System Support for Clusters of SMPs

David E. Culler

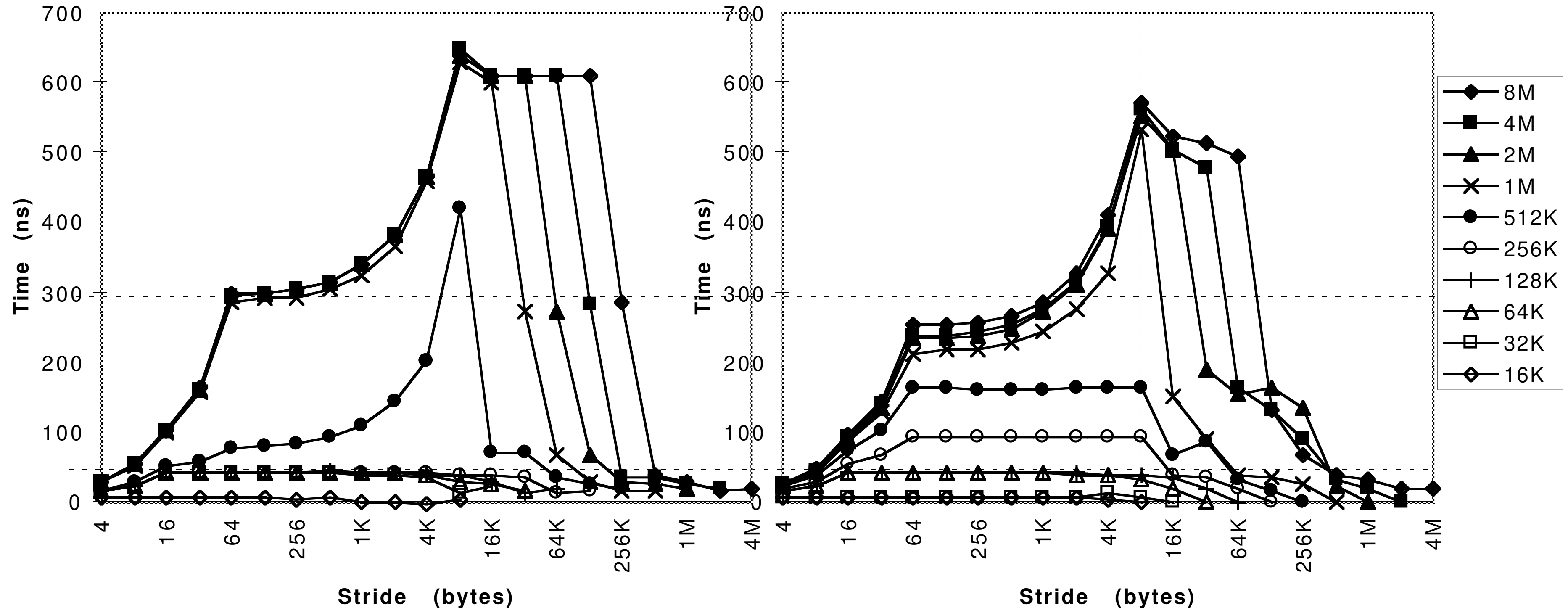
Steve Lumetta, Alan Mainwaring,

Brent Chun, Fred Wong

Experimental Setting

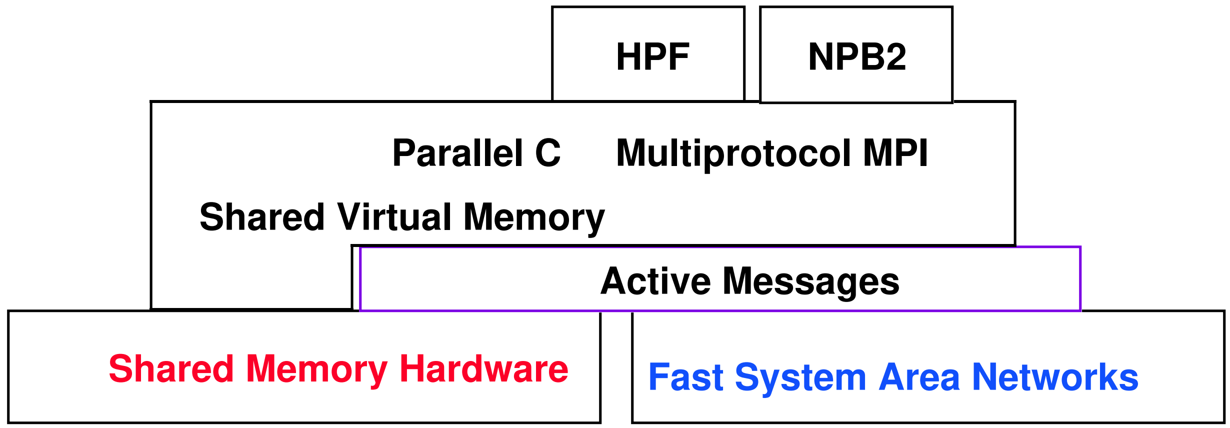
- **100+ cluster of Ultra 170s**
  - **167 MHz, 0.5 MB Cache, 128 MB mem, 2x4 GB disk**
  - **Myricom Lanai into fat-tree myrinet**
    - » **160 MB/s links, 8 port switches**
- **4 x 8-way cluster of Enterprise 5000s**
  - **167 MHz, 0.5 MB Cache, 512 MB mem, 7x4 GB disk array**
  - **1 (to 3) Myricom Lanai into fat-tree myrinet**

Towards a Microbenchmarking Characterizer



- **Nodal Memory System Hierarchy**
  - **300 ns (vs 250) to mem, 480 to L2 within SMP**

