Correspondence

A Client-Side Statistical Prediction Scheme for Energy Aware Multimedia Data Streaming

Yong Wei, Suchendra M. Bhandarkar, and Surendar Chandra

***Abstract*—The recent proliferation of streaming multimedia on a variety of mobile devices has severely tested their battery lifetime. The long running nature of typical streaming applications results in significant energy consumption by the wireless network interface card (WNIC) in these mobile devices. In this paper we explore linear prediction-based client-side strategies that reduce the WNIC energy consumption to receive multimedia streams by judiciously transitioning the WNIC to a lower power consuming *sleep* state during the no-data intervals in the multimedia stream, without explicit support from the multimedia servers themselves. Experimental results on popular streaming formats such as Microsoft Media, Real and Apple QuickTime show that a linear prediction-based strategy performs better than history-based strategies that use simple temporal averaging.**

***Index Terms*—Energy-aware computing, linear prediction, mobile computing, multimedia streaming.**

I. Introduction

The recent proliferation of multimedia capable mobile computing devices and networking technologies have created enormous opportunities for mobile device users to communicate with one another using multimedia streams. A necessary criterion for the mass acceptance of mobile devices is acceptable battery life of these devices. There has been dramatic improvement in energy-aware design of systems, both, in terms of hardware and software. Unfortunately, advances in hardware and software are not matched by a corresponding increase in battery life. Thus, the usefulness of these mobile devices in watching and/or hearing streaming multimedia is restricted by battery capacity. Future trends in battery technology do not promise dramatic improvements in battery capacity that will make this issue disappear. Consequently, hardware or software solutions need to be developed at the system or application level to prolong battery life.

Previous work on power management for mobile devices includes spin-down policies for disks [5]–[8], scheduling policies for reducing CPU energy consumption [9], [10] and managing wireless communications [11]–[14]. An IEEE 802.11b Wi-Fi connection is a popular way for mobile consumers to access the Internet wirelessly. The energy consumption of the wireless network interface can be significant, especially for smaller devices. Since media streaming applications are typically long running, the power consumption of these applications needs to be taken care of. Early work by Stemm *et al.* [2] reports that the network interface draws a significant amount of power. Although

Manuscript received May 11, 2004; revised November 3, 2005. This work was supported in part by a research grant from the State of Georgia Yamacraw Program in Embedded Software Systems to S. M. Bhandarkar and S. Chandra. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Chang Wen Chen.

Y. Wei and S. M. Bhandarkar are with the Department of Computer Science, University of Georgia, Athens, GA 30602 USA (e-mail: yong@cs.uga.edu; suchi@cs.uga.edu).

S. Chandra is with the Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556 USA (e-mail: surendar@cse.nd.edu).

Digital Object Identifier 10.1109/TMM.2006.876232

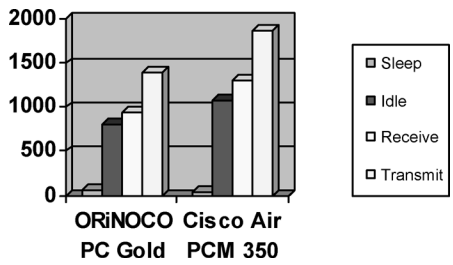

Fig. 1. Energy consumption rates of two WNIC's in various states.

dependent on the specific machine and wireless device, the energy consumption of wireless communication devices can represent over 50% of total system energy consumption for current handheld computing devices and up to 10% for high-end laptops [2]. Feeney *et al.* [18] also report the energy consumption measurements of an IEEE 802.11b WNIC in an *ad-hoc* networking environment and show that the energy consumption of the IEEE 802.11b WNIC has a complex range of behavior. Hence, it is important to look at techniques to reduce the energy consumed by the network interface used to download the multimedia stream.

The energy consumption rates of a wireless network interface card (WNIC) in the *sleep* state and in the *receive*, *transmit* or *idle* states are substantially different. Fig. 1 shows the power consumption rates of two popular WNIC's in the various aforementioned states [3]. The WNIC's energy consumption rate when receiving, transmitting data or when idling is substantially higher than when sleeping. Note that the WNIC cannot transmit, receive, or buffer data in the *sleep* state.

Lorch *et al.* [15] present a survey of software techniques for energy management. Havinga *et al.* [16] present an overview of energy management techniques for multimedia streams. Aggarwal *et al.* [17] describe techniques for processing video data for transmission under low power situations. A popular strategy to reduce the energy consumption of wireless network devices is by switching them to the lower power *sleep* state. Systems employing a strategy which enables switching of the WNIC to a low power consumption *sleep* state can achieve energy savings whenever possible without modifying the underlying application and without user-visible latency. Frequent switching to a low power consumption state also promises the added benefit of allowing the batteries to recover, thus exploiting the battery recovery effect [4].

Media transcoding is a popular strategy used to reduce the stream fidelity. This strategy reduces the stream size, and hence reduces the amount of network traffic. Reducing the network traffic has the potential of reducing the total energy consumed. However, if care is not taken to return the WNIC to the low power-consuming state for as often and as long as possible, reducing the amount of transmitted data will have a negligible effect on the overall client energy consumption.

The basic principle underlying the proposed energy-saving approach is to predict the time durations during which to suspend communication by switching the WNIC to a *sleep* state. Our analysis of typical streams shows that the WNIC spends most of the time waiting for stream packets in a higher energy consuming *idle* state. Even for a high bandwidth 2000 Kbps stream, the WNIC spends over 56% of the time in the *idle* state; illustrating the potential for significant energy savings. Our policies operate on the multimedia client without explicit coordination or help from the multimedia server. Multimedia/video data is

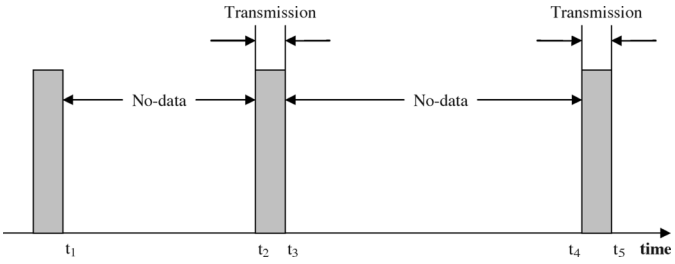

Fig. 2. Simplified stream packet transmission in a wireless network.

typically transmitted in the form of bursts of data packets with no-data periods between successive bursts. The bursty nature of the traffic is a consequence of the media streaming format and other network-related factors such as available network bandwidth, the buffering mechanism of the wireless access point and the traffic congestion control mechanism. Also, the bandwidth requirement of the multimedia stream is typically much less than what is provided by IEEE 802.11b, thus causing multimedia data traffic to appear bursty. The WNIC can be switched to its *sleep* state during these no-data intervals in order to save energy. Since we operate without explicit coordination with the server, this energy conservation strategy requires proper estimation of the time interval during which no data is expected to be received. If the WNIC is suspended too often or for too long a time duration during the wrong time periods, the users will miss the data sent to this client. On the other hand, if the WNIC is not suspended long and frequently enough, savings in energy consumption may not be appreciable.

