Link Quality Prediction for Wireless Devices with Multiple Radios

Qiuyi Duan, Lei Wang, Charles D. Knutson and Daniel Zappala
Computer Science Department
Brigham Young University
Provo, UT 84602-6576
{qiuyi,lei,knutson,zappala}@cs.byu.edu

Abstract

Communication between wireless devices ought to be as simple as possible; they should be able to seamlessly switch between different radios and network stacks on the fly in order to better serve the user. To make this a possibility, we consider the challenging problem of predicting link quality in a changing mobile environment. In this paper we present an algorithm that uses Weighted Least Squares Regression to predict whether a given link can meet application requirements in terms of throughput, delay, and jitter. We use a simulation study to demonstrate that our algorithm is able to predict link quality accurately and stably in a frequently changing mobile environment. The prediction algorithm is more accurate than several alternative algorithms, and the overhead caused by the link measurements is negligible in terms of throughput and power consumption.

1. Introduction

Wireless devices ought to make it easier for users to communicate with each other. Complicating this vision, however, is the reality that no single wireless technology dominates the market nor provides the desired functionality in all situations. Cellular, WiFi, and Bluetooth interfaces are all used for different situations today. It is likely that wireless technologies will continue to proliferate and that devices will continue to contain multiple radios and network stacks.

To cope with this reality, devices ought to be able to seamlessly switch between available network connections on the fly in order to provide access to available services. For example, if a person wants to transfer images from a cellphone to a laptop, the devices should cooperate to make this happen however they can, regardless of whether this utilizes a Bluetooth or WiFi connection. Furthermore, as the availability or quality of a connection changes due to the activity of other nodes or interference from other devices, devices should cooperate to switch to the best available interface, taking into account power and performance tradeoffs. In other words, wireless devices should exploit their heterogeneity in order to provide better service to end users. This kind of communication should “just work”, rather than requiring the user to be involved.

In this paper, we focus on one aspect of radio selection, the ability to predict the availability and performance characteristics of each available interface. This is a key issue, since in many cases, wireless devices have several different radios to choose from, and need some mechanism for determining which interface is likely to satisfy the application in the near future.

Predicting link availability has received significant attention in ad hoc wireless networks, where it is principally used to help routing protocols provide stable routes [6, 3, 1]. In this work, nodes use a single WiFi radio and cooperate to maintain network connectivity. Each node tries to predict the probability that a link to its neighbor will continue to be available for some time into the future. The routing protocol then uses this metric to compute routes that will remain available for the longest time; this has shown to be more effective than using shortest path routing.

The issues we face in designing for heterogeneous wireless devices differ from this previous work in several fundamental ways. The main difference is that we are interested in predicting link quality, rather than simply link availability. With a single interface and a network of homogeneous devices, maintaining connectivity is most important. However, with multiple available interfaces between two communicating devices, our goal is to choose the interface that can best meet application requirements. Accordingly, we try to predict whether a link will meet the throughput, delay, and jitter requirements of a particular application.

Another difference with previous work is that we are considering devices in which each interface may potentially have its own network stack. For example, WiFi interfaces typically use a TCP/IP stack, but Bluetooth interfaces have their own stack. This means that we must devise a general algorithm that does not depend on a particular technology.

We must also handle interference as a common occurrence, since different technologies may share the same frequencies.

In this paper, we present a link quality prediction algorithm using Weighted Least Square Regression (WLSR). The algorithm allows mobile devices with multiple radios to statistically predict link quality based on a series of past measurements. Because the algorithm is sensitive to regression weights and the measurement window size, we use a simulation to determine good settings for these parameters. We also develop a method to compute an ideal prediction curve so that we can compare prediction accuracy of various algorithms.

Our simulation results demonstrate that the WLSR algorithm is able to predict link quality accurately and stably in a frequently changing mobile environment. The prediction algorithm is more accurate than several alternative algorithms, and the overhead caused by the link measurements is negligible in terms of throughput and power consumption.

