2021 第六届“数维杯”大学生
数学建模竞赛论文

题　　目　　**外卖送餐骑手的送餐方案设计**

摘　要

近年来，在“互联网+”的浪潮下，外卖行业一直保持着突飞猛进的发展状态。本文基于将理论与实际相结合、规范分析和实证分析相结合、横向比较和纵向比较相结合的基础上，为外卖送餐制定了合理的配送时长和奖惩措施，并提供了考虑极端天气的通用方案，以及依靠博弈论实现共赢战略，同时还提出了长距离跨区集中配送优化算法，缓解外卖行业危机。

针对问题 1，我们提出了层次分析法模型。该模型以外卖骑手交通事故率和外卖骑手平均收入等数据为依据，通过层次分析法构建外卖骑手订单完成质量评价指标体系。首先，建立递阶层次结构模型，然后借助 MATLAB 计算权重，并在制定的骑手合理配送时长设计方案的基础上提出相应的完成质量奖惩措施。

针对问题 2，我们构建了虚拟场景和超时惩罚目标函数，生成了骑手评分表。

针对问题 3，我们采用模糊控制建立额外订单配送费用收取金额、订单配送时长及订单配送提成设计方案的关系。首先，利用 MATLAB 中的 FUZZY LOGIC 模糊逻辑工具箱进行模糊推理，然后通过反模糊化转换为精确输出量，最后确认三者的关系。

针对问题 4，我们提出了采用博弈论结合模糊控制建立订单配送费、订单配送时长及用户幸福指数的关系。首先，定义博弈的特征函数、得益向量和博弈配置，然后通过 FUZZY LOGIC 模糊逻辑工具箱和反模糊化后的结果建立订单配送费、订单配送时长及用户幸福指数的关系。

针对问题 5，我们提出了长距离跨区集中配送优化算法。首先整合多名用户在外卖平台的下单信息，通过集中送餐路线模型假设，用 MATLAB 确定最短路径规划模型；然后通过禁忌搜索算法求解，在确定模型及假设成立的条件下求解外卖配送过程中最短路径。外卖平台根据已收集信息派单给配送员并给配送员提供最短路径。外卖配送员根据外卖平台提供的路径前往各家餐饮企业取餐。最后用户收到外卖商品，实现长距离跨区集中配送。

关键词：层次分析法；模糊算法；时间窗；MATLAB；集中配送算法

目　录

一、问题重述

外卖业务已经成为了大城市上班族每日生活中不可或缺的一部分。根据“美团”2020 年 6 月发布的《2019 中国即时配送行业发展报告》中显示，2019 年我国即食配送业务订单规模达到 182.8 亿单，比 2018 年增长了 37%，即时配送行业用户达到 4.21 亿人，比 2018 年增加了 17.6%。面对巨大的订单量及用户群体，外卖骑手的数量也呈现一种直线上升的趋势。2019 年通过“美团”的骑手总数达到 399 万人，同比增长了 23.3%。根据第三方平台数据显示，2020 年外卖骑手的数量突破 438 万。

在复杂的国内与国际经济环境及我国庞大的人口规模下外卖骑手间的竞争变得异常激烈。这给予了平台更多的订单配送提成单价压缩空间。其中常以拼命压缩配送时间和延长上班时间来换取较高收入的骑手为参照，以骑手配送效率低下为理由逐步压缩外卖骑手的订单配送提成。

在外卖骑手不断抱怨的同时商家方面也对于平台高达 20 个点抽成叫苦不迭，其中部分商家为了盈利出现了调理包加热、食材不新鲜、包装材料廉价等问题，最终消费者也成为了受害群体。

上述种种事实表明外卖行业中存在着严重的内卷现象。数据分析至关重要，但利用数据分析来掠夺弱势群体少有的财富和时间并不可取。请你通过数学建模的方法解决外卖平台、骑手、商家与消费者之间的如下问题：

问题 1：请充分考虑骑手的骑行安全与高质量服务等因素后，试制定一个合理的骑手配送时长设计方案，并提供对应的完成质量奖惩措施。

问题 2：您能否提出一种考虑多种因素在内的静态和动态订单配送提成定价与奖惩策略，它在不显著增加订单总体配送费用与总体配送效率的基础之上，能够使得骑手总体的满意度最高。

问题 3：随着全球极端气候的频发（如暴雨、暴雪及大风等），我们需要考虑极端气候条件下的额外订单配送费用收取金额、订单配送时长设计及订单配送提成设计方案。您能否提供一个考虑上述因素的通用方案？

问题 4：共享、共赢依然成为了新的经济发展模式。在外卖骑手的送餐危机中，外卖平台、骑手、商家与消费者之间显然没有实现这种共享与共赢的目标。

这不仅会导致内需的下降，也可能会影响到众多群体的幸福指数。您能否从博弈理论角度出发提出一个可行的共赢方案？

问题 5：目前外卖订单的配送距离约束相对较少且跨区域的配送较难实现，您能否设计出更长距离和跨区域的外卖订单的配送模式及其订单配送提成定价策略？

二、问题分析

2.1 问题 1 的分析

本章主要以外卖骑手交通事故率和外卖骑手平均收入等相关数据为依据，从外卖骑手自身利益的角度出发，考虑安全骑行意识和服务好评率这两项指标，依据层次分析法来构成外卖骑手订单完成质量评价指标体系。

2.2 问题 2 的分析

本章注重外卖人员的送达时间，提出了构建虚拟场景和超时惩罚目标函数，首先根据偏离上述最小惩罚成本，做个线性奖惩，然后根据这个函数，生成配送员评分表，最后将评分表动态评分展现给顾客，比如评分好的优先分配给难以忍受超时的顾客。

2.3 问题 3 的分析

问题 3 的分析，问题 3 要求我们建立考虑极端气候额外订单配送费、订单配送时长及提成方案的综合关系。首先用 MATLAB 中 FUZZY LOGIC 模糊逻辑工具箱，通过定义隶属函数将输入变量 x、t 输出变量 y 进行模糊化，为后续模糊推理做准备。然后根据存储隶属度矢量或隶属度函数的数据库以及存放所有模糊控制规则进行模糊推理。之后通过反模糊化。将模糊输出量经过与模糊推理过程得到的结果相反的映射作用转换为精确输出量。最终得出额外订单配送费、订单配送时长及提成的模糊逻辑关系曲面用于通用方案设计。

2.4 问题 4 的分析

问题 4 的分析，问题 4 要求我们以共享与共赢的目标，协调好外卖平台、骑手、商家与消费者之间关系。首先定义博弈的特征函数、得益向量和博弈配置。之后通过 Shapley 值得出博弈的解。最后将所定义 x、t、y 的参数输入 MATLAB 中 FUZZY LOGIC 模糊逻辑工具箱，通过反模糊化后的结果建立订单配送费、订单配送时长及用户幸福指数的关系。

2.5 问题 5 的分析

问题 5 要求我们打破外卖订单的配送距离约束，实现长距离跨区域的配送。我们采用长距离跨区集中配送优化算法。首先整合多名用户在外卖平台的下单信息，通过集中送餐路线模型假设，用 MATLAB 建立方程和约束条件以确定最短路径规划模型；然后通过禁忌搜索算法求解，在确定模型及假设成立的条件下求解外卖配送过程中最短路径。由计算方法规划出配送员进行取餐、送餐的最短路径。外卖平台根据已收集信息派单给配送员，并给配送员提供最短路径。外卖配送员根据外卖平台提供的路径前往各家餐饮企业取餐。最后用户收到外卖商品，实现长距离跨区集中配送。

三、模型假设

1. 问题一不考虑非外卖骑手自身行为引起的交通事故、极端天气和商家的出餐速度的影响。
2. 问题二中假设：

（1）配送中心、客户与餐馆的位置及其之间的距离已知；

（2）各客户对货物的需求及对时间的限制已知，且为常数；

（3）配送车辆的最大载货量、最远行驶距离、行驶速度固定、已知，且为常数

（4）每个客户都能被车辆服务，且服务需求能够被满足；

（5）单份外卖重量已知；

（6）客户能接受的最晚送达时间、外卖超时单位惩罚费用已知，且为常数；

（7）装载外卖时不考虑车辆空间、餐品体积等因素；

四、定义与符号说明

表 4-1 定义与符号说明

符号定义	符号说明
CI	一致性指标
RI	随即一致性指标
CR	检验系数
X_{ij}	0-1 变量
d_{ij}	最短距离

五、模型的建立与求解

5.1 问题 1 的模型建立与求解

本章首先建立递阶层次结构模型，确认目标为骑手的工作期望，将骑手遵守交通规则意识、准时率和外卖完整度为指标，以外卖骑手交通事故率和月平均收入等相关数据构造判断矩阵进行权重评定，最终骑手可选择的方案为高收入、稳定和适中的工作方式，并在所制定的合理的骑手配送时长设计方案的基础上提出对应的完成质量奖惩措施。

5.1.1 层次分析法（AHP）模型的建立

1. 指标体系构建原则

层次分析法是将与决策相关的元素分解为目标、准则、指标等层次，并在此基础上进行定性与定量的决策方法。根据判断矩阵对每个元素进行重要性比较，通过数值进行量化，最后采取判断矩阵计算的方法确定全部因素的数值，得出重要性权重，从而得到权重值排序。构建对外卖骑手订单完成质量评价体系的关键就是保证评价模型的精确性，因此应对对外卖骑手订单完成质量的各个因素进行全面的分析评估。本文总结了以下在因素选择的过程中应遵守的原则性要求：

