UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
DEPARTAMENTO DE CIÊNCIAS DE COMPUTAÇÃO
http://www.icmc.usp.br

SCC-603 – Algoritmo e Estrutura de Dados II / 2013
Prof.ª Rosane Minghim

4º Lista de Exercícios – Processamento Cosequencial

1) Suponha que você tenha 8MB de memória RAM para ordenar o arquivo de 800.000 registros visto em aula (ver seção 7.5.1 do livro-texto).
   a) Quanto tempo durará a ordenação do arquivo usando o algoritmo *merge sort* descrito na seção 7.5.1?
   b) Quanto tempo durará a ordenação do arquivo usando o algoritmo *key sort* descrito no capítulo 5?
   c) Por que o algoritmo *key sort* não funcionará se houver 1 MB de RAM disponível para a fase de ordenação?

2) Quantos posicionamentos (seeks) são necessários para uma intercalação em um único passo como descrito na seção 7.5, se um posicionamento toma 50ms, em média, e o tamanho do *buffer* interno disponível é 500K? e se for 100K?

3) O desempenho de um processo de ordenação é, geralmente, medido em função do número de comparações necessárias. Porque? Explique por que o número de comparações não é uma medida adequada para avaliar o desempenho de métodos de ordenação de grandes arquivos que não cabem em memória.

4) Simule e Projete um programa para fazer o seguinte:
   a) Examinar o conteúdo de dois arquivos ordenados, M1 e M2.
   b) Produzir um terceiro arquivo COMUM contendo a cópia dos registros que existam nos dois arquivos. (AND)
   c) Produzir um quarto arquivo DIFERENTE contendo os registros dos dois arquivos que existam em apenas M1 ou M2. (XOR)

5) Em nossas computações envolvendo *merge*, assumimos que só uma busca e um atraso rotacional são necessários para um único acesso sequencial. Se isso não ocorrer, mais tempo será necessário para a realização de I/O. Por exemplo, o arquivo de 80 MB usado no exemplo 7.5.1 do livro texto, para a fase de leitura do processo de geração de corridas, a leitura de cada corrida pode requerer muitos acessos. Assuma que o tamanho do *extent* para nosso drive hipotético é de 20.000 bytes (aproximadamente uma trilha), e que todos arquivos armazenados em blocos do tamanho da trilha devem ser acessados separadamente (uma procura e um atraso rotacional por bloco).
   a) Quantas buscas o passo 1 requer agora?
   b) Quanto tempo os passos 1, 2 e 3 levam?
   c) Qual o impacto de um aumento no tamanho do arquivo por um fator 10 terá no tempo total do *merge sort*?

6) Implemente o procedimento para *match* cosequencial descrito na seção 7.1 do livro texto.

7) Implemente o procedimento para *merge* cosequencial descrito na seção 7.1 do livro texto.

8) Simule a ordenação externa para os dados abaixo, usando:
    a. Intercalação em k-vias com corrida de tamanho pré-definido
    b. Intercalação em k-vias com *replacement selection*

Obs. Buffer de entrada – 6 slots, buffer da saída – 3 slots.

Dados: T – Z - A – S – D – L – R – J – H – F – D – G – I – Y – U – T – X – E – P – O – M – N – V – B - C

9) Para o exercício anterior responda: o valor de k, o tamanho de cada corrida, os valores das corridas, o número de *seeks* da etapa 1 (geração de corridas) e o número de *seeks* da etapa 2 (merge).

10) Suponha que um arquivo de dados com 6.000 registros, mantido em disco, deve ser ordenado em um computador cuja memória interna acomoda no máximo 600 registros por vez, usando o procedimento de intercalação visto em aula. Considere que serão geradas 10 corridas de 600 registros cada, e que será realizada uma intercalação em 10 vias. Essa mesma área de memória interna é usada como *buffer* de entrada para leitura de dados do disco, e o sistema conta com um buffer de saída adicional que acomoda 200 registros.
    a) Durante a intercalação, quantos registros serão lidos de cada corrida cada vez que ela é acessada? **Justifique**.
    b) Quantos *seeks* serão realizados **para ler dados** durante o processo de intercalação (excluindo a fase de geração de corridas)? Quantos serão realizados para escrever dados? **Justifique**.

Divirta-se!