Caracterização de Cargas de Trabalho em Estudos sobre Gerência de Memória Virtual

Hugo Henrique Cassettari, Edson Toshimi Midorikawa

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

{hugo.cassettari,edson.midorikawa}@poli.usp.br

Abstract. *The characterization of workloads for memory systems evaluation is important because it allows to detect factors in programs which may or may not favor the performance of each possible system configuration. This paper describes the Elephantools: a software tools package applied to the characterization of workloads used in experiments on virtual memory management. Providing visual and statistics resources for identifying memory access patterns inherent in specific programs, the Elephantools contributes both to the enrichment of studies on reference locality and to the understanding of some phenomena observed in simulations.*

Resumo. *A caracterização de cargas de trabalho utilizadas em avaliações de sistemas de memória é importante porque permite detectar fatores presentes nos programas que favorecem ou não o desempenho de cada possível configuração do sistema. Este artigo descreve o Elephantools: um pacote composto por ferramentas de software voltadas à caracterização de cargas de trabalho em experimentos sobre gerência de memória virtual. Oferecendo recursos visuais e estatísticos para a identificação de padrões de acesso à memória inerentes aos programas, o Elephantools contribui ao enriquecimento de estudos sobre localidade de referências e à compreensão de alguns fenômenos observados em simulações.*

1. Introdução

A utilização de uma política de substituição de páginas eficiente é fator fundamental em qualquer sistema de memória com paginação. Dizemos que ocorre uma falta de página quando um dado a ser processado se encontra em uma página que não está carregada na memória principal e, portanto, inacessível diretamente ao processador. Neste caso, a página precisa ser imediatamente alocada na memória para que a execução do processo continue. Contudo, se a memória principal já está totalmente preenchida, uma das páginas então carregadas deve ser sobreposta pela página que precisa ser trazida da memória secundária. Ocorre assim, nesse momento, a necessidade de uma substituição de página. A dificuldade está em decidir qual página deve deixar a memória, ou seja, qual a página menos importante para a execução do processo no momento da substituição. O sistema operacional é o responsável por esta decisão e, para isso, implementa um algoritmo de substituição de páginas. O critério utilizado pelo algoritmo elege a página que deixa a memória quando uma substituição precisa ser efetuada.

Midorikawa (1997) propôs o ambiente para avaliação de sistemas de memória Elephantus, o qual simula a execução de programas (cargas de trabalho) em diferentes contextos, isto é, considerando diversas políticas para substituição de páginas aplicadas em diferentes situações de disponibilidade da memória principal. A metodologia para a realização de estudos com o auxílio deste ambiente inclui a obtenção de arquivos de *trace* como cargas de trabalho, a realização de simulações com tais arquivos e um processo de análise sobre os resultados encontrados. Um arquivo de *traces* consiste basicamente na listagem cronológica dos diversos acessos à memória realizados por um processo. Em outras palavras, descreve passo a passo o comportamento de um dado programa quanto à utilização da memória. Estes arquivos são criados através de um gerador de *traces*, que executa o programa alvo e coleta os acessos à memória realizados ao longo de seu processamento (UHLIG; MUDGE, 1997).

Com os resultados obtidos através das simulações, procura-se determinar o melhor algoritmo de substituição de páginas para cada programa, assim como o porquê de um algoritmo ser mais eficiente do que outro na situação analisada. As características de localidade de um processo explicam a variação de desempenho observada entre os algoritmos em condições idênticas de simulação.

O pacote de ferramentas Elephantools, aqui apresentado, disponibiliza recursos para uma caracterização precisa das cargas de trabalho em termos de acessos à memória. Ilustrando comportamentos e identificando aspectos relevantes nos programas analisados, o pacote oferece meios para complementar e enriquecer estudos sobre localidade de referências e gerência de memória virtual – especialmente aqueles realizados através do ambiente Elephantus.

A Seção 2 deste artigo discute a importância da análise de localidade nos programas e a sua relação com o desempenho do sistema de memória. A Seção 3 descreve as ferramentas de software que compõem o pacote Elephantools, incluindo exemplos de sua utilização, enquanto a Seção 4 ilustra um estudo de caso em que o conjunto de ferramentas é empregado. A Seção 5, por fim, conclui o trabalho.

2. Análise de Localidade

Genericamente, em sistemas de memória virtual paginada, processos computacionais tendem a explorar, com maior ou menor grau de intensidade, a propriedade conhecida como localidade de referências. Esta propriedade afirma que apenas parte das páginas que compõem o espaço de endereçamento virtual de um programa são efetivamente necessárias à sua execução num certo intervalo de tempo. Em outras palavras, os dados processados em um determinado momento de execução normalmente se concentram em algumas poucas páginas. Esta propriedade é o que fundamenta e valoriza a importância da chamada hierarquia de memória, ou seja, a forma como os componentes físicos de memória são organizados na arquitetura de um computador. Dispositivos rápidos, porém dimensionalmente limitados, devem armazenar temporariamente os dados mais requisitados por um processo. Por outro lado, dispositivos de memória secundária – lentos, mas com grande capacidade de armazenamento – alocam a totalidade do espaço de endereçamento utilizado pelo processo.

O conceito de *working set* – conjunto de trabalho –, proposto por Denning (1968), também foi formalizado com base na propriedade de localidade de referências. O *working set* de um processo em execução é o conjunto das páginas requeridas para o

seu processamento em um dado intervalo de tempo. Ele representa, de certa forma, a concretização da propriedade de localidade: as páginas que compõem o conjunto retratam, por si só, uma localidade inerente ao momento de execução do processo.

A importância desta propriedade decorre do seguinte fato: o grau de localidade apresentado por um programa pode afetar diretamente o seu tempo de processamento, haja vista a influência que exerce na reutilização das páginas residentes e, conseqüentemente, no desempenho do sistema de memória. Logo, as políticas de substituição de páginas mais eficientes procuram explorar, cada uma ao seu modo, as características de localidade encontradas nos processos, sobretudo localidade temporal. Um caso de localidade temporal é observado quando algumas poucas páginas de memória são continuamente referenciadas durante o intervalo de tempo considerado. Não obstante, se os acessos à memória se restringem a uma região específica do espaço de endereçamento virtual, isto é, se as páginas referenciadas possuem endereços próximos entre si, observamos um caso de localidade espacial.

A localidade espacial é positiva quando está restrita a um conjunto de páginas reduzido, como um *loop* no qual poucas páginas são referenciadas – em relação ao tamanho de memória disponível. Um padrão de acessos à memória desse tipo faz com que o *working set* do processo também permaneça reduzido, induzindo à desejada reutilização das páginas residentes. Acompanhada de forte localidade temporal, portanto, a localidade espacial é interessante. Caso contrário, a presença ou não de localidade espacial é indiferente à grande maioria das políticas tradicionais de gerenciamento da memória principal. Ainda assim, pode ser muito importante para o gerenciamento eficiente de memórias *cache*, o que a torna relevante.

A presença de localidade temporal, por outro lado, é invariavelmente benéfica porque implica na reutilização pontual de páginas residentes. O LRU é um exemplo de algoritmo de substituição que procura prever e explorar tal propriedade no intuito de diminuir a ocorrência de faltas de página: seu critério é sempre substituir páginas de memória que não tenham sido recentemente referenciadas, ou seja, páginas que não demonstrem a tendência de acessos contínuos no momento de execução observado.

A análise gráfica dos padrões de acesso à memória apresentados pelos programas leva a conclusões a respeito do grau de localidade de referências associado aos mesmos. Um padrão designa uma regularidade na forma como os acessos à memória acontecem durante um período de execução do sistema. Esta regularidade também pode se dar em termos espaciais e/ou temporais.

2.1. Limitações dos Recursos Gráficos Tradicionais para Análise de Localidade

A interpretação visual das características de acesso à memória presentes nos programas é o procedimento para identificação de padrões mais utilizado em trabalhos que se propõem a analisar localidade de referências. Por permitirem a inferência de padrões de acesso de maneira intuitiva e didática, é muito comum o uso de gráficos que forneçam uma visão geral do comportamento dos programas, principalmente mapas de acesso.

Um mapa (ou gráfico) de acessos consiste essencialmente na representação visual de uma matriz de pontos que informa quais páginas de memória são referenciadas pelo programa estudado em cada momento de seu processamento. O eixo horizontal do gráfico corresponde ao tempo virtual da execução, mensurado em número de acessos,

enquanto o espaço de endereçamento virtual utilizado pelo programa é distribuído no eixo vertical.

Hatfield; Gerald (1971) já utilizavam gráficos desta natureza há mais de 30 anos para analisar otimizações na estrutura de programas que favoreciam o desempenho de sistemas com memória virtual. Diversos outros trabalhos mais recentes também introduzem gráficos de acesso para propor técnicas de identificação automática de padrões (CHOI et al., 1999); (CHOI et al., 2000); (KIM et al., 2000) ou para caracterizar cargas de simulação em experimentos (PHALKE, 1995); (GLASS; CAO, 1997); (MARKATOS, 1997).

Entretanto, é natural que haja sempre uma escala reduzindo o tamanho do gráfico, dimensionando-o de forma que ele ocupe um espaço adequado à sua visualização global. Desta maneira, muitos pontos são aglutinados, ocasionando eventualmente enormes perdas de informação. Quanto maior for o número de acessos à memória realizados pelo programa estudado, e maior o espaço de endereçamento total de memória que este utiliza, maior será a compressão visual do gráfico; conseqüentemente, menor será a precisão das informações. A Figura 1 exibe um exemplo de gráfico de acessos completo e um trecho ampliado do mesmo gráfico, evidenciando a aglutinação de pontos que pode acontecer nos mapas completos.

