. EXPERIMENTAL METHODOLOGY

Our experiments consisted of recording the fluctuating channel conditions along with GPS coordinates. We used three cars, A, B and C. Car A was set up to broadcast packets at 6Mbps at a rate of 100 packets per second. Cars B and C were set up with a capturing program that recorded the SNR value of the incoming data packets, along with the GPS timestamp and coordinates. The times recorded by Cars A, B, and C were then synchronized to correct the offsets in time from the local clocks. For brevity, the results presented in this paper only discusses results from Car B. We expect similar results can be obtained with the data collected in Car C.

The recorded SNR values were then fed into the Qualnet v4. 5 network simulator, and the rate adaptation algorithms were tested accordingly.

3.1 Experimental Setup

The computers onboard the cars were set up with PCMCIA cards with the Atheros Chipset. To further increase the range and power of the signals being transmitted, we used an external 7dBi antenna on each car.

Figure 1 – GPS *Device and Antenna Mount*

3.2 Traffic Routes

Three main traffic routes were tested for our experiments: an urban city environment, a residential district, along with a highway area. These three different scenarios were selected since we envision VANETs to be primarily used in these environments. In order to capture channel conditions in a more controlled environment, we also defined a static traffic route, in which the transmitter was left stationary, while the receiving vehicles traveled to and fro the transmitter.

3.2.1 Static Traffic Route Map

The static route we have defined is an open stretch of road approximately 600 meters in length. It is located in a very un-populated area, in between a cemetery and the I-405 Freeway as shown below:

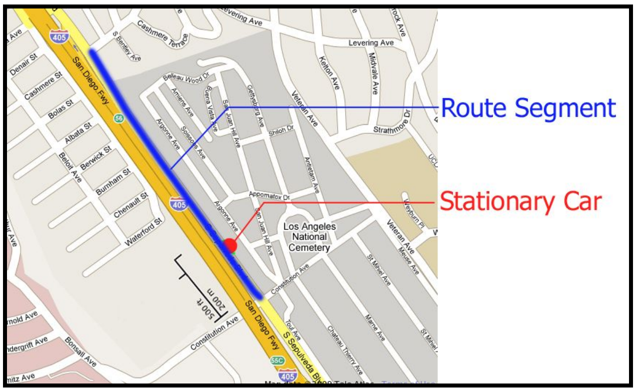

***Figure 2** – Static Traffic Route*

During this test, Car A was left stationary as indicated in Figure 2 above. Cars B and C were driven along the blue route segment (to and fro) as indicated in Figure 2. The speeds of Cars B and C averaged 30 MPH, and ranged from 0 MPH to 40 MPH.

3.2.2 Ongoing Traffic Route Map

The ongoing traffic routes we have selected depict three common scenarios encountered by the modern commuter: an urban, a residential, and a highway environment. Our experimental tests hence focus on obtaining the channel conditions from such areas. In our tests, these areas are defined as follows:

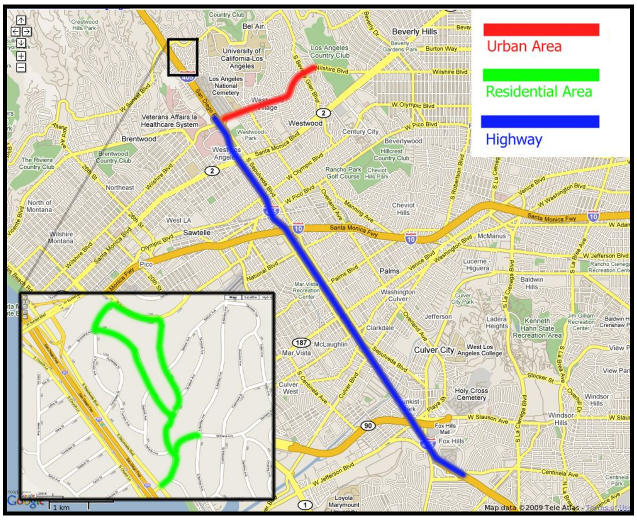

***Figure 3** – Urban, Residential, and Highway Areas*

During the highway test, Cars A, B, and C averaged speeds of approximately 55 MPH, and ranged from 0 MPH (i.e. brief instants) to 65 MPH.

