Performance of Application and System Benchmarks

Application and System Benchmarks

Lecture Outline

Following Topics will be discussed

- ❖ Peak and Sustained performance
- ❖ Benchmarks
- ❖ Classification of Benchmarks
- ❖ Macro Benchmarks
- ❖ Micro Benchmarks
- ❖ Results of benchmarks on PARAM 10000

Performance Characteristics

Approaches to measure performance

- Several Approaches exist to measure performance of a computer system
  - Summarize key architectural issues of a system and relate
  - Engineering or design considerations rather than theoretical calculations
  - Observe run times for defined set of programs

Performance Characteristics: Peak Performance

Peak Performance

- Measurement in MFLOPS
- Maximum number of operations that the hardware can execute in parallel or concurrently
- Rough hardware measure - reflects the cost of the system
- Rare instances of implication

Performance Characteristics: Sustained Performance

Sustained Performance

- Highest MFLOPS rate that can actual program achieve doing something recognizably useful for certain length of time
- It essentially provide an upper bound on what a programmer may be able to achieve

Efficiency rate = The achieved (sustained) performance divided by the peak performance

Note : The advantage of this number is "it is independent of any absolute speed".

Performance Characteristics: Sustained Performance

(Contd…)

Sustained Performance

How to measure sustained performance of a parallel system?

- Use benchmarks
- Wide Spectrum of Benchmarks in case of parallel computer
- SYSTEM BUYER's choice as per application requirements

Performance: Benchmarks

Benchmark

A benchmark is a test program that

- *supposedly* captures processing and data movement characteristics of a class of applications
- are used to reveal their architectural weak and strong points.
- are significantly more reliable than peak performance numbers.
- **Note** : Sometimes benchmark measurement may be slanted because of an idiosyncrasy of the compiler

(Contd…)

Performance: Benchmarks

Benchmark

A benchmark suite = set of programs + set of rules

- Platform, input data, the output data results, and the performance metrics
- Gives idea of performance and scalability of a parallel system
- Benchmarks can be full-fledged applications or just kernels

CDAC
The Supercomputing People

Performance: Benchmarks Classification

Benchmark Classification

- Benchmarks can be classified according to applications
  - Scientific Computing
  - Commercial applications
  - Network services
  - Multi media applications
  - Signal processing
- Benchmark can also be classified as
  - Micro benchmarks and Macro benchmarks

CDAC
The Supercomputing People

Performance: Micro Benchmarks

(Contd…)

Micro Benchmarks

Micro benchmarks tend to be synthetic kernels. Micro benchmarks measure a specific aspect of computer system.

- CPU speed
- Memory speed
- I/O speed
- Operating system performance
- Networking

Performance: Micro Benchmarks

(Contd…)

Micro Benchmarks

Representative Micro Benchmark Suits.

Name	Area
LINPACK LAPACK ScaLAPACK	Numerical Computing (Linear Algebra)
LMBENCH	System Calls and data movement operations in UNIX
STREAM	Memory Bandwidth

Performance: Macro Benchmarks

Macro Benchmarks

- A macro benchmark measures the performance of computer system as a whole.
- It compares different systems with respect to an application class, and is useful for the system BUYER.
- However, macro benchmarks do not reveal why a system performs well or badly.

(Contd…)

Performance: Macro Benchmarks

Macro Benchmarks

Name	Area
NAS	Parallel Computing (CFD)
PARKBENCH	Parallel Computing
SPEC	A mixed benchmark family
Splash	Parallel Computing
STAP	Signal Processing
TPC	Commercial Applications

Micro Benchmarks: LINPACK

Micro Benchmarks

The LINPACK

- ❖ The LINPACK benchmark was created and is maintained by Jack Dongarra at the University of Tennessee.
- ❖ It is a collection of Fortran subroutines that solve linear system of equations and linear least square problems.
- ❖ Matrices can be general, banded, symmetric indefinite, symmetric positive definite, triangular, and tri-diagonal square.
- ❖ LINPACK has been modified to ScaLAPACK for distributed-memory parallel computers.

Micro Benchmarks: LINPACK

(Contd…)

Objective : Performance of system of Linear equations (LU factorization) using MPI or PVM for a MIMD machine.

