International Journal of Wireless Information Networks, Vol. 8, No. 1, 2001

Fuzzy Reasoning for Wireless Awareness

Liang Cheng[1] and Ivan Marsic[1]

In hybrid communication environments where both wired and wireless links exist, performance of the quality-of-service (QoS) provision can be enhanced if an application knows whether there exist wireless links in the communication channel and adapts its behavior accordingly. This paper presents a scheme using fuzzy reasoning to make applications wireless-aware. Based on the studies of statistical patterns of round-trip time (RTT) in communication sessions via wired and wireless links, the mean value and variance of RTTs are used as fuzzy inputs, and the confidence of existence of wireless links in the communication channel as the fuzzy output. Simulations and experiments show that fuzzy reasoning for wireless awareness (FuRWA) is a feasible way to enhance QoS in hybrid communication environments at the application layer. FuRWA adds intelligence into endpoints without modifying protocol stacks, and can handle scenarios where a wireless link is an intermediate component of a communication path.

KEY WORDS: Wireless communication; quality of service; fuzzy logic; round-trip time.

1. INTRODUCTION

Portable computing devices, such as laptops, handheld computers, PDAs, palmtops, and pen-based computers, have become popular in recent years. In addition, wireless products ranging from local area networks (LANs) to wide area networks (WANs) are becoming commodities with the development of wireless communication technologies. Wireless computing, wireless communication, and wireless networks are becoming common in the daily life. This naturally leads to hybrid communication environments where both wired and wireless communication links exist.

There is a great amount of ongoing work on provisioning the current Internet with wireless accessibility and associated quality-of-service (QoS). Most of the research efforts are concentrated on the IP protocol layer problems such as mobile IP [13, 16] and routing protocols in MANET [4, 7], while some of them are focused on the transport layer, such as TCP performance in wireless networks [5, 9]. Relatively few researchers address the QoS problems at the application layer. One of the difficulties in addressing the QoS problems in a hybrid communication environment from the application's point of view lies in determining the characteristics of the communication links and making the application adaptive to it. An important network characteristic is the information about the existence of wireless links.

In this paper, round-trip time (RTT) data are used to detect wireless links in the communication path. Generally speaking, a RTT is the interval between the sending of a packet and receiving its acknowledgment. It includes both the network delays, such as router-queue delay and link-propagation delay as well as host-processing delay, such as the time that takes the sender and receiver to process the packet and the acknowledgment. Normally, the propagation delay is a significant contributor to the round-trip time. RFC 889 [11] states that RTTs over the wide area networks with wired links show Poisson distribution characteristics.

We conducted experiments on non-ambiguous measurement of RTT values of communication sessions

[1] Department of Electrical and Computer Engineering and the CAIP Center, Rutgers—The State University of New Jersey, 94 Brett Road, Piscataway, NJ 08854, USA; Tel: 1-732-445-0600, Fax: 1-732-445-4775; e-mail: chengl@caip.rutgers.edu; marsic@caip.rutgers.edu

via both wired and wireless links. A comparison study shows that the RTT values of communication sessions over wired links and wireless links have different statistical patterns. Moreover, the distribution patterns are overlapping, so that differentiation is not possible by a simple thresholding scheme. Based on this result, an approach is proposed, called *fu*zzy *r*easoning for *w*ireless *a*wareness (FuRWA), to embed intelligence into the application to detect wireless links by using fuzzy logic reasoning. Simulations and experiments are conducted to demonstrate its feasibility.

Compared to other methods for link identification, e.g., bandwidth measurements, FuRWA is simpler, faster and more network-friendly. For example, measuring available bandwidth [8] requires saturating the networks ("filling the volume of the pipe"). This requires search for the current throughput value and further requires much longer sessions than with FuRWA.

Based on FuRWA, we proposed two possible application scenarios with wireless-aware and adaptive applications. One of the scenarios rests on the unicast client-server paradigm and the other rests on the multicast paradigm. In the first scenario, the server provides different services to different clients depending on whether a wireless link was detected during the connection establishment phase. In the second scenario, the session manager creates a new multicast session in addition to the original one. The new session carries lower data traffic to and from the participants whose communication path includes wireless link(s). Wireless link detection using FuRWA takes place before the actual data are transmitted.

1.1. Related Work

To our best knowledge, there is no prior work on detecting the existence of the wireless links. Awareness about the existence of wireless links is important in improving TCP performance over the wireless links. One approach to wireless awareness is to embed wireless-specific mechanisms into the network and keep the end-points wireless-unaware [1]. In this case, the base stations or mobile switching stations support the wireless-specific TCP protocol and make the wireless link transparent to the applications. The restrictions are: (*i*) base stations are owned and administered by different autonomous entities that may not even be running IP internally, (*ii*) the base stations need to maintain significant information, such as TCP states.

Another approach is to make end hosts wireless aware. There are several efforts in this area that are loosely related to our work. Wireless links are implicitly detected in [3] and [17], where RTT and TCP throughput are used to differentiate transmission losses from congestion losses for TCP congestion avoidance. However, recent research shows that such techniques do not perform well [2]. Sinha *et al.* [15] also use RTTs for loss differentiation but in wireless WANs, so the RTTs provide more reliable information. These end-to-end mechanisms are focused on the TCP layer and thus have the drawbacks of requiring protocol-stack alterations. The method presented here focuses on enhancing the QoS at the application layer thus avoiding modifications to the underlying protocol stack.

Most, if not all, of the previous research on packet communication over wireless media presumes network topologies where the wireless link exists only as the last hop in the communication path or all communication links are wireless, as in MANETs [4, 7]. Thus they do not need to perform wireless link detection. The method presented here can be applied to these cases as well as to the scenario where a wireless link is an intermediate component along the communication path.

