针对物流选址规划的免疫匈牙利法

张恒硕[1)]

[1)](南开大学 人工智能学院 天津 300350)

摘 要 当下经济环境对物流有很高要求，而物流的重要成本因素便是选址问题。本文针对物流选址规划问题，回顾了智能算法和线性规划的解决方案，并结合二者，提出了一种免疫匈牙利法（Immune Hungarian Algorithm，IHA）。免疫匈牙利法兼具免疫算法（Immune Algorithm，IA）和匈牙利法（Hungarian method）的双重优点，前者的随机性在复杂问题具有高效性，后者的准确度能在简单问题确保最优解。当配送点数量在一定区间（10 左右）变化时，免疫匈牙利法的时间复杂度是常数级，在 0.01s 左右。针对实例实验问题，免疫匈牙利法的准确率在 80%左右，远高于变染色体智能算法的 31%。另外，免疫匈牙利法还具有良好的拓展性。

关键词 物流选址 智能算法 免疫算法 线性规划 匈牙利法 方案优化

Immune Hungarian Algorithm for Logistics Site Selection Planning

Zhang Heng-Shuo[1)]

[1)](School of Artificial Intelligence, Nankai University, Tianjin 30035)

Abstract The current economic environment has high requirements for logistics, and the important cost factor of logistics is site selection. This article reviews the solutions of intelligent algorithms and linear programming for logistics location planning problems, and proposes an Immune Hungarian Algorithm by combining the two methods. The Immune Hungarian method combines the dual advantages of Immune Algorithm and Hungarian method. The randomness of the former is efficient in complex problems, while the accuracy of the latter ensures optimal solutions in simple problems. When the number of delivery points varies within a certain range (around 10), the time complexity of the immune Hungarian method is constant, around 0.01 seconds. Regarding the experimental problem, the accuracy of the Immune Hungary method is around 80%, far higher than the 31% of the Chromosomal Intelligence algorithm. In addition, the Hungarian immunization method also has good scalability.

Key words Logistics site selection; Intelligent algorithm; Immune algorithm; Linear programming; Hungarian method; Plan optimization

1 引言

物流选址规划问题，作为智慧物流的一个重要组成部分，在商业快速发展、物品需要高效流动的当今社会，是一个具有重要意义的问题。在规划时，往往需要考虑大量的因素，比如选址成本（包含地段、容量等因素）、配送点和零售商合作关系（配送量、配送频次）、物品种类（配送需求、配送限制）等，其中最重要的是运营成本，主要指配送点向服务的零售商送货的成本。在优化运营成本的物流选址规划问题中，以配送点到零售商的距离代替运输成本，考虑如下最小化问题：

每个配送点P_i有m个选址方案P_iL_m，并能服务n个零售商S_n。其对应关系如下图图 1-1。

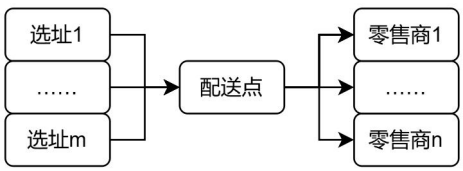

图 1-1 配送点的多选址与多服务

现有i个配送点P_i，j个零售商S_j（$j \le \sum_{x=1}^{i} n_x$，即一定可以覆盖所有零售商），并已知各选址到所有零售商的距离$C_{i_m j}$，如下表表 1 格式。求距离和最小的服务组合方案，如下图图 1-2。

表 1 各选址到各零售商的距离

距离	*PiLm*
Sj	$C_{i_m j}$

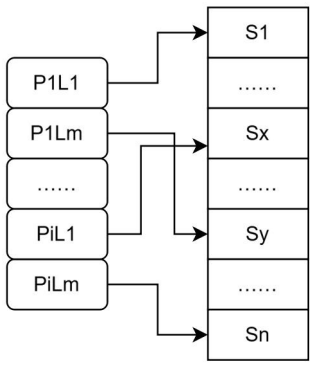

图 1-2 服务组合方案

如果把配送点看作一个序列，零售商看作相应的选项，则物流选址规划问题可以近似为旅行商问题（Traveling Salesman Problem, TSP）。一方面二者都是有权图的最小化问题，前者求解有向图的最短哈密顿通路，后者则求解近似的二分图最大匹配；另一方面二者都需要关注序列的合法性，前者不可以重复抵达同一座城市，而后者中每个配送点可服务的零售商数量受限。本文尝试使用常用于旅行商问题的智能算法（Intelligent Algorithms，IAs）来求解物流选址规划问题。

