UNIVERSIDADE FEDERAL DE PERNAMBUCO

CENTRO DE INFORMÁTICA

CURSO DE GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO

Avaliação e Comparação de Desempenho de Computadores: Metodologia e Estudo de Caso

Trabalho de Graduação

Aluno: Nelson Azoubel Ramos (nar@cin.ufpe.br)

Orientador: Paulo Romero Martins Maciel (prmm@cin.ufpe.br)

Recife, 28 de novembro de 2008

Assinaturas

Este trabalho é resultado do esforço do aluno de graduação em Engenharia da Computação Nelson Azoubel Ramos, sob a orientação do professor Paulo Romero Martins Maciel. Foi conduzido no Centro de Informática da Universidade Federal de Pernambuco. Todos abaixo estão de acordo com o conteúdo deste documento

_____

Paulo Romero Martins Maciel

_____

Nelson Azoubel Ramos

Agradecimentos

Agradeço primeiramente aos meus pais, Carlos e Clarice, a meus irmãos, Bruno e Carla, por me proporcionarem totais condições de chegar a este momento e aos meus avós: Léa ZL, Mendel ZL e Maria do Carmo pelo incondicional apoio.

Ao meu orientador Paulo Maciel pela oportunidade de realizar este trabalho e à Erica Sousa pela importante ajuda.

Agradeço aos amigos que fiz no CIn pelas intermináveis horas de estudo.

À coordenadora do curso, Edna Barros, pela assistência prestada sempre que necessitei.

Ao meu amigo Vitor Braga por emprestar seu notebook para o meu estudo de caso.

Agradeço à Rodolfo, o gerente, e a toda minha equipe de projetão, Smart Solutions.

Aos meus amigos pelo compartilhamento dos momentos bons e ruins ao longo da vida acadêmica.

Agradeço à Conceição Bezerra e a toda equipe do DEF do projeto e-fisco: Thiago, Silvia, Davison, Sandra e Edmar.

Agradeço a todos, que direta ou indiretamente, contribuíram para minha vida acadêmica.

Resumo

A concorrência no mercado de T.I impulsiona a melhoria das aplicações computacionais, que torna competitivas as soluções desenvolvidas. Este cenário propicia o surgimento de alternativas para solucionar, melhorar e satisfazer os requisitos computacionais dos diversos sistemas. Este trabalho tem como objetivo propor uma metodologia para avaliação de desempenho e comparação de dois computadores. Para validar o procedimento proposto, estudos de caso serão considerados para avaliar e comparar o desempenho de sistemas computacionais baseados na família de processadores X86. Serão adotados Benchmarks e Cargas Particulares durante o estudo de caso.

Palavras Chaves: Desempenho, Metodologia, Benchmarks.

Sumário

Índices de figuras

Índices de tabelas

1. Introdução

A análise de desempenho de sistemas computacionais está se tornando uma área de estudo crescente devido à constante busca pela satisfação dos clientes por parte das empresas que visam a uma contínua evolução dos produtos e serviços oferecidos. Sistemas criados sem a preocupação com o desempenho podem gerar atrasos, o não fornecimento do serviço desejado, falhas, perdas financeiras. Por exemplo, [8]:

- O orçamento planejado para o desenvolvimento do sistema de reservas de lugares do aeroporto de Denver teve um reajuste de mais dois bilhões de dólares devido às características de desempenho errôneas.

- O sistema de informação, desenvolvido pela IBM, para avaliação das competições individuais das Olimpíadas de Atlanta foi testado com 150 usuários e entrou em colapso, pois o número de usuários durante o período operacional ultrapassou a quantidade de mil pessoas.

A análise de desempenho é o conjunto de métodos que possibilita a análise temporal de um sistema, ela entra como uma ferramenta para auxiliar engenheiros e cientistas na busca do melhor desempenho aliado aos menores custos. A avaliação de desempenho de sistemas computacionais surgiu na década de 60 e é uma combinação de medição, interpretação e comunicação. Podemos destacar os seguintes objetivos: comparar alternativas, determinar o impacto de uma nova funcionalidade ou identificar o desempenho relativo entre dois sistemas.

Para estudar um determinado sistema computacional é necessária a existência de alguma carga de trabalho, em execução, para alterar o estado natural do sistema. No presente trabalho utilizaremos Benchmarks que são testes padronizados usados para medir o desempenho de diferentes sistemas em tipos específicos de aplicações. O analista precisa ter em mente o aspecto

do sistema que se deseja avaliar, ou seja, qual o objetivo da análise que está sendo feita para com isso selecionar o Benchmark adequado.

A partir das mudanças no estado do sistema geradas pelas cargas de trabalho, é possível a obtenção de parâmetros que expressem os estados do sistema. O processo de medição desses parâmetros é composto de várias etapas. É preciso a escolha de uma boa métrica para representar o aspecto em estudo. É muito comum a escolha de uma métrica equivocada, portanto o analista precisa ter o conhecimento teórico para saber quando uma determinada métrica é boa e útil para o estudo.

É muito difícil para o analista obter o valor verdadeiro no processo de medição, praticamente impossível, pois, incertezas são associadas aos valores coletados devido às limitações experimentais que são denominadas de erros. Essas incertezas podem alterar os resultados obtidos, logo é essencial um estudo detalhado dos erros para saber como controlá-los.

Com o resultado da medição o analista pode utilizar de modelos estatísticos para auxiliar o estudo do comportamento do sistema, para com isso conseguir prever e comparar, com um determinado grau de segurança, o comportamento dos sistemas computacionais. Os modelos proporcionam a avaliação dos resultados obtidos e dependendo da situação o analista pode sugerir melhorias e modificações nos sistemas para se obter o desempenho desejado, ou até mesmo impedir a continuidade de um projeto por inviabilidade.

Este trabalho proporá uma metodologia para avaliação computacional de desempenho e comparação cujos microprocessadores pertençam a família x86. Será realizada através de um estudo dos conceitos e técnicas de medição para obtenção dos dados corretos e de técnicas estatísticas para análise e comparação. Será uma metodologia que proporcionará ao analista que a utilizar flexibilidade para poder utilizá-la para os mais diversos objetivos.

Devido à crescente concorrência na indústria de microprocessadores, cujo custo é relativamente elevado em comparação com outros dispositivos, o estudo de caso será direcionado à análise de microprocessadores. Utilizaremos Benchmarks de interesse geral para microprocessadores com o intuito de termos bons parâmetros de estudo. A análise dos dados fará uso de ferramentas estatísticas para validarmos os resultados obtidos.

Após o estudo dos resultados obtidos poderemos tirar conclusões sobre as arquiteturas dos processadores e a natureza dos Benchmarks utilizados no estudo de caso.

1.1 – Estrutura do Trabalho

Este trabalho foi estruturado para permitir o perfeito entendimento da metodologia, até para os profissionais sem conhecimento da área de análise de desempenho. Está dividido da seguinte maneira: na próxima Seção apresentamos trabalhos e publicações relacionados, direta ou indiretamente, com nosso trabalho. No capítulo 2, serão comentados os fundamentos da análise estatística que será realizada no estudo de caso proposto. No capítulo 3, apresentaremos a teoria de medições e métricas de desempenho que são importantes para o desenvolvimento da metodologia e entendimento do estudo de caso e Benchmarks que serão as nossas cargas de trabalho. Os capítulos 4 e 5 apresentarão a metodologia proposta e o estudo de caso respectivamente.

1.2 – Trabalhos Relacionados

Em [6], o texto comenta sobre as novas técnicas de benchmarking que estão surgindo para fazer melhor uso das características multicores dos microprocessadores mais novos. Comenta que muitos arquitetos de sistemas embarcados questionam-se da existência de ganho real na migração dos sistemas para microprocessadores multicores. Para se comprovar se é vantagem a adoção de microprocessadores multicores estão surgindo uma nova geração de Benchmarks um pouco diferentes das anteriores, pois os benchmarks antigos somente exercitavam os processadores e não havia um grande relacionamento com fatores externos ao microprocessador. Para ter um bom rendimento, os Benchmarks para microprocessadores multicores devem utilizar o máximo possível a largura da banda de memória. Além da largura de banda de memória, outro critério de novos benchmarks para microprocessadores é a escalabilidade que ele possui, ou seja, a capacidade de aumentar a utilização dos recursos da máquina, não só do processador. O material conclui que as arquiteturas multicores necessitam de uma nova idéia de Benchmarking.

Em [5], o trabalho apresenta um conjunto de Benchmarks para conduzir análises de desempenho em software e hardware com aplicações de ponto flutuante decimal. Dá ênfase em aplicações aritméticas, são flexíveis e modulares para serem usados nas mais variadas soluções. São elas computação monetária e análise financeira. Foram desenvolvidos em C utilizando a biblioteca Decnumber modificada para aperfeiçoar as operações. A partir das modificações realizadas na biblioteca, a maioria dos Benchmarks passou mais de 75% da sua execução em funções decimais de ponto-flutuante e com um rápido suporte de hardware a essas instruções os resultados podem melhorar sensivelmente.

Em [7], o trabalho descreve uma ferramenta para simulação de memória cache, o DCMSim. É destinada a análise e comparação de desempenho de memórias cache de microprocessadores. Pode ser configurado para vários tipos de arquitetura de memória cache. A motivação surgiu da dificuldade na aprendizagem dos conceitos de memória cache pelos meios tradicionais. A

ferramenta oferece ao analista a possibilidade de adequá-la aos mais diferentes cenários reais e foi desenvolvida em Java devido à capacidade multiplataforma proporcionada.

Em [10], a publicação propõe uma metodologia, juntamente, com um framework para avaliação de desempenho de uma aplicação baseada no desempenho de outras aplicações existentes que já possuam um conjunto de Benchmark para a avaliação pretendida. São comparadas as características da aplicação de interesse com às das aplicações que possuem um Benchmark. Baseado nas similariedades pode-se prever o desempenho da aplicação de interesse, o ponto central do estudo foi como diferenças de características podem ser traduzidas em diferenças de desempenho.

2 - Fundamentos

Para um perfeito entendimento do estado real de um sistema, é necessário o conhecimento de alguns conceitos essenciais. Este capítulo abordará esses conceitos que serão utilizados posteriormente.

Na avaliação da qualidade das amostras em estudo existem três critérios inerentes aos dados da medição que, em muitos casos, são interpretados de forma incorreta, são eles[1][8]:

- Exatidão(Accuracy): Diferença entre o valor medido e o valor referência.

- Precisão(Precision): É uma medida da dispersão do conjunto de dados obtidos na medição. Está relacionada à repetitibilidade dos dados.

- Resolução(Resolution): É a menor alteração do dado que pode ser detectada pelo instrumento de medição

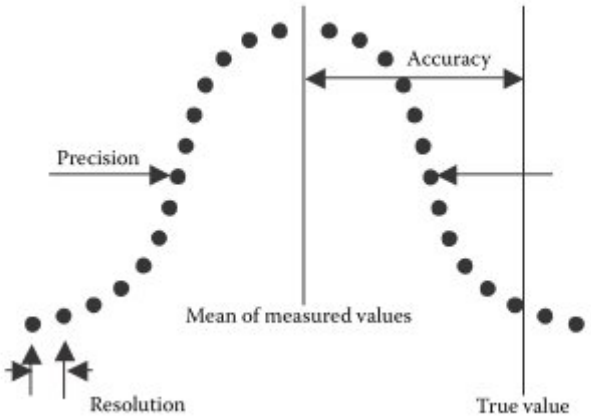


Figura 2.1 – Resolução, exatidão e precisão

Erros, no processo de medição, são tudo que modificam os valores dos dados coletados para estudo. Por exemplo, o compartilhamento do sistema entre diferentes processos que fogem ao controle do analista pode alterar as amostras obtidas. Como conseqüência, várias medições de um mesmo evento podem levar a resultados diferentes[1].

Basicamente, podemos dividir os erros em dois tipos:

- **Sistemático:** É resultado de problemas na medição, como o mau uso do aparelho de medição. Por exemplo, uma mudança na temperatura pode causar uma alteração no período do clock levando a resultados distorcidos. Erros sistemáticos tendem a ser constantes durante as medições e alteram a exatidão dos dados coletados.

