Rapid PHY Selection (RPS):
A Performance Evaluation of Control Policies

Ken Christensen
Department of Computer Science and Engineering
University of South Florida
Tampa, FL 33620
christen@cse.usf.edu
(813) 974-4761

This material is based upon work funded by the National Science Foundation under grant CNS-0520081.

Acknowledgments

- **Much of the work was done by Chamara Gunaratne as a graduate student at USF**
- **Much of the work was done in collaboration with Bruce Nordman of LBNL**

Agenda

- **Motivation and introduction**
- **Control policies**
- **Effects on higher layers**
- **Summary and future directions**

Agenda

- **Motivation and introduction**
- **Control policies**
- **Effects on higher layers**
- **Summary and future directions**

Motivation

- **Goal is *Energy Efficient Ethernet***
- **Many Ethernet links are idle most of the time**
  - This applies especially to desktop links
  - Does not apply to all Ethernet links
- **Match link rate with link utilization for efficiency**
  - Lower link rate consumes less power
  - Must always have highest link rate available if/when needed
- **Savings potential (at the wall socket)**
  - 2 to 4 W per link for 1 Gb/s versus 100 Mb/s
  - 10 to 20 W per link for 10 Gb/s versus 1 Gb/s

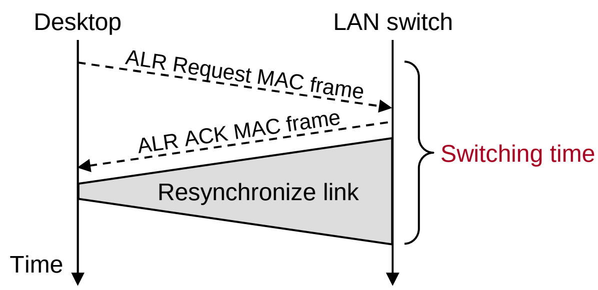
Motivation <u>continued</u>

- **Existing laptop NICs drop link rate to save power**
  - When entering sleep or off state
  - When on battery power
- **Existing auto-negotiation is used**
  - Requires 3 to 4 seconds to switch link data rate
  - Not acceptable for "real time" use
- **What is needed**
  - A fast method to switch link rate for real time use

Introduction to RPS

- **RPS is Rapid PHY Selection**
  - A possible mechanism for fast switching of link rate
  - Supported on both ends of a link

- **RPS mechanism could be a MAC frame handshake**



Introduction to RPS <u>continued</u>

- **RPS is a *mechanism* only**

- **Need a *control policy* to determine when to switch link rate**
  - Outside the scope of any possible RPS standard
  - Multiple control policies are possible
    - This can be a competitive advantage for vendors

- **We investigate <u>technical feasibility</u>**
  - The control policy determines possible energy savings

Agenda

- **Motivation and introduction**
- **Control policies**
- **Effects on higher layers**
- **Summary and future directions**

Control policy trade-off

- **Fundamental trade-off is…**

Time in low data rate versus packet delay

- **Want lowest possible packet delay?**
  - Use only highest link rate at all times
- **Want lowest possible energy use?**
  - Use only lowest link rate at all times

Seek a balance

Control policy goals

- **A good control policy is…**

1) Simple

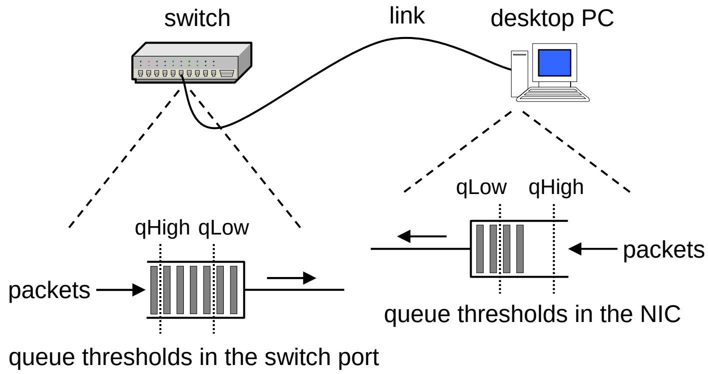
2) Responsive to changes in link utilization

