队伍编号	202248
题号	C

基于蚁群算法的仓内拣货作业调度优化分析

摘 要

本题是典型的仓内拣货作业调度问题。如何规划好仓内拣货路线使拣货时间尽可能短、整体仓库拣货运作效率高成为本题的核心。

针对问题一，本题是模型建立和算法设计的基础。先对货架与复核台进行编码，再对两点相对位置进行分类求算距离。距离的求解需要分别使用横轴和纵轴的坐标值，进而计算平面图上路径沿这两个方向的路程分量之和。

针对问题二，采用蚁群算法建立拣货优化模型，将距离矩阵输入模型，经过 200 次算法仿真迭代得到最短回路得到拣货理想路线，进而计算出出库时间。

针对问题三，属于不存在复核台工作量堆积现象的拣货路径规划问题。基于第二题的蚁群算法建立优化模型，运行程序得到 5 个任务单各自的最短路径线路。再运用分类组合的思想将所有路线尽可能地按“一个任务单的终点复核台位置为下一个任务单的起始复核台位置”来排布，得到理想的任务单领取顺序的总理想拣货路线，再求出总的出库时间与各个复核台利用率。

针对问题四，本文章采用了简化上所建立的优化模型的方案求解，设定任务单拣货路线起点终点相同。以一个统计量均值来代替各个任务单拣货理想路径长度，得到一个统计时间均值代替各个任务单的拣货完成时间，以整体水平消除个体的差异，减少影响因子。基于问题三蚁群优化模型，通过优化拣货人员在复核台的人数分配和拣货人员手中的任务单数量分配，得到理想拣货路线，出库时间为单个复核台最大工作时间和第一个任务单的拣货完成时间之和。

针对问题五，根据构建的模型，对拣货人员在复核台的人数分配和拣货人员手中的任务单数量分配进行优化，得到优化分配方案。代入模型计算，即可得到增加一个可正常工作复核台后出库时的整体估计水平，获得定性、定量地分析结果。

针对问题六，我们从畅销品离复核台的位置与畅销品所在货架拥挤程度两个因素出发。根据距离复核台长短以及实际位置情况，我们把货架划分为四个区域：A 区，B 区，C 区，D 区。并从货架上畅销品占比在四个区域呈现逐次递减的趋势得到基本的货架商品摆放方案。

关键词：蚁群算法；路径规划；统计运筹；信息素；

目录

第一章 问题背景

随着经济的不断发展，商品的流通量急剧增加、市场经济一体化以及客户的订单向着多品种小批量的方向发展。面对如此复杂的物流环境，仓内拣货工作发挥着越来越大的作用。拣货是指拣货员将用户所定的货物从储存位置取出，按用户分类集中、处理放置，为货物配送做准备。在多品种、小批量订单的配送活动中，人工拣货仍然是最重要的拣货方式，人工拣选时间约占整个配送作业的30%~40%，其成本约占总成本的90%。可见拣选作业的效率直接影响着配送中心的效率。

仓库作业主要包括采购、接货、验收、入库、在库 管理、拣货作业、包装、出库等环节，其中拣货作业是影响仓库效率的主要因素之一。[1]仓库商品出库的流程主要包括定位、组单、拣货、复核、打包等步骤。拣货作业是仓库作业的核心部分。尤其是在采取人至物的拣货方式的仓库中，拣货路径对仓库效率的影响尤为突出。如何能在短时间内得出最优的拣货路径以缩短拣货行走距离，成为现在研究者感兴趣的一个问题。对于物流管理来说，仓库作业是决定物流企业发展的重要因素。传统的劳动密集型仓库拣货作业，无论是投入的时间成本，还是人力成本都约占整个仓库作业成本 60%左右。[2]拣货作业的时间分为启动、寻找、行走、拣取等，其中行走时间约占拣货时间的 50%。[3]可见，缩短行走距离可以有效的减少拣货作业的时间。而所选的拣货路径对行走距离的长度和时间有着直接影响，因此，拣货路径的优化对于提高拣货作业效率乃至整个仓库的物流活动的效率都起着关键的作用。

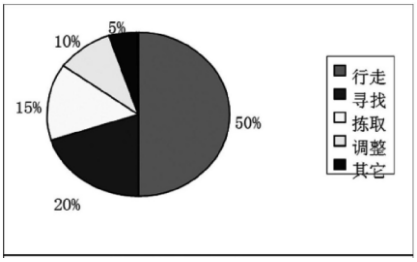


图 1-1 仓内作业时间占比图

关于拣货路径优化算法方面，相对于传统的拣货路径的策略，启发式算法更为简单易懂，灵活多变，结果也更为接近最优路径。李建斌等人将蚁群算法，禁忌算法，模拟退火算法与企业普遍使用的 S 型启发式策略进行对比，仿真结果表明，节约拣货时间约 13．35%。[4]王永波等人提出了蚁群一粒子群优化算法来解决基于 $RFID$ 的大型仓库拣货路径问题。该方法相对于单一的蚁群或粒子群算法有了较大改善，路径优化结果更为明显，实用性更强。[5] 马宪民等人提出了自适应视野的人工鱼群算法求解 最短路径问题，通过调整人工鱼觅食行为的视野，来 改进人工鱼群算法，并通过仿真结果表明改进后的 人工鱼群算法比基本鱼群算法和蚁群算法具有更高的收敛速度和精度。[6]

上述文献对拣货路径的策略和优化算法都做了较为细致的研究，但是很少涉及到在仓库的实际应用。本赛题的不同点在于拣货开始和结束的复核台可以不一致，即并非求闭合路径下的最短路径问题，这说明了上面的模型算法还存在不足之处。本论文结合升级改良的蚁群遗传算法，对仓库拣货路径进行最大程度的迭代优化，通过多重智能优化，得出最优拣货路径的策略。

第二章 问题重述与分析

2．1 数据分析

本论文基于附件提供的数据进行模型的建立和算法的设计求解。附件一仓库数据工作表中有四个表格，分别为货架、货格、复核台以及任务单的信息。包括货架、货格、复核台的位置及大小，货格和货架的关系。一个任务单包含多个订单，一个订单商品包含多个货格，一个货格需要拣多件商品货架的关系。

附件一的表格已经明确了货格和复核台相互之间的距离，为路径规划和优化模型的建立奠定了基础。货格和复合的坐标值是左下角的坐标，不能直接满足路径问题的求解条件，例如：拣货员是站在每个货格的中点位置前进行取货，在货格之间行走的距离也是基于中点之间的折线距离。所以我们将数据中货格和复合台的坐标调整为拣货员拣货位置（平面图上货格和复核台最近一条线的中点）。基于数据的坐标，我们建立二维平面的模型，并设计算法将计算 x 轴和 y 轴上距离D_x_{ij}，D_y_{ij}，为解决后面算法的设计做好铺垫。附加二与附件三是仓库的平面结构图，我们对仓库内货格与复核台进行编码后，简化了拣货员在搜索货格信息时的繁杂性，建立拣货路径规划的模型。

2．2 第一题分析

第一题是模型建立和算法设计的前提和基础。由于拣货员在仓库中拣货时，需要在货格之间、货格与复核台之间、复核台与复核台之间行走，所以必须要设计一种计算 3000 个货格和 13 个复核台总共 3013 个元素之间距离的方法。为了方便拣货员在拣货时货格的搜索，我们用编码的方式将 3013 个元素进行十进制数的编码 1~3013。距离的求解需要分别使用横轴和纵轴的坐标值，进而计算平面图上路径沿这两个方向的路程分量之和。

2．3 第二题分析

本题是已知起点，未知终点的动态路径优化问题。以复核台$FH10$为起点，假设所有复核台都正常工作，终点复核台未知，求解最终的出库时间。为解出一个任务单完成出库的时间，必须算出拣货员在仓库拣货的最短时间。我们利用第一题得到的距离矩阵，通过对原始名称进行编码得到任务单的货格、复核台之间对应的距离关系，设计出符合要求的距离矩阵，使用蚁群算法来最终求出拣货员的理想拣货路线，获得最短拣货路径。把最短路径通过速度-路程公式求出纯运动时间t_0，再算出取货所用时间t_1和复核打包时间t_3，则可算出最终的出库时间t_{out}。

2．4 第三题分析

本题是对第二题的拓展，限制可正常使用的复核台为$FH03$和$FH11$，任务单由 1 个拓展为 5 个，起始位置为复核台$FH03$，单人拣货，但仍是典型的动态路径优化问题。由

于拣货人员运动时间和拣货下架时间之和基本都是每个任务单复核打包的时间的1~2倍，所以每次拣货完成时，复核打包流程都已经结束，不存在复核台工作量堆积现象。本团队采用分而治之的思想，以每个任务单为研究对象，分别为每个任务单进行编码、设计出符合要求的距离矩阵，运用蚁群算法对其进行建模得到单个任务单的最短路径线路。直到将所有的任务单对应的最短路径线路都求出来之后，再运用分类组合的思想将所有路线尽可能的按“一个任务单理想路线的终点复核台位置为下一个任务单理想路线的起始复核台位置”来排布，从而得到理想的任务单领取顺序与总的理想拣货路线，进而求出总的纯运动路程，通过速度-路程公式求出总纯运动时间t_{all_0}，再算出总取货所用时间t_{all_1}总和最后一个任务单复核打包时间t_{all_2}，则可算出最终的出库时间t_{all_out}。

2．5 第四题分析

本题是对第三题的进一步拓展，可正常使用的复核台限制为4台，任务单个数为49份，拣货员增至9人。由于多人同时在一个货格拣货，是不考虑等待的时间，且下架完毕拣货员将拣货车送往某个复核台后拣货员无需等待，可继续领取拣货车和任务单，开始下一个任务单，所以每个拣货员的拣货过程都是独立的，每一个任务单仍是典型的路径优化问题。影响出库时间大小的因素主要有拣货员单个任务单理想路线长度、拣货人员在复核台的人数分配、以及拣货人员手中的任务单数量分配，在本题中，影响较大的是拣货人员在复核台的人数分配、以及拣货人员手中的任务单数量分配。由于本题多达49个任务单，数量很大，而复核台也多达4个，拣货人员更是多达9个之多，且初始、结束复核台均不知，因此实际算出精确值的算法路线是交错复杂的，而且也不必要。我们我采用合理简化模型的方法来求解：将每个任务单的拣货完成时间用一个定量均值来估计；考虑到本题的交叉复杂与随意性，人为限定拣货员的初始、终点复核台相同；由于复核台进行工作时，必定会出现任务单待复核打包堆积的现象的，所以要尽可能的平均每个复核台的工作量，使得复核台一旦开始工作必定运行直至完成出库。

出库时间的计算为单个复核台最大工作时间和第一个任务单的拣货完成时间之和。通过对每个拣货人员负责的任务单数和复核台的拣货人员人数进行最优化分配，即可求出最短出库时间，并求出各复核台利用率；通过求出每个任务单的拣货理想路径，即可完成总的理想路线的规划。

2．6 第五题分析

本题是对第四题的延续，需要考虑增加一个正常工作的复核台对最短出库时间的影响。进一步解读可得到，增加一个正常工作的复核台会使出库时间减小多少。通过第四题得到的简化模型的分析，通过调整每个拣货人员负责的任务单数和复核台的拣货人数，即可定性、定量地分析最短出库时间的变化。

2．7 第六题分析

我们从畅销品离复核台的位置与畅销品所在货架拥挤程度两个因素出发。根据距离复核台长短以及实际位置情况，我们把货架划分为四个区域：A区，B区，C区，D区。并从货架上畅销品占比在四个区域呈现逐次递减的趋势得到基本的货架商品摆放方案。

第三章 模型假设

3.1 我们认为，所有的数据来源于权威的原始数据，并且这些数据都是真实可靠的。
3.2 每个任务单之间是相互独立的，没有必然的联系。
3.3 每个拣货员的运动速度均是$1.5m/s$，包括折线拐弯处。
3.4 各题分析中复核台均不会出现宕机情况，能满足题目设定的商品复合使用要求。
3.5 假设拣货过程中不会出现人员拥挤而耽误时间的情况。
3.6 每个复核台的工作效率都相同，每个拣货员的工作效率也相同。
3.7 多个拣货员从一个复核台出发时，我们认为是同时的
3.8 我们不考虑货格与复核台的垂直距离。