3.3 Qualnet Simulation Environment

The source code for the Qualnet Network Simulator was customized to fit our needs. After gathering the SNR and timestamp information from the experimental traces, these results were then fed into the Qualnet simulator. Several measures were taken to ensure a fair testing environment was present for all rate adaptation algorithms. First, experimental time was synchronized with the simulation time as much as possible, down to a millisecond resolution. This was achieved by synchronizing the GPS time with the local clock, and obtaining the local time in microsecond granularity with the gettimeofday system call. Second, our obtained SNR value from the experimental traces was applied only to transmitted data packets in the simulation. All other packets and management frames were given Qualnet's default SNR value: this ensured that routes were established between the nodes in the simulator prior to the transmission of any data frames. In all of our tests, Qualnet's default SNR value allowed successful transmission of management frames, since the nodes were placed only 10 meters apart.

4. QUALNET IMPLEMENTATION

A total of four base algorithms were tested in our simulation environment, along with two variants of the Robust Rate Adaptation Algorithm (RRAA) [5]. The algorithms we tested include AMRR [2], Auto-Rate Fallback (ARF) [2], the Rate Adaptation for Mobile (RAM) algorithm [16], RRAA, along with two RRAA variants.

4.1 AMRR

The AMRR algorithm associates a frame retransmission count c_0, c_1, c_2, c_3 for each rate used. Typically r_3 is chosen to be the minimum rate available (i.e. 6Mbps for IEEE 802.11a). The value of r_0 is typically set to the current transmission rate. The intermediate values r_1 and r_2 are chosen to be two consecutive lower rates. To ensure short-term variations in our VANET environment are quickly acted upon, we chose $c_0=1$, $c_1=1$, $c_2=1$, and $c_3=1$.

4.2 ARF

Auto-Rate Fallback (ARF) was the first rate adaptation to be published. In ARF, each sender attempts to increase the transmission rate after a fixed number of successful transmissions. In our implementation, this number is set to 10. ARF decreases the transmission rate after 2 consecutive failures. When the rate is increased, the first transmission must succeed, otherwise the rate is immediately decreased.

4.3 RAM

The Rate Adaptation for Mobile (RAM) algorithm presents an SNR receiver-based approach for rate adaptation: the receiver provides feedback information to the sender in order to control the transmission rate. To select the proper rate for the next frame transmission, RAM uses a throughput vs. rate table. For each rate and SNR pair in the table, a value G(R, S) is used to calculate the expected throughput at rate R and SNR value S at the receiver. The value for expected throughput is calculated as:

$$G(R,S) = \frac{L(R,S)}{T(R,S)},$$

where L(R, S) and T(R, S) are the total amount of data received successfully at rate R and SNR value S and the total amount of transmission times for such frames, respectively. The table is updated after successfully receiving a data frame.

The RAM algorithm uses SNR estimation to predict the SNR value of the upcoming frame at the receiver. This strategy has been implemented in RAM to deal with high SNR fluctuation.

The estimation strategy employs a moving average that is calculated as follows:

$$\begin{cases} S_{avg} = (1-\delta)\cdot S_{avg} + \delta \cdot S_{curr}, \\ DEV_{avg} = (1-\rho)\cdot DEV_{avg} + \rho \cdot |S_{curr} - S_{avg}|, \end{cases}$$

$$S_{est} = S_{avg} - \eta \cdot DEV_{avg},$$

where δ, ρ, and η are design parameters. In our implementation, we set δ = ρ = 0.1 and η = 1, as suggested in [16]. After predicting the SNR value S_{est}, the receiver looks up the table and selects the rate for the next transmission as follows:

$$R^* = \arg\max_R G(R, S_{est}).$$

4.4 RRAA

The design of the Robust Rate Adaptation Algorithm (RRAA) for IEEE 802.11 based networks attempts to maximize the aggregate throughput of a network in the presence of varying channel conditions. Specifically, the RRAA algorithm seeks to achieve these goals:

(1) ***Robustness Against Random Loss:*** this goal dictates that the algorithm should maintain stability in rate behavior and throughput performance when presented with mild channel variations.

(2) ***Responsive to Drastic Channel Changes:*** this goal dictates that the algorithm should respond quickly to significant channel changes. Specifically, the algorithm should be highly responsive in the following two scenarios:

- ***a.*** The algorithm is able to track quickly the rate decrease/increase associated with a channel change.
- ***b.*** The algorithm is able to respond quickly in the presence of hidden terminals, and/or interference from other devices such as microwave ovens, cordless phones, etc.

