O Efeito da Compressão de Páginas no sistema DSM PULSAR

Mario Donato Marino, Edward D. M. Ordonez e Sérgio T. Kofuji [1]

Departamento de Computação e Sistemas Digitais
Escola Politécnica da Universidade de São Paulo
Av. Prof. Luciano Gualberto, 158-Trav.-03
Cidade Universitária -São Paulo - SP
CEP - 05508 - 900

RESUMO

PULSAR é o primeiro DSM (Distributed Shared Memory System) implementado em software que além de utilizar a Consistência de Liberação, utiliza também o algoritmo de compressão de Huffman para minimizar a quantidade de dados transmitida para a manutenção da consistência. Está sendo avaliado com dois programas aplicativos: uma Eliminação de Gauss e o Water (SPLASH I). Os resultados ao se adotar o PULSAR com compressão são muito promissores: para o aplicativo Gauss, houve uma redução de 91% dos dados transmitidos pela rede e o *speedup* atingiu o máximo de 17 %, ambos comparados ao mesmo programa com o PULSAR sem compressão. Para o Water houve uma redução máxima de 30% dos dados transmitidos e um *speedup* de cerca de 3%, comparados a versão PULSAR sem compressão.

ABSTRACT

PULSAR is the first DSM Software (Distributed Shared Memory System) that besides using the Release Consistency, it uses the Huffman algorithm of Compression in order to minimize the amount of data transmitted to maintain the consistency. It is being evaluated with two applications programs: Gauss Elimination and Water (SPLASH I). The results of adopting PULSAR with compression are very promissory: for the Gauss application, there was a reduction of 91% of data transmitted through the net and the *speedup* achieves the maximum of 17% when compared to the same program running with PULSAR without compression. For the Water there was a maximal reduction of 30 % of data transmitted and the *speedup* was about 3% comparing to the PULSAR version without compression.

[1] Pesquisadores do Laboratório de Sistemas Integráveis (LSI/EPUSP)
E-mail: (mario, edmoreno, kofuji)@lsi.usp.br

1) Introdução

Um sistema DSM (*Distributed Shared Memory*) [Stum90] é uma abstração de um espaço de endereçamento compartilhado entre os nós de uma rede de computadores (distribuída); com a adoção desta abstração, para o programador é como se ele possuísse memória compartilhada, permitindo que programas paralelos escritos para máquinas de memória fisicamente compartilhada sejam facilmente portados. Essa abstração é conhecida como DSM (*Distributed Shared Memory*), como mostrado na figura 1, onde existem vários computadores (nós) interconectados por uma rede e que não possuem sua memória fisicamente compartilhada.

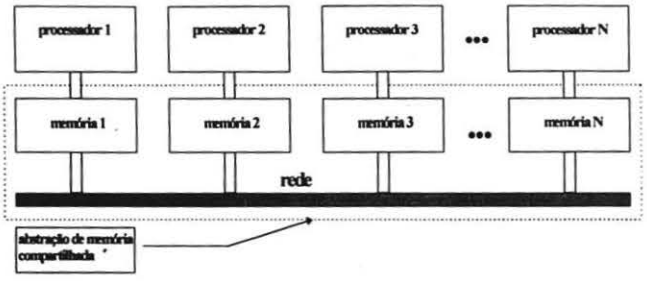

figura 1 - abstração de memória compartilhada

A adoção desta abstração [Stum90] implica em:

- localizar e acessar os dados compartilhados,
- trocar e/ou replicar os dados
- manter a consistência dos dados entre os nós da rede.
- como foi dito, deve permitir a portabilidade com os programas paralelos escritos para memória compartilhada.

Levando em conta que nos programas paralelos a sincronização é necessária para estabelecer uma coordenação entre os processos, é interessante utilizar um modelo de consistência de memória que garanta a consistência seqüencial nos pontos de sincronização do programa de modo a reduzir o número de mensagens para a manutenção da consistência. Os modelos que caem nestas características são os modelos de consistência de liberação (*release consistency*) que, portanto, **somente mantêm a consistência do programa aplicativo nos pontos de sincronização** (barreiras e semáforos).

