Scheduling of dog walking problem

（没有材料的paper我就直接用中文描述了）

这个project就是为一个专门负责提供遛狗服务的商家做一个program来安排她每日遛狗的行程。

这算是个小企业主，自己经营一家遛狗公司，专门接一些帮助客户在他们没空的时段帮忙遛他们的狗，附件的excel里是商家提供的数据，目前她登记的狗的数量就是31只（包括她自己的狗-0号Glenn和客户的30只），各项parameters分别表示：

Customer No. 顾客编号

Postcode 住户地址编号（这个似乎要转换成地理坐标用来计算点到点之间的距离）

Coordinates 坐标，计算点到点距离

Dog Name 狗的名字

Breed 狗的种类

Size 狗的大小

（以上三个就是作为基本项到时候直接读取和输出）

Time 到客户家领狗的时间（从早上开始）

Flexibility e.g.对于0号顾客的狗Glenn，领取的时间是12点，flexibility是1小时，说明从12点到下午1点这个时间段都可以去领狗。

Days 一般来说就是算weekdays的5天，对于那些2 times per week的就是五天里任意选两天

R/NR 这个算是备注，就是说明到底这个客户的订单是regular的还是no regular的

Duration 每次遛狗的时长

Notes 也是备注，holiday的可以先不纳入排程范围或另外单独排程

每一次的job（一次完整的遛狗）的时间包括准备时间的setup time（即从前一个地点去往下一个地点的时间），遛狗时间duration和将狗还回狗主人家的后续时间，但这里就不考虑后续时间而将这个时间记入下一个job的setup time。另外商家因为需要吃午餐的原因所以需要将中午的一个吃饭的特定时间段（设定为20分钟-30分钟）也假设为一个job（即此job就是从上午最后一个job结束后返回商家自己的家中）。

另外一个需要设置的是关于狗的compatibility，商家为了效率可以用车子一次性载最多3只狗，即同一时间段商家可以遛最多3只最少1只狗，但狗与狗之间不一定能够和谐相处，于是在sheet2里面有狗的compatibility表格，具体参数象征如下：

-1:    Not compatible – the dogs cannot be walked together.

0:     Unknown - this would be set for a new dog (i.e. this dog with all other dogs would be 0, or –1 where the other dog must be walked alone)

        This dog should be walked alone until the dog walker gets to know it better.

1:    Probably compatible – Natalie thinks these dogs should be ok together.

2:    Compatible – Natalie has previously walked these dogs together or thinks they would be fine together.

If you could work with these values directly, that would simplify the configuration data for the application, but if you wish to change the values used, then that

would not be a problem (e.g. using values 0 to 3  instead of values –1 to 2)

1和2都可以看做狗可以放到一起。

导师的建议是整个过程可以分成以下步骤：

1，成功读取sheet1数据，将有效的订单数据（即有登记regular的狗的数据匹配到每个工作日里并初步估算每个工作日大概所需的工作时间（从早上8点开始，加上中午吃饭的时间）；

2，进行判断狗的compatibility的程序来确保value为0或-1的两只狗不会被安排在一起；

3，运用beam search（基于branch-and-bound heuristic的algorithm）的算法进行节点的筛选最终得出每一天的遛狗排程；

4，将所得排程结果display在一张新的work sheet里。

（另外对program窗口的要求就是能够实现读数据，计算，列数据）

一般还是采用像上次那样在距离上一节点距离短的所有节点里找到符合上述条件的点进行删选和入列。

经纬度算法：

设两点A、B的经、纬度分别为(jA,wA)(jB,wB)，则半径为R的球面上两点间的最短距离(大圆弧)为：

弧AB=R\*arccos[sin(wA)sin(wB)+cos(wA)cos(wB)\*cos(jA-jB)]

地球是一个近乎标准的椭球体，它的赤道半径为6378.140千米，极半径为6356.755千米，平均半径6371.004千米。如果我们假设地球是一个完美的球体，那么它的半径就是地球的平均半径，记为R。如果以0度经线为基准，那么根据地球表面任意两点的经纬度就可以计算出这两点间的地表距离（这里忽略地球表面地形对计算带来的误差，仅仅是理论上的估算值）。设第一点A的经纬度为(LonA, LatA)，第二点B的经纬度为(LonB, LatB)，按照0度经线的基准，东经取经度的正值(Longitude)，西经取经度负值(-Longitude)，北纬取90-纬度值(90-Latitude)，南纬取90+纬度值(90+Latitude)，则经过上述处理过后的两点被计为(MLonA, MLatA)和(MLonB, MLatB)。那么根据三角推导，可以得到计算两点距离的如下公式：

C = sin(MLatA)\*sin(MLatB)\*cos(MLonA-MLonB) + cos(MLatA)\*cos(MLatB)

Distance = R\*Arccos(C)\*Pi/180

这里，R和Distance单位是相同，如果是采用6371.004千米作为半径，那么Distance就是千米为单位。

如果仅对经度作正负的处理，而不对纬度作90-Latitude(假设都是北半球，南半球只有澳洲具有应用意义)的处理，那么公式将是：

C = sin(LatA)\*sin(LatB) + cos(LatA)\*cos(LatB)\*cos(MLonA-MLonB)

Distance = R\*Arccos(C)\*Pi/180

以上通过简单的三角变换就可以推出。

如果三角函数的输入和输出都采用弧度值，那么公式还可以写作：

C = sin(LatA\*Pi/180)\*sin(LatB\*Pi/180) + cos(LatA\*Pi/180)\*cos(LatB\*Pi/180)\*cos((MLonA-MLonB)\*Pi/180)

Distance = R\*Arccos(C)\*Pi/180

附:源代码

**[plain]** [view plaincopy](http://blog.csdn.net/yl2isoft/article/details/16367901)

1. /\*\*
2. \* google maps的脚本里代码
3. \*/
4. private const double EARTH\_RADIUS = 6378.137;
5. private static double rad(double d)
6. {
7. return d \* Math.PI / 180.0;
8. }
10. /\*\*
11. \* 根据两点间经纬度坐标（double值），计算两点间距离，单位为米
12. \*/
13. public static double GetDistance(double lat1, double lng1, double lat2, double lng2)
14. {
15. double radLat1 = rad(lat1);
16. double radLat2 = rad(lat2);
17. double a = radLat1 - radLat2;
18. double b = rad(lng1) - rad(lng2);
19. double s = 2 \* Math.Asin(Math.Sqrt(Math.Pow(Math.Sin(a/2),2) +
20. Math.Cos(radLat1)\*Math.Cos(radLat2)\*Math.Pow(Math.Sin(b/2),2)));
21. s = s \* EARTH\_RADIUS;
22. s = Math.Round(s \* 10000) / 10000;
23. return s;
24. }

NR 暂时不做考虑，先放到一边。

一周两次的狗，也放到一边。狗的数量上有变化。

下午5点之前能回到（备选）。

Flexibility 小于这个时间段的也接单。

谷歌定位不要计算入时间。