如果每个配送点只服务一个零售商，那问题简化为典型的分配问题，可以使用匈牙利法等线性规划（Linear programming，LP）方法进行求解。

智能算法具有随机性，能快速找到优秀解，但可能错过最优解；而匈牙利法作为一种剪枝遍历方法，在效率上较差，但不会错过最优解。本文尝试结合二者，发扬各自优点，获得准确而高效的物流选址规划问题模型。

2 综述

智能算法是一类模拟自然现象或生物行为的计算方法，具有较强的鲁棒性和自适应能力，能够在不确定和复杂的环境中找到近似最优解，常用于解决复杂优化问题。传统的智能算法有遗传算法（Genetic Algorithm，GA）、粒子群优化算法（Particle Swarm Optimization，PSO）、蚁群算法（Ant Colony Optimization，ACO）、差分进化（Differential Evolution，DE）等，在它们的基础上，逐渐衍生出免疫算法、狼群算法（Wolf pack algorithm，WPA）等新型智能算法。

在智慧物流方面，智能算法起到了巨大的作用，常用于复杂或繁琐的任务。在 2008 年和 2010 年，Wei Ding 等人[1]和 S. -z. Bai 等人[2]分别就粮食配送中心和乳制品配送中心的选址问题，提出了相应的遗传算法模型。在 2009 年，Q. -k. Cao 等人使用模拟退火的方法，对物流仓库选址和配送费用问题进行综合优化[3]。同样结合模拟退火的思想，T. Liu 等人在 2020 年构建了基于改进 k-means 聚类算法进行物流站点优化选址的数学模型，其可以获得经济成本的最优位置分布[4]。近年来，针对越来越精细化、复杂化的物流选址规划问题，学者们相继提出了新的智能算法模型。X. -l. Zhu 等人在 2023 年利用粒子群算法设计了一种智能物流配送站选址模型，使智能物流布局功能更加完善[5]。而在 2024 年，X. Wang 等人就应急救援中及时性和不确定性的应急物流问题，基于免疫优化算法实现了多中心物流仓储选址方案[6]。

在人为处理简单物流选址规划问题时，线性规划是最常用到的方法。X. Liu 等人在 2012 年对比了大量的配送中心选址方法，最终基于线性规划构建了物流网络模型[7]。然而，虽然单纯形法、匈牙利法等线性规划方法能确保找到最优解，但其时间复杂度分别为$O(mn^2)$或$O(n^3)$（其中m是约束数，n是变量数，即配送点数量）。在处理实际的复杂物流选址规划问题时，计算量随变量数增加而快速增加，这通常是不利的，因此一般很少单独使用线性规划方法进行求解。

为了平衡智能算法的随机性和线性规划方法的复杂度，有学者尝试将二者结合起来。早在 2012 年，S. Li, G 等人就从动态系统的自动化设计这一任务出发，提出了结合匈牙利法和遗传算法的算法 HAGP[8]。此后的 2020 年，C. -J. Huang 等人提出了将匈牙利法引入遗传算法解决旅行商问题的思路，其中，遗传算法的选择操作使用的是精英保留策略（Elitism）[9]。

3 免疫匈牙利法

在免疫算法评估适应值的阶段引入匈牙利法，便得到了免疫匈牙利法。免疫算法作为主体部分，

主要功能是搜索各配送点的选址方案组合；而匈牙利法部分则是求解具体选址与零售商的对应服务方案。

3.1 免疫算法

作为遗传算法的衍生算法，免疫算法相较其他智能算法，具有较好的鲁棒性和适应性，并凭借记忆功能具备较好的全局搜索能力。F. A. Imam 等人曾大量比较各种智能算法，找到了免疫算法的特点[10]。免疫算法还具有良好的兼容性，能与多数智能算法相结合，取得更好的效果，如 R. Shen 提出了一种融合了免疫算法机制的粒子群优化算法[11]。

免疫匈牙利法使用基本的免疫算法，在变异时采用单点变异，即各基因的取值以一定概率变成合法取值范围内的另一个值。在求解过程中，基因序列在初始化后将一直保持合法性，不会因为变异操作越限，这极大简化了主体迭代部分的复杂度，减少了耗时。

