Projeto A2: PageRank

Fundação Getúlio Vargas - Escola de Matemática Aplicada
Bacharelado em Ciência de Dados
Curso de Álgebra Linear
Professor: Yuri Fahham Saporito
Alunos: Gianlucca Devigili e Maisa O. Fraiz

1. Intrudução

O presente trabalho se trata da implementação de uma adaptação do algoritmo PageRank, desenvolvido por Sergey Bin e Larry Page, de quem o algoritmo leva o nome, para grafos não direcionados aplicados às fronteiras dos Estados dos EUA, de modo a calcular os estados com mais fronteiras. Para tal utilizamos de bibliotecas python como networkx para gerar o grafo e matplotlib para sua vizualização.

2. O algoritmo PageRank

O Pagerank é um algoritmo criado por Sergey Brin e Larry Page, fundadores da multinacional Google, em 1996. Ele foi criado com a função de servir como uma métrica para estimar a importância das páginas na internet, organização do sistema de busca de forma que os resultados mais relevantes apareçam primeiro para o usuário.

De acordo com Google (2020), o a relevância de uma página é calculada, dentre diversos outros fatores decorridos da sofisticação do algoritmo, através da relevância das páginas que possuem links que apontem para ela.

O cálculo do PageRank se dá por meio de quantos links existentes se conectam para uma página P qualquer. Cada página P_j contém L_j links. Se um link de P_j redireciona para P_i , então P_i receberá $\frac{1}{L_j}$ do PageRank de P_j. Considere B_i como o conjunto de páginas cujos links redirecionam para P_i. O PageRank de P_i será:

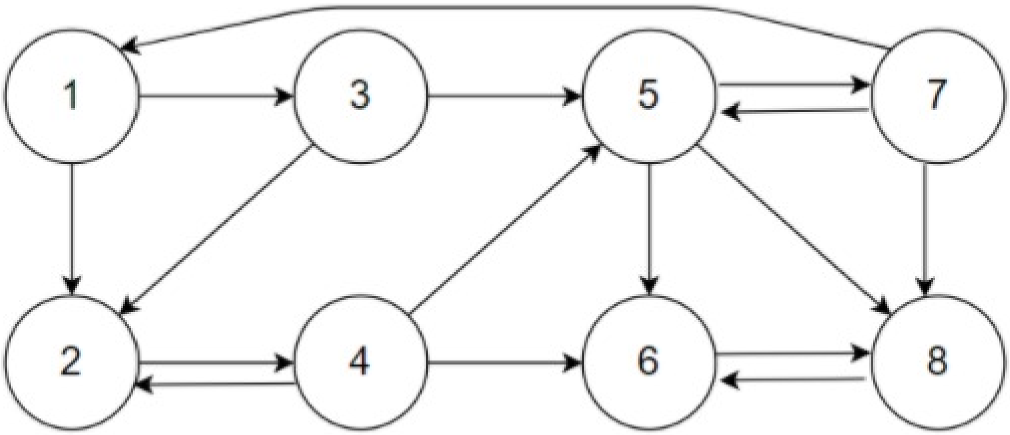
$$PR(Pi) = \sum_{Pj \in Bi} \frac{PR(Pj)}{Lj}$$

Para calcular o PageRank usando Álgebra Linear, é criada uma matriz A tal que cada entrada A_{ij} será $\frac{1}{L_j}$ se P_j tiver um link que redirecione para P_i. Se P_j e P_i não forem conectados, A_{ij} será nulo. Enquanto P_j conter pelo menos um link, A será uma matriz de Markov. O PageRank pode ser calculado descobrindo o autovetor estacionário de A, ou seja, o vetor I tal que $AI = I$.

2.1. Exemplo com grafo simplificado

Utilizaremos aqui o exemplo dado por Austin (2006): Imagine que existem apenas 8 páginas representadas pelo seguinte grafo:

Figura 1: Grafo Exemplo



Fonte: AUSTIN, 2006

Cada nó (vértice) do grafo é uma página e cada aresta indica um link entre duas páginas. A matriz relacionada ao exemplo é:

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 0 \\ 1/2 & 0 & 1/2 & 1/3 & 0 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 1/3 & 1/3 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/3 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/3 & 1 & 1/3 & 0 \end{bmatrix}$$

que tem como vetor estácionário (pageranks):

