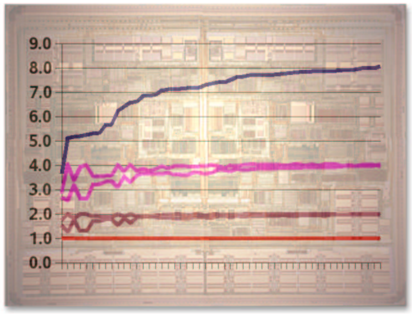
Throughput Computing - Why and How ?



Partha Tirumalai, Ruud van der Pas,
Scalable Systems Group, Sun Microsystems

High-Performance Computing on Sun - Today and Tomorrow
RWTH Aachen University, Germany
October 5-6, 2004

Objectives

- Why
  - *Explain Sun's Throughput Computing strategy*
- How
  - *UltraSPARC IV Architecture*
  - *Back up Sun's Throughput Computing strategy with data*

Outline

- ❑ *Sun's Throughput Computing Strategy*
- ❑ *UltraSPARC IV Architecture Overview*
- ❑ *The PEAS Test Suite*
  - *Testing Circumstances*
  - *Description Of The Test Suite*
  - *PEAS Single Processor Results*
  - *PEAS Throughput Results*
- ❑ *Conclusions*

Sun's Throughput Computing Strategy

The Big Bang Is Happening - 4 Converging Trends

Network Computing Is Thread Rich

Web services, Java™ applications, database transactions, ERP . . .

Attributes of Various Workloads

	Web Services			Client Server		Data Warehouse
Attribute	**Web** (Web99)	**App Serv** (JBB)	**Data** (TPC-C)	**SAP 2T**	**SAP 3T** (DB)	**DSS** (TPC-H)
Application Category	Web Server	Server Java	OLTP	ERP	ERP	DSS
Instruction-level Parallelism	Low	Low	Low	Medium	Low	High
Thread-level Parallelism	High	High	High	High	High	High
Instruction/Data Working Set	Large	Large	Large	Medium	Large	Large
Data Sharing	Low	Medium	High	Medium	High	Medium

The Big Bang Is Happening - 4 Converging Trends

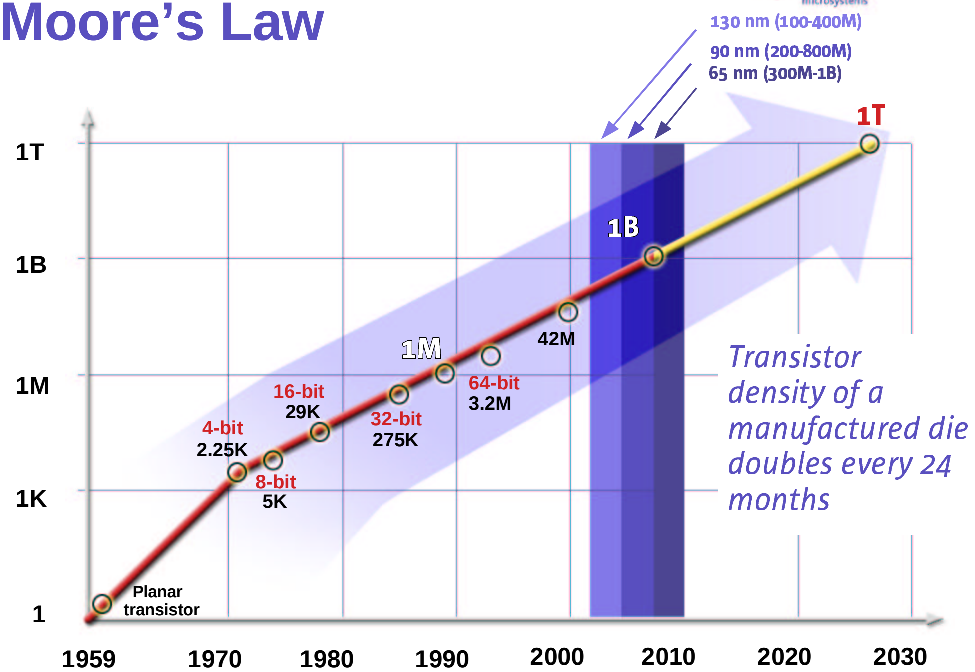
Network Computing Is Thread Rich

Web services, Java™ applications, database transactions, ERP . . .

Moore's Law

A fraction of the die can already build a good processor core; how am I going to use a billion+ transistors?

Moore's Law



The Big Bang Is Happening - 4 Converging Trends

Network Computing Is Thread Rich

Web services, Java™ applications, database transactions, ERP . . .

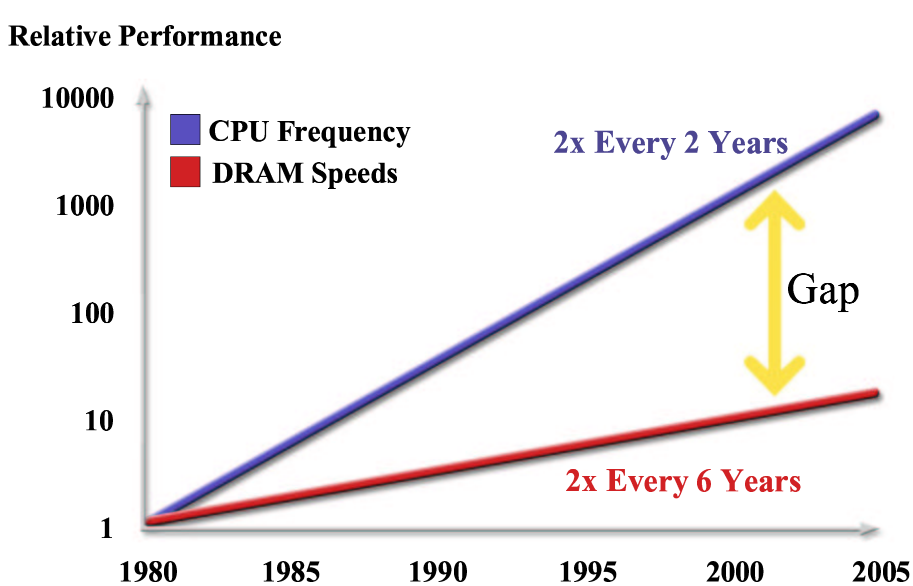
Moore's Law

A fraction of the die can already build a good processor core; how am I going to use a billion transistors?

Worsening Memory Latency

It's approaching 1000s of CPU cycles! Friend or foe?

Memory Bottleneck



The Big Bang Is Happening - 4 Converging Trends

Network Computing Is Thread Rich

Web services, Java™ applications, database transactions, ERP . . .