The paper is organized as follows. Section 2 studies the statistical patterns of the RTT values over wired and wireless links by a series of experiments. The differences in the patterns are used to determine whether there exist wireless links in the communication channel before the data is transmitted. The fuzzy reasoning engine of FuRWA is described in Section 3 and the simulation and experimental results are presented in Section 4. The details of the two possible application scenarios for using FuRWA in the unicast client/server paradigm and the multicast paradigm are exposed in Section 5. Finally, Section 6 draws conclusions and proposed future research.

2. RTT STATISTICAL PATTERNS

Measuring RTT values is critical for TCP performance because TCP dynamically sets an appropriate retransmission timeout value based on the RTT measurement [10]. The original method for mesuring RTT in TCP is as follows. Every time TCP sends a datagram, it records the time instant. When an acknowledgment (ACK) for the datagram arrives, TCP again gets the time instant and takes the difference between the two times as the current RTT value. However, this method suffers from the so-called *retransmission ambiguity* prob-

Table I. Karn/Partridge Algorithm for Measuring RTT in TCP

for each datagram with sequence number i
- record the sending time instance t_{si};
- when ACK for datagram with sequence number i is received, record the receiving time instant t_{ri};
- **if** datagram i without retransmission
  $\text{RTT}_i = t_{ri} - t_{si}$;
  end if

end for

Table II. Algorithm for Measuring RTT in FuRWA

for $i = 1$ to *maxSamples*
- the client sends a handshake packet with a unique ID, integer $i \in [1, maxSamples]$, to the server, records the time instant t_s, and starts a timer with the timeout value T_0.
- upon receiving the handshake packet, the server bounces the packet back to the client with the same unique ID i. No other computation is performed.
- when the client receives the reply packet with the ID i, it records the receiving time instant t_r, stops the timer, computes the difference between the sending and receiving time instants, and sets $\text{RTT}[i] = t_r - t_s$

  if the timer expires
  $\text{RTT}[i] = T_0$
  end if

end for

lem since an ACK acknowledges the receipt of a datagram rather than an individual transmission. To state it another way, when a datagram gets retransmitted and an ACK gets received at the sender, it is impossible to determine if the ACK should be associated with the first or the second transmission of the datagram.

The Karn/Partridge algorithm [10] solves the retransmission ambiguity problem by simply not taking samples of RTT whenever TCP retransmits a datagram. It only measures RTT for datagrams that have been sent without retransmission. Table I shows this algorithm.

In our algorithm, the retransmission ambiguity problem is avoided by using a unique datagram ID for each packet sent. In other words, the algorithm for measuring RTT in FuRWA is packet-ID-transmission-oriented instead of datagram-sequence-number-oriented as in TCP.

FuRWA algorithm for measuring the RTT values is straightforward and based on the client/server model, as presented in Table II. The algorithm uses User Datagram Protocol (UDP) packets as probes. RTT[*maxSamples*] is an array recording the sampled RTT values in a communication session, where *maxSamples* is the maximum number of samples per measurement.

The handshake packets are sent in a stop-and-wait fashion rather than all at once because this yields more accurate RTT measurements. Every packet gets processed immediately as received at the end host, without being delayed in a buffer.

One measurement consists of *maxSamples* samples of RTT collected in a cycle shown in Table II. After each measurement, the mean value and variance of the collected RTTs are computed according to equations (3) and (4) described in Section 3 below. All the RTTs along with their mean value and variance are recorded in a database. In the experiments, the measurement and computation is conducted once every three seconds.

Two series of experiments have been conducted: client and server communicate across the Internet or across a LAN. In both cases, the size of the UDP packets is 0.5 KB. The reason for using small size packets, i.e., ≤0.5 KB, is that for small size packets link bandwidth is not related to RTT (see, e.g., Section 1.1.4 in [14]).

The remaining parameters of interest are as follows. *maxSamples* is set as 128 and T_0 is set to 140 ms if there exist wireless links in the communication channel, while it is set to 100 ms otherwise. The IP addresses of the servers in the Internet and the LAN sessions are 152.3.17.193 and 128.6.37.99, respectively. The IP addresses of the wired and wireless clients in both cases are 128.6.37.117 and 128.6.37.101, respectively. A Proxim spread spectrum wireless LAN was used in the wireless case. A wireless laptop with RangeLAN2 7400 PC card communicates via the RangeLAN2 7510 Ethernet Access Point as the base station. We believe that wireless products from other companies should demonstrate similar statistical patterns to those found with this testbed. In the Internet experiments, the geographic distance between the client and the server was about 500 miles, and the Internet distance was 9 hops. These two series of experiments were conducted continuously over several days. The results for the mean value and variance of RTTs are shown in Figure 1.

According to Figures 1a and 1b, the distributions of the mean values of RTTs are very different in the wired and wireless cases. First, the mean value in the wired case is smaller than that in the wireless case and there is almost no overlap between their distributions. Second, the shape of the distribution in the wired case is more pulse-like, unlike the wireless case where it shows significant spreading. Thus, the RTT variance in the wired case is smaller than in the wireless case as shown in Fig-

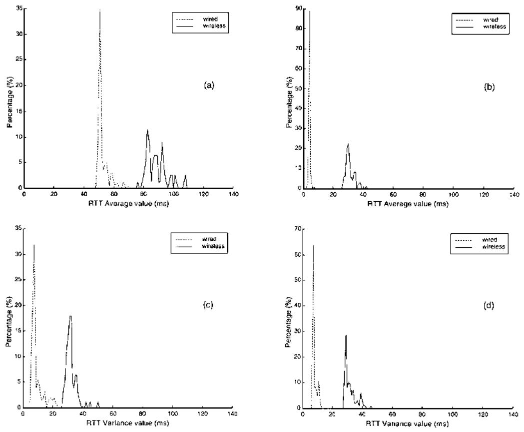

Fig. 1. Distributions of RTT average and variance. (a) Distribution of RTT average for an Internet session. (b) Distribution of RTT average for a LAN session. (c) Distribution of RTT variance for the Internet session. (d) Distribution of RTT variance for the LAN session.

ures 1c and 1d. The RTT variances of both the Internet and LAN sessions in the wired case are small while those of the wireless case are relatively large. These characteristics of the statistical patterns for both cases were confirmed by additional experiments with various small size UDP packets.

