基于拓扑处理的 Logit 网络加载算法

专　　业：交通信息工程及控制

硕 士 生：辛松歆

指导教师：李　军

摘 要

城市交通网交通分配是城市交通规划的一个重要组成部分，通过交通量分配所获得的路段交通量资料是检验城市交通规划是否合理的主要依据。随机交通分配可以反映出行者对不同路径的认识误差，分析出行者对不同路径的选择概率，从而对出行者的路径选择行为进行分析。Logit 模型和 Probit 模型是两种最重要的随机交通模型，它们的求解往往通过仿真或网络加载算法来确定固定成本下的交通分配状态，然后采用 Sheffi 和 Powell 所建议的逐次平均法来计算。Dial 于 1971 年提出了一种算法，使得 Logit 的分析求解成为可能，而 Probit 目前只能通过仿真来求解。Dial 算法虽然计算效率高，但其对“合理路径”的定义过于严格，导致了分配结果中一些路径阻抗较小的线路没被使用，而路径阻抗较大的线路反倒被使用的不合理现象，限制了 Logit 模型在实际中的应用。

本文对 Dial 算法进行回顾，分析其不足产生的原因，并在此基础上提出了一种基于拓扑处理求解 Logit 型网络加载模型的新算法——TPDial 算法，文中还根据新算法中拓扑处理的次数定义了“Single-pass”和“Double-pass”算法。算法通过拓扑处理删除环路中的特定路段来排除所有包含环路的路径，并根据拓扑排序确定节点计算顺序来计算路段权重和流量。新算法保持 Dial 算法的高效性的同时对合理路径的定义进行了改善，大大降低了 Dial 算法的误差。特别是，对于无环网络的应用，可以得到与理论值完全一致的结果。文中还通过计算实例对不同算法的计算精度和效率进行了比较，计算实例表明新算法可降低 Dial 算

法中合理路径定义过于严格所带来的误差，且其计算的高效性完全可以用于大型网络或动态交通分配的计算。

关键词：交通工程　Logit 型交通分配　Dial 算法　拓扑遍历　最短路径

An algorithm for Logit network loading problem based on topological sorting

Major: Traffic Information Engineering and Control

Name: Xin Songxin

Supervisor: Li Jun

ABSTRACT

Traffic assignment is an important part of traffic planning. The flow of each link calculated by traffic assignment is always used to estimate the rationality of traffic planning. Stochastic assignment models assume that travelers have the same error distribution in the utility term. In the case of stochastic assignment, these errors are theorized to result from perception errors of travelers. For networks with fixed costs, Logit and Probit stochastic user equilibrium models are two of the most important models. The models can be solved through the method of successive averages proposed by Powell and Sheffi, both of them require a process of network loading. Up to now, simulation method remains the most feasible method to load network of Probit model. Dial proposed an algorithm for the Logit network loading problem. The algorithm is very efficient and does not require path enumeration. However, it is found that the definition of "reasonable" paths in Dial's algorithm is so strict that it may lead to some problems.

In this paper Dial's algorithm is reviewed and the related issues are discussed in details. A new algorithm based on topological sorting is presented, which excludes all cycles by removing certain links from loops only when it is necessary. The new algorithm loosens the definition of reasonable paths, improved the order of calculating, so as to reduce the errors of Dial's algorithm while retaining the efficiency of Dial's.

The new algorithm is called "TPDial's algorithm". Both "single-pass" and "double-pass" optimization method are presented according to number of shortest route calculations. Numerical examples show that the new algorithm can reduce errors introduced by the strict definition of "reasonable route" in Dial's algorithm. Specially, its outcome of the network without looping paths is identical with the theory.

Key words: traffic engineering, Logit network loading, Dial's algorithm, topological scan, shortest path

引 言

“万事始于规划”，说明了人们在日常生活中进行规划的重要性，个人、家庭、单位、城市、地区、国家均不例外，有了切实可行的规划，才能促使人们瞄准确定的目标努力。作为社会经济发展基础的交通基础设施也是如此，做好交通规划是合理调整交通结构、均衡交通需求等的重要手段，其原理又是支撑交通规划的理论基础。在我国的城市交通发展历程中，越来越显露出没有合理进行交通规划的问题，造成了目前多数大城市交通问题“头疼医头，脚痛医脚”的被动局面。所以，迫切需要利用科学的手段与方法进行合理的交通规划。人们经过几十年的努力，提炼出了经典的四阶段法，即交通发生、交通分布、交通方式划分和交通分配。

交通分配是交通需求四阶段预测的最后阶段，也是城市交通规划的一个重要组成部分，通过交通量分配所获得的路段交通量资料是检验城市交通规划是否合理的主要依据。最优化理论、图论、计算机技术的发展，为交通分配模型和算法的研究及开发提供了坚实的基础，通过几十年的发展，交通分配成为交通规划诸问题中被国内外学者研究得最深入、取得研究成果最多的内容。

Logit 型随机交通分配模型由于存在 Dial 分析算法，可以针对不同要求进行有效地分析，特别是对出行者的动态选择行为分析极为有用，因而得到了研究者广泛的重视；但 Dial 算法是否存在着不足？其产生不足的原因又是什么？是否有改进的地方？本文力求在保持 Dial 算法高效性的前提下，针对 Dial 算法存在的缺陷，采用拓扑处理的方法，提出一种新的 Logit 网络加载算法，并对算法进行深入讨论研究。

本文分为三部分，第一部分为文章的文献综述部分，包括第 1、2 章，其中第 1 章介绍了交通规划、交通分配的相关概念及常用的交通分配模型和方法，第 2 章介绍了 Logit 模型的理论基础及其加载算法回顾；第二部分为文章的核心，包括第 3、4 章，其中第 3 章重点介绍了基于拓扑处理的 Logit 网络加载算法，第 4 章为基于拓扑处理的 Logit 网络加载算法的分析与证明；第三部分是第 5 章为算法的实现与应用部分；第四部分是第 6 章为总结部分，总结研究成果并提出值得继续深入讨论研究的问题。

第1章 绪 论

纵观社会经济发展历史和交通运输发展历程，可以看出一个普遍的现象[1]：城市的形成与演变取决于交通，城市的发展又促进了交通，交通发展与城市演变相互影响，兴衰与共，是不可分离的有机整体。然而交通运输与社会经济的发展不总是同步的，常常出现两种情形[2]：一是交通运输超前于社会经济的发展。如美国在19世纪的经济尚不很发达时，于19世纪30年代至20世纪20年代大力进行铁路建设，全国铁路总里程达41万公里，之后敏感地洞察到运输需求的变化，又大力发展公路、内河和民航等运输方式。目前美国拥有650万公里公路、4万公里内河航道以及45万公里的民航线。近200年来，美国的交通运输基本上保持超前于国民经济的发展，这无疑是其经济长期快速发展的一个重要因素。另一种情形是交通运输滞后于社会经济发展。在汽车时代开始之初，欧洲一些国家的大城市由于交通设施没能适应汽车这种新的交通工具的发展，出现了严重的交通问题，拥挤的交通冲击着城市居民的出行和生活。目前许多发展中国家，如我国，无论是跨地区的大交通，还是城市内的交通，都面临问题成堆的局面，交通成为各级政府和广大人民最关心和头痛的问题，已成为制约国民经济发展的主要瓶颈。

1.1 交通规划

所谓城市交通规划，是指为城市居民的交通行为提供合适的交通设施，改善以致优化城市交通条件，并创造良好的城市环境[1]。它是有计划地引导交通的一系列行动，即规划者如何提示各种目标，又如何将提示的目标付诸实施的方法。

1.1.1 交通规划的发展历程

交通规划理论的产生和发展大致可分作4个阶段[2]。

(1) 萌芽阶段。交通问题和交通建设几乎从人类文明开始就出现了，在汽车出现以前和汽车出现之初，交通建设比较简单，但仍然存在交通规划的过程。如中国一些古代城市的道路网就表现成棋盘形状，这明显是在建路之前有关人员作

了全盘的计划和打算，也就是作了规划。二次大战末期，发达国家汽车逐渐增多，已有的城市路网的容量上和形态上与汽车的出行需求之间的矛盾逐渐表现出来。美国就于 1944 年进行针对交通出行的家庭访问调查，并作了数据统计分析，这是人类有史以来第一次交通调查。20 世纪 50 年代，美国还借用系统分析方法对城市道路网的布局进行了分析。但这些都只是局部的尝试性的探索，作为系统的交通规划理论尚未形成，这是交通规划理论的萌芽阶段。

(2) 四步法阶段。以 1962 年美国芝加哥市发表的《Chicago Area Transportation Study》为标志，交通规划理论正式诞生。1962 年美国制定的联邦公路法规定凡 5 万人口以上城市，必须制定以城市综合交通调查为基础的都市圈交通规划，方可得到联邦政府的公路建设财政补贴，这项法律直接促成交通规划理论的形成和发展。开始，交通预测只是关于交通发生、交通分布、交通分配三个阶段的预测。20 世纪 60 年代后期，日本广岛都市圈的交通规划首次提出了对不同交通方式进行划分这一新的预测内容。此后，交通规划变成了交通发生、交通分布、交通方式划分和交通分配四个步骤，这就是交通规划的四步法（也叫四阶段法）理论。

(3) 非集计模型（**Disaggregate model**）阶段。关于非集计模型的研究最早也是始于 20 世纪 60 年代后期（前面提到的广岛都市圈的交通规划），首先用于交通方式划分，20 世纪 70 年代后，McFadden 等学者对它作了深入的研究，并推向实用化。四步法是将个人的交通分区为单位的模型，而非集计模型的分析单位是个人，对调查得到的数据不进行统计处理，而是引入效用理论、概率论的方法进行直接分析研究的。非集计模型至今仍在发展中。

(4) 均衡模型加计算机技术阶段。自 1975 年 LeBlanc 发明 Beckmann 均衡交通分配模型（1956）的算法以来，人们借助各种现代的应用数学工具（神经网络方法、数学规划方法等等）展开了关于均衡问题的数学模型及其算法研究，迅速发展的计算机技术使得大规模的，复杂的非线性数学规划模型及其算法得以能够实现。这方面的研究一直是 20 多年来交通规划的主要研究内容，而且其触角已经延伸到交通规划以外的领域，如智能交通系统（Intelligent Transportation System，ITS）。

1.1.2 交通规划步骤

整个交通规划的工作结构如图 1-1 所示，包含 8 个工作阶段[2]：①目标确定、②组织工作、③数据调查、④相关基本模型分析、⑤分析预测、⑥方案设计、⑦方案评价、⑧方案实施过程中的信息反馈和修改。在这 8 项工作中，交通调查、交通预测、方案设计、方案评价这四项工作是交通规划的主要内容，而其中交通预测（交通发生、交通分布和方式划分）和交通分配又是传统交通规划理论的重中之重，俗称“四阶段法”。

- 交通规划
  - 确定目标
  - 组织工作
  - 交通调查
  - 社会活动模型（人口、用地、经济等）
  - 分析预测
    - 生成量预测（产生量预测、吸引量预测）
    - 分布预测
    - 方式划分
  - 方案设计
    - 定性分析
    - 定量分析

    （以交通分配为基础）
  - 方案评价
    - 网络技术评价
    - 经济效益评价
    - 社会环境评价
  - 反馈修改

图 1-1 交通规划工作结构图

1.2 交通分配

1.2.1 交通分配的发展概况

城市交通网络交通分配是城市交通规划的一个重要环节。所谓交通分配就是把各种出行方式的空间 OD 量分配到具体的交通网络上，通过交通分配所获得的路段、交叉口交通量资料，是检验道路规划网络是否合理的主要依据[3]。交通分配是交通需求预测“四阶段法”中的一个阶段，由于交通分配过程是以网络为基础的（在现状路网或规划路网上进行），因此，这一阶段的工作往往是在路网规

划方案设计时进行。

人们当初进行交通流分配的研究时，多采用全无全有的最短路径方法，该方法处理的是非常理想化的城市交通网络，即假设网络上没有交通拥挤，路阻是固定不变的，一个 OD 对间的流量都分配在“一条路径”，即最短路径上。随着实际应用和理论研究的深入，研究人员发现该最短路径方法对于城市之间非拥挤公路网的规划设计过程中的交通流分配是比较合适的，但对于既有的城市内部拥挤的交通网络，该方法的结果与网络实际情况出入甚大。实际网络中，路网上存在着较严重的拥挤，路阻是随着交通流量的增加而递增的，出行的流量会在“多条路径”中权衡选择。所以在 1952 年，著名交通问题专家 Wardrop 提出了网络均衡分配的第一、第二定理，人们开始采用系统分析方法和均衡分析方法来研究交通拥挤时的交通流分配，带来了交通流分配理论的一次大的飞跃。首先，人们进行了确定性的分配研究，其前提是假设出行者能够精确计算出每条路径的阻抗，从而能做出完全正确的选择决定，且每个出行者的计算能力和水平是相同的。可见确定性分配反映了网络的拥挤特性，反映了路阻随流量变化的实际，该方法是一次理论的进步。但是，进一步研究实际网络中出行者的出行行为发现，现实中出行者对路段阻抗的掌握只能估计而得。因为出行者的计算能力和水平是各异的，对同一路段不同出行者的估计值不会完全相同。所以，在 1977 年，对交通流分配理论研究最积极、活跃的美国加州大学伯克利分校的 Daganzo 教授及麻省理工的 Sheffi 教授提出了随机性分配的理论，其前提是认为出行者对路段阻抗的估计值与实际值之间的差别是一个随机变量，出行者会在“多条路径”中选择，同一起讫点的流量会通过不同的路径到达目的地。随机性理论和方法的提出，在拟合、反映现实交通网络实际的进程中又推进了一大步。

然而，随着近年来交通拥挤的进一步加重和拥挤在时间和空间范围上的扩大以及智能交通系统研究的进展，人们在由注意新路网的规划设计逐步转向重视既有路网的管路控制的进程中，更加意识到：路网上的拥挤性、路径选择的随机性、交通需求的动态性是同时存在并交互作用的，其机理是纷繁复杂的。确定性分配能够较好的反映网络的拥挤性，随机性分配能够较好地反映出行选择行为的随机性，但是要真正地符合路网实际情况，还有更重要更基本的交通需求的时变性需要反映出来。也就是说，需要一种交通流分配方法能够将路网上交通流的拥挤性、

路径选择的随机性、交通需求的时变性综合集成地刻画反映出来，这正是研究交通问题的人们一直积极探索的问题。

1.2.2 交通分配的基本概念

(1) 交通阻抗

交通阻抗（或者称为路阻）是交通分配中经常提到的概念，也是一项重要指标，它直接影响到交通路径的选择和流量的分配。道路阻抗在交通分配中可以通过路阻函数来描述。所谓路阻函数是指路段行驶时间与路段交通负荷，交叉口延误与交叉口负荷之间的关系。在具体分配过程中，由路段行驶时间及交叉口延误共同组成出行交通阻抗。

交通网络上的路阻，应包含反映交通时间、交通安全、交通成本、舒适程度、便捷性和准时性等等许多因素。根据这些因素建立一个科学严密、解释性强的函数模型是非常困难的。经过大量的理论分析和工程实践，人们得出影响路阻的主要因素是时间，因此交通时间常常被作为计算路阻的主要标准。交通阻抗由路段上的阻抗和节点处的阻抗两部分组成。

① 路段阻抗。在诸多交通阻抗因素中，时间因素是最主要的。对于单种交通网络，出行者在进行路径选择时，一般都是以时间最短为目标。有些交通网络，路段上的行驶时间与距离成正比，与路段上的流量无关，如城市轨道交通网。有些交通网络，如公路网、城市道路网，路段上的行驶时间与距离不一定成正比，而与路段上的交通流量有关，此时就选择时间作为阻抗。这类行驶时间与距离、流量的关系比较复杂，这种关系可以广义地表达为：

$$C_a = f(\{V\}) \tag{1-1}$$

即路段a上费用C_a不仅仅是路段本身流量的函数，而且是整个路网上流量V的函数。这个一般化的公式在城市道路网上是比较多见，因为交叉口的存在，不同路段上的流量会相互影响。

对于公路网而言，由于路段比较长，这一关系可以进一步简化，因为大部分时间花费在路段上而不是在交叉口上，这时式（1-1）可以写成：

$$C_a = f(V_a) \tag{1-2}$$

即路段 a 的费用只与该路段的流量 V_a 及其特性相关，这个假定简化了对路段函数的建立和标定，以及交通分配模型的开发。

对于公路行驶时间函数的研究，既有通过实测数据进行回归分析的，也有进行理论研究的。其中被广泛应用的是由美国公路局（Bureau of Public Road，BPR）开发的函数，被称为 BPR 函数，形式为：

$$t_a = t_0\left[1+\alpha\left(\frac{q_a}{c_a}\right)^{\beta}\right] \tag{1-3}$$

其中：t_a—— 路段 a 上的阻抗；

t_0—— 零流阻抗，即路段上为空静状态时车辆自由行驶所需要的时间；

q_a—— 路段 a 上的交通量；

c_a—— 路段 a 上的实际通行能力，即单位时间内路段实际可通过的车辆数；

α、β—— 阻滞系数，在美国公路局交通分配程序种，α、β 参数的取值分别为 0.15 和 4，也可由实际数据用回归分析求得。

由式（1-3）可知，行驶时间是路段流量的单调递增函数。

② 节点阻抗。节点阻抗指车辆在交通网络节点处主要指在交叉口处的阻抗。交叉口阻抗与交叉口的形式、信号控制系统的配时，交叉口的通过能力等因素有关。在城市交通网络的实际出行时间中，除路段行驶时间外，交叉口延误占有很大的比重，特别是在交通高峰期间，交叉口拥挤阻塞比较严重时，交叉口延误可能会超过路段行驶时间。