- 综合性原则。外卖骑手订单完成质量评价体系是一个有机整体，他应该是综合的、全面的。该评价体系应从外卖骑手订单完成质量的各个方面进行评价，筛选出有效的、有价值的影响因子，客观的反映出外卖骑手跑订单的主要特征。
- 科学性原则。对外卖骑手订单完成质量评价体系构建时，应有相应的理论指导，并结合科学的方法，如发放调查问卷，这样既能反映出被评价对象真实情况，也能在逻辑上严谨合理。
- 因素优化原则。外卖骑手订单完成质量的影响因素很多，通常先按照特性分出几个方面，再向下进行细分。影响因素并不是单独存在产生影响的，各个因素组成一个整体，既相互关联，又相互制约。为了能够准确地反映其对评价体系的影响，应当选择最具代表性的诸个影响因素进行分析。

2. 层次分析法（AHP）模型基本原理及步骤

层次分析法（AHP）是美国运筹学科学家、匹兹堡大学教授萨蒂创立的一种系统分析与决策的综合评价方法。AHP 的主要特点是通过建立层次结构将人为判断转化为几个因素重要性的比较，从而将难以量化的定性判断转化为重要性的比较。在很多情况下，决策者可以直接利用 AHP 进行决策，大大提高了决策的有效性、可靠性和可行性。AHP 将复杂问题分解为多个构成因素，并根据支配关系将这些因素构成层次结构。通过两两比较，确定了决策方案相对重要性的总排序。整个过程体现了人类决策思维的基本特征，克服了其他方法的缺点，避免了决策者的主观判断。

图 5-1 层次分析法（AHP）建立步骤示意图

AHP 模型建立过程主要分为以下四个步骤（如图 1 所示）：

Step1：通过分析系统中各因素之间的关系，运用 AHP 建立了系统的递阶层次结构模型，如图 2 所示，从上到下分别称为目标层、准则层、指标层和方案层。

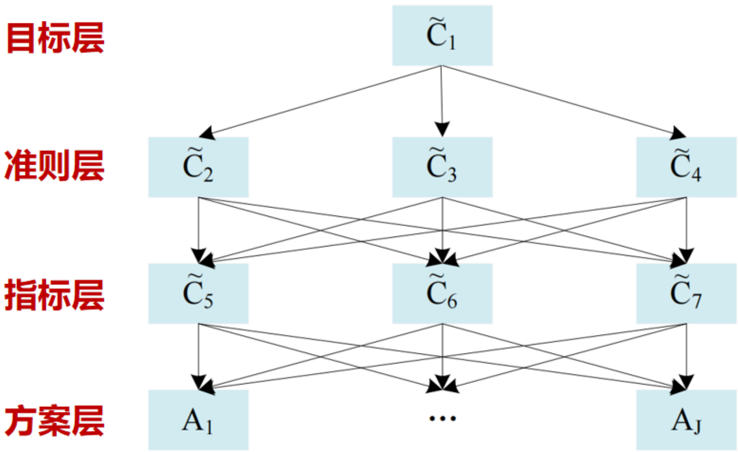


图 5-2 递阶层次结构模型

Step2：构造判断矩阵，将复杂的优选问题划分为多个层次的框架体系，利用九标度法构建多层次指标的判断矩阵。打分值及其含义如表 5-1 所示：

表 5-1 判断矩阵

标度值	含义解释
1	两个指标相比，重要性一样
3	两个指标相比，前者比后者稍微重要
5	两个指标相比，前者比后者明显重要
7	两个指标相比，前者比后者强烈重要
9	两个指标相比，前者比后者极端重要
2、4、6、8	上述相邻判断的中间值

Step3：计算特征向量并检验一致性。通过数学方法判断判断矩阵的一致性，只有一致性检验通过，才能避免相同重要等级的指标发生冲突，从而确保模型合理有效。

对应于判断矩阵最大特征根λ_{max}的特征向量，经归一化（使向量中各元素之和等于 1）后记为 W。W 的元素为同一层次因素对于上一层次因素某因素相对重要性的排序权值，这一过程称为层次单排序。能否确认层次单排序，则需要进

行一致性检验，所谓一致性检验是指对 A 确定不一致的允许范围。其中，n 阶一致阵的唯一非零特征根为 n；n 阶正互反阵 A 的最大特征根$\lambda \geq n$，当且仅当$\lambda = n$时，A 为一致矩阵。

为了衡量不一致程度，引入了矩阵的一致性指标 *CI* 和平均随机一致性指标 *RI*。计算两者的比值称为随机一致性比 *CR*，如果 *CR* 小于或等于 0.01，则认为矩阵的一致性是可以接受的，否则判断需要进一步调整。

首先，计算一致性指标 *CI*：

$$CI = \frac{\lambda - n}{n - 1}$$

CI=0，有完全的一致性；*CI* 接近于 0，有满意的一致性；*CI* 越大，不一致越严重。为衡量 *CI* 的大小，引入随机一致性指标 RI：

$$RI = \frac{CI_1 + CI_2 + CI_3 + \cdots + CI_n}{n}$$

其中，随机一致性指标 *RI* 和判断矩阵的阶数有关，一般情况下，矩阵阶数越大，则出现一致性随机偏离的可能性也越大，其对应关系如表 2：

表 5-2 平均随机一致性指标 *RI* 标准值

n	1	2	3	4	5	6	7	8	9	10	11	12
RI	0	0	0.50	0.87	1.12	1.27	1.35	1.42	1.45	1.49	1.52	1.56

考虑到一致性的偏离可能是由于随机原因造成的，因此在检验判断矩阵是否具有满意的一致性时，还需将 *CI* 和随机一致性指标 *RI* 进行比较，得出检验系数 *CR*，公式如下：

$$CR = \frac{CI}{RI}$$

一般，如果$CR < 0.1$，则认为该判断矩阵通过一致性检验，否则就不具有满意一致性，需要修改判断矩阵。

Step4：根据排序权值，计算各方案的复合排序权值，进行方案优化。

5.1.2 层次分析法（AHP）模型模型的求解

1. 递阶层次结构模型

针对问题一，本章首先建立递阶层次结构模型，确认目标为骑手的工作期望，将骑手遵守交通规则意识、准时率和外卖完整度为指标，以外卖骑手交通事故率和月平均收入等相关数据构造判断矩阵进行权重评定，最终骑手可选择的方案为高收入、稳定和自主的工作方式。

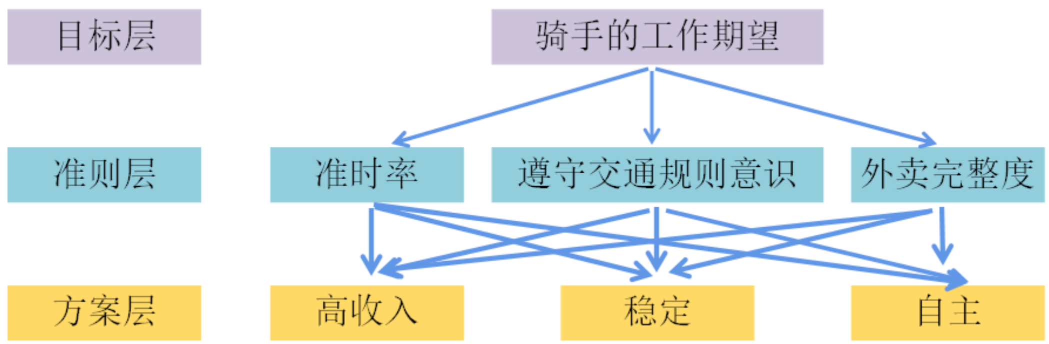


图 5-3 递阶层次结构模型示意图

2. 判断矩阵

根据笔者查到的外卖送餐数据结果组织，2018 年上半年整个上海外卖送餐行业所造成的交通伤亡事故共计 76 起，从平均天数上看每 2.5 天就会出现一例送餐员伤亡事故，并且“美团”和“饿了么”两大平台在交通伤亡事故中的占比均为 26%。笔者也从网上查到了诸多典型案例。2018 年初，一名“饿了么”平台送餐员骑电动车在送餐路途中与机动车发生碰撞，随后不幸身亡；而在 2018 年 4 月，一名达达送餐员同样是在送餐路途中与垃圾车相撞最终不幸身亡。

对于追求高收入的骑手来说，在行车安全上存在侥幸心理，对他们而言收入是最重要的，而与收入最相关的因素就是订单的准时率，其严重影响顾客对订单的满意度。其次才会在意行车安全，由于追求订单的准时率，外卖的完整情况相对而言没有很重要。