Moore's Law

A fraction of the die can already build a good processor core; how am I going to use a billion transistors?

Worsening Memory Latency

It's approaching 1000s of CPU cycles! Friend or foe?

Growing Complexity of Processor Design

Forcing a rethinking of processor architecture — modularity, less is more, time-to-market

The Big Bang Is Happening - 4 Converging Trends

Network Computing Is Thread Rich

Web services, Java™ applications, database transactions, ERP . . .

Moore's Law

A fraction of the die can already build a good processor core; how am I going to use a billion transistors?

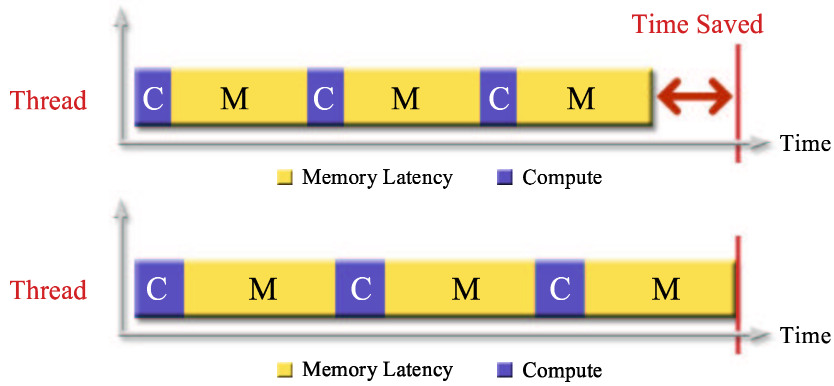
Worsening Memory Latency

It's approaching 1000s of CPU cycles! Friend or foe?

Growing Complexity of Processor Design

Forcing a rethinking of processor architecture – modularity, less is more, time-to-market

Typical Complex High Frequency Processor



Note: Up to 75% Cycles Waiting for Memory

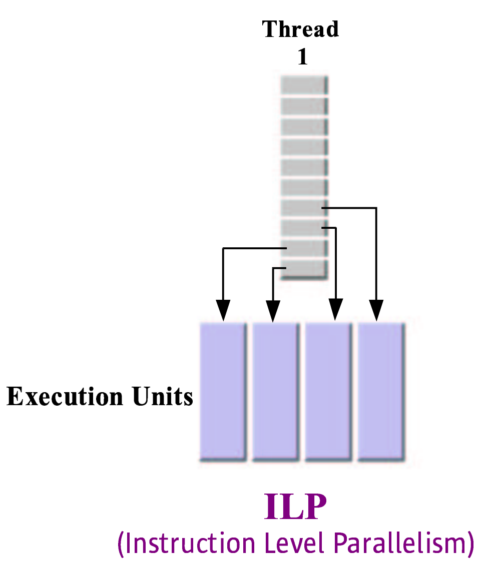
Design Complexity

- *Large caches*
- *Superscalar design to exploit ILP*
- *Out-of-order execution*
- *Very high clock rates*
- *Deep pipelines*
- *Speculative prefetches*
- *High power dissipation*
- *.....*

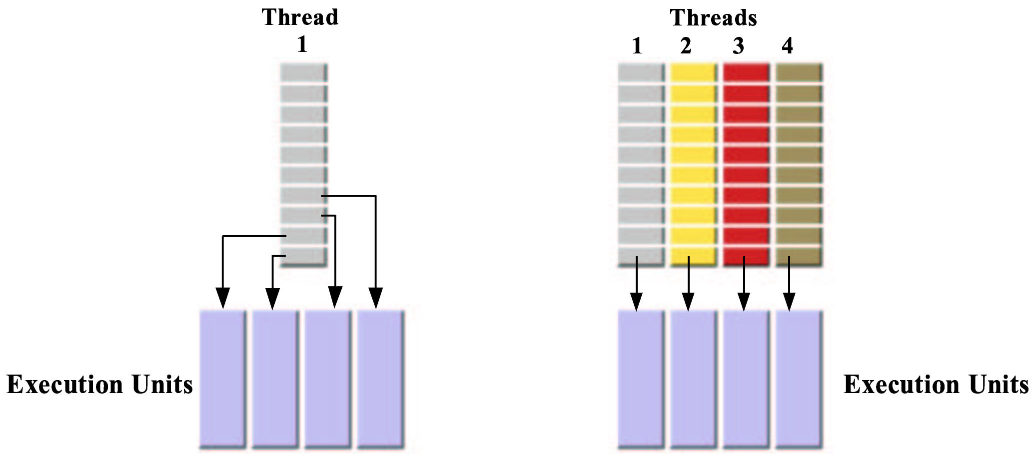
Implications If We Continue Like This

- Memory is the BIG bottleneck
  - Large working sets => high cache miss rates
  - Performance dominated by memory stall time
- MHz is misleading indicator of performance
  - No need to wait faster
  - Higher MHz is not power efficient
- Single thread processors deliver diminishing returns
  - Complexity and power consumption increase much faster than performance

