Arquitetura de Computadores

Desempenho de Máquinas Digitais

Objetivos da Aula

- descrever os principais fatores que influenciam o desempenho
- ensinar como medir, informar e documentar os aspectos relativos ao desempenho de um computador

Fatores de maior influência do hardware

- A utilização por parte do compilador das instruções da máquina na geração do código de um programa
- A maneira como o hardware implementa uma instrução
- A maneira como a memória e os dispositivos de E/S se comportam durante o processamento do programa

Definição de Desempenho

- Depende do enfoque
- Exemplo clássico do desempenho do avião:
  - velocidade de cruzeiro? ou
  - capacidade de transporte de passageiros?
- Computadores
  - tempo de execução de um único programa?
  - tempo de execução de um grupo de programas usuais (workload)?
    - Aqui entra o conceito de “throughput”, que representa a quantidade de trabalho executada em um intervalo de tempo

Throughput e Tempo

- throughput e tempo de execução são dois conceitos que muitas vezes se confundem, para esclarer, considere a seguinte modificação:
  - a) substituição dos processadores presentes nas máquinas por um modelo mais rápido (upgrade).
  - b) alocação de processadores adicionais em um sistema que usa vários processadores para executar programas diferentes.
- Como afetar o throughput, o tempo de execução ou ambas as coisas?

- ''Throughput’’ - O throughput pode ser traduzido como a taxa de transferência efetiva de um sistema.
- A taxa de transferência efetiva de um determinado sistema (uma rede de roteadores por exemplo) pode ser menor que a taxa de entrada devido às perdas e atrasos no sistema

Throughput e Tempo

- mudanças no tempo de execução afetam o throughput e vice-versa
  - foco no tempo de execução ou tempo de resposta.
- “Para maximizar o desempenho temos que minimizar o tempo de resposta!”
- Matematicamente isto pode ser expresso como:

Throughput e Tempo

- Matematicamente isto pode ser expresso como:

$$Performance = \frac{1}{\text{tempo de execução}}$$

$$P = \frac{1}{te}$$

- Onde:
- P = performance ou desempenho.
- *te* = tempo de execução.

Throughput e Tempo

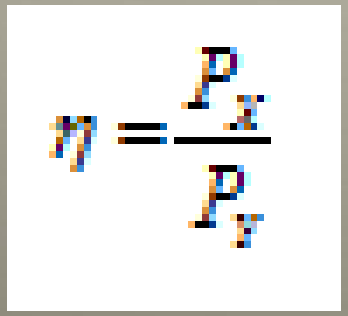
- Isto significa que, considerando duas máquinas X e Y, se o desempenho de X for melhor que o desempenho de Y, teremos:

$$P_X > P_Y$$
$$\frac{1}{te_X} > \frac{1}{te_Y}$$

  - se X é mais rápido que Y, o tempo de execução de Y é maior que o tempo de execução de X

Throughput e Tempo

- Como muitas vezes é preciso relacionar o desempenho de duas máquinas digitais de modo quantitativo, mostrando quantas vezes uma máquina é mais rápida que outra, foi criada o conceito de performance relativa.



η ➔ performance relativa.

Throughput e Tempo

- Dessa forma, se a máquina X for n vezes mais rápida que a máquina Y:

$$\eta = \frac{P_x}{P_y} = \frac{te_y}{te_x}$$

Throughput e Tempo

- Exemplo:
  - Sabe-se que uma máquina A roda um determinado programa em 10 segundos e que, além disso, é 50% mais rápida que outra máquina B. Calcule o tempo de execução do mesmo programa na máquina B?

- Resposta:

$$te_B = 1,5te_A$$

$$te_B = 1,5 \cdot 10$$

$$te_B = 15s$$

Medidas de Desempenho

- tempo de execução
  - medida de desempenho mais usual em um computador
    - “tempo de resposta”
- tempo total decorrido para completar uma dada tarefa, incluindo
  - acesso a disco e a memória, atividades de E/S, etc.