Chandra [1] describes a client-side history-based scheme that transitions the WNIC to a power saving *sleep* state during the no-data intervals of a multimedia stream. The history-based scheme predicts the length of the next no-data interval by computing the average of the lengths of the past k successive no-data intervals. The Microsoft Media format was observed to benefit immensely from this client-side history-based scheme on account of the fact that Microsoft Media transmits large data packets at fairly regular time intervals. The benefits in the case of the Real and Apple QuickTime media formats were less apparent on account of the fact that the no-data time interval lengths were less regular. The work in this paper is a refinement of the client-side history-based scheme presented in [1]. Specifically, we present a statistical linear prediction-based strategy to predict the occurrences and lengths of the no-data time intervals. These predictions are used to select the time periods during which the communication is suspended (i.e., the WNIC is powered down to its *sleep* state) in a client-server environment where multimedia streams are being transmitted from a server to a mobile, power-constrained client.

The remainder of the paper is organized as follows. In Section II, we describe the proposed linear prediction-based scheme and present a brief outline of the history-based scheme proposed in [1] in the context of energy aware multimedia data streaming. In Section III, we describe the experimental setup, evaluation methodologies, measurement metrics and experimental results that compare the performance of the proposed linear prediction-based scheme with that of the history-based scheme. In Section IV, we provide a detailed interpretation of the experimental results. In Section V, we conclude the paper and outline directions for future research.

II. Linear Prediction-Based Approach

In a client-server wireless network environment, data packets are transmitted by the server in the form of discrete bursts, as shown in Fig. 2. This behavior is dependent on the particular multimedia streaming format used; for this work, we use the unmodified streaming formats of the Microsoft Media, Real and Apple QuickTime multimedia servers using the UDP protocol. Chandra [1] provides more details on the video data transmission statistics. Between two successive bursts there is a time interval during which there is no data being transmitted by the server. We refer to this time interval as the no-data interval. In Fig. 2, the lengths of the two no-data intervals are $t_2 - t_1$ and $t_4 - t_3$ respectively. The sequence of these no-data interval lengths can be looked upon as a time series. Empirical observations suggest that the length of a no-data interval bears statistical correlation to previously observed no-data interval lengths. This provides the motivation for the formulation of a client-side statistical linear prediction-based scheme to predict future no-data interval length values based on previous observations.

Linear prediction is a mathematical operation where a future value of a time series is estimated as a linear function of previously observed samples [19]. A common representation of the linear prediction model is given by

$$x'(n) = \sum_{i=1}^{p} a_i x(n-i) \tag{2.1}$$

where $x'(n)$ is the estimated or predicted no-data interval length, $x(n-i)'s$ are the previously observed no-data interval length values, and $a_i's$ are the predictor coefficients. The error generated by this estimate is given by

$$e(n) = x(n) - x'(n) \tag{2.2}$$

where $x(n)$ is the true no-data interval length value and $x'(n)$ the predicted value of the no-data interval length. A linear predictor optimizes the estimate by minimizing the estimation error. The two adjustable parameters of a linear prediction model are the model order p and the width of the time window used for training. The algorithm used in our approach is the one proposed by Burg [20]. The appropriate values of p and the width of the training time window are chosen empirically.

In a client-server wireless network environment, if the predicted lengths of the no-data intervals are frequently longer than the actual ones, the user will experience packet losses in the data stream being downloaded because many data packets arrive at the client's WNIC while it is in the *sleep* state. On this account, it is useful to add a relatively small negative bias to the sleep interval lengths predicted by the linear prediction algorithm in order to lower the data drop rate. This ensures that the client's WNIC is transitioned to the *idle* state *before* the next data packet arrives. Thus, the biased estimate of the no-data interval length is given by

$$x_b'(n) = x'(n) - B = \sum_{i=1}^{p} a_i x(n-i) - B \tag{2.3}$$

where $x_b'(n)$ is the biased prediction of the no-data time interval length. However, if the bias B is too large in magnitude, the resulting savings in battery energy may not be appreciable.

The history-based prediction scheme described in Chandra's previous work [1] predicts the no-data interval length by averaging the observed no-data interval lengths over the past k receive-idle cycles. It also varies the dependence of the prediction on past history by offsetting the predicted no-data interval length with a bias B as follows:

$$x'(n) = \frac{1}{k} \sum_{i=1}^{k} x(n-i) - B \tag{2.4}$$

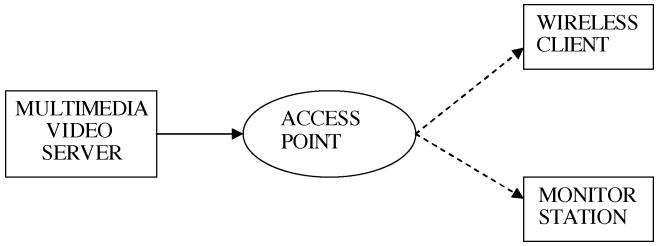

Fig. 3. Experimental setup.

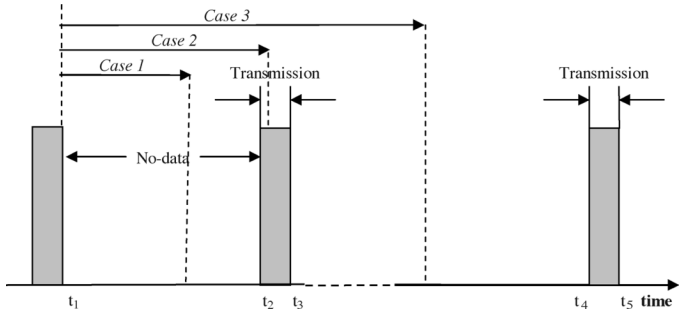

Fig. 4. Predicted no-data period and the WNIC energy consumption model.

Note that the history-based prediction scheme can be looked upon as a special (i.e., degenerated) case of the linear prediction scheme where all the predictor coefficients are identical.

III. Experimental Setup

A. System Description

The experimental system consists of a multimedia server with a wireless access point, and a mobile client with a wireless network interface card (WNIC), as shown in Fig. 3. The mobile client has a client-side proxy that is responsible for transitioning the WNIC to a low-power consumption *sleep* state during the predicted no-data time intervals. Ideally, since no data transfers are expected during the no-data time interval, there should be no loss of data. The traffic between the multimedia server's wireless access point and the mobile client is monitored by a monitoring station, which records the traffic flow in trace files. The multimedia stream used for our experiment is the *Wall* theatrical trailer. The *Wall* trailer is 1:59 min long and is digitalized to a high quality video stream.

Simulation of the client-side proxy is done using a typical WNIC power consumption model. We use the following published power parameters of a *Wavelan* 2.4 GHz wireless network interface card [16]; *sleep* state: 177 mw, *idle* state: 1319 mw, *receive* state: 1425 mw and *transmit* state: 1675 mw. We assume that the transition from the *sleep* state to *idle* state takes 250 μs and the wireless network provides a useful bandwidth of 4 Mbps.

B. Performance Metrics

In order to measure the efficiency of our approach, the following performance metrics are used.

Total Energy Consumed: This is defined as the total amount of energy consumed (in mJoules) by the client-side WNIC to receive the streaming video data transmitted by the multimedia server. The goal is to minimize this metric when the client-side WNIC receives a video clip.

Energy Consumed per KB Received: This is defined as the amount of energy consumed (in mJoules) per Kilobyte of data received by the client-side WNIC. The goal of our experiment is to minimize this metric. As we will see in Section IV, due to the inaccuracy of client-side prediction, some of the streaming video data packets transmitted by the multimedia server will be dropped. This metric measures the energy efficiency of the client-side WNIC in terms of the energy expended for the amount of useful data it has received.

