UNIVERSIDADE FEDERAL DE MINAS GERAIS
Departamento de Ciência da Computação

Disciplina	Curso	Turmas	Período
Algoritmos e Estruturas de Dados II			2°

Professores
Erickson Nascimento, William Schwartz

Trabalho Prático 2 - Ordenação/Semelhança entre Imagens

Data de entrega: 14/05/2013
Valor: 10 **pontos**

Introdução

O objetivo deste trabalho é buscar por correspondências entre imagens diferentes que representam a mesma cena. Uma forma de buscar por essas correspondências é criar um identificador para cada imagem e então comparar esses identificadores. Aqueles mais similares indicam alta probabilidade de terem sido computados para a mesma cena. Imagens de uma mesma cena podem ter identificadores ligeiramente diferentes, fato que é explicado pelas mudanças na iluminação da cena ou mesmo por possíveis ruídos no sistema de aquisição das imagens (câmera, filmadores, etc.). Uma forma de criar um identificador para as imagens é calcular um vetor chamado de Descritor. Esse vetor possui um conjunto de valores que, juntos, identificam uma imagem. Um dos algoritmos mais utilizados para a criação desses descritores é o algoritmo SIFT (*Scale Invariant Feature Tranform*). O SIFT contrói descritores de pontos importantes das imagens, ou seja, dado um ponto de destaque de uma imagem, o SIFT cria uma representação numérica das informações relevantes deste ponto. Depois de obter os descritores do SIFT de diversas imagens, basta calcular a distância Euclidiana entre os descritores. Quanto menor a distância obtida, maior a semelhança existente entre um par de imagens.

O que deve ser feito

Dado um conjunto de N imagens, onde cada imagem é representada por um descritor SIFT (vetor de 128 *floats*) que será disponibilizado, deve-se:

1. Fornecer K candidatos mais similares de cada imagem ($0 < K < N$). Uma imagem não pode ser similiar a ela mesma. Para resolver este problema, deve-se implementar um algoritmo de ordenação com complexidade $O(nk)$, baseado no algoritmo *QuickSort*.

2. Fornecer P pares de imagens mais similares ($1 < P <= N$). Uma imagem não pode ser o par dela mesma. Para resolver este problema, devem ser utilizados dois algoritmos de ordenação, um com complexidade $O(n^2)$ e outro com complexidade $O(n \cdot log(n))$.

3. A linha de comando deve ser a seguinte:

```
./exec entrada.txt saida.txt
```

Na resolução de ambos os problemas deve ser criado um Tipo Abstrato de Dados contendo o nome da imagem e o vetor de 128 valores *float* que representa cada descritor.

Entrada

A entrada consiste em um arquivo de texto passado por linha de comando. A primeira linha contém os valores de N, K, P e um valor inteiro para indicar qual algoritmo de ordenação será utilizado para encontrar os pares de imagens mais próximas: 1 para algoritmo de ordenação com complexidade $O(n^2)$ ou 2 para o algoritmo de ordenação com complexidade $O(n \cdot log(n))$. Os quatro valores são separados por espaço.

Cada uma das N linhas seguintes contém o nome de uma imagem e os 128 valores do tipo *float* que represetam um descritor desta imagem, também separados por espaços. Considere o nome da imagem com no máximo 40 caracteres — formado por caracteres maiúsculos, minúsculos e numéricos, [a-z,A-Z,0-9] — e o tipo *float* com 4 casas decimais.

A seguir é apresentado um exemplo da entrada. Para este exemplo, foi considerado que o descritor é representado por um vetor de *float* de tamanho 5, apenas para demonstração, mas no trabalho deve-se considerar sempre vetores de tamanho 128. Observe que a base de dados contém 5 imagens; deseja-se encontrar as 3 imagens mais similares de cada imagem; deseja-se encontrar os 5 pares de imagens mais similares; e o algoritmo de ordenação utilizado para encontrar os pares mais próximos é de complexidade quadrática.

```
5 3 5 1
img1 0.0420 0.0010 0.0450 0.1250 0.0000
img2 0.0050 0.0000 0.9800 0.1900 0.0050
img3 0.0600 0.0200 0.5900 0.2400 0.0030
img4 0.0300 0.0000 0.0500 0.1600 0.0370
img5 0.1280 0.0800 0.2000 0.0000 0.0200
```