对于追求稳定的骑手来说，人身安全大过一切，其次注重准时率，最后注重外卖完整度。

首先客观地比较这三种指标的重要性，其判断矩阵为下表 5-3：

表 5-3 安全意识、准时率、外卖完整度三种指标的重要性

	安全意识	准时率	外卖完整度
安全意识	1	5	4
准时率	1/5	1	1/2
外卖完整度	1/4	2	1

安全意识的判断矩阵为下表：

表 5-4 安全意识的判断矩阵

安全意识	高收入	稳定	自主
高收入	1	1/5	1/3
稳定	5	1	2
适中	3	1/2	1

准时率的判断矩阵为下表：

表 5-5 准时率的判断矩阵

准时率	高收入	稳定	自主
高收入	1	4	3
稳定	1/4	1	2
适中	1/2	1/2	1

外卖完整度的判断矩阵为下表：

表 5-6 外卖完整度的判断矩阵

外卖完整度	高收入	稳定	自主
高收入	1	1/5	1/3
稳定	5	1	2
适中	3	1/2	1

3. 计算权重

计算权重的方法有算术平均法、几何平均法和特征值法三种。为了减少计算造成的误差，本章使用上述三种方法分别计算阶次权值，并使用数学软件 MATLAB 对模型进行求解和分析。判断矩阵一致性检验的代码见附件 1；三种方法计算权重的代码见附件 2；试验结果见下表，表 5-7；三种方法的权重计算结果见表 5-8 至 5-10；候选方案排序见表 5-11。

表 5-7 一致性检验结果

检验矩阵	结果	结论
三指标矩阵	0.0370 < 0.1	满足一致性
安全意识矩阵	0.0825 < 0.1	满足一致性
准时率	0.0088 < 0.1	满足一致性
外卖完整度	0.0311 < 0.1	满足一致性

表 5-8 算术平均法权重计算结果

	指标权重	高收入	稳定	自主
安全意识	0.6333	0.1038	0.6651	0.2311
准时率	0.2605	0.6999	0.1935	0.1066
外卖完整度	0.1062	0.0796	0.2468	0.6555

表 5-9 几何平均法权重计算结果

	指标权重	高收入	稳定	自主
安全意识	0.6370	0.1929	0.6585	0.2628
准时率	0.2583	0.6738	0.1061	0.2255
外卖完整度	0.1047	0.0786	0.1007	0.7010

表 5-10 特征值法权重计算结果

	指标权重	高收入	稳定	自主
安全意识	0.6370	0.1929	0.6585	0.2628
准时率	0.2583	0.6738	0.1061	0.2255
外卖完整度	0.1047	0.0786	0.1007	0.7010

表 5-11 候选方案排序

计算方法	候选方案		
	高收入	稳定	自主
算术平均法	0.3746	0.4539	0.1714
几何平均法	0.3721	0.4575	0.1704
特征值法	0.3721	0.4575	0.1704

5.1.3 结论

外卖行业的特点是多劳多得，外卖送餐人员的工资包括基本工资和派送订单的提成，公司里的全职送单员工有保底收入，每按时送达一单、且没有投诉和差评就可以得到 8 元钱的收入，几乎所有的外卖平台对外卖送餐人员都有类似的奖惩制度，比如，很多外卖平台或者餐饮店规定，如果外卖送餐人员送单得到了好评便可获得奖励，而如果送达超时的话，扣钱则是对外卖送餐人员一种普遍的约束手段。有的外卖平台规定了两种超时制度，普通超时和严重超时。普通超时的时限是30分钟以内，在此期间不会扣钱，严重超时是送餐时限超过了30分钟，这种情况下每单要扣3块钱。现实中很多外卖送餐人员由于闯红灯、骑车速度过快等诸多原因与货车、汽车、电动车或者行人相撞。

其实可以按工作风格和期望来制定奖惩措施，不能只因评价来奖励骑手，有的骑手追求效率，有的骑手追求服务，所以应当为了鼓励骑手风格多样化而设置奖惩措施。配送时间应由骑手自主选择，但是可以按区域内的平均配送时间设置一个配送时间的底限。追求稳定的只要在配送时间的底限之内送达就不会扣钱，追求高收入的则需要超前送达，但是要设置超前送达时间的上限维护骑手和交通安全。而骑手的奖励应当分为速达型和服务型，不能仅仅通过顾客的好评或差评来评定骑手的订单完成质量。顾客下单时也应当可以选择是速达型还是服务型，如果并不着急，可以选择服务型；如果着急用餐，则优先选择速达型。速达型的骑手需要完成的指标在于在相同的距离之内比平均骑手的送达速度提前在合理的时间，而服务型骑手需要完成的指标则是外卖的完整无损和服务态度的热情程度。当然这个所谓的合理依旧需要用大数据来衡量这一尺度。

5.2 问题 2 的模型建立与求解

外卖服务的一大价值核心就是时效性迅速及时性、外卖在途时间、外卖服务人员业务处理时间和外卖服务人员响应速度等方面的内容。所以时效性要素既有递送速度的要求，又有时间的要求。在外卖服务过程中很多的业务都是带有很大的随机性，只要客户有需求，就必须在客户要求的时间范围内到达，否则客户会选择其他外卖平台。

此外，在处理业务过程中仍然对时间有严格的要求，如果不能在最短的时间内处理完业务，就会影响下一个客户的业务开展。那么如何有效提高外卖服务送达速度，并能保证客户的时间要求。从目前的外卖行业所采用的交通运输工具类型和订单完成过程来看,对于大部分发展比较成熟的外卖骑手来说，能否获得高收入的根本是能否提供准确的时间保证。对于客户来讲，外卖服务质量的时效性体现就是能够按照客户的要求，在客户要求和预期的时间内有效完成取餐和送餐任务。只有满足了客户的这一要求，才能有效提高客户对服务质量的满意程度。

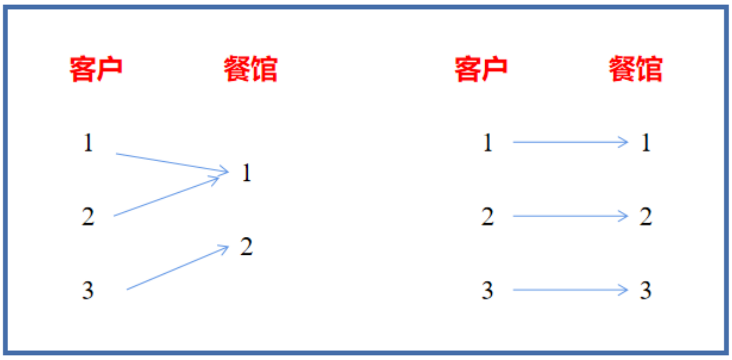
5.2.1 模型的建立

1. 假设条件

- 配送中心、客户与餐馆的位置及其之间的距离已知；
- 各客户对货物的需求里及对时间的限制已知，且为常数；
- 配送车辆的最大载货量、最远行驶距离、行驶速度固定和已知，且为常数
- 每个客户都能被车辆服务，且服务需求能够被满足；
- 单份外卖重量已知；
- 客户能接受的最晚送达时间、外卖超时单位惩罚费用已知，且为常数； 装载外卖时不考虑车辆空间、餐品体积等因素；

2. 设计虚拟配送场景：

外卖送中存在一对多的情形包括多名客户向一家餐馆订餐和一名客户同时向多家餐馆订餐，这大大增加了模型的复杂性。因此设立虚拟点对客户和餐馆的需求进行拆分，如图 5-4 所示原情形为客户 1、客户 2 同时向餐馆 1 下单。在传统路径问题中，配送员会多次访问餐馆 1 使模型复杂化。因此，引入餐馆 1 的虚拟点餐馆 3，设置客户 2 向虚拟餐馆 3 下单，形式 1 对 1 的简化模型[1]。



5-4 配送场景示意图

3. 构建超时惩罚目标函数：

目的：最小化时间惩罚成本外卖配送的时间窗采用单边软时间窗。这里设置的时间窗惩罚函数如下面公式所示为酬送员 k 到达客户的时间；1h 为客户 1 能接受的最晚送达时间，一般为系统设定的订单预定送达时间；P 为超时惩罚费用。如果订单送达，则不会受到惩罚；如果订单在 2h 内送达，则会产生与延误时间呈正相关的惩罚。因此最小化超时惩罚成本目标函数如下式所示，表示每个订单的惩罚成本:

$$P_i(T_{K_i}^A) = \begin{cases} 0, T_{K_i}^A < LT_i \\ (T_{K_i}^A - LT_i)P', LT_i \le T_{K_i}^A \end{cases}$$

5.2.2 模型的求解

根据偏离上述最小惩罚成本，做个线性奖惩，根据这个函数，生成一个送员评分表，将评分表动态评分展现给顾客，比如评分好的优先分配给难以忍受超时的顾客，反之亦然。

5.3 问题 3 的模型建立与求解

5.3.1 模糊控制模型的建立

由于考虑极端气候条件下的额外订单配送费用、订单配送时长及提成设计方案之间关系复杂。难以找到精确的数学函数对它们之间的关系进行分析计算，无

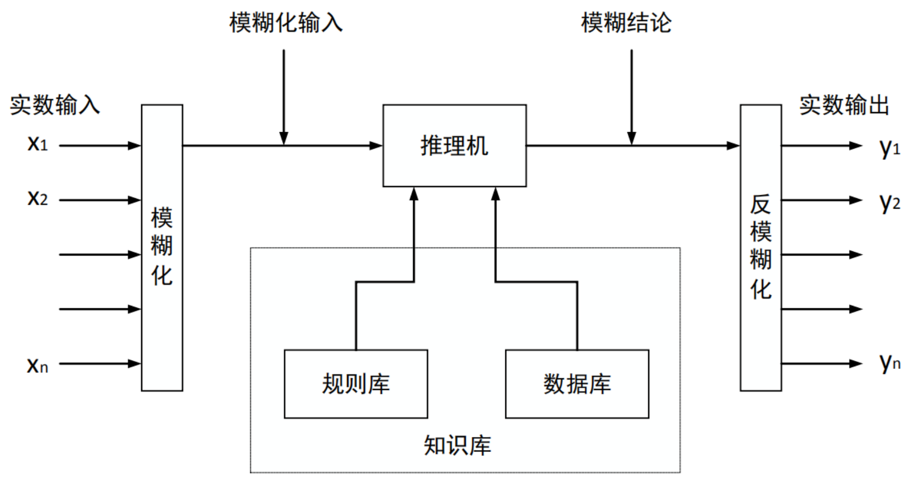
法直接找到一个通用的设计方案。模糊逻辑作为一种模糊数学方法，不需要精确的数学模型，可通过专家经验制定模糊控制规则对问题进行求解，且这种方法具有很强的鲁棒性、适应性及较好的容错性。因此该文拟采用模糊逻辑控制理论，运用模糊工具箱，选取隶属度函数，定义额外订单配送费用 x 以及订单配送时长 t 为输入量，订单提成 y 为输出量，对其进行参数优化研究[2]。