Drop Rate: This is defined as the percentage of data dropped due to longer-than-actual predicted no-data interval lengths. The goal of our experiment is to minimize this metric as well.

IV. Experimental Results

We use the wireless traffic trace files obtained by the monitoring station to perform the simulation. A WNIC cannot receive or buffer data when it is in the *sleep* state. If the predicted *sleep* interval is shorter than the actual one, the client-side WNIC wakes up at the end of predicted sleep interval and transitions to an *idle* state, ready to receive the burst of data packets. If the predicted *sleep* interval is longer than the actual one, the client-side WNIC sleeps through the end of the estimated *sleep* interval. Depending on the time when the WNIC wakes up, part of or the entire burst of data following the actual no-data interval is considered to be lost. In our experiments, we use the *Wall* theatrical trailer that is digitized to a high quality video stream. The Microsoft Media, Real and Apple QuickTime streaming formats are each used for the wireless transmission of the stream. Experimental measurements of the energy metric and drop rate are made as the value of the negative bias B is systematically varied.

To describe the power consumption model used to calculate the client-side WNIC energy consumption, we use the simplified stream data transmission model shown in Fig. 2. The power consumption of the WNIC in each of the four states, i.e. *sleep*, *idle*, *receive* and *transmit* is denoted by P_{sleep}, P_{idle}, $P_{receive}$ and $P_{transmit}$ respectively. The predicted idle period is denoted by T_p, and the energy consumption of the WNIC is denoted by EC. Then the predicted *sleep* period falls in one of the following three cases, as shown in Fig. 4.

Case 1: $T_p \leq t_2 - t_1$, i.e. the predicted *sleep* period is shorter than or equal to the actual *sleep* period. During the time period T_p the WNIC's energy consumption is given by $T_p \times P_{sleep} = T_p \times 177$ mW. The WNIC then wakes up and persists in the *idle* state until time instant t_2. The WNIC's energy consumption during this period is given by $(t_2 - t_1 - T_p) \times P_{idle} = (t_2 - t_1 - T_p) \times 1319$ mW. During the time period from t_2 to t_3, the WNIC receives data packets resulting in energy consumption given by $(t_3 - t_2) \times P_{receive} = (t_3 - t_2) \times 1425$ mW. After having received the data burst, the WNIC goes back to the *sleep* state until the end of the next predicted *sleep* period. Hence the energy consumption in this case is given by

$$EC_1 = T_p \times P_{sleep} + (t_2 - t_1 - T_p) \times P_{idle} + (t_3 - t_2) \times P_{receive}. \tag{4.1}$$

Case 2: $T_p > (t_2 - t_1)$ and $T_p \leq (t_3 - t_1)$, i.e., the predicted *sleep* period is longer than the actual no-data period but shorter than or equal to the no-data period plus the data transmission period. In this case, during the predicted *sleep* period T_p, the WNIC energy consumption is given by $T_p \times P_{sleep} = T_p \times 177$ mW. Since the WNIC wakes up in the middle of data burst, part of the data in the burst is dropped. To receive the remainder of the data in the burst, the energy expended by the WNIC is given by $(t_3 - t_1 - T_p) \times P_{receive} = (t_3 - t_1 - T_p) \times 1425$ mW. After having finished receiving the data burst, the WNIC goes back to the *sleep* state until the end of the next predicted sleep period. The total energy consumption in this case is given by

$$EC_2 = T_p \times P_{sleep} + (t_3 - t_1 - T_p) \times P_{receive}. \tag{4.2}$$

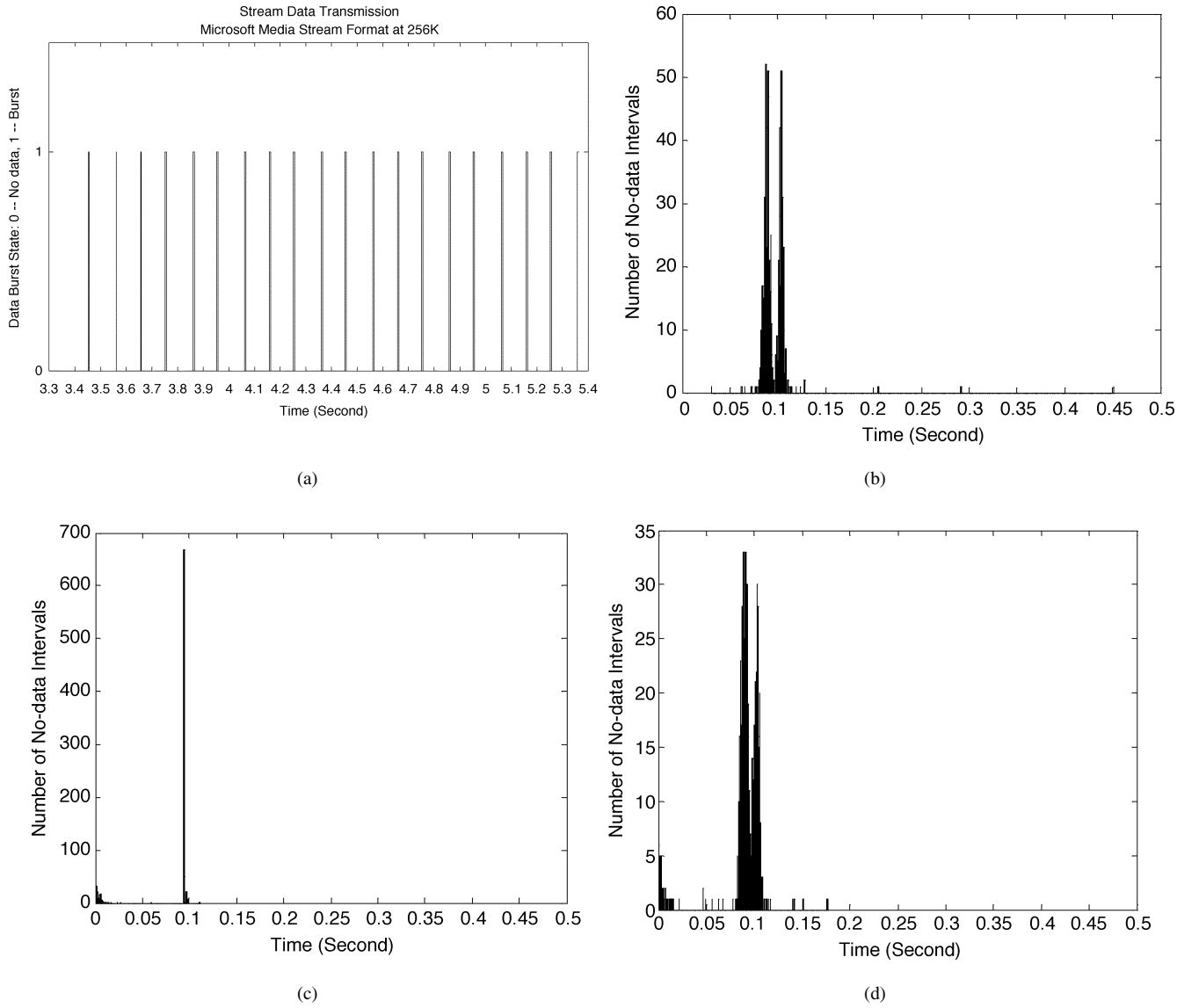

Fig. 5. (a) Data burst intervals for microsoft media streaming data at 256 Kbps. (b) No-data interval length histogram for actual data, Microsoft Media format at 256 Kbps. (c) No-data interval length histogram history-baseed estimation, Microsoft Media format at 256 Kbps. (d) No-data interval length histogram for linear prediction-based estimation, Microsoft Media format at 256 Kbps.

