设计文档

赛题分析

题目的背景是平安产险首推的爆款服务“5*10 城市急速现场查勘”，全国地市级以上城市的平安客户，如日间发生需现场查勘的交通事故，可在 5-10 分钟内享受理赔全流程的极致体验。

假设查勘员每天 0 点以平安国际金融中心为起始点（0，0），在 100*100 的网格地图中工作，题目给出的条件有如下几个：每天查勘车的成本为 10 万元；行驶成本为 0.5 元每公里；车速为 40km/h；事故分时间段发生概率有差异；查勘车处理事故耗时 30 分钟；此外还有引申条件：不可达区域和事故高发区域。需要设计算法得到最佳查勘车分配方式。此外，得分公式为

总得分=每起事故平均等待时间*0.618+总成本*0.382

根据题目可以获知以下几点信息：每天发生事故总数是确定的，为 312 起；由于网格的移动方式，因此可以得到最大的移动距离为(0,0)->(100,100)，其距离为 200km，因此通过计算可以得知查勘车最大的移动时间为 200/40=5 小时，再加上处理事故的 30 分钟消耗，因此最长的车辆从接到任务到完成任务的时间跨度长达 5.5 小时，这也是等待时间的最大值（不考虑车辆全部被占用的情况下）；引申条件中的不可达区域是指在该时间段内不可达，因此在该时间段内发生的事故均需要视为不可达最后时刻发生的数据,但是需要将推迟发生的数据得到的等待时间加上原本数据处理等待的时间;赛题中给到的一点是查勘车成本按照最大查勘车数量计算,这样就会产生一个问题即赛题中提到的查勘车在派出之后处理完事故需要等在原地，直到派往下一个事故发生点，也就是说，如果在最开始的时刻即安排最大的查勘车，通过算法安排车辆查勘之后，如果让其没有接收到任务就一直等在原地，此时的消耗成本为 0，因此这样的理解方式赛题举例的按时段分配就没有了意义。

算法设计

根据题目条件，等待时间 T 尽量少，成本 S 尽量低，以这两个条件设定 N（N 为勘察车数量，由赛题分析部分可知，N 可以设定为一个相对固定值，该固定值需要使用算法求解最佳值）。

$$\mathrm{T} = \frac{\sum A_{ti}}{N}$$

其中A_{ti}为第 i 起事故等待时间。

$$\mathrm{S} = \mathrm{N} * 100{,}000 + \sum_{i=1}^{N}\sum_{j=1}^{M_i} Sp_j * 0.5$$

其中M_i为第 i 辆车的总任务次数，Sp_j为每次任务所行驶的路程。

将以上两个公式相加就可以得到最终分数的计算公式：

$$\text{Sum} = \text{S} + \text{T} = \frac{\sum A_{ti}}{N} + \text{N} * 100{,}000 + \sum_{i=1}^{N}\sum_{j=1}^{M_i} Sp_j * 0.5$$

于是可以得到：

$$\text{Sum} = \mathcal{F}(\text{N}, A_{ti}, Sp_j)$$

而A_{ti}和Sp_j又是与N和分配方式相关，因此该题可以简化为求N的最佳值。

该题可以建模为一个线性规划模型。对于一个确定的N，需要通过一定的分配方式来获得最后的解。因此分配方式将决定算法的优劣。在分配问题中，典型的算法是动态规划算法，每一种分配方式都有一个可行解，本题中需要求解一个最小解，因此需要完成的任务就是求解最优解。但是针对本题，选择使用另外一种分配策略：贪心算法。通过一次次迭代求解最优解的过程中，得到的数据表明，分配规则的连续变化将导致最优解的线性变化，因此可以采用贪心算法来求解局部最优解并假定其为全局最优解。

贪心算法在分配的每一步都会采用局部最优选择，虽然总体上无法和动态规划得到的最优解相似，但是在该问题中依然可以采用该算法。

具体实现的步骤是，在每起事故发生的时候，得到的信息有事故发生的位置信息，时间信息，以及查勘车的状态。通过遍历所有的查勘车，计算出每辆查勘车的分数，然后选择最小的分数的车辆分配任务。

$$\text{Point} = \mathcal{F}(A_{ti}, Sp_j)$$

分数为A_{ti}和Sp_j的函数。

在实现贪心算法是过程中，需要注意的是对于输入的时间序列，需要进行排序，按时间从小到大进行排序。

通过贪心算法得到在确定性N的情况下的分数，就可以得到分配方案，但是事实上N是一个变量，该问题为一个线性规划问题，但是由于N只能取整数，因此该问题是一个整数型线性规划问题，该问题是一个NP难问题。但是在分配阶段采用的算法是贪心算法，因此每次只要输入数据是确定的，对于给定的N，此时输出也是确定的。因此可以直接采用遍历算法，遍历N在一定范围内的值，计算每个N的分数，然后取出最小分数的N和分配方案。

因此可以得到整个算法的伪代码：

```
bestSource=Float.MAX_VALUE;
bestCarNum;
For carNum in range(carMin,carMax):
    currentSource=0;
    for timeList in AccidentList:
        arrangeSource= Float.MAX_VALUE;
        bestCarArrange;
        for carID in CarID:
            hereSource=Point(carID);
            if hereSouce< arrangeSource:
                arrangeSource=hereSource;
                bestCarArrange.append(this.carArrange);
    currentSource=Source(carNum)
    if currentSource< bestSource:
        bestSource= currentSource;
        bestCarNum=carNum;
```

程序设计

程序设计采用的语言是 Java，JDK 版本为 jdk1.8.0_151。程序中使用了一个 jar 包 jxl-2.6.jar，该 jar 包是用来读取和写入 excel 文件，输入文件为 input.xls，输出文件为 output.xls。输入文件的格式为从第二行开始，第一列为事故发生时间，第二列为事故发生横坐标，第三列为事故发生纵坐标。该输出文件是由程序随机生成，在程序中可以直接调用 ExcelProcess.writeExcelRandom()，生成的文件名为 input.xls，在项目文件夹下。

输出文件为 output.xls，因为输入是 30 天的数据，题目中要求是每天凌晨从总部出发，因此将输入的数据按天，每天执行一次代码。输出文件的格式为第一行存储的是 totalCarNum，表示使用的车的数量，第二行存储的是 totalWaitTime，表明总的等待时间，第三行是 totalMoveDistance，表示车辆总的行驶里程。接下来的行数存储的是车辆的具体安排信息，例如车辆编号（从 0 开始），车辆出发点，到达点，运行距离，等待时间，事故发生时间等信息。