CHEETAH (Circuit-Switched High-speed End-to-End Transport Architecture)

Malathi Veeraraghavan
Univ. of Virginia
mv@cs.virginia.edu

- Outline
  - Research goals
  - Topology & equipment
  - Schedule & Status
  - Extension of work

Team & Acknowledgment

- Team (PI/Co-PIs):
  - Malathi Veeraraghavan, Univ. of Virginia
  - Nagi Rao, Bill Wing, Tony Mezzacappa, ORNL
  - John Blondin, NCSU
  - Ibrahim Habib, CUNY
- UVA funding sources:
  - NSF EIN grant ANI-0335190
  - NSF ITR small grant ANI-0312376
  - DOE FG02-04ER25640

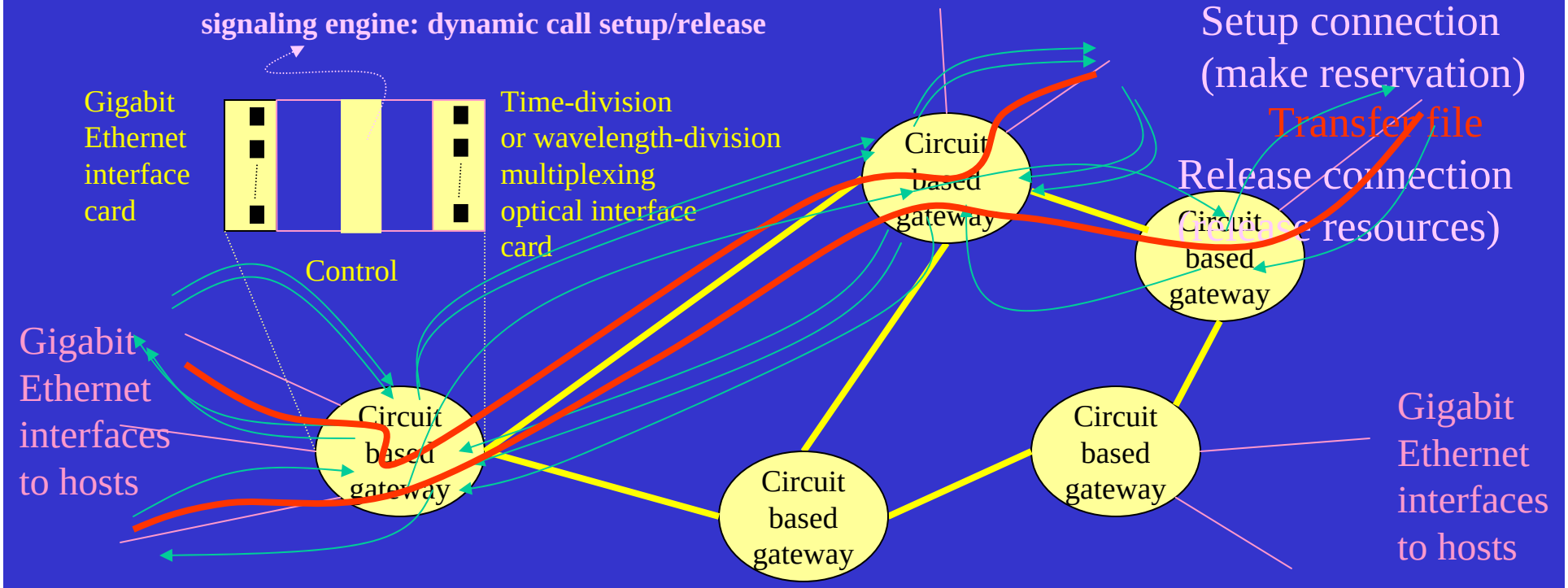
UVA team members

- Postdoc: Xuan Zheng
- Graduate students
  - Xiuduan Fang
  - Zhanxiang Huang
  - Anant P. Mudambi
  - Xiangfei Zhu

Research goals

- Deploy a shared circuit-switched network to complement connectionless IP service
  - establish, use & release, rate-guaranteed high-bandwidth host-to-host circuits
  - support TSI (Terascale Supernova Initiative) applications
    - large file transfers
    - remote visualization
    - remote computational steering
  - leverage equipment that supports protocols for the creation of distributed (large-scale) shared circuit-switched networks
- Extend solution:
  - to a connection-oriented internet
  - to partial connections (impt.: setup triggered from end hosts)

Primary research goal: Highly dynamic circuits (held for only for ms-to-sec/min for file transfers) (requests for circuits generated by file transfer applications)

