Instituto Federal de Alagoas - IFAL
Campus Maceió
Estrutura de Dados
Prof. Ivo Calado

Lista de exercícios – Semana 04

10 de fevereiro de 2021

1) Na área de Sistemas Operacionais um dos aspectos mais interessantes é como como a fragmentação de memória vai ocorrendo ao longo do tempo conforme novos processos são criados e terminados. Nesse aspecto, é de atribuição do Sistema Operacional o monitoramento das áreas de memória alocadas e livres a fim de determinar que áreas de memória estão disponíveis para alocação para um novo processo criado.

   Para a realização de tal gerenciamento, duas funcionalidades devem ser providas pelo Gerenciador de Memória. São elas:

   (a) A forma de armazenamento da informação das áreas alocadas e livres;

   (b) O algoritmo de busca de áreas disponíveis.

   Sobre a forma armazenamento das informações, as duas principais estratégias a serem utilizadas são o **Mapa de Bits** e as **Listas Encadeadas**. Especificamente sobre a estratégia de **Listas Encadeadas** ela funciona como segue. Considere que a memória é formada por N blocos que são marcados como **livres** ou **ocupados**.

   Por exemplo, na Figura 1 apresenta-se o estado de uma memória na qual 5 processos (A, B, C, D e E) estão alocados (topo da figura) e como tal informação pode ser armazenada. A forma de armazenamento da informação sobre os blocos de memória consiste da utilização de uma única lista encadeada onde cada elemento da lista mantém o estado do bloco de memória (livre ou ocupado), o início do bloco, a extensão do bloco e o identificador do processo.

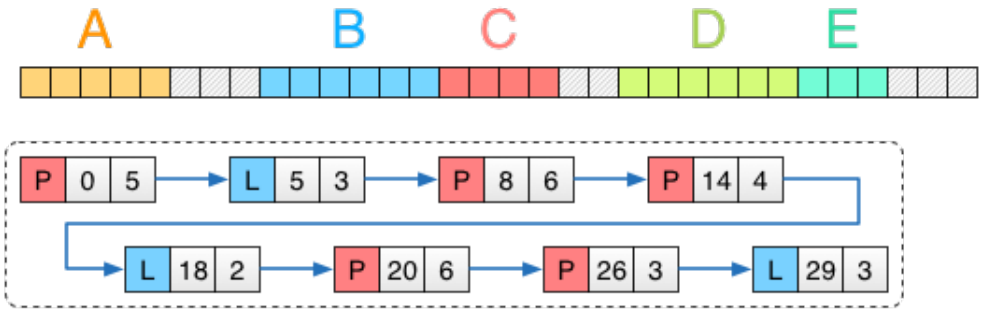

Figura 1: Modelo de memória em blocos e utilização de listas encadeadas.

É importante destacar que para o gerenciador de memória não faz sentido manter dois blocos consecutivos disponíveis, de tal modo que no cenário que dois blocos consecutivos se tornem disponíveis, o gerenciador de memória deve aglutinar os blocos em um único bloco. Por exemplo, considerando o cenário anterior, suponha um cenário em que o processo B seja encerrado, o modelo de memória deve ser o apresentado na Figura 2.

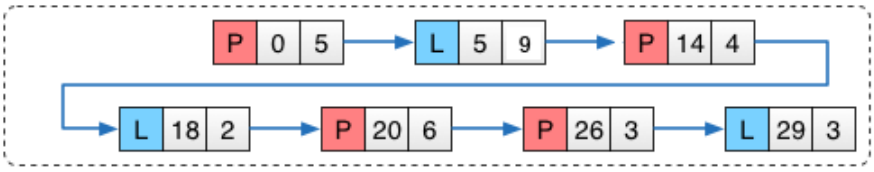

Figura 2: Lista encadeada após a liberação de um espaço.

Por fim, além do armazenamento, o gerenciador de memória deve especificar qual algoritmo de busca por espaços disponíveis devem ser utilizados. Tais algoritmos recebem como entrada o espaço de memória desejado e retorna o bloco que atende ao requisito. Os principais algoritmos são os seguintes:

- *first_fit*: busca o primeiro bloco disponível que atende ao tamanho desejado.
- *best_fit*: busca o bloco que melhor encaixa para o tamanho buscado.
- *worst_fit*: busca o bloco que pior encaixa para o tamanho buscado.

Com base no que foi apresentado, o trabalho da equipe é o desenvolvimento de TAD que represente o gerenciador de memória. Ao ser inicializado o TAD deverá receber um inteiro representando o tamanho da memória em número de blocos. Fazendo uso de uma lista duplamente encadeada, desenvolvida pela própria equipe, o TAD deve apresentar as seguintes funcionalidades:

- *def criar_novo_processo(id_processo, tamanho, estrategia_alocacao)*:
  - **id_processo**: representa o identificador do processo a ser criado;
  - **tamanho**: representa o tamanho do processo, em número de blocos a ser criado;
  - **estrategia_alocacao**: pode assumir os valores *first_fit*, *best_fit* e *worst_fit* representando a estratégia a ser seguida pelo gerenciador de memória.

    O referido método deve fazer alocação de memória para o processo modificando a lista encadeada interna a fim de representar a memória após a criação do novo processo. O método deverá retornar **verdadeiro** em caso de sucesso na alocação (ie, conseguir encontrar um bloco disponível que caiba o novo processo e **falso**, caso contrário.
- *def deletar_processo(id_processo)*:
  - **id_processo**: representa o identificador do processo a ser deletado.

    O gerenciador deverá buscar e liberar o espaço necessário, fazendo a devida aglutinação dos blocos de memória disponíveis.
- *def imprimir_dump_memoria()*: O método deve imprimir o status atual da memória, descrevendo os blocos, o tamanho de cada bloco e o status (ocupado (O)/disponível (D)), conforme o exemplo abaixo:

  `[O, 0, 5] -> [D, 6, 9] -> [O, 14, 5]`

Observação: Será permitido grupos de até 3 integrantes para a implementação da atividade supracitada.