

Desempenho e Estabilidade em Processamento Paralelo

Monografia apresentada como trabalho de avaliação da disciplina SCE217 – Programação Concorrente do curso de Bacharelado em Ciência da Computação do ICMC - USP.

1º. Semestre de 2003

Data: 16 de junho de 2003

Alexandre Gomes de Siqueira nº USP 3309009

Elena Balachova nº USP 3309292

Everton Kazuo Iwamoto nº USP 3309184

Flávio Monteiro Wanderley nº USP 3280896

Resumo

Este trabalho apresenta conceitos básicos sobre métodos para medir performance e analisar estabilidade em processamento paralelo, sendo esses conceitos de extrema importância para a formação de um Bacharel em Ciência da Computação. No decorrer deste texto serão apresentados os conceitos mais relevantes acompanhados de problemas e dificuldade para quantificar o desempenho e a estabilidade em programas paralelos e que têm por função deixar claro ao leitor a matéria que se estuda. O assunto abordado trata de conceitos ligados à disciplina de Programação Concorrente, mais especificamente, relacionados a métodos de avaliar o desempenho e a estabilidade em processamento paralelo.

Índice

1.INTRODUÇÃO.....	9
2.PERFORMANCE.....	11
O PRINCIPAL OBJETO DESSE TÓPICO É APRESENTAR QUAIS MEDIDAS USAR OU QUAL NÍVEL DE DETALHE DO SISTEMA USAR PARA ANALISAR O DESEMPENHO DO SISTEMA PARALELO. PARA ISSO, SERÃO INTRODUZIDOS CONCEITOS BÁSICOS DE SPEEDUP E PERFORMANCE A FIM DE SE TER UMA RAZOÁVEL BASE SOBRE IDÉIAS QUE TEM SURGIDO NA LITERATURA.....	
2.1MELHORANDO PERFORMANCE EM PROCESSAMENTO PARALELO.....	11
A MAIS ÓBVIA FUNÇÃO QUE MEDE O DESEMPENHO DE UM SISTEMA É O TEMPO DE EXECUÇÃO DE UM CÓDIGO PARTICULAR. PORÉM, CÓDIGOS DIFERENTES E TAMANHOS DE DADOS FORNECEM TEMPOS DE EXECUÇÃO DIFERENTES, O QUE TORNA DIFÍCIL A COMPARAÇÃO. TAXAS SÃO MAIS USADAS, COMO POR EXEMPLO, NÚMERO DE OPERAÇÕES POR UNIDADE DE TEMPO, MEGAFLOPS OU INSTRUÇÕES POR UNIDADE DE CLOCK.....	11
EM GERAL, O SPEEDUP DE UM SISTEMA SOBRE OUTRO PARA UM DADO CÓDIGO É MUITO UTILIZADO PARA COMPARAR DOIS SISTEMAS. PARA A COMPUTAÇÃO PARALELA, O SPEEDUP DE UM SISTEMA COM UM ÚNICO PROCESSADOR DIVIDIDO POR UM COM MÚLTIPLOS PROCESSADORES É OUTRA BOA ESCOLHA PARA A FUNÇÃO QUE MEDE O DESEMPENHO. ESSA RELAÇÃO TRATA-SE DA FUNÇÃO DE PERFORMANCE RELATIVA, QUE PODE SER EXPRESSA COMO:.....	11
.....	11
UTILIZANDO O TEMPO DE EXECUÇÃO T COMO A FUNÇÃO PERF, SPEEDUP S_p COMO A FUNÇÃO REL PERF E RESTRINGINDO PARA UM ÚNICO CÓDIGO, TEMOS:.....	11
2.2S_p VS. SISTEMA PARALELO.....	11
PARA SIMPLIFICAR A ANÁLISE DE DESEMPENHO EM SISTEMAS PARALELOS, IREMOS DISCUTIR APENAS O NÚMERO DE PROCESSADORES UTILIZADOS EM UMA MÁQUINA PARALELA, IGNORANDO OUTROS FATORES QUE PRODUZEM VARIAÇÕES NO DESEMPENHO. A FIGURA 2.1 DEMONSTRA O QUE DEVE SER ESPERADO QUANDO MAIS PROCESSADORES SÃO USADOS PARA UMA CERTA COMPUTAÇÃO.....	12
.....	12
O SPEEDUP NÃO PODE EXCEDER p , PARA UMA CERTA COMPUTAÇÃO COM p -PROCESSADORES EM PARALELO, AO MENOS SE HOUVER MUDANÇA NA COMPUTAÇÃO OU OUTROS RECURSOS ARQUITETURAIS SÃO ALTERADOS (POR EXEMPLO, MEMÓRIA POR PROCESSADOR).....	12
ANALISANDO A FIGURA 2.1, SURGEM AS SEGUINTE PERGUNTAS:.....	12
ATÉ QUANDO A PERFORMANCE PODE AUMENTAR À MEDIDA QUE SE AUMENTA O NÚMERO DE PROCESSADORES?.....	13
PARA QUAL VALOR DE p A PERFORMANCE CAI DEMAIS ABAIXO DA LINHA IDEAL?.....	13
QUE MUDANÇAS PODEM SER FEITAS EM UM SISTEMA PARALELO PARA MELHORAR A CURVA PERFORMANCE VS. NÚMERO DE PROCESSADORES PARA UM DADO SISTEMA?.....	13
PRIMEIRAMENTE, DESIGNERS PODEM ALTERAR O HARDWARE E A ARQUITETURA DO SISTEMA PARA MELHORAR A PERFORMANCE. UMA ALTERNATIVA SÃO SOFTWARES MELHORES QUE PODEM SER UTILIZADOS. FINALMENTE, OS USUÁRIOS PODEM ESCREVER PROGRAMAS MELHORES E USAR MELHORES ALGORITMOS E ASSIM, EXPLORAR MELHOR O HARDWARE E COM ISSO, PRODUZIR UM MELHOR DESEMPENHO.....	13
2.3S_p VS. TIPO DE COMPUTAÇÃO.....	13