- One can use LINPACK (ScaLAPACK version) maintained by Jack Dongarra, University of Tennessee
- Estimated performance equations exist but one should know the following characteristics of any parallel computer
  - Algorithm time
  - Communication time
  - Computation time

Micro Benchmarks: LINPACK

(Contd…)

LINPACK : Optimization

- Arrangement of data in local memory of each process
- Hierarchical memory features' usage for uniprocessor code performance
- Arrangement of matrix elements within each block
- Matrix blocks in the local memory
- Data in each block to be contiguous in physical memory
- Startup sufficiently large - it is preferable to send data as one large message rather than as several smaller messages

Micro Benchmarks: LINPACK

(Contd…)

LINPACK : Optimization

- ❖ Overlapping communication and computation
- ❖ Tradeoff between load imbalance and communication
- ❖ Portability and ease of code maintenance
- ❖ Assignment of processes to processors (topology)

Micro Benchmarks: LAPACK

Micro Benchmarks

LAPACK : (Used for SMP node performance)

- ❖ To obtain sustained performance on one SMP node for Numerical Linear Algebra Computations
- ❖ Highly tuned libraries of Matrix Computations may yield good performance for LAPACK
- ❖ Exploit BLAS Levels 1,2,3 to get performance
- ❖ Underlying concept
  - ➢ Use block partitioned algorithms to minimize data movement between different levels in hierarchical

Micro Benchmarks: LAPACK

(Contd…)

Micro Benchmarks

LAPACK : (Used for SMP node Configuration)

- Goal is to modify EISPACK and LAPACK libraries run efficiently on shared-memory architectures
- LAPACK can be regarded as a successor to LINPACK and EISPACK
- LAPACK gives good performance on current Hierarchical memory computing systems
  - Reorganizing the algorithms to use block operations for matrix computations
  - Optimized for each architecture to account for the memory hierarchy

Micro Benchmarks: ScaLAPACK

ScaLAPACK (Parallel LAPACK)

- Library of functions for solving problems in Numerical Linear Algebra Computations on distributed memory systems
- It is based on library 'LAPACK', ScaLAPACK stands for "Scalable LAPACK".
- LAPACK obtains both portability and high performance by relying on another library the BLAS (Basic Linear Algebra Sub-program Library)
- The BLAS performs common operations such as dot product matrix vector product, matrix-matrix product.

Micro Benchmarks: ScaLAPACK

(Contd…)

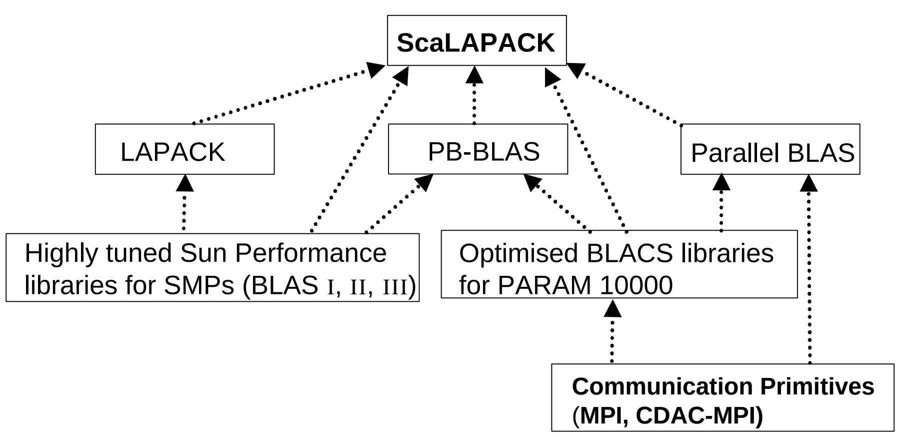
<u>ScaLAPACK (Parallel LAPACK)</u>

The separate libraries are :

- PBLAS (Parallel BLAS)
- BLACS (Basic Communication Sub-programs)
- In order to map matrices and rectors to processes, the libraries (BLACS, PBLAS, and ScaLAPACK) rely on the complementary concepts of process grid and block cyclic mapping.
- The libraries create a virtual rectangular grid of processors much like a topology in MPI to map the matrices and vectors to physical processors

Micro Benchmarks: ScaLAPACK

(Contd…)



Micro Benchmarks: LMbench

- The LMbench benchmark suite is maintained by Larry McVoy of SGI.
  - Focus attention on basic building blocks of many common computing systems' performance issues
  - It is a portable benchmark aiming at attempting to measure the most commonly found performance bottlenecks in a wide range of system applications - latency and bandwidth of data movement among processors, memory, network, file system and disk.
  - It is a simple, yet very useful tool for identifying performance bottlenecks and for the design of a system. It takes no advantage of SMP system. It is meant to be a uniprocessor benchmark.

