分　类　号：TP39　　单位代码：10183
研究生学号：2017532090　　密　　级：公开

吉 林 大 学

硕士学位论文

(学术学位)

基于蚁群算法的需求可拆分车辆路径问题研究

Research for Split Delivery Vehicle Routing Problems based on Ant Colony Algorithms

作者姓名：杨文哲
专　　业：计算机应用技术
研究方向：机器学习
指导教师：周柚
培养单位：计算机科学与技术学院

2020 年 5 月

基于蚁群算法的需求可拆分车辆路径问题研究

Research for Split Delivery Vehicle Routing Problems based on Ant Colony Algorithms

作者姓名：杨文哲

专业名称：计算机应用技术

指导教师：周柚

学位类别：工学硕士

答辩日期：2020 年 5 月 30 日

吉林大学硕士学位论文原创性声明

本人郑重声明：所呈交学位论文，是本人在指导教师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：杨文哲

日期：　　2020 年 5 月 31 日

摘 要

基于蚁群算法的需求可拆分车辆路径问题研究

当今世界经济的迅猛发展大大促进了物流运输的发展，物流产业作为各国经济的重要组成部分，已经广泛的引起了人们的重视。目前物流的发展程度决定了一个国家的发展水平，它可以提高一个国家的国民生产总值，引导国家经济的发展方向。减少运输成本可以有效提高物流行业的效益，最常用的方法就是使运输路径最短。

传统的车辆路径问题（Vehicle Routing Problem，VRP）就是研究运输路径最小化的问题。因为它要求每个顾客的需求必须由一辆运输车进行配送，所以当一些顾客的需求量大于车辆容量时，就会出现无法满足所有顾客需求的情况。为了解决这个问题，相关学者开始考虑对顾客的需求进行拆分，并提出了需求可拆分车辆路径问题（Split Delivery Vehicle Routing Problem，SDVRP）。本文在研究 SDVRP 问题的过程中发现它没有考虑到在运输途中消耗物资的情况，因此本文将路径消耗加入到 SDVRP 问题的约束条件中，提出了运输途中货物消耗的需求可拆分车辆路径问题（Split Delivery Vehicle Routing Problem with Goods Consumed during Transit，SDVRP-GCT）或者称为带有路径消耗的需求可拆分车辆路径问题。SDVRP-GCT 问题不仅可以适用于物流配送问题，同时也可以应用于生活中其他问题的解决。

本文是在 SDVRP 问题的基础上，分析了传统的 SDVRP 问题的研究现状与短板，并将 SDVRP 问题与真实世界中的物流场景相结合，提出一种更符合现实物流运输情形的 SDVRP-GCT 问题，随后基于蚁群算法设计了三种扩展算法进行问题求解。本文进行实验所用的数据集包括：SDVRPLIB 数据集、VRP web 中的大规模数据集和数学建模网站中收集的旅游规划数据集。首先本文将三种扩展算法应用于 SDVRP 问题中，并将得到的三种算法的结果与数据集提供的最优解进行对比，结果表明三种扩展算法能够改善目前存在的最优解，这说明在解决 SDVRP 问题上，本文提出的三种扩展算法表现出了很好的性能。随后本文假设车辆在运输途中的商品消耗与路径长度成比例，将 SDVRP 基准数据集转化为 SDVRP-GCT 数据集，然后将三种扩展的蚁群算法应用在转化后的 SDVRP-GCT

数据集上，得到的实验结果表明在不同类型的数据集上，三种扩展算法均有较好的表现。其次，本文将三种扩展的蚁群算法应用在 VRP web 中的四个大规模数据集上，实验结果表明本文扩展的算法在数据规模很大时仍然能有很好的解决问题。最后，本文将扩展的蚁群算法应用在自收集的旅游路线规划问题上，证明了 SDVRP-GCT 问题在现实生活场景中有更广泛的应用。

综上所述，本文的主要研究工作包括：研究 SDVRP 问题的背景和模型，提出 SDVRP-GCT 问题并对其进行数学建模；扩展三种蚁群算法，可以同时求解 SDVRP 问题和 SDVRP-GCT 问题；设计实验，通过实验证明本文扩展的三种蚁群算法可以有效的求解 SDVRP 问题和 SDVRP-GCT 问题；最后将生活中的旅游规划问题建模成 SDVRP-GCT 问题，并用本文提出的三种扩展算法进行求解，从而证明本文工作在现实生活中有广泛的应用前景。

关键字：

车辆路径问题，需求可拆分车辆路径问题，蚁群算法，运输途中的货物消耗，数学建模

Abstract

Research for Split Delivery Vehicle Routing Problems based on Ant Colony Algorithms

Nowadays, the rapid development of the world economy has greatly promoted the development of logistics and transportation. As an important part of the economy of various countries, the logistics industry has attracted people's attention. At present, the development level of logistics determines the development level of a country. It can improve a country's GDP and guide the development direction of national economy. Reducing transportation costs which is to make the shortest transportation path can effectively improve the efficiency of the logistics industry.

The traditional vehicle routing problem (VRP) is to study the problem of transportation path minimization. Because it requires that each customer's demand must be delivered by one transport vehicle, when the demand of some customers is greater than the vehicle capacity, it will not meet the needs of all customers. In order to solve this problem, the relevant scholars began to consider to split customer’s demand, and proposed the Split Delivery Vehicle Routing Problem (SDVRP). In the process of SDVRP research, it is found that the goods consumption during transit is not taken into account. Therefore, we add the route consumption to the constraint condition and put forward Split Delivery Vehicle Routing Problem with Goods Consumed during Transit (SDVRP-GCT). SDVRP-GCT can not only be applied to logistics distribution, but also to other problems in daily life.

Based on the SDVRP problem, this paper analyzes the current research situation and short board of the traditional SDVRP problem, combines the SDVRP problem with the real world logistics scene, proposes a more realistic SDVRP-GCT problem, and designs three extended algorithms based on ant colony algorithm to solve SDVRP and SDVRP-GCT. The data sets used in this paper includes: SDVRPLIB datasets, large-scale datasets in VRP web and tourism planning dataset collected in mathematical modeling website. Firstly, these three extended algorithms are applied to SDVRP, and

the results of the three algorithms are compared with the optimal solutions provided by data sets. The results show that these three extended algorithms can improve the existing optimal solutions, which shows that these three extended algorithms proposed in this paper have good performance in solving SDVRP problems. Then, we assume that the commodity consumption of vehicles during transit is proportional to the path length. The SDVRP benchmark data sets is transformed into SDVRP-GCT data sets, and these three extended ant colony algorithms are applied to the transformed SDVRP-GCT data sets. The experimental results show that our algorithms performed well in different types of data sets. In addition, we carry out experiments on four large-scale data sets in VRP web, and the experimental results show that our algorithms have good stability and still perform well when the nodes scale is large. Finally, these three extended ant colony algorithms is applied to the tourism route planning problem, which proves that the SDVRP-GCT problem has a wide range of application in real life.

To sum up, the main research work of this paper includes: studying the background and model of SDVRP, putting forward and modeling SDVRP-GCT problem; extending three ant colony algorithms, which can solve SDVRP and SDVRP-GCT problem at the same time; designing experiments, which prove that the three extended ant colony algorithms in this paper can effectively solve SDVRP and SDVRP-GCT problem; finally, the living tourism planning problem is modeled as SDVRP-GCT problem, and solved by three extended algorithms proposed in this paper, which proves that this work has a wide application prospect in real life.

Keywords:

Vehicle Routing Problem, Split Delivery Vehicle Routing Problem, Ant Colony Optimization Algorithms, Goods Consumed during Transit, mathematical modeling.

目 录

第 1 章 绪论

1.1 研究背景及意义

在全球各个国家，物流行业的利润在 GDP 中都占有很大的比重，物流行业的发展程度不仅能代表一个国家的现代化发展水平，也能代表一个国家的经济实力。伴随着我国经济水平的大幅提高以及计算机技术的进步，我国的物流行业也得到了迅猛的发展。据中国物流与采购联合会于 2019 年 7 月发布的《2019 年上半年物流运行通报》[1]统计数据显示，2019 年上半年全国的物流总额达到了 139.5 万亿元。由此可见，物流行业在促进我国经济发展的过程中起到了至关重要的作用。如果各大企业能够对运输车辆进行合理的路线规划和正确的调度，就能减少资源浪费，降低车辆的运输成本。这样不仅提高了物流配送的效率，又给国家的经济进步做出了巨大贡献。

在整个物流行业中最核心的环节就是运输配送，由于商品本身的成本因素是不可控的，所以主要通过降低运输成本来获得更大的利润。车辆的运输路线会在很大程度上影响配送的及时性，路途成本和总利润，因此利用信息技术设计出合理的车辆路径对于降低运输成本，提高客户满意度，提高物流质量，提高企业利润是至关重要的。车辆路径问题的目标是使运输成本或者路径长度最小化，所以为了进一步促进物流业的发展，相关研究学者们开始从事车辆路径问题的研究。结果表明研究车辆路径问题不仅推动了物流行业的发展，也提高了国家现代化水平。

车辆路径问题除了应用在物流运输行业，同时也应用在社会中各个领域[2]，比如机场飞机调度，医院护士排班，邮局邮件投递，美团外卖送餐，紧急情况下的食物补给和物资运输，以及医疗救援等等。例如当疫情爆发时需要往疫区运送大量的医疗物资，比如口罩，药品，防护衣服等等，为了保证救援物资尽快以最快速度到前线，运输路径的优化是至关重要的。通过路径规划尽量减少运输时间与运输消耗，就能及时将药物送到医患人员手中，拯救更多患者的性命。因此加大车辆路径问题的研究力度刻不容缓，需要更多有价值的研究成果为我国物流运输，救援物资发放等等提供技术支持。

作为一个典型的组合优化问题，车辆路径问题（Vehicle Routing Problem，VRP)[3]主要是通过为一系列有容量限制的车辆进行路线规划，使所有顾客（只允许被访问一次）的需求得到满足，并达到一些预期的目标（路径长度最短，运输时间最少，运输成本最低等等）。但是在一些实际情况中，当顾客需求很小时会增加车的使用数量和车辆空载率，浪费车辆资源；当车的需求超过车的装载容量时则会导致这些车辆需求不能被满足。为了节约成本，减少车辆个数，必须考虑将顾客的需求拆分成多个车辆进行服务的情况，这启发了许多学者对需求可拆分车辆路径问题（Split Delivery Vehicle Routing Problem，SDVRP）[4]的研究。SDVRP 问题研究的重点是如何有效的使用车辆以及选择成本最低的路线，使企业能在最短的时间内为所有顾客进行服务并使所花费的运输总成本最小。

根据实际情况，在物流配送的过程中，不只顾客的需求会占用车辆的容量，路途中所消耗的水，燃油和食物等也会占用车的容量。因此，本文充分考虑到路途中的物资消耗也会影响路径选择，并对带有路径消耗的需求可拆分车辆路径问题（Split Delivery Vehicle Routing Problem with Goods Consumed during Transit，SDVRP-GCT）进行研究。值得注意的是，路途中的物资消费和客户需求都不一定是实际商品，也可以是虚拟的东西。“Goods”是一个总称，不局限于实际的物品，也可以是时间，碳排放量，客户满意度等等。本文首先针对新提出的 SDVRP-GCT 问题进行全面分析并建立数学模型；其次分析了求解组合优化问题的算法的优缺点，选择基于蚁群算法进行扩展，并结合 SDVRP-GCT 问题的模型设计了三个扩展的蚁群算法，最后在十八个基准数据集和一个现实数据集上进行实验，分析实验结果以说明本文所提出的扩展算法的有效性。

1.2 国内外研究现状

VRP 问题自 Dantzig 等人提出以来，因其能大量节约运输成本，提高企业效益等优势，受到了人们极大的关注。相关研究学者从不同的角度，不同的实用场景对该问题进行了深入的分析与研究。本文基于时间顺序来介绍几种经典的 VRP 变形。容量车辆路径问题（Capacitated VRP，CVRP）[3]考虑了每辆车的容量，要求每辆车访问的所有顾客的需求总和必须小于等于车辆容量。与时间相关的车辆路径问题（Time-Dependent VRP，TDVRP）[5]考虑了相对于时间的成本变化，比如反复出现的交通拥堵问题[6]会影响车辆的行驶速度，进而影响运输成本。带有

取货和配送的 VRP（VRP with Pickup and Delivery Problem）[7]可以追溯到 Wilson 和 Weissberg 研究的拨号叫车服务[8]，它主要考虑在进行配送服务的同时满足顾客的取货要求。多仓库 VRP（Multi-depot VRP，MDVRP）[9]问题来源于各种现实物流情形，比如在餐食、化工产品、包装食品的配送中都包含多个仓库，每个客户都由其中一个仓库的车辆提供服务（即每条路线必须从同一个仓库开始和结束），研究结果表明通过使用多仓库运输能在很大程度上节省运输成本[10]。

传统的 VRP 处理的问题都处在确定的情形中，在路径寻优之前所有的信息都是已知的和静态的，而现实生活中有很多不确定事件随机发生，比如车辆故障，交通拥堵和客户的随机需求等等。动态 VRP (dynamic VRP, DVRP)[11]的研究涉及到动态操作，比如当客户发出在线请求时应立即分配适当的车辆进行服务。DVRP 在现实生活中有很多方面的应用，比如动态车队管理，分销系统管理，快递服务，维修或救援服务，拨号服务，紧急服务，以及出租车服务等等。带有时间窗口的 VRP (VRP with Time Window)[12]要求每个客户都有一个交付截止日期和最早交付时间，除了要考虑车辆在运输途中的行驶成本之外，还要考虑车辆早到达时等待客户所产生的成本，这个时间窗口分为硬时间窗[13]和软时间窗[14]。在大多数 VRP 的变形问题中都要求每个客户只能被访问一次，但是这个限制在很多现实情形中是不成立的，因为当顾客需求量很小时会导致车辆的空载率很高，增加运输成本，而当顾客需求量很大时，有些超出车的装载容量的顾客需求则不能被满足。Dror 等人考虑放宽约束条件，允许多个车辆对一个客户进行配送。一开始人们普遍认为将顾客需求进行拆分会大大增加运输成本，然而大量实验表明拆分操作可以在很大程度上降低运输车辆数并减少总行驶距离，成本节约最多可达到 50%[15]。开放 VRP(Open VRP)[16]中考虑到一些企业没有物流服务，他们选择将配送任务承包给专业的物流公司，此时外部车辆只需要完成承包的任务即可，他们在为所有的顾客进行配送之后没有义务返回仓库。Repoussis 等人[17]的研究表明在现实的物流配送服务中有很多 OVRP 类型的问题，例如学校餐的递送、校车的路线选择、列车通过隧道的计划等。Li, Leung 和 Tian[18]等人认为在运输途中每辆运输车的型号并不相同，且运输成本与路径长度并不是线性相关，并提出了更符合外包承运人运输的实际情况的异构固定车队 OVRP 问题。多级 VRP (Multi-echelon VRP, MEVRP)研究的是多级配送策略，其中最常见的是两级 VRP (2EVRP)[19]的情况，即把仓库到顾客的服务拆分成了仓库到中间仓库，中间仓库