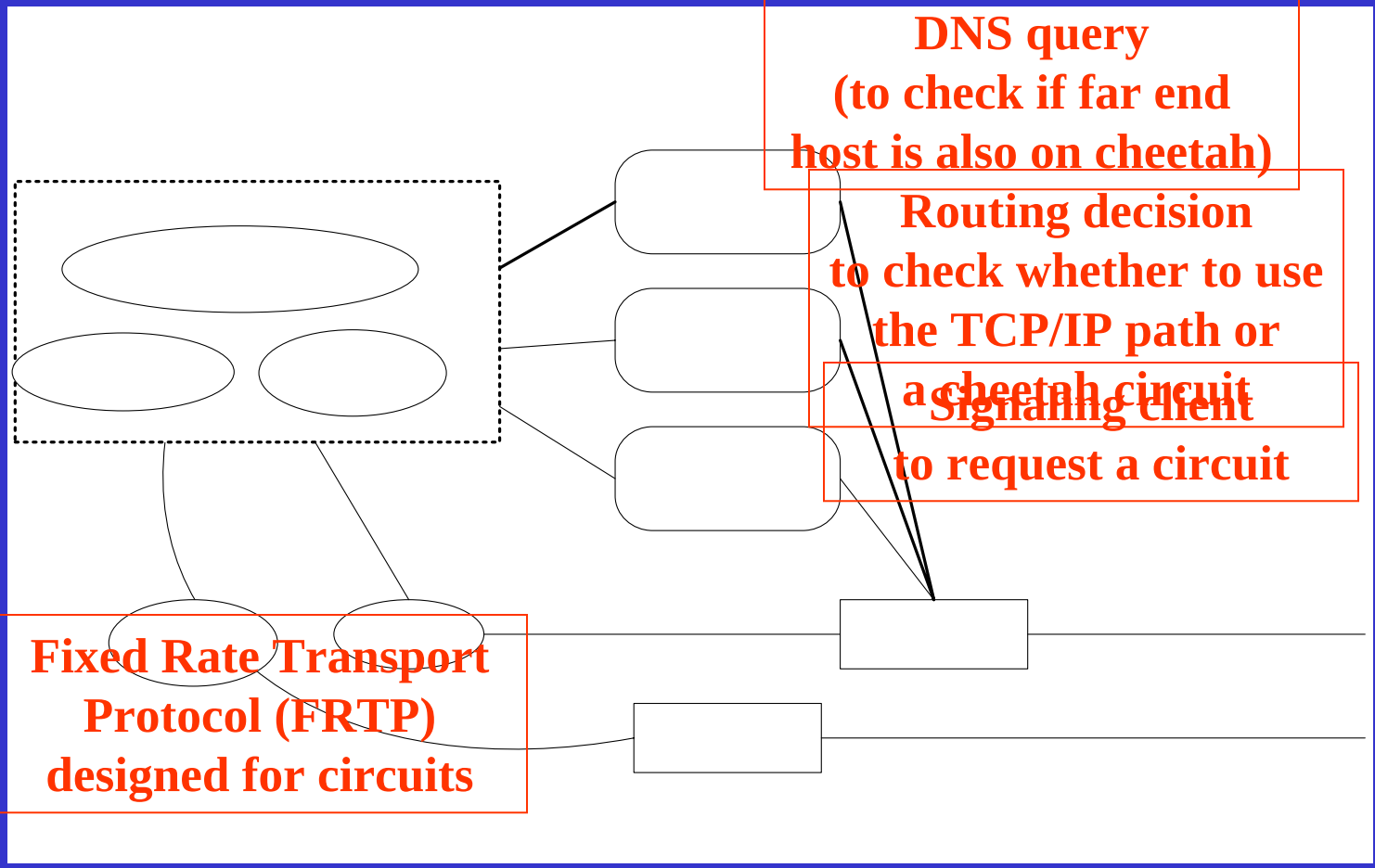


- Gateways available that can crossconnect a Gigabit Ethernet port to an equivalent-rate time-division or wavelength-division multiplexed signal dynamically

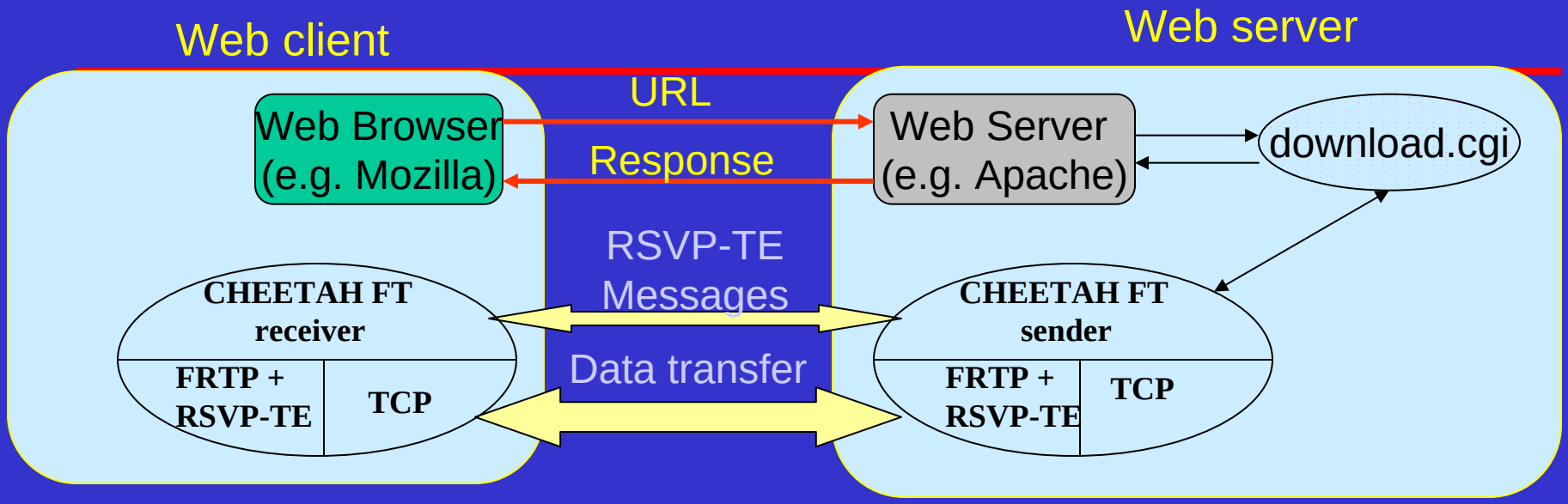
Strategy

- Leverage the presence of connectionless IP services (host-to-host IP connectivity; DNS)
- Use off-the-shelf circuit-based gateways
  - that support GMPLS routing and signaling protocols for such dynamic circuit setup/release
- Implement cheetah software to run on end hosts
- Integrate with host applications that will create a large offered load for circuits

