Chapter 1-3

Fundamentals of Computer Design

What we learned last class ?

- Trends in Technology
- Cost trends
- Dependability
- Performance metrics
- What program we chose to measure the performance ?

Topics in Chapter

1.1 Why take this course ?

1.2 Classes of computers in current computer market

1.3 Defining computer architecture and What's the task of computer design?

1.4 Trends in Technology

1.5 Trends in power in Integrated circuits

1.6 Trends in Cost

1.7 Dependability

1.8 Measuring, Reporting and summerizing Perf.

1.9 Quantitative Principles of computer Design

1.10 Putting it altogether

1.9 Quantitative Principles

- Take advantage of parallelism
- Principle of Locality
- Focus on the common case
- Amdahl's Law
- CPU Performance Equation

Take advantage of parallelism

- Most important methods of improving performance
- Parallelism levels
  - System level: use multiple processors
  - Instruction level:
    - Pipelining
  - Operation level:
    - set-associate cache
    - Pipelined function unit

Principle of Locality

- Program Property: Programs tend to reuse data and instructions they have used recently.
- Rule of thumb:
  - a program spends 90% of its execution time in only 10% of the code.
- Tempor
  - Recent accessed in the near future.
- Spatial locality
  - Items whose addresses are near one another tend to be referenced close together in time.

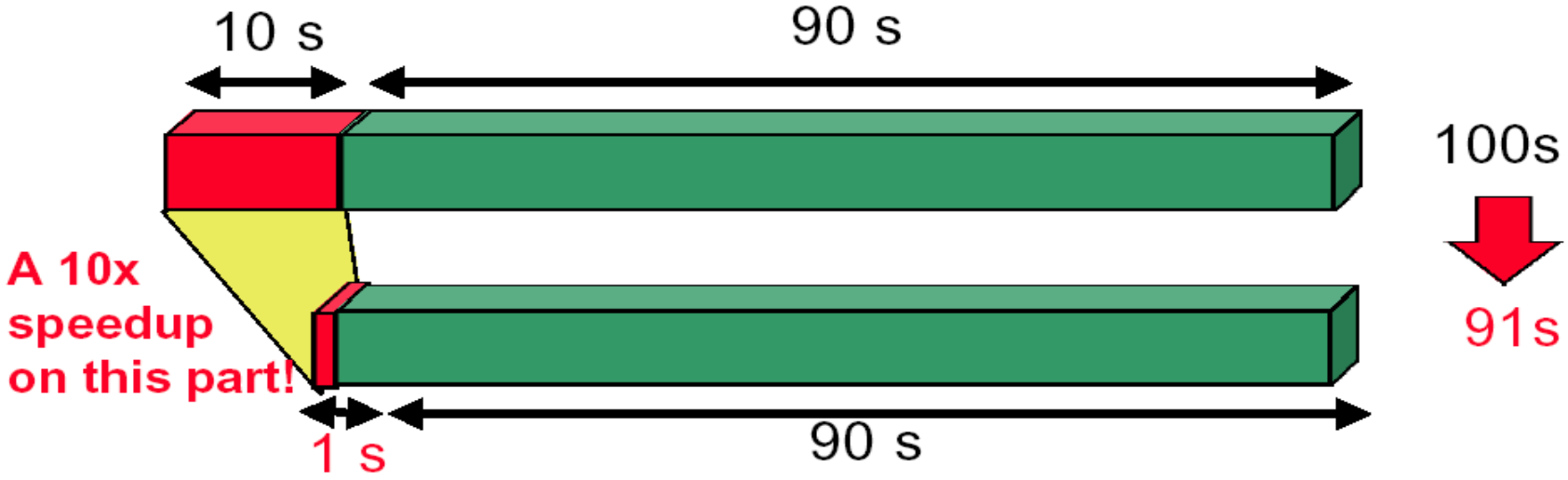
Any example ?

Focus on the common case

- The most important and pervasive principle of computer design.
  - Power, resource allocation, performance, dependability.
  - Rule of thumb: simple is fast.
  - Frequent case is often simpler and can be done faster.
- A fundamental law, called *Amdahl's Law*, can be used to quantify this principle.

