Workshop on New Visions for Large-Scale Networks: Research & Applications
Vienna, VA, USA, March 12-14, 2001

The Future of High-Performance Networking

(The 5?, 10?, 15? Year Outlook)

Wu-chun Feng

feng@lanl.gov
http://home.lanl.gov/feng

Research & Development in Advanced Network Technology (RADIANT)
Computer & Computational Science (CCS) Division
Los Alamos National Laboratory
and
Department of Computer & Information Science
The Ohio State University

Outline

- History of Networking in High-Performance Computing (HPC)
  - TCP in HPC?
  - Current Solutions for HPC
- Future of High-Performance Networking: TCP vs. ULNI
  - The Road to a HP-TCP
    - What's Wrong with TCP?
    - Solutions?
- Crossroads for High-Performance Networking

Los Alamos National Laboratory RADIANT Research and Development in Advanced Network Technology

TCP for High-Performance Computing?

- Problems with TCP for HPC in the Mid-1990s   Today & Tomorrow
  - Computing Paradigm: *Cluster or supercomputer.*   + *computational grid*
  - Network Environment: *System-area network (SAN).*   + *wide-area network (WAN)*
  - ~~TCP Performance (mid-90s): *Latency: O(1000 µs). BW: O(10 Mb/s).*~~   Too heavywgt
  - TCP Performance (today): *Latency: O(100 µs). BW: O(500 Mb/s).*
  - TCP Performance (optimized): *Latency: 95 µs. BW: 1.77 Gb/s. [Trapeze TCP]*
- Solution   ➢ Problem: ULNIs do *not* scale to WAN. Must use TCP (despite conflicts).
  - User-level network interfaces (ULNIs) or OS-bypass protocols
    - Active Messages, FM, PM, U-Net. Recently, VIA (Compaq, Intel, µsoft)
  - ULNI Performance (mid-90s): *Latency: O(10 µs). BW: O(600-800 Mb/s).*
  - ULNI Performance (Reference: Petrini, Hoisie, Feng, Graham, 2000.)
    - *Latency: 1.9 µs.*   *BW: 392 MB/s = 3.14 Gb/s.*
    - *User-Level Latency: 4.5 µs.*   *User-Level BW: 307 MB/s = 2.46 Gb/s.*

Latency & BW problems *not* specific to HPC

- Medicine – M. Ackerman (NLM).
  - ECG: 20 GB *now* @ 100% reliability! Neurological: Smoothness @ 80% ok.
- Remote collaboration & bulk-data transfer – R. Mount (SLAC). 100 GB → 22 hrs.
- Integrated multi-level collaboration – T. Znati (NSF/ANIR).