Layered Approach

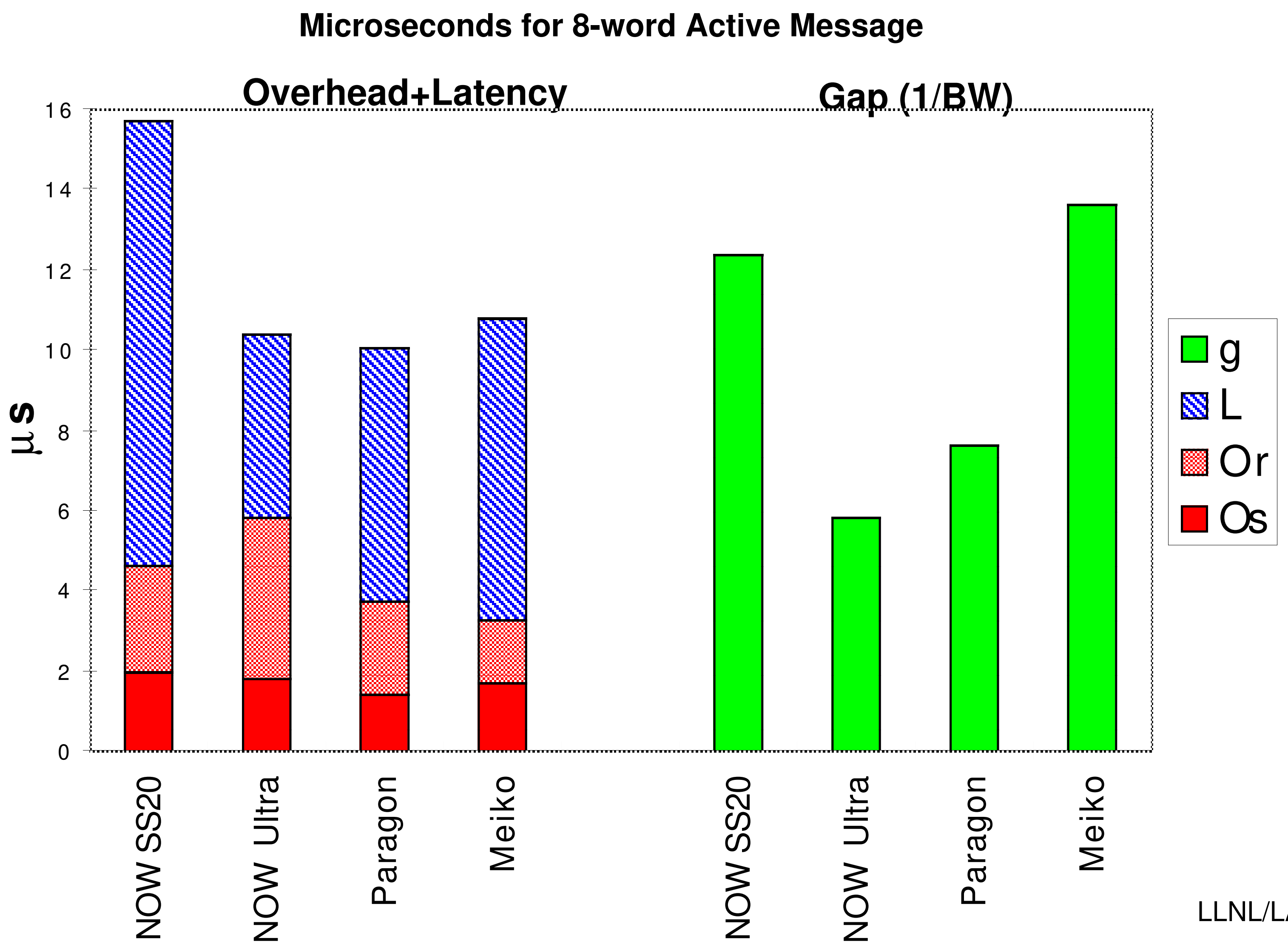


- What is the best level to go multiprotocol?
- How much of the structure is exposed?
- Quantify Trade-offs

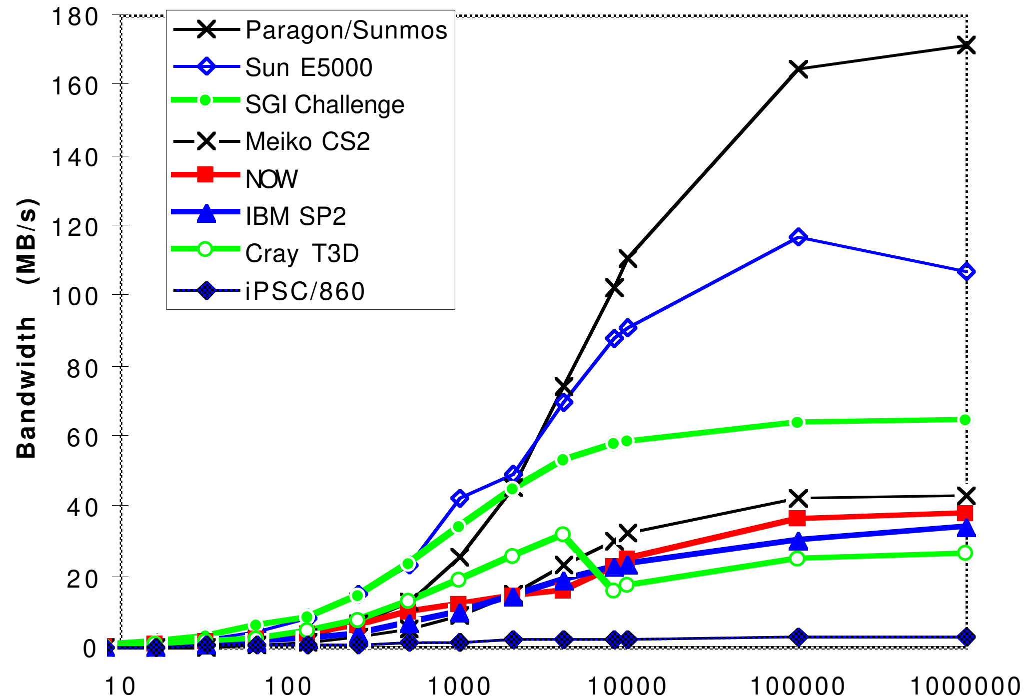
Status

- **Complete NOW-based System**
  - Active Messages
  - MPI, Split-C, HPF
  - Mantis Debugger
  - Glunix
- **Clumps operational**
  - Inter-SMP Active Messages
  - Intra-SMP Active Messages
  - Multiprotocol layers in debug
  - Split-C
  - GlobEx execution layer

