Implementação paralela do algoritmo PageRank em Pyspark

Eulaliane Aparecida Gonçalves

INF, Universidade Federal do ABC, Brasil

INFORMAÇÕES DO ARTIGO

História do Artigo:

Recebido 27 de setembro de 2017

Recebido em forma revisada 04 de dezembro de 2017

Aceito 08 de dezembro de 2017

Palavras-chave:

PageRank

Página Web

Pyspark

RESUMO

O crescimento exponencial da World Wide Web impulsionou o desenvolvimento de metodologias de avaliação das páginas web, visando identificar a relevância de uma página, para os motores de busca. Diante deste contexto, este artigo apresenta o funcionamento do algoritmo PageRank e sua implementação de forma paralela por meio do PySpark. A metodologia é fundamentada em pesquisa bibliográfica, com embasamento teórico sobre o funcionamento do Algoritmo PageRank para melhor compreensão de sua implementação.

1 INTRODUÇÃO

São inúmeras as informações disponíveis na rede web, assim como os internautas em busca dessas informações, o que torna necessário a automatização desse processo e a garantia de confiabilidade das informações encontradas.

Essa automatização é feita pelo uso dos motores de busca da web, um sistema que permite pesquisar informações na rede e apresentar os resultados aos usuários. Mas para garantir a confiabilidade é imprescindível medir a importância de uma página web, porém, só será possível tal implementação dado a ocorrência de vários critérios, como a freqüência relativa de dados, sua localização e sua exposição.

Com base nesse sistema, o algoritmo PageRank tem como objetivo tornar esses motores de busca resistentes às páginas da web filtradas por meio dessa medição, que avalia a popularidade entre as páginas. Fazendo com que o número de ligações de uma página web a outras páginas influenciem em sua avaliação.

2 ALGORITMO PAGERANK

O algoritmo PageRank classifica uma página web de acordo com as páginas que estão vinculadas a ela, e a quantidade de vínculos geram o seu rank. O que faz do algoritmo um método recursivo, aplicado em toda rede web.

Figura 1: PageRank

$$\boldsymbol{PR(P_0) = p \times \sum_{i=1}^{n} \frac{PR(P_i)}{c(P_i)} + \frac{1-p}{n}}$$

Fonte: VASCONCELOS

O **PR(P_0)** é o PageRank de P_0, **PR(P_i)** é o PageRank de P_i, que está vinculada a P_0. **C(P_i)** é o número de vínculos, **n** é o número de páginas, e **p** é fator. A ideia da fórmula é criar uma distribuição de probabilidade sobre as páginas web.

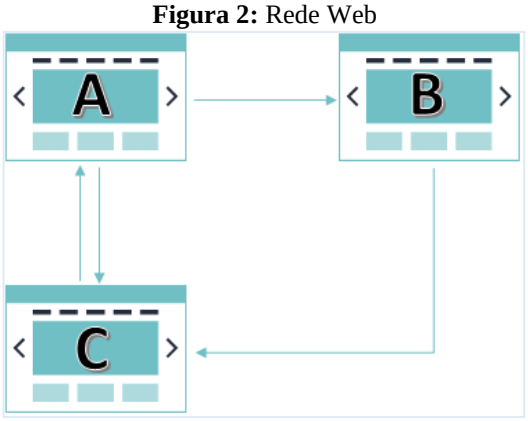
2.1. Funcionamento do algoritmo

Um cálculo matemático é realizado para saber quantos links cada página web recebeu, emitindo quais são os links de entrada e a nota da página, em seguida, a nota atribuída a página é dividida entre todos os links, ou seja, entre as páginas vinculadas.

2.1.1. Exemplo do PageRank

Dado uma rede web composta por três páginas: A, B e C, onde as setas indicam os links criados entre as páginas.

Figura 2: Rede Web



Fonte: Elaborado pelo autor, com base em VASCONCELOS

$$PR(P_A) = 0{,}85 \times PR(P_C) + 0{,}15$$

$$PR(P_B) = 0{,}85 \times PR\left(\frac{P_A}{2}\right) + 0{,}15$$

$$PR(P_C) = 0{,}85 \times \left(PR\left(\frac{P_A}{2}\right) + PR(P_B)\right) + 0{,}15$$

Totalizando três PageRank, o maior rank obtido é o da página C, que recebe dois links, página A e B. Já em segundo lugar no rank fica a página A, que recebe o link da página com maior rank, a página C.