1. 模糊逻辑控制理论的产生和基本原理

模糊逻辑的概念最先由美国数学家 Zadeh 于 1965 年提出，模糊逻辑解决问题的整体思路是：①将已知的模糊规则归纳为前因（模糊输入集合）和结果（模糊输出集合）两论域（集合的描述范围）间的模糊蕴含关系；②将前因论域的现有知识与结果得到的模糊蕴含关系进行合成运算，推出结果。

(1) 模糊控制系统

模糊控制器是模糊控制的核心，一般为多输入多输出控制器，结构如图所示，主要包括输入量模糊化接口、推理机、输出量解模糊接口、数据库和规则库五个部分。



5-5 模糊控制器示意图

Step1：输入量模糊化：将输入量转换为控制器可识别的模糊矢量，该模糊矢量将作为模糊推理的输入为后续模糊推理做准备，模糊化过程由论域变换和模糊化两部分组成。

- 论域变换：将精确输入量经一定的尺度变换，得到与之一一映射变换量，通常论域范围是闭区间，区间外的值则为边界饱和值，最后在论域变化值的基础上进行一定比例的转换作为最终的变换输入。

- 模糊化：论域变换得到的变换输入也是精确值，经过模糊转换后变成模糊量，一般用模糊集合对其进行描述。通常在其论域内定义若干模糊子集和隶属度函数。只有经过模糊化的变量才能与知识库中的信息相对应，才能进行模糊推理，而模糊化通常指的值将变量用模糊子集或模糊语言描述的方式。

Step2：知识库：模糊控制的核心部分，主要包括数据库和规则库两部分。

- 数据库：存储隶属度矢量或隶属度函数。
- 规则库：储存所有模糊控制规则，是模糊控制的核心，其准确性取决于专家知识的准确度。

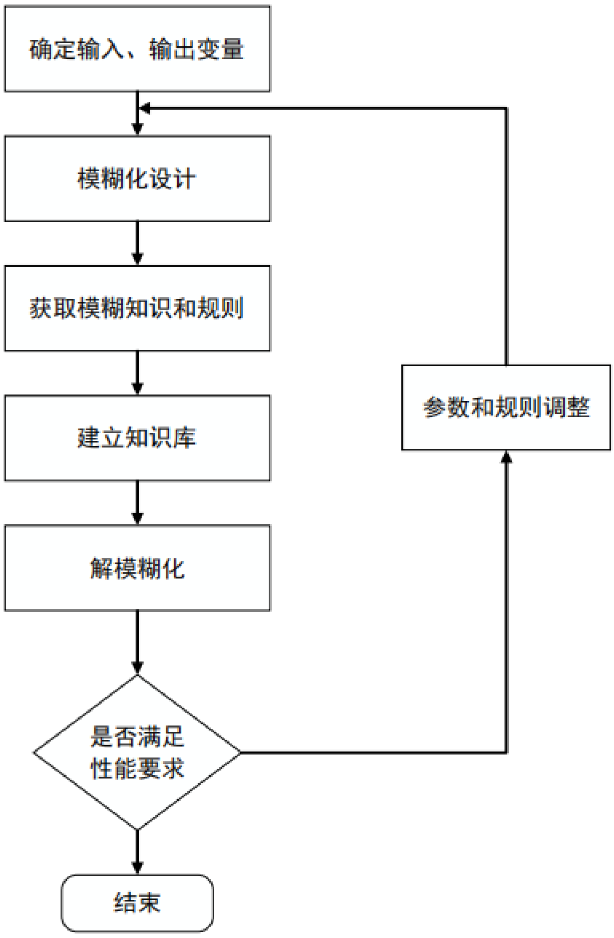
Step3：推理机：通过接受输入的模糊矢量，以规则库存储的模糊规则和数据库存储的隶属度函数实现对模糊矢量的模糊推理，最终求得模糊输出量的过程。以 2 输入 1 输出的模糊控制规则为例，有

R1：IF E is A_1 and F is B_1 THEN U is C_1
R2：IF E is A_2 and F is B_2 THEN U is C_2
.
.
.
Rn：IF E is A_n and F is B_n THEN U is C_n

其中语言变量是 E、F，$A_1 \dots A_n$ 是 E 的模糊集合，$B_1 \dots B_n$ 是 F 的模糊集合，所有规则是并列存在关系，是“或”的关系，因此整个规则库模糊关系可表示为 $R=\bigcup_{i=1}^{n} R_i$。若变量 E、F、U 的模糊论域分别为 A_i、B_i、C_i，则模糊推理结果表示为 $C_i=(A_i \times B_i) \circ R_i$，与模糊子集响应隶属度相对应。

Step4：反模糊化：将模糊输出量经过与模糊化过程相反的映射作用转换为精确输出量的过程，该精确值将作为执行机构的输入进行下一步计算。

- 解模糊：一个输入量可能会因多条模糊规则被激活而得到较多的输出值 C_i，构成一个输出值组合 $\bigcup C_i$ 该组合将通过一定的算法转换为论域内的变量，应用较广泛的方法有重心法、最大隶属度法。
- 论域反变换：与论域变换过程完全相反，论域反变换是将解模糊得到的变量通过一定比例转换为可被用于实际控制的精确值。模糊控制器功能大多可通过计算机实现，其设计流程图如图所示：



5-6 设计流程图

(2)模糊逻辑系统在 MATLAB 里的设计与仿真

MATLAB 是一个计算核心，针对不同的应用围绕着这个计算核也开发了许多不同的应用程式，称为工具箱。如 SIMULINK、CONTROL SYSTEMS、FUZZY LOGIC 等。FUZZY LOGIC 是 MATLAB 提供的模糊逻揖工具箱，该工具箱可以很方便捷的实现模糊控制器的设计和仿真。模糊推理是使用模糊逻辑将给定的输入到输出的映射公式化的过程，用户在 MATLAB 的模糊逻辑工具箱中可以很方便的建立模糊推理系统的各个模块，主要包括：FUZZY（基本模糊推理系统编辑器）、mfedit（隶属度函数编辑器）、ruleview（模糊推理规则观察器）、surfview（模糊推理输出特性曲面观察器）。fuzzy 用于处理系统一些高级属性：系统输入、输出变量的个数、名称等。mfedit 用于定义各种语言变量的隶属度函数的形状。Ruleview 和 surfview 主要用于用户对系统的观察，它们都是只读工具。模糊推理规则观察器是一个基于 MATLAB 的图形显示命令，通过它可以反映那个规

则的作用较为积极、隶属度函数的独立变化是如何影响系统的结果的。用户还可以利用模糊推理输出曲面观察器生成系统的输出特性曲面图，该图反映了模糊输入对模糊输出结果的影响[3]。

5.3.2 基于模糊控制模型设计方案的建立

本章采用模糊控制实现对考虑极端气候条件下的额外订单配送费、订单配送时长及提成设计控制，本节利用模糊逻辑控制理论，以订单提成为输出目标，额外订单配送费及订单配送时长为输入变量，得出一个通用的设计方案。

1. 模糊控制器结构

通常选择额外订单配送费用 x 以及订单配送时长 t 为模糊控制器的输入语言变量，控制器结构选择如图所示。

2. 输入输出变量选取

选择双输入单输出结构的控制器，根据实际应用场合确定模糊控制器的输入、输出变量。在对本问题实现控制时，根据其自身结构特点，额外订单配送费用 x 以及订单配送时长 t 为输入量，订单提成 y 为输出量。

3. 输入输出隶属度函数的确定

确定输入输出变量后，需要将输出输出变量根据变量大小转换为用“大”、“中”、“小”等语言变量等级衡量的模糊语言。将输入量 x 用语言变量{零，较小，小，较大，大}五个语言变量值表示;将输入量 t 用语言变量{零，较短，短，较长，长}五个语言变量值表示；订单提成 y 可用{零，较少，少，较多，多}五个语言变量值描述。

用五个语言变量分别描述输入、输出三个变量，模糊集合分别为：

$$x=\{\mathrm{NB,NS,ZE,PS,PB}\}$$

$$t=\{\mathrm{NB,NS,ZE,PS,PB}\}$$

$$y=\{\mathrm{NB,NS,ZE,PS,PB}\}$$

两输入变量量化论域为： $x=\{0,10,20,30,40\}$，$t=\{20,30,40,50,60\}$

输出变量量化论域为： $y=\{0,5,10,15,20\}$

模糊变量输入到模糊控制器后，控制器内部存储程序要自发执行模糊规则，要求将输入量用隶属度函数进行描述。通常为了计算方便，隶属度函数选择三角形分布或高斯分布，因为这类隶属度函数均呈正态分布。本文采用三角形函数作为输入、输出变量的隶属度函数。

