VARUN RAJAMUDI (002108570)

DIKSHA BHATIA (002175782)

SHREEKAR SS (001545668)

Program Structures & Algorithms

Fall 2021

Team Project

Task

Implementing MSD radix sort for Simplified Chinese characters. Additionally, completing a literature survey of relevant papers and comparing our method with Dual-pivot Quicksort, Huskysort, and LSD radix sort. Benchmarked the different sorting methods for above different inputs and plotted the graph. Wrote Unit test cases for our MSD Radix sort implementation.Wrote a paper which explains the work that we have done in some details and describes the work of at least two related technical papers.

Proposed Solution

MSD radix sort is one of the best sorting methods to sort strings or characters. Since we are sorting Chinese characters whose Unicodes are unknown, we are taking the help of a Pinyin translation of those Chinese characters. So we are first converting Chinese words to its corresponding Pinyin words and then we are sorting those Pinyin words using MSD radix sort and converting back to Chinese words using HashMap. We are comparing different sorting methods like Dual pivot, Husky and LSD Radix sort for 250k, 500k, 1M, 2M, 3M and 4M inputs.

Changes Made

BenchmarkAndSortChinese.java:

Path: edu.neu.coe.info6205.finalProject.BenchmarkAndSortChinese.java

- This is the new java class to benchmark and to fetch the sorted Chinese characters. Here we are calling convertChineseToPinyin function to get the Array of strings of pinyin output to the respective Chinese inputs.

- Using the above output then we are calling sort functions of MSD, LSD, Quicksort Dual Pivot and Pure Husky sort.
- After sorting the pinyin characters, we are calling convertPinyinToChinese function to get the corresponding Chinese name to the sorted Pinyin names.
- In the main method of this class, we are benchmarking the time using Benchmark_Timer class for MSD Radix sort, LSD Radix sort, Quicksort Dual Pivot and Pure Huskysort.(Note: The benchmark time includes fetching the Chinese inputs from the file, converting the Chinese names to pinyin, sorting the pinyin names and converting it back to Chinese array)

ChineseStringUtil.java

Path: : edu.neu.coe.info6205.util.ChineseStrinUtil.java

convertChineseToPinyin()

- Imported “sfuffledChinese.txt” file using File class.
- Reading the shuffledChinese.txt file using BufferReader class.
- Created the format object for HanyuPinYinOutputFormat which gives Pinyin output in lowercase, with tone mark and with Unicode.
- Reading the text file line by line and then converting the Chinese text to corresponding Chinese output using toHanyuPinyinStringArray using PinyinHelper class.
- Converted Pinyin output is added into the list of strings and then it is converted to Array of strings which is the return type of this function.
- The Pinyin output and the respective Chinese input is also added to the HashMap. Key is the Pinyin name and the Value is the Chinese name.

convertPinyinToChinese()

- This function is created to fetch the Chinese name from the HashMap after sorting the Pinyin characters. The return type of this function is Array of strings which return the sorted Chinese names.

writeResultFile()

- This function is used to write the sorted Chinese output names into the file.

MSDStringSort.java

Path: edu.neu.coe.info6205.sort.counting.MSDStringSort.java

- Added a for loop in sort() function to recursively sort for each character value.
- Updated the radix variable value to 478 as the largest Pinyin input size is 478.

