Cross-Layer Adaptation for TCP-Based Applications in WCDMA Systems

Inés Cabrera Molero[‡], Niels Möller[‡], Justus Petersson[*], Robert Skog[*],
Åke Arvidsson[†], Oscar Flärdh[‡], and Karl H. Johansson[‡]

[‡]Department of Signals, Sensors and Systems
Royal Institute of Technology
SE-100 44 Stockholm, Sweden
inescm@kth.se, {niels,oscarf,kallej}@s3.kth.se

[*]Ericsson AB
Kistagången 26, SE-164 80 Stockholm, Sweden
{Justus.Petersson,Robert.Skog}@ericsson.com

[†]Ericsson AB
Box 1505, SE-125 25 Älvsjö, Sweden
Ake.Arvidsson@ericsson.com

***Abstract*— In this paper, we consider TCP-based applications over a high bandwidth wireless channel, such as the High-Speed Downlink Packet Access (HSDPA) channel in a WCDMA system, in two setups: a nominal one that employs end-to-end TCP Reno and a new one that employs Cross-Layer Adaptation (CLA) in the form of Radio Network Feedback (RNF). For the CLA setup, the Radio Resource Management unit in the Radio Network Controller (RNC) provides a proxy with reports on the radio link bandwidth and the queue level. The proxy transport layer takes appropriate actions on these reports. By doing so, it utilizes the air-interface spectrum more efficiently and keeps the layer-2 queue in the RNC close a predetermined level. These new control mechanisms are evaluated through ns-2 simulations. In a number of realistic use cases it is shown that the new CLA setup reduces the time to serve users, and substantially increases the radio link utilization and decreases the required buffer size in the RNC.**

I. Introduction

In a near future, applications traditionally enjoyed over fixed Internet will be requested over high-speed cellular networks. File download, web browsing, and video telephony are examples of applications already available today in cellular networks [1], [2], whereas applications such as voice over IP, radio, television broadcast, and real-time gaming, are likely to emerge. Different applications set different requirements on the systems over which they are carried. Besides large bandwidth, important performance requirements for good service delivery are low end-to-end delay and low delay variation, see [3] for performance results on streaming services. Concerns have been raised if the current OSI stack is suitable for the highly dynamic wireless medium [4]. A plausible modification is to let higher-layer protocols in wireless networks dynamically adapt to varying radio conditions.

The available bandwidth in wireless networks is rapidly varying and follows a multi-modal distribution. It is therefore difficult to compensate for the variations with the current end-to-end mechanisms in the wired Internet, such as TCP. The large end-to-end round-trip time (RTT) that is common in wireless networks makes the control problem even harder [5]–[7]. For high-bandwidth and long-delay networks, TCP throughput is limited by the maximum TCP window size, which is less than or equal to 64 KB. The exponential back-off, slow start, and congestion avoidance mechanisms in TCP further limit the throughput and may also force over-dimensioned layer-2 buffers in the cellular system.

There are several proposals for improving TCP performance over wireless networks. One approach is to improve the TCP algorithm in the end points, e.g., Eifel [8] and TCP Westwood [9]. Deploying an improved TCP version, however, requires changes at the sending server or at the receiving terminal (or sometimes both), which might be less attractive in many situations. A different approach to improve TCP over wireless is to introduce a proxy between the server and the terminal, i.e., splitting the connection into one connection between the server and the proxy, and another one between the proxy and the terminal. This approach is thus often denoted split TCP [10]. It has the advantage that changes in the system are made only to the proxy and possibly the terminal; the server will see an ordinary wired network. With split TCP, one can in many cases use an arbitrary transport protocol between the proxy and the terminal. This may be an attractive option if the operator has control over the networking software running on the terminals, but when supporting off-the-shelf terminals, deployment of a specialized transport protocol is difficult.