We demonstrate the utility of our prediction algorithm by using it in a radio switching architecture that enables multi-radio devices to switch to a different radio when the current one becomes unavailable due to mobility or interference. This combination allows devices to increase throughput or lower power consumption, based on user preference.

2. Link Quality Prediction

Consider two devices – a laptop and a PDA, for example – each with a WiFi and a Bluetooth interface. The key concept we explore in this paper is how to predict the quality of each interface so that the device can dynamically choose the one that is most likely to meet application requirements.

We use three metrics to express application QoS requirements: throughput, delay, and jitter. The application specifies its requirements in terms of a threshold that must be met – if the link can support these requirements, then it is said to be "qualified" for that application. The device then selects the best qualified link based on user preference, which could for example favor performance or power savings.

2.1. Link Quality Measurements

To determine the quality of a link between two devices, we periodically send link-layer assessment queries from one device to another. To make a query, the primary device sends a *link quality request* to the secondary device using the appropriate network stack, requesting its real-time quality information. Upon receiving the request, the secondary device measures it signal-to-noise ratio (SNR) and includes this in a *link quality response* using the same radio.

To compute the throughput of the link, the primary device averages its own SNR with the value in the response. It then uses this average to estimate the throughput of the link using Shannon's capacity formula in a Rayleigh Fading Environment [5]. Using this estimate provides an upper bound on capacity, ensuring that the system will never conclude that a link is unqualified when it is actually suitable. In addition, this method uses little power and overhead as compared to measuring the channel over a sustained period of time.

The primary device uses the round-trip time for the request and response as a measure of the link delay. The jitter is calculated as the difference between the delay for this request and the previous request.

To ensure the accuracy of our measurements, both the request and response are given priority in the device OS, so that the round-trip time gives an accurate measure of link delay, including any MAC negotiation. In addition, we pause active data traffic during the query to avoid any conflicts that might occur if multiple radios in a single device attempted to operate concurrently.

For this paper, we use a fixed query interval of 1 second. As part of our future work we will consider dynamically varying the query interval during stable periods, to reduce overhead and conserve battery life.

2.2. Prediction Algorithm

The wireless device keeps a window of past measurements and then tries to predict future link quality based on these measurements. The window contents are kept in FIFO order, so that a new measurement replaces the oldest measurement.

We develop a prediction algorithm based on the Weighted Least Square Regression (WLSR) algorithm [2]. WLSR is an efficient prediction method that makes good use of small data sets. The only state required is the set of measurements considered (we use 5 to 30 measurements), and the algorithm can be implemented with about a hundred lines of code. No training or learning is required. Because WLSR applies weights to the measurements, we can treat them with different levels of importance according to their ages. This makes WLSR well-suited for a frequently changing mobile environment, since only recent performance measurements are useful in predicting future availability.

The WLSR algorithm takes as input a window of measurements for a given QoS metric and predicts the value of the metric for the next scheduled measurement period (e.g. 1 second in the future). This prediction includes both a mean and a standard deviation.

To predict whether a link can meet an application's QoS requirements, we construct a probability density function

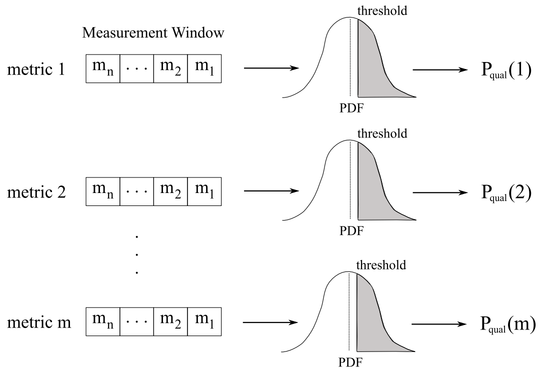

Figure 1. Qualification probability

(pdf) for each metric at the prediction time, using the predicted mean and standard deviation. We use a t distribution for the pdf since the sampled data set is small. The qualification probability, P_{qual}, is the area under the pdf that meets the threshold (to the right for throughput and to the left for delay and jitter). We consider the link to be qualified for this metric if P_{qual} is at least 50% of the total area. This is equivalent to the mean of the pdf meeting the QoS threshold for the metric under consideration. This process is illustrated in Figure 1.

