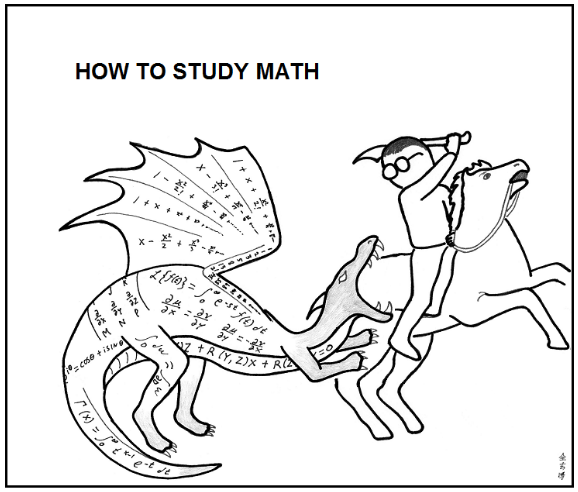
Notes: Pagerank III



Don't just read it; fight it!

--- Paul R. Halmos

1 Alternative Motivations

Pagerank can be used in many situations besides web ranking. In particular, if you can formalize your problem as somehow finding the "most important" nodes in a graph where something important is "flowing" between those nodes, then you can use pagerank.

Problem 1. Describe how pagerank can be used for the following tasks.

1. You are in charge of advertising the new Apple iGadget. You want to deliver free samples to social media influencers, with the hope that they will write good reviews that cause many people to purchase the iGadget. How do you select which people to give these free samples to?

2. You are the director of the CIA and you want to kill terrorists. How do you identify likely terrorists?

NOTE: Former CIA director Michael Hayden famously said "We kill people based on metadata." This is exactly what he was referring to. See:

`https://www.justsecurity.org/10318/video-clip-director-nsa-cia-we-kill-people-based-metadata/`

Problem 2. Create an example problem statement similar to the problem statements from Problem 1 above, and describe how the problem can be solved using pagerank.

Problem 3. Describe a graph problem where it would NOT make sense to use pagerank.

2 Alternative Algorithms

Problem 4. There are many algorithms for computing top eigenvectors, and any of these algorithms can be used to compute the pagerank vector. In this problem, we will investigate the *exponentially accelerated power method* (EAPM). This is a divide and conquer algorithm that can achieve the same accuracy ϵ as the power method with only a logarithmic number of iterations.

Recall that in the standard power method, at each iteration we perform the computation in Eq (5.1) from the text. It is reproduced below:

$$\mathbf{x}^{(k)T} = \mathbf{x}^{(k-1)T}\bar{\bar{\mathbf{P}}}. \tag{1}$$

Unraveling the recursion, we have that

$$\mathbf{x}^{(k)T} = \mathbf{x}^{(0)T}\bar{\bar{\mathbf{P}}}^{k}. \tag{2}$$

In the standard power method, we solve Eq (2) with a procedure involving k matrix-vector operations. The key idea of the EAPM is to "reparenthesize" these operations to instead solve $\log k$ matrix-matrix operations.

The output of the EAPM after K iterations is the vector $\mathbf{y}^{(K)}$, which is defined as

$$\mathbf{y}^{(K)} = \frac{\mathbf{x}^{(0)}\mathbf{Q}_K}{\|\mathbf{x}^{(0)}\mathbf{Q}_K\|_2}, \tag{3}$$

where

$$\mathbf{Q}_k = \begin{cases} \bar{\bar{\mathbf{P}}} & \text{if } k = 0 \\ \mathbf{Q}_{k-1}\mathbf{Q}_{k-1} & \text{otherwise} \end{cases}. \tag{4}$$

In the standard power method, the matrix $\bar{\bar{\mathbf{P}}}$ is not stored explicitly; we substitute the definition of $\bar{\bar{\mathbf{P}}}$ into Eq (1) above and perform all calculations on the sparse $\mathbf{P}$ matrix. In the EAPM, the $\bar{\bar{\mathbf{P}}}$ matrix is stored explicitly as a dense matrix, and each $\mathbf{Q}_k$ is also stored as a dense matrix.

1. What is the runtime of calculating $\mathbf{Q}_k$ given $\mathbf{Q}_{k-1}$?

2. What is the runtime of computing $\mathbf{y}^{(K)}$ in terms of K?

3. As with the standard power method, we do not know the total number of iterations of the exponential power method in advance. Instead, we iterate until the answer is "good enough". For this problem, we will directly compare the distance between the Kth iterate to the unknown pagerank vector, and iterate until the following condition is satisfied:

$$\|\mathbf{y}^{(K)} - \boldsymbol{\pi}\|_2 \leq \epsilon, \tag{5}$$

where ϵ is a predetermined small constant value. The *Deeper Inside Pagerank* paper does not discuss the EAPM algorithm, so I have provided below a simple theorem (without proof) about its convergence rate.

Theorem 1. For all K, the EAPM satisfies

$$\|\mathbf{y}^{(k)} - \boldsymbol{\pi}\|_2 \leq \alpha^{2^k}. \tag{6}$$

Use Theorem 1 above to compute the number of iterations K required to achieve a target ϵ value.

4. What is the runtime of computing $\mathbf{y}^{(K)}$ in terms of ϵ?

3 Review Questions

Problem 5. For each statement below, circle `True` if the statement is known to be true, `False` if the statement is known to be false, and `Open` if the statement is not known to be either true or false.

