Improving Cluster Performance

- Service Offloading
- Larger clusters may need to have special purpose node(s) to run services to prevent slowdown due to contention (e.g. NFS, DNS, login, compilation)
- In cluster e.g. NFS demands on single server may be higher due to intensity and frequency of client access
- Some services can be split easily e.g. NSF
- Other that require a synchronized centralized repository cannot be split
- NFS also has a scalability problem if a single client makes demands from many nodes
- PVFS tries to rectify this problem

Multiple Networks/Channel Bonding

- Multiple Networks : separate networks for NFS, message passing, cluster management etc
- Application message passing the most sensitive to contention, so usually first separated out
- Adding a special high speed LAN may double cost

- Channel Bonding: bind multiple channel to create virtual channel
- Drawbacks: switches must support bonding, or must buy separate switches
- Configuration more complex
  - See Linux Ethernet Bonding Driver mini-howto

Jumbo Frames

- Ethernet standard frame 1518 bytes (MTU 1500)
- With Gigabit Ethernet controversy on MTU
  - Want to reduce load on computer i.e. number of interrupts
  - One way is to increase frame size to 9000 (Jumbo Frames)
  - Still small enough not to compromise error detection
  - Need NIC and switch to support
  - Switches which do not support will drop as *oversized* frames
- Configuring eth0 for Jumbo Frames
  ifconfig eth0 mtu 9000 up
- If we want to set at boot put in startup scripts
  - Or on RH9 put in /etc/sysconfig/network-scripts/ifcfg-eth0
    MTU=9000
- More on performance later

Interrupt Coalescing

- Another way to reduce number of interrupt
- Receiver : delay until
  - Specific number of packets received
  - Specific time has elapsed since first packet after last interrupt
- NICs that support coalescing often have tunable parameters
- Must take care not to make too large
  - Sender: send descriptors could be depleted causing stall
  - Receiver: descriptors depleted cause packet drop, and for TCP retransmission. Too many retransmissions cause TCP to apply congestion control reducing effective bandwidth

Interrupt Coalescing (ctd.)

- Even if not too large, increasing causes complicated effects
  - Interrupts and thus CPU overhead reduced
    - If CPU was interrupt saturated may improve bandwidth
  - Delay causes increased latency
    - Negative for latency sensitive applications

Socket Buffers

- For TCP, send socket buffer size determines the maximum window size (amount of unacknowledged data "in the pipe")
  - Increasing may improve performance but consumes shared resources possibly depriving other connections
  - Need to tune carefully
- Bandwidth-delay product gives lower limit
  - Delay is Round Trip Time (RTT): time for sender to send packet, reciever to receive and ACK, sender to receive ACK
  - Often estimated using ping (although ping does not use TCP and doesn't have its overhead!!)
    - Better if use packet of MTU size (for Linux this means specifying data size of 1472 + ICMP & IP headers = 1500

Socket Buffers (ctd.)

- Receive socket buffer determines amount that can be buffered awaiting consumption by application
  - If exhausted sender notified to stop sending
  - Should be at least as big as send socket buffer
- Bandwidth-delay product gives lower bound
  - Other factors impact size that gives best performance
    - Hardware, software layers, application characteristics
  - Some applications allow tuning in application
- System level tools allow testing of performance
  - ipipe, netpipe (more later)

Setting Default Socket Buffer Size

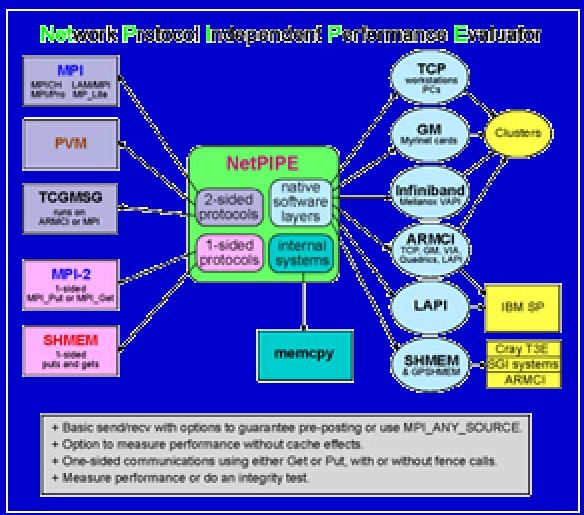
- /proc file system
  - /proc/sys/net/core/wmem_default send size
  - /proc/sys/net/core/rmem_default receive size
- Default can be seen by cat of these files
- Can be set by e.g.

  Echo 256000 > /proc/sys/net/core/wmem_default
- Sysadm can also determine maximum buffer sizes that users can set in
  - /proc/sys/net/core/wmem_max
  - /proc/sys/net/core/rmem_max
  - Should be at least as large as default!!
- Can be set at boot time by adding to /etc/rc.d/rc.local

Netpipe - http://www.scl.ameslab.gov/netpipe/

- **NETwork Protocol Independent Performance Evaluator**
- Performs simple ping-pong tests, bouncing messages of increasing size between two processes
- Message sizes are chosen at regular intervals, and with slight perturbations, to provide a complete test of the communication system
- Each data point involves many ping-pong tests to provide an accurate timing
- Latencies are calculated by dividing the round trip time in half for small messages (< 64 Bytes)
- NetPIPE was originally developed at the SCL by Quinn Snell, Armin Mikler, John Gustafson, and Guy Helmer

Netpipe Protocols & Platforms



Performance Comparison of LAM/MPI, MPICH, and MVICH on a Cluster Connected by a Gigabit Ethernet Network

Hong Ong and Paul A. Farrell
Dept. of Mathematics and Computer Science
Kent, Ohio
Atlanta Linux Showcase Extreme Linux 2000