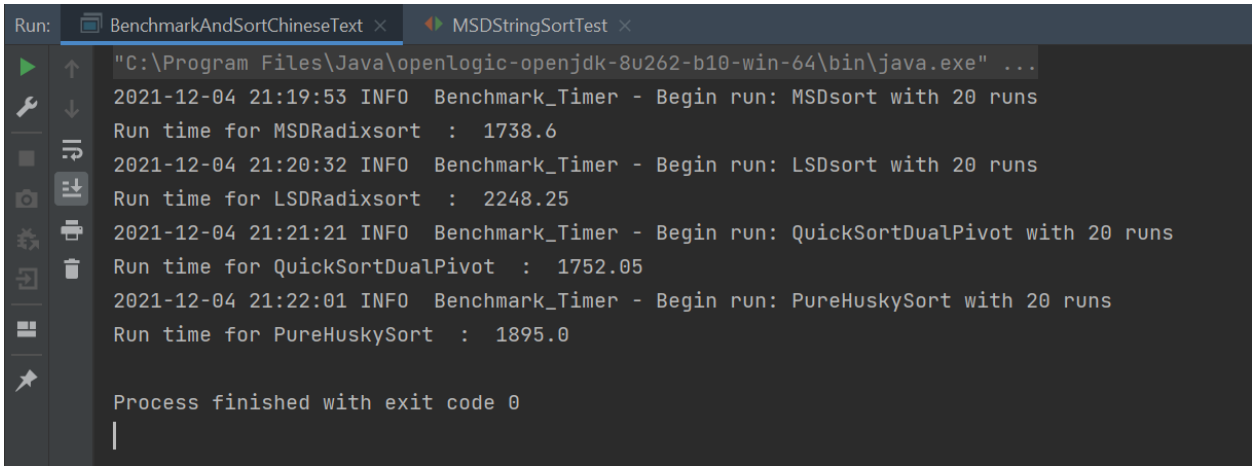
LSDStringSort.java

Path: edu.neu.coe.info6205.sort.counting.LSDStringSort.java

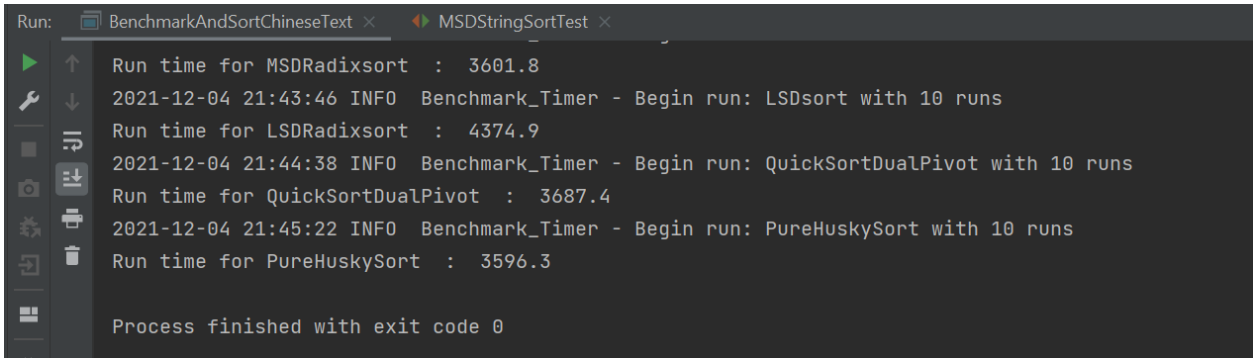
- Updated the radix variable value to 478 as the largest Pinyin input size is 478.

Output

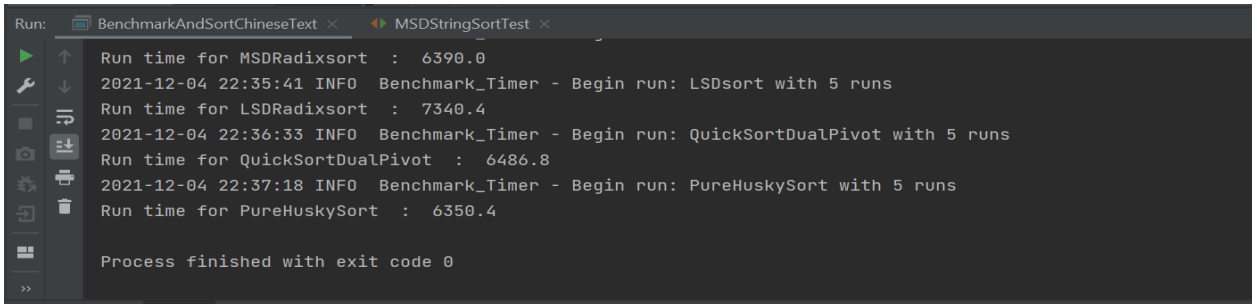
Input size = 250k, Number of runs = 20



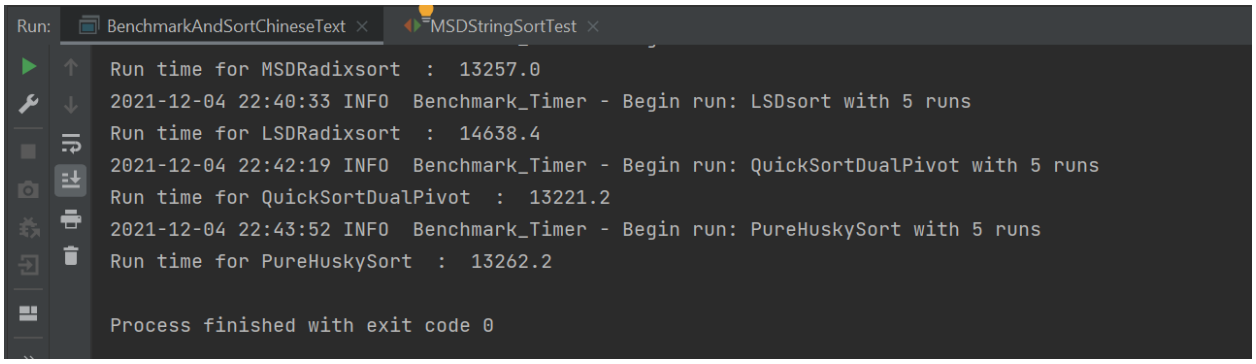
Input size = 500k, Number of runs = 10



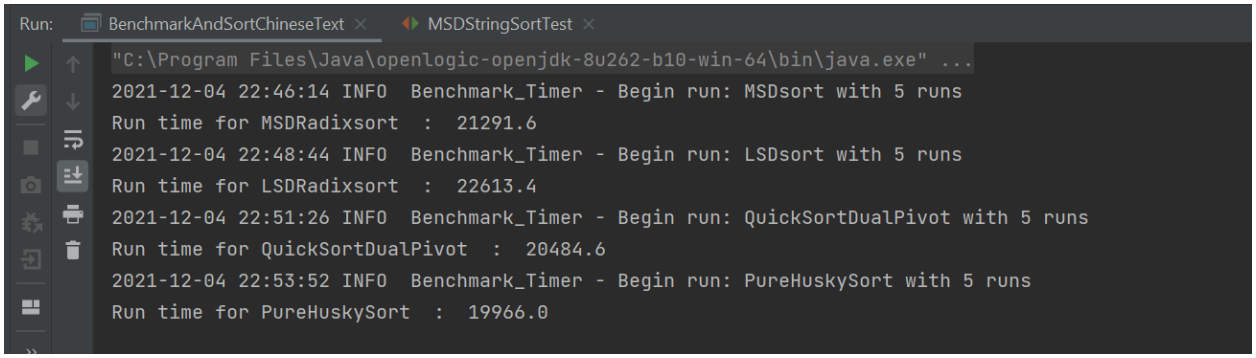
Input size = 1 Million, Number of runs = 5



