中国邮递员问题

一、引言

在城市交通规划、物流配送和环卫清扫等领域，路径规划问题是提升运营效率和降低成本的关键环节之一。如何规划出一条最短的路径，使所有街道或道路至少经过一次，并尽量减少重复行走过程，是实际操作中常见的需求。中国邮递员问题（Chinese Postman Problem, CPP）便是针对这一类问题提出的经典模型，为解决复杂路径规划提供了系统的理论支撑和实践指导。

中国邮递员问题最早由管梅谷先生于 20 世纪 60 年代提出，其核心思想来源于城市邮递员在投递邮件时，规划出一条高效的路径以确保所有街道都被覆盖至少一次，并最终返回出发点。该问题可以抽象为图论中的路径优化问题，即在一个带权连通图中找到一条最短的闭合路径，使得图中的每条边至少被遍历一次，并且总行程最短。这一问题与现实生活中的许多路径规划需求高度相似，自提出以来便受到了广泛关注，并在物流、交通和巡检等领域得到实际应用。

中国邮递员问题的求解思路与欧拉回路密切相关。在图论中，如果一个连通图的所有顶点的度均为偶数，即该图为欧拉图，那么就可以通过求解欧拉回路，直接得出最优解。然而，在实际问题中，大多数图并不天然满足欧拉图的条件，通常存在若干个度数为奇数的顶点，使得图无法直接构建欧拉回路，因此求解问题的核心在于如何处理奇度顶点，将其转换为偶数度，进而使整个图转变为欧拉图，以便求解欧拉回路。

本文将围绕中国邮递员问题的理论基础，探讨在无向图的基础上解决中国邮递员问题的方法，并结合具体实例和代码实现，对所提出的解决方案进行验证与分析。

二、问题定义与数学建模

2.1 问题定义

中国邮递员问题（Chinese Postman Problem, CPP）描述了一个典型的路径规划任务。在一个带权连通图中，设计一条最短路径，使得图中所有边都至少经过

一次，并最终返回出发点。该问题的核心目标是最小化重复路径的代价，减少不必要的行程。具体而言，给定一个无向连通图$G=(V,E)$，其中：

- $V=\{v_1,v_2,\ldots,v_n\}$表示顶点集，$|V|=n$；
- $E=\{e_1,e_2,\ldots,e_m\}$表示边集，$|E|=m$；
- 每条边$e\in E$代表一条街道或路径，非负权值$w:E\to R^+$,表示经过该路径的成本。

需要寻找一条闭合路径C,使得C至少涵盖E中的每条边一次，并使路径总长度$L(C)$最小，即

$$min\sum_{e\in C}w(e)$$

2.2 问题分类

根据无向图的性质，中国邮递员问题的求解大致可以分为两种情况。

若无向图G是欧拉图，即图中所有顶点的度数均为偶数，则存在一条欧拉回路直接满足条件，最短路径即为欧拉回路本身。

若图中存在奇度顶点，则不存在欧拉回路，需要增加额外的边，将所有奇度顶点配对，使得图中所有顶点的度均为偶数，从而将非欧拉图转化为欧拉图。

2.3 数学建模

定义顶点v的度为$d(v)$，若对于一个顶点$v\in V,d(v)\equiv 0(\mathrm{mod}\,2)$，则称该顶点为奇度顶点。

设图中奇度顶点集合为$S=\{v_1,v_2,\ldots,v_k\}$，其中k为偶数。

对于非欧拉图，需要在图中添加一定量的边，使得每个顶点的度均为偶数。可以通过在S中找到最小权完美匹配来实现，即配对S中的顶点，使得匹配边的总权值最小。换而言之，需要实现