LogP Microbenchmarks - NOW

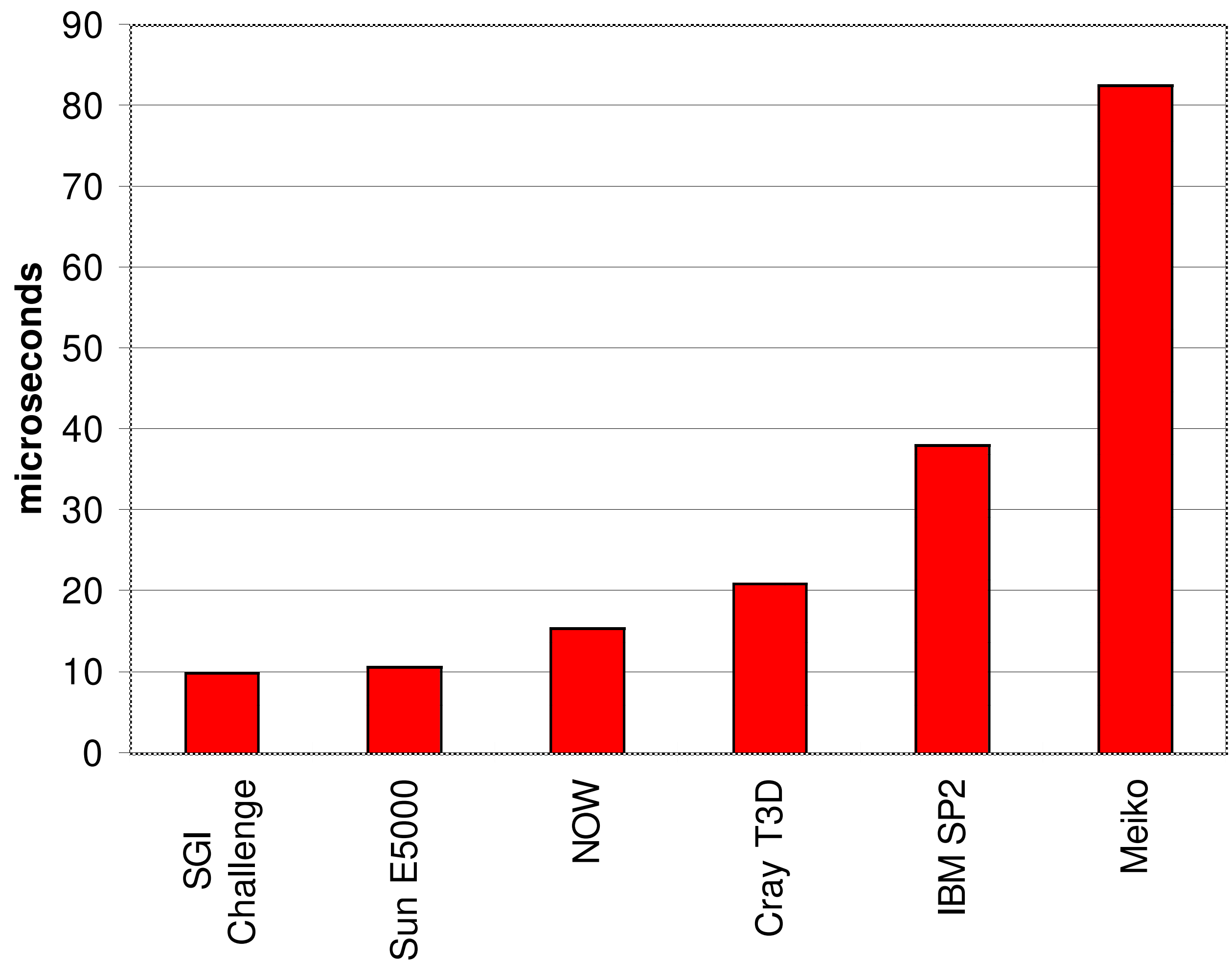


MPI on (GAM) Active Messages



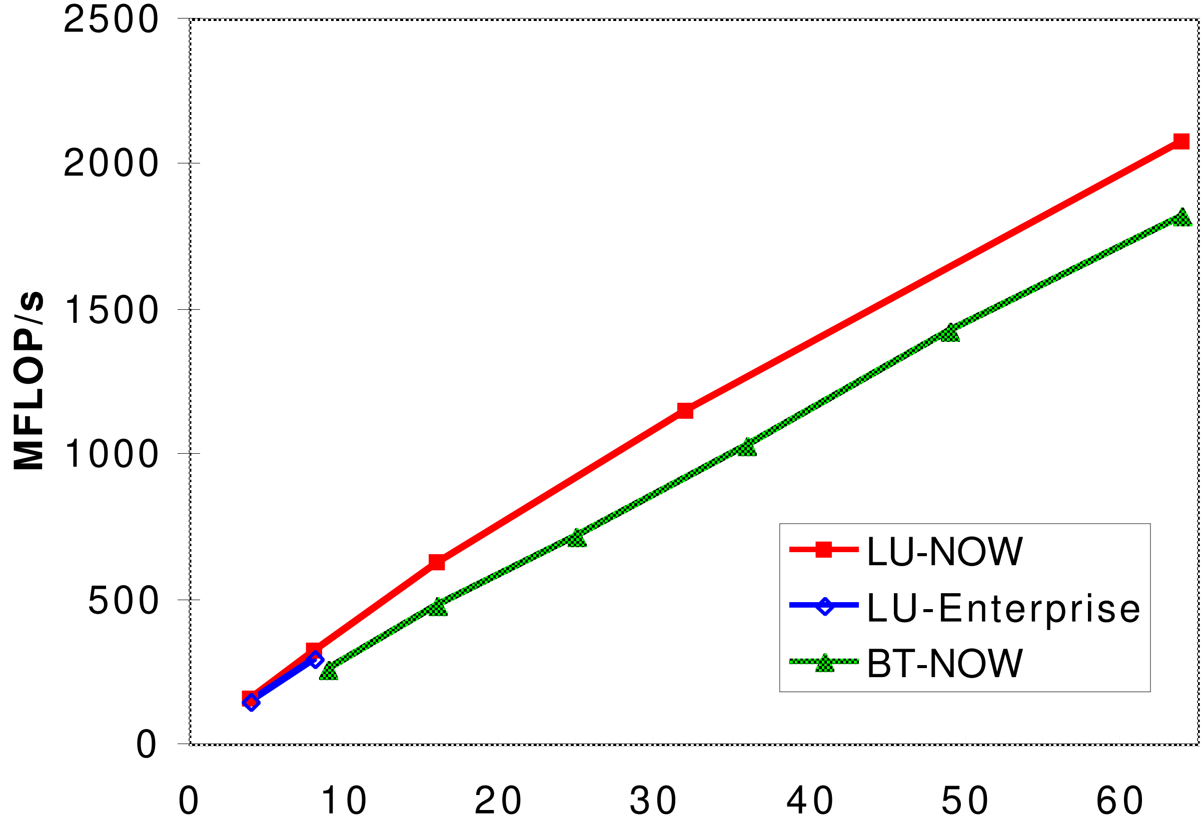
- **Operationsl on AM-II also**
- **Multithreaded versions**

