Advanced Computer Architecture (0630561)

Lecture 7

Computer Performance Factors

Prof. Kasim M. Al-Aubidy

Computer Eng. Dept.

Objective:

- How fast runs a parallel computer at its *maximal* potential?
- How fast execution can we expect from a parallel computer for a *concrete application*?
- How do we measure the performance of a parallel computer and the performance improvement we get by using such a computer?

Performance Metrics:

latency: response time, execution time

- good metric for fixed amount of work (minimize time)

throughput: bandwidth, work per time

- = (1 / latency) when there is NO OVERLAP
- > (1 / latency) when there is overlap
  - in real processors, there is always overlap (e.g., pipelining)
- good metric for fixed amount of time (maximize work)

comparing performance

- A is N times faster than B iff
  - perf(A)/perf(B) = time(B)/time(A) = N
- A is X% faster than B iff
  - perf(A)/perf(B) = time(B)/time(A) = 1 + X/100

Performance Metrics:

MIPS (millions of instructions per second)

- (instruction count / execution time in seconds) x 10^{-6}
- – but instruction count is not a reliable indicator of work
  - Prob #1: work per instruction varies (FP mult >> register move)
  - Prob #2: instruction sets aren't equal (3 Pentium instrs != 3 Alpha instrs)

MFLOPS (millions of floating-point operations per second)

- (FP ops / execution time) x 10^{-6}
- like MIPS, but counts only FP operations
  - FP ops have longest latencies anyway (problem #1)
  - FP ops are the same across machines (problem #2)

CPU Performance Equation:

processor performance = seconds / program

- separate into three components (for single core)

$$\frac{\text{instructions}}{\text{program}} \times \frac{\textbf{cycles}}{\textbf{instruction}} \times \frac{\text{seconds}}{\text{cycle}}$$

architecture (ISA) compiler-designer	implementation (micro-architecture) processor-designer	realization (physical layout) circuit-designer

instructions / program: *dynamic* instruction count

- mostly determined by program, compiler, ISA

cycles / instruction: CPI

- mostly determined by ISA and CPU/memory organization

seconds / cycle: cycle time, clock time, 1 / clock frequency

- mostly determined by technology and CPU organization

CPU Performance Equation:

famous example: (RISC vs. CISC)

- assume
  - instructions / program: CISC = P, RISC = 2P
  - CPI: CISC = 8, RISC = 2
  - T = clock period for CISC and RISC (assume they are equal)
- CISC time = P x 8 x T = 8PT
- RISC time = 2P x 2 x T = 4PT
- RISC time = CISC CPU time/2

the truth is much, much, much more complex

- actual data from IBM AS/400
  - CISC time = P x 7 x T = 7PT
  - RISC time = 3.1P x 3 x T/3.1 = 3PT

Execution Time (T):

$$T = I_c * CPI * t$$

T: CPU time (seconds/program) needed to execute a program.

Ic: Number of Instructions in a given program.

CPI: Cycle per Instruction.

t: Cycle time. t=1/f, f=clock rate.

- The CPI can be divided into TWO component terms;
  - processor cycles (p)
  - memory cycles (m)
- The instruction cycle may involve (k) memory references, for example; k=4; one for instruction fetch, two for operand fetch, and one for store result.

$$T = I_c * (p + m * k) * t$$

System Attributes:

$$T = I_c * (p + m * k) * t$$

The above five performance factors (Ic, p, m, k & t) are influenced by these attributes:

FACTORS	Ic	p	m	k	t
Instruction set architecture.	X	X			
Compiler technology.	X	X	X		
CPU implementation & control		X			X
Cache & memory hierarchy				X	X

- **The instruction set architecture affects program length and p.**
- **Compiler design affects the values of I_C, p & m.**
- **The CPU implementation & control determine the total processor time= p*t**
- **The memory technology & hierarchy design affect the memory access time= k*t**

MIPS Rate:

- The processor speed is measured in terms of million instructions per seconds.
- MIPS rate varies with respect to:
  - Clock rate (f).
  - Instruction count (Ic).
  - CPI of a given machine.

$$MIPS = \frac{I_c}{T * 10^6} = \frac{f}{CPI * 10^6} = \frac{f * I_c}{N * 10^6}$$

Where N is the total number of clock cycles needed to execute a given program.