3) Does not cause oscillation of link rate

- **Resulting in…**
  - Low and bounded packet delay
  - Maximized energy savings

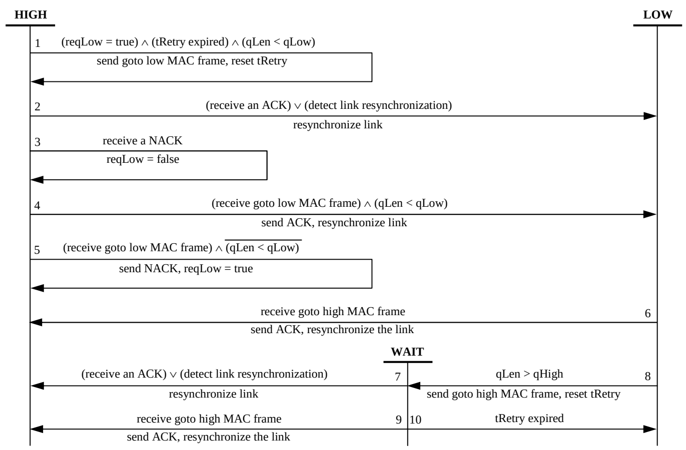
Dual-threshold policy

- **Use thresholds in output buffers**
  - If queue is above qHigh then switch to high rate
  - If queue is below qLow then switch to low rate



Dual-threshold policy continued

- **FSM for dual-threshold policy**



Dual-threshold policy continued

- **A pseudocode process description**
  - Handshake details omitted

Executes on receiving a frame…

```
if (link rate is low)
  if (buffer length greater than qHigh)
    handshake for high link rate
```

Executes on transmitting a frame…

```
if (link rate is high)
  if (buffer length less than qLow)
    handshake for low link rate
```

Dual-threshold policy <u>continued</u>

- **This policy can cause oscillation of link rate**
  - Will occur with smooth traffic at medium utilization
- **Can be dampened with timers**
  - "up" timer for high link rate
  - "down" timer for low link rate
- **But, there may be a better policy…**

Utilization-threshold policy

- **Make explicit utilization measurement part of policy**
  - Prevents drop to low rate when not warranted

Executes on receiving a frame…

```
if (link rate is low)
  if (buffer length greater than qHigh)
    handshake for high link rate
```

Executes on transmitting a frame…

```
if (link rate is high)
  if (buffer length less than qLow)
    if (measured utilization less than low target)
      handshake for low link rate
```

Performance evaluation

- **Implemented simulation model of RPS and policies**
  - Queuing model built using CSIM19
  - Used trace and synthetic traffic as input
  - Varied the link rate switching time
- **Control variables**
  - Utilization of input traffic
  - Burstiness of input traffic
  - Link rate switching time
- **Response variables**
  - Packet (or burst) delay
  - Time in low link rate

Time scales of interest

- **How long is a link idle between bursts of packets?**
- **Time the user leaves his or her PC**
  - Seconds, minutes, to hours
- **Time between data transfer bursts (active use)**
  - Milliseconds to seconds
    - For example, web surfing

Trace traffic characteristics

- **Collected packet level traces from 100 Mb/s links**
  - Note very low average utilization

Trace	Duration	Description	Avg util.
USF #1	0.5 hours	Link to "busiest" user in USF	4.11 %
USF #2	0.5	Link to 10th busiest user	2.63
USF #3	0.5	Link to an average user	0.03
PSU #1	2.0	Link to a desktop PC	0.13
PSU #2	2.0	Link connecting two switches	1.01
PSU #3	2.0	Link connecting switch to router	1.03

PSU traces are from Suresh Singh at Portland State University

Results with trace traffic

- **Simulation results for dual-threshold policy**
  - Use 1 millisecond link rate switching time