ILP versus TLP



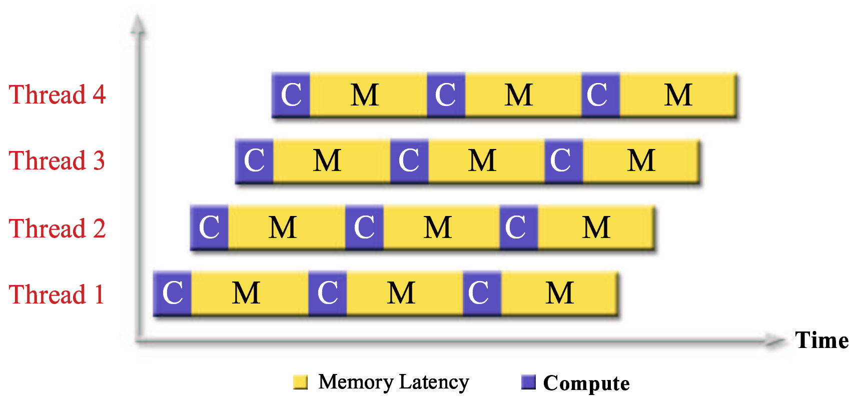
ILP versus TLP



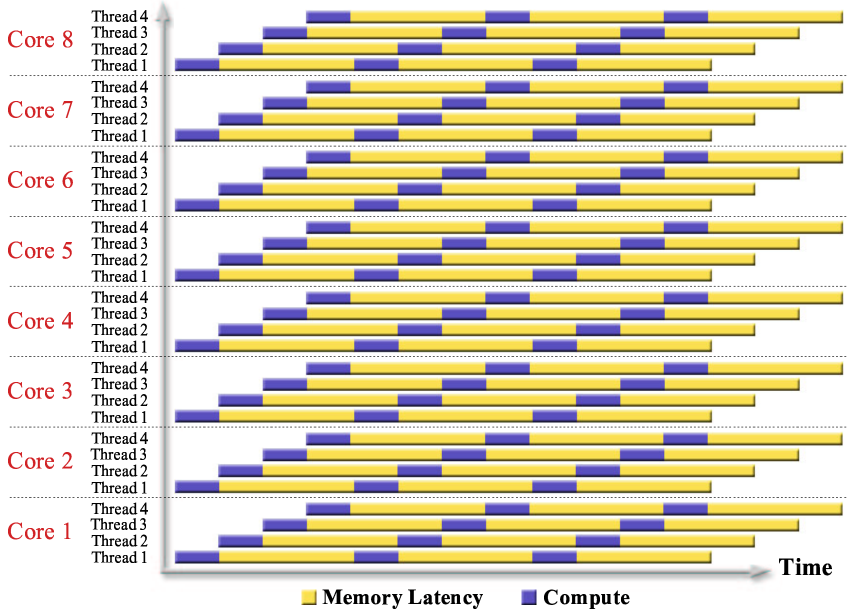
ILP
(Instruction Level Parallelism)

TLP
(Thread Level Parallelism)

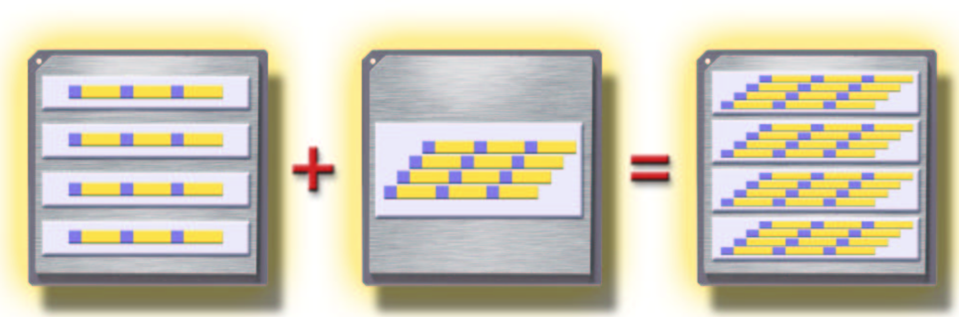
Chip Multithreading (CMT)



CMT – Multiple Multithreaded Cores



TLP (Thread Level Parallelism)



CMP
(chip multiprocessing)

SMT*
(simultaneous multithreading)

CMT
(chip multithreading)

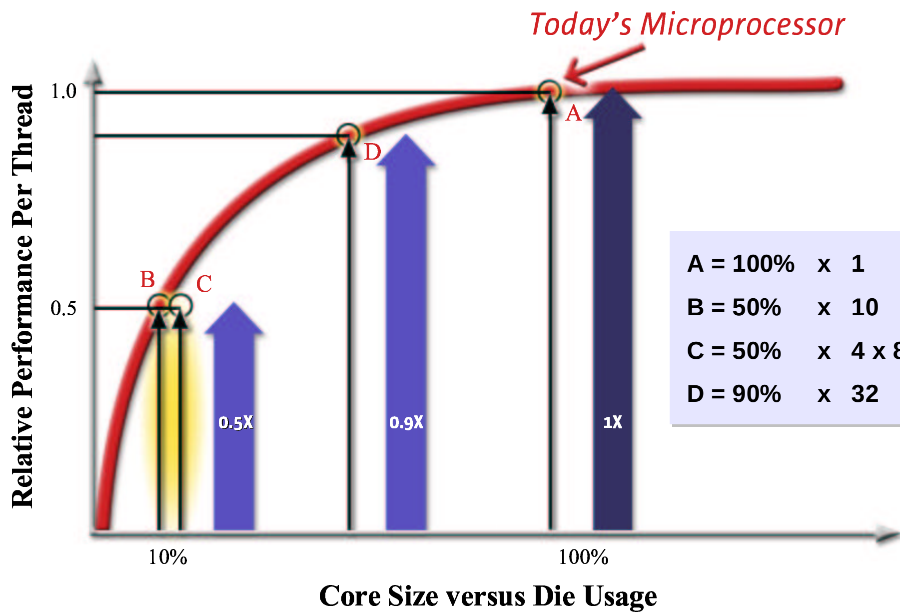
n m cores per processor

m n threads per core

n x m mxn threads per processor

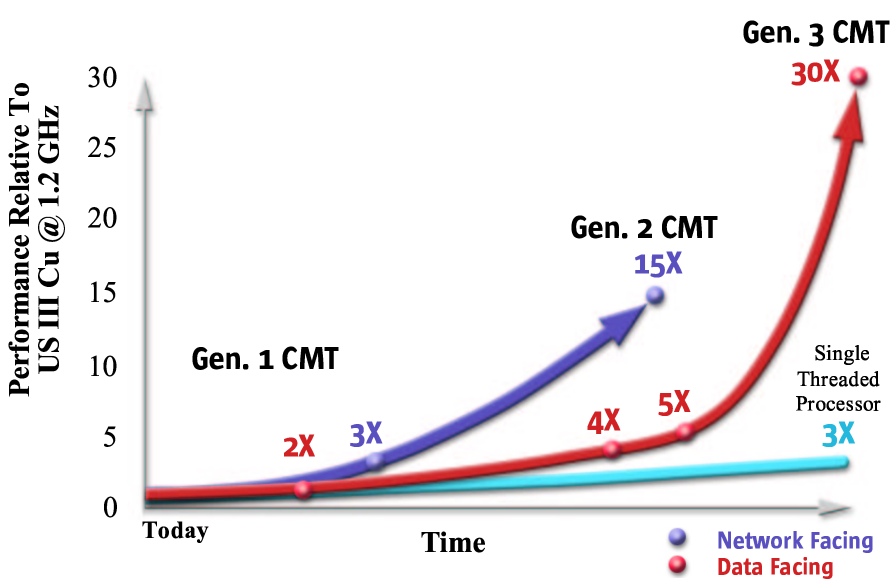
* Not all SMT is created alike

How Can CMT Deliver?