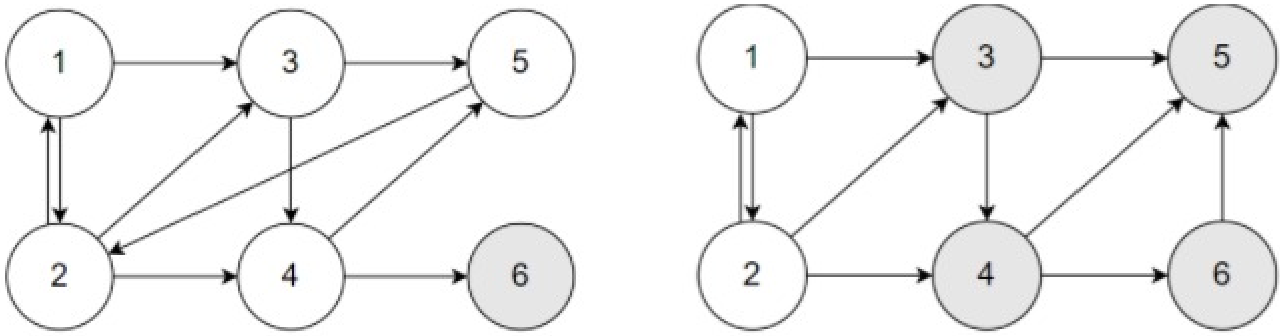
$$I = \begin{bmatrix} 0.0600 \\ 0.0675 \\ 0.0300 \\ 0.0675 \\ 0.0975 \\ 0.2025 \\ 0.1800 \\ 0.2950 \end{bmatrix}$$

No exemplo, o página 8 tem uma relevância maior, ou seja, em um algoritmo de pesquisa consideraria ela mais relevante e apareceria por primeiro, sendo seguida pela página 6 e assim por diante.

2.2. Possíveis problemas

No caso da coluna A_j conter apenas valores nulos, significa que o vértice P_j é um vértice disconexo. Também é possível haver um conjunto de páginas que formem um ciclo de links entre si, de forma que vez que o usuário entra nelas, não é possível sair. Em ambos dos casos, o cálculo do PageRank falha.

Figura 2: Grafo com Erros no PageRank



Fonte: AUSTIN, 2006

Esse problema pode ser resolvido escolhendo um valor $0 \leq \alpha \leq 1$ que determina a probabilidade de, ao percorrer o grafo, ser redirecionado para uma nó qualquer, independentemente das ligações. Quanto mais próximo α for de 1, mais peso têm as ligações e mais tempo levará o processo para descobrir o autovetor. A Google usa $\alpha = 0.85$, levando entre 50 e 100 iterações do método de potencialização para achar valores de PageRank satisfatoriamente aproximados.

2.3. Outras Aplicações

O PageRank também é muito utilizado fora da Google, como na medicina, no desenvolvimento de softwares, no esporte e na bibliometria. Dentre algumas aplicações do PageRank, estão:

- Utilizando $\alpha = 0.92$ em uma rede de interações de proteínas, o pagerank pode ser usado para descobrir quais genes estão relacionados com a diabetes tipo 2;
- Estudos acerca de um tipo de câncer pancreático, que encontram genes que preveem se o paciente sobreviveria à doença com um pagerank utilizando $\alpha = 0.3$;
- Algoritmo **Monitor Rank:** ao retornar uma lista ordenada dos possíveis responsáveis por um erro na programação, utilizado por administradores de sistemas para o diagnóstico e solução de erros;
- Dados geográfico como prever tráfego e movimento humano utilizando um grafo onde as ruas são representadas por arestas e suas intersecções por vértices. Neste caso α é gerado a cada iteração de acordo com a probabilidade da viagem acabar em determinada rua. Tal tipo de aplicação é usado em softwares de transporte urbano que fazem uso de GPS;
- Criar uma rede de vencedores em esporte, onde cada time é um nó e cada jogo é uma linha. Em uma partida entre dois times, A e B, o time que ganha passa seus pontos para o outro;
- Pode ser usado para medir a influência de revistas científicas e artigos com base nas citações entre elas;
- Algoritmo **ItemRank**: utilizado para recomendar itens como produtos em *e-commerces* ou filmes e séries em plataformas de _streaming_;
- Em redes sociais, o pagerank pode ser usado para prever potenciais conexões e amizades entre usuários, recomendar perfis a serem seguidos e estimar a influência dos usuários;
- Também existem o **TrustRank** e **BadRank**, que analisam a possibilidade de um site estar abusando de spam para aumentar o seu PageRank, tais algoritmos são utilizados pelo *Google* e outros sistemas de pesquisa para evitar relatarem como relevantes página.