到客户的两级配送过程。它的目的是减少各级车辆的运输费用，进而减少总的运输成本。多级运输系统目前存在于许多现实的物流配送服务中，比如报纸发行、送货上门服务以及快递邮政服务。在传统的车辆路径问题中，人们普遍关注的是车辆路径对企业的经济影响而忽略了运输中的环境污染问题，因此基于不同配送策略中的环境影响，Erdogan 等人考虑了更多与可持续物流问题相关的约束条件，提出了新的车辆路径模型并将它们定义为绿色车辆路径问题 (Green Vehicle Routing Problems, GVRP) [20]，GVRP 的特点是通过实施有效的途径来满足环境问题和经济指标，从而降低能源消耗，协调环境和经济成本。

1.3 研究目标与内容

本文分析了现有的 SDVRP 的研究现状与短板，将其与真实世界中的物流场景相结合，提出一种更符合现实物流运输的研究问题；在考虑拆分顾客需求的同时考虑物流运输途中的物资消耗同样也会占用车辆容量的情况。一个典型的例子是在进行灾区救援任务时，长途运输的过程中也会消耗大量的资源，如水、食物和燃料等，这些物资的装载会影响顾客的配送。许多非货物运输问题也可以归纳为此类问题，例如在规划旅游路线时，不仅要考虑花在每个旅游景点的时间，而且在路途中花费的时间也应该作为旅行成本的一部分。基于这些现实情况，本文首先在 SDVRP 问题的基础之上提出了带有运输途中物资消耗的需求可拆分车辆路径问题或者简称为带有路径消耗的需求拆分车辆路径问题 (SDVRP-GCT)。在 SDVRP-GCT 中，车辆的装载容量被分为顾客需求容量和路径消耗容量。随后本文对该问题进行了数学建模。SDVRP-GCT 问题的优化目标扩展了传统 VRP 或 SDVRP 的优化目标，即增加了对运输过程中货物消耗的考虑，并且不仅要保证路径长度最小化，而且还要使路径上的总消耗量最小。然后本文针对 SDVRP 和 SDVRP-GCT 问题，提出了三种扩展的蚁群算法进行求解，并且在 SDVRP 基准数据集和转换后的 SDVRP-GCT 数据集以及自收集的真实数据集上进行实验，同时对实验结果的最小值，平均值，运行时间等进行全面的对比分析，最终得出不同算法的优缺点以及所适用的场景。

主要研究内容如下：

（1）归纳总结现有的 SDVRP 问题的变形问题以及求解算法，介绍 SDVRP 问题的一般模型与整数模型。

（2）考虑路径上物资消耗对物流运输的影响，提出带有路径消耗的需求可拆分车辆路径问题（SDVRP-GCT），并对该问题进行数学建模。

（3）分析现有的解决 SDVRP 问题算法的优缺点，阐述选择 ACO 算法进行扩占的原因，并基于 ACO 算法中最经典的三种算法（蚂蚁系统，蚁群系统，最大最小蚁群算法）结合新问题模型的特点设计了三种基于蚁群算法的扩展算法对 SDVRP 和 SDVRP-GCT 问题进行求解。本文在十八个基准 SDVRP 数据集和对应转化后的 SDVRP-GCT 数据集以及一个自收集的真实 SDVRP-GCT 数据集上进行实验，并分析实验结果，得出有指导意义的结论。

1.4 研究的技术路线与方法

本文的研究方法是：首先进行背景分析和研究国内外文献资料；随后构建问题模型，并结合新问题模型的特点对蚁群算法进行扩展，提出三种扩展的蚁群算法进行求解；最后在基准数据集以及转化后的数据集上进行实验并进行结果分析。本文的研究路线图如下图 1.1 所示：

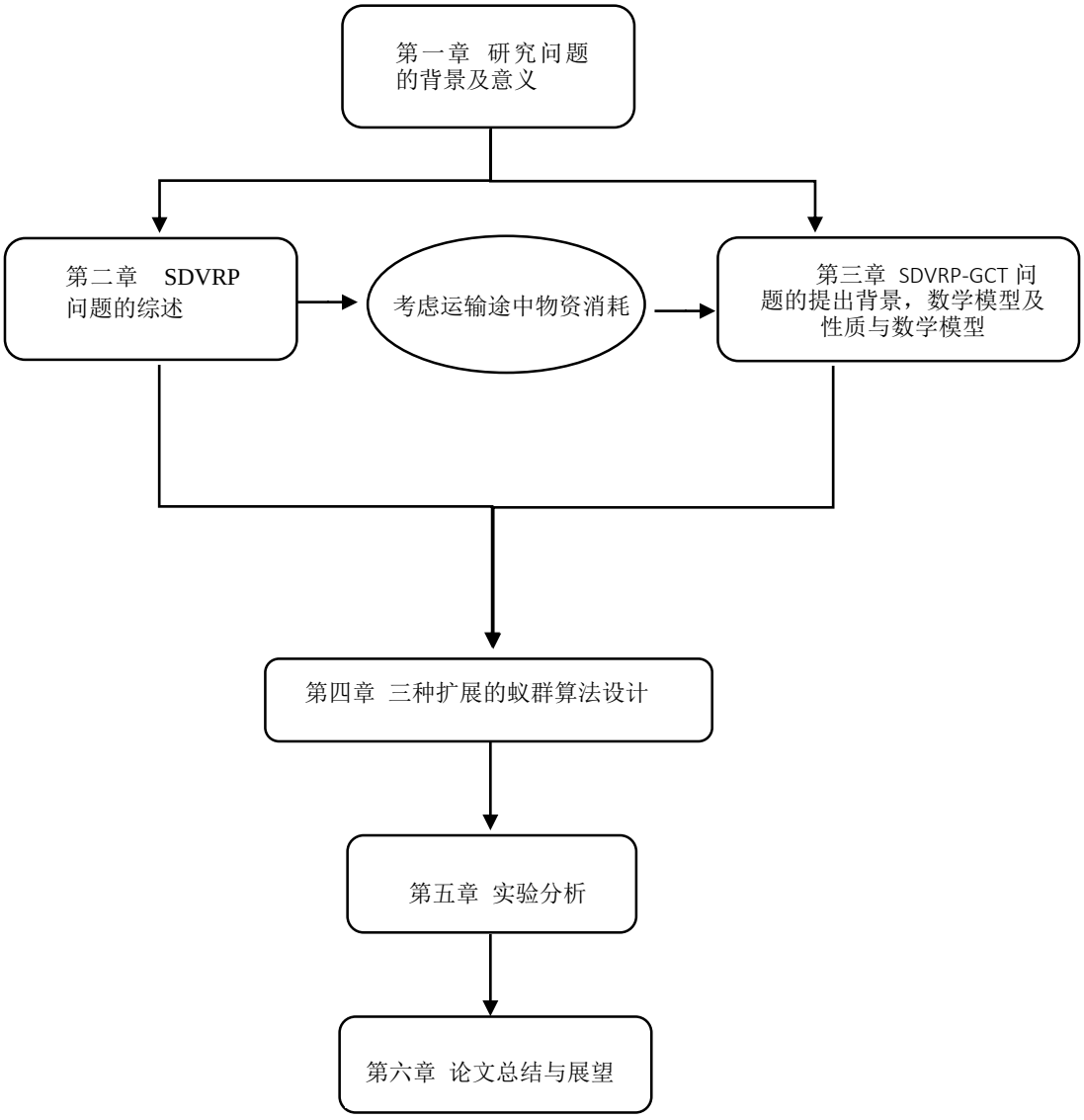


图 1.1 本文的研究路线

本文以物流产业中的车辆运输路径优化为基础，通过阅读大量国内外学者的文献以及分析实际生活中的物流运输情况，放宽每个顾客需求只能由一辆车进行服务的限制条件。本文通过研究 SDVRP 问题的数学模型，考虑到物流运输途中的物资消耗可能占用车辆容量，提出了一个更符合现实配送情况的 SDVRP-GCT 问题。全文的总体结构与安排如下：

第一章 绪论。介绍 VRP 问题的研究背景及意义，研究现状，技术路线和本文的结构框架。

第二章 SDVRP 问题的研究。首先介绍 SDVRP 问题的定义，以及国内外关于 SDVRP 问题的研究现状，分析现有的解决 SDVRP 问题的求解算法，然后介绍 SDVRP 问题的一般模型与整数模型。

第三章 SDVRP-GCT 问题研究。阐述 SDVRP-GCT 问题的提出背景，然后对 SDVRP-GCT 问题进行描述，最后建立模型，并分析该问题的基本性质。

第四章 三种扩展的蚁群算法设计。首先对蚁群算法概念，分类，原理，算法流程进行介绍，然后分析蚁群算法的优点，根据 SDVRP 和 SDVRP-GCT 问题的特点，设计了三种扩展的蚁群算法解决 SDVRP 和 SDVRP-GCT 问题，

第五章 实验设计与分析。将三种扩展算法应用在十四个 SDVRP 基准数据集，四个大规模 SDVRP 数据集和一个自收集真实 SDVRP 数据集上，随后对比三种算法的实验结果，并分析实验结果，最终得出实验结论。

第六章 论文总结与展望，总结整篇论文的工作内容与研究所做出的贡献，分析现有工作的不足，并展望未来的研究方向。

第 2 章 需求可拆分车辆路径问题研究

2.1 SDVRP 问题概述

SDVRP 问题是在传统 VRP 问题的基础上衍生出的一个新问题。传统的 VRP 中要求每个顾客有且仅能被一辆车进行服务，在 SDVRP 中这一约束条件被放宽为每个顾客点可以被一辆或者多辆车进行配送服务，这样既可以降低运输成本，又能够提高车辆的配送效率。SDVRP 问题主要研究如何对顾客需求进行正确的拆分，才能使路径长度达到最小。下图 2.1 展示了 VRP 与 SDVRP 的区别。

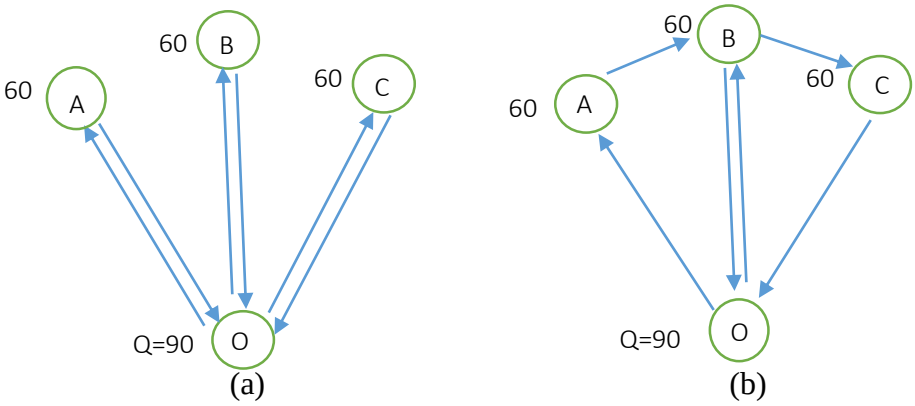


图 2.1 VRP 和 SDVRP 示例图

图 2.1(a)展示了一个包含四个节点的 VRP 示例。假设每辆车的容量是 90，O 为仓库结点， A、B 和 C 为三个顾客点，圆圈内数字表示每个顾客的需求。VRP 中规定在对客户进行货物配送时每个顾客有且仅能被访问一次，如果同时满足 A、B 和 C 三个顾客的需求，总共需要三辆车，车辆行驶的总路径长度为 2AO+2BO+2CO。图 2.1(b)展示了一个 SDVRP 实例。 SDVRP 继承了 VRP 的所有构成要素，包括一队车辆，一个仓库节点，一组确定的顾客。当顾客需求允许拆分时，可以将顾客 B 的需求拆分成 30 和 30，分配给两辆车来配送，此时总共需要两辆车，车辆行驶的总路径长度为 OA+AB+BO+OB+BC+CO。如图可视 AB+BC 的长度小于 AO+CO，即客户需求拆分之后车辆的行驶路线长度更短且配送所需的车辆数目更少，此时的运输成本小于顾客需求未拆分前的成本。

2.2 SDVRP 问题的研究现状

在过去的几十年里，研究学者们以传统的 SDVRP 问题为基础，考虑了多种 SDVRP 的变形问题。例如，带有时间窗口的 SDVRP（SDVRP with Time Window）[21]要求每个客户只能由一个或者多个车辆在指定的时间间隔内进行配送服务；带有货物权重的 SDVRP (Split Weighted VRP)[22]认为货物的重量会对运输成本产生影响，重量也可以表示客户的权重或者客户的优先级等。在多仓库 SDVRP（Multi-depot SDVRP）[23]中，仓库位置没有提前设定，它考虑在满足顾客需求的前提下确定多个仓库的位置和最佳运输路线。在商品受限的 SDVRP（Commodity-constrained SDVRP）[24]中，顾客可以有多种类型的商品需求，当车辆为顾客进行配送服务时，必须提供客户要求的某类商品的全部数量。具有最低交付额的 SDVRP（SDVRP with Minimum Delivery Amounts）[25]允许顾客需求被拆分，但每次访问都需要有一个最低的配送数量。具有 3D 装载约束的 SDVRP (SDVRP with 3D Container) [26]考虑使用最少数量的 3D 容器装载货物使装载成本最小。具有不确定的旅行时间和需求的 SDVRP (SDVRP with uncertain travel time and demands) [27]考虑为灾区人民运送关键物资时规划车辆路线。

2.3 SDVRP 问题的求解算法

自从 SDVRP 正式被定义以来，许多学者研究了它的解空间，解难度，解特性，解的上界与下界，求解的算法等。求解 SDVRP 的算法大致上分为两类，一类是精确算法，另一类是启发式算法。

2.3.1 精确算法

Dror 等人在 1994 年提出了第一个解决 SDVRP 问题的精确算法，他们提供了 SDVRP 的混合整数公式并结合一系列有效的不等式设计了弧柱方程。Belneguer 等人提出了切平面法[28],首先通过加入一些线性约束条件，切割掉非整数的可行解，然后将符合约束条件的解限制在整数范围内，最终找到问题的最优解。Jin 等人[29]提出了两阶段有效不等式算法，第一阶段创建覆盖所有需求的集群，并建立一个下界，第二阶段通过求解相应的问题来计算每个集群中的最小距离。Archetti 等人[30]提出了两种精确的分支限界算法，首先通过分枝将不符合约

束条件的非整数解排除，不断缩小问题的求解范围，最终找到问题的最优解，该算法在许多实验数据集上进行了大规模的测试，为最优解提高了下界值。

2.3.2 启发式算法

虽然通过精确算法能够得到问题的最优解，但是随着问题规模的增加，求解问题的复杂度会以指数的幅度增长。因此，精确算法在解决规模较小的问题时表现出了更稳定的性能，而在处理大规模问题时，人们往往采用启发式算法来获得“次优解”或“满意解”。