Amdahl's Law

- The performance improvement to be gained from using some faster mode of execution is limited by the fraction of the time the faster mode can be used.
- Example



Amdahl's law

$$\text{Execution time after imrpovement} = \frac{\text{Execution time affected by the improvement}}{\text{Amount of improvement}} + \text{Execution time unaffected}$$

- Increasing the clock rate would not affect memory access time
- Using a floating point processing unit does not speed integer ALU operations

Amdahl's law

- Amdahl's law defines the speedup

$$\text{Speedup} = \frac{\text{Performance with enhancement}}{\text{Performance without enhancement}} = \frac{\text{Execution time w/o enhancement}}{\text{Execution time with enhancement}}$$

- If we know two factors:
  - **Fraction enhanced : Fraction of computation time in original machine that can be converted to take advantage of the enhancement.**
  - **Speedup enhanced in enhanced mode : Improvement gained by enhanced execution mode:**

$$\text{Exec time}_{new} = \text{Exec time}_{old} \times \left(\left(1 - Fraction_{enhanced}\right) + \frac{Fraction_{enhanced}}{Speedup_{enhanced}} \right)$$

Speedup Equation

$$Speedup_{overall} = \frac{ExecTime_{old}}{ExecTime_{new}} = \frac{1}{\left((1 - Fraction_{enhanced}) + \frac{Fraction_{enhanced}}{Speedup_{enhanced}}\right)}$$

- Example:
  - A server system with an enhanced CPU(10 times faster than the original one) used for Web serving. Assuming the original CPU is busy with computation 40% of the time and is waiting for I/O 60% of the time.
- Answer:
  - $Fraction_{enhanced} = 0.4$, $Speedup_{enhanced} = 10$
  - $Speedup = \frac{1}{0.6 + \frac{0.4}{10}} = \frac{1}{0.64} = 1.56$

Another Example

- Implementations of floating-point (FP) square root vary significantly in performance
- Two enhancement proposal
  - One proposal is to **enhance the FPSQR hardware** and speed up this operation by a factor of 10.
  - The alternative is just to try to make **all FP instructions** in the graphics processor run faster by a factor of 1.6;
- Assuming
  - FP square root (FPSQR) is responsible for **20%** of the execution time of a critical graphics benchmark.
  - FP instructions are responsible for a total of **50%** of the execution time for the application.
  - The design team believes that they do both enhancement with the same effort.
- Compare these two design alternatives.

Solution of the example

$$\text{Speedup}_{\text{PPSQR}} = \frac{1}{(1-0.2)+\frac{0.2}{10}} = \frac{1}{0.82} = 1.22$$

$$\text{Speedup}_{\text{PP}} = \frac{1}{(1-0.5)+\frac{0.5}{1.6}} = \frac{1}{0.8125} = 1.23$$

Example for dependability-1

- Given:
  - Failure rate$_{system}$ = 10*1/1000,000 + 1/500,000 + 1/200,000 + 1/100,000

$$= \frac{10 + 2 + 5 + 5 + 1}{1{,}000{,}000}$$

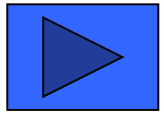
$$= 23000/1000{,}000{,}000$$

So, Power failure% = 5/23 = 0.22

Example for dependability-2

- Reliability of the power supply can be improved via redundancy:
  - 200000 → 830,000,000 ~ 4150X 5/23
- Reliability improvement

$$= \frac{1}{(1-0.22) + 0.22/4150} = 1.28$$



What the Amdahl's Law imply ?