- **Aleatório:** Erros aleatórios são erros não determinísticos e incrontroláveis. Afetam a precisão dos dados obtidos e são resultado de limitações da ferramenta de medição, leitura feita pelo observador do experimento ou presença de uma atividade aleatória, não compreendida, no sistema. É de difícil controle, portanto, são usados procedimentos estatísticos para caracterizá-lo e quantificá-lo.

Controlar o experimento que está sendo realizado minimiza o impacto dos erros sistemáticos na exatidão das medições. Erros aleatórios obedecem tipicamente uma curva Gaussiana, ou seja, quando se obtém amostras grandes da população os valores tendem a seguir uma curva de Gauss (curva normal) centrada na média dos valores.

É difícil quantificar a exatidão do processo de medição, pois a exatidão é atribuída ao erro sistemático. É preciso a calibragem do aparelho de medição para um valor padrão e um cuidado durante a medição. Porém, podemos utilizar um intervalo estimado do parâmetro estatístico para quantificar o erro aleatório do processo de medição, intervalo de confiança.

2.1 – Intervalo de confiança

Expressa a idéia de um determinado nível de confiança de que a média dos valores se encontra naquele intervalo. Se a média real estiver fora desse intervalo, as chances de observarmos as amostras que observamos de fato seriam muito pequenas. [1][4]

Dada uma função densidade de probabilidade f(x) e uma variável aleatória X, temos por definição que:

$$F(b) - F(a) = \mathrm{P}(a \leq X \leq b) = \int_a^b f(x)\,dx$$

Suponha que X tenha distribuição normal, logo a f.d.p tem a forma de uma Gaussiana centrada na média, N(μ, σ²), sendo μ a média e σ² a variância(seção 4.2), a f.d.p será da forma:

$$f(x; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

A distribuição normal pode estar na forma normal reduzida se μ = 0 e σ = 1, então efetuando os cálculos, a f.d.p será da forma:

$$f(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right).$$

Fórmula da distribuição de probabilidade normal reduzida

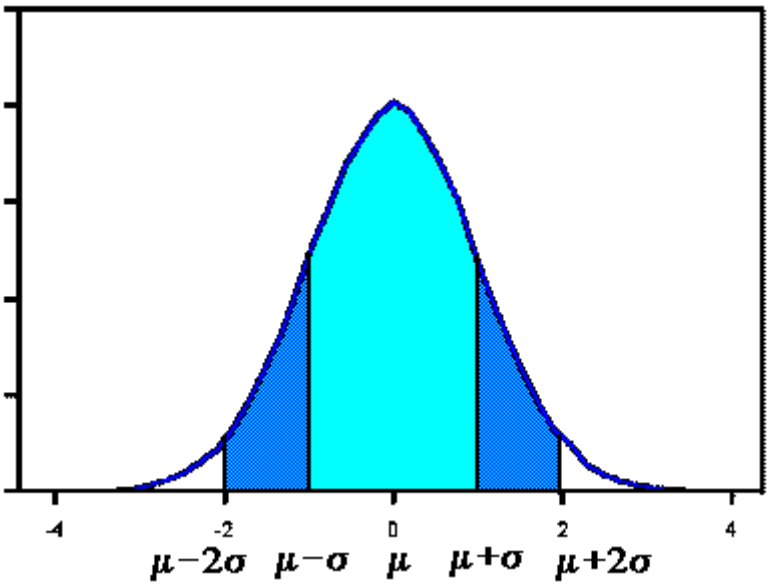


Figura 2.2 - Função distribuição de probabilidade normal reduzida

A fórmula normal padrão não pode ser calculada pelos caminhos comuns. O problema reside no fato de que não podemos aplicar o teorema fundamental do cálculo(Apêndice A). Porém, métodos de integração numérica podem ser empregados para calcular integrais na forma normal reduzida. As integrais da forma reduzida são tabuladas para P($X \leq s$).

Podemos transforma na forma reduzida qualquer distribuição normal e assim utilizarmos a tabulação. Tendo X distribuição $N(\mu, \sigma^2)$, então, fazendo $Y = (X - \mu) / \sigma$ terá distribuição N(0,1). Conseqüentemente:

$$P(a \leq X \leq b) = P((a - \mu) / \sigma \leq Y \leq (b - \mu) / \sigma)$$

$$= \Phi((b - \mu) / \sigma) - \Phi((a - \mu) / \sigma),$$ sendo Φ a fórmula tabelada.

De acordo com o teorema central do limite: qualquer que seja a distribuição de probabilidade da variável aleatória, a distribuição das médias amostrais da população tenderá a uma distribuição normal à medida que o tamanho da amostra cresce, com média μ e desvio padrão $\sigma / n^{1/2}$. O teorema central do limite é muito importante, pois permite utilizar a distribuição normal

para realizar inferências da média amostral seja qual for a forma da distribuição da população.

Tendo X uma distribuição normal qualquer, $\bar{X}$ terá uma distribuição da forma N(μ, σ² / n), sendo n o tamanho amostral. Para obtermos uma média com uma confiança de (1- σ) no resultado, utilizaremos a técnica do intervalo de confiança como mostra a figura abaixo:

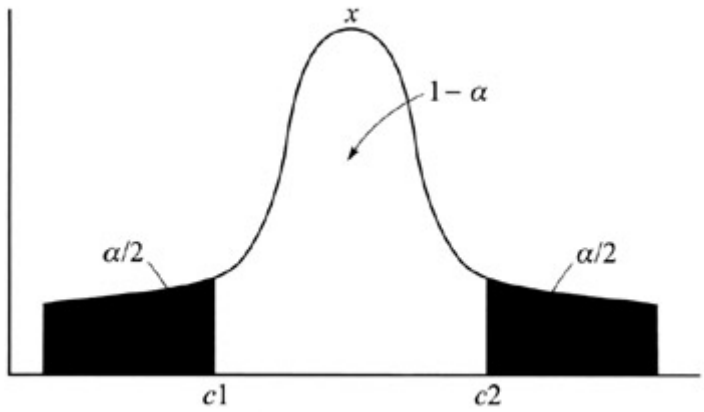


Figura 2.3 - A probabilidade do valor real, x, está no intervalo de confiança (c_1, c_2) é 1 − α

Sabemos que $Z = \frac{\bar{X} - \mu}{\sigma/\sqrt{n}}$ tem distribuição N(0 ; 1). Usaremos este fato da seguinte maneira:

Considere,

$\Phi(z) - \Phi(-z) = P\left(-z \leq \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \leq z \right)$

$\Phi(z) - (1 - \Phi(z)) = P\left(-z \leq \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \leq z \right)$

$2\Phi(z) - 1 = P\left(-z \leq \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \leq z \right)$

$= P\left(\bar{X} - (z\,\sigma)/\sqrt{n} \leq \mu \leq \bar{X} + (z\,\sigma)/\sqrt{n} \right)$

Esta ultima expressão de probabilidade deve ser interpretada de forma cuidadosa. Ela não significa que a probabilidade do parâmetro μ cair dentro de um intervalo seja igual a 2Φ(z) – 1. Ela significa que 2Φ(z) – 1 é a probabilidade de que o intervalo aleatório

($\bar{X}$ - (z σ)/$\sqrt{n}$ ≤ μ ≤ $\bar{X}$ + (z σ)/$\sqrt{n}$) contenha μ.

Exemplo [4]: Suponha-se que X represente a duração da vida de uma peça de equipamento. Admita-se que 100 peças sejam ensaiadas, fornecendo uma duração de vida média $\bar{X}$ = 501,2 horas. Suponha-se que σ seja conhecido e igual a 4 horas, e que se deseje obter um intervalo de confiança de 95% para a média μ, teremos:

$$2\Phi(z) - 1 = P\left(-z \leq \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \leq z \right) = (1 - \sigma) = 0{,}95$$

$$\Phi(z) = (0{,}95 + 1) / 2$$

$$\Phi(z) = 0{,}975$$

Consultando a tabela de distribuição normal reduzida(Apêndice C), chegamos à z = 1,96. O intervalo de confiança será:

I.C = (501,2 – 4/10 * 1,96; 501,2 + 4/10 * 1,96)

I.C = (500,4; 502,0)

Ao afirmar que (500,4; 502,0) constitui um intervalo de confiança de 95% para μ, não quer dizer que 95% das vezes a média amostral cairá nesse intervalo. Estamos afirmando que 95% das vezes a média estará contida no inveralo ($\bar{X}$ - (z σ)/$\sqrt{n}$; $\bar{X}$ + (z σ)/$\sqrt{n}$).

2.2 – Teste T pareado

Este teste é utilizado para verificar se existe alguma diferença estatística entre duas médias. Por exemplo, dada duas médias com seus respectivos intervalos de confiança desejamos saber se elas são equivalentes do ponto de vista de estatístico[1].

Os valores de ambos os conjuntos de medições são relacionados para a formação de pares correspondentes, para isso, as amostras deverão ter o mesmo tamanho. Para verificar se existe uma diferença estatística significante entre médias de dois conjuntos distintos de valores, precisamos calcular o intervalo de confiança da média das diferenças dos pares relacionados. Se este intervalo incluir o '0', concluiremos que não existe uma diferença estatística significante.

Sendo $b_1, b_2, \ldots, b_n$ o cojunto das medições do conjunto B e $a_1, a_2, \ldots, a_n$ o conjunto das medições do conjunto A. Idealizamos um conjunto das respectivas diferenças $D=\{ d_1, d_2, \ldots, d_n \}$ tal que $d_1 = b_1 - a_1$ e assim sucessivamente, ou seja, relacionamos cada par das medições para formar um novo conjunto. Utilizando a mesma derivação da seção anterior chegaremos a um intervalo de confiança (c_1, c_2) para média das diferenças, considerando uma amostra grande com n>30, teremos:

$c_1 = \overline{d} - z_{1-\alpha/2}\frac{s_d}{\sqrt{n}}$ $\qquad$ $c_2 = \overline{d} + z_{1-\alpha/2}\frac{s_d}{\sqrt{n}}$, onde $\overline{d}$ é a média aritmética das diferenças d_i e s_d o correspondente desvio padrão das amostras.

2.3 - Técnica de Bootstrap

É um procedimento utilizado para obtermos aproximações de distribuições amostrais quando a teoria não pode dizer-nos qual a forma de distribuição da população. Aplicada principalmente quando o tamanho da amostra é pequeno (dificuldade ou alto custo para obtenção de amostras maiores). [8]

A idéia básica é que não tendo toda a população para estudo, o ideal é fazer o melhor com a amostra que se dispõe, ou seja, a técnica trata a amostra observada como se ela representasse exatamente toda a população. As reamostragens desta amostra mestre representam o que se deve obter quando se retiram muitas amostras da população original. A distribuição Bootstrap da

estatística, baseada em muitas reamostras, representa uma distribuição amostral desta estatística. Com um número grande de reamostras, acabamos encontrando uma distribuição normal centrada na média das médias da reamostragem, logo podemos aplicar a teoria de intervalo de confiança. Segue abaixo o processo: [8]

1. Selecione uma amostra aleatória de tamanho n.
2. Selecione uma amostra da amostra (re-amostra) de tamanho n com reposição.
3. Calcule a estatística (a média, por exemplo) desta amostra.
4. Repita os passos de 1 a 3, m vezes. (m é grande)
5. Classifique as m estatísticas (médias, por exemplo) em ordem ascendente.
6. Em função do nível de confiança desejado (1 - σ), determine os valores que estejam a ((σ /2)*100%) acima do menor valor e ((σ /2)*100%) abaixo do maior valor.

3 – Medição e métricas de desempenho

Antes de estudar desempenho de sistemas, devemos determinar quais os parâmetros são de interesse para medição. As características básicas de um sistema que se precisa medir estão ligadas à idéia de evento, que é entendido como a mudança no estado de um sistema. Fica a critério do analista a sua definição, tipicamente são[1]: a contagem de quantas vezes um evento ocorre, duração do evento ou o valor atribuído ao evento.

Com a junção das medições acima, podemos derivar uma métrica para o estudo do desempenho de um sistema. Existem várias métricas utilizadas na avaliação de desempenho de sistemas, porém nem todas as métricas são boas o suficiente para evitar erros e conclusões precipitadas. Existe a necessidade de uma boa métrica para avaliação de desempenho. Um critério criado e utilizado por analistas de desempenho classifica uma métrica como ideal quando ela satisfaz as seguintes características:

- ***Linearidade:*** Se o valor da métrica se modificar a uma determinada taxa, o desempenho da máquina deve se modificar à mesma taxa.