One of the reasons for the RTT-distribution-pattern differences between wired and wireless cases is that wireless MAC (Media Access Control) standards, such as 802.11, use three-way-handshake to avoid the hidden-termination problem, which contributes to an additional delay and RTT spread. In addition, wireless links have relatively high loss-rate compared to that of wired links due to the characteristics of physical media, which also contributes to RTT differences due to the retransmission at the MAC layer.

Based on this result, the existence of wireless link(s) in the communication channel can be automatically detected by analyzing the RTT distribution patterns for the current session. However, simple algorithms such as constant threshold are not suitable for the task since the wired and wireless RTT distributions overlap. For example, Figures 1a and 1b show that the RTT average in the wired case is roughly in the range (4, 7) in the LAN and (45, 75) in the Internet, while for the wireless case it is in the range (25, 45) in the LAN and (75, 110) in the Internet. Therefore, no constant threshold can distinguish these cases for both LAN and Internet sessions. Overlaps are even more serious in diverse environments such as transcontinental sessions so that wired and wireless RTT distributions overlap in a single communication scenario, i.e., LAN only or the Internet environment only. Similar analysis applies to RTT variances. Therefore, a more powerful method is required to identify the characteristics of the communication links. We use fuzzy logic as an intelligent reasoning method for this task. We chose it over other classification techniques such as neural networks, since fuzzy reasoning is less computationally intensive.

Fuzzy logic was first introduced by L. A. Zadeh in l965 [18], and it has since been widely used to construct intelligent systems. In fact, fuzzy logic with its intrinsic non-linearity has similarity with the reasoning conducted by human beings, i.e., computing with words [19]. A key feature of fuzzy logic is that it can deal with the uncertainties that exist in physical systems. The fuzzy reasoning engine of FuRWA uses the mean values and variances of the RTTs as the fuzzy inputs, and outputs the confidence about the existence of wireless links in the communication channel.

3. FUZZY REASONING ENGINE OF FuRWA

3.1. Review of Fuzzy Logic Theory

3.1.1. Fuzzy Sets

Fuzzy sets have been interpreted as membership function μ_X associated with each element x in the universe of discourse U with a number $\mu_X(x)$ in the interval [0, 1] as:

$$\mu_X : U \longrightarrow [0, 1] \tag{1}$$

A *fuzzifier* maps crisp data $x \in U$ into a fuzzy set $X \in U$, and μ_X gives the degree of membership of x to the fuzzy set X, i.e., a real number in the range [0, 1] where 1 denotes full membership and 0 denotes no membership. Therefore, fuzzy sets can be considered as an extension of classical crisp sets: crisp sets only permit full membership or no membership while fuzzy sets permit partial membership.

The language of human beings has many fuzzy words such as "cool," "comfortable," "hot" and so on. Thus fuzzy sets are often expressed by words (*fuzzy linguistic variables*).

3.1.2. Fuzzy Rules and Fuzzy Reasoning

Each fuzzy rule in the rulebase has p antecedent clauses that define conditions and one consequent clause that defines the corresponding action. As is well known, a rule with q consequents can be decomposed into q rules, each having the same antecedents and one different consequent. The general form of the lth fuzzy rule in the rulebase is:

$$R^l : \text{IF } x_1 \text{ is } F_1^l \text{ and } x_2 \text{ is } F_2^l \text{ and } \cdots x_p \text{ is } F_p^l \text{ THEN } y \text{ is } G^l \quad (2)$$

where F_k^l and G^l are fuzzy sets associated with the input and output fuzzy variables x_k and y, $k = 1, \cdots, p$. The information embedded in the fuzzy rules can be numerically processed by *fuzzy reasoning.*

3.1.3. Fuzzy Logic System

A *fuzzy logic system* processes crisp data at the input and produces crisp data at the output. Therefore a *fuzzifier* is used at the input of the system to convert crisp to fuzzy data, whereas a *defuzzifier* is used at the output to convert fuzzy into crisp data [12].

3.2. Fuzzy Reasoning for Wireless Awareness (FuRWA)

The basic idea of FuRWA is motivated by the measurements and studies of the RTT values of the communication sessions described in Section 2. As demonstrated, the mean value and variance of RTTs between two communication endpoints via a wired data channel are small, while they are relatively large when there exist wireless links in the data channel. If the RTT values collected by the application show an abnormal pattern such as large mean value and variance, then the application should deduce the existence of wireless link(s). Based on this idea, the fuzzy reasoning engine of FuRWA can be realized as described below.

3.2.1. Description of the Fuzzy Input

The input variables of the fuzzy reasoning engine are defined as the mean value t, equation (3), and the biased estimate of variance δ_t, equation (4), of the round-trip times t_i, $i = 1, 2, \cdots$, *maxSamples.* Generally, the RTT value t_i ranges from zero to infinite milliseconds. However, there are always some timers such as the connection timer and retransmission timer in the connection-oriented applications to deal with the case of packet loss, i.e., when RTT equals infinity. Therefore one can assume that t_i is constrained to $[0, T_{\max}]$ without the loss of generality.

$$t = \frac{1}{n} \sum_{i=1}^{n} t_i \quad (3)$$

$$\delta_t = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (t_i - t)^2} \quad (4)$$

According to (3) and (4) the discourse of the fuzzy input variables t and δ_t should be $[0, T_{\max}]$. Now a fuzzifier is needed to convert the crisp values t and δ_t into fuzzy data.

