Projeto de Algoritmo com Implementação nº 1

MC458 — Projeto e Análise de Algoritmos I

Prof. Pedro J. de Rezende

2º Semestre de 2020

1 Problema

Seja V um vetor de n números reais e k um inteiro com $1 \leq k \leq n$. Neste projeto, serão analisados três algoritmos para o problema de obtenção **dos k menores elementos de V em ordem crescente.**

Exemplo Suponha $n = 6$, $V = [3.5, 2.2, -4.1, 1.3, 8.7, -2.0]$ e $k = 3$. A resposta esperada é: $[-4.1, -2.0, 1.3]$.

2 Abordagem

Considere as seguintes alternativas de abordagem para esse problema:

- **Método 1:** selecionar o i-ésimo menor elemento de V para $i = 1, 2, ..., k$ utilizando um algoritmo que percorre k vezes o vetor V.
- **Método 2:** ordenar o vetor inteiro utilizando QuickSort e tomar os k primeiros elementos na ordem obtida.
- **Método 3:** utilizar um Heap de mínimo para encontrar os k menores elementos em ordem crescente.

O método mais adequado depende da relação entre k e n.

3 Objetivo

O objetivo deste projeto é analisar o desempenho de algoritmos baseados nesses métodos para diferentes valores de k, sobre um dado vetor V de um milhão de elementos.

4 Implementação

Seguindo os métodos apresentados, você deverá utilizar três algoritmos para obter os k menores elementos em ordem crescente de um vetor de tamanho n.

É fornecida uma implementação do algoritmo QuickSort que deverá ser usada para a implementação do Método 2.

Com exceção do algoritmo QuickSort fornecido, todos os demais algoritmos deverão ser efetivamente implementados por você, em C ou C++. Não utilize funções contidas nas bibliotecas de C ou C++ que implementam os algoritmos ou estruturas de dados requeridos.

Além disso, cada método é mais eficiente numa determinada faixa de valores de k, considerando um vetor de um dado tamanho. Em particular, note que existem valores k_1 e k_2 tais que:

- se $k < k_1$, o Método X é mais rápido que os demais para o vetor V dado;
- se $k_1 \leq k \leq k_2$, o Método Y é mais rápido que os demais para o vetor V dado;
- se $k > k_2$, o Método Z é mais rápido que os demais para o vetor V dado;

onde X, Y e Z identificam, em alguma ordem, os três métodos descritos.

Faça uma implementação de uma função *eficiente* que determine os valores de k_1 e k_2. No que se segue, vamos identificar essa função chamando-a de "Método 0". Não se preocupe com variações nos valores encontrados em diferentes execuções dessa função. Por se tratar de uma análise empírica, os valores obtidos estarão sujeitos a desvios de até 10%, aproximadamente.

5 Entrada

O vetor de entrada deve ser lido de um arquivo de texto. O nome do arquivo de entrada, o método a ser executado e o valor de k devem ser recebidos, nessa ordem, através da linha de comando. O método será indicado por um único caractere, que pode ser '0', '1', '2', ou '3', representando as alternativas 0 da Seção 4, ou 1, 2 ou 3, da Seção 2, respectivamente.

Exemplo: supondo que seu programa se chame 'k_{min}', uma linha de comando esperada seria:

```
./kmin vet-1000000.ins 0
```

Nesse exemplo, seu programa deveria ler a instância do arquivo '$vet-1000000.ins$' e utilizar o Método 0 para encontrar k_1 e k_2. Outro exemplo seria:

```
./kmin vet-1000000.ins 2 50000
```

Neste caso, seu programa deveria ler a instância do arquivo '$vet - 1000000.ins$' e resolvê-la utilizando o Método 2 com $k = 50000$.

Os arquivos de entrada têm o seguinte formato: na primeira linha há um número inteiro, representando o número de elementos do vetor (isto é, o parâmetro n do problema). Em seguida, são fornecidos n doubles, um em cada linha, constituindo o vetor de entrada.

6 Saída

Seu programa deve imprimir apenas o tempo de execução em segundos (double com 6 casas decimais de precisão) seguido de um caractere '\n'.

7 Arquivos providos

Para simplificar a implementação, fornecemos quatro funções:

- a primeira ordena um vetor com o algoritmo QuickSort;
- a segunda deve ser usada para medir os tempos de execução do seu programa;
- a terceira verifica se seu algoritmo identificou corretamente os k menores elementos de V em ordem crescente. Supondo que seu algoritmo forneça os k menores elementos em um vetor-resposta R, essa função recebe o vetor V, os inteiros n e k, e o vetor R. O retorno dessa função indica se a resposta de seu algoritmo está correta. Caso algum erro seja detectado, a resposta fornecida pelo seu algoritmo e a resposta esperada serão impressas automaticamente pela função.
- a quarta recebe um vetor R com os k menores elementos, cria um arquivo chamado "kmin.out" e escreve o conteúdo de R dentro dele. O uso dessa função no final do seu código é **obrigatório**, pois será usada para realizar a correção do seu código.