The RRAA algorithm uses a loss ratio that is calculated based on the number of lost frames, and is calculated as follows:

$$P = \frac{\#\_lost\_frames}{\#\_transmitted\_frames}$$

The number of both lost and transmitted frames is counted over an estimation window that is dependent on the current transmission rate. After the transmission of every frame, a counter in our implementation of RRAA keeps track of the number of lost frames, and the number of transmitted frames. The loss ratio is then calculated accordingly.

4.4.1 RRAA-BASIC

The RRAA-BASIC algorithm uses the loss estimation and rate change modules to adjust the transmission rate. Each rate has three parameters associated with it: an estimation window size, a Maximum Tolerable Loss (P_{MTL}) Threshold, and an Opportunistic Rate Increase (P_{ORI}) Threshold.

When first initialized, RRAA-BASIC selects the highest transmission rate and later adapts the rate as necessary. As depicted in (Figure 4), the estimation window associated with the highest rate is selected during initialization.

Upon receiving a frame, the loss ratio is updated, and the counter that keeps track of the number of frames sent using the current estimation window is decremented. If the counter equals zero, then the maximum amount of frames for the current estimation window have been transmitted, and the algorithm performs the rate selection accordingly.

```
R=highest_rate;
counter=ewnd(R);
while true do
   rcv_tx_status(last_frame);
   P = update_loss_ratio();
   if( counter == 0 )
      if     (P > PMTL) then R = next_lower_rate();
      elseif (P < PORI) then R = next_high_rate();
      counter = ewnd(R);
   send(next_frame,R);
   counter--;
```

Figure 4 *– Implementation of RRAA-BASIC*

If the current loss ratio for the estimation window is greater than the P_{MTL} value of the current transmission rate, then the frame loss for the current rate is too high, and the algorithm switches to use the next lower rate. If the current loss ratio is less than the P_{ORI} value of the current transmission rate, then the algorithm switches to use the next higher rate in order to take advantage of the available channel capacity.

4.4.2 RRAA-DYN

An optimization technique for RRAA can be made by allowing the algorithm to change the transmission rate in the middle of an estimation window. Such a technique is especially suited to improve responsiveness to rapid channel fluctuations, e.g. due to mobility. After transmission of each frame, the best (worst) possible loss ratio is calculated, assuming all of the remaining frames in the window all succeed (fail). If the best possible loss ratio already exceeds the P_{MTL} value for that rate, then the rate is immediately decreased and a new estimation window starts. Similarly, the rate is immediately increased when the worst possible loss ratio is smaller than P_{ORI}.

4.4.3 RRAA-HIST

RRAA-HIST is a modified version of the RRAA-BASIC algorithm, were the packet loss ratio is a historical factor. That is, it is not reset to zero at the end of the estimation window.

5. RESULTS

Our simulation results show a very strong correlation between SNR and distance, as expected. Figure 5 below shows this relation:

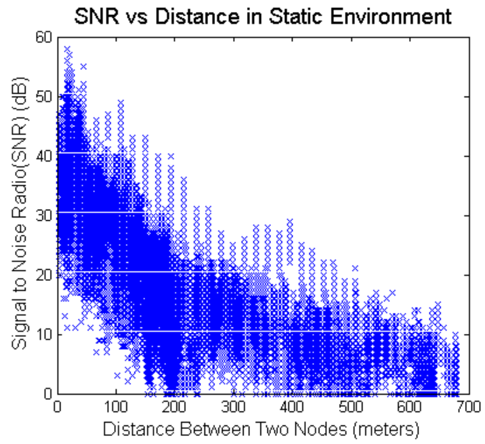

***Figure 5** – SNR vs. Distance*

Our experimental test lasted a total of approximately 3500 s. As shown by the denser areas in Figure 5, the majority of this time was spent in regions where the distance between cars was less than approximately 200 m. During the highway scenario, the distance between our cars reached the neighborhood of 700 m. Our cars generally stayed very close to one another during the city and residential area tests.

5.1 Static Case

The distance between Cars A, B and C ranged from approximately 20 m to 700 m during the static test case. During this test, Cars B and C were driven back and forth along the blue route segment shown in Figure 2. Figure 6 below shows the Throughput vs. Time graph for the data we obtained for Car B. We can see that at time 463 s and time 515 s, cars A and B were closest to one another, and they were farthest from one another at times 440 s, 485 s, and 540 s.