As stated in the previous section, a fuzzifier maps a crisp data $x \in U$ into a fuzzy set $x \in U$, where U is the discourse of the data. Generally there are two kinds of fuzzifiers called *singleton fuzzifier* and *nonsingleton fuzzifier* [12]. In the first case, a crisp data $x \in U$ is mapped into a fuzzy set X with support x_i, where $\mu_X(x_i) = 1$ for $x_i = x$ and $\mu_X(x_i) = 0$ for $x_i \neq x$. In the second case, the crisp data $x \in U$ is mapped into a fuzzy set X with support x_i, where $\mu_X(x_i) = 1$ for $x_i = x$ and decreases while moving away from $x_i = x$. Conceptually, the nonsingleton fuzzifier implies that the given input x is most likely to be the correct value of all the values in its immediate neighborhood. However, because of the uncertainties in the input, neighboring points are also likely to be the correct values, but to a lesser degree. It is up to the designer to determine the shape of the membership function based on an estimate of the uncertainties present. Generally, it is a logical choice for the membership function to be symmetric about x since the effect of uncertainties is most likely to be equal on all data.

In this paper the Gaussian membership function $\mu_X(x_i) = \exp\,[-(x - x_i)^2/2\sigma^2]$ is used, where the variance σ^2 reflects the width (spread) of $\mu_X(x_i)$, because of computing simplicity and control of spread. Note that larger values of the spread of μ_X imply that more uncertainties are anticipated to exist in the given data. Also the discourses of the fuzzy input variables t and δ_t are divided into three fuzzy sets as shown in Figure 2. The corresponding fuzzy linguistic variables are S (Small), M (Medium), and L (Large). The set of these three fuzzy linguistic variables for fuzzy input t and δ_t can be denoted as (5) and (6):

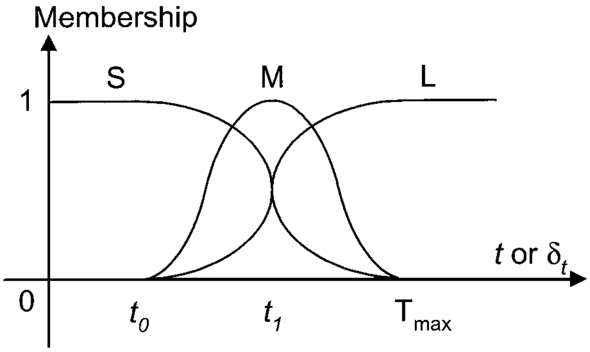

Fig. 2. Discourse and fuzzy sets of fuzzy input variable.

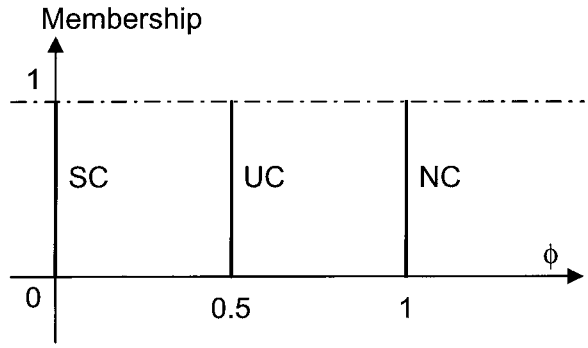

Fig. 3. Discourse and fuzzy sets of fuzzy output variable.

$$S(t) = \{S_t, M_t, L_t\} = \{F_t^1, F_t^2, F_t^3\} \quad (5)$$

$$S(\delta_t) = \{S_\delta, M_\delta, L_\delta\} = \{F_\delta^1, F_\delta^2, F_\delta^3\} \quad (6)$$

3.2.2. *Description of Fuzzy Rules and Fuzzy Output*

The fuzzy rules used in the reasoning engine are a special case of the general expression of equation (2) where G^l is now a singleton fuzzy set, i.e., a crisp value. The confidence about the existence of wireless links in the communication channel is set as the output of the fuzzy reasoning engine. Here the discourse of fuzzy output variable ϕ has been divided into three singleton fuzzy sets as shown in Figure 3.

The corresponding fuzzy linguistic variables are SC (Strong Confidence), UC (Uncertain), and NC (No Confidence). The set of these three fuzzy linguistic variables for fuzzy output ϕ can be denoted as:

$$S(\phi) = \{SC, UC, NC\} = \{F_\phi^1, F_\phi^2, F_\phi^3\} \quad (7)$$

The fuzzy rules can be expressed as:

$$R^{l,m}: \text{ IF } t \text{ is } F_t^l \text{ and } \delta_t \text{ is } F_\delta^m \text{ THEN } \phi \text{ is } F_\phi^{n(l,m)} \quad (8)$$

where l, m, $n(l, m) \in I$, $I = \{1, 2, 3\}$ and the relationship $n(l, m)$ can be nonlinear. The description of the specific fuzzy rules is given in Table III.

For example, the first rule states that: IF t is S and δ_t is S THEN ϕ is *NC*. This rule means that if both the mean value and the variance of measured RTTs are small then there is no confidence about the existence of wireless links in the communication channel. Another example rule states that: IF t is L and δ_t is S THEN ϕ is *UC*. This means that even if the variance of RTTs is small but the mean value of RTTs is large, the existence of wireless links is still uncertain because it is possible that all the test packets were lost and all the RTTs were set as the timeout values.

