Code copyright descriptions:

Parts of codes for method APPX-- finding a minimum perfect matching in a general graph, which include files “Matching.cpp”, “BinaryHeap.cpp”, and “Graph.cpp”, are from the standard codes shared on the website Chinese Software Developer Network (CSDN). Due to copyright issue, they cannot be shared in our code. To use these files, one can download them from the link https://download.csdn.net/download/weixin_42105169/18558404?username=u01388261 and adjust the parameters as necessary. The method **SolveMinimumCostPerfectMatching** to find a minimum perfect matching in a general graph is called in the file “APPX\src\main.cpp”, using the codes as shown below:

```
1107        Matching MM(G);
1108        pair< list<int>, double > solution = MM.SolveMinimumCostPerfectMatching(cost);
```

Note: The parameterized constructor Matching receives a paramter G, a graph instance, and then initializes a Matching instance MM. The method **SolveMinimumCostPerfectMatching** of the instance MM solves the minimum cost perfect matching problem with a parameter cost. This parameter in the method is a vector representing the cost of edges in the graph G. If the graph does not have a perfect matching, a const char * exception will be raised. Otherwise, it returns a pair, where the first element is a list of the indices of the edges in the matching and the second element represents the cost of the matching.

Code instructions:

To run the code smoothly and correctly, please set the parameters in the file “APPX\src\main.cpp” as described below.

1. Set the depot number.

```
#define type 6 //type of depot: 2/4/6
```

2. Set the instance type.

```
#define data_type 0 //type of instance: SD=0, p/pr=1
```

3. Set the file name.

```
string depots = "SD19.txt"; //instance file name
```

4. Set the absolute path of the instance.

- For the instances in the folder “data\SD set”, modify the string variable “path_0_depots”.
- For the instances in the folder “data\P set”, modify the string variable “path_1_depots”.

```
string path_0_depots = "/home/li/CLionProjects/APP/instance/SD set/k=6/"+depots; //instance file path
string path_1_depots = "/home/li/CLionProjects/APP/instance/P set/"+depots;
```

5. Set the vertex number, which is the sum of depot number and customer number.

```
#define vetex_number 198 // num of vetex( = depots num + customer num )
```

6. Set the upper bounds for the following parameters to avoid overflow errors.

```
#define transfer_idx_limit 20000  //upper bound for permulation and combination
#define transfer_limit 2000   //upper bound for edge exchange times except single edge exchange
#define transfer_limit_all 100000   // upper bound for edge exchange times
```

7. Set the number of feasible solutions by Algorithm 2.

```
#define add_limit 100000
```

8. Set the absolute path for the result.

```
    string result_path = "/home/li/CLionProjects/APP/RESULTS/k"+to_string(type)+"/"+ depots;
```

The directories of the results are shown as follows: