

**Universidade de São Paulo**  
**Instituto de Ciências Matemáticas e de Computação**  
**Departamento de Sistemas de Computação**

# **Avaliação de Desempenho de Sistemas Computacionais**

## **Aula 3**

**Marcos José Santana**  
**Regina Helena Carlucci Santana**

# Conteúdo

1. Planejamento de Experimentos 😊



2. Técnicas para Avaliação de Desempenho

3. Análise de Resultados

# Conteúdo

1. Planejamento de Experimentos 😊

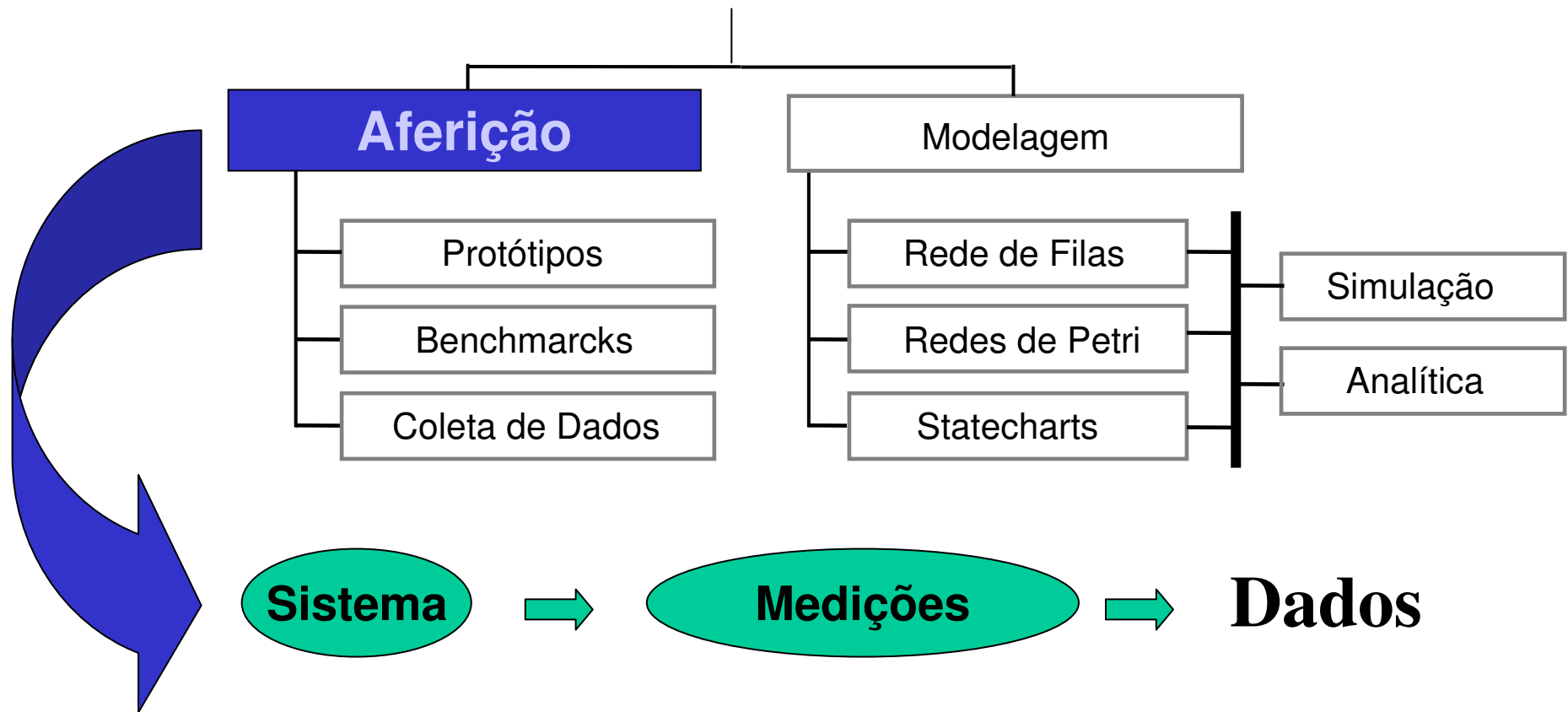
2. Técnicas para Avaliação de Desempenho 😊

- Apresentação das técnicas
- Técnicas de Aferição:
  - Protótipos, Benchmarks e Monitores
- Técnicas de Modelagem:
  - Solução Analítica e por Simulação
- Exemplos

