第十一届中国大学生服务外包创新创业大赛

A12 基于算法的配送路线优化系统

团队编号： 1905397

团队名称： 逮虾户

团队成员： 李琦 徐乐乐

提交时间： 2020.05.31

目录

1 项目概述

1.1 项目背景

随着信息技术的发展，现代物流作为“第三个利润源泉”是一种先进的组织方式和管理技术，已被世界各国广泛采用，并形成产业化，在国民经济中发挥越来越重要的作用，这一管理技术，正受到日益广泛的重视，并面临巨大的发展机遇，在现代物流中，配送是一个重要的与消费者直接相连的环节，一方面可以体现企业的核心竞争力，另一方面通过线路优化，可以提高企业的运作效率，降低配送成本，实现物流科学化。目前市场上仓储类管理系统已经比较完善，但对于不同需求的路线规划还存在一定的短板，基于当前站好“最后一岗”的要求，引用路线优化算法，开发出具有实用性，先进性，高可靠性的路线规划系统。

由单一起点出发，配送多个终点后再回到起点，根据车辆数量，承载限制，不同车辆服务成本、运行里程限制等条件选择最优运输路径，使成本最小化，配送订单最大化，满载率最大化（如由一个配送中心向各个销售点配送货物，通过算法确定配送中心每辆车的配送方案，包括配送至哪个客户，配送量，下一个配送目的地）。

1.2 项目目标

通过算法（蚁群算法、遗传算法等，可任意选择）模型构建，利用计算机语言进行开发（不限定开发语言），最终可通过设置配送点坐标或距离，各配送点业务量，车辆装载等参数，程序能自动计算出最优路线顺序。

2 总体架构

通过分析系统功能需求，考虑系统的方便性和可移植性，本系统采用 B/S 架构，系统总体架构如图 2-1 所示：

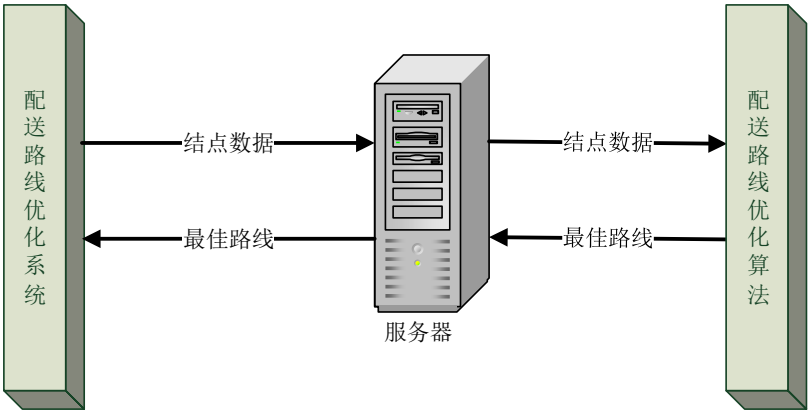

图 2-1 系统总体架构

前端借助 Web 技术搭建用户界面，实现数据及界面的良好展示；

服务端使用 Flask 框架搭建，接收用户数据请求及路线规划结果的回执；

配送路线优化算法以蚁群算法为核心优化配送路线。

3 系统功能清单

3.1 需求分析

现有一个配送网络如图 3-1 所示，图中 P 为配送中心，其余 A-I 为客户的接货点，所有客户的位置距离固定，各边上的数字为公里数，括号内的数字为需输送到各接货点的货物量，单位为吨。假设该配送中心有最大装载重量为 2 吨和 5 吨的两种货车，并限制车辆一次运行路线距离不超过 35 公里，每个派送点只由一辆车服务一次，每辆车只能服务一条路线，车辆一律由配送中心出发，完成任务后返回配送中心，快递车辆配送过程中无装货，只考虑卸货。每个点卸货时间固定为 5 分钟，车辆每小时行驶距离为 10 千米，每个派送人员工作时间为 8 小时，请参考内容信息通过算法和程序计算需要的车辆数，最优路径及配送时间。

