Benchmark Design for Characterization of Balanced High-Performance Architectures

Alice E. Koniges

Lawrence Livermore National Laboratory,
koniges@llnl.gov *www.rzg.mpg.de/~ack*

Rolf Rabenseifner

High-Performance Computing-Center Stuttgart (HLRS), University of Stuttgart,
rabenseifner@hlrs.de *www.hlrs.de/people/rabenseifner*

Karl Solchenbach

Pallas GmbH Hermülheimer Straße 10, D-50321 Brühl, Germany
solchenbach@pallas.com *http://www.pallas.com*

IPDPS 2001
Workshop on Massively Parallel Computing
San Francisco, Apr. 23 - 27, 2001

H L R S

pallas

Outline

- Goals
- Survey of available benchmarking
- Definition of the b_eff and b_eff_io benchmarks
- Results
- Summary
- Future plans

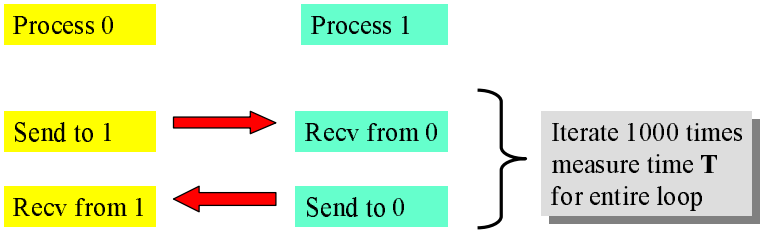
H L R S

pallas

Goals for Communication and File-I/O Benchmarking

- Measure the time needed for exchange of information between
  - processes themselves, and
  - processes and disk
- Model the message passing patterns of real applications
- Provide a number for quick comparison of different systems
- Can't just measure simple send/receive or one I/O access:
  - Clock resolution
  - I/O caching
- So, traditional approach is to measure loops over specific patterns and quote e.g.,
  - 1) Ping-Pong Bandwidth
  - 2) Bi-Section Bandwidth
  - 3) Maximum I/O Bandwidth

Simple example: the ping-pong benchmark:



Result: Time for single message = **T** / 2000

Problems with this approach (Hempel)

- Receiver in ping-pong is always ready to receive
  - receive in 'solicited message' mode
  - delays or intermediate copies can be avoided
- Effects of contention on the network are not seen (only two processes)
- Data may be cached between loop iterations:
- Point-to-point performance is very sensitive to
  - relative timing of send / receive
  - the protocol (dependent on message length)
  - contention / locks
  - cache effects
  - Ping-pong results have very limited value

For details, see: Rolf Hempel: Basic message passing benchmarks, methodology and pitfalls. SPEC Workshop on Benchmarking Parallel and High-Performance Computing Systems, Wuppertal, Germany, Sept. 13, 1999.

Popular Parameters for Describing Systems

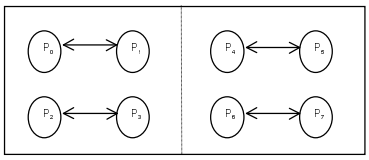
- **Latency:** time for 0 byte message
  - Problem-message passing implementations use different protocols for different size messages. Difficult to extrapolate a meaningful latency for good MPI implementations.
- **Bandwidth:** asymptotic throughput for long messages $b = b_{\infty}$
  - Problem -Not realistic for real application codes
- **Bisection Bandwidth:** The rate at which communication can take place between one half of a computer and the other.
  - Problem -Can depend heavily on processor grouping (see next slide)
- **R = R_{max}** (LINPACK Value) used in Top500
  - Limited-would like to know the balance between this application speed estimate and other aspects including communication scalability and I/O performance
- **R = p × SPEC-rate**
  - Problem-Not realistic for real applications

Effect of Processor Grouping

A good benchmark should NOT allow the vendor to obtain excellent results by limiting study to only best case groupings. Communication patterns should be representative of real applications.