交通工程学中，对信号交叉口的延误有过大量的研究，直接目的是为信号控制交叉口的配时，点控、线控和面控系统的设计以及交叉口通过能力的计算而进行。节点处的阻抗可分为两类：

第一类为不分流向类：在某个节点各流向的阻抗基本相同，或者没有明显的规律性的分流向差别。对这类问题比较好处理，用一个统一的值 D_i 表示车辆在节点 i 的延误。

第二类为分流向类：不同流向的阻抗不同，且一般服从某种规律。城市道路网就是这样，车辆在城市道路的交叉口一般有三个流向：直行、左转、右转，所延误的时间差别明显，且一般服从规律：右转<直行<左转。其实，车辆在城市间公路网的节点处也存在同样的延误规律，但是公路网的路段长，车辆在节点处的延误相对于路段上的行驶时间非常小，可以近似为0，这样就可以将之归于上述的“不分流向类”对待。但是，城市道路网交叉口密集，相邻交叉口之间的路段往往只有几百米，车辆在交叉口某些流向的延误时间接近甚至超过路段上的行驶时间，故不可忽略，而且必须分流向计算。

分流向计算时，一般D_{ij}表示来自节点i的车辆在交叉口j的延误，其可以用Webster延误公式表示。

1958年英国TRRL研究所（Transport and Road Research Laboratory，TRRL）的F.V.Webster等人根据排队论理论，提出了一个计算交叉口延误的模型。该模型中主要包括两部分，一部分是车辆到达率为固定均值时产生的正常相位延误即均匀延误，另一部分是车辆到达率随机波动时所产生的附加延误。其具体形式为：

$$t_w=\frac{T(1-\lambda)^2}{2(1-\lambda X)}+\frac{X^2}{2Q(1-X)}-0.65\left(\frac{T}{Q^2}\right)^{\frac{1}{3}}X^{(2+5\lambda)} \tag{1-4}$$

式中：T—— 信号周期长度

λ—— 进口道有效绿灯时间与信号周期长度之比，即绿信比；

Q—— 进口道的交通流量；

X—— 饱和度，$X=Q/S$，S为进口道通过能力。

上式是Webster在Monter-Carlo模拟结果的基础上，校正了当初由排队论理论推导出的交通工程师们常用的延误公式的形式。

人们在实际应用Webster延误公式中发现，当进口饱和度较小时，该公式计算结果比较合理，但是当进口饱和度较大时，如当饱和度趋于1时，求得的延误趋向于无穷大。即饱和度越接近于1，求得的延误越不正确，更无法计算过饱和情况下的延误。一般认为Webster公式适用范围为饱和度的取值在0～0.67之间，即当$0\le X\le 0.67$时，Webster公式计算的结果才是合适的，当饱和度超过这个范围时，公式则不适用。

Webster 公式的提出，对交叉口延误的计算起到了很大的推动作用，但是由于该模型在饱和度上的局限，使得该模型很难直接应用于拥挤的交通网络，即饱和度较大的网络。所以在实际应用中，许多理论研究者或交通工程师对模型进行了不同的修正，派生出了不同类型的改进公式。

⑵ 路段与路径

① 路段。交通网络上相邻两个节点之间的交通线路称为“路段”。

② 路径。交通网络上任意一 OD 点对之间，从发生点到吸引点一串连通的路段的有序排列叫做这一 OD 点对之间的路径。一 OD 点对之间可以有多条路径。

③ 最短路径。一 OD 点对之间的路径中总阻抗最小的路径叫“最短路径”。

1.3 交通分配模型的分类

对于交通分配，国内外均进行过较多的研究，数学规划方法、图论方法及计算机技术的发展，为合理的交通分配模型的研制及应用提供了坚实的基础。交通分配模型可分为均衡模型与非均衡模型两大类，并以 Wardrop 第一、第二定理为划分依据[4~5]。Wardrop 第一原理指出，在道路网的利用者都知道网络状态并试图选择最短路径时，网络会达到这样一种均衡状态，所有使用的路线都比没有使用的路线费用（花费的时间）小；第二原理认为，车辆在网络的分布，使得网络上所有车辆的总出行时间最小[6]。如果交通分配模型满足 Wardrop 第一、第二原理，则该模型为均衡模型。并且，满足第一原理的称为用户（优化）均衡模型（User Equilibrium，UE），满足第二原理的称为系统最优（均衡）模型（System Optimization，SO）。如果分配模型不使用 Wardrop 原理，而是采用模拟方法，则被称为非均衡模型。

均衡分配原理在理论上结构严谨，思路明确；但其数学规划模型维数太多，约束条件也多，且为非线性规划问题。关于非线性数学规划问题的算法设计在数学上也是一个高难度问题。1956 年 Beckmann 等提出了描述这个均衡问题的一个数学规划模型[7]。经过了 20 年之后，即 1975 年才由 LeBlanc 等学者将 Frank-Wolfe 算法用于求解 Bechmann 模型的算法[8]，从而形成了现在的实用解法。Wardrop 原理——Bechmann 模型——LeBlanc 算法这三点突破是交通分配问题研究的三

个里程碑，也是现在交通分配理论的基础。以前由于受到计算机速度及内存空间的限制，很少在大型网络分析的实际工程中应用。

相比之下，非均衡模型具有结构简单、概念明确，计算简便等优点，近 20 年来在实际工程中，尤其是大型网络的交通分析中得到了广泛的应用，特别是有许多模型根据中国的具体交通状况进行修正后，分配结果更加合理。所以非均衡交通分配方法已经在中国的城市交通分析、区域公路网络分析中广泛应用，效果良好[9]。

最近几年，随着计算机技术的快速发展，使得交通分配实用方法也有了很大改进，如大部分非均衡模型向着平衡模型靠拢，即在模型建立和分配算法中尽量接近 Wardrop 原理；同样，为了使均衡模型能达到实用的目的，大部分均衡模型的求解不再采用解析法，而是利用计算机的数值计算快速的优点求近似解。因此，目前的交通分配模型已经很难严格区分均衡模型和非均衡模型，学术界与工程界已经不再去苛求二类模型的定义与标准，而只强调交通分配模型的适用性及其分配精度。

1.3.1 非均衡交通分配

非均衡分配方法按其分配手段可分为单级和迭代分配方法，按分配形态可分为单路径与多路径两类，概括起来如表 1-1 所示。

表 1-1 非均衡分配方法

分配手段 / 形态	单级分配方法	迭代分配方法
单路径型	全无全有分配	容量限制加载分配
多路径型	静态多路径分配	多路径容量限制加载分配

(1) 最短路交通分配方法

最短路交通分配是一种静态的交通分配方法，在该分配方法中，取交通阻抗为常数，即假设车辆的平均行驶车速不受交通负荷的影响。每一 OD 点对的 OD 量被全部分配在连接该 OD 点对的最短线路上，其他道路上分配不到交通量。这种分配方法的优点是计算相当简便，其缺点是出行量分布不均匀，出行量全部集中在最短路上。这种分配方法是其他各种交通分配方法的基础。

(2) 容量限制加载分配方法

容量限制加载分配方法是一种动态的交通分配方法，它考虑了交通阻抗与交通负荷之间的关系，即考虑了公路通行能力的限制，比较符合实际情况。采用这种方法时，需先将 OD 表中的每一 OD 量分解成 K 部分，即将原 OD 表分解成 K 个 OD 分表，然后分 K 次用最短路分配模型分配 OD 量，每次分配一个 OD 分表，并且每分配一次，交通阻抗用修正函数修正一次，直到把 K 个 OD 分表全部分配在网络上。

(3) 多路径交通分配方法

与单路径（最短路）分配方法相比，多路径分配方法的优点是克服了单路径分配中流量全部集中在最短路上这一不合理现象，使各条可能的出行路线均分配到交通量，各出行线路长度的不同决定了它所分配到的流量的大小。阻抗为常数的多路径分配方法有两种模型：Logit 模型和 Probit 模型。

(4) 多路径容量限制加载分配

尽管多路径分配模型能同时考虑最短路因素及随机因素，其分配结果比较合理，但由于模型中没有考虑交通阻抗与交通负荷的关系及通行能力的限制，在拥挤网络上的分配仍有较大误差，在多路径容量限制加载分配方法中，充分考虑了这些因素，使分配模型的适应性更加广泛。与容量限制加载分配方法一样，采用多路径容量限制加载分配方法时，需先将 OD 表中的每一 OD 量分解成 K 部分，即将原 OD 表分解成 K 个 OD 分表，然后分 K 次用最短路分配模型分配 OD 量，每次分配一个 OD 分表，并且每分配一次，交通阻抗用修正函数修正一次，直到把 K 个 OD 分表全部分配在网络上。所不同的是，容量限制加载分配方法每次分配采用最短路分配模型，而在多路径容量限制加载分配方法中，每次分配采用静态的多路径分配模型。

1.3.2 均衡交通分配

(1) 用户均衡分配模型

1952 年 Wardrop 提出用户均衡分配原理之后，曾经在很长一段时间内没有一种严格的模型可求出满足这种均衡准则的交通分配方法，这也自然成了交通流分配研究者的重要课题。1956 年 Beckmann 等学者提出了一种满足 Wardrop 准则的数学规划模型，正是这一数学规划模型奠定了交通分配问题的理论基础。后来的一些分配模型，如弹性需求分配模型、组合分配模型等都是在 Beckmann 模型的基础上扩展得到的。

下面简单地介绍一下 Beckmann 交通均衡分配模型。

首先，均衡分配过程中应该满足交通流守恒的条件，即 OD 间各条路径上的交通量之和应等于 OD 交通总量。根据上述定义的变量和参数，用公式可以表示为：

$$\sum_{k \in W_{rs}} f_k^{rs} = q_{rs} \quad \forall r, s \tag{1-5}$$

式中，f_k^{rs}——出发地为 r，目的地为 s 的 OD 间的第 k 条路径上的流量；

q_{rs}——出发地为 r，目的地为 s 之间的 OD 交通量；

W_{rs}——出发地为 r，目的地为 s 之间的所有路径的集合。

其次，路径交通量 f_k^{rs} 和路段 a 上的交通量 x_a 之间应该满足如下的条件，即路段上的流量应该是由各个 (r,s) 对的途经该路段的路径的流量累加而成，公式表示为：

$$x_a = \sum_r \sum_s \sum_k f_k^{rs} \delta_{a,k}^{rs} \quad \forall a \in L \quad \forall r \in R \quad \forall s \in S \quad \forall k \in W_{rs} \tag{1-6}$$

式中，$\delta_{a,k}^{rs}$——路段—路径相关变量，即 0-1 变量，如果路段 a 属于从出发地为 r，目的地为 s 的 OD 间的第 k 条路径，则 $\delta_{a,k}^{rs}=1$，否则 $\delta_{a,k}^{rs}=0$；

L ——网络中路段的集合；

R ——网络中出发地的集合；

S ——网络中目的地的集合。

同时，路径的总阻抗和路段的阻抗之间应该满足如下的条件，即路径的阻抗应该是该路径的各个路段的阻抗的累加，公式表示为：

$$c_k^{rs} = \sum_a t_a(x_a)\delta_{a,k}^{rs} \quad \forall a \in L \quad \forall r \in R \quad \forall s \in S \quad \forall k \in W_{rs} \tag{1-7}$$

式中，t_a——路段 a 的交通阻抗；

$t_a(x_a)$——路段 a 以流量为自变量的阻抗函数。

最后，路径流量应该满足非负约束，即 $f_k^{rs} \geq 0 \quad \forall k, r, s$。

Beckmann 把上述条件作为基本约束条件，用取目标函数极小值的方法来求解均衡分配问题，提出的交通均衡分配模型如下：

$$\min: \ Z(X) = \sum_a \int_0^{x_a} t_a(\omega)d\omega$$

$$s.t. \begin{cases} \sum_k f_k^{rs} = q_{rs} \\ f_k^{rs} \geq 0 \end{cases} \tag{1-8}$$

其中，$x_a = \sum_r \sum_s \sum_k f_k^{rs}\delta_{a,k}^{rs}$

可以通过数学推导证明该模型与 Wardrop 用户均衡原理是一致的，具体可以参考文献[10]。

Beckmann在1956年提出的上述数学规划模型沉睡了20年之后，即直到1875年才由 LeBlanc 等学者将 Frank-Wolfe 算法用于求解 Beckmann 模型，最终形成了目前广泛应用的一种解法，通常称为 F-W 解法。

Beckmann 模型是一个非线性规划模型，而对非线性规划模型即使现在也没有普遍通用的解法，只是对某些特殊的模型才有可靠的解法，Beckmann 模型就是一种特殊的非线性规划模型。

F-W 方法的前提是模型的约束条件必须都是线性的。该方法是用线性规划逐步逼近非线性规划的方法，它是一种迭代法。在每步迭代中，先找到目标函数一个最速下降方向，然后再找到一个最优步长，在最速下降方向上截取最优步长得到下一步迭代的起点，重复迭代直到找到最优解为止。概括而言，该方法的基本思路就是根据一个线性规划的最优解而确定下一步的迭代方向，然后根据目标函数的一维极值问题求最优迭代步长。

(2) 系统最优分配模型

Wardrop 还同时提出了第二原理即系统最优分配问题。系统最优分配的定义是在拥挤的网络中，交通量应该按照使得路网中总阻抗即总行驶时间最小的原则进行分配。

系统最优原理比较容易用数学模型来表述，其目标函数是网络中所有用户总的阻抗最小，约束条件和用户均衡分配模型一样。

因此，系统最优分配模型是：

$$\min:\ \tilde{Z}(X)=\sum_a x_a t_a(x_a)$$

$$s.t.\ \begin{cases}\sum_k f_k^{rs}=q_{rs}\\ f_k^{rs}\geq 0\end{cases} \tag{1-9}$$

其中，$x_a=\sum_r\sum_s\sum_k f_k^{rs}\delta_{a,k}^{rs}$

总结而言，该模型称为系统最优（均衡）模型（System Optimization），简写作 SO。相应地，Beckmann 模型称为用户（最优）均衡模型（User Equilibrium），简写作 UE。

(3) 系统最优分配和用户均衡分配的关系

对阻抗函数进行变换，令：

$$\tilde{t}_a(x_a)=t_a(x_a)+x_a\frac{\mathrm{d}t_a(x_a)}{\mathrm{d}x_a} \tag{1-10}$$

则：
$$\int_0^{x_a}\tilde{t}_a(\omega)\mathrm{d}\omega=\int_0^{x_a}[t_a(\omega)+\omega\frac{\mathrm{d}t_a(\omega)}{\mathrm{d}\omega}]\mathrm{d}\omega=\int_0^{x_a}[t_a(\omega)\mathrm{d}\omega+\omega\mathrm{d}t_a(\omega)]$$

$$=\int_0^{x_a}\mathrm{d}[t_a(\omega)\omega]=x_a t_a(x_a) \tag{1-11}$$

因此，如果用$\tilde{t}_a(x_a)$作为阻抗函数，则此时用户最优分配模型完全可以转换为系统最优分配模型，所以进行该阻抗函数下的用户最优分配，得到的解就是系统最优分配的解。也就是说，对阻抗函数进行变换后，可以按照用户最优模型的算法来求解系统最优模型。

从一定意义上讲，第一原理更能真实地反映交通网络中用户的实际选择出行路径的行为，基于第一原理的 Beckmann 模型和其 F-W 算法得出的结果也更能符

合交通网络的实际分配结果；而第二原理反映的则是交通系统管理者的主观愿望，一般情况下它与交通网络的实际分配情况存在差异，但是它可以作为对系统评价的指标，为管理者提供一种决策依据。从此意义上说，第二原理是道路系统管理者所希望的分配原则，尤其在智能交通系统获得广泛应用之后。

1.3.3 随机分配方法

网络均衡问题实质上就是一个分配的问题，在分配过程中逐步达到均衡[10]。道路利用者总是力图选择从起点到终点之间阻抗最小的路径，但并不意味着所有利用者都会选择同一路径。因为路段阻抗随交通量变化而变化，结果路径阻抗也因交通量分布不同而变化。只有当不存在道路利用者能单方面改变其路径来降低其行驶时间时，一个稳定状态才算达到了，这就是所谓的“用户均衡（User Equilibrium）”即 UE 问题。它是一个确定性交通流分配问题，即认为道路利用者能够精确计算每条路径的真实阻抗并做出完全正确的择路决策。

实际中，道路利用者对路段阻抗只能是一种对真正阻抗的估计，这种估计值与实际值之间的差别是一个随机变量，相应地就有随机用户均衡的问题，即任何一个道路利用者均不可能通过单方面改变其路径来降低其所估计的行驶时间时，一个稳定平衡状态才算达到了，这就是“随机用户均衡（Stochastic User Equilibrium）”即 SUE 问题。它是一个随机性交通流分配问题，分配中路径选择仍然遵循 Wardrop 第一原理，差别在于道路利用者选择的是自己估计阻抗最小的路径而已，所以同一 OD 对之间有多条路径被选择。