The main contribution of this paper is a new proxy-based scheme for improving both the user experience of wireless

Internet, and the utilization of existing infrastructure. We propose a new custom protocol between the Radio Network Controller (RNC) of the cellular system and a proxy that resides between the Internet and the cellular system. Letting different layers of the OSI stack collaborate with each other is generally denoted as Cross-Layer Adaptation (CLA). In our case, the data-link layer within the RNC provides information to the transport layer within the proxy, and we refer to this communication as Radio Network Feedback (RNF). We use the layer-2 trigger mechanism presented in [11]. When the radio link bandwidth changes, the RNC sends an RNF message with the radio connection's new bandwidth to the proxy. The proxy then takes appropriate action by adjusting the TCP window size. The computation of a suitable window size in the proxy also takes into consideration the queue in the RNC, which needs to be controlled at a suitable level due to delay fluctuations and other uncertainties in the cellular system. Note that the proposed scheme does not require any changes to the terminals or to the Internet server, but they may utilize any of the standard TCP protocols.

Our proxy solution uses the estimated bandwidth delay product directly for rate control. This idea is also proposed in [12], which uses cross layer adaptation locally in the mobile node, and computes its advertised window from information about the radio link bandwidth. Localizing the cross layer interaction to the mobile node is attractive, but it is difficult to address outage-related problems this way, since during an outage, no signalling from the mobile node is possible. Using a split-connection proxy for TCP over WCDMA is evaluated in [13]. The proxy setup in this study improves the slow start behavior, but doesn't consider the response to bandwidth changes and outages.

The rest of the paper is organized as follows. In Section II, we describe the new architecture based on RNF signaling between RNCs and proxies. Section III describes how the RNF information is used in the TCP stack of the proxy, to control the window size and hence also the sending rate. We present and discuss our simulation results in Section IV, and conclude in Section V.

II. Architecture

The system architecture is shown in Figure 1, where two TCP connections have been established: one between the terminal and the proxy and one between the proxy and the server. The RNC encapsulates state information in RNF messages, which are sent to the proxy via UDP. The state information includes the available radio bandwidth for the specific TCP flow and the corresponding queue length in the RNC. The RNF messages are sent every time the bandwidth changes, but also periodically with a relatively long period time, e.g., one second.

We will compare this system to the nominal setup, in which there is a direct TCP connection between the terminal and the server.

In a typical scenario, a mobile terminal wants to request a file from a remote web server. It initiates a TCP connection to the proxy, and sends an HTTP request. The proxy receives the request, initiates another TCP connection to the server, and forwards the request. Upon receipt of the request, the server begins to send the file according to its standard TCP algorithm. The transferred data is received by the proxy, where it is buffered and then forwarded to the mobile terminal as soon as possible. Note that the bottleneck for the end-to-end connection in most practical situations is in the cellular system. Even if there is a TCP connection between the proxy and the terminal, the proxy does not need to utilize the standard TCP congestion control. The idea here is to modify the control of the proxy's sending rate to adapt to the varying wireless conditions. In this way, the TCP connection to the remote server does not suffer from the effects of the wireless link, and the connection to the mobile terminal is tailored to the current link characteristics. The control algorithm of the proxy is described next.

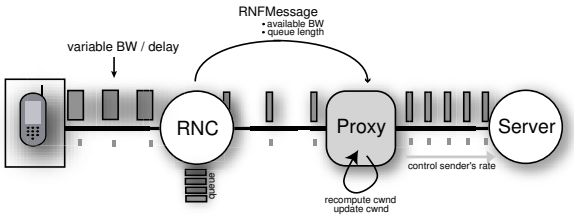

Fig. 1. The RNC sends RNF messages to the proxy, which adjusts its sending rate accordingly.

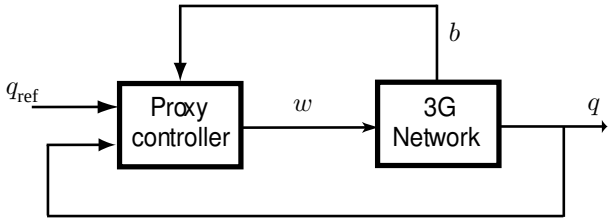

Fig. 2. The controller that determines the TCP window used by the proxy. It uses feedback from the RNC queue length, and feed-forward from the available bandwidth. Both are measured by the RNC and sent to the proxy in the form of RNF messages.