4. 建立模糊控制规则表

设计模糊控制器的关键在于模糊控制规则的设计，确立准确的模糊规则能使模糊控制器取得非常良好的控制效果[4]。手动策略是集合大量学习、试验及经验演化形成的知识体系，控制规则是在此基础上发展形成的。模糊规则常用条件语句 if-then 进行描述，表达形式如下：

If x is_i A and_j t is BC

其中：$0\le \mathrm{A}_i \le 40$, $20\le \mathrm{B}_j \le 60$, $0\le \mathrm{C}_l \le 20$,共 25 条模糊规则。

设计的模糊控制规则表如表所示：

表 5-12 模糊控制规则表

y		t				
		NB	**NS**	**ZE**	**PS**	**PB**
x	NB	NB	NB	NB	NB	NB
	NS	NS	NS	NB	NB	NB
	ZE	ZE	ZE	ZE	NB	NB
	PS	PS	PS	PS	ZE	NB
	PB	PB	PB	PB	PS	ZE

5.3.3 结论

Mamdain 和 Sugeno 两种模糊推理方法应用最为广泛，本文采用 Mamdain min-max 模糊推理方法。 模糊推理后的结果不是确定值，一般是模糊集合或隶属度函数，需要进行反模糊化过程，将输出的模糊量转换为可被用于后续控制的确定值。反模糊化得到的结果会因为反模糊化方法的不同存在差异，本文采用重心法，去模糊化后的模糊推理结果如图所示。

5.4 问题 4 的模型建立与求解

5.4.1 博弈论模型的建立

博弈论（Game Theory）是研究决策主体的行为发生直接相互作用时的决策以及这种决策的均衡问题，因此也称“对策论”或“赛局理论”，这一理论最重要的特点是强调了经济主体之间的直接相互联系和影响。竞合理论是 Nalebuff 和 Brandenburger 基于对竞争本身固有的缺点的认识而提出的，该理论的主要观点是: 企业在市场经营过程中要摈弃恶性竞争战略，懂得寻求合作伙伴，在竞争基础上建立合作关系，而且这种关系是持续的动态的，从而实现多方共赢。博弈理论常用来研究具有竞争或者斗争性质现象的理论与方法，目前广泛应用于经济、政治、军事战略等领域中。

外卖服务的特殊之处在于，除了商家和用户，骑手在中间扮演了不可或缺的角色，而骑手的成本往往需要商家和用户承担。在当前外卖骑手送餐配送过程中，平台会过智能算法和数据分析不断压缩骑手的配送时间。平台的数据分析驱动下，外卖骑手正在向更快更廉价的趋势发展。平台团队把系统优化到极致，对骑手的体力和脑力也带来了极高的要求和挑战。外卖骑手不断抱怨的同时商家方面也对于平台高达 20 个点抽成叫苦不迭，其中部分商家为了盈利出现了调理包加热、食材不新鲜、包装材料廉价等问题，最终消费者也成为了受害群体。

1. 博弈论理论的产生和基本原理

早在 1881 年埃奇沃思在他的书，数学心理学已产生合作博弈的想法。埃奇沃斯的数学心理学通过联盟交换经济与有限的参与模型描述的一场比赛。每个参与者对一种商品与初始禀赋和预算约束。根据预算约束通过参与者之间的联盟配置资源。如果分配给参与者的贸易投资组合配置属于可行，集总所有各式各样的联盟的实力，这个配置是可行的。他提供合同曲线来描述续订合同，合同曲线是市场博弈的核心，当参与者数目逐渐增加，博弈市场将萎缩到只有竞争，结束配置。博弈论中，个人的效用不仅取决于自己的选择，还取决于其他人的选择，从而使多方的个人效用达到最佳的平衡。博弈论是研究局中人当下的决策对其他个体决策的影响，又反过来影响局中人自身的决定和平衡。

2. 博弈的特征函数与得益向量

特征函数的指给定一个有限的参与人集合 N，合作博弈的特征型是有限数对(N, v)，其中特征函数 v 是从 $2^N=\{S|S\subseteq N\}$ 到实数集 R^N 的映射 $v:2^N\to R^N$ 且 V(∅)=0 需要再次强调的是，合作博弈的基本假设是大联盟可以形成。大联盟不一定形成时，不仅要考虑收益的均衡配置，还有考虑均衡时的联盟或联盟分割。那就是联盟形成理论所要考虑的问题了。

特征函数是超可加的，其具体解释为，博弈中若联盟 S 和联盟 T 是两个互相独立的联盟，则他们联盟之后的得益最少要等于单独行动（即不联盟）时双方得益相加之和。用公式表示为，当 S，T 满足 $S\cap T=\{\varnothing\}$ 则有：$V(S\cup T)\geq V(S)+V(T)$ 直观上来看，如果一个博弈是超可加的，就意味着“整体大于部分之和”。也就是说，如果两个不相交的联盟能够实现某种剩余，那么这个联盟联合起来也至少可以实现这种剩余。超可加博弈是现实生活中很普遍的一类博弈。

3. 得益向量和博弈配置

具体解释为，博弈中若联盟 S 和联盟 T 是两个互相独立的联盟，则他们在 n 人合作博弈问题中，特征函数分析了各个局中人可能形成联盟的情况及每一种联盟情况下的得益值。那么很自然的，联盟形成后的联盟内部各成员之间利益（得益值）分配的问题就紧接着被提出了。无论从学术性和现实性的角度来看，n 人合作博弈研究中最重要的部分是各局中人经过协商并确定联盟的形式后，合作联盟所获收益（并非博弈模型本身得出的结果）的分配方案。显然，博弈系统中利益分配的公正性和合理性会直接影响联盟的形成以及维持，只有公正合理的利益分配才能保证局中人的加入和稳定。因为局中人都会理性的倾向于参加一个可以得到最多利益分配的联盟，并在由于某些关键局中人的参与将会使得联盟的收益大大增加，从而能为其他局中人带来额外的收益。所以如果局中人之间的收益分配方案不合理，就无法确定是否会形成稳定的联盟。得益向量和博弈配置结合起来共同描述某类合作博弈中各局中人的收益分配问题。

在特征函数为 V 的 n 人博弈中，如果得益向量 $x=\{x_1,x_2,\ldots,x_n\}$ 满足 $x_i\geq v(i)(\forall i\in N$ 且 $\sum_{i=1}^{n}x_i=v(N))$ 则称该向量为合作博弈的一个配置。合作博弈的配置应当符合两个条件：

(1) 有效性 $\sum_{i=1}^{n} x_i = v(N)$

(2) 个体理性式 $x_i \ge v(i)$

一般的，用 E(V)表示合作博弈联盟中的所有得益向量的集合。在上式中，x_i 是指分配给第个局中人的利益（得益值），v(i)是第个局中人单独行动时的特征函数值（视为博弈整体得益都属于这一个局中人，v(N)是指当全部局中人都参与到博弈中时，所有的局中人从博弈系统中所能得到的最大得益。有效性一式描述的是帕累托最优条件，它表示所有局中人共同作用下的总得益完全用于分配给每个局中人。个体有效性说明局中人都是理性的，在评价某个分配方案的时候，都会把留着该联盟中所能分配到的得益与离开联盟单干所能获得的好处相比较，如果后者更多，他就宁愿离开联盟。反正，则会留在该联盟中。

当合作博弈中出现不同的符合条件的得益向量时，会有不同得益向量之间的优劣问题，假设有得益向量 x_i 和 y_i，在都满足 $\sum_{i=1}^{n} x_i = v(N) = \sum_{i=1}^{n} y_i$ 的条件下，各局中人对得益向量优劣偏好不同一定会因为影响到得益而不同。

4. 博弈的解

博弈中，最难的地方就在于建立一个统一的解的概念。解得概念就是从各种各具良好特质的解之中选出最优的配置方案。利益分配中，常用的方法包括核心法、Shapley 值法、改进核心法、Nash 协商法、不对称 Nash 协商法、分配因子法等，其中应用最广泛的是 Shapley 方法。Shapley 值法是由 Shapley 提出，针对 n 人合作博弈提出的一种利益分配方法。所谓合作博弈，是指博弈局中，如果局中人能够达成如协议、承诺、威胁等具有完全的约束力且可强制执行约定，则该博弈关系称为合作博弈。合作博弈联盟内部成员之间可以互相交换信息。

Shapley 值方法的定义以及性质介绍如下：

假设有个 n 局中人或者 n 个企业从事某项经济活动。他们之间互相组成的每一种组合，通过组内合作都会得到一定的收益。当局中人之间的利益是非对抗性的时候，组中合作人数的增加会带来利益的增加，这样局中人全体共同参与的合作将带来最大的收益。n 个局中人合作后的利益分配就构成 n 人合作博弈。Shapley 值法为解决 n 人博弈中的每个参与人的利益分配提供了一种方法，它的主要原理是，根据联盟中每一个成员对联盟产生的边际贡献来分配联盟的总收益，

