运筹学大作业

Author: 常毅成

2025-01-16

目录

1 引言 **1**

2 问题描述 **1**

3 模型建模 **1**

3.1 集合与索引 1
3.2 参数定义 1
3.3 决策变量 2
3.4 目标函数 2
3.5 约束条件 2
3.5.1 中转站租赁容积 2
3.5.2 车辆数量限制 2
3.5.3 客户需求满足 2
3.5.4 库存平衡 2
3.5.5 运输与车辆路径关联 2
3.5.6 车辆路径逻辑 2
3.5.7 未完成订单非负 2
3.5.8 车辆容量限制 2
3.5.9 客户一次只能由一辆车服务 3
3.5.10中转站分组约束 3

4 方法与实现 **3**

4.1 数据准备 3
4.2 模型建立 3
4.2.1 决策变量定义 3
4.2.2 目标函数实现 3
4.2.3 约束条件实现 4

5 结果与可视化 **5**

6 结论 **6**

企业产品运输车辆调度优化任务

常毅成 22354010

摘　要： 本报告针对货运车辆中转运输调度问题，构建了一个线性规划模型，旨在通过优化车辆调度策略以最小化运输和未完成订单的惩罚等的总成本。采用 **Gurobi** 优化器进行求解，模型考虑了仓库、中转站及客户之间的运输路径、车辆容量限制、客户需求满足以及中转站租赁容积等多种约束条件。通过对模型的求解与分析，提出了优化的运输调度方案，并利用可视化工具对结果进行了展示与解读。本次实验让我自身收益良多，切身感受到了整数规划模型建立和运筹学的魅力。

关键词： 运输调度问题、Gurobi、多种约束条件、可视化

1 引言

在现代物流与供应链管理中，运输调度问题是影响企业运营效率和成本控制的关键因素之一。尤其是在多中转站、多客户的复杂网络环境下，如何优化运输路径和车辆调度，以达到成本最小化和服务水平最大化，是研究的热点。传统的运输调度方法往往依赖于经验和启发式算法，难以在大规模和复杂环境下获得最优解。随着优化算法和计算能力的提升，利用数学模型和优化软件如 Gurobi 来解决运输调度问题成为可能。

本报告旨在通过构建数学优化模型，并利用 Gurobi 优化器求解，研究货运车辆在仓库、中转站与客户之间的调度策略。模型将考虑运输成本、车辆容量、客户需求以及中转站租赁等因素，通过优化调度策略，最小化总成本并满足客户需求。报告将详细介绍问题的背景、模型的建立、Gurobi 求解过程及结果分析，最终提出优化的运输调度方案。

2 问题描述

本研究的问题背景涉及一个仓库、两个中转站及 16 个客户的运输网络。具体问题为如何在多个时间周期内，合理安排货物从仓库通过中转站运输到各客户，既满足客户的需求，又控制运输成本和未完成订单的惩罚成本。主要考虑的因素包括：

- 仓库、中转站与客户之间的运输成本和时间。
- 中转站的租赁容积及其费用。
- 车辆的容量和数量限制。
- 客户需求的满足情况以及未完成订单的惩罚成本。

通过建立数学模型，求解上述问题，旨在优化整体运输调度策略，提升物流效率，降低运营成本。

3 模型建模

3.1 集合与索引

- $N_f = \{0\}$：仓库节点。
- $N_s = \{1, 2\}$：中转站节点。
- $N_p = \{3, 4, \ldots, 18\}$：客户节点。
- $T = \{1, 2, \ldots, T_{\text{max}}\}$：时间周期集合。

3.2 参数定义

- $RentCost$：中转站单位容积租赁费用。
- $RentSize$：中转站租赁容积。
- $HoardCost$：单位货物在中转站的囤积成本。
- $TravelCost1[i]$：仓库到中转站 i 的运输成本。
- $TravelCost2[j]$：仓库到客户 j 的运输成本。
- $DispatchCost[i, j]$：中转站 i 到客户 j 的车辆使用成本。
- $TimeoutCost$：客户超时订单的惩罚成本。
- Q：车辆最大容量。
- $K[i]$：中转站 i 的车辆总数量。
- M：大 M 值，用于条件约束。
- $demand[t, j]$：客户 j 在时间周期 t 的需求量。