Baixe o código em C ou o código em C++ de acordo com sua escolha de linguagem. A descrição do uso dessas funções se encontra junto com o código. São fornecidos dois arquivos. O arquivo '`kmin.c`' (ou '`kmin.cpp`') contém a descrição das funções fornecidas e um pequeno modelo para seu código. O arquivo '`rotinas_c.o`' (ou '`rotinas_cpp.o`') contém as rotinas fornecidas já compiladas. Em caso de dúvidas relativas à compilação com este arquivo, veja os comandos na Seção Experimentos. Para extrair todos os arquivos do pacote use um dos comandos:

```
tar -xzf codigo_c.tar.gz
tar -xzf codigo_cpp.tar.gz
```

8 Experimentos

Após a finalização do seu código, a implementação deve ser avaliada através de uma série de experimentos. **Antes de iniciá-los, lembre-se de compilar o código com a opção `-O3` para ativar as rotinas de otimização do compilador!** Utilize um dos seguintes comandos para compilação:

```
gcc -O3 kmin.c rotinas_c.o -o kmin
g++ -O3 kmin.cpp rotinas_cpp.o -o kmin
```

9 Testes

Como a avaliação será sobre o tempo obtido pelas implementações, não haverá testes no Susy para avaliar o seu código. Porém, ele será submetido a uma avaliação posterior, visando comparar os resultados com os valores descritos no relatório (poderá haver uma variação de até 10% nos valores).

Visando facilitar os testes do seu código, será disponibilizado um script de testes. Para utilizá-lo, baixe o arquivo `experimentos.tar.gz` da pasta de arquivos auxiliares do Susy. Extraia todos os arquivos do pacote usando o comando:

```
tar -xzf experimentos.tar.gz
```

Inicialmente, execute o script '`experimentos.py`' no diretório em que se encontra o executável do seu programa. O script mostrará, para alguns diferentes valores de k, o tempo de execução de cada algoritmo sobre aquele vetor. A execução do script é esperada levar no máximo 60 segundos. Esse script recebe como único parâmetro o nome do executável do seu programa. Supondo que o executável do seu programa se chama '`kmin`', use o seguinte comando para executá-lo:

```
python experimentos.py kmin
```

Após a execução do script '`experimentos.py`', será gerado um arquivo de texto chamado '`resultados.dat`'.

10 Análise dos algoritmos

1. Crie um arquivo `pdf` para o seu relatório, chamado '`PA#NNNNNN.pdf`', onde # é o número do presente PA e `NNNNNN` é o seu R.A., e descreva brevemente a complexidade de tempo em pior caso de cada um dos três algoritmos implementados, em função

de k e n (utilize no máximo 5 linhas por algoritmo). Não é necessário demonstrar a complexidade de tempo de nenhum algoritmo ou de procedimentos auxiliares que tiverem sido estudados nas aulas teóricas. Apenas indique qual a complexidade das principais operações utilizadas e qual a complexidade de tempo total de cada algoritmo. Se for conveniente expressar a complexidade com uma relação de recorrência, indique-a junto com sua fórmula fechada, sem necessidade de explicitar como resolvê-la.

2. Em seguida, utilizando as complexidades de tempo descritas (em 1.) para os três algoritmos implementados e os seus conhecimentos sobre algoritmos de ordenação, escreva um parágrafo (no máximo 10 linhas) fornecendo uma breve explicação para os resultados experimentais observados. Em particular, você deve ter notado, durante os experimentos, que para cada método considerado, existem pelo menos alguns valores de k para os quais este método é capaz de resolver o problema mais rapidamente que os demais. Aponte qual característica de cada algoritmo é responsável por esse fenômeno.

3. Finalmente, inclua uma única linha no final do relatório, dizendo quais foram os valores de k_1 e k_2 encontrados. Esses valores devem ser a média aritmética de **três** execuções distintas de sua função que denominamos de "Método 0".

11 Critérios de avaliação

A avaliação consistirá na atribuição de conceitos para o Projeto:

- **Inadequado**: se o Código não compila ou não implementa nenhum dos métodos corretamente.
- **Satisfatório**: se o Código compila, implementa algum método corretamente, mas não todos.
- **Completo**: se o Código compila e implementa todos os métodos corretamente.

A conversão desses conceitos para notas seguirá o que está descrito na página da disciplina levando-se ainda em conta a qualidade do conteúdo e completude do Relatório submetido.

12 Entrega

Observe os seguintes quesitos para realizar a entrega de seu trabalho:

- A submissão do arquivo do Relatório deve ser feita via Google Classroom na Tarefa correspondente a esta Atividade.

- A sua submissão do arquivo de código será feita **exclusivamente** através do sistema SuSy. Para acessá-lo, basta utilizar seu R.A. e sua senha da DAC (Ex: `usuário: 123456` e `senha: senha-da-DAC`).
- O arquivo com o seu código deve se chamar '`kmin.c`' ou '`kmin.cpp`', de acordo com a linguagem usada. O SuSy não irá alertá-lo sobre erros nos nomes, portanto, tenha atenção na hora de fazer suas submissões;
- No SuSy, serão aceitas **até 5 submissões** por aluno, sendo preservada apenas a última submissão.
- O prazo para submissão das resoluções se encerrará às 23hs do dia indicado no Google Classroom. Envios realizados (do programa no Susy ou do Relatório do Google Classroom) após esse horário serão considerados **atrasados**. Se o atraso for de **até duas horas** após o encerramento do prazo regular de submissão, as resoluções submetidas ainda serão corrigidas e receberão nota integral. Resoluções enviadas com **mais de 2hs** de atraso **não serão corrigidas** e receberão nota **zero**.

Link para submissão no SuSy: https://susy.ic.unicamp.br:9999/mc458a/PA1.

AVISO: as ferramentas de detecção de plágio do SuSy serão utilizadas durante a avaliação das submissões!