Micro Benchmarks: LMbench

(Contd…)

- Bandwidth : Memory Bandwidth; Cached I/O Bandwidth
- Latency : Memory read Latency; Memory Write Latency
- Signal Handling Cost
- Process creation costs; Null System Call; Context Switching (To measure System overheads)
- IPC Latency: Pipe, TCP/ RPC/UDP Latency, File System Latency; Disk Latency
- Memory latencies in nanoseconds - smaller is better

  Clock Speed (Mhz) : **292** , L1 Cache : **6**, L2 Cache : **33**, Main memory : **249**

Micro Benchmarks: STREAM

- The Stream is a simple synthetic benchmark maintained by John McCalpin of SGI.
  - It measures sustainable memory bandwidth (in MB/s) and the corresponding computation rate.
  - The motivation for developing the STREAM benchmark is that processors are getting faster more quickly than memory, and more programs will be limited in performance by the memory bandwidth, rather than by the processor's speed.
  - The benchmark is designed to work with data sets much larger than the available cache
  - The STREAM Benchmark performs four operations for number of iterations with unit stride access.

Micro Benchmarks: STREAM

(Contd…)

Name	Code	Byte/ Iteration	Flop/ Iteration
COPY	a(i)=b(i)	16	0
SCALE	a(i)=q x b(i)	16	1
SUM	a(i) = b(i) + c(i)	24	1
TRIAD	a(i) =b(i) + q x c(i)	24	2

- Machine Balance Metric =

$$\frac{\text{Peak floating-point (flop/s)}}{\text{Sustained TRIAD memory bandwidth (word/s)}}$$

- The machine balance metric can be interpreted as the number of flop that can be executed in the time period to read/write a word.
- The machine balance values of many systems have been increasing over the years, implying that memory bandwidth lags more and more behind processor speed.

Micro Benchmarks: STREAM

(Contd…)

Array size = 10000000

Your clock granularity/precision appears to be 1 microseconds.

Stream Results on PARAM 10000

Function	Rate (MB/s)	RMS time	Min time	Max time
Copy:	206.6193	0.7816	0.7744	0.7958
Scale:	206.9323	0.7858	0.7732	0.8030
Add:	211.0202	1.1485	1.1373	1.1730
Triad:	220.6552	1.0994	1.0877	1.1222

LLCBench

- LLCbench is used to determine the efficiency of the various sub-systems that affect the performance of an application.
- The sub-systems are:
  - Memory
  - Parallel processing environment
  - System libraries

LLC : BLAS Bench

- Evaluate compiler efficiency by comparing performance of reference BLAS and hand tuned BLAS of the vendor.
- Evaluate the performance of vendor provided BLAS routines in MFLOPS.
- Provide info for performance modeling of applications that make heavy use of BLAS.
- Evaluate compiler efficiency by comparing performance of reference BLAS and hand tuned BLAS of the vendor.
- Validate vendor's claims about the numerical performance of their processor.
- Compare against peak cache performance to establish bottleneck – memory or CPU.

LLC : Cache Bench

- Evaluate the performance of the memory hierarchy of a computer system.
- Focus on the multiple levels of cache.
- Measures – Raw bandwidth in MBps.
- Combination of 8 different benchmarks on cache and memory

LLC : MPI Bench - Goals

- Evaluate the performance of MPI
- Can be used over any Message passing layer
- Interpretation of results is left to the user
- Uses flexible and portable framework to be able to be used over any message passing layer.
- Tests Eight Different MPI Calls.
  - Bandwidth; Roundtrip; Application Latency;Broadcast
  - Reduce; All Reduce;Bidirectional Bandwidth;All to All

Macro Benchmarks : NAS

The NPB suite

- The NAS Parallel Benchmarks (NPB) is developed and maintained by the Numerical Aerodynamics Simulation (NAS) program at NASA Ames Research Centre.
- The computation and data movement characteristics of large scale Computational Fluid Dynamics (CFD) applications.
- The benchmarks are EP, MG, CG, FT, IS, LU, BT, SP.

