_____ / _____ / _____

Mecanismos de alocação

First Fit

Este algoritmo percorre a lista de áreas livres disponíveis na memória, até encontrar o primeiro espaço livre que acomode a requisição de memória que foi efetuada. Supondo que uma requisição de 120 inteiros tenha sido feita, este algoritmo entregaria a primeira área livre encontrada que fosse maior ou igual a 120. A sigla FIRST-FIT pode ser traduzida por o "primeiro que cabe. A principal vantagem dessa estratégia é a rapidez com que as solicitações são atendidas. Como desvantagem pode-se apontar o caráter randômico das alocações, o que deixa ao acaso o mapa geral de alocações.

Best Fit Este algoritmo percorre toda a lista de áreas livres fazendo a alocação pedida naquela área que minimize a sobra de espaço (a quantidade de inteiros perdidos ao final do bloco alocado). Por exemplo, digamos que uma requisição de 80 inteiros tenha sido feita, e ao percorrer a lista de áreas livres tenham sido encontradas as seguintes: 100, 280, 90, 500 e 320. Este algoritmo escolheria a terceira área, que é a que menor área desocupada deixaria ao final (apenas 10 inteiros). A sigla BEST-FIT pode ser traduzida por "o menor que cabe. A vantagem dessa estratégia é a conservação de blocos grandes. Por outro lado, ele tem como desvantagem a necessidade de pesquisar toda a lista a cada alocação, além de fragmentar a memória em muitos blocos pequenos, cada um deles podendo ter a sua sobra ao final. Este esquema privilegia as grandes alocações, no sentido de que requisições grandes tem boa chance de serem atendidas.

Worst Fit Este algoritmo percorre toda a lista de áreas livres e aloca a maior área que tenha sido encontrada na cadeia de áreas livres e que comporte a requisição de espaço solicitada. Usando o mesmo exemplo do caso anterior (solicitando-se 80, e comportando a lista os valores 100, 280, 90, 500 e 320) seria alocada a área de 500 inteiros, que passaria a contar agora apenas com 420 inteiros. A sigla WORST-FIT pode ser traduzida por "o pior que cabe. A vantagem desta estratégia é que ela não gera blocos muito pequenos, mas em compensação ela tende a fragmentar os blocos grandes, além é claro, de ter que percorrer toda a cadeia de áreas livres para achar "a pior".

Variantes Como toda algoritmo associado a uma estrutura de dados, as estratégias acima comportam diversas modificações. O objetivo sempre é encontrar um ponto de equilíbrio entre objetivos opostos quais sejam: minimizar o tempo gasto na alocação e maximizar o uso da memória. As variantes mais conhecidas são:

- Existência de um valor mínimo de alocação. Sempre que a diferença entre o valor pedido na alocação e o tamanho do bloco selecionado para atender aquela alocação for menor que um valor fixo, todo o bloco será alocado. Essa modificação, se de um lado desperdiça certas áreas ao final das alocações, por outro lado mantém a lista de blocos livres pequena - ou pelo menos, menor do que seria sem esse conceito.
- Ajuste de tamanho. Antes de se proceder ao atendimento da solicitação de alocação, o tamanho pedido é arredondado - para cima - para um valor múltiplo de alguma constante. Essa estratégia também garante a minimização de pequenos pedaços ao final dos blocos alocados.
- Encontro imediato. Nos algoritmos de BEST-FIT ou WORST-FIT, sempre que na pesquisa é encontrado um bloco com tamanho igual ao pedido (ou com diferença menor que uma dada constante) o bloco é imediatamente alocado, sem ter que se proceder à toda pesquisa na lista de blocos livres.
- Segregação de memória. Os blocos são previamente criados e posteriormente não são divididos. Por exemplo, cria-se uma coleção de blocos de 100 inteiros, outra de 500, outra de 1000 e assim por diante.

Estratégias de alocação: exemplo prático Seja o problema do empacotamento no supermercado: Existe uma compra de objetos cujos pesos variam entre 1Kg e 10Kg. Cada sacola de mercado agüenta 15 Kg, e obviamente há um interesse do mercado em minimizar o uso de sacolas. Lembre que: esta-se abstraindo a questão da dimensão da compra...

Obviamente este problema se aplica a objetos em containers, pessoas em elevadores, alunos em vans, etc etc

A solução ótima deste problema tem complexidade ... maior do que 2^n, mas existem algoritmos mais rápidos e que são satisfatórios. Por exemplo, Suponha uma seqüência de M pacotes pesando $\frac{S}{2} - \epsilon$, seguida de outra seqüência de M pacotes pesando $\frac{S}{2} + \epsilon$ (S=limite da sacola e ϵ=um pequeno valor, por exemplo 0,01 Kg). Está claro que a solução ótima é usar M sacolas. Mas, na seqüência como foi dada, cada dois pacotes da primeira metade ocuparão uma sacola, e depois, cada pacote da segunda metade usará 1 sacola.