Testing Environment Hardware

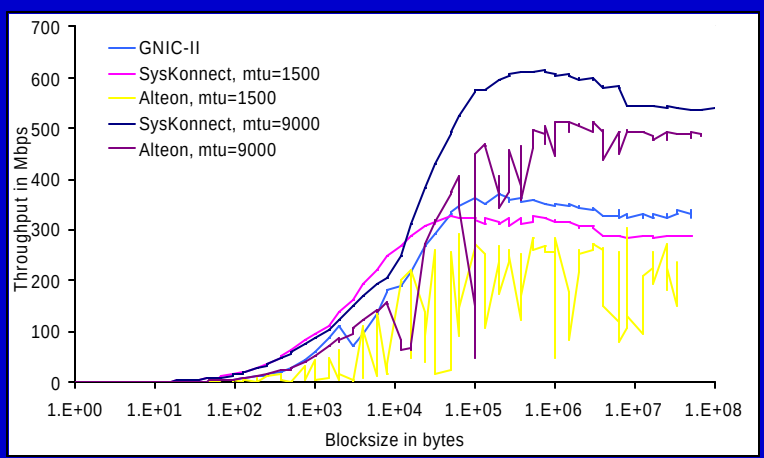
- Two 450MHz Pentium III PCs.
  - 100MHz memory bus.
  - 256MB of PC100 SD-RAM.
  - Back to back connection via Gigabit NICs.
  - Installed in the 32bit/33MHz PCI slot.
- Gigabit Ethernet NICs.
  - Packet Engine *GNIC-II* (Hamachi v0.07).
  - Alteon *ACEnic* (Acenic v0.45).
  - SysKonnect *SK-NET* (Sk98lin v3.01).

Testing Environment - Software

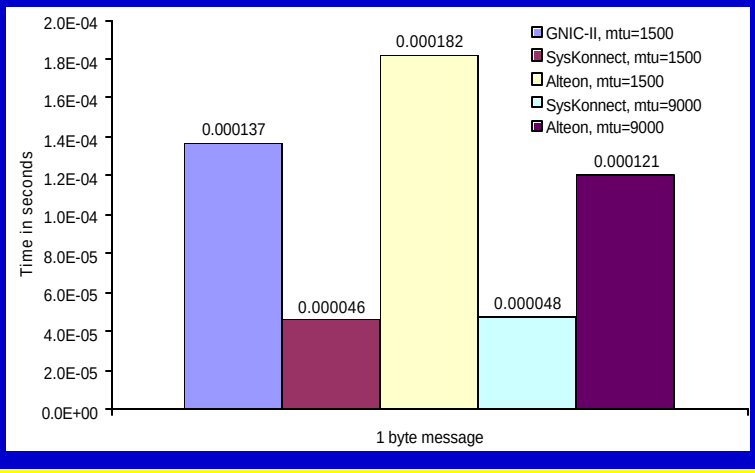
- Operating system.
  - Red Hat 6.1 Linux distribution.
  - Kernel version 2.2.12.
- *Communication interface.*
  - *LAM/MPI v6.3.*
  - *MPICH v1.1.2.*
  - *M-VIA v0.01.*
  - *MVICH v0.02.*
- Benchmarking tool.
  - NetPIPE v2.3.

TCP/IP

Performance - Throughput



TCP/IP
Performance - Latency



Gigabit Over Copper Evaluation

- DRAFT *Prepared by Anthony Betz*
- *April 2, 2002*
- *University of Northern Iowa*
- *Department of Computer Science.*

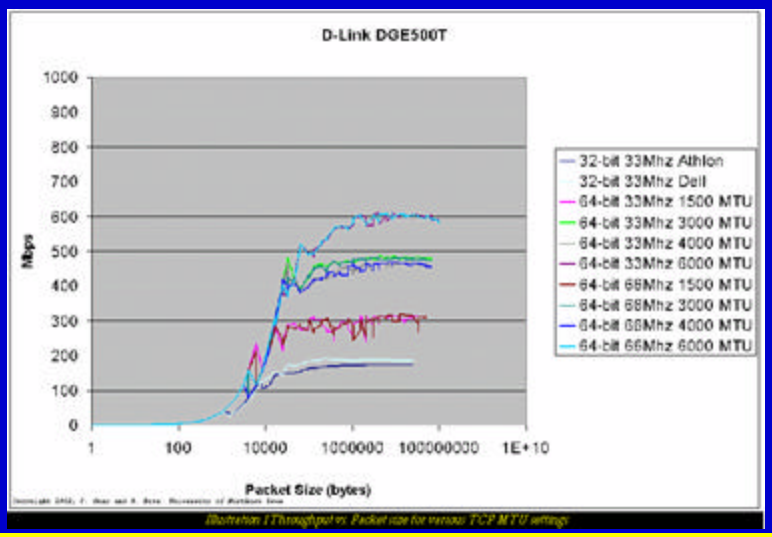
Testing Environment

- Twin Server-Class Athlon systems with 266MHz FSB from QLILinux Computer Systems
  - Tyan S2466N Motherboard
  - AMD 1500MP
  - 2x64-bit 66/33MHz jumper-able PCI slots
  - 4x32-bit PCI slots
  - 512MB DDR Ram
  - 2.4.17 Kernel
  - RedHat 7.2
- Twin Desktop-Class Dell Optiplex Pentium-Class systems
  - Pentium III 500 Mhz
  - 128MB Ram
  - 5x32-bit PCI slots
  - 3x16-bit ISA slots

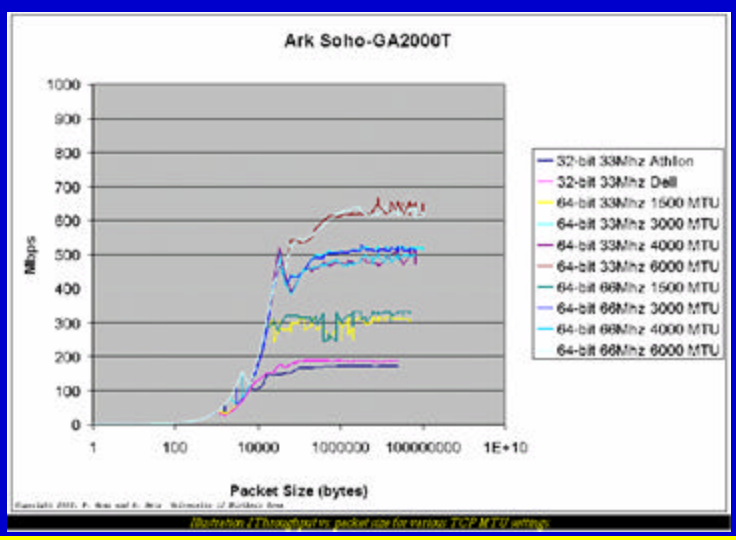
Cards Tested

- D-Link DGE 500T (32-bit) $45
  - SMC's dp83820 chipset, driver ns83820 in 2.4.17 kernel
- ARK Soho-GA2500T (32-bit) $44
- ARK Soho-GA2000T $69
- Asante Giganix $138
  - Same as D-Link except dp83821 chipset
- Syskonnect SK9821 $570
  - driver used was sk98lin from the kernel source
- 3Com 3c996BT $138
  - driver bcm5700, version 2.0.28, as supplied by 3Com
- Intel Pro 1000 XT $169
  - Designed for PCI-X, Intel's e1000 module, version 4.1.7
- Syskonnect SK9D2 $228

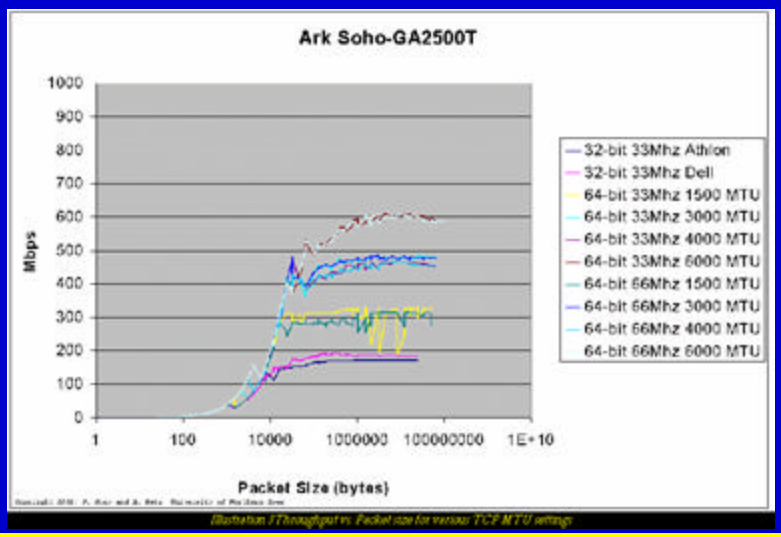
D-Link DGE-500T



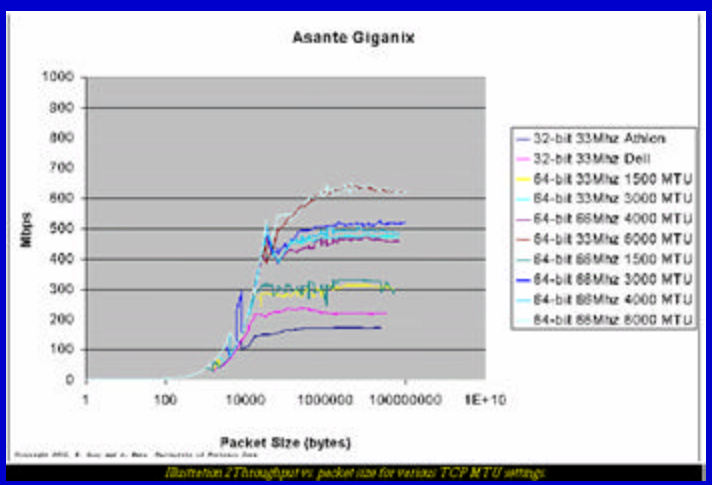
Ark Soho-GA200T



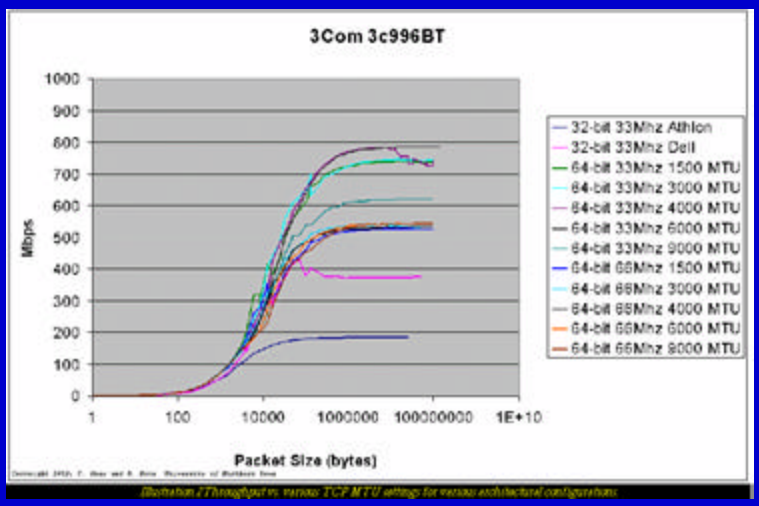
Ark Soho-GA2500T



Asante Giagnix

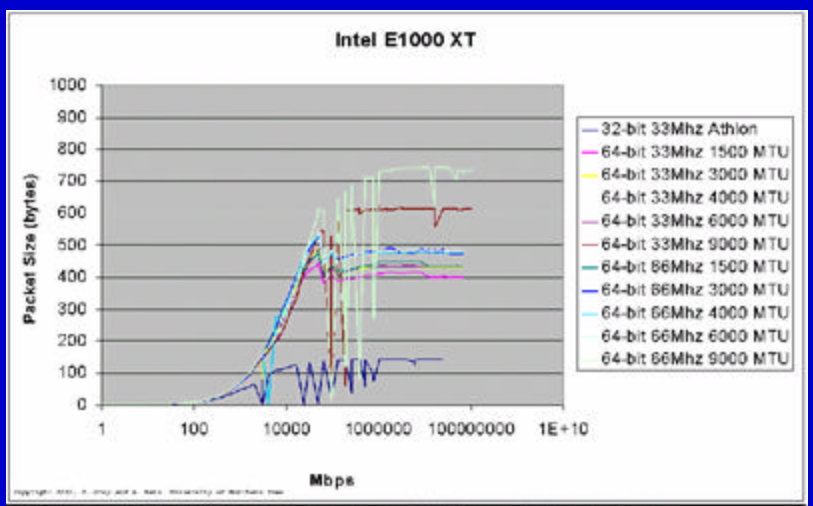


3Com 3c996BT



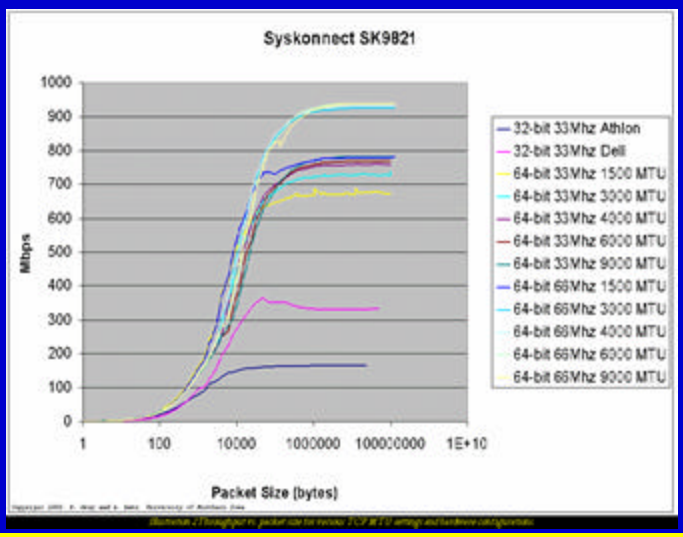
DiSCoV KENT STATE UNIVERSITY 12 January 2004 Paul A. Farrell Cluster Computing 23

Intel E1000 XT

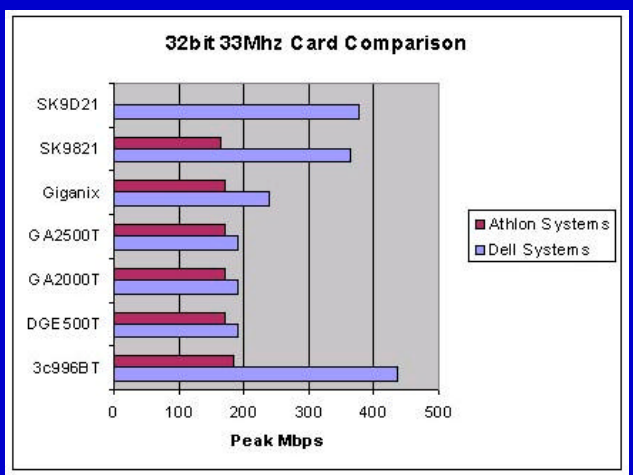


DiSCoV KENT STATE UNIVERSITY 12 January 2004 Paul A. Farrell Cluster Computing 24