Current Solutions for HPC

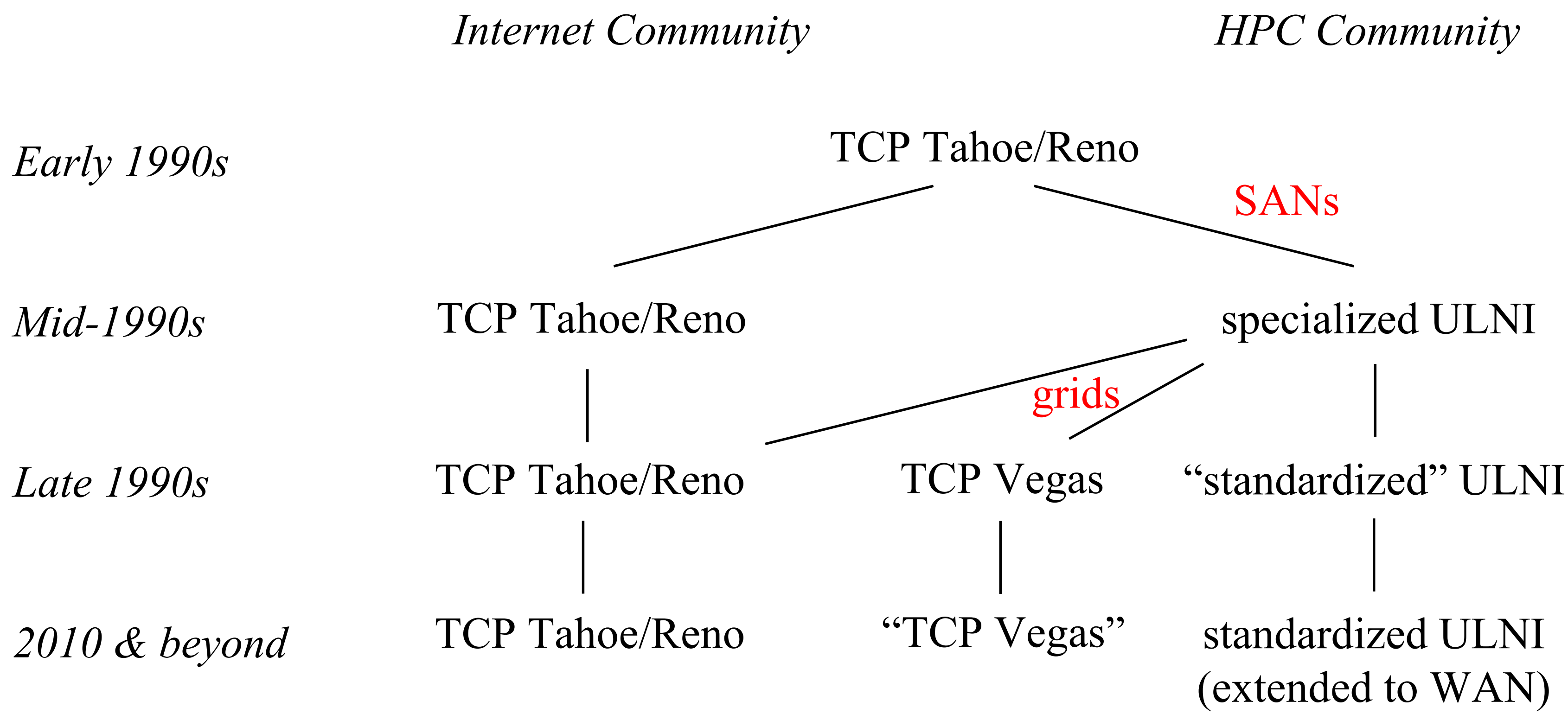
- Computational Grid (WAN) — Cost-effective for commodity, "high-end" supercomputing. Better fault tolerance than large-scale supercomputers.
  - TCP
    - \+ Infrastructure to deal with routing (ARP, IP).
    - o Congestion control (conflict between Internet & HPC communities).
    - – Non-adaptive flow control.
    - – High latency (even over short distances) and low bandwidth.

    } "Re-inventing" ULNI
- Cluster Computing or Supercomputing (SAN)
  - User-Level Network Interface (ULNI) or OS-Bypass Protocol
    - \+ Negotiated flow control.
    - \+ Low latency and high bandwidth.
    - – No automatic infrastructure to deal with routing.
    - – No congestion control.

    } "Re-inventing" TCP
- Each community is working to address the negatives.
  - Will each community continue as separate thrusts? Sociologically? Technically?

Los Alamos National Laboratory RADIANT Research and Development in Advanced Network Technology