$$min\sum_{(v_i,v_j)\in S}d(v_i,v_j)$$

其中，$d(v_i,v_j)$表示顶点 v_i和v_j之间的最短路径长度。

此类问题的求解关键在于如何在S中构造最优匹配，以最小代价将G转化为欧拉图。

2.4 基本原理

- 对于一个连通图而言，欧拉回路存在的充分必要条件是图中所有顶点的度均为偶数。

- 在任意无向图中，奇度顶点的个数必为偶数个。

三、问题分析及算法设计思路

3.1 总体思路

解决中国邮递员问题的算法可以分为四个主要阶段：

- 判断图的连通性

  解决 CPP 的前提条件是图为连通图，判断图是否连通来决定是否进行下一步计算。
- 判断图的欧拉性质

  通过计算图中每个顶点的度，判断是否满足欧拉图或半欧拉图的条件。
- 识别奇度顶点

  若存在奇度顶点，找出所有奇度顶点并记录。
- 最小成本匹配

  在奇度顶点之间建立最短路径，配对所有奇度顶点，将图转化为欧拉图。
- 构建欧拉回路

  在转化后的图中使用欧拉回路算法生成最终路径，并输出最优解。

3.2 关键算法模块及原理

3.2.1 判断图的连通性

在图论中，只有连通图才能构建完整的遍历路径。如果图不连通，则中国邮递员问题无解。具体问题中，可以采用多种算法来判断图是否连通，例如，可以使用 DFS 或 BFS 遍历图，检查是否能够访问所有顶点。或者查找任意两点间的最短路径，判断是否存在。如果图是连通的，则继续进行下一步逻辑；如果图不连通，则返回空回路。

由于本算法涉及最短路径查找，因此采用后一种办法，简化代码实现。

3.2.2 识别奇度顶点

识别奇度顶点是进行下一步逻辑的基础，具体识别及分支过程如下：

1. 遍历每个顶点，计算其度数。
2. 若某顶点的度数为奇数，则将其标记为奇度顶点并加入奇度顶点集合。
3. 遍历完成后，检查奇度顶点集合是否为空，若为空，说明图为欧拉图，可以直接计算欧拉回路；否则，需要添加边后再计算欧拉回路。

3.2.3 最小成本匹配

最小成本匹配是解决奇度顶点的关键步骤。其目的是在奇度顶点之间寻找代价最小的路径进行配对，从而将图转化为欧拉图或半欧拉图。

具体实现思路如下：

1. 计算图中任意两点之间的最短路径，构建一个奇度顶点的完全子图。
2. 在完全子图中进行最小成本匹配，使得奇度顶点之间的配对代价最小化。
3. 将配对路径的边加入原始图中，增加边或者边的重复次数，使得所有顶点的度均为偶数。

最短路径计算常用 Floyd-Warshall 或 Dijkstra 算法计算任意两点之间的最短路径。

匹配时采用 Edmonds Blossom 算法或匈牙利算法进行最小权完美匹配。

3.2.4 构建欧拉回路

在完成最小成本匹配后，图已经转化为欧拉图，接下来需要构建一条欧拉回路，以获得最终的最优路径。

具体实现思路如下：

1. 从任意顶点出发，不断沿着未遍历的边行进，直到返回起点，生成一条欧拉回路。
2. 如果在遍历过程中发现尚有未遍历的边，则以未遍历边为起点继续构建子回路，直到所有边都遍历完毕。
3. 最终回路由多个子回路拼接而成，即为求解结果。