MPI Start-up

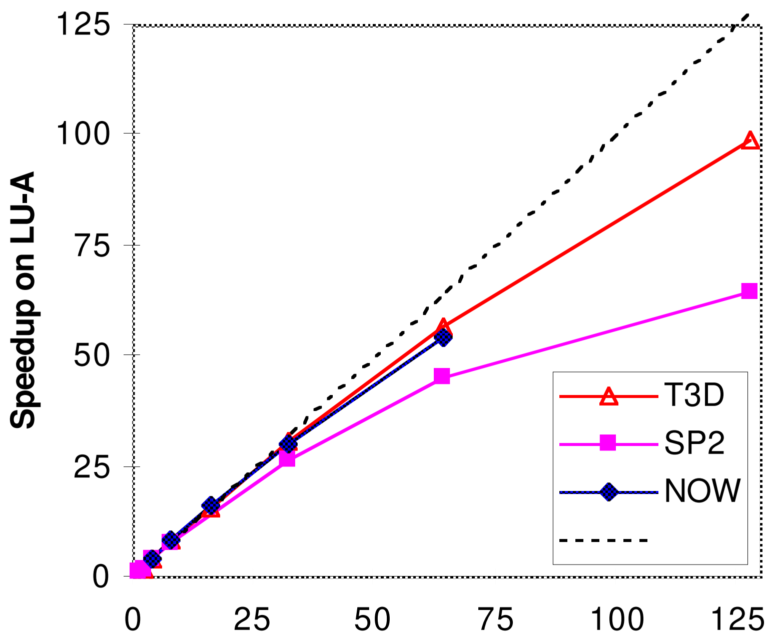


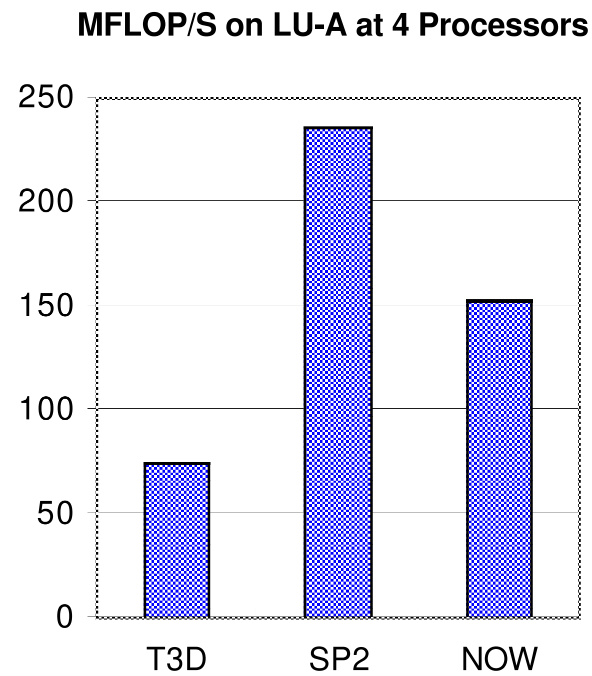
90
80
70
60
50
40
30
20
10
0
microseconds
SGI Challenge
Sun E5000
NOW
Cray T3D
IBM SP2
Meiko

