南开大学期末结课论文

# 基于城市道路车流量的城市交通红绿灯时长优化模型

**所在学院**:物理科学学院

**论文作者**:0920程序设计基础第十八组

**指导教师**:李妍教授

2024 年 12 月 19 日

0920程序设计基础期末结课论文

# 基于城市道路车流量的城市交通红绿灯时长优化模型

**论文作者**:0920程序设计基础第十八组

**组长**:段鹏宇(学号:2413468)

**组员**:常浩(学号:2411274)
韩雨晨(学号:2412917)
胡炼(学号:2413206)
张博艺(学号:2413718)

(以上排名按照拼音首字母排序,不分先后)

2024 年 12 月 19 日

# 摘要

## 摘要

本文研究背景是基于城市道路车流量的城市交通红绿灯时长优化模型（以下简称：交通优化模型），主要从理论模型的建立和代码的实现两方面，讨论了本课题在研究和开发过程中解决的主要问题以及使用的主要技术。本文首先简单地介绍了课题的研究背景，以及建立该交通优化模型的实际意义，并进一步分析了国内外研究现状。

接着，详细地阐述了生成交通优化模型的理论模型：本文的交通优化模型既对实际问题进行了抽象和适当的简化，又兼顾到诸多影响城市道路车流量的因素（如：道路形状、路口的走向性等）。从而使得交通优化模型不仅能够较好的匹配具体路段车流量情况，而且能够更好满足该模型的实际意义。

同时，本文详细阐述了算法的研究与实现：如何获取与处理数据、如何通过处理庞大数据计算红绿灯时长。

最后，本文总结整个过程中的经验得失，并展望进一步发展前景。

# 关键词

**Keywords**: 城市交通拥堵,红绿灯时长优化,车流量,拥堵指数模型,数据处理与算法

# 目录

# Part I

# 正文

# Chapter 1

# 引言

## 1.1 课题背景介绍

随着经济的飞速发展以及人们生活水平的不断提高,私家车以惊人的速度普及开来。然而,在私家车给人们带来便捷出行体验的同时,一个日益严峻的问题也逐渐凸显出来,那就是城市交通拥堵正逐渐成为人们生活中的普遍问题。

而伴随数字化浪潮来袭,大数据、人工智能、物联网技术蓬勃发展,为交通治理革新带来曙光。海量交通数据实时采集成为可能,摄像头捕捉车流动向、传感器监测道路承载,手机定位追踪人群出行轨迹,积累起珍贵信息富矿。依托这些数据,构建智能交通流量优化模型时机已然成熟。

## 1.2 实际意义

### 1.2.1 缓解交通拥堵,提升出行效率

构建城市交通流量模型,旨在打破交通“僵局”,动态调控信号灯时长、智能规划出行路线、均衡路网车流量分配,用科技手段重塑城市交通流畅脉络。极大提升居民的生活舒适度与工作效率,保障城市经济活动高效运转。

### 1.2.2 节能减排,助力绿色可持续发展

交通拥堵时,车辆频繁怠速、启停,燃油消耗剧增且尾气排放大幅超标。据研究,拥堵状态下汽车油耗比顺畅行驶高出 2 - 3 倍,有害气体排放浓度可达正常行驶的 5 - 10 倍。机动车尾气排放成为空气污染重要源头,对雾霾形成影响显著。

因此,优化交通流量模型的目的还在于减少不必要燃油消耗,氮氧化物、颗粒物等污染物排放随之削减,助力城市空气质量达标天数逐年增加,缓解环保压力,推动城市向低碳绿色转型,助力城市迈向绿色、高效、宜居新征程。

## 1.3 国内外研究进展

### 1.3.1 国内研究成果

国内研究团队利用传感器、监控设备和智能终端等采集交通数据,并进行清洗、整合和存储。同时,国内团队积极致力于建立交通流量优化模型,考虑道路容量限制、信号灯控制等约束条件,通过优化算法求解得到最优方案。