3.3 程序实现流程图

以下是解决 CPP 的程序实现流程图，展示解决中国邮递员问题的各个阶段及逻辑分支。

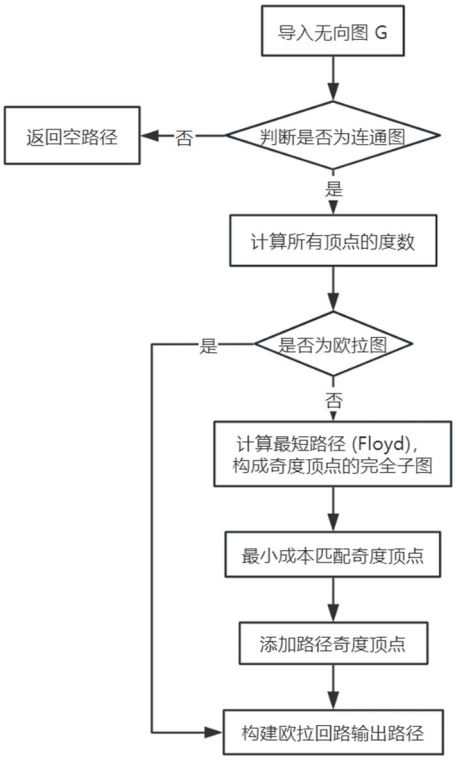


图 1 CPP 算法流程图

实际实现时，某个程序功能的实现可能也会附带另外一个功能的实现，为了减少多于内容，可能会将部分程序功能合并，使其不完全严格遵守上述流程，但程序实现基本思路不变。

3.4 具体示例分析

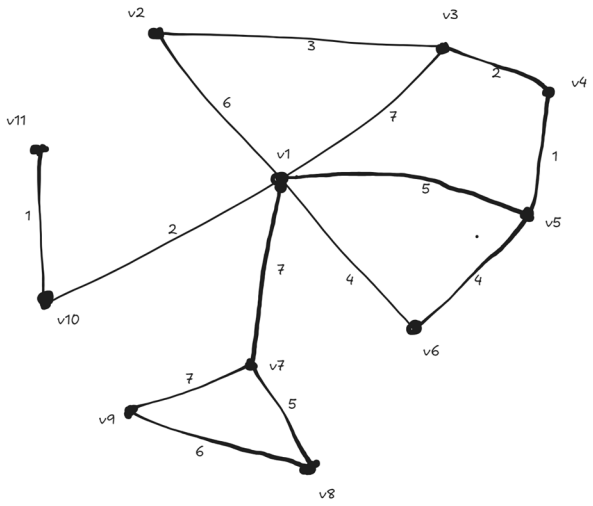


图 2 CPP 示例图

如上图，我们很容易发现 v3、v5、v7、v11 四个顶点都为奇数，该图显然不是欧拉图，因此，我们需要添加一些边来将其变为欧拉图。添加“边”是，并不是直接在两个奇度顶点之间任意添加一条边，实际添加两点间路径上的边的可重复遍历次数，为了方便处理，我们将其抽象化为一条边，该边的代价为两点间通路代价，为了最优，我们应该选择最短路径以减少总路径长度。

不过，如果奇度顶点很多，我们可能会面临很多选择，例如在上述例子中，因为有四个奇度顶点，可能会产生三种不同的情况：

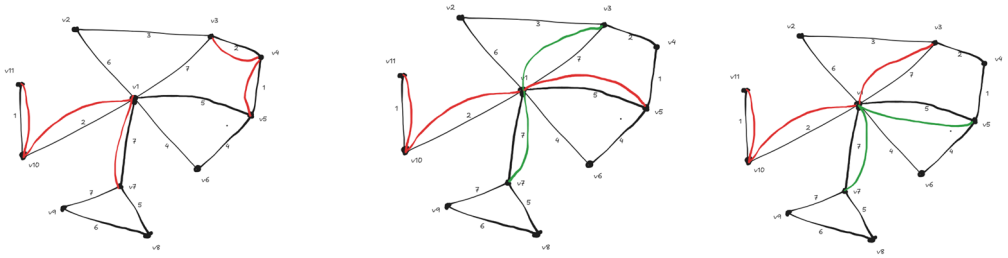


图 3 可能的添加边的方法

显然，这三种情况中只有第一种情况符合我们的要求，因为只有在这种情况之下，构建新边的代价是最小的，即我们需要实现极度顶点间最小权完美匹配，使得新形成的欧拉回路代价最小。在添加边完成后，就可以构建欧拉回路解决 CPP。

四、数据结构与算法实现分析

本部分详细介绍解决中国邮递员问题（CPP）的数据结构的选择、主要算法的实现逻辑。

4.1 数据结构

4.1.1 邻接矩阵

邻接矩阵用于存储图中顶点之间的边权值，方便查询任意两点之间的路径长度。之所以选择邻接矩阵而不选用邻接表，主要是出于一下考虑：

1. 在中国邮递员问题中，常需要处理稠密图。邻接矩阵适用于稠密图，空间复杂度为 $O\left(V^2\right)$，能高效表示任意两点之间的连通性。而邻接表更适用于稀疏图，在稠密图中由于链表存储过多边，可能导致内存分散，查询速度下降。

2. 邻接矩阵允许在常数时间 $O(1)$ 内查询任意两点是否存在边及其权值，而邻接表需要遍历链表或向量来查找特定边，查询复杂度为 $O(n)$。

邻接矩阵具体采用二维动态数组的方式表示，借助 vector 容器来存储，以下为一个具体示例的定义：

```
vector<vector<ll> > W = {
{0, 2, INF, 1},
{2, 0, 3, INF},
{INF, 3, 0, 4},
{1, INF, 4, 0}
};
```