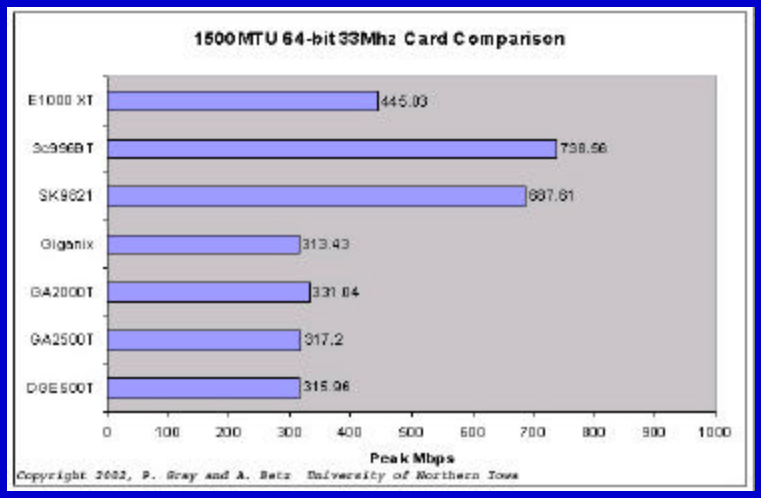
Syskonnect SK9821



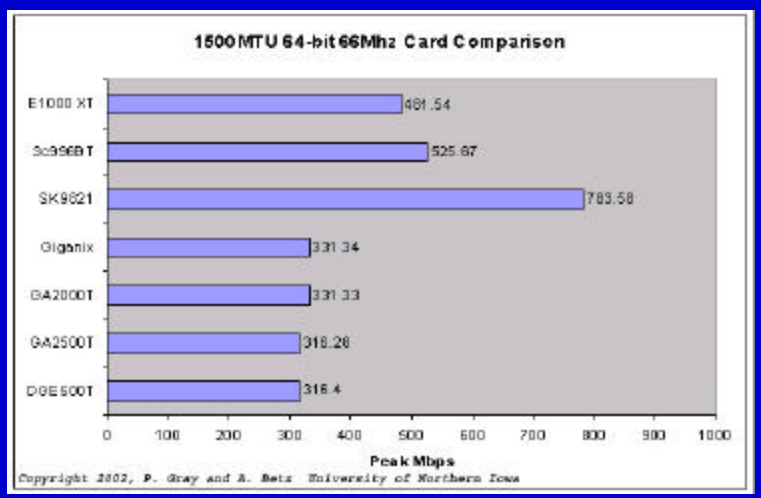
32bit 33MHz



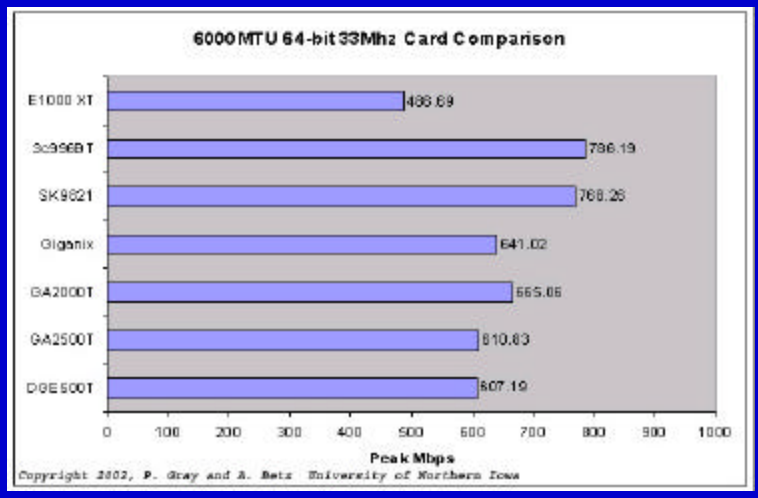
64bit/33MHz MTU 1500



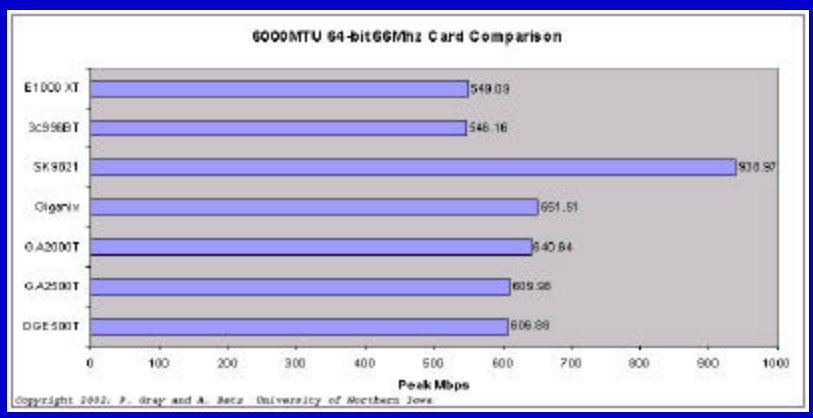
64bit/66MHz MTU 1500



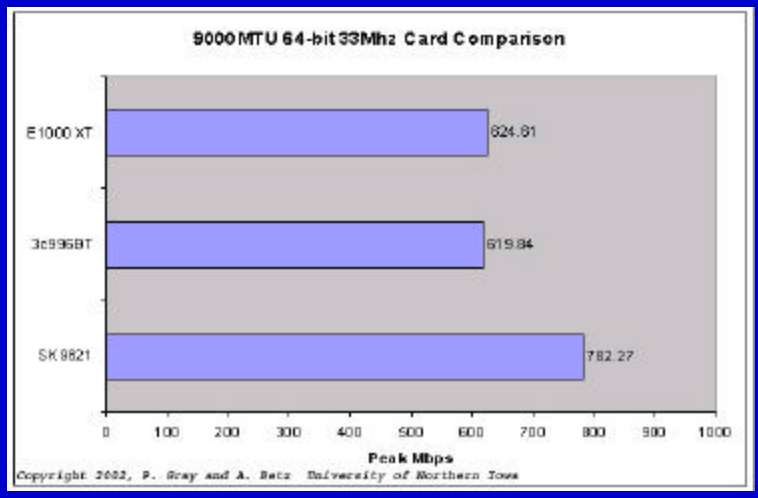
64bit/33MHz MTU 6000



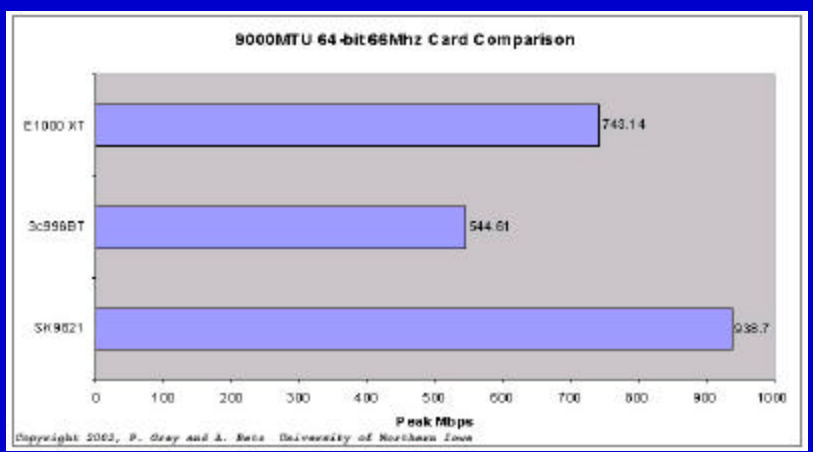
64bit/66MHz MTU 6000



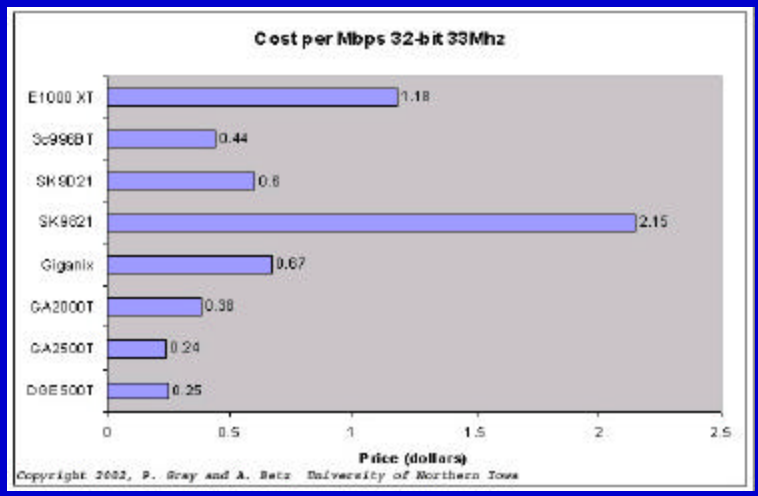
64bit/33MHz MTU 9000



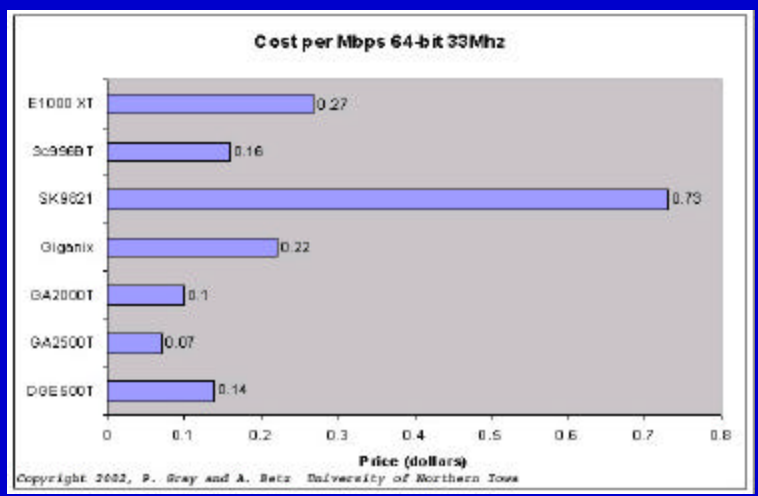
64bit/66MHz MTU 9000



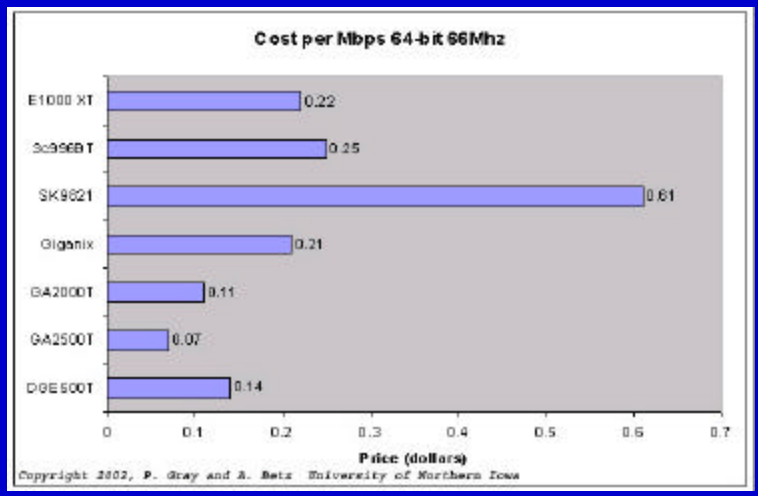
Cost per Mbps 32bit/33MHz



Cost per Mbps 64bit/33MHz



Cost per Mbps 64bit/66MHz



DiSCoV KENT STATE UNIVERSITY 12 January 2004 Paul A. Farrell Cluster Computing 35

Integrating New Capabilities into NetPIPE

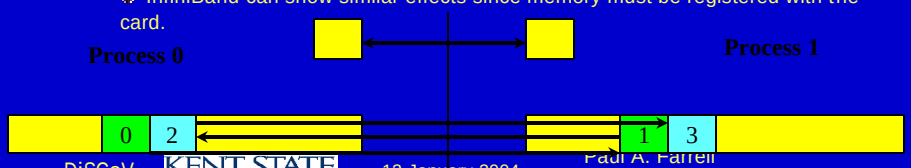
Dave Turner, Adam Oline, Xuehua Chen, and Troy Benjegerdes

Scalable Computing Laboratory of Ames Laboratory
This work was funded by the MICS office of the US Department of Energy

DiSCoV KENT STATE UNIVERSITY 12 January 2004 Paul A. Farrell Cluster Computing 36

Recent additions to NetPIPE

- Can do an integrity test instead of measuring performance.
