Network modeling:
choosing the next generation of challenges

Jim Kurose
Department of Computer Science
University of Massachusetts
http://www.cs.umass.edu/~kurose
Overview

- introduction: opening thoughts
- modeling granularity
- modeling: on beyond performance
- application-level networks
- summary
Pat on the back: successes!

- open loop networks:
  - loss, delay, throughput
  - “Kleinrock legacy”
- bounding techniques:
  - network calculi
- self-similarity, LRD
- small, closed-loop nets
  - TCP models
Model complexity, use

Adapted from [Hluchyj 2001]
Disclaimer!

Network modeling: *choosing* the next generation of challenges

- choice of research problems *intensely* personal
- just my own observations...
- ... and in the past I’ve worked on statistical QoS guarantees, multicast, IP video servers, ...
Overview

- introduction: opening thoughts
- **modeling granularity**
- on beyond performance
- application-level networks
- conclusion
The “right” level of abstraction

- Why do we model?
  - evaluate/compare new/existing mechanisms (“mine is better than …”)
  - dimensioning/provisioning
  - insight into why protocols work well

- packet: modeling unit since early 60’s
  - evaluation needed in ‘realistic” wide-area setting:
    - large scale
    - high-speed
Switching speeds increases over time

[adapted from Hui 1997]
Raising the level of abstraction

- packet too microscopic!
- flow-level abstractions
  - call models in telephony
  - WWW transfer: document is workload unit
  - fluid models

modeling challenge: from the micro to the macro, in multi-node scenarios
Flow-level modeling: simple example

- **flow rate**: determined by link capacity, sharing requirements
- **example**: 3 sources, 4 links

- **multi-hop**
- **coupled behavior**
"microanalysis" of TCP gives:

Floyd, Ott: \( B(p,R) = \frac{a}{R\sqrt{p}} \)

PFTK: \( B(p,R) \propto [R(4p/3)^{1/2} + T_0 3(3p/4)^{1/2}p(1+32p^2)]^{-1} \)
TCP: from the micro to the macro

"microanalysis" of TCP gives:

Floyd, Ott: \[ B(p,R) = \frac{a}{R\sqrt{p}} \]
TCP: from the micro to the macro

“fluid analysis” of TCP gives:

\[ B_1(p, RTT) = \frac{a}{(RTT + \bar{q}/C)\sqrt{p}} \]
\[ B_2(p, RTT) = \frac{a}{(RTT + \bar{q}/C)\sqrt{p}} \]
\[ B_1(p, RTT) + B_2(p, RTT) = C \]
\[ p = drop\_function(\bar{q}) \]

[Misra, Baras, Ott 1999; Firoiu, Borden 2000]
TCP: from the micro to the macro

"fluid analysis" of TCP gives:

\[
B_2(p,\text{RTT}) = \frac{a}{(\text{RTT} + \frac{q_1}{C_1} + \frac{q_2}{C_2})\sqrt{p_e-e}}
\]

[Firoiu, Yeom, Zhang 2001; Bu, Towsley 2001]
TCP: flow-level modeling

- preceding analyses for long-lived flows
- modeling short-lived flows:
  - flows arrive according to Poisson process
  - general workload (e.g., transfer size)
  - idealized bandwidth sharing: $M/G/\infty$
    processor sharing

... the beginnings of flow-level (fluid based) multihop analysis!
Simulation: packets versus fluids

Packet-based simulation:
- network traffic: packets
- # sources, data rates increase, so too does simulation workload

Fluid-based simulation:
- network traffic: continuous fluid
  - rate changes at discrete points in time
  - rate constant between changes
- can modulate rate at different time scales
  - single modeling paradigm for many time scales
  - abstract out fine-grained details: simulation efficiency
Packets and fluids

packet arrivals

fluid ~ packet

fluid ~ burst

fluid ~ session

Intuitively: fluid simulation can be more “efficient”
**FIFO packet multiplexer event rate**

*Simple:* each arrival event generates one departure event

9 arrival events

9 departure events

- rearranged in time
- arrivals to downstream router

output multiplexer
FIFO fluid multiplexer:

**Case:** $C > \sum \text{input fluid rates}$
- no queueing
- fluids "pass through" multiplexer with no change in event rates
FIFO fluid multiplexer: more interesting!

