Projeto Big Data e Inteligência na Web: Implementação Distribuída do PageRank

Daniel Gonçalves da Silva

Introdução

Como parte da matéria de “*Big Data* e Inteligência na *Web*” do curso de Ciência da Computação (Mestrado da Universidade Federal do ABC - UFABC), foi implementado o algoritmo distribuído do PageRank. Este algoritmo é um dos mais famosos utilizados pela Google para construir um ranking de páginas da web por relevância, dando a cada página uma classificação de “qualidade”. A implementação distribuída do algoritmo foi realizada via MapReduce com o Apache Spark.

Um *IPython Notebook* foi utilizado neste projeto contendo todos os códigos propostos com este relatório.

Algoritmo PageRank

O objetivo do PageRank é conseguir um ranking de páginas da web com base em sua relevância, a partir de uma representação dessas páginas na forma de um grafo (onde um nó aponta para outro nó, criando uma referência).

É uma distribuição de probabilidade em todos os nós, em outras palavras, em todas as páginas da web. Numa visão de matemática, o PageRank de uma página n, notado P(n), poderia ser representado pela seguinte relação recursiva:

$$P(n) = \sum_{n \in L(n)} , \frac{P(m)}{C(m)}$$

Onde L (n) é o conjunto de páginas que conduzem a n e C (m) é o número de links da página m. A página n recebe contribuições do conjunto de páginas que o conduzem, sendo estas maiores quanod as páginas que o conduzem, sendo estas maiores quando as páginas são referenciadas pelo algoritmo. Isso é bastante intuitivo e nos dá orientação sobre o que deve ser uma página da Web bem referenciada. O PageRank é uma medida da relevância ou qualidade de uma página, não é surpreendente que as páginas mais referenciadas sejam aquelas às quais um grande número de páginas se referem. Além disso, a qualidade da página aumentará se as páginas que o referenciam sejam páginas de qualidade, ou seja, páginas com um PageRank alto.

Materiais e Métodos

Versões Utilizadas no Experimento

- Python 3.5.2;
- IPython 6.2.1;
- Jupyter Notebook 5.2.2;
- Spark 2.0.0;
- Hadoop 2.7.

Fluxograma da Implementação do PageRank

A implementação do algoritmo do PageRank seguiu o fluxograma abaixo, que pode ser entendido pela sequência:

1. Abre arquivo e divide em primeiro e segundo valor assim que encontrar espaço;
2. Após dividido, é agrupados todos os id que possui mesma referência;
3. É adicionado o valor de rank inicializado por 1.0 na posição [1][1] da matriz;
4. É agrupado o ID do link com os ranks;
5. Para cada valor de ID é atualizado o valor do rank chamando uma função que atualiza o rank (flatMap);
6. A última etapa é agrupar (reduce) com mesmo valor de ID e aplicar o fator de normalização que multiplica por 0.85 e soma 0.15.