3.3 决策变量

- $y1[i,t]$：仓库到中转站 i 在时间 t 运输的货物量（连续变量）。
- $y2[j,t]$：仓库直接到客户 j 在时间 t 运输的货物量（连续变量）。
- $y3[j,t]$：中转站到客户 j 在时间 t 运输的货物量（连续变量）。
- $x[i,j,t]$：车辆是否从中转站 i 到客户 j 在时间 t（二进制变量）。
- $l[j,t]$：客户 j 在时间 t 的未完成订单量（连续变量）。
- $h[i,t]$：中转站 i 在时间 t 的库存量（连续变量）。
- $w[i]$：中转站 i 的租赁容积是否被租赁（0-1 二进制变量）。

3.4 目标函数

目标函数旨在最小化总成本，包括租赁成本、囤积成本、运输成本、车辆使用成本以及未完成订单的惩罚成本。数学表达式如下：

$$\begin{aligned}\text{Minimize} \quad & \sum_{i\in N_s} RentCost \times RentSize \times w[i] \\ & + \sum_{i\in N_s}\sum_{t\in T} HoardCost \times h[i,t] \\ & + \sum_{i\in N_s}\sum_{t\in T} TravelCost1[i] \times y1[i,t] \\ & + \sum_{j\in N_p}\sum_{t\in T} TravelCost2[j] \times y2[j,t] \\ & + \sum_{i\in N_s}\sum_{j\in N_p}\sum_{t\in T} DispatchCost[i,j] \times x[i,j,t] \\ & + \sum_{j\in N_p}\sum_{t\in T} TimeoutCost \times l[j,t]\end{aligned} \tag{1}$$

3.5 约束条件

3.5.1 中转站租赁容积

确保每个中转站的运输量不超过其租赁容积。

$$\sum_{t\in T} y1[i,t] \le RentSize \times w[i], \quad \forall i \in N_s \tag{2}$$

3.5.2 车辆数量限制

每个中转站的车辆使用不超过其总数量。

$$\sum_{j\in N_p}\sum_{t\in T} x[i,j,t] \le K[i], \quad \forall i \in N_s \tag{3}$$

3.5.3 客户需求满足

确保每个客户在每个时间周期内的需求被满足或记录为未完成订单。

$$y2[j,t]+y3[j,t]+l[j,t] \ge demand[t,j-3], \quad \forall j \in N_p, \forall t \in T \tag{4}$$

3.5.4 库存平衡

维护中转站的库存平衡。

$$h[i,t] = \begin{cases} y1[i,t] - \sum_{j\in N_p} y3[j,t], & t = 1 \\ h[i,t-1] + y1[i,t] - \sum_{j\in N_p} y3[j,t], & t > 1 \end{cases}, \quad \forall i \in N_s, \forall t \in T \tag{5}$$

3.5.5 运输与车辆路径关联

确保运输量与车辆路径的一致性。

$$y3[j,t] \le M \times x[i,j,t], \quad \forall i \in N_s, \forall j \in N_p, \forall t \in T \tag{6}$$

3.5.6 车辆路径逻辑

每辆车在一个时间点只能执行一次运输任务，且不超过中转站的车辆总数量。

$$\sum_{j\in N_p} x[i,j,t] \le K[i], \quad \forall i \in N_s, \forall t \in T \tag{7}$$

3.5.7 未完成订单非负

确保未完成订单量非负。

$$l[j,t] \ge 0, \quad \forall j \in N_p, \forall t \in T \tag{8}$$

3.5.8 车辆容量限制

确保运输量不超过车辆的最大容量。

$$y3[j,t] \leq Q \times x[i,j,t], \quad \forall i \in N_s, \forall j \in N_p, \forall t \in T \tag{9}$$