III. Proxy controller

The proxy controls the sending rate by adjusting the TCP window size based on information about the available radio link bandwidth and the RNC queue. The control output is the TCP window size (cwnd) denoted w. We use feedback from the RNC queue length, denoted q, which we want to control so that it stays close to a desired level q_{ref}. We also use feed-forward from the radio link bandwidth, denoted b. Bandwidth variation act as a disturbance which we can measure but not affect, which is why this part of the controller is a feed-forward compensator. The control structure is shown in Figure 2.

The window size should usually reflect the bandwidth-delay product, i.e., $b\tau$, where b denotes the available bandwidth and τ is the RTT. Since we want to keep the RNC queue q close

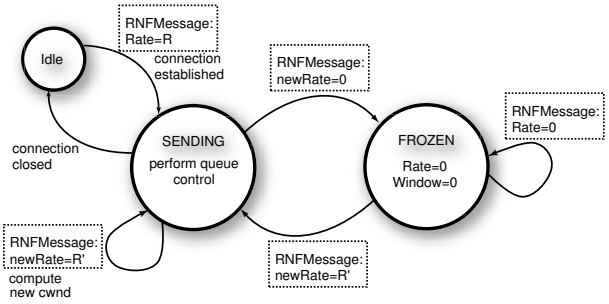

Fig. 3. State diagram of the proxy controller

to q_{ref}, it is reasonable to set

$$w = b\tau + q_{\text{ref}}. \tag{1}$$

The feed-forward part of the proxy controller handles the bandwidth variations and it is event driven: Whenever the proxy is informed of a change in the bandwidth b, a new congestion window is computed according to Equation (1). To avoid packet bursts, a small delay is introduced between the initial packets sent from the proxy. Such an approach was suggested in [14] to reduce the impact of bursts in slow start.

The feedback part of the proxy controller is intended to adapt to delay changes and uncertainties in the network. Periodically, e.g., once every second, the RNC sends an RNF message with its average queue length. The proxy adjusts its window size by

$$w_{\text{new}} = w_{\text{old}} + (q_{\text{ref}} - q) \tag{2}$$

When the delay in the network increases, more of the packets in a window will be in transit, and fewer will be waiting in the RNC buffer, leading to a decreased queue length. The feedback law of Equation (2) responds by increasing the window size w. Figure 3 shows the state diagram of the resulting proxy controller.

A challenge in split TCP is how to handle the buffer in the proxy, which is needed because of the bandwidth variations in the connection between the proxy and the terminal. Even if the traffic to the terminal is controlled as described above, the amount of buffered data in the proxy can increase considerably, leading to scalability problems when applying the proposed proxy controller in practice. It is necessary to set a limit on the amount of data per connection that the proxy can buffer. The remote server should thus be informed when the proxy is running out of space. The connection between the server and the proxy uses the receive window sent from the proxy to adjust the server sending rate. A reasonable buffer size for a TCP flow is the bandwidth-delay product for connection between the server and the proxy [15].

IV. Simulation results

In order to evaluate the performance of the proposed proxy controller in some practical situations, a set of use cases was defined and a ns-2 simulation study was performed for each of them. The proxy setup in Figure 1 is compared to a nominal setup, in which there is no proxy but only direct TCP connection between the terminal and the server. The links between the server and the RNC are set to 2 Mbps. The wireless link between the RNC and the terminal has a lower time-varying bandwidth and a one-way delay of 60 ms. We also use an average RTT (server-terminal-server) of 300 ms.

