Lecture 5 and Lab5 notes

Performance

1. definition

   $$Performance = \frac{1}{Execution\ time}$$

2. Performance ratio

   what does "x is n times faster than y" mean?

   $$Performance\ ratio = \frac{Performance_x}{Performance_y} = \frac{Execution\ time_y}{Execution\ time_x}$$

3. Measuring execution time

   1. Elapsed time: total responce time, including all aspects

      processing, IO, OS overhead, idle time

   2. CPU time: time spent processing a specific task

      Minus IO time, other jobs' shared

      Includes **user CPU time**(time that the CPU spends executing code in the user space of a program)

      and **system CPU time**(includes the time spent executing system calls, handling hardware interrupts, and managing system resources such as memory and devices)

4. CPU clocking

   • Operation of digital hardware governed by a constant-rate clock

   • Notions: Clock cycle, Clock period, Clock rate/frequency

   • **Clock period** (Clock cycle time): duration of a clock cycle

   • **Clock frequency** (rate): cycles per second

   • $Clock\ cycle\ time = 1/Clock\ rate$

5. CPU time

   $$\begin{aligned} CPU\ time &= Clock\ cycles\ counts * Clock\ period \\ &= \frac{Clock\ cycles\ counts}{Clock\ rate} \end{aligned}$$

   performance can be improved by:

   • reducing the number of cycle counts

   • Increasing clock rate

6. **IC** and **CPI**

   • IC: instruction count for a program, determined by program, ISA, compiler, have nothing to do with the computer itself

• CPI: cycles per instrucion. Generally, CPI means average CPI, because each kind of instruction have different cost. Determined by CPU hardware

• Thus

$$
\begin{aligned}
Clock\ cycles &= IC * CPI \\
CPU\ time &= IC * CPI * T_c \\
&= \frac{IC * CPI}{f}
\end{aligned}
$$

7. Considering different instruction types

$$Clock\ cycles = \sum_{i=1}^{n}(CPI_i * IC_i)$$

$$Avg\ CPI = \frac{\sum_{i=1}^{n}(CPI_i * IC_i)}{\sum_{i=1}^{n} IC_i}$$

Compiler optimization:

• minimize execution time: e.g. GNU gcc -O3

• minimize code size: e.g. GNU gcc -Os

8. Factors

	Instruction Count	CPI	Clock Rate
Algorithm	√	√	
Language	√	√	
Compiler	√	√	
ISA	√	√	√
Organization(hardware)		√	√

Energy dissipation

1. **Energy consumption = dynamic energy + static energy**

• Dynamic energy: (energy spent when transistors switch from 0→1 1→0) is primary

• Static energy: leakage

2. Energy and power

$$
\begin{aligned}
Energy &\propto Capacitive\ load * Voltage^2 \\
&\propto CV^2
\end{aligned}
$$

$$
\begin{aligned}
Power &\propto \frac{1}{2} * Capacitive\ load * Voltage^2 * Frequency\ switched \\
&\propto \frac{1}{2} CV^2 f
\end{aligned}
$$

Multiprocessor

1. a pitfall: Improving an aspect of a computer and expecting a proportional improvement in overall performance.
2. Amdahl's law
   - Calculate **how much a computation** can be sped up by running part of it in parallel

$$T_{improved} = \frac{T_{affected}}{improvement\ factor} + T_{unaffected}$$

Benchmark

1. Select a set of programs that represent the workload
2. Run these programs on each computer
3. Compare the average execution time (**geometric mean**)

$$\sqrt[n]{\prod_{i=1}^{n} Execution\ time_i}$$

4. SPEC ratio

$$\frac{excution\ time_{reference\ machine}}{excution\ time_{measure\ machine}}$$