- If an enhancement is only usable for a fraction of task, then the total speedup will be no more than 1/ (1-F).
- Serve the guide
  - to how much an enhancement will improve performance
  - to how to distribute resource to improve cost-performance
- Useful for comparing
  - the overall system performance of two alternatives,
  - two CPU design alternatives
- We can improve the performance by
  - increasing the $\text{Fraction}_{\text{enhanced}}$
  - or, increasing the $\text{Speedup}_{\text{enhanced}}$

The CPU Performance Equation

- The "Iron Law" of processor performance:
  - Often it is difficult to measure the improvement in time using a new enhancement directly.
- CPU Performance Equation

$$\text{CPU time} = \text{CPU clock cycles for a program} \times \text{Clock cycle time}$$

$$\text{CPU time} = \frac{\text{CPU clock cycles for a program}}{\text{Clock rate}}$$

Calculation of CPU Time

$$\text{CPU time} = \text{Instruction count} \times \text{CPI} \times \text{Clock cycle time}$$

Or

$$\text{CPU time} = \frac{\text{Instruction count} \times \text{CPI}}{\text{Clock rate}}$$

$$\text{CPU time} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock cycle}}$$

Architecture --> Implementation --> Realization

Compiler Designer Processor Designer Chip Designer

Component of performance	Units of measure
CPU execution time for a program	Seconds for the program
Instruction count	Instructions executed for the program
Clock cycles per instructions (CPI)	Average number of clock cycles/instruction
Clock cycle time	Seconds per clock cycle

Related technologies

- CPU performance is dependent upon 3 characteristics:
  - clock cycle (or rate) (CCT)
  - clock cycles per instruction (CPI)
  - instruction count. (IC)

	Inst Count	CPI	Clock Rate
Program	X		
Compiler	X	(X)	
Inst. Set.	X	X	
Organization		X	X
Technology			X

- One difficulty: It is difficult to change one in isolation of the others.

Other format of CPU Performance Equation

$$\text{CPU clock cycles} = \sum_{i=1}^{n} \text{IC}_i \times \text{CPI}_i$$

$$\text{CPU time} = \left(\sum_{i=1}^{n} \text{IC}_i \times \text{CPI}_i \right) \times \text{Clock cycle time}$$

$$\text{CPI} = \frac{\sum_{i=1}^{n} \text{IC}_i \times \text{CPI}_i}{\text{Instruction count}} = \sum_{i=1}^{n} \frac{\text{IC}_i}{\text{Instruction count}} \times \text{CPI}_i$$

Example of CPUtime calculation

- Suppose we have made the following measurements:
  - **Frequency of FP operations (other than FPSQR) = 25%**
  - **Average CPI of FP operations = 4.0**
  - **Average CPI of other instructions = 1.33**
  - **Frequency of FPSQR = 2%**
  - **CPI of FPSQR = 20**
- Two design alternatives
  - **decrease the CPI of FPSQR to 2**
  - **decrease the average CPI of all FP operations to 2.5.**
- Compare these two design alternatives using the CPU performance equation.

Answer to the question

$$\text{CPI}_{\text{original}} = \sum_{i=1}^{n} \text{CPI}_i \times \left(\frac{\text{IC}_i}{\text{Instruction count}} \right)$$

$$= (4 \times 25\%) + (1.33 \times 75\%) = 2.0$$

$$\text{CPI}_{\text{with new FPSQR}} = \text{CPI}_{\text{original}} - 2\% \times (\text{CPI}_{\text{old FPSQR}} - \text{CPI}_{\text{of new FPSQR only}})$$

$$= 2.0 - 2\% \times (20 - 2) = 1.64$$

$$\text{CPI}_{\text{new FP}} = (75\% \times 1.33) + (25\% \times 2.5) = 1.625$$

- Since the CPI of the overall FP enhancement is slightly lower, its performance will be marginally better.

Compare the result with that from Amdahl's law

- This is the same speedup we obtained using Amdahl's Law:

$$\text{Speedup}_{\text{new FP}} = \frac{\text{CPU time}_{\text{original}}}{\text{CPU time}_{\text{new FP}}} = \frac{\text{IC} \times \text{Clock cycle} \times \text{CPI}_{\text{original}}}{\text{IC} \times \text{Clock cycle} \times \text{CPI}_{\text{new FP}}}$$

$$= \frac{\text{CPI}_{\text{original}}}{\text{CPI}_{\text{new FP}}} = \frac{2.00}{1.625} = 1.23$$

- Suppose that there are four types of operations in an application. After making enhancements to the original function units, each type of operation gains performance improvement as shown in the following table.