- Streaming mode measures performance in 1 direction only.
  - Must reset sockets to avoid effects from a collapsing window size.
- A bi-directional ping-pong mode has been added (-2).
- One-sided Get and Put calls can be measured (MPI or SHMEM).
  - Can choose whether to use an intervening MPI_Fence call to synchronize.
- Messages can be bounced between the same buffers (default mode), or they can be started from a different area of memory each time.
  - There are lots of cache effects in SMP message-passing.
  - InfiniBand can show similar effects since memory must be registered with the card.



DiSCoV KENT STATE UNIVERSITY 12 January 2004 Paul A. Farrell Cluster Computing 37

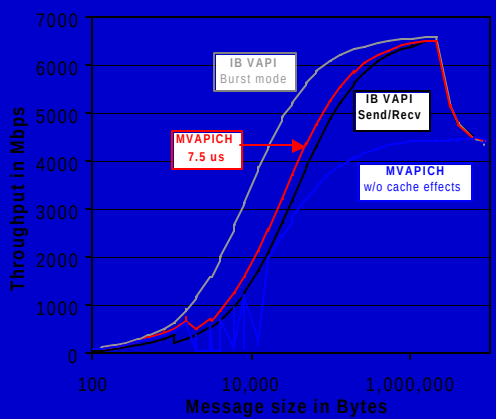
Performance on Mellanox InfiniBand cards

A new **NetPIPE** module allows us to measure the raw performance across InfiniBand hardware (**RDMA** and **Send/Recv**).

Burst mode preposts all receives to duplicate the Mellanox test.

The **no-cache performance** is much lower when the memory has to be registered with the card.

An **MP_Lite** InfiniBand module will be incorporated into LAM/MPI.



MP_Lite on IT

