M.Tech
CIT/ME

Course name : Advanced Microprocessor – Elective

Topic – 1

* computer system performance and its evaluation : A brief account
* CPU performance equation and differentiation of CIS & RISC architectures

Performance summarization :

Example 1

Suppose we have two programs or process – P1 and P2 and three computors : A, B, C

Suppose A is 10 tis faster than B for P1
B is 10 tis faster than A for P2
A is 20 tis faster than C for P1
C is 50 tis faster the A for P2
C is 5 tis faster the B for P2 and so on

From here : Performance of a Program on a m/c depends on type of Progrm (i.e Application)

: Taking individually, one statement may be of some use ; but relative performance is unclear wrsto m/cs.

Example 2

	Comp A	Comp B	Comp C
P1 (secs)	1	10	20
P2 (secs)	1000	100	20
Total tie (secs)	1001	110	40

(Execution tis of 2 Programs in m/cs)

Summary of tis table is as!

B is $\frac{1001}{110}$ ts faster than A for P1 and P2.

C is $\frac{110}{40}$ ts faster than B for P1 and P2.

C is $\frac{1001}{40}$ ts faster than A for P1 ed P2.

The above table can give relative performance of machine if P1 and P2 run equal no. of ts.

In that case C is faster for P1 and P2

Suppose: Time taken for P1 on m/c A = T_1

" " P2 " " B = T_2

" " P1 " " C = T_3

Av. Execution te = $\frac{T_1 + T_2 + T_3}{3}$;

Similarly Av. execution te of P2 can be found

Thus the av. execution te that tracks total execution te is the arithmetic mean:

$\frac{1}{n} \sum_{i=1}^{n} Time_i$ ($Time_i$ is execution te of i^{th} Program of total of n work load)

Weighted execution te: As shown above, for equal mix ie if programs run equal no. of ts execution te is governed by arithmetic mean.

When there is unequal mix of programs in the work load, then we assign weightage (w_i) to each program to indicate relative frequency of program in work load

eg. If 20% of tasks in work load are P1 $\Rightarrow$ weightage = 0.2

80% of tasks in work load are P2 $\Rightarrow$ weightage = 0.8

Weightage Arithmetic mean $\sum weightage_i \times Time_i$ (Weightage is freq. of i^{th} Prog in work load.

$Time_i$: is Execution te of i^{th} Prog

Conclusion of above examples:

"Performance of a computer is application specific"

∴ ~~For a part~~ This leads us to an important design principle for performance enhancement

"MAKE COMMON CASE FAST"

In general purpose computers used for all types of applications; having invariably conflicting requirements attainment of this design principle is not possible.

~~Therefore~~ This has led us to design ~~computers~~ different computers for different applications:

e.g. For I/o applications

For caputational probs

and ~~&~~ Embedded ~~processors for~~ Systems/ ~~for~~ V. specific applications.

Amdhal's law

The law states, "The performance improvement to be gained by using some faster mode of execution is limited by fraction of tie the faster mode can be used."

The law defines speed up that can be gained by using a particular feature.

Suppose enhancement has been made

$$\text{Speed up} \triangleq \frac{\text{Performance of entire task using enhancement when possible}}{\text{Performance of entire task without using enhancement}}$$

$$= \frac{\text{Execution time for entire task without enhancement}}{\text{Execution tie of entire task using enhancement when possible}}$$

Speed up tells us how much faster ~~m/c~~ task will run using the machine as opposed to orginal m/c.

Execution enhancement depends on two factors:

1. The Fraction of computation time in orginal machine that can be converted to take advantage of enhancement

   e.g. If unutalized time of m/c is 60 secs & Suppose '10' secs of m/c are utalized ~~out~~ ~~of~~ out of 60 secs. by virtue of enhancent

   ∴ Fraction enhanced = 10/60

2. Speed up Enhanced:

   This is improvement gained by enhanced execution mode, i.e, how much faster the task would run if the enhanced mode was used for a program. This may be given as: $\frac{\text{Time of orginal mode (Ex. time)}}{\text{Time of Enhanced mode (Ex. tie)}}$

eg. Suppose cg in its orginal mode 5 secs are required to run a progran. If the m/c is enhanced and in enhanced mode the progn runs only in 2 secs

Then improvent (Enhancement) = $5/2 > 1$

Exople: Suppose using enhancement processor is 10 tis faste the orginal processor. Assume orginal processor is busy with capitalis 40% tie and waiting for I/O 60% tie.

What is overall Speed up gained by incorporating enhancement.

Some important definitions.

1. Execution time: This can be defined in different ways depending on what we count. Most straight forward definition is: Wall clk time, Response time or Elapsed time. This time is latency to complete a task.
Elapsed time includes:
   a) Disk access
   b) Memory access
   c) I/O activities
   d) Operating system overheads etc.
Thus elapsed time is time is time as seen by the user.