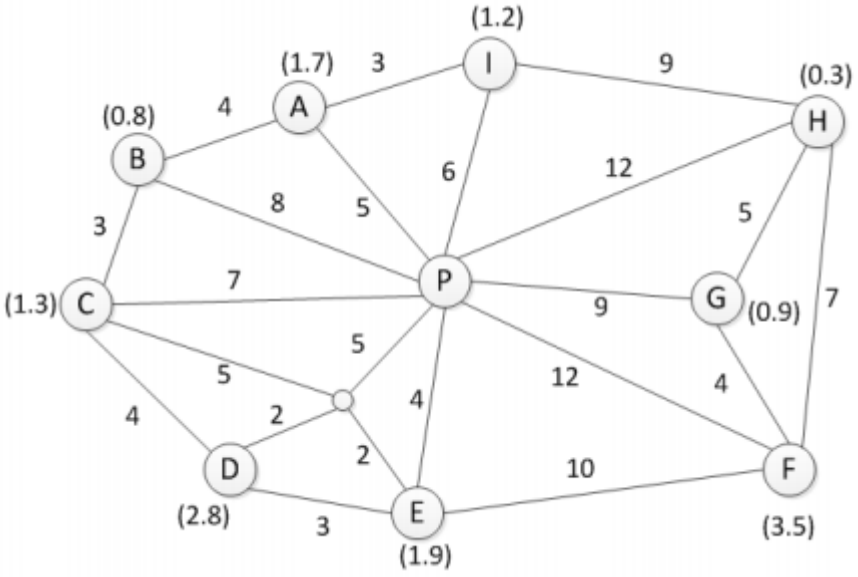

图 3-1 配送网络图

3.2 功能结构设计

配送路线优化系统功能结构如图 3-2 所示：

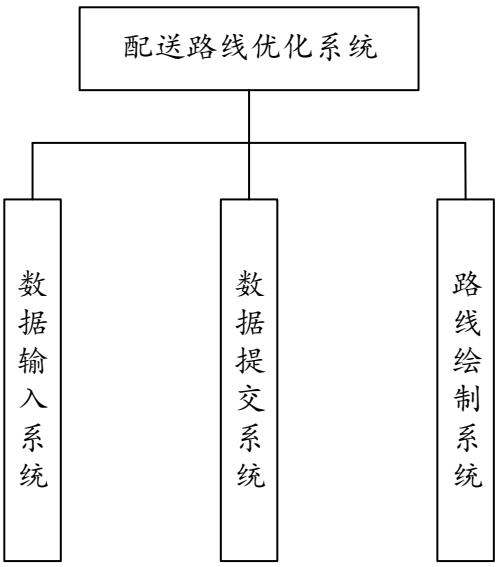

图 3-2 配送路线优化系统功能图

数据输入系统是负责相关数据的输入和提取工作，由系统将数据逐条输入或解析指定格式的相关文件并规格化处理，以便进行下一步处理的工作。

数据提交系统是负责将数据输入系统规格化后的数据提交到后台服务系统的工作，将相关数据提交到后台并进行线路规划任务。

路线绘制系统是负责绘制线路规划结果的工作，接受线路规划结果，处理得当后在前端页面进行规划路线的绘制和展示工作。

4 运行环境

4.1 硬件环境

硬件环境指计算机及其外围设备组成的计算机物理系统，本项目使用 B/C 架构设计模式，减少硬件设备的投入。结合本系统硬件环境，所需硬件环境配置如表 4-1 所示：

表 4-1 硬件环境配置表

序号	产品	产品型号	备注
1	PC Server	无	无

4.2 软件环境

软件环境是指建立在硬件系统之上的软件系统，它与硬件关系十分密切。硬件系统只有配合相关的软件系统，才能正常运行，系统软件平台屏蔽了硬件的实现细节，使得建立在系统软件基础之上的应用软件不需要考虑硬件的实现细节。结合本系统软件环境，所需软件环境配置如表 4-2 所示：

表 4-2 软件环境配置表

序号	产品	产品名称	备注
1	操作系统	Windows/Linux/Unix 等主流操作系统均可	无
2	Python	Python 3	安装 Flask 插件