Um exemplo Seja uma instância do problema: Compra: 2, 5, 4, 7, 1, 3 e 8 Kg. Uma possível resposta para sacolas com limite máximo de 10Kg é S_1=8 e 2 Kg; S_2=3 e 7Kg e S_3=5, 1 e 4 Kg.

Existem 2 versões deste algoritmo, chamadas ON-LINE e OFF-LINE. Na primeira cada pacote deve ser colocado na sua sacola definitiva, ANTES que chegue o segundo pacote.

Na versão OFF-LINE, todos os pacotes estão disponíveis antes de começar a distribuição. **Estratégia 1: NEXT FIT** É a mais simples: ao chegar um pacote, verifica-se se ele cabe na sacola atual. Se couber, vai, senão fecha a sacola atual e começa outra. Esta estratégia nunca usa mais do que $2 \times X$ sacolas (X=número ótimo de sacolas). Questão: quantas sacolas esta estratégia usaria no exemplo acima ? R=_____.

Estratégia 2: FIRST FIT Verifica todas as sacolas já usadas para ver se o pacote cabe em alguma. A primeira possível é a escolhida. Se nenhuma serve, começa-se nova sacola. Esta estratégia nunca usa mais do que $1.7 \times X$ sacolas. Questão: idem. R=_____.

Estratégia 3: BEST FIT Verifica todas as sacolas abertas. Escolhe a que tem o menor espaço onde este pacote cabe. Nunca mais do que $1.7 \times X$, mas este número é quase nunca alcançado freqüentemente. Questão: idem. R=_____.

Estratégia 4: WORST FIT Verifica todas as sacolas abertas. Escolhe a que tem o maior espaço onde o pacote cabe.

Estratégia 5: Versão OFFLINE Os pacotes devem ser ordenados em ordem decrescente. Daí aplica-se a estratégia FIRST ou BEST. Ambas dão o mesmo resultado, logo usar-se-á aqui a FIRST. O desempenho aqui nunca é maior do que $\frac{4 \cdot X + 1}{3}$ sacolas.

Acompanhe o exemplo Seja o seguinte conjunto de compras com o limite da sacola=15Kg.

1	3	3	8	3	10	4	6	8	3
7	2	7	10	2	7	5	6	4	6

Que dará o seguinte resultado em número de sacolas:

	NEXT	FIRST	BEST	WORST	OFF
1	1,3,3,8	1,3,3,8	1,3,3,8	1,3,3,8	10,5
2	3,10	3,10,2	3,10,2	3,10	10,4,1
3	4,6	4,6,3,2	4,6,3,2	4,6	8,7
4	8,3	8,7	8,7	8,3	8,7
5	7,2	7,7	7,7	7,2,6	7,6,2
6	7	10,5	10,5	7,2,4	6,6,3
7	10,2	6,4	6,4	10	4,3,3,3,2
8	7,5	6	6	7,5	
9	6,4			6	
10	6				

Um exemplo completo para você ver Seja o seguinte conjunto de compras com limite da sacola=15Kg)

9	7	3	8	1	2	5	1	10	5
2	3	8	1	9	9	4	1	2	3

Veja:
Para o método NEXTFIT, a quantidade de sacolas é: **9**
Para o método FIRSTFIT, a sacola número 3 terá peso: **15**
Para o método BESTFIT, a primeira sacola terá: **4 itens**
Para o método WORSTFIT, a última sacola terá peso: **11**
Para o método OFFFIRSTFIT, a sacola mais CHEIA terá: **4 itens**.
Obs: mais CHEIA significa com mais itens (não necess. a mais pesada)

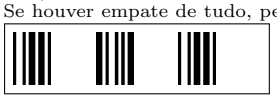
☞ - Para você fazer

Seja o seguinte conjunto de compras com limite da sacola=15Kg)

2	10	5	6	1	4	9	6	9	7
9	4	8	7	10	9	9	1	6	2

1. Para o método **NEXTFIT**, a quantidade de sacolas é de _____ sacolas.
2. Para o método **FIRSTFIT**, a sacola número 5 terá peso de _____ kilos.
3. Para o método **BESTFIT**, a primeira sacola terá _____ itens.
4. Para o método **WORSTFIT**, a última sacola terá peso de _____ kilos.
5. Para o método **OFFFIRSTFIT**, a sacola mais CHEIA terá _____ itens.

Obs: 1. mais CHEIA significa com mais itens (não necessariamente a mais pesada)
2. Se houver empate de tudo, pegue a sacola de menor número



- 1 - /