- The CPU Time (T) can also be written as

$$T = I_c * CPI * t = \frac{I_c * 10^{-6}}{MIPS}$$

MIPS rate of a given computer is directly proportional to the clock rate and inversely proportional to the CPI.

Throughput Rate (W_p):

- The Throughput Rate is defined by:

$$W_p = \frac{f}{I_c * CPI} = \frac{MIPS * 10^6}{I_c}$$

- The CPU Throughput is a measure of how many programs can be executed per second, based on MIPS rate and average program length (Ic).
- System Throughput Rate (Ws) is a measure of how many programs a system can execute per unit time.
- Why Ws<Wp?
- Because additional system overheads caused by the I/O, compiler & OS when multiple programs are interleaved for CPU execution by multiprogramming or time sharing operation.

- For CPU design:

$$CPU\ clock\ cycles = CPI * I_c = \sum_{i=1}^{n} CPI_i * I_{ci}$$

- The overall CPI is given by:

$$\sum_{i=1}^{n} CPI_i * I_{ci} = \frac{\sum_{i=1}^{n} CPI_i * I_{ci}}{I_c} = \sum_{i=1}^{n} CPI_i * \frac{I_{ci}}{I_c}$$

Where;

CPI_i: represents the average number of instructions per clock for instruction (i).

Ic_i: represents number of times instruction (i) is executed in a program.

Example:

Suppose you have made the following measurements;

- Frequency of FP operations (other than FP SQR)= 25%
- Average CPI of FP operations= 4
- Average CPI of other operations=1.33
- Frequency of FPSQR=2%
- CPI of FPSQR=20

- Assume that TWO design alternatives are to decrease the CPI of FPSQR to 2, or to decrease the average CPI of all FP operations to 2.5. Compare these two design alternatives?

$$CPI_{original} = \sum_{i=1}^{n} CPI_i * \frac{I_{ci}}{I_c} = 4 * 25\% + 1.33 * 75\% = 2$$

- CPI (with new FPSQR)= CPIoriginal - 2%*[CPIoldFPSQR-CPInewFPSQR]

  » = 2-2%*[20-2]=1.62 for design 1

- CPI (with new FP)= [75%*1.33] + [25%*2.5]= 1.625 for design 2

$$Speedup_{(newFP)} = \frac{CPU\ Time_{(original)}}{CPU\ Time_{(newFP)}} = \frac{I_c * clockcycle * CPI_{(original)}}{I_c * clockcycle * CPI_{(newFP)}}$$

$$Speedup_{(newFP)} = \frac{CPI_{(original)}}{CPI_{(newFP)}} = \frac{2}{1.625} = 1.23$$

Example:

Consider the execution of a task with **100000** instructions on **500** MHz processor. The program consists of **FOUR** major types of instructions:

Instruction Type	CPI	Instruction%
Integer arithmetic	1	60%
Floating point arithmetic	2	20%
Load/Store	4	10%
Memory Reference	6	10%

When the task is executed on a uniprocessor;

- Calculate the average **CPI**?
- Determine the corresponding **MIPS** rate?

Solution:

Average CPI= 1*0.6+2*0.2+4*0.1+6*0.1= 2 cycles/instruction.

$$MIPS = \frac{f}{CPI} = \frac{500MHz}{2cycles/instr} = 250$$

Example:

- Now, when the task given in the previous example is executed on a **FOUR**-processor system with shared memory. Due to the need for synchronization among the **FOUR** program parts, **2000** extra instructions are added to each part.
  - Calculate the average **CPI**?
  - Determine the corresponding **MIPS** rate?
  - Calculate the speedup factor of the **FOUR**-processor system?
  - Calculate the efficiency of the **FOUR**-processor system?
  - Show the interconnection network of this system?

Solution:

Average CPI= 2 cycles/instruction.

MIPS= (4*500MHz)/2=1000

Speedup= [Tex1/Tex4]

Tex1=[Ic/MIPS]=100000/250=0.400 msec

Tex4= =[Ic/MIPS]=[100000+4*2000]/1000=0.108 msec

Speedup=0.4/0.108=3.703

Efficiency=Speedup/#Processors=3.703/4=92.59%