CS 361, Lecture 29

Jared Saia
University of New Mexico

Outline

- Clever and Google
- Final Exam Review
- Course Evaluations

The *Clever* Approach

- Key observation: Certain pages point mainly to sites that are authoritative.
- These pages are called good *Hubs*
- Examples:
  - www.yahoo.com
  - "Jared's Big Page Of Links To Cool Sites"
- Question: But then how do we define what makes a page a good Hub?
- The *Clever* algorithm and Google solve this problem

The *Clever* Approach

- We use a mutually-recursive definition of the notion of Authoritative and Hub-like
- Good *Hub* pages point to many good *Authoritative* pages
- Good *Authoritative* pages are pointed to by many good *Hub* pages

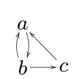
The Clever Algorithm

High Level Idea:

- Clever collects all pages containing the search term, or pages that are neighbors of pages containing the search term.
- It then iteratively propogates "authority" scores and "hub" scores
  - All pages start with equal authority and hub scores. In each step:
  - The pages with hub weight confer authority weight to the pages they point to.
  - The pages with authority weight confer hub weight to the pages *that point to them*

Adjacency matrix

- Consider the following web graph over the web pages $\{a, b, c\}$:



- The Adjacency Matrix is:

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

- We can think of the Clever algorithm as operating on this adjacency matrix A

The Clever Algorithm

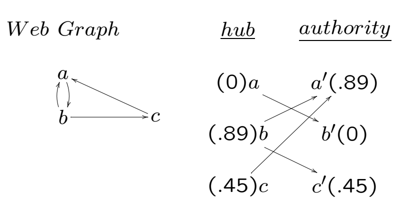
Algorithm CLEVER:
Input: Graph G (G has n nodes and adjacency matrix A)
Constants: $\epsilon > 0$, $t(= 10)$

1. $\vec{w}_h^0 \leftarrow$ length n all 1's vector, $\vec{w}_a^0 \leftarrow \vec{0}$, $i \leftarrow 0$
2. Repeat
   (a) $i \leftarrow i + 1$
   (b) $\vec{w}_a^i \leftarrow A^T \vec{w}_h^{i-1}$, $\vec{w}_h^i \leftarrow A \vec{w}_a^{i-1}$
   (c) $\vec{w}_h^i \leftarrow \vec{w}_h^i / |\vec{w}_h^i|$, $\vec{w}_a^i \leftarrow \vec{w}_a^i / |\vec{w}_a^i|$
3. Until $|\vec{w}_h^i - \vec{w}_h^{i-1}| \leq \epsilon$ and $|\vec{w}_a^i - \vec{w}_a^{i-1}| \leq \epsilon$
4. (Auth. nodes, Hub nodes) $\leftarrow$ top t weight nodes in $\vec{w}_h^i, \vec{w}_a^i$

What does this algorithm compute?

- If E is the set of edges in the link graph, the alg returns the vectors w_a and w_h such that $|w_a| = |w_h| = 1$ and $\sum_{(s,t) \in E} w_h[s] * w_a[t]$ is maximized
- This is the same as returning the vectors w_a and w_h such that $|w_a| = |w_h| = 1$ and $w_h^T A w_a$ is maximized.

Example Weight Assignment



Singular Value Decomposition(SVD)

$$A = \underbrace{\begin{bmatrix} | & & | \\ u_1 & \cdots & u_m \\ | & & | \end{bmatrix}}_{U:m\times m} \underbrace{\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{bmatrix}}_{D:m\times n} \underbrace{\begin{bmatrix} - & v_1 & - \\ & \vdots & \\ - & v_n & - \end{bmatrix}}_{V^T:n\times n}$$

If A is $m \times n$, $n \leq m$ with rank r then $A = UDV^T$ where U, V orthonormal, D diagonal with $\sigma_1, \ldots \sigma_r$ positive and nonincreasing and $\sigma_j = 0 \; \forall j > r$.

$\{u_i\}$ called left singular vectors, $\{v_i\}$ called right singular vectors, $\{\sigma_i\}$ called singular values.

SVD is unique up to multiplicities in singular values.

Crucial Fact

Let u_i be the i-th column of U. Then:

Fact 1 *For any matrix* A, $max_{|x|=1,|y|=1} x^T A y = \sigma_1(A)$ *and* $maxarg_{|x|=1,|y|=1} x^T A y = (u_1, v_1)$.

Link Analysis (Kleinberg, 1997)

Input: Web graph with adjacency matrix $A(= UDV^T)$ for pages relevant to a query.
Output: Vectors w_a, w_h giving "authority" and "hub" scores for all pages in the query set. If E is the set of edges in the link graph, gives $|w_a| = |w_h| = 1$ maximizing $\sum_{(s,t) \in E} w_h[s] * w_a[t]$ $(= w_h^T A w_a)$.

Algorithm: $w_h \leftarrow u_1$, $w_a \leftarrow v_1$
(this is same as limiting vectors of a weight updating algorithm).

Google

- Google does something similar to Clever but somewhat simpler.
- Google assigns pages "authorities" based on the first singular vector of a different adjacency matrix.
- This singular vector is related to the stationary probability distribution of a random walk on the web graph.
- Google computes the authority scores offline so it is much faster (but possibly less accurate)

Take Away

- Clever and Google solve a kind of graph optimization problem to rank web sites
- Linear Algebra is used to solve this optimization problem
- Cool example of CS Theory in the real world

Final Review Session

Review session times:

- Today at 7pm
- Friday at 1pm
- Monday at noon (feel free to bring a lunch)

The Final

- 7 questions, 20 points each
- Problems will be closely related to hw problems and in-class exercises
- There will be some time pressure, so make sure you can e.g. solve recurrences both quickly and correctly.
- I expect a class mean of between 80 :(and 100 :) points out of 140

Question 1

Collection of true/false, multiple choice, and matching questions on:

- Asymptotic notation (I ask you to find the proper asymptotic relationship between pairs of functions)
- Sorting algorithms (insertion, selection, mergesort, heapsort, bubblesort, bucketsort, lowerbounds)
- Heaps (height, number of nodes, heap algorithms and invariant, where is the max and max?)
- Search Trees (heights, number of nodes, algorithms and invariants, where is the max and min?)
- Hash Tables (algorithms, what is a good hash function, collision resolution)

Question 2

Collection of short answer questions on:

- Asymptotic notation (I give you a bunch of functions and ask you to give me the simplest possible theta notation for each)
- Deciding which data structure to use for certain types of problems
- Solving recurrence(s) with Master Theorem

Question 3

- A question on annihilators and recursion trees (like problems 1-3 of hw 3)
- You'll need to know the formula for sum of an infinite convergent series

Question 4

- A question on using annihilators to solve a recurrence with both homogeneous and non-homogeneous parts

Question 5

- A question on writing and solving a recurrence for the run time of a recursive function
- Similar to the Stoogesort problem in hw5

Question 6

- Question on search trees and/or skip lists
- Similar to Questions 2,3 and/or 4 on Cris Moore's CS 361 final from last semester

Question 7

- Proving the correctness of an algorithm using loop invariants
- You may need to give both the loop invariant and the proof for initialization, maintenance and termination

Course Evaluations

Instructor Option Questions:

- E) Did you find the class project beneficial?
- F) What would you suggest to improve the project portion of the course?