

PUBLIC TRANSIT SYSTEM NETWORK MODELS :  
CONSIDERATION OF GUIDEWAY CONSTRUCTION ,  
PASSENGER TRAVEL AND DELAY TIME ,  
AND VEHICLE SCHEDULING COSTS

A THESIS

Presented to

The Faculty of the

Division of Graduate Studies and Research

by

Gunter Pielbusch Sharp

In Partial Fulfillment

of the Requirements for the Degree

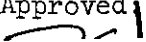
Doctor of Philosophy


in the School of Industrial and Systems Engineering

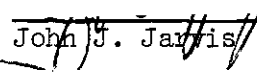
Georgia Institute of Technology

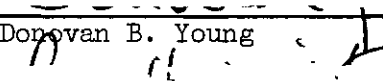
September, 1973

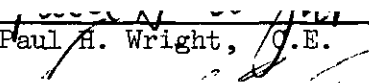
PUBLIC TRANSIT SYSTEM NETWORK MODELS:  
CONSIDERATION OF GUIDEWAY CONSTRUCTION,  
PASSENGER TRAVEL AND DELAY TIME,  
AND VEHICLE SCHEDULING COSTS

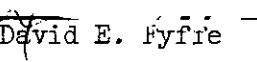
Approved, 

  
Paul S. Jones, Chairman

  
John J. Jarvis

  
Donovan B. Young

  
Paul H. Wright, D.E.

  
David E. Fyfe

Date approved by Chairman: Sept. 14, 1973

## ACKNOWLEDGMENTS

I wish to express my sincere appreciation to Dr. Paul S. Jones, my dissertation advisor, whose insight and labor contributed not only to the direction and completion of this study, but also to my appreciation of the research effort.

Dr. Paul Gray deserves credit for introducing me to the area of public transit network models.

Special thanks are also due to Dr. John J. Jarvis, Dr. Paul H. Wright, Dr. Donovan B. Young, and Dr. David E. Fyffe, who served on the reading committee and provided many helpful recommendations.

Dr. Robert N. Lehrer and Dr. William W. Hines deserve my appreciation for encouraging me to return to graduate school and for providing the financial support which allowed me to pursue full-time studies.

Finally, and most importantly, I especially acknowledge the contribution of my wife, Maria. Without her patient understanding and sacrifice the successful completion of this research would not have been possible.

## TABLE OF CONTENTS

|  | Page |
|--|------|
| ACKNOWLEDGMENTS . . . . .                                | ii   |
| LIST OF TABLES . . . . .                                 | vi   |
| LIST OF ILLUSTRATIONS . . . . .                          | vii  |
| NOMENCLATURE . . . . .                                   | ix   |
| SUMMARY . . . . .  | xii  |
| Chapter  |      |
| I. INTRODUCTION . . . . .                                | 1    |
| Background   |      |
| General Problem Statement                                |      |
| Problem Importance                                       |      |
| II. PREVIOUS WORK . . . . .                              | 8    |
| Urban Transportation Planning                            |      |
| Fixed-Charge Single-Commodity Network Flows              |      |
| Multicommodity Network Flows                             |      |
| Scheduling and Routing                                   |      |
| Synthesis of Fixed-Cost Route Configurations             |      |
| III. METHOD OF APPROACH . . . . .                        | 23   |
| Objectives of Research                                   |      |
| Specific Problem Statement                               |      |
| Some Network Terminology                                 |      |
| Formulation of Problem                                   |      |
| Overview of Solution Procedure                           |      |
| Guideway Insertion-Deletion                              |      |
| Revised Network Representation of Routes                 |      |
| Route Insertion-Deletion                                 |      |
| IV. DETAILED DESCRIPTION OF SOLUTION PROCEDURE . . . . . | 46   |
| Penalty-Cost Multicommodity Flow Assignment Routine      |      |
| Conceptual Version                                       |      |
| Actual Version   |      |

## TABLE OF CONTENTS (Continued)

|  | Page |
|--|------|
| IV. (Chapter continued)                                  |      |
| Route Insertion and Route Insertion Parameters           |      |
| Route Configurations Allowed                             |      |
| Initial Adjustments to the Set of Routes                 |      |
| Constructing the Revised Network                         |      |
| Arc Insertion Improvement Parameters                     |      |
| Increasing Service on Existing Routes                    |      |
| Appending Guideway Arcs to Existing Routes               |      |
| Route Construction                                       |      |
| Route Deletion and Route Deletion Parameters             |      |
| Eliminating Excess Capacity                              |      |
| Arc Deletion Improvement Parameters                      |      |
| Overview of Route Deletion                               |      |
| Guideway Insertion                                       |      |
| Guideway Deletion  |      |
| Summary of Solution Procedure                            |      |
| V. COMPUTATIONAL RESULTS . . . . .                       | 83   |
| Construction of the Revised Network                      |      |
| Penalty-Cost Multicommodity Flow Assignment Routine      |      |
| Route Insertion-Deletion Algorithm                       |      |
| Guideway Insertion-Deletion Algorithm                    |      |
| Test Problem E   |      |
| Test Problem C   |      |
| Test Problem D   |      |
| Summary of Results                                       |      |
| Convergence  |      |
| Execution Times  |      |
| VI. FUTURE EXTENSIONS . . . . .                          | 136  |
| Obtaining an Initial Set of Open Guideways               |      |
| Alternative Method of Generating Guideway                |      |
| Improvement Parameters                                   |      |
| Alternative Method of Generating Arc Deletion Parameters |      |
| Consistent Designation of Revised Network Nodes and Arcs |      |
| Multiple Guideway Changes                                |      |
| Computer Program Improvements                            |      |
| Packing the Data   |      |
| Adaptive Switching Rules                                 |      |
| Bounds on Optimal Solution                               |      |

## TABLE OF CONTENTS (Concluded)

|   | Page |
|---|------|
| VIII. CONCLUSIONS AND RECOMMENDATIONS . . . . . | 151  |
| BIBLIOGRAPHY . . . . .                          | 154  |
| VITA . . . . .                                  | 159  |

## LIST OF TABLES

| Table  | Page |
|--|------|
| 1. Travel Demands in Problem C . . . . .   | 86   |
| 2. Problem Cla, Iteration 0<br>Infeasible Arcs, Commodity Set K1 . . . . .                 | 90   |
| 3. Problem Cla, Iteration 1<br>Infeasible Arcs, Commodity Set K1 . . . . .                 | 91   |
| 4. Problem Cla, Iteration 2<br>Infeasible Arcs, Commodity Set K1 . . . . .                 | 93   |
| 5. Problem Cla; Improvement Parameters at<br>Final Iteration . . . . .                     | 94   |
| 6. Problem Cl, Results of Route Algorithm . . . . .  | 101  |
| 7. Problem E, Travel Demands . . . . .   | 105  |
| 8. Guideway Algorithm, Results of Problem E,<br>Starting with E1. . . . .                  | 106  |
| 9. Guideway Algorithm, Results of Problem C,<br>Starting with C1 . . . . .                 | 111  |
| 10. Guideway Algorithm, Results of Problem C,<br>Starting with C2 . . . . .                | 114  |
| 11. Problem D, Travel Demands . . . . .  | 120  |
| 12. Guideway Algorithm, Results of Problem D,<br>Starting with D1 . . . . .                | 123  |
| 13. Summary of Computational Experience for<br>Multicommodity Assignment Routine . . . . . | 134  |
| 14. Summary of Computational Experience for<br>Guideway Algorithm . . . . .                | 135  |
| 15. Example for Computing Guideway Weighting Factors . . . . .                             | 140  |

## LIST OF ILLUSTRATIONS

| Figure   | Page |
|--|------|
| 1. Comparison of Traveler Oriented Design with Construction Oriented Design . . . . .            | 5    |
| 2. Macro-Relationship Between Algorithms . . . . .   | 33   |
| 3. Guideway Insertion-Deletion Algorithm with Imbedded Second Algorithm . . . . .                | 36   |
| 4. Original Network . . . . .  | 40   |
| 5. Revised Network . . . . .   | 42   |
| 6. Conceptual Route Insertion-Deletion Algorithm . . . . .                                       | 45   |
| 7. Penalty-Cost Multicommodity Flow Assignment Routine, Conceptual Version . . . . .             | 48   |
| 8. Penalty-Cost Multicommodity Flow Assignment Routine, Actual Version . . . . .                 | 52   |
| 9. Reversal Run Route . . . . .  | 57   |
| 10. Increasing Service on Existing Routes . . . . .  | 64   |
| 11. Appending Guideway Arcs to Existing Routes . . . . .   | 66   |
| 12. Route Construction Routine. . . . .  | 71   |
| 13. Deletion Part of Route Algorithm . . . . .   | 72   |
| 14. Relationships Among Routines . . . . .   | 80   |
| 15. Flowchart of Solution Procedure . . . . .  | 82   |
| 16. Station Locations and Possible Guideways in Problem C. Open Guideways in Problem C1. . . . . | 85   |
| 17. Vehicle Routes and Frequencies in Problem C1a . . . . .                                      | 88   |
| 18. Revised Network in Problem C1a . . . . .   | 89   |



## LIST OF ILLUSTRATIONS (Concluded)

| Figure  | Page |
|---|------|
| 19. Problem C1, Changes During Route-Insertion . . . . .  | 96   |
| 20. Problem C1, Results of Route Construction Phase,<br>at Iteration 4 . . . . .                                  | 98   |
| 21. Problem E1, Open Guideways and Starting<br>Vehicle Routes . . . . .   | 104  |
| 22. Problem E, Starting with E1, Guideway Changes . . . . .   | 107  |
| 23. Open Guideways in Problem C2 . . . . .  | 113  |
| 24. Open Guideways in Problem D1 . . . . .  | 119  |
| 25. Problem D, Starting With D1, Vehicle Routes at<br>Beginning and End of Guideway Algorithm Iteration 0 . . . . | 122  |
| 26. Problem D, Starting With D1, Final Sets of Guideways<br>and Routes . . . . .                                  | 125  |
| 27. Problem C, Final Sets of Guideways and Total Costs,<br>Various Runs . . . . .                                 | 126  |
| 28. Problem C1, Starting With C1a, Initial and Final<br>Routes for Guideway Set C1 . . . . .                      | 128  |
| 29. Problem C1, Starting With C1b, Initial and Final<br>Routes for Guideway Set C1 . . . . .                      | 129  |
| 30. Problem C1, Starting With C1c, Initial and Final Routes<br>for Guideway Set C1 . . . . .                      | 130  |
| 31. Problem D, Final Sets of Guideways and Total Costs,<br>Various Runs . . . . .                                 | 132  |
| 32. Example for Computing Guideway Weighting Factors . . . . .  | 138  |
| 33. Example Relating to Multiple Guideway Changes . . . . .   | 147  |

## NOMENCLATURE

|            |  |
|------------|--|
| $a_{mij}$  | incidence designator for arc $(i,j) \in$ route $m$ , $a_{mij} = 1$ if $(i,j)$ is included in route $m$ , $a_{mij} = 0$ otherwise |
| $A$        | the set of all possible guideway arcs in a transit network   |
| $A_1$      | the subset of guideway arcs that are open, or constructed  |
| $A(i)$     | $\{j   (i,j) \in A\}$ , or "after $i$ "  |
| $B$        | the set of arcs in a network   |
| $B_1$      | the set of infeasible arcs in a network  |
| $B(i)$     | $\{j   (j,i) \in A\}$ , or "before $i$ "   |
| $b_{ij}$   | the capacity of an arc   |
| $c_{ij}$   | the operating cost per vehicle on arc $(i,j)$  |
| $d_{ij}$   | the passenger travel time cost on arc $(i,j)$  |
| $d_w$      | the passenger waiting time cost, per time unit   |
| $d_t$      | the passenger transfer time and nuisance cost, per transfer  |
| $e$        | a very small number  |
| $e_1$      | a small number   |
| $\epsilon$ | the amount by which the length of an infeasible arc is increased   |
| $E_1$      | a large number   |
| $E_2$      | a large number, $E_2 > E_1$  |
| $f_{ij}$   | the passenger flow on arc $(i,j)$  |
| $f_{ij}^k$ | the flow of passenger commodity $k$ on arc $(i,j)$   |

## NOMENCLATURE (Continued)

|             |   |
|-------------|---|
| $f_{mij}^k$ | the flow of passenger commodity $k$ on arc $(i,j)$ in route $m$   |
| $g$         | the capacity of a vehicle   |
| $h_{ij}$    | guideway priority rating  |
| $k^*$       | the commodity to be reassigned next from its shortest to its second shortest path   |
| $K$         | the set of passenger commodities, or travel demands between node pairs $ij$   |
| $K_I$       | the set of commodities flowing over infeasible arcs   |
| $l_{ij}$    | the length of a guideway arc in the route construction routine  |
| $l_{kn}$    | the length of the $n^{\text{th}}$ -shortest path for commodity $k$  |
| $l_{kq}$    | the length of the "longest" path for commodity $k$  |
| $m_{knij}$  | a logic operator for inclusion of arcs in paths, $m_{knij} = 1$ if arc $(i,j)$ is in the $n^{\text{th}}$ -shortest path for commodity $k$ and no shorter path, $m_{knij} = 0$ otherwise |
| $M$         | the set of vehicle cycles, or vehicle routes  |
| $n_1$       | the number of infeasible arcs in the shortest path  |
| $n_2$       | the number of arcs in the second shortest path that would be infeasible if the commodity were assigned to it  |
| $N$         | the set of nodes in a network, or the set of stations   |
| $p_1$       | the length of the shortest path   |
| $p_2$       | the length of the second shortest path  |
| $p_{ij}$    | the fixed cost for arc $(i,j)$ , or guideway $(i,j)$  |
| $q_1$       | the average percent vehicle occupancy, throughout the network   |

## NOMENCLATURE (Concluded)

|          |  |
|----------|--|
| $q_2$    | the average percentage of passengers on a vehicle who experienced a waiting time at the origin of the arc being traversed                    |
| $q_3$    | the average percentage of passengers on a vehicle who experienced a transfer at the origin of the arc being traversed                        |
| $r^k$    | the travel demand for commodity k, per operating time period   |
| $r_i^k$  | the net travel demand for commodity k at station i, per operating time period  |
| $s^k$    | the number of route transfer occurrences for commodity k   |
| $t_{kn}$ | the number of guideway arcs in the $n^{\text{th}}$ -shortest path for commodity k  |
| $t_w$    | the average waiting time experienced by passengers when boarding a vehicle   |
| $u_k$    | the amount by which each infeasible arc needs to be lengthened to cause the second shortest path to become the shortest path for commodity k |
| $u_{ij}$ | incidence variable for arc (i,j), $u_{ij} = 1$ if arc (i,j) is open, $u_{ij} = 0$ otherwise  |
| $U$      | a very large number  |
| $v_m^k$  | the number of waiting time occurrences for commodity k on route m  |
| $v_{ij}$ | the variable flow cost on arc (i,j)  |
| $w_{ij}$ | guideway weighting factor  |
| $y_m$    | the number of vehicles traversing route m per operating time period  |

## SUMMARY

The planning process for transportation network systems usually follows a sequence of steps: trip generation, trip distribution, modal split, station location, guideway location, trip assignment, vehicle routing, and vehicle scheduling. This sequential procedure reduces the quality of the design solutions so obtained and diminishes the chances of obtaining optimal or near-optimal solutions.

In this dissertation there is developed a procedure for solving in a unified way the four-component problem that follows after estimating the travel demand and determining the station locations: How should one simultaneously select fixed-cost guideways for inclusion in the network, determine vehicle routes and route service frequencies, and assign passengers to origin-destination paths in order to minimize the total of construction costs, passenger travel and delay time costs, and vehicle operating costs, while satisfying the total transportation demand?

The problem is formulated as a multicommodity transshipment problem where each commodity is restricted to unique origin and destination nodes, with fixed costs on arcs and the inclusion of routing, scheduling, and other constraints. In view of discouraging results by previous researchers using exact methods on subproblems of this problem, the decision was made to develop a heuristic procedure.

The master program contains two separate arc insertion-deletion type algorithms, one imbedded within the other. The first algorithm selects from the set of all possible guideways that subset which is to comprise the final network. The second, imbedded algorithm determines the best flow assignment, vehicle routes, and route service frequencies for a given subset of open guideways. The second algorithm operates on a revised network wherein each vehicle route is represented by a set of nodes and arcs, and boarding and route transfer occurrences are represented by dummy nodes and arcs. Both algorithms make repeated use of a penalty-cost multicommodity flow assignment routine. The master program alternates between the two algorithms until no more cost improvement can be made.

A FORTRAN V computer program has been written and tested on a number of small problems. The typical example contains 12 stations, up to 25 possible two-way guideways, up to 50 travel demands, and up to 15 vehicle routes each containing a maximum of 12 arcs. Execution times range from 1 to 18 minutes on the Univac 1108.

## CHAPTER I

### INTRODUCTION

#### Background

Network models have been used to estimate transportation requirements for almost two decades. Over this period highly developed procedures have been established to estimate highway planning needs, as the Highway Planning Package (47), and to estimate public transit needs, notably bus and rail rapid transit, as the Urban Mass Transit Administration (UMTA) System (49). These procedures have been widely used as keystones of the comprehensive transportation planning studies required of all urban areas within the United States with populations exceeding 50,000.

To use either the Highway or the UMTA package the service area is divided into a large number of zones in which trips can originate or terminate - a large metropolitan area can have 500 or more zones. Travel is thereafter treated in terms of zones, and trips are distributed as zone-to-zone movements. Because of the large number of zones treated the analysis is cumbersome, requiring voluminous input data and considerable computer time. These program batteries typically divide the planning process into four sequential steps:

- . Trip generation - trips are generated for each zone in terms of population and other demographic characteristics
- . Trip distribution - trips are distributed to zone-to-zone pairs
- . Modal split - trips are assigned to transportation modes
- . Trip assignment - trips are assigned paths from origin to destination based on a shortest time or distance criterion.

These models are heavily based on existing urban development and transportation networks that are completely defined before the analysis has begun. Furthermore, the models lack suitable bases for evaluating transportation alternatives.

The impending introduction of advanced public transit systems with characteristics that are quite different from conventional automobile, bus, and rail rapid transit systems suggests a need for new tools and techniques that are capable of identifying optimal or near-optimal services. This interest has given birth to some new procedures for solving transportation network problems in which the network is not given.

Such network synthesis procedures generally follow a sequence of steps similar to that for analyzing existing networks:

- . Demand estimation - comprising trip generation, distribution, and modal split
- . Station location - stations are located to be accessible to a certain percentage of potential transit riders



- . Guideway location - guideways are located to minimize travel time, or meet some other criterion
- . Trip assignment - trips are assigned paths from origin to destination based on a shortest time or distance criterion
- . Vehicle routing - vehicles are assigned to routes to minimize fleet size, minimize carriage miles, minimize passenger travel time, or some other criterion
- . Vehicle scheduling - vehicles are scheduled to minimize carriage miles, minimize intermediate stops, provide at least minimum service, etc.

The above series of steps is usually repeated a number of times before the solution is accepted. The chief drawback to such a sequential procedure is that it cannot achieve an optimal transportation system except by remote chance. Furthermore, the probability of obtaining good, near-optimal solutions is diminished by the sequential process. In some applications the steps of the above procedure are considered two at a time and, more rarely, three at a time.

#### General Problem Statement

The research problem examined in this dissertation is to solve in a unified way the four-component problem that follows after estimating the travel demand and determining the station locations: How should one determine simultaneously the guideway location, the trip assignment, and the vehicle routing and scheduling in order to minimize the total of construction costs, passenger travel and delay time costs, and vehicle operating costs, while satisfying the total

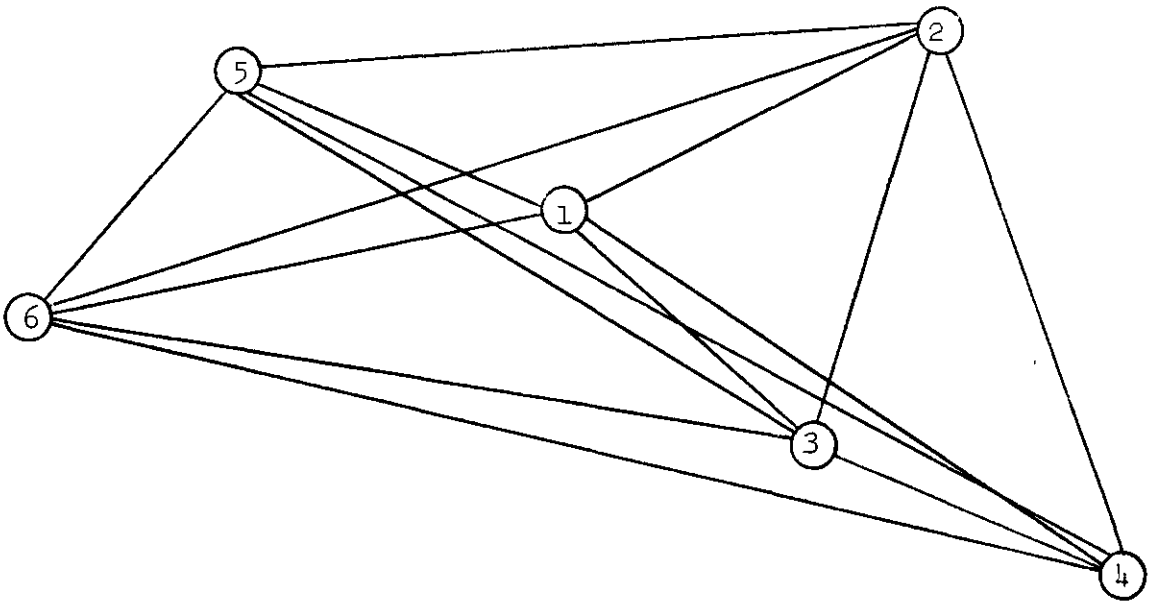
transportation demand?

So stated, the problem is formulated as a multicommodity transshipment problem where each commodity is restricted to unique origin and destination nodes, with fixed costs on arcs and the inclusion of routing, scheduling, and other constraints. To date, little work of a practical nature has been done to solve this problem.

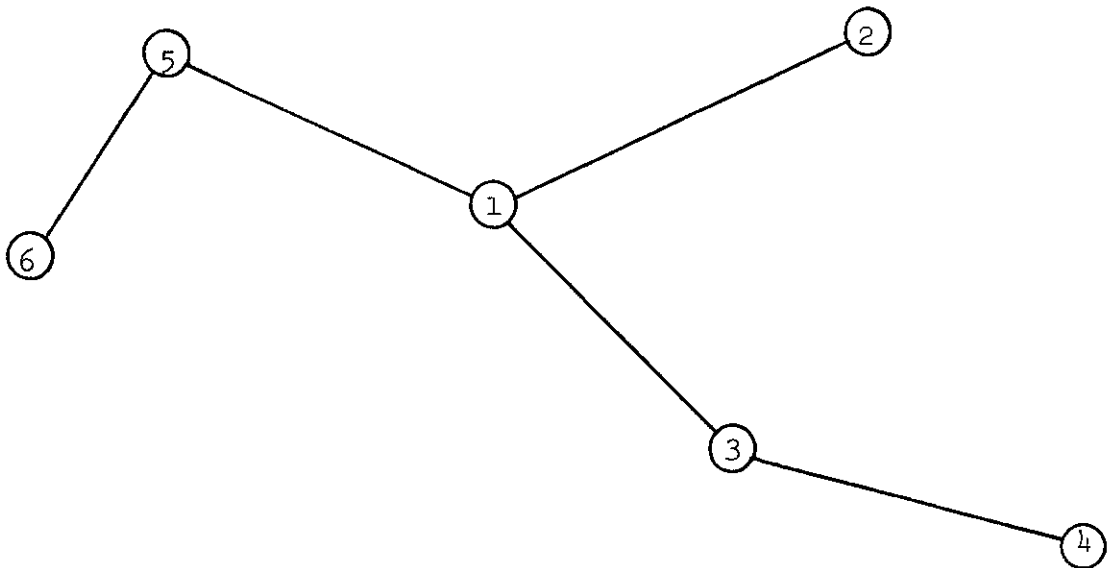
The conflict between construction costs and passenger travel time costs is depicted in Figure 1 (22, p. 140), showing two ways that six stations can be connected. A traveler oriented design results in a maximally connected network, where there is a direct two-way link between each pair of stations. A construction oriented design results in a minimal spanning tree, where the arc weights correspond to the guideway construction costs between stations. The traveler oriented design results in excessive construction costs while the construction oriented design results in poor service to some travelers.

Similarly, there are conflicts between passenger travel time and vehicle routing costs, and between passenger delay time and vehicle scheduling costs. Travelers would benefit most from frequent, direct, shortest-distance routes involving little or no waiting time, no transfer time, no intermediate stops, and a minimum of travel time. This goal is clearly in conflict with economical load factors on vehicles traversing efficient routes.

The problem addressed in this dissertation is that of obtaining



Traveler Oriented Design - Maximally Connected Network



Construction Oriented Design - Minimal Spanning Tree

Figure 1. Comparison of Traveler Oriented Design With Construction Oriented Design.

an acceptable balance among construction costs, passenger convenience, and routing and scheduling efficiency.

#### Problem Importance

Public transit has suffered much from the development of the automobile transportation system. Privately owned transit companies are almost nonexistent. Municipally owned systems are fighting - and often losing - the cost-patronage-frequency of service battle. Clearly, if public transit is to survive, new services are needed, and means must be found to define and apply these services in a near-optimal manner, both from the viewpoint of the operator and of the traveler.

Considerable emphasis is being placed today on the speed and convenience of transit systems, inasmuch as any system must compete with the private automobile. Speed is related to door-to-door time while convenience may be expressed by frequency of service and the number of transfers involved in a journey.

Thus, it is significant to have a solution procedure that can balance construction and operating costs with passenger travel time and convenience.

Other potential applications of a solution procedure for the fixed-cost, multicommodity transshipment problem are given by Billheimer (8): commodity distribution, mail routing, production and inventory control, solid waste disposal networks, and computer systems design. All of these seem amenable to a more realistic

formulation by the inclusion of routing, scheduling, and other constraints.

## CHAPTER II

### PREVIOUS WORK

When the research problem presented in Chapter I is stated in mathematical terms, it becomes an exceedingly large and complex mathematical programming problem. To date, no successful work on this problem has appeared in the literature. It is appropriate, therefore, to examine previous work on subproblems of the research problem.

Thus, the literature review will include, besides the general field of urban transportation planning, previous work on fixed-charge single-commodity network flows, multicommodity network flows, scheduling and routing, and synthesis of fixed-cost route configurations. The difficulties and successes encountered by other researchers on these subproblems will provide meaningful insight for developing a tractable formulation and solution of the research problem.

#### Urban Transportation Planning

A recent study by Meyer, Kain, and Wohl (31) benefits from related studies conducted at the RAND Corporation and provides general background information on the urban transportation problem. The researchers do not attempt to study alternative guideway locations, but rather assumed a typical line-haul configuration with associated residential collection and downtown distribution systems. Station

spacings on the line-haul system are assumed fixed, and certain minimum schedule frequencies are established for line-haul and collection and distribution systems to allow comparison of rail, bus, and automobile modes. Thus, the primary decision variables in the research problem are either fixed or assumed by Meyer, Kain, and Wohl in their work. The number of alternatives examined consist of the various ways that rail, bus, and automobile can be combined for a door-to-door transit journey.

The main results of their study are the statistical data depicting the demand for urban transportation and the comparative costs for the modes of travel considered.

Similarly, other general studies do not address the specific problem under examination here, or they follow a sequential solution procedure as mentioned in Section I. Some of these studies are Creighton (10), Beckman, McGuire, and Winston (5), Kresge and Roberts (25), and Lundberg and Brown (28).

The Highway Planning (47) and UMTA packages (49) mentioned previously utilize shortest-route assignment and deal mainly with existing networks. Thus, they are not well suited for use in designing networks.

#### Fixed-Charge Single-Commodity Network Flows

One of the better known subproblems of the problem under examination is the fixed-charge single-commodity network flow problem.

In one of the earliest papers on this subject Hirsch and Dantzig (20) prove that for a fixed-charge single-commodity trans-

shipment problem the minimum cost solution occurs at an extreme point. This theorem is the basis for a number of exact and heuristic solution techniques developed by others for the fixed-charge transportation problem.

Balinski (2) developed one of the earlier approximate methods. His procedure is to approximate the discontinuous arc-flow cost curves by linear cost curves and then alternate among problems equivalent and similar to the original problem in order to determine upper and lower bounds on the solution to the original problem.

Kuhn and Baumol (26) incorporated several devices into their approximate algorithm: The first of these is a degeneracy forcing routine, relying on the argument that when the fixed costs are approximately equal, an optimal solution will have the fewest arcs open that are necessary for a feasible solution. Second is the routing procedure to ship the most at least cost, or start by assigning the largest shipments to their least-cost routes. Third is Vogel's Approximation Method, whereby one finds the lowest and second lowest cost routes from each supply point to each demand point and then forms error penalties for using the second lowest cost route. The purpose here is to minimize the sum of the shipments multiplied by the respective error penalties.

Dwyer (13) followed Kuhn and Baumol's first device by obtaining the largest order of degeneracy with the aid of completely reduced matrices. This results in a cumbersome but exact technique for equal fixed costs, and in an approximate, combinatorial approach



for the general case of unequal fixed costs. The largest problem illustrated with the latter approach is eight sources by 12 demand points (8 x 12).

Cooper and Drebes (9) derived a modified adjacent extreme points method in which they recalculate certain arc-flow costs at times as

$$\hat{v}_{ij} = v_{ij} + p_{ij}/f_{ij}, \quad f_{ij} > 0 \quad (2-1)$$

where  $v_{ij}$  is the variable flow cost on arc (i,j),  $p_{ij}$  is the fixed cost of opening arc (i,j), and  $f_{ij}$  is the flow on arc (i,j). The entering basis vector is selected from those eligible as the one with the least fixed cost, and the vector leaving the basis as the one having the greatest fixed cost. The algorithm also employs a limited backtracking method to move away from local minima. Computational results are good, with optimal solutions obtained in 90 percent of the randomly generated test problems. Deviation from optimality was as high as 10 percent, but for the largest problems, 15 x 30, deviation from optimality did not exceed 3 percent.

One of the first exact methods of solving the fixed-charge single-commodity transportation problem is Murty's (33) method of ranking the extreme points. The technique consists of obtaining the optimal solution to the problem considering only the variable costs and then progressively examining adjacent vertices. The algorithm terminates with the use of bounds on the solution to the original

problem. Murty's algorithm is limited in that it works well only when the fixed costs are relatively small.

Gray's (17) exact solution procedure is based on decomposing the original mixed integer problem into a master integer problem and a series of transportation subproblems. This method is particularly suitable where the fixed costs are relatively large.

Frank (16) has adapted a network defender-attacker algorithm for determining the minimal cost of opening enough arcs to satisfy total demand. This is then used as a subroutine for obtaining bounds in an exact solution procedure, such as Murty's algorithm, for example.

Steinberg (44) appears to have done the most extensive work in solving large problems with exact techniques. His branch-and-bound algorithm, programmed on an IBM 7072, has solved several 15 x 30 problems in an average time of 21 minutes. Steinberg also has developed several heuristic algorithms, including one using a Monte Carlo method to find a better solution vector several base changes away from a locally optimum point. He reports that the heuristic procedures almost always provide very good, if not optimal, solutions.

Marks, Liebman, and Bellmore (29) have developed an approach using the out-of-kilter algorithm for locating fixed-cost facilities in a transshipment network. The technique appears to be useful for only a small number of source nodes, however.

The group theoretic approach to integer programming has been incorporated into a branch-and-bound algorithm for the fixed-charge transportation problem by Kennington (24). He reports solving problems

as large as  $7 \times 11$  with computation times ranging from 3.5 seconds to 3.7 minutes on a U-1108. Rardin (38) has extended this approach to the general fixed-charge network problem.

In a slightly different vein, Zangwill's (54) dynamic programming algorithms for minimum concave cost flows are restricted to single source-multiple destination, and multiple source-single destination networks.

In summary, the previous published work relating to the fixed-charge single-commodity network flow problem consists of exact techniques limited by their applicability to specialized networks or by their machine time, and heuristic techniques of moderate to very good success. The group theoretic approach, which is still in its infancy, evidently has much to contribute to the exact methods.

#### Multicommodity Network Flows

A brief review of some of the work on multicommodity flows is appropriate since the research problem is formulated in terms of multiple commodities. All of the studies mentioned here deal with problems having linear arc-flow cost functions. The few studies that deal with fixed-charge multicommodity flows are discussed in a later section on network synthesis.

Ford and Fulkerson (14) first suggested a method for obtaining maximum flow in a capacitated multicommodity network by formulating the problem in terms of arc-chain flows and using decomposition. A shortest chain algorithm is used to generate columns entering the basis. Jarvis (23) showed that one obtains an equivalent problem by

using a node-arc formulation.

Tomlin (46) applied the same approaches, the node-arc and the arc-chain formulations, to solving the minimum cost multicommodity flow problem.

Further extensions of this decomposition approach were made by Cremeans, Smith, and Tyndall (11) and by Wollmer (51). These authors handle the case where limited resources are shared by sets of arcs in a capacitated multicommodity network.

The above techniques are all column generation, or price-directive, procedures. Swoveland (45) has developed a resource-directive decomposition algorithm in which rows are generated by solving single-commodity, maximum flow, minimum cost subproblems. His experience indicates that column generation techniques are much faster computationally.

A more powerful approach to multicommodity network flows is found in compact inverse methods. Hartman and Lasdon (19) have developed a generalized upper bounding algorithm for a node-arc formulation. The size of the working basis is the number of currently saturated arcs, and most of the computations are additive and graph theoretic. Grigoriadis and White (18) perform primal partition programming, also using a working basis that corresponds to the currently saturated arcs. They report computational experience on eight test problems, ranging in size from 176 by 315 to 309 rows by 1078 columns in the linear programming version. Their partitioning algorithm ranged from 2.7 to 21.7 times faster than ordinary linear programming.

### Scheduling and Routing

Much of the work in scheduling consists of determining the minimum fleet size for a fixed schedule of trips between stations or nodes. A variety of techniques have been used to solve this problem:

Dantzig and Fulkerson (12) formulated it as a large transportation problem. Levin (27) solves the minimum fleet size problem by finding the minimum number of chains for an acyclic graph. Bartlett (3) takes a different approach in minimizing total idle time of the transport units, using some results of finite ordered sequences of numbers.

A slightly more general formulation of the problem, where trips are constrained within certain time intervals, has been solved by Martin-Löf (30) with a branch-and-bound procedure.

Determining a schedule itself is a much more difficult problem, and results in this area are disappointing. Among the methods presented in the literature are dynamic programming - Nemhauser (34), Salzborn (40), and Newell (35); calculus - Newell (35); integer programming - Levin (27) and Schwartz (41); simulation - Staub (43); and approximate solution to a two-point-boundary-value problem - Bett (7). All but the simulation procedure apply only under restrictive circumstances, such as a single transit line, or limited variability of departure times. The Schwartz study is interesting as an example of a barge scheduling problem, where one must determine commodity assignments on tow units and the departure times of tow and traction units on a river. For an example of 4 ports and T time units in the scheduling

period the number of constraints for the problem are  $(32 + 26T)$ .

The problem of routing commodities and scheduling transport units to serve the demand has been approached by at least two methods. Bartlett and Charnes (4) use a simple linear programming solution to determine the assignment and routing of locomotives to meet transport demand in a railroad system. Their problem is simplified by the restriction of cyclic routes to no more than three arcs, however. For an urban transport system the number of possible cyclic routes can reach astronomical proportions, and linear programming solutions involve too many variables and constraints.

Dynamic programming concepts lead to approximate algorithms by Young (53) and Hyman and Gordon (21). The procedure typically starts by assigning the first vehicle to maximize the objective function; then the second vehicle is assigned, then the third, etc. Young's method allows for reassignment of any vehicle during the process, whereas Hyman and Gordon's method does not. Young's algorithm obtains a first-order optimality condition; in other words, it is impossible to transfer travelers to some vehicle and to discharge others from the same vehicle such that a net gain to the system can result. Hyman and Gordon's algorithm is simply a one-pass operation. Both methods are essentially applicable to single-commodity problems in that no travel demand spans a decision node.

Summarizing the work on routing and scheduling, one must conclude that no exact techniques exist for realistic-size problems in urban transportation, and that the approximate methods that do

exist need to be improved.

### Synthesis of Fixed-Cost Route Configurations

The previous work most pertinent to the research problem relates to synthesizing networks with fixed-cost arcs. The typical problem statement is to solve a fixed-charge multicommodity network flow problem; in other words, select a set of arcs to minimize the total of fixed arc costs and variable arc flow costs subject to multicommodity flow requirements.