CONSIDERANDO UMA VARIEDADE DE APLICAÇÕES SENDO RESOLVIDO EM UM DADO SISTEMA PARALELO COM UM NÚMERO FIXO DE PROCESSADORES, PODE-SE OBSERVAR O NÍVEL DE DESEMPENHO QUE PODE SER OBTIDO. NO MOMENTO, SERÁ EVITADO VARIAR O PROGRAMA E A ESTRUTURA DE DADOS USADA PARA RESOLVER UM DADO PROBLEMA PARA SIMPLIFICAR A ANÁLISE. A VARIAÇÃO SERÁ FEITA APENAS NO TAMANHO DOS DADOS, N , PARA UM DADO PROGRAMA E ESTRUTURA DE DADOS.....	13
COM ESSA RESTRIÇÃO, FICA INTUITIVAMENTE CLARO QUE PARA UM DADO NÚMERO DE PROCESSADORES P , PARA UM PEQUENO TAMANHO DE DADOS NÃO HÁ UM SPEEDUP PRÓXIMO DE P . À MEDIDA QUE O TAMANHO DO DADO AUMENTA, A PERFORMANCE AUMENTA. A FIGURA 2.2 MOSTRA O GRÁFICO PRODUZIDO A MEDIDA QUE O TAMANHO DO DADO AUMENTA.....	13
2.4Sp vs. P.....	14
SEND O TEMPO NECESSÁRIO $T(i)$ PARA UMA DETERMINADA COMPUTAÇÃO UTILIZANDO i PROCESSADORES, É POSSÍVEL DEFINIR A MEDIDA PADRÃO DO SPEEDUP UTILIZANDO P PROCESSADORES:..	14
.....	14
ASSUME-SE QUE NENHUMA CARACTERÍSTICA DO SISTEMA É VARIADA A NÃO SER O NÚMERO DE PROCESSADORES. APESAR DE MUITOS PROBLEMAS SEREM DESCONSIDERADOS PARA ANALISAR O SPEEDUP, É POSSÍVEL OBSERVAR QUE O SPEEDUP EM UM SISTEMA REAL OBEDECE À EQUAÇÃO 2.1 E TEM COMO CARACTERÍSTICA A FIGURA 2.1.....	14
NA FIGURA 2.1, FOI MOSTRADO DOIS PONTOS DE INTERESSE NA CURVA. O PONTO DE RETORNO REDUZIDO DEMONSTRA O PONTO ONDE SE ADICIONA MAIS PROCESSADORES E SE OBTÉM POUCO AUMENTO NA PERFORMANCE. O PONTO SEM RETORNO É O PONTO ONDE SE ADICIONA MAIS PROCESSADORES, PORÉM NÃO HÁ AUMENTO DE PERFORMANCE.....	15
.....	15
2.5Sp vs. N.....	15
NESTE TÓPICO, SERÁ CONSIDERADO QUE O NÚMERO DE PROCESSADORES É CONSTANTE E O TAMANHO DO DADO N É VARIADO PARA UM DADO PROGRAMA. PODE-SE ASSIM OBTER A SEGUINTE EQUAÇÃO:.....	15
.....	15
ATRAVÉS DESSA EQUAÇÃO É POSSÍVEL DESCOBRIR E MELHORAR A PERFORMANCE DO HARDWARE E SOFTWARE DE UM DADO SISTEMA QUANDO SUA CURVA (SP,N) — FIGURA 2.2 — É COMPARADA COM OUTROS SISTEMAS PARA ALGUM PROGRAMA. QUANDO MÚLTIPLAS COMPUTAÇÕES SÃO COMPARADAS, PODE-SE DESCOBRIR SOFTWARES PARA MELHORAR O SISTEMA COMO UM TODO.....	15
UTILIZANDO UMA MÁQUINA COM PROCESSADORES VETORIAIS, ATRAVÉS DA FIGURA 2.2 É POSSÍVEL OBSERVAR QUE A PERFORMANCE AUMENTA QUANDO O PIPELINE ESTÁ EM SUA FASE INICIAL ATÉ QUANDO O SISTEMA PARALELO É SATURADO. NESTE PONTO OS PROCESSADORES ESTÃO ACESSANDO A MEMÓRIA CACHÊ PARA A MAIORIA DOS DADOS. QUANDO O VALOR DE N AUMENTA AINDA MAIS, O SISTEMA COMEÇA A AUMENTAR O NÚMERO DE ACESSOS A MEMÓRIA PRINCIPAL ATÉ QUE, FINALMENTE, O TAMANHO DO DADO EXCEDE A MEMÓRIA PRINCIPAL E ASSIM, COMEÇA A ACESSAR O DISCO O Q DEGRADA CONSIDERAVELMENTE A PERFORMANCE DO SISTEMA.....	15
PARA AUMENTAR A PERFORMANCE DE UM DETERMINADO SISTEMA COM P PROCESSADORES VETORIAIS UTILIZANDO DADOS DE TAMANHO PEQUENOS OU GRANDES, É NECESSÁRIO QUE O SOFTWARE SEJA CAPAZ DE AUMENTAR A PERFORMANCE.	15
PARA DADOS PEQUENOS, PARA EXPLORAR TOTALMENTE O SISTEMA PARALELO, MAIS CÓDIGOS PARALELOS DEVEM SER ENCONTRADOS ATÉ QUE O GRAU DE PARALELISMO SEJA SUFICIENTE PARA SATURAR OS PROCESSADORES EM PARALELO. PARA TIPO DE DADOS GRANDES, MAIS REFERÊNCIA DE DADOS LOCAIS DEVEM SER ENCONTRADOS PARA EVITAR ACESSOS AO DISCO, OU SEJA, O FLUXO DE COMUTAÇÃO DEVE SER MODIFICADO DE MODO QUE O ESPAÇO DE ENDEREÇAMENTO SEJA MAIS OU MENOS UNIFORME.....	16
2.6LIMITAÇÕES NA PERFORMANCE.....	16

