Prediction-Based Recovery from Link Outages in On-Board Mobile Communication Networks

Adeel Baig[*,†] Mahbub Hassan[*,†] Lavy Libman[†]

[*]School of Computer Science and Engineering
University of New South Wales
Sydney, NSW 2052, Australia
{abaig,mahbub}@cse.unsw.edu.au

[†]National ICT Australia
Bay 15, Australian Technology Park
Eveleigh, NSW 1430, Australia
Lavy.Libman@nicta.com.au

***Abstract*— In on-board mobile networks, such as those proposed for (and employed in) public transport vehicles, users are connected to a local network that attaches to the Internet via a mobile router and a wireless link. Central and coordinated management of mobility in a single router, rather than by each user device individually, has numerous advantages; however, it also means that link outages, e.g. due to signal degradation or handoff failure, may have an immediate impact on a potentially large number of connections. We argue that the advance knowledge of public transport routes, and their repetitive nature, allows a certain degree of *prediction* of impending link outages, which can be used to offset their catastrophic impact. Focusing on the TCP protocol and its extension known as *Freeze-TCP*, we study how the performance of the protocol depends on the outage prediction probability. In particular, we propose a Markov model of Freeze-TCP and, using simulations, show that it accurately predicts the performance improvement gained by outage prediction.**

I. Introduction

There is a growing interest in introducing broadband Internet services to public transport passengers by deploying high-speed local area networks (LANs) on-board public transport vehicles. These on-board LANs are connected to the Internet via on-board mobile routers (MR). Passengers simply connect their devices to the on-board LAN and start enjoying Internet services. Mobility of the entire LAN is managed transparently by the MR, which connects via a wireless link to an outside router using an extension of Mobile IP [1]. Thus, a typical on-board mobility architecture (Figure 1) consists of three main components: a high-speed mobile LAN (MLAN), mobile router, and mobile wireless Internet connection (MWIC) [2]. The MLAN provides a local high-speed connectivity to the on-board passengers (as well as devices integrated in the vehicle). The MR facilitates communication between the MLAN and the global communication infrastructure (Internet). It may also be equipped with such features as a data server and query manager to enable proxying/caching of on-board user requests. The MWIC connects the MR to the outside world, using either a land-based (cellular) wireless station or a satellite (such as Inmarsat's Swift64 mobile packet data service [3]). This architecture is becoming popular, as is evidenced by a growing number of commercial systems and research projects [3]–[9].

A critical issue in a mobile networking architecture is the possibility of temporary outages in the wireless link that connects the MR to the Internet. An outage, i.e. a period during which all packets transmitted by the MR are lost or corrupted, can be caused by various factors, such as loss of signal (e.g.

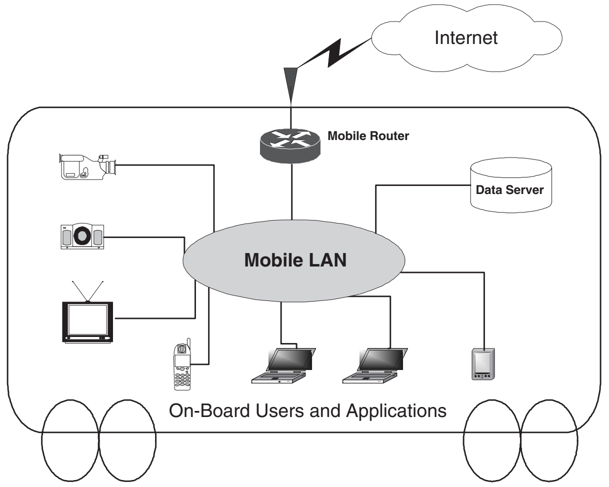

Fig. 1. On-board communication architecture.

when the vehicle passes through a tunnel or high-rise building blocking the line-of-sight), interference, or handoff failure between neighboring base stations. Obviously, a link outage disrupts existing connections and on-going services. In mobile networks, the problem is exacerbated by the presence of many users, which means that a single link outage may impact a potentially large number of existing connections. Recovering from such unexpected disruptions can be slow and costly.

