Readme

b07901082
徐緯丞

This is README file for Data Structure pa1
==========
Directory inside “program/” folder:

bin/   executable binary
input/   input data
output/ output result
src/   source C++ codes
==========
How to compile:

type “make” under “program/”
==========
How to run:

type”./bin/PageRank_SearchEngine” under “program”
==========
How to clean the result:

type “make clean” under “program/”
==========
Analysis of the time and space complexity:

1. input:
   number of pages: n = 501
   number of words in one page: m = 20
   number of words in list.txt: l
   number of words in dictionary(i.e. distinct words): s
2. space complexity:
   pageRelation: O(n^2)
   dictionary: O(n*m)
   C: O(n)
   cal_pageRank():
       pageRank: O(n)
       pageOrder: O(n)
       //???
   reverseIndex():
       everyWord: O(s)
   SearchEngine():
       pageOrder: O(n)
       page_reverseOrder: O(n)
       word_searched: O(l)
       checkPageExist: O(l*m)
       sort_Rank, and_sort_Rank, or_sort_Rank: O(n)

3. time complexity:
   pageRelation: O(n^2)
   dictionary: O(n*m)
   C: O(n^2)
   cal_pageRank():
       fout: O(n)
   reverseIndex():
       everyword: O(s)
       sort(): O(s^2)
       fout: O(s^2)
   searchEngine():
       pageOrder: O(n)
       page_reverseOrder: O(n)
       word_searched: O(l)
       checkPageExist: O(l*m)
       sort_Rank, and_sort_Rank, or_sort_Rank: O(n^2)
       fout: const.

==========
Data structure:

2D vector, vector of list<string>
==========
Algorithm:

1. build PageRelation, dictionary to store distinct words, and C
2. calculate pageRanks for 12 different arguments
3. build reverse index
4. input list.txt into search engines with 12 different arguments