PERFORMANCE EM SISTEMAS PARALELOS É LIMITADA POR DIVERSOS FATORES ARQUITETURAIS E DE SOFTWARES. A COMPUTAÇÃO ENVOLVE UM DADO PARTICULAR E UM DETERMINADO CÓDIGO. JUNTOS, PODE-SE DETERMINAR UM Γ QUE CORRESPONDE A PORCENTAGEM QUE DEVE SER EXECUTADO PARALELO E O RESTANTE $(1 - \Gamma)$ SERÁ EXECUTADO SEQUENCIALMENTE. COM ISSO, UTILIZANDO O SPEEDUP (EQUAÇÃO 2.1) PODE-SE CHEGAR NA EQUAÇÃO 2.3:.....	16
ESSA RELAÇÃO É CONHECIDA COMO LEI DE AMDAHL'S QUE É UMA DAS LEIS MAIS FUNDAMENTAIS NO ESTUDO DE SISTEMAS PARALELOS. A RELAÇÃO ENTRE O SPEEDUP E Γ É MOSTRADA NA FIGURA 2.3.....	16
.....	17
2.7 EFEITOS DE COMPILADOR E SOFTWARE.....	17
O DESEMPENHO EM UM SISTEMA PARALELO É EXTREMAMENTE SENSÍVEL À ESTRUTURA DO SOFTWARE. O COMPILADOR É O SOFTWARE QUE FAZ A INTERFACE ENTRE O CÓDIGO FONTE DO USUÁRIO E O HARDWARE, PORTANTO, O COMPILADOR É DE EXTREMA IMPORTÂNCIA NO DESEMPENHO.....	17
OS COMPILADORES PODEM SER DIVIDIDOS EM TRÊS ESTÁGIOS A FIM DE OTIMIZAR O DESEMPENHO. O PRIMEIRO ESTÁGIO É UM REESTRUTURADOR GLOBAL QUE TEM COMO OBJETIVO FAZER AS TRANSFORMAÇÕES QUE SÃO NECESSÁRIAS PARA UMA CERTA CLASSE ARQUITETURAL E TAMBÉM PARA CERTAS MÁQUINAS ESPECÍFICAS. ELE PODE REESTRUTURAR O CÓDIGO A FIM DE REDUZIR O OVERHEAD DE COMUNICAÇÃO OU MAXIMIZAR O NÍVEL DE PARALELISMO DO CÓDIGO.....	18
O SEGUNDO ESTÁGIO É UM OTIMIZADOR DO CÓDIGO RESULTANDO GERADO PELO PRIMEIRO ESTÁGIO. ESTE ESTÁGIO OTIMIZA O CÓDIGO ESPECIFICAMENTE PARA O SISTEMA.....	18
O ÚLTIMO ESTÁGIO GERA O EXECUTÁVEL EM CÓDIGO DE MÁQUINA. O RESULTADO ESPERADO É O DE MÁXIMA PERFORMANCE PARA O SISTEMA. PODE OCORRER QUE O RESULTADO SEJA UMA BAIXA PERFORMANCE, POIS PODE HAVER DEFICIÊNCIA NO CÓDIGO ORIGINAL GERADO PELO USUÁRIO, O QUE DIFICULTA A OTIMIZAÇÃO GERADA PELO COMPILADOR. ESSAS DEFICIÊNCIAS PODEM SER REMOVIDAS REESCREVENDO O CÓDIGO OU UTILIZANDO BIBLIOTECAS OTIMIZADAS.....	18
3.PERFORMANCE SPEEDUP.....	19
SPEEDUP E EFICIÊNCIA TÊM SIDO USADOS LARGAMENTE COMO MEDIDAS DE DESEMPENHO. EMBORA HÁ MUITAS DIFICULDADES E PROBLEMAS EM SE UTILIZAR ESSAS MEDIDAS, ELES FUNCIONAM RAZOAVELMENTE BEM EM MUITAS SITUAÇÕES PRÁTICAS. OS DIVERSOS PARÂMETROS - A ARQUITETURA, O SOFTWARE, A ESTRUTURA DE DADOS, A ESTRUTURA DO PROGRAMA - AFETAM A PERFORMANCE DO SISTEMA. TODOS ESSES PARÂMETROS POSSUEM DIFERENTES VALORES NO MUNDO REAL E COM ISSO, PODE GERAR UMA CERTA CONFUSÃO NA DISCUSSÃO DA PERFORMANCE. OS DOIS PRINCIPAIS PARÂMETROS QUE LIMITAM A PERFORMANCE EM UM SISTEMA PARALELO SÃO: O NÚMERO DE PROCESSADORES USADOS E O TAMANHO DO DADOS PARA UM DETERMINADO PROBLEMA.....	19
NESTE TÓPICO SERÃO APRESENTADOS OS LIMITES REAIS DO SPEEDUP, ALGUNS NÍVEL DE DESEMPENHO E TAMANHO DE DADOS UTILIZADOS... 19	
3.1 LIMITES PRÁTICOS DO SPEEDUP.....	19
EM UM SISTEMA PARALELO REAL, QUANTO MAIS PROCESSADORES PARALELOS SÃO UTILIZADOS, MAIS O OVERHEAD DO LIMITANTE ARQUITETURAL DO SISTEMA SERÁ NOTADO. ESSAS LIMITAÇÃO INCLUI O LIMITANTE DE BANDA E A LATÊNCIA DA TRANSFERÊNCIA DOS DADOS. O LIMITANTE DO SISTEMA DE COMUNICAÇÃO LIMITA O POTENCIAL DO SPEEDUP DE QUALQUER COMPUTAÇÃO FAZENDO COM QUE A CURVA DIVIRJA DA CURVA IDEAL CONFORME O NÚMERO DE PROCESSADORES USADOS CRESCE.....	19