Medidas de Desempenho

- o tempo em um computador é compartilhado e o processador pode trabalhar mais de um programa ao mesmo tempo
  - o sistema deve otimizar o throughput
  - é necessário fazer distinção entre o:
    - tempo total para a execução de um programa
    - tempo total gasto pelo processador trabalhando em proveito deste programa
      - tempo de processador.

Medidas de Desempenho

- tempo de resposta visto por um usuário
  - tempo total de execução
    - e não o tempo de processador
- O tempo de processador
  - tempo gasto na execução das instruções do programa (tempo do usuário) e o
  - tempo gasto pelo sistema operacional para executar tarefas em beneficio do próprio programa

Medidas de Desempenho

- A obtenção de todos estes dados é complexa e geralmente obtida por
  - simulação dos dispositivos eletrônicos ou
  - através de ferramentas de hardware especiais

Medidas de Desempenho

- Um fator que facilita a análise do desempenho das maquinas atuais é o fato de que todos os computadores usam o clock, que roda a uma taxa constante e que determina momentos de ocorrência de eventos do próprio hardware.
  - Esses intervalos de tempo discretos são chamados de
    - ciclos de clock ou de relógio.

Relação Entre as Métricas

- Usuários e projetistas enxergam o desempenho sob diferentes métricas, então vamos estabelecer uma relação entre as métricas utilizadas por cada um deles.

Relação Entre as Métricas

- desempenho do processador
  - a medida mais básica é o tempo de execução gasto no processador.
  - relação entre os ciclos de clock e o tempo do ciclo de clock

$$te = NC.T_c = \frac{NC}{f_c}$$

Onde:

NC ➔ Número de ciclo de clocks do processador para o programa.

T_c ➔ periodo de clock.

f_c ➔ freqüência de clock.

Relação Entre as Métricas

- a solução mais obvia para a melhora do desempenho de um computador é a elevação da freqüência de clock
- entretanto, duas máquinas com freqüência de clock idênticas não necessariamente apresentam o mesmo desempenho

Relação Entre as Métricas
Exemplo

- Suponha que um dado programa roda em 10 segundos em um computador A, com clock de 400MHz. Um novo computador B deverá ser construído para rodar o mesmo programa em 6 segundos.
- A tecnologia utilizada para aumentar a freqüência da máquina provoca reflexos em outros parâmetros de desempenho, fazendo com que a máquina B necessite de 1,2 vezes mais ciclos de clock do que a máquina A para executar tal programa. Qual a freqüência de clock necessária para a nova máquina?

Resolução:

$te = NC.Tc = NC / fc$

$compA - Te = 10s$

$fc = 400MHz$

$compB - Te = 6s$

$fc = ?$

$compA$

$NC = 10.400M$

$compB$

$NC = 1,2.4000M = 4800M$

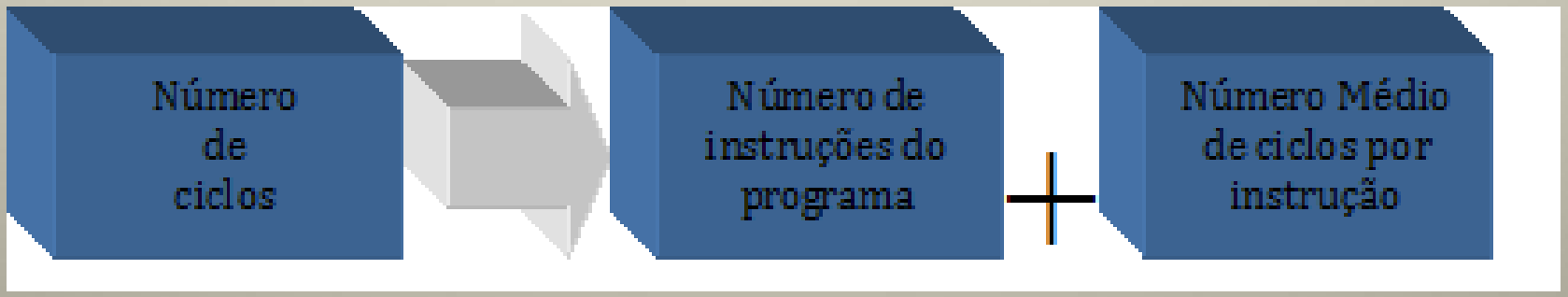
$fcB = NC / Te = 4800M / 6 = 800MHz$

Interface Hardware/Software

- influência do número de instruções necessárias à execução de um programa

- Uma maneira de conceituar o tempo de execução é torná-lo igual ao número de instruções executadas, multiplicado pelo tempo médio de execução de cada instrução

Interface Hardware/Software



- NC = NI . CPI
  - onde CPI = clocks cycles per instruction

Interface Hardware/Software

- Conclusão
  - Tempo do processador
    - = Número de instruções . CPI . Ciclo de clock
  - ou
  - Tempo do processador
    - = (Número de instruções . CPI) / Frequência de clock

Interface Hardware/Software

$$t_E = NI.CPIT_c = \frac{NI.CPI}{f_c}$$

Interface Hardware/Software
Exemplo

- Supor que temos duas implementações diferentes da mesma arquitetura do conjunto de instruções. A máquina A tem um período de ciclo de 1ns e uma CPI de 2,0; considerando um programa qualquer. A máquina B tem um ciclo de clock de 2ns e uma CPI de 1,2 para o mesmo programa. Qual das duas máquinas executa este programa mais rapidamente? Calcule quanto uma é mais rápida que a outra.

$$Te = NI.CPI.Tc$$

$$NI = Te / CPI.Tc$$

$$NI = Te / CPI.Tc$$

$$TeA = NI.2.1n$$

$$TeB = NI.1,2.2n$$

$$TeA / TeB = 1 / 1,2$$

$$TeB = 1,2TeA$$

Interface Hardware/Software

- como determinar os valores de cada um desses parâmetros?
  - Pode-se rodar um programa para medir o tempo de processador.
  - O período de clock pode ser obtido no manual do fabricante do hardware.
  - A quantidade de instruções e a CPI são mais difíceis de obter.

Interface Hardware/Software

- A quantidade de instruções pode ser medida através da utilização de ferramentas de software especiais
  - uma segunda opção seria a utilização de contadores em nível de hardware
- Já a CPI depende de muitos fatores, variando com a aplicação ou com implementações diferentes das mesmas instruções. Algumas vezes obtem-se a CPI por simulação

Interface Hardware/Software

- Em algumas situações é possível calcular o número de ciclos do processador da seguinte forma:

$$NC = \sum_{i=1}^{n} CPI_i \times C_i$$

- Onde:
  - C_i = número de instruções da classe i
  - CPI_i = número médio de ciclos por instruções para cada classe i
  - n = número de classes de instrução.

Interface Hardware/Software
Exemplo

- Um projetista de compilador está tentando decidir entre duas seqüências de código para uma dada máquina. Para tanto, dispõe dos seguintes dados:

Classe de Instrução	CPI por Classe
A	1
B	2
C	3

Interface Hardware/Software Exemplo

- O código a ser gerado pode seguir as seguintes seqüências:

	Número de instruções por Classe		
Sequência de código	A	B	C
1	2	1	2
2	4	1	1

  - Qual das seqüências executa mais instruções?
  - Qual é executada mais rapidamente?
  - Qual a CPI para cada seqüência?

- Número de instruções executadas por sequência:
  - 2 + 1 + 2 = 5; sequência 1;
  - 4 + 1 + 1 = 6; sequência 2.

    - Logo a sequência 1 executa menos instruções que a 2.

- Podemos usar a equação dos ciclos de clock, baseada na quantidade de instruções e na CPI para encontrar o número total de ciclos de clock para cada sequência:

$$NC = NI$$

Aplicando a Fórmula teremos:

$$\text{Ciclos de Clock}_1 = (2 \times 1) + (1 \times 2) + (2 \times 3) = 10 \text{ Ciclos}$$

$$\text{Ciclos de Clock}_2 = (4 \times 1) + (1 \times 2) + (1 \times 3) = 9 \text{ Ciclos}$$

- A sequência dois é a mais rápida, pois executa mais instruções com um menor número de ciclos de clock.