第四章 符号约定与说明

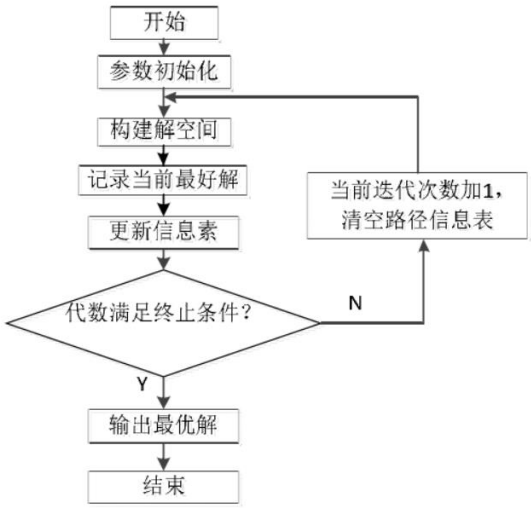
符号	说明
$\boldsymbol{\alpha}$	信息启发式因子
$\boldsymbol{\beta}$	期望启发式因子
$\boldsymbol{\rho}$	信息挥发因子
$\boldsymbol{m}$	蚁群数量
$\boldsymbol{t_0}$	拣货员运动时间
$\boldsymbol{t_1}$	拣货下架用时
$\boldsymbol{t_2}$	复核打包时间
$\boldsymbol{N}$	最大迭代次数
$\boldsymbol{\Delta\tau_{ij}^{k}}$	第k只蚂蚁在本次循环中留在路径D_{ij}的信息素
$\boldsymbol{P_i}$	仓库编码中第i 点对应货格或者复核台
$\boldsymbol{x_i}$	第i点物体的左下角横坐标
$\boldsymbol{y_i}$	第j点物体的左下角纵坐标
$\boldsymbol{D_{ij}}$	第i点物体与第j点物体的最短路径
$\boldsymbol{D\_x_{ij}}$	第i点物体与第j点物体在x轴上的距离
$\boldsymbol{D\_y_{ij}}$	第i点物体与第j点物体在y轴上的距离
$\boldsymbol{d}$	绕障碍物走横竖偏移量
$\boldsymbol{w_1}$	货格的宽度

第五章 模型建立

5．1 蚁群算法简介

自然界中的蚂蚁在觅食过程中，总能找到一条从蚁巢到事物源头的最优路径，原因在于，蚂蚁会在运动过程中所经过的路径上释放信息素，而后续的蚂蚁总能根据前面的蚂蚁留下的信息素选择要走的路径，路径上信息素的浓度越高代表这条路径被选择的次

数越多，而后续的蚂蚁选择这条路径的几率更大，整个过程形成了一个正反馈过程，从而逐渐逼近最优解，即蚁巢到食物源的最优路径。[7]受启发于自然界中蚂蚁的这种觅食行为，意大利学者 Dorigo 等人提出了基于蚂蚁种群的蚁群算法，并用该方法解决了一系列的组合优化问题。[8]蚁群算法的基本流程如下图表 5-1 所示：



图表 5-1 蚁群算法流程图

蚁群算法能很好地解决路径规划等优化问题，虽然求解的结果不一定是全局最优，但优化的效果往往能减少运算的复杂性，也是比较理想优化的结果。对于一般的优化问题，比如边数较少的TSP问题，规模相对较小，为求得问题最优解，可以采用精确算法求解。但是当问题规模较大时，算法的计算复杂度将大大增加，对于设计仿真以及实际部署，精确算法的求解性交比较低。这种问题上突出智能算法的优越性。对于本模型中，仓库货格数量的庞大性使问题难以用精确算法求解。本论文基于改良的蚁群算法和遗传算法的相互结合，对变种的最短路径优化问题进行求解，得到较好的路径规划结果。

5．2 蚁群算法的基本操作

蚁群算法主要操作包括设置参数、构建解空间、更新信息素、最优解的输出。

（1） 参数设置

在算法寻优之前需要对算法中的参数进行设置，如蚁群规模m，Q为信息素常系数，信息启发因子式α，期望启发式因子β，信息素挥发因子ρ，最大迭代次数N。如下图表 5-2 所示。

	定义	参数影响分析
α	信息启发式因子	α值越大，蚂蚁选择之前走过的路径可能性就越大，搜索路径的随机性减弱， α越小，蚁群搜索范围就会减少，容易陷入局部最优
β	期望启发式因子	β值越大，蚁群就越容易选择局部较短路径，这时算法的收敛速度是加快了，但是随机性却不高，容易得到局部的相对最优
m	蚁群数量	m数目越多，得到的最优解就越精确，但是会产生不少重复解，随着算法接近最优值的收敛，信息正反馈作用降低，大量的重复工作，消耗了资源，增加了时间复杂度
ρ	信息挥发因子，1－ρ表示残留因子	ρ过小时，在各路径上残留的信息素过多，导致无效的路径继续被搜索，影响到算法的收敛速率；ρ过大，无效的路径虽然可以被排除搜索，但是不能保证有效的路径也会被放弃搜索，影响到最优值的搜索

图表 5-2 蚁群算法参数含义图

蚁群算法中主要参数的选择无法判断是否最优，只能通过模型调参找到合适的解，参数设置如下图表 5-3 所示。

参数	信息启发因子式α	期望启发式因子β	信息素挥发因子ρ
取值范围	[0,5]	[0,5]	[0.2,0.5]

图表 5-3 参数范围取值参考表

（2） 构建解空间

在路径优化问题中，解空间的构成需要先将各个蚂蚁置于不同的出发点，对每个蚂蚁按照下式计算其下一个待访问的点位，知道所有蚂蚁访问完所有的点。每只蚂蚁对点的访问顺序构成了解空间。

$$P_{ij}^{k}=\begin{cases}\dfrac{\tau_{ij}^{\alpha}(t)*\eta_{ij}^{\beta}(t)}{\sum\tau_{ij}^{\alpha}(t)*\eta_{ij}^{\beta}(t)} & , \quad allowed_k=(0,1,\dots,n-1)\\ 0\end{cases}$$

式中，η_{ij}表示边弧D_{ij}的能见度，我们取$\eta_{ij}=\frac{1}{D_{ij}}$，$\alpha$表示轨迹的重要性，$\beta$表示能见度的重要性。

（3） 信息素更新

为避免残留信息素过多而淹没启发信息，在每只蚂蚁走完一步或者寻优结束都要对路径上的新信息素进行更新，按下式所示的信息素更新方程更新信息。

$$\tau_{ij}(t+n)=(1-\rho)*\tau_{ij}(t)+\Delta\tau_{ij}$$

式中，ρ我们可以表示为路径上的信息素的衰减度，则$1-\rho$表示消失程度。经过 n 时刻，蚂蚁完成一次周游，其路径上的信息素按照下式进行调整。

$$\Delta\tau_{ij}(t+n)=\sum_{k=1}^{m}\Delta\tau_{ij}^{k}$$

式中，$\Delta\tau_{ij}^{k}$表示第k只蚂蚁在本次循环中留在路径D_{ij}上的信息素，其计算公式如下所示，

$\Delta\tau_{ij}$表示在本次循环中路径D_{ij}上的信息素增量。

$$\Delta\tau_{ij}^{k}=\begin{cases}\dfrac{Q}{D_{ij}}, & \text{若第 k 只蚂蚁在本次循环中经过 } D_{ij}\\ 0, & \text{其他}\end{cases}$$

式中，Q为信息素常系数，D_{ij}表示第i点物体与第j点物体的最短路径。

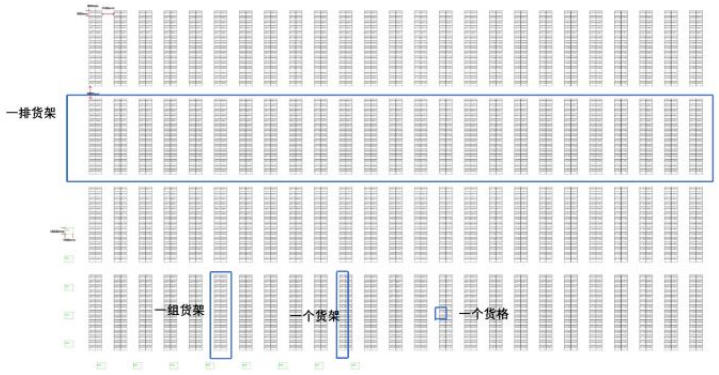
（4） 最优解的输出

每次迭代结束之后需要判断迭代次数是否满足终止条件，若满足转至寻优结束，输出最优解，否则继续进行迭代。

第六章 问题一解答

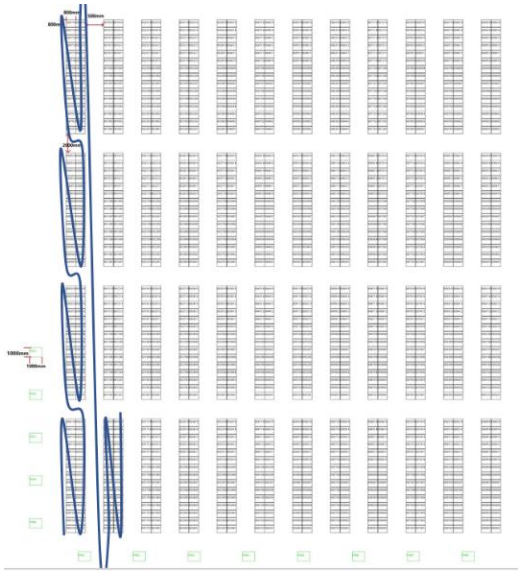
6．1 定义仓库货格编码

本题仓库有 13 个复核台，4 排货架，其中每排 25 组货架，每组 2 个货架，共 50 个货架，每个货架包含 15 个货格。对应含义直观表示为图 6-1。水平方向每组货架之间的距离为 1500 毫米，竖直方向相邻两排货架纵向距离为 2000 毫米，货格长宽都是 800 毫米，复核台长宽都是 1000 毫米。



图表 6-1 货架含义直观示意图

通过观察附件 1：仓库数据中的货格表格，我们发现货格从 S00101，S00102,一直到 S20015 对应的仓库货格走向如下图 5-2 所示。



图表 6-2 仓库货格走向图

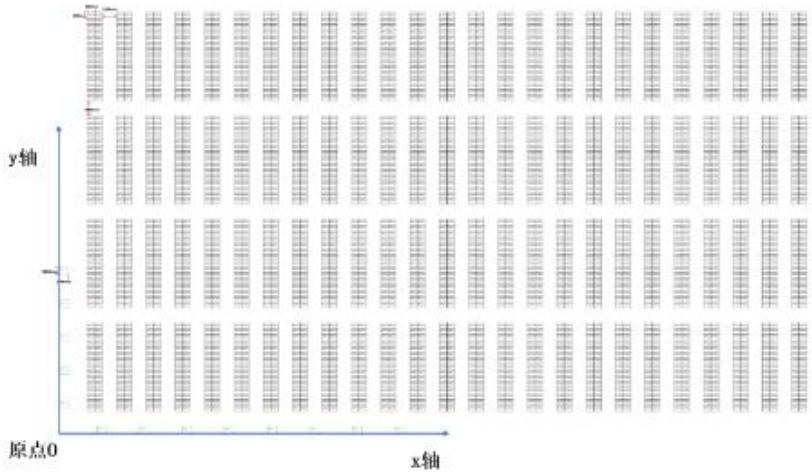
根据以上分析，我们得到货格以及复核台相对应的编码如下图 6-3 所示。详尽的从 1 到 3013 的编码对应在支撑材料：“**附件 1：货格与复核台编码.xlsx**”

名称	编码
S00101	1
S00102	2
S00103	3
S00104	4
S00105	5
S00106	6
S00107	7
S00108	8
S00109	9
S00110	10
S00111	11
S00112	12
S00113	13
S00114	14
S00115	15
S00201	16
S00202	17
S00203	18
…	…
S20008	2993
S20009	2994
S20010	2995
S20011	2996
S20012	2997
S20013	2998
S20014	2999
S20015	3000
FH01	3001
FH02	3002
FH03	3003
FH04	3004
FH05	3005
FH06	3006
FH07	3007
FH08	3008
FH09	3009
FH10	3010
FH11	3011
FH12	3012
FH13	3013

图表 6-3 货格与复核台编码表

6．2 建立坐标系

根据**“附件 1：仓库数据.xlsx”**，题目构建了以仓库下侧复核台下沿所在直线为 x 轴，仓库左侧复核台左沿所在直线为 y 轴，其对应的交点为原点 0 的二维平面直角坐标系。如下图 6-4 所示。



图表 6-4 二维平面直角坐标系图

6．3 生成距离矩阵并建立模型

本题中，我们记编码中第i点为P_i，记其第i点横坐标为x_i，第i点纵坐标为y_i。由此，我们记第i点P_i与第j点P_j两点最短路径为D_{ij}。同时，我们记货格宽度为w_1，复核台宽度为w_2，绕障碍物走横竖偏移量为d。得到下图表 6-5。

符号	含义	最小值	最大值
P_i	仓库编码中第i点对应货格或者复核台	1	3013
x_i	第i点物体的左下角横坐标	0	78200
y_i	第j点物体的左下角纵坐标	0	56200
D_{ij}	第i点物体与第j点物体的最短路径	0	D_{ijmax}
D_x_{ij}	第i点物体与第j点物体在x轴上的距离	0	78200
D_y_{ij}	第i点物体与第j点物体在y轴上的距离	0	56200
d	绕障碍物走横竖偏移量	750	750
w_1	货格的宽度	800	800
w_2	复核台的宽度	1000	1000

图表 6-5 距离等符号含义表

简单分析发现，货格与货格之间的最短路径计算和货格与复核台之间的最短路径计算方法存在差别，于是我们对两点距离计算进行了类别划分，如下表 6-6。

类别	含义	编码点的范围
1	货格P_i与货格P_j之间的最短路径D_{ij}	$1 \le i \le 3000$； $1 \le j \le 3000$
2	仓库下侧复核台P_i与货格P_j之间的最短路径D_{ij}	$3001 \le i \le 3008$； $1 \le j \le 3000$
3	仓库左侧复核台P_i与货格P_j之间的最短路径D_{ij}	$3009 \le i \le 3013$； $1 \le j \le 3000$
4	复核台P_i与复核台P_j之间的路径D_{ij}	$3001 \le i \le 3013$； $3001 \le j \le 3013$

