EP3 de MAC0422

Gabriel Capella (8962078) e Luís Felipe de Melo Costa Silva (9297961)

IME-USP

November 20, 2016

Problema

- Implementar um simulador de gerência de memória com diversos algoritmos para gerência do espaço livre e para substituição de páginas.
- Os algoritmos de gerência de espaço livre são:
  1. First Fit
  2. Next Fit
  3. Best Fit
  4. Worst Fit
- Os algoritmos de substituição de página são:
  1. Optimal
  2. Second-Chance
  3. Clock
  4. LRU (Quarta Versão)

Implementação

- A linguagem escolhida foi Python (versão 3.5.2 - `python3`), que é orientada a objetos e é uma linguagem de script ao mesmo tempo.
- Isso facilitou bastante a leitura do arquivo de trace, no armazenamento das informações, na alocação e desalocação da memória utilizada na execução do programa e no tratamento dos objetos durante os trabalhos.

Módulos

ep3.py

- Implementa o console no terminal, a interação com o usuário.
- É um envelope. Apenas lê as informações e então chama as funções responsáveis pelo comando lido.
- Os comandos são os que foram definidos pelo enunciado (`carrega`, `espaco`, `substitui`, `executa` e `sai`), com os respectivos argumentos.
- Utiliza a biblioteca `cmd` do Python.
- Depende do arquivo gerenciador.py

Módulos

gerenciador.py

- Faz o trabalho pesado pedido pelo módulo anterior (possui as funções que ele chama).
- Possui as seguintes funções:
  - Leitura do arquivo de trace.
  - Escolha do algoritmo de gerência de espaço livre e do algoritmo de substituição de página.
  - Execução do simulador com as informações recebidas.
- Depende dos módulos proccess.py e substitui.py

Módulos

substitui.py

- Possui duas classes: `Pagina` e `Substitui`.
- A classe `Pagina` inicializa o objeto `Pagina` com as seguintes informações: qual memória ela está, o número do processo, o número que o processo enxerga, o bit R, a próxima referência (para o Optimal), onde ele está na memória e quando ela foi chamada pela última vez.
- A classe `Substitui` possui as seguintes funções:
  - Impressão de informações no console.
  - Atribuição das informações lidas no console.
  - Função que tira um processo pronto da fila.
  - Configuração do bit R.
  - Page fault (tem os algoritmos de substituição).
  - Acesso à "memória."
- Depende dos arquivos espaco.py, process.py e gerenciador_arquivo.py

Módulos

substitui.py

- Nesse módulo temos uma lista de unidades de alocação.
- Quando um processo solicita acesso a alguma posição da memória x, calculamos $\left\lceil \frac{x}{s} \right\rceil$ para descobrir a página que o processo quer acessar.
- Exemplo: Seja $x = 82$ e $s = 4$, $\left\lceil \frac{x}{s} \right\rceil = 20$.
- Usando o exemplo, procuramos se alguma unidade de alocação já apresenta a página 20 e seja desse processo.
- Se existir uma unidade que satisfaça o requisito, tentamos passá-la para a memória física.
- Se não existir, a criamos na memória virtual e então tentamos passá-la para a memória física.

Módulos

process.py

- Possui a definição do objeto `process`, que armazena as seguintes informações:
  - PID
  - Nome do processo
  - Início e fim
  - Páginas que ele acessa
  - Quando acessa cada página.

Módulos

gerenciador_arquivo.py

- Esse arquivo cuida de escrever nos arquivos `/tmp/ep3.men` e `/tmp/ep3.vir`. Caso não existam, são criados.
- Possui as funções:
  - Cria os arquivos e os enche com -1.
  - Escreve em `/tmp/ep3.men`.
  - Escreve em `/tmp/ep3.vir`
- **Debug**: Para ver o que os arquivos representam, basta executar o comando `od -i -v tmp/ep3.men` ou `od -i -v tmp/ep3.vir`

Módulos

espaco.py

- Esse módulo tem os algoritmos de gerência de memória implementados.
- Basicamente, possui duas funções:
  - Devolver um endereço de espaço (usando o algoritmo escolhido).
  - Liberar um espaço de memória que não será mais usado.
- Tanto a memória física quanto a virtual implementadas utilizam esse módulo.