- Considerando que a sequência de código 2, gasta menos ciclos de clock no total, mas executa mais instruções, é de se esperar que ela tenha uma CPI mais baixa.

$$NC = NI \times CPI$$

$$CPI = \frac{NC}{NI}$$

$$CPI_1 = \frac{NC_1}{NI_1} = \frac{10}{5} = 2$$

$$CPI_2 = \frac{NC_2}{NI_2} = \frac{9}{6} = 1,5$$

Interface Hardware/Software
Exemplo

- Conclusão
  - é arriscado utilizar um único fator (quantidade de instruções) para avaliar o desempenho.

Programas para Avaliar a Desempenho

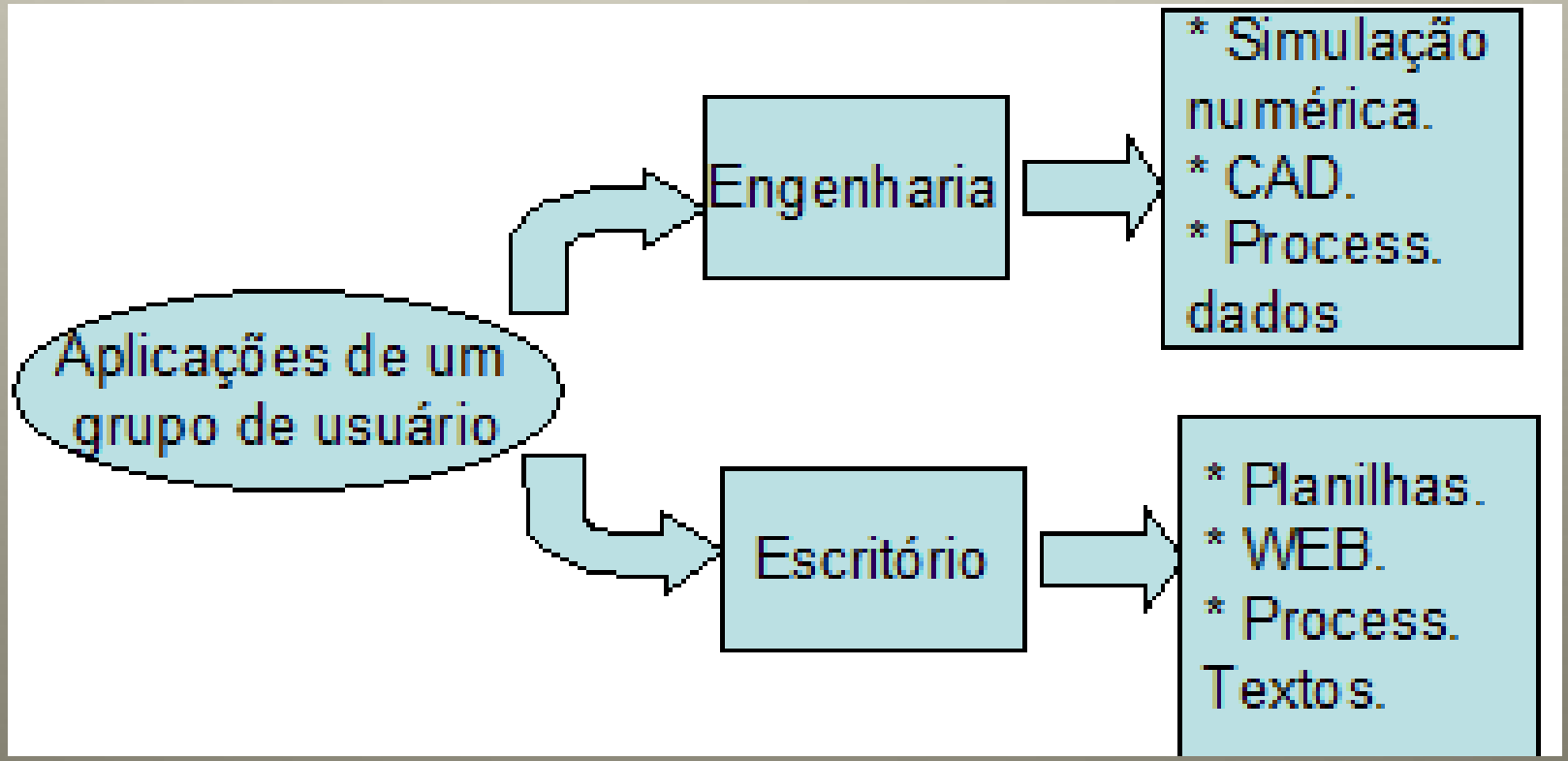
- Um usuário de computador que rode sempre os mesmos programas (workload) seria um bom candidato para avaliar uma nova máquina, embora esta não seja a situação mais usual.
  - Por isso, a alternativa mais utilizada são os benchmarks.