UM OUTRO PROBLEMA OCORRE QUANDO SE VARIA O TAMANHO DO DADO. PARA UM DADO PEQUENO, É IMPOSSÍVEL UTILIZAR MAIS DO QUE UM PROCESSADOR. À MEDIDA QUE O TAMANHO DO DADO AUMENTA, A PERFORMANCE TAMBÉM AUMENTA. ESSA É A FASE DO SISTEMA INSATURADO.....	19
DURANTE A FASE SATURADA DO SISTEMA, O LIMITANTE PRINCIPAL DO SPEEDUP É O SISTEMA DE COMUNICAÇÃO. NESSA FASE, A MEDIA QUE O DADO AUMENTA, A PERFORMANCE SOFRE POUCAS MODIFICAÇÕES. QUANDO O DADO TORNA-SE SUFICIENTE GRANDE, A PERFORMANCE CAI O QUE CARACTERIZA UM SISTEMA SOBRECARGADO.....	19
3.2 NÍVEIS DE PERFORMANCE.....	20
DEFINIR MEDIDAS DE PERFORMANCE É MUITO MAIS FÁCIL DO QUE USÁ-LAS. NA PRÁTICA, HÁ QUALITATIVAMENTE NÍVEIS DE PERFORMANCE ACEITÁVEL E NÃO ACEITÁVEIS, CUJA DEFINIÇÃO DEPENDE DO USUÁRIO. A TABELA 3.1 MOSTRA OS NÍVEIS DE DESEMPENHO PRESENTE NA LITERATURA.....	20
NOME.....	20
SPEEDUP.....	20
ALTA PERFORMANCE.....	20
$\frac{P}{2} \leq S_P \leq P$	20
2.....	20
MÍNIMA ALTA PERFORMANCE.....	20
$S_P = \frac{P}{2}$	20
2.....	20
PERFORMANCE INTERMEDIÁRIA.....	20
$\frac{P}{2^{\log(P)}} \leq S_P \leq \frac{P}{2}$	20
$2^{\log(P)}$	20
NÍVEL THRESHOLD.....	20
$S_P = \frac{P}{2^{\log(P)}}$	20
2.....	20
PERFORMANCE INACEITÁVEL.....	20
$1 \leq S_P < \frac{P}{2^{\log(P)}}$	20
2.....	20
3.3 TAMANHO DOS DADOS.....	20
O TAMANHO DOS DADOS DEFINIDOS PELO USUÁRIO PODE SER DIVIDIDO EM TRÊS TAMANHOS:.....	20
SUDS (SMALL USER-DEFINED DATA SIZE).....	20
LUDS (LARGE USER-DEFINED DATA SIZE).....	20
MUDS (MEDIUM USER-DEFINED DATA SIZE).....	20
SUDS É TÍPICAMENTE PARA DEPURAÇÃO DURANTE O DESENVOLVIMENTO DO CÓDIGO. O RESULTADO GERADO POR ESSE TIPO DE DADO PODE NÃO TER MUITO SIGNIFICADO, PORÉM ELE PODE DIZER SE O PROGRAMA ESTÁ CORRETO.....	21
JÁ O LUDS PODE LEVAR MAIS DO QUE UMA HORA PARA A EXECUÇÃO COMPLETA. SERVE PARA ANALISAR A MEMÓRIA DO SISTEMA.....	21
MUDS É UM TAMANHO ENTRE LUDS E SUDS. POSSUI UM SIGNIFICADO FÍSICO E POSSUI EXECUÇÃO RÁPIDA, MESMO EM UM SISTEMA TIME-SHARED. ESTE TAMANHO É GERALMENTE USADO PARA MEDIÇÕES.....	21
EXISTEM TAMBÉM O TAMANHO DE DADOS DEFINIDO PELA ARQUITETURA. PODE SER DIVIDIDO EM:.....	21
SADS (SMALL ARCHITECTURE-DEFINED DATA SIZE).....	21
MADS (MEDIUM ARCHITECTURE-DEFINED DATA SIZE).....	21
LADS (LARGE ARCHITECTURE-DEFINED DATA SIZE).....	21
EM SADS, O TAMANHO É DEFINIDO DE MODO QUE O SPEEDUP SE ENCONTRA NO NÍVEL DE PERFORMANCE. ANALOGAMENTE, MADS DEVE SER UM TAMANHO NA QUAL O SPEEDUP É PRÓXIMO A	

MÍNIMA ALTA PERFORMANCE. EM LADS, O TAMANHO DO DADO DEVE CAUSAR UMA PERDA SUBSTANCIAL NA PERFORMANCE. PODE-SE QUANTIFICAR LADS COMO O PONTO ONDE O SPEEDUP CAI ABAIXO DE $P/2$	21
4. ESTABILIDADE.....	22
ATÉ AGORA, A DISCUSSÃO PARA MEDIR O DESEMPENHO UTILIZAVAM-SE SISTEMAS DIFERENTES E UM ÚNICO CÓDIGO. PARA A ANÁLISE DA ESTABILIDADE SERÁ UTILIZADO UM ÚNICO SISTEMA COM P PROCESSADORES EXECUTANDO UM OU MAIS CÓDIGOS.....	22
CONSIDERANDO UM CONJUNTO SIMILAR DE COMPUTAÇÕES QUE PODEM VARIAR SOBRE K CÓDIGOS E/OU VARIAR O TAMANHO DOS DADOS NI PARA OS CÓDIGOS, PODE-SE DEFINIR ESTABILIDADE COMO SENDO.....	22
.....	22
É POSSÍVEL OBSERVAR QUE:.....	22
.....	22
QUANTO MAIS PRÓXIMO DE 1, O SISTEMA SE CARACTERIZA COMO SENDO ESTÁVEL. MAIS PRÓXIMO DE 0, UM SISTEMA INSTÁVEL.....	22
SE HOUVER VARIAÇÃO APENAS NO CÓDIGO, E UTILIZAR APENAS UM DADO DE ENTRADA, OBTÉM-SE A FUNÇÃO ESTABILIDADE DE PROGRAMA:.....	22
.....	22
PODE-SE TAMBÉM VARIAR APENAS OS DADOS DE ENTRADA UTILIZANDO APENAS UM DETERMINADO CÓDIGO. COM ISSO, É OBTIDO A FUNÇÃO ESTABILIDADE DE DADOS:.....	22
A PARTIR DESSAS FUNÇÕES, SÃO DEFINIDOS PARÂMETROS QUE CARACTERIZAM SE UM SISTEMA É ESTÁVEL OU NÃO. UM BOM PARÂMETRO Π PARA DEFINIR UM SISTEMA ESTÁVEL É:.....	22
.....	22
SISTEMAS SERÃO CONSIDERADOS ESTÁVEIS CASO A FUNÇÃO ESTABILIDADE ESTÁ ENTRE $1/6$ E 1 . SISTEMAS INSTÁVEL SÃO AQUELES QUE ESTÃO ABAIXO DE $1/6$	22
A SEGUIR ALGUMAS MEDIDAS DE ESTABILIDADE REALIZADAS:.....	22
DÉCADA.....	23
SISTEMA.....	23
PERFECT BASELINE INSTABILITY.....	23
1970.....	23
VAX 780.....	23
5.....	23
.....	23
DEC 6000-410.....	23
4.7.....	23
1980.....	23
STARDENT 3010.....	23
5.1.....	23
.....	23
SUN SPARC 2.....	23
5.37.....	23
1990.....	23