3.2 匈牙利法

作为求解分配问题的常用方法，匈牙利法适用于二分图最大匹配问题。为将其应用于一对多匹配问题，需要虚拟一定量的点，填补成方阵效率矩阵，再进行求解。以下是具体过程：

3.2.1 构造效率矩阵

a. 针对$j=\sum_{x=1}^{i} n_x$的问题，将配送点P_i选取的选址方案P_iL_m所对应的效率向量C_{i_m-}复制n_i次，如下图图 3-1，所有配送点构成的效率矩阵为边长为$j=\sum_{x=1}^{i} n_x$的方阵。

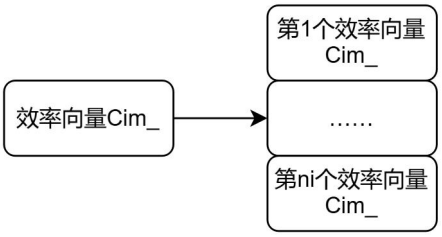

图 3-1 效率向量C_{i_m-}复制n_i次

b. 针对$j>\sum_{x=1}^{i} n_x$的问题，在 a.得到的$\sum_{x=1}^{i} n_x$行j列矩阵的基础上，另填补$(j-\sum_{x=1}^{i} n_x)$行全为 0 的虚拟配送点效率矩阵，形成边长为j的方阵效率矩阵。

3.2.2 匈牙利法求解

a. 分配问题：针对效率矩阵(c_{xy})，求解决策变量$a_{xy}=\begin{cases}1，P_x\text{的选址服务}S_y\\0，P_x\text{的选址不服务}S_y\end{cases}$，使目标函数$z=\sum_{x=1}^{j}\sum_{y=1}^{j} c_{xy}a_{xy}$最小化。另有约束$\sum_{x=1}^{j} c_{xy}=1,\forall x\in\{1,2,\ldots,j\}$和$\sum_{y=1}^{j} c_{xy}=1,\forall y\in\{1,2,\ldots,j\}$。

b. 克尼格（Knig）定理：由$b_{xy}=c_{xy}-u_x-v_y$的行、列约简得到的新效率矩阵(b_{xy})，其具有和原效率矩阵(c_{xy})一致的分配问题最优解a_{xy}。在匈牙利法求解中，通过约简得到足量的零元素。

c. 求解过程：行、列约简，覆盖零元素，调整效率矩阵，重复上述三个步骤直到得到最优解。代码中使用匈牙利法的函数 linear_sum_assignment()，其参数是效率矩阵，输出决策变量的位置序列a_{xy}。

3.2.3 获得适应值

由位置序列a_{xy}和效率矩阵(c_{xy})计算得到$z=\sum_{x=1}^{j}\sum_{y=1}^{j} c_{xy}a_{xy}$，该值越小，说明该组合方案越优。

3.3 免疫匈牙利法

结合上述的免疫算法和匈牙利法，可得到免疫匈牙利法，以下给出算法的全流程，图 3-2 则给出了算法的流程图。

算法 1 免疫匈牙利法

输入：

1) *length*（染色体上基因数量，对应配送点数目），
2) *values*（可能的基因型，对应配送点的可能选址），
3) *generations*（迭代次数），
4) *pop_size*（种群大小），
5) *clone*（克隆因子），
6) *pm*（变异概率），
7) C_{i_mj}（原始距离矩阵），

输出：

配送点选址与和零售商的最佳组合方式

过程：

1) 读取并处理距离数据，设置算法参数；
2) 创建初始种群；
3) 计算适应度值：
   ① 构建效率矩阵；
   ② 使用匈牙利法获得当前选址的最优组合方案；
   ③ 计算适应度值并保存；
4) 迭代：