NAS Parallel Benchmarks: NOW vs SMP

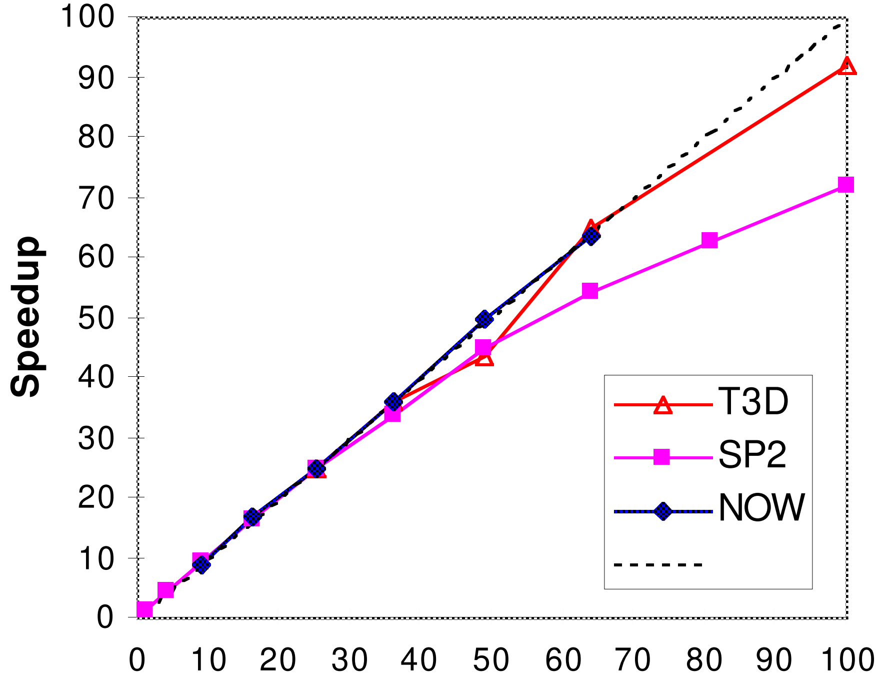


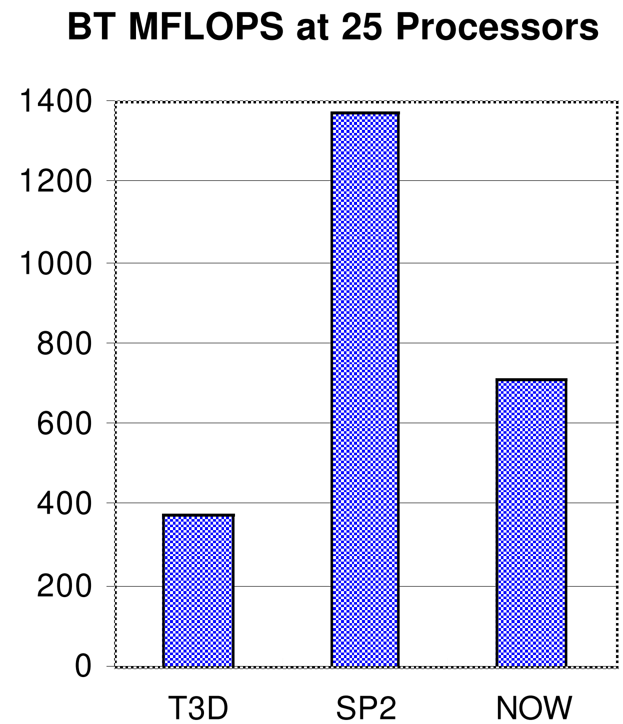
NAS Parallel Benchmarks - LU





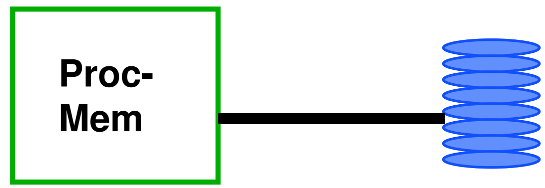
NAS Parallel Benchmarks - BT



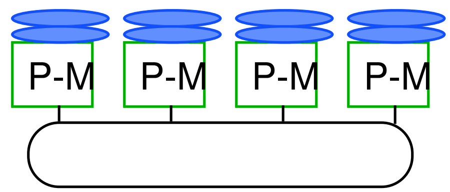


A look at Serious File I/O

- **Traditional I/O system**

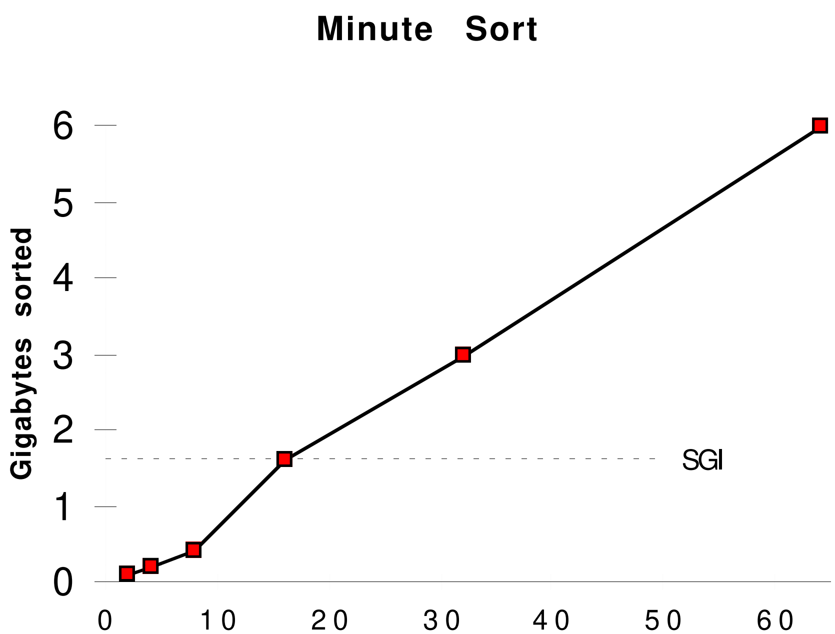


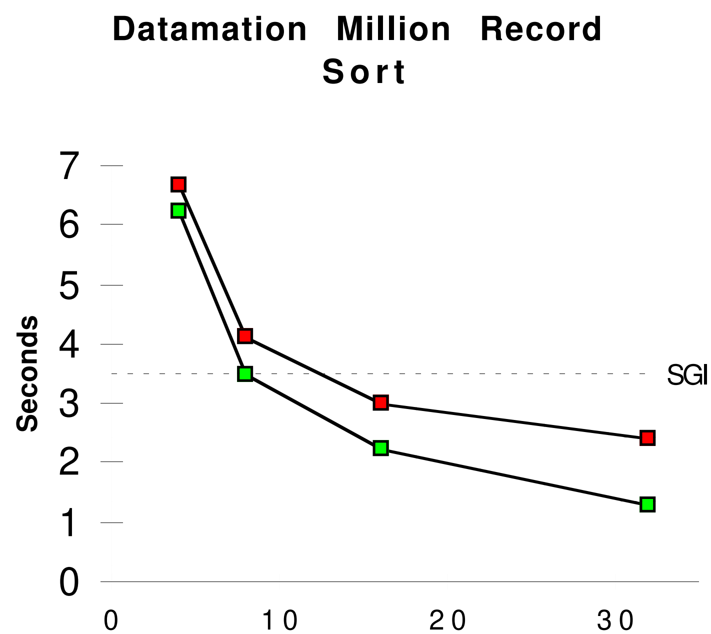
- **NOW I/O system**



- **Benchmark Problem: sort large number of 100 byte records with 10 byte keys**
  - **start on disk, end on disk**
  - **accessible as files (use the file system)**
  - **Datamation sort: 1 million records**
  - **Minute sort: quantity in a minute**
- **CLUMPS ???**

World-Record Disk-to-Disk Sort





Sort: Keys to Success

- **Performance Isolation: highly tuned local disk-to-disk sort**
  - manage local memory
  - manage disk striping
  - memory mapped I/O with m-advise, buffering, threads
  - manage overlap
- **Efficient Communication**
  - completely hidden under disk I/O
  - competes for I/O bus bandwidth
- **Self-tuning Software**
  - probe available memory, disk bandwidth, trade-offs

- **CLUMPS change the trade-offs**

Rethinking Communication

- **Low latency, high bandwidth => direct hardware access to the network**
- **At any time, only a small fraction of the potential applications are really communicating.**
- **Communication within an application is very bursty**