Operation Type	IC (total 100)	CPI before enhancement	CPI after enhancement
Op1	10	2	1
Op2	30	20	15
Op3	35	10	3
Op4	25	4	1

1) What's the enhanced speedup of each operation respectively after improving?

2) What's the overall speedup of the application respectively after only improving Op2 or Op4?

   - Hint: Calculate the overall speedup considering only one of the enhanced function units is used. There are 2 situations.

Performance & price-performance

- Factors that responsible for the wide variation in price
  - Different levels of expandability
  - Use of cheaper disks and cheaper memory
  - Cost of CPU varies
  - Software differences
  - Lower-end system use PC commodity parts in fans, power supply, support chip sets
  - Commoditization effect

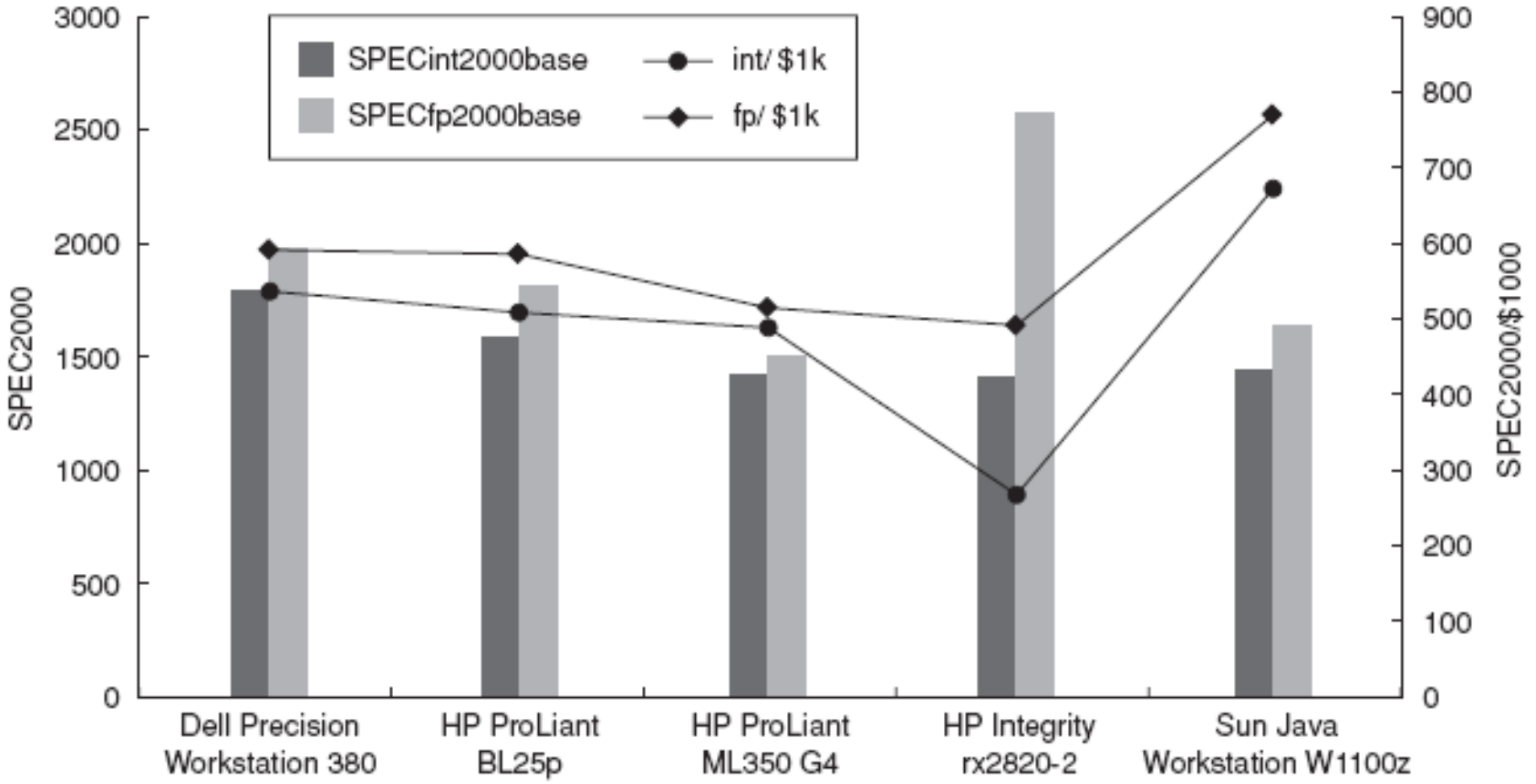
Five desktop and rack-mountable systems

Vendor/model	Processor	Clock rate	L2 cache	Type	Price
Dell Precision Workstation 380	Intel Pentium 4 Xeon	3.8 GHz	2 MB	Desk	$3346
HP ProLiant BL25p	AMD Opteron 252	2.6 GHz	1 MB	Rack	$3099
HP ProLiant ML350 G4	Intel Pentium 4 Xeon	3.4 GHz	1 MB	Desk	$2907
HP Integrity rx2620-2	Itanium 2	1.6 GHz	3 MB	Rack	$5201
Sun Java Workstation W1100z	AMD Opteron 150	2.4 GHz	1 MB	Desk	$2145

- 1GB ECC SDRAM，80GB disk

- Expandability(可扩展性):
  Sun Java worktation < Dell < HP BL25p
- Cost of processor: die size, L2 cache, clock rate, cache size
- Softerware difference