当不考虑拥挤效应的时候，随机均衡模型就简化成非均衡随机模型。而非均衡随机模型在引入交通阻抗用修正函数修正，通过迭代运算，其结果也向着平衡模型靠拢。另一方面，为了使均衡模型能达到实用的目的，大部分均衡模型的求解不再采用解析法，而是利用计算机的数值计算快速的优点求近似解。对于 SUE 问题主要利用 F-W 算法的思想，采用求解无约束极小值问题的最速下降方法来计算，而在每次迭代中仍采用的是非均衡随机分配方法，所以非均衡随机分配方法仍是求解随机均衡模型的基础。

随机交通分配，也称为多路径分配，可以反映出行者对不同路径的认识误差，分析出行者对不同路径的选择概率，从而对出行者的路径选择行为进行分析。阻

抗为常数的多路径分配方法有两种模型：Logit 模型和 Probit 模型。Probit 型随机交通分配模型目前只能用蒙特卡罗仿真的方法来进行，并难以获得出行者对不同路径的选择概率；而 Logit 型随机交通分配模型由于存在分析算法，可以针对不同要求进行有效的分析，特别是对出行者的动态选择行为分析极为有用，因而得到了研究者广泛的重视。

1.3.4 动态交通分配

动态交通分配理论从提出至今经过了 20 多年的发展，在理论研究和方法应用上都有了一定的进步，但不同于静态交通分配已经有其成熟的理论和方法，动态交通的研究仍然处于发展阶段，主要原因是考虑了时间变动因素后，建立合适的数学模型和设计合适的算法变得十分困难。

Merchant 和 Nemhauser 于 1978 年第一个以数学规划方法对动态交通分配问题进行了开创性的研究[11~12]，他们提出了用离散时间，非凸的非线性规划来表达的系统最优分配模型。Ho 在 1980 年为这个 M-N 模型提出了分段线性化算法[13]，而后又提出了应用嵌套式分解算法在超立方并行计算机上求解的方法[14]。为了进行有效的最优性分析，1987 年 Carey 在 M-N 模型的基础上进行了改进[15]，构造了一个非线性的凸规划模型。上述各个阶段的模型为动态交通分配带来了巨大的推力，但是各个模型的最大缺点是局限于多个起点和单个终点的简单网络，用于理论分析可以，但是如果要应用到现实中的多个起点和多个终点的城市交通网络还需大量的研究和努力。

1980 年，Luque 和 Friesz 提出了应用最优控制理论解决动态系统最优模型的新思想，他们将 M-N 模型改进成为一个连续时间的最优控制问题[16]，随后 Wie、Friesz 和 Tobin 在 1990 年，Ran 和 Boyce 等人在 1992 年发表的文章中建立的模型均采用了最优控制理论方法建模[17~19]。最优控制理论方法建立的模型具有易于分析的优点，但是对于 Friesz 和 Wie 等人提出的模型，目前仍然没有有效的成熟的求解算法。Papageorgiou 和 Mayr 使用的直接最优法只适用于小规模网络[20]，Janson 先把连续问题变成很多静态分配问题[21]，然后逐个求解，类似于模拟方法。

Janson 在 1990 和 1991 年相继发表文章，提出了动态交通分配的多目标规划模型，之后 Jayakrishnan 和 Tski 等人于 1995 年对该模型进行了改进，使其趋于

更加完善。Friesz 和 Bernstein 在 1993 年提出了动态系统最优的变分不等式模型。这些模型极大地丰富了动态交通分配的研究方法，从不同角度为解决动态分配问题做出了有益的尝试[10]。

另外，Mahmasani 和 Peeta 在 1993 年提出了一个计算机模拟的动态交通分配模型，考虑了随时间变化的交通需求以及交通拥挤条件下排队的形成等影响因素，但是该模型没有考虑对小区 OD 数据转换到路段 OD 数据，在路网阻抗的计算上采取了简化处理的方法[10]。

在国内对于动态交通分配理论和方法的研究还处于起步的阶段。纵观国内外对动态交通分配理论和方法的研究，到目前为止从研究方法角度而言，可以分为：数学规划建模方法、最优控制理论建模方法、变分不等式理论建模方法和计算机模拟方法等四种途经。

第 2 章 Logit 分配模型及其加载算法回顾

2.1 非集计方法概述

20 世纪 60 年代日本学者提出交通方式划分的“非集计模型方法（Disaggregate Model Method）”概念和模型，他们借用经济学的效用理论，在这个问题上开创了交通方式划分的非集计模型的研究。非集计模型吸引了大批的学者的研究兴趣，至今仍是交通规划理论中的一个热门问题。后来人们发现它不仅可以应用于交通方式划分，还可以用来解决诸如交通发生、交通分布、交通分配等所有的有关选择的问题。

以交通分区为研究单位的，将分区中个人或家庭的调查数据进行统计处理，如求平均值、求比例等；再用这些统计值来标定交通发生、分布、方式划分模型中的参数。在这个过程中关于个人和家庭的原始数据在统计时被集中处理了，也就是被集计化了，这种方法叫集计方法（Aggregate Method），得出的模型叫集计模型（Aggregate Methods）。集计方法存在的缺点是：为了保证模型的精度，要求相当规模的样本容量；而且在统计求和过程中没有充分利用各个个体（个人和家庭）的全部调查数据，即存在信息浪费。因此集计方法所需要的调查费用是很大的。

非集计方法（Disaggregate Method）的分析对象是个体，它将个体的原始数据不作任何统计处理直接用来构造模型，它的特点是：调查所得的个人的数据能得到充分的运用；要求的样本容量较小。

可供选择的交通方式或路径，叫做“选择枝（Alternative）”。如果一共只有两个选择枝可供选择，就是一个二项选择问题，否则就是多项选择问题。实际中，碰到较多的是多项选择问题，而且往往不同的出行者可选择的范围不同，即有不同的选择枝集合。

某个选择枝具有令人满意的程度叫做“效用（Utility）”。关于效用我们首先作以下基本假定，这些假定是基于人们通常的心理选择行为，是非集计模型的基础：

① 个人在每次抉择中总选择效用值最大的选择枝；

② 个人关于每个选择枝的效用值由个人自身的特性和选择枝的特性共同决定。

2.2 Logit 选择模型概述

效用是由选择枝本身的特征和个人的社会经济特性两方面的因素决定的，但我们不能对影响效用的全部因素进行量测，所以应该将效用看作随机变量。令

$$U_{nj} = V_{nj} + \varepsilon_{nj} \tag{2-1}$$

式中：U_{nj}——个人 n 关于选择枝 j 的效用；

V_{nj}——能够观测到的因素构成的效用确定项；

ε_{nj}——不能够观测到的因素构成的效用随机项。

为了书写方便，如无必要，一般省去表示个人的下标 n，简写为：

$$U_j = V_j + \varepsilon_j \tag{2-2}$$

为了叙述简便起见，在下面的推导过程中，假定一共只有两个选择枝。根据前面的基本假定，某出行者选择选择枝 1 的概率为

$$\begin{aligned} P_1 &= \Pr(U_1 > U_2) \\ &= \Pr(V_1 + \varepsilon_1 > V_2 + \varepsilon_2) \\ &= \Pr(\varepsilon_2 < V_1 - V_2 + \varepsilon_1) \\ &= \int_{-\infty}^{+\infty} \Pr(\varepsilon_1 = y, \varepsilon_2 < V_1 - V_2 + y)\mathrm{d}y \\ &= \int_{-\infty}^{+\infty} [\int_{-\infty}^{V_1 - V_2 + y} f_{12}(y,z)\mathrm{d}z]\,\mathrm{d}y \end{aligned} \tag{2-3}$$

其中，$f_{12}(y,z)$ 是 ε_1 和 ε_2 的联合概率密度函数。

如果假定 ε_1 和 ε_2 相互独立且具有相同的概率分布，其密度函数为 f，则其联合分布密度函数 $f_{12}(y,z) = f(y)f(z)$。于是

$$P_1 = \int_{-\infty}^{+\infty} f(y)[\int_{-\infty}^{V_1 - V_2 + y} f_{12}(z)\mathrm{d}z]\,\mathrm{d}y \tag{2-4}$$

进一步假定ε_1和ε_2都服从二重指数分布（又叫 Gumbel 分布、Weibull 分布或极值分布），其概率分布函数和概率密度函数分别为：

$$F(y)=\exp[-\exp(-\theta y)],\quad f(y)=\theta F(y)\exp(-\theta y) \tag{2-5}$$

其中，$\theta>0$为参数，可以推得它与$\varepsilon=\varepsilon_1$的均值和方差具有关系：

$$E(\varepsilon)=\frac{\gamma}{\theta},\quad D(\varepsilon)=\frac{\pi^2}{6\theta} \tag{2-6}$$

其中，γ是 Euler 常数，约等于 0.5772。

把式（2-5）代入式（2-4），得到：

$$\begin{aligned}P_1&=\int_{-\infty}^{+\infty}f(y)[\int_{-\infty}^{V_1-V_2+y}f_{12}(z)\mathrm{d}z]\,\mathrm{d}y\\&=\theta\int_{-\infty}^{+\infty}\exp(-\theta y)F(y)F(V_1-V_2+y)\mathrm{d}y\end{aligned} \tag{2-7}$$

令：$w=F(y)F(y+V_1-V_2)$，则

$$w=\exp[-\exp(-\theta y)\cdot(1+\exp(\theta V_2-\theta V_1))] \tag{2-8}$$

$$\frac{\mathrm{d}w}{\mathrm{d}y}=\theta w\exp(-\theta y)\cdot[1+\exp(\theta V_2-\theta V_1)] \tag{2-9}$$

由于当$y=\infty$时，$w=\exp(0)=1$；由于当$y=-\infty$时，$w=\exp(-\infty)=0$。故有

$$\begin{aligned}P_1&=\theta\int_{-\infty}^{+\infty}w\exp(-\theta y)\mathrm{d}y\\&=\int_0^1\frac{w\exp(-\theta y)}{w\exp(-\theta y)[1+\exp(\theta V_2-\theta V_1)]}\mathrm{d}w\\&=\frac{1}{1+\exp(\theta V_2-\theta V_1)}\\&=\frac{\exp(\theta V_1)}{\exp(\theta V_1)+\exp(\theta V_2)}\end{aligned} \tag{2-10}$$

此为二项 Logit 模型，简记为：BNL（Binary-nomail Logit）。

如果有多个选择枝，由于一般各人根据自己的实际情况可选择的范围不一定相同，设个人 n 选择枝的集合为 A_n，令

$$A=\bigcup_{n=1}^{N} A_n \tag{2-11}$$

为了统一起见，就设每个人的选择枝集合都为 A，并用 J 表示 A 中选择枝的数目。同理可得多项 Logit 模型：选择选择枝 j 的概率为

$$P_j=\frac{\exp(\theta V_j)}{\sum_{i=1}^{J}\exp(\theta V_i)}=\frac{1}{1+\sum_{i\neq j}[\theta(V_i-V_j)]} \tag{2-12}$$

多项 Logit 模型又被简记为 MNL（Multi-nomial Logit）。

2.3 Logit 交通分配模型

对于路段阻抗为常数的交通网络，Logit 分配模型由于其表达式简单，易于计算，在交通分配中被广泛地运用而占有重要的地位。

不失一般性，假设网络为一交通网络并由有向图 $G(N,A)$ 来表示，其中 N 为节点的集合，A 为弧的集合。设网络中任意一条弧路段 a 实际的交通阻抗(或弧的权)为 t_a，则从网络中任一起点 r 到终点 s 的路径 k 的阻抗为：

$$c_k^{rs}=\sum_a \delta_{a,k}^{rs}\cdot t_a \tag{2-13}$$

式中如果弧 a 在路径 k 上则 $\delta_{a,k}^{rs}=1$，否则 $\delta_{a,k}^{rs}=0$。

对于每个道路利用者总是选择自己认为阻抗最小的路径 k，此时称为道路利用者主观判断的阻抗值为“感知阻抗”，用 C_k^{rs} 表示，则有

$$C_k^{rs}=c_k^{rs}+\varepsilon_k^{rs},\quad \forall k,r,s \tag{2-14}$$

式中，ε_k^{rs}——随机误差项，有 $E[\varepsilon_k^{rs}]=0$。

根据 Wardrop 路径选择原则，第 k 条路径被选择的概率为：

$$\Pr(k)=\Pr(C_k^{rs}\le C_l^{rs}),\quad \forall l\neq k;\forall k,r,s \tag{2-15}$$

根据效用理论中有关“效用”的定义，可以用路径感知阻抗的负值来表示选择的效用U_k，即：

$$U_k = -C_k^{rs} = -c_k^{rs} - \varepsilon_k^{rs} \tag{2-16}$$

此时，路径的选择就是一个多项选择中选择效用最大的选择枝的问题。根据随机效用理论，假定ε_k^{rs}相互独立，且服从相同的 Gumbel 分布（此时，可以用一个ε表示所有的ε_k^{rs}）的条件下，路径k的选择概率为：

$$\Pr(k) = \frac{\exp(-\theta \cdot c_k^{rs})}{\sum_{k \in P_{rs}} \exp(-\theta \cdot c_k^{rs})} \tag{2-17}$$

式中，P_{rs}为所有从起点r到终点s的路径的集合；$\theta > 0$为模型参数，与ε的方差有关，可以证明$\theta = \frac{\pi^2}{6D(\varepsilon)}$，它是用户对阻抗的判断误差的放大因子，$\theta$越小则用户对阻抗的判断误差越大，反之则误差越小。当θ趋向于无穷时，用户对阻抗值的判断不存在误差，模型就变成确定型的网络加载模型。

这就是标准的 Logit 分配模型，它是一个随机分配模型，建立在出行者选择某条路径的概率同该路径的阻抗的负指数幂成正比。

假设从起点r到终点s的交通需求量为Q_{rs}，即可求出路段a上交通量为

$$x_a = \sum_{k \in P_{rs}} \delta_{a,k}^{rs} \cdot f_k^{rs} \tag{2-18}$$

式中，$f_k^{rs} = \Pr(k) \cdot Q_{rs}$为路径$k$上的交通流量。

2.4 Logit 型网络加载算法回顾

2.4.1 Dial 算法

用 Logit 模型求路径选择概率需要把从起点r到终点s之间所有的路径都找出来，这在实际中是非常困难的。Dial 在 1971 年提出了一种算法[22]，使得 Logit 的分析求解成为可能。

考虑一个用有向图$G(N,A)$表示的交通网络，其中 N 为所有结点的集合，A 为所有路段的集合。rs 为网络中的任意 OD 对，r 为起点，s 为迄点。用$r_{(i)}$和$s_{(i)}$分别表示网络中任一结点 i 到起点 r 及迄点 s 的最短路的权值。Dial 算法分为“Double-pass”和“Single-pass”两种，由于原理类似，故在此仅介绍“Single-pass”算法。

Dial 算法的核心步骤分为三步。

首先，根据下式计算路段的“似然值”

$$L_{(i\to j)}=\begin{cases}\exp\left[\theta\cdot\left(r_{(j)}-r_{(i)}-t_{(i\to j)}\right)\right] & \text{如果} r_{(i)}<r_{(j)}\\ 0 & \text{其他情况}\end{cases} \tag{2-19}$$

式中$\theta>0$为 Logit 参数，$t_{(i\to j)}$为路段$(i\to j)$的阻抗。

接着，Dial 算法的“前推过程”将结点按照$r_{(i)}$进行升序排列，并依次计算路段$(i\to j)$的“权重”$w_{(i\to j)}$：

$$w_{(i\to j)}=\begin{cases}L_{(i\to j)} & \text{如果} i=r\\ L_{(i\to j)}\sum\limits_{m\in I(i)}w_{(m\to i)} & \text{其他情况}\end{cases} \tag{2-20}$$

式中$I(i)$表示结点 i 的所有上游结点的集合。

最后，Dial 算法用将结点根据$r_{(i)}$ 进行降序排列，并用“逆推过程”来获得路段流量$x_{(i\to j)}$：

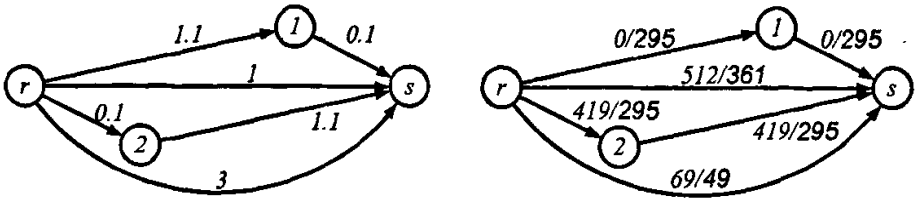
$$x_{(i\to j)}=\frac{w_{(i\to j)}}{\sum\limits_{m\in O(j)}w_{(m\to j)}}\cdot\left[q_{rj}+\sum_{m\in I(j)}x_{(j\to m)}\right] \tag{2-21}$$

式中q_{rj}为起点 r 到节点 j 的出行需求，$O(j)$为结点 j 的所有下游结点的集合(如果$O(j)$为空集，则$x_{(i\to j)}$为零)。

可以证明，Dial 算法与 Logit 模型是等价的[23]。

Dial 算法通过公式（2-19），巧妙地将所有不满足“合理性”假设的路段排除在计算之外，从而保证公式（2-20）和（2-21）中等式右边的权重和流量在计算时，要么为零要么已经计算过，并排除了所有的环路。但无论是“Double-pass”

