## Alocação de área em disco sob WINDOWS (fat)

O algoritmo de alocação de área precisa conciliar algumas questões importantes e em certa medida contraditórias, a saber:

1. alocação encadeada, para permitir que arquivos não precisem estar em contigüidade física. Esta característica permite que um arquivo comece hoje, seja fechado e dentro de alguns dias, seja aberto para extensão, podendo crescer (ou diminuir) sem que haja necessidade de transferência física dos dados já gravados. Esta característica permite também um certo grau de (pseudo) paralelismo, através do qual dois ou mais processos podem estar gravando seus arquivos.
2. Eficiência de busca, tanto para buscar um dado quanto para alocar um bloco novo a um arquivo já existente, sem que o sistema perca muito tempo nessas operações. Isto implica em overhead mínimo possível.
3. Facilidade de acessar e alocar blocos, incluindo-se aqui a simplicidade necessária e suficiente para recuperar discos danificados e/ou arquivos excluídos por engano.

Surge agora o conceito de unidade de alocação: Trata-se do quantum de área em disco que é oferecido a qualquer processo sempre que há requisição de espaço. Este é o valor mínimo de alocação. Nunca um arquivo terá a si alocada uma área menor do que esta unidade (embora possa acabar não usando integralmente o que foi alocado). Os americanos chamam esta unidade de alocação de cluster, que foi traduzida para o português como grânulo.

Tipicamente um cluster pode ser de 2048 bytes ou de 1024 bytes. Clusters muito grandes causam fragmentação do disco (pelo desperdício de áreas não ocupadas em cada cluster) enquanto clusters pequenos implicam em grandes tabelas de alocação. Isto posto, e como regra geral, quanto menor o cluster mais eficiente é o sistema.

Como já descreveu Peter Norton, em seus livros, qualquer disco em Windows (fat) está organizado em 4 áreas (eventualmente discos rígidos tem uma quinta área que descreve o volume).

1. Registro de Boot: contém características físicas do disco e um pequeno programa de carga do núcleo do sistema operacional (SO) quando a máquina é ligada e este disco é o especificado para inicializar a máquina.
2. Área de diretório , que comporta diversas informações sobre cada um dos arquivos (nome, tamanho, data de atualização, atributos etc). Aqui interessam-nos duas informações: o nome do arquivo e o cluster em que ele se inicia.
3. Área de FAT (File alocation table). Trata-se de uma tabela que será explicada após ver-se a área 4.
4. Área de dados do disco. É a maior (mais de 95Está dividida em clusters e cada cluster é numerado.

Agora pode-se voltar à definição da FAT. Ela faz a ligação entre os diversos clusters que compõe um arquivo.

Assim por exemplo, suponhamos que o arquivo AAAA ocupa 4 clusters, o 2, o 7, o 8 e o 10. No diretório, ocorre a entrada: AAAA começa em 2. Na segunda posição da FAT (que equivale ao cluster 2) o sistema coloca o número do próximo cluster do arquivo, no caso 7. Na 7 entrada da FAT está o número do próximo cluster que é o 8. Na oitava entrada consta 10 e finalmente na décima entrada tem-se um terminador de maneira a avisar que a cadeia acaba lá.

Vale ressaltar que o primeiro cluster do arquivo está identificado no diretório. A entrada da FAT correspondente a este cluster inicial contém o endereço do próximo cluster e assim por diante.

Neste exercício vai ser reproduzida a estratégia de alocação de áreas em disco do sistema operacional windows(fat), que veio sendo usada desde o longínquo 1981 no DOS, até as versões atuais do Windows. Para reproduzir a estratégia, considere:
Imagine um disco com 30 clusters, numerados de 1 a 30,
O tamanho de cada cluster é de 500 bytes,
Ocupe sempre os primeiros clusters livres,
Os clusters defeituosos, devem ser marcados com -2 no início do exercício e assim ficar até o final.
Informe ao final, qual a disposição da FAT
Significados: 0 = cluster livre; -1=terminador; -2=cluster defeituoso