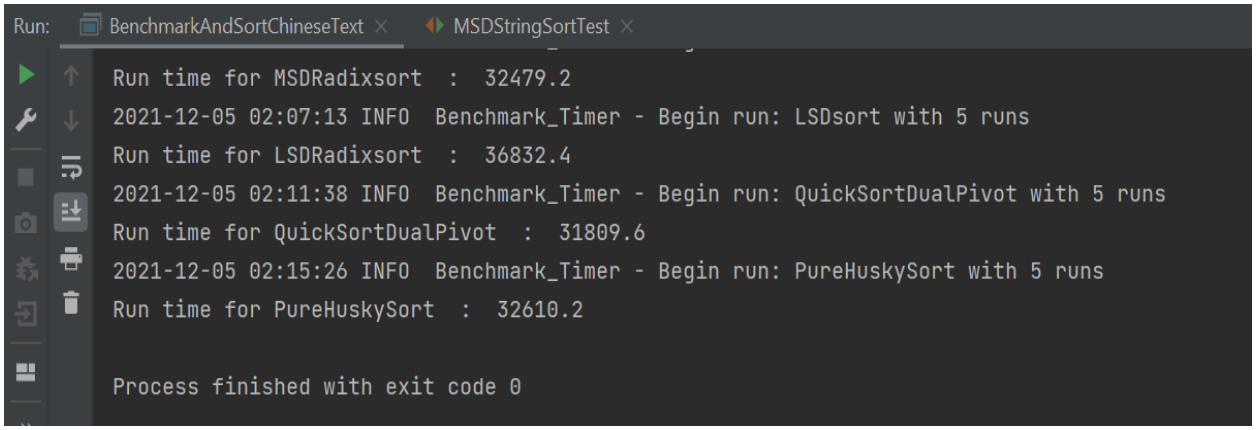
Input size = 2 Million, Number of runs = 5



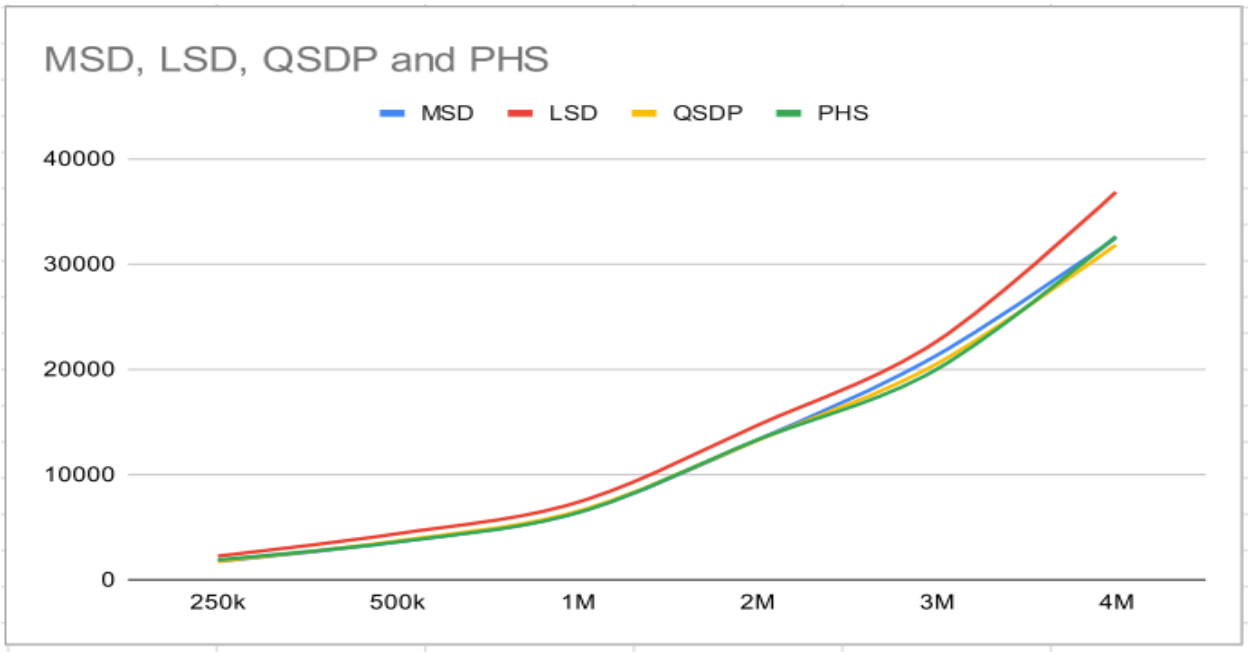
Input size = 3 Million, Number of runs = 5