4.3 系统开发环境

系统开发环境所需配置如表 4-3 所示：

表 4-3 系统开发环境配置表

序号	软件	版本	备注
1	Sublime	3	无
2	Office	2016	无
3	Visio	2013	无

5 系统关键技术

5.1 蚁群算法

配送路线优化问题与多回路运输问题（Vehicle Routing Problem, VRP）在本质上是相似，甚至可以说是相同的。从图论的角度来看，该类问题的实质是在一个带权完全无向图中，找一个权值最小的 Hamilton 回路。该问题的可行解是所有顶点的全排列，随着顶点数的增加，会产生组合爆炸，它是一个 NP 完全问题。

在早期，各国学者同时精确算法求解该类问题，比如线性规划法、动态规划法等。精确算法的优势在于它总能获取该类问题的最优解，是现实配送问题的最优配送路线。但精确算法随着配送结点的增加，问题规模的扩大，精确算法的时间复杂度呈现爆炸增长的趋势，精确算法对于规模较大的 VRP 问题将变得无能为力。

近似算法或启发式算法（Heuristic Algorithm）是相当于最优化算法提出的。启发式算法是基于直观或经验构造的算法，在可接受的时间和空间下给出待解决组合优化问题的每一个实例的可行解，但该可行解不一定是最优解且与最优解的判离程度不可估计。启发式算法的缺点在于不能得出问题的最优解，无法估计最终结果的偏离程度，但其能够在优先的时间和空间下得出较为贴近最优解的可行解。

蚁群算法是一种基于种群的搜索算法，它模拟真实蚂蚁的行为，寻找最短路径。蚂蚁的运动是基于信息素（pheromone）的，信息素由其他蚂蚁在路径上沉积。蚁群算法被广泛应用于解决寻路、调度、模糊控制、网络路由、图像处理等问题。蚁群算法首先在搜索空间的不同位置防止若干“蚂蚁”，这些“蚂蚁”用于构建候选解决方案。式(5-1)表示概率转移规则，即在时间 t 时，“蚂蚁”k 基于启发式信息和信息素浓度，下一步移动的概率，这是每一只“蚂蚁”的局部更新策略。

$$p_{ij}^{k}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}\times[\eta_{ij}(t)]^{\beta}}{\sum_{s\epsilon allow_k}[\tau_{is}(t)]^{\alpha}\times[\eta_{is}(t)]^{\beta}}, j\epsilon allow_k\\ 0, j\notin allow_k\end{cases} \quad (式\ 5\text{-}1)$$

其中$p_{ij}^{k}(t)$表示t时刻蚂蚁k从城市i向城市j转移的概率。$\tau_{ij}(t)$表示t时刻城市i与城市j连接路径上的信息素浓度。$\eta_{ij}(t)$为启发函数，表示蚂蚁从城市i转移到城市j的期望程度。α和β分别表示信息素因子和启发函数因子。$allow_k$表示可以蚂蚁k尚未经过的城市的集合。

在所有的蚂蚁完成一步移动后，信息素的值将会按照式(5-2)更新，其中ρ表示信息素的挥发程度，$\Delta\tau_{ij}^{k}$表示第k只蚂蚁在城市i与城市j连接路径上释放信息素而增加的信息素浓度，$\Delta\tau_{ij}$表示所有蚂蚁在城市i与城市j连接路径上释放信息素而增加的信息素浓度。

$$\begin{cases}\tau_{ij}(t+1)=(1-\rho)\times\tau_{ij}(t)+\Delta\tau_{ij}, 0<\rho<1\\ \Delta\tau_{ij}=\sum_{k=1}^{m}\Delta\tau_{ij}^{k}\end{cases} \quad (式\ 5\text{-}2)$$

而蚂蚁经过路径后所遗留的信息素浓度与路径的长度相关，蚂蚁经过路径后遗留信息素将按照式(5-3)更新，其中Q表示表示信息素常数，L_k表示第k只蚂蚁经过路径的总长度。

$$\Delta\tau_{ij}^{k}=\begin{cases}\dfrac{Q}{L_k}, 若蚂蚁k从城市i到了城市j\\ 0, 其他\end{cases} \quad (式\ 5\text{-}3)$$