Scott (42) examined in detail a special case of this problem, namely an uncapacitated network in which each commodity flow is of the same magnitude. Beginning with an integer programming approach, he concluded that for any realistic problem the model would be so large as to preclude the possibility of numerical computation. Next, he prepared a backtrack programming algorithm to obtain optimal solutions for 26 example networks containing 7 to 10 nodes. Solution times ranged from less than one minute to more than one hour on an IBM 360/65.

These results provided the incentive to develop two heuristic algorithms: A forward algorithm starts with a minimal spanning tree where the arc lengths correspond to the fixed arc costs. Arcs are added or deleted if the net change in the total of fixed and variable costs is improved. A backward algorithm works similarly except that it begins with a maximally connected network. Solution times for each of the 26 previous examples were less than one minute. The forward algorithm found the optimal solution in 21 of the 26 examples,

with deviations as high as 10.4 percent; the backward algorithm did better, finding the optimal solution in 24 examples, and a maximum deviation from the optimal value of 2.3 percent.

Billheimer's (8) arc insertion-deletion algorithm is a successful extension of Scott's backward algorithm, designed to solve the fixed-charge multicommodity network flow problem. The method starts with a maximally connected network, or one with all arcs open, and assigns each travel demand to the shortest route between origin and destination. Next are computed improvement parameters for each arc, based on the fixed cost saving from closing an arc and the increased variable flow costs in the network due to some flow being rerouted. That arc with the greatest improvement parameter is then deleted, and the process is repeated. Similarly, improvement parameters are computed for opening arcs that are closed in the current network, leading to insertion of arcs. The algorithm switches among the arc deletion, the arc insertion, and the shortest path assignment routines until no further improvement can be made.

A different approach to a similar problem is O'Connor and DeWald's (37) sequential deletion algorithm. The problem setting includes given travel demands at fixed nodes, arc capacity functions that increase step-wise with arc investment, and uniform unit variable flow costs and unit construction costs throughout the network. The objective is to determine investment levels for the various arcs in order to minimize the total of investment and flow costs. The algorithm begins with a maximally connected network and then evaluates



all possible networks with one two-way arc deleted. The best of these networks is selected as a successor, and the process is repeated by evaluating all possible networks formed from the successor network by deleting one two-way arc. The algorithm terminates when no successor exists that improves total costs while preserving feasibility.

O'Connor and DeWald have been able to prove optimality of the algorithmic solution for networks of four nodes, but not for larger networks. An interesting feature of the problem is that it assumes captive, single-passenger vehicles, whereas Billheimer's formulation assumes dual-mode vehicles that enter and exit the system according to the passengers' travel demand.

Ochoa-Rosso's (36) work on mixed-integer programming for stage-wise synthesis of multicommodity transportation networks should also be mentioned here. If one reduces the problem to a single stage and assumes no arcs open in the existing network, then the procedure determines a network to minimize total unit costs subject to arc capacities and budgetary constraints on arc construction costs.

Yaged (51) has done extensive work on multicommodity communications networks with continuous, increasing, concave arc-flow cost functions, and symmetric demand. The procedure finds local optima by minimizing a concave function over a convex set. An interesting feature of the solution is that if one takes the derivative of each arc-flow cost function, evaluated at the optimal network flow on the arc, to be the variable arc flow cost; then each commodity follows a shortest path from origin to destination. For fixed-charge arc-flow

cost functions Yaged repeats the procedure a number of times, using several different sets of continuous, concave arc-flow cost functions to approximate the original discontinuous function.

Ridley's (39) formulation of a transportation problem bears some resemblance to Yaged's communications network. Ridley assumes travel time on each arc varies linearly and inversely with arc investment. His objective is to minimize total travel time subject to multicommodity flow requirements and a total budget constraint. A branch-and-bound solution procedure has been developed for 0-1 investment decisions for each arc.

Ishmael (22) considers the special case of selecting potential arcs and nodes to form spanning tree networks. This is applicable where travel demand is to and from a central business district (CBD). Two objective functions are considered: First, minimize the total of construction, vehicle operating, and passenger travel time costs subject to maximum allowable trip times for passengers boarding at each station; second, minimize the total travel time for users, subject to total system costs. A heuristic algorithm is available for the first objective function, and an exact algorithm for both objective functions. A typical running time to obtain a solution with the exact algorithm for a 55-node, 243-arc problem was 25 seconds on a Honeywell 635. The drawback of Ishmael's work is that the algorithms are not suited to multiple origin, multiple destination problems, as the author himself admits (22, p. 334). The assumption of a spanning tree inevitably centered at a CBD effectively results in a single-

commodity problem.

Still another approach is that of Aburto-Avila (1), who maximizes profits in a situation where expected demand for travel between each node pair is inversely related to travel time. The important difference here is the absence of any flow requirements. Otherwise, the problem has the now-familiar fixed arc costs and linear passenger travel time costs. Aburto-Avila solves the problem as a linear program with 0-1 variables, coefficients, and right-hand side. The optimal solution is obtained by implicit enumeration while a faster, heuristic procedure yields approximate solutions.

An example of the naivete behind some of the literature articles on fixed-cost route configurations may be found in Morlok's (32) recommendations for determining transportation systems. He suggests that the optimal operating policy with respect to schedules and fares be obtained by linear programming, for each network alternative. Furthermore, this suggestion was made in the context of stage-wise implementation, where a linear program would have to be solved for each possible network at each stage. Dynamic programming would then be used to select the networks to be implemented at each stage. The review of scheduling and routing work in the previous section should convince the reader that linear programming is not the answer for determining schedules and fares.

An implementation of the above idea is given by Bergendahl (6). The example has five potential two-way arcs to add to an existing network, and involves four time periods. The solution involved 120

linear programming solutions and a total of 98 possible investment programs. Running time was 3.5 minutes for the linear programs and 0.5 minutes for the dynamic programming routine on an IBM 360/67. The solution procedure, incidentally, does not find a true optimum since only arc additions are allowed at each stage.

In summary, the few workable exact techniques for synthesizing fixed-cost route configurations apply only under special restrictions. Of the heuristic techniques developed, the ones most promising of application to the research problem are Billheimer's arc insertion-deletion algorithm and O'Connor and DeWald's sequential deletion algorithm.

## CHAPTER III

### METHOD OF APPROACH

#### Objectives of Research

If any general conclusion can be drawn from the literature survey in Section II, it is that few authors have been able to develop exact techniques applicable to subproblems of the research problem attacked here. The exact techniques that do exist pertain to fixed-charge single-commodity network flows and to multicommodity network flows; no exact techniques were found to solve realistic-size scheduling and routing problems.

Exact techniques for synthesis of fixed-cost route configurations required inordinate amounts of computer time. It is therefore unrealistic to attempt the development of an exact technique for solving the research problem set forth in Section I, given the present state of computer technology.

On the other hand, previous authors have succeeded in constructing good to very good approximate algorithms for some of the subproblems. This success with heuristic procedures suggests that the pursuit of heuristics is the method of approach most likely to produce good, workable solutions to the research problem.

Accordingly, it is the objective of this research to formulate, develop, and test an approximate algorithm for determining simultaneously the guideway location, the trip assignment, the vehicle routes, and

the route service frequencies in order to minimize the total of construction costs, passenger travel and delay time costs, and vehicle operating costs, while satisfying given travel demands between given station locations. The techniques used in the algorithm are based on the arc insertion-deletion routines developed by Scott (42) and Billheimer (8).

### Specific Problem Statement

#### Some Network Terminology

The problem of determining guideways, trip assignments, vehicle routes, and route service frequencies can be conveniently represented in network terms. Consider a given connection network  $(N,A)$ , with the set of nodes  $N$  representing station locations and the set of one-way arcs  $A$  representing possible one-way guideways connecting the stations. These station locations and possible guideways are predetermined by the transportation planner. The one-way guideways are assumed to be paired and uncapacitated.

Travel demand occurs between node pairs  $ij$ , and each passenger traveling on the system can be represented as a unit of flow. Since passengers traveling from  $i$  to  $j$  are not interchangeable with passengers traveling from  $k$  to  $\ell$ , the units of flow must be differentiated according to origin, or destination, or both. Here it is expedient to consider each group of passengers traveling from some node  $i$  to some node  $j$  as a commodity. The travel demands for passengers then becomes multicommodity flow requirements.

A vehicle route consists of a series of connected arcs, oriented so that the head of one is incident at the tail of the next, or, in network terms, a path. If the route ends at the same node as its starting point, the path is a cycle. It is convenient to restrict all vehicle routes to cycles, with vehicles stopping at all stations in a cycle.

Passenger flow units must be assigned to trips, or assigned to paths that are contained in a vehicle route or the union of two or more vehicle routes. Passenger flow assignment is restricted by the capacity and frequency of service of vehicles assigned to the route or route union. Furthermore, vehicles cannot be assigned to routes unless all arcs of the route are open, or all guideways traversed by the route are constructed.

Since the station locations are assumed fixed, the cost of including an arc  $(i,j)$  in the transportation network can be represented by a fixed cost  $p_{ij}$ . Each arc  $(i,j)$  has also associated with it a variable passenger travel time cost  $d_{ij}$ , common to all passenger commodities, and a variable vehicle operating cost  $c_{ij}$ . It is assumed that "vehicles" throughout the system operate singly or in a single capacity combination.

Two types of passenger delay time costs are considered, a waiting time cost and a transfer time cost. The waiting time cost  $d_w$  is incurred at the passenger's origin node. This cost is inversely proportional to the frequency of route service from the passenger's origin node to his destination node. The transfer time and nuisance

cost  $d_t$  is incurred whenever a passenger must transfer from one vehicle route to another.

### Formulation of Problem

Using the above terminology, then, the problem is to select for a network a subset of the set of arcs  $A$ , to assign vehicles to routes subject to feasibility constraints, and to route flow on the network subject to multicommodity flow requirements and vehicle capacity constraints on the arcs. The objective is to minimize the total of construction costs  $p_{ij}$ , passenger travel time costs  $d_{ij}$ , passenger delay time costs  $d_w$  and  $d_t$ , and vehicle operating costs  $c_{ij}$ .

One thus obtains:

$$\begin{aligned}
 \text{Minimize } Z = & \sum_{(i,j) \in A} u_{ij} p_{ij} + \sum_{k \in K} \sum_{m \in M} \sum_{(i,j) \in A} f_{mij}^k d_{ij} & (3-1) \\
 & \text{(guideway fixed costs)} & \text{(travel time costs)} \\
 & + d_w \sum_{k \in K} \sum_{m \in M} v_m^k / y_m + d_t \sum_{k \in K} s^k \\
 & \text{(waiting time costs)} & \text{(transfer time costs)} \\
 & + \sum_{m \in M} \sum_{(i,j) \in A} y_m a_{mij} c_{ij} \\
 & \text{(vehicle operating costs)}
 \end{aligned}$$

Subject to

Passenger flow requirements at each node:



$$\sum_{m \in M} \sum_{j \in A(i)} f_{mij}^k - \sum_{m \in M} \sum_{j \in B(i)} f_{mji}^k = r_i^k, \quad k \in K, i \in N \quad (3-2)$$

Vehicle capacity on each arc:

$$\sum_{k \in K} f_{mij}^k \leq y_m a_{mij} g, \quad m \in M, (i,j) \in A \quad (3-3)$$

Arc feasibility:

$$\sum_{m \in M} y_m a_{mij} \leq u_{ij} U, \quad (i,j) \in A \quad (3-4)$$

Symmetric arc property:

$$u_{ij} - u_{ji} = 0, \quad (i,j) \in A \quad (3-5)$$

Waiting time occurrences:

$$v_m^k = \frac{1}{2} \sum_{(i,j) \in A} \left| f_{mij}^k - \sum_{\ell \in N} f_{mj\ell}^k \right|, \quad k \in K, m \in M \quad (3-6)$$

Transfer time occurrences:

$$s^k = \frac{1}{2} \sum_{m \in M} \sum_{(i,j) \in A} \left| f_{mij}^k - \sum_{\ell \in N} f_{mj\ell}^k \right| - 1, \quad k \in K \quad (3-7)$$

Integrality and nonnegativity:

$$u_{ij} = 0, 1, \quad (i,j) \in A \quad (3-8)$$

$$f_{mij}^k \leq 0, \quad k \in K, m \in M, (i,j) \in A \quad (3-9)$$

$$y_m = \text{a nonnegative integer,} \quad m \in M \quad (3-10)$$

$$v_m^k = \text{a nonnegative integer,} \quad k \in K, m \in M \quad (3-11)$$

$$s^k = \text{a nonnegative integer,} \quad k \in K \quad (3-12)$$

Where

$(N,A)$  is the given network of nodes  $i \in N$  and possible arcs  $(i,j) \in A$

$K$  set of commodities, or travel demands between node pairs  $ij$

$M$  set of cycles, or vehicle routes

$A(i)$   $\{j | (i,j) \in A\}$  , or "after  $i$ "

$B(i)$   $\{j | (j,i) \in A\}$  , or "before  $i$ "

$u_{ij}$  incidence variable for arc  $(i,j)$ ,  $u_{ij} = 1$  if the arc is open, or the guideway connecting stations  $i$  and  $j$  is constructed,  $u_{ij} = 0$  otherwise

$f_{mij}^k$  flow of commodity  $k$  on arc  $(i,j)$  in route  $m$  during the operating time period, or the number of passengers of commodity  $k$  traveling from  $i$  to  $j$  on a vehicle on route  $m$

$y_m$  number of vehicles traversing route  $m$  per operating time period, with vehicles making stops at each station in the route

$v_m^k$  number of waiting time occurrences of commodity  $k$  on route  $m$  during the operating time period, or the number of passengers of commodity  $k$  boarding route  $m$

$s^k$  number of transfer occurrences of commodity  $k$  from one route to another during the operating time period, or the number of

|           |  |
|-----------|--|
|           | passengers of commodity $k$ undergoing a transfer as part of their origin-destination journey  |
| $p_{ij}$  | fixed cost of opening arc $(i,j)$ , or the cost of opening the guideway connecting stations $i$ and $j$  |
| $d_{ij}$  | variable unit flow cost on arc $(i,j)$ , or variable passenger travel time cost on guideway connecting stations $i$ and $j$  |
| $d_w$     | passenger waiting time cost per time unit  |
| $d_t$     | passenger transfer time and nuisance cost  |
| $c_{ij}$  | variable cost of operating and amortizing a vehicle on arc $(i,j)$   |
| $a_{mij}$ | incidence designator for arc $(i,j) \in$ route $m$ , $a_{mij} = 1$ if $(i,j) \in m$ , $a_{mij} = 0$ otherwise  |
| $r_i^k$   | net supply of commodity $k$ at node $i$ per operating time period, distributed uniformly during that period; $r_i^k > 0$ at the origin node for commodity $k$ and $r_i^k < 0$ at the destination node, and $r_i^k = 0$ elsewhere; the number of passengers of commodity $k$ who wish to board at station $i$ |
| $g$       | capacity of a vehicle  |
| $U$       | a large number   |

The first summation in the objective function (3-1) represents the arc fixed costs. Each  $p_{ij}$  would normally be the equivalent cost of owning and maintaining during the operating time period the exclusive one-way guideway connecting stations  $i$  and  $j$ . Because of the arc symmetry equations (3-5) each  $p_{ij}$  represents one-half the cost of owning and maintaining the two-way guideway between  $i$  and  $j$ .

The second summation in the objective function represents the passenger travel time costs. The cost  $d_{ij}$  reflects primarily the arc traversal time but also includes stopping times and passenger processing times. It is assumed that travel is on exclusive guideways or

uncongested streets; hence, travel time is not a function of flow. All flow units of a commodity follow the same path.

The third part, the passenger waiting time costs, reflects the assumption that passengers of commodity  $k$  assigned on route  $m$  are distributed uniformly on evenly-spaced vehicles on route  $m$ . Actually, this simplification results in a lower bound for the true waiting time costs.

The transfer time and nuisance cost  $d_t$  is assumed constant for each such transfer. If the actual timetable schedule to be eventually constructed is designed to provide for coordinated schedules on connecting routes, then  $d_t$  will be relatively small. In such a case the waiting time cost  $d_w$  incurred by passengers transferring onto the route will also be small, and  $d_t$  can be adjusted downward so that  $(d_t + (d_w/y_m))$  is a reasonable representation of the total time and nuisance cost involved.

In the last part of the objective function, the vehicle operating costs, the cost  $c_{ij}$  is based on arc traversal time and power consumption. It is assumed that vehicle capital costs are recovered on a mileage basis and therefore included in this  $c_{ij}$ .

The multicommodity flow requirements equations (3-2) are expressed for each commodity and each node. Equations (3-3) limit the total flow of passengers on each arc of each route to the total vehicle capacity provided during the operating time period. The arc feasibility constraints (3-4) prevent vehicles from being assigned to closed arcs, and the symmetric arc property equations (3-5) simply

reflect the assumption concerning two-way guideways.

In order to determine the waiting time occurrences for a particular passenger commodity  $k$  it suffices to count the number of times that the flow of commodity  $k$  into a node does not equal the flow of the commodity out of the node on the same route. Since all flow units of a commodity follow the same path, any imbalance must represent either vehicle boarding or deboarding by all passengers of commodity  $k$ . The absolute value operator in equations (3-6) counts such imbalance occurrences for each commodity on each route, and since boarding and deboarding are paired, the result is divided by two.

The transfer time occurrences are counted in a similar manner, except that the summation is taken over all routes for each commodity, and the first boarding-deboarding imbalance pair is not considered a transfer.

For a maximally connected network of 10 stations, and with each vehicle route limited to touching five stations, the above formulation results in approximately

|           |                      |
|-----------|----------------------|
| 5,200,000 | continuous variables |
| 1,100,000 | integer variables    |
| 450,000   | constraints.         |

#### Overview of Solution Procedure

The solution procedure for the research problem as formulated in the previous section consists of two algorithms, the second imbedded within the first. The first selects from the set  $A$  of

possible arcs, or guideways, that subset A1 which is to comprise the final network. The second determines the best flow assignment, vehicle routes, and route service frequencies for a given subset A1 of arcs. The macro-relationship between the two algorithms is shown in Figure 2.

Both algorithms are of the arc insertion-deletion type, 'used successfully by Scott (42) and Billheimer (8). They rely heavily upon the finding of shortest and second shortest paths for commodities and the subsequent computation of improvement parameters. Both also require a capacitated, minimum-cost multicommodity flow algorithm as a subroutine.

#### Guideway Insertion-Deletion

The essential aspects of an arc insertion-deletion algorithm may be described in the context of the following simplified problem:

$$\begin{aligned} \text{Minimize} \quad Z1 = & \sum_{(i,j) \in A} u_{ij} p_{ij} + \sum_{k \in K} \sum_{(i,j) \in A} f_{ij}^k d_{ij} & (3-13) \\ & \text{(arc fixed costs)} \quad \text{(variable flow costs)} \end{aligned}$$

Subject to

Flow requirements:

$$\sum_{j \in A(i)} f_{ij}^k - \sum_{j \in B(i)} f_{ji}^k = r_i^k, \quad k \in K, \quad i \in N \quad (3-14)$$

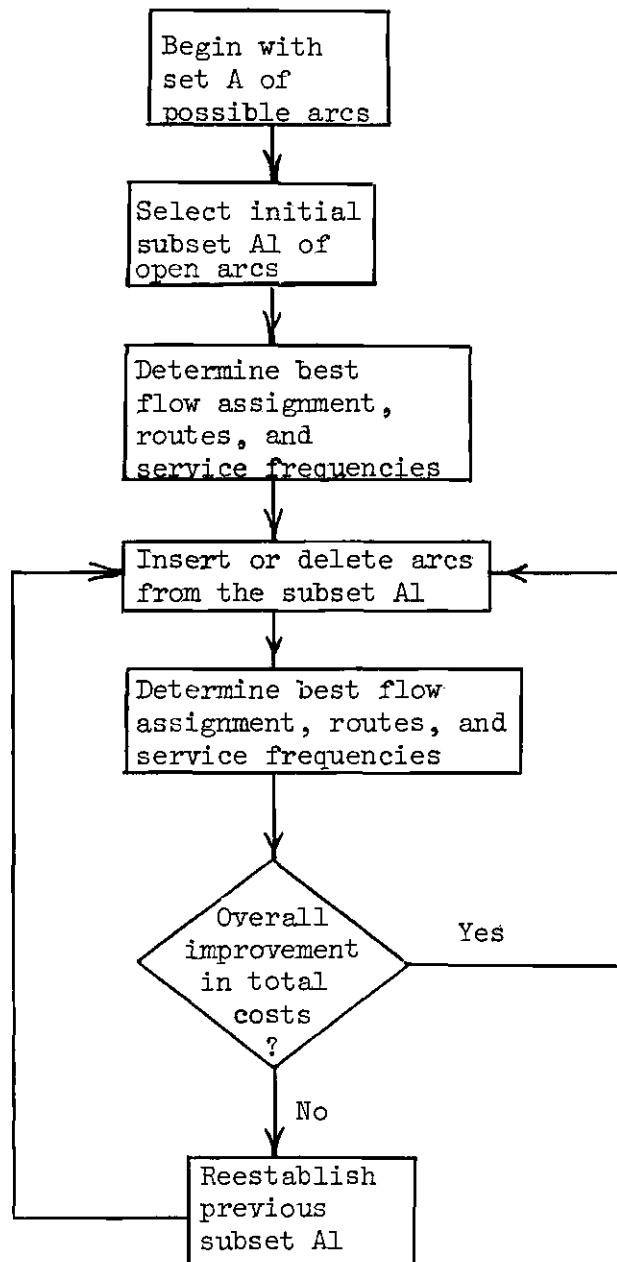


Figure 2. Macro-Relationship Between Algorithms.

Arc feasibility:

$$\sum_{k \in K} f_{ij}^k \leq u_{ij} b_{ij}, \quad (i,j) \in A \quad (3-15)$$

Integrality and nonnegativity:

$$u_{ij} = 0,1, \quad (i,j) \in A \quad (3-16)$$

$$f_{ij}^k \geq 0, \quad k \in K, (i,j) \in A \quad (3-17)$$

where

$f_{ij}^k$  is the flow of commodity  $k$  on arc  $(i,j)$

$b_{ij}$  is the capacity of arc  $(i,j)$  if the arc is open,

and other terms are as previously defined.

Starting with an initial set  $A_1$  of open arcs, the method assigns flow so as to minimize the total variable flow costs in the network  $(N, A_1)$ . Then arc improvement parameters are computed to reflect the trade-offs between fixed and variable costs, insertion parameters for closed arcs and deletion parameters for open arcs:

$$\begin{aligned} \left[ \begin{array}{c} \text{Arc insertion} \\ \text{improvement} \\ \text{parameter} \end{array} \right] &= \left[ \begin{array}{c} \text{Change in fixed costs } p_{ij} \\ \text{from adding arc } (i,j) \end{array} \right] (+) \quad (3-18) \\ &\text{plus} \\ &\left[ \begin{array}{c} \text{Change in variable costs } d_{ij} \\ \text{from rerouting some flow} \end{array} \right] (-) \end{aligned}$$



$$\begin{aligned}
 \left[ \begin{array}{c} \text{Arc deletion} \\ \text{improvement} \\ \text{parameter} \end{array} \right] &= \left[ \begin{array}{c} \text{Change in fixed costs } p_{ij} \\ \text{from removing arc } (i,j) \end{array} \right] \begin{array}{c} (-) \end{array} \quad (3-19) \\
 &\quad \text{plus} \\
 &\quad \left[ \begin{array}{c} \text{Change in variable costs } d_{ij} \\ \text{from rerouting some flow} \end{array} \right] \begin{array}{c} (+) \end{array}
 \end{aligned}$$

A negative improvement parameter indicates that the change in fixed plus variable costs leads to an overall cost reduction. The arc insertion-deletion algorithm first computes improvement parameters for arc insertion. The arc with the most negative improvement parameter, if one exists, is then inserted into the network. The flow is reassigned and total fixed and variable costs are computed to verify the overall cost reduction. New improvement parameters are computed for arc insertion and the process is repeated.

When there exists no arc with a negative improvement parameter, the algorithm switches to arc deletion. The algorithm alternates between insertion and deletion until no more overall cost improvement occurs.

Figure 3 shows the result of applying this type of arc insertion-deletion algorithm to the research problem as formulated in equations (3-1) through (3-12). Each time a guideway is added to or removed from the set A1 the second, imbedded algorithm is executed to obtain the best flow assignment, vehicle routes, and service frequencies.

The computation of exact guideway improvement parameters would involve consideration of the vehicle operating costs  $c_{ij}$ , the

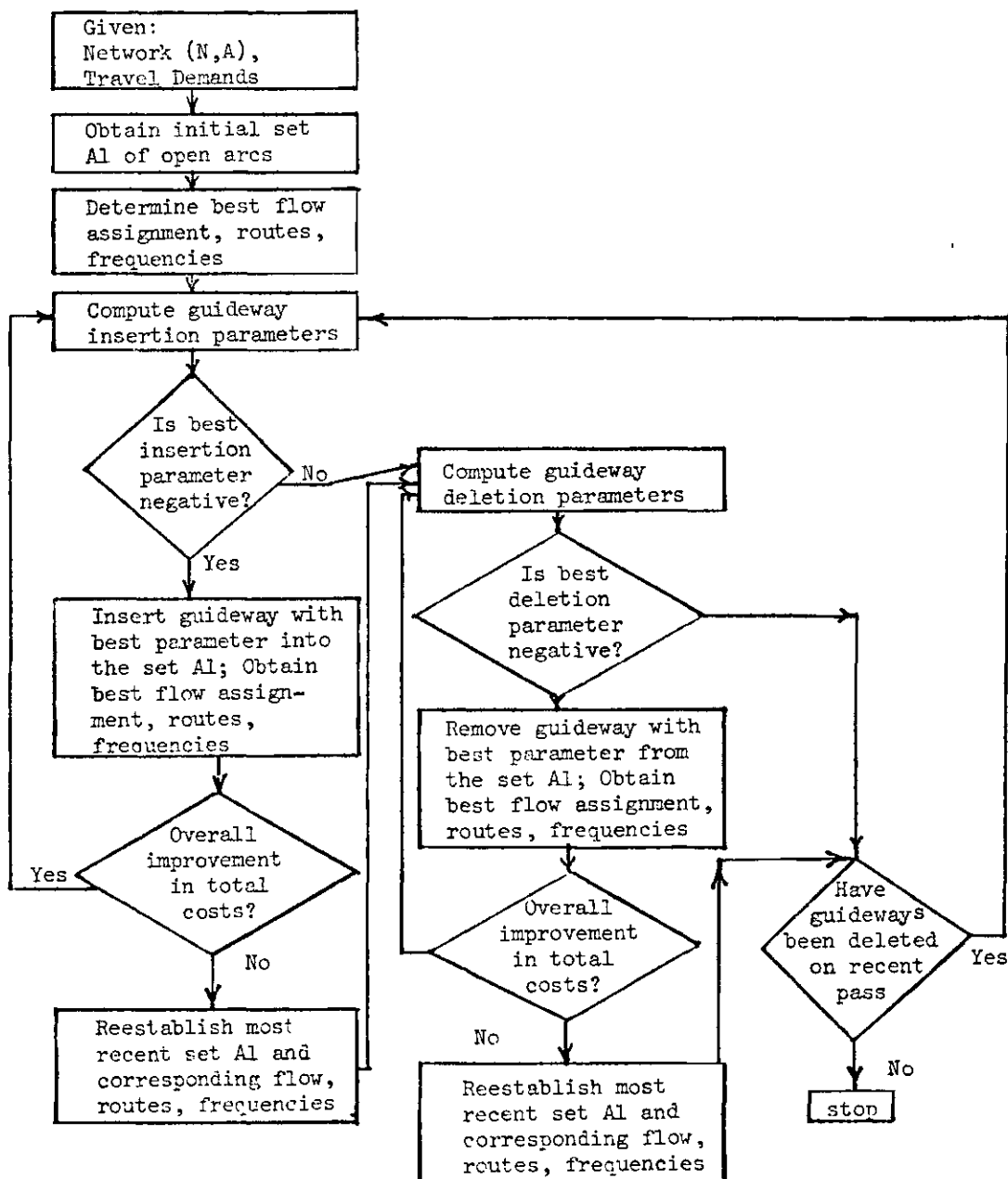


Figure 3. Guideway Insertion-Deletion Algorithm  
With Imbedded Second Algorithm.

passenger delay time costs  $d_w$  and  $d_t$ , and the various constraints missing from the simplified formulation of (3-13). Hence, approximate guideway improvement parameters are used, as described in the next chapter. Accordingly, every change in the set A1 leading to an indicated improvement in total costs must be verified, and provision is made for such a check in the algorithm.

#### Revised Network Representation of Routes

The second algorithm obtains the solution to the following problem:

$$\begin{aligned}
 \text{Minimize} \quad Z2 = & \sum_{k \in K} \sum_{m \in M} \sum_{(i,j) \in A} f_{mij}^k d_{ij} + d_w \sum_{k \in K} \sum_{m \in M} \left[ \frac{v_m^k}{y_m} \right] \quad (3-20) \\
 & \quad \quad \quad (\text{travel time costs}) \quad \quad \quad (\text{waiting time costs}) \\
 & + d_t \sum_{k \in K} s^k + \sum_{m \in M} \sum_{(i,j) \in A} y_m a_{mij} c_{ij} \\
 & \quad \quad \quad (\text{transfer time costs}) \quad \quad \quad (\text{vehicle operating costs})
 \end{aligned}$$

Subject to

Passenger flow requirements:

$$\sum_{m \in M} \sum_{j \in A(i)} f_{mij}^k - \sum_{m \in M} \sum_{j \in B(i)} f_{mji}^k = r_i^k, \quad k \in K, \quad i \in N \quad (3-2)$$

Vehicle capacity:

$$\sum_{k \in K} f_{mij}^k \leq y_m a_{mij} g, \quad m \in M, (i,j) \in A \quad (3-3)$$

Arc feasibility:

$$\sum_{m \in M} y_m a_{mij} = 0, \quad (i,j) \notin A \quad (3-21)$$

Waiting time occurrences:

$$v_m^k = \frac{1}{2} \sum_{(i,j) \in A} \left| f_{mij}^k - \sum_{\ell \in N} f_{mj\ell}^k \right|, \quad k \in K, m \in M \quad (3-6)$$

Transfer time occurrences:

$$s^k = \frac{1}{2} \sum_{m \in M} \sum_{(i,j) \in A} \left| f_{mij}^k - \sum_{\ell \in N} f_{mj\ell}^k \right| - 1, \quad k \in K \quad (3-7)$$

Integrality and nonnegativity:

$$f_{mij}^k \geq 0 \quad (i,j) \in A \quad (3-9)$$

$$y_m, v_m^k, s^k \text{ nonnegative integers} \quad (3-10), (3-11), (3-12)$$

The only difference between the constraints here and those for the original formulation (3-1) through (3-12) is that (3-4), (3-5), and (3-8) have been replaced by (3-21).

Some of the major difficulties in minimizing  $Z_2$  in (3-20) arise from the vehicle capacity restriction on each route arc and the consideration of waiting and transfer time costs. These items are not represented by the guideway network and hence usually not incorporated into a network solution. Mathematically the difficulties manifest themselves by the fractional term  $v_m^k / y_m$  in the objective function and the constraints regarding vehicle capacities, waiting times, and transfer times.

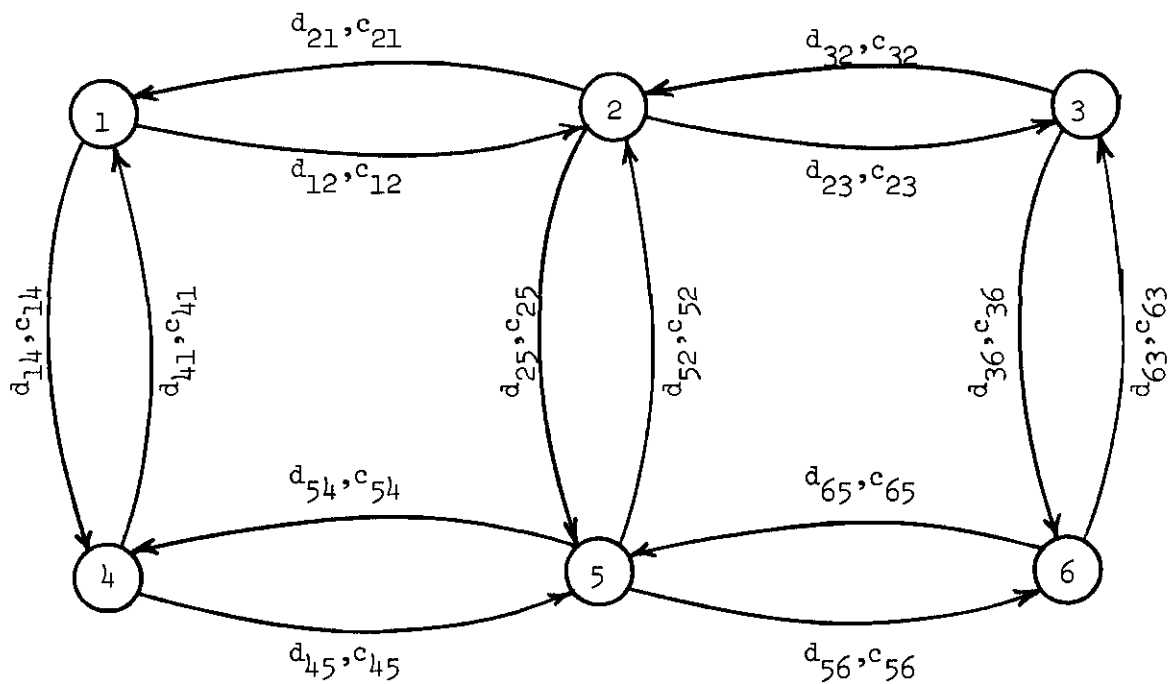
The method used to overcome these obstacles is to construct a revised network wherein each route is represented by a set of nodes and arcs, and boarding and transfer occurrences are represented by dummy nodes and arcs. When all the nodes and arcs are properly connected, this revised network will be equivalent to the original guideway network with its set of vehicle routes.

Consider the network in Figure 4, containing six stations and seven two-way guideways. There are three designated routes: reversal runs a and b, and the loop c. Frequency of service per operating time period is given by  $y_a$ ,  $y_b$ , and  $y_c$ . Passenger travel time costs  $d_{ij}$  and vehicle operating costs  $c_{ij}$  are shown for each guideway arc.

A new network is constructed according to the following rules:

(1) For each station  $i$  in route  $m$  designate a route node  $im$ , without duplicating stations in the same route.

(2) Within each route  $m$  connect the nodes  $im$ ,  $jm$ , ... by route arcs corresponding to the route in the original network. The



| Route | Stations      | Service    |
|-------|---------------|------------|
| a     | 1-2-3-2-1     | $Y_a = 10$ |
| b     | 1-4-5-6-5-4-1 | $Y_b = 4$  |
| c     | 3-6-5-2-3     | $Y_c = 8$  |

Figure 4. Original Network.

length of each route arc  $(im, jm)$  is the travel time cost  $d_{ij}$  of the corresponding arc  $(i,j)$  in the original network. The capacity of each arc  $(im,jm)$  is  $y_m g$ .

(3) For each station  $i$  of the original network designate an origin node  $io$  and a destination node  $iz$ .

(4) From each new origin node  $io$  extend an arc to all route nodes  $im$  of the same original station number. These arcs are boarding arcs, of length  $d_w/y_m$ , and uncapacitated.

(5) From each route node  $im$  extend an arc to the corresponding origin node  $io$ . These arcs are transfer arcs, of length  $d_t$ , and are uncapacitated.

(6) From each route node  $im$  extend an arc to the corresponding destination node  $iz$ . These arcs are terminal arcs, of length zero, and uncapacitated.

(7) Where a net supply of commodity  $k$  exists at station  $i$  of the original network, designate this supply to be at node  $io$  of the new network. Likewise, a net demand at station  $i$  becomes a net demand at node  $iz$ .

Applying these rules to the network in Figure 4 results in the revised network in Figure 5. For example, route a has three route nodes,  $1a$ ,  $2a$ , and  $3a$ , and 4 route arcs. From origin node  $10$  there are boarding arcs to nodes  $1a$  and  $1b$ , of length  $d_w/10$  and  $d_w/4$ , respectively; and from  $1a$  and  $1b$  there are transfer arcs coming back to  $10$ , and terminal arcs leading to destination node  $1z$ .

