**Information Retrieval (CS60092)**

**Computer Science and Engineering, Indian Institute of Technology Kharagpur**

**Session: Autumn 2012 – 2013**

**Class Test 1**

**Time:** 1 hour

**Full Marks:** 20

*Attempt all questions.*

*Use of calculator is allowed.*

*State any assumptions made clearly.*

**Q. 1>** For the document collection:

*D*1*: catholic church in brisbane*

*D*2*: garden city church brisbane*

*D*3*: brisbane courier garden city*

*D*4*: where in brisbane catholic church*

**a.** Draw the term-document incidence matrix.

**b.** Draw the inverted index that would be built. **(1 + 1 = 2)**

**A. 1> a.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *D*1 | *D*2 | *D*3 | *D*4 |
| *catholic* | 1 | 0 | 0 | 1 |
| *church* | 1 | 1 | 0 | 1 |
| *in* | 1 | 0 | 0 | 1 |
| *brisbane* | 1 | 1 | 1 | 1 |
| *garden* | 0 | 1 | 1 | 0 |
| *city* | 0 | 1 | 1 | 0 |
| *courier* | 0 | 0 | 1 | 0 |
| *where* | 0 | 0 | 0 | 1 |

**A. 1> b.**

*catholic*

1

4

*church*

1

2

4

*in*

1

4

*brisbane*

1

2

3

4

*garden*

2

3

*city*

2

3

*courier*

3

*where*

4

**Q. 2>** What would be the best query processing order for the Boolean queries below, given the following term postings size:

*poison* 4133

*blue* 97002

*dart* 1079

*life* 27145

*frog* 466

*cycle* 3162

**a.** (poison OR blue) AND (dart OR frog) AND (life OR cycle)

**b.** (cycle OR blue) AND (poison OR frog) AND (dart OR life) **(1 + 1 = 2)**

**A. 2> a.** (poison OR blue) -> 4133 + 97002 = 101135

(dart OR frog) -> 1079 + 466 = 1545

(life OR cycle) -> 27145 + 3162 = 30307

So the best order is

((dart OR frog) AND (life OR cycle)) AND (poison OR blue)

Any commutative other order is fine, like *(a and b) and c* is same as *c and (b and a).*

**A. 2> b.** (cycle OR blue) -> 3162 + 97002 = 100164

(poison OR frog) -> 4133 + 466 = 4599

(dart OR life) -> 1079 + 27145 = 28224

So the best order is

((poison OR frog) AND (dart OR life)) AND (cycle OR blue)

**Q. 3>** What would be the permuterm vocabulary for “cat”? **(1)**

**A. 3>** cat$, at$c, t$ca, $cat

**Q. 4>** What is the likely effect of (a) Stemming and (b) Lemmatization on

**(i)** Vocabulary size: Increase, Decrease, Unpredictable?

**(ii)** Precision: Increase, Decrease, Unpredictable?

**(iii)** Recall: Increase, Decrease, Unpredictable? **(3)**

**A. 4> (a)** Stemming

(i) Vocabulary: Decrease (*bring, bringing, brings* all become *bring*) (If someone specifically states we add stemmed terms to existing list, then Increase is fine, because sometimes stem is a new term, like *duplic*)

(ii) Precision: Unpredictable

(iii) Recall: Increase

**A. 4> (b)** Lemmatization:

(i) Vocabulary: Decrease (Same logic, actually the decrease is more here as even *brought* becomes *bring;* doesn’t matter wrt marks)

(ii) Precision: Unpredictable

(iii) Recall: Increase

**Q. 5>** Let the relevance of top ten documents (leftmost = Rank 1) retrieved for a query be:

R, NR, R, R, NR, R, NR, R, NR, NR

where R = relevant and NR = non-relevant.

For this list, plot the (i) Precision-Recall curve and (ii) Interpolated Precision-Recall curve. **(3 + 3 = 6)**

**A. 5> (i)**

|  |  |  |
| --- | --- | --- |
| Doc | Recall | Precision |
| R | 0.20 | 1.00 |
| NR | 0.20 | 0.50 |
| R | 0.40 | 0.67 |
| R | 0.60 | 0.75 |
| NR | 0.60 | 0.60 |
| R | 0.80 | 0.67 |
| NR | 0.80 | 0.57 |
| R | 1.00 | 0.63 |
| NR | 1.00 | 0.56 |
| NR | 1.00 | 0.50 |

**A. 5> (ii)**

|  |  |  |
| --- | --- | --- |
| Doc | Recall | Interpolated Precision |
| R | 0.20 | 1.00 |
| NR | 0.20 | 1.00 |
| R | 0.40 | 0.75 |
| R | 0.60 | 0.75 |
| NR | 0.60 | 0.75 |
| R | 0.80 | 0.67 |
| NR | 0.80 | 0.67 |
| R | 1.00 | 0.63 |
| NR | 1.00 | 0.63 |
| NR | 1.00 | 0.63 |

**Q. 4>** Let the top ten documents (leftmost = Rank 1) returned by an IR system for three queries be graded for relevance as (6-point relevance scale, 0-5):

*q*1: 5, 5, 3, 3, 5, 4, 2, 1, 0, 0

*q*2: 4, 3, 0, 2, 2, 1, 5, 5, 5, 5

*q*3: 4, 4, 5, 5, 5, 2, 1, 1, 1, 1

nDCG@10 = DCG@10/IDCG@10. DCG@*p* of a graded ranked list of *p* documents is given by

where *p* = 10 in this case, *reli* is the relevance rating of document at Rank *i*.

Assume IDCG@*p* = DCG@*p* for a list of *p* documents where each document has the maximum rating (5 in this case).

nDCG = Normalized Discounted Cumulated Gain

DCG = Discounted Cumulated Gain

IDCG = Ideal Discounted Cumulated Gain

Find the average nDCG@10 of the system for this result set. Show each step of the computation. **(6)**

**A. 4>**

15.288

4.079

15.063

IDCG for all = 22.718

Thus, nDCG for *q*1 = 15.288/22.718 = 0.673

nDCG for *q*2 = 14.079/22.718 = 0.620

nDCG for *q*3 = 15.063/22.718 = 0.663

Thus, average nDCG of system = (0.673 + 0.620 + 0.663)/3 = 0.652 **Ans.**