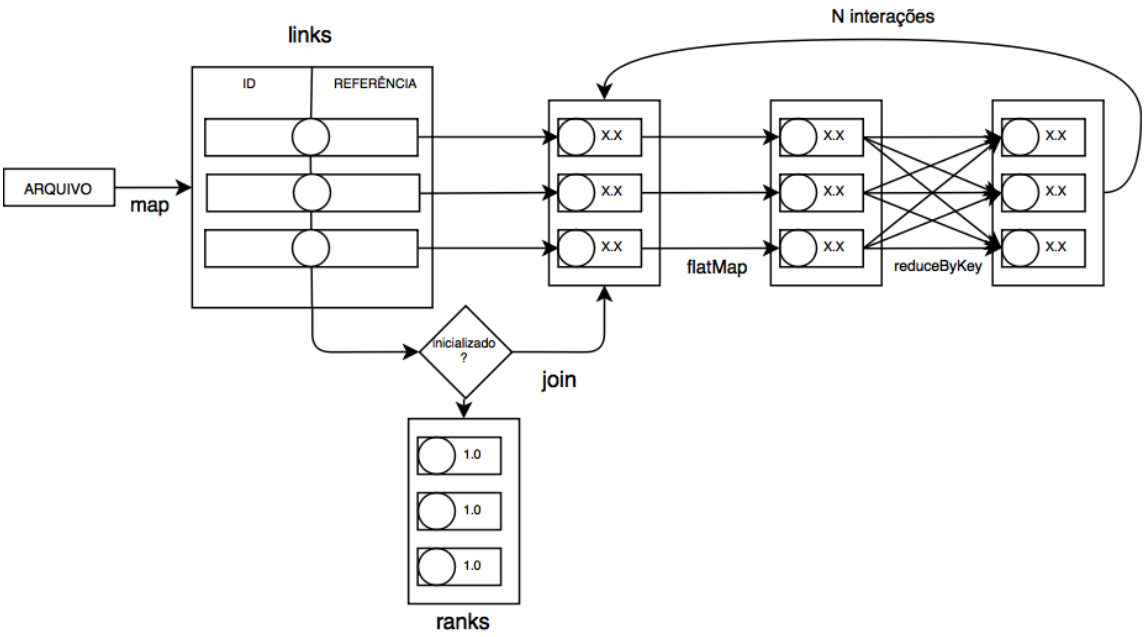

Figura 1: Fluxograma da Implementação do PageRank

Aplicação do PageRank na Base de Teste

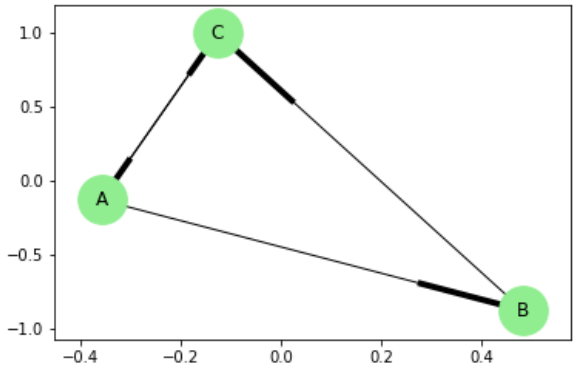

Figura 2: ilustração do relacionamento entre os nós

Na figura acima, temos uma suposição de relacionamento entre os nós onde A referencia B e C, B referencia C e C referencia A, ilustrada abaixo:

	from	to
0	A	C
1	B	C
2	C	A
3	A	B

Figura 3:
base de teste

Nessa suposição, temos como melhor referencia o nó C o qual recebe referência tanto de A como de B. Este exemplo ilustrado foi aplicado no Jupyter Notebook neste link (https://github.com/danielgoncalvesti/BIGDATA2017/blob/master/Projeto/pagerank-webgoogle.ipynb). Abaixo temos o resultado do ranking executado no Jupyter Notebook, com apenas uma iteração:

```
ID: C  Ranking: 1.4249999999999998.
ID: A  Ranking: 1.0.
ID: B  Ranking: 0.575.
```

Figura 4: Resultado da base de teste

A aplicação na base real (web-Google.txt – 5.105.043 registros) pode ser acessada em:
https://github.com/danielgoncalvesti/BIGDATA2017/blob/master/Projeto/pagerank-webgoogle.ipynb

Conclusão

Este é um exemplo bastante intuitivo de aplicação do paradigma MapReduce, o que me ajudou a ver mais claramente sobre a distribuição de cálculos em geral. Na era do Big Data, essas habilidades são essenciais, já que é necessário distribuir o processamento para cada nó da rede e transformar os dados antes de enviá-los para um única máquina que fará a tarefa de juntar as respostas das transformações dos dados um único resultado.

Referências

1. Holden Karau, Andy Konwinski, Patrick Wendell. Learning Spark: Lightning-Fast Big Data Analysis
2. Acessado em 20 de Novembro de 2017: https://en.wikipedia.org/wiki/PageRank
3. Acessado em 22 de Novembro de 2017: https://snap.stanford.edu/data/web-Google.html
4. Acessado em 22 de Novembro de 2017: https://github.com/spark-notebook/spark-notebook