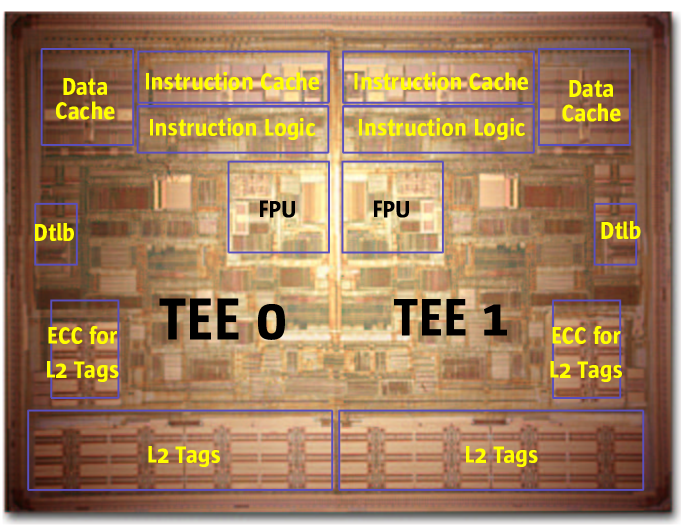
A = 100%	x	1	=	1x
B = 50%	x	10	=	5x
C = 50%	x	4 x 8	=	16x
D = 90%	x	32	=	29x

Throughput Computing



UltraSPARC IV Architecture Overview

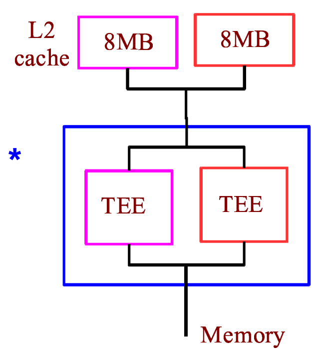
UltraSPARC IV



US IV - Primary Design Objectives

- *Improve single thread performance*
- *Double throughput performance*
- *Retain instruction set/binary compatibility*
  - *All old binaries run unmodified*
- *Operate within existing systems*
  - *Board upgrade; not a forklift upgrade*
  - *Co-habitate chassis with UltraSPARC III Cu*
- *Leverage UltraSPARC III Cu design*
  - *Lower risk, reduce time-to-market*

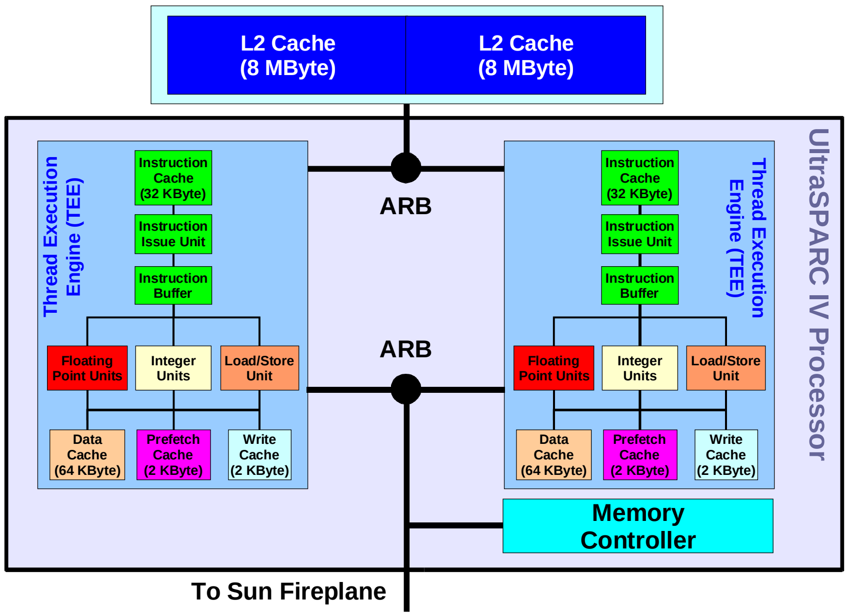
UltraSPARC IV - Key Features



- *64-bit SPARC V9 ISA + VIS2.0*
- *Dual-thread CMT design*
  - *UltraSPARC III Cu enhanced TEE's **
- *TEE's share the interface to L2*
  - *But the L2 cache is not shared*
- *TEE's share the interface to memory*
- *Higher compute density*
  - *Benefits workloads that are mostly latency bound and have parallelism that CMT can exploit*
- *Clock rate: 1.05GHz and 1.2GHz (initially).*

*) TEE = Thread Execution Engine.

UltraSPARC IV - Block Diagram



US IV - Performance Enhancements

- *Less sub-blocking in L2 cache (E$)*
  - *128B lines with 8MB E$, 64B lines with 4MB E$ (per TEE)*
  - *64k E$ tags (per TEE) (4X over US III Cu)*
- *2-way L2 cache has LRU replacement*
  - *UltraSPARC III Cu has random replacement*
- *Miscellaneous improvements*
  - *HW Prefetch and Second Load can be used simultaneously*
  - *Enhanced FPU: Fewer cases of unfinished fpop traps*
  - *Hash indexing Write Cache feature*
- *Support for higher CPU frequency*
  - *New Sun Fireplane clock ratios and L2 cache modes*

The PEAS Test Suite

Evaluating The CMT Design

- ***Indiscriminate resource sharing can impact the benefits of a CMT design***
- ***Compared to a corresponding UltraSPARC III Cu based system, each TEE in UltraSPARC IV:***
  - ***Shares the paths to the L2 cache and memory with the other TEE***
  - ***Sees a slight increase in latency to L2 and memory (necessitated by the need for arbitration)***

Q ***To what extent does contention for shared resources limit the benefits targeted by the UltraSPARC IV design?***

Systems Used

System	Processor	Speed (MHz)	TEEs	#Boards	Inter connect	Memory (GB)	Solaris
SF 6800	US III Cu	1200	8	2	Fireplane	32	112233-09
SF V440	US IIIi	1280	4	1	JBUS	16	112233-09
SF E6900 (lab system)	US IV	1200	8	1	Fireplane	32	s10_39 (pre-release)

Note: Performance is not only a function of the processor, the system plays an important role as well