The performance of the proxy setup was evaluated in terms of the average link utilization and the time to serve a user (TTSU). The TTSU is defined as the time elapsed from when the connection is established (SYN packet sent by the requester) until the last data packet is received at the terminal. We consider three use cases, aimed at modeling common services to be provided over a WCDMA system. Use case 1 models a small (4 MB) file transfer. Use case 2 imitates web browsing and is based on statistical data [16], [17]. In this use case, three files (whose sizes vary between 51.5 KB and 368.5 KB) are downloaded to a terminal during one web session. Use case 3 models a large (25 MB) file transfer. Table I summarizes the results.

A 113.2 KB file download is illustrated in Figure 4, where the available bandwidth of the wireless link is shown by the dashed line. The bandwidth variations are due to the varying conditions of the radio link. The dotted line shows the utilization for the nominal setup with TCP Reno and without proxy. For this setup the slow start Threshold was set to 50, in order to ensure that the TCP sender does not enter congestion avoidance phase prematurely. The solid line shows the utilization with the new proxy solution. Examining this figure, we can make the following observations:

- None of the setups achieves full link utilization, but the proxy setup tracks the bandwidth variations much better than the nominal setup, and it adjusts more quickly to the available bandwidth.
- There is an initial delay of 0.4 seconds, which corresponds to the TCP connection establishment, and affects both setups equally.
- In the proxy setup, it still takes some time to reach full speed. This is because the rate is limited by the slow start phase of the TCP connection between the server and the proxy.
- In the nominal setup we see oscillations in the rate during the slow start phase. This is a result of the bursty behavior of standard TCP slow start.
- It appears that the sending rate in the proxy setup occasionally exceeds the available bandwidth. That is not possible, and the apparent excesses are artifacts of the sampling of the ns-2 simulation.

Outages and disconnections, which are typical for wireless connections, affect the performance of TCP connections considerably. Let us study the time to transfer a 4 MB file for both setups, when there is an outage in the middle of the file transfer. Figure 5 shows how the TTSU depends on the duration of the outage. For the proxy setup, an outage only increases the TTSU by the precise duration of the outage. The proxy controller is able to successfully resume the transmission

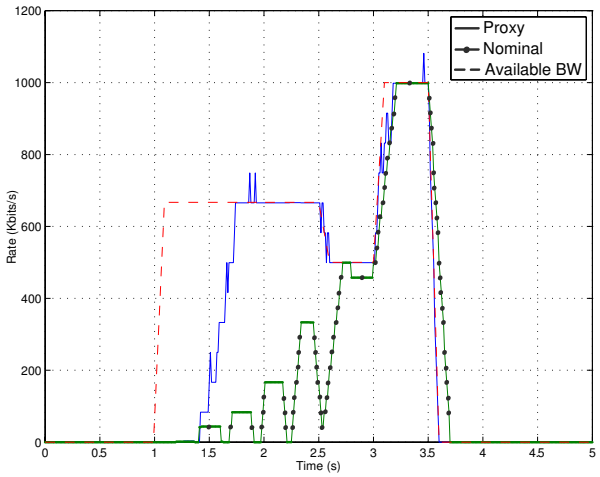

Fig. 4. Available bandwidth (dashed) over the wireless link, compared to the actual utilization for the proxy setup (solid) and the nominal setup (dotted).

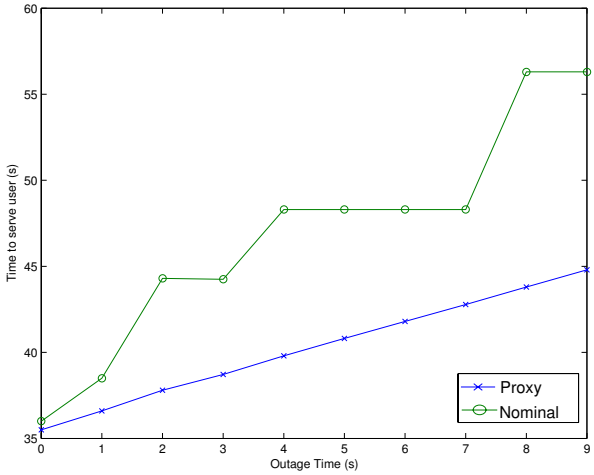

Fig. 5. Comparison of the time to serve a user as a function of the duration of an outage during a 4 MB file transfer for the proxy and the nominal setups.

immediately after the terminal is reconnected. For the nominal setup, on the other hand, the increase in TTSU is significantly larger than the outage duration, due to the exponential back-off mechanism in TCP Reno.