配送车辆和出发的物流中心(多车型、多物流中心)随机生成，配送车如果回到出发点就完成 1 次配送过程，如果全部客户点完成配送则完成 1 次完整迭代。

配送车辆在选择下一个城市时不仅要考虑蚁群算法中城市的转移概率还要考虑车辆行驶距离限制和运载能力限制，不仅保证有下一个城市的配送空间，还得在配送完下一个客户点后能回到出发点。

为使最后结果能在较短路径下获得更高的满载率，每次迭代的量化指标按照式(5-4)计算，其中$load_rate_i$表示配送车第 i 次配送的满载率，L_i表示蚂蚁(配送车)第 i 次配送行驶的距离。

$$\text{Indicator}=\frac{1}{n}\sum_{i=1}^{n} load\_rate_i+\frac{100}{\sum_{i=1}^{n} L_i} \quad \text{(式 5-4)}$$

本次路线规划算法的流程图如图 5-1 所示：

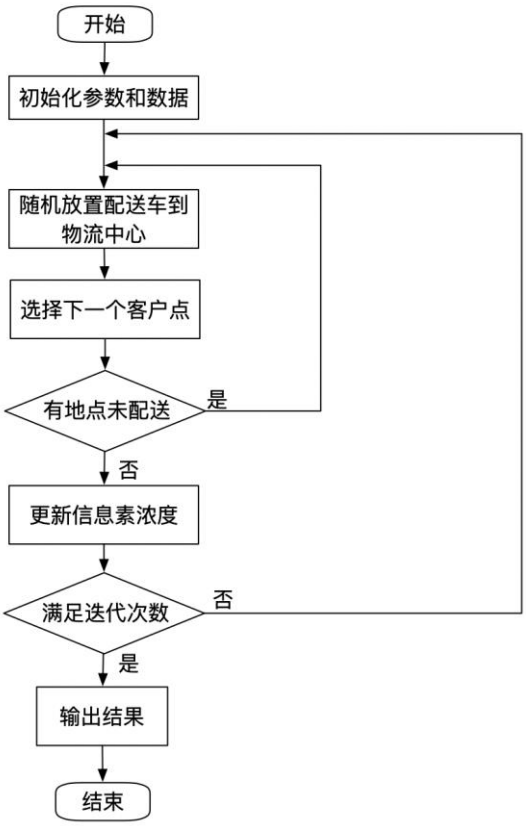

图 5-1 蚁群算法流程图

5.2 Flask 框架

Flask 是一个轻量级的可定制框架，使用 Python 语言编写，较其他同类型框架更为灵活、轻便、安全且容易上手。它可以很好地结合 MVC 模式进行开发，开发人员分工合作。Flask 具有很强的定制型，可以在保持核心功能简单的同时实现功能的丰富与扩展。

Flask 主要包括 Werkzeug 和 Jinja2 连个核心函数库，他们分别负责业务处理和安全方面的性能，这些基础函数为 web 项目开发提供了丰富的基础组件。Werkzeug 库十分强大，功能比较完善，支持 URL 路由请求集成；Jinja2 库支持自动 HTML 转移功能，能够很好的控制外部黑客的脚本攻击。

本系统选用 Flask 作为后端框架，主要原因有如下 3 点：

1) 后续的路径规划算法研究，主要开发语言也是应用 python，整个系统统一开发语言，便于开发和维护；

2) Flask 因为灵活、轻便和高效的特点被业界认可，拥有内置服务器和单元测试，便于学习掌握；

3) Flask 中用于灵活的 Jinja2 模板引擎，提高了前端代码的复用率，这样可以提高开发效率和有利于后期开发还维护。

Flask 的基本模式为在程序里将一个视图函数分配给另一个 URL，每当用户访问这个 URL 时，系统就会执行给该 URL 分配好的视图函数，获取函数的返回值并将其显示到浏览器内，其工作过程如图 5-2 所示：