Segundo Carter [Cart95], os sistemas de memória virtual compartilhada e distribuída **(DSM)** caracterizaram-se basicamente por duas fases distintas:

(i) por um elevado número de mensagens para a manutenção da consistência (por meio de mensagens de invalidação e atualização), utilizando principalmente a consistência seqüencial, como por exemplo o Ivy [Li86];

(ii) por uma redução drástica do número de mensagens para a manutenção da consistência por meio da adoção de técnicas como as múltiplas escritas concorrentes, mecanismo para cessar o envio de atualizações desnecessárias e principalmente a adoção da consistência de liberação que aproveita os pontos de sincronização natural dos programas paralelos, barreiras e semáforos por exemplo, para manter o sistema consistente. São exemplos da segunda fase o Munin [Cart93], o Midway [Bers91, Bers93] (consistência de entrada), o TreadMarks [Kele95] (consistência de liberação preguiçosa) e o Quarks [Apwq95 ; Cart95] (domínio público).

O sistema PULSAR [Mari95], que faz parte da segunda fase, não só procura diminuir a utilização de uma grande quantidade de dados pela minimização do número de

mensagens (para a manutenção da coerência), por meio da consistência de liberação, mas também procura diminuir a quantidade de dados transmitida utilizando-se compressão de dados, especificamente o algoritmo de Huffman [Segu89].

O sistema PULSAR é o primeiro DSM a utilizar compressão de dados para a minimização da quantidade de dados transmitida para a manutenção da consistência, que apesar de poder utilizar os protocolos de invalidação e atualização, neste estudo somente será utilizado o protocolo de invalidação. Além disso o sistema é o primeiro DSM a operar sobre uma rede de computadores pessoais (PCs) com Unix (Unix FreeBSD 2.X). O PULSAR é totalmente compatível com o TreadMarks [Kele95] e com o Quarks [Apwq95; Cart95], bastando somente a troca das primitivas no programa aplicativo.

O sistema PULSAR é avaliado com dois programas aplicativos: uma eliminação de Gauss (com condensação pivotal) e o Water, aplicativo que faz parte do conjunto SPLASH I [Sing91] de Stanford.

Tais programas aplicativos são submetidos a uma rede de computadores pessoais até 5 nós, **comparando-se o desempenho e a quantidade de dados transmitida** por cada um dos nós, com o PULSAR operando com compressão versus PULSAR sem compressão.

Portanto o objetivo principal é avaliar o efeito da compressão de dados num sistema DSM, isto é, o que acontece com a quantidade de dados transferida e quanto uma possível diminuição de quantidade de dados transferida influi no desempenho do aplicativo no sistema.

O sistema PULSAR utilizando compressão conseguiu uma diminuição do tempo de execução de 5% a 17 % no caso da Eliminação de Gauss e de até 3% no caso do Water.

Com relação à quantidade de dados transmitida, o sistema PULSAR utilizando compressão transmitiu até 91 % menos dados do que o mesmo sem compressão quando executada a eliminação de Gauss. No Water, utilizando-se o sistema PULSAR com compressão, reduziu-se a quantidade de dados transmitida em até 30%.

2) Alguns Sistemas DSM

Vai-se citar alguns sistemas DSM e como o PULSAR se encaixa dentro deles.

O primeiro sistema DSM foi o Ivy [Li86]. Ele utiliza a consistência seqüencial, o que acarreta o envio de um grande número de mensagens para a manutenção da coerência.

O primeiro sistema DSM implementado em *software*, cujos programas apresentaram desempenho compatível com os mesmos programas convertidos para passagem de mensagens foi o Munin [Cart93]. Suas características principais são:

- primeiro sistema DSM (integrado à memória virtual), que conseguiu uma redução drástica do número de mensagens para a manutenção da consistência.
- consistência de liberação (*release*), por meio do envio de mensagens de atualização;
- múltiplos protocolos de escrita, permitindo que mais de um nó escreva ao mesmo tempo em uma mesma página; as informações das modificações das páginas feitas em cada nó são armazenadas em estruturas conhecidas por *diffs* [Cart93]. Adotando esta técnica, foi o primeiro a eliminar o efeito do falso compartilhamento; portanto minimiza a quantidade de dados transmitida pela transmissão do *diff* e não de páginas.
- mecanismo de cessação do envio de mensagens de atualização que não serão utilizadas pelo nó que as recebe;
- suporta semáforos distribuídos e barreiras.
- opera em estações SUN 3/60;

O Midway [Bers91; Bers93] apresenta as seguintes características:

- a consistência de entrada (*entry consistency*), que requer a associação de cada variável compartilhada a uma variável de sincronização;
- todos os protocolos do Munin e o de consistência de entrada;
- precisa-se um sofisticado compilador para detectar modificações em variáveis;
- os aplicativos devem ser adaptados para se associar cada variável compartilhada a uma variável de sincronização;