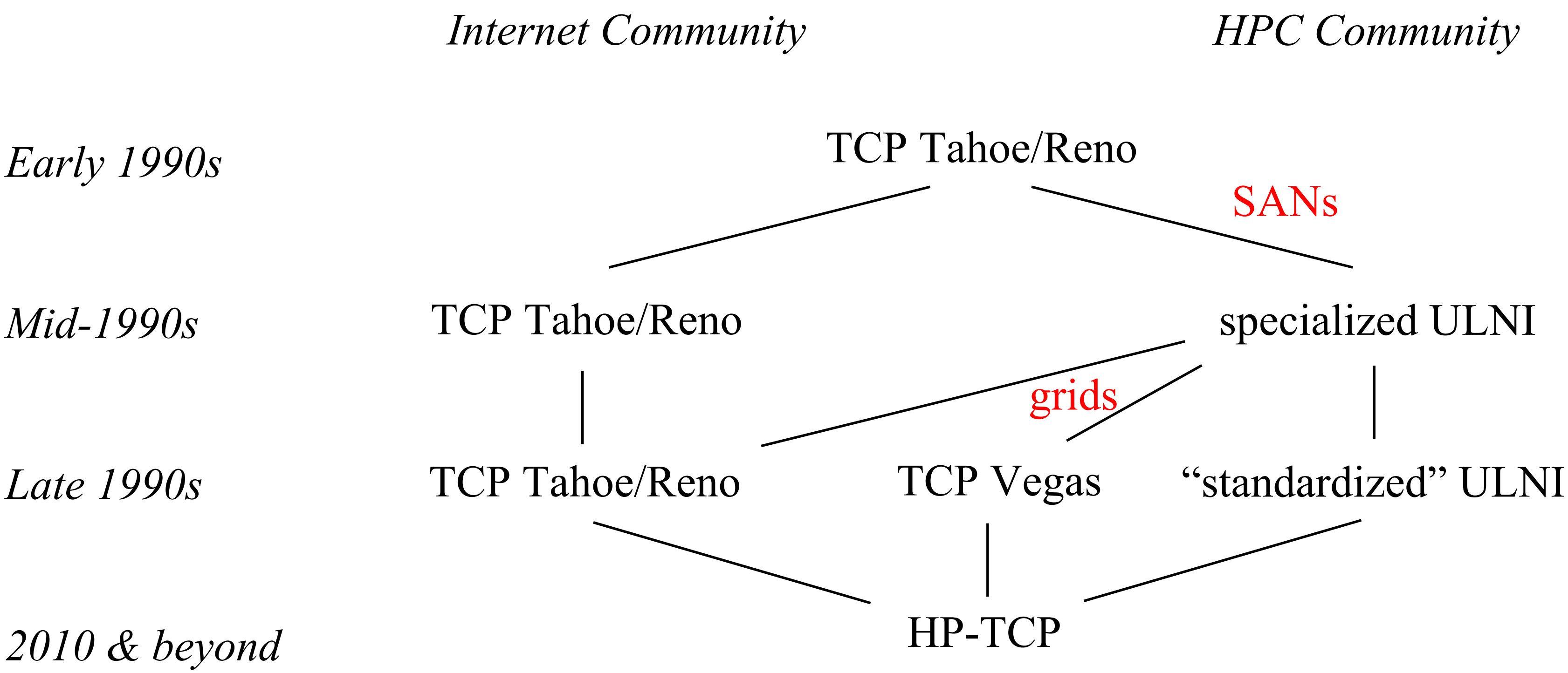
Future of High-Performance Networking



+ active queue management in network
(to ensure fairness)

Los Alamos National Laboratory
RADIANT
Research and Development in Advanced Network Technology

Future of High-Performance Networking



+ active queue management in network
(to ensure fairness)

The Road to a HP-TCP

10GigE packet interarrival: 1.2 µs
Null system call in Linux: 10 µs

What's Wrong with TCP?