### 1.3.2 国外研究成果

国外研究相对更深入和广泛,涉及城市交通规划、交通政策制定、交通管理等多个方面。此外,国外许多团队也在对模型本身进行创新。欧洲的研究人员提出了基于随机用户均衡(SUE)的交通分配模型,该模型能够更好地反映出行者的路径选择行为,提高了交通流量分配的准确性。一些研究还将交通分配模型与城市规划相结合,为城市交通基础设施的规划和建设提供了科学依据。

# Chapter 2

# 问题的分析与解决

## 2.1 理论模型

先对问题进行抽象简化,然后再进行具体的建模计算各种参量

### 2.1.1 对实际问题的抽象与简化

假设已有了一个道路网,所需要关注的是每一条路上的车流量与路口红绿灯之间的有机关联。

首先对路口进行抽象化,将一条道路分为很多段,每一段都直接连接着两个路口,而道路的尽头处则也看作是一个路口。然后将每个路口视作一个点,路口与路口之间的道路部分则视作一根线段。

于是得到一张无向图$G = (V; E)$,其中,点集$V$代表所有路口的集合,而边集$E$代表两个路口之间道路部分的集合。而道路网中的道路在图$G$就表现为一条路$p$,用集合$P$来表示图$G$中对应着所有道路(并非图$G$的所有路)的集合。而对于某个路口$v$,与其相邻的所有道路部分的集合记为$N(v)$。

不难看出,两个相邻的路口间通常只有一个道路部分; 出于因区分双行道和单行道而导致的数据采集与理论分析困难问题的考虑,以及单条单向道路对整体的影响较小且可忽略不计, 将把所有道路视作双行道,并且双行道的两侧车道的车流量相同,且数值上均与该道路部分的车流量相同。

另外,考虑到道路形状会对结果产生较小的影响,但难以采集和难以估计其影响模式,忽略之。因此与某一条道路$i$相关的量仅考虑其长度$L_i$。

| 现实物体 | 抽象物体 |
|---|---|
| 道路网 | 图$G$ |
| 路口 | 点$v$ |
| 道路部分 | 边$e$ |
| 道路集 | 集合$P$ |

此外,路口具有走向性,为了研究方便,假设道路网由横向道路网和纵向道路网组合形成。于是可以得到一个无交并:$E = E_{row} \sqcup E_{column}$。于是一个路口可以视作四个道路部分的交汇,其中这四个道路部分有两个是横向的,有两个是纵向的。对于那些$N(v) < 4$的路口,不妨虚拟出假想的道路部分与其连接,这些道路部分相关的数据均为0,也就是对结果不产生影响。

为了能更精准的描述道路网中的量,将进行一些简单的数学处理。

对于图$G$中的每一个路口$v \in V$,定义一个单射$id : V \longrightarrow N$,使得每个路口都对应一个唯一的整数,这个整数称为路口$v$的id。用$I_v$来表示所有路口id的集合。类似地,也为每一条道路定义其id,并用$I_p$来表示所有道路id的集合。

结合道路网的走向,横向道路的id集合即为$I_p^{row} = id(E_{row})$,而纵向道路的id集合则为$I_p^{column} = id(E_{column})$。

下面考虑时间的表示,将时间按照5分钟一段进行划分,那么一天的时间则可以划分为288段。方便起见,用集合$J_t = \{1, 2, ..., 288\}$来表示这288段时间段的集合,并记5分钟为一个时间单位$\Delta T$。