Programas para Avaliar a Desempenho

- Os benchmarks são programas que formam workloads com o objetivo de estimar a performance do workload real, utilizado por determinados grupos de usuários.
  - Hoje em dia acredita-se que os melhores programas benchmarks são aplicações reais.
  - **Workload**, em computação, é o envio de trabalhos a serem processados em um ambiente computacional na nuvem de processamento. ou seja, o trabalho a ser desenvolvido será realizado nos processadores na internet e o trabalho pronto direcionado ao dispositivo de saída do solicitante.

Programas para Avaliar a Desempenho

- X



Programas para Avaliar a Desempenho

- Quais os maiores problemas nestes casos?
  - O uso de aplicações reais como benchmarks torna fácil encontrar meios de acelerar artificialmente a execução dos benchmarks.
  - O compilador pode reconhecer fragmentos especiais de códigos dos benchmarks e gerar uma seqüência de instruções que sejam particularmente eficientes na execução desse fragmento.
  - O projetista de hardware pode também fazer com que determinado conjunto de instruções rode mais rápido por pertencer a um benchmark.

Programas para Avaliar a Desempenho

- TAIS ARTIFÍCIOS AJUDARIAM NA EXECUÇÃO DE OUTROS PROGRAMAS?
- Classificação usual dos programas utilizados para realizar a medida do desempenho
  - Sintético - são aqueles cujo código não faz nenhuma computação útil, não representam nenhuma aplicação. Na prática, somente exercita alguns componentes do computador. Geralmente tentam determinar a frequência media de instruções típicas comumente utilizadas e recria-las em um programa.

Programas para Avaliar a Desempenho

- Kernel – são baseados no fato de que a maior parte da computação de um programa é concentrada em uma pequena parte de seu código. Esta pequena parte, chamada de núcleo (kernel), é extraída do programa e usada como benchmark.
  - Deve ser ressaltado que eles não servem para avaliar completamente o desempenho de uma máquina.
  - São bastante interessantes por sua simplicidade e pequeno tamanho

Programas para Avaliar a Desempenho

- Algoritmo – são algoritmos bem definidos, geralmente implementações de métodos conhecidos em computação numérica como, por exemplo, os métodos de resolução de equações lineares.

- Aplicação - programas completos que resolvem problemas científicos bem definidos. Um exemplo é o SPEC.

Comparação e Documentação de Desempenho

- Uma vez selecionados os programas para usar como benchmarks, deve-se selecionar exatamente o que se deseja medir.
- Existem duas possibilidades:
  - Tempo de resposta.
  - Throughput.
- Após esta decisão, deve-se catalogar as informações de forma clara, assim como se trata qualquer informação técnica.