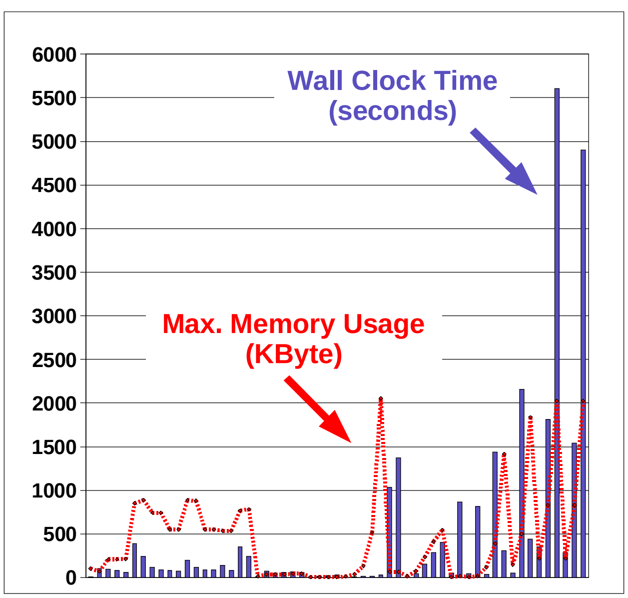
About PEAS

- *PEAS = Performance Evaluation Application Suite*
- *Consists of 18 user applications plus kernels derived from real applications*
  - *Chemistry, Physics and Mathematics*
  - *Fortran 77, Fortran 90 and C programs*
- *Applications are not aggressively tuned*
  - *Level of tuning differs per application*
  - *All applications compiled with -fast (S1S8)*
    - ✔ *For the C programs some more options were used*
    - ✔ *Prefetch level selected per application and version*
- *PEAS is meant to represent what our (advanced) users do*
  - *Or don't do ......*

Contents of PEAS

Name	Characteristic(s)	Versions	Datasets	Total
JOB1	3D FFT	2	1	2
JOB2	Matrix Multiply	3	1	3
JOB3	CFD kernel/FEM	2	1	2
JOB4	CFD kernel/FEM	2	1	2
JOB5	CFD kernel/FEM	2	1	2
JOB6	CFD kernel/FEM	2	1	2
JOB7	CFD kernel/FEM	2	1	2
JOB8	CFD kernel/FEM	2	1	2
JOB9	CFD kernel/FEM	2	1	2
JOB10	CFD kernel/FEM	1	1	1
JOB11	CFD kernel/FEM	3	1	3
JOB12	Chemistry	2	1	2
JOB13	CFD/Finite Difference	1	9	9
JOB14	Chemistry	2	1	2
JOB15	CFD/Multigrid	1	5	5
JOB16	Neural network	2	2	4
JOB17	CFD application	2	3	6
JOB18	Quantum Physics	2	3	6
			Total number of runs	57

PEAS Resource Requirements



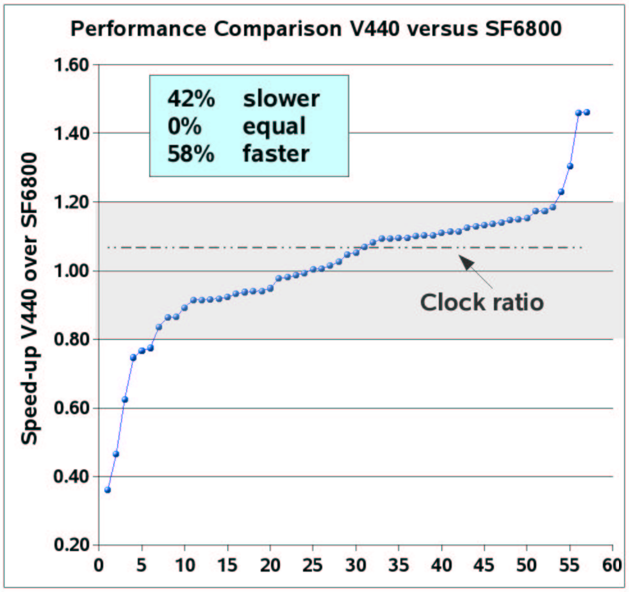
- *Some jobs run for a short time and take little memory*
- *Others run for a short while, but take quite some memory*
- *Other jobs run long, but don't need much memory*
- *Others run for a long time and need quite some memory*

Representing The Results

- *We are primarily interested in the overall picture*
- *To facilitate interpretation of results, we have sorted them*
- *In our comparison, the Sun Fire 6800 system has been used as the reference system*
  - *Therefore, the Sun Fire 6800 results were sorted*
  - *The other results have been adapted accordingly*
- *Please note that as a result of this, in the charts we loose the connection between the JobID and the application*
  - *In other words, the same JobID on two different graphs may represent a different application*

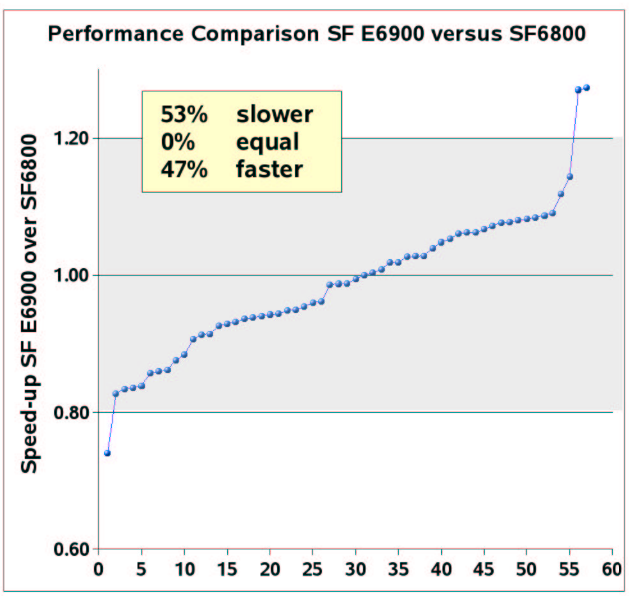
PEAS Single Processor Results

US IIIi @ 1280 versus US III Cu @ 1200



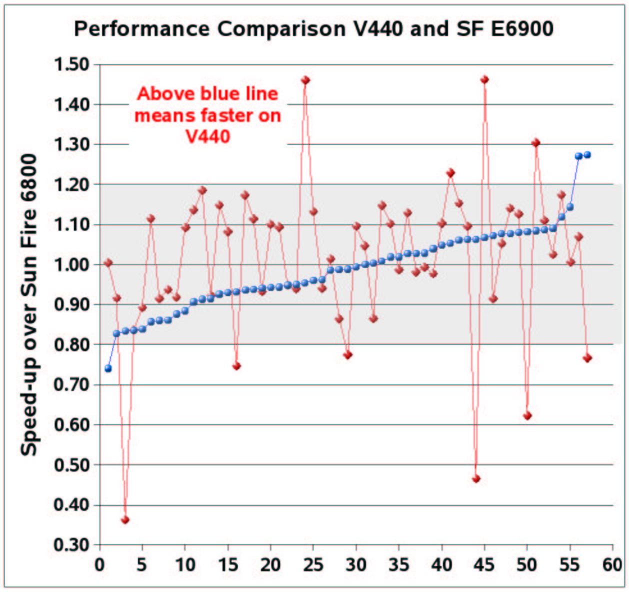
- *Results are sorted with respect to the Sun Fire 6800*
- *On 27 results (47%) the ratio of the clock cycles is exceeded*
- *A total of 47 results are in the [0.80 , 1.20] range*
- *In other words, in 82% of the cases, the two systems perform within +/- 20% of each other*

