Algoritmo Adaptativo LRU-WAR com Detecção de Freqüência de Acessos de Páginas

Jacinto C. A. Cansado, João Henrique de S. Pereira, Edson T. Midorikawa

Escola Politécnica da Universidade de São Paulo (EPUSP)
Departamento de Engenharia da Computação e Sistemas Digitais (PCS)
05508-900, São Paulo - SP

jacinto.cansado@poli.usp.br, joaohs@ctbc.com.br,
edson.midorikawa@poli.usp.br

Abstract. *The efficient memory usage in high performance computing systems with timesharing is a difficult challenge. Some research areas on adaptive algorithms concerning memory page replacement, analyze the memory access behavior seeking to maintain in memory the pages that will be used in a near future and discarding the others. This is important due to the high cost of treating page faults. The proposal is to analyze the influence of page access frequency on the adaptive algorithm LRU-WAR, using its structure and applying page replacement with the access frequency analysis, as a function of its execution state. The comparative performance analysis is conducted by using trace files that represent different memory access behaviors.*

Resumo. *O uso da memória de forma eficiente, em sistemas multiprogramados de alto desempenho, é um grande desafio. Nesse sentido, linhas de pesquisa por algoritmos adaptativos de substituição de páginas de memória têm sido desenvolvidas com o intuito de explorar o comportamento de acesso à memória por programas, procurando manter na memória as páginas que serão utilizadas em um futuro próximo e descartando as demais devido ao alto custo do tratamento de uma falta de página. O presente trabalho avalia a influência da freqüência de acesso a páginas no algoritmo adaptativo LRU-WAR utilizando sua estrutura como base e aplicando a substituição de página com a análise da freqüência de acesso em função de seu estado de execução. Uma análise comparativa do desempenho é realizada utilizando-se arquivos de trace que representam diferentes comportamentos de acesso à memória.*

1. Introdução

O uso da memória virtual proporciona a execução de processos sem que os mesmos estejam carregados inteiramente na memória, permitindo dessa forma que apenas os trechos de programas que estão sendo executados, em um determinado instante, estejam residentes nesta [Silberschatz 2005]. Essa técnica permite que a memória seja compartilhada por vários processos com padrões de acesso imprevisíveis, dificultando, para não se dizer impossibilitando o desenvolvimento de um único algoritmo de substituição de páginas eficiente, que explore as características temporais e espaciais de cada processo individualmente.

A política de substituição adotada tem uma forte influência no desempenho do sistema computacional, portanto, políticas de substituição eficientes estão sendo pesquisadas e desenvolvidas. Segundo [Midorikawa 2007] o uso de uma política de substituição de páginas eficiente é de fundamental importância para que um Sistema Operacional execute tarefas que exijam alto desempenho.

O algoritmo de substituição de páginas Ótimo é o de melhor desempenho, porém é impossível implementá-lo em um sistema real devido à impossibilidade de se conhecer antecipadamente qual página carregada na memória demorará mais tempo para ser utilizada, sendo, portanto, passível de substituição.

Já que é impossível essa previsão, algumas aproximações são propostas, como os algoritmos amplamente utilizado pelos Sistemas Operacionais modernos que avaliam de alguma forma o passado recente para prever o futuro próximo. Como exemplo, tem-se o algoritmo LRU (*Least Recently Used*), o qual substitui a página residente na memória acessada há mais tempo, em outras palavras, o algoritmo substitui a página que está há mais tempo residente na memória sem ser acessada, e que tem portanto baixa probabilidade de ser acessada em um futuro próximo.

Segundo [Cassettari e Midorikawa 2004] o algoritmo LRU se mostra mais eficiente em comparação com outros algoritmos tradicionais como o FIFO (*First In, First Out*), que substitui a página carregada há mais tempo, o MRU (*Most Recently Used*), que substitui a página acessada mais recentemente e o LFU (*Least Frequently Used*), que substitui a página que recebeu menos acessos.

Embora o algoritmo LRU seja o mais utilizado, em função de sua eficiência e simplicidade, existem situações que ele, comprovadamente, apresenta deficiências [Jiang e Zhang 2002]. As situações que apresentam essas deficiências são: a) acessos seqüenciais em um grande número de páginas localizadas em endereços de memórias distintas; b) acesso dentro de grandes loops, levando ao acesso de muitas páginas por iteração; e, c) freqüência de acesso irregular, imprevisível, para uma mesma página.

Este trabalho possui o objetivo de apresentar uma proposta de adaptação do algoritmo LRU-WAR (*LRU with Working Area Restriction*), com a introdução da avaliação da freqüência de reutilização das páginas no critério de substituição. A proposta baseia-se na utilização da estrutura adaptativa básica e a aplicação da substituição de página com a análise da freqüência de acesso em função de seu estado de execução.

