

Response Time Guarantees for Networked Control Systems

Hans Hansson

Mikael Sjödin

Andreas Ermedahl

Department of Computer Systems, Uppsala University, Uppsala, Sweden

E-mail: {hansh, mic, ebbe}@docs.uu.se

Abstract

We present an overview of the research on real-time networking in the Hard-Real-Time Systems research group at the department of Computer Systems, Uppsala University. This includes a presentation of the general motivation and goals, as well as results and plans. On a more technical level, we present our method for providing timing guarantees in ATM-networks.

1 Introduction

Motivated by the need to provide system wide timing guarantees in distributed real-time systems, and based on earlier work on timing analysis for the CAN-bus, we have developed Response-Time Analysis (RTA) for providing real-time guarantees in Asynchronous Transfer Mode (ATM) communication networks. RTA is a method for analysis of network traffic which can be used by an admission control mechanism to determine if the timing requirements of a new connection can be guaranteed without violating the requirements of already admitted connections. This type of admission control is required by several multimedia and almost all real-time control applications.

RTA is based on fixed-priority real-time CPU schedulability analysis. This type of analysis has been extended by several researchers (in particular by members of the Real-Time Systems Group at the University of York) to be applicable in a wide range of scenarios. As a result, it is now possible to develop end-to-end analysis for distributed control systems with a hierarchical communication topology, consisting of for instance a high-speed ATM backbone network interconnecting a set of production cells consisting of controller nodes interconnected with a low speed bus (e.g. CAN).

This paper presents and discusses our work on timing guarantees for networked real-time systems, but before focusing on networking we will give a brief overview of current research in the Hard-Real-Time Systems research group at the department of Computer Systems, Uppsala University. Current research include:

- Worst case execution time (WCET) analysis. Here we are both considering high-level analysis (source code analysis), with the ambition to use semantic information to bound the execution times of programs [5], and low-level analysis (object-code/computer architecture

related analysis) focussing on taking cache-memories, execution pipelines, and other architectural features into account [15]. As a first effort in combining high and low-level analysis we are currently studying methods to keep track of how compiler optimisations influence execution times. The WCET work is performed in cooperation with the compiler vendor IAR Systems AB.

- Methods for real-time systems software design. Based on the software development methodology outlined for the distributed real-time platform BASEMENT [8], we are currently developing a methodology for software development for automotive control systems. We intend to combine high-level formal methods based modelling and verification with translation to executable code and scheduling of target system resources. This work is performed in cooperation with Mecel AB and the Design and Analysis of Real-Time Systems (DARTS) group at our department.
- Methods for providing timing guarantees in networked real-time systems. This work includes analysis of CAN-buses [27, 26] and ATM networks [6, 22], as well as the above mentioned BASEMENT system. We provide strict guarantees and analysis by applying results from real-time scheduling theory. In BASEMENT we used static cyclic scheduling and in the more recent work we are using fixed priority scheduling [1].

Our general scientific approach is to extend and adopt theory, guided by case-studies and experiments. This includes development of prototype tools. Our focus is on practically useful techniques for developing provably correct hard real-time systems.

The reminder of this paper is organised as follows: In Section 2 we present our vision for timing guarantees in networked factory systems. Section 3 presents our Response Time Analysis method for providing timing guarantees in ATM-networks, and in Section 4 we evaluate RTA by a comparison with some alternative methods. Finally, in Section 5 we conclude and present some future plans.

2 A General Vision

The use of communication networks in modern factories and other control systems is rapidly increasing. Many applications using the services of these networks have strict timing requirements. This include applications related to the command and control of equipment used in

the production process. For these applications, statistical guarantees, e.g. in terms of “message loss probability” or “average message delay”, are not sufficient. Instead, a firmer commitment is needed from the network, saying that it will *always* deliver *all* messages *completely* and *within* specified deadlines (with reservation for hardware failure and external physical interference).