The queue length at the RNC (the bottleneck) is important to study, because it affects not only the response time and the radio link utilization, but also the necessary buffer allocation at the RNC. Figure 6 shows the length of the RNC queue during a 4 MB file transfer for, three different cases. Current RNC implementations often use very large buffers, represented by the dashed curve in the figure. In the nominal setup, this leads to a build up of a large queue, since the RNC never drops packets due to overflow. The dotted curve corresponds to a smaller buffer of 25 packets, which is better suited for TCP. We see that the sending TCP enters the congestion avoidance phase. The buffer is sufficiently large to achieve full utilization during congestion avoidance; with a significantly

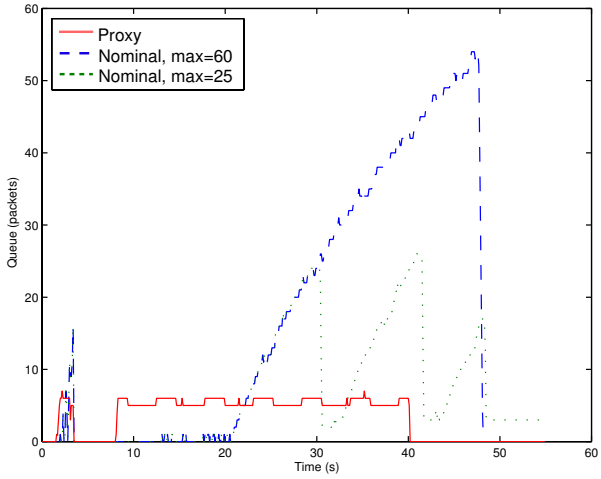

Fig. 6. Queue length at the RNC for a 4 MB file transfer. We compare the nominal setup with a large, 60 packet, buffer (dashed), a smaller, 20 packet, buffer better suited for TCP (dotted), and for the proxy setup with periodic feedback about the queue (solid). The proxy controller is able to keep the queue at a small and fairly constant length.

		TTSU [s]	**Utilization** [%]
Use case 1	Nominal	47.3	77%
	Proxy	39.2	98%
Use case 2	Nominal	3.9	51%
	Proxy	3.0	78%
Use case 3	Nominal	233.6	61%
	Proxy	220.5	98%

TABLE I

COMPARISON OF TIME TO SERVE A USER (TTSU) AND LINK UTILIZATION FOR THREE USE CASES.

smaller buffer, the queue would become periodically empty, which leads to reduced link utilization.

Finally, the solid curve shows the queue size in the proxy setup. The feedback control maintains a small and fairly constant queue. Hence the buffering needs in the RNC is reduced significantly compared to the nominal setup where the sender uses TCP's congestion avoidance algorithm.

Finally, we summarize the achieved TTSU link utilization for the three use cases, in Table I. The results for use case 1 show that TTSU is reduced by about 17% for the proxy setup. The average link utilization is close to 100% for the proxy setup, while it is about 77% for the nominal setup. Similar improvements are present also for the other use cases, basically reflecting that the nominal setup is not able to fully utilize the wireless resources, while the proxy adapts to achieve a higher utilization. The improvements are larger when (possibly many) small files are transferred, as indicated by the web browsing in use case 2. Many outages and bandwidth variations also worsen the performance for the nominal setup, compared to the new proxy solution.

