MATHEMATICAL PROGRAMMING ALGORITHMS FOR EQUILIBRIUM ROAD TRAFFIC ASSIGNMENT

by

Seungjae Lee

A thesis submitted to the University of London

for the degree of Doctor of Philosophy

Centre for Transport Studies

University College London

January 1995

To my parents

ABSTRACT

The equilibrium approach to representing interactions between the supply and demand sides of traffic assignment has been used widely in the estimation of traffic flows on road networks. Although this approach is quite reasonable, there is a considerable gap between the observed and modelled values of cost and flow. This gap can be reduced by relaxing some of the restrictive assumptions behind the models used in order to enhance their realism.

This study investigates the solutions of various advanced road traffic assignment models. Priority and signal controlled junctions are modelled in traffic assignment in order to enhance the realism of junction analysis. A multiclass assignment is modelled to represent different groups of users. These problems are known to be non-separable because traffic cannot be segmented in such a way that the costs incurred by any one segment vary only with the flow within that segment. Existence, uniqueness and stability properties of solutions to these problems are investigated. These analyses are important to know the reliability and repeatability of any solutions that are calculated. Analyses of these properties lead to some guidelines for using these detailed models. A number of new solution algorithms are developed to solve the resulting traffic assignment problems. These algorithms belong to the general category of simplicial decomposition which solves the problem by dividing it into two subproblems: a linear and a master subproblem which are solved alternately. One of the advantages of these algorithms is that they operate in a lower dimensional space than that of original feasible region and hence allow large-scale problems to be solved with improved accuracy and speed of convergence. These improved algorithms give many choices to the traffic management studies.

Two substantial networks have been used to compare the performance of new algorithms on the various models developed. They have performed favourably by comparison with existing algorithms. A small example network has been used to investigate existence, uniqueness and stability properties using the models. In a priority controlled model, a unique stable solution has been obtained using the model whilst in a signal controlled model, multiple and unstable solutions have been obtained. In a multiclass model, a unique solution has been obtained in terms of the total class flow whilst multiple solutions have been obtained in terms of each class flow. These results correspond well to the theoretical analyses of these models, which has shown to have indeterminate behaviour and by the nature of these models assumed, the degree of non-separability is ordered according to priority controlled, multiclass and signal controlled models.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to Dr. Ben Heydecker for his great supervision and continuing encouragement throughout the period of this study. It would be impossible to complete this study without his patient and valuable supervision.

I would like to thank Professor Richard Allsop and Dr. J. D. Addison for their discussions and support from time to time. In particular, I am indebted to Dr. J. D. Addison who read a final draft carefully and made many helpful comments to improve it. The discussions with Dr. Yasuo Asakura of Ehme University and Dr. Takamas Akiyama of Kyoto University were appreciated in the early stages while they spent their sabbatical periods at the Center for Transport Studies.

Thanks to privileged comments from both Mr. M. J. Smith of University of York and Dr. S. Powell of London School of Economics, the thesis improved greatly.

Thanks also go to all the members at the Centre for Transport Studies who have been supportive during the period of this research.

Financial support of this research by Deans' Scholarship of University College London is gratefully acknowledged.

My greatest debt is to my father and mother for providing everything from moral to financial supports. I also would like to thank my sisters for continuing encouragement.

Finally, I would like to thank the Father in heaven for knowing him and guiding me.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

1.1 GENERAL BACKGROUND

As urban infrastructure grows rapidly, the role of transportation systems becomes more important to give mobility and accessibility. Its importance as a basic element of urban areas is recognised most dramatically when it does not operate properly. A requirement for comprehensive transportation planning has therefore emerged in order to use urban areas effectively and efficiently.

The transportation planning process conventionally consists of four stages: trip generation, trip distribution, modal split and traffic assignment. This process is directly related to the behaviour of individuals decisions making. Trip generation is a stage to estimate if travellers make a trip for some purposes. The trip distribution stage estimates a demand for travellers going from some places (or, origins) to others (or, destinations). The modal split stage estimates the usage of the various modes of transportation that are available for each origin-destination pair. Finally, the traffic assignment stage estimates traffic flows and conditions in transportation systems for each mode. The results of traffic assignment give a transportation planner or a traffic engineer specific information relating to estimated traffic flows and travel costs which can then be used to support decisions on policies.

The road traffic assignment process estimates the flow pattern on the links of a road network, given the demand associated with each origin and destination pair of the network, and the travel cost functions for each link of the network. The equilibrium approach to representing interactions between the supply and demand sides of the transport process has been used widely for traffic assignment.

Although this approach is quite reasonable, there is often a considerable gap between observed and modelled values of costs and flows. This gap can be reduced by relaxing some of the restrictive assumptions behind the models that are used in order to enhance their realism. Assumptions relating to the demand side have been relaxed by incorporating some dynamic elements into the analysis (see, Vythoulkas,1990 and others). Assumptions concerning the supply side have been relaxed by considering the detailed modelling of traffic behaviour that is expressed by cost functions. In this thesis, a relaxation of these latter assumptions is investigated.

Conventionally, link travel times are represented by flow-delay functions. However, road users choose their routes so as to minimise their whole travel costs, and this typically involves passing through one or more junctions. Details of junction delays are therefore required to represent traffic movements. Another feature that can be introduced to improve realism in a conventional traffic flow-delay function is that of the multiple user class, each with distinct behaviour. A concept of multiple user classes can be distinctive characteristics of each class such as physical sizes and types of vehicles, cost function types, and road network restrictions. The resulting enhanced cost functions can lead to more accurate estimates of flow by representing more realistic congestion effects and interactions on route choice in the traffic assignment process.

As increasingly detailed cost functions are introduced, the problem of monotonicity of the cost function and hence existence of a unique stable solution arises. The condition of separability, that the cost on the link is only a function of the traffic flow on that link itself, is violated when detailed junction models are incorporated. Thus, non-separability arises because of influential interactions between different elements of traffic on links either when multiple user class behaviour is modelled, or at junctions

when priorities passing through junction or traffic responsive signal controls are modelled. This non-separability can result in some problems because the resulting detailed cost functions do not normally show the property of monotonicity required to guarantee the existence of a unique stable solution and the convergence of solution algorithms.

The good behaviour of a traffic assignment model depends on the existence of a unique stable solution: if a model has a unique stable solution, we say that it has good behaviour. Conditions of good behaviour have been studied by various authors. A sufficient condition for good behaviour due to Smith (1979) is that the demand be within capacities, and the Jacobian matrix of the cost function be positive definite or equivalently that the cost function be strictly monotone. A corresponding necessary condition due to Heydecker (1983) is that the Jacobian matrix of the cost functions be everywhere a P matrix (the condition for a P matrix is that the determinants of all principal submatrices are everywhere non-negative).

The class of simplicial decomposition algorithms has been shown to be promising to solve the equilibrium road traffic assignment problem (Larsson and Patriksson, 1992 and others). These algorithms follow the simplicial decomposition principle of Danzig and Wolfe (1960). They consist of two procedures: a linear subproblem and a master subproblem which the algorithm solves alternately. The linear subproblem is obtained as a linear approximation to the original problem and corresponds to a shortest path search. The solutions to successive linear subproblems are used as the extreme points of a simplex. The master subproblem then solves the original nonlinear problem over the convex hull generated by this simplex. One of the advantages of simplicial decomposition algorithms is that they solve the nonlinear problem in a space of lower dimension than that of the original feasible region and hence allow large-scale network

problems to be solved using advanced techniques with improved properties and speed of convergence.

1.2 OBJECTIVES

The first objective of this study is the modelling of equilibrium road traffic assignment problems with non-separable cost functions. Non-separability arises because of influential interactions among different elements of traffic on links or at junctions. Link and junction interactions arise respectively when different classes of road users share a link, and when junction priorities or signal controls are modelled. In this thesis, multiclass traffic assignment is studied as a link interaction case, and traffic assignments with priority and signal control junction models are studied as junction interaction cases. In each of these non-separable cases, travel times on several links depend on the flow on those links and also on the flow on other links.

This objective leads to the development of solution methods for equilibrium road traffic assignment problems with non-separable cost functions. First, four new simplicial decomposition algorithms are developed to solve the separable problem. Second, a diagonalisation solution procedure for non-separable cost, which solves a sequence of separable cost problems representing the non-separability property, is adopted for use in conjunction with these simplicial decomposition algorithms. Algorithms of this kind are of immediate importance because they are useful tools for the study of detailed analysis of transportation systems.

The importance of these extensions stems from the recognition that separable traffic assignment models fail to capture essential features of traffic congestion by not

representing real phenomena. In reality, the performance of traffic on each link is affected by vehicles of other classes, oncoming traffic and different priority rules at junctions and signal control policies. Different kinds of vehicles have different occupancy rates, speeds, values of time, route choice criteria and cost functions. The model developed here has the potential to play an important role in the more accurate estimation of travel behaviour and conditions to help to develop, manage and evaluate transport planning policies to alleviate traffic congestion.

The second objective is to analyse some properties such as existence, uniqueness and stability of the solutions of these models in relation to the theoretical conditions for good behaviour. These analyses show that if a cost function of a traffic assignment model satisfies the theoretical conditions for good behaviour, that model has a unique stable solution. These solution properties are related to the details of the delay formulae, which include topological and geometrical characteristics of the junctions and the effects on the cost function. The details of the delay formulae are caused by priorities of interacting vehicular movements, signal control policies and different perceptions of travel costs for different classes. These analyses are important to know the reliability and the repeatability of calculated traffic equilibria, and thus lead to some guidelines for using detailed models.

1.3 OUTLINE OF THIS STUDY

In chapter 2, fundamentals of traffic assignment are described. The historical development of traffic assignment is surveyed briefly. The traffic assignment process is described in terms of demand and supply interactions. Network representations for detailed modelling are explained in terms of nodes and links. Cost functions to represent

effects of congestion are reviewed to select appropriate ones. Beckmann's formulation is discussed in terms of equivalence conditions. The more powerful variational inequality formulation is then introduced. Conditions for good behaviour are represented in terms of the Jacobian matrix. Finally, some general solution algorithms for solving nonlinear problems are discussed.

In chapter 3, the simplicial decomposition principle is introduced in terms of a linear subproblem and a master subproblem. Existing algorithms including the Frank-Wolfe and the Schittenhelm algorithms are represented in terms of this principle. Four new algorithms are introduced and discussed. The performance of these algorithms are compared using a separable traffic assignment example.

In chapter 4, traffic assignment with priority controlled junction modelling is presented in terms of capacity calculation, cost function analysis, formulation and numerical analysis. In capacity calculation, the effects of junction geometry and priority rules are explained and the TRL PICADY capacity formulae are introduced. In cost function analysis, the steady-state Pollaczek-Khinchine formula and an extension to it are introduced. The conditions on the Jacobian matrix for good behaviour are tested, and some properties such as uniqueness and stability are investigated using a small network. Two larger numerical examples are used to analyse the efficiency of the various solution methods including the four new algorithms presented in this thesis.

In chapter 5, traffic assignment with signal controlled junction modelling is presented in terms of historical developments, signal optimisation problems, cost function analysis and numerical analysis. Existing methods for signal optimisation are reviewed. Parameters of optimisation are introduced and discussed. In particular, Webster's green time calculation method is described. In cost function analysis, Webster's

steady-state delay formula and an extension of it are introduced. The Jacobian matrix analysis for good behaviour is presented and compared with numerical results to analyse uniqueness and stability of the combined signal and traffic assignment using a small network. Two larger numerical examples are used to analyse the efficiency of the various solution methods.

In chapter 6, multiclass traffic assignment is presented in terms of historical developments, concepts of multiclass, cost function analysis, formulation and numerical analysis. The concept of multiclass assignment is presented in terms of its effects and implementation. Cost functions for multiple classes are described. The Jacobian matrix of this problem is analysed to study uniqueness and stability in the case of a small example network. Two larger numerical examples are used to analyse the efficiency of the various solution methods.

In chapter 7, a general summary of these various models is outlined. General discussion and conclusions are presented on the basis of the findings of the whole study. Finally, some possibilities for future studies draw this chapter to a close.

CHAPTER 2. FUNDAMENTALS OF TRAFFIC ASSIGNMENT

2.1 INTRODUCTION

This chapter reviews the fundamentals of the traffic assignment process and describes some existing solution methods. The historical developments of traffic assignment and solution methods are surveyed. The traffic assignment problem is described in terms of interactions between supply and demand. Network representations of supply side at various levels of details are introduced. Various formulations of this problem are described as each of a convex mathematical programme and variational inequality. The role and effect of cost functions in these formulations are reviewed. In particular, satisfaction of conditions for good behaviour is introduced in terms of the Jacobian matrix of the cost functions. Solution algorithms are briefly presented for each of the separable and non-separable cases.

2.2 HISTORICAL DEVELOPMENT OF TRAFFIC ASSIGNMENT

Traffic assignment is the interactive process between travel demand and transportation supply. Travel demand is people's (or users') desire to move from one place (or origin) to another place (or destination). Transportation supply is a set of facilities such as streets and junctions in road networks. This representation of travel demand and transportation supply induces the traffic flow pattern and travel cost as an equilibrium point. Wardrop's first principle (1952) expressed this kind of equilibrium as a state in which for each origin-destnation pair no user can reduce their costs by selecting a different route. Wardrop's second principle (1952) identifies a state in which the total users' travel cost is minimised with respect to choice of routes. Beckmann, McGuire and Winsten (1956) formulated equilibrium traffic assignment as a convex mathematical

programme using the concept of Wardrop's first principle.

The formulation of Beckmann et al (1956) assumed that the cost of using each link of the network varies only with the flow on that link so that there is no interaction between flows on different links. This is called the *separable* cost case. However, in practice there are many cases in which the flow on one link affects the cost of travel on others. This is called the *non-separable* cost case. Examples of the non-separability arise at a priority-controlled junction in which the capacity of the minor stream is a decreasing function of the major flows and hence the minor stream cost function is influenced by the major flows, and at signal-controlled junctions in which the green time is determined appropriately by the flows approaching them. There are also interactions between different vehicle classes on each link that can be represented in this way.

Non-separable cost models have attracted a large amount of researches into their properties, possible reformulations into better understood mathematical programming problems and algorithms for their solutions. The traffic assignment problem with link interactions was first formulated as a variational inequality by Smith (1979). He showed that Wardrop's conditions of user equilibrium are equivalent to a variational inequality. Aashtiani and Magnanti (1981) reformulated this condition as a nonlinear complementarity. However, these two formulations were proven to be essentially equivalent by Magnanti (1984). Smith (1979) proved that a sufficient condition for uniqueness and stability of traffic equilibria is that the Jacobian of the link cost function is positive definite. Heydecker (1983) presented a corresponding necessary condition for these properties.

An extended two-way street traffic assignment problem has been suggested by Dafermos (1971), who included a discussion of the existence, uniqueness, and stability of

the equilibrium flow pattern when there is an interaction between flows in one direction and flows in the opposite direction. Dafermos (1972), and Van Vliet, Bergman and Scheltes (1986) presented multiclass user equilibrium assignment formulations in which several classes of vehicles that have different performance and cost functions interact on each link. Braess and Koch (1979) proved the existence of equilibria in *asymmetric multiclass user assignment* in which the effect on cost c_a, on link a, of the flow v_b, on link b, is not the same as the effect on cost c_b, on link b, of the flow v_a on link a. This can be represented as:

$$\frac{\partial c_a}{\partial v_b} \neq \frac{\partial c_b}{\partial v_a}, \text{ for some } a \neq b \qquad (2.1)$$

They proved that if the individual cost functions in the multiclass user assignment are continuous and monotone, then there is at least one user equilibrium flow pattern. In an *elastic demand model*, the demand is a function of travel costs between each origin and destination. Dafermos (1981) presented a multimodal network equilibrium problem with elastic demand in which the link cost depends on the entire load pattern and the travel demands. Abdulaal and LeBlanc (1979) have studied combining modal split and equilibrium assignment as the more general case. Fisk and Nguyen (1981) proved the existence and uniqueness of an asymmetric two-mode (auto and public transit) equilibrium model. Daganzo (1983) studied stochastic network equilibrium with multiple vehicle types in which the users' perceptions of the cost are probabilistic.

Solution algorithms commonly applied to the traffic assignment problem use successive linear approximations to the objective function. The most well known is the Frank-Wolfe (F-W) algorithm (1956). Algorithms of this type include the PARTAN search direction (Luenberger, 1989, pp254-257), the modified search direction of Fukushima (1984), the modified step lengths of Weintraub, Ortiz and Gonzales (1985), Wolfe's away step (1974) and Holloway's extension (1974), among others. These

extensions have been derived to overcome the slow convergence of the Frank-Wolfe algorithm in its later stages.

The simplicial decomposition principle of Dantzig and Wolfe (1960) provides an important framework to describe many nonlinear algorithms. Solution methods belonging to this principle solve alternately two subproblems, a linear and a master one. The linear subproblem is to solve a linear approximation of the original problem: the solution of this corresponds to a vertex of the feasible region. In the master subproblem, the reduced-dimensional version of the original problem is solved in the convex hull generated by the solutions to all previous linear subproblems.

Von Hohenbalken (1975,1977) gave a theoretical background for simplicial decomposition methods. The convergence result obtained by Von Hohenbalken (1975) allowed for the use of column dropping, that is the possibility of removing extreme points from the feasible set if they have small weights in the current solutions. Hearn, Lawphongpanich and Nguyen (1984) proved finite convergence of a simplicial decomposition method with using column dropping in the case of the bounded feasible set. Sacher (1980) extended this bounded case to an unbounded feasible set. Shetty and Ben Daya (1988) proposed a similar algorithm to that of Sacher (1980) but in the master problem they use either a reduced gradient or a gradient projection method. In the master problem, Von Hohenbalken (1977) used Newton directions, whilst Pang and Yu (1984) approximated the master problem by a quadratic programme. Hearn et al (1984) used the projected Newton method of Bertsekas and Gafni (1982) to solve the master problem, Dembo and Tulowitzki (1988) used the PARTAN search direction and Larsson and Patricksson (1992) used either a reduced gradient method or an approximated Newton's method.

Decomposition schemes have been applied to the non-separable problem by Bertsekas and Gafni (1982), and Aashitiani and Magnanti (1981) using Newton type methods. Each of them used Lemke's algorithm (1965) to solve the linear subproblem. Smith (1983) used a column generation technique based on reformulation of the variational inequality as a non-convex mathematical programme.

One of the heuristic algorithms that has been used for the non-separable case is based on an iterative diagonalisation (or relaxation) procedure, where each iteration requires the solution of a full-scale separable user equilibrium problem (Fisk and Nguyen,1982). A proof of the convergence for this procedure is given by Dafermos (1982) under the condition that the cost function is strictly monotone. Another approach is a streamlined version of the first one; it performs a single iteration of the equilibrium solution procedure at each iteration of the diagonalisation procedure. This streamlined version has been found to be more efficient than the first one (Sheffi, 1985).

Van Vliet et al (1986) used a diagonalisation algorithm to solve traffic assignment with multiple user classes. The formulation of this allows user-class specific network variations to be represented, such as use restrictions of a network for some classes, and variations in the value of travel time and perceived travel cost. In this model, the cost incurred by travellers of each class depends on the flows of traffic in all classes on that link. Dafermos (1981) presented a Newton projection method for calculating multimodal equilibria within the diagonalisation framework.

2.3 GENERAL DESCRIPTION OF THE EQUILIBRIUM APPROACH

A transportation system consists of two components; a transportation supply and travel demand. The transportation supply is provided by a set of facilities for the users of

the transportation network. The travel demand is a number of users wishing to travel via the network. A representation of transportation supply and travel demand is necessary to compute the flow pattern which results from the interaction between them. The resolution of this interaction is known as traffic assignment.

2.3.1 Transportation supply

The transportation supply is a set of facilities which can be identified with the road network. The road network includes streets and junctions through which traffic moves. The term "network " is used to describe a structure that can be represented mathematically by two sets: a set of nodes (or, points or vertices) and a set of directed links (or, line segments or arcs) connecting them. A graph is used to represent the network mathematically: a directed graph is a pair of set G=(N, L), where N is a set of nodes and L is a set of directed links. Furthermore, the network representations of a road system have associated with their impedance functions that represent the cost of travel. Impedance which affects flows on the network can represent time, cost, disutility or other measures. When the flow involves people, the term "generalised cost" is used instead.

In transportation planning, the demand for travel is represented by dividing the study area into smaller subareas called *zones*. We assume that each of these zones is homogeneous. The size of zone can vary from a small block to a whole town. Each zone is represented by a special node called a *centroid.* Each centroid represents either or both a source or a sink where trips originate and terminate respectively. Once a set of centroids are defined, the desired movements through the network can be expressed in terms of an origin-destination matrix. This matrix indicates the demand per unit time for travel between each origin and destination (O-D) pair.

Flow-delay functions are used to represent travel costs c_a on each link a. It can be time taken to travel along the link or some more general notions of cost. Travel cost is a measure of transportation supply. In general, we express the travel cost for each link as a function of the vector of all link flows **v**.

Definition : supply feasibility

If there is an asymptotic capacity Q_a on link a, then the set of link flow vector **v** is *supply feasible* if $0 \leq v_a < Q_a$ for each $a \in L$. We denote the set of all supply feasible flows by δ. Note that if there is no such capacity, then all positive flows are supply feasible.

2.3.2 Travel demand

Travel demand represents the users' desire to travel. This can be categorised by trip purposes (work, study, business, leisure), modes (private car, public transport, on foot) and time of day. It is expressed numerically by the number of users wishing to travel per unit time between each origin and destination pair. If travel demand depends on the cost of travel between O-D pairs, it is described as elastic demand. Otherwise, travel demand is assumed to be a fixed.

Travel demand can be estimated using methods belonging to these categories:

1) Direct observation methods: using field observation, extensive surveys and interviews. Home interviews, roadside interviews, vehicle following methods and aerial photography methods belong to this category. All of these methods are expensive.

2) Indirect methods: using synthetic methods such as trip distribution models in the four stage transport planning process.

3) Simplified method: using partial traffic counts and conventional transport models in the four stage transport planning process. This is an inexpensive method based on synthetic mathematical principles.

Associated with each (o,d)$\in$B is a positive real number T_{od} called the *demand* per unit time for travel between O-D pairs (o,d) where B is a set of all O-D pairs. A path p is a set of links that connect a sequence of nodes without loop. We denote by P_{od} the set of all paths for O-D pair (o,d). The flow on each path p is denoted by t_p. The path cost C_p is the sum of the costs of the links that comprise it. The flow on link a is the sum of the flows on the paths that include the link. The link-path incidence matrix δ represents the relation between links and paths, where each row is associated with link a and each column with path p. The elements δ_{ap} of the row corresponding to link a take the value of 1 in the columns corresponding to the paths containing link a and the value of 0 in the other cases. Thus for link a and path p,

$$\delta_{ap} = \begin{cases} 1, & \text{if } a \in p \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

Definition : demand feasibility

The vector, **t** of the path flows t_p is *demand feasible* if for each O-D pair (o,d), the sum of flows on all paths, (P_{od}) for that od pair equals the demand T_{od} from origin o to destination d, and each path flow t_p is non-negative. This is represented mathematically as follows:

$$\sum_{p \in P_{od}} t_p = T_{od}, \quad \forall (o,d) \in B \tag{2.3}$$

$$t_p \geq 0 \qquad \forall\, p \in P_{od} \quad \forall (o,d) \in B$$

We denote the set of all demand feasible flows by $\mathcal{B}$. This demand feasibility condition defines a convex set because it is represented by a set of linear inequalities.

Definition: Convex set

A set $\mathcal{S} \in R^n$ is *convex* if the convex combination of any two points in $\mathcal{S}$ also belongs to $\mathcal{S}$. Mathematically,

$$\begin{aligned} &v_0, v_1 \in \mathcal{S}, \beta \in [0,1], \\ &v_2 = (1-\beta)v_0 + \beta v_1, \\ &\Rightarrow v_2 \in \mathcal{S}. \end{aligned} \tag{2.4}$$

2.3.3 Demand and supply interaction

Travel demand depends on travel costs whilst travel costs depend on flows. A mutual interaction between demand and supply is thus induced. This interaction has two simultaneous effects; the determination of demand as a function of travel cost from origin to destination, and the determination of a flow pattern due to the dispersion of demand.

Through this interaction between travel demand and transportation supply the system is in a state of equilibrium if the path flows obtained in the current iteration according to a principle of route choice give rise to travel costs that are the same as those in the previous iteration.

Definition : supply and demand feasibility
A set of flows is *supply and demand feasible* or just *feasible* if it is both supply feasible and demand feasible.

2.3.4 Wardrop's principles

If users are assumed to be rational such that each of them chooses their own paths in order to minimise their own individual costs of travel from origin to destination, an equilibrium arises when no user can reduce his own cost by selecting a different path. In this state, all paths used between O-D pairs have the same cost and less than that of any unused path. This condition is expressed formally by Wardrop's first principle (1952): "the journey times on all routes actually used are equal, and less than those which would be experienced by a single vehicle on any unused route". As a result of this principle, the paths actually used are minimum cost paths in the existing state. This flow pattern is descriptive because it describes at least approximately the real phenomenon of individual route choice in a transportation network.

Wardrop's second principle represents the state in which the total cost of travel in the whole network is minimised. It is known as the system optimum principle and is prescriptive because the flows of the system optimum must usually be forced upon the users in order to minimise the total cost.

2.4 NETWORK REPRESENTATION

There are various ways of representing a network according to the level of detail required for the intended purpose. For example, linked intersections in Figure 2-1 can be represented as a simple way in Figure 2-2 or a detailed way in Figure 2-3. The simplest way is to use one node to represent a whole junction and one link to represent a road between junctions. The detailed way of representing a network is according to the streams in an approach and possible turning movements. This detailed way of representing a network has an advantage that the movement of flow or even turning movement can be estimated. In an approach, many streams are represented as links. The node is used to distinguish among links. The level of detail depends on the data and cost of analysis. In general, more detailed representations lead to more accurate estimations in modelling. The analysis of impact of the level of detail used has been studied by Bovy and Jansen (1983).

Allsop (1992) set out some of principles describing how to represent a road junction by a set of nodes and links in traffic assignment models:

1. Any point at which streams of traffic merge or diverge must be represented by a node.
2. Traffic travels in one direction along each link.
3. Traffic entering a node may leave the node by any link starting at the node.
4. The cost of travel is made up of costs of travel along the links traversed.
5. The cost of travel along a link is a known function of the flows of traffic on that and

other links.

He also pointed out that the representation of traffic management measures requires that

1. Traffic entering a junction in one stream may leave the junction only by certain permitted exits.
2. Different streams of traffic entering a junction from the same approach road may incur different delays at the junction.

He commented that a junction can be represented in a way that meets both of these sets of requirements as follows:

1. Representing each stream of traffic entering the junction by a link ending at its own entry node.
2. Representing each exit from the junction by an exit node with a link starting at it and leading away from the junction.
3. Representing the permitted movements from each entering stream by links from its entry node to the relevant exit nodes.
4. Where an exit node has only one link ending at it, that node is omitted and the link starting at it is combined with the link ending at it.
5. Where an entry node has only one link starting at it, and that link ends at an exit node, that entry node is omitted and the link ending at it is combined with the link starting at it.

The work presented in this thesis follows these principles. In particular, Charlesworth's network is represented according to this way. In appendix 2, there is a sketch of this network which has 6 junctions. The detailed way of representing these junctions is shown in appendix A2.4. Junctions are represented by as few nodes as possible without losing information of detailed movements.

2.5 FORMULATIONS

2.5.1 Mathematical representation of Wardrop's first principle

Wardrop's first principle (1952) can be expressed mathematically as a set of path flows t_p that satisfy the following relations;

$$\left.\begin{array}{l} t_p > 0 \Rightarrow C_p = M_{od} \\ t_p = 0 \Rightarrow C_p \geq M_{od} \end{array}\right\} \forall p \in P_{od}, \forall od \in B \qquad (2.5)$$

where P_{od} is the set of paths from origin o to destination d.

t_p is the traffic flow on path p

C_p is the travel cost of using path p

M_{od} is the minimum travel cost from origin o to destination d.

This representation makes the implicit assumptions that each traveller has perfect information or knowledge of the travel time on each of the routes that he could use, and each traveller has an identical route choice criterion based on the travel cost. However, in reality each traveller chooses a route based on the perceived travel cost. The perceived travel cost varies between people. The identical route choice criterion based on only the travel time excludes the different preferences for components of a generalised cost. Components of the generalised cost include distance, travel time, tolls, delay, number of stops, and scenery. In reality, each traveller makes a route choice based on some combinations of these attributes. We then assume that each traveller has a probabilistic distribution for the perceived travel cost. This leads to the development of a stochastic traffic assignment (see, Sheffi 1985, Lee 1990 and Lee et al 1990).

2.5.2 Beckmann's formulation

Beckmann et al (1956) formulated equilibrium assignment according to Wardrop's first principle as a convex mathematical programme;

$$\textbf{PROBLEM 1: } \min_{\mathbf{t}} \; z(\mathbf{t}) \tag{2.6}$$

$$\text{s.t.} \quad t_p \geq 0 \;\; \forall\, p \in P_{od}, \forall od \in B$$

$$\sum_{p \in P_{od}} t_p = T_{od} \qquad \forall\, od \in B$$

$$\text{where } z(\mathbf{t}) = \sum_{a \in L} \int_{v=0}^{v_a} c_a(v)\, dv$$

$$v_a = \sum_{od} \sum_{p \in P_{od}} t_p \delta_{ap}, \quad \forall\, a \in L, \text{ and}$$

$$\delta_{ap} = \begin{cases} 1, & \text{if } a \in p \\ 0, & \text{otherwise} \end{cases}$$

Equivalence of this formulation to Wardrop's first principle can be shown by forming a Lagrangean function and finding derivatives with respect to the path flow t_p. First, the Lagrangean function can be written as:

$$L(\mathbf{t}, \psi) = z(\mathbf{t}) + \sum_{od} \psi_{od} \left(T_{od} - \sum_{p \in P_{od}} t_p \right) \tag{2.7}$$

where ψ_{od} is the Lagrange multiplier associated with the constraint, $\sum_{p \in P_{od}} t_p = T_{od}$ of the problem 1.

According to the Kuhn Tucker conditions (Taha, 1982) at the optimal point, the Lagrangean satisfies the following conditions with respect to path flow:

$$t_p \frac{\partial L}{\partial t_p} = 0 \text{ and } \frac{\partial L}{\partial t_p} \geq 0, \quad \forall p \in P_{od}, \forall od \in B$$

$$\frac{\partial L}{\partial \psi_{od}} = 0, \quad \forall od \in B \tag{2.8}$$

$$t_p \geq 0, \quad \forall p \in P_{od}, \forall od \in B$$

Firstly, taking the first derivatives of the Lagrangean function with respect to t_p and equating it to zero, we have:

$$\frac{\partial L}{\partial t_p} = C_p - \psi_{od} \quad \forall p \in P_{od}, \forall od \in B \tag{2.9}$$

Secondly, taking the first derivatives of the Lagrangean function with respect to ψ_{od} and equating it to zero, we have:

$$\frac{\partial L}{\partial \psi_{od}} = T_{od} - \sum_{p \in P_{od}} t_p \quad \forall od \in B \tag{2.10}$$

We have thus obtained the following equations:

$$\begin{aligned} t_p(C_p - \psi_{od}) &= 0 \quad \forall p \in P_{od}, \forall od \in B \\ (C_p - \psi_{od}) &\geq 0 \quad \forall p \in P_{od}, \forall od \in B \\ t_p &\geq 0 \quad \forall p \in P_{od}, \forall od \in B \\ \sum_{p \in P_{od}} t_p &= T_{od} \quad \forall od \in B \end{aligned} \tag{2.11}$$

In the first equation of (2.11), we have two possibilities with respect to t_p: $t_p = 0$ and $t_p \neq 0$: in view of the third equation of (2.11), the equation, $t_p \neq 0$ reduces to $t_p > 0$. If $t_p = 0$, then $(C_p - \psi_{od}) \geq 0$. If $t_p > 0$, then $(C_p - \psi_{od}) = 0$. These represent Wardrop's first principle that if path $p \in P_{od}$ is used, then C_p, the cost of travel on path p is the same as the minimum cost, ψ_{od} for that od pair. Otherwise, the path p is not used and the cost of travel on it, C_p is greater than the minimum cost, ψ_{od} for that od pair. The last two

equations of (2.11) represent the nonnegativity and flow conservation conditions respectively.

This formulation can be used in the separable case in which each link travel cost is an increasing function $c_a(v_a)$ of the flow on that link alone. In this case, the Jacobian matrix $J = \frac{\partial c}{\partial v}$ has strictly positive diagonal elements $J_{aa} = \frac{\partial c_a}{\partial v_a} > 0$ and zero off-diagonal ones $J_{ab} = 0$ ($a \neq b$). This increasing and separable cost function is a sufficient condition in order to formulate Wardrop's equilibrium as a convex mathematical programme as it guarantees the convexity of the objective function. In more realistic modelling, we have to consider the effect on the cost function for each link of the flows on other links. A more general formulation of equilibrium that can accommodate interactions of this kind is introduced in the next subsection.

2.5.3 Variational inequality formulation

Smith (1979) showed that Wardrop's condition of user equilibrium is equivalent to a variational inequality formuation as:

Theorem (Smith, 1979): The feasible path flow vector $\mathbf{t} \in \mathcal{B} \cap \mathcal{S}$ is an equilibrium flow if for any demand feasible path flow vector $\mathbf{s} \in \mathcal{B}$,

$$\mathbf{C(t)} \cdot \mathbf{(t-s)} \leq \mathbf{0}$$

where $\mathbf{C(t)}$ is the vector of path cost at the flow $\mathbf{t}$.

The related variational inequality problem is as follows:

PROBLEM 2: Find a feasible path flow vector $\mathbf{t} \in \mathcal{B} \cap \mathcal{S}$ such that $\forall\, \mathbf{s} \in \mathcal{B}$,

$$\mathbf{C(t)} \cdot \mathbf{(t-s)} \leq \mathbf{0} \qquad (2.12)$$

This path flow representation can be changed to the link flow representation using the incidence relation of paths and links:

$$v_a = \sum_{od} \sum_{p \in P_{od}} t_p \delta_{ap}, \quad \forall a \in L \tag{2.13}$$

$$\text{where } \delta_{ap} = \begin{cases} 1, & \text{if } a \in p \\ 0, & \text{otherwise} \end{cases}$$

The resulting problem of the variational inequality formulation based on link is:

PROBLEM 3: Find a feasible link flow vector $\mathbf{v} \in \mathcal{B} \cap \mathcal{S}$ such that $\forall \mathbf{u} \in \mathcal{B}$

$$\mathbf{c(v)} \cdot (\mathbf{v-u}) \leq \mathbf{0} \tag{2.14}$$

where $\mathbf{c(v)}$ is the link cost at the flow $\mathbf{v}$.

Smith (1979) defined an *assignment process* using the variational inequality formulation in equation (2.14) as: the ordered pair $(\mathbf{v}, \mathbf{u})$ with $\mathbf{v}, \mathbf{u} \in \mathcal{B}$ is an assignment process if $\mathbf{v} \in \mathcal{S} \cap \mathcal{B}$ and

$$\mathbf{c(v)} \cdot (\mathbf{v-u}) \geq \mathbf{0}$$

The assignment process can be interpreted as a dynamical adjustment process: $\mathbf{v}$ may be thought of as today's link-flow and $\mathbf{u}$ may be thought of as tomorrow's link-flow. This process of route choice is directed towards routes of lower costs until Wardrop's equilibrium is reached. Note that $\mathbf{v}$ is a Wardrop's equilibrium if and only if the assignment process starting from $\mathbf{v}$ is $(\mathbf{v}, \mathbf{v})$.

2.5.4 Gap functions

Hearn (1982) established that any solution of the mathematical programme

$$\begin{aligned} &\min_{\mathbf{v} \in \mathcal{B}} G(\mathbf{v}) \\ &\text{where } G(\mathbf{v}) = \max_{\mathbf{u} \in \mathcal{B}} \mathbf{c(v)} \cdot (\mathbf{v - u}), \quad \forall \mathbf{v} \in \mathcal{B} \end{aligned} \tag{2.15}$$

is a solution of the variational inequality formulation of the equation (2.14).

Under the condition that $\mathbf{c(v)}$ is an increasing and separable function, $G(\mathbf{v})$ is a convex

function (for proof, see Hearn, 1982).

In order to scale the values of the gap function, we can use a mean gap function, $\overline{G}(v)$, which is calculated as the gap function divided by the demand as:

$$\overline{G}(\mathbf{v}) = \frac{G(\mathbf{v})}{\sum_{od \in B} T_{od}} \tag{2.16}$$

This gap function is useful to monitor the convergence of an algorithm irrespective of the amount of demand because it expresses the gap value in terms of mean excess cost per trip. In this thesis, this scaled gap function will be used in the presentation of results.

Nguyen and Dupuis (1984) considered a subtly different function. They showed that any solution of

$$\begin{aligned} &\min_{\mathbf{v} \in \mathcal{S}} g(\mathbf{v}) \\ &\text{where } g(\mathbf{v}) = \max_{\mathbf{u} \in \mathcal{S}} \mathbf{c}(\mathbf{u}) \cdot (\mathbf{v} - \mathbf{u}), \quad \forall \mathbf{v} \in \mathcal{S} \end{aligned} \tag{2.17}$$

is also a solution of variational inequality formulation of Wardrop's principle.

Fisk (1984) mentioned that minimisation of these gap functions corresponds to infinitely constrained problems. However, Smith (1983) defined a family of objective functions,

$$K_p(\mathbf{v}) = \left(\sum_{i \in \mathcal{I}} [\mathbf{c}(\mathbf{v})(\mathbf{v} - E_{(i)})]_+^p \right)^{1/p} \tag{2.18}$$

for any integer $p \geq 1$ where $[z]_+ = \max(0,z)$ and $E_{(i)}$ is an extreme point of the feasible region $\mathcal{S}$ and $\mathcal{I}$ indexes all such points. Smith (1983) showed that any solution of

$$\min_{\mathbf{v}} K_p(\mathbf{v}) \tag{2.19}$$

is a solution of the variational inequality formulation (2.14) of Wardrop's principle. This

formulation has the advantage that calculation of the objective function is a finite process. Hearn et al (1984) pointed out that the relation of the gap function in equation (2.15) to equation (2.18) is:

$$G(\mathbf{v}) = K_{\infty}(\mathbf{v}) \tag{2.20}$$

where

$$K_{\infty}(\mathbf{v}) = \lim_{p \to \infty} K_p(\mathbf{v}) \quad \forall \mathbf{v} \in \mathcal{B}$$

2.6 COST FUNCTIONS

2.6.1 Introduction

Cost functions are used to represent the transportation supply by indicating the impedance on each link and its variation with flow. The cost functions can be developed by either an empirical or a theoretical relation between the flow and the cost. In general, costs are incurred on links as well as at junctions. Links generally correspond to streets and nodes to junctions. The properties of the cost function influence the convergence behaviour of solution methods.

2.6.2 Link cost functions

Branston (1976) surveyed some travel cost functions and reviewed their applicability using observed data. He reviewed cost functions in two categories: empirical and theoretical approaches. Empirical approaches included cost functions that were derived by fitting them to observed data. In theoretical approaches, the cost function is derived from theoretical consideration of the relationship between flow and

travel cost as a sum of the running time and the queuing time at junctions. In general, physical characteristics such as length, width, parking restriction, turning movements and signal timing determine a form of the cost function and the values of parameters in the cost function.

The notation used to describe this is:

v is the flow on a link

c(v) is the travel cost at flow v

t_0 is the travel cost at zero flow

Q^P is the practical capacity on a link

Q^S is the steady state capacity on a link

α_1, α_2, α_3 and α_4 are parameters representing characteristics of a road.

Note that the subscript a for representing link a is omitted in this section.

Highway Capacity Manual (1985) defined the *practical capacity* as the maximum number of vehicles that can pass a given point on a roadway or in a designated lane during one hour without the traffic density being so great as to cause unreasonable delay, a hazard or to restrict to the driver's freedom to manoeuvre under prevailing roadway and traffic conditions. On the other hand, the *steady state capacity* is defined as the capacity in which fluctuations such as the microscopic and stochastic characteristics of a traffic stream are ignored over time.

When flow and capacity are measured on a lane basis, the above symbols are denoted by a prime, e.g.,

v' is the flow per lane on a link

Q'^S is the steady state capacity per lane on a link.

2.6.2.1 Empirical approaches

Irwin, Dodd and Von Cube (1961) proposed a cost function with two linear segments.

$$c(v) = t_a + \alpha_1(v' - Q'^p) \text{ if } v' < Q'^p$$
$$c(v) = t_a + \alpha_2(v' - Q'^p) \text{ if } v' \geq Q'^p \qquad (2.21)$$
$$\text{where } t_a = t_0 + \alpha_1 Q'^p$$

Mathematical programming has some difficulties in using these functions because they are not smooth at Q'^p. However, they can be used to provide a value of travel cost at any level of flow. The function is shown in Figure 2-4.

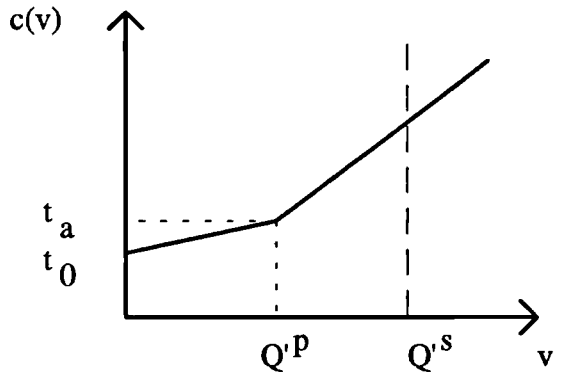


Figure 2-4. Irwin, Dodd and Von Cube's cost function

Smock (1962) proposed a cost function which is smooth. This function was used in the Detroit Area Transportation Study:

$$c(v) = t_0 \exp(v/Q^s) \qquad (2.22)$$

In application, Smock estimated the value of Q^s for each link by averaging the capacities of the intersections at each end. This value will not be equal to the steady state capacity unless the end intersections have capacities which are equal and less than those of all other points on the link. This function was incorporated in a heuristic iterative capacity restraint assignment procedure in order to use in the city of Flint, Michigan, USA.

Mosher (1963) proposed a logarithmic cost function. He chose this function because it has the property that the change in cost per unit flow with increasing flow is small for low flows but increases as capacity is approached. Note that this cost has an asymptote at $v = Q^s$:

$$c(v) = t_0 - \ln(1 - v/Q^s) \qquad (2.23)$$

where $v < Q^s$

However, in a realistic network, the travel cost remains finite unless the road system fails completely. The problem of this asymptotic function is that in an iterative assignment computation, it is quite possible that the flows assigned on some links during the early stages exceed the corresponding capacities so that the travel cost cannot be calculated with functions of this kind. Figure 2-5 shows the relation between link flow and cost for this asymptotic cost function.

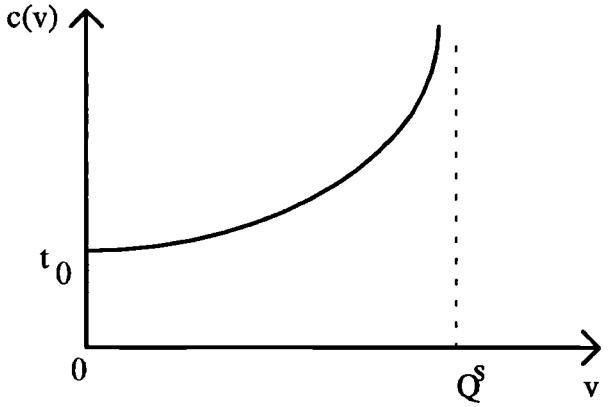


Figure 2-5. The asymptotic link cost function

The Bureau of Public Roads (BPR, 1964) developed a well known function:

$$c(v) = t_0\left[1+\alpha_1\left(\frac{v}{Q^p}\right)^{\alpha_2}\right] \qquad (2.24)$$

The parameters α_1=0.15 and α_2=4 that were proposed for normal use imply that the

travel cost at capacity is 15 percent higher than the free flow travel cost. The value of α_1 is the delay at capacity expressed as a proportion of free flow travel cost. The value of α_2 determines the slope of the curve and in turn it represents the degree of the congestion effect. This function has no asymptote, so can be evaluated at any flow.

Soltman (1965) developed a congestion function for the Pittsburgh Area Transportation Study:

$$c(v) = t_0 2^{v/Q^p} \quad \text{where } v/Q^p \leq 2 \tag{2.25}$$

This function was used in Schneider's (1963) combined distribution and assignment model in the Pittsburgh Area Transportation Study. However, there are no examples of comparison between observed and assigned flows.

The Traffic Research Corporation (1966) used the following function for Winnipeg area:

$$c(v) = \alpha_1 + \alpha_2 (v' - \alpha_3) + \sqrt{\alpha_2^2 (v' - \alpha_3)^2 + \alpha_4} \tag{2.26}$$

where α_1, α_2, α_3 and α_4 are parameters.

This function relates the travel cost per unit distance to flow per lane on the links. The links of Winnipeg network were divided into categories according to their speed limits and the practical capacities per lane. For each category, there were a wide scatter of points at low flows, but there were few points at high flows. Hence, the conditions at high flows are simulated. The cost function was then fitted to this curve. A good comparison between assigned and observed travel costs was shown by Florian and Nguyen (1975).

Overgaard (1967) generalised the cost function of Soltman (1965) and Smock (1962):

$$c(v) = t_0 \alpha_1^{(v/Q^p)^{\alpha 2}} \tag{2.27}$$

Here the value of α_1 is the ratio of the travel cost per unit distance at practical capacity to that at zero flow. Soltman assumed a value of 2 for α_1 whilst Smock assumed a value of the base of the natural logarithm, e. Overgaard found that suitable value for α_1 varied between 1.0 and 1.7 in a Toronto network.

2.6.2.2 Theoretical approaches

In theoretical approaches, the travel cost on a link is calculated as the sum of the average running cost (from the link entry to the tail of the exit queue) and the average cost spent in queuing at junctions at that flow, i.e.,

$$c(v) = \frac{(t_r + t_q)}{D} \tag{2.28}$$

where t_r and t_q are the running and queuing travel costs respectively, and D is the length of the link.

Campbell, Keefer and Adam (1959) proposed a cost function which relates the speed to the ratio of flow to capacity for a signalised urban arterial in Chicago Area Transportation Study.

$$c(v) = \begin{cases} t_0 & (v/Q \le 0.6) \\ t + \alpha(v/Q - 0.6) & (v/Q > 0.6) \end{cases} \tag{2.29}$$

Wardrop (1968) developed a cost function which shows the relationship between overall speed and flow on streets in central London. Because it was formulated in terms of overall speed in the network rather than speeds on individual links, Wardrop's original model may be described as a network cost function. The method includes both the queuing cost at signalised junctions and the running cost between signalised junctions. This method differs from that of Campbell et al which used a heuristic procedure to calculate the delay at a junction for any flow. On the other hand, Wardrop derived approximate formulae relating average delay and flow for both vehicle-actuated and fixed time traffic signals. For both types of signal he showed that the relation between average delay and flow was approximately:

$$c(v) = \frac{t_0}{(1 - v/Q_s)} + \frac{\alpha_2}{(1 - v/\alpha_1)D} \qquad (2.30)$$

where D is a link length.

The role of the cost function is its use in assignment procedures. Solution algorithms are used in order to solve traffic assignment. Solution algorithms for Wardrop's first principle that based on the formulation of Beckmann et al (1956) require that cost functions be easily and quickly integrable. This in turn implies that its definite integral can be evaluated analytically. The BPR and Campbell et al functions among others (see Table 2-1) meet this requirement. However, Overgaard and TRC functions can be expensive in computational time. Table 2-1 shows the degree of difficulty in the integral of cost functions c(v) reviewed in this section.

Type of cost function	Form of integral	Degree of difficulty
Irwin et al	polynomial	easy
Smock	exponential	fairly easy
Mosher	logarithmic	fairly easy
BPR	polynomial	easy
Soltman	polynomial	easy
TRC	polynomial	difficult
Overgaard	numerical	very difficult
Campbell et al	polynomial	easy
Wardrop	polynomial	easy

Table 2-1 The degree of difficulty in the integral of cost functions

2.6.3 Junction delay formulae

A link cost function represents the relationship between travel time and flow on that link. This ignores any interaction with other links. In general, the junction delay is a function of the flows on several links approaching a junction. Priority and signal controlled junctions are typical examples of cases in which the flow on one link can have a profound effect on the cost of using another. Estimates of junction delays can be obtained from queuing theory.

2.6.3.1 Priority controlled junction

We now introduce some terms that are widely used in the priority junction analysis. There are

Gap : the time interval between successive vehicles passing a point on the roadway.

Critical gap : the duration of the shortest acceptable gap if the gap acceptance distribution is a step function.

Gap acceptance distribution : each gap is accepted by the waiting vehicle's driver with a probability that is related to its duration by this distribution. A decision on accepting a gap is made at the commencement of each gap.

Tanner (1962) presented a formula for estimating the capacity of a minor road movement at a priority junction. He assumed that arrivals on each of major and minor roads are Poisson, a constant minimum headway for each of major and minor road vehicles, and that a fixed critical gap in major road traffic is acceptable to a minor road vehicle. The capacity of the minor road is then:

$$\mu_a = \frac{v_b(1-\rho_b)}{\exp[v_b(\kappa - 1/Q_b)]\{1-\exp(-v_b/s_a)\}} \tag{2.31}$$

where v_a is minor road flow

v_b is major road flow

s_a is the departure rate in the absence of all other traffic.

Q_b is the capacity in major road b.

$\rho_b = v_b/Q_b$

κ is the critical gap

The mean delay on the minor road can be expressed as:

$$d_a = \frac{E(y^2)/(2Z) + v_a Z \exp(-v_b/s_a)[\exp(v_b/s_a) - v_b/s_a - 1]/v_b}{1 - v_a Z[1-\exp(-v_b/s_a)]} \tag{2.32}$$

where y denotes the duration of a block of vehicles and has first and second moments:

$$E(y) = \frac{\exp[v_b(\kappa - 1/Q_b)]}{v_b(1-\rho_b)} - \frac{1}{v_b} \tag{2.33}$$

$$E(y^2) = \frac{2\exp[v_b(\kappa - 1/Q_b)]\{\exp[v_b(\kappa - 1/Q_b)] - v_b\kappa(1-\rho_b) - 1 + \rho_b - \rho_b^2 + \rho_b^2/[2(1-\rho_b)]\}}{v_b^2(1-\rho_b)^2}$$

$$Z = E(y) + 1/v_b$$

Tanner's formula depends upon explicit knowledge of the critical gap and minimum headways rather than geometric features. Because of this it is difficult to use in junction design. Kimber and Coombe (1980) developed a capacity formula for minor roads based on the empirical study of various junctions. This formula is incorporated in PICADY (Semmen, 1985) as follows:

$$\mu_a = G_a(K_a + H_a - Y_a \sum_b e_{ab} v_b) \tag{2.34}$$

where G_a, K_a, H_a, Y_a and e_{ab} are parameters to represent the geometric characteristics.

The mean delay incurred at a priority controlled junction can be estimated using the Pollaczek-Khinchine formula for steady state:

$$d_a = \frac{1}{2\mu_a}\left[\frac{\rho_a}{1-\rho_a}\right] \qquad \text{where } \rho_a = \frac{v_a}{\mu_a} \tag{2.35}$$

2.6.3.2 Signal controlled junction

Terms that are used in the description of a signal controlled junction are as follows:

Cycle: Complete sequence of signal indications.

Cycle length : The total time to complete one cycle, denoted by the symbol C.

Phase: The part of a cycle allocated to any combination of traffic movements receiving right-of-way simultaneously.

Green time: The time within a given phase during which the green indication is shown, denoted by the symbol G.

Lost time: Time during which the intersection is not effectively used by any movement; these times occur during the changing interval (while the intersection is clearing), and at the beginning of each phase as the first few cars in a standing queue experience start-up delays. Lost time is denoted by L.

Effective green time: The time during a given phase that is effectively available to the permitted movements, denoted by the symbol g.

Green ratio: The ratio of effective green time to the cycle length denoted by $\lambda = g/C$.

Effective red: The time during which given movements or set of movements are effectively not permitted to move. It is the cycle length minus the effective green time, denoted by r. That is, $r = C - g$.

The US Highway Capacity Manual (HCM, 1985) defined *capacity at a signalised junction* for each stream as the maximum rate of flow (for the subject approach) which may pass through the intersection from that stream under prevailing traffic, roadway, and signal conditions. Traffic conditions include flows on each approach, the distribution of vehicles by movements (left, through, right), the vehicle type distribution within each movement, the location and use of bus stops within the intersection area, pedestrian crossing flows, and parking movements within the intersection area. Roadway conditions include the basic geometry of the intersection, including the number and width of lanes, gradients and land-use location (including parking lanes). Signal conditions include a full definition of the signal phasing, timing, type of control, and an evaluation of signal progression on each approach.

Capacity at signalised junctions is based on the concept of saturation flow and saturation flow rates. *Saturation flow rate* (HCM, 1985) is defined as the maximum rate of flow that can pass through a given intersection approach or lane group under prevailing traffic and roadway conditions during effective green time. Saturation rate is denoted by the symbol s, and is expressed in units of vehicles per hour of effective green time.

The capacity of a given approach or lane group can be stated as:

$$Q=s\ g\ /C \tag{2.36}$$

The *degree of saturation,* X is defined as the ratio of flow rate to capacity v/Q (or, vC/(sg), or $v/(\lambda s)$). Values of X range from 0.0, when the flow rate is 0.0, to 1.0 when the flow rate equals capacity.

Signal controlled junction delay can then be estimated using Webster's two-term formula (1958) for steady state:

$$d(v,\Lambda,X,C)=\frac{9}{10}\left[\frac{C(1-\Lambda)^2}{2(1-\Lambda X)}+\frac{X^2}{2v(1-X)}\right] \tag{2.37}$$

where $X = v / (s\Lambda)$

This delay does not cause a junction interaction because it does not depend on any other links. However, if we introduce a signal control policy in order to adjust the signal timings according to the flows, this will cause junction interactions. We now introduce Webster's (1958) signal control policy as follows:

$$\min_{\lambda} \max_{a} X_a$$
$$\text{where } X_a = v_a / (\Lambda_a s_a) \tag{2.38}$$

$$\Lambda_a = \sum_{i=0}^{n} A_{ia} \lambda_i$$

$$\text{where } A_{ia} = \begin{cases} 1, & \text{if link a has green during stage i} \\ 0, & \text{otherwise} \end{cases}$$

2.6.4 Combination of link cost and junction delay

In this study, we assume that total travel time on the link consists of the sum of free-flow travel time, and each of link and junction based delays. The link based delay represents any increase in running time due to congestion and the junction delay represents the delay caused by the junction control methods such as priority or signal control rules. We can model the total travel time as:

$$T_a(\mathbf{v}) = t_0 + \gamma[c_a(v_a) - t_0] + (1-\gamma)d_a(\mathbf{v}) \quad \forall a \in L \tag{2.39}$$

where t_0 is the free-flow travel time, $c_a(v_a)$ is the link based travel time, $d_a(\mathbf{v})$ is the junction delay and γ is a parameter in the range of [0, 1] to control the relative influence of link and junction delays and to represent the user's perception between two delays. That is, in addition to the free-flow time:

If γ=0, only junction delay is considered in route choice.

If γ=0.5, the balanced effect of link and junction delays is considered.

If γ=1, only link delay is considered.

2.6.5 Good behaviour

Some properties of solutions to traffic assignment such as existence, uniqueness,

stability and sensitivity are of considerable practical importance. Good behaviour of a traffic assignment model depends on these properties: if a model has a unique stable solution, we say that it has good behaviour. In this section, we note some theorems reviewed by Harker and Pang (1990). Before introducing these existing theorems, we define some of terminology. Let $c_a(\cdot)$ be the cost function for link a and let v_b be the traffic flow on link b.

Definition : *the Jacobian matrix* J of the cost function, $c_a(\cdot)$ on the link flow, v_b is the matrix of partial derivatives,

$$J_{ab} = \partial c_a / \partial v_b,\ 1 \le a, b \le L \tag{2.40}$$

Definition : *symmetry*

A matrix J is *symmetric* if $J_{ab}=J_{ba}$, for all a,b

Definition : *separability*

A cost function is *separable* if $J_{ab} = 0$ for $a \neq b$

Definition : *strictly monotone function*

A cost vector $\mathbf{c}(\mathbf{v})$ is *strictly monotone* if

$$[\ \mathbf{c}(\mathbf{v})-\mathbf{c}(\mathbf{u})\ ] \cdot (\mathbf{v}-\mathbf{u}) > \mathbf{0} \quad \forall\ \mathbf{v},\mathbf{u},\ \mathbf{v} \neq \mathbf{u} \tag{2.41}$$

Definition : *positive definite*

A matrix **M** is *positive definite* if

$$\mathbf{x}^t\mathbf{M}\mathbf{x} > \mathbf{0} \quad \forall\ \mathbf{x} \in R^n,\ \mathbf{x} \neq \mathbf{0} \tag{2.42}$$

Note that a cost function is strictly monotone if and only if its Jacobian is positive definite. A separable cost function is strictly monotone if and only if it is a strictly increasing function.

2.6.5.1 Existence of solutions

A theoretical tool for establishing the existence of equilibria is the fixed point theorem of Brouwer. This demonstration of existence requires that the cost function be

continuous and that the feasible region be a nonempty, compact (i.e., closed and bounded) and convex set.

Theorem (Smith, 1979)

Let $\mathcal{S}$ be a closed, convex subset of R_+^m such that $\mathcal{S} \cap (\mathcal{S}\text{-}B) \neq \varnothing$, let $c:\mathcal{S} \rightarrow R_+^m$ be continuous on $\mathcal{S}$-B where $B=\bar{\mathcal{S}} \cap \overline{R_+^m - \mathcal{S}}$ is the boundary of $\mathcal{S}$ in R_+^m, and let c be such that $\forall v_0 \in \mathcal{S}\text{-}B$, $\exists k \in (0,1)$, $\forall v \in \mathcal{S}\text{-}\overline{k\mathcal{S}}$, $c(v)(v\text{-}v_0) > 0$, then there is a Wardrop equilibrium in $\mathcal{S} \cap (\mathcal{S}\text{-}B)$.

2.6.5.2 Uniqueness

Uniqueness of the equilibrium solution can be shown if the link cost function is strictly monotone (or, if separable, strictly increasing). There are theorems for uniqueness as follows:

Theorems (Karamardian, 1976; Minty, 1962; Smith, 1979)

Let $\mathcal{S} \cap \mathcal{B}$ be a nonempty and convex subset of R^n and let $\mathbf{c}(\cdot)$ be a strictly monotone mapping from $\mathcal{S}$ into R^n. Then the problems 2 and 3 have at most one solution.

For proofs, see the references above.

2.6.5.3 Stability

There are several studies of stability in traffic assignment. Beckmann et al (1956) proposed a stability study to investigate how changes in initial flows and thus initial costs affect the equilibrium travel pattern. Figures 2-6 and 2-7 respectively show examples of stable and unstable behaviour. Horowitz (1984) followed the same concept as Beckmann et al (1956) to investigate whether or not use of different initial flows in a solution

procedure leads to different equilibrium travel patterns.

Smith (1984) considered the stability of a dynamic model of traffic assignment, which is a continuous time adjustment mechanism for selecting routes. In this model, the rate of change of flow is specified by a function that describes the swapping among alternative paths. Using Lyapunov's theorem (see, for exmple, Zangwill 1969), Smith (1984) was able to show that user equilibrium is stable if path costs are differentiable and monotone.

It is necessary to define a concept of stability. In this study, we classify stability as global and local. The concept of the global stability is similar to that of Beckmann et al (1956) and Horowitz (1984) as follows:
A traffic assignment model is *globally stable* for a certain problem if any initial flows in the feasible region lead to an equilibrium by a dynamical adjustment process of this model.

Local stability is a weaker concept than that of global stability so that if global stability is satisfied, local stability also satisfied. Local stability is defined as follows:
A traffic assignment model is *locally stable* for a certain problem if any feasible perturbation from an equilibrium that is sufficiently small leads to a return to that equilibrium by a dynamical adjustment process of this model.

Note that in Figures, 2-6 and 2-7, the light line having an arrow indicates a dynamical adjustment process. Each step of the horizontal line to reaching one of the demand and supply curves represents a swapping rate of traffic flows whilst that of the vertical line represents a changed cost. In Figure 2-6, this adjustment process is repeated until it reaches an equilibrium which is *stable*. On the other hand, this process of Figure 2-7 fails to reach an equilibrium which is *unstable*.

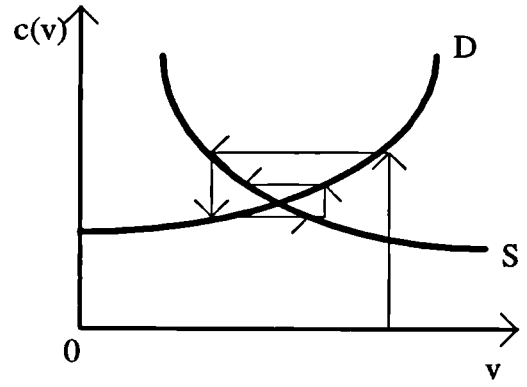


Figure 2-6 Example of a stable equilibrium

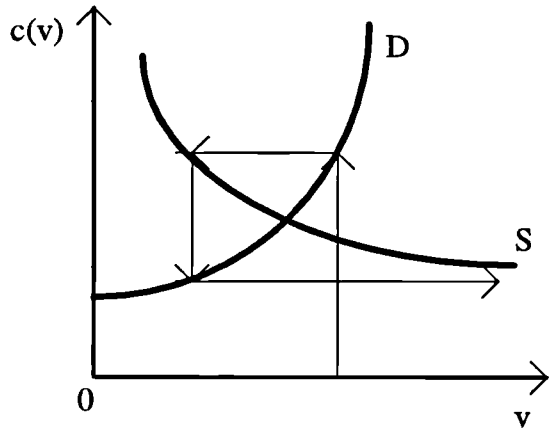


Figure 2-7 Example of an unstable equilibrium

2.6.5.4 Sensitivity

Sensitivity analysis investigates how changes in the travel cost functions affect the direction of the change in the equilibrium pattern and the incurred travel costs. Dafermos and Nagurney (1983, 1984) analysed the direction of change in an equilibrium flow pattern resulting from changes in the cost functions and from the additional improvement of routes. Their analysis exploited the variational inequality formulations of network equilibrium. The question of changes in the equilibrium flow pattern induced by improvements relates directly to the network design problem. Chao and Friesz (1984),

and Tobin (1984, 1986) investigated how to calculate efficiently the magnitude and direction of flow pattern induced by changing the cost functions without solving the equilibrium problem again.

2.6.5.5 Conditions for good behaviour

Good behaviour of an assignment formulation can be identified by the existence of unique stable solution. Smith (1979) proved that a *sufficient condition* for good behaviour is that the Jacobian matrix of the cost function be everywhere positive definite. Heydecker (1983) proved that a corresponding *necessary condition* for good behaviour is that the Jacobian matrix of the cost function be everywhere a P matrix.

2.6.5.6 Satisfaction of conditions for good behaviour

This thesis investigates the good behaviour conditions and their associated properties in cases of a model of priority controlled junctions in traffic assignment, a model of signal controlled junctions in traffic assignment, and a multiclass traffic assignment model. First, the cost functions of these models are investigated to determine whether or not they satisfy the conditions for good behaviour. Second, in order to examine the uniqueness and stability aspects of good behaviour, each of these models is investigated to see whether different initial points lead to the same equilibrium of each of these models in a small example network. Furthermore, the values calculated using these models are compared with a symmetric equilibrium obtained by an analytical method.

The conditions for good behaviour and their associated properties for each model are given in Tables 2-2 and 2-3. In Table 2-2, traffic assignment models are classified according to the necessary and sufficient conditions for good behaviour. The BPR (1964)

function often used in separable traffic assignment formulation satisfies both of these conditions. However, the Tanner and PICADY formulae used in priority controlled junction modelling satisfy only the necessary condition when there are no hooking right turns. On the other hand, Webster's delay formula with Webster's signal setting method used in signal controlled modelling does not satisfy either condition.

In Table 2-3, traffic assignment models are classified according to the necessary and sufficient conditions for good behaviour and their associated properties. In particular, Table 2-3 shows that the necessary and sufficient conditions for good behaviour affect their associated properties. For example, Webster's delay formula with Webster's signal setting method in signal controlled junction modelling does not always satisfy the necessary and sufficient conditions for good behaviour, and this results in this model having multiple unstable equilibria. Chapters 4 and 5 will present a detailed analysis relating to Tables 2-2 and 2-3.

		Necessary condition	
		yes	no
Sufficient -	yes	Guaranteed good behaviour eg, BPR and other increasing separable functions	Cannot occur
condition	no	Indeterminate behaviour depending on network eg, Tanner*, PICADY* formulae	Known bad behaviour in some networks eg, Webster s delay with Webster's min max control policy

Table 2-2 Necessary and sufficient conditions for good behaviour

(Key: * these cases are subject to the exception of hooking right turns)

	BPR	Tanner	PICADY	Webster
Necessary con.	Yes	Yes*	Yes*	Not always
Sufficient con.	Yes	Not always	Not always	Not always
Existence	Yes	Yes	Yes	Yes
Uniqueness	Yes	Yes*	Yes*	No
Global stability	Yes	Yes*	Yes*	No

Table 2-3. Test for the satisfaction of good behaviour

(Key: * these cases are subject to the exception of hooking right turns)

2.7 SOLUTION ALGORITHMS

2.7.1 Separable case

Algorithms commonly applied to the traffic assignment problem are based on a linear approximation of an objective function. The most well known of these is the Frank-Wolfe (F-W) algorithm (see section 2.2). The F-W algorithm is applicable to any nonlinear optimisation problem with a pseudoconvex objective function and linearly constrained feasible set. The performance of this algorithm is well recognised and it is applicable to road traffic assignment in large networks using a small amount of computer memory. Disadvantages include poor convergence in later stages and the algorithm can generate cyclic flows (see Janson et al,1987).

The Frank-Wolfe algorithm consists of two steps: finding a direction and finding a step length. The direction is obtained by optimising a linear approximation of the objective function while maintaining feasibility. The step length is then determined in

order to optimise the objective function by searching in this direction. In this case feasibility is always maintained because the directions are calculated to lie within the feasible set.

The step of finding a direction is obtained by taking the first order of Taylor's series for the objective function. Let $z(\cdot)$ be an objective function, and $\mathbf{v}$ and $\mathbf{u}$ be traffic flows. This is written for $\mathbf{u}$:

$$z(\mathbf{u}) = z(\mathbf{v}^n) + \nabla z(\mathbf{v}^n + \beta(\mathbf{u}-\mathbf{v}^n)) \cdot (\mathbf{u}-\mathbf{v}^n) \quad \text{for some } \beta \in (0,1) \tag{2.43}$$

where superscript n represents the iteration number n.

A suitable linear approximation to $z(\mathbf{u})$ is achieved by setting β equal to zero

$$z(\mathbf{u}) \cong z(\mathbf{v}^n) + \nabla z(\mathbf{v}^n) \cdot (\mathbf{u}-\mathbf{v}^n) \tag{2.44}$$

The search direction can then be identified from the solution to the linear programme in $\mathbf{u}$

$$\min_{\mathbf{u}} \nabla z(\mathbf{v}^n) \cdot \mathbf{u} + [\, z(\mathbf{v}^n) - \nabla z(\mathbf{v}^n) \cdot \mathbf{v}^n \,] \tag{2.45}$$

Note that $[\, z(\mathbf{v}^n) - \nabla z(\mathbf{v}^n) \cdot \mathbf{v}^n \,]$ is a constant, so we can omit it from the objective function of the linear programme.

In order to find a steplength in a direction $\mathbf{d}^n = (\mathbf{u}^n - \mathbf{v}^n)$, we solve the uni-dimensional problem to minimise z.

$$\min_{\beta} z(\mathbf{v}^n + \beta \mathbf{d}^n), \quad \beta \in [0,1] \tag{2.46}$$

Because the feasible region is convex, any point on the line segment between $\mathbf{v}^n$ and $\mathbf{u}^n$ will satisfy the constraints and thus be feasible.

The algorithm solves these steps of direction search and step length alternately until various convergence criteria are satisfied. As stopping criteria, the difference of an objective value in the successive iterations or a maximum iteration number can be used.

One good aspect of this algorithm is that at every iteration we have a lower bound on the optimal value of objective function. Because by convexity, we have

$$z(\mathbf{v}^*) \geq z(\mathbf{v}^n) + \nabla z(\mathbf{v}^n) \cdot (\mathbf{v}^* - \mathbf{v}^n) \qquad (2.47)$$

where $\mathbf{v}^*$ is the optimal solution, and $\mathbf{v}^n$ is the feasible solution at iteration n.

In addition, we have that

$$z(\mathbf{v}^n) + \nabla z(\mathbf{v}^n) \cdot (\mathbf{v}^* - \mathbf{v}^n) \geq z(\mathbf{v}^n) + \nabla z(\mathbf{v}^n) \cdot (\mathbf{u}^n - \mathbf{v}^n) \qquad (2.48)$$

This is because $\mathbf{u}^n$ minimises $\nabla z(\mathbf{v}^n) \cdot \mathbf{u}$ for every feasible $\mathbf{u}$. Therefore $z(\mathbf{v}^n) + \nabla z(\mathbf{v}^n) \cdot (\mathbf{u}^n - \mathbf{v}^n)$ for every iteration n is a lower bound on $z(\mathbf{v}^*)$.

In other words, the step of direction search consists of finding minimum cost paths at the current costs and assigning all of the demand between each origin and destination to the associated path. The resulting assignment is called *all-or-nothing assignment.* This usually leads to the overloading of some of links and thus corresponds to an unrealistic travel pattern. In the all-or-nothing assignment, we assign all the demand T_{od} to the shortest path from origin o to destination d. The resulting assignment corresponds to an extreme point of the feasible region in path-space.

A minimum cost tree building algorithm is used to find the minimum costs between origin and destination pairs. Bellman's optimality principle (1957) is applied to calculate the minimum costs path. This is a recursive process to find the minimum cost path to each point by comparing every possible combination of links reaching it:

$$M_{on} = \min_{(b,n)} M_{ob} + c_{bn} \qquad \forall n \in N \qquad (2.49)$$

where M_n is the minimum cost of travel between o and n.

This leads to lengthy computing times. Many efficient algorithms have been devised to reduce this.

A difficulty inherent in steepest descent direction algorithms such as Frank-Wolfe is when the optimal point is in the interior of the feasible region. If $\mathbf{v}^*$ is an interior optimal point, then $\nabla z(\mathbf{v}^*) = 0$. In a small neighbourhood of $\mathbf{v}^*$, the individual components of the gradient may oscillate from positive to negative because they are near zero. Thus the geometric direction indicated by the gradient may change rapidly as the sequence of points approaches the optimal solution. This can cause the solution method to follow a zig-zag path near convergence.

2.7.2 Non-separable case

For non-separable problems, most of the algorithms that have been developed follow the framework of linear approximation and diagonalisation methods. Pang and Chan (1982) reviewed these methods. In this non-separable case, the cost function, $c_a(\mathbf{v})$ is a function of traffic flows on other links, represented by $\mathbf{v}=(v_1,v_2,...,v_L)$. At the beginning of these methods, an iterate $\mathbf{v}^0$ is given and a linear approximation $\mathbf{A}(\mathbf{v}^n)$ of the function $\mathbf{c}(\mathbf{v})$ at the point of $\mathbf{v}^n$ is made to obtain

$$\mathbf{c}^n(\mathbf{v}) = \mathbf{c}(\mathbf{v}^n) + \mathbf{A}(\mathbf{v}^n)\cdot(\mathbf{v}-\mathbf{v}^n) \tag{2.50}$$

The resulting variational inequality problem of finding $\mathbf{v}$ such that

$$\mathbf{c}^n(\mathbf{v})\cdot(\mathbf{u}-\mathbf{v}) \geq 0 \quad \forall\, \mathbf{u} \in \mathcal{S} \tag{2.51}$$

is solved at each iteration n to obtain the new iterate $\mathbf{v}^{n+1}$. The algorithms continue until a convergence criterion is met.

Some choices of $\mathbf{A}(\mathbf{v}^n)$ are as follows:

$\mathbf{A}(\mathbf{v}^n) = \nabla\, \mathbf{c}(\mathbf{v}^n)$: Newton type method

$\mathbf{A}(\mathbf{v}^n)$ = Diagonal part of $\nabla\, \mathbf{c}(\mathbf{v}^n)$,(ie, $A_{ab} = \delta_{ab} \dfrac{\partial c_a}{\partial v_b}$): linearised Jacobi methods

$\mathbf{A}(\mathbf{v}^n)$ is a symmetric positive definite: a projection method

Alternately, a diagonalisation algorithm referred to as the nonlinear Jacobi method has been used by many authors including Abduaal and LeBlanc (1979), Ahn (1979), Florian and Spiess (1982), Dafermos (1982), and Pang and Chan (1982) among others. In a diagonalisation algorithm, at each iteration n, diagonalisation is performed such that all the flows on links other than the flow on current link, v_k are fixed. Thus,

$$\tilde{c}_k^n(v_k) = c(v_1^n, v_2^n, \ldots, v_{k-1}^n, v_k, v_{k+1}^n, \ldots, v_L^n) \tag{2.52}$$

This diagonalisation algorithm gives a separable cost function that has a diagonal Jacobian matrix which is identical with the diagonal part of the Jacobian matrix of the full cost function. In the diagonalisation algorithm, the influence of link interaction is ignored within each iteration, and the separable cost functions are updated in this respect at the end of each iteration.

The following is a diagonalisation algorithm:

Step 0: Set n=0. Initialisation. Find a feasible link flow $\mathbf{v}^n$.

