MSD radix sort for Chinese

Rongjin Dang

Peixin Yao

Zixuan Zhu

Program Structure & Algorithms INFO 6205

2021.12.5

Northeastern University

Introduction

This project refers to the alphabetical sorting idea of Xinhua Dictionary, and finally decides to achieve the sorting of Chinese characters by sorting pinyin letters, tones and Unciode. This paper will give a comprehensive description and explanation of all the techniques and ideas needed to realize the algorithm as well as the algorithm steps(Figure1).

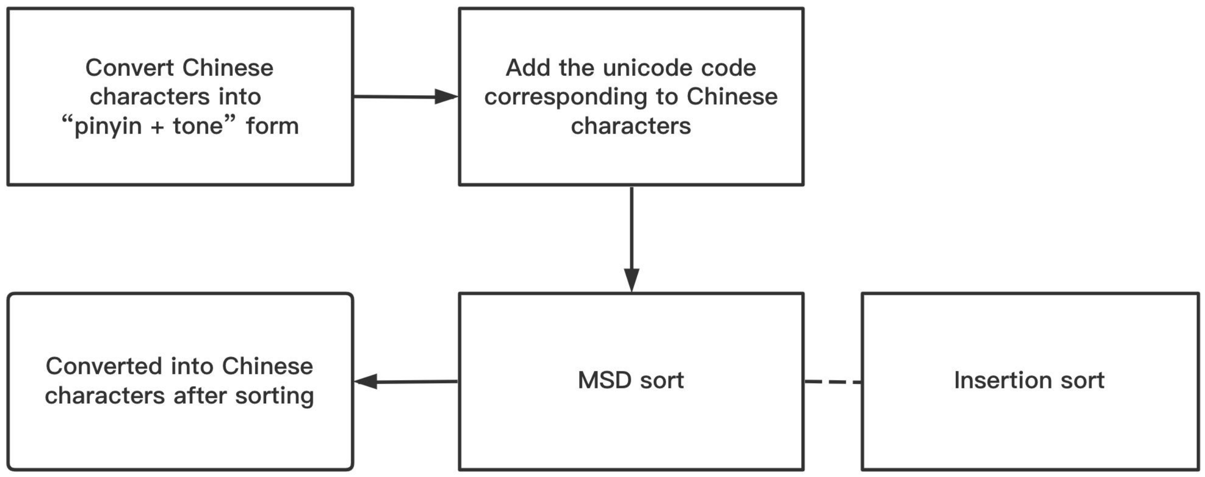
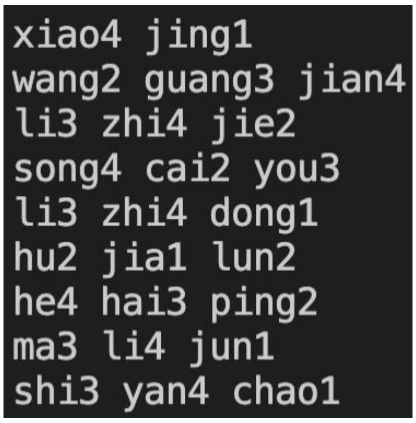

Convert Chinese characters into “pinyin + tone” form
Add the unicode code corresponding to Chinese characters
MSD sort
Insertion sort
Converted into Chinese characters after sorting

Figure1 algorithm steps

1. Using PinYin4J to transform Chinese into pinyin with tone：

Pinyin4j uses a character-Pinyin hash table to produce all valid Pinyins for a given character (Wei& Louise, 2009). This project uses PinYin4J to realize the function of converting Chinese characters into the form of "Pinyin + tone", so as to conduct alphabetical ordering of pinyin later.

Using defaultFormat.setCaseType(HanyuPinyinCaseType.LOWERCASE) define the pinyin converted from Chinese characters as lowercase. Using defaultFormat.setToneType(HanyuPinyinToneType.WITH_TONE_NUMBER) define the pinyin with tone. The Pinyin tones are represented by number. Figure 2 shows the output of the Chinese characters transformed by PinYin4J.



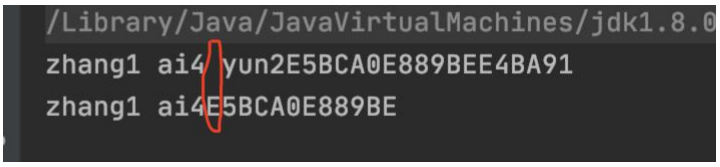
```
xiao4 jing1
wang2 guang3 jian4
li3 zhi4 jie2
song4 cai2 you3
li3 zhi4 dong1
hu2 jia1 lun2
he4 hai3 ping2
ma3 li4 jun1
shi3 yan4 chao1
```

Figure 2 Chinese characters are converted to the "pinyin tone" form through PinYin4J