Entrada

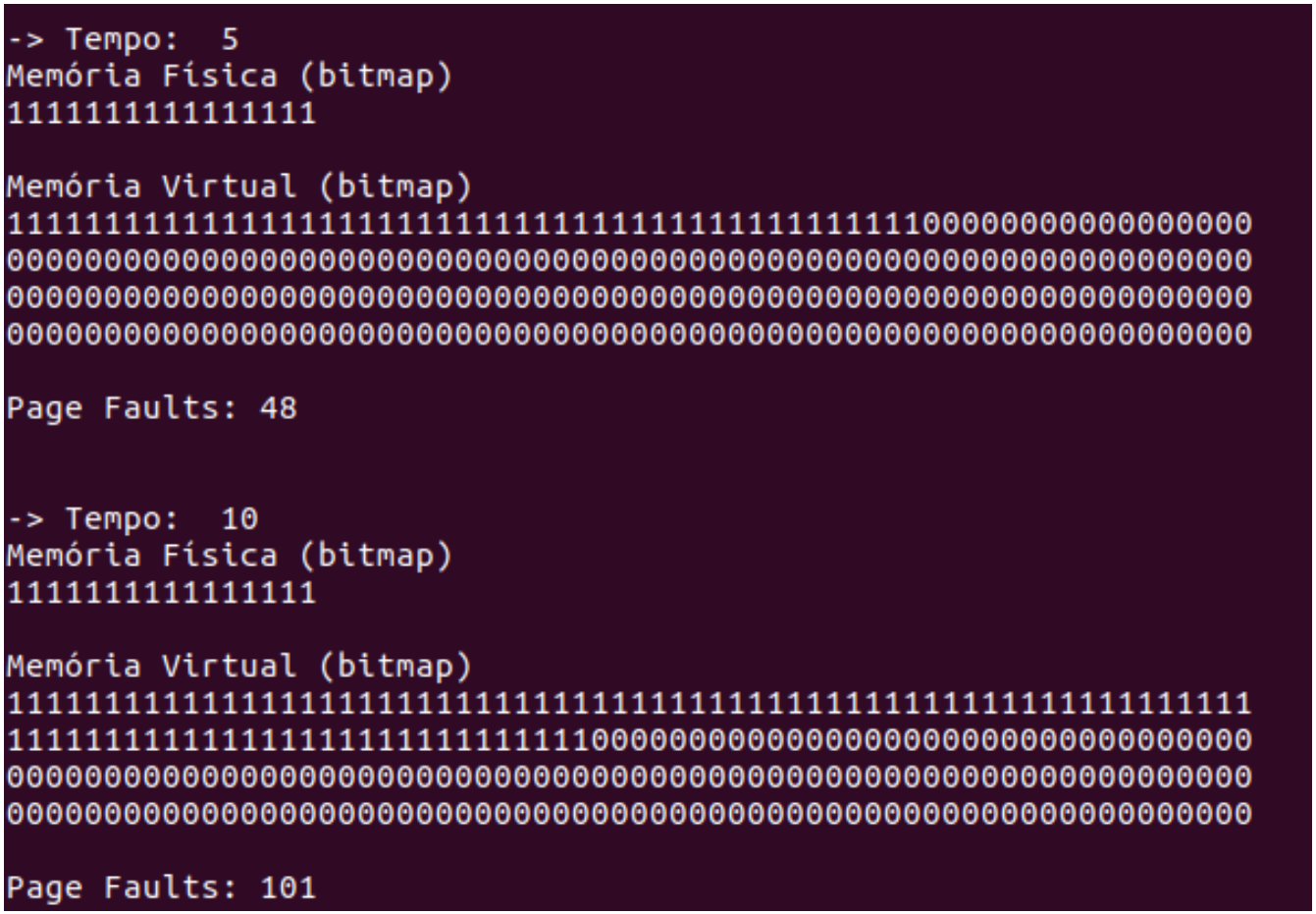
- O módulo do console trabalha com essa parte.
- Um exemplo está abaixo:

```
(ep3): carrega trace
(ep3): substitui 1
(ep3): espaco 4
(ep3): executa 10
```

- Como podemos ver, o arquivo trace será lido, o algoritmo de substituição de página utilizado será o Optimal, o de gerenciamento de espaço livre será o Worst Fit e a impressão da execução será de 10 em 10 "segundos".

Saída

- A imagem abaixo é um exemplo de saída:



Saída

- No arquivo de trace foi definido que a memória física teria tamanho 64, a virtual, 1024, e que o tamanho das páginas (s e p) seria 4.
- Podemos ver então, no bitmap da memória física, 16 bits (64/4) e no da virtual, 256 (1024/4).
- 0 é para posição livre e 1 é para posição ocupada.
- Depois disso é mostrado o número de Page Faults do início da execução até o momento exibido.