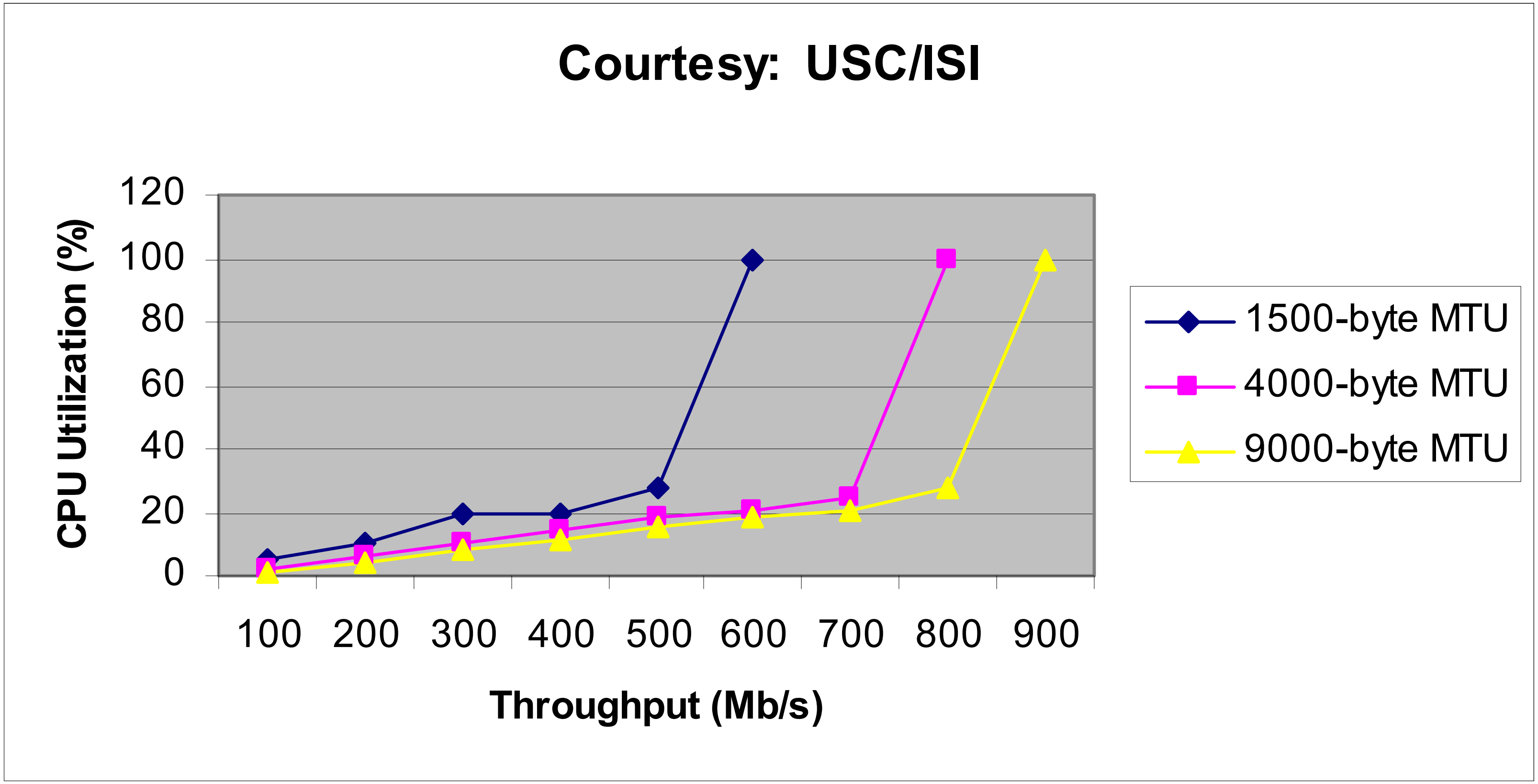
- Host-Interface Bottleneck
  - Software
    - Host can only send & receive packets as fast as the OS can process them.
      - Excessive copying.
      - Excessive CPU utilization.
  - [Hardware (PC) *Not anything wrong with TCP per se.*
    - PCI I/O bus. 64 bit, 66 MHz = 4.2 Gb/s. Solution: InfiniBand?]
- Adaptation Bottlenecks
  - Flow Control
    - No adaptation currently being done in any standard TCP.
    - Static-sized window/buffer is supposed to work for both the LAN & WAN.
  - Congestion Control
    - Adaptation mechanisms will *not* scale, particularly TCP Reno.
    - Adaptation mechanisms *induce* burstiness to the aggregate traffic stream.

Host-Interface Bottleneck (Software)

- First-Order Approximation
  - deliverable bandwidth = maximum-sized packet / interrupt latency
  - e.g., 1500-byte MTU / 50 μs = 30 MB/s = 240 Mb/s   625 Mb/s – 900+ Mb/s
    CC   no CC
- Problems
  - Maximum-sized packet (or MTU) is only 1500 bytes for Ethernet.
  - Interrupt latency to process a packet is quite high.
  - CPU utilization for network tasks is too high. (See next slide.)
- Solutions Intended to Boost TCP Performance
  - Reduce frequency of interrupts, *e.g., interrupt coalescing or OS-bypass.*
  - Increase effective MTU size, *e.g., interrupt coalescing or jumbograms.*
  - Reduce interrupt latency, *e.g., push checksums into hardware, "zero-copy"*
  - Reduce CPU utilization, *e.g., offload protocol processing to NIC.*

666-MHz Alpha with Linux

(Courtesy: USC/ISI)



Note: The congestion-control mechanism does *not* get “activated” in these tests.

