USO DA TÉCNICA STU PARA AVALIAÇÃO DO DESEMPENHO DE BIBLIOTECAS INTERVALARES[1]

Rafael Linden Sagula[2]
Tiarajú Asmuz Diverio[3]
João Cesar Netto[4]
Instituto de Informática e CPGCC da UFRGS
Caixa Postal 1506491501-970 - Porto Alegre - RS - Brazil
e-mail {sagula, diverio, netto}@inf.ufrgs.br

ABSTRACT:
In this paper, we present the STU (Standard Time Unit) technique, which is used with performance avaluation of interval software packages. Comparisons were done using a classic aproach to benchmarks and the results have been showed. Interval libraries are evaluated using the STU method and the obtained results right the problems caused by the usage of different environments.

KEYWORDS:
Standard Time Unit, benchmarking, Performance evaluation, interval software.

INTRODUÇÃO

A evolução dos supercomputadores está vinculada à necessidade de resolver problemas como da Computação Numérica, que cada vez necessitam de mais recursos computacionais, principalmente velocidade de processamento, para que o trabalho possa ser concluído em tempo razoável ([HWA84]).

Em 1980, a IEEE propôs um padrão de aritmética binária de ponto-flutuante. Kulisch, em 1981 ([KUL81]), propôs uma aritmética de alta exatidão composta das padronizações do IEEE754, dos arredondamentos direcionados, do produto escalar ótimo e do somatório escalar ótimo. Nessa aritmética, os valores aproximados só diferem do valor exato por um arredondamento. Se for acrescentado à aritmética de alta exatidão a matemática intervalar e algoritmos de inclusão intervalar com a convergência justificada pelo Teorema do Ponto-Fixo de Brouwer, pode-se transferir para o próprio computador a tarefa de verificar o resultado.

AVALIAÇÃO DE DESEMPENHO

Existem atualmente várias bibliotecas matemáticas que implementam a aritmética intervalar e de alta exatidão, bem como o produto escalar ótimo e a soma ótima. Entre os principais softwares intervalares destacam-se Clemmeson, Intlib, Profil/Bias, C-XSC, Pascal-XSC e libavi.a. Essa ú ltima foi desenvolvida pelo GMC do II/UFRGS. Assim, surge a necessidade de avaliar o desempenho desses pacotes.

Entretanto, os mesmos foram implementados em ambientes totalmente heterogêneos e, por isso, a simples aplicação de um benchmark não fornece resultados satisfatórios. Isso se

1 Trabalho desenvolvido como projeto de diplomação no Bacharelado em Ciência da Computação [SAG97]
2 Bacharel em Ciência da Computação pela UFRGS e mestrando do CPGCC
3 Professor orientador no trabalho de diplomação e na bolsa de pesquisa
4 Co-orientador no trabalho de diplomação

deve ao fato de que alguns segundos de computação em uma máquina não representam a mesma quantidade de processamento em outra diferente.

A técnica proposta consiste em colocar dentro de uma mesma escala valores diferentes, permitindo assim sua comparação. Para isso, é introduzido o conceito de STU (ou Standard Time Unit), que provê essa escala padrão para os benchmarks. Os resultados obtidos mostram-se satisfatórios e algumas conclusões importantes podem ser retirados do trabalho.

AVALIAÇÃO DO DESEMPENHO DE SOFTWARE INTERVALAR

Um importante trabalho realizado no sentido de medir o desempenho de pacotes intervalares foi realizado pelo professor G. F. Corliss ([COR93]) em conjunto com diversos fornecedores desse tipo de *software*. O trabalho consiste em 5 pequenos programas que são reescritos para aproximadamente 15 bibliotecas e linguagens diferentes a fim de testá-los quanto à qualidade dos resultados e tempo de processamento. Os programas estão divididos da seguinte forma:

- Programa 1: Exercita +, −, * e /.
- Programa 2: Exercita as funções elementares.
- Programa 3: Exercita operações com matrizes e vetores.
- Programa 4: Método de Newton intervalar com uma variável.
- Programa 5: Método de otimização global com uma variável.