The purpose of this revised network is to represent as variable

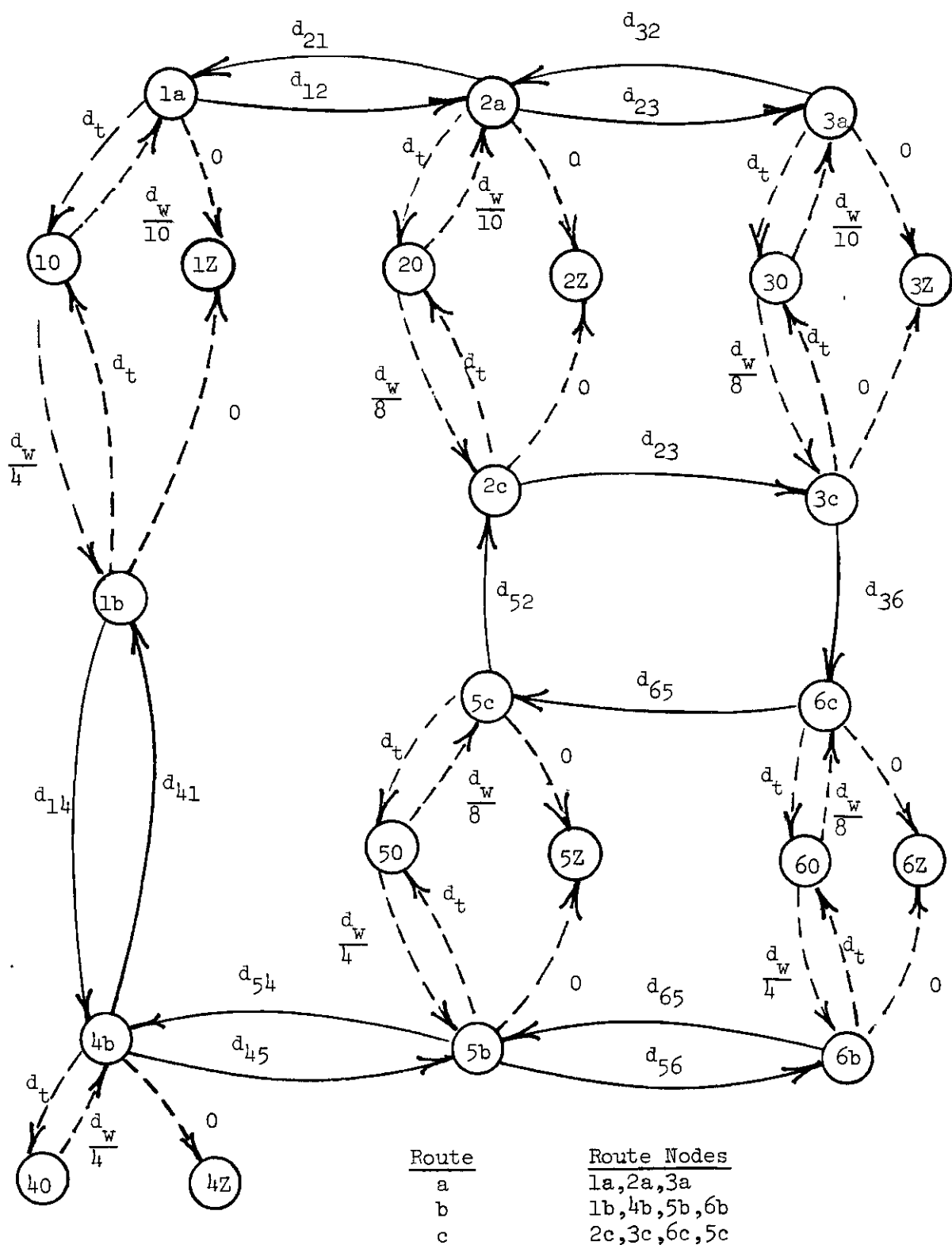


Figure 5. Revised Network.



arc flow costs the passenger waiting time cost  $d_w/y_m$  and the transfer time and nuisance cost  $d_t$ . Consider a passenger traveling from station 1 to 6 in the original guideway network of Figure 4. On the revised network of Figure 5 he can travel by the direct route b: 1o-1b-4b-5b-6b-6z, incurring an average waiting time cost of  $d_w/4$  and travel time costs of  $(d_{14} + d_{45} + d_{56})$ . Or he may select route a between stations 1 to 2 and then route c between stations 2 and 6: 1o-1a-2a-2o-2c-3c-6c-6z, incurring average waiting time costs of  $d_w/10$  for route a and  $d_w/8$  for route c, travel time costs of  $(d_{12} + d_{23} + d_{36})$ , and also a transfer cost of  $d_t$ . Another possibility is route a between stations 1 and 3 and route c between 3 and 6. In a large network the selection of a least-cost path is not obvious, and the assignment of all passenger commodities to achieve minimum total cost, subject to vehicle capacity restrictions, becomes a large and difficult problem.

In the revised network in Figure 5 all the waiting time, travel time, and transfer costs are represented explicitly as variable arc flow costs. The selection of a least-cost path for a single commodity then becomes a shortest path network problem, and the assignment of all passenger commodities becomes a capacitated, minimum cost multicommodity network flow problem. The advantage here is that efficient solution techniques exist for such problems (15, 45).

#### Route Insertion-Deletion

So far nothing has been said about the vehicle amortization and operating costs  $c_{ij}$ , which are included in the objective function (3-20).

These  $c_{ij}$  are considered to be fixed costs for inserting route arcs into the revised network. For example, in Figure 5 the fixed arc costs are

$$\text{Route a: } 10(c_{12} + c_{23} + c_{32} + c_{21})$$

$$\text{Route b: } 4(c_{14} + c_{45} + c_{56} + c_{65} + c_{54} + c_{41})$$

$$\text{Route c: } 8(c_{36} + c_{65} + c_{52} + c_{23}).$$

Boarding, transfer, and terminal arcs have zero fixed cost since there is no vehicle cost associated with those travel activities.

In this framework, then, the problem of minimizing Z2 in (3-20) subject to the constraints becomes another fixed-charge, capacitated, multicommodity flow problem: select a set of fixed-cost arcs for inclusion in a network to minimize the total of fixed plus variable costs. Again, an arc insertion-deletion algorithm may be used to obtain a solution. The differences between the simplified problem given by (3-13) through (3-17) and the present problem are the number of arcs involved and the logic for maintaining a meaningful route structure. Hence, the arc improvement parameters lead to route improvement parameters, and arc insertion and deletion become route insertion and deletion and route modification.

Adapting an arc insertion-deletion algorithm to the situation of handling routes results in the conceptual route insertion-deletion algorithm shown in Figure 6. Again, the details of the algorithm and computation of improvement parameters are left for the next chapter.

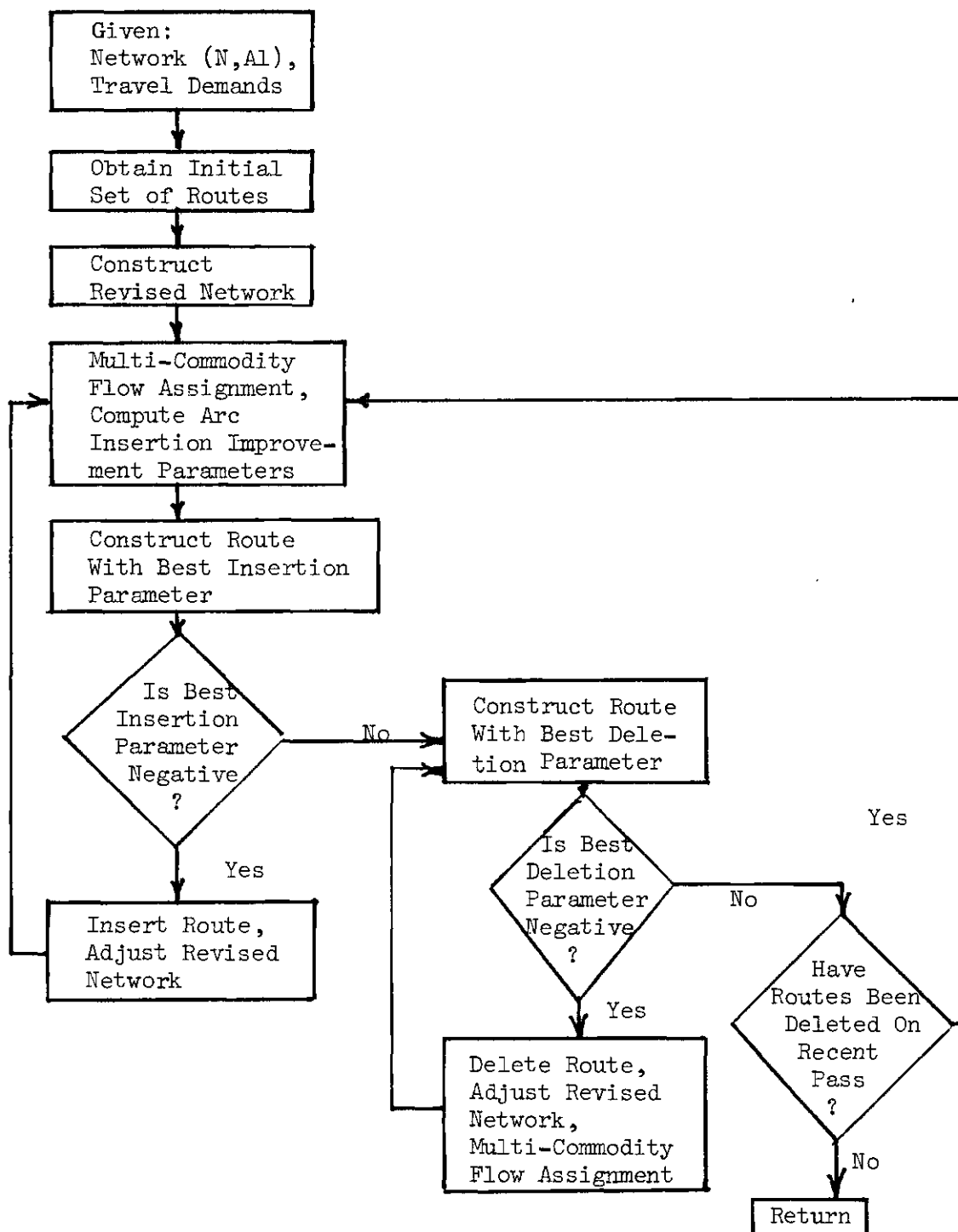


Figure 6. Conceptual Route Insertion-Deletion Algorithm.

## CHAPTER IV

## DETAILED DESCRIPTION OF SOLUTION PROCEDURE

Penalty-Cost Multicommodity Flow Assignment RoutineConceptual Version

The use of the two insertion-deletion algorithms described previously will require numerous executions of a capacitated, minimum cost multicommodity flow routine. At the same time one would prefer to do a minimum of work in determining the improvement parameters. The penalty-cost flow assignment routine was selected to achieve both of these objectives.

The problem to be solved by this flow assignment routine is the following:

$$\text{Minimize } Z_3 = \sum_{k \in K} \sum_{(i,j) \in B} f_{ij}^k d_{ij} \quad (4-1)$$

Subject to

Flow requirements:

$$\sum_{j \in A(i)} f_{ij}^k - \sum_{j \in B(i)} f_{ji}^k = r_i^k, \quad k \in K, \quad i \in N \quad (4-2)$$

Arc capacities:

$$\sum_{k \in K} r_{ij}^k \leq b_{ij}, \quad (i,j) \in B \quad (4-3)$$

Nonnegativity:

$$r_{ij}^k \geq 0, \quad k \in K, \quad (i,j) \in B \quad (4-4)$$

where  $B$  is the set of arcs in the network, and all other terms are as previously defined.

The conceptual version of this routine may be described by the following steps, also shown in Figure 7:

- (1) Set all arc infeasibility parameters to zero.
- (2) In the given network  $(N,B)$  assign each commodity to its shortest path, disregarding arc capacities. Compute total flow on each arc.
- (3) Determine the set  $B_1 \subset B$  of infeasible, or oversaturated, arcs. If there are none, or  $B_1 = \phi$ , stop.
- (4) For the set  $B_1$  of infeasible arcs increase the length of each arc by one unit  $e$ .
- (5) For each arc in the set  $B_1$  add to the previous infeasibility parameter the quantity - (excess flow  $\times e$ ). Return to step 2.

If there is a feasible solution to the multicommodity flow assignment problem with no split commodities, the above procedure will normally terminate after a finite number of steps. The case with split commodities can be handled by modifying step 2.

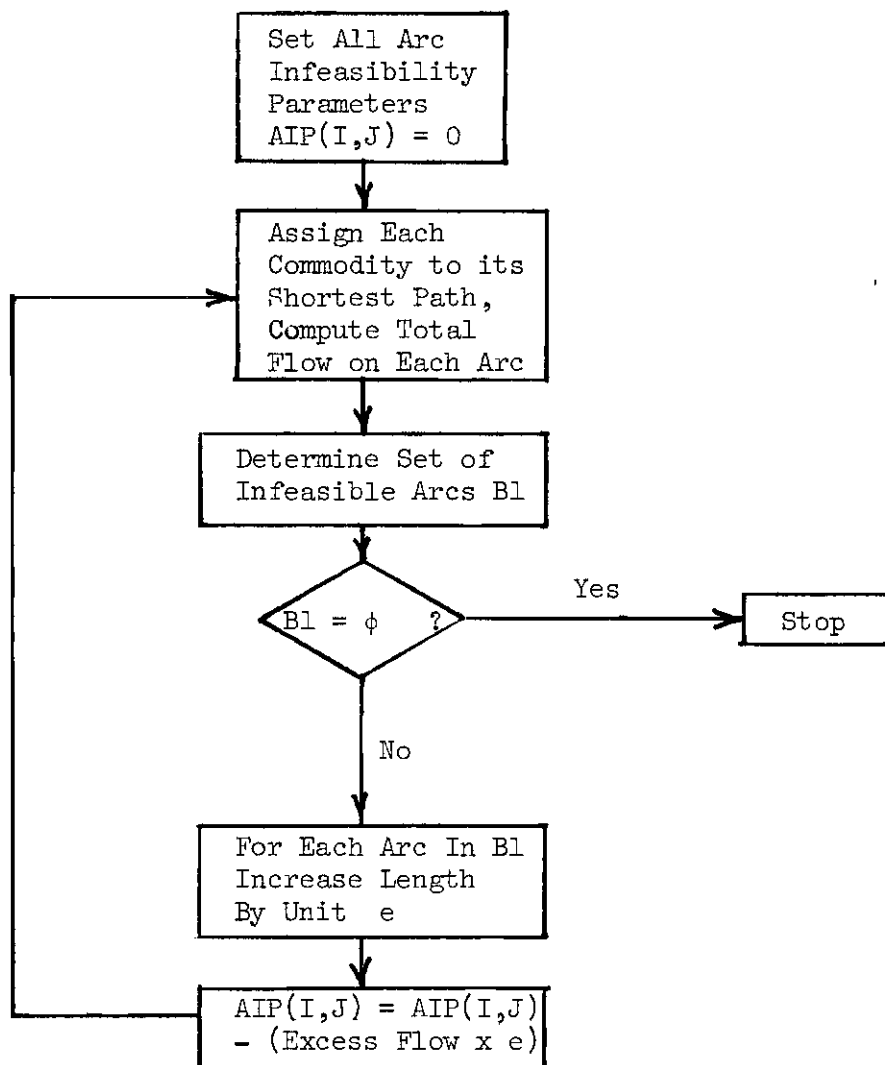


Figure 7. Penalty-Cost Multicommodity Flow Assignment Routine, Conceptual Version.

The flow assignment so obtained will be a good, but not necessarily optimal, solution to the multicommodity flow assignment problem. The infeasibility parameter for each arc will represent the cumulative infeasibility of that arc during the process of changing from an initial infeasible flow assignment to a final feasible assignment.

#### Actual Version

An interesting feature of the solution procedure is the following: the commodity to be reassigned next is the one with the smallest difference between an infeasible shortest path and a more feasible (or less infeasible) second shortest path. By itself, this feature is desirable from the standpoint of modeling a user-oriented transit system.

More important, however, is the fact that it leads to an efficient implementation version of the penalty-cost assignment routine:

- (1) Set all arc infeasibility parameters to zero.
- (2) In the given network  $(N, B)$  assign each commodity to its shortest path, disregarding arc capacities. Compute total flow for each arc.
- (3) Determine the set  $B_1 \subset B$  of infeasible arcs. If there are none, stop.
- (4) Determine the set  $K_1 \subset K$  of commodities for which the shortest path contains infeasible arcs. If a new commodity enters the set  $K_1$ , flag it. On the first iteration all commodities in the

set  $K_1$  are flagged.

(5) For each commodity  $k \in K_1$ , determine the shortest and second-shortest path lengths,  $p_1$  and  $p_2$ , respectively, the number of infeasible arcs  $n_1$  in the shortest path, and the number of inadequate arcs  $n_2$  in the second-shortest path, such that  $n_2 < n_1$ . An inadequate arc is one that would be infeasible if commodity  $k$  were assigned to it. The  $p_1$  is available from step 2 or step 11, the  $n_1$  from step 4. The  $p_2$  remains unchanged from the previous iteration for unflagged commodities. For flagged commodities a second-shortest path must be computed. If the second-shortest path does not contain fewer inadequate arcs than there are infeasible arcs in the shortest path, select the third-shortest path, etc. If there is no  $n^{\text{th}}$ -shortest path such that  $n_2 < n_1$  for all  $k \in K_1$ , then the network is infeasible.

(6) For each commodity  $k \in K_1$ , compute

$$u_k = \frac{p_2 - p_1}{n_1 - n_2} . \quad (4-5)$$

(7) Select  $k^* \in K_1$  such that  $u_{k^*}$  is minimized. Let  $\text{eps} = u_{k^*}$ . Commodity  $k^*$  becomes the next commodity to be reassigned.

(8) For the set  $B_1$  of infeasible arcs increase the length of each arc by  $\text{eps}$ .

(9) For each arc in the set  $B_1$  add to the previous infeasibility parameter the quantity

$$-(\text{excess flow} \times \text{eps}).$$



(10) Reassign commodity  $k^*$  from its shortest to its second-shortest path, and make the corresponding changes in total flow for each arc included in either of these paths.

(11) For each commodity  $k \in K_1$ ,  $k \neq k^*$ , replace

$$p_1 \text{ by } (p_1 + n_1 \times \text{eps})$$

(12) Examine the second-shortest paths for each commodity  $k \in K_1$ . If  $n_2 \neq 0$ , flag the commodity. Return to step 3.

Figure 8 shows the flowchart for the above implementation version of the penalty-cost assignment routine. Several of the steps in the routine are critical to minimizing the computations involved:

Step 6 determines, for any particular commodity, the minimum amount  $u_k$  by which each arc in  $B_1$  would have to be increased so that the second shortest path would become the shortest path for that commodity. Assume, for example, that for commodity  $k$  the shortest path has length  $p_1 = 100$  and contains  $n_1 = 3$  infeasible arcs, and that the second shortest path has length  $p_2 = 110$  and contains  $n_2 = 1$  inadequate arc. Then  $u_k = (110 - 100)/(3 - 1) = 5$ . Increasing each arc in  $B_1$  by 5 will result in  $p_1 = 100 + (3)(5) = 115$ , and  $p_2 = 110 + (1)(5) = 115$  or  $p_2 = 110 + (0)(5) = 110$ , depending on whether the one inadequate arc in the second shortest path was already infeasible or not. At this point assigning the commodity to either of the paths would result in the same path length of 115. If the one inadequate arc was not infeasible before, it will be after the commodity is routed

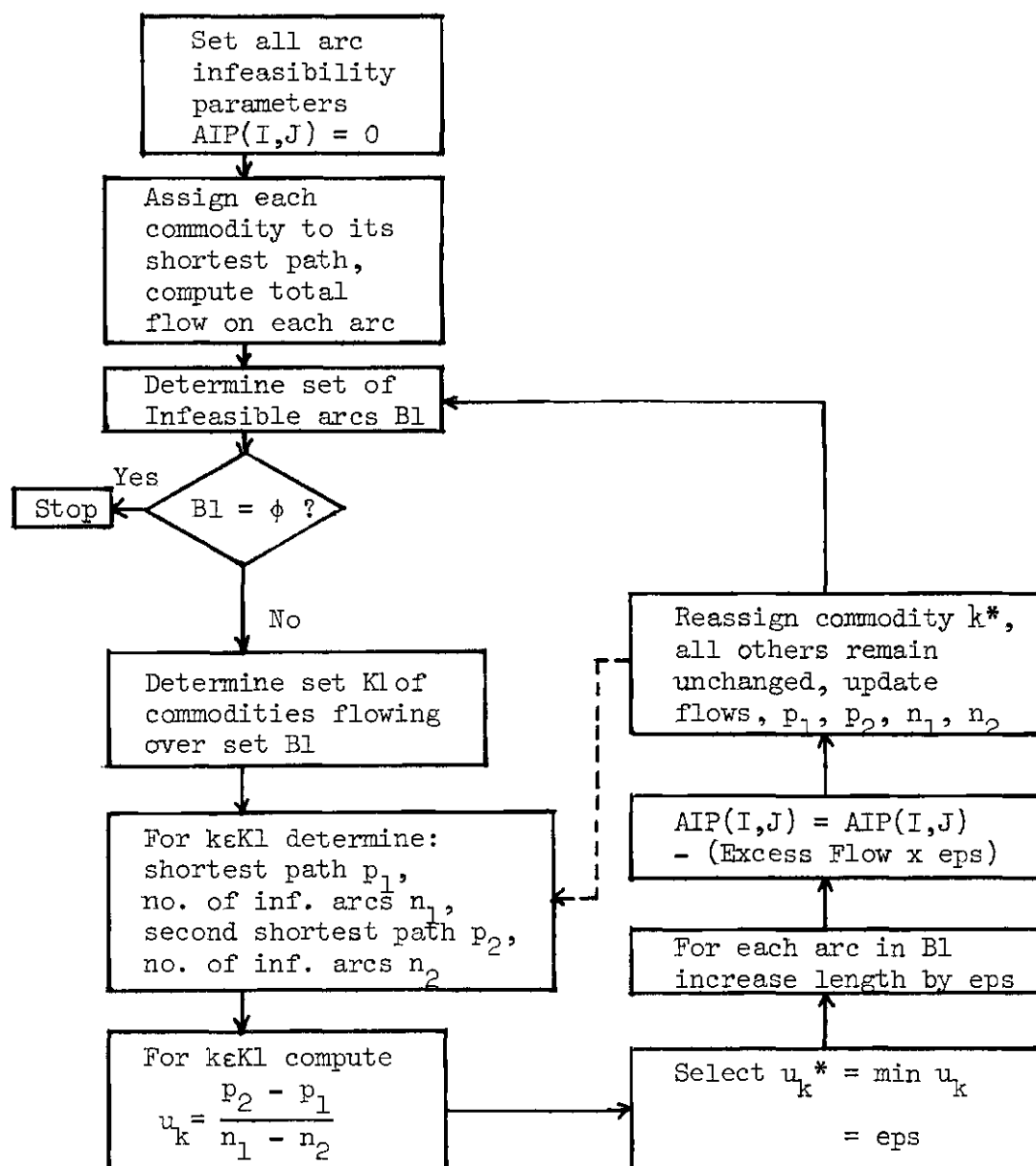


Figure 8. Penalty-Cost Multicommodity Flow Assignment Routine, Actual Version.

on the second shortest path, and hence that arc will have its length increased by  $u_k = 5$ .

In step 7 the minimum of these  $u_k$  is selected, over all  $k \in K_1$ . This  $k^*$  then represents the commodity that would be first induced to change to a more feasible second shortest path if each arc in  $B_1$  were incremented repeatedly by a small amount  $e$ , as would be done in the conceptual version of the routine.

A more feasible second shortest path is obtained using rather simple procedures. If the number of infeasible arcs in the shortest path is  $n_1 = 1$ , increase temporarily the length of the one infeasible arc by some large number  $E_1$ , and then find the shortest path in the resulting network.

If  $n_1 > 1$  then the penalty cost concept is applied again, on a smaller scale. Increase temporarily the length of each infeasible arc in the shortest path by some small number  $e_1$  and find the shortest path in the resulting network. If the second shortest path found in this manner is not more feasible, or  $n_2 \geq n_1$ , increase temporarily each inadequate arc in this path by  $e_1$ , and, retaining the previous temporary arc length increases, again find the shortest path in the resulting network. Repeat the above procedure, retaining all temporary arc length increases, until a more feasible  $n^{\text{th}}$ -shortest path is found.

Excessive computation of such paths can be avoided by observing that if

$$u_k' < \frac{p_n - p_1}{n_1}, \quad (4-6)$$

where

$u_k'$  is the current best value of  $u_k$ ,

$p_n$  is any  $n^{\text{th}}$ -shortest path found for commodity  $k$  according to the rules given above,

then

$$u_k' < \frac{p_n - p_1}{n_1} \leq \frac{p_2 - p_1}{n_1 - n_2} \quad (4-7)$$

where  $p_2$  is the first more feasible  $n^{\text{th}}$ -shortest path found. In such a case commodity  $k$  cannot become  $k^*$  and the routine can continue to the next commodity.

Updating the stored information is quick since only commodity  $k^*$  changes its path. For all other commodities  $k \in K_1$  the path lengths increase by  $(n_1 \times \text{eps})$ . Next are determined the new values of  $n_1$  and  $n_2$ . If  $n_2 = 0$ , the second shortest path remains the same, as does its length  $p_2$ . If  $n_2 \neq 0$ , some other path may become shorter during the next iteration, and the commodity is flagged for such recomputation. Likewise, if the commodity was dominated as a result of (4-6), then its second shortest path must also be recomputed during the next iteration. It may happen that, as a result of reassigning commodity  $k^*$ , some other commodity drops from the set  $K_1$ , in which case even greater computational savings are achieved.

Occasionally a commodity in the set  $K_1$  will find its only feasible path blocked by some other commodity that has yet to be reassigned. If the blocked commodity is not dominated by the criterion

of (4-6), perhaps because it is first in the list of  $K_1$ , then it is declared temporarily disabled and removed from the set  $K_1$ , its  $n_1$  is set equal to 0, and it is allowed to remain on its infeasible path. The criterion for such action is if the second shortest path length  $p_2$  exceeds some large number  $E_2$ , where  $E_2 > E_1$ .

When the set  $K_1$  becomes empty, all such temporarily disabled commodities are placed back in  $K_1$ , and the routine is started again. This action is repeated as often as necessary until all commodities are assigned to feasible paths. If no commodity can be reassigned to a more feasible path and no commodity can be removed from the disabled list, the network is declared infeasible. In such a case oversaturated arcs at the final iteration will have their infeasibility parameters incremented by a large, negative number.

Summarizing, the penalty-cost routine obtains a good, but not necessarily optimal, solution to the minimum cost multicommodity flow problem. An important by-product is the computation of the arc infeasibility parameters. This parameter represents the cumulative infeasibility of an arc during the process of changing from an initial infeasible flow assignment to a final feasible assignment, measured in terms of variable arc flow costs.

#### Route Insertion and Route Insertion Parameters

The insertion portion of the route insertion-deletion algorithm contains three major phases:

- (1) Increasing service on an existing route
- (2) Appending guideway arcs to existing routes

(3) Route construction.

In addition, a number of subroutines, bookkeeping operations, and network logic considerations play a significant role:

- (4) Route configurations allowed
- (5) Initial adjustments to the set of routes
- (6) Constructing the revised network
- (7) Computation of arc insertion improvement parameters.

For convenience of exposition the latter are discussed first.

Route Configurations Allowed

The number of possible routes in even a small network quickly reaches unmanageable proportions. For example, in a maximally connected network with 10 stations the total number of possible routes of length 10 arcs or less is 25,000. In order to limit the computational burden routes are constrained with respect to length, form, and starting point.

A route may be either a loop or a reversal run. A loop route must have exactly one arc incident to and exactly one arc incident from each station included in the route. A reversal run is simply a route which doubles back on itself. Frequency of route service on a loop must be the same on all arcs of the loop. A reversal run may have multi-zone service, on the other hand. Consider the reversal run in Figure 9, starting at node 1: service between stations 1 and 2 is 15 per time period, between stations 2 and 3 it is 12 per time period, and between 3 and 5 it is 10. This implies that during one time period there are 10 vehicle trips from station 1 to 5 and

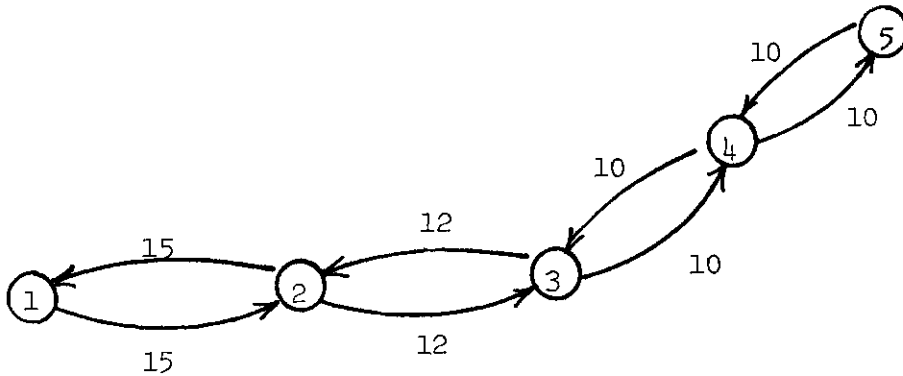


Figure 9. Reversal Run Route.

back, two vehicle trips from 1 to 3 and back, and three vehicle trips from 1 to 2 and back. As mentioned previously, vehicles stop at all intermediate points.

A route is constrained to begin and end at designated major stations, which are predetermined by the transportation planner and read as input data. Thus, a loop route must contain at least one major station, while a reversal run must have a major station at one end, with the more frequent service operating from that station.

Besides reducing the computational burden of the algorithm, there are at least two physical reasons for this restriction. First, the actual timetable scheduling of vehicles in any system will involve the use of the same vehicle on different routes. Hence, it will be easier to connect various routes for traversal by one vehicle if there are a limited number of starting and end points for routes. Second, the starting and end points of routes will normally require some means for storage of idle vehicles during the off-peak operating

hours. Stations with such storage capacity are usually limited in a transit system, for reasons of fixed cost, space, personnel facilities, etc.

Any method for selecting major stations results in more routes serving those particular stations and hence invariably biases the final solution. Among the possible criteria the transportation planner might use are: (1) the number of guideways incident at a station, (2) the net travel demand at a station, and (3) the net flow across a station on unconstrained shortest path assignment of all commodities.

#### Initial Adjustments to the Set of Routes

The guideway insertion-deletion algorithm, the calling program, provides the initial set of routes, which is the best set of routes from the previous iteration of the guideway algorithm. For the 0<sup>th</sup> iteration of the guideway algorithm a set of routes is provided in the input data.

The routes are first checked for feasibility with respect to guideways. If any route covers a closed guideway then that route must be adjusted. If it happens to be a loop route it is necessarily deleted entirely. Reversal run routes are either deleted or shortened, depending upon where the guideway occurs within the route and the availability of a major station for use as a starting point of the shortened route.

Second, a check is made to ascertain that all open guideways are covered by at least one route. If this is not the case, then a zero-level route is established for each uncovered guideway. Such a



zero-level route is a reversal run route two arcs long and with zero frequency of service. This procedure is necessary in order to obtain insertion improvement parameters for regular routes that may eventually cover that guideway.

### Constructing the Revised Network

After the initial adjustments have been made to the routes, the revised network is constructed according to the rules given in Chapter III.

In the case of a reversal run with multi-zone service the length of the boarding arc, representing the average waiting time for a vehicle on the route, is determined by an averaging process: the number of vehicles  $y_m$  per period on the route is taken to be the average of the number of vehicles per period serving each station in the route. The boarding arc length is then computed in the usual manner as  $d_w/y_m$ . Any attempt at a more precise determination of average waiting times would depend upon the paths of the commodities, which are unknown at this point. For the same reason no attempt is made to model the situation where the average waiting time may be less for a commodity as a result of two or more routes being available.

The computation of average waiting times as  $d_w/y_m$  is intended for uniformly distributed travel demand for each commodity. This seems well suited for peak-hour transit needs, when service is frequent and passenger arrivals at stations are independent of vehicle timetables. For infrequent service a maximum average waiting time is used on the assumption that a passenger times his arrival at a station.

Other situations may be modeled appropriately.

Following the construction of the revised network, the penalty-cost multicommodity routine assigns the flow and produces the arc infeasibility parameters.

#### Arc Insertion Improvement Parameters

At each iteration of the multicommodity assignment routine the infeasibility parameter of each arc is incremented by  $-(\text{excess flow} \times \text{eps})$ . The cumulative result of this process provides a rough measure of the potential savings in variable flow costs to be achieved by increasing the capacity of the arc.

For example, assume the shortest path for commodity  $k$ , with travel demand 100, contains  $n_1 = 1$  infeasible arc, carrying an excess flow of 100, and that the second shortest path is  $p_2 - p_1 = 20$  units longer. Incrementing the one infeasible arc by 20 units will cause the commodity to switch to the feasible second shortest path, at an increase in total variable flow costs of  $(20)(100) = 2000$  units.

The incrementation of the arc infeasibility parameter records all such potential savings, in an approximate manner. If the shortest path contains  $n_1 > 1$  infeasible arcs, then each of these needs to be incremented only by  $(p_2 - p_1)/n_1$  to cause the commodity to switch paths. In such a case the same potential savings are shared by the  $n_1$  arcs. All these matters are automatically considered in the computation of  $u_k$ , the selection of  $u_k^* = \text{eps}$ , and the incrementation of the arc infeasibility parameters.

Increasing the capacity of an arc, from either zero or a

positive level, will increase the fixed costs. However, it is likely that by rerouting flow some fixed-cost savings may be achieved elsewhere in the network. Hence, the arc infeasibility parameters provide by themselves a priority ranking for increasing arc capacities. Thus, in the route insertion-deletion algorithm, referring to arcs in the revised network, arc insertion improvement parameters are defined to be the arc infeasibility parameters.

Route insertion improvement parameters are then obtained by summing the insertion parameters for arcs contained in the route. Since boarding arcs, transfer arcs, and terminal arcs are uncapacitated, no parameters are obtained for these. Since it may be desirable to construct a completely new route, it is also useful to cumulate the arc improvement parameters to guideways.

In determining how much capacity increase should be made for an arc one needs to know the maximum excess flow, or the additional capacity desired, on the arc during the multicommodity assignment routine. This information is obtained for each arc along with the incrementation of the arc infeasibility parameter. At the end of the routine this additional capacity desired is also cumulated to guideways.

#### Increasing Service on Existing Routes

The first method of "inserting" a route to increase network capacity is to increase the service on an existing route. The route to be increased is selected as that one with the best insertion improvement parameter. For loop routes the route parameter is obtained

by summing the arc parameters for all arcs in the route.

It is possible, however, to increase only a section of a reversal run route. Referring again to Figure 9, service increases can be made on the following sections:

1-2, 1-2-3, 1-2-3-4, 1-2-3-4-5,  
2-3, 2-3-4, 2-3-4-5,  
3-4, 3-4-5.

Increases on sections beginning at 1 may be of any magnitude.

Increases on sections beginning at 2 are limited to three vehicles, since the service between 2 and 3 cannot exceed 15, the service on the next branch closer to the major station. Likewise, the sections beginning at 3 cannot be increased by more than two vehicles per period, and no increase is possible on the section between 4 and 5.

The route insertion improvement parameters are determined for all sections of a reversal run route. If two sections of unequal length have the same improvement parameter, the shorter section is preferred since it will result in a smaller fixed-cost increase for the same indicated route improvement parameter.

Once the route or section has been selected, it is subjected to a preliminary test of the route improvement parameter versus a cut-off value, performed to limit the number of iterations in this section. Next, the service on the route or section is increased by one vehicle per time period, a revised network is constructed, the multicommodity routine is entered to assign the flow, and total fixed

and variable costs are obtained.

If there is overall improvement, the new insertion improvement parameters are obtained, and the next route or route section is selected for a service increase. Overall deterioration leads to the reestablishment of the previous set of routes, service frequencies, flow assignment, and insertion improvement parameters. The algorithm then attempts to append arcs to existing routes, as described later. Figure 10 depicts the general relationships among the steps described above.

If the previous set of routes was infeasible, then overall costs will deteriorate because of the increase in fixed costs. However, the increased service will normally reduce infeasibility, and the change is therefore accepted. In the case of an infeasible network those arcs that are oversaturated at the final flow assignment will have very low, or extreme, values for infeasibility parameters. Hence, those arcs will tend to have their capacity increased first.

#### Appending Guideway Arcs to Existing Routes

This phase of the algorithm tries to extend existing reversal run routes by appending pairs of arcs to them. Such route lengthening is allowed only at the end away from the major station in order to maintain a consistent multi-zone route structure.