- ***Confiança:*** Uma métrica de desempenho é dita confiável se dados dois sistemas A e B, o sistema A tem desempenho superior ao sistema B, logo a métrica terá que indicar o mesmo.

- ***Repetibilidade:*** Uma métrica tem repetibilidade, se o mesmo valor é medido a cada vez que o experimento é executado. Isso significa que uma boa métrica deverá ser determinística.

- ***Fácil medição:*** Se uma métrica é de difícil medição é improvável que desejem utilizá-la em algum estudo.

Várias métricas de desempenho propostas para uso no campo computacional não são adequadas ou são interpretadas de forma incorreta[1].

A taxa de clock, muito utilizada como indicador de desempenho, presume que um microprocessador de 250 MHz é mais rápido na resolução de um problema que um de 200 MHz. Porém, essa avaliação ignora a computação envolvida em cada ciclo de clock, a comunicação entre o processador e a memória, a comunicação entre o processador e os subsistemas de entrada/saída, portanto a taxa de clock, por si só, não nos garante que um sistema tem melhor desempenho.

Mips, milhões de instrução por segundo, é uma métrica que tenta representar uma velocidade. Enquanto no mundo físico uma velocidade é representada pela distância por segundo, Mips define a unidade de distância como sendo a execução de uma instrução. O problema, na utilização do Mips como uma métrica, deve-se ao fato que dependendo da arquitetura, os processadores podem realizar diferentes quantidades de operações em uma mesma instrução. Por exemplo, processadores podem calcular funções senoidais em uma simples instrução, enquanto outros podem requerer várias multiplicações, adições dentre outras.

Fórmula da métrica, n = número de instruções, t_e = tempo total

$$MIPS = \frac{n}{t_e \times 10^6}$$

Freqüentemente, estamos interessados na rapidez que um dado programa é executado, portanto, uma métrica fundamental na análise de desempenho de um determinado sistema é o tempo decorrido na execução de uma rotina. Simplesmente, o sistema que executa uma determinada aplicação em menos tempo terá melhor desempenho. Podemos comparar o tempo diretamente ou utilizar taxas relacionadas. Sem uma medição de tempo, precisa e exata, fica impossível comparar e analisar o desempenho de diferentes sistemas. A técnica padrão para medição de tempo em um sistema computacional é o uso de um “cronômetro” para medir o tempo decorrido para realização de um determinado evento. Em vez de um cronômetro propriamente, um sistema computacional possui um contador que é

incrementado a cada ciclo de clock. O intervalo de tempo é medido subtraindo a contagem inicial da contagem final do contador, portanto teremos o número de ciclos de clock decorridos.

3.1 - Índices de tendência central

O desempenho de um sistema é algo complexo e depende de vários fatores. É, portanto, difícil relacioná-lo com um único valor. Por exemplo, um determinado microprocessador pode ser otimizado para uma classe de aplicações, logo quando forem executadas outras aplicações o desempenho será inferior. Efetuar medições em aplicações que utilizem diferentes tipos de instruções e representá-las por um único número pode levar o analista a conclusões precipitadas[1].

Na tentativa de relacionar um conjunto de medições a um único número, foram criadas relações que indiquem a tendência central das medições, são elas:

- ***Média:*** É a métrica mais comum para representar a tendência central. Importante para o analista o entendimento da definição das diferentes médias, para saber usá-las apropriadamente, são elas:
  1. $\bar{x}_{\mathrm{A}} = \frac{1}{n}\sum_{i=1}^{n} x_i.$ - **Média Aritmética**, é utilizada quando a soma total dos dados tem algum valor físico de interesse. Por exemplo, a soma dos tempos de uma determinada rotina.
  2. $\bar{x}_{\mathrm{H}} = \frac{n}{\sum_{i=1}^{n} 1/x_i}$ - **Média Harmônica** – Usada para taxas de valores inversamente proporcionais como velocidade e tempo. Não é apropriada para dados que se encaixem na média aritmética.
  3. $\bar{x}_{\mathrm{G}} = \sqrt[n]{x_1 x_2 \cdots x_i \cdots x_n} = \left(\prod_{i=1}^{n} x_i\right)^{1/n}.$ - **Média Geométrica** – É utilizada para valores normalizados.

4. $\bar{x}_{\mathrm{H,w}} = \frac{1}{\sum_{i=1}^{n} w_i / x_i}$. - **Média Ponderada** – É um tipo de média aritmética.

- **Mediana*:*** É a medida que separa a metade inferior da amostra, da metade superior. Metade da população terá valores inferiores ou iguais à mediana e a outra metade terá valores superiores ou iguais à mediana.

  Exemplo: Mediana(1, 4, 5, 6,7) = 5

  Mediana(1, 2, 3, 5, 7,8) = (5+3) / 2 = 4

- **Moda*:*** É o valor que surge com mais freqüência nas medições.

  Exemplo: Moda(1, 2, 3, 4, 2, 2,1) = 2

.

3.2 – Variabilidade

Quando representamos uma quantidade de dados por um único valor, omitimos o grau de dispersão dos dados que estão sendo representados. Nesse contexto, entra o conceito de variabilidade. Quando os dados medidos estão muito próximos um dos outros indica uma pequena variabilidade[1][4].

Dado que σ é o desvio padrão da amostra, uma métrica muito utilizada. Temos por definição que a variabilidade é σ ². O desvio padrão é da mesma unidade dos valores medidos, ou seja, se estamos medindo tempo em segundos, o desvio padrão será em segundos também.

Variabilidade de uma amostra: note que nela estão contidas todas as variações dos dados em relação à média

$$\sigma^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}$$

3.3 – Benchmarks

Para medir a velocidade máxima de um carro é preciso que ele esteja em movimento. Do mesmo jeito, para avaliar o desempenho de um sistema computacional é necessário que um programa esteja em execução. A melhor e mais fácil maneira de avaliar o desempenho de uma determinada aplicação é executá-la no sistema de estudo, porém nem sempre é possível, pois o custo para executá-la pode não ser vantajoso. Às vezes, a aplicação não existe e o desenvolvedor precisa saber se é possível desenvolver determinado tipo de aplicação para um sistema[1].

Nesse contexto de dificuldades, surgem os Programas de Benchmarks que são programas usados para medir o desempenho de um sistema computacional. A idéia central é que os Benchmarks possuam características das aplicações que o desenvolvedor planeja executar no sistema computacional, portanto cada Benchmark é criado para simular um determinado tipo de aplicação. O analista utiliza os Benchmarks para obter métricas e com elas avaliar o desempenho do sistema.

3.3.1 – Tipos de Benchmarks

É importante que um Benchmark seja de fácil utilização em diferentes sistemas. Se não for de fácil utilização é bem provável que não seja usado ou utilizado de forma incorreta. Não é uma tarefa fácil a portabilidade dos Benchmarks nos diferentes sistemas. O trabalho do analista se tornou mais fácil com o surgimento de tecnologias portáveis.

Dada a variedade dos domínios de aplicações que possuem diferentes características de execução, existe uma infinidade de Benchmarks que são criados para atender às necessidades dos diferentes domínios e conseqüentemente aos diferentes usuários. Por exemplo, desenvolvedores de um novo sistema computacional, geralmente, são interessados por

Benchmarks que exercitem componentes específicos ao longo da evolução do projeto. Podemos dividir os Benchmarks em vários tipos de acordo com algumas características.

A definição de desempenho é relativa. Uma das primeiras e mais aceitas medidas de desempenho é o tempo gasto para a execução de uma simples instrução. Desde que quase todas as instruções de um computador durassem quase o mesmo tempo para a execução, saber o tempo para execução de uma instrução seria suficiente para determinar o desempenho. Então seria simples, se a instrução escolhida for a adição, a máquina que detiver a adição mais rápida terá melhor desempenho como um todo, tornando o critério extremamente vago.

Com intuito de melhorar o desempenho, os arquitetos começaram a desenvolver instruções de microprocessadores que utilizem o mínimo de instruções de clock possíveis.

Uma instrução de acesso a memória principal utiliza mais ciclos de clock que uma operação aritmética que se desenvolve com os registradores do processador. O fato das instruções não serem homogêneas torna os processadores e os sistemas cada vez mais complexos, conseqüentemente, a avaliação de desempenho a partir de uma única instrução imprecisa.

Dentro desse contexto de dificuldades, Jack C.Gibson propôs, em 1950, a “Gibson instruction mix”, mistura de instruções de Gibson, como uma métrica de desempenho. A idéia chave foi agrupar as instruções em diferentes classes sendo que duas instruções que pertençam a uma mesma classe necessitam do mesmo número de ciclos de clock para serem processadas. O número de instruções de uma classe executadas por uma aplicação é usado para formulação de uma média ponderada. A média ponderada é, portanto, a métrica utilizada para comparação entre dois sistemas. O tempo total requerido por uma aplicação é expresso pela seguinte fórmula:

$$\textit{Tempo de execução} = N \times CPI \times T_{clock}$$

Onde N é o numero total de instruções, CPI, “Cycles Per Instruction”(números de ciclos de clock por instrução), é a média ponderada do numero de ciclos de clock necessários para a execução de todas as instruções do programa e T_{clock} é o tempo de um ciclo de clock. A idéia central da “Gibson instruction mix” está na CPI, note que quando usado como métrica de desempenho um pequeno CPI corresponde a um melhor desempenho.

Por exemplo, dada a tabela abaixo:

Classe de Instrução	**Porcentagem das instruções executadas**	**Números de ciclos de clock utilizados**
1	33.4	2
2	23.2	3
3	18.1	3
4	10.3	4
5	7.8	5
6	7.2	7

Calculando a CPI, temos:

CPI=(2*0.334)+(3*0.232)+(3*0.181)+(4*0.103)+5*(0.078)+7*(0.072)= 3.213

Se o T_{clock} do processador for 8ns, por exemplo, o tempo de execução total será: *Tempo de execução* = 23, 842, 128 × 3.213 × 8 × 10-9 = 0.61 s.

Nota-se, que usando a CPI, a avaliação de desempenho fica prejudicada, pois está fortemente ligada ao conjunto de instruções que está sendo executado no estudo. O valor da CPI para um sistema específico depende, exclusivamente, do conjunto de instruções que está em execução. O número de ciclos de clock por instrução varia de sistema para outro, o que dificulta o uso da métrica. A métrica ignora os efeitos no desempenho ocasionados pelas operações de Entrada/Saída e hierarquia de memória, por exemplo.

Benchmarks Sintéticos

Enquanto a idéia anterior se baseia nas instruções que estão sendo executadas no programa, a idéia do Benchmark Sintético é desenvolver um programa que execute um determinado mix de instruções. Em outras palavras, um Benchmark Sintético é um complemento do ítem anterior.

Um Benchmark sintético é um programa artificial no qual o conjunto de operações que ele executa é selecionado para simular uma determinada classe de aplicações. O objetivo é que sendo o conjunto de instruções, o mesmo, das aplicações reais, o desempenho obtido na execução será um bom parâmetro para a avaliação do desempenho da aplicação real.

Existem alguns problemas na utlização de Bechmarks Sintéticos, os mesmos não conseguem simular o impacto sobre o desempenho com a modificação na ordem das instruções, pois diferentes ordens de instruções podem ocasionar diferentes dependências entre elas. Com o aumento da complexidade das aplicações fica difícil a escolha de um Benchmark específico para ela. Porém, mesmo com os problemas relatados o uso é atrativo, pois ele abstrai muito detalhes das aplicações reais.

3.3.2 - Estratégias de Benchmark

A maioria dos Benchmarks baseia a medida do desempenho no tempo requerido para sua execução. Existem outras estratégias que podem ser empregadas em um Benchmark. Podemos dividir as estratégias em três categorias:

- Medida do tempo requerido para executar computação fixa de instruções.

- Medida da quantidade de computação em um determinado tempo.

- Híbrida, uma mistura das duas estratégias anteriores

Benchmarks de computação fixa

A velocidade é um termo comum para uso quando se avalia desempenho em qualquer área. Na física, velocidade é a distância percorrida por unidade de tempo. Em sistemas computacionais não temos uma distância como na física, mas podemos fazer uma analogia com alguma métrica de interesse. O que desejamos então é definir uma "velocidade" ou taxa para os sistemas computacionais, definida como R = W / T, onde W é a computação efetuada e T o tempo de duração, o valor de W pode ser intríseco ao benchmark. Como a computação pode variar de máquina para outra, se quisermos comparar duas máquinas teremos que calcular a computação em ambas e só então apresentarmos a velocidade relativa 'S'. Abaixo o sistema 1 será S vezes mais rápido que o sistema 2.