However, unlike most wireless end devices (e.g. mobile phones), whose mobility behavior is generally unpredictable, the routes of public transport vehicles are known in advance and repetitive. This important feature can be used to *predict* link outages to a certain extent. For example, the MR may record such information as signal strength and available bandwidth on the wireless link, at various locations and times on its route. Analysis of the recorded information over time may reveal, for instance, that signal loss occurs with a high probability at certain locations (e.g. a tunnel), times-of-day (e.g. rush-hour interference), or weather conditions. While nothing can be done about the wireless link outage itself, the MR can send an advance signal to existing connection endpoints before the link goes down (and when it comes up again). This helps avoid the chaotic and uncoordinated attempts by the individual connections to resume their data flows, which may be subject to contention, excessively long timeouts, and other undesirable features. Indeed, our focus in this paper is not on the specific techniques for outage prediction, which remain a subject of ongoing research. We merely assume that outages can be predicted sufficiently in advance with a certain probability; our goal, then, is to propose a proactive mechanism to take advantage of this ability and explore the quantitative relation between the prediction quality and network performance.

In this paper, our particular focus is on TCP, the ubiquitous transport control protocol of the Internet. It is generally accepted that, in its classic form, TCP is ill-suited for wireless communications: it takes any packet loss to be an indication of congestion (a reasonable assumption for the majority of fixed communication links), whereas losses in wireless links are mainly caused by random bit errors, e.g. due to signal fading. With the growing acceptance of wireless and mobile computing, the study of techniques to improve TCP performance in wireless environments has become an active area of research. Some of the proposals to that end include Snoop [10], where the base station can make local retransmissions to the mobile host; Indirect TCP [11], which splits the communication into two separate TCP connections, between the mobile host to its base station and from there to the remote (fixed) endpoint; M-TCP [12], where the base station "chokes" the remote sender by advertising a small window size; and WTCP [13], which uses rate-based (rather than window-based) flow control altogether. All of the above methods, however, require significant support from the wireless base station, and are therefore unsuitable for our on-board mobile networking scenario. By contrast, Freeze-TCP, recently proposed in [14], which uses zero-window advertisements by the end hosts themselves, is a fully end-to-end approach and therefore well-suited to our context of outage prediction. Consequently, our study focuses particularly on the performance of Freeze-TCP in the presence of outage prediction in the mobile router.

Our contribution is twofold. First, we describe a Markovian model for the behavior of Freeze-TCP sources in the presence of partially predicted link outages. We then apply this model to explore the dependence of the Freeze-TCP throughput on such parameters as the frequency and duration of outages and their prediction probability. In particular, our results demonstrate that, for a given pattern of outages, there is a monotonic dependence between the average throughput attained by Freeze-TCP and the prediction probability, which becomes more convex as the maximum TCP window size increases and/or the link becomes more unstable. Our model predictions are also backed by *ns-2* simulations.

The rest of the paper is organised as follows. Section II provides a brief overview of the Freeze-TCP mechanism. The Markov model of Freeze-TCP is described in Section III. Section IV discusses the results obtained by applying the model in several scenarios, and compares them with actual simulation results. Section V concludes the paper and presents some possible directions for further study.

II. Overview of Freeze-TCP

Before proceeding with our analysis, we briefly recall the operation of the Freeze-TCP protocol; for further details see [14]. Freeze-TCP was designed to prevent undesirable timeouts (which lead to unnecessary slow-start and congestion avoidance phases of an already active connection) when a mobile host is handed off to a new wireless base station. To achieve that, once notified of the impending handoff by the MAC layer, the mobile host transmits an acknowledgment carrying a zero window advertisement (ZWA). Upon receiving the ZWA, the remote (fixed) host freezes all data transmission and enters a zero window probing mode, during which it attempts sending one-byte probes at intervals specified by its persistence timer, until the mobile host is reconnected. In addition, once the mobile host is reconnected at the new base station, it transmits a triple acknowledgment (TR-ACK) with a non-zero window advertisement (NZWA), enabling the remote host to resume its data transmission.[1]