O TreadMarks, primeiro DSM a conseguir desempenho próximo de uma máquina de memória fisicamente compartilhada [Kele95], apresenta as seguintes características:

- utiliza a consistência de liberação preguiçosa (*lazy release consistency*);
- pode utilizar protocolos de atualização, invalidação e híbrido;
- múltiplas escritas concorrentes, criação preguiçosa de *diffs*;
- transmite sempre *diffs* exceto a primeira busca (da página)
- implementado por uma biblioteca a nível de usuário;
- utiliza *sockets* e protocolo UDP para a minimização de *overheads*;
- pelo fato de utilizar a consistência de liberação preguiçosa, armazena uma grande quantidade de *diffs*, isto é muita memória implicando, em certas condições [Kele95] a se executar o processo de *garbage collection*;
- opera em estações SUN (com Unix 4.1.X) e Silicon Grafics (Irix);

O Quarks [Apwq95], DSM de domínio público, possui as seguintes características:

- utilização de consistência de liberação como a do Munim e de múltiplos protocolos de consistência quando opera com protocolo de atualização; pode operar também com invalidações;
- implementado por uma biblioteca a nível de usuário;
- utiliza um pacote de *threads* na implementação,
- utiliza *sockets* e protocolo UDP;
- opera em estações SUN (com Unix Sun OS 4.1.X) e Silicon Grafics (Irix);

O sistema PULSAR [Mari95], seguindo a tendência internacional, apresenta as seguintes características:

- consistência de liberação como a do Munin[Cart93];
- implementado em forma de biblioteca a nível de usuário;
- tanto pode operar com invalidações, como também com atualizações e, neste caso, utiliza a técnica de múltiplos protocolos de escrita do Munin [Cart93];
- utiliza sockets e protocolo TCP;
- é o primeiro DSM a utilizar compressão de páginas, visando a diminuição da quantidade de dados transmitida.
- opera em rede de estações SUN (Unix Sun OS 4.1.X);
- é o primeiro a operar em rede de PCs (Unix FreeBSD 2.X);

3) Avaliação

3.1) Programas de Avaliação

O sistema PULSAR é submetido a dois programas de avaliação: uma eliminação de Gauss e o Water (SPLASH I).

O aplicativo eliminação de Gauss é implementado com condensação pivotal e somente apresenta barreiras para a sincronização entre os processos. Este aplicativo foi executado com matrizes de tamanho 600 x 600, 900 x 900 e 1300 x 1300.

No caso do programa Water, que faz parte do conjunto de aplicativos do SPLASH I e utiliza para sincronização entre processos tanto barreiras como semáforos, os parâmetros utilizados foram:

- intervalo de tempo entre os passos TSTEP = $1{,}5.10^{-16}$,

- número de moléculas: 64, 164 e 364 moléculas,
- ordem do método corretor-previsor NORDER = 6,
- freqüência de salvamento de dados igual a 1,
- arquivo de entrada para as partículas LWI12.

3.2) Ambiente de Avaliação

Os aplicativos mensionados no item anterior são executados sobre o sistema PULSAR no protótipo do SPADE II [Kofu95], constituído por:

- nós: Pentiums 100 Mhz, 32 MB de memória e disco IDE de 1 GB;
- interconexão: placas de rede ISA padrão *ethernet* (10 Mbits/s).

Para a obtenção das métricas deste estudo, a rede de interconexão constituinte do SPADE II foi totalmente isolada de qualquer rede externa.

O sistema operacional utilizado é o Unix FreeBSD 2.X. Os aplicativos são executados e os resultados são extraídos com o PULSAR operando em 3, 4 e 5 nós.

4) Métricas de Avaliação

Neste estudo são obtidas as seguintes métricas com e sem compressão: quantidade total de bytes transmitida por cada nó, constituída pela quantidade de bytes correspondentes ao total de pedidos de página, quantidade de bytes correspondentes ao total de páginas enviadas por cada nó, quantidade de bytes correspondentes ao total de mensagens de coerência enviados por cada nó e quantidade de bytes ao total de mensagens de sincronização enviados por cada nó.

Os resultados a serem discutidos neste trabalho são calculados a partir das seguintes métricas: *(i)* o *speedup* (s) (de um nó) que consiste na razão entre a diferença entre o tempo de execução do programa utilizando o PULSAR sem compressão e o tempo de execução do mesmo programa utilizando o PULSAR com compressão dividido pelo tempo de execução do aplicativo sem compressão; *(ii)* a razão de compressão de páginas (cp) (de um nó) que consiste na razão entre a diferença entre quantidade de páginas (em bytes) transmitida utilizando o PULSAR sem compressão e utilizando o PULSAR com compressão dividido pela quantidade de páginas utilizando o PULSAR sem compressão; *(iii)*a razão de compressão total (ct) (de um nó) que consiste na razão entre a diferença entre quantidade total de bytes transmitida utilizando o PULSAR sem compressão e utilizando o PULSAR com compressão dividido pela quantidade total de bytes utilizando o PULSAR sem compressão.

