Andrew Jalbert, Jameson Dy, & Leonid Olekh

Professor Yau

CSCI 230

21 May 2017

Project 4 Write Up

Program Design Explanation

We implemented a Hashtable, HashMap, and an ArrayList in this program. The HashTable was used to store the number of hits between each compared file, and it was stored only if the number of hits was passed the threshold. The Hashtable also stored the file names as keys so that we could print out the names associated with the hits. The HashMap was used to count the hits. This was done by generating an entire map for the first file (the file being compared) and then generating short hashcodes for the second file. The hashcodes were generated by storing the chunk into a string. This string was converted to a hash code. If the second file’s hashcode was in the map, then a hit is generated. The program was coded to only generate the first file’s hashmap once. Then, for each file compared, the program saved time by not generating a hashmap each iteration. Our program also saved space by storing one hashmap at a time. The ArrayList was used to sort and print the hits in decreasing order. This was done by generating a collection from the Hashtable, and then using a comparator to sort the stored values. The given class, DocumentReader, was modified to fit the comparing and sorting algorithms. To deal with the trade-off of space vs time, we saved short instances of each hashmap, and we only created a hashtable when the method to run a file set was called. By changing some variables from private members to temporary variables, we only allocated memory when needed. To save time, we used hashing based data structures so that storing was constant. But, we had to use a nested for loop to iterate through the file set. These nested loops caused a run time close to n2.

Program Results

Small Set

Enter file directory, length of chunks, and match threshold: small\_set 6 200

file1 file2 Collisions

----- ----- ----------

jrf1109.txt sra31.txt 1646

abf0704.txt edo26.txt 685

abf0704.txt abf70402.txt 541

catchmeifyoucan.txt tyc12.txt 398

bef1121.txt edo14.txt 393

catchmeifyoucan.txt hal10.txt 314

catchmeifyoucan.txt ecu201.txt 290

Time elapsed: 1.01

Medium Set

Enter file directory, length of chunks, and match threshold: medium\_set 6 200

file1 file2 Collisions

----- ----- ----------

catchmeifyoucan.txt tyc12.txt 398

catchmeifyoucan.txt hal10.txt 314

catchmeifyoucan.txt ecu201.txt 290

Time elapsed: 3.962

Large Set

Enter file directory, length of chunks, and match threshold: large\_set

6

200

file1 file2 Collisions

----- ----- ----------

hce209.txt smu0.txt 1828

bgt61.txt sra31.txt 1646

pko4.txt pko5.txt 1307

prz134.txt prz143.txt 1306

pko254.txt pko259.txt 1228

edo109.txt emt186.txt 1199

aky140.txt aky147.txt 1155

euz1.txt euz98.txt 1050

hte174.txt hte198.txt 1034

ecu259.txt ecu260.txt 845

pko0.txt pko24.txt 842

bah68.txt bah69.txt 827

aky152.txt aky156.txt 822

esv232.txt saq94.txt 818

ecu201.txt esv24.txt 816

bah228.txt erk14.txt 776

bgt157.txt bgt160.txt 765

bgt3.txt esv294.txt 759

saq54.txt saq65.txt 656

bah228.txt ehc182.txt 495

ehc182.txt erk14.txt 482

catchmeifyoucan.txt tyc12.txt 398

catchmeifyoucan.txt hal10.txt 314

catchmeifyoucan.txt ecu201.txt 290

catchmeifyoucan.txt esv24.txt 233

Time elapsed: 884.589

Docs50 & up

Program still running since the first computer was invented

Project Discussion

Overall, we enjoyed this project. However, we ran into some problems. The biggest problem we ran into was running time. For the small and medium set it took very little time. For the large set, and the additional sets it took extremely long. This was due to our algorithm. Since we did a nested for loop, the run time was proportional to n2. From this project, we learned how important runtime is. We also learned about memory management. Lastly, we learned how to improve a program’s runtime. But, we could not further improve our program’s runtime. We found this project to be a little difficult, but we enjoyed the challenge. For help, we discussed the project with classmates. We also used the online lecture slides and the book. No one was assigned a specific role. Each member contributed to the program just about equally. However, Andrew and Leo worked more on the comparison algorithm, and Jameson worked more on the storing, sorting, and printing method.