图表 6-6 类别划分表

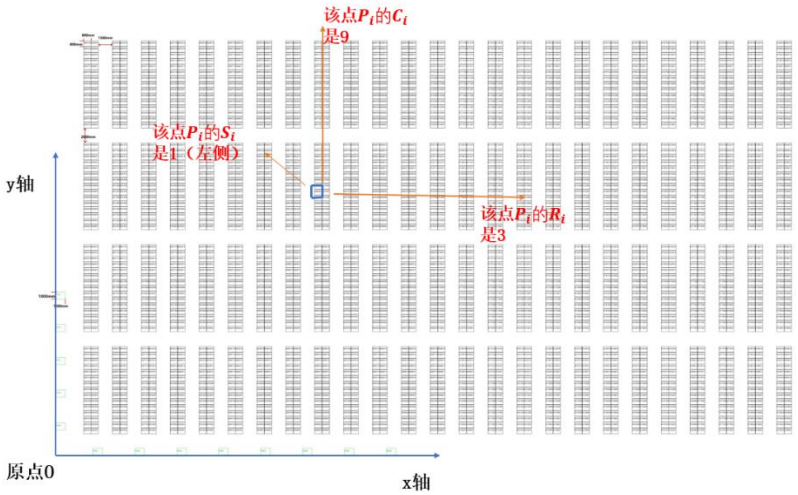
6.3.1 类别 1 最短路径计算

根据分析，货格P_i与货格P_j所在的货架的排数，货架组中的左右侧，以及纵向上所在组货架的列数等因素会对最短路径计算产生不同的影响。我们构建如下符号表 6-7 为后续的路径D_{ij}计算做准备。

符号	含义	最小值	最大值
R_i	第i点P_i所在的货架排数（Row）	1	4
S_i	第i点P_i所在左右侧（Side）（0 代表右侧，1 代表左侧）	0（右）	1（左）
C_i	纵向上所在组货架的列数（Column）	1	25

图表 6-7 第i点P_i所在的货架排数等符号表

为进一步理解三参数，我们列举具体例子如图表 6-8 所示。



图表 6-8 具体三参数例子图

我们得到货格P_i所在的货架的排数R_i为该点的编码i除于 30（一组货架的货格数）再向上取整（公式采用 ceil 表示）得到的数再对数字 4 做求余（公式采用%表示）。

$$R_i = \left\{ceil(\frac{i}{30})\right\}\%(4)$$

货格P_i所在左右侧S_i为点的编码i除于 15（一个货架的货格数）再向上取整（公式采用 ceil 表示）得到的数再对数字 2（一组货架有两个货架，分别是左侧，右侧）做求余（公式采用%表示）。当结果为 0，表示右侧；结果为 1，表示左侧。

$$S_i = \left\{ceil(\frac{i}{15})\right\}\%(2)$$

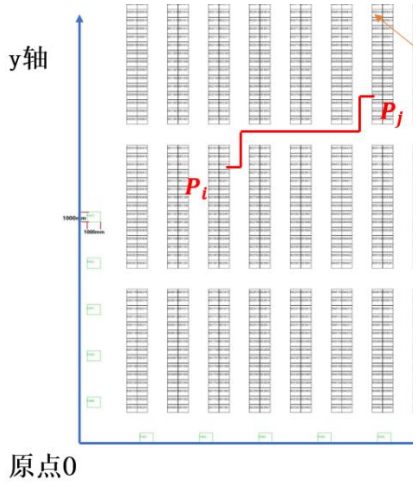
纵向上所在组货架的列数C_i为点的编码i除于 15（一个货架的货格数）再向上取整（公式采用 ceil 表示）得到的数再对数字 8（纵轴上一列货架有 8 个货架）做向上取整（公式采用 ceil 表示）。

$$C_i = \left\{ceil(\frac{\left\{ceil(\frac{i}{15})\right\}}{8})\right\}$$

6.3.1.1 货格P_i与货格P_j在不同排货架

当货格P_i与货格P_j在不同排货架时候，其在 y 轴上的距离分析较为简单，如下图 6-

9 所示。



图表 6-9 货格P_i与货格P_j在不同排货架的分析图

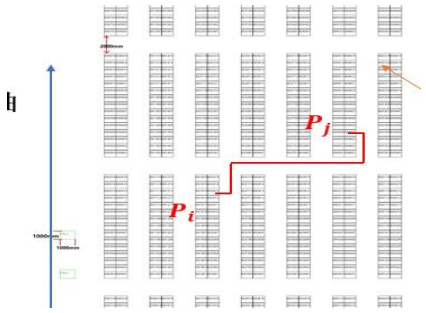
则得到，此两点在 y 轴上的距离

$$D\_y_{ij} = |(y_i + 400) - (y_j + 400)| = |(y_i) - (y_j)|$$

由于上公式中(x_i, y_i) 表示与编码i 对应的货格或者复核台的左下角的坐标，而我们是从货格P_i外侧中点出发，达到目标货格P_j外侧中点。所以有一个向上的$\frac{w_1}{2}$的偏移。

（1）货格P_i所在左右侧S_i与货格P_j所在左右侧S_j相等，如下图 5-9 所示。即

$$S_i = S_j = 1 \text{ 或者} S_i = S_j = 0$$



图表 6-10 $R_i \neq R_j$ $S_i = S_j$情况图

则得到，此两点在 x 轴上的距离 $D_x_{ij} = |(x_i) - (x_j)| + 2 * d$两点的最短路径为

$$D_{ij} = D\_x_{ij} + D\_y_{ij}$$

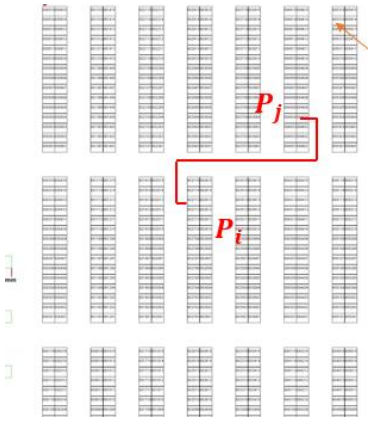
（2）货格P_i所在左右侧S_i与货格P_j所在左右侧S_j不相等，如下图 6-11 所示。即

（2）-1 货格P_i所在左右侧S_i在左边，与货格P_j所在左右侧S_j在右边。也即

S_i=1，S_i=0.

（2）-1-a 货格P_i纵向上所在组货架的列数（Column）$C_i \leq C_j$。情况如下

图 5-10

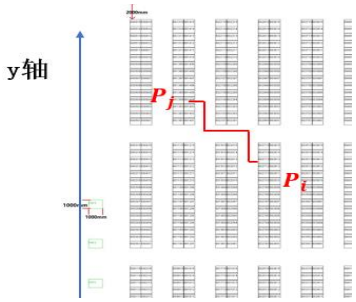


图表 6-11 $R_i \neq R_j$;S_i=1，$S_j = 0$;$C_i \leq C_j$示意图

则得到，此两点在 x 轴上的距离 $D\_x_{ij} = |(x_i) - (x_j)| + 4 * d + w_1$

两点的最短路径为$D_{ij} = D_x_{ij} + D_y_{ij}$

（2）-1-b 货格P_i纵向上所在组货架的列数（Column）$C_i > C_j$。情况如下图 6-12

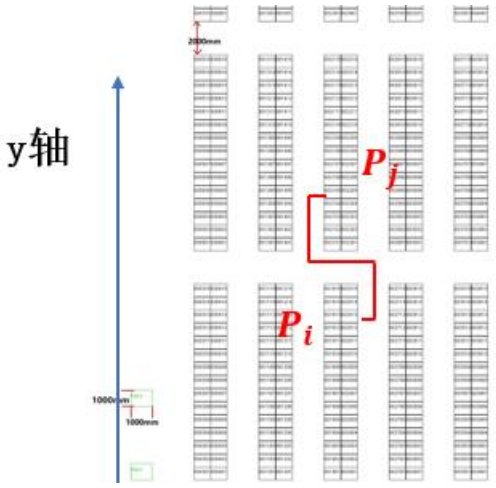


图表 6-12 $R_i \neq R_j$;S_i=1，$S_j = 0$;$C_i > C_j$示意图

则得到，此两点在 x 轴上的距离 $D_x_{ij} = |(x_i) - (x_j)| - w_1$两点的最短路径为$D_{ij} = D_x_{ij} + D_y_{ij}$

（2）-2 与（2）-1 有对应关系的， 货格P_i所在左右侧S_i在右边，与货格P_j所在左右侧S_j在左边。也即S_i=0，S_i=1.

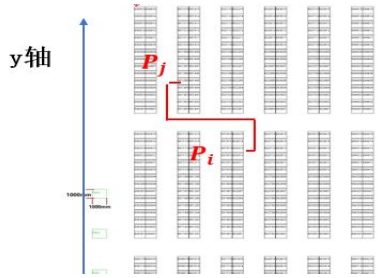
（2）-2-a 货格P_i纵向上所在组货架的列数（Column）$C_i \leq C_j$。情况如下图 6-13。



图表 6-13 $R_i \neq R_j$;S_i=0, $S_j = 1$;$C_i \leq C_j$示意图

则得到，此两点在 x 轴上的距离 $D_x_{ij} = |(x_i) - (x_j)| - w_1$两点的最短路径$D_{ij} = D_x_{ij} + D_y_{ij}$

（2）-2-b 货格P_i纵向上所在组货架的列数（Column）$C_i > C_j$。情况如下图 6-14



图表 6-14 $R_i \neq R_j$;S_i=0, $S_j = 1$;$C_i > C_j$示意图

则得到，此两点在 x 轴上的距离 $D_x_{ij} = |(x_i) - (x_j)| + 4 * d + w_1$两点的最短路径为$D_{ij} = D_x_{ij} + D_y_{ij}$

6.3.1.2 货格P_i与货格P_j在相同排货架 $R_i = R_j$

（1）当货格P_i与货格P_j在同一列，同一侧。也即S_i= S_j;$C_i = C_j$; $R_i = R_j$。

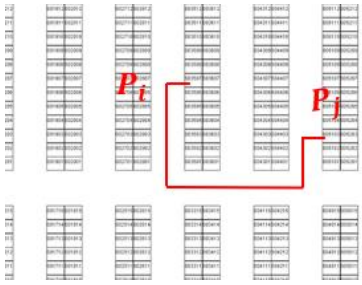
此两点在 x 轴上的距离 $D_x_{ij} = |(x_i) - (x_j)| = 0$此两点在 y 轴上的距离 $D\_y_{ij} = |(y_i) - (y_j)|$。两点的最短路径为$D_{ij} = D_x_{ij} + D_y_{ij}$

（2）当货格P_i与货格P_j在同一通道。也即$|(x_i) - (x_j)| = w_1 + 2 * d$=2300

此两点在 x 轴上的距离 $D_x_{ij} = |(x_i) - (x_j)| = 2300$此两点在 y 轴上的距离

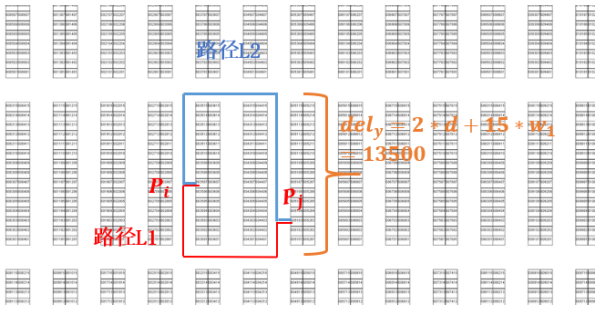
$D\_y_{ij} = |(y_i) - (y_j)|$，两点的最短路径为$D_{ij} = D_x_{ij} + D_y_{ij}$

（3）当货格P_i与货格P_j在相同排货架，但不在同一通道时，如下图 6-15 所示。



图表 6-15 $R_i = R_j$示意图

我们分析，发现当$R_i = R_j$时候，货格P_i与货格P_j之间明显有两条路径。如下图所示。



图表 6-16 $R_i = R_j$ 货格P_i与货格P_j路径图

我们发现$L1$，$L2$两条路径 y 轴上的距离之和是$2 * del_y = 2 * (2 * d + 15 * w_1) = 27000$。由此，我们可以得到 P_i 到 P_j 的 y 轴上的距离$D_y_{ij} = \min(L1, L2)$ 。我们记P_i往下路径 L1 所走的距离$tempy_i$ ，P_j往下路径 L2 所走的距离$tempy_j$。

$$tempy_i = (y_i + \frac{w_1}{2}) - (2250 + 14000 * (R_i - 1))$$

$$tempy_j = (y_j + \frac{w_1}{2}) - (2250 + 14000 * (R_j - 1))$$

（3）-1 由此，$L1 = tempy_i + tempy_j$。当$L1 < \frac{2*del_y}{2}$时，从P_i 到 P_j 采用 L1 路径。