Macro Benchmarks: PARKBENCH

PARKBENCH

- The PARKBENCH (PARallel Kernels and BENCHmarks)
- General features
  - For distributed-memory multicomputers, shared-memory architectures coded with Fortran 77.
  - Support PVM or MPI for message passing
  - Fortran 90 and HPF versions
- Types :
  - Low-level benchmarks
  - Kernel benchmarks
  - Compact application and HPF compiler benchmarks

Communication Overheads : P-COMS

P-COMS: Communication Overhead Measurement Suites developed by C-DAC

Point-to-Point Communication calls in MPI	
Collective Communication	Collective Communication and Computation
Broadcast Gather Scatter Total Exchange Circular Shift	Barrier Reduction Scan All-to-All Non-Uniform Data

Communication Overheads : P-COMS

(Contd…)

P-COMS

- Measuring point-point communication (ping-pong test) between two nodes
- Measuring point-point communication involving *n* nodes (Hot-potato test or Fire Brigade Test)
- Measuring collective communication performance
- Measuring collective communication and computation performance

Note : MPI provides rich set of library calls for point-to point and collective communication and computation library calls

Communication Overheads : P-COMS

(Contd…)

P-COMS

Interpretation of overhead measurement data

- Method 1 : Present the results in tabular form
- Method 2 : Present the data as curve
- Method 3 : Present the data using simple, closed-form expression

PARAM 6.4 GF - Consolidated Specifications

Shared Memory	: Node level and run replicated UNIX OS
Architecture Type	: Cluster of SMPs
Nodes	: E250 Enterprise Server- 400 MHz dual CPU
Peak Computing Power	: 6.4 Gigaflops
Aggregate Main Memory	: 2.0 Gbytes
Aggregate Storage	: 8.0 Gbytes
System Software	: HPCC
Networks	: High bandwidth, low latency SANs PARAMNet and FastEthernet
Message Passing library	: MPI
Compilers and Tools	: Sun Workshop

PARAM 10000 Configuration of 8 Processors

- Shared memory at Node level
- Node run replicated UNIX OS
- Nodes connected by low latency high throughput System Area Networks PARAMNet/ FastEtherNet/MyriNet
- Standard Message Passing interface(MPI) and CDAC-MPI
- C-DAC High Performance Computing and Communication software for Parallel Program Development and run time support

(Contd…)

Micro Benchmarks: LAPACK

LAPACK Performance : (Used for SMP node Configuration)

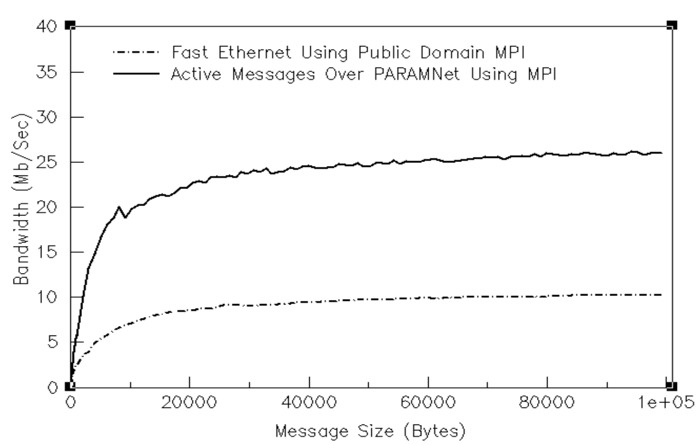
- Several other routines of LAPACK performance varies from **250** Mflop/s to **550** Mflop/s
- **700** Mflop/s on one node having DUAL CPU for some matrix computations (BLAS libraries)

Note : Sun Performance Computing Libraries have been used and the performance, in terms of **Mflop/s** can be improved by choice of matrix size, band size, and using several options of Sun performance libraries etc.

