

# Medindo Desempenho nos Computadores

DCA0104 - Arquitetura de Computadores

Diogo Pedrosa

[diogo@dca.ufrn.br](mailto:diogo@dca.ufrn.br)

DCA - CT - UFRN

# Medindo Desempenho

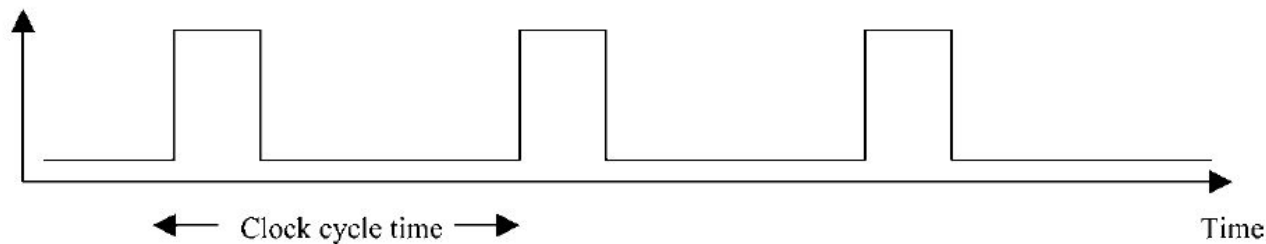
- Quais parâmetros devem ser usados?
  - Memória principal, capacidade de armazenamento em massa, processador mais novo, ...
- Situação
  - Usuário final - tempo usado para executar um programa
  - Gerente de sistemas - total de trabalho em um intervalo de tempo
- Tempo de execução e vazão
- Métrica é decidida por comparação
- Quão rápido um programa pode ser executado em um dado computador?

# Medindo Desempenho

- O que impacta na execução:
  - Algoritmo - determina o número de instruções do código-fonte e o número de operações de E/S realizadas
  - Linguagem de programação, compilador e arquitetura - determina o número de instruções de máquina para cada instrução em alto nível (comandos em código-fonte)
  - Processador e sistemas de memória - determina a velocidade em que as instruções podem ser executadas
  - Sistemas de E/S (hardware e SO) - determina a velocidade em que as operações de E/S podem ser executadas

# Medindo Desempenho

- Necessário determinar o tempo usado pelo computador para executar um determinado programa
- Tempo do ciclo de *clock* (período)



Elementos sequenciais do computador têm suas mudanças de estado sincronizadas pelo sinal de *clock* do sistema.

# Medindo Desempenho

- Contagem do número de instruções executadas em um programa  
- contagem do número TOTAL de ciclos de *clock* usados neste programa

The diagram illustrates the formula for CPU time,  $CPU_{time} = \frac{N_{ciclos}}{f_{clk}}$ . Three callout boxes provide context for the variables:

- A box on the left labeled "tempo de CPU (ou tempo de execução)" has an arrow pointing to the  $CPU_{time}$  term in the numerator.
- A box at the bottom labeled "frequência do *clock*" has an arrow pointing to the  $f_{clk}$  term in the denominator.
- A box on the right labeled "Contagem dos ciclos de *clock* do programa executado" has an arrow pointing to the  $N_{ciclos}$  term in the numerator.

$$CPU_{time} = \frac{N_{ciclos}}{f_{clk}}$$

# Medindo Desempenho

- Instruções diferentes de uma mesma arquitetura têm quantidades de ciclos de *clock* diferentes - valor médio chamado de *CPI* (ciclos de *clock* por instrução)

$$CPI = \frac{N_{ciclos}}{N_{instruc}}$$

Contagem das instruções  
executadas na execução  
do programa

# Medindo Desempenho

- Usando *CPI* para calcular o tempo de execução...

$$CPU_{time} = N_{instruc} \times CPI \times T$$



Período do ciclo de *clock*

# Medindo Desempenho

- Usando *CPI* para calcular o tempo de execução...

$$CPU_{time} = \frac{N_{instruc} \times CPI}{f_{clk}}$$



# Medindo Desempenho

- Os conjuntos de instruções têm classes que agregam instruções de uma mesma categoria (por exemplo, instruções que requerem operações na ULA, instruções que acessam a memória principal, instruções de desvio, ...)

# Medindo Desempenho

- Supondo que um programa, executando em um processador de uma determinada arquitetura, com  $n$  classes diferentes de instruções, mas com  $CPI$  de cada classe conhecida...

$$CPI = \frac{\sum_{i=1}^n CPI_i \times I_i}{N_{instruc}}$$

$CPI_i \rightarrow$  total de ciclos de *clock* por instrução de cada classe  $i$  do conjunto de instruções na execução do programa

$I_i \rightarrow$  número de vezes que uma instrução do tipo  $i$  foi executada no programa

# Exemplo

Calcular o valor de *CPI* para uma máquina *A* na qual as seguintes medidas de desempenho foram gravadas quando executou-se um determinado conjunto de programas de avaliação. A frequência do *clock* da CPU é de 200 MHz.

