


# Arquitetura e Organização de Computadores – 5cop090



**Questões de desempenho**

## Objetivos do Capítulo

- Compreender as principais questões de desempenho que se relacionam com o projeto do computador.
- Explicar as razões de mudar para a organização multicore e entender a relação entre recursos de cache e de processador em um chip único.
- Fazer distinção entre organizações multicore, MIC e GPGPU.

## Objetivos do Capítulo

- Resumir algumas das questões relacionadas à avaliação do desempenho do computador.
- Discutir os benchmarks do SPEC.
- Explicar as diferenças entre as médias aritmética, harmônica e geométrica.

# Elaboração do projeto visando o desempenho

- O que dá aos processadores Intel x86 ou computadores mainframe da IBM uma potência incrível é a busca implacável de velocidade pelos fabricantes de chip de processador.
- Consequentemente, os projetistas de processadores deverão aparecer com técnicas ainda mais elaboradas para alimentar o processador. Por exemplo:
  - Realização de pipeline
  - Predição de desvio
  - Execução superescalar
  - Análise de fluxo de dados
  - Execução especulativa

## Elaboração do projeto visando o desempenho

- **Realização de pipeline:** a execução de uma instrução envolve uma série de operações, como buscar a instrução, decodificar as diversas partes do código de operação (opcode), buscar operandos, realizar cálculos e assim por diante. Utilizar o pipeline possibilita que um processador trabalhe simultaneamente em diversas operações ao mover os dados ou instruções em um pipe conceitual com todos os estágios do pipe processando simultaneamente. Por exemplo, enquanto uma instrução está sendo executada, o computador está decodificando a instrução seguinte. Esse é o mesmo princípio visto em uma linha de montagem.

# Elaboração do projeto visando o desempenho

- **Predição de desvio:** o processador antecipa o código de instrução buscando a partir da memória e prediz quais desvios ou grupos de instruções provavelmente serão processados a seguir. Se o processador acerta a maior parte do tempo, ele pode pré-buscar as instruções corretas e agrupá-las, de modo que seja mantido ocupado. Os exemplos mais sofisticados dessa estratégia predizem não somente o próximo desvio, mas diversos desvios à frente. Assim, a predição dos desvios potencialmente aumenta a quantidade de trabalho disponível para o processador executar.

# Elaboração do projeto visando o desempenho

- **Execução superescalar:** é a capacidade de enviar mais de uma instrução em todos os ciclos de clock de processador. Em efeito, diversos pipelines paralelos são usados.

# Elaboração do projeto visando o desempenho

- **Análise de fluxo de dados:** o processador analisa quais instruções são dependentes dos resultados, ou dados, umas das outras, a fim de criar uma lista otimizada de instruções. De fato, as instruções listadas para serem executadas quando prontas, independentemente do pedido do programa original. Isso previne atrasos desnecessários.



# Elaboração do projeto visando o desempenho

- **Execução especulativa:** usando previsão de desvio e análise do fluxo de dados, alguns processadores especulativamente executam instruções antes de seu surgimento real na execução do programa, mantendo os resultados em locais temporários. Isso permite que o processador mantenha seus mecanismos de execução ocupados o máximo possível, executando instruções que provavelmente serão necessárias.

## Balanco do desempenho

- Embora a capacidade de processamento do processador tenha crescido em uma velocidade espantosa, **outros componentes críticos do computador não a acompanharam.**
- O resultado é a necessidade de procurar o **balanco do desempenho.**
- É um ajuste **da organização e da arquitetura** para compensar a diferença entre as capacidades dos diversos componentes.
- O problema criado por essas diferenças **é particularmente importante na interface entre o processador e a memória principal.**

## Balanço do desempenho

- Outra área de foco de projeto é o **tratamento dos dispositivos de E/S**.
- À medida que os computadores se tornam mais rápidos e mais capazes, aplicações sofisticadas são desenvolvidas para dar suporte ao uso de periféricos com demandas intensas de E/S.
- A chave em tudo isso é o equilíbrio.
- Os projetistas constantemente lutam para equilibrar as demandas de fluxo e processamento dos componentes do processador, memória principal, dispositivos de E/S e estruturas de interconexão.