2.2. Implementação paralela do Algoritmo PageRank

O algoritmo será implementado utilizando a linguagem de programação Python para o Spark, Pyspark, que permite o processamento distribuído de grandes volumes de dados.

Conforme apresentado no exemplo de PageRank, foram utilizadas três páginas linkadas entre si, e aplicada a fórmula da figura 1, resultando no PageRank entre as páginas A, B e C.

```
sc = SparkContext.getOrCreate()

def atualizar(urls, rank):
    for i in urls: yield(i, rank/len(urls))

paginas = "paginaA","paginaB"
paginas0 = "paginaB","paginaC"
paginas1 = "paginaC","paginaA"
paginas2 = "paginaA","paginaC"
```

```
links = (sc
.parallelize([paginas,paginas0,paginas1,p
aginas2]).map(lambda x: [x[0],x[1]])
.cache())

grupos = (links.groupByKey().cache())

ranks = (grupos.map(lambda x: (x[0],
1.0)))

for x in range(2): gruposRankeados =
grupos.join(ranks)

rankAtual = (gruposRankeados
.flatMap(lambda x: atualizar(x[1][0],
x[1][1])))

rankFinal = (rankAtual
.reduceByKey(lambda x,y: x+y)
.mapValues(lambda d: d * 0.85 + 0.15))

for (pagina, rank) in
rankFinal.sortBy(lambda x:-x[1]).take(3):
    print("\n\nA %s está com %s no
ranking." % (pagina, rank))
```

2.2.1. Explorando o Algoritmo

sc: cria uma conexão com um cluster Spark.

def atualizar: função utilizada para percorrer todos as páginas e seus respectivos ranks.

paginas ... paginas2: são as páginas que irão compor o rank.

links: as páginas são transformadas em uma RDD de listas.

grupos: as listas são agrupadas pela chave.

ranks: atribui o valor 1 para todas as chaves.

gruposRankeados: Adiciona ranks inicializados com 1.0 na posição [1][1] da matriz.

rankAtual: aplica a função atualizar no grupo criado anteriormente.

rankFinal: soma todos os valores que contém o mesmo id.

for: percorre o rank aleatoriamente e retorna os 3 primeiros colocados.

2.2.2. Resultado do Algoritmo

A paginaC está com 1.425 no ranking.
A paginaA está com 1.0 no ranking.
A paginaB está com 0.575 no ranking

Assim como foi demonstrado no exemplo do algoritmo PageRank, a implementação em Pyspark resultou na exata classificação. O maior rank foi obtido pela página C, e o segundo pela página A, deixando a página B na terceira colocação.

3 CONCLUSÃO

Este artigo, apresentou o funcionamento e os cálculos utilizados para obtenção do PageRank. Que apesar de conter centenas de milhares de páginas web, comprovou que relacionar essas páginas não é tão complexo como parece.

O algoritmo apresenta uma fórmula de simples implementação, já que sua formulação consiste em conceitos de probabilidade e álgebra linear. O que permitiu a implementação do algoritmo em um ambiente distribuído, o Pyspark, com a finalidade de mostrar a otimização de um sistema de busca.

4 REFERÊNCIAS

[1] SOBEK, Markus. **A Survey of Google's PageRank**. eFactory. Disponível em: < http://pr.efactory.de/>. Acessado em: 06 dezembro 2017, 14:55:00.

[2] ROGERS, Ian. **The Google Pagerank Algorithm and How It Works**. Disponível em: <http://www.cs.princeton.edu/~chazelle/courses/BIB/pagerank.htm>. Acessado em: 06 dezembro 2017, 16:36:00.

[3] FRASSON, Prof. Miguel. **O algoritmo PageRank do Google.** Depto. de Matemática Aplicada e Estatística – SME ICMC-USP. 27 de novembro de 2015. Disponível em: <http://conteudo.icmc.usp.br/pessoas/frasson/AL/2015-2sem/frasson-google.pdf>. Acessado em: 07 dezembro 2017, 20:17:00.

[4] VASCONCELOS, Paulo. **Google PageRank: matemática básica e métodos numéricos**. CMUP. Disponível em: <http://cmup.fc.up.pt/cmup/mecs/googlePR.pdf>. Acessado em: 07 dezembro 2017, 23:52:00.