Este trabalho está organizado da seguinte forma. A seção 2 apresenta alguns dos algoritmos de substituição de páginas adaptativos relacionando algumas de suas características. A seção 3 descreve a adaptação do algoritmo LRU-WAR com detecção de freqüência de acessos de página. A análise dos resultados das simulações e a avaliação do desempenho são abordadas na seção 4. A seção 5 finaliza o presente trabalho com a apresentação das principais conclusões e também sugerindo futuros trabalhos nesta linha de pesquisa.

2. Trabalhos Relacionados

Os algoritmos de substituição de páginas têm como objetivo principal descartar a página que causará menos impacto no desempenho do sistema, tendo em vista que a penalidade imposta por uma falta de página é considerável. Existem duas principais dimensões que

são abordadas pelos algoritmos: a dimensão recência e a dimensão freqüência. A dimensão recência leva em conta o tempo decorrido desde a última referência a uma determinada página, sendo utilizada em algoritmos como LRU e LIRS (*Low Inter-reference Recency Set*), por exemplo [Jiang e Zhang 2002].

Por sua vez, a dimensão freqüência leva em conta o número de vezes que uma página foi referenciada, sendo utilizada em algoritmos como LFU e FBR (*Frequency Based Replacement*) [Robertson 1990] como exemplos. Existe ainda uma classe que aborda ambas dimensões, levando em conta tanto a recência quanto a freqüência. O algoritmo LRFU (*Least Recently Frequently Used*) é um exemplo dessa classe [Lee et al. 2001].

A eficiência dos algoritmos existentes varia em função do comportamento de acesso à memória apresentado pelos processos. Portanto não existe um único algoritmo que apresente um desempenho razoável que cubra os mais variados padrões de acesso à memória. Nesse sentido um algoritmo capaz de identificar determinados padrões, previamente *conhecidos* e de alguma maneira adaptar-se a essas condições, apresentará melhor desempenho.

Diversos trabalhos sobre algoritmos adaptativos para substituição de páginas foram publicados nos últimos anos, sendo que em sua grande maioria baseado no LRU. Como exemplos, existem os algoritmos SEQ [Glass e Cao 1997] e EELRU (*Early Evictation LRU*) [Smaragdakis, Kaplan e Wilson 1999]. Ambos utilizam a adaptabilidade, onde em uma determinado estado aplicam o critério de substituição LRU.

O objetivo de ambos é detectar dinamicamente a mudança de comportamento de acesso à memória não genuinamente seqüencial para um comportamento genuinamente seqüencial, pois para o primeiro caso o critério LRU é o mais eficiente e para o segundo caso o MRU apresenta um melhor desempenho. O algoritmo LRU-WAR segue a mesma linha do algoritmo EELRU com uma melhor eficiência em termos de sobrecarga [Cassettari e Midorikawa 2004].

Na linha de algoritmos que propõe a utilização da freqüência de reutilização das páginas foram propostos trabalhos como o LRFU [Lee et al. 2001], LIRS [Jiang e Zhang 2002] e ARC [Megiddo e Modha 2003]. Os algoritmos LRFU e LIRS utilizam mecanismos de controle adicionais que consideram a freqüência de utilização na proteção de páginas na memória, descartando as que foram utilizadas menos freqüentemente e ainda considerando a questão da recência aplicando o modelo do LRU puro. Já o algoritmo ARC é uma versão adaptativa do algoritmo 2Q [JOHNSON e Sasha 1994] através da criação de uma fila separada de páginas residentes não reutilizadas.

O presente trabalho é uma adaptação do algoritmo LRU-WAR, notadamente com foco na recência, com a introdução da avaliação da freqüência de reutilização das páginas no critério de substituição. Nesta proposta é feita a substituição de página pela análise da freqüência de acesso em função de seu estado de execução, sendo que para operações normais utiliza-se o critério LRFU-WAR e para operações seqüenciais o critério MRFU-WAR.

O critério de substituição LRFU-WAR aplica o algoritmo LFU fora da área de trabalho, que é a região LRU do algoritmo LRU-WAR. O MRFU-WAR aplica o

algoritmo LFU na área de trabalho, que é a região de páginas utilizadas mais recentemente do LRU-WAR.

Nas avaliações de desempenho é importante a comparação com outros algoritmos de substituição de páginas e também com o algoritmo Ótimo [Belady 1996], um algoritmo teórico que apresenta o melhor resultado possível, ou seja, substitui a página que demandará maior tempo a ser substituída.