Table III. Fuzzy Rules

$\delta_t \setminus \phi \setminus t$	S	M	L
S	NC	NC	UC
M	NC	UC	UC
L	NC	SC	SC

3.2.3. *Description of Fuzzy Reasoning*

The general form of fuzzy reasoning can be expressed as follows: for each fuzzy input pair t and δ_t, the corresponding fuzzy set F of the fuzzy output ϕ has the membership function as (9):

$$\mu_F(\phi) = \perp_{l,m \in I}(\diamond(\mu_{F_t^l}(t), \mu_{F_\delta^m}(\delta_t), \mu_{F_\phi^{n(l,m)}}(\phi)) \quad (9)$$

$$\mu_F(\phi) = \text{Max}_{l,m \in I}(\text{Min}(\mu_{F_t^l}(t), \mu_{F_\delta^m}(\delta_t), \mu_{F_\phi^{n(l,m)}}(\phi)) \quad (10)$$

where "$\perp$" is a decompositional operator, and "$\diamond$" is a compositional operator. In this paper Max is used as decompositional operator and Min as the compositional operator. Then (9) can be rewritten as (10).

3.2.4. *Description of the Defuzzifier*

The method of gravity-of-mass (GOM) is used to perform defuzzification. It can be expressed as (11):

$$\phi = \frac{\int \phi \mu_F(\phi) d\phi}{\int \mu_F(\phi) d\phi} \tag{11}$$

3.2.5. Simplification

Because the fuzzy output sets are singleton ones, the description of fuzzy reasoning and defuzzification can be simplified and combined together to be expressed as (12), where core($F_\phi^{n(l,m)}$) represents the core value of the fuzzy set $F_\phi^{n(l,m)}$:

$$\phi = \frac{\sum_{l,m=1}^{3} \mu_{F_t^l}(t)\mu_{F_\delta^m}(\delta_t)\text{core}(F_\delta^{n(l,m)})}{\sum_{l,m=1}^{3} \mu_{F_t^l}(t)\mu_{F_\delta^m}(\delta_t)} \tag{12}$$

Table IV. Algorithm of FuRWA for Simulations and Experiments

count is initialized to zero;
set *maxCycles*;
while true **do**
- extract/measure 100 RTT values; // measured in experiments
- increment *count* by one;
- compute the mean value t and variance δ_t of these 100 RTTs as the inputs of the fuzzy reasoning engine of FuRWA;
- get the output of the fuzzy reasoning engine as the confidence of the existence of wireless links in the communication channel ϕ;

  if $\phi \in$ [0.3, 0.7] and *count* < *maxCycles*
      continue;
  else if $\phi > 0.7$
      wireless link exists;
  else if $\phi < 0.3$
      wireless link does not exist;
  else
      wireless link exists;
  end if
  break;
end do

4. SIMULATIONS AND EXPERIMENTS

4.1. Simulations with FuRWA

Four different simulation scenarios exist, i.e., Internet or LAN session with or without wireless links. To test the effectiveness of the fuzzy engine of FuRWA, a new dataset, which includes 10,000 data for each scenario, is collected by the RTT measurement. The simulation was designed as follows:

(1) Choose a scenario and set the maximum number of detection cycles, *maxCycles*, for the scenario.
(2) For each cycle, *randomly* extract 100 consecutive RTT values from the corresponding scenario dataset collected.
(3) Compute the mean value and the variance of these 100 RTTs as the inputs of the fuzzy reasoning engine.
(4) Get the output of the fuzzy reasoning engine as the confidence of the existence of wireless links in the communication channel.
(5) If the output is UC (Uncertain), extra detection cycles are required to draw conclusion about the existence of wireless links. If the number of executed cycles is larger than *maxCycles*, then the conclusion is drawn as that a wireless link exists. Otherwise, the output (SC or NC) is used as the conclusion of the detection.
(6) Verify whether the conclusion matches the actual scenario. If they match, count this as a correct detection. Otherwise count it as a false detection.

Repeat the procedure 10,000 times for each scenario.

The algorithm is summarized in Table IV. The parameters are set as follows:

- The mean values of Gaussian membership functions of the fuzzy linguistic variables S_t, M_t, L_t are 10, 50, and 90 respectively.
- The variances of Gaussian membership functions of the fuzzy linguistic variables S_t, M_t, L_t are each 20.
- The mean values of Gaussian membership functions of the fuzzy linguistic variables S_δ, M_δ, L_δ are 10, 20, and 30, respectively.
- The variances of Gaussian membership functions of the fuzzy linguistic variables S_δ, M_δ, L_δ are all 3.
- NC, UC, and SC range from [0, 0.3), [0.3, 0.7], and (0.7, 1], respectively.

The details of other components of FuRWA are described in Section 3. The input-output relationship of the fuzzy reasoning engine of FuRWA is illustrated in Figure 4. The relationship is nonlinear, indicating the potential to differentiate any RTT pattern.

The simulation results show no false detection, i.e., false detection rate $P_f = 0$. Thus the relation between the