Solutions to Boost TCP Performance

(many non-TCP & non-standard)

- Interrupt Coalescing
  - Increases bandwidth (BW) at the expense of even higher latency.
- Jumbograms
  - Increases BW with minimal increase in latency, but at the expense of more blocking in switches/routers and lack of interoperability.
- ULNI or OS-Bypass Protocol
  - Increases BW & decreases latency by an order of magnitude or more.
  - Integrate OS-bypass into TCP?

    VIA over TCP (IETF Internet Draft, GigaNet, July 2000).
- Interrupt Latency Reduction (possible remedy for TCP)
  - Provide "zero-copy" TCP (*a la* OS-bypass) but OS still middleman.
  - Push protocol processing into hardware, e.g., checksums.

- *Which ones will guide the design of a HP-TCP?*

Benchmarks: TCP

- TCP over Gigabit Ethernet (via loopback interface)
  - Theoretical Upper-Bound: 750 Mb/s due to the nature of TCP Reno.
  - Environment: Red Hat Linux 6.2 OS on 400-MHz & 733-MHz Intel PCs; Alteon AceNIC GigE cards; 32-bit, 33-MHz PCI bus.
  - Test: Latency & bandwidth over loopback interface.
    - Latency: O(50 μs).
    - Peak BW w/ default set-up: 335 Mb/s (400) & 420 Mb/s (733).
    - Peak BW w/ *manual* tweaks by network gurus at both ends: 625 Mb/s.
      - Change default send/receive buffer size from 64 KB to 512 KB.
      - Enable interrupt coalescing. (2 packets per interrupt.)
      - Jumbograms. *Theor. BW: 18000 / 50 = 360 MB/s = 2880 Mb/s.*
  - Problem: OS is the middleman. Faster CPUs provide slightly less latency and slightly more BW. 10GigE BW for a high-speed connection wasted.

Solution: ULNI?

Problem?
• Congestion control
• Data copies

Los Alamos National Laboratory RADIANT Research and Development in Advanced Network Technology

What's Wrong with TCP?

- Host-Interface Bottleneck
  - Software
    - A host can only send and receive packets as fast as the OS can process the packets.
      - Excessive copying.
      - Excessive CPU utilization.
  - [Hardware (PC) *Not anything wrong with TCP per se.*
    - PCI I/O bus. 64 bit, 66 MHz = 4.2 Gb/s. Solution: InfiniBand?]
- Adaptation Bottlenecks
  - Flow Control
    - No adaptation currently being done in any standard TCP.
    - Static-sized window/buffer is supposed to work for both the LAN and WAN.
  - Congestion Control
    - Adaptation mechanisms will *not* scale, particularly TCP Reno.

Los Alamos National Laboratory RADIANT Research and Development in Advanced Network Technology

Adaptation Bottleneck

- Flow Control
  - Issues
    - No adaptation currently being done in any standard TCP.
    - 32-KB static-sized window/buffer that is supposed to work for both the LAN and WAN.
  - Problem: Large bandwidth-delay products require flow-control windows as large as 1024-KB to fill the network pipe.
  - Consequence: As little as 3% of network pipe is filled.
  - Solutions
    - *Manual* tuning of buffers at send and receive end-hosts.
      - Too small → low bandwidth. Too large → waste memory (LAN).
    - *Automatic* tuning of buffers.
      - PSC: Auto-tuning but does not abide by TCP semantics, 1998.
      - LANL: Dynamic flow control, 2000. Increase BW by 8 times. Fair?
    - *Network striping & pipelining* with default buffers. Unfair. UIC, 2000.