<b>Categoria de instrução</b>	<b>Porcentagem de ocorrência (%)</b>	<b>Nº de ciclos de <i>clock</i> por instrução</b>
Operação na ULA	38	1
Acesso à memória	15	3
Desvios	42	4
Outros	5	5

# Exemplo

Assumindo um total de  $N_{instruc}$  instruções executadas...

$$CPI_A = \frac{\sum_{i=1}^4 CPI_i \times I_i}{N_{instruc}}$$

$$CPI_A = \frac{0,38N_{instruc} \times 1 + 0,15N_{instruc} \times 3 + 0,42N_{instruc} \times 4 + 0,05N_{instruc} \times 5}{N_{instruc}}$$

$$CPI_A = 0,38 \times 1 + 0,15 \times 3 + 0,42 \times 4 + 0,05 \times 5$$

$$CPI_A = 2,76$$

# Observações

- *CPI* - reflete a organização e a arquitetura do processador
- Contagem de instruções - reflete a arquitetura e o compilador usado
- Outro parâmetro: milhões de instruções por segundo

$$MIPS = \frac{N_{instruc}}{CPU_{time} \times 10^6} = \frac{f_{clk}}{CPI \times 10^6}$$

# Exemplo

Considerando que o mesmo conjunto de programas de avaliação tenha sido testado em uma máquina *B*, com coleta de medidas apresentada a seguir, qual é a taxa *MIPS* para essa máquina *B* e, também, *A*? Considere que ambas têm uma frequência de 200 MHz.

<b>Categoria de instrução</b>	<b>Percentual de ocorrência (%)</b>	<b>Nº de ciclos de <i>clock</i> por instrução</b>
Operações na ULA	35	1
Acesso à memória	30	2
Desvios	15	3
Outros	20	5

# Exemplo

$$CPI_A = 2,76$$

$$MIPS_A = \frac{f_{clkA}}{CPI_A \times 10^6} = \frac{200 \times 10^6}{2,76 \times 10^6} = 70,24$$

$$CPI_B = 0,35 \cdot 1 + 0,30 \cdot 2 + 0,20 \cdot 5 + 0,15 \cdot 3 = 2,40$$

$$MIPS_B = \frac{f_{clkB}}{CPI_B \times 10^6}$$

$$MIPS_B > MIPS_A$$

$$MIPS_B = \frac{200 \times 10^6}{2,4 \times 10^6} = 83,67$$

# Observações

- Evita-se o uso da taxa MIPS para medida de desempenho quando se tem máquinas de arquiteturas diferentes



# Exemplo

Calcular o tempo de execução e taxa MIPS para os computadores testados a seguir. Considerar uma frequência de 200 MHz.

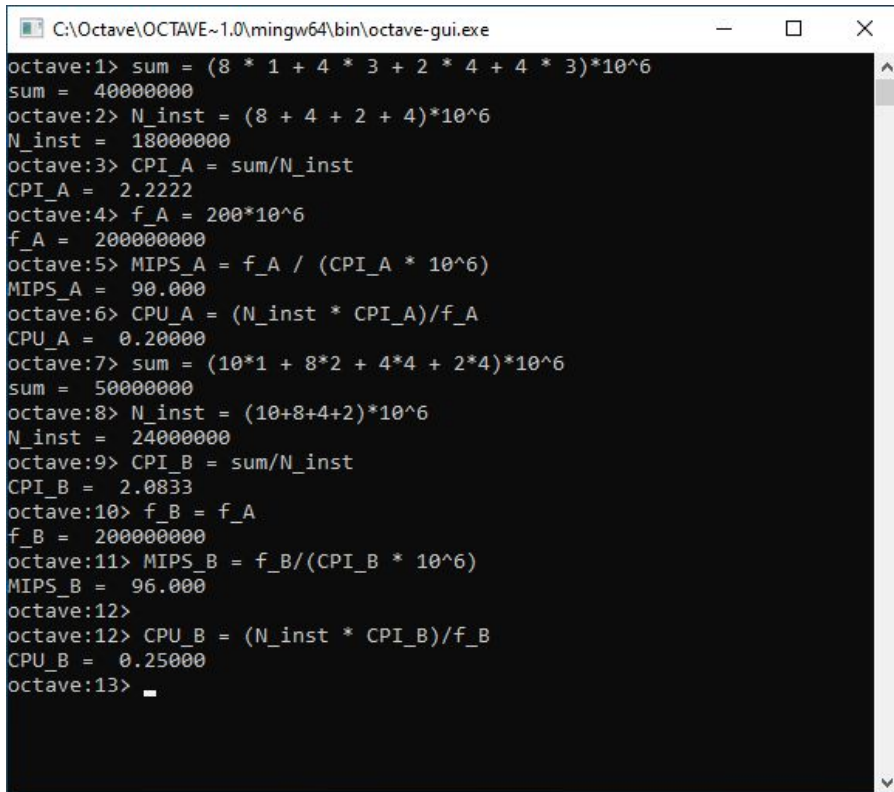
# Exemplo

Categoria	Núm. instruções (em milhões)	Núm. ciclos de <i>clock</i> por instrução
<b><i>Máquina A</i></b>		
ULA	8	1
Memória	4	3
Desvios	2	4
Outros	4	3
<b><i>Máquina B</i></b>		
ULA	10	1
Memória	8	2
Desvios	2	4
Outros	4	3

# Exemplo

Ver exemplo resolvido no Octave.