1. `True` `False` `Open` Let $f(n) = 1/(1+n)$. Then $f = \Omega(n^{-2})$.

2. `True` `False` `Open` Let A and B be $n \times n$ matrices. The optimal algorithm for computing the matrix product AB has runtime $O(n^4)$.

3. `True` `False` `Open` Let A and B be $n \times n$ matrices. The optimal algorithm for computing the matrix product AB has runtime $\Theta(n^2 \log n)$.

4. `True` `False` `Open` Let A and B be $n \times n$ matrices. The optimal algorithm for computing the matrix product AB has runtime $\Omega(n^{2.1})$.

5. `True` `False` `Open` Let A and B be $n \times n$ matrices. The optimal algorithm for computing the matrix product AB has runtime $\Omega(n^2)$.

6. `True` `False` `Open` Let A and B be $n \times n$ matrices. The optimal algorithm for computing the matrix product AB has runtime $\Omega(n^{2.5})$.

7. `True` `False` `Open` Let A be an $n \times n$ matrix and k a natural number. Then the matrix exponential A^k can be computed in time $O(n^{2.807} \log k)$.

8. `True` `False` `Open` Let A be an $n \times n$ matrix and k a natural number. Then the matrix exponential A^k can be computed in time $O(n^2 \log n \log k)$.

9. `True` `False` `Open` The matrix chain ordering problem can be solved in time $\theta(n)$.

10. `True` `False` `Open` Computing $\mathbf{x}^T\mathbf{x}$ is faster than computing $\mathbf{x}\mathbf{x}^T$.

11. `True` `False` `Open` There exist $n \times n$ matrices A and B with $\text{nnz}(A) = O(1)$ and $\text{nnz}(B) = O(1)$ such that the product AB satisfies $\text{nnz}(AB) = O(1)$.

12. `True` `False` `Open` For all $n \times n$ matrices A and B with $\text{nnz}(A) = O(1)$ and $\text{nnz}(B) = O(1)$, the product AB must satisfy $\text{nnz}(AB) = O(1)$.

13. `True` `False` `Open` There exist $n \times n$ matrices A and B with $\text{nnz}(A) = O(1)$ and $\text{nnz}(B) = O(1)$ such that the product AB satisfies $\text{nnz}(AB) = \Omega(n)$.

14. `True` `False` `Open` There exists an $n \times n$ matrix A such that $\text{nnz}(A) = \Omega(n^3)$.

15. `True` `False` `Open` There exists an $n \times n$ matrix A such that $\text{nnz}(A) = O(n^3)$.

16. `True` `False` `Open` Let A be an $n \times n$ matrix that is both primitive and stochastic. Then A has exactly one eigenvalue equal to 1.

17. `True` `False` `Open` The pagerank vector $\boldsymbol{\pi}$ can never have a 0 entry.

18. `True` `False` `Open` The $\mathbf{P}$ matrix is primitive.

19. `True` `False` `Open` The $\bar{\bar{\mathbf{P}}}$ matrix is stochastic.

20. `True` `False` `Open` For all graphs, $\text{nnz}(\bar{\bar{\mathbf{P}}}) = 0$.

21. `True` `False` `Open` For all graphs, the node with the largest in degree must have the highest pagerank.

22. `True` `False` `Open` According to the *Deeper Inside Pagerank* paper, the $\mathbf{P}$ matrix constructed from the web graph satisfies $\text{nnz}(\mathbf{P}) = O(1)$.

23. `True` `False` `Open` According to the *Deeper Inside Pagerank* paper, Google uses a value of $\alpha = 0.8$ when computing pagerank.

24. `True` `False` `Open` The L1 norm of the pagerank vector must be equal to 1. That is, $\|\boldsymbol{\pi}\|_1 = 1$.

25. `True` `False` `Open` The exponentially accelerated power method is faster than the standard power method for computing the pagerank of sparse graphs with many nodes.

26. `True` `False` `Open` If the power method is taking too long to converge, then increasing the α hyperparameter will make the algorithm run faster.

27. `True` `False` `Open` The number of iterations of the standard pagerank algorithm depends on the choice of personalization vector $\mathbf{v}$.

28. `True` `False` `Open` Decreasing the α hyperparameter increases the effect of the personalization vector $\mathbf{v}$ on the computed pagerank.

29. `True` `False` `Open` Decreasing the α hyperparameter will decrease the pagerank of the node with the highest pagerank.

30. `True` `False` `Open` Decreasing the α hyperparameter will increase the pagerank of the node with the lowest pagerank.

31. `True` `False` `Open` Increasing α decreases the subdominant eigenvalue of the $\bar{\bar{\mathbf{P}}}$ matrix.

32. `True` `False` `Open` Increasing α is guaranteed to make power method require more iterations to converge to the same residual ϵ.

33. `True` `False` `Open` As the number of nodes n in the graph increases (i.e. the dimensions of the $\mathbf{P}$ matrix increase), the number of iterations required by the power method will also increase.

34. `True` `False` `Open` The runtime of a single iteration of the power method using Equation (5.1) increases as α increases.

35. `True` `False` `Open` The runtime of a single iteration of the exponentially accelerated power method using Equation (5.1) increases as ϵ decreases.

36. `True` `False` `Open` If n is small, $\mathbf{P}$ is dense, and you require an extremely accurate estimate of the pagerank, then the exponentially accelerated power method will likely be faster than the standard power method.