It is important to observe that Freeze-TCP does not require any modifications whatsoever in the remote (fixed) host; the latter is merely expected to proceed as in normal TCP upon receiving a zero window acknowledgment. No changes are called for in the base stations either. Therefore, Freeze-TCP can be naturally adapted to our mobile networking context, with the only exception being that the ZWA/NZWA advertisements are initiated by the mobile router (MR), rather than the mobile host itself. In terms of implementation, this only requires the MR to keep track of the active TCP connections of the on-board users, and insert appropriate ZWA and NZWA packets upon prediction of an outage and after the outage ends, respectively. Consequently, the performance enhancement can be achieved without any changes in existing user devices.[2]

III. Markov Model for Freeze-TCP

Our experiments are based on a Markov chain model, which captures the behavior of both the TCP sender and the link. For simplicity, we follow a exponential two-state model for the wireless link. Under this model, the link is assumed to alternate between an 'up' state, in which it is error-free, and a 'down' state, in which no communication is possible. The time spent in every state is exponentially distributed. This model was first described by Gilbert [15] and is commonly used in performance analysis over wireless links (e.g., some recent examples include [16], [17]).

We now proceed to describe the chain structure in detail. A state is defined by a triplet $\langle W, C_{th}, f\rangle$, where W is the size of the last successfully transmitted window, C_{th} is the current congestion window threshold between slow-start and congestion avoidance phases, and f is a boolean value denoting "freezing". Every state must have $0 \leq W \leq C_{th} \leq W_m$,

[1]The purpose of the triple acknowledgment is to make the remote host perform a Fast-Retransmit of packets possibly sent (and lost) before the ZWA was received, and, thus, avoid the undesirable consequences of a timeout.

[2]Alternatively, the MR can merely broadcast a signal notifying all on-board users about the imminent outage, and another signal after the outage is over, and leave the Freeze-TCP mechanics to the discretion of the end hosts. This alternative avoids the need to track all active TCP connections at the MR and interfere with their end-to-end semantics, but requires the individual on-board devices to implement Freeze-TCP, and, in particular, be able to respond to such signals. In terms of performance, the difference between the alternatives is merely an extra delay on the on-board LAN, which, in practice, is likely to be negligible.

where W_m is an external parameter of the model, denoting the maximum TCP window size (in packets).[3] States with $W < C_{th}$ model TCP's 'slow-start' phase, whereas $W = C_{th}$ correspond to the 'congestion avoidance' phase. A value of $f = 0$ corresponds to normal operation, whereas $f = 1$ is a 'frozen' (persistent) state, which the protocol enters upon reception of a ZWA packet and leaves upon reception of a NZWA packet.

The states in our model capture the internal status of TCP with a granularity of a window; that is, the transitions among the states do not occur after every packet. This reduces the total number of states considerably, thereby facilitating the subsequent analysis. There is an approximation involved with this approach, however, in that the effect of a link failure in the middle of a window transmission cannot be captured with precision. Instead, below we take a conservative view, deeming any link failure to have occured at the start of the corresponding window transmission.

There are four kinds of transitions possible among the states; these are outlined below. The first three types are from regular ($f = 0$) states; the last kind applies to 'frozen' ($f = 1$) states.

1) If everything is normal, and the link stays up during the round-trip time, TCP advances normally. Thus, from any state in the slow-start phase ($\langle W, C_{th}, 0\rangle$ where $0 < W < C_{th}$), it will advance to $\langle \min(2W, C_{th}), C_{th}, 0\rangle$. From $0 = W < C_{th}$ (a slow-start state just after a timeout in the previous window), it will proceed to $\langle W = 1, C_{th}, 0\rangle$. Finally, from a congestion-avoidance state ($\langle W, C_{th}, 0\rangle$ where $W = C_{th}$), the transition is to $\langle W + 1, C_{th} + 1, 0\rangle$, unless $W = W_m$, in which case it stays in the same state (the window is already at the maximum size).
2) If the link goes down and its failure is not predicted in time, a timeout occurs and TCP sets the congestion window threshold value to be half the current window size, before restarting in a slow-start phase. This corresponds to a transition from $\langle W, C_{th}, 0\rangle$ to $\langle 0, \lfloor W/2\rfloor, 0\rangle$.
3) If the link goes down but its failure is predicted in time, TCP freezes its current state instead of falling back into slow-start. This corresponds to a transition from $\langle W, C_{th}, 0\rangle$ to $\langle W, C_{th}, 1\rangle$, for all states where $W \neq 0$. Observe that 'freezing' is not possible from a state with $W = 0$, as that means that a timeout has already occured.
4) From a 'frozen' state ($\langle W, C_{th}, 1\rangle$), there is a transition to the corresponding regular state $\langle W, C_{th}, 0\rangle$ when the link resumes operation.

