FACOM- UFU
Professor: Wendel Melo

Quinto trabalho de Organização e Recuperação da Informação 2020-02

Descrição

(Exercício adaptado de CS50 IA - 2020 de HarvardX). Este trabalho consiste no cálculo da métrica de PageRank para uma base de dados genérica.

Deve ser entregue apenas um **único** programa desenvolvido em Python 3 que realize a tarefa solicitada. O programa deve usar apenas as bibliotecas padrão Python 3, isto é, as bibliotecas que já vem com a instalação padrão do interpretador da linguagem, com exceção da biblioteca *BeautifulSoup*, que pode vir a ser utilizada (não é obrigatório, todavia).

O trabalho deve ser feito **individualmente** e o código gerado deve ser anexado na respectiva tarefa do *MS Teams* até o dia 04/11/2021.

Aviso importante: se for detectado cópia ou qualquer tipo de trapaça entre trabalhos, todos os envolvidos serão punidos com a nota zero. Portanto, pense bem antes de pedir para copiar o trabalho do seu coleguinha, pois ele poderá ser punido também!

Antes de começar a desenvolver, certifique-se de que você compreendeu os slides sobre *PageRank*. Após estudar os slides, volte aqui e leia a descrição novamente.

A entrada do programa

Seu programa deverá receber um diretório (cujo caminho é passado pela linha de comando). Você deve assumir que todo arquivo dentro do diretório passado pela linha de comando faz parte da base de dados. Você pode assumir também que não há subdiretórios dentro do diretório da base de dados. Por exemplo, suponha que desejamos calcular a métrica *PageRank* para a base de dados inserida no diretório *base0* (presente no mesmo diretório do nosso programa). Assumindo que seu programa se chame pagerank.py, seu programa poderia ser invocado (no Unix) com a linha

> python3 pagerank.py base0

Destaca-se mais uma vez que, dessa vez, seu programa deve considerar que todos os arquivo dentro do diretório recebido como argumento de entrada pertencem a base de dados (dessa vez, não há um arquivo listando toda a base como em trabalhos anteriores).

A base de documentos

A base de documentos é composta por documentos HTML que podem ter ligações (links) para outros documentos da base, documentos externos e ligações para o próprio documento. Neste trabalho, estamos interessados apenas nas ligações entre documentos da própria base (demais ligações devem ser ignoradas). Assuma que as ligações sempre são feitas através de tags HTML *<a href="link">*. Para

este trabalho, será preciso extrair o grafo de ligações dos documentos da base para o cálculo do *PageRank*.

A saída do programa

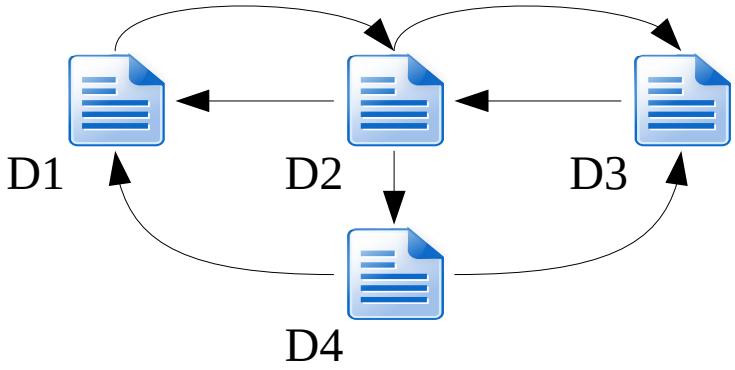
Seu programa deverá gerar **quatro** arquivos de saida:

1. *links_origem.txt*
2. *links_destino.txt*
3. *pg_amostragem.txt*
4. *pg_iterativo.txt*

É importante que os arquivos de saída gerados tenham exatamente os respectivos nomes apontados aqui. **Não invente nomes da sua cabeça, sob pena de perda de pontos!**

O arquivo de saída *links_origem.txt*

O arquivo *links_origem.txt* deve apresentar informações sobre o grafo de ligações da base, em um formato de certo modo similar a representação de grafos por lista de adjacências. Por exemplo, suponha que a base seja composta pelos documentos D1, D2, D3 e D4, com as ligações conforme representado no diagrama abaixo:



Nesse caso, teremos o seguinte conteúdo para o arquivo *links_origem.txt*:

```
D1: D2
D2: D1 D3 D4
D3: D2
D4: D1 D3
```

exemplo de arquivo de saída links_origem.txt

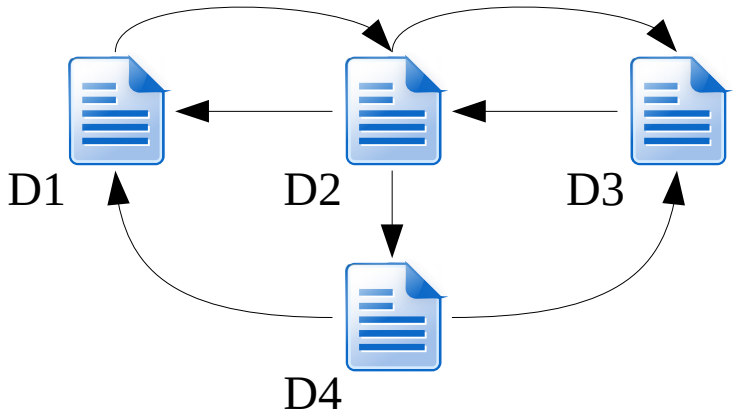
Repare que, cada linha do arquivo está no formato:

<doc>: <lista de documentos para os quais *doc* aponta >

Note que o número de linhas no arquivo *links_origem.txt* é o número de documentos no diretório na base de dados. Observando, por exemplo, a última linha do arquivo *links_origem.txt*, percebemos que D4 possui links para os arquivos D1 e D3.

O arquivo de saída links_destino.txt

O arquivo *links_destino.txt* tem formato similar ao arquivo *links_origem.txt*, com a diferença que, agora devem ser representados os documentos que apontam para cada um dos documentos da base. Assim, ao representar a linha do documento X, deve-se listar agora os documentos que apontam para X, e não mais os documentos para os quais X aponta. Por exemplo, suponha que a base seja composta pelos documentos D1, D2, D3 e D4, com as ligações conforme representado no diagrama abaixo (o mesmo diagrama usado na descrição do arquivo *links_origem.txt*):



Nesse caso, teremos o seguinte conteúdo para o arquivo *links_destino.txt*:

```
D1: D2 D4
D2: D1 D3
D3: D2 D4
D4: D2
```

exemplo de arquivo de saída links_destino.txt

Note que, agora cada linha do arquivo está no formato

<doc>: <lista de documentos que apontam para doc >

Observando a primeira linha do arquivo exemplo *links_destino.txt*, percebemos que o documento D1 é apontado pelos D2 e D4.

É válido ressaltar que, nos arquivos *links_origem.txt* e *links_destino.txt*, devem ser representados os nomes reais dos arquivos (não é preciso remover a extensão). Também é importante frisar que links duplicados entre os arquivos serão ignorados. Se, por exemplo, o arquivo A conter dois links para o documento B, será contabilizado como se houvesse apenas uma única ligação de A para B.

O arquivo de saída pg_amostragem.txt

O arquivo *pg_amostragem.txt* deverá trazer o cálculo do *PageRank* de cada arquivo pelo método da amostragem (consulte o material da aula para mais detalhes). Considere que você deve realizar 10000 transições aleatórias de documento corrente. (é recomendado, todavia, escrever o código de modo a ser fácil modificar o número de transições aleatórias). Considere também que o fator de *dumping d* é 0,85, isto é, há 85% de probabilidade de um usuário comum seguir um link do documento corrente e 15% de probabilidade do mesmo usuário pular para um documento aleatório da base. Em cada linha do arquivo *pg_amostragem.txt* devemos ter o nome de um arquivo da base, e seu respectivo valor de *PageRank* (entre 0 e 1).

Por exemplo, para o base de testes *base0*, poderíamos ter como arquivo *pg_amostragem.txt*:

```
1.html 0.2272
2.html 0.4317
3.html 0.214
4.html 0.1271
```

exemplo de arquivo de saída pg_amostragem.txt

Obs: para implementar a probabilidade de 85% de chance do usuário seguir um link, pode-se gerar um número aleatório entre 0 e 1. Se o número for menor que 0,85, seu programa deve implementar o seguimento de um link aleatório do documento corrente (considere que cada documento linkado pelo arquivo corrente tem igual probabilidade de ser seguido). Caso contrário, seu programa deve implementar o salto aleatório para um documento qualquer da base (considere que cada documento da base tem igual probabilidade de ser o próximo a ser acessado, incluindo o próprio documento corrente!). Note que, cada execução pode gerar resultados ligeiramente diferentes devido à natureza aleatória do algoritmo.

O arquivo *pg_iterativo.txt*

O arquivo *pg_iterativo.txt* deverá trazer o cálculo do *PageRank* de cada arquivo pelo método iterativo (consulte o material da aula para mais detalhes). Para o critério de parada do algoritmo, considere que os valores de *PageRank* convergiram quando a diferença entre os *PageRank's* de duas iterações consecutivas não é maior que 10^{-6}.

Por exemplo, para o base de testes base0, poderíamos ter como arquivo *pg_iterativo.txt*:

```
1.html 0.21991357059363245
2.html 0.4292092493937253
3.html 0.21991357059363245
4.html 0.13096360941900992
```

exemplo de arquivo de saída pg_iterativo.txt

Corretor automático

Não deixe de rodar o corretor para **todas** as bases de teste

```
> python3    waxm_corretor_page_rank.pyc   base0   pagerank.py
> python3    waxm_corretor_page_rank.pyc   base1   pagerank.py
> python3    waxm_corretor_page_rank.pyc   base2   pagerank.py
```