启发式算法在解决问题时首先设定一个初始解，然后按照启发式的原则选择下一个解，如果这个解优于之前的解，则从较优解出发寻找下一个解，直到达到迭代次数为止。启发式算法的求解速度快，并且在大规模的数据集上表现出了较稳定的性能。Dror 等人在 1990 年提出的本地搜索算法是第一个关于解决 SDVRP 问题的启发式算法。目前最常用的启发式算法主要包括：模拟晶体的退火原理而提出的模拟退火算法（Simulated Annealing）[31]；模拟生物界的“优胜劣汰，适者生存”的规律而演变出的遗传算法（Genetic Algorithm）[32]；禁忌搜索（Tabu Search）算法[33]是建立一个禁忌表记录已经访问过的结点，在后面的搜索中在未访问的点中进行搜索，减少了搜索的复杂度，从而快速找到最优解；蚁群优化（Ant Colony Optimazation）[34]是模拟蚁群寻找食物时的行为，利用蚁群觅食时的正反馈机制，移动规则，觅食规则等方式快速逼近最优解；人工神经网络（Artificial Neural Network）[35]是对人脑神经元网络进行抽象，模拟人脑的信息处理方式，它首先将给定的数据输入到网络模型中，通过一系列运算得到输出值，然后根据模型预测值与真实数据值之间的差距调整网络模型中的参数，指导模型输出结果达到预期值，最后利用训练好的网络模型对问题进行求解；粒子群优化（Particle Swarm Optimization）[36]是模拟鸟群捕食行为，将问题的每个解视为粒子，每个粒子都有位置向量和速度向量，根据目标函数计算出当前粒子的适应值，然后进行迭代直到找到问题的最优解。

2.4 SDVRP 问题建模

在对 SDVRP 问题进行数学建模时，首先介绍一下本文所研究的 SDVRP 问题的前提条件：（1）本文研究的所有实体的属性，即车辆类型、仓库的位置和

客户的位置和需求在整个服务期间都将保持不变。此外，所有参与的客户都不会相互影响。（2）所有车辆都有相同的容量，且必须从仓库出发最终返回仓库。（3）所有车辆提供的服务必须是单一的装货问题（单一的卸货问题）。（4）车辆在整个运输途中为顾客服务的需求总量必须小于等于车的容量。

2.4.1 一般模型

在 SDVRP 问题中，一个配送中心（$depot$）有N个客户，假设图$G=(V,A)$，V是结点集合，包含0和V_c，其中0表示配送中心，$V_c=\{1,2,...N\}$表示顾客集合；配送中心有M辆相同型号的运输车，每辆车的最大容量为Q，第i个顾客的需求量q_i可以小于、等于或大于车的容量Q；每辆配送车的出发点和返回点都必须是仓库，边集合$A=\{(i,j)|i,j\in V,i\neq j\}$表示任意两顾客之间的连线，距离矩阵$D=(d_{ij})$表示顾客$i$和顾客$j$之间的距离$d_{ij}$，且每个客户可能会被多次访问；$S$是在满足所有顾客需求的前提下需要的最少车辆数，其中$S=\lceil\frac{\sum_{i=1}^{N}q_i}{Q}\rceil$。

定义决策变量如下：

表 2.1 SDVRP 问题中的参数列表

Symbol	Definition
N	顾客的数量
M	车的数量
d_{ij}	顾客i和顾客j之间的距离
Q	车辆的装载容量
x_{ijk}	车辆k是否通过边(i,j)的二进制标识符
y_{ik}	车辆k在客户点i的配送数量
q_i	第i个顾客的需求量

此时数学模型可以用公式(2.1)-公式(2.8)来表示：

$$Z=\min\sum_{i=0}^{N}\sum_{j=0,j\neq i}^{N}\sum_{k=1}^{M}x_{ijk}\cdot d_{ij} \quad \text{.....(2.1)}$$

$s.t.$

$$0\leq y_{ik}\leq q_i,\ i\in\{0,1,\ldots,N\},\ k\in\{1,\ldots,M\}\text{.....(2.2)}$$

$$\sum_{i=0}^{N}\sum_{k=1}^{M}x_{ijk}\geq 1,\ j\in\{0,1,\ldots,N\}\text{.....(2.3)}$$

$$\sum_{i=0}^{N} x_{ipk} - \sum_{j=0}^{N} x_{pjk} = 0,\ p \in \{0,\ldots,N\},\ k \in \{1,\ldots,M\} \text{ ..... (2.4)}$$

$$\sum_{k=1}^{M} y_{ik} = q_i,\ i \in \{0,1,\ldots,N\} \text{ .....(2.5)}$$

$$\sum_{i=1}^{N} y_{ik} \le Q,\ k \in \{1,\ldots,M\} \text{ .....(2.6)}$$

$$0 < q_i < Q,\ i \in \{0,1,\ldots,N\} \text{.....(2.7)}$$

$$\sum_{i=0}^{N} \sum_{j=0,j\neq i}^{N} x_{ijk} \le |s| - 1,\ 2 \le |s| \le n-1 \text{.....(2.8)}$$

其中公式(2.1)中定义的目标函数是为了最小化车辆行驶的总路线长度；公式(2.2)表示第k辆汽车给顾客i的配送量不能超过顾客i的需求量，且只有在车辆k给顾客i进行配送时y_{ik}才有意义；公式(2.3)表示至少有一辆车为客户点j进行配送服务；公式(2.4)表示到达某点的车辆数应该等于离开该点的车辆数；公式(2.5)表示所有顾客的需求任务必须全部被完成；公式(2.6)为车辆容量约束，即车的容量必须大于等于车在运输途中为顾客配送的总数量；公式(2.7)用于消除支路限制条件。

2.4.2 整数模型

在分析 SDVRP 一般模型的过程中，我们发现如果不对q_i进行整数拆分会大大增加问题的复杂程度，并且在实际配送过程中配送量也应该是整数，否则会给客户带来许许多多的麻烦，因此公式(2.9)-公式(2.16)给出 SDVRP 的整数模型：

$$Z = \min \sum_{i=0}^{N} \sum_{j=0,j\neq i}^{N} \sum_{k=1}^{M} x_{ijk} \cdot d_{ij} \text{.....(2.9)}$$

$s.t.$

$$\sum_{i=0}^{N} \sum_{k=1}^{M} x_{ijk} \ge 1,\ j \in \{0,1,\ldots,N\} \text{ .....(2.10)}$$

$$\sum_{i=0}^{N} x_{ipk} - \sum_{j=0}^{N} x_{pjk} = 0,\ p \in \{0,\ldots,N\},\ k \in \{1,\ldots,M\} \text{.....(2.11)}$$

$$\sum_{k=1}^{M} y_{ik} = q_i,\ i \in \{0,1,\ldots,N\} \text{ .....(2.12)}$$

$$\sum_{i=1}^{N} y_{ik} \leq Q,\ k \in \{1,\ldots,M\} \quad \text{.....(2.13)}$$

$$0 < q_i < Q,\ i \in \{0,1,\ldots,N\} \quad \text{.....(2.14)}$$

$$0 < y_{ik} \leq q_i,\ i \in \{0,1,\ldots,N\},\ y_{ik} \in \{1,2,...,q_i\},\ k \in \{1,2,...,M\} \quad \text{(2.15)}$$

$$\sum_{i=0}^{N} \sum_{j=0, j\neq i}^{N} x_{ijk} \leq |s| - 1,\ 2 \leq |s| \leq n-1 \quad \text{.....(2.16)}$$

在整数模型中增加约束条件(2.15)是为了保证将每个客户的需求任务进行整数拆分，它不仅仅是为了求解方便，也是为了更符合实际物流配送中的情况，比如牲畜运输，集装箱运输，人员运输等等都需要进行整数规划。

2.5 本章小结

本章首先展示一个示意图，解释了 SDVRP 和 VRP 的本质区别并对 SDVRP 问题进行概述，随后总结了 SDVRP 问题的国内外研究现状，然后简要介绍了目前在求解 SDVRP 问题时广泛采用的两类算法-精确算法和启发式算法，最后给出 SDVRP 的一般模型和整数模型，为下面提出带有路径消耗的 SDVRP 问题奠定了基础。

第 3 章 带有路径消耗的需求可拆分车辆路径问题研究

3.1 SDVRP-GCT 问题的提出背景

在 SDVRP 问题的研究中，人们考虑了各种实际情况，比如带有时间窗口的 SDVRP 问题，同时有取货送货任务的 SDVRP，多仓库的 SDVRP 和有商品限制的 SDVRP 等等，但是人们往往忽略了在配送过程中工作人员可能需要消耗食品，水等，车辆也需要消耗汽油等物资，在运输过程中车辆的容量不是全部用来为顾客进行配送，路途中的消耗也要占用车辆的容量。一个典型例子就是在为沙漠或者雷雨暴雪恶劣环境影响的地区进行救援或者补给时，救援队进行长途旅行（包括返程）需要消耗大量的物资，这些物资都会占用车辆的装载容量，所以车的容量并不是全部用于满足客户点的需求。在另一种情形下，一个旅行家希望开展全国范围内的自驾游，他计划以最少的时间游览完所有的 5A 级景点，此时不能只考虑他在景区内游玩花费的时间，他开车从一个景点到达另一个景点的过程中也需要花费时间。从上述两个场景可以看到运输途中消耗的并不一定是实际的物资，它也可以是时间等虚拟的东西。本文认为带有运输途中物资消耗的 SDVRP 问题是传统 SDVRP 问题的扩展，在这种情况下必须保证每辆车的容量除了用于顾客需求配送，还要满足车辆在运输途中的消耗。本文将这类未被充分研究的问题称之为带有路径消耗的需求可拆分车辆路径问题（SDVRP-GCT）。

3.2 SDVRP-GCT 问题的问题描述

本文首先给出 SDVRP 和 SDVRP-GCT 的示意图，然后对 SDVRP-GCT 问题进行数学建模。SDVRP-GCT 中有一些与 SDVRP 相同的元素，比如配送时只有一个仓库（$depot$），一系列确定的顾客节点和需求量，一个连接所有顾客点和仓库的网络图，图中的每条边表示两个顾客点之间的路径。图 3.1(a)是一个标准 SDVRP 问题，图中的#0表示仓库节点，#1,#2,和#3表示三个顾客点，每个顾客允许多辆配送车访问。需要特别注意的是每辆车都要从仓库出发，最终返回仓库。图 3.1(b)中每个顾客节点上的数字表示该顾客的需求量，边上的数字表示车在运输途中的消耗量。假如每辆车的容量都等于 15，对于 SDVRP 问题，一辆配送车

可以先访问节点1，然后访问节点3，因为这两个节点的需求量之和等于车的容量15。而在 SDVRP-GCT 问题中，一辆车不能同时给节点1和节点3进行配送，因为两个顾客需求量再加上路径消耗量是 2+10+3+5+2 = 22，超出了车辆的容量。

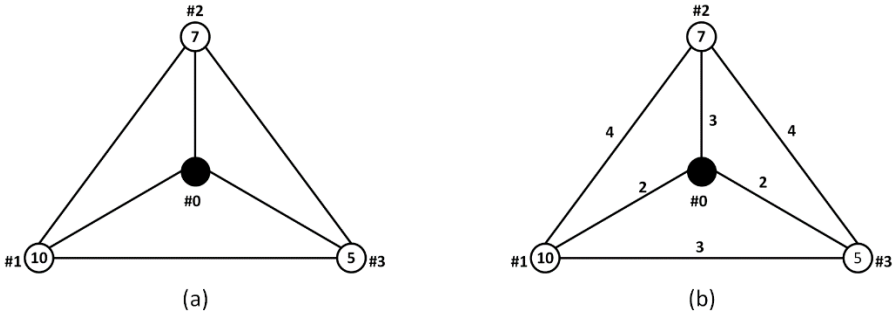


图 3.1 SDVRP 和 SDVRP-GCT 示例图

本文在给出 SDVRP-GCT 的数学描述之前，首先给出问题的假设：

（1）仓库和顾客点的坐标，每位顾客的需求量，以及每条路径上的消耗在整个交付期内都保持不变；

（2）任意两个顾客之间只有一条连通的路径，并且两个方向上的路径消耗是相等的；

（3）消耗只包括顾客的需求和路线消耗，所有其他的消耗，比如维修费，人员工资，服务时间等等，都不被考虑在内；

（4）运输途中的路径消耗量与路线距离成比例；

（5）任何顾客订购的货物和商品都可以直接从仓库获得，顾客与顾客之间并不进行交互。

（6）所有车辆的容量相同，客户对配送车辆没有偏好。

3.3 SDVRP-GCT 问题建模

SDVRP-GCT 中使用的所有参数和变量如表 3.1 所示，本文假设这个物流配送网络由N个顾客（节点），M辆车和一个仓库组成，仓库用节点 0 表示。网络中的每条边都与一定数量的路径消耗相关，每个顾客的需求由一辆车或者多辆车进行交付。

与 SDVRP 相比，本文在对 SDVRP-GCT 进行建模时考虑了车辆在运输途中的消耗。每辆车的容量被分为顾客需求容量和路径消耗容量，每辆车从仓库出发

到返回仓库过程中的总的消耗量（包括路径消耗和顾客需求）必须小于等于车的容量。

表 3.1 SDVRP-GCT 中的参数列表

Symbol	Definition
N	顾客的数量
M	车辆的数量
q_i	顾客i的需求量
r_{ij}	车辆从顾客i到顾客j所需的路径消耗量
d_{ij}	顾客i和顾客j之间的距离
x_{ijk}	车辆k是否通过边(i,j)的二进制标识符
y_{ik}	车辆k在客户点i的配送数量
Q	每辆车的容量
Q_c	每辆车的客户需求量
Q_r	每辆车的路径消耗量

SDVRP-GCT 的数学模型用公式(3.1)-(3.10)来表示：

$$Z = \min \sum_{i=0}^{N} \sum_{j=0, j\neq i}^{N} \sum_{k=1}^{M} x_{ijk} \cdot d_{ij} \quad \text{..... (3.1)}$$

$s.t.$

$$\sum_{i=0}^{N} \sum_{k=1}^{M} x_{ijk} \geq 1,\ j \in \{0,1,\ldots,N\} \quad \text{..... (3.2)}$$

$$\sum_{j=0}^{N} \sum_{k=1}^{M} x_{0jk} \leq M \quad \text{..... (3.3)}$$

$$\sum_{j=1}^{N} x_{0jk} = \sum_{j=1}^{N} x_{j0k} \in \{0,1\},\ k \in \{1,\ldots,M\} \quad \text{..... (3.4)}$$

$$\sum_{i=0}^{N} x_{ipk} - \sum_{j=0}^{N} x_{pjk} = 0,\ p \in \{0,\ldots,N\},\ k \in \{1,\ldots,M\} \quad \text{(3.5)}$$

$$\sum_{k=1}^{M} y_{ik} = q_i,\ y_{ik} = 1,2,...,Q,\ i \in \{0,1,\ldots,N\} \quad \text{.... (3.6)}$$

$$0 \leq y_{ik} \leq q_i,\ i \in \{0,1,\ldots,N\},\ k \in \{1,\ldots,M\} \quad \text{.... (3.7)}$$

$$\sum_{i=1}^{N} y_{ik} + \sum_{i=0}^{N}\sum_{j=0}^{N} x_{ijk} r_{ij} \le Q,\ k \in \{1,\ldots,M\} \text{ ..... (3.8)}$$

$$Q - (r_{0i} + r_{i0}) \ge 0,\ i \in \{0,1,\ldots,N\} \text{ ..... (3.9)}$$

$$\sum_{i=0}^{N}\sum_{j=0,j\neq i}^{N} x_{ijk} \le |s| - 1,\ 2 \le |s| \le n-1 \text{ ..... (3.10)}$$

