COMPUTER COMMUNICATION REVIEW

A Publication of ACM SIGCOMM

Volume 32, Number 5 November, 2002
ISSN #: 0146-4833

Contents

Information for Authors

By submitting your article for distribution in this Special Interest Group publication, you hereby grant to ACM the following non-exclusive, perpetual, worldwide rights:

- to publish in print on condition of acceptance by the editor
- to digitize and post your article in the electronic version of this publication
- to include the article in the ACM Digital Library
- to allow users to copy and distribute the article for noncommercial, educational or research purposes

However, as a contributing author, you retain copyright to your article and ACM will make every effort to refer requests for commercial use directly to you.

Additional information for authors is available at the CCR website: http://www.acm.org/sigcomm/ccr

Notice to Past Authors of ACM-Published Articles

ACM intends to create a complete electronic archive of all articles and/or other material previously published by ACM. If you have written a work that was previously published by ACM in any journal or conference proceedings prior to 1978, or any SIG Newsletter at any time, and you do NOT want this work to appear in the ACM Digital Library, please inform permissions@acm.org, stating the title of the work, the author(s), and where and when published.

Workshop Report:

ACM SIGCOMM Workshop on Computer Networking: Curriculum Designs and Educational Challenges

Jim Kurose[1], Jörg Liebeherr[2], Shawn Ostermann[3], Theresa Ott-Boisseau[4]

[1] Department of Computer Science, University of Massachusetts, Amherst, kurose@cs.umass.edu
[2] Department of Computer Science, University of Virginia, jorg@cs.virginia.edu
[3] Department of Computer Science, Ohio University, ostermann@cs.ohiou.edu
[4] San Diego Supercomputing Center, University of San Diego, theresa@sdsc.edu

Abstract

This year's annual ACM Sigcomm Conference featured a one-day workshop entitled "Computer Networking: Curriculum Designs and Educational Challenges." The goal of the workshop was to bring together faculty from a broad spectrum of four-year colleges and universities, industry engineers and scientists, and others with an interest in networking education to discuss curriculum design and teaching practices in the field of computer networks. Eighty-nine people participated in this first-ever workshop focused solely on the educational aspects of the networking field. Workshop activities included panels on undergraduate curricula, laboratory-based courses, and graduate curricula. This report summarizes the workshop's presentations, discussions, and findings, as well as plans for future education-related activities.

1. Introduction

Perhaps reflecting the diverse, dynamic, and rapidly expanding set of topics within the field of computer networks itself, curricula and teaching practices in our field are also continuously in flux – with new topics, new courses, and new approaches to teaching being explored at colleges and universities around the world. Among the approaches towards networking curricula, one finds the more quantitative (electrical engineering) style of teaching networking versus a more software/algorithmic (computer science) approach, the more "hands-on" lab-based approach versus more traditional in-class lecture-based approach; the bottom-up approach towards the subject matter versus a top-down approach. New topics, such as peer-to-peer and mobile networking, and increased interest in more established topics, such as security, are further fueling changes to the content of networking courses. And yet, amidst this constant change and growth, the field of networking is arguably entering "middle age" at forty years old, with perhaps a body of "core" topics emerging that many networking educators feel should be mastered by all. The emergence of core networking material is also evidenced by the inclusion of a number of networking topics in the recent IEEE/ACM 2001 Computing Curricula report [IEEE/ACM 2001].

It is against this backdrop of continuing change and evolution, as well as the emergence of fundamental, core topics in our field, that the 2002 Sigcomm Education Workshop was held. The workshop itself consisted of three panels - on undergraduate curriculum, laboratory courses, and graduate curriculum. Each panel also had a corresponding breakout session. The topics posed for discussion at the workshop were the following:

- **The development of curricula for a first (undergraduate) course in networking.** What are the "core" topics that should be covered in a first course? Are there are small set of approaches towards teaching such a course (e.g., a more quantitative EE-style versus a more software/algorithmic CS-style; "hands-on" versus in-class lectures; bottom-up versus top-down approaches)? What are the roles of labs and/or programming projects? Which undergraduate multi-course sequences are possible? What is the relationship of such course(s) to the recent ACM/IEEE 2001 Computing Curricula report? What materials can be shared by instructors?
- **Laboratory courses.** What approaches can be taken in developing "hands-on" laboratory-based courses at the undergraduate/graduate level? How do laboratory-based exercises relate to traditional lecture-based courses? What topics might be covered, and how should they best be covered? What software, lab materials, and experiences can be shared?

- **The evolution of graduate-level curricula.** A first graduate-level networking course has often been an accelerated/augmented version of an introductory-level undergraduate course, while more advanced graduate courses have often focused on a single sub-area. Several schools have recently introduced multi-course graduate-level course sequences, some within the context of networking/telecommunications MS and PhD programs. What should graduate courses in our field look like? Which courses in specific subareas are desirable, and what might their content be? Is there a set of advanced, foundational material that applies broadly across the field at the graduate level?

Sections 2, 3 and 4 of this report summarize the presentations and discussions in each of these three areas. Section 6 concludes this report with a listing of resources identified by (or prepared by) workshop participants, as well as a discussion of planned future activities. A copy of this report, a collection of 25 white papers prepared by workshop participants in advance of the workshop, a copy of presentation overheads, and additional educated-related information can be obtained from the ACM SICCOMM Education web pages at http://www.acm.org/sigcomm/education.

2. Undergraduate Curriculum

In brief opening remarks, Shawn Ostermann of Ohio University presented the goals for the first session: *(i)* begin a new discipline-wide discussion of undergraduate networking education; *(ii)* begin the formation of a task force to create a report describing existing programs and outlining different teaching models that work well; and *(iii)* make specific recommendations when "rough consensus" is possible.

2.1 Undergraduate Curricula: Presentations

The presentations contained a wealth of valuable information about networking in general, teaching experiences, example classes and curricula, things to try, things to avoid, and things to ponder. The value of these presentations was not just in their detail, but perhaps more importantly in the tone and direction that they helped set for the discussions that followed. The following summary addresses only the salient points of the presentations that led to the recommendations noted at the end of this section. Readers are strongly encouraged to review the presentations in detail at http://www.acm.org/sigcomm/education.

Russell Clark from the Georgia Institute of Technology led off the morning with the first presentation. He presented a brief overview of Georgia Tech's five undergraduate networking courses and then discussed the first, "Introduction to Networking," in depth. This course is primarily a survey course and does not include a lab component. The course is offered twice a year to groups of up to 120 students at a time. The emphasis is on core concepts and best practices, reinforced with written assignments and socket programming. The presenter set the stage for later discussions with a final slide on "Challenges and Opportunities" that included:

- Top-down vs. bottom-up vs. neither– how should we teach networking?
- Theory vs. hands-on?
- How much programming and when?
- Dealing with large class sizes

Ralph Droms from Cisco Systems followed next, with the perspective of an academic researcher who recently moved from Bucknell University to Cisco Systems. The presentation focused on the experiences gained in teaching a networking course for many years and the lessons learned. Different from the program at Georgia Tech, Bucknell offers an introductory course with hands-on exercises, and involves a smaller number of students. The course includes a weekly lab component that covers topics ranging from packet tracing to implementing application protocols. The course culminates with a final project involving a large, distributed simulation. The presentation concluded with a pair of open questions that many have faced and pondered over the years: *(i)* how do we get students to use existing tools rather than "reinventing the wheel"; and *(ii)* how do we foster "problem analysis" and lead students away from their tendency toward "design by emacs"?

The third presentation, by Michael Greenwald of the University of Pennsylvania, emphasized his University's challenges in integrating a large and diverse student population that includes computer scientists, electrical engineers, and telecommunications engineers. The scope of the problem was emphasized by a listing of the university's 15 courses in networking-related areas. Of particular interest was the discussion of identifying "who are

our students?" when the population includes CS, EE, Business, and Liberal Arts majors. Each of the student groups has good reasons for needing to understand computer networking, but seemingly tackles the problem from a different angle. His presentation led to discussions on efficient use of classroom and instructor time in the face of these competing demands.

The next presentation was by Dave Morgan of Fidelity Investments, who provided insight into the kinds of jobs and experiences that undergraduate students find after leaving the university. He is involved in hiring 5 to 10 new college graduates each year for a unit with roughly 500 IT professionals. Among the skills valued in new undergraduate hires is an ability to use common tools, and experience in working with teams. Experience at Fidelity Investments has shown that a common problem among new hires is their failure to understand and anticipate the amount of network traffic caused by the new network applications that they design. Lack of first hand measurement and monitoring experience in school is seen as contributing to the problem and led to discussions of possible laboratory exercises in class.

The final presentation was by Craig Partridge from BBN and the Chair of ACM SIGCOMM. Craig Partridge emphasized the importance of having students "write code". He stressed that requiring students to write software is the best way to get them to internalize and understand advanced concepts. It has the further benefit of helping them recognize when they do not fully understand the intricacies of a networking topic. Finally, it leaves them with a marketable skill, which he stated should always be a goal in an undergraduate course. In addition to the benefits of having students write code, he added that it carries the usual burdens of forcing the instructor to make judgments about how much one can push the students without losing large portions of the class.

2.2 Undergraduate Curricula: Breakout Session

The afternoon break-out session on undergraduate curricula continued the lively discussion begun during earlier presentations, undertaking the challenge of identifying the most important concepts that needed to be part of an undergraduate networking curriculum. There seemed to be a general consensus throughout the day that much of what a practicing expert in networking needed to understand would need to come from a graduate degree, and that programs with the expertise and critical mass to provide such a high-quality graduate degree would continue to exist only at a relatively small subset of the world's colleges and universities. At the same time, a widely held belief was that the general concepts of computer networking are sufficiently important that they need to be presented at the undergraduate level to as wide an audience as possible. Supporting this belief is the fact that many colleges and universities now offer (or at least desire to offer) at least one undergraduate course in computer networking.

These discussions led to the consideration of the following question: "If students were only to take a single undergraduate networking course, what topics should that course contain?" It was noted that the hypothetical course being considered would have three basic purposes:

- To teach students enough about computer networks and the implications of their widespread deployment that students can make intelligent choices about how they use such networks;
- To pique the interest of at least a few students in further studying in the area of computer networks;
- To lay a common foundation so that graduate programs can make reasonable assumptions about what students already know.

An alternative phrasing of the question helped sharpen the discussion to focus on what truly constitutes core material – "Inadequate coverage of which topics, if not mastered by students who are supposed to be networking-literate, would be deeply embarrassing to us?" Two additional questions were posed and also helped frame the ensuing discussion: "*(i)* What do students need to understand about the network to use it safely and effectively in their day-to-day activities? *(ii)* What basic knowledge should students have upon entering a graduate program in networking?" It was noted that the answers to these questions are likely not to be the same, but that a topic appearing in the answer to all of these questions would be an excellent candidate for being considered a "core" topic.

With the question of "What would be deeply embarrassing if students did not know?" as a "prod" for identifying core topics, the breakout discussion group began the process of identifying such topics. There seemed to be rough consensus that the topics shown below in Table 1 should all be covered in an undergraduate networking course. We group the topics here by general area for clarity. The reader should not infer from this ordering that we are recommending a "bottom-up" over a "top-down" approach as both approaches have merit, as do their variations.

Physical network basics
- Digital channels
- Errors and error detection
- Understanding of one shared media access protocol (e.g., CSMA)
- A little bit about wireless LANs
- Shannon and Nyquist limits

Concepts
- Circuit switching vs. packet switching
- Framing and encapsulation

Network interconnection
- Moving packets through multiple networks (routing, internetworking)
- Addressing and forwarding

Protocols
- What a protocol is and how it is specified
- Reliable windowed/pipelined data transfer in the face of errors (including basics of TCP)
- Congestion control

Exposure to
- Client/Server programming
- Socket programming
- Managing and configuring a remote device
- Current application protocols and how they work

Recurring themes
- Security as a daily reality
- Encryption as a solution
- Elements of performance (transmission, propagation delay)

Table 1: A First Pass at Important Undergraduate Networking Topics

Workshop participants noted that the material on security was particularly important and should be a recurring theme throughout the course. The emphasis should be on "what all students need to understand about network security", not just (or even) the theory underpinnings of security. In particular, the following topics should be addressed at appropriate points in the class: using checksums to protect data, implications of clear-text passwords, the implications of using credit cards on the web, authentication, privacy (e.g., using PGP in e-mail).

2.3 Where do we go from here?

The topics listed in Table 1 are the result of an admittedly short and time-constrained discussion at the workshop. Nonetheless, we were sufficiently encouraged with the amount of "rough consensus" that we agreed to take the next logical step. Our next goal is, therefore, to begin the evolutionary process of converting the minimal list from Table 1 into a SIGCOMM-sponsored supplement to the IEEE/ACM 2002 Computing Curriculum recommendations. Interested parties are encouraged to contact Shawn Ostermann at ostermann@cs.ohiou.edu to become involved in this process.

3. Lab-based courses

Traditionally, computer networks courses have not provided students with hands-on access to networking equipment and software. In fact, courses that expose students to actual network environments are still mostly absent in an undergraduate curriculum. While introductory networking courses have a tendency to teach networking concepts at a relatively abstract level, lab courses emphasize how networking concepts are applied in an operational network. The basic assumption for offering a lab-based course is that hands-on lab exercises lead to a deeper understanding of networking principles. Most lab courses are offered as a second course in computer networks. Alternatively, an introductory network course can be supplemented with lab exercises.

The lab panel at the workshop surveyed different approaches that can be taken in developing "hands-on" laboratory-based courses at the undergraduate and graduate level. The lab panel consisted of designers and/or instructors of lab-based courses, and included Ann Burroughs (Humboldt State University), Magda El Zarki (University of California at Irvine), Doug Comer (Purdue University), Michael Williams (University of Akron), and Nick McKeown (Stanford University). The panel was chaired by Jörg Liebeherr (University of Virginia).

Ann Burroughs shared her experience with setting up a network teaching lab at Humboldt State University. The Computing Science Department at Humboldt State offers two undergraduate courses: a breadth-oriented course (Telecommunications) which is a core requirement, and a depth-oriented elected course (Network Design and Implementation). Ann Burroughs' networking lab is integrated into the Telecommunications course. The lab equipment at Humboldt State was established with an equipment donation of Cisco 7000 routers obtained through the NSF-supported Internet Teaching Lab project [CAIDA], which distributed sets of Cisco 7000 routers to over 25 educational institutions for the purpose of setting up teaching labs. Ann Burroughs discussed issues involved in integrating a lab component into an existing lecture course.

Doug Comer's Xinu lab was one of the first teaching labs for computer networks education. Established in 1984, the Xinu laboratory is used for research and instruction in operating systems and networks, and includes 20 workstations, 24 front-end workstations on Gigabit Ethernet, several back-end systems, and more than 20 network processors, as well as IP routers and switching equipment. Doug Comer gave examples of lab experiments for undergraduate and graduate students in the Xinu Lab. At the undergraduate level, lab exercises include network programming, where students build a client and server application and a concurrent web server, measurement experiments, where students compare the throughput of Ethernet hubs and switches, and protocol analysis experiments, where students study IP fragmentation and trace TCP connections. As examples of graduate-level lab exercises, students design and implement a software-based IP router with certain advanced features, such as multicast, NAT, or SNMP. Other lab exercises ask students to design and implement an IPsec box on a network processor platform, and a voice service over IP.

Doug Comer discussed how a lab course fits into a networking curriculum. An undergraduate course in computer networks provides breadth and exposes students to the concepts and the terminology of networking. The outcome of such a course is that students can state the purpose and function of hardware and software components of a network and know the role of communication protocols. In addition, students learn to write programs using a computer network. A graduate curriculum in networking strives for complete mastery of the subject, that is, understanding the design and implementation of protocols, being able to build correct and efficient system components, knowing how to architect large-scale networks, and being able to discuss tradeoffs and limitations of computer networks.

Doug Comer emphasized that lab courses are absolutely essential for a networking curriculum. Labs permit students to "learn by doing." Labs also reinforce concepts that are presented in a lecture and provide a concrete understanding of details. Doug Comer pointed out that the equipment in a lab does not need to use the most recent or fastest technology, and that a lab course with old equipment should always be preferred to not offering a lab course.

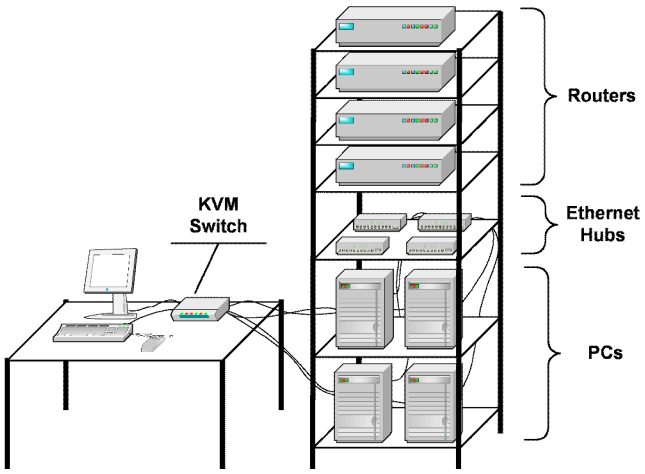


Figure 1: Open Lab Approach

Magda El Zarki discussed the design of a lab course with an *open lab* approach, where *open lab* refers to the fact that the lab equipment is located in a public area and that students perform lab experiments without supervision. The equipment for the open lab networking course is shown in Figure 1. It consists of four Linux PCs, four routers, and four Ethernet hubs. The PCs and routers are controlled from a single keyboard and monitor which is attached to a Keyboard-Video-Monitor (KVM) switch. The lab equipment is not connected to the Internet. An advantage of a rack-based lab is that the lab equipment can be easily duplicated. Currently, UC Irvine has five of the racks shown in Figure 1, and additional racks are being added.

Using the open lab approach, an instructor can manage a lab course with 50 - 70 students with only one teaching assistant and one grader. The lab course is taught every quarter to senior graduate students and first year graduate students. In the lab course, students complete eight labs over a period of 10 weeks. The lab topics include single segment networks, static routing, routing protocols, LAN switching, TCP and UDP, multicast, NAT, DHCP, DNS, and SNMP. The lecture component of the lab course consists of one three-hour long lecture per week. Each lecture gives an overview of the topics of a specific lab and demonstrates some of the experiments on equipment that is located in the lecture room. Each lab is structured in three phases. In the first phase, the "prelab", students read material and answer prelab questions which prepare them for the lab exercises. In the second phase, the "lab exercises", students work on the lab equipment, following step-by-step instructions given in a lab manual. Lab experiments consist of traffic measurements with a protocol analyzer tool. In the third phase, the "postlab report", students analyze the data that was gathered in the lab and prepare a report.

Michael Williams gave a presentation on the participation of the University of Akron in the Cisco Networking Academy program. The Cisco academy is a partnership program of Cisco Systems with educational, business, government, and other organizations, that evolved from a program to support a network curriculum at high schools. Cisco academy courses are offered as a combination of hands-on lab exercises and online study programs. Colleges and universities participate in the program by providing space, purchasing equipment for lab exercises, and providing personnel for teaching and supervision. The teaching material is supplied by Cisco Systems. The Cisco academy generally offers two certification programs, the Cisco Certified Network Associate (CCNA) and the Cisco Certified Network Professional (CCNP). At the University of Akron, the CCNA and CCNP program each consist of four eight-week courses, which are designed to be completed in two semesters each. Students enroll in the courses as for any other credit course. The courses can be applied to some bachelor's and associate degree programs at the University of Akron.

In his presentation, Nick McKeown gave an overview of his network infrastructure labs. The goal of the labs is to have students design, implement, deploy and debug their own infrastructure elements, such as IP routers, Ethernet switches, and elements of their own creation. The networking labs developed by Nick McKeown have a hardware component and a software component. NetFPGA [NetFPGA] is a hardware platform, which consists of a circuit board with eight Ethernet interfaces and user-programmable FPGAs. Students use these boards to architect, design and deploy their own hardware in an operational network. For example, students implement an Ethernet switch, an

IP router, or a firewall. The design process follows the industry standard flow. That is, students start with an implementation on a Verilog platform, followed by a simulation and verification phase, and a synthesis phase. At the end of the design process, the software is downloaded to the circuit boards and tested on a network that is connected to the campus network. A prototype of the NetFPGA platform has been developed in 2002, and classroom use is planned for 2003.

The software component of Nick McKeown's lab is called *virtual router* [Virtual Router]. The virtual router labs are intended for large networking classes with more than one hundred students, where it is not practical to provide each student with a dedicated computer with kernel-level access. In the virtual router lab exercises, students architect, design and deploy an IP router as a user-level system. Virtual router clients can be interconnected to form a virtual network topology for routing IP packets. The virtual router implementations have been used since 2001, and a release of the platform to other institutions is planned for summer 2003.

In the breakout session, which was led by Shiv Kalyanaraman (Rensselaer Polytechnic Institute), workshop participants identified different styles of labs that are being offered at various institutions. One broad group of lab exercises focuses on network programming. Some programming exercises focus on socket programming exercises, where students build application-layer services. Other programming exercises ask students to build network components, generally involving kernel-level programming or special hardware, such as FPGAs or network processors. Another group of lab exercises focuses on configuration and measurements of networking equipment. These labs try to find a balance between teaching an understanding of networking hardware and software and vendor-specific configuration skills.

There are numerous network simulation tools available that can be used in a lab course (ns-2 [NS], Opnet [Opnet], SSFnet [SSFnet] and GloMoSim [GloMoSim]). Simulation packages seem to have advantages when teaching very large classes. Recently, several emulation environments have become available that offer the opportunity to conduct lab exercises in a mixed hardware and software environment. The X-Bone [X-Bone] software can be used to deploy network-level services by tunneling IP traffic between X-bone capable systems. Some emulation platforms (Entrapid [Huang 1999], Vmware [Vmware]) can execute multiple operating system images, and can emulate a network with multiple routers and hosts on a single workstation. Emulab [Emulab] is a network emulator at the University of Utah that consists of several hundred PCs in racks that can be remotely configured.

The discussions in the panel and the breakout sessions showed a broad spectrum of lab courses that are used in computer networks education. A comparison of the different approaches (programming, configuration, measurement, and simulation/emulation) made clear that each approach has a distinct set of advantages for teaching certain aspects of computer networks.

The fact that most of the lab courses discussed at the workshop were introduced only recently points to a trend of faculty beginning to introduce labs into the networking curriculum. Instructors commented on the significant student interest in lab courses, and the generally overwhelmingly positive feedback. A problem with creating a new lab course is the considerable initial time commitment by instructors. Thus, the community could greatly benefit from a repository of existing computer networks lab courses that allow instructors to take advantage of earlier efforts at other institutions.

4. Graduate Curriculum

Just as schools are experiencing changes within their undergraduate networking curricula, so too are they experiencing change at the graduate level. These changes are being fueled by increased student interest in networking at both the undergraduate and graduate level, as well as by developments within the field itself. With more undergraduate students taking an introductory networking course, workshop participants noted that more students are arriving to graduate school ready to take advanced networking courses. Workshop participants also noted a marked increase in the number of graduate students interested in network-related courses. The increase in student preparedness, the increase in interest, and the ever-increasing growth in the scope of the field itself has resulted in tremendous demand for a larger and richer selection of graduate-level courses. Several schools have responded to this demand by creating MS and PhD level programs with specializations in networking and/or telecommunications.

The goal of the session on the graduate curricula was to survey various schools' approaches towards defining a graduate networking curricula. A number of questions were posed to the panel (and later discussed during the breakout session): *(i)* to what extent are the first graduate courses being taught an accelerated first (introductory)

course in networking versus an advanced course that would build on a prerequisite introductory course? *(ii)* is there an emerging set of *graduate-level* "core" material, that might build on undergraduate core material, such as that listed in Table 1? *(iii)* to what extent do graduate level courses focus on theory versus practice, *(iv)* what are expectations of industry regarding graduate-level courses, and *(v)* what is being done at your school?

The graduate curriculum panel participants were Ken Calvert (U. Kentucky), Scott Jordan (UC Irvine), Raj Yavatkar (Intel), and Ty Znati (U. Pittsburgh, and Program Director in the NSF CISE Division of Advanced Network Infrastructure and Research). The panel was chaired by Jim Kurose (University of Massachusetts).

Ken Calvert began the graduate curricula panel by describing the first networking course at the graduate level at the University of Kentucky. The course is an advanced introductory course that does not assume a previous course in networking. Topics covered include many of those listed in Table 1: channels, bandwidth and coding; framing, errors and ARQ; routing; transport protocols; queueing models; congestion control and avoidance; QoS; MAC protocols; application protocols (HTTP and SMTP); digital media, and the future Internet. The course emphasizes principles – important topics that have a more-than-5-year half-life, and illustrates these principles with real-world examples. There is a significant programming component to the course.

One important aspect of an introductory graduate-level course noted by Calvert (and seconded by many others at the workshop) was the varying degree of background and preparedness among incoming students. While some students have had no previous exposure to networking, other students may have already had an introduction to networking at the undergraduate level. Backgrounds even varied among those who have had previous exposure to networking – some students have had a more theoretically-oriented first course, while others have had more of a standards-based course, while yet others have had primarily a programming-oriented course. Calvert suggested that defining a topical interface between an undergraduate course, and a subsequent first graduate-level course would be a good first step in solving this problem.

Scott Jordan of the University of California at Irvine (UCI) was the next panelist. He described the proposed Masters and PhD program in "Networked Systems" at UCI. The new program is a collaborative effort between the Department of Electrical and Computer Engineering, and the Department of Information and Computer Science. The program defines three core courses (Computer Networks, Computer Network Laboratory, Network Systems Seminar); five concentration areas (networks, performance, middleware, communications, and operations research), each with a number of courses; and two breadth areas (computer systems engineering, and management of technology).

Ty Znati from the University of Pittsburgh, also currently a Program Director in the Advanced Networking Infrastructure and Research Division at the National Science Foundation, then described the Telecommunications Ph.D. program at the University of Pittsburgh. The degree is awarded as a Ph.D. in Information Science with a Telecommunications concentration. Like the proposed PhD program at UCI, the Pitt program clusters graduate courses into different areas: networking, communication systems, computer communications, telecommunications administration, telecommunications economics and policy, human communications, and wireless communications. The program offers a 24-credit certificate program, a 36-credit masters degree, and the PhD degree.

Raj Yavatkar from Intel provided an industry perspective on graduate education. He began by noting two particular deficiencies that he often finds in incoming students: that they have little hands-on experience, and not enough of an understanding of *system* design and architecture. He then identified the background that he would like to see in all graduate-level students: the basics of efficient protocol design, security as a first class object, performance analysis and evaluation, and system design and architecture. Raj Yavatkar then outlined the contents of two graduate-level courses that would prepare students well for industry positions. He noted that these two courses built on an assumed undergraduate course in networking, and emphasized that one hands-on project and one simulation-oriented project were a critically important component of such courses. The Intel IXA network processor program was noted as an example of how projects have (and could be) integrated into graduate level courses.

While the undergraduate breakout session was able to find a rough consensus on core topics that might be taught in a first undergraduate networking course, no such consensus was reached in the graduate curriculum discussion. The one common theme that did emerge was the recognized need to teach students advanced foundational material that "has a long shelf life" (i.e., would be useful to students for many years to come), and much of the discussion centered on what specific topics should be taught in such a course. Performance evaluation (modeling, simulation, and experimental design), a "science" of protocol design, and large-scale system building were topics that many felt should be taught in such a graduate "core" course.

5. Conclusions

While a one-day workshop can only begin to address the many issues facing networking educators, the general consensus was that the workshop had been an extremely valuable forum for learning about what others are doing, exchanging ideas, and promoting thought-provoking discussion. A number of resources and ongoing activities have resulted from the workshop, and can be accessed via the ACM SIGCOMM Education web page (http://www.acm.org/sigcomm/education):

- A listing of networking courses is being maintained by Maurice Aburdene at http://www.eg.bucknell.edu/~aburdene/networkcourses/. In addition, a list of courses; has been maintained by the Internet Engineering Curriculum group at CAIDA: http://www.caida.org/outreach/iec/courses/;
- A listing of lab-related courses and resources is being maintained by Sanjay Jha at http://www.cse.unsw.edu.au/~sjha/netlab.html;
- A mailing list for people interested in discussing issues related to networking education: sigcomm_education@cs.umass.edu;
- A compendium of white papers submitted in advance of this workshop to stimulate thought and discussion;
- Organizational planning for a follow-up workshop.

More information about these activities can be found at http://www.acm.org/sigcomm/education. All members of the networking community are invited to become involved in these efforts.

Acknowledgements

Travel grants, awarded to selected workshop participants, were made possible with support from the Division of Advanced Networking Infrastructure and Research at the National Science Foundation, under award ANIR-0226866**.**

References

[CAIDA] ITL – Internet Teaching Laboratories, http://www.caida.org/outreach/itl

[Emulab] netlab – Based on emulab, http://www.emulab.net

[GloMoSim] GloMoSim – Global Mobile Information Systems Simulation Library, http://pcl.cs.ucla.edu/projects/glomosim/

[Huang 1999] X.W. Huang, R. Sharma, and S. Keshav, The ENTRAPID Protocol Development Environment, Proc. IEEE Infocom '99, March 1999. http://www.cs.cornell.edu/skeshav/papers/entrapid.pdf

[IEEE/ACM 2001] CC-2001 Task Force, Computing Curriculum (Final Draft), http://www.computer.org/education/cc2001/final

[NetFPGA] The NetFPGA Project, http://klamath.stanford.edu/NetFPGA/

[NS] The Network Simulator – ns2, http://www.isi.edu/nsnam/ns/

[OPNET] OPNET homepage, http://www.opnet.com

[SSFnet] The Scalable Simulation Framework, http://www.ssfnet.org

[Virtual Router] The Virtual Router Project, http://klamath.stanford.edu/vr/vr.html

[Vmware] The vmware Homepage, http://www.vmware.com/

[X-Bone] The X-Bone – A System for Automated Overlay Network Deployment, http://www.isi.edu/xbone

ATM: A Retrospective on Systems Legacy
or
"A technology with a fabulous future behind it?"

Jon Crowcroft & Derek McAuley
jon.crowcroft@cl.cam.ac.uk, derek.mcauley@intel.com

Abstract

The following four papers were selected from submissions for a proposed workshop that was to have been held during the 2002 ACM SIGCOMM Conference. Due to time, we cancelled the event, but the papers capture some of the past, present and future lessons to be gleaned from the whole ATM experience, and we felt that these lessons should be provided with a forum.

Broadband ISDN is not a very catchy phrase; nor is *Asynchronous Transfer Mode*. However, networking research, development and standards groups in the last two decades of the 20th century were alive with a debate over the merits of these technologies. At the same time, the Internet grew from a research baby to a commercial adult. In some places, there was tension, in others, harmony, between the work carried out on ATM and on IP.

Last year, some folks in the SIGCOMM community felt that it would be good to have a retrospective look at ATM, so that we could pull out some lessons for the future. The broad remit we felt we had was to examine the political, technical and economic pressures that bore on networking during this period of very rapid innovation and growth.

The four papers here cover four aspects of ATM and we've chosen them to span the problems and the solutions, as well as the past and the future:

The Technology Charles Kalmanek, from AT&T, describes the actual technology behind ATM, including the range of different visions that different networking communities had. He takes us from the early days of a unified replacement for the SDH/SONET system that the telcos required, through to today's reality of widespread use of ATM as a layer 2 technology in much of the broadband access networks (DSL) and in some ISP's and corporate network cores where hard QoS is required.

Performance Dan Grossman, from Motorola, takes a look at the performance trade-offs inherent in the architectural technology choices of Cell Switching, and the Virtual Channel style of network service. The goals of low latency for voice, and of QoS assurance proved to have perhaps unexpected costs.

Control Planes Simon Crosby, Sean Rooney, Rebecca Isaacs and Herbert Bos, from Cplane, IBM, Microsoft and LIACS respectively, examine signalling systems in the most general sense of the term. They also include an informal analysis of the business case for end-to-end ATM services versus IP.

End Systems Jonathan Smith, from the University of Pennsylvania, shows how many of the ideas that ATM forced us to revisit in operating systems' networking support have re-emerged as the performance curve gets to the point where the time to service an IP packet is similar to

that of ATM cells in older systems. This is one of the more important technical lessons that can be generalised in the old adage: "What goes around comes around".

Between the lines you may read many things in these four contributions. We don't want to spoil things by spelling all of them out, but the way that different Fora develop standards is an important factor in the breadth of the success of those standards. An open approach has the advantages of being susceptible to close technical, but also business case analysis by the broader community. We should not neglect the role and influence of government funding in growing technologies too. Thus socio-economic comparisons between the 3G and ATM communities are inevitable, as are comparisons between the government research agencies' effect in funding say Active Networks, Grid and Peer-To-Peer, with the 1990s funding of B-ISDN in Europe and the US.

Enjoy.

A Retrospective View of ATM

Charles Kalmanek
AT&T Labs Research
180 Park Avenue, Room A113
Florham Park, NJ 07932
crk@research.att.com

ABSTRACT
ATM was the focus of active research and significant investment in the early to mid 1990's. This paper discusses several visions for ATM prevalent at the time, and analyzes how ATM evolved during this period. The paper also considers the implications of this history for current connection-oriented technologies, such as optical transport networks and MPLS.

Keywords
ATM, transport networks, flow switching, MPLS.

1. INTRODUCTION
Asynchronous Transfer Mode (ATM) networking had its origins as a switching and multiplexing technology suitable for the design of high capacity switches. The essential features of ATM are a fixed-length packet (called a *cell*), which is switched based on a virtual circuit identifier in the cell header. End-hosts request that the network set up a virtual circuit via a signaling (control) protocol that allows them to specify the desired quality of service. Quality of service per virtual circuit is provided through admission control and switch scheduling algorithms, allowing delay-constrained traffic, such as voice and circuit-emulated TDM traffic, to share a single network infrastructure with bursty data traffic. The cell size was kept small to support low delay for voice (although introducing enough delay that echo cancellation is needed.)

For a period of time in the early to mid 1990's, investment and research on ATM exploded, based on an expectation that ATM would revolutionize networking. For telecom providers, ATM promised to unify a number of disparate networks (voice, private line, data) on a single switching network. The fixed cell size fit well with designs for large self-routing switch fabrics suitable for the construction of very high-capacity switches. ATM's proponents anticipated that ATM would be ubiquitous, and that end-to-end quality of service would enable an entirely new class of network applications to be built.

The reality today is far different. ATM is used today to provide Virtual Private Network (VPN) services to businesses, consisting primarily of point-to-point virtual circuits connecting customer sites. ATM services represented a $ 2B business in 2001. ATM also provides the underpinnings of Digital Subscriber Loop (DSL) services, which are growing rapidly. In DSL access networks, ATM enables local exchange carriers to switch subscriber traffic to different Internet Service Providers. ATM is also used as the core network infrastructure for large Frame Relay networks and for some IP networks. While these uses of ATM are important and should be viewed as a mark of success for ATM technology, there is a perception in the network research community that ATM "failed." Indeed, when compared with the grandiose visions that many of its proponents had, ATM was not as successful as it might have been. This paper explores some of the visions for ATM that were pursued both by telecommunications service providers and the research community in the early to mid 1990's and presents some of the technical and business issues that drove the evolution of ATM.

2. MANY VISIONS FOR ATM
Starting from a small set of initial design principles, the development of ATM technology progressed in a number of different directions, based on the business and technical visions of the companies and individuals who were driving the technology. This section summarizes several of the early visions for ATM prevalent in the research community and industry. While attempting to give a balanced overview of the work that was going on, I have no illusions that this survey is exhaustive or that is does justice to any one of these visions. However, I hope that it gives some notion of the tremendous scope that the ATM community was trying to address.

One vision of ATM's role in telecom networks was that it would provide a *single multi-service network*. I distinguish between two variants of this vision. One is that it would serve as a multi-service core network supporting primarily data services such as Frame Relay, IP, and ATM service, possibly with some DS1, DS3 or higher rate private line and voice services. The other is that it would eventually replace the circuit-switched TDM hierarchy and provide a *next-generation transport network*, supporting long-term capacity (bandwidth) management for all services. In the first instance, an ATM core network would support multiple service edges, such as frame relay, IP, ATM, and possibly private line and voice. A high level view of this architecture is shown in Figure 1, which gives an example of multiple networks, each with its own "edge" switches connected over dedicated links. Figure 2 utilizes a single core network to connect edge switches for each service. This approach optimizes link utilization through statistical multiplexing of edge-to-edge traffic over the core network. It was also understood that the introduction of hierarchy could improve the scalability of the network from a routing perspective. Rather than scaling independent networks as the number of customers grew, the networks within each region could be scaled independently, and interconnected over a core network running the ATM PNNI routing and signaling protocol. The edge-core approach also had the potential to reduce network operations expense through the consolidation of individual service networks.

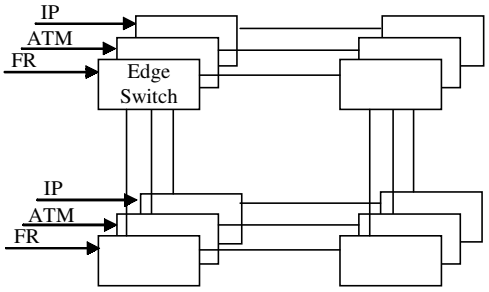


Figure 1: Multiple Edge Networks

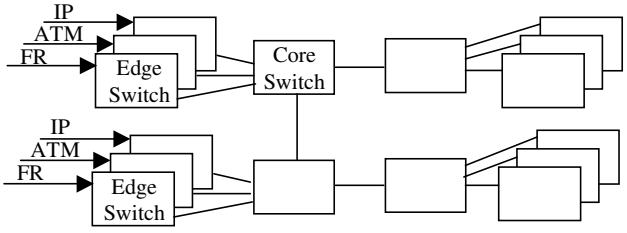


Figure 2: Edge-Core Architecture

To consider ATM as a next-generation transport network, we first need to understand how transport networks are built. Transport networks have traditionally been based on a hierarchy of time-division multiplex (TDM) switches. Figure 3 presents typical network architecture, simplified from [1]. DS1 private line or voice trunks are aggregated and/or switched in a Digital Cross-connect System with DS3 interfaces and a switching granularity of DS1, denoted as a DCS-3/1. DS3 private line or aggregated traffic demands are switched in a DCS-3/3 supporting DS3 interfaces and a switching granularity of DS3. These DS3s and other higher rate signals are carried over SONET rings[1] or linear chains that provide restoration in case of link or interface failure. SONET links are transported between central offices by an Optical Transmission System (OTS). Due to the existence of multiple layers of network hierarchy, this network architecture often results in inefficient overall network utilization, for a number of reasons. One is that the partial utilization at each network layer is compounded as you go up the hierarchy. Another reason is that circuits at a given layer may end up being routed inefficiently as a result of capacity planning processes that are designed to maximize the utilization of the layer that carries them. For example, a DS1 circuit may not follow the shortest path between two central offices because it is routed over a pre-existing DS3 circuit following a less efficient path.

From its earliest design, ATM was intended to support virtual circuits across a wide range of rates. With the promise of ATM switches with aggregate capacities of 10's of Gbps in the mid 1990's growing to 100's of Gbps by the end of the decade, network designers began to seriously consider using ATM in the transport layer, supporting multiple services and consolidating several layers of the TDM hierarchy. Figure 4 illustrates a network architecture in which all transport bandwidth management below OC-48 is done at the ATM layer. In the figure, ATM and IP layer demands are carried over an optical cross-connect (OXC) layer, which takes on the restoration function supported in the SONET layer in Figure 3. Since bandwidth management on the ATM network occurs on a slow time scale, ATM VC setup only occurs on provisioning time scales. By eliminating the hierarchical transport network architecture below OC-48, ATM promised to simplify the task of managing the transport network and to improve overall network efficiency.

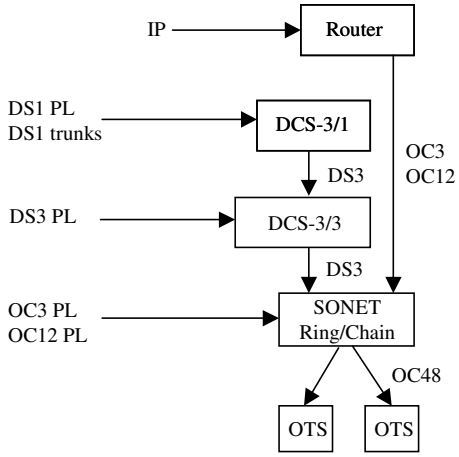


Figure 3: Traditional Transport Network

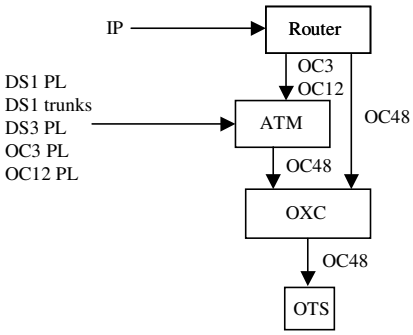


Figure 4: ATM-based Transport Network

A second vision of ATM, fostered by the research community, envisioned the use of ATM as a universal end-to-end packet service [2]. ATM's emphasis on end-to-end quality-of-service was an important part of this vision. In the end-to-end vision, desktop computers, networked appliances and large servers would all support ATM, and set up end-to-end connections with quality-of-service when needed. There was a tremendous amount of work on ATM host interfaces, low cost ATM interface chips, and ATM LAN switches. Protocol stacks were developed for end-hosts by extending the Berkeley socket layer to allow applications to directly establish ATM virtual circuits [3]. ATM's small cell size seemed particularly well suited to managing quality-of-service in access networks, where bandwidth is scarce, such as DSL and wireless networks [4], etc. Small cells implied that links could be

[1] While SONET ring technology does not scale very well to large networks, large SONET cross-connects were not yet available in the early to mid 1990's.

scheduled on a fine time scale, allowing delay-sensitive applications to be supported alongside elastic applications.

There was also a significant amount of research on ATM signaling. In addition to ATM standards, ATM signaling was adapted to support connection handoff for mobile wireless devices [5]. A lightweight ATM signaling protocol was proposed in [6] which established forwarding state for a connection very efficiently, while allowing additional signaling along the slow path to establish QoS for the connection. The "open signaling" community proposed that ATM switches should support a standard low-level control protocol [25], allowing service providers to customize their ATM control plane. The research community also pursued even more radical ideas, such as desk-area networks [7], in which ATM was used as the fabric in a desk-area distributed computing environment comprising processing resources, I/O devices, and storage. As a result of the investment by the vendor community, leading edge products were available causing some enterprises to start to use ATM in their server environment, and to gear up to push ATM to the desktop.

The above vision of ATM was of interest to ATM purists. At the same time, there began to be significant interest in IP flow switching concepts [8] that promised to better integrate ATM with IP. If one looks at IP traffic, a significant fraction of the "flows" are small transactions, such as DNS lookups, for which the overhead of ATM connection setup is on par with or larger than the duration of the transaction. Rather than setting up a connection for short transactions, IP flow switching proposed to setup ATM *shortcut* connections only for long-lived flows [9, 10], offloading slow IP routers and leveraging high-performance ATM switches. The basic idea is illustrated in Figure 5, which shows "default" connectivity via ATM permanent virtual circuits (PVCs) using solid lines, and a shortcut connection that has been set up dynamically between two routers using a dashed line. The flow switching architecture proposed to enhance ATM switches by putting smart algorithms for detecting IP "flows" on ATM line cards. Service providers began to investigate how they could use ATM flow switching as a way of more tightly integrating the IP layer with the ATM layer than was envisioned in the architecture shown in Figure 2.

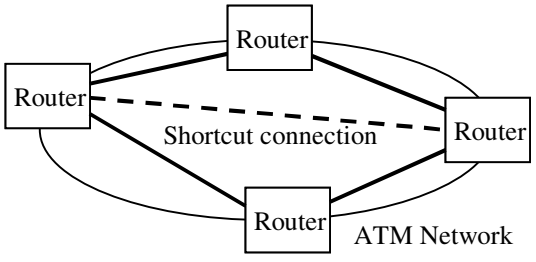


Figure 5: IP Flow Switching

The flow switching concepts were further developed in both the IETF and the ATM Forum. Using the Next-Hop Resolution Protocol (NHRP) [11], a router queries a Next-Hop Server to determine the ATM address of the next IP hop towards an IP destination. An alternative approach [10] utilized extensions to IP routing to carry the ATM address of the next-hop router. In addition, the Multicast Address Resolution Server (MARS) architecture [12] addressed the problem of mapping IP multicast forwarding onto ATM's connection-oriented services. MARS is based on point-to-multipoint VC's and uses either VC meshes or multicast servers to support the IP multicast service. A Multicast Address Resolution Server maintains a mapping from IP address to a set of ATM addresses in a Logical IP Subnet (LIS), and is updated by a host in the LIS when it joins or leaves an IP multicast group.

Another set of activities focused on enabling ATM to support the large embedded base of software built on bridged LANs. Initial implementations of LAN emulation on ATM were available around 1994, and the ATM Forum developed the LAN Emulation (LANE) specification [13]. LANE provides transparent support for Layer 2 Ethernet bridging services: the ATM LAN emulation protocol stack is shown in Figure 6. LANE's primary goal was to support common end-host drivers, such as Network Driver Interface Specification (NDIS) from Microsoft, which runs over bridged LANs. The function of the LAN emulation layer was to exactly mimic the MAC layer interface of a bridged LAN, so the higher layers would think they were running over a standard Ethernet or token ring network. An ATM LAN bridge would support fragmentation of MAC frames into ATM cells, emulating all of the functions supported by LAN bridges on top of ATM and inter-working with existing bridged networks.

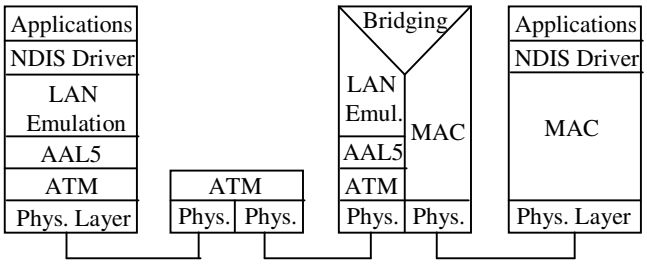


Figure 6: LAN Emulation

None of these visions of how ATM would evolve proved to be correct. The question is why?

3. HOW IT PLAYED OUT

The vision of end-to-end ATM with quality-of-service was extremely compelling to many people. Internet service was widely seen to be unpredictable, and ATM promised a solution. The ATM Forum developed a comprehensive Traffic Management framework, supporting five classes of service: CBR, VBR-rt, VBR-nrt, ABR and UBR[2]. This framework developed many of the key traffic management concepts that are in common use today: traffic descriptors, shapers, policers, priority and weighted fair scheduling, as well as signaling support for connection admission control and QoS routing. CBR defines mechanisms to support delay-constrained constant bit rate traffic, while ABR supports network feedback to sources allowing them to adapt their sending rate to the max-min fair rate of the bottleneck link along the path of the connection [14, 15]. This

[2] CBR = constant bit rate; VBR-rt = Variable Bit Rate (real time); VBR-nrt = Variable Bit Rate (non real time); ABR = Available Bit Rate; UBR = Unspecified Bit Rate.

work contributed a significant number of innovations to packet networking.

However, end-to-end ATM faced a deployment challenge due to the law of network externalities. Until ATM deployment reached a critical mass, end-to-end ATM QoS couldn't be realized. Since most existing applications were based on IP, use of ATM QoS for IP applications would need to be mediated by a (non-existent) IP QoS application programming interface. Moreover, ATM deployments would bear the burden of inter-working with existing applications and hosts that had not been upgraded to ATM. Another factor limiting the realization of end-to-end QoS is that, despite the variability of Internet performance, new network technologies are often initially deployed in enterprises. Here, the application drivers for end-to-end QoS never really materialized. And in this environment, high-speed Ethernet-based LANs began to dominate the desktop thereby reducing the perceived need for ATM's traffic management framework.

Nonetheless, the momentum behind ATM deployment was strong enough that for a period of time, it seemed possible that ATM to the desktop might succeed. Economic forces worked against it, however. In the early 1990's, an ATM host adaptor cost roughly $3K. By the mid 1990's, this price had fallen to roughly $1K. At that time, Ethernet adaptors cost about $100. This price difference was a significant impediment to widespread adoption of ATM.

We can also consider the vision of ATM as a future core data or transport network. Since ATM provides a flexible bandwidth management capability, it seemed very well suited to the role of a multi-service core network or even core transport network, replacing SONET rings with a more efficient mesh structured bandwidth management layer. Unfortunately, there were a number of factors that made it difficult to realize this vision. One factor is that growth for data services through the mid 1990's dramatically exceeded expectations. This growth included Frame Relay, IP, ATM and private line services. When the total Frame Relay, ATM, IP and DS1 private line demands were considered, the switch capacities of commercially available ATM switches were not adequate to support the requirements of large central offices.

Another issue is that each service emphasized a slightly different set of requirements, such that the union of the service requirements was difficult for vendors and service providers to cope with. For example, private line has stringent requirements on reliability and restoration capabilities, and requires ATM line card support for TDM circuit emulation; voice has requirements for high switched virtual circuit (SVC) setup rates; and ATM PVC and SVC services for data likely require high port densities and reasonable reliability, at a significantly different target for price/performance than either voice or private line. If we focus specifically on transport network evolution, switch reliability was an issue. Digital Cross-connect Systems typically are designed to meet less than 3 minutes per year of downtime: relatively immature ATM switches could not meet this requirement. One way of dealing with some of these problems would have been to select different vendors for the edge and core switch nodes. However, due to relatively slow progress on ATM signaling standards, vendor inter-operability was delayed. As a result of these considerations, service providers gave up on the idea of a single core network, and took the more conservative approach of evolving separate IP, transport, and ATM networks. The one exception was Frame Relay networks, which evolved to run over ATM core networks since higher speed ATM interfaces and switch capacities were needed to support growing Frame Relay demand.

It is also important to recognize that ATM was being standardized and deployed just as IP was beginning to pick up steam. To be successful, ATM clearly needed to provide some inherent advantages (and few disadvantages) in carrying IP traffic. To some, ATM's high-speed and the emerging flow switching technologies seemed at the time like a winning combination. However, flow switching actually introduced uncertainty in how the IP layer would best utilize ATM. This uncertainty and the complexity of the underlying technical issues may have slowed rather than accelerated ATM deployments in large-scale data networks. Standardization in the IETF and ATM Forum's Multi-Protocol over ATM (MPOA) group [24] seemed likely to take a long time. In addition, flow switching introduced another layer of complexity into the architecture, requiring vendors to understand and be competitive in ATM switching, IP routing, *and* IP flow switching. To utilize flow switching effectively, service providers would need to provision IP flow detection algorithms on edge switches. Before developing software tools to do this, there needed to be strong evidence that flow switching would provide either significant cost savings or end-user performance improvements. Demonstrating cost savings in a large service provider network would depend on a combination of factors, including the ratio of router interface costs to ATM interface costs, the amount of traffic that would actually utilize shortcuts, etc. Demonstrating end-user performance improvements would depend on being able to deliver shortcut traffic with better end-to-end performance (e.g., throughput, delay) than routed traffic. While this might have been possible, the improvement would have to be significant to justify a new architecture.

Another issue with IP over ATM was support for IP multicast. The IP multicast community was extremely vocal about both the need for IP multicast and the difficulties of supporting it in ATM. As a result, ATM standards evolved to include support for point-to-multipoint unidirectional virtual circuits, and [12] defined a set of mechanisms to support IP layer multicast using them. An alternative multicast model using core-based trees was defined in [16]. Nonetheless, the debate about IP multicast over ATM contributed to the uncertainty about ATM, despite the fact that IP multicast has *still* not been widely deployed.

While these issues were being played out in the marketplace, there was also significant investment in IP router technology, due to the tremendous growth in IP traffic demands. Improvements in silicon technology and algorithms for IP forwarding table lookups [17, 18] resulted in a situation where commercially available ATM switches did not offer any speed advantage over commercially available IP routers. For a multiplexing layer to make sense, it needs to offer some speed advantage over the demands that it will be carrying. In fact, router vendors began to use ATM switch technology to increase their aggregate switching capacity, and interface rates on ATM switches often lagged behind those of IP routers.