Velocidade relativa

$$S = R1 / R2 = (W1/T1) / (W2/T2)$$

$$R1 = S*R2$$

O conceito de computação fixa segue o conceito proposto por Gene Amdahl, conhecido como lei de Amdahl. Segundo a lei: "O maior aumento de desempenho possível introduzindo melhorias numa determinada característica é limitado pela percentagem em que essa característica é utilizada". Para avaliarmos o desempenho de um determinado tipo de aplicação, simulamos as classes de intruções mais comuns.

3.3.3 - Exemplos de Bechnmarks

O Whetstone, foi desenvolvido primeiramente em Algol 60 em 1972 pelo National Physical Laboratory na Inglaterra, faz parte de vários programas de benchmark atuais, cujo resultado indica o número de vezes por segundo em que o processador é capaz de executar a sua rotina. O desempenho do processador neste teste é um bom indicativo do seu desempenho em jogos 3D

e em aplicativos científicos. A unidade de medida do Whetstone é o “Millions of Whetstone Instructions per Second” (MWIPS). Este valor está relacionado à capacidade do processador em executar os seguintes tipos de instruções: ponto flutuante, lógicas(if then else) e matemáticas (seno ,cosseno). As vantagens deste Benchmark são o seu tamanho reduzido e simplicidade no código.

Desenvolvido em 1984 inicialmente em ADA, o benchmark Dhrystone tem por objetivo a avaliação de operações aritméticas simples, operações com string, decisões lógicas, e acessos de memória com intenção de refletir as atividades da CPU nas aplicações de computação de propósito mais geral. Tem como unidade de medida o ”Dhrystone Millions of Instruction per Second” (DMIPS). A principal diferença entre os dois é que o Whetstone tem maior ênfase nas operações numéricas, enquanto o Dhrystone tem sua maior ênfase nas funções de String. Sua aplicação principal é na análise da eficiência de combinações hardware/compilador em máquinas de pequeno e médio porte, devido ao parse que é gerado, internamente, para análise sintática do código.

O Everest Ultimate Edition é um programa propietário, completo, de diagnóstico para o computador. É fabricado pela *Lavalys* e utilizado pela *HP* (Hewlett-Packard) para diagnósticos. Exibe as características do hardware, oferece parâmetros para comparação, gera e exibe relatórios e apresenta alguns aspectos do que está ocorrendo em tempo no computador (como processos em andamento). É um sistema profissional de informações e diagnóstico de hardware e software que oferece uma série de Benchmarks para avaliação de desempenho, dentre os quais a velocidade de leitura na memória principal. Pode ser encontrado juntamente com sua documentação no endereço: http://www.lavalys.com/

4 – Metodologia

Este capítulo tem como objetivo propor uma metodologia para avaliação computacional de desempenho e comparação de computadores. Este processo contemplará a mensuração de dados das plataformas, seleção e definição de métricas de interesse. A metodologia proposta terá flexibilidade para poder ser usada em diferentes plataformas e arquiteturas de computadores e para os mais diversos objetivos. Ficará a critério do analista a escolha.

Como o sistema em estudo já existe, podemos realizar medições, caso o sistema não existisse poderíamos desenvolver modelos teóricos de desempenho, porém estão fora do escopo do trabalho. Muitos fatores precisam ser levados em consideração no planejamento de uma análise de desempenho como: o tempo disponível para estudo, a flexibilidade do sistema, custo do projeto, clareza e o nível de exatidão dos resultados.

Com a utilização de cargas de trabalho, respeitando determinadas precondições, realizaremos coletas de métricas de interesse com o objetivo de obtermos dados para realização de um estudo detalhado. Logo, apontaremos que computador tem melhor desempenho sobre as determinadas condições, com cargas de trabalho utilizadas e o grau de confiança no resultado apresentado. Poderemos propor melhorias ou inviabilizar um projeto a partir da análise dos dados.

A metodologia será desenvolvida seguindo seguintes passos:

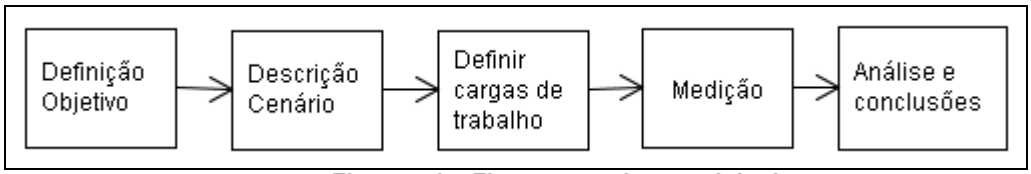


Figura 4.1 – Fluxograma da metodologia

5. 1 – Descrição do cenário

Definido o objetivo(Seção anterior), nessa fase, é preciso o conhecimento do(s) sistema(s) do experimento e entender como estarão configuradas as

máquinas. Cabe ao analista um estudo detalhado do(s) sistema(s) que ele pretende avaliar. Aqui, identificaremos os componentes a serem avaliados e o estado desses componentes, ou seja, precisamos padronizar o ponto de partida da avaliação. Como nossa metodologia é de avaliação e comparação, precisamos proporcionar aos sistemas computacionais condições equivalentes.

O cenário para avaliação serão dois computadores. Os computadores utilizados no experimento em questão deverão ter as configurações analisadas referindo aos seguintes aspectos: sistema operacional, processadores, memória RAM e disco rígido.

Determinaremos precondições para os sistemas em questão. Terão como objetivo proporcionar condições iguais aos computadores e padronizar o estado inicial das medições. Sendo assim, apontaremos que computadores têm melhor desempenho respeitando um determinado conjunto de precondições.

A memória cache dos microprocessadores desempenha importante papel no processamento das máquinas. O uso da cache diminui o overhead sobre os processadores, uma vez que a memória RAM é bem mais lenta que os processadores. Portanto é interessante que os processadores acessem a cache em vez da memória principal.

Eliminaremos todos os processos não vitais às máquinas em questão. O objetivo será dedicar o maior processamento possível da máquina às cargas de trabalho. Será um trabalho manual haja vista a dificuldade de automatizar a funcionalidade.

Desativaremos as placas de rede para desconectarmos as máquinas de redes internas ou externas. Processamento é gasto na implementação dos protocolos de comunicação.

Ambos os sistemas deverão permanecer sobre a mesma temperatura ambiente, pois a temperatura influencia bastante no desempenho dos circuitos integrados.

Ambos os sistemas deverão ficar isolados, pois precisamos evitar interferência de terceiros. O sistema operacional será configurado para não entrar em estado de espera, porque o estado de espera pode interromper a execução dos processos. As precondições, resumidas, serão as seguintes.

Tabela 4.1 – Resumo das precondições estabelecidas para metodologia

Precondição	**Objetivo**
Reiniciar o sistema operacional	Limpar a memória cache do processador
Eliminar todos os processos não vitais à máquina	Muitos processos consomem processamento e podem alterar o resultado da medição
Desconectar a máquina de conexões	Processamento é gasto nas comunicações entre máquinas
Manter a mesma temperatura	A temperatura pode alterar os resultados obtidos
Sistema operacional não deverá entrar em estado de espera	Evitar a interrupção da medição.

4.2 – Cargas de Trabalho

Com o cenário definido, é preciso decidir o rumo da avaliação que está sendo feita. O analista precisa responder a pergunta: “O que queremos avaliar?”. Precisa ficar claro o objetivo do estudo que está sendo realizado, ou seja, que aspecto dos dois sistemas deseja-se comparar. Por exemplo, pode-se querer comparar a capacidade de operações com ponto flutuante dos dois sistemas.

A reposta aos questionamentos anteriores será dada pelo analista com a escolha da carga de trabalho a ser utilizada no experimento. As cargas gerarão, no sistema, os estímulos necessários para criação das métricas de interesse que serão coletadas na fase posterior da metodologia.

Para a carga de trabalho faremos o uso da técnica de Benchmarking que é o processo de comparação de desempenho em diferentes sistemas. Em ambientes computacionais, um Benchmark é tipicamente um software que realiza um conjunto restrito e pré-definido de operações e retorna um resultado em algum formato (uma métrica), que descreve o estado do sistema. (Seção 3.3)

Devido à complexidade do trabalho de Benchmarking, muitos erros são cometidos por parte da equipe de analistas por não conhecimento da métrica utilizada, erros no processo de coleta ou fatores externos podem alterar o resultado normal do Benchmarking.

A portabilidade vem sendo uma tendência natural das tecnologias emergentes. Esta metodologia pretende acompanhá-la. Deixará a critério do analista a utilização de cargas portáveis.

4.3 - Medição

Na fase de medição, o analista precisa ter compreendido o sistema como um todo: cenário, componentes relevantes, cargas de trabalho e objetivo da análise que está sendo efetuada, pois esta é a fase mais importante da metodologia. Erros na medição poderão provocar conclusões equivocadas, consequentemente, resultados catastróficos no futuro.

O processo de medição seguirá um roteiro estabelecido visando a um andamento correto e padronizado. Utilizaremos, durante o processo, o documento de protocolo de medição. (Apêndice E)

Será feita a coleta dos dados para um estudo detalhado e a partir dos resultados teremos como apontar com um determinado grau de segurança, que sistema desempenhará melhor sobre o cenário proposto. Ao final os resultados poderão ser analisados e justificados.

Nosso processo de medição seguirá o seguinte fluxograma:

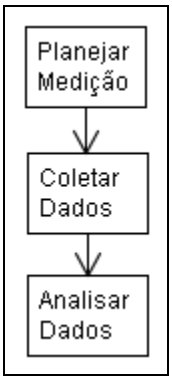


Figura 4.2 – Fluxograma do processo de medição

Planejar Medição

Com o perfeito conhecimento do sistema, seus componentes relevantes, interfaces e critérios de avaliação[8]. O analista pode planejar todo o processo de medição que será realizado. O insumo desta etapa será um documento de medição(Apêndice E) que será utilizado no decorrer das outras etapas do processo de medição. Nesta fase deverão ser designados os analistas responsáveis pela medição e análise dos dados.

Existem diferentes tipos de métricas de desempenho que podemos desejar utilizar. O analista precisa selecionar uma boa métrica que reflita a característica que ele deseja comparar ou avaliar (Seção 3). As diferentes estratégias de medição dos valores estão ligadas a idéia de evento (Seção 3). É necessária a definição da frequência de medição, ou seja, o número de medições em um determinado tempo e a resolução utilizada no experimento(Seção 2).

A classificação do evento a ser medido será uma métrica de contagem. Métricas, nesta categoria, contam o número de vezes que determinado evento ocorre. Por exemplo, uma métrica de contagem de eventos seria o número de faltas de página em um sistema de memória virtual.

Com a classificação do evento definida, podemos definir a estratégia de medição a ser utilizada no experimento. Utilizaremos a estratégia de *Tracing* na carga de trabalho. Além da contagem do número de eventos ocorridos, o *tracing* armazena informações secundárias. Por exemplo, armazenaremos a quantidade de eventos juntamente com o tempo decorrido para o mesmo.

Dentre as várias técnicas para gerar um *tracing*, utilizaremos a modificação do código fonte. É a técnica mais simples e caracteriza-se pela colocação de um *tracing* em determinados locais do código para geração das métricas de interesse. Uma vantagem é que o analista pode escolher, somente, eventos específicos para serem registrados. A desvantagem é que inserir muitos pontos de *tracing* no código pode perturbar o resultado do experimento, pois aumenta o overhead sobre o sistema, portanto utilizaremos apenas um ponto de trace.

Com a definição da estratégia de medição, a carga de trabalho e do tipo de evento a ser avaliado, poderemos planejar como e onde serão coletados os dados para estudo. Por se tratar de uma metodologia de comparação, os dados deverão ser coletados, preferencialmente, paralelamente no mesmo local, pois evitaremos alterações nos dados devido à temperatura.

O ideal é uma coleta feita por meio de um arquivo de texto. A carga de trabalho deverá gravar se possível, em texto, as métricas de interesse, e assim, facilitará o trabalho do analista encarregado da coleta, pois terá, facilmente, todos os dados de forma prática, afastando a possibilidade de erro humano na coleta(Erro sistemático). Serão gravadas somente as métricas de interesse para evitarmos alguma leitura de um dado, de forma errada, na etapa de análise dos dados, pois o analista encarregado da análise pode se confundir na leitura dos mesmos.