公式(3.1)是 SDVRP-GCT 问题的目标函数，保证所有车辆的总路径长度最小；由于路径上的消耗量与路径长度成正比，因此该目标函数等价于最小化总路径消耗量；公式(3.2) 表示允许一辆或者多辆车访问客户点j；公式(3.3)表示同一时刻允许离开仓库的车辆数最多为M辆；公式(3.4)保证每辆离开仓库的车最终都要返回仓库；公式(3.5) 表示到达某顾客点的车辆数必须等于离开某顾客点的车辆数；公式(3.6) 表示顾客的所有需求任务必须被完成；公式(3.7)表示第k辆汽车给顾客i的配送数量不能超过顾客i的需求量；公式(3.10)用于消除支路限制条件。

在 SDVRP-GCT 问题中，本文将车辆容量Q划分为顾客需求容量Q_c和路径消费容量Q_r，它们分别对应每个顾客的需求量和运输途中的消耗量。本文根据是否考虑路径消耗，将 SDVRP-GCT 分为两种特定的情形。这两个情形描述如下：

（1）不考虑线路消耗容量Q_r。在这种情形下，对车辆的唯一约束是顾客需求容量Q_c，此时 SDVRP-GCT 问题转变成了一个标准的 SDVRP 问题，公式(3.9)应该被删除，公式(3.8)应该改为：

$$\sum_{i=1}^{N} y_{ik} \le Q_c,\ k \in \{1,\ldots,M\} \text{.....(3.11)}$$

（2）考虑路径消耗容量Q_r。在这种情况下，所有车辆都需要考虑路径消耗容量Q_r和顾客需求量Q_c。此时公式(3.8)是车辆容量约束，即车辆在路径上的消耗量加上顾客的需求量要小于车辆的容量；公式(3.9)表示车辆的容量Q必须要大于车从仓库结点到任意一个顾客点的往返路径消耗之和。

Archetti 等人在 2015 年和 2011 年发表的论文[37][38] 都证明了 SDVRP 属于 NP 难问题。SDVRP-GCT 问题比 SDVRP 问题更复杂，它考虑了运输过程中的物资消耗，因此 SDVRP-GCT 也是一个 NP-hard 问题。群智能算法是解决 NP 难问题时经常被使用的算法，为了解决 SDVRP-GCT 问题，本文考虑使用群智能算法进行求解。蚁群算法属于群智能算法的一种，该算法引入了正反馈机制，在求解 TSP

以及 VRP 问题中表现出了极好的鲁棒性，因此本文基于蚁群算法提出了三种扩展的算法来解决 SDVRP-GCT 问题，本文将在后面的章节详细介绍这三种算法。

3.4 本章小结

本章首先介绍了 SDVRP-GCT 问题的提出背景，随后通过一个实际的示例图说明了 SDVRP-GCT 与 SDVRP 在问题求解时的差异，然后为 SDVRP-GCT 问题建立数学模型，最后分析了 SDVRP-GCT 问题属于 NP 难问题以及选择蚁群算法求解 SDVRP-GCT 问题的原因，为下面介绍求解 SDVRP-GCT 问题的三种扩展蚁群算法奠定了基础。

第 4 章 三种扩展蚁群算法的实现

4.1 蚁群算法概述

蚁群算法是 Marco Dorigo 受到蚁群觅食行为的启发后，于 1992 年在博士论文中提出的一种智能仿生算法，它能在合理计算时间内得到组合优化问题的近似最优解。在大量的研究中人们发现蚁群之所以能很快的聚集在一条寻找食物的最短路径上，是因为蚁群中的蚂蚁在寻找路径时通过释放一种微量的化学物质-信息素进行交流。每只蚂蚁都会在途经过的路上释放信息素，其他蚂蚁在寻找食物的时候可以根据路径上信息素的残留程度选择出信息素浓度高的路径，这样一来该路径被选择的次数越多，信息素的浓度就越高，蚂蚁越有可能选择这条路径，因此在一段时间后所有的蚂蚁都会聚集在寻找食物的最短或者近似最短路径上。如果在最短路径上突然放置一个障碍物，蚁群也能很快的适应新的环境，迅速改变路径，找到食物的最优路径。下图 4.1[39]展示了蚁群的搜索机制。

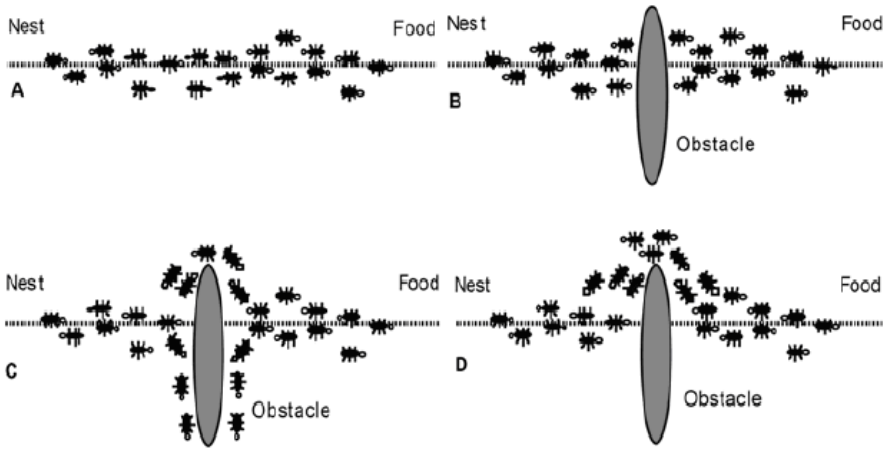


图 4.1 蚁群觅食行为示意图[39]

4.2 求解 SDVRP-GCT 问题的蚁群算法提出背景

起初相关学者们提出蚁群算法是为了解决旅行商（TSP）问题，后来大量研究都证明了蚁群算法在离散优化问题，组合优化问题上都有非常好的性能。由于 SDVRP-GCT 是一个首次被提出的问题，没有现成的算法可以直接使用，蚁群算

法与其他启发式算法相比，在求解组合优化问题时表现出了很好的稳定性，并且该算法模型易于修改，易与其他算法结合，在寻找最短路径的时候具有并行性，因此本文在解决 SDVRP 和 SDVRP-GCT 问题时选择基于蚁群算法进行扩展。

蚁群优化算法（ACO）是一类蚁群算法的统称，它是一个集成框架，包括了蚂蚁系统(Ant System，AS)[40]，精英蚂蚁系统[41]，快速蚂蚁系统[42],基于排序的蚂蚁系统[43]，蚁群系统（Ant Colony System，ACS）[44]，最大最小蚁群算法(Max-Min Ant System，MMAS)[45]等等。AS，ACS 和 MMAS 是 ACO 算法中最经典的三种算法，这些算法的实现过程是不同的，没有一种算法在各类问题上的性能都优于其它蚁群算法。比如 AS 算法是首次被提出的蚁群算法，它可以快速的找到求解方案，ACS 算法引入了全局和局部更新规则来调整信息素浓度使算法更加通用，MMGCT 算法通过限制信息素浓度的最大最小值，每次迭代只更新最佳路径中的信息素，将所有信息素浓度初始化为最大值能防止算法过早收敛到局部最优解。为了解决 SDVRP-GCT 问题，本文在现有的 AS，ACS，MMAS 算法的基础之上提出了三种扩展的蚁群算法，并分别给扩展算法命名为用于解决 SDVRP-GCT 问题的蚂蚁系统算法(Ant System for SDVRP-GCT，ASGCT)，用于解决 SDVRP-GCT 问题的蚁群系统算法(Ant Colony System for SDVRP-GCT，ACGCT)，用于解决 SDVRP-GCT 问题的最大最小蚁群算法(Max-Min Ant System for SDVRP-GCT，MMGCT)，下一小节介绍三种扩展算法的实现过程。

4.3 三种扩展的蚁群算法设计

4.3.1 ASGCT 算法的实现

本文在设计 ASGCT 算法的过程中，将蚂蚁看成是车辆，城市看成是顾客，构造运输路径的过程即是一个向路径中不断添加顾客节点的过程。首先定义所有的车辆必须全部从$depot$点出发，每辆车在出发之前都有一个容量Q，在选择下一节点时，并不是将所有未被访问过的顾客加入$allowed_k$，而是只将顾客需求Q_c大于 0，且满足路径消耗限制条件的顾客加入到$allowed_k$中。当选择完下一顾客之后，如果车辆的剩余装载容量不够车辆返回仓库，则对顾客需求进行拆分，直到$allowed_k$为空。起初所有蚂蚁都被放置在$depot$点上，然后将每条路径上的初始信息素设为 0，当蚂蚁选择路径的时候，也就是当蚂蚁k由当前顾客结点i转移到下

一个顾客结点j的时候，要计算$allowed_k$中的每一个顾客被选择的概率，然后根据轮盘赌选择出下一个满足配送限制条件的顾客。每个顾客的概率计算公式(随机比例规则)如下式(4.1)所示：

$$P_{ij}^{k}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}(t)]^{\beta}}{\sum_{u\in \text{allowed}_k}[\tau_{iu}(t)]^{\alpha}[\eta_{iu}(t)]^{\beta}}, & j\in allowed_k\\ 0, & otherwise\end{cases} \quad \text{.....(4.1)}$$

其中$allowed_k$中存储的是需求量大于 0 且满足路径消耗限制的所有顾客点；η_{ij}表示启发式信息，它是一个静态的值，只取决于两顾客点之间的距离，即$\eta_{ij}=1/d_{ij}$，d_{ij}表示两顾客结点之间的距离；α是信息启发式因子，反映了路径上残留的信息素对寻找最短路线的影响程度；β表示期望启发式因子，反映了启发式信息对寻找最短路线的影响程度。我们可以看到当路线(i,j)的长度越小时，η_{ij}的值越大，顾客j计算得到的概率值越大，则路线(i,j)更容易被选择，选择路线(i,j)的蚂蚁个数越多，路线上的信息素也会越多，顾客j被选择的概率也更高，这就是蚁群算法中的正反馈机制；τ_{ij}表示城市i和j连接的路径上的信息素浓度，它是一个随迭代次数增加逐渐变化的参数，当一次迭代过程结束之后，即所有城市的需求都已被满足，路线上的信息素矩阵τ_{ij}更新如下式(4.2)和(4.3)所示：

$$\tau_{ij}(t+1)=(1-\rho)\tau_{ij}(t)+\Delta\tau_{ij}(t,t+1)\text{.....} \quad (4.2)$$

$$\Delta\tau_{ij}(t,t+1)=\sum_{k=1}^{M}\Delta\tau_{ij}^{k}(t,t+1) \quad \text{.....(4.3)}$$

其中ρ表示信息素的挥发速度，它提高了全局的搜索能力；t表示迭代次数；$\Delta\tau_{ij}(t,t+1)$表示完成一次迭代过程后城市i和城市j之间路线上的信息素增加量初始时刻该值设为 0；$\Delta\tau_{ij}^{k}(t,t+1)$表示蚂蚁$k$在从城市$i$走到城市$j$的途中释放的信息素含量，该式的计算采用蚁周模型（Ant-Cycle Model），如下式(4.4)所示：

$$\Delta\tau_{ij}^{k}(t,t+1)=\begin{cases}z/C_k, & i,j\in R_k\\ 0, & otherwise\end{cases} \quad \text{.....(4.4)}$$

其中z是一个常数，表示蚂蚁在一次迭代过程中释放出的信息素总量，z值越大则一次迭代路线上积累的信息素越多，蚂蚁在下一次迭代中会迅速聚集在信息素高的路径上，算法会很快收敛，但同时也可能使算法进入局部最优解，影响算法的求解性能；C_k表示蚂蚁k在当前迭代过程中走过的路线距离；R_k表示蚂蚁k在当前迭代过程中所走过的路线。

为了更好的阐释 ASGCT 算法求解 SDVRP-GCT 问题的步骤，本文使用伪代码说明算法流程，ASGCT 的伪代码如下所示：

Algorithm 1: ASGCT

Input: α, β, ρ, M, z, Q, maxiter;
Input: Initialize $\tau_{ij}(0) = \tau_0$ for all node pairs
Output: shortest path
Set iterate index $t = 0$, current position $now = 0$;
for *t = 1 to maxiter* **do**
  for *ant k= 1 to M* **do**
    Set $demand$ = demand of nodes;
    Set capacity v_k = Q;
    while $\sum_{j=1}^{N} demand_j \neq 0$ **do**
      Put nodes required constraints that $demand_j > 0$ and
       $v_k - (c_{now,j} + c_{j,0}) > 0$ into $allowed_k$;
      if *$allowed_k$ is not empty* **then**
        Compute transition probability according to (4.1);
        Select the subsequent node j according to roulette wheel selection,
         then add j into the path;
        $v_k = v_k - c_{now,j}$, now = j;
        if $v_k - demand_j \geq c_{j,0}$ **then**
          $demand_j = 0$; $v_k = v_k - demand_j$;
        end
        else
          $demand_j = demand_j - (v_k - c_{j,0})$;
          $v_k = c_{j,0}$;
        end
      end
      else
        Return to depot 0; Reload set v_k = Q;
      end
    end
    if *ant k has not returned to the depot* **then**
      Add 0 to the path;
    end
  end
  Save the shortest paths of all M ants in this iteration;
  Update pheromone according to (4.2), (4.3) and (4.4);
  $t = t + 1$;
end

4.3.2 ACGCT 算法的实现

ACGCT 算法是基于 ACS 算法扩展出来的解决 SDVRP 和 SDVRP-GCT 问题的算法，为了在解空间中寻找到更好的结果，ACS 算法引入了伪随机比例转换规则，局部更新规则和全局更新规则。具体来说主要有以下三点不同。

（1）首先在蚂蚁选择下一个城市结点时运用的状态转移公式被称为伪随机比例转换公式，公式如下所示：

$$j=\begin{cases}\arg\max_{j\in\text{allowed}_k}\{[\tau(i,j)]^{\alpha}[\eta(i,j)]^{\beta}\}, & \text{if } q\le q_0\\ P_{ij}^{k}(i,j), & otherwise\end{cases} \quad .....(4.5)$$