O algoritmo 3P [Cassettari e Midorikawa 2005] propõe a redução do custo de implementação dos algoritmos LRU e LRU-WAR, através do controle de uma fila circular, que representa a memória principal, por meio do reposicionamento de três ponteiros, Clock, Antecipado e Circular. No sentido de se avaliar o desempenho comparativo do algoritmo proposto, utiliza-se os seguintes algoritmos: Ótimo, LRU, LRU-WAR e 3P.

Uma das formas de se avaliar o desempenho dos algoritmos propostos é através de simulações utilizando-se de arquivos de traces. Esses arquivos fornecem o padrão de acesso à memória realizado pelo programa de interesse. Segundo [Uhlig e Mudge 1997] os arquivos de traces são resultados da aplicação de três estágios de processamento: *trace collection*, *trace reduction* e, finalmente, *trace processing*.

Os arquivos de traces são fundamentais nas simulações em pesquisas sobre memória virtual. No presente trabalho, são utilizados traces que representam programas com os mais variados padrões de acesso a memória, por exemplo, padrões fortemente seqüenciais, padrões que apresentam alta localidade temporal, padrões com alta característica de acesso a páginas distintas e padrões com poucas páginas distintas.

2.1. Apresentação do algoritmo LRU-WAR

O algoritmo LRU-WAR inova no sentido de propor um algoritmo adaptativo em função do comportamento de acesso à memória apresentado pelos diferentes programas residentes e minimizando as deficiências apresentadas pelo algoritmo LRU. De acordo com [Cassettari e Midorikawa 2004] o algoritmo LRU-WAR utiliza uma fila de páginas organizada em função da recência de acesso, sendo que no início da fila está a página acessada mais recentemente (MRU) e no final da fila a página acessada há mais tempo (LRU).

Um importante conceito para o entendimento deste algoritmo refere-se à área de trabalho, definida por W, como sendo a região de recência, que contém todas as páginas acessadas entre duas faltas de páginas consecutivas. Assim, W varia em função da localidade temporal e espacial apresentada pelos processos.

Uma área de trabalho W com valor alto significa que um grande número de páginas foram acessadas entre duas faltas de páginas consecutivas, sugerindo uma tendência de acessos pelas páginas residentes na memória, e que o conjunto de trabalho utilizado é expressivo. Isto traz um bom resultado em termos de falta de página e, portanto, a adoção da política LRU é adequada.

Já uma área de trabalho W com valor baixo indica que poucas páginas foram acessadas entre duas faltas de páginas consecutivas, sugerindo uma tendência de troca do conjunto de trabalho, e que a política LRU está deixando de ser adequada. O algoritmo utiliza dois parâmetros, o L e o C. O parâmetro L é utilizado para dividir a

fila LRU em duas regiões, a Região Seqüencial e a Região LRU. O parâmetro L é considerado como sendo L=MIN[50, M/2], onde M é o tamanho da memória disponível.

O parâmetro C, conhecido como "carência mínima", delimita uma região protegida, localizada no início da fila LRU, dentro da região seqüencial e engloba algumas posições que possuem uma recência muito baixa e limita o tamanho mínimo da área de trabalho W. O valor de C, adotado originalmente é 5. A Figura 1 ilustra as divisões lógicas da fila LRU para uma memória de 20 páginas. Fonte: [Cassettari e Midorikawa 2004].

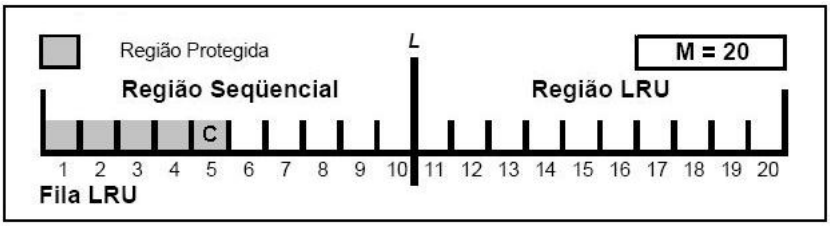

Figura 1. Divisões lógicas da fila LRU utilizada pelo algoritmo LRU-WAR

O funcionamento original do algoritmo LRU-WAR baseia-se em três estados de execução, o estado de Tendência Original, o estado de Tendência Seqüencial e o estado de Operação Seqüencial. A transição entre os estados depende da comparação da área de trabalho W com o divisor de regiões L. Enquanto W > L assume-se que o processo apresenta características LRU e utiliza-se este critério de substituição, trocando-se a última página da fila.