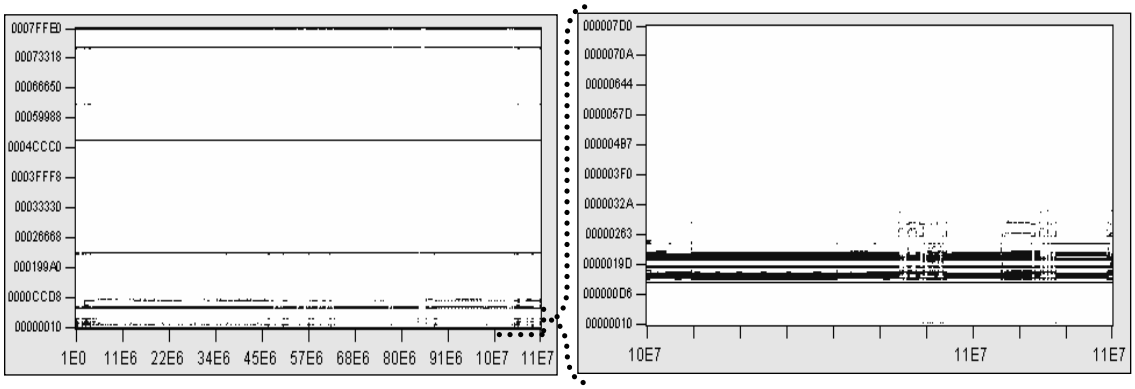

Figura 1. Exemplo de gráfico de acessos completo e respectiva ampliação de um pequeno trecho

Obviamente, a utilização de gráficos de acesso com tamanho original tornaria a análise muito mais precisa, porém sua reprodução completa para documentação impressa seria praticamente inviável. A própria visualização em tela provavelmente teria de ser feita por partes, impedindo assim uma noção geral do comportamento do programa e dificultando a identificação de fases de processamento. A conclusão é que os gráficos de acesso convencionais consistem em um recurso certamente útil, mas limitado se isoladamente empregado.

Outro tipo de gráfico muito usual é o que ilustra as chamadas superfícies de localidade *(locality surfaces)*. Propostas por Grimsrud et al. (1996) como uma técnica para quantificar a presença de localidade temporal e espacial nos programas, tais superfícies são utilizadas principalmente em estudos sobre gerenciamento de memórias *cache* (SORENSON; FLANAGAN, 2001). O objetivo da técnica é detectar trechos com localidade de referências em processos – de acordo com a reutilização das páginas e de seus endereços físicos – e construir superfícies tridimensionais representativas, com regras específicas de interpretação.

As superfícies de localidade caracterizam os acessos à memória do ponto de vista macroscópico, considerando todo o programa. A terceira dimensão do gráfico, representada pelo eixo z, indica a quantidade de acessos em que a distância (tempo virtual decorrido entre duas referências a uma mesma página) e o deslocamento espacial (diferença dos endereços virtuais das páginas acessadas consecutivamente) obtidos são aqueles representados pelas respectivas coordenadas x e y.

De acordo com o relevo da superfície, é possível concluir se os acessos à memória são predominantemente seqüenciais – sem reutilização pontual de páginas –, ou se ocorrem na forma de *loops*, ou ainda se apresentam alta localidade temporal, por exemplo. A Figura 2 ilustra a superfície de localidade gerada para o estudo do programa Gnuplot. As farpas diagonais no gráfico indicam presença de longos acessos seqüenciais. Por sua vez, o pico em torno da origem da superfície (coordenada 0,0,0 do gráfico) denota forte reutilização de algumas poucas páginas vizinhas.

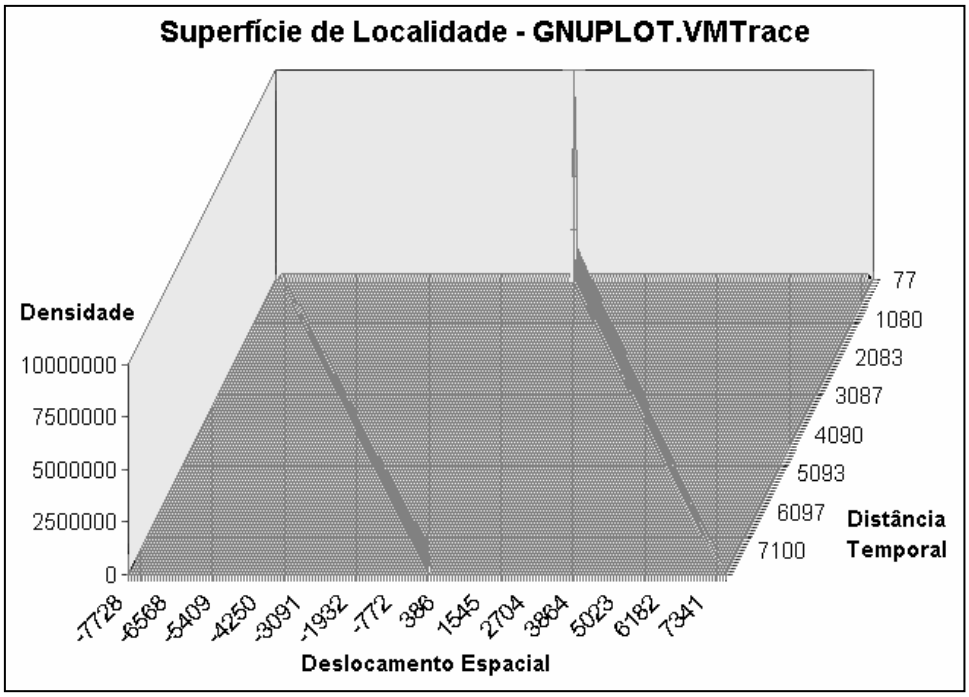

Figura 2. Exemplo de superfície de localidade (programa Gnuplot)

Apesar de representativas, as superfícies de localidade não informam em que ordem cada referência à memória ocorre, mesclando possíveis fases de processamento distintas – as quais, por sua vez, podem demonstrar características únicas de acesso. A alternativa para superar esta limitação seria a criação de uma superfície para cada fase de processamento, mas isto implicaria na identificação prévia das mesmas e, portanto, na utilização de recursos adicionais para visualização ou reconhecimento automático de padrões de acesso à memória. Além disso, superfícies de localidade também podem apresentar o problema da aglutinação de pontos, dependendo das escalas dos eixos x e y do gráfico, tornando a análise logicamente menos precisa.

Nota-se uma certa escassez de recursos de análise, sobretudo gráficos, que representem de forma completa, eficiente e precisa as características de localidade de referências encontradas em programas. O pacote de ferramentas Elephantools foi idealizado e desenvolvido para suprir esta demanda científica.

3. O Pacote Elephantools

Três ferramentas originais, complementares entre si, integram o pacote Elephantools, cuja meta é oferecer uma ou mais formas de representação para caracterizar diversos aspectos relevantes do modo como os programas acessam a memória. A primeira ferramenta gera gráficos de acesso convencionais, em duas dimensões, mas disponibiliza diferentes opções de visualização e ampliação. Outra ferramenta fornece dados para gerar superfícies de acesso tridimensionais, minimizando a perda de informações inerente aos gráficos bidimensionais. A terceira ferramenta processa e formata uma série de dados analíticos e gráficos referentes aos acessos realizados em cada página de memória e às suas inter-relações. O conjunto de ferramentas oferece recursos, assim, para um amplo processo de análise de localidade e para estudos de exploração do modelo LRU.

3.1. TelaTrace

TelaTrace foi a primeira das três ferramenta desenvolvidas, projetada com a finalidade de auxiliar na caracterização de padrões de acesso à memória presentes em programas típicos e também em programas paralelos (CASSETTARI; MIDORIKAWA, 2002). De fato, trata-se basicamente de uma ferramenta de visualização de acessos. Arquivos de *trace* fornecem os dados necessários para a criação de janelas de visualização referentes aos programas que representam. Os gráficos de acesso – ou mapas de acesso – gerados são bidimensionais, mas podem ser ampliados. A resolução máxima de uma ampliação é definida pelo próprio usuário ao iniciar o uso da ferramenta, que é composta de dois módulos: um módulo de pré-processamento dos *traces*, codificado em linguagem C; e um módulo gráfico, codificado em Java. O módulo de pré-processamento produz uma matriz de pontos com a resolução especificada pelo usuário, enquanto o módulo gráfico se alimenta desta matriz para construir uma interface gráfica adequada à visualização completa ou ampliada do comportamento do programa, em termos de utilização da memória virtual.

A interface gráfica da ferramenta dispõe de duas janelas, conforme ilustra a Figura 3: uma janela principal e outra de aproximação. A janela principal (janela superior da Figura 3) apresenta todos os acessos à memória realizados pelo programa analisado. Se este for um programa paralelo, cada processador é diferenciado por uma cor específica e o tipo de acesso (leitura/gravação) é identificado pela tonalidade de cor. Onde houver sobreposição de acessos, a ferramenta TelaTrace também realiza uma operação de sobreposição das cores, permitindo que as características dos acessos sejam identificadas inclusive nestes casos. Para facilitar a representação do tempo virtual, a notação numérica “nEe” é empregada, cuja interpretação é $n*10^e$. Assim, por exemplo, a notação 13E6 representa o número $13*10^6$ = 13.000.000 de acessos. As páginas de memória – no eixo vertical – são identificadas por meio de endereços hexadecimais.