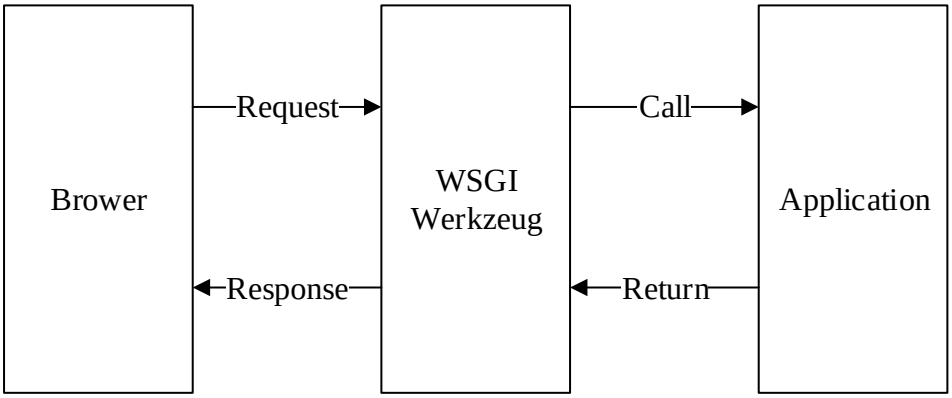

图 5-2 Flask 框架工作过程

6 组织保障

一个优秀的项目组织队伍对于项目的成功与否十分关键，根据本系统的管理需求，将组内人员分工如下：

职务	职责	人员
前端设计	在本系统中负责页面的界面设计与开发工作	李琦
后端设计	在本系统中负责后端服务的开发与优化工作	李琦
算法设计	在本系统中负责路线优化算法的测试与编写工作	徐乐乐

7 实验结果

7.1 路径规划

（注：以下实验皆在同一数据集下所得，数据集中含有配送点 20 个，物流中心 2 个，车型 2 种）

表 7-1 实验结果

实验组别	α	β	$1-\rho$	Q	平均载货率	平均行驶距离	平均量化指标
1	1	2	0.5	100	0.9875	121.1	1.81382
2	2	2	0.5	100	0.9875	125.1	1.78856
3	3	2	0.5	100	0.9875	130.6	1.75566
4	4	2	0.5	100	0.9875	133.2	1.73898
5	5	2	0.5	100	0.9875	134.7	1.71676
6	1	1	0.5	100	0.9875	122.5	1.80498
7	1	3	0.5	100	0.9875	121.5	1.81322
8	1	4	0.5	100	0.9875	124.4	1.79193
9	1	5	0.5	100	0.9875	121.3	1.81264
10	1	2	0.4	100	0.9875	123.8	1.79584
11	1	2	0.6	100	0.9875	120	1.82209
12	1	2	0.7	100	0.9875	119.1	1.82763
13	1	2	0.8	100	0.9875	119.6	1.82413
14	1	2	0.9	100	0.9875	119.2	1.82705