Quando W <= L assume-se uma tendência de mudança de comportamento e um novo estado de execução é assumido: o estado de Tendência Seqüencial. Se esse comportamento persistir por um certo número de faltas de páginas fica caracterizada essa mudança, levando o estado de execução para o estado de Operação Seqüencial. Em qualquer momento em que a área de trabalho (W) for maior que o divisor das regiões seqüencial e LRU (L), o valor de W é zerado e o algoritmo passa a operar no estado de Tendência Original (LRU). A Figura 2 ilustra o diagrama de estado do funcionamento do algoritmo LRU-WAR.

A proposta do presente trabalho é utilizar a estrutura adaptativa apresentada anteriormente para o algoritmo LRU-WAR e proceder uma análise da freqüência, ou seja, do número de acessos de cada página contida nas respectivas regiões seqüencial e LRU, substituindo a página menos acessada em cada uma das regiões.

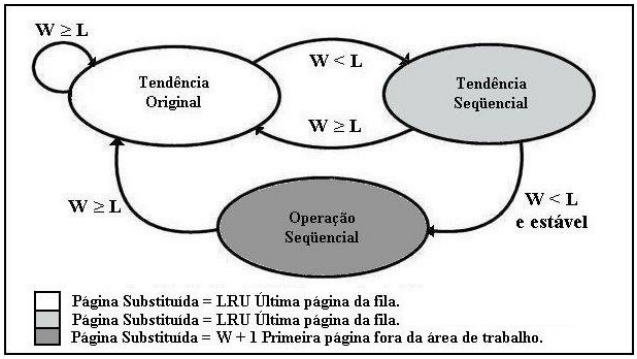

Figura 2. Diagrama de estados do Algoritmo LRU-WAR

3. Proposta para Detecção da Freqüência de Acessos no LRU-WAR

O algoritmo LRU-WAR melhora o desempenho em ambientes com característica de acessos seqüenciais, porém, para ambientes multiprogramados esta melhora na troca de páginas é menos expressiva. Visando a melhoria do desempenho do LRU-WAR, em ambientes multiprogramados, é necessário acrescentar mecanismos para informação adicional neste algoritmo. Nesse sentido, este trabalho propõe utilizar a análise de freqüência por estado de execução, de forma a reduzir as trocas de páginas do algoritmo LRU-WAR, em ambientes multiprogramados.

3.1. Trocas de Páginas por Estado de Execução

Nas trocas de páginas para os traces Multi1 e Multi2, por exemplo, há predominância dos estados de execução de Tendência Original e Seqüencial, do LRU-WAR, conforme apresenta a Figura 3. Para estes traces, o estado de execução de Operação Seqüencial ocorre com menor freqüência.

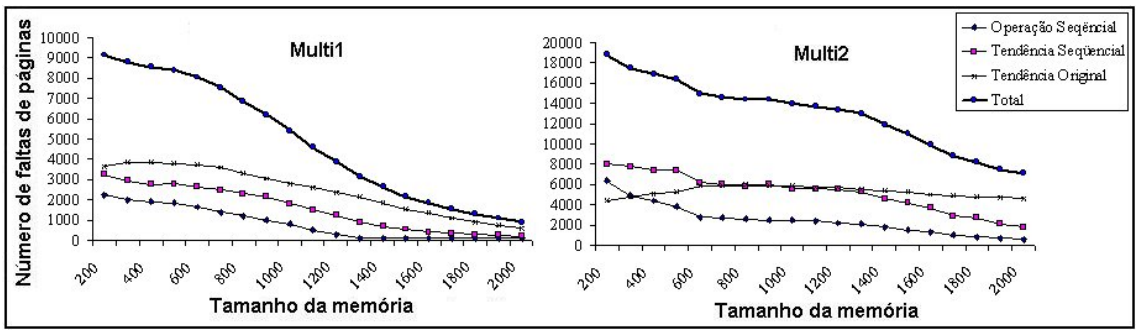

Figura 3. Troca de Páginas por Estado de Execução

De forma geral, a Figura 4 apresenta a média dos estados de execução para os traces Multi1, Multi2, Multi3, CS e 2_POOL.

Para aproveitar as características do bom desempenho pela adaptabilidade do LRU-WAR, este estudo propõe acrescentar, neste algoritmo, a informação sobre a detecção da freqüência de acessos por estado de execução, de forma a melhorar seu desempenho para ambientes com multiprogramação, sem impactar o bom desempenho nos acessos seqüenciais.