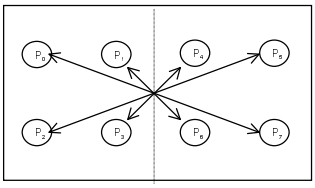
Nominal best case grouping

SMP-node 1 SMP-node 2



Nominal worst case grouping

SMP-node 1 SMP-node 2



What about an I/O Benchmark? Starting-Points:

- Application benchmarks
  - using real, I/O-intensive applications
- File system benchmarks
  - measuring several parameters around the most friendly disk-usage-pattern
- Hardware benchmarks
  - maximum bandwidth of the disk — special-benchmark
- Why a new benchmark for parallel I/O?
  - application / file system / hardware independent
  - but, average on possible application scenarios
  - portable

  ==> MPI-I/O based benchmark

Starting-Points — the I/O Parameter Space

- How to define and measure one characteristic I/O bandwidth value?
- The I/O parameter space — 20 orthogonal parameters:
  - **Application parameters:**
    - (a) the size of contiguous chunks in the memory, (b) on disk, (c) ... (f)
  - **Usage aspects:**
    - (a) how many processes are used
    - (b) how many parallel processors and threads are used for each process.
  - **I/O interface:**
    - (a) Posix I/O buffered or (b) raw,
    - (c) special filesystem I/O of the vendor filesytem,
    - (d) MPI-I/O.
  - **MPI-I/O aspects:**
    - (a) access methods, i.e., first writing of a file, rewriting or reading, (b) ...
    - (c) coordination, i.e., collectively or noncollectively, (d) ... (f)
  - **Filesystem parameters:**
    - (a) which filesystem is used,
    - (b) how many nodes are used as I/O servers, (c) ... (f)

  (full list, see paper)

Existing I/O Benchmarking Techniques

- An example of I/O benchmarking papers:

  "Performance of the IBM General Parallel File System,"
  Terry Jones, Alice Koniges, R. Kim Yates,
  Proceedings of the International Parallel and Distributed
  Processing Symposium, May 2000. Also available as UCRL JC135828

- many hours of dedicated benchmarking time is used
- characterizing a specific system
- not portable
- Rule: Balanced HPC systems should be able to write the total memory in 10 minutes to disk

  ==> An I/O benchmark should not need hours!

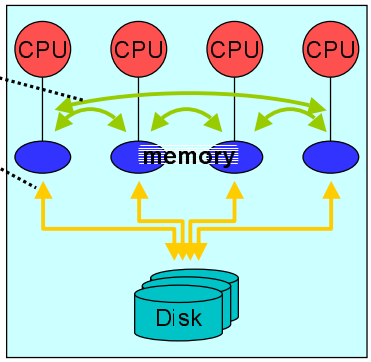
  — 10 minutes may be enough to overrun any cache!

Effective Communication & I/O Bandwidth Benchmarks

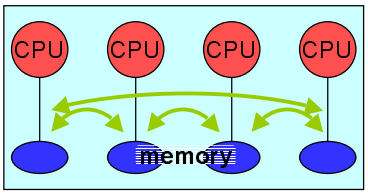
Goals

- Parallel **Communication** Benchmark
- Parallel **File-I/O** Benchmark

  each process is involved!
- Detailed insight
  - bandwidth experiments of several
    - I/O or communication patterns
    - chunk or message sizes
- One characteristic value
  - based on experiments above
  - averaging
- Appropriate execution time for rapid benchmarking



Definition of the Effective Communication Bandwidth Benchmark: b_eff



- www.hlrs.de/mpi/b_eff/
- **Authors:** **Karl Solchenbach, Hans-Joachim Plum, and Gero Ritzenhoefer (Pallas), Rolf Rabenseifner (HLRS)**

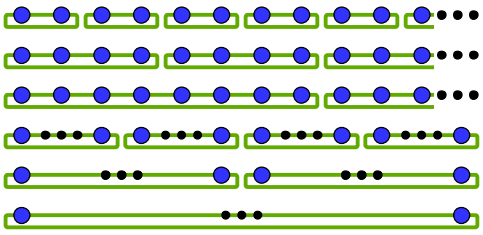
- 6 ring patterns
- 30 random patterns
- 13 additional patterns
- 21 message sizes
- 3 communication methods
- 3 times repeated