IBM RS6000.....	23
5.....	23
.....	23
.....	23
WORKSTATION (1993).....	23
SPECrATIO INSTABILITY.....	23
DEC 3000 MODEL 500X.....	23
7.77.....	23
HP 9000 MODEL G/H/160.....	23
3.70.....	23
IBM RS 6000-POWERSTATION 370/375.....	23
4.84.....	23
MOTOROLA SERIES 900.....	23
5.19.....	23
SGI INDIGO 2.....	23
4.56.....	23
SUN SPARCSTATION 10 MODEL 40.....	23
4.16.....	23
.....	23
5.CONCLUSÃO.....	24
NESTA MONOGRAFIA FORAM DISCUTIDOS MÉTODOS DE MEDIA DE DESEMPENHO E OS PROBLEMAS RELACIONADOS. FOI POSSÍVEL OBSERVAR AS DIFICULTADOS PARA SE OBTER UMA MEDIA CONSISTENTE QUE REFLITA DE FORMA REAL O COMPORTAMENTO DO SISTEMA PARALELO. FORAM APRESENTADOS OS FATORES QUE AFETAM A PERFORMANCE DO SISTEMA É AS RAZÕES PELA QUAL NÃO SE CONSEGUE OBTER UM SISTEMA PARALELO IDEAL.....	
	24
FOI RESSALTADA A NECESSIDADE DE SE OBTER UMA BOA MEDIA DE DESEMPENHO, POIS ESTA PODERÁ SER UTILIZADA PARA MELHORAR O SISTEMA COMO UM TODO. ENFIM, FOI APRESENTADO O CONCEITO DE ESTABILIDADE E COMO MEDIR E CLASSIFICAR SE UM SISTEMA É ESTÁVEL OU NÃO.....	
	24
6.REFERÊNCIAS BIBLIOGRÁFICAS.....	25

1. Introdução

A computação paralela surgiu devido à necessidade de aumentar a velocidade computacional para solucionar grandes problemas em um tempo menor quando comparado o mesmo problema resolvido por computadores seqüenciais. Para atingir o alto desempenho em computação paralela são necessários métodos para quantificar o desempenho em processamento paralelo e a partir destes, pode-se também avaliar a estabilidade do sistema.

Quando se pretende avaliar um sistema complexo, o primeiro desafio está em encontrar a técnica adequada para realizar de forma consistente a análise. Muitas questões surgem, incluindo: Qual a função de performance deve ser usada e como o sistema e o código podem ser caracterizados?

De uma maneira genérica, podem-se agrupar as técnicas de avaliação de desempenho em dois grupos, de certa forma, complementares. O primeiro relaciona aquelas técnicas que realizam experimentação no sistema e o segundo grupo relaciona as técnicas que criam abstrações desse sistema, através das quais são feitas inferências sobre o seu funcionamento e de seus componentes. Uma taxonomia para essas técnicas é proposta em (Santana et al., 1997), onde os autores sugerem uma divisão em duas classes: as técnicas de aferição e as de modelagem, sendo que o grande diferencial entre os dois grupos está em ter-se ou não o sistema implementado. Para os casos em que o sistema já existe e, conseqüentemente, pode ser averiguado empiricamente, as técnicas de aferição são mais recomendadas. Estão nessa classe, por exemplo, os *benchmarks*, os protótipos e a coleta de dados (através de monitores de *hardware* e/ou de *software*). Mais referências sobre esse grupo de técnicas podem ser encontradas em (Cortés, 1999) e (Jain, 1991). Em contrapartida, para os sistemas inexistentes, Santana et al. (1997) sugerem as técnicas de modelagem.

Nesta monografia, serão apresentadas apenas as técnicas de aferição, sendo analisado apenas uma pequena parte de todo o processo, ou seja, obter uma grande análise superficial de todo o processo. Com isso, pretende-se discutir as dificuldades que pode criar a degradação da performance do sistema e métodos pelo qual o sistema macroperformance pode ser expressa. Pretende-se também sintetizar algumas idéias que tem surgido na

literatura para o teste de paralelismo e com isso fornecer um embasamento nesta área de pesquisa.

2. Performance

O principal objeto desse tópico é apresentar quais medidas usar ou qual nível de detalhe do sistema usar para analisar o desempenho do sistema paralelo. Para isso, serão introduzidos conceitos básicos de Speedup e performance a fim de se ter uma razoável base sobre idéias que tem surgido na literatura.

2.1 Melhorando Performance em Processamento Paralelo

A mais óbvia função que mede o desempenho de um sistema é o tempo de execução de um código particular. Porém, códigos diferentes e tamanhos de dados fornecem tempos de execução diferentes, o que torna difícil a comparação. Taxas são mais usadas, como por exemplo, número de operações por unidade de tempo, megaflops ou instruções por unidade de clock.