以确保分配的公正性。设集合 $N=\{1,2,K\ ,n\}$ 如果对于 N 的任意子集 S，都对应着一个实值函数 $V(S)$，满足 $V(\phi)=0;V(S_1 \mathrm{U} S_2)\geq v(S_1)+V(S_2)$（其中 $S_1 \mathrm{U} S_2=\phi,\phi$ 为空集 x 此条性质称为超可加性），此时称 $[N,V]$ 为 n 人合作对策，V 为博弈的特征函数。在上面所述经济活动中定义为 n 人集合，S 为 n 人集合的任一个子集，$V(S)$ 为合作联盟的收益。用 X_i 表示 N 中的成员 i 从合作联盟最大收益中应得到的一份收入。

(1) $V(S)>\sum_{i\in s}V(i)\ 2$ 意即合作联盟所得的整体利益要大于各企业独立决策时所得的利益之和。

(2) $X_i\geq v(i), i=1,2,K\ n$，其中 $V(i)$ 是成员 i 不与任何其他局中人结盟时的收益。意即个体合理性，表示企业□在合作联盟中所分得的收益，不小于其参加合作联盟之前本身独立决策时所能获得的收益。

$\sum_{i\in s}X_i\geq v(S)$，即子集中的所有成员在最大收益 $V(N)$ 中所获得收益之和都不小于在子集 S 中获得的收益之和。

5.4.2 基于模糊控制的博弈论模型设计方案的建立

基于第三问所建立的模糊控制模型，我们将博弈论理论运用其中，建立订单配送费用 x 订单配送时长 t，及用户幸福指数 y 相关的模糊博弈论模型。本节利用模糊逻辑控制理论，以订单提成为输出目标，订单配送费用及订单配送时长为输入变量，得出一个外卖平台、骑手、商家与消费者之间实现共享与共赢的目标设计方案。

1. 模糊控制器结构

通常选择订单配送费用 x 以及订单配送时长 t 为模糊控制器的输入语言变量，控制器结构选择如图所示。

2. 输入输出变量选取

选择双输入单输出结构的控制器，根据实际应用场合确定模糊控制器的输入、输出变量。在对本问题实现控制时，根据其自身结构特点，订单配送费用 x 以及订单配送时长 t 为输入量，用户幸福指数 y 为输出量。

3. 输入输出隶属度函数的确定

确定输入输出变量后，需要将输出输出变量根据变量大小转换为用“大”、“中”、“小”等语言变量等级衡量的模糊语言。将输入量 x 用语言变量{少，较少，中，较多，多}五个语言变量值表示；将输入量 t 用语言变量{零，较短，短，较长，长}五个语言变量值表示；用户幸福指数 y 可用{差，较差，中，较好，好}五个语言变量值描述。

用五个语言变量分别描述输入、输出三个变量，模糊集合分别为：

$$x=\{\mathrm{NB},\mathrm{NS},\mathrm{ZE},\mathrm{PS},\mathrm{PB}\}$$

$$t=\{\mathrm{NB},\mathrm{NS},\mathrm{ZE},\mathrm{PS},\mathrm{PB}\}$$

$$y=\{\mathrm{NB},\mathrm{NS},\mathrm{ZE},\mathrm{PS},\mathrm{PB}\}$$

两输入变量量化论域为： $x=\{0,5,10,15,20\}$， $t=\{20,30,40,50,60\}$

输出变量量化论域为： $y=\{2,4,6,8,10\}$

模糊变量输入到模糊控制器后，控制器内部存储程序要自发执行模糊规则，要求将输入量用隶属度函数进行描述。通常为了计算方便，隶属度函数选择三角形分布或高斯分布，因为这类隶属度函数均呈正态分布。本文采用三角形函数作为输入、输出变量的隶属度函数。

4. 建立模糊控制规则表

设计模糊控制器的关键在于模糊控制规则的设计，确立准确的模糊规则能使模糊控制器取得非常良好的控制效果。手动策略是集合大量学习、试验及经验演化形成的知识体系，控制规则是在此基础上发展形成的。模糊规则常用条件语句 if-then 进行描述，表达形式如下：

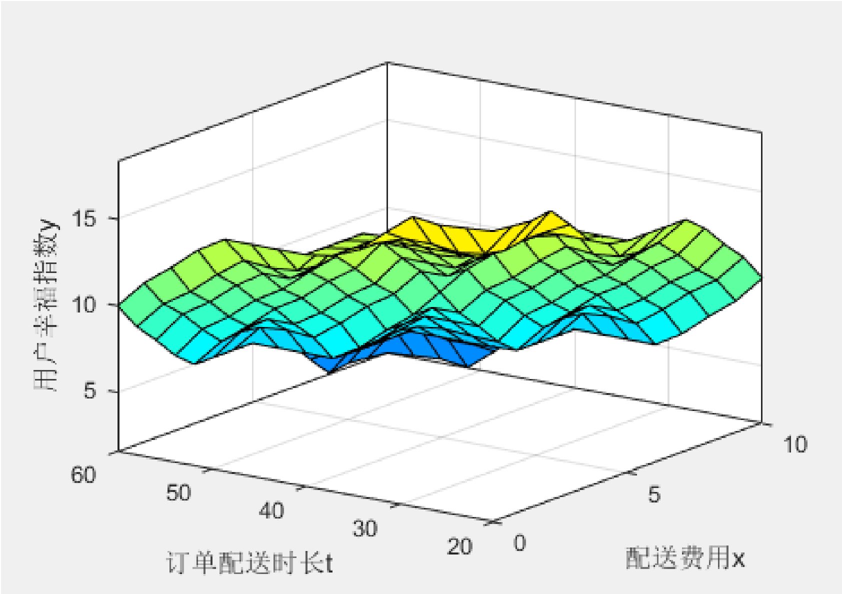
$$\mathrm{If}\ x\ \mathrm{is}\ A_i\ \mathrm{and}\ t\ \mathrm{is}\ B_j\ C$$

其中： $0\le \mathrm{A}_i\le 20$， $20\le \mathrm{B}_j\le 60$， $0\le \mathrm{C}_l\le 10$，共 25 条模糊规则。

5.4.3 结论

Mamdain 和 Sugeno 两种模糊推理方法应用最为广泛，本文采用 Mamdain min-max 模糊推理方法。 模糊推理后的结果不是确定值，一般是模糊集合或隶属度函数，需要进行反模糊化过程，将输出的模糊量转换为可被用于后续控制的

确定值。反模糊化得到的结果会因为反模糊化方法的不同存在差异，本文采用重心法，去模糊化后的模糊推理结果如图所示。



5-7 模糊推理结果图

5.5 问题 5 的模型建立与求解

5.5.1 长距离跨区集中配送优化模型的建立

配送方式一般包含五种形式：平台专送、众包、快送、混合送、商家自配送。平台专送指平台自建或加盟人员承接商家所有订单，保障一定的接单率。是平台自营全职骑手为主的配送形式，营业时间早上九点至晚上九点，配送范围 3 公里为主。其配送人员正规化，给用户优质的形象，配送速度最快。但成本稍高，配送时间和区域有限制。收费主要包括商家提成＋顾客配送费；平台快送指的是众包人员承接商家所有订单，保障一定的接单率，兼职配送人员为主的配送形式，它的营业时间 24 小时，配送范围最大 5 公里左右。其优势在于相对专送配送范围相对大且成本要低于专送，特别是针对用户端收费低于专送 1-2 元。但由于配送范围扩大，配送速度要慢于专送。收费包括商家提成＋顾客配送费；平台众包指利用社会闲散力量帮助商家进行配送的形式，营业时间 24 小时，配送范围最大 5 公里左右。商家可自行选择是否发众包，非全托管。其相对专送配送范围相对大，成本较低。但配送速度和配送人员整体服务要慢于专送、快送。收费模式

为按照距离收费；混合送---专送+快送：专送运力不足时可以由快送人员帮忙送餐；商家自配送是商家依靠自有运力或者第三方众包运力进行产品配送的形式。优势在于自配送对外卖的区域限制小，时间限制小。自配送的成本，跟店铺单量和单均价有很大关系，如果单量起伏不定，而且客单价偏低，分布比较散，那么自有配送成本就会变得很高，相反则成本低。但单量低且分散，导致成本高，雨雪等恶性天气很难保证运力，整体配送速度和服务低[5]。

本文提出一种更长距离和跨区域的外卖订单的集中配送模式算法，用于解决更长距离和跨区域外卖配送集中地区配送过程中配送时间长、运力浪费的问题。相应订单配送提成定价策略可以主要根据配送距离按照距离收费。首先集中送餐路线模型假设，假设在外卖配送过程中满足的基本条件；其次建立最短路线模型建模，建立方程和约束条件确定最短路径规划模型；之后用禁忌搜索算法求解，在确定模型及假设成立的条件下求解外卖配送过程中最短路径，并用 MATLAB 软件进行计算机仿真。实现长距离和跨区域的外卖订单将具有近似收货地点的外卖商品统一配送，对取送外卖商品路径进行合理规划，节约运力的同时减少配送总时长。

1. 长距离跨区集中配送配送优化算法内容

外卖长距离跨区集中配送模式下路线优化算法包括:集中送餐路线模型假设,假设在外卖配送过程中满足的基本条件;最短路线模型建模,建立方程和约束条件确定最短路径规划模型;禁忌搜索算法求解,在确定模型及假设成立的条件下求解外卖配送过程中最短路径,并用 MATLAB 软件进行计算机仿真。

2. 所述送餐路线模型假设包括:

（1）每个餐饮企业接收外卖订单后都有外卖商品提供给配送员。且外卖商品总体积应小于外卖配送箱最大容积,外卖配送员所使用的外卖配送箱为统一规格,最大体积为 43 升；