3.5.9 客户一次只能由一辆车服务

每个客户在每个时间点最多由一辆车服务。

$$\sum_{i \in N_s} x[i,j,t] \leq 1, \quad \forall j \in N_p, \forall t \in T \tag{10}$$

3.5.10 中转站分组约束

确保每个中转站仅服务其负责的客户组。中转站 1 负责客户 3-10，中转站 2 负责客户 11-18。

$$x[i,j,t] = 0, \quad \forall i \in N_s, \forall j \notin \text{ResponsibleGroup}(i), \forall t \in T \tag{11}$$

4 方法与实现

为了解决上述车辆调度优化问题，本研究采用了 Gurobi 优化器，通过 Python 编程实现模型的构建和求解。Gurobi 作为一种高效的数学优化软件，能够处理线性规划、整数规划等多种优化问题，适用于本研究中的车辆调度问题。

4.1 数据准备

首先，读取需求数据并进行初始化，包括定义仓库、中转站、客户及时间周期等集合。同时，设置运输成本、车辆容量、车辆数量以及其他相关参数。

4.2 模型建立

利用 Gurobi 的 Python 接口，定义决策变量、目标函数及约束条件。

4.2.1 决策变量定义

在 Gurobi 中，决策变量通过 ‘model.addVars‘函数进行定义。具体变量如下：

- $y1[i,t]$：仓库到中转站 i 在时间 t 运输的货物量（连续变量）。
- $y2[j,t]$：仓库直接到客户 j 在时间 t 运输的货物量（连续变量）。
- $y3[j,t]$：中转站到客户 j 在时间 t 运输的货物量（连续变量）。
- $x[i,j,t]$：车辆是否从中转站 i 到客户 j 在时间 t（二进制变量）。
- $l[j,t]$：客户 j 在时间 t 的未完成订单量（连续变量）。
- $h[i,t]$：中转站 i 在时间 t 的库存量（连续变量）。
- $w[i]$：中转站 i 的租赁容积是否被租赁（0-1 变量）。

4.2.2 目标函数实现

目标函数通过 Gurobi 的 ‘quicksum‘函数进行线性组合，实现总成本的最小化。具体实现代码如下：

Listing 1: 目标函数定义

```
# 1. 中转站租赁成本
rent_cost = quicksum(RentCost *
    RentSize for i in N_s for t in T)

# 2. 中转站存货成本
hoard_cost = quicksum(HoardCost * k[i,
    t] for i in N_s for t in T)

# 3. 仓库到中转站运输成本（火车）
transport_cost1 =
    quicksum(TravelCost1[N_s.index(i)]
    * y1[i, t]
                    for i in N_s for
                        t in T)

# 4. 仓库直接到客户运输成本（货车）
transport_cost2 =
    quicksum(TravelCost2[N_p.index(j)]
    * y2[j, t]
                    for j in N_p for
                        t in T)

# 5. 车辆使用成本（货车从中转站到客户）
# 使用 node_to_index 映射 i 和 j 到
    DispathCost 的索引
vehicle_transport_cost = quicksum(
   DispathCost[node_to_index[i],
       node_to_index[j]] * x[i, j, t]
   for i in N_s for j in
       customer_groups[i] for t in T
```

```
)

# 6. 客户需求的超时成本
timeout_cost = quicksum(TimeoutCost *
    l[j, t] for j in N_p for t in T)

#
    目标函数为总成本的负数（因为要最大化负成本即最小化
model.setObjective(
    -(rent_cost +
     hoard_cost +
     transport_cost1 +
     transport_cost2 +
     vehicle_transport_cost +
     timeout_cost),
    GRB.MAXIMIZE
)
```