Cheetah software on end hosts



Demo #1 (at SC2004): Web Application

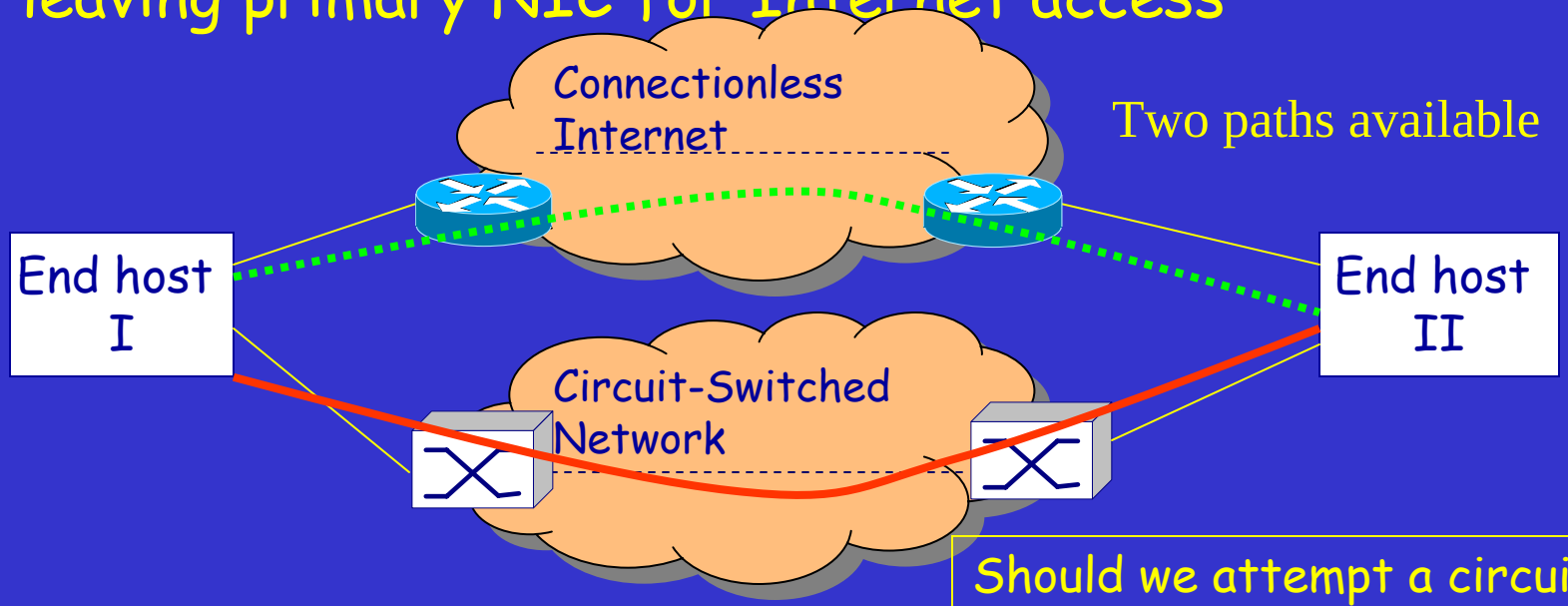


- At the web server side
  - Hyperlink to file is a CGI script (download.cgi); filename embedded in hyperlink
  - Download.cgi is started automatically at server when user clicks hyperlink, which triggers CHEETAH FT sender
  - CHEETAH FT sender initiates CHEETAH circuit setup by calling RSVP-TE client.
- At the web client side
  - A CHEETAH FT receiver is running as daemon to receive the user data

Goal: Make usage of circuit-switched network seamless to user

Tactics: Lower the crossover file size above which circuits make sense

- Use second NICs at hosts for circuit connectivity leaving primary NIC for Internet access



- Attempt circuit setup
- If rejected, fall back to using TCP/IP

Should we attempt a circuit setup for ALL file transfers?

Or is there a crossover file size below which we use the TCP/IP network and above which we attempt a circuit setup?

Two metrics to decide the crossover file size

- Delay
- Utilization

Delay metric

- Need measurements for both paths:
  - TCP/IP path: packet loss rate, bottleneck link rate and RTT
  - circuit: call-blocking prob. and setup delay
- For most regions of operation,
  - in wide-area scenarios
    - attempt a circuit setup even for very small files
  - in local-area scenarios
    - if link rate is 100Mbps
      - crossover file size: 100's of KBs to ones of 1MBs
    - if link rate is 1Gbps
      - crossover file size: 1s to 10s of MBs

Utilization metric

- Two opposing factors
  - If the crossover file size χ is increased
    - per-circuit utilization increases
      - transfer time should exceed call setup delay
    - traffic load decreases because fewer files will exceed χ
      - aggregate utilization decreases

Aggregate utilization u_a

(simple ErlangB formula)

$$u_a = \frac{(1-P_b)\times\rho}{m}, \text{ where } P_b = \frac{\rho^m / m!}{\sum_{k=0}^{m} \rho^k / k!}$$

ρ: traffic load

m: number of circuits

P_b: call blocking probability

Assuming file size follows Pareto distribution

- Define fractional offered load

For a 1% call blocking probability $P_b = 0.01$

ρ	m	u_a
1	4	24.8%
10	17	58.2%
100	117	84.6%

$$\rho' = \frac{\rho}{E(X)} P(X \geq \chi) E[X \mid X \geq \chi] = \rho\left(\frac{k}{\chi}\right)^{\alpha-1}$$

χ	ρ′ (fraction of ρ)
40KB	81%
330KB	71%
80MB	51%

In other words

- To make the circuit-switched network a truly shared network ("dynamic")
  - many ongoing short-lived calls
  - ErlangB model works best with large loads and high m
- Why share?
  - a solution to offer rate-guaranteed connections w/o intense sharing on a call-by-call basis will be expensive
- GMPLS signaling implementations at the switches
  - manage bandwidth of the switch interfaces by implementing a simple "complete sharing" model
    - dole out bandwidth until you run out and then reject calls
  - works well when the number of circuits that can be doled out per interface is large AND holding times are short

Other apps besides file transfers for such shared circuit-switched networks

- Delay/jitter-sensitive applications like high-quality video-telephony, Internet games
  - lower-rate but longer-lived (still in minutes)
    - average telephone call
    - multiple cameras in a casual office/home setting
  - chose SONET switches to get low granularity (MPLS is a fine choice too)

Problems with this vision

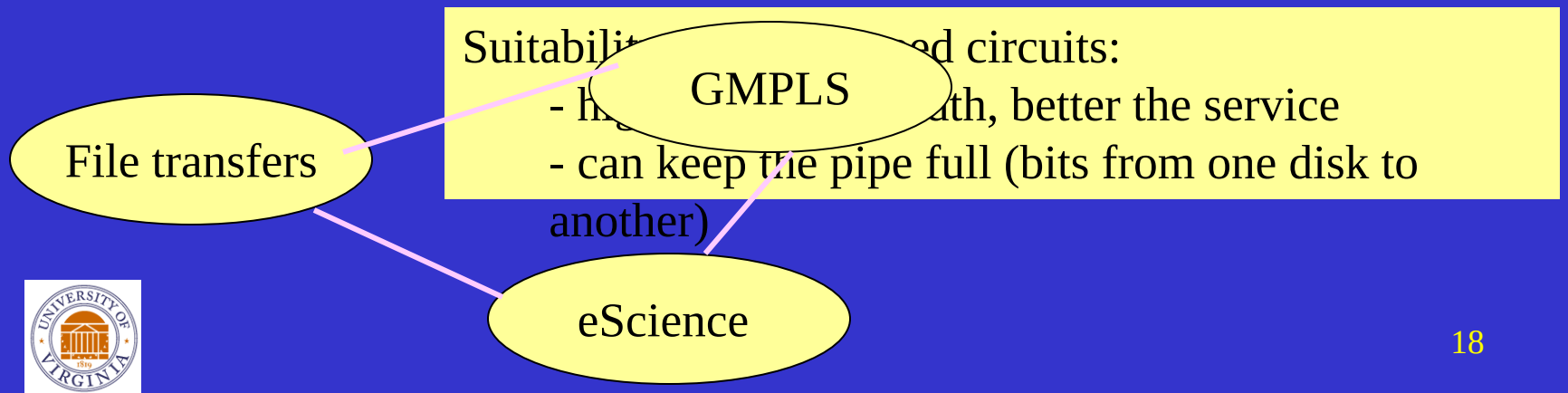
- Problem 1:
  - Fear on the part of network operators to enable user-application triggered call setups
- Problem 2:
  - eScience apps, the driver for these networks, pose clashing requirements