Case 3: $T_p > t_3 - t_1$, i.e. the predicted *sleep* period is too long. Consequently the WNIC wakes up during the next no-data interval and the data transmitted during the period $[t_2, t_3]$ is dropped. The WNIC persists in the *idle* state until the beginning of the following data burst. During the predicted *sleep* period T_p, the WNIC's energy consumption is given by $T_p \times P_{sleep} = T_p \times 177$ mW. The WNIC then persists in the *idle* state until the beginning of the next data burst. The energy consumption during this period is given by $(t_4 - t_1 - T_p) \times P_{idle} = (t_4 - t_1 - T_p) \times 1319$ mW. The energy consumption in this case is given by

$$EC_3 = T_p \times P_{sleep} + (t_4 - t_1 - T_p) \times P_{idle}. \tag{4.3}$$

We compare the WNIC energy consumption results obtained using the linear prediction-based approach and the history-based approach described in [1]. The results presented in this section are obtained when the *Wall* video segment is transmitted by the multimedia server in the Microsoft Media, Real, and Apple QuickTime streaming formats. The multimedia server transmission bandwidth is set to result in a streaming bandwidth of 256 Kbps and 512 Kbps. The nature of the data bursts resulting from each of the aforementioned streaming formats is depicted in Figs. 5(a), 6(a), and 7(a). As pointed out by Chandra [1], due to differences in the characteristics of the underlying media stream formats, the time series comprising of the lengths of the no-data intervals between successive data bursts exhibit different statistical properties. The Microsoft Media server transmits large data packets at fairly regular intervals. The Real and Apple QuickTime players tend to exhibit greater variation in the packet sizes and inter-packet arrival times. Figs. 5(b), 6(b), and 7(b) show the histograms of the no-data interval lengths for each of the above streaming formats. To compare the performance of the history-based and linear prediction-based approaches, the histograms of the predicted no-data interval lengths obtained using the history-based approach and the linear prediction-based approach are presented in Figs. 5(c), 6(c), and 7(c) and Figs. 5(d), 6(d), and 7(d), respectively, for each of the aforementioned streaming formats. A comparison of the distribution of the actual no-data interval lengths and the distributions of the predicted no-data interval lengths for each of the three streaming formats

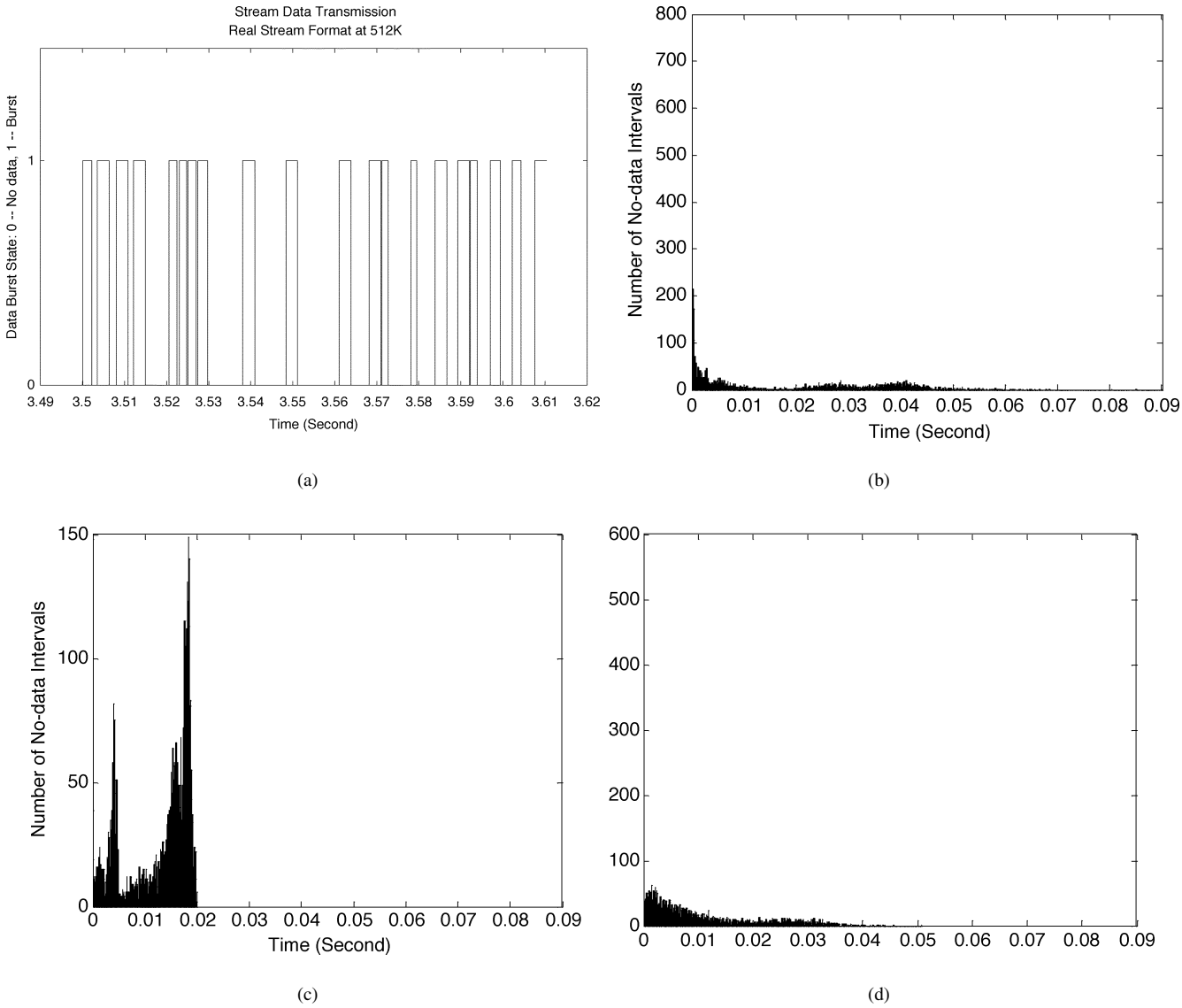

Fig. 6. (a) Data burst intervals for real streaming data at 512 Kbps. (b) No-data interval length histogram for actual data, real format at 512 Kbps. (c) No-data interval length histogram, history-based estimation, real format at 512 Kbps. (d) No-data interval length histogram, linear prediction-based estimation, real format at 512 Kbps.

shows that the linear prediction-based approach yields no-data interval length distributions that are much closer to the actual distributions when compared to the history-based approach. When the distribution of the no-data interval lengths spans a broad range, the history-based approach is observed to be incapable of preserving the statistical properties of the actual no-data intervals.