For 1 Gb/s the delays would be smaller

Trace	10 Mb/s	100 Mb/s	Delay	Time in low rate
USF #1	7.60 ms	0.09 ms	2.79 ms	99.42 %
USF #2	3.95	0.08	1.81	99.81
USF #3	196.29	0.05	1.48	99.99
PSU #1	33.51	0.18	5.63	99.99
PSU #2	2321.31	0.12	9.55	99.12
PSU #3	1147.83	0.51	4.07	99.83

Time in 10 Mb/s

Results with trace traffic <u>continued</u>

- **Simulation results for dual-threshold *with timers***
  - Use 1 millisecond link rate switching time

Lower delay is due to less oscillation

Trace	10 Mb/s	100 Mb/s	Delay	Time in low rate
USF #1	7.60 ms	0.09 ms	2.40 ms	96.91 %
USF #2	3.95	0.08	1.67	99.58
USF #3	196.29	0.05	1.25	99.98
PSU #1	33.51	0.18	5.11	99.88
PSU #2	2321.31	0.12	0.98	94.26
PSU #3	1147.83	0.51	3.16	99.45

Time in 10 Mb/s

Results with trace traffic <u>continued</u>

- **Simulation results for utilization-threshold policy**
  - Use 1 millisecond link rate switching time

Even lower delay, but less time in low rate

Trace	10 Mb/s	100 Mb/s	Delay	Time in low rate
USF #1	7.60 ms	0.09 ms	1.47 ms	78.21 %
USF #2	3.95	0.08	1.48	95.09
USF #3	196.29	0.05	1.11	99.92
PSU #1	33.51	0.18	4.76	99.80
PSU #2	2321.31	0.12	0.50	96.47
PSU #3	1147.83	0.51	2.57	98.54

Time in 10 Mb/s

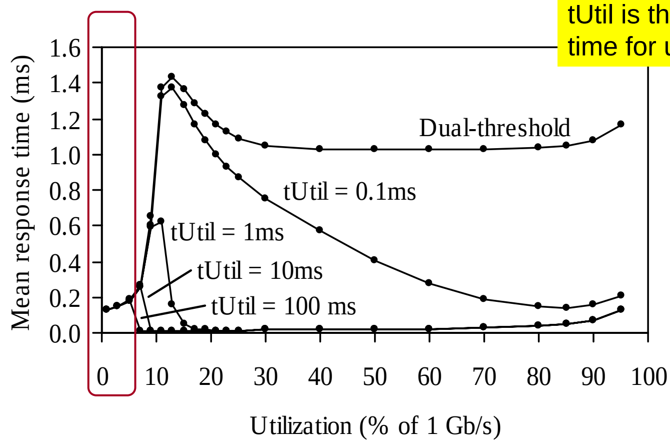
Simulation with Poisson traffic

- **Use Poisson to model "smooth" traffic**
  - Model a 1 Gb/s link
- **Link switching time is 1 millisecond**
- **For utilization-threshold policy**
  - tUtil is the time period used for sampling utilization
  - Set utilization threshold to be 5% of 1 Gb/s

Results for Poisson traffic

- **Results for both policies**
  - Dual-threshold stays high due to oscillation!

tUtil is the sample time for utilization



Simulation with bursty traffic

- **Synthetically generated 1 Gb/s bursty traffic**
  - Pareto burst size, exponential interburst time
  - Mean burst size is small (about 8.4 KB)
    - Small burst size is worst case for energy savings
- **Link switching time set to 1 millisecond**