Step 1: Diagonalisation. Solve the diagonalised problem

$$\tilde{\mathbf{c}}^n(\mathbf{v}) \cdot (\mathbf{u} - \mathbf{v}) \geq \mathbf{0} \quad \forall \mathbf{u} \in \mathcal{S} \tag{2.53}$$

This can be solved using an existing algorithm for the separable case. This step yields the link flow $\mathbf{v}^{n+1}$.

Step 2: Convergence test. If $\mathbf{v}^{n+1} \approx \mathbf{v}^n$, stop. If not, set n=n+1 and go to step 1.

Sheffi (1985) showed that a streamlined version of the diagonalisation algorithm performs better than the original one. The streamlined version performs only one iteration within step 1 of each outer iteration. Thus Sheffi's streamlined diagonalisation algorithm differs from the usual one in step 1 by using the following form of it.

Step 1S: Streamlined diagonalisation. Solve the scalar problem in β

$$\tilde{\mathbf{c}}^n(\mathbf{v}^n + \beta(\mathbf{u}^n - \mathbf{v}^n)) \cdot (\mathbf{u}^n - \mathbf{v}^n) \geq \mathbf{0} \quad \forall \mathbf{u}^n \in \mathcal{B} \tag{2.54}$$

where $\mathbf{u}^n$ is the all-or-nothing assignment at cost $\tilde{\mathbf{c}}^n(\mathbf{v}^n)$.

The advantage of this streamlined version of the diagonalisation method is that it can reduce the computational time in Step 1, which includes an all-or-nothing assignment. This all-or-nothing assignment has shortest path search which is one of the most time consuming routines in the algorithm.

CHAPTER 3. SIMPLICIAL DECOMPOSITION ALGORITHMS

3.1 INTRODUCTION

In this chapter, a general simplicial decomposition algorithm is introduced. This is a solution approach for a nonlinear programme which uses a linear and a master subproblem. Frank-Wolfe, and its variants such as PARTAN search direction (Luenberger,1989, pp254-257; Arezki and Van Vliet, 1990), the modified search direction of Fukushima (1984), the modified steplengths of Weintraub et al (1985), Wolfe's away step (1974), Holloway's extension (1974), and Schittenhelm's algorithm (1990) are each described in terms of the general simplicial decomposition algorithm. New algorithms belonging to the simplicial decomposition class are proposed and tested using an example of the separable traffic assignment problem in order to compare their performance with other algorithms.

3.2 GENERAL SIMPLICIAL DECOMPOSITION ALGORITHM

3.2.1 General nonlinear problem

The following notation is adopted. Let $\mathcal{B}$ be the demand feasible region, $W \subset \mathcal{B}$ be a set of points and H(W) be the convex hull of W, ie,

$$H(W) = \{ v | v = \sum_{i=1}^{n} \beta_i u_i, \quad \sum_{i=1}^{n} \beta_i = 1, \quad \beta_i \geq 0, \quad u_i \in W \ \ (1 \leq i \leq n)\}$$

We now introduce some of mathematical definitions:

Definition: The set of points $W=\{u_1,...,u_n\}$ is *affinely independent* if

$$\sum_{i=2}^{n} \beta_i (u_i - u_1) = 0 \quad \Rightarrow \beta_i = 0, \ \ (2 \leq i \leq n)$$

In this case, W is (n-1) dimensional. We refer to W as an affine basis of H(W) and H(W) is a simplex.

Definition: $z:R^n \rightarrow R$ is a *convex* function

if $z(\beta u + (1-\beta)v) \leq \beta z(u) + (1-\beta)z(v)$ for any β, $0 \leq \beta \leq 1$ and for all $v,u \in R^n$.

Definition: Let v^* be the unique solution of $\min_v \{ z(v) \mid v \in H(W) \}$ and let $W^* = \{ u_i \mid \nabla z(v^*)(u_i - v^*) = 0, u_i \in W \}$. We refer to $H(W^*)$ as the *optimal face* of $H(W)$.

Definition: A set, W is said to be *compact* if any sequence of points in W contains a convergent subsequence whose limit is also in W. More explicitly, given a sequence $\{z^n\}$ in W, where W is compact, there exists a subsequence $\{z^n \mid n \in N'\}$ where $N' \subset N$ such that

$$z^n \rightarrow z^\infty, \quad n \in N' \text{ and}$$

$$z^\infty \in W.$$

In Euclidean space, Zangwill (1969, pp154) showed that compact sets correspond to closed and bounded sets. Thus a compact set must contain all of its edges and cannot extend infinitely in any direction. The points generated by most solution algorithms to the traffic assignment problems can be contained in such sets.

We then consider the following general nonlinear convex problem that includes various formulations of traffic assignment. This problem, P1 is to find v so as to minimise an objective function z(v).

$$P1 : \min \{ z(v) \mid v \in \mathcal{B} \}$$

where z(v) is a continuously differentiable convex function and $\mathcal{B}$ is the feasible region.

In the traffic assignment problem the function z(v) can have the following form:

$$z(v) = \sum_a \int_{v=0}^{v_a} c_a(v)dv$$

where $c_a(v)$ is a link cost.

The nonlinear convex problem, P1 over a bounded polyhedral set, $\mathcal{B}$ can be written in a

simpler way as follows:

$$v^* = \arg \min_{v \in \mathcal{B}} z(v)$$

3.2.2 General simplicial decomposition algorithm

Simplicial decomposition (SD) algorithms are based on Caratheodory's theorem which can be stated as follows.

Caratheodory's Theorem: Let $W \subset R^n$ be a nonempty set and let H(W) be its convex hull, then every $v \in H(W)$ lies in the relative interior of at least one of finite numbers of simplexes whose vertexes are points of W. The union of this collection of simplexes equals W.

For proof, see Rockafellar (1970, pp155-170).

The result of this theorem is that any point in a bounded polyhedral set, P can be described by a convex combination of the extreme points of P.

The general form of the simplicial decomposition (SD) algorithm for the solution of the problem, $\min_v \{z(v) | v \in \mathcal{B}\}$ is as follows:

General simplicial decomposition algorithm

Step 0:(Initialisation)

Iteration n=0

Identify a feasible point, $v^n \in \mathcal{B}$.

Set $W^n=\{v^n\}$.

Step 1:(Linear subproblem)

Let $c(v^n)=\nabla z|_{v=v^n}$ and

$u^n = \arg \min_u \{c(v^n)u : u \in \mathcal{B}\}$

If $c(v^n)(u^n - v^n) \geq 0$, stop: optimum solution is v^n.

Otherwise $W^{n+1} = W^n \cup \{u^n\}$

Step 2:(Master subproblem)

Let $v^{n+1} = \arg \min_v \{ z(v) : v \in H(W) \}$

n=n+1

Return to step 1.

In step 1, the linear subproblem is obtained by taking linear approximation of the objective function to obtain extreme points of $\mathcal{B}$. In traffic assignment, this corresponds to the all-or-nothing assignment to minimum cost routes. This is solved by finding a minimum cost path from each origin to each destination and assigning all the associated travel demand onto it. The computational time is taken mainly in the minimum path search because it examines many possible way to reach the destination from origin.

Some properties of the simplex generated by the linear subproblem are as follows:

1. If v is an element of n-1 simplex, H(W), then v can be expressed uniquely as an interior combination of the points, $u_1,...,u_n$,

$$v = \sum_{i=1}^{n} \beta_i u_i, \quad \sum_{i=1}^{n} \beta_i = 1, \ \beta_i \geq 0, \ \ i = 1,...,n$$

2. If v is an element of n-1 simplex, H(W), and the weight, β_i for some i=1,...,n is positive in the unique expression of v as a convex combination of $u_1,...,u_n$, then $H(u_1,...,u_{i-1}, v, u_{i+1},...,u_n)$ is also n-simplex.

3. If $H(u_1,...,u_n)$ is n-1 simplex, then $H(u_1,..,u_{i-1},u_{i+1},...,u_n)$, for each i is n-2 simplex.

Corollary to Caratheodory's theorem: if the demand feasible region of P1 denoted as $\mathcal{B}$ is bounded, then any element of $\mathcal{B}$ is expressible as a convex combination of its extreme points, of which there are only finite numbers. P1 can be rewritten in terms of the extreme points as

$$\text{P2: } \min_{\beta} \{ z(A\beta): \sum_{i=1}^{n} \beta_i = 1,\ \beta_i \geq 0,\ \ i = 1,\ldots,n \}$$

where n is the total number of extreme points, and u_i corresponding to column i of the matrix **A** is the i^{th} extreme point of $\mathcal{B}$.

The master subproblem of the simplicial decomposition algorithm is a reduced-dimensional version of original one that is solved over the convex hull of the points that have been generated by the linear subproblem. This problem can be solved using various techniques available for nonlinear programmes. Well known examples include uni-dimensional search techniques such as golden section search, Newton's method and the secant method. Von Hohenbalken (1977) used Newton directions, whilst Pang and Yu (1984) approximated the master program by a quadratic program. Hearn et al (1984) used the projected Newton method. Dembo and Tulowitzki (1988) used the PARTAN search direction. Shetty and Ben Daya (1988) proposed to use either a reduced gradient or a gradient projection method. Larsson and Patricksson (1991) solved this using either a scaled reduced gradient method or an approximated Newton's second order method.

3.3 REPRESENTATION OF SOME EXISTING ALGORITHMS USING SIMPLICIAL DECOMPOSITION

3.3.1 Introduction

In this section, we review some of existing algorithms, which are represented as simplicial decomposition forms. These algorithms are Frank-Wolfe variants and have been developed to overcome the slow convergence of the Frank-Wolfe algorithm. We will use an example of a simple figure to compare the progress of these algorithms.

3.3.2 Frank-Wolfe algorithm

The Frank-Wolfe algorithm is well recognised and widely adopted to solve large-scale traffic assignment. However, it has a slow zigzagging convergence in its later stages (Janson et al, 1987). This is mainly because search directions generated by solving the linear subproblem tend to be normal to the steepest descent directions of the objective function as the iteration proceeds. The Frank-Wolfe algorithm can be described in terms of a general simplicial decomposition algorithm. In this case, the linear subproblem is the same as for the general SD.

Frank-Wolfe algorithm as a SD algorithm

Step0:(Initialisation)

n=0

same as general simplicial decomposition algorithm

Step1:(Linear subproblem)

same as general simplicial decomposition algorithm

Step2:(Master subproblem)

Let $\beta^* = \arg\min_{\beta} \{ z(\beta u^n + (1-\beta) v^n) \mid 0 \leq \beta \leq 1 \}$

Set $v^{n+1} = \beta^* u^n + (1-\beta^*) v^n$

n=n+1

Return to step 1.

The convergence of this algorithm is depicted on the graph of the example (see Figure 3-1). At first, the point v^0 is obtained in the initialisation step. In the linear subproblem, u^0 is obtained by all-or-nothing assignment. This generated point lies at the extreme point of the feasible region. The internal point v^1 is obtained by minimising the objective function value between v^0 and u^0. In this case, a uni-dimensional search technique is used to minimise the original objective function. In the next iteration process, the extreme point u^1 is obtained by the linear subproblem and then internal point v^2 is calculated as the optimal linear combination of v^1 and u^1. As the iteration proceeds, v^n tends to the optimal value v^* and the descent direction becomes orthogonal to the gradient vector of the objective function and thus converges in a zigzag manner.

3.3.3 PARTAN search direction

The PARTAN technique (Luenberger, 1989, pp254-257 and Arezki and Van Vliet, 1990) introduces an extra line search into the Frank-Wolfe algorithm. After each iteration other than the first, a line search is calculated in the usual Frank-Wolfe direction and then a second one is calculated in the direction of the last but one iterate. This method can overcome the slow convergence approach in the final stages of the Frank-Wolfe algorithm. The PARTAN search algorithm can also be described in terms of a simplicial decomposition form as follows:

PARTAN search algorithm

Step0:(Initialisation)

same as general simplicial decomposition algorithm

n=0

Identify a feasible point, $v^0 \in \mathcal{B}$.

Step1:(Linear subproblem)

same as general simplicial decomposition algorithm

Step2:(Master subproblem)

If (n=0), Line-search as in the Frank-Wolfe to give v^{n+1}.

If ($n \geq 1$) then, PARTAN search direction as:

Let $\beta^* = \arg \min_{\beta} \{ z(\beta u^n + (1-\beta) v^{n-2}) : 0 \leq \beta \leq 1 \}$

Set $v^{n+1} = \beta^* u^n + (1- \beta^*) v^{n-2}$

n=n+1

Return to step 1.

The convergence of this algorithm is depicted in the Figure 3-2. Up to the iteration one, the process of convergence in the PARTAN search is the same as the Frank-Wolfe algorithm. However, from iteration two, the internal point is obtained by the linear combination of u^2 and v^0. This changed direction avoids the slow convergence at the later stages in the Frank-Wolfe algorithm.

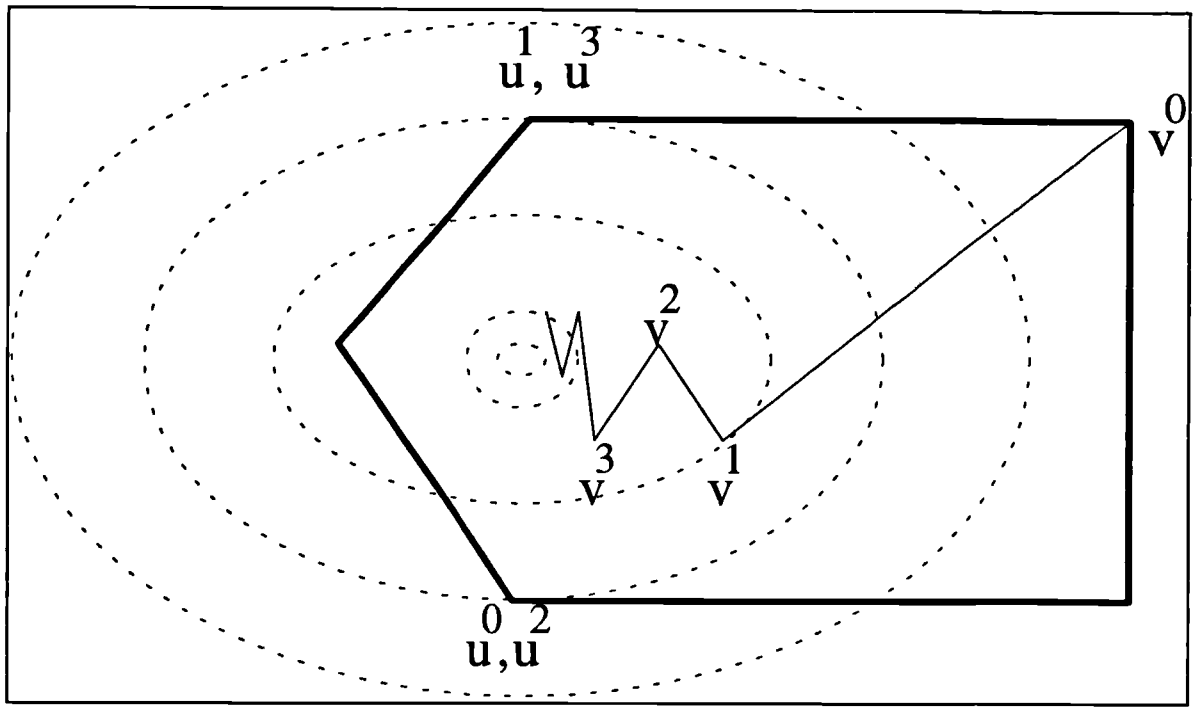


Figure 3-1. Graphical representation of the Frank-Wolfe algorithm

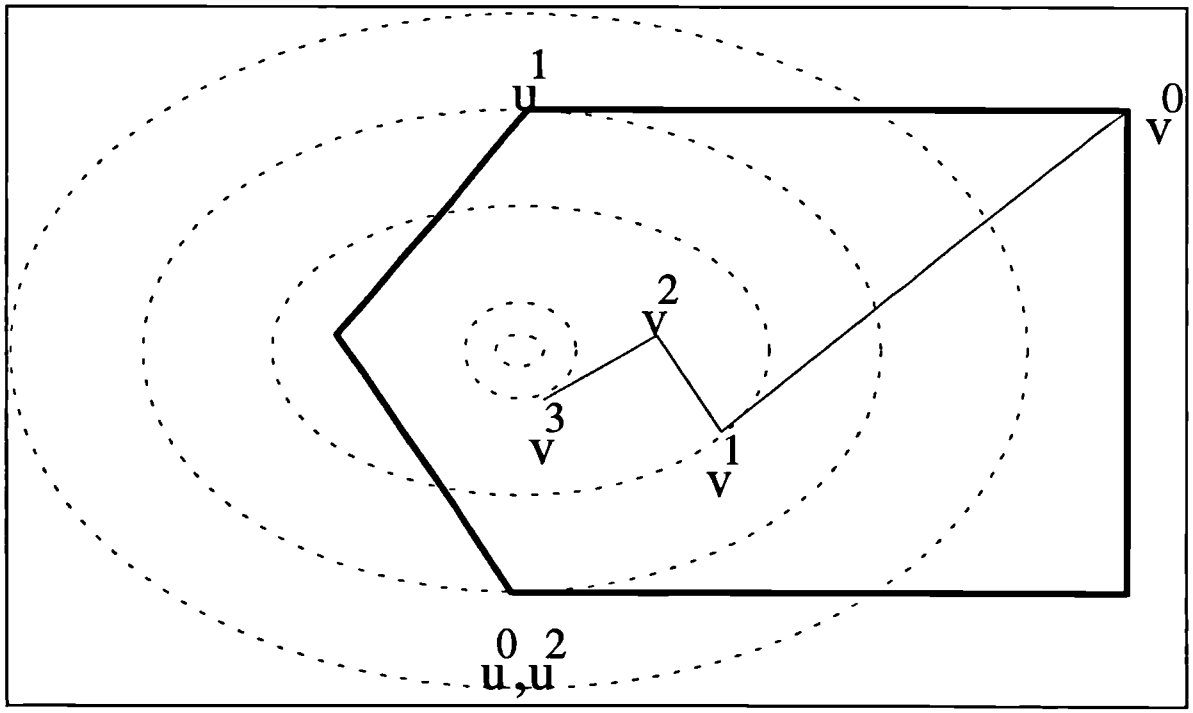


Figure 3-2 Graphical representation of PARTAN search direction

3.3.4 Fukushima's modified search direction (1984)

This algorithm modified the search direction by introducing components of earlier solutions. Every iteration a directional derivative value of the objective function is calculated to choose either the ordinary Frank-Wolfe direction or a modified one. This algorithm can be described in terms of general simplicial decomposition algorithm:

Fukushima's algorithm

Step0:(Initialisation)

same as general simplicial decomposition algorithm

n=0

Step1:(Linear subproblem)

same as general simplicial decomposition algorithm

Step2:(Master subproblem)

choose β^i, i=n-q,...,n, such that

$$\sum_{i=n-q}^{n}\beta^i = 1,\ \beta^i \geq 0,\ i = n-q,\ldots,n$$

and put

$$s^n = \sum_{i=n-q}^{n}\beta^i (u^i - v^i)$$

where q = min{n,L} -1, ie, q=n-1 for n ≤ L and q= L-1 for n>L, where L is the number of links in the network, put

$w^n = u^n - v^n$

Compute the directional derivatives

$$\tau_1^n = \nabla z(v^n)s^n / \|s^n\| \quad (\tau_1^n = 0 \text{ if } s^n = 0)$$

and

$$\tau_2^n = \nabla z(v^n)w^n / \|w^n\|$$

and set

$$d^n = \begin{cases} s^n, \text{if } \tau_1^n < \tau_2^n \\ x^n, \text{if } \tau_1^n = \tau_2^n \\ w^n, \text{if} \tau_1^n > \tau_2^n \end{cases}$$

where x^n is any vector on the line segment $[v^n, w^n]$.

Solve the uni-dimensional problem to obtain the step size β

$\beta^* = \arg \min_{\beta} \{v^n + \beta d^n\}$

and put

$v^{n+1} = v^n + \beta^* d^n$.

n=n+1

Go to the step 1.

3.3.5 Weintraub, Ortiz and Gonzalez's (1985) modified step size

This algorithm modifies the step-size defined as a form of compensation for deficiencies in the search directions of the Frank-Wolfe algorithm. The modification of this algorithm involves increasing the step size of the uni-dimensional optimisation while maintaining feasibility and assuring monotonic improvement in the objective function. This algorithm can be described in terms of a simplicial decomposition algorithm as follows.

Weintraub and Ortiz and Gonzalez's algorithm

Step0:(Initialisation)

same as general simplicial decomposition algorithm

n=0

Step1:(Linear subproblem)

same as general simplicial decomposition algorithm

Step2:(Master subproblem)

Let $\beta^* = \arg\min_{\beta} \{ z(\beta u^n + (1-\beta) v^n) : 0 \leq \beta \leq 1 \}$

Let $\beta^n = \max(1, \beta^* \eta^n)$

where η^n is a predetermined modification for the step size in iteration n.

Set $v^{n+1} = \beta^n u^n + (1- \beta^n) v^n$

n=n+1

Return to step 1.

3.3.6 Holloway's extension (1974)

In this method, the uni-dimensional search in the master subproblem of the Frank-Wolfe algorithm is replaced as follows:

Holloway's algorithm

Step0:(Initialisation)

same as general simplicial decomposition algorithm

n=0

Step1:(Linear subproblem)

same as general simplicial decomposition algorithm

Step 2: (Master subproblem)

Let $\beta^* = \arg\min_{\beta} \{ z(\sum_{i \in H W)} \beta^i u^i): \sum_{i \in H W)} \beta^i = 1, \ \beta^i \geq 0, \ \forall u^i \in W \}$

Set $v^{n+1} = \beta^* u^n + (1- \beta^*) v^n$

n=n+1

Return to step 1.

3.3.7 Wolfe's away step

Florian (1977) proposed the use of Wolfe's away step in the Frank-Wolfe algorithm. This step includes determining the maximum cost path from each origin to each destination and reducing the flows along the maximum cost path by transferring them proportionally to other paths that are used. The difficulty, however, is in identifying the maximum cost path because of the existence of cycles. The modified way of implementing the away's step is to adopt Holloway's restriction strategy where computations are restricted to previously generated extreme points.

3.3.8 Schittenhelm's algorithm

Schittenhelm's algorithm (1990) can be described in terms of simplicial decomposition as follows:

Schittenhelm's algorithm

Step0:(Initialisation)

same as general simplicial decomposition algorithm

n=0

Step1:(Linear subproblem)

same as general simplicial decomposition algorithm

Step 2: (Master subproblem)

Let $u^* = \arg\min_u \{c(v)u | u \in W\}$

Let $p^* = \arg\max_u \{c(v)u | u \in W\}$

Let $\beta^* = \arg\min_\beta \{ z(v^n + \beta (u^*-p^*)) \mid 0 \leq \beta \leq 1 \}$.

where u^* is a minimum cost point of W and p^* is a maximum cost point of W.

Set $v^{n+1} = v^n + \beta^* (u^* - p^*)$

n=n+1

Return to step 1.

The convergence progress of Schittenhelm's algorithm is depicted in Figure 3-3. In the initialisation step of iteration 0, the point v^0 is obtained. In the linear subproblem, the point u^0 is obtained by all-or-nothing assignment: this point lies at an extreme of the feasible region. In the master subproblem, the internal point v^1 is obtained by equilibrating the costs between v^0 and u^0 using a uni-dimensional search technique to minimise the original objective function. In iteration 1 of the linear subproblem, the new extreme point, u^1 is obtained and the set of retained points is augmented to, $W^1=\{v^0, u^0, u^1\}$. In the master subproblem, the minimum cost extreme point, $u^* = u^1$ and the maximum cost extreme point, $p^* = v^0$ of the retained set, W^1 are identified. A one-dimensional search technique is then used to obtain the internal point, v^2 by searching from v^1 in the direction u^1-v^0 so as to equilibrate the assignments. In iteration 2 of the linear subproblem, the new extreme point, u^2 is obtained by all-or-nothing assignment and the updated convex hull is made by the extreme points, v^0, u^0, u^1 and u^2. The master subproblem equilibrates the maximum point, $p^* = u^0$ and minimum point, $u^* = u^2$ in the convex hull, $H(v^0, u^0, u^1, u^2)$, and obtain the internal point v^3 by equilibrating between u^2 and u^0. This is repeated until the process finds an optimal point v^*.

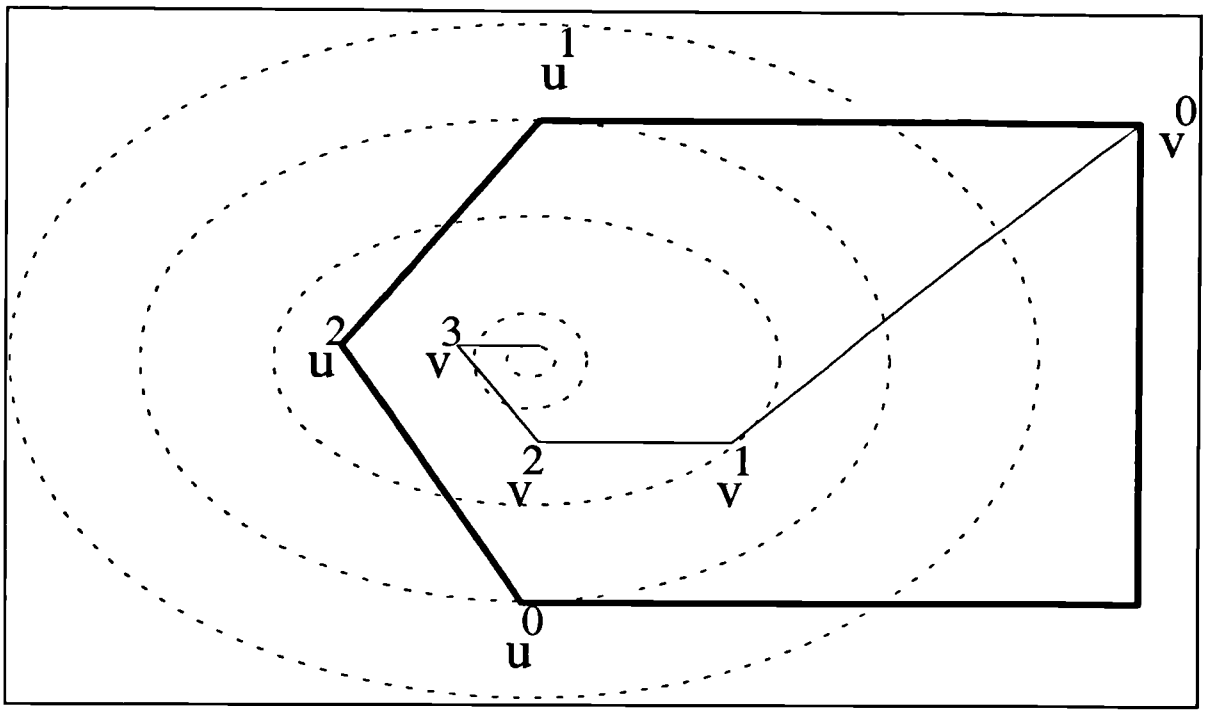


Figure 3-3 Graphical representation of Schittenhelm's algorithm

3.4 FOUR NEW ALGORITHMS

3.4.1 Introduction

In this section, we propose four new algorithms which have the simplicial decomposition form. They have been developed on the basis of observations of the convergence behaviour of existing algorithms. Algorithm 1 has been developed as an elaboration to Schittenhelm's algorithm. Algorithm 2 has been made by combining Schittenhelm's algorithm in the initial stage and Algorithm 1 in the later stages. Algorithm 3 combines column generation, which generates a predetermined number of extreme points, in the initialisation step and Schittenhelm's algorithm afterwards. Algorithm 4 is similar to Algorithm 3 but in the later stages, Algorithm 1 is used rather than Schittenhelm's algorithm. We now consider each of these algorithms in detail.

3.4.2 Algorithm 1

This algorithm (Lee 1992a) belongs to the simplicial decomposition class and can be specified as follows. Note that the steps 0 and 1 are the same as those of the general simplicial decomposition algorithm.

Algorithm 1

Step 0:(Initialisation)

Iteration n=0

Identify a feasible point, $v^n \in \mathcal{B}$. Set $W^n=\{v^n\}$.

Step 1:(Linear subproblem)

Let $c(v^n)=\nabla z\,|_{v=v^n}$ and

$u^n = \arg \min_u \{c(v^n)u : u \in \mathcal{B}\}$

If $c(v^n)(u^n - v^n) \geq 0$, stop: optimum solution is v^n.

Otherwise $W^{n+1} = W^n \cup \{u^n\}$

Step 2: (Master subproblem)

Let $v^{n+1} = \arg \min_v \{ z(v),\ v \in H(W^{n+1}) \}$

n=n+1

Return to step 1.

In the master subproblem, v^{n+1} is obtained by repeating the master subproblem of Schittenhelm's algorithm until the assignments are equilibrated approximately within the convex hull, $H(W^{n+1})$.

In Figure 3-4, the convergence of Algorithm 1 is depicted. In the initialisation step of iteration 0, the point v^0 is obtained. In the linear subproblem, the point u^0 is obtained by all-or-nothing assignment. In the master subproblem, the internal point v^1 is obtained by equilibrating the costs between v^0 and u^0. At that time, a one-dimensional search technique is used to minimise the original objective function. In iteration 1 of the linear subproblem, the new extreme point, u^1 is obtained and the generated extreme points, $W=\{v^0, u^0, u^1\}$ are retained in order to make a convex hull. In the master subproblem, the minimum extreme point, u^1 and the maximum extreme point, v^0 are identified in this convex hull, $H(v^0, u^0, u^1)$. A one-dimensional search technique is then used to obtain the internal point, v^2 between these points, v^0 and u^1. Up to this step, this algorithm is the same as Schittenhelm's algorithm. However, Algorithm 1 repeats this process of equilibration. In this case, it identifies the minimum point, u^1 and the maximum point, u^0 in the convex hull, $H(v^0, u^0, u^1)$. In this step, the one-dimensional search makes the problem converge to the optimal value.

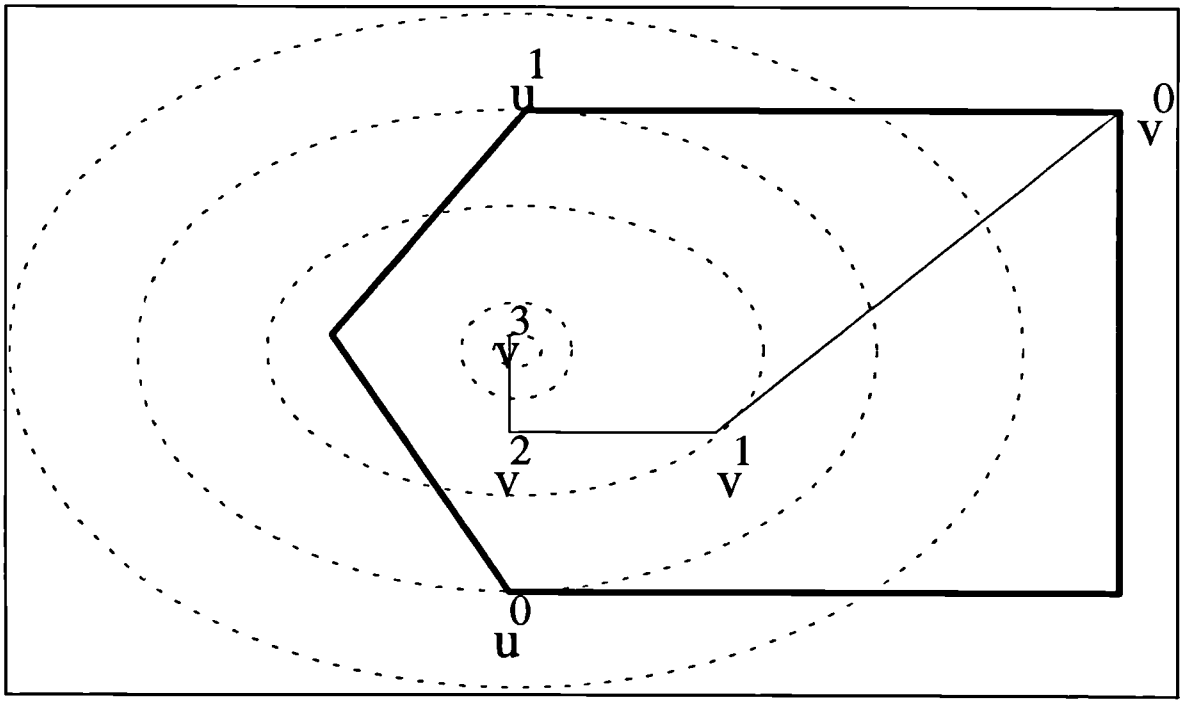


Figure 3-4 Graphical representation of Algorithm 1

Convergence of Algorithm 1 in the separable case

Theorem. Suppose that

(1) Algorithm 1 generates the sequence, $\{v^n\}_{n\in N}$, and a solution set, W^n, is in a non-empty convex compact set, $\mathcal{B}$, where $W^n \subseteq \mathcal{B}$.

(2) $c_a(v)$ is a positive strictly increasing continuous separable cost function. This in turn

$z(v) = \sum_a \int_{v=0}^{v_a} c_a(v)dv$ is a continuously differentiable strictly convex function.

Algorithm 1 either terminates at an equilibrium v* or generates a sequence $\{v^n\}_{n\in N}$ that has a limit point, v*, which is an equilibrium, as $n \to \infty$.

Proof (It is acknowledged that Dr JD Addison proved this theorem)

We assume that the algorithm does not terminate. As the algorithm does not terminate, it follows that $\nabla z(v^n) \neq 0$ for all n. Let $H = \overline{\cup_n H(W^n)} \subseteq \mathcal{B}$. H is closed, convex and compact. It is closed and convex by construction. As $\mathcal{B}$ is compact so is H. Let v^∞ be such that

$$z(v^\infty) = \inf_{v\in H} z(v) = \lim_{n\to\infty} z(v^n)$$

The second equality follows from the definition of H and the definition of v^n as the solution of the master subproblem on $H(W^n)$. Because the v^n is the successive solutions to the master subproblem, the sequence $\{z(v^n)\}$ decreases monotonically. The strict convexity of z means that v^∞ is unique.

We show that $\lim_{n\to\infty} v^n = v^\infty$. Suppose that $\{v^n\}$ has an accumulation point w, which must lie in H as H is closed. Then there is a subsequence v^{nj} which converges to w. Now by continuity, $z(w) = \lim_j z(v^{nj}) = z(v^\infty)$ and so $w = v^\infty$.

It remains to prove that v* lies in H where v* minimises z in $\mathcal{B}$, ie

$$z(v^*) = \inf_{v \in \mathcal{B}} z(v)$$

For v in H, $\nabla z(v^\infty)(v - v^\infty) \geq 0$. To see this, suppose that there is a u in H such that $\nabla z(v^\infty)(u - v^\infty) < 0$. Then the derivative of the function c(t) = z((1-t)v^∞ + tu) at t=0 shows it is decreasing there. Since the segment [v^∞, u] lies in H, this contradicts the choice of v^∞.

Suppose that v* is not in H. Since v* is not in H, it cannot be v^∞, so it follows from convexity that there exists u in $\mathcal{B}$ for which $\nabla z(v^\infty)(u - v^\infty) < 0$. Since $\lim_{n \to \infty} v^n = v^\infty$, it follows from the continuity of ∇z that there is a n such that $\nabla z(v^n)(u - v^n) < \nabla z(v^n)(u^n - v^n) < 0$. This contradicts the choice of u^n. We must have $v^* \in H$ and so $v^\infty = v^*$. Proof is completed.

3.4.3 Algorithm 2

This algorithm (Lee, 1992b) is a hybrid algorithm of the Schittenhelm and Algorithm 1. In the initial stages, Schittenhelm's algorithm is faster than Algorithm 1 because Schittenhelm's algorithm equilibrates the convex hulls in exact. Usually, the convex hull becomes bigger as the iterations progress so that the minimum point in the convex hull changes accordingly. On the other hand, Algorithm 1 performs better in the later stages because it saves computational time in the linear subproblem. This is because Algorithm 1 equilibrates the convex hulls exactly so that the algorithm repeats the linear subproblem and the master subproblem fewer times than Schittenhelm's algorithm. The proposed Algorithm 2 has some advantages of each of the Schittenhelm algorithm and Algorithm 1.

Algorithm 2

Step0:(Initialisation)

same as general simplicial decomposition algorithm

Step1:(Linear subproblem)

same as general simplicial decomposition algorithm

If a predetermined parameter is greater than iteration number of the master subproblem in the Schittenhelm algorithm, go to step 3.

Step 2: (Master subproblem in Schittenhelm's algorithm)

Let $u^* = \arg\min_u \{c(v)u | u \in W\}$

Let $p^* = \arg\max_u \{c(v)u | u \in W\}$

Let $\beta^* = \arg\min_\beta \{ z(v^n + \beta (u^*-p^*)) \mid 0 \leq \beta \leq 1 \}$.

where u^* is a minimum cost point of W and p^* is a maximum cost point of W.

Let $v^{n+1} = v^n + \beta^* (u^* - p^*)$

n=n+1

Return to step 1.

Step 2: (Master subproblem in Algorithm 1)

Let $v^{n+1} = \arg\min_v \{ z(v), \ v \in H(W) \}$

n=n+1

Return to step 1.

The convergence proof of Algorithm 2 is essentially the same as the Algorithm 1 as follows:

Convergence of Algorithm 2 in the separable case

Theorem. Suppose that

(1) Algorithm 2 generates the sequence, $\{v^n\}_{n \in N}$, and a solution set, W^n is in a non-empty convex compact set, $\mathcal{B}$, which is $W^n \subseteq \mathcal{B}$.

(2) $c_a(v)$ is a positive strictly increasing continuous separable cost function. This in turn

$z(v) = \sum_a \int_{v=0}^{v_a} c_a(v)dv$ is a continuously differentiable strictly convex function.

Algorithm 2 either terminates at an equilibrium v* or generates a sequence $\{v^n\}_{n\in N}$ that has a limit point, v* which is an equilibrium, as $n \to \infty$.

Proof (It is acknowledged that Dr JD Addison proved this theorem)

We assume that the algorithm does not terminate. As the algorithm does not terminate, it follows that $\nabla z(v^n) \neq 0$ for all n. Let $H = \overline{\cup_n H(W^n)} \subseteq \mathcal{B}$. H is closed, convex and compact. It is closed and convex by construction. As $\mathcal{B}$ is compact so is H. Let v^∞ be such that

$$z(v^\infty) = \inf_{v\in H} z(v) = \lim_{n\to\infty} z(v^n)$$

The second equality follows from the definition of H and the definition of v^n as the solution of the master subproblem on $H(W^n)$. Because the v^n is the successive solutions to the master subproblem, the sequence $\{z(v^n)\}$ decreases monotonically. The strict convexity of z means that v^∞ is unique.

We show that $\lim_{n\to\infty} v^n = v^\infty$. Suppose that $\{v^n\}$ has an accumulation point w, which must lie in H as H is closed. Then there is a subsequence v^{nj} which converges to w. Now by continuity, $z(w) = \lim_j z(v^{nj}) = z(v^\infty)$ and so $w = v^\infty$.

It remains to prove that v* lies in H where v* minimises z in $\mathcal{B}$, ie

$$z(v^*) = \inf_{v\in\mathcal{B}} z(v)$$

For v in H, $\nabla z(v^\infty)(v - v^\infty) \geq 0$. To see this, suppose that there is a u in H such that $\nabla z(v^\infty)(u - v^\infty) < 0$. Then the derivative of the function c(t) = z((1-t)v^∞ + tu) at t=0 shows it is decreasing there. Since the segment [v^∞, u] lies in H, this contradicts the choice of v^∞.

Suppose that v* is not in H. Since v* is not in H, it cannot be v^{∞}, so it follows from convexity that there exists u in $\mathcal{B}$ for which $\nabla z(v^{\infty})(u-v^{\infty})<0$. Since $\lim_{n\to\infty} v^n = v^{\infty}$, it follows from the continuity of ∇z that there is a n such that $\nabla z(v^n)(u-v^n)<\nabla z(v^n)(u^n-v^n)<0$. This contradicts the choice of u^n. We must have $v^* \in H$ and so $v^{\infty} = v^*$. Proof is completed.

3.4.4 Algorithm 3

This algorithm (Lee, 1992b) is a hybrid of column generation in the initial stages and Schittenhelm's algorithm in the later stages. Column generation generates a predetermined number of extreme points in the initialisation step. These points are used to generate a convex hull within which the algorithm equilibrates flows using Schittenhelm's method. This algorithm can reduce the computational time by generating a number of extreme points before equilibrating convex hulls in the initial stages.

Algorithm 3

Step 0:(Initialisation)

n=0

A feasible point $v^n \in \mathcal{B}$. Set $W=\{v^n\}$.

Step 0-1: (Column generation)

Repeat step 0-1 until a predetermined number of extreme points are obtained.

Let $c(v^n)=\nabla z\,|_{v=v^n}$ and

$u^n = \arg\min_u \{c(v^n)u : u \in \mathcal{B}\}$

$W = W \cup \{u^n\}$

$v^{n+1} = u^n$

n=n+1

Step 1: (The linear subproblem)

Let $c(v^n)=\nabla z\big|_{v=v^n}$ and

$u^n = \arg\min_u \{c(v^n)u : u \in \mathcal{B}\}$

If $c(v^n)(u^n - v^n) \geq 0$, Stop: optimum solution is v^n.

Otherwise $W = W \cup \{u^n\}$

Step 2: (Master subproblem as in Schittenhelm's algorithm)

Let $u^* = \arg\min_u \{c(v)u | u \in W\}$

Let $p^* = \arg\max_u \{c(v)u | u \in W\}$

Let $\beta^* = \arg\min_\beta \{ z(v^n + \beta (u^*-p^*)) \mid 0 \leq \beta \leq 1 \}$.

where u^* is a minimum cost point of W and p^* is a maximum cost point of W.

Let $v^{n+1} = v^n + \beta^* (u^* - p^*)$

$n=n+1$

Return to step 1.

3.4.5 Algorithm 4

This algorithm (Lee, 1992b) is a hybrid algorithm of column generation in its initial stages and Algorithm 1 in its later stages. As with Algorithm 3, this algorithm generates a predetermined number of extreme points in the initialisation step. The generated points make a convex hull and then the algorithm equilibrates the convex hulls by Algorithm 1.

Algorithm 4

Step 0:(Initialisation)

A feasible point $v^n \in \mathcal{B}$. Set $W=\{v^n\}$.

Iteration n=0

Step 0-1: (Column generation)

Repeat step 0-1 until a predetermined number of extreme points are obtained.

Let $c(v^n)=\nabla z\,|_{V=V^n}$ and

$u^n = \arg\min_u \{c(v^n)u : u \in \mathcal{B}\}$

$W = W \cup \{u^n\}$

$v^{n+1} = u^n$

$n=n+1$

Step 1: (The linear subproblem)

Let $c(v^n)=\nabla z\,|_{V=V^n}$ and $u^n = \arg\min_u \{c(v^n)u : u \in \mathcal{B}\}$

If $c(v^n)(u^n - v^n) \geq 0$, Stop: optimum solution is v^n.

Otherwise $W = W \cup \{u^n\}$

Step 2: (Master subproblem in Algorithm 1)

Let $v^{n+1} = \arg\min_v \{ z(v), v \in H(W) \}$

$n=n+1$

Return to step 1.

The convergence proof of Algorithm 4 is essentially the same as the Algorithm 1 as follows:

Convergence of Algorithm 4 in the separable case

Theorem. Suppose that

(1) Algorithm 4 generates the sequence, $\{v^n\}_{n\in N}$, and a solution set, W^n is in a non-empty convex compact set, $\mathcal{B}$, which is $W^n \subseteq \mathcal{B}$.

(2) $c_a(v)$ is a positive strictly increasing continuous separable cost function. This in turn

$z(v) = \sum_a \int_{v=0}^{v_a} c_a(v)dv$ is a continuously differentiable strictly convex function.

Algorithm 4 either terminates at an equilibrium v* or generates a sequence $\{v^n\}_{n\in N}$ that has a limit point, v* which is an equilibrium, as $n \to \infty$.

Proof (It is acknowledged that Dr JD Addison proved this theorem)

We assume that the algorithm does not terminate. As the algorithm does not terminate, it follows that $\nabla z(v^n) \neq 0$ for all n. Let $H = \overline{\cup_n H(W^n)} \subseteq \mathcal{B}$. H is closed, convex and compact. It is closed and convex by construction. As $\mathcal{B}$ is compact so is H. Let v^∞ be such that

$$z(v^\infty) = \inf_{v\in H} z(v) = \lim_{n\to\infty} z(v^n)$$

The second equality follows from the definition of H and the definition of v^n as the solution of the master subproblem on $H(W^n)$. Because the v^n is the successive solutions to the master subproblem, the sequence $\{z(v^n)\}$ decreases monotonically. The strict convexity of z means that v^∞ is unique.

We show that $\lim_{n\to\infty} v^n = v^\infty$. Suppose that $\{v^n\}$ has an accumulation point w, which must lie in H as H is closed. Then there is a subsequence v^{nj} which converges to w. Now by continuity, $z(w) = \lim_j z(v^{nj}) = z(v^\infty)$ and so $w = v^\infty$.

It remains to prove that v* lies in H where v* minimises z in $\mathcal{B}$, ie

$$z(v^*) = \inf_{v\in\mathcal{B}} z(v)$$

For v in H, $\nabla z(v^\infty)(v - v^\infty) \geq 0$. To see this, suppose that there is a u in H such that $\nabla z(v^\infty)(u - v^\infty) < 0$. Then the derivative of the function $c(t) = z((1-t)v^\infty + tu)$ at t=0 shows it is decreasing there. Since the segment $[v^\infty, u]$ lies in H, this contradicts the choice of v^∞.

Suppose that v^* is not in H. Since v^* is not in H, it cannot be v^∞, so it follows from convexity that there exists u in $\mathcal{B}$ for which $\nabla z(v^\infty)(u - v^\infty) < 0$. Since $\lim_{n\to\infty} v^n = v^\infty$, it follows from the continuity of ∇z that there is a n such that $\nabla z(v^n)(u - v^n) < \nabla z(v^n)(u^n - v^n) < 0$. This contradicts the choice of u^n. We must have $v^* \in H$ and so $v^\infty = v^*$. Proof is completed.

The advantage of the proposed new algorithms is to reduce the number of shortest path searches, which is the main time consuming part to solve the traffic assignment problem, as well as the number of the master subproblem. In particular, Algorithms 1, 2 and 4 reduce the number of shortest path searches whilst Algorithms 3 and 4 reduce the number of the master subproblem by using column generation. A sensitivity analysis can be readily performed due to the fact that much of the required information is stored routinely as part of the algorithm.

The related class of SD algorithms has the following advantages:

(1) The linear subproblem produces a set of feasible points within the convex hull of which the master subproblem calculates equilibrium.

(2) All the constraints of the original problem are satisfied by the linear subproblem; thus the generated points are always primal feasible.

(3) A sensitivity analysis may be easier than other algorithms due to retention of much of information that is required for such calculation, especially with respect to changes in O-D flows and network topology, and more complex models where traffic assignment arises as subproblems.

3.5 NUMERICAL EXAMPLE

The proposed simplicial decomposition algorithms, Schittenhelm's algorithm (1990) and the F-W algorithm have been coded in FORTRAN to test and compare their

performance on a NESS 286 personal computer running at 8 MHz clock speed, using the Microsoft compiler version 4.0 and mathematical coprocessor with 80287. The Sioux Falls network shown in Figure 3-5 (Vythoulkas, 1990) is used for this numerical test. The origin-destination matrix is given in Table 3-1 and the network data in Table 3-2. The separable Bureau of Public Roads (BPR, 1964) link cost function (see equation 2.24 in chapter 2) is used.

Dijkstra's algorithm (1959) is used in the shortest path routine. In the master subproblem of Schittenhelm's algorithm and the new algorithms a secant method is used to equilibrate the minimum and maximum cost paths in the retained set W. To store paths efficiently, the forward star data structure is used (see Sheffi 1984, pp125-129). The objective function, z(v) is used as a minimand (see section 2.5.2). As stopping criteria, either the relative difference of the objective function, z(v) value in the successive iterations as $\left|z(v^n) - z(v^{n-1})\right| \leq \varepsilon$, or the maximum iteration number of an algorithm are used.

Table 3-3 shows a summary of the performance of each algorithm. Note that there are two objective values: Beckmann's z(v) and the mean gap function, $\overline{G}(v)$ (see equation 2.16). However, the z(v) is only an appropriated measure here because in the master subproblem of each algorithm, z(v) is used to calculate the move size, β rather than $\overline{G}(v)$. Algorithm 1 terminates fastest at an objective value (or, z(v)) of 3923 vehicle-seconds in 154.9 seconds of CPU time. The furthest convergent point is an objective value of 3863 vehicle-seconds which is achieved by Algorithm 4. Algorithms 2, 3 and 4 have similar convergent points in terms of the number of column generation whilst the CPU time is taken more as the number of column generations increases.

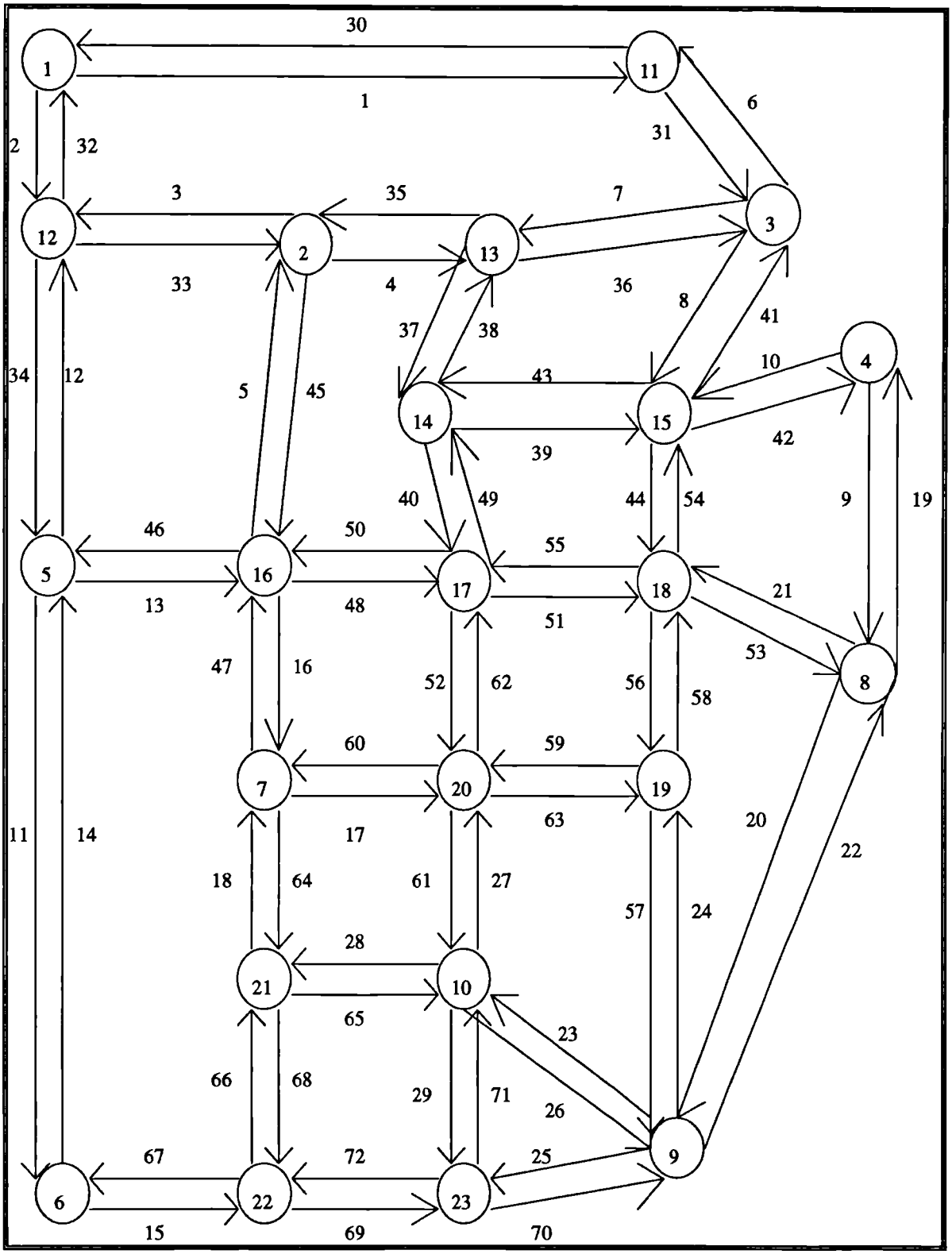


Figure 3-5 The Sioux Falls network

(Key: $\xrightarrow{n}$ link number n)

O-D pair	Demand
1-17	2900
2-19	2800
3-20	2600
4-20	2800
5-19	2600
6-17	2300
7-15	2200
8-13	2800
9-14	2700
10-15	2800
11-20	2600
12-18	2700

Table 3-1 Origin-destination pairs and demand.

Link	Capacity	Link	Capacity	Link	Capacity
1	3800	25	2400	49	2600
2	3200	26	2600	50	2800
3	3400	27	2700	51	2200
4	3000	28	2400	52	2500
5	3200	29	2800	53	3200
6	3400	30	3800	54	2700
7	3000	31	3400	55	2200
8	2700	32	3200	56	2700
9	2800	33	3400	57	3000
10	2400	34	3600	58	2700
11	3800	35	3000	59	2400
12	3600	36	3000	60	3000
13	3400	37	2800	61	2700
14	3800	38	2800	62	2500
15	3400	39	2600	63	2400
16	2800	40	2600	64	2400
17	3000	41	2700	65	2400
18	2400	42	2400	66	2400
19	2800	43	2600	67	3400
20	3200	44	2700	68	2400
21	3200	45	3200	69	3000
22	3200	46	3400	70	2400
23	2600	47	3800	71	2800
24	3000	48	2800	72	3000

Table 3-2 Network data

Figure 3-6 shows the objective function values for F-W, Schittenhelm's algorithms, and Algorithm 1 in terms of cumulative CPU time. In this case, Schittenhelm's algorithm runs faster than Algorithm 1 in the initial stages. However, in the later stages, Algorithm 1 runs faster than Schittenhelm's algorithm. That is because of the number of iterations in the master subproblem of Schittenhelm's algorithm. Schittenhelm's algorithm equilibrates only once the maximum and minimum paths in the retained set W for each origin-destination pair. This reduces computational time in the initial stages whilst this causes more computational time in the later stages because it results in more frequently calculation of minimum paths. On the other hand, Algorithm 1 overcomes this disadvantage of Schittenhelm's algorithm by equilibrating the maximum and minimum paths repeatedly until the retained minimum set is equilibrated. In addition, the most time consuming part of traffic assignment is the minimum path search. In the later stages, Algorithm 1 reduces the number of the minimum path searches.

Figure 3-7 shows the values of the cumulative CPU time in each of the Frank-Wolfe, Schittenhelm's algorithm and Algorithm 1 as the iterations proceed. The Frank-Wolfe algorithm has a constant slope which is less than for either of the other algorithms. That is because in each execution of the master subproblem of the Frank-Wolfe algorithm, the objective function is minimised only once in the master subproblem. Similarly, Schittenhelm's algorithm has an approximately constant slope. This is because the time taken in finding minimum and maximum paths is similar in each iteration. The slope of Algorithm 1 is not constant. The maximum CPU usage is during iteration 8, and the slope of Algorithm 1 is variable. In the initial stages, the slope of Algorithm 1 is higher than the other algorithms. However, as iterations proceed, the slope becomes lower than the others. In addition, Algorithm 1 terminates after fewer iterations than the other algorithms so the total CPU usage is less.

Figure 3-8 shows the results of Algorithms 2, 3 and 4 when one column generation

is used. Algorithm 3 performs better than the others in the initial stages whilst each of Algorithms 2 and 4 terminates to further points after 110 and 78 seconds respectively of CPU time. In Figure 3-9, the objective values of these algorithms are depicted when two column generations are used. Algorithms 3 and 4 perform better than when one column generation is used in the initial stages. The pattern in the convergence of these algorithms is similar in this case. The terminating points lie in the similar region. Figure 3-10 shows the results of these algorithms when three column generations are used. Algorithm 2 performs better than the others in the initial stages. After 142 CPU seconds, all of the algorithms have similar terminating points. In Figure 3-11, these algorithms are plotted when four column generations are used. Algorithm 2 again performs better than Algorithms 3 and 4 in the initial stages. After 100 seconds of CPU time, the pattern of convergence of these algorithms becomes similar. However, Algorithm 4 runs a further point.

Figure 3-12 shows the value of the objective function in Algorithm 2 according to the number of Schittenhelm's master subproblems performed before switching to Algorithm 1. Algorithm 2 runs quickly when Schittenhelm's master subproblem is used once. However, use of some iterations more than once of Schittenhelm's master subproblem leads to a further terminated point in the long run. Figure 3-13 shows the change of objective function in Algorithm 3 according to the number of column generations performed. Algorithm 3 with one and two column generations runs better in the initial stages whilst Algorithm 3 with three and four column generations terminates further after 90 seconds of CPU time. Figure 3-14 shows the value of the objective function in Algorithm 4 according to the number of column generations. Algorithm 4 with one column generation runs quickly in the initial stages. After 90 seconds of CPU time, these algorithms have similar patterns of convergence.

In summary, the new simplicial decomposition Algorithm 1 has been shown to run quickly when solving a separable equilibrium traffic assignment problem. The advantage of this algorithm is that it reduces the number of shortest path searches which are the main time-consuming part of solving the traffic assignment problem. In terms of furthest terminated point, Algorithm 4 reaches an objective value of 3863 vehicle-seconds whilst Algorithms 2 and 3 terminate at 3876 and 3878 vehicle-seconds respectively. Algorithm 3 performs very well to terminate at 3878 vehicle-seconds when two column generations are used initially. When fewer than two column generations are used, Algorithm 3 terminates at 3920 vehicle-seconds. Algorithm 4 has shown no evidence to speed up when column generation is used in the initial stage. The behaviour of Algorithm 4 differs according to whether or not column generation is used. However, the number of column generation used makes relatively little difference to the values in this test network beyond a small change in the CPU time used. In some cases, the ultimate value of the objective function was achieved well before the algorithm terminated.

Number of column generation	Algorithms	Iterations	CPU (Seconds)	Objective values	
				Beckmann z(v) (Vehicle-seconds)	Mean gap $\overline{G}(v)$ (Seconds)
	Frank-Wolfe	51	315.2	3922	0.012
	Schittenhelm	37	261.9	3922	0.026
	Algorithm 1	18	154.9	3923	0.017
1 column generation	Algorithm 2	40	159.6	3876	0.015
	Algorithm 3	50	351.3	3920	0.026
	Algorithm 4	40	257.8	3863	0.010
2 column generations	Algorithm 2	40	160.0	3876	0.015
	Algorithm 3	50	490.3	3878	0.025
	Algorithm 4	40	258.7	3863	0.010
3 column generations	Algorithm 2	40	160.0	3876	0.015
	Algorithm 3	50	481.0	3878	0.025
	Algorithm 4	40	259.7	3863	0.010
4 column generations	Algorithm 2	40	159.0	3876	0.015
	Algorithm 3	50	472.0	3878	0.026
	Algorithm 4	40	260.2	3863	0.010

Table 3-3 Summary of performance of each of algorithms
(Note * ; CPU time includes each of input and output times)

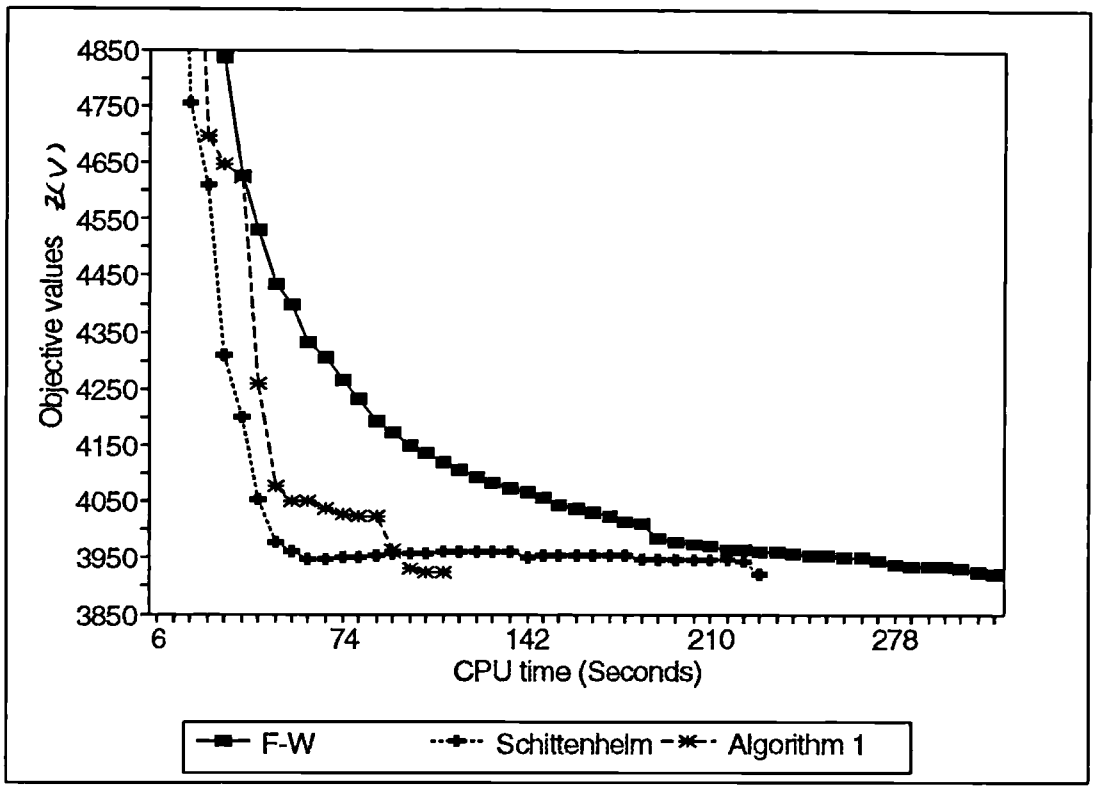


Figure 3-6 Results of Frank-Wolfe, Schittenhelm and Algorithm 1

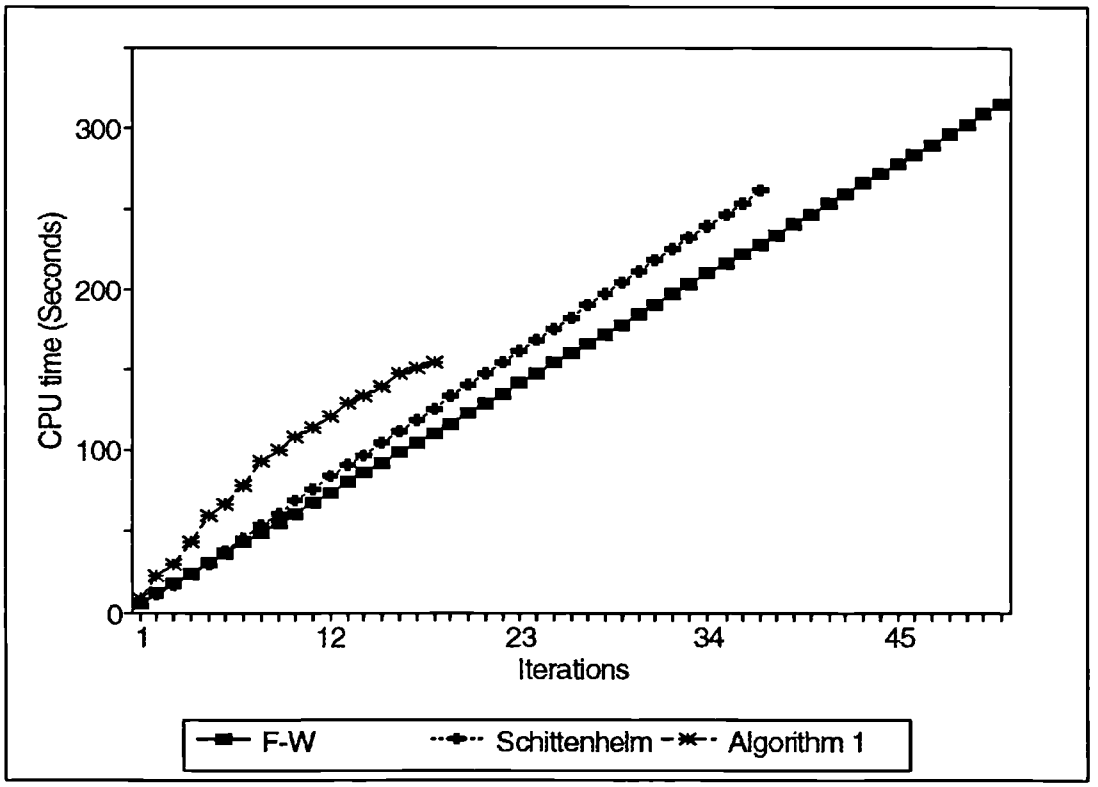


Figure 3-7 Comparison of CPU time in Frank-Wolfe, Schittenhelm and Algorithm 1

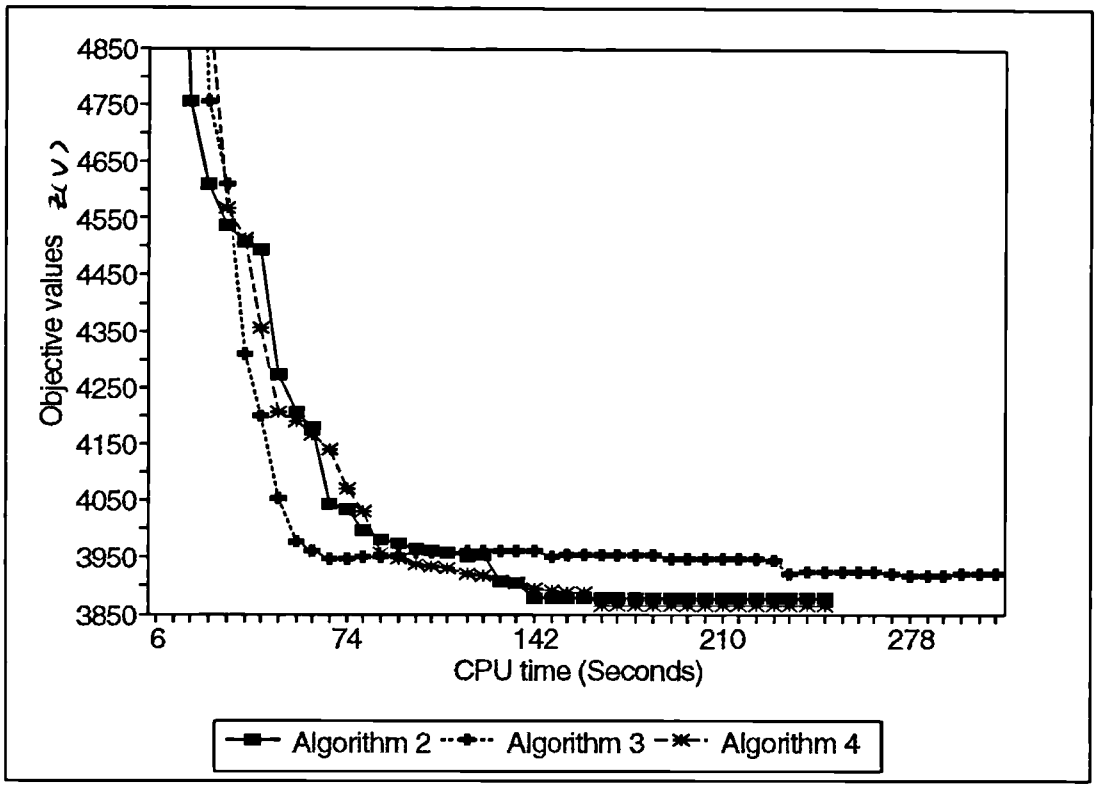


Figure 3-8 Comparison of Algorithm 2, 3, 4 when one column generation is used

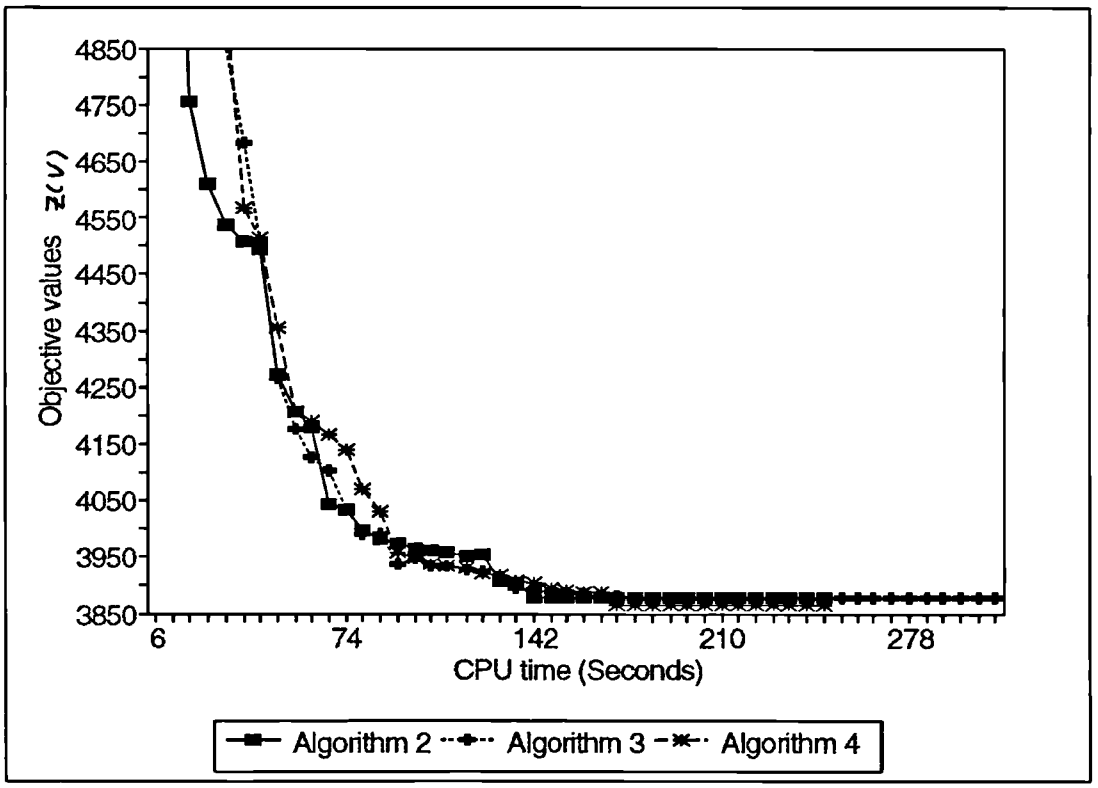


Figure 3-9 Comparison of Algorithm 2, 3, 4 when two column generations are used.

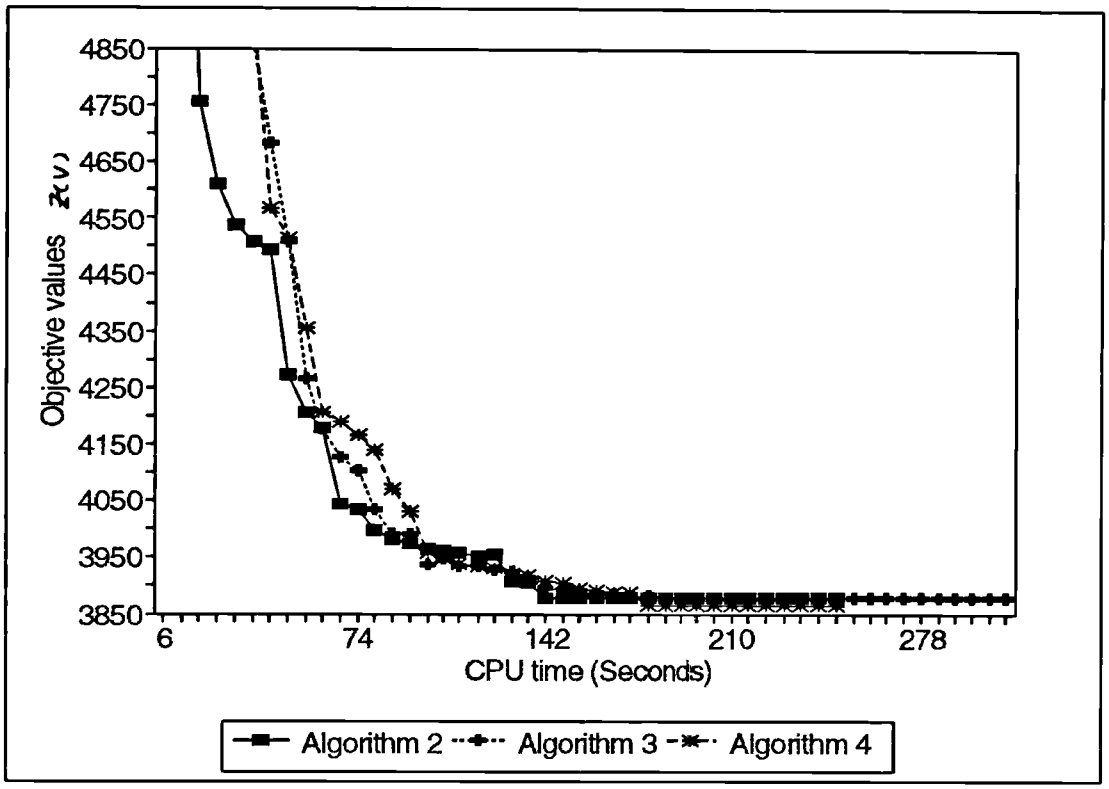


Figure 3-10 Comparison of Algorithm 2, 3, 4 when three column generations are used.