Here the guideway insertion improvement parameters are used to select the guideway pair with the best sum of parameter values. Then a preliminary test is made against the fixed cost for the guideway pair. Next, a route is located to which the pair may be appended. If there

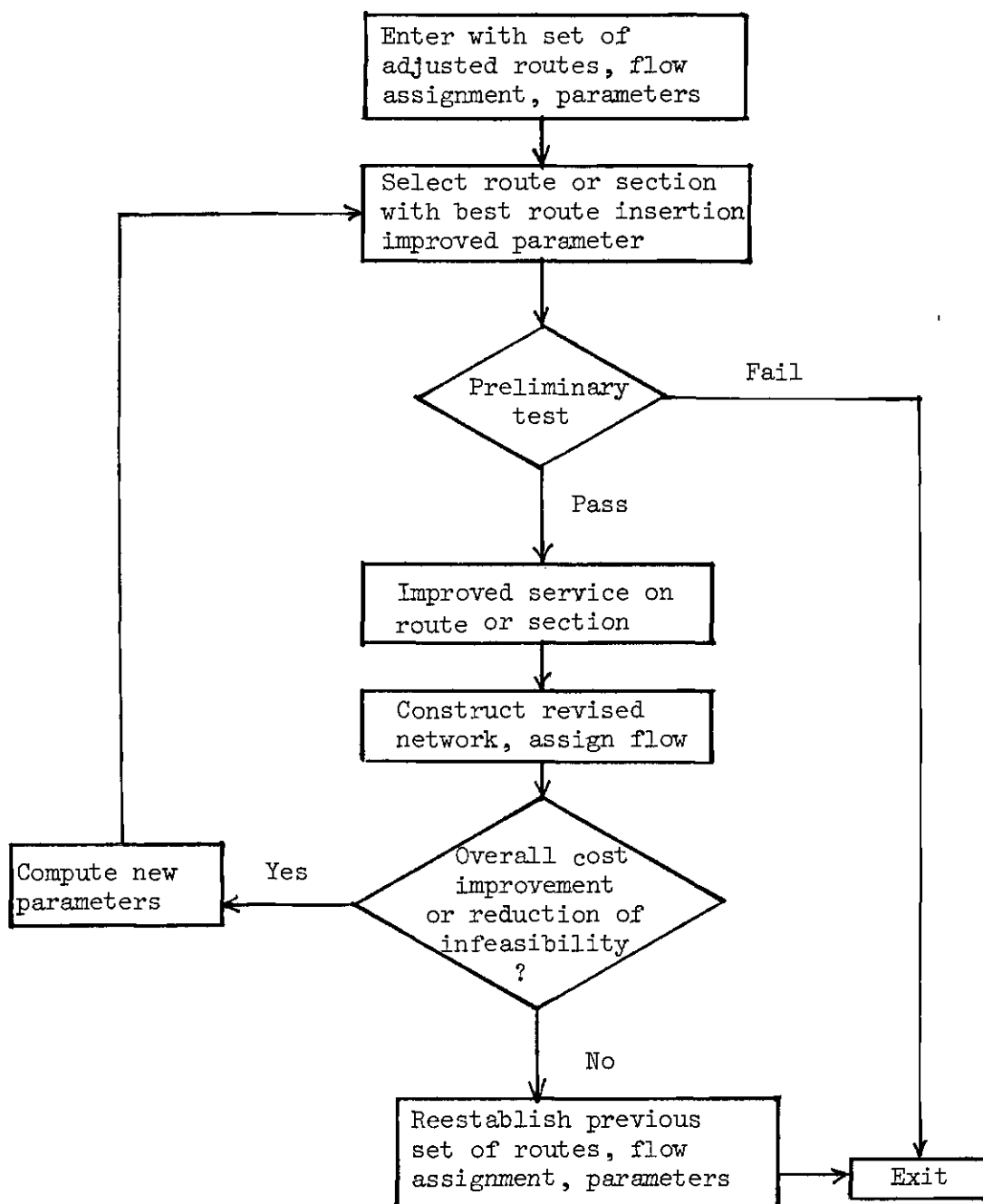


Figure 10. Increasing Service on Existing Routes.

exists more than one such route, the selection process favors the one for which the service frequency, on the last section of the reversal run route, most closely approaches the capacity desired on the new section.

Since zero-level routes constitute an artificial device to obtain improvement parameters, they are not considered for lengthening. Anyway, lengthening a zero-level route would achieve no increase in capacity since the service on the new section could not exceed the zero service frequency on the old section.

If the guideway pair can be appended, then any corresponding zero-level route is dropped. The revised network is constructed, the multicommodity routine is entered to assign the flow, and total costs are computed.

An overall cost improvement leads to computation of new parameters and the selection of the next guideway pair. In the case of deterioration the previous routes, flow, and parameters are reestablished, and the guideway pair is placed in a hold list. The next guideway pair is then selected.

If a guideway pair cannot be appended to a route, it is also placed in the hold list, and the next pair is selected. Any time the appending of a pair results in an overall cost improvement, the hold list is cleared of all entries.

Figure 11 shows the flowchart for this phase of the algorithm. The phase is exited either by failing the preliminary test or by exceeding the allowable length of the hold list.

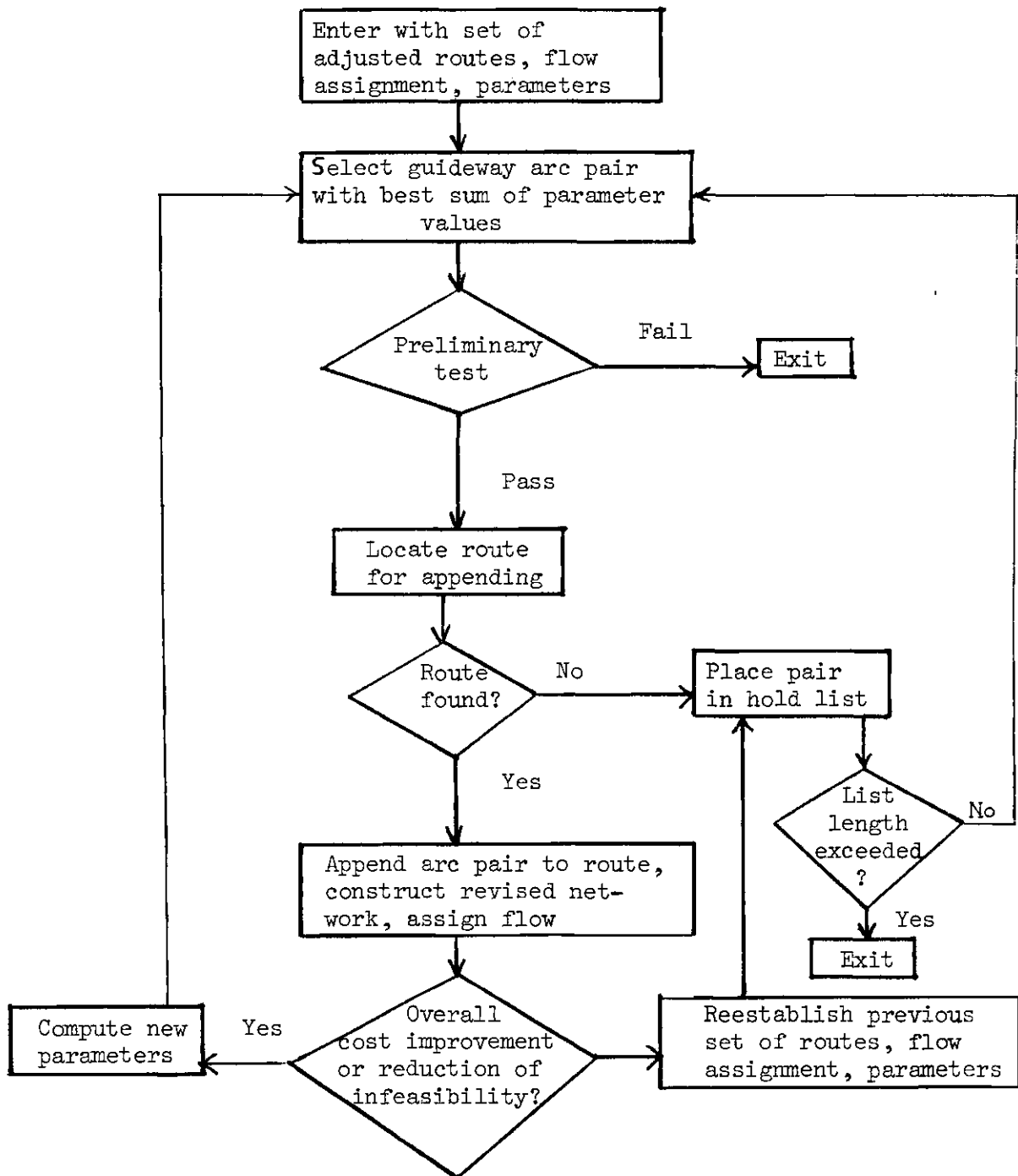


Figure 11. Appending Guideway Arcs to Existing Routes.



### Route Construction

A third method of inserting a route into the network is to construct an entirely new route. The idea is to select guideway arcs for the new route so that the resulting route improvement parameter has a large, negative value.

The route is constructed by a routine which builds a cycle of the most negative length on the graph of open guideway arcs. The length of each arc in the cycle is defined to be:

$l_{ij}$  = guideway insertion improvement parameter, if negative; otherwise,

$l_{ij}$  = vehicle operating cost  $c_{ij}$   
- savings in passenger transfer time costs.

If the insertion parameter is negative, it already reflects the major change in total costs expected from inserting the arc into a route. If not, no variable cost savings are expected, and the measure of change in total costs due to inserting the arc is the vehicle operating cost less the savings in passenger transfer costs.

The route insertion improvement parameter for the cycle so constructed is simply the length of the cycle. In this phase of the algorithm, then, a route parameter also reflects the potential savings in transfer time costs between guideway arcs for which the infeasibility parameters are zero.

The route constructed must meet the usual restrictions concerning major station starting point, loop or reversal run form, and maximum number of arcs. Accordingly, the route starting point is

selected to be a major station at the end of a guideway arc with an extreme parameter value.

The negative cycle is constructed in two steps. First, an out-tree is constructed using the route starting point as the origin. The rules governing tree construction are the same as for Whiting and Hillier's minimum path tree algorithm (50), except that no cycles are permitted at this stage. The tree is constructed by successively adding to the tree the node closest to the origin.

In executing this part of the routine the origin node is initially labeled with the permanent value zero, and all other nodes are tentatively labeled with a value of infinity. The terminal node of each arc emanating from the origin node is then assigned a new label based on its distance from the origin node and is entered in the cumulative table. The labels of all nodes in the table are compared, the smallest label is declared permanent, and the node possessing this label is entered in the tree. The terminal node of each arc emanating from the newly entered node is entered in the cumulative table and tentatively relabeled with the smaller of the following two values: (1) its existing label, and (2) the sum of the newly entered node label and its distance from this node. If a node is already in the permanent tree, then it is not entered in the cumulative table, thus avoiding cycles. The minimum of the labels in the cumulative table is then declared permanent, the node possessing this label is added to the tree, and the relabeling process is repeated. If routes are limited to  $b$  arcs in length, the out-tree construction process

continues until all nodes within  $b/2$  arcs from the origin are permanently labeled.

The second step of the negative cycle routine is to take each of the nodes in this out-tree and find paths back to the route starting point. For reversal runs this implies doubling back along the corresponding guideway arcs. For loop routes the procedure is to construct, for each node  $j$  of the out-tree, an in-tree with node  $j$  as its origin. The rules governing tree construction are the same with one additional restriction: the in-tree may not contain any node that is an intermediate node in the out-tree path from the route starting point to node  $j$ . The out-tree path from the starting point to  $j$  is then combined with the in-tree path from  $j$  to the starting point to form a loop route. The restriction regarding intermediate nodes in the out-tree path is needed so that the resulting loop route has exactly one arc incident to and exactly one arc incident from each node contained in the route.

The second step thus produces two potential routes for each node in the out-tree. The route with the most negative length, or the best insertion improvement parameter, is then selected for testing.

Since transfer time savings depend upon the particular route being constructed, they cannot be determined in advance and must therefore be computed at each stage of the tree building process. The savings for placing two adjacent guideway arcs in a route are

$$\begin{bmatrix} \text{transfer} \\ \text{time} \\ \text{savings} \end{bmatrix} = \begin{bmatrix} \text{total flow of all commodities} \\ \text{flowing over the two guideway} \\ \text{arcs in succession} \end{bmatrix} (d_t). \quad (4-8)$$

If another route already contains the two arcs in succession, then no transfer time savings are attributed, as it is assumed the commodities can flow on that route and indicate any desired capacity increase by the infeasibility parameters. No transfer time savings are attributed to flow not already flowing over the two guideway arcs in succession.

Before the constructed route is inserted into the revised network for testing, it is compared with existing routes. If it matches an existing route or a section of a route, then service is simply increased on the latter route. If not, a check is made to see if any zero-level routes can be dropped, and then the revised network is constructed.

Figure 12 shows the general steps of this section of the algorithm. Success leads back to the phase on increasing service on existing routes, and failure leads to route deletion.

#### Route Deletion and Route Deletion Parameters

##### Eliminating Excess Capacity

One obvious way of reducing the fixed costs  $c_{ij}$  in the revised network without increasing unduly the variable costs  $d_{ij}$ ,  $d_w$ , and  $d_t$  is to reduce service on routes with excess capacity.

The first phase of this process selects the route or route section with the maximum uniform excess capacity. Uniform excess

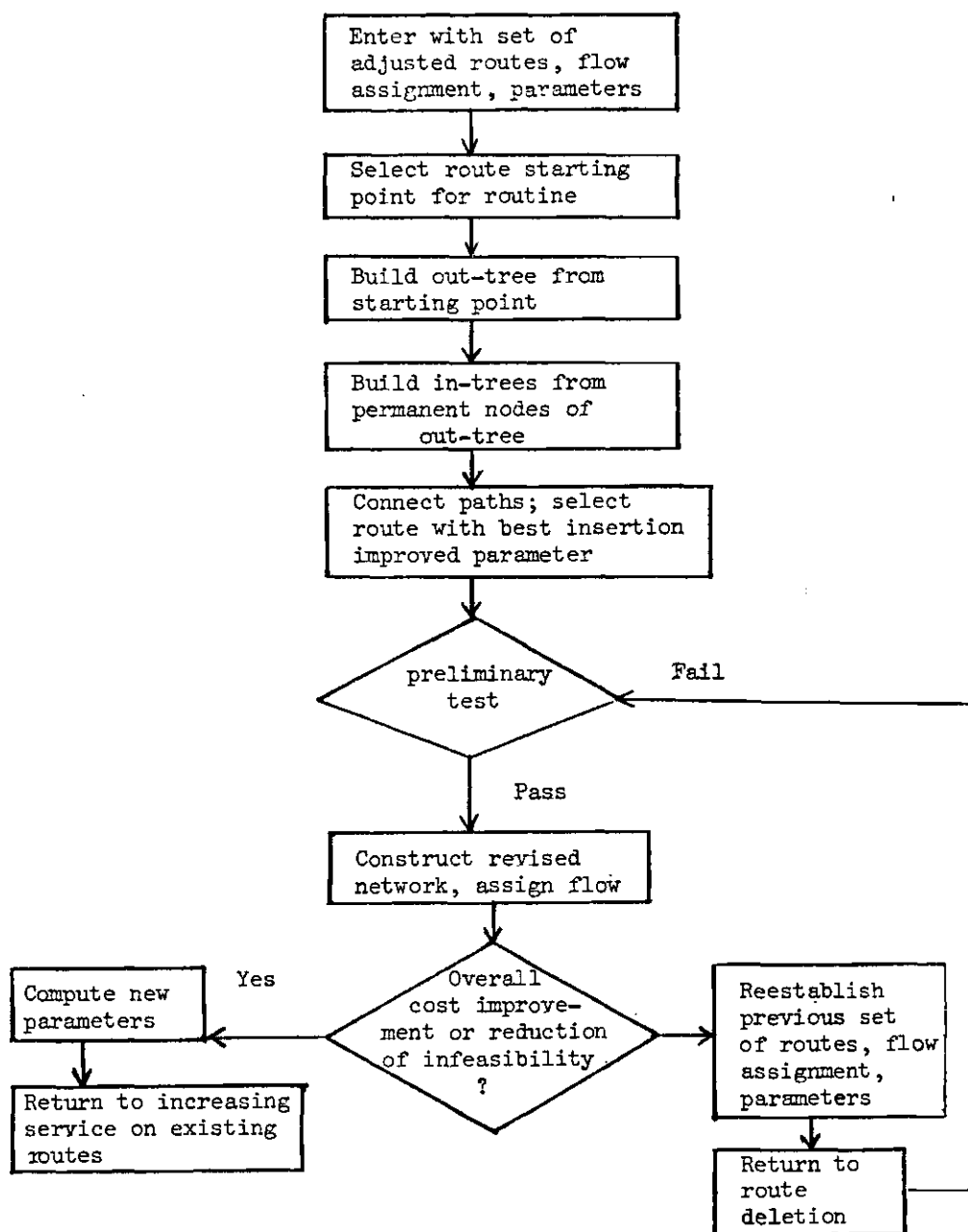


Figure 12. Route Construction Routine.

capacity is defined as the minimum excess capacity of any arc in the route or section. Thus, the selection process obtains

$$\left[ \begin{array}{l} \text{maximum uniform} \\ \text{excess capacity} \end{array} \right] = \begin{array}{c} \text{maximum} \\ \text{over routes} \\ \text{and sections} \end{array} \left[ \begin{array}{l} \text{minimum} \\ \text{over arcs} \\ \text{in route} \\ \text{or section} \end{array} \begin{array}{c} (\text{excess capacity}) \\ \text{of arc} \end{array} \right] \quad (4-9)$$

The route or section so selected then has service reduced on it by the excess capacity computed in (4-9), and the total fixed and variable costs are obtained for the new revised network. If overall improvement occurs, the process is repeated, whereas deterioration leads to the second phase.

In the second phase the route with the maximum total excess capacity is selected, where

$$\left[ \begin{array}{l} \text{maximum total} \\ \text{excess capacity} \end{array} \right] = \begin{array}{c} \text{maximum} \\ \text{over routes} \end{array} \left[ \sum_{\text{arcs}} \begin{array}{c} (\text{excess capacity}) \\ \text{of arc} \end{array} \right] \quad (4-10)$$

Each arc of the selected route has service reduced by this excess capacity. If the effect is to improve overall costs in the revised network, the process is repeated. If deterioration occurs, the algorithm enters phase three and computes arc deletion parameters for the selective reduction of route sections.

#### Arc Deletion Improvement Parameters

While the arc and route insertion improvement parameters are a natural by-product of the multicommodity flow routine, the deletion parameters are not so easily obtained. There is nothing in the flow

assignment routine that measures the potential increases in variable costs due to reducing arc capacities.

One method of obtaining deletion improvement parameters for several arcs at a time is the following: reduce service on each arc of a route by one vehicle per time period, construct the revised network, and obtain the usual arc infeasibility parameters. Then define arc deletion parameters as follows:

$$\begin{aligned} \left[ \begin{array}{c} \text{arc deletion} \\ \text{improvement} \\ \text{parameter} \end{array} \right] &= \left[ \begin{array}{c} \text{-arc infeasibility} \\ \text{parameter} \end{array} \right] (+) \\ &\quad \text{plus} \\ &\quad \left[ \begin{array}{c} \text{saving in vehicle} \\ \text{operating cost} \end{array} \right] (-) \end{aligned} \quad (4-11)$$

The arc infeasibility parameter for each arc is a measure of the potential savings in variable costs to be obtained by increasing the capacity of the arc. Since service on the arc had been reduced, it is also an indication of potential variable cost increases due to reducing capacity. The sign of the negative infeasibility parameter needs to be changed to indicate a potential cost increase. The other element in the deletion parameter is the fixed cost saving.

These arc deletion parameters are then used to determine which section of a reversal run route to reduce. The criterion is to maximize the sum of the arc deletion parameters over the section. The initial selection of a route for computation of deletion parameters is done on the basis of maximum total excess capacity (4-10).

### Overview of Route Deletion

The three phases in the deletion part of the route insertion-deletion algorithm are all similar to each other and generally follow the same sequence of operations. Figure 13 shows the three phases and the relationships among them. Certain preliminary tests in each phase have been omitted from the flowchart for simplification.

Phase 1 selects for testing the route or route section with the maximum uniform excess capacity. Phase 2 selects entire routes on the basis of maximum total excess capacity. Phase 3 examines only reversal run routes, using maximum total excess capacity as the selection criterion.

If phase 2 is exited and the route tested was a reversal run, then all the information is already available for computing arc deletion parameters. On the other hand, if the unsuccessful test in phase 2 involved a loop route, then another reversal run route must be selected for phase 3. Likewise, if the arc deletion parameters indicate that the entire route is to have service reduced, then the results of that operation are available.

When reducing reversal run routes the multi-zone service levels must remain feasible; that is, the more frequent service must occur near the major station. Also, each open guideway must be covered by either one or more regular routes or a zero-level route.

The exit from phase 3 is also the exit from the deletion part of the route insertion-deletion algorithm. If an overall cost improvement has occurred in any of the three phases, then the insertion part



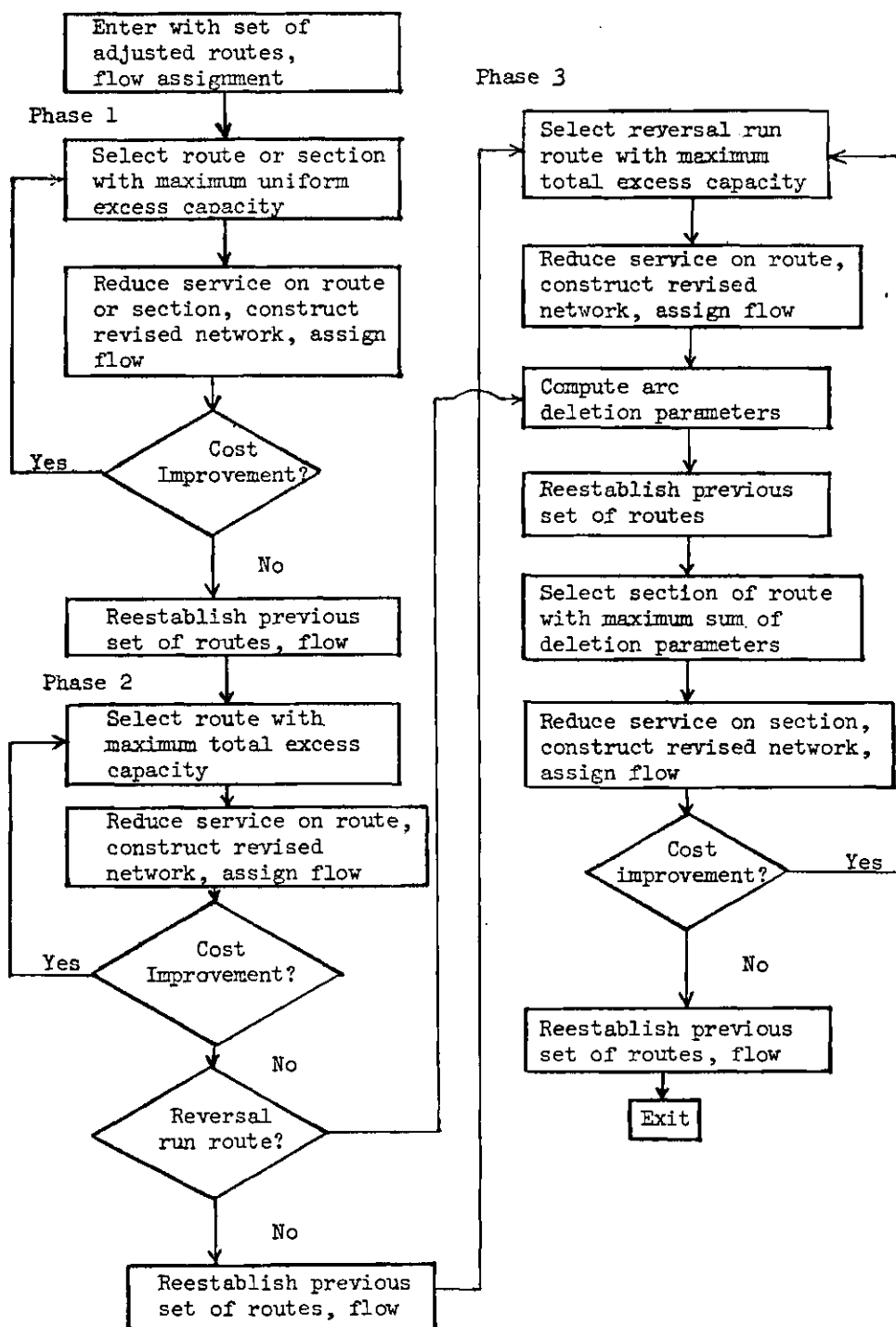


Figure 13. Deletion Part of Route Algorithm.

is entered again. The algorithm alternates between insertion and deletion until no further cost improvement can be made in either part.

### Guideway Insertion

The route algorithm just described determines the best passenger flow, vehicle routes, and route service frequencies for a given set of open guideway arcs  $A_1$ . This algorithm is then imbedded in a guideway insertion-deletion algorithm that makes changes in the set  $A_1$ , as shown in Figure 3.

The method of obtaining guideway insertion improvement parameters is similar to that for route parameters:

- (1) Recall the final set of routes from the previous execution of the route algorithm.
- (2) Establish a zero-level route for each closed guideway.
- (3) Construct a revised network from the routes in 1. and 2.
- (4) Enter the multicommodity routine and assign the passenger flow.
- (5) Cumulate the arc infeasibility parameters to the guideways, thus obtaining guideway infeasibility parameters.
- (6) For each closed guideway define

$$\begin{bmatrix} \text{guideway} \\ \text{insertion} \\ \text{improvement} \\ \text{parameter} \end{bmatrix} = \begin{bmatrix} \text{guideway} \\ \text{infeasibility} \\ \text{parameter} \\ (-) \end{bmatrix} + \begin{bmatrix} \text{fixed cost} \\ p_{ij} \text{ of} \\ \text{guideway} \\ (+) \end{bmatrix} . \quad (4-12)$$

The potential saving in passenger travel and delay time costs

$d_{ij}$ ,  $d_w$ , and  $d_t$  from opening a guideway is indicated by the guideway infeasibility parameter. The other cost element is the guideway fixed cost  $p_{ij}$ , of course.

The guideway pair with the best, or most negative, sum of improvement parameters is then declared open. The route algorithm is used to obtain the best flow, routes, and frequencies, and then the results are evaluated. If there is a decrease in the sum of guideway costs  $p_{ij}$ , vehicle costs  $c_{ij}$ , and passenger time costs  $d_{ij}$ ,  $d_w$ , and  $d_t$ , the guideway insertion was successful and the entire process is repeated. If not, the selected guideway is closed again, the previous set of routes and frequencies is reestablished, and the guideway algorithm proceeds to deletion.

When called, the route algorithm normally begins with the final set of routes from the previous execution. The first operation then is the establishing of a zero-level route for the newly-opened guideway. Then follow the usual steps of increasing service on routes, appending arcs, constructing routes, etc.

#### Guideway Deletion

The difficulties in obtaining route deletion parameters carry over directly to the guideway algorithm. A further complication is the fact that closing a guideway forces readjustments in the existing set of routes.

The process devised for selecting guideways to delete is based on comparing passenger flows and guideway fixed costs:

- (1) Recall the final flow assignment from the previous execution of the route algorithm.
- (2) Cumulate the flow on route arcs to the guideways.
- (3) For each open guideway define

$$\left[ \begin{array}{l} \text{guideway} \\ \text{deletion} \\ \text{improvement} \\ \text{parameter} \end{array} \right] = \left[ \begin{array}{l} \text{passenger} \\ \text{flow on} \\ \text{guideway} \end{array} \right] \div \left[ \begin{array}{l} \text{fixed cost} \\ P_{ij} \text{ of} \\ \text{guideway} \end{array} \right] \quad (4-13)$$

The guideway with the smallest parameter is selected for deletion, subject to the condition that all stations remain connected. The reasoning is that a smaller flow on the guideway implies fewer passengers will have to alter their paths, and the result will be a lesser increase in passenger time costs  $d_{ij}$ ,  $d_w$ , and  $d_t$ . The fixed cost acts inversely and thus favors greater cost savings.

After the selected guideway is declared closed, the route algorithm is entered to obtain the best flow, routes, and frequencies. On the first iteration of the route algorithm the starting routes are adjusted to conform to the newly-closed guideway.

As usual, a decrease in the overall costs leads to repetition of the deletion process. If the deletion was unsuccessful, the selected guideway is opened again, and the previous routes and frequencies are reestablished.

In the same manner as the route algorithm, the guideway algorithm alternates between insertion and deletion until no further cost improvements can be made.

### Summary of Solution Procedure

The research problem attacked is to select fixed-cost guideways for inclusion in a transit network, determine vehicle routes and route service frequencies, and assign passengers in the form of multi-commodity flow to the different vehicle routes. The objective is to minimize the sum of guideway fixed costs, passenger travel and delay time costs, and vehicle operating costs.

Four separate but closely inter-related techniques are used to reach an approximate minimum cost solution:

- (1) guideway insertion-deletion algorithm
- (2) route insertion-deletion algorithm
- (3) penalty-cost multicommodity flow assignment routine
- (4) shortest path assignment routine.

The relationships among these routines is shown in Figure 14.

The shortest path assignment routine calculates the least cost path for each passenger commodity with respect to travel time, waiting time, and transfer time costs. The shortest paths are calculated without regard to arc capacity constraints. The penalty-cost multicommodity flow assignment routine tests for excessive flow on individual arcs and reassigns commodity movements to arcs that are not overloaded on the basis of the least cost increase between the shortest path and the second shortest path. All these operations are performed on the revised network, constructed from the set of vehicle routes.

Once a feasible flow assignment has been achieved, it is declared the best flow assignment for the set of routes. Arc improvement

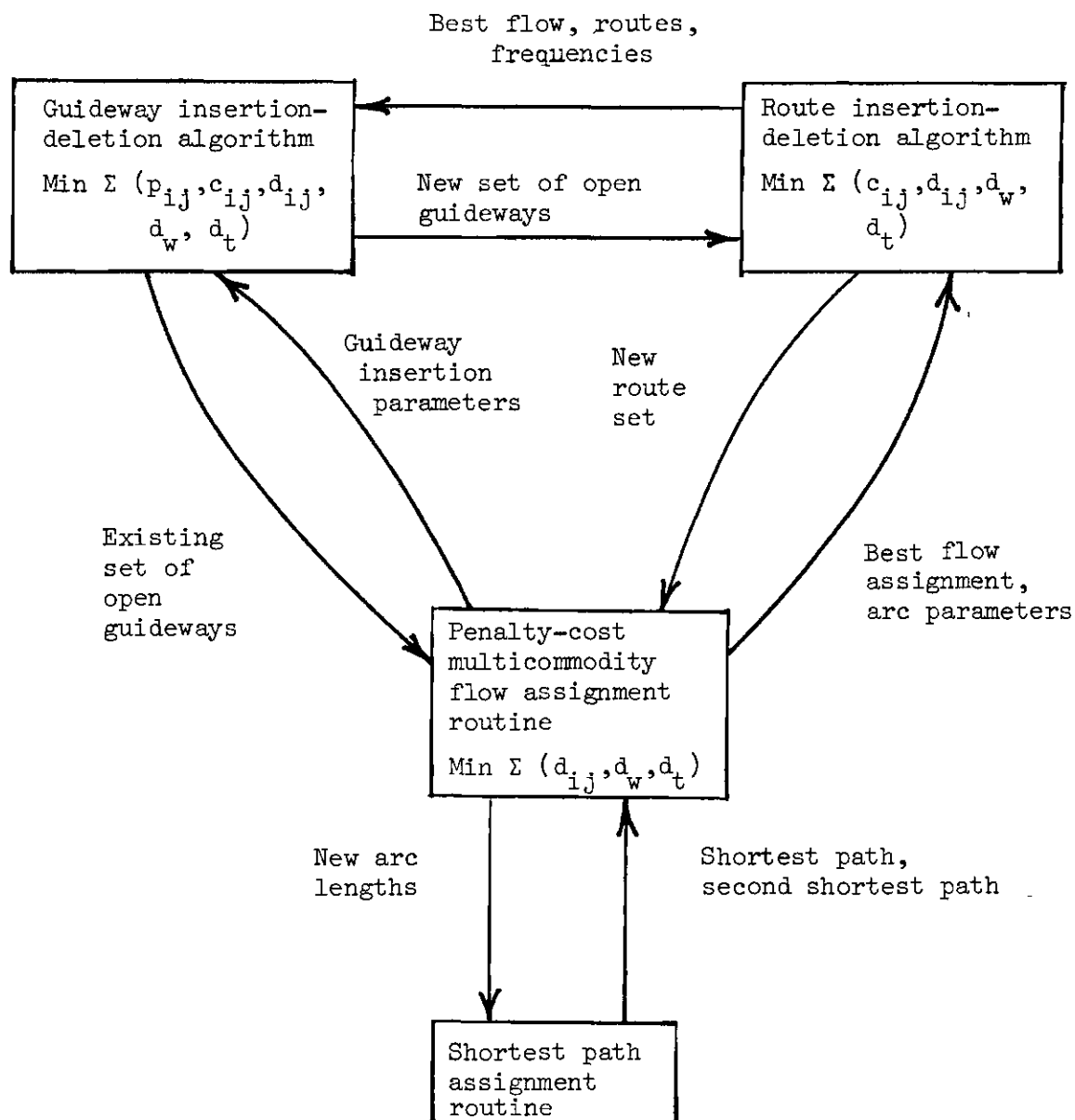


Figure 14. Relationships Among Routines.

parameters are then computed from the infeasibility parameters generated by the multicommodity flow routine. The route insertion-deletion routine checks the level of traffic on each route, and decides whether new routes are needed or whether routes can be dropped, using the information embodied in the arc improvement parameters.

When the route algorithm can make no more improvements in the sum of vehicle operating and passenger time costs, it sends the best set of routes to the guideway algorithm. The latter changes the set of open guideways in trying to minimize the grand total of costs.

The guideway algorithm also calls on the multicommodity flow routine, to obtain guideway insertion improvement parameters. Again, the operations of the multicommodity routine are performed on a revised network, constructed from the set of regular vehicle routes plus a zero-level route for each closed guideway.

Figure 15 shows a simplified flowchart of the solution procedure. A successful change in any phase of either guideway or route algorithm results in a repeat of that phase, indicated by a dashed line. A solid line indicates normal progress of the program or the exit from a phase after an unsuccessful change.

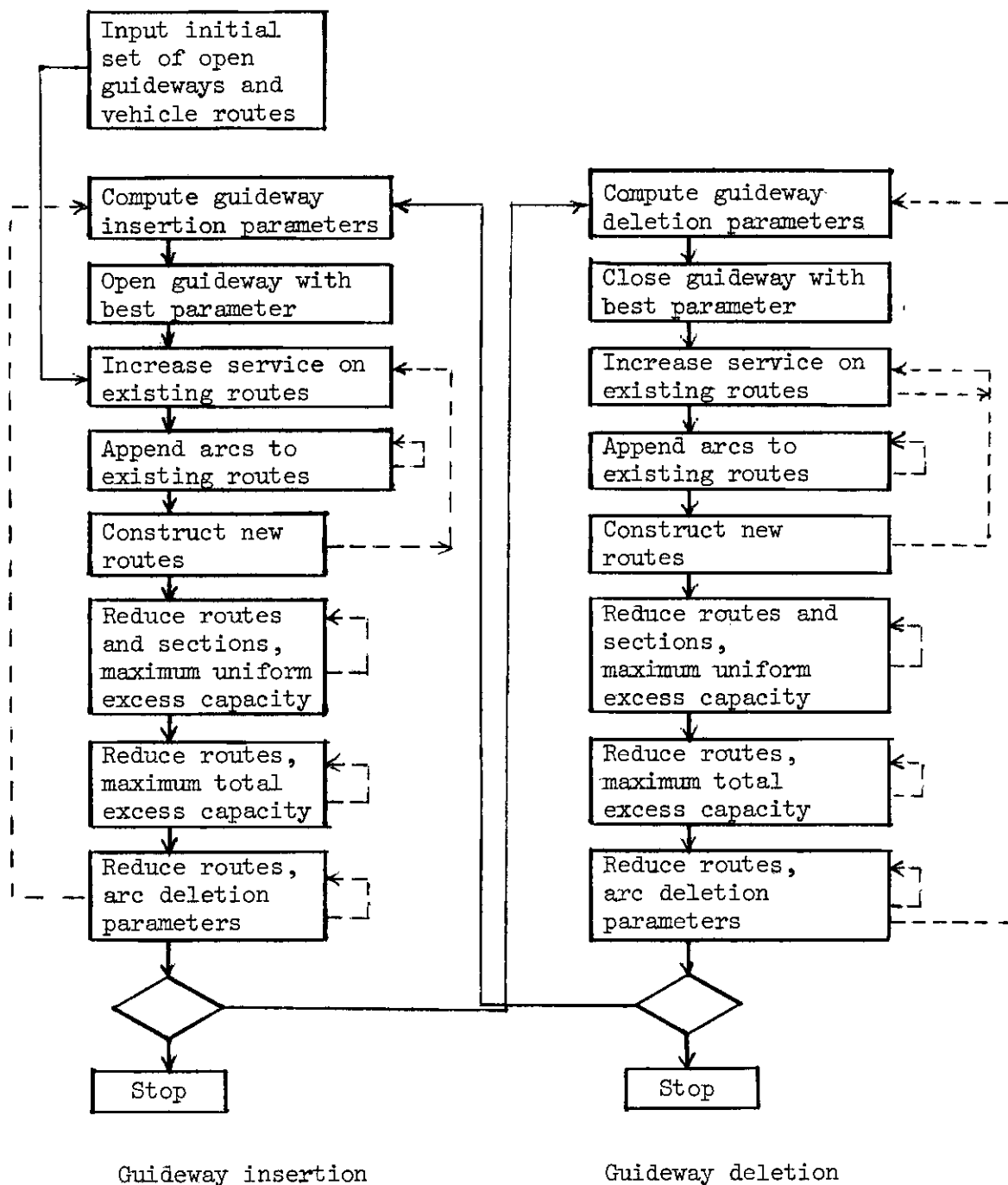


Figure 15. Flowchart of Solution Procedure.



