这次老师提供的constructive heuristics就是简单的在排序过程中运用枚举和比较的方法。

时间上仍旧是按之前那样设置Times windows来将时间转化成数值来比较大小。依旧是我上次发邮件里写到的那样。

不同的点在于这次要先把所有不同地址间的距离先算出来。也就是类似compatibility的匹配结果那样要让程序在匹配完初始时间和flexibility的情况下对所有的可能结果进行距离上的比较，选取time cost 最小也就是距离最短的放在sequence最前面，walk location也是这样，先确定狗和那些walk location的要求里的契合程度最高，然后在排 一个job 的时候计算完取狗的最小时间再找距离最近的walk location。

因为当狗三只狗一起遛的话walk duration是重合的所以一般情况下必然比分开单独遛三只狗要花三次duration时间来的短，以此来凭借最短的processing time(=setup time + normal walk duration + return time， 这个时间是一个完整的job时间。)来进行排程。

另外确认了dog walker每天中午要30分钟吃午饭的时间，这个duration可以灵活安排在中午的任何时段，不算上从其他狗主人那里回到家和下午从家出发去到另一个狗主人家的时间。所以还是像刚开始那样把她回家吃饭这个作为一个per day in a week 和duration=30mins 然后start time在11点（考虑到有部分工作是从12点开始）的order。

地址的距离(时间)检索说是要套用google distance matrix API来计算距离和花费时间。其实主要就是花费时间。检索过程在如下范例表格：（单位秒）

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Customer number |  | 地址1 | 地址2 | 地址3 | … | 地址m-1 | 地址m |
| 0 | 地址1 |  | 267 | 321 | …. | 543 | 325 |
| 1 | 地址2 |  |  | 441 | …. | 677 | 4363 |
| 2 | 地址3 |  |  |  | …. | 1354 | 346 |
| …. | …. | …. | | | | | |
| m-1 | 地址m-1 |  |  |  | …. |  | 213 |
| m | 地址m |  |  |  | …. |  |  |

这个地址就是要把所有的包括遛狗者，其他所有狗主人和walk location 的地址都放进去检索。在确定完多少只狗可以一起走的情况下再取最小距离的狗主人家和walk location距离。

Walk location 这个。。我其实还不知道怎么把那个思路描述出来。。计算出狗对多少个items是适应的然后再去和walk location所具备的条件去匹配？