Universidade Autónoma de Lisboa

Apresentação do artigo “A PageRank Algorithm based on Gauss-Seidel iterations”

André Ferreira Nº20151076

18 de Dezembro de 2019

Introdução

- Abordamos o problema do PageRank de associar um valor de importância relativa a todas as páginas da Web na Internet, para que um mecanismo de pesquisa possa usá-los para classificar as páginas que serão exibidas ao utilizador.
- A principal ideia do algoritmo PageRank é que a classificação de uma página da Web dependa do número de links que se conectam a essa página e da classificação correspondente.
- A principal contribuição desta apresentação é apresentar um algoritmo distribuído a ser executado por uma família de processadores, cada um armazenando um subconjunto de todo o gráfico de rede. Essa solução atende a dois critérios: i) converge mais rapidamente para a solução do que o Power Method; ii) minimiza as comunicações entre diferentes processadores.

Intuição por detrás do PageRank

- O PageRank procura uma classificação normalizada que depende do número de vizinhos e da classificação desses nós. Isso se traduz na classificação do nó i, sendo a soma ponderada das classificações de seus vizinhos

- Para evitar o problema de múltiplos valores próprios iguais a um, o trabalho propõe encontrar o vetor próprio da matriz a seguir M ∈Rn×n:

$$x^\star = Mx^\star, \quad x^\star \in [0,1]^n, \quad \sum_{i=1}^{n} x_i^\star = 1$$

$$M = (1-m)A + \frac{m}{n}S, \quad m = 0.15, \quad S := 1_n 1_n^\intercal$$

Formulações equivalentes para o PageRank [1]

- O Power Method procura o eigenvector
- Usando o Power Method torna-se:

$$x(k+1) = Mx(k) = (1-m)Ax(k) + \frac{m}{n}1_n$$

- O rank vector também é um eigenvector:

$$(I_n - (1-m)A)x = \frac{m}{n}1_n.$$

Problema

- Encontrar um algoritmo que possa explorar o fato de que subconjuntos de páginas são armazenados no mesmo processador;
- Resolver a equação:

$$(I_n - (1-m)A)x = \frac{m}{n}1_n$$

- Com restrições:

$$x^\star \in [0,1]^n, \quad \sum_{i=1}^{n} x_i^\star = 1$$

Power Method como Jacobi Method

- O Jacobi Method é dado pela iteração:

$$x(k+1) = D^{-1}(b - Rx(k))$$

- Para uma equação linear $Ax = b$ partição $A = D + R$, onde $D = \text{diag}(A)$ e $R = A - \text{diag}(A)$;

- O Jacobi Method é normalmente usado para resolver equações diagonalmente dominantes.

PageRank usando Gauss-Seidel

- Particionamento A = L + D + U, para matrizes triangulares inferiores e superiores L e U e diagonal D;
- O Gauss-Seidel Method torna-se:

$$x(k+1) = (L+D)^{-1}(b - Ux(k))$$

- O método pode ser escrito para aproveitar os valores atualizados

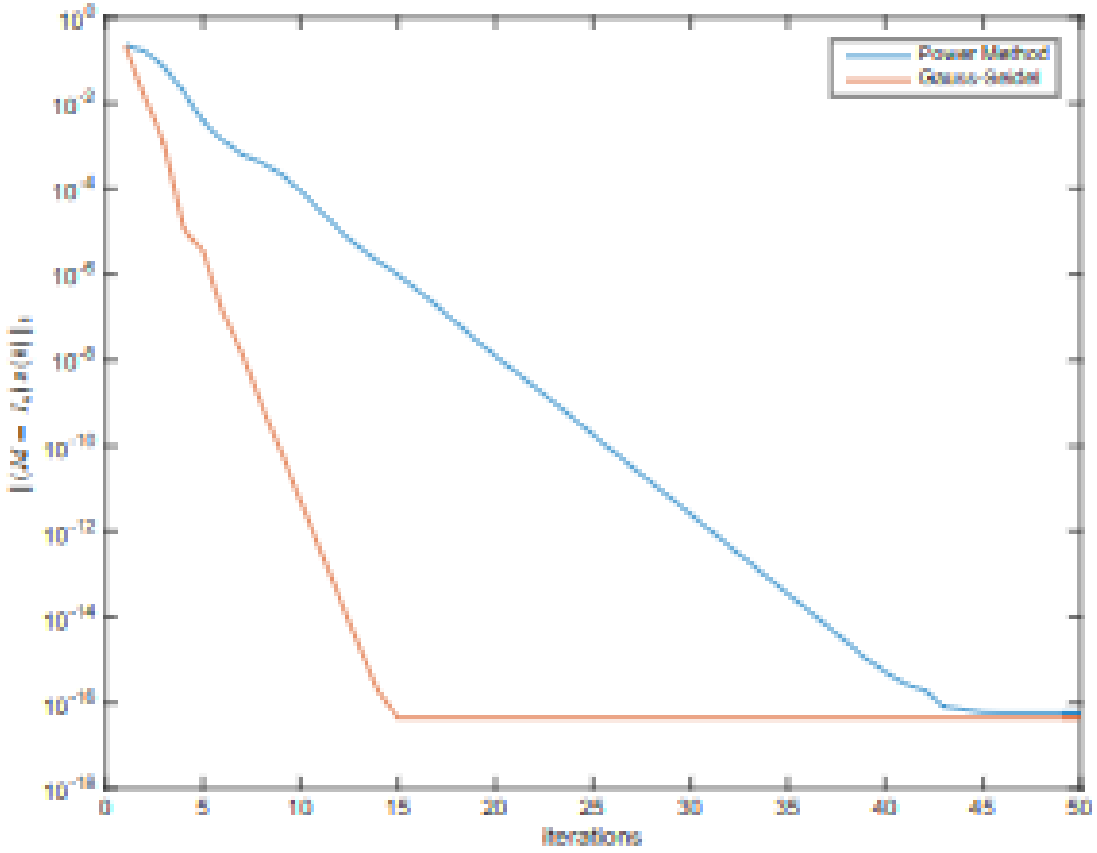
$$x_i(k+1) = \frac{1}{A_{ii}} \left(b_i - \sum_{j=1}^{i-1} A_{ij} x_j(k+1) - \sum_{j=i+1}^{n} A_{ij} x_j(k) \right)$$

Resultados

- Resultado teórico: $\rho(T_{gs}) < \rho(T_p)$
- É simulado:
  - Projeção no n-simplex;
  - Normalização;
  - Versões aleatórias e assincronas.

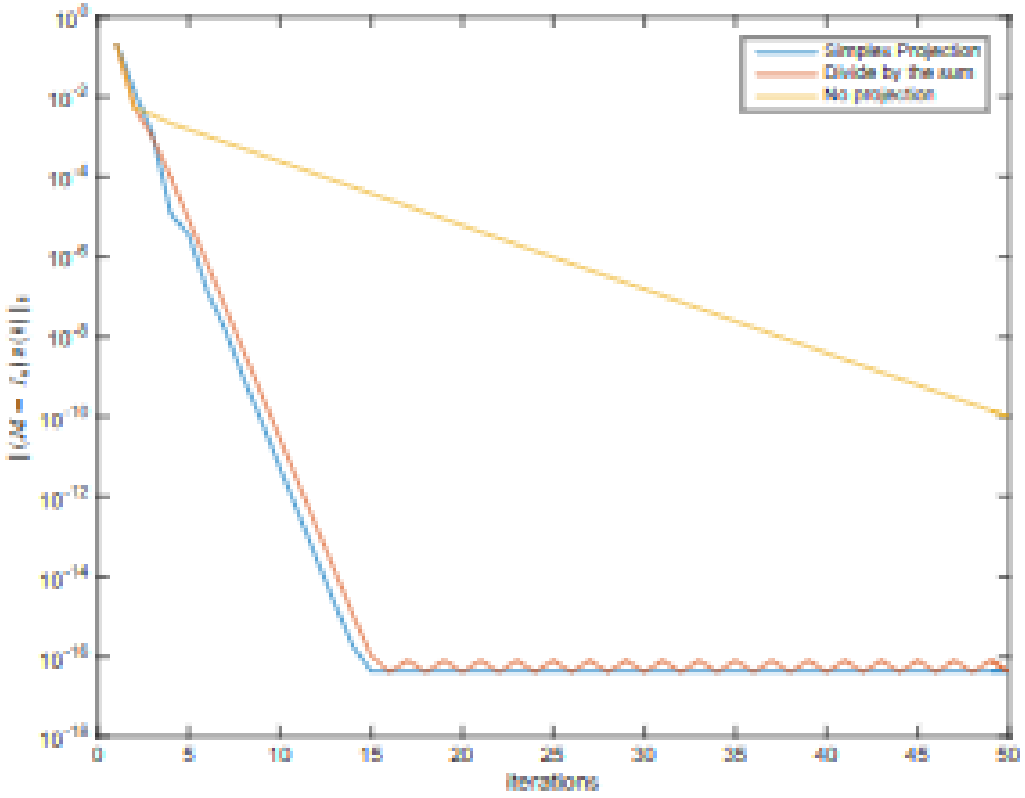
Simulação de Resultados [2]