## CHAPTER V

### COMPUTATIONAL RESULTS

A FORTRAN V computer program has been written for the solution procedure, and several small test problems have been solved on the Univac 1108. The program occupies a total of approximately 29,000 words in core, 15,000 for instructions and 14,000 for data. Virtually all data is maintained in integer form.

This chapter presents selective examples illustrating the major routines and summary data on all the test problems solved. Test problems are designated by symbols of the type C, Cl, and Cla, where

- C    designates the given station locations and travel demands, and the set of all possible guideways,
- l    represents a set of open guideways for problem C,
- a    represents a set of vehicle routes for problem Cl.

In this framework, then, the guideway algorithm works with problem C and passes on to the route algorithm problems Cl, C2, C3, etc. The route algorithm works on a problem of type Cl and passes on to the multicommodity assignment routine problems Cla, Clb, Clc, etc.

#### Construction of the Revised Network

The revised network is constructed just before executing the multicommodity routine. The problems handled are therefore of the type

Cla.

Figure 16 shows the 12 station locations and 21 possible guideway pairs in problem C, and the 15 open guideway pairs in problem Cl. The travel demands in problem C are given in Table 1. The vehicle routes and service frequencies in problem Cla are depicted graphically in Figure 17.

The construction of the revised network can be imagined as a linking together of the routes in Figure 17. The resulting network of 51 nodes and 115 arcs is shown in Figure 18. In addition to the six regular routes given there were established four zero-level routes for the four uncovered, open guideways: (4,5), (3,6), (6,9), and (10,11).

Passenger travel time and waiting time costs are both assumed to be \$0.03/minute, while a transfer time cost equivalent to 15 minutes is used. The operating time period is taken to be one hour.

#### Penalty-Cost Multicommodity Flow Assignment Routine

The first step of the multicommodity routine is to assign each commodity to its shortest path. This results in nine infeasible arcs and 16 commodities flowing over them at the 0<sup>th</sup> iteration, shown in Table 2. The computation of  $\epsilon_{ps} = 12$ , rounded up to  $\epsilon_{ps} = 13$ , results in commodity 27 being reassigned to its second shortest path. Commodity 15 is dominated by the criterion in (4-6), and it is flagged by the 99 in the  $n_2$  column. The infeasibility parameter of each arc in Table 2 is incremented by  $-(13 \times \text{excess flow})$ .

At iteration 1, shown in Table 3, there are still nine infeasible

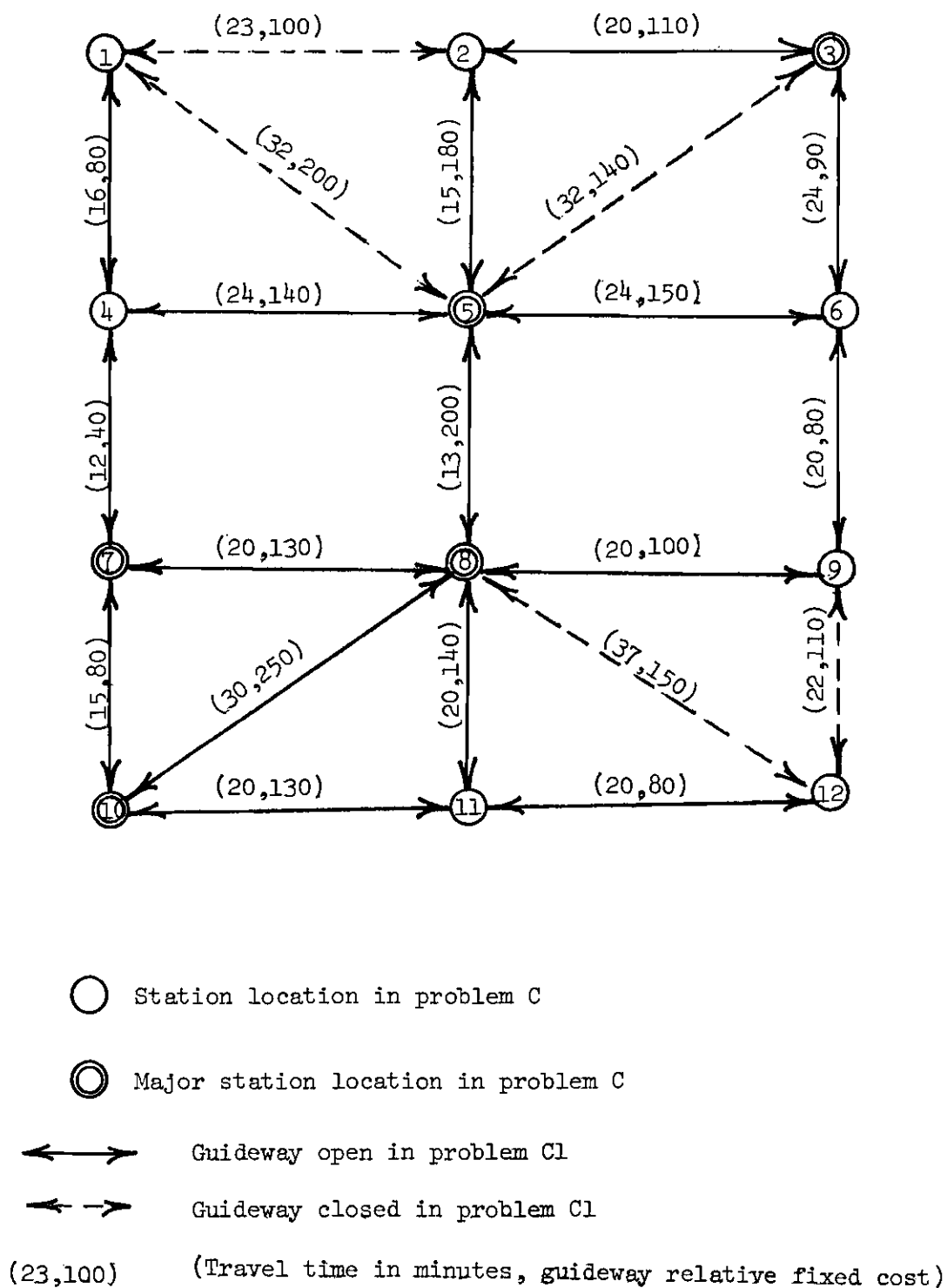


Figure 16. Station Locations and Possible Guideways in Problem C. Open Guideways in Problem C1.

Table 1. Travel Demands in  
Problem C.

| Commodity Number | Origin Station | Destination Station | Travel Demand/Hr |
|------------------|----------------|---------------------|------------------|
| 1                | 1              | 4                   | 500              |
| 2                | 1              | 5                   | 200              |
| 3                | 1              | 7                   | 200              |
| 4                | 1              | 8                   | 400              |
| 5                | 2              | 3                   | 200              |
| 6                | 2              | 5                   | 300              |
| 7                | 2              | 6                   | 100              |
| 8                | 2              | 8                   | 600              |
| 9                | 3              | 2                   | 300              |
| 10               | 3              | 5                   | 400              |
| 11               | 3              | 6                   | 400              |
| 12               | 3              | 8                   | 400              |
| 13               | 3              | 9                   | 100              |
| 14               | 4              | 1                   | 200              |
| 15               | 4              | 5                   | 200              |
| 16               | 4              | 7                   | 100              |
| 17               | 4              | 8                   | 500              |
| 18               | 5              | 2                   | 100              |
| 19               | 5              | 4                   | 100              |
| 20               | 5              | 6                   | 100              |
| 21               | 5              | 8                   | 500              |
| 22               | 6              | 3                   | 100              |
| 23               | 6              | 5                   | 300              |
| 24               | 6              | 8                   | 700              |
| 25               | 6              | 9                   | 200              |
| 26               | 7              | 4                   | 100              |
| 27               | 7              | 5                   | 100              |
| 28               | 7              | 8                   | 800              |
| 29               | 7              | 9                   | 200              |
| 30               | 7              | 11                  | 100              |
| 31               | 8              | 5                   | 100              |
| 32               | 8              | 7                   | 200              |
| 33               | 8              | 9                   | 100              |
| 34               | 8              | 11                  | 300              |
| 35               | 9              | 6                   | 100              |
| 36               | 9              | 8                   | 700              |
| 37               | 9              | 11                  | 300              |
| 38               | 10             | 5                   | 100              |
| 39               | 10             | 8                   | 700              |
| 40               | 10             | 11                  | 600              |

Table 1. Continued.

| Commodity Number | Origin Station | Destination Station | Travel Demand/Hr |
|------------------|----------------|---------------------|------------------|
| 41               | 11             | 8                   | 500              |
| 42               | 11             | 10                  | 300              |
| 43               | 11             | 12                  | 100              |
| 44               | 12             | 8                   | 500              |
| 45               | 12             | 10                  | 100              |
| 46               | 12             | 11                  | 200              |

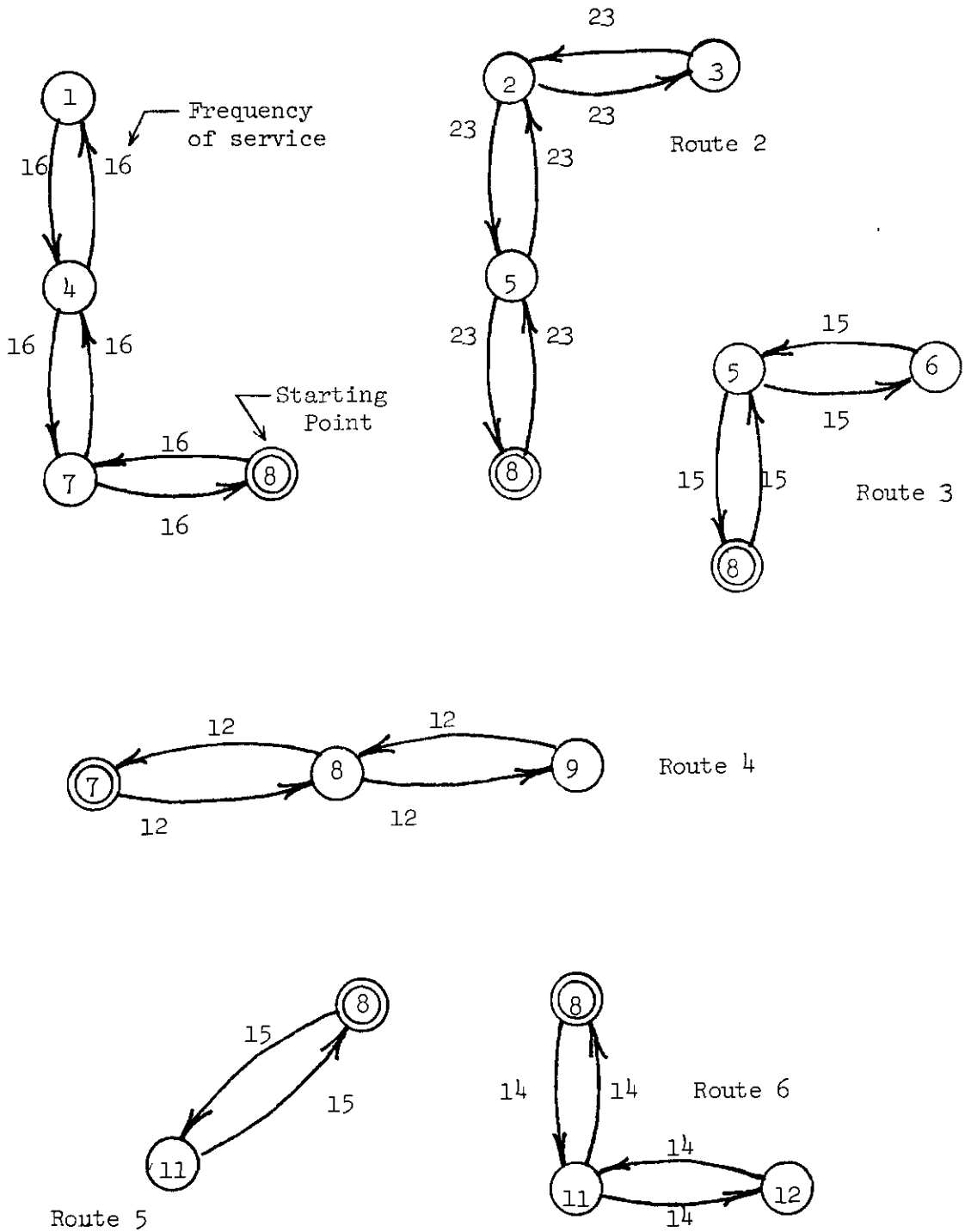


Figure 17. Vehicle Routes and Frequencies in Problem 1a.

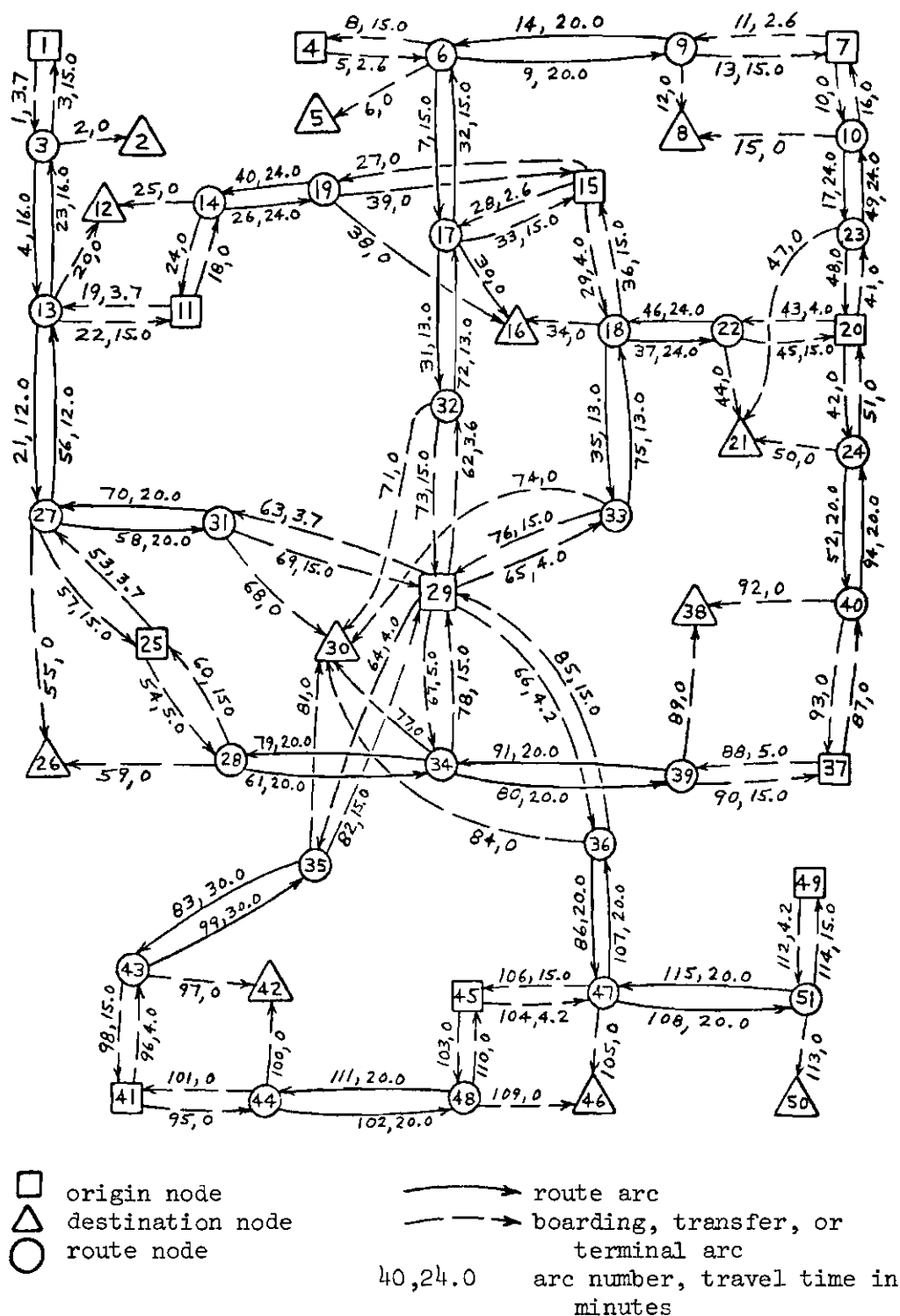


Figure 18. Revised Network in Problem 1a.

Table 2. Problem C1a, Iteration 0.  
Infeasible Arcs, Commodity Set K1.

| Arc |  | Flow | Excess |
|-----|--|------|--------|
| 17  |  | 500  | 500    |
| 26  |  | 400  | 400    |
| 40  |  | 100  | 100    |
| 49  |  | 100  | 100    |
| 52  |  | 300  | 300    |
| 58  |  | 1900 | 300    |
| 94  |  | 100  | 100    |
| 102 |  | 600  | 600    |
| 111 |  | 400  | 400    |

| k    | $n_1$ | $n_2$ | $p_1$ | $p_2$ | $u_k$ |
|------|-------|-------|-------|-------|-------|
| 2    | 1     | 0     | 587   | 1022  | 435   |
| 4    | 1     | 0     | 517   | 716   | 199   |
| 11   | 1     | 0     | 240   | 805   | 565   |
| 13   | 2     | 0     | 440   | 905   | 233   |
| 15   | 1     | 99    | 240   | 663   | -     |
| 17   | 1     | 0     | 357   | 556   | 199   |
| 19   | 1     | 0     | 240   | 663   | 423   |
| 22   | 1     | 0     | 240   | 806   | 564   |
| 25   | 1     | 0     | 200   | 809   | 609   |
| → 27 | 1     | 0     | 543   | 555   | 12    |
| 28   | 1     | 0     | 237   | 249   | 12    |
| 30   | 1     | 0     | 629   | 641   | 12    |
| 35   | 1     | 0     | 200   | 809   | 609   |
| 40   | 1     | 0     | 200   | 732   | 532   |
| 42   | 1     | 0     | 200   | 731   | 531   |
| 45   | 1     | 0     | 592   | 931   | 339   |

|            |             |
|------------|-------------|
| ICAND = 27 | INCMNT = 13 |
|------------|-------------|

Note: Values for  $p_1$ ,  $p_2$ , and  $u_k$  are in tenths of a minute.



Table 3. Problem Cla, Iteration 1.  
Infeasible Arcs, Commodity Set K1.

| Arc |  | Flow | Excess |  |  |
|-----|--|------|--------|--|--|
| 17  |  | 500  | 500    |  |  |
| 26  |  | 400  | 400    |  |  |
| 40  |  | 100  | 100    |  |  |
| 49  |  | 100  | 100    |  |  |
| 52  |  | 300  | 300    |  |  |
| 58  |  | 1800 | 200    |  |  |
| 94  |  | 100  | 100    |  |  |
| 102 |  | 600  | 600    |  |  |
| 111 |  | 400  | 400    |  |  |

| k          | $n_1$ | $n_2$ | $P_1$      | $P_2$ | $u_k$ |
|------------|-------|-------|------------|-------|-------|
| 2          | 1     | 0     | 600        | 1022  | 422   |
| 4          | 1     | 0     | 530        | 716   | 186   |
| 11         | 1     | 0     | 253        | 805   | 552   |
| 13         | 2     | 0     | 466        | 905   | 219   |
| 15         | 1     | 99    | 253        | 676   | -     |
| 17         | 1     | 0     | 370        | 556   | 186   |
| 19         | 1     | 0     | 253        | 663   | 410   |
| 22         | 1     | 0     | 253        | 806   | 553   |
| 25         | 1     | 0     | 213        | 809   | 596   |
| → 28       | 1     | 0     | 250        | 249   | -1    |
| 30         | 1     | 0     | 642        | 641   | -1    |
| 35         | 1     | 0     | 213        | 809   | 596   |
| 40         | 1     | 0     | 213        | 732   | 519   |
| 42         | 1     | 0     | 213        | 731   | 518   |
| 45         | 1     | 0     | 605        | 931   | 326   |
| ICAND = 28 |       |       | INCMNT = 1 |       |       |

arcs, but the excess flow on arc 58 has been reduced from 300 to 200. For the 15 commodities in the set K1 it is necessary to calculate only one second shortest path, for the dominated commodity 15. Commodity 28 is the candidate for reassignment with a value of  $\text{eps} = 1$ , rounded up from  $\text{eps} = 0$ . Actually, commodity 28 was tied with 27 for reassignment at iteration 0.

Table 4 shows the situation at iteration 2, with eight infeasible arcs and 11 commodities in K1. The reassignment of 28 also freed commodities 4, 17, and 30.

Problem Cla required 13 iterations before achieving feasibility, and thus the best flow assignment. The infeasibility parameters for arcs and guideways are presented in Table 5.

For the typically small problems tested the multicommodity routine usually required slightly fewer iterations than the number of commodities in the set K1 at the first iteration. Table 13 in a later section summarizes the computational experience with this routine.

At times the routine failed to obtain a feasible solution when one existed. Such a situation can occur when it is not possible to reassign a commodity from its shortest, infeasible path to a feasible, second shortest path because some other commodity on the latter path needs to be reassigned first. If this other commodity is flowing over a feasible path, it is not considered for reassignment and hence it blocks the reassignment of the first, infeasible commodity.

Table 4. Problem Cla, Iteration 2.  
Infeasible Arcs, Commodity Set K1.

| Arc |  | Flow | Excess |
|-----|--|------|--------|
| 17  |  | 500  | 500    |
| 26  |  | 400  | 400    |
| 40  |  | 100  | 100    |
| 49  |  | 100  | 100    |
| 52  |  | 300  | 300    |
| 94  |  | 100  | 100    |
| 102 |  | 600  | 600    |
| 111 |  | 400  | 400    |

| k          | $n_1$ | $n_2$ | $p_1$        | $p_2$ | $u_k$ |
|------------|-------|-------|--------------|-------|-------|
| 2          | 1     | 0     | 601          | 837   | 236   |
| 11         | 1     | 0     | 254          | 805   | 551   |
| → 13       | 2     | 0     | 468          | 905   | 218   |
| 15         | 1     | 0     | 254          | 677   | 423   |
| 19         | 1     | 0     | 254          | 663   | 409   |
| 22         | 1     | 0     | 254          | 806   | 552   |
| 25         | 1     | 0     | 214          | 809   | 595   |
| 35         | 1     | 0     | 214          | 809   | 595   |
| 40         | 1     | 0     | 214          | 732   | 518   |
| 42         | 1     | 0     | 214          | 731   | 517   |
| 45         | 1     | 0     | 606          | 931   | 325   |
| ICAND = 13 |       |       | INCMNT = 219 |       |       |

Table 5. Problem Cla. Improvement  
Parameters at Final Iteration.

| Arc Number | Parameter | Guideway          |                        | Parameter |
|------------|-----------|-------------------|------------------------|-----------|
|            |           | Origin<br>Station | Destination<br>Station |           |
| 17         | -749      | 3                 | 6                      | -749      |
| 26         | -413      | 4                 | 5                      | -413      |
| 40         | -127      | 5                 | 4                      | -127      |
| 49         | -169      | 6                 | 3                      | -169      |
| 52         | -436      | 6                 | 9                      | -436      |
| 58         | - 13      | 7                 | 8                      | - 13      |
| 94         | -182      | 9                 | 6                      | -182      |
| 102        | -958      | 10                | 11                     | -958      |
| 111        | -583      | 11                | 10                     | -583      |

### Route Insertion-Deletion Algorithm

This algorithm is illustrated using the same test problem C1, shown in Figure 16 and Table 1. The initial set of vehicle routes are the same as presented in Figure 17. The first subproblem passed on to the multicommodity routine is therefore problem C1a. A vehicle operating cost of \$1.00 per minute is used.

An unusual feature of the improvement parameters for this problem can be observed in Table 5: only one guideway listed is covered by a regular route. The algorithm thus bypasses the first phase, increasing service on routes, and proceeds to append guideway arcs to routes.

The following changes are made at iterations 1, 2, and 3:

- 1: append (10,11) to route 5
- 2: append (3,6) to route 3
- 3: append (6,9) to route 4.

These changes are shown in Figure 19 and also listed in Table 6. The guideway pair (4,5) cannot be appended to any route and is placed in the hold list. Just before entering the route construction phase the guideway improvement parameters are

|       |       |
|-------|-------|
| (4,5) | -434  |
| (5,4) | -127  |
| (7,8) | - 28. |

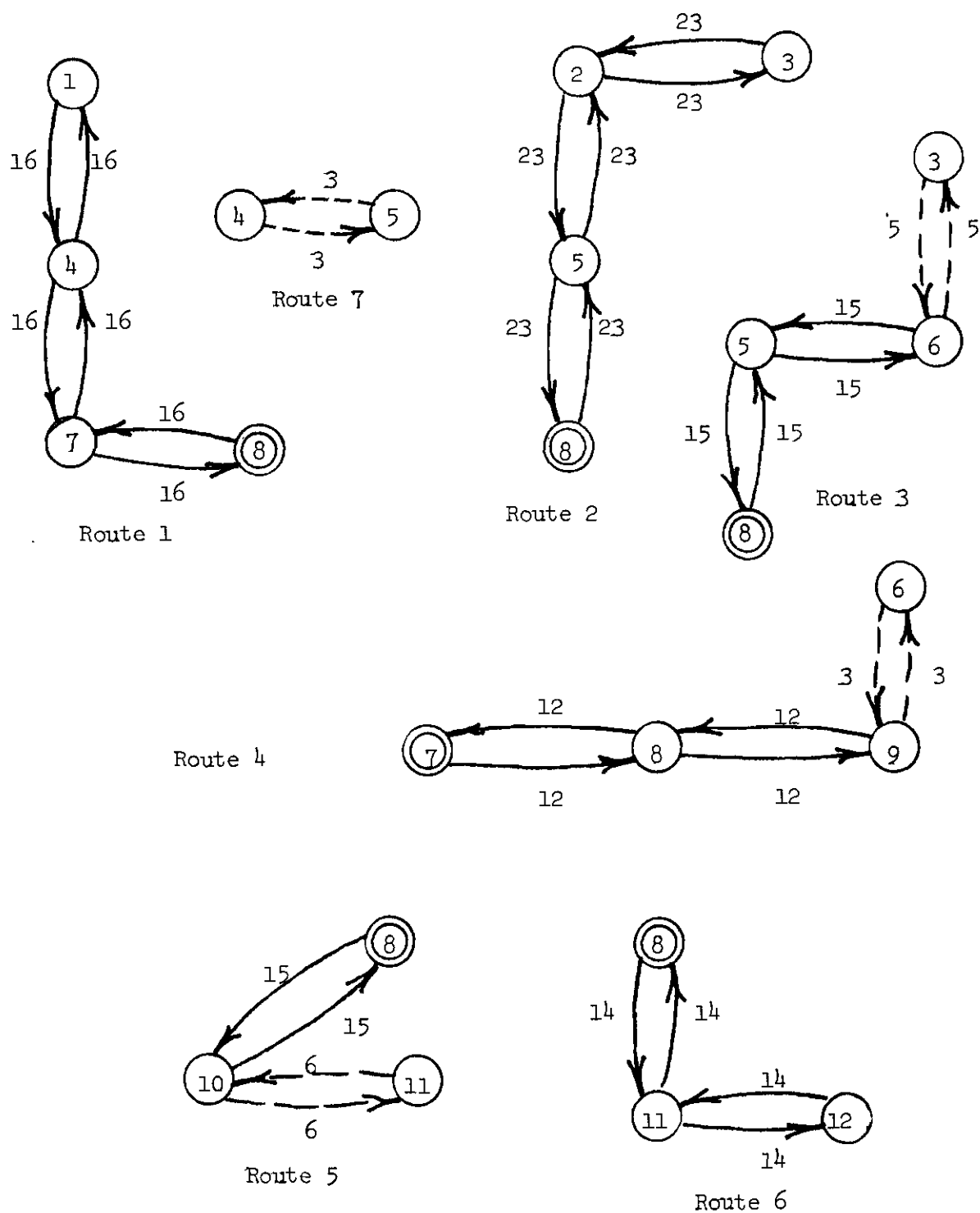


Figure 19. Problem C1, Changes During Route-Insertion.  
(Changes are indicated by dashed arcs)

The out-tree built during the guideway construction phase is shown in Figure 20, along with the various reversal run and loop routes examined. The route selected for insertion at iteration 4 is 5-4-5, also shown in Figure 19.

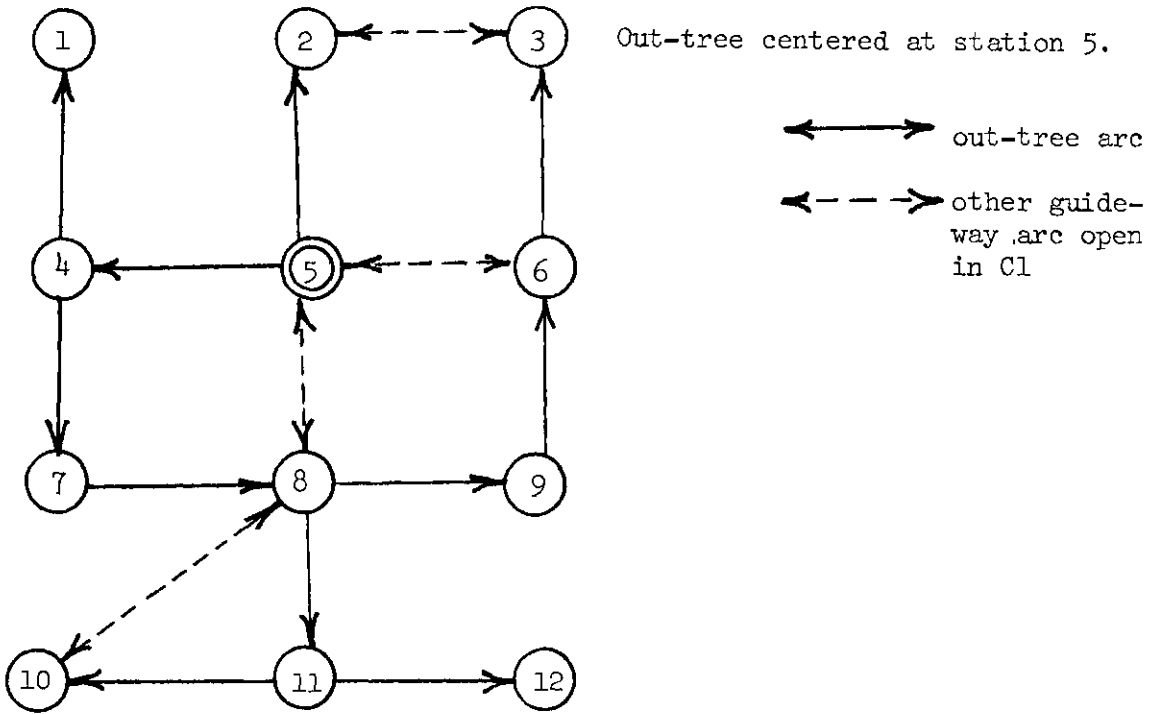
The algorithm goes through an unsuccessful operation, constructing route 5-8-7-8-5, before entering route deletion.

In the deletion part of the algorithm the first phase does most of the work in Problem C1. The criterion of maximum uniform excess capacity over routes and sections (4-9) is used to select routes for reduction of service.

The first such iteration reduces service on the last branch of route 2 from 23 vehicles per hour to 11 per hour. This is shown in Table 6 at iteration 7. Iteration 8 reduces service on the first branch of route 5. At a later iteration, number 10, the service on the first two branches of route 2 is reduced.

After several more iterations the deletion part of the algorithm enters phase 2. Here route 1 is selected, on the basis of maximum total excess capacity (4-10). The service level is reduced by one vehicle per hour, as shown in Table 6 at iteration 14.

Here the multicommodity assignment routine detects an infeasible network, and the route algorithm rejects the change. Since route 1 is a reversal run, all the information is kept and the algorithm switches to phase 3. The following deletion parameters are then generated for route 1:



Routes examined:

5-2-5  
 5-4-5  
 5-4-7-4-5  
 5-4-7-8-5  
 5-4-7-8-7-4-5  
 5-4-7-8-5  
 5-4-7-8-11-8-4-5  
 5-4-7-8-11-10-11-8-7-4-5  
 5-4-7-8-11-12-11-8-7-4-5  
 5-4-7-8-9-8-7-4-5  
 5-4-7-8-9-6-5  
 5-4-1-4-5  
 5-4-7-8-9-6-9-8-7-4-5  
 5-4-7-8-9-6-5  
 5-4-7-8-9-6-3-6-9-8-7-4-5  
 5-4-7-8-9-6-3-2-5

Figure 20. Problem C1, Results of Route Construction Phase, at Iteration 4.



|           |       |       |       |       |       |       |
|-----------|-------|-------|-------|-------|-------|-------|
| Arc       | (8,7) | (7,4) | (4,1) | (1,4) | (4,7) | (7,8) |
| Parameter | -19   | -11   | -15   | +1021 | -11   | +11   |

The high value for guideway (1,4) stems from the infeasibility of the network. The section of route 1 with the best route deletion parameter consists of arcs (8,7), (7,4), (4,7), and (7,8). Accordingly, the service on those branches is reduced by one unit from the previous service level of 16. Service on the last branch, arcs (4,1) and (1,4), is kept at the previous level of 13. This operation is performed at iteration 15.

Iterations 16 and 17 repeat the steps done at 14 and 15. At iteration 18 route 2 is selected for reduction, with infeasibility again occurring. The arc deletion parameters are:

|           |       |       |       |       |       |       |
|-----------|-------|-------|-------|-------|-------|-------|
| Arc       | (8,5) | (5,2) | (2,3) | (3,2) | (2,5) | (5,8) |
| Parameter | -12   | -14   | -19   | +1017 | +1022 | -12   |

The only section of route 2 with a negative route deletion parameter is the first branch, containing arcs (8,5) and (5,8). Service on that branch cannot be reduced because it would result in less service on (8,5) than on (5,2), violating the multi-zone route form. Thus, the route algorithm exits phase three and, at the same time, the deletion part.

Since successful changes were made in the deletion part, the insertion part is entered again. After a futile effort to construct a

route the algorithm stops. The final set of routes in problem C1 is listed at the end of Table 6.

### Guideway Insertion-Deletion Algorithm

#### Test Problem E

The operation of the guideway algorithm is illustrated with three examples. The first of these, test problem E, is a small problem designed to see if the algorithm can achieve a fairly obvious optimal solution. Problem E is shown in Figure 21 and Table 7. The guideway travel times and fixed costs and the travel demands have been structured so that the optimal guideway set consists of the tree within the perimeter of the network.

The progress of the guideway algorithm is detailed in Table 8 and illustrated graphically in Figure 22. At the end of the 0<sup>th</sup> iteration the improvement parameters for closed guideways are

|           |         |
|-----------|---------|
| arc (2,3) | -2,356  |
| (3,4)     | -10,783 |
| (4,5)     | -3,246. |

The two-way guideway connecting stations 3 and 4 has the best value and is thus selected for insertion. The route algorithm appends (3,4) to one of the initial routes and effects a substantial saving in passenger time and vehicle operating costs:

(6,4) (4,6) becomes (6,4) (4,3) (3,4) (4,6).