Observar que os parâmetros que foram medidos foram obtidos por uma média dos mesmos entre os nós constituintes da rede do sistema o PULSAR.

5) Análise dos Resultados

Conforme dito, os programas aplicativos Eliminação de Gauss e Water são submetidos ao sistema o PULSAR, sendo executados em 3, 4 e 5 nós.

Para as tabelas que se seguem neste item, logo abaixo, vale a seguinte legenda:

PSC = PULSAR sem utilizar compressão ; PCC = PULSAR utilizando compressão
tp = média da quantidade de páginas enviadas por um nó em bytes
tap = média da quantidade de bytes utilizados nos pedidos de página
tc = média da quantidade de bytes de utilizados na consistência por um nó
ts = média da quantidade de bytes de sincronização utilizados por um nó
tt = total de bytes transmitidos por um nó;

$$s = \text{speedup};\ s = \frac{\text{tempo de execução do programa no PSC} - \text{tempo de execução do programa no PCC}}{\text{tempo de execução do programa no PSC}} \times 100\%$$

ds = *speedup* mínimo a *speedup* máximo;

$$ct = \frac{tt(PSC) - tt(PCC)}{tt(PSC)} \times 100\% \; ; \qquad cp = \frac{tp(PSC) - tp(PCC)}{tp(PSC)} \times 100\%$$

Observar que tt **não** é igual à soma de tp + tc + ts, isto porque algumas mensagens funcionam tanto para manter a coerência, como também para sincronização, implementado deste modo para minimizar o número de mensagens.

5.1) Eliminação de Gauss

tamanho	**600 x 600**		**900 x 900**		**1300 x 1300**	
	PSC	PCC	PSC	PCC	PSC	PCC
tp	2130000	48000	3535000	75000	5800000	123416
tap	4500	4500	7500	7800	11500	11600
tc	180000	180000	537000	528000	1730000	1650000
ts	140000	140000	483000	485000	1650000	1580000
tt	2300000	245000	4080000	630000	7600000	1800000

tabela 1 - Gauss - 3 nós

	600 x 600	**900 x 900**	**1300x1300**
ds	7 a 8	5 a 8	5 a 7
cp	97,8	97,9	97,9
ct	89,3	84,6	76,3

tabela 2 - Gauss - 3 nós

5.1.1) PULSAR com 3 nós

Como se pode observar nas tabelas 1 e 2 a quantidade de dados transmitida por cada nó (observando **tt** e também **tp**) reduziu-se drasticamente quando foi utilizada a compressão de dados. A razão **cp** foi de aproximadamente 98%, uma redução excepcional, praticamente eliminando a quantidade de dados referentes à transmissão de páginas. A razão **ct** foi de 76% (1300x1300) a 89% (600x600), o que novamente é uma redução enorme da quantidade total de dados transmitida. Pode-se notar também que não se utilizando compressão, a quantidade maior de dados que passava na rede era devido às invalidações de página (76 a 93% do total transmitido), ao passo que utilizando compressão, agora a quantidade maior de dados é constituída por mensagens de sincronização e consistência (76 a 93% do total transmitido) e as páginas passando a constituir apenas 7 a 20% do total.

Com respeito ao *speedup* (**ds**), para a matriz de entrada de 600x600, 900x900 e 1300x1300, conseguiram-se *speedups* da ordem de 8%, 7% e 6%, respectivamente (vide tabela 2). Comparando-se estes desempenhos, acredita-se que para matrizes maiores que 1300 x 1300 o aumento de desempenho seja da mesma ordem (6 a 8%). Pode-se justificar que o aumento de desempenho não foi maior, pois acredita-se que os fatores limitantes do desempenho são a rede utilizada (*ethernet* de 10 Mbits/s) e a quantidade de mensagens, embora se utilize a consistência de liberação (só transmitindo as mensagens para a manutenção da consistência durante as operações de sincronização).