A janela inferior da Figura 3 é a janela de aproximação, composta por vários campos: uma janela de aproximação base (à esquerda), uma janela de aproximação sucessiva (à direita), diversos botões de controle e sensores. A janela de aproximação base amplia uma determinada região do gráfico completo. Esta região pode ser selecionada por meio de botões de controle – que servem para avançar ou retroceder a visualização de um trecho da janela principal – ou diretamente através de um clique na própria janela principal, que exibe o gráfico completo. Caso se deseje uma aproximação

maior, uma subregião pode ser definida, bastando um clique na janela de aproximação base para que a área de interesse seja mostrada na janela de aproximação sucessiva.

As duas janelas de aproximação possuem ainda sensores que informam a página e o tempo virtual da posição do gráfico apontada pelo *mouse*. Outros sensores indicam se esta posição contém ou não acessos realizados pelos processadores disponíveis – até 8 processadores –, diferenciando também o tipo de acesso.

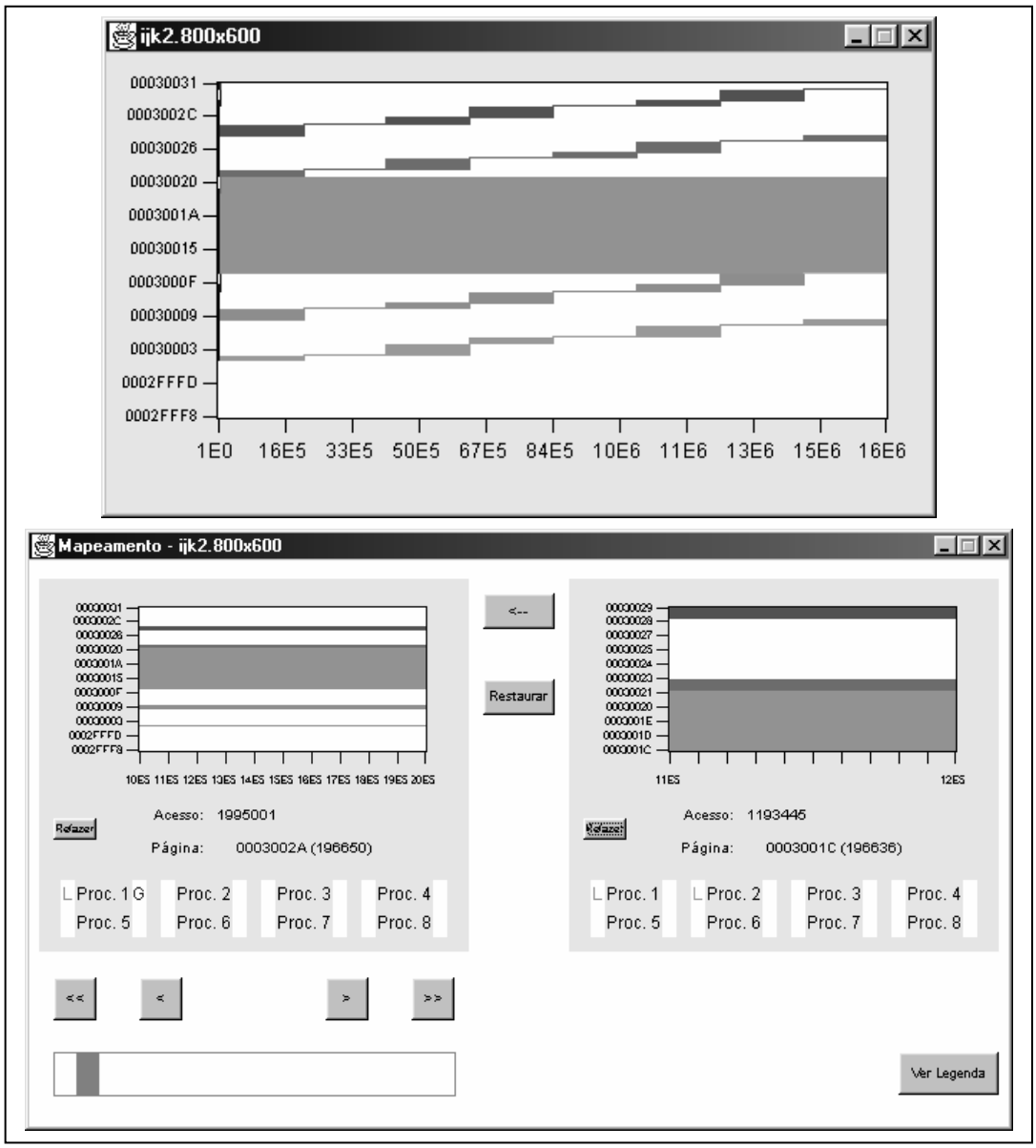

Figura 3. Interface gráfica da ferramenta TelaTrace: janelas principal e de aproximação

Apesar de possuir funcionalidades interessantes e permitir uma análise mais detalhada dos programas sem perder a praticidade da visualização global, a ferramenta TelaTrace não gera gráficos tridimensionais. A terceira dimensão é importante porque agrega a informação de densidade às posições dos gráficos tradicionais, ou seja, informa quantos pontos foram aglutinados em cada posição dos mesmos. Para viabilizar este recurso, a ferramenta Mapa3D foi desenvolvida.

3.2. Mapa3D

Desprovido de interface gráfica, o Mapa3D tem como objetivo formatar um mapa de acessos tridimensional, gerando dados que possam ser convertidos em pontos e desenhados por uma ferramenta gráfica genérica. Codificada em linguagem C, a ferramenta disponibiliza dois tipos de saída: na forma de coordenadas x, y, z; e na forma de uma matriz numérica. Neste último formato, o eixo x compõe a primeira linha da matriz; o eixo y preenche a primeira coluna; e as demais linhas e colunas correspondem à variação de valores no eixo z, formando a superfície do gráfico.

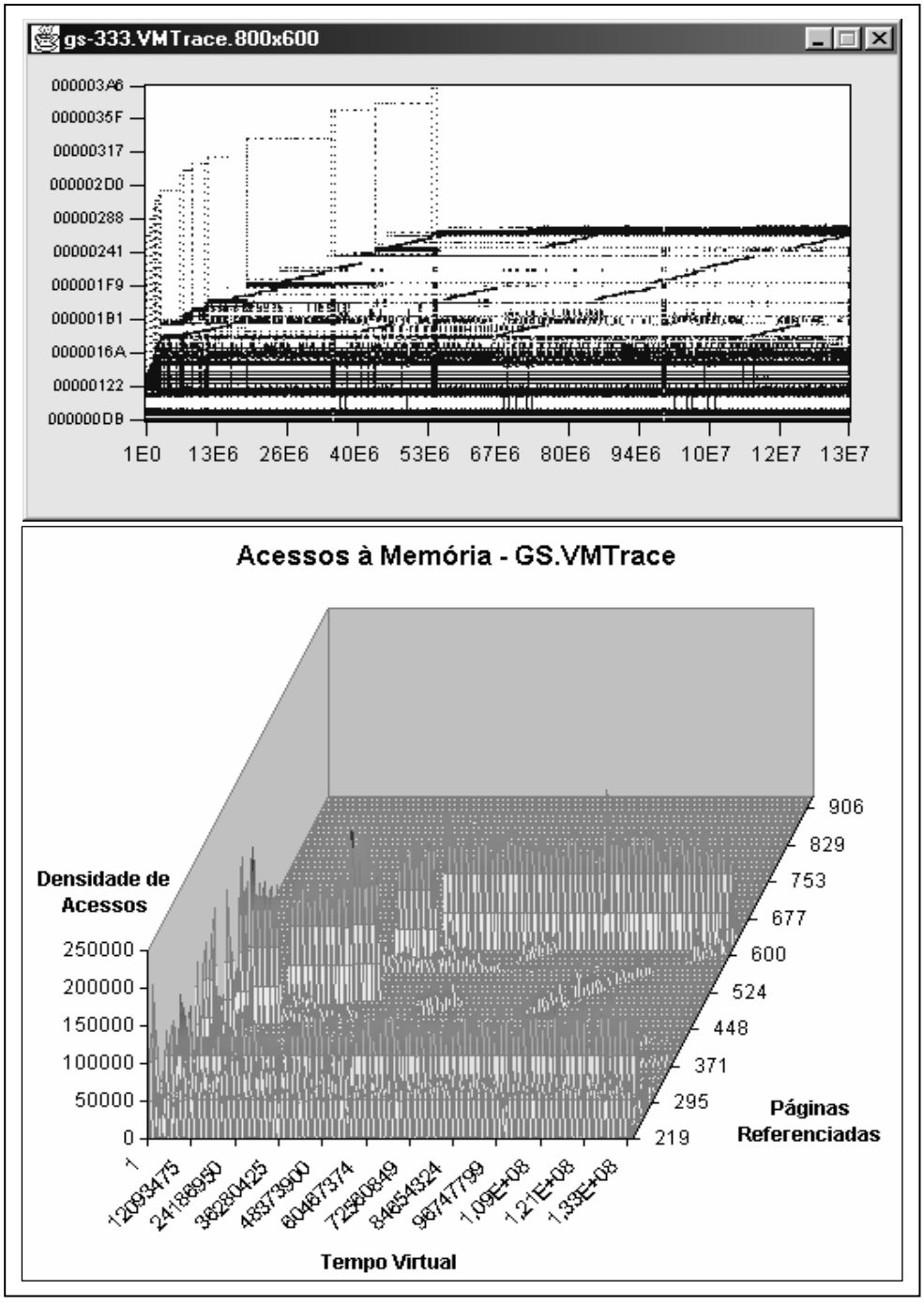

Figura 4. Exemplo de gráficos de acesso bidimensional e tridimensional (GS)

A Figura 4 mostra um exemplo de mapa de acessos bidimensional – gerado pela ferramenta TelaTrace – e um exemplo de mapa de acessos tridimensional – criado a partir da ferramenta Mapa3D e desenhado pelo Microsoft Excel –, ambos referentes ao mesmo programa (GhostScript). O gráfico tridimensional oferece informações muito úteis sobre a distribuição dos acessos entre as diversas posições de memória referenciadas. Todavia, apesar desta grande vantagem sobre os gráficos bidimensionais equivalentes, tal estratégia não é empregada em trabalhos de análise de localidade publicados na literatura.