The corresponding transition rates depend on the transition type as well as the state from which they occur; indeed, states with $W = 0$, which denote that a timeout has just been suffered, require special consideration. The transition rates are detailed below.

- For all transitions of type 1 occuring from a state with $W > 0$, the rate is $\frac{1}{\max(W\cdot t_x, T+t_x)}$, where T is the connection's round-trip time, t_x is the transmission time of a single packet, and W is the window size at the state *to* which the transition is made.[4]
- For all transitions of type 1 from a state with $W = 0$ (i.e. after the link is down), the rate is the reciprocal of the average link 'down' time.
- For all transitions of type 2 occuring from a state with $W > 0$, the rate is the misprediction probability (i.e. one minus the prediction probability) times the reciprocal of the average link 'up' time.
- For all transitions of type 2 from a state with $W = 0$ (all of which are to the state $\langle 0, 0, 0\rangle$), the rate is $\frac{1}{2T+t_x}$. Indeed, if the link is down during a second consecutive timeout (which is twice the round-trip time, due to Karn's exponential backoff), TCP will eventually resume operation in a congestion avoidance mode.
- For all transitions of type 3 (which, by definition, are from states with $W > 0$), the rate is the prediction probability times the reciprocal of the average link 'up' time.
- For all transitions of type 4, the rate is the reciprocal of the average link 'down' time.

Observe that, as per the transitions described above, many of the states are unreachable. These include, for example, all slow-start states where W is not 0 or a whole power of 2, or where $C_{th} > W_m/2$, as well as all 'frozen' states with $W = 0$. Furthermore, many functional equivalences among states can be noted; for instance, it makes no difference whether C_{th} is 0, 1, 2, or 3. Therefore, the analysis of the chain actually involves much fewer states than may seem at first.

Figure 2 presents a diagram of the functional part of the chain, with the exception of the 'frozen' states and transitions involving them (of types 3 and 4). Thus, it is to be kept in mind that, for any state in the diagram with $W > 0$, there is a corresponding 'frozen' state, with transitions back and forth between them only (at the rates outlined above); there are no other transitions within the 'frozen' states.

[3]We assume that the maximum window size is too small to cause congestion on the connection's path; therefore, TCP will not back off before reaching this window size.

[4]Recall that the interpretation of W is the size of the last window successfully transmitted; hence, the number of packets transmitted in a transition between states is given by W at the transition's destination state.

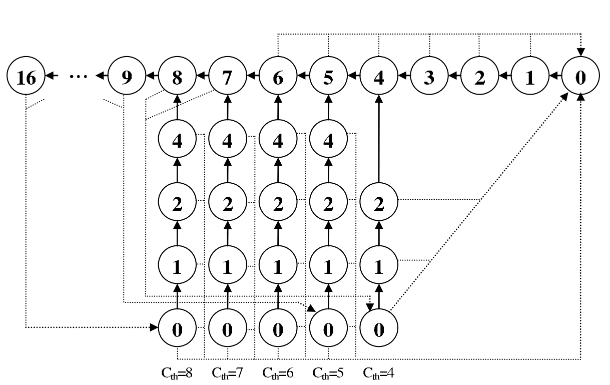

Fig. 2. The Freeze-TCP Markov chain model (excluding the 'frozen' states), for $W_m = 16$. The solid arrows correspond to transitions of type 1; the dotted arrows are transitions of type 2. Transitions of types 3 and 4 (to and from the 'frozen' states) are not shown.

The model is solved by finding the steady-state probabilities of the chain states (this involves merely an inversion of the transition rate matrix), and, subsequently, using them to compute the expected throughput. The protocol throughput in a state $\langle W, C_{th}, 0\rangle$ is $\frac{W}{T+t_x}$ packets per time unit, unless $W \cdot t_x > T$, in which case the throughput is limited by $1/t_x$. Obviously, the throughput in all the $\langle W, C_{th}, 1\rangle$ ('frozen') states is 0. Averaging the throughputs according to the steady-state probabilities yields the throughput performance of the protocol.