Problem 1: the operator's fear

- Of allowing end host applications to generate signaling requests for rate-guaranteed connections
  - Some strange host sends RSVP-TE messages and ties up bandwidth
    - Have IPsec devices on control-plane ports
    - Enforce authentication and integrity of RSVP-TE messages
  - An application could set up a circuit and hold forever
    - One answer: billing mechanism (commercial)
    - Research and education context:
      - Write RSVP-TE client software at end hosts to limit holding time
        - » similar to the concept of MTU
        - » disallow holding circuits for very long durations
        - » rejoin queue and compete for bandwidth
    - Ideal: add holding time parameter to RSVP-TE and enforce at switches

Problem 2: clashing eScience application needs

- Long-lived vs. short-lived calls
  - A scientist would like to hold a high-bandwidth circuit for 2-3 hours
    - for remote visualization
    - a 1TB file move on an end-to-end 1Gbps circuit takes ~2.3 hours



Limits sharing

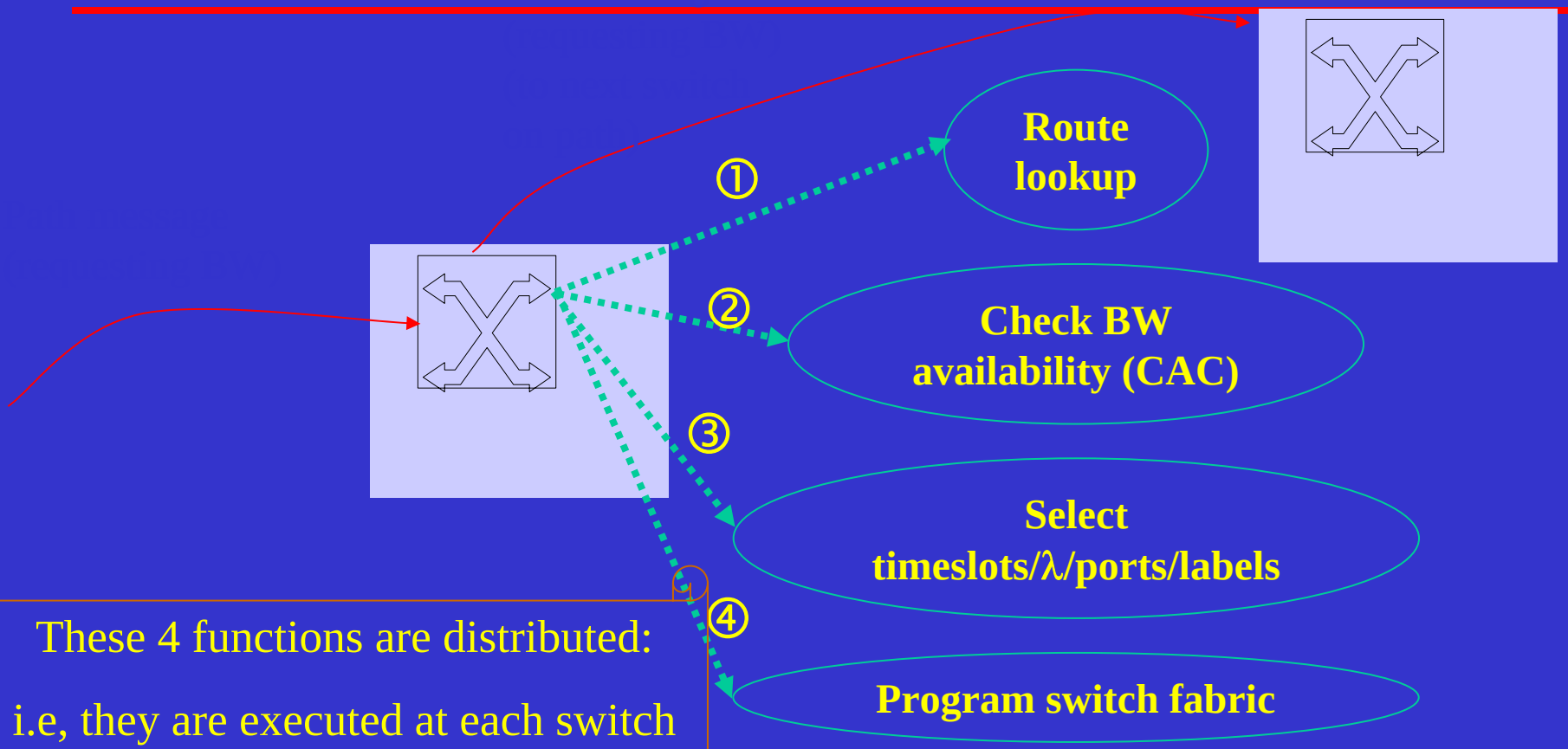
- For long-lived calls
  - Cannot use complete-sharing bandwidth management engines
  - Mean waiting time related to mean holding time
- Example:
  - The first 10 calls, each for 1Gbps, fill up a 10GbE link and everyone else waits
- Need software to support advance reservations

Pushing us backwards into non-scalable networks (centralized contol)

- Current GMPLS standards/implementations do not support bandwidth management into the future
- Hence, external schedulers being written to manage bandwidth into the future
- Negates goals of the triumvirate of GMPLS protocols
  - LMP to discover neighbors
  - OSPF-TE for routing
  - RSVP-TE for signaling

  which is to create large-scale networks by having each switch have control-plane software to manage its own bandwidth and respond to requests for reservations