The overall availability of the link, (P_{avail}), is given by averaging the qualification probability of each metric, where m is the number of metrics.

$$P_{avail} = \sum_{i=1}^{m} 1/m * P_{qual}(i), \qquad (1)$$

We consider the link to be acceptable for the application if its predicted performance meets the QoS requirements of all relevant metrics. Accordingly, we rate the link as available if $P_{avail} \geq 50\%$. It is possible that the overall availability is greater than 50%, even when the link is not qualified for one or more of the metrics. In this case, we artificially assign the overall availability to 40%, so that the link is considered unavailable.

3. Results

We perform a simulation study to evaluate the feasibility and accuracy of the prediction algorithm using ns-2.28. We implemented an interface interference model, the link measurement mechanism, and the prediction algorithm.

We use a topology that includes two mobile devices with both WiFi and Bluetooth radios, as well as 10 Bluetooth devices and 20 WiFi devices. The two multi-radio devices communicate using a VoIP application running over UDP, since this application uses all three of the link quality metrics – throughput, delay, and jitter. We note that the choice of application and transport protocol does not affect the prediction accuracy.

The simulation topology is designed so that the link quality of the two radios varies due to both mobility and interference. To simulate mobility we move the multi-radio devices in and out of range of each other. To simulate interference we turn on and off the Bluetooth and WiFi devices.

To evaluate the accuracy of our prediction, we compare the predicted availability curve to an *ideal curve* generated with the benefit of hindsight as we look at all of the measurements, rather than just the recent past. Given an ideal curve, we then compare it to the prediction curve and calculate the *prediction error rate*, which is the ratio of the number of incorrect predictions to the total number of predictions made during the simulation. The prediction is defined as incorrect if the link is estimated to be unavailable while it is actually available based on the ideal curve, and vice versa.

We also calculate the prediction overhead by measuring the *throughput loss* and the *power consumption* due to making queries.

3.1. WLSR Weights and Window Size

The performance of the WLSR algorithm depends on the weighting of each metric and the measurement window size. To determine the proper settings for these parameters, we performed a full factorial experiment. We simulated scenarios where both the Bluetooth and WiFi radios drop out periodically, and where both radios encounter interference from other nodes with a single radio.

For each scenario, we run an experiment with each possible combination of the WLSR weight κ and the measurement window size n. The weight κ varies from 1.0 to 1.6, in increments of 0.1, and the window size n varies from 5 to 30, in increments of 5. We repeat each experiment five times. Because of space constraints, we omit the table of results. With other parameters we tested the prediction error rate either increases or remains the same.

When responsiveness is preferred, a small measurement window and a larger weight work best. This gives the highest weight to the most recent of a small number measurements, so that the prediction is likewise more responsive. As stability is preferred, a larger window and smaller weight begin to perform better.

Based on our results, we believe a measurement window size of 10 and a WLSR weight of 1.3 provide a good balance between reactivity and stability. These settings perform well across all the experiments.

Prediction Algorithm		WLSR	EWMA	Signal
Mobility	WiFi	1.62%	2.71%	1.44%
	BT	6.13%	13.62%	11.09%
Interference	WiFi	12.30%	17.75%	22.45%
	BT	9.18%	9.35%	9.68%
Combined	WiFi	10.90%	15.35%	18.64%
	BT	6.62%	12.52%	12.71%
Average		**7.79%**	**11.88%**	**12.67%**

Table 1. Prediction accuracy comparison

3.2. Comparison to Other Algorithms

We evaluate the WLSR prediction algorithm by comparing it to another two alternative algorithms: Exponential Weighted Moving Average (EWMA) and a prediction based on the signal strength only.

We test three scenarios: mobility, interference, and a combination of both conditions. We run each simulation for 240 seconds, with one link measurement query per second for each radio. We repeat each simulation five times and combine the results.