其中，INF 为一个极大值，表示路径不存在。具体操作过程中，可以通过改变 vector<vector<ll> > W 中的具体内容来测试多个图，也可以允许输入来进行图的具体内容填充。为了简便起见，本程序采用前者。

4.1.2 其他数据结构

为了使程序功能可以正常实现，本程序还采用了多种辅助数据结构，比如：

vector<vector<ll> > shortest_path; 任意两点之间的最短路径矩阵

vector<vector<ll> > edge_repeat_num; 边的重复次数，用于 Hierholzer 算法

vector<ll> degree; 存储每个顶点的度数

vector<ll> odd_vertices; 存储奇度顶点

unordered_map<ll, unordered_map<ll, string> > path; 存储两点间最短路径

string min_cost_path; 最小成本路径，即完成构建的欧拉回路

同时，还提供数组、队列、栈等数据结构，以满足 Edmonds 算法最小权完美匹配、Hierholzer 算法寻找欧拉回路的需求。

4.2 主要算法说明

由于算法实现代码较多，此处不贴出详细代码，只介绍思路。具体代码见 cpp 附件。Hierholzer 算法实现逻辑与 3.2.4 逻辑基本相似，这里不在赘述。

4.2.1 Floyd-Warshall 算法

Floyd-Warshall 算法是一种解决任意两点间最短路径问题的经典动态规划算法。其核心思想是通过逐步引入中间顶点，不断更新路径长度，从而求出图中所有顶点对之间的最短路径，同时可以判断图是否连通。算法实现基本思路如下：

1. 构建一个邻接矩阵形式的路径长度表（即 shortest_path），初始化为图的边权值，表示顶点之间的直接路径长度。如果两点之间没有直接路径，则设为无穷大（INF）。

2. 遍历所有可能的中间顶点 k，对任意一对顶点 i 和 j，如果 i 通过 k 到达 j 的路径比直接从 i 到 j 更短，则更新最短路径并使用辅助矩阵 last_nods 记录路径中每个顶点到达目标顶点时的最后一个中间顶点。

3. 在计算完成后，检查所有顶点对之间的路径长度。如果存在任意一对顶点之间的最短路径仍为无穷大，说明图不连通，返回 false。

4. 借助 ast_nods，通过逐步回溯的方式构建完整路径。

4.2.2 Edmonds Blossom 算法

Edmonds Blossom 算法用于解决最小成本完美匹配问题，其核心目标是在图中奇度顶点之间进行最小成本匹配，使得图可以以最小的代价转化为欧拉图，便于后续构建欧拉回路。具体实现思路如下：

1. 识别图中所有奇度顶点，构建一个完全子图，每对奇度顶点之间的边权为二者之间的最短路径长度（通过 Floyd-Warshall 算法计算）。

2. 尝试通过寻找增广路径来扩展当前的匹配，从而降低匹配成本。

3. 当在增广路径中发现奇环（长度为奇数的环）时，将其收缩为一个顶点，暂时忽略环内的细节，继续执行匹配，从而有效地处理奇环问题，避免匹配进入死循环。

4. 找到增广路径后进行路径回溯，更新匹配状态，并扩展匹配集合，最终消除所有奇度顶点。

五、实验与分析

在本研究中，我们基于中国邮递员问题（CPP）提出的求解策略，通过 C++ 实现了路径规划算法。为了验证算法的有效性，我们进行了若干实验。

5.1 实验环境

编程语言：C++20
文件编码：UTF-8
操作系统：Windows 11
集成开发环境：Clion 2024
工具集：MinGW 11.0 w64

5.3 实验目的

验证程序的正确性，判断程序找到的路径是否为最优路径，以对程序的有效性做出评估。

5.4 实验过程

本实验主要采用五组不同的实验数据，其中一组非连通图、两组组欧拉图（一组为完全图）、两组非欧拉（一组为普通连通图，一组为完全图）；通过将程序运行结果与实际最优结果进行比较，验证程序在复杂性不同的图中的有效性表现。

本实验主要采用测试样例数据如下：

	0	**1**	**2**	**3**	**4**
0	0	4	∞	∞	∞
1	4	0	∞	∞	∞
2	∞	∞	0	3	∞
3	∞	∞	3	0	∞
4	∞	∞	∞	∞	0

表 1 测试用例 1（非连通图）

	0	**1**	**2**	**3**	**4**
0	0	2	4	∞	∞
1	2	0	1	4	∞
2	4	1	0	3	2
3	∞	4	3	0	5
4	∞	∞	2	5	0