In general, a factory communication system consists of several subsystems and subnetworks, e.g. as shown in Figure 1. The depicted communication system consists of:

- A set of production cells, some containing a multiprocessor and some a fieldbus (e.g. CAN [2]) interconnecting a set of nodes with I/O-devices (sensors and actuators) for interaction with the controlled processes,
- A set of nodes with specific tasks, e.g. a Human Machine Interaction (HMI) node handling operator interactions, a number cruncher providing high performance computation services, and a central database for storage of production information and logging, etc., and
- A backbone network (in our case an ATM network) for interconnecting the different subnetworks and nodes.

It should be noted that this is just an example of a factory communication system. Other topologies and configurations are certainly possible (i.e. a further substructure into subnetworks within production cells or multiple MMI nodes etc.). Also, in this type of environment there will be a mix of traffic, including real-time control messages, multimedia (e.g. for monitoring) and less critical messages such as those related to collecting statistical data.

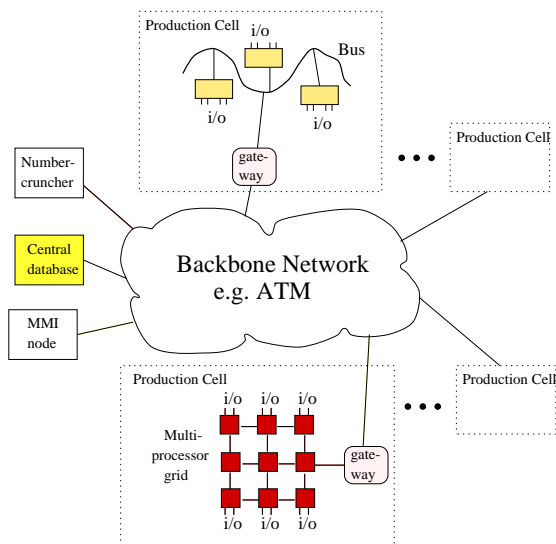


Figure 1. A factory communication system

It is our ambition to provide end-to-end real-time guarantees in such a heterogeneous system. This requires an holistic method which in an integrated way can be applied to scheduling and analysis of all involved resources (i.e. the different types of networks, CPUs, etc.). Fixed Priority Scheduling (FPS) and the associated Response Time Analysis theory [1] provides a powerful basis for this.

RTA is based on traditional CPU schedulability analysis [14, 12] but has also proven useful to determine response-times for other types of resources, e.g. *Controller Area Networks* (CAN) [27] and disk drives [24]. Methods for combining the scheduling analysis of different resources in distributed environments has also been developed [25].

In the following sections we will present how we have applied RTA to scheduling of cells in ATM-networks. The developed analysis is one component in the holistic analysis for providing end-to-end guarantees in heterogeneous distributed real-time systems.

3 Response Time Analysis of ATM Networks

Several methods for providing exact guarantees in real-time networks have been proposed in the literature [7, 4, 16, 17, 28, 18, 19, 13, 23, 20, 9]. They all provide or assume some mechanism of bandwidth sharing and have an associated analysis that can be used by an *admission control* procedure to *a priori* determine if the *Quality of Service* QoS requirements of a new connection be met, while not violating the requirements of already admitted connections,

A positive answer to this question means that the connection can be admitted. A negative answer means that if the new connection is admitted the QoS of the new connection, as well as the QoS of already admitted connections, may be violated.

We will here outline the Response-Time Analysis (RTA) method introduced in [9], and extended and evaluated in [10, 6, 22]. We have focused on ATM, since it is likely that ATM will be used in industrial control systems and other *safety critical* applications, simply because it will be the cheapest high bandwidth communication technology available.

RTA is attractive since it uses a very simple priority queuing mechanism that provides a clear separation between bandwidth and deadline requirements. This gives RTA a potential of offering a bandwidth utilisation outperforming currently popular bandwidth reservation schemes, such as Weighted Fair Queuing (WFQ) [4, 16, 17]. We test this hypothesis by comparing RTA and WFQ for some realistic traffic scenarios. In addition, we compare RTA with the Calculus for Network Delays (CND) [3], which is an alternative analysis method when using priority queuing.