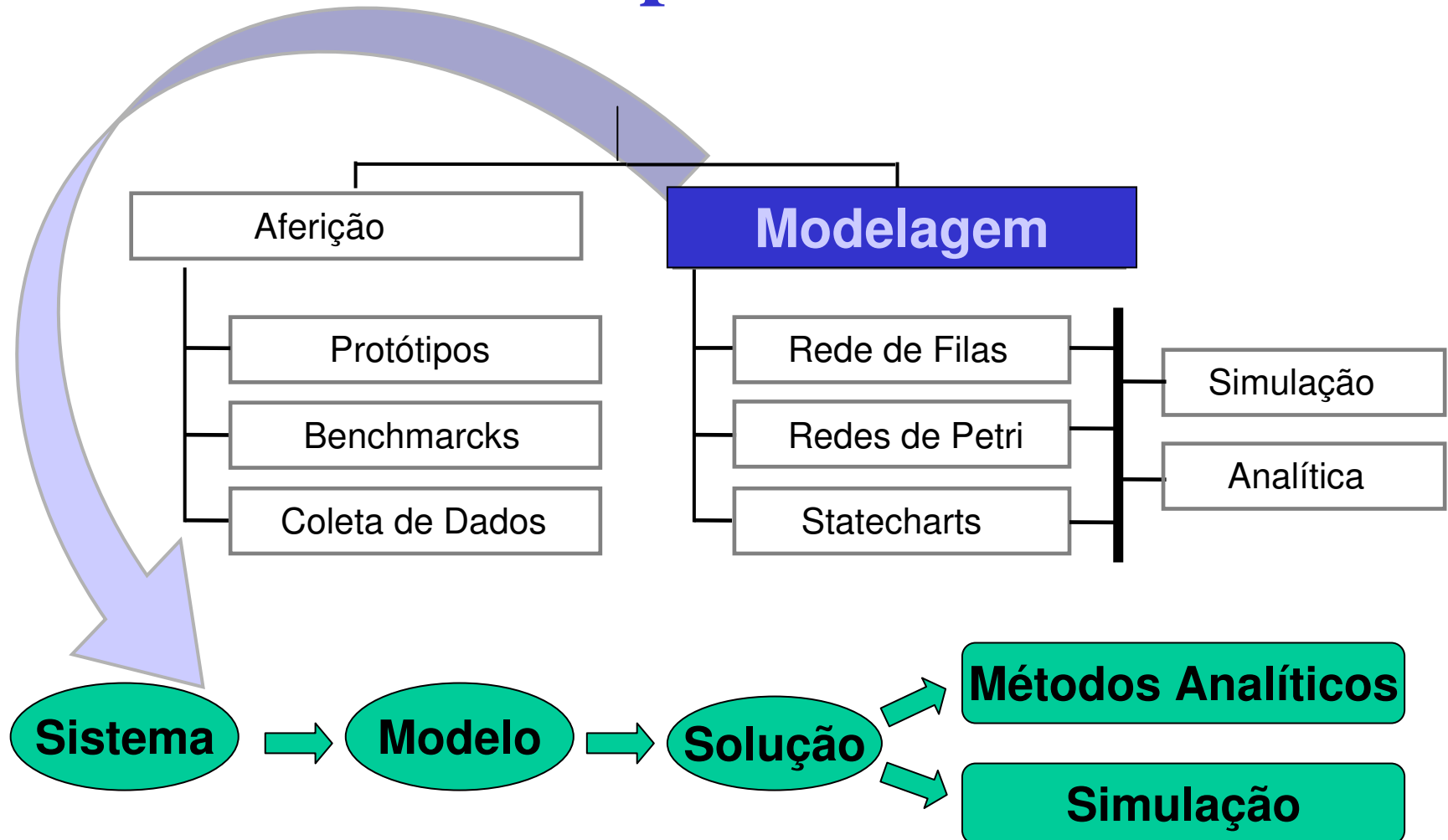
3. Análise de Resultados



# Técnicas de Avaliação de Desempenho



# Técnicas de Avaliação de Desempenho



# Técnicas de Avaliação de Desempenho

## Aferição

- Medidas no próprio sistema
- Sistema deve existir e estar disponível
  
- Experimentação restrita
- Muito cuidado com aquisição dos dados

# Técnicas de Avaliação de Desempenho


## Modelagem

- Desenvolvimento de um modelo
- Não é necessário ter o sistema disponível
- Grande flexibilidade
  