**Case:** \( C < \Sigma \) input fluid rates

- Output fluid rate affected by rate changes of other *input* flows!
- Output event rate > input event rate: *ripple effect*
Ripple effect: bad news!

- ripples propagates to downstream routers
  - where they can be magnified
  - propagate further
- no ripple effect in packet simulation
WFQ fluid multiplexer: more interesting!

**Case:** $C < \sum$ input fluid rates

- WFQ provides isolation among flows
- Queueing smooths out input rate variation within flow classes
Fluid simulation observations

- fluid event rates depend strongly on flow interactions, (service disciplines, link rates, propagation delays).
- ongoing efforts:
  - network “calculus” of fluid event rates empirical investigation, comparison
  - techniques reducing fluid simulation event rates

[Liu, Figueredo, Kurose, Towsley 2001]
Overview

- introduction: opening thoughts
- modeling granularity
- on beyond performance
- application-level networks
- conclusion
**On beyond performance**

- **we excel** in data plane
  - loss, throughput, delay

- **Q: Is performance really the major roadblock?**
  - “robustness”
  - “complexity of control”
  - maintainability
  - adaptability
  - reconfigurability
  - security

- **modeling these is hard!**
  - “efficiency” not the most important measure!
  - little/no past work!
  - metrics and models undefined!

[Misra, Baras, Ott 1999; Firoiu, Borden 2000]
# Performance, availability

<table>
<thead>
<tr>
<th>1990s: high performance:</th>
<th>2001: availability 365x24x7</th>
</tr>
</thead>
<tbody>
<tr>
<td>- optimize for common case:</td>
<td>- but be prepared for exceptional events</td>
</tr>
<tr>
<td>- H. Ford (manufacturing)</td>
<td>- component failures</td>
</tr>
<tr>
<td>- E. Nahum (networking student)</td>
<td>- flash crowds</td>
</tr>
<tr>
<td>- deferrable customers</td>
<td>- critical infrastructure</td>
</tr>
<tr>
<td>- expensive resources</td>
<td>- ubiquity</td>
</tr>
<tr>
<td></td>
<td>- ubiquity</td>
</tr>
<tr>
<td></td>
<td>- inexpensive resources</td>
</tr>
</tbody>
</table>
Example: soft state control

Conventional wisdom: “soft-state is robust, less complex than hard-state signaling”

- really?
  - soft-state protocols (IGMP, RSVP) have added (optional) hard-state mechanisms
  - hard-state protocols (ST-II) have soft-state timeouts

- how to define “robustness”?
- how to define “complexity”?
Example: soft state control

Is soft-state really more robust, less complex than hard-state signaling?

- control plane (rather than data plane) question
- posing/answering such a question is:
  - hard: no well-accepted models, paradigms
  - easy: little/no past research
  - important: a fundamental question
Overview

- introduction: opening thoughts
- modeling granularity
- on beyond performance
- application-level networks
- conclusion
Application-layer overlay networks
Application-layer overlay networks
Application-layer overlay networks

- Internet provides single source-destination route
- Resilient Overlay Network (RON): route to intermediate application-level relays to make alternate paths possible
  - performance (QoS)-sensitive routing
  - how to setup/manage/measure routes?

[Anderson 2001]
Application-layer overlay networks

Application-layer multicast

- IP-layer multicast not widely deployed
- application-level multicast to copy/distribute packets
  - how to setup/manage/measure routes?
  - performance overhead

[Chu 2001]
Application-layer overlay networks

Information storage/distribution:
- peer-to-peer: Napster/Gnutella/Kaaza
- publish-subscribe paradigm
- application-level infrastructure to locate, retrieve, cache information?
Summary

- lots of successes to be proud of!
- a few of the new frontiers:
  - higher-level modeling abstractions in data plane
  - application-level overlay, P2P networks
  - “on beyond performance”:
    - the “….. ibilities”
    - modeling in control, management planes
The end

Thanks!

Slides available at
http://gaia.cs.umass.edu/kurose/ips_01.ppt