(6+30+13) x 21 x 3 x 3 = 9261 experiments

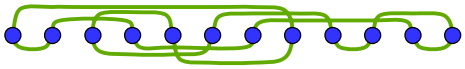
- 5 - 20 msec / experiment —> benchmark completes in a few minutes

Definition of b_eff — communication patterns and sizes

- **6 ring patterns**
  - ring size = 2
  - 4
  - 8
  - max(#PE/4, 16)
  - max(#PE/2, 32)
  - #PE



- **30 random patterns**



- **21 message sizes**
  - 1, 2, 4, 8, 16, 32, 64, 128, 256, 512 byte, 1kB, 2kB, (12 sizes)
  - 9 logarithmic equidistant sizes: 4kB, ..., L_{max}= memory per PE / 128

Definition of b_eff — averaging

One characteristic accumulated communication bandwidth number
:= average bandwidth on several communication patterns
average on different message sizes
maximum over different MPI programming methods

$$b_eff = \text{logavg}\Big(\text{logavg}_{\text{ring pat}} \big(\text{avg}_{L}\big(\max_{\text{method}}\big(\max_{\text{rep}}(b_{\text{pat,L,method,rep}}) \big) \big) \big), \; \text{logavg}_{\text{random pat}} \big(\text{avg}_{L}\big(\max_{\text{method}}\big(\max_{\text{rep}}(b_{\text{pat,L,method,rep}}) \big) \big) \big) \Big)$$

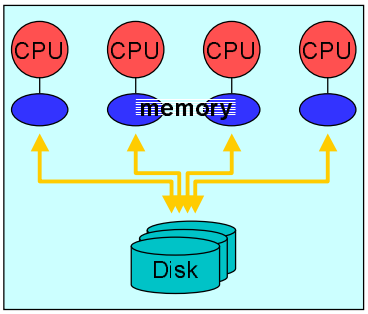
with

- $b_{\text{pat,L,method,rep}}$ = accumulated bandwidth of each experiment
  over all processes
- methods: MPI_Sendrecv, MPI_Alltoallv, and nonblocking Irecv&Isend&Waitall
- pat & L: patterns and message sizes, see previous slide
- rep: repetition number = 1..3
- avg: arithmetic mean
- logavg: geometric mean

Features of Effective Bandwidth benchmark

- Based on MPI, source code is available
- Measures total architecture, not only point-to-point
- Checks performance of architecture and not the quality of the MPI implementation
- Suited for MPP-architectures and clusters
- Runs on any number of processors
- Results are easy to understand
- Generates a single number b_{eff} (like LINPACK R_{max})

Definition of the Effective File-I/O Bandwidth Benchmark: b_eff_io



- 5 I/O patterns
- 7 chunk sizes
- 3 accesses (initial write, rewrite, read)
- 3 compute partition sizes (number of parallel benchmark processes)
- benchmark completes in ~30 minutes
- www.hlrs.de/mpi/b_eff_io/

Definition of b_eff_io

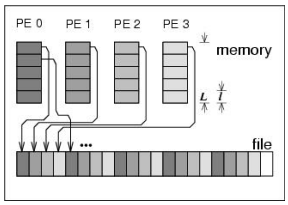
(Release 1.0)

b_eff_io := Maximum over all usage and filesystem parameters } manually

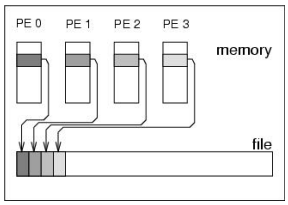
Average on write, rewrite, read
Average on five access pattern types
Average on several chunk size values*)
of measured bandwidth
} auto-matically, in time T=30 min.