# Melhorias na organização e na arquitetura do chip

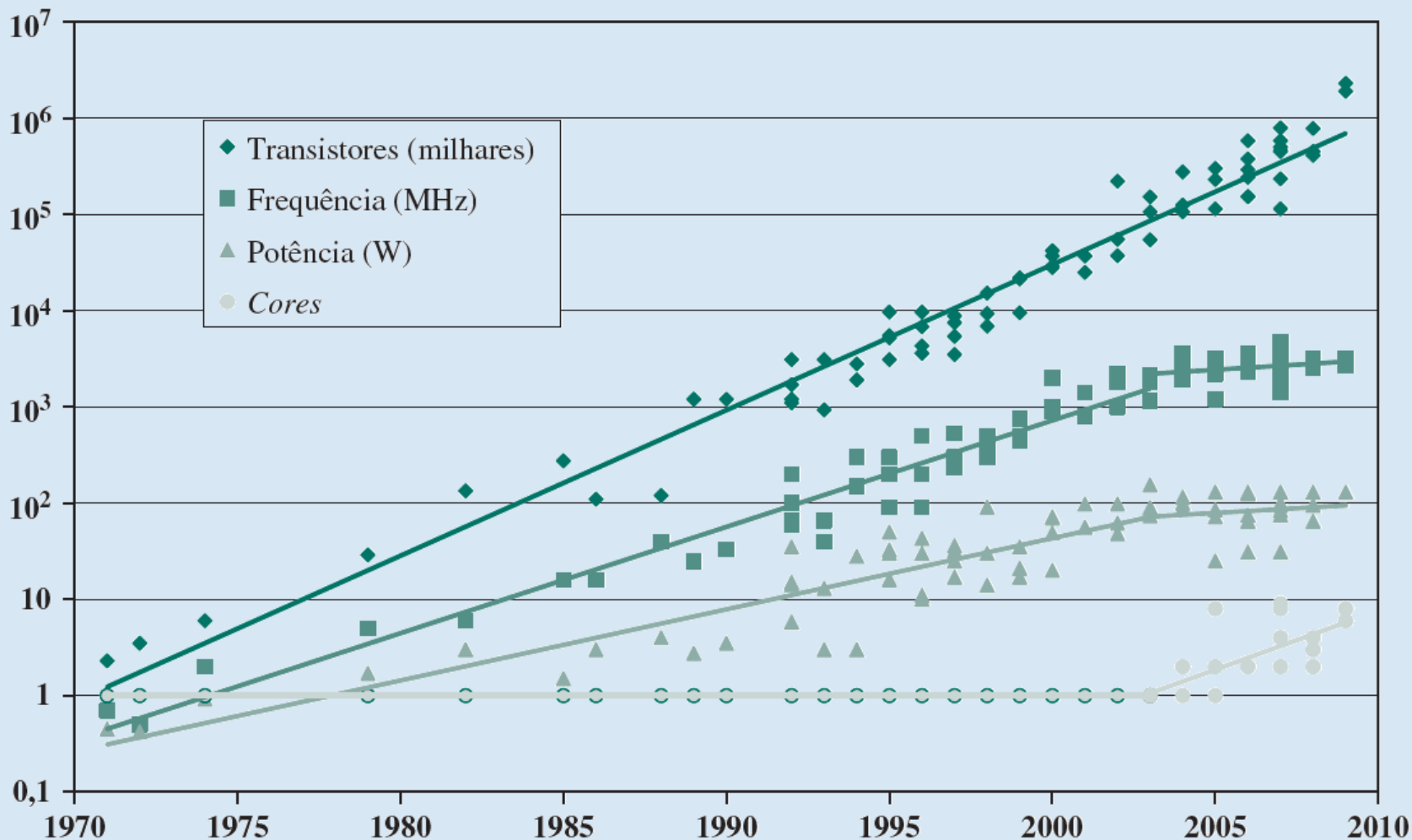
- Permanece a necessidade de aumentar a velocidade do processador e para isso, existem três técnicas:
  1. Aumentar a velocidade de hardware do processador.
  2. Aumentar o tamanho e a velocidade das caches interpostas entre o processador e a memória principal.
  3. Fazer mudanças na organização e na arquitetura do processador, que aumentam a velocidade efetiva da execução da instrução.

# Melhorias na organização e na arquitetura do chip

- À medida que a velocidade do clock e a densidade lógica aumentam, diversos obstáculos tornam-se mais significativos:
  1. **Potência:** à medida que a densidade da lógica e a velocidade do clock em um chip aumentam, também aumenta a densidade de potência (Watts/cm<sup>2</sup>).
  2. **Atraso de RC:** o atraso aumenta à medida que o produto RC aumenta.
  3. **Latência e taxa de transferência da memória:** a velocidade de acesso à memória (latência) e a taxa de transferência limitam as velocidades do processador.

# Melhorias na organização e na arquitetura do chip

## ➤ Tendências dos processadores:



## Multicore, MICs e GPGPUs

- O uso de diversos processadores em um único chip, também chamado de múltiplos *cores* ou **multicore**, proporciona o potencial para **aumentar o desempenho sem aumentar a frequência do clock**.
- A estratégia é **usar dois processadores simplificados** no chip em vez de um processador mais complexo.
- Os chips de dois *cores* foram rapidamente seguidos por chips de quatro *cores*, oito, dezesseis e assim por diante.
- Os fabricantes de chip estão agora no processo de dar um enorme salto, **com mais de 50 cores por chip**.

## Multicore, MICs e GPGPUs

- O salto no desempenho têm levado ao surgimento de um novo termo: **muitos cores integrados** (MIC — do inglês, *Many Integrated Cores*).
- A estratégia do multicore e do MIC envolve uma **coleção homogênea de processadores de uso geral em um único chip**.
- Ao mesmo tempo, os fabricantes de chip estão buscando outra opção de desenvolvimento:
- um chip com múltiplos processadores de uso geral mais **unidades de processamento gráfico** (GPUs — do inglês, *Graphics Processing Units*).



## Multicore, MICs e GPGPUs

- Em termos gerais, **uma GPU é um core desenvolvido para desempenhar operações paralelas em dados gráficos.**
- Tradicionalmente encontrada em uma placa plug-in (adaptador de vídeo), **ela é usada para codificar e renderizar gráficos 2D e 3D, bem como para processar vídeos.**
- Quando uma grande gama de aplicações é suportada por um processador, o termo **GPUs de computação de uso geral (GPGPUs** — em inglês, *General-Purpose computing on GPUs*) é usado.