此两点在 y 轴上的距离

$$D\_y_{ij} = L1 = tempy_i + tempy_j$$
$$= \left(y_i + \frac{w_1}{2}\right) - (2250 + 14000 * (R_i - 1)) + (y_j + \frac{w_1}{2}) - (2250 + 14000 * (R_j - 1))$$

对于其从P_i 到 P_j 两点在 x 轴上的距离计算，情况如同 5.1.3.1 货格P_i与货格P_j

在不同排货架的分类分析一致，不再赘述。

（3）-2 由此，$L1 = tempy_i + tempy_j$。当$L1 \geq \frac{2*del_y}{2}$时，从$P_i$ 到 P_j 采用 L2 路径。

此两点在 y 轴上的距离

$$D\_y_{ij} = 2 * del_y - L1 = 2 * del_y - (tempy_i + tempy_j)$$

$$D\_y_{ij} = 2 * (2 * d + 15 * w_1) - \left\{\left(y_i + \frac{w_1}{2}\right) - (2250 + 14000 * (R_i - 1)) + \left(y_j + \frac{w_1}{2}\right) - (2250 + 14000 * (R_j - 1))\right\}$$

对于其从P_i 到 P_j 两点在 x 轴上的距离计算，情况如同 5.1.3.1.1 货格P_i与货格P_j在不同排货架的分类分析一致，不再赘述。

6.3.1.3 综合以上，得到货格P_i与货格P_j的最短路径$D_{ij} = D_x_{ij} + D_y_{ij}$如下。

$\|(x_i) - (x_j)\| + 2 * d + \|(y_i) - (y_j)\|$	$R_i \neq R_j; S_i = S_j$	1
$\|(x_i) - (x_j)\| + 4 * d + w_1 + \|(y_i) - (y_j)\|$	$R_i \neq R_j; S_i = 1, S_j = 0; C_i \leq C_j$	2
$\|(x_i) - (x_j)\| - w_1 + \|(y_i) - (y_j)\|$	$R_i \neq R_j; S_i = 1, S_j = 0; C_i > C_j$	3
$\|(x_i) - (x_j)\| - w_1 + \|(y_i) - (y_j)\|$	$R_i \neq R_j; S_i = 0, S_j = 1; C_i \leq C_j$	4
$\|(x_i) - (x_j)\| + 4 * d + w_1 + \|(y_i) - (y_j)\|$	$R_i \neq R_j; S_i = 1, S_j = 0; C_i > C_j$	5
$\|(y_i) - (y_j)\|$	$R_i = R_j; S_i = S_j; C_i = C_j$	6
$2 * d + \|(y_i) - (y_j)\|$	$R_i = R_j; \|(x_i) - (x_j)\| = w_1 + 2 * d$	7
$\|(x_i) - (x_j)\| + 2 * d + L1$	$R_i = R_j$; 不在同一通道; $L1 < \frac{2*del_y}{2}$; $S_i = S_j$	8
$\|(x_i) - (x_j)\| + 4 * d + w_1 + L1$	$R_i = R_j$; 不在同一通道; $L1 < \frac{2*del_y}{2}$; $S_i = 1, S_j = 0; C_i \leq C_j$	9
$\|(x_i) - (x_j)\| - w_1 + L1$	$R_i = R_j$; 不在同一通道; $L1 < \frac{2*del_y}{2}$; $S_i = 1, S_j = 0; C_i > C_j$	10
$\|(x_i) - (x_j)\| - w_1 + L1$	$R_i = R_j$; 不在同一通道; $L1 < \frac{2*del_y}{2}$; $S_i = 0, S_j = 1; C_i \leq C_j$	11
$\|(x_i) - (x_j)\| + 4 * d + w_1 + L1$	$R_i = R_j$; 不在同一通道; $L1 < \frac{2*del_y}{2}$; $S_i = 1, S_j = 0; C_i > C_j$	12
$\|(x_i) - (x_j)\| + 2 * d + (2 * del_y - L1)$	$R_i = R_j$; 不在同一通道; $L1 \geq \frac{2*del_y}{2}$; $S_i = S_j$	13
$\|(x_i) - (x_j)\| + 4 * d + w_1 + (2 * del_y - L1)$	$R_i = R_j$; 不在同一通道; $L1 \geq \frac{2*del_y}{2}$; $S_i = 1, S_j = 0; C_i \leq C_j$	14
$\|(x_i) - (x_j)\| - w_1 + (2 * del_y - L1)$	$R_i = R_j$; 不在同一通道; $L1 \geq \frac{2*del_y}{2}$; $S_i = 1, S_j = 0; C_i > C_j$	15
$\|(x_i) - (x_j)\| - w_1 + (2 * del_y - L1)$	$R_i = R_j$; 不在同一通道; $L1 \geq \frac{2*del_y}{2}$; $S_i = 0, S_j = 1; C_i \leq C_j$	16
$\|(x_i) - (x_j)\| + 4 * d + w_1 + (2 * del_y - L1)$	$R_i = R_j$; 不在同一通道; $L1 \geq \frac{2*del_y}{2}$; $S_i = 1, S_j = 0; C_i > C_j$	17

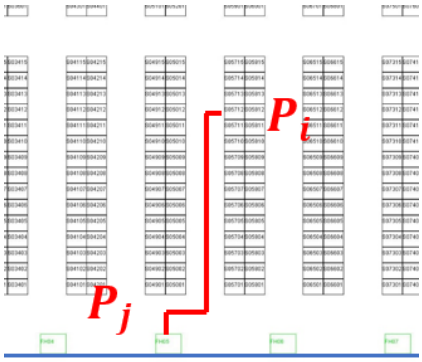
图表 6-17 货格P_i与货格P_j最短路径$D_{ij} = D_x_{ij} + D_y_{ij}$公式图

其中，$L1 = tempy_i + tempy_j = \left(y_i + \frac{w_1}{2}\right) - (2250 + 14000 * (R_i - 1)) + (y_j + \frac{w_1}{2}) - (2250 + 14000 * (R_j - 1))$

6.3.2 类别 2 最短路径计算

我们知道，仓库下侧复核台P_i与货格P_j之间的最短路径D_{ij} 。其复核台P_i与货格P_j在

y 轴上的距离为 $D\_y_{ij} = \left(y_j + \frac{w_1}{2}\right) - (y_i + w_2)$。其模型如下图表 6-18 所示。



图表 6-18 复核台P_i与货格P_j路径示意图

在 x 轴上的距离分析与货格之间的分析一致。

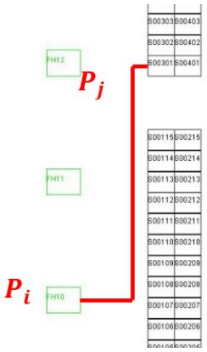
6.3.2.3 综合以上，得到复核台P_i与货格P_j的最短路径$D_{ij} = D_x_{ij} + D_y_{ij}$如下。

$$(x_j) - \left(x_i + \frac{w_2}{2}\right) + \left(y_j + \frac{w_1}{2}\right) - (y_i + w_2)$$
$$\left(x_i + \frac{w_2}{2}\right) - (x_j) + 2 * d + \left(y_j + \frac{w_1}{2}\right) - (y_i + w_2)$$
$$(x_j + d + w_1) - \left(x_i + \frac{w_2}{2}\right) + d + w_1 + \left(y_j + \frac{w_1}{2}\right) - (y_i + w_2)$$
$$\left(x_i + \frac{w_2}{2}\right) - x_j - d$$

$$S_j = 1; (x_j - d) \geq (x_i + \frac{w_2}{2}) \quad 1$$
$$S_j = 1; (x_j - d) < (x_i + \frac{w_2}{2}) \quad 2$$
$$S_j = 0; (x_j + d + w_1) \geq (x_i + \frac{w_2}{2}) \quad 3$$
$$S_j = 0; (x_j + d + w_1) < (x_i + \frac{w_2}{2}) \quad 4$$

6．3．3 类别 3 最短路径计算

6.3.3.1 当 货格P_j在第一列组货架而且在左侧时候，即$S_j = 1; C_j = 1$, 其示意图如下 5-18 $S_j = 1; C_j = 1$, 复核台P_i与货格P_j路径示意图。



图表 6-19 $S_j = 1; C_j = 1$, 复核台P_i与货格P_j路径示意图

此两点在 x 轴上的距离

$$D\_x_{ij} = (x_j) - (x_i + w_2) = 2000$$

此两点在 y 轴上的距离

$$D\_y_{ij} = \left(y_j + \frac{w_1}{2}\right) - (y_i + \frac{w_2}{2})$$

两点的最短路径为

$$D_{ij} = D\_x_{ij} + D\_y_{ij}$$

6.3.3.2 当 货格P_j在第一排组货架而且复核台P_i在$3009 \le i \le 3011$。示意图 6-20 复核台P_i在$3009 \le i \le 3011$, 复核台P_i与货格P_j路径示意图.

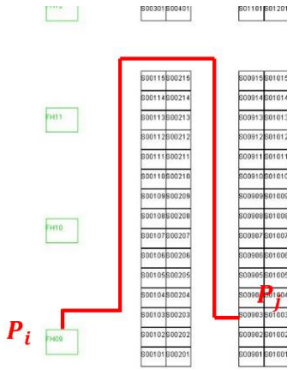
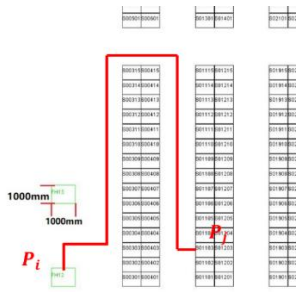


图 6-20 复核台P_i在$3009 \le i \le 3011$, 复核台P_i与货格P_j路径示意图

该分析与上面货格P_i与货格P_j在同一行的分析一致，其中$L1$，$L2$两条路径 y 轴上的距离之和是$2 * del_{y_{desk}} = 2 * (2 * d + 15 * w_1) - w_2 = 26000$。由此，我们可以得到 P_i 到 P_j 的 y 轴上的距离$D_y_{ij} = \min(L1, L2)$ 。我们记P_i往下路径 L1 所走的距离 $tempy_i$ ，P_j往下路径 L2 所走的距离$tempy_j$。篇幅有限，不再赘述。

6.3.3.3 当 货格P_j在第二排组货架而且复核台P_i在$3012 \le i \le 3013$。示意图 5-20 复核台P_i在$3012 \le i \le 3013$, 复核台P_i与货格P_j路径示意图.



图表 6-21 复核台P_i在$3012 \le i \le 3013$, 复核台P_i与货格P_j路径示意图

该分析与上面货格P_i与货格P_j在同一行的分析一致。详尽见下总结公式。

6．3．3．4 对于复核台P_i与货格P_j路径的其余情况如货格P_j 与货格P_i在不同行（$R_i \neq R_j$）的分析一致。不再赘述。

6．3．4　对于复核台P_i与复核台P_j路径

此两点在 x 轴上的距离 $D\_x_{ij} = |(x_i) - (x_j)|$ 此两点在 y 轴上的距离 $D\_y_{ij} = |(y_i) - (y_j)|$。两点的最短路径为$D_{ij} = D_x_{ij} + D_y_{ij}$。

6．4 生成距离矩阵D

综合以上分析，我们分四大类别得到对应的最短路径公式。

生成以下物体P_i与物体P_j距离D矩阵。

编码	P_1	P_2	P_3	…	P_i	…	P_{3013}
P_1	D_{11}	D_{12}	D_{13}	…	D_{1i}	…	$D_{1,3013}$
P_2	D_{21}	D_{22}	D_{23}	…	D_{2i}	…	$D_{2,3013}$
P_3	D_{31}	D_{32}	D_{33}	…	D_{3i}	…	$D_{3,3013}$
…	…	…	…	…	…	…	…
P_i	D_{i1}	D_{i2}	D_{i3}	…	D_{ii}	…	$D_{i,3013}$
…	…	…	…	…	…	…	…
P_{3013}	$D_{3013,1}$	$D_{3013,2}$	$D_{3013,3}$	…	$D_{3013,i}$	…	$D_{3013,3013}$

图表 6-22 物体P_i与物体P_j距离矩阵D示意图

对应计算结果如下图 6-23 物体P_i与物体P_i距离矩阵D表格，详尽的结果请查看“附件 4:计算结果.xlsx”中 sheet1 的 Ques1。

编码	P_1	P_2	P_3	…	P_i	…	P_{3013}
P_1	0	2300	3100	…	D_{1i}	…	20600
P_2	2300	0	2300	…	D_{2i}	…	19800
P_3	3100	2300	0	…	D_{3i}	…	19000
…	…	…	…	…	…	…	…
P_i	D_{i1}	D_{i2}	D_{i3}	…	0	…	$D_{i,3013}$
…	…	…	…	…	…	…	…
P_{3013}	20600	19800	19000	…	$D_{3013,i}$	…	0

图表 6-23 物体P_i与物体P_j距离矩阵D图表格

第七章 问题二解答

7．1 获取距离矩阵$\boldsymbol{D}$

对题目要求的任务单$T0001$的商品货格进行编码，通过得到的编码利用题目一的计算距离的方法得到的距离矩阵进行索引，选出目标货格与复核台间的距离关系，设计出只包含货格的距离矩阵D。