Table 6. Problem C1, Results of Route Algorithm.

| Iter. No. | Change  | Pass. Time Costs | Vehicle Optg. Costs | Total  | Impt. |
|-----------|---|------------------|---------------------|--------|-------|
| 0         | Initial solution, starting routes are:  |                  |                     |        |       |
|           | 1 (8,7) (7,4) (4,1) (1,4) (4,7) (7,8)<br>16 16 16 16 16 16  |                  |                     |        |       |
|           | 2 (8,5) (5,2) (2,3) (3,2) (2,5) (5,8)<br>23 23 23 23 23 23  |                  |                     |        |       |
|           | 3 (8,5) (5,6) (6,5) (5,8)<br>15 15 15 15  |                  |                     |        |       |
|           | 4 (7,8) (8,9) (9,8) (8,7)<br>12 12 12 12  |                  |                     |        |       |
|           | 5 (8,10) (10,8)<br>15 15  |                  |                     |        |       |
|           | 6 (8,11) (11,12) (12,11) (11,8)<br>14 14 14 14  | 16,136           | 7,834               | 23,970 | -     |
| 1         | Append (10,11) to route 5<br>(8,10) (10,11) (11,10) (10,8)<br>15 6 6 15                                   | 14,863           | 8,074               | 22,937 | 1,034 |
| 2         | Append (3,6) to route 3<br>(8,5) (5,6) (6,3) (3,6) (6,5) (5,8)<br>15 15 2 2 15 15                         | 14,224           | 8,314               | 22,538 | 398   |
| 3         | Append (6,9) to route 4<br>(7,8) (8,9) (9,6) (6,9) (9,8) (8,7)<br>12 12 3 3 12 12                         | 13,872           | 8,434               | 22,306 | 233   |
| 4         | Construct new route 7<br>(5,4) (4,5)<br>3 3   | 13,626           | 8,578               | 22,204 | 102   |
| 5         | Construct new route 8<br>(5,8) (8,7) (7,8) (8,5)<br>1 1 1 1   | 13,728           | 8,644               | 22,372 | -168  |
| 6         | Reestablish previous routes   | 13,626           | 8,578               | 22,204 | -     |
| 7         | Reduce service on route 2 section by 12 units<br>(8,5) (5,2) (2,3) (3,2) (2,5) (5,8)<br>23 23 11 11 23 23 | 13,753           | 8,098               | 21,851 | 353   |
| 8         | Reduce service on route 5 section by 7 units<br>(8,10) (10,11) (11,10) (10,8)<br>8 6 6 8                  | 13,870           | 7,678               | 21,548 | 302   |

Table 6. Continued.

| Iter.<br>No. | Change  | Pass. Time<br>Costs | Vehicle Optg.<br>Costs | Total  | Impvt. |
|--------------|---|---------------------|------------------------|--------|--------|
| 9            | Reduce service on route 6 section by<br>6 units<br>(8,11) (11,12) (12,11) (11,8)<br><u>14</u> <u>8</u> <u>8</u> <u>14</u>   | 13,996              | 7,438                  | 21,434 | 114    |
| 10           | Reduce service on route 2 section by<br>5 units<br>(8,5) (5,2) (2,3) (3,2) (2,5) (5,8)<br><u>18</u> <u>18</u> 11    11 <u>18</u> <u>18</u>  | 14,045              | 7,158                  | 21,203 | 232    |
| 11           | Reduce service on route 3 section by<br>5 units<br>(8,5) (5,6) (6,3) (3,6) (6,5) (5,8)<br><u>10</u> <u>10</u> 5    5 <u>10</u> <u>10</u>  | 14,123              | 6,788                  | 20,911 | 292    |
| 12           | Reduce service on route 6 section by<br>4 units<br>(8,11) (11,12) (12,11) (11,8)<br><u>10</u> 8    8 <u>10</u>  | 14,170              | 6,628                  | 20,798 | 113    |
| 13           | Reduce service on route 1 section by<br>3 units<br>(8,7) (7,4) (4,1) (1,4) (4,7) (7,8)<br>16    16 <u>13</u> <u>13</u> 16    16   | 14,212              | 6,532                  | 20,744 | 54     |
| 14           | Reduce service on route 1 by one unit,<br>compute deletion parameters<br>(8,7) (7,4) (4,1) (1,4) (4,7) (7,8)<br>15    15    12    12    15    15<br>-19   -11   -15 +1021   -11   +11 | Infeasible network  |                        |        |        |
| 15           | Reduce service on route 1 section by<br>1 unit<br>(8,7) (7,4) (4,1) (1,4) (4,7) (7,8)<br><u>15</u> <u>15</u> 13    13 <u>15</u> <u>15</u>   | 14,219              | 6,468                  | 20,687 | 57     |
| 16           | Reduce service on route 1 by one unit,<br>compute deletion parameters<br>(8,7) (7,4) (4,1) (1,4) (4,7) (7,8)<br>14    14    12    12    14    14<br>-19   -11   -15 +1021   -11   +15 | Infeasible network  |                        |        |        |
| 17           | Reduce service on route 1 section by<br>1 unit<br>(8,7) (7,4) (4,1) (1,4) (4,7) (7,8)<br><u>14</u> <u>14</u> 13    13 <u>14</u> <u>14</u>   | 14,226              | 6,404                  | 20,630 | 57     |
| 18           | Reduce service on route 2 by one unit,<br>compute deletion parameters<br>(8,5) (5,2) (2,3) (3,2) (2,5) (5,8)<br>17    17    10    10    17    17<br>-12   -14   -19 +1017 +1022   -12 | Infeasible network  |                        |        |        |

Table 6. Continued.

| Iter.<br>No. | Change  | Pass. Time<br>Costs | Vehicle Optg.<br>Costs | Imovt. |      |
|--------------|---|---------------------|------------------------|--------|------|
| 19           | Reestablish previous routes                                 | 14,226              | 6,404                  | 20,630 | -    |
| 20           | Construct new route 8<br>(5,8) (8,7) (7,8) (8,5)<br>1 1 1 1 | 14,325              | 6,470                  | 20,795 | -164 |
| 21           | Reestablish previous routes<br>Stop, Final routes are       | 14,226              | 6,404                  | 20,630 | -    |
| 1            | (8,7) (7,4) (4,1) (1,4) (4,7) (7,8)<br>14 14 13 13 14 14    |                     |                        |        |      |
| 2            | (8,5) (5,2) (2,3) (3,2) (2,5) (5,8)<br>18 18 11 11 18 18    |                     |                        |        |      |
| 3            | (8,5) (5,6) (6,3) (3,6) (6,5) (5,8)<br>10 10 5 5 10 10      |                     |                        |        |      |
| 4            | (7,8) (8,9) (9,6) (6,9) (9,8) (8,7)<br>12 12 3 3 12 12      |                     |                        |        |      |
| 5            | (8,10) (10,11) (11,10) (10,8)<br>8 6 6 8                    |                     |                        |        |      |
| 6            | (8,11) (11,12) (12,11) (11,8)<br>10 8 8 10                  |                     |                        |        |      |
| 7            | (5,4) (4,5)<br>3 3  |                     |                        |        |      |

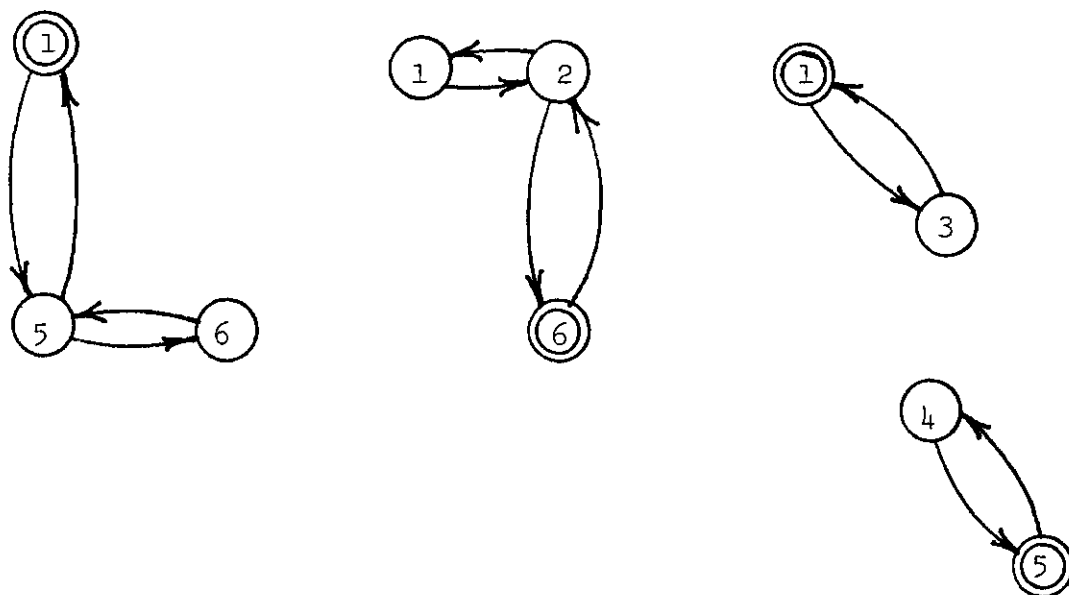
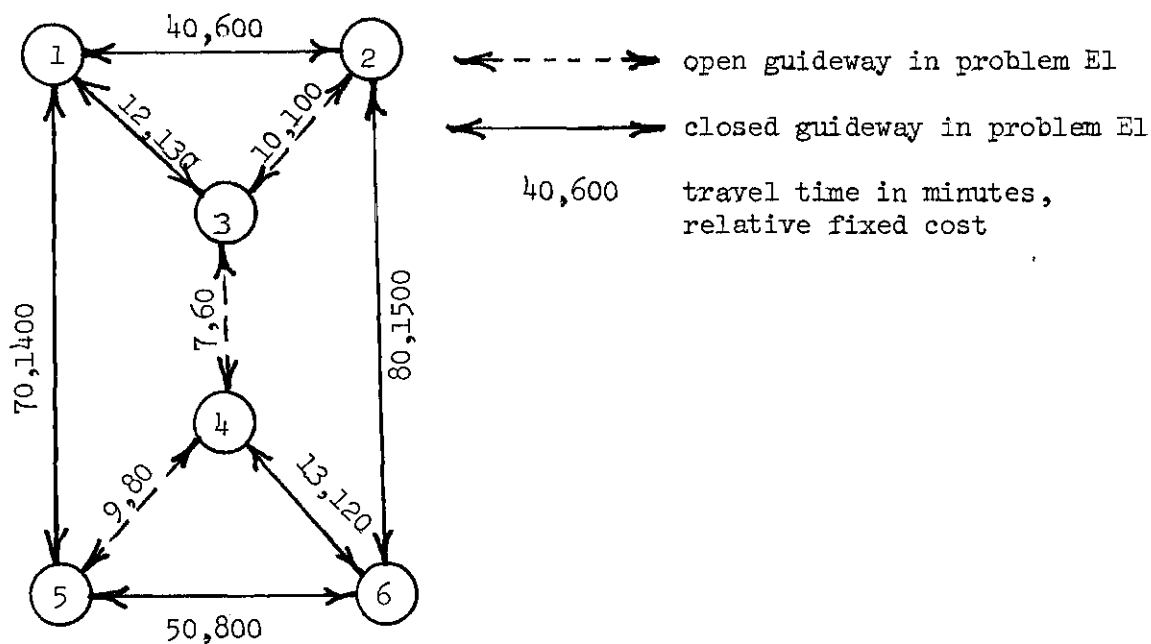


Figure 21. Problem E1, Open Guideways and Starting Vehicle Routes.

Table 7. Problem E, Travel Demands.

| Commodity<br>Number | Origin<br>Station | Destination<br>Station | Travel<br>Demand |
|---------------------|-------------------|------------------------|------------------|
| 1                   | 1                 | 2                      | 100              |
| 2                   | 1                 | 3                      | 400              |
| 3                   | 1                 | 4                      | 300              |
| 4                   | 1                 | 5                      | 100              |
| 5                   | 1                 | 6                      | 100              |
| 6                   | 2                 | 1                      | 100              |
| 7                   | 2                 | 3                      | 400              |
| 8                   | 2                 | 4                      | 200              |
| 9                   | 2                 | 5                      | 100              |
| 10                  | 2                 | 6                      | 100              |
| 11                  | 3                 | 1                      | 200              |
| 12                  | 3                 | 2                      | 200              |
| 13                  | 3                 | 4                      | 400              |
| 14                  | 3                 | 5                      | 100              |
| 15                  | 3                 | 6                      | 100              |
| 16                  | 4                 | 1                      | 100              |
| 17                  | 4                 | 2                      | 100              |
| 18                  | 4                 | 3                      | 400              |
| 19                  | 4                 | 5                      | 200              |
| 20                  | 4                 | 6                      | 200              |
| 21                  | 5                 | 1                      | 100              |
| 22                  | 5                 | 2                      | 100              |
| 23                  | 5                 | 3                      | 300              |
| 24                  | 5                 | 4                      | 400              |
| 25                  | 5                 | 6                      | 100              |
| 26                  | 5                 | 1                      | 100              |
| 27                  | 6                 | 2                      | 100              |
| 28                  | 6                 | 3                      | 300              |
| 29                  | 6                 | 4                      | 400              |
| 30                  | 6                 | 5                      | 100              |

Table 8. Guideway Algorithm, Results of  
Problem E, Starting with E1.

| Iter. | Change                       | Pass. Time<br>Costs | Vehicle Optg.<br>Costs | Guideway<br>Fixed Costs | Total  | Impvt. |
|-------|------------------------------|---------------------|------------------------|-------------------------|--------|--------|
| 0     | Start                        | 17,735              | 5,434                  | 4,550                   | 27,719 | -      |
| 1     | Insert (3,4)                 | 10,174              | 4,072                  | 4,610                   | 18,856 | 8,863  |
| 2     | Insert (4,5)                 | 10,174              | 4,072                  | 4,690                   | 18,936 | -80    |
| 3     | Delete (2,6)                 | 9,969               | 2,976                  | 3,110                   | 16,055 | 2,801  |
| 4     | Delete (1,5)                 | 10,052              | 2,966                  | 1,710                   | 14,728 | 1,327  |
| 5     | Insert (4,5)                 | 6,172               | 2,606                  | 1,790                   | 10,568 | 4,160  |
| 6     | Insert (2,3)                 | 4,549               | 2,354                  | 1,890                   | 8,793  | 1,775  |
| 7     | Insert (1,5)                 | 4,549               | 2,354                  | 3,290                   | 10,193 | -1,400 |
| 8     | Delete (5,6)                 | 4,444               | 2,098                  | 1,090                   | 7,632  | 1,161  |
| 9     | Delete (1,2)                 | 4,392               | 1,394                  | 490                     | 6,276  | 1,356  |
|       | Stop,<br>Final Total costs = | 6,276               |                        |                         |        |        |



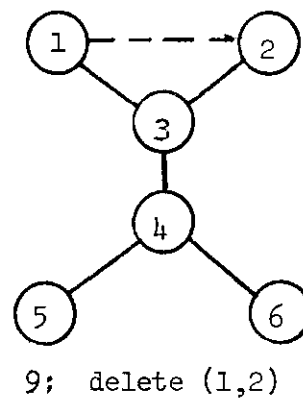
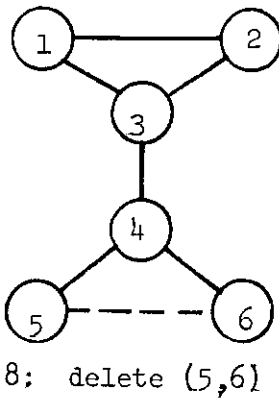
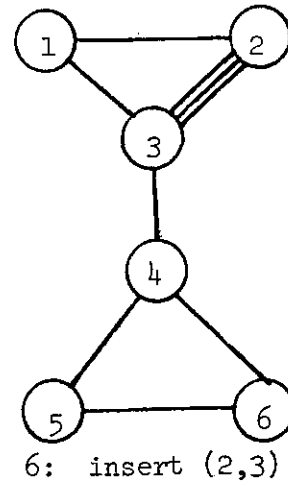
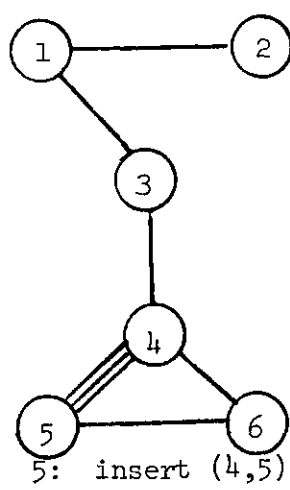
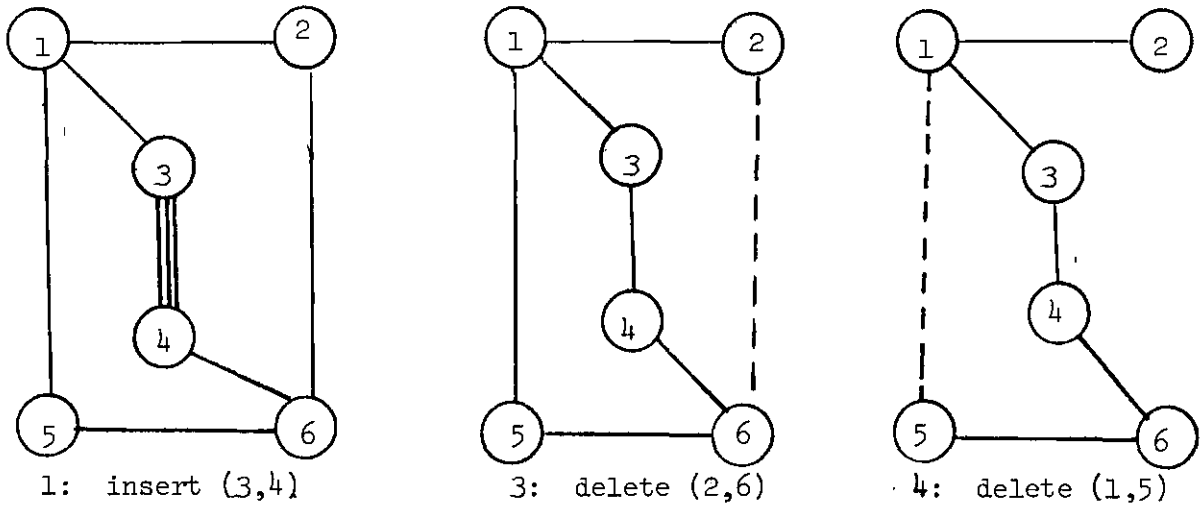


Figure 22. Problem E, Starting With E1, Guideway Changes.

Next, an attempt is made to insert (4,5). Because of limited investigation of the route algorithm no savings are achieved with respect to passenger time or vehicle costs. The guideway cannot be appended to any route, and the first newly constructed route using the guideway results in a cost deterioration. No improvement occurs during route deletion either.

At iteration 3 the guideway (2,6) is deleted, resulting in savings of fixed plus variable costs. The guideway deletion necessitates changing one of the starting routes

(6,2) (2,1) (1,2) (2,6) becomes (2,1) (1,2)

and this restructuring leads to further route improvements.

After deleting (1,5) the algorithm inserts (4,5) and (2,3). At iteration 7 an attempt is made to reinsert (1,5). Guideway (5,6) is then deleted at iteration 8. Here again savings are made in passenger and vehicle costs. The guideway (5,6) was flowless but covered by a route. The route restructuring thus reduced vehicle costs. Further investigation in the route insertion phase produced a new route, providing the savings in passenger costs.

This same set of circumstances is repeated at iteration 9, where guideway (1,2) is deleted. The route algorithm by itself did not obtain the savings until the routes were forcibly restructured.

As expected, the final set of guideways is the tree at iteration 9.

### Test Problem C

The second example illustrating the guideway algorithm is test problem C, which has already been described in Figure 16 and Table 1 in the previous section. Beginning with the set of open guideways and initial routes for problem C1, the 0<sup>th</sup> iteration of the guideway algorithm is the same as the execution of the route algorithm for problem C1, previously described.

At the end of the 0<sup>th</sup> iteration the improvement parameters for closed guideways are

|           |        |
|-----------|--------|
| arc (1,2) | - 417  |
| (1,5)     | -1,296 |
| (3,5)     | -1,012 |
| (7,10)    | - 216  |
| (8,12)    | -1,242 |
| (9,12)    | - 461. |

The two-way guideway connecting stations 1 and 5 has the best value and is thus selected for insertion.

The route algorithm tries to append arc (1,5) to route 1, but this leads to a deterioration of passenger time and vehicle costs:

|         |       |       |       |          |          |       |       |       |
|---------|-------|-------|-------|----------|----------|-------|-------|-------|
| arc     | (8,7) | (7,4) | (4,1) | (1,5)    | (5,1)    | (1,4) | (4,7) | (7,8) |
| service | 14    | 14    | 13    | <u>6</u> | <u>6</u> | 13    | 14    | 14 .  |

The construction routine within the route algorithm then builds a new route

arc (5,1) (1,5)  
 service 3        3

for a slight cost improvement. After some minor adjustments to the other routes, the first guideway algorithm iteration finishes, showing an overall cost deterioration of 111, as shown in Table 9.

The insertion of guideway (1,5) is rejected, the guideway algorithm exits the insertion phase, and the two-way guideway (4,5) is selected for deletion. The second guideway algorithm iteration produces no substantial changes in the routes or flow. There is an overall cost deterioration of 88, and the deletion of (4,5) is rejected. At this point the program stops with the final set of guideways, routes, and flow, shown in Table 9.

Problem C was tested a second time using a different starting set of open guideways. This starting set C2, shown in Figure 23, contains 19 open guideways. The initial routes are the same as for problem C1. Table 10 shows the results of this run.

At iteration one the two-way guideway (1,5) is inserted unsuccessfully. Thereafter, a number of flowless guideways are deleted, (8,12), (4,5), (9,12), (1,2), and (7,10). Since there are no routes traversing these guideways the only changes in total costs are due to guideway fixed cost savings.

At iteration seven guideway (8,10) is deleted, forcing an initial readjustment in route 5:

Table 9. Guideway Algorithm, Results of Problem C, Starting with C1.

---

Starting set of open guideways is

(1,4) (2,3) (2,5) (3,6) (4,5) (4,7) (5,6) (5,8)  
 (6,9) (7,8) (8,9) (8,10) (8,11) (10,11) (11,12)

Initial routes are:

(8,7) (7,4) (4,1) (1,4) (4,7) (7,8)  
 16 16 16 16 16 16

(8,5) (5,2) (2,3) (3,2) (2,5) (5,8)  
 23 23 23 23 23 23

(8,5) (5,6) (6,5) (5,8)  
 15 15 15 15

(7,8) (8,9) (9,8) (8,7)  
 12 12 12 12

(8,10) (10,8)  
 15 15

(8,11) (11,12) (12,11) (11,8)  
 14 14 14 14

| Iter. | Change              |             |        |
|-------|---------------------|-------------|--------|
| 0     | Starting set        | Costs:      |        |
|       |                     | Travel time | 14,226 |
|       |                     | Vehicle     | 6,404  |
|       |                     | Guideway    | 1,900  |
|       |                     | Total       | 22,530 |
| 1     | Open guideway (1,5) | Costs:      |        |
|       |                     | Travel time | 14,035 |
|       |                     | Vehicle     | 6,506  |
|       |                     | Guideway    | 2,100  |

Table 9. Continued.

---

|   |                                 |             |        |
|---|---------------------------------|-------------|--------|
| 1 | Open guideway (1,5) (Continued) | Total       | 22,641 |
|   |                                 | Improvement | - 111  |
| 2 | Close guideway (4,5)            | Costs:      |        |
|   |                                 | Travel time | 14,470 |
|   |                                 | Vehicle     | 6,388  |
|   |                                 | Guideway    | 1,760  |
|   |                                 | Total       | 22,610 |
|   |                                 | Improvement | - 88   |

---

Stop

Final routes are

(8,7) (7,4) (4,1) (1,4) (4,7) (7,8)  
 14 14 13 13 14 14

(8,5) (5,2) (2,3) (3,2) (2,5) (5,8)  
 18 18 11 11 18 18

(8,5) (5,6) (6,3) (3,6) (6,5) (5,8)  
 10 10 5 5 10 10

(7,8) (8,9) (9,6) (6,9) (9,8) (8,7)  
 12 12 3 3 12 12

(8,10) (10,11) (11,10) (10,8)  
 8 6 6 8

(8,11) (11,12) (12,11) (11,8)  
 10 8 8 10

(5,4) (4,5)  
 3 3

Final total costs = 22,530

---

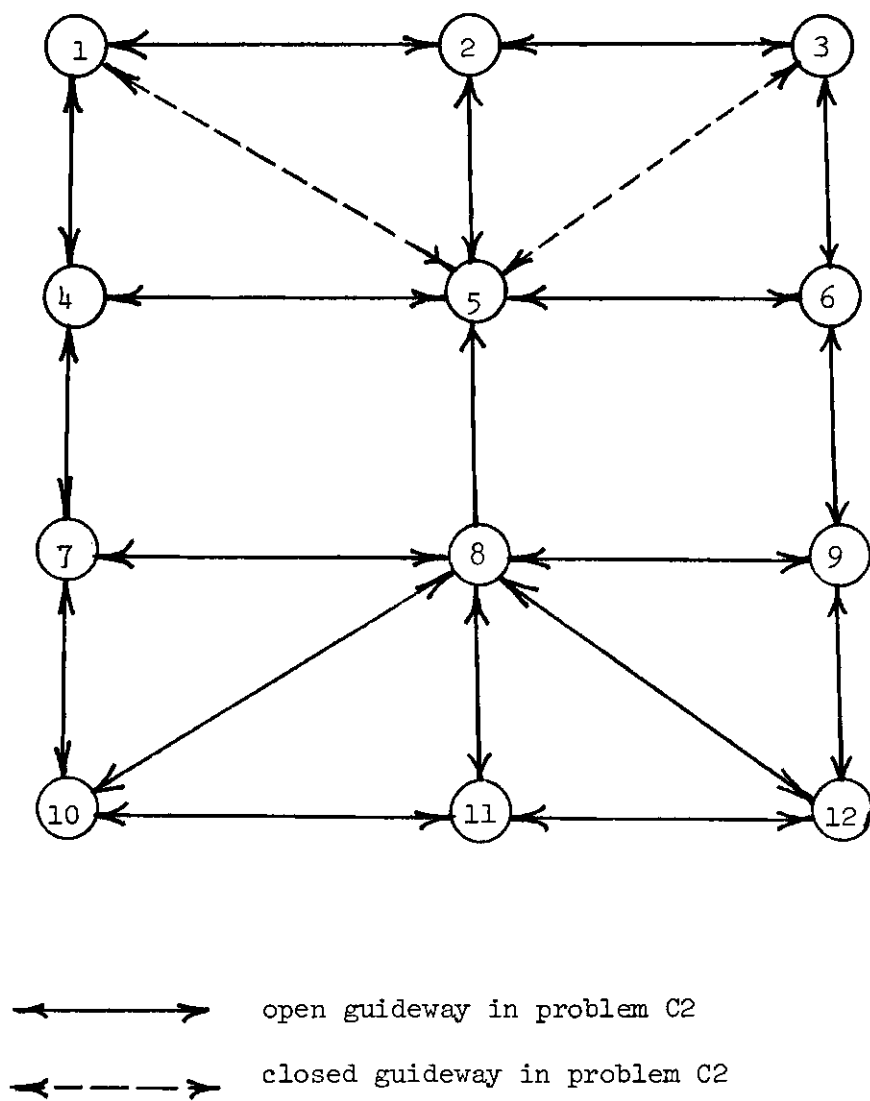


Figure 23. Open Guideways in Problem C2.

Table 10. Guideway Algorithm, Results of  
Problem C, Starting With C2.

---

Starting set of open guideways is:

(1,2) (1,4) (2,3) (2,5) (3,6) (4,5) (4,7) (5,6) (5,8) (6,9)  
(7,8) (7,10) (8,9) (8,10) (8,11) (8,12) (9,12) (10,11) (11,12)

Initial routes are:

(8,7) (7,4) (4,1) (1,4) (4,7) (7,8)  
16 16 16 16 16 16

(8,5) (5,2) (2,3) (3,2) (2,5) (5,8)  
23 23 23 23 23 23

(8,5) (5,6) (6,5) (5,8)  
15 15 15 15

(7,8) (8,9) (9,8) (8,7)  
12 12 12 12

(8,10) (10,8)  
15 15

(8,11) (11,12) (12,11) (11,8)  
14 14 14 14

Iter.                      Change

0                      Starting set

Costs:

Travel time    14,435

Vehicle            6,428

Guideway           2,340

Total            23,203

1                      Open guideway (1,5)

Costs:

Travel time    14,435

Vehicle            6,428

Guideway           2,540



Table 10. Continued.

|   |   |             |        |
|---|---|-------------|--------|
| 1 | Open guideway (1,5) (Continued)               | Total       | 23,403 |
|   |   | Improvement | - 200  |
| 2 | Close guideway (8,12),<br>completely flowless | Costs:      |        |
|   |   | Travel time | 14,435 |
|   |   | Vehicle     | 6,428  |
|   |   | Guideway    | 2,190  |
|   |   | Total       | 23,053 |
|   |   | Improvement | 150    |
| 3 | Close guideway (4,5),<br>completely flowless  | Costs:      |        |
|   |   | Travel time | 14,435 |
|   |   | Vehicle     | 6,428  |
|   |   | Guideway    | 2,050  |
|   |   | Total       | 22,913 |
|   |   | Improvement | 140    |
| 4 | Close guideway (9,12),<br>completely flowless | Costs:      |        |
|   |   | Travel time | 14,435 |
|   |   | Vehicle     | 6,428  |
|   |   | Guideway    | 1,940  |
|   |   | Total       | 22,803 |
|   |   | Improvement | 110    |
| 5 | Close guideway (1,2),<br>completely flowless  | Costs:      |        |
|   |   | Travel time | 14,435 |
|   |   | Vehicle     | 6,428  |

Table 10. Continued.

---

|   |  |             |        |
|---|--|-------------|--------|
| 5 | Close guideway (1,2),<br>completely flowless (Continued) | Guideway    | 1,840  |
|   |  | Total       | 22,703 |
|   |  | Improvement | 100    |
| 6 | Close guideway (7,10),<br>completely flowless            | Costs:      |        |
|   |  | Travel time | 14,435 |
|   |  | Vehicle     | 6,428  |
|   |  | Guideway    | 1,760  |
|   |  | Total       | 22,623 |
|   |  | Improvement | 80     |
| 7 | Close guideway (8,10)                                    | Costs:      |        |
|   |  | Travel time | 14,601 |
|   |  | Vehicle     | 6,628  |
|   |  | Guideway    | 1,510  |
|   |  | Total       | 22,739 |
|   |  | Improvement | -116   |
| 8 | Open guideway (1,5)                                      | Costs:      |        |
|   |  | Travel time | 14,427 |
|   |  | Vehicle     | 6,428  |
|   |  | Guideway    | 1,960  |
|   |  | Total       | 22,815 |
|   |  | Improvement | -192   |

Stop

Table 10. Continued.

---

Final open guideways are:

(1,4) (2,3) (2,5) (3,6) (4,7) (5,6) (5,8) (6,9)  
 (7,8) (8,9) (8,10) (10,11) (11,12)

Final routes are:

(8,7) (7,4) (4,1) (1,4) (4,7) (7,8)  
 16 16 13 13 16 16

(8,5) (5,2) (2,3) (3,2) (2,5) (5,8)  
 18 18 11 11 18 18

(8,5) (5,6) (6,3) (3,6) (6,5) (5,8)  
 10 10 5 5 10 10

(7,8) (8,9) (9,6) (6,9) (9,8) (8,7)  
 12 12 3 3 12 12

(8,10) (10,11) (11,10) (10,8)  
 8 7 7 8

(8,11) (11,12) (12,11) (11,8)  
 10 8 8 10

Final total costs = 22,623

---

|         |        |         |         |        |         |         |         |
|---------|--------|---------|---------|--------|---------|---------|---------|
| arc     | (8,10) | (10,11) | (11,10) | (10,8) | becomes | (10,11) | (11,10) |
| service | 8      | 7       | 7       | 8      |         | 7       | 7.      |

The resulting revised network is infeasible, and five iterations of the route algorithm are necessary merely to achieve feasibility. At the end of guideway algorithm iteration seven route five has the following form:

|         |         |        |       |       |        |         |
|---------|---------|--------|-------|-------|--------|---------|
| arc     | (10,11) | (11,8) | (8,7) | (7,8) | (8,11) | (10,11) |
| service | 14      | 9      | 1     | 1     | 9      | 14.     |

The additional service on the first two branches compensates for the deleted service on (8,10).

The deletion of (8,10) is unsuccessful. At iteration eight the guideway algorithm inserts (4,5), also without success, and consequently stops. The final set of open guideways and vehicle routes is shown in Table 10.

#### Test Problem D

The third example to illustrate the guideway algorithm is shown in Figure 24 and Table 11. The network consists of 12 stations and 25 possible guideways, of which 19 are open in the starting problem D1. Figure 25 shows the form of the vehicle routes at the beginning and end of the 0<sup>th</sup> guideway algorithm iteration. The results of the guideway algorithm are summarized in Table 12.

At iteration one an attempt is made to insert (1,4) into the network. The next three iterations remove successfully (11,12), (10,11),

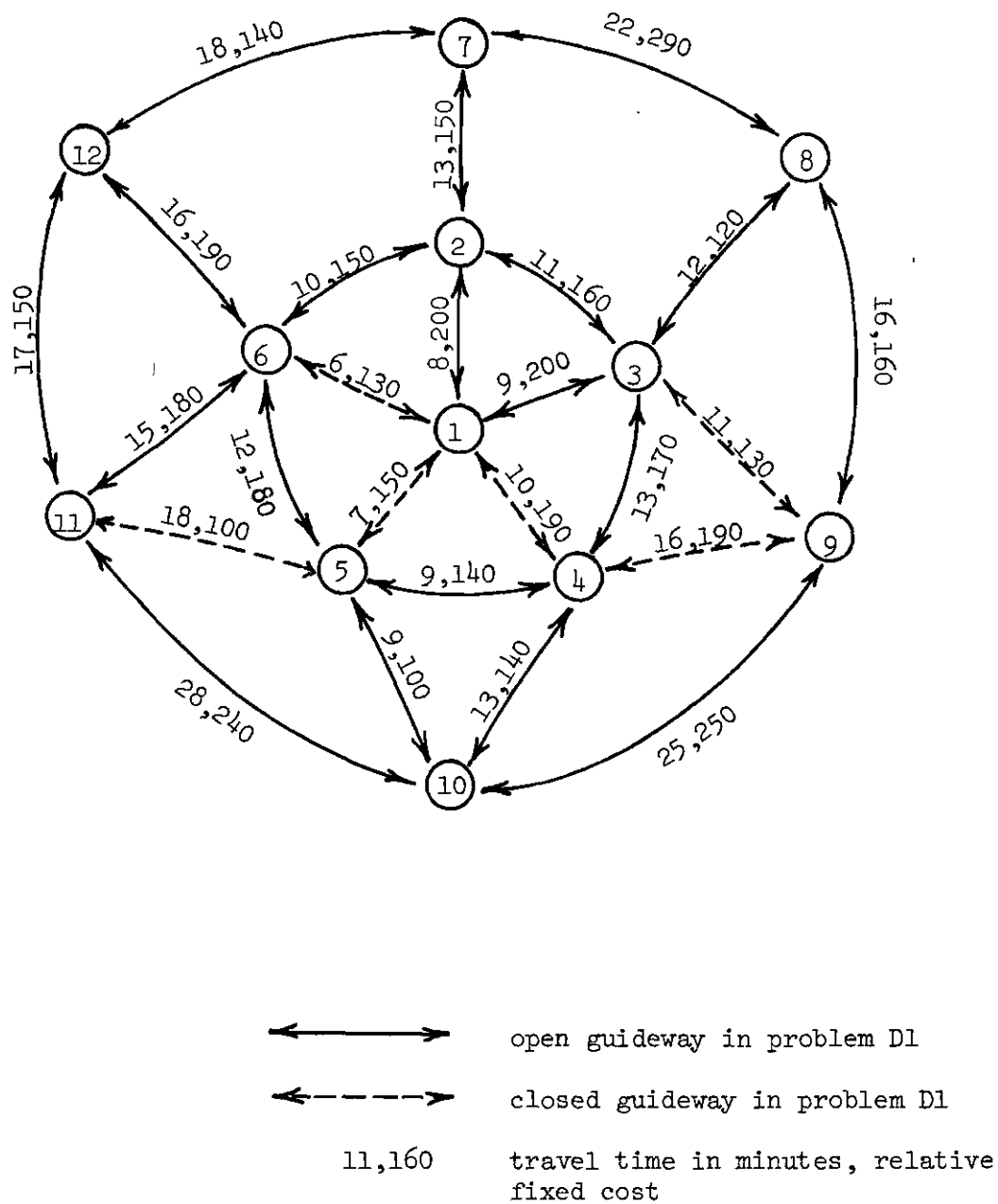


Figure 24. Open Guideways in Problem D1.