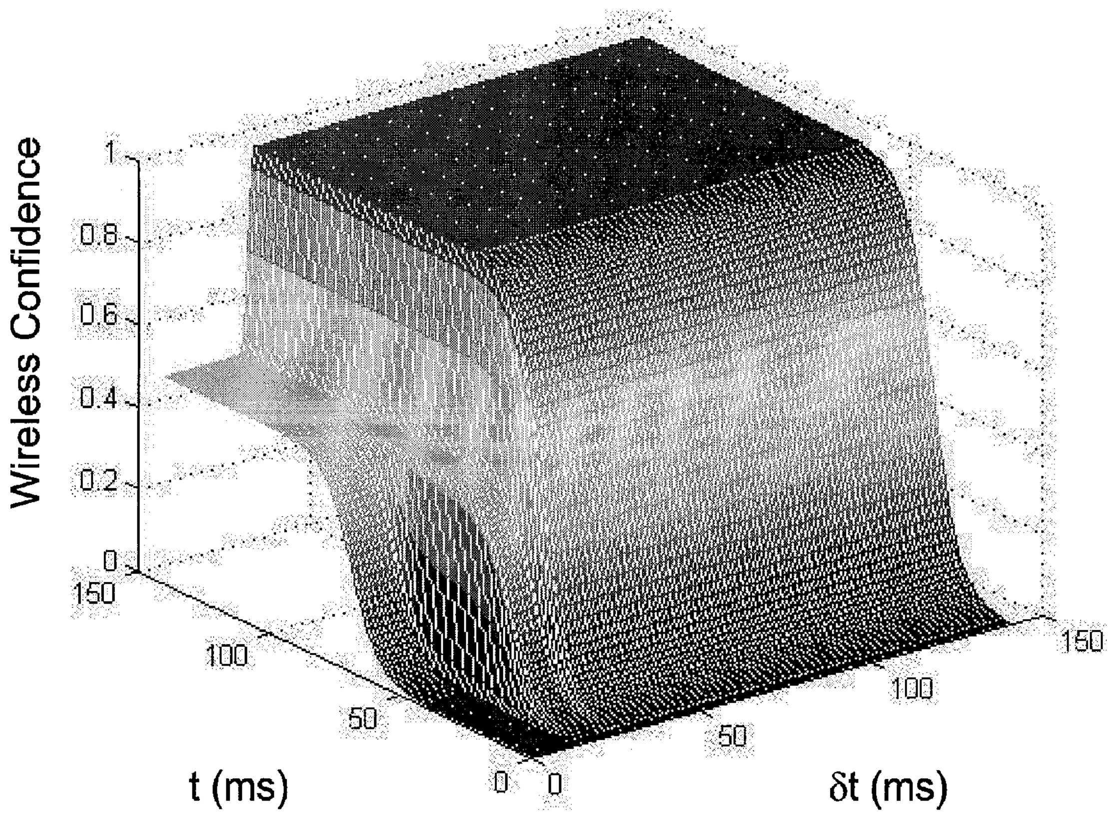

Fig. 4. Nonlinear input-output relationship of FuRWA engine.

uncertainty rate P_u and the correct detection rate P_c is $P_u = 1 - P_c$. Figure 5 shows the relationship between P_c and the number of detection cycles. The graph implies that the uncertainty rate diminishes with the increasing number of detection cycles. In fact, the results of scenarios A and B show no uncertainty at all. In case an uncertainty occurs due to network transient behavior, extra detection cycles can be carried out until the uncertainty is removed. Moreover the correctness of the simulation is proved by the following experiments.

4.2. Experiments with FuRWA

A simple videoconferencing tool, implemented in the Java programming language, was used to test the feasibility of the FuRWA scheme in the LAN environment. The videoconferencing session multicasts through a hybrid LAN that includes both fixed workstations with wired links and a mobile laptop with wireless links. The Proxim wireless LAN described in Section 2 was used for the wireless connectivity. The mobile laptop was moved at a walking speed within a 30-meter radius from the base station. No roaming or handoff was considered. Both cases were tested with the laptop being in and out of the line-of-sight of the base station.

The algorithm of FuRWA is the same as that for the simulations (see Table IV). Based on the simulation results, we set *maxCycles* = 4 for the experiments because it can deal with scenario B, and C perfectly. As for scenario D, if the output of FuRWA is still UC after four detection cycles, we can draw the correct conclusion that there exists a wireless link. The average time needed by the algorithm to reliably detect the existence of wireless links ranges from 0.5 to 10 seconds, mainly due to the measurement time.

Due to the physical constraints of wireless links in our hybrid LAN, the packet loss rate at the mobile laptop increases sharply from 2–4% to 80% when the source's traffic rate is greater than 335 Kbps. Thus the video stream received at the mobile laptop shows unacceptable jitter and sometimes all frames are lost so that the video is frozen.

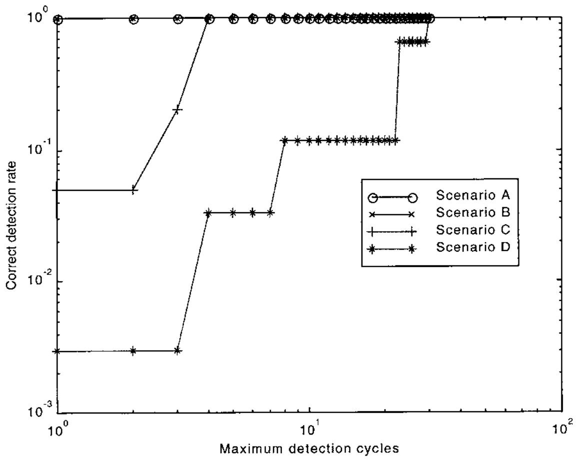

Fig. 5. Simulation results: Scenario A: Lan session without wireless links. Scenario B: Internet session with wireless links. Scenario C: Internet session without wireless links. Scenario D: LAN session with wireless links.

When the FuRWA scheme is used, the data traffic to the mobile laptop is reduced because the session manager is aware of the wireless links' existence and sends the mobile laptop a lightweight video stream via user-defined reduced-resolution/framerate video frames.

5. APPLICATION SCENARIOS

5.1. Unicast Client/Server Paradigm

In recent decades, the client/server paradigm has grown to be the most common design for network applications. It provides a logical breakdown of application functionality. In an ideal environment, the server side of the application handles all common processing, and the client side handles user-specific processing. Today many Internet applications are of the client/server type, such as Web browsing, email, FTP, telnet, and so on. Our idea for embedding wireless awareness into applications implementing such a paradigm is illustrated in Figure 6.

In Figure 6, the fully geared client is connected with the server by wire links such as 10BaseT, while the partially geared client has wireless link(s) in its communication channel to the server. It is also assumed that the clients have sufficient computing power to handle RTT computation in real time.