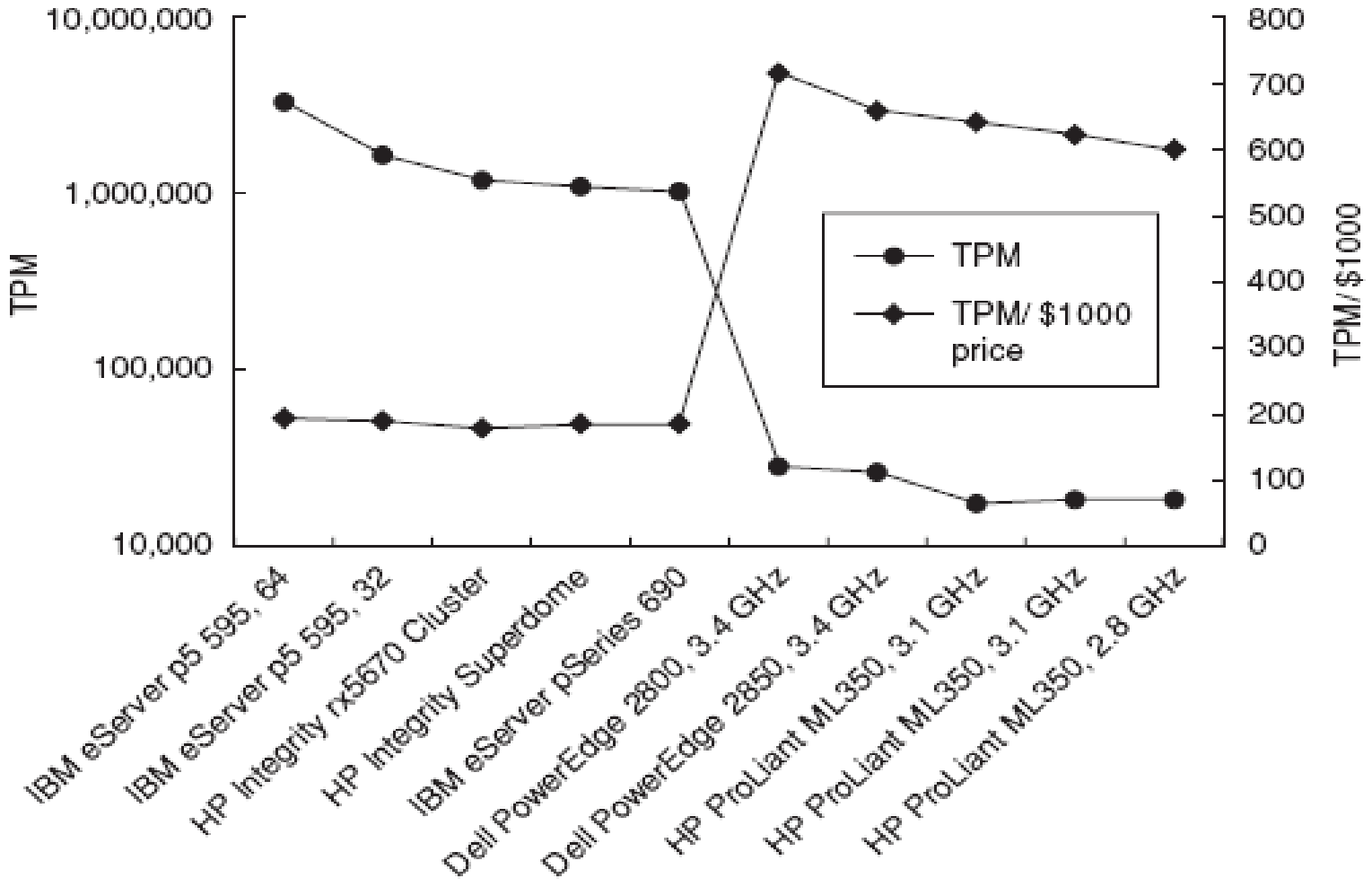
Price-performance www.spec.org



Messurements-1

- For Servers Fig 1.17, 1.18
  - TPC-C : standard industry benchmark for OLTP
    - Reasonable approximation
    - Measure total system performance
    - Rules of measurement are very complete
    - Vendors devote significant effort
    - Report both performance & price-performance