tamanho	**600 x 600**		**900 x 900**		**1300 x 1300**	
	PSC	PCC	PSC	PCC	PSC	PCC
tp	2390000	58000	3980000	87000	6540000	138000
tap	5000	5000	8900	8800	15400	13100
tc	180000	172000	456000	440000	1360000	138000
ts	109000	115000	368000	373000	1200000	1250000
tt	2580000	247000	4450000	553000	7920000	1500000

tabela 3 - Gauss - 4 nós

	600 x 600	900 x 900	1300x1300
ds	7 a 15	4 a 11	5 a 10
cp	97,6	97,8	97,9
ct	90,4	87,6	81,0

tabela 4 - Gauss - 4 nós

A utilização da compressão não aumentou o tempo de execução porque ela é suficientemente rápida se comparada ao tempo de espera por uma página que foi pedida (3 a 4 ms) ou ao envio e confirmação do recebimento de um *diff*, como também é rápida se comparada ao tempo de espera nas barreiras.

A redução excepcional da quantidade de dados transmitida (tabelas 1 e 2) pode ser justificada porque o algoritmo de Huffman encontrou muitas cadeias de *strings* semelhantes dentro das páginas, isto é dentro das matrizes a serem eliminadas (Gauss).

Também observa-se que à medida que os tamanhos das matrizes aumentam, aumenta também a quantidade total de dados transmitida por cada nó (**tt** na tabela 1).

5.1.2) PULSAR com 4 nós

Analogamente ao item 5.1.1, como se pode observar pelas tabelas 3 e 4, reduziu--se drasticamente a quantidade de dados transmitida por cada nó (**tt** e também **tp**) quando foi utilizada a compressão de dados.

A razão **cp** foi de aproximadamente 98%, uma gigantesca redução, da mesma ordem que para 3 nós, praticamente eliminando a quantidade de dados referentes à transmissão de páginas. A razão **ct** foi da ordem de 81% (1300x1300) a 91% (600x600), o que novamente é uma redução enorme da quantidade total de dados transmitida, ficando também próxima às **cts** para 3 nós. Com relação à quantidade de dados transmitida, isto é, não se utilizando compressão, a quantidade maior de dados que passava na rede era devido às invalidações de página (83 a 93% do total transmitido), ao passo que utilizando compressão, agora a quantidade maior de dados é constituída por mensagens de sincronização e consistência (75 a 90% do total transmitido) e as páginas passando a constituir apenas 9 a 23% do total.

Na tabela 3 pode-se observar um aumento de desempenho (**ds**) da versão com compressão em relação à versão sem compressão da ordem de 4 a 15%.

A observação do item 5.1.1 com relação ao aumento do tamanho das matrizes também é valida aqui, isto é, à medida que se aumenta o tamanho das matrizes, aumenta-se a quantidade de dados total transmitida por cada nó.

tamanho 600 x 600			**900 x 900**		**1300 x 1300**	
	PSC	PCC	PSC	PCC	PSC	PCC
tp	2550000	67000	4450000	98500	6980000	153000
tap	5300	5300	9000	9000	14000	14000
tc	180000	170000	433000	426000	1150000	1090000
ts	88000	94000	294000	302000	963000	985000
tt	2740000	251000	4700000	550000	8160000	1200000

tabela 5 - Gauss - 5 nós

	600 x 600	**900 x 900**	**1300x1300**
ds	4 a 7	7,8 a 17	10 a 14
cp	97,6	97,8	97,8
ct	90,8	88,3	85,3

tabela 6 - Gauss - 5 nós

5.1.3) PULSAR com 5 nós

Analogamente aos itens 5.1.1 e 5.1.2, observando-se as tabelas 5 e 6 (**tt** e **tp**), reduziu-se drasticamente a quantidade de dados transmitida por cada nó quando utilizada a compressão de dados.

A razão **cp** foi de aproximadamente 98%, uma redução excelente, da mesma ordem que para 3 e 4 nós, praticamente eliminando a quantidade de dados referentes à transmissão de páginas. A razão **ct** foi de 85% (1300x1300) a 91% (600x600), o que novamente é uma redução enorme da quantidade total de dados transmitida, ficando também próxima às **cts**

para 3 e 4 nós. Também vale a mesma observação dos itens 5.1.1 e 5.1.2, isto é, não se utilizando compressão, a quantidade maior de dados que passava na rede era devido às invalidações de página (86 a 95% do total transmitido), ao passo que utilizando compressão, agora a quantidade maior de dados é constituída por mensagens de sincronização e consistência (71 a 86% do total transmitido) e as páginas passando a constituir apenas 13 a 27% do total. Observando a tabela 5, percebe-se um aumento de desempenho (**ds**) da versão com compressão em relação à versão sem compressão da ordem de 4 até 17%.