还是“Single-pass”算法，“合理路径”的定义都过于严格，导致实际生活中被大量使用的路段被排除，被分配的交通量为零，使得 Dial 算法在实际中的应用受到限制[24~25]。



(*a*) 网络阻抗　　(*b*) 交通流量分配（Dial Single-pass/理论值）

图 2-1 Dial 算法的缺陷

Dial 算法的缺陷可由图 2-1 所示的交通网络来说明。假设起点为点 r，终点为点 s，从 r 到 s 的交通需求为 1000。假设 Logit 参数 $\theta=1.0$，图 2-1(b)中给出了“Dial Single-pass”算法分配的交通流结果，括号内为根据 Logit 模型定义的理论值。很显然，路径 $r\to1\to s$ 的路抗都为 1.2，而 $r\to s$ 的路抗为 3.0（2 条路径中阻抗较大的那一条），但前者所分配的交通流量为 0，而后者却为 69，Dial 算法给出的结果显得很不合理，同理论计算的结果也有很大出入。

2.4.2 Bell_Akamatsu 全路径算法

由于 Dial 算法存在上述缺陷，Bell 提出了两种计算方法[25]。第 1 种计算方法为近似算法，其计算结果与计算的循序有关，因而其结果不具备唯一性和稳定性，且缺乏理论依据证明计算结果和 Logit 模型的定义相符。Bell 的第 2 种方法假设网络的权重矩阵 W 的求和序列收敛并通过矩阵求逆来计算 Logit 模型，即

$$\sum_{l=1}^{\infty}W^{l}=(I-W)^{-1}-I \tag{2-22}$$

Wong 的研究[26]表明如果网络不存在回路，则求和序列收敛；如果网络包含回路，则存在某个值 θ_0，当 $\theta>\theta_0$ 时序列收敛，否则序列不收敛。Akamatsu 后来证明 Bell 这一计算方法和包含所有路径的 Logit 模型是等价的，并建立了一个仅包含路段流量变量的模型[27]。Bell_Akamatsu 算法没有排除任何路径，即所有无限循环的回路均包括在计算中。对于一个大型网络而言，Bell_Akamatsu 算法

缺乏有效的计算方法，因为对于一个包含数千个节点的中型交通网络而言，由于矩阵 W 的维数为 $n \times n$（n 为网络中节点的个数），矩阵求逆所带来的计算工作量是极其巨大的。虽然可换成用迭代法来求解[28]，但是由于 Dial 算法固有的计算顺序，很容易造成迭代的不收敛，且迭代法的迭代效率也制约着运算的速度。

2.5 综述总结

综上所述，源于非计集方法的 Logit 型随机交通分配模型考虑出行者对路径旅行时间的认识偏差，是一种更符合实际的分配模型。由于存在分析算法——Dial 算法，可以针对不同要求进行有效的分析，特别是对出行者的动态选择行为分析极为有用，因而也得到了研究者广泛的重视。

随着社会的发展，交通规划网络日趋庞大，而且交通流动态分析也日渐受到重视，Dial 算法具有的高效性可以满足发展的要求，但 Dial 算法的计算存在较大的误差。虽然 Bell_Akamatsu 算法可以改进 Dial 算法的缺陷，但其计算效率仅仅可以用于微型网络，难以满足大型网络应用的要求。所以本文将力求在保持 Dial 算法高效率的前提下，通过放宽了“合理”的定义，改进了节点计算顺序，使得 Dial 算法中“不合理”但现实中却被采用的路径参加计算。这种新算法便是下一章讲述的基于拓扑处理的 Logit 网络加载算法。

第 3 章 基于拓扑处理的 Logit 网络加载算法

考虑图 3-1 所示网络，根据 Dial 算法，路径 $r\rightarrow1\rightarrow3\rightarrow2\rightarrow s$ 的交通流量将为零。Dial 算法“前推过程”所决定的节点计算顺序为$(r, 1, 2, 3, s)$。不难看出如果将计算顺序改为$(r, 1, 3, 2, s)$，同时去掉式（2-19）中有关合理路段的限制，即所有路段的似然值均按 $L_{(i\rightarrow j)}=\exp\left[\theta\cdot\left(r_{(j)}-r_{(i)}-t_{(i\rightarrow j)}\right)\right]$ 计算，可以得到和利用 Logit 定义来计算完全相同的结果。由于图 3-1 是一个无环的网络，$(r, 1, 3, 2, s)$实际上是该网络的一个拓扑排序。这个例子表明，对于无环网络，采用拓扑序列来计算可以得到和和理论值完全相同的结果。有环网络不存在拓扑排序，因此不能采用该方法，但有环网络可以通过断开环路中的某些路段变为无环网络，从而可以采用拓扑排序进行计算，这就是本文对 Dial 算法进行改进的出发点。

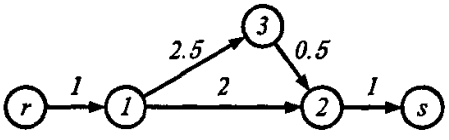


图 3-1 示例网络

3.1 拓扑处理法

拓扑处理法是新算法的关键步骤，通过这个步骤确定了路段的状态以及节点计算顺序。

在介绍拓扑处理法之前，先为路段定义一个状态变量 $\xi_{(i\rightarrow j)}$。如果 $\xi_{(i\rightarrow j)}=0$，表示路段没有被处理；$\xi_{(i\rightarrow j)}=-1$，表示该路段已从路网中删除；$\xi_{(i\rightarrow j)}=1$，表示该路段的似然值是“可计算的”。

拓扑处理法的步骤如下：

步骤 0：初始化。计算起点 r 到所有节点的最短路 $r_{(i)}$，并将所有路段状态变量置零，即设 $\xi_{(i\rightarrow j)}=0$。并将起点 r 的所有入弧以及迄点 s 的所有出弧删掉，同时将这些弧段的状态设为 $\xi_{(k\rightarrow l)}=-1$。

步骤 1：建立一个集合 $K=\{r\}$ 和一个空队列 Q。

步骤 2：从 K 中选取一个所有入弧状态为-1 或 1 的节点 i；如果不存在这样的点，则选取 $r_{(i)}$ 最小的点 i。将 i 从 K 中删除并追加到队列 Q 中，将 i 所有入弧状态设为 $\xi_{(k\to i)}=-1$。

步骤 3：遍历节点 i 的每一个下游节点 j，将 j 加入到 K 中，并设 $\xi_{(i\to j)}=1$；

步骤 4：如果 K 为空集，结束；否则转向步骤 2。

从该算法可以看出，对于无环网络而言，“步骤 2”每一次从 K 中选取的点，其入度（进入该节点弧段的条数）都为 0，算法本身是一种拓扑排序算法。而对于有环网络，“步骤 2”则是本算法的关键，它通过删除特定路段，将有环网络变为无环网络，同时也决定了节点的计算顺序，以保证一个节点被计算时其入弧状态为 1 或-1。

设网络的弧数为 n，节点数为 m，最短路的计算采用 Dijkstra 算法[29]，其算法复杂度为 $O(n^2)$，而拓扑处理法中从 K 中选取一个点的复杂度也为 $O(n^2)$。因此拓扑处理法的复杂度为 $O(n^2)$。如果路网和集合 K 均采用优先序列结构改进[30]，则拓扑处理法计算时间复杂度将降低为 $O(m+n\log n)$。

下面以一个简单的网络示例（见图 3-2）来说明拓扑处理法具体的处理过程及步骤，具体处理步骤如表 3-1 所示。

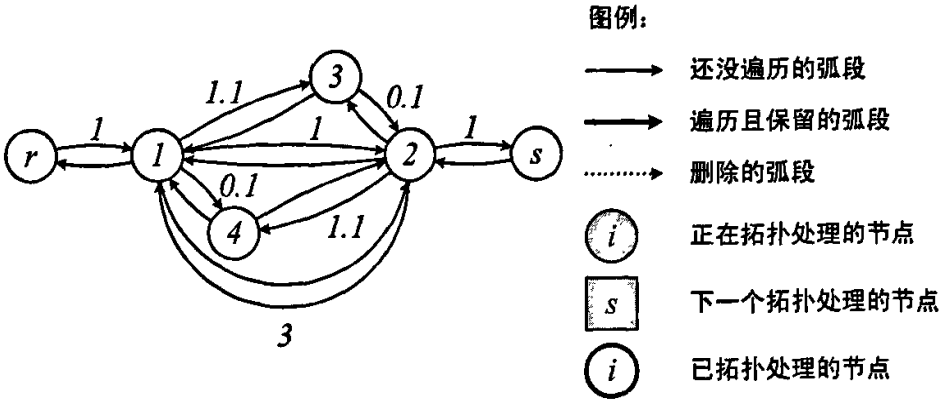
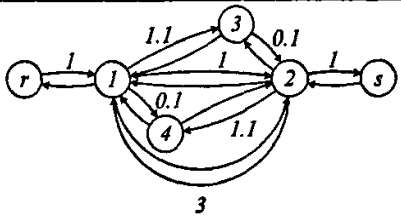
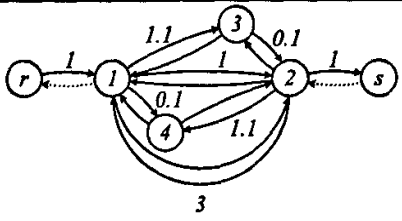
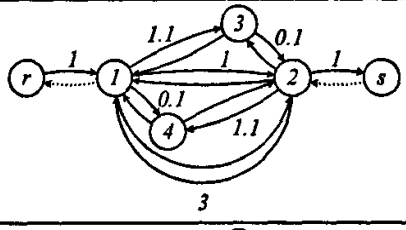
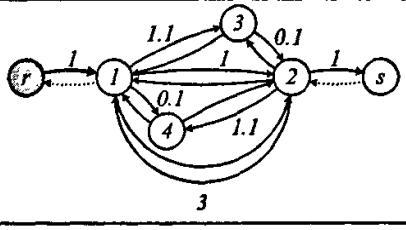
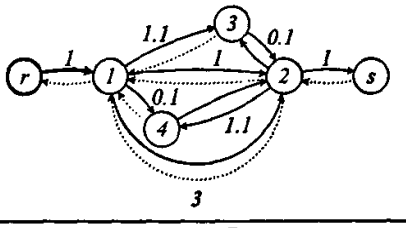
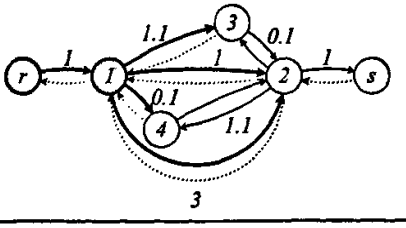
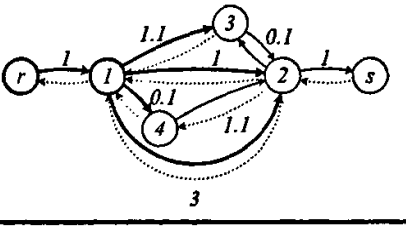
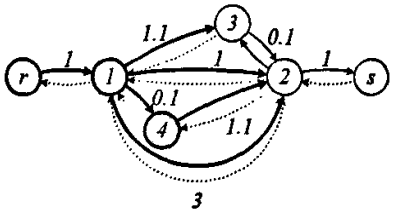
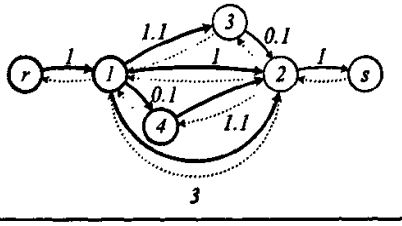
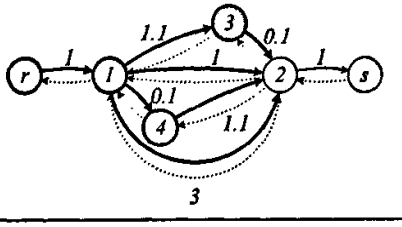
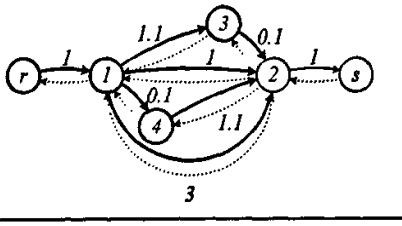
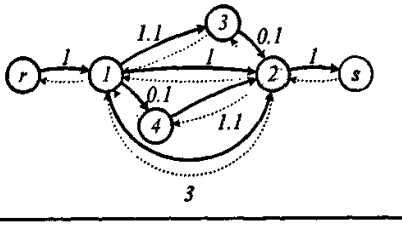
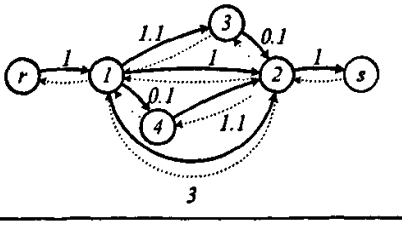
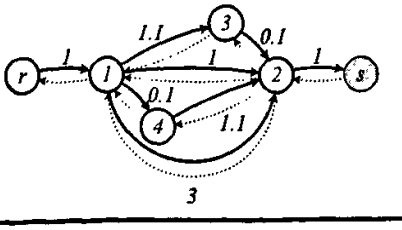


图 3-2 拓扑处理法示例网络

表 3-1 拓扑处理法示例图解说明

步骤	网络状态		说明
(1)		K: Q:	网络初始状态，集合 K 和队列 Q 均为空；
(2)		K: r Q:	将起点 r 的所有入弧以及迄点 s 的所有出弧删掉，并将 r 放入集合中；
(3)		K: Q: r	从 K 中选取一个所有入弧状态为-1 或 1 的节点，此次选取节点 r，将 r 从 K 中删除并追加到队列 Q 中；
(4)		K: 1 Q: r	遍历节点 r 的每一个下游节点 1，将节点 1 加入到 K 中，并将节点 r 还没遍历的出弧的状态变量设为 1；
(5)		K: Q: r 1	因为 K 中没有一个所有入弧状态为-1 或 1 的节点，所以从 K 中选取选取 $r_{(i)}$最小的节点 1，将节点 1 从 K 中删除并追加到队列 Q 中；
(6)		K: 3 2 4 Q: r 1	遍历节点 1 的每一个下游节点 2、3、4，将节点 2、3、4 加入到 K 中，并将节点 1 还没遍历的出弧的状态变量设为 1；
(7)		K: 3 2 Q: r 1 4	因为 K 中没有一个所有入弧状态为-1 或 1 的节点，所以从 K 中选取选取 $r_{(i)}$ 最小的节点 4，将节点 4 从 K 中删除并追加到队列 Q 中；

步骤	网络状态		说明
(8)		K: 3 2 Q: r 1 4	遍历节点 4 的每一个下游节点 2，因节点 2 已在 K 中，无需重复。只需并将节点 4 还没遍历的出弧的状态变量设为 1；
(9)		K: 2 Q: r 1 4 3	因为 K 中没有一个所有入弧状态为-1 或 1 的节点，所以从 K 中选取选取 $r_{(i)}$ 最小的节点 3，将节点 3 从 K 中删除并追加到队列 Q 中；
(10)		K: 2 Q: r 1 4 3	遍历节点 3 的每一个下游节点 2，因节点 2 已在 K 中，无需重复。只需并将节点 3 还没遍历的出弧的状态变量设为 1；
(11)		K: Q: r 1 4 3 2	从 K 中选取一个所有入弧状态为-1 或 1 的节点，此次选取节点 2，将 2 从 K 中删除并追加到队列 Q 中；
(12)		K: s Q: r 1 4 3 2	遍历节点 2 的每一个下游节点 s，将节点 s 加入到 K 中，并将节点 2 还没遍历的出弧的状态变量设为 1；
(13)		K: Q: r 1 4 3 2 s	从 K 中选取一个所有入弧状态为-1 或 1 的节点，此次选取节点 s，将 s 从 K 中删除并追加到队列 Q 中；
(14)		K: Q: r 1 4 3 2 s	s 为最后一个节点，此时 K 为空集，拓扑处理法结束，此时的网络状态也是最终等效网络的形状。

3.2 基于拓扑处理的 TPDial Single-pass 算法

由于最短路的计算是上述算法中最耗时的环节，提高计算效率的最好办法是减少最短路的计算次数。前面描述的拓扑处理法虽然通过改进可以把时间复杂度降低为$O(m+n\log n)$，但由于从K中选取一个点的复杂度和最短路的计算时间复杂度是一样的，所以实际上一次拓扑处理的过程其实是两倍最短路的计算时间，这对于算法在大型网络或动态交通分配中的应用[31]是不利的。

文献[32]提出了一种基于拓扑排序的最短路径算法与拓扑处理法求解节点计算顺序的算法接近。借鉴此算法进行改进，改进的思想是将拓扑遍历的求解节点计算顺序与求最短路径结合起来，同时利用 Dial 单步算法的特点，一方面减少最短路的次数，同时避免对每一 OD 进行拓扑遍历[33]。这样可以在提高加载算法的计算效率的同时改进 Dial 算法对合理路径的定义。