Portanto, a despeito de sua simplicidade, a ferramenta Mapa3D representa um recurso inovador que contribui diretamente para a realização de estudos mais precisos. Observa-se que os eixos x e y permanecem inalterados em relação aos gráficos tradicionais, ao passo que o eixo z contabiliza o número de acessos aglutinados em cada ponto desenhado. Outros ângulos de observação podem ainda ser utilizados para complementar a visualização padrão (frontal) das superfícies. A Figura 5 demonstra a visão lateral do gráfico de acessos tridimensional exibido na Figura 4. Esta variação angular consegue eliminar o problema da ocultação de algumas áreas.

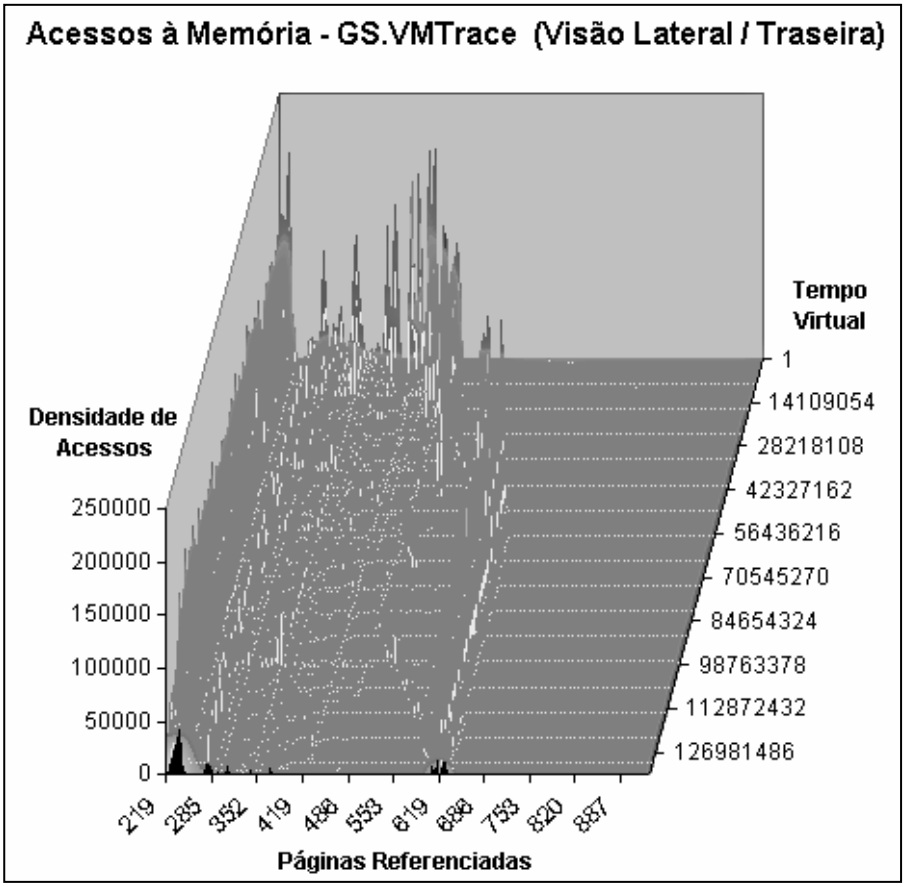

Figura 5. Exemplo de visão lateral em gráfico de acessos tridimensional (GS)

Os mapas de acesso à memória, porém, não fornecem sozinhos todas as informações necessárias para uma análise completa da localidade dos programas. A ferramenta *Trace Explorer* foi então projetada para oferecer um grande número de dados estatísticos e visuais, que podem ser combinados e investigados de acordo com o objetivo da pesquisa do usuário.

3.3. Trace Explorer

Entre outras informações sobre os programas, representados por seus respectivos arquivos de *trace*, a ferramenta Trace Explorer disponibiliza as seguintes saídas:

- *Mapa de recência dos acessos:* Gráfico que indica a posição na fila LRU ocupada pelas páginas acessadas ao longo do processamento do programa, considerando uma memória disponível de tamanho ilimitado. Indica também, por conseqüência, o número de páginas distintas referenciadas entre dois acessos consecutivos a uma mesma página e pode ser chamado ainda de mapa de distâncias LRU (*stack distance*). Apenas a posição da página pontualmente referenciada é considerada em cada instante. A ferramenta gera tanto um mapa bidimensional como um mapa tridimensional.

- *Histograma da recência dos acessos:* Quantifica o número de acessos ocorridos nas várias posições da fila LRU. Além da forma tradicional – em colunas –, pode ser apresentado por meio de uma curva, acumulada ou não, que descreve a quantidade de acessos realizados em função da recência das páginas neles referenciadas.

- *Mapa de variação da recência dos acessos:* Calcula a variação de posição na fila LRU entre dois acessos consecutivos a uma mesma página. Em outras palavras, subtrai a posição na fila que a página ocupava no último acesso da posição observada no acesso corrente.

- *Mapa de distância entre acessos:* Indica a distância entre acessos consecutivos a uma mesma página, determinada através do tempo virtual, isto é, do número de acessos realizados em outras páginas, inclusive acessos repetidos. Este gráfico, portanto, informa o intervalo de tempo que cada página leva para ser reutilizada durante a execução do programa.

- *Relatório analítico*, contendo diversos dados estatísticos sobre os acessos à memória realizados pelo programa, como:
  - Número total de acessos e de páginas referenciadas;
  - Maior e menor página referenciada (endereços);
  - Posição média dos acessos na fila LRU e desvio padrão;
  - Variação média da posição dos acessos na fila LRU e desvio padrão.

Além disso, a ferramenta Trace Explorer ainda trabalha com diversos outros dados analíticos, como o número de páginas distintas referenciadas entre três acessos consecutivos a uma mesma página, a porcentagem de acessos consecutivos a uma mesma página em que a variação da recência é zero – freqüência de reutilização estável –, o número de outras páginas que começam ou deixam de ser referenciadas entre cada acesso consecutivo a uma mesma página – entrada/saída no *working set* pontual do processo –, etc.

Como a ferramenta não possui interface gráfica própria, os mapas citados consistem em coordenadas de pontos gravadas em arquivos de dados. O conjunto destas coordenadas forma um gráfico, que pode ser desenhado por qualquer ferramenta visual, como Gnuplot ou Microsoft Excel. A Figura 6 reúne dois exemplos de mapas, ambos referentes ao programa P2C. O primeiro é um mapa de recência dos acessos realizados e o segundo exibe a variação desta recência entre acessos a uma mesma página de memória. A variação em questão pode ser positiva ou negativa, pois a posição de uma página na fila LRU pode aumentar ou diminuir entre acessos consecutivos a ela.

