



Silicon Graphics, Inc.

Tools & Libraries

Presented by:
CJ Suchyta, III, Ph.D.
SGI Benchmarking Specialist

October 19, 2005



Software Development Environment

Intel® Compilers and debuggers:

- ifort (supports Fortran 77, 90 and 95)
- icc (C and C++)
- Current version is 9.0

Other alternatives:

- Gnu tools, gcc, g77 and g++, come with the SGI Linux™ Environment
- ORC, the Open Research Compiler: based on SGI Pro64™:
<http://ipf-orc.sourceforge.net/>

Software Development Environment (cont'd)

- Debuggers
 - gdb, Intel® idb,, Intel® Trace Collector, Intel® Thread Checker
 - Etnus® TotalView® (<http://www.etnus.com/>)
- Performance Analysis
 - profile.pl (pfmon), SGI® Histx
 - Intel® VTune™
 - SGI Open|SpeedShop™ to be released in 2006
 - Intel® Trace Analyzer, Intel® Thread Profiler
- System Analysis
 - SGI® Performance Co-Pilot™
- Libraries
 - SGI®: Message Passing Toolkit (MPT)
 - Scientific Computing Software Library (SCSL), cpuset, FFIO
 - Intel®: Math Kernel Library (MKL)
 - Intel® Performance Primitives (support a/v codecs and vector math)

Profiling Tools

pfmon

- Open source performance monitoring tool
- Collects, analyzes, and displays Itanium[®] performance data for Linux systems
- Handles both user- and kernel-level profiling
- System-wide profiling
- Very low overhead
- Version 2.0 compatible with Linux 2.4 kernels
- Version 3.0, with many enhancements, compatible with Linux 2.6 kernels

Profiling Tools (cont.)

profile.pl

- A perl script that uses `pfmon` and auxiliary scripts to perform procedure-level profiling of an unmodified binary
- Useful for both user-time and system-time application profiling; can also be used to profile the Linux kernel
- Handles multiprocessor codes by invoking `pfmon` in system-wide mode and applying a `cpu mask`; accepts `dplace` flags as arguments to facilitate pinning processes to cpus
- Works on statically-linked executables
- Shipped with ProPack; see man page for details

Profiling Tools (cont.)

lipfpm (Intel® Itanium® Processor Family Performance Monitor for Linux®) perfex like event counting tool. Program wide stats.

histx HISTogram eXecution profiling tool. ip and callstack sampling from timer events and performance counter overflows; handles pthreads, OpenMP™, fork, exec. Subroutine stats.

samppm repeatedly samples events during application execution

iprep, csrep filters to convert Histx output into reports.

dumppm: parses samppm output into human-readable form

lipfpm and **histx** do not work with statically linked executables or libraries.

Histx package available from: http://www.sgi.com/products/evaluation/altix_histx/

lipfpm example

- % lipfpm -c bw stream.1

• Function	Rate (MB/s)	Avg time	Min time	Max time
• Copy:	3188.8937	0.0216	0.0216	0.0217
• Scale:	3154.0994	0.0218	0.0218	0.0219
• Add:	3784.2948	0.0273	0.0273	0.0274
• Triad:	3822.2504	0.0270	0.0270	0.0272

- lipfpm summary

- =====

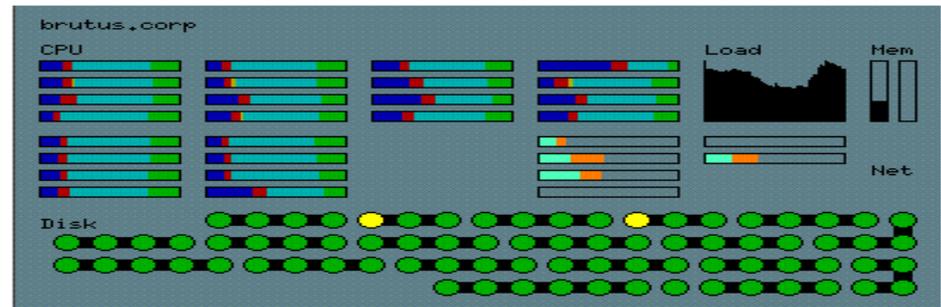
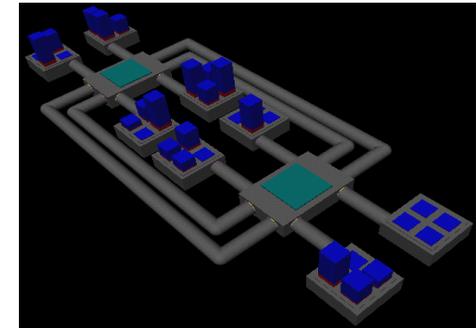
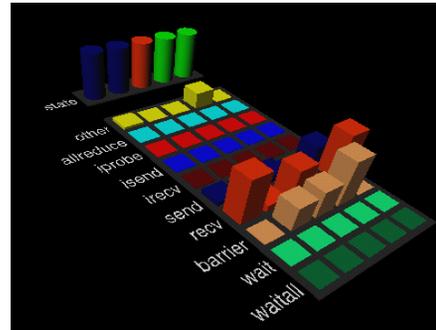
- L1 Data Cache Read Misses -- all L1D read misses will be counted..... 10791782
- L2 Misses..... 55595108
- L3 Reads -- L3 Load Misses (excludes reads for ownership used to satisfy stores)..... 55252613
- CPU Cycles..... 3022194261
- Average read MB/s requested by L1D..... 342.801
- Average MB/s requested by L2..... 3531.96
- Average data read MB/s requested by L3..... 3510.2