4.2.3 约束条件实现

每个约束条件通过循环遍历相关的集合，并使用‘model.addConstr‘方法逐一添加至模型中。以下为部分关键约束的代码示例：

Listing 2: 关键约束定义

```
# 1. 初始化约束
# 初始时刻所有中转站库存为0
for i in N_s:
    model.addConstr(k[i, 0] == 0,
        f"Initial_Inventory_{i}")

# 2. 车辆路径约束

# 2.1
    每个中转站在每个时间段最多派出K[i]辆车
for i in N_s:
    for t in T:
        model.addConstr(
            quicksum(x[i, j, t] for j in
                customer_groups[i]) <=
                K[i],
            f"Vehicle_Limit_{i}_{t}"
        )

# 2.2 每个客户在每个时间段最多被一辆车服务
for j in N_p:
    for t in T:
        model.addConstr(
            quicksum(x[i, j, t] for i in
                N_s) <= 1,
            f"One_Vehicle_Per_Customer_{j}_{t}"
        )

# 3. 货物量约束

# 3.1 中转站库存不超过租赁容积
for i in N_s:
    for t in T:
        model.addConstr(
            k[i, t] <= RentSize,
            f"Inventory_Capacity_{i}_{t}"
        )

# 3.2 客户需求满足
for j in N_p:
    for t in T:
        model.addConstr(
            y2[j, t] + y3[j, t] + l[j, t]
                >= r_i_t[j, t],
            f"Demand_Constraint_{j}_{t}"
        )

# 3.3 车辆容量限制
for i in N_s:
    for j in customer_groups[i]:
        for t in T:
            model.addConstr(
                y3[j, t] <= Q * x[i, j, t],
                f"Vehicle_Capacity_{i}_{j}_{t}"
            )

# 4. 状态更新约束

for t in T:
    for i in N_s:
        if t < timeInterval - 1:
            # 中转站库存更新
            model.addConstr(
                k[i, t + 1] == k[i, t] +
                    y1[i, t] -
                    quicksum(y3[j, t] for
```

```
                    j in
                    customer_groups[i]),
                f"Inventory_Update_{i}_{t}"
            )
    for j in N_p:
        if t < timeInterval - 1:
            # 未完成订单更新
            model.addConstr(
                l[j, t + 1] == l[j, t] +
                    r_i_t[j, t] - y2[j,
                    t] - y3[j, t],
                f"Unmet_Order_Update_{j}_{t}"
            )

# 5. 车辆路径与运输货物的关联约束

for i in N_s:
    for j in customer_groups[i]:
        for t in T:
            #
                如果有车辆从i到j，y3[j,t]可以大于0
            #
                如果没有车辆从i到j，y3[j,t]必须为0
            model.addConstr(
                y3[j, t] <= Q * x[i, j, t],
                f"Y3_Capacity_Link_{i}_{j}_{t}"
            )
```