*) defines the size of contiguous chunks written to disk and the contiguous chunk in memory written by each MPI call

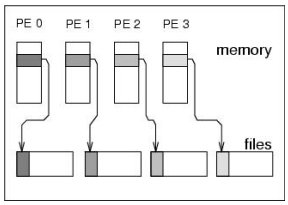
Definition of b_eff_io — the Pattern Types



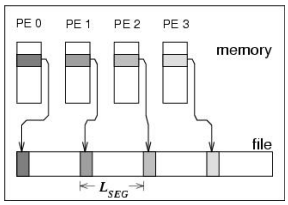
type 0: scattering *)



type 1: strided coll.



type 2: noncoll., separated



type 3 noncoll. / 4 coll. segmented

Pattern that can be optimized

Chunk sizes on disk:
- max (2MB, memory of one node/128) *)
- wellformed: 1MB, *) 32 kB, 1 kB,
- non-wellformed: 1MB+8B, *) 32 kB+8B, 1kB+8B

*) double weighted

Definition of b_eff_io — Bandwidth measurement

- Bandwidth measurement

```
MPI_Barrier()
start_time = MPI_Wtime()   at root only
repeat
   MPI_File_write() or MPI_File_read()
   MPI_Barrier()
   conti = (MPI_Wtime()- start_time) < time_unit
   MPI_Bcast(conti)
while conti
if (write access) MPI_File_sync()
MPI_Barrier()
end_time = MPI_Wtime()   at root only
bandwidth = (accumulated data size)
            / (end_time - start_time)
```


Output of the b_eff_io benchmark program

- the b_eff_io value

weighted average bandwidth for **write** : 21.530 MB/s on 16 processes
weighted average bandwidth for **rewrite** : 29.472 MB/s on 16 processes
weighted average bandwidth for **read** : 93.602 MB/s on 16 processes
Total amount of data written/read with each access method: 17589.682 MBytes
= 26.8 percent of the total memory (65536 MBytes)
b_eff_io of these measurements = 59.552 MB/s
on 16 processes with 128 MByte/PE and scheduled time=30.0 min
on sn6715 hwwt3e 2.0.5.34 unicosmk CRAY T3E
total memory / b_eff_io = 65536 Mbytes / 59.552 MB/s **= 18.3 min**.

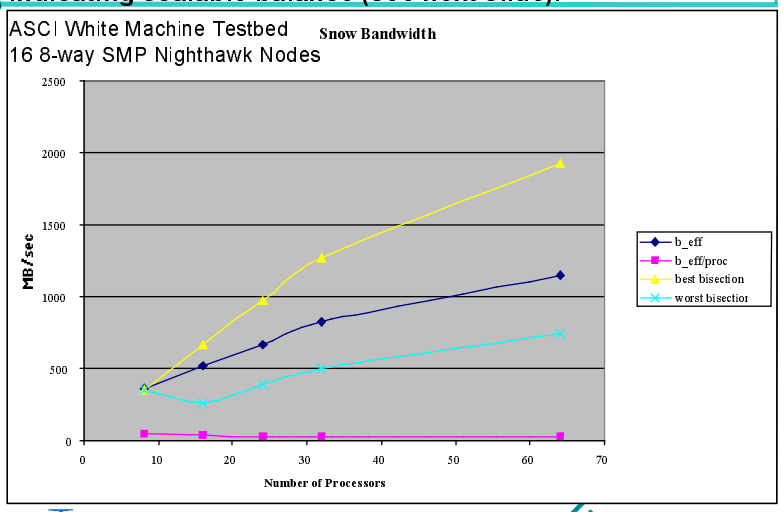
- detailed results
  - as ASCII table
  - one page with 3+5 plots
    - **all measurements sorted by access:** write / rewrites / reads
    - **and same sorted by pattern types:** type-0 / type-1 / type-2 / type-3 / type-4