① 在种群中排序选择优质个体；
② 克隆和变异；
③ 更新种群；
④ 获取世代最优，并与全局最优比较；

5) 转换全局最优为方案。

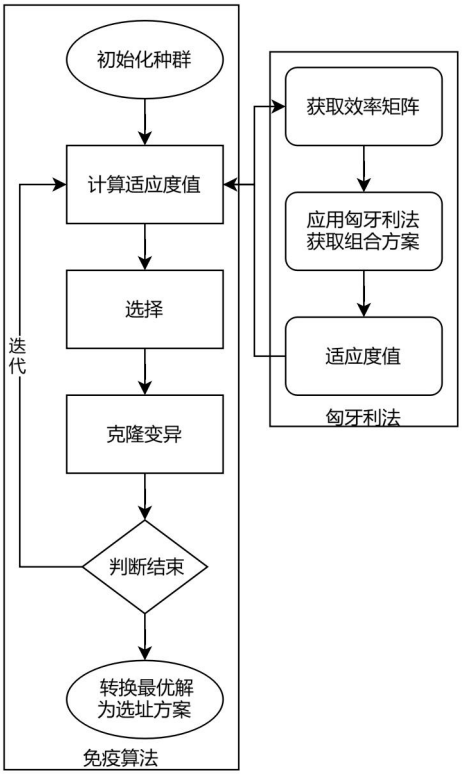

图 3-2 免疫匈牙利法流程图

为减少适应值评估次数，每个个体在生成后只会评估一次适应值，其后将该值存储起来，用于之后的比较。

4 免疫匈牙利法的实验研究

4.1 实例实验结果

以下求解一个简单实例：有 5 个配送点（P1、P2、P3、P4、P5），每个配送点有 3 个选址（L1、L2、L3），要给 10 个零售商（S1、S2、S3、S4、S5、S6、S7、S8、S9、S10）配送商品。一个配送点只能服务两个零售商，一个零售商只接受一个配送点服务。附录中给出了各选址到各零售商的距离表格，并给出了针对该问题的具体代码。

其求解结果如下：

- 最优方案的距离和为 551
- 方案如下：

P1.L2 到 S10,S9
P2.L1 到 S4,S5
P3.L3 到 S3,S1
P4.L1 到 S2,S8
P5.L3 到 S7,S6

该问题变量较少，较为简单，需要的种群迭代次数较少，以 10 轮为例，代码运行时间为 0.00951s。还可以使用遍历选址方案再使用匈牙利法的暴力方法进行求解，得到与上述一致的结果，耗时 0.00200s。这里并没有体现出免疫匈牙利法的优越性，下一部分会就耗时与变量数的关系进行进一步讨论。

4.2 与其他方法比较

4.2.1 暴力遍历

在前一部分与暴力遍历方法的比较中，面对简单少变量问题，免疫匈牙利法并没有表现出优越性，这是由于评估次数（种群大小×迭代次数，100×10）与遍历总数（$3^5 = 243$）在同一数量级上。在变量数增加的过程中，二者都会因为匈牙利法求解复杂化而增加耗时。暴力遍历要遍历的方案数几何增加，耗时也几何增加；而免疫匈牙利法的耗时随迭代次数增加，本文设计在连续十个世代结果保持不变则认为收敛到最优方案，这是常数级别的增加。下图图 4-1 给出了变量数增加过程中，两种方法求解的耗时。可以发现，在暴力遍历的耗时快速增加的同时，免疫匈牙利法的耗时变化不大，维持在 0.01s左右。这说明本方法很适合解决复杂问题。

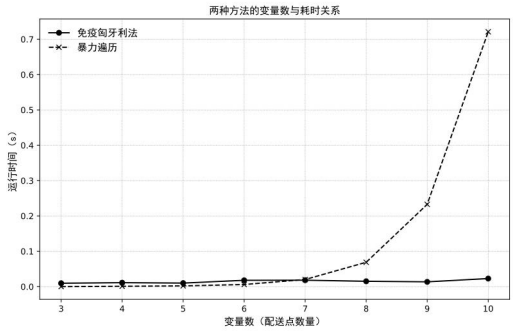

图 4-1 免疫匈牙利法与暴力遍历的耗时与变量数的关系

4.2.2 变染色体智能算法

本问题是典型的多峰问题，由于解有可行性要求，不同优质解之间的差别较大，普通的智能算法很难收敛到最优解。Z.-G. Chen 等人针对多峰问题，提出了一种基于分布式多峰个体（DIMP）的框架和融合了两种新机制的分布式个体差分进化（DIDE）算法[12]。本文为追求代码的简单性，没有融入复杂的机制，采用一种与免疫匈牙利法有相