- Resultados estocásticos
- Necessita validar modelo e solução

# Conteúdo

## 1. Planejamento de Experimentos 😊

## 2. Técnicas para Avaliação de Desempenho

- Apresentação das técnicas 😊
- Técnicas de Aferição: 
  - Protótipos, Benchmarks e Monitores
- Técnicas de Modelagem:
  - Solução Analítica e por Simulação
- Exemplos

## 3. Análise de Resultados



# Técnicas de Aferição

- Construção de Protótipos
  - Sistema em Projeto
- Benchmarks
  - Comparação entre Sistemas
  - Avaliar partes específicas de um Sistema
- Monitores ou Coleta de Dados
  - Avaliar um Sistema ou partes dele

# Técnicas de Aferição

- Construção de Protótipos

**Versão simplificada de um sistema computacional que contém apenas características relevantes para a análise do sistema**

# Técnicas de Aferição

## Construção de Protótipos

- uma implementação simplificada do sistema real;
- abstração das características essenciais;
- sistemas em fase de projeto;
- produz resultados com boa precisão;
- recomendado para verificação do projeto final;
- problema: custo e alterações.

# Construção de Protótipos

- 1) Analisar se o sistema é um bom candidato a prototipação
  - Viabilidade da prototipação do sistema;
  - Custo
  - Dificuldades em alterar o protótipo
- 2) Delimitar e conhecer perfeitamente os domínios funcionais e comportamentais do sistema
  - Definir o objetivo da avaliação baseando-se nos objetivos do projeto
  - Abstrair as características essenciais
  - Verificar a possibilidade de obter os dados necessários para a avaliação do protótipo

# Construção de Protótipos

## 3) Desenvolver o protótipo

- Software
- Hardware

## 4) Testar e Validar o protótipo

- Garantir que as simplificações feitas não afetaram a precisão do protótipo

## 5) Coletar e Analisar os dados do protótipo

- Definir a estratégia de coleta de dados no protótipo
- Definir os dados a serem coletados

# Construção de Protótipos

## Concluindo.....

- Ótima opção para verificação de projetos
- Bom para alguns tipos de sistemas
- Custo pode ser um problema
- Flexibilidade não é ponto forte!

# Técnicas de Aferição

- Coleta de Dados - Monitores

**Ferramenta para observar as atividades de um sistema coletando as características relevantes para a análise do sistema**



**Ferramenta =  
Monitor**

# Monitores

## Avaliar o Desempenho e Identificar Pontos Críticos

- Objetivos:
  - Determinar partes mais utilizadas
  - Determinar gargalos
  - Ajustar Parâmetros
  - Caracterizar Carga de Trabalho
  - Determinar Parâmetros para modelos



# Monitores

- oferece os melhores resultados;
- problema central  $\Rightarrow$  interfere com o sistema e o sistema TEM de existir!
- Dois tipos básicos de abordagens:
  - Monitores de Software e de Hardware.

# Monitores

## Formas de Implementação

Define o nível em que o monitor será implementado

1. Hardware
2. Software

# Monitores

## Forma de Implementação

### Hardware

- monitor de hardware que é conectado com o sistema (observador silencioso)
- não interfere no funcionamento normal do sistema medido
- captura eventos rápidos
- apresenta dificuldades em fazer medidas em nível de software
- técnica cara

# Monitores

## Forma de Implementação

### Software

#### Vantagens:

- generalidade
- flexibilidade
- para medidas em nível de programas
- clock virtual

#### Desvantagens:

- ele pode interferir com o normal funcionamento do sistema
- não captura eventos que ocorrem rapidamente

# Monitores

## Forma de Implementação - Exemplos

### Software

Rotina inserida nos protocolos de comunicação para medir o tempo gasto em uma transação em arquivos

### Hardware

Hardware adicionado ao sistema para espionar e contabilizar o tempo gasto em uma transação em arquivos

# Comparação entre monitores de Software e Hardware

<b>Critério</b>	<b>Hardware</b>	<b>Software</b>
<b>Domínio</b>	Eventos de Hardware	Eventos de SO e Software
<b>Taxa de Entrada</b>	Alta ( $10^5$ / Seg)	Depende do proc.
<b>Resolução</b>	Nanosegundos	Milisegundos
<b>Conhecimento Necessário</b>	Hardware	Software
<b>Capacidade de Armazenamento</b>	Limitada pelo armazenamento disp.	Limitada pela sobrecarga
<b>Largura de Entrada</b>	Obtém vários dados simultâneos	Único processador – um evento