The values of the energy metric for the history-based and linear prediction-based approaches are compared for a given value of the drop rate. In order to obtain a fair comparison, the number of previous observations (number of previous actual no-data intervals) used in the estimation is set to be the same for both, the linear prediction-based approach and the history-based approach. The graphs in Figs. 8–10 plot the client-side WNIC total energy consumption versus the drop rate. The magnitude of the negative bias added to the value of the predicted *sleep* interval length is indicated near the corresponding data point in each figure. Experiment results show that the linear prediction-based approach yields a lower total WNIC energy consumption compared to the history-based approach, for a given value of the drop rate when the number of previous observations used is the same. Moreover, when no negative bias is added, the linear prediction-based approach always yields a better performance than the history-based approach in terms of the drop rate and the total WNIC energy consumption. This implies that, statistically speaking, the linear prediction-based approach estimates the *sleep* interval length values for the client-side WNIC more accurately than does the history-based approach.

When a WNIC is in the *sleep* state, it neither can receive nor buffer the incoming data. Consequently, if the estimated *sleep* interval length is longer than its actual value, the client-side WNIC persists in the *sleep* state when the data burst arrives. Accordingly, this burst of data is considered lost. It is useful to decrease the data drop rate with a properly chosen negative bias that is added to the estimated length of the no-data interval. For a fixed magnitude of negative bias, the more accurate the estimation, the higher the percentage of predicted *sleep* intervals that are shorter than their actual counterparts. From Figs. 8–10, we can see that for the same magnitude of negative bias that is added to the predicted *sleep* period, the linear prediction-based approach results in a lower data drop rate compared to the history-based approach. This can be attributed to the fact that the *sleep* interval length value estimated

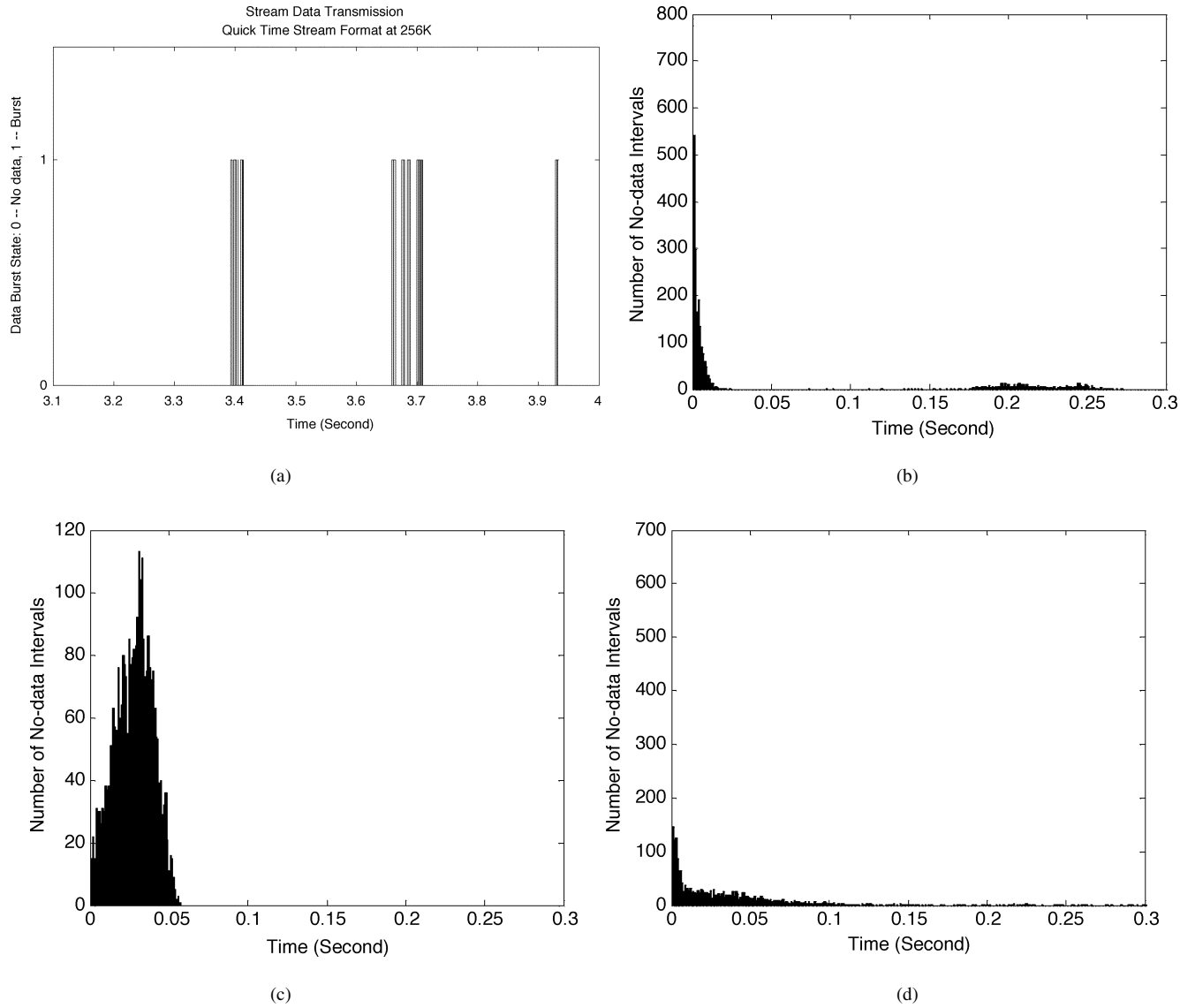

Fig. 7. (a) Data burst intervals for apple QuickTime streaming data at 256 Kbps. (b) No-data interval length histogram for actual data, apple QuickTime format at 256 Kbps. (c) No-data interval length histogram, history-based estimation, apple QuickTime format at 256 Kbps. (d) No-data interval length histogram, linear prediction-based estimation, apple QuickTime format at 256 Kbps.

by the linear prediction-based approach lies within a smaller neighborhood about the actual *sleep* interval length value. With the addition of a negative bias of relatively small magnitude, most of the estimated *sleep* intervals are observed to lie within their actual counterparts in the case of the linear prediction-based approach, thus resulting in a relatively low drop rate.

With no negative bias added to the predicted *sleep* interval length values, there is a higher probability that the predicted values of the *sleep* interval lengths are longer than their actual counterparts. Some of the predicted values are so long that entire data bursts are dropped by the client-side WNIC, as indicated by *Case 3* in Fig. 4. In this situation, the client-side WNIC wakes up during the no-data interval, and persists in the *idle* state until the following data burst. The WNIC energy consumption in this situation is defined by (4.3). The energy consumed by the client-side WNIC during its *idle* state is given by $(t_4 - t_1 - T_p) \times P_{idle} = (t_4 - t_1 - T_p) \times 1319\ \mathrm{mW}$ (Fig. 4). Since the power consumption of the WNIC in the *idle* state is higher than in the *sleep* state if the no-data intervals between data bursts are long, the energy consumption due to overestimation of the *sleep* interval is large. A small amount of negative bias applied to the predicted values of the *sleep* interval lengths can reduce the probability of overestimation as defined by *Case 3* (Fig. 4) and therefore reduce the client-side WNIC power consumption and also the drop rate. The Microsoft Media and the Apple QuickTime streaming formats are characterized by long no-data intervals accompanying relatively short data bursts. As shown in Fig. 8, in the case of the Microsoft Media format, when the bias magnitude lies in the range [0, 0.006], the total client-side energy consumption and the data drop rate exhibit a decreasing trend with increasing value of the bias magnitude for the history-based approach. As shown in Fig. 10, a similar trend can be observed in the case of the Apple QuickTime format when the bias magnitude values are in the range [0, 0.01] for the linear prediction-based approach that uses 18 previous observations.