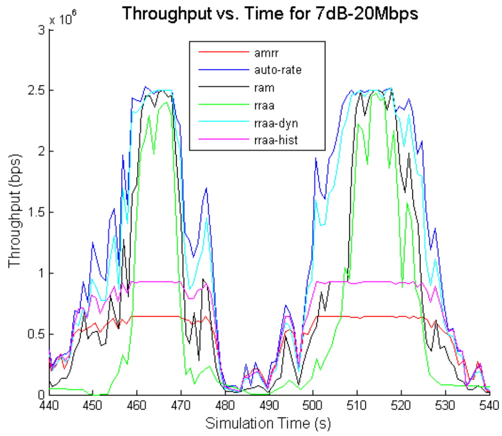

***Figure 6** – Throughput vs. Time for Static Case*

Figure 6 also shows that RRAA generally stays below the evidenced capacity of the link, as shown by Auto-Rate and RRAA-DYN. It is also interesting to note that Auto-Rate and RRAA-DYN generally out-perform all other algorithms. This is in-line with our expectations, since these two algorithms were designed to handle rapidly fluctuating channel conditions. On the other hand, AMRR and RRAA-HIST generally stay well below the envelope defined by the other algorithms. This is also in-line with our expectations, since RRAA-HIST uses a historical packet loss ratio to determine the next transmission rate to use. Using historical information with a rapidly fluctuation channel is less than ideal. We also expected AMRR to not perform as well as other algorithms, since it expects more than 90% of the packets to be transmitted successfully during the previous period in order to increase the transmission rate for the next outgoing frame.

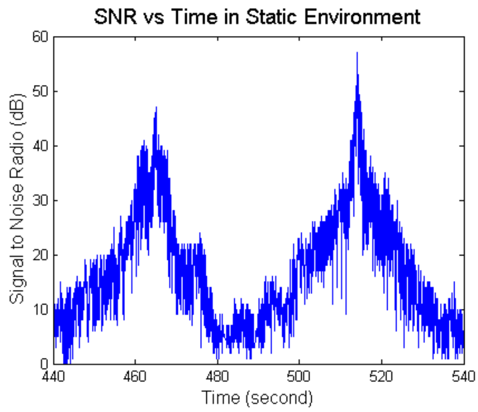

***Figure 7** – SNR vs. Time for Static Case*

Figure 7 captures the channel conditions as a function of time for the static test case. We can see a strong direct correlation between the SNR and the Throughput vs. Time plot shown in Figure 6.

5.2 TCP Tests

Our tests with TCP show that the RAM algorithm constantly out-performs the other algorithms. This can be expected, since RAM does a greater number of retransmissions (i.e. 10 more) at the MAC layer than any other algorithm. Increasing the number of retransmissions at the MAC layer prevents TCP from increasing its backoff timer, thereby increasing its throughput.

It is also interesting to note the performance of RRAA-DYN. Although it is slightly lower than the instantaneous throughput for RAM, it still consistently out-performs other algorithms. This is an indicator that RRAA-DYNs ability to change its transmission rate in the middle of an estimation window is indeed a good optimization technique when facing rapidly changing channel conditions.

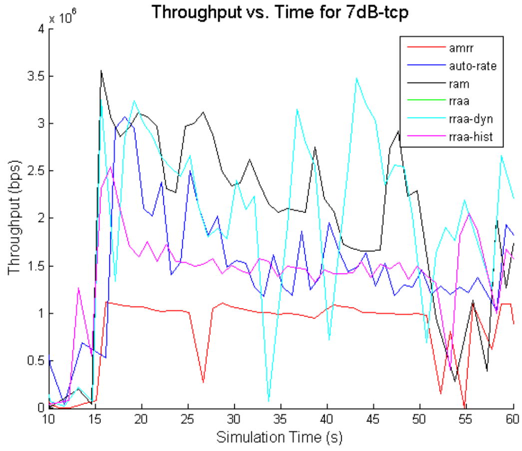

***Figure 8** – TCP vs. Time*

The spikes in RRAA-DYN are in-line with the spikes in the channel conditions, as shown in Figure 9 below:

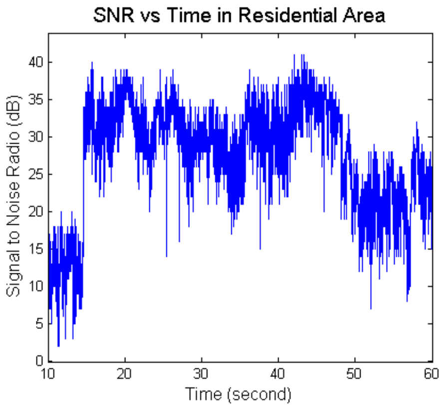

***Figure 9** – SNR vs. Time for TCP Test*

As per our observations in Section 6.1, the AMRR and RRAA-HIST algorithms generally have lower throughput than the other algorithms.