An additional benefit with priority queuing is its efficient integration of hard real-time traffic with other QoS classes which do not require absolute guarantees. Non hard real-time traffic is simply assigned lower priority, meaning both that it will not interfere with the hard real-time traffic and that it can use any bandwidth not used by the real-time traffic.

3.1 Asynchronous Transfer Mode

Asynchronous Transfer Mode (ATM) [11] is a high performance, connection oriented, network architecture. Over each connection a *stream* of fixed size packets, called *cells*, are transmitted. An ATM network consists of a set of

ATM switches. Each switch routes cells from a set of *input ports* to a set of *output ports* based on stream identifiers in the cell-headers. At the output port of an ATM switch a *port controller* multiplexes cells from different streams onto the outgoing line. The output ports of an ATM switch are governed by a *scheduling policy*, e.g. First In, First Out (FIFO), which decides what cells to dequeue from the *output queue*. Our RTA assumes ATM switches using priority queuing, which is in accordance with current switch technology (e.g. Fore Systems ASX-200BX, ASX-1000 and Cisco LS 1010 all use priority queuing).

3.2 Traffic characterisation

To be able to analyze network delays a model of the traffic is needed. Often traffic is characterised in *network terminology*, using terms such as “average bandwidth”, “maximum burst size” and “burst bandwidth”. Since such terms mean little (or nothing) for an engineer developing hard real-time applications, we use the following more application oriented parameters, to characterise a connection i :

T_i is the shortest *period* with which messages are generated.

N_i denotes the *maximum number of cells* needed for a message. An ATM cell has 48 bytes of payload, thus, for a stream with maximum message size of X bytes $N_i = \lceil X/48 \rceil$.

J_i is the maximum *jitter* of a traffic source, e.g. caused by the operating system scheduling on the host where the application executes.

D_i is the *deadline* for end-to-end message delivery. This is the hard real-time requirement which always must be met.

t_i is the *inter-cell delay* after a message has passed through a cell spacer (e.g. a leaky bucket) at the source node.

3.3 Response-Time Analysis

In this section we will outline RTA.

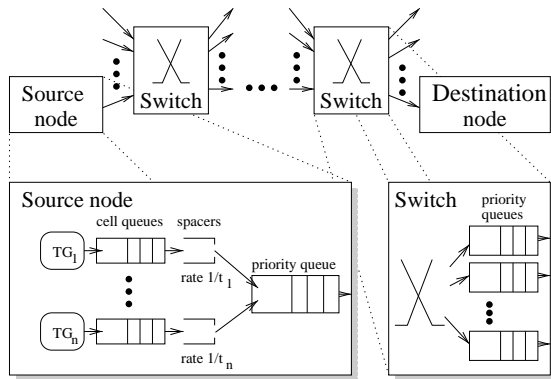


Figure 2. The network architecture

Figure 2 shows the general structure of the considered network. In the figure, TG_i denotes a traffic generating process, which generates messages to be transmit-

ted. These messages are segmented into cells, which are queued in a cell queue associated to TG_i . A cell spacer moves cells with a rate of $\frac{1}{t_i}$ from the cell queue to the prioritised output queue. The reason for using a cell spacer is to reduce the burstiness of the traffic and thereby reduce congestion and increase network utilisation. From the output queue, cells are sent to the first switch. In the switch, cells are placed in the appropriate output queue, and transmitted further to the next switch until the destination is reached.

The following formula defines the worst case queuing delay in one of the queues along the path of an ATM connection. Provided that there is at least one cell from stream i in the queue from time 0 when the first cell (numbered 0) arrives, the queuing delay for the k -th cell in stream i is given by:

$$Dequeue(i, k) = \left(1 + k + \sum_{j \in sp(i)} Arrived(j, Arrival(i, k)) + \sum_{j \in hp(i)} Arrived(j, Dequeue(i, k)) \right) \tau$$

The first summand captures that at most one cell of a lower priority stream gets sent before cell k . The second summand captures the delay caused by the k cells in stream i which have been enqueued before cell k . The third summand¹ captures interference caused by the cells from streams sharing priority with stream i which have been queued before, or at the same time as, cell k . $Arrived(i, t)$ denotes the maximum number of cells from stream i which has arrived at time t , and $Arrival(i, k)$ denotes the earliest arrival time of cell k . The fourth summand² denotes interference caused by higher priority streams arriving before the dequeuing of cell k .