A observação do item 5.1.1 e 5.1.2 com relação ao aumento do tamanho das matrizes também é valida aqui, portanto, à medida que se aumenta o tamanho das matrizes, aumenta-se a quantidade de dados total transmitida por cada nó.

5.1.4) Comparação entre 3, 4 e 5 nós

Com relação à quantidade de dados transmitida, comparando-se as tabelas de 1 a 6, observa-se que quanto mais nós, maior a quantidade de dados a ser transmitida (páginas, sincronização e consistência) pois mais nós podem utilizar as mesmas páginas, sendo necessária a manutenção da consistência entre eles, portanto um aumento do número de mensagens, conseqüentemente de dados trafegados pela rede.

Observando-se as tabelas 1, 3 e 5, para o mesmo tamanho de matriz de eliminação, para o PULSAR com e sem compressão, a quantidade em bytes de páginas transmitidas e a quantidade total transmitida por cada nó, mantiveram-se mais ou menos as mesmas (cerca de 10 a 20%) à medida que o número de nós aumentou.

Comparando-se os *speedups* (**ds**), tabelas 2, 4 e 6, percebe-se que o *speedup* manteve-se sempre da ordem de 4 a 15 %, à medida que se aumenta o número de nós, levando-se a intuir que ainda aumentando o número de nós o *speedup* deve permanecer nesta faixa.

5.2) Water (SPLASH I)

	64 moléculas		164 moléculas		364 moléculas	
	PSC	PCC	PSC	PCC	PSC	PCC
tp	4100000	3100000	16000000	12600000	44000000	34500000
tap	6300	6400	27000	26000	75000	76000
tc	204000	201000	1480000	1450000	6100000	6200000
ts	168000	167000	1300000	1300000	5560000	5700000
tt	4460000	3440000	18200000	15400000	55800000	48000000

tabela 7 - Water - 3 nós

	64 moléculas	164 moléculas	364 moléculas
ds	0	1 a 3	1 a 3
cp	24,4	21,2	21,6
ct	22,9	15,4	14,0

tabela 8 - Water - 3 nós

5.2.1) PULSAR com 3 nós

Como se pode observar pelas tabelas 7 e 8 a quantidade de dados transmitida por cada nó (observando **tt** e também **tp**) reduziu-se quando foi utilizada a compressão de dados. A razão **cp** foi de aproximadamente 21 a 24%, uma boa redução. A razão **ct** foi da ordem de 14 a 23%. Ao contrário da eliminação de Gauss, onde utilizando a compressão a quantidade de dados transferida predominante é de consistência e sincronização, no caso do Water, mesmo utilizando compressão, ainda a quantidade maior de dados que trafega pela rede continua sendo de páginas, em torno de 80 a 90% tanto com, como sem compressão.

Observando-se a tabela 8, o parâmetro **ds** (diminuição do tempo de execução do aplicativo) do sistema PULSAR com compressão foi da ordem de 0% a 3% melhor do que o sistema o PULSAR sem compressão, *speedups* menores que os obtidos a eliminação de Gauss, já que o algoritmo de Huffman não conseguiu resultados tão bons como quando aplicado à eliminação de Gauss, e também levando-se em consideração que o Water é um aplicativo de granularidade fina e que apresenta muita sincronização.

Como dito anteriormente, acredita-se que o algoritmo de Huffman no aplicativo Water não tenha obtido a razão **cp** tão alto quanto na eliminação de Gauss porque não encontrou seqüências repetidas de *strings* nas páginas que ele utiliza (tipo de dados). Mas mesmo assim obteve uma redução de até 23%.

Vale o mesmo comentário feito no item 5.1.1 sobre o tamanho das matrizes, somente que, no caso do Water, aplicado ao número de moléculas, isto é, quanto maior o número de moléculas, maior a quantidade de dados transmitida por cada nó.

	64 moléculas		**164 moléculas**		**364 moléculas**	
	PSC	PCC	PSC	PCC	PSC	PCC
tp	4200000	3140000	16600000	12900000	48300000	38000000
tap	6600	6800	28000	25200	83000	81000
tc	207000	204000	1400000	1400000	6000000	6000000
ts	125000	123000	980000	980000	4700000	4200000
tt	4500000	3500000	19000000	15000000	58700000	47900000

tabela 9 - Water - 4 nós

	64 moléculas	164 moléculas	364 moléculas
ds	2 a 3	2 a 3	2 a 3
cp	25,2	22,3	21,3
ct	22,2	21,0	18,4

tabela 10 - Water - 4 nós

5.2.2) PULSAR com 4 nós

Como se pode observar pelas tabelas 9 e 10, a quantidade de dados transmitida por cada nó (**tt** e **tp**) reduziu-se quando utilizada a compressão de dados. A razão **cp** foi de aproximadamente 21 a 25%. A razão **ct** foi cerca de 18 a 22%. Também vale a mesma observação do item 5.2.1, isto é, mesmo utilizando compressão, ainda a quantidade maior de dados que trafega pela rede continua sendo de páginas.

