

## CS 276B Winter 2004-2005 midterm solutions

### 1. Pagerank

What we were looking for:

We're adding a link from A to B. Now suppose that there was a page C such that A points to C, which in turn points to B, giving B relatively high pagerank from A through C to B. The introduction of the new link from A to B results in less pagerank flowing from A to C, and in turn into B through C. Thus, the extra contribution to B's pagerank from the new link out of A may be swamped by the diminished contribution from C to B. Hence, it's not obvious in general if the pagerank of B is monotone in links into B.

Other answers that received varying degrees of partial credit:

- A has no in-links or is unpopular and so its link to B won't help B
- A is a duplicate of B and the link will be ignored
- A has lots of out-links already so the new one to B will hardly ever get followed

### 2. Pattern-based extraction

There's a variety of possible patterns (and ways of extending RAPIER to have two fillers) but you want roughly the below. Here, you certainly want to be extracting two slots ("fillers"), preferably jointly, though we allowed to separate patterns that matched one slot each providing each was selective enough. Below, we count the comma as a token in the `list:length(2)`. If you ignore punctuation, `list:length(1)` would be sufficient. Note that according to the Rapiere syntax, there can be a list of fillers, but an element of that list that is a "list" element cannot have further internal structure

HEADQUARTERS RELN

Company Filler	inter-fillers	Location filler
1) class: company	1) list: length 2	1) class: location
	2) word: {headquartered, headquarters}	
	3) word: in	

We gave progressively less partial credit as people only had one filler, didn't make the pattern specific enough (e.g., no "headquarters" constraint or company pattern could match location), etc.

### 3. Link analysis

Scoring: 10 points for pagerank and 10 points for hub/authority. Partial credit for correct ordering (about 50%) and additional partial credit for values that were close but not exactly right.

Power method works, but the below is simpler and takes a only few lines of work.

Pagerank:

Since the indegree of A is 0, the rank of A is  $.1 * 1 / 3 = .033$  (from teleport).

By symmetry and uniqueness of the solution,  $B = C$ .

Since  $A + B + C = 1$ ,  $B = C = 29/60$ .

Hubs and Authorities:

Authority: There is no concept of teleport, so  $A = 0$ ,  $B = C$ , so  $A = 0$ ,  $B = C = 1$ .

Hub: Use the Authority scores above, and iterate once to get  $A = 2$ ,  $B = 1$ ,  $C = 1$ .

Normalized,  $A = 1$ ,  $B = 1/2$ ,  $C = 1/2$

#### 4. Parallel crawling

5 points each. Partial credit of 2 points for answers that were arguably correct but missed the key point.

a) Why does it hurt to have more c-procs? “This trend is because the number of inter-partition links increases as the Web is split into smaller partitions, and thus more pages are reachable only through inter-partition links.” What explains the relatively modest improvement in coverage as the number of seeds grows, following the initial jump? Once you have a certain ratio between c-procs and seeds, the limiting factor is the inability to follow inter-partition links.

b) Need more c-procs (and thus more bandwidth) and coverage decreases.

c)  $(N-I)/I = 2.5 \rightarrow N = 3.5I \rightarrow$  each page is downloaded an average of 3.5 times

d) In site hash we’re exchanging far fewer links per page (0.5 versus 5) because most of the links on a given page point to other pages on the same site (and thus hash to the same partition and don’t need to be exchanged).

e) Impact is higher on sender and receiver system resources – URL exchange only takes about 0.4% of network bandwidth, but each passed message causes two context switches between kernel and user mode.

f) “This result is because a popular URL appears multiple times between backlink exchanges. Therefore, a popular URL can be transferred as *one* entry (URL and its backlink count) in the exchange, even if it appeared multiple times. This reduction increases as c-proc’s exchange backlink messages less frequently.”

#### 5. Named entity recognition

a) To solve this, you could work out a full Viterbi algorithm matrix for the data, but given that you only need to know about peripheral elements, you can considerably short circuit the computation. (Note however that for the initial word, you *do* need to look beyond it, though, because future transition and emission probabilities do influence the Viterbi state sequence.

For the first occurrence, the most taggings are E S O and O O O (and nothing further can effect things, since you’re O at time 3 in both cases).

$$P(E S O) = (0.95-y)*1*1*0.01*1*0.1$$

$$P(O O O) = y*0.1*(1-x)*0.1*(1-x)*0.1$$

So, to get E S O rather than O O O one needs  $0.95-y > y(1-x)^2$

Another possible tagging is O O P E. To avoid that you need  $0.95 - y > 2 y x (1-x)$ .

For the second occurrence, the two plausible final tag sequences are O P E and O O O. Starting from the transition to the second state given

$$P(P E|O) = x*0.2$$

$$P(O O|O) = (1-x)*0.1*(1-x)*0.1$$

So, get O P E if  $x > 0.05*(1-x)^2$

b) Why? The model can begin in the entity state and so can make the first word an entity, but it cannot then make the second word an (you can't tag successive things E). (It could also just make the second word an entity.)

To get the correct effect, moving from E to E must be possible, and starting this way must be preferred. The following HMM (among many others) will work:

A	P	S	O	E	Π	
P	0	0	0	1.0	P	0.05
S	0	0	1.0	0	S	0
O	0.1	0	0.9	0	O	0.6
E	0	0.9	0	0.1	E	0.4

$$P(O O O) = 0.6*0.1*0.9*0.1*0.9*0.01 = 0.0000486$$

$$P(E S O) = 0.4*1.0*0.9*0.01*1.0*0.01 = 0.000036$$

$$P(E E S) = 0.4*1.0*0.1*1.0*0.9*0.01 = 0.00036$$

The third tagging is correctly chosen.

Part a) was worth 20 points and part b) 10 points, partial credit was given for having the general idea of the constraints needed but getting actual arithmetic wrong.