Deverá ser realizado um número de medições suficientemente grande, n>>30, nas respectivas máquinas para termos uma boa estimativa da população(Seção 2). Durante o processo de medição, as máquinas permanecerão isoladas para evitarmos interferências externas.

O produto desta fase será um protocolo (Apêndice E) onde o avaliador especificará todos os detalhes da coleta para garantir a representatividade dos dados[8]. O analista encarregado da coleta dos dados terá a responsabilidade sobre o documento, que deve ser conhecido e entendido por todos aqueles que irão utilizá-lo.

Coletar Dados

Com o documento de protocolo, o analista reponsável pela medição dará início ao processo de coleta para obtenção dos dados planejado na fase anterior. Deverá registrar, além dos dados, as condições no momento da medição. O analista deve ter um perfeito conhecimento da metodologia e da ferramenta medição.

Com o final da medição o analista deverá realizar uma aferição sobre os dados coletados. Ele deverá verificar se a resolução, precisão e exatidão foram aceitáveis. (Seção 2) Se não forem, ele deverá tentar ajustar o processo de medição ou até mesmo voltar à etapa de planejamento da medição. O experimento pode ser revisto, pois é melhor corrigir neste momento que deixar para depois, seria bem mais custoso. O analista deverá verificar a existência de *outliers,* dados completamente discrepantes da maioria das medições, e tentar justificá-los.

O produto desta etapa será o mesmo protocolo de medição com todas as informações sobre a coleta, juntamente, com os dados coletados. O analista terá total responsabilidade sobre os dados coletados.

Analisar Dados

Com o documento de medição preenchido e em mãos(Apêndice E), o analista responsável pela análise pode dar início à análise dos dados que é feita interpretando os resultados obtidos. A análise é necessária para o estudo das características de desempenho de um sistema. A partir da interpretação dos resultados, o analista pode inferir que sistema desempenhará melhor sobre o determinado cenário com a carga de trabalho utilizada, podendo então justificar o porquê daqueles resultados foram obtidos.

A metodologia utilizará para estimação e comparação de parâmetros, respectivamente as técnicas estatísticas: Intervalo de confiança e Teste T pareado. (Seção 2). É recomendável a utilização de Softwares estatísticos para auxiliar o analista no desenvolver das técnicas.

Com o intervalo de confiança poderemos estimar, a partir da amostra coletada, a média dos valores com um determinado grau de segurança. Vemos então porque é extremamente importante uma medição bem feita, ela estimará o valor da média, ou seja, será o retrato do sistema.

Com o teste T pareado, podemos verificar se as médias de dois sistemas computacionais, com um determinado grau de segurança, são equivalentes

estatisticamente. Se for comprovada a equivalência, podemos concluir, para as amostras coletadas, a semelhança de desempenho.

Às vezes, a distribuição amostral não é normal, porém precisamos de uma distribuição normal para fazer uso da técnica de Intervalo de confiança e teste T pareado. Para resolver esse problema, a metodologia fará uso da técnica de Bootstrap que a partir de reamostragens tenderá a uma distribuição normal centrada na média que queremos estimar. (Seção 2)

De posse dos resultados do estudo em questão, o analista pode interpretá-los e, então, tomar decisões para melhorar o desempenho de um determinado sistema, modificações em hardware/ software ou impedir o desenvolvimento do sistema por inviabilidade.

É importante salientar que o analista pode necessitar voltar às etapas anteriores da metodologia para reavaliar possíveis decisões que foram tomadas. Por exemplo, o analista pode querer uma nova métrica, modificar o cenário utilizado ou alterar as cargas de trabalhos.

5.0 – Estudo de caso

Este capítulo realizará um estudo de caso com dois computadores portáteis que têm microprocessadores da família x86, seguindo a metodologia apresentada no capítulo anterior (Figura 4.1). Devido à crescente concorrência, o estudo de caso para análise de desempenho dos dois computadores será direcionado para a avaliação dos microprocessadores.

O mercado atual de processadores apresenta-se, basicamente, dividido entre duas grandes empresas do setor: AMD e INTEL, portanto será comparado o desempenho de produtos das respectivas marcas com o intuito de obtermos um parâmetro no que diz respeito ao desempenho de produtos concorrentes de mercado.

A AMD, Advanced Micro Devices**,** é uma empresa americana fundada em 1969, fabricante de circuitos integrados com sede em Sunnyvale, Califórnia. É a segunda maior fabricante de processadores da família x86.

A Intel, Intel Corporation, é uma empresa americana, maior fabricante de circuitos integrados. Foi a criadora da série de processadores da família x86. Com sede em Santa Clara, Califórnia.

Nosso estudo será direcionado à análise de desempenho de processadores da família x86, mais especificamente, trataremos de microprocessadores “Dual Core”(Apêndice D). São microprocessadores com duas unidades de processamento, cores, trabalhando em conjunto. O processador Dual Core foi lançado por uma razão específica, os recursos atuais em busca do maior desempenho estão acabando. O clock dos microprocessadores não pode mais ser aumentado com facilidade e o aumento da memória cache não traz um ganho de desempenho considerável assim como o barramento externo *FSB*.

5.1 – Descrições do cenário

Utilizaremos computadores que possuem os microprocessadores concorrentes de mercado: AMD Turion X2 e o INTEL Core 2 Duo. Ambos dominam o mercado de microprocessadores “Dual Core” para computadores portáteis. Para colocá-los como foco principal de análise serão oferecidos aos microprocessadores recursos computacionais similares: memória principal, sistema operacional e disco rígido.

O AMD Turion 64 X2 é um processador de 64-bits destinado a computadores móveis criado para concorrer com o Core 2 Duo. O Turion foi lançado no dia 17 de maio de 2006.

O INTEL Core 2 Duo foi lançado em 27 de julho de 2006, significou também a reunião das linhas de processadores de mesa e portáteis em um só microprocessador. Apesar de ser o sucessor do Pentium IV, sua arquitetura foi baseada no Pentium III.

O cenário utilizado no experimento em questão deverá satisfazer as precondições da Tabela 5.2 do capítulo anterior. Os computadores têm a seguinte configuração:

- Sistema operacional: Microsoft Windows XP Professional SP2
- Hard Disk: 120 GB com 5400 RPM
- Memória RAM: 2GB, DDR2 667MHZ

Utilizaremos para avaliação os seguintes processadores:

Tabela 5.1 – Especificação do Turion X2

Turion X2 64 bits	
Modelo	TL 58
Barramento FSB	667 MHZ
Cache	1 MB
Clock	1900 MHz por core

Tabela 5.2- Especificação do Core 2 Duo

Core 2 Duo	
Modelo	T 7100
Barramento FSB	667 MHZ
Cache	2 MB
Clock	1800 MHz por core

5.2 – Definições das Cargas de trabalho

Como está definido na metodologia (Seção 4.2), precisamos aferir aqui o objetivo do estudo que está sendo realizado. Realizaremos três análises de diferentes tipos de aplicações. Utilizaremos no experimento em questão os Benchmarks Sintéticos (Seção 3.3): Whetstone na versão em C/C++ e o Dhrystone na versão em Java e o Everest, aplicativo de diagnóstico.

Analisaremos os sistemas com relação a aplicações científicas com o Whetstone. Manipulação de Strings com o Dhrystone, muito importante na realização do parser na análise sintática em compiladores. Em [6], vemos que um aspecto importante para o desempenho em microprocessadores multicores é a capacidade de leitura/escrita na memória principal, logo, por estarmos tratando de processadores multicores, utilizaremos o Everest para avaliar a leitura na memória principal. Com estes três aspectos, poderemos ter uma boa análise por se tratarem de características distintas.

5.3 - Medição

O processo de medição será conduzido conforme o fluxograma Figura 4.2. O analista encarregado do projeto da medição analisou os dois sistemas detalhadamente e as cargas de trabalho utilizadas. Nesta fase, utilizamos o protocolo de medição (Apêndice E).

Planejamento da Medição

Como em 4.3 foi determinado, definimos os dados a serem medidos:

- ***Whetstone***: MWIPS
- ***Dhrystone***: DMIPS
- ***Everest:*** Mega Bytes por segundo

Os dados medidos serão os intrínsecos aos Benchmarks utilizados como carga de trabalho. São métricas, amplamente, aceitas para os tipos de aplicações a que se destinam(Seção 3). Os Benchmarks Dhrystone e Whetstone foram compilados nas respectivas máquinas, modificamos os códigos-fonte para geração do *tracing* desejado. Utilizaremos apenas um *tracing* para diminuir o *overhead* sobre o sistema. O Everest por se tratar de um executável o *tracing* já é interno, portanto não foi possível modificar o código. Como pudemos modificar o código fonte no caso dos Benchmarks Wheststone e Dhrystone, foi possível a escrita da métrica de interesse pelo próprio Benchmark em um arquivo de texto. No caso do Everest, como não pudemos alterar o código fonte, a coleta terá que ser feita manualmente pelo analista responsável pela coleta.

Para obtermos uma boa aproximação utilizaremos a resolução (Seção 2) de três casas decimais. Planejamos realizar um total de 50 medições em cada sistema. Optamos pelas medições sequenciais, um dado após o outro, pois as máquinas não puderam ficar disponíveis por muito tempo.

Foram criados os três protocolos de medição referentes aos três Benchmarks, (Apêndice E) com os dados dos sistemas em análise para uso na fase posterior do processo de medição.

Coleta dos Dados

Foram colocados ambos os computadores sobre o mesmo ambiente refrigerado a 25 graus Celsius, pois alterações de temperatura podem modificar o desempenho dos mesmos. Portanto, todo processo de coleta foi feito paralelamente para os dois sistemas computacionais.

Os pontos de *tracing* gerados foram guardados em um arquivo de texto na raiz do disco rígido, no caso dos Benchmarks whetstone e Dhrystone. Para o Everest foram feitas coletas sequenciais pelo analista, pois se trata de um aplicativo que não oferece o recurso para a gravação em um arquivo de texto. Foram realizadas paralelamente cinquenta medições em sequencia nos sistemas do estudo de caso, porém, devido ao alto custo de medição, fizemos apenas dez medições para o Benchmark Everest.

Na aferição dos dados, a analista não detectou grandes discordâncias nas medições. Os dados foram aceitáveis.

Após o término das medições, o analista preencheu o protocolo de medição com as informações da coleta e com os dados obtidos para análise e estudo na fase posterior da metodologia.

Análise dos dados

Com o protocolo de medição preenchido com as características do sistema e com os dados coletados, o analista responsável pela análise deu início ao processo de análise dos dados. Ao final pudemos avaliar os resultados obtidos e tentar justificá-los.

Utilizamos na análise dos dados a ferramenta estatística Minitab. Minitab que é um programa de computador proprietário voltado para fins estatísticos. É muito utilizado nas universidades nos cursos introdutórios de estatística. Também é utilizado em empresas num nível mais avançado de utilização, tendo funções mais específicas voltadas para gerenciamento. Pode ser encontrado, juntamente com uma documentação própria, no endereço: http://www.minitab.com/. Utilizaremos, na análise, as funcionalidades *"Graphical Summary"* e *"T paired test"* do Minitab, ambas encontram-se nas funcionalidades estatísticas básicas.