When the bias magnitude value is chosen large enough such that all the predicted no-data interval length values comply with *Case 2* (Fig. 4), then increasing the bias magnitude value results in a greater amount of data received by the client-side WNIC (i.e., a lower drop rate) at the expense of increased energy consumption. This explains the

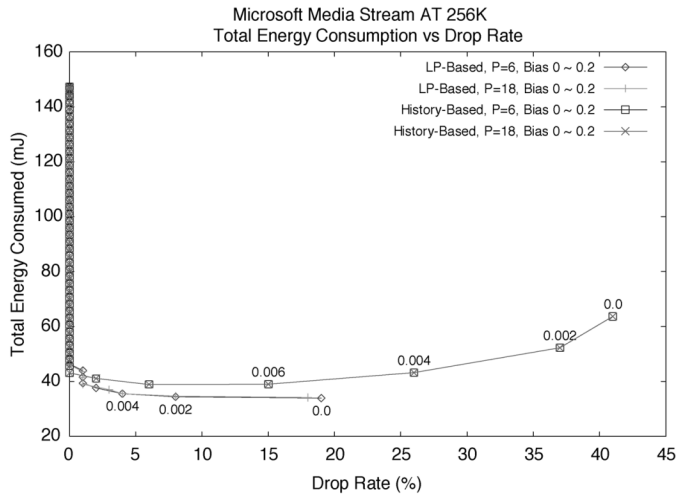

Fig. 8. Total WNIC energy consumption vs. drop rate, results of the linear prediction-based approach, and history-based approach, Microsoft Media format at 256 Kbps.

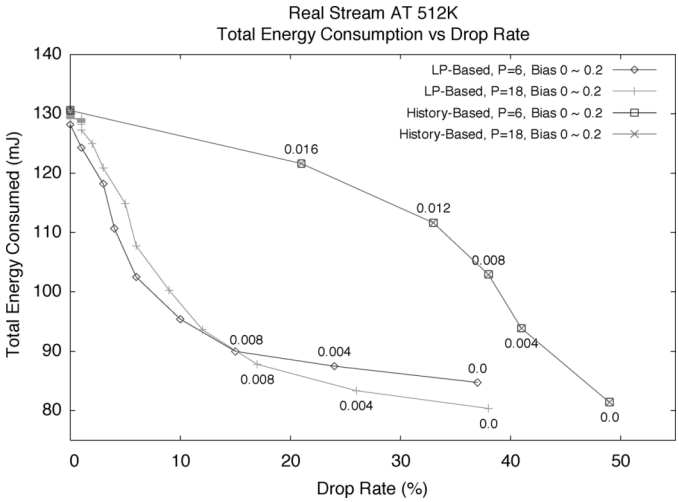

Fig. 9. Total WNIC energy consumption versus drop rate, results of the linear prediction-based, and history-based approach, real format at 512 Kbps.

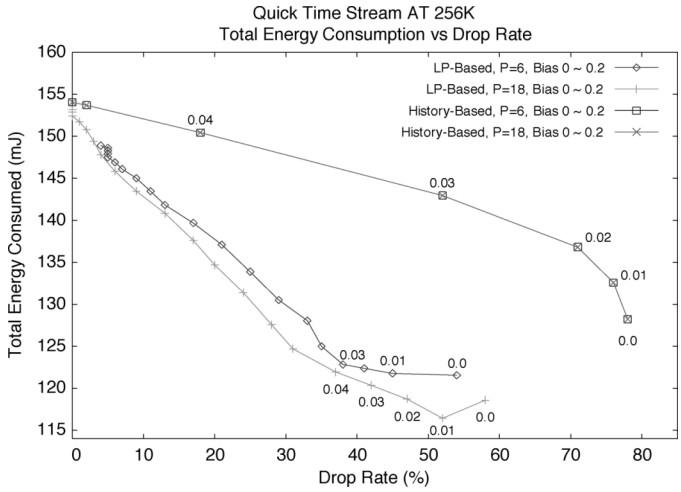

Fig. 10. Total WNIC energy consumption vs. drop rate, results of the linear prediction-based approach, and history-based approach, Apple QuickTime format at 256 Kbps.

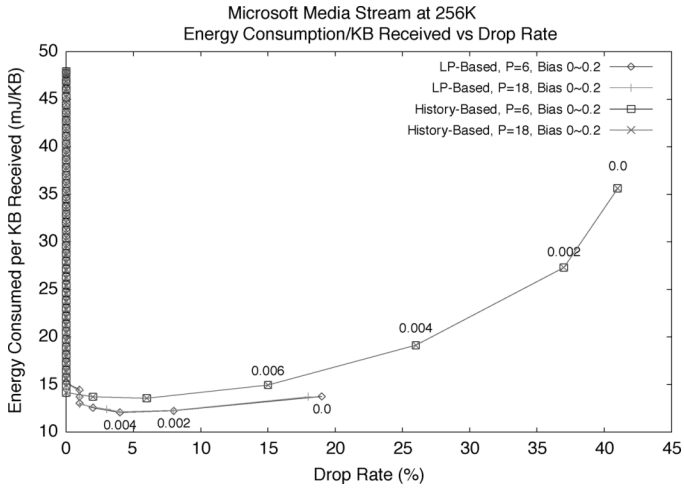

Fig. 11. Energy consumption per KByte of data received, results of history- and linear prediction-based approaches, Microsoft Media format at 256 Kbps.

trend where the drop rate decreases but the total WNIC energy consumption increases with increasing bias magnitude value. This trend can be observed in Fig. 9 in the case of the Real format for both, the history-based approach and the linear prediction-based approach. A similar trend can be observed in the case of the Apple QuickTime format (Fig. 10) for the history-based approach and for the linear prediction-based approach when the bias magnitude value is greater than 0.01. However, once the bias magnitude value crosses a certain threshold value such that all the predicted no-data interval length values comply with *Case 1* (Fig. 4) then any further increase in the bias magnitude value only increases the total WNIC energy consumption (since the client-side WNIC spends more time in the *idle* state waiting for the data burst to be received) without any decrease in the drop rate. This trend can be clearly observed in the case of the Microsoft Media format (Fig. 8), for the history-based approach when the bias magnitude value exceeds 0.12 and for the linear prediction-based approach when the bias magnitude value exceeds 0.01. In the limiting case when the bias magnitude value equals the longest actual no-data interval length, the client-side WNIC will be in the *idle* or *receive* state for the entire duration of the streaming session. This is tantamount to the complete absence of any client-side prediction scheme. In this case the energy metric values for both, the history-based approach and the linear prediction-based approach converge to same point, which corresponds to the absence of any client-side prediction whatsoever.