其中q是均匀分布在[0,1]内的随机数，q_0是一个参数($q_0\in[0,\ 1]$)，η , α, β, ρ, $\Delta\tau_{ij}(t,t+1)$和$P_{ij}^{k}(i,j)$与 4.3.1 中介绍的含义相同。这种状态转移规则与之前的随机比例规则一样，它倾向于向短边和信息素量大的边转移，参数q_0决定了利用之前蚂蚁走过的路径和探索新路径的相对重要程度，即结点 r 的蚂蚁选择下一个节点s的时候需要首先抽样一个随机数q，如果q小于等于q_0，则蚂蚁选择之前蚂蚁走过的最短路径，反之蚂蚁计算每个点的概率值，然后按照轮盘赌选择新的路径。

（2）在构造最优解的过程中，对所有的路径进行局部信息更新规则，也就是在选择下一城市结点的时候，更好的利用了局部最优解，即当蚂蚁选择下一个城市时，当前城市结点i到下一城市结点j的信息素按照如下局部更新规则进行更新：

$$\tau_{ij}(t+1)=(1-\epsilon)\tau_{ij}(t)+\epsilon\tau_0 \quad .....(4.6)$$

其中ϵ表示局部更新规则中的信息素蒸发率；τ_0表示路径中信息素的初始值，该值一般被设为一个接近于 0 的正数

（3）只允许所有迭代过程中所走路线最短的蚂蚁释放信息素，即在完成本次迭代之后，应用(4.7)(4.8)更新全局最短路径上的信息素，(4.7)(4.8)也被称为全局更新规则。全局更新规则与伪随机比例规则的结合是为了使搜索更具有针对性，使蚂蚁在全局最佳路线的邻域中搜索，算法更易找到最优解。

$$\tau_{ij}(t+1)=(1-\rho)\tau_{ij}(t)+\rho\Delta\tau_{ij}(t,t+1)..... \quad (4.7)$$

$$\Delta\tau_{ij}(t,t+1)=1/Cost^{gb} \quad .....(4.8)$$

其中的上标gb表示在算法迭代t次后的全局最佳路径，$Cost^{gb}$表示全局最佳路径的长度。

求解 SDVRP-GCT 的 ACGCT 算法如下：

Algorithm 2: ACGCT

Input: α, β, ρ, ϵ, M, Q, q_0, $maxiter$;
Input: Initialize $\tau_{ij}(0) = \tau_0$ for all node pairs
Output: shortest path
Set iterate index $t = 0$, current position $now = 0$;
for *t = 1 to maxiter* **do**
  for *ant k= 1 to M* **do**
    Set $demand$ = demand of nodes;
    Set capacity v_k = Q;
    while $\sum_{j=1}^{N} demand_j \neq 0$ **do**
      Put nodes required constraints that $demand_j > 0$ and $v_k - (c_{now,j} + c_{j,0}) > 0$ into $allowed_k$;
      if *$allowed_k$ is not empty* **then**
        Compute transition probability according to (4.5);
        Select the subsequent node j according to roulette wheel selection, then add j into the path;
        $v_k = v_k - c_{now,j}$, now = j;
        local update pheromone according to (4.6);
        if $v_k - demand_j \geq c_{j,0}$ **then**
          $demand_j = 0$; $v_k = v_k - demand_j$;
        end
        else
          $demand_j = demand_j - (v_k - c_{j,0})$;
          $v_k = c_{j,0}$;
        end
      end
      else
        Return to depot 0; Reload set v_k = Q;
      end
    end
    if *ant k has not returned to the depot* **then**
      Add 0 to the path;
    end
  end
  Save the shortest paths of all M ants in this iteration;
  Update global pheromone according to (4.7) and (4.8) for the global-best solution;
  $t = t + 1$;
end

4.3.3 MMGCT 算法的实现

在 ASGCT 和 ACGCT 算法的搜索过程中，通过增加最短路径上的信息素浓度值可以获得更好的性能，但是也可能会引起搜索过早进入停滞状态，即所有蚂蚁都聚集在信息素浓度最大的路线上，不能再探索出新的更短的行走路线，因此实现蚁群算法最佳性能的一个关键就是利用搜索中发现的最短路线并与有效避免搜索停滞的机制相结合，最大最小蚁群算法（MMAS）就实现了这样一个需求。为了利用算法运行过程中发现的最短路径，则在每次迭代结束后，只更新最短路径上的信息素浓度。这个路线可能选择全局最短路线也可以是局部最短路线（每次迭代中找到的最短路线）。为了避免搜索陷入停滞，所有路线上的信息素范围限制在一个区间[τ_{min},τ_{max}]内。此外在算法开始之前将所有边上的信息素浓度设置为最大值τ_{max}，以这种方式使算法在开始时就能深入的探索解决方案。

由于 MMAS 在问题求解时的这些改进，使得 MMGCT 算法在求解 VRP 问题时表现出了更稳定的性能，下面来介绍 MMGCT 算法所用到的公式以及 MMGCT 算法的伪代码。

（1）在算法开始之前将所有边上的信息素浓度设置为最大值τ_{max}，即$\tau_{ij}(t)=\tau_{max}$。

（2）在 MMGCT 中，每次迭代完成后根据公式(4.8)选择一条最短路径进行信息素更新，在公式(4.9)中本文选择使用的是当前迭代中的最佳路径而不是全局最佳路径，其原因是当选择全局最优解的时候，算法过于集中在此最优解的邻域内，不易于发现更好的解决方案，这限制了算法的搜索空间，而使用局部最优解的时候每次迭代过程中寻找到的最优解很可能是不同的，此时信息素在各条局部最优路线上的增加是分散的，避免了某一条路径上的信息素含量过高。

$$\tau_{ij}(t+1)=(1-\rho)\tau_{ij}(t)+\Delta\tau_{ij}(t,t+1) \quad \text{(4.9)}$$

$$\Delta\tau_{ij}(t,t+1)=z/Cost^{ib} \quad \text{(4.10)}$$

其中ib（iteration-best）表示本次迭代过程中的最佳解决方案，$Cost^{ib}$表示本次迭代中的最短路线的距离，$\Delta\tau_{ij}(t,t+1)$与 4.3.1 中介绍的相同。

（3）如果某条路线上的信息素浓度明显高于其他路线上的信息素，则意味着蚂蚁在构造路径的时候将先考虑这条路径，此时蚂蚁会一遍一遍的重复构造相同的行驶路径，为了避免这种停滞的局面，就要限制路径上的信息素含量，避免其过高或者过低，控制路径上的信息素在τ_{max}和τ_{min}之间，即如果$\tau_{ij}(t)<\tau_{min}$，

则令$\tau_{ij}(t)=\tau_{min}$；如果$\tau_{ij}(t)>\tau_{max}$，则令$\tau_{ij}(t)=\tau_{max}$。τ_{max}和τ_{min}的计算公式如下所示：

$$\tau_{\max}=1/(\rho\cdot Cost^{ib}) \quad \text{.....(4.11)}$$

$$\tau_{\min}=\tau_{\max}\cdot(1-\sqrt[n]{P_{best}})/((N/2-1)\cdot\sqrt[n]{P_{best}}) \quad \text{.....(4.12)}$$

其中P_{best}是一个常数，如果P_{best}太小可能会导致$\tau_{min}>\tau_{max}$，为了避免这一现象，P_{best}一般设置为 0.05。

求解 SDVRP-GCT 的 MMGCT 算法如下所示：

Algorithm 3: MMGCT

Input: α, β, ρ, ϵ, M, Q, q_0, maxiter;
Input: Initialize $\tau_{ij}(0) = \tau_{max}$ for all node pairs
Output: shortest path
Set iterate index $t = 0$, current position $now = 0$;
for *t = 1 to maxiter* **do**
  for *ant k= 1 to M* **do**
    Set *demand* = demand of nodes;
    Set capacity v_k = Q;
    while $\sum_{j=1}^{N} demand_j \neq 0$ **do**
      Put nodes required constraints that $demand_j > 0$ and $v_k - (c_{now,j} + c_{j,0}) > 0$ into $allowed_k$;
      if $allowed_k$ *is not empty* **then**
        Compute transition probability according to (4.5);
        Select the subsequent node j according to roulette wheel selection, then add j into the path;
        $v_k = v_k - c_{now,j}$, now = j;
        if $v_k - demand_j \geq c_{j,0}$ **then**
          $demand_j = 0$; $v_k = v_k - demand_j$;
        end
        else
          $demand_j = demand_j - (v_k - c_{j,0})$;
          $v_k = c_{j,0}$;
        end
      end
      else
        Return to depot 0; Reload set v_k = Q;
      end
    end
    if *ant k has not returned to the depot* **then**
      Add 0 to the path;
    end
  end
  Save the shortest paths of all M ants in this iteration;
  Update global pheromone according to (4.9) and (4.10) for the global-best solution;
  Compute τ_{max} and τ_{min} according to (4.11) and (4.12);
  if $\tau_{ij}(t) < \tau_{min}(t)$ **then**
    $\tau_{ij}(t) = \tau_{min}(t)$;
  end
  else if $\tau_{ij}(t) > \tau_{max}(t)$ **then**
    $\tau_{ij}(t) = \tau_{max}(t)$;
  end
  $t = t + 1$;
end

4.4 本章小结

本章主要对蚁群算法进行了概述，介绍了蚁群算法的基本原理，并阐述了选择蚁群算法进行扩展的原因，并根据 SDVRP-GCT 问题的特点来进行扩展，然后介绍了三种扩展算法之间的区别与每种算法的特点，算法实现过程中所用到的公式，每个参数的含义，最后给出了 ASGCT，ACGCT，MMGCT 算法的伪代码。

第 5 章 实验设计与分析

5.1 三种扩展算法在 SDVRP 基准数据集上实验

为了测试提出的三种扩展的蚁群算法应用在 SDVRP 问题上的性能，本文选取了 SDVRPLIB 中的 14 个 SDVRP 基准数据集进行实验。因为当运输途中消耗的货物量设置为 0 时，SDVRP-GCT 问题就变成了一个标准的 SDVRP 问题，因此此时可以直接使用提出的 ASGCT、ACGCT 和 MMGCT 三种扩展算法进行实验。首先展示本文使用的数据集，如表 5.1 所示。第一列表示数据集的名称，其中 S 后面的数字表示顾客的数量，D 后面的数字表示客户的需求水平（数字越大表示客户平均需求量就越多）。例如，S51D1 表示在该数据集中有 51 个客户，在选取的所有包含 51 个客户的数据集中，该数据集的客户平均需求量最小。“MNV”表示每个数据集中使用到的最少车辆数目；“Dimension”表示对应数据集的顾客数量；“Q”表示对应数据集的车辆装载容量；“D”表示对应数据集的客户需求范围，例如在 S51D1 中顾客的需求量是在车辆容量的 1%-10%范围内随机产生的。“Optimal_solution”表示数据集贡献者提供的最优解。这些数据集都可以通过 URL: http://www.uv.es/belengue/sdvrp.html 进行下载。

表 5.1 十四个 SDVRP 基准数据集的详细信息

Name	MNV	Dimension	Q	D	Optimal_value
S51D1	3	5	160	⌈0.01Q,0.1Q⌋	458
S51D2	9	51	160	⌈0.1Q,0.3Q⌋	726
S51D3	15	51	160	⌈0.1Q,0.5Q⌋	972
S51D4	27	51	160	⌈0.1Q,0.9Q⌋	1677
S51D5	23	51	160	⌈0.3Q,0.7Q⌋	1440
S51D6	41	51	160	⌈0.7Q,0.9Q⌋	2327
S76D1	4	76	160	⌈0.01Q,0.1Q⌋	594
S76D2	15	76	160	⌈0.1Q,0.3Q⌋	1147
S76D3	23	76	160	⌈0.1Q,0.5Q⌋	1474
S76D4	37	76	160	⌈0.1Q,0.9Q⌋	2257
S101D1	5	101	160	⌈0.01Q,0.1Q⌋	716
S101D2	20	101	160	⌈0.1Q,0.3Q⌋	1393
S101D3	31	101	160	⌈0.1Q,0.5Q⌋	1975
S101D5	48	101	160	⌈0.3Q,0.7Q⌋	2915

本文首先直接评估三种扩展的蚁群算法在 SDVRP 基准数据集上的有效性。具体来说就是把路径消耗的比例系数 h 设置为 0（即在运输途中没有货物消耗），然后将实验得到最短路径长度与数据集贡献者提供的最优解进行比较。需要注意，在本实验中数据集的结点 1 表示仓库，车辆容量等于数据集中的 Q，即实验中的车辆容量等于 160。本文在每个数据集上都进行了 20 次独立的实验，表 5.2 展示了三个扩展算法在每个数据集上得到的最短路径长度与数据集最优解之间的比较。

表 5.2 三个扩展算法在 14 个 SDVRP 基准数据集上的结果比较

Name	Optimal _value	Shortest Route Length		
		ASGCT	ACGCT	MMGCT
S51D1	458	478.44	476.30	466.25
S51D2	726	775.40	753.12	**724.07**
S51D3	972	1039.49	1031.07	996.81
S51D4	1677	1687.84	**1667.97**	**1620.41**
S51D5	1440	**1428.66**	**1423.40**	**1380.21**
S51D6	2327	2395.19	2387.82	2348.00
S76D1	594	649.42	623.62	612.70
S76D2	1147	1199.38	**1143.87**	**1130.36**
S76D3	1474	1594.18	1574.8	1532.84
S76D4	2257	2304.28	2270.84	**2232.96**
S101D1	716	777.25	756.71	744.46
S101D2	1393	1524.47	1515.06	1467.68
S101D3	1975	2020.34	2008.19	1997.26
S101D5	2915	3010.29	2967.30	2942.52

通过表 5.2 可以看到，ASGCT 在 S51D1 数据集上得到的最短路径长度优于数据集中提供的最优解，ACGCT 在 S51D4、S51D5 和 S76D2 数据集上得到的最短路径优于最优解，而 MMGCT 则在 S51D2、S51D4、S51D5、S76D2 和 S76D4 这五个数据集上得到的最短路径长度优于最优解。

为了进一步量化所提出的求解算法与最优解之间的性能差异，本文使用下面的公式来测量相对差异，并且在表 5.3 中展示了三种算法在各个数据集上的 gap 值。

$$\text{gap} = \frac{\text{Len} - \text{Optimal\_value}}{\text{Optimal\_value}} \quad \text{.....(5.1)}$$

表 5.3 三个扩展算法在 14 个 SDVRP 基准数据集上的结果比较

Name	Gap (%)		
	ASGCT	ACGCT	MMGCT
S51D1	4.53	3.99	1.80
S51D2	6.80	3.74	**-0.30**
S51D3	6.94	6.08	2.55
S51D4	0.65	**-0.54**	**-3.37**
S51D5	**-0.79**	**-1.15**	**-4.15**
S51D6	2.93	2.61	0.90
S76D1	9.33	4.99	3.15
S76D2	4.57	**-0.27**	**-1.45**
S76D3	8.15	6.84	3.99
S76D4	2.10	0.61	**-1.07**
S101D1	8.55	5.69	3.98
S101D2	9.44	8.76	5.36
S101D3	2.30	1.68	1.13
S101D5	3.27	1.79	0.94
Avg.gap(%)	4.91	3.20	0.96