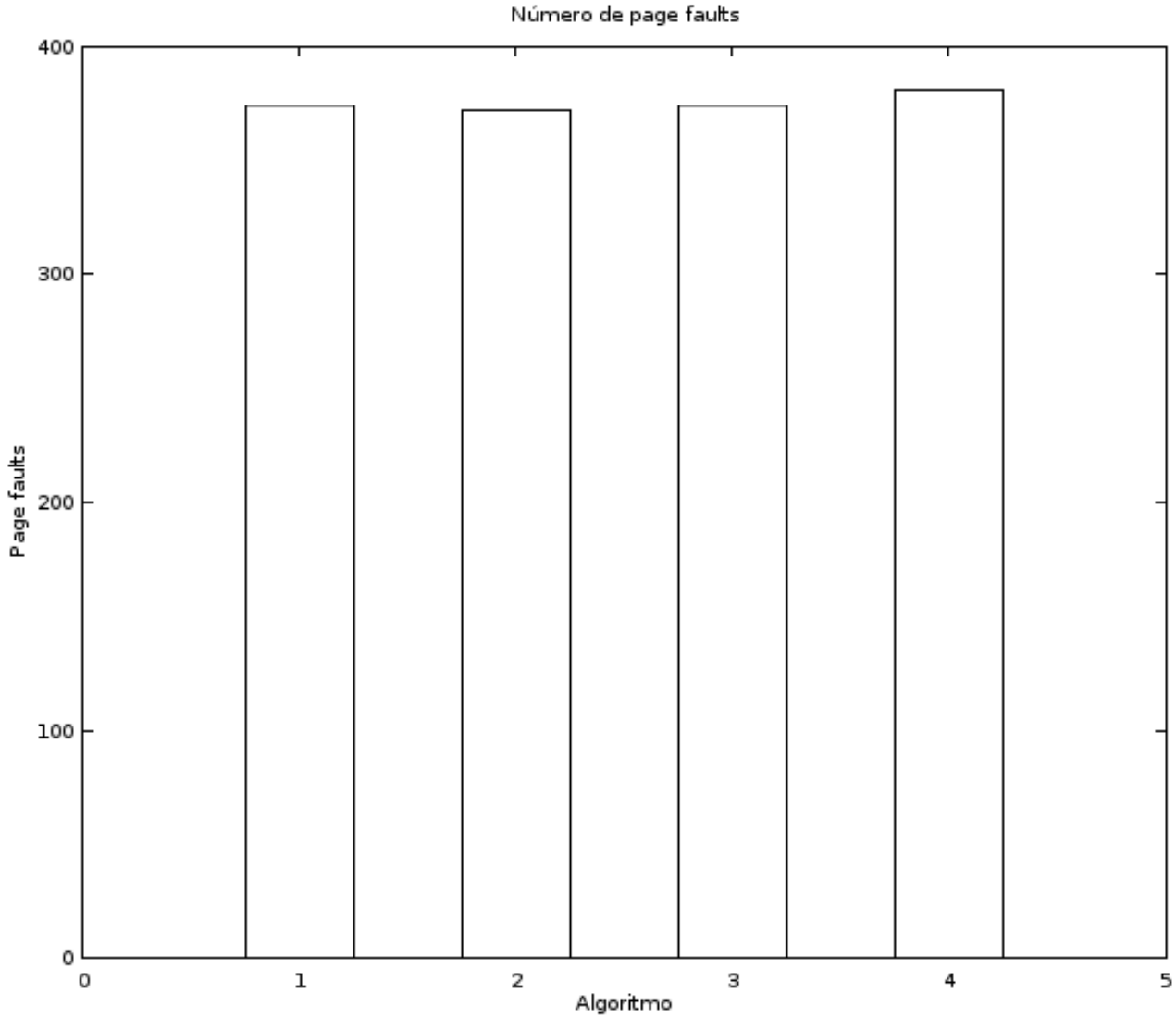
Testes

- Fizemos dois tipos de teste: Número de Page Faults (para comparar os algoritmos de substituição de página) e Tempo de execução (para comparar os algoritmos de gerenciamento de espaço livre).
- Para testar, fizemos um script em Python que substitui o módulo ep3.py, e apenas chamava as funções de gerenciador.py com os argumentos necessários, pela quantidade de vezes que fossem precisas.
- O número de Page Faults não varia em repetidas execuções, pois os algoritmos são determinísticos.
- O tempo de execução varia, pois mesmo que os algoritmos sejam previsíveis, o computador de uso geral mostra variações quando rodamos os comandos.
- O arquivo de trace tinha 100 processos que eram simulados em uma memória física de tamanho 64, tamanho virtual 2048, s e p iguais a 4. Cada processo durava no máximo tempo 15.