Los Alamos National Laboratory RADIANT Research and Development in Advanced Network Technology

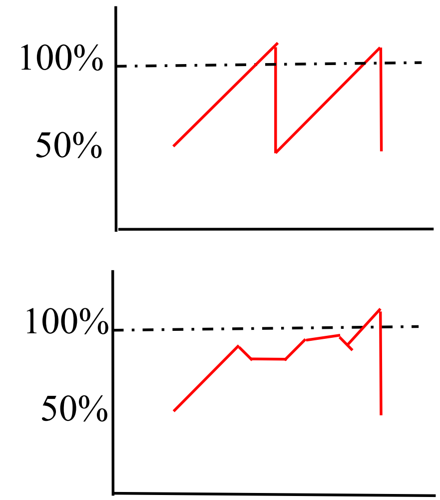
Adaptation Bottleneck

- Congestion Control
  - Adaptation mechanisms will *not* scale due to
    - Additive increase / multiplicative decrease algorithm (see next slide).
      - *Induces* bursty (i.e., self-similar or fractal) traffic.
  - TCP Reno congestion control
    - Bad: Allow/induce congestion.
      Detect & recover from congestion.

      Analogy: "Deadlock detection & recovery" in OS.
    - Result: "At best" 75% utilization in steady state.
  - TCP Vegas congestion control
    - Better: Approach congestion but try to *avoid* it.
      Usually results in better network utilization.

      Analogy: "Deadlock avoidance" in OS.

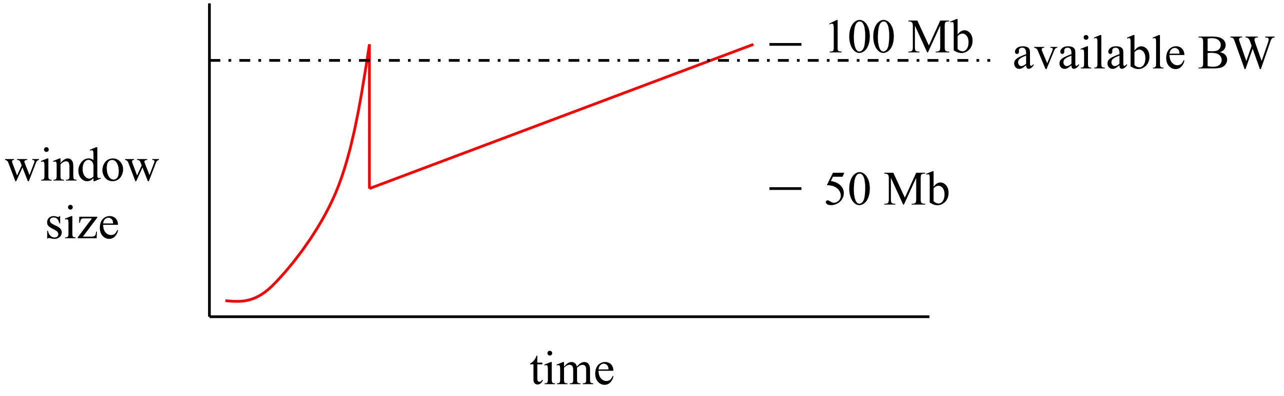
Utilization vs. Time



"Optimal" Bandwidth

- The future performance of computational grids (as well as clusters & supercomputers trying to get away from ULNI scalability problems) looks bad if we continue to rely on the widely-deployed TCP Reno.

  Example: High BW-delay product: 1 Gb/s WAN * 100 ms RTT = 100 Mb
- Additive increase
  - when window size is 1 ⟶ 100% increase in window size.
  - when window size is 1000 ⟶ 0.1% increase in window size.



Re-convergence to "optimal" bandwidth takes nearly 7 minutes! (Performance is awful if network uncongested.)

AIMD Congestion Control

- Stable & fair (under certain assumptions of synchronized feedback) but
  - ➢ Not well-suited for emerging applications (e.g., streaming & real-time audio and video)
    - Its reliability and ordering semantics increases end-to-end delays and delay variations.
    - Multimedia applications *generally* do not react well to the large and abrupt reductions in transmission rate caused by AIMD.
  - ➢ Solutions
    - Deploy "TCP-friendly" (non-AIMD) congestion-control algorithms, e.g., binomial congestion-control algorithms such as inverse increase / additive decrease (MIT).
    - Provide a protocol that functionally "sits" between UDP & TCP, e.g., RAPID: Rate-Adjusting Protocol for Internet Delivery
      - n% reliability (vs. 100% reliability of TCP) where $0\% < n \leq 100\%$
      - Provide high performance *and* utilization regardless of network conditions.
      - Hmm … what about wireless?