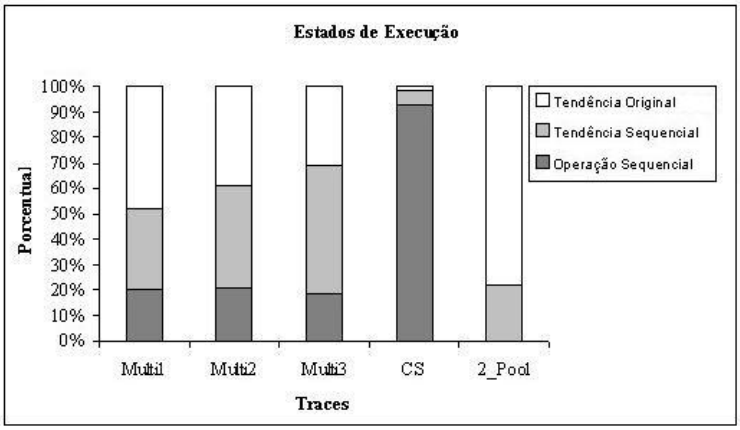

Figura 4. Média de Troca de Páginas por Estado de Execução

3.2. Detecção da Freqüência de Acessos por Estado de Execução

O algoritmo LRU-WAR define 3 estados de execução para a adaptabilidade da escolha da página a ser substituída, conforme a Figura 2.

A informação da freqüência de acessos possui bom desempenho para ambientes com característica não seqüenciais e os estudos do LRU-WAR mostram que sua adaptabilidade melhora a eficiência para os ambientes com Operação Seqüencial, devido à restrição da área de trabalho deste algoritmo.

Este trabalho propõe associar estas duas características através do acréscimo da informação de freqüência ao LRU-WAR, conforme sua área de trabalho e seu estado de execução.

Para os estados de execução em Tendência Original e Tendência Seqüencial é proposto utilizar a informação da freqüência fora da área de trabalho no algoritmo LRU-WAR. Para o estado de execução em Operação Seqüencial, a informação sobre a freqüência de acessos não é relevante para as páginas fora da área de trabalho, porque as páginas menos recentemente utilizadas desta região (região LRU) possuem maior probabilidade de novo acesso. Neste estado de execução a informação sobre a freqüência possui relevância para as páginas dentro da área de trabalho, pois, estas são as mais recentemente acessadas e possuem menor probabilidade de novo acesso em Operação Seqüencial.

O LRU-WAR propõe uma área protegida, pois as páginas acessadas menos recentemente possuem elevada probabilidade de novo acesso. Porém, o acréscimo da informação da freqüência de acessos minimiza este impacto, o que permite para esta proposta detectar a freqüência de acessos na área de trabalho, para a Operação Seqüencial.

A Tabela 1 sintetiza a presente abordagem relacionando os estados de execução do LRU-WAR e a área para detecção da freqüência de acessos.

Tabela 1. Estados de Execução e Pontos de Substituição do Algoritmo LRU-WAR com Detecção de Freqüência

Estado de Execução	Tamanho da Área de Trabalho W	Critério de Substituição	Ponto de Substituição (Posição na Fila LRU)
Tendência Original	Maior que L	LRFU-WAR	M até W+1
Tendência Seqüencial	Menor ou Igual a L	LRFU-WAR	M até W+1
Operação Seqüencial	Menor ou Igual a L e estável	MRFU-WAR	W até MRU

Esta proposta mantém as características de adaptabilidade do LRU-WAR e acrescenta a informação da freqüência de acessos a este algoritmo.

3.3. Algoritmo LRU-WAR com Detecção de Freqüência de Acesso de Páginas

Para esta proposta não há alterações expressivas na lógica do algoritmo LRU-WAR. As alterações em seu pseudocódigo resumem-se às linhas 22, 24 e 29.

No algoritmo original, a linha 22 apresenta a remoção da página W+1 e as linhas 24 e 29 a remoção da página M (página LRU). Para esta proposta, o pseudocódigo é atualizado para:

```
1.  Se a página acessada está carregada na memória:
2.     P = posição da página na fila LRU;
3.     Se P > W:
4.        Se N > 0: /* Se está em operação seqüencial */
5.           INÉRCIA = 0; /* Encerra a operação seqüencial */
6.           Se P > W+1 e P ≤ W+TC+1 e (N ≤ MP ou N < 50):
7.              TC = TC + N; /* Detecta erro e aumenta TC */
8.           N = 0;
9.        W = P; /* Aumenta área de trabalho */
10.    Reordena a página na primeira posição da fila.
11. Senão (a página acessada não está carregada na memória):
12.    Se a memória está cheia:
13.       Se W ≤ L:
14.          INÉRCIA = INÉRCIA + 1;
15.             Se W ≤ C:
16.                W = C + 1;
17.             Se INÉRCIA >= W+TC: /* Operação seqüencial */
18.                Se N < M ou N < 50:
19.                   N = N + 1;
20.                Se TC > C:
21.                   TC = TC - 1;
22.             Remove a página menos acessada na Área de Trabalho;
23.          Senão (INÉRCIA < W+TC): /* Tendência seqüencial */
24.             Remove a página menos acessada fora da Área de Trabalho;
25.       Senão (W > L): /* Tendência LRU */
26.          INÉRCIA = 0;
27.          W = 0;
28.          N = 0;
29.          Remove a página menos acessada fora da Área de Trabalho;
30.    Carrega a página acessada no início da fila.
```

Estas alterações foram realizadas para que a análise da freqüência do acesso às páginas fosse feita por área, conforme o estado de execução do algoritmo original.