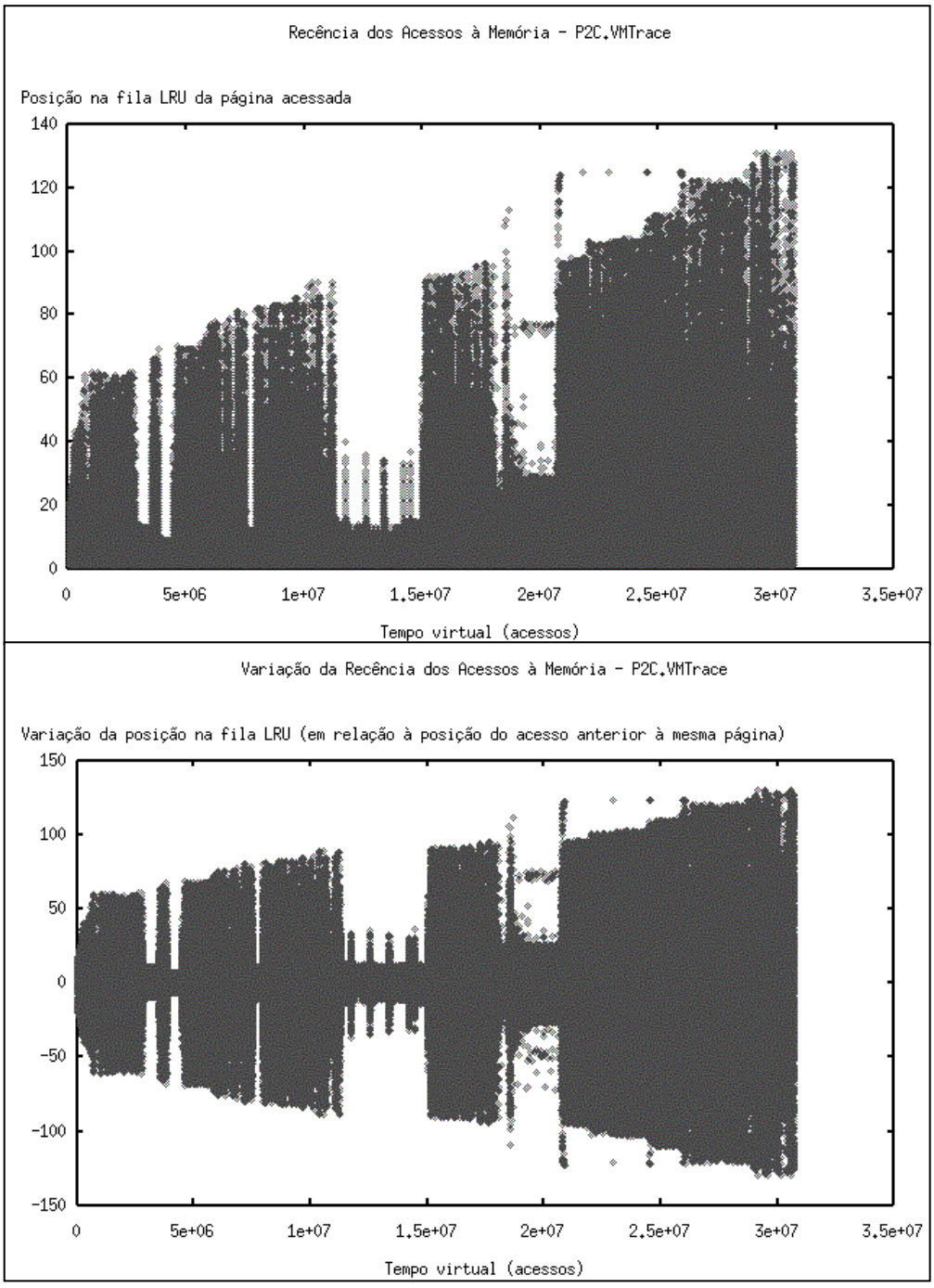

Figura 6. Exemplo de mapas criados pela ferramenta Trace Explorer (P2C)

Alguns arquivos de saída são gerados em duas versões: dados do programa (visão global) e dados específicos de cada página de memória. O relatório estatístico também exibe dados individuais relativos às páginas, as quais são listadas tanto por ordem crescente de endereços, como por ordem decrescente de acessos recebidos. A Figura 7 ilustra dois exemplos de gráficos que se referem ao comportamento de uma página específica associada ao programa P2C: recência dos acessos à página AB e distância (intervalo de tempo) entre estes acessos. O eixo horizontal dos gráficos considera apenas as referências à página analisada e não todos os acessos realizados pelo programa.

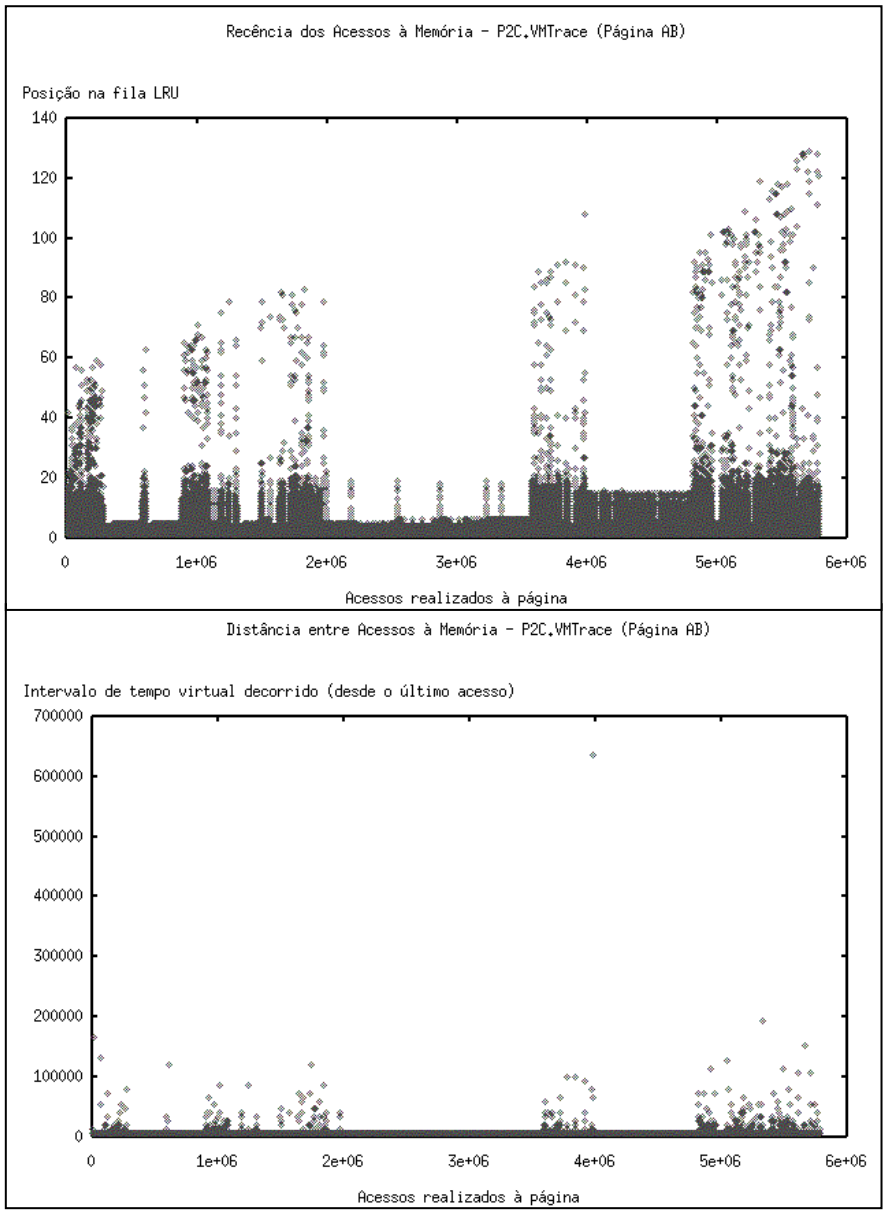

Figura 7. Exemplo de mapas com dados de uma única página (P2C, página AB)

Em conjunto, portanto, as três ferramentas que compõem o pacote Elephantools permitem diversos tipos de análise em programas. Entre outros aspectos relacionados à utilização da memória, elas possibilitam identificar:

- Padrões de acesso;
- Localidades de referência temporais e/ou espaciais, além da amplitude e densidade destas localidades;
- A distribuição de acessos no espaço de endereçamento virtual utilizado pelos programas;
- A freqüência de reutilização das páginas de memória (distância entre acessos);
- A interação de uma página com outras, em termos de acessos entrelaçados;
- O tamanho mínimo de memória que cada programa precisa para que não haja nenhuma necessidade de substituição;
- A posição na fila LRU que as páginas ocupam quando são acessadas.

4. Estudo de Caso

Os principais recursos para análise de localidade oferecidos pelo pacote de ferramentas Elephantools são demonstrados, nesta seção, através de sua aplicação prática no estudo de alguns programas. Sete arquivos que compõem o pacote de *traces* VMTrace representam tais programas, originalmente empregados por Smaragdakis; Kaplan; Wilson (1999) nas simulações do algoritmo adaptativo para substituição de páginas EELRU, sendo também por eles disponibilizados. O nome do pacote designa a ferramenta de geração de *traces* utilizada em sua captação, desenvolvida pelos próprios autores. A seguir são resumidamente descritos os programas analisados, alguns dos quais já referenciados em exemplos de figuras anteriores:

- *Espresso:* Simulador de circuito;
- *GCC:* Compilador C/C++ do projeto GNU, versão 2.7.2;
- *Gnuplot:* Gerador de gráficos do projeto GNU;
- *Grobner:* Programa matemático que trabalha com Bases de Gröbner;
- *GS:* GhostScript 3.33, interpretador PostScript;
- *Lindsay:* Simulador de hipercubo;
- *P2C:* Tradutor de programas em Pascal para C.

A Tabela 1 lista uma série de dados estatísticos a respeito dos *traces* que compõem o pacote VMTrace. Os dados da tabela, coletados pela ferramenta Trace Explorer, caracterizam os aspectos temporal – tempo virtual do processamento – e espacial – espaço de endereçamento ocupado – associados aos arquivos. Informam também algumas estatísticas relacionadas à recência dos acessos à memória, fornecendo subsídios para uma avaliação geral dos programas aplicada ao modelo LRU.

Tabela 1. Estatísticas gerais referentes aos arquivos de *trace* analisados

	Arquivo	Total de Acessos à Memória	Total de Páginas Acessadas	Menor Página (dec./hex.)	Maior Página (dec./hex.)	Posição dos Acessos na Fila LRU			
						Média Geral	Desvio Padrão	Var. Média (módulo)	Var. Zero (% Acessos)
VMTrace	Espresso	326938361	77	100 (64)	180 (b4)	1,82	1,21	0,59	64,67%
	GCC	37524334	458	688 (2b0)	1150 (47e)	3,33	6,90	2,61	39,95%
	Gnuplot	68458509	7718	130 (82)	7857 (1eb1)	5,26	115,74	5,89	34,94%
	Grobner	7787835	67	94 (5e)	166 (a6)	2,07	2,45	1,33	49,01%
	GS	134371942	558	219 (db)	934 (3a6)	2,03	3,63	1,17	65,92%
	Lindsay	123690749	521	89 (59)	614 (266)	2,78	8,32	2,63	57,88%
	P2C	30722431	132	165 (a5)	300 (12c)	3,00	4,97	2,31	51,47%

A Tabela 1 evidencia que todos os *traces*, exceto Gnuplot, possuem a característica de não consumirem muita memória em sua execução, bastando apenas um pequeno espaço de endereçamento virtual para armazená-los. Esta característica facilita a visualização de padrões de acesso presentes em determinados trechos de processamento por meio dos gráficos de acesso. As quatro últimas colunas da tabela também sugerem alta localidade temporal: as páginas são reutilizadas, em média, rapidamente. Isto significa que os acessos à memória realizados pelos programas normalmente se concentram em páginas que ocupam as primeiras posições da fila LRU. Além disso, a última coluna da tabela demonstra que há uma certa regularidade no intervalo entre acessos a uma mesma página. Ou seja, uma página tende a ocupar sempre a mesma posição na fila LRU quando é acessada, sem variações. Conclui-se, então, que algumas poucas páginas são maciçamente referenciadas durante o processamento destes programas, confirmando a existência de significativa localidade temporal. As demais páginas, no entanto, não recebem muitos acessos.