O trabalho de Corliss está todo baseado na premissa que as máquinas utilizadas são iguais em termos de desempenho. Para os pacotes analisados até aqui isso pode ser verdade, entretanto para a biblioteca *libavi* isso não pode ser assumido, uma vez que essa foi desenvolvida no ambiente do supercomputador Cray e as demais em microcomputadores 486 de 50MHz e estações SparcServer 330. Dessa forma, o desempenho da biblioteca fica extremamente beneficiado pelo alto desempenho da máquina.

O *benchmark* proposto nesse trabalho é feito em três etapas diferentes. A primeira consiste em medir o desempenho dos pacotes utilizando os 5 programas-teste. Na segunda etapa, o *benchmark* Linpack é utilizado para medir o desempenho da máquina. E, por fim, normaliza-se os primeiros resultados com os demais. Isso permite que os resultados possam ser analisados independentemente da máquina.

Benchmark Intervalar STU

O ponto principal do benchmark é a obtenção de uma unidade de tempo padrão ou STU (*Standard Time Unit*) para que os tempos não sejam mais medidos em segundos e sim nessa unidade.

A metodologia utilizada para o *benchmark* intervalar foi a seguinte: extrair os tempos de execução dos 5 programas (**T[k]**, onde $1 \le k \le 5$) e também o melhor resultado obtido com o *benchmark* Linpack (L). Calcular o STU para 10^6 operações de ponto-flutuante (op), seguindo a expressão: **STU = (10^6*op)/L**. Com isso, pode-se calcular o valor normalizado (N[i], onde $1 \le i \le 5$) para cada T, da seguinte forma: **N[i] = T[i]/STU**, para $1 \le i \le 5$.Simplificando as expressões acima, obtém-se que **N[i] = T[i] / ((10^6 *op)/L)** = (L*T[i])/(10^6*op), para $1 \le i \le 5$.

As unidades utilizadas foram segundos, para o STU e T; e *FLOPS* para o L. Entretanto, se for utilizado *MFLOPS* para o valor de L, o que é mais comum, pode-se simplificar ainda mais as expressões acima, de forma a chegar na expressão:

$N[i] = L*T[i]/op$, para $1 \leq i \leq 5$.

O valor de N pode ser considerado como uma taxa e não possui unidade. Se isso for considerado, pode-se suprimir a divisão por op e a expressão fica ainda mais simples.

$N[i] = L*T[i]$, para $1 \leq i \leq 5$.

Dessa forma, os resultados do *benchmark* intervalar não podem ser apresentados isoladamente, mas com o valor de L utilizado e a origem do mesmo (quem o mediu). Isso possibilita uma análise melhor dos resultados.

TESTES E RESULTADOS

Na TABELA 1, são apresentados os valores de T (tempo de execução em segundos) para cada um dos pacotes testados por Corliss e também das bibliotecas Profil e *libavi*.

TABELA 1 – Valores de T para algumas bibliotecas intervalares (em segundos).

Pacote	Test 1	Test 2	Test 3	Test 3a	Test 4	Test 5
Clemmeson PC, MS Fort 7.0	19,99	–	–	12,58	–	–
INTLIB PC, MS Fort 7.0	89,86	89,86	–	13,73	142,48	288,8
INTLIB Sparc 1+, f77	35,58	74,96	–	11,65	110,71	221,71
C-XSC PC, Borl C++ 3.1	52,34	56,85	78,64	25,65	27,95	50,15
Pascal-XSC PC, Borl C++ 3.1 Win	40,65	74,26	106,8	32,9	50,7	81,74
Pascal-XSC SparcServer 330, gcc	32	341,8	61,3	24,5	122,7	178,1
Profil/BIAS SparcServer 330, gcc	8,5	12,2	27,8	1,4	5,5	5,4
Libavi, F90, Cray Y-MP2E/232	4,57	0,16	11,72	–	0,22	–

Os valores de L para essas duas máquinas foram retirados de [DON97]. O valor de L para o supercomputador Cray é 604, para o 486 é 2 e para a Sun é 2,5. Com esses dados em mãos, pode-se calcular a versão normalizada da TABELA 1. Para isso, basta utilizar a expressão matemática da seção anterior. Com isso, obtém-se os valores apresentados na TABELA 2.

