Performance

Lucas

27 de setembro de 2015

pg 26 O transistor é um interruptor simples de ligar/desligar, controlado por eletricidade. O circuito integrado (CI) combina dezenas de centenas de transistores em um único chip. VLSI (milhares de transistores), chamamos de circuito integrado de larguíssima escala (very large-scale integrated circuit).

pg 28 Tempo de Resposta (Responde Time) também chamado Tempo de Execução. O tempo total requerido para o computador completar uma tarefa qualquer, incluindo acessos ao disco, memória, I/O, overhead do sistema operacional, tempo de execução da CPU, etc.

Vazão (Throughput) também chamado de largura de banda. É outra medida de desempenho, é o número de tarefas completadas por unidade de processamento.

pg 29 Na página 29 temos uma equação simples, comparativa: $Performance_x = \frac{1}{Execution\ Time_x}$ Se o desempenho de x é melhor que o de y então $Performance_x > Performance_y$ ou ainda $\frac{1}{Execution\ Time_x} > \frac{1}{Execution\ Time_y}$ ou também $Execution\ Time_x < Execution\ Time_y$. Isso significa que o tempo de execução em y é mais longo que o tempo de execução em x, bastante simples.

$$\frac{Performance_x}{Performance_y} = \frac{Execution\ Time_y}{Execution\ Time_x} = n$$

pg 30 O conceito de medir desempenho de computador se baseia em: entender qual montante de (trabalho) ele estará processando, em um dado tempo. O processador que realizar o mesmo montante no menor tempo, teoricamente é mais rápido. Tempo de execução do programa é medido em segundos por programa. A definição mais direta é chamada de 'wall clock time', 'response time', ou 'elapsed time'. Esses termos significam o total para uma tarefa ser completada.

pg 31 Por consistência, vamos manter a distinção entre desempenho, baseado em 'elapsed time' e por sua vez baseada em CPU 'execution time'. Clock cycle (Ciclos de Relógio) também chamado de 'clock tick', 'period', 'cycle'. É o tempo para um período de relógio (geralmente em picosegundos, microsegundos, nanosegundos, ou segundos, depende do problema), usualmente o relógio do processador, o qual está rodando à uma certa taxa constante (por exemplo: 4 GHz)

pg 32 Uma fórmula simples que relaciona fatores simples é:

$$CPU\ execution\ time\ for\ a\ program\ = CPU\ clock\ cycles\ for\ a\ program \times Clock\ cycle\ time$$

Um exemplo genérico de melhoramento de performance usando essa fórmula acima é:

$$CPU\ time_A = \frac{CPU\ clock\ cycles_A}{Clock\ rate_A}$$

$$\frac{Seconds}{Program} = \frac{CPU\ clock\ cycles_A}{f \times 10^n\ Hz}$$

$$CPU\ clock\ cycles_A = (t\ [\mathbf{s}]) \times (f \times 10^n\ [\mathbf{Hz}])$$

Onde t é por exemplo: 12 segundos, ou 3 nanosegundos. E $f \times 10^n$ é por exemplo: 9×10^9 hertz, ou 9 Gigahertz. E assim por diante.

$$\frac{Seconds}{Program} = \frac{(CPU\ clock\ cycles_B = CPU\ clock\ cycles_A \times scalar\ cost)}{Clock\ Rate_B}$$

assim, podemos simplificar para:

$$\frac{Seconds}{Program} = \frac{(CPU\ clock\ cycles_A \times scalar\ cost)}{Clock\ Rate_B}$$

Ou seja se eu tenho um computador A que processa um programa qualquer em 13 segundos, sendo que esse tem 3 GHz. Para que eu gere um computador B que processe o mesmo 30% mais rápido que o computador A. Eu precisaria que ele tivesse 4,68 GHz se quisemos que o CPU B processasse a mesma informação em 10 segundos (3 segundos a menos, melhoria de 30%), pagando um custo de 1,4x ciclos a mais, aí vai do seu critério, quanto você quer pagar a mais?

Changes	$\frac{Seconds}{Program} = \frac{(CPU\ clock\ cycles_A \times scalar\ cost)}{Clock\ Rate_B}$
$\uparrow \frac{Seconds}{Program_B}$ (e.g: $3\ s \rightarrow 4\ s$)	Decresce $Clock\ rate_B$
$\uparrow (CPU\ clock\ cycles_A \times improvement)$	Aumenta $Clock\ rate_B$

pg 33 O termo clock cycles per intruction CPI, o qual é a média numérica de ciclos de clock para cada instruções executada, varia bastante de classes de instruções para classes de instruções, exemplos o CPI de instruções do tipo R não vai ser o mesmo CPI para instruções do tipo I, de fato instruções do tipo I contém em seu corpo branch, loads, stores, essas instruções contém veneno para o datapath, da forma que ele leva mais ciclos para processar.

$$CPU\ clock\ cycles = Instructions\ for\ a\ program \times Average\ clock\ cycles\ per\ instruction$$

Suponha você agora que temos um processador X com tempo para cada ciclo de relógio de N pico segundos e em média faz s ciclos por instrução (CPI), e um processador Y com tempo de clock cycle de K pico segundos e em média faz p ciclos por instrução (CPI). Quem é mais rápido?

