Network Simulation via Hybrid System Modeling: A Time-Stepped Approach

Amogh Kavimandan
amoghk@dre.vanderbilt.edu

Wonsuck Lee
wonsuck@research.bell-labs.com

Marina Thottan
marinat@research.bell-labs.com

Aniruddha Gokhale
gokhale@dre.vanderbilt.edu

Ramesh Viswanathan
rv@research.bell-labs.com

Institute for Software Integrated Systems

Vanderbilt University
Nashville, Tennessee
Network Simulation via Hybrid System Modeling: A Time-Stepped Approach

Agenda

• Motivation
• Related Work
• Hybrid Model
• Results
• Conclusion & Future Work
Motivation

• Problems with current simulation tools
  - Scalability: Event-based tools (ns-2) scale poorly as the network size
  - Storage and computational requirements: Storage and Computational complexity is $O(n)$, where $n$ is total number of packets.
  - Depict behavior over a large time period: Fluid/analytical models keep track of average quantities, hence can not model transient behavior in congestion control
Related Work

• Hybridization
  - Guo et al. [2000]: Consider a chunk, group of evenly-spaced packets as unit of computation, each node checked at every time step
  - Gu et al. [2004]: Integration of fluid model with packet-level simulation to give detailed traffic analysis
  - Yan et al. [1999]: Nodes treated as fluid servers which process workloads

• Hybrid Modeling
  - Bohacek [2003]: Continuous dynamic behavior of network within a Finite State Machine (FSM)
Hybrid Model (1/2)

- Hybrid Approach – use two models of computation to completely describe the system
- Model TCP (New-Reno) using Ordinary Differential Equations
- Model queues as having discrete states
  - Queue empty
  - Queue full
  - Queue neither empty nor full
- Departure rate (and hence drop rate) at each queue/router is determined by available bandwidth

A hybrid system in which TCP evolves *continuously*, while queue states are *discrete*
Network Simulation via Hybrid System Modeling: A Time-Stepped Approach

Hybrid Model (2/2)

slow-start
\[
\frac{d}{dt} w_f(t) = \frac{\log 2}{RTT_f} w_f(t)
\]
\[
s_f(t) = \frac{1.45 w_f(t)}{RTT_f}
\]

congestion-avoidance
\[
\frac{d}{dt} w_f(t) = \frac{1}{RTT_f}
\]
\[
s_f(t) = \frac{1.45 w_f(t)}{RTT_f}
\]

fast-recovery / fast-retransmit
\[
w_f(t) = w_f^\rightarrow/2 + \text{ack}(t)
\]
\[
s_f(t) = \frac{1.45 w_f(t)}{RTT_f}
\]
\[
\zeta(t) = (w_f^\rightarrow - w_f(t_{0r} - RTT_f)) + \eta(t) - H(t)
\]

if \( w_f(t) \geq 1 \), then
\[
\text{time-out over}
\]
\[
d_{f_{\text{drop}}} = \int_{t_{0r}}^{t} d_f(t') dt'
\]

if \( w_f(t) < 1 \), then
\[
\text{time-out}
\]
\[
w_f(t) = 1
\]
\[
s_f(t) = 0
\]
## Slow start Model

### Slow-start
- Congestion window increased by one on each ACK (can happen each $RTT$).
- If at $t$, window size is $w_f(t)$ rate of change is given as:

\[
\frac{d}{dt} w_f(t) = \frac{w_f(t)}{RTT_f}
\]

or

\[
\frac{d}{dt} w_f(t) = \frac{\ln mss}{RTT_f} w_f(t)
\]

$mss$ (maximum segment size) is decided before connection establishment.
### Congestion Avoidance Model

#### Congestion Avoidance (CA)
- Congestion window increased by 1 every ACK.

\[
W_{f_{\text{new}}} = W_{f_{\text{old}}} + \frac{L}{W_{f_{\text{old}}}}
\]

And the rate of change is

\[
\frac{d}{dt} w_f(t) = \frac{L}{RTT_f}
\]

where \( L \) is usually 1.
Fast Retransmit/Fast Recovery & Timeout

\textbf{fr/fr}

- Entered when a drop is detected (by way of triple dupACKs).
- Increase congestion window by 1 per ACK.

\textbf{Timeout}

- set \textit{tout} to 0 when fr/fr is entered. If it exceeds \textit{RTO} enter slow-start with congestion window = 1.

In each iteration: solve the ODE that governs TCP at that time, determine the send and drop (if any) rates at each queues and update queue sizes.
while $t < T_{end}$ do
  $t = t + h$
  wake up the sources as scheduled
  solve equations for all sources and queues
  for all sources do
    switch mode:
    case mode == slow start
      if $cwnd \geq ssthresh$ then
        mode = c.a. ($t$+delay)
      else if drop then
        mode = fr/fr ($t$+delay)
      else
        remain in slow start
    case mode == fr/fr
      if $ndrop = 0$ then
        mode = c.a.
      else
        remain in fr/fr
    case mode == c.a.
      if drop then
        mode = fr/fr ($t$+delay)
      else
        remain in c.a.
  end
Results (1/3)

- Simulation shows good match in C.A. Some mismatch in slow-start and fr/fr stage – bursty nature of ns-2 sending rate

- Hybrid variation much smoother (C.A.) - continuous evolution of TCP, lack of arrival randomness

Congestion window variation against time for ns-2 and hybrid simulation, for single source-destination pair shown above
Results (2/3)

Average queue size (hybrid:ns-2) is shown, for topology shown above

<table>
<thead>
<tr>
<th>Connections</th>
<th>Q_1</th>
<th>Q_2</th>
<th>Q_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>53</td>
<td>(23.7:24.8)</td>
<td>(21.0:22.3)</td>
<td>(8.7:10.5)</td>
</tr>
<tr>
<td>86</td>
<td>(24.4:23.8)</td>
<td>(20.5:22.5)</td>
<td>(11.8:13.0)</td>
</tr>
<tr>
<td>129</td>
<td>(28.5:28.5)</td>
<td>(13.9:16.4)</td>
<td>(21.0:20.6)</td>
</tr>
</tbody>
</table>
Results (3/3)

Q2 variation for different connection sizes. […] ns-2 and [ ] hybrid

• Q2 had highest discrepancy with ns-2 (~6-7%). Variation not significant
Conclusion & Future Work

- Demonstrated the strength of hybrid system modeling & simulation for TCP
- Extensive experiments showed good agreement with event-based simulators
- Study stability and convergence
- Support for cross-layer simulations – how disruptions at one layer affect other layers
- Investigate potential speedup of simulation by varying iterative time constant $h$