

Computer Science Department  
Boston University

**Networks PhD Depth Exam**  
**November 2002**

1. Check that there are 6 questions over 9 pages including this page. Answer all questions. Turn in your answers by Thursday, November 14, 4:00pm, at MCS-140F.
2. You are free to consult any notes, books and papers during the examination—make sure to include your references. You are NOT allowed to consult with any person during the examination.
3. You may typeset your answers (in which you should feel free to include hand-drawn figures), or you may neatly write your answers. Please answer each question in a separate writeup.
4. Although questions are of different length, they are of equal weight.
5. Please indicate the question number and your code number on each answer sheet. Do not write your name on the answer sheet.
6. If you have any doubt as to the interpretation of a question, make a reasonable assumption and explain your interpretation in your answer. No explanations will be given during the exam.
7. Your answers should show research maturity and depth, so you may support your answers not only by analysis, but possibly by measurements or simple simulations. Be creative!

**Networks PhD Depth Exam**  
**October 2002**  
**Question 1**

Consider the problem of measuring one-way delays over network paths—for example, in the Internet. Say we are interested in measuring the time it takes for each packet in a series to travel from node  $A$  to node  $B$ . Let's assume that we have instrumentation at each node; that is, we can actively send packets from  $A$  to  $B$  or from  $B$  to  $A$ , and furthermore assume that it is possible to capture packet traces at node  $A$  and node  $B$  (for example, using `tcpdump`). Timestamps in the packet logs are taken using the clock on the local system ( $A$  or  $B$ ).

- (a) First assume that the clocks on the two systems are synchronized to a common timebase, like GPS. List (at least two) properties of the paths from  $A$  to  $B$ , and from  $B$  to  $A$ , that you can learn something about? What exactly can you learn? What analyses would you perform in order to elucidate those properties?

Don't restrict yourself just to those properties that are precisely determined; consider any path properties that you could learn *anything* about and explain the analysis approach needed.

- (b) A typical problem that occurs in practice is that the clocks involved are *not* synchronized. Assume the clocks are not necessarily synchronized (but do run at the same rate) how would your answer in (a) change?
- (c) Now let us assume that clocks do not run at the same rate, but both clocks run at constant rate (this is in fact typical of stock PC hardware). We wish to estimate the difference in rates of the two clocks, using the one-way delay measurements.
- (i) Describe why this is a difficult problem.
  - (ii) Pose this as a statistical estimation problem. Describe how you would solve it as an estimation problem.
  - (iii) Pose this as an optimization problem. Describe how you would solve it as an optimization problem.

**Networks PhD Depth Exam**  
**October 2002**  
**Question 2**

- (a) Why are overlay networks of interest? That is, what value do they provide over and above IP networks? Why is end-system multicast growing in popularity, while IP multicast is not?
- (b) Explain the relationship between routing in overlay networks and IP. What ideas from routing protocols developed for IP unicast and IP multicast can be used in overlay networks? What ideas cannot? Give examples, referring to both proposed and commonly used routing protocols.
- (c) Explain why the routing problem is different in overlay, ad hoc, and IP networks. What assumptions made in IP routing are not true in overlay networks and in ad hoc networks? What similarities are there between routing in ad hoc and overlay networks? Explain.
- (d) Do overlay networks violate the end-to-end principle? Comment. If you conclude that they do violate the end-to-end principle, what does this imply for their eventual impact in the Internet? If you conclude that they do not violate the end-to-end principle, explain why not.

**Networks PhD Depth Exam**  
**October 2002**  
**Question 3**

Consider the following variant of traditional Additive-Increase Multiplicative-Decrease (AIMD)<sup>1</sup> control used in the Transmission Control Protocol (TCP). Define a *congestion epoch* as the sequence of additive increases ending with one multiplicative decrease of the transmission window. Let  $w_0$  denote the value of the window at the beginning of the congestion epoch. In this modified AIMD, during multiplicative decrease (i.e. upon the detection of packet loss, say at time  $t$ ), only  $w_0$  is halved rather than the value of the current window  $w(t)$ . For the simple case of two competing flows, answer the following questions:

- (a) Define *convergence to fairness* as the time it takes for the two competing flows to converge to their fair share of the bottleneck resource. Does this variant to AIMD converge to fairness faster or slower than the original AIMD used in TCP? Measure the difference in terms of the number of additive increase steps needed.
- (b) Is the modified-AIMD more or less efficient than the original AIMD in terms of allocated resources? What is the percentage improvement or degradation?
- (c) Define *smoothness* by the magnitude of window oscillations. Is the modified-AIMD more or less smooth than the original AIMD? What is the percentage improvement or degradation?
- (d) Is a flow that uses the modified AIMD TCP-friendly, i.e. do you expect it to achieve the same throughput as a regular TCP under the same delay and loss conditions?