本文在表 5.3 中列出了三个算法在 14 个 SDVRP 基准数据集上的 gap，可以更清楚的表现三种扩展算法应用于 SDVRP 问题上的效果。由表 5.3 可知，ASGCT 的 gap 范围是[-0.79,9.44]，ACGCT 的 gap 范围是[-1.15,8.76]，MMGCT 的 gap 范围是[-4.15,5.36]，由此可以看出三种算法 gap 的波动幅度非常接近。因此可以看出三个拓展算法被用于解决 SDVRP 问题时具有一定的稳定性。同时可以看到的是，表 5.3 中的部分 gap 为正值，即扩展的三个算法在这些数据集上的结果大于数据集本身提供的最优值，这并不意味着三个算法求解出的 SDVRP 问题的解不够好。同时可以注意到的是，ASGCT，ACGCT 和 MMGCT 在 14 个数据集上的平均 gap 是可以接受的足够小的值，分别为 4.91%，3.20%和 0.96%。这意味着这三个拓展算法计算出的解已足够接近已知的最优解，尤其是 MMGCT 算法。因此可以证明本文提出的三个拓展算法可以应用于 SDVRP 问题，并可以得出稳定的解。下面展示 20 次独立运行的实验结果的平均值与标准方差，如下表 5.4 所示。表 5.4 表明三种算法应用于 SDVRP 问题得到的解足够稳定。

表 5.4 三个扩展算法在 14 个 SDVRP 基准数据集上的结果比较

Name	Optimal_ value	Average (std)		
		ASGCT	ACGCT	MMGCT
S51D1	458	484.98 (3.72)	484.63 (6.86)	476.84 (5.99)
S51D2	726	785.75 (4.42)	768.75 (6.39)	741.93 (8.88)
S51D3	972	1055.46 (7.93)	1041.03 (3.79)	1019.16 (12.51)
S51D4	1677	1697.40 (5.05)	1681.02 (5.32)	1640.93 (10.42)
S51D5	1440	1456.13 (8.09)	1452.42 (11.53)	1402.86 (10.06)
S51D6	2327	2411.54 (8.14)	2410.10 (8.11)	2359.17 (6.52)
S76D1	594	656.43 (4.39)	638.40 (5.44)	622.98 (6.19)
S76D2	1147	1205.93 (3.11)	1173.83 (0.78)	1151.30 (8.98)
S76D3	1474	1614.96 (7.35)	1597.18 (9.90)	1551.91 (13.70)
S76D4	2257	2334.19 (11.70)	2280.77 (7.19)	2259.44 (23.69)
S101D1	716	787.52 (4.93)	776.48 (9.30)	762.31 (7.27)
S101D2	1393	1539.41 (5.76)	1523.71 (6.34)	1491.00 (11.72)
S101D3	1975	2052.15 (11.44)	2035.46 (11.15)	2030.69 (15.59)
S101D5	2915	3029.02 (8.15)	2984.10 (9.40)	2962.11 (16.51)

5.2 三种扩展算法在 SDVRP-GCT 数据集上的实验

由于 SDVRP-GCT 是本文提出的一个全新的问题，没有基准数据集可以直接使用，所以本文通过设置一个比例系数 h 来生成运输途中的物资消耗量。本文假设路途中的消耗量只与路径长度成比例，不受其他因素的影响，所以路途中的消耗量可以设置为路径长度与比例系数 h 的乘积。本文首先研究 h 值的大小对三种扩展的蚁群算法结果的影响。具体来说，首先在每组实验中分别设置 h=0、0.1、0.3 和 0.5。值得注意的是 h=0 时 SDVRP-GCT 就变成了一个标准的 SDVRP 问题，所以我们在这里直接使用上节中所得到的实验结果。而在 h=0.1、0.3 和 0.5 时分别做 20 次独立实验并选择出最好的实验结果进行对比。图 5.1 中用柱状图展示了随着 h 增大，三种算法得到的实验结果的变化趋势。

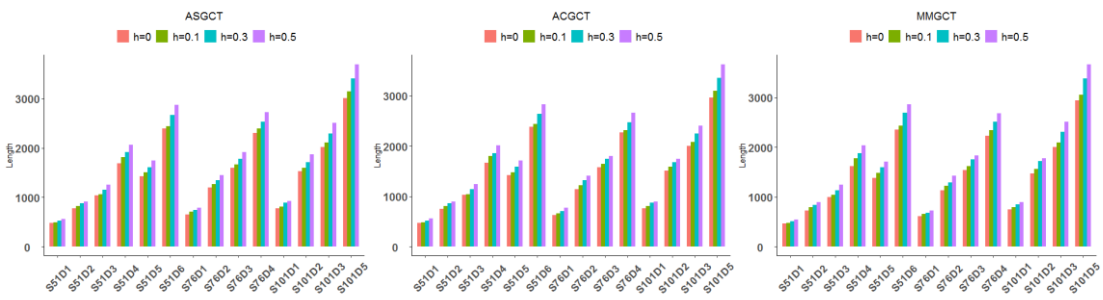


图 5.1 三种扩展算法在不同 h 时的最短路径长度比较

正如预期的那样，随着 h 的逐渐增加，车辆在运输的过程中消耗的货物量逐渐增加，此时用于装载顾客需求的容量逐渐减小，所以总的路径长度在增加。为了进一步评估三个扩展算法在新的 SDVRP-GCT 问题上的性能，本文以 h=0.1 为例进行实验然后详细的分析所得到的实验结果，并将结果展示在下表 5.5 中，最后从实验得到的最短路径长度，所需车辆数目，以及算法运行时间几个方面分别进行分析。

表 5.5 三种扩展算法在 14 个转化后的 SDVRP-GCT 上的结果比较

Name	ASGCT		ACGCT		MMGCT	
	Min	Avg(std)	Min	Avg(std)	Min	Avg(std)
S51D1	490.21	499.16(5.20)	481.85	487.98(4.61)	**473.34**	483.90(6.46)
S51D2	824.08	847.30(8.39)	806.78	816.91(6.33)	**789.19**	811.26(12.52)
S51D3	1065.51	1082.15(6.98)	**1036.60**	1056.46(17.97)	1043.19	1064.91(11.17)
S51D4	1815.09	1830.01(8.92)	1801.44	1819.24(7.67)	**1773.70**	1805.31(15.31)
S51D5	1502.25	1532.60(10.21)	**1479.22**	1506.42(11.26)	1485.07	1515.06(13.23)
S51D6	2443.97	2479.82(14.47)	2434.71	2450.68(7.92)	**2428.95**	2474.72(13.99)
S76D1	705.44	716.60(5.77)	665.23	681.37(8.79)	**655.56**	680.21(16.07)
S76D2	1263.20	1278.18(7.89)	1221.40	1260.07(13.66)	**1216.70**	1243.99(17.10)
S76D3	1661.41	1685.88(12.53)	1643.98	1668.23(11.63)	**1620.52**	1662.03(17.92)
S76D4	2396.63	2421.87(12.43)	**2321.08**	2361.50(16.82)	2344.62	2377.14(18.50)
S101D1	813.45	837.38(8.66)	803.16	820.23(9.38)	**793.60**	813.49(12.41)
S101D2	1596.01	1629.53(11.26)	1593.21	1613.29(11.81)	**1565.73**	1598.78(21.42)
S101D3	2109.01	2134.06 (11.93)	**2086.62**	2108.90(9.85)	2088.62	2111.64(17.35)
S101D5	3142.76	3204.34(30.51)	3098.82	3121.55(11.31)	**3057.73**	3105.59(19.53)

表 5.5 展示了在 14 个 SDVRP 基准数据集转换后的 SDVRP-GCT 数据集上的实验结果。其中 Min 表示算法从 20 次独立运行中获得的最短路径长度，Avg 表示 20 个实验结果的平均值，std 表示 20 次结果的标准方差，加粗的数字表示该算法在该数据集上的最短路径长度优于其他算法得到的最短路径长度。首先从

最短路径的角度分析三个算法。从表 5.5 可以看出 ACGCT 和 MMGCT 得到的最短路径长度总是优于 ASGCT 得到的实验结果，此外可以观察到在 S51D1、S51D2、S76D1、S101D1 和 S101D2 这些顾客平均需求量较低的数据集上，MMGCT 得到的最短路径长度要优于 ACGCT 得到的结果，而在 S51D3、S51D5、S76D4 和 S101D3 这些顾客的平均需求量较高的数据集上，ACGCT 的表现要优于 MMGCT，并且在各个数据集上 20 次结果的平均值也遵循相似的规律。除此之外还可以从表 5.6 展示的三种算法所需车辆数目和算法运行时间的角度分析三个扩展算法的性能。

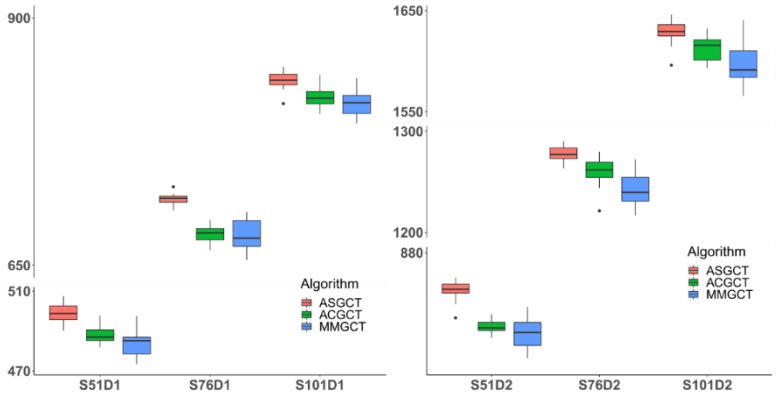
表 5.6 三种扩展算法在 SDVRP-GCT 数据集上的结果比较

Name	ASGCT		ACGCT		MMGCT	
	NV	T(s)	NV	T(s)	NV	T(s)
S51D1	3	**14.28**	3	17.49	3	17.76
S51D2	10	**16.82**	10	20.27	10	19.96
S51D3	15	**18.37**	15	21.99	15	23.53
S51D4	29	**23.90**	29	29.40	29	29.02
S51D5	24	**23.10**	24	27.19	24	27.00
S51D6	42	**33.97**	42	34.93	42	34.83
S76D1	5	**34.33**	5	36.01	5	37.42
S76D2	16	**39.44**	16	40.63	16	43.15
S76D3	24	**43.41**	24	46.23	24	47.30
S76D4	38	**51.09**	38	53.96	38	52.42
S101D1	6	**60.74**	6	60.74	6	61.04
S101D2	21	70.96	21	75.56	21	**70.03**
S101D3	32	**74.91**	32	78.78	32	76.88
S101D5	50	90.96	50	92.55	50	**88.72**

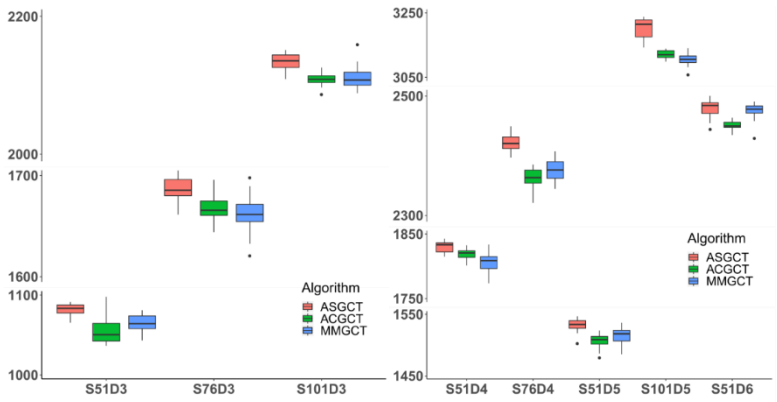
从表 5.6 可以看到，三个算法所得需的车辆数目相同，而从算法运行时间上看，ACGCT 在 12 个数据集上的运行时间都要比其他两个扩展算法短，在 S101D2 和 S101D5 上 ACGCT 算法的运行时间虽然比 MMGCT 的运行时间略长，但是仍然比 ACGCT 的算法运行时间要短。由此可以得出虽然 MMGCT 和 ACGCT 算法得到的最短路径长度优于 ASGCT，但是 ASGCT 算法的运行速度相对来说更快，算法所需的时间较短。

为了进一步研究三种扩展算法在解决 SDVRP-GCT 问题的性能上的差异，本文用 boxplot 图将 20 次实验结果以可视化的形式展示出来。图 5.2 展示了每个数据集上得到的 20 次实验结果的总体分布。Boxplot 图的顶部和底部边缘分别表示结果的上四分位数（Q3）和下四分位数（Q1），将盒子水平分割的线段代表结

果的中位数，内四分位数（IQR）等于上四分位数减去下四分位数，最大观测值等于 Q3+1.5*IQR，最小观测值等于 Q1-1.5*IQR，箱体外的须线是上下四分位数到最大最小观测值之间的延伸线。而盒子外的点代表异常值或者称为离群点，即在最大最小观测值范围外的点。箱线图对于确定大多数数据的位置很有用。图 5.2a 和 5.2b 为三种扩展蚁群算法在较低顾客需求量问题上的路径长度分布，图 5.2c 和 5.2d 为较高需求量问题的路径长度分布。箱线图对于确定大多数数据的位置很有用。从 boxplot 图中可以看出， ASGCT 得到的路径长度明显大于 ACGCT 和 MMGCT 得到的路径长度，说明 ACGCT 和 MMGCT 在所有 14 个问题上的表现都优于 ASGCT。为了进一步分析 ACGCT 和 MMGCT 算法性能的差异，本文使用单因素方差分析得到 ACGCT 和 MMGCT 结果对比的 p-value 值，并展示在表 5.7 中。从图 5.2 和表 5.7 能够看到对于 S76D1、S101D1、S51D2、S51D3、S76D3 和 S101D3 数据集，ACGCT 和 MMGCT 的性能没有显著差异。对于其他需求较低的问题，如 S51D1、S76D2 和 S101D2, MMGCT 箱形图的位置明显低于 ACGCT 箱形图的位置。ACGCT 与 MMGCT 的 p 值均小于 0.05，即 MMGCT 的表现明显优于 ACGCT。在其余 5 个需求较高的问题中，ACGCT 在 S51D5、S51D6 和 S76D4 上的表现明显好于 MMGCT，因为在这 3 个数据集中，ACGCT 算法结果构成的箱线图与 MMGCT 相比存在明显的向下倾斜，且 p 值小于 0.05。因此，ACGCT 在需求较高的问题上比 MMGCT 表现更好。结合以上的分析能够得出结论，ACGCT 和 MMGCT 的算法性能要优于 ASGCT，但是 ASGCT 算法运行时间较短，速度更快。MMGCT 在解决需求较低的问题性能更好，而 ACGCT 比 MMGCT 更适合解决需求较高的问题。



