Aluno: João Victor Walcacer Giani

Professora: Aldriene Silva

Disciplina: Organização de Computadores

Atividade dos Slides da Aula 07 – 04/09/24

Questão 01 – Resposta: Letra D) Best-fit.

Questão 02 – Resposta: Letra B) 20 KB.

Questão 03 – Resposta: Letra B) Next-fit.

Questão 04 – Resposta: Letra C) Proteção de acesso entre processos.

Questão 05 – Resposta: Letra B) de endereços lógicos do processo em endereços físicos da memória principal.

Questão 06 – Resposta: Letra E) A, B e C estão corretas.

Questão 07 – Resposta: Letra C) Estende o endereçamento de memória através do espaço no disco rígido.

Atividade da Apostila – 07/09/2024

Questão 01 – Resposta: A função da MMU (Memory Management Unit) na gerência de memória é traduzir endereços lógicos (ou virtuais) gerados pelo processador em endereços físicos na memória RAM, permitindo a implementação de memória virtual e proteção de processos, além de gerenciar o acesso à memória e otimizar o uso dos recursos.

Questão 02 – Resposta: O TLB (Translation Lookaside Buffer) é uma cache usada pela MMU para armazenar mapeamentos recentes de endereços lógicos para endereços físicos, reduzindo o tempo de acesso à memória ao evitar buscas demoradas nas tabelas de páginas para cada tradução de endereço.

Questão 03 – Resposta: O TLB é crucial para o desempenho da MMU porque acelera a tradução de endereços lógicos em físicos, reduzindo a latência associada à busca nas tabelas de páginas na memória principal. Sem o TLB, cada acesso de memória exigiria uma busca completa nas tabelas, o que pode ser lento e ineficiente. O TLB armazena mapeamentos recentes, permitindo que a tradução de endereços seja realizada rapidamente, melhorando a eficiência e o desempenho geral do sistema.

Questão 04 – Resposta: Nos métodos de gerenciamento de memória com partições fixas, ocorre fragmentação interna, pois as partições têm tamanhos fixos e podem ter espaço não utilizado dentro delas. Já nos métodos com

partições variáveis, ocorre fragmentação externa, pois o espaço livre na memória é dividido em blocos de tamanhos variados, e o espaço disponível pode ser fragmentado em pequenas partes não contíguas, dificultando a alocação para novos processos.

Questão 05 – Resposta: O endereço físico é o local real na memória RAM onde os dados estão armazenados, enquanto o endereço virtual é o endereço usado pelo processador e pelos programas para acessar a memória. A MMU traduz endereços virtuais em endereços físicos, permitindo que os sistemas operacionais utilizem técnicas como a memória virtual para gerenciar e isolar a memória entre diferentes processos.

Questão 06 – Resposta: Na monoprogramação, apenas um processo é executado na memória principal de cada vez, enquanto na multiprogramação, vários processos são mantidos na memória ao mesmo tempo, permitindo que o sistema operacional alterne entre eles para maximizar o uso dos recursos e melhorar a eficiência do processamento.

Questão 07 – Resposta: Swapping é o processo de mover processos ou partes deles entre a memória RAM e o disco rígido para liberar espaço na memória para novos processos. Isso pode prejudicar o desempenho do sistema computacional porque o acesso ao disco rígido é muito mais lento do que o acesso à memória RAM. Se o sistema realiza swapping com frequência, isso pode causar uma desaceleração significativa, conhecida como "thrashing", onde o sistema passa mais tempo trocando dados entre a RAM e o disco do que realmente executando os processos.

Questão 08 – Resposta: O algoritmo *best-fit* é mais interessante quando queremos minimizar a fragmentação interna, pois ele aloca o menor bloco de memória que é suficiente para o processo, deixando o mínimo de espaço desperdiçado dentro da partição. Já o *worst-fit* é mais útil quando queremos minimizar a fragmentação externa, pois ele aloca o maior bloco de memória disponível, esperando que os blocos restantes sejam grandes o suficiente para futuros processos. No entanto, *worst-fit* pode resultar em mais fragmentação interna se o espaço restante for muito pequeno.

Questão 09 – Resposta:

Algoritmo *First-Fit*:

Espaço desperdiçado total (First-Fit): 288K + 183K + 188K = 659K.

Algoritmo Best-Fit:

Espaço desperdiçado total (Best-Fit): 88K + 83K + 88K + 174K = 433K.

Algoritmo *Worst-Fit*:

Espaço desperdiçado total (Worst-Fit): 388K + 83K + 188K = 659K.