注：α为信息素因子，β为启发函数因子，ρ为信息素挥发程度，Q 为信息素常数，

$量化指标 = 载货率 + \frac{100}{行驶距离}$

7.2 界面操作展示

提交数据后界面展示如图 7-1 所示：

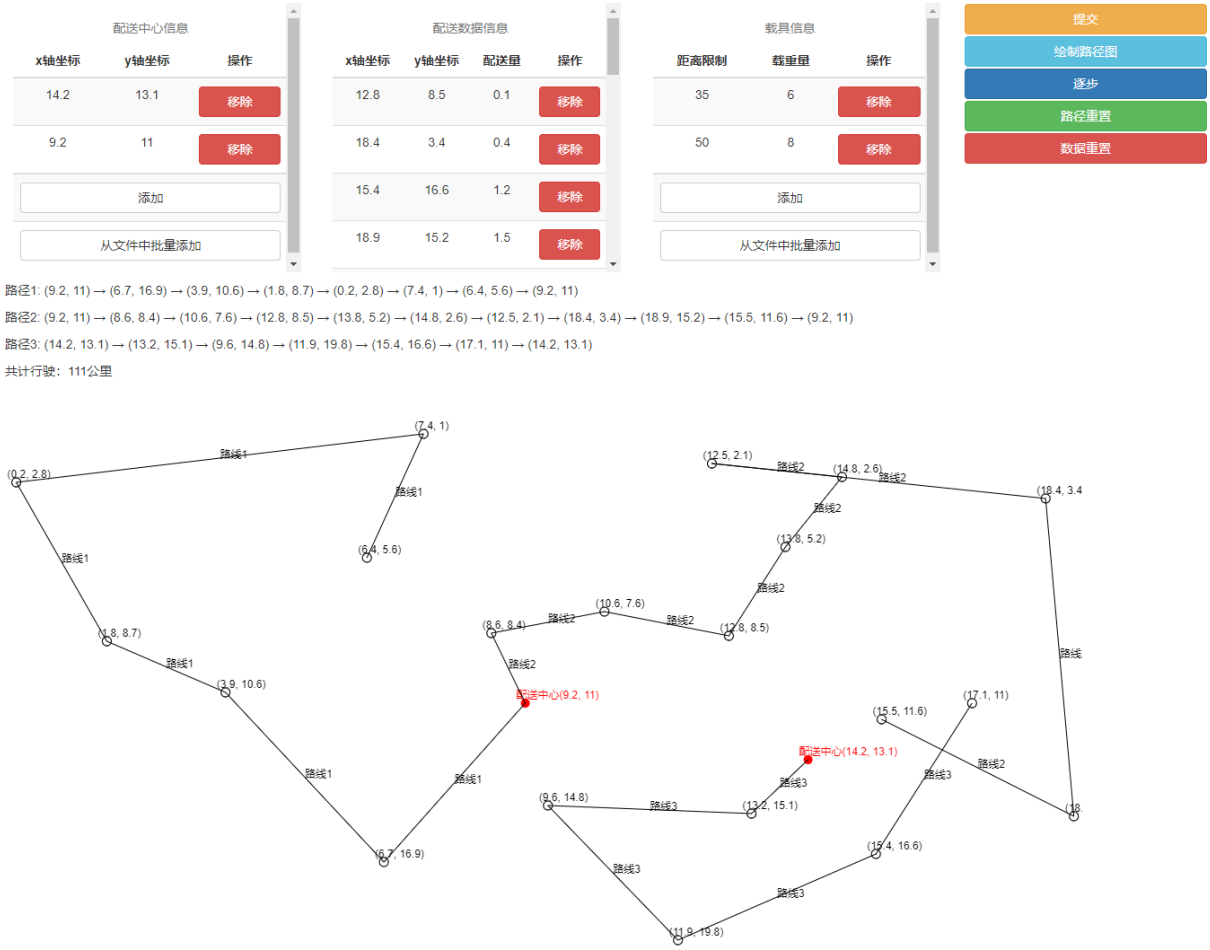
配送中心信息
x轴坐标
y轴坐标
操作
14.2
13.1
移除
9.2
11
移除
添加
从文件中批量添加
配送数据信息
x轴坐标
y轴坐标
配送量
操作
12.8
8.5
0.1
移除
18.4
3.4
0.4
移除
15.4
16.6
1.2
移除
18.9
15.2
1.5
移除
载具信息
距离限制
载重量
操作
35
6
移除
50
8
移除
添加
从文件中批量添加
提交
绘制路径图
逐步
路径重置
数据重置
路径1: (9.2, 11) → (6.7, 16.9) → (3.9, 10.6) → (1.8, 8.7) → (0.2, 2.8) → (7.4, 1) → (6.4, 5.6) → (9.2, 11)
路径2: (9.2, 11) → (8.6, 8.4) → (10.6, 7.6) → (12.8, 8.5) → (13.8, 5.2) → (14.8, 2.6) → (12.5, 2.1) → (18.4, 3.4) → (18.9, 15.2) → (15.5, 11.6) → (9.2, 11)
路径3: (14.2, 13.1) → (13.2, 15.1) → (9.6, 14.8) → (11.9, 19.8) → (15.4, 16.6) → (17.1, 11) → (14.2, 13.1)
共计行驶：111公里
(7.4, 1)
(0.2, 2.8)
(6.4, 5.6)
(1.8, 8.7)
(3.9, 10.6)
(6.7, 16.9)
配送中心(9.2, 11)
(12.5, 2.1)
(14.8, 2.6)
(18.4, 3.4
(13.8, 5.2)
(10.6, 7.6)
(8.6, 8.4)
(12.8, 8.5)
(15.5, 11.6)
(17.1, 11)
配送中心(14.2, 13.1)
(9.6, 14.8)
(13.2, 15.1)
(15.4, 16.6)
(11.9, 19.8)
路线1
路线2
路线3

图 7-1 界面展示