现在来抽象化红绿灯模型,对于路口$v$的红绿灯,将其视作一个函数:

$$L_v : J_t \longrightarrow \{0, 1\}$$

当其值为0时,横向为红灯,纵向为绿灯; 当其值为1时,横向为绿灯,纵向为红灯。由于道路网中可能存在大量的路口红绿灯,单独对每一个路口红绿灯作考虑是不现实的,因此引入假设:

$$\forall t \in J_t, v \in V, L_v(t) = L(t)$$

然后用红绿时间比$\eta$来操控这些红绿灯,这里红绿时间比定义为红绿灯在一个周期内红灯时间与绿灯时间的比值。假设一个周期内红灯时间占比为$\alpha$,绿灯时间占比为$\beta$,那么有:

$$\begin{cases} \alpha + \beta = 1 \\ \eta = \frac{\alpha}{\beta} \end{cases}$$

最后,来考虑用于描述车流量的物理量。假定在时间段$j \in J_t$中,道路部分$i$的车流平均速度为$v_{i,j}$,车辆通过总路程为$S_{i,j}$

至此,对现实问题的抽象和简化已经完成,下表汇集了所有在模型建立过程中将会用到的变量:

| 名称 | 变量名 |
|---|---|
| 道路$i$的长度 | $L_i$ |
| 时间集 | $J_t$ |
| 路口id集 | $I_v$ |
| 道路id集 | $I_p$ |
| 红绿时间比 | $\eta$ |
| 车流平均速度 | $v_{i,j}$ |
| 车流通过总路程 | $S_{i,j}$ |

### 2.1.2 建立模型

现在先截取某段时间$j \in J_t$,并先对一个道路部分$i$进行处理。

考虑到车流可以被视为由大量的车辆组成,这些车辆在宏观尺度上也可以近似为连续的介质,因此可以用流体的观点来类比研究车流。于是参考管道流动中流体的哈森-威廉姆斯方程:

$$Q = 0.285CD^{2.63}\left(\frac{\Delta P}{L}\right)^{0.54}$$

因此需要的方程可以看成$D = f(Q, C)$的形式, 其中$D$是一个指数,用于衡量这个路口的拥堵情况, $C$是道路承载量,只和道路自身有关, $Q$是车流流量。

下面对这些参量分别进行推导。

首先是指数$Q$,这里要给出$f$的形状。不妨定义拥堵指数为一个衡量道路拥堵情况的数,并且拥堵指数越大,道路越堵。因此拥堵指数$D$和流量$Q$成正比;而流量$Q$一定时,道路承载量$C$越大,拥堵指数$D$越小。所以可以写出一个初步的式子:

$$D = \lambda\frac{Q^a}{C^b}$$

其中$\lambda, a, b$均为正数。由于最终只需要$D$的相对大小,所以两个次幂$a, b$以及$\lambda$的绝对大小是无需考虑的,不妨在$\lambda = 1, a = 1$的情况下讨论。为了方便计算,在两边取对数,而$\ln D$和$D$的单调性是一致的,可以将其等视之。于是得到了一个更好计算的公式:

$$D = \ln Q - blnC$$

其中$b$是一个用于衡量道路承载量对拥堵指数作用效果的影响因子,其值恒为正。

其次是道路承载量$C$,由于之前所做的理想化假设,道路承载量仅与道路长度有关, 又因为需要的是$C$的相对大小,所以不妨直接定义道路承载量为道路长度,即$C = L$。

最后是车流流量$Q$,因为不会引起歧义,所以在这里省略$v$和$S$的下标。此外假设车流量为$Q$,车辆数量为$N$,并给其编号$1, 2, .., N$,第k辆车的平均速度为$v_k$,第k辆车通

过的路程$S_k$。按照速度和路程的定义,得到:

$$S_k = v_k \Delta T$$

$$v = \frac{1}{N} \sum_{k=1}^{N} v_k$$

于是得到

$$S = \sum_{k=1}^{N} S_k = \sum_{k=1}^{N} v_k \Delta T = N \Delta T v$$

整理得

$$N = \frac{S}{v \Delta T}$$

又由流量的定义

$$Q = \frac{N}{\Delta T}$$

得到最终的计算公式

$$Q = \frac{S}{v \Delta T^2}$$

至此,仿照着哈森-威廉姆斯方程,建立了单个道路部分的道路拥堵指数模型:

$$\begin{cases} D = \ln Q - blnC \\ C = L \\ Q = \frac{S}{v \Delta T^2} \end{cases}$$