Several alternatives to traditional AIMD congestion control such as the variant above, the binomial<sup>2</sup> and the SIMD<sup>3</sup> algorithms, have been proposed to better support multimedia applications.

- (f) What kind of “better support” are they claimed to provide? Do you agree with this claim?
- (g) Are there other ways to provide such multimedia support without any modification to the regular AIMD rules? Describe other approaches that may not require any modification to AIMD and compare the pros and cons of these methods with AIMD variants/alternatives.
- (h) Most, if not all, AIMD variants/alternatives proposed for multimedia support are evaluated assuming RED (Random Early Drop) routers. However, most of today’s routers operate in a tail-drop manner! Do these AIMD variants/alternatives indeed require RED support? Explain why or why not.
- (i) Do RED routers maintain the end-to-end principle? Explain why or why not.

---

<sup>1</sup>**Reading List:** “Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks,” D.-M. Chiu and R. Jain, Computer Networks and ISDN Systems, Vol. 17, pp. 1-14, 1989.

<sup>2</sup>**Reading List:** TCP-friendly Congestion Control for Real-time Streaming Applications, D. Bansal and H. Balakrishnan. In Proceedings of IEEE INFOCOM 2001.

<sup>3</sup>**Reading List:** Jin, Shudong; Guo, Liang; Matta, Ibrahim; Bestavros, Azer. TCP-friendly SIMD Congestion Control and Its Convergence Behavior, Technical Report BUCS-2001-015, May 8, 2001. The paper also appears in IEEE ICNP 2001.

**Networks PhD Depth Exam**  
**October 2002**  
**Question 4**

Consider the problem of routing in a backbone IP network using the link-state OSPF or IS-IS routing protocol. Routing from a source node  $s$  to a destination node  $d$  is typically done along the least-weight path, where the weight of a path is the sum of the weights of the links along that path. Assume link weights are set administratively and are static. Now, you have done some measurements on the backbone and observed short-term link failures. These failures result in re-routing traffic and few links (say 10% of all links) get overloaded, i.e. the utilization of a link after re-routing increases beyond, say, 50%. To overcome this overload problem, someone proposed a so-called “deflection routing” scheme. In this scheme, before a router forwards a packet, it checks the utilization of the next link (hop) along the current least-weight path. This utilization is measured over a certain time period as the fraction of time the link is busy transmitting a packet. If the next hop’s utilization is beyond a certain threshold, the packet is deflected toward a neighboring node other than the next hop.

- (a) Given that link overloads never exceed 80%, why are these short-term overloads a problem?
- (b) State the condition(s) that must be met for this packet deflection routing to yield loop-free paths.
- (c) If the topology of the backbone network is such that every pair of nodes are connected by a set of equal-length paths (i.e. paths of equal hops), determine the range of link *weights* that would automatically satisfy the condition(s) of part (b).

Consider a variation of the above packet deflection routing scheme, where a packet originating from source  $s$  is always deflected exactly once to an intermediate destination node, not necessarily a direct neighbor. From there, the packet is routed to the actual destination  $d$ .

- (d) For this variant of deflection routing, answer the same questions in parts (b) and (c) above. Compare the two deflection routing schemes in terms of implementation cost and expected delay performance.

Another solution to the overload problem is to let the routing protocol adapt to link weights that *measure* the current state of links. The weight of a link could reflect some load measure, e.g. utilization or delay. Consider the following three link weight definitions: (1) Each link weight measures link utilization, and the path weight is the sum of the link weights; (2) Each link weight measures link utilization, and the path weight is the max of the link weights; (3) Each link weight measures link delay, and the path weight is the sum of the link weights. You are asked which of the above path weight definitions would you recommend to use. As part of your recommendation, contrast to the above “deflecting routing” solutions and address the following issues:

- (e) How would you define the *stability* of a routing system? How does this stability (or lack thereof) affect the performance of applications?
- (f) Which of the three path weight definitions would yield a more stable system and which would be easier to implement? Explain why. How do these three path weight definitions compare to the “revised ARPANET routing metric” <sup>4</sup>?

---

<sup>4</sup>**Reading List:** Peterson and Davie. Computer Networks: A Systems Approach. 2nd Edition, pp. 301–304.