2. Using Unicode to solve the problem that some Chinese have some pinyin and tone.

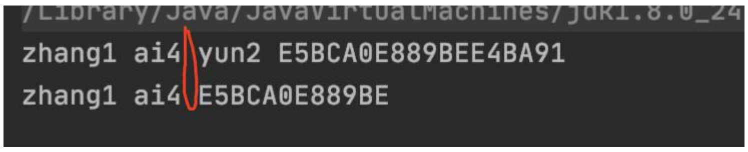
It is possible for Chinese characters to have the same pinyin and tone, for example "Zhao Si" and "Zhao Si" have the same pinyin structure and tone (zhao4 si4). This project mainly uses UTF-8 encoding to solve this problem. The CMA is Unicode based, and can handle both Simplified and Traditional Chinese text from a variety of locales, including Mainland China, Taiwan, Hong Kong, and Singapore (Thomas, 2000). The main solution is to use ConvertToUTF8 method to convert each Chinese character into its corresponding UTF-8 code. When the characters cannot be sorted after using pinyin and tone, the UTF-8 code can be used to distinguish them.

Before we add Unicode, We add space strings(" ") at end of pinyin. This will avoid the confusion caused by different pinyin lengths. For example{"张艾","张艾云"}. "张艾" should be in front of "张艾云". But if We can’t add space strings at end of pinyin. Corresponding pinyin+unciode for "张艾" is zhang1 ai4E5BCA0E889BE. Corresponding pinyin+unciode for "张艾云" is zhang1 ai4 yun2E5BCA0E889BEE4BA91. As shown in Figure 3. Because charAt(space) is 32 smaller than Letters and numbers. So "张艾云" will be in front of "张艾". But if we add space strings(" ") at end of pinyin, as shown in Figure 4. They have same space. They compare the following chars. Because we've changed all the characters to lowercase pinyin. charAt(lowercase) is larger than charAt(uppercase) and charAt(numbers). Unicode is made up of letters and Chinese characters. So the Chinese names having longer length will come after he Chinese names having shorter length. "张艾" will be in front of "张艾云".



```
/Library/Java/JavaVirtualMachines/jdk1.8.0
zhang1 ai4 yun2E5BCA0E889BEE4BA91
zhang1 ai4E5BCA0E889BE
```

Figure 3



```
/Library/Java/JavaVirtualMachines/jdk1.8.0_24
zhang1 ai4 yun2 E5BCA0E889BEE4BA91
zhang1 ai4 E5BCA0E889BE
```

Figure 4

3. Use map(or aux array) to record Chinese

When we use algorithms to sort pinyin, we use an auxiliary array or an map to record the one-to-one correspondence between pinyin and Chinese names. And we find that using aux array is more quick.

Auxiliary array: before we sort pinyin, we create a new auxiliary array to record Chinese. When in MSD radix sort, the element in pinyin array change its index. It's corresponding Chinese name will change same index. When pinyin array has been sorted, the Chinese array also has the correct ordered.

Map: before we sort pinyin, we use map to record pinyin and Chinese. When pinyin array has been sorted, we get Chinese array from the map.

4. Use MSD to sort pinyin, and when lo<=hi+15 we use insertion sort

The first line switches to insertion sort for small buckets(Juha& Tommi, 2009). In this project, when the size of the small buckets <=15, we use insertion sort, which will double the efficiency of the algorithm. Because after previous MSD radix sort, the inversions of the small buckets are greatly reduced. For a partially sorted small array, insertion sort is an efficient way.

The MSD Radix Sort starts with the most-significant digit and partitions the keys accordingly(Amr& Hosam, 2016). When sub array can't have correct order after allocating the highest. The sub array will allocate according to the second highest. Until the size of sub array less than 15, we use insertion sort.

5. Get ordered Chinese.

Finally, we get ordered Chinese from aux array or map (As we mentioned in step 3).

6. Compare MSD, LSD, Timsort, Dual-pivot, Quicksort, Huskysort

MSD(aux array)： transform Chinese to pinyin need time O(N), and need extra space(pinyin array) N. In sort, we also need extra space N auxChinese array to record(just like aux array in normal MSD). So the Chinese MSD the extra space N+DR , which is normal MSD need extra Space, add O(N), which is transform Chinese to pinyin need, and add O(N), which is auxChinese array. The extra Space is O(N+DR)+O(N)+O(N)=O(3N+DR). The the time for normal MSD is between O(Nlog(R)N) and O(WN), because the pinyin array is long and the depth of element in array is long. So the Chinese MSD time is unstable. the time of it is normal MSD time + transformation time, between O(Nlog(R)N) +O(N) and O(WN)+O(N). From my test, in this project the Chinese MSD time is about WN/2.