US IV @ 1200 versus US III Cu @ 1200



- *Results are sorted with respect to the Sun Fire 6800*
- *In 95% of the cases, the two systems perform within +/- 20% of each other*

Performance Comparison

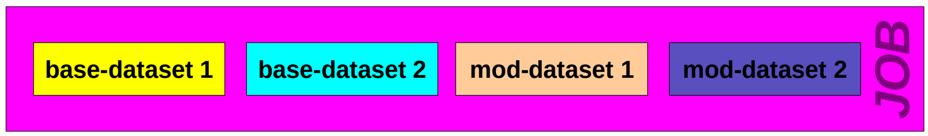


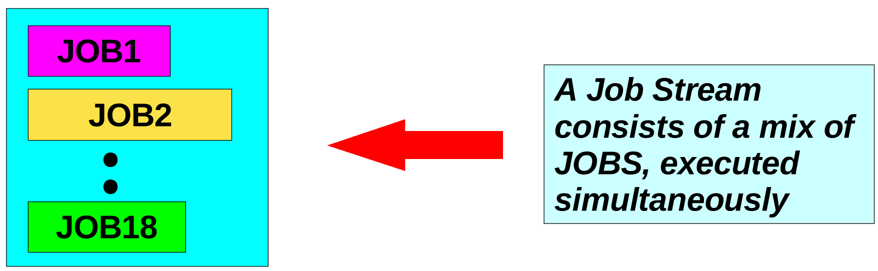
- *Results are sorted with respect to the Sun Fire E6900*
- *In 82% of the cases, the two systems perform within +/- 20% of each other*
- *But note the individual differences for the various runs*

PEAS Throughput Results

Definition of a Job Stream/1 *

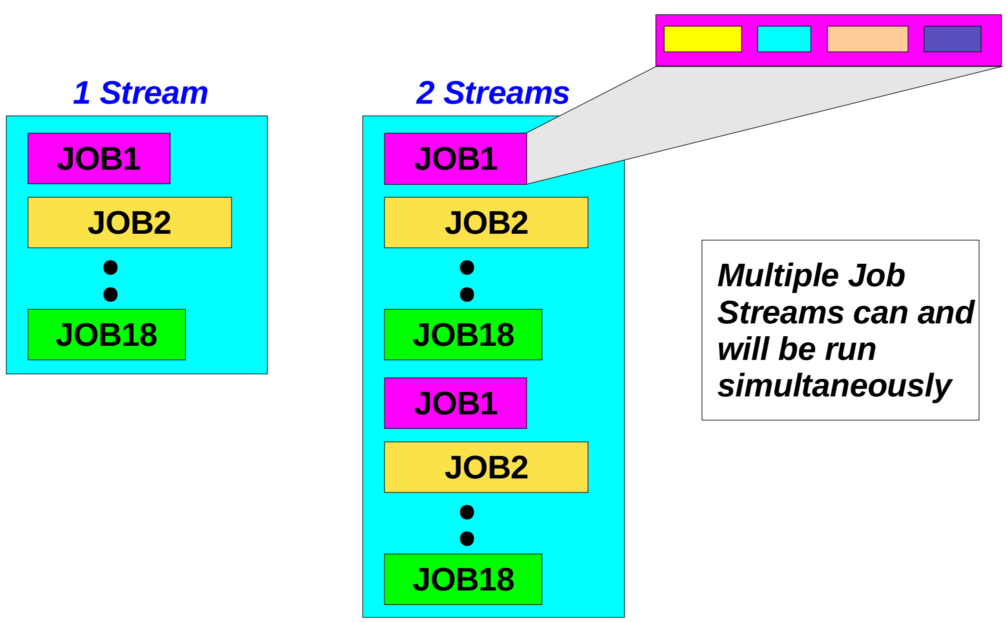
A JOB consists of a series of runs, sequentially running the version(s) of one specific application on the dataset(s):





****) A Stream in this context should not be confused with the STREAMS benchmark !***

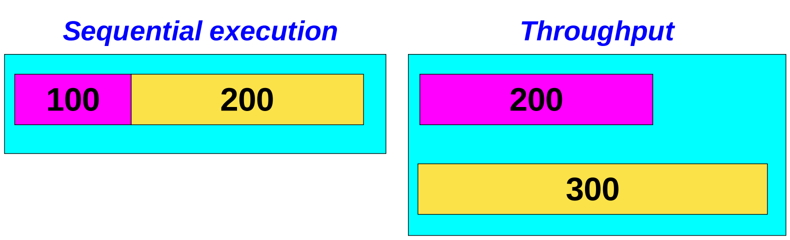
Definition of a Job Stream/2



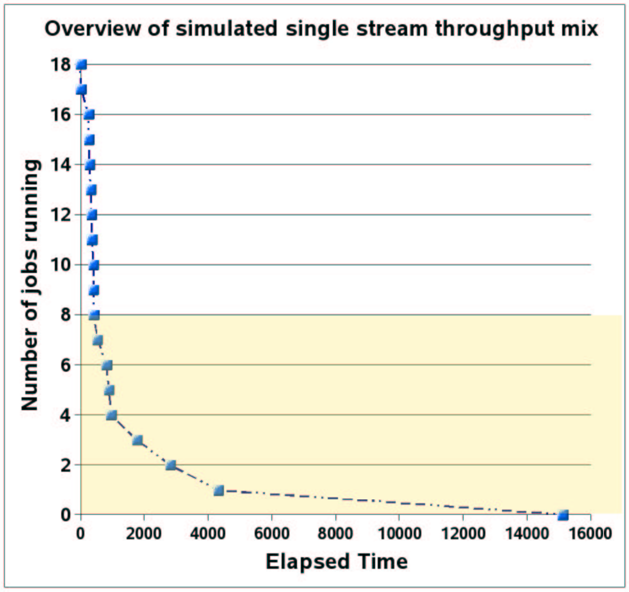
Comparison Of Measured Versus Simulated Results

Measured versus Simulated Results

- ***Using the standalone serial wallclock timing, one can simulate the behavior in a throughput environment***
  - ***This assumes ideal conditions and is therefore a simplification***
- ***Example (single processor system):***