Using $(a \text{ rem } b)$ to denote be the remainder, and $(a \text{ div } b)$ to denote the integer part, of a/b , we define

$$Arrival(i, k) = \max(0, (k \text{ div } N_i)T_i + (k \text{ rem } N_i)t_i - J_i)$$

Intuitively, for a stream with messages consisting of N_i cells queued at the source with a period of T_i , $Arrival(i, k)$ is equal to the difference in queuing time of cell 0 and cell k (which is the $(k \text{ rem } N_i)$ th cell in message $(k \text{ div } N_i)$) minus the jitter J_i , which may cause cells to arrive closer together. The time of arrival may however not be negative, which is captured by the “ $\max(0, \dots)$ ”.

$$Arrived(i, t) = \underbrace{((t + J_i) \text{ div } T_i) N_i}_{\text{Whole messages}} + \underbrace{\min \left[\left(((t + J_i) \text{ rem } T_i) \text{ div } t_i \right) + 1, N_i \right]}_{\text{A partial message}}$$

¹ $sp(i)$ denotes the set of streams which have the same priority as i (except stream i itself)

² $hp(i)$ denotes the set of streams which have higher priority than stream i

Similarly, $Arrived(i, t)$ is equal to the number of cells in complete messages arrived at time t plus the number of cells in the last message which may only have arrived partially. Note that the jitter allows time between the two first messages to $T_i - J_i$ instead of a full period, T_i .

Knowing both the enqueueing and dequeueing time of each cell k we also know the queuing delay of cell k . Thus, the maximum queuing delay for a cell in stream i is

$$Q_i = \max_k (Dequeue(i, k) - Arrival(i, k))$$

The number of cells (i.e. k -s) which have to be investigated to find Q_i is bounded, but it is beyond the scope of this paper to derive and prove that bound. For a more thorough discussion on the theory of RTA we refer to [21].

3.4 Traffic Shape Modification

In order to apply RTA on a subsequent switch it is necessary to capture how the traffic shape change when passing through the output queue. In general, the added jitter from a network component is equal to the difference between the maximum and the minimum time a cell is delayed in that component. In the case of an output queue the maximum delay is Q_i and the minimum delay is 0. Thus the new jitter, J'_i , becomes

$$J'_i = J_i + Q_i$$

Also, the *remaining time budget* for the stream will decrease. This means that for the subsequent switch the deadline will be tighter. The remaining deadline D'_i becomes

$$D'_i = D_i - Q_i$$

4 Evaluation of RTA

In this section we present a comparison of RTA with two other analysis methods: the analysis presented in [16, 17] for Weighted Fair Queuing (WFQ), and Cruz's Calculus for Network Delays [3]. For a more detailed presentation of the comparison we refer to [6, 22].

We have chosen WFQ to represent the rate-based service disciplines (i.e. disciplines based on bandwidth reservations) since it is the most well known such method. CNP is evaluated since it is an alternative method to RTA for analysing priority queuing.

Since both CNP and RTA both are applicable to both FIFO and priority queuing (PRIQ) we will study both methods on both scheduling policies.

4.1 Measuring Performance

A good way to understand how well an analysis method performs is to study how much of the network resources it allows to be allocated before the Call Admission Control (CAC) fails. In our comparison the CAC is performed for scenarios consisting of whole set of streams. The CAC is successful if all streams in the scenario can be guaranteed to meet their respective deadlines.

We will study the *admission probability* [18] curve for each method. The admission probability for a specific

level of network load is defined as the probability of performing a successful CAC for a scenario with that load.

