Report of Our Search Engine MileStone3 (DEV)

*The 20 test queries:

Bad query:

1. Computer – fast, irrelevant
2. Architecture – fast, irrelevant
3. Productivity – fast, irrelevant
4. Cristina Lopes – fast, irrelevant
5. mobile device – fast, irrelevant
6. artificial intelligence – fast, irrelevant
7. google doc – fast, irrelevant
8. online course – fast, irrelevant
9. computer game science – fast, irrelevant
10. master of software engineering – fast, irrelevant
11. a b c – fast, irrelevant (M2 run time error)

Good query (Defined as either runtime < 300ms or effectiveness):

1. ACM – fast, irrelevant
2. machine learning – fast, irrelevant
3. ablactate – fast, relevant
4. resource – fast, irrelevant
5. uci – fast, irrelevant
6. informatic – fast, irrelevant
7. ACM machine learning, fast, irrelevant
8. ACM machine learning uci, fast, irrelevant
9. go ACM machine learning uci, irrelevant

*Major improvement on runtime efficiency for queries that doing poorly we store the inverted index into multiple pickle files on disk.

When we read from these pickle files, we read line by line

so that the previous line in the memory will get removed when we read the second line.

*Evaluation on effectiveness/efficiency:

1. cristina lopes

    approx runtime(in sec) (M2: 0.8, M3: 0.01) -> efficiency: M3 way faster

    conclusion: effectiveness: irrelevant, efficiency: M3 better

2. a b c

    approx runtime(in sec) (M2: --, M3: 0.039) -> efficiency: M3 way faster

    conclusion: effectiveness: irrelevant, efficiency: M3 better

3. computer

    approx runtime(in sec) (M2: 0.6, M3: 0.05) -> efficiency: M3 way faster

    conclusion: effectiveness: irrelevant, efficiency: M3 better

4. Architecture

    approx runtime(in sec) (M2: 0.4, M3: 0.003) -> efficiency: M3 way faster

    conclusion: effectiveness: irrelevant, efficiency: M3 better

5. Productivity

    approx runtime(in sec) (M2: 0.4, M3: 0.016) -> efficiency: M3 way faster

    conclusion: effectiveness: irrelevant, efficiency: M3 better

6. ACM

    approx runtime(in sec) (M2: 0.2, M3: 0.007) -> efficiency: M3 way faster

    conclusion: effectiveness: (4/5)irrelevant, efficiency: good

7. machine learning

    approx runtime(in sec) (M2: 0.119, M3: 0.120) -> efficiency: M2 tie with M3

    conclusion: effectiveness: irrelevant, efficiency: similar(both good)

8. master of software engineering

    approx runtime(in sec) (M2: 0.65, M3: 0.055) -> efficiency: M3 way faster

    conclusion: effectiveness: irrelevant, efficiency: M3 better

9. mobile device

approx runtime(in sec) (M2: 2.1, M3: 0.032) -> efficiency: M3 way faster

conclusion: effectiveness: (4/5)irrelevant, efficiency: M3 better

10. artificial intelligence

approx runtime(in sec) (M2: 0.65, M3: 0.031) -> efficiency: M3 way faster

conclusion: effectiveness: (4/5)irrelevant, efficiency: M3 better

11. google doc

approx runtime(in sec) (M2: 0.4, M3: 0.008) -> efficiency: M3 way faster

conclusion: effectiveness: irrelevant, efficiency: M3 better

12. online course

approx runtime(in sec) (M2: 0.7, M3: 0.05) -> efficiency: M3 way faster

conclusion: effectiveness: irrelevant, efficiency: M3 better

13. computer game science

approx runtime(in sec) (M2: 0.65, M3: 0.04) -> efficiency: M3 way faster

conclusion: effectiveness: irrelevant, efficiency: M3 better

14. ablactate

approx runtime(in sec) (M2: 0.04, M3: 0.003) -> efficiency: M3 way faster

conclusion: effectiveness: relevant, efficiency: M3 better

15. resource

approx runtime(in sec) (M2: 0.28, M3: 0.04) -> efficiency: M3 way faster

conclusion: effectiveness: (4/5)irrelevant, efficiency: M3 better

16. uci

approx runtime(in sec) (M2: 0.14, M3: 0.006) -> efficiency: M3 way faster

conclusion: effectiveness: (3/5)irrelevant, efficiency: M3 better

17. informatic

approx runtime(in sec) (M2: 0.2, M3: 0.024) -> efficiency: M3 way faster

conclusion: effectiveness: irrelevant, efficiency: M3 better

18. ACM machine learning

approx runtime(in sec) (M2: 0.18, M3: 0.02) -> efficiency: M3 way faster

conclusion: effectiveness: irrelevant, efficiency: M3 better

19. ACM machine learning uci

approx runtime(in sec) (M2: 0.18, M3: 0.024) -> efficiency: M3 way faster

conclusion: effectiveness: irrelevant, efficiency: M3 better

20. go ACM machine learning uci

approx runtime(in sec) (M2: 0.27, M3: 0.022) -> efficiency: M3 way faster

conclusion: effectiveness: irrelevant, efficiency: M3 better

* Comparison of Ranking performance (effectiveness)

M2:

*Rank by term-frequnce(tf) only

*No word-stemming procedure

M3:

*Rank by tf-idf weighting that combined with tag scores

# w_(t,d) = (1+log(tf_(t,d))) x log(N/df_t)

#tag_score = {"p": 1, "b": 10, "strong": 15, "h6": 20, "h5": 25, "h4": 30, "h3": 35, "h2": 40, "h1": 45, "title": 50}

*Using Porter Stemming on both input queries and words in data file

*Conclusion: Base on the evaluation of 20 test queries,

improvements on scoring calculation lead to the huge difference of urls top results effectiveness. As the comparison of the searching result between M2 and M3, test queries such as "resource", "ACM", "uci", "mobile device", "artificial intelligence", have 1 or 2 urls are the same. However, for more than half of the test queries, the URLs of the searching result between M2 and M3 are irrelevant.

*Comparison of Runtime performance (efficiency < 300ms)

M2:

*To find the related token information(postings) of input words:

line by line file search in 27 text. files(a.txt, b.txt, c.txt, ... special.txt)

M3:

*To find the related token information(postings) of input words:

line by line file search in 27*27 pickle. files(aa.txt, ab.txt, ac.txt, ... special.txt)

*Conclusion:Base on the evaluation of 20 test queries,

improvements on data format and file search algorithm lead to the huge boost of runtime performance. As the comparison of the runtime result between M2 and M3, test queries, such as "resource", "ACM", "uci", "ablactate ", "informatic", "machine learning", ..., in M2 are slightly below 300ms. However, for all the test queries, the runtime results of M3 are all below 300ms, some are even below 100ms.