Universidade Federal de Minas
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Algoritmos e Estruturas de Dados II
2º Semestre de 2014

Trabalho Prático 2
Sistema de Recomendação

Valor: 20 pontos
Data de entrega: 21/11/2014 (Inadiável)

Neste terceiro trabalho iremos explorar mais uma vez o problema de recomendação de filmes, visto no primeiro trabalho, porém utilizando estruturas de dados mais complexas. No trabalho anterior foi desenvolvido um sistema de recomendação de filmes utilizando duas técnicas, a de filmes mais vistos e de filtragem colaborativa. Na primeira é oferecido aos usuários um conjunto de filmes com base na opinião de todos os demais usuários sem considerar as preferências específicas do indivíduo. Já a segunda realiza uma filtragem dos usuários, onde as recomendações levam em consideração apenas os perfis mais similares.

Em relação ao trabalho anterior, modificaremos a estrutura de armazenamento das informações sobre os filmes e a visualização dos usuários. A abordagem anterior utilizava um limite de tamanho fixo para os vetores, gerando assim dois problemas. O primeiro é a limitação do número de filmes e usuários que o sistema pode suportar, que não pode ser ultrapassado. Já o segundo ocorre quando é limite esperado é atingido com pouca frequência, e o programa acaba ocupando mais espaço do que necessita em quase todas as vezes em que é executado. Para resolver esse problema, neste trabalho serão utilizadas duas *listas*, uma para os filmes e outra para a visualização dos usuários.

A recomendação dos itens mais populares deverá agora excluir os filmes que cada usuário já assistiu, sendo portanto um conjunto diferente para cada usuário. Caso o usuário tenha visto todos os filmes, essa linha deve ser deixada em branco. A ordem de exibição dos filmes é a mesma da definida no trabalho anterior, juntamente com os critérios de desempate:

$$FilmeMaisVisualizado > FilmeMaisRecente > FilmeDeMaiorMovieId \tag{1}$$

Como a recomendação dos filmes mais assistidos será feita de forma individual para cada usuário, a contagem das visualizações terá de ser armazenada de forma adequada, visando um baixo custo no tempo de acesso. Para isso você deverá utilizar uma *tabela hash*, que possui chave igual à popularidade do filme. Como antes, sempre haverá o risco de ocorrer empate no número de visualizações, nesse caso isso irá gerar uma colisão na *hash*, que deverá ser tratada com uma *árvore binária* (Figura 1). Para que esse tratamento de colisões seja eficiente, a escolha da chave da *árvore* deve fazer com que o caminhamento central retorne os filmes na ordem da impressão para um mesmo número de visualizações. Dessa forma para retornar a recomendação por filmes mais vistos, basta realizar uma ordenação decrescente da *hash* e imprimi-lo de forma linear, juntamente com as respectivas *árvores*.

No caso da recomendação personalizada, o cálculo da similaridade continua o mesmo, utilizando Jaccard:

$$J(u_1, u_2) = \frac{itens(u_1) \cap itens(u_2)}{itens(u_1) \cup itens(u_2)} \tag{2}$$

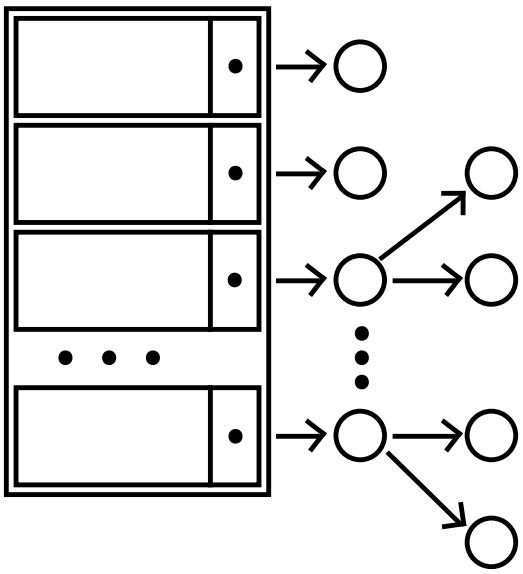

Figura 1: Hash que utiliza tratamento de colisão com arvore

Assim como foi definido para os filmes mais populares, a recomendação personalizada também utilizará o tipo abstrato de dados *hash*, com tratamento de colisões através de *árvore binária*. Nesse caso a chave da *hash* será o coeficiente de Jaccard [2], e a chave da *árvore* deverá atender o critério de desempate:

$$UsuárioDeMaiorId > FilmeMaisRecente > FilmeDeMaiorMovieId \tag{3}$$

Note que para este trabalho, como forma de simplificação, em caso de empate no coeficiente de Jaccard [2] deve-se levar em consideração primeiramente o usuário que possuir o **maior** identificador, ao contrário do primeiro trabalho.