Table 11. Problem D, Travel Demands.

| Commodity<br>Number | Origin<br>Station | Destination<br>Station | Travel<br>Demand |
|---------------------|-------------------|------------------------|------------------|
| 1                   | 1                 | 2                      | 300              |
| 2                   | 1                 | 8                      | 200              |
| 3                   | 1                 | 10                     | 200              |
| 4                   | 1                 | 11                     | 200              |
| 5                   | 2                 | 1                      | 300              |
| 6                   | 2                 | 4                      | 200              |
| 7                   | 2                 | 5                      | 200              |
| 8                   | 2                 | 12                     | 200              |
| 9                   | 3                 | 5                      | 300              |
| 10                  | 3                 | 6                      | 300              |
| 11                  | 3                 | 10                     | 200              |
| 12                  | 3                 | 11                     | 200              |
| 13                  | 4                 | 2                      | 300              |
| 14                  | 4                 | 6                      | 300              |
| 15                  | 4                 | 7                      | 200              |
| 16                  | 4                 | 8                      | 200              |
| 17                  | 5                 | 2                      | 300              |
| 18                  | 5                 | 3                      | 300              |
| 19                  | 5                 | 9                      | 200              |
| 20                  | 5                 | 12                     | 200              |
| 21                  | 6                 | 3                      | 300              |
| 22                  | 6                 | 4                      | 400              |
| 23                  | 6                 | 8                      | 200              |
| 24                  | 6                 | 9                      | 200              |
| 25                  | 7                 | 1                      | 300              |
| 26                  | 7                 | 3                      | 300              |
| 27                  | 7                 | 6                      | 300              |
| 28                  | 7                 | 9                      | 200              |
| 29                  | 7                 | 11                     | 200              |
| 30                  | 8                 | 6                      | 200              |
| 31                  | 8                 | 2                      | 300              |
| 32                  | 8                 | 10                     | 200              |
| 33                  | 9                 | 12                     | 200              |
| 34                  | 9                 | 1                      | 200              |
| 35                  | 9                 | 2                      | 300              |
| 36                  | 9                 | 7                      | 200              |
| 37                  | 9                 | 11                     | 100              |
| 38                  | 10                | 1                      | 300              |
| 39                  | 10                | 3                      | 300              |
| 40                  | 10                | 6                      | 300              |
| 41                  | 10                | 8                      | 200              |
| 42                  | 10                | 12                     | 200              |
| 43                  | 11                | 1                      | 200              |

Table 11. Continued.

| Commodity<br>Number | Origin<br>Station | Destination<br>Station | Travel<br>Demand |
|---------------------|-------------------|------------------------|------------------|
| 44                  | 11                | 4                      | 300              |
| 45                  | 11                | 7                      | 200              |
| 46                  | 11                | 9                      | 100              |
| 47                  | 12                | 3                      | 300              |
| 48                  | 12                | 4                      | 300              |
| 49                  | 12                | 8                      | 200              |
| 50                  | 12                | 10                     | 200              |

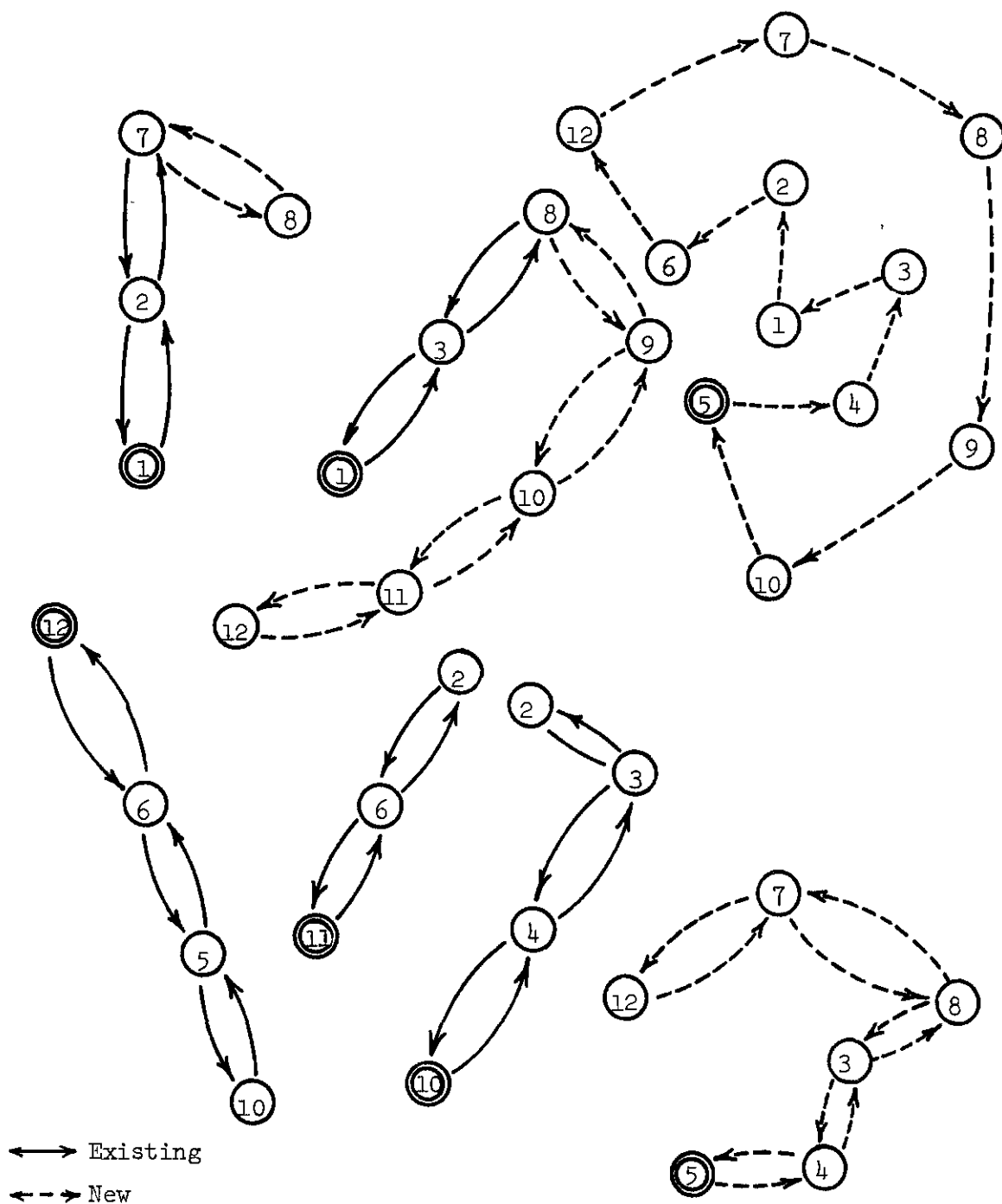


Figure 25. Problem D, Starting With D1, Vehicle Routes at Beginning and End of Guideway Algorithm Iteration 0.



Table 12. Guideway Algorithm, Results of Problem D,  
Starting with D1.

| Iter. | Change         | Pass. Time<br>Costs | Vehicle Optg.<br>Costs | Guideway<br>Fixed Costs | Total  | Impvt. |
|-------|----------------|---------------------|------------------------|-------------------------|--------|--------|
| 0     | Start          | 27,346              | 7,597                  | 3,310                   | 38,253 | -      |
| 1     | Insert (1,4)   | 28,092              | 6,911                  | 3,500                   | 38,503 | -250   |
| 2     | Delete (11,12) | 27,357              | 7,529                  | 3,160                   | 38,046 | 206    |
| 3     | Delete (10,11) | 27,804              | 7,305                  | 2,920                   | 38,029 | 17     |
| 4     | Delete (7,8)   | 29,140              | 5,844                  | 2,630                   | 37,614 | 415    |
| 5     | Insert (1,4)   | 28,657              | 5,820                  | 2,820                   | 37,297 | 316    |
| 6     | Insert (1,5)   | 28,454              | 5,904                  | 2,970                   | 37,328 | -31    |
| 7     | Delete (1,4)   | 28,657              | 5,820                  | 2,630                   | 37,107 | 190    |
| 8     | Insert (1,4)   | 28,657              | 5,820                  | 2,820                   | 37,297 | -190   |
| 9     | Delete (3,8)   | 30,125              | 7,106                  | 2,510                   | 39,741 | -2,634 |

Stop

Final total costs = 37,107

and (7,8). At this point the computer program is restarted, in the guideway insertion phase, and (1,4) is again selected for insertion, at iteration five. This time the insertion is successful, but not because of flow on (1,4). During the removal of (7,8) at iteration four the route algorithm reached a local minimum and stopped. During iteration five this local minimum was bypassed and the route algorithm proceeded to make further improvements totaling 507. At the end of iteration five the guideway (1,4) is completely flowless and not covered by a regular route. The further improvement of 507 is sufficient to result in an overall cost improvement, however, despite the added fixed cost for (1,4).

At iteration six the algorithm tries to insert (1,5), unsuccessfully. At iteration seven the flowless guideway (1,4) is deleted for a saving equal to its fixed cost. The computer program is again restarted, and after two unsuccessful iterations, the algorithm stops. The final sets of guideways and vehicle routes are shown in Figure 26.

### Summary of Results

#### Convergence

Problem C and D were started with several different sets of open guideways and vehicle routes. The results from these computer runs are compared here and evaluated.

Figure 27 shows the final guideways and total costs for six runs of problem C. The runs starting with C1a and C2 have already been described. The starting set C3 is a tree of 11 two-way guideways.

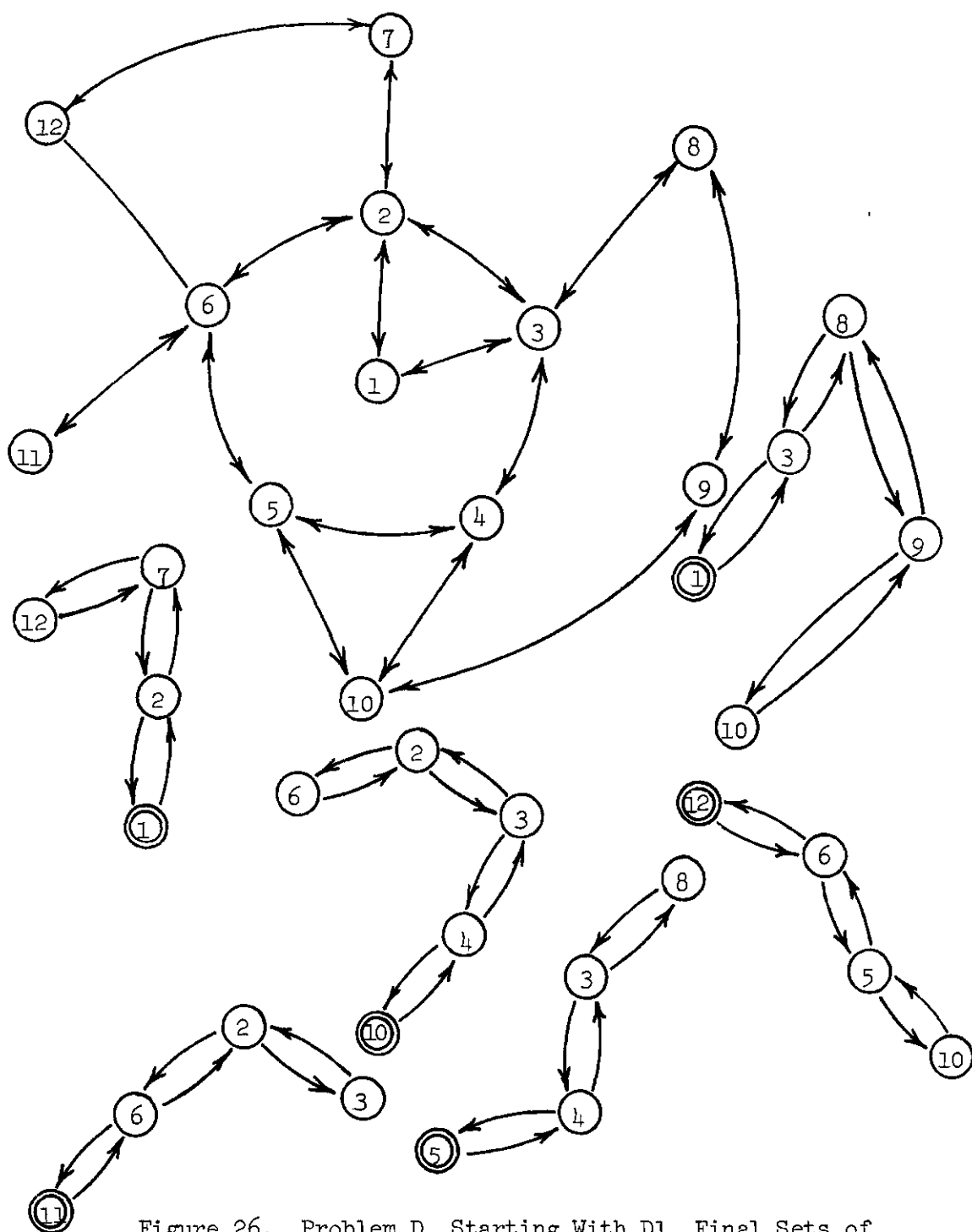


Figure 26. Problem D, Starting With D1, Final Sets of Guideways and Routes.

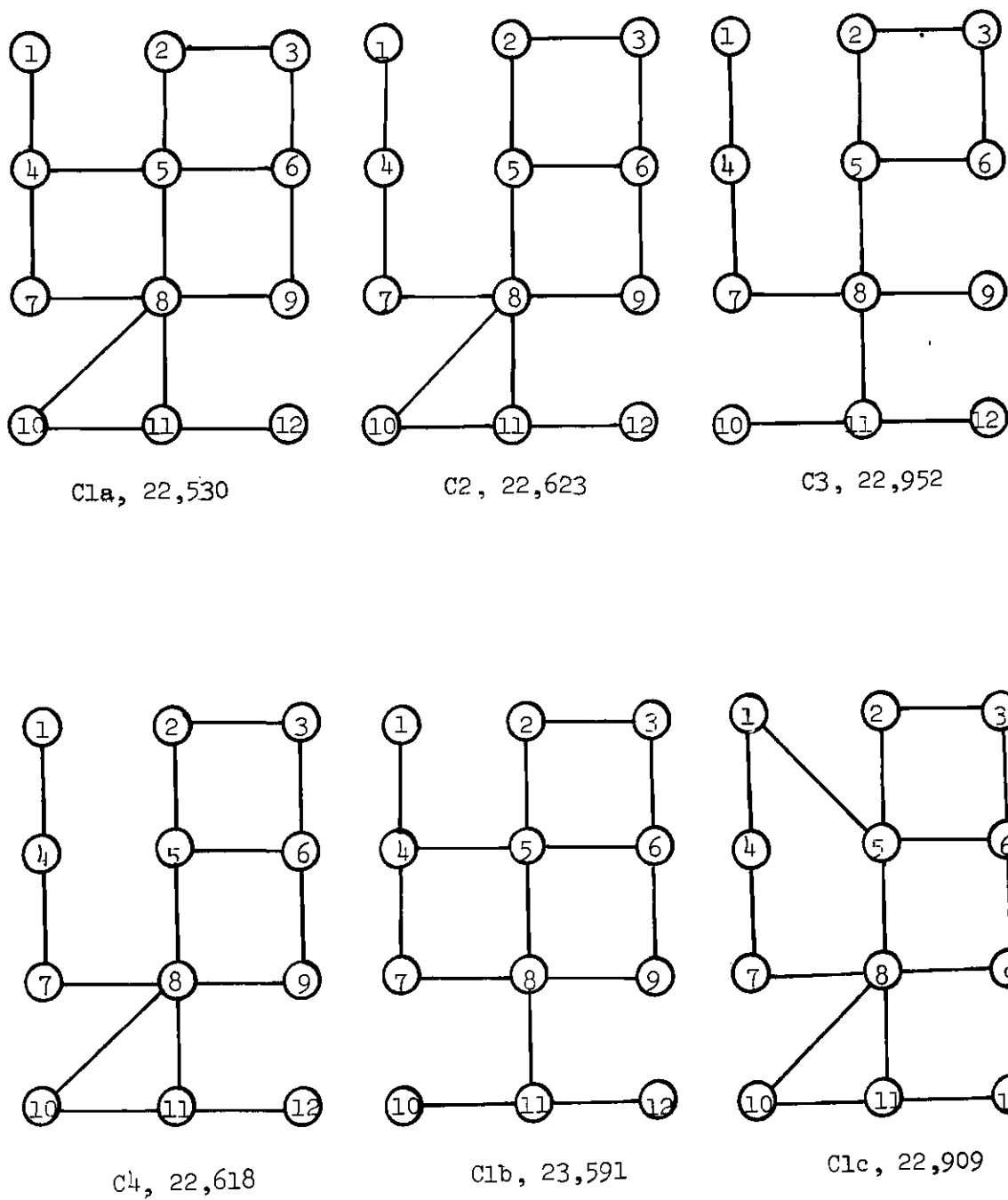


Figure 27. Problem C, Final Sets of Guideways and Total Costs, Various Runs.

The set C<sub>4</sub> is identical to that for C<sub>1a</sub> except that (1,5) has been substituted for (4,5). It will be recalled that when the algorithm started with C<sub>1a</sub>, it attempted to insert (1,5) and then delete (4,5), both changes being unsuccessful. Starting with C<sub>4</sub> the algorithm deleted (1,5) but made no other changes.

The sets C<sub>1b</sub> and C<sub>1c</sub> have the same open guideways as C<sub>1a</sub> but the initial routes differ. It will be instructive to examine these cases in more detail. Figures 28, 29, and 30 show the initial and final vehicle routes at guideway algorithm iteration zero. This allows one to compare the results of the route algorithm, for a given set of open guideways, when started with different initial routes.

The initial route set in C<sub>1a</sub>, Figure 28, consists of CBD-oriented routes. These are changed only slightly, by appending and by constructing one new route. The starting set in C<sub>1b</sub>, Figure 29, consists entirely of reversal runs connecting two stations each. Again, the route changes are limited to a few cases of appending.

A radically different approach is taken in C<sub>1c</sub>, Figure 30. Here the starting set contains only one route, a reversal run connecting stations 8 and 5. The route algorithm then appends four guideway pairs to this initial route and also constructs four new routes.

The respective costs for these three solutions of the route algorithm are

| Starting Set    | Passenger Time Costs | Vehicle Optg. Costs | Sum     |
|-----------------|----------------------|---------------------|---------|
| C <sub>1a</sub> | 14,226               | 6,404               | 20,630  |
| C <sub>1b</sub> | 15,616               | 6,082               | 21,698  |
| C <sub>1c</sub> | 14,296               | 7,464               | 21,760. |

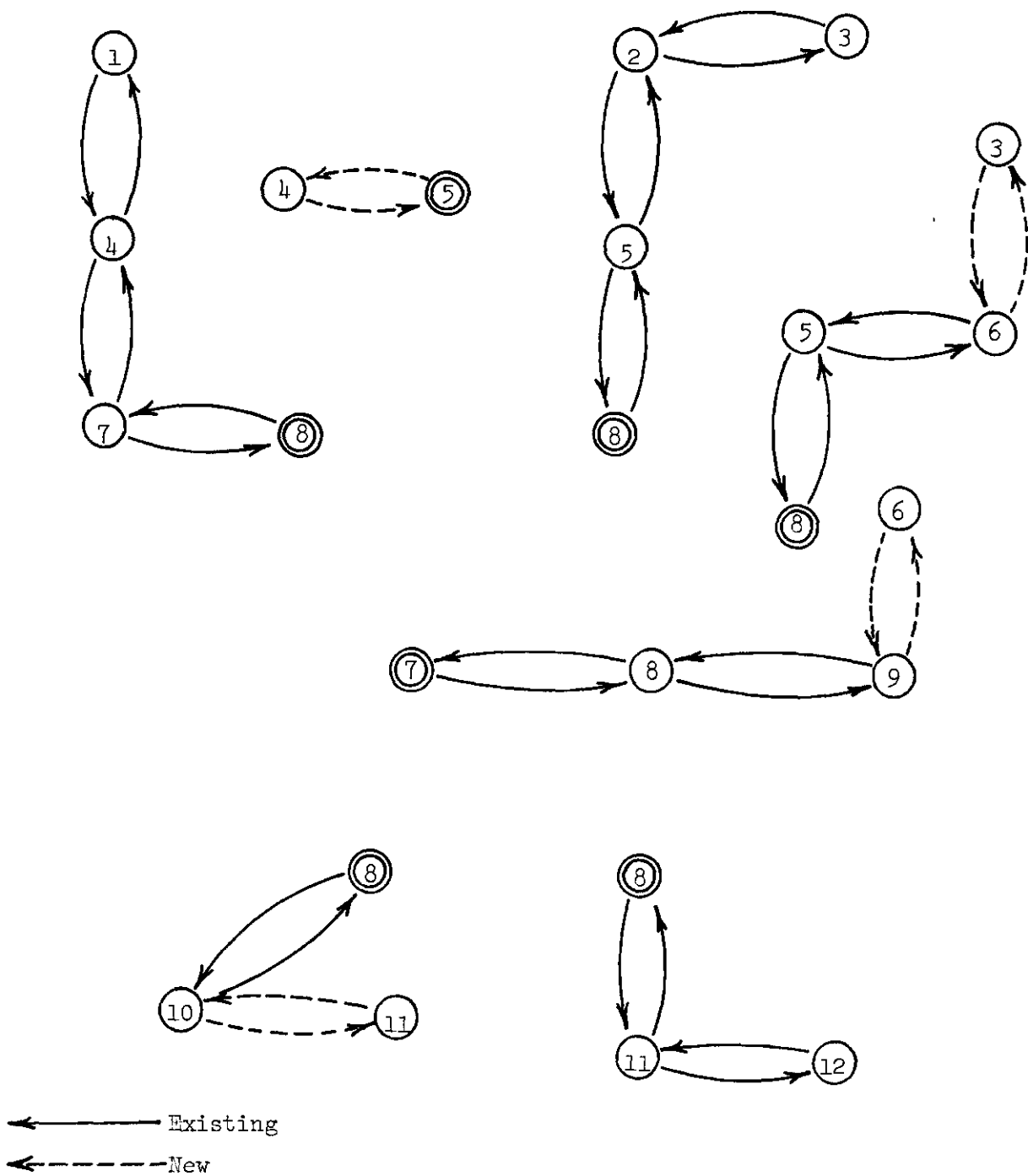


Figure 28. Problem C1, Starting With Cla, Initial and Final Routes for Guideway Set C1.

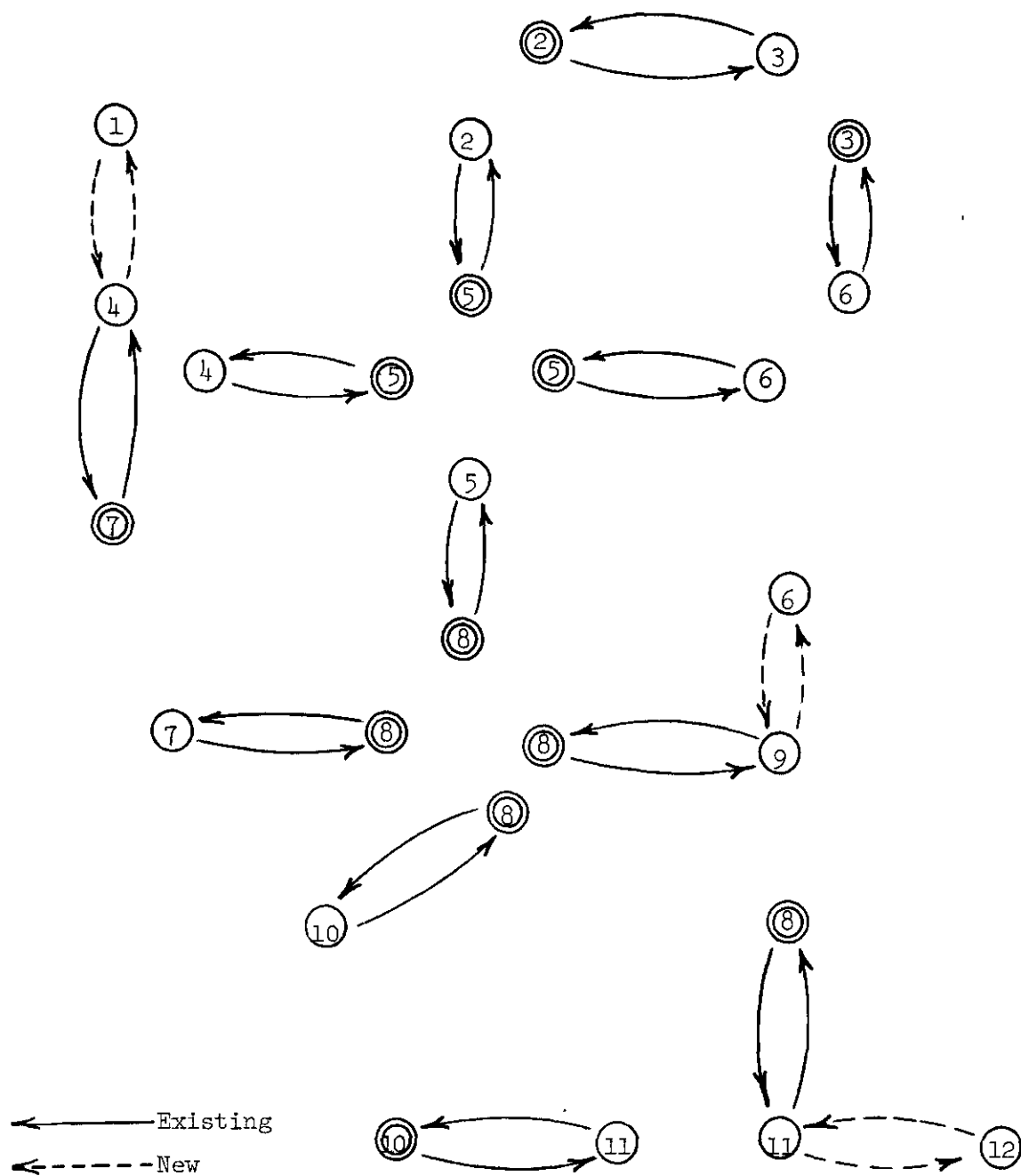


Figure 29. Problem C1, Starting With Clb, Initial and Final Routes for Guideway Set C1.

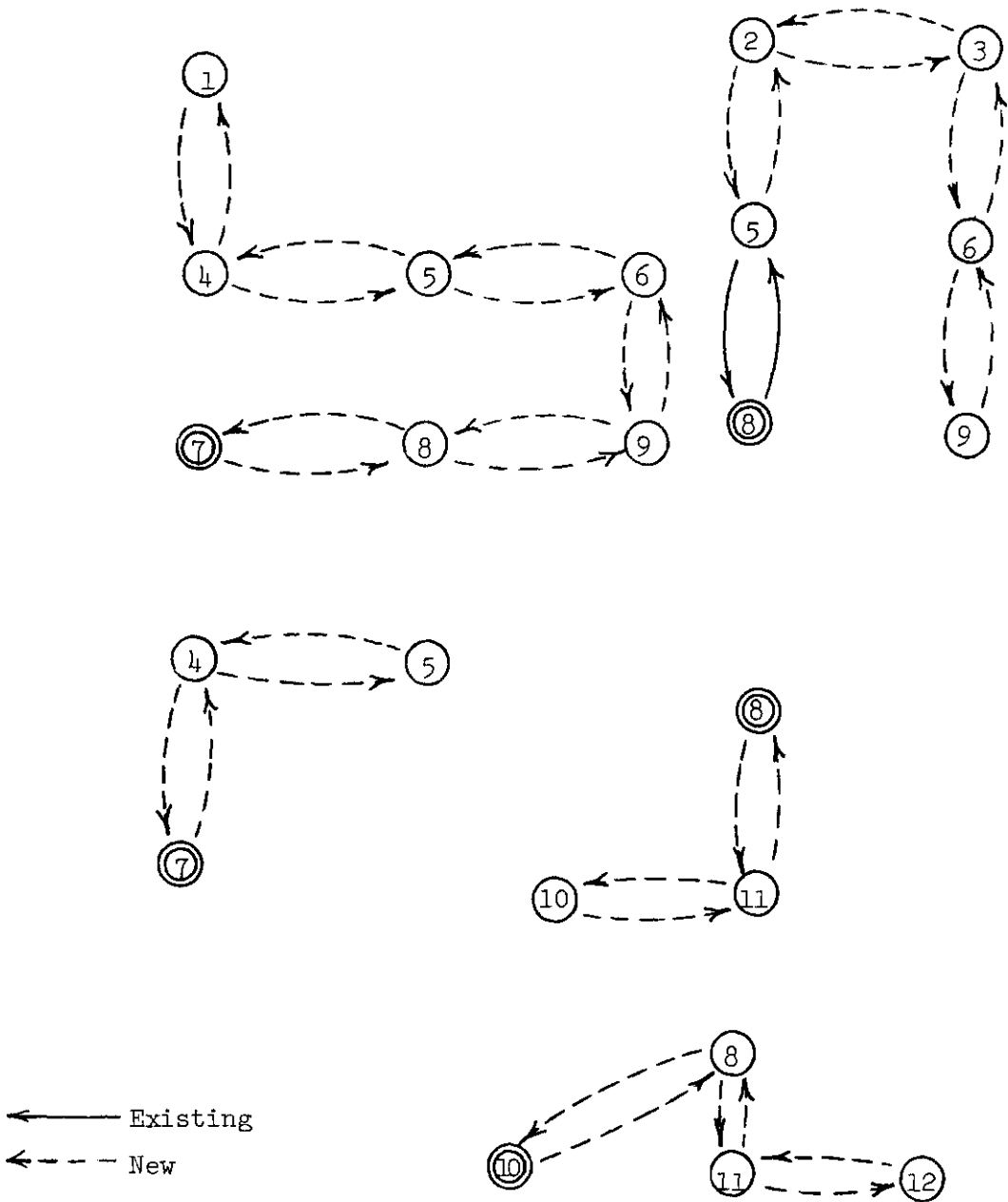


Figure 30. Problem C1, Starting With C1c, Initial and Final Routes for Guideway Set C1.



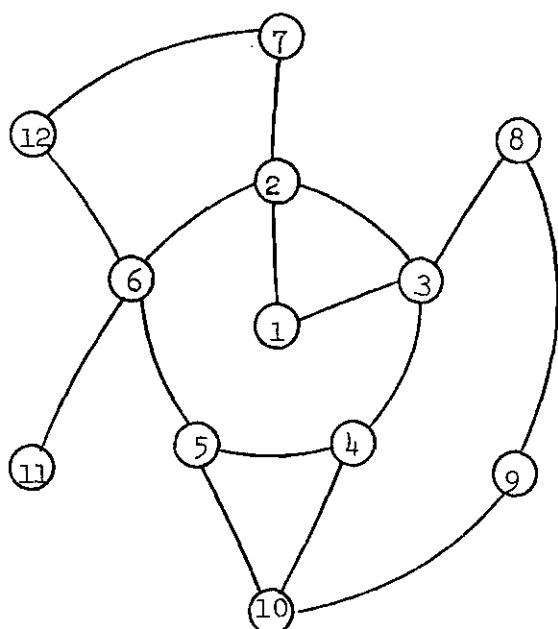
It is seen that the route algorithm is dependent upon the set of starting routes and that the quality of the solutions obtained varies. The main difficulties appear to be failure to try combining routes, and failure to restructure poor routes while maintaining feasibility.

The results of four different runs for Problem D are shown in Figure 31. Problem D1a is the one described in the previous section. The starting set of guideways included 19, of which three were deleted, (11,12), (10,11), and (7,8). The guideway (1,4) was inserted and subsequently deleted.

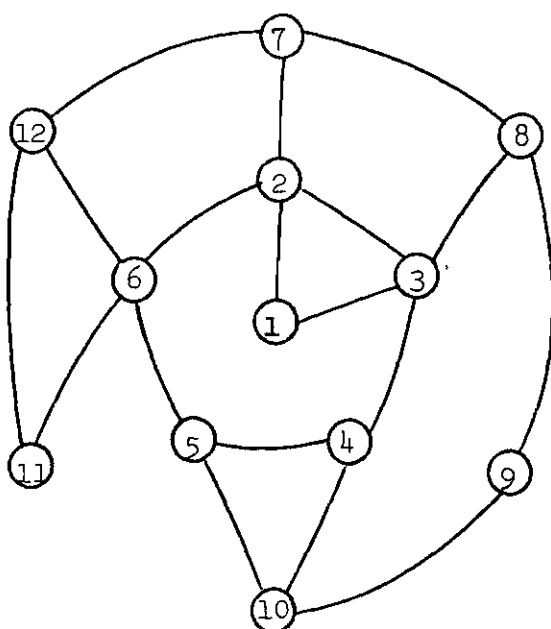
Problem D1b started with the same set of guideways but a different set of routes. Only one guideway change occurred during this run, the deletion of (10,11). There is a substantial difference in the final total costs for the two runs, 37,107 for D1a and 35,461 for D1b. Part of the difficulty may be attributed to the route algorithm. When all 19 guideways were open for D1a the route algorithm obtained total passenger time and vehicle costs of 34,943. At the end of the run starting with D1b, when only 18 of these same guideways were open, the route algorithm obtained costs of 32,391.

Another aspect of the problem lies in selecting guideways to delete. In executing the run starting with D1a the algorithm tried to delete (3,8), resulting in a large cost increase. Since (3,8) is included in all of the final solutions, it appears that the guideway deletion criterion should be changed.

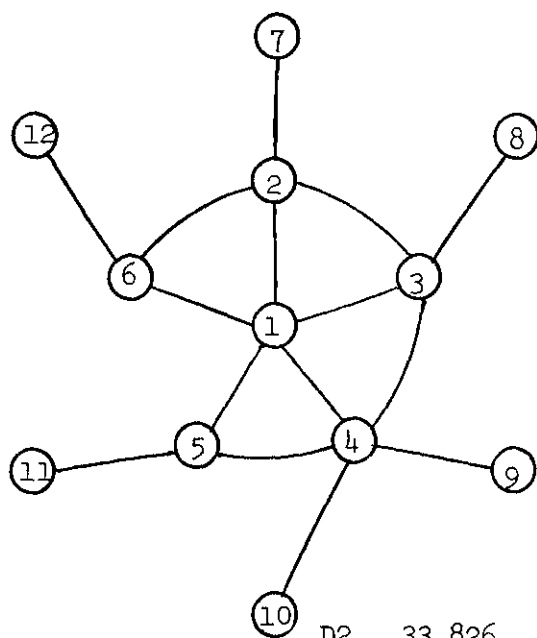
The run beginning with D2 produced no changes in the guideway network and resulted in the lowest total costs. Likewise, no changes



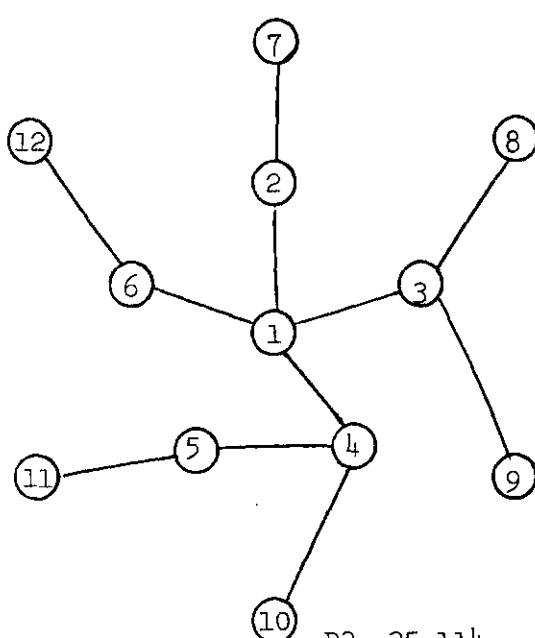
D1a, 37,107



D1b, 35,461



D2, 33,826



D3, 35,114

Figure 31. Problem D, Final Sets of Guideways and Total Costs, Various Runs.

were made in the starting set D3, a tree structure. An attempt was made to insert (9,10), but this resulted in a cost increase. The guideway with the second best improvement parameter was (3,4), but this insertion was not tested. This again points to the need for a more valid guideway improvement parameter.

#### Execution Time

The computer running times for the multicommodity flow assignment routine and for the guideway algorithm are presented in Tables 13 and 14, respectively. The flow assignment routine accounts for most of the time spent during each iteration of the route algorithm. A separate table for the latter is thus not needed.

Table 13 indicates that the execution time of the flow assignment routine is influenced by the size of the network and the percentage of infeasible networks constructed. The ratio of arcs to nodes may be a factor, but there is insufficient data to verify this.

The times for the guideway algorithm range from 0.25 to 3.08 minutes per iteration, and 0.97 to 18.12 minutes per problem. No clear patterns emerge here as the times seem dependent upon starting point and the type of network involved.