通之处的变染色体智能算法进行比较，以证明前者的优越性。

变染色体智能算法将染色体分为两个部分，一部分对应配送点的选址，另一部分对应零售商选取的配送点，其结构如下图图 4-2。在进行变异的时候，两部分分别变异，且不会同时进行。配送点选址部分为单点变异。零售商选取配送点部分有三种变异方式，分别是交换、区间反转和基因移动，其原理如下图图 4-3。三种变异只会发生一种，第一种的概率较高。

图 4-2 变染色体智能算法的染色体

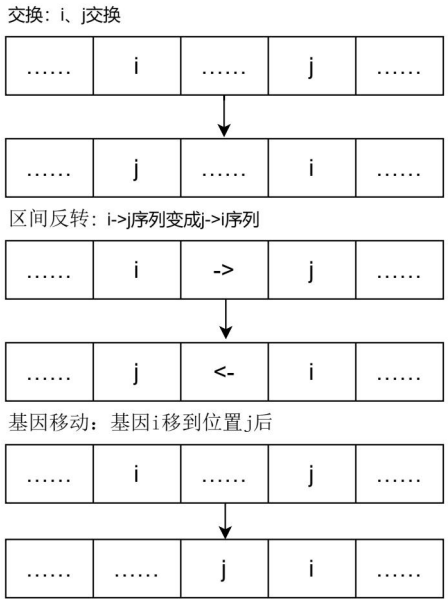

图 4-3 三种变异方式

本部分用实例测试了变染色体免疫算法，其虽然可能收敛到最优解，但结果正确率（最优解获得率）和结果稳定性（收敛结果与最优解差距）都较差，在实例问题中获得最优解的概率为 31%，而免疫匈牙利法是 80%。尤其值得指明的是，前者迭代 100 轮，后者迭代 10 轮，即使大量增加时间消耗，前者也无法在准确度上企及后者。免疫匈牙利法减少了基因数量，每个选址方案都用匈牙利法获得对应的最优组合，相当于把解域从整个三维山体变成了二维等高线图，这极大减少了搜索的复杂度。

4.2.3 遗传匈牙利法

在推广到复杂问题时，本模型需要权衡两个重点，一个是匈牙利法的高时间复杂度，另一个是智能算法随机性导致的迭代次数增加，后者的增加又将体现在前者的评估次数上。在实际问题中，考虑配送中心的辐射范围，一座城市的配送点数目其实是有限的，这说明不需要过度复杂化问题。

将免疫匈牙利法中的免疫算法换做遗传算法，并做适当修改，就可以得到遗传匈牙利法（Genetic Hungarian Algorithm，GHA）。遗传匈牙利法中的遗传算法部分应用单点交叉和单点变异，并用轮盘赌方法进行选择。为保证适应值高的个体被选的几率高，采用补数方法表示适应值，即用一个适当大的值减去匈牙利法求解的最小距离和。

以下两图分别展示了免疫匈牙利法和遗传匈牙利法在变量数增加时的耗时和最终结果（由暴力遍历得到全局最优解）。这里使用的数据是随机生成的，在同一变量数下使用一致的数据，俩个方法都进行 5 次，求耗时和结果的平均值。两个方法都迭代到 100 次或连续 10 个世代结果不变为止。

