Large Scale Geographic Data Transformation

A Solution Based on Baidu Map

Silin DU

Department of Management Science and Engineering, Tsinghua University
dsl21@mails.tsinghua.edu.cn

May 22, 2022

清華大學
Tsinghua University

Contents

1 Introduction **3**
1.1 Background 3
1.2 Data 3

2 Our Solution **4**
2.1 Pretests 4
2.2 Details 4

3 Conclusion **5**

1 Introduction

1.1 Background

When dealing with geographic data, we always need to perform the (inversed) geographic encoding task, such as querying the points of interests (PoIs) according to GPS coordiates, transforming the address into GPS coordiates, etc.

Fortunately, some map service providers open their APIs for individual developers, such as Baidu Map and Google Map. We choose the Geoencoder API provided by Baidu Map to transform the addresses (in text) into corresponding GPS coordiates.

In many cases, geographic data is collected by several sensors automatically and the quantity is huge, probably over 10M. It's difficult to handle with such large scale geographic data, since each individual developer in each day can only call the API no more than 300K times.

How to perform large scale geographic data transformation within a reasonable time is the focus of this project.

1.2 Data

Our task is to query the corresponding GPS corrdinates of addresses from different provinces in China. The original data are stored in 31 files with different length. The statistics of each file is shown in Table 1. The total number of data is around 50M.

Table 1: Data Statistics

File Name	Data Length	File Name	Data Length
prov44	5,750,534	prov61	1,113,420
prov32	4,875,197	prov45	1,036,779
prov37	3,732,120	prov36	1,009,106
prov33	3,251,779	prov53	986,436
prov31	3,207,900	prov12	845,332
prov11	2,153,197	prov23	839,355
prov41	2,072,953	prov22	837,629
prov51	2,035,760	prov52	827,954
prov13	1,962,199	prov15	774,629
prov42	1,738,211	prov62	611,008
prov21	1,651,516	prov65	598,118
prov34	1,566,368	prov46	342,895
prov35	1,533,310	prov64	247,826
prov43	1,237,827	prov63	149,818
prov14	1,159,999	prov54	73,195

Table 1 —— Next Page

Table 1 —— Continued

File Name	Data Length	File Name	Data Length
prov50	1,116,238		
Total: 49,338,608			

2 Our Solution

2.1 Pretests

Because of the limitations of individual accounts, we need around 167 days for transformation if we only have one account. Thus, we collect 27 AKs for this project and use three servers to run the codes.

Before formal transformation, we perform some pretests and get the following observations.

1. It takes around 6 hours to consume the limitation of a AK (300K).
2. Query might be failed because of some irregular tokens/symbols.
3. Some addresses are not clear enough, we might add some prefix (province/city name) to improve the precision.

2.2 Details

We first group files together to ensure that each group contains roughly the same length of data. Finally, we get 11 groups, each of which contains around 4.5M data. After that, we segement the files in each group into several parts with length no more than 295,000 , each of which is named like *group id_original file name_part id*. The statistics of each group is shown in Table 2.

Table 2: Group Statistics

Group ID	File Name	Number of Parts	Data Length
1	prov44	20	5,750,534
2	prov32	17	4,875,197
3	prov37, prov43	18	4,969,947
4	prov33, prov35	18	4,785,089
5	prov31, prov34	17	4,774,268
6	prov11, prov41	16	4,226,150
7	prov51, prov13	14	3,997,959
8	prov42, prov21, prov14	16	4,549,726

Table 2 —— Next Page

Table 2 —— Continued

Group ID	File Name	Number of Parts	Data Length
9	prov50, prov61, prov45, prov36	16	4,275,543
10	prov53, prov12, prov23, prov22, prov52	16	4,336,706
11	prov15, prov62, prov65, prov46, prov64, prov63, prov54	14	2,797,489

Before transformation, the following configurations are needed to be predefined.

- group_file: path of the file that contains file names and corresponding province names.
- log_path: path to save the logs.
- ak_path: path of the file that stores all AKs.
- file_path: the directory of the data (after grouping and segmentation).
- error_path: path to save the error files.
- target_path: path to save the transformed data.
- group_id: range from 1 to 11.
- split_id: index of the part to be transformed.
- key_id: range from 0 to 26.

After running the code, the logs will save the following information.

- The configurations specified above.
- Error information: e use $(-1, -1)$ to label the addresses that failed to be transformed.
- Error address: we save the error addresses and the corresponding reasons in new `CSV` files.
- Start Time.
- Finished Time.

3 Conclusion

It took around 10 days for the whole transformation. And the overall error rate is around 0.57%, which varies in different provinces. The followings are most common error information.

- Connection Error: mostly due to the network environment of the server.
- Null Address: missing values.
- Server Internal Error: error raised by the server of Baidu Map.
- Parameter Invalid.

We conclude that the preprocessing is quite important to decrease the error rate.