Em geral, o speedup de um sistema sobre outro para um dado código é muito utilizado para comparar dois sistemas. Para a computação paralela, o speedup de um sistema com um único processador dividido por um com múltiplos processadores é outra boa escolha para a função que mede o desempenho. Essa relação trata-se da função de performance relativa, que pode ser expressa como:

$$rel\ perf(system\ A,code\ 1,systemB,code) = perf(system\ A,code\ 1) / perf(system\ B,code\ 2)$$

Utilizando o tempo de execução T como a função $perf$, speedup Sp como a função $rel\ perf$ e restringindo para um único código, temos:

$$Sp(system\ A, systemB,code\ 1) = T(system\ A,code\ 1) / T(system\ B,code\ 1)$$

2.2 Sp vs. Sistema Paralelo

Para simplificar a análise de desempenho em sistemas paralelos, iremos discutir apenas o número de processadores utilizados em uma máquina paralela, ignorando outros fatores que produzem variações no desempenho. A Figura 2.1 demonstra o que deve ser esperado quando mais processadores são usados para uma certa computação.

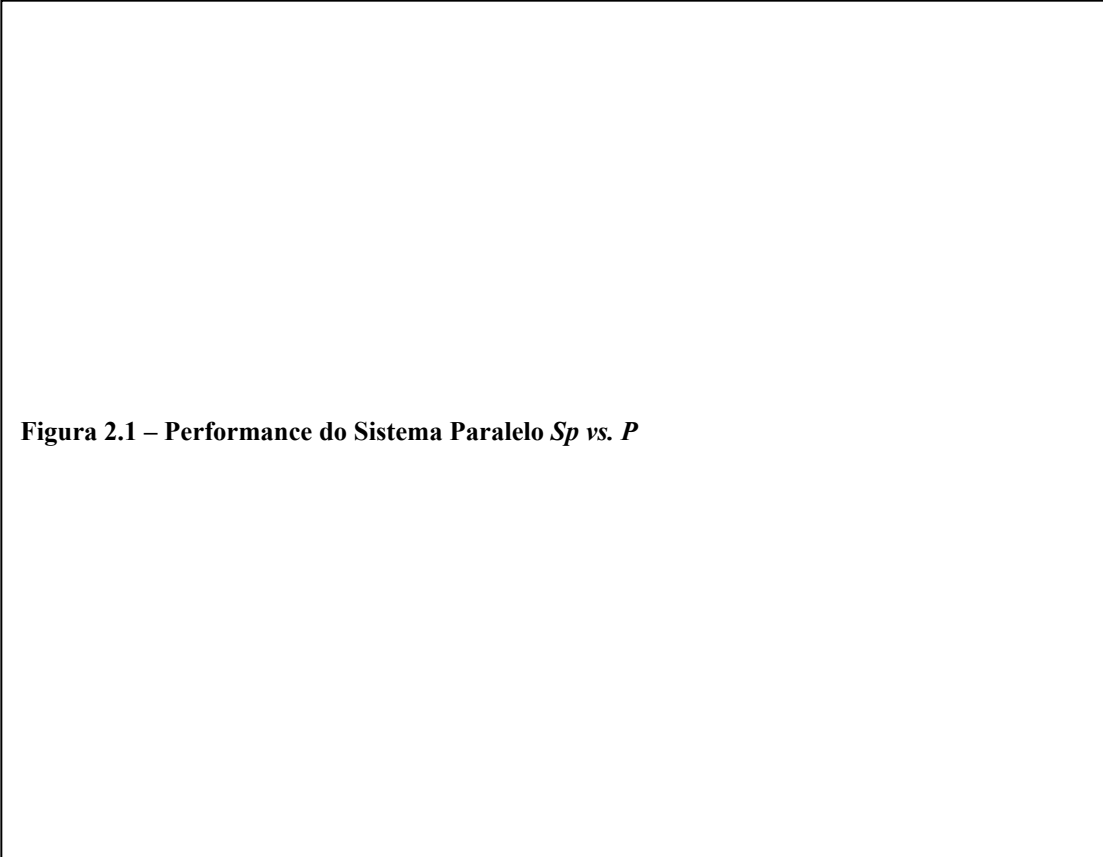


Figura 2.1 – Performance do Sistema Paralelo Sp vs. P

O speedup não pode exceder p , para uma certa computação com p -processadores em paralelo, ao menos se houver mudança na computação ou outros recursos arquiteturais são alterados (por exemplo, memória por processador).

Analisando a figura 2.1, surgem as seguintes perguntas:

- Até quando a performance pode aumentar à medida que se aumenta o número de processadores?
- Para qual valor de p a performance cai demais abaixo da linha ideal?
- Que mudanças podem ser feitas em um sistema paralelo para melhorar a curva performance vs. Número de processadores para um dado sistema?

Primeiramente, designers podem alterar o hardware e a arquitetura do sistema para melhorar a performance. Uma alternativa são softwares melhores que podem ser utilizados. Finalmente, Os usuários podem escrever programas melhores e usar melhores algoritmos e assim, explorar melhor o hardware e com isso, produzir um melhor desempenho.

2.3 Sp vs. Tipo de Computação

Considerando uma variedade de aplicações sendo resolvido em um dado sistema paralelo com um número fixo de processadores, pode-se observar o nível de desempenho que pode ser obtido. No momento, será evitado variar o programa e a estrutura de dados usada para resolver um dado problema para simplificar a análise. A variação será feita apenas no tamanho dos dados, n , para um dado programa e estrutura de dados.

Com essa restrição, fica intuitivamente claro que para um dado número de processadores P , para um pequeno tamanho de dados não há um speedup próximo de P . À medida que o tamanho do dado aumenta, a performance aumenta. A figura 2.2 mostra o gráfico produzido a medida que o tamanho do dado aumenta.

Figura 2.2 – Sp vs. n

2.4 Sp vs. P

Sendo o tempo necessário $T(i)$ para uma determinada computação utilizando i processadores, é possível definir a medida padrão do speedup utilizando P processadores:

$$Sp = T(1) / T(P) \quad (2.1)$$

Assume-se que nenhuma característica do sistema é variada a não ser o número de processadores. Apesar de muitos problemas serem desconsiderados para analisar o speedup, é possível observar que o speedup em um sistema real obedece à equação 2.1 e tem como característica a figura 2.1.