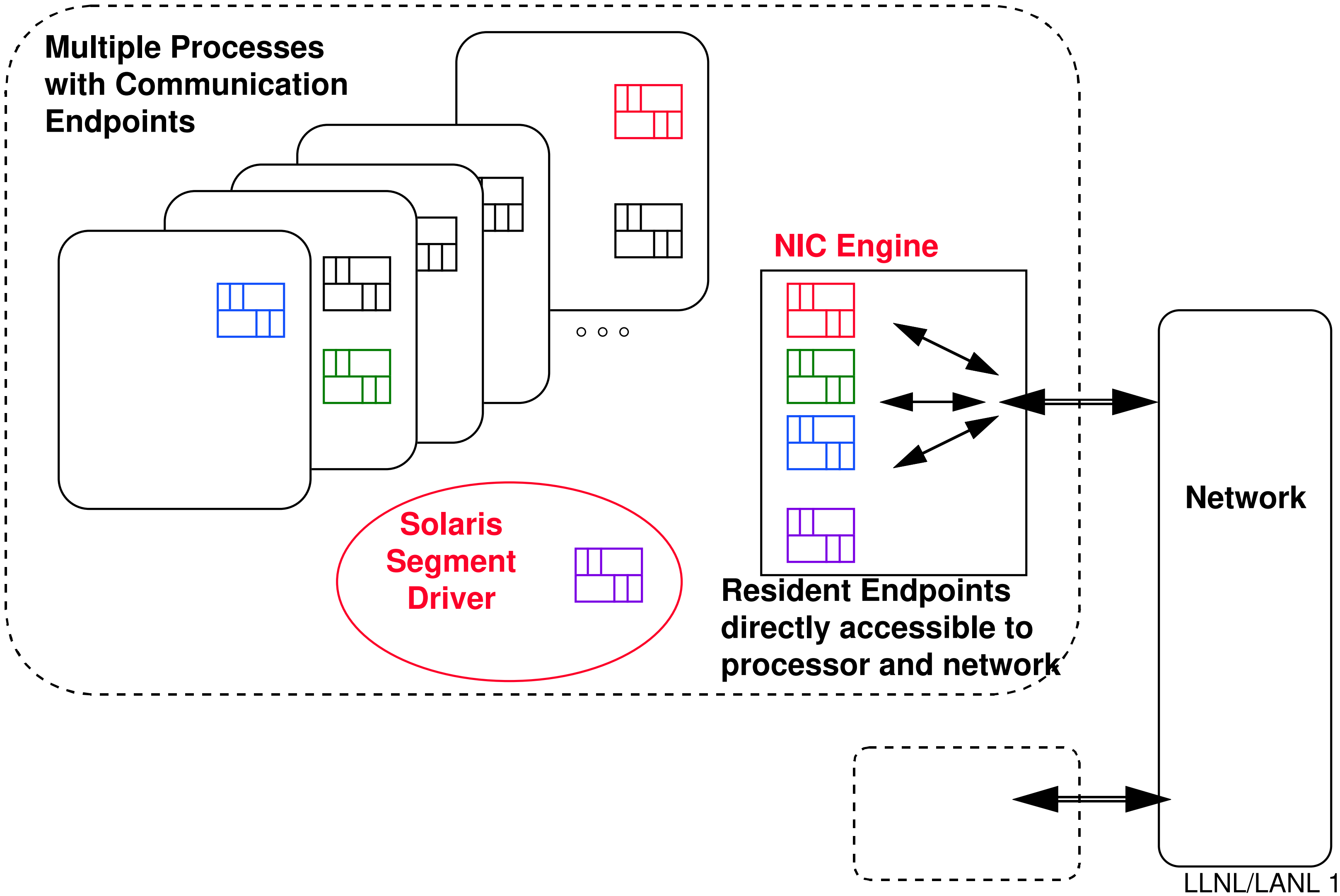
=> Provide each application with its own virtual network (s)

Bind virtual networks on demand to physical communication resources

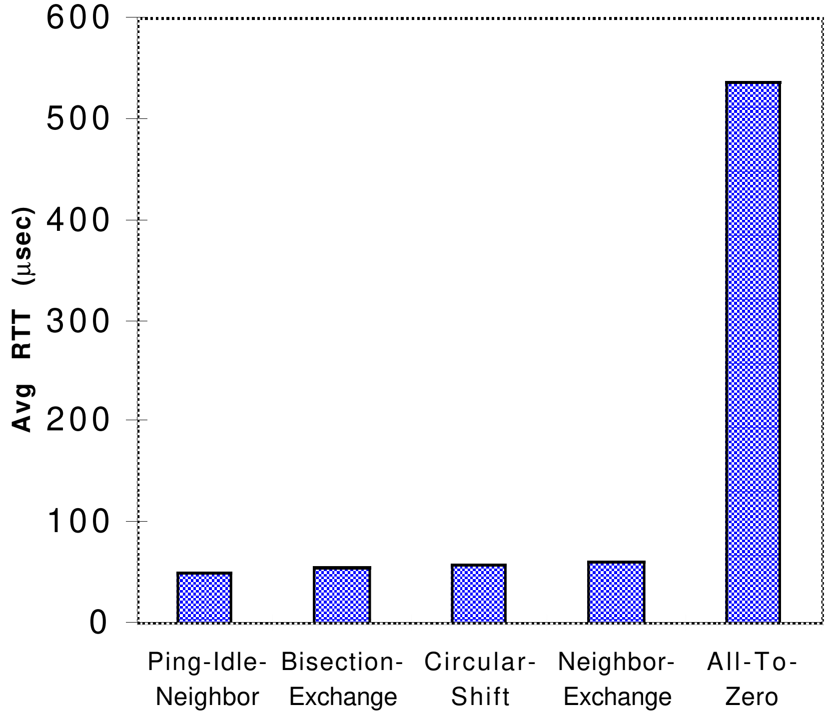
=> provide generality for NOWs

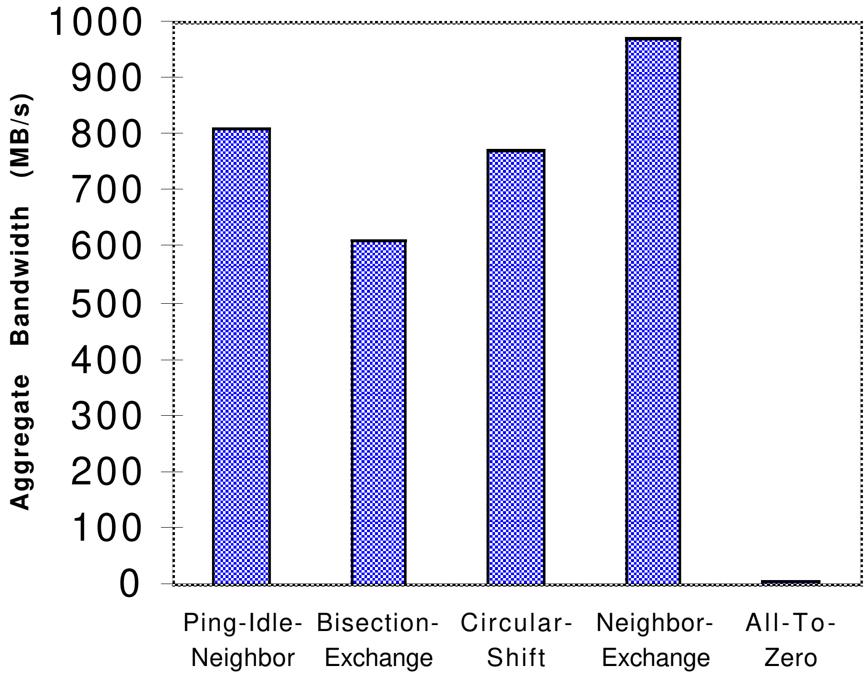
=> Critical for CLUMPS!