Open|SpeedShop™

- What is Open|SpeedShop™?
 - Open Source Performance Tool Comprised of
 - Based on SGI® Speedshop™ software concepts for the IRIX® operating system
 - Developed for Multi-Platform Support
 - Implemented Using Open Source Components
 - DOE/NNSA co-funded via ASC (Advanced Simulation and Computing Program) PathForward
 - Partnering with Universities of Wisconsin & Maryland
- Schedule
 - Project Began: July, 2004
 - Quarterly Milestone Releases
 - First Development Phase Ends: mid-2006

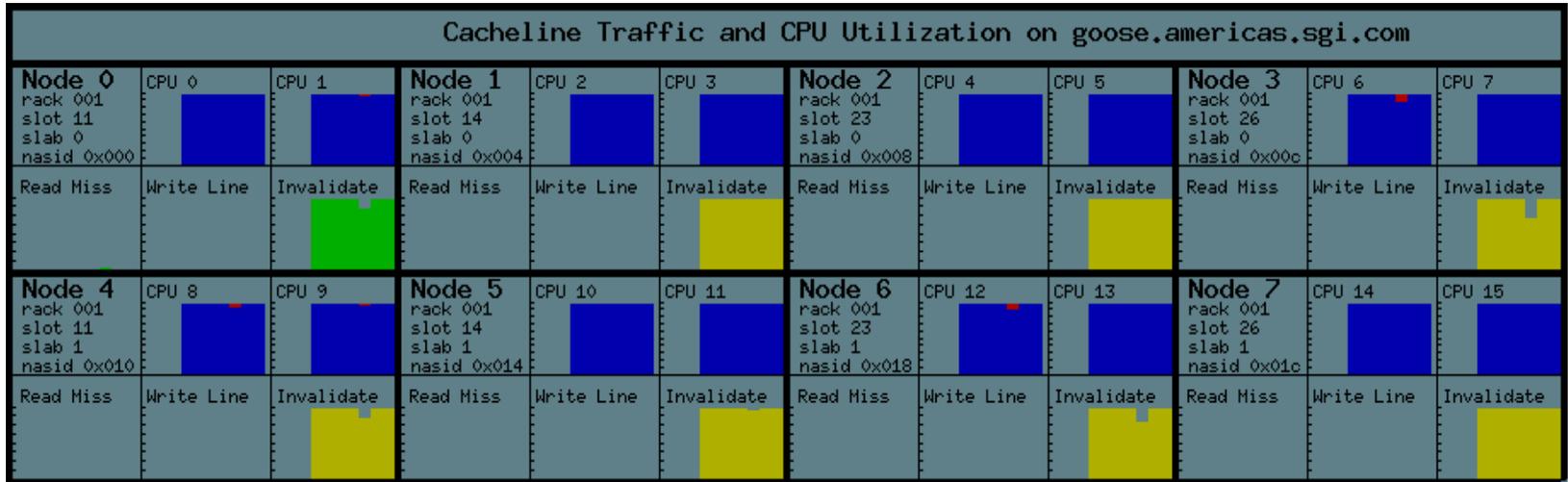
SGI® Performance Co-Pilot™

- System-level performance monitoring and management tool
- Command-line and graphical interfaces
- Distributed and scalable
- Extensive number of performance metrics
- See *Performance Co-Pilot™ for IA-64 Linux User's and Administrator's Guide* at <http://techpubs.sgi.com/>