7．2 获取理想路线

利用蚁群算法建立动态优化模型，将设计好的距离矩阵D输入到优化模型，进行优化迭代，最终得到一个最短拣货路线回路，其长度记为c_0。理想路线的求解划分为四个部分。

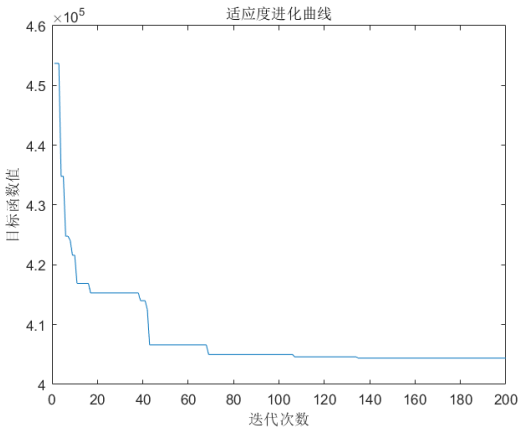
第一步，计算起始复核台$FH10$ 到与它最近的一个货格的距离D_{FH10_amin}，标记该货格为P_a；

第二步，利用距离矩阵D，运用 Matlab 工具查询出离货格P_a最近的两个货格，分别记为P_b, P_c。以此得到P_b, P_c到所有复核台的最小距离，分别记录为D_{FH_bmin}， D_{FH_cmin}。

第三步，计算$(D_{ab} - D_{FH_bmin})$以及$(D_{ac} - D_{FH_cmin})$的值。取大值的货格所在点（b或者c）为该任务单优化拣货路线的终点复核台的邻接点。

第四步，经过使用 Matlab 仿真基于改进型蚁群算法求解迭代，我们得到理想拣货路线的长度$L = \{D_{FH10_amin} + c_0 - max\{(D_{ab} - D_{FH_{bmin}}), (D_{ac} - D_{FH_cmin})\}\}$。

其中，程序迭代过程的适应度进化曲线如图 7-1 所示。



图表 7-1 Matlab 仿真迭代过程适应度进化曲线图

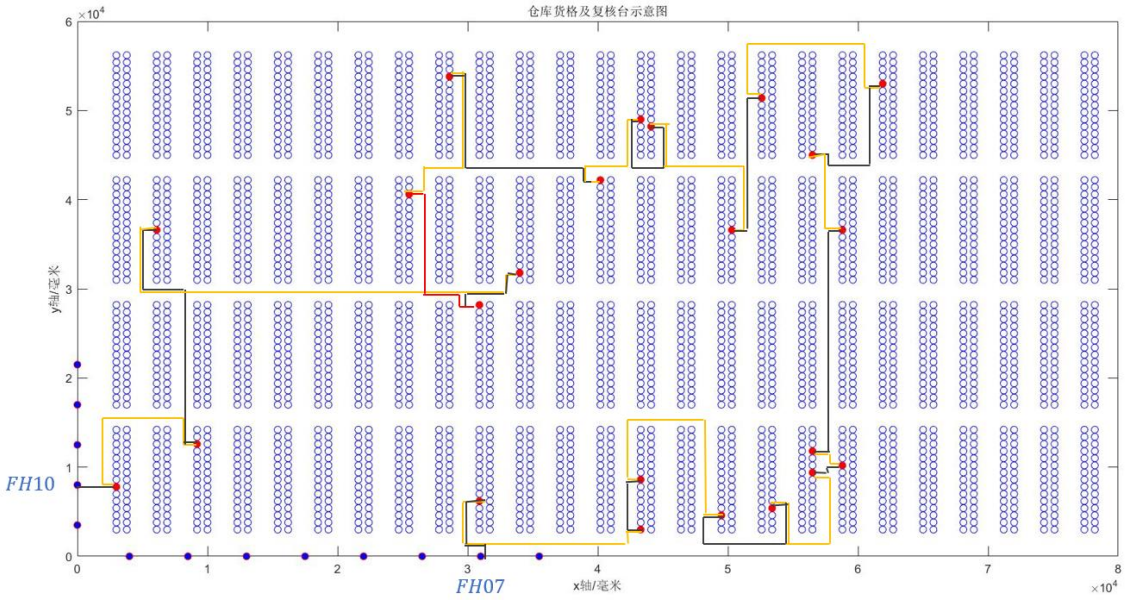
由上图，可以知道本次完成任务单$T0001$拣货路线规划仿真中，其改进型蚁群算法具有良好的迭代收敛效果。

多次考量及实验仿真后我们得到以下任务单$T0001$拣货路线。

$FH10 \rightarrow S00107 \rightarrow S01713 \rightarrow S01308 \rightarrow S08502 \rightarrow S07515 \rightarrow S06213 \rightarrow S07212 \rightarrow S10115 \rightarrow S11106 \rightarrow S11205 \rightarrow S12608 \rightarrow S13509 \rightarrow S15911 \rightarrow S14401 \rightarrow S14908 \rightarrow S13812 \rightarrow S14510 \rightarrow S13809 \rightarrow S13004 \rightarrow S12103 \rightarrow S10508 \rightarrow S10501 \rightarrow S07305 \rightarrow$

$FH07$。

将其任务单$T0001$拣货路线可视化，如下图 7-2 所示。



图表 7-2 任务单$T0001$拣货路线

7．3 求取出库时间

任务单$T0001$拣货理想路线长度$L=\{D_{FH10_amin}+c_0-max\{(D_{ab}-D_{FH_{bmin}}),(D_{ac}-D_{FH_cmin})\}\}=372200mm$。由速度-路程公式得知，拣货员运动时间$t_0=\frac{L}{v}=248.13\,s$，拣货下架用时$t_1=177\,s$，复核打包总时间$t_2=300\,s$，则出库时间$t_{out}=t_0+t_1+t_2=725.13\,s\approx 12.09\,min$。

第八章 问题三解答

8．1 模型再分析

由于每一个任务单都是独立的，拣货员不能同时进行多个任务单。且下架完毕拣货员将拣货车送往某个复核台后拣货员无需等待，可继续领取拣货车和任务单，开始下一个任务单，所以当一个拣货员的工作量有多个任务单时，他执行每一个任务单的过程都是独立的，可把多个任务单的总的工作进程分成每个任务单进程之和。

在对任务单先后的规划中，本题遵循两大排布原则。

一，一个任务单理想路线的终点复核台位置为下一个任务单理想路线的起始复核台位置。

二，复核台工作量尽可能均匀。

经过我们详尽分析，推演举例计算。我们发现拣货人员运动时间和拣货下架时间之

和基本都是每个任务单复核打包的时间的 1~2 倍，这一基本规律。因而，每次拣货完成时，复核打包流程都已经结束，不存在复核台工作量堆积现象。按照“一个任务单理想路线的终点复核台位置为下一个任务单理想路线的起始复核台位置”的原则，来对求出的单任务单理想路线进行排布顺序。我们设计以下解决模型方案。

8．1．1 求解单个任务单的理想路线

记当下求解的任务单的编号为x，则对于任务单x拣货路线规划分析如下。

（1）获取距离矩阵D

对题目要求的每个复核台、第i任务单的商品货格进行编码，通过得到的编码利用题目一 3013*3013 距离矩阵D_{ij}得到的本题所需货格和复核台的索引，选出目标货格、复核台间的距离关系，得到只包含货格的距离矩阵D_{ij}；

（2）获取理想路线

利用蚁群算法建立动态优化模型，将设计好的距离矩阵D输入到优化模型，进行优化迭代，最终得到一个最短拣货路线回路，其长度记为c_{0_x}。理想路线的求解划分为四个部分。

第一步，计算起始复核台$FH10$ 到与它最近的一个货格的距离 $D_{FH10\_amin}^{\ x}$，标记该货格为P_a^x；

第二步，利用距离矩阵D，运用 Matlab 工具查询出离货格P_a^x最近的两个货格，分别记为P_b^x，P_c^x。以此得到P_b^x，P_c^x到所有复核台的最小距离，分别记录为$D_{FH_bmin}^{\ x}$，$D_{FH\_cmin}^{\ x}$。

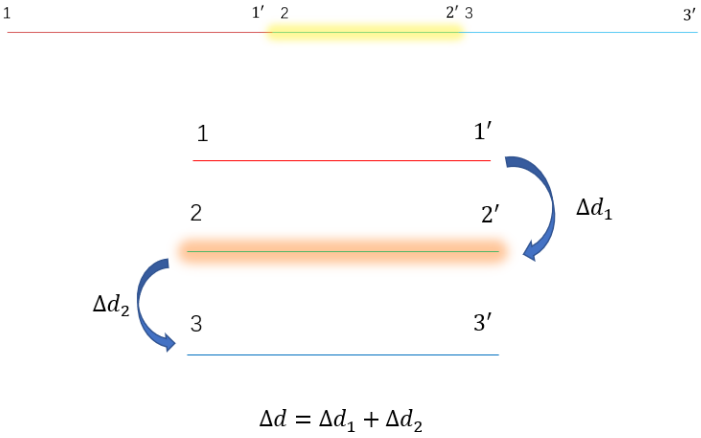
第三步，计算$(D_{ab} - D_{FH_bmin}^{\ x})$以及$(D_{ac} - D_{FH_cmin}^{\ x})$的值。取大值的货格所在点（$b$或者$c$）为该任务单优化拣货路线的终点复核台的邻接点。

第四步，经过使用 Matlab 仿真基于改进型蚁群算法求解迭代，我们得到理想拣货路线的长度$L_x = \left\{D_{FH10_amin}^{\ x} + c_{0_x} - max\{\left(D_{ab} - D_{FH_bmin}^{\ x}\right), (D_{ac} - D_{FH_cmin}^{\ x})\}\right\}$。

求取单拣货完成时间：$t_0^x = \frac{L_x}{v}$，拣货下架时间t_1^x，单次任务单拣货完成时间$t_{out}^x = t_0^x + t_1^x$。

（3）计算所有任务单的理想路线

将所有单次任务单理想路线罗列出来，按照“一个任务单理想路线的终点复核台位置为下一个任务单理想路线的起始复核台位置”和“复核台工作量尽可能均匀”的原则来排布，尽可能的排布连接成一条链；若出现多条链，则需要在理想路线长度加入每一条链结束端到另外一条链起始端的路程的总和Δd，如下图 8-1 所示。出库时间将变长。进而，根据排布原则得到任务单的先后划分以及总的理想路线S。



图表 8-1 Δd等效图

（4）计算出库总时间与复核台利用率

在复核台工作量尽可能均匀的情况下，复核台除最后一个任务单的复核打包时间外，其他复核打包时间已包含在了拣货员拣货的时间中了，则出库时间与最后一个的任务单的复核打包时间有关。每个复核台总工作时间为t_{fi}，最后一个任务单所需要的复核打包时间为t_2'，则出库总时间$t_{all_out} = \sum_{x=0}^{5} t_0^x + \frac{\Delta d}{v} + t_2'$，单个复核台利用率$\eta_i = \frac{t_{fi}}{t_{all_out}}$。

8．2 模型效果分析

经过 Matlab 仿真分析，我们得到拣货员P任务单领取顺序为

$$T0005 \rightarrow T0004 \rightarrow T0002 \rightarrow T0006 \rightarrow T0003$$

同时我们得到其理想拣货路线为：

FH03→S02505→S02104→S02301→S03104→S03202→S05514→S07912→S06214→S07712→S07608
→S10103→S14204→S14701→S13007→S13105→S12213→S11308→S09703→S06004→S06515→S05815
→S03405→FH03→S02604→S05903→S05812→S07604→S08106→S07402→S09704→S09009→S12307
→S15310→S14612→S14706→S14210→S14404→S11904→S10913→S10203→S10406→S09613→S09508
→S07710→S05510→S03114→S02903→S02706→S01815→FH11→S01115→S02911→S03013→S05314
→S06303→S06910→S06413→S07208→S08705→S16008→S15702→S15408→S14615→S13915→S11610
→S11410→S12101→S10604→S09805→S09807→S09210→S07612→S08303→S05715→S05111→S06101
→FH11→S00414→S03510→S03611→S03601→S04410→S06814→S09301→S10814→S10709→S09010
→S11311→S12113→S12305→S03510→S14506→S15408→S13408→S13314→S12707→S06912→S06213
→S03203→S04010→S03214→S02311→FH03→S07301→S09704→S12901→S12315→S12911→S12315
→S16004→S14315→S04711→S02315→S00615→S02913→S04511→S06403→S08804→S07807→S09207
→S08403→S08302→S08106→FH03

其中任务台$FH03$工作时间：t_{f1}=1110s；任务台$FH11$工作时间：t_{f2}=840s。

任务单拣货完成总时间$\sum_{x=0}^{5} t_0^x + \frac{\Delta d}{v}$=1109.87$s$ + 1262s = 2371.87s。