# Comparação entre monitores de Software e Hardware

<b>Critério</b>	<b>Hardware</b>	<b>Software</b>
<b>Sobrecarga</b>	Nenhuma	Variável - <5%
<b>Portabilidade</b>	Grande	Pequena
<b>Erros</b>	Mais fácil de ocorrer	Raro
<b>Custo</b>	Alto	Baixo
<b>Disponibilidade</b>	Grande— mesmo com crash	Para durante crash
<b>Flexibilidade</b>	Baixa	Alta

# Monitores - Exemplo

## Ganglia

- Monitor para *clusters* e *grids*
- Métricas e forma de coleta configuráveis
- Pode ser baseado em evento ou amostragem
- Em uso por mais de 500 clusters
- Possui um núcleo + ferramentas auxiliares



# Monitores - Ganglia

Núcleo:

- daemon que deve estar em todos os nós do cluster
- responsável por coletar infos dos nós

Ferramentas:

- Gmetric – permite adicionar métricas durante monitoração
- Gmetad – armazenar infos coletadas
- Diversas outras

# Monitores - Ganglia

- Propagação da info coletada é feita por multicast
- Informações enviadas em um documento XML
- Informações armazenadas em um banco de dados
- Utiliza XDR para transporte dos dados



selecionar a métrica e a granularidade de tempo

métricas globais de um cluster

visualização dos estados de cada nó

# Monitores

## Concluindo....

- Podem gerar resultados bastante confiáveis.
- O sistema deve existir e estar disponível.
- Cuidado com a interferência do Monitor nos resultados
- Dois tipos básicos de abordagens:
  - Monitores de Software e de Hardware.

# Técnicas - Aferição

## Concluindo....

- Protótipos
  - Sistema não existe
  - Fase de Projeto
  - Avaliar comportamento ou desempenho
- Monitores
  - Avaliação de sistemas existentes – real ou protótipo

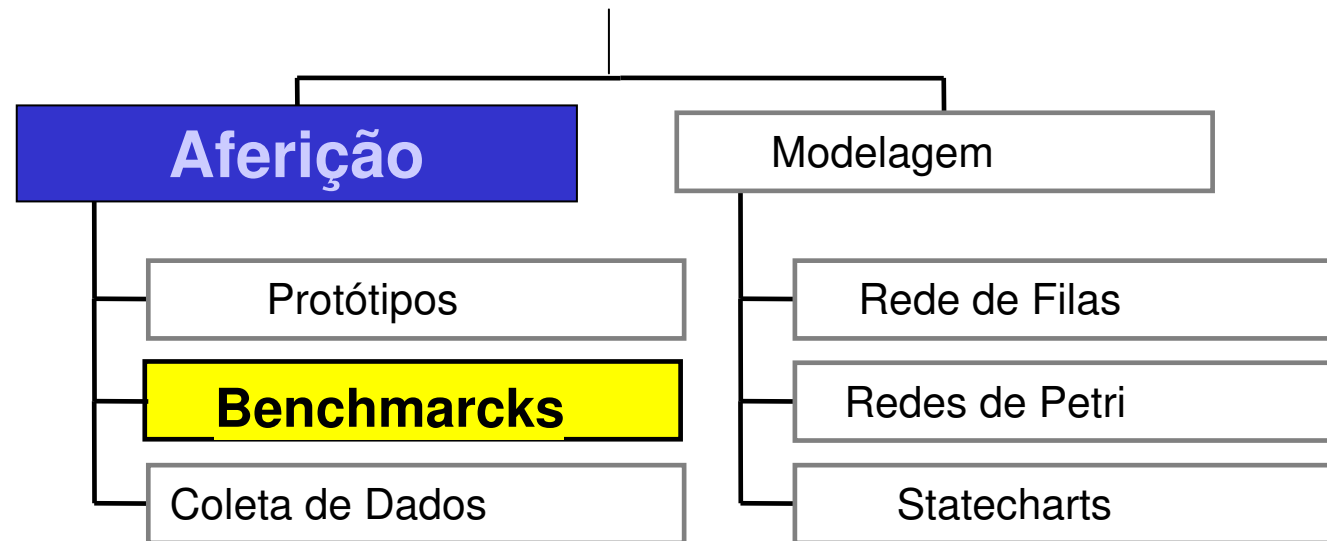
# Técnicas - Aferição

## Problema ....

- Protótipos
- Monitores

**Como comparar com outros sistemas?**

# Técnicas de Avaliação de Desempenho



# Benchmarks



- Instrumento fixo, que permite comparar uma medida (mark - marca) a um padrão preestabelecido
- Deve-se ter um ponto de observação (bench - banco)
- Ponto fixo ou referência para comparações