Comparação e Documentação de Desempenho - Exemplo

	Computador A	Computador B
Programa 1 (s)	1	10
Programa 2 (s)	1000	100
Tempo total	1001	110

- Poderia-se afirmar que para o programa 1 a máquina A é dez vezes mais rápida que a B.
- Poderia-se também afirmar que para o programa 2 a máquina B é dez vezes mais rápida que a A.

Comparação e Documentação de Desempenho - Exemplo

- Este tipo de documentação acaba se tornando um tanto quanto confusa.
- Uma maneira simples e clara é utilizar o conceito de performance.

$$h = \frac{P_B}{P_A} = \frac{T_{eA}}{T_{eB}} = \frac{1001}{110} = 9{,}1$$

Comparação e Documentação de Desempenho

- Também deve ter ficado claro que se o workload seja composto unicamente pela execução dos programas 1 e 2, num número idêntico de vezes, pode-se afirma que a máquina B é 9,1 vezes mais rápida que a máquina A.
- No entanto, caso um dos programas rode mais vezes que o outro, pode-se apelar para a média aritmética ponderada, atribuindo pesos a cada um dos programas.

Comparação e Documentação de Desempenho

- O conjunto de benchmarks mais usado atualmente é o SPEC (System Performance Evaluation Cooperative).
  - A medida final do desempenho é obtida por meio de média geométrica das razões.

Comparação e Documentação de Desempenho

- Dada uma arquitetura do conjunto de instruções, as melhoras no desempenho costumam ser decorrentes de:
  - Acréscimo na frequência de clock.
  - Melhora da organização do processador.
  - Modificação no projeto do compilador que resulta em menor número de instruções.

Comparação e Documentação de Desempenho

- Dados práticos comprovam que, quando a frequência de clock cresce a um determinado fator, o desempenho cresce segundo um fator menor.
  - A explicação para este comportamento é a perda do desempenho do sistema de memória.

MIPS, MOPS e FLOPS

- O MIPS (Million Instructions per Second) é uma medida ainda muito utilizada na performance.
  - Um desdobramento tão popular, quanto equivocado do MIPS é o MIPS de pico, que não leva em consideração as diferenças entre os programas.

$$MIPS = \frac{NI}{te.10^6}$$

MIPS, MOPS e FLOPS

- Medidas de performance também são realizadas utilizando MOPS e FLOPS.
  - milhões de operações por segundo e
  - operações de ponto flutuante por segundo.

Falhas na Melhora do Desempenho

- Muitas vezes, usuários e projetistas enganam-se ao esperar que a melhora em um dos elementos que influencia no desempenho, resulte em uma melhora no desempenho total igual ao ganho inicial.

Falhas na Melhora do Desempenho Exemplo

- Considere por exemplo uma máquina que execute um determinado programa em 100 segundos, onde 80 segundos são utilizados com operações de multiplicação.
- Qual seria a melhora obtida no desempenho total, caso alguma mudança fosse executada na arquitetura da máquina para conseguir executar as operações de multiplicação 5 vezes mais rápido?

Falhas na Melhora do Desempenho Exemplo

$$te = \frac{ta}{mm} + tnf = \frac{80}{5} + 20 = 36s$$

- Onde:
- • ta = tempo afetado.
- • mm = montante da melhora.
- • tnf = tempo não afetado.

- Considerando que estamos buscando uma performance 5 vezes melhor que a anterior o novo tempo de execução deve ser de 20 segundos.
- Em suma não há o que se possa melhorar na multiplicação de forma a conseguir uma melhora de 5 vezes na performance, pois a multiplicação responde por 80% do Workload.

- A melhora da performance possível, considerando um dado aperfeiçoamento, é limitada pela frequência de uso da operação que sofreu a melhora.
- *Esse conceito é conhecido em computação como lei de Amdahl.*

Falhas na Melhora do Desempenho Exemplo

- Outra possível causa de falha na documentação do desempenho é a utilização da unidade de medida MIPS. Como dito anteriormente, que dentre outros problemas, não permite uma medida real de desempenho porque o valor de MIPS varia de programa para programa dentro de um mesmo computador, impedindo assim que se determine um número MIPS característico para a máquina.