(Contd…)

Communication Overheads: P-COMS

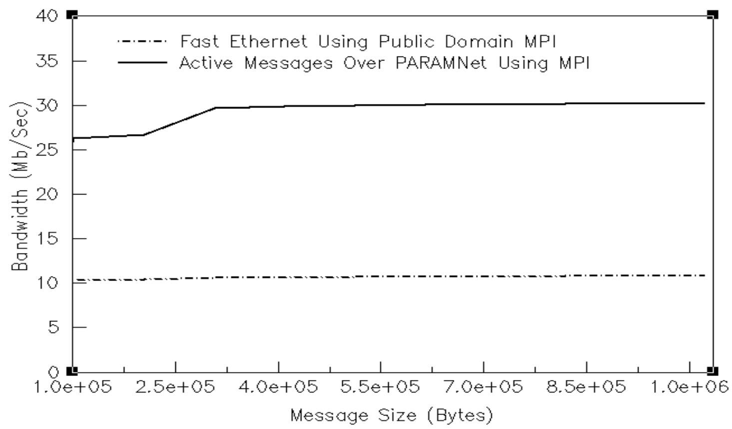
Performance of "**`Round Trip`**" Communication Primitive



Communication Overheads: P-COMS

(Contd…)

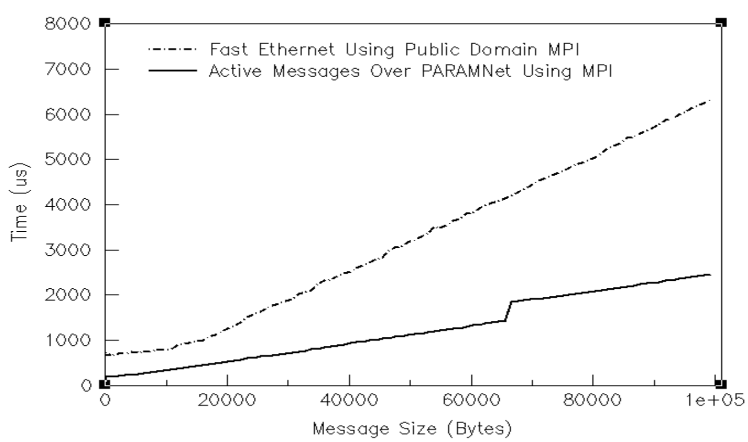
Performance of “**`Round Trip`**” Communication Primitive



Communication Overheads: P-COMS

(Contd…)

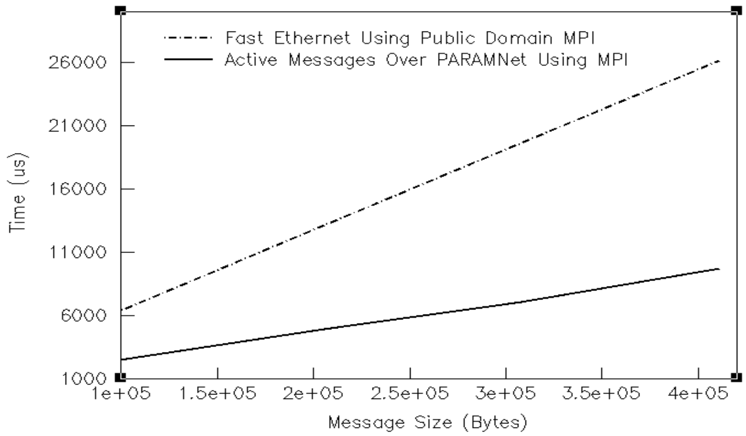
Performance of “**`Global Scatter`**” Communication Primitive



Communication Overheads: P-COMS

(Contd…)

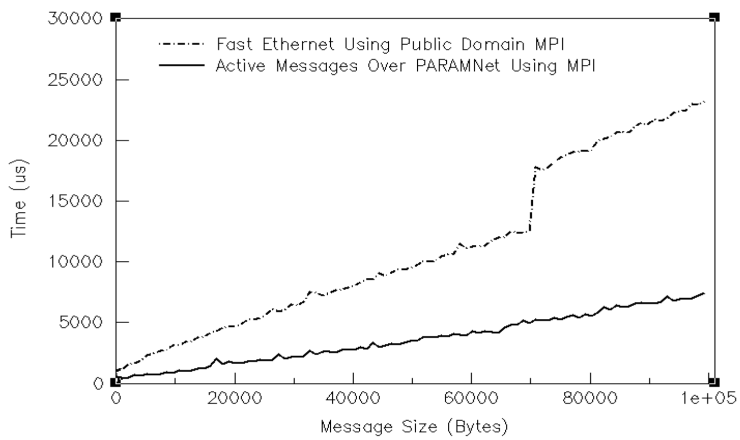
Performance of “**`Global Scatter`**” Communication Primitive