It is also useful to consider the evolution of link technologies and their impact on ATM. Around 1992, when work on ATM was getting started, 100 Mbps shared FDDI rings were the fastest

switching technology in widespread use. There was no other cost effective high-speed link technology for the LAN – Fast Ethernet was not yet available. ATM took advantage of the newly developed Fibre channel line coding chips, which promised relatively inexpensive 155 Mbps LAN links. In the WAN, ATM SONET interfaces at 155 Mbps and 622 Mbps were commonly used as the interface between backbone routers. Then, in 1994 - 1995 Fast Ethernet came out, and around 1997, Gigabit Ethernet appeared. In 1997, Packet over Sonet (POS) also appeared, supporting high-speed IP over HDLC over SONET without the overheads of ATM. When compared with IP router technology, ATM had a number of initial advantages: high capacity switching, high-speed links, etc., but lost advantage after advantage as IP router technologies advanced.

The end result was that ATM never reached critical mass for going to the desktop, given the cost of network adaptors, network externalities and the existence of competing Ethernet technology. ATM was also not ready to support the needs of a multi-service core network or core transport network. Finally, proposals to integrate ATM and IP, such as IP flow switching, were complex and were ultimately superceded by advances in IP router design, which incorporated many of the innovations that had been developed in ATM.

It is clear that ATM fell short of the technological vision that many people had. Where ATM *succeeded* was as a Layer 2 switching technology that is used in access/aggregation networks and as a core network for Frame Relay and native ATM services. Layer 2 VPNs consisting of point-to-point PVC's provide high reliability connectivity services to enterprises at a lower price point than private line services. ATM is also widely deployed as part of DSL access networks. The latter application is shown in Figure 7, where traffic from a DSL Access Multiplexer is carried over ATM to an ISP. It is important to note that ATM networks today carry a significant amount of Frame Relay traffic, DSL traffic, both enterprise and carrier voice traffic, and some backbone IP traffic.

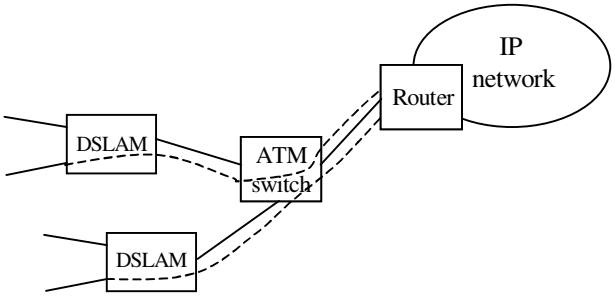


Figure 7: ATM-based DSL Access

ATM services continue to provide some technical advantages over IP services. ATM is more mature than IP in its ability to provide stringent quality of service guarantees. While IP differentiated services support class-based quality of service, IP differentiated services face a number of deployment challenges in large ISPs, including the difficulty getting good traffic data as an input to traffic engineering, and performance problems in legacy routers when quality of service features are enabled. In addition, ATM's guaranteed bandwidth on demand, fast re-route, and OAM features are important to many large customers. None of these features has been thoroughly integrated into IP as yet. Identifying and sectionalizing problems with end-to-end service in IP, even within a single ISP, is difficult. In time, MPLS may be able to support these features well, but ATM will be the technology of choice for some customers for many years to come.

4. LESSONS FOR TODAY

One natural question given this background is what lessons can be learned from ATM that might be applicable in the latest incarnations of connection-oriented technology: optical transport networks and MPLS. Large SONET cross-connects, typically with OC-48 or OC-192 ports and STS-1 switching granularity are now commercially available and are the basis of a new generation of transport infrastructure. These switches use variants of ATM's PNNI signaling or MPLS signaling (called Generalized MPLS) protocols [19] to set up and manage connections. For the foreseeable future, SONET technology appears to be the likely basis of the transport infrastructure, perhaps augmented by the use of transparent optical switching for managing large optical bit pipes at some point in the future.

For IP networks, MPLS is being touted as providing a routing and switching layer that can enable multiple types of traffic to share a common packet switched infrastructure. This sounds familiar, and it is worth understanding how MPLS is evolving in order to understand whether it will succeed where ATM failed. Like ATM, MPLS is a virtual circuit technology. In fact, MPLS has borrowed a number of the essential ideas of ATM: virtual circuit switching, fast re-route, and the notion of a single network infrastructure. However, MPLS does not attempt to solve an end-to-end problem, but rather focuses on a single administrative domain and is tightly integrated with the IP forwarding paradigm. To support quality of service, MPLS reuses IP differentiated services. MPLS does not support fast/dynamic connection setup like ATM SVC's.

One key application of MPLS is support for Layer 2 and Layer 3 virtual private networks (VPNs) on an MPLS label switched core network. As with ATM, the opportunity here is to reduce network operations expense through the consolidation of individual service networks. The IETF "Martini" encapsulation [20] allows frame relay, ATM, Ethernet, and IP packets to be transported over an MPLS network. An MPLS tunnel between ingress and egress label-switched routers (LSRs) can carry packets associated with different services – the egress LSR uses the innermost label to distinguish the service to which packets are to be de-multiplexed. This approach is designed to support ATM permanent virtual circuits, although it does have some deficiencies. For example, the Martini encapsulations do not support service inter-working such as between Frame Relay and ATM. In addition, the related signaling extensions [21] were not designed to support switched virtual circuit setup. However, another approach to ATM over MPLS provisions an overlay ATM network on MPLS tunnels, and runs PNNI among the overlay network edges. This approach allows switched virtual circuits to be supported. In addition to supporting ATM, Frame Relay and Ethernet "virtual circuits," MPLS protocols are also being developed to support Transparent (bridged) LAN service over MPLS [22].

Layer 3 VPNs are supported using extensions [23] of the IP Border Gateway Protocol (BGP) that provide support for private address spaces and virtual private IP networks on a shared MPLS core network. Given that many enterprises use private IP addressing and do not want mission-critical applications exposed

to the Internet, the isolation provided by Layer 3 VPNs based on MPLS are likely to be important, and will compete with Layer 2 and IPsec-based VPN's. Note also that the ability to run virtual private IP networks on a shared core network allows large Internet service providers to resell IP backbone network capacity. Since the cost structure of an ISP depends on economies of scale, this drives down costs. The RFC2547 approach may eventually change the way that ISPs handle Internet routing. While IP forwarding and MPLS label switching currently co-exist in core network routers, in time IP forwarding tables could essentially disappear from core routers in an MPLS enabled network – Internet default-free routes would only exist as one of the virtual routing and forwarding tables in an MPLS provider edge router.

While it is impossible to predict the future, it is clear that MPLS has avoided a number of the pitfalls that plagued earlier visions for ATM. First, MPLS is not attempting to provide an end-to-end solution -- it is clearly targeted at service provider core networks. As a result, MPLS doesn't need to be ubiquitous to be successful. Second, MPLS protocols are an extension of existing IP protocols, while ATM's control plane evolved to be quite complex, including signaling, routing, MPOA, LANE, etc – all of which were needed *in addition to* IP protocols. This may simplify development and deployment. Third, MPLS is riding the same technology curve as IP routers, which suggests that switch capacity will not put MPLS at a disadvantage relative to IP.

There are still interesting questions about the role of MPLS in ISP networks. For example, the cost and performance tradeoffs among restoration alternatives at different layers (IP layer rerouting, MPLS fast re-route, and transport network restoration) are an active area of research. In addition, as mentioned earlier, transport networks and ATM networks are traditionally more reliable than IP networks – in part because routing problems in one ISP's network can affect other providers. RFC2547 isolates routing in Layer 3 VPNs from Internet routing, which directly addresses this problem for VPN customers.

5. CONCLUSION

This paper surveys several of the visions for ATM explored by the networking community in the early to mid 1990's. Many of the traffic management concepts developed in ATM have become part of networking practice, and ATM is widely used to support VPN's, DSL access networks, and as a core networking technology for Frame Relay and some voice and IP services. Nonetheless, ATM did not succeed in revolutionizing networking. Economic factors, network externalities, the complexity of emerging standards and implementations, and the rapid development of alternative technologies were all factors which made it difficult for ATM to take over the world as many people expected.

From an historical perspective, the debate between connection-oriented and connectionless networking technologies has existed since the early days of packet switching. Even with the explosive growth in IP communications over the last decade, it appears that the tension between the two *technologies* is alive and well. Connection-oriented technologies are the basis of the optical transport networks that underlie most data networks below OC-48 rates, while MPLS is now becoming mature enough to support VPN services in large ISP backbones. It seems likely that connection-oriented technologies will continue to play a significant, if largely invisible, role in data networks at Layers 1.5 and 2 for some time.

6. ACKNOWLEDGMENTS

The author gratefully acknowledges the contributions of Tom Afferton, Elie Francis, Han Nguyen and K.K. Ramakrishnan, for their careful feedback on early drafts of this paper. K.K. Ramakrishnan provided the perspective on the evolution of datalink technologies and ATM.

7. REFERENCES

[1] R. D. Doverspike, S. Phillips, and J. R. Westbrook, *Future Transport Network Architectures,* IEEE Communications Magazine, August 1999.

[2] I. M. Leslie, D. R. McAuley, and D. L. Tennenhouse, *ATM Everywhere?*, IEEE Network Magazine, March 1993.

[3] R. Sharma and S. Keshav, *Signaling and Operating Systems Support for a Native ATM Applications*, Proc. SIGCOMM '94.

[4] J. Condon, et al., *Rednet: A Wireless ATM Local Area Network using Infrared Links*, Proc. MOBICOM '95.

[5] R. Yuan et al., *A Signaling and Control Architecture for Mobility Support in Wireless ATM Networks*, Mobile Networks and Applications, Special Issue on Wireless ATM, Vol. 1, Issue 3, December 1996.

[6] G. Hjalmtysson and K.K. Ramakrishnan, *UNITE: An Architecture for Lightweight Signaling in ATM Networks*, Proc. Infocom '98.

[7] M. Hayter and D. McAuley, *The Desk Area Network*, Operating Systems Review, Vol. 25, No. 4, pp. 14-21, 1991.

[8] P. Newman, G. Minshall, and T. Lyon, *IP Switching: ATM under IP*, IEEE/ACM Transactions on Networking, Vol. 6, No. 2, April 1998.

[9] A. Shaik, J. Rexford, and K. G. Shin, *Load-Sensitive Routing of Long-Lived Flows*, Proc. SIGCOMM '99.

[10] C.R. Kalmanek, A. Lauck, and K.K. Ramakrishnan, *SUBMARINE: An Architecture for IP Routing over Large NBMA Subnetworks*, Proc. INFOCOM '99.

[11] J. Luciani et al., *NBMA Next-Hop Resolution Protocol*, IETF Request for Comments 2332, April 1998.

[12] G. Armitage, *Support for Multicast over UNI 3.0/3.1 based ATM Networks*, IETF Request for Comments 2022, November 1996.

[13] *LAN Emulation over ATM Version 2.0 – LUNI Specification*, ATM Forum, AF-LANE-0084.000, 1997.

[14] A. Charny, K.K. Ramakrishnan, and T. Lauck, *Time Scale Analysis and Scalability Issues for Explicit Rate Allocation in ATM Networks*, IEEE/ATM Transactions on Networks, Vol. 4, No. 4, August 1996.

[15] *Traffic Management Specification Version 4.0*, ATM Forum, AF-TM-0056.00, 1996, available at:ftp://ftp.atmforum.com/pub/approved-specs/af-tm-0056.000.pdf.

[16] M. Grossglauser, K.K. Ramakrishnan, *SEAM: A Simple and Efficient Architecture for ATM Multipoint-to-Multipoint Communication*, Proc. Infocom'97.

[17] M. Waldvogel et al., *Scalable High-Speed IP Routing Lookups*, Proc. Sigcomm '97.

[18] M. Degermark et al., *Small Forwarding Tables for Fast Routing Lookups*, Proc. Sigcomm '97.

[19] E. Mannie (ed.), *Generalized Multi-protocol Label Switching (GMPLS) Architecture*, IETF Work in Progress.

[20] L. Martini et al., *Encapsulation Methods for Transport of [Ethernet frame, ATM Cells, etc.] over MPLS Networks*, IETF Work in Progress.

[21] L. Martini et al., *Transport of Layer 2 Frames over MPLS*, IETF Work in Progress.

[22] L. Andersson et al., *PPVPN L2 Framework*, IETF Work in Progress.

[23] E. Rosen and Y. Rekhter, *BGP/MPLS VPNs*, IETF Request for Comments 2547, March 1999.

[24] *Multi-Protocol over ATM Technical Specification Version 1.0*, ATM Forum, AF-MPOA-0087.000, 1997.

[25] Newman P., et al., *Ipsilon's General Switch Management Protocol Specification Version 2.0*, Request for Comments 2297, March 1998.

Rationalizing Key Design Decisions in the ATM User Plane

Daniel B. Grossman
Motorola, Inc.
dan.grossman@motorola.com

Abstract

Any technology requires some number of key design decisions. In the case of the ATM user plane, the choices to use fixed length, 53 byte cells and virtual connections were unorthodox from the perspective of many in the networking research community. This paper attempts a technical justification for those design decisions.

1 Introduction

Technological superiority is not the sole determinant of market success in technology wars; witness, for example, the results of the Betamax vs VHS wars. ATM, which was once hoped to be the underlying substrate for next generation multiservice networks, never achieved its potential. Its proponents permitted it to be drawn into a technology war with IP, and in the end, it was badly battered in the marketplace. Its opponents were able to portray it as complex and inefficient, and this image did much to undermine it.

ATM's largest vulnerability in the war of perception was in three related design decisions: its use of fixed sized protocol data units (PDUs), or cells, the 48-octet cell payload size, and its stateful, or connection-oriented nature. These were very different from design decisions made in IP, and thus unorthodox from the perspective of many in the networking research community. Some of the overhead that resulted from these decisions quickly attracted a derisive tag: the cell tax. This stuck, and contributed significantly to ATM's image problem. This paper presents a technical defense for these design decisions.

2 Fixed Size Cells

Fixed sized PDUs are the most singular design decision in the ATM technology. Much of the benefit of ATM, and many other design decisions, flowed from the fixed size cell.

2.1 Deterministic service times

Fixed length PDUs have deterministic service times. Deterministic service times reduce average waiting times in congested systems. As an approximation, recall that an M/D/1 queuing system has half the average waiting time of an M/M/1 queueing system [1]. This means that cell-based networks can operate with higher link utilization than variable packet based networks while maintaining acceptable delay and loss. Anecdotally, the Internet backbone is engineered to 30% average loading, while ATM networks can be engineered to 80% average loading [2]. Deterministic service also leads to tighter upper bounds on delay for real-time traffic than non-deterministic service times. This is especially useful when real-time and non-real time traffic shares the same physical links, for low-rate links, and in the presence of constant rate real-time traffic having dissimilar rates.

Queueing systems with deterministic service times are easier to analyse than those with general service times. This property can be used to simplify the design and reduce the runtime complexity of admission control, scheduling and other networking algorithms.

2.2 Granularity mismatch

Variable length packet forwarders can only forward integral packets, yet must account for rate and other functions of time in units of bytes. This inconsistency can perturb network dynamics and add complexity. In the ATM forwarding path, the cell is only unit of interest, and no such inconsistency exists.

For example, policers in variable length packet networks accumulate credits in units of bytes, while admit/mark/ drop decisions must occur on packet boundaries. Thus, if a packet arrives when the number of available credits is positive and non-zero but

less than the packet length, the policer must either mark or drop the whole packet, or 'borrow' credits against future packets. Each of these alternatives discriminates against larger or smaller packets (respectively), and over or under polices (respectively) [3]. Since policers in cell-based networks make admit/mark/drop decisions and accumulate credits on units of one cell, they more precisely police to traffic contracts. Thus, admission control decisions can be more aggressive without violating service guarantees.

Similarly, schedulers in variable length packet forwarder can decide which of several queues to service next next only on a per-packet basis. When the scheduling policy is fairness (or weighted fairness), a simple scheduling algorithm, such as (weighted) round robin or (weighted) fair queueing is biased against sources that send small packets. More sophisticated algorithms, such as deficit (weighted) round robin, are required to correct this bias, but they add complexity and discriminate to a lesser extent against sources that send larger packets [4]. In networks with fixed length PDUs, the unit of scheduling is the size of the PDU, and therefore no such discrimination can occur: simpler schedulers are fair.

For real time flows, it is desirable to service PDUs as close as possible to an ideal departure time. Various kinds of calendar scheduling schemes have been used to do this. Fixed sized PDUs greatly simplify the calendar scheduling problem, since the size of the PDU need not be taken into account in determining whether it can be scheduled before a more urgent one.

2.3 Hardware Implementation

Queueing systems, buffer managers, interconnects and other datapath elements can be greatly simplified by fixing the size of the PDUs they carry. Complexity metrics such as interconnect width, memory sizing, number of control lines, and amount of handshaking can be reduced by fixing the length of PDUs. These metrics can affect die size, clock speed and pin counts. For example, a fixed sized PDU allows pipeline stages to be made synchronous, permits parallel processing without having to perform internal buffering to avoid misordering, and requires only a single size buffer pool, without need for fragmentation. Deterministic service times remove blocking, and facilitate cycle count budgeting. These advantages are so significant that many Ethernet switches and IP routers use fixed size data units in their fabrics.

3 53 Octets

The 48-octet payload and 5-octet header of the ATM cell were a notorious political compromise between the needs of speech and data transmission. At the time that the cell size needed to be set, the French Administration planned a telephony transport network with budgeted packetization delays of 4 ms, or 32 octets per cell with 64 kbit/s PCM; this was small enough to avoid the need for echo cancellation on calls within continental France. Opinion at the time was that optimal cell size for data transport would be between 64 and 256 octets. The compromise, debated over several meetings of the former CCITT SG 13 [5], is not quite optimal for either telephony or data, but may be globally optimal for the mix of traffic that would be carried in a multiservice network.

- For PCM telephony, a full cell has a packetization delay of 6 ms, which is usually within the delay budgets of modern Voice-over-ATM systems, even those designed for transcontinental calls.

- Measured packet size distributions in the Internet have a sharp peak at 40 octets, which is the size of an IPv4 datagram containing a TCP SYN, FIN or ACK, with neither IP nor TCP options. Such a packet fits exactly into a single cell when AAL5 is used. Unfortunately, LLC/SNAP encapsulation is frequently used instead of the so-called null encapsulation. It adds 6 octets of overhead, and therefore forces TCP control packets to cross a cell boundary.

- Two MPEG2 transport streams packets exactly fit into eight cells when AAL5 is used.

In addition, most switch implementations require some internal overhead to be prepended to each PDU to form an internal forwarding unit. Switch hardware usually operates in units of bytes that are powers of 2; thus a 64 bytes internal forwarding unit (including 16 bytes of overhead) can be extremely convenient.

The “cell tax” slur arises from the five octets of overhead required for 48 octets of payload. Recollect that all packet-based networks have header overhead. ATM cell header overhead is 9.5%. PCM telephony systems that can accommodate a 6 ms packetization time suffer only this overhead. MPEG2 transport over ATM suffers an additional 3.2% overhead due to the AAL5 trailer, in the default mode of operation (two MPEG2 TS packets in an AAL5 PDU). For IP

transport over AAL5, including LLC/SNAP encapsulation, overhead as measured in the Internet is 20 percent [6]. Without LLC/SNAP encapsulation, it would be 15%, leading one to speculate that if header overhead were truly meaningful to network operators, LLC/SNAP encapsulation would have been eliminated. More important, operators can more than compensate for ATM header overhead by more aggressive traffic engineering, which is made possible by deterministic queueing.

4 Virtual Connections

Packet-based networking systems may be characterized as being either connection oriented or connectionless. Selection of one of these design approaches has broad implications on the network architecture and its implementations. This has been a deeply controversial subject since the early 1970s. For the environment for which ATM was intended, connections and their attendant state information have a number of advantages, and some disadvantages.

Much processing is done once in connection-oriented systems during connection establishment, but done for each packet in connectionless systems. This makes connection-oriented systems better optimized for long-lived communications, as would be expected for telephony, video delivery, and similar applications. Connectionless systems are better optimized for short lived communications such as DNS lookups. A majority of packets in the Internet are associated with long-lived flows, even though the majority of flows are short lived [7]. Attempts by the ATM Forum to create a native connectionless mode to complement the connection oriented mode were unfortunately not successful. On the other hand, one could imagine that applications might have evolved differently if the underlying network encouraged design for long-lived flows.

Per-packet forwarding decisions in ATM nodes are can be implemented with simple table lookups or binary content-addressable memories, over the VPI and/or VCI fields, with O(1) computational complexity. The size of these tables can scale with the number of VP and/or VC links expected to cross each interface. IP requires longest match lookup over the IPv4 or IPv6 destination address fields. To the author's knowledge, the least computationally complex longest match algorithm has lookup complexity of O(LogN) and update complexity of O(kLog$_k$N) [8]. Ternary CAM devices help, but are complex and presently expensive. Forwarding tables scaled to global Internet presently need tens of thousands of entries.

State information is required by policers, shapers, markers, schedulers, per-connection queues, flow control and admission control, which are needed to support services with QoS guarantees. In an ATM network, this state information is inherently associated with a virtual connection link through the VPI/VCI. In connectionless networks, packet classification is necessary to associate a packet with its state information. Classifiers have between O(logN) and O(N) complexity, and mechanisms and policies for configuring classifiers (especially in the core) are complex and likely to be imprecise.

State information can be used in conjunction with per-connection queues and scheduling algorithms to enforce network policies (e.g., fairness) among best effort connections. Connections that transmit at a rate exceeding their share of link bandwidth fill their queues, invoking discard policy. As a result, these connections do not interfere with better-behaved connections. Forwarders for connectionless networks may, of course, implement classification and per-flow queueing, at cost in complexity as noted above. Otherwise, misbehaved connections can at best be dealt with in a statistical fashion, using an active queue management algorithm such as RED. While there is some evidence that such algorithms improve the stability of the Internet [9], there is also evidence that they are not effective [10,11], that they are hypersensitive to configurable parameters [12], and that they do not address non-responsive flows. Per-flow queueing and longest queue discard are a far better solution.

Connection-oriented networks do not suffer from many of the security vulnerabilities that are often exploited in connectionless networks; for example, attacks that depend on address spoofing, such as the SYN attack, are not possible in connection oriented networks. State information also makes tracing an attack back to its originating interface significantly easier. Further, state information is useful as a basis for security associations and audits, and simplifies security design and implementation. In addition, usage accounting, as required for usage-based billing, requires state information. In cases such as virtual private networks, IP-in-IP and IPSEC tunnels, with their considerable overhead, are used as a substitute for connection state.

5 Conclusions

ATM's key design objectives were to operate in a public network environment, to carry real time and non-real time traffic and to be optimized for hardware implementation in switches. A series of design decisions flowed from those objectives, including virtual connections and fixed size, 53-octet cells. From a technology perspective, it can be argued that those decisions withstood the test of time. Unfortunately, the arguments for these decisions are subtle, and the counter-arguments were simple and were made persuasive by repetition. It is not clear how much the "cell tax" slur contributed to limiting the success of ATM. There were other issues, such as the experience curve resulting from a large previously installed base of Ethernet, slow deployment of capabilities to simplify configuration (especially SVCs), lack of ATM-aware applications software, delays in critical standards, and numerous business decisions made by some industry players. Nonetheless, market perception was an important factor, both directly and in its effects these other issues.

Perhaps the most important lesson that can be drawn from this paper is that proponents of novel networking technologies need to justify their design trade-offs in a way that will be understood by the market, and to reinforce their justification. They must also be prepared respond forcefully and credibly when these trade-offs are attacked. ATM's proponents did neither.

References

[1] Kleinrock, Leonard Queueing Systems vol. 1 (New York, John Wiley and Sons, 1975) p. 191

[2] Private conversations

[3] Bernet, Y; Blake, S. Grossman, D; Smith, A "An Informal Management Model for Diffserv Routers" RFC 3290 (April, 2002)

[4] Shreedhar, M.; Varghese, G. "Efficient Fair Queueing using Deficit Round Robin" *ACM Computer Communication Review*, vol. 25, no. 4, pp. 231–242, Oct. 1995.

[5] Reports of the meetings of CCITT SG 13, 1988

[6] Thompson, K.; Miller, G.J.; Wilder, R. "Wide-area Internet traffic patterns and characteristics" *IEEE Network*, Volume: 11 Issue: 6 , Nov.-Dec. 1997 pp. 10 -23

[7] ibid.

[8] Ruiz-Sanchez, M.A.; Biersack, E.W.; Dabbous, W. , "Survey and taxonomy of IP address lookup algorithms" *IEEE Network* , Volume: 15 Issue: 2 , March-April 2001 pp. 8 -23

[9] Braden, B. et al, "Recommendations on Queue Management and Congestion Avoidance in the Internet", RFC 2309 (April 1998)

[10] May, M.; Bolot, J.; Diot, C.; Lyles, B., "Reasons not to deploy RED", technical report, June 1999.

[11] Christiansen, M; Jeffay, K; Ott, D; Smith, F.D. "Tuning RED for Web Traffic", *Proc. ACM SIGCOMM*, August 2000

[12] Firoiu, V.; Borden, M., "A Study of Active Queue Management for Congestion Control", *Proc. IEEE Infocom 2000*

A Perspective on how ATM Lost Control

Simon Crosby*, Sean Rooney†, Rebecca Isaacs‡, Herbert Bos§

Abstract

Contrary to the initial high expectations, ATM failed to become the universal network technology covering all services and running from the desktop to the backbone. This paper tries to identify the technological problems that contributed to this failure.

1 Introduction

From a service provider perspective, the Internet is a nightmare. Although enterprises are reaping tremendous benefit from lower communications costs and new IP-based applications, the anticipated profits to service providers from the Internet economy have not materialised, and record numbers of carriers and their suppliers are going out of business. The roots of the problem lie in the architectural principles of IP which delivers just one service: unreliable, insecure, best effort connectivity. Although tremendous benefit is gained from IP at the edge of the network due to its inherent simplicity and because it allows the application provider to be entirely freed from network constraints, there is an assumption that the providers will continue to increase capacity so that IP based applications will deliver high quality. Thus in the IP model the provider has to deliver constantly growing amounts of bandwidth, but is totally excluded from the value chain. Effectively the provider takes the role of a bit pipe, with no way to link their revenues to the value their customers derive from the network[1]. Furthermore, the best effort service offered by IP is inadequate for enterprise customers who have relied for years on the dependable alternatives of leased lines, Frame Relay and ATM.

Notwithstanding the ubiquity of IP at the edges of the network, ATM still has very healthy growth numbers. All DSL traffic crosses ATM, it comprises the core of most IP networks, ATM VPNs are commonplace and there is even a considerable market for circuit emulation services, including voice over ATM. The lack of any kind of service guarantees together with the difficulty for service providers to charge for value added to an IP network, suggests that ATM is a missed opportunity. So what went wrong?

2 The ATM value proposition

Borrowing from earlier control planes, specifically the narrowband Q.931 signalling standard, the ATM standardization community developed a User-Network Interface (UNI), and subsequent Network-Network Interface (NNI), that would allow users to dynamically signal for connectivity with the appropriate bandwidth and QoS guarantees that applications required. The notion of a User-Network signalling interface was seen as a crucial part of the value proposition for ATM.

From the point of view of the service provider, IP is flawed because there is no way to link revenues to the value customers derive from the network. In this respect, ATM had the potential to gain advantage over IP, provided that a suitable UNI could be developed. However, the UNI standardised by the ATM Forum was defective in at least three ways. Firstly, it was flawed because the UNI was tied to a specific

*simon@cplane.com, CPLANE Inc, USA
†sro@zurich.ibm.com, IBM Research, Switzerland
‡risaacs@microsoft.com, Microsoft Research, UK
§herbertb@liacs.nl, LIACS, The Netherlands

[1]Data has grown to roughly 60% of the traffic volume in some carriers but accounts for a tiny fraction of the profits so providers are being forced to put more assets in place to support their lowest margin business.

technology (ATM) requiring the signalling layer to be implemented 'on the box'. This means that every access interface, in particular every operating system, edge router and device that might be attached to the network in the future, must implement a complex, heavyweight signalling stack. It doesn't take much insight to see that to require that every OS implement the signalling stack was an act of insanity which doomed ATM to failure. Secondly, the model provided no accountability for signalling. That is, when the user exercises the UNI, and signals for an ATM connection, there is no hook into the billing system that allows the provider to account for and therefore bill for the value delivered over the interface. Finally, the ATM UNI was extraordinarily heavyweight and complex. Because it tried to define all possible service models for all possible applications, present and future, it became bulky and difficult to understand. At the same time, its closed nature made it impossible to provide new services for applications with specific requirements that were not catered to by the standards.

Thus the ATM UNI signalling protocol was crippled from the outset, imposing impossible burdens on both service providers and the users of the network. Perhaps it is time to re-evaluate the need for signalling across the User-Network interface, and this time to get it right. The key requirements are clear: separate the UNI from the underlying network delivery technology, make it easy for application users at the edge of the network, and allow providers to bill for services delivered using it.

Let's step back for a moment to Signalling System Number 7 (SS7), a network of signalling channels used in the Public Switch Telecommunication Network. Why was it valuable? The answer is remarkably simple: It separated the signalling from the equipment. The SS7 Service Control Point allowed the provider to implement value propositions (albeit limited ones) independent of the underlying switched network. The provider owned the application (eg 800 number or 'follow me') and on the basis of performing some function in response to a user interaction via the 'UNI', the provider could charge, and make money. Although designed on this model, ATM control is too complex and presents the wrong level of abstraction to application developers and users. Small wonder then that almost all ATM connections are set up using the hopelessly impoverished pseudo signalling protocol SNMP: Providers 'provision' connections (usually sPVCs or PVCs), turning the dynamic nature of ATM into a sham, and its capabilities simply into a flexible virtual topology on top of fixed TDM capacity. This allows providers to traffic engineer for IP, deliver basic DSL 'pipes' and some voice traffic, with QoS, but the oft-touted B-ISDN dynamic service interface has never been deployed.

3 Open Signalling

SS7 on the PSTN was successful because it offered service providers a way to add value to the network and to charge accordingly, but the system is closed to third parties, inhibiting the development and deployment of new and innovative services. The need for an open network was not recognised by the ATM Forum, who, by not specifying the interface between the control plane and the physical network, tied the control software to the physical devices. Although architecturally distinct, cell forwarding and connection establishment functions tend to be integrated, and this results in switch-specific control systems which can only be maintained and evolved by vendors.

In academia such concerns led to the establishment of the OPENSIG forum to develop a more flexible control plane for ATM. Open signalling systems require that there be a clear divide between the control plane and the data plane, with an open, switch-independent control API. Open signalling ideas were heavily influenced by the industrially developed IP Switching[4] which discarded entirely the standard ATM control plane. An IP Router built over an ATM switch made a local decision as to whether the packets of a flow should be forwarded at the software IP layer or, if the flow was sufficiently long-lived, to cut through onto the ATM hardware layer. IP Switching was ultimately not successful because hardware IP routers became commonplace, but its General Switch Management Protocol for communication between the IP control layer and the ATM forwarding layer is an example of a switch-independent

control API which has since been standardized by the IETF[3].

The large number of proposals to replace the ATMF control plane seriously questions the basic assumption of the standardization bodies that there is a single control interface which will provide for the needs of all applications and services. In response to this, the open signalling community addressed the need to support a plurality of ATM control systems. Work done at Cambridge, using virtual resource partitioning, showed how this could be achieved[7] and was extended to support novel control planes[5, 1] and a virtual network service[8, 6, 2]. These academic efforts have in turn been adopted by industry through the Multiservice Switching Forum and the IEEE P1520 proposed standard for network APIs.

4 Lessons learnt with ATM

IP is the basic unifier of the enterprise application suite, and is the primary vehicle for applications of value. This indicates that the control procedures for interacting with the network should occur at a much higher layer, specifically the application layer, to allow applications to directly interact with the network to dynamically request the security, bandwidth QoS and other attributes that are required.

Pipe based services (whether ATM or IP based) relegate the service provider to the role of simple bit shifter, but the value proposition has moved outside the network. The right way to solve the control problem is to ensure that it is *out of band* so it allows customers to dynamically request network services to meet application specific needs (bandwidth, QoS, etc). It should also be technology independent, so that the *service* layer concepts of value can be delivered independent of the underlying delivery technology (ATM, GigE, MPLS, IP or whatever comes next). This motivates for development of 'application layer signalling' which enables direct interaction between application components and the network, without requiring detailed knowledge of the network infrastructure, and encompassing both servers and networks.

Why is application layer signalling important? Primarily because the only successful service provider models are those that allow customer-network interaction. This is possible in the PSTN, but it is impossible in the 'best effort pipes' based IP network. Providers need to be able to interact with their customers, to implement services for which their customers are prepared to pay a premium. If they are unable to do so, they will become just utility bits-per-second providers.

5 Conclusion

Ultimately open signalling failed; ATM's control plane has been abandoned and ATM is used as one of many layer-2 protocols in the public internet. Retrospectively it can be seen that simplicity of deployment was the killer argument for IP. However, ease of deployment does not equate with simplicity of management and IP's lack of precise control over the data path makes it hard for network operators and service providers to differentiate their offerings in a meaningful way. ATM's relegation as nothing more than a flexible way to compensate for the most blatant failings of IP, as far as reliability and predictability of service are concerned, is a result of over-complex protocols without an adequate understanding of the business driver.

6 Acknowledgments

Thanks are due to Kobus van der Merwe for his valuable insights and discussions.

References

[1] Bos, H. Open extensible network control. *Journal of Network and Systems Management 8*, 1 (Mar. 2000).

[2] Isaacs, R., and Leslie, I. Support for resource-assured and dynamic virtual private networks. *IEEE Journal on Selected Areas in Communications 19*, 3 (Mar. 2001), 460–472. Special Issue on Active and Programmable Networks.

[3] Newman, P., Edwards, W., Hinden, R., Hoffman, E., Ching, F., Lyon, T., and Minshall, G. Ipsilon's General Switch Management Protocol specification version 2.0. RFC 2297, Mar. 1998.

[4] Newman, P., Minshall, G., and Lyon, T. IP switching: ATM under IP. *IEEE/ACM Transactions on Networking 6*, 2 (Apr. 1998), 117–129.

[5] Rooney, S. The Hollowman: An innovative ATM control architecture. In *Integrated Network Management V* (San Diego, USA, May 1997), A. Lazar, R. Saracco, and R. Stadler, Eds., IFIP & IEEE, Chapman & Hall, pp. 369–380.

[6] Rooney, S., van der Merwe, J. E., Crosby, S. A., and Leslie, I. M. The Tempest: A framework for safe, resource-assured programmable networks. *IEEE Communications Magazine 36*, 10 (Oct. 1998), 42–53.

[7] van der Merwe, J. E., and Leslie, I. Switchlets and dynamic virtual ATM networks. In *Integrated Network Management V* (San Diego, USA, May 1997), A. Lazar, R. Saracco, and R. Stadler, Eds., IFIP & IEEE, Chapman & Hall, pp. 355–368.

[8] van der Merwe, J. E., Rooney, S., Leslie, I., and Crosby, S. The Tempest—a practical framework for network programmability. *IEEE Network Magazine 12*, 3 (May/June 1998), 20–28.

The Influence of ATM on Operating Systems

Jonathan M. Smith*
CIS Department, University of Pennsylvania
jms@cis.upenn.edu

Abstract

The features of ATM offered many attractions to the application community, such as fine-grained multiplexing and high-throughput links. These created considerable challenges for the O.S. designer, since a small protocol data unit size (the 48 byte "cell") and link bandwidths within a (binary) order of magnitude of memory bandwidths demanded considerable rethinking of operating system structure.

Using an historical and personal perspective, this paper describes two aspects of that rethinking which I participated in directly, namely, those of new event signalling and memory buffering schemes. Ideas and techniques stemming from ATM network research influenced first research operating systems and then commercial operating systems. The positive results of ATM networking, although indirect, have benefitted applications and systems far beyond the original design goals.

1 Introduction

The various technical contributions of the US "Gigabit Testbed" program[Computer 90] were both manifold and undersold. Amongst the more significant contributions were those made in the AURORA Gigabit Testbed, which linked Penn, Bellcore, IBM Research and MIT in the Northeast corridor of the United States [Clark 93]. While AURORA experimented with two packet formats, "Packet Transfer Mode" (PTM) and "Asynchronous Transfer Mode" (ATM), in retrospect the ATM-centric research had significantly more impact on modern operating system software.

*Work supported by the NSF and ARPA under Cooperative Agreement NCR-8919038 with CNRI, by Bell Communications Research under Project DAWN, by an IBM Faculty Development Award, and by the Hewlett-Packard Corporation.

It is is worthwhile to set the stage. While AURORA was initiated in 1990 under the HPCA ("Gore Bill") the research was actually kicked off at Penn and MIT in 1989 under Project DAWN, funded by Bellcore. The goal was to build an integrated LAN/WAN infrastructure using an OC-48 SONET channel provided by Bell Atlantic, MCI and Nynex. Using a clever OC-12 cross-connect topology devised by Dave Sincoskie of Bellcore, we were able to concurrently operate PTM over SONET from Penn to IBM and MIT, and ATM over SONET from Penn to Bellcore to MIT. While supercomputers were used in several other testbeds, AURORA focused on workstation technology, as it was believed (correctly, in retrospect) that delivering high throughputs to this class of machine would have more long-term impact. The available workstation technology was the first generation of RISC computers, which were characterized by an increased number of simpler instructions executed per clock cycle, higher clock rates, and relatively poor DRAM and I/O throughputs given the computational performance. As it is today, multiple levels of cache memory were used to resolve some of the CPU/DRAM performance "gap" [Patterson 02]. A major problem that we faced as network subsystem architects was that the advantages of caching only apply once the data is *in* the memory hierarchy, while a major role of the network subsystem is moving the data into the hierarchy, before any such advantages apply.

One major research direction pursued at Penn was an architectural study of the challenges in delivering the substantial bandwidths (while not so impressive today, the OC-3c bandwidths we targeted to in 1989/1990 were over 13 times as fast as the commonly used LAN technology). This architectural study took the form of a hardware/software codesign, where a hardware subsystem, in the form of an ATM host interface [Traw 91, Traw 93], was designed in concert with new operating system sup-

port [Smith 90, Smith 93, Chung 95] designed to expose the strengths of the ATM technology, in particular its substantial bandwidth. While a secondary consideration at the time, another lasting contribution came from exposing the fine-grained control of multiplexing possible with ATM [Nahrstedt 96].

The remainder of this paper is organized in three sections. The next section, on "Buffer Management", discusses some of the challenges associated with memory architectures in the early 1990s and connects a thread of research results which led to changes in many commercial operating systems. Section 3 covers changes in event-signalling architectures, which have had considerable influence on the research community but perhaps less on the commercial front. Finally, in Section 4 we conclude the paper, observing first that ATM-driven advances in software paved the way for Fast Ethernet and faster LANs, and second that the same fundamental stresses amongst trendlines are still in place as network engineers and workstation designers continue to march to different drummers.

2 Buffer Management

David Farber, my colleague at the University of Pennsylvania, had worked with Bob Kahn to get the Gigabit Testbed initiative underway. Kahn had set a "Gigabit litmus test", which demanded a gigabit per second throughput, delivered to or from *a single application*. In early 1990, I began considering the shape of an operating system appropriate for the support of a workstation in an environment with very high performance networks.

While it was by then quite clear that a supercomputer [Borman 89] could deliver or accommodate such throughputs, it was also clear that workstations would be extremely challenged if they were called upon to meet the "litmus test". Even if the technical challenges of the host adaptor hardware were met, it was unclear if the workstation could handle the data flow from the adaptor. While the merits of the RISC workstation technology were clear from a computational perspective, it was unclear how they would handle data management, particularly in the performance regimes the ATM technology required.

I laid out principles for a new operating system called "UPWARDS" [Smith 90] (for "U Penn Wide Area Research Distributed System") intended for a network with bandwidth within an order of magnitude of memory bandwidth.

These included (quoting from the paper):

- UPWARDS scheduling is entirely synchronous; the only "interrupt" allowed is that of the system clock driving the scheduler. Interrupts complicate device drivers and are really an artifact of a desire for high levels of multiprocessing, rather than high performance computing with low levels of multiprogramming. In addition, interrupts defeat architectural features necessary for very high performance, since as caches and pipelines.

- The UPWARDS interprocess communication mechanism is shared memory.

While like many operating systems designs, UPWARDS was never fully realized, it provided considerable guidance in what was to follow. Others were of course working in this space, in particular Clark and Tennenhouse [Clark 90] had likewise noted the importance of memory bandwidth, with their focus being on protocol architectures. The vision of UPWARDS IPC being shared memory was driven, in fact, by a belief that a distributed shared memory model for distributed computing [Smith 91] would provide an appropriate basis for a high-performance protocol architecture.

2.1 Hardware Context, O.S. Presumptions and Basic Arithmetic

Working with IBM gave AURORA researchers access to a particularly potent workstation technology, the RS/6000, which was just being made available at the time the testbeds were forming. The IBM RS/6000 had been recently introduced [Bakoglu 90] and was IBM's entry in the engineering workstation market. All indications were that it was quite competitive [Simmons 90], being equal to supercomputers one generation old (on computational workloads).

The outstanding feature of the RS/6000 was its relatively large memory bandwidth (using a tool we developed, we measured a Model 320 to have 1 Gbit/s of bandwidth, the Model 530 to have 1.8 Gbit/s of memory bandwidth, and the Model 580 to have 2.5 Gbit/s of memory bandwidth). Others confirmed this later with benchmarks [McVoy 96]. The I/O bus technology, the Microchannel, was an apparent limitation. We chose to work around this by applying a specialized style of direct-memory-access transfer called streaming, which provided twice the throughput of

normal DMA, giving us some hope of reaching the application performance targets for AURORA. The Microchannel throughput in this mode was 320 Mbit/s, adequate for the OC-3c data rate of the adaptor.

Looking at some basic performance metrics for this machine, it appeared to be a reasonable platform, but it was clear that some new thinking was required: the off-the-shelf operating system (AIX) would have to be modified, both to reduce copying and to support multimedia. I began to examine how the UPWARDS ideas could be implemented using AIX (IBM's version of UNIX) as a starting point on the RS/6000.

2.2 Reducing Copying

Even with the Gbit/s range memory throughputs on the RS/6000, it was clear [Watson 87] that reducing buffer copying would have considerable impact on performance. We avoided some software-driven copying by use of a hardware feature of the IBM RS/6000, a set of Translation Control Words (TCWs) that provided virtual addressing capability to I/O devices such as our host adaptor. The challenge of structuring memory and choosing where it was addressed for adaptor transfers, as well as scheduling transfers to allow maximum concurrency with other processing (such as that required by the application), led to a "double buffering" strategy.

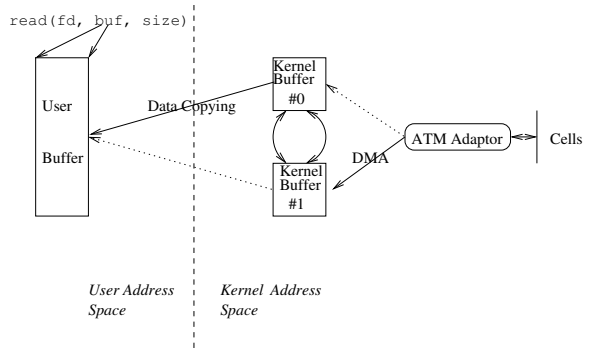


Figure 1: Kernel Buffer Ring - Concurrency with Copying

Management of a pair of buffers (this is a fairly standard scheme for network adaptor device drivers, sometimes generalized as a *buffer ring*), as shown in Figure 1, gave the system considerable throughput (up to about 110 Mbps), and considerable concurrency. This performance was achieved by spreading the per-packet overheads over many bytes of data by using large 32KB or 64KB datagrams. However, the performance of the system for small buffers was less impressive, dropping into the tens of megabits per second for packet sizes more typical of application traffic (50-1000 bytes).

The shared memory communication model of UPWARDS led to exploring a lighter-weight data transfer mechanism between user processes and the ATM adaptor. Recoding of the driver support for the UNIX `read()/write()` calls [1] allowed transfers directly to or from a user buffer passed to the system call, as shown in Figure 2. Even with the heavyweight control operations involved in interpreting the system calls, this essentially achieved "zero-copy" operation for the data transfers, as only the required DMA from the adaptor was required to get the data into application memory.

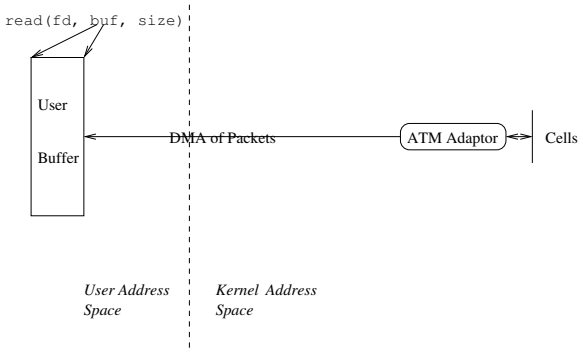


Figure 2: No copy (other than required DMA), *but* no concurrency

Of course, the application was blocked while this occurred, so there was little concurrent activity for the process using the mapped buffer. Thus we sought to remove the system call overhead for small packets by implementing a shared memory region between kernel and user space, with a carefully orchestrated passing of control of the shared regions between user process and kernel to ensure that all available concurrent operation of the hardware could be exploited. The architecture is illustrated in Figure 3.

While all of these configurations were implemented and were able to fully utilize the ATM link at the larger packet sizes, the shared memory interface far outperformed the system-call based scheme on small packets, had the least effect on other applications, and fully exploited the ability of the bus-mastering

[1] I/O to a UNIX[Thompson 78] "raw" device is passed to a device-specific routine, *e.g.* `atm_read()` and `atm_write()`

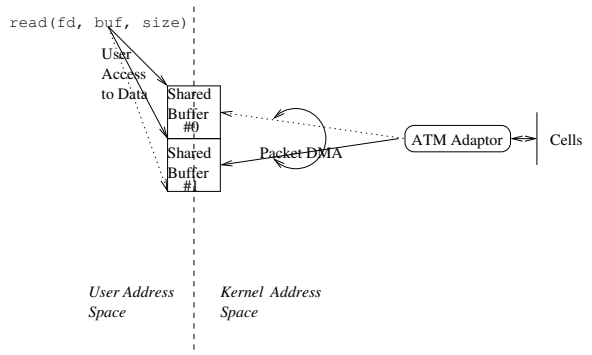


Figure 3: Memory buffers shared between user and kernel address spaces

adaptor to operate concurrently with the processor.

These implementations were used to support applications ranging from a wide-area distributed shared memory [Shaffer 96] to a telerobotics system [Nahrstedt 96]. The applications used a TCP/IP [Alexander 94] implemented using a virtual device abstraction[2] used to cope with the fact that the ATM adaptor was implemented as *two* Microchannel cards, one for ATM segmentation, with the other for ATM reassembly.

These experiments demonstrated that one of the basic intuitions of the initial UPWARDS, that shared memory models for communications would lead to high performance, were experimentally sound. We address the interrupt issue below in Section 3.

2.3 Related and Follow-On Work; Commercial Impact

The most important follow-on work in the research community was that of Peterson and Druschel. In a paper presented in Tucson in 1992 [Traw 92], we discussed the basic double-buffering strategy which allowed application processing of data in one buffer while allowing a streaming transfer into another buffer. Druschel [Druschel 93], moving buffer management issues into the mainstream operating systems community, adopted this basic strategy in his "fbuf" buffer management architecture. However, rather than using VM protection coupled to double-buffering, Druschel used the type system of his implementation language to control access to specialized buffers called "fbufs". Fbufs' data typing allowed implementation of the buffer control protocol without the extra overheads of address space setup and teardown for virtual address translation, showing that the buffering schemes we developed could be made even more efficient. Druschel [Druschel 94] demonstrated fbufs in an implementation for Bruce Davie's Osiris ATM adaptor. Druschel's design exhibited adaptor to host memory transfers at over 500 Mbps for UDP packets [Druschel 94] on an extremely fast processor, the Alpha, which had just been introduced by the Digital Equipment Corporation. A host-to-host UDP throughput of about 300 Mbps could be inferred from the upper bound on the TurboCHANNEL write throughput reported in the paper.

While as far as I am aware, TCP was never demonstrated with this system,[3] fbufs are a nice improvement over the basic shared memory abstraction which started with UPWARDS. It also nicely illustrates how work done initially for networking goals can influence mainstream operating systems research. Indeed, this influence can persist; refinement of this architecture continued with LRP [Druschel 96].

The most important work done concurrently was by the Hewlett-Packard Laboratories [Dalton 93, Banks 93, Edwards 94] in Bristol, UK. There, a high performance host adaptor architecture was developed which allowed a direct mapping of substantial buffer memory on the adaptor directly into application address space. The power of this approach was that it was then reasonably straightforward to write an application-level TCP stack which directly manipulated buffer memory on the adaptor, minimizing copying across the bus to which the adaptor was attached. In many ways, this user-level TCP/IP set directions for the user-implemented protocols in vertically structured operating systems such as the exokernel [Engler 95].

Commercial vendors paid considerable attention to the software architectures we developed. One example is the work of Chu [Chu 96], who adapted the shared memory techniques and fbufs to the commercial Solaris operating system supplied with Sun Microsystems computers. The Chu work supported a 622 Mbit/s ATM adaptor developed by Sun.

[2]The virtual device was defined by constructing a driver *overlay* which redirected system calls for the virtual device to appropriate invocations of driver entry points for the physical devices.

[3]A problematic DMA controller was rumored to have made repeatable measurements difficult.

3 Event Signalling

Event-signalling within the network subsystem between the hardware network interface device and the software device driver is typically accomplished via polling or device-generated interrupts. In our implementation of an OC-3c ATM host interface for the IBM RS/6000 family of workstations [Traw 93, Smith 93], we replaced the traditional forms of this crucial function with "clocked interrupts." Clocked interrupts, like polling, examine the state of the network interface to observe events which require host operations to be performed. Unlike polling, which requires a thread of execution to continually examine the network interface's state, clocked interrupts perform this examination periodically upon the expiration of a fine-granularity timer. In comparison to interrupts, clock interrupts are generated indirectly by the timer and not directly by the state change event.

3.1 Hardware/Software Context, O.S. Presumptions and Basic Arithmetic

Consider a system with an interrupt service overhead of C seconds per interrupt, and k active channels, each with events arriving at an average rate of λ events per second. Independent of interrupt service, each event costs α seconds to service, e.g., to transfer the data from the device. The offered traffic is $\lambda * k$, and in a system based on an interrupt-per-event, the total overhead will be $\lambda * k * (C + \alpha)$. Since the maximum number of events serviced per second will be $1/C + \alpha$, the relationship between parameters is that $1 > \lambda * k * (C + \alpha)$. Assuming that C and α are for the most part fixed, we can increase the number of active channels and reduce the arrival rate on each, or we can increase the arrival rate and decrease the number of active channels.

However, for clocked interrupts delivered at a rate β per second, the capacity limit is $1 > \beta * C + \lambda * k * \alpha$. Since α is very small for small units such as characters, and C is very large, it makes sense to use clocked interrupts, especially when a reasonable value of β can be employed. In the case of modern workstations, C is about a millisecond. Note that as the traffic level rises, more work is done on each clock "tick," so that the data transfer rate $\lambda * k * \alpha$ asymptotically bounds the system performance, rather than the interrupt service rate. We note that traditional interrupt service schemes can be improved, e.g., by aggregating traffic into larger packets (this reduces λ significantly, while typically causing a slight increase in α), by using an interrupt on one channel to prompt scanning of other channels, or masking interrupts and polling some traffic intensity threshold.

One would therefore expect that for application workloads characterized by high throughput, heavy multiplexing, and/or "real-time" traffic, clocked interrupts should be more effective than either traditional polling or interrupts.