As Figure 6 shows, there is a wireless-aware connection proxy at the server side. When a client asks for a connection and service, it first communicates with the

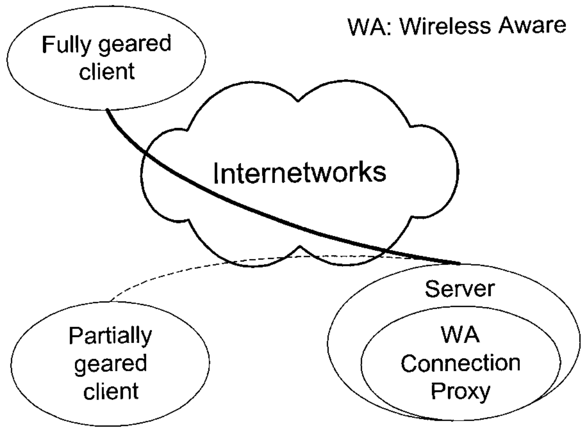

Fig. 6. Wireless awareness in the unicast client/server paradigm.

proxy that measures the round trip time and uses the fuzzy reasoning engine to draw a conclusion about the existence of the wireless links.

The server provides different levels of quality-of-service (QoS) to different clients based on the conclusion drawn by the wireless-aware connection proxy. Thus, a live video server can send high-resolution/framerate video streams to the fully geared clients and normal or low-resolution video streams to partially geared clients to adapt to the limited bandwidth, high packet loss rate, and stringent power constraints of the wireless link.

5.2. Multicast Paradigm

The multicast paradigm is used when the application features multiple participants. Thus network resources, such as bandwidth, are used in an efficient way by reducing the number of datagram copies transmitted in the network.

However, if one of the participants communicates with the multicast session via a wireless link and the application is not aware of it, then the data channel to that participant may easily get congested so that the performance of the application will be degraded, especially for that participant. For example, in a videoconferencing scenario, some participants may be using mobile devices such as wireless laptops or wearable computers. As more participants join the session, the video streams to the mobile devices will experience unacceptable delays and jitter if the rate-control by the codec of the videoconferencing application is not aware of the existence of the wireless links and does not adapt to them by using lower transmission rates.

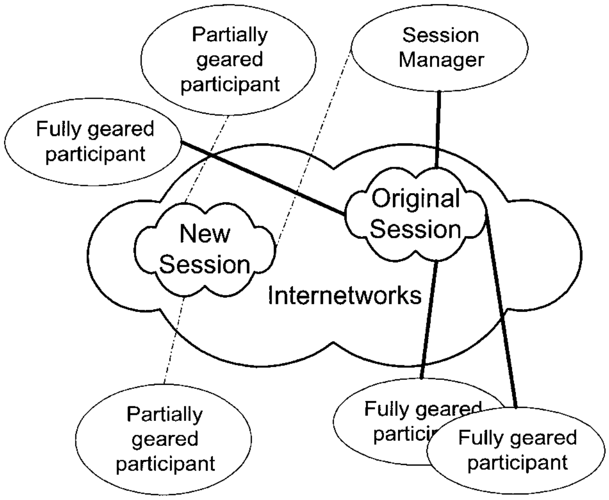

Fig. 7. Wireless awareness in the multicast paradigm.

Figure 7 shows the scheme to embed wireless awareness (WA) into the multicast applications. The procedure can be described as follows:

(1) The session manager announces the original multicast session.
(2) A participant who wants to join the session handshakes with the session manager via a separate point-to-point channel so that the information about the round-trip time gets collected. If the session manager concludes that the participant is partially geared, it creates a new multicast session for partially geared participants, if such does not already exist, and requests the partially geared participant to join the new session. Otherwise the participant joins the original session.
(3) The session manager acts as a bridge between the new session and the original one. In fact, it receives the traffic streams from one session and converts them to suit the other session and dispatches the converted stream to the other session. Finally, every participant can send its data stream to and receive the data streams from others via the corresponding multicast session.

The videoconferencing application used in the experiment reported in Section 4 serves as an example. After the session manager announces the videoconferencing session, participants can join, send and receive the live video streams. The session manager will do the following during the videoconferencing session:

- Create a new multicast session for the participants communicating via wireless channels such as those using mobile laptops.
- Notify videoconferencing applications running on the mobile laptops to send and receive low-resolution/framerate video streams instead of normal ones.
- Receive normal-resolution/framerate video streams from the original session, convert them into low-resolution/framerate streams and send them to the new session.
- Receive low-resolution/framerate video streams from the new session, convert them into normal-resolution/framerate streams and send them to the original session.

In this way, traffic is properly adjusted for the wireless links so to avoid congestion, and all the participants can receive acceptable video streams except for the resolution/framerate difference between the fully and partially geared participants.

6. CONCLUSIONS

In hybrid communication environments, in which there exist both wired and wireless communication links, performance of QoS provision can be enhanced by knowing whether there exist wireless links in the communication channel, and making the application adaptive to it. In fact, for the application to achieve behavior adaptation, it first needs to be aware of the network characteristics. Awareness is the bsis of adaptation.

This paper presents a scheme called FuRWA (*fu*zzy *r*easoning for *w*ireless *a*wareness), which uses RTT distribution analysis to make applications wireless-aware. Based on the studies of different statistical patterns of round-trip time over wired and wireless links, the mean value and variance of RTTs are used to distinguish wireless links. This information is input into a fuzzy reasoning engine and the output is the confidence about the existence of wireless links in the communication channel. According to the simulation and experimental results, FuRWA is a feasible way to enhance quality-of-service in hybrid communication environments.

FuRWA differs from other approaches by providing QoS at the application layer. Another distinguishing point is that it can deal with the scenario where a wireless link is an intermediate component along the communication path.

As part of our future research, we plan to continue with thorough investigation and analysis of the RTT distribution patterns in both the Internet and LAN environments. In complex hybrid communication scenarios such as transcontinental communications with large distances and different backbone qualities, the wireless LAN delay and Internet delay for a wireless user could be either smaller or greater than the Internet delay alone for a wired user, depending on their respective Internet delays. We will investigate the robustness of FuRWA in such environments.