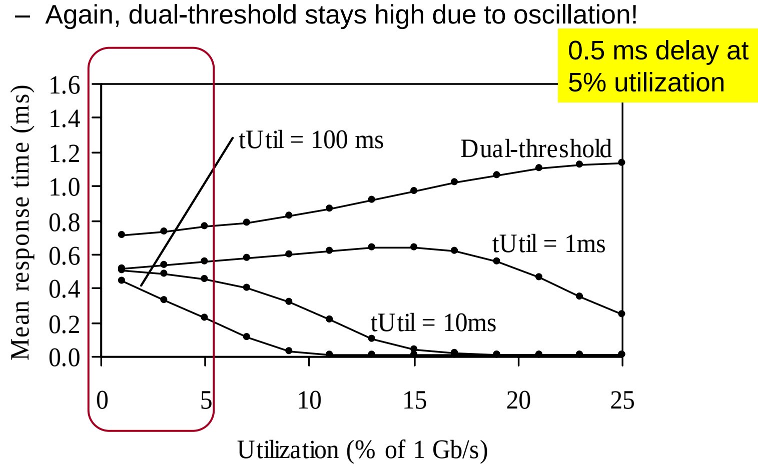
25 of 45

IEEE 802.3 EEE Study Group – January 15, 2007
Monterey, CA

USF UNIVERSITY OF SOUTH FLORIDA

Results for bursty traffic

- **Results for mean packet delay**
  - Again, dual-threshold stays high due to oscillation!



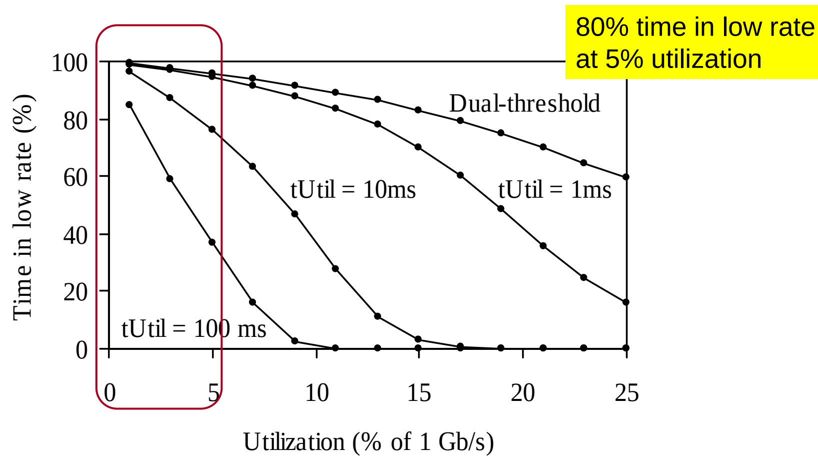
26 of 45

IEEE 802.3 EEE Study Group – January 15, 2007
Monterey, CA

USF UNIVERSITY OF SOUTH FLORIDA

Results for bursty traffic continued

- **Results for time in low data rate**



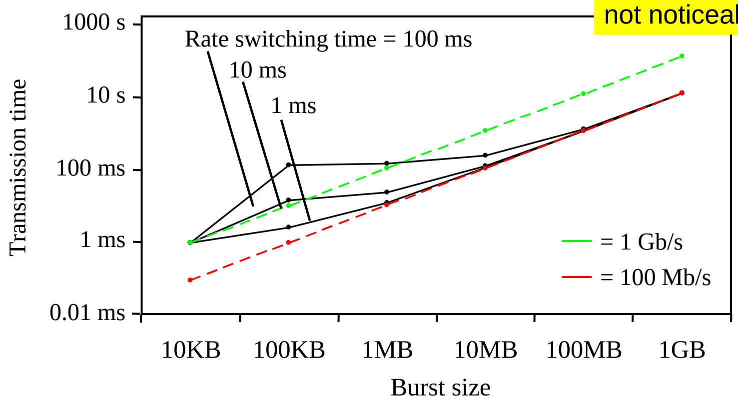
Simulation of a single burst

- **Transfer time of a single burst**
  - Vary the burst size
  - Vary the link switching time
- **At what point can RPS be noticed by a user?**

Results for transfer time of a burst

- **Results for transfer time of a single burst**
  - Note that human “just noticeable” time is 100 milliseconds

RPS is probably not noticeable



Wrap-up of control policy topic

- **Multiple control policies are possible**
  - Three have been shown here
  - Each side is independent and will interoperate
- **Trade-off in packet delay versus energy saved**
- **Performance summary:**
  - About 0.5 millisecond increase in packet delay
  - About 80% time in low data rate

Is increase in delay noticeable to a user?