Testes

Page Faults

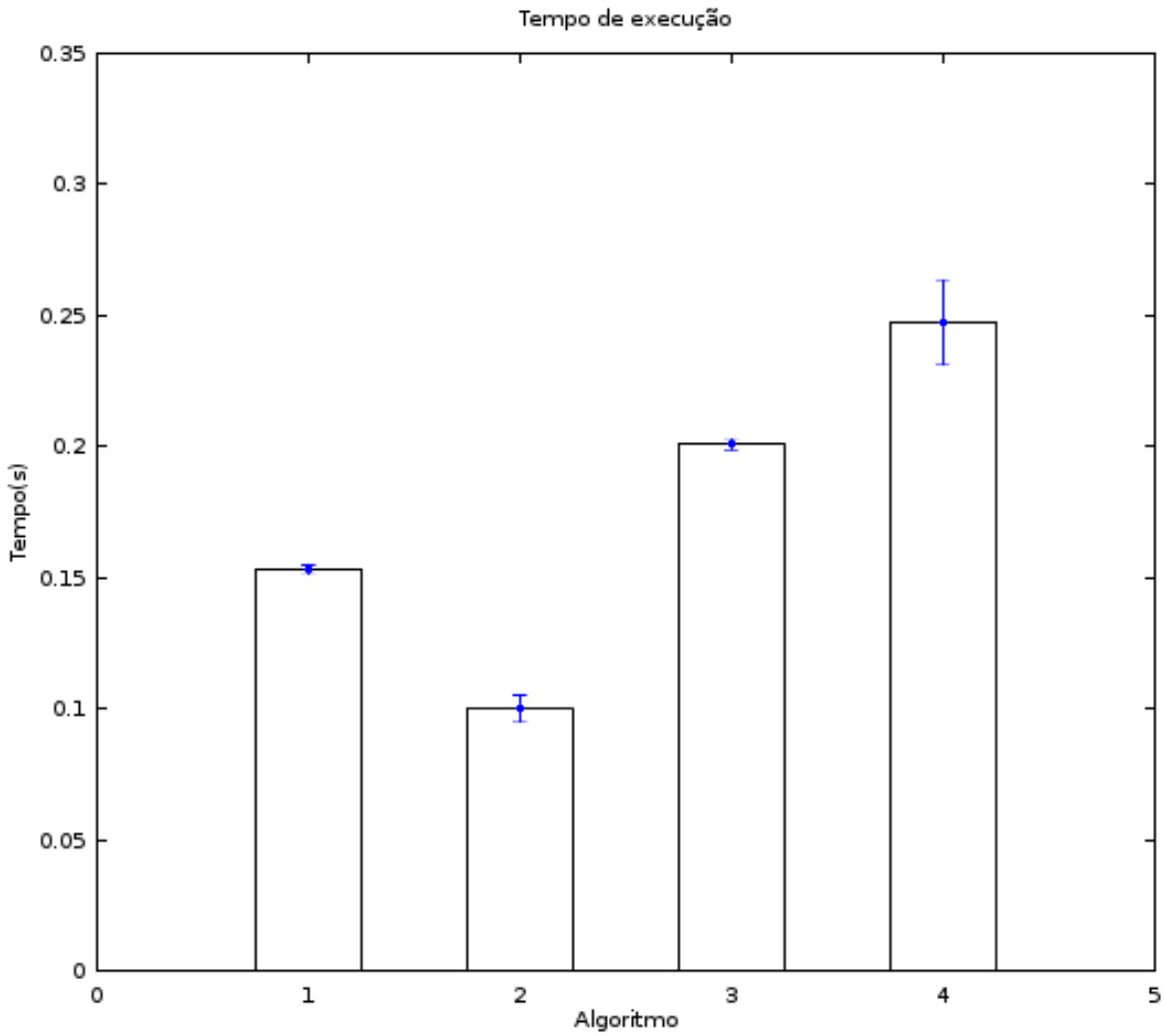
- Estes testes foram executados uma vez.
- Para testar os diferentes algoritmos de forma justa, fixamos um algoritmo de gerenciamento de espaço livre. No caso, todos foram executados com o First Fit.
- Os resultados foram: (legenda para o gráfico a seguir)
  1. Optimal: 374 page faults.
  2. Second-Chance: 372 page faults.
  3. Clock: 374 page faults.
  4. LRU: 381 page faults.


Número de page faults
400
300
200
100
0
Page faults
0
1
2
3
4
5
Algoritmo

Testes

Tempo de execução

- Estes testes foram executados trinta vezes.
- Para testar os diferentes algoritmos de forma justa, fixamos um algoritmo de substituição de página. No caso, todos foram executados com o Optimal.
- Os resultados foram: (legenda para o gráfico a seguir)
  1. First Fit: 0.1531049 s - [0.1511358, 0.1550740].
  2. Next Fit: 0.1000248 s - [0.09488377, 0.10516583].
  3. Best Fit: 0.2010095 s - [0.198864, 0.203155].
  4. Worst Fit: 0.2473311 s - [0.2310442, 0.2636181].
- ALGORITMO: MÉDIA DE TEMPO - INTERVALO DE EXECUÇÃO
- Não compensava medir cada acesso à memória, por isso calculamos o tempo de execução inteiro.


Tempo de execução
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
Tempo(s)
0
1
2
3
4
5
Algoritmo