O "Graphical Summary" apresenta um resumo estatístico da amostra coletada (Seção 2). Colocamos a amostra coletada em uma das colunas e executamos a

funcionalidade, que além da média e do intervalo de confiança a partir da amostra coletada oferece outras informações importantes para a análise: desvio padrão e gráfico Box-plot. (Apêndice B)

O “T paired test” executa o Teste T pareado, verificará se os dois sistemas são equivalentes estatisticamente (Seção 2). O teste nos dará como resultado a média e o intervalo de confiança da diferença entre as médias. Se o ‘0’ estiver contido no intervalo, podemos concluir com o determinado grau de segurança que para as amostras coletadas, os sistemas são equivalentes em desempenho. (Seção 2)

Faremos uso da técnica de Bootstrap. Utilizaremos a ferramenta estatística Statdisk que pode ser encontrada em: http://www.statdisk.org/. (Seção 2)

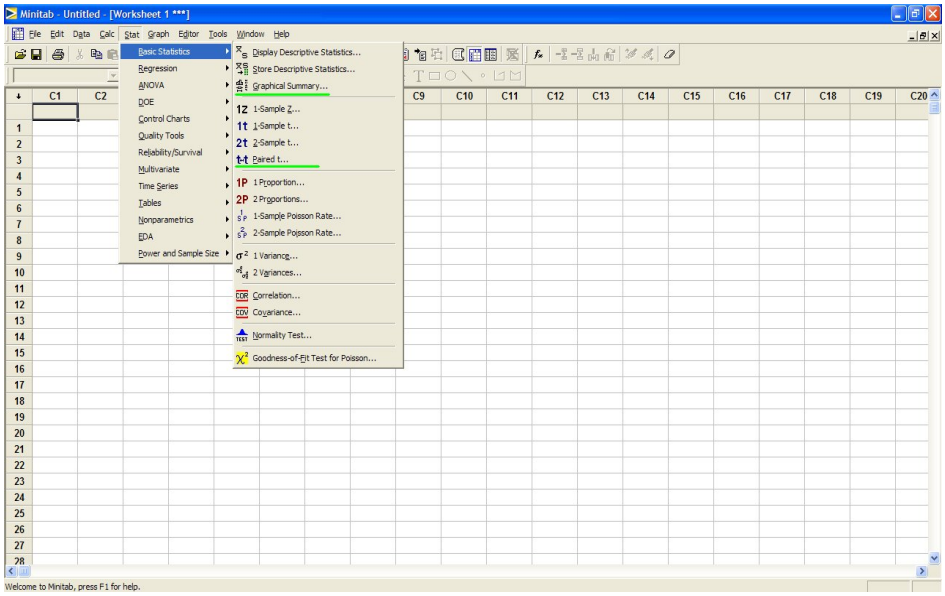


Figura 5.1 – Funcionalidades utilizadas do Minitab

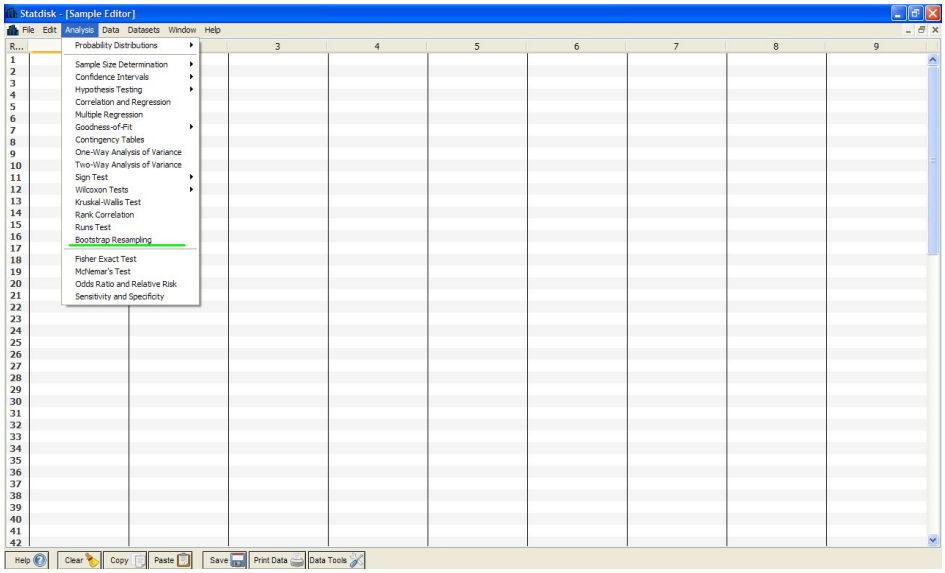


Figura 5.2 – Funcionalidade utilizada do Statdisk

Abaixo, as análises com as respectivas cargas de trabalho com reamostragens nos dados obtidos, técnica de Bootstrap:

Whetstone

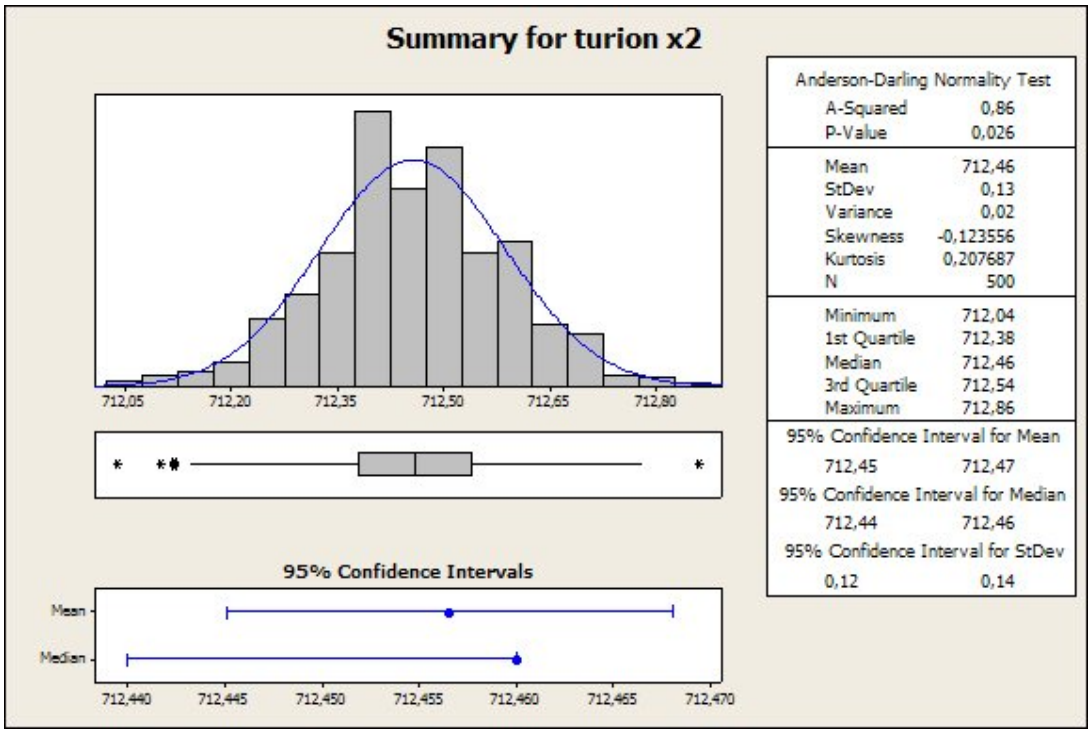


Figura 5.3 – Resumo estatístico do Turion X2 para o Whetstone

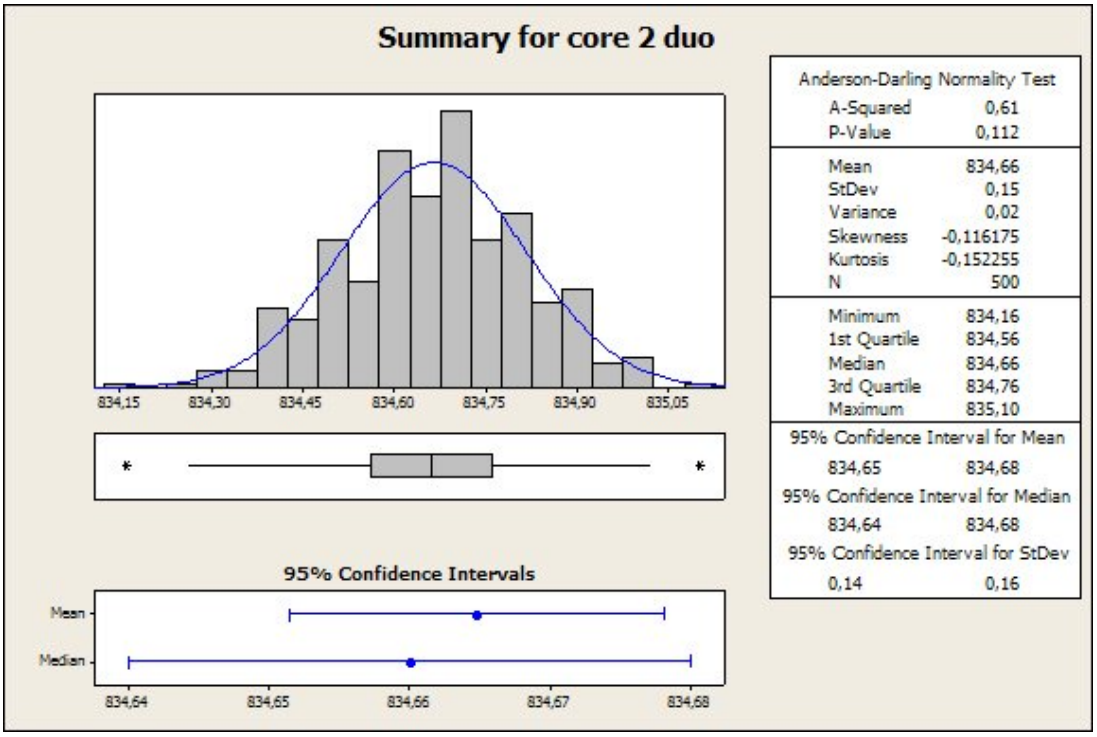


Figura 5.4 - Resumo estatístico do Core 2 duo para o Whetstone

Paired T-Test and CI: turion x2; core 2 duo

```
Paired T for turion x2 - core 2 duo

              N      Mean  StDev  SE Mean
turion x2   500   712,457  0,131    0,006
core 2 duo  500   834,665  0,151    0,007
Difference  500  -122,208  0,201    0,009


95% CI for mean difference: (-122,226; -122,191)
T-Test of mean difference = 0 (vs not = 0): T-Value = -13573,96  P-Value =
     0,000
```

Figura 5.5 – Teste T de paridade do Turion X2 X Core 2 Duo para o Whetstone

Tabela 5.3– Resumo dos valores para o Turion x2 (Apêndice B)

Média	Desvio Padrão	Primeiro Quartil	Mediana	Terceiro Quartil	I.Confiança 95% da média
712,46	0,13	712,38	712,46	712,54	(712,45 ; 712,47)

Tabela 5.4 – Resumo dos valores para o Core 2 Duo (Apêndice B)

Média	Desvio Padrão	Primeiro Quartil	Mediana	Terceiro Quartil	I.Confiança 95% da média
834,66	0,15	834,56	834,66	834,76	(834,65; 835,68)

Analisando as informações da Figura 5.3 para o Turion X2, a média a partir da amostra coletada está contida com 95% de confiança no intervalo(712,45; 712,47) MWPS. Vemos a ocorrência de outliers(Apêndice B), no processo de medição, que podem ter sido resultado de processos que compartilharam o processador com o Benchmark.

No Core 2 Duo, Figura 5.4, vemos um processo mais estável, pois existiram somente dois *outliers*. A média a partir da amostra coletada está contida com 95% de confiança no intervalo (834,65; 835,68) MWPS.

Analisando o "t pair test", Figura 5.5 (Seção 2), é de se notar que '0' não pertence ao intervalo de confiança da média das diferenças. Podemos concluir, a partir das amostras coletadas que os sistemas não são equivalentes estatisticamente, o Core 2 Duo obteve um melhor rendimento no cenário proposto.

O experimento mostrou que provavelmente existiram processos no Turion x2 que utilizaram o processador durante a medição. No Core 2 Duo, aparemente, a precondição de encerramento de processos foi mais bem executada. Podemos inferir que o Core 2 Duo é um microprocessador mais indicado para aplicações científicas ou matemáticas no dado cenário proposto, considerando as amostras coletadas, com 95% de confiança. Exemplo: Matlab, mathcad ou Scilab.