Our Markov TCP model bears a certain similarity to the one introduced in [18] for TCP Reno, and later extended in [19] for other TCP flavours (SACK and Vegas). The most obvious difference, and indeed the motivation behind this work, is the inclusion of 'frozen' states, to model the behavior of Freeze-TCP. Another important difference follows from our focus on the wireless link, whose performance varies due to physical factors only, and is independent of the source's behavior (as opposed to congestion). Accordingly, there is only one chain in our model, which takes into account both the source and the link (unlike the framework of [18], [19], which makes the interaction between the two explicit by separating the respective models). Furthermore, since (following [15]) we assume the link to be error-free when up, random packet losses cannot occur and, therefore, there are no states to capture the "fast recovery" mode of TCP. Finally, a subtle difference is that we provide separate states for all values of C_{th} up to $W_m/2$, and not just those which are whole powers of 2, as in [18], [19]. This makes the agreement between our model and simulations somewhat more precise, for only a modest cost in complexity.

IV. Analysis and Simulation Results

To verify the validity of the Freeze-TCP Markov model, we have implemented a simulation platform, using the *ns-2* simulator [20]. For that purpose, we modified the code of the TCP sink so as to enable it to send explicit ZWA and NZWA packets to the source. The times at which these advertisements are sent are taken from a pre-generated file that lists the time instants when the link goes 'up' or 'down' (sampled from an exponential distribution), and whether the 'down' state is predicted. A ZWA is sent half a round-trip time before the outage (to enable the advertisement reach the other endpoint before the link goes down), whereas a NZWA is sent as soon as the outage is over.

We have run the simulations and compared them to the Markov model solution for a variety of scenarios; however, due to space limits, we only present a selection of representative cases here (Figure 3). All these cases involve a simple topology of two nodes connected by a 1Mbps link, engaged in a long FTP transfer using packets of 1000 bytes. The one-way delay in each direction was taken to be 1.5 seconds (for a round-trip delay of 3 seconds). In the top figure, the link is taken to have an average 'up' time of 150 seconds ($50 \cdot RTT$) and an average 'down' time of 30 seconds ($10 \cdot RTT$). The bottom figure plots the results for a less stable link, with an 'up' time average of only 30 seconds ($10 \cdot RTT$) and 'down' average of 12 seconds ($4 \cdot RTT$).[5] In both cases, we use maximum TCP window sizes of 32 and 128 packets, and a probability of successful prediction that is varied between 0 and 100%. The plots show the TCP throughput, as predicted by the respective Markov model solution, as solid curves; the throughput achieved in the actual simulation runs is shown with 'x'-marks.

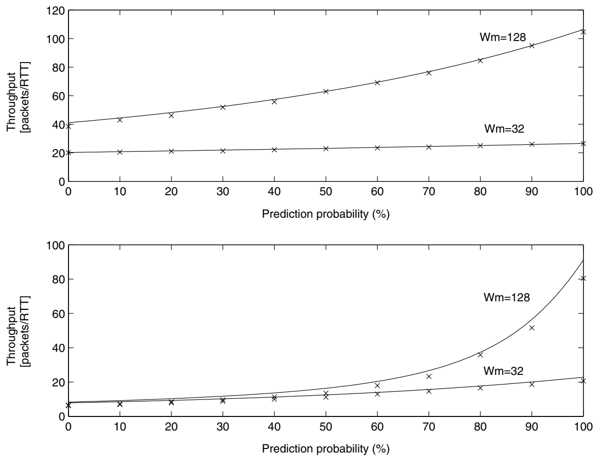

Fig. 3. Average throughput: 'up' time of $50 \cdot RTT$ and 'down' of $10 \cdot RTT$ (top); 'up' time of $10 \cdot RTT$ and 'down' of $4 \cdot RTT$ (bottom).