- Evolução do erro do Power Method e Gauss-Seidel com uma projecção n-simplex em escala logarítmica.
- Gauss-Seidel é mais rápido que o Power Method



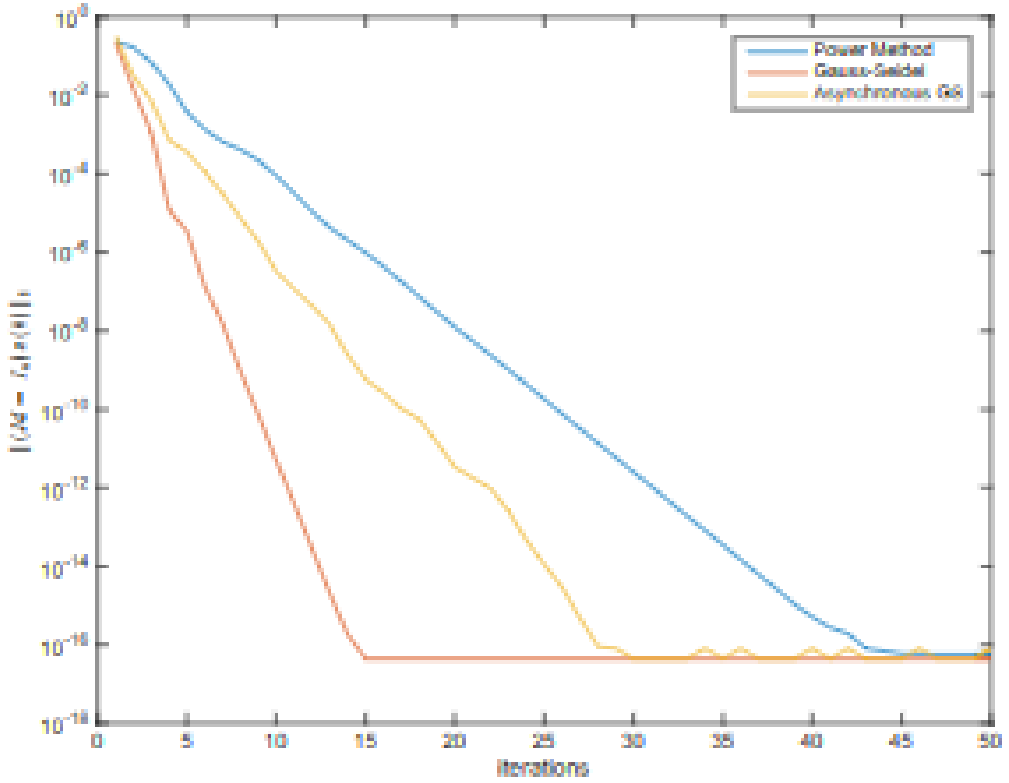
Simulação de Resultados

- O impacto das projeções na convergência do algoritmo.
- A performance é afetada com “no projection”
- O “Simplex Projection” e o “Divide by Sum” são semelhantes



Simulação de Resultados

- Gauss-Seidel sequencial contra erros assíncronos de Gauss-Seidel.
- A versão assíncrona possui um desempenho intermediário.



Conclusões finais

- Mostrou-se que o Gauss-Seidel é mais rápido do que o Power Method;
- Gauss-Seidel não mantêm o “Sum of the state”;
- Projeção: Implica duas rondas de Comunicações;
- Normalização: Não trouxe maior complexidade;

References:

[1] D. Silvestre, J. Hespanha and C. Silvestre, "A PageRank Algorithm based on Asynchronous Gauss-Seidel Iterations," *2018 Annual American Control Conference (ACC)*, Milwaukee, WI, 2018, pp. 484-489. doi: 10.23919/ACC.2018.8431212

[2] Daniel Silvestre, "OPTool—An optimization toolbox for iterative algorithms, ", *SoftwareX*, Volume 11, 2020, pp. 100371. doi: 10.1016/j.softx.2019.100371