**Q1**

/k:​ within k words respectively, /S:​ in the same sentence. /P:​ same paragraph.

One document contains some paragraphs, one paragraph contains some sentences and one sentence contains some words. In the preprocessing period, we can transfer some specific punctuation (!?.) which indicates the end of a sentence into tokens to support /S. Also, we can transfer ‘/n’ which indicates the end of a paragraph into tokens to support /P. We can store them in the positional inverted index. Here is the modification:

We can record the position of each term as following format:

*Document\_id (paragraph\_id, sentence\_id, word\_id)*

where paragraph\_id is the paragraph serial number in the document, sentence\_id is the serial number of the sentence in one paragraph, and the word\_id is the word position in one sentence.

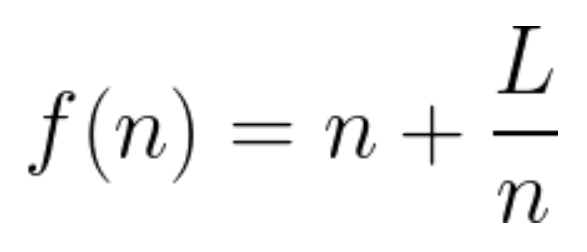
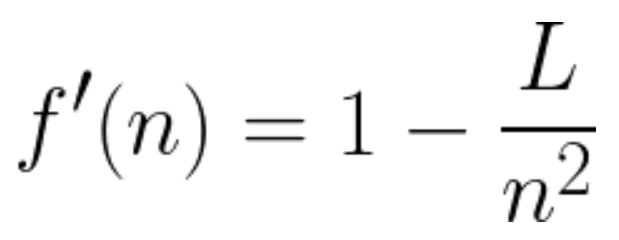
The positions which satisfy the /k operator can be detected by calculating the word\_id. So, this modification can support /k.

**Q2**

(1)

The worst time complexity of the sequential search is O(n)

Assumed that there are n pointers, and the length of each part is L/n. We firstly find the proper pointer in Step 1. It is a sequential search in n pointers, so the complexity is n. In Step 2, we have to find the target segment, and it is a sequential search in L/n elements. So, the complexity is L/n.

We can find the minimum number of pointers when f’(n)=0 that is .

So, choosing skip pointers has the best worst-case performance.

(2)

Assume that there are n pointers.

The time complexity of the binary search is O().

We still assume that there are n pointers and the length of each part is L/n.

In Step 1, the complexity is

In Step 2, the complexity is

So, the total complexity is which is equal to

is a constant. Thus, the number of skip pointers has nothing to do with the complexity. All numbers can achieve the same complexity when 0<n≤L

(3)

Assume that there are n pointers.

In Step 1, the complexity is

In Step 2, the complexity is

So, , then

When f’(n)=0, we have , then we have found the best number of skip pointers which is

**Q3.**

(1)

The scoring function can be written as

where for all A, B and C is 1, so

We can assume , the function is

When the maximum tf is 1, 10, 100, 1000 respectively, we can get the table as follow.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Max tf | 1 | 10 | 100 | 1000 |
| A | 6 | 15 | 17.65 | 17.96 |
| B | 2 | 5 | 5.88 | 5.99 |
| C | 1 | 2.5 | 2.94 | 2.99 |

If

So, and the maxscores for these terms are 18, 6, 3

(2)

From (1), we can rewrite the function into

First calculating D1 and D2, the results are score(D1)=9 and score(D2)=15.9, which are the top-2 results and τ’ = 9.

If the documents only contain B and C, the maximum score is less than 9=6+3 (the answer from (1)). So, we only need to calculate the documents which contain A.

Thus, the results of score(D5) and score(D8) are:

Score(D5)=16.3 and top-2 results are D5 and D2

Score(D8)=16 and top-2 results are D5 and D8

The postings that are accessed for scoring by this algorithm are:

A: D1, D2, D5, D8

B: D1, D5

C: D1, D2, D5, D8

Totally accessed 10 postings.

Q4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Precision | recall | No. | precision | recall |
| 1 | 1/1=100% | 1/8=12.5% | 11 | 4/11=36.36% | 4/8=50% |
| 2 | 2/2=100% | 2/8=25% | 12 | 4/12=33.33% | 4/8=50% |
| 3 | 2/3=66.67% | 2/8=25% | 13 | 4/13=30.77% | 4/8=50% |
| 4 | 2/4=50% | 2/8=25% | 14 | 4/14=28.57% | 4/8=50% |
| 5 | 2/5=40% | 2/8=25% | 15 | 5/15=33.33% | 5/8=62.5% |
| 6 | 2/6=33.33% | 2/8=25% | 16 | 5/16=31.25% | 5/8=62.5% |
| 7 | 2/7=28.57% | 2/8=25% | 17 | 5/17=29.41% | 5/8=62.5% |
| 8 | 2/8=25% | 2/8=25% | 18 | 5/18=27.78% | 5/8=62.5% |
| 9 | 3/9=33.33% | 3/8=37.5% | 19 | 5/19=26.32% | 5/8=62.5% |
| 10 | 3/10=30% | 3/8=37.5% | 20 | 6/20=30% | 6/8=75% |

(1) There are 6 relevant documents in 20 documents. So, the precision of the system on the top-20 is P@20=6/20=30%.

(2) There are 6 relevant documents in totally 8 relevant documents. So, R@20 = 6/8=75%

F1=2\*0.3\*0.75/(0.3+0.75)=0.4286

(3) The uninterpolated precisions of the system at 25% recall are: 100%, 66.67%, 50%, 40%, 33.33%, 28.57%, 25%

(4) When the interpolated precision is 33%, the document number should >= 9. So, the maximum precision is 4/11=36.36% at the 11th document.

(5) MAP is 1/6\*(1/1+2/2+3/9+4/11+5/15+6/20)=0.5551

(6) We can get the largest possible MAP when both the 21st and 22nd document are relevant.

So, the largest possible MAP is 1/8\*(1/1+2/2+3/9+4/11+5/15+6/20+7/21+8/22)=0.5034

(7) We can get the smallest possible MAP when both the 9999th and 10000th document are relevant.

The smallest possible MAP is 1/8\*(1/1+2/2+3/9+4/11+5/15+6/20+7/9999+8/10000)=0.4165

(8) (0.5034+0.4165)/2=0.45995

0.5551-0. 45995=0.0952

The maximum large is 0.0952