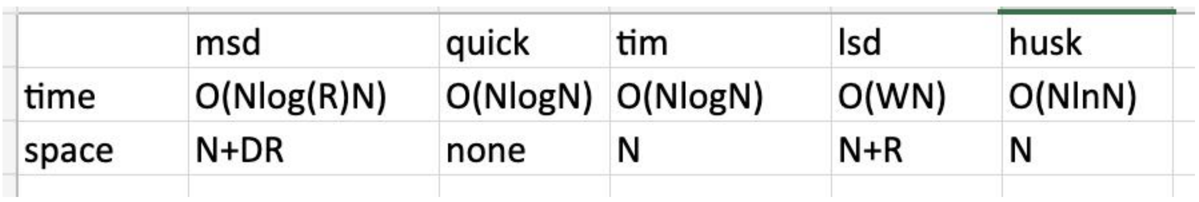
lSD(aux array):transform Chinese to pinyin need time O(N), and need extra space(pinyin array) N. In order to make every element have the same length, we need to add Maxlength-element.lenght(G=Maxlength-element.lenght) space string in short element, which average need O(GN) time. Chinese LSD need extra space is N+R, which is normal LSD need space, add N, which is transform Chinese to pinyin need. Chinese LSD need extra space 2N+R. The time is O(WN)+O(N)+O(GN)~O(WN+GN), O(GN) is adding G space string to make all elements have the same length need time.

TimSort(aux array):transform Chinese to pinyin need time O(N), and need extra space(pinyin array) N. In normal TimSort, it need extra space N. But in Chinese Timsort, we still need Space N of merge. So the Chinese TimeSort Need extra Space N, which is transform Chinese to pinyin need, add N, which is normal TimSort need, add N, which is merge operation need. So the TimSort need extra space N+N+N=3N. The time of Chinese TimSort is O(N)+O(NlogN)~O(NlogN). TimSort advantage: TimSort take advantage of insertion sort. So when the array is ordered or partially ordered, the time is less O(NlogN). In The best case, the time is O(N)

Dual-pivot Quicksort(aux array):transform Chinese to pinyin need time O(N), and need extra space(pinyin array) N. The normal Dual-pivot Quicksort don't need extra space, So Chinese Dual-pivot Quicksort need N extra space for transforming Chinese to pinyin. The time of Chinese Dual-pivot Quicksort is also O(NlogN). Dual-pivot Quicksort advantage: when the array is ordered, partially ordered or have many same keys, the time will not reach O(N^2), the time is about O(NlogN).

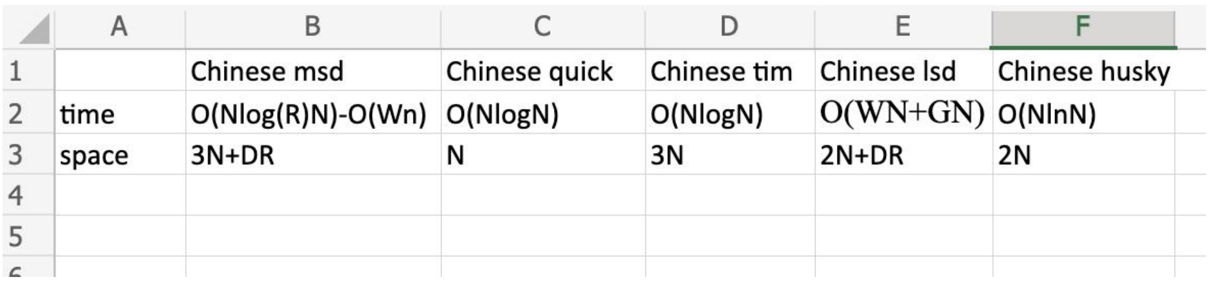
PureHuskySort: Transform Chinese to pinyin need time O(N), and need extra space(pinyin array) N. The PureHuskySort need extra space N, which is pinyin array need, add N(husky code need space). So the extra Space is 2N. The time of PureHuskySort is O(NlnN). PureHuskySort advantage: quicksort can sort an array of (64-bit) longs very quickly. So PureHuskySort use long[] to compare.

The normal MSD, LSD, Dual-pivot Quicksort, Timsort, PureHuskySort



	msd	quick	tim	lsd	husk
time	O(Nlog(R)N)	O(NlogN)	O(NlogN)	O(WN)	O(NlnN)
space	N+DR	none	N	N+R	N

The MSD, LSD, Dual-pivot Quicksort, Timsort, PureHuskySort For Chinese(including Transformation Chinese to pinyin, sort pinyin, and get Chinese)



	A	B	C	D	E	F
1		Chinese msd	Chinese quick	Chinese tim	Chinese lsd	Chinese husky
2	time	O(Nlog(R)N)-O(Wn)	O(NlogN)	O(NlogN)	O(WN+GN)	O(NlnN)
3	space	3N+DR	N	3N	2N+DR	2N
4						
5						
6						

References

Wei, Louise(2009), Chinese Pinyin-Text Conversion on Segmented Text, Lecture Notes in Computer Science, 5729,428. DOI:10.1007/978-3-642-04208-9_19

Thomas EMERSON(2000), Segmenting Chinese in Unicode, Google Scholar, 56(3), 631-644. DOI: https://doi.org/10.7202/1008337ar

Kärkkäinen J., Rantala T. (2008) Engineering Radix Sort for Strings, Lecture Notes in Computer Science,5280. DOI: https://doi.org/10.1007/978-3-540-89097-3_3

Elmasry, A., & Mahmoud, H. (2011). Analysis of swaps in radix selection. Advances in Applied Probability, 43(2), 524-544. doi:10.1239/aap/1308662491