Pela tabela 10, o desempenho **ds** (diminuição do tempo de execução do aplicativo) do PULSAR com compressão foi da ordem de 2% a 3% melhor que o PULSAR sem compressão, desempenho menor do obtido na eliminação de Gauss, já que o algoritmo de Huffman não conseguiu resultados tão bons como na eliminação de Gauss, lembrando que o Water é um aplicativo de granularidade fina e apresenta muita sincronização.

Acredita-se que o algoritmo de Huffman no aplicativo Water não tenha obtido a razão cp tão alto quanto na eliminação de Gauss porque não encontrou seqüências repetidas de *strings* nas páginas que ele utiliza (seus tipos de dados). Mesmo assim obteve uma redução de 18 a 22% na quantidade de dados transmitida por cada nó.

Também vale a observação do item 5.2.1 sobre o número de moléculas, quanto maior o número de moléculas, maior a quantidade de dados transmitida por cada nó.

	64 moléculas		**164 moléculas**		**364 moléculas**	
	PSC	PCC	PSC	PCC	PSC	PCC
tp	3960000	2870000	15100000	11500000	45800000	37000000
tap	6400	6000	28100	25000	79000	80000
tc	203000	200000	1350000	1330000	5700000	5700000
ts	101000	100000	813000	789000	3360000	3350000
tt	4270000	3170000	17300000	12650000	55000000	46100000

tabela 11 - Water - 5 nós

	64 moléculas	164 moléculas	364 moléculas
ds	0	0	2 a 3
cp	27,5	23,8	19,2
ct	25,8	26,9	16,2

tabela 12 - Water - 5 nós

5.2.3) PULSAR com 5 nós

Observando-se as tabelas 11 e 12, a quantidade de dados transmitida por cada nó (**tt** e **tp**) reduziu-se utilizando a compressão de dados. A razão **cp** foi de aproximadamente de 19 a 27,5%. A razão **ct** reduziu-se de 16 a 26%. Valem as mesmas observações do item 5.2.1 e 5.2.2, isto é, mesmo utilizando compressão, ainda a quantidade predominante (maior) de dados que trafega pela rede continua sendo de páginas.

Observando-se a tabela 10 , observou-se que o desempenho (diminuição do tempo de execução do aplicativo) do sistema o PULSAR com compressão foi no máximo 3% melhor do que o sistema o PULSAR sem compressão, desempenho menor que a eliminação de Gauss, já que o algoritmo de Huffman não conseguiu resultados tão bons como quando aplicado na eliminação de Gauss, também lembrando que o Water é um aplicativo de granularidade fina e que apresenta muita sincronização. Mas mesmo assim obteve uma redução de 16 a 26%, com relação à quantidade de dados transmitida. Também vale a observação dos itens 5.2.1 e 5.2.2 sobre o número de moléculas, isto é, maior o número de moléculas, maior a quantidade de dados transmitida por cada nó.

Observando-se as tabelas 7, 9 e 11, percebe-se que, para o mesmo número de moléculas, tanto a quantidade em bytes de páginas transmitidas, como a quantidade total transmitida por cada nó manteve-se mais ou menos a mesma, à medida que o número de nós aumentou. Esta observação também vale para as mensagens de sincronização e coerência.

5.2.4) Comparação entre 3,4 e 5 nós

Com relação à quantidade de dados transmitida, observando-se as tabelas de 7 a 12, quanto mais nós, maior a quantidade de dados a ser transmitida, tanto de dados (páginas), de sincronização e de consistência, que é perfeitamente justificado pois existem mais nós que podem utilizar as mesmas páginas, sendo necessária a manutenção da consistência entre eles, portanto um aumento do número de mensagens e conseqüentemente de dados.

Observando-se as tabelas 7, 9 e 11 percebe-se que, para o mesmo tamanho de matriz de eliminação, para o PULSAR com e sem compressão, tanto a quantidade em bytes de páginas transmitidas, como a quantidade total transmitida por cada nó mantiveram-se mais ou menos as mesmas (cerca de 10 a 20%) à medida que o número de nós aumentou.