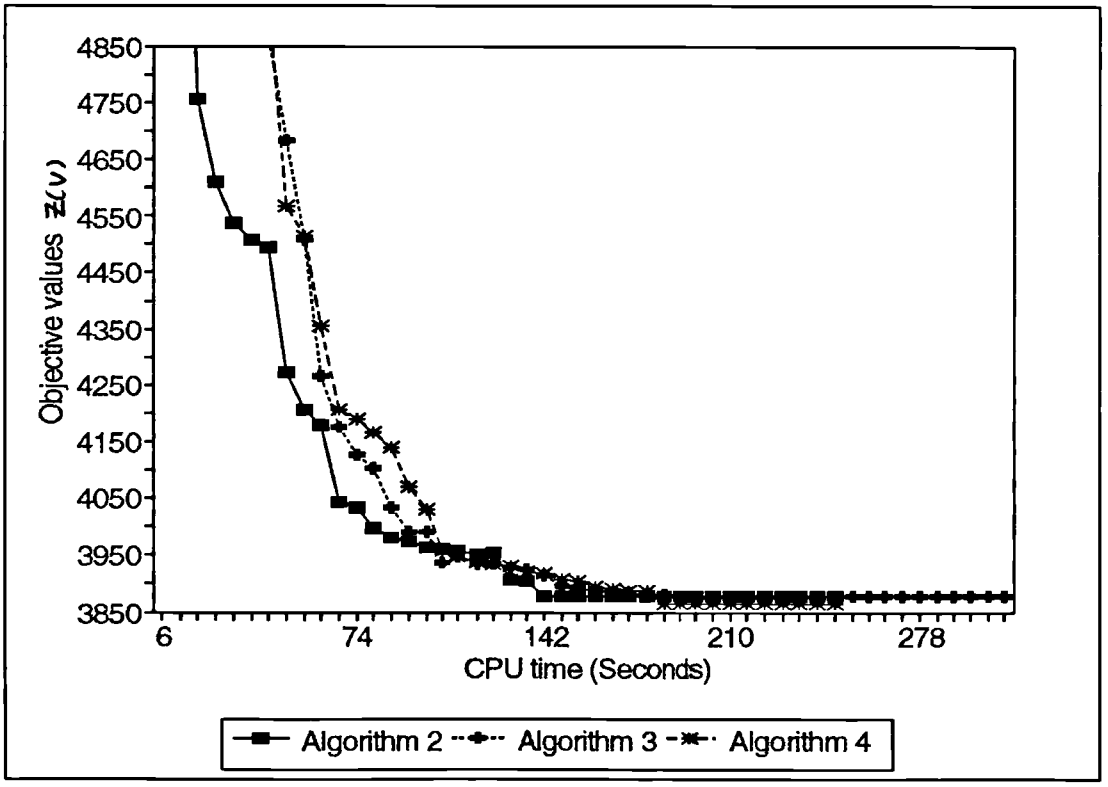


Figure 3-11 Comparison of Algorithm 2, 3, 4 when four column generations are used.

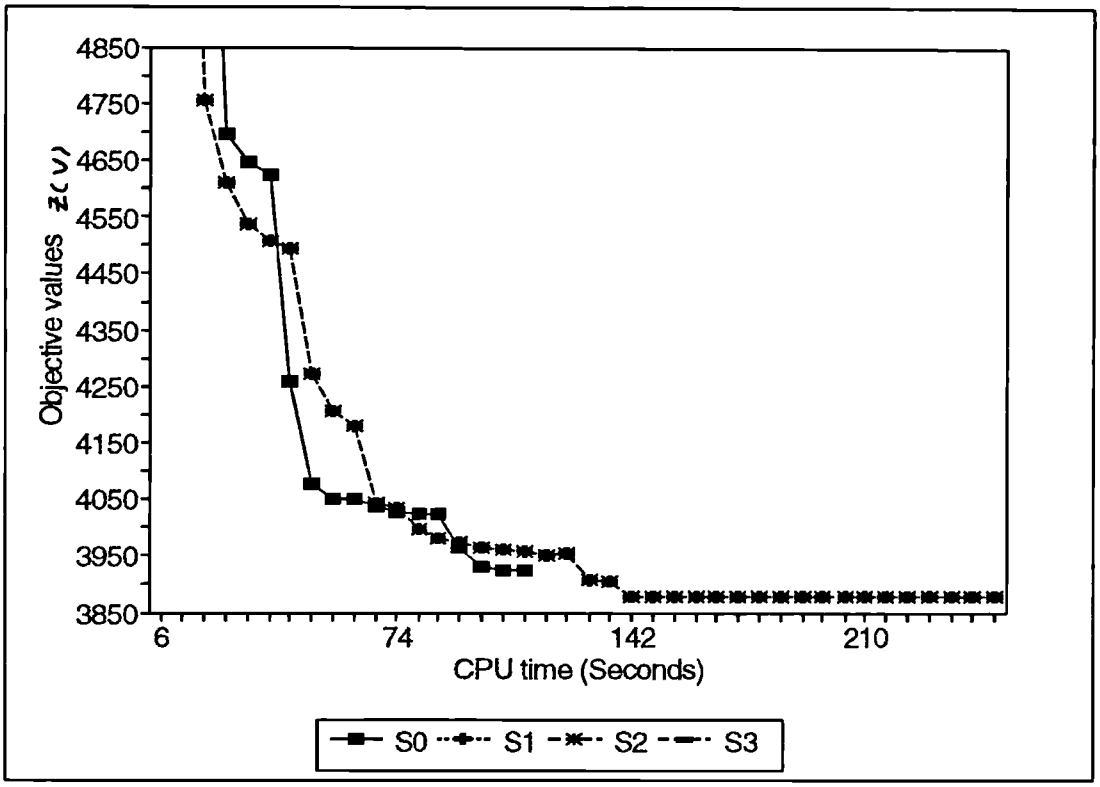


Figure 3-12 Results of Algorithm 2 in terms of the number of Schittenhelm's algorithm (Key: S0, S1,S2 and S3: one, two, three and four Schittehelm's algorithm is used respectively before switching to the master subproblem in Algorithm 1)

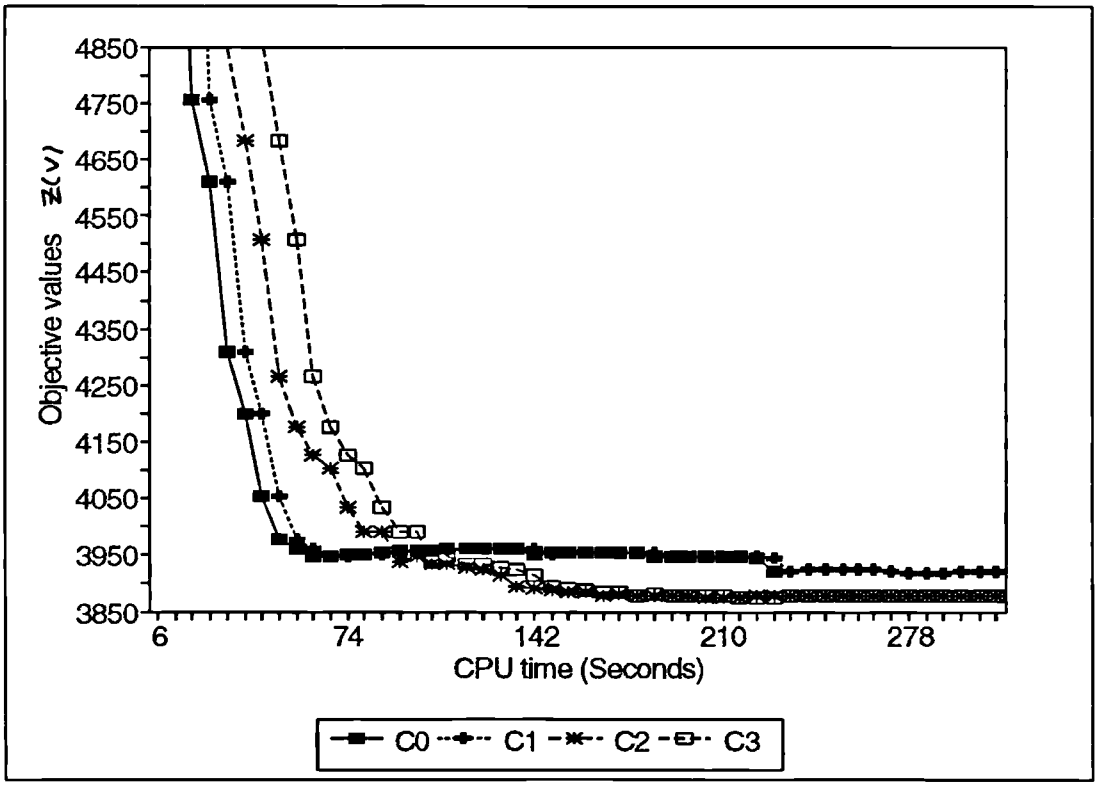


Figure 3-13 Results of Algorithm 3 in terms of the number of column generation (Key: C0,C1,C2 and C3: one, two, three and four column generations are used respectively as initial extreme points)

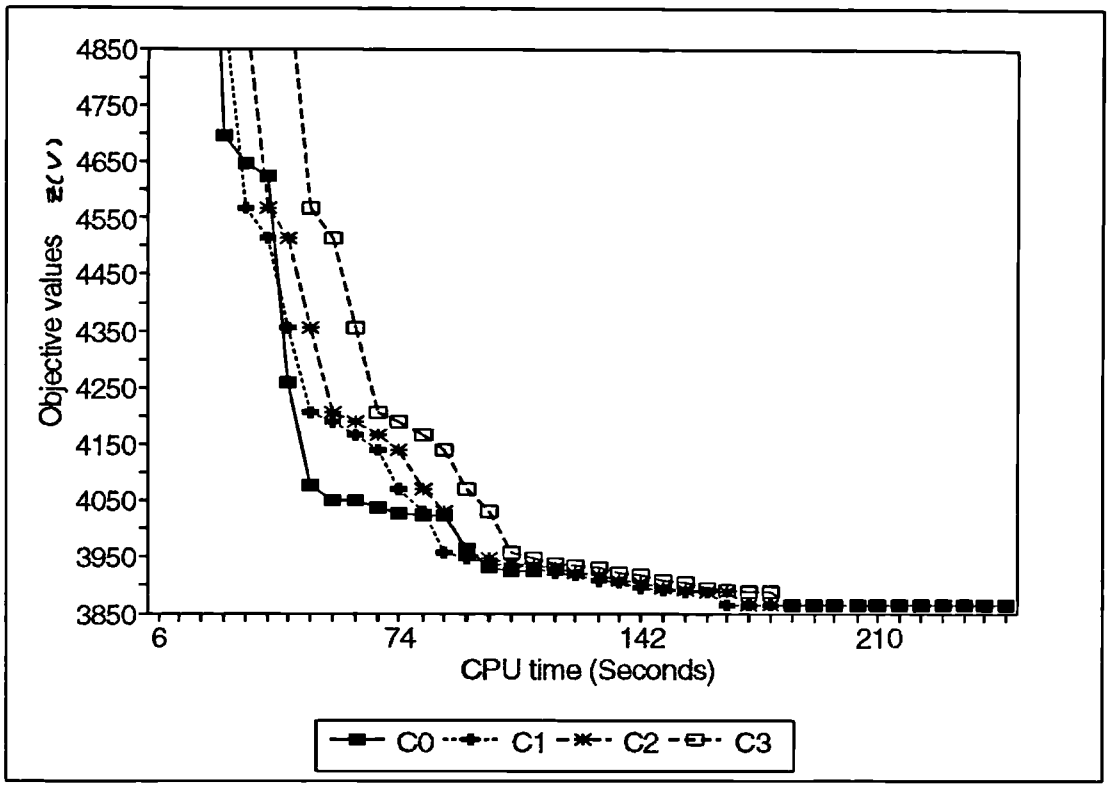


Figure 3-14 Results of Algorithm 4 in terms of the number of column generation (Key: C0,C1,C2 and C3: one, two, three and four column generations are used respectively as initial extreme points)

CHAPTER 4 TRAFFIC ASSIGNMENT WITH PRIORITY-CONTROLLED JUNCTION MODELLING

4.1 INTRODUCTION

This chapter shows how traffic assignment incorporating a detailed model of priority controlled junctions can be solved using a diagonalisation algorithm. Priority controlled junctions are modelled in which the capacity of the non-priority movements depends on the flows on the priority stream. In particular, the empirical capacity formula developed by Kimber and Coombe (1980) is used in the calculation of the capacity of the minor streams. Traffic assignment with priority controlled modelling is a non-separable problem because in some cases the capacity, and hence the cost, for each link depends on the flow on some other links. The convergence behaviour of this kind of assignment model is here investigated in terms of analytical and experimental analysis. This shows how detailed modelling plays an important role in influencing the stability and uniqueness of solutions.

4.2 CAPACITY CALCULATION

The concept of the capacity on a minor road differs from that of a major road. The capacity on a minor road is related to the controlling flow on the major road. There are two approaches to calculate the capacity on the minor road in terms of the flows on the major road. The empirical approach for this is based on analysing survey data whilst the gap-acceptance approach is based on the mathematical description of individual drivers' behaviour. In this section we review these two approaches in turn.

4.2.1 Empirical approach

The empirical approach consists of observing the capacity and possible explanatory variables at a number of intersections, and then analysing these data. One of the well-known models is PICADY (Priority Intersection Capacity And DelaY) which has been established by Transport Research Laboratory (TRL) in the UK. The PICADY program models capacities, queues and delays at priority-controlled junctions. In particular, three-arm junctions, four-arm junctions, and left-right and right-left staggered junctions are accommodated.

The PICADY model has a relationship expressing the capacity on the minor stream as a function of the junction geometry and flows on the major stream: this is due to Kimber and Coombe (1980). They derived formulae to estimate the capacity on the minor road as functions of traffic flows in the priority streams and functions of the geometric features of the junction. The formula for a three-arm junction is based on deriving the formulae of other types appropriately. The PICADY program incorporates these formulae so that the capacity, μ_a of the non-priority stream is related to the flows of priority flows as follows:

$$\mu_a = G_a (K_a + H_a - Y_a \sum_{b \in B_a} e_{ab} v_b)$$

where G_a, H_a and Y_a are constants which depend on the geometry of the junction, K_a is a constant which depends on the stream, e_{ab} is the coefficient for the appropriate major flow v_b, and B_a is the set of links b that has priority over link a.

Figure 4-1 shows a simple example of a priority controlled junction. Major flow v_b influences the capacity of the minor flow v_a. In other words, v_b is the controlling flow and v_a is the controlled flow. The capacity of link a is determined by the flow on link b as shown in Figure 4-2.

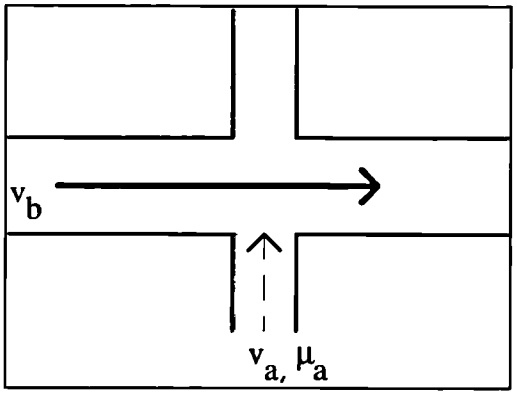


Figure 4-1 The major stream vs. minor stream

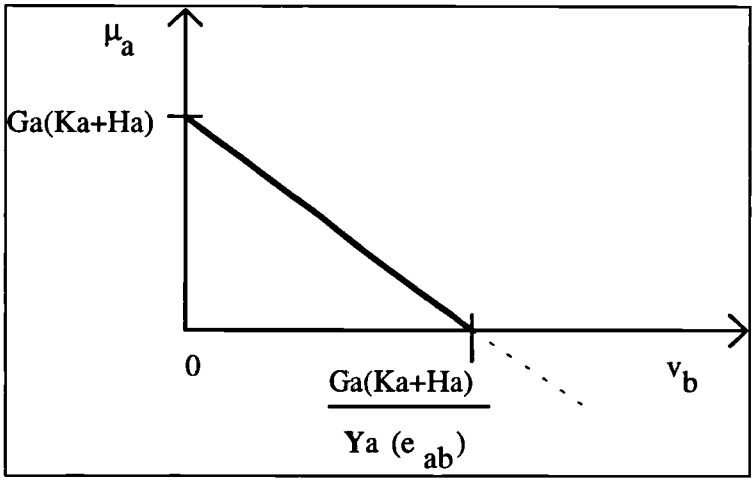


Figure 4-2 Relation of capacity on the minor stream and a flow on the major stream

As a detailed example, the PICADY model represents capacities for three-arm junctions such as that shown in Figure 4-3 as follows:

$$\mu_{bc} = G_{bc} \{ K_{bc} + H_{bc} - Y_{bc}(e_{bc,ac} v_{ac} + e_{bc,ab} v_{ab}) \}$$

$$\mu_{ba} = G_{ba} \{ K_{ba} + H_{ba} - Y_{ba}(e_{ba,ac}v_{ac} + e_{ba,ab}v_{ab} + e_{ba,ca}v_{ca} + e_{ba,cb}v_{cb}) \}$$

$$\mu_{cb} = G_{cb} \{ K_{cb} + H_{cb} - Y_{cb}(e_{cb,ab}v_{ab} + e_{cb,ac}v_{ac}) \}$$

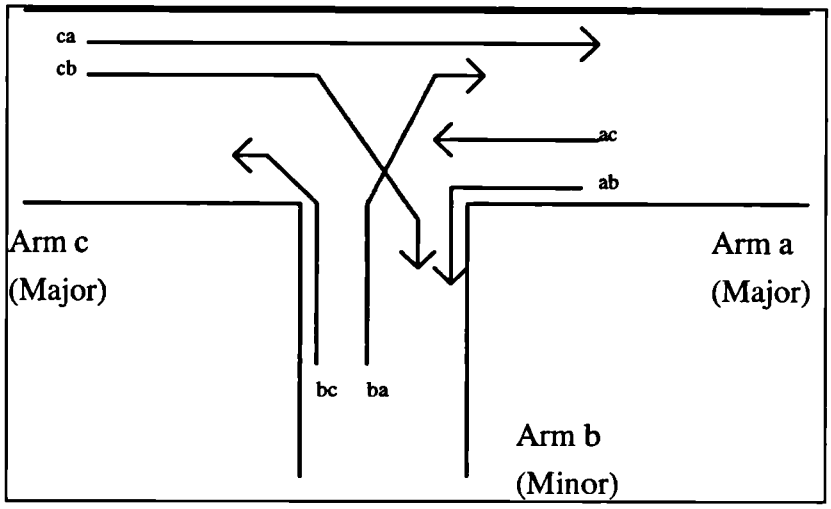


Figure 4-3 The layout of three-arm junction

One of the advantages of this empirical relationship for capacity is that it can be used in either a steady-state or a time dependent delay formulae. This also leads to an analysis of queue lengths and delays. PICADY uses Whiting's co-ordinate transformation method (Kimber and Hollis, 1979) to estimate queue length and delay. This transformation method is based on an approximation to M/G/1 queues in a simple queuing theory. This method estimates queue lengths and delays for a peak period during which the steady-state is not achieved, and arrival rates might possibly exceed capacity for some of the time period.

4.2.2 Gap-acceptance approach

The gap-acceptance approach is based on a probabilistic analysis of drivers' behaviour. Catchpole and Plank (1986) defined two important concepts of driver behaviour: consistency and homogeneity. We review the gap-acceptance approach using these concepts of driver behaviour. A driver on the minor road, on reaching the head of the queue, can choose to cross (or merge with) the major stream only if the time to the next major arrival is greater than a certain value (*critical gap*). If delayed, this driver

either retains the same critical gap for subsequent gaps (headways) in the major stream, in which case the driver is *consistent* in his choice of critical gap, or chooses a new independent critical gap, in which case the driver is *inconsistent.* After the departure of the lead vehicle, the next lead vehicle moves up to the head of queue, and the process of rejecting and accepting major stream gaps is repeated. Drivers on the minor stream may choose critical gaps either from the same distribution as others, in which case the drivers are *homogeneous* with respect to critical gaps, or from different distributions depending on the driver and vehicle type, in which case the drivers are *heterogeneous.*

Four distinct models arise according to the possible combinations of consistency and homogeneity.

Consistent and homogeneous model

Given the headway distribution, h(t) of the major flow v_b, it is possible to derive a capacity formula. The capacity, μ_a on the minor road can be expressed as the ratio of the mean number of vehicles inserting into a single gap over the mean length of a gap:

$$\mu_a = v_b \int_0^\infty h(t)n(t)dt$$

where n(t) is the number of vehicles able to leave the queue during a gap duration t.

Louah (1991) reviewed the capacity by classifying this according to whether n(t) is a discrete or a continuous function. In the *discrete gap-acceptance model,* one vehicle can leave if the gap duration is between t_c and $t_c + t_f$, where t_f is the time necessary for the second vehicle in the queue to reach the give-way line after the first one has left the queue, which is called the *follow-up time* or the *move-up time.* Two vehicles can insert

into a gap in the range $[t_c + t_f , t_c + 2t_f)$ and so on (see Figure 4-4), so that n vehicles can insert into a gap in the range $[t_c + (n-1)t_f, t_c + nt_f)$. On the other hand, in the *continuous gap-acceptance model*, this step function can be approximated with a continuous line such that no vehicle can insert into a gap until a threshold t_0 and a non-integer number of vehicles varying linearly with t beyond t_0 can insert into a gap with a slope equal to the saturation flow $1/t_f$. The correspondence between the two models occurs when $t_0 = t_c - t_f / 2$.

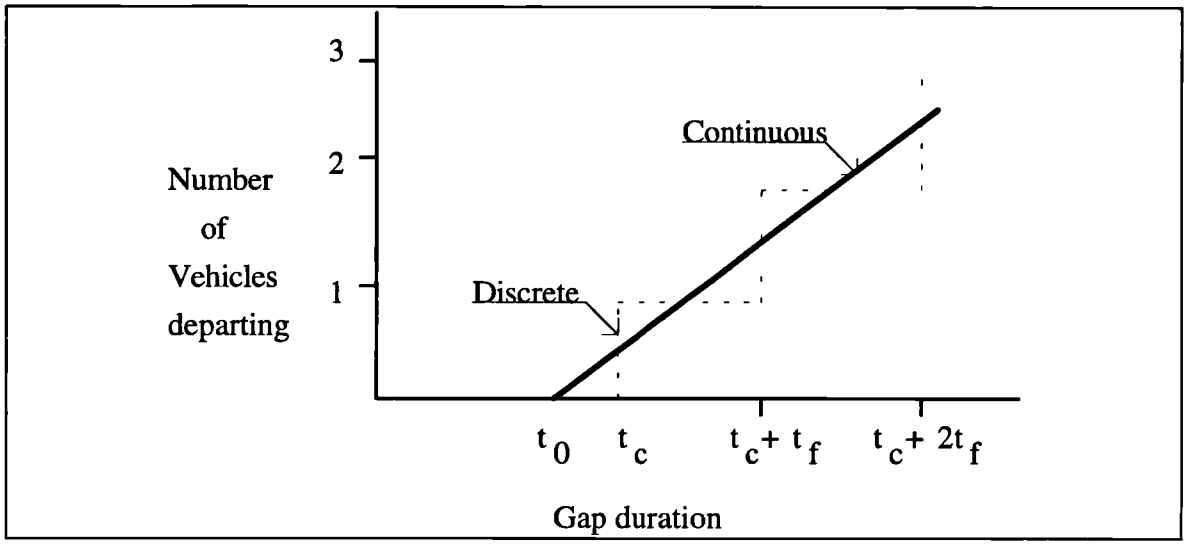


Figure 4-4 Number of vehicles able to insert into a gap (Source: Louah, 1991)

From the discrete n(t) function, we find the capacity on the minor road to be:

$$\mu_a = v_b \sum_{i=1}^{\infty} i \int_{t_c+(i-1)t_f}^{t_c+it_f} h(t)dt$$

From the continuous n(t) function, we get the capacity on the minor road to be:

$$\mu_a = v_b \int_{t}^{\infty} \left(\frac{t - t_0}{t_f} \right) h(t)dt$$

If it is assumed that the headway distribution, h(t) is exponential, then the capacity formula in the discrete case is obtained:

$$\mu_a = \frac{v_b \exp(-v_b t_c)}{1-\exp(-v_b t_f)}$$

The capacity formula in the continuous case is obtained:

$$\mu_a = \frac{\exp(-v_b t_0)}{t_f}$$

The former of these two formulae is the basis of the German guidelines and the latter one is US Highway Capacity Manual (1985, pp10.32 - 10.37). A formula generalised by the discrete one is the regular-random distribution. This combines a proportion P of bunched vehicles with a uniform headway H, and the remaining (1-P) vehicles with a shifted exponential distribution. The particular case when P=vH leads to Tanner's formula (1962) (See section 2.6.3.1).

Inconsistent and homogeneous model

Each minor road vehicle samples from a common critical gap distribution. For each decision concerning the acceptability of a major-stream gap, a new critical gap is sampled independently from this distribution.

Evans, Herman and Weiss (1964) solved this problem for fixed move-up time and for arbitrary distributions of critical gap and major headway. Plank and Catchpole (1984) gave a solution for arbitrary move-up time, critical gap and headway distributions.

Consistent and heterogeneous model

Each individual minor vehicle has a constant critical gap which is distributed between the population of vehicles. This is usually treated as a special case of the inconsistent and heterogeneous model.

Inconsistent and heterogeneous model

Each minor vehicle samples from a critical-gap distribution. The distribution used depends on the type of vehicle involved. This model includes each of the previous models as a special case. Catchpole and Plank (1986) developed a general formula for capacity.

4.3 DELAY FUNCTIONS

Delay at priority junctions consists of two main components: geometric and congestion delay. The geometric delay arises from the layout of the junction whilst the congestion delay arises from the vehicle by vehicle interactions. Kimber, Summersgill and Burrow (1986) defined the *geometric delay* as the delay that a vehicle would incur if it passed through the junction in complete isolation, but the driver does not know this in advance. Kimber et al (1986) defined the *congestion delay* as that delay generated by the queuing and service processes for non-priority vehicles. For the remainer of this section, we consider only the congestion delay.

Tanner's delay formula (1962) is obtained by assuming that arrivals on each of the major and minor roads are Poisson, a constant minimum headway for each of major and minor road vehicle, and that a fixed critical gap in major road traffic is acceptable to all minor road vehicles: this is a consistent and homogeneous model. For the resulting capacity and delay formulae, see section 2.6.3.1.

The steady-state delay, however, is valid only when the mean arrival rates are stationary. If traffic exceeds the capacity, then no steady-state delay can arise: this non-

stationary condition occurs often in the real world. For the non-stationary condition, Kimber and Hollis (1979) developed a sheared delay formula that combines a steady state stochastic delay with a deterministic queuing delay. The steady state delay can be obtained by queuing theory. Kimber et al (1979) derived the steady state formula for the M/M/1 and M/G/1 cases. The deterministic queuing delay represents the growth and decay of the queue for the condition that traffic flow exceeds the capacity. Figure 4-5 shows that the sheared delay is transformed such that the distances Y and Z of steady state and deterministic delays respectively are equal.

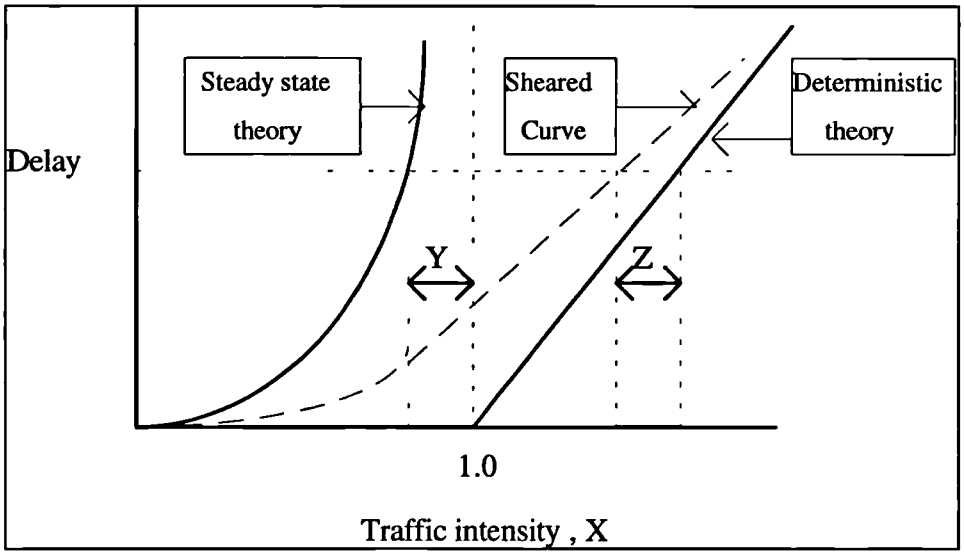


Figure 4-5 The sheared delay formula (Source: Kimber and Hollis,1979)

The formula due to Kimber and Hollis (1979) for sheared delay is:

$$d_a = \sqrt{(A^2 + B)} - A$$

where

$$A = \frac{t}{2}(1-X) - \frac{1}{\mu}(L_0 - C + 2)$$

$$B = \frac{4}{\mu}\left[\frac{t}{2}(1-X) + \frac{XtC}{2} - \left(\frac{L_0+1}{\mu}\right)(1-C)\right]$$

in which X is traffic intensity, t is duration, L_0 is initial queue length and C is a constant.

Assume that the arrival distribution of vehicles is Poisson and service is regular, then delay in the minor stream for steady state, according to the Pollaczek-Khinchine formula is:

$$d_a(v_a,\mu_a)=\frac{1}{2\mu_a}\left[\frac{X_a}{1-X_a}\right]$$

where $X_a = v_a / \mu_a$

For the initial stages of traffic assignment, the traffic flow often exceeds the capacity for some links. To deal with this case, the original Pollaczek-Khinchine formula can be extended for the oversaturated region. In this region, an increase in the flow will cause a big increase of the delay. In this study, we will use the linearly approximated delay formula about an intensity X* as (see Figure 4-6):

$$\bar{d}(X)=\begin{cases}\dfrac{X}{2\mu(1-X),} & X\le X^* \\ \dfrac{1}{2\mu}\left[\dfrac{X^*}{1-X^*}+\dfrac{(X-X^*)}{2(1-X^*)^2}\right], & (X>X^*)\end{cases}$$

which gives the objective function of traffic assignment (see section 2.5.2) as follows:

$$z(X)=\begin{cases}\dfrac{1}{2\mu}[-\ln(1-X)-X], & X\le X^* \\ \dfrac{1}{2\mu}\left[-\ln(1-X^*)-X^*+\dfrac{(X-X^*)^2}{4(1-X^*)^2}\right], & (X>X^*)\end{cases}$$

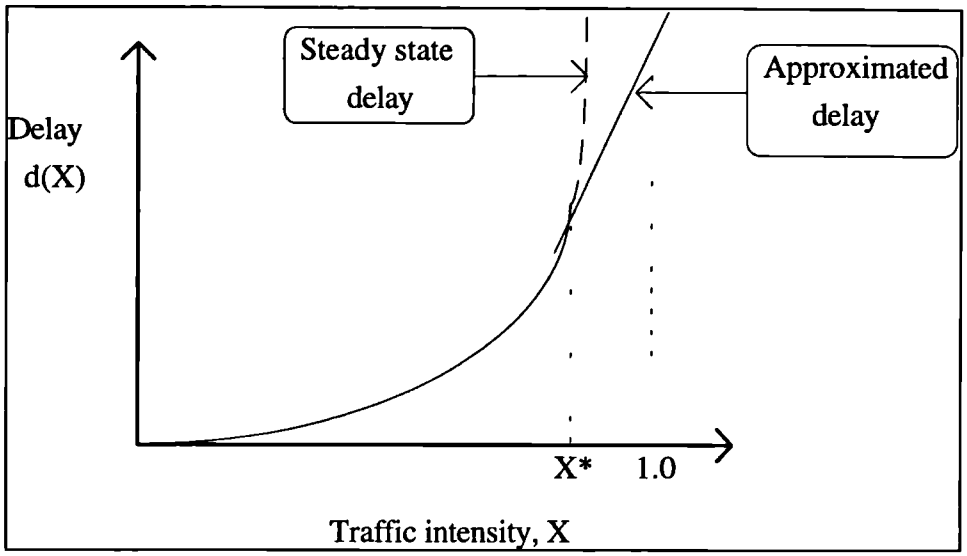


Figure 4-6 Approximated delay curve for junction delay.

4.4 FORMULATION

We will adopt the mathematical model proposed by Smith (1982) in order to formulate priority controlled junction modelling in road traffic assignment. The network consists of a set of n nodes and a set of L links. Each link has a start and a terminal node. We can group links according to their terminal nodes. Let junction n be

$$J_n = \{l_{a1}, l_{a2}, \ldots, l_{am_n}\}$$

corresponding to the set of links terminating at node n_a, where m_n is the number of links in junction n. Then the set of links, L is

$$L = J_1 \cup J_2 \cup \ldots \cup J_n$$

In road networks, the journey time is spent at junctions and on links. We therefore represent that the link cost is a weighted combination of running time on links and junction delays. The cost incurred in using link a, $T_a(\mathbf{v})$, is a weighted summation of link cost, $c_a(v_a)$ and junction delay, $d_a(\mathbf{v})$ (see section 2.6.4):

$$T_a(\mathbf{v}) = t_0 + \gamma[\, c_a(v_a) - t_0] \; + \; (1-\gamma)\, d_a(\mathbf{v})$$

where γ represents a relative magnitude of two delays and a user's perception on travel

cost.

In this study, link cost is calculated from the BPR (1965) function steady state formula. Junction delay, $d_a(\mathbf{v})$, depends on the junction type, and is calculated from the approximated Pollaczek-Khinchine formula (see section 4.3).

In this cost function, the link cost, $c_a(v_a)$ is only a function of the flow on link a, whereas the junction delay, $d_a(\mathbf{v})$ is a function of several flows approaching junction a in order to reflect any interacting movements. This relaxation makes the cost function non-separable:

$$\frac{\partial T_a}{\partial v_b} \neq 0 \qquad \text{for some } a \neq b$$

This non-separability can be accommodated by Smith's variational inequality formulation of an equilibrium assignment. Let $\mathcal{B}$ and $\mathcal{S}$ be the demand and supply feasible sets respectively: these are convex subsets of R^m (see section 3.2). Then $\mathbf{v} \in \mathcal{S} \cap \mathcal{B}$ is a Wardrop equilibrium if and only if

$$\mathbf{u} \in \mathcal{B}$$
$$\mathbf{T}(\mathbf{v}) \cdot (\mathbf{v} - \mathbf{u}) \leq \mathbf{0}$$

The equilibrium route choice model allocates flow to paths in the network so that at the solution, used paths between each o-d pair have the same travel costs, and unused paths have higher costs. Smith (1982) proved that if the cost function is monotone and continuous in the feasible region, the set of Wardrop equilibria is convex.

4.5 SOLUTION METHOD

An iteration of a *dynamic adjustment process* for solving this combined priority controlled junction modelling and traffic assignment is the combination of the calculation

of minor road capacities for the given feasible traffic flows, and the assignment of travel demand using these minor road capacities. The dynamic adjustment process continues until a convergence criterion is satisfied. In this dynamic adjustment process we adopt a diagonalised procedure because capacities on minor links are calculated whilst traffic flows are assumed to be fixed, and traffic flows are calculated whilst minor road capacities are fixed and so on. An existing diagonalisation procedure can therefore be used and the following is a diagonalisation of Algorithm 1.

A diagonalisation of Algorithm 1 for the combined priority control and traffic assignment

Step 0: (Initialisation)

Iteration n = 0

Identify a feasible point, $\mathbf{v}^n \in \mathcal{S}$

Set $W^n = \{\mathbf{v}^n\}$

Step 1: (Linear subproblem)

1-1. Calculate minor road capacities based on the PICADY formula

$$\mu_a^n = G_a(K_a + H_a - Y_a \sum_{b \in B_a} e_{ab} v_b^n)$$

1-2. Calculate the total link cost

$$T_a(\mathbf{v}^n, \mu_a^n) = t_0 + \gamma[c_a(v_a^n) - t_0] + (1-\gamma)\bar{d}_a(\mathbf{v}^n, \mu_a^n)$$

where $c_a(v_a^n)$ is the BPR function and $\bar{d}_a(\mathbf{v}^n, \mu_a^n)$ is the approximated Pollaczek-Khinchine formula

1-3. Perform all-or-nothing assignment based on the total link cost

$$\mathbf{u}^n = \arg \min_{\mathbf{u}} \{\mathbf{T}(\mathbf{v}^n, \mu^n) \cdot \mathbf{u} \ ; \ \mathbf{u} \in \mathcal{S}\}$$

1-4. Convergence test

If $G(\mathbf{v}^n, \mu^n) = \mathbf{T}(\mathbf{v}^n, \mu^n) \cdot (\mathbf{v}^n - \mathbf{u}^n) \approx 0$, stop: optimum solution is $\mathbf{v}^n$

Otherwise $W^{n+1} = W^n \cup \{\mathbf{u}^n\}$

Step 2: (Master subproblem)

Let $\mathbf{v}^{n+1} = \arg \min_{\mathbf{v}} \{G(\mathbf{v}, \mu); \ \mathbf{v} \in H(W^{n+1})\}$

n=n+1

Return to step 1.

The other simplicial decomposition algorithms introduced in chapter 3 can also be represented as a diagonalisation form as the case of Algorithm 1. For example, the Schittenhelm algorithm is represented as a diagonalisation form by replacing the master subproblem of the diagonalised version of Algorithm 1 into that of Schittenhelm's. For Algorithms 3 and 4, the initialisation step of Algorithm 1 is replaced by each of initialisation steps in Algorithms 3 and 4. For a detailed description of these algorithms, see section 3.4.

We will prove that a diagonalisation of Algorithm 1 converges to an equilibrium, under the condition that the cost function is mostly separable: strictly convex. This proof is similar to that of the separable case. The convergence of the other algorithms can be proved in a similar way to this algorithm. This convergence proof is related with the global stability. In other words, if an algorithm converges to a solution from any arbitrary initial feasible point, this shows that the assignment is globally stable.

Convergence of a diagonalisation of Algorithm 1

Theorem.

Suppose,

condition (1), that $\mathcal{B}$ is non-empty, convex compact set, and the diagonalisation of Algorithm 1 generates the sequence $\{\mathbf{v}^n\}_{n\in N}$ where $\mathbf{v}^n \in \mathcal{B}$.

condition (2), that the cost $\mathbf{T}(\mathbf{v},\mu(\mathbf{v}))$ is a positive strictly monotone continuous cost function of $\mathbf{v}$. Then the diagonalisation of Algorithm 1 either terminates at an equilibrium point, $(\mathbf{v}^*,\mu^*)$ where $\mu^*=\mu(\mathbf{v}^*)$ or generates a sequence $\{\mathbf{v}^n, \mu^n\}_{n\in N}$ that has a limit point as $n \to \infty$ which is an equilibrium, $\{\mathbf{v}^*, \mu^*\}$.

Proof (It is acknowledged that Dr JD Addison proved this theorem)

Let $G(\mathbf{v}) = \max_{\mathbf{u}\in\mathcal{B}}\{\mathbf{T}(\mathbf{v},\mu(\mathbf{v}))\cdot(\mathbf{v}-\mathbf{u})\}$. We assume that the algorithm does not terminate. As the algorithm does not terminate, it follows that $G(\mathbf{v}^n) \neq 0$ for all n. Let $H = \overline{\cup_n H(W^n)} \subseteq \mathcal{B}$. H is closed, convex and compact. It is closed and convex by construction. As $\mathcal{B}$ is compact so is H. Let $\mathbf{v}^\infty$ be such that

$$G(\mathbf{v}^\infty) = \inf_{\mathbf{v}\in H} G(\mathbf{v}) = \lim_{n\to\infty} G(\mathbf{v}^n)$$

The second equality follows from the definition of H and the definition of $\mathbf{v}^n$ as the solution of the master subproblem on $H(W^n)$. Because the $\mathbf{v}^n$ is the successive solutions to the master subproblem, and from the condition (2) G(v) is a continuously differentiable strictly convex function (Hearn, 1982), so the sequence $\{G(\mathbf{v}^n)\}$ decreases monotonically. The strict convexity of G means that $\mathbf{v}^\infty$ is unique.

We show that $\lim_{n\to\infty} \mathbf{v}^n = \mathbf{v}^\infty$. Suppose that $\{\mathbf{v}^n\}$ has an accumulation point **w**, which must lie in H as H is closed. Then there is a subsequence $\mathbf{v}^{nj}$ which converges to **w**. Now by continuity, $G(\mathbf{w}) = \lim_j G(\mathbf{v}^{nj}) = G(\mathbf{v}^\infty)$ and so $\mathbf{w} = \mathbf{v}^\infty$.

It remains to prove that $\mathbf{v}^*$ lies in H where $\mathbf{v}^*$ minimises G in $\mathcal{B}$, ie

$$G(\mathbf{v}^*) = \inf_{\mathbf{v}\in\mathcal{B}} G(\mathbf{v})$$

For **v** in H, $\mathbf{T}(\mathbf{v}^\infty)\cdot(\mathbf{v}-\mathbf{v}^\infty) \geq \mathbf{0}$. To see this, suppose that there is a **u** in H such that $\mathbf{T}(\mathbf{v}^\infty)\cdot(\mathbf{u}-\mathbf{v}^\infty) < \mathbf{0}$. Then the gap function $G((1-t)\mathbf{v}^\infty + t\mathbf{u})$ at t=0 shows it is decreasing there. Since the segment $[\mathbf{v}^\infty, \mathbf{u}]$ lies in H, this contradicts the choice of $\mathbf{v}^\infty$.

Suppose that $\mathbf{v}^*$ is not in H. Since $\mathbf{v}^*$ is not in H, it cannot be $\mathbf{v}^\infty$, so it follows from convexity that there exists **u** in $\mathcal{B}$ for which $\mathbf{T}(\mathbf{v}^\infty)\cdot(\mathbf{u}-\mathbf{v}^\infty) < \mathbf{0}$. Since

$\lim_{n\to\infty} \mathbf{v}^n = \mathbf{v}^\infty$, it follows from the continuity of G that there is a n such that $\mathbf{T}(\mathbf{v}^n)\cdot(\mathbf{u}-\mathbf{v}^n) < \mathbf{T}(\mathbf{v}^n)\cdot(\mathbf{u}^n-\mathbf{v}^n) < \mathbf{0}$. This contradicts the choice of $\mathbf{u}^n$. We must have $\mathbf{v}^* \in H$ and so $\mathbf{v}^\infty = \mathbf{v}^*$. Proof is completed.

4.6 COST FUNCTION ANALYSIS FOR GOOD BEHAVIOUR

4.6.1 Introduction

The existence, uniqueness and stability of solutions are important properties when traffic assignment is analysed. Smith (1979) proved the uniqueness and stability of traffic equilibria under the sufficient condition that the Jacobian matrix of the cost function is everywhere positive definite. We can test the Jacobian matrix using existing mathematical properties: a matrix M is positive definite if and only if the determinants of all principal submatrices of the symmetric matrix $M + M^T$ are everywhere strictly positive. A principal submatrix M_{ij} is derived by deleting the corresponding ith row and jth column from the matrix.

Heydecker (1983) introduced the necessary condition that the Jacobian matrix of the cost function is everywhere a P matrix. The condition for the P matrix is that the determinants of all principal submatrices of the matrix are everywhere non-negative. This condition is necessary to prove the existence of a unique stable solution. When non-separable cost functions are analysed, the necessary condition may fail in respect of a principal submatrix representing some junctions or part of a junction. Heydecker (1983) showed how to test for such a case as described in section 4.4. The Jacobian matrix is blocked into each junction. The test of the Jacobian matrix can be done by testing each

block in turn. If any block fails to satisfy the necessary condition, then the assignment can behave badly.

Analysis of detailed modelling at priority controlled junctions shows that they produce a cost function which does not satisfy the sufficient condition: the good behaviour condition may not be satisfied by the junction delay term. We calculate the total travel time consisting of free flow travel time, and link and junction delays:

$$T_a(\mathbf{v}) = t_0 + \gamma[\, c_a(v_a) - t_0] \; + \; (1-\gamma)\, d_a(\mathbf{v})$$

where γ represents a relative weighting of the two delays.

When this travel time is used, the resulting Jacobian matrix can be expressed as below:

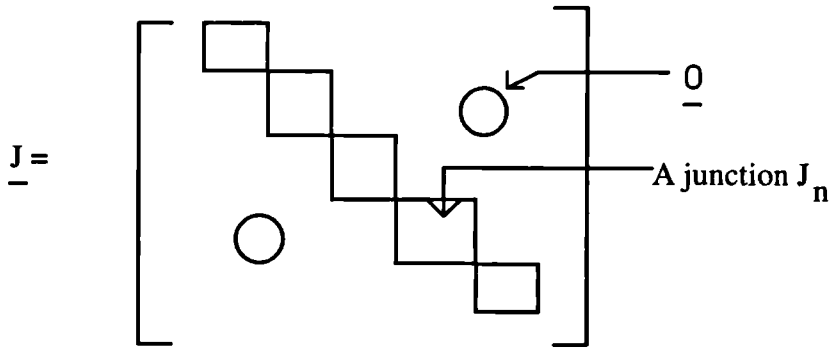
$$J_{ab} = \gamma \frac{\partial c_a}{\partial v_b} \delta_{ab} + (1-\gamma) \frac{\partial d_a}{\partial v_b}$$

$$\text{where } \delta_{ab} = \begin{cases} 1, & \text{if } a = b \\ 0, & \text{otherwise} \end{cases}$$

Note that the link travel time function, $c_a(v_a)$ contributes only to the diagonal elements of this matrix. Note that $d_a(\mathbf{v})$ is the original Pollaczek-Khinchine formula in this study of good behaviour: this is appropriate because we are concerned only with equilibrium assignments that are within capacity. Convergence behaviour of the problem depends on the relative weighting allocated to the junction and link delay terms, which are controlled by the parameter γ.

4.6.2 Jacobian matrix analysis using some examples

We will show that the Jacobian matrix of a priority controlled junction is not in general positive definite. However, apart from hooking right-turn movements at a cross-roads, it is a P-matrix. The general form of the Jacobian matrix is:



Apart from the case of hooking right-turn movements, the matrix can be rearranged by numbering the movements in order of decreasing priority so that all elements above the diagonal have value 0 thus:

$$J_n = \begin{pmatrix} +small & 0 \\ large & +medium \end{pmatrix}$$

Thus, as the determinant of all principal sub-matrices will be non-negative, and thus the matrix is P, so the necessary condition is satisfied. This means that this priority controlled junction modelling can behave well in all known example networks.

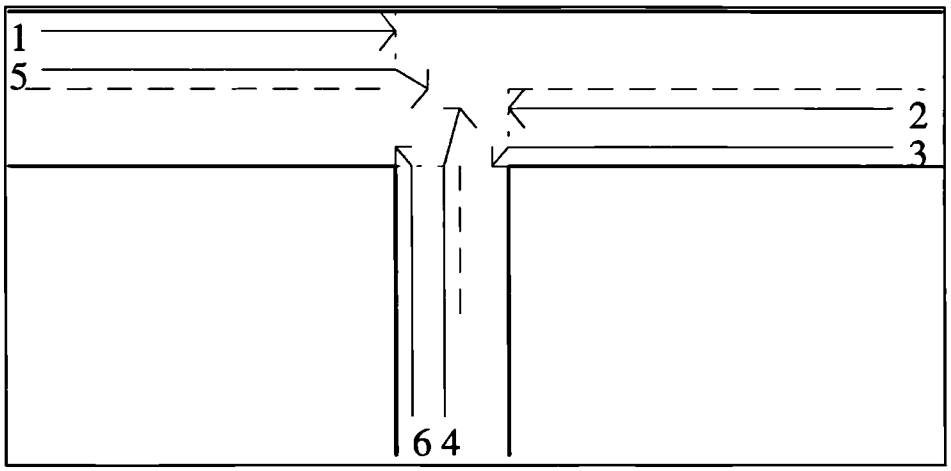


Figure 4-7 Priority junction layout

For a specific case, the Jacobian matrix of a priority controlled junction in Figure 4-7, which is clearly a P matrix, can be represented as follows:

$$\underline{J} = \begin{bmatrix} J_{11} & & & & & \\ 0 & J_{22} & & & \underline{0} & \\ 0 & J_{32} & J_{33} & & & \\ J_{41} & J_{42} & 0 & J_{44} & & \\ J_{51} & J_{52} & J_{53} & J_{54} & J_{56} & \\ 0 & J_{62} & 0 & J_{64} & 0 & J_{66} \end{bmatrix}$$

We now test the two stream case in the priority junction shown in Figure 4-1. If we assume the PICADY capacity relationship (see, Semmens, 1985, pp12-13) between link a and b as follows:

$$\mu_a = 700 - 0.189 v_b$$

The delay on link a is

$$d_a = \frac{1}{2\mu_a}\left(\frac{v_a}{\mu_a - v_a}\right)$$

Then, the Jacobian matrix can be obtained as follows:

$$J = \begin{bmatrix} \frac{\partial d_a}{\partial v_a} & \frac{\partial d_a}{\partial v_b} \\ \frac{\partial d_b}{\partial v_a} & \frac{\partial d_b}{\partial v_b} \end{bmatrix}$$

The Jacobian matrix J is positive definite if and only if $\det(J + J^T) > 0$, ie

$$4\frac{\partial d_a}{\partial v_a}\frac{\partial d_b}{\partial v_b} - \left(\frac{\partial d_a}{\partial v_b} + \frac{\partial d_b}{\partial v_a}\right)^2 > 0$$

where the derivatives are obtained as follows:

1. $\frac{\partial d_a}{\partial v_a} = \frac{1}{2}\frac{1}{(\mu_a - v_a)^2}$ 2. $\frac{\partial d_a}{\partial v_b} = \frac{\partial d_a}{\partial \mu_a}\frac{\partial \mu_a}{\partial v_b} = \frac{0.189}{2}\frac{v_a(2\mu_a - v_a)}{\mu_a^2(\mu_a - v_a)^2}$ 3. $\frac{\partial d_b}{\partial v_a} = 0$

We can use these partial derivatives in the left-hand side of the above formula as:

$$\frac{1}{(\mu_a - v_a)^2}\frac{1}{(\mu_b - v_b)^2} - \left(\frac{0.189}{2}\right)^2 \frac{X_a^2(2-X_a)^2}{(\mu_a - v_a)^4} > 0$$

where $X_a = v_a/\mu_a$.

Thus, if $(\mu_b - v_b)^2 < 100\ (\mu_a - v_a)^2$, the condition for good behaviour is satisfied so that the existence of unique and stable solution is guaranteed. In other words, too much spare capacity on the major road can lead to bad behaviour of equilibrium assignments.

4.6.3 Stability test

In this section, we will test the priority control model to see whether the theoretical conditions for good behaviour work well with respect to the experiments. For this test, firstly, we derive and analytically obtain the values and signs of the Jacobian matrix of the model. Secondly, we run the priority control model using different initial points and see whether or not the model obtains a unique stable solution. Thirdly, we compare the first and second test results to see if the theoretical conditions for good behaviour correspond well with the results of the experiments using the model.

We use the simple symmetric network shown in Figure 4-8 to test these. It has 2 origins, 2 destinations, 6 nodes and 8 links. The demand from origin 1 is entirely to destination 5 and from origin 2 to destination 6. Each of nodes 3 and 4 represents a priority controlled junction in which each of links 3 and 4 is non-priority. The capacity of each priority link is 2000 vehicles / hour. We vary the demand for each origin-destination pair, and the weighting factor γ, and the initial traffic flows in order to calculate the Jacobian value as well as to test uniqueness and stability.

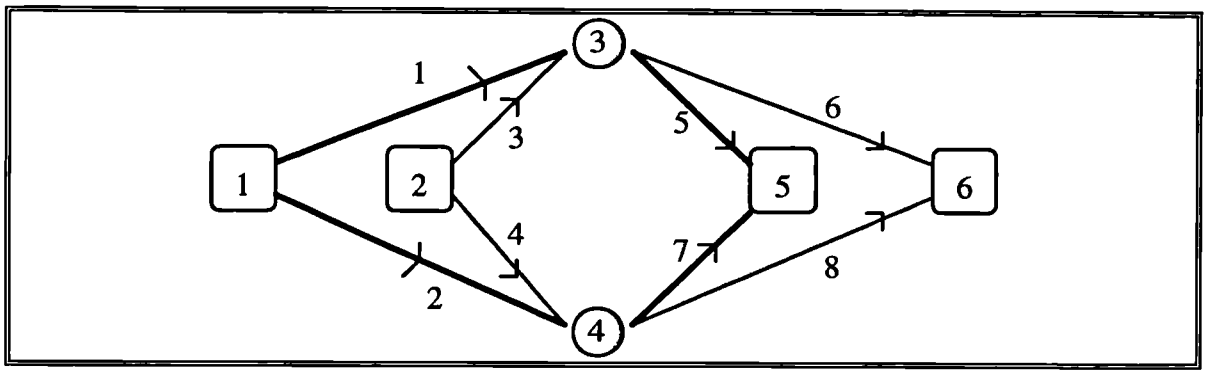


Figure 4-8 Example network

Key :	n	origin or destination n
	n	node n

In this example, an equilibrium solution of traffic flows and non-priority capacities is clearly provided by the symmetric values:

$$\mathbf{v}^* = (v^*_1, v^*_2, v^*_3, v^*_4) = 1/2(T_{15}, T_{15}, T_{26}, T_{26})$$

$$\mu^*(\mathbf{v}^*) = (\mu^*_3(\mathbf{v}^*), \mu^*_4(\mathbf{v}^*))$$

Tables 4-1 and 4-2 show the symmetric equilibrium traffic flows and capacities on a non-priority link respectively calculated at various levels of demand T_{15} and T_{26}. The lower-left parts of the tables are empty because the values of the upper-right parts are symmetric to those of the lower-left parts due to the symmetric example network. The symbol -, in the lower-right parts indicates that the region is overloaded so that steady-state values cannot be calculated appropriately here because we are using only the original Pollaczek-Khinchine formula in good behaviour study.

O/D	T_{26}					
	v^*_1/v^*_3	200	800	1400	2000	2600
T_{15}	200	100/100	100/400	100/700	100/1000	100/1300
	800		400/400	400/700	400/1000	400/1300
	1400			700/700	700/1000	-
	2000				-	-
	2600				-	-

Table 4-1 The symmetric equilibrium traffic flows for calculating Jacobian matrix

O/D	T_{26}					
	μ^*_3	200	800	1400	2000	2600
T_{15}	200	791	791	791	791	791
	800		724	724	724	724
	1400			658	658	-
	2000				-	-
	2600				-	-

Table 4-2 The symmetric equilibrium capacity on non-priority link 3 for calculating the Jacobian matrix

The values in Tables 4-1 and Table 4-2 are used to calculate the Jacobian matrix in Table 4-3 which gives the signature of the Jacobian matrix in terms of the sign of the determinant of its symmetric part, $\det(J+J^t)$ and its diagonal elements J_{11} and J_{22} as the value of the parameter γ is changed from 0.0 to 0.75 by an increment of 0.25. In this table, when γ=0 corresponding to the maximum junction effect, the value of $\det(J+J^t)$ is negative, whilst in all other cases, the values of $\det(J+J^t)$ and its elements J_{11} and J_{22} are all positive. The values of the determinants of the Jacobian increase as the weighting factor, γ, approaches 1.0.

O/D	T_{26}					
	γ=0.0 0.25 0.5 0.75	200	800	1400	2000	2600
T_{15}	200	-/+/+ +/+/+ +/+/+ +/+/+	-/+/+ +/+/+ +/+/+ +/+/+	-/+/+ +/+/+ +/+/+ +/+/+	-/+/+ +/+/+ +/+/+ +/+/+	-/+/+ +/+/+ +/+/+ +/+/+
	800		-/+/+ +/+/+ +/+/+ +/+/+	-/+/+ +/+/+ +/+/+ +/+/+	-/+/+ +/+/+ +/+/+ +/+/+	-/+/+ +/+/+ +/+/+ +/+/+
	1400			-/+/+ +/+/+ +/+/+ +/+/+	-/+/+ +/+/+ +/+/+ +/+/+	-

Table 4-3 The signs of the Jacobian
(Key: the signs of each symbol indicate respectively the signs of det $(J+J^t)$ / J_{11}/ J_{22} ; and the signature - in the bottom right corner indicates overloaded region)

We define global stability as:

Definition: global stability: An equilibrium of the combined priority control and traffic assignment model is *globally stable* if any initial flow in the feasible region leads to that equilibrium traffic flows and the associated minor road capacities by the dynamic adjustment process.

Note that in this study of stability, minor road capacities were calculated according to the PICADY formula and the assignment was calculated by a diagonalisation of the Frank-Wolfe algorithm. Note also that if an equilibrium is globally stable, then it is also locally stable because global stability implies local stability.

In order to test global stability, we use three extreme assignment cases as initial starting values. In all these tests, the value $\gamma = 0$, which gives the maximum junction effect in the cost function, is used.

Test 1. When initial flows are given as: $(v_1, v_2, v_3, v_4)=(T_{15}, 0, 0, T_{26})$

We obtain the calculated traffic flows and minor capacities on the links by a diagonalisation of the Frank-Wolfe algorithm. These results are shown respectively in Tables 4-4 and 4-5 using initial traffic flow $\mathbf{v}^0=(T_{15}, 0, 0, T_{26})$. We compare the resulting calculated equilibrium traffic flows with those symmetric values in Table 4-1 to test the global stability of that equilibrium. To determine the stability or otherwise, we followed the definition of the global stability given above. The conclusions drawn from this global stability test are in Table 4-6. Table 4-7 represents the number of iterations used by the diagonalisation of the Frank-Wolfe algorithm to terminate and the gap function value in equation (2.15) at that termination.

O/D	T_{26}					
	v^n_1/v^n_3	200	800	1400	2000	2600
T_{15}	200	100/100	100/400	100/700	100/1000	100/1300
	800		400/400	400/700	400/1000	400/1300
	1400			700/700	700/1000	-
	2000				-	-
	2600				-	-

Table 4-4 The calculated traffic flows by F-W in global stability test 1
(Key: v^n is traffic flow on link after iteration n which is shown in Table 4-7)

O/D	T_{26}					
	μ^n_3	200	800	1400	2000	2600
T_{15}	200	791	791	791	791	791
	800		724	724	724	724
	1400			658	658	-
	2000				-	-
	2600				-	-

Table 4-5 The results of calculated capacity on non-priority link 3 by F-W in global stability test 1.
(Key: μ^n is capacity on non-priority link after iteration n which is shown in Table 4-7)

O/D	T_{26}					
		200	800	1400	2000	2600
T_{15}	200	S	S	S	S	S
	800		S	S	S	S
	1400			S	S	-
	2000				-	-
	2600					-

Table 4-6 The result of global stability test 1
(Key: S represents stability that symmetric equilibrium is obtained in the test)

O/D	T_{26}					
	n/gap	200	800	1400	2000	2600
T_{15}	200	15/0.0	5/0.0	10/0.0	50/21.0	9/0.0
	800		8/0.0	50/0.47	10/0.0	16/0.0
	1400			50/2.5	50/2.0	-
	2000				-	-
	2600				-	-

Table 4-7 The number of iteration used by F-W in global stability test 1
(Key: n represents the number of iteration used; and gap represents the gap value)

Test 2. When initial flows are given as: $(v_1, v_2, v_3, v_4)=(T_{15}/2, T_{15}/2, T_{26}/2, T_{26}/2)$

We obtain the calculated traffic flows and minor capacities on link by a diagonalisation of the Frank-Wolfe algorithm. These results are shown respectively the same as in Tables 4-4 and 4-5 using initial traffic flows $\mathbf{v}^0=(T_{15}/2, T_{15}/2, T_{26}/2, T_{26}/2)$. We compare the resulting calculated traffic flows in Table 4-4 with those symmetric values in Table 4-1 to test the global stability of that equilibrium. The results of global stability test are the same as in Table 4-6. Table 4-8 represents the number of iterations used by the diagonalisation of the Frank-Wolfe algorithm to terminate and the gap function value in equation (2.15) at that termination.

O/D	T_{26}					
	n/gap	200	800	1400	2000	2600
T_{15}	200	12/0.0	9/0.0	24/0.0	50/21.0	6/0.0
	800		50/0.03	50/0.72	50/2.0	16/0.0
	1400			50/14.0	50/4.0	-
	2000				-	-
	2600				-	-

Table 4-8 The number of iteration used by F-W in global stability test 2
(Key: n represents the number of iteration used; and gap represents the gap value)

Test 3. When initial flows are given as: $(v_1, v_2, v_3, v_4)=(T_{15}, 0, T_{26}, 0)$

We obtain the calculated traffic flow and non-priority capacities on link by a diagonalisation of the Frank-Wolfe algorithm shown respectively the same as in Tables 4-4 and 4-5 using initial traffic flow $\mathbf{v}^0=(T_{15}, 0, T_{26}, 0)$. We compare the resulting calculated traffic flows in Table 4-4 with those symmetric values in Table 4-1 to test the global stability of that equilibrium. To determine the stability or otherwise, we followed definition of global stability given above. The results of the global stability test are the

same as in Table 4-6. Table 4-9 represents the iteration number used by the diagonisation of the Frank-Wolfe algorithm to terminate and the gap function value in equation (2.15) at that termination.

O/D	T_{26}					
	n/gap	200	800	1400	2000	2600
T_{15}	200	37/0.0	5/0.0	34/0.0	50/11.0	39/0.0
	800		11/0.0	30/0.0	26/0.0	25/0.0
	1400			50/11.8	50/28.0	-
	2000				-	-
	2600				-	-

Table 4-9 The number of iteration used by F-W in global stability test 3
(Key: n represents the number of iteration used; and gap represents the gap value)

These global stability tests show that the combined priority control and traffic assignment model appears to have a unique solution which is independent on the initial values. Thus this problem appears to be globally stable and in particular the symmetric equilibrium is obtained by all three global stability tests, and these solutions are associated with a P matrix of the Jacobian analysis. Global stability indicates that local stability is also satisfied in traffic assignment with priority controlled modelling.

4.7 NUMERICAL EXAMPLES

4.7.1 Introduction

In this section, the solution methods developed in chapter 3 under the diagonalisation procedure described in chapter 2 are applied to two example networks each of which includes several priority controlled junctions. The results of this help us to compare the performance of these various algorithms when used in networks with this kind of the non-separable cost function. As the first example, the network and travel demand devised by Charlesworth (1977) are used: this has 5 origins, 5 destinations, 46

nodes and 78 links (for the figures and data of the network, see Appendix 2). The Sioux Falls network (see Figure 3-6) is used as the second example: this has 12 origins, 12 destinations, 23 nodes and 72 links. In each case, capacities on minor links are obtained using the PICADY model. The values used for the parameters G_a, K_a, H_a, Y_a, e_a are those determined empirically by Semmens (1985, pp12-13) in accordance with the structures of the networks. As stopping criteria in each algorithm, either a required level of the gap value as $G(v^n) \leq \varepsilon$, or a maximum number of iterations are used.

Algorithms 1, 2, 3 and 4, the Frank-Wolfe and the Schittenhelm algorithms have been tested to compare their performance. In particular, Algorithm 2 used one iteration of the master subproblem of Schittenhelm's before switching to the master subproblem of Algorithm 1. Algorithms 3 and 4 also used one column generation in the initialisation step. In these tests, the weighting factor γ, which is used to combine junction and link delays, is varied from 0.0 to 1.0 in increments of 0.25. In Tables 4-10, 11, 12, 13 and 14, the final values of each algorithm are compared in terms of the iteration number, CPU time on SUN Sparc station (model 370 GX) and the mean gap value from equation (2.16) as the weighting factor γ is varied on Charlesworth's network. In the same way, Tables 4-15, 16, 17, 18 and 19 show the final values for each algorithm for the Sioux Falls network. The performance of each algorithm as a whole is shown in Figures 4- 9, 10, 11, 12 and 13 as the weighting factor γ is varied on Charlesworth's network. In the same way, the performance of each algorithm is shown in Figures 4-14, 15, 16, 17 and 18 on the Sioux Falls network.

4.7.2 Charlesworth's network

In Table 4-10, the performance of the algorithms on Charlesworth's network is shown when the weighting factor is γ =0.0 which means that junction delay but not link

delay is considered in the choice of routes. Algorithm 3 runs fastest ending with a mean gap value of 12.768 seconds in 14.82 CPU seconds of SUN Sparc station 370 GX. However, Algorithm 1 runs furthest ending with a mean gap value of 0.386 seconds in 84.08 CPU seconds. Note that each algorithm is stopped by the maximum iteration number. Figure 4-9 shows the whole process of convergence for each algorithm. The pattern for each algorithm except the Frank-Wolfe includes fluctuations in gap. In fact, the Frank-Wolfe algorithm improves the mean gap value by about 0.061 seconds with each iteration. However, the value is too small to see in this large scale figure. Figure 4-19 shows the final traffic flows obtained by each algorithm compared with those of Algorithm 3, and Figure 4-20 shows the final minor road capacities for this case. These figures show close agreement between in flows and capacities on most links calculated by each of the algorithms except the Frank-Wolfe. In this case, all of the algorithms performed reasonably well except for the non-convergence of the Frank-Wolfe algorithm, as indicated by the substantial final gap value.

In Table 4-11, the performance of the algorithms is shown when the weighting factor γ =0.25, which includes some allowance for link delays. Algorithm 3 again runs fastest terminating with a small mean gap of 2.698 seconds in 14.75 CPU seconds. In this case, the Schittenhelm algorithm runs slightly further to a mean gap value of 2.633 seconds in just 15.01 CPU seconds. Note that each of the algorithms is stopped by the maximum iteration number. Figure 4-10 shows the whole process of the convergence for each algorithm. The pattern of each algorithm except the Frank-Wolfe algorithm includes fluctuations in gap. Note that Algorithms 1 and 2 stabilise after 30-40 CPU seconds. Figure 4-21 shows the final traffic flows obtained by each algorithm compared with those of Algorithm 3, and Figure 4-22 shows the final minor capacities for this case. As in the case γ=0, these figures show close agreement between in flows and capacities estimated by algorithms except the Frank-Wolfe.

In Table 4-12, the performance of the algorithms is shown when the weighting factor γ =0.5. This case corresponds to equal weight being given to link and junction delays. Algorithm 3 again runs fastest to a mean gap value of 7.117 seconds in 14.73 CPU seconds. However, Algorithm 2 runs furthest to a mean gap value of 0.971 seconds in 82.37 CPU seconds. Figure 4-11 shows the whole process of the convergence for each algorithm. Note that Algorithm 1 stabilises after 20 seconds of CPU, and Algorithm 2 also stabilises for a while and then fluctuates. Figure 4-23 shows the final traffic flows obtained by each algorithm compared with those of Algorithm 3, and Figure 4-24 shows the final minor road capacities for this case. These figures show substantial differences between the new algorithms and the Frank-Wolfe algorithm in flows and capacities on some links and close agreement on others. In particular, there appear to be three groups of flows: F-W, Algorithm 4 and the others. In the case of the F-W, this can be attributed to non-convergence, as indicated by the substantial final gap value. On the other hand, the final gap values for the other algorithms are similar, and it is not clear why Algorithm 4 should have distinct flows.

In Table 4-13, the performance of algorithms on the Charlesworth network is shown when the weighting factor γ =0.75 which corresponds to 1/3 weighting on junction delay. Algorithm 3 again runs fastest to a mean gap value of 7.297 seconds in 14.75 CPU seconds. However, Algorithm 2 runs furthest to a mean gap value of 0.551 seconds in 82.47 CPU seconds. In Figure 4-12, the pattern of each algorithm except the Frank-Wolfe algorithm includes fluctuations in gap. Note that the cycle of fluctuations in Algorithm 4 is larger than previous weighting values, and Algorithm 2 reduces the gap value after 40 seconds of CPU. Figure 4-25 shows the final traffic flows obtained by each algorithm compared with those of Algorithm 3, and Figure 4-26 shows the final minor road capacities for this case. In these figures, there are more close agreements

between in flows and capacities by each algorithm than earlier cases.

In Table 4-14, the performance of the algorithms is shown when the weighting factor γ =1.0 which corresponds to the separable case in which only link delay is considered. In this case, Algorithm 2 converges fastest to a mean gap value of 0.000 seconds in 11.80 CPU seconds. Algorithms 1, 4 and the Schittenhelm algorithm also converge to a mean gap of zero whilst Algorithm 3 terminates after 31 iterations with a mean gap close to zero. Note that Algorithms 1, 2 and 4, and Schittenhelm algorithm terminate by the required level of the gap value. In Figure 4-13, each algorithm except the Frank-Wolfe algorithm converges to a mean gap value of zero, though this cannot be shown explicitly on a logarithmic scale. Figure 4-27 shows the final traffic flows obtained by each algorithm compared with those of Algorithm 3, and Figure 4-28 shows the final minor road capacities for this case. These figures show that apart from F-W the final values differ slightly because in each case the gap value converges to near zero and, due to separability, the equilibrium is unique.

In Figure 4-39, final mean gap values of each algorithm are depicted for the Charlesworth network as the weighting factor γ is changed. An expectation of the effect of γ on the final gap value is to reduce the final gap as the weighting factor becomes 1.0 which means only the link based cost. In particular, Algorithm 2 reduces the final gap value steadily as the weighting factor increases to 1. However, due to non-monotonic patterns in gap value, the final gap values in the other algorithms have been increased when the weighting factors are 0.5 and 0.75. In terms of performance of the algorithms, Algorithm 3 runs quickly in all the cases as does the Schittenhelm algorithm. Algorithms 1, 2 and 4 run with a further convergent value of gap. Note that the Schittenhelm algorithm works well for $\gamma = 0$ and 0.25 and fairly well at other values.

4.7.3 Sioux Falls network

In Table 4-15, the performance of the algorithms on the Sioux Falls network is shown when the weighting factor γ =0.0. Algorithm 3 runs fastest to a mean gap value of 0.932 seconds in just 3.74 CPU seconds. However, Algorithm 1 runs furthest to a slightly lower mean gap value of 0.914 seconds in 39.61 CPU seconds. Figure 4-14 shows the whole process of the convergence for each algorithm. The pattern of each algorithm except the Frank-Wolfe algorithm includes fluctuations in gap in the first stage and slight changes in the later stages. Figure 4-29 shows the final traffic flows obtained by each algorithm compared with those of Algorithm 3, and Figure 4-30 shows the final minor road capacities for this case. These figures show substantial differences between flows and capacities on some links and close agreement on others calculated by each algorithms.

In Table 4-16, the performance of the algorithms is shown when γ =0.25. Algorithm 3 again runs fastest to a mean gap value of 1.387 seconds in 3.61 CPU seconds. However, Algorithm 1 terminates with the smallest mean gap value of 0.108 seconds in 39.90 CPU seconds. Figure 4-15 shows the whole process of the convergence for each algorithm. The pattern of each algorithm is broadly similar to that in the previous case. Note that in Algorithm 2 the gap value increases in the final stages. Figure 4-31 shows the final traffic flows obtained by each algorithm compared with those of Algorithm 3, and Figure 4-32 shows the final minor road capacities for this case. Note that the Frank-Wolfe, Algorithms 1, 3 and 4 have distinct different flow groups, and minor road capacities range mostly from 500 to 800.

In Table 4-17, the performance of the algorithms is shown when the weighting factor γ =0.5. Algorithm 3 again runs fastest to a mean gap value of 0.199 seconds in

3.74 CPU seconds. However, Algorithm 2 runs furthest to a mean gap value of 0.069 seconds in 39.68 CPU seconds. Figure 4-16 shows the whole process of the convergence for each algorithm. The pattern of each algorithm is broadly similar to that in the previous cases. Note that Algorithm 2 has stabilised more than in the previous case. Figure 4-33 shows the final traffic flows obtained by each algorithm compared with those of Algorithm 3, and Figure 4-34 shows the final minor capacities for this case. Note that by comparison with previous cases, the minor road capacities have more cluster values.

In Table 4-18, the performance of the algorithms is shown when γ =0.75. Algorithm 3 again runs fastest to a mean gap value of 0.265 seconds in 3.45 CPU seconds. However, Algorithm 2 runs furthest to a mean gap value of 0.137 seconds in 38.40 CPU seconds. Figure 4-17 shows the whole process of the convergence for each algorithm. The pattern of each algorithm except the Frank-Wolfe algorithm includes fluctuations of gap in the first stage and slight changes in the later stage. Note that Algorithm 1 reduces the gap value dramatically in the initial stage whilst it goes up afterwards. Figure 4-35 shows the final traffic flows obtained by each algorithm compared with those of Algorithm 3, and Figure 4-36 shows the final minor road capacities for this case. Note that traffic flows and minor road capacities are closer to the diagonal line than in the cases with the smaller weighting values.

In Table 4-19, the performance of the algorithms on the Sioux Falls network is shown when the weighting factor γ =1.0 which means that only link delay is considered in the choice of routes. In this case, Algorithm 3 converges fastest to a mean gap value of 0.000 seconds in 3.26 CPU seconds, and all the algorithm except F-W also converge to a mean gap value of 0.000 seconds. Figure 4-18 shows the whole process of the convergence for each algorithm. The pattern of each algorithm except the Frank-Wolfe

algorithm includes fluctuations in the first stage and slight changes in the later stage. Figure 4-37 shows the final traffic flows obtained by each algorithm compared with those of Algorithm 3. Figure 4-38 shows the final minor road capacities for this case. These figures show that the final values differ slightly as a gap value converges to zero, and as the equilibrium is unique due to separability of the cost function.