Communication Overheads: P-COMS

(Contd…)

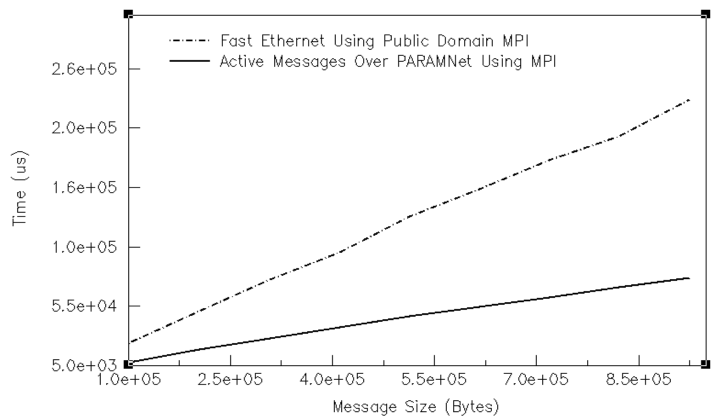
Performance of “**`Allreduce`**” Communication Primitive



Communication Overheads: P-COMS

(Contd…)

Performance of “**`Allreduce`**” Communication Primitive



Micro Benchmarks: ScaLAPACK

(Contd…)

Results of Selective ScaLAPACK routines

XSGBLU: Single Precision Banded LU factorization and Solve

Processors Used	**Matrixsize/ BlockSize**	**Mflop/s Achieved (*)**	
		FastEtherNet (TCP/IP) using `mpich`	AM over PARAMNet using `CMPI`
04 (S)	5800/120	1141.35	1334.97
08 (D)	7400/80	2166.93	2383.51
08 (D)	7400/100	2059.23	2275.14

* *Indicates that optimization is in progress and further improvement in performance can be achieved (**S** indicates single processor of one node is used and **D** indicates both processors have been used for computation).*

Micro Benchmarks: ScaLAPACK

(Contd…)

Results of Selective ScaLAPACK routines

XSLLT: Single Precision Cholesky factorization and Solve

Processors Used	Matrixsize/ BlockSize	Mflop/s Achieved (*)	
		FastEtherNet (TCP/IP) using mpich	AM over PARAMNet using CMPI
08 (D)	1240/30	665.77	1052.15
08 (D)	1240/20	626.27	1033.55

* *Indicates that optimization is in progress and further improvement in performance can be achieved (**D** indicates both processors have been used for computation).*

Micro Benchmarks: LINPACK

(Contd…)

Results of LINPACK

Processors Used	Matrixsize/ BlockSize	Mflop/s Achieved (*)	
		FastEtherNet (TCP/IP) using `mpich`	AM over PARAMNet using `CMPI`
08 (D)	8000/40	1899.74(*)	3100.68(*)
08 (D)	10000/50	1749.83(*)	3180.79(*)

* *Indicates that optimization is in progress and further improvement in performance can be achieved (**D** indicates both processors have been used for computation).*

LINPACK : hplbench on PARAM 10000

System configuration : 300 MHz (quad CPU) Solaris 2.6

- One Node 1 CPU - 190 Mflop/s
- One Node 2 CPU - 363 Mflop/s
- One Node 4 CPU - 645 Mflop/s

Processors Used (SS = single Switch)	Matrixsize/ BlockSize	Mflop/s Achieved (*)	
		FastEtherNet using mpich	HPCC (KSHIPRA)
8 (1,8) (SS)	4000/60		1219
16 (4,4) (SS)	6000/40		2100
16(8,2) (SS)	7808/64	1945	2347
24(8,3) (SS)	12000/80	2677	3256
32(8,4) (SS)	12000/60	3500	4484

(Contd…)