Na figura 2.1, foi mostrado dois pontos de interesse na curva. O **ponto de retorno reduzido** demonstra o ponto onde se adiciona mais processadores e se obtém pouco aumento na performance. O **ponto sem retorno** é o ponto onde se adiciona mais processadores, porém não há aumento de performance.

2.5 Sp vs. N

Neste tópico, será considerado que o número de processadores é constante e o tamanho do dado n é variado para um dado programa. Pode-se assim obter a seguinte equação:

$$Sp(P,n) = T(1,n) / T(P,n) \quad (2.2)$$

Através dessa equação é possível descobrir e melhorar a performance do hardware e software de um dado sistema quando sua curva (SP,n) – figura 2.2 – é comparada com outros sistemas para algum programa. Quando múltiplas computações são comparadas, pode-se descobrir softwares para melhorar o sistema como um todo.

Utilizando uma máquina com processadores vetoriais, através da figura 2.2 é possível observar que a performance aumenta quando o pipeline está em sua fase inicial até quando o sistema paralelo é saturado. Neste ponto os processadores estão acessando a memória cachê para a maioria dos dados. Quando o valor de n aumenta ainda mais, o sistema começa a aumentar o número de acessos a memória principal até que, finalmente, o tamanho do dado excede a memória principal e assim, começa a acessar o disco o que degrada consideravelmente a performance do sistema.

Para aumentar a performance de um determinado sistema com p processadores vetoriais utilizando dados de tamanho pequenos ou grandes, é necessário que o software seja capaz de aumentar a performance.

Para dados pequenos, para explorar totalmente o sistema paralelo, mais códigos paralelos devem ser encontrados até que o grau de paralelismo seja suficiente para saturar os processadores em paralelo. Para tipo de dados grandes, mais referência de dados locais devem ser encontrados para evitar acessos ao disco, ou seja, o fluxo de comutação deve ser modificado de modo que o espaço de endereçamento seja mais ou menos uniforme.

2.6 Limitações na Performance

Performance em sistemas paralelos é limitada por diversos fatores arquiteturais e de softwares. A computação envolve um dado particular e um determinado código. Juntos, pode-se determinar um γ que corresponde a porcentagem que deve ser executado paralelo e o restante $(1 - \gamma)$ será executado seqüencialmente. Com isso, utilizando o speedup (equação 2.1) pode-se chegar na equação 2.3:

$$Sp = \frac{T(1)}{\gamma T(1)/(1 - \gamma)T(1)} = \frac{P}{\gamma + (1 - \lambda)P} \quad (2.3)$$

Essa relação é conhecida como Lei de Amdahl's que é uma das leis mais fundamentais no estudo de sistemas paralelos. A relação entre o speedup e γ é mostrada na figura 2.3.

Figura 2.3 – Limite de speedup da lei de Amdahl

Analisando a figura 2.3, é possível verificar as seguintes limitações:

1. Para uma dada computação, o speedup máximo tem um limite superior de $1 / (1 - \gamma)$. Este é o chamado gargalo seqüencial de um programa. Quando γ aumenta, o speedup aumenta proporcionalmente;
2. Para alcançar um bom speedup, é importante fazer o gargalo menor possível;
3. Quando um problema consiste das duas partes acima (γ e $1 - \gamma$), devemos fazer a maior parte executar mais rapidamente. Em outras palavras, deve-se otimizar a que é mais executada.

2.7 Efeitos de Compilador e Software

O desempenho em um sistema paralelo é extremamente sensível à estrutura do software. O compilador é o software que faz a interface entre o código fonte do usuário e o hardware, portanto, o compilador é de extrema importância no desempenho.

Os Compiladores podem ser divididos em três estágios a fim de otimizar o desempenho. O primeiro estágio é um reestruturador global que tem como objetivo fazer as transformações que são necessárias para uma certa classe arquitetural e também para certas máquinas específicas. Ele pode reestruturar o código a fim de reduzir o overhead de comunicação ou maximizar o nível de paralelismo do código.

O segundo estágio é um otimizador do código resultando gerado pelo primeiro estágio. Este estágio otimiza o código especificamente para o sistema.

O último estágio gera o executável em código de máquina. O resultado esperado é o de máxima performance para o sistema. Pode ocorrer que o resultado seja uma baixa performance, pois pode haver deficiência no código original gerado pelo usuário, o que dificulta a otimização gerada pelo compilador. Essas deficiências podem ser removidas reescrevendo o código ou utilizando bibliotecas otimizadas.

3. Performance Speedup

Speedup e Eficiência têm sido usados largamente como medidas de desempenho. Embora há muitas dificuldades e problemas em se utilizar essas medidas, eles funcionam razoavelmente bem em muitas situações práticas. Os diversos parâmetros - a arquitetura, o software, a estrutura de dados, a estrutura do programa - afetam a performance do sistema. Todos esses parâmetros possuem diferentes valores no mundo real e com isso, pode gerar uma certa confusão na discussão da performance. Os dois principais parâmetros que limitam a performance em um sistema paralelo são: o número de processadores usados e o tamanho do dados para um determinado problema.

Neste tópico serão apresentados os limites reais do speedup, alguns nível de desempenho e tamanho de dados utilizados.

3.1 Limites Práticos do Speedup

Em um sistema paralelo real, quanto mais processadores paralelos são utilizados, mais o overhead do limitante arquitetural do sistema será notado. Essa limitação inclui o limitante de banda e a latência da transferência dos dados. O limitante do sistema de comunicação limita o potencial do speedup de qualquer computação fazendo com que a curva divirja da curva ideal conforme o número de processadores usados cresce.

Um outro problema ocorre quando se varia o tamanho do dado. Para um dado pequeno, é impossível utilizar mais do que um processador. À medida que o tamanho do dado aumenta, a performance também aumenta. Essa é a fase do sistema insaturado.

