Project Report

518030910059 惠宇龙

Introduction

There is a surge in urban metro traffic in China in the past couple years, which causes overcrowding in metro systems of many cities. This application is used to analyze the data from Hangzhou-metro and may offer some help to the manager. Generally speaking, it focuses on practicality. And the main use of it concludes:

1.make an image of a certain station flow

2.seek the correct paths. (which two are mandatory part)

3.compare the different station-flows at the same time

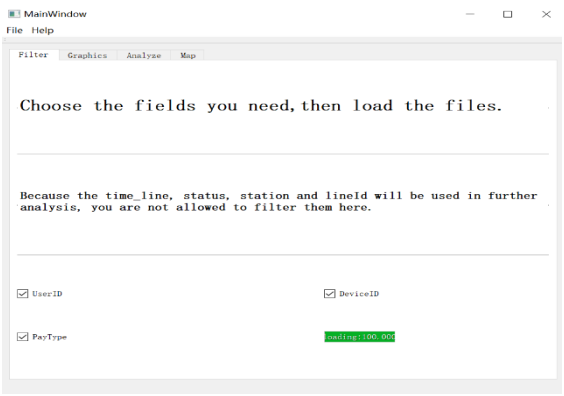
4.sort and present the busiest station/line during a certain time.

Implementation Details.

1.Overall arrangement

As shown in the pictures below, the main window is composed by menu and four tabs. (I think the program is simple so I do not set status bars or tool bars). The four tabs have their own functions as written, which are simple enough to understand.

2.Filter and files-loading

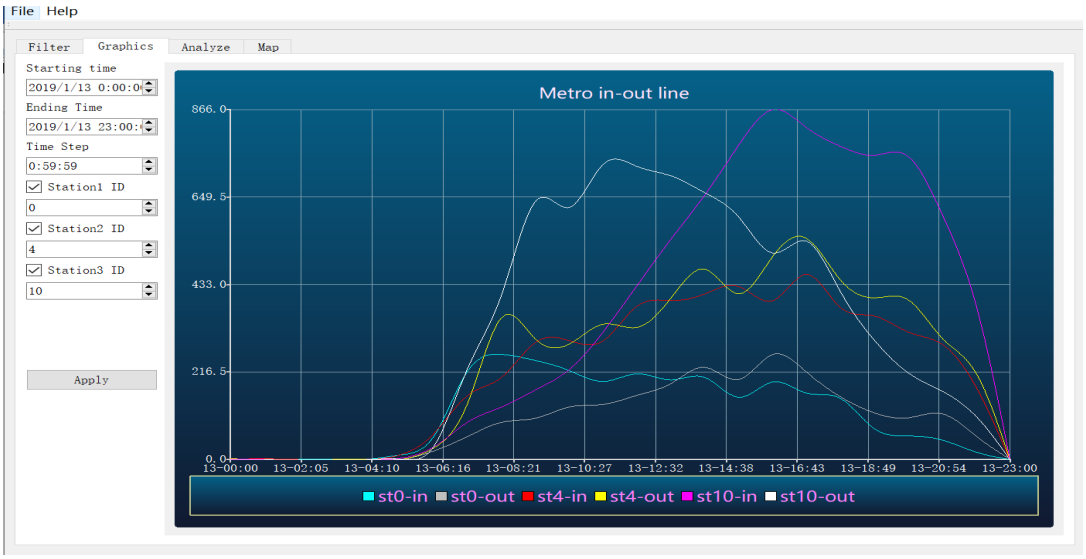


In the filter, I just set 3 fields for choosing. Because in the further analyze, it needs all the stations' information to count the busiest line and the busiest station. (time-line, status and line-ID are not set optional for the similar reason).

The user is advised to load the map before the detailed data, because the map will help to get the range of stations, and check the number-overflow when input the station later. (default range:0~80).

I write a thread to load the file, and set a process bar. Because loading may take some time and the window will not get stuck with another threads. Then I use "qfiledialog" to choose the files (you can choose several files at one time), and approach the files through the filenames. I define a struct "metro" and create a vector container to memorize the data. Specially, the data is loaded just as string without more processing which can decrease the time effectively to 40s for all files.

3.Graphics

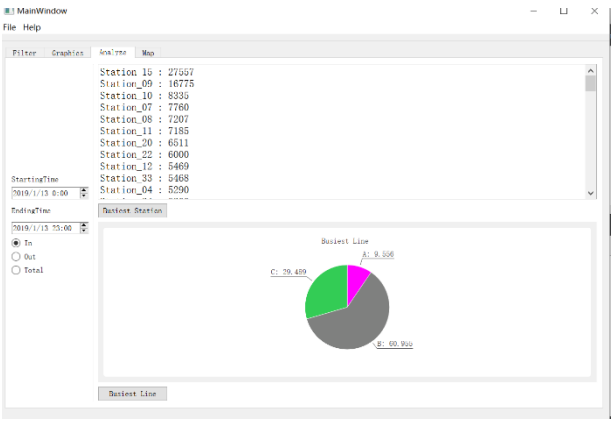


As you can see, the user can tune the time and implement 3 stations in the one picture. (After trying and modulating, I find that three curves are the most suitable with ensuring a clear view.)

In this part I create an int array according to the start, end and step. Then I filter the data with station, if it is chosen, I transform the timeline string in container to qdatatime and calculate the correspond place in the array in order to count. Then I use qchart to paint the line and smooth it with simpline in qt library.

4.Analyze

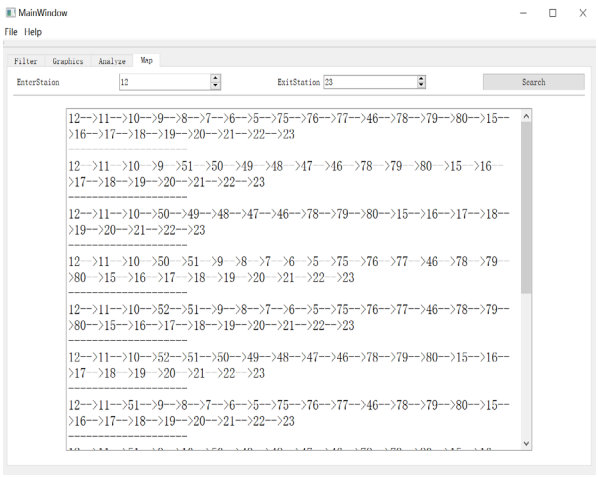
Let the user to input the start/end time and in/out/total pattern. I create two arrays for stations and lines and filter the data with time and status and



add it to the corresponding stations and lines. Then I just sort the stations by its flow and present the busiest lines on a pie chart.

5.Map

Let the user to input the enter/exit station. Then I use DFS to find the correct paths, (see the code for more details) and present them on a text browser.



Result

1.About the program, it shows a not-bad time performance: loading all of the files will take only 45 seconds, while drawing and other operations will take less than 10 seconds.

2.About the performing result, it has been shown above.

3.As it is stated in introduction part, the motivative functions have been implement: (I will explain the analyzing，while others are simple)

(1) Busiest station: With the help of the data, we can decide how many in-check/out-check devices are need for different stations. The total flow tells us which station needs the most police to control (deferring with time)，the most space so that it can be less crowded.

(2) Busiest line: During different period, the busy-line data will certainly differ and this part provides the crowd-information to help the

manager adjust the time gap between subway departures in time.

Discussion

1. I make some optimization to decrease the loading time, just loading the string and doing no processing, which will be done when painting. This measure helps to low the loading time to 45 second, while others may spend more than 2 minutes. For exchange, while others spend 3 seconds to paint, I used 6 seconds. But I think this exchange is worthy.

2. Some students process the data ignoring the date and just analyze base on "one average day", but I see the big difference from the weekday and weekend, so I save the date when painting.

3. By analyzing, station 15 and line-B are far crowded than others, which means more police and shorter gap time.

4.For example, station-0 is more crowded during weekdays, so we can indicate there may be some companies or apartments instead of the mall or scenic. Besides, in the morning the main flow is "in". This can confirm the apartments instead of company, and the manager can trans the escalator and device to in-mode.