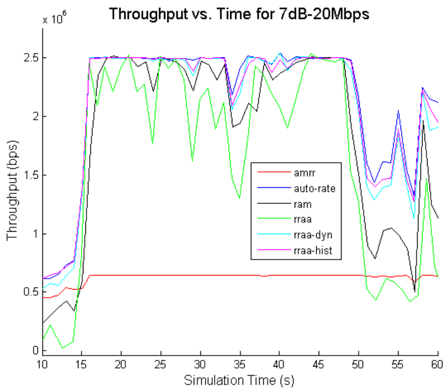

***Figure 10** – UDP vs. Time*

Figure 10 depicts the UDP analog for the same experimental trace as the TCP case in Figure 8. In this case, however, we note that the increased re-transmissions at the MAC layer for RAM slightly hurt its performance, as compared to the other algorithms. We also see that AMRR never attempts to increase its rate, since the channel conditions vary dramatically.

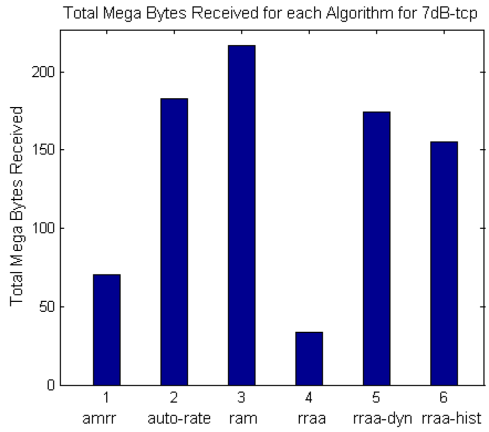

***Figure 11** – Total Mega Bytes Received for TCP*

Figure 11 shows that RAM achieves the highest number of bytes transmitted, followed closely by Auto-Rate and RRAA-DYN. These results show that retransmissions at the MAC layer may be more effective than the dynamic rate adaptation strategies taken by Auto-Rate and RRAA-DYN.

The rate usage histogram for this case is shown in Figure 12. The RRAA algorithm performs strikingly aggressive, transmitting data frames at 54Mbps for a majority of the time. Despite RRAAs aggressive transmission rate, its total bytes received is surprisingly low, when compared to the other algorithms. Our investigations led us to conclude that RRAAs aggressiveness is due to the highly favorable channel conditions available after approximately 15 s, and before approximately 50 s (see Figure 9). During this period, RRAA transmits at the highest rate possible, 54Mbps. However, this induces significant packet loss in TCP. The congestion control mechanism in TCP then exponentially increases its backoff timer, thereby preventing RRAA from successfully transmitting packets in subsequent intervals and therefore reducing the total number of bytes sent (see Figure 11).

Variants of RRAA, such as RRAA-DYN and RRAA-HIST, have been tweaked in our implementation to reduce the transmission rate after two unsuccessful transmissions (i.e. similar to Auto-Rate).

Another disadvantage of using RRAA in this scenario is that it uses the highest transmission rate at the very beginning of the transmission. This creates significant packet loss, and triggers TCPs congestion control mechanism. Our Qualnet simulation output trace shown in Figure 12 below depicts the jump in simulation time due to TCPs exponentially increasing backoff timer:

Sim Time	Exp Time
0.0119610020	0.0130000000
0.0122290030	0.0130000000
0.0126590040	0.0130000000
0.0127920050	0.0130000000
0.0140860060	0.0230000000
0.0160730070	0.0230000000
0.0210480080	0.0230000000
5.6728930240	5.6760000000
5.6732510250	5.6760000000
5.6737800260	5.6760000000
5.6742820270	5.6760000000
5.6744690280	5.6760000000
5.6755110290	5.6760000000
5.6789470300	5.6860000000
29.6934999140	29.7020000000
29.6938669150	29.7020000000
29.6944769160	29.7020000000