4.1. Padrões de Acesso à Memória

O pacote VMTrace apresenta uma grande variedade de padrões de acesso à memória em seus arquivos, como ilustram os gráficos de acesso exibidos na Figura 8. O traço comum é que os programas tendem a explorar a localidade temporal, ainda que em poucas páginas e com maior ou menor grau de intensidade. Os mapas de acesso tridimensionais visualizados na Figura 9 (e também nas Figuras 4 e 5) reforçam esta constatação: nota-se forte presença de localidade temporal restrita, em geral, a um pequeno conjunto de páginas de memória.

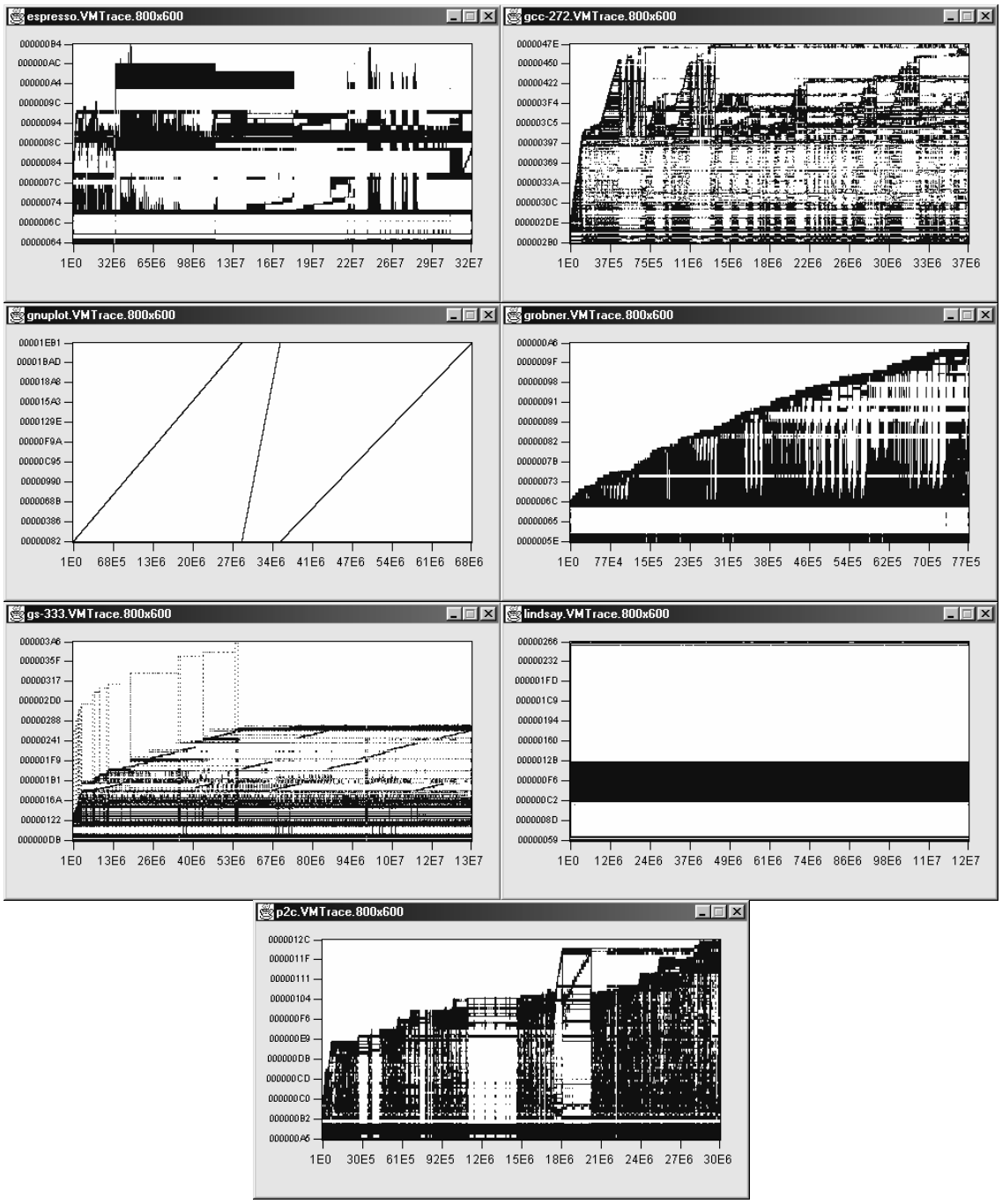

Figura 8. Gráficos de acesso bidimensionais (respectivamente: Espresso, GCC, Gnuplot, Grobner, GhostScript, Lindsay e P2C)

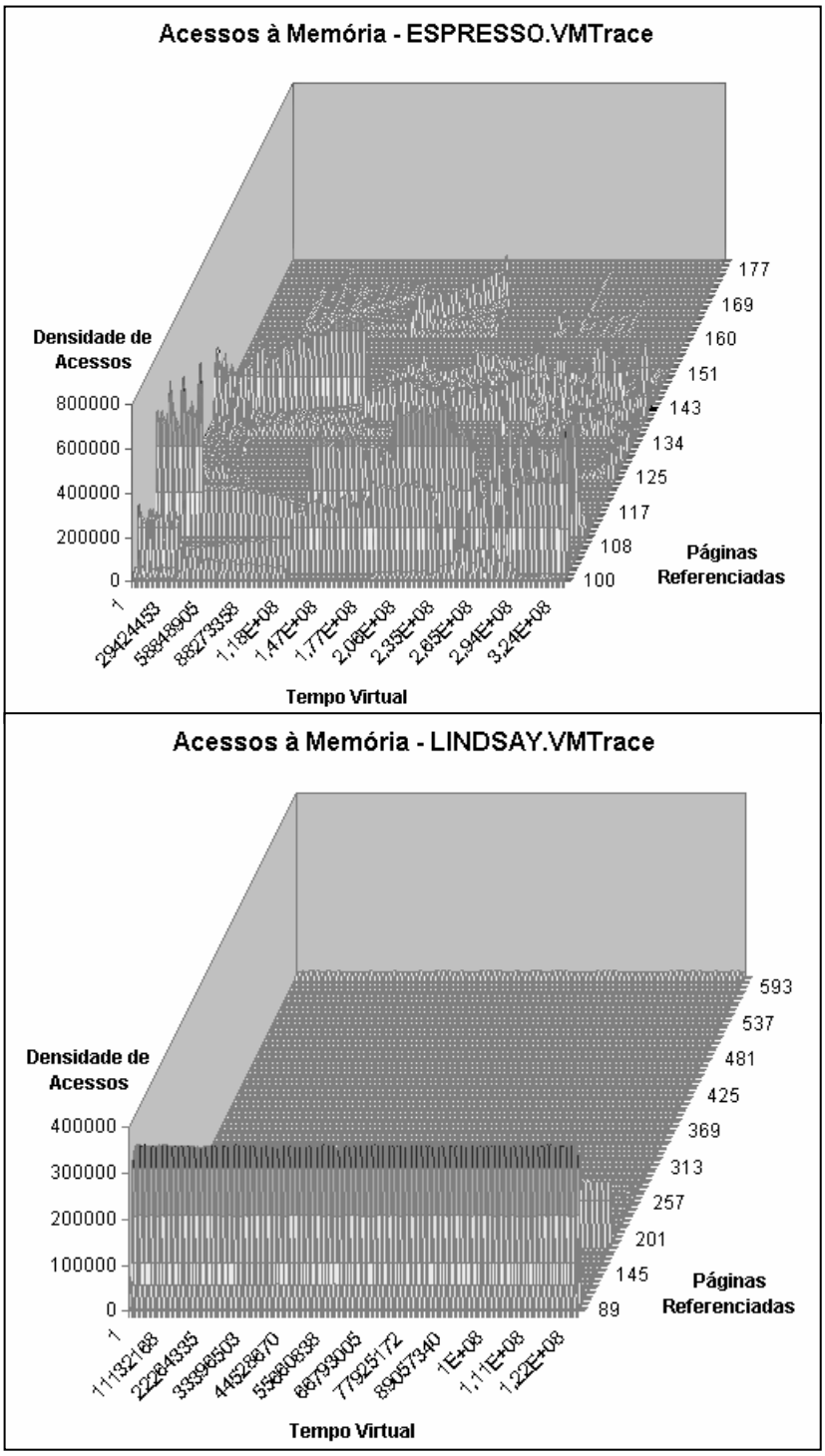

Figura 9. Gráficos de acesso tridimensionais (Espresso e Lindsay)

Os programas Grobner e Gnuplot são os únicos que apresentam padrões seqüenciais de acesso à memória bem definidos – caracterizados por linhas diagonais que permeiam os mapas de acesso –, o primeiro intercalado com referências a muitas outras páginas e o segundo intercalado com acessos a poucas páginas que exibem alta localidade temporal. Em contrapartida, os demais arquivos VMTrace se caracterizam principalmente pelo já comentado aspecto da localidade, sem a presença de outros padrões regulares facilmente identificáveis.