We show the average prediction error for this experiment in Table 1. In almost all scenarios, the WLSR algorithm is able to predict link quality more accurately than the other two algorithms. The only exception is the prediction for WiFi in the mobility scenario, where the error rate of WLSR is only slightly higher than that of signal strength model. This is a good result for WLSR, since signal strength prediction is mainly useful for mobility prediction, and WLSR does nearly as well. On average, more than 90% of the predictions using WLSR are correct, as compared to the ideal curve.

To further illustrate the prediction accuracy of the WLSR algorithm, we plot one of the simulation results for the mobility scenario in Figure 2, using a randomly selected replication seed. The measurement curve represents the actual link status considering all relevant QoS metrics, either available (shown as 1.0) or unavailable (presented as 0.0). The ideal curve is generated from the measurement and based on user's preference. We then show the predictions for all three algorithms. On each prediction we plot the threshold at 0.5; if a prediction is above the threshold then the link is predicted to be available during that period.

This figure shows why the WLSR prediction is more accurate than the EWMA and signal strength algorithms. It is able to closely match the ideal curve, with very little latency when changes occur. In general, the EWMA algorithm reacts more gradually to changes in link status, taking an extra second or two to react. Using only signal strength is so sensitive to rapid changes in Bluetooth mobility that it does not provide the stability we prefer during these periods.

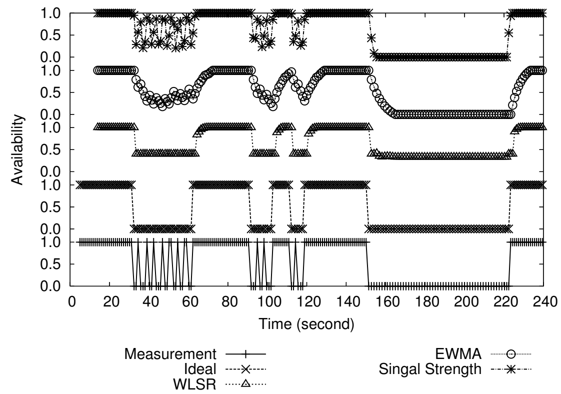

Figure 2. Bluetooth link quality prediction during mobility

Transport	UDP			TCP
Workload	**Light**	**Medium**	**Heavy**	–
Throughput	-0.05%	-0.12%	-0.40%	-0.19%
Power (queries)	0.0034%	0.0034%	0.0032%	0.0030%
Power (workload)	0.0066%	0.0144%	0.0592%	0.0660%

Table 2. Link measurement overhead

3.3. Overhead

Periodic link quality measurements impose overhead in terms of throughput and power consumption. To evaluate overhead we generate UDP traffic under three workloads – light (10 packets/s), medium (100 packets/s) and heavy (1000 packets/s), with 128 byte packets. For a fourth workload we generate a constant stream of TCP traffic. In all cases we measure throughput both with and without queries and then calculate throughput loss as a ratio. We calculate the power consumed as a percentage of the overall battery life per minute, and we do this separately for the queries and the background workload. The initial battery level is 10 Watt-hours, which is the typical battery level for PDAs. Table 2 shows the average overhead as computed from five replications of the experiments.

These results show that the overhead for link quality measurements is very low, with almost negligible throughput loss and power consumption. In all cases, the throughput loss is below 0.5%, and the power consumed is generally small compared to the workload. The throughput loss during light UDP loads is smaller because the link is usually not busy when queries are sent.

4 Dynamic Radio Selection

To illustrate the utility of our prediction algorithm, we use the results of the prediction to dynamically select the

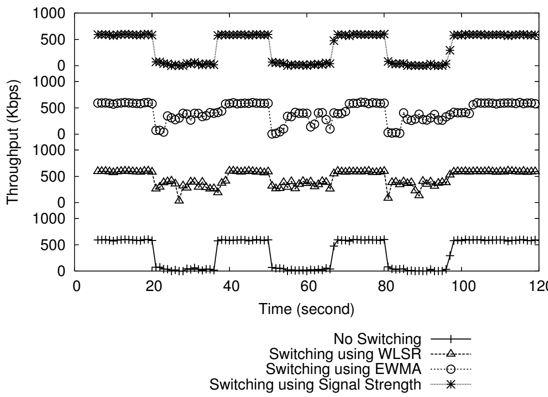

Figure 3. Dynamic radio selection

best available radio. For this simulation we use the Quality of Transport (QoT) architecture [4], which acts as a shim between the application/session layer and the transport layer. QoT always chooses the best available radio, based on the results of link prediction algorithm and user preference. For example, if two radios provide a connection between the devices, then the user may prefer to favor power conservation or throughput.