3.2 Clocked Interrupts

The ATM host adaptor we developed for the IBM RS/6000 [Traw 93] was designed with clocked-interrupt based event-signalling in mind. One artifact of this was that there was no support for interrupt generation, an omission that would plague us later, when we were evaluating whether the clocked interrupt scheme performed as we had intended and needed comparison against interrupts to make the case. While a comparative evaluation would have strengthened the technical case for clocked interrupts, we were able to demonstrate two key assertions. First, clocked interrupts could deliver throughputs up to the hardware limits of the RS/6000, in this case slightly over 130 Mbit/s on the Model 580 processor. Second, clocked interrupts provided excellent support for applications with multimedia streams; this was demonstrated by teleoperating [Nahrstedt 96] a robot arm over an ATM network using several RS/6000s.

The question of whether any advantage actually accrued to clocked interrupts intrigued us, so we set out to study it in a second generation of our host interface architecture. The second generation combined segmentation and reassembly in a single board, and absorbed other useful features from the design of the Afterburner [Banks 93, Dalton 93]. It also operated at OC-12c, or four times the link rate of the RS/6000 adaptor which operated at the OC-3c link rate of 155 Mbit/s.

To get tight control of the clock on the PA-Risc we altered the clock support subsystem to perform clock division, and used the basic single-copy TCP/IP stack developed by HP Bristol [Edwards 94, Edwards 95]. We were then able to construct a polling based scheme using a user process, apply traditional interrupts, and operate clocked interrupts at a variety of clock rates. We used a 640+ SONET compliant link to interconnect two HP 9000 Model

755s back-to-back, and then performed experiments using the `netperf` [IND 95] performance analysis tool, with machines both unloaded (Figure 4) and loaded with a computationally intensive workload (Figure 5).

While the experimental setup and results here are covered in [Chung 95] and [Smith 99], we have the data graphically in this paper rather than in the previous tabular format. For these two diagrams, the label "Poll" with a rate indicates the clocked interrupt service rate, that is, how many times per second the device status is checked. 0.4Khz means that the device is checked 400 times per second. There is considerable variability in performance, and this is relatively more important for small buffer sizes, as the performance for large buffers will be dominated by data transfer limits.

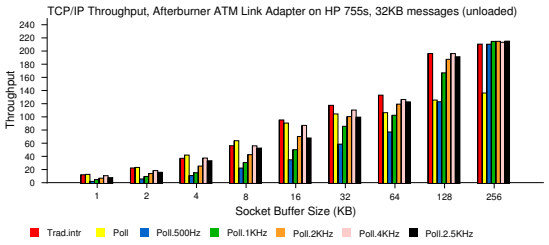


Figure 4: Throughput versus clocking, unloaded machine

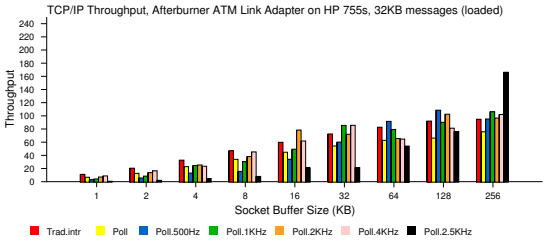


Figure 5: Throughput versus clocking, loaded machine

It is clear from these results that at high polling rates, the clocked interrupt scheme is able to keep up with the traditional interrupt scheme, which is almost everywhere the best performer, with the exception of polling, which does best for small packet sizes. In a lightly-loaded environment, interrupts would appear to be the best solution, except for some anomalous, but repeatable results which show polling best for small socket buffer sizes.

Another important factor in networking performance, and perhaps a more important parameter for distributed applications is the round-trip latency induced by the software supporting the adaptor. Since the hardware was a constant, we could directly compare the software overheads of the three schemes. This was done with the following test. An artificial network load was created using `netperf` with a socket buffer size of 262144 bytes and operating it continuously. Against this background load, ICMP ECHO packets of 4K bytes were sent to the TCP/IP receiver, which was where the event-signaling performance differences would be evident. Sixty tests were done to remove anomalies. Our results showed that traditional interrupts and clocked interrupts at 500 Hz performed similarly, yielding minimum, average and worst case times of 5/12/18 ms, and 4/11/25 ms, respectively. When the systems were not loaded, the performances were 3/3/3 ms and 4/4/6 ms. This suggests that clocked interrupts performed slightly better under heavy load, but slightly worse under unloaded conditions.

We draw three conclusions about this event-signalling architecture. First, clocked interrupts can provide throughput equivalent to the best throughput available from traditional interrupts; both methods provide better performance than polling as implemented here. Second, clocked interrupts provide higher throughput when the processor is loaded by a computationally-intensive process; this suggests that clocked interrupts may be a viable mechanism for heavily loaded systems such as servers, which might also suffer from Ramakrishnan's *receive livelock*. Third, clocked interrupts provide better round-trip delay performance when systems are heavily loaded for large ICMP ECHO packets.

3.3 Related and Follow-On Work; Commercial Impact

The work by Ramakrishnan [Ramakrishnan 93] on "receive livelock" pointed out a real problem with O.S. management of interrupts on fast network adaptors. While Ramakrishnan studied FDDI, the basic problems of an operating system entering a completely interrupt-driven state were applicable to ATM, and where SAR was not done in hardware (as in first generation ATM host interfaces from Fore Systems, which did SAR in software), could be far worse than for FDDI. Mogul and Ramakrishnan [Mogul 97]

noted that clocked interrupts were immune from such considerations, but developed a hybrid architecture, which was interrupt-driven under light load, but switched to polling when a certain threshold of load was passed.

A large and influential body of work at the University of Cambridge [Black 95, Roscoe 95, Hyden 94, Leslie 96] was driven by the need to support multimedia. The resulting *Nemesis* operating system had many features, including Black's scheduling scheme [Black 95] and its architecture as a single address space operating system [Roscoe 95] which enabled high-performance. In many ways this operating system was well-suited to the fine-grained resource multiplexing of which ATM was capable. This is not surprising, as a variety of ATM technologies had their origin at Cambridge in, or slightly before, this period.

The most important derivative work was done by Peter Druschel. In his work on Soft Timers [Aron 00], he confirmed that clock-driven activity was a powerful paradigm for networking subsystems. In a heavily multiplexed subsystem, Morris [Morris 99] demonstrated the value of a polling-like scheme, supporting the basic analysis of traditional interrupts versus polling we presented above. The Click router has had direct commercial impact, as it has been productized by Mazu Networks, Inc.

4 Conclusions

In our discussion of changes in operating system technology, we noted that RISC architectures relied heavily on caches to overcome mismatches in speed between CPUs and DRAMs. Buffer copying tracks DRAM performance rather than base processor performance. In the network attachment arena, the small ATM cell size of *ca.* 50 bytes took about 3 microseconds to arrive at 130 Mbps; 1500 bytes arrives in 12 microseconds at 1 Gbps and 1.2 microseconds at 10 Gbps, or within a rough order of magnitude of the figures with ATM in the early 1990s.

We have shown in this paper how operating systems were revamped to meet the challenges posed by the ATM networks of the early 1990s, and as the paragraph above illustrates, those challenges have not retreated but have in some sense regrouped and reappeared. At the same time, the use of the Internet to bear various forms of multimedia, such as voice and streaming media, has again required operating systems [Muir 01] to examine scheduling and event signalling, both for processing of network traffic and for delivery of that traffic to host applications.

If there is a single lesson to take home from this paper, it is that the ATM technology served as a vanguard to force a rethinking of operating system architecture that might otherwise have been delayed until today, and while operating systems continue to evolve, the effects of this rethinking have already been felt.

5 Acknowledgments

I appreciate the great work of my former graduate students Brendan Traw, Jeffrey Chung, Klara Nahrstedt, Michael Massa, John Shaffer, and Scott Alexander which pushed the state of operating systems for networks ahead considerably. I learned a good deal about hardware working with Brendan Traw and Bruce Davie; I hope they learned some roughly equivalent amount about system software.

I also appreciate the insightful comments of Craig Partridge on both a variety of technical points and their presentation in this paper. I apologize in advance to anyone whose work I left out; the paper was not intended as a comprehensive survey.

References

[Alexander 94] D. Scott Alexander, C. Brendan S. Traw and Jonathan M. Smith, "Embedding High Speed ATM in UNIX IP" *USENIX High-Speed Networking Symposium*, Oakland, CA pp. 119-121 (August 1994)

[Aron 00] M. Aron and P. Druschel, "Soft timers: efficient microsecond software timer support for network processing", *ACM TOCS* 18(3), pp. 197-228 (2000).

[Bakoglu 90] H. B. Bakoglu, G. F. Grohoski and R. K. Montoye, "The IBM RISC System/6000 processor: Hardware overview", *IBM Journal of Research and Development*, 34(1), pp. 12-22 (January, 1990).

[Banks 93] D. Banks and M. Prudence, "A High-Performance Network Architecture for a PA-RISC Workstation," *IEEE JSAC*, 11(2), pp. 191-202 (Feb. 1993).

[Black 95] R. J. Black, "Explicit Network Scheduling", Ph.D. Thesis, University of Cambridge, (April 1995).

[Borman 89] D. A. Borman, "Implementing TCP/IP on a Cray Computer", *ACM Computer Communications Review*, 19(2), pp. 11-15, (April 1989).

[Chu 96] J. Chu, "Zero-Copy TCP in Solaris", *Proceedings, 1996 USENIX*, pp. 253-264 (January 1996).

[Chung 95] J. D. Chung, C. B. S. Traw and J. M. Smith, "Event-Signaling within Higher-Performance Network Subsystems," *Proceedings, HPCS '95*, Mystic, CT, pp. 220-225 (August 1995).

[Clark 90] D. D. Clark and D. L. Tennenhouse, "Architectural Considerations for a New Generation of Protocols", *Proc. SIGCOMM 1990*, Philadelphia, PA (September 1990).

[Clark 93] David D. Clark, Bruce S. Davie, David J. Farber, Inder S. Gopal, Bharath K. Kadaba, W. David Sincoskie, Jonathan M. Smith, and David L. Tennenhouse, "The AURORA Gigabit Testbed," *Computer Networks and ISDN Systems* 25(6), pp. 599-621, North-Holland (January 1993).

[Computer 90] IEEE Computer Staff, "Gigabit Network Testbeds," *IEEE Computer*, 23(9), pp. 77-80 (September 1990).

[Dalton 93] C. Dalton, *et al.*, "Afterburner: A network-independent card provides architectural support for high-performance protocols," *IEEE Network*, pp. 36-43 (July 1993).

[Davie93] Bruce S. Davie, "The Architecture and Implementation of a High-Speed Host Interface", *IEEE JSAC* 11(2), pp. 228-239 (Feb. 1993).

[Druschel 93] Peter Druschel and Larry Peterson, "Fbufs: A high-bandwidth cross-domain data transfer facility", *Proc. SOSP*, 1993.

[Druschel 94] Peter Druschel, Larry Peterson and Bruce Davie, "Experiences with a High-speed Network Adaptor: A Software Perspective", *Proc. ACM SIGCOMM*, pp. 2-13, (August-September 1994).

[Druschel 96] Peter Druschel and Gaurav Banga, "Lazy Receiver Processing (LRP): A Network Subsystem Architecture for Server Systems", *Proc. OSDI*, pp. 261-275, 1996.

[Edwards 94] A. Edwards, G. Watson, J. Lumley, D. Banks, C. Calamvokis and C. Dalton, "User-space protocols deliver high performance to applications on a low-cost Gb/s LAN," in *Proceedings, 1994 SIGCOMM Conference*, London, UK, 1994.

[Edwards 95] Aled Edwards and Steve Muir, "Experiences implementing a high-performance TCP in user-space", in *Proceedings, ACM SIGCOMM Conference*, Cambridge, MA, pp. 196-205, (August-September 1995),

[Engler 95] D. Engler, M. F. Kaashoek, and J. O'Toole, "Exokernel: An Operating System Architecture for Application-Level Resource Management", *Proc. SOSP*, December 1995.

[Hyden 94] E. Hyden, "Operating System Support for Quality of Service", Ph.D. Thesis, University of Cambridge (February 1994).

[IND 95] Hewlett-Packard Information Networks Division, "Netperf: A Network Performance Benchmark (Revision 2.0)", February 15, 1995.

[Leslie 96] Ian M. Leslie, Derek McAuley, Richard Black, Timothy Roscoe, Paul T. Barham, David Evers, Robin Fairbairns and Eoin Hyden, "The Design and Implementation of an Operating System to Support Distributed Multimedia Applications", *IEEE Journal of Selected Areas in Communications*, 14(7), pp. 1280-1297, (1996).

[McVoy 96] Larry McVoy and Carl Staelin, "*lmbench*: Portable tools for Performance Analysis", *Proceedings, 1996 USENIX Conference*, San Diego, CA, pp. 279-294 (January 1996).

[Mogul 97] J. Mogul and K. K. Ramakrishnan, "Eliminating Receive Livelock in an Interrupt-Driven Kernel", *ACM TOCS*, 15(3), pp. 217-252, 1997.

[Morris 99] R. Morris, E. Kohler, J. Jannotti and M. F. Kaashoek, "The Click Modular Router", *Proc. SOSP*, pp. 217-231 (1999).

[Muir 01] Steve Muir, "Piglet: An Operating System for Network Appliances", Ph.D. Thesis, CIS Department, University of Pennsylvania, 2001.

[Nahrstedt 96] K. Nahrstedt and J. M. Smith, "Design, Implementation and Experiences of the OMEGA End-Point Architecture", *IEEE JSAC* 14(7), pp. 1263-1279 (September 1996).

[Patterson 02] D. A. Patterson and J. L. Hennessy, "Computer Architecture: A Quantitative Approach (3d Ed.)", Morgan Kauffman, 2002.

[Ramakrishnan 93] K. K. Ramakrishnan, "Performance Considerations in Designing Network Interfaces," *IEEE JSAC* 11(2), pp. 203-219 (Feb. 1993).

[Roscoe 95] T. Roscoe, "The Structure of a Multi-Service Operating System", Ph.D. Thesis, University of Cambridge (August 1995).

[Shaffer 96] John H. Shaffer, "The Effects of High-Bandwidth Networks on Wide-Area Distributed Systems", Ph.D. Thesis, University of Pennsylvania, 1996.

[Simmons 90] M. L. Simmons and H. J. Wasserman, "Los Alamos Experiences with the IBM RS/6000 Workstation", Technical Report LA-11831-MS (March 1990).

[Smith 90] Jonathan M. Smith, "UPWARDS Operating System: Goals and Relevance to Supercomputing," *Lawrence Livermore Laboratories/IDA Supercomputing Research Center Workshop on Supercomputer Operating Systems*, Livermore, CA (July 10-12 1990).

[Smith 91] Jonathan M. Smith and David J. Farber, "Traffic Characteristics of a Distributed Memory System", in *Computer Networks and ISDN Systems*, **22**(2), September 1991, pp. 143–154.

[Smith 93] Jonathan M. Smith and C. Brendan S. Traw, "Giving Applications Access to Gb/s Networking," *IEEE Network* 7(4), pp. 44-52, (July 1993).

[Smith 99] J. M. Smith, J. D. Chung and C. B. S. Traw, "Interrupts", *Encyclopedia of Electrical and Electronics Engineering*, Volume 10, Wiley(1999), pp. 667-673.

[Thompson 78] K. L. Thompson, "UNIX Implementation," *The Bell System Technical Journal*, 6(2), pp. 1931-1946, (July-August 1978).

[Traw 91] C. Brendan S. Traw and Jonathan M. Smith, "A High Performance ATM Host Interface for ATM Networks", *Proceedings, ACM SIGCOMM Conference*, Zurich, SWITZERLAND, pp. 317-325 (September 1991).

[Traw 92] C. Brendan S. Traw and Jonathan M. Smith, "Implementation and Performance of an ATM Host Interface for Workstations", *Proceedings, IEEE Workshop on the Architecture and Implementation of High-Performance Communications Subsystems (HPCS '92)*, Tucson, AZ, pp. 101-104 (February 1992).

[Traw 93] C. Brendan S. Traw and Jonathan M. Smith, "Hardware/Software Organization of a High-Performance ATM Host Interface," *IEEE JSAC* 11(2), pp. 240-253 (Feb. 1993).

[Watson 87] Richard W. Watson and Sandy A. Mamrak, "Gaining Efficiency in Transport Services by Appropriate Design and Implementation Choices" *ACM Transactions on Computer Systems*, 5(2), pp. 97-120 (May 1987).

A Taxonomy and Design Considerations for Internet Accounting

Michel Kouadio*
mkouadio@acm.org
(Corresponding Author)*

Udo Pooch
pooch@cs.tamu.edu

Dept. of Computer Science, Texas A&M University,
501B Harvey Bright Bldg., College Station TX. 77843-3112.

ABSTRACT

Economic principles are increasingly being suggested for addressing some complex issues related to distributed resource allocation for QoS (Quality of Service) enhancement. Many proposals have been put forth, including various strategies from Pricing Theory and market-based insights for security in the Internet. A central need of these endeavors lies in the design of efficient accounting architecture for collecting, storing, processing and communicating relevant technical information to parties involved in transactions. This paper proceeds to a systematic classification of the major characteristics of accounting models with the aim of highlighting important features to optimize for building effective architectures.

Categories and Subject Descriptors:

C.2. Computer-Communication Networks. C.2.3. Network Operations: Network Management, Network Monitoring. C.2.4. Performance of Systems: Design Studies, Measurement Techniques. C.2.5. Local and Wide-Area Networks: Internet.

General Terms:

Measurement, Economics, Management.

Keywords:

Accounting Architecture, Pricing, Classification.

1. INTRODUCTION

The development of protocols for resource request and allocation is central to QoS provisioning efforts on the Internet. Two major architectures and protocols have been proposed to allow users to negotiate the availability and reservation of resources prior to initiating transactions.

The first was the Integrated Services Networks (IntServ) with the Reservation Setup Protocol (RSVP) [3], [36], and [43]. The second, the Differentiated Services Networks (DiffServ), requires that users negotiate a Service Level Agreement (SLA) that provides upfront their traffic expected profile and terms of service provision with the network [2], [29]. In DiffServ, SLA is negotiated through a Bandwidth Broker (BB), which is an entity in a domain that keeps track of all resource allocated, and accepts or rejects new requests accordingly. BB is also designed to cooperate with its peers in other domains to confirm resource availability from source to destination.

In recent years, economic models of network management have emerged as complements to these architectures, or alternative frameworks purporting to contribute flexible solutions to distributed resource allocation issues and enhance QoS. Network economic approaches are fundamentally frameworks based on Game Theory and Pricing Theory that require resource owner (producer) to set "rules of the game" and leave to each user (consumer) the selection of an algorithmic behavior that optimizes his utility [9], [16], and [44]. Pricing studies generally posit that putting a price tag on network services will moderate and lead to self-regulation in user requests, thus ultimately resulting in the equilibrium of supply and demand of resources.

Application of economic principles to network operations introduces specific challenges encompassing modeling, computational complexities and user-network interactions. How to operate a network by auctioning resources? What signaling protocol may allow users and networks to exchange information on resource supply and demand to sustain tight real-time QoS constraints? Efficient accounting mechanisms should therefore play an important role in this picture by appropriately providing the necessary technical information to assess resource availability and support different charging policies. We define an accounting architecture as the set of components for monitoring resource availability, and for collecting, storing, processing and communicating various parameter data that define user QoS profile and consumption of network resources.

This paper investigates issues in the design of accounting architecture that may be used to support economic models of network management. In Section 2, we highlight the relationship between accounting and pricing. Section 3 provides a review of some of the main accounting studies found in the literature. Next, in Section 4, we propose six main criteria around which to organize and understand accounting models. Then we go on from Sections 5 through 10, to explore the implications of each of these characteristics from an overall performance and efficiency perspective. We wrap up our classification in Section 11, by stressing important design considerations and elements to optimize for building an effective accounting architecture.

2. ACCOUNTING AND PRICING

A great deal of research in economic models for the Internet has concentrated on pricing policies and strategies. Tutorials on pricing and the economics of networks may be found in [10] and [19]. General surveys of pricing models are conducted in [11], and recent trends in charging studies for Integrated Internet Services are reviewed in [40]. While pricing studies deal mainly with policy and charging strategies, they are closely dependent on accounting architectures and protocols for collecting, processing and communicating the supporting technical data. Hence, accounting architecture appears to be an enabling technology for economic-based resource allocation and management in networking.

Figure 1 presents the relationships between pricing model, accounting architecture and the network. As an example, pricing strategies such as auction and dynamic pricing require that the underlying accounting monitor the availability of resource and communicate relevant information to parties involved in the transaction (e.g. network and user agents). Pricing policies and accounting mechanisms have reciprocal influences. Accounting architecture and protocols determine the scope of feasible pricing models, and pricing models set out features and requirements for accounting. IETF (the Internet Engineering Task Force) has created a working group in Authentication, Authorization and Accounting (AAA) to investigate some of the issues involved [1], [5], [6], [31].

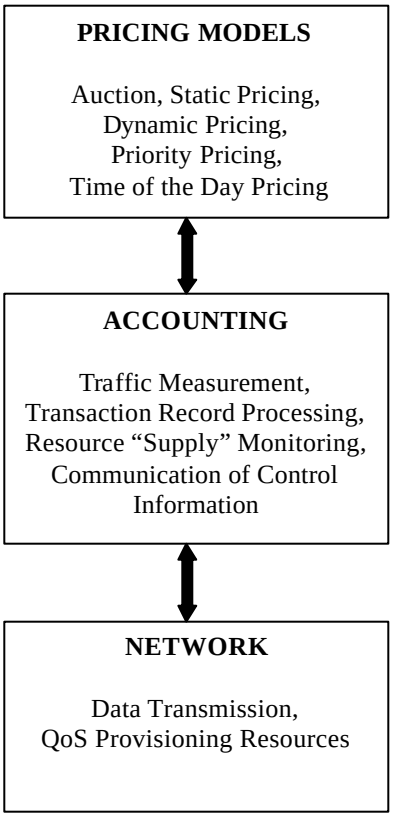


Figure 1: Layered Presentation of Pricing, Accounting and the Network.

3. REVIEW OF ACCOUNTING STUDIES

This review includes explicitly designed accounting models as well as some "implied" architectures put forth as part of pricing studies. Indeed, there is a natural interplay between pricing and accounting; and these fields being relatively new, their terminology is still evolving and is not standard in all studies.

3.1. Accounting for Optimal Pricing Models

Optimal pricing models aim to achieve maximal efficiency in the distribution of resources and maximal welfare in the use of network resources. They form the bulk of recent research in pricing and most are based on Game Theory [42]. The theoretical economical foundations of optimal pricing models is explained in [27] and [39]. Optimality refers to the perfect equilibrium of the supply of network resource and the demand by users, due to the regulating effect of the pricing strategy. Basically, the network is modeled as a supplier of QoS provisioning resources (bandwidth, buffer etc.), and users as consumers, are represented by agents, each endowed with a budget. The repeated interaction between the network advertising its resources (lightly loaded, overloaded etc.), and competing users adapting their strategies to optimize their QoS allocations is bound to reach an optimal point of convergence where each user is most satisfied with his gain and has no incentive to deviate for another QoS profile [16] and [24]. Example proposals are introduced in [9], [25], [30] and [44]. Typical strategies to reach optimality include dynamic pricing of congestion and auction [26], [28] and [34].

There is a diverse body of accounting mechanisms intertwined with these pricing models, to provide the architectural support. The architectures are generally not explicitly defined in these studies, but rather implied by the charging strategies. They share some common characteristics that follow from their features:

1. Resource monitoring: each network element (e.g. router, switch, and server) is endowed with mechanisms to collect data on aggregate resource consumption and availability (bandwidth, buffer, CPU), and sets conditions for release to users accordingly.

2. User-network signaling: there are typically two options that include interactive signaling like those used for auction models, or a one-way dynamic price advertising from the network. In the first option, a communication protocol allows network elements to advertise resource availability and current clearance conditions. Users submit resource requests and possibly their utility (valuation of transaction). The interaction continues until supply equals demands. The second option involves only the network setting clearance price and simply feeding it back to users who then may choose to adapt their QoS parameters accordingly. That was the option used in the network game simulator built at Microsoft Research and Cambridge [18] and [20]. The signaling load is often an issue with these models because of the frequent exchanges between users and networks, especially for auction-like strategies.

3. Data collection: typically data sent by users is measured in each network node and rated against the clearance price, based on the local resource availability conditions. At issue is the scalability of these mechanisms and their real-time performance when multiple nodes are involved from source to destination.

4. Network models: accounting architectures designed for these models may in principle work with most of QoS-enhanced protocols. Furthermore, they may be seen as alternatives to IntServ (RSVP) and DiffServ models of resource allocations. Reservation and classes of service are superseded by dynamic user appropriation of resources based on price level; therefore each application may be viewed as having their own class of service and continuously reserving the resources that they need. However, the communication model supported is unicast and does not extend to multicast.

3.2. Edge Pricing Architecture

In an attempt to deal with the practical scalability and flexibility issues of the accounting structures of optimal pricing models, Shenker suggested an architecture paradigm aimed at making accounting local to each network provider [35]. It is of interest to note that the induced architecture of optimal pricing models may require that uniform accounting mechanisms be implemented in different network domains. Such close inter-dependencies limits the flexibility that each network provider would have in the selection of different accounting schemes and charging models.

The main idea of the Edge Pricing architecture is to advocate that all accounting mechanisms be set up at the edge of each provider network, and appropriate cost approximations be performed to support different pricing models. Such approximations include the expected congestion conditions and traffic path from source to destination. That architecture fosters interconnection agreements between network providers, and has the advantage of breaking down the accounting complexity into local issues. The design approach is intended to provide flexibility in the scope of charging models that may be supported by each network domain.

Still, the Edge Pricing architecture is more a design framework than a detailed accounting model. It does not address either detailed data collection method, or data communication between parties. It left as open issues how to support models that charge receivers and multicast, which are not local issues to each provider.

3.3. Zone-based Cost Sharing Model

Clark is one of the first to consider cost sharing between sender and receiver [8]. The proposed accounting architecture recommends that the Internet be subdivided into zones where service would be provided at a uniform, distance-insensitive way. Zones might be site, city, region or country. Both sender and receiver would exchange signaling information on the zones for which they are willing to pay. That information is to be put into each packet header by the sending source when transmission starts. Each router is to be configured to examine the packet header to determine if any of the sender or the receiver is willing to pay for QoS resources in the zone where they are located, and to perform the relevant traffic metering. Therefore the accounting involves willingness-to-pay signaling messages, payment aware packet header and traffic measurement in routers along the path from source to destination.

In the case of multicast, in order to limit the load of signaling messages from receivers to the source, routers in the zone where the sender has expressed interest in paying, provide enhanced services and simply forward the packets. Then, as packets leave the sender-paying zone, routers in subsequent zones would mark packets to explicitly request from receivers whether they are willing to pay for enhanced services. Each receiver would be equipped with an agent "Receiver Traffic Meter" that makes decision on packet payment for zones of interest and would respond to such demands.

The significance of this architecture is best understood in relation with some popular activities on the Internet, such as web browsing, or emerging multimedia applications, such as videoconferencing and video-on-demand. As a matter of fact, in these cases, it is natural that pricing models target receivers of transactions, instead of charging the sender only. It is then important that accounting architectures provide mechanisms to support such cost sharing.

3.4. Internet Demand Experiment (INDEX)

Developed at UC Berkeley, INDEX is a field experiment that provides an architecture for charging and billing users for Internet traffic [12], [32]. The primary goal was to study user reaction to different QoS and usage sensitive pricing, as opposed to the currently flat pricing system used in the Internet. That study proposes exact data measurement over sampling for charging purposes to enhance credibility of collected data to users. The system is made up of many components. Users are equipped with a security agent that authenticates operations with the network and a purchasing agent that makes payment decisions. At the network entry point sits a billing gateway, a dedicated device such as a router that provides transaction metering and consolidates accounting records. An access controller coordinates user authentication procedures.

That architecture can support different static charging models, and provides a way to select which of the sender or the receiver to account for by explicitly asking confirmation from each end-user involved before allowing a transaction establishment. It deals with unicast traffic and does not support multicast. It is designed to work in the traditional Internet and does not extend to IntServ or DiffServ.

3.5. Generic Charging and Accounting (GenCA)

Departing from accounting schemes that are designed to support a few specific pricing models, Hartanto and Carle proposed a policy-based configurable accounting architecture for the European Information Infrastructure and Services Pilot, also known as SUSIE [21]. Their study developed a general architecture and billing system framework made of the following layers and functions:

- *Metering Layer:* tracks and records usage of resources.
- *Collecting Layer:* accesses data provided by meters and forwards them to accounting entities.
- *Accounting Layer:* consolidates the collected data into accounting records.
- *Charging Layer:* assesses costs of usage from the accounting records.
- *Billing Layer:* collects all charges for a customer over a time period and creates a bill.

In that framework, each new transaction by a user triggers the retrieval of his profile from a policy database to collect accounting and charging related parameters. These parameters are sent to metering entities to collect data relevant to the charging formula from the contract between the user and the provider. Typical parameters include volume, bandwidth, priority and duration. The data collected is sent to accounting consolidation entities to form records that later serve for charging.

GenCA accounting model is flexible and adaptable. Each user may have a custom-charging and service level agreement contract with a provider, and the accounting system is configured to measure, record and process traffic parameter data for the QoS resources in that contract. The system can support Integrated Services and Differentiated Services Networks. Provisions are made to support both unicast and multicast. That accounting model is bound for implementation in some European countries.

3.6. CATI (Charging and Accounting Technologies for the Internet)

CATI is a project developed in Switzerland as part of a charging program on Electronic Commerce transactions initiated by the Swiss National Science Foundation [17]. The accounting scheme encompasses many network protocols including Integrated and Differentiated Services Networks, ATM and MPLS. CATI accounting model uses pre-defined user profiles that contains default information on:

- Sender identity
- Default IP provider
- Payment method
- Invoice/billing information
- Default QoS
- Default cost-sharing scheme

Users have the opportunity to change some of the default parameters in their configuration file before initiating a transaction.

Two components are used in a network provider domain: Internal Service Broker (ISB) that manages service local to the provider and External Service Broker (ESB) that negotiates service with neighboring domains. Service Level Agreements (SLA) between a provider and his internal users or his peer providers are negotiated in form of volume of traffic, duration and price before the start of a session and can be dynamically updated.

Accounting signaling is performed through an extended RSVP protocol. The extension includes new objects on pricing scheme, identity of the payer of the transaction (e.g. sender or receiver), bidding fields where auction is applicable, fields for the type of service requested, and network provider information. The new RSVP objects extensions allow for example the implementation of auction pricing models by the use of bidding fields. That architecture was also used to implement dynamic pricing models between network providers, or Service Level Agreement Trading (SLAT), for an automatic negotiation of traffic volume for interconnection agreement.

3.7. Accounting and Charging Multicast

Herzog suggested a distributed architecture called "One Pass Accounting", consisting of an accounting message that flows from the source through the multicast tree [22]. As the message traverses the tree, multicast nodes allocate costs to connected members. The accounting scheme of this approach requires the configuration of each network node in the multicast tree for running the costing algorithm. It is designed to support charging models based on link costs, and therefore this study appears to be more relevant to integrated network environments with the RSVP protocol that works with pre-determined paths. This architecture may not easily be extended to support other network models, such as DiffServ, where traffic path may vary. Overall, a link-by-link accounting mechanism may appear limited in flexibility and scalability, due to the complexity of node configuration.

Another multicast accounting architecture was proposed as an extension of the Generic Accounting Architecture [7]. This model sets up metering devices in edge nodes and all multicast routers of each domain. For a given multicast session, data collected in these meters is fed back to the traffic sender provider domain where the accounting server proceeds to a consolidation and compute the allocation share of each participant.

In both studies, the accounting architectures are designed to work across domains, on the whole multicast tree, which require somehow uniform mechanisms and inter-domain standards.

4. CHARACTERIZING ARCHITECTURES FOR INTERNET ECONOMICS

Accounting models for network economics differ by many characteristics. The design principles, parameters handled and components involved are crucial to the overall efficiency of each model. A taxonomy provides a general picture that helps highlight main model components, patterns and possible points that deserve more attention and efforts.

Accounting models may have many characteristics and this taxonomy, although broad is not meant to be exhaustive. Remember that accounting characteristics may be as broad as the set of architectural requirements of all conceivable pricing models. Still, we believe that the most recurrent characteristics can be classified in six major concepts:

1. Policy scope: describes the different charging policies and economic principles that may be directly supported, or with minor adaptation, by the architecture. That often results from the design approach.

2. Source of data collection: specifies where in the network data is collected to feed the accounting system.

3. Methods of data collection: is a concept for understanding the granularity of data collection.

4. Collected elements: refers to the different kind of data measured and provided by the accounting model; this provides insights in the kind of charging and pricing policies that may be supported.

5. Interactivity: determines whether the accounting model has any provision to allow users to submit various parameters such as their valuation of utility and prices, or if all rules and parameters are set by network providers.

6. Supported network protocols: indicates the scope of enhanced network service protocols and thus environments where the accounting system may operate.

Figure 2 synthesizes the classification of these characteristics, while Sections 5 through 10, provide further explanations.

5. POLICY

The scope of charging policies supported by an accounting architecture is typically determined by its design principles. Accounting models may thus be subdivided into two major groups: those that are driven by charging and pricing policies, and those that start from architectural grounds.

5.1. Close-Policy Accounting Models

Most of the works reviewed in the Section 3, start with investigating some specific pricing policy to eventually reach an accounting model that may support the proposed model. This may be considered a top-down approach or accounting driven by pricing-policy. Specific economic principles or pricing policy permeate the architecture and therefore the scope of policies, which it may directly support, is limited to those it embodies. Such accounting systems can readily support the pricing model and policy they relate to. However, they are generally not very flexible insofar as substantial effort may be needed to adapt them to support other policies. Whether they are centralized or distributed, their design entails the support of a limited set of policies. That may be particularly so, if the model implementation involves multiple network nodes. Obviously, adapting the accounting mechanisms in all nodes involved to support other policies may be demanding for an internetwork the size of the

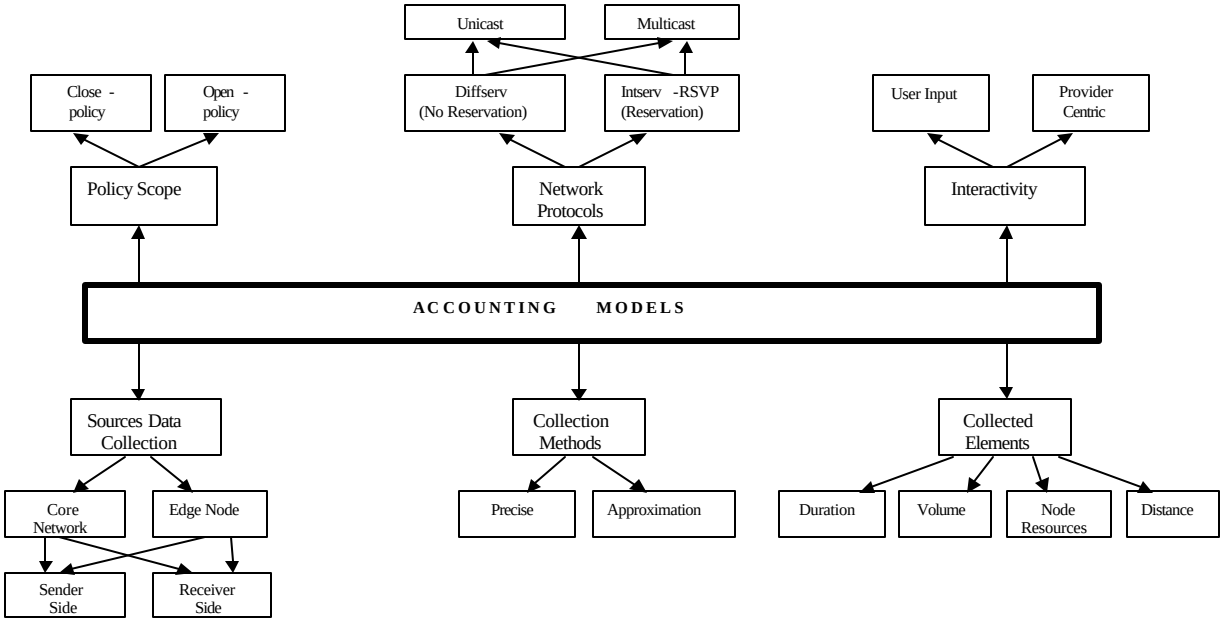


Figure 2: Taxonomy of Accounting Architecture Characteristics.

Internet. Hence, these models lean towards uniformity in accounting and policy across the Internet.

Such models may be called close-policy accounting models, because by virtue of their design, they are more or less explicitly geared to support one or a fixed set of policies. They may require, depending on the policies that they embody, minor or substantial change in the network architecture for their implementation. Examples include the induced accounting model of Smart-Market [28], One-Pass Accounting protocol for multicast suggested in [22], and some architectures for auctions and optimal pricing models [18] and [26].

5.2. Open-Policy Accounting Models

Accounting architecture may be independent of any specific pricing policy. Such model may be general so that a wide variety of economic models may be built on top of it. The scope of the pricing models supported, represents an important element that determines the flexibility of an accounting model. Accounting models whose design start by investigating architectural issues may be viewed as using a bottom-up approach. The advantage of such architectural-driven accounting models is that they fully integrate network architecture and protocol from the start. However, being independent of any specific policy, they may require systematic though limited adaptation work, in order to support any particular economic model. Since they generally provide a flexible foundation for supporting multiple pricing models, we will describe them as open-policy accounting models. Edge-Pricing architecture is an example of this category [35]. It recommends an architectural design that would allow every provider to implement at the edge of his network a local accounting suitable to his economic policies and leaves inter-domain accounting to interconnection agreements.

Other examples that may be viewed as open-policy accounting include Arrow [15], CATI [17] and Generic Charging and Accounting (GenCA) [21].

6. SOURCES OF DATA COLLECTION

An important aspect of an accounting model lies in the source of data collection. That is typically carried out in the edge or the core of the network, and may target one side of the transaction.

6.1. Network Nodes

6.1.1. Network Core

This requires the configuration of multiple, if not all network nodes (routers/switches) in a domain or the Internet. This approach has the advantage of providing very detailed measurements at all stages of the network topology. It may provide a basis for dynamic pricing models that are based on the general network status, the actual path of traffic and multicast cost allocation schemes that assess imputations on a link basis. However, it does not scale well and it may bring more complexity in the engineering of the network. The overhead incurred may be very high and that may adversely affect network performance. Systems based on this approach may be involved as far as implementation, maintenance and upgrade are concerned. They are not very flexible, because any change in the

mode of operation requires change in each node. Examples of accounting models that require data collection in the core of the network include the induced accounting models of Smart-market [28], congestion-based optimal pricing models [33], and that of "Link Weighting" charging model [13].

6.1.2. Network Edge

Data is collected from some few nodes that serve as interfaces between users and the rest of the network, typically ingress or/and egress nodes. All accounting and pricing decisions may thus be managed at that local level. That leaves the internal structure of the network less complex with no additional overhead. However, this approach does not provide enough detailed elements on operations involving inner nodes such as multicast branching points and congestion bottlenecks. In general, it does not readily provide useful information for pricing and cost allocation to be applied to receivers. A typical example is Edge-Pricing architecture [35].

6.2. Side of Data Collection

The source of data collection may further be concerned with the side of transaction where elements are collected; that may lie with either the sender or the receiver. The implications of this architectural decision have a direct impact on pricing policies that may thus be supported.

6.2.1. Sender Side

Accounting on the sender side is the most popular form found in the literature. It has the advantage that information about the originator of the traffic is readily available to the network provider before any transactions take place. That also helps support pricing systems with built-in incentives for efficient network resource usage at the source of transmission, and auction models that require frequent reactions from the traffic sender. The main limitation of accounting on the sender side lies with the difficulty of dealing with charging models involving protocols such as multicast where costs are to be distributed among all participants (receivers), and require collecting information on both sides. Examples of this category include the accounting architectures of auction models [26], [34], that of INDEX [32], and Edge-Pricing architecture [35].

6.2.2. Receiver Side

Setting up accounting mechanisms on the side of the receiver of Internet transactions, is an option mostly used in multicast studies. Accounting on receiver sides is also used in unicast models that suggest ways to charge the traffic initiator. The difficulty of such systems is that in most cases, receiver information is only available in a remote region or domain, and it is not readily available for use where the traffic starts. Accounting mechanisms may therefore involve significant signaling and trusted relationships among Internet Service Providers etc. An example of such models may be found in [38]. Zone-based Cost Sharing architecture [8] and GenCA [21] also have provisions for accounting on the receiver side.

7. METHODS OF DATA COLLECTION

The methods used to collect data are important to accounting models, because different approaches determine the granularity of the collected elements and have implications on audit capacities, as well as the production of billing records. A good design must minimize the level of complexity and overhead that its implementation may incur.

7.1. Precise-Collection

The accuracy and fine-grained collection of cost elements is important for providing a full picture of resource consumption to users and help providers assess the availability of resource supply. Yet, the accuracy of the collection adds up to the complexity of the accounting system and increases the overhead in the computational load brought onto the system, especially if the data collected is the packet volume of traffic or congestion status [4], [21].

7.2. Approximation

Traffic measurements may be approximated to avoid costly accurate collection. Sampling may be used to that purpose, by exploiting statistical properties of traffic. Data is collected at various frequency intervals and extrapolated to determine the approximate resource consumption of each user. Sampling is typically used for volume of traffic, the packets sent over the network. The main limitation of sampling stems from its inability to provide a detailed figure of transactions, which may be necessary in case of disagreement between providers and users.

Approximation may also take the form of estimating user consumption from overall parameters of a negotiated SLA. The concept of effective bandwidth is a case in point [23] and [37].

8. COLLECTED COST OBJECTS

The collected cost objects provide a measurement of the resources to be charged for. There may actually be a great variety of elements to be collected based on the potential range of charging policies. In this sense, close-policy accounting models have the advantage of narrowing down these elements since they are designed to support specific policies. Open-policy models that are general and not designed to support explicit policies may collect more elements than necessary in many instances, and may need proper configuration to avoid needless overhead. Still, from an architectural viewpoint, there are commonly selected elements that are very representative of those found in most accounting models: duration, volume, distance, and node resources. A combination of several of these elements may be found in an accounting architecture proposal.

A. Duration: this refers to measurement for the length of time that traffic from a given source uses the network resources. It is more applicable to reservation-based transactions such as RSVP in IntServ, where it is easier to predict the amount of resources used per unit of time. In DiffServ, accounting for duration may also suit transactions in "Expedited Forwarding" class, which provides virtually no delay. Collecting traffic duration is easy to carry out with only a modest level of overhead.

B. Volume: the count of bytes or packets sent over the network during a session by a transaction is a commonly collected cost object. Contrary to duration measurement, volume metering involves dealing with the actual load put on the network. Measuring volume may require detailed metering, which may be costly from an overhead perspective. Volume-based accounting is used in NZGate [4]. In practice, a combination of volume and duration is quite frequent in the form of rate (i.e. Rate=Volume/Duration).

C. Distance: this generally represents the number of hops traversed by packets from source to destination. Distance may be evaluated from the network topology. Geographical information and IP address may also be used, although Internet address structure is

generally not directly expressive. Another method for distance evaluation involves the exploitation of IP packet TTL (Time To Live), which is decremented anytime that a packet crosses a router. Distance evaluation for accounting is used in [13].

D. Network Node Resources: packets need to use some amount of node resources (CPU and buffer) in order to meet QoS requirements such as low delay, low jitter and specific Per Hop Behavior (PHB) in DiffServ. There are accounting models that collect the usage of those elements, to typically support routing optimization policy, congestion control and distributed resource allocations [14], [33].

9. SUPPORTED NETWORK PROTOCOLS

Accounting models may be designed for specific network protocols and environments. The two main Internet QoS protocols that have major different accounting requirements are IntServ (Integrated Services Networks) and DiffServ (Differentiated Services Networks). Along each of these network environments, the accounting architecture may also target unicast or multicast.

9.1. QoS Protocols

A. IntServ: accounting for Integrated Services Networks with the RSVP protocol involves dealing with reserved and guaranteed resources, and pre-determined paths. In such an environment, all traffic parameters are well known in advance (e.g. duration, volume, and class of service). Data collection is hence made easier. Therefore, these QoS reservation parameters may be collected for accounting purposes as soon as resource availability and reservation are confirmed. Examples of accounting models for Integrated Services networks include Arrow [15], OCIT [41], One-Pass Accounting [22].

B. DiffServ: in Differentiated Services Networks, resources availability is checked and confirmed either by Bandwidth Brokers or light-weight RSVP across domains. There is no formal reservation in the sense that these resources are made available to other traffic if and when they are not used. Therefore, in DiffServ data must be collected on the fly and that brings about more performance constraints. Accounting needs to be multiform to cope with the heterogeneous systems and policy requirements of the various interconnected domains. Since there is no strict reservation per se, and no guarantee of QoS provisioning, it is important to use various adaptation strategies to account for packets whose class of service gets demoted from one domain to another. Interconnection may therefore be more relevant and accounting models need to be flexible enough to accommodate all these strategies and policies. Some representative studies in this category may be found in [7] and [17].

9.2. Communication Mode

The Internet has two major mode of transmission with very different accounting requirements.

A. Unicast: most pricing studies have implicit accounting mechanisms for unicast. In this case, communication is on a one-to-one basis and the identity of parties involved, is available in the packet header. Other information may be fairly easily collected.

B. Multicast: accounting methods for this mode require a distributed collection of information on many users. However, the identity of participants is not directly provided by standard multicast protocols. Heterogeneous QoS requests and traffic merging over shared trees, require that a method be devised to collect and consolidate accounting data in order to determine the cost to be attributed to each user. Multicast accounting models are proposed in [7] and [22].

10. INTERACTIVITY

Communication among users on one hand, and between users and the network on the other, may be indispensable in some approaches and therefore various parameters need to be exchanged. For instance, charges may be negotiated or determined solely on the basis of the cost elements collected. The interactivity component of an accounting system provides the opportunity to exchange custom information.

A. User input: users may submit their valuation of utility and reach an agreement through interaction with the network provider. In this case, accounting protocols or mechanisms provide the necessary elements for collecting such inputs, process them and send feedback to users. User input may take various forms: a bid to have a transaction completed, adaptive resource requests such as a strategy in Game Theory or utility characterization. INDEX architecture [32], and CATI [17] have provisions for user input. The level of interactivity may be a source for overhead in the accounting scheme and deserves careful design.

B. Provider-centric: accounting systems may operate solely from rules determined by providers, and standardized parameters are indiscriminately collected from traffic flows. The network provider is thus the only one involved in determining resource allocation policies and conditions. Such an approach has the potential advantage of reducing the level of signaling characteristic of interactive systems, and may have better performance. Yet, that approach limits user choices and self-optimization strategies. For example, the lack of interactive feature in accounting scheme may preclude implementation of auction like models.

11. DESIGN ISSUES

The design of accounting architecture and protocol needs to meet some important performance requirements. The taxonomy helps highlight some desirable characteristics. These may be very challenging to achieve in practice. Each proposal may show strength in some areas and limitations in some others. We further propose minimum engineering design requirements that should not be overlooked in order to optimize efficiency and effectiveness.

11.1. Scalability

This characteristic of accounting mechanisms refers to the ability for a model to be extended to multiple network nodes, domains, users etc., without incurring any significant additional load or overhead. This feature is all the more important in a large and heterogeneous network like the Internet, that accounting systems should be able to handle the high growth in the user population and the domains. For example, an architecture to support dynamic pricing and auction, should reliably provide price information from

and to users, most probably within some deadline. How to design a real-time, reliable communication mechanism to reach all users without much overhead? The characteristics of the taxonomy that are more relevant to scalability are the source of data collection and collected objects.

11.2. Flexibility

The flexibility of an accounting model refers to its ability to be easily used or adapted to support various network and computing environments, as well as different charging circumstances (e.g. pricing policies). An accounting system flexibility may be greatly enhanced if it can support a wide range of the characteristics surveyed in the taxonomy. The issue for a provider, is to have a single accounting architecture, on top of which different pricing policies can be implemented simultaneously or as needed over time.

11.3. Complexity-Overheads

The more sophisticated accounting models are, the more likely complex they are, and they increase the computational load in the network. For instance, detailed data collection requires a lot of CPU power and memory to store accounting records. An important feature of accounting systems that may add up to the overhead, is the communication and signaling mechanism to support the exchange of various type of information. The volume of information exchanged and its frequency may have major impact on the accounting system performance. For instance, in auction pricing models, it is important to provide a way to transfer user bids, which may be carried by dedicated bidding messages or it may be embedded in a portion of the packet. The former scenario may induce a lot of messages while the latter may use a significant portion of the header or the payload. A highly efficient and effective accounting scheme should leverage existing network architectures and protocols for a transparent integration.

11.4. Robustness

This feature refers to the ability of an accounting system to sustain operation under heavy loads. The system should reliably run and remain stable under various circumstances. For instance, an accounting system that has a very high frequency of signaling with poor communication control may have its various components severed from each other at time of congestion; the various components may not be able to communicate properly and the system may produce unreliable results.

11.5. Security

Accounting systems should include mechanisms to insure the integrity of data collected and prevent unauthorized users from having access and manipulating the records. Accounting ultimately leads to pricing and models of resource allocation for improved QoS performance. The absence of appropriate security mechanisms may not only cause prejudice to users, but also hurt the provider by allowing unauthorized access to the network infrastructure without proper data collection. Security includes authorization and authentication of users; as well as the protection of the integrity of the network system.

12. CONCLUSION AND FUTURE DIRECTIONS

In this paper, we investigated some of the main features of accounting architectures and how they contribute to overall efficiency. The features considered are not exhaustive; nevertheless they are very relevant in building effective accounting architecture. The insights provided by this taxonomy lay the ground for understanding and assessing models.

The effectiveness and the efficiency of accounting models are crucial in implementing viable pricing and economic based resource allocation models in the Internet. The bulk of research in network economics has focused on pricing models, and accounting architecture is still in need of further investigations. The design of efficient accounting is all the more difficult that some of the design issues described in this paper are hard to optimize altogether. Design approach must hence perform some trade-off, and optimization may ultimately depend on the class of pricing models that are likely to be implemented in a specific domain or by a set of cooperating providers.

We see multicast accounting as a fertile research area. As a matter of fact, multicast was designed to help improve network performance by having users share a single copy of QoS intensive applications. From an economic and game theoretic perspectives, users will cooperatively go for multicast, only if the cost allocation schemes allow clear saving over unicast. The support of equitable and incentive compatible cost allocation methods, requires detailed accounting of each user contribution to the global cost. The issues are that members are anonymous, have heterogeneous QoS demands; traffic is aggregated, the multicast tree topology has several splitting points where traffic diverges, and the tree crosses several domains under different administrative authorities. Hence, sophisticated accounting is important to capture the different cost relevant parameters in order to extract the share of each participant. It is still an open issue on whether, and how to make multicast accounting a local issue to each domain, so that each network provider may have more choice in the selection of their accounting model and cost allocation schemes.

Guided by the considerations made in this paper, we designed a new token-based accounting architecture (TOKENAC), with the aim of optimizing several of the components that we identified. It is designed to support different static and dynamic pricing models, as well as multicast. We leverage the principle of token-bucket, used for traffic regulation to perform QoS accounting, and thus avoid dedicated traffic metering. That architecture is currently being tested through simulation.

13. REFERENCES

[1] Aboba B., Arkko J. and Harrington D. 2000. Introduction to accounting management. *Internet RFC 2975 (October), (http://www.rfc-editor.org/).*

[2] Blake S., Black D., Carlson M., Davies E., Wang Z. and Weiss W. 1998. An architecture for differentiated services. *IETF Internet RFC 2475* (Dec.), (URL: http://www.rfc-editor.org/).

[3] Braden R., Zhang L., Berson S. and Jamin S. 1997. Resource Reservation Protocol (RSVP)- Version 1, Functional Specification. *Internet RFC 2205* (September), (URL: http://www.rfc-editor.org/).

[4] Brownlee Nevil. 1996. New Zealand experiences with network traffic charging. *Journal of Electronic Publishing (JEP),* vol. 2, no. 1, (May), (www.press.umich.edu/jep/econTOC.html).

[5] Brownlee Nevil. 1997. Traffic flow measurement: architecture. *IETF RFC 2063* (January), (URL:http://www.rfc-editor.org/).

[6] Calhoun P., Akhtar H., Arkko J., Guttman E., Rubens A. and Zorn G. 2001. DIAMETER base protocol. *IETF Internet Draft*, (www.ietf.org/).

[7] Carle G.,Hartanto F., Smirnov M. and Zseby T. 1999. Charging and accounting for QoS-enhanced IP multicast. *Proceedings of IFIP Six International Workshop on Protocols for High-Speed Networks*, Salem, MA, (August 25-27).

[8] Clark David. 1996. Combining sender and receiver payments in the Internet. In *Interconnection and the Internet, Proceedings of Telecommunications Research Policy Conference 1996,* Gregory Rosston and Waterman (Eds.), Lawrence Erlbaum Pub.(1997), Mawah, New Jersey, pp. 95-112.

[9] Cocchi R., Shenker S., Estrin D. and Zhang L. 1993. Pricing in computer networks: motivation, formulation and example. *IEEE/ACM Transactions on Networking*, vol. 1, no. 6 (December), pp. 614-627.

[10] Courcoubetis Costas. 1998. Pricing and the economics of networks. *Infocom'98 Tutorial, San Francisco, USA, March 29-Apr.2*.

[11] DaSilva Luiz A. 2000. Pricing for QoS-enabled networks: a survey. *IEEE Communications Surveys and Tutorials,* vol. 3, no. 2, Second Quarter 2000, pp. 2-8.

[12] Edell R., Mckeown N. and Varaiya P.P. 1995. Billing users and pricing for TCP. *IEEE Journal on Selected Areas in Communications,* vol. 13, no. 7 (September), pp. 1162-1175.

[13] Einsiedler H. J. and Hurley P. 1998. Link weighting: an important basis for charging in the Internet. *Technical Report SSC/1998/017*, Communication Systems Division, EPFL (Swiss Polytechnic University, Lausanne), (URL:http://lrcwww.epfl.ch/PS_files/publications.html).

[14] Fankhauser G. and Plattner B. 1999. DiffServ bandwidth brokers as mini-markets. *International Workshop on Internet Services Quality Economics*, MIT, Cambridge, Massachusetts (December 2-3).

[15] Fankhauser G., Stiller B. and Plattner B. 2000. Arrow: a flexible architecture for an accounting and charging infrastructure in the next generation Internet. *NETNOMICS: Economic Research and Electronic Networking,* vol. 1, no. 2, pp. 201-223, (URL: www.baltzer.nl/netnomics/contents/1999/1-2.html.

[16] Ferguson D.F, Nikolau C., Sairamesh J. and Yemini Y. 1996. Economic models for allocating resources in computer systems. In Market Based Control of Distributed Systems, Scott Clearwater (Ed.), World Scientific Press, River Edge, New Jersey, pp. 156-183.

[17] Foukia N. and Billard D. 1999. Charging and accounting technology for the Internet: the CATI project. *6th OpenView University Association WS (HPOVUA'99)*, Bologna, Italy (June 12-15), (www.hpovua.org/PUBLICATIONS/PROCEEDINGS/6_HPOVUAWS/).

[18] Ganesh A. and Laevens K. 2000. Dynamics of congestion pricing. *Technical Report MSR-TR-2000-70,* Microsoft, Research, Cambridge (June), (http://research.microsoft.com/research/network/).

[19] Gibbens R. and Key P. 2000. Distributed control and resource pricing. *Tutorial ACM SIGCOMM 2000,* Stockholm (Aug. 29). (Also:http://research.microsoft.com/users/pbk/GKSig2kResPriceTutorial_files/frame.htm.)

[20] Gibbens R. and Key P. 2001. Distributed control and resource marking using best-effort routers. IEEE Network, vol. 15, no. 3 (May), pp. 54-59. (See also: http://www.research.microsoft.com/network/).

[21] Hartanto Felix and Carle George. 1999. Policy based billing architecture for Internet differentiated services. *Proceeding of IFIP International Conference on Broadband Communications*, BC'99 (November), Hong Kong.

[22] Herzog S., Shenker S., and Estrin D. 1997. Sharing the cost of multicast trees: an axiomatic analysis. *IEEE/ACM Transactions on Networking*, vol. 5, no. 6 (December), pp. 847-860.

[23] Kelly F.P. 1996. Notes on Effective Bandwidth. In Stochastic Networks: Theory and Applications, F. Kelly et al. (Eds.), pp. 141-168 Oxford University Press, 1996.

[24] Kelly F.P. et al. 1998. Rate control in communication networks: shadow prices, proportional fairness and stability. *Journal of Operational Research Society*, vol. 49, pp. 237-252.

[25] Key Peter and Derek McAuley. 1999. Differential QoS and pricing in networks: where flow control meets game theory. *IEE Proceedings Software,* vol. 146, no. 2 (March), pp.39-43.

[26] Lazar A.A. and Semret N. 1997. Auctions for network resource sharing. *CTR Technical Report No. 468-97-02,* Columbia University, New York, (URL: http://comet.columbia.edu/publications/techreports.html).

[27] Mackie-Mason and Varian H.. 1995. Pricing congestible network resources. *IEEE Journal on Selected Areas in Communications,* vol. 13, no. 7 (Sept.), pp.1141-1149.

[28] Mackie-Mason and Varian H. 1995. Pricing the Internet. In Public Access to the Internet, Brian Kahin and James Keller (Eds.), MIT Press 1995, Cambridge MA, pp. 269-314.

[29] Nichols K., Jacobson V., and Zhang L. 1999. A two-bit differentiated services architecture for the Internet. *IETF Internet RFC 2638 (July),* (http://www.rfc-editor.org/).

[30] Orda A. and Shimkim N. 2000. Incentive pricing in multi-class systems. *Telecommunication Systems*, vol 13, no. 2/4, pp. 241-267.

[31] Pras A. et al. 2001. Internet accounting. *IEEE Communications Magazine*, vol. 39, no. 5 (May), pp. 108-113.

[32] Rupp B., Edell R.,Chand and Varaiya P. 1998. INDEX: a platform for determining how people value the quality of their Internet access. *Sixth International Workshop on Quality of Service (IWQoS 98)*, Napa-California (May 18-20), pp. 85 –90.

[33] Sairamesh J, Ferguson D. and Yemini Y. 1995. An approach to pricing, optimal allocations, and quality of service provisioning in high-speed networks. *Proceedings of the IEEE INFOCOM'9* (Apr. 2-6), Boston MA, vol. 1, pp.1111-1119.

[34] Semret N., Campbell A. and Lazar A. 1999. Market pricing of differentiated Internet services. *IEEE Proceedings of 7th International Workshop on Quality of Service (IEEE/IFIP IWQOS'99)*, London,UK, pp. 184-193.

[35] Shenker S., Clark D., Estrin D. and Herzog S. 1996. Pricing in computer networks: reshaping the research agenda. *ACM Computer Communication Review*, vol. 26, no. 2 (April 1996) pp. 19-43. Also in *Telecommunications Policy Research Conference*, vol. 20, no.3 (April), pp. 183-201.

[36] Shenker S., Partrige C. and Guerin R. 1997. Specification of guaranteed quality of service. *IETF Internet RFC 2212 (September)*, (URL: http://www.rfc-editor.org/).

[37] Songhurst D.J. (Ed.). Charging Communication Networks: from Theory to Practice. Songhurst D.J. (Eds.), Elsevier (1999), Amsterdam-Holland, pp-23-36.

[38] Sprenkels R., Parhonyi R., Pras A. Beijnum B. and Goede L. 2000. An architecture for reverse charging in the Internet. *IEEE Workshop on IP-Oriented Operations and Management (IPOM 2000)*, Cracow, Poland (September).

[39] Stahl Dale O. 1995. A critical survey of Internet pricing proposals. *Proceedings of OECD Workshop on the Economics of Information Society*, Istanbul, Turkey (December 14-15).

[40] Stiller B., Fankhauser G., Plattner B. and Weiler N. 1998. Charging and accounting for integrated Internet services: state of the art, problems, and trends. *INET'99: The Internet Summit* (July 21-24), Geneva-Switzerland.

[41] Stiller B., Fankhauser G., Joller G.,Reichl P. and Weiler N. 1999. Open charging and QoS interfaces for IP telephony. *INET'99:The Internet Summit* (June 22-25), San Jose CA.

[42] Von-Newman John and Oskar Morgenstern. 1944. Theory of Games and Economic Behavior. Princeton University Press, Princeton-New Jersey.

[43] Wroclawski J. 1997. Specification of the controlled load network element service. IETF Internet RFC 2211 (September). (URL: http://www.rfc-editor.org/)

[44] Yaiche H., Mazumdar R. and Rosenberg C. 2000. A Game theoretic framework for bandwidth allocation and pricing in broadband networks. *IEEE/ACM Transactions on Networking*, vol. 8, no. 5 (October), pp. 667-678.

Implementation Experience with MANET Routing Protocols

Kwan-Wu Chin, John Judge, Aidan Williams and Roger Kermode
Sydney Networks and Communications Lab
Motorola Australia Research Centre
12 Lord St, Botany, NSW, Australia 2019
{*kwchin, johnj, aidan, rkermode*}*@arc.corp.mot.com*

ABSTRACT

This paper outlines our experience with the implementation and deployment of two MANET routing protocols on a five node, four hop, network. The work was prompted by the lack of published results concerning the issues associated with the implementation of MANET routing protocols on actual wireless networks, as opposed to results of simulation experiments. We examined implementations of two distance vector MANET routing protocols and found a number of problems with both protocols during the course of our experiments. The most significant was that neither protocol could provide a stable route over any multi-hop network connection. The route discovery process of both protocols is fooled by the transient availability of network links to nodes that were more than one hop away. Packets transmitted over a fading channel cause the routing protocol to conclude incorrectly that there is a new one hop neighbor that could provide a lower metric (hop count) route to even more distant nodes. This can occur even when nodes are stationary, mobility resulted in even less route stability. We implemented a simple signal strength based neighbor selection procedure to test our assertion that fading channels and unreliable network links were the cause of the failure of the routing protocols. The result was that neighbor discovery and the filtering for neighbors with which nodes could communicate reliably enables the creation of reliable multihop routes. Based on our experiences, we outline several recommendations for future work in MANET research.

1. INTRODUCTION

The term ubiquitous computing was coined by Mark Weiser to describe a state of computing in which users are no longer aware of computation being done [28]. The emergence of smart environments, where devices are embedded pervasively in the physical world, has sparked many new research areas and represents a step towards ubiquitous computing. To this end, researchers have begun to outline plans to achieve ubiquitous computing. For example, Basu et al. [3] advocate the vision of *power-up-n-play* for smart environments in which no predefined infrastructures are installed and, when powered up, the devices "intelligently" configure and connect themselves to other devices. Bhagwat et al. [4] also focus on the interoperability of sensor devices and present three research issues: (1) distributed algorithms for self-organizing devices, (2) packet forwarding, and (3) Internet connectivity.

Mobile **a**d-hoc **net**work (MANET) routing protocols play a fundamental role in a possible future of ubiquitous devices. Current MANET commercial applications have mainly been for military applications or emergency situations[25]. However, we believe that research into MANET routing protocols will lay the groundwork for future wireless sensor networks and wireless plug-n-play devices. The challenge is for MANET routing protocols to provide a communication platform that is solid, adaptive and dynamic in the face of widely fluctuating wireless channel characteristics and node mobility.

The paper discusses our experience while implementing and deploying two distance vector MANET routing protocols. We examined both a public domain implementation of the Ad Hoc On-Demand Distance Vector (AODV) [21] routing protocol and implemented our own version of the Destination-Sequenced Distance Vector (DSDV) [20] routing protocol. The choice of routing protocols was pragmatically based on what (little) was available at the time this work was carried out. The AODV implementation was the freely available MAD-HOC implementation [15]. This implementation was based on an earlier draft of the AODV protocol and includes some MAD-HOC specific extensions. Where AODV is referred to in this paper we mean the MAD-HOC implementation unless otherwise stated. At the time our work was carried out this was the only public domain MANET routing protocol implementation that had a license suitable for our use and that we could get to compile, run and work on our network. Faced with no other available public domain code and reluctant to base our work solely on one protocol implementation we coded an alternative. DSDV was chosen due to it's relative simplicity and the fact that it is a table based protocol rather than an "on demand" protocol like AODV. Our implementation was based largely on the paper by Perkins et al. [20].

Both protocols were deployed on a five hop, four node testbed based on Linux workstations and 802.11b wireless LAN cards configured to use the Lucent ad hoc mode. We found that neither protocol could provide stable multihop network routes. The main cause was the failure of the route discovery processes in provisioning for unreliable links which are inherent in wireless channels. The route discovery process was fooled by transient link availability with nodes that were too distant for reliable communication to take place. A couple of routing packets sent over this link is enough to temporarily fool the routing protocol into assuming a more direct (lower

hop count) route exists to the desired destination.

To test the assertion that transient link availability was the cause of the failure of the routing protocols we developed a signal quality based neighbor selection program called *powerwave*. The inclusion of *powerwave* for neighbor selection stabilized multi-hop routes for both routing protocols to the point where they could carry useful amounts of user data.

A number of extensive simulation studies on various MANET routing protocols have been performed by various researchers [25][5][16][8][7]. However, there is a severe lacking in implementation and operational experiences with existing MANET routing protocols. Previous implementation experiences include wireless Internet gateways (WINGS) [11], implementation of ODMRP [2], AODV implementation by Royer et al. [24] and ABR implementation by Toh et al. [27]. These studies only highlighted performance issues specific to the protocol being used. By far the most extensive implementation study to date was conducted by Maltz et al. [17] in describing their implementation of DSR.

Unlike previous work, our work reports on the experience of building an operational ad-hoc network that is capable of carrying useful data. We report several interesting observations not reported elsewhere for the use of MANET protocols within pico-cell environments. It is worthwhile noting that this paper's objective is to report on the operational feasibility of existing routing protocols and efforts undertaken to create a reliable ad-hoc network. In many ways this is a step back towards fundamental issues and away from the MANET routing protocol aspects usually examined in simulation studies. Whereas simulation studies commonly report on performance metrics such as throughput, latency and packet loss this paper reports on the fundamental issue of "do MANET routing protocols work". The answer is yes but, in the case of the two distance vector protocols we examined, only if the inherent unreliable and transient nature of wireless network links are taken into account.

This paper is organized as follows. In Section 2 we provide a brief summary of AODV and DSDV. This is followed by implementation details of both these protocols in Section 3. In Section 4 we describe the testbed used for our experiments. Section 5 presents the problems and observations gained from setting up the testbed and running the routing protocols over it. In Section 6, we present the workings of *powerwave*. Based on our experience with MANET routing protocols, we discuss issues and problems encountered in relation to existing routing protocols and propose some future directions in Section 7. Finally, the conclusions are presented in Section 8.

2. BACKGROUND

In this section we review the workings of the AODV and DSDV MANET routing protocols. Comprehensive reviews of other routing protocols are available in [25],[12] and [5].

AODV is characterised as an on-demand (also called reactive) routing protocol. Routes are created as needed at connection establishment and are maintained for the duration of the communication session. During route discovery a node broadcasts a route request (RREQ) message for a given destination address. Nodes that have a route to the destination respond to the RREQ by sending a route reply (RREP) message to the source and record the route back to the source. Nodes that do not have a route to the destination rebroadcast the RREQ message after recording the return path to the source. In the event of link breakage a route error (RERR) message is sent to the list of nodes (referred to as precursors) that rely on the broken link. Upon receipt of a RERR message, the corresponding route is invalidated and a new RREQ may be initiated by the source to reconstruct the route [21]. The time-to-live (TTL) field is used in RREQs for an expanding ring search to control flooding. Successive RREQs use larger TTLs to increase the search for destination node.

Unlike AODV, DSDV [20] is a table-driven (or proactive) routing protocol and is essentially based on the basic distributed Bellman-Ford routing algorithm [1]. Each node in the network maintains a routing table consisting of the next hop address, routing metric and sequence number for each destination address. To guarantee loop free operation, routing updates from a given node are tagged with a monotonically increasing sequence number to distinguish between stale and new route update messages. Nodes periodically broadcast their routing tables to neighbouring nodes. Given sufficient time, all nodes will converge on common routing tables that list reachability information to each destination in the network. Route updates are generated and broadcast throughout the network when nodes discover broken network links. Nodes that receive a route update check to see if the sequence number specified in the route update message is higher than the sequence number recorded in their own routing table before accepting the update. DSDV reduces routing messages overheads by supporting both full and incremental updates of routing tables.

The main characteristic of table-driven protocols is that a route to every node in the network is always available regardless of whether or not it is needed. This results in substantial signaling overhead and power consumption [25]. Furthermore, table driven protocols transmit route updates regardless of network load, size of routing table, bandwidth and number of nodes in the network [5]. Interested readers are referred to Toh et al. [25] for a qualitative comparison based on simulation experiments between flavors of both on-demand and table-driven routing protocols.

3. ROUTING PROTOCOL IMPLEMENTATIONS

This section presents implementation details of the AODV and DSDV protocols used in our experiments and provides a background to the discussions and observations which will follow regarding the deployment and implementation issues we have encountered.

3.1 MAD-HOC Implementation of AODV

The AODV routing protocol used in our experiments was implemented by the MAD-HOC group [15] and can be obtained from *http://mad-hoc.flyinglinux.net* [1]. There are two

[1]At the time of our experiments there were two publicly available MANET routing protocols, CMU's DSR and MAD-HOC's AODV. We chose MAD-HOC's AODV over

main components to the MAD-HOC implementation: (1) *packet_capture* and (2) *aodv_daemon*.

The *packet_capture* program captures packets that traverse the network interface and triggers the *aodv_daemon* when particular packets are seen. The capture mechanism is implemented using the libpcap library [14]. Three types of packets are of interest: address resolution protocol (ARP) packets, Internet control message protocol (ICMP) packets and Internet protocol (IP) packets. Un-answered ARP requests from a host indicate that a route to a given destination is required, *packet_capture* extracts the destination IP address from the ARP packet, and passes the address to the *aodv_daemon*. *aodv_daemon* then generates a route request for the destination. When an ICMP message is parsed *packet_capture* determines whether the ICMP message received is of type ICMP DEST UNREACH, ICMP UNREACH HOST or ICMP UNREACH HOST UNKNOWN. If the message matches the above ICMP types, the *aodv_daemon* is notified of a link breakage to a given destination address. All other ICMP messages are ignored. When a link break is detected, the *aodv_daemon* issues a route error message to all hosts using the broken link. The source address of data packets intercepted by *packet_capture* are passed directly to *aodv_daemon* to update the route lifetime which the data packets arrived on. The MAD-HOC AODV implementation used hello messages, periodic broadcasts, to maintain a local connectivity list.

The main problem with the MAD-HOC AODV implementation was that buffering was not performed while route construction was in progress. In practical terms, we found that a *telnet* session had to be initiated multiple times before a session could be established. When running ping over a four hop route, with the default one second gap between successive pings, the first five packets were usually lost before the route was successfully established.

3.2 DSDV Implementation

The second routing protocol we chose to experiment with was DSDV. The choice was made due to DSDV's simplicity, thus enabling us to easily code up and debug the operation of DSDV on our testbed. DSDV's simplicity proved valuable during our experimentation especially when explaining the poor operation of DSDV on our testbed.

Our DSDV implementation was based on the ACM SIGCOMM'94 paper by Perkins et al. [20] with the addition of a neighbor handshake protocol to check for bi-directional links. Our DSDV implementation used the Multi-threaded Routing Toolkit (MRT) [19] for platform independence and for interfacing with the kernel routing table, socket and file input/output (IO). In addition, MRT also provided some convenient data structures for holding information regarding machine interfaces and utilities for manipulating IP addresses. Due to the small scale of our testbed, the incremental update aspects of DSDV were not implemented (all the routes could easily fit in the one packet). The hysteresis timers were also not implemented as we did not have many alternate routes of the same hop count.

CMU's DSR due to extensive documentation, and hardware and operating system compatibility with our testbed.

3.2.1 The SEEN Metric and State

The original paper describing DSDV [20] specified that DSDV assumes bi-directional links but does not include any mechanism for ensuring a link was bi-directional before a route was put in place. It was found that such a mechanism was crucial with fading channels. We extended DSDV through the inclusion of a handshake protocol that makes use of the SEEN metric to signal that a new neighbor had been detected.

The SEEN metric was defined as an integer value outside the range of one to INFINITY[2]. DSDV nodes advertise a route to a node with $metric = SEEN$ on the reception of a packet from a neighbor for the first time. All other nodes, apart from the node listed as the route destination, ignore this route. On receiving a routing advertisement for itself with a $metric = SEEN$ a node makes and advertises a route to the sending node. Nodes will only advertise a route to another node with a SEEN metric for a short period of time, if no reciprocal route advertisement is received then the SEEN state times out and the route is no longer advertised. The signaling process used in the discovery of a bi-directional neighbor using the SEEN metric is illustrated in Figure 1.

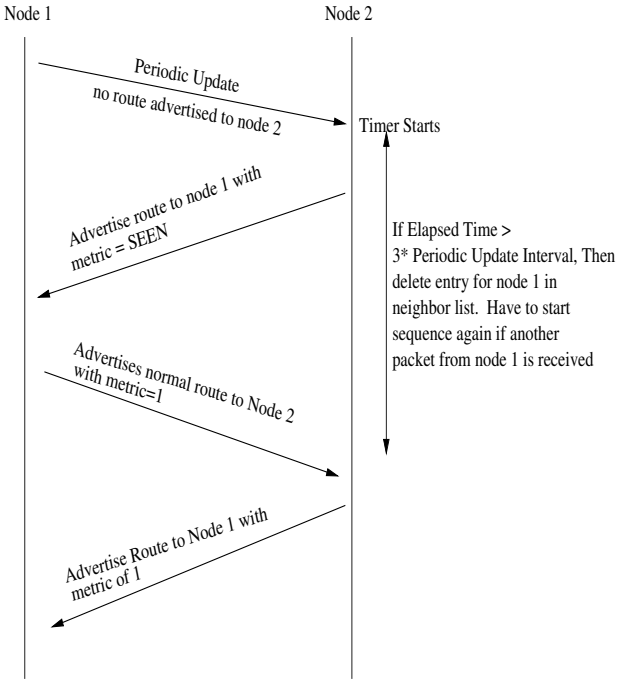


Figure 1: DSDV's Signaling Process Using SEEN Metric, No Existing Route Between Node 1 and Node 2

4. TESTBED

Figure 4 shows the network topology of our testbed. Our testbed consisted of two notebooks and three desktop computers, equipped with Lucent Wavelan IEEE 802.11b PCMCIA cards and running Linux (Debian with 2.2.15 kernel). We used version 6 of the Linux driver from Lucent for the IEEE 802.11b cards, with the transmit rate set to 1 Mb/s

[2]INFINITY itself was defined as 16 and is used to signal that a destination is no longer reachable

and the operation mode set to ad-hoc[3]. The lowest channel rate was chosen to avoid the cards stepping down transmission rates automatically (a feature that we could not otherwise disable). The cards were configured to transmit on an otherwise unused channel to avoid interference from other IEEE 802.11b devices in our lab. To limit the transmission range, we wrapped each card with a metallic anti-static bag. As a result, we managed to drop the transmission range from 250 meters to approximately five meters. This enabled us to create a four hop network in our lab and avoid the problem of having to locate the experiment in a large field.

It is important to note that the anti-static wrapping did not alter the radio propagation characteristics of an indoor office environment consisting of soft partitions. The observed radio propagation behavior, i.e., Rayleigh Fading, of the testbed is consistent with Hashemi [13]'s study on indoor radio propagation models. Figure 2 and 3 show a comparison of the signal-to-noise ratio as measured on our testbed and that of Rayleigh fading respectively. As can be seen, both experimental and theoritical model agrees, hence the anti-static wrapping did not alter the fading behavior of the channel which contributes to transient links. Readers who are interested in indoor radio propagation models and Rayleigh fading are referred to [13] and [23].

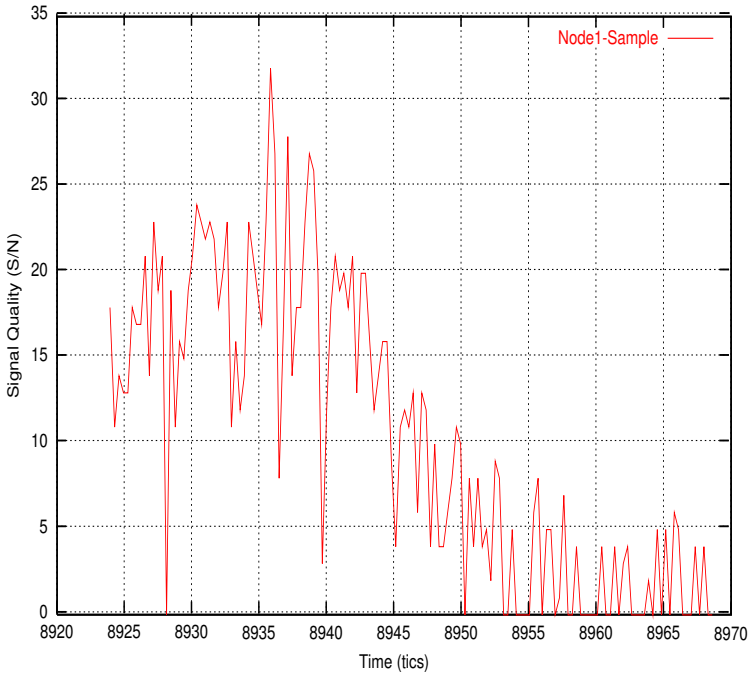


Figure 2: Signal Level Measurements from Our Testbed, Soft Partition Office.

In order to verify we have a working ad-hoc network we ran the following experiments. The first experiment consisted of an application residing on mobile host 2 (MH_2) that transmits UDP packets to the *discard* service on MH_1. We then monitored the number of packets transmitted and received as MH_2 moved along the line of hosts toward, or away from $MH1$. Motion towards MH_1 was referred to as "downstream" while motion away from MH_1 was referred

[3]Lucent's propriety ad-hoc mode

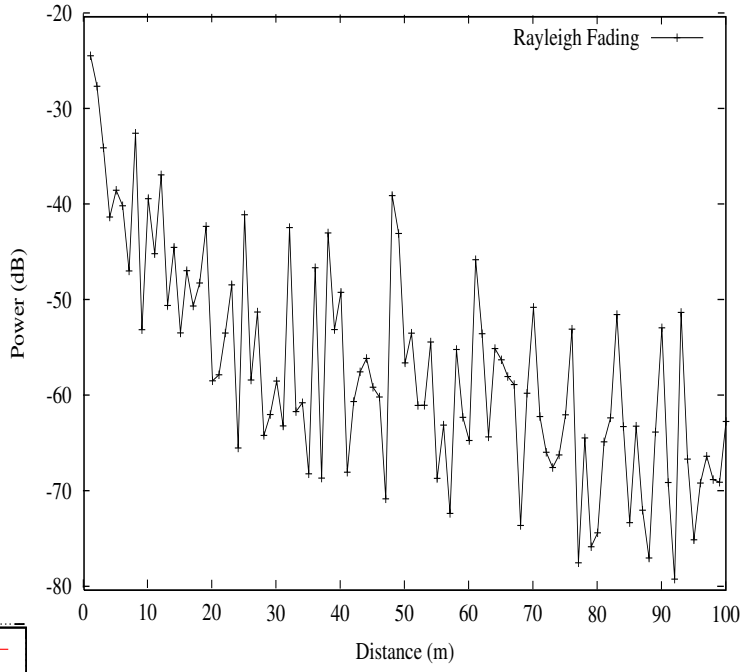


Figure 3: Rayleigh Fading. The figure was generated by calculating the received power when two nodes starting at distance 0m, and then calculating their received power after moving them apart at increment of 5m.

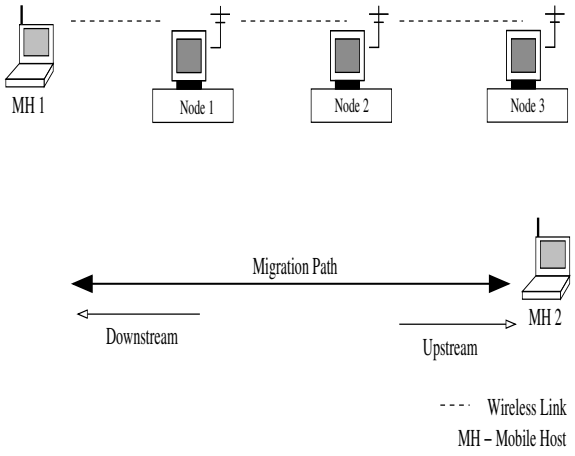


Figure 4: Testbed Topology

to as upstream. In the second experiment, we performed file transfers (using FTP) between MH_1 and MH_2. In our experiments no other sessions were present and the network traffic in our experiments consisted entirely of data transfer between the mobile nodes and routing messages. Moving MH_2 along the line of nodes exercised the adaptive features of the routing protocols. The nodes were placed such that MH_2 should route packets through each of $node1$, $node2$ and $node3$ in turn as it is moved upstream. Each of the fixed nodes was placed so that it could communicate reliably with adjacent neighbors but could not send or receive packets reliably to the other more distant fixed nodes.

5. EXPERIMENTAL OBSERVATIONS

5.1 Fading and Transient Network Links

It was found that transient radio links resulted in poor operation of both the routing protocols examined where no reliable routes could be established. The poor operation was due to the creation and maintenance of routes without taking the stability, or quality, of the network links comprising the route into account. The fundamental problem was that successful transmission of a datagram over a wireless network link is probabilistic, regardless of lower level protocols. In practice this probabilistic effect became evident in two ways; occasional dropped packets on a normally "good quality" network link and occasional successful packet transmissions on a normally "poor quality" network link. We found that the occasional dropped packet did not present much of a problem for either of the routing protocols examined. On a "good" network link the link layer acknowledgements in 802.11 replaced lost unicast packets and the routing protocols appeared to be able to handle the occasional lost broadcast, or multicast, packet. In contrast the occasional appearance of a channel between two nodes that could not normally communicate was disruptive to the routing protocols on our testbed. The problem manifested itself in the creation of network routes that were not suitable for the reliable transmission (and reception) of user data. These routes were chosen over other route options by the protocols selecting for lowest hop routes, regardless of any sort of measure of route quality. As stated in the introduction a similar effect for the DSR routing protocol has been observed on another testbed [18].

We found that it was practically impossible to establish a stable *telnet* session between nodes over a three or four hop route on our testbed. For example when using the topology described in Figure 4, we found that $Node_1$ could still detect $Node_3$'s signal occasionally despite careful placement and orientation. As a result we observed that both nodes would randomly receive a packet from the other. If AODV was engaged in a route building process it would use the unreliable one hop route from $Node_1$ to $Node_3$ in preference to the two hop alternative. DSDV would replace the existing two hop route between the nodes with the unreliable one hop route. Very little user data would be transmitted over this unreliable route and user sessions would hang pending the reestablishment of the more reliable two hop route.

In a related work, Maltz et al. [17] reported similar behavior while building a MANET testbed and experimenting with Dynamic Source Routing (DSR) routing protocol. The following modifications to DSR were suggested to overcome the problem of routing over unreliable links: (1) monitor route error on links, (2) use the geographic positioning system (GPS) to determine the neighbor proximity (assuming physical proximity will provide the best channel) and (3) combine GPS with route error monitoring. Reliability was tested over a three node, two hop network with the nodes arranged in a line. The network included packet filtering software to prevent packets from being transmitted directly from one end node to the other. They found that an FTP file transfer between the end nodes was more reliable when the packet filtering software was enabled. Ramanathan et al. [22] also reported problems with transmission range when testing out their quality of service (QoS) based routing protocols. However, no solutions to unreliable links were suggested.

Published articles reporting on MANET routing protocol performance often rely on simulation experiments. Experiments run on our testbed uncovered considerable difference in the probability of successfully receiving packets on a MANET node versus the probability of successful packet reception in some simulation environments. In a simulation environment, such as *ns-2* [10], it is generally assumed that the probability of receiving a packet is effectively one (pending collisions etc) and once a node moves out of another node's signal range, or a given distance, this drops to zero. However, our experiments have shown that this is unrealistic; signals tend to decay slowly and there is no cutoff point. We suspect that the use of simplistic radio propagation models in MANET simulation environments has led to inaccurate assessments of the performance of various routing protocols, especially those which utilize hop count as the dominant route selection metric. Thus, one area for future work is the incorporation of better radio propagation models that support channel fading and other inputs to the probabilistic nature of wireless channels. For example, Rappaport [23] lists a number of factors that affect fading in an in-door environment such as multi-path propagation, mobile node speed, surround object speed and signal bandwidth.

5.2 Handoff in a MANET

In conventional cellular networks, the signal-to-noise ratio (SNR) of the connection between mobile phone and base stations is monitored to determine when to hand off from one base station to another. In a MANET, current protocols do not predict when a link's SNR will fall below a threshold. The periodic HELLO messages in AODV and route update timers in DSDV are not used to anticipate hand off, they indicate presence or absence of a neighbor node. Consequently, the route maintenance process at both AODV and DSDV is only initiated after link breakage already ocurred.

DSDV behaves differently depending on the mobile nodes direction of movement. DSDV pro-actively changed to a lower hop count route if one was available, but hung on to a route until it is explicitly broken should a lower hop count route not be available. The effect with DSDV was smooth handover when MH_2 (in Figure 4) was moving downstream but no handover in the upstream direction.

In the upstream direction two things would prompt a new (higher hop count) route to be used. First, the connection to the previous fixed node would have to timeout prompting

a switch to the next best available route being advertised by the new neighbor. Or second, the link between the previous fixed node would have to break along with a route advertisement being received from the new neighbor with a higher hop count and a higher sequence number. The new sequence number would then invalidate the old route and cause the new route to be used instead.

5.3 AODV Specific Issues

5.3.1 Pico cell size and AODV's timers

A problem encountered were AODV's default parameters. Since the transmission range of each node was reduced in our testbed to less than 5m, we had in effect constructed a network with pico sized cells. In this environment the default MAD_HOC AODV timers unnecessarily prolonged route construction and required tunning before an acceptable performance could be achieved. The parameters we changed are listed on Table 1. AODV's parameters as specified in [21] are left to the implementors, however recent drafts have used more conservative parameters than those in the MAD-HOC implementation shown in Table 1.

BCAST_ID_SAVE is used to prevent over flooding of RREQ messages. When a new RREQ is intercepted, the information within the RREQ is recorded and the information is added to an interval queue along with a time interval (current time plus BCAST_ID_SAVE). In the event of another RREQ appearing within this time interval, the RREQ is discarded.

RREQ_RETRIES bounds the number of RREQs for a given destination. The default value is two. We found this value to be too conservative, and found that five was more appropriate value.

ACTIVE_ROUTE_TIMEOUT is used to determine the lifetime of a given route. The lifetime of each route maintained by a given node is refreshed after observing data packets or HELLO messages on that route. In a pico-cell environment, the default value needs to be small. In our testbed where nodes moved at slow walking pace, the time for a node to traverse given cell was around five and we found a route timeout value of one second was appropriate.

Both NODE_TRAVERSAL_TIME and NET_DIAMETER had to be modified to suit our network topology. The NODE TRAVERSAL TIME was modified to increase the route construction time. The default value of NET_DIAMETER was set to 35 nodes and this was changed to five to reflect the number of nodes in our testbed.

The last parameter to be modified was ALLOWED HELLO LOSS which determines how many HELLO messages are lost before a link is considered broken. Routes were timing out frequently in our testbed and we set the ALLOWED HELLO LOSS parameter to five to increase stability.

The optimization of AODV by changing the parameters to suit our testbed was done on a trial and error basis. To date there are no published guidelines or heuristics for setting AODV's parameters or adapting them to a given network. The parameters shown in Table 1, and the other AODV parameters that have been defined in the AODV specification [21], would most likely have to be modified for use in other networks.

5.3.2 ARP Interactions

The reliance of the MAD-HOC AODV implementation on sniffing ARP packets to signal the need for route construction led to two problems. The first problem was that packets were not buffered while the route was being built. As mentioned in Section 3 this led to packets being dropped and the need to start an application such as *telnet* a number of times before a route was actually built. The second problem was that a route will never be constructed if there is an entry in the ARP cache. Spurious ARP cache entries exist for one or more reasons. Either the two nodes in question had once been adjacent, and the ARP cache entry had yet to time out, or an ARP reply was un-expectedly received from a remote node (over an unreliable link) and the cache then prevented a more reliable route being found.

One work around to these problems was to regularly flush the ARP cache and to start applications multiple times while waiting for the route building process to complete. In practice this would be achievable by using ping and waiting for a successful reply before starting the intended application. A better solution is the one proposed in [24] that uses a *netlink* socket to communicate routing information with the kernel space and a dummy route for buffering data packets pending route construction.

5.4 DSDV

5.4.1 Route Stability

The first thing we noticed about our DSDV implementation was its relative stability compared to the MAD-HOC's AODV implementation. DSDV was less affected by unreliable connections to distant nodes. This was mainly due to the use of the SEEN metric (requiring a handshake before the link would be used in routes) and less interaction with the ARP cache as the routing table was pre-populated with host routes (negating the need to ARP).

However DSDV was adversely affected by transient link availability. Even when all the network nodes were stationary the routing table would slowly "churn" as routes were constructed to distant nodes and then timeout.

6. SIGNAL QUALITY BASED NEIGHBOR SELECTION

Our observations/experiments showed that the main shortcoming with both AODV and DSDV to be a failure to handle the unexpected availability of a channel to a distant node. The subsequent use of one hop links to distant neighbors resulted in unreliable routes over which very little user level data could be sent. The cause of this problem was the failure of the routing policy deamons in each node to differientiate between "good" and "bad" one hop neighbors. We hypothesized that if nodes could filter for reliable one hop neighbors and use only these neighbors as next hop gateways, the resultant routes should be reliable.

To verify our hypothesis we implemented a neighbor selection based on signal strength (called *powerwave*). We found that its use resulted in reliable multi-hop connections on our

Parameters	Default Values	New Values
BCAST_ID_SAVE	30000ms	3000ms
RREQ_RETRIES	2	5
RREP_WAIT_TIME	$(3 \times NODE\_TRAVERSAL\_TIME \times NET\_DIAMETER)/2$	No Change
NODE_TRAVERSAL_TIME	100ms	10ms
NET_DIAMETER	35	5
ACTIVE_ROUTE_TIMEOUT	9000ms	1000ms
ALLOWED_HELLO_LOSS	2	5

Table 1: *MAD-HOC's AODV Parameters*

testbed, and proves that neighbor selection is desirable and probably necessary in MANET environments.

6.1 Signal Based Route Selection

The *powerwave* implementation of neighbor selection was developed to be transparent to the routing protocol and used packet filtering to block routing messages from neighbors deemed unreliable. With neighbor selection we wanted to identify nodes one hop distant to which packets could be reliably sent and and make these available to the routing daemon.

Operating as a sublayer beneath the routing protocols assisted routing protocols in selecting routes over reliable network links. Our aim was to provide a generic neighbor discovery framework that we could use to test implementations of MANET routing protocols.

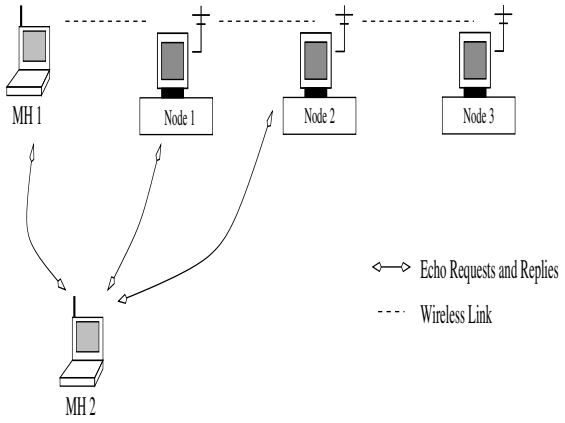


Figure 5: Measuring Signal Quality

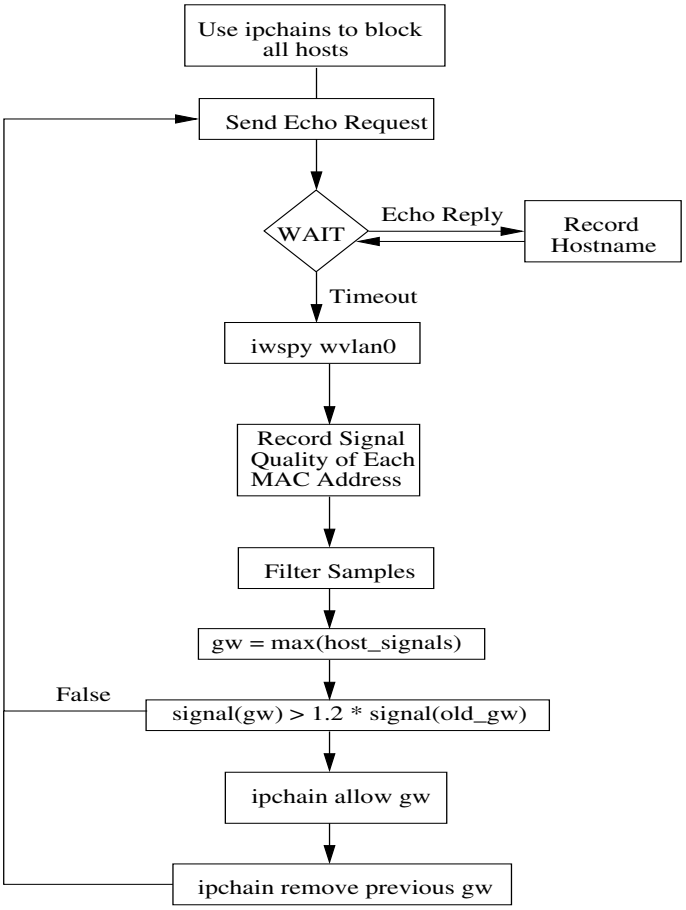


Figure 6: Flow-chart for Powerwave (Mobile)

Figure 6 shows the workings of our *powerwave* program on the mobile node. The value 1.2 was derived from measuring the signal strength on our testbed and determining an appropriate threshold that constitutes good signal strength. Before the program starts, the following *ipchains* rule is executed to filter out all messages (for AODV):

```
ipchains -A input -p udp -d 255.255.255.255 1303 -s 0.0.0.0 -j DENY
```

After the ipchains rule has been executed, echo requests were broadcasted and the SNR of replies were gathered. The signal strength associated with each link-layer address was then recorded and averaged. Averaging was required due to the random nature of a single SNR sample. Figure 7 shows raw SNR samples versus a moving average. The 'best' gateway[4] to route packets through was calculated based on previously recorded signal quality compared to current signal quality for each responding node. Note that the signal qualities used for comparison were averaged values. We tried using a fixed threshold value (20 dB) to determine the change of gateway. However, we found that due to the varying signal quality from multiple nodes, the choice of gateway tended to fluctuate frequently. Simply using a threshold value on the received signal quality was not effective and we found it did not yield reliable routes. Once the best gateway to route packets through was found, the following *ipchains* rule was executed (for AODV) to allow HELLO messages from the gateway:

```
ipchains -R input 1 -p udp -d 0/0 1303 -s ! %s -j DENY
```

We found that the *powerwave* program was also required at

[4]Next hop node through which to route outgoing packets

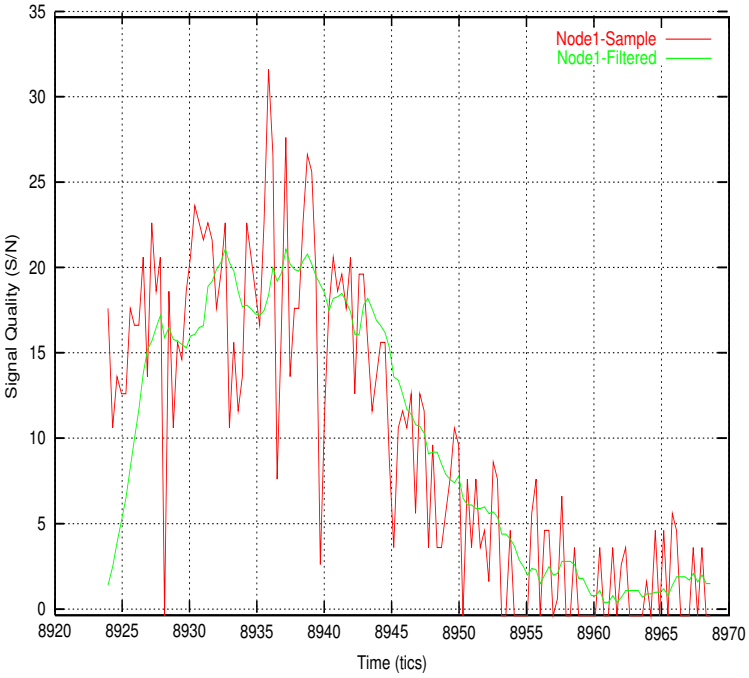


Figure 7: Sample vs. Filtered Signal Quality

stationary nodes in our testbed[5]. To ensure reliable links to their neighbors and more importantly to filter out HELLO messages from MH_2 that were transmitted over unreliable links. The reasons why *powerwave* was required on the static nodes were as follows. During route construction, a node downstream may have a shorter hop count, due to HELLO messages from MH_2, hence a RREP would be returned directly to MH_2 instead of being routed through the designated gateway. Since MH_2 ignores RREP messages from all nodes except for the designated gateway, MH_2 would then conclude that a route to MH_1 was impossible, resulting in the cancellation of the route construction process.

Powerwave programs running on stationary nodes required the following modifications:

- *Ipchain rules.* In the static nodes, specific rules were used to block out HELLO messages from non-neighboring nodes. For example, $Node_2$ (from Figure 4) only needs to listen to $Node_1$ and $Node_3$. The corresponding *ipchain* rules used to block out the appropriate nodes on $Node_2$ (for AODV) were:

```
# clean everything out
ipchains -F
# default deny
ipchains -A input -p udp -d 0/0 1303 -j DENY
ipchains -I input 1 -p udp -s node1 --dport 1303 -j ACCEPT
ipchains -I input 1 -p udp -s node3 --dport 1303 -j ACCEPT
# set up rule to be replaced blocking AODV from mobile
ipchains -I input 1 -p udp -s 10.1.0.100 --dport 1303 -j DENY
ipchains -L
```

  The *ipchains* configurations shown above are static which is unrealistic in a MANET where all nodes may move. However, the above rules can be adapted easily to moving nodes by imposing them dynamically using sampled SNRs of packets from neighboring nodes.

- *No echo request broadcast.* Echo requests were not needed since each node can read the signal quality of the echo request emitted by MH_2.

- *Interested in MH_2 only.* In our experiments, stationary nodes were only interested in receiving packets from MH_2. Once MH_2 is in range (quality above a given threshold) an *ipchains* rule was executed to allow routing packets to be passed to the routing daemon.

- *Thresholding.* The thresholding mechanism at stationary nodes was different to how thresholding was done at MH_2, where a fixed value was used instead of using a percentage of the averaged signal quality over time. To determine the threshold value at $Node_1$ to $Node_3$ and MH_1, graphs of SNRs collected from *powerwave* program were plotted. From these graphs, we determined a suitable threshold value, 10 dB. Thus if the signal quality of MH_2 exceeded 10 dB, *ipchains* was executed to allow the receipt of packets from MH_2. This threshold value was an arbitrarily selected value that was dependent on our network configuration. Determining an adaptive method that does not use thresholding is the subject of future work.

The *powerwave* program suffers from two shortcomings: (1) inefficient bandwidth consumption, and (2) inefficient interaction with AODV and DSDV. In the first case, *powerwave* on MH_2 broadcasts a continuous stream of echo messages in order for it (and other nodes) to measure the signal strength of packets received from each node. This increases contention time of other nodes wishing to transmit thereby reducing throughput of the network. In the second case, *powerwave* relies on blocking of HELLO messages from "bad" neighbors. Merely blocking routing messages leaves detection of broken links to the protocol timers. In future revisions, *powerwave* will signal the loss of a neighbor and also the appearance of a new neighbor directly to the routing protocol. Thereby routing protocols can be made aware of link-breakages and new neighbors in a timely manner.

While AODV and DSDV choose routes based on hop count, there are some MANET routing protocols such as SSA [9] that choose routes based on signal quality. Our experience with *powerwave* showed that a signal quality based routing protocol has to incorporate some form of stability metric after a route has been established to avoid the transfer of route as soon as a better signal link becomes available.

A similar approach to *powerwave* was also taken by Maltz et al. [18] where a program called *macfilter* was developed to filter out traffic from unwanted MAC addresses. A novel usage of *macfilter* was the emulation of a MANET where multiple nodes could be placed closely together and the signals from neighboring nodes filtered appropriately to give a different topology. The main difference between *macfilter* and *powerwave* is that *powerwave* uses SNR to dynamically determine which IP addresses to filter out whereas *macfilter* is statically configured for the topology in question.

An interesting conclusion from Matlz et al.'s work was that

[5]See Figure 4

they found neighbor selection to be important [18]. Our work further reinforces this believe, and we envisage more research work in the development of neighbor selection in MANET research.

7. DISCUSSIONS AND FUTURE WORK

7.1 Unstable Links

The majority of MANET routing protocols described in the literature were designed to handle topology changes and do not take unreliable links into account. Currently, only signal stability based adaptive routing (SSA) [9], ABR [26], and longest life routing protocol (LLRP)[29] support the notion of reliable routes. The route metrics use by SSA are average signal strength and route stability. By using these route metrics, packets will always be routed through the most reliable route (possibly closest node). Thereby route reconstruction cost is reduced and reliability of established route increases [9].

Unlike SSA, ABR only use route stability as the routing metric. Route stability is defined as the number of HELLO messages observe from a given neighbor. Hence, a neighbor with a given HELLO message count is considered stable. In both SSA and ABR, the destination has to choose the best route to take from a number of alternatives recorded from the various route requests received [29]. Further, once a route is setup there are no considerations for degraded links along the route. Routes are only rebuilt once they are broken.

The immediate future work is to re-evaluate existing hop based routing protocols with the addition of unreliable links.

7.2 Smooth Handoff

The notion of smooth handoff in MANET routing protocols has generally been overlooked. Improvements may be made by intelligently monitoring surrounding neighbors and determining whether a given node is able to prime an upstream/downstream node with a route to the destination. We found that a relatively smooth handover could be achieved by generating regular RREQs from MH_2. In other words, when a node detects a new neighbor a special message could be sent to prime the new neighbor, with routes to other new receiver nodes without waiting for existing routes to break.

Pro-active route construction will cause unnecessary traffic and duplicate routes which may then lead to the difficulty of removing invalidated routes. Further, the problem becomes more complicated if mobility is taken into account. Unlike traditional one hop wireless networks (e.g., cellular) where base-stations are fixed, the handoff decisions in MANETs are much more complicated.

It is interesting to note that the *powerwave* neighbor selection process had the side-effect of enabling a degree of handoff. The neighbor selection process filtered out neighbors before the network link disappeared entirely. User datagrams could still be forwarded over the link while the routing policy engine was finding a new route. It worked in our implementations because the routing parameters and the rate at which MH_2 moved matched.

7.3 Topology Dependent Parameters

Our experiments showed that the protocol parameters in both MAD-HOC's AODV and DSDV required some tuning before they would work properly. The determination of suitable timer values depended on channel rates, network topologies and mobility patterns [8]. The impact of these parameters on the performance of upper layer protocols is left for future work.

One method to allow for adaptive parameters is to introduce additional information. Protocols may rely on GPS, for example location aided routing protocols, to gather more information such as network topology and nodes proximity. Once the range of adjacent nodes are estimated, parameters may be adjusted accordingly.

7.4 Neighbor Selection Sub-Layer

The Internet MANET encapsulation protocol (IMEP) [6] is a mechanism to aggregate and encapsulate control messages. Also, IMEP provides a generic multi-purpose layer containing various common functionalities for MANET routing protocols. However, in the IMEP specification no consideration for signal strength was presented. It may be possible to use IMEP for filtering neighbors based on link stability rather than just to list neighbors that are in range.

Given the observations obtained from our experiments, one possible area of work is to extend upon IMEP's functionalities to incorporate mechanisms to shield wireless defects, and also offer various routing metrics which could be used by routing protocols.

8. CONCLUSION

In this paper we have outlined our implementation and deployment experiences with MAD-HOC's AODV and DSDV. Our experiments have provided insights into the real world deployment of MANETs and highlight issues that require further investigation. These are:

1. *Handling unreliable/Unstable links.*
2. *Minimizing the dependacy on topology specific parameters.*
3. *Mechanisms for handoff and reducing packet loss during handoff.*
4. *Incorporating neighbor discovery and filtering into a neighbor selection sub-layer.*

The first issue is a result of the current prevailing MANET protocol development/testing environments which appear to consist almost entirely of simulation experiments using *ns-2* and Glomosim. In implementing two MANET routing protocols, rather than simulating them, we discovered that the variability of networking conditions in the radio environment was such that the routing protocols did not work as reported in the literature. This led to the development of *powerwave*, and it was found that neighbor selection is crucial in the operation of MANET routing protocols. We believe our observations pertaining to unreliable/unstable links are not restricted to MAD-HOC's AODV implementation given that current AODV specification relies on hop

count and does not take into account the reliability of a given route or link.

The second issue is specific to a given routing protocol. As argued, having pre-configured parameters for a given topology is inappropriate given the inherent dynamic nature of MANETs, and affects the operation of routing protocols. Therefore, methods for adaptive adjustment of these parameters are required.

On the third issue, current MANET routing protocols do not appear to consider pre-emptive route construction based on signal strength in a similar way to how handoffs are done in cellular networks. We have observed that knowing whether a node is going upstream or downstream has added benefit. The concept of handoff, from one route that has a high probability of near term breakage to another route which is more stable is a possible area for future research.

Finally, there is scope for the development of a neighbor selection sub-layer like IMEP that incorporates a range of metrics that could be used by routing protocols. Various filters and heuristics could be developed which will be beneficial to MANET routing protocols.

9. ACKNOWLEDGEMENTS

The authors would like to thank the reviewers for the constructive feedbacks on the presentation and content of this paper.

10. REFERENCES

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows, theory, Algorithms, and Applications*. Prentice-Hall, 1993.

[2] S. H. Bae, S.-J. Lee, and M. Gerla. Unicast performance analysis of the ODMRP in a mobile ad-hoc network testbed. In *Proceedings of IEEE ICCCN'2000*, Las Vegas, USA, 2000.

[3] P. Basu and T. D. C. Little. Task-based self-organisation in large smart spaces:issues and challenges. In *DARPA/NIST/NSF Workshop on Research: Issues in Smart Computing Environment*, Atlanta, USA, 1999.

[4] P. Bhagvat, C. Bisdjikian, P. Kermani, and M. Naghshineh. Smart connectivity for smart spaces. In *DARPA/NIST/NSF Workshop on Research: Issues in Smart Computing Environment*, Atlanta, USA, 1999.

[5] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad-hoc network routing protocols. In *Proceedings of the 4th ACM/IEEE Internation Conference on Mobile Computing and Networking (MOBICOM'98)*, Dallas, Texas, Oct. 1998.

[6] M. S. Corson and V. Park. An internet MANET encapsulation protocol (IMEP) specification. Internet Draft: draft-ietf-manet-imep-spec-00.txt, Nov. 1997.

[7] S. R. Das, R. Castaneda, and J. Yan. Simulation based performance evaluation of mobile, ad hoc network routing protocols. In *Proceedings of Seventh International Conference on Computer Communications and Networks (ICCCN'98)*, 1998.

[8] S. R. Das, C. Perkins, and E. M. Royer. Performance comparison of two on-demand routing protocols for ad-hoc networks. In *Proceedings of IEEE INFOCOM'200*, Tel-Aviv, Israel, 2000.

[9] R. Dube, C. D. Rais, K.-Y. Wang, and S. K. Tripathi. Signal stability based adaptive routing (SSA) for ad-hoc mobile networks. *IEEE Personal Communications*, 4(2):36–45, Feb. 1997.

[10] K. Fall and K. Varadhan. The VINT project. *ns* notes and documentation. http://www.isi.edu/nsnam/ns/.

[11] J. J. Garcia-Luna-Aceves, D. Beyer, and T. Frivold. Wireless internet gateways (WINGS). In *Proceedings IEEE Milcom'97*, Monterey, CA, 1997.

[12] M. Gerla, G. Pei, and S. J. Lee. Wireless, mobile ad-hoc routing. In *IEEE/ACM FOCUS*, New Brunswick, USA, May 1999.

[13] H. Hashemi. The indoor radio propagation channel. *Proceedings of the IEEE*, 81(7), July 1993.

[14] Lawrence Berkeley National Lab. Libpcap: User-level packet capture library. ftp://ftp.ee.lbl.gov/libpcap-0.4.tar.Z, Feb. 1997.

[15] F. Lilieblad, O. Mattsson, P. Nylund, D. Ouchterlony, and A. Roxenhag. MAD-HOC AODV Implementation. Telecommunications Systems Lab, Techical Report. http://fl.ssvl.kth.se/.

[16] D. A. Maltz, J. Broch, J. Jetcheva, and D. B. Johnson. The effects of on-demand behavior in routing protocols for multi-hop wireless ad-hoc networks. *IEEE Journal on Selected Areas in Communications special issue on mobile and wireless networks*, Aug. 1999.

[17] D. A. Maltz, J. Broch, and D. B. Johnson. Experiences designing and building a multi-hop wireless ad-hoc network testbed. Techical Report, CMU-CS-99-11, Mar. 1999.

[18] D. A. Maltz, J. Broch, and D. B. Johnson. Lessons from a full-scale multihop wireless ad hoc network testbed. *IEEE Personal Communications*, 8(1), Feb. 2001.

[19] Merit Network Inc. Multi-threaded routing toolkit. MRT Programmers Guide. http://www.merit.edu/mrt/mrt_doc/.

[20] C. Perkins and P. Bhagvat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. *ACM Computer Communications Review*, pages 234–244, Oct. 1994.

[21] C. E. Perkins, E. M. Royer, and S. R. Das. Ad hoc on-demand distance vector (AODV) routing. draft-ietf-manet-aodv-06.txt, July 2000.

[22] R. Ramanathan and R. Hain. An ad hoc wireless testbed for scalable, adaptive QoS support. In *Proceedings of IEEE WCNC'2000*, Chicago, IL, USA, 2000.

[23] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice-Hall, 1996.

[24] E. M. Royer and C. Perkins. An implementation study of the AODV routing protocol. In *Proceedings of the IEEE Wireless Communications and Networking Conference*, Chicago,IL, Sept. 2000.

[25] E. M. Royer and C.-K. Toh. A review of current routing protocols for ad-hoc mobile wireless networks. *IEEE Personal Communications*, 6(2):46–55, Apr. 1999.

[26] C.-K. Toh. Associativity-based routing for ad-hoc mobile networks. *Wirelss Personal Communications Journal*, 4(2), Dec. 1997.

[27] C.-K. Toh and M. Delawar. Implementation and evaluation of an adaptive routing protocol for infrastructureless mobile networks. In *IEEE International Conference on Computer Communications and Networks (ICCCN'2000)*, Las Vegas, USA, Oct. 2000.

[28] M. Weiser. The computer for the 21st century. *Scientific American*, 265(3):94–104, Sept. 1991.

[29] S.-C. M. Woo and S. Singh. Longest life routing protocol (LLRP) for ad hoc networks with highly mobile nodes. In *Proceedings of IEEE WCNC'2000*, Chicago, IL, USA, 2000.

Efficient Micro-Mobility using Intra-domain Multicast-based Mechanisms (M&M)

Ahmed Helmy
Electrical Engineering Department
University of Southern California
helmy@usc.edu

Muhammad Jaseemuddin
Electrical and Computer Engineering
Ryerson University
jaseem@ee.ryerson.ca

Ganesha Bhaskara
Electrical Engineering Department
University of Southern California
bhaskara@usc.edu

Abstract

One very important metric in evaluation of IP mobility protocols is handover performance. Handover occurs when a mobile node changes its network point-of-attachment. If not performed efficiently, handover delays, jitters and packet loss directly impact applications and services. With the Internet growth and heterogeneity, it becomes crucial to design efficient handover protocols that are scalable, robust and incrementally deployable. Mobile IP (MIP) has been shown to exhibit poor handover performance during micro-mobility. We propose a new architecture for providing efficient and smooth handover, while being able to co-exist and inter-operate with other technologies. Specifically, we propose an intra-domain multicast-based mobility architecture, where a visiting mobile is assigned a multicast address to use while moving within a domain. Efficient handover is achieved using standard multicast join/prune mechanisms.

Two approaches are proposed and contrasted. The first introduces the concept of proxy-based mobility, while the other uses algorithmic mapping to obtain the multicast address of visiting mobiles. We show that the algorithmic mapping approach has several advantages over the proxy approach, and provide mechanisms to support it.

Simulations used to evaluate our scheme and compare it to other micro-mobility schemes - CIP and HAWAII. The proactive handover results show that both M&M and CIP show low handoff delay and packet reordering depth as compared to HAWAII. The reason for M&M's comparable performance with CIP is that both use bi-cast in proactive handover. M&M, however, handles multiple border routers in a domain, where CIP fails. Also using a proactive path setup mechanism, we show that M&M clearly outperforms CIP in case of reactive handover.

1. Introduction

The growth of mobile communications necessitates efficient support for IP mobility. IP mobility addresses the problem of changing the network point-of-attachment transparently during movement. When the mobile node moves away from its current network point-of-attachment, *handover* is invoked to choose another suitable point-of-attachment. In such an environment, handover latency and mobility dynamics pose a challenge for provisioning of efficient handover.

Several studies [1][8] show that Mobile IP [3], the proposed standard, has several drawbacks ranging from triangle routing and its effect on network overhead and end-to-end delays, to poor performance during handover due to communication overhead with the home agent. Several micro-mobility approaches attempt to modify some mechanisms in Mobile IP (MIP) to improve its performance [4][5]. However, as we will show, such approaches suffer from added complexity and, in general, do not achieve the best handover performance.

We follow a different approach to IP mobility using multicast-based mobility *(M&M)* [1]. In such paradigm, each mobile node is assigned a multicast address to which it joins through the access routers it visits during its movement. Handover is performed through standard IP-multicast join/prune mechanisms. Such approach, however, is not suitable for inter-domain IP mobility, for several reasons. First, the architecture requires ubiquitous multicast deployment, which is only partially supported in today's Internet. M&M should be designed for incremental deployment, and to allow co-existence with other IP mobility protocols. Second, the multicast state kept in the routers grows as the number of mobile nodes becomes larger. This problem may be alleviated using state aggregation [38]. Third, allocating a globally unique multicast address for every mobile node requires a global multicast address allocation scheme, and wastes multicast resources. Furthermore, mobile nodes incur security delay with every handover, which may overshadow architectural mechanisms that attempt to reduce handover delays.

To alleviate these problems, we propose new schemes for *intra-domain* multicast-based micro-mobility that allow for incremental deployment. In this architecture, a mobile node is assigned a multicast address within a domain for use with *micro* mobility. The allocated multicast address is locally scoped (i.e., unique only domain-wide). This allows for domain-wide address allocation schemes. Packets are multicast-tunneled to the mobile node within the domain. The multicast address of a mobile does not change throughout its movement within the domain. This allows for lighter-weight security during handover, as it is used for micro-mobility (i.e., intra-domain).

In this paper we present two different approaches to multicast-based micro mobility, one approach is based on *mobility proxies* and the other based on a novel scheme for *algorithmic mapping*. We compare such approaches and

show that algorithmic mapping provides a more scalable and robust approach, and we develop efficient, yet simple, mechanisms to realize it. Furthermore, we conduct extensive simulations to compare the handover performance of our approach to other routing-based micro-mobility schemes. The proactive handover performance results show that our scheme performs as well as CIP and much better than HAWAII. Furthermore, it handles multiple border routers in a domain where CIP fails. For reactive handover M&M has a clear edge over CIP.

The paper is outlined as follows. Section II introduces multicast-based mobility. Section III gives overview of the intra-domain architecture, and discusses the proxy-based approach. Section IV describes the algorithmic mapping approach in detail. Section V gives evaluation and comparison results. Section VI discusses related work. Section VII concludes.

2. Multicast-based Mobility (M&M)

Performance during handover is a significant factor in evaluating performance of wireless networks. IP-multicast [25][2] provides efficient location independent packet delivery. The receiver-initiated approach for IP-multicast enables receivers to join to a nearby branch of an already established multicast tree. Multicast-based mobility (M&M) [1][8] uses this concept to reduce latency and packet loss during handover.

In multicast-based mobility, each mobile node (MN) is assigned a multicast address. The MN, throughout its movement, joins this multicast address through locations it visits. Correspondent nodes (CN) wishing to send to the MN send their packets to its *multicast* address, instead of unicast. Because the movement will be to a geographical vicinity, it is highly likely that the join from the new location, to which the mobile recently moved, will traverse a small number of hops to reach the already-established multicast distribution tree. Hence, performance during handover improves considerably. An overview of this architecture is given in Figure 1. As the MN moves, it joins to the assigned multicast address through the new access router (AR). Once the MN starts receiving packets through the new location, it sends a prune message to the old AR to stop the flow of the packets down that path. Thus completing the smooth handover process. In spite of its promise, we believe that many issues need to be addressed to realize multicast-based mobility in today's Internet. These issues include scalability, multicast address allocation, multicast deployment and security.

Scalability of Multicast State: The state created in the routers en-route from the MN to the CN is source-group (*S, G*) state. With the growth in number of mobile nodes, and subsequently, number of groups (*G*), the number of states kept in the routers increases. In general, if there are '*x*' MNs, each communicating with '*y*' CNs on average, with an average path length of '*l*' hops, then number of states kept in the routers is '*x*y*l*' states. Clearly, this does not scale.

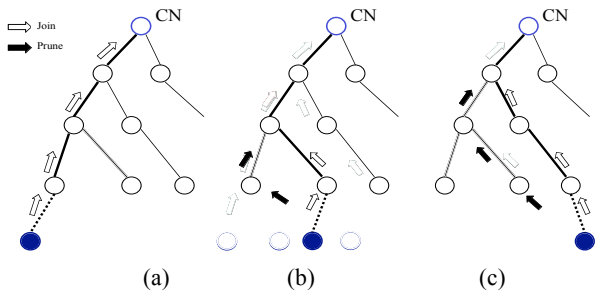


Figure 1: Multicast-based mobility. As the MN moves, as in (b) and (c), the MN joins the distribution tree through the new location and prunes through the old location.

Multicast Address Allocation: Inter-domain M&M requires each MN to be assigned a globally unique multicast address. Using a global multicast address for each MN may be wasteful and requiring uniqueness may not be practica1.

Ubiquitous Multicast Deployment: Inter-domain M&M assumes the existence of inter-domain multicast routing. We believe, however, that incremental deployment and interoperability should be an integral part of any architecture for IP mobility.

Security Overhead: Security is critical for mobility support, where continuous movement of mobiles is part of the normal operation. Such setting is prone to remote redirection attacks, where a malicious node redirects to itself packets that were originally destined to the mobile. The problem is even more complex with multicast, where any node may join the multicast address as per the IP-multicast host model. These security measures are complex and may incur a lot of overhead. If such measures are invoked with every handover, however, it may overshadow the benefits of efficient handover mechanisms[2].

To address the above issues, we propose a new approach for intra-domain multicast-based mobility.

3. Intra-domain Architectural Overview

In our intra-domain architecture, a mobile node is assigned a multicast address to which it joins while moving. The multicast address, however, is assigned only within a domain and is used for *micro* mobility. While moving between domains, an inter-domain mobility (e.g., Mobile IP) protocol is invoked. In Mobile IP (MIP) [3], every mobile

[1] Multicast address allocation is an active area of research [15]. We envision the number of MNs to grow tremendously.

[2] Providing a comprehensive security solution for IP mobility is beyond the scope of this work. We believe, however, that our schemes relaxes security requirements during handover.

node (MN) is assigned a home address and home agent (HA) in its home subnet. When the MN moves to a foreign subnet, it acquires a care-of-address (COA) through a foreign agent (FA). The MN informs the HA of its COA through a registration process. Packets destined to the MN's home address are intercepted by the HA in the home subnet, then it tunnels them to the MN's COA. This is known as triangle routing. We will use the Mobile IP model to discuss inter-domain routing in the following sections.

Several mechanistic building blocks are needed to realize our proposed architecture. First, when the mobile moves into a new domain it is assigned a multicast address. What is the address allocation scheme? Second, packets destined to the mobile are multicast-tunneled by an encapsulator to the mobile node. How are the encapsulator(s) selected and where are they placed? To answer these questions, we investigate two different approaches: (1) *Proxy*-based architecture, and (2) *Algorithmic mapping* architecture.

3.1 Reference Architecture

We consider an IP network for a single domain, as shown in Figure 2. The network is connected to the Internet through Border Routers (BRs). An Access Point (AP) is the radio point of contact for a mobile node. A number of APs are connected to an Access Router (AR). From the access router's point of view, each AP is a node on a separate subnet. When a mobile moves from one AP to another without changing AR is an intra-AR handover case that can be specific to AR implementation and is not considered in this paper.

When a mobile moves into a new domain it is assigned a multicast care of address (MCOA). It is also assigned a unicast address that is unique within the domain, called regional care of address (RCOA). Since MCOA is used for routing packets within the domain, there is no need to assign COA at every subnet. The RCOA is a unique unicast address on the *m-subnet*. The m-subnet is a unique subnet that is characterized by the mobility where mobile nodes can use their RCOA to establish communication through any AR at the edge of the network. Hence, the m-subnet can be viewed as a logical subnet formed by all APs at the edge of the network. All ARs include the prefix for m-subnet in their router advertisements [37].

Address allocation and management is discussed later in this paper. When a mobile moves from one AR to another, it is said to handover from old AR (AR_{old}) to new AR (AR_{new}). We use this terminology throughout the rest of the paper.

First, we shall describe the proxy-based approach and discuss the problems associated with it.

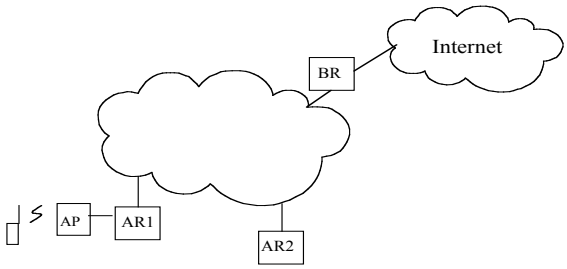


Figure 2: Reference mobility domain network

3.2 Proxy-based Architecture

When a mobile node moves into a new domain, it contacts its access router (AR). The AR performs the necessary per-domain authentication and security measures, and then assigns RCOA for the mobile node (MN). As shown in Figure 3, the AR then sends a *request* message to the mobility proxy (MP) to obtain a multicast address for the visiting MN. The request message includes the home address of the mobile node and its home agent's address. Upon receiving the request the MP performs two tasks. The first is to register on behalf of the mobile node its own address as COA with the MN's home agent. The second task is to assign a multicast address for the visiting MN, send a *reply* message to the AR and keep record of this mapping. The mapping is used for packet encapsulation later on. In this scheme, the MP remains transparent to the MN, which makes the placement of MPs within the domain flexible without notifying every MN.

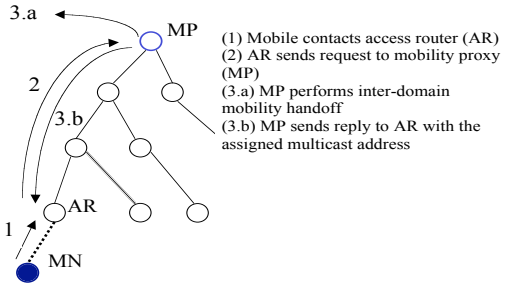


Figure 3: Event sequence as the mobile node moves into a domain

Once this step is complete, the visiting MN joins the assigned multicast address (*G*). The joins are sent to the proxy-group pair (*MP, G*) and are processed as per the underlying multicast routing. The MN continues to move within the domain using the same multicast address. The scope of the assigned multicast address is local to the domain. Handover is performed using standard join/prune mechanisms and only lightweight intra-domain security is required in this case.

Packets sent to the MN's home address are tunneled by the HA to the MP using inter-domain mobility. The packets

are then encapsulated by the MP, based on the recorded mapping, and sent down the multicast tree to the MN. The MN uses the unicast RCOA for sending packets. To avoid single-point-of-failure scenarios multiple MPs are used. These MPs are typically placed at the border of the domain or at the center of the network[3]. An algorithm similar to [24] may be used for dynamic MP liveness and election mechanisms.

Several issues need to be addressed in the above architecture. First, the MPs need to maintain unicast-to-multicast address mapping for all visiting MNs. The scalability of such a scheme is of question. Second, complex robustness algorithms are needed to maintain MP liveness information, requiring initial configuration and setup. Third, the service disruption effect of MP failure is not clear. Since the MP registers its own address with the home agent and is used to encapsulate incoming packets, this introduces a third-party-dependence problem that is undesirable. In addition, MPs should run a multicast address allocation scheme to ensure collision-free address assignment.

To address these problems we propose a novel approach based on *algorithmic mapping* that obviates the need for explicit unicast-to-multicast mapping, and eliminates the need for complex address allocation.

4. Algorithmic Mapping Architecture

This section provides detailed address management, duplicate address detection, and inter-AR handover.

4.1 Overview

In this scheme we assume there is a one-to-one mapping between an RCOA and MCOA. When a mobile moves into a new domain it is assigned RCOA by the AR and the mobile performs inter-domain handover i.e., it registers the RCOA with its home agent. The AR automatically infers the multicast address (MCOA) for the mobile node from the assigned unicast address (RCOA) through a straight forward *algorithmic mapping*, described later in this section. The AR then triggers a Join message for MCOA to establish the multicast tree. Packets destined to the MN's home address are tunneled to its RCOA by the HA. When these packets arrive in the foreign domain they are identified by the border router (BR) as being destined to a node on the m-subnet. As shown in Figure 4, the BR maps the destination unicast address to the multicast address and transmits the packets to the MN down the multicast tree. The serving AR changes the destination address from multicast to the unicast address. Since the destination address is modified twice within the network and restored to the RCOA by the AR, the packet does not cause security association violation at the mobile node.

[3] Network center are nodes with min(max distance) to any other node [26].

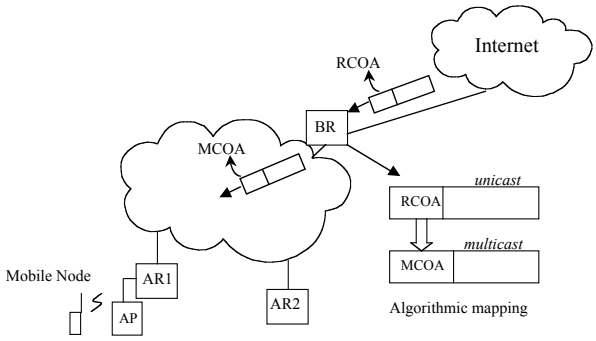


Figure 4: Architectural view: A packet is unicast to the RCOA and arrives at the border router (BR) for the mobile node. The BR intercepts the packet and performs algorithmic mapping from the RCOA to MCOA. The packet is then multicast within the domain.

This architecture provides several advantages over the proxy-based approach. It avoids the third party dependence on the MP. Moreover, since algorithmic mapping is used, no explicit RCOA-MCOA mapping is kept or maintained by the encapsulator, which solves the mapping scalability problem and provides a more robust mechanism.

4.2 Address Management

The number of multicast addresses required is proportional to the number of mobile nodes in the domain. The scope of an MCOA is local to the domain where it is used. The IPv6 multicast addressing provides facility to define scope within the address [32]. Hence, in the rest of the paper we consider IPv6 address for both RCOA and MCOA.

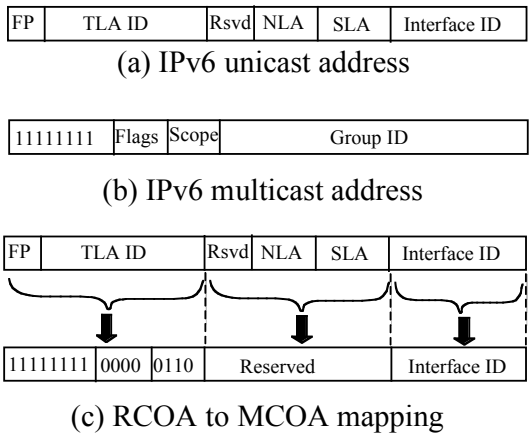


Figure 5: Algorithmic mapping

The standard IPv6 unicast and multicast address architectures [32] are shown in Figure 5 (a) and (b). We modify the group bits to include interface ID as the group ID. The remaining reserved bits of the group ID are ignored by multicast routing. The 64-bit interface ID address space is large enough for all the mobiles within a domain. We also

define a new scope: micro-mobility scope with value 0x6. The SLA is a 16-bit long field, used to create local hierarchy and identify subnets [33]. A single subnet ID, identifying m-subnet, is defined for assigning RCOA.

When a mobile moves into a foreign domain it is assigned an RCOA. The AR forms the MCOA by replacing the <FP, TLA ID> bits of the RCOA with the multicast <FP, flag (0000), scope (0110)> values. This provides a simple, yet very efficient and unique *algorithmic mapping*. The mobile acquires RCOA on the m-subnet through either DHCP [34] or autoconfiguration. The auto-configuration requires duplicate address detection (DAD) [35] on every subnet. In our scheme the mobile obtains RCOA and MCOA once it is connected to the network. We propose a scheme in [39] that detects address duplication within the m-subnet, which is performed once at the AR during initial address assignment. The mobile afterward is able to move freely without running DAD at any other AR. In any case, when a mobile first connects to the network it must performs a high latency inter-domain handover, hence duplicate address resolution latency is overshadowed by this handover latency.

4.3 Intra-domain Handover

When a mobile moves from one AR to another, a handover event takes place between the two routers. The handover involves route repair that is path setup inside the network to redirect the incoming traffic flow to the new AR. In proactive handover the link between the MN and new AR is established prior to its disconnection with the old AR. Hence a smooth handover, i.e. handover with low packet loss, can take place by exploiting the fact that the new AR is known a priori and bi-casting packets to both access routers is possible. In reactive handover an abrupt disconnection may cause the MN to switch over to the new AR. The route repair in this case can only be initiated from the new AR, hence bi-casting cannot reduce packet loss. Multicasting allows proactive path setup to the new access router before the mobile is actually connected to it. This can minimize packet losses in reactive handover where bi-casting fails. Moreover, bi-casting being a special case of multicasting, multicasting-based solution, e.g. M&M, performs equally well for achieving proactive handover. In this section we describe one handover scheme where proactive path setup is used to achieve smooth reactive handover.

We define a set of adjacent access routers as the Coverage Access Router Set (CAR-set). The adjacency can be established based on the adjacency of the radio coverage area of the serving AR in case of cellular wireless network. The serving AR is called the Head of the CAR-set. Thus, there is a unique CAR-set defined for every AR. For example, in Figure 6 AR1 to AR7 constitute a CAR-set for AR1, which is the serving AR for the mobile. The mobile can move to any of the ARs in the CAR-set without interruption in the packet flow. The idea of CAR-set is similar to Handoff-Affected Router Group (HARG) proposed in [41]. The HARG is a group of routers in the network that are affected by the handoff when a mobile node moves from one access point to another and need to do route repair. The fundamental difference with HARG is that the CAR-set is a set of access routers that are selected to receive the packets destined to the mobile node.

A site-local multicast group address is assigned to each CAR-set, called CAR-set group address (CGA). Every AR that is a member of a CAR-set must join the corresponding CGA, which serves as a control channel for the members to exchange the control signals. For example, in Figure 6, all the access routers surrounding AR1 join CGA1 to become members of AR1's CAR-set (CGA1). Similarly, AR1 must also join six other CAR-sets corresponding to adjacent routers AR2 to AR7.

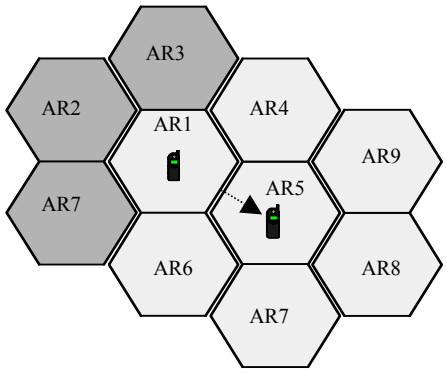


Figure 6: Handover across CARS

We define three new control signals as follows:

1. *J-message* causes the receiving router to *join* the multicast group identified in the message.

2. *L-message* causes the receiving router to *leave* the multicast group identified in the message.

3. *HO* message exchanged between the two routers involved in handover. Its parameters include the mobile's RCOA and MCOA.

We explain the handover algorithm by using the example depicted in Figure 6. Consider the MN moving from AR1 to AR5. When connectivity is established between the MN and AR5, the AR5 multicasts a J-message <MCOA> to the members of its CAR-set (CGA5) requesting them to join the mobile's MCOA. It then sends HO <RCOA, MCOA> message to AR1 to initiate the prune process. When AR1 receives HO message it multicasts an L-message <MCOA> to members of its CAR-set (CGA1) requesting them to leave the MCOA.

Although the ordering of (J => HO => L) messages ensures that L-message is initiated after J. The order of message reception, however, is not guaranteed to both CAR-sets. Depending on the order of arrival of J and L messages at an AR that is a member of both CAR-sets, it may leave the MCOA whereas it is supposed to have remained joined to

that group. To ensure consistency between Join and Leave messages we introduce the following mechanism. Each AR keeps its membership status in a 4-tuple <MCOA, Serving Access Router (SR), CGA, State> table. The table contains an entry corresponding to every mobile roaming in a CAR-set of which the access router is a member. There are two states defined: Joined and Left. The rules for updating the table specify that an AR only accept L-message for a MCOA, if the source of the L-message matches the SR in the MCOA's entry (i.e., the AR has joined the MCOA on the request of the same SR)[4]. Otherwise, the L-message is discarded. The AR accepts all J-messages and creates/updates the related MCOA entry to include the source of the J-message (as the SR), CGA to the SR's CGA (as the entry's CGA), and the state to Joined.

Consider the example shown in Figure 6. Assume that the mobile's MCOA is MG and after power up in the domain it connects to AR1, which then multicasts a J-message to its CAR-set (CGA1). When AR4 receives the J-message, it joins MG and creates an entry corresponding to the MCOA in Joined state as shown in Figure 7 (a). Later when the MN moves to AR5 it becomes the new serving router. Then AR5 sends a multicast J-message to its CAR-set (CGA5) followed by a HO message to the old serving router AR1. Since AR4 is a member of both CGA1 and CGA5, it receives both J-message from AR5 and L-message from AR1. After receiving the J-message the table entry is updated as shown in Figure 7 (b). If received after the J-message, the L-message is discarded. Thus, AR4 remains joined to MG. If received before the J-message, however, the L-message may cause AR4 to leave the MG, which interrupts packet flow to AR4 until it receives the J-message and joins the MG group. The interruption may be minimized by delaying the leave operation. In most cases the HO message delay is sufficient to minimize the interruption. A simple scheme can be employed that periodically checks the table to purge all the entries that are in the Left state and consequently prune the corresponding multicast trees.

MCOA	Serving Router	CGA	State
MG	*AR1*	*CGA1*	*Joined*

(a)

MCOA	Serving Router	CGA	State
MG	*AR5*	*CGA5*	*Joined*

(b)

Figure 7: Table state at AR4 (a) when MN1 is connected to AR1, (b) after MN1 moved to AR5

[4] To account for lost L-message, or crash of the SR, a soft-state mechanism is used. SR sends periodic J-messages containing table changes (if any) and providing liveness.

5. Evaluation and Comparison

In order to evaluate the performance of M&M and compare it to other known schemes, and conducted detailed simulations for CIP [20], Hawaii [21] and M&M – the three routing-based mobility solutions[5]. We modified the network simulator, ns-2 [17] to incorporate M&M. We changed the implementation of mobile node and access router to add mobility detection, handover algorithm and multicast routing.

5.1 Performance Metrics

We used the following performance metrics to evaluate the performance of M&M and compare it to CIP and HAWAII.

- *Handoff delay* is defined as the difference between the time at which the MN received the last packet from the old access router and the first packet from the new access router.
- *Depth of packet reordering* is measured as the maximum difference in the sequence numbers of adjacent packets. This is a rough indicator of the size of the buffer needed to re-sequence the out of order packets.
- *Packet duplication* is the total number of packets duplicated in a single handoff. This is measured as the duration for which reordering occurs. Since CBR traffic is used, reordering duration gives an estimate of how many packets can be duplicated irrespective of the packet rate at the source.
- *Routing efficiency* is defined as the ratio of the number of hops between the root of the tree and the MN to the number of hops on the shortest path between the two. This gives a qualitative comparison of routing efficiency.

Mobility detection need not necessarily be a part of the micro-mobility protocol as this can be better achieved with additional information from lower layers.

5.2 Simulation Scenarios

To study the factors affecting the performance of the micro-mobility protocols we simulated a rich set of scenarios including tree topologies of varying depth ranging from 3 to 6. The link bandwidths were fixed at 10Mbps for wired links with delays varied from 10ms to 5ms to 2ms for all links. Detailed 802.11 models in ns-2 were used for the wireless part with cell overlap of 30m. Beacons spacing 200ms apart are used for mobility detection. State timeout of 1s (as lower

[5] We have also compared our scheme to hierarchical MIP [27] and seamless handoff [31] schemes using route-based analysis. Please refer to [39] for details. As was shown in [39] M&M achieved the min handoff delay and min overhead among the three classes.

bound) is set for the multicast protocol. We have used CIMS extensions of ns-2 that implement CIP and HAWAII[6]. In addition, we developed our own extensions of ns-2 to support M&M. The handoff mechanism for M&M, CIP and HAWAII are bi-cast, semi-soft handoff and Multi Stream Forwarding (MSF) [21], respectively. Both M&M and CIP use bi-cast technique whereby packets are bi-cast to both old and new ARs from a crossover point within the network. In contrast, HAWAII uses buffer and forward technique where the old AR buffers the packets and forwards them during route repair. Random mobility at 30m/s was the mobility pattern used for the MN. CBR traffic with packet size of 512 bytes and 10ms/packet was used. To avoid the side effects of mechanisms of other protocols (like congestion control mechanism of TCP) affecting the handoff delay and packet delivery performance, we chose CBR over UDP as opposed to FTP over TCP.

5.3 Simulation Results

We conducted simulations over different topologies, varying parameters like beacon timer, and link delays. Since mobility detection mechanism is not a part of the protocol, simulations were set-up such that mobility detection always succeeded when the MN moved from one access router to another. This was to prevent loss of packets due to failure of the underlying mobility detection scheme.

Graphs for different topologies show the similar trends; hence we select simple graphs for the tree topology with depth 3. Figure 8 shows the topology used in the simulation.

All the graphs follow a common format. Each graph shows data for M&M, CIP and HAWAII (in that order from left to right). The x-axis shows three sets of data corresponding to link delays of 10ms, 5ms and 2ms (again from left to right) for each protocol. Path lengths from the fork (crossover) router to old and new access routers vary along y-axis. For example, '3,2' means path length of 3 hops from the fork router to the old access routers and 2 hops from the fork router to the new access router. The z-axis shows the performance parameters under evaluation.

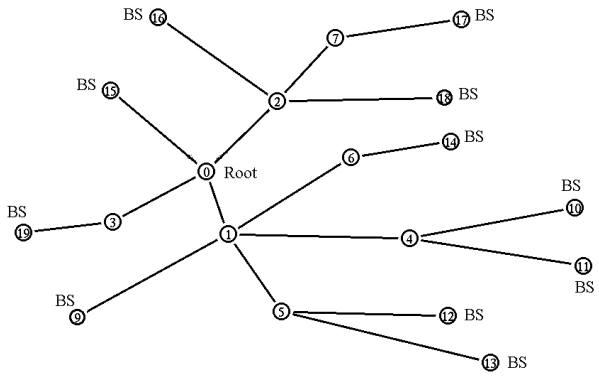


Figure 8: Simple tree topology

Figure 9 illustrates the handoff delays incurred by M&M, CIP and HAWAII with link delays 10, 5 and 2ms. From the graphs, we observe that the handoff delay for M&M and CIP is small as compared to that of HAWAII. Both CIP and M&M use bi-cast, which causes smooth handover with negligible handover delay. Whereas, HAWAII using the MSF, a buffer and forward scheme that consistently incurs long handoff delays.

Figure 10 shows the depth of reordered packets. We measured depth of reordering instead of the number of packets reordered because it indicates the size of buffer needed to re-sequence the out of order packets. It is obvious from the graph that the depth of reordering is small for M&M and CIP, whereas it is large for HAWAII. The out of sequence packets in M&M and CIP are dependent on the difference in the link delays from fork router to old and new access routers. The greater the difference, the greater will be the depth of reordering. In case of HAWAII the depth is large because the old access router buffers packets and then forwards it to the new access router via the crossover router. The crossover router also forwards the incoming packets to the new access router at the same time. This results in packets reaching the new AR out of order. Depth of reordering is dependent on the buffering duration and the link delays from the cross over router to the old AR. It is important to observe the duration for which reordering of packets occur. In M&M and CIP, reordering occurs as long as bi-casting is done. However, in HAWAII, reordering duration depends on the number of packets buffered at the old AR and the link delay from the old AR to the crossover point.

[6] We used the CIMS (Columbia IP Micro-mobility Suite) at http://comet.ctr.columbia.edu/micromobility/software.htm

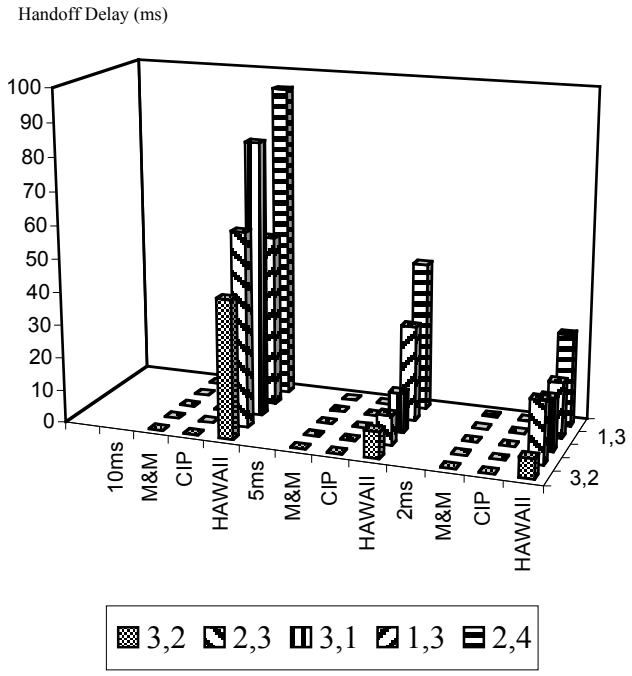


Figure 9: Handoff delay

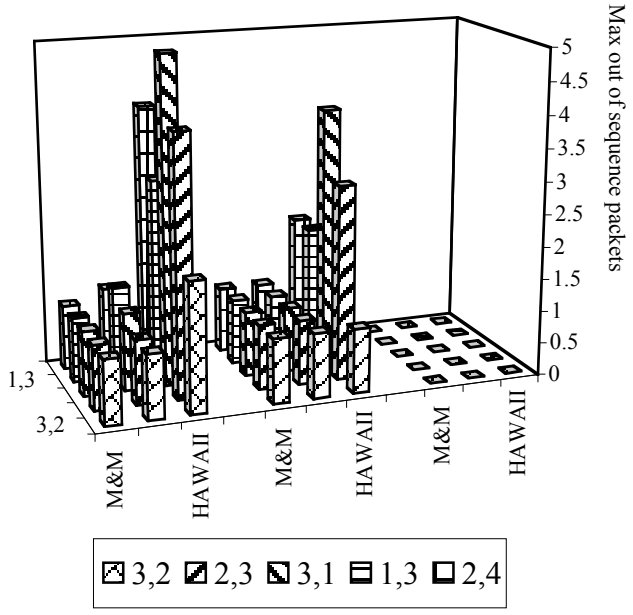


Figure 10: Maximum difference in sequence numbers of consecutive packets

The duration for which reordering of packets occurs indicates an estimate of the amount of packet duplication caused by a scheme. Figure 11 shows the reordering duration incurred by the three schemes. As previously mentioned, in case of M&M and CIP, the reordering occurs as long as bi-casting lasts causing large number of packet duplication as shown in the figure. Whereas, for HAWAII reordering duration depends on the number of packets buffered at the old access router and the link delay from the old access router to the crossover point, which shows relatively low number of duplications.

M&M uses the multicast path to route packets to the MN. In many cases the border router (BR) acts as the root (RP) of the multicast tree. CIP uses the shortest path along the reverse path from the MN to the BR to route packets from the BR to the MN. In most cases the routing in M&M is as efficient as CIP. In case of HAWAII routing is a function of topology and node mobility, which is generally less efficient than that of M&M and CIP.

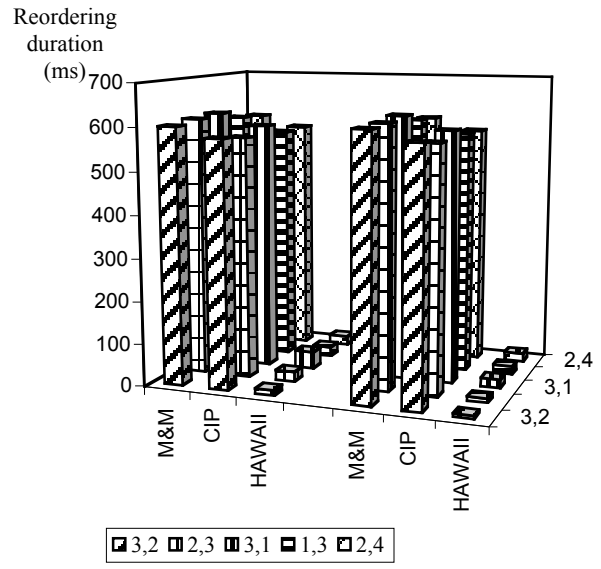


Figure 11: Reordering duration

Both HAWAII and CIP do not handle well the case where a domain contains multiple border routers. In particular, if packets enter the domain through one border router and leave through another border router, routing in CIP fails. M&M relies on the underlying multicast protocol to handle multiple border routers in a domain, which is often the case. For example, mechanisms exist in PIM-SM to deliver packets to the RP irrespective of the location of the sender (BR at which the packet enters the domain). The flexibility comes at the expense of possible reduction of routing efficiency, because packets are first tunneled to the RP and then delivered to the MN through the multicast tree. To alleviate this situation the BRs may be configured as candidate RP for the MCOA prefix, thus ensuring that one of the BRs becomes the RP.

5.4 Re-active Handover

M&M has a clear edge over any other unicast based micro-mobility protocol (e.g., Cellular IP and Hawaii) during reactive handover. Reactive handover occurs when a mobile node moves out of coverage, due to obstacles, lack of cell overlap, etc., then re-enters the coverage of a new cell. Some wireless technologies, e.g. IEEE 802.11, only support reactive handover. In such scenarios (that are not un-common), bi-casting (i.e., getting packets from both the old and new base stations) as used in CIP, is not possible. For bi-casting to occur the mobile needs to be connected to both base stations simultaneously. Prediction may be used to sent packets to potential future base stations, but bi-casting can only send to one new base station (extending bi-casting to send packets to multiple base stations is basically re-inventing multicasting). M&M, on the other hand, by virtue of being a multicast protocol, is able to send to multiple (2 or more) base stations. The CAR-set protocol presented in this paper pro-actively sends packets to all potential future base stations thus reducing delays and packet losses during re-active handover drastically. Figure 12 shows the number of packet losses incurred during reactive handover. Two scenarios were used, the first has 0m overlap between cells, and the second has 10m gap between cells. Hawaii and M&M incur very little or no packet loss, whereas CIP incurs significant packet loss. Hawaii's reduced packet loss is due to the buffering of packets (which comes at the expense of extended handover delays). Handover delay (from the point when the mobile node detects the new access router) is similar to that given in **Figure 9** above. It is quite clear that M&M has the best performance in terms of both packet loss (clearly outperforming CIP) and handover delays (clearly outperforming Hawaii) during reactive handover.

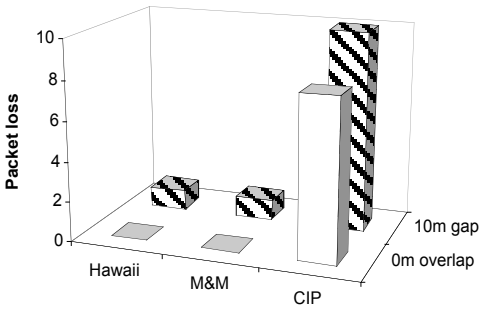


Figure 12: Packet loss for reactive handover

5.5 CAR-set Multicast Overhead

Multicasting packets to the CAR-set causes overhead of packet replication over links leading to the access routers that belong to the CAR-set. The extent of overhead depends upon the network topology and the size of the CAR-set. For a given MN, let the path from the RP to the MN contains L links on average. This is the path from the BR to the serving AR to the MN. Also, let n be the number of ARs other than the serving AR in the CAR-set. Hence, the CAR-set is $\{AR0, AR1, \ldots, ARn\}$, where $AR0$ is the serving AR. Furthermore, let Li be the number of links leading from ARi to the nearest point already branch from the RP to $AR0$, where i=1,2,....,n. If we measure the overhead by the number of additional links traversed by the replicated packets, then the overhead is $\sum Li$, called $Lsum$. The ratio $Lsum/L$ gives the measure of additional links carrying replication traffic due to packet replication for a given MN connected to $AR0$. The upper bound for the total replication traffic on the additional links for $AR0$ is m, where m is the number of MNs connected to $AR0$.

The replication traffic on a link consumes link bandwidth proportional to $k.b$, where k is the number of ARs for which the link is an additional link carrying replication traffic and b is the wireless bandwidth for each AR. Typically wireless bandwidth is much smaller than the bandwidth of the wired links and it constrains the traffic (including the replication traffic) over the wired links.

We adopt a two-dimensional approach to reduce the replication traffic by limiting the size of the CAR-set (space-dimension) and the duration (time-dimension) for which the replication is performed in the network.

In this paper we presented a simple static CAR-set algorithm, however a more dynamic algorithms can be designed by identifying the highly probable new ARs. We are exploring this area further. For reducing the duration of replication traffic we are working on a heuristic that can potentially reduce the overhead significantly, as follows. When an access router (old AR) detects that the signal as seen by the MN is fading and is an indicative of handover condition, it then triggers the ARs in the CAR-set to join the multicast group. To avoid packet losses, the handover condition must be detected early enough to provide time margin before actual handover required for multicast join to happen. Once the MN is connected to the new AR, the CAR-set members leave the group. Thus, the overhead due to the replication traffic is reduced to only the fraction of the time during which the CAR-set remains joined to the multicast group, that is, only as long as the handover condition exists.

6. Related Work

Several architectures have been proposed to provide IP mobility support. In Mobile IP (MIP) [3], every mobile node (MN) is assigned a home address and home agent (HA) in its home subnet. When the MN moves to another foreign subnet, it acquires a care-of-address (COA) through a foreign agent (FA). The MN informs the HA of its COA through a registration process. Packets destined to the MN are sent first to the HA, then are tunneled to the MN. This is known as

triangle routing, a major drawback of MIP. Route optimization in [4] attempts to avoid triangle routing by sending binding updates, containing the current COA of the MN to the correspondent node (CN). However, communication overhead during handover renders this scheme unsuitable for micro mobility.

In [16] end-to-end IP mobility is proposed, based on dynamic DNS updates. When MN moves, it obtains a new IP-address and updates the DNS mapping for its host name. This incurs handover latency due to DNS update delays and is not suitable for delay-bounded applications. Also, the scheme is not transparent to transport protocols.

In [10] the HA tunnels packets using a pre-arranged multicast group address. The access router, to which the MN is currently connected, joins the group to get data packets over the multicast tree. This approach suffers from the triangle routing problem; packets are sent to HA first and then to MN. Multicast-based mobility is proposed in [1] and [8]. Each MN is assigned only a unique multicast address. Packets sent to the MN are destined to that multicast address and flow down the multicast distribution tree to the MN. The CN tunnels the packets using the multicast address. This approach avoids triangle routing, in addition to reducing handover latency and packet loss. The study in [1] quantifies the superiority of handover performance for multicast-based mobility over Mobile IP protocols. These schemes, however, suffer from several serious practical issues, including scalability of multicast state, address allocation and dependency on inter-domain multicast. We address these issues in our work.

Several approaches have been proposed for micro mobility [18]. The general approaches include mobile-specific routing, hierarchical approaches and seamless handover. Mobile-specific route approaches include cellular IP [20] and Hawaii [21]. A domain-gateway registers its address with the HA (this has similarities to our proxy-based approach) and forwards the packets to the MN. The MN's home address is used within the domain. These approaches need special signaling to update mobile-specific routes and require changes in packet forwarding and unicast routing in all the routers. In cellular IP [20], signaling is data-triggered to create paths by having routers snoop on the data packets. Hawaii [21] proposes a separate routing protocol and requires explicit signaling from the mobiles. In a way, these approaches attempt to create a distribution tree using extra routing entries for the mobile, similar to multicast. Our approach builds upon existing multicast mechanisms as opposed to re-creating them.

Approaches based on seamless handover between old and new access routers, involve fairly complex signaling, buffering and synchronization procedures. Router-assisted smooth handoff in MIP [5], and edge mobility [22] belong to this category. Fast Handover in [31] introduces fast tunnel set-up between AR_{old} and AR_{new} as soon as the layer 2 handoff is detected. The tunnel avoids packet losses caused by path set-up delay inside the mobility domain. In a way it is complementary to our multicast-based routing inside the mobility domain. Unlike fast handover, however, our m-subnet idea considers the edge of the network as a single subnet and allows mobile node to carry RCOA and MCOA across ARs, which reduces the handover latency. Approaches using a hierarchy employ a gateway per-domain and need to keep a location database to map identifiers into locations. This mapping suffers from scalability and robustness problems as was noted earlier in this paper. In [12] a hierarchy of foreign agents is created at the local, administrative domain and global levels. In [19] a multi-level hierarchy is used in which packets from the HA arrive at a root FA where they are tunneled to a lower level FA and then to the MN. Hierarchical MIP [27] builds a network of tunnels (overlay network) between FAs. Work in [23] and [29] also uses a notion of mobility agent for localized handoff within a domain. We have shown in [39] that our multicast-based intra-domain mobility scheme outperforms seamless handover and hierarchical approaches and is simpler. This result is consistent with the comparison of routing-based (HAWAII and CIP) and tunneling-based (Hierarchical Mobile IP) schemes reported in [40]. It is shown that Hierarchical Mobile IP performs either equally well or inferior to the routing-based schemes, because it does not take advantage of the proximity of crossover router to the serving AR. In addition, our comparison results for CIP and HAWAII are generally consistent with the above study. However, we have used more complex topologies, scenarios of reactive handoff, and we investigated performance in a more detailed manner; instead of looking at averages we looked at specific metrics as function of the hop distance from the old AR to the fork router and the new AR to the fork router. Our results indicate that M&M performance during proactive handover matches that of CIP and is better than Hawaii. For reactive handover we show that M&M clearly outperforms CIP and Hawaii.