下面还是在$j$这个时间段,但是对路口$v$进行考虑。

先在无红绿灯的情况下进行,可以将该路口连接的所有道路部分的道路拥堵指数直接作和, 但考虑到城市中某些较重要道路可能会有更高的使用频率,更容易面临道路交通拥堵情况, 所以对于第$i$条道路定义一个道路重要度$\delta_i$, 于是路口$v$的拥堵指数可以定义为:

$$D_v = \sum_{i \in N(v)} \delta_i D_i$$

其中$D_i$为第$i$条道路的拥堵指数

接着将红绿灯纳入模型,由于红绿灯转换时间远小于$\Delta T$,所以在截取的时间段$j$中,可以忽略红绿灯转换产生的边界效应。为了描述红绿灯带来的交通影响,采用这种描述方式: 绿灯时对拥堵指数不作改变,红灯时使拥堵指数乘上一个常数$\kappa > 1$。

假设在时刻$j \in \{t|L(t) = 0\}$,路口横向为红灯,纵向为绿灯。设横向的道路为$1, 2$,纵向的为$3, 4$,那么此时的道路拥堵指数$D_v^r$可以作如下衡量:

$$D_v^r = \kappa \delta_1 D_1 + \kappa \delta_2 D_2 + \delta_3 D_3 + \delta_4 D_4$$

由对称性,若在时刻$j \in \{t|L(t) = 0\}$,路口横向为绿灯,纵向为红灯,则此时的道路拥

堵指数$D_v^g$可以作如下衡量:

$$D_v^g = \delta_1 D_1 + \delta_2 D_2 + \kappa\delta_3 D_3 + \kappa\delta_4 D_4$$

现在在整个时间段$j$中来看,有:

$$D_v = \alpha^\tau D_v^r + \beta^\tau D_v^g$$

其中$\tau$是一个参数,用于衡量时间对拥堵指数作用的效应。
代入前几个式子,得:

$$D_v = (\alpha^\tau\kappa + \beta^\tau)(\delta_1 D_1 + \delta_2 D_2) + (\alpha^\tau + \beta^\tau\kappa)(\delta_3 D_3 + \delta_4 D_4)$$

于是可以得到,在某个时间段$j$中,某个路口的拥堵指数可以表示为横向拥堵指数和纵向拥堵指数的一个线性组合,这个关系用数学表达式可以进行如下表示:

$$D_v = (\alpha^\tau\kappa + \beta^\tau)D_v^{row} + (\alpha^\tau + \beta^\tau\kappa)D_v^{column}$$

其中$D_v^{row}$为横向道路部分的拥堵指数,$D_v^{column}$为纵向道路部分的拥堵指数

至此得到了某个路口$v$的道路拥堵指数的计算公式。

现在,来考虑整张图$G$的道路拥堵指数的计算公式。同样在时间段$j$进行处理。取所有路口的拥堵指数的和作为该时间段的总拥堵指数,即:

$$D_{All} = \sum_{v\in G} D_v$$

结合前面所得式子,有:

$$D_{All} = \sum_{v\in G}(\alpha^\tau\kappa + \beta^\tau)D_v^{row} + (\alpha^\tau + \beta^\tau\kappa)D_v^{column}$$

$$= (\alpha^\tau\kappa + \beta^\tau)\sum_{v\in G} D_v^{row} + (\alpha^\tau + \beta^\tau\kappa)\sum_{v\in G} D_v^{column}$$