这些约束条件确保了模型在满足运输需求的同时，遵守车辆和中转站的容量限制，并维护中转站的库存平衡。

5 结果与可视化

模型求解后，输出了各中转站的租赁情况、运输计划以及未完成订单情况。通过可视化图表，可以直观地分析运输调度的效果。以下对主要结果进行详细讨论。

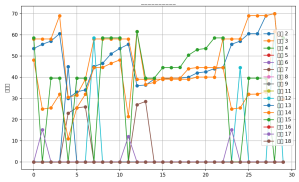

图 1: 每个客户运输量分析

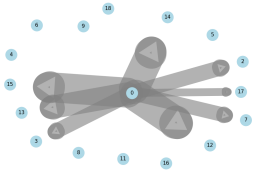

图 2: 仓库运输客户情况

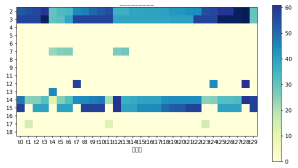

图 3: 仓库运输客户的热力图

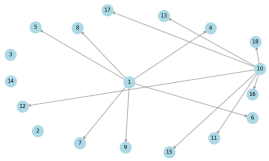

图 4: 中转站运输客户情况

通过上述图表的展示，可以全面了解运输情况的分布。

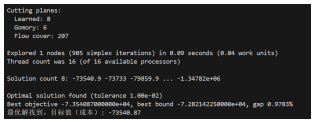

图 5: 最终结果

```
从仓库到中转站的运输量 y1:
中转站 1, 时间 0: 321.0
中转站 1, 时间 1: 321.0
中转站 1, 时间 2: 321.0
中转站 1, 时间 3: 387.0
中转站 1, 时间 4: 141.4
中转站 1, 时间 5: 143.5
中转站 1, 时间 6: 149.25
中转站 1, 时间 7: 321.0
中转站 1, 时间 8: 321.0
中转站 1, 时间 9: 321.0
中转站 1, 时间 10: 321.0
中转站 1, 时间 11: 321.0
中转站 1, 时间 12: 151.5
中转站 1, 时间 13: 152.0
中转站 1, 时间 14: 183.43
中转站 1, 时间 15: 188.9
```

图 6: 仓库到中转站运输情况结果

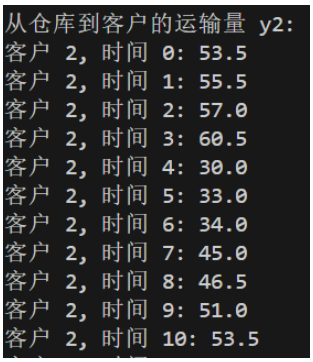

图 7: 仓库到客户运输结果部分显示

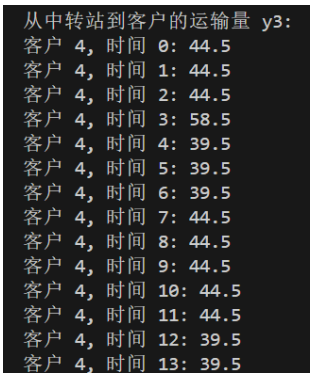

图 8: 中转站到客户运输结果部分显示

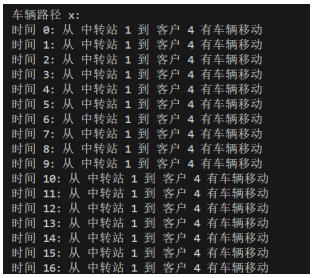

图 9: 车辆路径 x

6 结论

通过建立货运车辆中转运输调度的优化模型，本文成功求解了在多中转站、多客户和多个时间周期下的最优运输策略。模型在满足客户需求的同时，有效控制了运输成本和未完成订单的惩罚成本。使用 Gurobi 优化器，模型得到了高效的求解，结果表明优化调度显著提升了运输效率，减少了资源浪费。

具体而言，模型优化了中转站的租赁决策，合理分配了车辆资源，确保了高需求客户的运输需求得到满足，同时最小化了运输和囤积成本。通过可视化分析，识别了运输网络中的热点和瓶颈，为进一步优化运输网络布局提供了依据。

未来的研究可以进一步扩展模型，考虑运输时间的动态变化、库存管理的优化以及更多现实约束条件的引入，如运输时间窗、车辆类型多样性等，以提升模型的实用性和适用范围。此外，可以结合实时数据和动态调度算法，开发更加灵活和响应迅速的运输调度系统。

本次报告即使花了自己很多心思去完成，但可能仍有不足，希望老师谅解并指正。最后非常感谢金老师和姜老师一学期以来的精心授课，带我入门了运筹学。也感谢助教师兄师姐的付出！

参考文献

[1] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*. Available: `https://www.gurobi.com/documentation/`

[2] Leslie Lamport. *LaTeX: A Document Preparation System*. Addison-Wesley, 1986.

[3] Barfod, M., & Cordeau, J.-F. (2021). New developments in vehicle routing problem research: A comprehensive survey. *European Journal of Operational Research*, 295(1), 1-20.

[4] Wang, X., & Tang, J. (2022). Solving the Vehicle Routing Problem with Time Windows Using Gurobi Optimizer. *Journal of Transportation Systems Engineering and Information Technology*, 22(3), 90-100.