COMP6714 Assignment 1 18s2

Keshi Chen

z5142821

Q1

Extend the structure of posting into "paragraph" and "sentence" level, i.e. transfer end of sentences and end of paragraphs into tokens and then store them as inverted indices, then in the "merge" algorithm, we can compute the distance between two words, using galloping search to delete the non-matching (/k), or check if two words occur in the same sentence (/S) or paragraph (/P) to decide whether we can merge them.

Q2

(1) Let L be the length of the posting list, n be the number of skip pointers. Under given assumptions, in the worst case, the skip-pointer intersection algorithm contains two parts, one is that do sequential search on pointers, whose cost is $f_1(n) = n/2$, the other one is that do sequential search on target segment, whose cost is $f_2(n) = L/2n$. Hence the total cost of the algorithm is $f(n) = f_1(n) + f_2(n) = n/2 + L/2n$. In order to find the n corresponding to the minimum cost, we first let the derivation of f(n) to be 0, i.e. $f'(n) = 1/2 - L/2n^2 = 0$. Then we have $n = \sqrt{L}$.

Hence choosing $\sqrt{L}$ skip pointers has the best worst case performance under assumptions in the question-stem.

(2) The cost of binary search is log(N), where N is the number of elements.

For step 1, the cost is $f_1(n) = \log_2(n)$, where n is the number of skip pointers.

For step 2, the cost is $f_2(n) = \log_2(L/n)$, where L is the length of target segment.

Hence the total cost $f(n) = \log_2(n) + \log_2(L/n) = \log_2(L)$, thus the cost is only dependent on L which is a constant, so the number of skip pointers can be any value. However, in order to save memory and operation times, the best number of skip pointers should be 1.

(3) For step 1, the cost in binary search is $f_1(n) = \log_2(n)$.

For step 2, the cost in sequential search is $f_2(n) = L/2n$.

Hence the total cost $f(n) = \log_2(n) + L/2n$.

Let $f'(n) = 1/[n*\ln(2)] - L/2n^2 = 0$, we then have $n = L*\ln(2)/2$, leading to the minimum cost.

Hence the best number of skip pointers is $L*\ln(2)/2$.

Q3

(1) Substitute the formula with idf, k1, k3 and b, we have:

$\text{Score}(d, Q) \leq 6f(tf_{A,d}) + 2f(tf_{B,d}) + f(tf_{C,d})$, where $f(x) = 3x/(2+x)$.

If x → ∞, the function f(x) converges to 3, thus we can simply compute maxscores for the terms are 18, 6 and 3, respectively, without examining the posting list.

(2) Start from D1, score(D1) = 6f(1) + 2f(1) + f(1) = 9

score(D2) = 6 f(8) + 2 f(0) + f(2) = 15.9

So far, the top-2 documents list is {D2, D1}, thus we have τ′ = 9,

Since idf(C) + idf(B) + idf(A) = 1 + 2 + 6 = 9 ≤ τ′, remove postings belonging to C and B in min heap, the only remaining postings now belong to A, i.e. we only need to consider postings in A.

Hence the next document to be accessed is D5. score(D5) = 6f(3) + 2f(4) + f(2) = 16.3. Now the top-2 documents list is {D5, D2}, and τ′ = 15.9.

The next document to be accessed is D8. score(D8) = 6f(10) + f(1) = 16.

Now the the top-2 documents list is {D8, D5}

Now all postings in A's posting list have been visited. We can conclude that the top-2 documents are D8 and D5.

Hence, documents accessed for scoring by algorithm are D1, D2, D5 and D8.

Q4

(1) The precision: precision = P(relevant|retrieved) = 6/20 = 0.3

(2) The formula of F_1 score (with $\beta = 1$): F_1 = 2*precision*recall/(precision + recall),

where precision = 0.3, recall = 6/8 = 0.75.

Hence F_1 = 2*0.3*0.75/(0.3 + 0.75) = 0.4285

(3) By calculating the precision and recall with different top-K retrieved results from 1 to 20, we have following table:

K	1	2	3	4	5	6	7	8	9	10
Precision(%)	100	100	66.67	50	40	33.33	28.57	25	33.33	30
Recall(%)	12.5	25	25	25	25	25	25	25	37.5	37.5
K	11	12	13	14	15	16	17	18	19	20
Precision	36.36	33.33	30.77	28.57	33.33	31.25	29.41	27.78	26.32	30
Recall	50	50	50	50	62.5	62.5	62.5	62.5	62.5	75

From table, it can be concluded that uninterpolated precisions under 25% recall are 100%, 66.67%, 50%, 40%, 33.33%, 28.57% and 25%.

(4) Since 8 * 33% = 2.67, the number of relevant documents should be at least 3 to make the recall ≥ 33%. From the table, we know that when k ≥ 9, recall ≥ 33%.

And the maximum uninterpolated precision when k ≥ 9 is 36.36%, which is the interpolated precision at 33% recall.

(5) The relevant documents are retrieved at k = 1, 2, 9, 11, 15 and 20, thus MAP = (1/1 + 2/2 + 3/9 + 4/11 + 5/15 + 6/20)/8 = 0.4163.

(6) When the next two retrieved documents (i.e. the 21st and 22nd results) are relevant, there will be the largest possible MAP, which is (1/1 + 2/2 + 3/9 + 4/11 + 5/15 + 6/20 + 7/21 + 8/22)/8 = 0.5034.

(7) When the last two retrieved documents (i.e. the 9999th and 10000th results) are relevant, there will be the largest possible MAP, which is (1/1 + 2/2 + 3/9 + 4/11 + 5/15 + 6/20 + 7/9999 + 8/10000)/8 = 0.4165.

(8) By considering (6):

e_1 = |0.5034 – 0.4163| = 0.0871

By considering (7):

e_2 = |0.4165 – 0.4163| = 0.0002

Since $e_1 > e_2$, the largest error by calculating (5) instead of (6) and (7) can be 0.0871.