不难看出:

$$\sum_{v\in G} D_v^{row} = 2\sum_{i\in I_p^{row}} D_i$$

对偶地,也有:

$$\sum_{v\in G} D_v^{column} = 2\sum_{i\in I_p^{column}} D_i$$

于是这样便简单的给出了某时间段的拥堵指数的计算方式

最终,考虑一个总的拥堵指数,它表现为全天所有时间段的均值。也就是:

$$\mathcal{D} = \frac{1}{|J_t|} \sum_{j \in J_t} D_{All,j}$$

接下来,只需要编程,用代码实现求解在红绿时间比为何值时,总拥堵指数最小

## 2.2 算法的研究与实现

### 2.2.1 数据的获取与处理

程序所需数据源于深圳的数据开放平台,数据提供方式为数据集与api,其格式为xlsx。但是该数据集无法直接使用,于是需要进行数据处理。这里使用python编写了一个程序用于提取所需数据(见附录A)。

通过pandas库里的read方法, 将道路信息与车流量数据两个文档内的所需数据结合并进行字符串剪切处理, 最后输出了一个文件用于后续计算。所输出的数据格式为'D Id T 时间 S 车的总路程 V 车流速度 L 道路长度'

### 2.2.2 红绿灯时长的计算

在处理大量数据输入时,频繁地创建数组会导致效率低下和资源浪费。为了解决这一问题,采取了一种优化策略,即构建了一个名为“model”的结构体模板。这个模板预先定义了存储空间,用于存放输入数据中的$V$、$S$、$L$、$D$和$Q$等关键参数。通过这种方式不仅减少了内存分配和释放的频率,还提高了数据处理的速度和稳定性。此外,这种模板化的方法也使得代码更加整洁和易于维护,因为它将相关的数据项组织在一起,便于管理和访问。总的来说,这种结构体模板的应用显著提升了数据处理的效率和性能。

在处理数据的过程中常常会遇到数据量极为庞大的情况。在一些简单的运行框中输入这些数据时,会发现操作变得异常复杂,甚至可能导致运行框无法正常处理。为了有效解决这一问题,可以采用文件重定向的方法,也就是freopen函数。这种方法允直接从in文件中读取数据,避免了在运行框中手动输入大量复杂数据的麻烦。通过这样的操作,不仅能够使数据处理过程更加高效,还能减少因数据输入复杂而可能产生的错误,确保整个数据处理流程的顺畅进行。

因为计算时涉及到的参量较多,处理方式不通,采用函数分块来进行各个相关参数的计算,计算的函数功能分为以下部分：

void Quantity_traffic_count() 计算车流量的函数

void Decide_traffic_calculate() 计算所有道路D参数的函数

void minimun_calculate() 最小值计算函数

void text() 读取文件函数

void Input() 数据输入以及处理函数

算法采用了差分算法,将原本连续的函数进行离散处理,看做离散的数值点,从而通过确定步长,起始点与终点来进行每一点的计算。从而近似拟合函数的值,同时在每一步的比较中取较小的值,得到最小值。

# Chapter 3

# 总结与展望

## 3.1 本文总结

### 3.1.1 本文的主要内容

本文聚焦于城市交通拥堵问题，构建基于城市道路车流量的交通红绿灯时长优化模型。首先，鉴于私家车普及引发的交通拥堵现状，以及数字化技术发展为交通治理带来的机遇，阐述了研究的背景和实际意义，包括缓解拥堵、节能减排等方面。接着，详细介绍了国内外在交通流量优化模型方面的研究进展，为后续研究提供参考。

在问题分析与解决部分，通过对实际问题的抽象与简化，构建了无向图来表示道路网，引入相关变量描述道路、路口、红绿灯及车流量等信息。基于此，建立了道路拥堵指数模型，先针对单个道路部分，参考哈森-威廉姆斯方程确定拥堵指数与车流流量、道路承载量的关系，进而推导出车流流量公式；再对路口拥堵指数进行分析，考虑道路重要度及红绿灯影响，得到路口拥堵指数计算公式；最后得出整张图的拥堵指数计算方式，并以全天所有时间段的均值作为总拥堵指数。在算法实现上，介绍了数据获取与处理过程，包括从深圳数据开放平台获取数据并编写程序处理，以及为提高数据处理效率和性能采取的优化策略，如构建结构体模板和使用文件重定向方法；还阐述了采用差分算法计算红绿灯时长，通过函数分块计算相关参数。

### 3.1.2 本文的创新点

1. 模型构建创新：

将交通道路网抽象为无向图，综合考虑多种因素建立道路拥堵指数模型，能更精准地描述交通状况，尤其在处理路口拥堵指数时，引入道路重要度概念，使模型更贴合实际交通场景，为红绿灯时长优化提供了更合理的依据。

2.算法优化创新：

在数据处理方面，通过构建结构体模板减少内存分配与释放频率，提高数据

处理速度和稳定性；使用文件重定向方法避免手动输入大量数据的麻烦，确保数据处理流程顺畅，有效解决了处理大数据时面临的效率和操作复杂问题，提升了整个算法的性能。

## 3.2 今后工作的展望

后续研究可从多方面展开,举例如下：

一方面，进一步优化模型，考虑更多影响交通流量的因素，如不同时段车辆类型分布对道路承载能力的影响、恶劣天气条件下的交通变化等，使模型更加完善和精确；

另一方面，改进算法，探索更高效的计算方法，降低时间复杂度，以适应更大规模交通数据的处理需求。此外，还可将模型与实际交通管理系统相结合，进行实地测试与应用，根据实际反馈不断调整参数，提高模型的实用性和有效性，为城市交通智能化管理提供更有力的支持。

# 参考文献

[1] 胡晓敏,段宇晖,欧炜标,等. 交通流优化膨胀控制遗传规划算法 [J/OL]. 计算机应用研究, 1-7[2024-12-19]. https://doi.org/10.19734/j.issn.1001-3695.2024.05.0177.

[2] 姚荣涵,张晓彤,廉莲. 交叉口可变导向车道控制优化模型 [J]. 吉林大学学报(工学版), 2017, 47 (04): 1048-1054. DOI:10.13229/j.cnki.jdxbgxb201704006.

[3] 翟忠民. 动态交通组织中的信号控制技术 [J]. 中国交通信息产业, 2004, (02): 92-94.

# Part II

# 附录代码

# Chapter 4

# 附录A 处理数据的代码

```
import pandas as pd
SJ = pd.read_excel('SJ.xlsx',sheet_name='Sheet1')
DL = pd.read_excel('DL.xlsx',sheet_name='Sheet2')
def Length(Id,DL):
    matched_row = DL[DL.iloc[:, 0] == Id]
    if not matched_row.empty:
        return matched_row.iloc[0, 6]
    else:
        return None
def pen(row,SJ):
    row += -1
    ID = SJ.iloc[row,7]
    Id = str(SJ.iloc[row,7])
    S = str(SJ.iloc[row,0])
    V = str(SJ.iloc[row,1])
    T = str(SJ.iloc[row,5])
    L = str(Length(ID,DL))
    Str = 'D-'+Id+'-T-'+T+'-S-'+S+'-V-'+V+'-L-'+L+'\n'
    return(Str)
Row = SJ.shape[0]
i = 1
DS = '-'
while i <= Row:
    S = pen(i,SJ)
    DS = DS + S + '-'
    i += 1
with open('data.in','w') as f:
```

```
f.write(DS)
```

