UCT CSC3003 2006 :: XML/IR :: Supp Exam [25 marks]

Answer Questions 1 AND 2.

Then answer 3 questions from among Questions 3-6.

(Questions 3 and 4 are optional – Sonia will provide the other questions).

Question 2 : Boolean Information Retrieval [10]

Consider the following collection of documents:

DocumentID	Terms
Doc1	he sells sea shells sea
Doc2	he shells sea
Doc3	sea shells sea
Doc4	shells sells

1. Build inverted files for this document collection. Include a term occurrence count with each DocumentID listed. [4]
2. If the query "sea shells" is submitted to a Boolean-AND-based search engine, which results will remain after filtering? [1]
3. Using the following ranking formula, compute a ranking value for each result from the previous question. [3]

$$Similarity = \frac{1}{|D|} \sum_{t \in Q \cap D} (1 + \log_e f_{d,t}) . \log_e \left(1 + \frac{N}{f_t}\right)$$

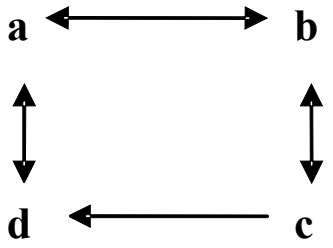
Assume that:

- D is the length of the document, including all terms in the document.
- Only terms common to both query and document and considered.
- N = the total number of documents in the result set.
- $f_{d,t}$ = term frequency of term t in document d
- f_t = number of documents term t appears in.

4. Briefly discuss two techniques that can be used to improve on precision. [2]

Question 3 : PageRank [10]

1. Apply the PageRank algorithm to the following link graph [5].



Start with equal ranks of 1/4 each. Stop after 3 iterations, not including the initial values. Show all calculations, including N(umber of forward links) values and B(ack links) sets.

(Hint: Remember that $r[i]_n = \sum_{j \in B[i]} \frac{r[j]_{n-1}}{N[j]}$)

3. Why do we need to remove sinks and leaks from the link graph? [2]
4. Why is PageRank not applicable to a collection of letters? [1]
5. Contrast the runtime performance of simple PageRank with simple HITS. [2]

Question 4 : XML / XSLT [10]

1. Write an XSLT template to transform

```
<basket>
   <number>123</number>
   <fruit>
      <name>apples</name>
      <type>granny smith</type>
   </fruit>
   <fruit>
      <name>grapes</name>
      <type>white </type>
   </fruit>
</basket>
```

into:

```
<basket>
   <number>123</number>
   <fruit>granny smith apples</fruit>
   <fruit>white grapes</fruit>
</basket>
```

Assume that the values such as “granny smith” may differ from one document to another. Assume the source namespace prefix is “source” and the destination prefix is “dest”. Assume that the number of <fruit> tags is unbounded in its defining XML Schema. Use the following as a starting point. [5]

```
<xslt:template match=”source:basket“>

. . .

</xslt:template>
```

2. Write an XML Schema type definition corresponding to the contents of the basket node in the source document. [5]