Naturally, the admission probability is highly dependent on the profile of the streams in the scenario. If the scenarios contains many streams with tight deadlines, the admission probability can be very low even for low network loads.

In addition to evaluating the admission probability, we use simulation to evaluate the precision of the analysis, by comparing observed worst-case response times with those obtained by analysis.

4.2 Network Topology

To avoid having too many variables we use a fixed network topology, illustrated in Figure 3. We define the network load as the load on the link to the destination node (link 10). We distribute the streams evenly over the 6 source nodes, thus, when the network load is 100%, the load on link 7 will be approximately 33%. We have chosen a moderate bandwidth of 100Mbps for each link in the network (i.e. $b = 10^8$).

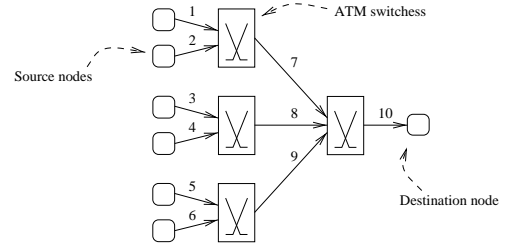


Figure 3. Network topology

4.3 Evaluation results

Figure 4 shows the admission probability curve for a traffic scenario dominated by control software messages, and Figure 5 presents an evaluation of the precision of the different analysis methods.

From the figures it is evident that RTA outperforms all the other analysis methods and that PRIQ has vastly better performance than FIFO.

The fact that CNP performs worse than RTA (both for PRIQ and FIFO) is not very surprising since CNP uses a traffic model which is more pessimistic than RTA's traffic model. The poor performance of WFQ is more surprising, since WFQ often is advocated, in the computer- and telecommunication communities, as the solution to provide QoS guarantees.

As far as precision is concerned, we observe that both PRIQ methods experience lower precision (higher overestimation) as the network load increases. The reason for this is that both RTA and CNP have higher precision for higher priority streams while low priority streams have relatively high probability to receive better service than the worst-case. When the load increases, more and more streams with higher priority will be added, and hence, the precision decreases.

The precision for WFQ is by far the lowest. The precision for WFQ *increases* as load increases, contrary to the PRIQ and FIFO policies. However, this is quite natural, since the analysis of WFQ considers each stream in isolation under the assumption that the whole bandwidth of each link is allocated. As the load increases, this assumption becomes more and more valid, thus the precision increases.

5 Conclusions

We have outlined our general ambitions and approach to provide real-time guarantees in networked control systems, and presented in some detail one important component in such a framework: the RTA method for admission control analysis in ATM networks. RTA uses traditional real-time response-time type analysis [12, 1] to calculate the worst-case response time of ATM-cells sent through an ATM-network using priority queued output buffers in the switches.

RTA can be incorporated with existing frameworks (e.g. Fixed Priority Scheduling [1]) for real-time system schedulability analysis, thereby allowing system designers to use the same technique for assessing the timely behaviour of the entire system, including both CPU and network scheduling.

We have compared RTA with Weighted Fair Queuing (WFQ), a method which makes fixed bandwidth allocations to guarantee both bandwidth and delay requirements, and the Calculus of Network Delays (CND), which just as RTA uses priorities to control the traffic flow.

In the evaluation we used a traffic scenario resembling traffic in a real-time system. The resulting comparisons show that RTA has both higher admission probabilities and higher precision in the analysis than both WFQ and CND. On the other hand, we have also made the observation that WFQ provides the fastest CAC. In fact, the RTA CAC is currently not fast enough to handle admission control in a relatively large network with dynamically arriving connections. On the hand, the majority of hard real-time systems (e.g. systems for process control) are quite small and typically have a relatively static set of hard real-time connections. For these type of systems, RTA seems to be the best alternative for providing hard real-time guarantees.