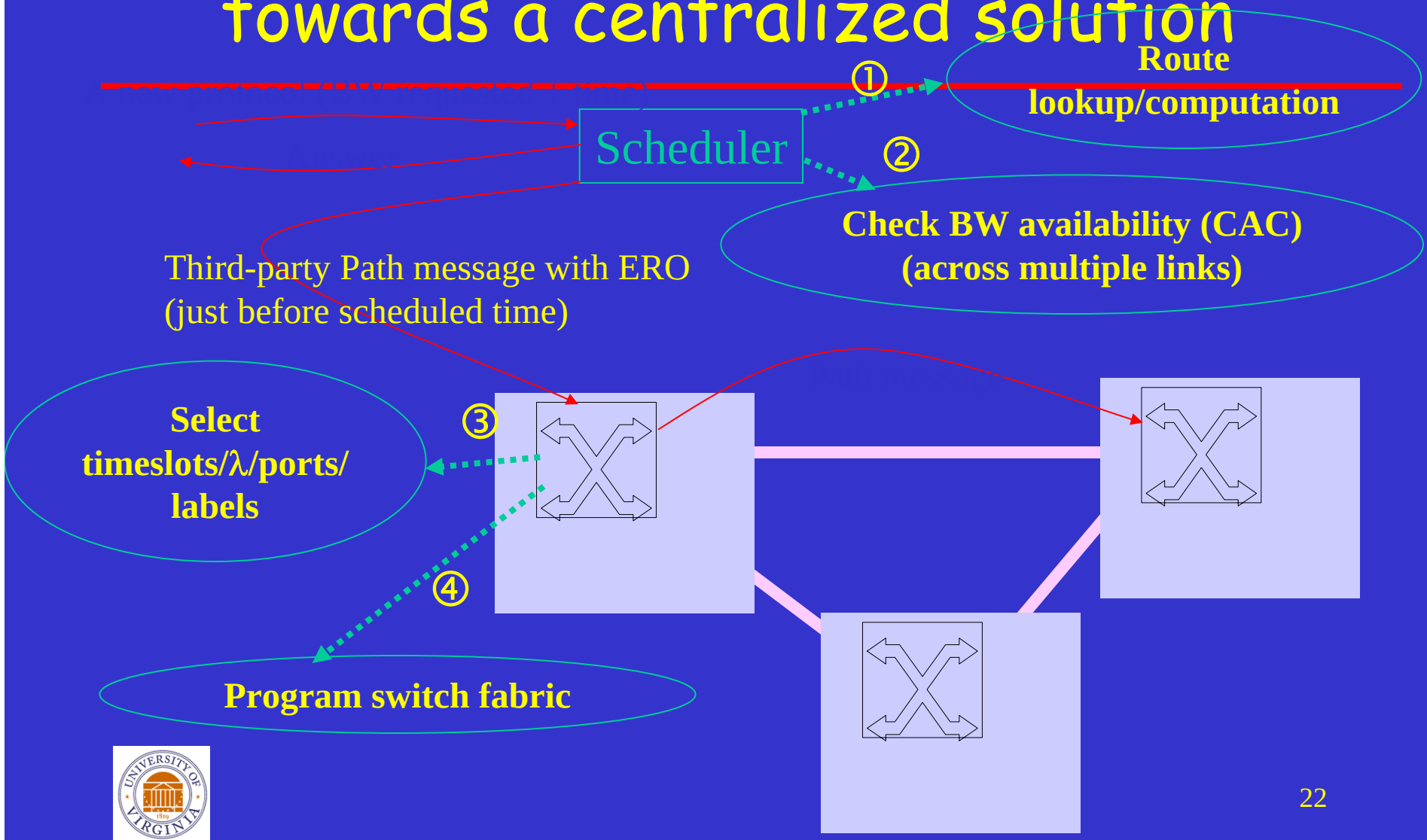
Steps executed by a GMPLS signaling engine at a switch for call setup



These 4 functions are distributed:

i.e, they are executed at each switch

Takes us partially "backwards" towards a centralized solution



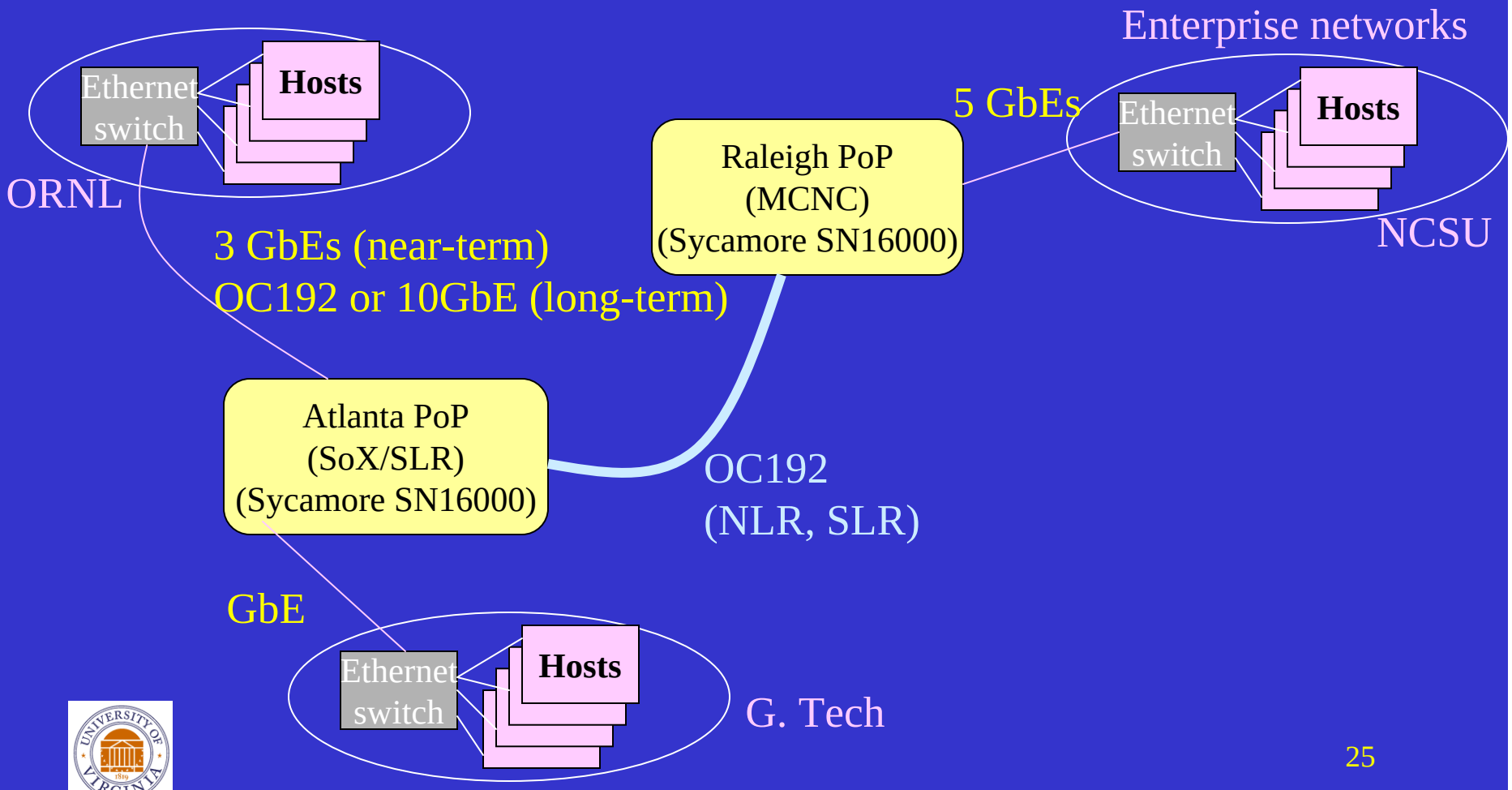
Reconcile these differences

- Partitioning bandwidth managed by external scheduler vs. bandwidth managed by GMPLS on-switch engines
  - will lead to inefficiencies
- Important to really connect this main driver (eScience) with GMPLS scalable solution
- Extend GMPLS to manage bandwidth over a period into the future (like airline reservation systems)

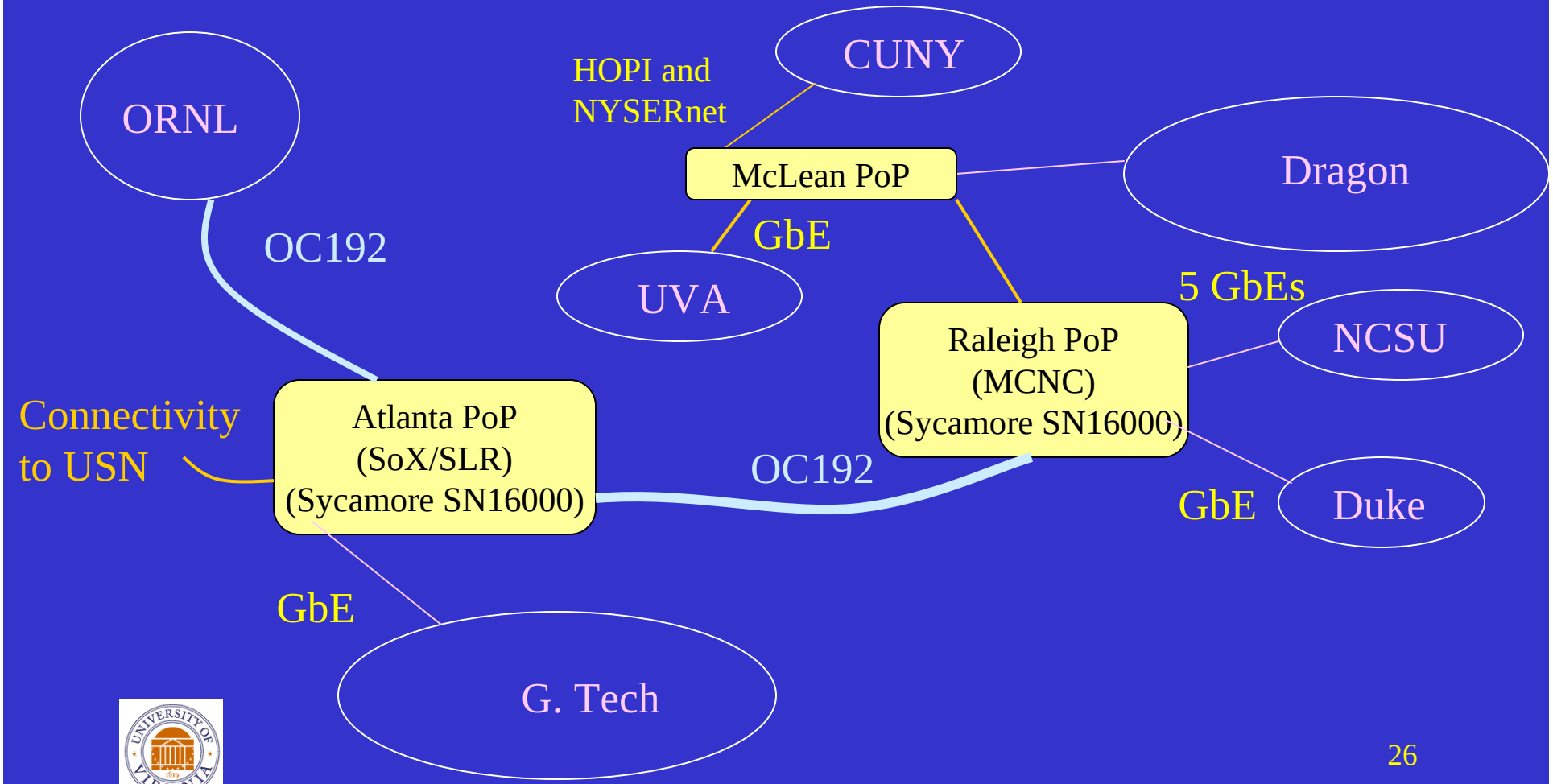
Status check

- Outline
  - Research goals
  - ➢ Topology & equipment
  - Schedule & status
  - Extension of work

CHEETAH Topology & equipment (May 1)



CHEETAH
Topology & equipment (year-end)



Status

- SN16000 switches purchased and installed (tested)
- NCSU – MCNC and MCNC – SOX/SLR connections up (will be lit May 1)
- SOX/SLR – ORNL MPLS tunnels up
- Cheetah software
  - RSVP-TE client, FRTP, Cheetah FT sender/receiver ready
- Applications
  - Web integration done
  - Testing PVFS/GridFTP
  - Soon to test Ensight remote viz. tool with cheetah