We are also exploring further possibilities of applying the FuRWA scheme as part of a general wireless awareness framework [6], integral with throughput measuring tools. We believe that the needs of Web information transfer and display in hybrid communication environments may be best met by combining FuRWA with recent World Wide Web technologies, such as XML. By the combination of content description via XML, content presentation via HTML scripting, content choice via a wireless-aware engine, and separation of content description, presentation and choice, it is possible to provide a compelling solution for small mobile devices or low computing-power devices to communicate optimally with other computers in hybrid communication networks.

ACKNOWLEDGMENTS

This research is supported by DARPA Contract No. N66001-96-C-8510, NSF KDI Contract No. IIS-98-72995, and the Rutgers Center for Advanced Information Processing (CAIP). The authors would like to thank Dr. Edward J. Devinney and anonymous reviewers for their valuable comments.

REFERENCES

1. H. Balakrishnan, S. Seshan, E. Amir, and R. Katz, Improving TCP/IP performance over wireless networks, *Proceedings of the 1st Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '95)*, pp. 2–11, Berkeley, CA, USA, November 13–15, 1995.
2. S. Biaz and N. H. Vaidya, Distinguishing congestion losses from wireless transmission losses: A negative result, *Proceedings of the IEEE 7th International Conference on Computer Communications and Networks*, pp. 722–731, Lafayette, Louisiana, USA, October 12–15, 1998.
3. L. Brakmo, S. O'Malley, and L. Peterson, TCP Vegas: new techniques for congestion detection and avoidance, *Proceedings of ACM SIGCOMM'94*, pp. 24–35, London, England, August 31–September 2, 1994.
4. J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and J. Jetcheva, A performance comparison of multi-hop wireless ad hoc network routing approach, *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'98)*, pp. 85–97, Dallas, Texas, USA, October 25–30, 1998.
5. K. Chandran, S. Raghunathan, S. Venkatesan, and R. Prakash, A feedback based scheme for improving TCP performance in ad-hoc wireless networks, *Proceedings of the 18th International Conference on Distributed Computing Systems (ICDCS'98)*, pp. 472–479, Amsterdam, Netherlands, May 26–29, 1998.
6. L. Cheng and I. Marsic, Wireless awareness for multimedia applications, *Proceedings of the IFIP International Conference on Communication Technologies 2000*, pp. 1376–1382, Beijing, P.R. China, August 22–24, 2000.
7. M. S. Corson and J. Macker, Mobile ad hoc networking (MANET): routing protocol performance issues and evaluation considerations, *Request for Comments (RFC) 2501*, January 1999.
8. A. B. Downey, Using pathchar to estimate Internet link characteristics, *Proceedings of ACM SIGCOMM'99*, pp. 241–250, Cambridge, MA, USA, August 30–September 3, 1999.
9. M. Gerla, R. Bagrodia, L. Zhang, K. Tang, and L. Wang, TCP over wireless multihop protocols: simulation and experiments, *Proceedings of IEEE ICC'99*, Vancouver, Canada, June 1999.
10. P. Karn and C. Patridge, Improving round-trip time estimates in reliable transport protocols, *ACM Transactions on Computer Systems*, Vol. 9, No. 4, pp. 364–373, November 1991.
11. D. L. Mills, Internet delay experiments, *Request for Comments (RFC) 889*, December 1983.
12. G. C. Mouzouris and J. M. Mendel, Nonsingleton fuzzy logic systems: theory and application, *IEEE Transactions on Fuzzy Systems*, Vol. 5, No. 1, pp. 56–62, 1997.
13. C. E. Perkins, *Mobile IP: Design Principles and Practice*, Addison-Wesley: Reading, 1997.
14. L. A. Peterson and B. S. Davie, *Computer Networks: A Systems Approach*, 2nd edition, Morgan-Kaufmann Publisher, 2000.
15. P. Sinha, N. Venkitaraman, R. Sivakumar, and V. Bharghavan, WTCP: a reliable transport protocol for wireless wide-area networks, *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom'99)*, pp. 231–241, Seattle, Washington, USA, August 15–19, 1999.
16. J. Solomon, *Mobile IP: The Internet Unplugged*, Prentice Hall, 1998.
17. Z. Wang and J. Crowcroft, A new congestion control scheme: slow start and search (*Tri-S*), *ACM Computer Communication Review*, Vol. 21, pp. 32–43, January 1991.
18. L. A. Zadeh, Fuzzy sets, *Information and Control*, Vol. 8, pp. 338–353, 1965.
19. L. A. Zadeh, Fuzzy logic = computing with words, *IEEE Trans. on Fuzzy Systems*, Vol. 4, No. 2, pp. 104–111, 1996.

Liang Cheng is a Ph.D. Candidate of Electrical and Computer Engineering at Rutgers University. He received the B.S. degree from Huazhong University of Science and Technology, Wuhan, P.R. China,

in 1994, and the M.S. degree from Tsinghua University, Beijing, P.R. China, in 1997. His current research interests include wireless data communications, multimedia networks, distributed computing, network performance and control, and intelligent agent for information processing.

Ivan Marsic is an Assistant Professor of Electrical and Computer Engineering at Rutgers University. He received the B.S. and M.S. degrees in Computer Engineering from University of Zagreb, Croatia, in 1982 and 1987 respectively, and Ph.D. degree in Biomedical Engineering from Rutgers University, New Brunswick, NJ, in 1994. Prof. Marsic is the chief architect for the DISCIPLE system, which is an advanced groupware system that enables teams, consisting of individuals with specific roles, to collaboratively ccess, manipulate, analyze, and evaluate multimedia data using geographically ditributed networked information systems. He has authored over 50 journal and conference papers, and 3 book chapters. His current research interests include groupware, mobile computing, computer networks, and human-computer interfaces.