DiSCoV KENT STATE UNIVERSITY 12 January 2004 Paul A. Farrell Cluster Computing 38

10 Gigabit Ethernet

Intel 10 Gigabit Ethernet cards

133 MHz PCI-X bus

Single mode fiber

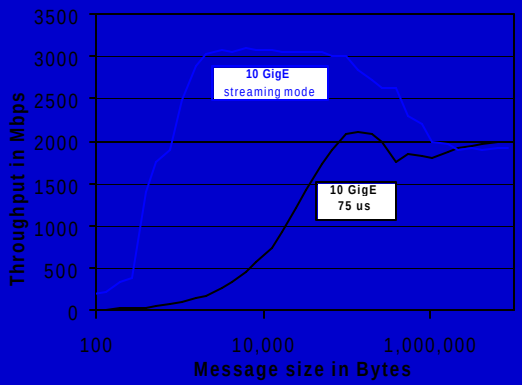
Intel **ixgb** driver

Can only achieve **2 Gbps** now.

Latency is **75 us**.

Streaming mode delivers up to **3 Gbps**.

Much more development work is needed.



DiSCoV KENT STATE UNIVERSITY 12 January 2004 Paul A. Farrell Cluster Computing 39

Channel-bonding Gigabit Ethernet for better communications between nodes

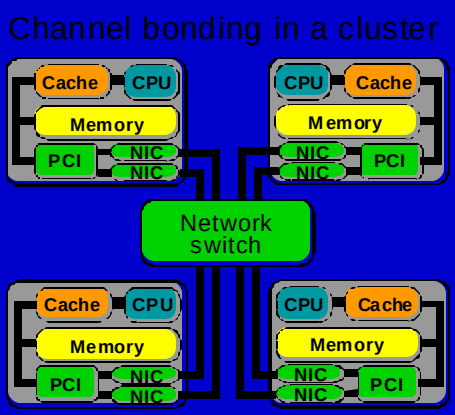
Channel-bonding uses 2 or more Gigabit Ethernet cards per PC to increase the communication rate between nodes in a cluster.

GigE cards cost ~$40 each.

24-port switches cost ~$1400.

➔ $100 / computer

This is much more cost effective for PC clusters than using more expensive networking hardware, and may deliver similar performance.



DiSCoV KENT STATE UNIVERSITY 12 January 2004 Paul A. Farrell Cluster Computing 40

Performance for channel-bonded Gigabit Ethernet

GigE can deliver 900 Mbps with latencies of 25-62 us for PCs with 64-bit / 66 MHz PCI slots.