2. CPU time: This is time during which CPU is executing. This time does not include waiting time etc. (as may be case in elapsed time).

CPU Time
- CPU time spent on program (user CPU time)
- CPU time spent on O.S.

Example: If we give a command in Unix: Time
Time: 90.7 u 12.9 s 2:39 65%.
- 90.7 u → User CPU time
- 12.9 s → System CPU time
- 2:39 → Elapsed time
- 65% → % of CPU time actually used (excluding waiting time)

So more than $\frac{1}{3}$rd of time was spent on waiting for I/O or running other programs (in multiprogramming envi.) or both.
Many times we ignore CPU time spent on O.S. because of inaccuracies in O.S.
∴ System Performance = Elapsed time on unloaded system
CPU Performance = User CPU time on an unloaded system.

CPU Performance Equation:

Computers are constructed using a clk running at a constant rate. These discrete time events are called clock periods, clk cycles etc.

Clk Period is designated by

a: Duration (n.s.)
b: Rate (GHz) or

CPU Time = CPU clk cycles per program × clk cycle time

$$= \frac{\text{CPU clk cycles for prog}}{\text{clk rate}}$$

In addition to ~~clk cycles~~ number of clk cycles needed to execute a program, we can also count the no. of instructions executed; This is also called Path length or Instruction Count (IC)

If we know no. of clk cycles & IC we can calculate No. of clk cycles per Instruction (CPI). Designers sometimes use Instructions per clk cycle (IPC) which is inverse of CPI

$$CPI = \frac{\text{CPU clk cycles for a program}}{IC}$$

↳ Figure of merit, provides insight into different styles of Instruction sets & implementations.

Total clk cycles = IC × Av. no. of clk cycles per instruction

CPU te = Instruction Count × clk cycle te × cycles per inst.

$$CPU\ te = \frac{\text{Instruction Count} \times \text{clk cycle te}}{\text{clk rate (IPC)}} \quad \text{cycles/inst.}$$

Expanding the formula in the units of measurement

$$\left(\frac{\text{Inst}}{\text{Progm}}\right)_{(Ic)} \times \left(\frac{\text{clk cycles}}{\text{Inst}}\right)_{(CPI)} \times \left(\frac{\text{seconds}}{\text{clk cycle}}\right)_{\left(\frac{1}{\text{clk rate}}\right)} = \frac{\text{Seconds}}{\text{Progm}} = CPU\ te$$

or ~~CPU te = $\frac{\text{time}}{\text{Progm}}$~~ = $\frac{\text{time}}{\text{cycle}}$

or $CP\ Time = \frac{\text{Time}}{\text{Progm}} = (I_c)(CPI)\left(\frac{1}{\text{clk rate}}\right)$ cycles/sec.

$\frac{\text{Inst}}{\text{Progm}} \times \frac{\text{clk sec}}{\text{cycles}}$

⇓ Reduce for CISC ⇓ Reduce for RISC.

CPU te or CPU performance is dependent on

a) clk cycles or clk rate (Processor Specific)
b) Clk cycles per inst
c) Instruction Count

A x% improvement in any one of them leads to x% improvement in CPU te.

It is difficult to change one of the parameters in totality because basic technology is involved in changing each characteristic are independent

; clk cycle te : HW technology & organization
CPI : organization inst. set architecture
IC : Inst set architecture & compiler tech.

Examples to Solve and Submit

1. Suppose we have two implementations of some instructions set architecture. Computer A has clk cycle te of 250 ps and CPI of 2.0 for some program. Computer B has clk cycle te of 500 ps. and CPI of 1.2 for same program. Which computer is faster for this program and by how much?
2. A program runs in 10 sec on computer A which has 4 GHz clk. We are trying to help computer designer to build a computer B that will run this progra in 6 sec. The designer has ~~desired~~ determined that a substantial increase in clk rate is possible; but this increase will effect rest of CPU design, causing computer B to require 1.2 times as many clk cycles g computer A for this progra What clk rate should designer target?
3. Suppose we are considering enhancement of server systes. The new CPU is 10 tims faste ~~than computation is~~ than orginal. Assuming that the orginal CPU is busy with computation 40% of te and is waiting for I/O 60% of the te. What is overall speedup gained by incorporating enhancement.
4. What is significance of MIPS rate? A 40 MHz processor was used to execute a bench mark program with following inst. mix

Inst. type	Inst count	Clk cycle count
Integer Arithmetic	45000	1
Data Tx	32000	2
F p op	15000	2
Control Tx	8000	2

Determine: CPI, MIPS rate and Executi te for program.

5. What are bench marks? Discuss SPEC bench mark
6. Dyfscentiate: Price Peformance, Reliability Peformace, Power peformace