# Lei de Amdahl

- A **lei de Amdahl** foi proposta primeiro por Gene Amdahl em 1967 (AMDAHL, 1967, 2013) e lida com o potencial *speedup* de um programa usando múltiplos processadores em comparação com um único processador.
- Considere um programa sendo executado em um único processador de modo que uma fração  $(1 - f)$  do tempo de execução envolve o código, que é *inerentemente sequencial*, e uma fração  $f$  que envolve o código que é *infinitamente paralelizável sem sobrecarga no escalonamento*.
- Considere que  $T$  é o tempo de execução total do programa que usa um único processador.

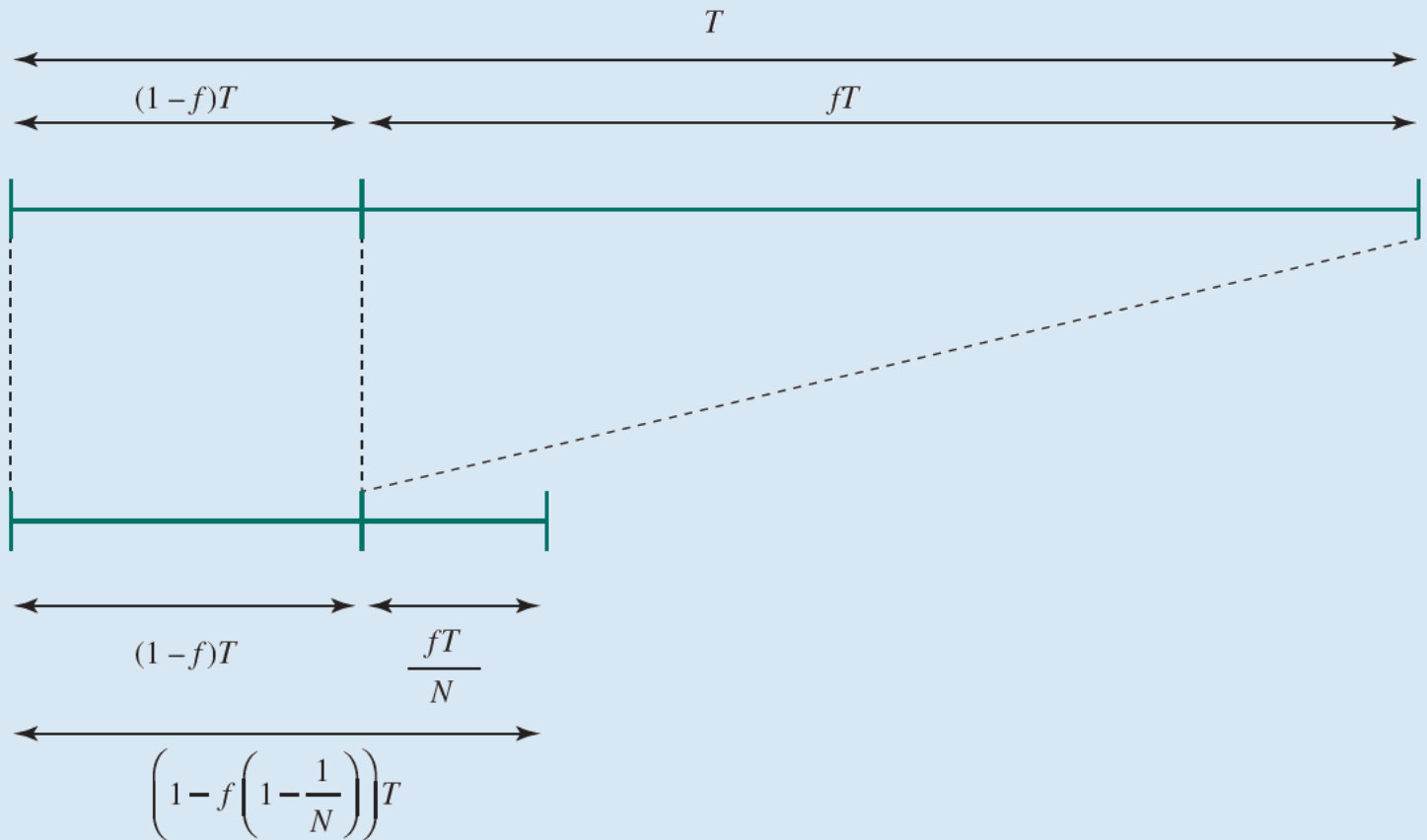
# Lei de Amdahl

- Então, o *speedup* mediante o uso de um processador paralelo com *N* processadores que exploram completamente a parte paralela do programa se dá da seguinte forma:

$$\begin{aligned} \text{Speedup} &= \frac{\text{Tempo para executar o programa em um único processador}}{\text{Tempo para executar o programa em } N \text{ processadores paralelos}} \\ &= \frac{T(1-f) + Tf}{T(1-f) + \frac{Tf}{N}} = \frac{1}{(1-f) + \frac{f}{N}} \end{aligned}$$