(a) (b)



(c) (d)

图 5.2 三种扩展算法在 SDVRP-GCT 上的 boxplot 比较

表 5.7 ACGCT 和 MMGCT 在 SDVRP-GCT 数据集上的 P-value 比较

Dataset	S51D1	S51D2	S51D3	S51D4	S51D5	S51D6	S76D1
p-value	**<0.05**	0.08	0.08	**<0.05**	**<0.05**	**<0.05**	0.78
Dataset	S76D2	S76D3	S76D4	S101D1	S101D2	S101D3	S101D5
p-value	**<0.05**	0.20	**<0.05**	0.06	**<0.05**	0.54	**<0.05**

5.3 三个扩展算法在大规模数据集上的实验

前面的实验主要是将三种扩展的蚁群算法应用在一些规模较小的问题上，但是蚁群算法在大规模数据集上仍然具有运行速度快，稳定性高等优点。为了测试算法在大规模数据集上的性能，本文从 VRP web 的 kelly 数据集中选择了前四个大规模的 SDVRP 基准测试集，因为这四个数据集涉及到的 SDVRP 问题是单一仓库的，这与本文 SDVRP 问题的描述一致，这些数据集都可以通过 URL：http://www.bernabe.dorronsoro.es/vrp/进行下载。所选数据集的细节如下表 5.8 所示，Name 表示数据集的名字，这四个数据集的客户数量范围是从 240 到 480，每个数据集都规定了不同的车辆容量，所有数据集中客户需求量的平均值为 20，标准差为 10，这说明每个问题的客户需求量较小，此外顾客的位置分布是以点为中心，像磁场一样向外扩散。

表 5.8 四个大规模 SDVRP 数据集的详细信息

Name	n	Capacity	Avg. demand	Std. demand
kelly01	240	550	20	10
kelly02	320	700	20	10
kelly03	400	900	20	10
kelly04	480	1000	20	10

5.3.1 三种扩展算法在大规模的 SDVRP 数据集上实验

本文先将三种扩展算法应用于四个大规模的 SDVRP 基准数据集上，并将实验结果与数据集提供的最优解进行对比，以评估三种扩展算法在大规模数据集上的性能。同样我们在每个数据集上都进行 20 次独立的实验，并将实验结果得到的最好结果，平均结果，标准方差和计算得到的 gap 值展示在下表 5.9 和表 5.10 中，如下所示：

表 5.9 三种扩展算法在大规模 SDVRP 数据集上的结果比较

Name	Optimal_value	ASGCT		ACGCT		MMGCT	
		Min	Gap (%)	Min	Gap (%)	Min	Gap (%)
kelly01	5646.46	5733.41	1.53	5721.39	1.33	5671.24	0.44
kelly02	8566.04	9066.69	5.84	8998.67	5.05	8832.60	3.11
kelly03	11649.06	**11647.51**	**-0.01**	**11633.56**	**-0.13**	**11551.52**	**-0.84**
kelly04	14639.32	15273.38	4.33	14709.22	0.48	**14479.30**	**-1.09**
Avg. gap(%)	-	-	2.93	-	1.68	-	0.41

表 5.10 三种扩展算法在大规模 SDVRP 数据集上的比较

Name	ASGCT		ACGCT		MMGCT	
	Avg	Std	Avg	Std	Avg	Std
kelly01	5778.96	33.73	5778.01	39.39	5736.00	31.25
kelly02	9179.71	71.88	9110.60	85.31	9025.45	117.72
kelly03	12072.32	171.40	11988.83	167.90	11753.85	106.57
kelly04	15619.42	176.26	15351.09	295.16	14909.72	216.12

表 5.9 中的 Min 表示 20 次实验结果中得到的最短路径长度结果，gap 表示最短路径长度与数据集最优解之间的差距。表 5.10 中的 Avg 表示 20 次试验结果的平均值，std 表示实验结果的标准方差。从表 5.9 可以看出三种扩展算法在 kelly03 数据集上得到的都要优于数据集提供的最优解，除此之外 MMGCT 在 kelly04 上得到的结果也优于数据集提供的最优解，并且根据 gap 计算公式，三个算法与最优解之间的平均 gap 值分别是 2.93%，1.68%和 0.41%。虽然本文设计的三种算法在这个实验中得到的结果并不全部优于最优解，但是 gap 相对较小，因此它们可以很好的解决 SDVRP 问题。此外本文所设计的扩展算法更具有一般性，它们同时也能解决 SDVRP-GCT 问题。

5.3.2 三种扩展算法在大规模 SDVRP-GCT 数据集上实验

在上述实验的基础上，本文进一步评估了这三种扩展算法在四个大规模 SDVRP-GCT 数据集上的性能，与上面数据集转化方法一致，先将 h 的值设为 0.1，数据集上路径上的货物消耗量就等于路径长度与 h 的乘积，然后在每个数据集上进行 20 次实验并将算法得到的最短路径长度，平均路径长度，标准方差，所需车辆数目，运行时间等结果分别展示在表 5.11、表 5.12 和表 5.13 中。

表 5.11 三种扩展算法在大规模 SDVRP-GCT 数据集上的比较

Name	Min_{ASGCT}	Min_{ACGCT}	Min_{MMGCT}
kelly01	6131.12	**5972.07**	5983.21
kelly02	9762.81	9563.83	**9440.57**
kelly03	13089.33	12806.85	**12520.46**
kelly04	16412.48	16196.34	**15711.38**

表 5.12 三种扩展算法在大规模 SDVRP-GCT 数据集上的结果比较

Name	ASGCT		ACGCT		MMGCT	
	Avg	Std	Avg	Std	Avg	Std
kelly01	6184.94	28.88	**6069.62**	45.37	6072.57	48.58
kelly02	9852.42	42.30	9685.41	64.93	**9596.86**	82.61
kelly03	13237.35	75.77	13071.63	115.72	**12717.87**	123.14
kelly04	16795.47	153.29	16615.24	219.82	**16086.33**	201.66

表 5.13 三种扩展算法在大规模 SDVRP-GCT 数据集上的比较

Name	ASGCT		ACGCT		MMGCT	
	NV	T(s)	NV	T(s)	NV	T(s)
kelly01	10	**483.96**	10	496.38	10	505.15
kelly02	11	**892.02**	11	903.74	11	916.93
kelly03	11	**1432.38**	11	1439.28	11	1498.54
kelly04	12	**2075.00**	12	2190.33	12	2174.68

表 5.11 中 Len（Algorithm）表示该算法在每个数据集上得到的最好结果，表 5.12 中 Avg 表示实验结果的平均值，Std 表示实验结果的平均方差，表 5.13 中 NV 表示在该数据集上所需的车辆数目，T 表示算法运行时间。从三个表格可以看到 ASGCT 算法运行时间比 ACGCT 和 MMGCT 都要少，在最短路径长度和平均路径长度上，MMGCT 和 ACGCT 算法得到的路径长度都要小于 ASGCT，这表明 ACGCT 和 MMGCT 的算法性能要优于 ASGCT。然后对比 MMGCT 算法和 ACGCT 算法所得到的最短路径长度，可以看出 MMGCT 的得到的结果在 kelly02、kelly03 和 kelly04 数据集上的最短路径长度都要小于 ACGCT 算法得到的实验结果，除了在 kelly01 数据集上，ACGCT 得到的最短路径长度略微低于 MMGCT，但是总的来说在大规模的 SDVRP-GCT 数据集上，MMGCT 算法表现的性能都要优于 ACGCT 和 ASGCT。综上所述可以得出，在大规模的 SDVRP-GCT 问题上，MMGCT 的表现最优，ACGCT 次之，ASGCT 的结果最差，但是 ASGCT 算法的运行速度最快。

5.4 三种扩展算法在真实数据集上的实验

为了说明本文所提出的 SDVRP-GCT 问题在现实生活中有更广泛的应用，我们基于 2015 年全国数学建模竞赛题目 F（URL：https://www.shumo.com/home/html/3168.html）收集了一个属于 SDVRP-GCT 问题的真实数据集，这个数据集可以描述如下：一位自驾游爱好者计划自己开车游览全国所有 5 A 级旅游景点。由于他居住在陕西省省会西安市，因此本文选择西安市作为唯一的中心站（*depot*）并假定在该省份所需的旅游时间为零。在这个现实的 SDVRP-GCT 问题中，旅行者每年只有四次可外出旅游的假期，每次旅行只有十五天时间(假设每个假期的最大持续时间相同)，这相当于 SDVRP-GCT 问题中车辆的容量 Q=15，顾客的需求是访问每个省的 5A 级景点所需的时间，而路线

消耗是开车到下一个省份所需的时间，且旅行者每次旅行结束后必须返回西安工作。为了对这个问题进行建模，我们从 2015 年全国数学建模大赛题目 F 中获取了省会之间的距离信息和每个 5A 级旅游景区的停留天数。根据题目描述每天行车时间不能超过 8 小时，高速公路上的平均行车速度为 90km/h。我们可以根据省会城市之间的距离信息(距离信息来源于建模大赛题目 F 的附录 3)来近似计算出在路上所花费的时间(四舍五入为半天的时间)。在每个省的旅游时间 (可能有多个 5A 级旅游景点) 等于该省所有 5A 级景点建议游览的天数总和（题目 F 的附录 1 中给出了每个省市 5A 景点的建议游览时间），总的成本等于在路上的行车时间加上游览景区的时间。每个省份所有 5A 景区所需的游览时间如下表 5.14 所示。

表 5. 14 中国 31 个省份的旅游时间

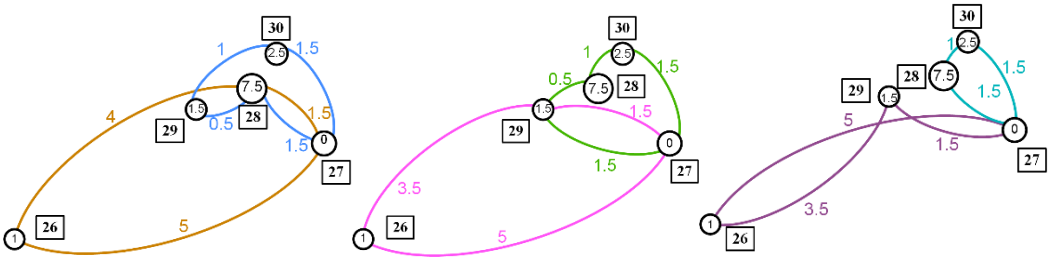
ProvinceID	1	2	3	4	5	6	7	8
Demand(day)	3.5	2	9	5.5	2.5	3.5	4.5	11
ProvinceID	9	10	11	12	13	14	15	16
Demand(day)	2	14	12	9.5	11.5	8.5	8	10
ProvinceID	17	18	19	20	21	22	23	24
Demand(day)	8.5	12	8	3.5	3.5	7.5	10	4
ProvinceID	25	26	27	28	29	30	31	-
Demand(day)	10	1	4	7.5	1.5	2.5	22.5	-

本文使用三种扩展的蚁群算法在该真实数据集上进行了实验。三个算法分别进行了 20 次独立实验，得到的最短旅行天数如表 5.15 所示。在表 5.15 中可以看到，MMGCT 算法得到的旅行天数最短，需要 145.5 天，ACGCT 算法得到的旅行天数次之，需要 148 天，ASGCT 得到的旅行天数最长，需要 148.5 天。三种旅行路线要想游览完所有的景点都需要 25 个假期，因为他每年只有四个假期，所有他需要 6 年半的时间才能游览完全国所有的 5A 级景点。

表 5. 15 三种扩展算法在真实的 SDVRP-GCT 数据集上的结果对比

ASGCT			**ACGCT**			**MMGCT**		
NV	Len(day)	T(s)	NV	Len(day)	T(s)	NV	Len(day)	T(s)
25	148.5	11.65	25	148	14.81	25	**145.5**	13.17

为了进一步研究实验结果，本文提供了三种算法在我国某一区域内的最优出行路线的可视化展示。如图 5.4 中区域 A 所示，区域 A 包括了省份 ID 分别为 26、27、28、29 和 30 的五个省，其中 ID 为 27 的省份为陕西省(省会西安为 depot)，其他省份所需出行时间如表 5.14 所示。本文将 ASGCT、ACGCT 和 MMGCT 得到的图 5.4 中区域 A 的出行路线可视化分别展示在图 5.3(a)、图 5.3(b)和图 5.3 (c)中。从图 5.3 中可以看到 ASGCT 获得的出行路线为:27 - 30 - 29 - 28 - 27,27 - 26 - 28 - 27；ACGCT 获得的出行路线为:27 - 30 - 28 - 29 - 27,27 - 29 - 26 – 27；MMGCT 得到的出行路线为:27 - 28 - 30 - 27 ,27 - 29 - 26 - 27。



（a） (b) (c)

图 5.3 三种扩展算法在区域 A 上得到的旅游路线可视化

三种算法得到的路线表明，要游览完区域 A 都需要两次出行。这些省份内的访问总天数(不包括省与省之间的驾车时间)为 1 + 7.5 + 1.5 + 2.5 = 12.5。R[algorithm]表示三种算法在路上行驶需要花费的时间，三种扩展算法穿越 A 区域所需的总天数分别为：R[ASGCT] = 1.5 + 1 + 0.5 + 1.5 + 5 + 4 + 1.5 = 15，R[ASGCT] = 1.5 + 1 + 0.5 + 1.5 + 1.5 + 3.5 + 5 = 14.5，R[ASGCT] =1.5 + 1 + 1.5 + 3.5 + 5 = 14。从时间计算可以看出 MMGCT 在区域 A 所需的旅行时间最短，ACGCT 次之，ASGCT 所需时间最长。我们将 MMGCT 所规划的旅游路线可视化展示在图 5.4 中。