Agenda

- Motivation and introduction
- Control policies
- **Effects on higher layers**
- Summary and future directions

Including humans (users)

31 of 45

IEEE 802.3 EEE Study Group – January 15, 2007
Monterey, CA

USF UNIVERSITY OF SOUTH FLORIDA

Effects of RPS on users

- Users will disable power management if annoying
- Yet… some users may tolerate slight annoyance in order to be “green”
- RPS with suitable policy should be entirely invisible
  - To *most* users

32 of 45

IEEE 802.3 EEE Study Group – January 15, 2007
Monterey, CA

USF UNIVERSITY OF SOUTH FLORIDA

Effects of RPS on users continued

- **Need to consider**
  - Effect of increased packet delay
  - If packet loss can occur (and its effect) due to buffer overflows

- **Human perception time is about 100 milliseconds**
  - Delays below this threshold are unnoticeable

Effects of RPS on desktop applications

- **Data transfer applications**
  - Web surfing
  - File downloading

- **Playback streaming application**
  - YouTube and etc.

- **Realtime streaming applications**
  - VoIP telephony

Try it!

Surprising result: Video players can buffer about six seconds of video in the first second of play. So even a four second link outage causes *no* effect.

Effects of RPS on TCP/IP

- **Will a switching delay cause packet loss?**
- **What is the effect of packet loss?**
- **We have some preliminary work**
  - Using the ns-2 simulator

35 of 45

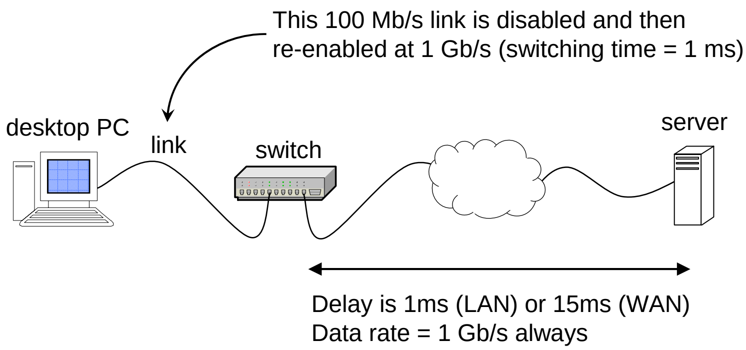
IEEE 802.3 EEE Study Group – January 15, 2007
Monterey, CA

USF UNIVERSITY OF SOUTH FLORIDA

Simulation of TCP/IP

Preliminary results

- **Use ns2 simulator to evaluate effects of RPS on TCP**
  - Study LAN and WAN environments
  - FTP between two nodes via an intermediate node
  - All default except `window_` set to 1000 (default is 20)



36 of 45

IEEE 802.3 EEE Study Group – January 15, 2007
Monterey, CA

USF UNIVERSITY OF SOUTH FLORIDA

Simulation of TCP/IP continued

Preliminary results

- **Important note on ns2**

All buffers are destroyed in an ns2 model when a link is disabled/enabled. So, results here are much worse than realistic.

We plan to develop a realistic model.

Simulation of TCP/IP continued

Preliminary results

- **ns2 LAN environment**
  - Shows TCP congestion window (transition at Time = 5 s)