表 2 测试用例 2（欧拉图）

	0	**1**	**2**	**3**	**4**
0	0	2	7	4	3
1	2	0	6	5	8
2	7	6	0	3	2
3	4	5	3	0	9
4	3	8	2	9	0

表 3 测试用例 3（欧拉图，完全图）

	0	1	2	3	4
0	0	∞	4	2	3
1	∞	0	∞	4	∞
2	4	∞	0	3	2
3	2	4	3	0	4
4	3	∞	2	4	0

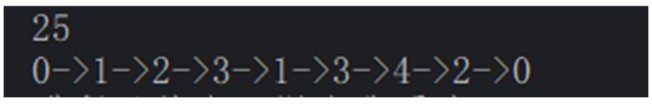
表 4 测试用例 4（非欧拉图，连通图）

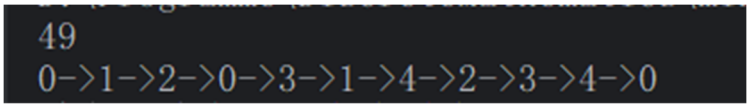
	0	1	2	3	4	5
0	0	3	5	7	2	8
1	3	0	4	6	1	7
2	5	4	0	3	6	2
3	7	6	3	0	4	5
4	2	1	6	4	0	3
5	8	7	2	5	3	0

表 5 测试用例 5（非欧拉图，完全图）

5.5 实验结果

对于测试样例中的连通图，程序可以找出最短路径，成功解决 CPP；对于非连通图，可以输出相关错误信息。具体实验结果依次如下：





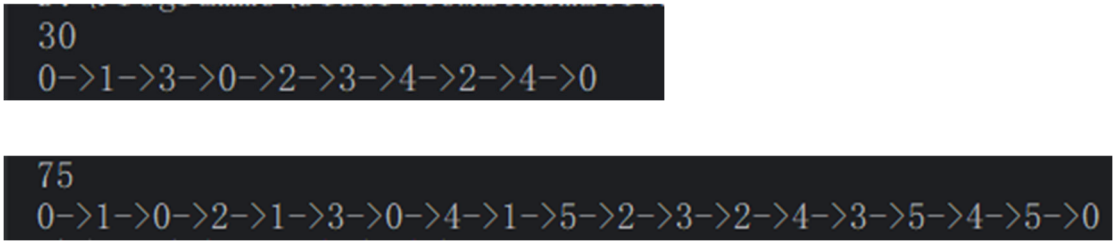


图 4 五种数据集实验结果

六、结论与展望

本文基于中国邮递员问题（Chinese Postman Problem，CPP）实现了一个高效的路径规划算法，成功地解决了不同类型图中的路径优化问题。通过借助 Floyd-Warshall 算法、Edmonds 算法和 Hierholzer 算法，能够很好地计算最短路径、解决奇度顶点匹配问题以及构建欧拉回路。实验结果表明，该算法能够有效处理不同类型的图，包括非连通图、欧拉图和非欧拉图，成功找出最优解，表现出良好的适用性。

不过，随着图规模的增加及实际应用场景的复杂化，该算法的消耗时间会急剧增加，有时难以保证效率，因此存在一些需要进一步研究的方向。未来的研究可以聚焦于如何通过并行计算、分布式计算或图算法优化等手段，提升算法在大规模图上的计算效率。通过算法优化，可以有效降低计算复杂度，提高大规模图计算的实时性和可操作性。

此外，在许多实际问题中通常涉及到的不是简单的无向图，例如在实际交通系统中，通常还会出现单向道，即只允许单向通行。这时候，将 CPP 问题拓展到有向图和混合图中成为另外一个进一步研究的方向，以适应更广泛的应用场景。

同时，在动态环境下的路径规划问题将是一个重要的研究方向。在现实应用中，图的结构可能会随着时间的推移而发生变化可能影响最短路径的选择。因此，如何在动态环境中进行路径规划，实时调整路径，以应对路网的变化，是一个值得研究的课题。

参考文献

[1] 管梅谷. 奇偶点图上的作业法[J]. 数学学报, 1960, 10: 263-266.
[2] Edmonds J, Johnson E. Matching, Euler tour and the Chinese Postman Problem[J]. Mathematical Programming, 1973, 5: 88-124.