（2）每辆外卖配送车从固定的餐饮企业出发,最后又回到原点,以方便下次配送；

（3）同一辆外卖配送车只能经过同一个餐饮企业一次,若该餐饮企业有多份订单由同一配送员配送则一次交付给配送员；

（4）“集中送餐”配送模式所追求的配送时间小于轻模式和重模式中的抢单配送送餐时间,由于这两种配送模式不对配送路线进行规划,因此假设无规划的配送平均时间为 n 分钟,故所求配送时间应小于 n 分钟,才能保证外卖的完好程度；

（5）集中送餐模式适用于餐饮企业位置较集中地区,假设各餐饮企业之间的距离小于 5000 米；

（6）订单所包含的餐饮企业具有最大值 m；

3. 最短路线模型建模包括：

G------餐饮企业所在点的集合

D------外卖配送总时间

d_{ij} ------餐饮企业 i 至餐饮企业 j 的最短距离

V_j------外卖配送员在餐饮企业 j 取走的外卖体积

$V_{总}$------外卖配送车的最大体积

S_k------餐饮企业 i 至送餐点 k 的距离

t_0------外卖配送员在餐饮企业取餐的时间

x_{ij}-------0-1 变重,具中: $x_{ij}=\begin{cases}1\text{配送车选择i-j这条线路}\\0\text{配送车不选择该路线}\end{cases}$

整个规划方案可以由下列一组方程和约束条件确定：

$$\min D=\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}d_{ij}x_{ij}$$

$$\sum_{j=1}^{n}X_{ij}=\sum_{j=1}^{n}X_{ji}\leq 1$$

$$\sum_{i=1}^{n}X_{ij}=1$$

$$\sum_{j=1}^{n}X_{ij}=1$$

$$\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\left(\frac{d_{ij}x_{ij}}{v}\right)+n*t_0+\frac{S_K}{v}\leq n\min$$

$$0\leq d_{ij}\leq 5$$

$$\sum_{j=1}^{n}V_i\leq 43$$

$$\mathrm{D},\mathrm{V_j},\mathrm{V_{总}},\mathrm{S_k},\mathrm{t_0}\geq 0$$

$$0\leq \mathrm{i}\leq \mathrm{n}-1, 0\leq \mathrm{j}\leq \mathrm{m}$$

式 1 为目标函数,求最短距离；式 2 确定了出发点,即必须从出发点出发，然后再回到出发点；式 3、式 4 确定了外卖配送车只能经过某餐饮企业一次，不能重复经过该餐饮企业；式 5 确定了原有配送模式中配送平均时间为 n 分钟，故所求配送时间应小于 n 分钟，才能保证外卖的完好程度；式 6 确定了从餐饮企业 i 至餐饮企业 j 的最大距离；式 7 限制了派给外卖配送员的订单中外卖的总体积；式 8 确定了最短距离、外卖配送员在餐饮企业 j 取走的外卖体积、外卖配送车的最大体积、外卖企业 i 至送餐点 k 的距离和外卖配送员在餐饮企业取餐的时间不小于 0，应为正值；式 9 确定了餐饮企业 i 和 j 的最大取值。

本算法能够实现长距离跨区相近收货地址的外卖商品用较短时间共同配送，用户在外卖平台下单后，外卖平台会根据用户收货地址及餐饮企业的位置为配送员规划出最优取送外卖商品路径，从而实现统一收货地址的外卖商品一次性送达，可以保证外卖商品的新鲜程度，同时减少配送时间，节省人力运力。

5.5.2 长距离跨区集中配送优化算法的实现

通过实例,并结合附图,对长距离跨区集中配送优化算法方案进一步具体说明。

- 用户下单步骤;多名用户在外卖平台选定不同餐饮企业的外卖商品进行下单,标明配送地址形成订单信息。
- 企业接单步骤:用户下单后,餐饮企业决定是否接单,不接单则订单取消;接单则打印订单信息,准备制作外卖商品。
- 平台处理步骤:餐饮企业接单后,外卖平台进行信息整合处理,通过计算方法规划出配送员进行取餐、送餐最短路径。

外卖平台根据已收集信息派单给配送员并给配送员提供最短路径同时将外卖配送员的信息反馈给各家餐饮企业，餐饮企业开始制作外卖商品。外卖配送员根据外卖平台提供的路径前往各家餐饮企业取餐。外卖配送员将外卖商品收集完毕后前往收货点送餐。外卖配送员通知用户取餐,用户收到外卖商品,配送完毕。

举例如下:

- 某大学学生袁某及其室友在某网络订餐平台下单，订购外卖如下：黄焖鸡 2 份；盖浇饭 2 份；烤肉饭 2 份；牛肉饭 1 份；汉堡 1 份。配送地址为该大学北区第八学生公寓楼。

- 五家餐饮企业分别接单，打印订单信息。
- 五家餐饮企业位置分布如下表所示：

5-13 五家餐饮企业位置分布表

餐饮企业名称	横纵坐标	标记点
旺客基黄焖鸡	(0.6,5.8)	C1
重庆鸡公煲	(4.3,0.7)	C3
小杨烧菜馆	(6.3,3.7)	C2
华莱士	(8.3,2.1)	C4
稻香烤肉饭	(12.6,2.9)	C5

针对伴随外卖配送新模式产生的数学模型,本发明利用禁忌搜索算法进行求解并通过 MATLAB 软件进行计算机仿真,得出餐饮企业最优循环路径如图所示,其最优巡回餐饮企业序列如下表所示。

5-14 最优巡回餐饮企业序列表

起点	最优巡回餐饮企业序列					
C1	C1	C2	C5	C4	C3	C1
C2	C2	C1	C3	C4	C5	C2
C3	C3	C4	C5	C2	C1	C3
C4	C4	C5	C2	C1	C3	C4
C5	C5	C2	C1	C3	C4	C5

在此实施例中,若选择 Cl 作为外卖配送员出发的起点,那么该配送员行走的路线为 C1→C2→C5→C4→C3→Cl。然而,由于该配送员在 C3 点进行外卖收取后,外卖的种类及数量已经达到了订单的要求,故该配送员可以直接从 C3 点出发,前往北八公寓进行配送。那么,该配送员的真实行走路线为 C1→C2→C5→C4→C3→北八。经测量,C1、C2、C3、C4 和 C5 节点的距离关系如下表所示。

5-15 距离关系表

两节点间距离 (m)	C1	C2	C3	C4	C5
C1	0	730	430	720	710
C2	730	0	1180	530	400
C3	430	1180	0	320	1140
C4	720	530	320	0	570
C5	710	400	1140	570	0

由上表可知 C1 至 C2 距离为 730 米,C2 至 C5 距离为 400 米,C5 至 C4 距离为 570 米,C4 至 C3 距离为 320 米.除此之外,查询地图可知,从 C3 点至北八宿舍的距离为 1700 米。配送员预计行驶总里程 S 为

$$S=\sum 各段行驶里程=730+400+570+320+1700=3720M$$

配送员驾车行驶的总时间 T1

$$T1=S/v=3720/10=0.372h=22.3\min$$

配送员进行外卖收取的总时间 T2 为

$$T2=\sum 在各外卖店收取外卖的时间 2\times 5=10\min$$

配送员进行外卖配送的总时间 T 为

$$T=T1+T2=22.3+10=32.3\min$$

5.5.3 结论

实施例中所要求的订单若采用基于长距离跨区共同配送的集中送餐模式,则学生从下单到通知学生收取外卖共需要 32.3 分钟;若采取原配送模式,则学生会分 5 批收到外卖,平均时间为 T3:

$$T3=\sum 外卖餐饮企业配送时间/餐饮企业个数=(46+46+46+38+38)/5=42.8\min$$

综上可知,长距离跨区集中送餐模式将配送时间缩短了 25%,提高了配送效率。也验证了长距离跨区集中送餐模式在外卖配送集中地区实施的可行性。其订单配送提成定价根据配送距离按照距离收费,助于调动外卖人员配送积极性。

六、模型的评价及优化

6.1 模型的优点

(1) 层次分析法是系统性的分析方法,简洁实用,所需定量数据信息较少;

(2) 模糊控制是一种基于规则的控制,它直接采用语言型控制规则,出发点是现场操作人员的控制经验或相关专家的知识,在设计中不需要建立被控对象的精确的数学模型,因而使得控制机理和策略易于接受与理解,设计简单,便于应

用。模糊控制是基于启发性的知识及语言决策规则设计的，这有利于模拟人工控制的过程和方法，增强控制系统的适应能力，使之具有一定的智能水平。模糊控制系统的鲁棒性强，干扰和参数变化对控制效果的影响被大大减弱，尤其适合于非线性、时变及纯滞后系统的控制

6.2 模型的缺点

(1) 层次分析法主观性过强，有一定的局限性；

(2) 模糊算法数据存在不确定性，有一定的偏差；

(3) 基于模型的控制算法及系统设计方法，由于出发点和性能指标的不同，容易导致较大差异

6.3 模型的推广

由于外卖骑手的数据来源稀少，并没有大量的数据可以用来评判，所以所采用的模型多带有一些合理的主观判断和构想，如果能在大数据的支撑下，可以更确定地采用因子分析和多元线性回归模型等来建立确切的函数。

参考文献

[1] 杜芊颖. 基于启发式算法的物流配送路径设计与实现[D]. 南京邮电大学, 2020.
[2] 魏铭琦. 考虑价格竞争的可持续闭环供应链网络规划[D]. 沈阳大学, 2021.
[3] 桑雅静. 生鲜产品冷链物流配送中心选址及配送问题研究[D]. 沈阳大学, 2021.
[4] 施文, 张伟, 王亚刚. 基于免疫算法的 p 模糊控制 agv 路径偏正[J]. 控制工程, 2021, 28(05): 870-876.
[5] 刘翰培, 王东署, 汪宇轩, et al. 移动机器人路径规划的模糊人工势场法研究[J]. 控制工程, 1-6.