**Exemplo**O enunciado é o mesmo. Suponha que os clusters defeituosos são:

| CLUSTERS DEFEITUOSOS | 3 | 4 | 12 | 23 |
|---|---|---|---|---|

Para cada ordem a seguir, será construida a FAT como sendo um vetor de 30 números

```
   0 0 -2 -2 0 0 0 0 0 0 0 -2 0 0 0 0 0 0 0 0 0 0 -2 0 0 0 0 0 0 0
1. Cria arquivo A com  200 bytes
   -1 0 -2 -2 0 0 0 0 0 0 0 -2 0 0 0 0 0 0 0 0 0 0 -2 0 0 0 0 0 0 0
2. Cria arquivo B com 1300 bytes
   -1 5 -2 -2 6 -1 0 0 0 0 0 -2 0 0 0 0 0 0 0 0 0 0 -2 0 0 0 0 0 0 0
3. Cria arquivo C com  400 bytes
   -1 5 -2 -2 6 -1 -1 0 0 0 0 -2 0 0 0 0 0 0 0 0 0 0 -2 0 0 0 0 0 0 0
4. Cria arquivo D com  800 bytes
   -1 5 -2 -2 6 -1 -1 9 -1 0 0 -2 0 0 0 0 0 0 0 0 0 0 -2 0 0 0 0 0 0 0
5. Expande arquivo C em  600 bytes
   -1 5 -2 -2 6 -1 10 9 -1 -1 0 -2 0 0 0 0 0 0 0 0 0 0 -2 0 0 0 0 0 0 0
6. Expande arquivo A em  500 bytes
   11 5 -2 -2 6 -1 10 9 -1 -1 -1 -2 0 0 0 0 0 0 0 0 0 0 -2 0 0 0 0 0 0 0
7. Cria arquivo E com  800 bytes
   11 5 -2 -2 6 -1 10 9 -1 -1 -1 -2 14 -1 0 0 0 0 0 0 0 0 -2 0 0 0 0 0
   0 0
8. Expande arquivo B em  100 bytes
   11 5 -2 -2 6 -1 10 9 -1 -1 -1 -2 14 -1 0 0 0 0 0 0 0 0 -2 0 0 0 0 0
   0 0
9. Expande arquivo D em 1000 bytes
   11 5 -2 -2 6 -1 10 9 15 -1 -1 -2 14 -1 16 -1 0 0 0 0 0 0 -2 0 0 0 0
   0 0 0
10. Expande arquivo B em 1100 bytes
   11 5 -2 -2 6 17 10 9 15 -1 -1 -2 14 -1 16 -1 18 -1 0 0 0 0 -2 0 0 0
   0 0 0 0
11. Expande arquivo A em  600 bytes
   11 5 -2 -2 6 17 10 9 15 -1 19 -2 14 -1 16 -1 18 -1 -1 0 0 0 -2 0 0
   0 0 0 0 0
12. Cria arquivo H com 1400 bytes
   11 5 -2 -2 6 17 10 9 15 -1 19 -2 14 -1 16 -1 18 -1 -1 21 22 -1 -2 0
   0 0 0 0 0 0
13. Cria arquivo F com 1000 bytes
   11 5 -2 -2 6 17 10 9 15 -1 19 -2 14 -1 16 -1 18 -1 -1 21 22 -1 -2 25
   -1 0 0 0 0 0
14. Expande arquivo C em  1100 bytes
   11 5 -2 -2 6 17 10 9 15 26 19 -2 14 -1 16 -1 18 -1 -1 21 22 -1 -2 25
   -1 27 28 -1 0 0
15. Apaga arquivo D
   11 5 -2 -2 6 17 10 0 0 26 19 -2 14 -1 0 0 18 -1 -1 21 22 -1 -2 25 -1
   27 28 -1 0 0
16. Apaga arquivo A
   0 5 -2 -2 6 17 10 0 0 26 0-2 14 -1 0 0 18 -1 0 21 22 -1 -2 25 -1 27
   28 -1 0 0
17. Cria arquivo A com  700 bytes
   8 5 -2 -2 6 17 10 -1 0 26 0 -2 14 -1 0 0 18 -1 0 21 22 -1 -2 25 -1
   27 28 -1 0 0
18. Cria arquivo G com  200 bytes
   8 5 -2 -2 6 17 10 -1 -1 26 0 -2 14 -1 0 0 18 -1 0 21 22 -1 -2 25 -1
   27 28 -1 0 0
```

## ☞ Para você fazer

| CLUSTERS DEFEITUOSOS | 2 | 13 | 15 | 23 |
|---|---|---|---|---|

Eis as ordens a executar no disco:

```
 1-Cria arquivo A com  800 bytes
 2-Cria arquivo B com  100 bytes
 3-Cria arquivo C com  600 bytes
 4-Cria arquivo D com 1400 bytes
 5-Cria arquivo F com  800 bytes
 6-Cria arquivo G com  400 bytes
 7-Expande (se existir) ou Cria (se nao) arquivo A em/com  600 bytes
 8-Expande (se existir) ou Cria (se nao) arquivo A em/com 1200 bytes
 9-Expande (se existir) ou Cria (se nao) arquivo A em/com  600 bytes
10-Expande (se existir) ou Cria (se nao) arquivo B em/com  100 bytes
11-Expande (se existir) ou Cria (se nao) arquivo C em/com 1000 bytes
12-Cria arquivo H com 1400 bytes
13-Apaga arquivo B
14-Expande (se existir) ou Cria (se nao) arquivo A em/com  500 bytes
15-Apaga arquivo D
16-Expande (se existir) ou Cria (se nao) arquivo C em/com  900 bytes
17-Cria arquivo E com 1200 bytes
18-Expande (se existir) ou Cria (se nao) arquivo B em/com  500 bytes
```

**Para ajudar, eis um mapa do disco**

| Lista de arquivos | | |
|---|---|---|
| Arquivo | Tamanho | Cluster inicial |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

**FAT** Lembre que a FAT é um vetor de números.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

**Sua resposta** Coloque aqui os valores das posições pedidas do vetor FAT. (apenas números).

6 _____, 13 _____, 14 _____, 24 _____.

Bibliografia: Desvendando o PC, Peter Norton