TABELA 2 – Valores normalizados do *benchmark*.

Pacote	1	2	3	3a	4	5
Clemmeson PC, MS Fort 7.0	39,98	–	–	25,16	–	–
INTLIB PC, MS Fort 7.0	179,72	179,72	–	27,46	284,96	577,6
INTLIB Sparc 1+, f77	88,95	187,4	–	29,125	276,775	554,275
C-XSC PC, Borl C++ 3.1	104,68	113,7	157,28	51,3	55,9	100,3
Pascal-XSC PC, Borl C++ 3.1 Win	81,3	148,52	213,54	65,8	101,4	163,48
Pascal-XSC SparcServer 330, gcc	80	854,5	153,25	61,25	306,75	445,25
Profil/BIAS SparcServer 330, gcc	21,25	30,5	69,5	3,5	13,75	13,5
Libavi, F90, Cray Y-MP2E/232	2760,28	96,64	6903,72	–	132,88	–

Quanto menor o valor, melhor é a biblioteca, pois são valores de tempo de execução (mesmo que normalizados). O *benchmark* apresenta a característica interessante de inverter os resultados de uma etapa para outra. Em geral, bibliotecas que executam em máquinas muito boas perdem muitas colocações se os seus resultados não acompanharem o bom desempenho da máquina sobre a qual estão rodando. Esse fato pôde ser observado com facilidade nesse

trabalho, pois a máquina Cray apresenta um desempenho muitas vezes superior às demais, entretanto a biblioteca *libavi* não acompanha essa relação. Assim, a biblioteca Libavi, que apresentou-se como a mais rápida em todos os teste na Tabela 1, foi a pior na Tabela 2, onde o pacote Profil mostrou-se como o melhor.

A principal vantagem desse método é que, ao final, pode-se ter uma noção do aproveitamento da máquina que cada biblioteca alcançou e isso independe da máquina sobre a qual o benchmark está executando.

CONCLUSÕES

Nesse trabalho, foram mostrados os aspectos práticos de avaliação de desempenho com enfoque para benchmarks. O objetivo principal, que é criar um mecanismo padrão de avaliação de desempenho de software intervalar, foi apenas iniciado, uma vez que um padrão somente pode ser considerado como tal se for amplamente aceito.

O benchmark criado mostrou-se bastante útil, pois suas várias fases permitem que análises com objetivos diferentes possam ser feitas em cima do mesmo conjunto de resultados e, principalmente, sem que seja necessário executar repetidas vezes a bateria de testes. A idéia de aproveitamento dos recursos disponíveis na máquina reflete-se bem na última fase do teste e ajuda o usuário preocupado com a relação custo/benefício.

Como trabalho futuro, pode-se usar outros benchmarks mais representativos que o Linpack (tal como o SPEC, por exemplo) com o objetivo de comparar a fidelidade dos resultados. Da mesma forma, a tecnica aqui apresentada não precisa necessariamente restringir-se a programas intervalares e pode ser perfeitamente aplicada a qualquer outro tipo de software ou biblioteca.

BIBLIOGRAFIA

[COR93] Corliss, G. F.: **Comparing *Software* Packages for Interval Arithmetic**. Preprint presented at SCAN'93, Vienna, 1993.

[DON97] DONGARRA, J. J.: **Performance of Various Computers Using Standard Linear Equations Software**. Disponível em http://www.netlib.org/ benchmark/performance.ps (01/Nov/1997)

[HWA84] HWANG, KAI; BRIGGS FAYE ALAYE. **Computer Architecture and Parallel Processing**. New York: McGraw Hill, 1984. 846p.

[KUL81] KULISCH,U.; MIRANKER,W.L **Computer arithmetic in theory and practice**, 1981.

[WEI91] Weiker, R. P.: A detailed look at some popular benchmarks. Parallel **Computing**, N.17 (1991), p.1153-1172.

[SAG97] Sagula, R. L.: Avaliação de Desempenho de Bibliotecas Intervalares. Porto Alegre; II da UFRGS, 1997. Trabalho de Diplomação. 52f.