Saída

A saída consiste em um arquivo de texto, cujo nome é passado por linha de comando. Cada uma das primeiras N linhas contém o nome de uma imagem (escolhido por ordem alfabética) e o nomes das K imagens mais silimares a ela acompanhados pelas distâncias encontradas (este nomes devem ser ordenados pela menor distância). As próximas P linhas contém os nomes dos P pares de imagens mais silimares, separados por vírgula (,) e acompanhados pela distância entre cada par. Esta impressão deve ser por ordem crescente de distância, ou seja, a primeira linha apresenta o par de imagens com a menor distância dentre todas as imagens. Observe que a distância entre um par de imagens (i, j) é igual à distância entre o par (j, i), para qualquer imagem, então este par deve ser impresso apenas uma vez, respeitando a ordem alfabética dos nomes das duas imagens. Caso a distância entre dois pares seja igual, imprima por ordem alfabética do nome das duas imagens.

A formatação da saída deve estar de acordo com o exemplo a seguir.

```
img1 - img4(0.0526) img5(0.2317) img3(0.5576)
img2 - img3(0.3975) img5(0.8162) img4(0.9314)
img3 - img2(0.3975) img5(0.4671) img4(0.5481)
img4 - img1(0.0526) img5(0.2538) img3(0.5481)
img5 - img1(0.2317) img4(0.2538) img3(0.4671)

img1,img4 - 0.0526
img1,img5 - 0.2317
img4,img5 - 0.2538
img2,img3 - 0.3975
img3,img5 - 0.4671
```

A primeira linha do exemplo mostra que para imagem $img1$, as imagens mais semelhantes são as imagens $img4, img5$ e $img3$. Observe que não há espaço entre o nome da imagem é os parênteses com a distância; o nome da imagem e das K mais similares é sepados por espaços e hífen (-).

Ao final de N linhas, imprima uma linha em branco.

A primeira linha depois da linha em branco mostra que o par de imagens mais silimares foram as imagens $img1, img4$ e a distância entre ela foi 0.0526. Observe que o nome das duas imagens é separado por vírgula (,) e a separação entre os nomes e a distância é feita por espaços e hífen (-).

Ao final do arquivo, insera uma linha em branco.

O que deve ser entregue

- Código fonte do programa em C (todos os arquivos .c e .h), devidamente comentado e documentado.
- Documentação sobre o trabalho (máximo de 10 páginas em formato pdf), com as seguintes seções:

    1. Introdução: descrição sucinta do problema a ser resolvido e visão geral sobre o funcionamento do programa. Cópias idênticas da especificação do trabalho nesta seção terão nota ZERO.
    2. Implementação: descrição sobre a implementação do programa. Deve ser explicada a estrutura de dados utilizada, o funcionamento de todas as funções e procedimentos utilizados, o formato de entrada e saída de dados, bem como todas as decisões tomadas na implementação do trabalho.
    3. Estudo de Complexidade: estudo da complexidade DETALHADO do tempo de execução dos procedimentos implementados e do programa como um todo (notação O).
    4. Testes: devem ser realizados diversos testes variando os valores de K e apresentados os gráficos de tempos de execução correspondentes. Além disso, devem ser apresentados gráficos comparando o tempo de execução obtido utilizando o algoritmo $O(n^2)$ e $O(n \cdot log(n))$, com P variando entre 30% de N e 100% de N. Faça também uma análise dos resultados dos testes comparando com o estudo de complexidade realizado na Seção anterior. É obrigatório apresentar gráficos de tempo de execução para os variados testes, caso contrário a nota para esta seção da documentação será ZERO.
    5. Conclusão: comentários gerais sobre o trabalho e as principais dificuldades encontradas em sua implementação.
    6. Bibliografia: bibliografia utilizada para o desenvolvimento do trabalho, incluindo sites da Internet se for o caso.

- O trabalho será entregue através do sistema de submissão de trabalhos práticos: `http://aeds.dcc.ufmg.br`, em um arquivo compactado (.zip) contendo os arquivos fonte. A documentação deve ser enviada em pdf, no link específico para sua submissão no sistema. Caso tenha algum problema na submissão pelo Prático, envie o trabalho por email para todos os monitores.

Comentários gerais

- Todos os TADs, vetores e matrizes deverão ser sempre declarados como ponteiros e instanciados dinamicamente.
- Técnica, clareza, legibilidade, organização, coesão, indentação, comentários, nomes adequados a variáveis, atributos e funções também vão valer pontos.
- Trabalhos copiados serão penalizados com a nota zero.
- Os trabalhos não serão aceitos, em hipótese nenhuma, fora do prazo.