The first scenario we consider is when the user prefers high throughput. While the multi-radio devices communicate, 20 pairs of WiFi nodes transmit nearby, simulating intensive interference for a 15 second interval, at 20s, 50s, and 80s. The simulation runs for 120 seconds. The multi-radio devices should be able to switch to Bluetooth when the WiFi radio becomes unavailable.

Figure 3 compares the performance of radio selection when using either the WLSR, EWMA, or signal strength prediction algorithms. We include the case for no radio switching (staying with WiFi the entire time) to illustrate the overall benefit of using both radios. Both the WLSR and EWMA algorithms improve performance significantly, whereas switching based on signal strength has almost no effect. Because the measured signal strength still satisfies the transceiver capture level, it does not handle interference well.

When comparing WLSR to EWMA, WLSR has shorter latency in the prediction algorithm, allowing radio selection to happen faster. Thus, although both algorithms result in throughput dropping for a short period, using WLSR has a tangible benefit. The overall benefit is reflected in Table 3, which summaries the throughput gain seen by each mechanism. Using WLSR improves throughput by 32% in this case, as compared to EWMA, which gains 22%.

The second scenario we consider is when the user prefers power savings. While the multi-radio devices communicate, they move out of range of the Bluetooth radio for a period of 30 seconds. This happens twice, once at 15s and

Prediction Method	**None**	**WLSR**	**EWMA**	**Signal**
Total MBytes	5.33	7.07	6.52	5.33
Throughput (kbps)	358.88	475.49	438.05	358.88
% Gain	-	32.49%	22.06%	0.00%

Table 3. Throughput improvement

Prediction Method	**None**	**WLSR**	**EWMA**	**Signal**
Power (Joules/s)	21.28	12.45	12.88	12.08
% Savings	-	-41.50%	-39.47%	-43.23%

Table 4. Power savings

again at 75s, with the simulation running for 120 seconds. We again compare the three prediction algorithms.

As shown in Table 4, all three algorithms provide significant power savings when compared to using the WiFi radio the entire time. In this case, using signal strength alone works very well, but the WLSR algorithm is almost as good. Since WLSR also handles interference well, these results show it is a good fit for a radio selection architecture.

5. Future Work

Our future work will focus on dynamically adjusting the link query interval, so that we can further lower overhead and power consumption during stable periods. We are also continuing to work on efficient radio switching algorithms.

References

[1] M. Gerharz, C. de Waal, M. Frank, and P. Martini. Link stability in mobile wireless ad hoc networks. In *Proceedings of the 27th Annual IEEE Conference on Local Computer Networks (LCN)*. IEEE, 2002.

[2] R. Jain. *The Art of Computer Systems Performance Analysis : Techniques for Experimental Design, Measurement, Simulation, and Modeling*. John Wiley and Sons, Inc., April, 1991.

[3] S. Jiang, D. He, and J. Rao. A prediction-based link availability estimation for mobile ad hoc networks. In *IEEE INFOCOM*, 2001.

[4] C. D. Knutson, H. R. Duffin, J. M. Brown, S. B. Barnes, and R. W. Woodings. Dynamic autonomous transport selection in heterogeneous wireless environments. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2004.

[5] W. C. Y. Lee. Estimate of channel capacity in rayleigh fading environment. *IEEE Transactions on Vehicular Technology*, 39(3):187–189, Aug. 1990.

[6] A. B. McDonald and T. Znati. A path availability model for wireless ad-hoc networks. In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 1999.