**Networks PhD Depth Exam**  
**October 2002**  
**Question 5**

The Hop Tree between a sender and a set of receivers is a labeled tree rooted at the sender, with all receivers as leaves. Internal nodes of that tree are the routers at which IP paths from the sender to the receivers diverge. An edge in the Hop Tree is labeled with the number of physical hops along that edge. For example, the Hop Tree between a sender and two recipients, will consist of a tree with a single internal node (which is the router at which the routes from the sender to each of the two receivers diverge) and three labeled edges (representing the path between the sender and the internal node, and the two paths from the internal node to each one of two receivers).

- (a) Write a program that will allow the *efficient* generation of potentially large Hop Trees. Your program should accept as input the name of a file containing a list of recipient IP addresses and should produce as output a list of all the edges in the Hop Tree (each edge represented by the IP addresses of the endpoints and the label for that edge). You can test your program on the set of IP addresses at <http://www.scriptroute.org:3967/>.
- (b) Let  $N$  denote the number of recipients and  $D$  denote the maximum number of (physical) hops between the sender and any one of the recipients. Derive the worst-case number of probes needed by your code.
- (c) What can you say about the average number of probes needed by your code, assuming the nodes in your Hop Tree are chosen at random from the set of routers in the Internet?
- (d) What practical considerations may hinder the performance (or quality) of the output of your procedure (or require you to add corrective procedures)?

**Networks PhD Depth Exam**  
**October 2002**  
**Question 6**

Your advisor requests that you read and understand some obscure paper on network pricing and fairness that he thinks is relevant to your research. You can't find it anywhere online, but luckily the library has a copy. Unfortunately for you, the library's copy has been badly damaged and all that's left is page three (attached). Time is running out, so you're going to have to satisfy your advisor's curiosity by reconstructing parts of this paper from the single page that you have. (Note: You will not be able to locate a copy of this paper, so please don't waste time trying.)

- (a) All subparts based on the set of flows described in the example subsection.
- 1) What is the allocation of rates to flows in this example that maximizes total throughput, i.e. maximizes the sum of the  $f_j$ 's? For full credit, you will need a proof that you cannot do better.
  - 2) Now consider the definition of equilibrium allocation defined in the paper. How would flows allocate tokens to their links to achieve the equilibrium allocation for this example? Please furnish the matrix of  $t_{ij}$  values.
  - 3) What is the resulting allocation of rates to flows in this example, i.e. what are the  $f_j$  values present at the equilibrium you found in 2)?
- (b) Write a few short paragraphs of the related work section for this paper encompassing four relevant citations. Make sure that the citations span an interesting range of material relevant to this paper. This will likely entail going well outside of the scope of your reading list.
- (c) Using the formalism provided in page three of the manuscript, propose a token allocation heuristic that converges to the equilibrium allocation for arbitrary flow graphs. If time allows, provide some preliminary evidence that your heuristic is a good one (formal or experimental evidence is best).

## References

In addition to references on your reading list, you may find the following references helpful as starting points for some of the questions in this exam. Note that some of the following references are not on your reading list.

- “End-to-End Packet Delay and Loss Behavior in the Internet,” J.-C. Bolot, Proceedings of SIGCOMM '93.
- “Critical Path Analysis of TCP Transactions,” P. Barford and M. E. Crovella, Proceedings of SIGCOMM 2000.
- “A Case For End System Multicast,” Yang-hua Chu, Sanjay G. Rao and Hui Zhang, Proceedings of ACM Sigmetrics 2000.
- “Enabling Conferencing Applications on the Internet Using an Overlay Multicast Architecture,” Yang-hua Chu, Sanjay G. Rao, Srinivasan Seshan, Hui Zhang, Proceedings of ACM SIGCOMM '01.
- “End-to-end Arguments in System Design,” J. Saltzer, D. Reed, and D. Clark, ACM Transactions on Computer Systems (TOCS), Vol. 2, No. 4, pp. 195-206, 1984.
- “Inference and Labeling of Metric-Induced Network Topologies,” Azer Bestavros, John W. Byers and Khaled Harfoush, Proceedings of IEEE INFOCOM '02. An earlier version appeared on the reading list as BUCS-TR-2001-010.
- “On the Marginal Utility of Network Topology Measurements,” Paul Barford, Azer Bestavros, John Byers and Mark Crovella, Proc. of the SIGCOMM Internet Measurement Workshop (IMW) '01.
- “Scaling of multicast trees: Comments on the Chuang-Sirbu scaling law,” G. Phillips, S. Shenker, and H. Tangmunarunkit. In *Proceedings of SIGCOMM*, pages 41–51, 1999.