Need to extend CHEETAH concept and apps to connection-oriented internet

- As plainly evident in the costs of creating a pure cheetah network
  - used VLANs in one segment
  - used MPLS tunnels in another
- Large numbers of engineers, salespeople and network admins
  - should expect multiple connection-oriented network solutions in the near-term
    - Packet-switched:
      - MPLS
      - VLAN capability in Ethernet switches
    - Circuit-switched:
      - WDM
      - SONET

Enabling not building

- ALREADY DEPLOYED; Just flick the switch!
  - Let user apps. peel out MPLS tunnel bandwidth as needed and put back to IP path when done
- Protocol interworking
  - User (data) plane - DONE
  - Signaling protocols (for connection setup/release)
  - Routing protocols (for reachability, topology, loading data dissemination)

Summary

- Rich new set of research problems
- Experimental challenges a plenty!
- Real opportunity to deploy a connection-oriented internetwork!
- Web site: http://cheetah.cs.virginia.edu

Backup slides

Mean TCP delays (using U. Mass model)

Cases	Packet loss rate	Bottleneck link rate	Round- trip prop. delay	Mean delay for a 1GB file (s)
Case 1	0.0001	100 Mbps	0.1ms	82.25
Case 2			5ms	89.45
Case 3			50ms	396.5
Case 4	0.0001	1Gbps	0.1ms	8.25
Case 5			5ms	39.6
Case 6			50ms	395.7
Case 7	0.001	100Mbps	0.1ms	82.93
Case 8			5ms	135.4
Case 9			50ms	1293
Case 10	0.001	1Gbps	0.1ms	8.64
Case 11			5ms	129.4
Case 12			50ms	1287
Case 13	0.01	100 Mbps	0.1ms	92.41
Case 14			5ms	471.7
Case 15			50ms	4417
Case 16	0.01	1Gbps	0.1ms	12.43
Case 17			5ms	441.7
Case 18			50ms	4387

Impact of propagation delay

Low impact of bottleneck link rate in wide-area networks

Impact of packet loss rate

Should the application attempt a circuit setup or not?

- Mean delay if a circuit setup is attempted

$$E[T_{cheetah}] = (1 - P_b)(E[T_{setup}] + T_{transfer}) + P_b(E[T_{fail}] + E[T_{tcp}])$$

P_b: call blocking probability in the circuit-switched network

If circuit setup fails, fall back to Internet path

Routing decision

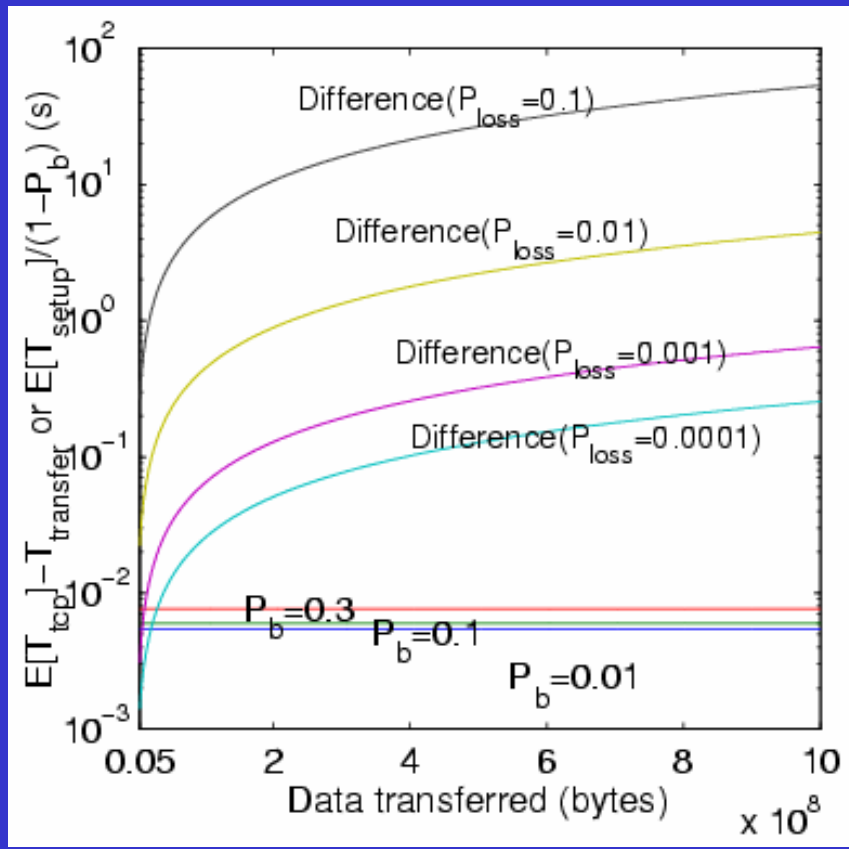
$$\text{if } \left(E[T_{cheetah}] \geq E[T_{tcp}]\right) \text{ use the TCP/IP path}$$

$$\text{if } \left(E[T_{cheetah}] \leq E[T_{tcp}]\right) \text{ attempt circuit setup}$$