Dhrystone

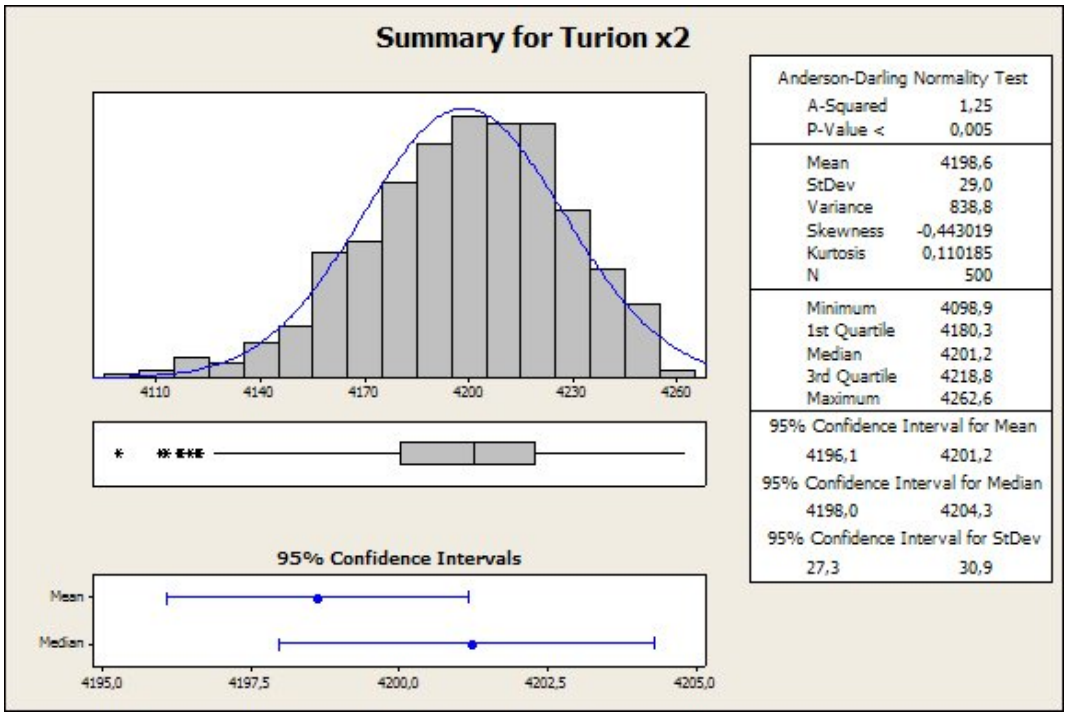


Figura 5.7 - Resumo estatístico do Turion X2 para o Dhrystone

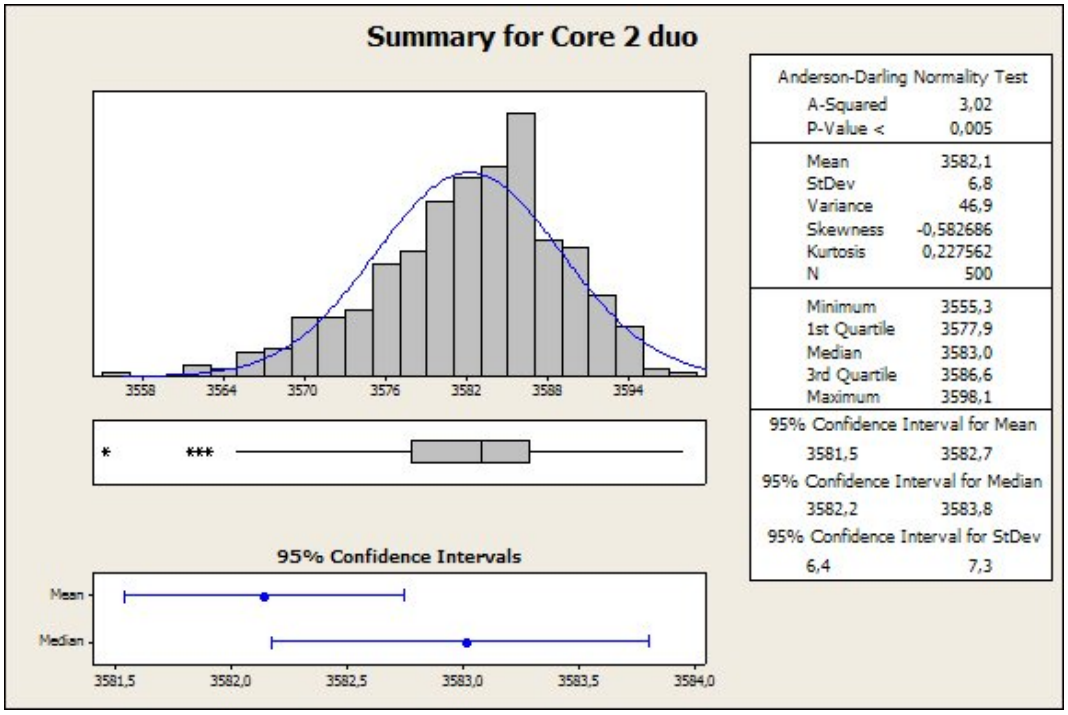


Figura 5.8- Resumo estatístico do Core 2 Duo para o Dhrystone com

Paired T-Test and CI: Core 2 duo; Turion x2

```
Paired T for Core 2 duo - Turion x2

              N     Mean  StDev  SE Mean
Core 2 duo  500  3582,14   6,85     0,31
Turion x2   500  4198,63  28,96     1,30
Difference  500  -616,49  29,89     1,34


95% CI for mean difference: (-619,12; -613,86)
T-Test of mean difference = 0 (vs not = 0): T-Value = -461,17  P-Value = 0,000
```

Figura 5.9 - Teste T de paridade do Turion X2 X Core 2 Duo para o Dhrystone

Tabela 5.5 – Resumo dos valores para o Turion x2 (Apêndice B)

Média	Desvio Padrão	Primeiro Quartil	Mediana	Terceiro Quartil	I.Confiança 95% da média
4198,6	29	4180,3	4201,2	4218,8	(4196,1 ; 4201,2)

Tabela 5.6 – Resumo dos valores para o Core 2 Duo (Apêndice B)

Média	Desvio Padrão	Primeiro Quartil	Mediana	Terceiro Quartil	I.Confiança 95% da média
3582,1	6,8	3577,9	3583	3586,6	(3581,5; 3582,7)

Analisando as informações da Figura 5.7 para o Turion X2, a partir da amostra coletada, a média está contida com 95% de confiança no intervalo(4191,9; 4200,5) DMIPS. Caracterizou-se por ser um procedimento extremamente instável, pois existem muitos *outliers*.

No core 2 Duo, Figura 5.8, vemos também uma grande instabilidade. A partir das amostras coletadas, a média está contida com 95% de confiança no intervalo(3568,6; 3595,9) DMWPS.

Analisando o teste t pareado, Figura 5.9, é de se notar que 0 não pertence ao intervalo de confiança da média das diferenças com 95% de confiança. Podemos

concluir que os sistemas não são equivalentes estatisticamente, o Turion X2 obteve um melhor redimento no cenário que foi proposto.

O analista pode concluir a partir do estudo realizado que o Turion X2 é um processador mais indicado para manipulação de Strings e números inteiros. É comum em compiladores devido ao parser, que é o processo de analisar uma sequência de entrada para determinar sua estrutura gramatical segundo uma determinada gramática formal. Essa análise faz parte de um compilador, junto com a análise léxica e análise semântica. Podemos justificar a grande instabilidade das medições ao uso da tecnologia Java no Benchmark Dhrystone. O Benchmark acaba sendo suportado pela máquina virtual, que é um outro processo em execução, portanto executa rotinas outras rotinas

Everest

Como a utilização da ferramenta Everest tem um alto custo de tempo, fizemos 10 medições e aplicamos a técnica de Bootstrap nos dados para 50 reamostragens. Os resultados seguem abaixo:

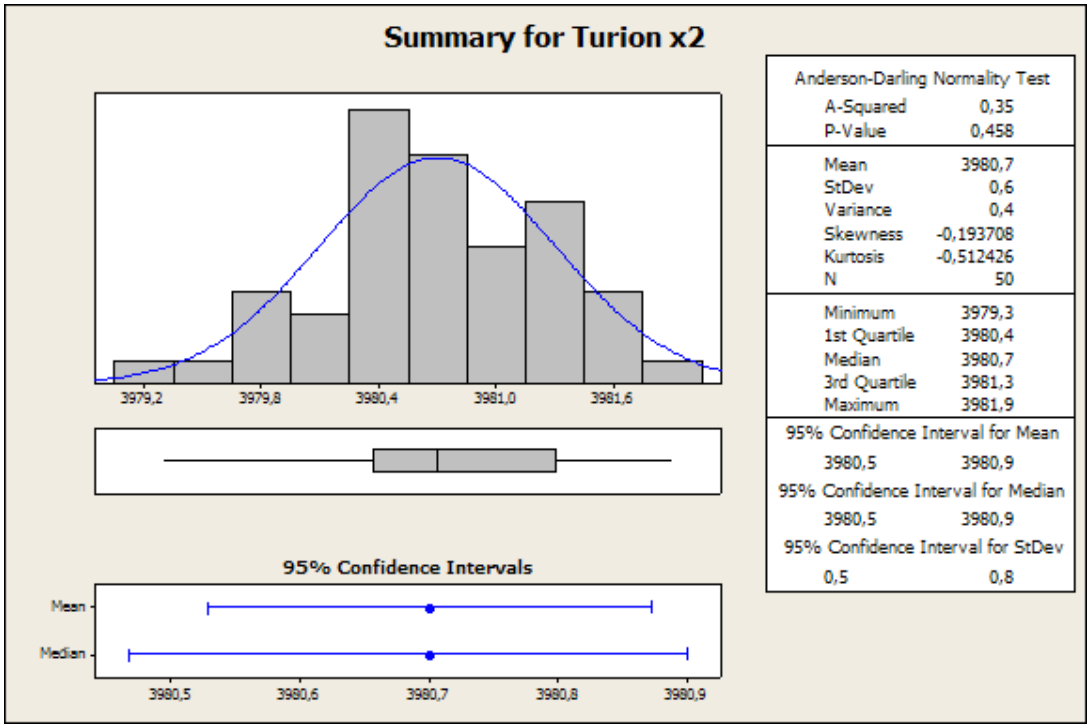


Figura 5.10 - Resumo estatístico do Turion X2 para o Everest

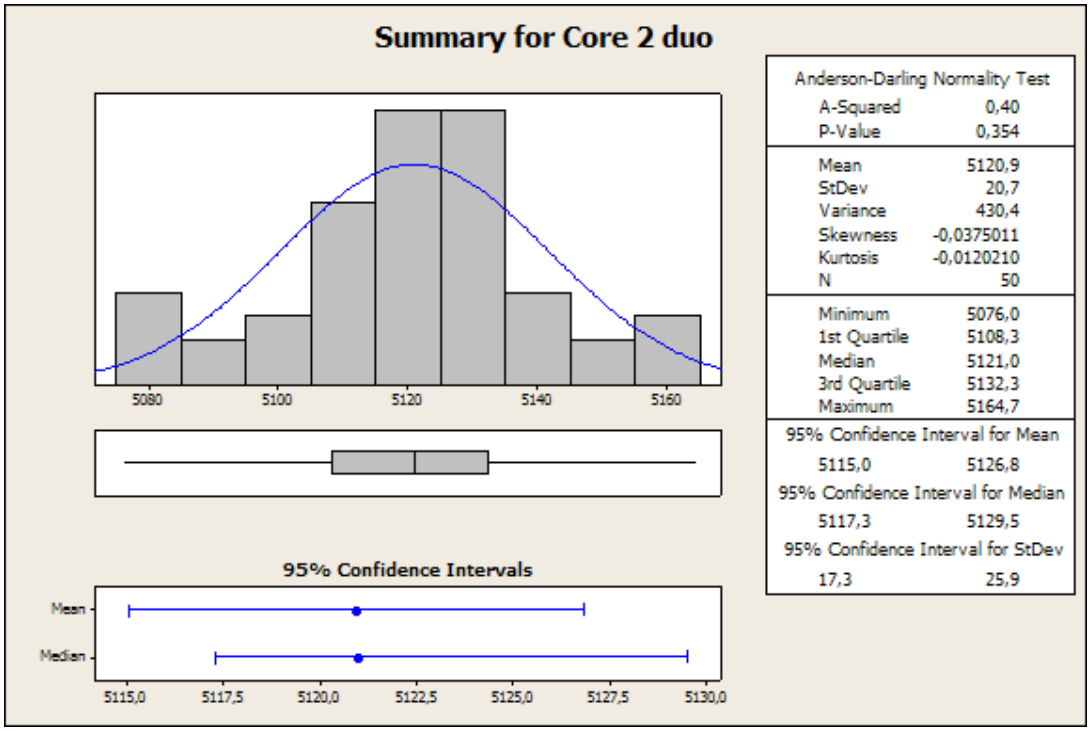


Figura 5.11- Resumo estatístico do Core 2 Duo para o Everest

```
Paired T for Turion x2 - Core 2 duo

             N      Mean  StDev  SE Mean
Turion x2   50   3980,70   0,61     0,09
Core 2 duo  50   5120,94  20,75     2,93
Difference  50  -1140,24  20,73     2,93


95% CI for mean difference: (-1146,13; -1134,35)
T-Test of mean difference = 0 (vs not = 0): T-Value = -388,86  P-Value = 0,000
```

Figura 5.12 - Teste T de paridade do Turion X2 X Core 2 Duo para o Everest

Tabela 5.7 – Resumo dos valores para o Turion x2 (Apêndice B)

Média	Desvio Padrão	Primeiro Quartil	Mediana	Terceiro Quartil	I.Confiança 95% da média
3980,7	0,6	3980,4	3980,7	3981,3	(3980,5; 3980,9)

Tabela 5.8 – Resumo dos valores para o Core 2 Duo (Apêndice B)

Média	Desvio Padrão	Primeiro Quartil	Mediana	Terceiro Quartil	I.Confiança 95% da média
5120,9	20,7	5108,3	5121	5132,3	(5115,0; 5126,8)

Analisando os gráficos acima, concluímos que em ambos os sistemas as amostras coletadas foram precisas e exatas, foi uma excelente coleta. Deve-se

bastante a qualidade do Everest como Benchmark, pois é um aplicativo profissional. Vemos o intervalo de confiança de 95% das médias do Turion X2 e do Core 2 duo respectivamente (3980,5; 3980,9) e (5115,0; 5126,8) MB/s.

