**一、处理过程记录**

**1．使用数据**

order.csv中dwv\_order\_make\_haikou\_1.departure\_time，O\_location，D\_location三列数据。

**2．两点之间的路程计算**

调用python的networkx库，读links.csv中的街道数据建立无向图（街道长度length，街道起点from\_node\_id，街道终点to\_node\_id），匹配时用内置函数nx.dijkstra\_path\_length()获得点对点的最短路。

**3．匹配规则**

每次考虑两个订单之间的匹配，记这两个订单为和，对任意订单：表示订单起点，表示订单终点，表示订单开始时间，表示订单行程长度（即和之间的最短路）。

3.1一个订单只在发起请求的10分钟之内进行匹配

即

3.2绕路不超过3km

对和考虑四条拼车路线，任意一条满足即视为和之间可形成拼车。

①

对于，绕路=

对于，绕路=

②

对于，绕路=

对于，绕路=0

③

对于，绕路=

对于，绕路=

④

对于，绕路=0

对于，绕路=

**4．最大匹配**

用一般图的最大匹配算法求解上一步形成的可共享网络的最大匹配。

**二、计算结果**

目前求解得到的结果中，原始1766个订单在最大匹配中有1762个订单，拼成率99.77%。

具体匹配关系在表格snw0501\_maxmatch.csv中，订单编号对应order.csv中第一列。