PA2 Report

B06703025 劉致豪

The program is compiled and executed on the server EdaU6.

The structure of the program:

```
bin/      mps
doc/      report.pdf
src/      main.cpp
          maxPlanarSubset.cpp
          maxPlanarSubset.h
makefile
README
```

The execution order of the program:

First, I put all in the main.cpp to execute the whole program, create vector Connection, matrix (vector in vector) Matrix M, and vector Choice, and read input, do mps, and write output.

Second, I compiled the core code in the file maxPlanarSubset.cpp, create constructor, fillMatrix, fillChoice, three parts in total.

Last, I used the header file maxPlanarSubset.h to declare the class of maxPlanarSubset.

About the structure of the file maxPlanarSubset.cpp:

read input: read the first line to confirm the points we have, and create an array Connection which stores the index of the connection points.

do mps: use maxPlanarSubset.fillMatrix and maxPlanarSubset.fillChoice.

write output: write down the total chosen chords MatrixM[0][Points-1] at the first line, and write down all the Connection index and it's element of which it's Choice element is 1 in the increasing order.

About the structure of the file maxPlanarSubset.cpp:

fillMatrix: create a matrix MatrixM which stores the total chords between point i and j, the MatrixM also implies the MIS(i,j), furthermore, there exists three cases, we give three different index for each case.

fillChoice: do the recursion and create an array Choice which stores the bool index of the chosen points/chords, then use the data in MatrixM to check the composition of the chosen points/chords, give 1 to the points/chords which has been chosen.

About the three cases of the part fillMatrix:

Case1: k out of M[i,j], MatrixM[i][j] = MatrixM[i][j-1].

Case2: k = i, MatrixM[i][j] = MatrixM[i+1][j-1]+1.

Case3: k in M[i,j], MatrixM[i][j] = max{MatrixM[i][j-1],MatrixM[i][k-1]+1+MatrixM[k+1][j-1]}.

About the three cases of the part fillChoice, recursion of the region (i, points-1):

Case1: MatrixM[i][j] = MatrixM[i][j-1], do recursion of the region (i, j-1).

Case2: k = i, Choice[i] = 1 and do recursion of the region (i+1, j-1).

Case3: MatrixM[i][j] = MatrixM[i][k-1]+1+MatrixM[k+1][j-1], Choice[k] = 1 and do recursion of the region (i, k-1) and the region (k+1, j-1).