Time-driven approach

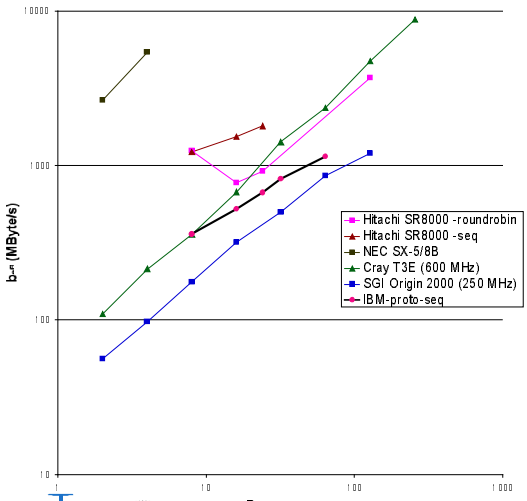
- **b_eff**
  - for each message size:
    loop length is based on
    execution time of next smaller message size
  - starting loop length for each pattern and method
    = 300 (release <= 3.3)
    = based on a quick latency measurement with 10 iterations (rel.>3.3)
- **b_eff_io**
  - first write
    & pattern types 0-2 (**scatter collective, shared collective, separated files**):
    • writing until scheduled time is over for each pattern and chunk size
  - first write & pattern types 3+4 (**segmented file, collective and not**):
    • pre-calculated repeating factors,
    • based on measured execution times with pattern types 0-2
  - rewrite & read: same amount of data as with *"first write"*

B_eff is monotonic. B_eff/proc is roughly constant indicating scalable balance (see next slide).

ASCI White Machine Testbed
16 8-way SMP Nighthawk Nodes



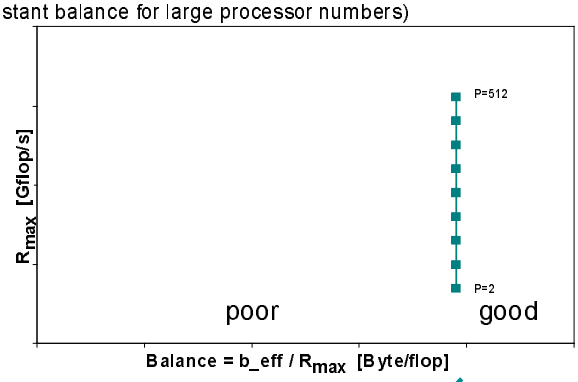
B_eff Scaling: current systems



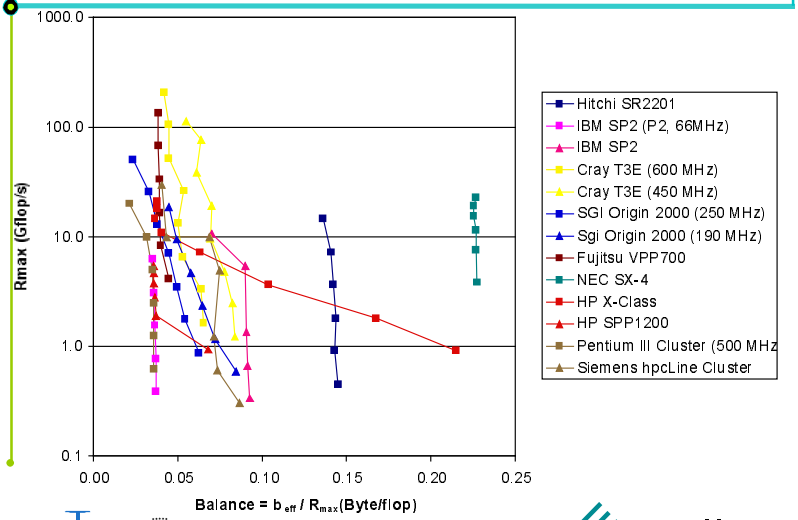
Also define balance as ratio ***balance*** = b_eff / Rmax

Goals for balance:

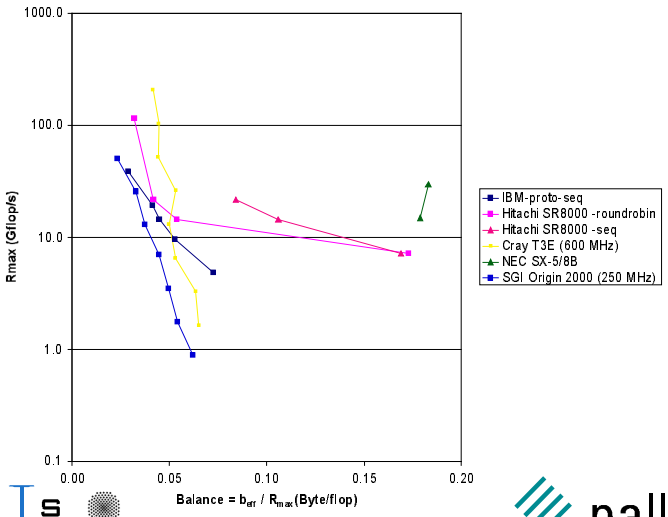
1. Sufficiently large number for target applications
2. Scalable Balance
   (constant balance for large processor numbers)



Balance: data ~ 1998



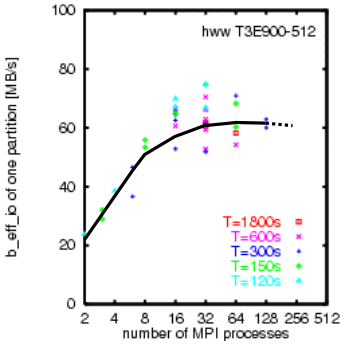
Balance: current systems

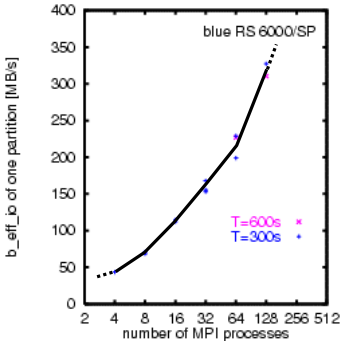


I/O Results — Comparing systems

- Cray T3E 900-512 at HLRS/RUS, Stuttgart
  - 512 processors
  - 10 striped Raid-disks, connected via GigaRing
  - mpt.1.3.0.2 with ROMIO, modified: using asynchronous I/O
    - www.hlrs.de/mpi/mpi_t3e.html#StripedIO
    - www.hlrs.de/mpi/ufs_t3e/
  - theoretical peak throughput = 300 MB/s
- IBM RS 6000/SP at LLNL, called "blue pacific"
  - 336 SMP nodes with each 4 processors
  - benchmark: using 1 processor per node
  - IBM General Parallel File System (GPFS) with 20 VSD I/O server
  - ROMIO
  - measured peak performance: 950 MB/s read, 690 MB/s write (on 128 nodes)
- NEC SX-5Be/32M2 at HLRS/RUS, Stuttgart
  - 2 SMP nodes with each 16 processors
  - benchmark only on one SMP node
  - SFS filesystem, 4MB block size
  - I/O requests less than 1 MB are cached on 2 GB filesystem-memory-cache

First Results — Comparing b_eff_io (#processes)





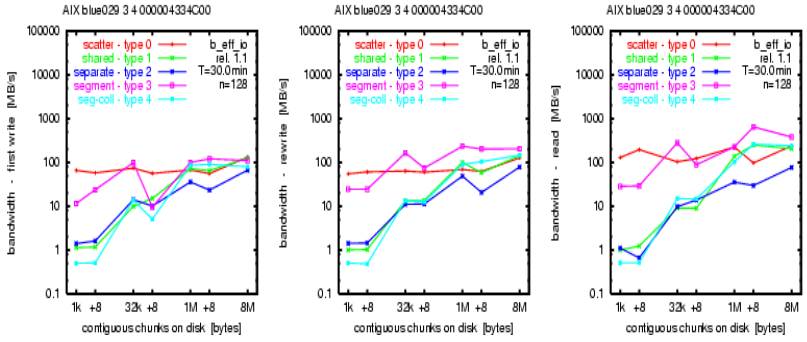
Full bandwidth on **Cray T3E**:
- about 30% of peak performance
- reached already with 8 processors!
  => optimal for any load:
  many small jobs ... one large job

Full Bandwidth **IBM SP**:
- about 35% of peak performance
- reachable only with high-CPU-count jobs
- higher absolute values
  (b_eff_io and total memory size)