# Benchmarks

- Empresas
  - Utilizam como modelo
  - Onde elas pretendem chegar
- Ponto fixo ou referência para comparações
- Definir um benchmark para a vida....
- Exemplo:
  - Termômetro

# Benchmarks

$T = 36,5^{\circ}$

Normal



$T = 38^{\circ}$   
Febre!!!

## Termômetro



Vinho



# Técnicas de Aferição

- **Benchmarks - Computação**

**Programa escrito em linguagem de alto nível, representativo de uma classe de aplicações, utilizado para medir o desempenho de um dado sistema ou para comparar diferentes sistemas**

# Benchmarks

- Abordagem muito utilizada para a avaliação de desempenho por aferição
- Exemplo

Qual a diferença entre um i5 e um i7?

Qual a influência no desempenho??

i5	i7
2 ou 4 núcleos	4 ou 6 núcleos
Não possui Hyper-threading	possui Hyper-threading – 2 núcleos lógicos para cada físico
DMI - Direct Media Interface (taxa de transferência ~2Gb/s)	QPI - Quick Path Interconnect (taxa de transferência >4,8Gb/s)
Quantidade de canais para acesso a memória – 2 (acessa 2 pentes ao mesmo tempo)	Quantidade de canais para acesso a memória – 3 (acessa 3 pentes ao mesmo tempo)

# Benchmarks

<http://www.cpubenchmark.net/>

## PassMark Performance Test

Processador	Benchmark	Preço (\$)
<a href="#">Intel Core i7 980X @ 3.33GHz</a>	10336	1000,00
<a href="#">Intel Core i7 975 @ 3.33GHz</a>	7007	994,49
<a href="#">Intel Core i5 760 @ 2.80GHz</a>	4510	205,00
<a href="#">Intel Core i5 680 @ 3.60GHz</a>	3,431	296,66
<a href="#">Intel Core i7 740QM @ 1.73GHz</a>	3521	546,00

# Benchmarks

## – Uso:

- Comparar desempenho de máquinas diferentes
- Reprojeter hardware e software
- Decidir sobre aquisição de sistemas
- Ajudar na otimização de programas
- Previsão de desempenho de aplicações em computadores específicos

# Benchmarks

## Como escolher um *benchmark*?

- Ideal -> aplicação do usuário
- O ideal pode ser inviável quando os sistemas são de propósito geral
- Necessita-se de algo mais amplo e representativo

programa escrito em linguagem de alto nível;  
representativo de alguma categoria de programação;  
que possa ser avaliado facilmente;  
que possua larga distribuição.

# Medidas de Desempenho Frequentemente Utilizadas

- Comum aos outros casos:
  - Tempo de resposta,
  - Utilização,
  - Throughput,
  - Tempo/Tamanho de filas.
- Freqüência de clock - MHZ
  - Pode ser UMA medida
  - Problemas – É necessário considerar:
    - Arquitetura do processador
    - Velocidade e quantidade de memória
    - Disco



# Aspectos Relacionados aos Benchmarks

## Problemas...

- Sistemas com configurações diferentes geram medidas de desempenho diferentes
- Otimização do compilador: influencia diretamente no desempenho medido

# Tipos de Benchmarks

- Benchmarks mais comuns
  - Whetstone, Linpack, Dhrystone
- Outros programas de *Benchmarks*
  - Stanford Small Programs *Benchmark Set*
  - EDN *Benchmarks*
  - Sieve of Eratosthenes
  - Livermore Fortran Kernels
  - Perfect Club *Benchmarks*
  - SPEC *Benchmarks*
  - EuroBen *Benchmarks*

# Tipos de Benchmarks

- Whetstone
  - Primeiro grande programa da literatura escrito para Benchmarking
  - Elaborado para análise de programação numérica de ponto flutuante intensivo
  - Apenas a versão Pascal é oficialmente controlada
  - resultado: número de loops por segundo

