Relatório de Paginação

Integrantes do grupo
Caio Guilherme dos Santos Silva RA: 10420097
Daniela Pereira da Silva RA: 10410906

1. Introdução

O Projeto de Implementação de Paginação tem como objetivo expandir os conhecimentos sobre a divisão e otimização do uso de memória no contexto de sistemas operacionais, visando uma aprendizagem prática através da implementação de um código de paginação simplificado.

2. Descrição da Implementação

Estruturas de Dados utilizadas

Página

```
typedef struct {
    int presente;
    int frame;
    int modificada;
    int referenciada;
    int tempo_carga;
    int ultimo_acesso;
} Pagina;
```

Processo

```c
typedef struct {
    int pid;
    int tamanho;
    int num_paginas;
    Pagina *tabela_paginas;
} Processo;
```

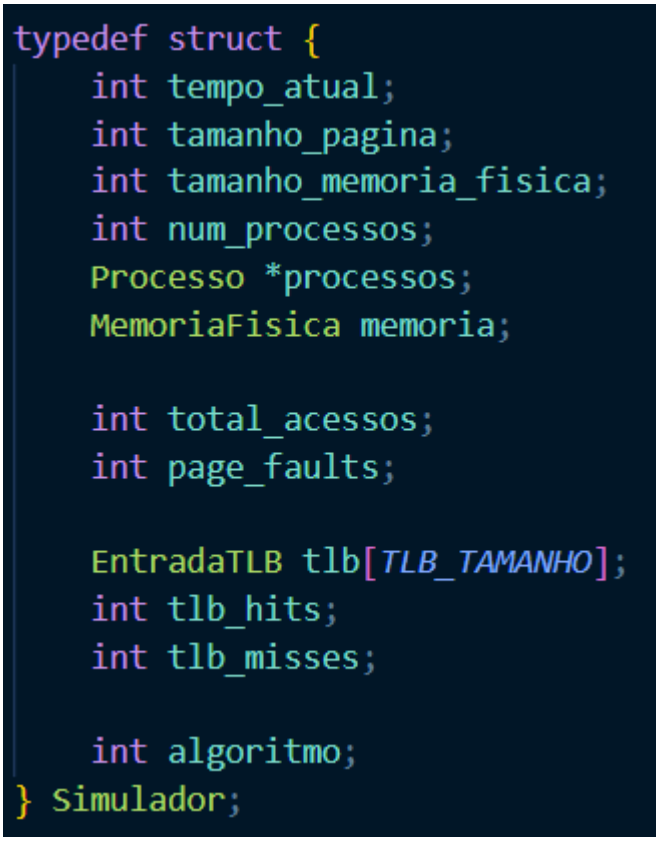
MemoriaFisica

```c
typedef struct {
    int num_frames;
    int *frames;
    int *tempo_carga;
} MemoriaFisica;
```

EntradaTLB

```c
typedef struct {
    int pid;
    int pagina;
    int frame;
    int tempo_carga;
    int ultimo_acesso;
    int valido;
} EntradaTLB;
```

Simulador



```c
typedef struct {
    int tempo_atual;
    int tamanho_pagina;
    int tamanho_memoria_fisica;
    int num_processos;
    Processo *processos;
    MemoriaFisica memoria;

    int total_acessos;
    int page_faults;

    EntradaTLB tlb[TLB_TAMANHO];
    int tlb_hits;
    int tlb_misses;

    int algoritmo;
} Simulador;
```

Algoritmos Implementados

FIFO

O algoritmo FIFO (First-In, First-Out) é uma das abordagens mais simples e intuitivas utilizadas na substituição de páginas em sistemas operacionais. Sua lógica se baseia na ideia de que a página que entrou há mais tempo na memória é a primeira a ser removida quando for necessário liberar espaço. Essa estratégia simula uma fila, onde os elementos são inseridos no final e removidos do início.

Embora sua implementação seja simples e eficiente em termos de tempo de processamento, o FIFO não leva em consideração se uma página foi usada recentemente ou se ainda é relevante para o processo em execução.

LRU

O algoritmo LRU (Least Recently Used) é uma estratégia de substituição de páginas que busca otimizar o uso da memória, removendo a página que está há mais tempo sem ser utilizada. Baseado no princípio da localidade temporal, o LRU assume que páginas acessadas recentemente tendem a ser utilizadas novamente em breve, enquanto aquelas que não são acessadas por um longo tempo provavelmente não serão necessárias imediatamente.

TLB

A TLB (Translation Lookaside Buffer) é um componente de cache, sua principal função é armazenar, de forma temporária, as associações mais recentes entre endereços de páginas virtuais e seus respectivos frames na memória física. Dessa maneira, ela reduz significativamente o tempo necessário para a tradução de endereços, evitando acessos frequentes às tabelas de páginas, que são mais lentos. Sendo implementado à base do algoritmo FIFO.

Decisões de Projeto

Modularização do código
A decisão pela modularização do código veio em visão da melhor organização e legibilidade.

TLB com FIFO
O TLB é implementado com FIFO, em função da simplicidade do código.

Limitações

Apesar de cumprir seu papel educativo e ilustrativo, o simulador de paginação desenvolvido apresenta algumas limitações em relação a um sistema operacional real. A primeira delas é a adoção de uma substituição de páginas de forma global, ou seja, compartilhada entre todos os processos, enquanto sistemas reais podem adotar políticas de substituição local por processo. Além disso, a implementação da TLB utiliza uma política simples de FIFO, sem explorar alternativas mais eficientes como o LRU específico para a TLB.

O simulador também não contempla funcionalidades como gerenciamento de bits de modificação (dirty bit), proteção de memória ou paginação em múltiplos níveis, que são comuns em arquiteturas modernas. Por fim, a interface é apenas textual e voltada para simulação em console, sem uma visualização gráfica dos estados da memória, o que poderia facilitar a compreensão dos eventos de substituição, TLB hits e page faults.

3. Análise Comparativa dos Algoritmos

Comparação entre os algoritmos implementados

A comparação de desempenho entre os algoritmos **FIFO** e **LRU** evidencia claramente as diferenças de eficiência na gestão da memória. O algoritmo FIFO, por sua simplicidade, não

leva em consideração o histórico de uso das páginas, o que frequentemente resulta em substituições de páginas que ainda seriam úteis, elevando o número de **page faults**. Já o algoritmo LRU, por monitorar o tempo do último acesso de cada página, consegue manter na memória aquelas que são mais prováveis de serem reutilizadas, reduzindo assim o número de faltas de página.

Nos testes realizados, o LRU demonstrou desempenho superior, especialmente em cenários com localidade temporal, apresentando uma taxa de page faults menor em comparação ao FIFO. No entanto, essa melhora tem um custo, pois o LRU demanda maior controle e armazenamento das informações de acesso, tornando sua implementação mais complexa. Ainda assim, a redução dos page faults torna o LRU uma escolha mais eficiente para ambientes que priorizam desempenho.

4. Conclusão

O projeto se demonstra como uma forma efetiva e dinâmica de se aprender sobre a implementação de paginação e divisão dinâmica e eficiente de memória

5. Refêrencias

Interação com ChatGPT:
https://chatgpt.com/share/683a6cf1-8da4-800d-b03d-8a3fc5bc2759