3.2.1 TPDial Single-pass 算法

考虑由节点$N=\{i\}$和路段$A=\{(i\to j)\}$组成的道路网络$G(N,A)$，分别用$L_{(i\to j)}$、$w_{(i\to j)}$和$x_{(i\to j)}$来表示路段$(i\to j)$的似然值、权重和交通流量。为每条路段定义一个状态变量$\xi_{(i\to j)}$，其取值为 0、1 或-1，分别表示该路段未被遍历、已遍历或被从路网中删除。设 OD 对$r-j$改进算法的步骤如下：

步骤 0：初始化。将所有路段变量置零，即设

$$\xi_{(i\to j)}=0;L_{(i\to j)}=0;w_{(i\to j)}=0;x_{(i\to j)}=0\ 。$$

步骤 1：设所有节点 i 到起点 r 的距离$r_{(i)}=\infty$；将所有路段的状态变量$\xi_{(i\to j)}=0$；设$r_{(r)}=0$；建立一个集合$K=\{r\}$和一个空队列Q。

步骤 2：从K中选取一个所有入弧状态为-1 或 1 的节点i；如果不存在这样的点，则选取$r_{(i)}$最小的点i。将i从K中删除并追加到队列Q中，将i所有入弧状态设为$\xi_{(k\to i)}=-1$。

步骤 3：遍历节点 i 的每一个下游节点 j，将 j 加入到 K 中，并设 $\xi_{(i\to j)}=1$；如果 $r_{(i)}+t_{(i\to j)}<r_{(j)}$，则更新 $r_{(j)}=r_{(i)}+t_{(i\to j)}$。

步骤 4：如果 K 为空集，转步骤 5；否则转向步骤 2。

步骤 5：依照 Q 的顺序分别计算路段的似然值和权重：

$$L_{(i\to j)}=\begin{cases}\exp\left[\theta\cdot\left(r_{(j)}-r_{(i)}-t_{(i\to j)}\right)\right] & \text{如果 } \xi_{(i\to j)}=1,\\ 0 & \text{其他情况}\end{cases} \tag{3-1}$$

$$w_{(i\to j)}=\begin{cases}L_{(i\to j)} & \text{如果 } i=r,\\ L_{(i\to j)}\cdot\sum_{m}w_{(m\to i)} & \text{其他情况}\end{cases} \tag{3-2}$$

步骤 6：依照 Q 的逆序计算路段的交通流量：

$$x_{(i\to j)}=\left(q_{rj}+\sum_{k}x_{(j\to k)}\right)\frac{w_{(i\to j)}}{\sum_{m}w_{(m\to j)}} \tag{3-3}$$

这里如果 rj 不是 OD 对则 $q_{rj}=0$。

新算法定义了一个包含所有状态 $\xi_{(i\to j)}=1$ 的路网上 Logit 型多路径分配模型，而所有 $\xi_{(i\to j)}\neq 1$ 的路段流量为零。

虽然新算法并不要求 $r_{(i)}$ 是最短路，但不难看出该算法“步骤 3”所得到的 $r_{(i)}$ 就是节点 i 到起点 r 的最短路，其证明可参阅文献[32]。新算法所得到的合理路径数目不少于单步的 Dial 算法，特别是如果网络是无环的，则 Q 就是网络的拓扑排序，从而算法可以得到和理论解完全一致的结果，这是新算法比 Dial 算法的优越之处。

改进的算法将拓扑处理和求解最短路径合并在一起，从而比文献[34～35]提到的算法节省了一次求解最短路径的计算，大大提高了计算效率。算法从起点出发，求解出起点到各个节点的最短路径 $r_{(i)}$ 以及等效网络的拓扑序列 Q。注意到算法的“步骤 2”是“选取 K 中 $r_{(i)}$ 最小的节点”，刚好可以利用“步骤 3”求解的 $r_{(i)}$。因为这次拓扑处理只计算了一次最短路，即 $r_{(i)}$，所以此算法称为“TPDial Single-pass”。

而类似地，Single-pass 算法也可以从终点出发，逆求出各个节点到终点的最短路径 $s_{(i)}$ 和相应的逆拓扑序列。

TPDial Single-pass 算法的可以用图 3-3 的流程图简单表示。

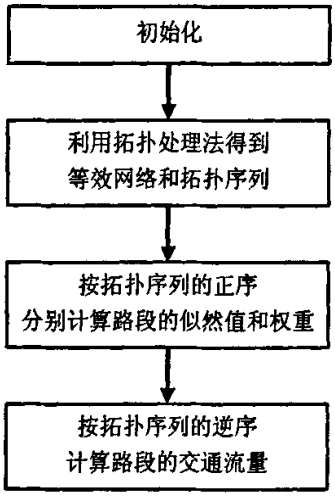


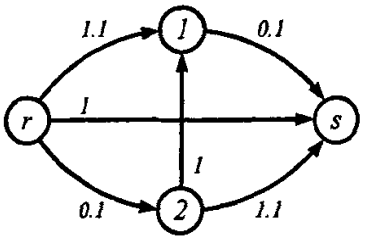
图 3-3 TPDial Single-pass 算法流程图

3.2.2 应用实例

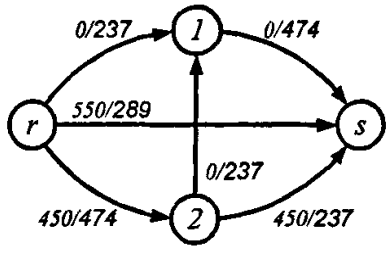
本节用实例来对 TPDial Single-pass 算法的计算精度与 Dial 算法进行比较。

(1) 无环网络的应用

例 3-1：图 3-4(*a*)的路网结构常常用于 Logit 模型的验证，OD 对 *r-s* 间的交通需求取为 1000，Logit 参数 $\theta=1$。图 3-4(*b*)给出了 Dial Single-pass 算法与 TPDial Single-pass 算法的分配结果。



(*a*) 网络阻抗



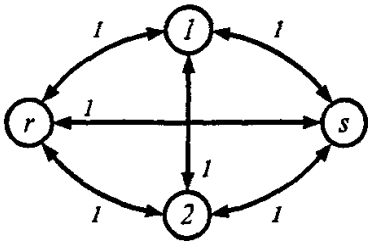
(*b*) 交通流量分配
(Dial Single-pass/理论值或 TPDial Single-pass)

图 3-4 无环网络的应用

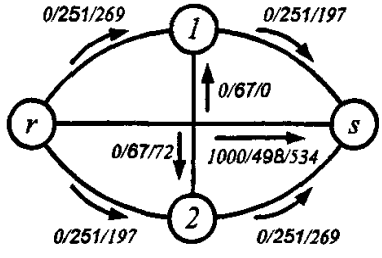
这里 Dial Single-pass 算法的节点计算顺序为$(r, 2, s, 1)$，导致路段$(2 \rightarrow 1)$和$(1 \rightarrow s)$从路网中删除，分配到的交通量为零；而 TPDial Single-pass 算法的节点计算顺序为$(r, 2, 1, s)$，所有的路段都参与流量分配。可以看出 TPDial Single-pass 算法对无环网络给出了与理论值是一样的结果，而 Dial Single-pass 算法的误差较大。

(2) 双向网络的应用

例 3-2：图 3-5(*a*)的是一个双向网络，OD 对 *r-s* 间的交通需求取为 1000，Logit 参数$\theta = 1$。图 3-5(*b*)给出了 Dial Single-pass 算法、TPDial Single-pass 算法及理论值的分配结果。



(*a*) 网络阻抗



(*b*) 交通流量分配

(Dial Single-pass/理论值/TPDial Single-pass)

图 3-5 双向网络的应用

此例中虽然 Dial Single-pass 算法和 TPDial Single-pass 算法的节点计算顺序均为$(r, 1, 2, s)$，但由于 Dial Single-pass 算法对有效路段严格定义的限制导致了$(2 \rightarrow 1)$和$(1 \rightarrow 2)$同时被删掉，而 TPDial Single-pass 算法是根据拓扑结构进行处理，在破除所有环路的前提下尽量保留更多的路段参与计算，所以其只删除了$(2 \rightarrow 1)$而保留了$(1 \rightarrow 2)$。由此例可以看出即便 Dial Single-pass 算法和 TPDial Single-pass 算法的节点计算顺序一样，但两者确定有效路段的方法不一样，前者是根据路段两端点$r_{(i)}$的关系来判断，而后者是根据是否构成环路来判断的，由于这种差异性导致了分配结果存在着较大的差异。

3.3 基于拓扑处理的 TPDial Double-pass 算法

如果已经从起点出发做了一次拓扑处理求出各个节点的$r_{(i)}$，恢复原来网络，即恢复之前求$r_{(i)}$时删掉的弧段，再通过从终点出发做一次拓扑处理，并将“TPDial Single-pass”算法的“步骤2”改成选“取K中$r_{(i)}$最大的节点”作为选择的标准，则可以将计算的精度进一步提高。因为总共进行了两次拓扑处理（最短路径计算），先计算了$r_{(i)}$，再计算$s_{(i)}$，参照 Dial 算法对 Double-pass 的定义，把在此基础上的 TPDial 算法称为“TPDial Double-pass”算法。

3.3.1 TPDial Double-pass 算法

考虑由节点$N=\{i\}$和路段$A=\{(i\to j)\}$组成的道路网络$G(N,A)$，分别用$L_{(i\to j)}$、$w_{(i\to j)}$和$x_{(i\to j)}$来表示路段$(i\to j)$的似然值、权重和交通流量。为每条路段定义一个状态变量$\xi_{(i\to j)}$，其取值为0、1或-1，分别表示该路段未被遍历、已遍历或被从路网中删除。对任一OD量为q_{rs}的OD对rs，“TPDial Double-pass”算法的步骤如下：

步骤0：初始化。将所有路段变量置零，即设

$$\xi_{(i\to j)}=0;L_{(i\to j)}=0;w_{(i\to j)}=0;x_{(i\to j)}=0\ 。$$

步骤1：设所有节点i到起点r的距离$r_{(i)}=\infty$；将所有路段的状态变量$\xi_{(i\to j)}=0$；设$r_{(r)}=0$；建立一个集合$K=\{r\}$和一个空队列Q。

步骤2：从K中选取一个所有入弧状态为-1或1的节点i；如果不存在这样的点，则选取K中$r_{(i)}$最小的点i。将i从K中删除并追加到队列Q中，将i所有入弧状态设为$\xi_{(k\to i)}=-1$。

步骤3：遍历节点i的每一个下游节点j，将j加入到K中，并设$\xi_{(i\to j)}=1$；如果$r_{(i)}+t_{(i\to j)}<r_{(j)}$，则更新$r_{(j)}=r_{(i)}+t_{(i\to j)}$。

步骤 4：如果 K 为空集，转步骤 5；否则转向步骤 2。

步骤 5：恢复为初始网络，将每条路段的状态变量置零，即设 $\xi_{(i\to j)}=0$；设所有节点 i 到终点 s 的距离 $s_{(i)}=\infty$；设 $s_{(s)}=0$，并清空队列 Q 同时令集合 $K=\{s\}$。

步骤 6：从 K 中选取一个所有出弧状态为-1 或 1 的节点 j；如果不存在这样的点，则选取 K 中 $r_{(j)}$ 最大的点 j；如果 $r_{(j)}$ 最大的节点不止一个，则选择 $s_{(j)}$ 最小的点 j；将 j 从 K 中删除并追加到队列 Q 中，将 j 所有出弧状态设为 $\xi_{(j\to k)}=-1$。

步骤 7：遍历节点 j 的每一个上游节点 i，将 i 加入到 K 中，并设 $\xi_{(i\to j)}=1$；如果 $s_{(j)}+t_{(i\to j)}<s_{(i)}$，则更新 $s_{(i)}=s_{(j)}+t_{(i\to j)}$。

步骤 8：如果 K 为空集，转步骤 9；否则转向步骤 6。

步骤 9：依照 Q 的逆序分别计算路段的似然值和权重：

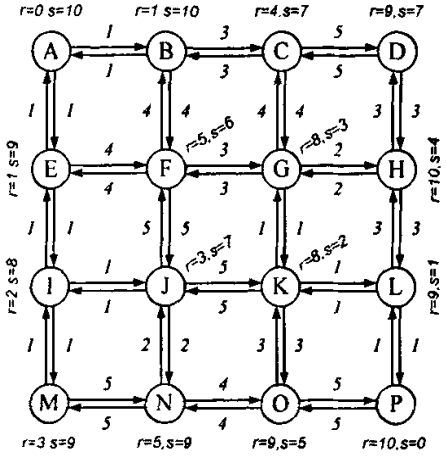
$$L_{(i\to j)}=\begin{cases}\exp\left[\theta\cdot\left(s_{(i)}-s_{(j)}-t_{(i\to j)}\right)\right] & \text{如果 } \xi_{(i\to j)}=1,\\ 0 & \text{其他情况}\end{cases} \tag{3-4}$$

$$w_{(i\to j)}=\begin{cases}L_{(i\to j)} & \text{如果 } i=r,\\ L_{(i\to j)}\cdot\sum\limits_{m}w_{(m\to i)} & \text{其他情况}\end{cases} \tag{3-5}$$

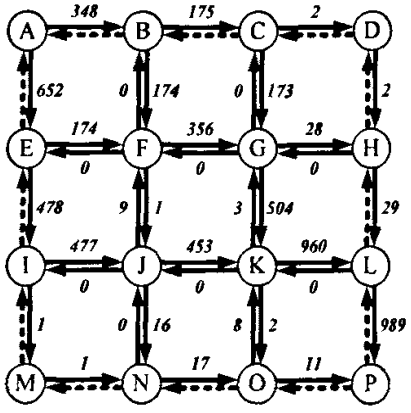
步骤 10：依照 Q 的顺序计算路段的交通流量：

$$x_{(i\to j)}=\begin{cases}q_{rs}\dfrac{w_{(i\to j)}}{\sum\limits_{m}w_{(m\to j)}} & \text{如果 } j=s,\\ \sum\limits_{k}x_{(j\to k)}\cdot\dfrac{w_{(i\to j)}}{\sum\limits_{m}w_{(m\to j)}} & \text{其他情况}\end{cases} \tag{3-6}$$

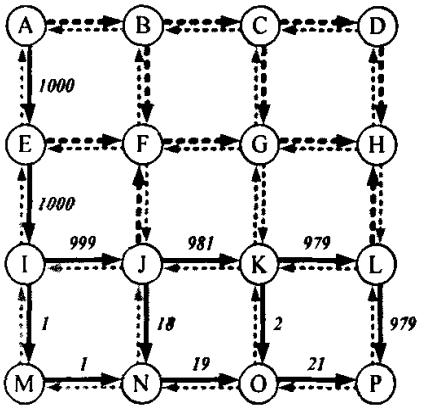
可以看出“TPDial Double-pass”算法是在“TPDial Single-pass”算法的基础上再做一次拓扑处理，总共进行了两次拓扑处理（最短路径计算），先计算了 $r_{(i)}$，再计算 $s_{(i)}$。注意到算法的“步骤 2”是“选取 K 中 $r_{(i)}$ 最小的节点”，刚好可以利用“步骤 3”求解的 $r_{(i)}$；而“步骤 6”是“选取 K 中 $r_{(j)}$ 最大的点 j”，刚好可以利用前面第一次拓扑处理求解出的 $r_{(i)}$；而“如果 $r_{(j)}$ 最大的节点不止一个，则选择 $s_{(j)}$ 最小的点 j”，这里的 $s_{(j)}$ 是利用“步骤 7”求解的 $s_{(j)}$。