# Chapter 5

# 附录B 进行理论计算的代码

```
#include<bits/stdc++.h>
using namespace std;
struct model
{
    double Velocity;
    double Shift;
    double Length_road;
    double Quantity_traffic;
    double Decide_traffic;
};

model D[300][300];
int number,t_number;
int mid=number/2;
const int T=5;
const int Zero=0;
double b;
double D_r=0,D_c=0;
double alpha=0,kappa=2;
double D_sum=0;
double alpha_record=0;
double tau=2;
double t=120;

void Input();
void Quantity_traffic_count();
void Decide_traffic_calculate();
```

```
double SumD_count(int m,int n);
void minimun_calculate();
void output();
void text();

int main()
{
    freopen("data.in","r",stdin);
    freopen("data.out","w",stdout);
    cin>>number>>t_number;
    cin>>b;
    cin>>kappa;
    cin>>tau;
    Input();
//  text();
    Quantity_traffic_count();
    Decide_traffic_calculate();
    D_r=SumD_count(Zero,mid);
    D_c=SumD_count(mid+1,number);
    minimun_calculate();
    output();
    return 0;
}

void text()
{
    for(int i=1;i<=number;i++)
    {
        for(int j=1;j<=t_number;j++)
        {
            cout<<"S-";
            cout<<D[i][j].Shift;
            cout<<"V-";
            cout<<D[i][j].Velocity;
            cout<<"L-";
            cout<<D[i][j].Length_road;
            cout<<endl;
        }
    }
```

```
}

void Input()
{
    for(int i=1;i<=number;i++)
    {
        for(int j=1;j<=t_number;j++)
        {
            char a;//=getchar();
            cin>>a;
            int del1;
            cin>>del1;
            cin>>a;
            int del2;
            cin>>del2;
            cin>>a;
            cin>>D[i][j].Shift;
            cin>>a;
            cin>>D[i][j].Velocity;
            cin>>a;
            cin>>D[i][j].Length_road;
        }
    }
}

void Quantity_traffic_count()
{
    for(int i=1;i<=number;i++)
    {
        for(int j=1;j<=t_number;j++)
        {
            double Intermediate_quantity;
            Intermediate_quantity=D[i][j].Velocity*T*T;
            D[i][j].Quantity_traffic=1.0*D[i][j].Shift/
            Intermediate_quantity;
        }
    }
}
```

```
void Decide_traffic_calculate()
{
    double b;
    for(int i=1;i<=number;i++)
    {
        for(int j=1;j<=t_number;j++)
        {
            D[i][j].Decide_traffic=log(D[i][j].Quantity_traffic)
            -b*log(D[i][j].Length_road);
        }
    }
}

double SumD_count(int m,int n)
{
    double Intermediate_quantity;
    for(int i=m;i<=n;i++)
        for(int j=1;j<=t_number;j++)
            Intermediate_quantity+=D[i][j].Decide_traffic;
    return Intermediate_quantity;
}

void minimun_calculate()
{
    //D=(kappa*D_r+D_c)*(pow(alpha,kappa))+(D_r+kappa*D_c)*
    (pow((1-alpha),kappa));
    double min=(kappa*D_r+D_c);
    for(alpha;alpha<=1;alpha+=0.01)
    {
        D_sum=(kappa*D_r+D_c)*(pow(alpha,tau))+(D_r+kappa*D_c)*
        (pow((1-alpha),tau));
        if(min>=D_sum)
        {
            min=D_sum;
            alpha_record=alpha;
        }
    }
}
```

```
void output()
{
    double T_red=t*alpha_record;
    double T_green=t*(1-alpha_record);
    cout<<T_red<<endl;
    cout<<T_green<<endl;
}
```

# Chapter 6

# 附录C 小组分工表

| 项目 | 负责成员 |
| --- | --- |
| 数据查找与收集 | 韩雨晨,胡炼 |
| 数据清洗与整理 | 韩雨晨,胡炼 |
| PPT制作 | 段鹏宇 |
| 报告展示与答辩 | 段鹏宇,常浩,韩雨晨,胡炼,张博艺 |
| 分析方向选择 | 段鹏宇,常浩,韩雨晨 |
| 道路id匹配 | 段鹏宇,张博艺 |
| 道路模型图绘制 | 张博艺 |
| 模型最佳参数的优度检测 | 常浩,张博艺 |
| 相关研究总结与论文书写 | 常浩,胡炼 |
| 理论模型建立 | 段鹏宇,常浩,胡炼 |
| 编程实现 | 韩雨晨,胡炼,张博艺 |