We observe that our model achieves a good accuracy in predicting the Freeze-TCP throughput. The fact that the predicted throughput is consistently higher, by a slight margin, than the value actually attained, can be explained by observing that, in our model (and similarly in [18], [19]), the TCP window in the congestion-avoidance phase is assumed to grow by one packet per round-trip period, whereas in reality, TCP increases the window size by the reciprocal of its current value upon reception of every acknowledgment. This effect causes the window to grow slightly more slowly (e.g., from a window of 2 packets, reception of two acknowledgments causes it to grow to $2 + \frac{1}{2} + \frac{1}{2.5} = 2.9$ after a round-trip time, which is slightly less than 3); thus, it takes a slightly higher number of RTT periods than predicted by the model to attain the maximum window size.

Another important conclusion from the plots is that an investment in a good prediction method, correctly predicting a high percentage of outages, becomes more worthwhile as the window sizes are larger or the link is less stable. Indeed, we observe that for $W_m = 32$ the lines are nearly linear, whereas for $W_m = 128$ they resemble quadratic curves; furthermore, the curves in the bottom part of Figure 3 are more convex than in the top part. This phenomenon can be explained as follows. The time it takes TCP to complete the slow-start and congestion avoidance phases is constant. Therefore, if most of the outages happen after TCP has already reached the maximum window size, then each unpredicted outage inflicts a

[5]These values were selected to illustrate the maximum benefits from prediction. Obviously, if the average 'up' time is higher than the RTT by many orders of magnitude, outage prediction becomes insignificant, since the slow-start and congestion avoidance phases take a negligible portion of time anyway. Conversely, if the link is so unstable that it frequently changes states within a single RTT, prediction is hardly possible since even the ZWA and NZWA notifications cannot be properly communicated.

fixed penalty on the TCP performance, namely, the difference between TCP's maximum throughput and that achieved during the slow-start and congestion avoidance phases. In that case, the throughput simply increases linearly in the percentage of outages that are predicted. However, if most of the outages happen while TCP is still in the congestion avoidance stage, then each additional correct prediction has an increasing contribution to the throughput, as it happens at a higher window size. The dependence of the throughput on the percentage of correctly predicted outages then becomes quadratic. This effect is graphically illustrated in Figure 4. Consequently, the more likely outages are to happen during the congestion avoidance phase – in other words, the larger the maximum window size and/or the less stable the link – the more convex is the throughput dependence on the prediction probability.

V. Conclusion

Integration of mobile communication infrastructure in public transport vehicles is recently becoming an important trend in wireless networking. In this paper, we have explored the possibility of improving on-board network performance through link outage predictions, which can be achieved by exploiting the predefined and repetitive routes of public transport vehicles. Specifically, we have advocated the use of the Freeze-TCP extension for on-board mobile networks. We proposed a Markovian model of Freeze-TCP to analyze the benefits of such outage predictions on TCP throughput. Using the model (and backing it by simulation), we have demonstrated that an investment in a good outage prediction technology, correctly predicting a high percentage of link outages, becomes increasingly worthwhile as outages occur more frequently and/or the TCP connections use larger windows.

For our analysis, we assumed a wireless link that, except during outages, suffers from neither random bit errors nor congestion. This enabled us to focus particularly on the extent that TCP performance is affected by the outage prediction quality. Extending the analysis, and in particular the Markov model of Freeze-TCP, so as to account for congestion and random losses, is the subject of ongoing work.

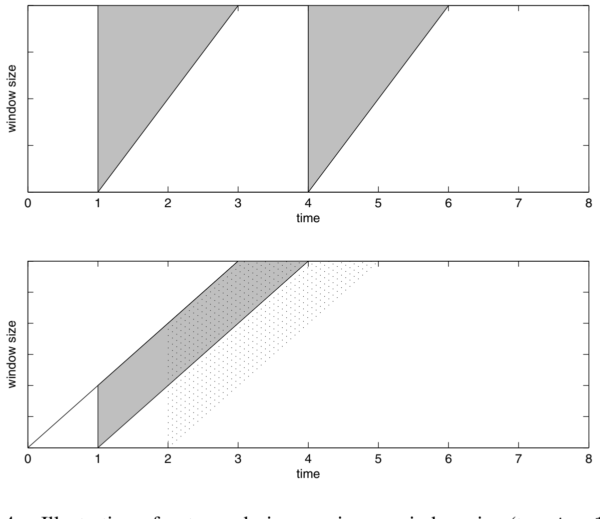

Fig. 4. Illustration of outages during maximum window size (top, $t = 1, 4$) vs. outages during congestion avoidance phase (bottom, $t = 1, 2$). The shaded areas represent the difference (in transmitted packets) between a predicted and an unpredicted outage. In the bottom figure, misprediction of the outage at $t = 2$ (dashed area) loses more than that at $t = 1$ (solid area).