The rest of the paper is organized as follows. In Section III, we provide our distributed token allocation and pricing model and define a natural notion of an equilibrium allocation, demonstrate that an equilibrium always exists, and provide some examples. In Section IV, we compare how our equilibrium allocation compares with other allocation methods, such as max-min fairness and allocations to maximize throughput. We then provide several distributed algorithms which provably converge to an equilibrium allocation along with experimental evaluation of their performance, including convergence time in practice.

### III. DEFINITIONS AND FORMULATION

Consider a set of  $m$  bandwidth-intensive flows which use fixed-path routes through a network of  $n$  links. Each of the links  $\ell_i$  has a capacity  $c_i$ , and each flow  $j$  uses a path  $P_j$  that comprises a subset of the links. An allocation of rates  $r_j$  to the flows is feasible if:

$$\forall i, \sum_{j|i \in P_j} r_j \leq c_i.$$

In order to establish a rate allocation, we allow flows to use the following distributed model. In the first half of each distributed round, each flow  $j$  has a set of  $t_j$  tokens from which it may assign a quantity  $t_{ij}$  to each link  $i$  along  $P_j$ , subject to the constraint that for all  $j$ ,  $\sum_i t_{ij} \leq t_j$ . We assume that arbitrary fractions of tokens may be placed on any link. Finally, we let  $w_i$  denote the total amount of tokens placed by all flows at a link  $i$  in a given round, i.e.  $w_i = \sum_j t_{ij}$ .

In the second half of each distributed round, each link  $i$  then computes a ‘‘fair’’ allocation to each of its participating flows. It does so by computing an allocation  $f_{ij}$  that is a weighted proportion of its capacity  $c_i$  weighted by the number of placed tokens, if any tokens were placed on it, or evenly across the participating flows if not:

$$f_{ij} = \begin{cases} c_i \cdot \frac{t_{ij}}{w_i} & \text{if } w_i > 0 \\ c_i \cdot \frac{1}{\#\{j|i \in P_j\}} & \text{otherwise} \end{cases}$$

Each flow then receives an allocation  $f_j$  equal to the minimum of the allocations provided by the incident links:  $f_j = \min_i f_{ij}$ . From this discussion, it should be clear that a round proceeds in three steps: the flows autonomously allocate tokens  $t_{ij}$ , the links then compute a proportional allocation  $f_{ij}$ , then the flows must accept the minimum of the  $f_{ij}$ ’s they have been allocated.

We now provide some simple facts and definitions, before formulating this problem as an optimization problem with quadratic constraints.

*Definition 1:* An allocation is *tight* if for all  $j$  and for all  $i$  such that that  $t_{ij} > 0$ ,  $f_{ij} = f_j$ .

A tight allocation is one in which every flow has wasted none of the tokens it has placed, i.e. removal of any token from any flow would cause that flow’s allocation to be reduced.

*Definition 2:* An allocation is *at equilibrium* if it is a tight allocation in which for all  $j$ ,  $\sum_i t_{ij} = t_j$ .

We have an equilibrium allocation when every flow has placed all of its tokens, and has wasted none of the tokens it has placed.

#### A. Equilibrium via optimization

To demonstrate that all configurations have an equilibrium, we formulate achieving the equilibrium as an optimization problem. In the equilibrium, the following three conditions are satisfied:

$$\begin{aligned} \sum_i w_i &= \sum_j t_j \\ \forall j: f_j \sum_{i \in P_j} w_i &= t_j \\ \forall i: \sum_{j \in R_i} f_j &= 1 \end{aligned}$$

Note that the second condition is quadratic. To frame this (somewhat arbitrarily) as an optimization problem, we set the objective to be  $\max \sum_j f_j$  for non-negative  $f_j$ , and convert the third equality above to an inequality. This is then a semidefinite program that can be solved with standard methods [1]. Moreover, the fact that the program is semidefinite implies that there is a unique solution when the input is non-degenerate.

Our next step is to develop algorithms capable of fast convergence to this optimal point, but first we present some simple examples to motivate the definitions.

#### B. Examples

For simplicity of exposition, our examples consider the case where all links have unit capacity and all flows are allocated the same number of tokens, i.e.  $\forall j, t_j = 1$ .

Now consider a network which is a ring with 5 links labelled 1 through 5, plus an additional link bisecting the ring labelled link 6, and consider the set of six flows using sets of links  $\{1, 2\}$ ,  $\{2, 3\}$ ,  $\{3, 4\}$ ,  $\{4, 5\}$ ,  $\{5, 1\}$  and  $\{1, 3, 6\}$  respectively.

Under the assumption of unit capacities, the max-min fair allocation gives  $1/3$  to the five flows using only links around the ring and  $2/3$  to the sixth flow using the cross-cutting link, for a total throughput of  $7/3$ .