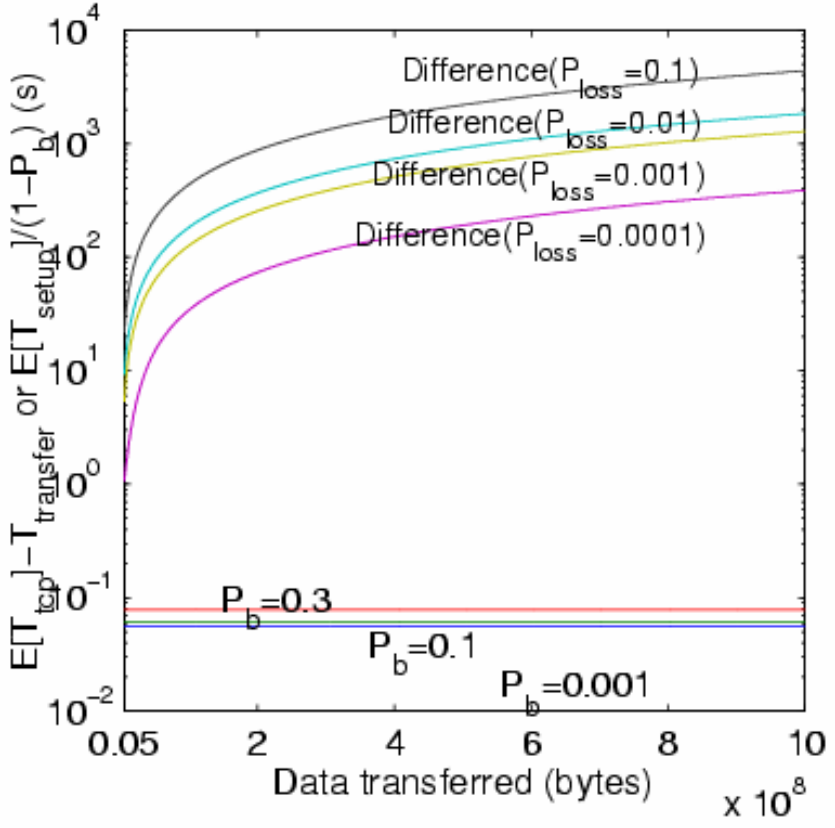
$$\text{if } \left(\frac{E[T_{setup}]}{1-P_b} \geq E[T_{tcp}] - T_{transfer}\right) \text{ use the TCP/IP path}$$

$$\text{if } \left(\frac{E[T_{setup}]}{1-P_b} \leq E[T_{tcp}] - T_{transfer}\right) \text{ attempt circuit setup}$$

Numerical results
link rate = 1Gbps



T_{prop} = 0.1ms



T_{prop} = 50ms

Crossover file sizes

When r_c = 100Mbps and T_{prop} = 0.1ms

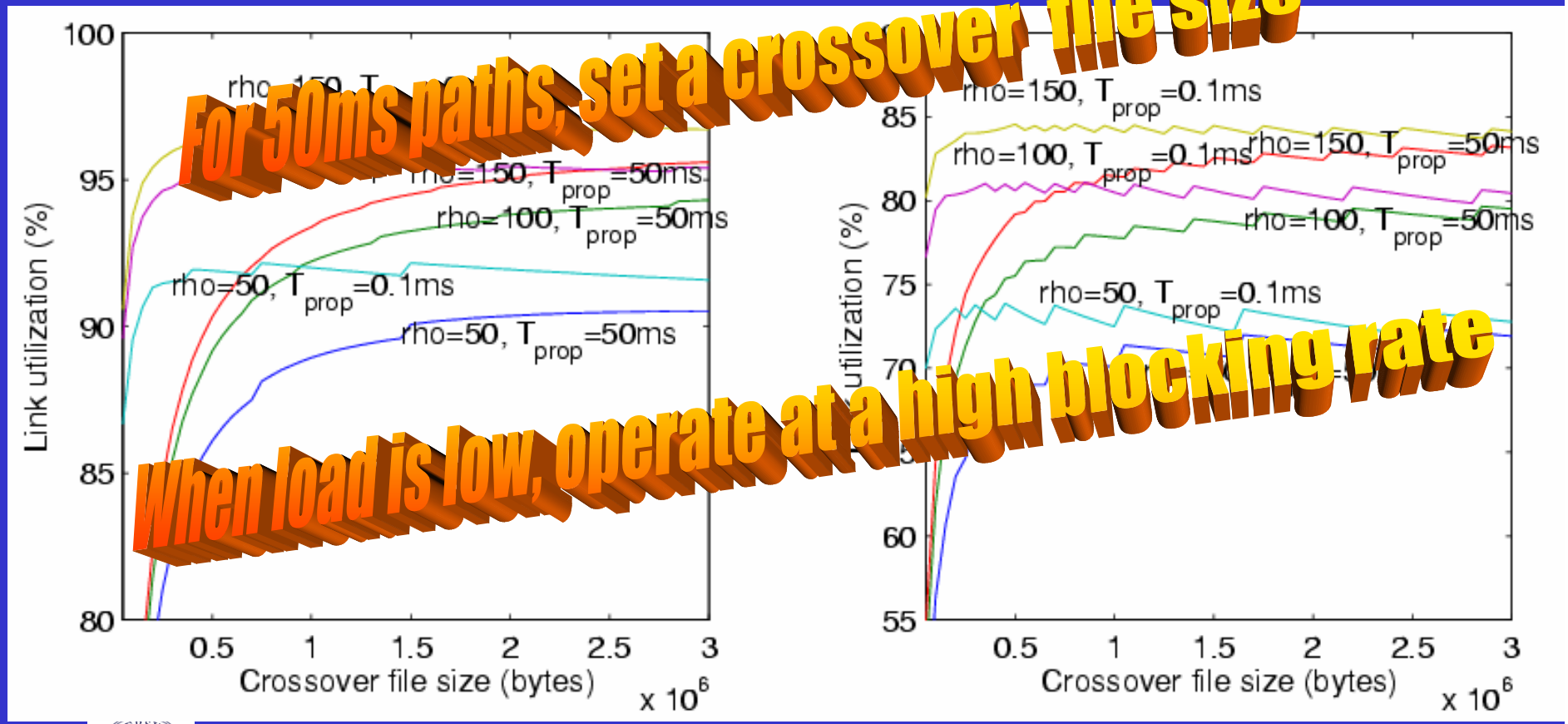
Measure of loading on: ckt. sw. network / TCP/IP path	Number of switches on the path k = 4		Number of switches on the path k = 20		
	P_b = 0.01	P_b = 0.1	P_b = 0.01	P_b = 0.1	P_b = 0.3
P_{loss} = 0.0001	610KB		2MB	2.65MB	3.4MB
P_{loss} = 0.001		KB	2MB	2.2MB	2.8MB
P_{loss} = 0.01	140KB	180KB	500KB	550KB	650KB

When Tprop = 50ms, in most regions of operation of the two networks, attempt a circuit

r_c = 1Gbps,
T_{prop} = 0.1ms

Measure of loading on: ckt. sw. network / TCP/IP path	P_b = 0.01	P_b = 0.1	P_b = 0.3
P_{loss} = 0.0001	22MB	24MB	30MB
P_{loss} = 0.001	9MB	10MB	12MB
P_{loss} = 0.01	1.2MB	1.4MB	1.8MB

Plot of utilization u with r_c= 100Mbps, k=20



P_b=0.3 P_b=0.01