$CPU\ clock\ cycles_A = (I\ \textbf{instr.} \times s\ \textbf{CPI})\ \textbf{ciclos}$

$CPU\ clock\ cycles_B = (I\ \textbf{instr.} \times p\ \textbf{CPI})\ \textbf{ciclos}$

$CPU\ time_A = CPU\ clock\ cycles_A \times Clock\ cycle\ time$[a]

$CPU\ time_A = I \times s \times N\ \textbf{pico segundos}$

$CPU\ time_B = CPU\ clock\ cycles_B \times Clock\ cycle\ time$

$CPU\ time_B = I \times p \times K\ \textbf{pico segundos}$

Basta comparar ambos os resultados de tempo de CPU e ver qual se desempenhou melhor.

$$\frac{CPU\ time_B}{CPU\ time_A} = \frac{I \times p \times K}{I \times s \times N} = \frac{p\ \textbf{CPI} \times K\ \textbf{pico segundos}}{s\ \textbf{CPI} \times N\ \textbf{pico segundos}}$$

Se por acaso, $p \times K > s \times N$ então CPU_B se desempenha melhor no conjunto de instruções que você passou como parâmetro. Caso contrário CPU_A se desempenha melhor.

$1,7\ \textbf{CPI} \times 190\ \textbf{pico segundos} < 2\ \textbf{CPI} \times 180\ \textbf{pico segundos}$ pois $323\ \textbf{pico segundos} < 360\ \textbf{pico segundos}$ então CPU_A tem 11% menos desempenho que CPU_B.

pg 35 Equação de Desempenho Clássica

$$CPU\ time = Instruction\ count \times CPI \times Clock\ cycle\ time$$

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

Essas fórmulas são bastante úteis pois elas separam os três fatores que afetam o desempenho. Nós podemos usar essas fórmulas para comparar duas implementações diferentes ou também para avaliar uma alternativa se nós sabemos os impactos nesses três parâmetros.

$CPU\ clock\ cycles = \sum_{i=1}^{n}(CPI_i \times C_i)$[b]

Por exemplo: $CPU\ clock\ cycles = 2 \times 3 + 1,7 \times 9 + 1 \times 5 = 26$ ciclos de clock. Veja que poderíamos considerar as contagens de cada classe como sendo $\times 10^6$ instruções $CPU\ clock\ cycles = 2 \times 3\ \times 10^6 + 1,7 \times 9\ \times 10^6 + 1 \times 5\ \times 10^6 = 26\ \times 10^6$ ciclos de clock. Não tem problema nenhum. Para calcular o CPI geral, temos a seguinte fórmula $CPI_{total} = \frac{CPU\ clock\ cycles}{Instruction\ count}$. Ora, então temos: Contagem 1 $3 + 9 + 5 = 17$ e aplicando a fórmula temos o seguinte: $CPI_{total} = \frac{26}{17} = 1,5$

$$Time = \frac{Seconds}{Program} = \frac{Instructions}{Program} \times \frac{Clock\ cycles}{Instruction} \times \frac{Seconds}{Clock\ cycle}$$

Sempre tenha em mente que a medida confiável de medição do desempenho (performance) é Tempo. Por exemplo: mudar o conjunto de instruções para uma contagem menor, pode liderar para tempo do ciclo de relógio maior (ou

[a] O fator tempo de ciclo de relógio, é o tempo de 1 ciclo de relógio. O montante total de ciclos para executar um programa completamente, executar uma tarefa qualquer, vai levar (pensando logicamente) n **cycles** $\times$ *clock cycle time*.

[b] CPI é uma média de ciclos por instruções.

seja, mais lento); ou também pode liderar para um maior CPI (média de ciclos por instruções) que infere mudanças de melhoria na contagem de instrução. Similarmente, por causa que CPI depende do tipo de instrução executada, **o código que executa o menor número de instruções pode não ser o mais rápido**.

1 Questões de performance

1.3.1

1.3.2

1.3.4

1.4.1

1.4.2

1.5.1a

1.6.1a

1.7.2

1.7.3

1.7.4

1.8.1

1.8.2

1.9.3a

1.10.2a

1.10.4a

1.10.5a

1.10.6a