In order to compare the energy efficiency of the client-side WNIC when receiving video streams in different formats, the energy consumption of the client-side WNIC per KByte of data received is plotted as a function of the drop rate for all the three media streaming formats for both, the history-based approach and the linear prediction-based approach (Figs. 11–13). The graphs in Figs. 11–13 do not exhibit the same consistency as their counterparts in Figs. 8–10 in terms of monotonicity of the function. This can be explained with the following analysis. If the total energy consumption of the client-side WNIC, EC_{total} (which is plotted versus the drop rate in Figs. 8–10), is expressed as a function of the drop rate, as follows:

$$EC_{total} = f(drop\_rate) \tag{4.4}$$

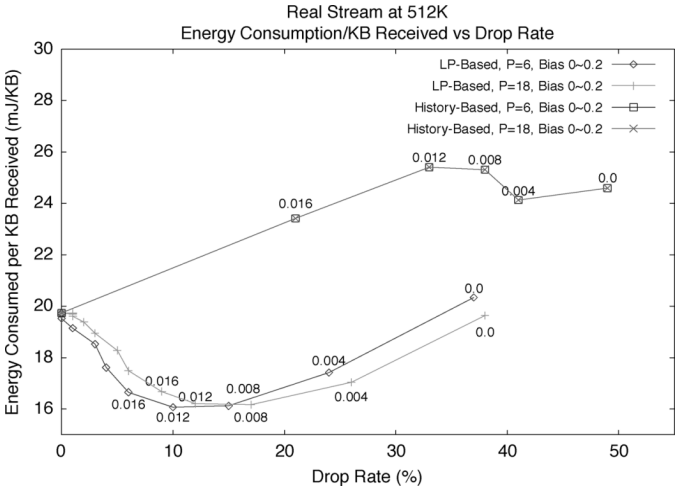

Fig. 12. Energy consumption per KByte of data received, results of history- and linear prediction-based approaches, real format at 512 Kbps.

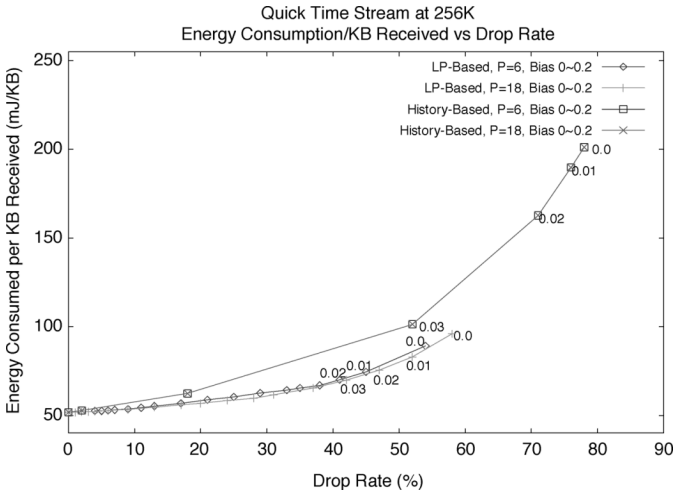

Fig. 13. Energy consumption per KByte of data received, results of history-& linear prediction-based approaches, Apple QuickTime format at 256 Kbps.

then the energy consumed by the client-side WNIC for each KByte of data received, denoted by EC_{KByte}, is given by

$$EC_{KByte} = \frac{f(drop\_rate)}{B \times (1 - drop\_rate)} \tag{4.5}$$

where B is the amount of data transmitted in the video stream. From (4.5), it can be observed that even if $f(drop\_rate)$ is monotonic with respect to $drop\_rate$, EC_{KByte} can still be non-monotonic with respect to $drop\_rate$. Nevertheless, the value of EC_{KByte} in the case of the linear prediction-based approach is observed to be much lower than that in the case of the history-based approach for a given value of $drop\_rate$ for all the three streaming media formats (Figs. 11–13). Conversely, the drop rate in the case of the linear prediction-based approach is observed to be much lower than that in the case of the history-based approach for a given value of EC_{KByte} for all the three streaming media formats (Figs. 11–13). This shows that the linear prediction-based approach is more energy efficient than the history-based approach regardless of the streaming media format.

V. Conclusions

The wireless network interface card (WNIC) of a mobile computing device accounts for a significant percentage of the overall client power consumption. In this paper, we have shown how linear prediction can be used to predict the length of the *sleep* time intervals for the client-side WNIC in order to reduce its energy consumption. The prediction model is trained using previously observed no-data intervals for a multimedia traffic stream. Experimental results show that, for a given value of additive (negative) bias, the statistical linear prediction-based approach yields, simultaneously, a lower data drop rate and a lower energy metric when compared to the history-based approach. In fact, the history-based approach can be looked upon as a special (i.e., degenerate) case of the linear prediction-based approach where all the predictor coefficients are identical in value.

Different popular multimedia streaming formats exhibit different data streaming characteristics. The Real and the Apple QuickTime media streams exhibit greater variation in the data packet sizes and inter-packet arrival times when compared to the Microsoft Media streams. This makes it hard for the prediction algorithm to reliably predict the lengths of no-data intervals. Nevertheless linear prediction-based approach is shown to be more robust than the history-based approach in its ability to predict the lengths of the no-data intervals, for all the three popular media stream formats explored in this work, namely Microsoft Media, Apple QuickTime, and Real.

Future research will investigate more sophisticated time series modeling and prediction methods. Problems scenarios where both the server and the client are power constrained (such as in a peer-to-peer *ad-hoc* mobile network) will also be investigated.

References

[1] S. Chandra, *Wireless Network Interface Energy Consumption Implications for Popular Streaming Formats, Multimedia Systems*. Berlin, Germany: Springer-Verlag, 2003, pp. 185–201.

[2] M. Stemm, P. Gauthier, D. Harada, and R. H. Katz, "Reducing power consumption of network interface in hand-held devices," in *Proc. 3rd Int. Workshop on Mobile Multimedia Communications (MoMuc-3)*, Princeton, NJ, Sep. 1996, pp. 103–112.

[3] E. Shih, V. Bahl, and M. Sinclair, *Reducing Energy Consumption of Wireless Mobile Devices Using a Secondary Low-Power Channel*. Cambridge, MA: MIT Lab. Comput. Sci., 2003, pp. 37–38.

[4] C. F. Chiasserini and R. R. Rao, "Pulsed battery discharge in communication devices," in *Proc. 5th ACM/IEEE Int. Conf. Mobile Computing and Networking (MOBICOM '99)*, Seattle, WA, Aug. 1999, pp. 88–95.

[5] J. Wilkes, Predictive Power Conservation Hewlett-Packard Labs, Tech. Rep. HPL-CSP-92-5, 1992.

[6] K. Li, R. Kumpf, P. Horton, and T. Anderson, "A quantitative analysis of disk drive power management in portable computers," in *Proc. USENIX Assoc. Winter Technical Conf.*, 1994, pp. 279–291.

[7] F. Douglis, P. Krishnan, and B. Bershad, "Adaptive disk spin down policies for mobile computers," in *Proc. 2nd USENIX Symp. Mobile and Location Independent Computing*, Monterey, CA, 1995, pp. 121–137.

[8] D. P. Helmbold, D. E. Long, and B. Sherrod, "A dynamic disk spin-down technique for mobile computing," in *Proc. 2nd ACM Int. Conf. Mobile Computing (MOBICOM96)*, 1996, pp. 130–142.

[9] M. Weiser, B. Welch, A. Demers, and S. Shenker, "Scheduling for reduced CPU energy," in *Proc. 1st Symp. Operating Systems Design and Implementation (OSDI)*, Monterey, CA, 1994, pp. 13–23.

