INFO6205 Final project

Huanlin Xiao 001523208

Run Hou 001090054

## Abstract

This paper is demonstrating the details about implementing MSD radix sort especially for the Chinese words. Except for this kind of sorting algorithm, we have adapted several algorithms to fit in the Chinese words, like LSD radix sort, Quick sort, Tim sort and husky sort.

According to our benchmark data, we have found that conclusion Husky sort takes the shortest time.

Finally, we show related work that have been done by other ones try to optimize the performance of MSD radix sort.

## Introduction

Radix sorting is totally different with other sorting algorithms based on comparing. Its most common formats including MSD and LSD radix sort. However, there are still some variants, including Adaptive Radix sort and Forward Radix sort [1]. In their paper, they describe how to effectively implement a radix sort. For adaptive radix sort, they focus on how to decide the number of buckets, which also determine the spread width of every step. Normally, adaptive radix sort takes 8 bits or 16 bits as alphabet sizes to optimize the whole performance. Regarding Forward radix sort, it combines the features of MSD sorting and LSD sorting. It still sorts the most significant unit, but bucketing process is different from the MSD.

Actually, the adaptive radix sort reminds us not using bitwise bucketing.

## 1 Sorting method

We finally choose java.text.Collator class as our foundation for transferring the Chinese character into comparable Class. There are some methods we have thought below.

### Pinyin

Pinyin is a kind of method to transferring the Chinese words into Latin alphabet, which has been adapted by ISO. Basically, we can just use pinyin instead of the Chinese words to sort them. There are rules about how to sort different Chinese words.

1. Compare each alphabet like comparing English words
2. If all alphabets are the same, then compare their tones
3. If their tones are the same, then compare their numbers of strokes

Generally speaking, its process of comparing is complex and hard to distinguish, especially for people whose mother tongue are not Chinese.

### The collator

Java language support a class for compare local-sensitive String comparison, java.text.Collator, which simplify comparison between the Chinese words. By calling getCollationKey method, we can get the corresponding CollationKey object. Instead of comparing two Chinese characters, we can compare them only by their CollationKey object. Further, we just need to compare the byte array laying under the CollationKey object.

### 2 Implementation

图示

描述已自动生成

### Radix sort interface

The ComparableStartAt interface abstract the requirement of radix sorting algorithm. It can compare two objects starting from special index, so it should be implemented when this class want to be sorted by radix sort.

### The Chinese words representation

Considering the compatibility of different sorting the Chinese words algorithms, we use the ChineseWords class to represent the actual characters. Besides, it can be implemented by using either Pinyin or Collactor class without influencing availability.

### Benchmark

图形用户界面, 应用程序

描述已自动生成

For other algorithms which rely on the Comparable interface, we simply transfer sorting

items into our designed class and restore them after sorting is done. It’s mainly because we notice it not only make the process more standardized, but also make the result more accurate.

## Result

Pictures below are benchmark of the algorithms implementing comparing Chinese words.

MSD radix sort

电脑屏幕截图

描述已自动生成

LSD radix sort

电脑萤幕画面

描述已自动生成

Quick sort - Dual Pivot

文本

描述已自动生成

Tim Sort

电脑萤幕的截图

描述已自动生成

Husky Sort

电脑萤幕的截图

描述已自动生成

图表

低可信度描述已自动生成

According to the results of BenchMark, LSD radix sort takes the most time. Husky sort takes the shortest time. Quick sort - dual pivot, Tim sort, and MSD radix sort takes almost the same time. we would recommend husky sort algorithm to sort those Chinese String.

## 4 Flaws

### Collator converting problem

During the project, we find there are some problems that the Collator class have, including homophone. But we think the most difficult part is how to convert a byte array to a long value when using husky sort. Finally, considering time limited, we chose just to use the first 8 bytes to represent a long value and it obviously lower the accuracy of sorting.

## References

[1] Andersson, A., & Nilsson, S. (1998). Implementing radixsort. *Journal of Experimental Algorithmics (JEA)*, 3, 7-es.