5.1 Future Plans

Our future work aims at fulfilling our vision of providing practically applicable end-to-end timing analysis for heterogeneous distributed control systems. This includes refinement of existing results, development of new analysis modules, development of prototype tools, and case-studies. In the relatively short time perspective we intend to

- combine RTA with multiprocessor allocation and scheduling,
- extend the evaluation of RTA for ATM to consider additional traffic scenarios and topologies,
- develop a more efficient CAC for RTA,

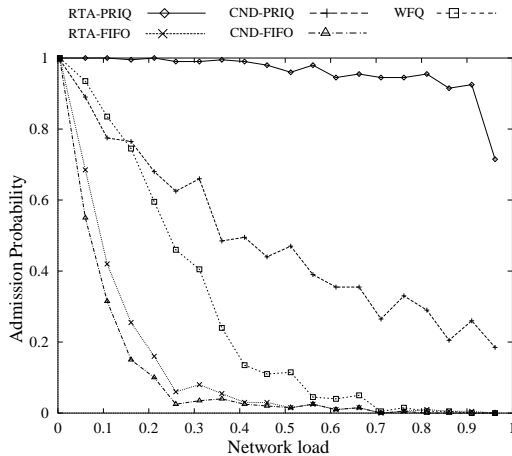


Figure 4. Admission probabilities for RTA, CND and WFQ

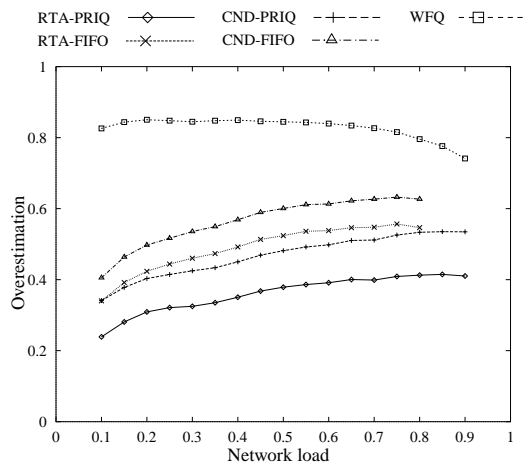


Figure 5. Precision of RTA, CND and WFQ

- develop simulation-based RTA methods and prototype tools, and
- perform a larger case-study to assess the practical applicability of our methods and tools.