In Figure 4-40, final gap function values of each algorithm are depicted as the weighting factor γ changes on the Sioux Falls network. An expectation of effects of γ on a final gap value is to reduce the final gap as the weighting factor becomes 1.0 which means only link cost. In particular, all but Algorithms 2 and 3, and the Schittenhelm algorithm reduce the final gap value steadily as the weighting factor increases to 1. In terms of performance of the algorithms, Algorithm 3 runs quickly all cases. Algorithms 1, 2 and 4 run with a further convergent value of gap.

4.7.4 Discussion and summary

The numerical tests show that the new algorithms run quickly. Furthermore, they terminate with substantially smaller gap values than those achieved by the Frank-Wolfe algorithm. However, they have non-monotonic patterns of the gap value in their iterations. These non-monotonic fluctuations are cyclic and arise during the solution of the master subproblem. Although we have to treat these non-monotonic patterns with care, the final gap values are close to zero, which means that the solution is near an equilibrium. For additional information on the convergence to an equilibrium, the estimated link flows and minor road capacities of the new algorithms are all similar to each other but differ substantially from those of the Frank-Wolfe algorithm. On the other hand, the Frank-Wolfe algorithm converges only slowly but steadily decreases an objective value at each iteration. The Frank-Wolfe algorithm requires much more

computational time to achieve any given level of convergence than do the new algorithms. Furthermore, the new algorithms reach a convergent level in few iterations as shown in the Figures from 4-9 to 4-18. These results correspond well to the convergence behaviour explained in Chapter 3: the new algorithms were developed to overcome the slow convergence of the Frank-Wolfe algorithm.

In summary of these test results, Algorithm 3 performs best in all the cases in terms of CPU time. However, Algorithms 1 and 2 run furthest in terms of gap value in most cases. In terms of both measures of CPU time and gap value, Algorithms 2 and 3, and the Schittenhelm algorithm perform well in most cases. The convergence pattern of each algorithm includes fluctuations. In particular, the pattern fluctuates mostly as the weighting value approaches zero. When the weighting value is unity, which corresponds to the case of only link cost, the gap value converges to zero. The pattern of convergence on the Charlesworth network fluctuates more than on the Sioux Falls. This can be explained according to the characteristics and structure of each network. Charlesworth's network has two main roads that provide alternate routes: choice between them can then cause gap values to fluctuate. On the other hand, the Sioux Falls network has a grid form so that many route choices are possible and the demand is spread more widely and the fluctuation is less.

	Iterations	CPU(Second)	Mean gap (Second)
Algorithm 1	31	84.08	0.386
Algorithm 2	31	81.17	5.063
Algorithm 3	31	14.82	12.768
Algorithm 4	31	81.40	3.289
Frank-Wolfe	100	53.46	115770.500
Schittenhelm	31	15.03	0.819

Table 4-10 Performance of algorithms in Charlesworth's network when γ =0.0 (Maximum junction effect)

	Iterations	CPU(Second)	Mean gap (Second)
Algorithm 1	31	82.38	4.094
Algorithm 2	31	82.65	5.102
Algorithm 3	31	14.75	2.698
Algorithm 4	31	80.40	3.433
Frank-Wolfe	100	53.42	87268.500
Schittenhelm	31	15.01	2.633

Table 4-11 Performance of algorithms in Charlesworth's network when γ =0.25

	Iterations	CPU(Second)	Mean gap (Second)
Algorithm 1	31	81.96	2.229
Algorithm 2	31	82.37	0.971
Algorithm 3	31	14.73	7.117
Algorithm 4	31	80.55	5.797
Frank-Wolfe	100	53.37	29102.200
Schittenhelm	31	14.99	6.502

Table 4-12 Performance of algorithms in Charlesworth's network when γ =0.5

	Iterations	CPU(Second)	Mean gap (Second)
Algorithm 1	31	82.43	3.908
Algorithm 2	31	82.47	0.551
Algorithm 3	31	14.75	7.297
Algorithm 4	31	81.92	3.407
Frank-Wolfe	100	53.38	58185.300
Schittenhelm	31	14.96	6.993

Table 4-13 Performance of algorithms in Charlesworth's network when $\gamma = 0.75$

	Iterations	CPU(Second)	Mean gap (Second)
Algorithm 1	14	13.56	0.000
Algorithm 2	13	11.80	0.000
Algorithm 3	31	14.57	0.000
Algorithm 4	14	13.00	0.000
Frank-Wolfe	100	52.72	2.320
Schittenhelm	27	12.62	0.000

Table 4-14 Performance of algorithms in Charlesworth's network when $\gamma = 1.0$ (No junction effect)

	Iterations	CPU(Second)	Mean gap (Second)
Algorithm 1	31	39.61	0.914
Algorithm 2	31	39.57	1.440
Algorithm 3	31	3.74	0.932
Algorithm 4	31	40.48	2.156
Frank-Wolfe	100	14.51	8956.096
Schittenhelm	31	3.92	1.023

Table 4-15 Performance of algorithms in Sioux Falls network when $\gamma = 0.0$ (Maximum junction effect)

	Iterations	CPU(Second)	Mean gap (Second)
Algorithm 1	31	39.90	0.108
Algorithm 2	31	42.01	1.962
Algorithm 3	31	3.61	1.387
Algorithm 4	31	42.33	0.212
Frank-Wolfe	100	14.35	4957.153
Schittenhelm	31	3.71	1.285

Table 4-16 Performance of algorithms in Sioux Falls network when γ =0.25

	Iterations	CPU(Second)	Mean gap (Second)
Algorithm 1	31	40.06	0.139
Algorithm 2	31	39.68	0.069
Algorithm 3	31	3.61	0.199
Algorithm 4	31	39.87	0.098
Frank-Wolfe	100	14.19	1096.495
Schittenhelm	31	3.79	0.193

Table 4-17 Performance of algorithms in Sioux Falls network when γ =0.5

	Iterations	CPU(Second)	Mean gap (Second)
Algorithm 1	31	37.33	0.163
Algorithm 2	31	38.40	0.137
Algorithm 3	31	3.45	0.265
Algorithm 4	31	39.68	0.255
Frank-Wolfe	100	14.41	601.049
Schittenhelm	31	3.57	0.265

Table 4-18 Performance of algorithms in Sioux Falls network when γ =0.75

	Iterations	CPU(Second)	Mean gap (Second)
Algorithm 1	27	24.28	0.000
Algorithm 2	13	4.15	0.000
Algorithm 3	31	3.26	0.000
Algorithm 4	31	34.80	0.000
Frank-Wolfe	100	14.02	0.012
Schittenhelm	31	3.35	0.000

Table 4-19 Performance of algorithms in Sioux Falls network when $\gamma = 1.0$ (No junction effect)

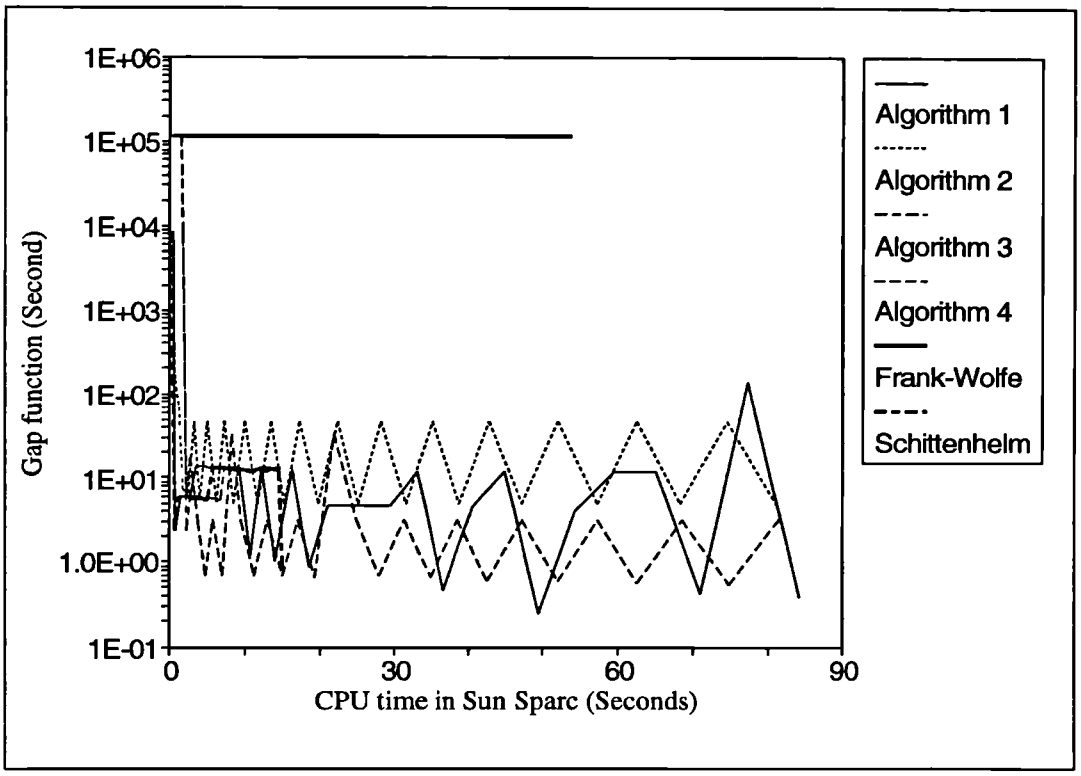


Figure 4-9 The performance of algorithms in Charlesworth's network when γ =0.0 (Maximum junction effect)

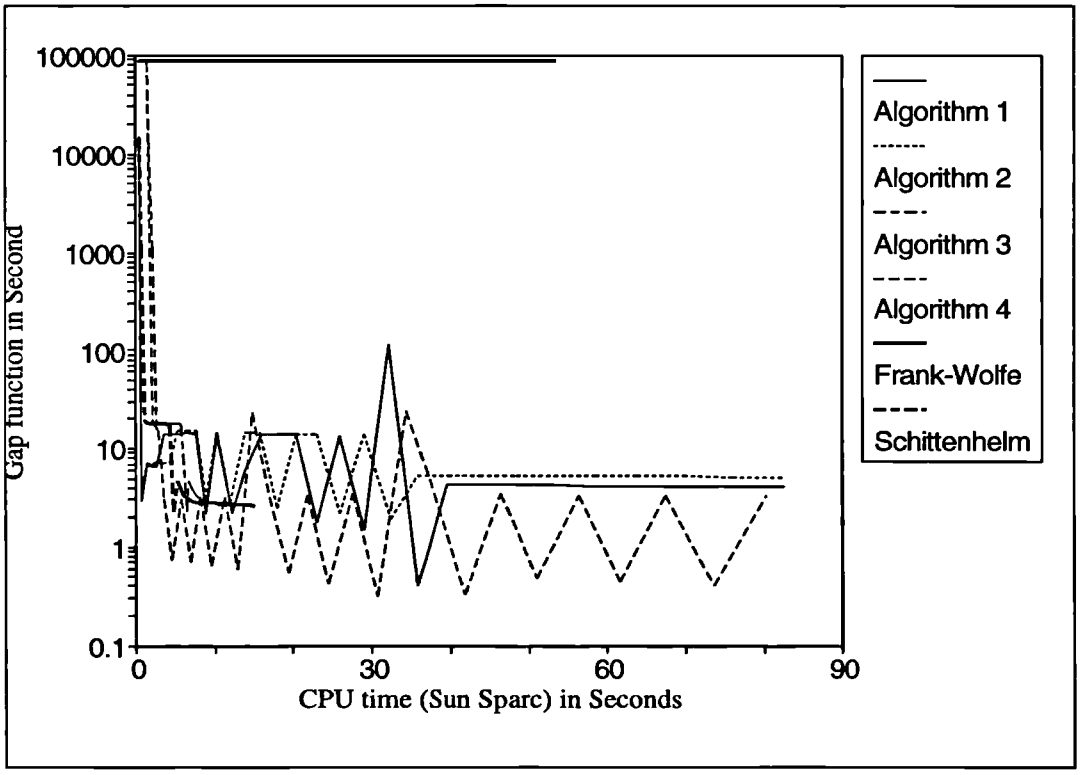


Figure 4-10 The performance of algorithms in Charlesworth's network when γ =0.25

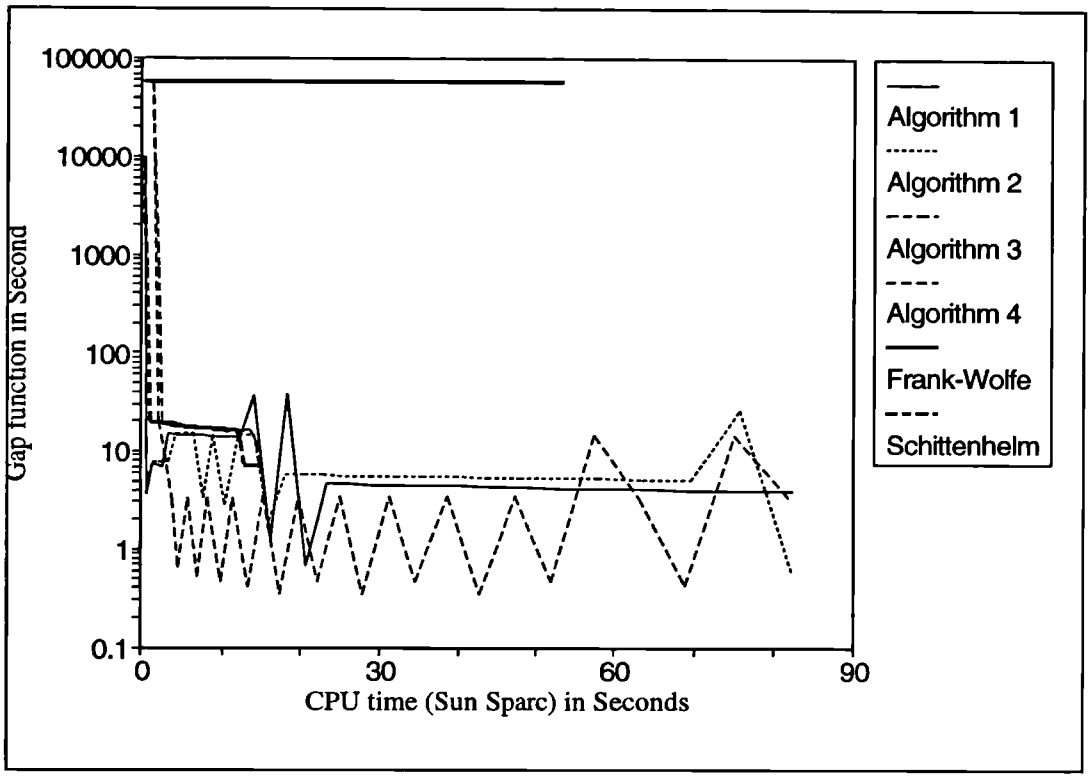


Figure 4-11 The performance of algorithms in Charlesworth's network when γ =0.5

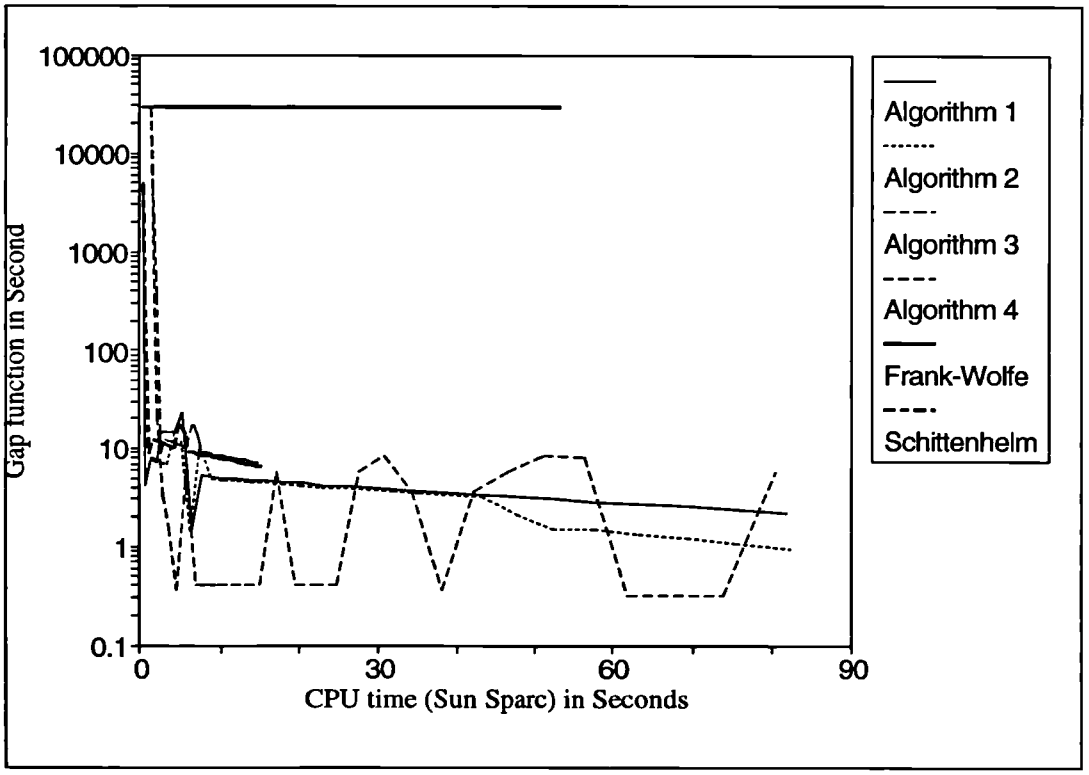


Figure 4-12 The performance of algorithms in Charlesworth's network when γ =0.75

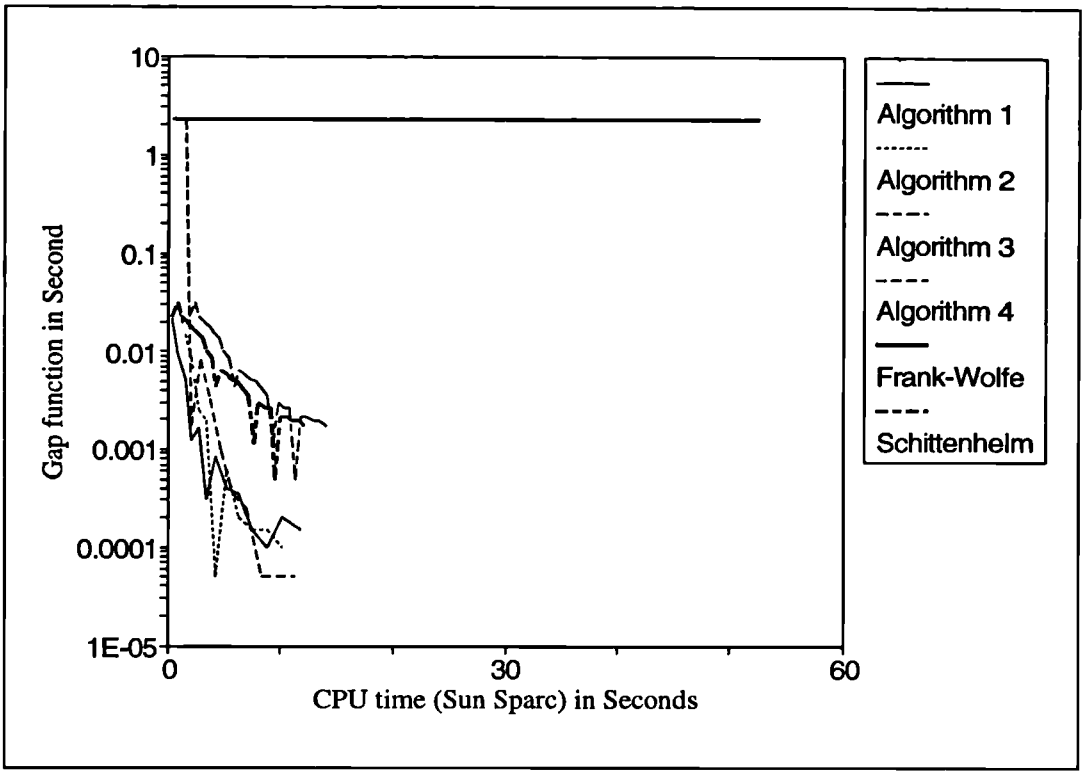


Figure 4-13 The performance of algorithms in Charlesworth's network when $\gamma = 1.0$ (No junction effect)

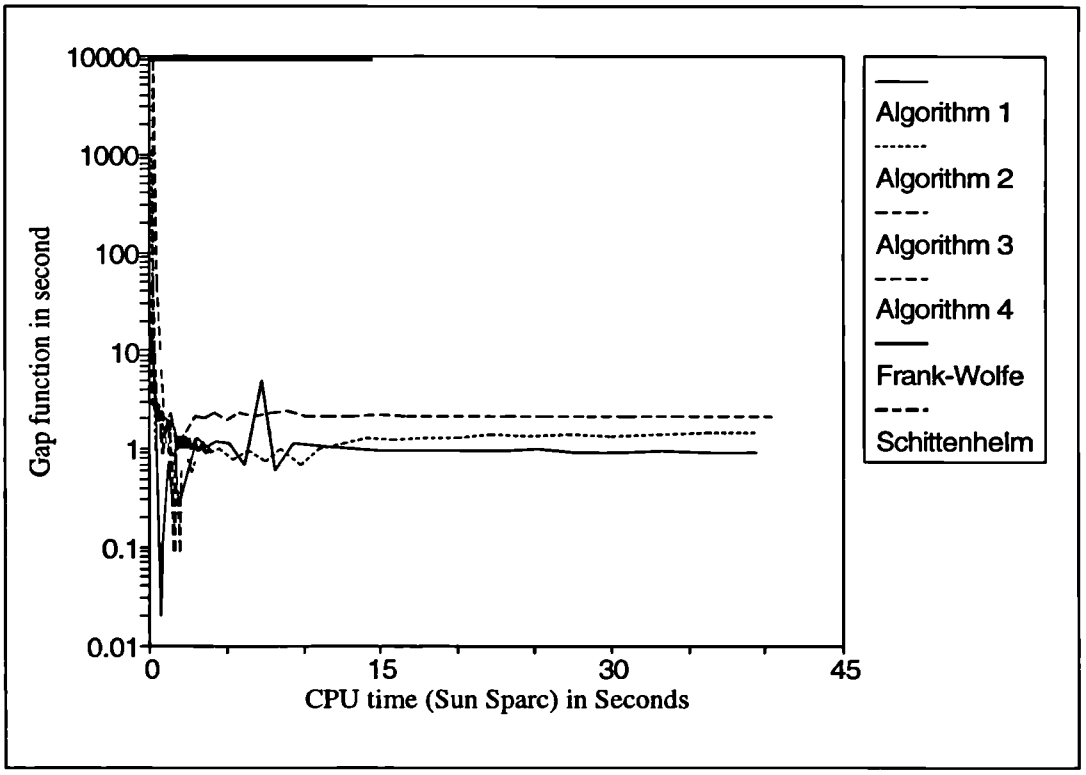


Figure 4-14 The performance of algorithms in Sioux Falls network when $\gamma = 0.0$ (Maximum junction effect)

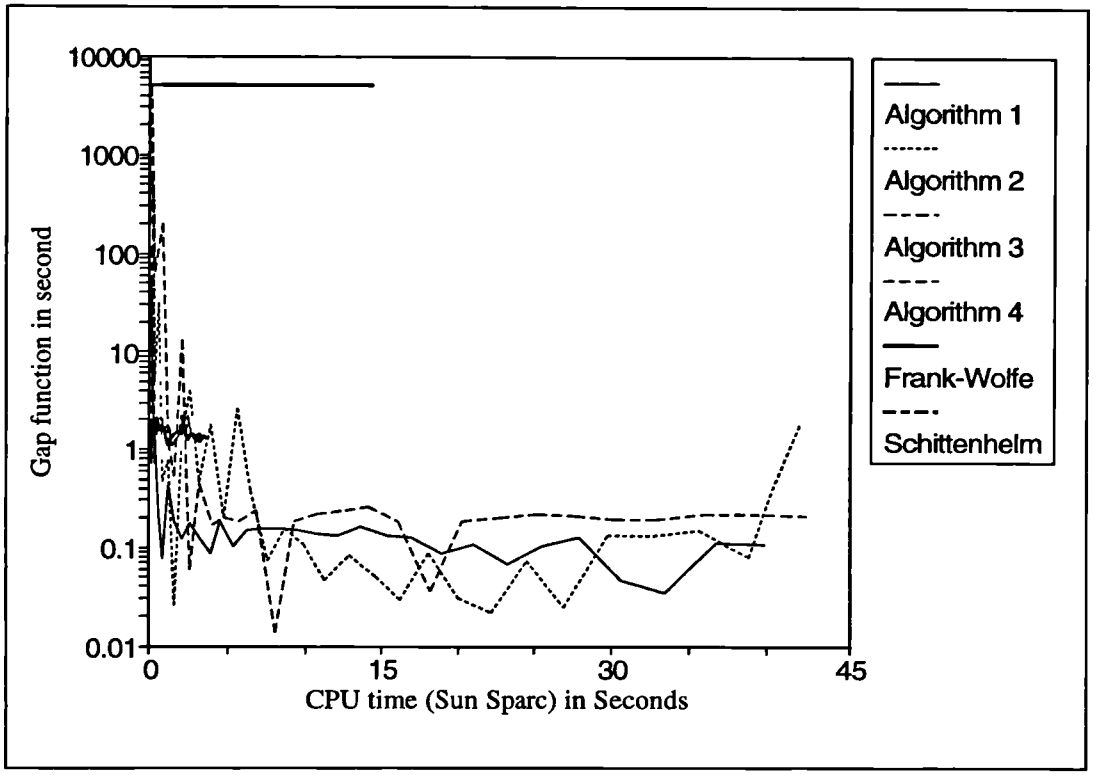


Figure 4-15 The performance of algorithms in Sioux Falls network when γ =0.25

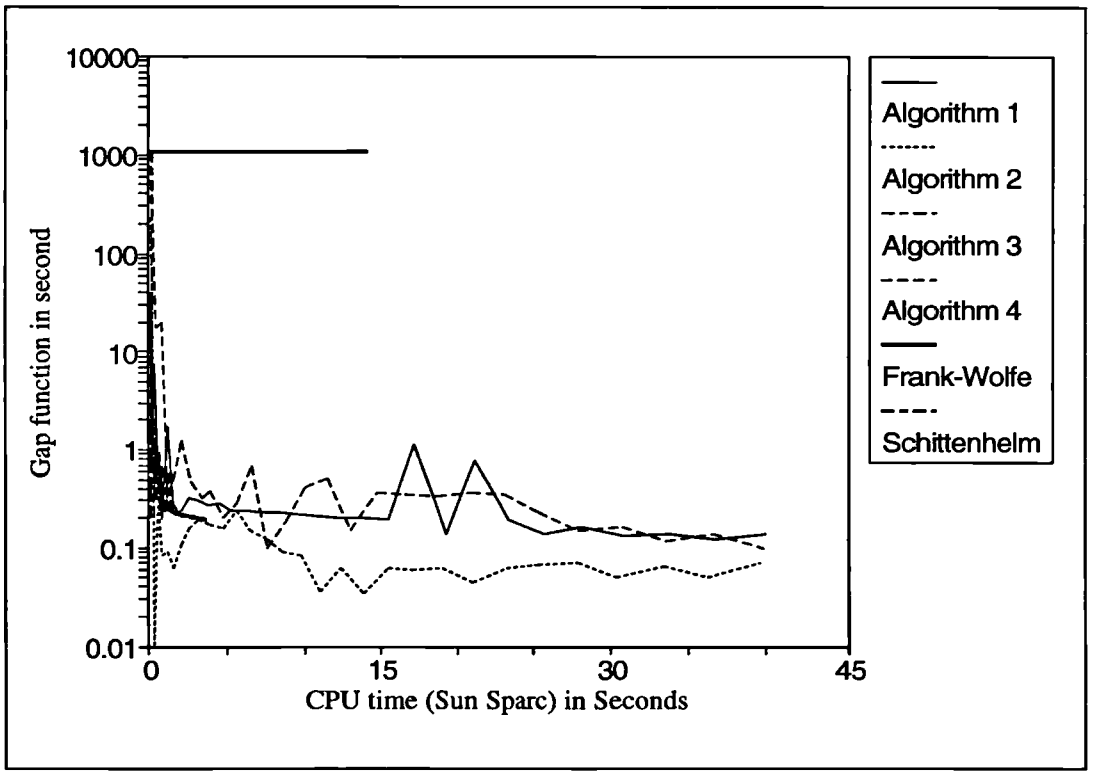


Figure 4-16 The performance of algorithms in Sioux Falls network when γ =0.5

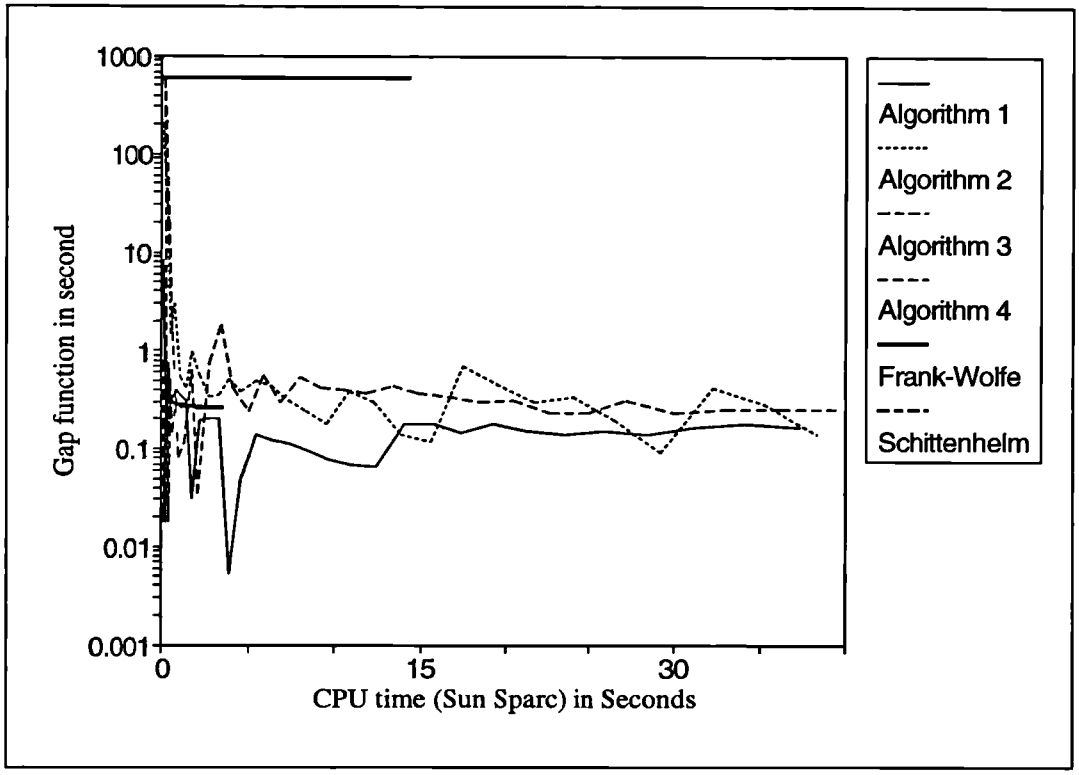


Figure 4-17 The performance of algorithms in Sioux Falls network when γ =0.75

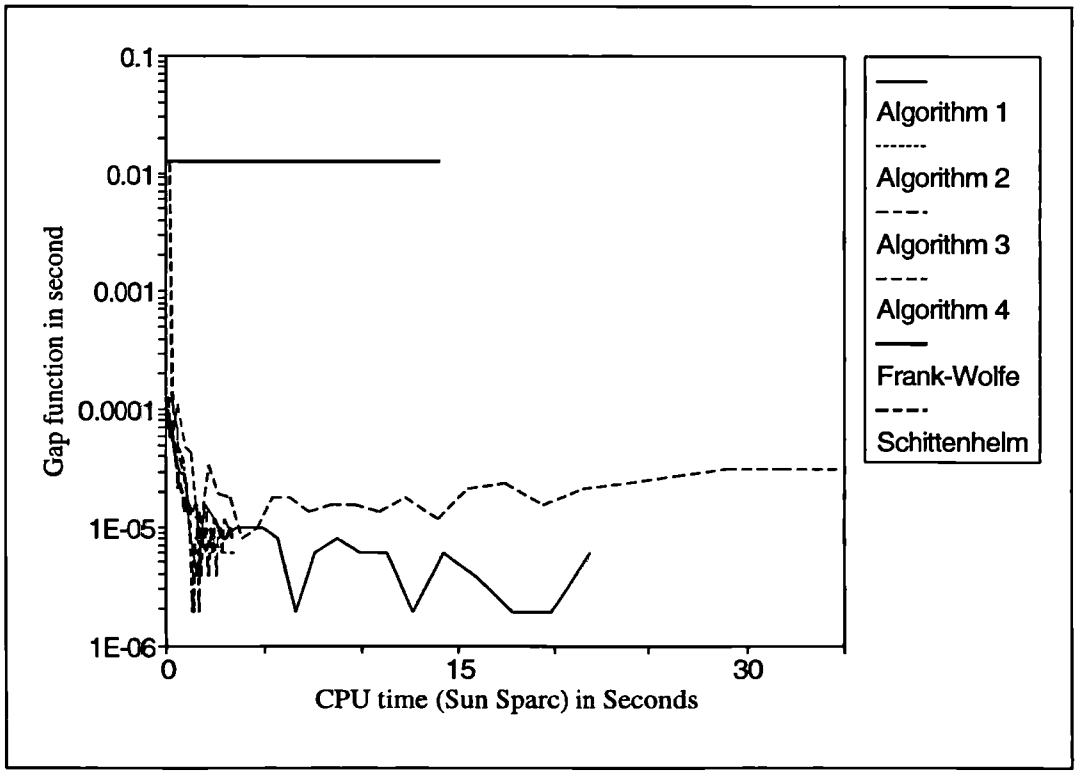


Figure 4-18 The performance of algorithms in Sioux Falls network when γ =1.0 (No junction effect)

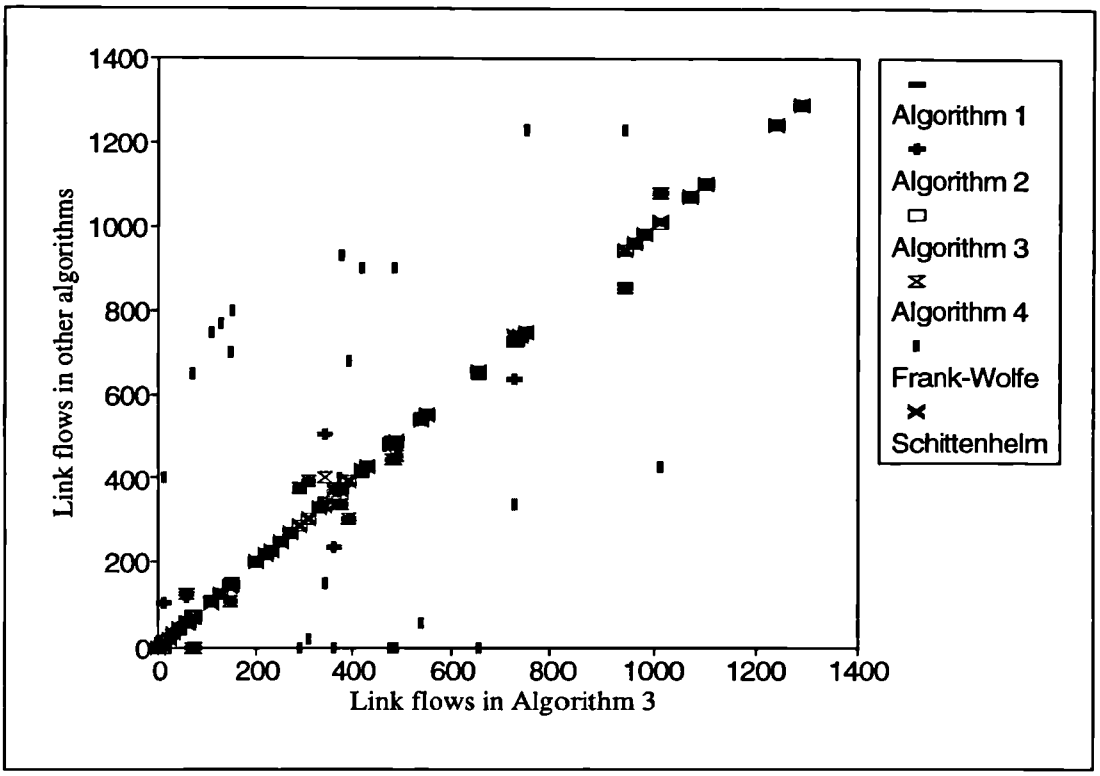


Figure 4-19 The comparison of traffic flows obtained by algorithms on Charlesworth's network when γ =0.0 (Maximum junction effect)

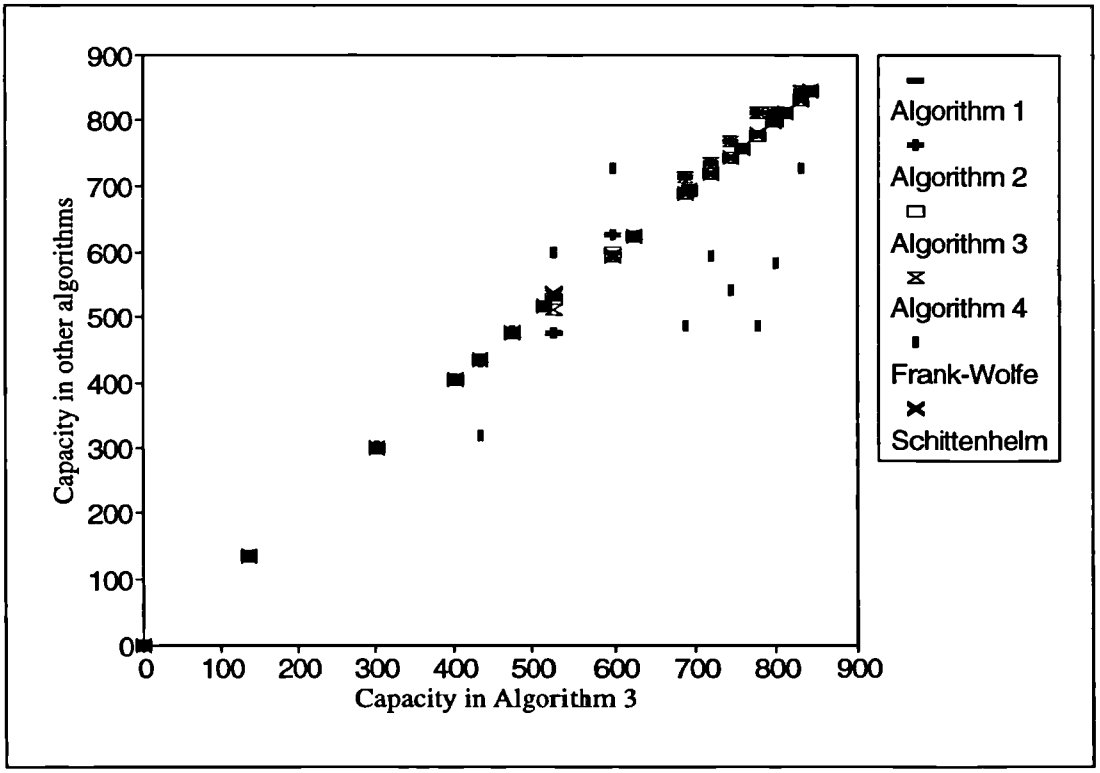


Figure 4-20 The comparison of minor capacities obtained by algorithms on Charlesworth's network when γ =0.0 (Maximum junction effect)

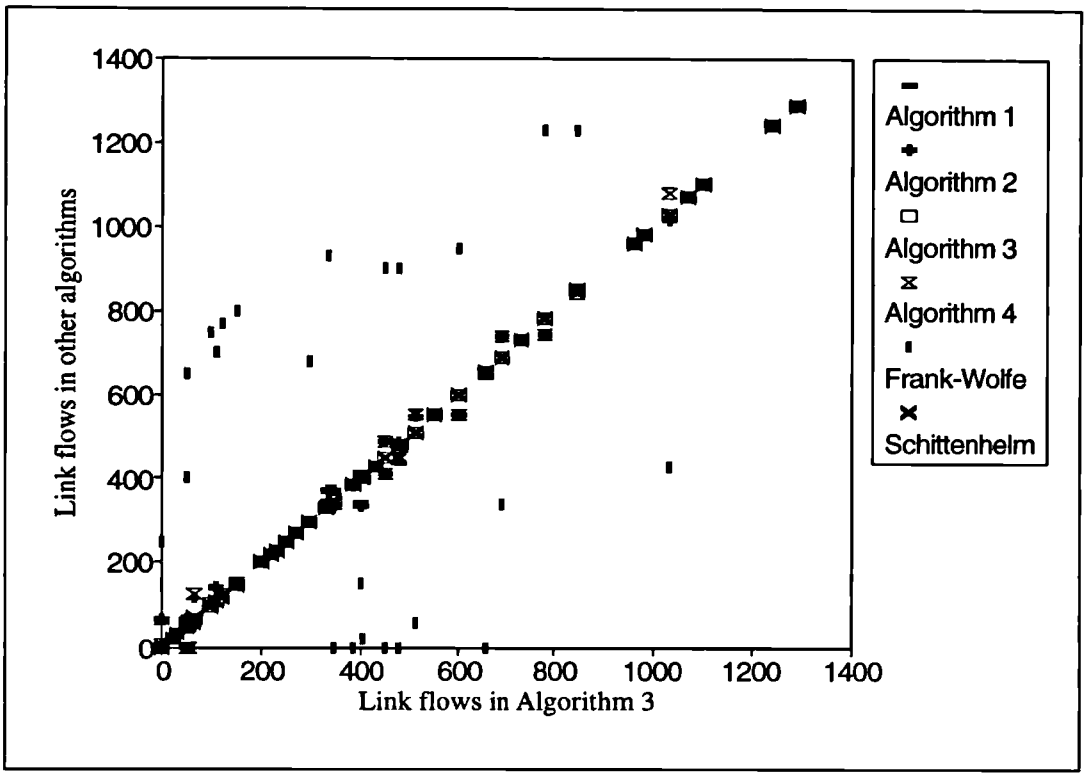


Figure 4-21 The comparison of traffic flows obtained by algorithms on Charlesworth's network when γ =0.25

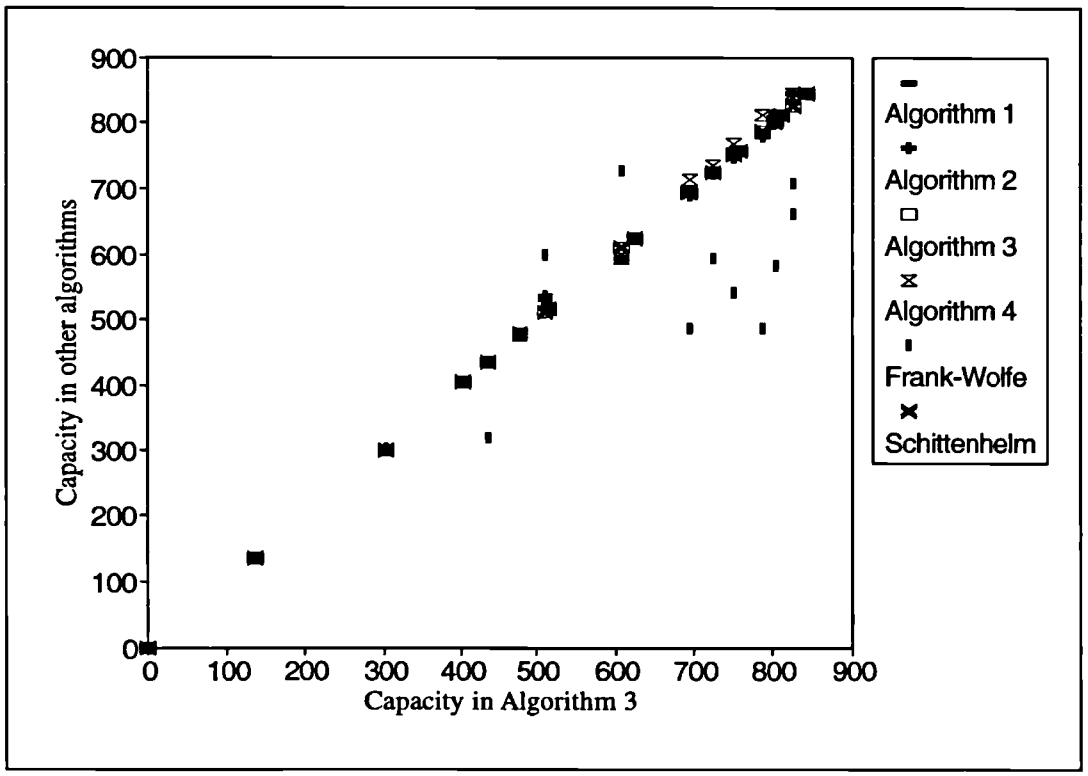


Figure 4-22 The comparison of minor capacities obtained by algorithms on Charlesworth's network when γ =0.25

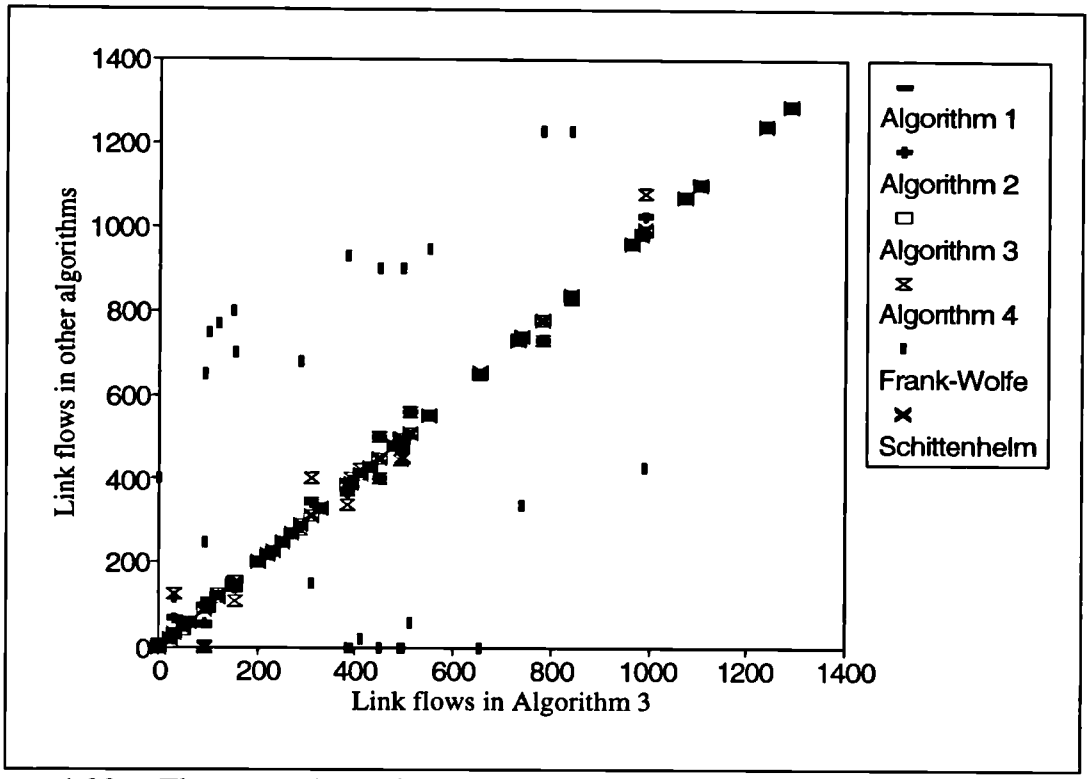


Figure 4-23 The comparison of traffic flows obtained by algorithms on Charlesworth's network when γ =0.5

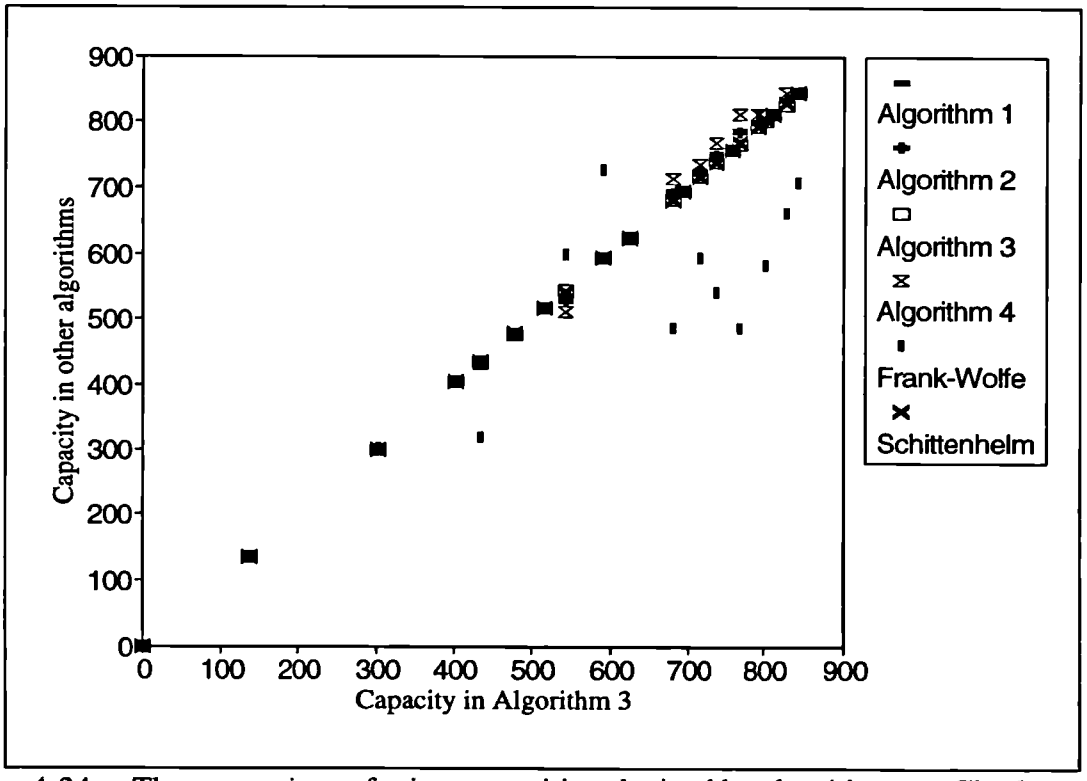


Figure 4-24 The comparison of minor capacities obtained by algorithms on Charlesworth's network when γ =0.5

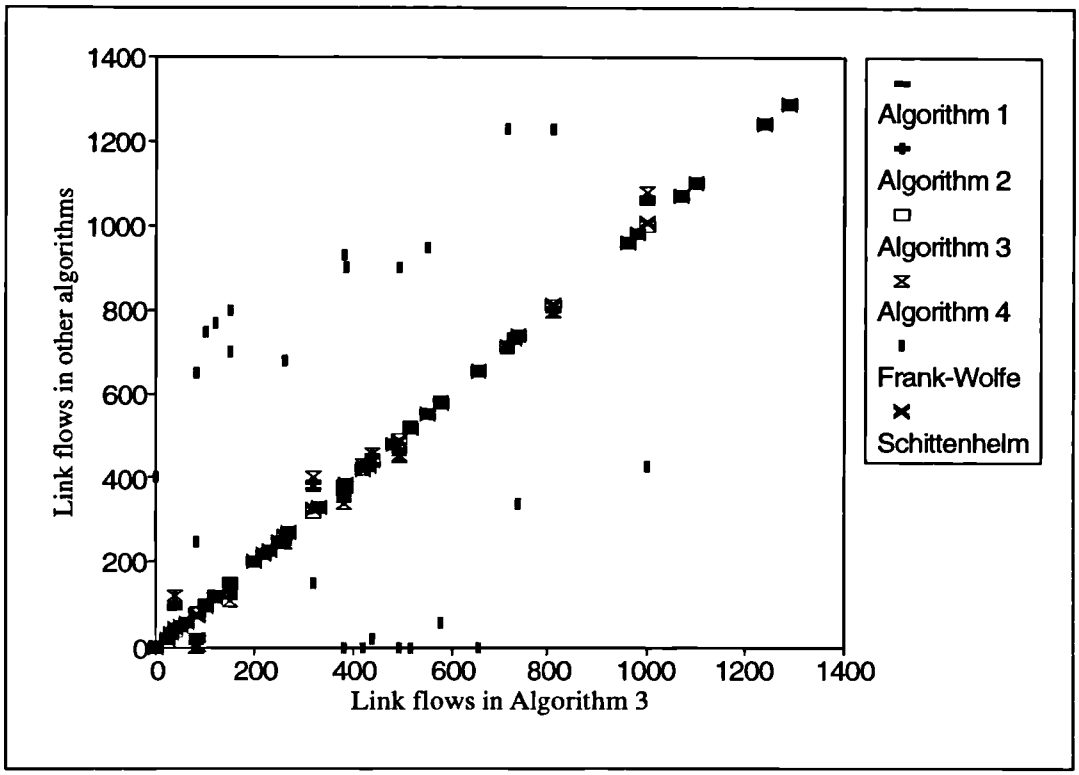


Figure 4-25 The comparison of traffic flows obtained by algorithms on Charlesworth's network when γ =0.75

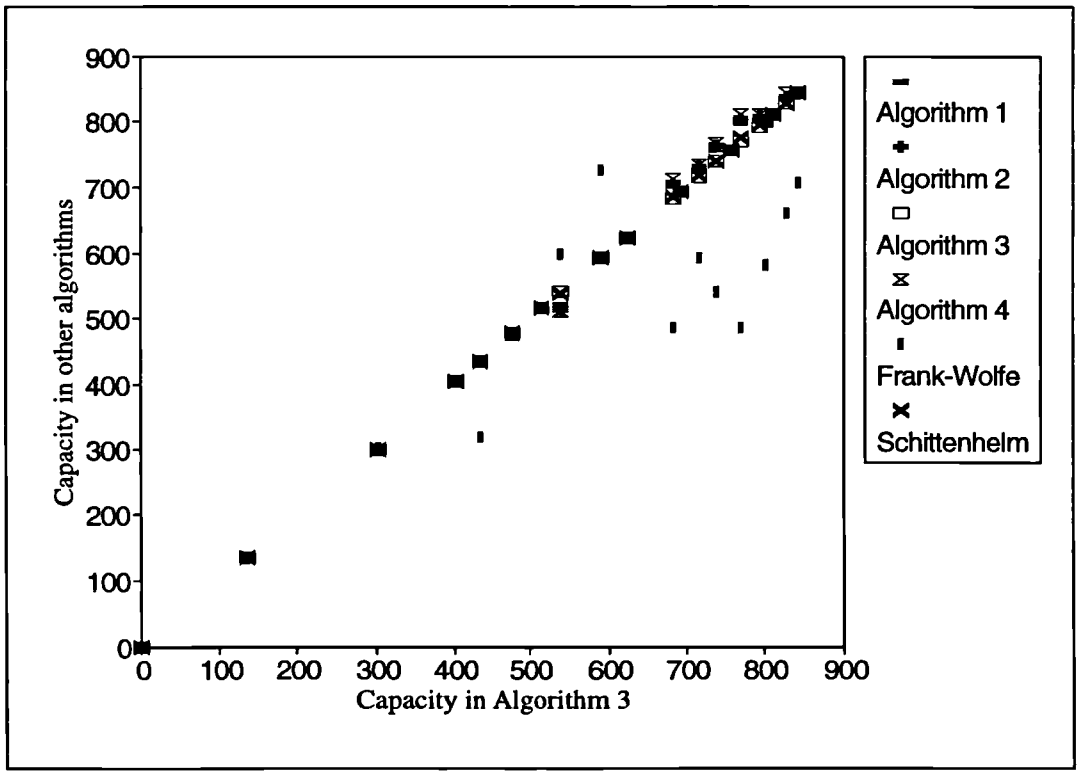


Figure 4-26 The comparison of minor capacities obtained by algorithms on Charlesworth's network when γ =0.75

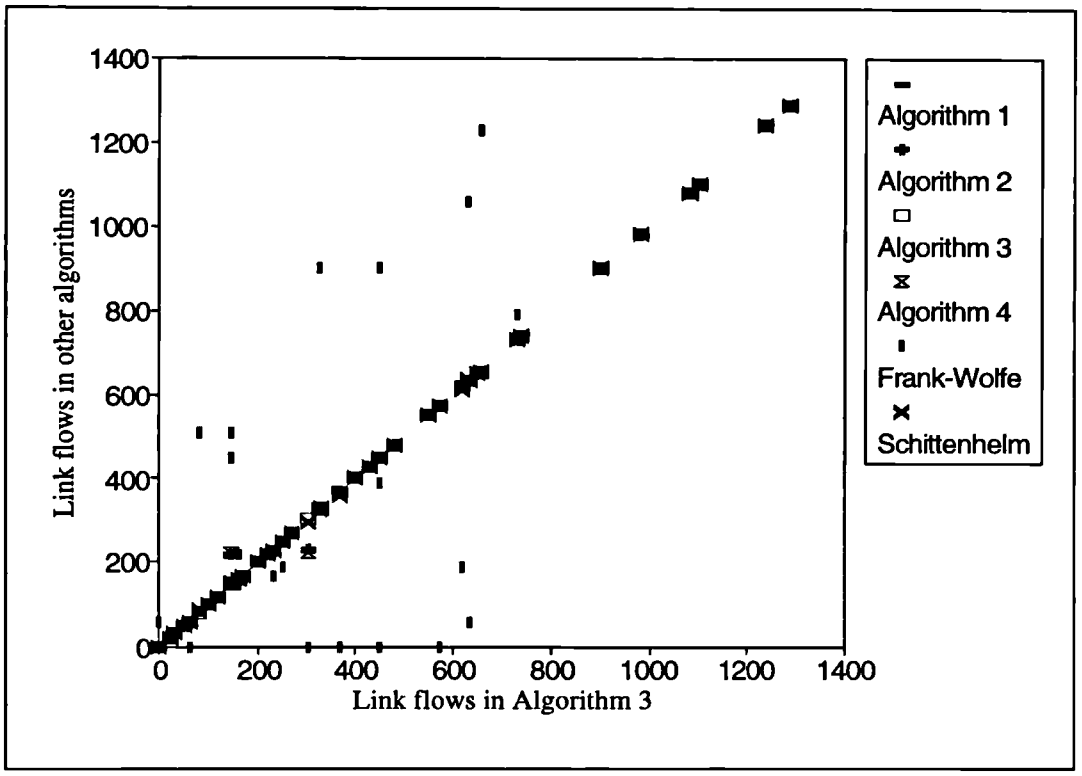


Figure 4-27 The comparison of traffic flows obtained by algorithms on Charlesworth's network when $\gamma = 1.0$ (No junction effect)

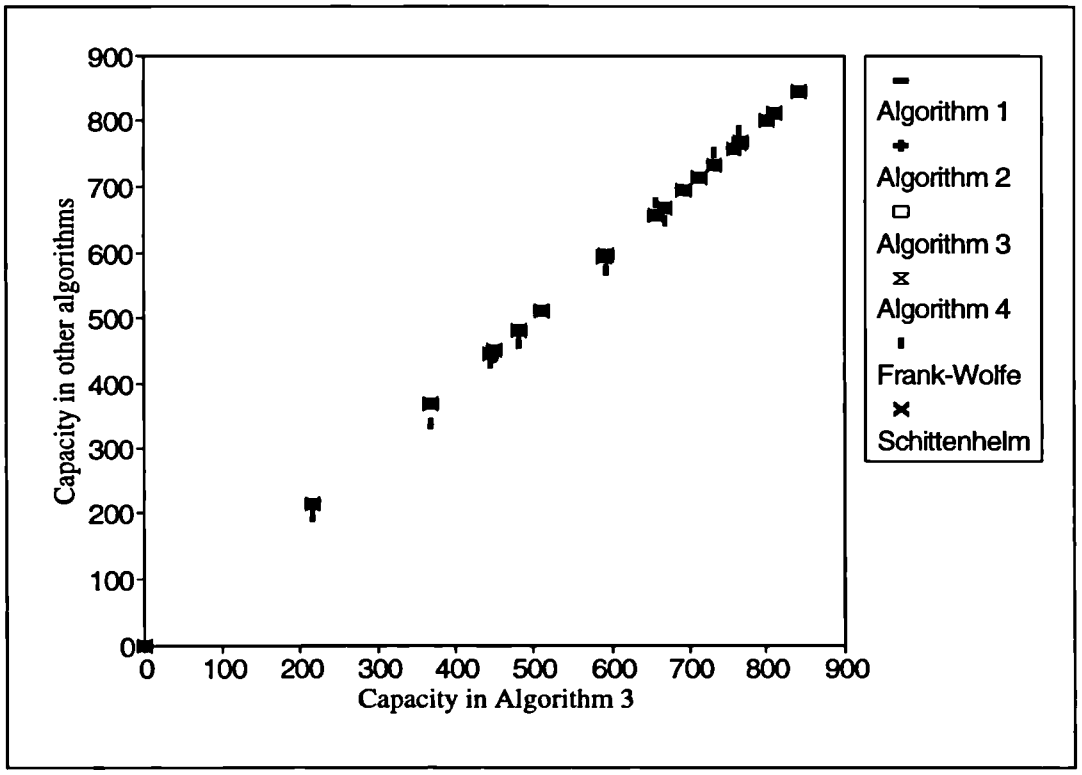


Figure 4-28 The comparison of minor capacities obtained by algorithms on Charlesworth's network when $\gamma = 1.0$ (No junction effect)

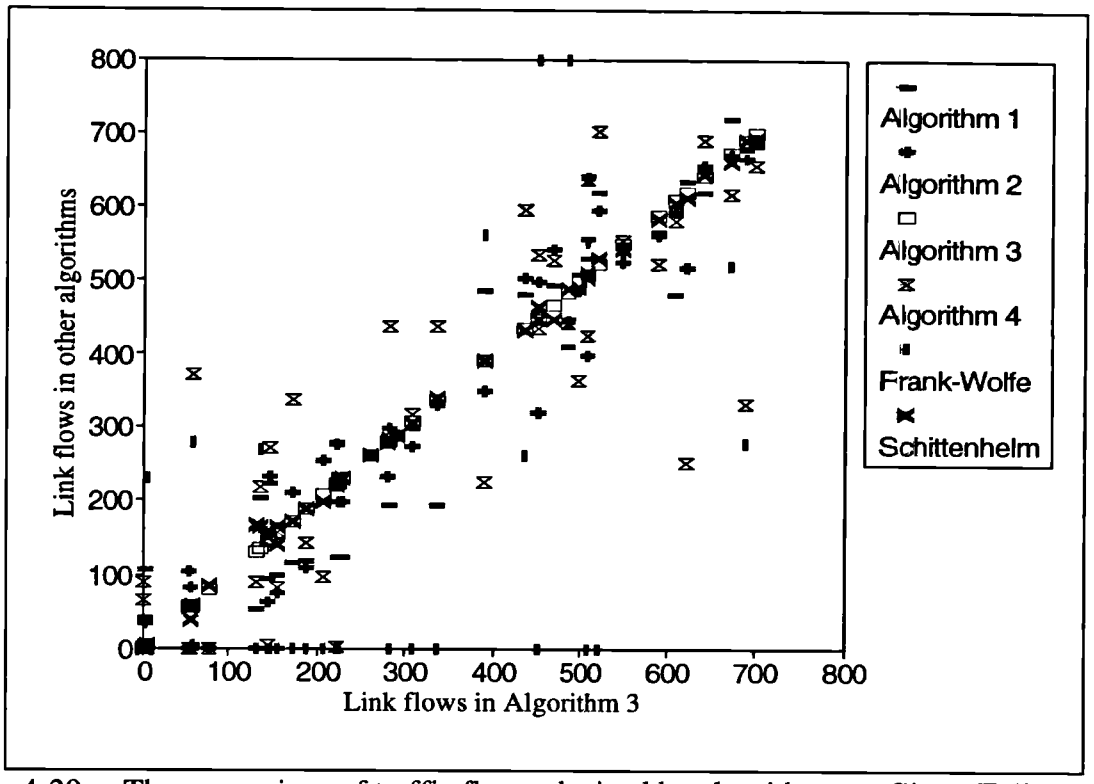


Figure 4-29 The comparison of traffic flows obtained by algorithms on Sioux Falls network when γ =0.0 (Maximum junction effect)

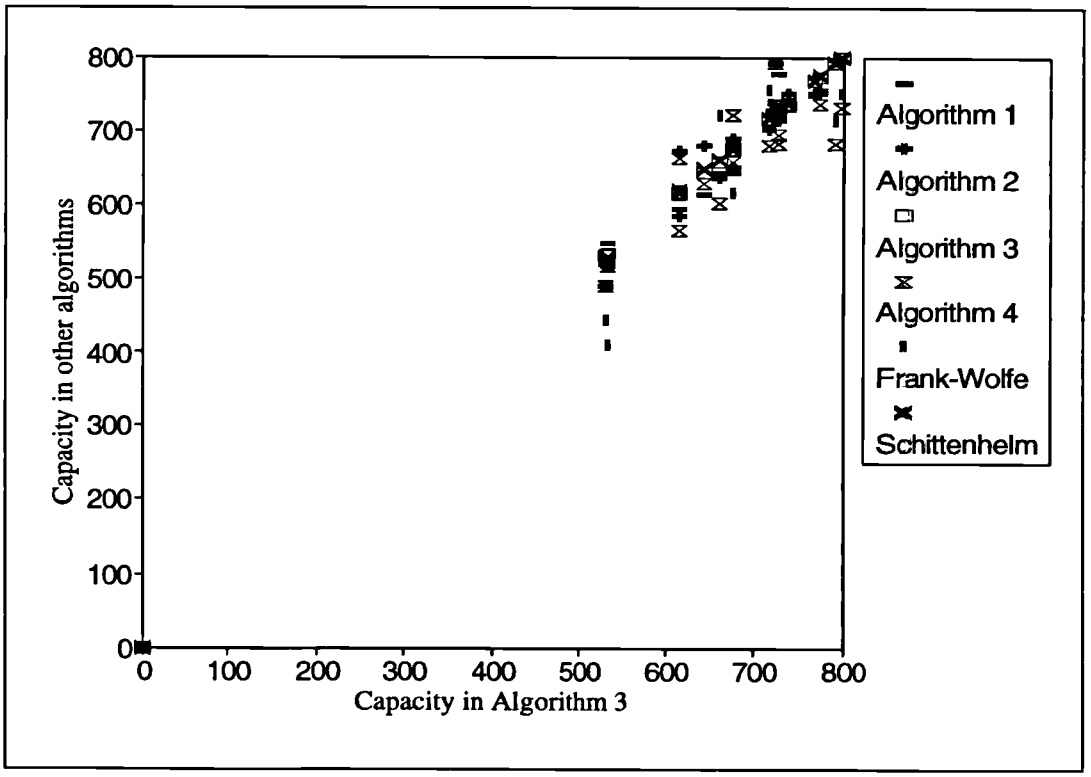


Figure 4-30 The comparison of minor capacities obtained by algorithms on Sioux Falls network when γ =0.0 (Maximum junction effect)

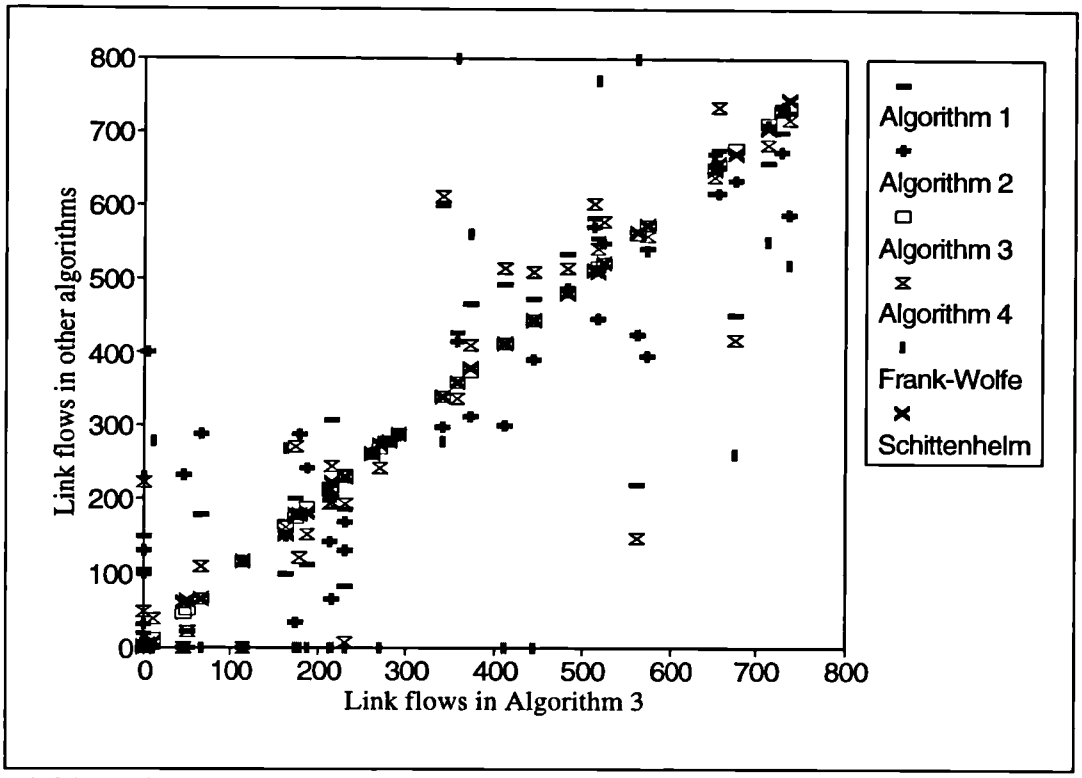


Figure 4-31 The comparison of traffic flows obtained by algorithms on Sioux Falls network when γ =0.25

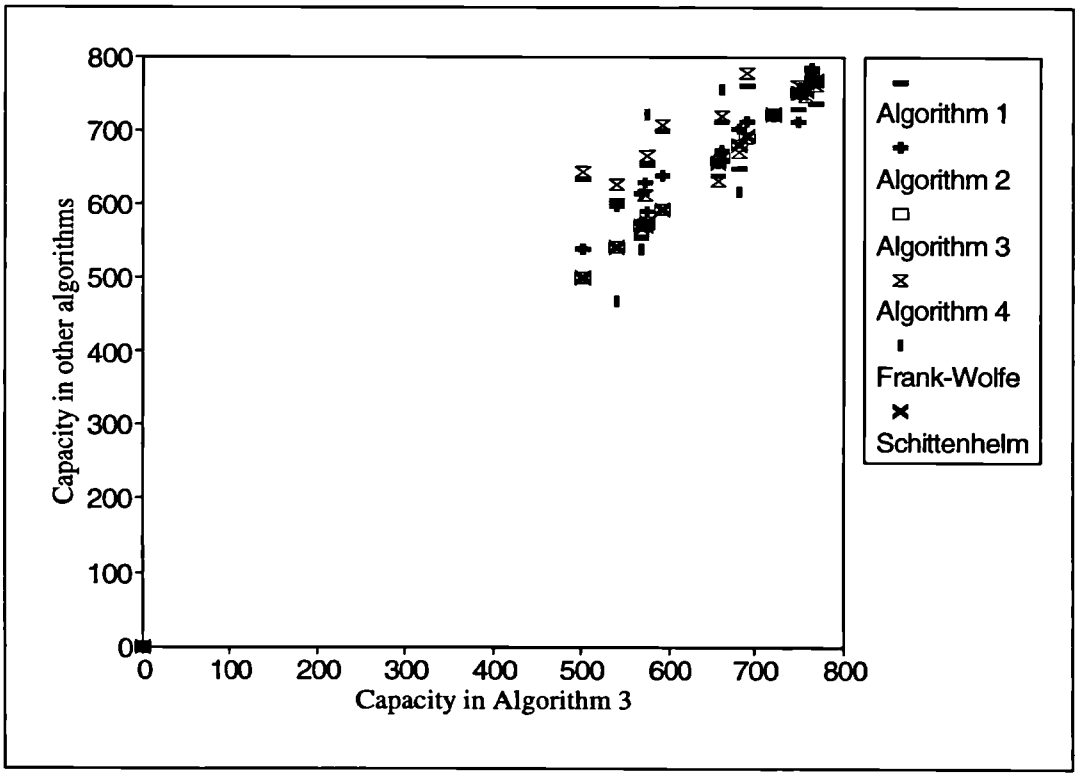


Figure 4-32 The comparison of minor capacities obtained by algorithms on Sioux Falls network when γ =0.25

