Trabalho 2 - Memória Virtual

TE244

12 de novembro de 2015

1 Objetivo

Compreender o funcionamento dos diversos algoritmos de substituição de página e estabelecer um comparativo de desempenho através da implementação de um simulador.

2 Descrição do Trabalho

Considere o programa em anexo seguir que simula a *string de referência* para acesso de páginas à memória.

A próxima página a ser acessada pode ser obtida pela chamada à função *getNext()*.

Considere que o sistema em questão possui uma memória física de MAXPAGS/2.

Deve ser implementada uma simulação para registrar o número de falhas de página caso sejam usados os algoritmos:

1. FIFO - First in First Out;
2. Segunda Chance;
3. LRU - Least Recently Used;
4. WSCLOCK.

Após o desenvolvimento dos programas, a simulação ser executada da seguinte forma:

1. Cada execução do programa deve acessar 1000*MAXPAGS.
2. A simulação deve ser executada N vezes.

3. Deve ser calculado o intervalo de confiança para o número médio de falhas de página, considerando que o erro será dado por $e = t_{1-\alpha/2,N-1} \cdot \frac{\sigma}{\sqrt{N}}$, onde $t_{1-\alpha/2,N-1}$ é o valor crítico da distribuição de tstudent para o nível de confiança α desejado e σ é o desvio padrão do número de falhas de página apresentados nas N simulações. Deve ser utilizado uma confiança de 95% (use 95% com uma tabela unicaudal e 90% com uma tabela bicaudal). Caso o erro esteja alto, maior que 5% do número de trocas de página observado, o valor de N deve ser aumentado e o processo repetido.

O resultado deve ser o número médio de falhas de página apresentado por cada algoritmo em estudo, além do erro com nível de confiança desejado. Trabalhos devem se apresentados ao professor, com entrega do código fonte.

1. Funcionamento total: 100 pontos;
2. Três algoritmos implementados, com funcionamento correto: 75 pontos;
3. Dois algoritmos implementados, com funcionamento correto: 50 pontos;
4. Um algoritmo implementado, com funcionamento correto: 25 pontos;

Em caso de cópias, as equipes envolvidas terão grau zero.

3 Programa para Geração da String de Referência

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define MAXPAGS 10000
#define PROB_TROCA_PROC 0.1
#define PROB_JUMP 0.01
#define PROB_LOOP 0.30
#define DIST_LOOP 3
#define PROC 4

//cada processo terá um índice de acesso à memória
int ultimo[PROC]={0, MAXPAGS/4, 2*MAXPAGS/4, 3*MAXPAGS/4};
int proc=0;

double (aleatorio())
{
   return( (double)random()/RAND_MAX);
}

// passagem de parâmetros por referência
void getNext(int *valor, int *bitm, int *bitr)
{
   //simula a preempção de processos
   if (aleatorio()<PROB_TROCA_PROC)
      proc=random()%PROC;

   //simula a realização de um loop pelo processo - vai voltar atrás
   if (aleatorio()<PROB_LOOP)
      ultimo[proc]=ultimo[proc]-random()%DIST_LOOP;

   //simula um jump realizado pelo processo, para uma posição qualquer
   if (aleatorio()<PROB_JUMP)
      ultimo[proc]=random()%MAXPAGS;
   else
      ultimo[proc]++;
   //finalmente, se nada disso aconteceu, o processo acessa a próxima página
   *valor=ultimo[proc];
   if (*valor%100 < 10)
       *bitm=1;
   else
       *bitm=0;
   *bitr=1;
}
```

Para testar o programa:

```
void main()
{
   int i, v, m, r;

   //manter esta linha no início do programa para garantir a aleatoriedade
   srandom(time(NULL));

   // teste de geração de uma sequência
   for (i=0;i<1000;i++) {
     getNext(&v, &m, &r);
     printf("\n Pagina: %d bitM=%d bitR=%d",v,m,r);
   }
   printf("\n\n");
}
```