Table 13. Summary of Computational Experience  
for Multicommodity Assignment Routine.

| Problem<br>Starting<br>Set | Average<br>Number<br>of Nodes | Average<br>Number<br>of Arcs | Average<br>Number of<br>Zero-Level<br>Routes | Average<br>Percent of<br>Infeasible<br>Networks | Average<br>Time Per<br>Iteration,<br>Seconds,<br>U-1108 |
|----------------------------|-------------------------------|------------------------------|--|---|---|
| C1b                        | 51.2                          | 111.8                        | 2.4  | 28%   | 1.300   |
| C1c                        | 48.3                          | 115.1                        | 4.6  | 72%   | 8.924   |
| C5                         | 40.0                          | 70.0                         | 0.0  | 89%   | 2.510   |
| D1a                        | 60.0                          | 157.4                        | 1.6  | 29%   | 9.241   |
| D1b                        | 64.1                          | 171.6                        | 3.7  | 4%  | 6.007   |
| D2                         | 48.0                          | 106.0                        | 0.0  | 0%  | .530  |
| D3                         | 43.0                          | 83.0                         | 0.0  | 36%   | 4.531   |
| E1                         | 22.0                          | 42.0                         | 0.0  | 0%  | .234  |

Note: Times given are for the 0<sup>th</sup> guideway algorithm iteration in each case.

Table 14. Summary of Computational Experience for  
Guideway Algorithm.

| Problem<br>Starting<br>Set | Number of<br>Iterations<br>of Guideway<br>Algorithm | Ave. No. of Route<br>Alg. Iter. per<br>Guideway Alg. Iter. | Ave. No. of<br>Infeasible<br>Networks per<br>Guideway<br>Alg. Iter. | Time,<br>Minutes,<br>U-1108<br>Per<br>Iter. Total |
|----------------------------|---|--|---|---|
| Cl <sub>a</sub>            | 3   | 14.0   | 2.3   | .32 .97   |
| Cl <sub>b</sub>            | 5   | 15.4   | 5.4   | .50 2.48  |
| Cl <sub>c</sub>            | 7   | 12.6   | 7.6   | 1.05 7.36   |
| C2                         | 9   | 7.2  | 1.6   | .25 2.22  |
| C3                         | 6   | 9.5  | 2.7   | .37 2.20  |
| C4                         | 5   | 13.0   | 3.0   | .28 1.38  |
| D1 <sub>a</sub>            | 10  | 16.5   | 2.2   | 1.81 18.12  |
| D1 <sub>b</sub>            | 5   | 29.4   | 1.3   | 3.08 15.40  |
| D2                         | 2   | 7.5  | 0.0   | .54 1.07  |
| D3                         | 2   | 16.5   | 5.5   | 2.01 4.02   |
| E1                         | 10  | 14.3   | 2.7   | .36 3.62  |

## CHAPTER VI

## FUTURE EXTENSIONS

The solution procedure developed and described in this thesis is by no means ready for designing transit networks of a realistic size. Nor is it particularly efficient for even small networks. In the sections that follow there are presented, as suggestions for future research, a number of alternative strategies for obtaining faster and better solutions and for handling larger networks.

Obtaining an Initial Set of Open Guideways

Other researchers using insertion-deletion algorithms typically start with either a minimal spanning tree or a maximally connected network. The previous chapter on computational results shows clearly that the computer program running time depends heavily upon the starting set of open guideways.

The initial solution proposed here is designed to obtain a better starting set and hence reduce the number of iterations required to achieve the best network. The essential concept is to use the differences between shortest paths and  $n^{\text{th}}$ -shortest paths for the commodities.

The method begins by finding the  $q$  shortest paths for each commodity on the guideway network  $(N,A)$ . As defined previously, the set  $A$  contains all possible guideway arcs. The length of each arc

$(i,j) \in A$  is taken to be the passenger travel time  $d_{ij}$ .

Next, for each arc  $(i,j) \in A$  there is computed a weighting factor:

$$w_{ij} = \sum_{k \in K} \sum_{n=1}^q (r^k) (\ell_{kq} - \ell_{kn}) (m_{knij}) \left( \frac{1}{t_{kn}} \right) \quad (6-1)$$

where

$K$  is the set of commodities, travel demands between node pairs  $ij$

$r^k$  the travel demand for commodity  $k$

$\ell_{kq}$  the length of the  $q^{\text{th}}$ -shortest path for commodity  $k$ , or the "longest" path

$\ell_{kn}$  the length of the  $n^{\text{th}}$ -shortest path for commodity  $k$

$m_{knij}$  a logic operator,  $m_{knij} = 1$  if arc  $(i,j)$  is in the  $n^{\text{th}}$ -shortest path for commodity  $k$  but no shorter path for commodity  $k$ ,  $m_{knij} = 0$  otherwise

$t_{kn}$  the number of arcs in the  $n^{\text{th}}$ -shortest path for commodity  $k$ .

The weighting factor  $w_{ij}$  for an arc represents the potential travel time cost saving to be achieved, summed over all commodities, by introducing the arc into the network. The  $(\ell_{kq} - \ell_{kn})$  is the difference between the  $q^{\text{th}}$ -shortest, or "longest", path and the  $n^{\text{th}}$ -shortest one. The logic operator  $m_{knij}$  attributes the saving only to those arcs that are included in the  $n^{\text{th}}$ -shortest path and no shorter path, thus preventing double counting of savings. The factor  $1/t_{kn}$  divides each saving attributed to an arc by the number of arcs in the  $n^{\text{th}}$ -shortest path for commodity  $k$ .

As an example consider the network in Figure 32 and the associated

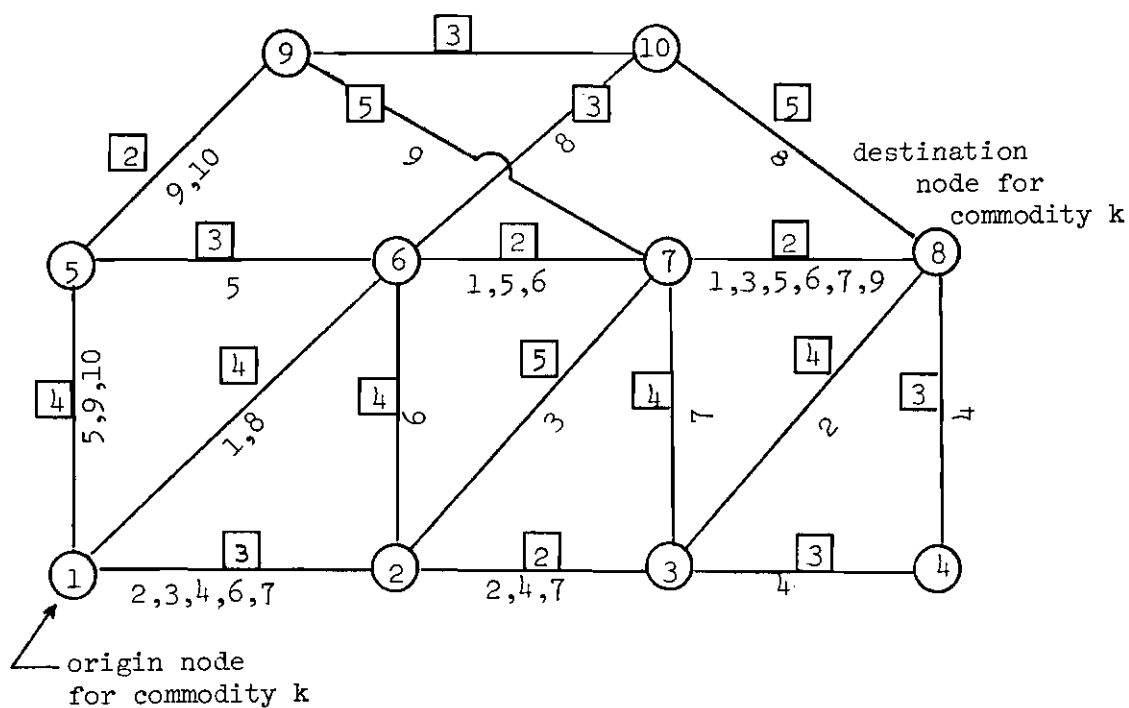


Figure 32. Example for Computing Guideway Weighting Factors.



arc-path incidence matrix in Table 15 for commodity  $k$ , going from node one to eight. If  $q$  is set equal to 10, then the longest path has length  $\ell_{kq} = \ell_{k10} = 14$ . Consider arc (1,2) in the network. It is in the 2nd, 3rd, 4th, 6th, and 7th-shortest paths. Thus, arc (1,2) has attributed to it a potential saving of  $r^k (14 - 9)(1/3)$  for commodity  $k$ . Arc (1,5) is credited with  $r^k (14 - 11)(1/4)$ , arc (1,6) with  $r^k (14 - 8)(1/3)$ , etc.

This weighting scheme thus determines the importance of each arc with respect to potential travel time cost savings for all commodities on that arc. If the vehicle operating costs  $c_{ij}$  vary differently than the arc traversal times, then the length of each arc may be taken to be  $(d_{ij} + c_{ij}/g)$ , where  $g$  is the capacity of a vehicle. The weighting factors will then reflect also some potential savings in this  $c_{ij}$ . In practice the number of shortest paths  $q$  may be limited to three, four, or five.

The weighting factors  $w_{ij}$  for each guideway pair may now be combined with the guideway fixed costs  $p_{ij}$  to form a priority ranking that reflects both variable and fixed costs:

$$h_{ij} = w_{ij} + w_{ji} - p_{ij} - p_{ji}. \quad (6-2)$$

These priority ratings  $h_{ij}$  can be used to select guideway pairs for the initial set  $A$ .

One way of doing so is to insert guideway pairs in descending order of  $h_{ij}$  until all stations are connected. The resulting network

Table 15. Example for Computing Guideway Weighting Factors. Arc-Path Incidence Matrix for Commodity (1,8).

| o.   | 1 | 1 | 1 | 2 | 2 | 2 | 3 | 3 | 3 | 4 | 5 | 5 | 6 | 6  | 7 | 9 | 9  | 10 |          |          |
|------|---|---|---|---|---|---|---|---|---|---|---|---|---|----|---|---|----|----|----------|----------|
| d.   | 2 | 5 | 6 | 3 | 6 | 7 | 4 | 7 | 8 | 8 | 6 | 9 | 7 | 10 | 8 | 7 | 10 | 8  | $l_{kn}$ | $t_{kn}$ |
| path |   |   |   |   |   |   |   |   |   |   |   |   |   |    |   |   |    |    |          |          |
| 1    |   |   | 1 |   |   |   |   |   |   |   |   |   | 1 |    | 1 |   |    |    | 8        | 3        |
| 2    | 1 |   |   | 1 |   |   |   |   | 1 |   |   |   |   |    |   |   |    |    | 9        | 3        |
| 3    | 1 |   |   |   |   | 1 |   |   |   |   |   |   |   |    | 1 |   |    |    | 10       | 3        |
| 4    | 1 |   |   | 1 |   |   | 1 |   |   | 1 |   |   |   |    |   |   |    |    | 11       | 4        |
| 5    |   | 1 |   |   |   |   |   |   |   |   | 1 |   | 1 |    | 1 |   |    |    | 11       | 4        |
| 6    | 1 |   |   |   | 1 |   |   |   |   |   |   |   | 1 |    | 1 |   |    |    | 11       | 4        |
| 7    | 1 |   |   | 1 |   |   |   | 1 |   |   |   |   |   |    |   | 1 |    |    | 11       | 4        |
| 8    |   |   | 1 |   |   |   |   |   |   |   |   |   |   | 1  |   |   |    | 1  | 12       | 3        |
| 9    |   | 1 |   |   |   |   |   |   |   |   |   | 1 |   |    | 1 | 1 |    |    | 13       | 4        |
| 10   |   | 1 |   |   |   |   |   |   |   |   |   | 1 |   |    |   |   | 1  | 1  | 14       | 4        |

will almost certainly contain more guideways than a spanning tree (constructed of two-way guideways equivalent to the pairs), but far fewer than the complete guideway network. Also, this network will probably have more guideway arcs than the final, best network. The reason for this lies in the fact that more than 100 percent of the travel cost savings are attributed to arcs.

Another way of selecting a starting set A is to construct a tree (of equivalent two-way guideways). Guideway pairs are inserted in descending order of  $h_{ij}$ , provided no cycles are formed.

Hopefully, the computations involved in determining these  $h_{ij}$  will be justified by the improved quality of the initial network.

### Alternative Method of Generating Guideway

#### Improvement Parameters

The guideway insertion and deletion parameters generated during the solution procedure, as described in Chapter IV, are influenced by the current set of vehicle routes. A method of eliminating such bias is to imbed the route algorithm directly into Billheimer's guideway insertion-deletion algorithm (8).

Billheimer's algorithm solves the following problem:

$$\text{Minimize } Z_4 = \sum_{(i,j) \in A} u_{ij} p_{ij} + \sum_{(i,j) \in A} \sum_{k \in K} f_{ij}^k v_{ij} \quad (6-3)$$

(guideway fixed costs)      (variable flow costs)

Subject to:

Flow requirements:

$$\sum_{j \in A(i)} f_{ij}^k - \sum_{j \in B(i)} f_{ji}^k = r_i^k, \quad k \in K, i \in N \quad (6-4)$$

Integrality:

$$u_{ij} = 0, 1, \quad (i, j) \in A \quad (6-5)$$

Non-negativity and feasibility:

$$\left. \begin{array}{l} f_{ij}^k = 0 \text{ if } u_{ij} = 0 \\ \geq 0 \text{ if } u_{ij} = 1 \end{array} \right\} (i, j) \in A \quad (6-6)$$

where

$v_{ij}$  is the variable unit flow cost on arc  $(i, j)$ ,  
and other terms are as previously defined.

This problem is identical to that posed for the general arc insertion-deletion algorithm in Chapter III, (3-13) through (3-17), except for the absence of arc capacity restrictions. Equation (6-6) implies that an open guideway arc has unlimited capacity.

The solution for (6-3) through (6-6) is obtained by an arc insertion-deletion algorithm. The absence of the capacity constraints simplifies the problem considerably in several ways:

- (1) Bounds can be established on variable cost savings for

including arcs. These bounds can then be used to identify certain arcs that must be included in the optimal solution and to identify others that must be excluded. This has the effect of reducing the problem size.

(2) Since the second shortest path can always accept the flow, there is no need to find  $n^{\text{th}}$ -shortest paths beyond the second one.

(3) Computing arc insertion and deletion parameters is much more direct, involving only a rerouting of relatively few commodities for each arc.

The route algorithm can be imbedded in Billheimer's guideway algorithm by appropriate definition of the  $v_{ij}$ :

$$v_{ij} = d_{ij} + \frac{c_{ij}}{g q_1} + d_w t_w q_2 + d_t q_3 \quad (6-7)$$

where

- $q_1$  is the average percent vehicle occupancy, throughout the network,
- $q_2$  is the average percentage of passengers on a vehicle who experienced a waiting time at the origin of the arc being traversed,
- $q_3$  the average percentage of passengers on a vehicle who experienced a transfer at the origin of the arc being traversed,
- $t_w$  the average waiting time experienced by passengers when boarding a vehicle,

and other terms are as previously defined.

The variable flow cost  $v_{ij}$  for each arc thus consists of

$d_{ij}$ , the variable passenger travel time cost on the arc,

- $c_{ij}/gq_1$     an average per passenger cost of operating a vehicle on the arc,  
 $d_w t_w q_2$     an average per passenger waiting time cost,  
 $d_t q_2$     an average per passenger transfer time cost.

This definition of  $v_{ij}$  is designed to allow one to work with the original guideway network in comparing guideway fixed costs  $p_{ij}$  versus variable costs  $d_{ij}$ ,  $c_{ij}$ ,  $d_w$ , and  $d_t$ , unencumbered by any set of vehicle routes. Values for the parameters  $q_1$ ,  $q_2$ ,  $q_3$ , and  $t_w$  can be determined from particular solutions to the route algorithm and updated as necessary.

A more sophisticated definition of  $v_{ij}$  would use different values of  $q_1$ ,  $q_2$ ,  $q_3$ , and  $t_w$  for different portions of the network. The differences in values would most likely relate to the dissimilar types of flow in the central and outer portions of the network.

#### Alternative Method of Generating Arc Deletion Parameters

While the arc insertion parameters are a natural by-product of the multicommodity assignment routine, the computation of arc deletion parameters is, at best, a piecemeal effort. The alternative method described below is suggested to remedy this situation.

The new procedure would use an entity similar to the arc infeasibility parameter, except that now excess arc capacity would be measured. Recall that at each iteration of the multicommodity routine the infeasibility parameter of an oversaturated arc is incremented by  $-(\text{excess flow} \times \text{eps})$ . This is step nine of the multicommodity routine, described in Chapter IV. After the commodity  $k^*$  has been reassigned

from its shortest to its second shortest path in step ten, some of the arcs will still have excess capacity. For each such arc define an excess capacity parameter, and increment it by (excess capacity  $\times$  eps).

At the termination of the multicommodity routine this parameter for an arc will represent the cumulative excess capacity of the arc during the process of changing from an initial infeasible flow assignment to a final feasible assignment. These parameters can then be used to determine which section of which route to reduce.

Just as the infeasibility parameters, the excess capacity parameters would be generated for all arcs each time a flow assignment is made.

#### Consistent Designation of Revised Network Nodes and Arcs

One of the most time consuming operations in the computer program is the multicommodity flow assignment routine. This routine is executed each time a change is made in the set of vehicle routes, be it a completely new route or a change in route service frequency on only one branch. Considerable time could be saved if the flow assignment routine were able to start with the flow from the previous execution.

A major reason for not doing so is the method of constructing the revised network. Nodes are generated in order of stations and arcs in order of stations within a route. Arcs are then reordered by node number and, within that ordering, by length. The flow assignment routine then takes advantage of this arc ordering.

Several modifications are needed if the multi-commodity flow routine is to start with a previous assignment:

(1) There must be a consistent system of designating route nodes and arcs in the revised network. In the current program a revised network node can be traced directly to its station and route. To this identification must be added the position within the route.

(2) A method must be devised for properly constructing new arcs and deleting old arcs, and for changing arc capacities and lengths, whenever changes are made in the set of routes.

(3) Deleted arcs must be kept in the network long enough for the multicommodity routine to reassign their flow. On the other hand, such deleted arcs must not be allowed to swell the list of total arcs beyond the declared matrix size.

(4) The multicommodity routine must be changed to handle unordered arcs.

(5) A way must be found for updating arc infeasibility parameters from one iteration to the next.

If the changes (1), (2), and (3) were made it might be worthwhile to attempt using an exact multicommodity routine. An additional benefit would then be the information embodied in the dual variables.

#### Multiple Guideway Changes

Inserting and deleting guideways one at a time overlooks the possibility of reducing total costs by making multiple guideway changes. The simplest such case is the simultaneous insertion of one guideway and deletion of another. Either change by itself may result in higher



total costs while the combined changes may produce an overall improvement. The guideway insertion parameters may be used for selecting the guideways involved in the substitution process.

More difficult is the extension to two or more guideways for insertion, deletion, or substitution. Consider the partial network shown in Figure 33. If there is a substantial travel demand between stations 1 and 3, the insertion of either (1,4) or (4,3) by itself will

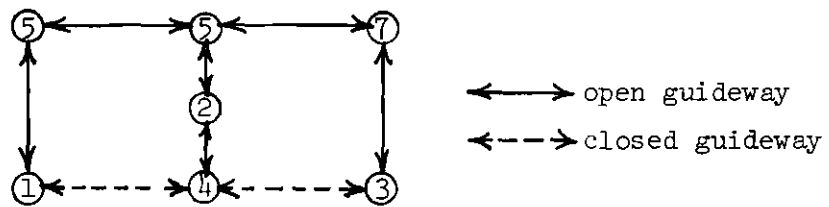


Figure 33. Example Relating to Multiple Guideway Changes.

do little to reduce variable costs for that demand. What is needed, of course, is the simultaneous insertion of (1,4) and (4,3). In order to reduce total costs it may be necessary to delete at the same time one or more guideways.

The problem of multiple guideway changes bears some resemblance to that of selecting route arcs to change. The differences are that guideways are either open or closed and that they need not conform to any pattern. Nevertheless, it may be possible to extend the general concepts of the route algorithm to the problem here.

## Computer Program Improvements

### Packing the Data

Some 14,000 words of core data storage are required for a problem consisting of 12 stations, 30 possible two-way guideways, 50 travel demands, and 15 routes of length not exceeding 12 arcs each. To handle problems of 50 stations, 100 guideways, 200 travel demands, and 50 routes of length 30 arcs, there are needed approximately 78,000 words. Considering the 15,000 words needed for instructions, the total core requirement for such a problem becomes 93,000 words. This amount exceeds the 65,000 word capability of the Univac 1108 on the Georgia Tech campus, and that of many other facilities.

The largest arrays involved are the shortest path and the second shortest path matrices, of size 200 x 30 each for the second problem described above. Since the number of revised network arcs is limited, to 2000 in this case, the largest entry in the shortest path matrix occupies only 11 bits ( $2^{11} = 2048$ ), or less than one-third of a 36-bit word. Hence, one can use the FLD function (48) to pack each 200 x 30 array into an 200 x 10 array. Similar storage space savings can be made with the other arrays involved. Such data packing makes it possible to handle problems of moderate size on existing computers without the use of external storage devices.

### Adaptive Switching Rules

There appear to be an endless number of switching rules to control the progress of computations through the various phases of the guideway and route algorithms. A few of these are available as options

in the existing program: the insertion part of the route algorithm can enter first the section on increasing service on routes or the section on appending guideways, and cut-off parameters for preliminary tests can be adjusted.

It may be helpful to expand on these to include such cases as the following: (1) Enter guideway deletion first if there are relatively few closed guideways. (2) If route construction is unsuccessful at the determined service frequency, test the same route with another frequency. (3) If a guideway insertion is unsuccessful do not allow the same guideway to become the first candidate for insertion at a later entry of the guideway insertion phase.

#### Bounds on Optimal Solution

Both Gray (17) and Billheimer (8) used bounds on variable and fixed costs to reduce the number of solutions to be examined. Recall Billheimer's problem (6-3) through (6-6). An upper bound on the fixed costs  $p_{ij}$  can be obtained as follows:

$$\begin{aligned} \left[ \begin{array}{l} \text{Upper bound on} \\ \text{fixed costs } p_{ij} \end{array} \right] &= \left[ \begin{array}{l} \text{Total costs } v_{ij} \text{ and } p_{ij} \text{ for} \\ \text{a feasible solution} \end{array} \right] & (6-8) \\ &- \left[ \begin{array}{l} \text{Minimum variable costs } v_{ij} \\ \text{when all guideways are open} \end{array} \right] \end{aligned}$$

The variable costs are minimized when all guideways are open. The fixed costs for the optimal solution cannot exceed the difference between the total costs for any feasible solution and this minimum variable cost. As better solutions are obtained the bound is tightened.

A lower bound on fixed costs may be obtained in the following manner: First, there is computed for each guideway the increase in variable flow costs for the travel demands between the endpoints if the guideway were closed and all others remained open. If this increase in variable costs  $v_{ij}$  exceeds the guideway fixed cost  $p_{ij}$ , then that guideway must be open in the optimal solution. All such guideways are identified, and then a minimal connected graph is constructed containing them. The fixed costs for this graph then represent a lower bound on those for the optimal solution. In the process of obtaining this bound the problem size has been reduced by identifying guideways that must appear in the optimal solution.

While the route structure complicates matters in the research problem, it should be possible to extend and apply the same principles to its solution.

## CHAPTER VII

### CONCLUSIONS AND RECOMMENDATIONS

The focus of this research has been to develop a procedure for solving in a unified way the four-component problem that follows after estimating the travel demand and determining the station locations for a proposed public transit network: How should one simultaneously select fixed-cost guideways for inclusion in the network, determine vehicle routes and route service frequencies, and assign passengers to origin-destination paths in order to minimize the total of construction costs, passenger travel and delay time costs, and vehicle operating costs, while satisfying the total transportation demand?

To date, no successful work on this problem has appeared in the literature. There are two main results of this research:

(1) The problem was formulated in a manner amenable to solution. The original multicommodity transshipment problem with all its constraints was decomposed into two problems, one imbedded within the other. The imbedded problem, the selection of vehicle routes and assignment of passenger flow, was further transformed into a simpler problem utilizing a revised network.

(2) A heuristic program to solve the problem was developed, written, and tested. The largest problem tested included 12 stations, 25 possible two-way guideways, 50 travel demands, and 15 routes of length not exceeding 12 arcs each. Solution times ranged from one to

18 minutes on the Univac 1108, with all data held in core.

These results indicate that the objectives of this research have been attained. During the process of testing the program with different examples, a number of problem areas were discovered relating to convergence and execution time. Accordingly, there are recommended for future research the following improvements and extensions: '

(1) The main emphasis in the route algorithm is on extending existing routes and building new routes. There needs to be incorporated a way to combine existing routes and to restructure poor routes. The alternative arc deletion parameters can be used here. These are described in Chapter VI and are a direct analogy of the arc insertion parameters. Also, a route substitution process needs to be devised to overcome the problems of changing one route at a time.

(2) The multicommodity flow assignment routine needs to be improved so that it will obtain with better reliability a feasible solution when one exists, and diagnose an infeasible network in a shorter time. The current routine is capable of handling problems with 12 commodities corresponding to 50 travel demands, 200 nodes, and 380 arcs.

(3) The selection of guideways for insertion and deletion needs to be reexamined. The alternative guideway improvement parameters, described in Chapter VI, appear better suited than those currently used since they avoid the bias of the existing route structure. Here, also, a substitution process would overcome the difficulties of changing one guideway at a time.

In addition to the above there are recommended a number of items to speed convergence and handle larger networks:

(4) An initial set of guideways should be selected using some method, such as described in Chapter VI. The computational experience indicates that the best solutions and execution times are obtained when one starts with a guideway set larger than a spanning tree but smaller than a maximally connected graph.

(5) The principles utilized by Billheimer (8), discussed in Chapter VI, should be extended to identify guideways that must be included in the optimal solution and to obtain bounds on fixed and variable costs.

(6) The data should be packed to enable the handling of larger problems in core.

(7) A method for obtaining an initial set of routes should be developed and tested. Such a method could be utilized at the beginning of the program or at each iteration of the guideway algorithm.

## BIBLIOGRAPHY

1. Aburto-Avila, Jose Luis, Optimal Design of Transportation Networks with Fluctuating Demand - A Case in Multicommodity Network Flows, Ph.D. Thesis, Stanford University, 1973.
2. Balinski, Michel L., "Fixed-Cost Transportation Problems," Naval Research Logistics Quarterly, Vol. 8, No. 1, 41-54, 1961.
3. Bartlett, T. E., "An Algorithm for the Minimum Number of Transport Units to Maintain a Fixed Schedule," Naval Research Logistics Quarterly, Vol. 4, No. 2, 139-149, 1957.
4. Bartlett, T. E., and A. Charnes, "Cyclic Scheduling and Combinatorial Topology; Assignment and Routing of Motive Power to Meet Scheduling and Maintenance Requirements; Part II, Generalization and Analysis," Naval Research Logistics Quarterly, Vol. 4, No. 3, 207-220, 1957.
5. Beckman, Martin, C. B. McGuire, and C. B. Winston, Studies in the Economics of Transportation, Cowley Commission for Research in Economics, Yale University Press, New Haven, 1956.
6. Bergendahl, Göran, "A Combined Linear and Dynamic Programming Model for Interdependent Road Investment Planning," Transportation Research, Vol. 3, No. 2, 211-228, 1969.
7. Bett, Titian J., A Mathematical Model Describing Passenger-Vehicle Interaction and its Applications to Vehicle Scheduling, Ph.D. Thesis, Marquette University, 1972.
8. Billheimer, John W., Optimal Route Configurations with Fixed Link Construction Costs, Ph.D. Thesis, Stanford University, 1970.
9. Cooper, L., and C. Drebes, "An Approximate Method for the Fixed Charge Problem," Naval Research Logistics Quarterly, Vol. 14, No. 1, 101-113, 1967.
10. Creighton, Roger L., Urban Transportation Planning, University of Illinois Press, 1970.
11. Cremeans, J. E., R. A. Smith and G. R. Tyndall, "Optimal Multi-commodity Network Flows with Resource Allocation," Naval Research Logistics Quarterly, Vol. 17, No. 3, 269-279, 1970.



12. Dantzig, G. B., and D. R. Fulkerson, "Minimizing the Number of Tankers to Meet a Fixed Schedule," Naval Research Logistics Quarterly, Vol. 1, No. 3, 217-222, 1954.
13. Dwyer, Paul J., "Use of Completely Reduced Matrices in Solving Transportation Problems with Fixed Charges," Naval Research Logistics Quarterly, Vol. 13, No. 3, 289-313, 1966.
14. Ford, L. R., and D. R. Fulkerson, "A Suggested Computation for Maximal Multicommodity Network Flows," Management Science, Vol. 5, No. 1, 97-101, 1958.
15. Ford, L. R., and D. R. Fulkerson, Flows in Networks, Princeton University Press, Princeton, New Jersey, 1962.
16. Frank, Ronald Stewart, On the Fixed-Charge Hitchcock Transportation Problem, Ph.D. Thesis, Johns Hopkins University, 1972.
17. Gray, Paul, "Exact Solution of the Fixed-Charge Transportation Problem," Operations Research, Vol. 19, No. 6, 1529-1538, 1971.
18. Grigoriadis, M. D., and W. W. White, "A Partitioning Algorithm for the Multicommodity Network Flow Problem," Mathematical Programming, Vol. 3, No. 2, 157-177, 1972.
19. Hartman, J. K., and L. S. Lasdon, "A Generalized Upper Bounding Algorithm for Multicommodity Network Flow Problems," Networks, Vol. 1, No. 4, 333-354, 1972.
20. Hirsch, W. M., and G. B. Dantzig, "The Fixed-Charge Problem," Naval Research Logistics Quarterly, Vol. 14, No. 3, 413-424, 1968.
21. Hyman, Warren, and Larry Gordon, "Commercial Airline Scheduling Technique," Transportation Research, Vol. 2, No. 1, 23-29, 1968.
22. Ishmael, P. C., Synthesis of Minimum Cost or Maximum Service Rapid Transit When Travel Time Cost is Included, Ph.D. Thesis, Arizona State University, 1972.
23. Jarvis, John J., "On the Equivalence Between the Node-Arc and Arc-Chain Formulations for the Multicommodity Maximal Flow Problem," Naval Research Logistics Quarterly, Vol. 16, No. 4, 525-529, 1969.
24. Kennington, Jeffery Lynn, Fixed-Charge Transportation Problem: A Group Theoretic Approach, Ph.D. Thesis, Georgia Institute of Technology, 1973.
25. Kresge, David T., and Paul O. Roberts, Techniques of Transport Planning, Vol. 2, Systems Analysis and Simulation Models, The Brookings Institution, Washington, D. C., 1971.

26. Kuhn, Harold W., and William J. Baumol, "An Approximate Algorithm for the Fixed-Charge Transportation Problem," Naval Research Logistics Quarterly, Vol. 9, No. 1, 1-15, 1962.
27. Levin, Amos, "Scheduling and Fleet Routing Models for Transportation Systems," Transportation Science, Vol. 5, No. 3, 232-255, 1971.
28. Lundberg, Barry D. and Robert L. Brown, Transit Operation and Scheduling, Barton-Ashman Associates, Inc., Prepared for Marquette University Urban Transportation Program Lecture Series on Mass Transit Planning and Engineering, November 1971.
29. Marks, David H., and Jon C. Liebman, Mathematical Analysis of Solid Waste Collection, Dept. of Geography and Environmental Engineering, Johns Hopkins University, February 1970.
30. Martin-Löf, Anders, "A Branch-and-Bound Algorithm for Determining the Minimal Fleet Size of a Transportation System," Transportation Science, Vol. 4, No. 2, 159-163, 1970.
31. Meyer, J. R., J. F. Kain, and M. Wohl, The Urban Transportation Problem, Harvard University Press, Cambridge, Mass., 1965.
32. Morlok, Edward K., "A Goal-Directed Transportation Planning Model," Transportation Research, Vol. 4, No. 2, 199-213, 1970.
33. Murty, Katta G., "Solving the Fixed-Charge Problem by Ranking the Extreme Points," Operations Research, Vol. 16, No. 2, 268-279, 1968.
34. Nemhauser, G. L., "Scheduling Local and Express Service," Transportation Science, Vol. 3, No. 2, 164-175, 1969.
35. Newell, G. F., "Dispatching Policies for a Transportation Route," Transportation Science, Vol. 5, No. 1, 91-105, 1971.
36. Ochoa-Rosso, Felipe, Applications of Discrete Optimization Techniques to Capital Investment and Network Synthesis Problems, Search and Choice in Transport Systems Planning, Vol. 3, Dept of Civil Engineering, Massachusetts Institute of Technology, 1968.
37. O'Connor, Arthur D., and Charles DeWald, A Sequential Deletion Algorithm for the Design of Optimal Transportation Networks, Paper presented at the 37th National ORSA Meeting, Washington, D. C., April 1970.
38. Rardin, Ronald, Private Communication (July 1973).
39. Ridley, T. M., "An Investment Policy to Reduce the Travel Time in a Transportation Network," Transportation Research, Vol. 2, No. 4, 409-424, 1968.

40. Salzborn, F. J. M., "Timetables for a Suburban Rail Transit System," Transportation Science, Vol. 3, No. 4, 297-316, 1969.
41. Schwartz, Nancy Lou, Economic Transportation Fleet Composition and Scheduling, With Special Reference to Inland Waterway Transport, Paper No. 91, Institute for Research in the Behavioral, Economic and Management Sciences, Purdue University, November 1964.
42. Scott, Allen J., "The Optimal Network Problem: Some Computational Procedures," Transportation Research, Vol. 3, No. 2, 201-210, 1969.
43. Staub, Daniel E., Simulation of an Automated Right-of-Way Mass Land Transportation System Using GASP II, M. S. Thesis, Marquette University, 1971.
44. Steinberg, David I., "The Fixed Charge Problem," Naval Research Logistics Quarterly, Vol. 17, No. 2, 217-235, 1970.
45. Swoveland, Cary, Decomposition Algorithms for the Multicommodity Distribution Problem, Working Paper No. 184, Western Management Science Institute, University of Calif., Los Angeles, October 1971.
46. Tomlin, J. A., "Minimum Cost Multicommodity Network Flows," Operations Research, Vol. 14, No. 1, 45-51, 1966.
47. U. S. Department of Commerce, Traffic Assignment Manual, Bureau of Public Roads, Washington, D. C., June 1964.
48. Univac 1100 Series FORTRAN V Programmer Reference Manual, UP-4060 Rev. 2, Sperry Rand Corporation, 1972.
49. Urban Mass Transit Administration, UMTA Transportation Planning System, NTIS No. PB 212-930, September 1972.
50. Whiting, P. D. and J. A. Hillier, "A Method for Finding the Shortest Route through a Road Network," Operational Research Quarterly, Vol. 11, Nos. 1-2, 1960.
51. Wollmer, R. D., "Multicommodity Networks With Resource Constraints: The Generalized Multicommodity Flow Problem," Networks, Vol. 1, No. 3, 245-264, 1972.
52. Yaged, B., Jr., "Minimum Cost Routing for Static Network Models," Networks, Vol. 1, No. 2, 139-172, 1972.
53. Young, Dennis R., "Scheduling a Fixed-Schedule, Common Carrier Passenger Transportation System," Transportation Science, Vol. 4, No. 3, 243-269, 1970.

54. Zangwill, Willard I., "Minimum Concave Cost Flows in Certain Networks," Management Science, Vol. 14, No. 7, 429-450, 1968.

## VITA

Gunter Pielbusch Sharp was born in Schroda, Poland, on January 25, 1943. He is the son of Dagmar Punga and Leo Julius Bette, of Reval, Estonia. He entered the United States in June 1952 and was graduated from Dreux American High School, Dreux Air Force Base, France, in June 1961.

After four years of undergraduate work at the Georgia Institute of Technology he was awarded the Degree of Bachelor of Industrial Engineering, Magna Cum Laude, in June 1965. Subsequently, on June 1966, he was awarded the Degree of Master of Science, Engineering-Economics Option, in the Department of Industrial Engineering, at Stanford University.

The next four years he was employed by engineering consulting firms, by Frank E. Basil, Inc., in Athens, Greece, and Riyadh, Saudi Arabia, and by the Austin Company in Atlanta, Georgia. His work experience included such items as feasibility studies, materials handling studies, engineering cost estimates, investment analyses, and administrative management.

In September 1970 he returned to the Georgia Institute of Technology, and he was awarded the Degree of Doctor of Philosophy in Industrial Engineering in December 1973. He is currently employed there as an Assistant Professor in the School of Industrial and Systems Engineering.

Gunter Sharp is married to the former Miss Maria Mandekos of Athens, Greece. They presently reside in Atlanta with their son, Alexander.