(GLEICH, 2014), (MILLER, 2020), (ASP, 2015).

3. Implementação

Para a implementação do algoritmo de pagerank utilizada no presente trabalho foi utilizado um grafo não direcionado onde cada nó (vértice) representa um dos Estados dos EUA e a existência de uma aresta entre dois nós indica que os dois estados fazem fronteira entre si.

Foram removidos os Estados do Havaí e do Alaska pois, como estes não possuem fronteiras com os demais Estados, o grafo seria desconexo caso estes estivesse inclusos, gerando assim colunas preenchidas inteiramente por zeros na matriz, impossibilitando o cálculo do pagerank

In [2]:

```
import re, sys, math, csv, types
import networkx as nx
import matplotlib.pyplot as plt
import plotly.graph_objects as go
```

A implementação do algoritmo pagerank no presente trabalho é uma adaptação de ASP (2015) para grafos não direcionados. Utilizamos o dataset *"stateborders.csv"* que contém todos os Estados dos EUA e suas respectivas fronteiras, onde a primeira e terceira coluna indicam uma aresta do grafo que representa o país[1]. Abaixo seguem as primeiras 10 linhas de amostra para visualização:

[1]. O *dataset* se encontra dessa maneira pois o algoritmo de Asp (2015) é utilizado para calcular o pagerank de outros datasets que fazem uso da segunda e da quarta coluna

In [4]:

```
filename = "stateborders.csv"
file = csv.reader(open(filename, 'r'), delimiter = ',')
data = [row for row in file]
for i_row in range(10):
    print(data[i_row])
```

```
['AL', '0', 'FL', '0']
['AL', '0', 'GA', '0']
['AL', '0', 'MS', '0']
['AL', '0', 'TN', '0']
['AZ', '0', 'CA', '0']
['AZ', '0', 'NM', '0']
['AZ', '0', 'NV', '0']
['AZ', '0', 'UT', '0']
['AR', '0', 'LA', '0']
['AR', '0', 'MS', '0']
```

In [ ]:

```

```

O grafo foi gerado com o *package networkx* onde cada vértice representa um Estados e a existência de uma aresta uv indica que o Estado u faz fronteira com o Estado v. O grafo gerado não é direcionado, como o habitual em parte das aplicações do *pagerank* pois trata-se de dados geográficos, ou seja,o Estado u ter fronteira com o Estado v implica que v faz fronteira com u

In [6]:

```
nodes = set([row[0] for row in data])
print(f'Nós do grafo: {nodes}', sep=" ")
```

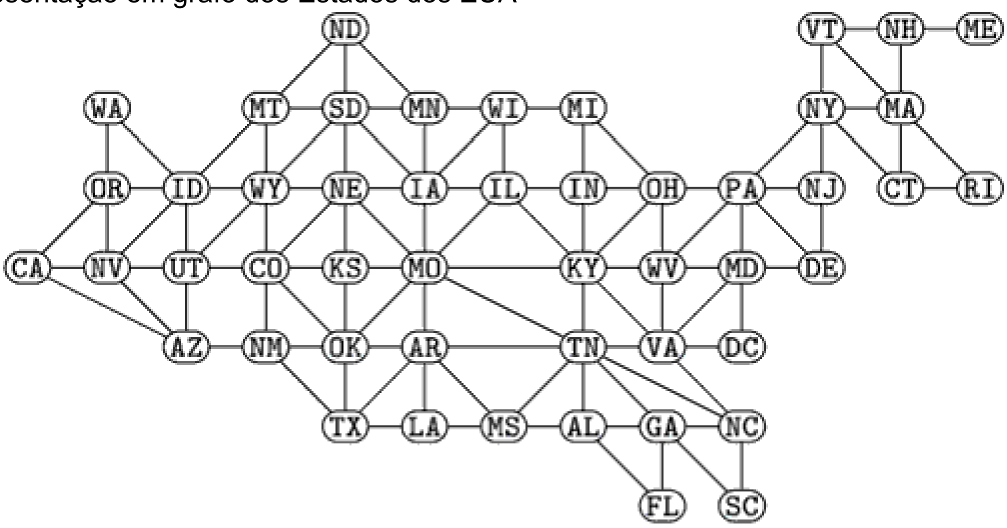
```
Nós do grafo: {'KY', 'ME', 'AR', 'RI', 'FL', 'MT', 'ID', 'CT', 'UT', 'MO',
'GA', 'WA', 'WI', 'MA', 'IN', 'AZ', 'OK', 'IA', 'MN', 'NY', 'CA', 'MI', 'N
M', 'TN', 'VT', 'WY', 'NV', 'SD', 'NE', 'SC', 'DC', 'ND', 'LA', 'DE', 'IL',
'TX', 'OH', 'CO', 'PA', 'MD', 'AL', 'NH', 'NC', 'WV', 'MS', 'NJ', 'VA', 'K
S', 'OR'}
```