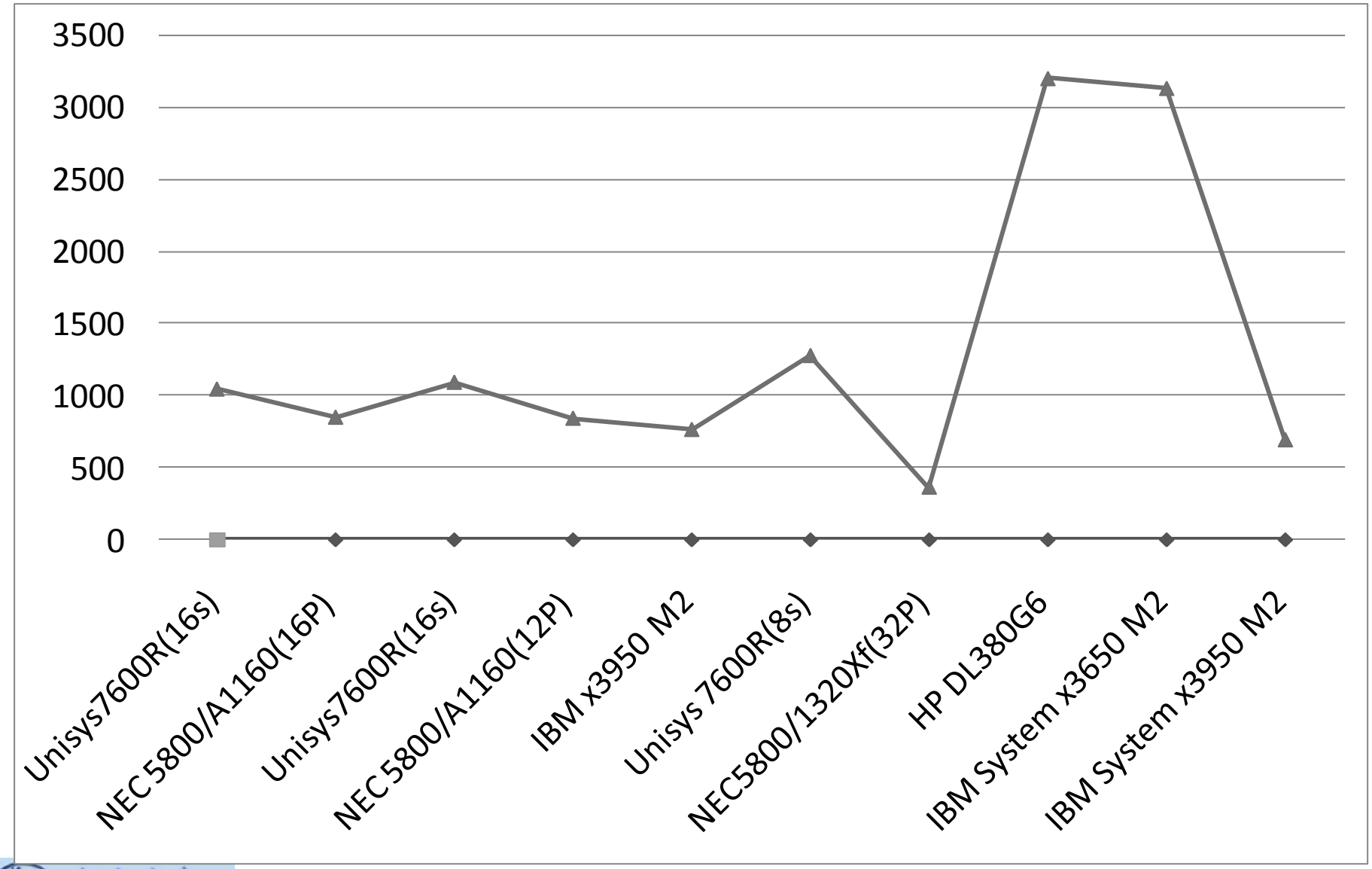
Price-performance TPC-C 2005.7



TPC-C & TPC-E

- TPC(TransactionProcessing PerformanceCouncil)
  - 10 members http://www.tpc.org
- TPC-C:
  - benchmark for OnLine Transaction Processing
  - simulates a complete environment where a population of terminal operators executes transactions against a database
- TPC-E
  - simulates the OLTP workload of a brokerage firm
- TPC-C&TPC-E
  - http://www.searchsv.com.cn/showcontent_13524.htm

tpsE & tpsE/&100000



Fallacies & pitfalls

- Pitfall(易犯的错误):
  - Falling prey to Amdahl's Law.
    - Not to try to improve some unit of small F
  - A single point of failure
    - An example of dependability
    - Don't forget the single fan !

Fallacy1 : the cost of the processor dominates the cost of the system

	Processor + cabinetry	Memory	Storage	Software
IBM eServer p5 595	28%	16%	51%	6%
IBM eServer p5 595	13%	31%	52%	4%
HP Integrity rx5670 Cluster	11%	22%	35%	33%
HP Integrity Superdome	33%	32%	15%	20%
IBM eServer pSeries 690	21%	24%	48%	7%
Median of high-performance computers	21%	24%	48%	7%
Dell PowerEdge 2800	6%	3%	80%	11%
Dell PowerEdge 2850	7%	3%	76%	14%
HP ProLiant ML350	5%	4%	70%	21%
HP ProLiant ML350	9%	8%	65%	19%
HP ProLiant ML350	8%	6%	65%	21%
Median of price-performance computers	7%	4%	70%	19%

Fallacy

- Benchmarks remain valid indefinitely
  - "Benchmarksmanship" "benchmark engineering"

- The rated mean time to failure of the disks is 1200000 hours or almost 140 years, so disks practically never fail.
  - Real-world MTTF is about 2-4 times worse than manufacturer's MTTF for ATA disks, 4-8 times worse for SCSI disks

    (ATA: AT Attachment)

    SCSI: Small Computer System Interface)

- Peak performance tracks observed performance.