V. CONCLUSIONS

TCP-based applications in a WCDMA system were studied in two setups: a nominal one that employs end-to-end TCP Reno

and a new one with and intermediate proxy. The proxy solution uses Radio Network Feedback (RNF) messages from the data-link layer in the RNC to the transport layer in the proxy. The proxy uses feed-forward control to adjust to a varying bandwidth, and feedback control to maintain the RNC queue length close to a desired value. It was shown in three realistic use cases that the new setup reduces time to serve users, substantially increases radio link utilization and decreases the needed buffer size in the RNC. The proposed control architecture might substantially improve the user experience of wireless Internet. The proposal is transparent to both end points, so no modifications need to be done to the terminal or the server.

Control theoretic analysis of our proposed controller is ongoing [18]. From the analysis, we gain understanding of the dynamical behavior of the system, we can find the relevant time constants, and we get help choosing the right values for the various control parameters.

ACKNOWLEDGMENT

The authors are grateful for discussions with Håkan Hjalmarsson, Krister Jacobsson and Mikael Johansson. The work was partially supported by European Commission through the Network of Excellence HYCON, by the Swedish Foundation for Strategic Research through an Individual Grant for the Advancement of Research Leaders, and by the Swedish Research Council.

REFERENCES

[1] http://www.3gpp.org.
[2] http://www.openmobilealliance.org.
[3] S. Chemiakina, L. F. Forti, R. Lalli, J. Petersson, and A. Terzani, "QoS enhancement for adaptive streaming services over wcdma," *IEEE JSAC on Recent Advances in Wireless Multimedia*, 2003.
[4] W. Kellerer, L.-U. Choi, and E. Steinbach, "Cross-layer adaptation for optimized B3G service provisioning."
[5] M. Kazantzidis and M. Gerla, "End-to-end versus explicit feedback measurement in 802.11 networks."
[6] D. Dutta and Y. Zhang, "An early bandwidth notification (EBN) architecture for dynamic bandwidth environment," in *IEEE International Conference on Communications*, vol. 4. IEEE, April 2002.
[7] H. Lim, K. Xu, and M. Gerla, "TCP performance over multipath routing in mobile ad hoc networks."
[8] R. Ludwig and R. H. Katz, "The Eifel algorithm: Making TCP robust against spurious retransmissions," *ACM Computer Communication Review*, vol. 30, no. 1, January 2000.
[9] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang, "TCP Westwood: bandwidth estimation for enhanced transport over wireless links," in *MobiCom*, Rome, Italy, 2001.
[10] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby, "Performance enhancing proxies intended to mitigate link-related degradations," RFC 3135, June 2001.
[11] S. Chemiakina and J. Petersson, "Radio network based adaptation method for packet switched services over 3G mobile systems," in *IST Mobile and Wireless Communications*, June 2004.
[12] D.-S. Lee and C.-C. Lin, "Window adaptive TCP for EGPRS networks," *Journal of Information Science and Engineering*, vol. 20, no. 5, pp. 805–820, September 2004.
[13] M. Meyer, J. Sachs, and M. Holzke, "Performance evaluation of a TCP proxy in WCDMA networks," *IEEE Wireless Communications*, pp. 70–79, October 2003.
[14] Y. Nishida, "Smooth slow-start: refining TCP slow-start for large-bandwidth with long-delay networks," in *23rd Annual Conference on Local Computer Networks*, October 1998, pp. 52–60.
[15] M. Gerla, M. Y. Sanadidi, R. Wang, and A. Zanella, "TCP westwood: Congestion window control using bandwidth estimation."
[16] C. H. Muntean, J. McManis, and J. Murphy, "The influence of Web page images on the performance of Web servers," *Lecture Notes in Computer Science*, vol. 2093, 2001.
[17] B. A. Huberman, P. L. T. Pirolli, J. E. Pitkow, and R. M. Lukose, "Strong regularities in World Wide Web surfing," *Science*, vol. 280, no. 5360, pp. 95–97, 1998.
[18] N. Möller, I. Molero, K. Johansson, J. Petersson, R. Skog, and Å. Arvidsson, "Using radio network feedback to improve TCP performance over cellular networks," in *IEEE Conference on Decision and Control*, 2005, (submitted).