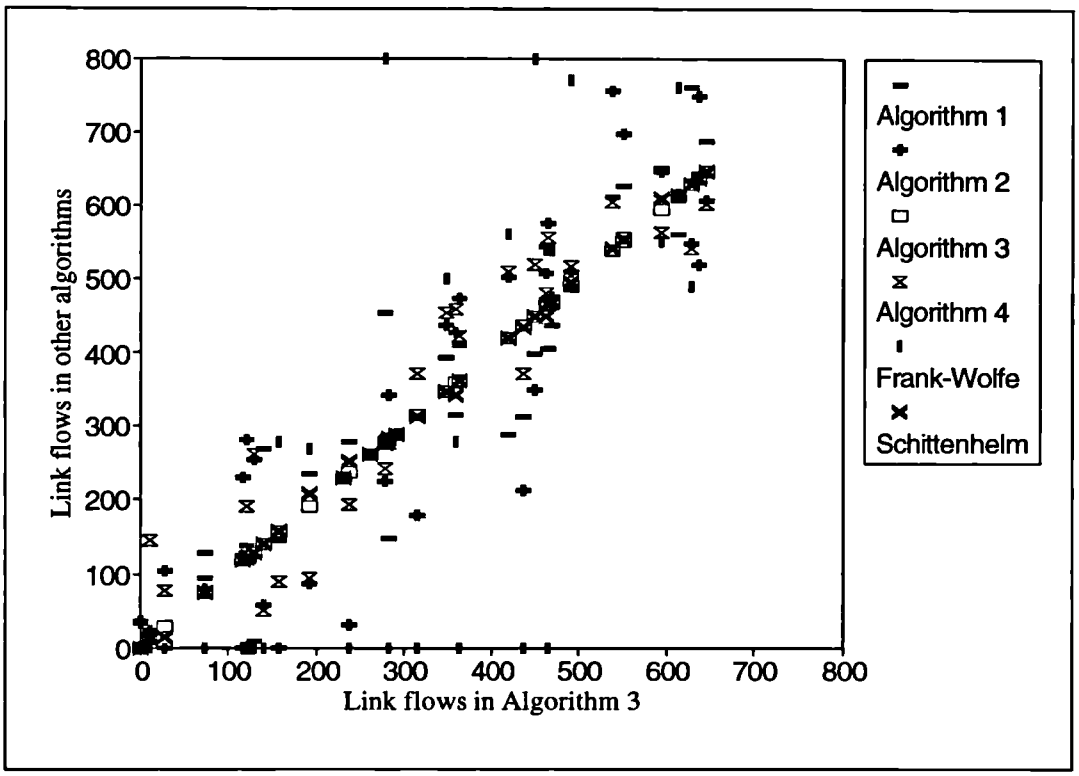


Figure 4-33 The comparison of traffic flows obtained by algorithms on Sioux Falls network when $\gamma = 0.5$

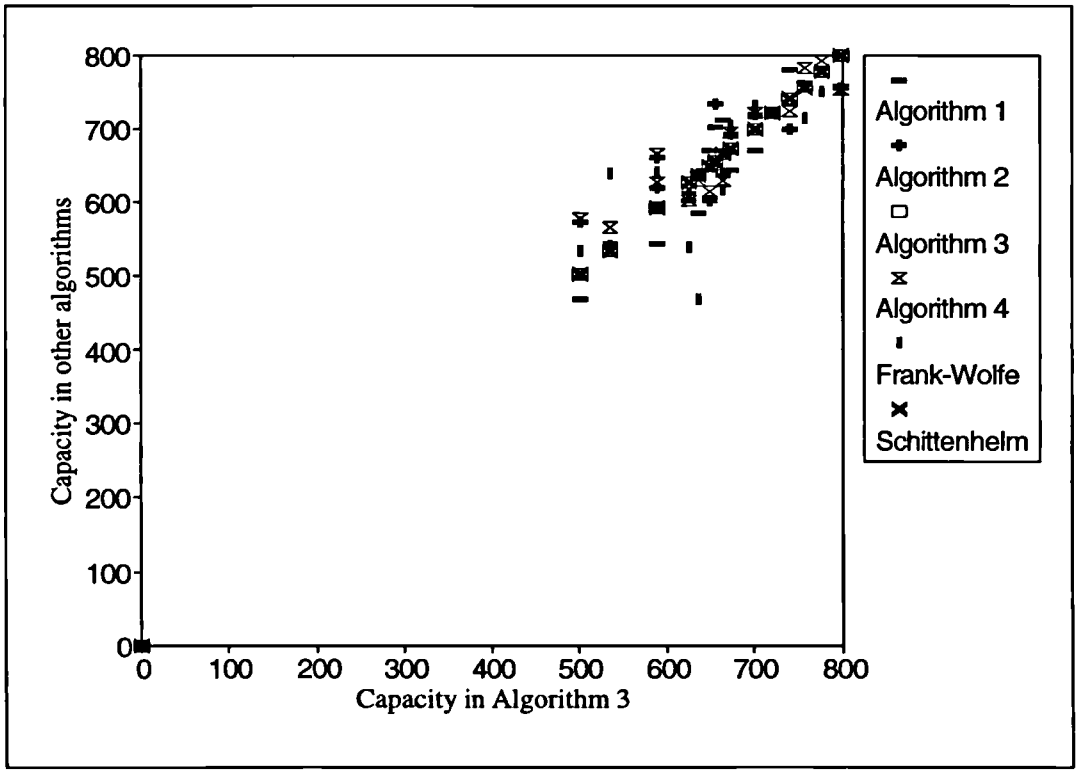


Figure 4-34 The comparison of minor capacities obtained by algorithms on Sioux Falls network when $\gamma = 0.5$

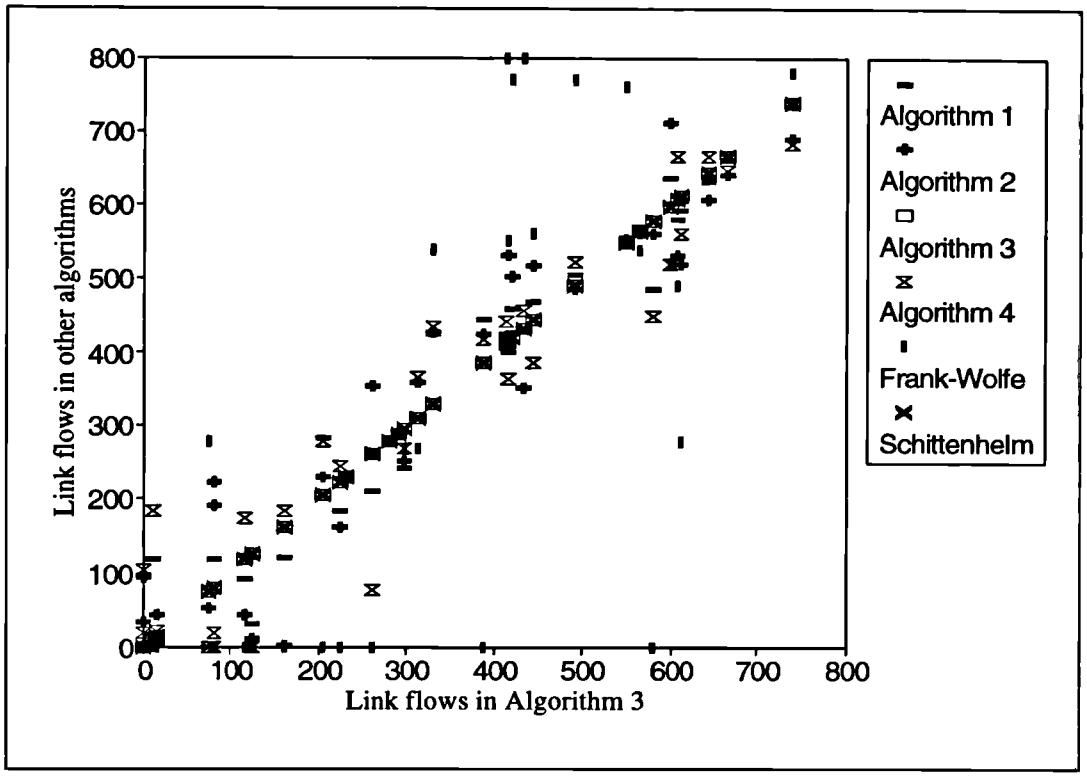


Figure 4-35 The comparison of traffic flows obtained by algorithms on Sioux Falls network when γ =0.75

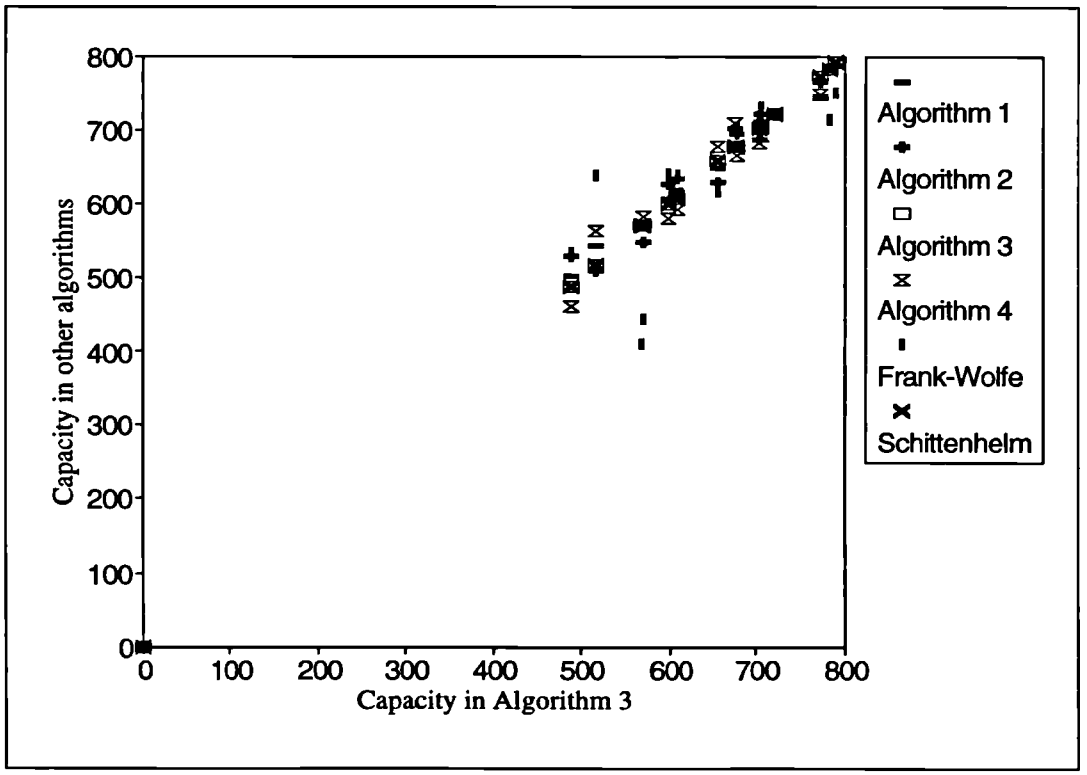


Figure 4-36 The comparison of minor capacities obtained by algorithms on Sioux Falls network when γ =0.75

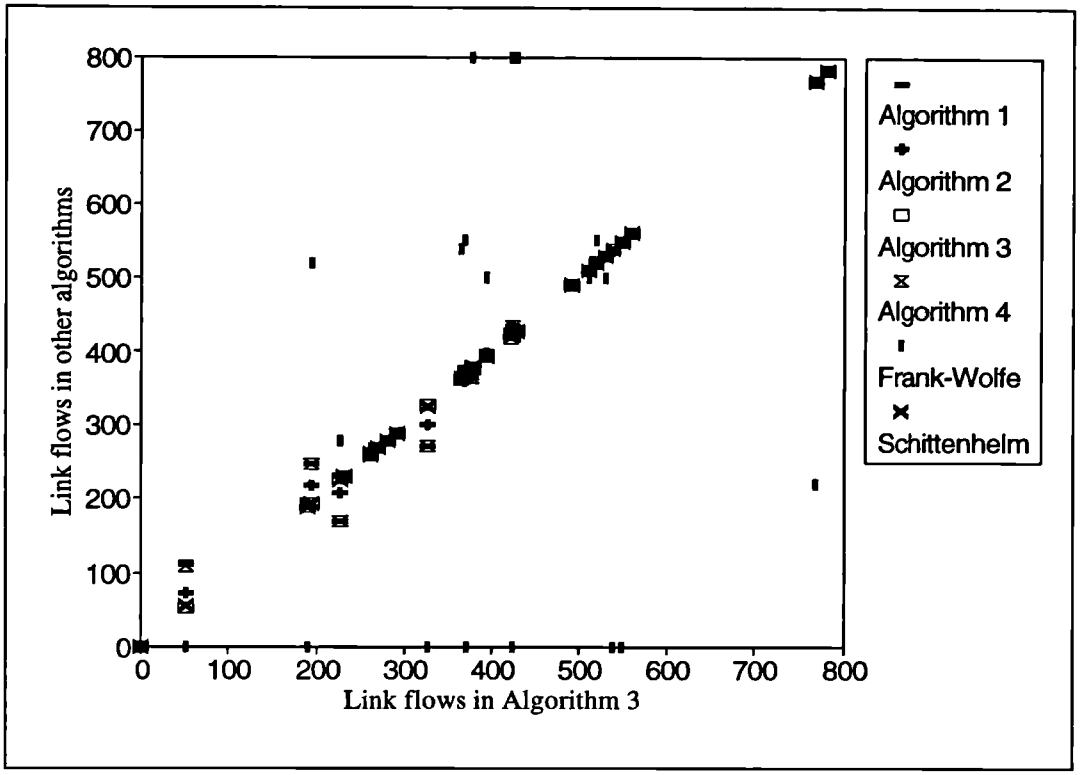


Figure 4-37 The comparison of traffic flows obtained by algorithms on Sioux Falls network when $\gamma = 1.0$ (No junction effect)

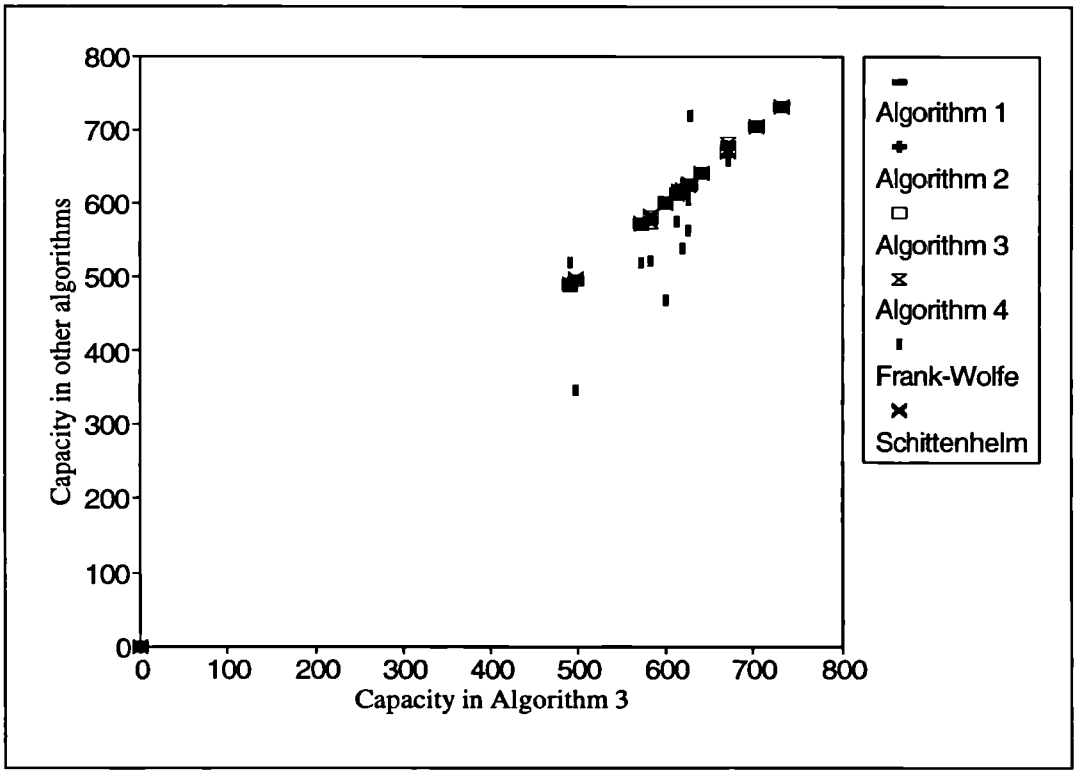


Figure 4-38 The comparison of minor capacities obtained by algorithms on Sioux Falls network when $\gamma = 1.0$ (No junction effect)

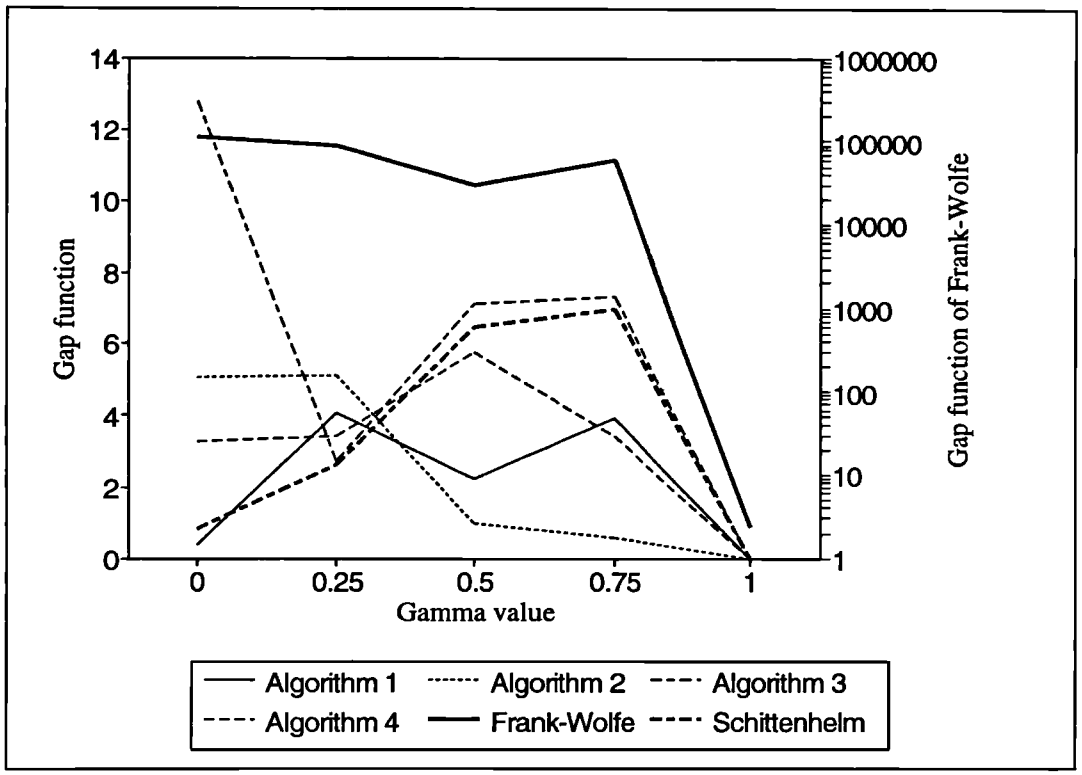


Figure 4-39 The gap function value of algorithms as γ changes on Charlesworth's network

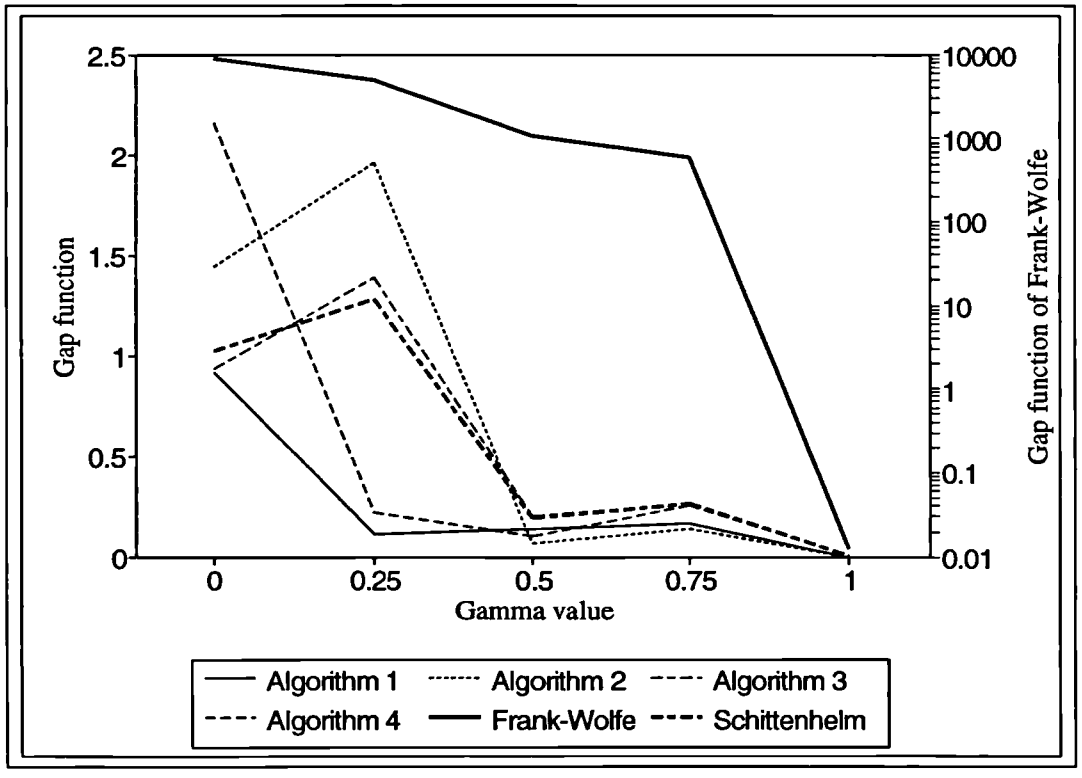


Figure 4-40 The gap function value of algorithms as γ changes on Sioux Falls network

CHAPTER 5 TRAFFIC ASSIGNMENT WITH SIGNAL-CONTROLLED JUNCTION MODELLING

5.1 INTRODUCTION

This chapter presents signal-controlled junction modelling in road traffic assignment. Signal optimisation is reviewed according to a single junction, linked junctions, and a combined signal control and traffic assignment. Delay functions that are used to analyse the performance of the signal-controlled junction are introduced. Solution algorithms are presented to solve this problem and to analyse some properties such as uniqueness and stability. A numerical analysis of this problem is presented.

5.2 SIGNAL OPTIMISATION

5.2.1 Introduction

The coverage of traffic control systems can be classified as one of :

1. individual junction control, when a junction is operated independently from other junctions;
2. arterial systems, when some junctions along a corridor are co-ordinated; and
3. network systems, when the road systems are controlled as a whole.

The style of traffic control systems can be classified as one of:

1. fixed time control, when it is based on the average conditions regardless of time and characteristics of traffic flows;
2. pre-timed control, when it is operated according to the period and day; and
3. actuated control, when it is operated by characteristics of traffic flow

measured at that time.

We will review signal control in the cases of an independent junction, linked junctions, and a combined signal control and traffic assignment in a fixed time control only.

5.2.2 Single junction

Webster (1958) provided systematic mathematical frameworks for signal timing calculation at a single junction. In his method, the arrival pattern of traffic is assumed to be a Poisson type, and green times are calculated according to the maximum ratio of arrival flow to saturation flow for the streams that have right of way during each stage. An approximate formula for the optimal cycle time to minimise overall junction delay was also developed.

Allsop (1971) formulated this signal timing calculation as a convex mathematical programming problem to minimise the total rate of delay on all approaches. Allsop (1972) presented an alternative formulation as a linear programme to maximise capacity. This approach is called *stage-based* because the variables of optimisation are the stage duration, and the stage sequence has to be specified in advance.

Zuzarte Tully (1976) developed a procedure to identify all possible stage sequences at a junction by using a graph theory. Gallivan and Heydecker (1988), and Heydecker and Dudgeon (1987) formulated the signal control problem, within which stage sequences are identified, as a linear programme and a convex mathematical programme to maximise capacity and to minimise delay respectively. This approach is

called *phase-based* because the variable of optimisation is the phase duration, and the stage sequence can be determined partially within the programme itself.

5.2.3 Linked signal control

Network co-ordination is a key issue in linked signal systems. As optimisation criteria, the number of vehicular stop and delay become important. The resulting formulation of this problem is mostly a nonlinear objective function and a non-convex constraint set. Little (1966) proposed a general formulation of arterial co-ordination as a mixed integer programme. Gartner and Little (1972) proposed a dynamic programming approach to solve the network-wide problem. Improta and Sforza (1982) formulated this problem as a binary integer programme. These mathematical methods are thus limited in the size of network that they can accommodate and only give sub-optimal solutions.

TRANSYT (Vincent, Mitchell and Robertson, 1980) is used as an alternative to the mathematical approach. TRANSYT simulates the motions of traffic in the network to obtain cyclic flow profiles. TRANSYT uses a heuristic procedure called hill-climbing to minimise approximately a weighted sum of delay and stops on the networks. This method can be used in a large network but gives only sub-optimal solutions.

5.2.4 Combined signal control and traffic assignment

5.2.4.1 Introduction

Traffic assignment and signal control have played important roles in the transportation planning and traffic engineering. Traffic assignment is used to represent users' behaviour in selecting routes so as to minimise their individual travel costs. A

signal control policy determines the optimal signal settings based on the total travel cost in the system.

These problems have been tackled independently even though the interaction between traffic assignment and signal control problems exists clearly. When the traffic assignment problem is solved in detail, signal control data are required in the cost functions. On the other hand, when signal control is solved, traffic flow is necessary to calculate signal control variables and a cost function.

To improve their realism, the mutual interaction between these two problems should be considered: signal setting gives a delay which will affect the drivers' route choice whilst changes in route choice can cause changes in signal settings.

There are two main approaches to the solution of this combined problem. Firstly, the global optimisation method is for solving the signal optimisation problem and the traffic assignment problem as a whole. Secondly, the iterative approach is to solve signal control and traffic assignment alternately.

5.2.4.2 Global solution approach

This approach is known as the network design problem in which the decision variables correspond to signal timings and the flows are constrained to be in equilibrium condition and consistent with those timings. The cycle time, stage matrix and offsets in some cases are usually assumed to be known values. Only the green split is considered to be a variable. The objective function of this model is the total travel cost expressed as a formulation of the green time and traffic flows. At the solution, the users' behaviour such as Wardrop's first principle should be met.

A bilevel approach can be used to solve the network design problem. This approach consists of upper and lower levels. The upper level concerns the decision of traffic controller with the objective to minimise the total costs on the systems. The total travel cost, expressed as a formulation of the green time and traffic flows, can be used as an objective function. The lower level represents the users' behaviour of selecting their routes. Wardrop's first principle can be used to solve the equilibrium flow pattern for given signal timings. This bilevel approach is a form of Stackelberg game (Fisk, 1984). In this game, a traffic controller decides a signal setting based on which a driver will choose his or her best route from each origin to destination pair.

Tan et al (1979) solved this as a hybrid optimisation using a special nonlinear programming technique. In this method, green splits were calculated, and other variables such as cycle time and offsets were assumed to be fixed. Fisk (1984) formulated this problem as a max-min game theory and proposed a penalty approach for its solutions. Marcotte (1983) suggested a constraint relaxation method, whilst Sheffi and Powell (1983) proposed a feasible descent method to solve this problem. Heydecker and Khoo (1990) developed an iterative method, which used a sequence of linear approximations to the assignment constraints and solved each of the resulting sub-problems by a direct search method.

Each of these methods in this solution approach suffers from their limitations. Firstly, they are restricted to the small size of the network problem. Secondly, they can represent only approximate unrealistic modelling because they need strong assumptions of a cost function and a network structure for better behaviour. Thirdly, their solutions guarantee only local solutions because the problem has a non-convex constraint and a non-linear objective function.

5.2.4.3 Iterative approach

An iterative approach is to solve signal control and traffic assignment alternately. The signal control problem yields some decision variables such as green time and cycle time which are calculated to optimise some measures of performance when flows are fixed. These can then be passed to the traffic assignment problem as fixed input values. The traffic assignment problem solves the user behaviour condition to give traffic flows which are calculated to optimise some measures of performance when some decision variables in signal control are fixed. These can then be passed to the signal control problem as fixed input values. This iterative procedure is repeated until mutually consistent values of the variable are obtained (see Figure 5-1).

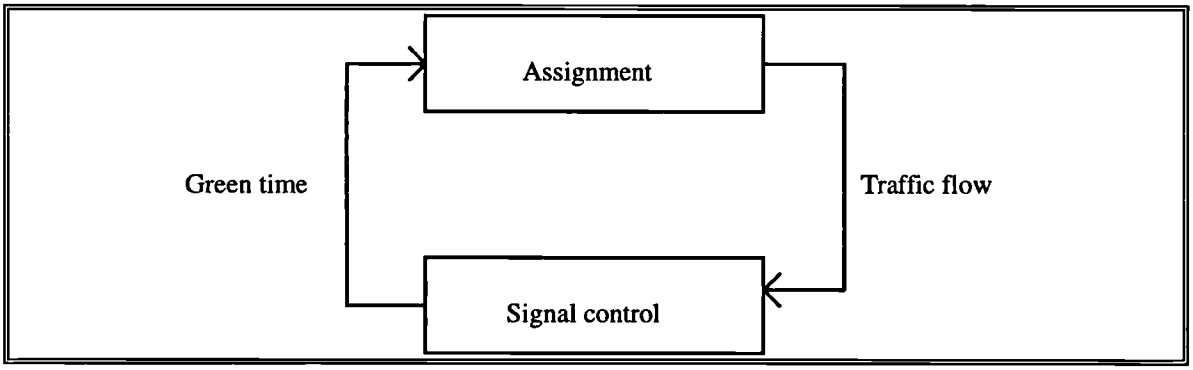


Figure 5-1 Iterative approach between traffic assignment and signal control

Allsop and Charlesworth (1977) introduced this iterative computational procedure for the combined signal setting and road traffic assignment problem. They used the TRANSYT (Vincent, Mitchell and Robertson, 1980) program to calculate the signal timing control parameters and the TRAFFIC (Nguyen and James-Lefebre, 1975) program to calculate the equilibrium assignment pattern.

Gartner, Gershwin, Little and Ross (1980) also used this iterative procedure for the combined signal control and road traffic assignment problem. They included the calculation of offsets in the signal control problem by formulating this as a mixed integer linear programme. The solution of this formulation thus requires a great computational effort.

Smith (1981) introduced a signal control policy that guarantees the existence of a traffic equilibrium that is consistent with it. Smith, Van Vuren, Heydecker and Van Vliet (1987) carried out a comparative stability test between Smith's policy and Webster's policy. They compared the difference in signal settings at equilibrium which results from different initial settings. They found that both Smith's policy and Webster's policy have a stable equilibrium under the low congestion case. However, at the high cogestion case, different initial settings give rise to a significant difference in the ultimate green times by Webster's policy whilst some difference by Smith's policy.

Cantarella, Improta and Sforza (1991) proposed an iterative procedure in which traffic signal setting is calculated in two successive steps: green timing and scheduling at each junction, and signal co-ordination on the network. Green timing and scheduling at a single junction are calculated according to a mixed binary linear programme. Signal co-ordination for the network is calculated by using a branch and bound technique.

5.3 COST FUNCTION

5.3.1 Introduction

The notation that is used here is as follows. Let

d_a be the junction delay on link a

c_a be the link based travel cost on link a

T_a be total cost on link a, corresponding the weighted sum of d_a and c_a

v_a be the total volume of traffic on link a

J_n be junction n

T_{od} be the total demand for travel from o to d

T be the matrix of (T_{od})

C be cycle time

g_a be effective green time for link a

s_a be saturation flow on link a

X_a be degree of saturation on link a $[v_a / (\lambda_a s_a)]$

λa be green time proportion for link a (g_a/C)

We need some definitions of the terms that are used in the signal control of road junctions. A junction can be described by a set of approaches. An *approach* is a part of a road leading to the junction such that all the traffic in the approach has a right of way simultaneously. The traffic at a junction is divided into streams. A *stream* is formed by all the users who cross the junction from the same approach, and form a single queue.

It is assumed that vehicles depart from a queue at a constant rate when they have right of way. The *effective green time*, g is the duration of the interval in which vehicles cross the stop line constantly. An *effective red time*, r represents the period during which vehicles cannot cross. The *saturation flow*, s is defined for each stream as the average flow that can cross the junction in unit time when a queue exists at the approach.

The *cycle time*, C is defined to be the sum of the effective green and red time as follows:

$$C = g + r$$

The cycle can be divided into *stages* during which the signal indications are constant at either red or green.

Allsop (1992) defined the traffic *capacity* of a stream with given signal timing as the average departing flow of traffic per unit time when every green time is saturated. This can be calculated as λs where λ is the proportion of cycle effective green (g/C) and s is the saturation flow in vehicles per unit time.

5.3.2 Delay function

We can denote the total travel time consisting of free-flow travel time, and link and junction delays:

$$T_a(\mathbf{v}) = t_0 + \gamma[\, c_a(v_a) - t_0] \; + \; (1-\gamma)\, d_a(\mathbf{v})$$

where t_0 is the free-flow travel time; $d_a(\mathbf{v})$ is junction delay depending on the junction type and calculated from Webster's formula; $c_a(v_a)$ is the link cost function and calculated from the BPR function; and γ is a parameter to control the relative influence of link and junction delays, and represents a user's perception on the travel cost.

Webster's two term delay is used commonly to estimate the mean delay at a fixed-time signal controlled junction for steady state:

$$d(v,\lambda,X,C) = \frac{9}{10}\left[\frac{C(1-\lambda)^2}{2(1-\lambda X)} + \frac{X^2}{2v(1-X)}\right] \qquad (5.1)$$

where $X = v / (s\lambda)$

Webster's delay can be decomposed into two components: uniform and random delay. The first term of the delay represents the uniform delay which is due to the

alternation of green and red intervals when vehicles arrive regularly at a rate less than capacity. The second term of the delay represents the random delay which is due to the randomness of arrivals of vehicles at the stop line and the consequent overflow of vehicles at the end of some effective green periods (see Figure 5-2).

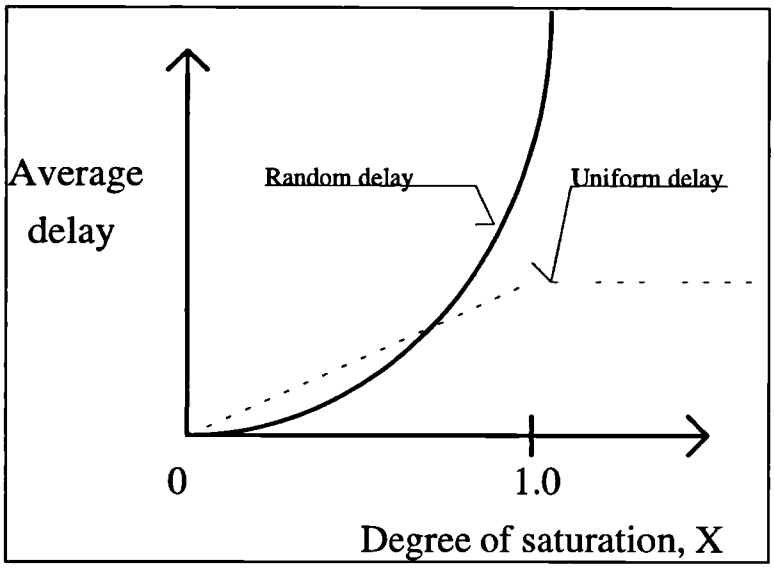


Figure 5-2 Uniform and random delay (Source: Allsop, 1992)

During the initial iterations of equilibrium assignment solution methods, the traffic flow may exceed the capacity on some links. To avoid numerical difficulties in this case, Webster's delay function should be extended into the oversaturated region. We approximate linearly this delay formula about a saturation rate, $X^* < 1$. Thus, for $v \geq X^* g s / C$, we use the approximation

$$\bar{d}(v,\lambda,X,C) = 0.9\left[d(X^*, {}^{X^* gs}\!/\!_C) + (v - {}^{X^* gs}\!/\!_C)\frac{dd}{dv}\bigg|_{v=\frac{X^* gs}{c}} \right] \qquad (5.2)$$

$$= 0.9\left[\frac{(C-g)^2}{2(C-gX^*)} + \frac{(v - {}^{X^* gs}\!/\!_C)(C-g)^2}{2(C-gX^*)} + \frac{X^{*2}}{2(1-X^*)^{X^* gs}\!/\!_C} + \frac{(v - {}^{X^* gs}\!/\!_C)(2X^* - X^{*2})}{2^{X^* gs}\!/\!_C (1-X^*)^2} \right]$$

Webster's method (1958) to calculate the cycle time and green time is used in this study. The cycle time that gives the minimum delay is given approximately by:

$$C_0 = \frac{1.5L + 5}{1 - Y}$$

$$\text{where } Y = \sum_{i=1}^{n} y_i *$$

$$L = \sum_{i=1}^{n} L_i$$

where

y_i* is the flow ratio v_i/s_i for the representative (or, maximum value) approach for stage i,

L is the total lost time per cycle,

n is the number of stages,

L_i is the duration of the effective intergreen time following stage i.

Green times are calculated according to the ratio of y* values as follows:

$$g_i = \frac{y_i *}{Y}(C - L)$$

$$r_i = C - g_i$$

5.4 FORMULATION AND SOLUTION METHOD

5.4.1 Introduction

The formulation and solution method in signal controlled junction modelling are very similar to those of priority controlled junction modelling. For the formulation, see section 4.4. We will repeat some of the parts in the priority case for a solution method.

5.4.2 Solution method

An iteration of a *dynamic adjustment process* for solving this combined signal controlled junction modelling and traffic assignment is the combination of the calculation of green times for the given feasible traffic flows, and the assignment of travel demand using these green times. The dynamic adjustment process continues until a convergence criterion is satisfies. We can say that the dynamic adjustment process of this model has a diagonalised procedure. This is because signal parameters are calculated whilst traffic flows are assumed to be fixed, and traffic flows are calculated whilst signal parameters are fixed and so on. An existing diagonalisation algorithm therefore can be used and the following is a diagonalisation of Algorithm 1.

A diagonalisation of Algorithm 1 for the combined signal control and traffic assignment

Step 0: (Initialisation)
Iteration n = 0
Identify a feasible point, $\mathbf{v}^n \in \mathcal{B}$
Set $W^n = \{\mathbf{v}^n\}$

Step 1: (Linear subproblem)
1-1. Calculate green times based on the Webster's min-max policy
(see equation 2.38 in chapter 2)
1-2. Calculate the total link cost

$$T_a(\mathbf{v}^n, \lambda_a^n) = t_0 + \gamma[c_a(v_a^n) - t_0] + (1-\gamma)\bar{d}_a(\mathbf{v}^n, \lambda_a^n)$$

where $c_a(v_a{}^n)$ is the BPR function and $\bar{d}_a(\mathbf{v}^n, \lambda_a{}^n)$ is the approximated Webster's formula
1-3. Perform all-or-nothing assignment based on the total link cost

$$\mathbf{u}^n = \arg \min_{\mathbf{u}} \{\mathbf{T}(\mathbf{v}^n, \boldsymbol{\lambda}^n) \cdot \mathbf{u} \ ; \ \mathbf{u} \in \mathcal{B}\}$$

1-4. Convergence test

If $G(\mathbf{v}^n, \boldsymbol{\lambda}^n) = \mathbf{T}(\mathbf{v}^n, \boldsymbol{\lambda}^n) \cdot (\mathbf{v}^n - \mathbf{u}^n) \approx 0$, stop: optimum solution is $\mathbf{v}^n$
Otherwise $W^{n+1} = W^n \cup \{\mathbf{u}^n\}$

Step 2: (Master subproblem)

Let $\mathbf{v}^{n+1} = \arg \min_{\mathbf{v}}\{G(\mathbf{v},\lambda);\ \mathbf{v} \in H(W^{n+1})\}$

n=n+1

Return to step 1.

The other simplicial decomposition algorithms in chapter 3 are similarly represented as the diagonalisation of Algorithm 1. For example, the diagonalisation of the Schittenhelm algorithm is represented by replacing the master subproblem of Algorithm 1 into that of Schittenhelm's. For Algorithms 3 and 4, the initialisation step of Algorithm 1 is replaced by each of initialisation steps in Algorithms 3 and 4. For the detailed description of the algorithms, see section 3.4.

5.5 COST FUNCTION ANALYSIS FOR GOOD BEHAVIOUR

5.5.1 Introduction

In this section, the good behaviour conditions for the combined signal control and traffic assignment are tested in a similar way to that of the priority control modelling. Firstly, we calculate symmetric equilibrium values of traffic flows and green times on the small example network, and obtain the values and signs of the Jacobian matrix. Secondly, we run the model to test whether or not the model obtain a unique stable equilibrium by using different initial points. Thirdly, we compare the two test results to see whether the theoretical conditions for good behaviour correspond well with the experimental results using the model.

The condition for good behaviour may not be satisfied by the junction delay term. When the travel-time formula,

$$T_a(\mathbf{v}) = t_0 + \gamma[\, c_a(v_a) - t_0] \ +\ (1-\gamma)\, d_a(\mathbf{v})$$

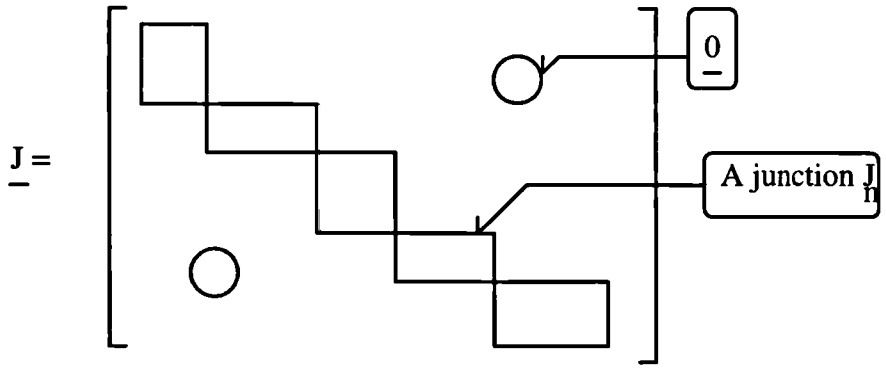
is used, the resulting Jacobian matrix can be expressed as:

$$J_{ab} = \gamma \frac{\partial c_a}{\partial v_b} \delta_{ab} + (1-\gamma) \frac{\partial d_a}{\partial v_b}$$

$$\text{where } \delta_{ab} = \begin{cases} 1, & \text{if } a = b \\ 0, & \text{otherwise} \end{cases}$$

Note that the link travel time function, $c_a(v_a)$ contributes only to the diagonal elements of the Jacobian matrix. In this study of good behaviour, the delay function, $d_a(\mathbf{v})$ used is the original Webster's formula. Convergence behaviour of this problem depends on the relative magnitude of the junction and link delay terms and the size of the parameter γ.

The Jacobian matrix analysis in the signal controlled junction is similar to that of the priority junction case. The Jacobian matrix of a signal controlled junction is not in general positive definite (see Heydecker, 1983). We can construct the Jacobian matrix by arrangement in the following form. The rectangles each represent signal controlled junctions while the circles each represent the zero vectors of derivatives of costs that are not related with other flows.



where J_n has magnitude of elements in the two stream junction as follows:

$$J_n = \begin{bmatrix} \frac{\partial T_a}{\partial v_a} & \frac{\partial T_a}{\partial v_b} \\ \frac{\partial T_b}{\partial v_a} & \frac{\partial T_b}{\partial v_b} \end{bmatrix}$$

The Jacobian matrix J is positive definite if and only if the symmetric part $J + J^T$ is, and hence:

$$\frac{\partial T_a}{\partial v_a} > 0, \quad \frac{\partial T_b}{\partial v_b} > 0 \text{ and}$$

$$4\frac{\partial T_a}{\partial v_a}\frac{\partial T_b}{\partial v_b} - (\frac{\partial T_a}{\partial v_b} + \frac{\partial T_b}{\partial v_a})^2 > 0$$

We can rewrite Webster's two term delay in order to calculate its derivatives more conveniently as:

$$d_a = \frac{C(1-\lambda_a)^2}{(1-v_a/s_a)} + \left(\frac{1}{\lambda_a s_a - v_a} - \frac{1}{\lambda_a s_a}\right)$$

where according to Webster's signal settings policy (1958),

$$\lambda_a = \frac{(C-L)v_a/s_a}{C\sum_k v_k/s_k}$$

In signal controlled junction modelling, the partial derivatives can be obtained by the following relation:

$$\frac{dd_a}{dv_b} = \frac{\partial d_a}{\partial v_a}\delta_{ab} + \frac{\partial d_a}{\partial \lambda_a}\frac{\partial \lambda_a}{\partial v_b}$$

where

$$\frac{\partial d_a}{\partial \lambda_a} = \frac{-2C(1-\lambda_a)}{(1-v_a/s_a)} - \frac{s_a}{(\lambda_a s_a - v_a)^2} + \frac{s_a}{(\lambda_a s_a)^2}$$

For Webster's signal control policy,

$$\frac{\partial \lambda_a}{\partial v_b} = \frac{(C-L)}{(C\sum_k v_k/s_k)^2}\left(\frac{v_a}{s_a s_b}\right)$$

In addition,

$$\frac{\partial d_a}{\partial v_a} = \frac{C(1-\lambda_a)^2}{s_a(1-v_a/s_a)^2} + \frac{1}{(\lambda_a s_a - v_a)^2}$$

For the BPR link-cost function,

$$\frac{dc_a}{dv_a} = \alpha_1 \alpha_2 t_0 \left(\frac{v_a}{Q_a}\right)^{\alpha_2 - 1}$$

We can indicates the relative values of the elements in the Jacobian matrix as:

$$J_n = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} = \begin{bmatrix} \text{small} & \text{large} \\ \text{large} & \text{small} \end{bmatrix}$$

This indicates that the Jacobian matrix is not a P-matrix and thus cannot be positive-definite. Thus, neither the necessary nor the sufficient conditions for good behaviour of assignment in arbitrary networks will be satisfied when signal-controlled junctions are modelled in this way. Because of this, the behaviour of equilibrium assignment with signal controlled junctions modelled in this way is indeterminate.

We now use a simple symmetric network (see Figure 5-3) which is the same as that used in a priority controlled problem to test the model. It has 2 origins and 2 destinations, 6 nodes and 8 links. The demand from origin 1 is entirely to destination 5 and from origin 2 to destination 6. Each of nodes 3 and 4 represents a signal controlled junction. The capacity of each link is 2000 vehicles / hour. We assume that signal is operated in two stages. The cycle time is taken as fixed at 60 seconds with a lost time of 4 seconds following each stage. Minimum green for each stage is 7 seconds and maximum green time for each stage is therefore 45 seconds. We vary the demand for

each origin-destination pair, the weighting factor γ, and the initial green times in order to calculate the Jacobian value as well as to test uniqueness and stability.

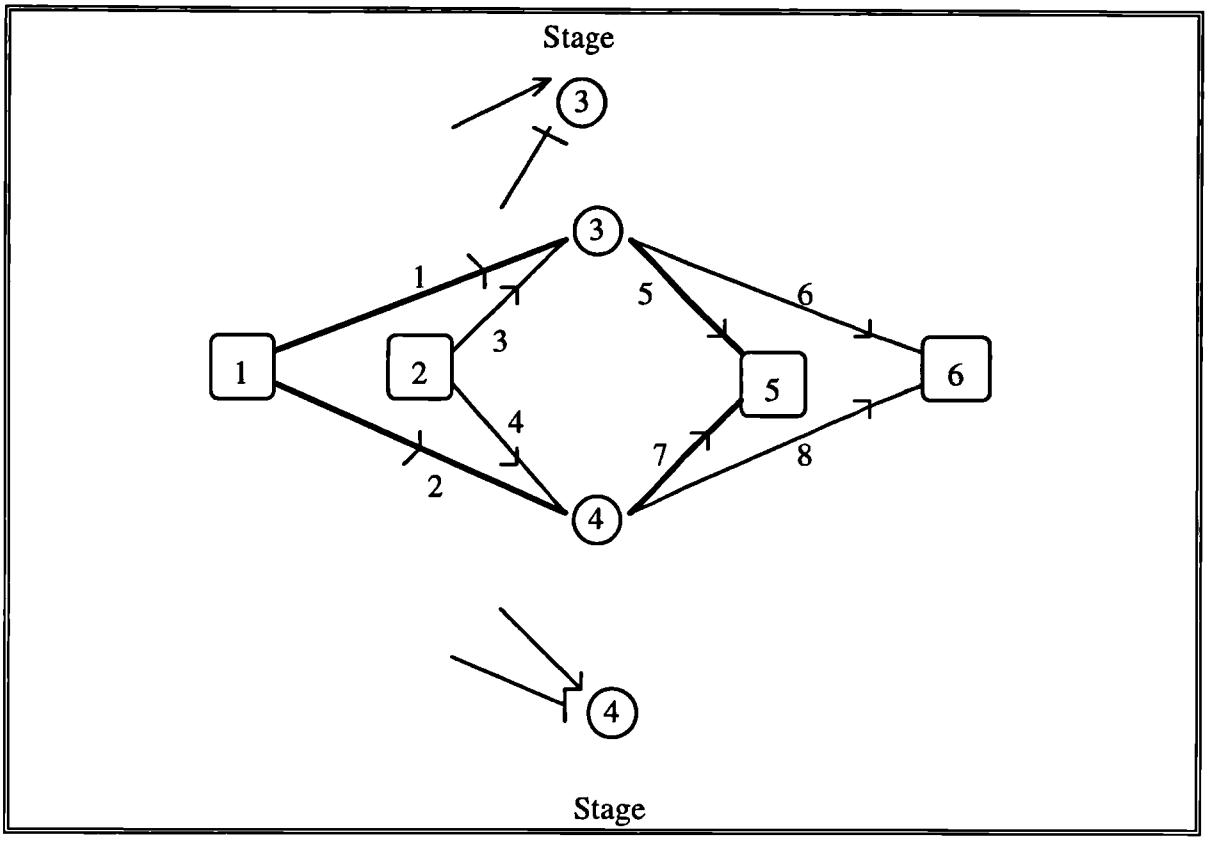


Figure 5-3 Example network

Key :	n	origin or destination n
	n	node n

In this example, an equilibrium solution of green time and traffic flows is clearly provided by the symmetric values:

$$\mathbf{v}^* = (v^*_1, v^*_2, v^*_3, v^*_4) = 1/2(T_{15}, T_{15}, T_{26}, T_{26})$$

$$\mathbf{g}^*(\mathbf{v}^*) = (g^*_1(\mathbf{v}^*), g^*_2(\mathbf{v}^*), g^*_3(\mathbf{v}^*), g^*_4(\mathbf{v}^*)) = (26, 26, 26, 26)$$

Tables 5-1 and 5-2 show the symmetric equilibrium traffic flows and green times respectively which are calculated at various levels of demands T_{15} and T_{26}. The lower-left parts of the tables are empty because the values of the upper-right parts are symmetric to those of the lower-left parts due to the symmetric example network. The

sign - in the lower-right parts indicates that the region is overloaded so that the values cannot be calculated appropriately because in the Jacobian analysis, we use only the original Webster's delay formula (equation 5.1).

O/D	T_{26}					
	v^*_1/v^*_3	200	800	1400	2000	2600
T_{15}	200	100/100	100/400	100/700	100/1000	100/1300
	800		400/400	400/700	400/1000	400/1300
	1400			700/700	700/1000	-
	2000				-	-
	2600				-	-

Table 5-1 The symmetric equilibrium traffic flows for calculating Jacobian matrix

O/D	T_{26}					
	g^*_1/g^*_3	200	800	1400	2000	2600
T_{15}	200	26/26	10/42	7/45	7/45	7/45
	800		26/26	19/33	15/37	12/40
	1400			26/26	21/31	-
	2000				-	-
	2600				-	-

Table 5-2 The symmetric equilibrium green times for calculating Jacobian matrix

Table 5-1 and Table 5-2 are used to calculate the Jacobian matrix in Table 5-3 which gives the signature of the Jacobian matrix in terms of the sign of each of its determinant, and its diagonal elements J_{11} and J_{22} as the value of the parameter γ is changed from 0.0 to 0.75 in increments of 0.25. In this table, a different signature of the Jacobian matrix is obtained only when the value of γ is 0.75 with the demand $(T_{15},T_{26})=(200,2600)$ which is lightly shaded. However, the values of the determinant of the Jacobian and its diagonal elements differ as the value of γ is varied. In the lower demand cases, which are upper-left parts of the table, the Jacobian values are positive whilst its elements J_{11} and J_{22} are negative. In the higher demand cases, which are the lower-right parts of the table, the Jacobian values are negative whilst its elements J_{11} and

J_{22} are positive. In shaded parts, all the values are negative. From this table, we can see that the Jacobian is not a P-matrix at the symmetric equilbrium in this network for any level of demand.

O/D			T_{26}			
	γ=0.0 0.25 0.5 0.75	200	800	1400	2000	2600
	200	+/-/- +/-/- +/-/- +/-/-	+/-/- +/-/- +/-/- +/-/-	+/-/- +/-/- +/-/- +/-/-	-/-/- -/-/- -/-/- -/-/-	-/-/- -/-/- -/-/- -/-/+
T_{15}	800		+/-/- +/-/- +/-/- +/-/-	-/-/- -/-/- -/-/- -/-/-	-/-/- -/-/- -/-/- -/-/-	-/+/+ -/+/+ -/+/+ -/+/+
	1400			-/-/- -/-/- -/-/- -/-/-	-/+/+ -/+/+ -/+/+ -/+/+	-

Table 5-3 The signs of the Jacobian
(Key: the signs of each symbol indicate respectively the signs of det $(J+J^t)$ / J_{11}/ J_{22} ; and the signature - in the bottom right corner indicates overloaded region)

We define global and local stability as:

Definition: global stability: An equilibrium of the combined signal control and traffic assignment model is *globally stable* if any initial flow in the feasible region leads to that equilibrium traffic flow and the associated green time by the dynamic adjustment process of this model.

Definition: local stability: An equilibrium of the combined signal control and traffic assignment model is *locally stable* if any initial perturbation from an equilibrium that is sufficiently small leads to a return to that equilibrium by the dynamic adjustment process of this model.

Note that in the present study of stability, green times are calculated according to the Webster's method and the assignment is calculated by a diagonalisation of the Frank-Wolfe algorithm.

5.5.2 Global stability test

In order to test the global stability, we use three extreme cases as initial values and run a diagonalisation of the Frank-Wolfe algorithm starting with these initial values to calculate the green time and the traffic flows. These calculated values by the F-W algorithm are compared with the symmetric equilibrium value in the small example network described above to see whether or not the calculated values correspond to the symmetric equilibrium value. In all these tests, the value $\gamma = 0$ is used.

Test 1. When initial flows are given as: $(v_1, v_2, v_3, v_4)=(T_{15}, 0, 0, T_{26})$

We obtain the calculated traffic flow and green time on the links by a diagonalisation of the F-W algorithm shown respectively in Tables 5-4 and 5-5 using initial traffic flow $\mathbf{v}^0=(T_{15}, 0, 0, T_{26})$. The number of iterations used by this algorithm, and the final gap value from equation (2.15) are shown in Table 5-6. We compare the calculated equilibrium traffic flow in Table 5-4 by the diagonalisation of the F-W with those symmetric value in Table 5-1 to test the global stability of that equilibrium. The results of this global stability test are shown in Table 5-7. To determine the stability or otherwise, we used definition of global stability given above. In this case, only three of the cases lead to the calculation of a symmetric equilibrium: these are the cases where the demand is most asymmetric.

O/D	T_{26}				
	v^n_1/v^n_3	200	800	1400	2000	2600
T_{15}	200	200/0	200/0	200/14	100/1000	116/1302
	800		800/0	800/14	800/603	578/1122
	1400			1386/14	700/1000	-
	2000				-	-
	2600					-

Table 5-4 The calculated traffic flows by the F-W in global stability test 1
(Key: v^n is traffic flow on link after iteration n which is shown in Table 5-6)

O/D	T_{26}				
	g^n_1/g^n_3	200	800	1400	2000	2600
T_{15}	200	45/7	45/7	45/7	7/45	7/45
	800		45/7	45/7	30/22	18/34
	1400			45/7	21/31	-
	2000				-	-
	2600					-

Table 5-5 The calculated green times by F-W in global stability test 1
(Key: g^n is green time after iteration n which is shown in Table 5-6)

O/D	T_{26}				
	n/gap	200	800	1400	2000	2600
T_{15}	200	1/0.0	1/0.0	50/16.12	50/4.25	50/589.2
	800		1/0.0	50/16.12	15/0.0	50/74.25
	1400			50/32.25	50/5.25	-
	2000				-	-
	2600					-

Table 5-6 The iteration number used by F-W in global stability test 1
(Key: n is the number of iteration used; and gap is the gap value)

O/D	T_{26}					
		200	800	1400	2000	2600
T_{15}	200	U	U	U	S	S
	800		U	U	U	S
	1400			U	U	-
	2000				-	-
	2600					-

Table 5-7 The result of global stability test 1
(Key: U and S represent stability and unstability whether symmetric equilibrium is obtained in the test respectively)

Test 2. When initial flows are given as: $(v_1, v_2, v_3, v_4)=(T_{15}/2, T_{15}/2, T_{26}/2, T_{26}/2)$

We obtain the calculated traffic flow and green time on the link by a diagonalisation of the F-W algorithm shown respectively in Tables 5-8 and 5-9 using initial traffic flows $v^0=(T_{15}/2, T_{15}/2, T_{26}/2, T_{26}/2)$. The number of iterations used by this algorithm, and the final gap value are shown in Table 5-11. We compare the calculated traffic flows from the F-W in Table 5-8 with those symmetric values in Table 5-1 to test the global stability of that equilibrium. The results of this global stability test are in Table 5-10. To determine the stability or otherwise, we followed the definition of the global stability given above. In this case, the symmetric assignments were all stable.

O/D	T_{26}					
	v^n_1/v^n_3	200	800	1400	2000	2600
T_{15}	200	100/100	100/400	100/700	100/1000	100/1300
	800		400/400	400/700	400/1000	400/1300
	1400			700/700	700/1000	-
	2000				-	-
	2600					-

Table 5-8 The calculated traffic flows by F-W in global stability test 2
(Key: iteration n is shown in Table 5-10)

O/D	T_{26}					
	g^n_1/g^n_3	200	800	1400	2000	2600
T_{15}	200	26/26	10/42	7/45	7/45	7/45
	800		26/26	19/33	15/37	12/40
	1400			26/26	21/31	-
	2000				-	-
	2600					-

Table 5-9 The calculated green times by F-W in global stability test 2
(Key: iteration n is shown in Table 5-10)

O/D	T_{26}					
	n/gap	200	800	1400	2000	2600
T_{15}	200	2/0.0	3/0.0	2/0.0	3/0.0	4/0.0
	800		3/0.0	4/0.0	4/0.0	50/0.06
	1400			50/0.06	50/4.38	-
	2000				-	-
	2600					-

Table 5-10 The iteration number used by F-W in global stability test 2

O/D	T_{26}					
		200	800	1400	2000	2600
T_{15}	200	S	S	S	S	S
	800		S	S	S	S
	1400			S	S	-
	2000				-	-
	2600					-

Table 5-11 The result of global stability test 2

Test 3. When initial flows are given as: $(v_1, v_2, v_3, v_4)=(T_{15}, 0, T_{26}, 0)$

We obtain the calculated traffic flows and green times by a diagonalisation of the F-W algorithm which are shown respectively in Tables 5-12 and 5-13 using initial traffic flow $\mathbf{v}^0=(T_{15}, 0, T_{26}, 0)$. The number of iterations required to converge or terminate, and the final gap value are shown in Table 5-14. We compared the calculated traffic flow by the F-W in Table 5-12 with those symmetric value in Table 5-1 to test the global stability of that equilibrium. The results of this global stability test are in Table 5-15. To determine the stability or otherwise, we followed the definition of the global stability given above. In this case, the calculated equilibria are symmetric in all but two cases.

O/D	T_{26}					
	v^n_1/v^n_3	200	800	1400	2000	2600
T_{15}	200	200/200	0/800	0/1386	100/1000	84/1302
	800		400/400	400/700	400/1000	222/1478
	1400			700/700	700/1000	-
	2000				-	-
	2600					-

Table 5-12 The calculated traffic flows in global stability test 3
(Key: iteration n is shown in Table 5-14)

O/D	T_{26}				
	g^n_1/g^n_3	200	800	1400	2000	2600
T_{15}	200	26/26	7/45	7/45	7/45	7/45
	800		26/26	19/33	15/37	7/45
	1400			26/26	21/31	-
	2000				-	-
	2600					-

Table 5-13 The calculated green times by F-W in global stability test 3
(Key: iteration n is shown in Table 5-14)

O/D	T_{26}				
	n/gap	200	800	1400	2000	2600
T_{15}	200	1/0.0	1/0.0	50/0.06	50/3.75	50/555.4
	800		3/0.0	4/0.0	5/0.0	50/20.75
	1400			3/0.0	50/2.12	-
	2000				-	-
	2600					-

Table 5-14 The iteration number used by F-W in global stability test 3

O/D	T_{26}					
		200	800	1400	2000	2600
T_{15}	200	S	U	S	S	S
	800		S	S	S	U
	1400			S	S	-
	2000				-	-
	2600					-

Table 5-15 The result of global stability test 3
(Key: U and S represent stability and unstability whether symmetric equilibrium is obtained in the test respectively)

These global stability tests show that this combined signal control and traffic assignment model appears to have multiple solutions which are dependent on the initial values. Thus this model is not always globally stable and in particular the symmetric equilibrium is not obtained. In the global test 2, good initial values such as those of equilibrium solutions allow returning to the symmetric equilibrium. In the cases of demands (T_{15}, T_{26}) = (200, 2000) and (200, 2600), stable solutions are obtained in all three global stability tests even though these cells are not associated with a positive definite Jacobian matrix. In the cases of demands (200,200), (800,800) and (1400,1400),

multiple solutions are obtained in these tests as (g_1, g_3)=(26,26) and (45,7). In the demand (800,1400), multiple green times, (19,33), and (45,7) are obtained whilst in the demand (800,2000), multiple green times, (15,37) and (30,22) are obtained via the global stability test. In each case, the final gap values indicate a good degree of convergence to equilibrium whether it is obtained symmetric equilibrium or not. The main exception to this is the demand (200,2600) in test 3 where the final gap value is more than 500 seconds after 50 iterations.

5.5.3 Local stability tests

To test local stability as defined above, we make a small perturbation of traffic flow about the symmetric equilibrium value and run the diagonalisation of the Frank-Wolfe algorithm using these values. In this case, we use the initial values $\mathbf{v}^* + \varepsilon$ with each of the perturbations ε=(+1,-1,+1,-1), (+1,-1,-1,+1) and (-1,+1,+1,-1).

Test 1. When the perturbation is ε=(+1,-1,+1,-1)

We obtain the calculated traffic flow and green time on the link by a diagnalisation of the F-W algorithm shown respectively in Table 5-16 and Table 5-17 using initial traffic flow $\mathbf{v}^0 = \mathbf{v}^* + (+1,-1,+1,-1)$. The number of iterations required to converge or terminate, and the final gap value are shown in Table 5-18. We compare the calculated traffic flows in Table 5-16 by the F-W with those symmetric values in Table 5-1 to test the local stability of that equilibrium. The results of the local stability test are Table 5-19. To determine the stability or otherwise, we followed the definition of the local stability given above. In this case, stability appears to obtain all cases except for the demand (200, 800) which converges to an all-or-nothing solution in just 5 iterations.

O/D	T_{26}					
	v^n_1/v^n_3	200	800	1400	2000	2600
T_{15}	200	100/100	200/0	100/700	100/1000	100/1300
	800		400/400	400/700	400/1000	400/1300
	1400			700/700	700/1000	-
	2000				-	-
	2600					-

Table 5-16 The calculated traffic flows by F-W in local stability test 1
(Key: iteration n is shown in Table 5-18)

O/D	T_{26}					
	g^n_1/g^n_3	200	800	1400	2000	2600
T_{15}	200	26/26	45/7	7/45	7/45	7/45
	800		26/26	19/33	15/37	12/40
	1400			26/26	21/31	-
	2000				-	-
	2600					-

Table 5-17 The calculated green time by F-W in local stability test 1
(Key: iteration n is shown in Table 5-18)

O/D	T_{26}					
	n/gap	200	800	1400	2000	2600
T_{15}	200	3/0.0	5/0.0	2/0.0	3/0.0	4/0.0
	800		2/0.0	4/0.0	4/0.0	50/4.06
	1400			50/0.06	4/0.0	-
	2000				-	-
	2600					-

Table 5-18 The iteration number used by F-W in local stability test 1

O/D	T_{26}					
		200	800	1400	2000	2600
T_{15}	200	S	U	S	S	S
	800		S	S	S	S
	1400			S	S	-
	2000				-	-
	2600					-

Table 5-19 The result of local stability test 1
(Key: U and S represent stability and unstability whether symmetric equilibrium is obtained in the test respectively)

Test 2. When the perturbation is ε=(+1,-1,-1,+1)

We obtain the calculated traffic flow and green time on link by a diagonalisation of the F-W algorithm shown respectively in Table 5-20 and Table 5-21 using initial traffic flow $\mathbf{v}^0 = \mathbf{v}^* + (+1,-1,-1,+1)$. The number of iterations required to converge or terminate, and the final gap value are shown in Table 5-22. We compare the calculated equilibrium traffic flows in Table 5-20 with those symmetric values in Table 5-1 to test the local stability of that equilibrium. The results of local stability test are in Table 5-23. To determine the stability or otherwise, we followed the definition of the local stability given above. In this case, only the cases with strongly asymmetric demands appear to be locally stable.

O/D	T_{26}					
	v^n_1/v^n_3	200	800	1400	2000	2600
T_{15}	200	200/0	200/0	100/700	100/1000	100/1300
	800		800/0	800/14	686/640	402/1298
	1400			1386/14	702/998	-
	2000				-	-
	2600					-

Table 5-20 The calculated traffic flows by F-W in local stability test 2
(Key: iteration n is shown in Table 5-22)

O/D	T_{26}					
	g^n_1/g^n_3	200	800	1400	2000	2600
T_{15}	200	45/7	45/7	7/45	7/45	7/45
	800		45/7	45/7	27/25	12/40
	1400			45/7	21/31	-
	2000				-	-
	2600					-

Table 5-21 The calculated green times by F-W in local stability test 2
(Key: iteration n is shown in Table 5-22)

O/D	T_{26}					
	n/gap	200	800	1400	2000	2600
T_{15}	200	4/0.0	5/0.0	2/0.0	3/0.0	4/0.0
	800		7/0.0	16/0.0	50/2374	50/1103
	1400			23/0.0	50/1006	-
	2000				-	-
	2600					-

Table 5-22 The iteration number used by F-W in local stability test 2

O/D	T_{26}					
		200	800	1400	2000	2600
T_{15}	200	U	U	S	S	S
	800		U	U	U	S
	1400			U	S	-
	2000				-	-
	2600					-

Table 5-23 The result of local stability 2
(Key: U and S represent stability and unstability whether symmetric equilibrium is obtained in the test respectively)

Test 3. When the perturbation is ε=(-1,+1,+1,-1)

We obtain the calculated traffic flow and green time on link by a diagonalisation of the F-W algorithm shown respectively in Table 5-24 and Table 5-25 using the initial traffic flow $\mathbf{v}^0 = \mathbf{v}^* + (-1,+1,+1,-1)$. The number of iterations required to converge or terminate, and the final gap value are shown in Table 5-26. We compare the calculated equilibrium traffic flows in Table 5-24 by the F-W with those symmetric values in Table 5-1 to test the local stability of that equilibrium. The results of the local stability test are in Table 5-27. To determine the stability or otherwise, we followed the definition of the local stability given above. As in the previous case, only the cases with strongly asymmetric demands appear to be locally stable.

O/D	T_{26}					
	v^n_1/v^n_3	200	800	1400	2000	2600
T_{15}	200	0/200	0/800	100/700	100/1000	100/1300
	800		0/800	0/1386	114/1361	398/1302
	1400			14/1386	698/1002	-
	2000				-	-
	2600					-

Table 5-24 The calculated traffic flows by F-W in local stability test 3
(Key: iteration n is shown in Table 5-26)

O/D	T_{26}					
	g^n_1/g^n_3	200	800	1400	2000	2600
T_{15}	200	7/45	7/45	7/45	7/45	7/45
	800		7/45	7/45	7/45	12/40
	1400			7/45	21/31	-
	2000				-	-
	2600					-

Table 5-25 The calculated green time by F-W in local stability test 3
(Key: iteration n is shown in Table 5-26)

O/D	T_{26}					
	n/gap	200	800	1400	2000	2600
T_{15}	200	3/0.0	5/0.0	2/0.0	5/0.0	4/0.0
	800		7/0.0	16/0.0	50/2374	50/1103
	1400			23/0.0	50/1005	-
	2000				-	-
	2600					-

Table 5-26 The iteration number used by F-W in local stability test 3

O/D	T_{26}					
		200	800	1400	2000	2600
T_{15}	200	U	U	S	S	S
	800		U	U	U	S
	1400			U	S	-
	2000				-	-
	2600					-

Table 5-27 The result of local stability 3
(Key: U and S represent stability and unstability whether symmetric equilibrium is obtained in the test respectively)

These local stability tests show that the symmetric equilibrium solution to this combined signal control and traffic assignment is often locally unstable. However, in the cases of demands (T_{15}, T_{26}) = (200, 1400), (200, 2000), (200, 2600), (800, 2600) and (1400, 2000), the solutions were found to be locally stable even though these cells are not associated with positive definite Jacobian matrix. In the cases of demands (200,200), (800,800) and (1400,1400), different perturbations lead to convergence to different solutions with (g_1, g_3)=(26,26), (45,7) and (7,45). In the case of demand (800, 1400), multiple green times, (19,33), (45,7) and (7,45) are obtained whilst in the demand (800,2000), multiple green times, (15,37), (27,25) and (7,45) are obtained in these local stability tests. These tests indicate that good initial starting values are required to obtain the symmetric equilibrium solutions. They also indicate that caution is required in interpreting the solution to any traffic assignment in a network with traffic responsive signal control as the calculated values might be only one of several possible ones.

5.6 NUMERICAL EXAMPLES

5.6.1 Introduction

In this section, the solution methods developed in chapter 3 under the diagonalisation procedure detailed in chapter 2 are applied to two example networks each of which includes several signal controlled junctions. The results of this help us to compare the performance of these various algorithms when used in networks with this kind of the non-separable cost function. As the first example, the network and travel demand devised by Charlesworth (1977) are used here: this has 5 origins, 5 destinations, 46 nodes and 78 links (for the figures and data of the network, see Appendix 2). The Sioux Falls network (see Figure 3-6) is used as the second example: this has 12 origins,

12 destinations, 23 nodes and 72 links. In each case, Webster's method detailed in section 5.3 is used to obtain green times. As stopping criteria for each algorithm, either a required level of gap value as $G(v^n) \leq \varepsilon$ or a maximum number of iterations are used.

Algorithms 1, 2, 3 and 4, the Frank-Wolfe and the Schittenhelm algorithms have been tested to compare their performances. In particular, Algorithm 2 used one number of the master subproblem of Schittenhelm's algorithm before switching to the master subproblem of Algorithm 1. Algorithms 3 and 4 also used one column generation in the initialisation step. In this test, the weighting factor γ, which is used to combine junction and link delays, is varied from 0.0 to 1.0 in increments of 0.25. Tables 5-28, 29, 30, 31 and 32 show that the final values for each algorithm are compared in terms of the iteration number, CPU time on SUN Sparc station 370 GX and mean gap value in equation (2.16) as the weighting factor γ is varied on the Charlesworth network. In the same way, Tables 5-33, 34, 35, 36 and 37 show the final values of each algorithm on the Sioux Falls network. The performance of each algorithm as a whole is shown in Figures 5-4, 5, 6, 7 and 8 as the weighting factor γ is varied on the Charlesworth network. In the same way, the performance of each algorithm is shown in Figures 5-9, 10, 11, 12 and 13 on the Sioux Falls network.

5.6.2 Charlesworth's network

In Table 5-28, the performance of the algorithms on the Charlesworth network is shown when the weighting factor γ =0.0 which means that junction delay but not link delay is considered in the choice of routes. Algorithm 3 runs fastest to a mean gap value of 12.239 seconds in 17.10 CPU seconds of SUN Sparc station 370 GX. However, Algorithm 4 runs furthest to a mean gap value of 6.429 seconds in 95.52 CPU seconds. Note that each algorithm is stopped by the maximum iteration number of 31. Figure 5-4

shows the whole process of the convergence for each algorithm. The pattern of each algorithm except the Frank-Wolfe algorithm fluctuates slightly in the initial stage and stabilises afterwards. Figure 5-15 compares the final traffic flows obtained by each algorithm with those of Algorithm 3, and Figure 5-16 shows the final green times for this case. These figures show substantial differences between flows and green times on some links and close agreement on others. Note that values from the Frank-Wolfe, Algorithm 4 and the others form three distinct flow groups.

In Table 5-29, the performance of the algorithms is shown when γ =0.25. Algorithm 3 again runs fastest to a mean gap value of 15.223 seconds in 17.08 seconds of CPU time. However, Algorithm 4 runs furthest to a mean gap value of 10.653 seconds in 96.60 seconds of CPU time. Note that each algorithm is stopped by the maximum iteration number. Figure 5-5 shows the whole process of the convergence for each algorithm. The pattern of each algorithm is more stable than in the earlier case. Figure 5-17 compares the final traffic flows obtained by each algorithm with those of Algorithm 3. Figure 5-18 shows the final green times for this case. Note that the pattern of values is more concentrated than the case when γ =0 as shown in Figures 5-15 and 5-16. These figures show substantial differences between algorithms in terms of flows and green times on some links and close agreement on others.

In Table 5-30, the performance of the algorithms is shown when γ =0.5. Algorithm 3 again runs fastest to a mean gap value of 11.996 seconds in 17.24 seconds of CPU time. However, Algorithm 2 runs furthest to a mean gap value of 0.783 seconds in 96.14 seconds of CPU time. Figure 5-6 shows the whole process of the convergence for each algorithm. The pattern of each algorithm is similar to the earlier cases. However, Algorithm 2 undergoes a sudden reduction in gap in the initial stage, then stabilises for a period before being reduced suddenly again. Figure 5-19 compares the final traffic flows

obtained by each algorithm with those of Algorithm 3, and Figure 5-20 shows the final green times for this case. These figures show substantial differences between the new algorithms and the Frank-Wolfe in terms of flows and green times on some links and close agreement on others.