***Figure 12** – Sudden Change in Simulation and Experimental Times*

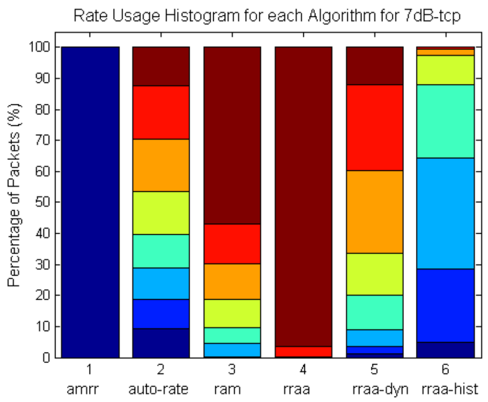

***Figure 12** – Rate Usage Histogram for TCP Test*

In our UDP tests shown in Figure 13, we see that RRAA performs aggressively, but not as aggressive as with the TCP test. This is due to the fact that RRAA is continuously updating (i.e. increasing) its packet loss ratio for rate selection with the packets that would have otherwise not been transmitted by TCP.

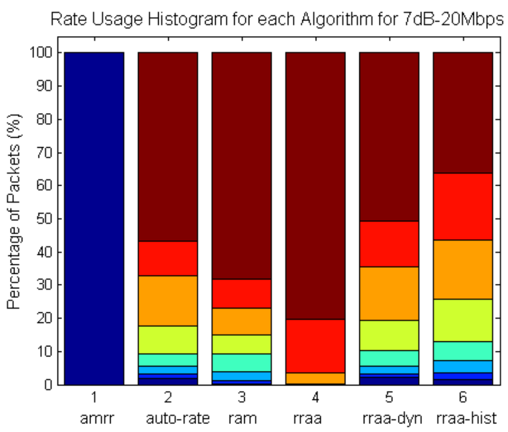

***Figure 13** – Rate Usage Histogram for UDP Test*

A similar test to the TCP test was performed with UDP. In this scenario, we observe that the total bytes received for RRAA closely approaches that of the other algorithms. This leads us to conclude that RRAA is indeed a good algorithm to use, specially for UDP scenarios.

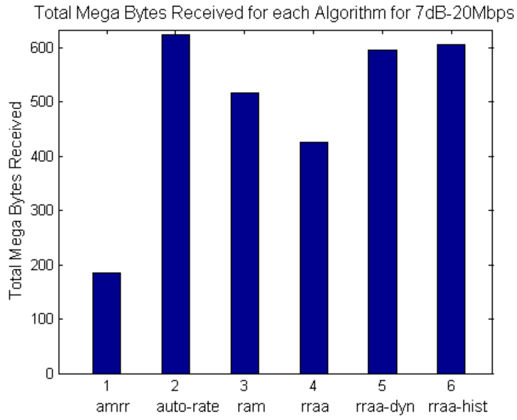

***Figure 14** – Total Mega Bytes Received for UDP*

5.3 Urban Environment

Our results for the urban environment are shown in Figures 13 and 14. From these graphs, we can see that the performance of each algorithm is directly related to the SNR value for the received frame. During our experiment in the urban environment, cars A, B and C generally remained very close to one another and averaged approximately 25 MPH, while ranging from 0 MPH to 35 MPH.

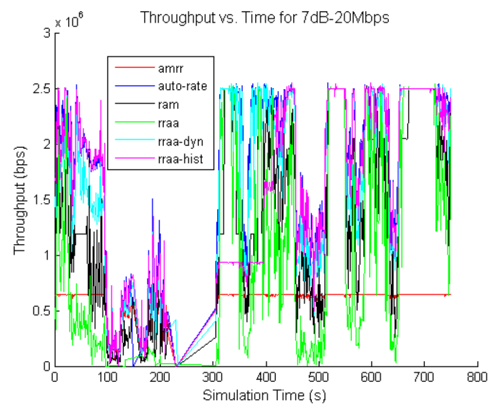

***Figure 13** – Throughput vs. Time for Urban Environment*

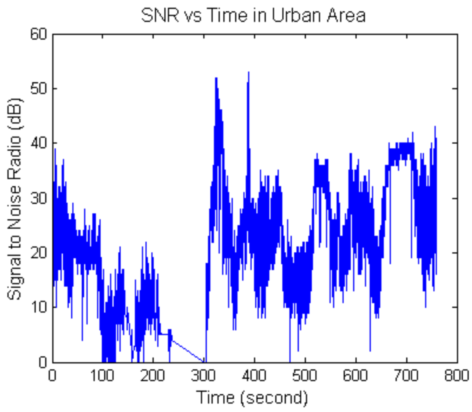

***Figure 14** – SNR vs. Time for Urban Environment*

5.4 Highway Area

Experimental results from the highway scenario are shown in Figures 15 and 16. Compared to our results from the urban environment, the highway scenario results depict increased channel noise, and a greater range in instantaneous throughput, as shown in Figure 15.