7. Concluding Remarks

We have presented a novel approach to IP micro-mobility using intra-domain multicast-based mobility. Our approach solves major challenging problems facing the deployment of multicast-based mobility. In terms of multicast state scalability we note that the multicast state growth is $O(G)$ for the architecture presented in this study, as opposed to $O(S \times G)$ in [1][8]. Our novel algorithmic mapping scheme from unicast to multicast address ensures collision-free assignment by providing unique and consistent mapping throughout the network. This solves the address allocation problem and provides robustness and per-domain privacy as multicast packets are not forwarded out of the domain. In addition, we present a new proactive path setup scheme to improve handover performance. Our extensive simulations show that:

- There is a significant difference in handoff delay and packet reordering performance between protocols using different types of handoff schemes. For example, M&M and CIP use bi-cast while HAWAII use buffer and forwarding.
- In most cases M&M and CIP show comparable routing efficiency and handoff performance because both use shortest path routing as opposed to HAWAII. Routing packets on the path that is not the shortest path from the root of the tree to the MN not only increases end-to-end delay, but also wastes bandwidth and creates extra mobile specific routing entries.
- Bi casting:
    - Masks handoff delays (virtually zero delay)
    - Produces duplicate packets
    - Shows small reordering depth depending on the difference in the path lengths from the fork router to the old and new access routers
- Buffering and forwarding
    - Incurs longer handoff delays
    - May produce large reordering depth

For proactive handover M&M performs as well as CIP, and it handles the case of multiple BR in a domain better than others. The M&M scheme clearly outperforms CIP and HAWAII in reactive handover because of its proactive (CAR-set) path setup capability. It uses multicast routing protocol, e.g. PIM-SM, which is more reliable with readily available robust implementation and people having more experienced managing it. All these factors facilitate the deployment of M&M per ISP domain. Furthermore, M&M naturally supports efficient multicasting to MNs.

In the future, we plan to conduct further simulations to evaluate a richer set of reactive handover scenarios. In addition, we plan to further pursue our approaches to reduce overhead of the CAR-set algorithm and conduct a detailed study of such overhead. We also would like to investigate M&M's support for efficient mobile-to-mobile communication.

8. References

[1] A. Helmy, "A Multicast-based Protocol for IP Mobility Support", ACM SIGCOMM Second International Workshop on Networked Group Communication (NGC 2000), Palo Alto, November 2000.

[2] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, V. Jacobson, M. Handley, C. Liu, P. Sharma, "Protocol Independent Multicast – Sparse Mode (PIM-SM): Protocol Specification", *RFC 2362/2117 of the IETF*, March '97/'98.

[3] C. Perkins, "IP Mobility Support", *RFC 2002, Internet Engineering Task Force*, October 1996.

[4] C. Perkins, D. Johnson, "Route Optimization in Mobile IP", *Internet Draft, Internet Engineering Task Force,* February 2000.

[5] C. Perkins and D. Johnson, "Mobility Support in IPv6", *Proceedings of MobiCom'96*, November 1996.

[6] D. Johnson, C. Perkins, "Mobility Support in IPv6", *Internet Draft, Internet Engineering Task Force,* March 2000.

[7] A. Myles, D. Johnson, C. Perkins, "A Mobile Host Protocol Supporting Route Optimization and Authentication", *IEEE Journal on Selected Areas in Communications,* vol. 13, No. 5, p. 839-849, June 1995.

[8] J. Mysore, V. Bharghavan, "A New Multicasting-based Architecture for Internet Host Mobility", *Proceedings of ACM MobiCom*, September 1997.

[9] J. Mysore, V. Bharghavan, "Performance of Transport Protocols over a Multicasting-based Architecture for Internet Host Mobility", *International Conference on Communications (ICC)'98,* vol. 3, p. 1817-1823, 1998.

[10] S. Seshan, H. Balakrishnan, R. Katz, "Handovers in Cellular Wireless Networks: The Daedalus Implementation and Experience", *Kluwer Journal on Wireless Networks,* 1995.

[11] H. Balakrishnan, S. Seshan, R. Katz, "Improving Reliable Transport and Handover Performance in Cellular Wireless Networks", *Proceedings of ACM MobiCom'95,* November 1995.

[12] R. Caceres, V. Padmanabhan, "Fast and Scalable Handovers for Wireless Internetworks", *Proceedings of ACM MobiCom'96,* November 1996.

[13] R. Caceres, V. Padmanabhan, "Fast and Scalable Wireless Handovers in Support of Mobile Internet Audio", *ACM Journal on Mobile Networks and Applications,* vol. 3, No. 4, December 1998.

[14] A. Acampora, M. Naghshineh, "An Architecture and Methodology for Mobile-Executed Handover in Cellular ATM", *IEEE Journal on Selected Areas in Communications*, vol. 12, No. 8, p. 1365-1375, October 1994.

[15] S. Kumar, P. Radoslavov, D. Thaler, C. Alaettinoglu, D. Estrin, M. Handley, "The MASC/BGMP Architecture for Inter-domain Multicast Routing", *Proceedings of ACM SIGCOMM,* August 1998.

[16] A. Snoeren, H. Balakrishnan, "An End-to-End Approach to Host Mobility", *Accepted to ACM MobiCom '00,* August 2000.

[17] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, H. Yu, "Advances in Network Simulation", *IEEE Computer,* vol. 33, No. 5, p. 59-67, May 2000.

[18] A. T. Campbell and J. Gomez IP Micro-Mobility Protocols, ACM SIGMOBILE Mobile Computer and Communication Review (MC2R), 2001

[19] E. Gustafsson, A. Jonsson, C. Perkins, Mobile IP Regional Registration, Internet-draft, draft-ietf-mobileip-reg-tunnel-02, March 2000.

[20] A. Campbell, J. Gomez, S. Kim, A. Valko, C. Wan, Z. Turanyi, "Design, implementation, and evaluation of cellular

IP" IEEE Personal Communications , Volume: 7 Issue: 4 , Page(s): 42 –49, Aug. 2000.

[21] R. Ramjee, T. La Porta, L. Salgarelli, S. Thuel, K. Varadhan, L. Li, "IP-based access network infrastructure for next-generation wireless data networks", IEEE Personal Communications, Volume: 7 Issue: 4, Page(s): 34 –41, Aug. 2000.

[22] A. O'Neill, G. Tsirtsis, S. Corson, Edge Mobility Architecture, Internet-draft, draft-oneill-ema-02.txt, July 2000.

[23] J. Kempf, P. Calhoun, C. Pairla, Foreign Agent Assisted Handover, Internet-draft, draft-calhoun-mobileip-proactive-fa-01.txt, June 2000.

[24] D. Estrin, M. Handley, A. Helmy, P. Huang, D. Thaler, "A Dynamic Bootstrap Mechanism for Rendezvous-based Multicast Routing", Proceedings of IEEE INFOCOM '99, New York, March 1999.

[25] S. Deering, D. Cheriton, "Multicast routing in data internetworks and extended lans", ACM Transactions on Computer Systems, pp 85-111, May 1990.

[26] E. Fleury, Yih Huang, P.K. McKinley. "On the performance and feasibility of multicast core selection heuristics", 7th International Conference on Computer Communications and Networks, Page(s): 296 –303, 1998.

[27] H. Soliman, C. Castelluccia, K. Elmalki, L. Bellier, "Hierarchical Mobile IPv6", Internet Draft, draft-ietf-mobileip-hmipv6-04.txt, July 2001.

[28] A. Misra, S. Das, A. Mcauley, A. Dutta, S. K. Das, "IDMP: An Intra-Domain Mobility Management Protocol using Mobility Agents", I-Draft, draft-misra-mobileip-idmp-00.txt, Jan 2000.

[29] G. Dommety, "Local and Indirect Registration for Anchoring Handoffs", I-Draft, draft-dommety-mobileip-anchor-handoff-01.txt, December 2000.

[30] K. Calvert, M. Doar, E. Zegura, "Modeling Internet Topology", *IEEE Comm,* p. 160-163, June 1997.

[31] G. Dommety, et al, "Fast Handovers for Mobile IPv6", Internet Draft, draft-ietf-mobileip-fast-mipv6-02.txt, July 2001.

[32] R. Hinden, S. Deering, "IP Version 6 Addressing Architecture", Internet Draft, draft-ietf-ipngwg-addr-arch-v3-06.txt, July 2001.

[33] R. Hinden, M. O'Dell, S. Deering, "An IPv6 Aggregatable Global Unicast Address Format", RFC 2374, July 1998.

[34] J. Bound, et al, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", Internet Draft, draft-ietf-dhc-dhcpv6-19.txt, June 2001.

[35] S. Thomson, T. Narten, "IPv6 Stateless Address Autoconfiguration", RFC 2462, December 1998.

[36] 3GPP, "3GPP TS Group Core Network; Numbering, addressing, and identification", 3GPP TS 23.003 V5.0.0 (2001-06), www.3gpp.org, June 2001.

[37] T. Narten, E. Normark, W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, December 1998.

[38] A. Helmy, "State Analysis and Aggregation Study for Multicast-based Micro Mobility", IEEE International Conference on Communications (ICC), May 2002.

[39] A. Helmy, and M. Jaseemuddin, "Efficient Micro-Mobility using Intra-domain Multicast-based Mechanisms (M&M)", *USC-CS-TR-01-747*, Aug 2001.

[40] A. Campbell, J. Gomez, S. Kim, Z. Turanyi, C-Y Wan, A. Valko, "Comparison of IP Micromobility Protocols", IEEE Wireless Communications, Vol. 9, No. 1, pp. 72-82, February 2002.

[41] Hongyi Li, "A Micro-Mobility Routing Protocol for Wireless Internet", IP-Based Cellular Networks (IPCN) 2001, May 17, 2001, Paris, France.

Hop-by-Hop Routing Algorithms For Premium Traffic *

Jun Wang†
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801, U.S.A.
junwang3@cs.uiuc.edu

Klara Nahrstedt
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801, U.S.A.
klara@cs.uiuc.edu

ABSTRACT

In Differentiated Service (DiffServ) networks, the routing algorithms used by the *premium* class traffic, due to the high priority afforded to that traffic, may have a significant impact not only on the premium class traffic itself, but on all other classes of traffic as well. The shortest hop-count routing scheme, used in current Internet, turns out to be no longer sufficient in DiffServ networks. This paper studies the problem of finding optimal routes for the premium-class traffic in a DiffServ domain, such that (1) no forwarding loop exists in the entire network in the context of hop-by-hop routing; and (2) the residual bandwidth on bottleneck links is maximized. This problem is called the Optimal Premium-class Routing (OPR) problem. We prove in this paper that the OPR problem is NP-hard.

To handle the OPR problem, first, we analyze the strength and weaknesses of two existing algorithms (Widest-Shortest-Path algorithm and Bandwidth-inversion Shortest-Path algorithm). Second, we propose a novel heuristic algorithm, called the *Enhanced Bandwidth-inversion Shortest-Path (EBSP) algorithm*. We prove theoretically the correctness of the EBSP algorithm, i.e., we show that it is *consistent* and loop-free. Our extensive simulations in different network environments show clearly that the EBSP algorithm performs better when routing the premium traffic in complex, heterogeneous networks.

Keywords

Hop-by-hop Routing, Differentiated Service, Premium Class, Saturate Bandwidth.

1. INTRODUCTION

Exploding traffic and expanding Quality-of-Service (QoS) requirements, coming from emerging multimedia applications, initiated intensive research in QoS provision and routing within the Internet. In this paper, we raise an interesting problem of how to find optimal routing schemes under both Differentiated Service (DiffServ) and hop-by-hop IP routing assumptions. By combining service differentiation and QoS routing together, the high-priority premium traffic should be transmitted in an efficient manner with low negative influences to other low-priority traffic. Before presenting our work in detail, some background knowledge is introduced in the following subsections.

1.1 Hop-by-Hop Shortest-path Routing

Hop-by-hop routing forms the basis of today's IP networks. Hop-by-hop routing means that routing decisions are made at each router independently and locally. For each incoming packet at a router, its destination address (maybe some other fields in its IP header, too) is used to get the next hop by consulting the router's routing table. Therefore, hop-by-hop routing is also referred to as destination-based table-driven routing. The hop-count shortest-path routing (SP) is the most commonly used method for IP routing in today's Internet. Algorithms of finding the shortest-paths between nodes, such as the Dijkstra's algorithm, can guarantee that no forwarding loop exists in a network. Figure 1 shows how the SP method works in a hop-by-hop scenario.

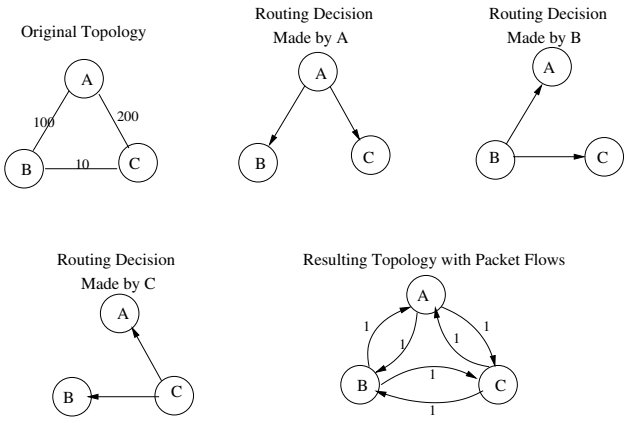


Figure 1: An example of hop-by-hop SP routing: the numbers next to the links in the original topology denote link capacities, while the labels in the resulting topology mean how many "flows" go through the links.

The resulting topology, shown in Figure 1, illustrates the real packet flows[1] between each pair of nodes. The num-

*This work was supported by NSF Grant under contract number NSF ANI 00-73802 and NSF CISE Grant under contract number NSF EIA 99-72884. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

†Please address all correspondences to Jun Wang and Klara Nahrstedt at Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, phone: (217) 244-5841, fax: (217) 244-6869.

[1]The word "flow" here is different from the definitions in

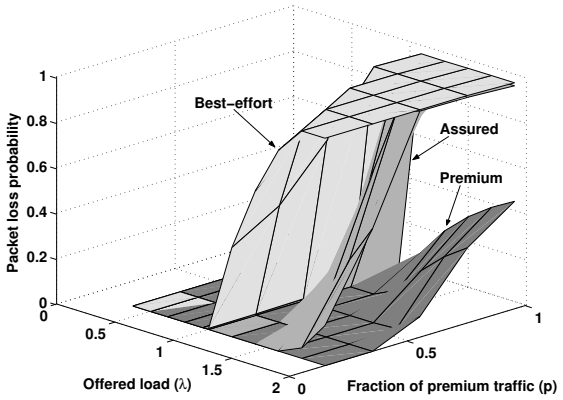
bers associated with the arcs are the numbers of flows going through that particular link in that direction.

In this paper, we investigate a QoS routing problem in the context of hop-by-hop routing, where bandwidth, rather than hop-count, is sensitive and the SP method turns out to be insufficient.

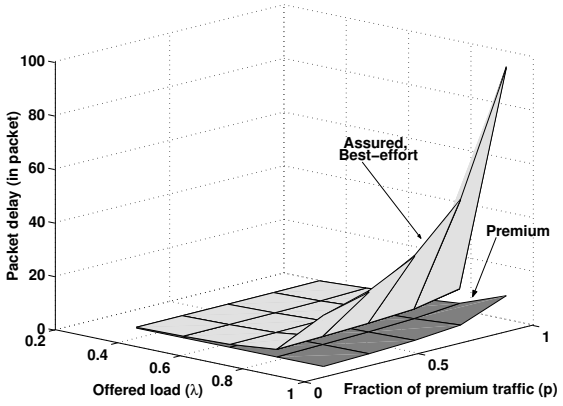
1.2 DiffServ Model and Inter-class Effects

In order to provision better end-to-end QoS, DiffServ scheme has been proposed as a cost-effective solution [6, 11]. In DiffServ networks, traffic is classified into three service classes: premium, assured and best-effort. The premium class traffic has the highest priority in comparison to other classes of traffic. Originally, the DiffServ scheme is decoupled from IP routing intentionally, meaning that all traffic between each source-destination pair follows the same path no matter which service class it belongs to and DiffServ itself has no effect on IP routing decisions. However, DiffServ does affect the queueing and drop behavior in routers. More specifically, two separate queues are used in DiffServ. One of them is used by premium traffic and the other is shared by non-premium traffic. The premium queue has absolutely higher priority than the other queue, therefore, as long as there are packets in the premium queue, these packets will be scheduled first. Due to premium traffic's high priority, a bad routing decision could lead to some problems for the low-priority traffic when the volume of premium class traffic is high. This means, without taking routing into consideration, the premium class traffic imposes very negative influences on other classes of traffic, especially when the network is highly loaded. We call this the inter-class effects [27]. In [21], the authors presented simple performance models and analysis of DiffServ schemes. However, to make a strong case of the negative impacts that the premium class traffic may impose on all other service classes in DiffServ networks, we have run simulations to measure the inter-class effects among *all* three service classes. Our results are shown in Figure 2.[2]

As we can see in Figure 2, the premium class traffic has significant inter-class effects on the assured class and best-effort class traffic with respect to some important metrics, such as the *packet loss probability* (Figure 2(a)) and the *packet delay* (Figure 2(b)). When the network is highly loaded (large offered load λ) or the fraction of premium traffic is high (large p), the traffic with low-priorities experiences severe performance degradations (such as higher packet loss rates and larger queueing delays). Therefore, we must take the inter-class effects into consideration when we choose routing algorithms for networks which support premium class traffic.



(a) Packet Loss Probability



(b) Packet Delay (in packet)

Figure 2: Measurements of the inter-class effects between all three classes in a DiffServ network

1.3 Routing and Bandwidth Reservation for Premium Traffic

During a certain time interval, presuming the full knowledge of the network topology, we can regard the topology metrics as static. Hence, we concentrate only on algorithmic aspects of static QoS routing schemes for the premium-class traffic in a DiffServ network.

In order to provide end-to-end QoS guarantees for the premium-class traffic between two nodes, certain amount of bandwidth must be successfully reserved on each link along the path between these two nodes. (How to implement such bandwidth reservation is out of the scope of this paper. We assume that on top of our routing scheme, some class-based resource reservation protocol, similar to the RSVP, is used for premium bandwidth reservation in the network.) If a link is shared by multiple paths between different pairs of nodes, the bandwidth it has to reserve should accommodate the summation of the premium traffic on all the paths sharing this link.

Integrated Service (IntServ) model or in RSVP. A "flow" here refers to a channel between two nodes, through which all premium traffic between the two nodes will follow. So each pair of source and destination in the network identifies a "flow". For instance, Flow $\overrightarrow{AB}$ indicates the channel from A to B such that all premium packets from A to B are transmitted through $\overrightarrow{AB}$.

[2]More simulation details and extensive results and measurements are presented in our technical report [27].

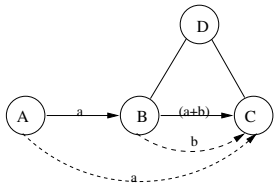


Figure 3: An example of bandwidth reservations for premium traffic: since link (B, C) is shared by flow $\vec{AC}$ and $\vec{BC}$, it must reserve $(a + b)$ amount of bandwidth.

Figure 3 depicts a scenario in which node A requires a amount of bandwidth to C while B requires b amount of bandwidth to C. Since the path $(A \rightarrow B \rightarrow C)$ is used for the flow $\overrightarrow{AC}$ according to SP routing, the link (B, C) is shared by flows $\overrightarrow{AC}$ and $\overrightarrow{BC}$, so it must reserve totally $(a + b)$ amount of bandwidth for both flows. As one can expect, with the variance of link capacities, the success of a reservation depends not only on how much bandwidth it tries to reserve and the capacity (more precisely, residual bandwidth) of each link along its path, but on the routing strategy as well.

In this paper, for both simplicity and expediency of problem definition, we assume each node in the topology tries to reserve same amount of bandwidth, say B, to all other nodes for its premium traffic. As we can see, given a network, for each routing scheme $\mathcal{R}$, there exists a maximum value of B, called the *saturated bandwidth* for $\mathcal{R}$ and denoted as $B_s(\mathcal{R})$, which saturates some link in the network. The saturated link is called the bottleneck link. Notice that the *saturate bandwidth* B_s is a virtual metric which is used only for finding better paths and for evaluating performance of different algorithms. (The larger B_s an algorithm can achieve, the better algorithm it will be.) In reality, we do not have to reserve that large amount of bandwidth for the premium traffic to always saturate some link(s) in the network. We can understand this from two different perspectives. Firstly, The routing scheme with maximum B_s is able to accommodate maximum amount of premium traffic in the network, thus maximizing the potential for future premium traffic growth. Secondly, in a real network, premium traffic may only reserve small portion of B_s bandwidth. Therefore, a routing scheme with larger B_s will leave more residual bandwidth to the best-effort traffic on those stringent links, thus reducing inter-class effects and bringing the whole network into a more load-balanced mode (for example, the chance of bandwidth starvation for low-priority traffic may be reduced). In fact, under the homogeneous bandwidth demand assumption, it is not difficult to verify that the optimality of the maximum saturate bandwidth is equivalent to the optimality of the minimum *relative congestion* among all links [9, 13, 15] (i.e., the ratio of the total amount of reserved bandwidth for the premium traffic, divided by the link capacity). As we can see from Figure 2, minimizing relative congestion of the premium traffic can minimize the negative inter-class effects on the low-priority traffic.

Therefore, an interesting problem is how to achieve the optimal value of B_s (denoted as B_{max}) by choosing optimal paths between nodes. More specifically,

$$B_{max} = max\{B_s(\mathcal{R}_1), B_s(\mathcal{R}_2), \cdots, B_s(\mathcal{R}_n)\},$$

where $\mathcal{R}_1$, $\mathcal{R}_2$, $\cdots$, $\mathcal{R}_n$ are all possible loop-free routing solutions for the given network. To find the optimal routing solution and the value of B_{max} is called the Optimal Premium-class Routing (OPR) problem and it is not a trivial task. An optimal local solution for a single source may not be the solution to the whole OPR problem. We will prove in Section 3 that this problem indeed is NP-hard. Since the SP routing algorithm is polynomial, it can not be optimal. Actually, as we will see later, it may be far away from the optimal solution. Using the same topology in Figure 1, but applying a different routing algorithm, Figure 4 illustrates a simple case where SP routing algorithm can NOT achieve B_{max}. From Figure 1 we know that, the saturate bandwidth that the SP algorithm can achieve is $B_s(\mathcal{R}_{sp}) = 10$ with link (B, C) as the bottleneck link. Figure 4 shows an optimal routing algorithm, which can achieve $B_{max} = 50$ with (A, B) as the bottleneck link.[3] Thus the problem that this paper will address turns out to be finding a heuristic hop-by-hop routing algorithm $\mathcal{R}$ which can achieve a better $B_s(\mathcal{R})$.

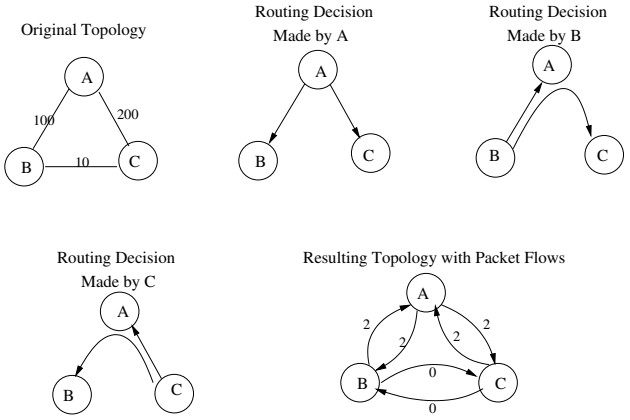


Figure 4: An example of an optimal algorithm for the given topology, which can achieve better saturate bandwidth allocation than the SP algorithm does.

In order to solve the OPR problem in polynomial time, heuristic approximation algorithms are needed. In section 4, we first apply two existing algorithms, the Widest-Shortest-Path (WSP) algorithm and the Bandwidth-inversion Short-Path (BSP) algorithm. Both of them are based on the generalized Dijkstra's algorithm. After showing both strength and weaknesses of them, we propose a novel approximation algorithm, called the Enhanced Bandwidth-inversion Shortest-Path (EBSP) algorithm. It is also a Dijkstra-based algorithm, therefore, it can run in $O(|V|^2)$ time, where $|V|$ is the total number of nodes in a given network. Extensive simulations in different network scenarios show that all three approximation algorithms outperform the hop-count

[3]Since link (A, B), which has capacity of 100, is shared by two flows $\overrightarrow{AB}$ and $\overrightarrow{CB}$, the maximum bandwidth for each flow is 50.

shortest-path algorithm on average. The results also confirm that, in a large-scale and heterogeneous network environment, our novel EBSP algorithm outperforms the other two existing algorithms in terms of much better saturate bandwidth.

We should emphasize that our contribution is in designing and evaluating routing algorithms which can find better paths for the premium traffic within a DiffServ domain, and in proving the correctness of the new EBSP algorithm, rather than in changing the DiffServ scheme itself. Moreover, we concentrate only on intra-domain IP layer routing.

The rest of the paper is organized as follows. After the presentation of our system model and assumptions in Section 2, we give rigorous NP-hardness proof of the OPR problem in Section 3. Three heuristic routing algorithms are then discussed and studied in Section 4. Simulations and results are illustrated in Section 5. Related work is covered in Section 6. Finally, Section 7 concludes this paper.

2. SYSTEM MODEL

We formally define a network model in this section. Based on this model, we give a more formal description of our assumptions and the problem we will address.

2.1 Network Model and Assumptions

Formally, a network is defined as a strongly connected directed graph $G(V, E)$, where V is the set of nodes (routers in the network) and E is the set of edges (links in the network), with cardinalities $|V|$ and $|E|$, respectively. Links are bidirectional with the same capacity in each direction. An edge from node x to y is represented as (x, y) and there is a positive bandwidth $b(x, y)$ associated with that edge. There is also a positive flow number $h(x, y)$, representing the total number of premium flows moving from x to y. Recall that all premium packets with the same source and destination addresses compose a premium flow. More generally, a weight function is associated with links, denoted as $w(x, y)$ if $(x, y) \in E$. The weight function may use metrics of a link, such as delay, bandwidth, hop count, etc., and map them into a real value. By default, we assume that the weight function is non-negative. A path p from v_1 to v_n is denoted as $p(v_1, v_n)$ (p_{v_1,v_n} for short) and $p(v_1, v_n) = \langle v_1, v_2, \cdots, v_{n-1}, v_n \rangle$, and it is *simple* if all nodes from v_1 to v_n are distinct. If v_1 and v_n are the same node, $p(v_1, v_n)$ forms a *loop*. We use $p \circ q$ to denote the concatenation of p and q. The same way as it is done for links, a weight function may be applied to paths too: $w(p(v_1, v_n)) = w(v_1, v_2) \oplus w(v_2, v_3) \oplus \cdots \oplus w(v_{n-1}, v_n)$, with $p(v_1, v_n)$ being a path from v_1 to v_n and "$\oplus$" being a binary operation. If the path p is empty, we have $w(p) = \bar{0}$. Weight values have a total order, denoted by "$\prec$". $w_1 \prec w_2$ means w_1 is *lighter* (better) than w_2, or w_2 is *greater* (worse) than w_1. For example, the capacity of a path is the minimum link capacity of all the links that comprise the path. Here $w_1 \oplus w_2$ means $min(w_1, w_2)$. Given a path $p(v_1, v_n) = \langle v_1, v_2, \cdots, v_{n-1}, v_n \rangle$ and a weight function w, specially, we define the *word-weight* of p, written $w_L(p)$, to be the sequence: $w_L(p) = w(p_{v_1,v_n})w(p_{v_1,v_{n-1}}) \cdots w(p_{v_1,v_2})$. Suppose we have two paths, p and p', with the word-weights $w_L(p) = \alpha_1\alpha_2 \cdots \alpha_n$ and $w(p') = \beta_1\beta_2 \cdots \beta_m$, respectively. We say p is *lexicographically lighter* than p' if either (i) $n < m$ and $\alpha_i = \beta_i$ for $1 \le i \le n$, or (ii) $\exists k$, $1 \le k \le \min(n, m)$, such that $\alpha_k \prec \beta_k$ and $\alpha_i = \beta_i$ for $1 \le i < k$. [25]

Given two nodes s and t in a network, we say that a path p^* is the:

- *lightest* path from s to t if $w(p^*) \preceq w(p)$, $\forall p$ from s to t;
- *S-lightest* path from s to t if p^* is a *lightest* path from s to t such that all of its sub-paths with source s are also *lightest* paths on their own;
- *L-lightest* path from s to t if its word-weight is lexicographically lighter than or equal to the word-weight of any other path from s to t.

Note that: (1) An S-lightest path is always a lightest path but not vice versa; and (2) An L-lightest path is always an S-lightest path but not vice versa.

For the premium class traffic, a bandwidth reservation from v_1 to v_n through a simple path $p(v_1, v_n)$ is successful if and only if the reservation with the same amount of bandwidth is successfully allocated on every link along the path. Therefore, each link (v_i, v_{i+1}) has to reserve certain amount of bandwidth not only for the premium traffic originated from v_i itself, but for all the transient traffic passing through all paths which are sharing the link as well.

Since premium class traffic experiences almost no queueing delays [11, 27], in this paper, we only consider bandwidth constraints.

The following are our assumptions:

1. Topology information can be obtained at each router by using some link-state routing protocol such as Open Shortest-Path First Protocol (OSPF).
2. Hop-by-hop routing is used in the given network.
3. Bandwidth demand for the premium traffic is homogeneous among all nodes, meaning that all nodes require the same amount of bandwidth to all other nodes for the premium traffic (denoted as B).

2.2 Optimal Premium-class Routing Problem

In Section 1, we have briefly introduced the Optimal Premium-class Routing (OPR) problem. Based on the system model and assumptions stated above, we give a more formal definition to the OPR problem as follows.

Given a network $G(V, E)$ with the bandwidth assignment $b(v_i, v_j)$ for any $v_i, v_j \in V$ and $(v_i, v_j) \in E$, and the assumptions outlined in 2.1, the OPR problem is to find the optimal routing scheme $\mathcal{R}_{opt}$ among all the possible loop-free hop-by-hop routing schemes on the network G (denoted as $\mathcal{R}_1, \mathcal{R}_2, \cdots, \mathcal{R}_n$), so that the optimal premium saturate bandwidth B_{max} is achieved, i.e., $B_s(\mathcal{R}_{opt}) = B_{max} = max\{B_s(\mathcal{R}_1), B_s(\mathcal{R}_2), \cdots, B_s(\mathcal{R}_n)\}$.

As stated in Section 6, a very interesting point of the OPR problem is that, an optimal local solution for a single source may not be the solution to the whole OPR problem. To address this problem, we need to consider both routing and bandwidth reservation in a global view.

2.3 Theoretical Definitions and Theorems

To solve the OPR problem, heuristic approximation algorithms are needed. Before introducing the heuristic algorithms, which we design and validate in Section 4, some fundamental definitions and theorems are presented in this section. We shall use them later to prove the correctness of those algorithms.

Since hop-by-hop routing scheme is used, in order to make the whole network work correctly, the routing algorithm should be able to guarantee the *consistency*, defined as follows.

DEFINITION 1. Routability: *A network $G(V, E)$ is said to be* routable *if every node $v \in V$ is able to find a simple path to all the other nodes.*

DEFINITION 2. Consistent routing: *Given a* routable *network $G(V, E)$, a routing scheme $\mathcal{R}$ is said to be* consistent *if (1) $\mathcal{R}$ can find a simple path between every pair of nodes in the network, i.e., for any two distinct nodes $s, t \in V$, $\mathcal{R}(s,t)$ is simple and non-empty, where $\mathcal{R}(s,t)$ denotes the path found by $\mathcal{R}$ between s and t; and (2) for any two distinct nodes $s, t \in V$, $\mathcal{R}(s,t) = \langle s, v_1, v_2, \cdots, v_n, t\rangle$ implies that $\mathcal{R}(v_i, t) = \langle v_i, v_{i+1}, \cdots, v_n, t\rangle$ (i.e., $\mathcal{R}(v_i, t)$ is the sub-path of $\mathcal{R}(s,t)$ between v_i and t) for all $i = 1, 2, \cdots, n$. That is, if node s makes a decision that the traffic to t will follow a certain path p, then all the on-path nodes along p should make the same decisions as s.*

THEOREM 1. *Given a routable network, a consistent hop-by-hop routing algorithm is loop-free.*

PROOF: The proof is straightforward. Given a routable network $G(V, E)$, every node $v \in V$ can find a simple path to every other node. Suppose a source node s chooses the path $\langle s, v_1, \cdots, v_n, t\rangle$ to the destination t, then according to the definition of *consistency*, $\langle v_i, v_{i+1}, \cdots, v_n, t\rangle$ is the sub-path from node v_i to t, for any $i = 1, \cdots, n$. By the definition of a *simple* path, there is no loop in the entire network. ■

Note that without consistency, loops may exist in the network, even if the path between every pair of nodes is simple. For example, if node a chooses the simple path $\langle a, b, c\rangle$ to node c, while node b chooses the simple path $\langle b, a, c\rangle$ to c, then there will be a forwarding loop between a and b in the network, although the two paths are loop-free individually.

Two topological properties, *isotonicity* and *strict isotonicity*, provide guarantees for the generalized Dijkstra's algorithms[4] to be consistent. Formal definitions of *isotonicity* and *strict isotonicity* were given in [25], and they state that the order relation between the weights of any two simple paths is preserved if both of them are either *prefixed* or *appended* by a common, third, simple path. That is, given two simple paths from s to t, say p and p', assuming $w(p) \preceq w(p')$ for some weight function w, then the *isotonicity* means: (i) if we prefix a common simple path q to both p and p', denoted as $q \circ p$ and $q \circ p'$ respectively, then $w(q \circ p) \preceq w(q \circ p')$; and (ii) if we append a common simple path q' to p and p', then $w(p \circ q') \preceq w(p' \circ q')$. In *strict isotonicity*, "$\preceq$" is replaced by "$\prec$". In [25], the authors also proved that if the strict isotonicity holds in a network for a weight function w, then the Dijkstra's algorithm can always find S-lightest paths (in terms of w) between nodes and it is sufficient to guarantee the routing consistency. However, if only the weaker condition, the isotonicity, holds, then a modified version of Dijkstra's algorithm, called the Dijkstra-Old-Touch-First (Dijkstra-OTF)[5], is needed to find L-lightest paths between nodes and guarantee the consistency.

As we will see next, an even weaker condition, called *left-isotonicity*, is sufficient to verify a *consistent* routing algorithm.

DEFINITION 3. Left-isotonicity: *Given a network topology G and a weight function w, let us consider any two paths p_1 and p_2, which are* prefixed *by a common path p, resulting in $p'_1 = p \circ p_1$ and $p'_2 = p \circ p_2$. Then the* **left-isotonicity** *holds, if $w(p_1) \preceq w(p_2)$ implies $w(p'_1) \preceq w(p'_2)$. Similarly, if $w(p_1) \prec w(p_2)$ implies $w(p'_1) \prec w(p'_2)$, then the* **strict left-isotonicity** *holds. If (strict) left-isotonicity holds for G, then G is said to be (strictly) left-isotonic.*

As we can see, the left-isotonicity states that the order relation between the weights of any two paths is preserved if both of them are *prefixed* by a common, third path. However, the order relation is NOT necessarily preserved if both of them are *appended* by a common, third, simple path.

Similarly, we define the *right-isotonicity* (R-isotonicity for short) by the other half of the isotonicity, that is, if two paths are appended by a common, third path, then their previous order relation holds.

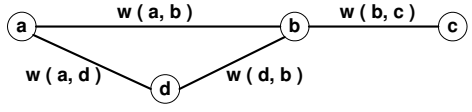


Figure 5: Suppose $w(a,d) \oplus w(d,b) \prec w(a,b)$ and $w(a,d) \oplus w(d,b) \oplus w(b,c) \succ w(a,b) \oplus w(b,c)$. Then neither S-lightest path nor L-lightest path exists between a and c.

Being weaker than isotonicity, left-isotonicity can not guarantee the existence of S-lightest or L-lightest paths between nodes. To see this, a simple example is shown in Figure

[4]In this paper, we use the term *generalized Dijkstra's algorithm* to refer to a Dijkstra-based algorithm which computes the "lightest", rather than the "shortest", path between two nodes. That is, the algorithm can handle some non-additive metrics.

[5]The Dijkstra-OTF computes L-lightest paths between nodes without explicitly computing or comparing word-weights.

5. Since the right half of isotonicity does not hold, it is valid that $w(a,d) \oplus w(d,b) \prec w(a,b)$ and $w(a,d) \oplus w(d,b) \oplus w(b,c) \succ w(a,b) \oplus w(b,c)$. Thus, path $\langle a,b,c \rangle$ is the only lightest path from a to c, and its sub-path $\langle a,b \rangle$ is not a lightest path. Therefore, neither S-lightest path nor L-lightest path exists from a to c in this scenario.

However, because we assume that weights are non-negative values, the lightest path always exists between each pair of nodes. In fact, we will argue later in Theorem 2 that given a strictly left-isotonic network G and a routing algorithm $\mathcal{R}$, $\mathcal{R}$ is consistent if it can guarantee to find the lightest path between every pair of nodes in G.

THEOREM 2. *Given a routable network $G(V,E)$, if the strict left-isotonicity property holds in G, then a routing algorithm $\mathcal{R}$ is consistent, as long as it can guarantee to always find the lightest path between any pair of nodes in G.*

PROOF: We prove this theorem by contradiction. Suppose there are multiple paths between a source node $s \in V$ and a destination node $t \in V$, and by running the algorithm $\mathcal{R}$ at s, a lightest path $p = \langle s, v_1, \cdots, v_n, t \rangle$ has been found. Let us assume $\mathcal{R}$ is not consistent, i.e., $\exists i, 1 \leq i \leq n$, such that v_i is the first node along the path p which picks up another different path $p' = \langle v_i, v'_{i+1}, \cdots, v'_j, t \rangle$ as the lightest path from v_i to t. Then, $w(p') \prec w(p_{i,t})$, where $p_{i,t} = \langle v_i, v_{i+1}, \cdots, v_n, t \rangle$ is the sub-path of p between v_i and t. Following the strict left-isotonicity, if we append a common prefix $q = \langle s, v_1, \cdots, v_i \rangle$ onto both p' and $p_{i,t}$, the order relation between two paths should be preserved, i.e., $w((q \circ p')) \prec w((q \circ p_{i,t}))$, where "$\circ$" stands for path concatenation. As we know, $q \circ p_{i,t} = p$, that is, $w((q \circ p')) \prec w(p)$. Therefore, p is NOT the lightest path between s and t, which contradicts our assumption and completes the proof. ■

Since another half of the isotonicity property (*Right-isotonicity*) does not hold, no existing Dijkstra-based algorithm can guarantee to find the lightest path between every pair of source and destination, thus incurring inconsistency. The following figure shows a counterexample.

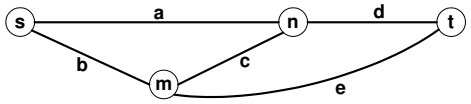


Figure 6: Suppose $a \prec (b \oplus c), (a \oplus d) \succ (b \oplus c \oplus d), (c \oplus d) \prec e, (b \oplus c \oplus d) \prec (b \oplus e), (b \oplus e) \prec (a \oplus d)$. Then inconsistency occurs.

In the case of Figure 6, we assume that:

(1) $a \prec (b \oplus c)$

(2) $(a \oplus d) \succ (b \oplus c \oplus d)$

(3) $(c \oplus d) \prec e$

(4) $(b \oplus c \oplus d) \prec (b \oplus e)$

(5) $(b \oplus e) \prec (a \oplus d)$

where (1) and (2) both hold because *R-isotonicity* is NOT preserved. On the other hand, both (3) and (4) hold due to *L-isotonicity* property. The relationship given in (5) leads to an inconsistency.

By running Dijkstra's algorithm at node s, the path from s to t found by node s will be $\langle s,m,t \rangle$. Obviously, it is NOT the lightest path $\langle s,m,n,t \rangle$.[6] However, if we run the same Dijkstra's algorithm at the on-path node m, the path from m to t found by m will be $\langle m,n,t \rangle$. Inconsistency occurs. Therefore, any existing generalized Dijkstra's algorithms are not sufficient to provide consistent routing any more. The fundamental reason behind this is that the existing generalized Dijkstra's algorithms can NOT find the lightest paths if such paths contain any non-optimal sub-paths (i.e., when the *isotonicity* does not hold). So, a new Dijkstra-based algorithm is needed to find lightest paths between nodes in strictly left-isotonic networks.

However, we observe that the strict left-isotonicity property does have a very nice feature which is shown in the following lemma.

LEMMA 1. *Given a strictly left-isotonic network $G(V,E)$, if a path $p = \langle s, v_1, \cdots, v_n, t \rangle$ is the lightest path between two nodes $s, t \in V$, then every sub-path $p_{i,t} = \langle v_i, \cdots, v_n, t \rangle$ (where $1 \leq i \leq n$) is the lightest path from v_i to t.*

PROOF: We can prove this lemma by contradiction, which is similar to the proof of Theorem 2. Due to space limitation, we omit the detailed proof in this paper. ■

Following Lemma 1, the lightest path from s to t must be based on the lightest paths from any intermediate nodes to t. If we consider a destination node t, and the lightest paths from all other nodes to t, then we can see that the lightest paths form exactly a *spanning tree* for the given network G rooted at t. Therefore, to find the lightest path from any s to t, we can start from the destination node t and perform relaxation step-by-step backward to s. Based on the above observation, we present a new algorithm WN-DIJKSTRA to find the lightest path from any s to t in a strictly left-isotonic network G, as shown in Algorithm 1.

In WN-DIJKSTRA, $\varpi[u]$ denotes the successor (or next hop node) of u, and $d[v]$ denotes the weight of the current path from v to t. The function call EXTRACT-MIN(Q) is the same as in the original Dijkstra's algorithm, which extracts a node from set Q with the lightest value of $d[v]$.

It is clear that WN-DIJKSTRA takes $O(|V|^2)$ execution time, the same as the original Dijkstra's algorithm.

THEOREM 3. **(Correctness of WN-Dijkstra's algorithm)** *If we run WN-Dijkstra's algorithm on a strictly*

[6]The direct link (s,n) is chosen by s as the lightest path to n, since $a \prec (b \oplus c)$ (Condition (1) above). Thereafter, (s,n) and its weight a, rather than the non-optimal sub-path $\langle s,m,n \rangle$ and its weight $(b \oplus c)$, are used to continue the relaxation from node n. This eliminates the chance of finding the real lightest path, which passes exactly through $\langle s,m,n \rangle$, since Condition (4) and (5) hold above.

```
WN-DIJKSTRA(G(V,E), w, s, t)
 1  for each node v ∈ V
 2  do d[v] ← ∞
 3     ϖ[v] ← NIL
 4  d[t] ← 0
 5  Q ← V
 6
 7  while Q ≠ ∅
 8  do u ← EXTRACT-MIN(Q)
 9     if u = s
10       then EXIT
11     for each v ∈ Adj[u]
12     do if w(v,u) ⊕ d[u] ≺ d[v]
13          then ϖ[v] ← u
14               d[v] ← w(v,u) ⊕ d[u]
```

Algorithm 1: WN-Dijkstra algorithm which finds the lightest path from s to t.

left-isotonic network $G(V,E)$ with non-negative weight function w, source s and destination t, then the lightest path is found from s to t with $d[s] = (weight\ of\ the\ lightest\ path)$ at termination.

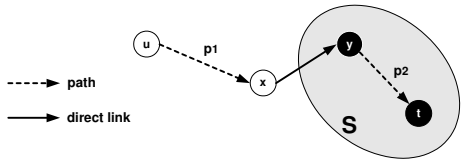


Figure 7: The proof of Theorem 3: path p can be decomposed into two sub-paths p_1 and p_2.

PROOF: Let $\delta(a,b)$ denote the weight of the lightest path from a to b. We claim that for each node $u \in V$, we have $d[u] = \delta(u,t)$ at the time when u is extracted from Q and this equality is maintained thereafter. Since $s \in V$, this claim holds for s, too. For convenience, we let S be the set of nodes that are extracted from Q before u.

We shall show this claim by contradiction. Let u be the first node for which $d[u] \neq \delta(u,t)$ when it is extracted from Q. Clearly, $u \neq t$ because t is the first node extracted from Q and $d[t] = \delta(t,t) = \bar{0}$ when it is extracted from Q. Because $u \neq t$, we have $S \neq \emptyset$ just before u is extracted from Q. Since the network is routable, there must be some path from u to t. Thus there must be a lightest path p from u to t, for the weight function is non-negative. Without loss of generality, we assume that x is the first node along p such that $x \in (V - S)$ (i.e., x is the first node along p which has not been extracted from Q yet), and let $y \in S$ be x's direct successor, i.e., $y = \varpi[x]$. Then p can be decomposed into two sub-paths p_1 and p_2 at x, as shown in Figure 7.

Now we show that $d[x] = \delta(x,t)$ when it is extracted from Q. Because u is chosen as the first node for which $d[u] \neq \delta(u,t)$ when it is extracted from Q, and because y is extracted from Q before u, we have $d[y] = \delta(y,t)$ when y is extracted from Q. Since p is the lightest path from u to t, and G is strictly left-isotonic, by Lemma 1, the sub-path from x to t is also a lightest path and thus $d[x] = w(x,y) \oplus d[y] = w(x,y) \oplus \delta(y,t) = \delta(x,t)$.

Because the weight function is non-negative, clearly we have $\delta(x,t) \preceq \delta(u,t)$, and thus

$$d[x] = \delta(x,t) \preceq \delta(u,t) \preceq d[u] \quad (1)$$

However, because x is still in Q when u is extracted from Q, we have $d[u] \preceq d[x]$. By Equation 1, the following equations hold:

$$d[x] = \delta(x,t) = \delta(u,t) = d[u]$$

Therefore, $d[u] = \delta(u,t)$, which contradicts our choice of u. We conclude that for any node $s \in V$, when it is extracted from Q, $d[s] = \delta(s,t)$, and the path found and stored in $\varpi[s]$ is the lightest path from s to t. ■

By Theorem 2 and Theorem 3, the WN-DIJKSTRA algorithm provides a *consistent* routing.

In Section 4, we will provide three heuristic algorithms to address the OPR problem. Their correctness is proven based on the definitions and theorems introduced above.

3. NP-HARDNESS OF OPR PROBLEM

In the hop-by-hop routing context, the routing consistency must be guaranteed (Section 2.3). Therefore, in a feasible solution to the OPR problem, if we consider any destination node t, and paths that all other nodes are taking to t, then the paths must form a spanning tree for the entire network rooted at t itself.

For the purpose of showing that the OPR problem is NP-Hard, it can be replaced by the following simplified problem P1.

P1 - *The simplified OPR problem*
INSTANCE: A network $G(V,E)$ and a destination node $t \in V$, where all edges in E have the same constant capacity C, and C is a non-negative integer.
QUESTION: Is there a spanning tree T in G, rooted at t, that satisfies the following constraint: for $\forall e \in T$, if $U(e) = \{u \in V |$ the path from u to t contains $e\}$, then $|U(e)| = (\sum_{u \in U(e)} 1) \leq C$?

As we can see, the problem P1 simplifies the original OPR problem in the following two senses: (1) The link capacities are unified to a constant integer C; (2) Instead of considering all nodes in V, only one node $t \in V$ is considered in P1. It is apparent that the original OPR problem is harder than P1. Therefore, our objective is to show that even Problem P1 is hard to solve.

Actually, the problem P1 is also a simplified variant of the *Capacitated Minimum Spanning Tree* problem in [7], where the length constraint is lifted and the edge capacities are unified to the same constant integer C.

At the first look, problem P1 seems very simple. If there is not the tree constraint (i.e., if T is not necessary to be a

tree), the problem becomes a special case of the *Maximum-Flow problem*, which is polynomially solvable by using the Ford-Fulkerson method [5]. However, problem P1 is surprisingly difficult to solve. Indeed, it is NP-hard. The NP-hardness is due to the combination of the tree constraint (due to the consistency requirement from hop-by-hop routing) and its essence of the *Bin Packing problem* [7]. It is clear that, the tree constraint deprives the nodes of their independence to choose paths to the destination t, thereby making it hard to solve. In fact, the NP-hardness holds even for a *Directed Acyclic Graph* (DAG).

Before we give the formal NP-hardness proof of P1, we briefly introduce one existing NP-hard problem, the *Non-uniform Load Balancing problem* [14, 10], from which we shall construct our reduction to P1. For ease of notation, we denote the *Non-uniform Load Balancing problem* as P0.

P0 - *The Non-uniform Load Balancing problem*
INSTANCE: A set of jobs $J = j_1, j_2, \cdots, j_k$, and a set of machines $M = m_1, m_2, \cdots, m_n$; for each job $j_i \in J$, there is a set $S_i \subset M$ on which j_i can be run; each job j_i has a *requirement* r_i which is equal to either 1 or 2.
QUESTION: Is there an assignment from J to M such that each job $j \in J$ is assigned to a machine $m \in M$ so that the sum of the requirements assigned to each machine is at most 2?

Problem P0 is NP-hard [14, 10]. Now we prove that the problem P1 is NP-hard, too.

THEOREM 4. *The simplified OPR problem,* P1*, is NP-hard.*

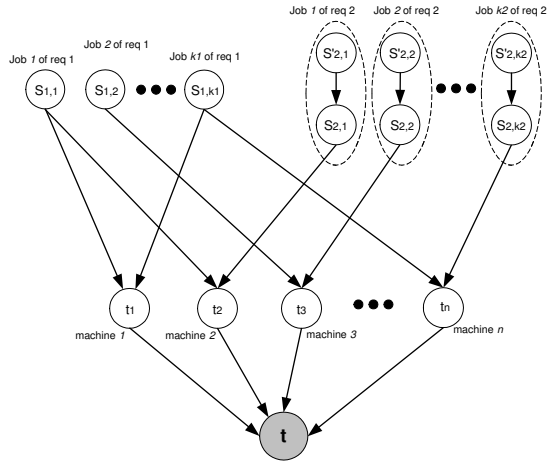


Figure 8: The NP-hardness reduction

PROOF: The reduction algorithm begins with an instance of P0. We construct a graph $G(V, E)$ to encode the instance of P0 step by step as follows. The construction is also shown in Figure 8.

(1) Without loss of generality, we suppose that, among k jobs in J, there are k_1 jobs with requirements of 1 and k_2 jobs with requirements of 2. For simplicity of notation, we denote by J_1 the set of jobs with requirements of 1 and J_2 the set of jobs with requirements of 2, respectively. Then, we have $|J_1| = k_1$ and $|J_2| = k_2$. It is apparent that $J = J_1 \bigcup J_2$ and $k = k_1 + k_2$.

(2) For each job j_i in J_1, we add one single node $s_{1,i}$ into V. For each job j_l in J_2, firstly, we add two nodes $s_{2,l}$ and $s'_{2,l}$ into V; secondly, a link between these two nodes, $(s'_{2,l}, s_{2,l})$ is added into E, as illustrated in Figure 8.

(3) For each machine m_i in M, we place a node t_i into V as well.

(4) A termination node, t, is added into V. For each node t_i, a link (t_i, t) is added into E, which connects t_i to t. So far, the set of nodes, V, is completed. We have

$$V = \{s_{1,1}, \cdots, s_{1,k_1}\} \cup \{s_{2,1}, \cdots, s_{2,k_2}\} \cup \{s'_{2,1}, \cdots, s'_{2,k_2}\} \cup \{t_1, \cdots, t_n\} \cup \{t\}$$

and $|V| = k_1 + 2k_2 + n + 1 = k + k_2 + n + 1$.

(5) For each job j_i in J, since it has a machine set $S_i \subset M$ on which it can run, for each machine $m_u \in S_i$, if j_i is in J_1, then we add one link $(s_{1,i}, t_u)$ into E; or if j_i is in J_2, then we add a link $(s_{2,i}, t_u)$ into E instead. Through these links, we encode the condition that job j_i can only be assigned to a machine in S_i, that is, node t_u is reachable from $s_{x,i}$ if and only if $m_u \in S_i$, where $x = 1$ or 2. Till now, the link set E is completed, too.

(6) Finally, we assign all links in E with the same capacity $C = 3$.

Figure 8 illustrates the construction of G. It is clear that such construction of G can be completed within polynomial time. Now the Problem P0 has been transformed into an instance of Problem P1 where the constructed G is the graph in P1, $C = 3$ and the question is to look for a spanning tree T for G that satisfies the capacity constraints.