In Table 5-31, the performance of the algorithms is shown when the weighting factor γ =0.75. Algorithm 3 again runs fastest to a mean gap value of 0.923 seconds in 16.91 seconds of CPU time. However, Algorithm 1 runs furthest to a mean gap value of 0.562 seconds in 93.22 seconds of CPU time. Figure 5-7 shows the whole process of the convergence in each algorithm. The pattern of each algorithm except the Frank-Wolfe algorithm fluctuates more than previous cases in the initial stages whilst the gap values reduce to about 1.0 second. Figure 5-21 compares the final traffic flows obtained by each algorithm with those of Algorithm 3. Figure 5-22 shows the final green times for this case. These figures show that differences between algorithms in terms of flows and green times are less than the earlier cases, and the final values are more on the diagonal line.

In Table 5-32, the performance of the algorithms on the Charlesworth network is shown when the weighting factor γ =1.0 which means that only link cost is considered in the choice of routes. Algorithm 4 converges fastest to a mean gap value of 0.000 seconds in 10.93 seconds of CPU time. Algorithms 1 and 2 also converge to a mean gap value of 0.000 seconds in 22.27 and 14.83 seconds of CPU time respectively. Figure 5-8 shows the whole process of the convergence for each algorithm. The pattern of each algorithm except the Frank-Wolfe algorithm fluctuates whilst converging to a smaller mean gap value near zero. Figure 5-23 compares the final traffic flows obtained by each algorithm with those of Algorithm 3. Figure 5-24 shows the final green times for this case. These figures show that the final values differ only slightly with the exception of the F-W

algorithm. This occurs because in this case of the separable cost function, the solution is unique and all the new algorithms converge towards it.

In Figure 5-35, final gap function values of each algorithm are depicted on the Charlesworth network as the weighting factor γ is changed. An expectation of the effects of γ on a mean gap function is to reduce a final gap value as γ is moved to 1.0 which means only link cost. All algorithms reduces the final gap value steadily as γ increases towards 1.0 except for the weighing factor γ =0.25. In terms of performance of algorithm, Algorithm 3 runs quickly all the cases. All the algorithms except the Frank-Wolfe converge or terminate at similar assignments.

5.6.3 Sioux Falls network

In Table 5-33, the performance of the algorithms on the Sioux Falls network is shown when the weighting factor γ =0.0. Algorithm 3 runs fastest to a mean gap value of 0.194 seconds in 3.63 CPU seconds. However, Algorithm 4 runs furthest to a mean gap value of 0.096 seconds in 37.96 CPU seconds. Note that each algorithm is stopped by the maximum iteration number of 31. Figure 5-9 shows the whole process of the convergence for each algorithm. The pattern of each algorithm fluctuates slightly in the initial stage and then stabilises afterwards. Figure 5-25 compares the final traffic flows obtained by each algorithm with those of Algorithm 3, and Figure 5-26 shows the final green times for this case. These figures show substantial differences between algorithms in terms of flows and green times on some links and close agreement on others. In particular, distinct flow groups are calculated by the Frank-Wolfe, Algorithm 4 and the others.

In Table 5-34, the performance of the algorithms is shown when γ =0.25. Algorithm 3 again runs fastest to a mean gap value of 0.234 seconds in 3.60 CPU seconds. However, Algorithm 4 runs furthest to a mean gap value of 0.196 seconds in 36.51 CPU seconds. Figure 5-10 shows the whole process of the convergence in each algorithm. The pattern of each algorithm is more stable than the earlier one. Note that it is all over by 2 seconds of CPU time. Figure 5-27 compares the final traffic flows obtained by each algorithm with those of Algorithm 3, and Figure 5-28 shows the final green times for this case. Note that when the other algorithms have zero link flows, the Frank-Wolfe ranges from 0 to 600. These figures show substantial differences between the new algorithms and the Frank-Wolfe in terms of flows and green times on some links and close agreement on others.

In Table 5-35, the performance of the algorithms is shown when γ =0.5. Algorithm 3 runs fastest to a mean gap value of 0.239 seconds in 3.66 CPU seconds. However, Algorithm 4 runs furthest to a mean gap value of 0.049 seconds in 36.82 CPU seconds. Figure 5-11 shows the whole process of the convergence in each algorithm. The pattern of each algorithm is similar to the previous cases. Note that Algorithm 4 outperforms the others with the mean gap value of 0.049 seconds, and the Frank-Wolfe algorithm also performs well with the mean gap value of 0.192 seconds. Figure 5-29 compares the final traffic flows obtained by each algorithm with those of Algorithm 3. In this figure, the final traffic flows by Algorithms 1, 2, 3 and the Schittenhelm are close agreements on most links whilst those by Algorithm 4 and the Frank-Wolfe are different on some links. Figure 5-30 shows the final green times for this case. These figures show that differences between algorithms are less in flows than green times on some links.

In Table 5-36, the performance of the algorithms is shown when γ =0.75. Algorithm 3 runs fastest to a mean gap value of 0.075 seconds in 3.68 CPU seconds.

However, Algorithm 4 runs furthest to a mean gap value of 0.005 seconds in 36.81 CPU seconds. In this case again, the Frank-Wolfe algorithm performs well. Figure 5-12 shows the whole process of the convergence in each algorithm. The pattern of each algorithm is similar to the previous cases. Note that Algorithm 3, the Frank-Wolfe and the others converge to separate degrees. Figure 5-31 compares the final traffic flows obtained by each algorithm with those of Algorithm 3. Figure 5-32 shows the final green times for this case. These figures still show substantial differences between the new algorithms except Algorithm 4 and the Frank-Wolfe in terms of flows and green times on some links and close agreement on others.

In Table 5-37, the performance of the algorithms on the Sioux-Fall network is shown when the weighting factor $\gamma = 1.0$. Algorithm 3 converges fastest to a mean gap value of 0.000 seconds in 4.01 CPU seconds. Algorithms 1, 2 and 4, and Schittenhelm algorithm converge also to a mean gap value of 0.000 seconds in 8.42, 13.98, 9.19 and 4.16 CPU seconds respectively. Note that Algorithms 1, 2, 4 are stopped by the convergence criterion of gap value whilst Algorithm 3, the Schittenhelm and the Frank-Wolfe algorithms are stopped by the maximum iteration number. Figure 5-13 shows the whole process of the convergence for each algorithm. The pattern of each algorithm except the Frank-Wolfe algorithm fluctuates whilst it converges to zero gap value. Figure 5-33 compares the final traffic flows obtained by each algorithm with those of Algorithm 3. Figure 5-34 shows the final green times for this case. These figures show that the final values differ slightly, as a mean gap value converges to zero, and as the equilibrium is unique due to separability.

In Figure 5-36, final gap function values of each algorithm are depicted as the weighting factor γ is changed on the Sioux Falls network. An expectation of effects of γ on a mean gap function is to reduce the final gap value as γ approaches 1.0 which means

only a link based cost. All the algorithms reduce the final gap value as γ is changed to 1.0 although there is an increase of the value at 0.25 and 0.5. In terms of performance of algorithm, Algorithm 3 runs quickly all the cases. All the algorithms except the Frank-Wolfe converge or terminate at similar assignments. However, when γ = 0.5 and 0.75, the Frank-Wolfe algorithm outperforms some other algorithms.

5.6.4 Discussion and summary

In these examples, the new algorithms run quickly and reach a good level of the convergence in relatively few iterations. For example, in the case of γ = 0.5 in Charlesworth's network, the gap is reduced to below 12 seconds by each of the new algorithms during iterations 1 to 4, and further iterations serve only to make marginal further reductions. Furthermore, final traffic flows and green times calculated by the new algorithms agree closely to show additional confirmation for the convergence.

Nonetheless, in some cases, the new algorithms cannot improve their gap values with further iteration and terminate with non-zero gap values. In particular, the gap values become constant at low but non-zero values after just few seconds of CPU time on the Sioux Falls network. Each of the new algorithms terminates with a gap value that is substantially smaller than that achieved by the Frank-Wolfe algorithm, although the Frank-Wolfe algorithm outperforms some other algorithms in the cases of γ = 0.5 and 0.75 on the Sioux Falls network.

There are three distinct groups of CPU time usage by the various algorithms. Algorithms 1, 2 and 4 use the greatest amount of CPU time typically about 95 seconds for Charlesworth's network and 36 seconds for the Sioux Falls network, the Frank-Wolfe algorithm uses a medium amount of about 28 seconds for Charlesworth's and 11 seconds

for the Sioux Falls network, and Algorithm 3 and Schittenhelm's algorithm use least about 17 seconds for Charlesworth's and 3 seconds for the Sioux Falls network. The usage of CPU time is closely related with the behaviour of each algorithm described in chapter 3. In accordance with this behaviour, the levels of convergence for each algorithm are ordered: the smallest gap values achieved by Algorithms 1, 2 and 4, and a medium gap value by Algorithm 3 and Schittenhelm and the largest gap value by the F-W algorithm. In summary, Algorithm 3 performs best in all the cases in terms of CPU time. However, Algorithm 4 terminates with a consistently small mean gap value, though each of Algorithms 1 and 2 has a slightly smaller gap in some cases. In terms of both measures of CPU time and the gap value, Algorithms 3 and 4 perform well in most cases.

The convergence pattern of the new algorithms fluctuates. The pattern fluctuates more as the weighting value, γ, approaches zero, which gives the maximum junction effect. Because the weighting factor in the cost function represents the degree of junction effects, it is indicative of the degree to which junction interactions are influential. As the junction effect increases, the non-separability increases so that the final gap value is higher. On the other hand, when there is no junction effect, the cost function becomes separable and the final gap value is closer to zero. In particular, when the weighting value is unity which is the case of link cost only, the gap value of each new algorithm converges to zero.

The pattern of convergence in the Charlesworth network shows more fluctuations than in the Sioux Falls one. This can be explained according to the characteristics and structure of each network. Charlesworth's network has two main roads that provide alternative routes: choice between them can then cause gap values to fluctuate. On the other hand, the Sioux Fall network has a grid form so that many route

choices are possible and thus the demand is spread more widely and the fluctuation reduces.

	Iteration	CPU(Second)	Mean gap (Second)
Algorithm 1	31	95.43	10.953
Algorithm 2	31	94.36	9.529
Algorithm 3	31	17.10	12.239
Algorithm 4	31	95.52	6.429
Frank-Wolfe	50	28.23	4988.523
Schittenhelm	31	17.41	12.209

Table 5-28 Performance of algorithms in Charlesworth's network when γ=0.0 (Maximum junction effect)

	Iteration	CPU(Second)	Mean gap (Second)
Algorithm 1	31	96.30	12.215
Algorithm 2	31	93.80	11.709
Algorithm 3	31	17.08	15.223
Algorithm 4	31	96.60	10.653
Frank-Wolfe	50	28.37	3713.087
Schittenhelm	31	17.26	14.997

Table 5-29 Performance of algorithms in Charlesworth's network when γ=0.25

	Iteration	CPU(Second)	Mean gap (Second)
Algorithm 1	31	95.92	4.387
Algorithm 2	31	96.14	0.783
Algorithm 3	31	17.24	11.996
Algorithm 4	31	96.96	4.674
Frank-Wolfe	50	28.53	2444.345
Schittenhelm	31	17.56	10.965

Table 5-30 Performance of algorithms in Charlesworth's network when γ=0.5

	Iteration	CPU(Second)	Mean gap (Second)
Algorithm 1	31	93.22	0.562
Algorithm 2	31	90.95	0.865
Algorithm 3	31	16.91	0.923
Algorithm 4	31	94.46	1.222
Frank-Wolfe	50	28.18	1181.512
Schittenhelm	31	17.07	0.928

Table 5-31 Performance of algorithms in Charlesworth's network when γ=0.75

	Iteration	CPU(Second)	Mean gap (Second)
Algorithm 1	17	22.27	0.000
Algorithm 2	14	14.83	0.000
Algorithm 3	31	16.54	0.001
Algorithm 4	12	10.93	0.000
Frank-Wolfe	50	27.49	2.320
Schittenhelm	31	16.79	0.001

Table 5-32 Performance of algorithms in Charlesworth's network when γ=1.0 (No junction effect)

	Iteration	CPU(Second)	Mean gap (Second)
Algorithm 1	31	36.22	0.146
Algorithm 2	31	36.09	0.191
Algorithm 3	31	3.63	0.194
Algorithm 4	31	37.96	0.096
Frank-Wolfe	50	11.91	0.594
Schittenhelm	31	3.80	0.193

Table 5-33 Performance of algorithms in Sioux Falls network when γ=0.0 (Maximum junction effect)

	Iteration	CPU(Second)	Mean gap (Second)
Algorithm 1	31	35.87	0.205
Algorithm 2	31	35.9	0.234
Algorithm 3	31	3.60	0.234
Algorithm 4	31	36.51	0.196
Frank-Wolfe	50	11.88	0.393
Schittenhelm	31	3.77	0.234

Table 5-34 Performance of algorithms in Sioux Falls network when γ=0.25

	Iteration	CPU(Second)	Mean gap (Second)
Algorithm 1	31	36.28	0.238
Algorithm 2	31	36.18	0.238
Algorithm 3	31	3.66	0.239
Algorithm 4	31	36.82	0.049
Frank-Wolfe	50	11.92	0.192
Schittenhelm	31	3.82	0.238

Table 5-35 Performance of algorithms in Sioux Falls network when γ=0.5

	Iteration	CPU(Second)	Mean gap (Second)
Algorithm 1	31	36.38	0.077
Algorithm 2	31	36.26	0.074
Algorithm 3	31	3.68	0.075
Algorithm 4	31	36.81	0.005
Frank-Wolfe	50	11.69	0.029
Schittenhelm	31	3.75	0.075

Table 5-36 Performance of algorithms in Sioux Falls network when γ=0.75

	Iteration	CPU(Second)	Mean gap (Second)
Algorithm 1	16	8.42	0.000
Algorithm 2	20	13.98	0.000
Algorithm 3	31	4.01	0.000
Algorithm 4	17	9.19	0.000
Frank-Wolfe	50	11.58	0.012
Schittenhelm	31	4.16	0.000

Table 5-37 Performance of algorithms in Sioux Falls network when γ=1.0 (No junction effect)

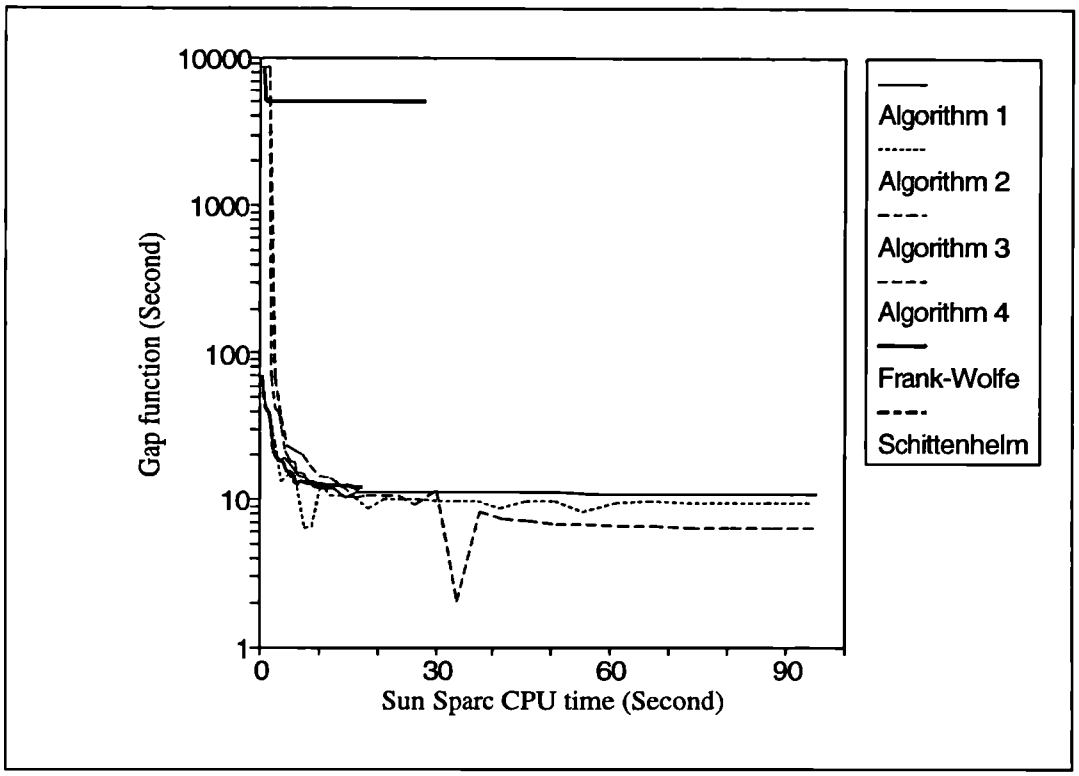


Figure 5-4 The performance of algorithms in Charlesworth's network when γ =0.0 (Maximum junction effect)

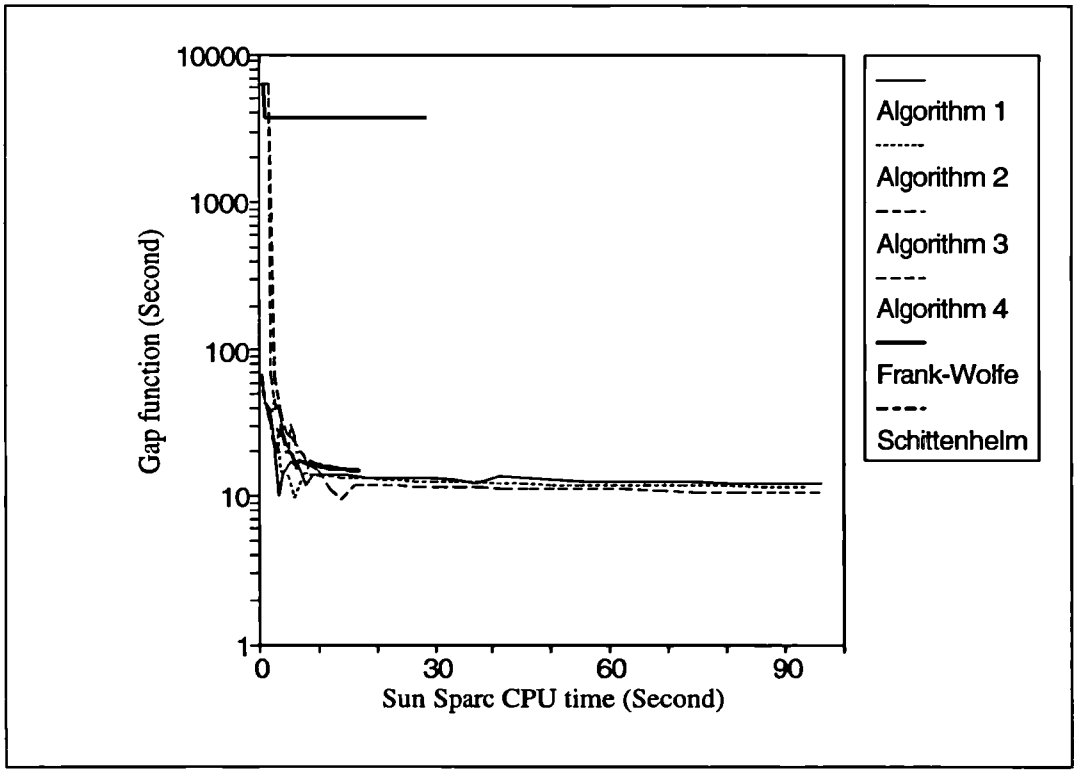


Figure 5-5 The performance of algorithms in Charlesworth's network when γ =0.25

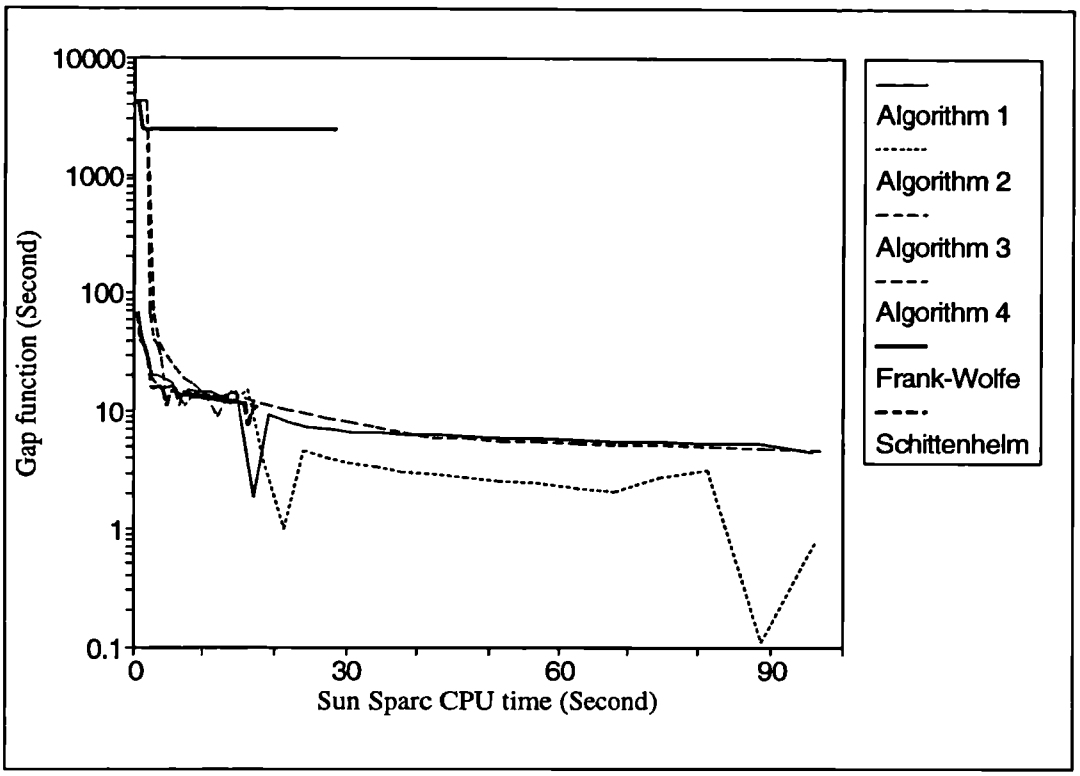


Figure 5-6 The performance of algorithms in Charlesworth's network when γ =0.5

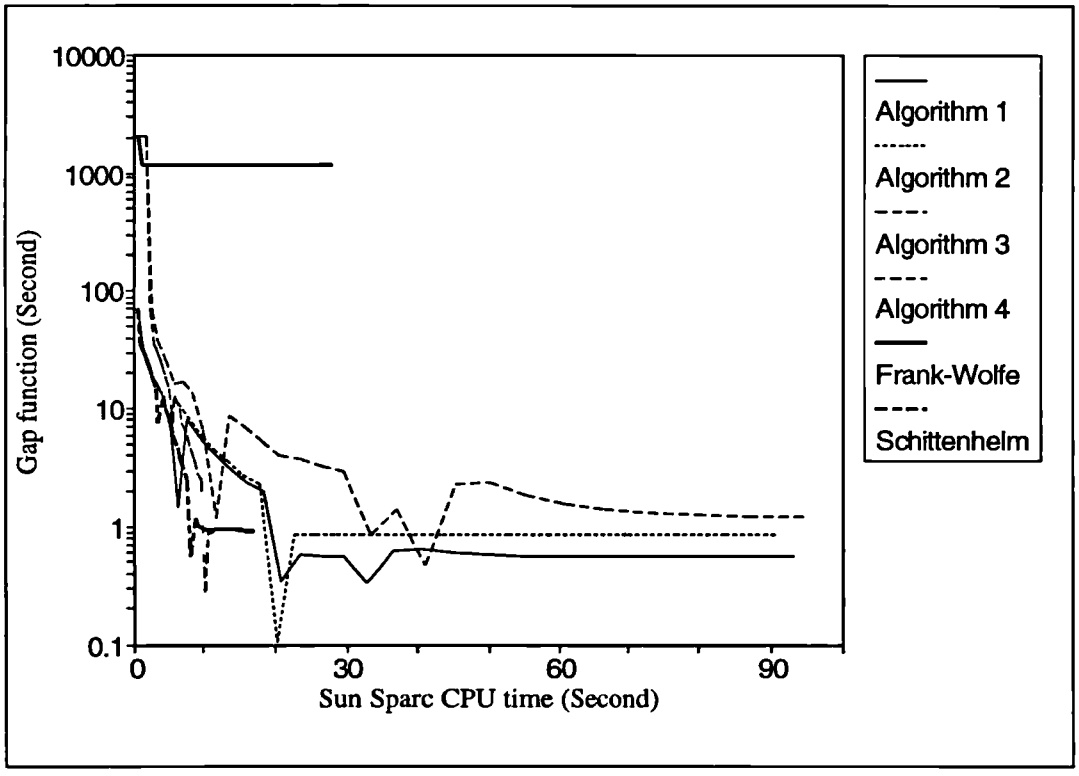


Figure 5-7 The performance of algorithms in Charlesworth's network when γ =0.75

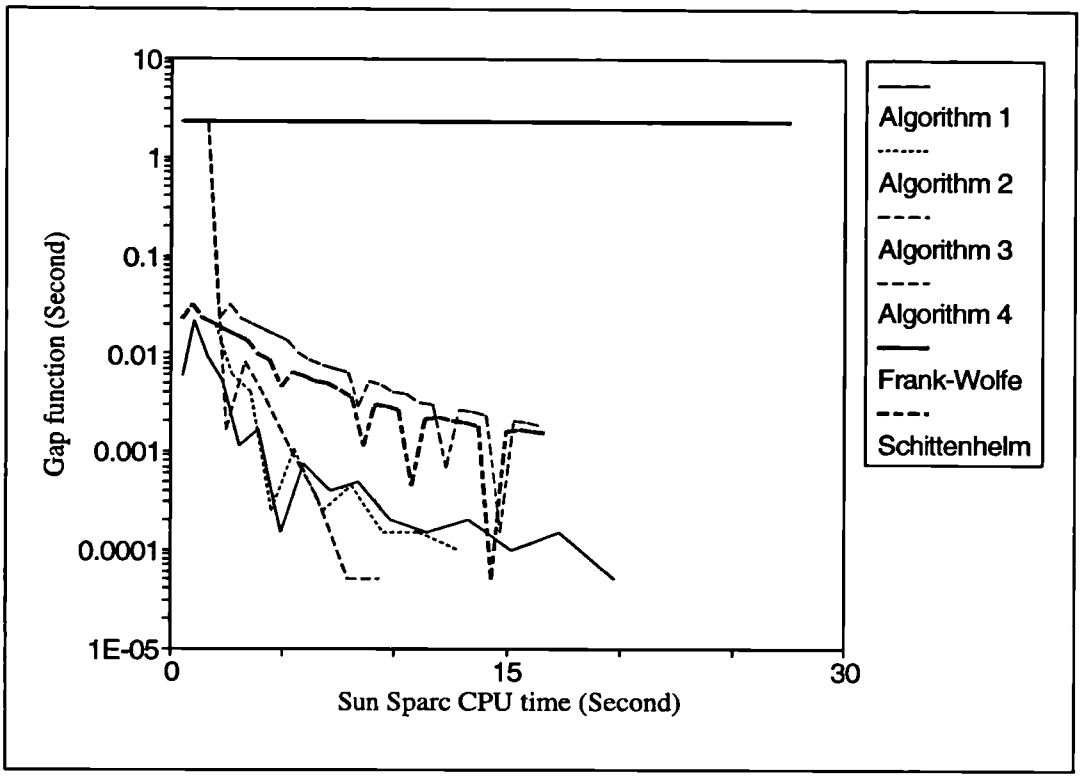


Figure 5-8 The performance of algorithms in Charlesworth's network when $\gamma = 1.0$ (No junction effect)

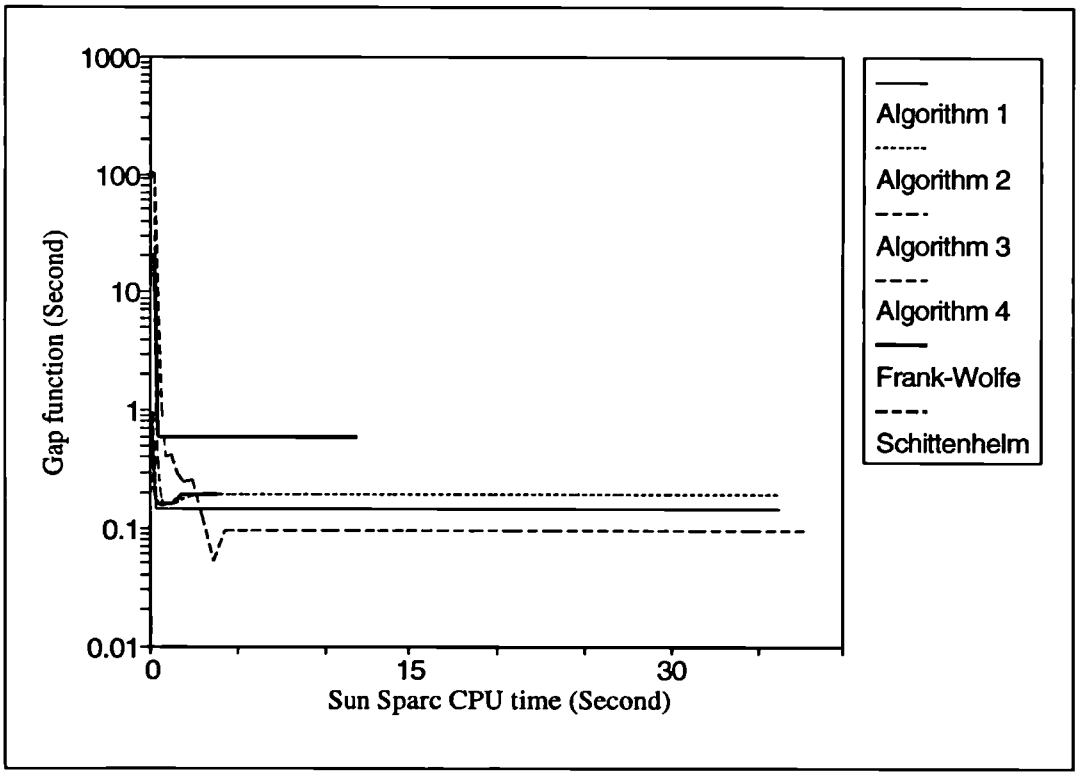


Figure 5-9 The performance of algorithms in Sioux Falls network when $\gamma = 0.0$ (Maximum junction effect)

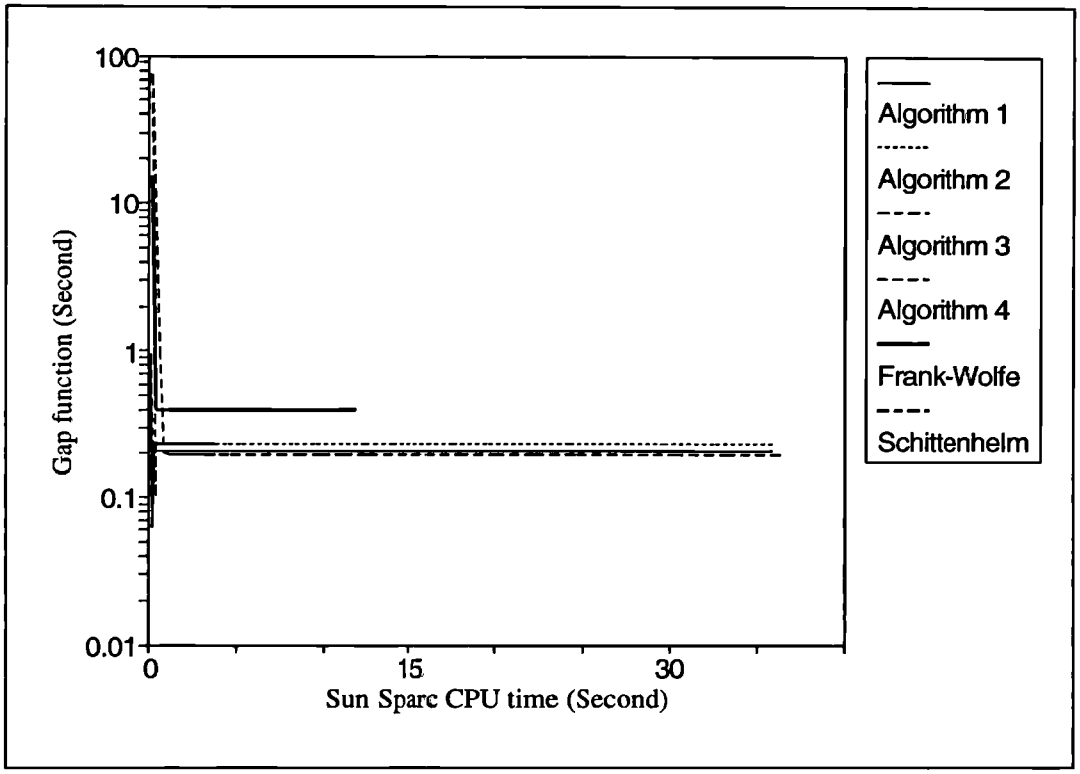


Figure 5-10 The performance of algorithms in Sioux Falls network when γ =0.25

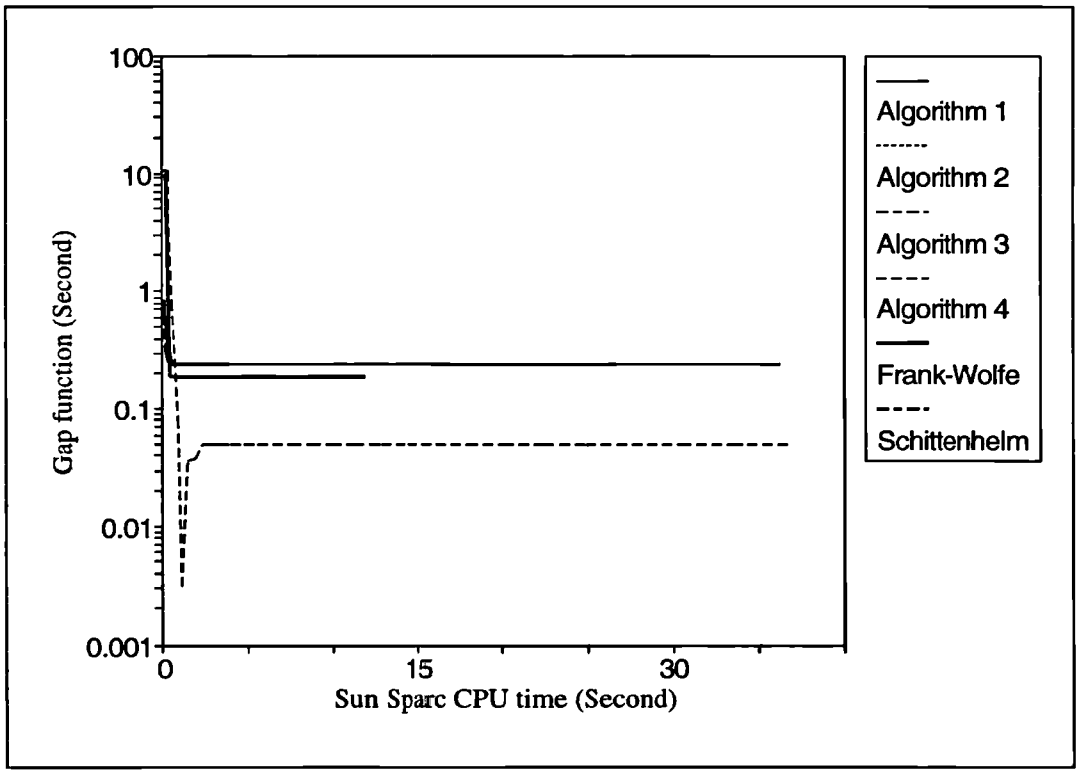


Figure 5-11 The performance of algorithms in Sioux Falls network when γ =0.5

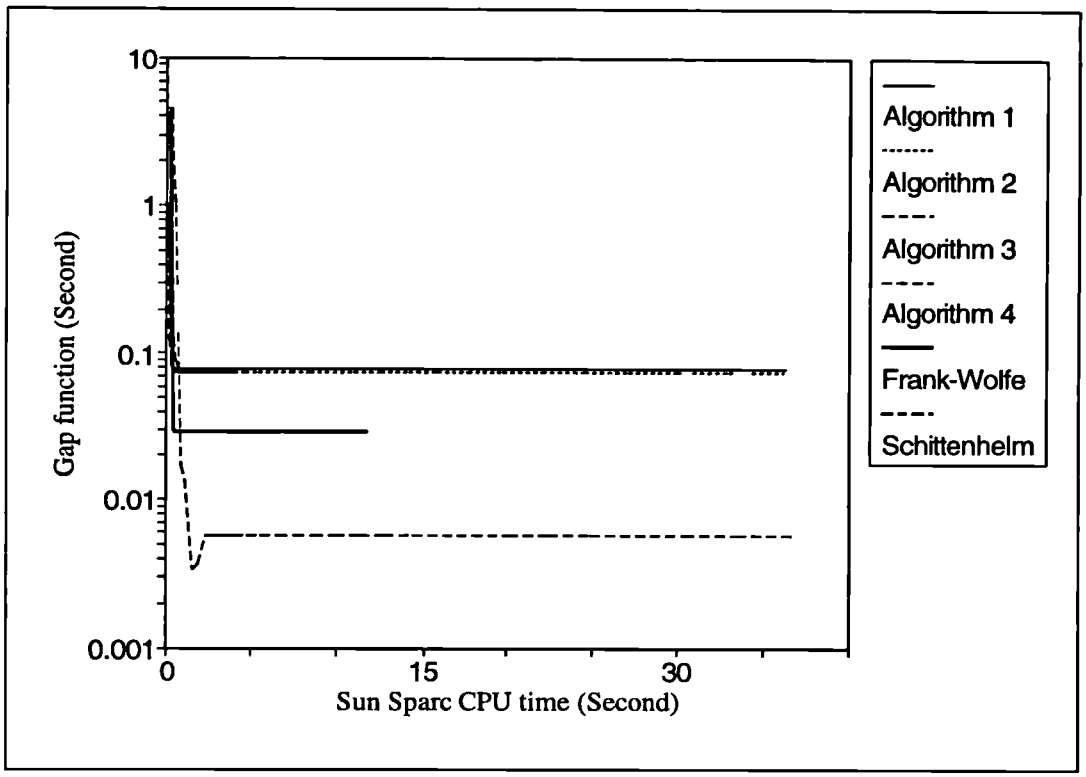


Figure 5-12 The performance of algorithms in Sioux Falls network when $\gamma = 0.75$

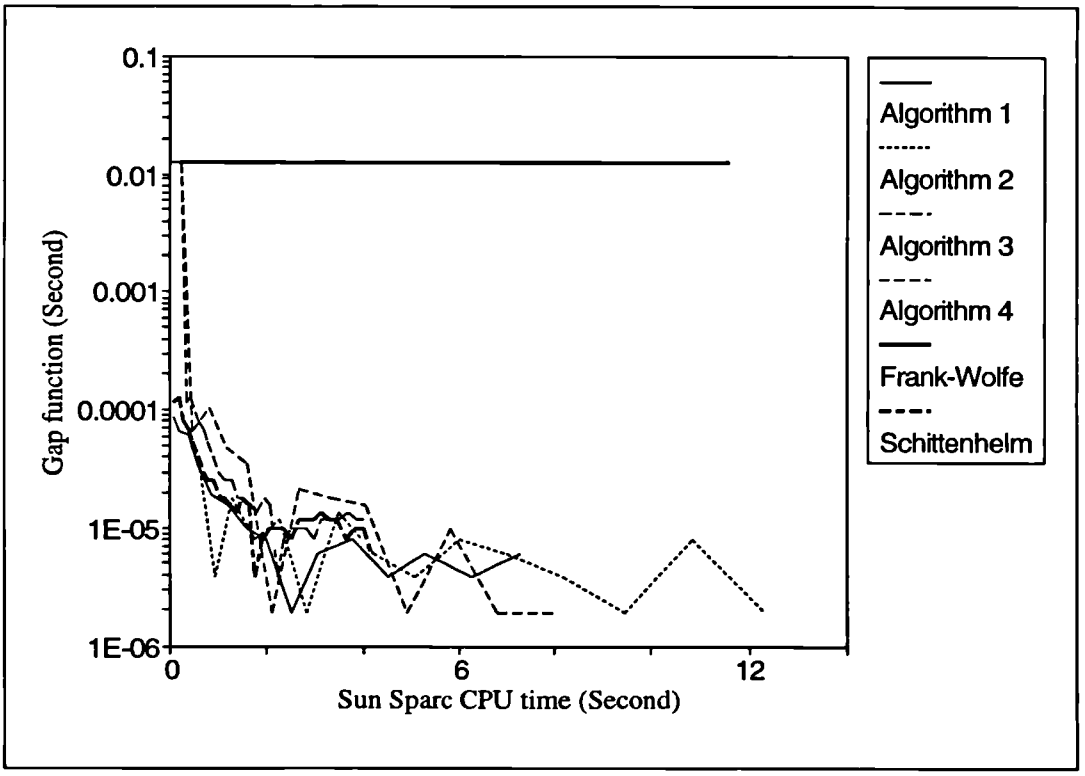


Figure 5-13 The performance of algorithms in Sioux Falls network when $\gamma = 1.0$ (No junction effect)

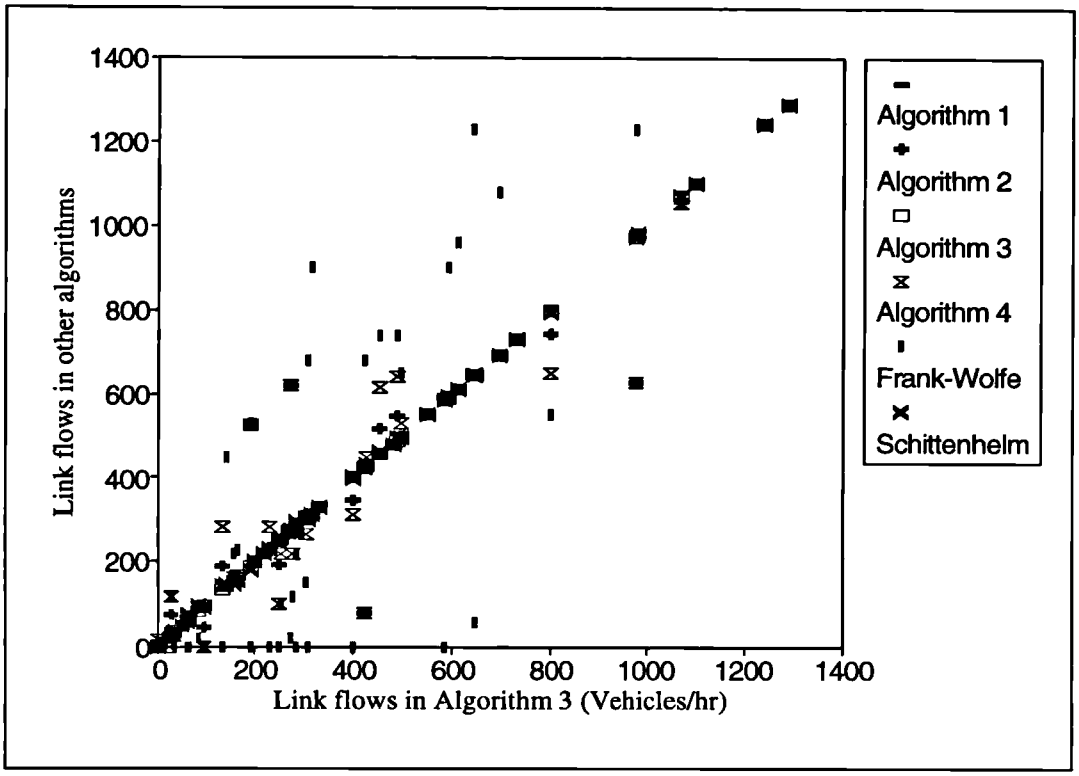


Figure 5-15 The comparison of traffic flows obtained by algorithms on Charlesworth's network when γ =0.0 (Maximum junction effect)

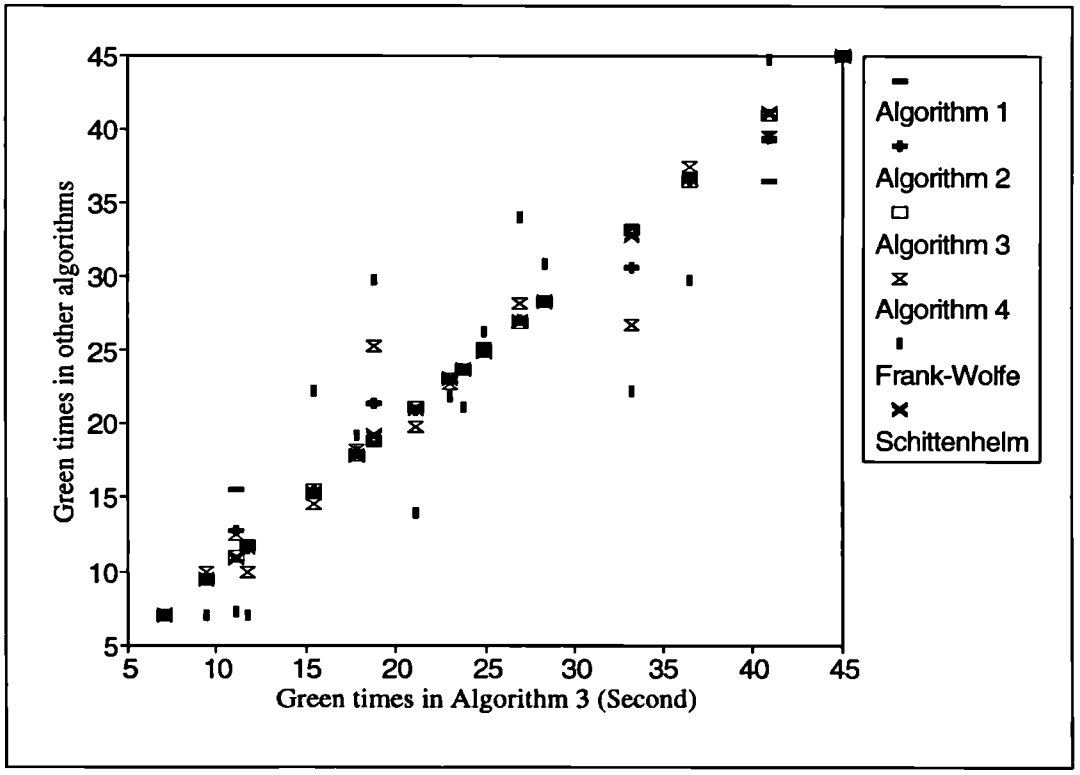


Figure 5-16 The comparison of green times obtained by algorithms on Charlesworth's network when γ =0.0 (Maximum junction effect)

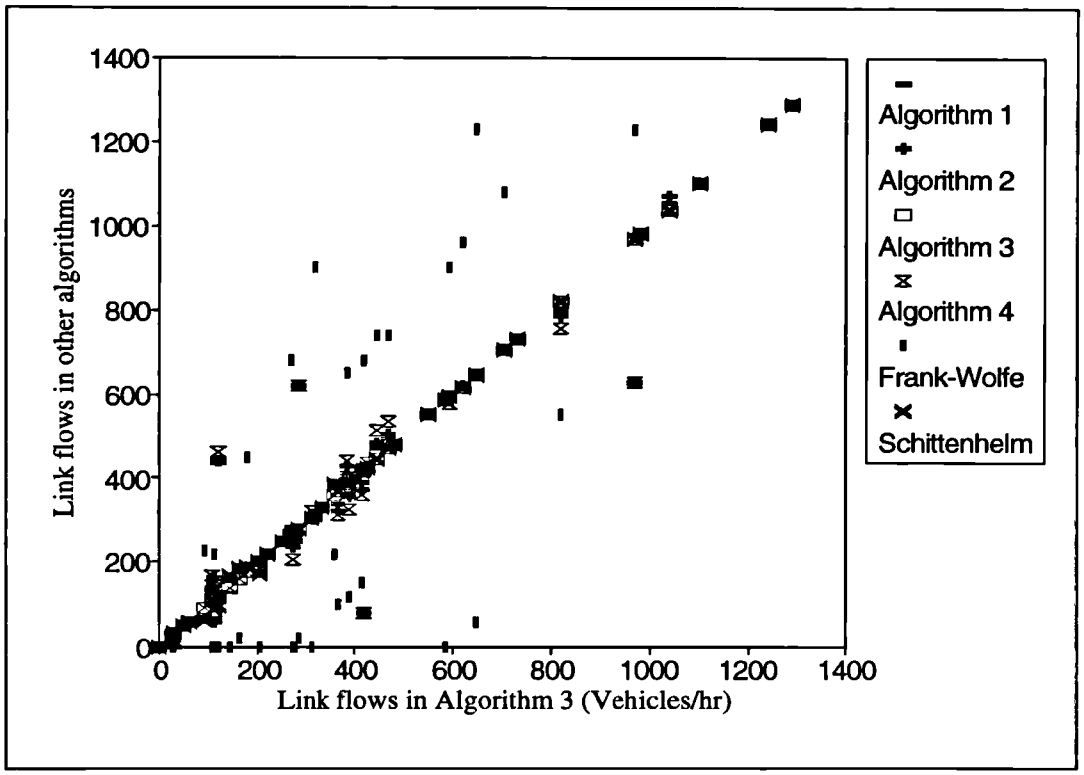


Figure 5-17 The comparison of traffic flows obtained by algorithms on Charlesworth's network when γ =0.25

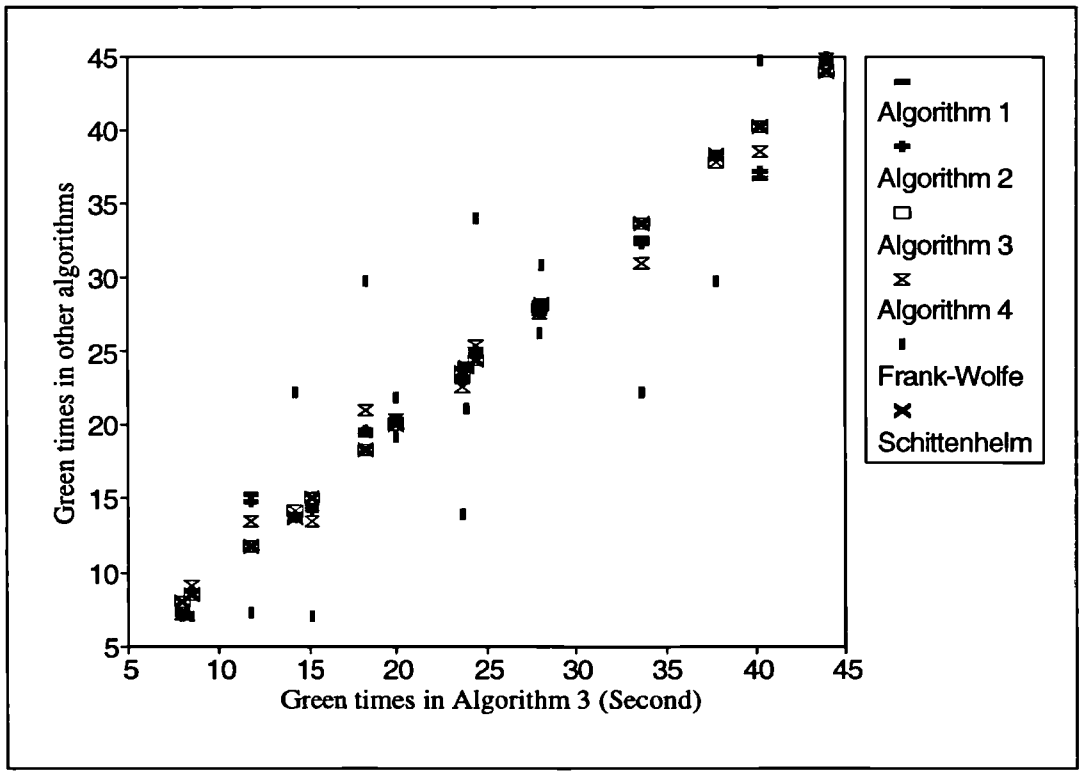


Figure 5-18 The comparison of green times obtained by algorithms on Charlesworth's network when γ =0.25

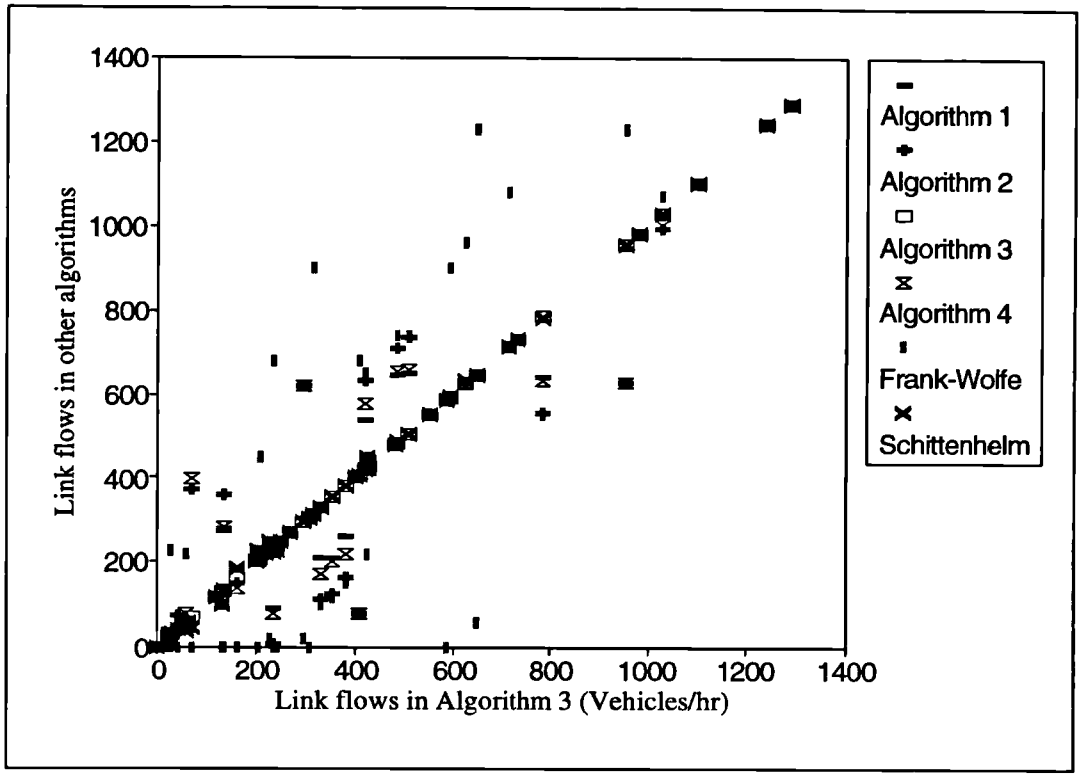


Figure 5-19 The comparison of traffic flows obtained by algorithms on Charlesworth's network when γ =0.5

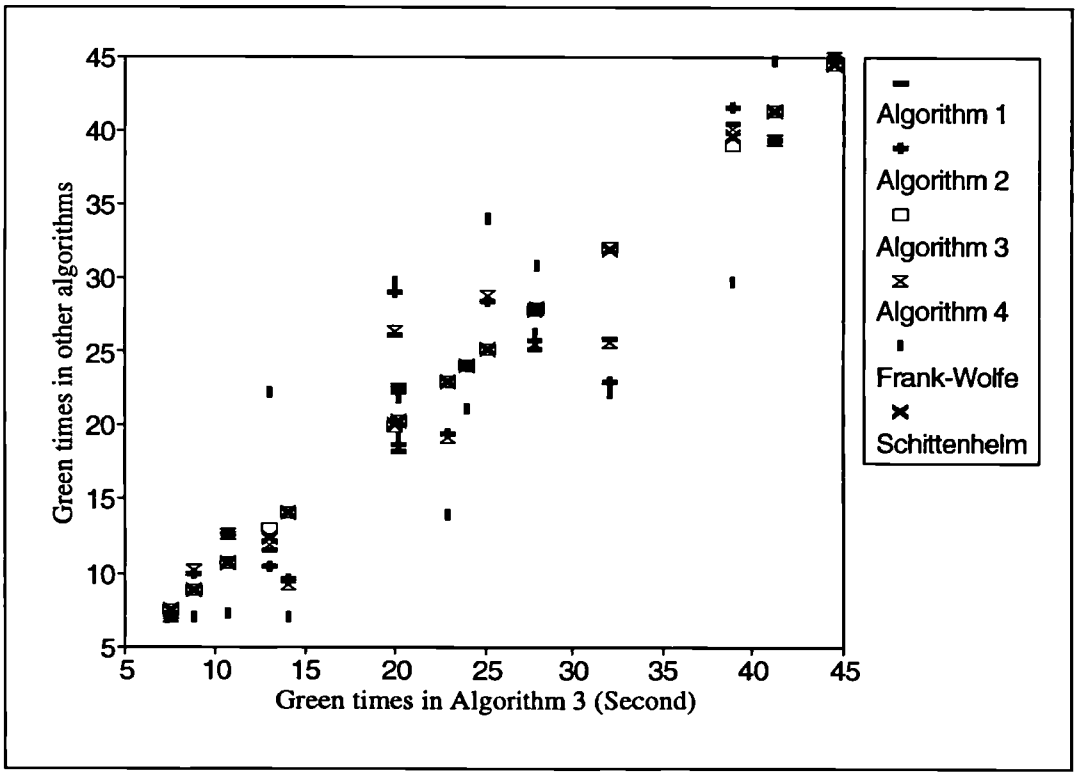


Figure 5-20 The comparison of green times obtained by algorithms on Charlesworth's network when γ =0.5

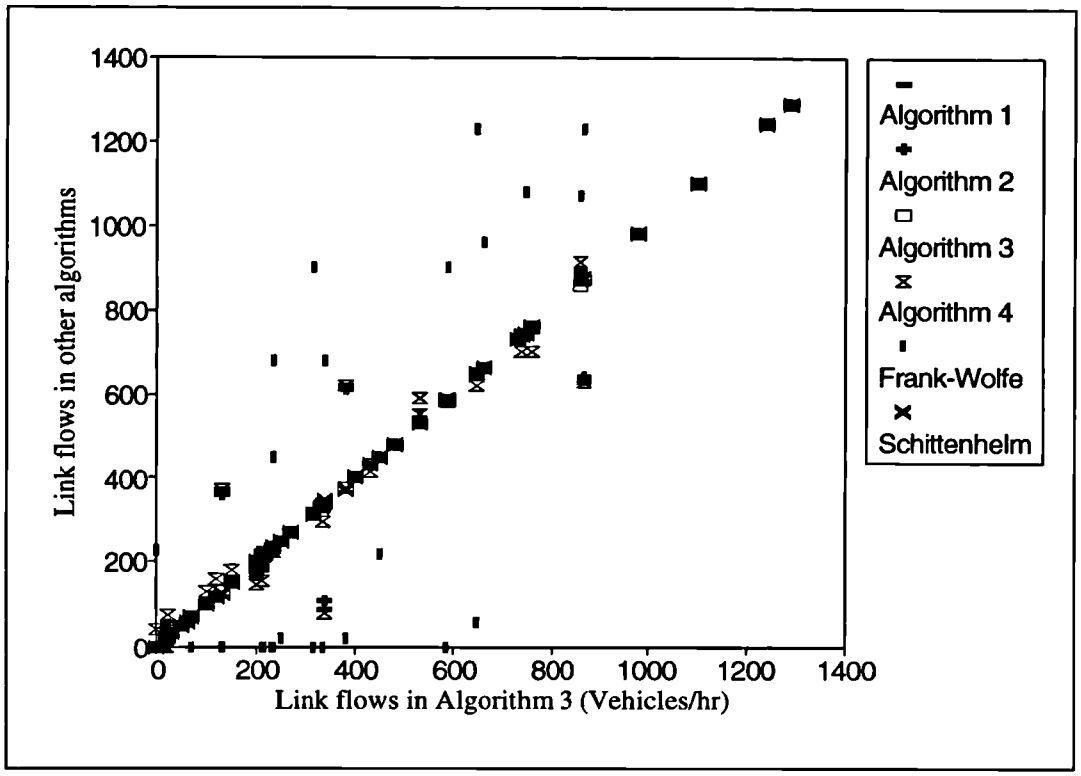


Figure 5-21 The comparison of traffic flows obtained by algorithms on Charlesworth's network when γ =0.75

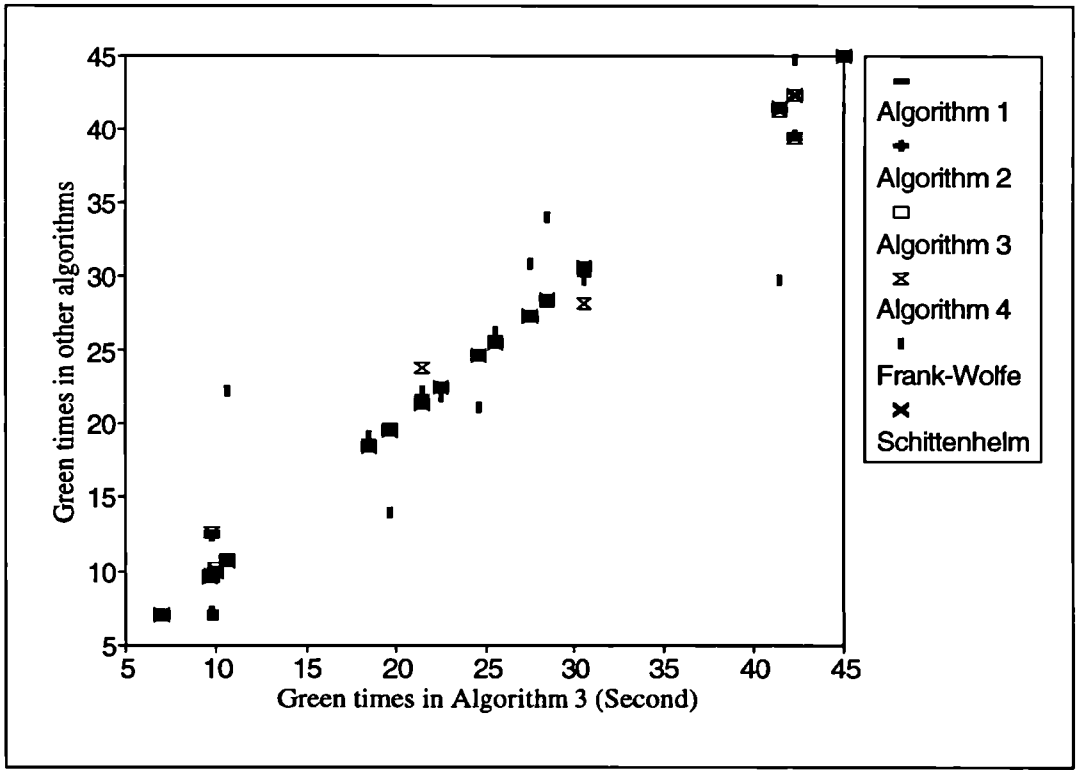


Figure 5-22 The comparison of green times obtained by algorithms on Charlesworth's network when γ =0.75

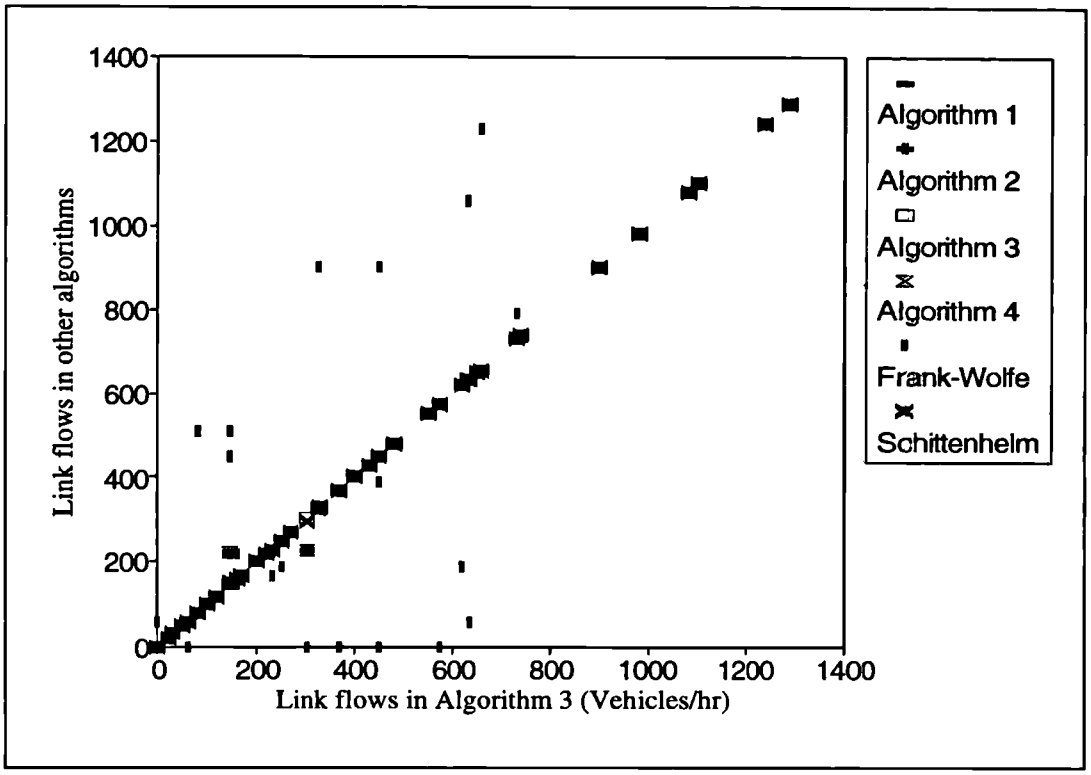


Figure 5-23 The comparison of traffic flows obtained by algorithms on Charlesworth's network when $\gamma = 1.0$ (No junction effect)

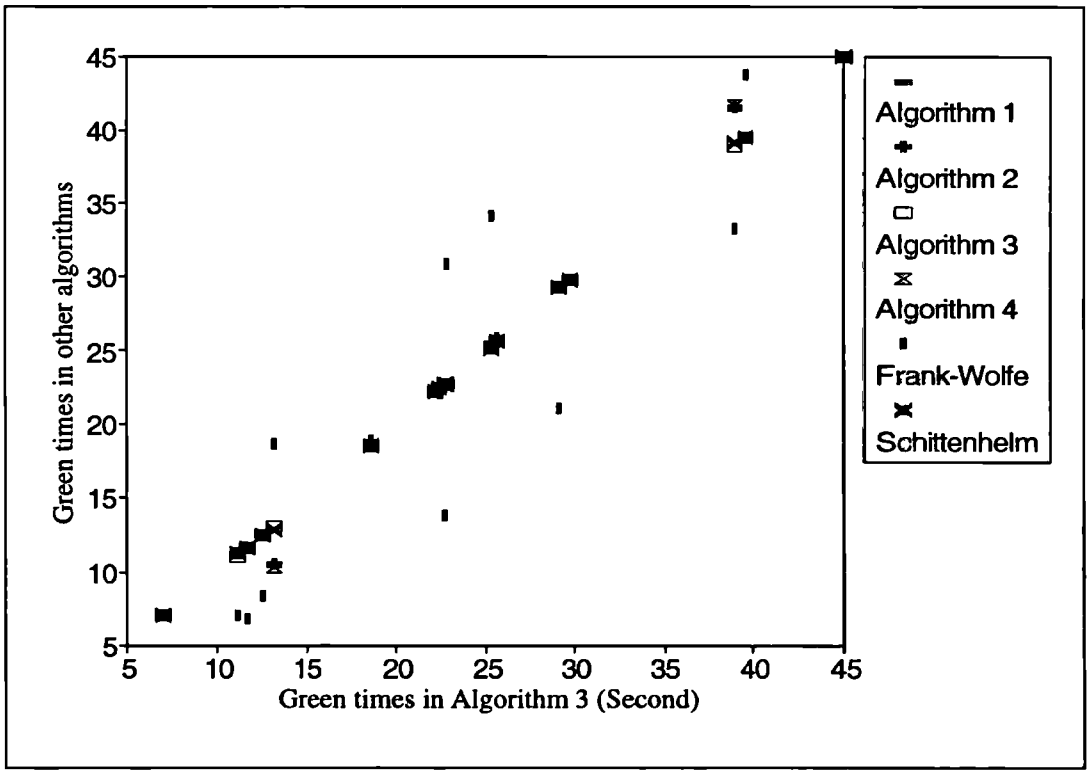


Figure 5-24 The comparison of green times obtained by algorithms on Charlesworth's network when $\gamma = 1.0$ (No junction effect)

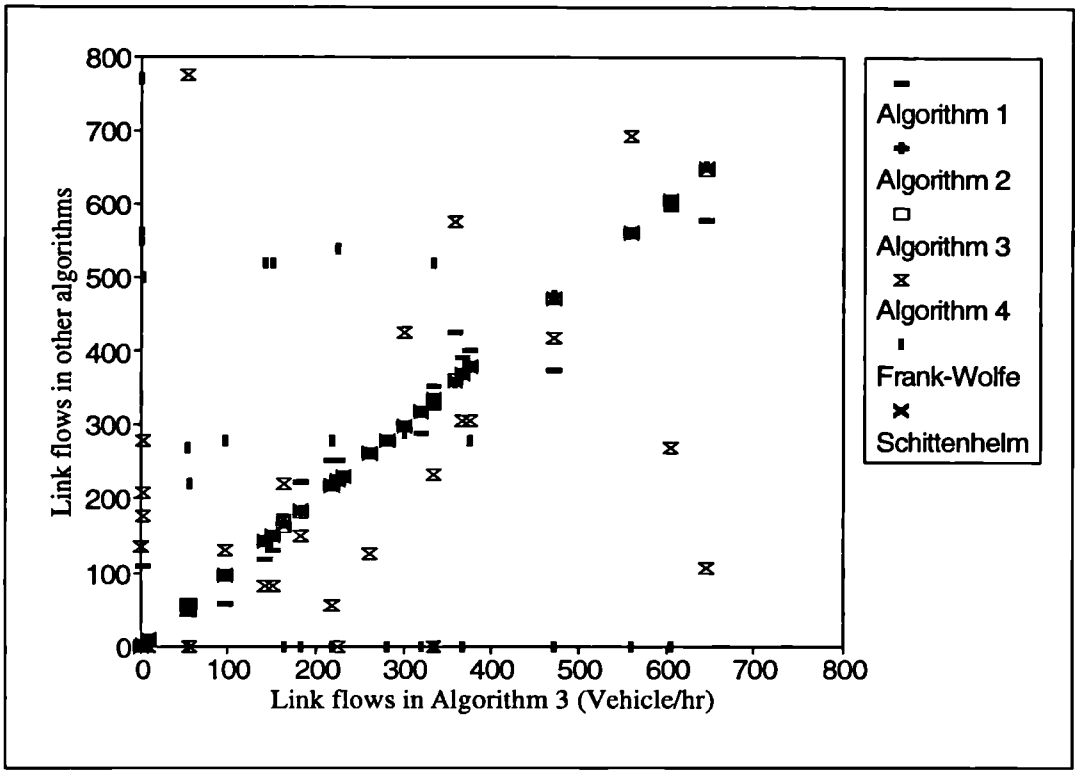


Figure 5-25 The comparison of traffic flows obtained by algorithms on Sioux Falls network when γ =0.0 (Maximum junction effect)

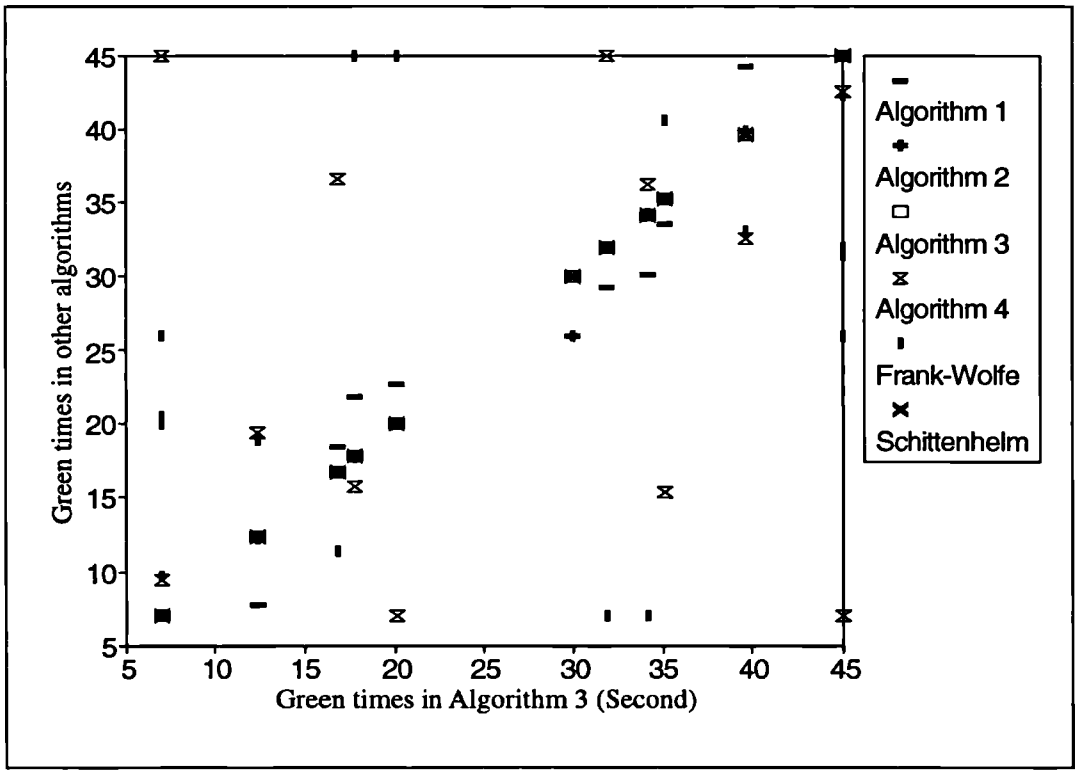


Figure 5-26 The comparison of green times obtained by algorithms on Sioux Falls network when γ =0.0 (Maximum junction effect)