Input size = 4 Million, Number of runs = 5

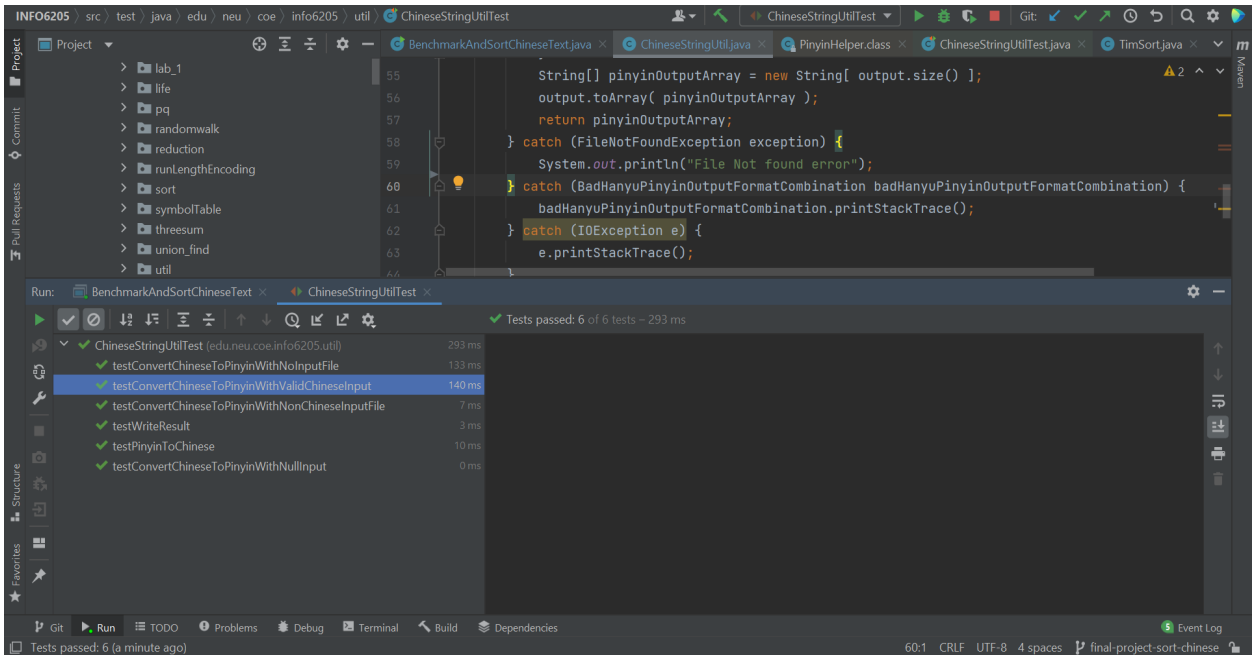


Graphical Representation



Unit tests result

We have written the test cases for all the files added to benchmark and conversion to pinyin from chinese characters.



Explanation of the test cases:

1. **testConvertChineseToPinyinWithValidChineseInput:** Asserts that tests output produces array of pinyin size of 4 with valid Input
2. **testConvertChineseToPinyinWithNullInput:** Asserts that tests that output produces array of pinyin size of 0 with empty text file
3. **testConvertChineseToPinyinWithNoInputFile:** Asserts that it throws File Not found exception when incorrect file input to the system
4. **testConvertChineseToPinyinWithNonChineseInputFile:** Assert that the logic would ignore any characters that were added as non chinese characters in the input
5. **testPinyinToChinese:** Assert that all pinyin values are converted to Chinese string

Conclusion

We tested different types of sorting algorithms with different output and we concluded that MSD Radix sort is one of the best suitable sorting algorithms for strings. Even though the space complexity is a little higher due to extra arrays, the cache misses is less compared to other sorting techniques. We strongly recommend using MSD Radix sort for array of string as it is faster and stable compared to other comparison based sorting.