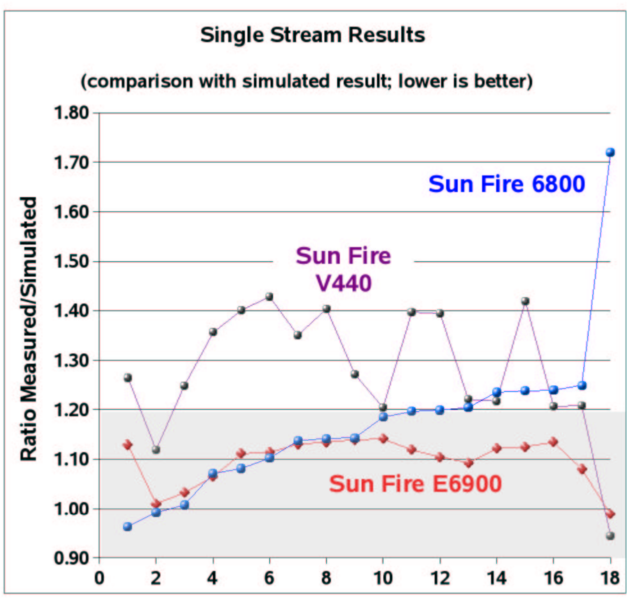


Example of a simulation



- *Simulation is for the Sun Fire 6800*
- *As can be seen, the run times vary significantly*
- *Most of the activity is in the early phase of the experiment*
- *Therefore, the total execution time is not a meaningful metric*
- *Instead, we will consider the behavior of individual jobs*

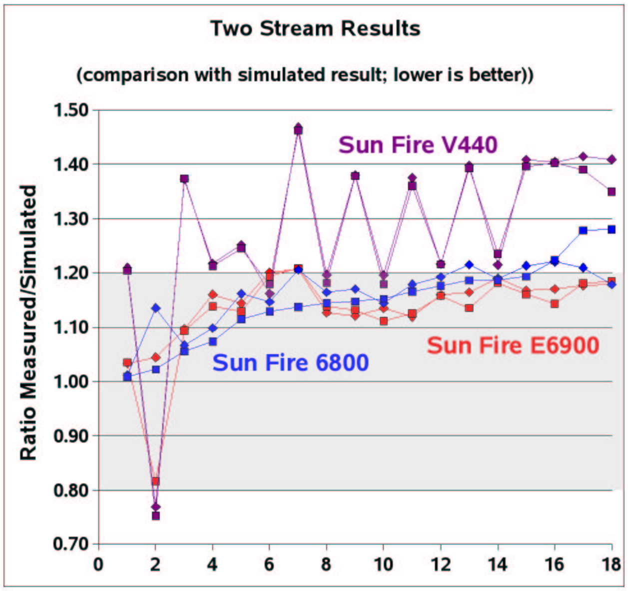
Measured/Simulated - One Stream (8GB)



Note: Results are sorted with respect to the Sun Fire 6800

- *The outlier with ID 18 is the first job that finishes; it runs for a very short time and is heavily impacted by the other jobs that are initiated*
- *The SF V440 has more difficulty handling the load*
- *On the SF E6900, 100% of the results are within +/- 20% of the simulated values*
- *This is 67% for the Sun Fire 6800*

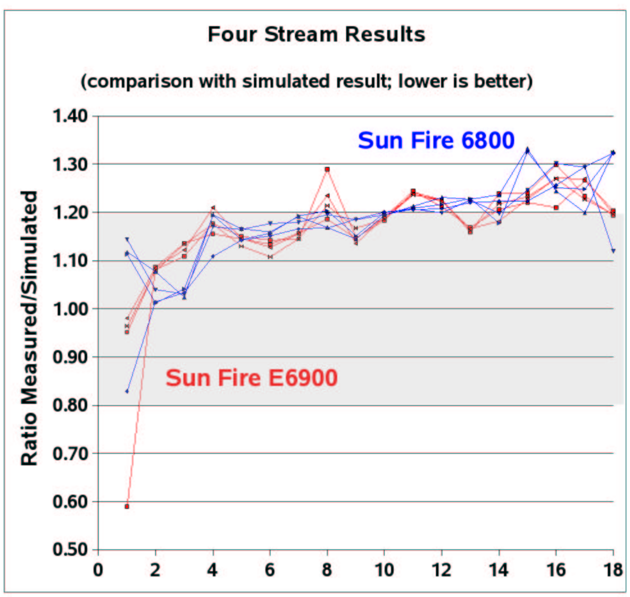
Measured/Simulated - Two Streams (16GB)



- *The outlier with ID 2 is the same first job that finishes very early*
- *The SF V440 has again more difficulty handling the load*
- *Note the similarity between the curves for one specific system*
- *~90% of the jobs on the SF E6900 are in the +/- 20% range of the simulated results*

Note: Results are sorted with respect to the Sun Fire 6800

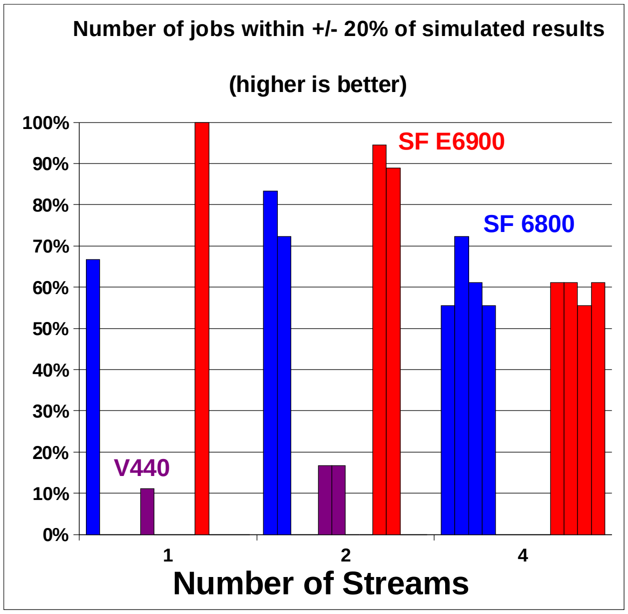
Measured/Simulated - Four Streams (32GB)



- *The outlier with ID 1 is the same first job that finishes very early*
- *Note the similarity between all the curves*
- *~60% of the jobs on the SF E6900 is in the +/- 20% range of the simulated results*