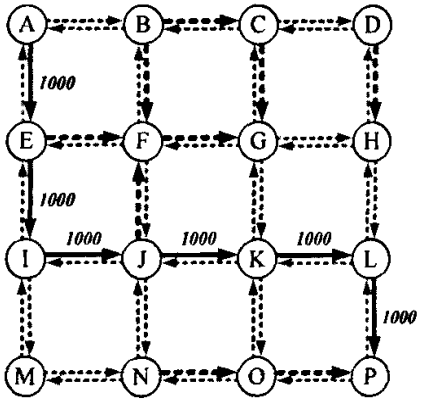
(*a*) 路网结构及各点最短路计算结果



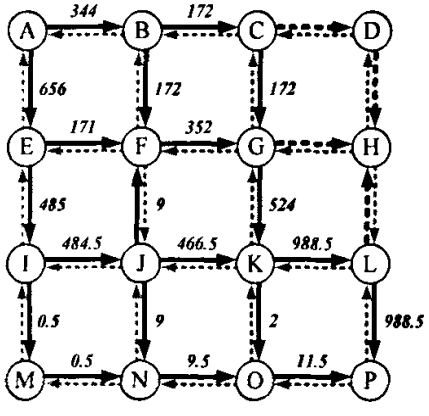
(*b*) 理论值（枚举法）



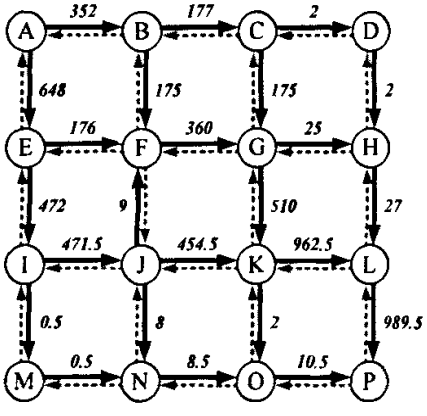
(*c*) Dial Single-pass 算法（只算 $r_{(i)}$）



(*d*) Dial Double-pass 算法



(*e*) TPDial Single-pass 算法（只算 $r_{(i)}$）



(*f*) TPDial Double-pass 算法（先算 $r_{(i)}$，再算 $s_{(i)}$）

图 3-6 各种算法的分配结果

表 3-1 各种算法参与计算的路径集

路径号	路径	路径长度	枚举法		Dial SinglePass【只算 $r_{(i)}$】		Dial DoublePass【算 $r_{(i)}$和 $s_{(i)}$】		TPDial SinglePass【只算 $r_{(i)}$】		TPDial DoublePass【先算 $r_{(i)}$，再算 $s_{(i)}$】	
			是否参与计算	流量	是否参与计算	流量	是否参与计算	流量	是否参与计算	流量	是否参与计算	流量
1	A→E→I→J→K→L→P	10	√	447.40	√	978.75	√	1000.00	√	465.89	√	453.56
2	A→B→C→G→K→L→P	11	√	164.59					√	171.39	√	166.86
3	A→B→F→G→K→L→P	11	√	164.59					√	171.39	√	166.86
4	A→E→F→G→K→L→P	11	√	164.59					√	171.39	√	166.86
5	A→B→C→G→H→L→P	14	√	8.19							√	8.31
6	A→B→F→G→H→L→P	14	√	8.19							√	8.31
7	A→E→F→G→H→L→P	14	√	8.19							√	8.31
8	A→E→I→J→N→O→P	14	√	8.19	√	17.93			√	8.53	√	8.31
9	A→E→I→J→F→G→K→L→P	14	√	8.19					√	8.53	√	8.31
10	A→B→C→D→H→L→P	16	√	1.11							√	1.12
11	A→E→I→J→K→O→P	16	√	1.11	√	2.43			√	1.15	√	1.12
12	A→B→C→G→K→O→P	17	√	0.41					√	0.42	√	0.41
13	A→B→F→G→K→O→P	17	√	0.41					√	0.42	√	0.41
14	A→E→F→G→K→O→P	17	√	0.41					√	0.42	√	0.41
15	A→E→I→J→F→G→H→L→P	17	√	0.41							√	0.41
16	A→E→I→AC→N→O→P	17	√	0.41	√	0.89			√	0.42	√	0.41
17	A→E→I→J→F→G→K→O→P	20	√	0.02					√	0.02	√	0.02
18	其余路径	-	√	13.59								
总计			184 条	1000.00	4 条	1000.00	1 条	1000.00	12 条	1000.00	17 条	1000.00

3.3.2 应用实例

本例采用4×4的棋盘状双向道路网，其路网结构如图 3-6(*a*)所示，A 为起点，P 为迄点。OD 流量为 $q_{rs}=1000$，Logit 参数 $\theta=1$。各种算法的计算结果如图 3-6 所示，其中理论值为枚举所有无环路径的分配结果。从分配结果可以看出：

(1) 保留的路段。枚举法所有的 48 条路段都保留，但有些路段虽然保留下来但却完全没有流量，如(B→A)，(C→B)等等，因为如果利用这些路段必然会使得路径中出现环路，所以这些路段在枚举法中虽然保留下来但却无法利用到。而 Dial Single-pass 算法有效路段有 23 条，占比 23/48=48%，实际参与计算的仅仅 12 条，占比 12/36=33%；Dial Double-pass 算法虽然有效路段有 15 条，占比 31%，但实际上参与计算的 6 条，占比 17%； TPDial Single-pass 算法保留路段有 24 条，占比 50%，实际参与计算的 20 条，占比 56%；TPDial Double-pass 算法保留路段有 24 条，占比 50%，实际参与计算的 24 条，占比 67%。可以看出 TPDial Double-pass 算法保留并参与计算的路段是最多的，而由于 Dial Double-pass 算法对有效路段的定义最为严格，所以最终保留并参与的路径最少。

此外，可以看出 TPDial 算法保留路段包含了 Dial Single-pass 算法的所有路段，而 Dial Single-pass 算法保留路段又包含了 Dial Double-pass 算法的所有路段。但 TPDial Double-pass 算法和 TPDial Single-pass 算法之间的路段集并无完全的包含与被包含的关系，例如 TPDial Double-pass 算法包含了路段(H→L)，而 TPDial Single-pass 算法则包含了路段(L→H)。

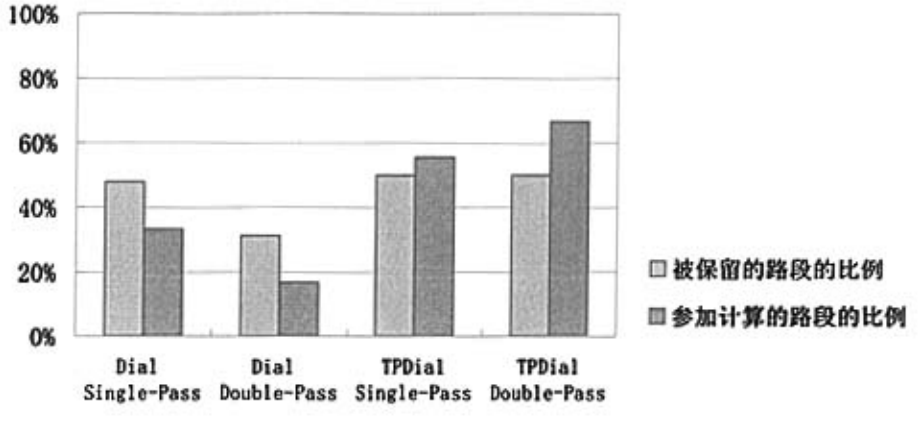


图 3-7 各种算法的路段集比较

⑵ 保留的路径。枚举法枚举出来的无环路径总数为 184 条。从表 3-1 可以看出 Dial Double-pass 算法参与计算的路径数只有 1 条，这一条为从起点 A 到终点 P 的最短路径 A→E→I→J→K→L→P，这条路径在枚举法中占总流量的 44.7%；Dial Single-pass 算法参与计算的路径数共有 4 条，这 4 条路径在枚举法中占总流量的比例为 45.7%；TPDial Single-pass 算法参与计算的路径数共有 12 条，这 12 条路径在枚举法中占总流量的比例为 96.0%；TPDial Double-pass 算法参与计算的路径数共有 17 条，这 17 条路径在枚举法中占总流量的比例为 98.6%。可以看出 TPDial Double-pass 算法参加计算的路径数是最多的，而且虽然只有 17 条路径参加计算，远远少于枚举出来的 184 条，但这仅占总路径数 9.2%的 17 条路径却是最重要的，因为有 98.6%的流量分配在这 17 条路径上，这同时也反映了 TPDial Double-pass 算法计算的高效性。

此外，可以看出这四种算法都包括了最短路径，而且 TPDial Double-pass 算法的路径集包含了 TPDial Single-pass 算法的路径集，TPDial Single-pass 算法的路径集包含了 Dial Single-pass 算法的路径集，而 Dial Single-pass 算法的路径集又包含了 Dial Double-pass 算法的路径集。

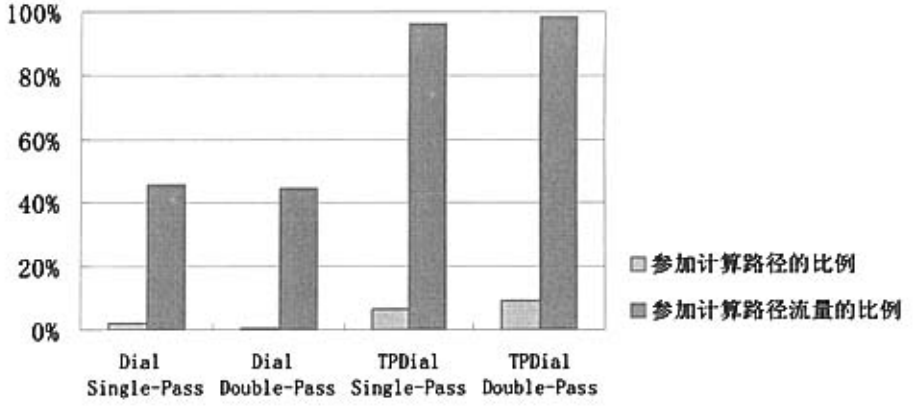


图 3-8 各种算法的路径集比较

(3) 计算的精度。整个网络选取相对误差公式

$$\varepsilon = \sqrt{\frac{\sum_{k=1}^{N_{arc}} (x_k - x_{k_0})^2}{\sum_{k=1}^{N_{arc}} x_{k_0}^2}} \tag{3-7}$$

其中 N_{arc} 为网络的路段总数，x_k 为各算法算出的路段交通流量，x_{k_0} 为枚举法的路段交通流量。

通过计算，Dial Double-pass 算法的误差最大，为 66.4%；TPDial Double-pass 算法的误差最小，为 1.1%；而 Dial Single-pass 算法和 TPDial Single-pass 算法的误差分别为 66.0%和 3.1%。可以看出 Dial 算法的误差比较大，很难应用到实际工程，而基于拓扑处理的新算法则很好地控制了误差。

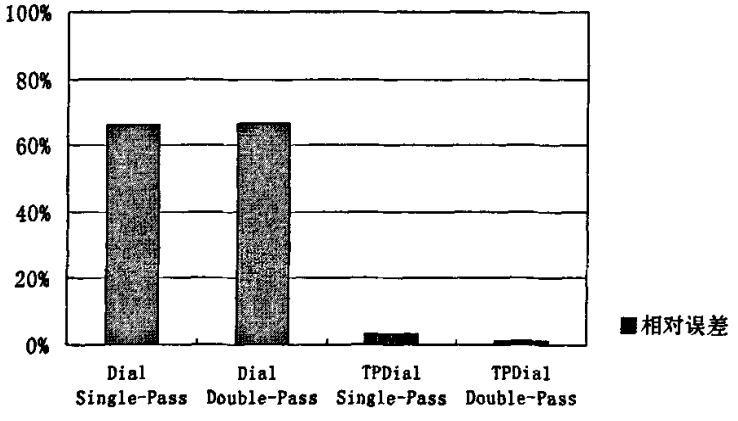


图 3-9 各种算法的相对误差比较

第 4 章 基于拓扑处理的 Logit 网络加载算法的分析与证明

4.1 TPDial 算法和 Logit 模型的等效证明

在 TPDial Single-pass 算法中，注意到路段$(i \to j)$的似然值和出行者在点 i 选择路段$(i \to j)$的概率成正比，因此出行者选择某条路径 k 的概率和路径上所有路段的似然值的乘积成正比，也就是说出行者选择某条联接起点 r 和终点 s 的路径 k 的概率为：

$$\Pr(k) = \Psi \cdot \prod_{i \to j} [L_{(i \to j)}]^{\delta_{i \to j,k}^{rs}} \tag{4-1}$$

其中Ψ是比例常数；如果路段$(i \to j)$在路径 k 上则$\delta_{i \to j,k}^{rs} = 1$，否则$\delta_{i \to j,k}^{rs} = 0$。

由$L_{(i \to j)} = \exp\left[\theta \cdot \left(r_{(j)} - r_{(i)} - t_{(i \to j)}\right)\right]$代入上式得：

$$\begin{aligned}
\Pr(k) &= \Psi \cdot \prod_{i \to j} \exp\left[\theta \cdot \left(r_{(j)} - r_{(i)} - t_{(i \to j)}\right) \cdot \delta_{i \to j,k}^{rs}\right] \\
&= \Psi \cdot \exp\left[\theta \cdot \sum_{i \to j} \left(r_{(j)} - r_{(i)} - t_{(i \to j)}\right) \cdot \delta_{i \to j,k}^{rs}\right] \\
&= \Psi \cdot \exp\left[\theta \cdot \sum_{i \to j} \left(r_{(j)} - r_{(i)}\right) \cdot \delta_{i \to j,k}^{rs}\right] \cdot \exp\left[-\theta \cdot \sum_{i \to j} t_{(i \to j)} \cdot \delta_{i \to j,k}^{rs}\right]
\end{aligned} \tag{4-2}$$

又∵ $$\sum_{i \to j} \left(r_{(j)} - r_{(i)}\right) \cdot \delta_{i \to j,k}^{rs} = r_{(s)} - r_{(r)} = u_{rs} \tag{4-3}$$

$$\sum_{i \to j} t_{(i \to j)} \cdot \delta_{i \to j,k}^{rs} = c_k^{rs} \tag{4-4}$$

式中，u_{rs}表示从起点 r 到终点 s 的最短路径长度。

∴ $$\Pr(k) = \Psi \cdot \exp(\theta \cdot u_{rs}) \cdot \exp(-\theta \cdot c_k^{rs}) = \Psi \cdot \exp[\theta \cdot (u_{rs} - c_k^{rs})] \tag{4-5}$$

由于所有路径被选择的概率之和为 1，即

$$\sum_k \Pr(k) = 1 \tag{4-6}$$

故有

$$\Psi = \frac{1}{\sum_l \exp[\theta \cdot (u_{rs} - c_l^{rs})]} \tag{4-7}$$

∴ 路径 k 被选择的概率为

$$\Pr(k) = \frac{\exp[\theta \cdot (u_{rs} - c_k^{rs})]}{\sum_l \exp[\theta \cdot (u_{rs} - c_l^{rs})]} = \frac{\exp(-\theta \cdot c_k^{rs})}{\sum_l \exp(-\theta \cdot c_l^{rs})} \tag{4-8}$$

上式跟 Logit 模型定义是相同的，这也证明了新算法与 Logit 模型是等价的。

TPDial Double-pass 算法的等效性证明类似。

4.2 TPDial 算法复杂度分析

最短路径问题作为网络优化的一个基本问题，国内外学者已经在此做了大量的研究，它们在时间复杂度、空间复杂度、实际运行效率、应用范围以及分类体系都各具特色[24-27]。交通网络是最短路径算法应用的主要领域，而 Dijkstra 算法仍是交通网络应用的首选算法，其时间复杂度可以达到 $O(n^2)$。如果利用 k 叉堆或 Fibonacci 堆进行改进，则 Dijkstra 算法的时间复杂度可以降低为 $O(m\log_k n)$ 或 $O(m+n\log n)$。由于基于拓扑处理的 Logit 网络加载算法必须得到网络的拓扑序列，显然利用 Dijkstra 算法是无法得到网络的拓扑序列，最终还是必须通过拓扑处理得到拓扑序列，这也加大了算法的时间复杂度。

为了寻找适合该加载算法的最短路径算法，必须从加载算法的特点入手。在引入了文献[32]的最短路径算法后，对于改进拓扑处理法本身，不仅是一种高效的计算最短路径算法，而且可以得到“拓扑序列”的计算顺序，这里所说的“拓扑序列”是将有环图变成无环图后的拓扑序列。

对于 TPDial Single-pass 或 TPDial Single-pass 算法，设网络的结点数为 n，弧数为 m，不难分析得到通过对优先序列的改进[30][32]，算法的时间复杂度可以降低为 $O(m+n\log n)$。在无环网络中应用更可以提高到 $O(m+n)$，完全可以适用大型网络或动态交通分配的计算。

4.3 TPDial 算法与其他算法的比较

4.3.1 计算精度的比较

(1) 数学归纳分析

对于有向无环网络，由于 TPDial 算法采用的是拓扑遍历的算法，按照拓扑序列的计算顺序，保留了完整的网络结构，这使得最后的计算结果跟理论值是一致。

而对于双向（无向）网络，分析如下。

图 4-1 计算序列示意图

① **Dial Double-pass 算法**

其有效路段集合中包括的路段均满足 $r_{(i)} < r_{(m)}$ 且 $s_{(i)} > s_{(m)}$，所以 Dial Double-pass 算法中参加计算的有效路径集 P_{DD} 中每条路径的上游节点 i 和下游节点 m 的关系满足 $r_{(i)} < r_{(m)}$ 且 $s_{(i)} > s_{(m)}$。

② **Dial Single-pass 算法（只算 $r_{(i)}$）**

其有效路段集合中包括的路段均满足 $r_{(i)} < r_{(m)}$，所以 Dial Single-pass 算法中删除的路段比 Dial Double-pass 算法少，参加计算的有效路径集 P_{DS}^{r} 中每条路径的上游节点 i 和下游节点 m 的关系满足 $r_{(i)} < r_{(m)}$。故 Dial Single-pass 算法中参加计算的有效路径集中至少包括了 Dial Double-pass 算法中参加计算的有效路径集，即 $P_{DS}^{r} \supseteq P_{DD}$。

③ **TPDial Single-pass 算法（只算 $r_{(i)}$）**

在拓扑处理的过程中，对于待处理的集合 K 中节点 i，是选取 $r_{(k)}$ 最小的节点进行处理，即 $r_{(i)} = \mathrm{MIN}\{r_{(k)}\}$，$\forall k \in K$，而对于在节点 i 之后处理的节点 m，其 $r_{(m)}$