Questão 10 – Resposta:

a. Algoritmo *First-Fit*:

1. 5K: Aloca na primeira partição que cabe, 10K. Memória após alocação: [5K, 4K, 15K, 18K, 7K, 9K, 12K, 13K].
2. 10K: Aloca na próxima partição que cabe, 15K (originalmente 20K). Memória após alocação: [5K, 4K, 10K, 8K, 7K, 9K, 12K, 13K].
3. 15K: Aloca na próxima partição que cabe, 18K (originalmente 18K). Memória após alocação: [5K, 4K, 10K, 8K, 7K, 9K, 12K, 13K].
4. 8K: Aloca na próxima partição que cabe, 9K (originalmente 9K). Memória após alocação: [5K, 4K, 10K, 8K, 7K, 1K, 12K, 13K].
5. 3K: Aloca na próxima partição que cabe, 4K (originalmente 4K). Memória após alocação: [5K, 3K, 7K, 1K, 12K, 13K].
6. 7K: Aloca na próxima partição que cabe, 12K (originalmente 12K). Memória após alocação: [5K, 3K, 7K, 5K, 13K].
7. 6K: Aloca na próxima partição que cabe, 13K (originalmente 13K). Memória após alocação: [5K, 3K, 7K, 5K, 6K].

Espaço desperdiçado (First-Fit): 5K + 1K + 5K = 11K.

b. Algoritmo *Best-Fit*:

1. 5K: Aloca na partição mais próxima que cabe, 7K. Memória após alocação: [10K, 4K, 20K, 18K, 2K, 9K, 12K, 13K].
2. 10K: Aloca na partição mais próxima que cabe, 12K. Memória após alocação: [10K, 4K, 20K, 18K, 2K, 9K, 2K, 13K].
3. 15K: Aloca na partição mais próxima que cabe, 20K. Memória após alocação: [10K, 4K, 5K, 18K, 2K, 9K, 2K, 13K].
4. 8K: Aloca na partição mais próxima que cabe, 18K. Memória após alocação: [10K, 4K, 5K, 10K, 2K, 9K, 2K, 13K].
5. 3K: Aloca na partição mais próxima que cabe, 4K. Memória após alocação: [10K, 1K, 5K, 10K, 2K, 9K, 2K, 13K].

6. 7K: Aloca na partição mais próxima que cabe, 9K. Memória após alocação: [10K, 1K, 5K, 10K, 2K, 2K, 2K, 13K].
7. 6K: Aloca na partição mais próxima que cabe, 13K. Memória após alocação: [10K, 1K, 5K, 10K, 2K, 2K, 6K].

Espaço desperdiçado (Best-Fit): 1K + 2K + 2K = 5K.

c. Algoritmo *Worst-Fit*:

1. 5K: Aloca na maior partição, 20K. Memória após alocação: [10K, 4K, 15K, 18K, 7K, 9K, 12K, 13K].
2. 10K: Aloca na maior partição restante, 18K. Memória após alocação: [10K, 4K, 15K, 8K, 7K, 9K, 12K, 13K].
3. 15K: Aloca na maior partição restante, 13K. Memória após alocação: [10K, 4K, 8K, 7K, 9K, 12K, 2K].
4. 8K: Aloca na maior partição restante, 12K. Memória após alocação: [10K, 4K, 8K, 7K, 9K, 4K].
5. 3K: Aloca na maior partição restante, 7K. Memória após alocação: [10K, 4K, 8K, 4K, 9K].
6. 7K: Aloca na maior partição restante, 9K. Memória após alocação: [10K, 4K, 8K, 4K, 2K].
7. 6K: Não há partições disponíveis restantes que sejam grandes o suficiente.

Espaço desperdiçado (Worst-Fit): 0K.

d. Algoritmo *Next-Fit*:

1. 5K: Aloca na primeira partição suficiente após o último alocamento, 10K. Memória após alocação: [5K, 4K, 20K, 18K, 7K, 9K, 12K, 13K].
2. 10K: Aloca na próxima partição suficiente, 20K. Memória após alocação: [5K, 4K, 10K, 18K, 7K, 9K, 12K, 13K].
3. 15K: Aloca na próxima partição suficiente, 18K. Memória após alocação: [5K, 4K, 10K, 3K, 7K, 9K, 12K, 13K].
4. 8K: Aloca na próxima partição suficiente, 12K. Memória após alocação: [5K, 4K, 10K, 3K, 7K, 9K, 4K, 13K].
5. 3K: Aloca na próxima partição suficiente, 4K. Memória após alocação: [5K, 1K, 7K, 9K, 4K, 13K].

6. 7K: Aloca na próxima partição suficiente, 13K. Memória após alocação: [5K, 1K, 7K, 9K, 6K].
7. 6K: Não há partições disponíveis restantes que sejam grandes o suficiente.

Espaço desperdiçado (Next-Fit): 1K + 3K + 6K = 10K.