References

- [1] N.C. Audsley, A. Burns, R.I. Davis, K.W. Tindell, and A.J. Wellings. Fixed Priority Pre-emptive Scheduling: An Historical Perspective. *Real-Time Systems*, 8(2/3):129–154, 1995.
- [2] Road Vehicles – Interchange of Digital Information – Controller Area Network (CAN) for High Speed Communication. ISO/DIS 11898, Feb. 1992.
- [3] R. L. Cruz. A Calculus for Network Delay, Part I and II. *IEEE Transactions On Information Theory*, 37(1):114–141, January 1991.
- [4] A. Demers, S. Keshav, and S. Shenker. Analysis and Simulation of a Fair Queueing Algorithm. *Internetwork: Research and Experience*, 1(1):3–26, September 1990.
- [5] A. Ermedahl and J. Gustafsson. Deriving annotations for tight calculation of execution time. In *Euro-Par’97 Parallel Processing*, number 1300 in Lecture Notes in Computer Science, pages 1298–1307, August 1997.
- [6] A. Ermedahl, H. Hansson, and M. Sjödin. Response-Time Guarantees in ATM Networks. In *Proc. 18th Real-Time Systems Symposium*, San Francisco, December 1997. IEEE Computer Society Press.
- [7] S. Jamaloddin Golestani. A Stop-and-Go Queueing Framework for Congestion Management. In *Proc. ACM SIGCOMM ’90*, pages 8–18, September 1990.
- [8] H. Hansson, H. Lawson, M. Strömberg, and S. Larsson. BASEMENT a distributed real-time architecture for vehicle applications. *Real-Time Systems*, 11(3):223–244, November 1996.
- [9] H. Hansson, M. Sjödin, and K. W. Tindell. Guaranteeing Real-Time Traffic Through an ATM Network. In *Proc. of the 30th Hawaii International Conference on System Sciences*, volume 5, pages 44–53. IEEE Computer Society Press, January 1997.
- [10] H. Hansson and M. Sjödin. Response-Time Guarantees for ATM Networked Control Systems. In *Proc. 1997 IEEE International Workshop on Factory Communication Systems*, pages 213–222, Barcelona, October 1997. IEEE Computer Society Press.
- [11] R. Händel, M.N. Huber, and S. Schröder. *ATM Networks Concepts, Protocols, Applications*. Addison-Wesley, second edition, 1994.
- [12] M. Joseph and P. Pandya. Finding Response Times in a Real-Time System. *The Computer Journal*, 29(5):390–395, 1986.
- [13] G. Karlsson. Capacity Reservation in ATM Networks. *Computer Communications, Special issue on Algorithms for ATM Networks*, March 1996.
- [14] C. Liu and J.W. Layland. Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment. *Journal of the ACM*, 20(1):46–61, 1973.
- [15] G. Ottosson and M. Sjödin. Worst-Case Execution Time Analysis for Modern Hardware Architectures. In *Proc. SIGPLAN 1997 Workshop on Languages, Compilers and Tools for Real-Time Systems (LCT-RTS’97)*, June 1997.
- [16] A.K. Parekh and R.G. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single Node Case. *IEEE/ACM Transactions on Networking*, 1(3):344–357, June 1993.
- [17] A.K. Parekh and R.G. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Multiple Node Case. *IEEE/ACM Transactions on Networking*, 2(2):137–150, April 1994.
- [18] A. Raha, N. Malcom, and W. Zhao. Performance Evaluation of Admission Policies in ATM Based Embedded Real-Time Systems. In *Proc. of the 19th Conference on Local Computer Networks*, 1994.
- [19] A. Raha, N. Malcom, and W. Zhao. Hard Real-Time Communications with Weighted Round Robin Service in ATM Local Area Networks. In *Proc. Engineering of Complex Computer Systems*, volume 1, pages 96–103. IEEE Computer Society Press, November 1995.
- [20] D. Saha, S. Mukherjee, and S.K. Tripathi. Multirate Scheduling for Guaranteed and Predictive Services in ATM Networks. In *Proc. 17th Real-Time Systems Symposium*, pages 155–164. IEEE Computer Society Press, December 1996.
- [21] M. Sjödin. Response-Time Analysis for ATM Networks. Department of Computer Systems, Uppsala University, Licentiate Thesis DoCS 97/92, May 1997. URL <http://www.docs.uu.se/~mic/lic.ps.gz>.
- [22] M. Sjödin. Response Time Analysis for ATM Networks. Technical Report DoCS 97/92, Department of Computer Systems, Uppsala University, 1997.
- [23] A. Stamoulis and J. Liebherr. S²GPS: Slow-Start Generalized Processor Sharing. In K. Jaffay, editor, *Participants Proc. of Workshop on Resource Allocation problems in Multimedia Systems (held in conjunction with IEEE Real-Time Systems Symposium)*. University of North Carolina at Chapel Hill, December 1996.
- [24] K.W. Tindell and A. Burns. Fixed Priority Scheduling of Hard Real-Time Multimedia Disk Traffic. *The Computer Journal*, 37(8):691–697, 1994.
- [25] K.W. Tindell and J. Clark. Holistic Schedulability Analysis For Distributed Hard Real-Time Systems. Technical Report YCS197, Real-Time Systems Research Group, Department of Computer Science, University of York, November 1994.
- [26] K.W. Tindell and H. Hansson. Babbling Idiots, Dual Priorities, and Smart CAN Controllers. In *Proc. 2nd International CAN Conference*, London, October 1995.
- [27] K.W. Tindell, H. Hansson, and A.J. Wellings. Analysing Real-Time Communications: Controller Area Network (CAN). In F. Jahanian and K. Ramamritham, editors, *Proc. 15th Real-Time Systems Symposium*, pages 259–263. IEEE Computer Society Press, 1994.
- [28] H. Zhang and D. Ferrari. Rate-Controlled Service Disciplines. *Journal of High Speed Networks*, 3:389–412, 1994.