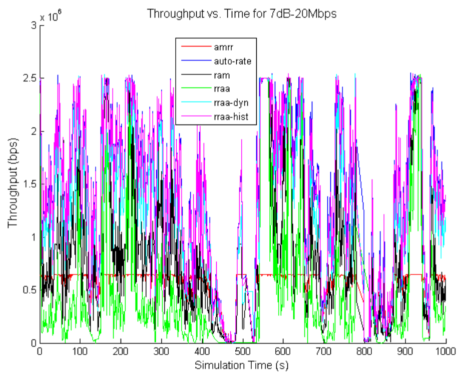

***Figure 15** – Throughput vs. Time for Highway Area*

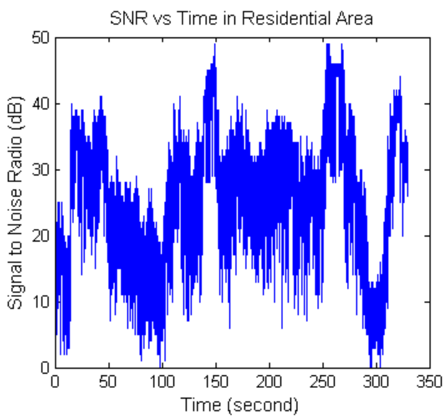

***Figure 16** – SNR vs. Time for Highway Area*

5.5 Residential Area

The noise level for the residential area looks surprisingly low. The instantaneous throughput for all algorithms is generally steady, with few troughs in the throughput plot.

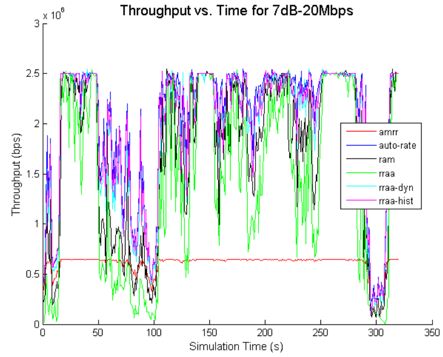

***Figure 17** – Throughput vs. Time for Residential Area*

Figures 17 and 18 also show a strong correlation between channel conditions and instantaneous throughput.

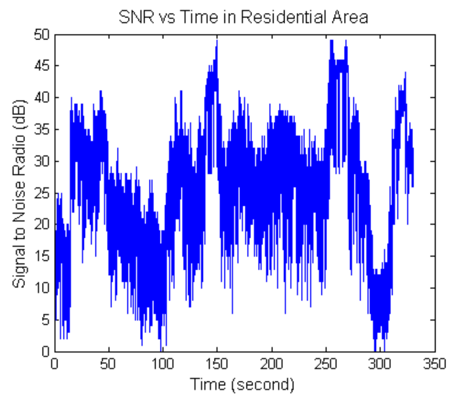

***Figure 18** – SNR vs. Time for Residential Area*

6. CONCLUSION & FUTURE WORK

We have presented a novel way to objectively compare various rate adaptation algorithms in IEEE 802.11 networks, and results from running such experiments.

Our results show that there is a strong correlation between the channel conditions (i.e. SNR value) and the overall throughput that can be achieved on the network. Adaptive schemes, such as those found in Auto-Rate and RRAA-DYN are shown to be the most effective under rapidly varying channel conditions. For our TCP tests, it was shown that RAM out-performed other algorithms, due to the high number of retransmissions it employs.

Algorithms such as RRAA-HIST and AMRR are shown to perform poorly, since they use historical information to perform rate selection. AMRR has very high expectations for rapidly fluctuating channels, expecting less than 10% packet loss in order to perform rate increase.

Disadvantages of RRAAs rate selection scheme were also seen in our TCP tests. Selecting a high initial transmission rate for RRAA is shown to be detrimental to its performance during the first few seconds of our simulation. We observed a conflict that occurs between RRAA and TCP, whereby TCP exponentially increases its backoff timer, while RRAA aggressively maintains a high transmission rate.

Future work in this area includes adding more rate adaptation algorithms for VANET applications to our simulation environment for performance evaluation. Through continuous and exhaustive evaluation of these algorithms, perhaps a hybrid algorithm that combines the strengths of all others will emerge.