In this paper, we have not made any assumptions about the particular method used to implement outage prediction in the mobile router, and only assumed that every outage can be independently predicted with a certain probability. Further research in this area is called for; in particular, a practical exploration of wireless connectivity (i.e. signal strength, bandwidth availability, *etc*) on public transport routes, and its dependence on such parameters as location and time-of-day, would make it possible to compare the merits of possible outage prediction implementations. This direction is currently being pursued within the eMOTION project at the University of New South Wales [21].

References

[1] V. Devarapalli, R. Wakikawa, A. Petrescu, and P. Thubert. Network mobility (NEMO) basic support protocol, June 2004. Internet draft: draft-ietf-nemo-basic-support-03.txt (work in progress).

[2] K-D. Lin and J-F. Chang. Communications and entertainment onboard a high-speed public transport system. *IEEE Wireless Communications*, 9(1):84–89, February 2002.

[3] Inmarsat's Swift64: http://www.inmarsat.com/swift64.

[4] European commission IST OverDRiVE: http://www.ist-overdrive.org.

[5] Nautilus 6 working group: http://www.nautilus6.org.

[6] icomera: http://www.icomera.com.

[7] PointShot Wireless: http://www.pointshotwireless.com.

[8] 21Net: http://www.21net.com.

[9] Connexion by Boeing: http://www.connexionbyboeing.com.

[10] H. Balakrishnan, S. Seshan, E. Amir, and R.H. Katz. Improving TCP/IP performance over wireless networks. In *Proc. ACM MobiCom*, Berkeley, CA, November 1995.

[11] A.V. Bakre and B.R. Badrinath. I-TCP: Indirect TCP for mobile hosts. In *Proc. IEEE International Conference on Distributed Computing Systems (ICDCS)*, pages 136–143, Vancouver, BC, Canada, May 1995.

[12] K. Brown and S. Singh. M-TCP: TCP for mobile cellular networks. *ACM SIGCOMM Computer Communication Review*, 27(5):19–43, October 1997.

[13] P. Sinha, N. Venkitaraman, R. Sivakumar, and V. Bharghavan. WTCP: A reliable transport protocol for wireless wide-area networks. In *Proc. ACM MobiCom*, Seattle, WA, August 1999.

[14] T. Goff, J. Moronski, D.S. Phatak, and V. Gupta. Freeze-TCP: A true end-to-end TCP enhancement mechanism for mobile environments. In *Proc. IEEE Infocom*, pages 1537–1545, Tel-Aviv, Israel, March 2000.

[15] E. Gilbert. Capacity of a burst noise channel. *Bell Systems Technical Journal*, 39:1253–1256, 1960.

[16] M. Miyoshi, M. Sugano, and M. Murata. Performance improvement of TCP on wireless cellular networks by adaptive FEC combined with explicit loss notification. In *Proc. IEEE Vehicular Technology Conference (VTC)*, pages 982–986, Birmingham, AL, May 2002.

[17] Z. Hadzi-Velkov and B. Spasenovski. Performance comparison of IEEE 802.11 and ETSI HIPERLAN type 1 under influence of burst noise channel. In *Proc. IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1415–1420, Chicago, IL, September 2000.

[18] C. Casetti and M. Meo. An analytical framework for the performance evaluation of TCP Reno connections. *Computer Networks*, 37(5):669–682, November 2001.

[19] A. Wierman, T. Osogami, and J. Olsén. A unified framework for modeling TCP-Vegas, TCP-SACK, and TCP-Reno. In *Proc. IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, Orlando, FL, October 2003.

[20] The network simulator — ns-2: http://www.isi.edu/nsnam/ns.

[21] Project eMOTION — network in motion: http://ocean.cse.unsw.edu.au/emotion.