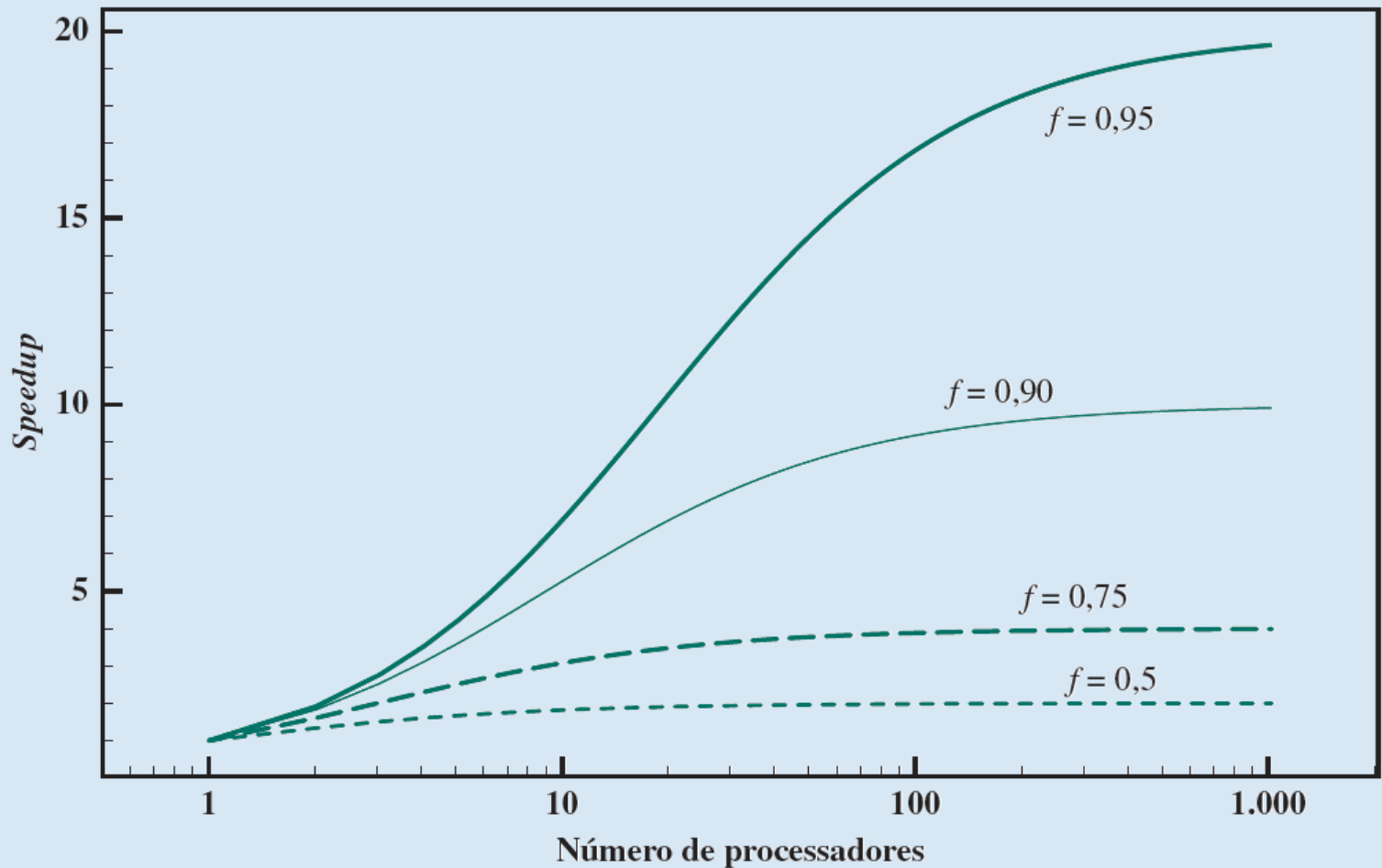
# Lei de Amdahl

- Ilustração da lei de Amdahl:



# Lei de Amdahl

- Lei de Amdahl para multiprocessadores:



# Lei de Amdahl

- A lei de Amdahl pode ser generalizada para avaliar um desenvolvimento ou uma melhoria de técnica em um sistema computacional.
- Considere qualquer aumento em uma característica de um sistema que resulta em um *speedup*.
- O *speedup* pode ser expresso como:

$$Speedup = \frac{\text{Desempenho depois do aumento}}{\text{Desempenho antes do aumento}} = \frac{\text{Tempo de execução antes do aumento}}{\text{Tempo de execução depois do aumento}}$$