First Results — Interpretation

- maximum bandwidth / partition sizes
- small influence of scheduled time T
- benchmarked platforms: MPI-I/O is optimal only for one pattern type
- but different optimal type on each platform
- non-wellformed data sizes: worse I/O bandwidth
- (re)write bandwidth << read bandwidth
- no chance to predict bandwidth for other patterns

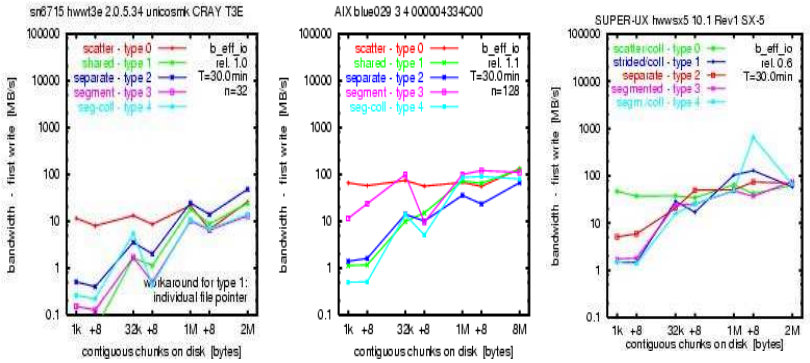
Detailed Results –IBM SP "Blue Pacific" at LLNL



weighted average bandwidth for write: 53.256 MB/s on 128 processes (25%)
weighted average bandwidth for rewrite: 68.252 MB/s on 128 processes (25%)
weighted average bandwidth for read: 65.058 MB/s on 128 processes (50%)

b_eff_io of these measurements = 62.906 MB/s on 128 processes

Results: "write" on Cray T3E, IBM SP, and NEC SX-5



Cray T3E
32 Pes
T=30 min.
(b_eff_io=57 MB/s)

IBM SP
128 nodes
T=30 min.
(b_eff_io=63 MB/s)

NEC SX-5
4 nodes
T=30 min.
(b_eff_io=60 MB/s)

Summary: B_eff_io

Effective I/O Bandwidth Benchmark (b_eff_io)

- characteristic average number for I/O bandwidth
- detailed information about several patterns:
  - access pattern types,
  - buffer sizes,
  - access methods (initial write, rewrite, read)
- 30 minutes for a first pass on any platform

Sample results

- Cray T3E900-512 and NEC SX-5 at HLRS
- IBM RS 6000/SP at LLNL ("blue pacific")

Usable on MPP systems, SMP systems, and on clusters of SMPs

- more info: www.hlrs.de/mpi/b_eff_io/

Summary: B_eff

Effective Communication Bandwidth Benchmark (b_eff)

- characteristic average number for accumulated communication bandwidth
- detailed information about several patterns:
  - ring patterns, random patterns, and some additional patterns,
  - 21 message sizes,
  - transfer methods (sendrecv, alltoallv, and nonblocking Irecv+Isend)
- balance = comparing b_eff with Rmax (LINPACK)
- ~3-5 minutes on any platform

Results on several platforms

Usable on MPP systems, SMP systems, and on clusters of SMPs

- more info: www.hlrs.de/mpi/b_eff/

Outlook

- www.top500clusters.org
- Issues
  - collecting hardware characteristics of clusters
  - several benchmark results
  - stored in a database
  - web-interface
  - each reader can define his own weights, and
  - can receive a personal weighted (ranked) list of all clusters
  - automatic b_eff_io for 3 different numbers of processors
- Status
  - hardware information: some clusters already stored in database
  - benchmarks and web-interface: under discussion
  - b_eff and b_eff_io under evaluation

Acknowledgements

Work by Lawrence Livermore National Laboratory
is performed under the auspices of the U.S. Department
of Energy by the University of California
under Contract W-7405-ENG-48, UCRL-VG-143637.

Further information:

www.hlrs.de/mpi/b_eff
www.hlrs.de/mpi/b_eff_io