Base de dados

Duas bases de dados com informações sobre os filmes a serem recomendados e seus usuários serão disponibilizadas. A primeira, relacionada aos filmes, traz a lista de informações sobre cada um dos n filmes que o usuário pode ter assistido. Para cada filme, as seguintes informações são disponibilizadas, sempre separadas por um caractere de tabulação:

```
movie_id        titulo          imdb_id         ano
1               Toy story       0114709         1995
```

A segunda base, relacionada aos usuários, traz o identificador do usuário seguido da lista de filmes que ele assistiu. Cada filme agora é representado por seu movie_id, que significa que o usuário já assistiu o filme.

Exemplo de arquivo de visualizações

```
user_id movie_ids
12      2       1       3       0
```

Arquivos de entrada

Seu programa deve receber como entrada um arquivo x.tst.i (teste número x). A primeira linha do arquivo contém o nome da base de dados de filmes seguida do nome da base de dados de usuários, o número de recomendações tanto para os mais populares quanto personalizada

(4 no exemplo), e o tamanho da *hash* (10 no exemplo), separadas por um caractere de tabulação. As linhas seguintes devem conter o identificador do usuário para o qual se deseja fazer a recomendação.

Exemplo do arquivo x.tst.i

```
metadata.txt    ratings.txt     4       10

321
543
12
...
```

O arquivo de saída, x.tst.o (saída do teste x), deve conter a primeira linha idêntica à do arquivo de entrada. Para as linhas seguintes, deve-se colocar o identificador do usuário seguido de ":" e do título dos filmes recomendados (nesse exemplo 4), separados por um caractere de tabulação. Primeiro você deve listar o nome dos filmes recomendados através da abordagem de filmes mais populares, e depois da filtragem colaborativa.

Exemplo do arquivo x.tst.o

```
metadata.txt    ratings.txt     4       10

15926:
Most popular
Blood Diamond   O Brother, Where Art Thou?      Miss Congeniality       Terminator Salvation

Personalizada
Blood Diamond   Inside Man      V for Vendetta  Terminator Salvation

52379:
Most popular
O Brother, Where Art Thou?      Miss Congeniality       Terminator Salvation    The Dark Knight

Personalizada
Iron Man        The Bourne Ultimatum    Evan Almighty
...
```

A saída será comparada com o gabarito no Prático.

Desafio

Como desafio, propomos a criação de uma interface gráfica em PHP que utilize as informações do IMDB (imdb_id) e uma adaptação de seu programa como backend. Ela deverá funcionar de forma semelhante ao Max do Netflix. Esse sistema apresenta um conjunto de filmes iniciais nos quais o usuário pode atribuir uma nota, a partir dessas notas uma sugestão de filme oferecida. Caso o usuário já tenha visto o filme recomendado, o sistema apresenta uma segunda sugestão.

Para adequar o problema a esse trabalho, vamos simplificá-lo de forma a utilizar apenas o fato do usuário ter assistido ou não um conjunto inicial de filmes. Para isso exiba ao usuário o conjunto de vinte filmes mais populares calculado pelo seu programa e peça que o usuário marque pelo menos um filme (evitando assim a divisão por zero no Jaccard [2]). Em seguida, utilize as visualizações desse usuário como entrada para seu programa para obter a lista de recomendações personalizada. Utilize as bases de 2109 usuários e 500 itens, para realizar a recomendação.

Para construir a interface utilize as informações presentes no IMDB sobre os filmes e exiba seu conteúdo aos usuários. Outro ponto interessante que pode ser incorporado à interface é a exibição dos posters dos filmes que fazem parte da interceção com o usuário de maior Jaccard, para justificar essa sugestão ("Porque você assistiu aos filmes:").

Comentários Gerais

1. Comece a fazer este trabalho logo, enquanto o problema está fresco na memória e o prazo para terminá-lo está tão longe quanto jamais poderá estar.
2. Clareza, indentação e comentários no programa também serão avaliados.
3. O trabalho é individual.
4. A submissão será feita pelo Prático (aeds.dcc.ufmg.br)
5. O Prático desconsidera espaços, quebras de linha e tabulações a mais de sua saída, portanto não é necessário alinhar de forma exata estes itens à saída padrão fornecida.
6. Trabalhos copiados, comprados, doados, etc., serão penalizados conforme anunciado.
7. Penalização por atraso: $(2^d - 1)$ pontos, onde d é o número de dias de atraso.