CacheLine Traffic and CPU Utilization on goose.americas.sgi.com											
Node 0 rack 001 slot 11 slab 0 nasid 0x000	CPU 0	CPU 1	Node 1 rack 001 slot 14 slab 0 nasid 0x004	CPU 2	CPU 3	Node 2 rack 001 slot 23 slab 0 nasid 0x008	CPU 4	CPU 5	Node 3 rack 001 slot 26 slab 0 nasid 0x00c	CPU 6	CPU 7
Read Miss	Write Line	Invalidate	Read Miss	Write Line	Invalidate	Read Miss	Write Line	Invalidate	Read Miss	Write Line	Invalidate
Node 4 rack 001 slot 11 slab 1 nasid 0x010	CPU 8	CPU 9	Node 5 rack 001 slot 14 slab 1 nasid 0x014	CPU 10	CPU 11	Node 6 rack 001 slot 23 slab 1 nasid 0x018	CPU 12	CPU 13	Node 7 rack 001 slot 26 slab 1 nasid 0x01c	CPU 14	CPU 15
Read Miss	Write Line	Invalidate	Read Miss	Write Line	Invalidate	Read Miss	Write Line	Invalidate	Read Miss	Write Line	Invalidate

pmshub



- To use:
pmshub -A (activation)
pmshub (run)
 - One node is doing local accesses, the rest remote: Color Code
 - Blue: user time
 - Red: system time
 - Green: local accesses
 - Yellow: remote accesses
- Need to be on machine, only one user at a time.

Profiling Tools from Intel®

- Intel® VTune™ Performance Analyzer
 - Collects, analyzes, and displays performance data for Windows® and Linux® systems
 - Applications (both single- and multi-threaded)
 - System-wide profile
 - No special build required
 - Very low overhead
 - Linux remote sampling, graphical Windows display
- Intel® VTune™ Performance Analyzer for Linux
 - Native Linux command line interface
 - Sampling and callstack experiments

Data Placement

- Processes drift from CPU to CPU.
- Memory stays on a node once it is physically allocated.
Poor data locality => poor performance.
- How do you avoid poor data locality caused by process drift?
Pin the process on a CPU.
- **dplace** binds processes to CPUs; once pinned they do not migrate. It is cpuset aware.

Running applications efficiently

- **Launching jobs with NUMATOOLS**

dplace (a generic tool for thread-node binding)

- OpenMP: `dplace -c0-255 -x2 ./a.out`
- MPI: `mpirun -np 256 dplace -c0-255 -s1 ./a.out`

- **MPI also can do this automagically via environment variable: `MPI_DSM_DISTRIBUTED` See “man mpi”**

Math and Scientific Libraries

SGI® Scientific Computing Software Library

SCSL 1.6.1

BLAS

FFT and signal processing

LAPACK

Parallel (OpenMP) version

Sparse solvers

i8 version

ScaLapack

BLACS

Intel® Math Kernel Library (MKL)

BLAS includes sparse BLAS FFT

LAPACK

Vector math library (VML)

Message Passing Toolkit MPT 1.12

- MPI 1.2 Standard with MPI-2 features
- shmem (cannot use shmem between partitions)
- shared memory optimizations
- single copy optimizations, direct copy send/recv
- Automatic detection of best communication layer (NUMALink, IB, TCP/IP).
- Supports hybrid programming paradigm (OpenMP+MPI)

MPI Performance (64P 1.5 GHz NL4 Bx2)