附录

附件 1

```
Disp('Please enter the judgment matrix A')
A=input('A=');
[n,n]=size(A);
%%Calculate CR
clc
[n,n]=size(A)
[V,D]=eig(A)
Max_eig=max(max(D))
CI=(Max_eig-n)/(n-1);
RI=[0 0 0.52 0.89 1.12 1.26 1.36 1.41 1.46 1.49 1.52 1.54 1.56 1.58 1.59];
CR=CI/RI(n);
disp('CI=');disp(CI);
disp('CR=');disp(CR);
if CR<0.10
    disp('due to CR<0.10，Acceptable!');
else
    disp(' due to CR>=0.10，It needs to be modified!');
end
```

Appendix 2: Weight calculation

```
disp('Please enter the judgment matrix A')
A=input('A=');
[n,n]=size(A);
%%Method 1: Arithmetic mean method
Sum_A=sum(A);
SUM_A=repmat(Sum_A,n,1);
Stand_A=A./SUM_A;
disp('Obtained by the arithmetic mean method is as follow:');
disp(sum(Stand_A,2)./n)
%%Method 2: Geometric averaging method
Prduct_A=prod(A,2);
Prduct_n_A=Prduct_A.^(1/n);
disp(' Obtained by the geometric average method is as follow:');
disp(Prduct_n_A./sum(Prduct_n_A))
%%Method 3: Eigenvalue method
[V,D]=eig(A);
Max_eig=max(max(D));
[r,c]=find(D==Max_eig,1);
disp('Obtained by the eigenvalue method method is as follow');
disp(V(:,c)./sum(V(:,c)))
```

附件 2

```
[System]
Name='Untitled'
Type='mamdani'
Version=2.0
NumInputs=2
NumOutputs=1
NumRules=25
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

[Input1]
Name=' x'
Range=[0 40]
NumMFs=5
MF1='NB':'trimf',[0 0 10]
MF2='NS':'trimf',[0 10 20]
MF3='PB':'trimf',[30 40 40]
MF4='ZE':'trimf',[10 20 30]
MF5='PS':'trimf',[20 30 40]

[Input2]
Name=' t'
Range=[20 60]
NumMFs=5
MF1='NB':'trimf',[20 20 30]
MF2='NS':'trimf',[20 30 40]
MF3='PB':'trimf',[50 60 60]
MF4='ZE':'trimf',[30 40 50]
MF5='PS':'trimf',[40 50 60]

[Output1]
Name=' y'
Range=[0 20]
NumMFs=5
MF1='NB':'trimf',[0 0 5]
MF2='NS':'trimf',[0 5 10]
MF3='PB':'trimf',[15 20 20]
MF4='ZE':'trimf',[5 10 15]
MF5='PS':'trimf',[10 15 20]
```

[Rules]
1 1, 2 (1) : 1
2 1, 2 (1) : 1
4 1, 4 (1) : 1
5 1, 5 (1) : 1
3 1, 3 (1) : 1
1 2, 2 (1) : 1
2 2, 2 (1) : 1
4 2, 4 (1) : 1
5 2, 5 (1) : 1
3 2, 3 (1) : 1
1 4, 1 (1) : 1
2 4, 2 (1) : 1
4 4, 4 (1) : 1
5 4, 5 (1) : 1
3 4, 3 (1) : 1
1 5, 1 (1) : 1
2 5, 1 (1) : 1
4 5, 2 (1) : 1
5 5, 4 (1) : 1
3 5, 5 (1) : 1
1 3, 1 (1) : 1
2 3, 1 (1) : 1
4 3, 1 (1) : 1
5 3, 2 (1) : 1
3 3, 4 (1) : 1

附件 3

```
#include<iostream>
#include<cstdlib>
#include<cstdio>
#include<algorithm>
#include<vector>
#include<queue>
using namespace std;
struct edge{
	int to,cap;
	friend bool operator<(const edge a,const edge b){
		return a.cap<b.cap;
	}
};
edge p[20001];
vector <edge> E[20001];
int	dist[20001];
bool mex[20001],used[20001];
int sg[20001];
int n,m,k;
void addedge(int from,int to,int c){
	E[from].push_back((edge){to,c});
}
void spfa(){
	memset(dist,-1,sizeof(dist));
	memset(used,0,sizeof(used));
	queue <int> q;
	q.push(1);
	dist[1]=0;used[1]=true;
	while(q.size()>0){
			int now=q.front();q.pop();
			for(int i=0;i<E[now].size();i++){
				edge tmp=E[now][i];
				if(dist[tmp.to]==-1||dist[tmp.to]>dist[now]+tmp.cap)
				{
					dist[tmp.to]=dist[now]+tmp.cap;
					if(!used[tmp.to]){
						q.push(tmp.to);
						used[tmp.to]=true;
					}
				}
			}
			used[now]=false;
```

```
		}
}
int main(){
	//freopen("b.in","r",stdin);freopen("b.out","w",stdout);
	int x,y,c;
	while(scanf("%d%d%d",&n,&m,&k)!=EOF)
	{
		for(int i=1;i<=n;i++)E[i].clear();
		for(int i=0;i<m;i++){
			scanf("%d%d%d",&x,&y,&c);
			addedge(x,y,c);
			addedge(y,x,c);
		}
		spfa();
		for(int i=1;i<=n;i++)
			p[i]=((edge){i,dist[i]});
		sort(p+1,p+n+1);
		int t=n;
		while(t>0){
			bool f=false;
			memset(mex,0,sizeof(mex));
			for(int i=0;i<E[p[t].to].size();i++){
				int tmp=E[p[t].to][i].to;
				if(dist[tmp]>p[t].cap){mex[sg[tmp]]=true;f=true;}
			}
			for(int i=0;1;i++){
				if(mex[i]==false){sg[p[t].to]=i;break;}
			}
			if(f==false) sg[p[t].to]=1;
			t--;
		}
		int ans=0;
		for(int i=0;i<k;i++){
			scanf("%d",&x);
			ans=ans^sg[x];
		}
		printf(ans?"Alice\n":"Bob\n");
	}

	//fclose(stdin);fclose(stdout);
	return 0;
}
```

```
[System]
Name='Untitled'
```

```
Type='mamdani'
Version=2.0
NumInputs=2
NumOutputs=1
NumRules=25
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

[Input1]
Name='订单配送费用 x'
Range=[0 10]
NumMFs=5
MF1='NB':'trimf',[0 0 2.5]
MF2='NS':'trimf',[0 2.5 5]
MF3='PB':'trimf',[7.5 10 10]
MF4='ZE':'trimf',[2.5 5 7.5]
MF5='PS':'trimf',[5 7.5 10]

[Input2]
Name='订单配送时长 t'
Range=[20 60]
NumMFs=5
MF1='NB':'trimf',[20 20 30]
MF2='NS':'trimf',[20 30 40]
MF3='PB':'trimf',[50 60 60]
MF4='ZE':'trimf',[30 40 50]
MF5='PS':'trimf',[40 50 60]

[Output1]
Name='用户幸福指数 y'
Range=[0 20]
NumMFs=5
MF1='NB':'trimf',[0 0 5]
MF2='NS':'trimf',[0 5 10]
MF3='PB':'trimf',[15 20 20]
MF4='ZE':'trimf',[5 10 15]
MF5='PS':'trimf',[10 15 20]

[Rules]
1 1, 3 (1) : 1
2 1, 3 (1) : 1
```

1 2, 3 (1) : 1
4 1, 5 (1) : 1
2 2, 5 (1) : 1
1 4, 5 (1) : 1
5 1, 5 (1) : 1
4 2, 5 (1) : 1
2 4, 5 (1) : 1
1 5, 5 (1) : 1
3 1, 4 (1) : 1
5 2, 4 (1) : 1
4 4, 4 (1) : 1
2 5, 4 (1) : 1
1 3, 4 (1) : 1
3 2, 2 (1) : 1
5 4, 2 (1) : 1
4 5, 2 (1) : 1
2 3, 2 (1) : 1
3 4, 2 (1) : 1
5 5, 2 (1) : 1
4 3, 2 (1) : 1

附件 4

```
void localsearchWNS(const vector< vector<int> > &v,vector<int> &x,int &costbest,int firstcity)
{
    swap(x[0],x[firstcity]);
    costbest=countDis(x,v);//countDis 用于计算花费，代码在此不再贴出
    vector<int> tempx=x;
    int num=x.size();
    while(true)
    {
        int min=costbest;
        for(int i=1;i<num-1;i++)
        {
            for(int j=i+1;j<num;j++)
            {
                swap(x[i],x[j]);
                int temp=countDis(x,v);
                if(temp<min)
                {
                    min=temp;
                    tempx=x;
                }
                swap(x[i],x[j]);
            }
        }
        if(min==costbest)
        {
            break;
        }
        costbest=min;
        x=tempx;
    }
}
```