最后一次任务单复核打包时间：t_2'=330s。

出库时间$t_{all_out} = \sum_{x=0}^{5} t_0^x + \frac{\Delta d}{v} + t_2'$=2701.87$s$。

任务台$FH03$利用率$\eta_1 = \frac{t_{f1}}{t_{all_out}}$=41.08%

任务台$FH11$利用率$\eta_2 = \frac{t_{f2}}{t_{all_out}}$=31.09%

第九章 问题四解答

9．1 模型再分析

本题可正常使用复核台个数为 4 个，任务单数量为 49 份，拣货人员为 9 人，初始复核台位置、终点复核台位置均未知。

影响出库时间大小的因素主要有拣货员单个任务单理想路线长度、拣货人员在复核台的人数分配、以及拣货人员手中的任务单数量分配，在本题中，影响较大的是拣货人员在复核台的人数分配、以及拣货人员手中的任务单数量分配。由于本题多达 49 个任务单，数量很大，而复核台也多达 4 个，拣货人员更是多达 9 个之多，且初始、结束复核台均不知，因此实际算出精确值的算法路线是交错复杂的，而且也不必要。因此，我采用合理简化的方法来求解，模型简化如下：

（1）对于一个拣货人员、单个任务单时，每次执行拣货理想路线的时间是不同的；对于一个拣货人员、多个任务单时，每个任务单的拣货理想路线的时间是不同的，但执行下来的总的时间为一个大概的定值，所以可以通过计算所有的任务单的拣货理想路线的时间均值，再乘以任务单数，也可得到总的时间定值；同理，对于多拣货员、多任务单时，也可以简化为求出一个定量均值来代表所有任务单的拣货理想路线用时的平均水平，进而估计所有任务单完成拣货的总时间。因此，本模型通过一个定量均值τ来代替每一个任务单的拣货完成时间，最终得到估计的拣货完成总时间与真实值的误差可以忽略。

（2）在有三个及以上的可正常使用的复核台时，复核台之间会往复充当初始复核台、终点复核台，再加上若有多个人执行拣货操作时，交错往复现象更为明显，具有明显的随机性，所以，当简单的规定拣货路线的初始、终点复核台都一样时，求取的时间估计值与真实值之间的误差是可以承受的。

（3）由于复核台进行工作时，是必定会出现任务单待复核打包堆积的现象的，所以要尽可能的平均每个复核台的工作量，也即订单数。因此需要对复核台上拣货人员的人数和每位拣货人员任务单数量、单号选择进行规划。

9．2 模型优化

9.2.1 重要参数确定

（1）分配每个拣货人员负责的任务单和复核台的拣货人员人数

将 49 个任务单，按照订单数的数量大小以降序的方式进行排序，得到数据梯度表格 9-1：

行标签	计数项:订单号
T0035	31
T0026	29
T0016	29
T0017	29
T0028	28
T0008	27
T0029	27
T0040	27
T0038	27
T0011	27
T0048	27
T0020	27
T0004	26
T0014	26
T0047	26
T0022	26
T0002	26
T0025	26
T0006	25
T0044	25
T0039	25
T0024	25
T0045	25
T0007	24
T0012	24
T0031	24
T0036	24
T0032	24
T0034	24
T0013	23
T0043	23
T0027	23
T0010	23
T0021	23
T0001	23
T0018	23
T0042	22
T0019	22
T0037	22
T0023	22
T0030	22
T0041	22
T0015	22
T0005	22
T0049	21
T0009	21
T0046	21
T0033	20
T0003	20
总计	1200

图表 9-1 任务单排序表

因为共有 9 位拣货人员，4 个可正常工作复核台，所以平均应 5 位拣货人员负责 5 份任务单，4 位拣货人员负责 6 份任务单；每个复核台应有 2~3 位拣货人员。根据模型简化方式知：每位拣货人员离开初始复核台之后，必定会回到起点复核台，所以**每个复核台的堆积现象是明显的**。一旦复核台开始工作，复核台必然不会停止直至所有任务单都复核打包完，所以总的出库时间只与第一个任务单的拣货理想路线完成时间以及单个复核台最大工作时间有关。而由模型简化处理规则知：每一个任务单拣货完成时间都用一个定量均值t_{mean}表示，所以此时要使出库时间最短，只需要调整四个复核台工作时间中最大时间到最小。为了让每个复核台工作时间均衡以及四个复核台中复核订单数最多的复核台的复核时间最小，就只需考虑拥有每个复核台上拣货人员数量和每个拣货人员拥有的任务单数。原理定性分析如下：拣货员执行一次任务单的时间之差一般为 10 秒左右，复核台复核的任务单每相差一个就相差 450 秒左右的时间，每个复核台拣货员数量相差一个也会导致的总时间相差为复核的任务单时间的数量级。

由此可得，本题影响出库主要因素是分配方式的差别。下面专注于拣货人员和任务单的分配。

我们把数据梯度表格的倒数 15 个任务单分配给前 3 位拣货人员，第 6、7、8、9 位

拣货人员分配得到数据梯度表格的前 24 个任务单，其他人平均分配剩余的 10 个任务单，并让前 3 位拣货人员都安置在同一个复核台，第 4、6 位拣货人员安置于同一个复核台，第 5、7 位拣货人员安置于同一个复核台，最后剩余拣货人员安置于同一个复核台。以上即可得到最短出库时间，并求出每个复核台利用率。本题用以下参数：

拣货员编号	复核台	任务单
1	FH01	T0049，T0009，T0046，T0033，T0003
2	FH01	T0023，T0030，T0041，T0015，T0005
3	FH01	T0001，T0018，T0042，T0019，T0037
4	FH03	T0012，T0031，T0036，T0032，T0034
5	FH10	T0013，T0043，T0027，T0010，T0021
6	FH03	T0035，T0026，T0016，T0017，T0028，T0008
7	FH10	T0029，T0040，T0038，T0011，T0048，T0020
8	FH12	T0004，T0014，T0047，T0022，T0002，T0025
9	FH12	T0006，T0044，T0039，T0024，T0045，T0007

图表 9-2 拣货员复核台任务单等对应分配表格

9．2．3 模型解答

（1）获取距离矩阵D_j

对题目要求的每个复核台、第i任务单的商品货格进行编码，通过得到的编码利用题目一的计算距离的方法得到的距离矩阵进行索引，选出目标货格、复核台间的距离关系，设计出包含货格、初始复核台、终点复核台的距离矩阵D_j；

（2）获取定量均值t_{mean}

选取多个设计好的距离矩阵带入到基于蚁群算法的优化路线模型中，得到一个个最小路径回路l_i，取 $l_{mean}=\frac{\sum l_i}{i_{max}}$,拣货下架平均时间$t_\tau=\frac{t_{\tau i}}{i_{max}}$，$t_{mean}=\frac{l_{mean}}{v}+t_\tau$

（3）拣货人员的理想拣货路线

由于按照模型简化处理原则，拣货人员已经被限制初始复核台跟终点复核台一致，则易实现多个单个任务单理想路线的连接，分别得到 9 位拣货人员总的理想拣货路线，见表格计算结果。

（4）计算出库总时间和复核台利用率

记每个复核台总工作时间为t_{fi}，由于复核台存在工作量堆积现象，且一直到所有任务单都拣货完成后，仍有任务单堆积待复核打包。则出库时间等于单个复核台的最大工作时间与第一个任务单拣货完成时间之和，即$T'=\max(t_{fi})+T_1'$，因为除第一个任务单外，其他任务单的拣货时间都包含于复核打包时间里了，则单个复核台利用率$\eta_i=\frac{t_{fi}}{T'}$。

9．3 模型效果分析

根据本题所给参数表格，基于蚁群算法的优化路径算法，我们通过 Matlab 仿真程序多次迭代，得到以下结果。取 20 个任务单的程序结果，取平均得到$t_{mean}\approx 445.6s$。复

核台最大运行时间：$\max(t_{fi}) = 180 * 30 = 5400s$，则出库时间$T' = \max(t_{fi}) + T_1' = 5845.6s$。

复核台 FH01 利用率：$\eta_1 = \frac{t_{f1}}{T'} \approx 92.38\%$

复核台 FH03 利用率：$\eta_2 = \frac{t_{f2}}{T'} = \frac{174*30}{5845.6} \approx 89.30\%$

复核台 FH10 利用率：$\eta_3 = \frac{t_{f3}}{T'} = \frac{160*30}{5845.6} \approx 82.11\%$

复核台 FH12 利用率：$\eta_4 = \frac{t_{f4}}{T'} = \frac{171*30}{5845.6} \approx 87.76\%$

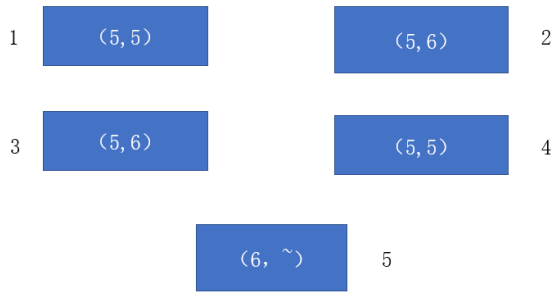
第十章 问题五解答

通过对第四题得到的简化模型的分析，调整每个拣货人员负责的任务单数和复核台的拣货人员人数，即可定性、定量地分析最短出库时间的变化。

每个拣货人员负责的任务单数和复核台的拣货人员人数的最优调整。

因为有 9 个拣货人员，5 个可正常工作的复核台，则为使复核台工作量尽可能均匀，应该是有 4 个复核台有两个人，一个复核台有一个人。则最短出库时间与有 2 个拣货人员的复核台有关。

接下来对拣货人员负责的任务单数进行最优调整。考虑简化模型的优化原则，总共有 49 个任务单，平均应有 4 个人分到 6 个任务单，5 个人分到 5 个任务单来使得复核台工作量尽可能均匀。由于第四题中估计每个任务单完成时间的定量均值t_{mean}已求出，这要求得出库时间只需算出单个复核台的最大工作时间，则只需考虑有两个拣货员的复核台的最大工作时间；而欲求最短出库时间，需要求得两个拣货员的复核台的最大工作时间的最小值。做出分配优化方案如下：



注：（5,6）表示复核台上第一个人有5个任务单，第二个人有6个任务单，
标号1,2,…,5是复核台标号，~表示第二个人不存在

图表 10-1 分配优化简化图

如上图，出库时间取决于有两个拣货员的复核台的最大工作时间，也就是与（5,6）相关。为得到最短出库时间，则必须使在保证有（5,6）的三个复核台的其中一个为最大工作时间的复核台的前提下，比如为编号 2，且使它的所有最大工作时间达到最小，比如在编号 2 复核台的所有工作时间取最短的一个时间。

深刻分析，我们得到分配优化如下：把订单号最多的前 6 个任务单分配给编号 5 复

核台，接下来分配编号 2/3/4 复核台。假设编号 2 复核台为最大工作时间的复核台，则对编号 2 复核台的拣货人员优先分配订单数较多的任务单，之后其他任务台随机组合不会对出库时间造成影响，如下表给出的分配方案：以 FH01 为上面的编号 2 复核台。

拣货人员编号	复核台编号	任务单编号
1	FH01	T0048，T0020，T0004，T0014，T0047
2	FH01	T0017，T0028，T0029，T0011，T0025，T0036
3	FH03	T0022，T0002，T0006，T0044，T0039
4	FH03	T0024，T0045，T0007，T0012，T0031，T0032
5	FH10	T0001，T0018，T0042，T0019，T0037
6	FH10	T0034，T0013，T0043，T0027，T0010，T0021
7	FH12	T0009，T0046，T0033，T0003，T0023
8	FH12	T0030，T0041，T0015，T0005，T0049
9	FHXX	T0035，T0026，T0016，T0008，T0040，T0038

则在增加任意一个可正常工作台的情况下，复核台最大运行时间：$\max(t_{fi}) = 166 * 30 = 4980s$，

出库时间为：

$$T' = \max(t_{fi}) + T_1' = 5425.6s$$

因此，在增加任意一个可正常工作台的情况下，出库时间是减小的，且出库时间范围估计在5425.6s 附近，缩短了约 7.2%的时间。

此

第十一章 问题六解答

本题是基于畅销品离复核台的位置与畅销品所在货架拥挤程度两个因素的仓内商品摆放问题，目标在于两因素中寻求一个效果较好的折中方案，使得全局上拣货效率高，以提高仓库拣货效率，减少非必要人力劳动资源开销。

本题中同排货架间通道宽度$2 * d = 1500mm$，实际场景中，当出现两拣货员在同一通道时候，很大程度上会存在避让问题，当更多人时候会存在拥挤，使得人流速度实际未能达到题干中的$1.5m/s$，因而整体拣货效率下降。同时，从前五题的解答以及对实际仓库的布局了解，我们发现在仓库左下角复核台与货架之间的距离是$2 * w_2 = 2000mm$，而且该位置相对其他货架具有离复核台较近的优势。因而，我们可以在该位置货架中尽可能多的摆放畅销品。

基于以上分析，关于畅销品离复核台的位置与畅销品所在货架拥挤程度两个因素的仓内商品摆放问题，我们得到以下建议。

（1）对仓库内的货架，根据距离复核台长短以及实际位置情况，我们把货架划分为四个区域：A 区，B 区，C 区，D 区。在某种程度上，属于同一区的货架离其最近的复核台的距离可以近似接近。区域划分如下图 11-1 所示。