We show that this transformation of the instance of P0 into G is a reduction. First, if there is a feasible allocation of jobs to machines in Problem P0, then it is not difficult to come up with a spanning tree T for G accordingly. Specifically, for any job $j_i \in J_1$, if it is assigned to machine m_u, then we take the corresponding link $(s_{1,i}, t_u)$; for any job $j_i \in J_2$, if it is assigned to machine m_u, then we take both links $(s'_{2,i}, s_{2,i})$ and $(s_{2,i}, t_u)$. Furthermore, we take all links (t_u, t). As such, we show that the resulting sub-graph, G', is a spanning tree for G: (1) since all jobs are assigned to some machines, all s_i nodes and s'_j nodes are covered in the resulting sub-graph $G'(V', E')$; since all links (t_u, t) are taken, all t_u nodes and t are covered. That is, all nodes in V are covered in G' $(V = V')$. (2) Since each job can be assigned to at most one machine, there are exactly k $(s_{x,i}, t_u)$ links in E', where $x = 1$ or 2; plus k_2 links $(s'_{2,i}, s_{2,i})$ and n links (t_u, t), we have totally $k + k_2 + n = k_1 + 2k_2 + n$ links in E'. By the construction of G, $|V| = k_1 + 2k_2 + n + 1$. So we have $|E'| = |V| - 1$, thereby making G' a spanning tree T rooted at t. (3) Since each machine has capacity 2, it is apparent that $|U(e)| = (\sum_{u \in U(e)} 1) \leq 3$ for any $e \in E'$, that is, the capacity constraints are simply satisfied. Therefore, T is a solution to Problem P1 on G.

Conversely, suppose that there is a solution T satisfying P1, i.e., T is a spanning tree for G rooted at t such that the capacity constraint of each link is preserved. Since $C = 3$, for each node t_u, at most 2 nodes can be contained in the sub-tree rooted at t_u. (Node t_u itself consumes 1 unit from the link (t_u, t). Therefore, at most two other paths can pass through this link) Since for each node $s'_{2,i}$, there is no direct link between it and t_u or t (there is only one link connecting it to corresponding $s_{2,i}$), its path to t has to pass through the corresponding node $s_{2,i}$. Furthermore, the path has to follow the same path from $s_{2,i}$ to t (otherwise, T will no longer be a tree). So the nodes $s'_{2,i}$ and $s_{2,i}$ are bound to each other in T. Hence, each sub-tree rooted at t_u can only have the following four options: contains 0 node, or any 1 node $s_{1,i}$, or any 2 nodes $s_{1,i}$ and $s_{1,j}$, or any one pair of bound nodes $s'_{2,i}$ and $s_{2,i}$. Following the spanning tree T, we can easily find an allocation of jobs accordingly: for any link $(s_{x,i}, t_u)$ in T, we just simply assign the corresponding job j_i to the machine m_u. As we discussed above, the link capacity constraint $C = 3$ guarantees that the workload at each machine does not exceed 2. Moreover, since all nodes in V are covered by T, all jobs are assigned. Therefore, the corresponding allocation is a solution to the instance of Problem P0. ■

As we can see from the proof and Figure 8, the NP-hardness holds even for a *Directed Acyclic Graph* (DAG).

4. ROUTING ALGORITHMS FOR PREMIUM TRAFFIC

The OPR problem, as we have defined in Section 2 and studied in Section 3, turns out to be very difficult to find an optimal solution in polynomial time since it is NP-hard. Therefore, in this section, we introduce two existing heuristic hop-by-hop routing algorithms that are based on the generalized Dijkstra's algorithm, as well as one novel algorithm based on the WN-Dijkstra's algorithm (Section 2). All of them can run in $O(|V|^2)$ [5].

4.1 Widest-Shortest-Path Algorithm (WSP)

The WSP algorithm is the simplest heuristic algorithm which can achieve a better saturate bandwidth ($B_s(\mathcal{R}_{wsp})$) than the basic hop-count shortest-path (SP) algorithm. When a tie occurs, the basic SP algorithm simply chooses the node with the smallest identifier to break the tie. However, the WSP always chooses the widest path among the set of shortest paths between any pair of source and destination. The WSP has been well-studied in [25, 15]. Although it does not have a *strict* isotonicity, the isotonicity still holds. Therefore, by using the Dijkstra-OTF algorithm proposed in [25], we can guarantee that WSP finds a loop-free L-lightest path from any source to any destination.

The simulation results (given in Section 5) show that, on average, the WSP algorithm outperforms the SP algorithm. However, since the WSP only chooses a path from those "shortest" paths (in the sense of hop-count) between two nodes, the gain is limited.

4.2 Bandwidth-inversion Shortest-Path Algorithm (BSP)

In order to overcome the WSP's limitation of performance gain, we need to choose a "best" path from a broader range. Therefore, the BSP algorithm is introduced. The BSP algorithm was studied and called the "Shortest-distance path algorithm" in [15]. It was used in a call-based or connection-oriented network (such as the ATM network).

The BSP is basically a shortest-path algorithm with the distance or weight function defined as

$$w(v_i, v_j) = \frac{1}{b_{i,j}}$$

and

$$w(p\langle v_1, v_2, \cdots, v_n\rangle) = \sum_{i=1}^{n-1} \frac{1}{b_{i,i+1}}$$

where $b_{i,j} = b(v_i, v_j)$ is the bandwidth of link (v_i, v_j). Since the weight function is additive and the strict isotonicity property holds, the BSP algorithm with this weight function can guarantee that packets are transmitted through the lightest path between any pair of source and destination without loop [25]. Therefore, it can be used as a hop-by-hop routing algorithm as well. If there are more than one lightest paths between the source and destination, the path with the least hop count is selected.

Although generally BSP can achieve much better saturate bandwidth ($B_s(\mathcal{R}_{bsp})$) than SP or WSP (the simulation results will be shown in Section 5), its performance varies drastically if topology changes, meaning that the chance of producing a worse $B_s(\mathcal{R}_{bsp})$ than the SP is high. For example, in Figure 1, if we change the $b(A, C)$ to 1000 and $b(B, C)$ to 90, then $B_s(\mathcal{R}_{bsp}) = 50$ and $B_s(\mathcal{R}_{sp}) = 90$, that is, the BSP gets even worse saturate bandwidth than the SP does.

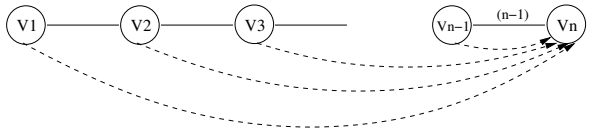


Figure 9: Links in a longer path have to handle more flows, therefore, B_s decreases: for example, according to the consistency of hop-by-hop routing, link (v_{n-1}, v_n) has to hold $(n-1)$ flows: $v_1\vec{v}_n, v_2\vec{v}_n, \cdots, v_{n-1}\vec{v}_n$. The longer the path is, the smaller a B_s will be.

The reason behind the failure of BSP is that it prefers a wider path too much. In fact, being related to both the link capacities along the path and the number of flows which are taking this path, there are two constraints behind the OPR problem: (1) On the one hand, a wider path may achieve higher saturate bandwidth; (2) On the other hand, when the wider path grows longer, according to the consistent routing policy, more nodes, hence more flows will have to share the same path, resulting in a decrease of the saturate bandwidth.[7] This phenomenon is illustrated in Figure 9.

[7]According to the hop-by-hop routing assumption, every node makes routing decisions independently without any coordination between each other. Therefore, taking only the bandwidth into consideration, we may experience the following behavior: if one node chooses a wider link, then other

Therefore, in order to prevent putting too many flows onto a wide path, the length of the chosen path should be carefully limited. Intuitively, a "penalty" associated with the hop count of a path can be used to prevent that it becomes too long. Hence, a novel algorithm based on the BSP is introduced as follows.

4.3 Enhanced Bandwidth-inversion Shortest-Path Algorithm (EBSP)

Intuitively, the introduction of a "penalty" helps to prevent a path becoming too long. The value of such "penalty" is related to hop count value. In order to guarantee the routing consistency, the new weight function with "penalty" should hold at least the left-isotonicity property (Section 2). One direct option is to add a constant term to the link cost, e.g. for a link of bandwidth b, its weight function can be (say) $1/b + c$, where c is a non-negative constant [22]. (The weight function of BSP then becomes a special case when $c = 0$.) However, the "penalty" may not necessarily be linear with the hop count. We claim that this penalty is likely to be exponential with respect to the hop count. The reason is briefly presented as follows. If we consider only one destination t, the routes from all the other nodes to t forms a spanning tree (also see Section 3). Hence, given a link e, the totally number of routes going through e to t could increase exponentially with respect to the depth of the sub-tree rooted at one endpoint of e. Therefore, if we observe conversely from a node v in the downstream of e, when we reach e during searching the route from v to t, the number of routes on e could potentially be an exponential function of the hop count of e along the current searching path. Therefore, we should use an exponential penalty in the new weight function.

The new weight function for a path $p(v_1, v_n) = \langle v_1, v_2, \cdots, v_n \rangle$ is then defined as follows.

$$w(p) = \sum_{i=1}^{n-1} \frac{\theta^{i-1}}{b_{i,i+1}} \tag{2}$$

where θ is a constant discount factor (in the first part of our simulation, we let $\theta = 2$). We can see that: (1) θ^i serves as an exponential penalty; and (2) for any two paths p_1 and p_2, $w(p_1) \prec w(p_2)$ if and only if $w(p_1) < w(p_2)$. Another interesting observation is that the BSP algorithm now becomes one special case of EBSP in which $\theta = 1$. On the other hand, when θ is set to be large enough, the EBSP tends to be another variant of the SP algorithm, just like the WSP. This observation leads us to a hint that we could find an optimal θ^* such that the EBSP algorithm yields even larger saturate bandwidth. We will address this issue later in our simulations.

Notice that the new weight function is no longer static. It dynamically changes with the hop count value. Before we use a generalized Dijkstra algorithm to find the lightest path between two nodes in terms of this new weight function, we should show that the strict left-isotonicity holds

nodes are very likely to choose the same link too, resulting in the decrease of B_s. This confirms our earlier statement that a local optimal solution for a single source node may not necessarily be the global optimal solution to the OPR problem.

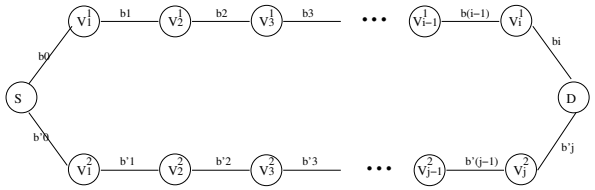


Figure 10: The topology used to prove the left-isotonicity of the weight function in Equation 2. Originally, there are two paths between node S and D: $p_1 = \langle S, v_1^1, v_2^1, \cdots, v_i^1, D \rangle$ and $p_2 = \langle S, v_1^2, v_2^2, \cdots, v_j^2, D \rangle$. We assume that p_1 is lighter than p_2.

for this new weight function. Suppose we have two paths from node S to D: $p_1 = \langle S, v_1^1, v_2^1, \cdots, v_i^1, D \rangle$ and $p_2 = \langle S, v_1^2, v_2^2, \cdots, v_j^2, D \rangle$, as shown in Figure 10. For convenience, let $S = v_0^1 = v_0^2$ and $D = v_{i+1}^1 = v_{j+1}^2$. According to the definition in Equation 2, we have

$$w(p_1) = \sum_{m=0}^{i} \frac{\theta^m}{b_m}$$

and

$$w(p_2) = \sum_{m=0}^{j} \frac{\theta^m}{b'_m}$$

where $b_m = b(v_m^1, v_{m+1}^1)$ and $b'_m = b(v_m^2, v_{m+1}^2)$. Without loss of generality, we suppose that p_1 is lighter than p_2, i.e., $w(p_1) \prec w(p_2)$. Then we have

$$\sum_{m=0}^{i} \frac{\theta^m}{b_m} < \sum_{m=0}^{j} \frac{\theta^m}{b'_m} \tag{3}$$

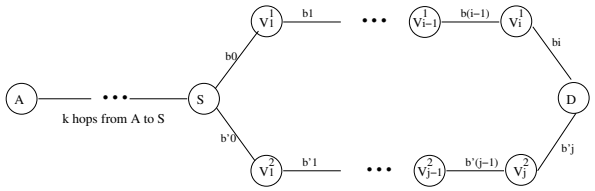


Figure 11: Topology used to prove the left-isotonicity of the weight function in Equation 2. After a common prefix is added to the original topology (Figure 10) at node S, we can prove that $p'_1 = \langle A, \cdots, S, v_1^1, v_2^1, \cdots, v_i^1, D \rangle$ is still lighter than $p'_2 = \langle A, \cdots, S, v_1^2, v_2^2, \cdots, v_j^2, D \rangle$. It means, the strict left-isotonicity holds.

Now a common prefix $q = \langle A, \cdots, S \rangle$ is added to the node S as shown in Figure 11. Suppose there are k hops from node A to S, then the weights for the two augmented paths from A to D, $p'_1 = q \circ p_1 = \langle A, \cdots, S, v_1^1, v_2^1, \cdots, v_i^1, D \rangle$ and $p'_2 = q \circ p_2 = \langle A, \cdots, S, v_1^2, v_2^2, \cdots, v_j^2, D \rangle$, are

$$w(p'_1) = w(A, S) + \sum_{m=0}^{i} \frac{\theta^{m+k}}{b_m} = w(A, S) + \theta^k \sum_{m=0}^{i} \frac{\theta^m}{b_m}$$

and

$$w(p_2') = w(A,S) + \sum_{m=0}^{j} \frac{\theta^{m+k}}{b'_m} = w(A,S) + \theta^k \sum_{m=0}^{j} \frac{\theta^m}{b'_m}$$

respectively. Following the inequality in Equation 3, we can easily draw the conclusion that $w(p_1') \prec w(p_2')$. Therefore, by Definition 3, the strict left-isotonicity holds. We can also show easily that the new weight function is NOT isotonic.

Following Theorem 2 and Theorem 3, an enhanced WN-Dijkstra's algorithm can be used to produce a consistent routing scheme for a network based on the new weight function given in Equation 2. We give the detailed description of this enhanced WN-Dijkstra's algorithm as follows, which is called *WN-Dijkstra-EBSP*.

```
WN-DIJKSTRA-EBSP(G(V,E), b, s, t)
 1  for each node v ∈ V
 2  do d[v] ← ∞
 3     ϖ[v] ← NIL
 4  Q ← V
 5  d[t] ← 0
 6
 7  while Q ≠ ∅
 8  do u ← EXTRACT-MIN(Q)
 9     if u = s
10        then EXIT
11     for each v ∈ Adj[u]
12     do if θ × d[u] + 1/b(v,u) < d[v]
13           then ϖ[v] ← u
14                d[v] ← θ × d[u] + 1/b(v,u)
```

Algorithm 2: WN-Dijkstra-EBSP algorithm which handles the OPR problem.

The asymptotic execution time of WN-DIJKSTRA-EBSP is $O(|V|^2)$, which is the same as the original Dijkstra's algorithm. Notice that we use the new weight function in Equation 2 to compute the weight for a path in line 12 and 14, which depends on the current node's hop count value and the bandwidth of the associated link.

By taking both the hop count and bandwidth into consideration, we expect that the EBSP algorithm can outperform both the SP and BSP algorithms with respect to the saturate bandwidth (i.e., $B_s(\mathcal{R}_{ebsp})$ should be better than $B_s(\mathcal{R}_{sp})$ and $B_s(\mathcal{R}_{bsp})$). We will show in Section 5 that the simulation results confirm our expectations very well.

5. SIMULATIONS AND RESULTS

In the previous section, we introduced three heuristic algorithms to solve the OPR problem. Their performance varies in different network topologies. To reveal the relationship between the performance and underlying network topologies, as well as to show the advantage of our new EBSP algorithm, extensive simulations have been carried out. In this section, we present these simulations and their results. Note that, although the algorithms can be used in a dynamic manner (for example, they can be plugged into a link state protocol, such as the OSPF), our simulations assume static routing. That is, given a topology, we perform routing statically.

5.1 Simulation Model

We design a topology generator to automatically generate topologies. The parameters we use for modeling topologies are: (1) Number of nodes N; (2) Maximum degree of each node D; and (3) Variance index of link capacities C_{var}. Specifically, we first generate N different nodes. Then, given a specific value of D, we generate a random number within $[1, D]$ for each node as its degree. The link destinations are also randomly selected among all nodes except the node itself. We normalize the base link capacity to 100 units and quantize C_{var} into 10 levels, from 1 to 10. For each link, we assign its capacity based on the value of C_{var}. For example, if $C_{var} = c, c \in [1, 10]$, we generate a random number between $[100, 100c]$ and assign it to a link as its capacity. The larger C_{var} is, the greater variance of link capacities. In this way, we can adjust the variance of link capacities easily by simply changing the value of C_{var}.

Since the OPR problem is NP-hard, it is impossible to find the optimal saturate bandwidth B_{max} for a given topology within polynomial time. Hence, comparisons between B_{max} and $B_s(\mathcal{R}_{wsp})$, $B_s(\mathcal{R}_{bsp})$, and $B_s(\mathcal{R}_{ebsp})$ are therefore infeasible. In order to quantify the performance and compare the three algorithms, we use the SP algorithm (the basic shortest-path algorithm) as our benchmark. Given a randomly generated topology G,[8] we first run the SP algorithm at each node of G so that the saturate bandwidth $B_s(\mathcal{R}_{sp})$ is obtained. Then, the three heuristic algorithms, WSP, BSP and EBSP, are executed on the same topology one by one. Their saturate bandwidths, $B_s(\mathcal{R}_{wsp})$, $B_s(\mathcal{R}_{bsp})$, and $B_s(\mathcal{R}_{ebsp})$ are obtained, respectively. We compare these three values with the benchmark, $B_s(\mathcal{R}_{sp})$, to observe how much benefit we can gain by using these three algorithms.

More specifically, we evaluate the performance of a particular algorithm $\mathcal{R}$ by using two metrics defined as follows.

(1) *Bandwidth Gain (β)* and *Average Bandwidth Gain ($\bar{\beta}$)*: β is defined as

$$\beta = \frac{B_s(\mathcal{R})}{B_s(\mathcal{R}_{sp})} \quad ,$$

where $B_s(\mathcal{R})$ can be $B_s(\mathcal{R}_{wsp})$, $B_s(\mathcal{R}_{bsp})$, or $B_s(\mathcal{R}_{ebsp})$. For each configuration (N, D, C_{var}), totally 2000 topologies are randomly generated and simulated. The *Average Bandwidth Gain* ($\bar{\beta}$) is then computed by

$$\bar{\beta} = \frac{\sum_1^{2000} \beta}{2000} \quad .$$

The larger $\bar{\beta}$ the algorithm $\mathcal{R}$ can achieve, the better.

(2) *Miss Rate (γ)*: For each given configuration (N, D, C_{var}), γ is simply measured as the fraction of total 2000 simulated topologies in which the algorithm $\mathcal{R}$ yields worse saturate bandwidth than $\mathcal{R}_{sp}$ does (i.e., $B_s(\mathcal{R}) < B_s(\mathcal{R}_{sp})$). That is,

$$\gamma = \frac{\text{Number of topologies in which } B_s(\mathcal{R}) < B_s(\mathcal{R}_{sp})}{\text{Total number of simulated topologies (2000)}}.$$

[8] If G is not *routable*, we simply drop it and generate another one.

The smaller γ, the better.

The following subsection demonstrates the detailed simulation results and analysis.

5.2 Simulation Results and Analysis

We present our simulation results in three parts. The first part (Section 5.2.1) focuses on the *Average Bandwidth Gain ($\bar{\beta}$)* and the *Miss Rate (γ)* with respect to different topology configurations, where we fix $\theta = 2$ in the EBSP algorithm. In the second part (Section 5.2.2) we study how θ affects the performance of the EBSP algorithm. Finally, we make a stronger and more realistic case in the third part (Section 5.2.3), applying some larger and more realistic network topologies generated by the famous topology generator BRITE. Note that in the first two parts, we do not intend to simulate realistic networks or large-scale ones. We focus only on the algorithms' average performance and its relationship with different topology configurations. Thus, a large number of topologies (2000 for each topology configuration) are randomly generated and simulated. While in the third part, we focus on evaluating the algorithms' performance in more realistic networks with much more nodes (up to 500).

5.2.1 Performance In Various Topologies

We illustrate the detailed simulation results in Figures 12, 13, 14 and 15. Each value of *bandwidth gain* or *miss rate* in these figures is the *average* value on 2000 randomly generated topology samples for a given topology configuration. So the results in this part reflect the overall performance of the algorithms.

We fix $\theta = 2.0$ for the EBSP algorithm in this part. How different θ values may affect the algorithm and the optimal θ values will be addressed next in Section 5.2.2.

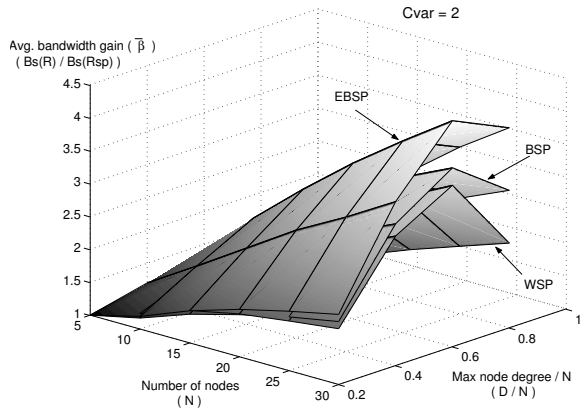


Figure 12: Average bandwidth gain ($\bar{\beta} = B_s(\mathcal{R})/B_s(\mathcal{R}_{sp})$) w.r.t. topology configurations ($C_{var} = 2$)

The figures show us some interesting and useful relations between performance of algorithms and different topology configurations.

General comparisons among WSP, BSP and EBSP: In general, the WSP is the best one among the three algorithms

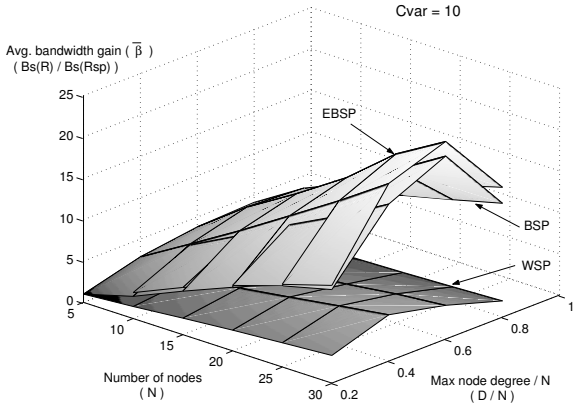


Figure 13: Average bandwidth gain ($\bar{\beta} = B_s(\mathcal{R})/B_s(\mathcal{R}_{sp})$) w.r.t. topology configurations ($C_{var} = 10$)

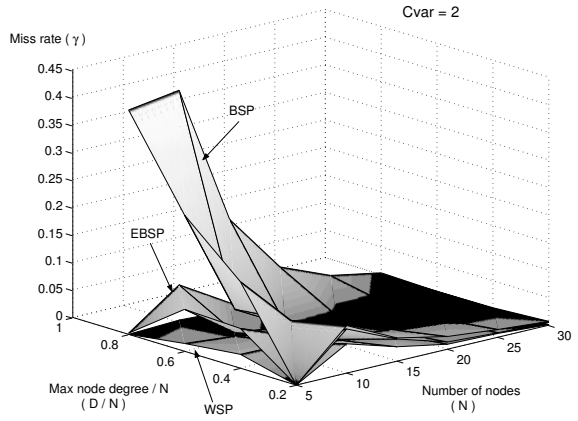


Figure 14: Miss rate (γ) w.r.t. different topology configurations ($C_{var} = 2$)

in terms of the miss rate and the BSP is the worst one (Figures 14 and 15), no matter what kind of network topologies are used. However, as far as the saturate bandwidth gain is concerned, WSP does not gain much generally (Figures 12 and 13). This is intuitively reasonable, because WSP is only a simple variant of the SP algorithm. For EBSP algorithm, it yields much larger values of bandwidth gain (in an order of magnitude better if compared with the results of WSP, especially when N and C_{var} are large) with small miss rates. The EBSP is always much better than the BSP in terms of the miss rate. For BSP, it yields fairly large bandwidth gains when N and C_{var} are large, but not as large as the EBSP does. Nevertheless, it always suffers from the highest miss rates among the three. We can observe in Figure 14 that, when $N = 10$, $D/N = 0.8$, the BSP has a miss rate as high as almost 0.45, meaning that in nearly half of the simulated topologies, it performs worse than the basic SP algorithm.

Comparison of the three algorithms in different topology configurations: If we compare the three algorithms when the capacity variance C_{var} is small, surprisingly, we find that WSP is the best in terms of both bandwidth gain and miss rate. However, the margin over EBSP is small. For large capacity

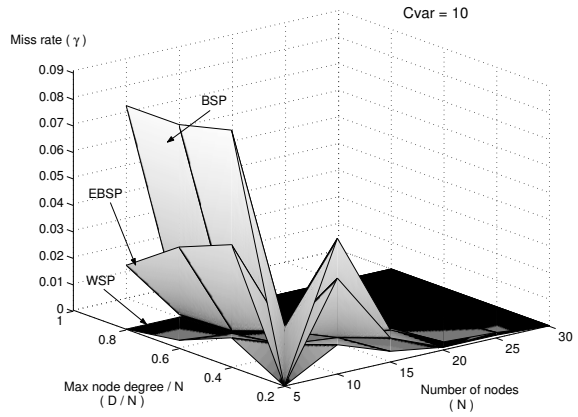


Figure 15: Miss rate (γ) w.r.t. different topology configurations ($C_{var} = 10$)

variances C_{var}, EBSP is definitely the best algorithm among the three - it not only achieves large saturate bandwidth values, but keeps fairly low miss rates as well, especially in case of large-scale, complex topologies. Therefore, we can draw a conclusion that both the WSP and EBSP are favorable for those networks with homogeneous links, meanwhile, the EBSP is more suitable for large-scale, complex networks with heterogeneous links. There is no significant advantage of BSP. Although it can yield comparable bandwidth gains as EBSP when N and D are small, it suffers from very high miss rates.

Performance trends w.r.t. C_{var}: Comparing Figures 12 and 13, as well as Figure 14 and 15, we find that when C_{var} increases from 2 to 10, the bandwidth gains of both BSP and EBSP increase significantly. In the meantime, their miss rates decrease with the increase of C_{var}. However, the performance of WSP does not change significantly with C_{var}.

Performance trends w.r.t. the maximum degree D: If we fix the value of N and C_{var}, by changing the values of D, we can observe the same trends for all three algorithms: when D increases, all three algorithms first yield larger saturate bandwidth values, but after a certain point, all of them start decreasing. Intuitively, the reason behind this phenomenon might be: at the beginning, when D is very small, the topology has very low connectivity, meaning that there are very few optional routes between nodes for the routing algorithms to choose. Therefore, all three algorithms yield almost the same results as SP does, hence small bandwidth gains. As D increases, there are more and more optional routes among nodes for the routing algorithms to choose. Thus, it is more and more likely for the algorithms to find better routes. As a result, the saturate bandwidth values increase. But after a certain turning point (different algorithms may have different turning points), the topology is getting so connected that the hop-count shortest paths become more preferable. Therefore, bandwidth gains start to decrease.

Brief conclusions: In summary, for a simple, homogeneous network, especially when the link capacity variance is small, EBSP or WSP should be used for the premium-class routing since they yield larger saturate bandwidth meanwhile keep miss rates low. However, for a complex, heterogeneous network, the EBSP algorithm is absolutely preferred since it achieves much higher saturate bandwidth gains as well as relatively low miss rates in such networks.

5.2.2 *Impacts of θ*

As we can see from the description of the EBSP algorithm, θ is an important constant. In the last subsection (Section 5.2.1), we fixed $\theta = 2.0$. In this part, we investigate its impacts on the performance in terms of the average bandwidth gain ($\bar{\beta}$).

Let $\bar{\beta}_{1.0}$, $\bar{\beta}_{2.0}$ and $\bar{\beta}_{\infty}$ denote the average bandwidth gains of EBSP when $\theta = 1.0, 2.0, \infty$, respectively. Based on the fact that $\bar{\beta}_{2.0} > \bar{\beta}_{1.0}$ and $\bar{\beta}_{2.0} > \bar{\beta}_{\infty}$,[9] for a given topology, intuitively, we project that there may exist an optimal θ, written θ^*, with which the EBSP algorithm achieves its maximal average bandwidth gain $\bar{\beta}^*$. To confirm this projection, we conduct another set of simulations and the results are shown in Figures 16 and 17.

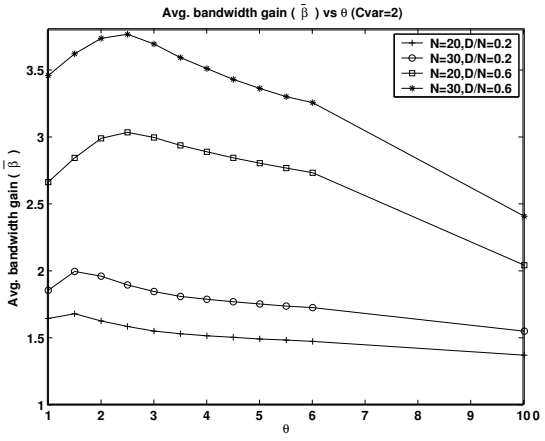


Figure 16: Average bandwidth gain w.r.t. θ ($C_{var} = 2$)

The figures show clearly that all curves of $\bar{\beta}$ versus θ have absolute maximum values, therefore, each topology configuration has an optimal θ^*. Notice that both figures are distorted a little bit. The reason is that the actual last point of each curve is for $\theta = 100$ (to mimic a very large θ). However, we put all last points in the figures at the position for $\theta = 10$ to make the details of curves clearer.

Furthermore, we conduct more simulations to obtain the optimal values of θ for different topology configurations. The results are demonstrated in Figure 18. Although the optimal values are scattered between 1.0 and 5.0, given topology configuration (N, D, C_{var}) ranges from $(5, 0.2, 2)$ to $(30, 0.8, 10)$, most of them are gathered between 1.5 and 3.0. This is exactly why we choose $\theta = 2.0$ in Section 5.2.1 when we evaluate the overall performance of EBSP. Figure 18 also reveals an overall trend between topology configuration and

[9] As we discussed before, $\theta = 1.0$ degrades EBSP to BSP and $\theta = \infty$ degrades EBSP to a variant of SP algorithm, both of which yield smaller $\bar{\beta}$ than when $\theta = 2.0$.

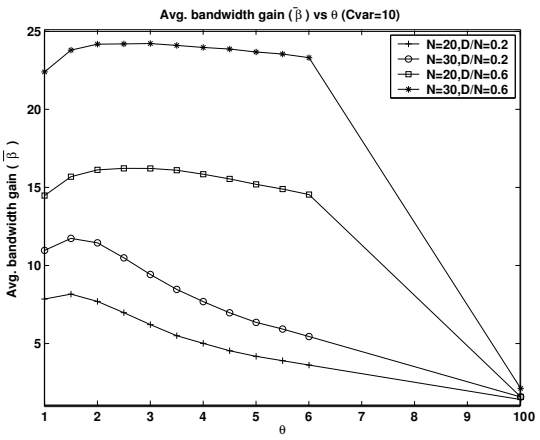


Figure 17: Average bandwidth gain w.r.t. θ ($C_{var} = 10$)

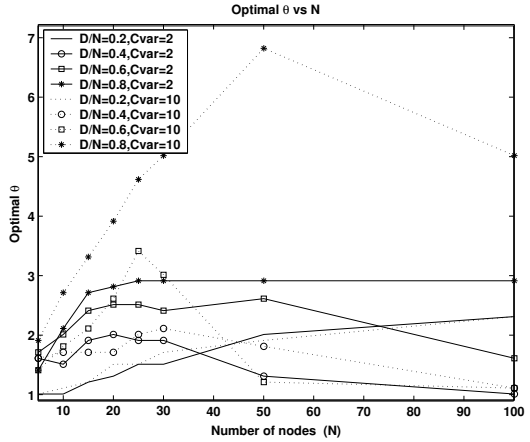


Figure 18: Optimal θ w.r.t. topology configurations

θ^*: topologies with larger number of nodes (N) or larger link capacity variance (C_{var}) tend to have larger optimal values of θ^*.

Brief conclusions: In summary, our simulation results observe an overall trend between topology configuration and θ^*: topologies with larger number of nodes (N) or larger link capacity variance (C_{var}) tend to have larger optimal values of θ^*. However, for medium-sized networks (e.g., $N = 20$ or 30), the values of θ^* are mostly gathered between 1.5 and 3.0, thus we can use $\theta = 2.0$ to evaluate the overall performance of EBSP in such networks. The performance of EBSP in large scale networks will be studied later in Section 5.2.3.

5.2.3 Special Cases with Larger Number of Nodes

In the previous parts, since we focused on average performance which was based on a large number of topologies (2000 for each topology configuration) to eliminate potential biases, we could not simulate very large scale networks because of high computation complexity (the maximum number of nodes we simulated was 100). Moreover, for each given topology configuration, the topologies were totally randomly generated. Some of them might not be realistic.

Topology Type		1 level router (IP) only
Topology Parameters	**Model Type**	Waxman: $\alpha_w = 0.15, \beta_w = 0.2$
	Network Size (HS)	1000 (number of squares)
	# of Nodes (N)	200, 300, 400, 500, respectively
	Node Degree (m)	20
	Node Placement	random
	Bandwidth Distr.	uniform
	Min. bandwidth	10
	Max. bandwidth	1024

Table 1: BRITE topology generator configuration

# of Nodes (N)	**Bandwidth Gain (β)**				
	WSP	**BSP**	**EBSP** ($\theta = 1.5$)	**EBSP** ($\theta = 2.0$)	**EBSP** ($\theta = 3.0$)
200	13.41	33.94	37.86	42.79	43.75
300	7.65	29.13	33.38	38.77	40.37
400	6.58	28.26	31.16	33.39	36.39
500	10.42	45.79	50.01	55.23	58.75

Table 2: Bandwidth gains (β) in large realistic networks

In this part, in order to further investigate the algorithms in larger and more realistic networks, we use a famous topology generator, BRITE[10], to generate more realistic topologies with larger number of nodes (up to 500 nodes), in which we compare the performance of the three algorithms.

We configure the BRITE as shown in Table 1 to generate our simulating topologies, where HS means "size of the main plane" [19] and it simulates the geometrical size of a network. Waxman model is used. Most configurations in the table (e.g., for the Waxman model, $\alpha_w = 0.15$ and $\beta_w = 0.2$) are default values. Totally, 4 topologies are generated and they consist of 200, 300, 400, and 500 nodes, respectively.

Table 2 shows the detailed simulation results. It is clearly demonstrated that all three algorithms, WSP, BSP, and EBSP, perform better than the SP algorithm in all 4 cases (i.e., the bandwidth gain, β, is always larger than 1). It is also shown that the EBSP algorithm outperforms WSP and BSP in all cases, no matter what value is used for θ (1.5, 2.0, or 3.0). Furthermore, for the EBSP algorithm itself, the results show that a larger scale network may favor a larger θ. That is, if the node degree is fixed, the more nodes a network consists of, the larger the optimal value θ^* will be. This conforms to the results in the previous part (Figure 18).

6. RELATED WORK

Extensive research has been conducted on QoS routing issues recently. In [3], S. Chen and K. Nahrstedt did a thorough survey on QoS routing algorithms, where routing problems were classified into four categories in terms of types of constraints and optimization targets. However, if we use a different criterion, routing mechanisms can also be categorized into flow-based routing and hop-by-hop routing. Most of existing QoS routing solutions focus on virtual circuit network models which are connection-based or flow-based. In [15, 16], the authors proposed and evaluated some new

[10]BRITE is developed by Alberto Medina et al at Boston University.[19]

Dijkstra-based QoS routing algorithms, such as the *shortest-distance path* algorithm which uses the reciprocal of link bandwidth as weight function, similar to the BSP algorithm in [26] and in this paper. In [12], the authors proposed the Minimum Interference Routing algorithm (MIRA) which routes a newly connection onto a path that does not interfere too much with those *critical links*. The algorithm shows good performance compared to other algorithms. In [23], the authors addressed the QoS routing from the precomputation perspective and proposed a hierarchical algorithm to solve the *All-Hops Problem*. In [1], the authors discussed path selection algorithms to support QoS routes in the context of extensions to the OSPF protocol. In [4, 28], the authors studied the *ticket-based routing algorithms* in an environment where state information was imprecise. All research work above was flow-based or connection-based and did not consider hop-by-hop routing constraints, where routing consistency and forwarding loop-freedom should be guaranteed.

In the hop-by-hop routing category, Van Mieghem, De Neve and F. Kuipers proposed the Tunable Accuracy Multiple Constraints Routing Algorithm (TAMCRA) to address QoS routing problems with multiple constraints in a hop-by-hop manner [20]. The authors proved that the TAMCRA is an exact QoS routing algorithm which can guarantee loop-freeness. The major differences between [20] and our paper include: (1) TAMCRA is not a consistent routing algorithm; and (2) TAMCRA solves the routing problem for one source-destination pair, i.e., given a source and destination, TAMCRA finds a sub-optimal route from the source to the destination; while our algorithms try to approach an optimal solution for the entire network, given multiple sources and destinations. Some basic issues on hop-by-hop routing algorithms, such as consistency, isotonicity, and search of optimal paths, were studied in [25]. The author provided an elegant algebra basis to study the correctness of Dijkstra-based QoS routing algorithms in the context of hop-by-hop routing in the Internet. We borrowed some definitions and results from his work.

Another related research area is the Type-of-Service (TOS) routing. Matta and Shankar proposed the TOS2 routing scheme in [18], where two TOS's were considered: *low delay* and *high throughput*. Two additive link cost functions were given accordingly. Dijkstra's algorithm was used. But the main focus of the paper was not routing algorithms, instead, the authors thoroughly analyzed the network stability and convergence features by using the Liapunov function method. In [24], the authors classified flows into *long-lived* and *short-lived* two types and proposed a hybrid routing scheme for them. Briefly, the paper suggested that dynamic routing should be used for the long-lived flows, while static preprovisioned paths should be used for the short-lived ones. However, their approach was basically in a flow-based manner, thus each long-lived flow could be routed individually and dynamically in the residual network to achieve high stability and load balance.

In [26], we raised the OPR problem for the first time. The idea of finding a solution to the OPR problem is somewhat similar to the idea of *Max-min routing*, which is also NP-hard and has been studied in [2, 8, 10, 17]. Although both problems are dealing with unsplittable "flows", a solution to the Max-min routing problem does not have to maintain routing consistency, i.e., each flow can be routed independently. Therefore, their results can not simply be applied to the OPR problem.

In this paper, we introduced two existing hop-by-hop QoS routing algorithms and proposed a new one (EBSP - Enhanced Bandwidth-inversion Shortest Path). All of them are related to the generalized Dijkstra's algorithm. We also provide rigorous proofs for the NP-hardness of the OPR problem and for the correctness of the novelly-proposed EBSP algorithm. The performance of EBSP, as well as the impact of EBSP's discount factor θ, was studied through simulations.

7. CONCLUSION

In order to service premium traffic efficiently while keeping its negative inter-class effects low, the premium class routing algorithm must be carefully chosen. In this paper, we argued that the choice of the optimal premium class routing algorithm leads to the Optimal Premium-class Routing (OPR) problem, which is NP-hard. We have briefly examined two existing routing algorithms (the Widest-Shortest-Path (WSP) algorithm and the Bandwidth-inversion Shortest-Path (BSP) algorithm) and designed the novel Enhanced Bandwidth-inversion Shortest-Path (EBSP) algorithm to compare and study the tradeoffs of these different solutions for the OPR problem. We also provided a rigorous proof for the correctness of the EBSP algorithm. Our simulation results showed that on average all three routing algorithms outperformed the basic hop-count shortest-path (SP) algorithm. Furthermore, our new EBSP algorithm significantly outperformed all other existing algorithms in large-scale, heterogeneous networks. As the DiffServ networks are becoming more and more heterogeneous and complex, our results strongly indicate that the EBSP routing algorithm is a strong candidate for routing of premium class traffic in DiffServ networks.

In this paper, we assume that each node requires the same premium bandwidth to other nodes. This is a very strong assumption. In our future work, we will address a more general OPR problem where the premium bandwidth demand can be heterogeneous. The existing EBSP algorithm's performance will be examined in the new environment. Possible enhancements will also be studied.

8. ACKNOWLEDGMENTS

The authors would like to acknowledge the help of Li Xiao and King-Shan Lui for their invaluable discussions and suggestions. The authors would also like to thank the anonymous reviewers for their detailed and insightful comments.

9. REFERENCES

[1] G. Apostolopoulos, D. Williams, S. Kamat, R. Guerin, A. Orda, and T. Przygienda. QoS Routing Mechanisms and OSPF Extensions. RFC 2676, Aug. 1999.

[2] D. Bertsekas and R. Gallager. *Data Networks*. Prentice Hall, 1990.

[3] S. Chen and K. Nahrstedt. An Overview of Quality-of-Service Routing for the Next Generation High-Speed Networks: Problems and Solutions. *IEEE Network, Special Issue on Transmission and Distribution of Digital Video*, Nov./Dec. 1998.

[4] S. Chen and K. Nahrstedt. Distributed QoS Routing with Imprecise State Information. In *Proceedings of 7th IEEE International Conference on Computer, Communications and Networks (ICCCN'98)*, pages 614–621, Lafayette, LA, October 1998.

[5] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. The MIT Press, Cambridge, Massachusetts, 1990.

[6] S. et. al. An Architecture for Differentiated Services. *RFC 2475*, December 1998.

[7] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman, San Francisco, CA, 1979.

[8] J. Jaffe. Bottleneck flow control. *IEEE Transactions on Communication*, 29:954–962, 1981.

[9] J. Kleinberg. Single-source Unsplittable Flow. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, pages 68–77, October 1996.

[10] J. Kleinberg, Y. Rabani, and E. Tardos. Fairness in routing and load balancing. In *40th Annual Symposium on Foundations of Computer Science*, pages 568 – 578, 1999.

[11] K.Nichols, V.Jacobson, and L.Zhang. A Two-bit Differentiated Services Architecture for the Internet. *RFC 2638*, July 1999.

[12] M. Kodialam and T. Lakshman. Minimum Interference Routing with Applications to MPLS Traffic Engineering. In *Proceedings of IEEE Infocom 2000*, March 2000.

[13] S. Kolliopoulos and C. Stein. Improved Approximation Algorithms for Unsplittable Flow Problems. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science*, 1997.

[14] J. Lenstra, D. Shmoys, and E. Tardos. Approximation algorithms for scheduling unrelated parallel machines. In *28th IEEE FOCS*, 1987.

[15] Q. Ma and P. Steenkiste. On path selection for traffic with bandwidth guarantees. In *5th IEEE International Conference on Network Protocols, Atlanta, GA*, pages 191 – 202, October 1997.

[16] Q. Ma and P. Steenkiste. Supporting Dynamic Inter-Class Resource Sharing: A Multi-class QoS Routing Algorithm. In *IEEE Proceedings of INFOCOM '99*, pages 649–660, 1999.

[17] Q. Ma, P. Steenkiste, and H. Zhang. Routing high-bandwidth traffic in max-min fair share networks. In *Proceedings of ACM SIGCOMM'96*, pages 206–217, Palo Alto, CA, Oct 1996.

[18] I. Matta and A. U. Shankar. Type-of-Service Routing in Datagram Delivery Systems. *IEEE Journal of Selected Areas in Communications*, 13(8), October 1995.

[19] A. Medina, A. Lakhina, I. Matta, and J. Byers. BRITE: An Approach to Universal Topology Generation. In *Proceedings of the International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunications Systems-MASCOTS '01*, Cincinnati, Ohio, August 2001.

[20] P. V. Mieghem, H. D. Neve, and F. Kuipers. Hop-by-hop Quality of Service Routing. *Computer Networks*, 37(3-4):407–423, 2001.

[21] M.May, J.C.Bolot, C.Diot, and A.Jean-Marie. Simple Performance Models of Differentiated Services Schemes for the Internet. In *Proceedings of IEEE Infocom 1999*, March 1999.

[22] A. Orda. Routing with end-to-end QoS guarantees in broadband networks. *IEEE/ACM Transactions on Networking*, 7(3), June 1999.

[23] A. Orda and A. Sprintson. QoS Routing: the Precomputation Perspective. In *Proceedings of IEEE Infocom 2000*, March 2000.

[24] A. Shaikh, J. Rexford, and K. Shin. Load-Sensitive Routing of Long-Lived IP Flows. In *Proceedings of ACM SIGCOMM'99*, 1999.

[25] J. Sobrinho. Algebra and algorithms for QoS path computation and hop-by-hop routing in the Internet. In *IEEE INFOCOM 2001, Anchorage, Alaska*, pages 727 – 735, April 2001.

[26] J. Wang and K. Nahrstedt. Hop-by-Hop Routing Algorithms For Premium-class Traffic In DiffServ Networks. In *Proceedings of IEEE Infocom 2002*, June 2002.

[27] J. Wang, Y. Wang, and K. Nahrstedt. Quantitative Study of Differentiated Service Model Using UltraSAN. *Tech. Report UIUCDCS-R-2001-2237, Department of Computer Science, University of Illinois at Urbana-Champaign*, July 2001.

[28] L. Xiao, J. Wang, and K. Nahrstedt. The Enhanced Ticket-based Routing Algorithm. In *Proceedings of IEEE ICC 2002*, New York, NY USA, April 28 - May 2 2002.

Crossover Scaling Effects in Aggregated TCP Traffic with Congestion Losses*

Michael Liljenstam
Institute for Security Technology Studies
Dartmouth College
45 Lyme Rd., Suite 200
Hanover, NH 03755
mili@ists.dartmouth.edu

Andy T. Ogielski
Renesys Corporation
Hanover, NH 03755
ato@renesys.com

ABSTRACT

We critically examine the claims that TCP congestion control contributes to the observed self-similar traffic rate correlations. A simulation model is designed to analyze aggregated traffic of many TCP file transfers, with network topologies large enough so that each transfer has independent packet losses due to competition with other TCP traffic. To separate the effects of session-level variability from network-level variability we examine traffic consisting of small fixed-size files, and of heavy-tailed distribution of file sizes, with small variance of inter-session periods.

We find that, with increasing packet loss rate, traffic rate scaling crosses over from the regime dominated by file size distribution to another scaling regime that is independent of file sizes. That loss-dominated scaling stretches over the timescales from RTT to the longest consecutive TCP timeouts (hundreds of seconds), and is not asymptotic. Analysis at the flow level exposes the mechanism of the crossover, from scaling dominated by variability of the flow ON-periods to that dominated by variability of the OFF-periods.

However, it is unlikely that TCP timeouts contribute much to observed Internet traffic correlation structure, as they would matter only if widespread congestion losses exceeding 10% dominated the typical behavior of the Internet.

Keywords

Self-similarity, TCP

1. INTRODUCTION

Explanation of ubiquitous observations of packet traffic self-similarity on a single link [14, 20] in terms of high variability of data transfer sizes at the session layer has been well supported by network measurements (LAN traffic [30], WAN traffic [20], and WWW traffic [5]).

However, recently there have been claims that congestion losses in the TCP sessions constituting the largest fraction of the Internet traffic could be responsible for self-similarity [26], or for scaling effects that can be confused with self-similarity [9, 12] on the time scales reported in measurements.

Veres and Boda [26] claimed on the basis of simulations of a few TCP connections competing in a single link that heavy losses induce chaotic behavior and self-similarity in individual connections, and that this contributes to self-similar traffic scaling at a more fundamental level than session-layer data size distributions. Guo et al. [12] and Figueiredo et al. [9] followed up on this observation by analyzing an approximate mathematical model of an infinitely long TCP connection with an externally imposed Bernoulli packet loss process. These authors confirmed the calculations with simulations of a single connection using a detailed TCP Tahoe model. They correctly recognized that loss-induced correlations cannot be asymptotically self-similar because TCP timeouts are bounded. Nevertheless, Figueiredo et al. also claimed that the time range of what they called "pseudo-self-similar" scaling (cf. [15]) [1] can be comparable to the range of scaling observed in traffic measurements, and thus TCP loss effects can be a plausible explanation for observed self-similarity.

These are interesting claims, challenging an established explanation, and they deserve evaluation under more realistic conditions: It has to be noted that they are based on studies of a *single* connection subjected to a rather artificial loss process. First, in a network consisting only of a single link [26], the behaviors of TCP connections are very highly correlated with one another, and exhibit peculiar periodic phase effects [11, 13]. Second, the assumption [12, 9] of constant probability, independent packet loss is lacking an experimental justification (see [19]). In real networks the aggregate traffic on any link mixes packets from connections that may have independently suffered from losses somewhere else along their paths, queueing losses are bursty, and most connections are rather short.

In this paper we present the analysis of a simulation experiment that has been carefully designed to compare the evolution of TCP traffic rate scaling behavior with increas-

*This work has been partially supported by the Defense Advanced Research Projects Agency (DARPA), under grant N66001-00-8065 from the U.S. Department of Defense. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the Department of Defense.

[1] "Pseudo-self-similarity" is not well defined: a Hurst parameter estimate obtained over a limited segment of a variance-time or wavelet plot can be inconsistent with the stationarity condition [29]

ing congestion losses, at the two extreme session-layer traffic scenarios: one with small, constant-size file transfers, another with heavy-tailed, Pareto file size distributions. We designed a network model suited to study aggregate TCP traffic, where the following requirements are satisfied: First, each TCP connection in the aggregate is subject to losses independently of others, at some other network location, and the losses are caused by a different set of competing TCP connections rather than artificially imposed from the outside. Second, packet loss rate can be controlled by changing the number of competing TCP connections. Third, the effects of TCP timeouts on the scaling behavior can be validated in a control experiment by varying the maximum timeout duration in the TCP implementation.

We find that, with increasing packet loss rate, traffic rate scaling crosses over from the regime dominated by file size distribution to a scaling regime that is dominated by TCP timeouts, and is independent of file size distribution. That loss-dominated scaling stretches over the timescales from the round-trip time (RTT) to the longest sequence of 12 consecutive TCP timeouts before the connection is dropped. In the heavy loss regime the range of loss-dominated scaling is quite large, reaching times on the order of hundreds of seconds. We expose the mechanism of the crossover behavior by traffic analysis at the flow level, showing that the scaling crossover is due to the change of the dominant contribution to traffic variability, from variability of the flow ON-periods to variability of the inter-flow OFF-periods.

The quantitative analysis shows that it is unlikely that TCP timeouts in the Internet can contribute much to observed self-similar Internet traffic correlations: that would require widespread, long-lasting congestion losses exceeding 10%. This is not the usual operating regime of existing networks.

The paper is organized as follows: In the following section we describe the experiment design, that allows to compare non-variable to highly variable traffic processes under varying loss conditions. This is followed by the presentation of the traffic rate scaling under such conditions. The changing mechanism of scaling is explained in the sections on the analysis of IP flow statistics.

We have made use of some fundamental definitions and theorems from the following sources: *self-similarity* [14], *heavy-tailed distributions* [16], and an ON/OFF source superposition model for *Fractional Brownian Motion* [30]. We also use the following statistical tools: scaling analysis using *Variance-Time (VT) plots* [14], *Multi-Resolution Analysis (MRA) based on wavelets* [1], and the *Rescaled adjusted range (R/S) statistic* [2]. Finally, we detect heavy-tailed distributions using Log-Log plots of the *Complementary Cumulative Distribution Function (CCDF)* [5]. Details of the experimental design can be found in the Appendix.

2. EXPERIMENT DESIGN

Network topology: In order to study aggregate TCP traffic with uncorrelated losses we must design network models that are more complex than one-link "dumbbell" topologies commonly used in studies of TCP dynamics. A parsimonious design satifying the requirements of our experiments is shown in Figure 1.

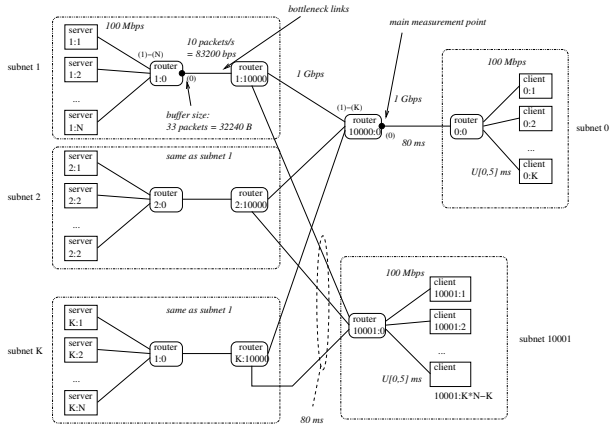


Figure 1: A family of network topologies used in simulation experiments described in this paper. K server networks, each with N server hosts, are shown on the left. The bottleneck links are between the two routers in the server networks. Two client networks are shown on the right. The client-server TCP connections are established so that in each server network exactly one server has one connection with one client in the upper right, while other servers' connections are with clients in the lower right. In this design, the TCP connections through the router labeled 10000:0 have independent losses.

The two building blocks are a server network and a client network. Each client host requests a file from one server host. We multiplex independent TCP connections on a link between the routers 10000:0 and 0:0 as follows[2]: For each of the K server networks, a single connection is established between one server and one host in client network 0 (upper right in the diagram), while the remaining $N-1$ servers have connections with clients in the network 10001 (lower right in the diagram). Thus, K connections always follow the path through the router 10000; and if the congestion losses are on the bottleneck links between two routers in each server network, these K connections will have independent losses.

The network is dimensioned so that for $N = 1$ a single TCP connection can flow without losses through a bottleneck link. The congestion loss rate can be systematically increased by increasing N: as more connections are added, there will be competition between them in the bottleneck link. We use the TCP Tahoe version because it was used in the prior work [26, 9, 12] that is addressed here. Details on the values of network parameters and the TCP parameters are listed in the Appendix A.

We measure packet statistics on the network interface (NIC) number 0 on router 1:0 to monitor the packet loss probability on the bottleneck links. The losses in the other server networks are similar due to symmetry. We measure the ag-

[2]We use the SSFNet's "Network, Host, Interface" topological naming convention for network elements. Thus a two-level hierarchy of networks, where network N_1 contains a subnetwork N_2 is written as $N_1 : N_2$; a host H in network N_2 is referred to as $N_1 : N_2 : H$; and a network interface I on host H is specified as $N_1 : N_2 : H(I)$.

gregate traffic at the NIC 0 on router 10000:0. At each measurement NIC we log packets arriving at the output queue *before* packet losses are counted.

The network models are constructed and simulated using the SSFNet simulator release 1.3 (source code, validation tests[3], and manuals are available at `http://www.ssfnet.org`).

Traffic models: Three traffic cases are studied. In each case every client repeatedly requests file transfers with a random, exponentially distributed wait time between the completion of one transfer and the start of the next one. Thus, the client-server traffic process is an alternating renewal ON/OFF-process [4] with exponentially distributed OFF-periods. The ON-period distributions are as follows:

repeated fixed size file transfers File sizes are fixed to 12000 bytes (12 packets).

heavy-tailed file sizes, modest tail weight File sizes are drawn from a Pareto distribution (same mean as for fixed size), and the shape parameter $\alpha = 1.8$ (close to 2 so not given much weight).

heavy-tailed file sizes, high tail weight File sizes are Pareto distributed with the same mean, and the shape parameter $\alpha = 1.2$.

With such an experiment design we can compare the contributions to aggregate traffic rate correlations due to network layer variability (congestion) and to session layer variability (file transfer renewal processes).

We note that in previous analytic studies [9, 12] of the loss-induced scaling behavior of a *single* TCP connection it was assumed that packet losses are independent, i.e., packet losses follow a Bernoulli process. However, we found that when packet losses are due to competition among multiple TCP connections in a congested queue, then for each connection the consecutive losses are definitely *not* independent. We performed the χ^2 tests for independence of two consecutive packet drop decisions (for a single connection) in the simulation experiments, and we found that the independence hypothesis is false at the 1% significance level, basically meaning 99% certainty in the conclusion.

Statistical significance: Mathematically speaking, self-similarity of a superposition of K ON/OFF alternating renewal processes is an asymptotic phenomenon, reached as first K and then the time scale approach infinity, in this order [23, 27], when either ON, or OFF, or both periods have a heavy-tailed distribution.

In practical experiments, this limit is gradually approached, with both K and the measurement time affecting the variability of the superposition and the range of time scales manifesting an approximate power law behavior. Since the corrections to scaling for finite K and finite times are not known in analytic form, in this work we have been gradually increasing K and the simulated time until sufficiently robust statistics were obtained. See Section 6 for more details.

To obtain sufficiently robust traffic variability in finite simulations with Pareto-distributed ON periods, we consider the interplay of two factors: the mean of the exponential OFF period distribution, and the number K of multiplexed TCP connections. It is a simple observation that if the mean is larger, one needs a larger K to achieve sufficiently high traffic rate variablity in a finite time interval.

In turn, degree of traffic variability influences the duration of simulated time and the number of independent experiments required for good statistics: statistics involving long-range dependent processes and heavy-tailed distributions converge very slowly with time and the number of samples [6, 24]. Therefore, both K and the length of simulated time must be large enough in order to achieve sufficiently small sampling errors in the wavelet, VT and R/S plots and to expose the scaling properties over several decades of time. Note that the required simulated time length depends on the RTT and bandwidth in the network model, because the number of sampled ON-periods increases with the number of TCP connections.

The remaining two concerns are: *i)* the initial start-up transient, and *ii)* that traffic load does not decrease in time due to dropped connections. We verified that the initial traffic transient is so short that it has no effect on the results. The simulated client implements error recovery: After connection drop due to 12 consecutive retransmission attempts or a timed-out connection attempt, the client requests a new file transfer after an exponentially-distributed inter-transfer wait time. Therefore, even with connection failures the traffic load is constant throughout the experiment.

In the data presented in this paper the values of K, exponential mean, and the simulated time range were chosen so that the crossover of scaling becomes clearly visible, and its mechanism can be unambiguously identified. For $K = 10$, and the OFF-period mean of 1.1 seconds the experiments running for 100,000 seconds (over 27 hours) of *simulated time* produce acceptable statistics for the time scales spanning three to four decades in time, from 1 to 10^3 or 10^4 seconds. The adequacy of our choice of the K parameter and other issues related to parameter sensitivity will be discussed in more detail in section 6.

3. SCALING ANALYSIS

We study the scaling behaviour of the packet rate process for each traffic scenario and loss rate, with the smallest packet count bin size of 100 ms. Figure 2 summarizes the results in terms of the wavelet energy plots and variance-time plots. All wavelet plots use the same Haar wavelet (D1) base to facilitate comparability of qualitative aspects. Other bases were tried but led to only small differences in estimates for reasonable fits. The R/S plots were also generated, but are not shown.

The Hurst parameter estimates $\hat{H}$ obtained using the three distinct methods have been summarized in Table 2. The table also shows the theoretical value of the Hurst parameter for a corresponding Fractional Brownian Motion (FBM)

[3]In particular, the SSFNet TCP implementation validation includes the ns-2 test suite [10], and testing with a version of the *tcpanaly* tool [18].

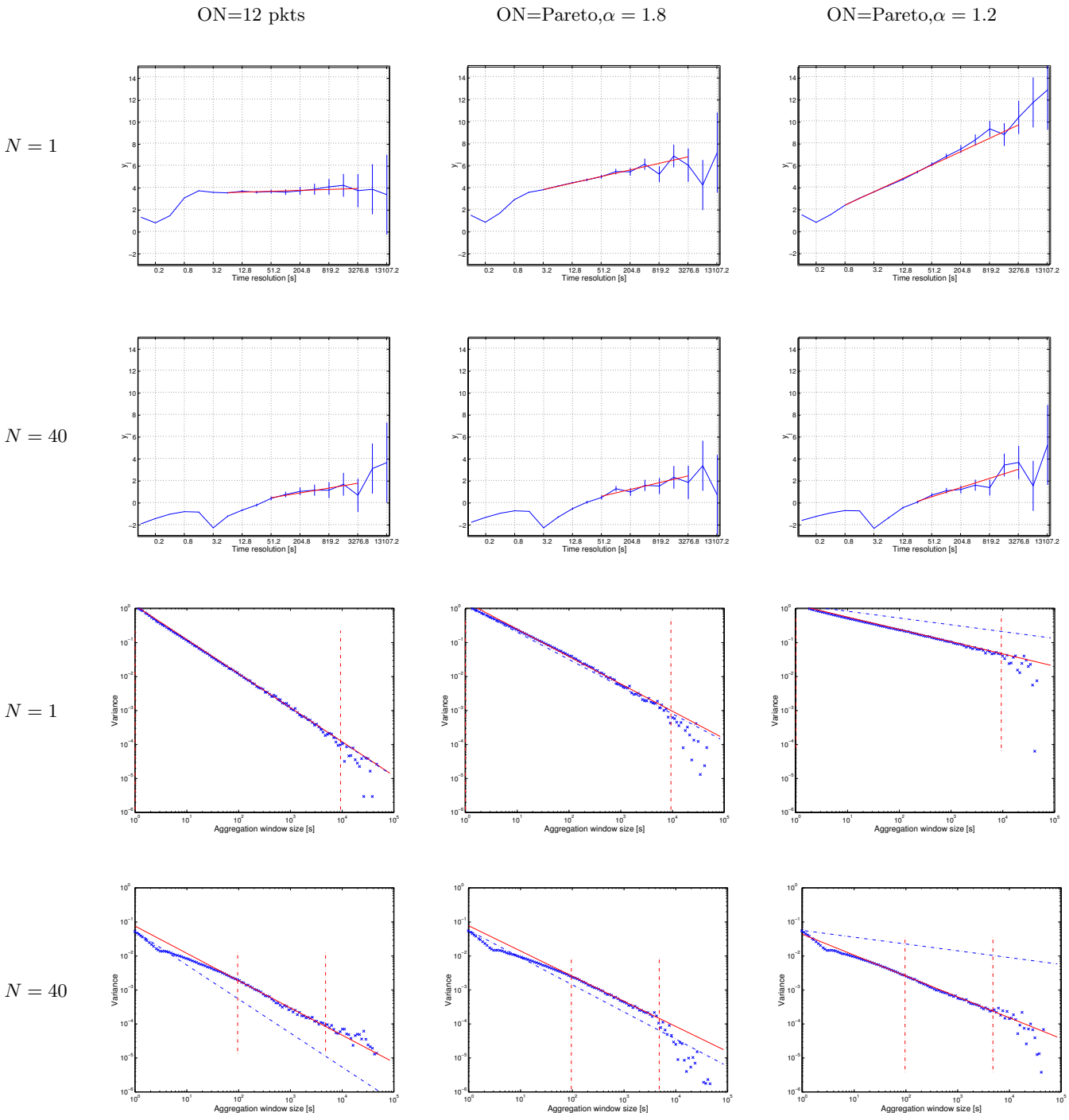


Figure 2: **Wavelet and variance-time (VT) scaling plots of the aggregate traffic rate ($K = 10$). Columns, from left to right: small fixed sized transfers (12 packets long), Pareto distributed file sizes with $\alpha = 1.8$, and Pareto with $\alpha = 1.2$. Rows labeled $N = 1$ correspond to aggregate traffic of K TCP connections without any packet losses, while rows labeled $N = 40$ correspond to aggregate traffic where each connection is suffering from packet losses exceeding 10% (see Table 1). Initial bin width is 100 ms. The slopes of the dashed lines in the VT plots follow the power law for the corresponding idealized FBM processes.**

Scenario		*Router 1:0(0) statistics*			*Router 10000:0(0)*		*Global statistics*		
File size	Server Hosts (N)	Packets	Dropped Packets	P[drop] estimate	Packets	Netflows	TCP connections attempted	failed TCP connection attempts	dropped TCP connections
fixed	1	523183	0	0	5234381	140081	348962	0	0
size	40	1477604	219944	0.149	313250	180040	808866	0	1
Pareto	1	539782	0	0	5405933	139885	349367	0	0
$\alpha = 1.8$	40	1527169	224393	0.147	324659	184304	814319	0	1
Pareto	1	525673	0	0	5247843	147961	369202	0	0
$\alpha = 1.2$	40	1562748	207911	0.133	328751	188043	937462	0	1

Table 1: Packet loss, netflow, and connection statistics for each traffic model.

process, where H is related to the shape of the heavy-tailed ON distribution by $H = (3 - \alpha)/2$.

For lossless traffic cases ($N = 1$) the wavelet plots in Figure 2 behave as expected: a flat plot for fixed-size file transfers, and increasing slopes for Pareto transfers with increasingly heavy tails. The dip at a fine scale (0.2 s for $N = 1$) is consistent with the presence of a periodic traffic component about the RTT value [8]. Note that the dip shifts to the right as traffic load increases ($N = 40$) as expected for an increase in RTT due to queuing delay: In this model, when packets encounter a full queue most of the time the RTT is about 3.3 seconds.

The Hurst parameter estimates for lossless traffic are close to theoretical values for the corresponding FBM processes. However, in the case of Pareto file size distribution with $\alpha = 1.2$ the estimate $\hat{H} \approx 0.8$ is smaller than the expected theoretical value $H = 0.9$. This negative bias can be attributed to undersampling of the heavy tail, even with simulation time of 10^5 seconds, as long range correlations are produced by rare instances of extremely large file transfers [24]. Nevertheless, this bias does not affect the analysis of the effects of TCP timeouts on the scaling properties.

Having established the baseline behavior of the aggregate traffic of lossless TCP connections, we proceed to analyze the effects of heavy packet losses. Recall that increasing packet losses are generated by increasing the number N of competing connections in the server network. Having verified the evolution of new scaling behavior with gradually increasing N, in this paper we only present the results for $N = 40$ which corresponds to packet loss rates of 13–14% (Table 1).

The most striking feature of the wavelet plots in Fig. 2 for an aggregate traffic of TCP connections with packet losses is that they become qualitatively similar for *both* fixed-size and Pareto distributions. Clearly, some loss-related mechanism affects the short–to–intermediate range scaling behavior, regardless of the file size distribution. Moreover, the same mechanism depresses long-range correlations for the heavy-tailed (especially $\alpha = 1.2$) Pareto distribution, as if the long file transfers became significantly rarer than in the lossless case. We also experimented with lower loads resulting in lower loss rates (results not shown here) but found that the scaling features found for fixed-sized transfers only become pronounced for high loss rates, typically above 10%.

Scenario		*Estimates*			*Th.*
File size	N	$\hat{H}$ (wavelet)	$\hat{H}$ (VT)	$\hat{H}$ (R/S)	H
fixed	1	0.52 [0.50,0.54]	0.50	0.55	0.5
size	40	0.61 [0.56,0.66]	0.61	0.60	
Pareto	1	0.65 [0.64,0.66]	0.60	0.62	0.6
$\alpha = 1.8$	40	0.65 [0.60,0.70]	0.63	0.64	
Pareto	1	0.80 [0.80,0.81]	0.82	0.78	0.9
$\alpha = 1.2$	40	0.71 [0.67,0.74]	0.70	0.85	

Table 2: Hurst parameter (H) estimates for the three distinct traffic processes, and theoretical values for the corresponding idealized Fractional Brownian Motion processes.

In the following sections we explain the structural mechanism of this phenomenon in terms of flow fragmentation by TCP timeouts.

4. TCP TIMEOUTS AND SCALING CROSSOVER

In this section we demonstrate that the range of TCP-induced scaling in the heavy packet loss regime is determined by the longest "gaps" in the TCP transmission, that are due to the accumulation of TCP timeouts during consecutive packet losses.

In order to determine how the maximum timeout duration relates to the range of loss-induced scaling we experiment with modifications of the timeout computation in the TCP implementation. Recall that in BSD-style TCP implementations the timeout value is calculated as $RTO \cdot 2^k$, where $0 \leq k \leq 6$, and then is restricted to be between 1 and 64 seconds. Therefore, before a connection is dropped due to 12 consecutive packet losses, the longest transmission gap is at least 383 seconds. Note that with Karn's algorithm, two retransmissions are considered consecutive if there was no RTT calculation between them. In other words, there are two consecutive retransmissions if a segment was retransmitted, and either the same segment is retransmitted again, or the next segment is retransmitted. Thus a case of, say, 11 consecutive retransmissions may occur in multiple ways, e.g., when the same segment is retransmitted 11 times, or when 11 consecutive segments are sent and each of them is retransmitted once, and so on.

It is desirable to reduce the statistical errors in the scal-

	$\hat{H}$		
	orig t.o. factor=64	max t.o. factor=8	max t.o. factor=2
Wavelet plot	0.61 [0.56,0.66]	0.70 [0.68,0.72]	0.53 [0.48,0.58]
VT plot	0.61	0.54	0.51
R/S plot	0.60	0.67	0.62

Table 3: Hurst parameter estimates $\hat{H}$ for TCP with reduced timeouts, $N = 40$.

ing plots at the scale corresponding to the estimated location of the crossover from loss-induced to asymptotic scaling. Therefore, we designed simulation experiments where the maximum value of the timeout factor was reduced from $2^6 = 64$ to 8 and 2, respectively. Figure 3 shows the scaling plots obtained with these modifications, and the unmodified scaling plot for comparison, for the repeated fixed-size file transfer traffic scenario. Table 3 gives the Hurst parameter estimates.

For a maximum timeout factor of 8 the VT plot shows a broad "knee" at the scale of about 100 seconds, and for a maximum timeout factor of 2, the location of the "knee" is at the scale of about 30 seconds. The "knee" in a VT plot demonstrates the crossover from short-time to asymptotic scaling regime. The corresponding feature in the wavelet plots is the onset of the asymptotic zero-slope regime. Thus, data indicates that the range of TCP timeouts is indeed responsible for the observed range of loss-induced scaling. For packet loss rate over 10% the crossover region is approximately at the time scale corresponding to the maximum duration of accumulated consecutive timeouts, which for standard TCP implementations and typical RTT values would be about 400 seconds.

5. FLOW STATISTICS AND MECHANISM OF CROSSOVER

A network *flow* on a link is defined as a sequence of IP packets traveling from a given source to a given destination, where arrivals of successive packets are separated by the time interval smaller than a predefined *flow threshold* Δt_f. In the model discussed here the source and destination are uniquely identified by the IP addresses of the corresponding hosts. Note that a flow is intrinsically defined relative to the time scale Δt_f, and the characterization of flow-level behavior needs to be reasonably robust with respect to the choice of Δt_f.

In this section we discuss the distributions of flow durations (ON intervals) and of inter-flow gaps (OFF intervals) for the three traffic scenarios, both without and with packet losses. As in previous sections, we consider side by side the cases of aggregate traffic without packet losses ($N = 1$), and with heavy packet losses ($N = 40$). Figure 4 shows the log-log plots of the complementary cumulative distribution functions (CCDF) for the ON-intervals.

Without packet losses, and at longer time scales, we see that for each traffic scenario the distribution of the ON intervals corresponds to the distribution of file sizes. There is a sharp cutoff on flow lengths in the case of fixed-size file transfers, and the heavy power-law tail for the case of Pareto-distributed file sizes, with the same value of the shape parameter α.[4]

With heavy packet losses ($N = 40$), however, the flow length distributions are significantly different, with much shorter flows dominating even for Pareto-distributed file sizes. In presence of packet losses and TCP timeouts the long file transfers are broken into many shorter flows, whose distribution depends on the value of flow threshold Δt_f.

The corresponding log-log plots of the complementary cumulative distribution functions (CCDF) for the OFF-intervals are shown in Figure 5. With no packet losses, and Δt_f greater than RTT, the OFF-interval distribution is generated exclusively by the exponential inter-flow distribution (mean 1.1 seconds). With heavy packet losses, however, at the longer time scales the inter-flow OFF-intervals are generated by TCP timeouts. Notice the presence of a substantial tail of the distribution reaching times in excess of 300 seconds, but given that there is an upper bound on the duration of the longest cumulative TCP timeout this tail does not stretch much further. Moreover, this tail is not well approximated by a (truncated) power law.

Asymptotic robustness of the ON- and OFF-interval distributions with respect to the choice of the flow threshold parameter Δt_f is a subtle matter. An elegant analysis of the Internet data presented in [30] illustrated the robustness of the measured tail behavior of ON- and OFF-interval distributions for high variability traffic by insensitivity to the choice of the flow threshold Δt_f as long as it is large enough. Note, however, that such analysis requires that at large time scales Δt_f is still smaller than the tail values of the inter-flow OFF-intervals, for otherwise for any finite duration dataset one would always end up with a single long flow when Δt_f is large enough.

Therefore, it is easy to explain why the flow size distributions under heavy loss condition shown in Figure 4 do not manifest any Pareto tail behavior when the file size transfers are Pareto distributed. Mathematically speaking, they should remain heavy-tailed since it's an asymptotic property, while the retransmission timeouts are bounded. Thus, in theory we should increase Δt_f above the range of the largest TCP timeouts, (approximately 400 seconds) to observe the robust behavior of the tail distribution of the ON-intervals.

Alas, what theory recommends is impossible in practice if the OFF-interval distribution is not heavy-tailed, and its mean is smaller than 400 s. In this paper the tradeoff for generating sufficiently high variability of aggregate traffic is that the mean OFF-interval must be rather small (1.1 s, but even if it were an order of magnitude larger the argument below still applies). Thus, if the inter-session OFF-intervals are random but not heavy-tailed, e.g., exponen-

[4]The maximum TCP window size in the model network is smaller than the end-to-end bandwidth-delay product. But since the flow measurement is carried out after the bottleneck link and *before* the large bandwidth link, after the slow start the packets will be arriving in a back-to-back fashion.

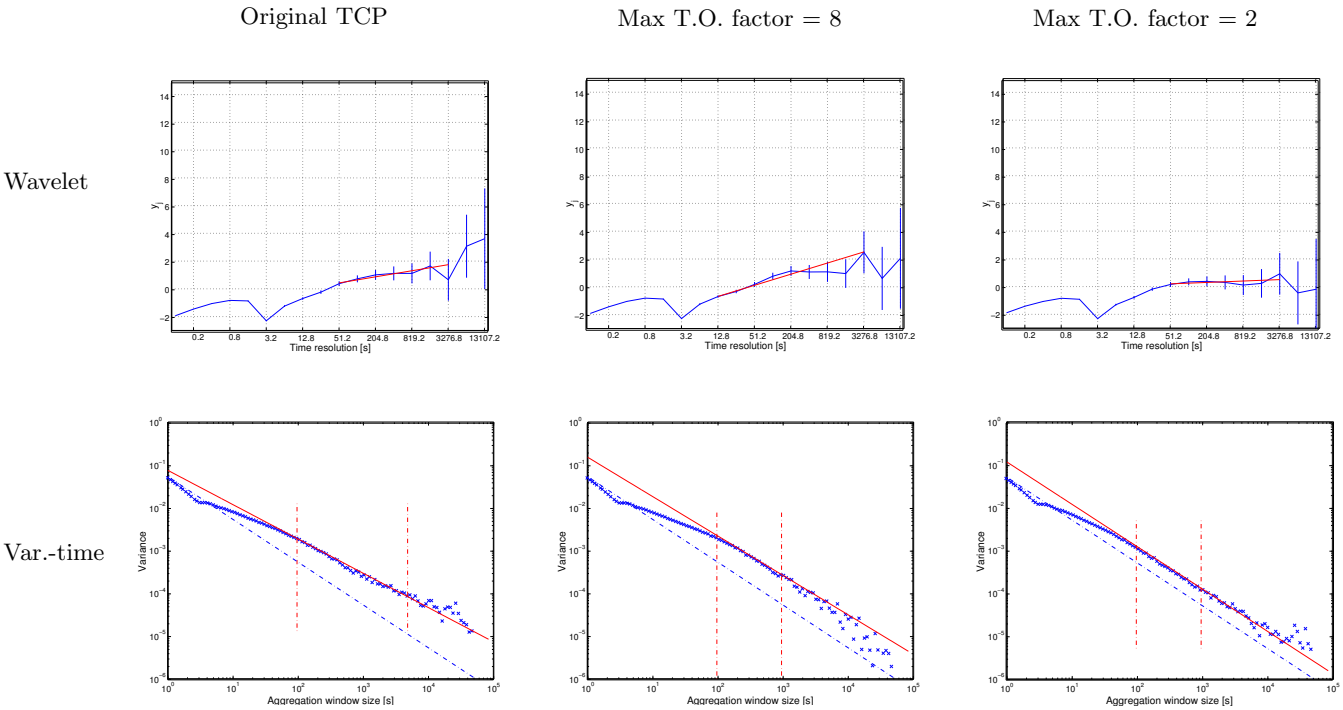


Figure 3: Scaling analysis with reduced maximum TCP timeouts, left to right: maximum timeout factor 64, 8, and 2; for TCP traffic of fixed-size (12 packets) data transfers, at heavy loss conditions ($N = 40$). Dashed lines in VT plots correspond to the slope with $H = 0.5$, i.e. short-range dependence. Note the reduction of the time range of loss-induced scaling with the reduction of the maximum timeout factor.

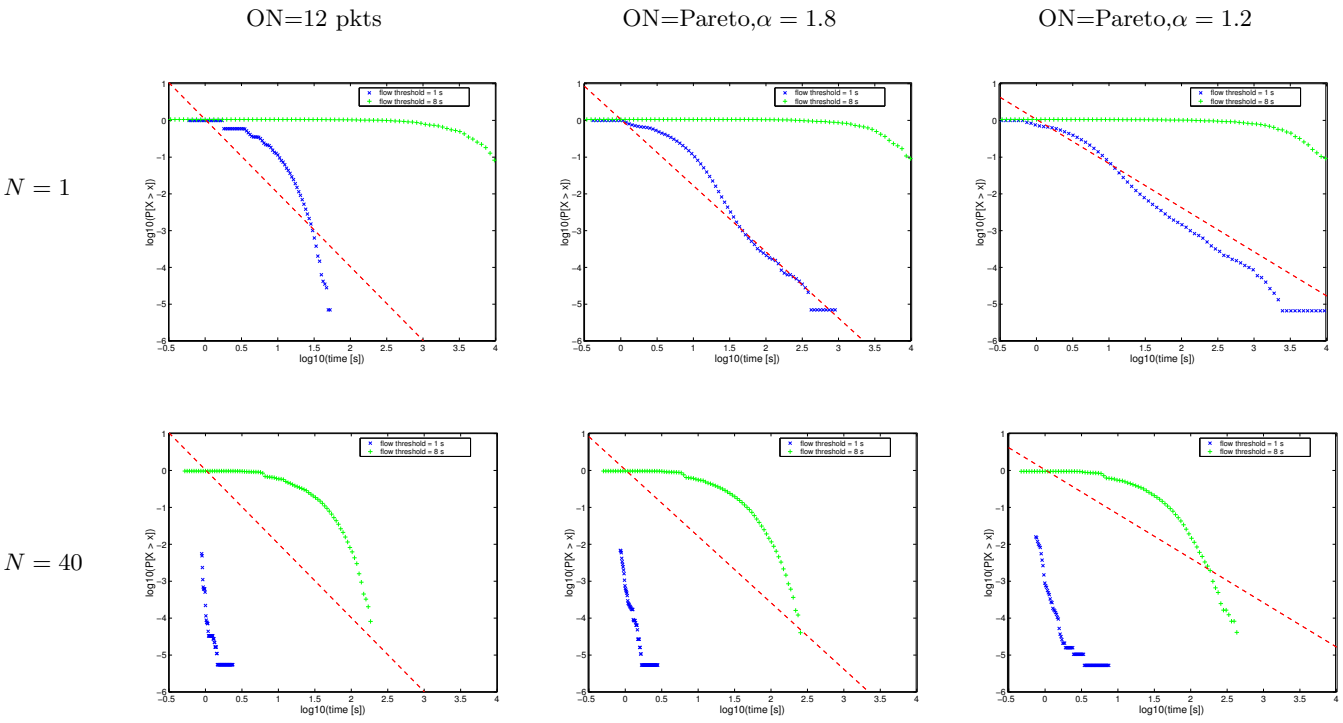


Figure 4: Complementary cumulative distribution function (CCDF) of flow ON-intervals for the three traffic scenarios ($K = 10$) and two different flow thresholds. Crosses show distribution for $\Delta t_f = 1$ s, and plus signs for $\Delta t_f = 8$ s. Columns from left to right: small fixed sized transfers (12 packets long), Pareto distributed file sizes with $\alpha = 1.8$, and Pareto with $\alpha = 1.2$. Top row: lossless traffic ($N = 1$), bottom: heavy packet losses ($N = 40$).

$N = 1$

$N = 40$

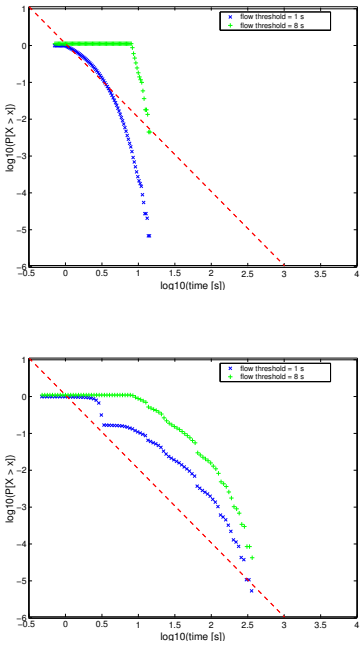


Figure 5: Complementary cumulative distribution function (CCDF) of flow OFF-intervals for the traffic scenario of small fixed-size transfers. The other two traffic scenarios generate identical plots, as the CCDFs are determined only by the packet losses and not by file sizes. Top: lossless traffic, bottom: heavy traffic losses.

tially distributed with mean $1/\lambda = 1.1$ s, then the probability of encountering an OFF-interval larger than 400 s is $P[X > 400] = e^{-400\lambda} \approx 1.2 \cdot 10^{-158}$. Even though theoretically it is true that the flows are heavy-tailed, for all practical time-scales we will never experimentally observe any tail.

In summary, in cases when inter-session OFF intervals have small mean and little variability, and TCP session sizes have large variability, the high packet losses will result in strongly reduced variability of the flow ON-intervals for the practical range of Δt_f. But this effect is counter-acted by the traffic correlations induced by a broad—but not heavy tailed—TCP timeout distribution over practically observable time scales. Now it is easy to see the reason why experiments appear to show self-similarity of TCP traffic under heavy loss conditions.

Since it is not feasible to beat the statistics by executing infinitely long simulations, we will expose this effect in a control experiment where the variability of TCP timeouts has been artificially eliminated.

5.1 Removal of TCP Timeout Variability

In order to separate the two effects of TCP timeouts: *i)* reduced variability of flow sizes (ON-intervals), and *ii)* increased variability of OFF-interals, we again experiment with a modification to the TCP protocol. In this experiment we use the Pareto-distributed file sizes ($\alpha = 1.2$), and reduce the variability in the OFF-intervals by *i)* setting all TCP timeouts to 4 seconds, and *ii)* setting the inter-session

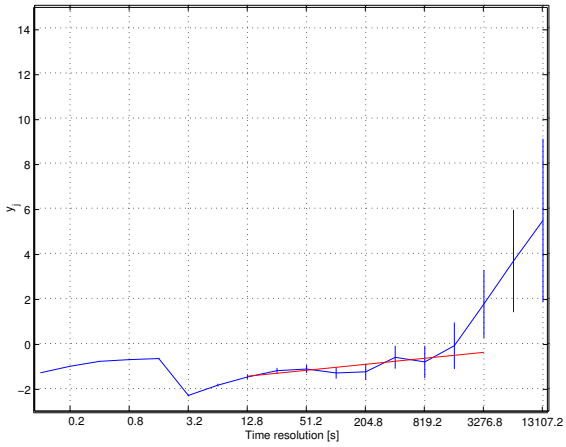


Figure 6: Wavelet plot for the Pareto file size distribution traffic scenario ($\alpha = 1.2$) with heavy packet losses, but with a modified TCP with the constant timeout length fixed at 4 seconds. The onset of asymptotic scaling is at the scale of longest consecutive timeouts; compare with Figure 2.

OFF-intervals constant, to 1.1 seconds. The resulting OFF-interval distribution for high packet losses ($N = 40$) shows that the maximum observed OFF-interval length has been reduced to less than 40 seconds. The reason that we observe OFF-intervals longer than 4 seconds is due to consecutive losses and timeouts.

The effect this modification has on the scaling is most noticeable in the wavelet scaling plot, shown in figure 6. High losses lead to a complete destruction of Pareto behavior for short to medium time scales, and the asymptotic Pareto scaling sets in only at time-scales above approximately 1000 seconds.

In a previous work [16] it was asserted that the TCP transport plays an important role in preserving self-similarity induced by session-level variability under packet loss conditions. Using simulations, these authors compared reliable and unreliable transport (TCP vs. UDP) and found that packet losses can erode self-similarity in UDP traffic, but concluded that reliable TCP transport preserves the long range dependence. However, we have shown here that TCP's effect on scaling is more subtle and caution is required in the analysis of the resulting scaling: loss-induced scaling can stretch to time scales on the order of 10^2 seconds, and even if scaling plots look deceptively like a power law in this range, the exponent is, in general, different from the asymptotic behavior. The reliable TCP transport does not simply preserve self-similarity under heavy packet losses, but leads to two *separate* scaling regimes with the crossover at the timescale of the longest timeouts.

6. STATISTICAL ERRORS AND PARAMETER SENSITIVITY

The network model parameters used in the experiments were chosen so that the simulations could unambiguously reveal the scaling phenomena, while remaining computationally practical for very long time scales that we studied. However, the experiments constitute only a few points in a vast parameter space, and we need to address the robustness of our conclusions with respect to changes in the network parameters.

The first set of questions concerns the convergence of statistics in the ON/OFF traffic superposition with heavy-tailed distributions: *i*) is $K = 10$, i.e. a superposition of ten traffic ON/OFF processes sufficient to approximate a self-similar aggregate process, and *ii*) how significant is the undersampling bias in the finite samples drawn from the heavy-tailed file size distributions [6, 24].

Convergence of the superposition model: We examined the effect of finite K on the Hurst parameter estimates by directly simulating the superposition of K ON/OFF alternating renewal processes (FBM model), with K increasing from 10 to $5 \cdot 10^3$, for the simulated time of 10^6 seconds.[5] ON/OFF-period distributions were chosen as in the network simulations: OFF-periods exponential with mean 1.1 s, and ON-periods Pareto with mean 1.8 s and $\alpha = 1.2$. We have observed that in each of these experiments the VT plot shows virtually no deviation from a pure power law for aggregation windows ranging from 1 to $5 \cdot 10^3$ seconds. However, the exponent estimate is a *random variable*. Figure 7 shows the derived Hurst parameter estimates together with the error bars.

As $K \to \infty$, the limit theorem predicts that the process converges to Fractional Gaussian Noise (FGN) with $H = 0.9$. Both VT and wavelet estimates show good agreement about $H = 0.85$ which is a little lower than the theoretical value. In the case of the VT estimator this could be explained by the negative estimator bias [22], for FGN with $H = 0.9$ of approximately -0.05. We attribute the scatter of the Hurst parameter estimates to undersampling: even for $K = 10,000$ and 10^6 seconds the extremely long ON-periods that produce long-range correlations are still not adequately sampled.

Given the stability of wavelet-based statistics, we conclude that $K = 10$ is sufficient to produce a *qualitatively* correct approximation to the self-similar aggregate traffic in this case. Our estimates from the FBM model should also be compared with the estimates obtained from Figure 2 (Table 2) for the lossless case with $\alpha = 1.2$. By changing the wavelet base (not shown) we get $H = 0.85$, i.e. a good match with the FBM model.

Timescales repesented in the data: We again consider simulated time-series of length $T = 10^6$ s, with K ON/OFF-processes. As in the previous paragraph let the mean ON-period $t_{ON} = 1.8$ s, the mean OFF-period $t_{OFF} = 1.1$ s, and let $t_{ON/OFF} = t_{ON} + t_{OFF} = 2.9$ s. The expected number of transfers for one client/server pair (C/S pair) would be $n = E[\#\text{transfers}] = \frac{T}{t_{ON/OFF}} \approx 3.4 \cdot 10^4$. With K simultaneous C/S pairs and no contention we get $n \cdot K$ transfers, which is close to numbers in Table 1, so n is a fairly good estimate of the number of Pareto samples drawn for each process in the lossless case. As stated in [6], the expected largest sample observed $E[Y]$ from n Pareto samples with mean $E[X]$, is $E[Y] \approx E[X]n^{1/\alpha}$, giving us here $E[Y] \approx 7.2 \cdot 10^3$ s. [24] also recommend examining the sampled data to keep track of the largest samples collected. We can do this by studying the rightmost top-row graph of figure 4. This indicates reasonable samples up to approximately $10^{3.5} \approx 3.2 \cdot 10^3$ with some samples up to 10^4. Thus, starting from 1 s timescale, we could expect to have useful data somewhere between three and four decades, which agrees with the VT plots in figure 2.

[5] An initial warmup period of 10^6 seconds was also simulated and discarded. This was important for stationarity in the case of $K = 10^4$, and thus for VT estimates, but insignificant for other cases.

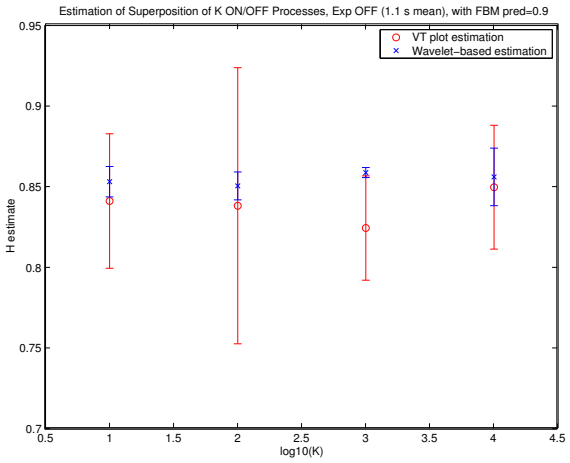


Figure 7: Estimates of the Hurst parameter H obtained in simulations of FBM by superposition of K alternating ON-OFF renewal processes, with Pareto-distributed ON periods and exponentially-distributed OFF periods. Circles: average H using VT plot estimation, Δt_fcrosses: average H using wavelet estimation method. Each data point represents the average of H estimates obtained from 5 independent time series of length 10^6 s with 90% confidence intervals. The VT plot used aggregation window sizes from 1 to $5 \cdot 10^3$ s.

For high loss, the number of transfers per C/S pair is drastically reduced, to about $2.8 \cdot 10^3$ (from table 1), leading to a largest expected sample of only $8.9 \cdot 10^2$ s. (Here we cannot rely on figure 4.) However, the VT plots show reasonable statistics to somewhat larger scales. This could be simply and artifact of the VT plot, but a possible explanation is that there are now many competing C/S pairs (total 390) that also draw Pareto samples, and could induce more long-lasting correlations through the competition in the bottleneck link (by suppressing other flows for long times).

Parameter sensitivity: Whereas the previous papers have focused on the loss probability as the sole factor determining the traffic rate correlation structure imposed by TCP, our experiments indicate that the situation is more complex. We examined the sensitivity to the following model parameters: *i)* increasing the bottleneck buffer size to 100

and 200 packets, and *ii)* increasing the inter-transfer OFF-period mean by a factor of 10 and 100. In both cases the load factor N was increased to maintain a comparable loss probability, and in both cases a break-down of asymptotic scaling became more evident. The latter is attributed to increased undersampling of the process due to the sparsening of the traffic, as each of these changes increases the mean of the flow OFF-periods.

By increasing these parameters, the distance between the smallest and the largest OFF-intervals (induced by time-out) tends to decrease, which reduces the range of the loss-induced scaling. In the case of the buffer size there is a connection between the buffer size and the inter-transfer times and TCP timeouts. For larger buffers and constant mean inter-transfer time, the buffer will not have time to empty between transfers or short time-outs. Thus, the flows become "compressed" and the OFF-periods shrink below the flow separation threshold.

7. CONCLUSIONS

In this report we have used a simulation model to illustrate and compare traffic rate correlations induced by the TCP protocol with those induced by high-variability at the session level.

We found that under heavy packet loss conditions the TCP timeouts induce strong traffic rate correlations stretching far into medium range time scales. Over their range, these correlations decay approximately like a power law with an exponent that is only weakly dependent on the session-level file size distribution. Under certain traffic scenarios there is a crossover in scaling from the time range dominated by TCP timeouts to the asymptotic time range dominated by session size distribution. We studied packet flow distributions for a structural explanation of our findings and observed *i)* the OFF-period distribution develop a noticeable tail as losses induce longer and longer timeouts, and *ii)* we were unable to detect heavy-tailed ON-periods as losses increased. By experimenting with a modification to the TCP mechanism where the timeouts are limited or fixed, we *i)* experimentally link the flow OFF-period distribution to the TCP induced scaling (in the absense of a mathematical link) and *ii)* couple the broken ON-periods to an "erosion" of self-similarity over medium-range time-scales.

We conclude that TCP in combination with extreme losses does not induce self-similarity in the traffic. Instead, heavy losses may lead to a form of "erosion" of self-similarity up to medium-range time-scales. This "erosion" is masked by medium-range scaling effects due to TCP timeouts.

Therefore, it is highly improbable that TCP contributes much to scaling measured in the Internet traffic, as *i)* the effect is sensitive to parameters such as source-level OFF-period distribution and buffer sizes, and *ii)* it would happen only if the normal day-to-day operating condition of the Internet manifested steady, widespread and extreme congestion losses exceeding 10% over many hours.

Acknowledgments

The authors thank Walter Willinger, AT&T Labs, for a number of illuminating discussions on the finer points of scaling analysis.

8. REFERENCES

[1] P. Abry and D. Veitch, "Wavelet Analysis of Long-Range-Dependent Traffic", *IEEE Transactions on Information Theory*, vol. 44, No. 1, Jan. 1998.

[2] J. Beran, "Statistics for Long-Memory Processes", Chapman & Hall, New York, NY, 1994

[3] L. Brakmo and L. Peterson, "Experiences with Network Simulation", *in Proceedings of the ACM SIGMETRICS conference on Measurement & modeling of computer systems*, pp. 80-90, Philadelphia, PA, May 1996.

[4] D. R. Cox, *Renewal Theory*, Chapman and Hall, New York, 1982.

[5] M. E. Crovella and A. Bestavros, "Self-similarity in world wide web traffic: Evidence and possible causes", *IEEE/ACM Transactions on Networking*, vol. 6, pp. 835-846, Dec. 1997.

[6] M. Crovella and L. Lipsky, "Long-Lasting Transient Conditions in Simulations with Heavy-Tailed Workloads", *in Proceedings of the 1997 Winter Simulation Conference*, pp. 1005–1022, Atlanta, GA, Dec. 1997.

[7] A. Erramilli, Onuttom Narayan, and Walter Willinger, "Experimental Queueing Analysis with Long-Range Dependent Packet Traffic", *IEEE/ACM Transactions on Networking*, 4(2):209-223, April 1996.

[8] A. Feldmann, A. Gilbert, P. Huang, and W. Willinger, "Dynamics of IP traffic: A study of the role of variability and the impact of control", *in Proceedings of the ACM/SIGCOMM'99*, Cambridge, MA, Aug. 1999.

[9] D. Figueiredo, B. Liu, V. Misra, and D. Towsley, "On the Autocorrelation Structure of TCP Traffic", *Tech. Report 00-55*, Dept of Computer Science, University of Massachusetts, Amherst, MA, Nov. 2000.

[10] S. Floyd, "Simulator tests", *http://www-nrg.ee.lbl.gov/nrg-papers.html*, July 1995.

[11] S. Floyd and V. Jacobson, "On Traffic Phase Effects in Packet-Switched Gateways" *Computer Communication Review (ACM)*, Vol. 21, No. 2, April 1991.

[12] L. Guo, M. Crovella, and I. Matta, "How does TCP generate Pseudo-self-similarity?", in *Proceedings of the Ninth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems*, pp. 215-223, Cincinnati, OH, Aug. 2001

[13] Y. Joo, V. Ribeiro, A. Feldmann, A. Gilbert, and W. Willinger, "TCP/IP Traffic Dynamics and Network Performance: A Lesson in Workload Modeling, Flow Control, and Trace-Driven Simulations", *SIGCOMM Computer Communication Review (ACM)*, Vol. 31, No. 2, April 2001.

[14] W. Leland, M. Taqqu, W. Willinger, and D. Wilson, "On the Self-Similar Nature of Ethernet Traffic" (Extended Version), *IEEE/ACM Transactions on Networking*, Vol. 2, No. 1, Feb. 1994.

[15] S. Manthorpe, I. Norros, and J. Y. L. Boudec, "The Second-Order Characteristics of TCP", *in Proc. of Performance'96*, Presented in the Self-Similarity hot-topic session, Lausanne, Oct. 1996.

[16] K. Park, G. Kim, and M. Crovella, "On the Relationship Between File Sizes, Transport Protocols, and Self-similar Network Traffic", *in Proc. IEEE International Conference on Network Protocols*, pp. 171–180, 1996.

[17] K. Park and W. Willinger, "Self-Similar Network Traffic: An Overview", in *Self-Similar Network Traffic and Performance Evaluation*, Edited by K. Park and W. Willinger, Wiley – Interscience, New York, NY, pp. 1-38, 2000

[18] V. Paxson, "Automated Packet Trace Analysis of TCP Implementations", *in Proceedings SIGCOMM Symposium (ACM)*, pp. 167-179, Cannes, France, Sept. 1997.

[19] V. Paxson, "End-to-End Internet Packet Dynamics", *IEEE/ACM Transactions on Networking*, pp. 277-292, Vol. 7, No. 3, June 1999.

[20] V. Paxson and S. Floyd, "Wide-Area Traffic: The Failure of Poisson Modeling", *IEEE/ACM Transactions on Networking*, Vol. 3, pp. 226-244, June 1995.

[21] V. Paxson and S. Floyd, "Why we don't know how to simulate the Internet", *in Proceedings of the 1997 Winter Simulation Conference*, Atlanta, GA, Dec. 1997.

[22] M. Taqqu and V. Teverovsky, "On Estimating the Intensity of Long-Range Dependence in Finite and Infinite Variance Time Series", *in A Practical Guide to Heavy Tails: Statistical Techniques and Applications*, Adler, Feldman, Taqqu, Editors, Birkhauser, Boston, 1998.

[23] M. S. Taqqu, W. Willinger and R. Sherman, "Proof of a Fundamental Result in Self-Similar Traffic Modeling", *Computer Communication Review (ACM)*, vol. 25, pp. 5–23, 1997.

[24] M. Roughan, J. Yates and D. Veitch, "The mystery of the Missing Scales: Pitfalls in the Use of Fractal Renewal Processes to Simulate LRD Processes", *ASA-IMA Conference on Applications of Heavy Tailed Distributions in Economics, Engineering and Statistics, Washington, DC, June 1999.*.

[25] D. Veitch, `http://www.emulab.ee.mu.oz.au/~ darryl/secondorder_code.html`, June 2001.

[26] A. Veres and M. Boda, "The Chaotic Nature of TCP Congestion Control", *in Proc. IEEE INFOCOM'2000*, (Tel Aviv, Israel), Apr. 2000.

[27] W. Willinger, V. Paxson and M. S. Taqqu, "Self-Similarity and Heavy Tails: Structural Modeling of Network Traffic" *in A Practical Guide to Heavy Tails: Statistical Techniques and Applications*, Adler, Feldman, Taqqu, Editors, Birkhauser, Boston, 1998.

[28] W. Willinger, DARPA Network Modeling and Simulation Principal Investigators Meeting, La Jolla, CA, March 2001.

[29] W. Willinger, personal communication, June 2001.

[30] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson, "Self-Similarity Through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level", *IEEE/ACM Transactions on Networking*, 5:71-86, 1997.

APPENDIX

A. EXPERIMENT PARAMETERS

A.1 Network Dimensioning

The TCP loss-induced traffic correlations will only show up in high packet loss regimes. It is well known that it is difficult to provoke TCP into high loss rate regimes since it will back off when congestion is detected. In order to be able to place a high load on the bottleneck links (meaning many TCP connections), simulate multiple bottlenecks, and still keep the simulations resource demands feasible, we set the bottleneck bandwidth quite small, 10 packets per second. Other link capacities were set high so that they would not interfere with the traffic (100 Mbps for individual connections or 1 Gbps for aggregate loads). A link delay is set to 80 ms for the measurement link and for all deflection links going to alternate clients. All other links have zero delay. The bandwidth-delay product is large: 10 million Bytes, or about 10,000 packets. The RTT is about 270 ms. The bottleneck link buffer size was set to 33 packets, which is one larger than the TCP window size, to ensure that a single connection will not experience any packet losses. Thus, losses are only induced due to competition among TCP connections. Bottleneck buffers use a droptail overflow policy. Other buffers are infinite. All server-subnetworks are equivalent. The links connecting the client hosts to the router in each of the client-subnetworks are given a random delay, uniform distribution on the interval $[0, 5]$ ms, at network initialization time.

A.2 TCP Parameters

Table 4 lists the TCP parameters used in the experiments. The model of the TCP protocol used is the Tahoe version (no fast recovery), with delayed ack option switched off for compatibility with prior studies (see Section 1).

Table 4: TCP parameters

Parameter	Value
Max Segment Size *MSS* (bytes)	1000
Receive Window Size (MSS)	32
Send Window Size (MSS)	32
Send Buffer Size (MSS)	128
Max Number of Retransmissions	12
Slow timer granularity (seconds)	0.5
Fast timer granularity (seconds)	0.2
Max Segment Lifetime (seconds)	60.0
Max connection idle time (seconds)	600.0
Delayed ACKs	no
TCP version	Tahoe

Table 5: File transfer traffic parameters

Parameter	Value
start time (seconds)	10.0
start window (seconds)	1.0
request size (bytes)	4

A.3 Traffic Startup, Routing, and Probes

The simulated model employs a static version of the OSPF routing protocol to establish routes. Enough time is left between the simulation start and the first TCP file transfers to allow OSPF to converge. In each client host the first transfer request time is randomized (uniform distribution) to occur within a delayed start window (see Table 5) in order to avoid artifacts due to synchronized connection initiations.

SIGCOMM Award Nominations

The SIGCOMM Award was initiated in 1989 as a means of honoring computer communication professionals for outstanding lifetime technical achievement in the fields of data and computer communications. The award consists of a plaque and a $2,000 honorarium. The award is presented at the annual SIGCOMM Conference, at which time the awardee is invited to deliver a technical address. In the following are guidelines for submitting a nomination.

(1) Self-nominations are not accepted.

(2) The nominee need not be a member of ACM SIGCOMM.

(3) The nominator must be a member of ACM SIGCOMM.

(4) Nominations must be received by the Chair of the SIGCOMM Award Committee no later than March 31st each year.

(5) Nominations that do not result in an award may be resubmitted/updated in subsequent years.

(6) Previous awardees are not eligible for future nominations.

(7) Members of the Award Committee are not eligible.

(8) Members of the SIGCOMM Executive Committee (Chairman, Vice Chairman, Secretary-Treasurer, Past Chairman, and Editor) are not eligible.

Material to be included in the nomination:

(1) Curriculum Vitae, including publications, of nominee.

(2) Concise statement (one sentence) of the work for which the award is being nominated. This statement will appear on the award plaque.

(3) Description of the nominee's role in the work justifying the nomination.

(4) Letters of recommendation from others discussing the rationale for the nomination and by what means the recommender knows of the nominee's work. It is recommended that at least three letters of recommendation be included in the nomination materials.

(5) Justification for declaring the nominee's work to be a major, lifetime contribution to computer communications.

The nomination should be made in the form of a letter addressed to the Chair of the SIGCOMM Award Committee. The nominator should solicit recommendations from colleagues in the field who are most familiar with the nominee's achievements. It is not recommended to send copies of published papers or books along with the nomination materials. The nominator is responsible for gathering all nomination materials and sending two copies of all materials to reach the Chair of the Award Committee before March 31st. Submissions can be made by email (preferred) or paper (two copies). No late nominations will be considered for the current year. They will be held over until the following year for consideration.

The Chair of the SIGCOMM Award Committee is:

Karen R. Sollins
MIT Lab for Computer Science
200 Technology Square
Cambridge, MA 02139
Voice: +1 617 253 6006
Fax: +1 617 253 2673
Email: sollins@lcs.mit.edu