Micro Benchmarks: NAS

System Configuration : 400 MHz (Dual CPU) Solaris 2.6 – PARAM 10000 - A cluster of SMPs

Description of Routine and Problem Size		Mflop/s Achieved (*)	
		FastEtherNet (TCP/IP) using mpich	HPCC software PARAMNet - CMPI
		8 Processors	8 Processors
MG	A	350.50	584.32
	B	374.07	629.40
LU	A	648.81	775.11
	B	668.92	801.42
CG	B	108.40	178.54
FT	A	127.71	146.48

(*) Indicates that optimization is in progress and further improvement in performance can be achieved

CDAC The Supercomputing People

Third Party CFD Application

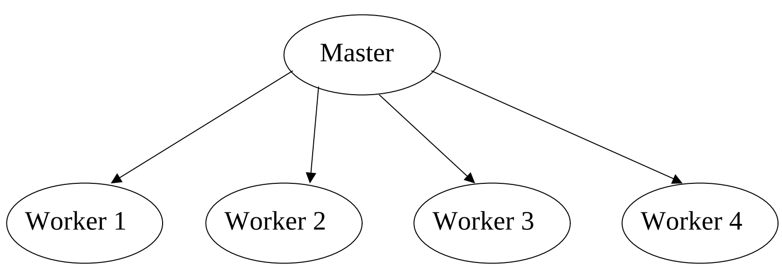
- **Description and objective :**

  Parallelization of GUES - a Computational Fluid Dynamics (CFD) Application on PARAM 10000 and extract performance in terms of Mflop/s.

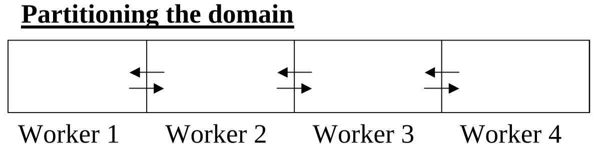
- **Rules :** Compiler options/Code restructuring/Optimization of the code is allowed for extracting performance of CFD code

- **Performance**, in terms of Mflop/s :
  - Performance of GUES serial program
  - Performance of parallel application with HPCC software (AM over PARAMNet using CDAC-MPI) and FastEtherNet (TCP/IP) using `mpich` (version 1.1)

Communication in CFD application



Partitioning the domain



Third Party CFD Application

(Contd…)

System Configuration : 400 MHz (Dual CPU) Solaris 2.6 – PARAM 10000 - A cluster of SMPs

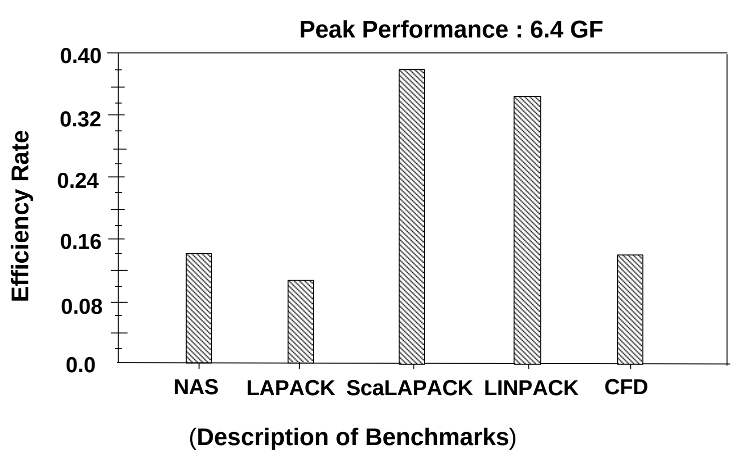
Single Processor Optimization : Used compiler optimizations; code restructuring techniques; Managing Memory overheads;

MPI : Packaging MPI library calls; Using proper MPI library Calls;

Grid Size	Processors	Mflop/s Achieved (*)	
		FastEtherNet (TCP/IP) using mpich	HPCC Software (PARAMNet – CMPI)
192 × 16 × 16	8 (D)	560 Mflop/s	1000 Mflop/s
384 × 16 × 16	8 (D)	600 Mflop/s	1020 Mflop/s

* *Indicates that optimization is in progress and further improvement in performance can be achieved. (**D** indicates two processors)*

Efficiency Rates for Different Benchmarks



Thank you