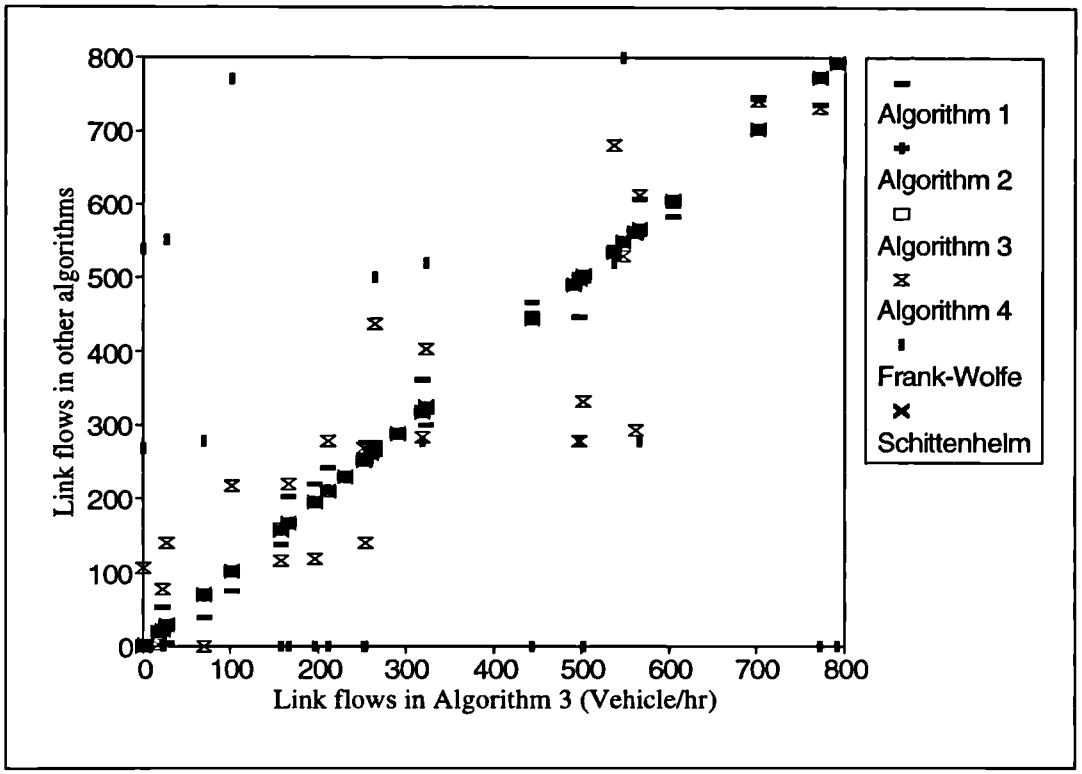


Figure 5-27 The comparison of traffic flows obtained by algorithms on Sioux Falls network when γ =0.25

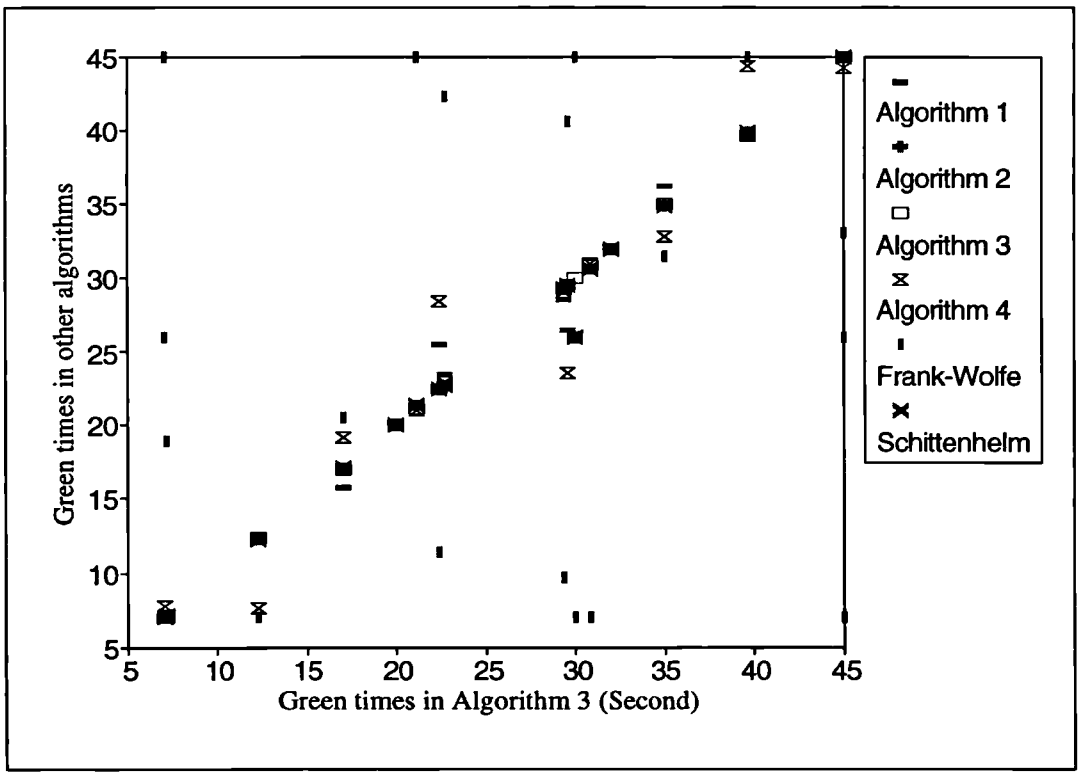


Figure 5-28 The comparison of green times obtained by algorithms on Sioux Falls network when γ =0.25

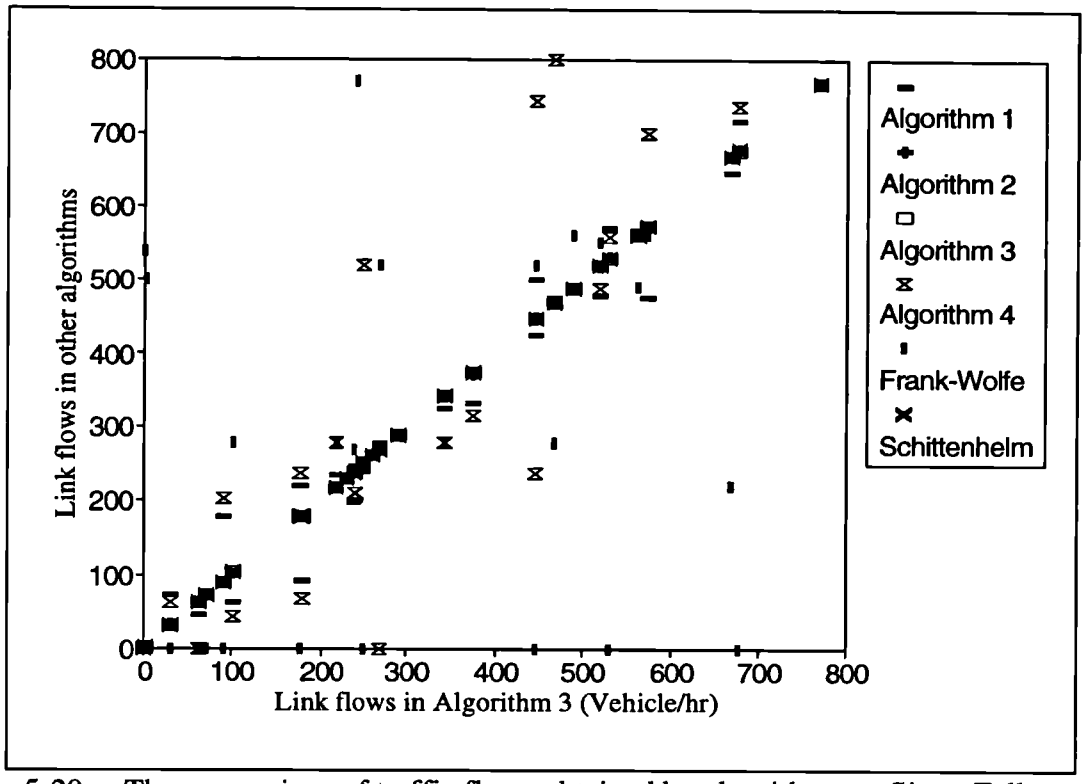


Figure 5-29 The comparison of traffic flows obtained by algorithms on Sioux Falls network when γ =0.5

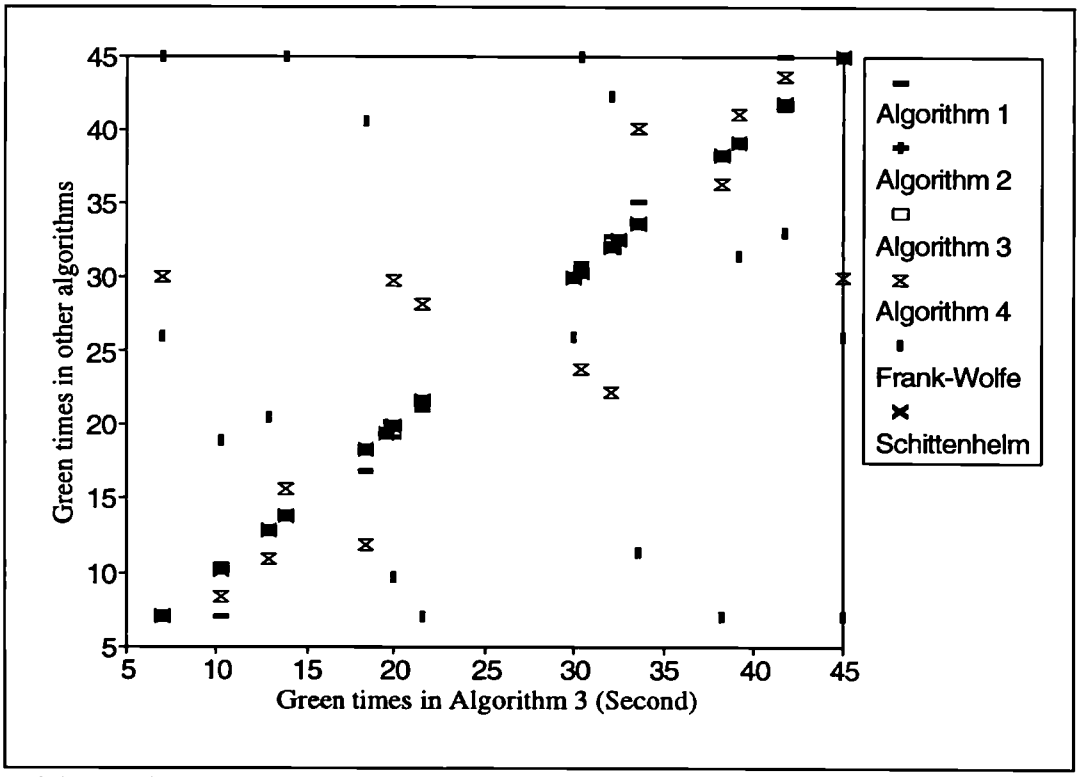


Figure 5-30 The comparison of green times obtained by algorithms on Sioux Falls network when γ =0.5

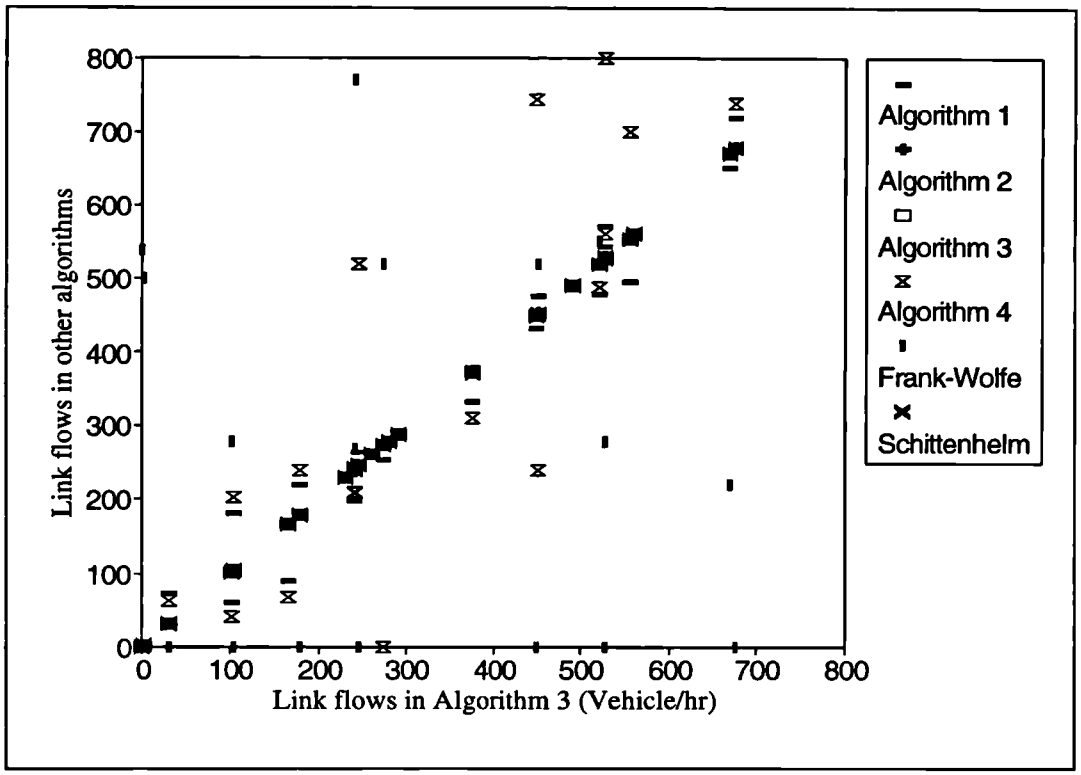


Figure 5-31 The comparison of traffic flows obtained by algorithms on Sioux Falls network when γ =0.75

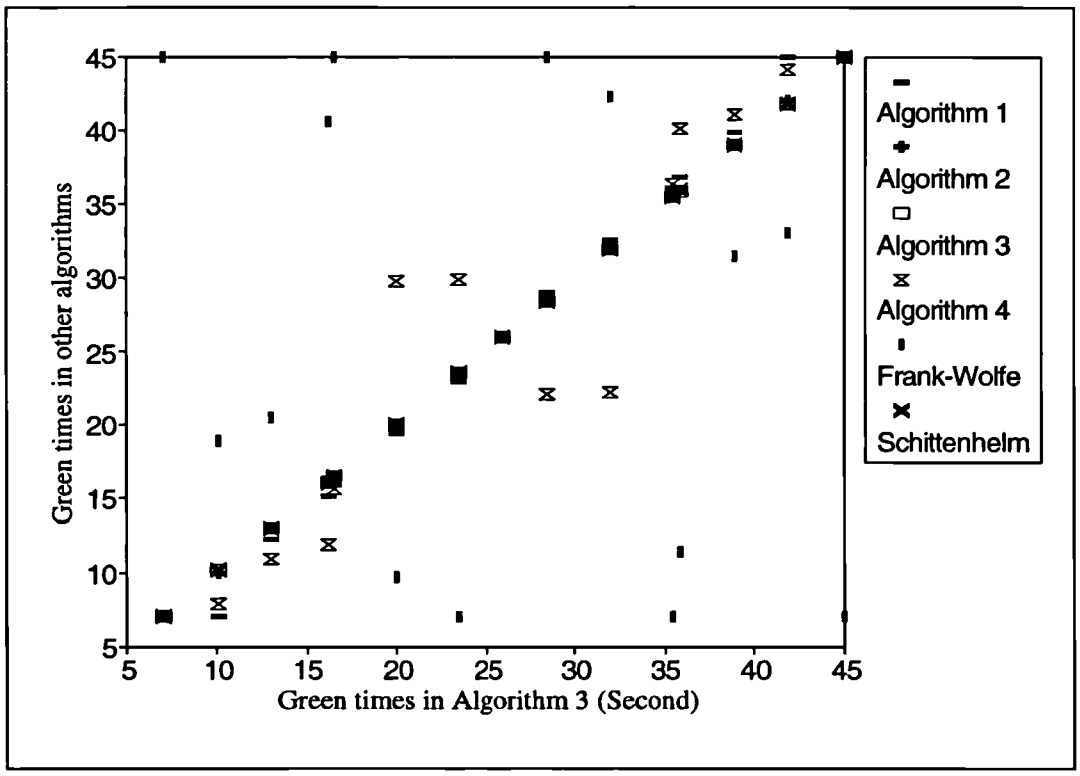


Figure 5-32 The comparison of green times obtained by algorithms on Sioux Falls network when γ =0.75

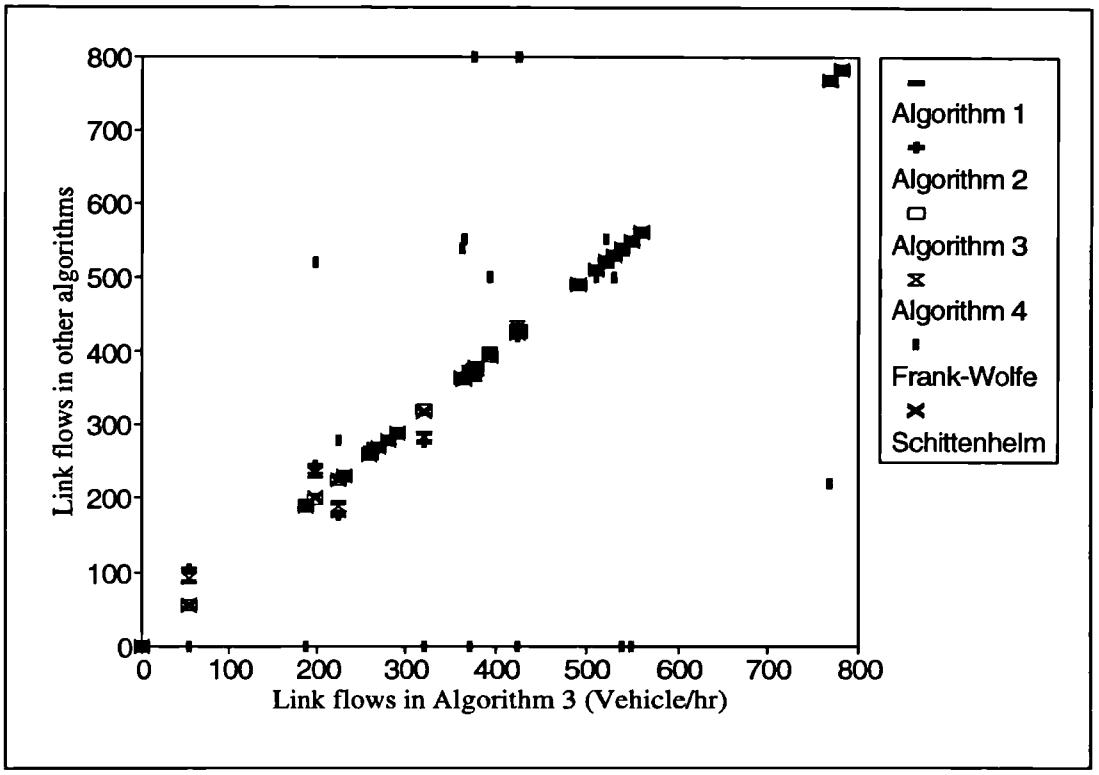


Figure 5-33 The comparison of traffic flows obtained by algorithms on Sioux Falls network when $\gamma = 1.0$ (No junction effect)

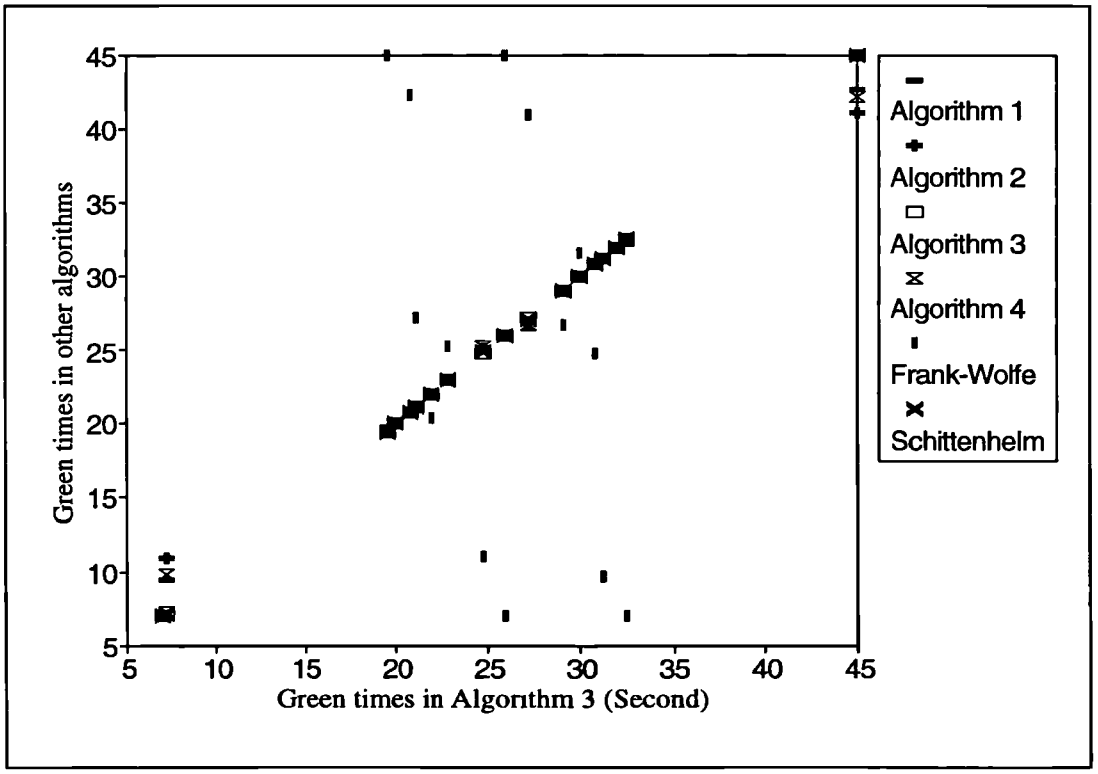


Figure 5-34 The comparison of green times obtained by algorithms on Sioux Falls network when $\gamma = 1.0$ (No junction effect)

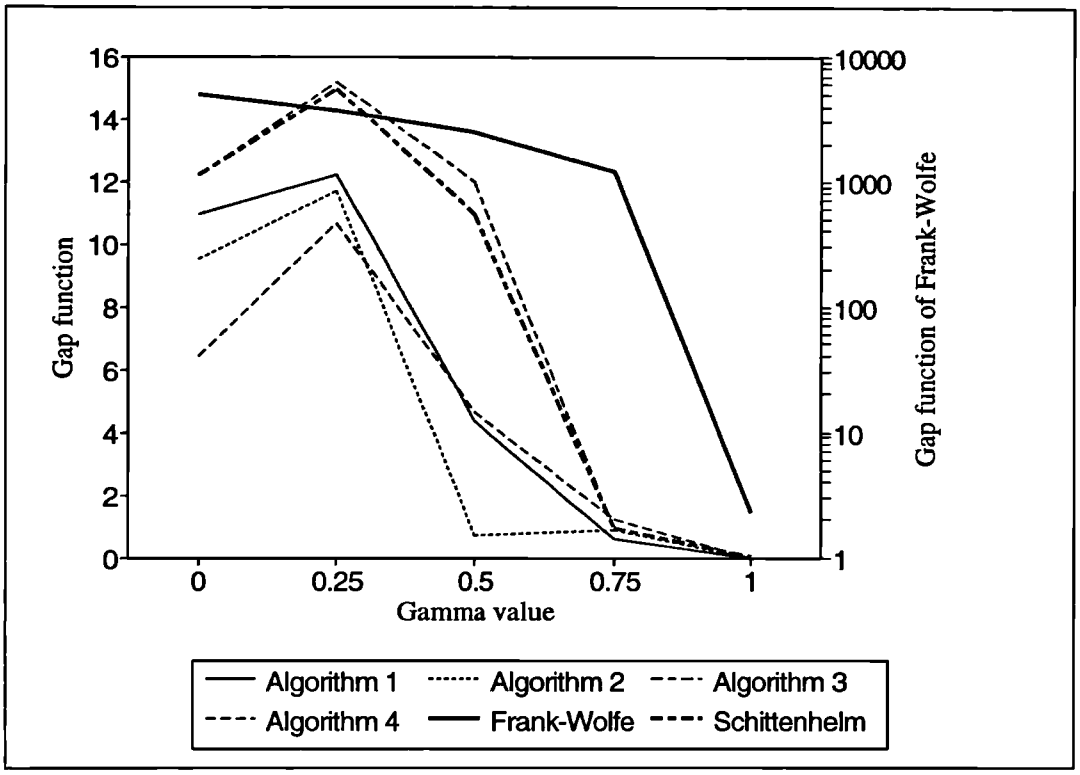


Figure 5-35 The gap function value of algorithms as γ changes on Charlesworth's network

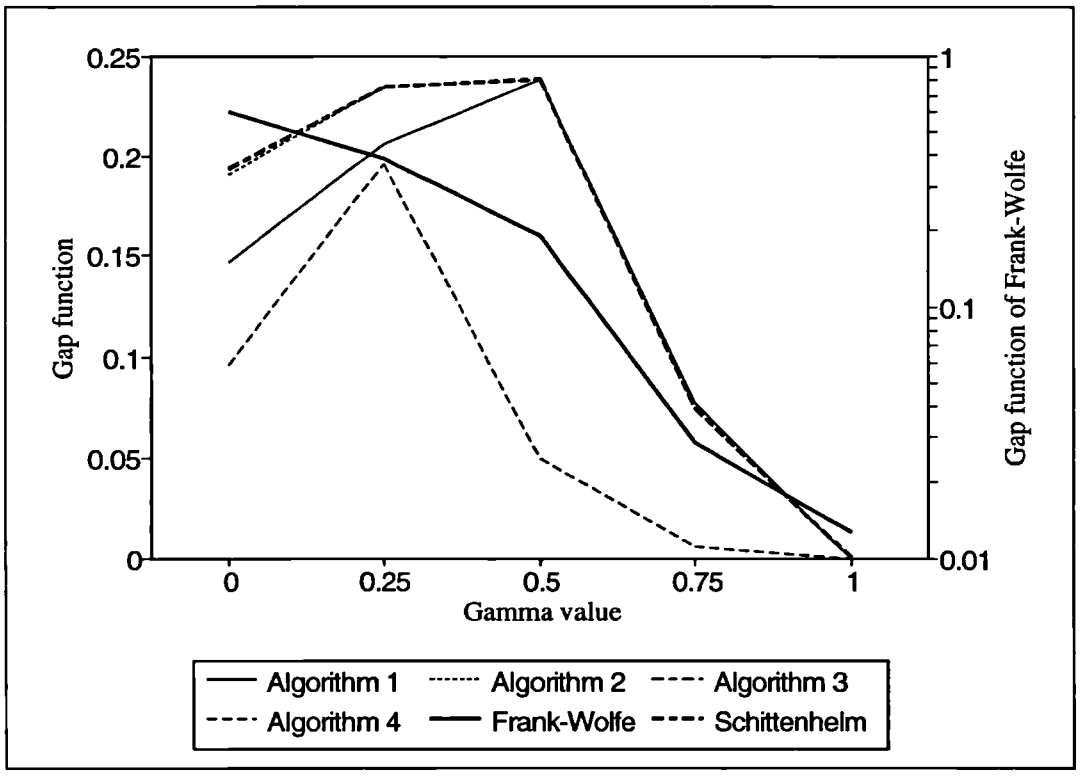


Figure 5-36 The gap function value of algorithms as γ changes on Sioux Falls network

CHAPTER 6 A MULTICLASS TRAFFIC ASSIGNMENT PROBLEM

6.1 INTRODUCTION

This chapter presents multiclass traffic assignment. Multiclass representation is introduced to represent variations in physical properties, route choice criteria and network restrictions. A cost function in the multiclass is discussed by considering interactions between different user classes. This will be related with the impact of congestion and different perceptions of cost when different vehicles and users are selecting their routes. A formulation of this problem is presented as a variational inequality. Solution method to this multiclass will be given in terms of a diagonalisation algorithm. A numerical example is analysed to test algorithms and properties such as existence of solution, and uniqueness and stability.

6.2 MULTICLASS REPRESENTATION

6.2.1 Introduction

In the multiclass traffic assignment as a link interaction, travel times on each link depend on the flows of different classes. The recognition of this stems from the reality that the performance of the link is affected by vehicles of many classes. The different vehicles have different occupancy rates, speeds, values of time and route choice criteria so that they affect the total cost function.

Some studies have been done to incorporate this concept of multiple classes into road traffic assignment. The historical development of multiclass models is here

reviewed briefly. Dafermos (1972), and Van Vliet, Bergman and Scheltes (1986) presented the multiclass user equilibrium assignment problem in which there are several classes of vehicles interacting on each link. In these models, the Jacobian matrix was taken as having a symmetric form which indicates that the impact on the cost of class i by class j equals to the impact on the cost of class j by class i. This is represented as $J_{ij} = J_{ji}$. Braess and Koch (1979) gave the condition of equilibria in asymmetric multiclass user assignment without the hypothesis of symmetry. This condition is that if a cost function for each class in the multiclass user assignment is continuous and monotone, then there is at least one user equilibrium flow pattern. Dafermos (1981) presented the general multimodal network equilibrium problem with elastic demand. In her paper the link cost for each class depends on the entire load pattern and the travel demands. For the more general asymmetric case, Abdulaal and LeBlanc (1979) studied the combination of modal split and equilibrium assignment. Fisk and Nguyen (1981) gave the existence and uniqueness properties of an asymmetric two-mode (auto and public transit) equilibrium model. Daganzo (1983) studied a stochastic user equilibrium model with multiple vehicle types in which he discussed a family of general link cost functions that can be used to model a multimodal transportation network.

6.2.2 Concept of class

The introduction of the concept of class into road traffic assignment allows us to categorise drivers and vehicles according to some features that affect route choice behaviour. The class can be categorised by the physical properties of vehicles, criteria of route choice and road network restrictions.

The physical properties of vehicles can be illustrated by the comparison between passenger car and heavy goods vehicles. Their marginal effect on congestion differs. Heavy goods vehicles are larger than passenger cars and thus occupy more road space. Heavy goods vehicles have poorer operating capabilities than passenger cars, particularly with respect to acceleration, braking, and the ability to maintain speed on uphill gradients.

Criteria of route choice also can be classified by the class of drivers. Some drivers select their routes according to distance or travel time or scenery or some combination of these. In particular, this is distinguished by the purpose of the trip. Work trips are mainly affected by travel time. However, trips for leisure are influenced by scenery or distance as well as travel time. Route guidance control can be classified into this category. It is because in the view of a system manager, an objective would be to direct traffic so as to minimise the total system cost of travel. However, the users are generally able only to make unilateral decisions so they try to minimise their own individual costs of travel.

Network restrictions can be one of the categories which occurs often in practice. Public transport (e.g., bus) has priority in order to boost its usage. In particular, in some urban areas right of way is given only to buses. In the bus lanes, the interaction is only between buses as all other vehicles are excluded. In this case, we need to consider only the interaction within the class of buses running in bus lanes.

6.2.3 Passenger car units (PCU's)

Heavy vehicles such as trucks and buses use different amounts of capacity from passenger cars. Some adjustments for vehicles other than passenger cars are required to obtain the value of capacity and delay functions. The US HCM (US Department of

Transport, 1985) computed adjustment values for these vehicles. The values represent the number of passenger cars that would consume the same amount of the highway capacity as one truck or bus under prevailing roadway and traffic conditions.

The prevailing roadway and traffic conditions are classified as:

1. Level terrain concerns any combination of grades, and horizontal or vertical alignment, that permits heavy vehicles to maintain approximately the same speed as passenger cars.

2. Rolling terrain concerns any combination of grades, and horizontal or vertical alignment, that causes heavy vehicles to reduce their speeds substantially below those of passenger cars, but not causing heavy vehicles to operate at crawl speeds for any significant length of time.

3. Mountainous terrain concerns any combination of grades, and horizontal or vertical alignment, that causes heavy vehicles to operate at crawl speeds for significant distances or at frequent intervals.

Crawl speed is the maximum sustained speed which trucks can maintain on a given extended uphill gradient. Passenger car units (PCU) for heavy vehicles on various highway segments are given in Table 6-1. More detailed adjustments of the passenger car units are calculated according to specific gradients and vehicular weight and length. For the values, see HCM (1985, pp3.11-17).

Factors	Type of terrain		
	Level	Rolling	Mountainous
PCU for trucks	1.7	4.0	8.0
PCU for buses	1.5	3.0	5.0

Table 6-1 PCU values for trucks and buses (Source: US DoT, 1985, pp3.13)

6.2.4 Generalised travel cost

Generalised cost is used to describe some factors that affect drivers' route choice. We then model drivers' behaviour by assuming that each of them selects their routes so as to minimise their own generalised travel cost. Some of these factors include in-vehicle time, excess time, distance, out-of pocket cost, value of travellers' time, comfort and convenience. A form of generalised cost, C_{ij}^k for journey from zone i to zone j for class k can be as follows:

$$C_{ij}^k = \alpha^k I_{ij}^k + \beta^k e_{ij}^k + \gamma^k d_{ij}^k + \delta^k o_{ij}^k + \theta t_{ij}^k \ldots$$

Where

I_{ij}^k is in-vehicle time for the journey

e_{ij}^k is excess time for the journey such as walking and waiting time

d_{ij}^k is distance for the journey

o_{ij}^k is out-of pocket cost such as toll and petrol cost

t_{ij}^k is travel time for the journey

θ is a parameter to represent different values of travel time

α^k β^k γ^k and δ^k are parameters to represent characteristics of each user.

Among components in the generalised cost, travel time can be estimated relatively easily and varies with flow. Moreover, the different values of travellers' time can be reflected by using different parameters in the travel time.

Value of time can be classified by working time values, and by non-working time values. The working time values are related to the costs of hiring an employee. These costs can include overhead costs such as insurance and pensions, and other costs which vary with working hours. These values can be measured by the gross wage rate of the employee. In 1985/6 National Travel Survey (see, COBA, 1989, pp8.3-8.7), values for occupational groups were obtained as Table 6-2 and Table 6-3. Note that these values are mainly for evaluation rather than behaviour analysis. In the survey, the non-working time values are related to all non-working journey purposes. This includes commuting time, and personal business, shopping and leisure travels by all modes.

Working time	
Classification	Value of time (unit: pence per hour)
Car driver	849.7
Car passenger	705.3
Bus passenger	701.2
Rail passenger	1066.1
Underground passenger	1050.0
All workers	841.6
Non-working time	
Standard appraisal value	207.5

Table 6-2 Resource values of time per person (Source: COBA, 1989)

Classification	Value of time (unit: pence per hour)
Working car	990.8
Light goods vehicle	859.0
Other goods vehicle	622.5
Public service vehicle	3213.7
Non-working car	383.9

Table 6-3 Resource values of time per vehicle (Source: COBA, 1989)

6.3 FORMULATION

It is assumed that there are K classes of vehicles on a link indexed by k=1,2,...,K. A transportation system extended to include multiclasses consists of two components: an extended transportation supply and an extended travel demand. The extended transportation supply is a set of facilities for the K classes of the transportation network. The extended travel demand is a number of K user classes using the network. A representation of the extended transportation supply and travel demand is necessary to compute the flow pattern which depicts the interaction between classes because delays are identified and depend on the weighted sum of flows. This is known as the extended traffic assignment process.

The extended traffic assignment problem is to determine the flow pattern on each path of the network, given specific demand associated with each class for each origin-

destination (O-D) pair in the road network, and travel cost functions for each user class on each link of the network.

6.3.1 An extended transportation supply

The extended transportation supply is a set of facilities for K user classes provided by the road network which can be represented by a directed graph. A directed graph for user class k is a pair of sets $G^k=(N^k, L^k)$, where N^k is a set of nodes for user class k and L^k is a set of directed links for user class k. This can be reduced to single network $\mathbf{G}=(\mathbf{N}, \mathbf{L})$ where $\mathbf{N}=N\times K$ and $\mathbf{L}=L\times K$ (where the symbol $\times$ denotes the Cartesian product of sets). A subset B^k of $N^k\times N^k$, where $(o^k, d^k)\in B^k$ implies that $o^k\neq d^k$, designates the set of origin-destination pairs for users in class k.

Volume-delay function represents travel costs c_a^k , k=1,2...,K, on link a of K user classes, which is a measure of transportation supply. Travel cost can be time spent to travel along the link or even a more generalised cost. In the simplest case of this, travel cost for each link is assumed to be a function of flow v_a^k, k=1,2,...,K, in each class on that link. This represents the number of users in each class entering the link in a time unit.

In this study, we assume that travel cost on each link of each class consists of a fixed and a variable component. The example of the fixed component is a distance, which is not changed by the amount of traffic flow on link. The example of the variable component is the BPR function, which is changed by the amount of traffic flow on link. The total link cost on link a of the class k is expressed as:

$$c_a^k = F^k + t(v_a)$$

where c_a^k is the total link cost on link a and F^k is a fixed cost for the class k and $t(v_a)$ is a travel time function for the total link flow on link a.

The traffic flow on link a is the summation of each class k on link a,

$$v_a = \sum_{k=1}^{K} \upsilon^k v_a^k$$

where υ^k is the PCU value for class k.

Cost functions for each class represent the behaviour of travellers. This means that the strength of the influence of users of class i on users of class j is indicated by the corresponding cell of the Jacobian matrix. The strength of the influence of users of class j on those of class i according to the Jacobian amount is not always the same, because the effects on the congestion between classes can differ.

6.3.2 An extended travel demand

The extended travel demand represents the K user classes' desire to travel, which can be categorised by purpose (work trip, study trip, business trip, leisure trip) and time of day. It is numerically expressed by the number of K user classes travelling between each origin-destination pair. If travel demand is a decreasing function of travel cost between O-D pairs, it is an elastic demand. Otherwise, the travel demand is assumed to be a fixed.

Associated with each $(o^k, d^k) \in B^k$ is a positive real number T_{od}^k called the demand for O-D pair (o,d) of user class k. The flow on each path p of user class k is denoted t_p^k. A flow conservation equation for user class k has to be satisfied for each pair (o^k, d^k);

$$\sum_{p \in P_{od}} t_p^k = T_{od}^k, \quad \forall (o^k, d^k) \in B^k, \quad \forall k \in K$$

where P_{od} is the set of all available paths for O-D pair and K represents user class.

6.3.3 An extended demand and supply interaction

The extended travel demand depends on travel costs and travel costs on links and hence paths depend on the flow pattern. A mutual interaction between demand and cost as a measure of supply is thus induced. This interaction has two simultaneous effects: the determination of demand as a function of travel costs, and the determination of flow pattern due to the dispersion of demand. If demand is assumed to be fixed, the interaction yields only the latter effect.

Through this interaction between travel demand and transportation supply the system is in a state of equilibrium if the path flows obtained in the current iteration give rise to travel costs that are consistent with those in the previous iteration according to a principle of route choice.

6.3.4 Extended assignment principles

If each user class is assumed to be homogeneous with respect to the route choice criterion which is to minimise their generalised travel costs, an equilibrium condition is reached, if at all, when no user can reduce his own cost by selecting different paths. In this condition, all paths used by each class between O-D pairs have the same cost. This condition can be stated as a formal extension of Wardrop's first principle (1952): "The journey times on all routes actually used by users of each class are equal and less than

those which would be experienced by a single vehicle of that class on any unused route". As a result of this principle, the paths actually used by each user class are shortest paths at the existing condition. This flow pattern is called descriptive because it describes at least approximately the real phenomenon of a transportation network.

Wardrop's second principle is known as the system optimum principle which represents the situation in which the total travel cost in the whole transportation system is minimised. This flow pattern is called prescriptive because the flows of the system optimum must be forced upon the K user classes.

The extended version of Wardrop's first principle(1952) for user class k can be expressed by the following relations;

$$\left.\begin{array}{l} t_p^k > 0 \Rightarrow C_p^k = M_{od}^k \\ t_p^k = 0 \Rightarrow C_p^k \geq M_{od}^k \end{array}\right\} \forall p \in P_{od}, \forall (o^k, d^k) \in B^k, \forall k \in K$$

where C^k_p is the travel cost of using path p by class k and M^k_{od} is the minimum travel cost from origin o to destination d by class k.

The extended traffic assignment problem can be formulated as a variational inequality. The extended Wardrop equilibrium conditions for user class k is equivalent to the following variational inequality formulation. The feasible path flows of user class k, t^k is an equilibrium if for each k and any feasible path flows of other users belonging to class k, s^k,

$$(s^k - t^k)C^k(t) \geq 0$$

Using the link and path incidence relation, we can represent the path oriented variational inequality as the link oriented variational inequality as:

$$(u^k - v^k)c^k(v) \geq 0$$

In general, we can use a gap function as an objective function, which shows the difference between current and previous assigned flows. The gap function could be the following form:

$$G(v) = \max_{u \in \delta} \{c(v)(v - u)\}$$

If the value of the gap function is zero, that is G(v)=0, the assigned value is an equilibrium.

6.4 SOLUTION METHOD

One of the heuristic algorithms for this non-separable case is based on an iterative diagonalisation (or relaxation) procedure, where each iteration requires the solution of a full-scale standard user equilibrium problem (Fisk and Nguyen, 1982). Another approach is a streamlined version of the first one; it uses a single iteration of the user equilibrium solution procedure to be performed at each iteration of the diagonalisation procedure. This streamlined version has been found to be more efficient (Sheffi, 1985, pp220-228).

Van Vliet, Bergman and Scheltes (1986) used a diagonalisation algorithm for traffic assignment with multiple user classes. The computation is done by assuming that every class of traveller has an individual copy of a network and that class uses only the network belonging to it. However, the calculations for individual networks include the interaction that the cost on each link depends on the flow assigned to corresponding links in other copies of the network. Dafermos (1981) presented a Newton projection method for solving the general multimodal equilibrium problem in the framework of the diagonalisation method.

In this study, the diagonalisation procedure is used to solve a sequence of separable cost problems for each class in turn, and we adopt the computational approach of Van Vliet et al (1986). The simplicial decomposition algorithms introduced in chapter 3 are used as the basis of a diagonalised strategy. In the case of Algorithm 1, this is as follows.

A diagonalisation of Algorithm 1 in multiclass assignment

Step 0: (Initialisation)

Iteration n=0

For each class k=1, 2, ... , K

Identify a feasible assignment, $v^{(n,k)} \in \Omega^{(k)}$

where (n,k) represents iteration n and class k

Set $W^{(n,k)} = \{v^{(n,k)}\}$

Repeat steps 1 and 2 for each class, k=1,...,K until convergence:

Step 1: (Linear subproblem)

1-1 Update link cost for class k, $c_a^k(v_a^n)$ based on a fixed cost for class k, F^k, and traval time function (here BPR function) for the total link flow on link a, $t(v_a^n)$

$$c_a^k(v_a^n) = F^k + t(v_a^n)$$

where

$$v_a^n = \sum_{k=1}^{K} \upsilon^k v_a^{(n,k)}$$

where υ^k is a PCU value for class k.

1-2 Perform all-or-nothing assignment based on $c_a^k(v_a^n)$

$$u^{(n,k)} = \arg \min_{u} \{c^k(v^n)u \ ; \ u \in \mathcal{B}^k\}$$

1-3 Convergence test

If $G(v^{(n\,k)}) = c^k(v^{(n\,k)})(v^{(n,k)} - u^{(n,k)}) \approx 0$, stop: optimum solution is $v^{(n,k)}$

Otherwise, $W^{(n+1\,k)} = W^{(n,k)} \cup \{u^{(n,k)}\}$

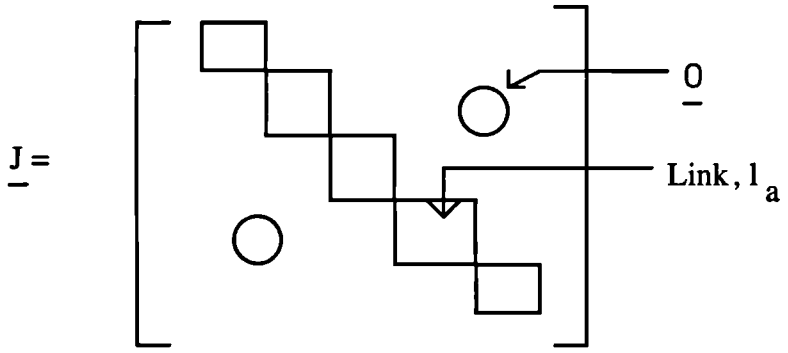
Step 2: (Master subproblem)

Let $v^{(n+1,k)} = \arg \min_{v} \{G(v); \ v \in H(W^{(n+1,k)})\}$

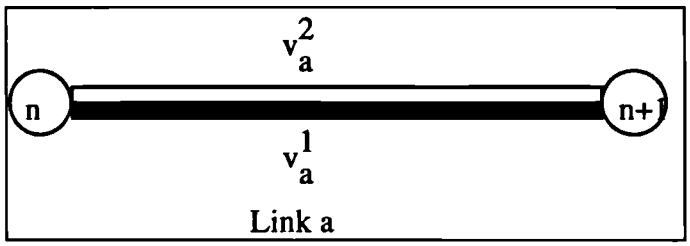
n=n+1

6.5 ANALYSIS FOR GOOD BEHAVIOUR

We will investigate a form of the Jacobian matrix in a multiclass assignment. The general form of the Jacobian matrix is:



where l_a has a form of non-zero elements. In the two class case, the link a has an interaction between class 1 and class 2 as follows:



Total traffic flow on link a is the summation of volumes of class 1 and 2 on link a as:

$$v_a = \sum_{k\ 1}^{2} \upsilon^k v_a^k$$

where υ^k is the pcu value of class k.

The cost on link a has the form of generalised cost of travel time and distance as:

$$c_a^k = \theta^k d_a^k + \theta^{k0} t_a(v_a)$$

where θ^k and θ^{k0} represent parameters of the values of time in imposing the preferences between distance d^k_a and travel time t_a by class k.

The Jacobian matrix of the two class case in the above figure is:

$$J_n = \begin{bmatrix} \frac{\partial c_a^1}{\partial v_a^1} & \frac{\partial c_a^1}{\partial v_a^2} \\ \frac{\partial c_a^2}{\partial v_a^1} & \frac{\partial c_a^2}{\partial v_a^2} \end{bmatrix}$$

where the derivative is a following form:

$$\frac{\partial c_a^k}{\partial v_a^j} = \theta^{k0} \frac{\partial t_a(v_a)}{\partial v_a^j}$$
$$= \theta^{k0} \upsilon^j \frac{\partial t_a(v_a)}{\partial v_a}$$

The resulting Jacobian matrix is:

$$J = \begin{bmatrix} \theta^{10}\upsilon^1 & \theta^{10}\upsilon^2 \\ \theta^{20}\upsilon^1 & \theta^{20}\upsilon^2 \end{bmatrix}$$

$$\det\ (J) = \theta^{20}\theta^{10} \det\begin{pmatrix} \upsilon^1 & \upsilon^2 \\ \upsilon^1 & \upsilon^2 \end{pmatrix} = 0$$

Furthermore, the symmetric part of the Jacobian matrix is

$$J + J^T = \begin{bmatrix} 2\theta^{10}\upsilon^1 & \theta^{10}\upsilon^2 + \theta^{20}\upsilon^1 \\ \theta^{10}\upsilon^2 + \theta^{20}\upsilon^1 & 2\theta^{20}\upsilon^2 \end{bmatrix}$$

and this has determinant

$$\begin{aligned} \det(J + J^T) &= 4\theta^{10}\upsilon^1\theta^{20}\upsilon^2 - (\theta^{10}\upsilon^2 + \theta^{20}\upsilon^1)^2 \\ &= -(\theta^{10}\upsilon^2 - \theta^{20}\upsilon^1)^2 \le 0 \end{aligned}$$

So, the Jacobian matrix is not a P-matrix and therefore cannot be positive definite, although it is on the boundary of having the P property. That is, the Jacobian matrix can never be positive definite, although if $(\theta^{10}\upsilon^2 - \theta^{20}\upsilon^1) = 0$, it is positive semi-definite. Even though this is analysed on two class case, this can apply to the K class case in the same way.

In this study, we assume that the pcu value of each class is given as $\upsilon^1=\upsilon^2=1$, and the parameter in the travel time for each class is $\theta^{10}=\theta^{20}=20$ so that the determinant of the Jacobian is zero. However, to represent the heterogeneity of the user by class, we assume that the parameters for the distance differ as $\theta^1 = 10$ and $\theta^2 = 100$. This means that the users in class 1 take account of the distance factor 10 times less than those of class 2 when they decide on their route choice.

To examine a unique solution property of this model, we use a simple symmetric network in Figure 4-1. It has 2 origins and 2 destinations, 6 nodes and 8 links. There are

two classes of users. The demands of class 1 and class 2 from origin 1 are the same as in Table 6-4, and are entirely to destination 5 and from origin 2 to destination 6. The capacity of each link is 2000 vehicles / hour. The generalised cost on link is a summation of distance and travel time. The distance of each link is 80 metre. The parameters of distance and travel time are the same as above.

(Unit: vehicles / hour)

Origin / Destination	5	6
1	800	0
2	0	800

Table 6-4 Demands for class 1 and class 2 in the small network in Figure 4-1

In this example, a symmetric equilibrium solution of traffic flow is clearly provided by:

$$v^* = (v^*_1, v^*_2, v^*_3, v^*_4)$$
$$=1/2(T^1_{15} + T^2_{15}, T^1_{15} + T^2_{15}, T^1_{26} + T^2_{26}, T^1_{26} + T^2_{26})$$

Table 6-5 shows that the calculated values using different solution algorithms on the small contrived network in Figure 4-1 are the same as the symmetric equilibrium solution in the total link cases whilst for each class the calculated values have two kinds: one is the same as those of the symmetric equilibrium solution, and the other is an all-or-nothing result for each class on link. Table 6-6 shows the performances of algorithms in terms of the used iteration number and cumulative CPU time to converge or terminate. Algorithms 3 and 4, and the Frank-Wolfe converge to gap function of 0.0. Algorithm 4 converges fastest in less than 0.01 seconds on a Sun Sparc station 370 GX.

	Algorithms					
	1	2	3	4	F-W	Schtthlm
v^*_1	800	800	800	800	800	800
v^n_1	800	800	800	800	800	800
$v^{(*,1)}_1$	400	400	400	400	400	400
$v^{(n,1)}_1$	400	800	800	400	800	400
$v^{(*,2)}_1$	400	400	400	400	400	400
$v^{(n,2)}_1$	400	0	0	400	0	400
v^*_3	800	800	800	800	800	800
v^n_3	800	800	800	800	800	800
$v^{(*,1)}_3$	400	400	400	400	400	400
$v^{(n,1)}_3$	400	800	800	400	800	400
$v^{(*,2)}_3$	400	400	400	400	400	400
$v^{(n,2)}_3$	400	0	0	400	0	400

Table 6-5 The symmetric equilibrium solution and calculated values using different algorithms when demands $T^1_{15} = T^2_{15} = 800$, Demands $T^1_{26} = T^2_{26} = 800$ are used.

Key:

v^*_a	Symmetric equilibrium total traffic flow on link a
v^n_a	Calculated total traffic flow on link a by the algorithms
$v^{(*,k)}_a$	Symmetric equilibrium traffic flow for class k on link a
$v^{(n,k)}_a$	Calculated traffic flow for class k on link a by the algorithms

	Iterations	CPU (Second)	Gap (Vehicle-second)
Algorithm 1	6	0.21	0.12
Algorithm 2	8	0.34	0.12
Algorithm 3	1	0.01	0.00
Algorithm 4	1	0.00	0.00
Frank-Wolfe	1	0.01	0.00
Schittenhelm	10	0.15	0.25

Table 6-6 Performances of algorithms in the small network shown in Figure 4-1 in multiclass assignment

6.6 NUMERICAL EXAMPLES

6.6.1 Charlesworth's network

The network and travel demands devised by Charlesworth (1977) are used as a first example to compare the performance of the various algorithms in multiclass assignment (for the figures and data of the network, see Appendix 2). In this case there are two user classes: the demands for each class are the same as shown in the Appendix A2.1 so that the total demand is doubled. Note that the BPR function is used here so that the capacity on the link is not limited (see section 2.6.2). The generalised cost function and parameters are the same as the preliminary example. In Table 6-7, the performance of each of the algorithms is shown in terms of iteration number, cumulative CPU time on

SUN Sparc station 370 GX and the mean gap value when they terminate. Algorithm 3 runs quickest to a mean gap value of 1.739 seconds in 32.43 CPU seconds. The other algorithms including the Frank-Wolfe perform well in this test. Note that each algorithm is stopped by the maximum number iteration criterion rather than the required level of gap value.

Figure 6-1 shows the comparative performances of each of algorithms in terms of CPU time and the mean gap value. The gap value changes only slightly as the iteration number increases. Figure 6-2 compares the final total link flows obtained by each of the algorithms with those of Algorithm 3. The differences in values between them are relatively small. Figures 6-3 and 6-4 respectively compare the final link flows of class 1 and class 2 obtained by each algorithm with those of Algorithm 3. The difference in values of class 1 flow occurs on only few links. In class 2, the difference is bigger in some cases. Figures 6-5, 6, 7, 8, 9 and 10 show that the proportion of class 1 link flow in terms of the total link flow plotted for each of Algorithms 1, 2 , 3 and 4, the Frank-Wolfe, and the Schittenhelm algorithm respectively. These figures indicate that the class 1 link flows are similar for each of algorithms. Most proportions of class 1 link flows in terms of the total link flows are around half.

	Iterations	CPU (Second)	Mean gap (Second)
Algorithm 1	31	166.86	1.935
Algorithm 2	31	169.85	1.739
Algorithm 3	31	32.43	1.739
Algorithm 4	31	163.85	2.129
Frank-Wolfe	100	87.06	2.258
Schittenhelm	31	33.32	1.885

Table 6-7 Performances of algorithms in the Charlesworth network in multiclass assignment

6.6.2 Sioux Falls network

As a second example, the network and travel demands of the Sioux Falls are used to compare the performance of the various algorithms in multiclass assignment (for the figures of the network, see Figure 3-6). In this case there are two user classes: the demands for each class are the same as shown in Table 3-1 resulting in the doubled demand with the BPR function. The generalised cost function and parameters are the same as the preliminary example. In Table 6-8, the performance of each of the algorithms is shown in terms of iteration number, cumulative CPU time on SUN Sparc station 370 GX and the mean gap value when they terminate. Algorithm 3 runs quickest to a mean gap value of 0.049 seconds in 6.63 CPU seconds. As in the first example, the other algorithms perform well and reach a similarly good level of convergence, as indicated by the mean gap value. However, as before each algorithm is stopped by the maximum iteration number criterion. Note that in this case, all of the gap values are similarly small, although the CPU time usage varies up to 10 times between algorithms.

Figure 6-11 shows the comparative performance of each of the algorithms in terms of CPU time and gap value. The gap value fluctuates in the initial stages and then stabilises. Figure 6-12 compares the final total link flows obtained by each of the algorithms with those of Algorithm 3. The difference in values between them is relatively small. Figures 6-13 and 6-14 compare the final link flows of class 1 and class 2 obtained by each of the algorithms with those of Algorithm 3 respectively. The difference in values of class 1 is relatively small while that of class 2 is larger in some cases with the Frank-Wolfe algorithm. Figures 6-15, 16, 17, 18, 19 and 20 show the proportion of class 1 link flow plotted against the total link flow when Algorithms 1, 2 , 3 and 4, the Frank-Wolfe, and the Schittenhelm algorithm are used respectively. These

figures indicate that the class 1 link flows are similar for each of the algorithms, though the flow on some links by the Frank-Wolfe algorithm is cluster. Most proportions of class 1 link flows in terms of the total link flows are shown to be around half with substantially more variation in this network than in the case of the Charlesworth network.

In summary, Algorithm 3 performs best in both the Charlesworth and the Sioux Falls networks in terms of CPU time. The Schittenhelm algorithm is good in terms of CPU time and level of convergence as indicated by gap value. The convergence pattern of each of the algorithms fluctuates in the initial stage and changes slightly afterwards. The proportion of class 1 link flow in terms of the total link flow on the Charlesworth network is less diverse than that of the Sioux Falls network because the Sioux Falls network has more possible route choices than Charlesworth's.

	Iterations	CPU (Second)	Mean gap (Second)
Algorithm 1	31	65.00	0.047
Algorithm 2	31	64.78	0.048
Algorithm 3	31	6.63	0.049
Algorithm 4	31	63.94	0.054
Frank-Wolfe	100	18.57	0.064
Schittenhelm	31	6.86	0.048

Table 6-8 Performances of algorithms in the Sioux Falls network in multiclass assignment

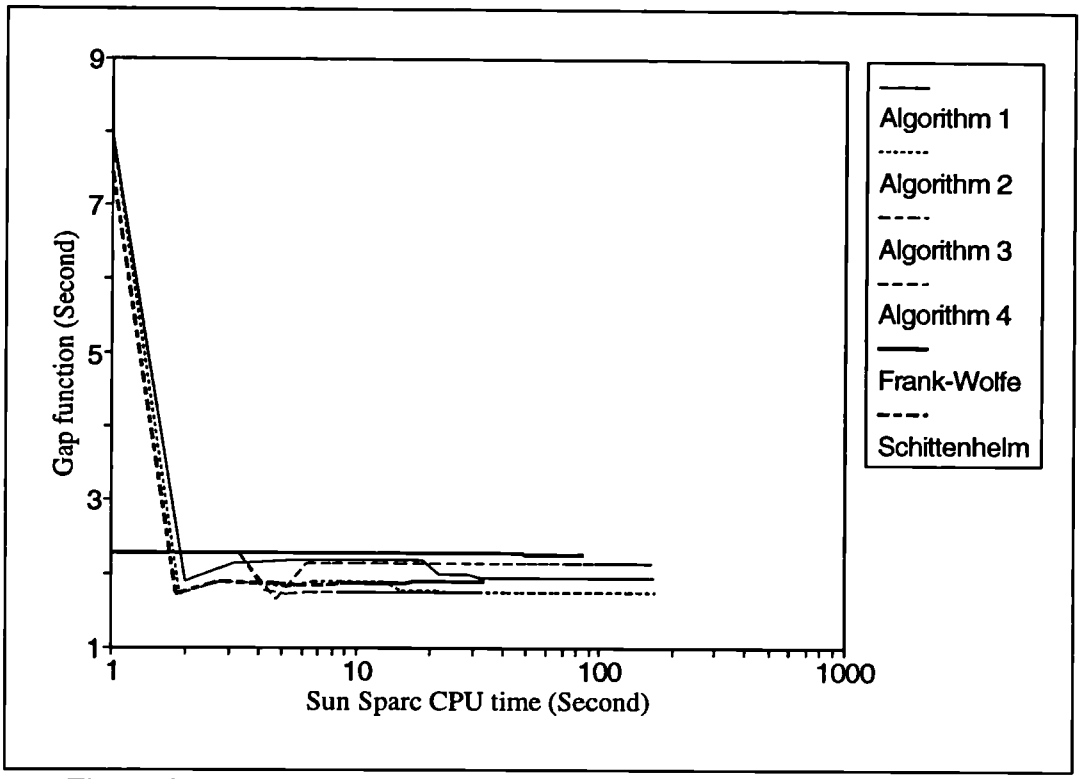


Figure 6-1 The performance of algorithms in Charlesworth's network

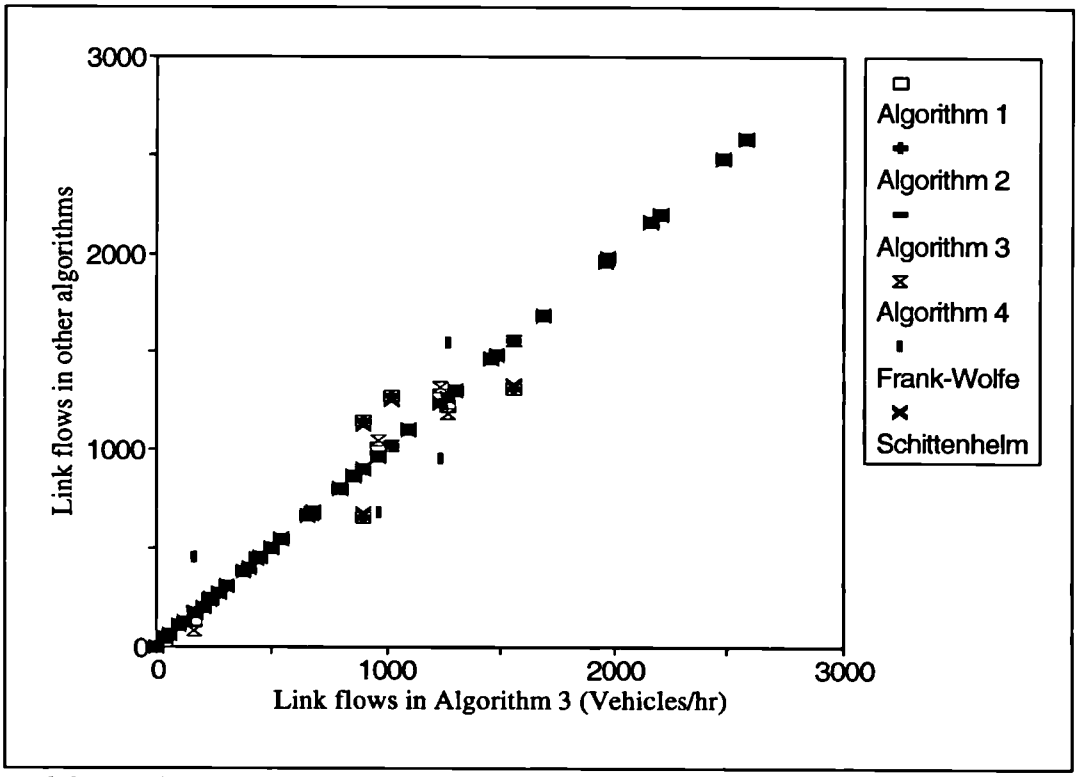


Figure 6-2 The comparison of total link flows obtained by algorithms on Charlesworths' network

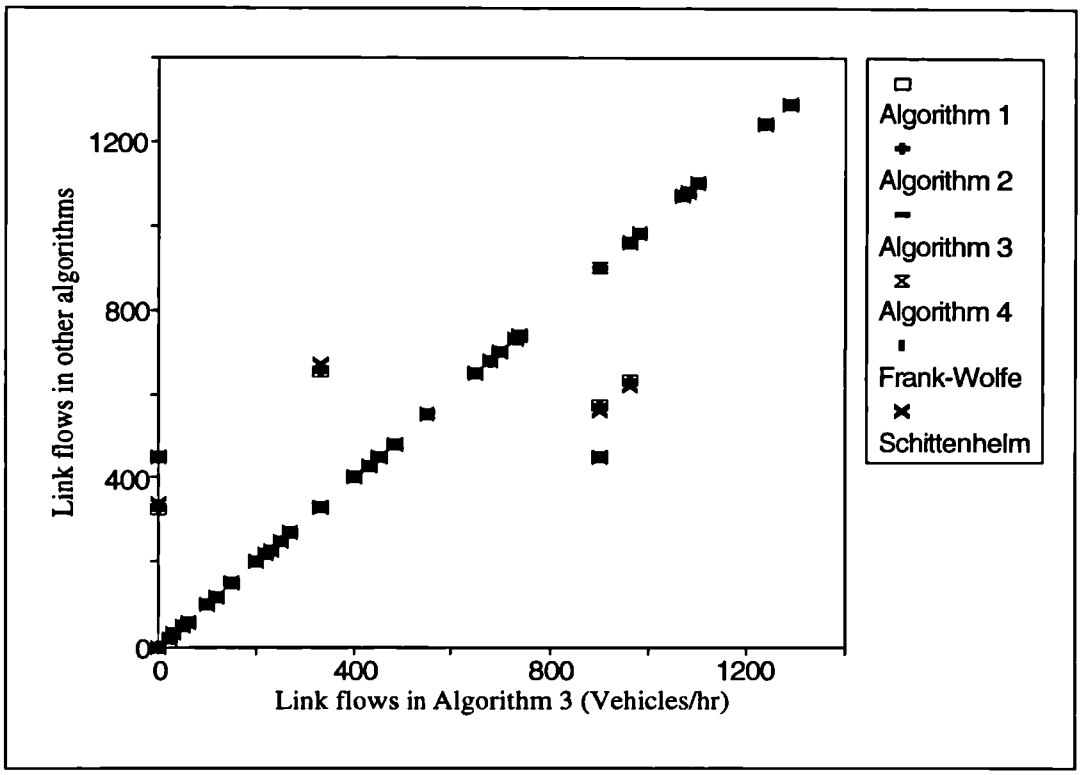


Figure 6-3 The comparison of link flows of class 1 obtained by algorithms on Charlesworths' network

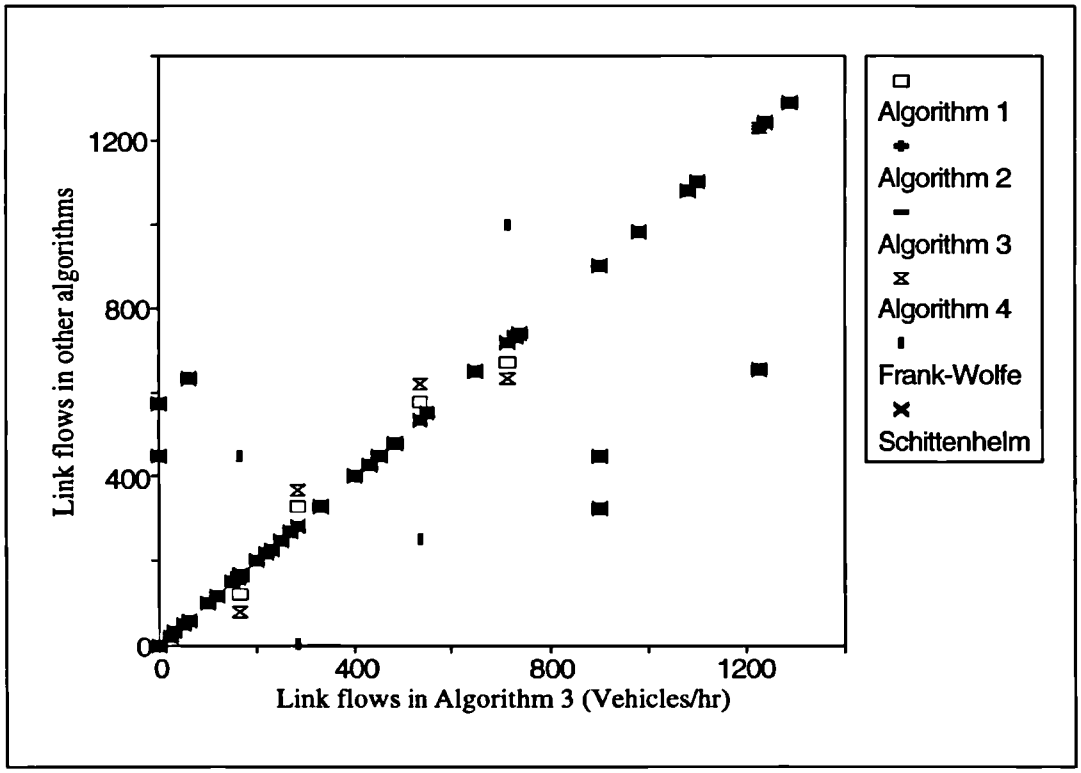


Figure 6-4 The comparison of link flows of class 2 obtained by algorithms on Charlesworths' network