# Tipos de Benchmarks

- Características do Whetstone
  - Possui alto percentual de dados e operações de ponto flutuante
  - Alto percentual de tempo de execução é gasto em funções matemáticas
  - Ao invés de variáveis locais, Whetstone utiliza muitos dados globais

# Whetstone

- Ranking das melhores máquinas
- Whetstone 97
- Última atualização – setembro 2006
- MWips, million whetstones instructions per second
- <http://www.cse.clrc.ac.uk/disco/Benchmarks/whetstone.shtml>  
**(Setembro 2006)**
- <http://homepage.virgin.net/roy.longbottom/whetstone%20results.htm>  
**(Dezembro de 2007)**

Rank	Machine	Mflop ratings (VL=1024)			Total CPU (seconds)	MWIPS
		N2	N3	N8		
1	Intel Woodcrest 3.0GHz 4MBL2 DC	1966	4588	2907	3.3	10560
2	Intel Woodcrest 3.0GHz-533 4MBL2 DC	1966	4588	3069	3.3	10451
3	IBM eServer p5 570/1.9	1966	1966	1625	6.2	6219
4	SunFire V20 2.2GHz (EKO)	1311	1298	1481	7.7	4496
5	IBM eServer p5 575/1.5	1966	1529	1315	7.8	4874
6	AMD Opteron852/2600 (EKO 2.2)	1513	1547	1771	8.1	4488
7	HP DL380 Pentium4/3600 (EM64T)	1966	1720	607	8.4	4408
8	Dell PowerEdge 1850/3600 1MBL2	1966	1720	607	8.4	4351
9	Dell PowerEdge 1850/3600 2MBL2	1966	1720	607	8.5	4370
10	AMD Opteron875/2200 DC (EKO 2.0)	1311	1251	1497	8.6	4543

VL = Vector loops

MWIPS = million whetstones instructions per second

N2,N3 e N8 – diferentes instruções de ponto flutuante no loop

# Tipos de Benchmarks

- Linpack
  - Trata-se de um *benchmark* de Kernel, desenvolvido a partir do Pacote Linpack de Rotinas de Álgebra Linear em 1976
  - Foi originalmente escrito e muito utilizado em Fortran, porém possui versão em C
  - Solução de uma matriz 100x100 utilizando decomposição L/U pelo método de Eliminação de Gauss (Linpack100)
  - Resultado: MFLOPS

# Tipos de Benchmarks

- Características do Linpack
  - Por ser um benchmark numérico, possui alto desempenho em operações de ponto flutuante.
  - Resultado é mostrado em Mflops/s
  - Trata-se de um programa pequeno, portanto muito ágil para ser executado
  - Maior capacidade com resolução de matrizes 300x300 e 1000x1000



# Tipos de Benchmarks

- Dhrystone
  - *Benchmark* sintético publicado por seu autor Reinhold Weicker da Siemens Nixdorf em 1984
  - Dhrystone é aplicável em sistemas não numéricos com tipos de dados inteiros, como sistemas operacionais, compiladores, editores de texto, etc
  - Resultado: número de loops por segundo

# Tipos de Benchmarks

- Características do Dhrystone
  - Não analisa operações de ponto flutuante
  - Processadores RISC possuem melhor desempenho que processadores CISC
  - Leva em consideração a localidade dos operadores
  - Para análise de processadores diferentes deve-se utilizar compilações de mesma linguagem para o Dhrystone

# Benchmarks

- **Para sistemas específicos:**
  - Servidores Web
  - Redes
  - HD
  - Servidores de e-mail
  - Virtualização
  - SOA
  - Servidores de arquivos
  - Etc.....

# Concluindo....

Benchmarks podem ser utilizados para verificar diversos tipos de sistemas ...

- Servidores Web,
- Banco de dados,
- Processadores,
- Redes de comunicação

# Concluindo....

sendo utilizados com diferentes objetivos...

- Codificação de vídeo e edição de imagens,
- Jogos,
- Processamento intensivo,
- Processamento de textos, etc.

# Concluindo....

Querendo avaliar diferentes características...

- Produtividade
- Desempenho
- Confiabilidade, etc.

# O importante é...

- Escolher o Benchmark adequado,
- Aplicar o Benchmark de forma adequada,
- Analisar os resultados obtidos com critério.

# Técnicas de Aferição