In [8]:

```
edges = [(row[0], row[2]) for row in data]
print("Exemplos de Aresta: ", end= "")
for i_edge in range(10):
    print(edges[i_edge], end=", ")
```

```
Exemplos de Aresta: ('AL', 'FL'), ('AL', 'GA'), ('AL', 'MS'), ('AL', 'TN'),
('AZ', 'CA'), ('AZ', 'NM'), ('AZ', 'NV'), ('AZ', 'UT'), ('AR', 'LA'), ('AR',
'MS'),
```

Figura 3: Representação em grafo dos Estados dos EUA



Fonte: (WEISSTEIN, 201-)

Os Estados do Alaska e Havaí foram removidos pois não possuem *"links"* com os demais estados, o que geraria erro no algoritmo como explicado anteriormente

Então calculamos o valor base de cada rank é dado pela razão:

$$\frac{1}{|V(G)|}$$

onde $|V(G)|$ é o número de nós do grafo. No presente exemplo o rank base de cada nó é:

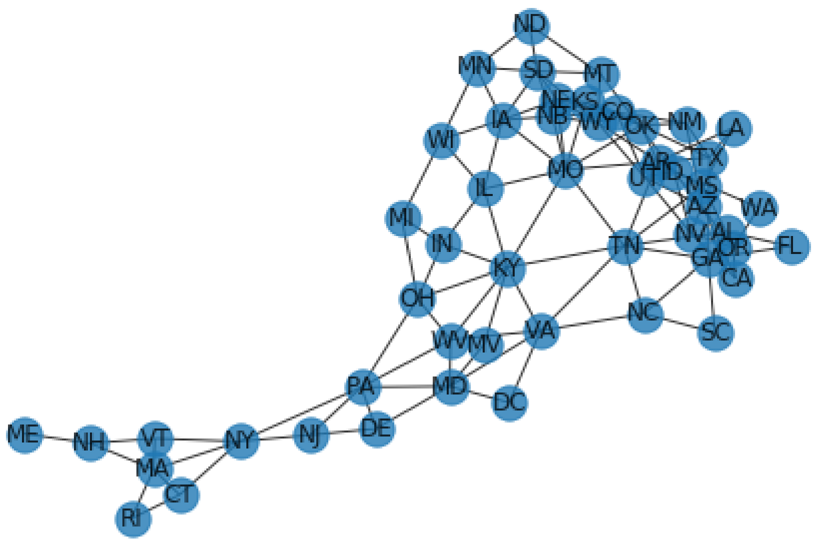
In [10]:

```
rank = 1/float(len(nodes))
print(f'{rank}')
```

```
0.02040816326530612
```

In [15]:

```
graph = nx.Graph()
graph.add_nodes_from(nodes, rank=rank)
graph.add_edges_from(edges)
nx.draw_spring(graph, with_labels =  True, alpha = 0.8)
```



In [ ]:

```
### 3.1. O Algoritmo em python

O cálculo do _pagerank_ em si é dado pelo algoritmo abaixo:
```

In [ ]:

```
V = len(graph)
alpha = 0.85
```

Utilizamos $\alpha = 0.85$, o mesmo utilizado pelo *Google* de modo a evitar que loops infinitos

