The PageRank Algorithm

Maximilien Danisch

Sorbonne Université

first_name.last_name@lip6.fr

Outline

Outline

1. Matrix/Vector Multiplication
   - Reminder
   - Algorithms
2. Top Eigenvectors
   - Definition
   - Power iteration method
3. PageRank
   - What is PageRank?
   - Computation using the power iteration method
   - Personalized PageRank

Matrix/Vector Multiplication

Given an n by n matrix M and a vector (n by 1 matrix) A, we want to compute the vector (n by 1 matrix) B such that:

$$B = M \times A$$

where $\forall i \in [\![1, n]\!]$, $B_i = \sum_{j=1}^{n} M_{ij} \times A_j$.

Classic matrix/vector multiplication

Algorithm 1 Classic matrix/vector multiplication

function PRODMATVECT(M, A)
  for i from 1 to n **do**
    $B[i] \leftarrow 0$
    for j from 1 to n **do**
      $B[i] += M[i][j] \times A[j]$
  return B

Question: Datastructure? Memory consumption? Running time?

Classic sparse matrix/vector multiplication

Algorithm 2 Classic sparse matrix/vector multiplication

function ProdMatVect(M, A)
  for i from 1 to n **do**
    $B[i] \leftarrow 0$
    for j from 1 to n such that $M[i][j] \neq 0$ **do**
      $B[i] {+}{=} M[i][j] \times A[j]$
  return B

Question: Datastructure? Memory consumption? Running time?

Edge Iteration Version

Algorithm 3 Edge iteration sparse matrix/vector multiplication

function PRODMATVECT2(M, A)
  for i from 1 to n **do**
    $B[i] \leftarrow 0$
  for each (i, j) such that $M[i][j] \neq 0$ **do**
    $B[i] {+}{=} M[i][j] \times A[j]$
  return B

Question: Datastructure? Memory consumption? Running time?

Outline

Matrix/Vector Multiplication

Definition: given an n by n matrix M and a vector (n by 1 matrix) A, A is an eigenvector of M if it has at least one non-zero value and if there exists a scalar $\lambda \in \mathbb{R}$ such that

$$M \times A = \lambda \times A$$

Definition: a top eigenvector is an eigenvector A such that its associated eigenvalue λ has maximum absolute value.

Power iteration method

Algorithm 4 Power iteration method

function POWERITERATION(M, t)
  $A \leftarrow$ random vector
  for i from 1 to t **do**
    $A \leftarrow$ MATVECTPROD(M, A)
    $A \leftarrow$ NORMALIZE(A) $\triangleright \forall i \in [\![1, n]\!], A[i] \leftarrow \frac{A[i]}{||A||_1}$
  return A

In practice (under some conditions), POWERITERATION converges to a top eigenvector when $t \to +\infty$.

Question: Running time?

Additional content: top-k eigenvectors

Algorithm 5 Power iteration for top-k eigenvectors

```
function PowerIteration2(M, k, t)
    for l from 1 to k do
        A_l ← random vector
        for i from 1 to t do
            A_l ← MatVectProd(M, A_l)
            for j from 1 to l do
                v ← ScalarProd(A_l, A_j)            ▷ Scalar product
                A_l ← A_l − v × A_j
        A_l ← Normalize(A_l)
    return A_1, A_2, ..., A_k
```

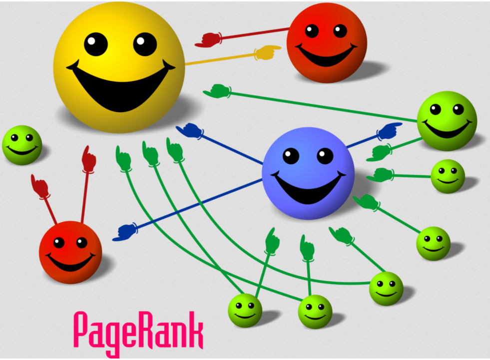
In practice (under some conditions), PowerIteration2 converges to top-k eigenvectors when $t \to +\infty$.

Question: Running time?

Outline

What is PageRank?

- PageRank is an algorithm used by Google Search to rank websites in their search engine results.
- It counts the number and quality of links to a page.
- The underlying assumption is that an important website is likely to receive links from other important websites.



Random walks

Pagerank is based on random walks:

- a walker starts at a node chosen uniformly at random
- it follows one of the out-links chosen uniformly at random
- go to 2

Score of a node v = probability to have the random walker on node v after an infinite number of steps.

Problem: what if the graph has a dead-end? has a spider trap? is cyclic?