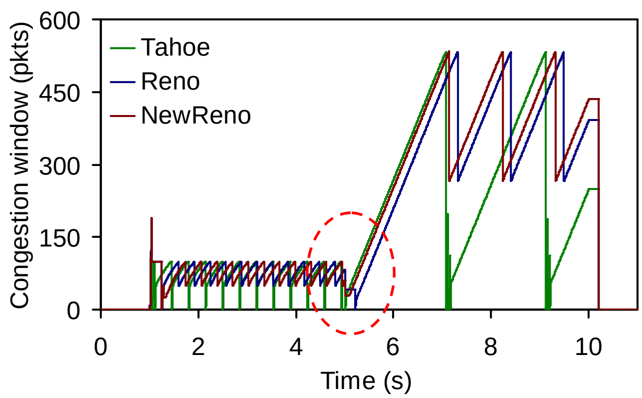


fig7.xls

Simulation of TCP/IP continued

Preliminary results

- **ns2 WAN environment**
  - Shows TCP congestion window

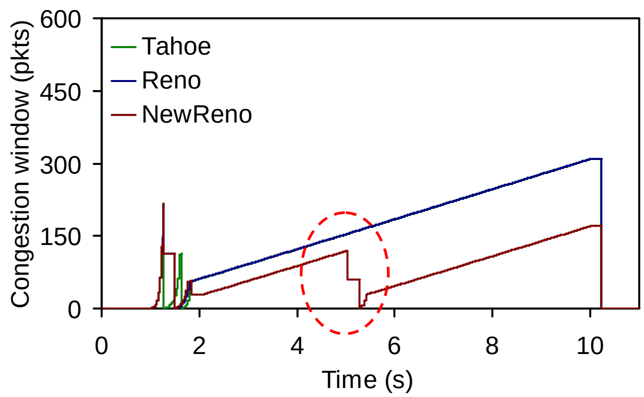


fig8.xls

Simulation of TCP/IP continued

Preliminary results

- **ns2 WAN environment**
  - Shows TCP congestion window (with 10 ms switching time)

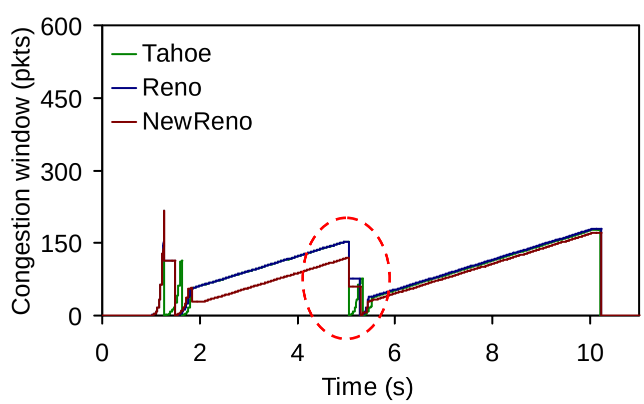


fig9.xls

Agenda

- **Motivation and introduction**
- **Control policies**
- **Effects on higher layers**
- **Summary and future directions**

Summary

- **Simple RPS control policies are possible**
- **Utilization-threshold control policy works well**
  - Added packet delay is very small
  - Energy savings can be considerable
- **Energy efficiency should be invisible**
  - Can be achieved with good RPS control policy

Future work

- **Further work is possible in control policies**
  - Context driven
  - Adaptive based on user input
- **Further work is possible in evaluating RPS effects**
  - Build a h/w device to "kill" link to emulate RPS switching
- **Currently experimenting with a "Soft-ALR"**
  - ALR = Adaptive Link Rate
  - ALR is RPS + control policy
  - Use auto-negotiation to switch link rate via a Linux script
    - Experience a 4 second link switch delay

Future work continued

- **Soft-ALR is now on SourceForge**

By Matt Landau (USF student)

The End

- **Any questions?**

- **Much of the material in this talk comes from…**

C. Gunaratne, K. Christensen, B. Nordman, and S. Suen, "Reducing the Energy Consumption of Ethernet with Adaptive Link Rate (ALR)," submitted to *IEEE Transactions on Computers* in January 2007.

C. Gunaratne, K. Christensen, and S. Suen, "Ethernet Adaptive Link Rate (ALR): Analysis of a Buffer Threshold Policy," *Proceedings of IEEE GLOBECOM 2006*, November 2006.

C. Gunaratne and K. Christensen, "Ethernet Adaptive Link Rate: System Design and Performance Evaluation," *Proceedings of the 31st IEEE Conference on Local Computer Networks*, pp. 28-35, November 2006.