Los Alamos National Laboratory RADIANT Research and Development in Advanced Network Technology

How to Build a HP-TCP?

- Host-Interface Bottleneck
  - Software
    - A host can only send and receive packets as fast as the OS can process the packets.
  - Hardware (PC)
    - PCI I/O bus. 64 bit, 66 MHz = 4.2 Gb/s. Solution: InfiniBand?

BW problems potentially solvable. Latency?
What happens when we go optical to the chip?

Based on past trends, the I/O bus will continue to be a bottleneck.

- Adaptation Bottlenecks
  - Flow Control
    - No adaptation currently being done in any standard TCP.
    - Static-sized window/buffer is supposed to work for both the LAN and WAN.
  - Congestion Control
    - Adaptation mechanisms will *not* scale, particularly TCP Reno.

Solutions exist but are not widely deployed.

TCP Vegas? Binomial congestion control?

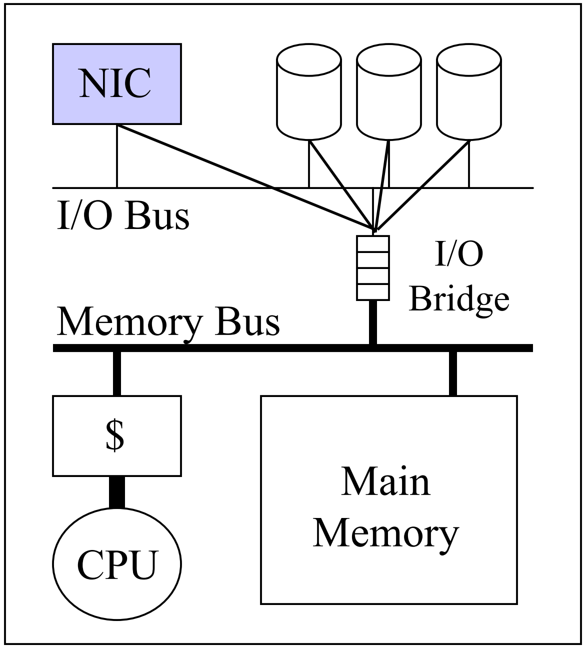
Los Alamos National Laboratory RADIANT Research and Development in Advanced Network Technology

Crossroads for High-Performance Networking

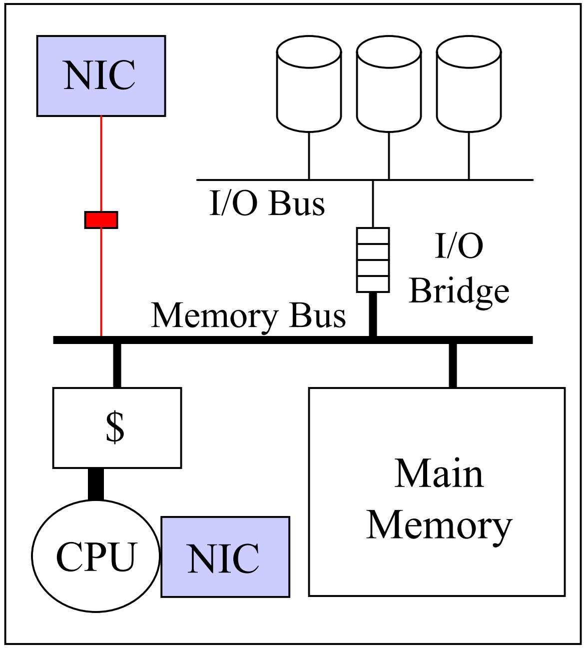
http://www.canet3.net/library/papers/wavedisk.doc

- Hardware/Architecture
  - ➢ InfiniBand helps but network-peer CPU (or co-processor) or network-integrated microprocessor architecture may be needed.

Today



Near Future



Far Future: Network is Storage
Memory- & disk-access too slow.