Virtual Networks

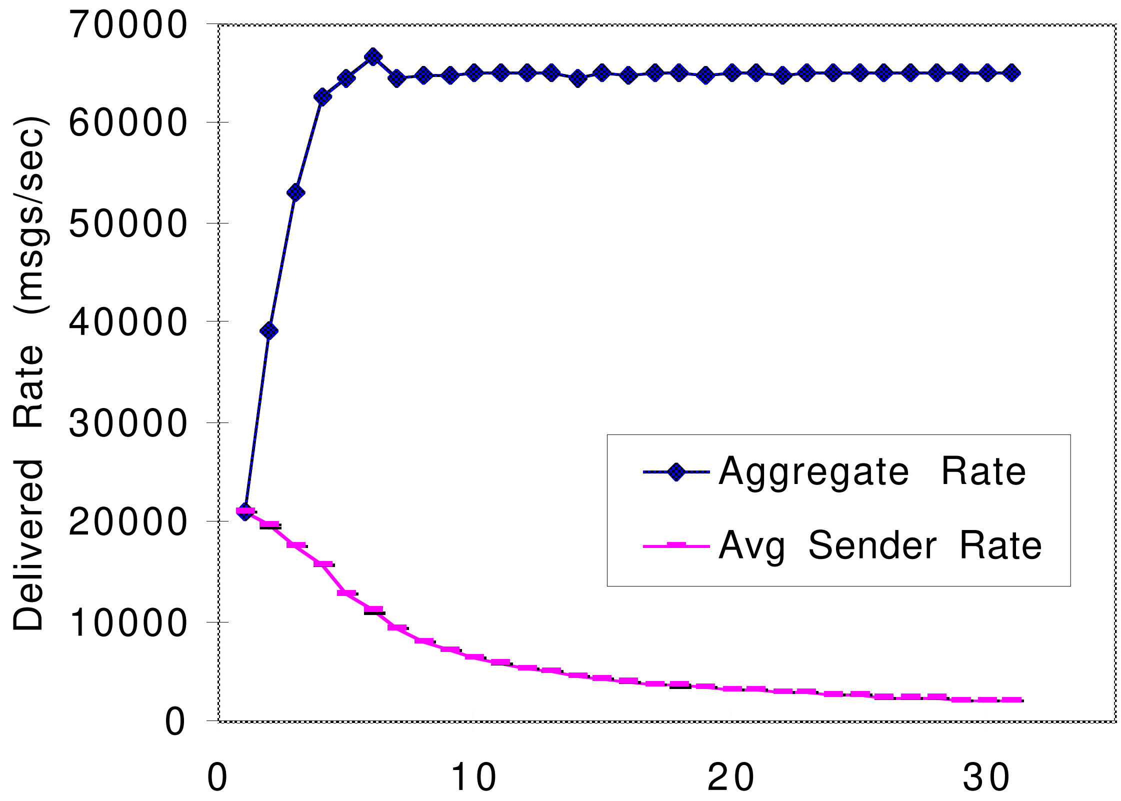


Performance of Current Version

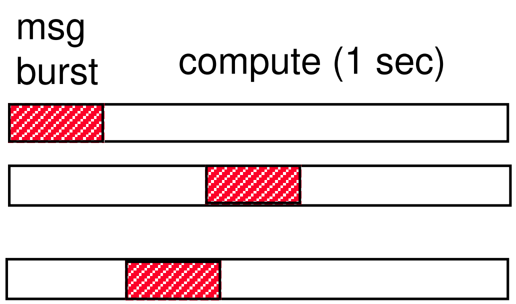


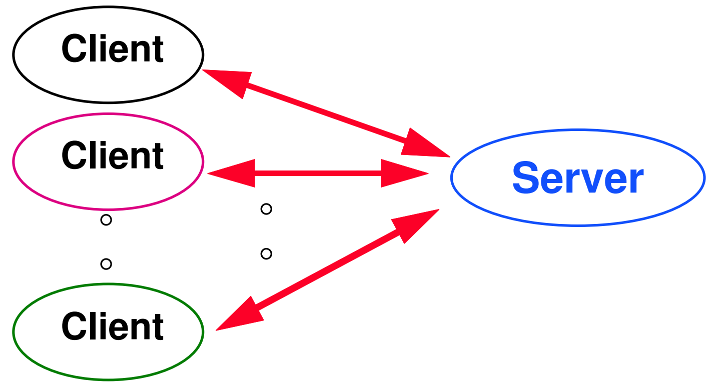


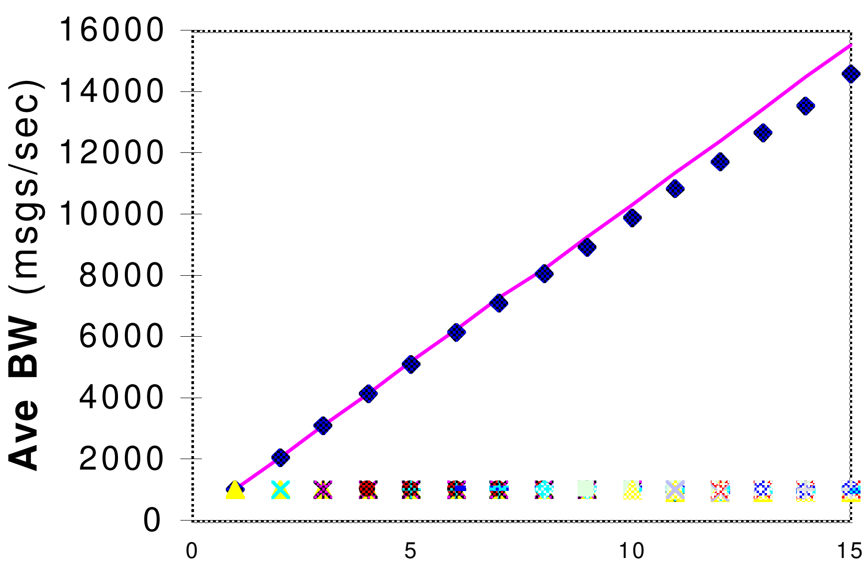
Fairness under heavy load

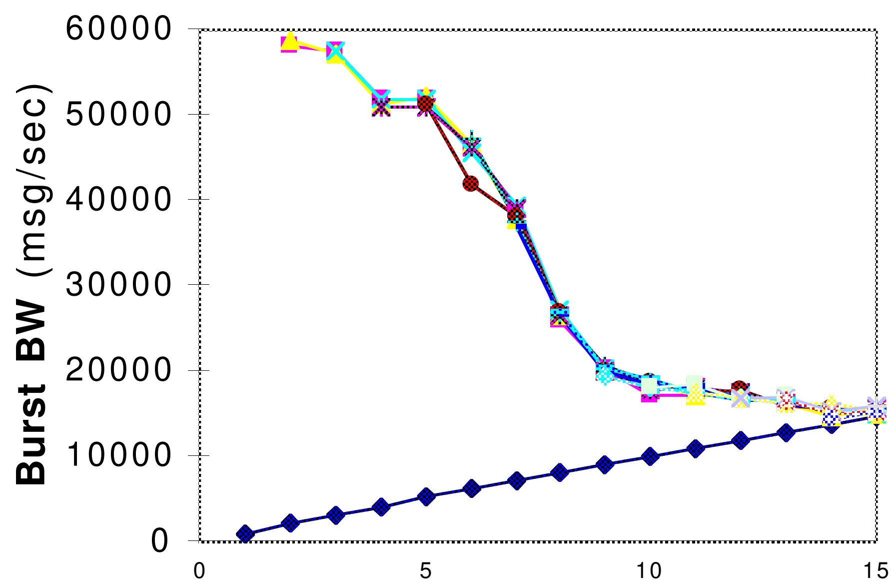


Service to Multiple Virtual Networks









Multiprotocol Active Messages

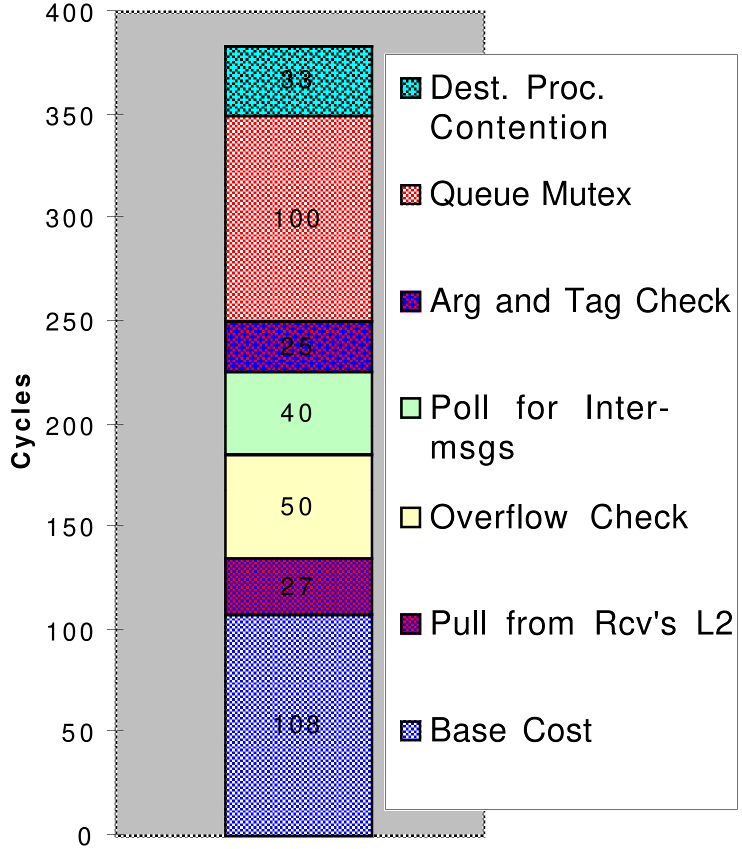
	Shared Memory AM-II	Myrinet AM-II
Send Overhead	2.3 μs	6.2 μs
Recv Overhead	1.2 μs	7.7 μs
Latency	0	14.6 μs
One-way-Time	3.5 μs	28.5 μs (vs 23 on NOW)
Gap	3.5 μs	16.9 μs
Bulk BW	158 MB/s	26 MB/s (echo test)
- N1/2 at 7800 bytes		

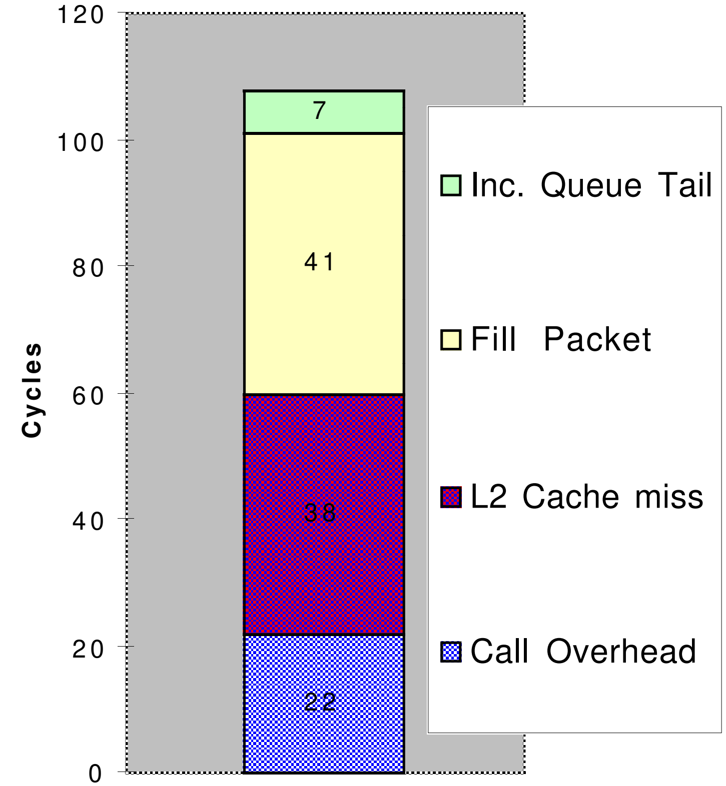
- Initial Application comparisons
- Evaluation of polling freqency

Multiprotocol Split-C

- **Naive version upon multiprotocol AM**
- **UPC redesign allows use of hardware shared memory within the node**
  - **clear distinction of private and shared**
  - **clean up on addressing / indexing for global arrays and pointers**

Breakdown of Send Overhead





Multiply Connected Clumps

- **Multiple processors share a pool of NICs**
- **How many are required?**
- **How are end-points assigned to NICs?**
- **Can a single end-point be stripped over NICs?**
- **How does multiple NICs impact network design trade-offs?**
  - **randomization across NICs**
  - **fault tolerance**

Collective Communication Operators

- Exploit tight coupling within the SMP
- Avoid intra-node contention
- SMPs prone to false sharing on permutations

Load Balancing

- Load balancing within SMP much cheaper than across
- Analytical models suggest that intra-node balancing has significant effect
- Building programming environment for complete study