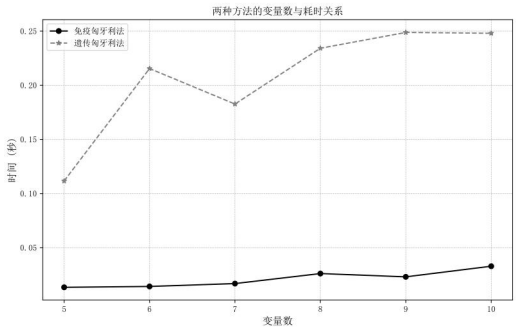

图 4-4 免疫匈牙利法与遗传匈牙利法的耗时与变量数的关系

由上图图 4-4 可以发现，遗传匈牙利法的耗时是免疫遗传算法的几十倍，这主要是因为前者很难收敛。在相同迭代次数下，前者的耗时大概是后者的 2 倍。

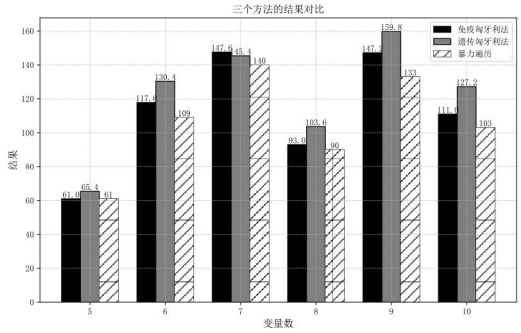

图 4-5 免疫匈牙利法、遗传匈牙利法与暴力遍历的结果

由上图 4-5 可以发现，免疫匈牙利法的结果要比遗传匈牙利法更接近暴力遍历的结果。在上述实例问题中，遗传匈牙利法的结果正确率要略高于免疫匈牙利法，但是结果稳定性要差很多。

免疫匈牙利法无论是耗时还是结果准确率，都要优于遗传匈牙利法，这说明免疫算法的使用是明智的。

5 总结与展望

免疫匈牙利法作为一种结合了智能算法和线性规划的方法，集合了二者优点，能够快速求得物流选址规划问题的最优解。本文测试的所有实例都是在一个配送点有三个选址、要服务两个零售商的背景下，这个条件改变并没有影响，只需要注意变异部分的合法性即可。

物流在时下的作用愈发凸显，能够帮助合理规划物流选址的方法是颇有意义的。免疫匈牙利法的机制极为简单，可拓展性极强，尤其是免疫算法部分，可以引入更多的机制，用于加快收敛、提高准确度等。

参 考 文 献

[1] Wei Ding, Yanyan Li and Qiuwen Zhang. An improved genetic algorithm for grain distribution centers locations. 2008 IEEE International Symposium on IT in Medicine and Education, 2008, pp. 11-15.
[2] S. -z. Bai, R. -z. Meng and S. -t. Zhang. Research on location selection of dairy distribution center. 2010 International Conference on Logistics Systems and Intelligent Management (ICLSIM), 2010, pp. 898-901.
[3] Q. -k. Cao, X. -k. Di and X. -x. Zhang. A Simulated Annealing Methodology to Estate Logistic Warehouse Location Selection and Distribution of Customers' Requirement. 2009 International Workshop on Intelligent Systems and Applications, 2009, pp. 1-4.
[4] T. Liu, X. Yang and Y. Yin. Selection Optimization Modeling of Logistics Site and Applications. 2020 5th International Conference on Information Science, Computer Technology and Transportation (ISCTT), 2020, pp. 571-574.
[5] X. -l. Zhu. A Particle Swarm Algorithm Based Model for Selecting the Location of Smart Logistics Distribution Centres. 2023 Smart City Challenges & Outcomes for Urban Transformation (SCOUT), 2023, pp. 211-217.
[6] X. Wang, G. Qin and H. Zhao. Based on Immune Optimization Algorithm for Multi-Storage Site Selection Algorithm. 2024 7th International Conference on Advanced Algorithms and Control Engineering (ICAACE), 2024, pp. 31-36.
[7] X. Liu. The site selection of distribution center based on linear programming transportation method. Proceedings of the 10th World Congress on Intelligent Control and Automation, 2012, pp. 3538-3542.
[8] S. Li, G. Yang and Q. Xie. Automatic Design Method of Dynamic Systems Based on Hungarian Algorithm and Genetic Programming. 2008 4th International Conference on Wireless Communications, Networking and Mobile Computing, 2008, pp. 1-4.
[9] C. -J. Huang. Integrate the Hungarian Method and Genetic Algorithm to Solve the Shortest Distance Problem. 2012 Third International Conference on Digital Manufacturing & Automation, 2012, pp. 496-499.
[10] F. A. Imam, O. Osanaiye, N. A. Imam, A. Nyangwarimam Obadiah, M. A. Rafindadi and S. Thomas. Hybridization of Artificial Immune System Algorithms with Other AI Algorithms: A Review. 2023 2nd International Conference on Multidisciplinary Engineering and Applied Science (ICMEAS), 2023, pp. 1-6.
[11] R. Shen, S. Liu, H. Zhang and L. Han. Immune Particle Swarm Optimization Algorithm Based on Optimal Chaos Search. 2022 IEEE 8th International Conference on Computer and Communications (ICCC), 2022, pp. 2256-2259.
[12] Z. -G. Chen, Z. -H. Zhan, H. Wang and J. Zhang. Distributed Individuals for Multiple Peaks: A Novel Differential Evolution for Multimodal Optimization Problems. in IEEE Transactions on Evolutionary Computation, vol. 24, no. 4, pp. 708-719, Aug. 2020.