Durante a fase saturada do sistema, o limitante principal do speedup é o sistema de comunicação. Nessa fase, a medida que o dado aumenta, a performance sofre poucas modificações. Quando o dado torna-se

suficiente grande, a performance cai o que caracteriza um sistema sobrecarregado.

3.2 Níveis de Performance

Definir medidas de performance é muito mais fácil do que usá-las. Na prática, há qualitativamente níveis de performance aceitável e não aceitáveis, cuja definição depende do usuário. A Tabela 3.1 mostra os níveis de desempenho presente na literatura.

Nome	Speedup
Alta performance	$\frac{P}{2} \leq Sp \leq P$
Minima alta performance	$Sp = \frac{P}{2}$
Performance intermediária	$\frac{P}{2 \log(P)} \leq Sp \leq \frac{P}{2}$
Nível threshold	$Sp = \frac{P}{2 \log(P)}$
Performance inaceitável	$1 \leq Sp < \frac{P}{2 \log(P)}$

3.3 Tamanho dos Dados

O tamanho dos dados definidos pelo usuário pode ser dividido em três tamanhos:

- SUDS (small user-defined data size)
- LUDS (large user-defined data size)
- MUDS (medium user-defined data size)

SUDS é tipicamente para depuração durante o desenvolvimento do código. O resultado gerado por esse tipo de dado pode não ter muito significado, porém ele pode dizer se o programa está correto.

Já o LUDS pode levar mais do que uma hora para a execução completa. Serve para analisar a memória do sistema.

MUDS é um tamanho entre LUDS e SUDS. Possui um significado físico e possui execução rápida, mesmo em um sistema time-shared. Este tamanho é geralmente usado para medições.

Existem também o tamanho de dados definido pela arquitetura. Pode ser dividido em:

- SADS (small architecture-defined data size)
- MADS (medium architecture-defined data size)
- LADS (large architecture-defined data size)

Em SADS, o tamanho é definido de modo que o speedup se encontra no nível de performance. Analogamente, MADS deve ser um tamanho na qual o speedup é próximo a mínima alta performance. Em LADS, o tamanho do dado deve causar uma perda substancial na performance. Pode-se quantificar LADS como o ponto onde o speedup cai abaixo de $P/2$.

4. Estabilidade

Até agora, a discussão para medir o desempenho utilizavam-se sistemas diferentes e um único código. Para a análise da Estabilidade será utilizado um único sistema com p processadores executando um ou mais códigos.

Considerando um conjunto similar de computações que podem variar sobre K códigos e/ou variar o tamanho dos dados N_i para os códigos, pode-se definir estabilidade como sendo

$$St(P, N_i, K) = \min \text{perf}(P, N_i, K \text{ codes}) / \max \text{perf}(P, N_i, K \text{ codes})$$

É possível observar que:

$$0 < St \leq 1$$

Quanto mais próximo de 1, o sistema se caracteriza como sendo estável. Mais próximo de 0, um sistema instável.

Se houver variação apenas no código, e utilizar apenas um dado de entrada, obtém-se a função **estabilidade de programa**:

$$St(P, K) = \min \text{perf}(P, K) / \max \text{perf}(P, K)$$

Pode-se também variar apenas os dados de entrada utilizando apenas um determinado código. Com isso, é obtido a função **estabilidade de dados**:

$$St(P, K) = \min \text{perf}(P, n) / \max \text{perf}(P, n)$$

A partir dessas funções, são definidos parâmetros que caracterizam se um sistema é estável ou não. Um bom parâmetro π para definir um sistema estável é:

$$\frac{1}{6} = \pi \leq St(P, K)$$

Sistemas serão considerados estáveis caso a função estabilidade está entre 1/6 e 1. Sistemas instável são aqueles que estão abaixo de 1/6.

A seguir algumas medidas de estabilidade realizadas:

D é c a d a	Sistema	Perfect Baseline Instability
	VAX 780	5
	DEC 6000- 410	4.7
	STARDE NT 3010	5.1
	SUN SPARC 2	5.37
	IBM RS6000	5

Tabela 1 – Perfect Workstation Instability

Workstation (1993)	SPECratio Instability
DEC 3000 Model 500X	7.77
HP 9000 Model G/H/160	3.70
IBM RS 6000- POWERstation 370/375	4.84
Motorola Series 900	5.19
SGI INDIGO 2	4.56
SUN SPARCstation 10 Model 40	4.16

Tabela 2 – SPEC Workstation Instability

5. Conclusão

Nesta monografia foram discutidos métodos de media de desempenho e os problemas relacionados. Foi possível observar as dificuldades para se obter uma media consistente que reflita de forma real o comportamento do sistema paralelo. Foram apresentados os fatores que afetam a performance do sistema e as razões pela qual não se consegue obter um sistema paralelo ideal.

Foi ressaltada a necessidade de se obter uma boa media de desempenho, pois esta poderá ser utilizada para melhorar o sistema como um todo. Enfim, foi apresentado o conceito de estabilidade e como medir e classificar se um sistema é estável ou não.

6. Referências Bibliográficas

CORTÉS, O. A. C. [Desenvolvimento e Avaliacao de Algoritmos Numericos Paralelos.](#)
Dissertação (Mestrado). ICMC-USP, São Carlos, 1999.

Francês, C. R. L. Statecharts Estocásticos e Queuing Statecharts: Novas Abordagens para
Avaliação de Desempenho Baseadas em Especificação Statecharts. Tese (Doutorado).
ICMC-USP, São Carlos, 2001

Kuck, David J. High Performance Computing. Primeira Edição, Oxford, 1996.

SANTANA, M.J.; SANTANA, R.H.C.; FRANCÊS, C. R. L.; ORLANDI, R.C. Tools and
Methodologies For Performance Evaluation of Distributed Systems – A Comparison Study.
In The Proceedings of the: SUMMER COMPUTER SIMULATION CONFERENCE,
Arlington, Virginia, 1997. *Proceedings*. Arlington, p. 124-28, 1997.