# Exemplo



```
C:\Octave\OCTAVE~1.0\mingw64\bin\octave-gui.exe
octave:1> sum = (8 * 1 + 4 * 3 + 2 * 4 + 4 * 3)*10^6
sum = 40000000
octave:2> N_inst = (8 + 4 + 2 + 4)*10^6
N_inst = 18000000
octave:3> CPI_A = sum/N_inst
CPI_A = 2.2222
octave:4> f_A = 200*10^6
f_A = 200000000
octave:5> MIPS_A = f_A / (CPI_A * 10^6)
MIPS_A = 90.000
octave:6> CPU_A = (N_inst * CPI_A)/f_A
CPU_A = 0.20000
octave:7> sum = (10*1 + 8*2 + 4*4 + 2*4)*10^6
sum = 50000000
octave:8> N_inst = (10+8+4+2)*10^6
N_inst = 24000000
octave:9> CPI_B = sum/N_inst
CPI_B = 2.0833
octave:10> f_B = f_A
f_B = 200000000
octave:11> MIPS_B = f_B/(CPI_B * 10^6)
MIPS_B = 96.000
octave:12> CPU_B = (N_inst * CPI_B)/f_B
CPU_B = 0.25000
octave:13> _
```

# Medindo Desempenho

- Outra taxa: *MFLOPS*
- Milhões de instruções de ponto flutuante executados por segundo
- Somente para instruções de ponto flutuante
- Não consistente

$$MFLOPS = \frac{N_{instruc}^{fp}}{CPU_{time}^{fp} \times 10^6}$$

Número de instruções de ponto flutuante executadas

Tempo de execução das instruções de ponto flutuante

# Com programas de avaliação

Total de programas de  
avaliação testados

$$MG = \sqrt[n]{\prod_{i=1}^n CPU_{time}^i}$$

Média geométrica

Tempo de execução do  $i$ -ésimo  
programa de avaliação no  
computador testado

# Com programas de avaliação

$$MA = \frac{1}{n} \sum_{i=1}^n CPU_{time}^i$$

Média aritmética

Tempo de execução do  $i$ -ésimo  
programa de avaliação no  
computador testado

Total de programas de  
avaliação testados

# Lei de Amdahl

- Equação que dá o incremento de velocidade em uma máquina após a realização de uma melhoria em seu projeto.
- Gene Amdahl, 1967
- Focado em processamento paralelo (múltiplos processadores)

$$Speedup = \frac{CPU_{time}^{before}}{CPU_{time}^{after}} = \frac{D_{after}}{D_{before}}$$

Diagram illustrating the Amdahl's Law equation with annotations:

- Tempo de execução antes da melhoria** (Execution time before improvement) points to  $CPU_{time}^{before}$ .
- Tempo de execução depois da melhoria** (Execution time after improvement) points to  $CPU_{time}^{after}$ .
- Desempenho depois da melhoria** (Performance after improvement) points to  $D_{after}$ .
- Desempenho antes da melhoria** (Performance before improvement) points to  $D_{before}$ .



# Lei de Amdahl

- Se considerarmos a aplicação para processadores simples e paralelos...

$$Speedup = \frac{CPU_{time}^{single}}{CPU_{time}^{parallel}}$$

Tempo de execução do programa em um processador único

Tempo de execução do programa em processadores paralelos

# Lei de Amdahl

- Considerando...
  - $f$  - fração do tempo de execução de um programa que pode ser paralelizável
  - $(1 - f)$  - fração do tempo de execução de um programa que não pode ser paralelizado
  - $T$  - tempo de execução do programa
  - $N$  - número de processadores paralelos

$$Speedup = \frac{T(1 - f) + Tf}{T(1 - f) + \frac{Tf}{N}} = \frac{1}{(1 - f) + \frac{f}{N}}$$

Se  $f$  for próximo de 0, não há incremento de velocidade com a adoção de processamento paralelo.

Se  $N$  for grande, o incremento de velocidade fica limitado a  $1/(1-f)$

# Referências

- Livro do Stallings: capítulo 2, seção 2.5 (“Avaliação de Desempenho”)
- Livro do Patterson: capítulo 4 (“Avaliando e Compreendendo o Desempenho”)
- Livro “Fundamentals of Computer Organization and Architecture” (Abd-El-Barr e El-Rewini): capítulo 1, seção 1.4 (“Performance Measures”)

# Com programas de avaliação

$$MA = \frac{1}{n} \sum_{i=1}^n CPU_{time}^i$$

Média aritmética

Total de programas de  
avaliação testados

Tempo de execução do  $i$ -ésimo  
programa de avaliação no  
computador testado