是在集合 K 中节点的基础上求解的，故有 $r_{(i)} \le r_{(m)}$，当且仅当 i 和 m 同在 K 中且 $r_{(i)} = r_{(m)}$ 时取等号。所以 TPDial SinglePass 算法中参加的有效路径集 P_{TS}^{r} 中每条路径的上游节点 i 和下游节点 m 的关系满足 $r_{(i)} \le r_{(m)}$。故 $P_{TS}^{r} \supseteq P_{DS}^{r} \supseteq P_{DD}$。

④ **TPDial Double-pass 算法（先算 $r_{(i)}$，再算 $s_{(i)}$）**

TPDial Double-pass 算法最后决定计算顺序的是第二次拓扑处理的过程，在第二次拓扑处理的过程中，对于待处理的集合 K 中节点 i，是选取 $r_{(k)}$ 最大的节点进行处理，即 $r_{(i)} = \mathrm{MAX}\{r_{(k)}\}$，$\forall k \in K$。如果处理（删除）离开 i 的弧段 $(i \to m')$ 满足 $r_{(i)} \ge r_{(m')}$，则跟 TPDial Single-pass 算法删除的弧段相同。

现在假设处理（删除）离开 i 的弧段 $(i \to m)$ 是 $r_{(i)} < r_{(m)}$ 的情况，说明 m 还未被遍历，不在集合 K 中，因为如果 m 和 i 同在集合中，由于是选取 $r_{(k)}$ 最大的节点进行处理，则先处理的是节点 m 而非节点 I，即 $r_{(i)} < r_{(m)}$ 的弧段 $(i \to m)$ 被保留。根据拓扑遍历的特点，要遍历到节点 m 则必须经过此时集合 K 中的某一节点，不妨假设为 j，$j \in K$，则有 $r_{(i)} \ge r_{(j)}$。所以 $r_{(i)} \le r_{(j)} < r_{(m)}$，也即对于路径 $i \to m \to j$ 存在路段 $(m \to j)$ 满足 $r_{(j)} < r_{(m)}$，该路段在 TPDial Single-pass 算法中是被删除的。所以在 TPDial Double-pass 算法中由于删除了路段 $(i \to m)$ 而删除了路径 $i \to m \to j$，但该路径在 TPDial Single-pass 算法中也必被删掉，因为 $(m \to j)$ 满足 $r_{(j)} < r_{(m)}$。换句话说，TPDial Double-pass 算法中没有的路径在 TPDial Single-pass 算法中也一定不存在。

又因为 TPDial Double-pass 算法中有可能出现上游节点 i 和下游节点 m 的关系满足 $r_{(i)} \le r_{(m)}$ 或 $r_{(i)} > r_{(m)}$。，这也使得其出弧集比 TPDial Single-pass 算法大。故 $P_{TD}^{rr} \supseteq P_{TS}^{r} \supseteq P_{DS}^{r} \supseteq P_{DD}$。

同理，可证 $P_{TD}^{ss} \supseteq P_{TS}^{s} \supseteq P_{DS}^{s} \supseteq P_{DD}$

综上所述，对于双向（无向）网络，可以得出在各种算法中参加计算的路径集是不一样的，但各个路径集存在着包含与被包含的关系。其中 TPDial Double-pass 算法的路径集包括了 TPDial Single-pass 算法的路径集， TPDial Single-pass 算法的路径集包括了 Dial Single-pass 算法的路径集，而 Dial Single-pass 算法的路径集又包括了 Dial Double-pass 算法的路径集。

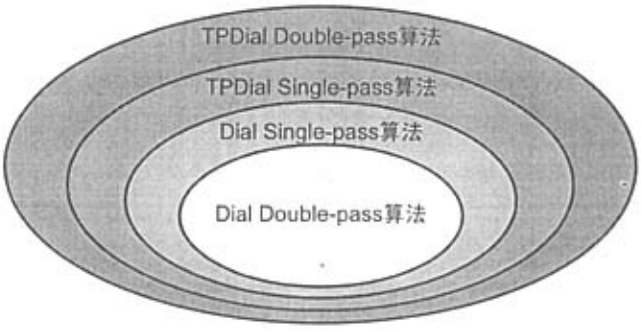


图 4-2 各种算法路径集的关系图

⑵ 实验结果分析

为了进一步验证各种算法的精度，选用一个节点数为 4×4 的棋盘状双向道路网(图 4-3)，这里 A 为起点，P 为迄点，OD 流量为 $q_{rs}=1000$，Logit 参数 $\theta=3.5/SP_{rs}$，其中 SP_{rs} 为起迄点间的最短路径长度。路段出行时间在给定区间的整数值上均匀分布进行随机取值，且同一节点对之间来回路段取相同值。图 4-4 给出了 10000 次随机计算的平均误差，理论值为枚举所有无环路径的计算结果。

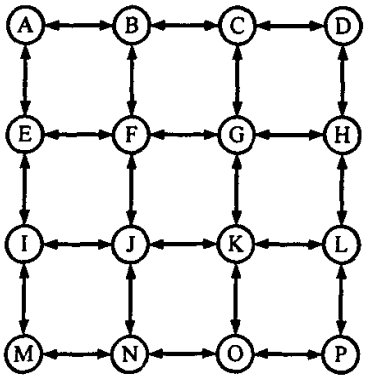


图 4-3 网络结构

计算结果表明，无论是 Single-Pass 还是 Double-Pass，TPDial 算法的精度优于 Dial 算法。当网络路段值的取值范围较窄时，TPDial Single-Pass 算法的明显

优于 Dial Single-Pass 算法，随着取值范围的扩大两者的差距变小。这和路网的结构有密切的关系，随着取值范围的增大，两者的合理路径集合趋向一致，两种算法的计算精度也趋于相等。需要说明的是算法误差较大一方面是 Logit 参数取值的影响，θ越小误差越大；另外理论上存在的一些无环路径，如 A→E→I→M→N→J→F→B→C→D→H→G→K→L→P，在实际上并不为出行者采用，因此在实际应用中各种算法的误差与实际相比要小一些。

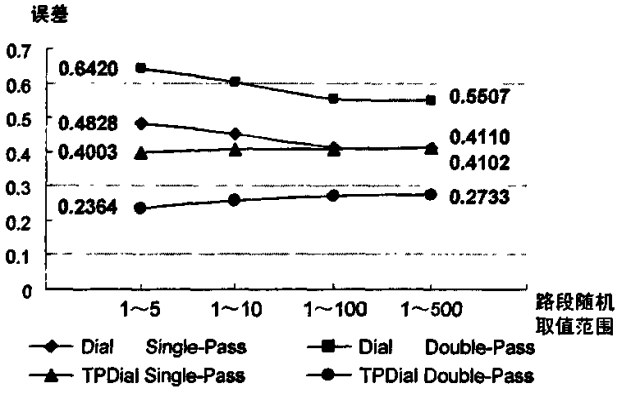


图 4-4 不同算法的误差比较

4.3.2 计算效率的比较

(1) 数学归纳分析

设网络的结点数为 n，弧数为 m，通过对优先序列的改进，Dial 和 TPDial 算法的时间复杂度均为$O(m+n\log n)$，而 TPDial 算法在无环网络中应用更可以提高到$O(m+n)$，完全可以适用大型网络或动态交通分配的计算，如表 4-1 所示。

表 4-1：不同算法各阶段的时间复杂度

步 骤	时间复杂度	Dial 算法		TPDial 算法	
		Single-pass	Double-pass	Single-pass	Double-pass
最短路径求 $R(i)$	$O(m+n\log n)$	√	√		
最短路径求 $S(i)$	$O(m+n\log n)$		√		
拓扑处理求 $R(i)$	$O(m+n\log n)$			√	√
拓扑处理求 $S(i)$	$O(m+n\log n)$				√
求路段似然值	$O(m)$	√	√	√	√
求路段权重	$O(m+n)$	√	√	√	√
求路段流量	$O(m+n)$	√	√	√	√

由表 4-1 可以看出最短路径的时间复杂度是整个算法中最复杂的一部分，也即最耗时间的一部分。它不但是计算似然值的前提，更是定义“合理路径”及排除了所有的环路的关键所在，这同样也是经典 Dial 算法的关键。所以最短路权的计算是基于 Dial 算法的 Logit 网络加载算法中不可或缺的环节，而最短路权的计算的时间复杂度又直接影响着算法本身的时间复杂度。

对于 Double-pass 算法，无论是 Dial 算法还是 TPDial 算法均需要计算两次最短路（拓扑处理），这比 Single-pass 算法多了一次。不同的是，在 Dial 算法中，Double-pass 算法计算两次最短路反倒使得“合理路段”的定义收紧了；而在 TPDial 算法中则是让更加的合理路段保留下来参与计算。

需要特别指出的是，对于多个 OD 对的分配，假设路网中起讫点的个数分别为 m_r 和 n_s，对于 Single-pass 算法而言，其需要完成的计算量为 m_r 或 n_s 次最短路的计算；而对于 Double-pass 算法而言，其需要完成的计算量则为 $m_r \cdot n_s$ 次最短路的计算；此时 Single-pass 算法的优势就体现出来了，其计算量仅相当于 Double-pass 算法的 $1/n_s$ 或 $1/m_r$。大中城市的交通网络往往包含数百个起讫点，OD 对数目更为巨大，对于一些需要实时处理的交通信息，Single-pass 算法的优势是明显的。

(2) 实验结果分析

本算例为 1000×1000 的棋盘状双向道路网（节点数为 1000000 个，弧段数为 3996000 条），来检验算法的效率。每条路段的阻抗都为 1，左上角节点为起点，右下角节点为迄点。OD 流量取 $q_{rs}=1000$，Logit 参数 $\theta=1$。由于各种算法的时间复杂度前面已作分析，所以此次只对 TPDial Single-pass 算法进行实验。图 4-5 给出了在 P4/2.4G，内存 1.5G 的计算机上的实验结果。

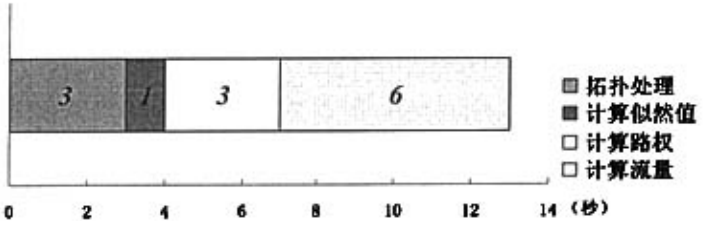


图 4-5 大型网络计算效率实验结果

由实验结果可以看出，拓扑处理、计算似然值、计算路权和计算流量的用时分别为 3 秒、1 秒、3 秒和 6 秒，共耗时 13 秒钟，这样的计算效率完全适用于大型网络。同时，该计算实例还表明计算最短路并不是以往研究中认为是最耗时间的一个部分。造成这一现象的原因是算例的最短路算法采用了极为有效的堆结构计算最短路[30][25]。以该 1000×1000 的方格网络为例，堆的平均长度为 667，但平均每次遍历深度仅为 4.66，即最终计算最短路的时间复杂度为 1000000×4.66，而在计算权重的时候为 1000000×4，计算流量时为 1000000×4×2 且需要做除法运算，因此计算流量的时间高于计算最短路。

第 5 章 算法的实现

5.1 交通网络的计算机表示方法

公路网络的交通分配是依托公路网络进行的，而公路网络一般都非常复杂，交通分配过程通常只能借助于计算机才能完成。因此，公路网络的计算机处理是进行交通分配最关键的前期工作，公路网络的计算机表示合理，能大大加快交通分配的运算速度，提高分配精度。

公路网络的计算机表示法包括公路网的节点表示、路段表示、邻接关系表示以及路权表示四个方面[9]。其中，公路网络节点是指有两条以上公路交叉的交叉口、公路几何要素发生重大变化的分界断面、多条公路交汇的枢纽；路段则是指网络中两个节点之间的公路路段，在公路网络分析中，认为同一公路路段内的几何要素（宽度、等级、车道数等）、交通信息（流量、车种组成、平均车速等）都是相同的；公路的网络邻接关系，也即拓扑关系，是计算机处理公路网络结构的基础，公路网络邻接关系通常用邻接表来表示；公路网络经过抽象后，各路段的路权（各种几何信息、交通信息、交通阻抗等）都是以邻接表所规定的顺序表示的。

5.2 交通网络的计算机运行结构

采用邻接表类型存储交通网络拓扑数据结构，在业界早已达成共识。而运行数据结构则纷繁多样，各有特色。而对于交通网络存储方法或运行结构的研究，大部分都是伴随着最短路算法一起研究的，因为交通网络是最短路径算法的主要应用领域，近年来也有很多的相关研究成果[36-39]。

基于各种运行结构的 Dijkstra 算法仍是交通网络最短路径算法的首选，如 ArcInfo 中的 Net-work 采用二叉堆优先级队列来实现 Dikstra 算法；Geostar 的网络分析采用快速排序的 FIFO 队列来实现 Dijkstra 算法；这几种算法均采用了拓扑邻接表的存储数据结构。

5.3 TPDial 算法的实现

5.3.1 采用邻接表存储交通网络

如前所述，采用邻接表类型存储交通网络拓扑数据结构，在业界早已达成共识。一般地，公路网络的邻接表可以作为一张关系表存放于数据库中。对于交通分配而言，一般需要节点表、邻接表和 OD 表三张表。这三张表的最基本的结构如表 5-1，表 5-2 和表 5-3 所示。

表 5-1 节点表结构

节点 ID	节点代号	X 坐标	Y 坐标	其他信息

表 5-2 邻接表结构

弧段 ID	起点	终点	阻抗	其他信息

表 5-3 OD 表结构

OD 号	起点	终点	OD 量	其他信息

图 5-1 基本的交通网络数据库设计示例

5.3.2 采用十字链表的运算结构

由于交通网络的节点数和道路数都比较大，所以在编程运算的时候都会采用更加优化的结构进行存储，这里所说的存储是运算过程中在内存的临时存储，所以也称之为运算结构。

交通网络简化后就是一个图，而对于图的运算结构一般有矩阵、邻接表、十字链表以及邻接多重表四种[40]。在此，主要介绍一下如何用十字链表来表示交通网络图。

在十字链表中，对应于有向图中每一条弧有一个结构体，对应于每个顶点也有一个结构体，而弧的信息也可以单独作为一个结构体，这里引入弧信息结构是为了解决同一对节点出现多条弧段的情况，即“重弧”的情况。各种结构体的结构如下所示：

表 5-4 节点结构

data	firstin	firstout

表 5-5 弧结构

tailvex	headvex	fistArcInfo	hlink	tlink

表 5-6 弧信息结构

ArcInfo	dist	next

节点结构由三个域组成：其中 data 域存储和节点相关的信息，如名称等；firstin 和 firstout 为两个链域，分别指向以该节点为弧头或弧尾的第一条弧。弧结构有五个域：其中 tailvex 和 headvex 分别指示弧尾和弧头这两个节点在图中的位置，链域 hlink 指向弧头相同的下一条弧，而链域 tlink 指向弧尾相同的下一条弧，而 fistArcInfo 指向多条重弧中的第一条，如果只有一条弧则指向该弧。弧信息结构有三个域：其中 ArcInfo 存储和弧段相关的信息，如名称等；dist 表示阻抗；而 next 指向多条重弧的下一条弧信息。例如图 2-1(*a*)用十字链表表示如图 5-2 所示。

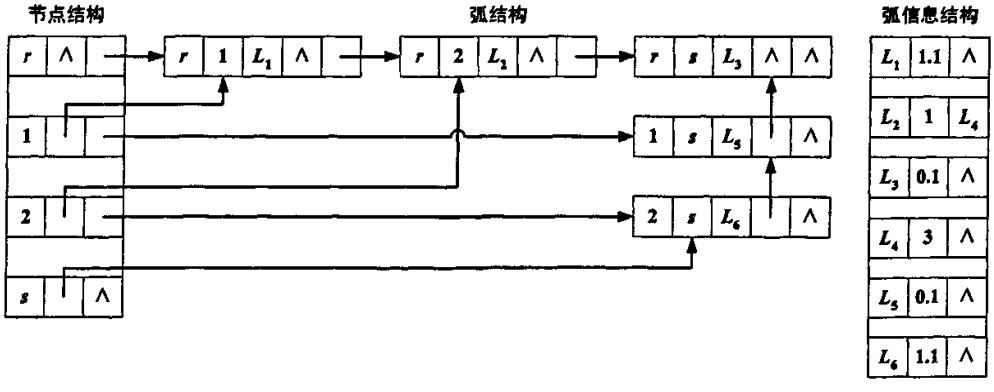


图 5-2 十字链表

5.4 Logit 网络加载分析软件的介绍

Logit 网络加载分析软件采用面向对象的设计方法，采用邻接表存储交通网络，用十字链表的运算结构。分析软件分为建模模块、运算模块和结果显示模块三部分组成，分别如图 5-3 所示。

(a) 软件主题

节点表：表

字段名称	数据类型	说明
NodeName	文本	节点名
NodeID	数字	节点ID
X	数字	X坐标
Y	数字	Y坐标
R_D	文本	Dijkstra算法求最短路r
S_D	文本	Dijkstra算法求最短路s
R_T	文本	拓扑处理算法求最短路r
S_T	文本	拓扑处理算法求最短路s