4. Análise dos Resultados das Simulações e Avaliação de Desempenho

Para a avaliação de desempenho desta proposta, os traces utilizados para testes de ambientes multiprogramados foram o Multi1, Multi2 e Multi3.

No entanto, esta proposta busca também preservar as boas características de desempenho deste algoritmo para em ambientes com forte comportamento seqüencial e não seqüencial. Para a avaliação do impacto nestes casos, foram utilizados os traces CS e 2_Pool. A Tabela 2 apresenta a descrição resumida para os traces utilizados.

Tabela 2. Traces Utilizados na Avaliação de Desempenho [Cassettari e Midorikawa 2005]

Arquivo de Trace	Descrição Resumida	Páginas Distintas
Mult1	Execução simultânea das aplicações cscope e cpp.	2606
Mult2	Execução simultânea das aplicações cscope, cpp e postgres.	5684
Mult3	Execução simultânea das aplicações cpp, gnuplot, glimpse e postgres.	7454
CS	Cscope, ferramenta que analisa códigos fontes de programas em C.	1409
2_Pool	Simulador sintético.	9939

4.1. Comparação do Desempenho do Algoritmo LRU-WAR modificado

Com o intuito de se verificar o comportamento das alterações introduzidas no algoritmo LRU-WAR foram feitas simulações utilizando-se os algoritmos: Ótimo, LRU, LRU-WAR, 3P e LRU-WAR com detecção de freqüência de acessos de página, identificado como LRU-WAR-FREQ. A Figura 5 apresenta a comparação do desempenho entre esses algoritmos para os traces Multi1 e Multi2.

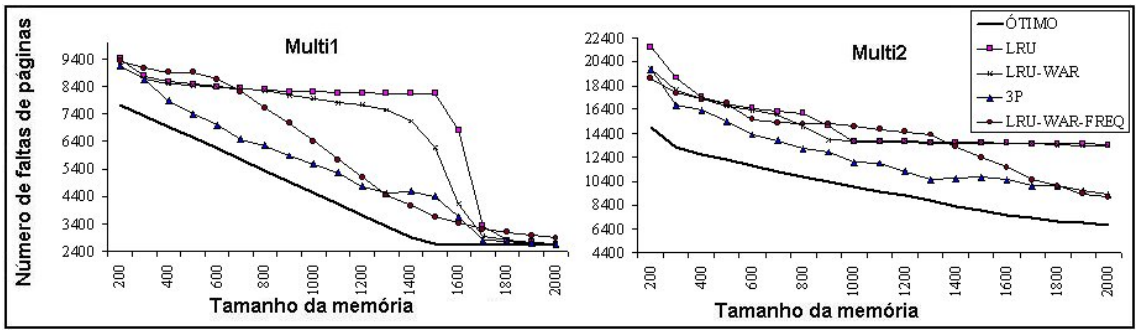

Figura 5. Comparação de Desempenho nos Traces Multi1 e Multi2

Esta comparação comprova a proximidade do desempenho entre os algoritmos LRU e LRU-WAR, para ambientes com característica de multiprogramação e a melhoria para o LRU-WAR-FREQ.

4.2. Impacto da Freqüência de Acessos no LRU-WAR

A detecção da freqüência de acessos na área de trabalho, para os estados de execução de Tendência Original e Seqüencial, melhoram o desempenho do LRU-WAR para ambientes multiprogramados com características similares às dos traces Multi1 e Multi2, sem prejudicar o desempenho para ambientes com forte Tendência Seqüencial e não seqüencial.

A Tabela 3 apresenta os resultados dos estudos de impacto nestes traces para estes estados de operação, onde o "Melhor Caso" refere-se ao melhor resultado obtido em termos do percentual do número de faltas de página para o referido tamanho de memória. De forma análoga, o "Pior Caso" refere-se ao pior resultado obtido. A coluna "Média %" resume a média das simulações para os diferentes tamanhos de páginas utilizados. Por exemplo, o melhor resultado para o trace Multi1 foi obtido com uma memória de tamanho 1400 e apresentou um número de falta de páginas 42,63% inferior ao algoritmo LRU-WAR, sendo que na média o desempenho foi melhor em 7,43%.