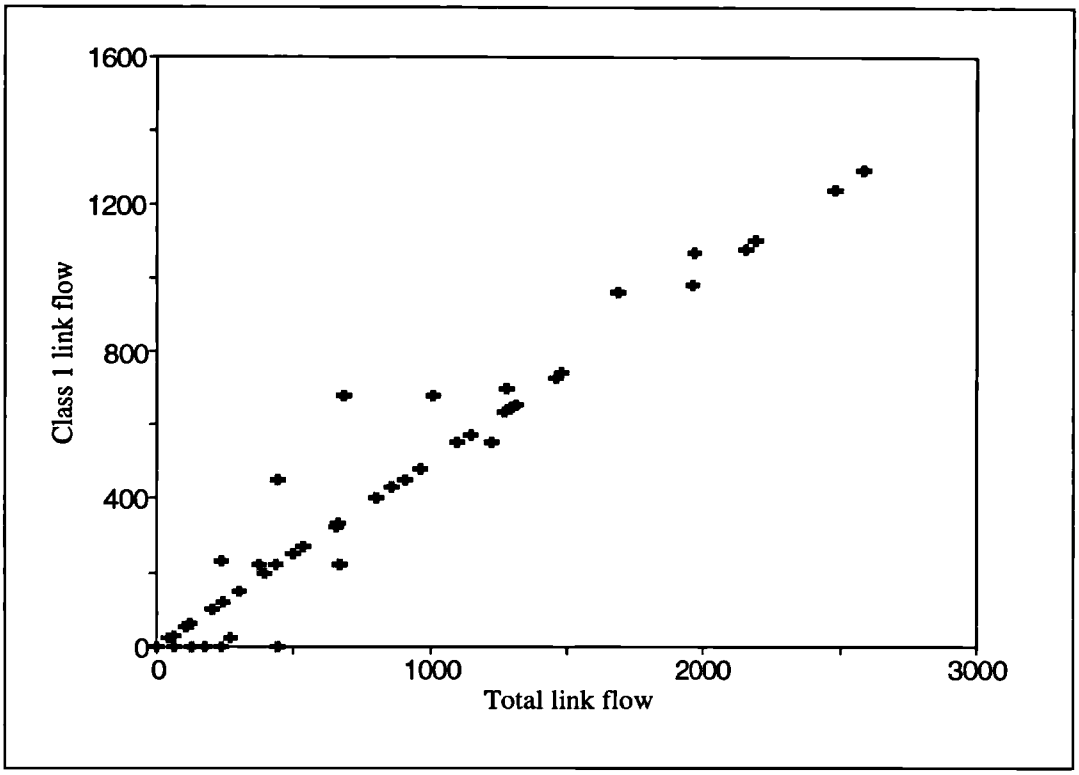


Figure 6-5 Link flows obtained by Algorithm 1 in Charlesworth's network

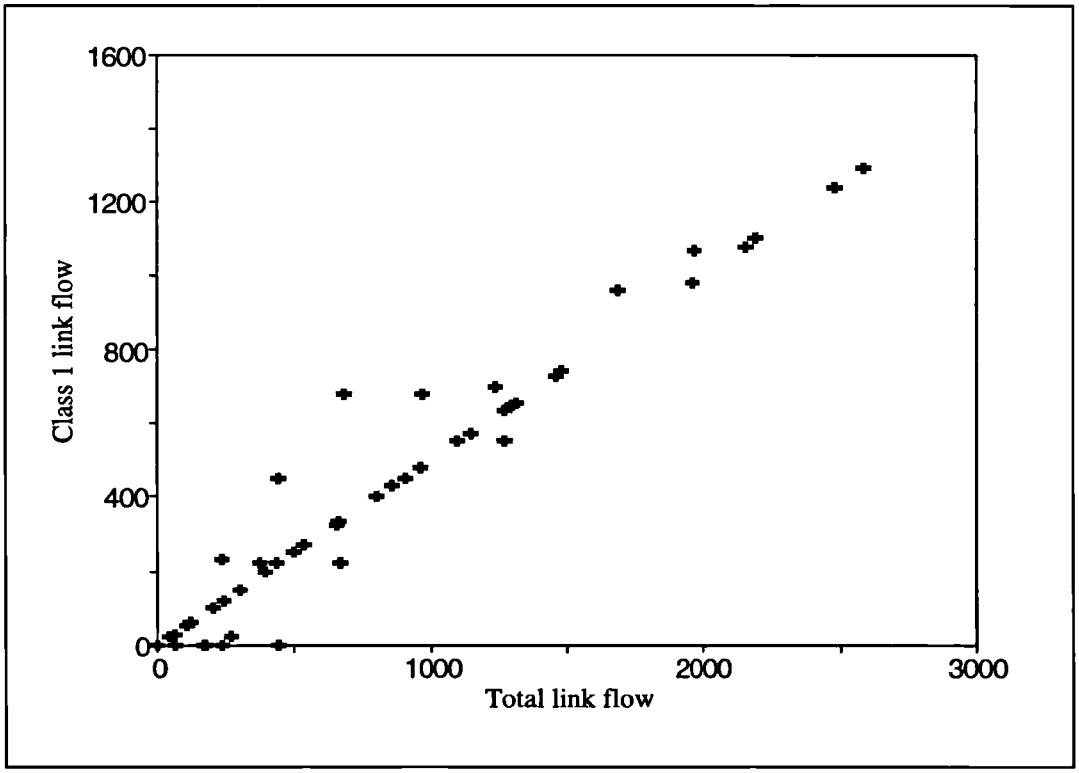


Figure 6-6 Link flows obtained by Algorithm 2 in Charlesworth's network

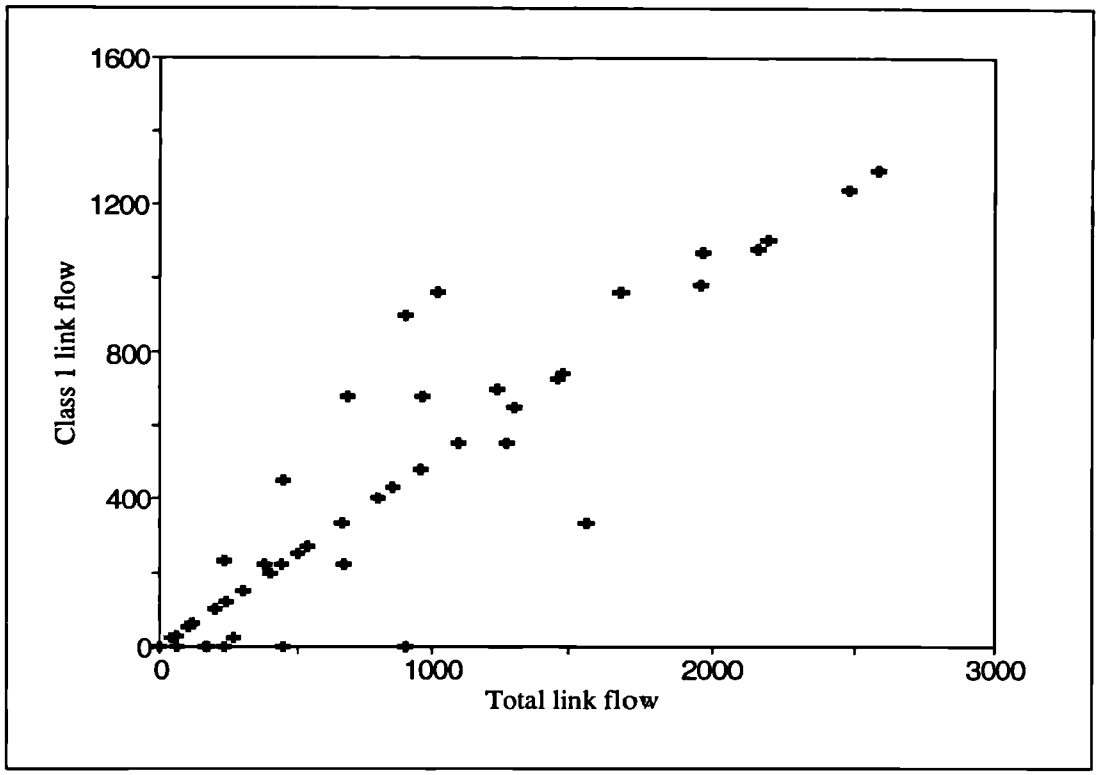


Figure 6-7 Link flows obtained by Algorithm 3 in Charlesworth's network

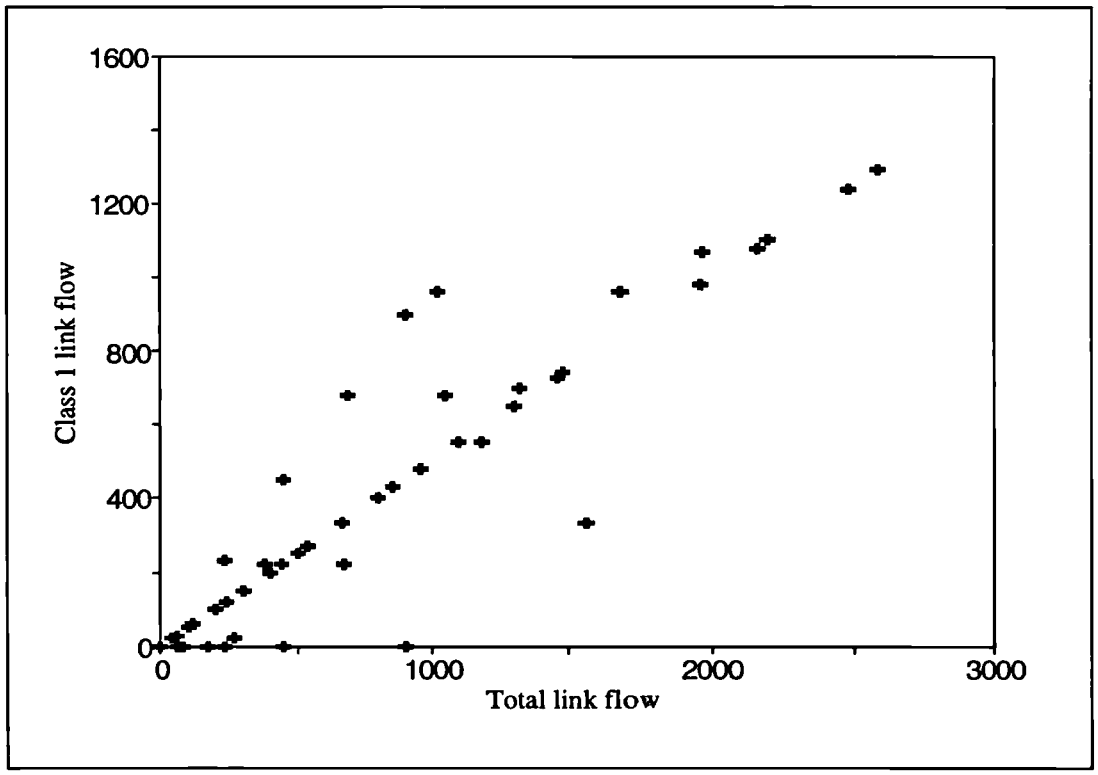


Figure 6-8 Link flows obtained by Algorithm 4 in Charlesworth's network

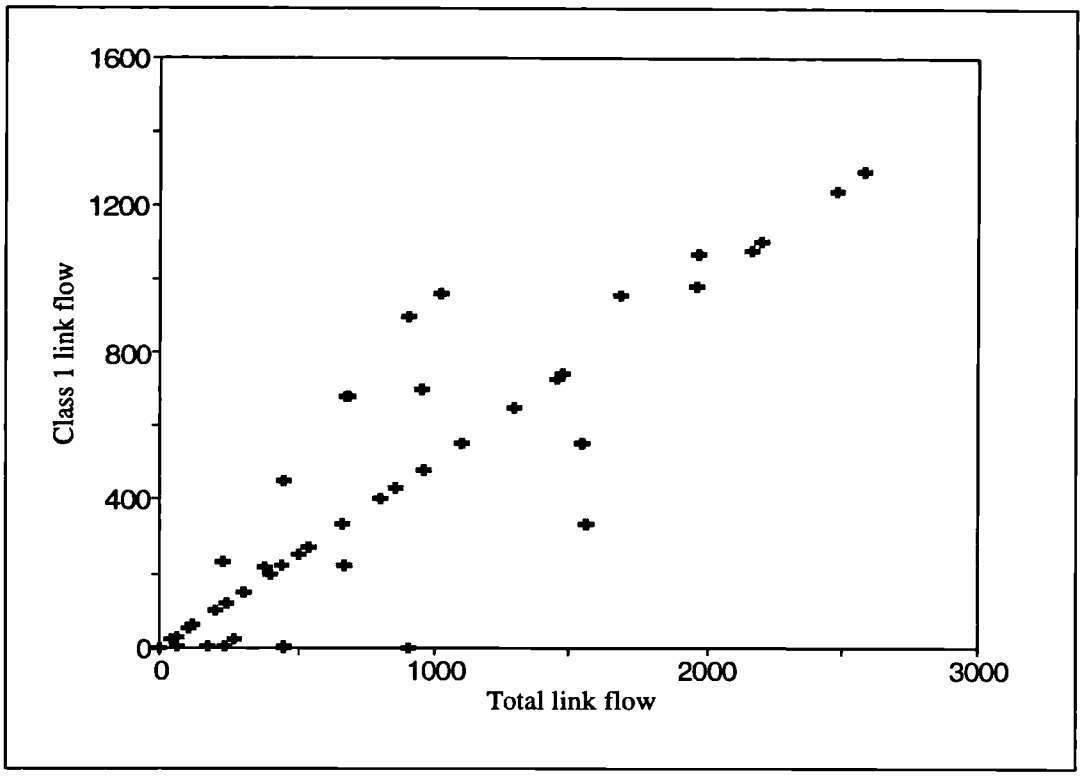


Figure 6-9 Link flows obtained by Frank-Wolfe algorithm in Charlesworth's network

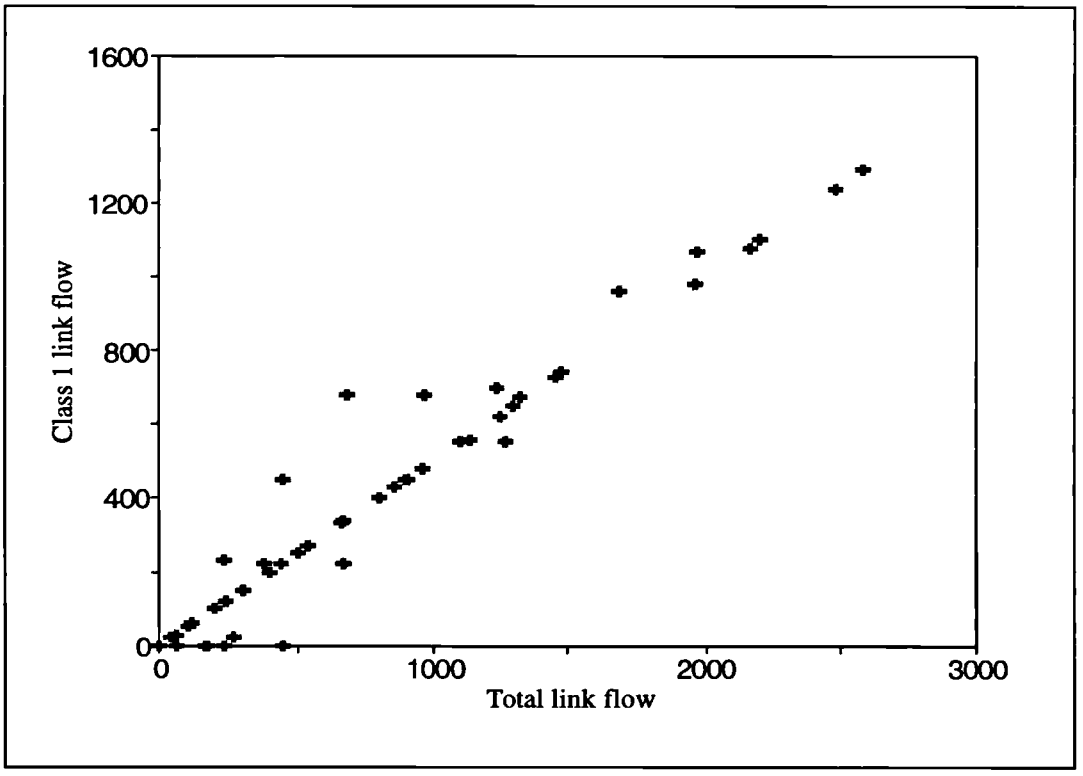


Figure 6-10 Link flows obtained by Schittenhelm algorithm in Charlesworth's network

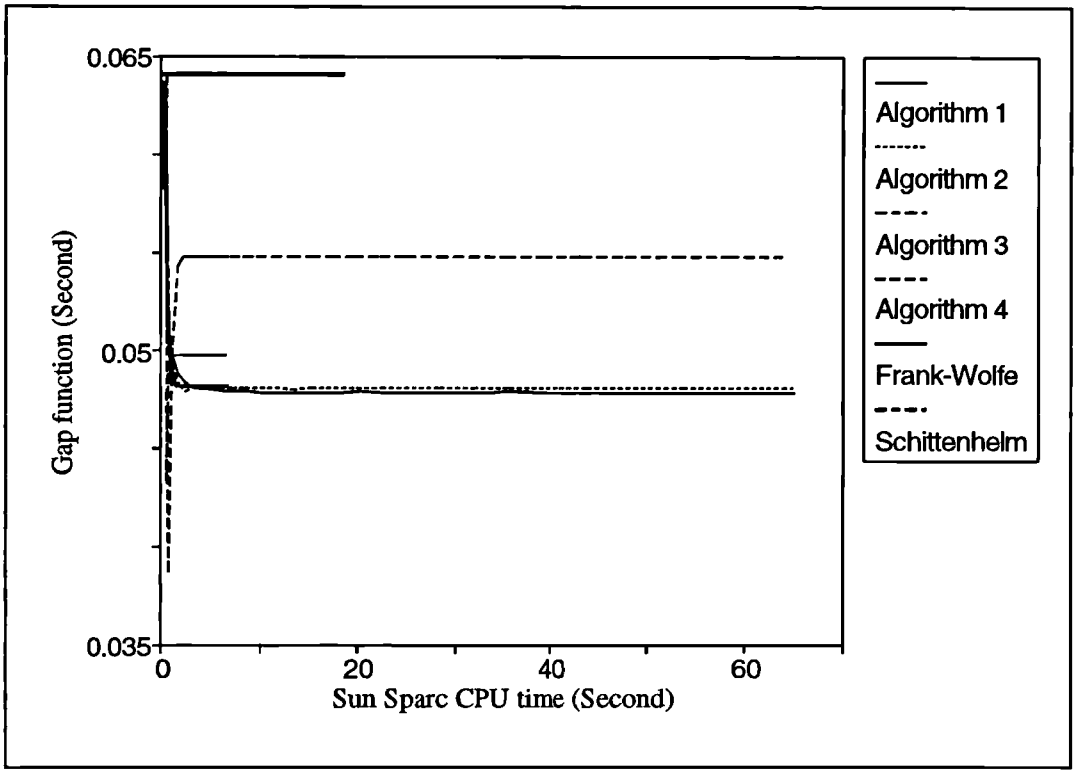


Figure 6-11 The performance of algorithms in Sioux Falls network

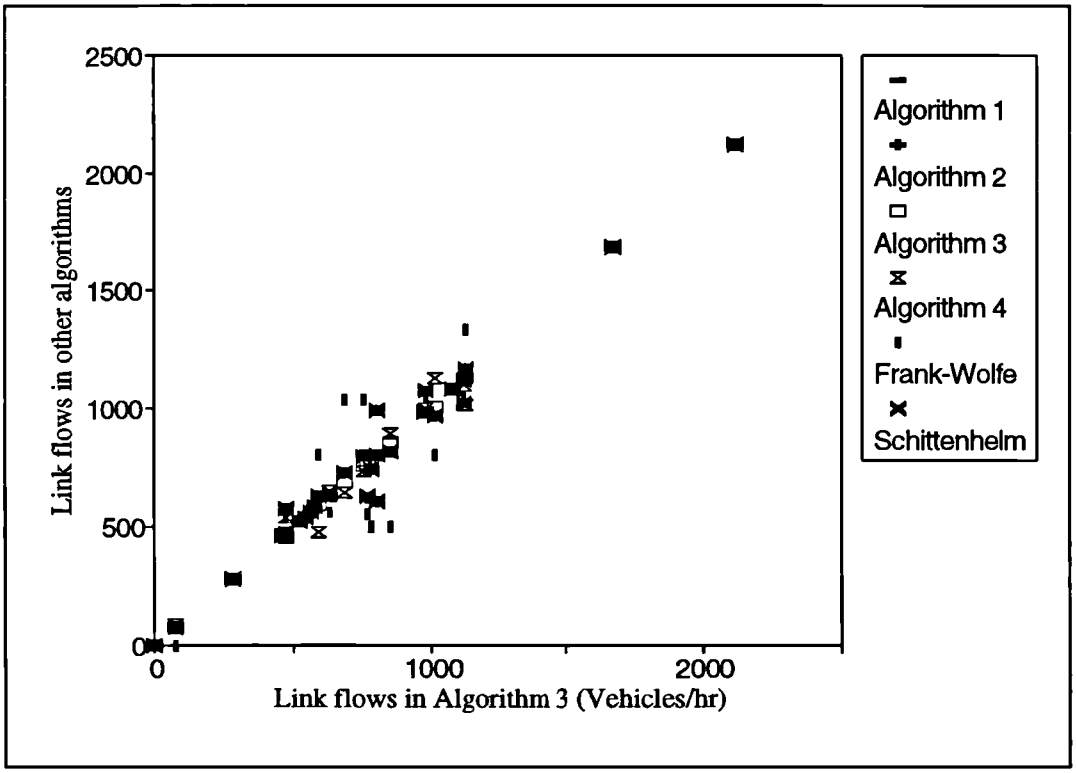


Figure 6-12 The comparison of total link flows obtained by algorithms on Sioux Falls network

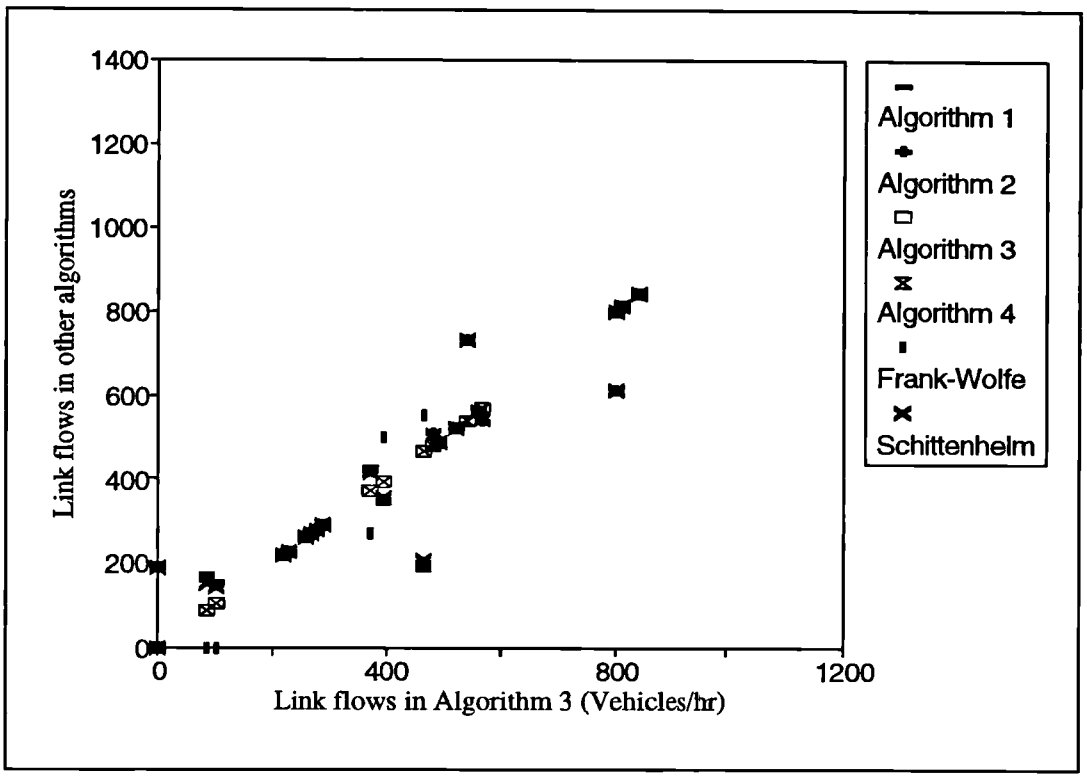


Figure 6-13 The comparison of link flows of class 1 obtained by algorithms on Sioux Falls network

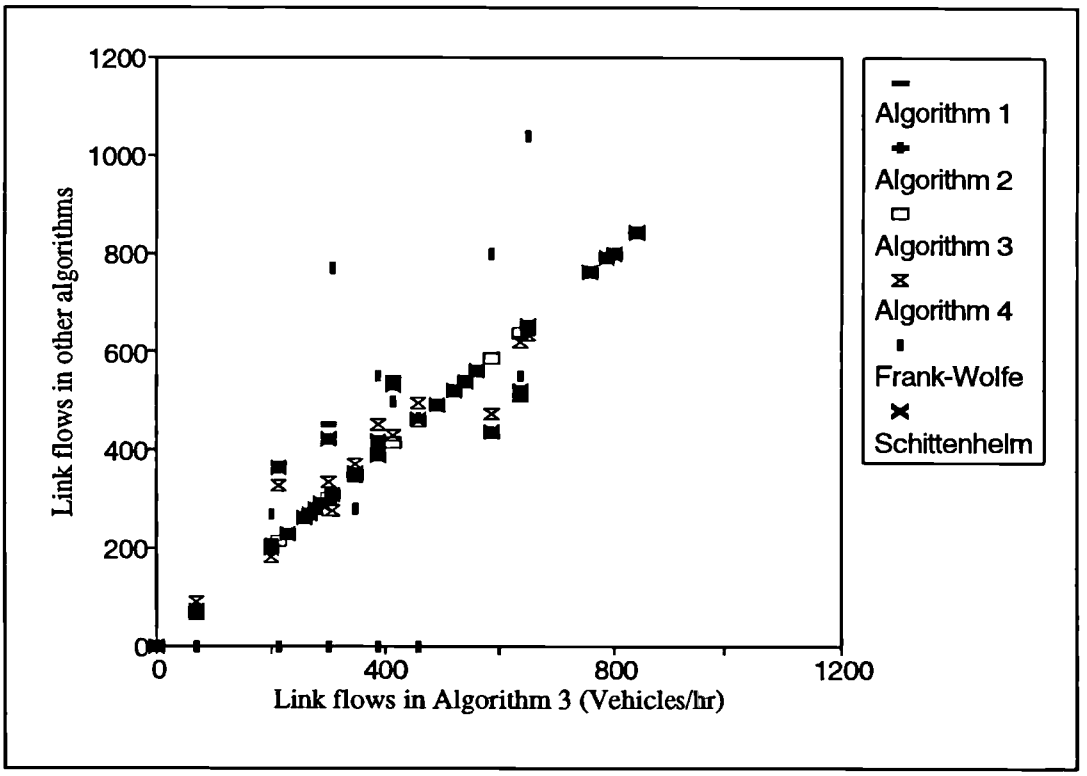


Figure 6-14 The comparison of link flows of class 2 obtained by algorithms on Sioux Falls network

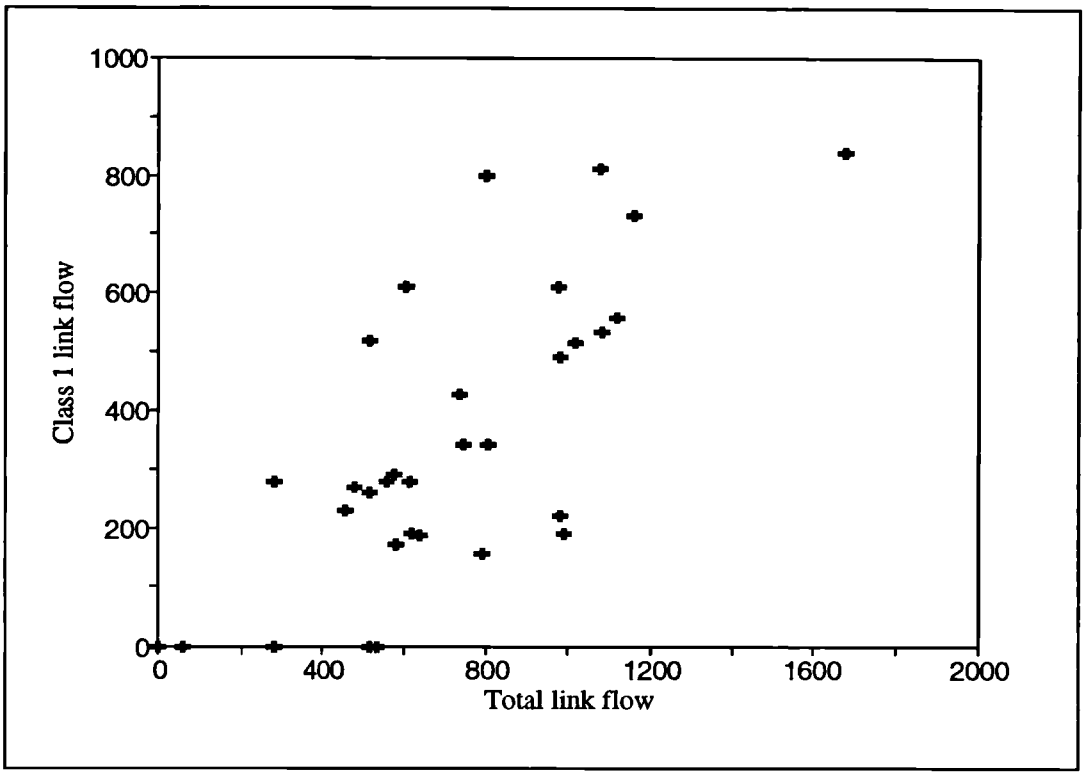


Figure 6-15 Link flows obtained by Algorithm 1 in Sioux Falls network

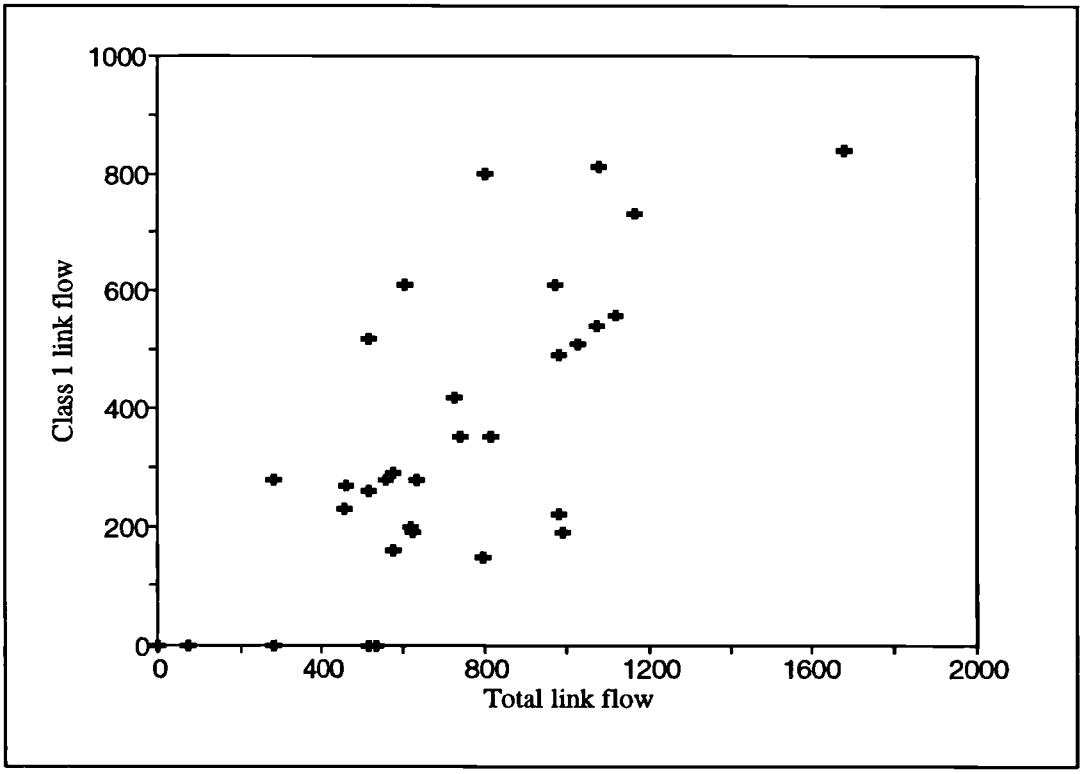


Figure 6-16 Link flows obtained by Algorithm 2 in Sioux Falls network

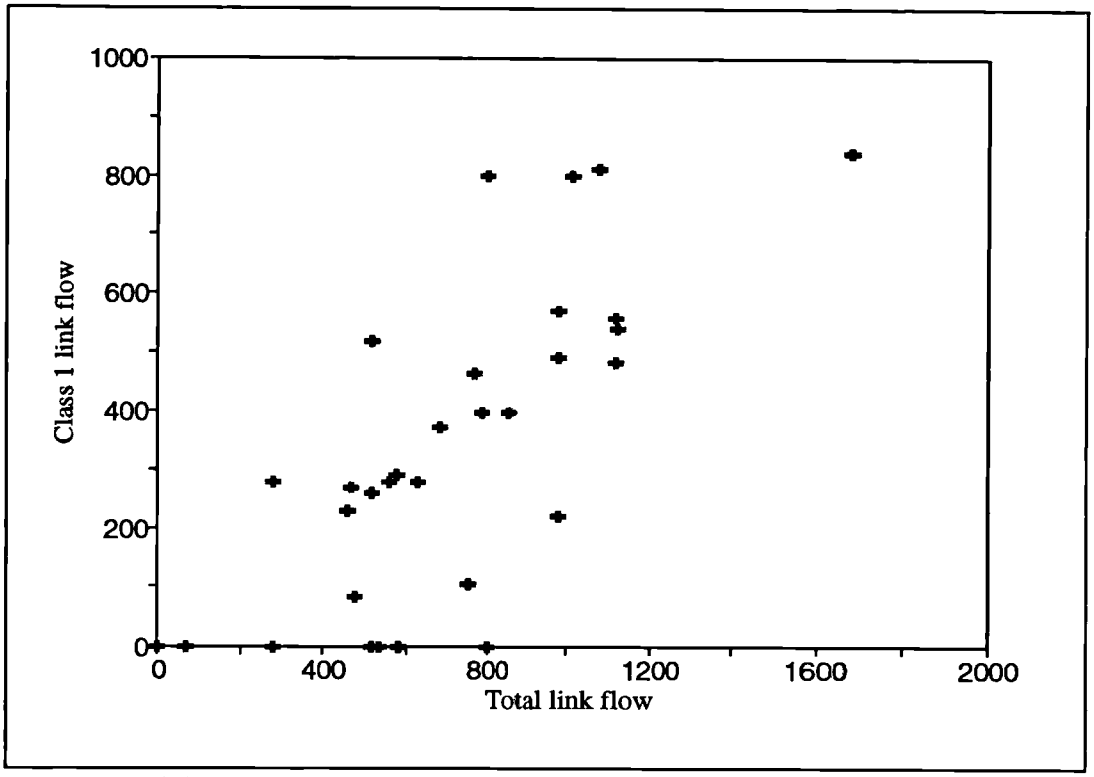


Figure 6-17 Link flows obtained by Algorithm 3 in Sioux Falls network

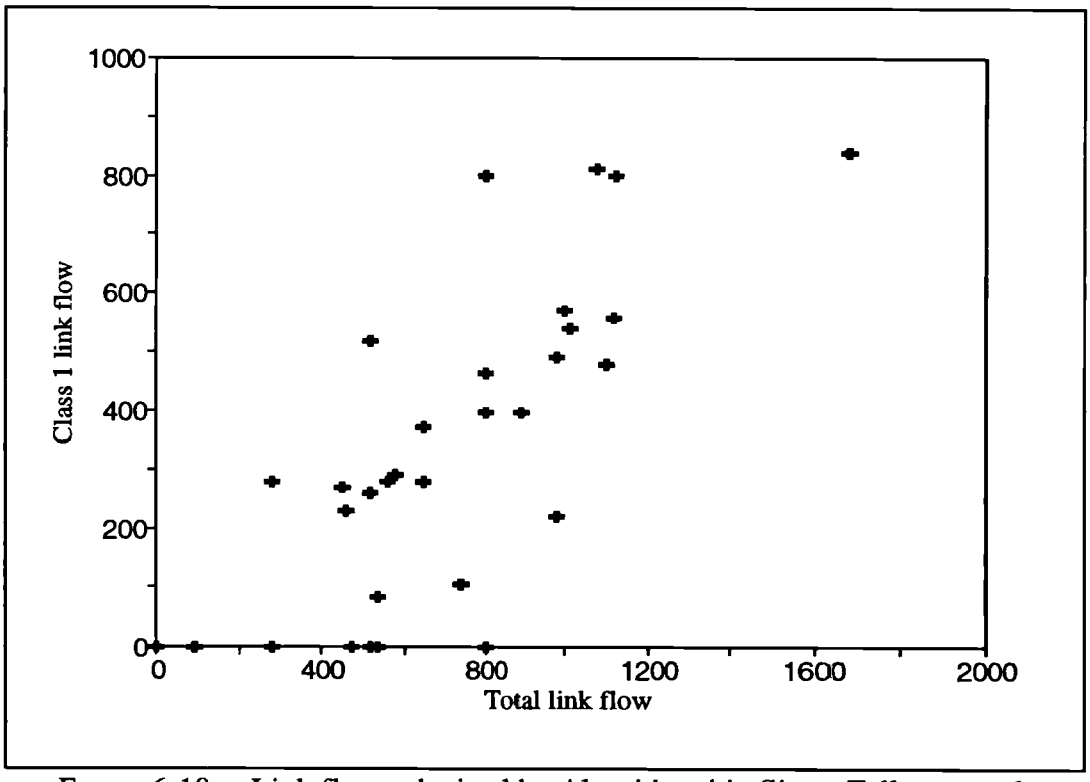


Figure 6-18 Link flows obtained by Algorithm 4 in Sioux Falls network

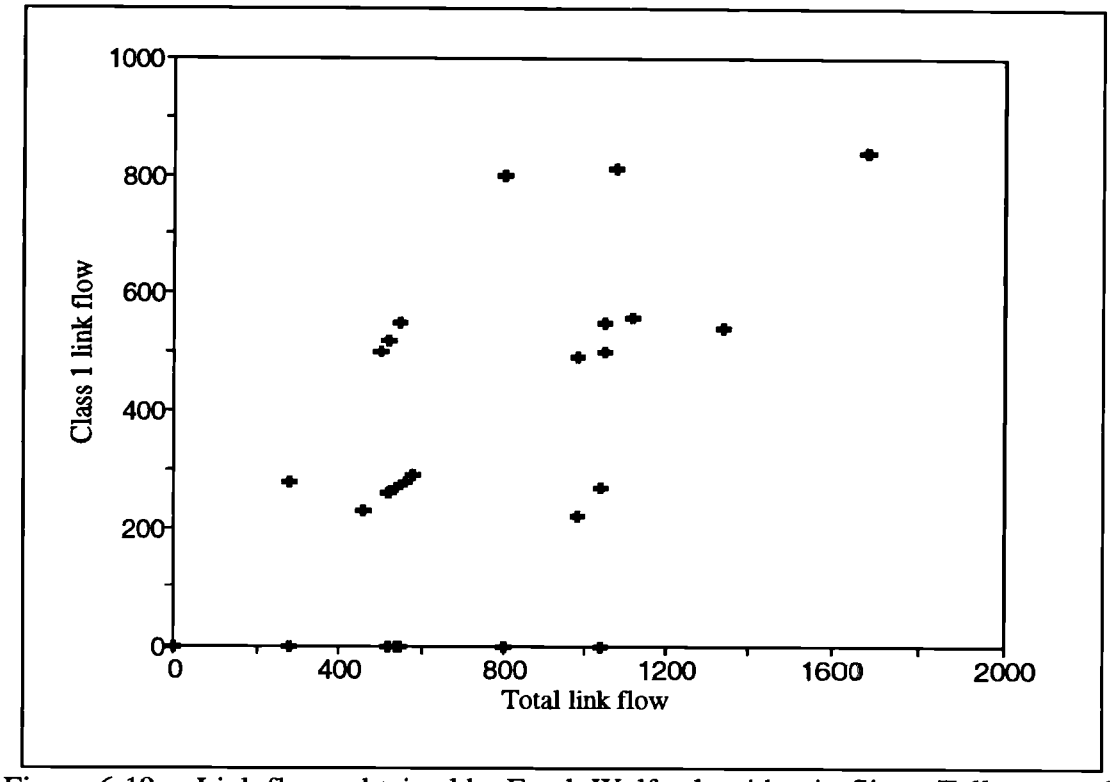


Figure 6-19 Link flows obtained by Frank-Wolfe algorithm in Sioux Falls network

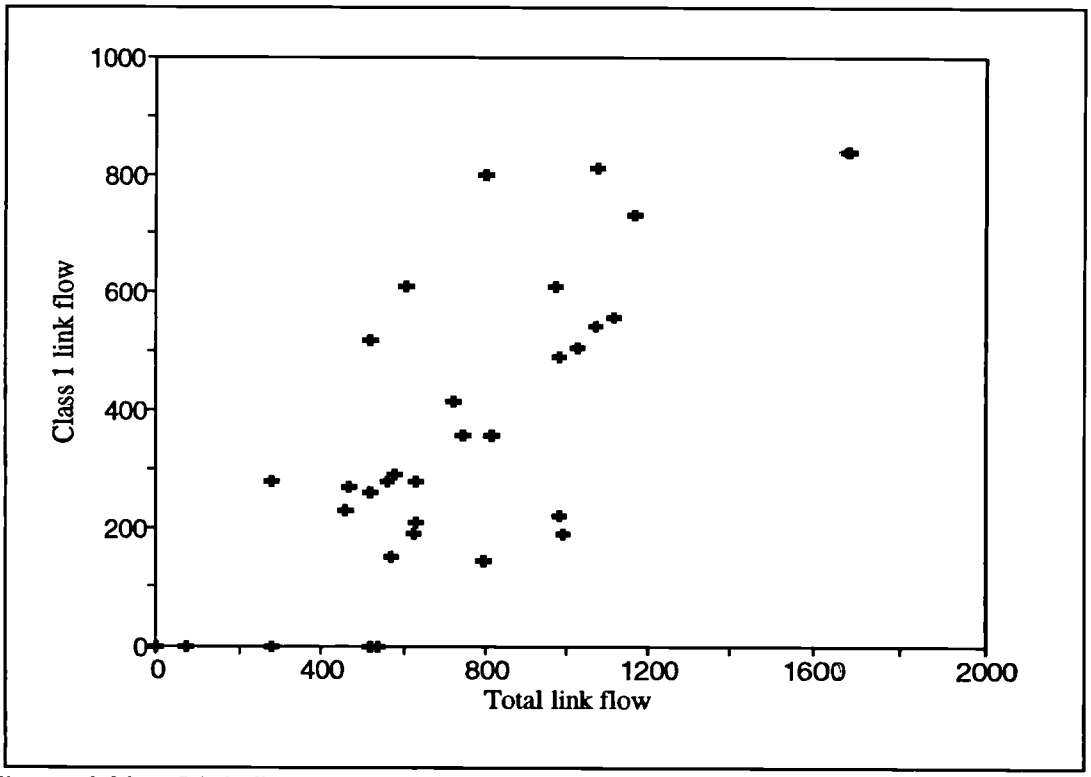


Figure 6-20 Link flows obtained by Schittenhelm algorithm in Sioux Falls network

CHAPTER 7 CONCLUSIONS AND SUGGESTIONS FOR FURTHER STUDIES

7.1 INTRODUCTION

In this study, a general model has been presented for equilibrium road traffic assignment problems with non-separable cost functions. Prime examples of non-separability are the multiclass traffic assignment as a link interaction, and the traffic assignment problem with priority and signal controlled junctions as junction interactions. Modelling of these cases has been chosen as study areas.

The importance of these extensions stems from the recognition that standard models fail to capture important features of traffic congestion by not representing real phenomena on roads. In reality, the performance of the link is affected by vehicles of other classes, oncoming traffic and different priority rules at junctions, and by signal control policies. The different vehicles have different occupancy rates, speeds, values of time and route choice criteria and total cost functions. The model developed here has the potential to play an important role in the estimation of real phenomena to help to develop, manage and evaluate transport planning policies to alleviate traffic congestion.

Four solution methods for the equilibrium road traffic assignment have been presented. These solution algorithms are based upon the simplicial decomposition principle. A diagonalisation solution procedure, which solves a sequence of separable cost problems representing the non-separability property, has been developed on the basis of the simplicial decomposition algorithm in order to use them in the equilibrium road traffic assignment in non-separable cost cases. These developed algorithms performed well in separable and non-separable cases on two example networks. In

particular, Algorithm 3 ran quickly in most cases. However, the other algorithms also performed favourably by comparison with the Frank-Wolfe algorithm, and terminated with further converged values of the objective function.

We have analysed some properties such as existence, uniqueness and stability of solutions using the models developed. Detailed delay formulae, which include topological and geometric characteristics of road junctions, and the effects on the cost function caused by priorities of interacting movements, signal controlled policies and different perceptions of travel costs for different classes, have also been analysed. These were studied in terms of the degree of detailed modelling. In priority controlled junction modelling, a stable unique solution has been obtained in experiments where different initial solutions were used to test whether or not the symmetric equilibrium solution is obtained. By contrast, in signal controlled junction modelling, the corresponding tests resulted in multiple and unstable equilibria. In multiclass traffic assignment, we obtained a unique solution in terms of total link flows, but in terms of each class of link flows, we obtained multiple solutions. The results of these tests have been related to theoretical sufficient and necessary conditions for good behaviour based upon the Jacobian matrix. The summaries of these results are shown in Tables 2-2 and 2-3.

In the section 7.2, summaries and conclusions of this study are presented in terms of chapters. In section 7.3, various suggestions are made for further studies related to this one.

7.2 SUMMARIES AND CONCLUSIONS

In chapter 2, we have presented fundamentals of traffic assignment. Brief historical development of this problem was surveyed in terms of separable and non-separable cases. The traffic assignment problem was explained as interactions between supply and demand. As transportation supply, network representation using graph theory, and zone, centroid, link, node and cost function were described. As travel demand, estimation methods of travel demand were described. Wardrop's principles were explained as demand and supply interactions. Good behaviour conditions, and their properties such as existence, uniqueness, stability and sensitivity were reviewed. Network representation to use traffic assignment was reviewed according to the degree of detail. Formulations of this problem were described as a conventional convex mathematical programme and variational inequality. Kuhn-Tucker conditions were derived to show equivalence of these formulations. Cost functions were reviewed in terms of link cost and junction delay. Link cost functions were further divided into empirical and theoretical approaches. In junction delay, priority and signal control cases were reviewed. Combination of link cost and junction delay was proposed to use in the detailed modelling. Tests for satisfaction of necessary and sufficient conditions for good behaviour were introduced. Solution algorithms were briefly presented for each of the separable and non-separable cases.

In chapter 3, the simplicial decomposition principle was introduced in terms of a linear subproblem and a master subproblem. Existing algorithms including the Frank-Wolfe and the Schittenhelm algorithms were described in terms of this principle. Four novel algorithms were presented and explained. The performance of each of these algorithms was compared using an example of the separable traffic assignment. The new simplicial decomposition Algorithm 1 has been shown to run quickly when solving a separable equilibrium traffic assignment problem. The

advantage of this algorithm is in reducing the number of shortest path searches which are the main time-consuming part of solving the traffic assignment problem. In terms of furthest convergence point, Algorithm 4 reaches 3863 vehicle-seconds whilst Algorithms 2 and 3 terminate at 3876 and 3878 vehicle-seconds respectively. Algorithm 3 performs very well to terminate 3878 vehicle-seconds when two column generations are used initially. When fewer than two column generations are used, Algorithm 3 terminates to 3920 vehicle-seconds. Algorithm 4 has shown no evidence to speed up when column generation is used in the initial stage.

In chapter 4, the traffic assignment problem with priority controlled junction modelling was presented in terms of capacity calculation, cost function analysis, formulation and numerical analysis. In capacity calculations, the effects of geometry in junctions and priority rules were explained. The TRL PICADY capacity formulae were used to calculate the minor capacities. In cost function analysis, the steady-state Pollaczek-Khinchine formula and an extended version of it were used to calculate the junction delay for the steady-state case and an initial overloaded case due to all-or-nothing assignment respectively. The Jacobian matrices for good behaviour were analysed. Some properties such as uniqueness and stability were investigated using a small example network. In this analysis, we have found that the priority controlled modelling in traffic assignment is globally stable. In other words, irrespective of initial solutions, it converged to a symmetric equilibrium value. Numerical examples are analysed in terms of the efficiency of solution methods. Algorithm 3 performed best in most cases but Algorithms 1, 2 and 4 terminated further in terms of gap values. The sensitivity of delay functions was tested by changing the weighting factor which combine link and junction delays. In general, the final gap values decrease as increasing emphasis was placed on link delays. The pattern of convergence in all algorithms except the Frank-Wolfe included fluctuations in gap. In particular, when they are applied to the Charlesworth example network, the pattern of convergence

included more fluctuations than in the Sioux Falls example network. The comparison of final traffic flows and capacities on minor links resulting from each of algorithms showed that some results are quite similar whilst others are not whether any attention is paid to junction delays. In particular, the results from the new algorithms are matched well among them but are separated from those of the Frank-Wolfe. However, when only link delays are considered, the final flows and capacities agree closely because a zero value of gap is achieved by all the algorithms except the Frank-Wolfe.

In chapter 5, the traffic assignment with signal controlled junction modelling has been presented in terms of historical developments, signal optimisation problems, cost function analysis, formulation, and numerical analysis. In signal optimisation, existing methods were reviewed and parameters of optimisation were explained. In particular, Webster's green time calculation method was explained. In cost function analysis, Webster's steady-state delay formula and an extended version of it were used to calculate the junction delay for steady-state case and an initial overloaded case due to all-or-nothing assignment respectively. Jacobian matrices were analysed in terms of good behaviour condition. Some properties such as uniqueness and stability were investigated using a small example network. In this analysis, we have found that when signal control is included in traffic assignment model, neither local nor global stability were satisfied. However, using good initial solutions, the assignment process converged to a symmetric equilibrium value. Numerical examples are analysed in terms of the efficiency of solution methods. Algorithm 3 performed best in most cases but in some Algorithms 1, 2 and 4 terminated to smaller values of gap. The sensitivity of the cost function was tested by changing the weighting factor which combine link and junction delays. In general, the final gap values decrease as increasing emphasis was placed on link delays. The pattern of convergence in all algorithms except the Frank-Wolfe included fluctuations in gap. In particular, when they are applied to the Charlesworth example network, the pattern of convergence included more fluctuations

than in the Sioux Falls example network. The comparison of final traffic flows and green times resulting from each of algorithms showed that some results are quite similar whilst others are not whether any attention is paid to junction delays. However, when only link delays are considered, the final flows and green times agree closely because a zero value of gap is achieved by all the algorithms except the Frank-Wolfe.

In chapter 6, a multiclass traffic assignment was presented in terms of historical developments, concepts of multiclass, cost function analysis and formulation and numerical analysis. This model represents an influential relationship between multiple vehicle classes. Travel units of the multiclass assignment consist of two classes, each of which has an individual cost function and influences other class's cost function. The multiclass traffic assignment was described in terms of extension to the supply and demand model. A variational inequality formulation to represent the extended Wardrop's condition was proposed. Cost functions for multiple classes were specified and described. The Jacobian matrix of this problem was analysed to study properties such as stability and uniqueness. This analysis showed that this problem can be formulated so as to satisfy at least the necessary condition for good behaviour. In a small example network, we have found that there was a unique equilibrium solution in terms of total link flows but multiple equilibria in terms of each class link flows. The computing method used in the algorithm was developed by assuming that each class of road users has an individual copy of the network and the single problem uses only the network belonging to it. However, the way of calculating the individual network is included the interaction that the cost of using a link not only depends on the volume on that link but also on the corresponding link of each other networks. Extended simplicial decomposition algorithms are developed in this framework. Numerical examples were analysed in terms of the efficiency of solution methods and good convergence behaviour. Algorithm 3 performed best and others also performed

favourably by comparison of the Frank-Wolfe. The pattern of convergence has a fluctuations in the initial stage and then in later stage stabilised.

Overall, the new algorithms for the various models developed have performed favourably by comparison with existing algorithms on two example networks. A small example network has been used to investigate existence, uniqueness and stability properties using the models. In the priority controlled model, a unique stable solution has been obtained whilst in signal controlled model, multiple and unstable solutions have been obtained. In the multiclass model, a unique solution has been obtained in terms of the total link flow whilst multiple solutions have been obtained in terms of flow in each class. These findings correspond well with the nature of the models assumed because we can classify the degree of non-separability according to the order of priority controlled model, multiclass model and signal controlled model. In Table 2-3, hypotheses were made for these results and they were showed in Chapters 4, 5 and 6.

We can conclude that new simplicial decomposition algorithms are of immediate importance because they outperform existing algorithms, are accurate, and these give many choices to analyse transport systems. We can also conclude that detailed modelling is an indispensable tool for managing and operating transport systems. However, we have to use these modelling with care because the Jacobian matrix analysis of these models shows that they do not usually satisfy the sufficient condition for good behaviour and thus can have multiple solutions: in such cases, the algorithms developed here will calculate at most one such solution and hence can be unreliable.

7.3 SUGGESTIONS FOR FURTHER STUDIES

A variety of networks should be tested further. In addition, an existing theory shows that if even one junction delay function in a network does not satisfy the monotonicity property, the whole network could fail to have a unique and stable solution. The validity of this theory should be studied in terms of network topology and the degree of detailed modelling. We also note that link and junction delays have different effects on the cost function. This different magnitude of costs by changing flows should be considered further in the sensitivity analysis.

The trade off between the detailed non-separable cost modelling and conventional separable cost modelling should be analysed in terms of the accuracy of estimation and the implementing cost of the enhanced model. Data preparation of modelling becomes costly according to the degree of detailed modelling. In the analysis of urban congestion, detailed junction modelling gives more accurate estimation of flows and more useful tool of traffic management and operation. On the other hand, data requirements become huge and costly if we use detailed modelling. We therefore choose a model according to the purpose and scope of studies.

These detailed models can be improved by incorporating more realistic components. In priority controlled modelling, we can include a roundabout case to give the model more general uses. In the signal controlled case, we can include more parameters of signal optimisation such as offsets for co-ordinated road networks. In the multiclass case, we can include the effects of route guidance by representing users of route guidance equipment separately. To use these models in traffic operation and management more powerfully, we can also include road pricing techniques.

REFERENCES

AASHTIANI, H.Z. AND T.L. MAGNANTI (1981) Equilibria on a congested transportation network. SIAM journal on Algebraic and Discrete Methods 2,213-226.

ABDUAAL, M., AND L.J. LEBLANC (1979) Methods for combining modal split and equilibrium assignment models. Transportation Science Vol13(4), 292-314.

ALLSOP, R.E. (1971) Delay-minimising settings for fixed time traffic signals at a single road junction. J. Inst. Math. Its Appl. 8, 164-85.

ALLSOP, R.E. (1972) Estimating the traffic capacity of a signalised road junction. Transp. Res. 6, 245-55.

ALLSOP, R.E. (1992) Inroduction to modelling for the design and evaluation of area-wide traffic management schemes. PTRC Course, London, June.

ALLSOP, R.E. AND J.A. CHARLESWORTH (1977) Traffic in a signal controlled road network. Traffic Eng. Control 18, 262-4.

AREZKI, Y. AND D. VAN VLIET (1990) A full analytical implementation of the PARTAN/Frank-Wolfe algorithm for equilibrium assignment. Transp. Sci. 24(1), 58-62.

BECKMANN, M, C. MCGUIRE AND C.B. WINSTEN (1956) Studies in the Economics of Transportation, Yale University Press, New Haven.

BELLMAN, R. (1957) Dynamic programming. Princeton University Press.

BERTSEKAS, D.P. AND E.M. GAFNI (1982) Projection methods for variational inequalities with application to the traffic assignment problem. Mathematical Programming Study 26.167-196.

BOVY, P.H.L. AND G.R.M. JANSEN (1983) Network aggregation effects upon equilibrium assignment outcomes. Transp. Sci. 17(3), 240-62.

BRAESS, D AND G. KOCH (1979) On the existence of equilibria in asymmetrical multiclass user transportation networks, Transportation Science 13, 56-63.

BRANSTON, D. (1976) Link capacity functions: a review. Transp. Res. 10, 223-6

CAMPBELL, E.W., L.E. KEEFER AND R.W. ADAMS (1959) A method for predicting speeds through signalised street sections. Highway Research Board Bulletin 230, 112-5.

CANTARELLA, G.E., A. IMPROTA AND A. SFORZA (1991) A procedure for equilibrium network traffic signal setting. Transp. Res. A 25.

CATCHPOLE, E.A. AND A.W. PLANK (1986) The capacity of a priority intersection. Transp. Res. B 20, 441-56.

CHAO, G.S. AND T.L. FRIESZ (1984) Spatial price equilibrium sensitivity analysis. Transp. Res. 18B(6), 423-40.

CHARLESWORTH, J.A. (1977) The calculation of mutually consistent signal settings and traffic assignment for a signal-controlled road network. Proceedings of the 7th International Symposium on Transportation and Traffic Theory, Tokyo.

COBA (1989) COBA Manual 9, Department of Transport in U.K.

DAFERMOS, S.C. (1971) An extended traffic assignment model with application to two-way traffic. Transportation Science Vol 5 (4),106-118.

DAFERMOS, S.C. (1972) The traffic assignment problem for multiclass user transportation networks, Transportation Science.367-389.

DAFERMOS, S.C. (1981) The general multimodal network equilibrium problem with elastic demand. Networks, 56-72.

DAFERMOS, S.C. (1982) Relaxation algorithms for the general asymmetric traffic equilibrium problem. Transportation Science Vol 16(2), 231-240.

DAFERMOS, S.C. AND A. NAGURNEY (1982) Sensitivity analysis for the asymmetric network equilibrium problem. Math. Programm. 28(2), 191-217.

DAFERMOS, S.C. AND F.T. SPARROW (1969) The traffic assignment problem for a general network. Journal of Research of the National Bureau of Standards 73B,91-118.

DAGANZO, C.F. (1983) Stochastic network equilibrium with multiple vehicle types and asymmetric, indefinite link cost Jacobians. Transportation Science 17, 282-300.

DANTZIG, G.B. AND P. WOLFE (1960) The decomposition algorithm for linear programming. Operations Research 8.101-111.

DEMBO, R.S. AND U. TULOWITTZKI (1988) Computing equilibria on large multicommodity networks: an application of truncated quadratic programming algorithms. Networks 18,273-284.

DIJKSTRA, E.W. (1959) Note on two problems in connection with graphs. Numer. Math.,1, 269-271.

EAVES, B.C. (1971) On the basic theorem of complementarity. Math. Progrmm.1. 68-75.

FISK, C.S. (1984) Optimal signal controls on congested networks. Proc. of the 9th Int. Sym. on Transp. and Traffic Theory. 197-216.

FISK, C.S. AND S. NGUYEN (1981) Existence and uniqueness properties of an asymmetric two-mode equilibrium model. Transportation Science 15, 318-329.

FISK, C.S. AND S. NGUYEN (1982) Solution algorithms for network equilibrium models with asymmetric user costs. Transportation Science Vol16(3), 361-381.

FLORIAN, M. (1977) An improved linear approximation algorithm for the network equilibrium problem. Proc. of 1977 IEEE conf. on Decision and Control, 812-8.

FLORIAN, M. AND H. SPIESS (1982) The convergence of diagonalisation algorithms for asymmetric network equilibrium problems. Transp. Res. 16B(6), 447-83.

FRANK, M. AND P. WOLFE (1956) An algorithm for quadratic programming. Naval Research Logistics Quarterly 3,95-110.

FUKUSHIMA, M. (1984) A modified Frank-Wolfe algorithm for solving the traffic assignment problem. Transportation Research 18B,169-177.

GALLIVAN, S. AND B.G. HEYDECKER (1988) Optimising the control performance of traffic signals at a single junction. Transp. Res., 22B (5), 357-70.

GARTNER, N.H., S.B. GERSHIN, J.D.C. LITTLE AND P. ROSS (1980) Pilot study of computer-based urban traffic management. Transp. Res. B 14, 203-17.

HARKER, P.T. AND J.S. PANG (1990) Finite-dimensional variational inequality and nonlinear complementarity problems: a survey of theory, algorithms and applications. Math. Progrmm. 48, 161-220.

HEARN, D.W. (1982) The gap function of a convex program. Oper. Res. Let. 1. 67-71.

HEARN, D.W., S. LAWPHONGPANICH AND S. NGUYEN (1984) Convex programming formulations of the asymmetric traffic assignment problem. Transportation Research 18B,357-365.

HEYDECKER, B.G. (1983) Some consequences of detailed junction modelling in road traffic assignment. Transportation Science, 17(3), 263-81.

HEYDECKER, B.G. (1990) Junction interactions in capacity-restrained traffic assignment. Note, University College London, unpublished.

HEYDECKER, B.G. AND I.W. DUDGEON (1987) Calculation of signal settings to

minimise delay at a junction. Proc. of the 10th Int. Sym. on Transp. and Traffic Theory,159-78.

HEYDECKER, B.G. AND T.K. KHOO (1990) The equilibrium network design problem. the 22nd Conference of Universities Transport Studies Group, January.

HCM (1985) Highway Capacity Mannual, SR 209, Transp. Res. Board.

HOLLOWAY, C.A. (1974) An extension of the Frank and Wolfe method of feasible directions, Mathematical Programming 6,14-27.

HOROWITZ, J.L. (1984) The stability of stochastic equilibrium in a two-link transportation network. Transp. Res. 18B(1), 13-28.

IRWIN, N.A., N. DODD AND H.G. VON CUBE (1961) Capacity restraint in assignment programs. Highway Res. Board Bulletin, 297, 109-27.

JANSON, B.N. AND ZOZAYA-GOROSTIZA,C. (1987) The problem of cyclic flows in traffic assignment. Transportation Research 21B,4,299-310.

KARAMARDIAN, S. (1969) The nonlinear complemetarity problem with applications, part I and II. Journal of Optimisation Theory and Applications 4, 87-98 and 167-181.

KIMBER, R. M. AND COOMBE, R.D. (1980) The traffic capacity of major/minor priority junctions. Transport Research Laboratory, SR 582.

KIMBER, R.M. AND HOLLIS, E.M. (1979) Traffic queues and delays at road junctions. Transport Research Laboratory, SR 810.

KIMBER, R.M., I. SUMMERSGILL AND I.J. BURROW (1986) Delay processes at unsignalised junctions. Transp. Res. B 20, 457-76.

LARSSON, T AND M. PATRIKSSON (1992) Simplicial decomposition with disaggregated representation for the traffic assignment problem. Transportation Science 26(1),4-17.

LAWPHONGPANICH, S. AND D.W. HEARN (1984) Simplicial decomposition of the asymmetric traffic assignment problem. Transportation Research 17B,123-133.

LEE, S. (1990) A study on stochastic user equilibrium assignment. Master thesis, Seoul National University.

LEE, S. (1992a) A simplicial decomposition algorithm to solve the traffic assignment problem. Paper presented to the 24th Conference of Universities Transport Studies Group, January.

LEE, S. (1992b) Novel algorithms for solving the traffic assignment problem, PTRC

European Transport, Highways and Planning 20th Summer Annual Meeting, Seminar **E**, 221-232.

LEE, S. (1992c) A simplicial decomposition algorithm to solve the traffic assignment problem with non-separable costs. Transfer Report to PhD, Transport Studies Group, University College London, Unpublished.

LEE, S. (1993a) Detailed junction modelling in road traffic assignment. Paper presented to the 25th UTSG conference, Southampton, January.

LEE, S. (1993b) Mathematical programming algorithms for equilibrium road traffic assignment problems. The Korean Federation of Science and Technology Society, August, 818-826.

LEE, S. (1993c) An analysis of detailed junction modelling in road traffic assignment. PTRC European Transport, Highways and Planning 21st Summer Annual Meeting, Seminar **D**, 295-306.

LEE, S., K. S. CHON AND K. W. LIM (1990) Stochastic user equilibrium assignment problems. J. of Transp. Res. Soc. of Korea, Vol 8(1), 55-72.

LEMKE, C.E. (1965) Bimatrix equilibrium points and mathematical programming. Management Science 11,681-689.

LEVENTHAL, T.G. NEMHAUSER AND L. TROTTER (1973) A column generation algorithm for optimal traffic assignment. Transportation Science 7,168-176.

LITTLE, J.D.C. (1966) The synchronisation of traffic signals by mixed integer linear programming. Oper. Res. 14, 568-94.

LOUAH, G. (1991) Priority intersection: modelling. In Concise Encyclopedia of Traffic and Transp. Systems edited by Papageorgiou. 331-35.

LUENBERGER, D. (1989) Linear and nonlinear programming. Addison-Wesley,Reading.

MAGNANTI, T. L. (1984) Models and algorithms for predicting urban traffic equilibria. Transportation Planning Models Edited by M.Florian, Elsevier Publication.

MARCOTTE, P. (1983) Network optimisation with continuous control parameters. Transp. Sci. 17, 181-97.

MINTY, G.J. (1962) Monotone operators in Hilbert space. Duke Math. Jou. 29, 341-46.

MOSHER, W.W. (1963) A capacity restraint algorithm for assigning flow to a transportation network.

NGUYEN, S. AND C. DUPUIS (1984) An efficient method for computing traffic equilibria in network with asymmetric transportation costs. Transp. Sci. 18(2), 185-202.

OVERGAARD, K.R. (1967) Urban transportation planning: traffic estimation. Traffic Quaterly, 197-218.

PANG, J.S. AND D. CHAN (1982) Iterative methods for variational and complementarity problem. Math. Progmm. 24, 284-313.

PANG, J.S. AND C.S. YU (1984) Linearized simplicial decomposition methods for computing traffic equilibria on networks. Networks 14,427-438.

PLANK, A.W. (1982) The capacity of a priority intersection-two approaches. Traffic Eng. Control 23, 88-92.

PLANK, A.W. AND E.A. CATCHPOLE (1984) A general capacity formula for an uncontrolled intersection. Traffic Eng. Control 25, 327-9.

ROCKAFELLAR, R.T. (1970) Convex analysis. Princeton Univ. Press, NJ,155-170.

SACHER, R.S. (1980) A decomposition algorithm for quadratic programming. Mathematical Programming 18,16-30.

SCHITTENHELM, H. (1990) On the integration of an effective assignment algorithm with path and path-flow management in a combined trip distribution and traffic assignment algorithm. PTRC European Transport, Highways and Planning 18th Summer Annual Meeting, Seminar E, 203-214.

SEMMENS, M. C. (1985) PICADY 2 : An enhanced program to model capacities, queues and delays at major and minor priority junction. Transport Research Laboratory, RR **36**.

SHEFFI, Y. (1985) Urban transportation networks: Equilibrium analysis with mathematical programming methods. Prentice Hall,New Jersey.

SHEFFI, Y AND W. B. POWELL (1983) Optimal signal settings over transportation networks. J. Transp. Eng. 109, 824-39.

SHETTY, C.M. AND M. BEN DAYA (1988) A decomposition procedure for convex quadratic programs. Naval Research Logistics Quarterly 35,111-118.

SMITH, M.J. (1979) The existence, uniqueness and stability of traffic equilibria. Transportation Research 13B,295-304.

SMITH, M.J. (1981) A theoretical study of traffic assignment and traffic control. Proceedings of the 8th International Symposium on Transportation and Traffic Theory, Toronto.

SMITH, M.J. (1982) Junction interactions and monotonicity in traffic assignment.

Transportation Research **16B**(1),1-3.

SMITH, M.J. (1983) An algorithm for solving asymmetric equilibrium problems with a continuous cost-flow function. Transportation Research 17B(5),365-371.

SMITH, M.J. (1984) The stability of a dynamic model of traffic assignment-an application of a method of Lyapunov. Transp. Sci. 18(4), 385-94.

SMITH, M.J., VAN VUREN, T., HEYDECKER, B.G. AND VAN VLIET, D. (1987) The interaction between signal-control policies and route choice. Proceedings of the 10th International Symposium on Transportation and Traffic Theory. Cambridge, Mass.

SMITH, M.J. AND T. VAN VUREN (1993) Traffic equilibrium with responsive traffic control. Transp. Sci. 27(2), 118-132.

SMOCK, R.J. (1962) An iterative assignment approach to capacity restraint on arterial networks. Highway Res. Board Bulletin, 347, 60-66.

SOLTMAN, T.J. (1965) Effects of alternate loading sequences on results from Chicago trip distribution and assignment model. Highway Res. Board, 114, 122-140.

TAHA, H. (1984) Operations research. Macmillan Publishing Co.

TAN, H., S.B. GERSHWIN, M. ATHANS (1979) Hybrid optimisation in urban traffic networks. Massachusetts Institute of Technology Report No. DOT-TSC-RSP-79-7.

TANNER, J.C. (1962) A theoretical analysis of delays at an uncontrolled intersection. Biometrika 49, 163-70.

TAYLOR, N.B. (1990) CONTRAM 5: an enhanced traffic assignment model. TRRL Research Report RR 249, Crowthorne: Transport and Road Research Laboratory.

TOBIN, R.L. (1986) Sensitivity analysis for variational inequalities. J. Opt. Theor. Appl. 48(1), 191-204.

TRAFFIC RESEARCH CORPORATION (1966) Winnipey Area Transportation Study. Technical Report.

U.S. BUREAU OF PUBLIC ROADS (1964) Traffic assignment manual. U.S. Department of Commerce, Washington, D.C.

VAN VLIET, D. (1982) SATURN-a modern assignment model. Traffic Engineering and Control 23, 578-581.

VAN VLIET, D. (1987) The Frank-Wolfe algorithm for equilibrium traffic assignment viewed as a variational inequality. Transp. Res. 21B(1), 87-9.

VAN VLIET, D., T. BERGMAN, W.H. SCHELTES (1986) Equilibrium traffic assignment with multiple user classes, PTRC European Transport, Highways and Planning 13rd Summer Annual Meeting, Seminar E. 1-12.

VON HOHENBALKEN, B. (1975) A finite algorithm to maximise certain pseudoconcave functions on polytopes. Mathematical Programming 9,189-206.

VON HOHENBALKEN, B. (1977) Simplicial decomposition in nonlinear programming algorithms. Mathematical Programming 13,49-68.

VYTHOULKAS, P.C. (1990) A dynamic stochastic assignment model for the analysis of general networks. Transportation Research 24B(6), 153-469.

WARDROP, J.G. (1952) Some theoretical aspects of road traffic research. Proceedings of the Institution of Civil Engineers, 1(2), 325-378.

WARDROP, J.G. (1968) Journey speed and flow in central London. Traffic Eng. Control 9, 528-32.

WEBSTER, F.V.(1958) Traffic signal settings. Road Research Technical Paper No.39. H.M.S.O.

WEINTRAUB, A.C. ORTIZ AND J. GONZALES (1985) Accelerating convergence of the Frank-Wolfe algorithm. Transportation Research 19B,113-122.

WOLFE, P. (1974) Algorithm for a least-distance programming problem. Mathematical Programming Study 1,190-205.

ZANGWILL, W.I. (1969) Nonlinear programming: A unified approach. Prentice Hall.

ZUZARTE TULLY, I.M. (1977) Synthesis of sequences for traffic signal controllers using techniques of the theory of graphs. PhD thesis, Univ. of Oxford.

APPENDIX 1. LIST OF MAJOR NOTATION

A1.1 Roman letters

A_{ia} Kronecker value for green time on link a and stage i
$\mathbf{B}$ Set of all O-D pair
C Cycle time for signal controlled junction
c_a Cost on link a
c_a^k Total link cost on link a for class k
C_p Cost on using path p
d Destination node
d_a Mean junction delay on link a
d_a^k Perceived distance on link a by class k
$\mathcal{D}$ Demand feasible set
e_{ab} Parameter to represent the geometric characteristics on link a
$f(v)$ Pseudoconvex function
F^k Fixed cost for the class k
g_a Effective green time on link a
$\mathbf{G}$ A directed graph
$G(v)$ Gap function
G_a Parameter to represent the geometric characteristics on link a
$h(t)$ Headway distribution
$H(W)$ Convex hull of W
Ha Parameter to represent the geometric characteristics on link a
J_a Junction a
J_{ab} The Jacobian matrix
Ka Parameter to represent the geometric characteristics on link a
$\mathbf{L}$ A set of links
M_{od} The minimum travel cost from origin 0 to destination d
$\mathbf{N}$ A set of nodes
o Origin node
P_{od} Set of all paths for O-D pair
Q_a Capacity on link a
Q^p Practical capacity on link
Q'^p Practical capacity per lane on link
Q^s Steady state capacity on link
Q'^s Steady state capacity per lane on link
r Effective red time
$\mathcal{S}$ Supply feasible set
S Feasible convex set
s_a Saturation flow on link a
t_0 Free flow travel time
t_p Path flow on path p

$T(v)$ Total link cost
T_{od} Demand for O-D pair
$\mathbf{v}$ Vector of traffic flow on link
v' Flow per lane on link a
v_a Traffic flow on link a
X_a Degree of saturation rate on signal controlled junction on link a
Y_a Parameter to represent the geometric characteristics on link a
y^*_i Representative approach for stage i
$z(t)$ The Beckmann's objective function

A1.2 Greek letters

$\underline{\alpha}$ Parameter for characteristics for travel cost function
β Parameter for combining extreme points
γ Parameter for combining link and junction delay
δ Kronecker delta for repesenting link and path relation
θ^k Parameter of value of time for class k
λ Ratio of effective green time
μ Capacity on minor link
ψ_{od} Langrange multiplier for each origin and destination pair
ρ_a Degee of saturation rate on link a in priority controlled junction
υ^k PCU value for class k

APPENDIX 2:CHARLESWORTH'S NETWORK AND DATA (Source: Charesworth, 1977)

A2.1 Origin and destination matrix of the network

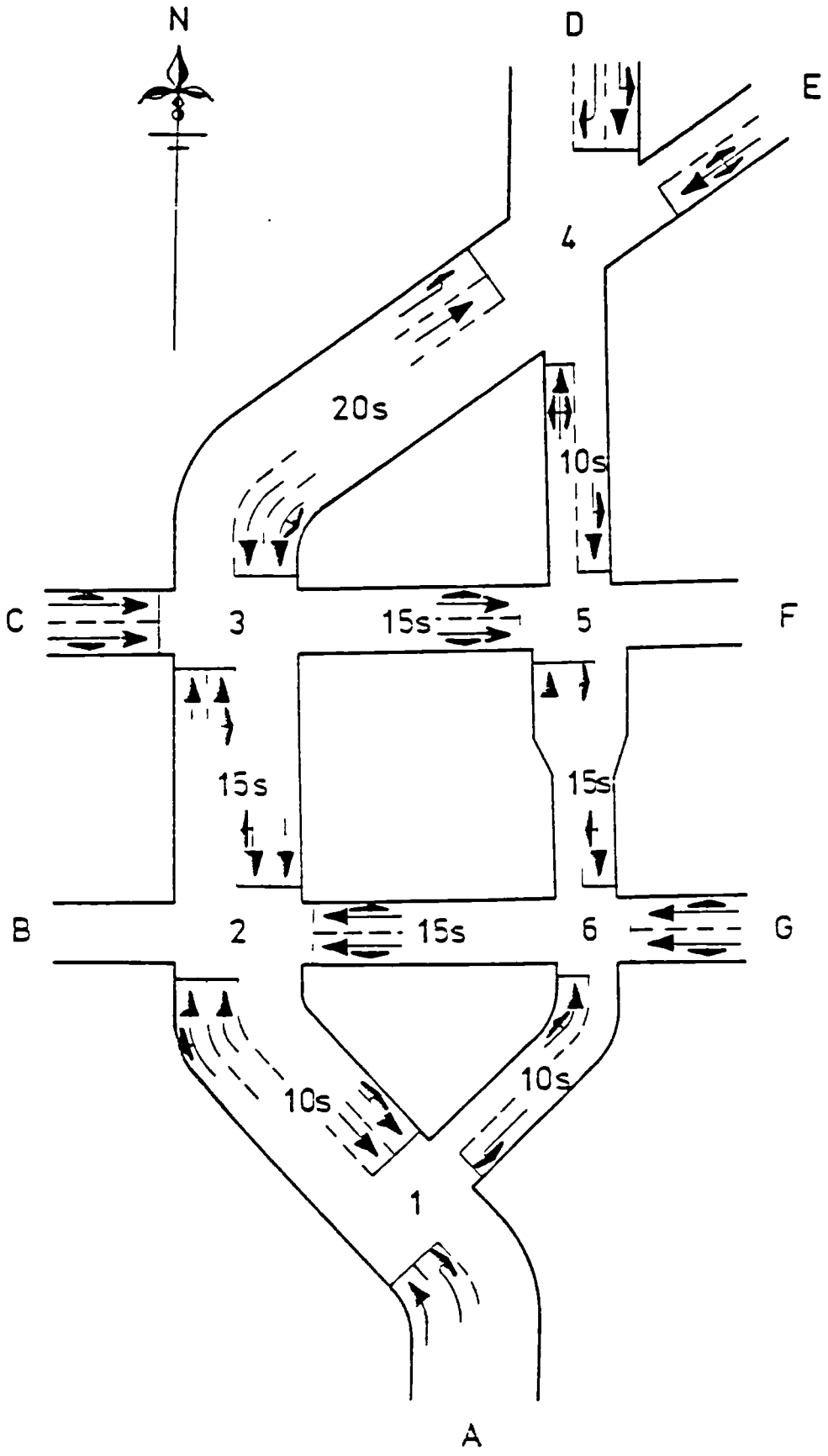
O \ D	A	B	D	E	F	Total
A	-	250	700	30	200	1180
C	40	20	200	130	900	1290
D	400	250	-	50	100	800
E	300	130	30	-	20	480
G	550	450	170	60	20	1250
Total	1290	1100	1100	270	1240	5000

A2.2 Capacity of the network

Link	Capacity	Link	Capacity	Link	Capacity
1012*	1800	1014	1600	1222	1600
1227	1600	1314	2000	1315	1500
1461	1800	2231	1600	2234	1600
2321	2000	2326	2000	2421	2000
2422	2000	2521	2000	2526	2000
2613	1400	2615	1500	2721	2000
2722	2000	3032	1400	3033	1400
3134	2000	3136	2000	3236	2000
3237	2000	3334	2000	3336	2000
3443	1800	3445	1850	3536	2000
3537	2000	3651	1600	3652	1600
3723	1300	3726	1300	4042	2000
4044	1800	4142	2000	4143	2000
4143	2000	4145	2000	4235	1600
4237	1600	4445	2000	4447	2000
4642	2000	4647	2000	4754	1700
5046	2200	5155	2000	5156	2000
5253	2000	5255	2000	5341	2200
5455	2000	5456	2000	5665	1700
6162	2000	6163	2000	6224	1600
6225	1600	6364	1850	6453	1850
6455	1700	6562	2000	6568	2000
6662	2000	6663	2000	6762	2000
6768	2000	6815	1500	7066	1800
7067	1800	1510	20000	2120	20000
4340	20000	4550	20000	5560	20000

(Key:* Link number is represented by joining nodes xy where x is the label of the upstream node and y the label of the downstream node)

A2.3 Diagram of Charlesworth's network



A2.4 Link and node representation of the network for assignment purpose