4.2. Inferência de Localidade

Em termos de localidade de referências, os gráficos de acesso bidimensionais oferecem uma visão geral de quais endereços de memória são referenciados pelo programa em execução, enquanto os gráficos tridimensionais informam a densidade de acessos em tais endereços. O gráfico tridimensional, sozinho, nem sempre permite a identificação de todas as páginas que são acessadas pelo programa. Por sua vez, o mapa com duas dimensões não diferencia uma posição do gráfico que recebeu apenas um único acesso, por exemplo, de uma posição que recebeu milhares de acesso – visto que, de acordo com as escalas do gráfico, uma posição pode concentrar até mesmo milhões de acessos aglutinados.

Localidades podem ser detectadas mais facilmente com a utilização dos gráficos tradicionais, com duas dimensões, mas a visualização 3D complementa a análise por informar – através dos valores no eixo z – a quantidade real de referências em cada posição do gráfico. Logo, a melhor maneira de se trabalhar com mapas de acesso é combinando as suas duas formas visuais de apresentação. Na Figura 4, por exemplo, várias linhas horizontais estão presentes no gráfico bidimensional, as quais indicam localidade temporal em diferentes páginas de memória. Entretanto, a visão tridimensional demonstra que somente em algumas destas páginas a localidade é realmente forte, formando paredes horizontais na superfície do gráfico.

Os gráficos de recência também indicam tendências de localidade importantes. Ding; Zhong (2003) analisam localidade como sendo uma função do número de páginas distintas referenciadas entre dois acessos consecutivos a uma mesma página. No modelo LRU, esta medida corresponde à própria posição que a página ocupa na fila LRU – contada a partir de zero – quando é reutilizada. A exibição tridimensional dos mapas de recência destaca as posições do gráfico mais referenciadas e, conseqüentemente, as posições na fila LRU em que a maioria dos acessos à memória acontecem. Como ocorre com os gráficos de acesso, uma análise conjunta das duas formas de apresentação deste mapa – bi e tridimensional – pode esclarecer muitas dúvidas.

Em todos os *traces* do pacote VMTrace, a recência dos acessos é visualizada de forma similar: a grande maioria das referências recai nas primeiras posições da fila LRU. Um exemplo representativo é apresentado na Figura 10, através da visão lateral do gráfico tridimensional relativo ao programa Grobner. O que se percebe é uma parede vertical na extremidade esquerda do gráfico, evidenciando uma forte presença de acessos na porção inicial da fila LRU e, portanto, a ocorrência constante de localidade temporal, ainda que em meio a acessos menos regulares, os quais se tornam mais significativos com o aumento gradativo do *working set* do processo – fato sugerido pela grande quantidade de pontos em posições mais altas da fila no gráfico bidimensional.

Outro gráfico importante é o mapa de recência com dispersão real de acessos, associado sempre a uma única página. Criado através da ferramenta Trace Explorer, trata-se de um gráfico de recência convencional, exceto pelo fato de seu eixo horizontal abranger todos os acessos à memória realizados pelo programa e não apenas aqueles referentes à página analisada. Assim, a distribuição vertical dos pontos do gráfico indica as posições na fila LRU que a página ocupa nos momentos em que é acessada, enquanto a distribuição horizontal corresponde ao tempo decorrido entre os diversos acessos. Por conseguinte, esses mapas – um por página – refletem simultaneamente o aspecto da recência dos acessos no modelo LRU e o aspecto da freqüência destes acessos.

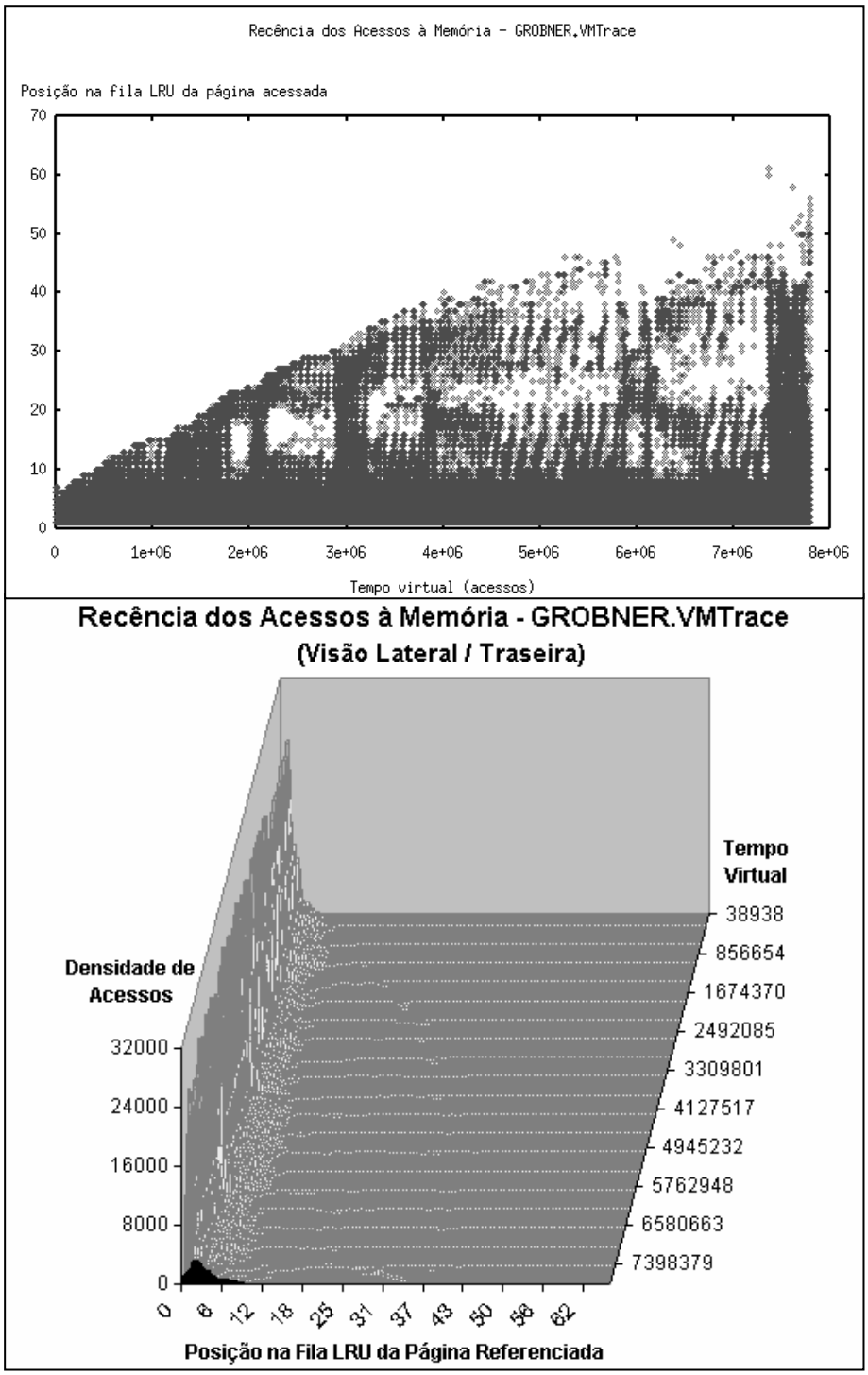

Figura 10. Gráficos de recência bidimensional e tridimensional (Grobner)

A Figura 11 exemplifica um gráfico de recência com dispersão real de acessos, relativo à página mais referenciada pelo programa Grobner (página 5E). O conjunto de seus pontos está contido no conjunto de pontos do gráfico bidimensional ilustrado na Figura 10. Este tipo de mapa representa, pois, a participação de uma página específica no universo da recência de todos os acessos à memória efetuados pelo programa. No caso particular da página 5E, é interessante observar que seu gráfico de recência individual apresenta características semelhantes às do gráfico global.