7. ACKNOWLEDGMENTS

Many thanks to my mentors, Kevin Lee, Uichin Lee and Professor Mario Gerla from the Network Research Laboratory at UCLA for their help and guidance in researching this topic.

8. REFERENCES

[1] Bicket, J. C., Bit-rate Selection in Wireless Networks. M.S. Thesis, Feb. 2005 /* SampleRate */

[2] Lacage, M., Manshaei, H., and Turletti. IEEE 802.11 Rate Adaptation: A Practical Approach. 2004., http://madwifi-project.org/attachment/wiki/UserDocs/RateControl/jbicket-ms.pdf?format=raw /* AMRR, ARF, ARRF, RBAR, etc. */

[3] MadWifi Project., http://www.madwifi.org

[4] Shankar, P.; Nadeem, T.; Rosca, J.; Iftode, L., "CARS: Context-Aware Rate Selection for vehicular networks," *Network Protocols, 2008. ICNP 2008. IEEE International Conference on* , vol., no., pp.1-12, 19-22 Oct. 2008

[5] Wong, S. H., Yang, H., Lu, S., and Bharghavan, V. 2006. Robust rate adaptation for 802.11 wireless networks. In *Proceedings of the 12th Annual international Conference on Mobile Computing and Networking* (Los Angeles, CA, USA, September 23 - 29, 2006). MobiCom '06

[6] Minstrel Rate Control. MadWifi Project, 2009. http://linuxwireless.org/en/developers/Documentation/mac80211/RateControl/minstrel

[7] Onoe Rate Control. MadWifi Project, 2009.

[8] http://madwifi-project.org/wiki/UserDocs/RateControl

[9] Ancillotti, E., Bruno, R., and Conti, M. 2008. Experimentation and performance evaluation of rate adaptation algorithms in wireless mesh networks. In Proceedings of the 5th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (Vancouver, British Columbia, Canada, October 27 - 28, 2008). PE-WASUN '08. ACM, New York, NY, 7-14. DOI 10.1145/1454609.1454612.

[10] Kamerman, A.; Monteban, L. WaveLAN(C)-II: a high-performance wireless LAN for the unlicensed band. Bell Labs Technical Journal 1997, 2, 118-133.

[11] J.Kim, S.Kim, S.Choi, and D.Qiao, "CARA: Collision-aware rate adaptation for IEEE 802.11 WLANs," in Proc. IEEE INFOCOM, Barcelona, Spain, March 2006.

[12] Qiao, D., Choi, S., Jain, A., and Shin, K. G. 2003. MiSer: an optimal low-energy transmission strategy for IEEE 802.11a/h. In Proceedings of the 9th Annual international Conference on Mobile Computing and Networking (San Diego, CA, USA, September 14 - 19, 2003). MobiCom '03. ACM, New York, NY, 161-175. DOI: 10.1145/938985.939003.

[13] Holland, G., Vaidya, N., and Bahl, P. 2001. A rate-adaptive MAC protocol for multi-Hop wireless networks. In Proceedings of the 7th Annual international Conference on Mobile Computing and Networking (Rome, Italy). MobiCom '01. ACM, New York, NY, 236-251. DOI: 10.1145/381677.381700.

[14] Wu, S., Biaz, S., Qi, B., and Zhang, K. 2007. BARA: a sender based rate adaptation in wireless networks. In Proceedings of the 45th Annual Southeast Regional Conference (Winston-Salem, North Carolina, March 23 - 24, 2007). ACM-SE 45. ACM, New York, NY, 179-184. DOI= http://doi.acm.org/10.1145/1233341.1233374

[15] Iperf, http://www.noc.ucf.edu/Tools/Iperf/

[16] Chen, X., Gangwal, P., Qiao, D. Practical Rate Adaptation in Mobile Environments. In Proceedings of IEEE International Conference on Pervasive Computing, 2009 (Galveston, Texas, March 9-13, 2009). PERCOM 2009. DOI=www.ece.iastate.edu/~daji/papers/percom2009.pdf

[17] Camp, J. Knightly, E. Modulation Rate Adaptation in Ruban and Vehicular Environments: Cross-layer Implementation and Experimental Evaluation. Department of Electrical and Computer Engineering, Rice University. Houston, Texas. 2008. DOI= networks.rice.edu/papers/com0541-camp.pdf