MPI Send/Receive latency

CPU 0-1: 1.1 μ sec
off-node (2 hops): 1.8 μ sec
CPU 0-63 (5 hops): 2.0 μ sec

Peak MPI Bandwidth

1400 MByte/sec

Bidirectional Bisection Bandwidth per CPU

790 MByte/sec/CPU

NUMAlink™ - Superior System Interconnect

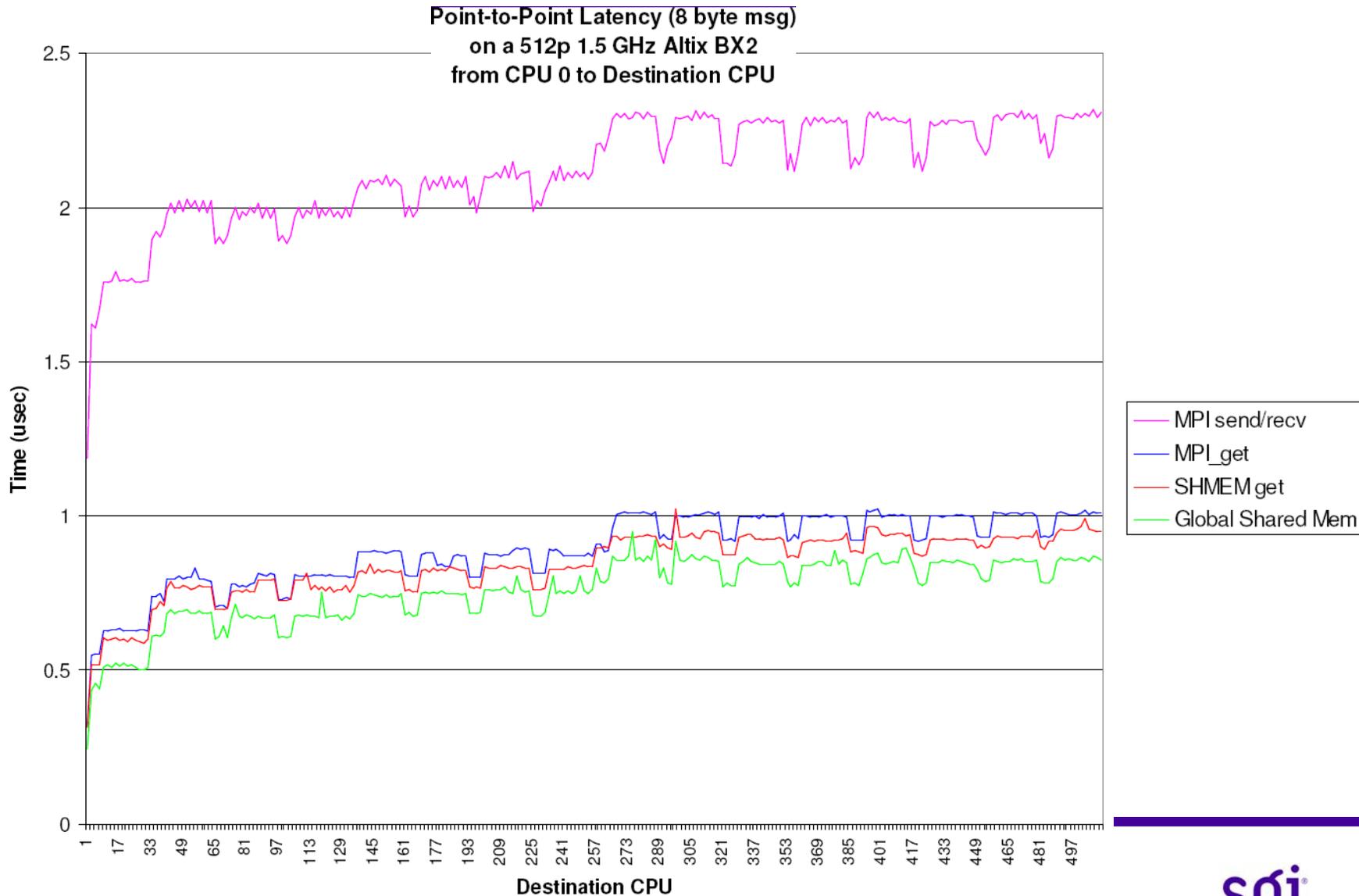
Technology	Vendor	MPI latency, usec, short msg	Bandwidth per link (unidirectional, MB/s)
NUMAlink 4 (Altix)	SGI	1	3200
RapidArray (XD1)	Cray	1.8	2000
QsNet II	Quadrics	2	900
Infiniband	Voltaire	3.5	830
High Performance Switch (Federation)	IBM	5	1000
Myrinet XP2	Myricom	5.7	495
SP Switch 2	IBM	18	500
Ethernet	Various	30	100

Source: SGI Whitepaper: SGI NUMAlink Evolution and Performance Benefits

| October 19, 2005 | Page 17



Point-to-Point Latency on a 512P SGI Altix



Many Choices

- Automatic parallelization
 - Intel compilers. Capable of automatically exploiting some limited forms of parallelism in serial codes
- OpenMP
 - Add preprocessor directives to code to achieve implicit parallelism
 - Ability to exploit incremental parallelism
- Native POSIX Threading Library (NPTL)
- Message Passing Interface (MPI)
 - SGI Message Passing Toolkit (MPT)
 - The default MPI library installed in /usr/lib.
 - No need for special compiler commands (e.g. MPICC) just link with -lmpi
 - Open source alternatives are available (MPICH, LAM, etc.) but are slower than SGI MPT.
 - Uses shared memory and global shared memory (XPMEM)
- SHMEM - Single sided message passing API supported in SGI MPI

sggi[®]