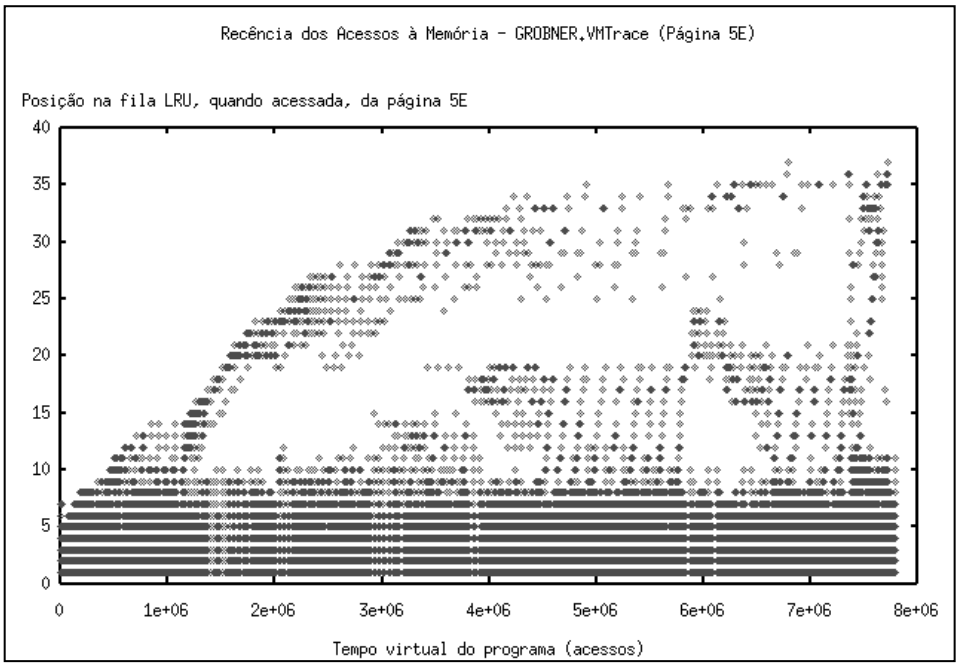

Figura 11. Recência dos acessos à página 5E com tempo virtual absoluto (Grobner)

Há ainda autores, como Phalke (1995), que preferem considerar apenas o intervalo de tempo entre acessos a uma mesma página de memória, isto é, a freqüência de reutilização das páginas. Este dado também é conhecido como IRG (*Inter-Reference Gap*). Uma análise nesse sentido, apesar de não muito significativa para algoritmos de substituição que exploram o modelo LRU, pode ser útil no contexto de algoritmos baseado em freqüência de acessos. Tal análise é facilmente realizada através dos gráficos de distância. Como estes gráficos podem ser gerados individualmente para cada página de memória, tem-se como observar com bastante precisão a freqüência de reutilização em todas elas. A parte inferior da Figura 7 exibe um exemplo de gráfico de distâncias referente a uma única página, cujo endereço hexadecimal é AB. Nota-se que a página é muito utilizada pelo programa P2C e os acessos tendem a se repetir em um curto espaço de tempo. Alguns poucos acessos acontecem após um período maior de inatividade, mas raramente ultrapassam o intervalo de 100.000 referências a outras páginas (tempo virtual).

Finalmente, a recência dos acessos realizados à memória pelos programas também pode ser representada na forma de histograma, saída adicional fornecida pela ferramenta Trace Explorer. Tais histogramas, assim como os mapas de recência, consideram um contexto de memória ilimitada, suficiente para acomodar todas as páginas referenciadas pelo programa. Este recurso merece destaque porque, quando direcionado a estudos específicos acerca de determinados algoritmos de substituição de páginas, permite a previsão de faltas de página em situações genéricas de disponibilidade de memória. Ao definirmos um tamanho de memória M, por exemplo, todo acesso cuja posição na fila LRU for maior que M (baixo grau de recência) resulta necessariamente em falta de página se a política de substituição LRU, ou mesmo FIFO, é adotada. Um outro exemplo: é possível estabelecer um tamanho mínimo de memória para que a taxa de faltas de página permaneça inferior a um certo limite desejado, quando programas previamente conhecidos e analisados são executados.

A Tabela 2 apresenta dados de um histograma relativo ao programa GCC. A primeira coluna é composta por intervalos de posições na fila LRU que, exceto para 0 e 1, seguem a notação "a – b", a qual deve ser interpretada como "de a (inclusive) até b". Ou seja: a ≤ x < b; com x pertencente ao conjunto de posições analisado na linha correspondente. Por exemplo, a linha cujo intervalo é apresentado como "2^5 – 2^6" contém os acessos nos quais a página referenciada ocupava uma posição na fila LRU entre 32 e 63, pois 64 (2^6) não faz parte do intervalo. As posições na fila LRU, neste tipo de estudo, são contadas a partir do zero, visto que assim representam efetivamente o número de páginas distintas referenciadas entre dois acessos consecutivos a uma mesma página.

Tabela 2. Histograma da recência dos acessos na forma tabular (GCC)

GCC.VMTrace - Posição dos Acessos na Fila LRU				
Pos. (a≤x<b)	*Acessos*	*Ac. Acum.*	*% Acessos*	*% Acum.*
0	0	0	0,00	0,00
1	16925646	16925646	45,11	45,11
2^1 - 2^2	11121766	28047412	29,64	74,75
2^2 - 2^3	6551133	34598545	17,46	92,21
2^3 - 2^4	2221110	36819655	5,92	98,13
2^4 - 2^5	443685	37263340	1,18	99,31
2^5 - 2^6	190001	37453341	0,50	99,81
2^6 - 2^7	50445	37503786	0,13	99,94
2^7 - 2^8	18181	37521967	0,05	99,99
2^8 - 2^9	1909	37523876	0,01	100,00

Os dados da Tabela 2 indicam que, na situação de execução representada pelo arquivo de *traces* analisado, uma memória de tamanho 16 já é suficiente para que a taxa de faltas de página se mantenha inferior a 2% durante o processamento do programa, considerando a política de substituição LRU. Porém, se a memória disponível for de pelo menos 64 páginas, é possível garantir que o programa irá causar menos de 0,2% de faltas de página, em relação ao número total de acessos (descontando o primeiro em cada página, quantidade mínima de acessos ao disco), com a utilização da mesma política.

5. Conclusão

Este artigo apresentou o Elephantools, um pacote composto por ferramentas visuais e de coleta de dados estatísticos sobre programas (TelaTrace, Mapa3D e Trace Explorer), muito úteis para a caracterização de cargas de trabalho e em estudos sobre o uso da memória pelos processos. As ferramentas não somente foram descritas, mas também aplicadas na caracterização de programas previamente utilizados em pesquisas sobre gerenciamento de memória virtual. Espera-se que os recursos disponibilizados para análise de localidade auxiliem no desenvolvimento de novos trabalhos sobre o tema e, até mesmo, na explanação simples, intuitiva e didática de conclusões acerca do comportamento de cargas de trabalho quanto ao aspecto da utilização da memória.

Trabalhos futuros incluem o aprimoramento das ferramentas existentes – por exemplo, manipulando formatos específicos de *traces* que atualmente não são reconhecidos, inclusive compactados – e o desenvolvimento de ferramentas adicionais que facilitem e enriqueçam a visualização das chamadas superfícies de localidade *(locality surfaces)*.

Lista de Referências

CASSETTARI, H.H.; MIDORIKAWA, E.T. Análise do padrão de acessos à memória de programas paralelos. In: CONGRESSO BRASILEIRO DE COMPUTAÇÃO, 2., Itajaí, 2002. *II CBCOMP:* Anais. Itajaí: UNIVALI, 2002. 1 CD-ROM.

CHOI, J. et al. An implementation study of a detection-based adaptive block replacement scheme. In: ANNUAL TECHNICAL CONFERENCE, 4., Monterey, 1999. *USENIX' 99:* Proceedings. Monterey: USENIX, 1999. p.239-252.

____. Towards application/file-level characterization of block references: a case for fine-grained buffer management. In: INTERNATIONAL CONFERENCE ON MEASUREMENT AND MODELING OF COMPUTER SYSTEMS, 25., Santa Clara, 2000. *SIGMETRICS' 00:* Proceedings. Santa Clara: ACM, 2000. p.286-295.

DENNING, P.J. The working set model for program behavior. Communications of the ACM, v.11, n.5, p.323-333, 1968.

DING, C.; ZHONG, Y. Predicting whole-program locality through reuse distance analysis. In: CONFERENCE ON PROGRAMMING LANGUAGE DESIGN AND IMPLEMENTATION, 25., San Diego, 2003. *PLDI'03:* Proceedings. San Diego: ACM, 2003. p.245-257.

GLASS, G.; CAO, P. Adaptive page replacement based on memory reference behavior. In: INTERNATIONAL CONFERENCE ON MEASUREMENT AND MODELING OF COMPUTER SYSTEMS, 22., Seattle, 1997. *SIGMETRICS' 97:* Proceedings. Seattle: ACM, 1997. p.115-126.

GRIMSRUD, K. et al. Locality as a visualization tool. IEEE Transactions on Computers, v.45, n.11, p.1319-1326, 1996.

HATFIELD, D.J.; GERALD, J. Program restructuring for virtual memory. IBM Systems Journal, v.10, n.3, p.168-192, 1971.

KIM, J.M. et al. A low-overhead high-performance unified buffer management scheme that exploits sequential and looping references. In: SYMPOSIUM ON OPERATING SYSTEM DESIGN AND IMPLEMENTATION, 4., San Diego, 2000. *OSDI' 2000:* Proceedings. San Diego: USENIX, 2000. p.119-134.

MARKATOS, E.P. Visualizing working sets. ACM SIGOPS Operating Systems Review, v.31, n.4, p.3-11, 1997.

MIDORIKAWA, E.T. *Uma nova estratégia para a gerência de memória para sistemas de computação de alto desempenho*. 1997. 193p. Tese (Doutorado) – Escola Politécnica, Universidade de São Paulo. São Paulo, 1997.

PHALKE, V. *Modeling and managing program references in a memory hierarchy.* 1995. 151p. Thesis (Doutorado) – Graduate School, Rutgers University. New Brunswick, 1995.

SMARAGDAKIS, Y.; KAPLAN, S.; WILSON, P. 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.

SORENSON, E.S.; FLANAGAN, J.K. Cache characterization surfaces and predicting workload miss rates. In: ANNUAL WORKSHOP ON WORKLOAD CHARACTERIZATION, 4., Austin, 2001. *WWC-4:* Proceedings. Austin: IEEE, 2001. p.129-139.

UHLIG, R.A.; MUDGE, T.N. Trace-driven memory simulation: a survey. ACM Computing Surveys, v.29, n.2, p.128-170, 1997.