Com o T pair test, fica claro a superiodade na leitura da memória principal do core 2 duo sobre o turion x2. Este fato é de extrema importância para o desempenho geral dos microprocessadores multicores segundo [6]. O resultado não é surpresa devido à cache do Core 2 duo ter 2 MB(Tabela 5.2), enquanto a memória cache do turion x2 ter 1 MB(Tabela 5.1). A memória cache torna a leitura bem mais rápida, pois a leitura na memória principal custa mais, computacionalmente.

Comentários

As ferramentas estatísticas utilizadas são muito práticas e eficientes, porque tornam a análise um procedimento bastante acessível, pois o analista não necessita de um domínio profundo das técnicas utilizadas. Porém, é preciso um estudo prévio sobre como utilizá-las para não prejudicar o processo. São ferramentas amplamente documentadas, o que facilita o aprendizado.

Além dos resultados obtidos que foram comentados anteriormente, um fato chamou à atenção no processo de medição e análise: a extrema instabilidade do Bechmark em Java. Apesar do ganho em portabilidade, a tecnologia mostrou-se não muito eficiente para utilização como Benchmarks. Pois, sua máquina virtual, que ao mesmo tempo em que dá suporte à aplicação, realiza outras atividades como o Garbage Collector, que é um processo usado no gerenciamento de memória nos sistemas computacionais para recuperar zonas de memória não mais utilizadas. A máquina virtual acaba compartilhando a CPU com a carga de trabalho.Uma solução para esse problema poderia ser a prévia eliminação dos *outliers* na fase de coleta dos dados.

6 – Conclusão

Este trabalho apresentou uma metodologia para avaliação e comparação de sistemas computacionais. A metodologia proposta é flexível para proporcionar ao analista a possibilidade de utilizá-la para os mais diversos objetivos. O estudo de caso Utilizou Benchmarks Sintéticos como cargas de trabalho, pois têm a capacidade de emular as principais características dos mais diversos tipos de aplicação, ou seja , podem ser utilizados para se atingir os objetivos desejados.

Para a escolha da carga de trabalho a ser utilizada, é importante um estudo da teoria de métricas e medições, pois muitas vezes não se sabe a métrica adequada para o problema em questão, tampouco a melhor forma de medição. Por se tratar de amostras aleatórias, a análise dos dados é feita utilizando ferramentas estatísticas, para quais existem softwares que executam os cálculos para o analista, o que torna a análise um procedimento extremamente prático. A análise de desempenho, que para muitos é conjunto de técnicas, se apresentou como uma ciência com diversas vertentes de estudo.

O estudo de caso realizado demonstrou a falta de Benchmarks específicos para microprocessadores multicores, fato esse, comentado por artigos científicos [8]. A maioria dos Benchmarks existentes para avaliação são antigos, mas em contrapartida os microprocessadores multicores são tecnologias recentes o que de certa forma deixa lacunas e, conseqüentemente, um bom campo de pesquisa.

Concluímos, no estudo de caso, que para o determinado cenário proposto, respeitando as precondições estabelecidadas e com base nas amostras coletadas, durante a fase de medição, que apesar de existirem diferenças para determinados tipos de aplicações, os processadores Turion X2 TL 58 e Core 2 Duo T7100 são processadores de desempenhos muito próximos. Se considerarmos que não levamos em conta a placa mãe dos computadores, os desempenhos não foram muito diferentes.

No processo de medição, a procurada portabilidade na tecnologia Java mostrou-se um fator de instabilidade devido ao uso da Máquina Virtual do Java que acaba sendo um processo que compartilha o microprocessador. O uso de programas profissionais como o Everest mostrou uma extrema estabilidade no processo de medição, porém encarece a metodologia por se tratar de programas não gratuitos.

A análise dos dados, na fase de medição da metodologia, mostrou a importância do estudo da estatística que por muitas vezes é deixada de lado pelos profissionais de tecnologia da informação.

Este trabalho surge como uma alternativa para os engenheiros e analistas que necessitem executar uma análise de desempenho, mesmo aqueles sem experiência alguma na área. Os capítulos anteriores à metodologia, que é de fácil compreensão, oferecem todo o conhecimento básico necessário para uma análise de desempenho consciente.

6.1 – Trabalhos Futuros

A partir da conclusão desse trabalho, surgem várias oportunidades que o complementam. Temos alguns trabalhos propostos: criar Benchmarks específicos para avaliação de microprocessadores multicores, estudar o desempenho da máquina virtual Java ou a criação de um modelo para metodologia proposta

7 – Referências

[1] Lilja, D. - Cambridge - Measuring Computer Performance,2000

[2] Wiley - Applied Statistic and Probability for Engineers, 2006

[3] Menascé, D.A., Almeida, V.A.F., Dowdy, L.W. - Performance by Design: Computer Capacity Planning by Example. 2005

[4] Meyer, P. – Probabilidade, Aplicações e Estatística,2000

[5] Wang, L., Tsen, C., Schulte, M., Jhalani, D. -Benchmarks and Performance Analysis of Decimal Floating-Point Applications, 2008

[6] Gal-On, S., Levy, M. – Measuring Multicore Performance, 2008

[7] Stefani,I., Cordeiro,E. , Soares,T. , Martins, C. -DCMSim: Uma ferramenta para simulação de memória cache,2005

[8] Avaliação de Desempenho. Disponível em: http://www.cin.ufpe.br/~prmm/IntTopAval.pdf, último acesso em 17/11/2008.

[9]Wolfram MathWorld: The Web's most extensive mathematics resource. Disponível em: http://mathworld.wolfram.com/, último acesso em 17/11/2008.

[10] Hoste, K. , Phansalkar,A. , Eeckhout, L. , Georges, A. , Lizy K.John, Koen de Bosschere – Performance Prediction based on Inherent Program Similarity, 2006

[11] Lizy Kurian John, Lieven Eeckhout– Performance Evaluation and Benchmarking, 2005

.

[12] Medeiros ,S. – Medição de energia elétrica, 1981

Apêndice A: Teorema Fundamental do Cálculo

Considere *f* uma função contínua de valores reais definida em um intervalo fechado [*a*, *b*]. Se *F* for a função definida para *x* em [*a*, *b*] por

$$F(x) = \int_a^x f(t)\,dt$$

então

$$F'(x) = f(x)$$

para todo *x* em [*a*, *b*].

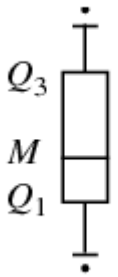
Considere *f* uma função contínua de valores reais definida em um intervalo fechado [*a*, *b*]. Se *F* é uma função tal que

$f(x) = F'(x)$ para todo *x* em [*a*, *b*]

então

$$\int_a^b f(x)dx = F(b) - F(a).$$

Apêndice B: Gráfico Box-Plot



Demonstra a variabilidade dos dados coletados sendo:

- Q_3: Terceiro Quartil, a quarta parte dos valores em ordem crescente.
- M: Mediana
- Q_1: Valor do primeiro quartil.
- Valores completamente a afastados dos demais são representados por '*', *outliers.*

Exemplo 1:

Amostra: 6, 47, 49, 15, 42, 41, 7, 39, 43, 40, 36

Amostra ordenada: 6, 7, 15, 36, 39, 40, 41, 42, 43, 47, 49

Q1 = 15

M = 40

Q3 = 43

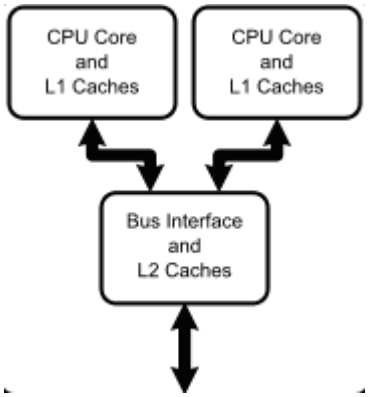
Apêndice C: Tabela Dist. Normal

Distribuição Normal : Valores de P(Z ≤ z) = A(z)

Parte inteira e primeira decimal de z	Segunda decimal de z									
	0	1	2	3	4	5	6	7	8	9
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.0	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.1	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.2	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.3	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.4	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.5	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.6	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.7	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.8	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.9	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2.0	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.1	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857
2.2	0.9861	0.9864	0.9868	0.9871	0.9875	0.9878	0.9881	0.9884	0.9887	0.9890
2.3	0.9893	0.9896	0.9898	0.9901	0.9904	0.9906	0.9909	0.9911	0.9913	0.9916
2.4	0.9918	0.9920	0.9922	0.9925	0.9927	0.9929	0.9931	0.9932	0.9934	0.9936
2.5	0.9938	0.9940	0.9941	0.9943	0.9945	0.9946	0.9948	0.9949	0.9951	0.9952
2.6	0.9953	0.9955	0.9956	0.9957	0.9959	0.9960	0.9961	0.9962	0.9963	0.9964
2.7	0.9965	0.9966	0.9967	0.9968	0.9969	0.9970	0.9971	0.9972	0.9973	0.9974
2.8	0.9974	0.9975	0.9976	0.9977	0.9977	0.9978	0.9979	0.9979	0.9980	0.9981
2.9	0.9981	0.9982	0.9982	0.9983	0.9984	0.9984	0.9985	0.9985	0.9986	0.9986
3.0	0.9987	0.9987	0.9987	0.9988	0.9988	0.9989	0.9989	0.9989	0.9990	0.9990
3.1	0.9990	0.9991	0.9991	0.9991	0.9992	0.9992	0.9992	0.9992	0.9993	0.9993
3.2	0.9993	0.9993	0.9994	0.9994	0.9994	0.9994	0.9994	0.9995	0.9995	0.9995
3.3	0.9995	0.9995	0.9995	0.9996	0.9996	0.9996	0.9996	0.9996	0.9996	0.9997
3.4	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9998
3.5	0.9998	0.9998	0.9998	0.9998	0.9998	0.9998	0.9998	0.9998	0.9998	0.9998
3.6	0.9998	0.9998	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
3.7	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
3.8	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
3.9	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

Apêndice D: Arquitetura Dual Core

Um processador multicore combina dois ou mais cores, normalmente chamados de unidade de processamento, no mesmo circuito integrado. Abaixo temos um diagrama de um microprocessador dual core genérico:



Vemos dois cores com respectivas suas caches L1 e a interface de barramento com a cache L2. No estudo de caso fizemos a abstração das caches L1. Foi considerada somente a cache L2 para a avaliação.

Apêndice E: Documento de Medição

Resposável:
Analista da Medição:
Local:
Data:

Descrição do Sistema	
Processador	
Memória Ram	
Disco rígido	
Sistema operacional	

Informações de coleta	
Carga de trabalho	
Métrica	
Resolução	
Frequencia	
Temperatura	
Qtde de medições	
Início	
Término	

Dados Coletados							