Comparando-se os *speedups* para 3, 4 e 5 nós, eles estiveram em torno de 3%.

6) Comentários e Conclusões

A análise dos resultados obtidos permite verificar que a utilização da compressão de dados é bastante viável em sistemas DSM pois apresenta bons ganhos de desempenho e uma considerável redução da quantidade de dados transmitida, dependendo, é claro, do algoritmo de compressão utilizado.

Para o aplicativo Gauss, as razões **cp** (razão entre a diferença entre quantidade de páginas (em bytes) transmitida utilizando o PULSAR sem compressão e utilizando o PULSAR com compressão dividido pela quantidade de páginas utilizando o PULSAR sem compressão) e **ct** (razão entre a diferença entre quantidade total de bytes transmitida utilizando o PULSAR sem compressão e utilizando o PULSAR com compressão dividido pela quantidade total de bytes utilizando o PULSAR sem compressão) mantiveram-se

praticamente constantes quando variamos o número de nós, podendo intuir que aumentando ainda mais o número deles, estas razões se manterão.

Conforme visto, para dois programas aplicativos, Gauss e Water, que possuem diferentes padrões de acesso à memória, diferentes estilos de programação e formas de sincronização, o algoritmo de compressão de Huffman apresentou um bom aumento de desempenho global e excelentes resultados com relação à quantidade de dados transmitida durante a execução dos mesmos. Obtiveram-se *speedups* de até 27 % no caso do Gauss e 3 % no caso do Water. Houve uma redução da quantidade de dados transmitida de até 98% no caso do Gauss e de até 30% no caso do Water.

Como trabalhos futuros fica a sugestão de avaliar outros métodos de compressão e compará-los com o algoritmo de Huffman também submetendo novos programas, avaliando seu *speedup* e quantidade de dados transmitida com e sem compressão.

Outras possibilidades interessantes seriam fazer as mesmas medidas trocando a rede de interconexão, isto é, por exemplo uma rede *fast-ethernet* (100 Mits/s) ou uma rede ATM.

8) Referências Bibliográficas

[Apwq95] **Aplication programming with Quarks**, user manual, University of Utah, 1995.

[Bers91] Bershad N. B., Zekauskas M. J., **Midway: A Shared Memory Parallel Programming for Distributed Memory Multiprocessors**, Technical Report CMU-CS-91-170, Carnegie-Mellon University, Pittsburgh, September 1991.

[Bers93] Bershad B. N., Zekauskas M. J., Sawdon W. A., **The Midway Distributed Shared memory System**, COMPCON 1993.

[Cart93] Carter J. B., **Efficient Distributed Shared Memory Based on Multi-Protocol Release Consistency**, PHD Thesis, Rice University, Houston, Texas, September, 1993.

[Cart95] Carter J. B., Khandekar D., Kamb L., **Distributed Shared Memory: Where We are and Where We Should Be Headed**, Computer Systems Laboratory, Universisty of Utah, 1995.

[Kele95] Keleher P., **Lazy Release Consistency for Distributed Shared Memory**, PHD Thesis, University of Rochester, Texas, Houston, January 1995.

[Kofu95] Kofuji, S. T., Considerações de Projeto e Análise do SPADE - um Multiprocesssador de Larga Escala baseado no padrão ANSI/IEEE-SCI, PHD Thesis, Escola Politécnica da Universidade de São Paulo, São Paulo, February 1995.

[Li86] Li K, **Shared Virtual Memory on Loosely Coupled Multiprocessors**, PHD Thesis, Yale University, 1986.

[Mari95] Marino, M. D., **Um Sistema de Memória Compartilhada e Distribuída Baseado em Consistência de Memória Relaxada**, Exame de Qualificação, Escola Politécnica da Universidade de São Paulo, October, 1995.

[Segu89], Segura M. C. G, **Uma Análise de Técnicas Reversíveis de Compressão de Dados**, Dissertação de Mestrado, Escola Politécnica da Universidade de São Paulo, 1989.

[Sing91] Singh P. J., Weber W., Gupta A., Splash: Stanford parallel Applications for Shared-Memory. Computer Architercture News, Technical Report CSL-TR-91-469, Stanford University, April 1991.

[Stum90] Stumm M., Zhou S., **Algorithms Implementing Distributed Shared Memory, University of Toronto**, IEEE Computer, v. 23, n. 5, p. 54-64, May 1990.