Note: Results are sorted with respect to the Sun Fire 6800

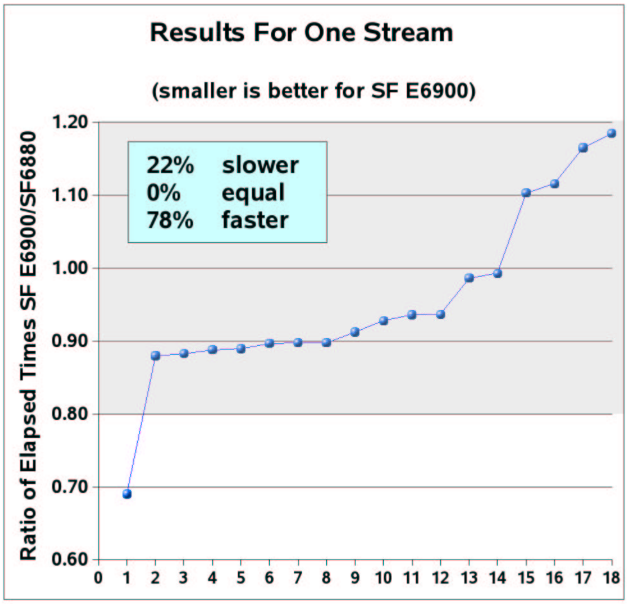
Summary Simulation Comparison



- *The higher the number, the more the system behaves in an "ideal" way*
- *Note that the SF E6900 is holding up very well compared to the Sun Fire 6800*

Sun Fire E6900 versus Sun Fire 6800

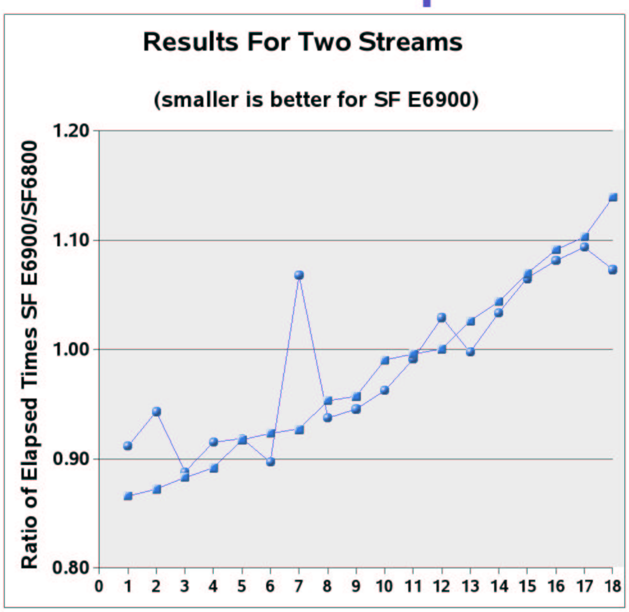
Single Stream Comparison



- *For 78% of the jobs, the SF E6900 is faster than the Sun Fire 6800*
- *A total of 94% jobs is within +/- 20% of the Sun Fire 6800*

Note: Results are sorted with respect to the Sun Fire 6800

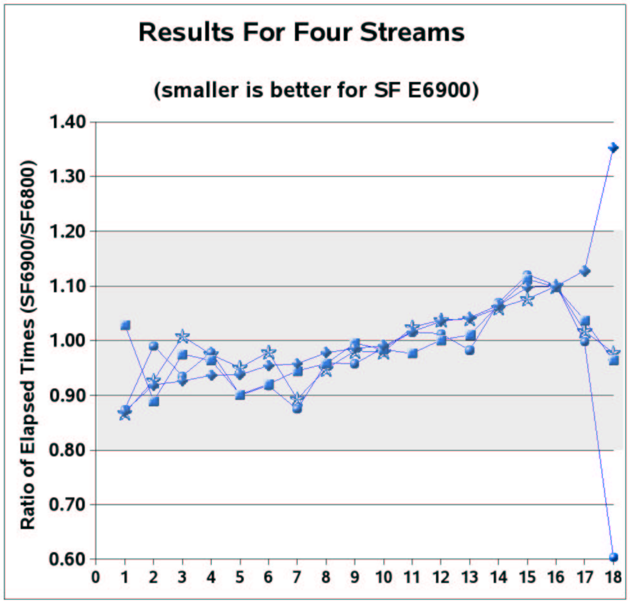
Two Stream Comparison



- *The job with ID 7 is again the first one that finishes and runs for a very short time*
- *For approx. 60% of the jobs, the SF E6900 is faster than the Sun Fire 6800*
- *For 100% of the jobs, the SF E6900 is within +/- 20% of the Sun Fire 6800*

Note: Results are sorted with respect to the Sun Fire 6800

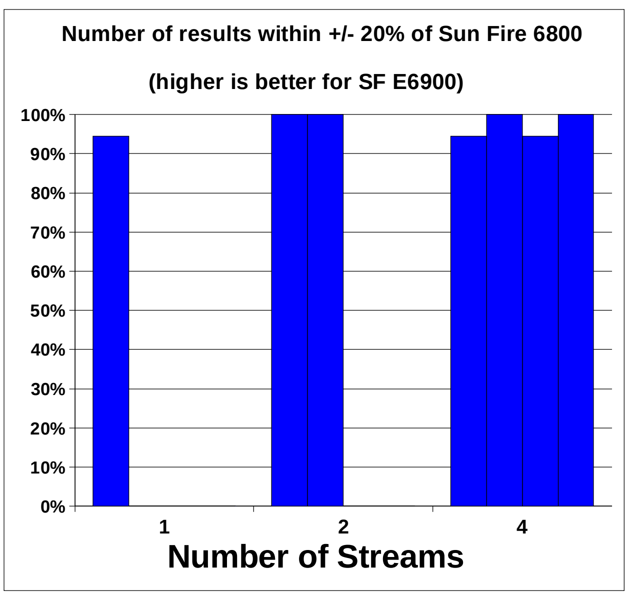
Four Stream Comparison



- The job with ID 18 is again the first one that finishes and runs for a very short time
- For approx. 50%-70% of the jobs, the SF E6900 is faster than the Sun Fire 6800
- For approx. 94%-100% of the jobs, the SF E6900 is within +/- 20% of the Sun Fire 6800

Note: Results are sorted with respect to the Sun Fire 6800

Summary Of Throughput Results



- For over 90% of the jobs tested, the SF E6900 system performs within +/- 20% of the Sun Fire 6800
- This is true for 1, 2 and 4 job streams

Conclusions

Conclusions

- *We believe that the SF E6900 system with 4 US IV processors running at 1200 MHz performs very well*
- *Parallelizing, tuning, and using the latest compiler and tools are key to getting the most from new systems*
- *PEAS - Compared to a SunFire 6800 system with 8 US III Cu processors running at 1200 MHz:*
  - *Even under a load of 4 streams, 50% of the jobs are faster on the SF E6900 system with 4 US IV processors*
  - *Even with 4 streams, over 90% of the jobs on the SF E6900 system perform within +/- 20% of the SF 6800 system*
- *Overall are our results an excellent demonstration of Sun's Throughput Computing strategy*

Thank You !