# Medidas básicas de desempenho do computador

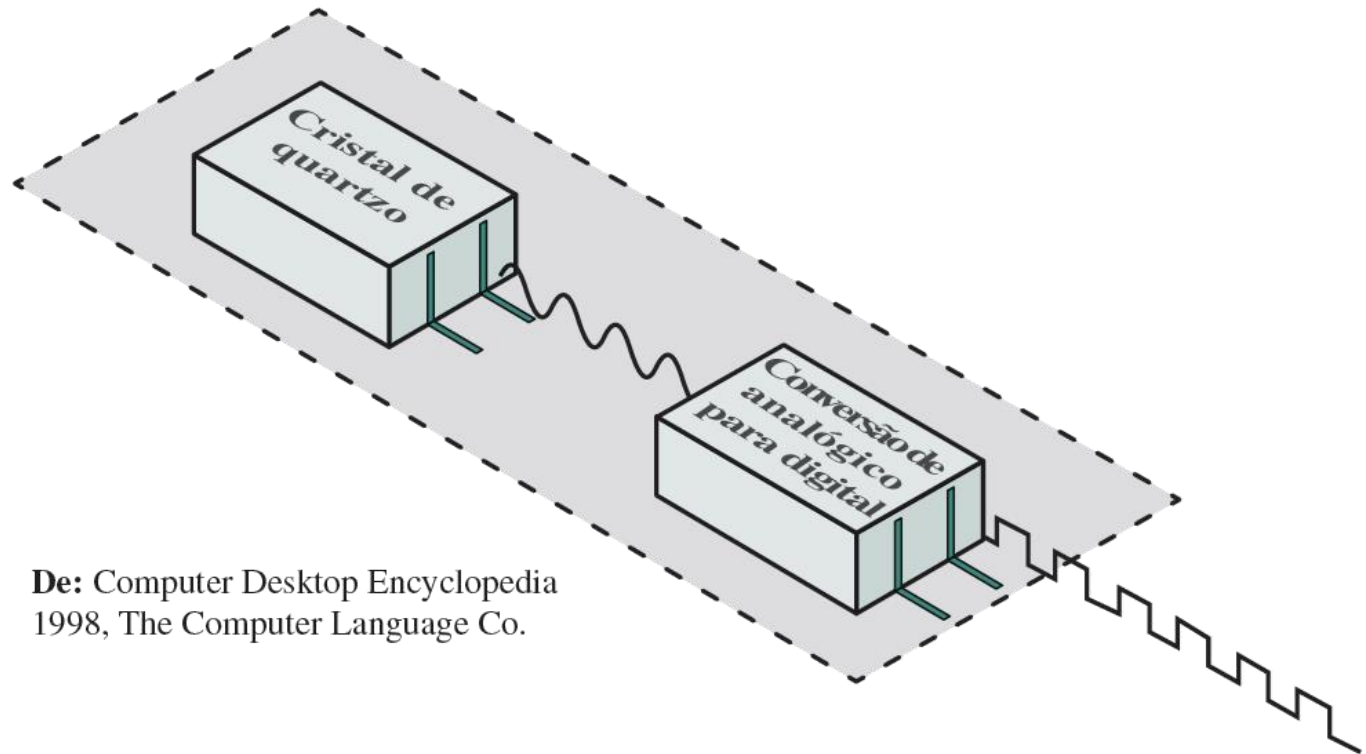
## Velocidade de clock

- A velocidade de um processador é ditada pela frequência de pulso produzida pelo clock, medida em ciclos por segundo, ou Hertz (Hz).
- A taxa de pulsos é conhecida como **frequência do clock** ou **velocidade de clock**.
- Um incremento (ou pulso) do clock é conhecido como um **ciclo de clock** ou um **período do clock**.
- O tempo entre os pulsos é o **tempo de ciclo**.

# Medidas básicas de desempenho do computador

## Velocidade de clock

- Clock de sistema:



De: Computer Desktop Encyclopedia  
1998, The Computer Language Co.



# Medidas básicas de desempenho do computador

## Taxa de execução de instrução

- Um parâmetro importante é a média de **ciclos por instrução** (CPI — do inglês, *Cycles Per Instruction*) para um programa.

- Podemos calcular um CPI geral como a seguir:

$$CPI = \frac{\sum_{i=1}^n (CPI_i \times I_i)}{I_c}$$

- O tempo  $T$  do processador necessário para executar determinado programa pode ser expresso como

$$T = I_c \times CPI \times \tau$$

# Medidas básicas de desempenho do computador

## Taxa de execução de instrução

- Um parâmetro importante é a média de **ciclos por instrução** (CPI — do inglês, *Cycles Per Instruction*) para um programa.

$$CPI = \frac{\sum_{i=1}^n (CPI_i \times I_i)}{I_C}$$

$CPI$  = média de ciclos por instrução

$CPI_i$  = número de ciclos exigidos para instrução do tipo  $i$

$I_i$  = número de instruções executadas do tipo  $i$  para determinado programa

$I_C$  = contagem de instruções para um programa : número de instruções de máquina executadas para esse programa até que ele rode até o fim ou por um intervalo de tempo definido.

# Medidas básicas de desempenho do computador

- O tempo  $T$  do processador necessário para executar determinado programa pode ser expresso como

$$T = I_c \times CPI \times \tau$$

Refinando a formulação acima temos:

$$T = I_c \times [\rho + (m \times K)] \times \tau$$

$m$  = número de referência de memórias necessárias

$K$  = razão entre o tempo de ciclo de memória e o tempo de ciclo do processador

$$\tau = \frac{1}{f}$$

# Medidas básicas de desempenho do computador

## Taxa de execução de instrução

- Fatores de desempenho e atributos de sistema:

	$I_c$	$p$	$m$	$k$	$\tau$
Arquitetura do conjunto de instruções	X	X			
Tecnologia do compilador	X	X	X		
Implementação do processador		X			X
Hierarquia de memória e cache				X	X

- Podemos expressar a taxa MIPS em termos da frequência do clock e do  $CPI$  da seguinte maneira:

$$\text{Taxa MIPS} = \frac{I_c}{T \times 10^6} = \frac{f}{CPI \times 10^6}$$

# Medidas básicas de desempenho do computador

## Taxa de execução de instrução

Exemplo: Considerando a execução de um programa que resulta na execução de 2 milhões de instruções em um processador de 400 MHz. O programa consiste em quatro tipos principais de instruções. A mistura de instruções e o CPI para cada tipo de instrução são mostrados na tabela a seguir:

Mistura de instruções e CPI

Tipo de instrução	CPI	Número de instruções (%)
Aritmética e lógica	1	60%
Load/store com acerto de cache	2	18%
Desvio	4	12%
Referência de memória com falha de cache	8	10%

$$CPI = 0,6 + (2 \times 0,18) + (4 \times 0,12) + (8 \times 0,1) = 2,24$$

$$Taxa\ MIPS = \frac{400 \times 10^6}{2,24 \times 10^6} \approx 178$$

# Medidas básicas de desempenho do computador

## Taxa de execução de instrução

- Milhões de instruções de ponto flutuante por segundo (MFLOPS).

$$\textit{Taxa MFLOPS} = \frac{\text{número de operações de ponto flutuante executadas em um programa}}{\text{tempo de execução} \times 10^6}$$

Medida comum em muitas aplicações científicas e de jogos.

# Medidas básicas de desempenho do computador

- Dado um conjunto de números reais  $n$  ( $x_1, x_2, \dots, x_n$ ), as três médias são definidas da seguinte maneira:

## Média aritmética

$$MA = \frac{x_1 + \dots + x_n}{n} = \frac{1}{n} \sum_{i=1}^n x_i$$

## Média geométrica

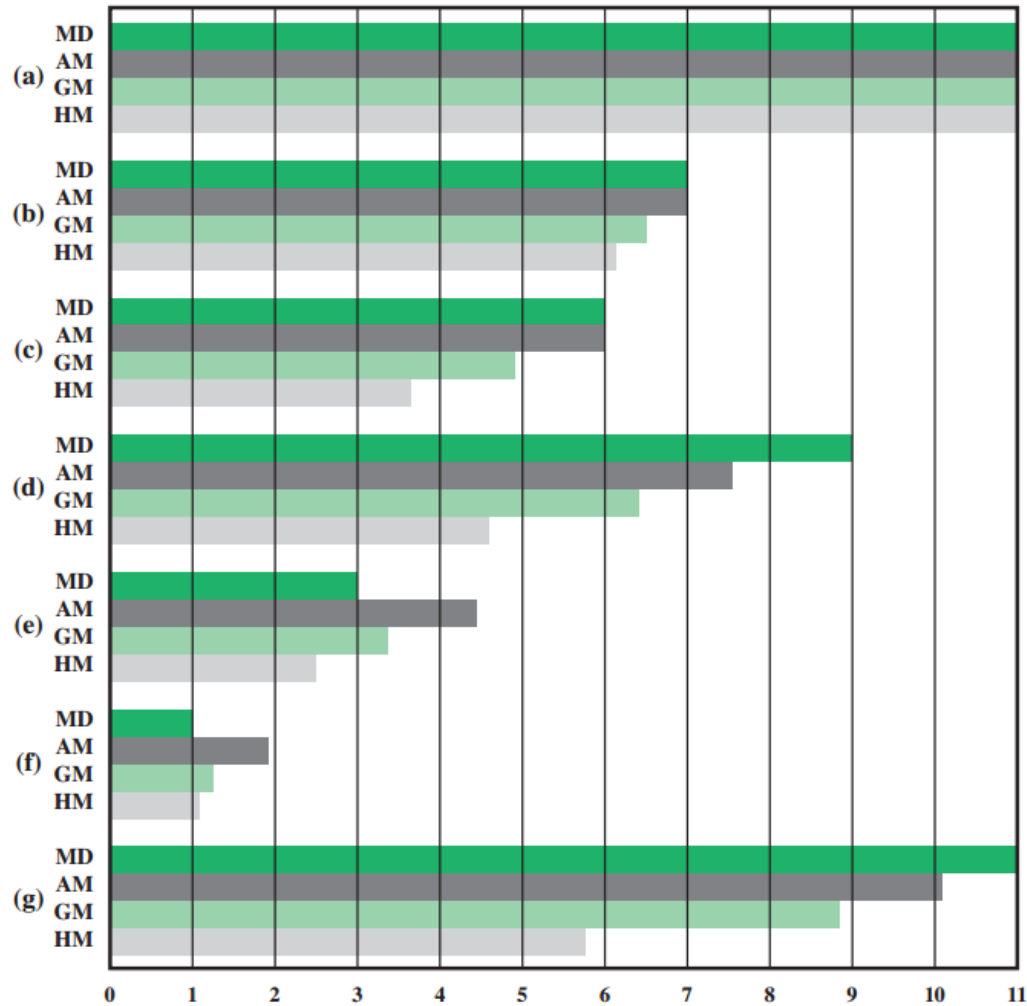
$$MG = \sqrt[n]{x_1 \times \dots \times x_n} = \left( \prod_{i=1}^n x_i \right)^{1/n} = e^{\left( \frac{1}{n} \sum_{i=1}^n \ln(x_i) \right)}$$

## Média harmônica

$$MH = \frac{n}{\left( \frac{1}{x_1} \right) + \dots + \left( \frac{1}{x_n} \right)} = \frac{n}{\sum_{i=1}^n \left( \frac{1}{x_i} \right)} \quad x_i > 0$$

# Medidas básicas de desempenho do computador

## ➤ Cálculo da média



(a) Constant (11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11)

(b) Clustered around a central value (3, 5, 6, 6, 7, 7, 7, 8, 8, 9, 11)