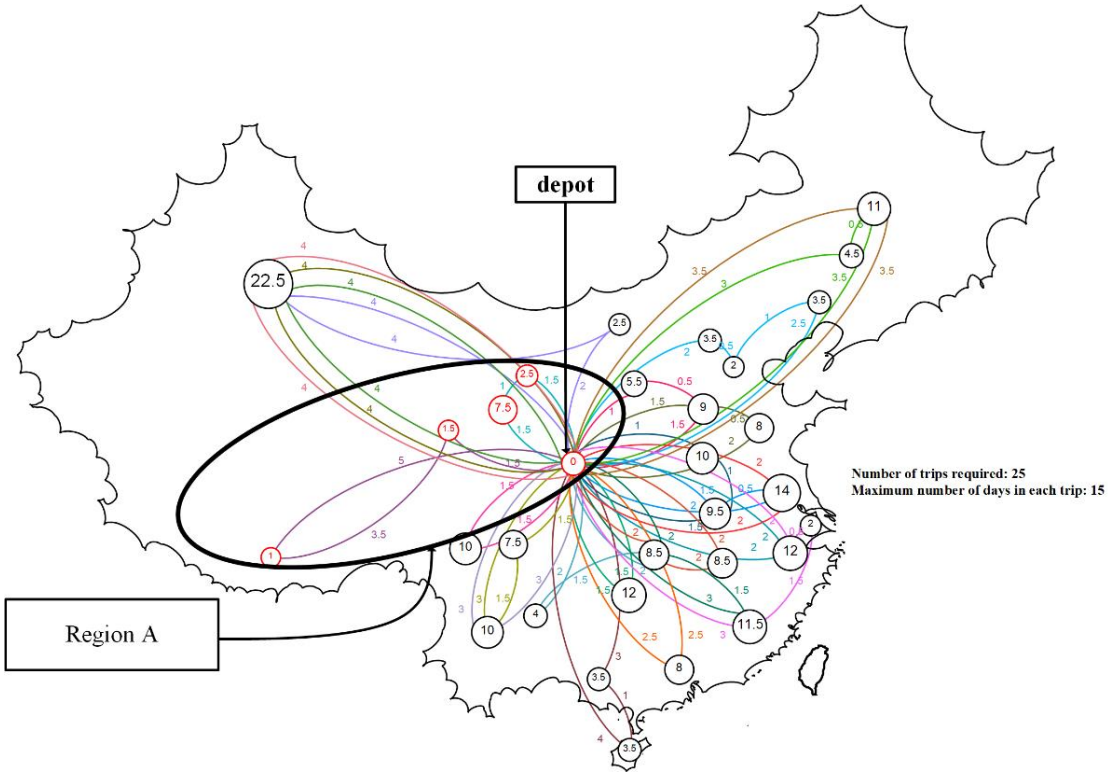


图 5.4 MMGCT 算法在真实数据集上得到的旅游路线可视化

5.5 实验结论

实验结果表明我们提出的扩展算法更具有一般性，不仅能在一些 SDVRP 基准数据集上找到比数据集提供的最优解更好的结果，而且能有效的求解 SDVRP-GCT 问题。虽然 ASGCT 的性能比其他两种算法差，但是该算法速度快，所需的运行时间较少，ACGCT 在顾客需求较高的数据集中，算法性能要优于 MMGCT，而在需求较低的数据集中，MMGCT 的算法性能不仅优于 ACGCT，并且在大规模的 SDVRP-GCT 数据集中也要优于 ACGCT。

5.6 本章小结

三种扩展的蚁群算法首先在十八个 SDVRP 基准数据集上进行了实验，并在其中一些数据集上得到了比当前文献中最优解更好的结果，证明了三种扩展算法能够有效的解决 SDVRP 问题。由于本文提出的 SDVRP-GCT 是一个新问题，目前没有直接可用的数据集和求解算法，因此本文假设路径消耗与路径距离成比例，将十八个 SDVRP 基准数据集转化为 SDVRP-GCT 数据集，然后利用文中设计的三种扩展算法在转化后的 SDVRP-GCT 数据集上进行实验并且得出实验结果。随

后通过对比分析，本文发现三种算法有不同的适用场景。ASGCT 算法的运行速度最快，ACGCT 适用于顾客需求较大的数据集，而 MMGCT 在顾客需求较小的数据集和大规模的数据集上性能更好。最后本文基于数学建模题目收集了真实世界中的 SDVRP-GCT 数据集，然后适用三种扩展算法分别进行旅游路线规划并得出了有效的旅行路线，在很大程度上节省了旅行者的时间。

第 6 章 论文总结与展望

6.1 论文工作总结

本文的研究背景是以 SDVRP 问题为基础，考虑了在实际物流运输过程中车辆在运输途中也需要消耗物资，提出了未被研究过的带有路径消耗的 SDVRP 问题并基于蚁群算法提出了扩展算法进行解决。本文主要进行的研究工作如下：

首先，我们阅读了大量的国内外文献，全面分析了相关学者所研究的 SDVRP 问题以及与其相关的各类变形问题，深入思考找到更符合现实物流配送问题的研究方向。在传统的 SDVRP 问题的基础之上考虑了车辆在运输途中的消耗，并且介绍了 SDVRP-GCT 问题的定义，数学模型以及该问题属于 NP 难问题的原因。

然后，本文分析了相关研究领域的专家们在求解 SDVRP 问题时用到的求解算法的优缺点。针对带有路径消耗的 SDVRP 问题，本文设计了三种扩展的蚁群算法来求解 SDVRP-GCT 问题，并且通过分析实验结果，得出了每种扩展算法所适合的情形以及该算法的优缺点。具体来说，本文将其应用于不同规模的 SDVRP 基准数据集、相应扩展的 SDVRP-GCT 数据集和一个自收集的真实世界数据集上，通过分析结果得出 ASGCT 算法的运行时间最短，ACGCT 算法适合于顾客需求量较高的数据集，而 MMGCT 在顾客需求量较少和规模较大的数据集上性能更好。

本文旨在将车辆运输途中的消耗引入到传统的 SDVRP 问题中，首先通过对 SDVRP-GCT 问题建模，证明了新问题的数学意义，然后设计相应的求解算法并将扩展算法应用在不同规模的数据集上，实验结果表明本文提出的扩展算法能很好的解决 SDVRP 和 SDVRP-GCT 问题。本文的研究希望引起更多学者对 SDVRP 问题的关注并引入更多有效的求解算法。

6.2 未来展望

本文证明了 SDVRP-GCT 问题的现实意义，以及三种扩展的蚁群算法在解决 SDVRP 和 SDVRP-GCT 问题时的有效性。但目前的研究仍存在一些不足，在未来的研究工作中可以从以下两个方面做出改进：

（1）问题模型方面

本文只考虑了车辆在路径上的消耗这一单一因素，然而在实际的物流配送任务会受到多重因素的影响，比如交通堵塞，时间窗，随机顾客需求等等，因此在问题建模的时候需要考虑到更多的影响因素，构建出更符合真实物流配送情形的问题模型。

（2）求解算法方面

本文是在经典的蚁群算法的基础之上，提出了三种扩展的蚁群算法，还有很多其他算法在求解 SDVRP 问题时有很好的性能。在未来的研究工作中可以着重于两个方面：第一是结合其它启发式算法，比如将遗传算法，禁忌算法等与蚁群算法相结合来求解 SDVRP 和 SDVRP-GCT 问题；第二是设计出更多基于精确算法的扩展算法，比如可以通过改进分支限界法，切平面法等等来求解 SDVRP 和 SDVRP-GCT 问题。

参考文献

[1] 中国物流与采购联合会，中国物流信息中心. 2019 年上半年物流运行通报. 2019.

[2] 卞晨,赵建东.车辆路径问题的发展及其应用[J].电脑知识与技术,2016,12(26):79-80+90.

[3] Dantzig G B, Ramser J H. The Truck Dispatching Problem[J]. Management Science, 1959, 6(1):80-91.

[4] Dror M, Trudeau P. Savings by Split Delivery Routing[J]. Transportation Science, 1989, 23(2):141-145.

[5] Cooke K L, Halsey E. The Shortest Route Through a Network with Time-Dependent Internodal Transit Times[J]. Journal of Mathematical Analysis and Applications, 1966, 14(3):493-498.

[6] Lecluyse C, Sörensen K, Peremans H. A network-consistent time-dependent travel time layer for routing optimization problems[J]. European Journal of Operational Research, 2013, 226(3):395-413.

[7] Sophie N. Parragh, Karl F. Doerner, Richard F. Hartl. A survey on pickup and delivery problems: Part I: Transportation between customers and depot[J]. Journal Fur Betriebswirtschaft, 2008, 58(1):21-51.

[8] Wilson H, Weissberg H. Advanced dial-a-ride algorithms research project: final report[J]. Technical Report, 1967, R76-20.

[9] Tillman, Frank A. The Multiple Terminal Delivery Problem with Probabilistic Demands[J]. Transportation Science, 1969, 3(3):192-204.

[10] Renaud J, Laporte G, Boctor F F. A tabu search heuristic for the multi-depot vehicle routing problem[J]. Computers & Operations Research, 1996, 23(3):229-235.

[11] Psaraftis H N. A Dynamic Programming Solution to the Single Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem[J]. Transportation Science, 1980, 14(2):130-154.

[12] Solomon, Marius M. Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints[J]. Operations Research, 1987, 35(2):254-265.

[13] Kallehauge B , Larsen J , Madsen O B G , et al. Vehicle Routing Problem with Time Windows[M]// Column Generation. Springer US, 2005.

[14] Kallehauge B. Formulations and exact algorithms for the vehicle routing problem with time windows[J]. Computers and Operations Research, 2008, 35(7):2307-2330.

[15] Archetti C, Savelsbergh M W P, Speranza M G. Worst-Case Analysis for Split Delivery Vehicle Routing Problems[J]. Transportation Science, 2006, 40(2):226-234.

[16] Sariklis D, Powell S. A Heuristic Method for the Open Vehicle Routing Problem[J]. Journal of the Operational Research Society, 2000, 51(5).

[17] Repoussis P P, Tarantilis C D, Ioannou G. The open vehicle routing problem with time windows[J]. Journal of the Operational Research Society, 2007, 58(3):355-367.

[18] Li X, Leung S C H, Tian P. A multistart adaptive memory-based tabu search algorithm for the heterogeneous fixed fleet open vehicle routing problem[J]. Expert Systems with Applications, 2012, 39(1):365-374.

[19] Perboli G, Tadei R, Vigo D. The Two-Echelon Capacitated Vehicle Routing Problem: Models and Math-Based Heuristics[J]. Transportation Science, 2009, 45(3):364-380.

[20] Erdogan S, Miller-Hooks E. A Green Vehicle Routing Problem[J]. Transportation Research Part E Logistics & Transportation Review, 2012, 48(1):0-114.

[21] Ho S C, Haugland D. A tabu search heuristic for the vehicle routing problem with time windows and split deliveries[J]. Computers & Operations Research, 2004, 31(12):1947-1964.

[22] 马彧妍. 最大最小蚁群算法求解 SDVRP 和 SDWVRP 问题的研究[D]. 2014.

[23] Ray S, Soeanu A, Berger J, et al. The multi-depot split-delivery vehicle routing problem: Model and solution algorithm[J]. Knowledge-Based Systems. 2014,71:238-265.

[24] Archetti C, Campbell A M, Speranza M G. Multicommodity vs. Single-Commodity Routing[J]. Transportation Science, 2016, 50(2):461-472.

[25] Han F W, Chu Y C. A multi-start heuristic approach for the split-delivery vehicle routing problem with minimum delivery amounts[J]. Transportation Research Part E: Logistics and Transportation Review, 2016, 88:11-31.

[26] Li X, Yuan M, Chen D, et al. A Data-Driven Three-Layer Algorithm for Split Delivery Vehicle Routing Problem with 3D Container Loading Constraint[C]//24th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD). 2018:528-536.

[27] Chung S H, Li Y. Disaster relief routing under uncertainty: A robust optimization approach[J]. IISE Transactions. 2018:1-18.

[28] Belenguer J M, Martinez M C, Mota E. A Lower Bound for the Split Delivery Vehicle Routing Problem[J]. Operations Research, 2000, 48(5):801-810.

[29] Jin M, Liu K, Bowden R O. A two-stage algorithm with valid inequalities for the split delivery vehicle routing problem[J]. International Journal of Production Economics, 2007, 105(1):228-242.

[30] Archetti C, Bianchessi N, Speranza M G. Branch-and-cut algorithms for the split delivery vehicle routing problem[J]. European Journal of Operational Research, 2014, 238(3):685-698.

[31] 邓士杰，支建庄，于贵波，等. 基于模拟退火算法的 TSP 问题研究[J]. 价值工程. 2012,(28):290-291.

[32] 李向阳. 遗传算法求解 VRP 问题[J]. 计算机工程与设计. 2004,(02):271-273+276.

[33] 李志萍，高兴国. 模拟退火法与禁忌搜索法求解 VRP 的对比分析[J]. 科技信息. 2010,(22):79-80.

[34] 苏涛，韩庆田，李文强，等. VRP 问题蚁群算法研究[J]. 计算机与现代化. 2012,(11):18-21+25.

[35] Zhan Z H, Zhang J, Li Y, et al. Adaptive Particle Swarm Optimization[J]. IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics. 2009,39,(6):1362-1381.

[36] Saeed Y , Ahmed K , Zareei M , et al. In-Vehicle Cognitive Route Decision using Fuzzy Modeling and Artificial Neural Network[J]. IEEE Access, 2019:1-1.

[37] Archetti C, Mansini R, Speranza M G. Complexity and Reducibility of the Skip Delivery Problem[J]. Transportation Science, 2005, 39.

[38] Archetti C, Feillet D, Gendreau M, et al. Complexity of the VRP and SDVRP[J]. Transportation Research Part C, 2011, 19(5):741-750.

[39] 蒋毅. 基于蚁群优化算法的车辆路径问题研究[D].吉林大学,2007.

[40] Dorigo M, Maniezzo V , Colorni A . Ant system: optimization by a colony of cooperating agents[J]. ieee transactions on systems man & cybernetics part b cybernetics a publication of the ieee systems man & cybernetics society, 1996, 26(1):29-41.

[41] 赵云涛，王胜勇，卢家斌，等. 蚁群优化算法及其理论进展[J]. 科技创新导报. 2012,(10):32-33.

[42] 倪庆剑，邢汉承，张志政，等. 蚁群算法及其应用研究进展[J]. 计算机应用与软件. 2008,(08):12-16.

[43] 基于 ASRank 和 MMAS 的蚁群算法求解飞机指派问题[J]. 管理工程学报. 2012,(02):148-155.

[44] Dorigo M, Gambardella L M. Ant colony system: a cooperative learning approach to the traveling salesman problem[J]. IEEE Transactions on Evolutionary Computation, 1997, 1(1):53-66.

[45] Stutzle T, Hoos H. MAX-MIN Ant System and local search for the traveling salesman problem[C]// IEEE International Conference on Evolutionary Computation (ICEC '97). IEEE, 1997.

致　谢

时光匆匆飞逝，我依稀还记得 2013 年，我与父母初次踏入吉林大学的校园的场景，仿佛就是眨眼之间我已经写完了硕士毕业论文，即将开始一段新的征程。这段记载着我青春的日子，值得我一生铭记于心。在这段学生生涯中，给过我无数帮助与关怀的人们，我永远心怀感激。

首先，在这里我想要感谢我的导师给我提供了很好的学习环境与学术氛围，锻炼我们的学术思维能力和人际交往能力，感谢他在学习中给我的研究指明了方向，在生活中给予我最及时的帮助。

其次，我想要感谢教研室的其他老师们，他们教会我人生要不停的奋斗与努力，绝不轻言放弃，我永远不会忘记所有老师的指导与帮助。感谢我的同学们在工作与学习中给我的陪伴与鼓励，与我一起分享我的快乐与忧愁。

然后，我想对我的家人表示深深的感谢，感谢父母为我提供温暖的港湾和扬帆起航的勇气，感谢我的家人在我的背后永远支持着我，你们的支持和鼓励让我无惧困难，不断前进。

最后，我想向评阅本论文的老师们致以真挚的感谢！