Tabela 3. Análise de Freqüência para Tendência Original e Seqüencial

Arquivo de Trace	Melhor Caso		Pior Caso		Média %
	Diferença %	(Memória)	Diferença %	(Memória)	
Mult1	-42,63	1400	10,74	1800	-7,43
Mult2	-31,56	2000	11,46	900	-4,12
Mult3	-2,66	2000	6,79	1400	3,17
CS	0,48	50	4,16	1250	2,31
2_Pool	-1,49	500	0,66	7500	-0,05

Para ambientes em Operação Seqüencial, a avaliação da freqüência na área não protegida do LRU-WAR não apresenta impacto positivo, porque as páginas desta área

próximas a W+1 possuem alta probabilidade de uso em um futuro próximo. A Tabela 4 comprova que o desempenho em ambientes seqüenciais (CS) seria prejudicado pela detecção de freqüência fora da área de trabalho, para a Operação Seqüencial.

Tabela 4. Análise de Freqüência para Operação Seqüencial (página LRU até W)

Arquivo de Trace	Melhor Caso		Pior Caso		Média %
	Diferença %	(Memória)	Diferença %	(Memória)	
Mult1	-43,07	1400	10,24	1800	-5,24
Mult2	-31,61	2000	8,94	1100	-2,13
Mult3	-0,05	2000	6,95	200	3,95
CS	0,56	1350	127,19	1150	58,55
2_Pool	-1,49	500	0,55	2000	-0,05

Nesta tabela, é comprovado para o trace CS que a aplicação de freqüência fora da área de trabalho, quando em Operação Seqüencial, impacta negativamente o desempenho médio em 58,55% sobre o LRU-WAR original. A curva de desempenho desta análise, para diferentes tamanhos de memória, é apresentada na Figura 6.

A subseção 4.2 apresenta os resultados desta abordagem.

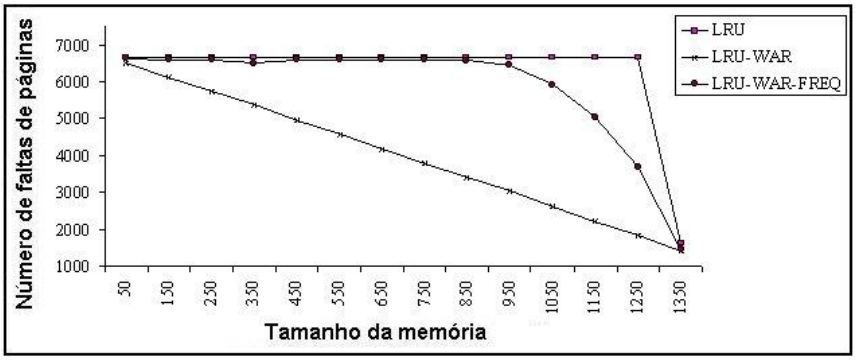

Figura 6. Impacto da Detecção de Freqüência fora da Área de Trabalho para a Operação Seqüencial

Porém, em Operação Seqüencial, a detecção de freqüência possui impacto positivo quando realizada na área próxima às páginas acessadas mais recentemente (região MRU), porque estas páginas possuem menor tendência de uso num futuro próximo, para este estado de execução.

4.3. Resultados com a Detecção de Freqüência por Estado de Execução

A detecção da freqüência de acessos apresenta resultados positivos ao ser aplicada na região de páginas mais recentemente acessadas para o estado de Operação Seqüencial, porque as páginas desta região não possuem tendência de novo acesso próximo. A Tabela 5 apresenta os resultados para a análise da freqüência de acessos às páginas na área de trabalho para a Operação Seqüencial, e fora da área de trabalho para a Tendência Original e Seqüencial.

Esta tabela apresenta a melhora de desempenho em relação à Tabela 3, resultado da aplicação da detecção de freqüência na área de trabalho, para o estado de execução em Operação Seqüencial.

Tabela 5. Análise de Freqüência para Operação Seqüencial (página W até MRU)

Arquivo de Trace	Melhor Caso		Pior Caso		Média %
	Diferença %	(Memória)	Diferença %	(Memória)	
Mult1	-43,35	1400	10,13	1800	-9,64
Mult2	-31,85	2000	10,08	900	-5,75
Mult3	-3,17	2000	5,20	1300	1,32
CS	-0,40	950	0,91	250	0,32
2_Pool	-1,49	500	0,55	2000	-0,05

5. Conclusão

Esta pesquisa apresenta uma proposta para melhora no desempenho do algoritmo LRU-WAR, a partir da análise da freqüência no acesso às páginas. A melhora no desempenho ocorre quando aplicado conforme o estado de execução definido pelo LRU-WAR e a área a ser analisada.