(c) Uniform distribution (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11)

(d) Large-number bias (1, 4, 4, 7, 7, 9, 9, 10, 10, 11, 11)

(e) Small-number bias (1, 1, 2, 2, 3, 3, 5, 8, 8, 11)

(f) Upper outlier (11, 1, 1, 1, 1, 1, 1, 1, 1, 1)

(g) Lower outlier (1, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11)

MD = median

AM = arithmetic mean

GM = geometric mean

HM = harmonic mean



# Média aritmética

- Uma **MA** é uma medida apropriada se a soma de todas as medidas for um valor significativo e interessante.
- A MA de todas as execuções é uma boa medida dos desempenhos de sistemas em simulações e um bom número para uso em comparação entre sistemas.
- A MA usada como uma variável baseada em tempo (por exemplo, segundos), como tempo de execução de programa, tem uma propriedade importante que é diretamente proporcional ao tempo total.
- Se o tempo total dobra, o valor médio segue o mesmo caminho.

## Média harmônica

- Para algumas situações, a taxa de execução de um sistema pode ser vista como uma medida útil do valor do sistema.
- Pode ser a **taxa de execução de instruções**, medida em MIPS ou MFLOPS, ou uma taxa de execução de programa, que mede a taxa na qual um dado tipo de programa pode ser executado.
- Considere como gostaríamos que a média calculada se comportasse.
- Não faz sentido dizer que gostaríamos que a taxa média fosse proporcional à taxa total, onde a taxa total é definida como a soma das taxas individuais.

# Média harmônica

- A soma das taxas pode ser uma estatística sem significado.
- De preferência, gostaríamos que a média fosse inversamente proporcional ao tempo total de execução.
- Vamos considerar um exemplo básico e vamos primeiro examinar como a MA se comporta.
- Suponha que tenhamos um conjunto de  $n$  programas de benchmark e gravamos os tempos de execução de cada programa em um dado sistema como  $t_1, t_2, \dots, t_n$ .

## Média harmônica

- Para simplificar, suponhamos que cada programa execute o mesmo número de operações  $Z$ .
- A taxa de execução para cada programa individual é  $R_i = Z/t_i$ .
- Usamos a MA para calcular a taxa de execução média:

$$MA = \frac{1}{n} \sum_{i=1}^n R_i = \frac{1}{n} \sum_{i=1}^n \frac{Z}{t_i} = \frac{Z}{n} \sum_{i=1}^n \frac{1}{t_i}$$

## Média harmônica

- Vemos que a taxa de execução de MA é proporcional à soma dos inversos dos tempos de execução, que não é a mesma conforme for inversamente proporcional à soma de tempos de execução.
- Assim, a MA não tem a propriedade desejada.
- A MH produz o seguinte resultado::

$$MH = \frac{n}{\sum_{i=1}^n \left( \frac{1}{R_i} \right)} = \frac{n}{\sum_{i=1}^n \left( \frac{1}{Z/t_i} \right)} = \frac{nZ}{\sum_{i=1}^n t_i}$$

## Média geométrica

- Com relação às mudanças nos valores, a **MG** confere peso igual para todos os valores no conjunto de dados.

- Para a MG de uma razão, a MG das razões iguala a razão das MGs:

$$MG = \left( \prod_{i=1}^n \frac{Z_i}{t_i} \right)^{1/n} = \frac{\left( \prod_{i=1}^n Z_i \right)^{1/n}}{\left( \prod_{i=1}^n t_i \right)^{1/n}}$$

- Pode haver casos em que a MA de um conjunto de dados é maior que aquela do outro conjunto, mas a MG é menor.

# Média geométrica

- Uma propriedade da MG que tem tido apelo na análise de benchmark é que ela proporciona resultados consistentes quando mede o desempenho relativo das máquinas.
- De fato isso é para o que os benchmarks são usados em primeiro lugar.
- Os resultados, como temos visto, são expressos em termos de valores que são normalizados para a máquina de referência.
- É seguro dizer que nenhum número único pode proporcionar todas as informações necessárias para comparar resultados entre sistemas.

## Benchmarks e SPEC

- Os **benchmarks** proporcionam orientações para os clientes que tentam decidir qual sistema comprar.
- Pode ser útil para vendedores e desenvolvedores na determinação de como desenvolver sistemas para atingir as metas de benchmark.
- Weicker (1990) lista as características desejadas de um programa de benchmark:
  1. É escrito em uma linguagem de alto nível, tornando-o portátil entre diferentes máquinas.



## Benchmarks e SPEC

2. Representa um tipo particular de estilo de programação, como programação de sistemas, programação numérica ou programação comercial.
  3. Pode ser medido com facilidade.
  4. Tem ampla distribuição.
- Um pacote de benchmark é uma coleção de programas, definidos em uma linguagem de alto nível, que, juntos, tentam oferecer um teste representativo de um computador em determinada área de aplicação ou de programação de sistema.

## Benchmarks e SPEC

- O mais conhecido conjunto de pacotes de benchmark é definido e mantido pela **Standard Performance Evaluation Corporation (SPEC)**, um consórcio da indústria.
- O pacote de benchmark mais conhecido da SPEC é o **SPEC CPU2006**.
- Trata-se do pacote padrão da indústria para aplicações com uso intensivo do processador.
- Ou seja, o SPEC CPU2006 é apropriado para medir o desempenho de aplicações que gastam a maior parte de seu tempo realizando cálculos, em vez de E/S.