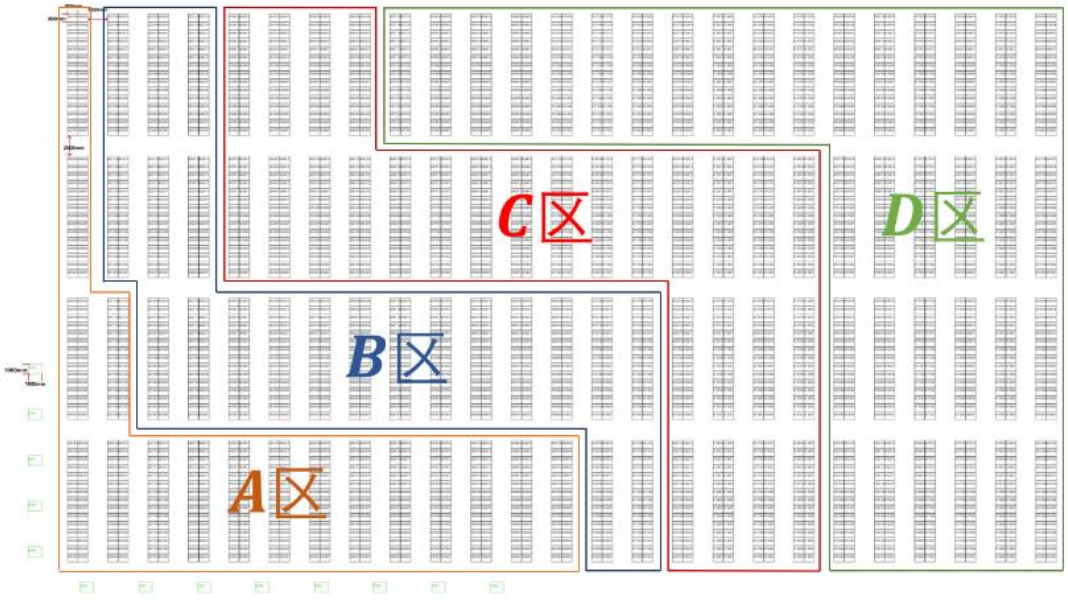


图 11-1 货架区域划分图

（2）本仓库有 200 个货架，我们设定 Num_k为第k个货架上存放的商品总数。我们根据仓库的历史数据分析得到商品的畅销程度，将商品划分为畅销品和非畅销品两类别。我们记Num_y_k为第k个货架上存放的畅销品数目。因而我们得到第k个货架上畅销品占比$\alpha_k = \frac{Num_y_k}{Num_k}$。针对将畅销品聚集在距离复核台较近的货架上而出现人流拥挤的问题，我们根据不同区域采取距离与畅销品比例α_k折中的方案。A 区中货架具有明显的短距离运送到复核台的优势，因而我们可以设定
A 区的货架上畅销品占比$\alpha_k = 85\%$，对于 D 区货架具有明显的长距离运送到复核台的劣势，我们把其货架上畅销品占比设定为$\alpha_k = 10\%$。具体分析，如下图表 11-2 所示。

区域	*A 区*	*B 区*	*C 区*	*D 区*
该区畅销品占比α_k	85%	65%	30%	10%

图表 11-2 区域畅销品占比表格

（3）针对 A 区 B 区中畅销品占比大于50%的实际情况，从货格上分析，我们可以将货架上的畅销品与非畅销品在货格上分隔摆放，在某种程度上，减缓了同一货架上畅销品位置堆积，造成具体某位置的拥挤的问题。

（4）商品摆放问题模型要根据季度有所应变。对于不同季节或者时段，其仓库内畅销品的畅销程度有所变化，仓库应根据数据分析，对区域内的畅销品类别进行恰当变化。

第十二章 模型评价与展望

12．1 模型评价

12.1.1 模型优点

（1）针对理想路线建立的的蚁群优化模型，本文采用了典型的路径优化算法——蚁群算法，它是寻找优化路径的概率型算法，具有分布计算、信息正反馈和启发式搜索的特征，本质上是进化算法中的一种启发式全局优化算法，使得优化模型具有完美的理论支撑，模型更加完善。
（2）在优化模型中，本文深度分析了各个题目是否存在复核台工作量堆积的问题，针对不同情况对基础蚁群优化模型进行改善，应用于解决越来越复杂的拣货路线优化问题，更具有客观性和信服度。
（3）优化模型灵活度高，通过改变距离矩阵的含参种类，得到只含货格的距离矩阵、含有货格和初始复核台的距离矩阵、含有货格和初始、终点复核台的距离矩阵，可得到三种适宜的优化路径计算方式，适用面较广。
（4）第四、五题模型中，我们充分考虑到拣货员单个任务单理想路线长度、拣货人员在复核台的人数分配、以及拣货人员手中的任务单数量分配对出库时间的影响，并对影响因子的重要性进行划分，合理消除弱影响因子的影响，减少自变量；针对问题的高复杂性、随意性，在误差允许范围内，以估计整体水平为目的，人为简化规定初始、终点位置相同，极大降低了计算量。简化优化模型的建立，对出库时间整体水平的估计具有很高的可信度，适用于高复杂环境情况。

12.1.2 模型缺点

（1）蚁群算法具有自身的局限性。蚁群算法具有较强的鲁棒性、优良的分布式计算机制、易于与其他算法相结合等优点，但是蚁群算法同样具有明显的缺点，搜索初期信息素比较匮乏，导致信息素的指导作用较弱，算法易于陷入局部最优解，在解决大型优化问题时，蚁群算法存在搜索空间和搜索时间上的矛盾,易出现过早收敛于局部最优解及计算时间过长的弱点,算法的稳定性较差。并且蚁群算法的参数对算法的性能影响很大，需要进行大量的调参过程。
（2）优化模型中对理想拣货路线的确定是通过三段路线组合而成的，每一段路线的最优的相加，有时候不一定是从初始复核台到终点复核台的最优理想路线，存在路径长度上的一些偏差，但在有多个任务单时最终得到的出库时间的误差还是较小的，大致处于以真实值为平均的数据上下波动状态。
（3）第四、五题的简化优化模型，由于是存在消元思想与简化计算方式，所以得到的结果必然会与真实值有一定误差的，但仍能反映出真实值的水平。

12．2 后期展望

（1）针对蚁群算法固有的局限性，后期我们会对蚁群算法的参数设计进行优化。由于遗传算（GA）具有计算过程简单、全局搜索能力强，易与其他算法相结合的优点，我们将采用 GA 对蚁群算法的参数进行组合优化，通过选择、交叉、变异、重插入和基于蚁群算法的解码等操作，对蚁群算法中的参数ρ、α、β进行组合优化。
（2）优化模型的程序中，存在批量输出理想路径不方便的问题，需要人工手动地进行组合、并手动分批对元素序号编译得到原始名称，时间耗费大，所以我们将会对这方面进行程序的优化补充。

参考文献

[1] 陈荣,谢浩,张水旺.基于人工鱼群算法的多区型仓库人工拣货路径优化研究[J].南阳理工学院学报,2019,11(06):6-13.

[2] 李诗珍. 配送中心拣货作业优化设计与控制研究 [D] . 成都:西南交通大学，2008.

[3] Laura Garcia-Hernandez,Antonio Arauzo-Azofra,Lorenzo Salas-Morera,et al. Ｒecycling Plants Layout Design by Means of an Interactive Genetic Algorithm [J] . Intelligent Automation & Soft Computing，2013，19(3) :457－468.

[4] 李建斌，周玮，陈峰. B2C 电子商务仓库拣货路径优化策略应用研究 [J] . 运筹与管理，2014(1) :7－14

[5] 王永波，温佩芝，李丽芳，等. 大型仓库拣货路径优化 算法研究 [J] . 计算机仿真，2013，30(5) :337－340.

[6] 马宪民，刘妮. 自适应视野的人工鱼群算法求解最短 路径问题[J] . 通信学报，2014(1) :1－6.

[7] 孙力娟, 王良俊, 王汝传. 改进的蚁群算法及其在 TSP 中的应用研究[J]. 通信学报, 2004, 025(010):111-116.

[8] Dorigo, M. and Gambardella, L.M. (1997) Ant Colonies for the Travelling Salesman Problem. Biosystems, 43, 73-81.