附录 1 实例的距离表格

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
P1.L1	58	96	91	77	94	68	89	73	68	96
P1.L2	72	99	90	83	79	68	81	82	54	52
P1.L3	65	96	98	60	92	59	83	61	67	99
P2.L1	66	57	52	53	52	87	90	80	65	76
P2.L2	90	84	78	100	61	55	98	91	72	94
P2.L3	62	50	63	53	93	87	86	90	76	57
P3.L1	91	51	83	70	63	81	85	90	83	76
P3.L2	62	85	76	81	98	69	91	52	84	86
P3.L3	70	65	52	65	69	64	81	69	81	77
P4.L1	74	52	61	70	71	53	72	52	80	79
P4.L2	55	73	95	75	62	90	96	80	76	97
P4.L3	80	88	83	83	75	58	57	74	94	62
P5.L1	68	59	77	65	91	81	70	96	65	68
P5.L2	96	86	71	90	69	86	95	84	56	92
P5.L3	97	65	75	96	78	63	51	83	83	75

附录 2 针对实例的免疫匈牙利法代码

```
import random
import pandas as pd
import numpy as np
from scipy.optimize import linear_sum_assignment
import time

# 数据
df = pd.read_excel('data.xlsx')
df = df.values
df = df[:, 1:11].astype(np.int64)
# 参数
length = 5              # 个体基因数量——配送点数目
values = [1, 2, 3] # 可能基因型——配送点的可能选择
generations = 10     # 迭代次数
pop_size = 100        # 种群大小
clone = 2              # 克隆因子
pm = 0.01               # 变异概率

# 个体类
class Individual:
    def __init__(self, genes):
        self.genes = genes
        self.fitness_value = None
    def fitness(self):   # 匈牙利法评估适应值，储存以减少评估次数
        if self.fitness_value is None:
```

```
                # 抽取基因型对应的行，构成匈牙利法的效率矩阵
                i = np.arange(length)
                r = (3 * i + self.genes - 1).repeat(2)
                cost_matrix = df[r, :]
                # 匈牙利法求 min 值
                row_ind, col_ind = linear_sum_assignment(cost_matrix)
                self.fitness_value = cost_matrix[row_ind, col_ind].sum()
            return self.fitness_value

# 克隆和变异
def clone_and_mutate(selected, clone, pm):
    new_population = []
    for individual in selected:
        for _ in range(clone):
            cloned = Individual(individual.genes.copy())
            for i in range(length):
                if random.random() < pm:
                    cloned.genes[i] = random.choice(values)
            new_population.append(cloned)
    return new_population

# 转化最优解方案
def solution(individual):
    i = np.arange(length)
    r = (3 * i + individual.genes - 1).repeat(2)
    cost_matrix = df[r, :]
    row_ind, col_ind = linear_sum_assignment(cost_matrix)
    for j in range(length):
        print(f"P{j + 1}.L{individual.genes[j]}到S{col_ind[2 * j] + 1},S{col_ind[2 * j + 1] + 1}")

# 主函数
start_time = time.time()
population = [Individual([random.choice(values) for _ in range(length)]) for _ in range(pop_size)]
best_individual = min(population, key=lambda x: x.fitness())
# 保留迭代过程中的最优个体
for generation in range(generations):
    # 选择适应度低的个体
    population.sort(key=lambda x: x.fitness(), reverse=False)
    selected = population[:pop_size // 2]
    # 克隆并变异，获取新种群
    new_population = clone_and_mutate(selected, clone, pm)
    population = new_population[:pop_size]
    # 记录最佳个体
    generation_best_individual = min(population, key=lambda x: x.fitness())
    print(f"第{generation + 1}轮: 最优方案{generation_best_individual.genes}的花费为{generation_best_individual.fitness()}")
    if generation_best_individual.fitness() < best_individual.fitness():
        best_individual = generation_best_individual
# 输出最终结果
print("最优方案", best_individual.genes, "的花费为", best_individual.fitness())
solution(best_individual)
# 运行时间
end_time = time.time()
print("运行时间：%.5f 秒" % (end_time - start_time))
```