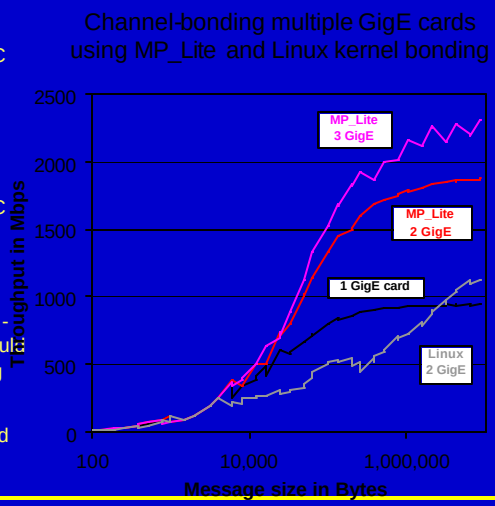
Channel-bonding 2 GigE cards / PC using MP_Lite doubles the performance for large messages.

Adding a 3rd card does not help much.

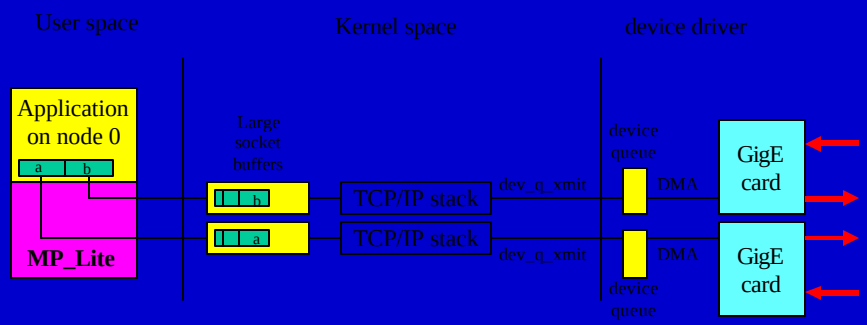
Channel-bonding 2 GigE cards / PC using Linux kernel level bonding actually results in poorer performance.

The same tricks that make channel-bonding successful in MP_Lite should make Linux kernel bonding working even better.

Any message-passing system could then make use of channel-bonding on Linux systems.



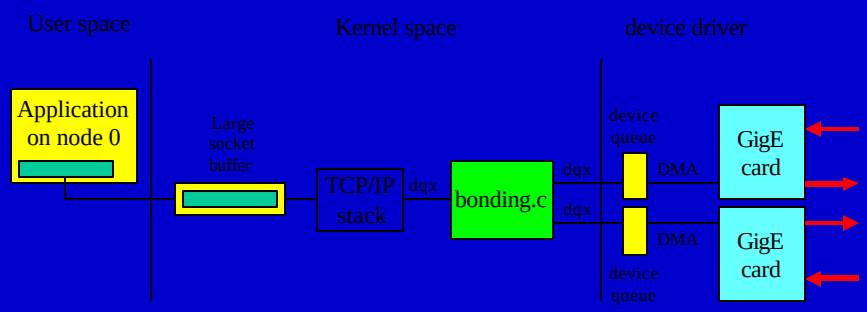
Channel-bonding in MP_Lite



Flow control may stop a given stream at several places.

With MP_Lite channel-bonding, each stream is independent of the others.

Linux kernel channel-bonding



A full device queue will stop the flow at bonding.c to both devi ce queues.

Flow control on the destination node may stop the flow out of th e socket buffer.

In both of these cases, problems with one stream can affect both streams.

DiSCoV KENT STATE UNIVERSITY 12 January 2004 Paul A. Farrell Cluster Computing 43

Comparison of high-speed interconnects

InfiniBand can deliver **4500 - 6500 Mbps** at a **7.5 us** latency.

Atoll delivers **1890 Mbps** with a **4.7 us** latency.

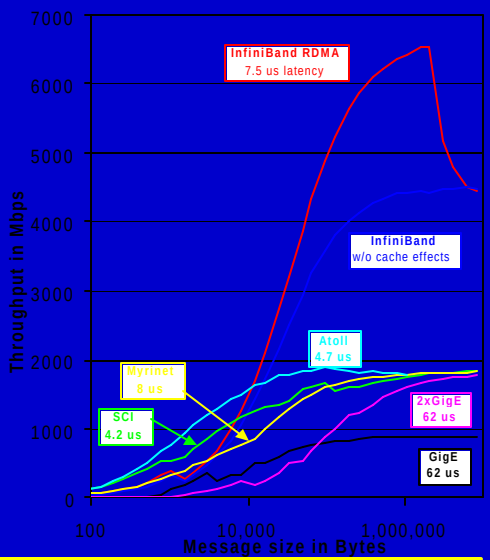
SCI delivers **1840 Mbps** with only a **4.2 us** latency.

Myrinet performance reaches **1820 Mbps** with an **8 us** latency.

Channel-bonded GigE offers **1800 Mbps** for very large messages.

Gigabit Ethernet delivers **900 Mbps** with a **25-62 us** latency.

10 GigE only delivers **2 Gbps** with a **75 us** latency.

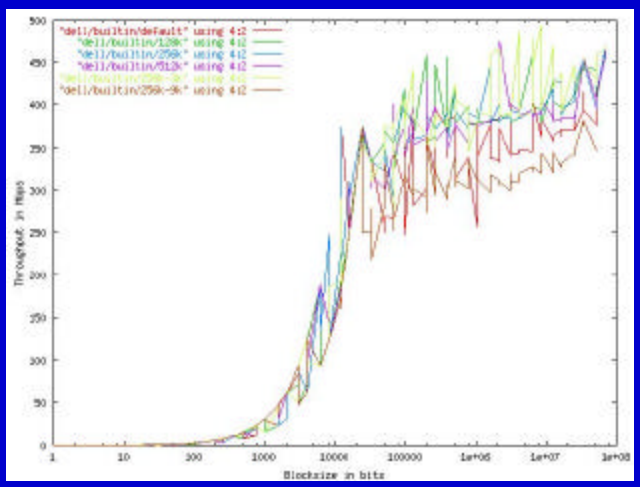


DiSCoV KENT STATE UNIVERSITY 12 January 2004 Paul A. Farrell Cluster Computing 44