附录

附录一：问题一求解距离矩阵相关代码

1.1 软件名称：matlab R2018
1.2 相关代码

```
%%
data=xlsread('D:\matlabcode\mathor\storage_ok_final.xlsx','B:C');
disp(data);
n=length(data);
disp(n);
%disp(data_lr(1,1));
%%
result=zeros(3013);
for i=1:3000
    fprintf('%d\t%d\n',i,j);
    for j=i:3000

        if i==j
            result(i,j)=0;
            continue;
        end

        %处理
        %100 个 storage
        storage_i=ceil(i/30);
        storage_j=ceil(j/30);
        row_i=rem(storage_i,4);
        row_j=rem(storage_j,4);
        %判断左右边
        test_i=ceil(i/15);
        test_j=ceil(j/15);
        lr_i=rem(test_i,2);%==0 为右边
        lr_j=rem(test_j,2);
        %分成 25 列，为计算?x 左右类型分类
        lie_i=ceil(test_i/8);
        lie_j=ceil(test_j/8);
```

```
if row_i~=row_j
    if lr_i==lr_j   %左左 or 右右

        del_x=abs(data(i,1)-data(j,1))+750*2;
        del_y=abs(data(i,2)-data(j,2));
        result(i,j)=del_y+del_x;
        result(j,i)=result(i,j);
        continue;


    elseif lr_i>lr_j %左--右边
        if lie_i<=lie_j  %大种
            del_x=abs(data(i,1)-data(j,1))+750*4+800;
            del_y=abs(data(i,2)-data(j,2));
            result(i,j)=del_y+del_x;
            result(j,i)=result(i,j);
            continue;

        else
            %小，直线
            del_x=abs(data(i,1)-data(j,1))-800;
            del_y=abs(data(i,2)-data(j,2));
            result(i,j)=del_y+del_x;
            result(j,i)=result(i,j);
            continue;
        end


    else %lr_i<lr_j
        if lie_i>lie_j
            del_x=abs(data(i,1)-data(j,1))+750*4+800;
            del_y=abs(data(i,2)-data(j,2));
            result(i,j)=del_y+del_x;
            result(j,i)=result(i,j);
            continue;

        else
            del_x=abs(data(i,1)-data(j,1))-800;
            del_y=abs(data(i,2)-data(j,2));
            result(i,j)=del_y+del_x;
            result(j,i)=result(i,j);
            continue;
        end
    end
```

```
    %-----------------------------------------
else %==同一行-----------------------------------
    %-----------------------------------------
    temp_x_del=abs(data(i,1)-data(j,1));


    if temp_x_del<1  %同一咧


        del_x=abs(data(i,1)-data(j,1))+750*2;
        del_y=abs(data(i,2)-data(j,2));
        result(i,j)=del_y+del_x;
        result(j,i)=result(i,j);
        continue;

    elseif temp_x_del==2300%临列，不需要穿越


        del_x=abs(data(i,1)-data(j,1))-800;
        del_y=abs(data(i,2)-data(j,2));
        result(i,j)=del_y+del_x;
        result(j,i)=result(i,j);
        continue;

    else


        %hengxian=[2250;16250;30250;44250];
        shu_i=data(i,2)+400-2250;
        while shu_i>14000
            shu_i=shu_i-14000;
        end


        shu_j=data(j,2)+400-2250;
        while shu_j>14000
            shu_j=shu_j-14000;
        end

        shu_i_j_sum=shu_i+shu_j;
        if shu_i_j_sum>(13500)%关键值 长方形模型  折中选择


            %向上
```

```
del_y=27000-shu_i_j_sum;

if lr_i==lr_j
    %左左 or 右右

    del_x=abs(data(i,1)-data(j,1))+750*2;

    result(i,j)=del_y+del_x;
    result(j,i)=result(i,j);
    continue;

elseif lr_i>lr_j


     %左--右边
    if lie_i<=lie_j

         %大种
        del_x=abs(data(i,1)-data(j,1))+750*4+800;
% del_y=abs(data(i,2)-data(j,2));
        result(i,j)=del_y+del_x;
        result(j,i)=result(i,j);
        continue;

    else


        del_x=abs(data(i,1)-data(j,1))-800;
% del_y=abs(data(i,2)-data(j,2));
        result(i,j)=del_y+del_x;
        result(j,i)=result(i,j);
        continue;
    end
else
    if lie_i>lie_j


        del_x=abs(data(i,1)-data(j,1))+750*4+800;
% del_y=abs(data(i,2)-data(j,2));
        result(i,j)=del_y+del_x;
        result(j,i)=result(i,j);
        continue;

    else
```

```
            del_x=abs(data(i,1)-data(j,1))-800;
  %  del_y=abs(data(i,2)-data(j,2));
            result(i,j)=del_y+del_x;
            result(j,i)=result(i,j);
            continue;
        end
    end


else


    %向上
    del_y=shu_i_j_sum;

    if lr_i==lr_j
        %左左 or 右右

        del_x=abs(data(i,1)-data(j,1))+750*2;

        result(i,j)=del_y+del_x;
        result(j,i)=result(i,j);
        continue;

    elseif lr_i>lr_j


         %左--右边
        if lie_i<=lie_j



             %大种
            del_x=abs(data(i,1)-data(j,1))+750*4+800;


   % del_y=abs(data(i,2)-data(j,2));
            result(i,j)=del_y+del_x;
            result(j,i)=result(i,j);


            continue;


        else
```

```
                    del_x=abs(data(i,1)-data(j,1))-800;
                % del_y=abs(data(i,2)-data(j,2));
                    result(i,j)=del_y+del_x;
                    result(j,i)=result(i,j);
                    continue;
                end
             else
                if lie_i>lie_j


                    del_x=abs(data(i,1)-data(j,1))+750*4+800;
                % del_y=abs(data(i,2)-data(j,2));
                    result(i,j)=del_y+del_x;
                    result(j,i)=result(i,j);
                    continue;

                else

                    del_x=abs(data(i,1)-data(j,1))-800;
              %  del_y=abs(data(i,2)-data(j,2));
                    result(i,j)=del_y+del_x;
                    result(j,i)=result(i,j);
                    continue;
                end
              end
          end
      end
  end
end



%%
%货格
fuhetai=xlsread('D:\matlabcode\mathor\fuhetai_data.xlsx','B:C');
temp_fuhetai=fuhetai;
for i=3001:3008
    for j=1:3000
        fprintf('%d\t%d\n',i,j);
        %1-8 的复核台定位到上边
        temp_fuhetai(i-3000,1)=fuhetai(i-3000,1)+500;
        temp_fuhetai(i-3000,2)=fuhetai(i-3000,2)+1000;
```

```
%100 个 storage
storage_i=ceil(i/30);
storage_j=ceil(j/30);
row_i=rem(storage_i,4);
row_j=rem(storage_j,4);
%判断左右边
test_i=ceil(i/15);
test_j=ceil(j/15);
lr_i=rem(test_i,2);%==0 为右边
lr_j=rem(test_j,2);
%分成 25 列，为计算?x 左右类型分类
lie_i=ceil(test_i/8);
lie_j=ceil(test_j/8);
%%
%左
if lr_j==1
    if (data(j,1)-750)>=temp_fuhetai(i-3000,1)
        del_x=data(j,1)-temp_fuhetai(i-3000,1);
        del_y=data(j,2)+400-temp_fuhetai(i-3000,2);
        result(i,j)=del_x+del_y;
        result(j,i)=result(i,j);
        continue;

    else
        del_x=abs(temp_fuhetai(i-3000,1)-data(j,1))+750*2;
        del_y=data(j,2)+400-temp_fuhetai(i-3000,2);
        result(i,j)=del_x+del_y;
        result(j,i)=result(i,j);
        continue;
    end
else
    %右边
    if (data(j,1)+750+800)>=temp_fuhetai(i-3000,1)
        del_x=(data(j,1)+750+800)-temp_fuhetai(i-3000,1)+750+800;
        del_y=data(j,2)+400-temp_fuhetai(i-3000,2);
        result(i,j)=del_x+del_y;
        result(j,i)=result(i,j);
        continue;

    else
        del_x=abs(temp_fuhetai(i-3000,1)-data(j,1))-800;
        del_y=data(j,2)+400-temp_fuhetai(i-3000,2);
        result(i,j)=del_x+del_y;
        result(j,i)=result(i,j);
```

```
                continue;
            end

        end
    end
end


%%数列
%
for i=3009:3013
    for j=1:3000
        fprintf('%d\t%d\n',i,j);

        %100 个 storage

        storage_j=ceil(j/30);%46--2

        row_j=rem(storage_j,4);%2--2
        %判断左右边

        test_j=ceil(j/15);%4--youbian
       %==0 为右边
        lr_j=rem(test_j,2);
        %分成 25 列，为计算?x 左右类型分类

        lie_j=ceil(test_j/8);  %diyilie
        %%最左边的搞定先
        if (((test_j==1||test_j==3)||test_j==5)||test_j==7)
            del_x=(data(j,1))-temp_fuhetai(i-3000,1)-1000;
            del_y=abs((data(j,2)+400)-(temp_fuhetai(i-3000,2)+500));
            result(i,j)=del_x+del_y;

            result(j,i)=result(i,j);
            continue;

        end
        sum_two=13500*2-1000;
        %%第一行
        if i<3012&&row_j==1%第一行
            shu_i=temp_fuhetai(i-3000,2)-2250;
            shu_j=(data(j,2)+400)-2250;
            sum_shu_i_j=shu_j+shu_i;
            if sum_shu_i_j>(sum_two/2)%上走
                del_y=sum_two-sum_shu_i_j;
```

```
        if lr_j==0%you边
            del_x=(data(j,1)-temp_fuhetai(i-3000,1)-500)+800+750*2;
            result(i,j)=del_x+del_y;
            result(j,i)=result(i,j);
            continue;
        else%左边
            del_x=data(j,1)-temp_fuhetai(i-3000,1)-500;
            result(i,j)=del_x+del_y;
            result(j,i)=result(i,j);
            continue;

        end
    else
        %下走
        del_y=sum_shu_i_j;
        if lr_j==0%you边
            del_x=(data(j,1)-temp_fuhetai(i-3000,1)-500)+800+750*2;
            result(i,j)=del_x+del_y;
            result(j,i)=result(i,j);
            continue;
        else%左边
            del_x=data(j,1)-temp_fuhetai(i-3000,1)-500;
            result(i,j)=del_x+del_y;
            result(j,i)=result(i,j);
            continue;
        end
    end
end
%--------------------------------------

if i>=3012&&row_j==2%第二行
    shu_i=temp_fuhetai(i-3000,2)-16250;
    shu_j=(data(j,2)+400)-16250;
    sum_shu_i_j=shu_j+shu_i;
    if sum_shu_i_j>(sum_two/2)%上走
        del_y=sum_two-sum_shu_i_j;
        if lr_j==0%you边
            del_x=((data(j,1)-temp_fuhetai(i-3000,1))-500)+800+750*2;
            result(i,j)=del_x+del_y;
            result(j,i)=result(i,j);
            continue;
        else%左边
            del_x=data(j,1)-temp_fuhetai(i-3000,1)-500;
            result(i,j)=del_x+del_y;
            result(j,i)=result(i,j);
```

```
            continue;
        end
    else
        %下走
        del_y=sum_shu_i_j;
        if lr_j==0%you 边
            del_x=(data(j,1)-temp_fuhetai(i-3000,1)-500)+800+750*2;
            result(i,j)=del_x+del_y;
            result(j,i)=result(i,j);
            continue;
        else%左边
            del_x=data(j,1)-temp_fuhetai(i-3000,1)-500;
            result(i,j)=del_x+del_y;
            result(j,i)=result(i,j);
            continue;
        end
    end
end


if data(j,2)>temp_fuhetai(i-3000,2)
    del_y=(data(j,2)+400)-(temp_fuhetai(i-3000,2)+1000);
    if lr_j==1%左边
        del_x=data(j,1)-temp_fuhetai(i-3000,1)-500;
        result(i,j)=del_x+del_y;
        result(j,i)=result(i,j);
        continue;
    else%youbian
        del_x=(data(j,1)-temp_fuhetai(i-3000,1)-500)+800+750*2;

        result(i,j)=del_x+del_y;
        result(j,i)=result(i,j);
        continue;
    end
else
        %下
    del_y=abs(temp_fuhetai(i-3000,2)-(data(j,2)+400));
    if lr_j==1%左边
        del_x=data(j,1)-(temp_fuhetai(i-3000,1)+500);
        result(i,j)=del_x+del_y;
        result(j,i)=result(i,j);
        continue;
    else%youbian
        del_x=(data(j,1)-temp_fuhetai(i-3000,1)-500)+800+750*2;
```

```
                result(i,j)=del_x+del_y;
                result(j,i)=result(i,j);
                continue;
            end
        end
    end

end
for i=3001:3013
    for j=i:3013
        fprintf('%d\t%d\n',i,j);
        del_x=abs(temp_fuhetai(i-3000,1)-temp_fuhetai(j-3000,1));
        del_y=abs(temp_fuhetai(i-3000,2)-temp_fuhetai(j-3000,2));
        result(i,j)=del_x+del_y;
        result(j,i)=result(i,j);
        continue;
    end
end

xlswrite('D:\matlabcode\mathor\my_result_all_3013.xlsx',result);
```

附录二：改进型蚁群算法求解问题 2 问题 3 程序

2.1 软件：Matlab R2018

2.2 代码

```
%%-------------------------------------------------------------------------
%% 主要符号说明
%% NC_max 最大迭代次数
%% m 蚂蚁个数
%% Alpha 表征信息素重要程度的参数
%% Beta 表征启发式因子重要程度的参数
%% Rho 信息素蒸发系数
%% Q 信息素增加强度系数
%% R_best 各代最佳路线
%% L_best 各代最佳路线的长度

%%第一步：变量初始化
clear all; %清除所有变量
close all; %清图
clc ;      %清屏
m=50;     %% m 蚂蚁个数
Alpha=1;  %% Alpha 表征信息素重要程度的参数
Beta=5;   %% Beta 表征启发式因子重要程度的参数
Rho=0.1;  %% Rho 信息素蒸发系数
NC_max=200;    %%最大迭代次数
Q=100;         %%信息素增加强度系数
n=27;   %点的个数
D =xlsread( ‘’D:\matlabcode\mathor\my_result_all_3013.xlsx',result’ );
D=reshape(D,n,n)
for i = 1:n
    for j = 1:n
        if i == j
            D(i,j)=eps;
        end
    end
end

Eta=1./D;           %Eta 为启发因子，这里设为距离的倒数
Tau=ones(n,n);      %Tau 为信息素矩阵
Tabu=zeros(m,n);    %存储并记录路径的生成
NC=1;                %迭代计数器，记录迭代次数
R_best=zeros(NC_max,n);        %各代最佳路线
L_best=inf.*ones(NC_max,1);    %各代最佳路线的长度
L_ave=zeros(NC_max,1);         %各代路线的平均长度
while NC<=NC_max         %停止条件之一：达到最大迭代次数，停止
```

```
    %%第二步：将 m 只蚂蚁放到 n 个货格上
    Randpos=[];   %随即存取
    for i=1:(ceil(m/n))
        Randpos=[Randpos,randperm(n)];
    end
    Tabu(:,1)=(Randpos(1,1:m))';

    %%第三步：m 只蚂蚁按概率函数选择下一货格位置，完成各自的周游
    for j=2:n     %所在货格不计算
        for i=1:m
            visited=Tabu(i,1:(j-1)); %记录已访问的货格，避免重复访问
            J=zeros(1,(n-j+1));       %待访问的货格
            P=J;                      %待访问货格的选择概率分布
            Jc=1;
            for k=1:n
                if length(find(visited==k))==0   %开始时置 0
                    J(Jc)=k;
                    Jc=Jc+1;                          %访问的货格个数自加 1
                end
            end

            %下面计算待选货格的概率分布
            for k=1:length(J)
                P(k)=(Tau(visited(end),J(k))^Alpha)*(Eta(visited(end),J(k))^Beta);
            end
                P=P/(sum(P));
                %按概率原则选取下一个货格地点
                Pcum=cumsum(P);     %cumsum，元素累加即求和
                Select=find(Pcum>=rand); %若计算的概率大于原来的就选择这条路线
                to_visit=J(Select(1));
                Tabu(i,j)=to_visit;
        end
    end
if NC>=2
    Tabu(1,:)=R_best(NC-1,:);
end

%%第四步：记录本次迭代最佳路线
L=zeros(m,1);     %开始距离为 0，m*1 的列向量
for i=1:m
    R=Tabu(i,:);
    for j=1:(n-1)
        L(i)=L(i)+D(R(j),R(j+1));    %原距离加上第 j 个货格到第 j+1 个货格的距离
    end
    L(i)=L(i)+D(R(1),R(n));       %一轮下来后走过的距离
```

```
    end
    L_best(NC)=min(L);            %最佳距离取最小
    pos=find(L==L_best(NC));
    R_best(NC,:)=Tabu(pos(1),:); %此轮迭代后的最佳路线
    L_ave(NC)=mean(L);            %此轮迭代后的平均距离
    NC=NC+1                       %迭代继续

    %%第五步：更新信息素
    Delta_Tau=zeros(n,n);         %
    for i=1:m
        for j=1:(n-1)
            Delta_Tau(Tabu(i,j),Tabu(i,j+1))=Delta_Tau(Tabu(i,j),Tabu(i,j+1))+Q/L(i);
            %此次循环在路径（i，j）上的信息素增量
        end
        Delta_Tau(Tabu(i,n),Tabu(i,1))=Delta_Tau(Tabu(i,n),Tabu(i,1))+Q/L(i);
        %此次循环在整个路径上的信息素增量
    end
    Tau=(1-Rho).*Tau+Delta_Tau; %考虑信息素挥发，更新后的信息素
    %%第六步：禁忌表清零
    Tabu=zeros(m,n);              %%直到最大迭代次数
end

%%第七步：输出结果
Pos=find(L_best==min(L_best)); %找到最佳路径（非0为真）
Shortest_Route=R_best(Pos(1),:) %最大迭代次数后最佳路径
Shortest_Length=L_best(Pos(1)) %最大迭代次数后最短距离


plot(L_best) %
hold on
plot(L_ave,'r') %
title('平均距离和最短距离')      %标题
xlabel('迭代次数')
```