# Benchmarks e SPEC

Outros pacotes SPEC são os seguintes:

- **SPECviewperf**
- **SPECwpc**
- **SPECjvm2008**
- **SPECjbb2013 (Java Business Benchmark)**
- **SPECsfs2008**
- **SPECvirt\_sc2013**

<http://www.spec.org/benchmarks.html>

## Benchmarks e SPEC

Para entender melhor os resultados publicados de um sistema usando CPU2006, definimos os seguintes termos usados na documentação da SPEC:

- **Benchmark:** um programa escrito em uma linguagem de alto nível que pode ser compilado e executado em qualquer computador que implemente o compilador.
- **Sistema em teste:** é o sistema a ser avaliado.
- **Máquina de referência:** cada benchmark é executado e medido em sua máquina para estabelecer o tempo de referência para tal benchmark.

## Benchmarks e SPEC

- **Métrica de base:** o compilador padrão com mais ou menos configurações padrão deve ser usado em cada sistema em teste para atingir resultados comparativos.
- **Métrica de pico:** possibilita aos usuários tentar otimizar o desempenho do sistema ao otimizar a saída do compilador.
- **Métrica de velocidade:** é simplesmente uma medida do tempo que leva para a execução de um benchmark compilado.
- **Métrica de taxa:** é uma medida de quantas tarefas um computador pode cumprir em certa quantidade de tempo.

## Exercícios

1. Um programa de benchmark é executado em um processador de 40 MHz. O programa executado consiste em 100.000 execuções de instrução, com a seguinte mistura de instruções e quantidade de ciclos de clock:

Tipo de Instrução	Quantidade de instrução	Ciclos por instrução
Aritmética de inteiros	45000	1
Transferência de dados	32000	2
Ponto flutuante	15000	2
Transferência de controle	8000	2

Determine o CPI efetivo, a taxa MIPS e o tempo de execução para esse programa.

## Exercícios

2. Considere duas máquinas diferentes, com dois conjuntos de instruções diferentes, ambos tendo uma taxa de clock de 200 MHz. As medições a seguir são registradas nas duas máquinas rodando determinado conjunto de programas de benchmark:

Tipo de Instrução	Quantidade de instruções (milhões)	Ciclos por instrução
Máquina A		
Aritmética e lógica	8	1
Load e store	4	3
Desvio	2	4
Outros	4	3
Máquina B		
Aritmética e lógica	10	1
Load e store	8	2
Desvio	2	4
Outros	4	3

- Determine o CPI efetivo, a taxa MIPS e o tempo de execução para cada máquina.
- Comente os resultados.

## Exercícios

3. Quatro programas de benchmark são executados em três computadores com os seguintes resultados:

	Computador A	Computador B	Computador C
Programa 1	1	10	20
Programa 2	1.000	100	20
Programa 3	500	1.000	50
Programa 4	100	800	100

A tabela mostra o tempo de execução em segundos, com 100.000.000 de instruções executadas em cada um dos quatro programas. Calcule os valores em MIPS para cada computador para cada programa. Depois, calcule as médias aritmética e harmônica considerando pesos iguais para os quatro programas e classifique os computadores com base na média aritmética e na média harmônica.



# Exercícios

4. A tabela a seguir baseada em dados relatados na literatura (Heath, 1984), mostra os tempos de execução, em segundos, para cinco diferentes programas de benchmark em três máquinas.

Benchmark	Processador		
	R	M	Z
E	417	244	134
F	83	70	70
H	66	153	135
I	39.449	35.527	66.000
K	772	368	369

- Calcule a métrica de velocidade para cada processador para cada benchmark, normalizada para a máquina R. Ou seja, os valores de razão para R são todos iguais a 1,0. Outras razões são calculadas pela equação da média geométrica, com R tratado como o sistema de referência. Depois, calcule o valor da média aritmética para cada sistema usando a equação de cálculo da média aritmética. Essa foi a técnica utilizada por Heath (1984).
- Repita a parte (a) usando M como máquina de referência. Esse cálculo não foi tentado por Heath (1984).
- Qual máquina é a mais lenta, com base em cada um dos dois cálculos anteriores?
- Repita os cálculos das partes (a) e (b) usando a média geométrica. Qual máquina é a mais lenta, com base nos dois cálculos?

# Exercícios

5. Considere o exemplo do exercício 1 para o cálculo da taxa média CPI e MIPS, que produziram o resultado  $CPI = 2,24$  e  $MIPS = 178$ . Agora, suponha que o programa possa ser executado em oito tarefas paralelas ou threads com aproximadamente o mesmo número de instruções executadas em cada tarefa. A execução é em um sistema com 8 processadores, em que cada processador (cores) tem o mesmo desempenho do único processador usado originalmente. A coordenação e a sincronização entre as partes acrescentam mais 25.000 execuções de instrução a cada tarefa. Considere os mesmos tipos de instruções do exemplo para cada tarefa, mas aumente o CPI para referência à memória com cada miss de cache para 12 ciclos, em virtude da concorrência com a memória.

- a) Determine o CPI médio.
- b) Determine a taxa MIPS correspondente.
- c) Calcule o fator de speedup.
- d) Compare o fator de speedup real com o fator de speedup teórico determinado pela lei de Amdahl.