[10] K. Govil, E. Chan, and H. Wasserman, "Comparing algorithms for dynamic speed-setting of a low-power CPU," in *Proc. 1st ACM Int. Conf. Mobile Computing and Networking (MOBICOM95)*, 1995, pp. 13–25.

[11] I. Imielinski, M. Gupta, and S. Peyyeti, "Energy efficient data filtering and communications in mobile wireless computing," in *Proc. USENIX Symp. Location Dependent Computing*, 1995, pp. 109–119.

[12] A. Datta, A. Celik, J. Kim, D. E. VanderMeer, and V. Kumar, "Adaptive broadcast protocols to support power conservant retrieval by mobile users," in *Proc. Data Engineering Conf. (ICDE)*, 1997, pp. 124–133.

[13] R. Kravets and P. Krishnan, "Power management techniques for mobile communication," in *Proc. 4th Int. Conf. Mobile Computing and Networking (MOBICOM98)*, 1998, pp. 157–168.

[14] S. Singh, M. Woo, and C. S. Raghavendra, "Power-aware routing in mobile ad hoc networks," in *Proc. 4th ACM/IEEE Int. Conf. Mobile Computing and Networking*, 1998, pp. 181–190.
[15] J. Lorch and A. J. Smith, "Software strategies for portable computer energy management," *IEEE Pers. Commun. Mag.*, pp. 60–7, Jun. 1998.
[16] P. J. M. Havinga, "Mobile Multimedia Systems," Ph.D. thesis, Univ. Twente, Enschede, The Netherlands, 2000.
[17] P. Agrawal, J. C. Chen, S. Kishore, P. Ramanathan, and K. Sivalingam, "Battery power sensitive video processing in wireless networks," in *Proc. IEEE PIMRC98*, Boston, MA, 1998, pp. 116–120.
[18] L. M. Feeney and M. Nilsson, "Investigating the energy consumption of a wireless network interface in an ad hoc networking environment," in *Proc. IEEE INFOCOM 2001*, Anchorage, AK, 2001, vol. 3, pp. 1548–1557.
[19] J. D. Hamilton, *Time Series Analysis*. Princeton, NJ: Princeton Univ. Press, 1994.
[20] J. Makhou, "Maximum entropy spectral analysis," in *Modern Spectrum Analysis*, D. G. Childers, Ed. New York: IEEE Press, 1978, pp. 34–48.

Interactivity-Loss Avoidance in Event Delivery Synchronization for Mirrored Game Architectures

Claudio E. Palazzi, Stefano Ferretti, Stefano Cacciaguerra, and Marco Roccetti

***Abstract*—Since the expansion of their market and their challenging requirements, Massively Multiplayer Online Games are gaining increasing attention in the scientific community. One of the key factors in this kind of application is represented by the ability to rapidly deliver game events among the various players over the network. Employing in this context Mirrored Game Server architectures and adapting RED (Random Early Detection) techniques borrowed from network queuing management, we are able to show sensible benefits in upholding interactivity and scalability, whilst preserving game state consistency and game evolution fluency at the player's side.**

***Index Terms*—Consistency, event delivery service, interactivity, massively multiplayer online game, online entertainment.**

I. Introduction

In the last decade, thanks to their impressive progression in plunging players into terrifically realistic and capturing virtual worlds, online games have expanded their market with a persistent and accelerating growth. Nowadays, Massively Multiplayer Online Games (MMOGs) are further extending the boundaries of what has been defined *the tenth art* with the possibility of simultaneously engaging millions of players located all over the world. Unfortunately, MMOG is a highly interactive application, characterized by strict real time requirements, harder than those accomplishable by traditional Internet protocols [1].

It is widely accepted that a suitable architecture able to efficiently manage large-scale distributed games may make use of a constellation of mirrored *Game State Servers* (GSSs), which are deployed over the network in a limited number [2]. GSSs cooperate to maintain replicas of the same game state view. Having multiple replica servers allows each client to connect in a client–server fashion to the *closest mirror*, thus reducing the communication latency. Each GSS assembles all game events of its engaged players, creates an updated game state, and then forwards it to all the other GSS peers. Prominent advantages of this approach are the absence of a single point of failure, having the networking complexity maintained by the servers, and the possibility to implement authentication. Nevertheless, synchronization schemes are still required to ensure the global consistency of the game state hold by the various servers.

Highly interactive applications as MMOGs are extremely sensitive to delays in event deliveries. In case of an intense traffic in the network or when excessive computational loads are slowing down some GSSs, the responsiveness of the system may be jeopardized. A proficient synchronization algorithm for MMOGs should be able to face both these two situations in order to preserve a high level of interactivity and an identical contemporary view of the game state among all the nodes in the system.

We present a novel synchronization mechanism able to uplift the playability degree of MMOGs by maintaining the event delivery delays under a human-perceptivity threshold. The core element at the basis of our scheme is the adoption of proactive queue management techniques [3] combined with the use of the semantics of the game [4]. In essence, events that can be considered *obsolete* since the arrival of *fresher* ones may be discarded at a given GSS utilizing a dropping probability which depends on the perceived responsiveness at that GSS. Limiting the number of game events in the system provides two beneficial results: it alleviates the processing burden and reduces the impact of network latency on playability. As a consequence, the game event delivery time results minimized.

The remainder of this paper is organized as follows. Section II reviews principal works in the field of online game interactivity. In Section III, we discuss some design issues at the basis of our work and in Section IV we present our novel scheme. The simulative environment and the corresponding results are provided, respectively, in Sections V and VI. Finally, Section VII concludes the paper.

II. Related Work

Several research works have already brought contributions to the factual development of efficient synchronization schemes. *Compression* and *aggregation* consider networking having a dominant position when dealing with delays and thus with the playability of a MMOG [5]. These schemes, however, pay the achieved latency benefits with increased computational costs. Moreover, aggregation wastes time as the transmission of available events is delayed till the point when other events arrive that can be aggregated.

In the attempt to reduce both the traffic load in the network and the computational cost to process each game event, *Interest Management* techniques have been devised for games where generated events are relevant only for a small fraction of users [6]. A tradeoff relationship exists between the amount of computation spared at the destination by receiving only a limited number of events and that one expended at the sending GSS to implement the filtering scheme.

Slightly detaching playability from the real responsiveness of the network, optimistic algorithms can be utilized to avoid delay perception. In case of lousy interactivity among GSSs, in fact, an optimistic approach executes events before really knowing if ordering would require processing other on-the-way events first. Game instances are thus processed without waiting for other possibly coming packets. *Rollback* based techniques are exploited to reestablish the consistency of the

Manuscript received November 4, 2004; revised October 7, 2005. This work was supported by the Italian MIUR via the Interlink Project The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Zhengyou Zhang.

C. E. Palazzi is with the Dipartimento di Scienze dell'Informazione, Università di Bologna, 40127 Bologna, Italia, and also with the Computer Science Department, University of California, Los Angeles, CA 90095 USA (e-mail: cpalazzi@cs.unibo.it).

S. Ferretti, S. Cacciaguerra, and M. Roccetti are with the Dipartimento di Scienze dell'Informazione, Università di Bologna, 40127 Bologna, Italia (e-mail: sferrett@cs.unibo.it; scacciag@cs.unibo.it; roccetti@cs.unibo.it).

Digital Object Identifier 10.1109/TMM.2006.876229