Para os traces MULTI houve melhora no desempenho, onde observa-se que esta ocorre em maior amplitude nas trocas de páginas em Tendência Original e em Operação seqüencial. Para os traces CS e 2_POOL, a abordagem realizada neste estudo não traz variações significativas, o que garante a boa eficiência do LRU-WAR para ambientes com estas características de comportamento.

Em trabalhos futuros, sugere-se pesquisar o ponto ótimo para a detecção de freqüência por estado de execução e a criação de um parâmetro dinâmico para esta identificação do ponto ótimo, conforme a probabilidade da tendência de entrar em operação Seqüencial ou Original (LRU). Também é sugerido avaliar o comportamento do algoritmo LRU-WAR com detecção de freqüência durante sua execução e utilizar os parâmetros registrados para a adaptação do algoritmo durante novas execuções.

Esse tipo de técnica é conhecida como "*profiling*". Um exemplo seria o registro do percentual que o programa passa em cada um dos três estados do algoritmo e utilizar essa informação para aumentar ou diminuir a penalidade da carência para entrar no modo seqüencial. Esses parâmetros poderiam ser reavaliados a cada execução no sentido de mantê-los mais sintonizados com o ambiente de execução utilizado.

Lista de Referências

Belady, L. A. (1996) "A study of replacement algorithms for a virtual storage computer". IBM System Journal, v.5, n.2, p.78-101.

Cassettari, H. H., Midorikawa, E. T. (2004) "Algoritmo Adaptativo de Substituição de Páginas LRU-WAR: Exploração do Modelo LRU com Detecção de Acessos Seqüenciais". In: Anais do I Workshop de Sistemas Operacionais (WSO 2004), Salvador, BA.

Cassettari, H. H., Midorikawa, E. T. (2005) "Algoritmo de Substituição de Páginas 3P: Acrescentando Adaptatividade ao Clock". In: Anais do II Workshop de Sistemas Operacionais (WSO 2005), São Leopoldo, RS.

Glass, G., Cao, P. (1997) "Adaptive Page Replacement Based on Memory Reference Behavior", In Proc. of the ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS'97), Seattle, pp.115-126.

Jiang, S., Zhang, X. (2002) “LIRS: An Efficient Low Inter-Reference Recency Set Replacement Policy to Improve Buffer Cache Performance”, In Proc. of the ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS’02), Marina Del Rey, pp.31-42.

Johnson, T., Shasha, D. (1994) “2Q: a low overhead high performance buffer management replacement algorithm”. In: International Conference on Very Large Data Bases, 20., Santiago, 1994. *VLDB’ 94:* Proceedings. Santiago: Morgan Kaufmann, 1994. p.439-450.

Lee, D. et al. (2001) “LRFU: a spectrum of policies that subsumes the Least Recently Used and Least Frequently Used policies”. IEEE Transactions on Computers, vol.50, n.12, p.1352-1361.

Megiddo, N., Modha, D.S. (2003) “ARC: a self-tuning, low overhead replacement cache”. In: Conference on File and Storage Technologies, 2., San Francisco, 2003. *FAST’ 03:* Proceedings. San Francisco: USENIX, 2003. p.115-130.

Midorikawa, E. T., Piantola, R. L., Cassettari, H. H. (2007) “Influência dos Parâmetros de Controle no Desempenho de Algoritmos Adaptativos de Substituição de Páginas”. In: Anais do IV Workshop de Sistemas Operacionais (WSO 2007), Rio de Janeiro, RJ.

Piantola, R. L., Midorikawa, E. T. (2008) “Ajustando o LRU-WAR para uma Política de Gerência de Memória Global”. In: Anais do V Workshop de Sistemas Operacionais (WSO 2008), Belém do Pará, PA.

Robinson, J. T., Devarakonda, M. V. (1990) “Data cache management using frequency-based replacement”. In: ACM SIGMETRICS Conference on Measurement and Modeling of computer systems.

Silberschatz, A., Galvin, P. B.; Gagne, G. (2005) “Operating Systems Concepts”. 7th ed., Wiley.

Smaragdakis, Y., Kaplan, S., Wilson, P. (1999) “EELRU: simple and effective adaptive page replacement”. In: International Conference on Measurement and Modeling of Computer Systems, 24., Atlanta, 1999. SIGMETRICS’ 99: Proceedings. Atlanta: ACM, 1999. p.122-133.

Uhlig, R. A., Mudge, T. N. (1997) “Trace-driven memory simulation: a survey”. ACM Computing Surveys, v.29, n.2, p.128-170.