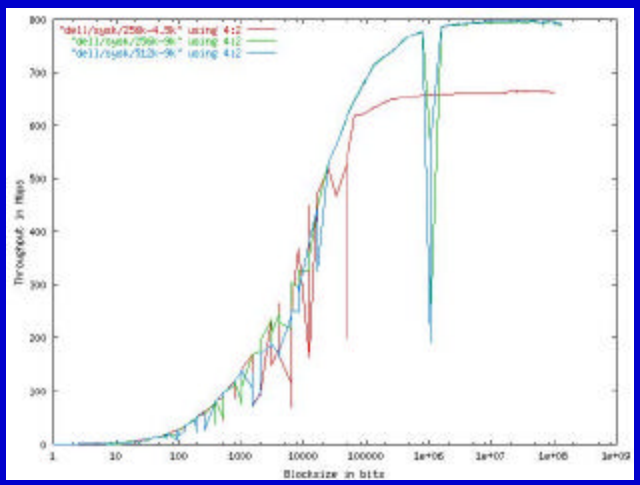
Recent tests at Kent

- Dell Optiplex GX260 - Fedora 10.0
  - Dell motherboard
  - Builtin GE
    - Intel 82540EM (rev 02) built-in GE
      - PCI revision 2.2, 32-bit, 33/66 MHz
    - Linux e1000 driver
  - SysKonnect SK9821 NIC
- RocketCalc Xeon 2.4 GHz
  - SuperMicro X5DAL-G motherboard
  - Intel 82546EB built-in GE
    - 133MHz PCI-X bus
    - http://www.intel.com/design/network/products/lan/controllers/82546.htm
  - Linux e1000 ver 5.1.13
- RocketCalc Opteron
  - Tyan Thunder K8W motherboard
  - Broadcom BCM5703C builtin GE on PCI-X Bridge A (64bit?)
  - tg3 driver in kernel

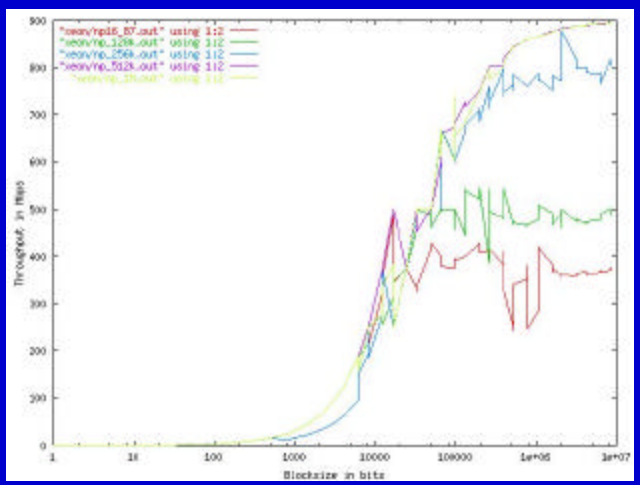
Dell Optiplex GX260 – builtin



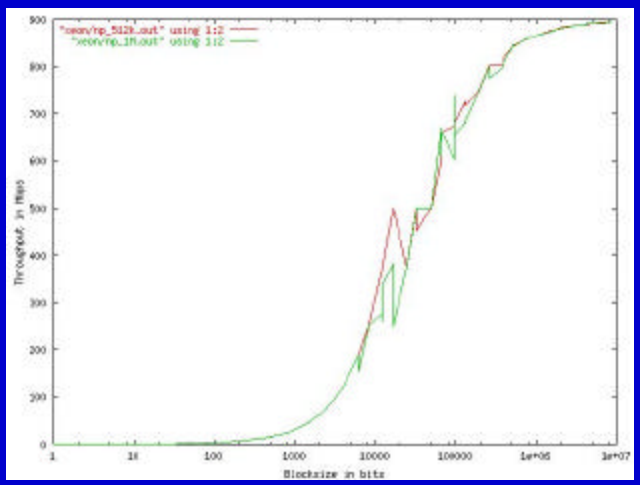
Dell Optiplex GX260 - SysKonnect



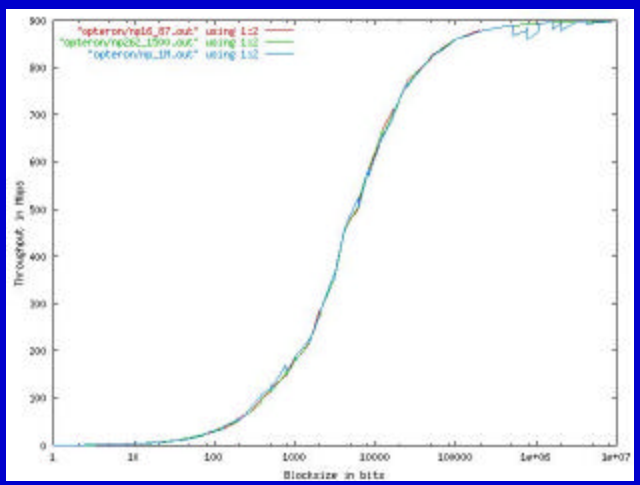
Xeon



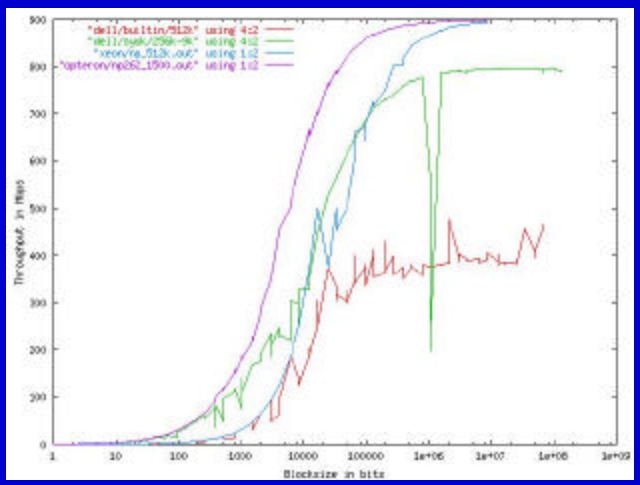
Xeon



Opteron



Summary


"dell/builtin/512k" using 4:2
"dell/bak/256k-5k" using 4:2
"xeon/re_512k.out" using 1:2
"opteron/re262_1500.out" using 1:2
Throughput in Mbps
Blocksize in bits
0
100
200
300
400
500
600
700
800
900
1
10
100
1000
10000
100000
1e+06
1e+07
1e+08