拓扑网络：表

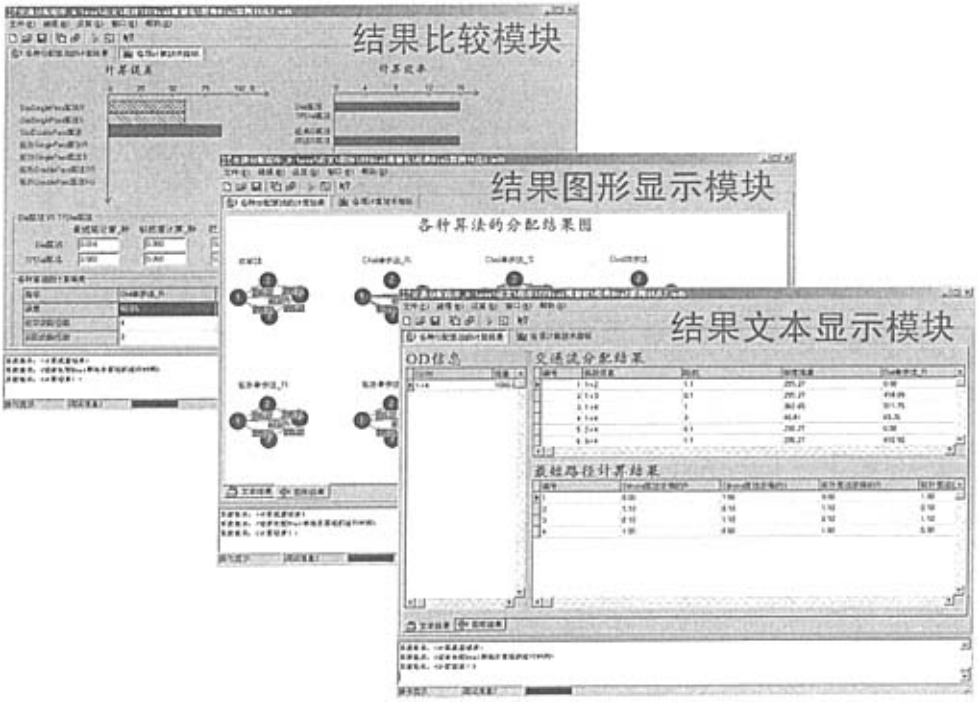
字段名称	数据类型	说明
LinkID	数字	弧ID号
LinkName	文本	弧代号
TailVEX	数字	起点
HeadVEX	数字	终点
Dist	文本	阻抗
Flow_STD	文本	流量_枚举法
Flow_DialSR	文本	流量_Dial Single-pass(r)
Flow_DialSS	文本	流量_Dial Single-pass(s)
Flow_DialD	文本	流量_Dial Double-pass
Flow_TPDialSR	文本	流量_TPDial Single-pass(r)
Flow_TPDialSS	文本	流量_TPDial Single-pass(s)
Flow_TPDialDSR	文本	流量_TPDial Double-pass(sr)
Flow_TPDialDRS	文本	流量_TPDial Double-pass(rs)
Type_DialSR	文本	状态变量_Dial Single-pass(r)
Type_DialSS	文本	状态变量_Dial Single-pass(s)
Type_DialD	文本	状态变量_Dial Double-pass
Type_TPDialSR	文本	状态变量_TPDial Single-pass(r)
Type_TPDialSS	文本	状态变量_TPDial Single-pass(s)
Type_TPDialDSR	文本	状态变量_TPDial Double-pass(sr)
Type_TPDialDRS	文本	状态变量_TPDial Double-pass(rs)

OD表：表

字段名称	数据类型	说明
OD	文本	ID号
Original	数字	起点
Destination	数字	终点
Flow	文本	总流量

(b) 数据库设计

(*c*) 建模模块



(*d*) 结果显示模块

图 5-3 Logit 网络加载分析软件

第6章 结 语

6.1 主要的研究结论

城市交通网交通分配是城市交通规划的一个重要组成部分，通过交通量分配所获得的路段交通量资料是检验城市交通规划是否合理的主要依据。随机交通分配可以反映出行者对不同路径的认识误差，分析出行者对不同路径的选择概率，从而对出行者的路径选择行为进行分析。Probit 型随机交通分配模型目前只能用蒙特卡罗仿真的方法来进行，并难以获得出行者对不同路径的选择概率；而 Logit 型随机交通分配模型由于存在分析算法，可以针对不同要求进行有效的分析，特别是对出行者的动态选择行为分析极为有用，因而得到了研究者广泛的重视。Dial 于 1972 年提出了一种算法，使得 Logit 的分析求解成为可能。Dial 算法虽然计算效率高，但其对“合理路径”的定义过于严格，导致了分配结果中一些路径阻抗较小的线路没被使用，而路径阻抗较大的线路反倒被使用的不合理现象，限制了 Logit 模型在实际中的应用。

本文对 Dial 算法进行回顾，分析其不足产生的原因，并在此基础上提出一种基于拓扑排序的改进算法。新算法根据进行拓扑处理的次数又细分为 TPDial Single-pass 算法和 TPDial Double-pass 算法。

新算法具有以下主要的优点：

(1) 高效性。新算法的高效性表现在两方面，一方面是用少量的路径集得到了与理论值非常接近的交通分配结果；另一方面是算法的时间复杂度均为 $O(m+n\log n)$，而 TPDial 算法在无环网络中应用更可以提高到 $O(m+n)$，完全可以适用大型网络或动态交通分配的计算。特别是对于多个 OD 对的分配，TPDial Single-pass 算法具有更好的计算优势。

(2) 准确性。新算法克服了 Dial 算法的缺陷，对于无环网络的应用可以得到与理论值完全一致的计算结果。而对于双向网络的应用，新算法放宽了“合理路径”的定义，引入拓扑计算序列，使得计算精度得到了大幅度的提高。特别是 TPDial Double-pass 算法，虽然比 TPDial Single-pass 算法多计算了一次最短路，

但其效果跟 Dial Double-pass 算法不同，Dial Double-pass 算法计算两次最短路反倒使得“合理路段”的定义收紧了；而 TPDial Double-pass 算法则是让更多的合理路段保留下来参与计算，进而提高整体的计算精度。

(3) 合理性。新算法将拓扑处理和最短路径求解进行无缝地结合，而不再重复进行最短路的计算，设计合理又大大提高运算的效率。而对于“合理路段”的选择，新算法克服了 Dial 算法“算术式”的定义，而采用了根据拓扑结构进行“动态式”的定义；淘汰了 Dial 算法根据节点最短路大小而确定的“数字式”计算序列，引入了根据网络结构而确定的“拓扑式”计算序列。

(4) 稳定性。新算法不存在高阶矩阵求逆和迭代法求解所带来的不收敛问题，具有相当稳定的效果。

(5) 实用性。新算法的设计采用了面向对象的方法，容易通过计算机进行实现，进而为大型网络的应用以及对于一些软件应用平台的移植提供了便利。

6.2 问题与讨论

本文是对 Logit 网络加载算法的研究采用拓扑处理方法进行的一个尝试，对此研究还处于起步阶段，加之研究学习时间的限制，有些研究未能进一步开展。本文主要存在以下几点问题：

(1) 由于没改变 Logit 模型，故 Logit 模型本身的缺陷仍存在。基于拓扑处理的 Logit 网络加载算法是采用动态处理的方法，该方法能否用于克服 Logit 模型的缺陷将有待进一步研究。

(2) 本文对于分配结果的讨论仅仅限于跟枚举法的对比讨论，而不是通过实际的交通量调查的结果进行对比讨论，故讨论结果可能跟实际会有些偏差。但新算法的提出还是对于 Logit 模型的应用带来了新的尝试。由于调查数据的限制，有关 Logit 模型参数θ的讨论也没能进一步展开，这将在今后的实践中不断加以完善。

(3) 对于新算法的应用，由于时间的限制，没有移植到一些交通的商业软件平台（如 TransCAD、Paramics 等等）上实验，以及针对拥挤交通网络进行试验，故在研究方法上仍需要进一步的探讨。

参考文献

[1] 王炜, 徐吉谦, 杨涛, 李旭宏等著. 城市交通规划理论及其应用[M]. 南京: 东南大学出版社, 1998

[2] 刘灿齐编著. 现代交通规划学[M]. 北京: 人民交通出版社, 2001

[3] 王炜, 徐吉谦, 杨涛, 李旭宏等著. 城市交通规划[M]. 南京: 东南大学出版社. 1999

[4] Meyer D., Urban Transportation Planning[M], Mcgraw-Hill Book Company, New York. 1984

[5] Carey M., The dual of the traffic assignment problem with elastic demands[J]. Transportation Research, 1985, 19B(3): 227-237

[6] Wardrop J.G., Some theoretical aspects of road traffic research[C]. Proc. Inst. Of Civil Eng., 1952, Part II 1(2): 325-378

[7] Beckmann M. J., Mcguire C. B., Winsten C.B., Studies in the Economics of Transportation[M]. Yale University Press, New Haven, 1956

[8] LeBlanc L. J., Morlok E. K., Pierskalla W. P., An efficient approach to solving the road network equilibrium traffic assignment problem[J], Transportation Research, 1975, 9B(3): 308-318

[9] 王炜, 邓卫, 杨琪等著. 公路网络规划建设与管理方法[M]. 北京: 科学出版社. 2001

[10] 邵春福主编. 交通规划原理[M]. 北京: 中国铁道出版社. 2004

[11] Merchant D. K., Nemhauser G. L., A model and algorithm for dynamic traffic assignment problems[J]. Transportation Science, 1978, 12: 183-199

[12] Merchant D. K., Nemhauser G. L., Optimality conditions for a dynamic traffic assignment problems[J]. Transportation Science, 1978, 12: 200-207

[13] Ho J. K., A successive linear optimization approach to dynamic traffic assignment problem[J]. Traffic Science, 1980, 14:295-305

[14] Ho J. K., Solving the dynamic traffic assignment problem on a hypercube multicomptuer[J]. Transportation Research, 1990, 24B:443-451

[15] Carey M., Optimal time-varying flows on congested network[J]. Operations Research, 1987, 35: 58-69

[16] Luque F. J., Friesz T. L., Dynamic traffic assignment considered as a continuous time optinal control problem[C]. PIMS/ORSA Joint National Meeting, Washington, 1980, 5-7

[17] Wie B. W., Friesz T. L., Toblin R. L., Dynamic user optimal traffic assignment on congested multidestination networks[J]. Transportation Research. 1990, 24B: 431-442

[18] Ran B., Boyce D. E., LeBlanc L. J., A new class of instantaneous dynamic user-optimal

traffic assignment models[J]. Operations Research, 1992, 41: 192-202

[19] Ran B., Boyce D. E., LeBlanc L. J., Dynamic user-optimal route choice models based on stochastic route travel times[C]. The Second Int. Capri Seminar on Urban Traffic Networks, 1992

[20] Papageorogion M., Mayr R., Comparison of direct optimization algorithm for dynamic network flow control[J]. Optimal Control Application and Methods, 1988, 9:175-185

[21] Janson B. N., Dynamic traffic assignment for urban road network[J]. Transportation Research. 1991, 25B: 143-161

[22] Dial R. B., A probabilistic multipath traffic assignment model which obviates path enumeration [J]. Transportation Research, 1971, 5: 83-111.

[23] Sheffi, Y., Urban Transportation Network[M]. Massachusetts Institute of technology, Newton.1984

[24] LI Jun, Kawakamis., Which path is reasonable: A paradox in Dial's algorithm[C]. Proceedings of JSCE 3th International Summer Conference. 2001.

[25] Bell G. H., Alternatives to Dial's LOGIT Assignment Algorithm[J]. Transportation Research, 1995, 29B: 287-296.

[26] Wong S.C., On the convergence of Bell's logit assignment formulation[J]. Transportation Research. 1999, 33B: 609-616.

[27] Akamatsu T., Cyclic Flows, Markov Process and Stochastic Traffic Assignment[J]. Transportation Research, 1996, 30B: 369-386.

[28] 李军, 聂佩林, 余志. 全路径 logit 随机分配模型的求解算法[J]. 中山大学学报(自然科学版). 2004, 5: 124-126

[29] Dijkstra E.W., A note on two problems in connection with graphs [J]. Numer. Math., 1959, 1:269~271

[30] Fredman M.L., Tarjan D.E., Fibonacci heaps and their uses in improved network optimization algorithms [J]. J. Assoc. Comput. Mach., 1987, 34: 596-615

[31] Ran B., Boyce D. E., Modelling Dynamic Transportation Networks: An Intelligent Transportation System Oriented Approach[M]. Springer, Berlin, 1996

[32] 周玉林. 单源最短路径问题的改进算法[J], 上饶师范学院学报, 2001, 3: 18-22

[33] 辛松歆, 李军. 一种改进的Logit型多路径交通分配算法[J]. 中山大学学报(自然科学版). 已接收

[34] 李军, 辛松歆, 蔡铭. 基于拓扑遍历的 Logit 交通分配算法[J]. 中国公路学报, 2005, 4: 87-90

[35] Li Jun, Xin Songxin, Liu Chunlu, AN ALGORITHM FOR LOGIT NETWORK LOADING PROBLEM BY TOPOLOGICAL SORTING[C]. Proceedings of the Eastern Asia Society for

Transportation Studies, 2005, Vol. 5: 1209-1217

[36] Deo N., Pang C. Y., Shortest Path Algorithms: Taxonomy and Annotation [J]. Networks, 1984, 14: 275-323.

[37] Cherkassky B.V., Goldber A.V., Radzik T., Shortest Paths Algorithms: Theory and Experimental Evaluation [J]. Mathematical Programming. 1996, 73: 129-174.

[38] Zhan F. B., Noon C. E., Shortest Path Algorithms: An Evaluation Using Real Road Networks [J]. Transportation Science, 1998, 32(1): 65-73.

[39] 陆峰, 最短路径算法分类体系与研究进展[J]. 测绘学报, 2001, 3: 269-275

[40] 严蔚敏, 吴伟民. 数据结构[M]. 北京:清华大学出版社. 1996

附 录

1. 硕士在读期间发表的论文

[1] 李军，辛松歆，蔡铭. 基于拓扑遍历的 Logit 交通分配算法[J]. 中国公路学报，2005，4: 87-90（EI 收录，收录号为：05469481554）

[2] Li Jun, Xin Songxin, Liu Chunlu, AN ALGORITHM FOR LOGIT NETWORK LOADING PROBLEM BY TOPOLOGICAL SORTING[C]. Proceedings of the Eastern Asia Society for Transportation Studies, 2005, Vol. 5: 1209-1217

[3] 辛松歆，李军. 一种改进的 Logit 型多路径交通分配算法[J]. 中山大学学报(自然科学版). 已接收

2. 参加的项目

2004.3　　风资源评估小组和公交换乘平台开发小组交叉学习 GIS 开发

2004.7　　参与交通模型仿真平台交通调查

2004.9　　参与 GPS1 研究小组前期准备工作

2004.3 至今　　交通规划与政策小组、交通仿真小组学习研究交通规划和仿真平台建设

3. 参加的学术会议

2004.6　　参加第四届交通运输领域华人学者国际会议

2005.11　　参加全国智能运输系统标准化技术委员会年会

4. 参加的创新性活动

2004.9　　参加首届全国研究生数学建模竞赛并获得全国一等奖

获奖论文为：《研究生录取的最大匹配模型》

2005.9　　参加第二届全国研究生数学建模竞赛并获得全国一等奖

获奖论文为：《城市出租车交通规划综合模型》

后 记

时光荏苒，岁月如梭，康乐园校道两旁的羊蹄甲开了又谢，谢了又开，转眼间，三年的研究生生活匆匆而过，心中半是喜悦半是不舍。我衷心感谢中山大学七年来对我的培育，感谢各位良师对我的教诲和指导，感谢诸多朋友的关心和帮助……

三年前我有幸师从李军老师，从事城市交通政策与规划方面的学习和研究。这三年来，李老师给予我一切可能的机会进行学习和实践，从学做事到学做人，我收获的不仅仅是知识，更获益于李老师在生活上的谆谆教诲。李老师严谨求实的治学态度，勇于创新、无私奉献的敬业精神，宽厚坦诚的待人气度，积极乐观的人生态度，都让我终生受用。承蒙老师们的厚爱，自己得以在短短三年间，在各方面都得到了较大的进步。

在掩卷之际，我由衷地感谢这些给过我无数帮助的人们：感谢导师李老师对我学术上无私的传授和生活上无微不至的关怀，从论文选题到修改，李老师都给予我耐心的指导；衷心感谢工学院和智能交通研究中心各位领导和老师，特别是余志教授对我的教导和关怀；智能交通研究中心各位老师渊博的知识、严谨的治学态度、丰富的学术思想以及认真热情的处世态度都给了我莫大的启发和帮助，使我终生受益。

感谢在一起学习的何兆成老师，曾雪兰师姐，龚峻峰师兄以及智能交通研究中心其他同学和朋友对我的帮助和鼓励。

在这三年的研究生生活中，得到了在一起学习生活的交通信息工程及控制专业2004级的同学、朋友们给我的无私的帮助和关心，在此一并致谢。

最后，还要感谢我的家人和女友在学习和生活上无限的帮助、照顾、体谅和包容！

在中山大学的七年学习生活是我人生中最宝贵时光，我将珍藏这份美好的回忆，永远铭记这七年的师生情、同学谊。

辛松歆

二零零七年五月于康乐园

原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，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。对本文的研究作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：李松敏

日　期：　2007 年 5 月 24 日