In [14]:

```
ranks = dict()

for key, node in graph.nodes(data=True):
    ranks[key] = node.get("rank")

for i in range(10):
    for key, node in graph.nodes(data=True):
        rank_sum = 0
        curr_rank = node.get('rank')

        neighbors = graph[key]
        for n in neighbors:
            if ranks[n] is not None:
                outlinks = len(list(graph.neighbors(n)))
                rank_sum += (1 / float(outlinks)) * ranks[n]

        # O cálculo do pagerank em si
        ranks[key] = ((1 - float(alpha)) * (1/float(V))) + alpha * rank_sum

ranks
```

Out[14]:

```
{'KY': 0.031098238496603155,
 'ME': 0.008690297327355826,
 'AR': 0.023272239793941652,
 'RI': 0.012819113318186455,
 'FL': 0.01050084530942565,
 'MT': 0.01645788646439453,
 'ID': 0.025068637345229045,
 'CT': 0.017690505344719405,
 'UT': 0.0200971726775254,
 'MO': 0.03158022671024823,
 'GA': 0.026277407926118906,
 'WA': 0.010461653121992995,
 'WI': 0.016707964193556007,
 'MA': 0.028677564943525295,
 'IN': 0.017014006909051325,
 'AZ': 0.01757704202637932,
 'OK': 0.022328991174056862,
 'IA': 0.02549420393068461,
 'MN': 0.016436311442864 6,
 'NY': 0.026088829248207206,
 'CA': 0.014328656298116398,
 'MI': 0.013690965450595 7,
 'NM': 0.01637268139750077,
 'TN': 0.031099642920266485,
 'VT': 0.018000584339950353,
 'WY': 0.025696465332665695,
 'NV': 0.021673079858910902,
 'SD': 0.02592563129727083,
 'NE': 0.021455369137679283,
 'SC': 0.010480612349900652,
 'DC': 0.010270436314230656,
 'ND': 0.013079305897692257,
 'LA': 0.013347302492903406,
 'DE': 0.014883922298236787,
```

```
 'IL': 0.01948958490024708,
 'TX': 0.020384914150546413,
 'OH': 0.021106138950102136,
 'CO': 0.02529579239547562,
 'PA': 0.02688236817705955,
 'MD': 0.02670263765534503,
 'AL': 0.018076694914764355,
 'NH': 0.02030328080222592,
 'NC': 0.017972779060813345,
 'WV': 0.020974989583665837,
 'MS': 0.017165447544051676,
 'NJ': 0.015401724252367318,
 'VA': 0.025082474252165237,
 'KS': 0.01823608227862098,
 'OR': 0.018682978865157206,
 'NB': 0.021459641569701694,
 'MV': 0.0100282337078692865}
```

Referências

AUSTIN, David. **How Google Finds Your Needle in the Web's Haystack. American Mathematical Society**, 2006. Disponível em: http://www.ams.org/publicoutreach/feature-column/fcarc-pagerank (http://www.ams.org/publicoutreach/feature-column/fcarc-pagerank). Acesso em: 14 nov. 2020.

ASP, Timothy. **PageRank.** 2015. Disponível em: https://github.com/timothyasp/PageRank (https://github.com/timothyasp/PageRank). Acesso em: 18 nov. 2020.

GLEICH, David F. **PAGERANK BEYOND THE WEB**. 2014. Disponível em: https://arxiv.org/pdf/1407.5107.pdf (https://arxiv.org/pdf/1407.5107.pdf). Acesso em: 16 nov. 2020.

GOOGLE. **Como funcionam os algoritmos da Pesquisa:** classificar páginas úteis. Classificar páginas úteis. Disponível em: https://www.google.com/search/howsearchworks/algorithms/ (https://www.google.com/search/howsearchworks/algorithms/). Acesso em: 14 nov. 2020.

MILLER, Colton. **A HISTORY LESSON ON PAGERANK**. 2020. Disponível em: https://www.boostability.com/a-history-lesson-on-pagerank/#:~:text=Google%20founders%20Larry%20Page%20and,backbone%20behind%20Google%20search% (https://www.boostability.com/a-history-lesson-on-pagerank/#:~:text=Google%20founders%20Larry%20Page%20and,backbone%20behind%20Google%20search% Acesso em: 14 nov. 2020.

WEISSTEIN, Eric W. **Contiguous USA Graph.** 201-. Disponível em: https://mathworld.wolfram.com/ContiguousUSAGraph.html (https://mathworld.wolfram.com/ContiguousUSAGraph.html). Acesso em: 14 nov. 2020.