Random walks

Pagerank is based on random walks:

1. a random walker starts at node *u*
2. it then teleports to a random node with probability α
3. if it does not teleport then:
   - it follows one of the outlinks chosen uniformly at random
   - if the node has no outlinks it teleports to a random node
4. go to 2

Analogy with a surfer on the web...

PageRank(v) $=$ probability to have the random walker on node v after an infinite number of steps.

The transition matrix of a graph

Definition: given a directed graph G with n nodes and m directed edges, its transition matrix T is define as follows.

- T is an n by n matrix with m non-zero values
- for each directed edge (u, v) in G, $T_{vu} = \frac{1}{d^{out}(u)}$

Definition: if G has no dead-end (nodes with $d^{out} = 0$), then the PageRank vector P is given by the following equation:

$$P_{t+1} = (1 - \alpha) \times T \times P_t + \alpha \times I$$

where I is the vector such that each entry equals to $\frac{1}{n}$.
Usually $0.1 \leq \alpha \leq 0.2$.

Question: P is the top eigenvector of which matrix?

The transition matrix of a graph

If G has dead-ends then the augmented transition matrix T' should be used instead of T:

- for each directed edge (u, v) in G, $T'_{vu} = \frac{1}{d^{out}(u)}$
- if $d^{out}(u) = 0$, then $\forall v$, $T'_{vu} = \frac{1}{n}$

Definition: the PageRank vector P is given by the following equation:

$$P_{t+1} = (1 - \alpha) \times T' \times P_t + \alpha \times I$$

where I is the vector such that each entry equals to $\frac{1}{n}$. Usually $0.1 \leq \alpha \leq 0.2$.

Question: What if the graph has many dead-ends?

Power iteration

$$P_{t+1} = (1-\alpha) \times T' \times P_t + \alpha \times I$$

Algorithm 6 Power iteration to compute PageRank

function $\textsc{PowerIteration}(G, \alpha, t)$
  $T \leftarrow$ transition matrix of graph G
  $P \leftarrow \frac{1}{n} \times I$
  for i from 1 to t **do**
    $P \leftarrow \textsc{MatVectProd}(T, P)$
    $P \leftarrow (1-\alpha) \times P + \alpha \times I$
    $P \leftarrow \textsc{Normalize2}(P)$ $\triangleright\ \forall i \in [\![1, n]\!],\ P[i] \mathrel{+}= \frac{1-||P||_1}{n}$
  return P

Personalized PageRank

$$P_{k+1} = (1-\alpha) \times T'_{P_0} \times P_k + \alpha \times P_0$$

Algorithm 7 Power iteration to compute rooted PageRank

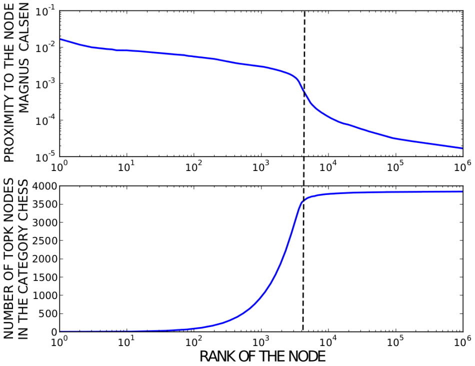
function $\textsc{PowerIteration}(G, P_0, \alpha, t)$
  $T \leftarrow$ transition matrix of graph G
  $P \leftarrow \frac{1}{n} \times I$
  for i from 1 to t **do**
    $P \leftarrow \textsc{MatVectProd}(T, P)$
    $P \leftarrow (1-\alpha) \times P + \alpha \times P_0$
    $P \leftarrow \textsc{Normalize2}(P)$ $\triangleright$ $\forall i \in [\![1, n]\!], P[i] {+}{=} P_0[i]\frac{1-||P||_1}{n}$
  return P

Definition: the *Rooted PageRank in u* is the personalized PageRank such that $P_0[u] = 1$.

Rooted PageRank as a “proximity metric”

- The distance may not be a good “proximity metric”. Why?
- Rooted Pagerank might be better.

Experiments in the Wikipedia network:



https://tel.archives-ouvertes.fr/tel-01207046

Additional content: the push method

There are more efficient ways to compute an approximation of the rooted PageRank:

Local graph partitioning using pagerank vectors
R. Andersen and F. Chung and K. Lang
FOCS06

`http://www.cs.cmu.edu/afs/cs/user/glmiller/public/Scientific-Computing/F-11/RelatedWork/local_partitioning_full.pdf`