Software Bottleneck?
Offload as much protocol processing to the NIC CPU

Los Alamos National Laboratory RADIANT Research and Development in Advanced Network Technology

Crossroads for High-Performance Networking

Software

	Internet Community		*HPC Community*
Early 1990s		TCP Tahoe/Reno	
Mid-1990s	TCP Tahoe/Reno		specialized ULNI
Late 1990s	TCP Tahoe/Reno	TCP Vegas	standardized ULNI
Tomorrow	TCP Tahoe/Reno	"TCP Vegas"	standardized ULNI (extend to WAN, e.g., add IP routing, congestion control, MTU discovery)

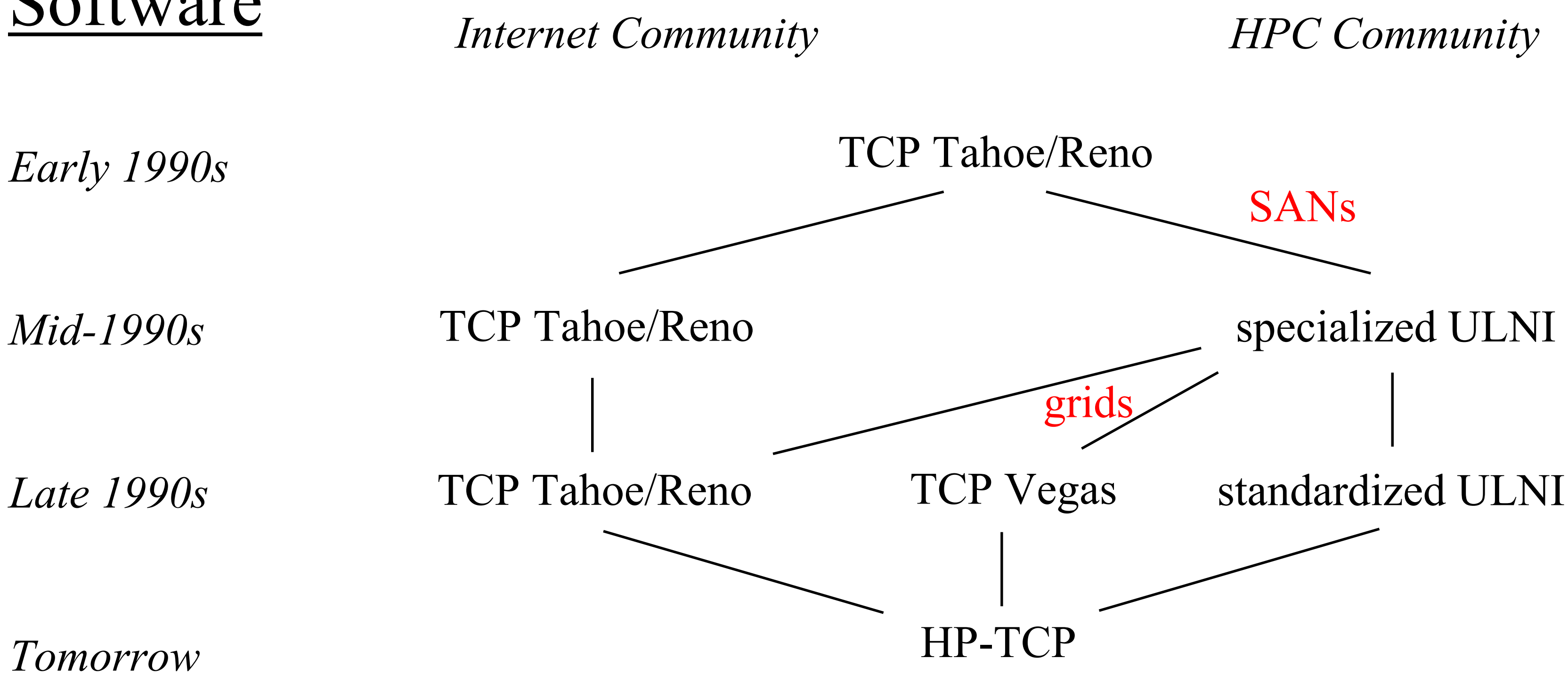
SANs

grids

Los Alamos National Laboratory RADIANT Research and Development in Advanced Network Technology

Crossroads for High-Performance Networking

Software



Los Alamos National Laboratory
RADIANT
Research and Development in Advanced Network Technology

That’s all folks!