REPORT NO. DOT-TSC-RSPA-79-18

NETWORK AGGREGATION IN TRANSPORTATION PLANNING MODELS

Russell R. Barton
Donald W. Hearn

MATHTECH, INC.
P.O. Box 2392
Princeton NJ 08540

JUNE 1979
FINAL REPORT

Prepared for

U.S. DEPARTMENT OF TRANSPORTATION
RESEARCH AND SPECIAL PROGRAMS ADMINISTRATION
Office of Transportation Programs Bureau
Washington DC 20590

Technical Report Documentation Page

1. Report No. DOT-TSC-RSPA-79-18	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle NETWORK AGGREGATION IN TRANSPORTATION PLANNING MODELS		5. Report Date June 1979 6. Performing Organization Code
7. Author(s) Russell R. Barton and Donald W. Hearn		8. Performing Organization Report No. DOT-TSC-RSPA-79-18
9. Performing Organization Name and Address Mathtech, Inc.* P.O. Box 2392 Princeton NJ 08540		10. Work Unit No. (TRAIS) RS905/R9516 11. Contract or Grant No. DOT-TSC-1443 13. Type of Report and Period Covered Final Report 9/77 - 9/78
12. Sponsoring Agency Name and Address U.S. Department of Transportation Research and Special Programs Administration Office of Transportation Programs Bureau Washington DC 20590		14. Sponsoring Agency Code
15. Supplementary Notes *Under contract to:	U.S. Department of Transportation Research and Special Programs Administration Transportation Systems Center Cambridge MA 02142	

16. Abstract

This report contains six papers addressed at mathematical and computational aspects of an extraction aggregation model often employed in transportation planning studies. This model concerns the optimal flowing of an extracted subnetwork of a given network. Nonlinear decompositions are developed, duality theory is explored as a tool for measuring error, and heuristic methods are tested. An overview section summarizes prior work on network aggregation and the six papers of this report.

17. Key Words Aggregation, Duality, Network, Traffic Assignment, Multicommodity Flow, Decomposition, Nonlinear Programming	18. Distribution Statement DOCUMENT IS AVAILABLE TO THE PUBLIC THROUGH THE NATIONAL TECHNICAL INFORMATION SERVICE, SPRINGFIELD, VIRGINIA 22161		
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 142	22. Price

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

PREFACE

This report summarizes a study of network aggregation in transportation planning models conducted under contract DOT-TSC-1443 for the U.S. Department of Transportation. Funding for the work was provided under the Transportation Advanced Research Project (OST/TSC) with additional support provided by the Urban Mass Transportation Administration. The goals of this study have been to perform exploratory research, both theoretical and computational, on a model of extraction aggregation practices often employed in transportation planning studies. The research has yielded six independent papers which form the body of this report. An overview section is included which summarizes prior work and the six papers.

The contract under which this work was performed was guided by Edwin J. Roberts and Mike Nienhaus, TSC-213, and Robert Crosby, Office of the Secretary. Principal consultant was Harold W. Kuhn, Princeton University.

Special acknowledgment is due to Michael Florian and Sang Nguyen of the University of Montreal for sharing their computer programs and ideas.

METRIC CONVERSION FACTORS

Approximate Conversions to Metric Measures

Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
in	inches	2.5	centimeters	cm
ft	feet	30	centimeters	cm
yd	yards	0.9	meters	m
mi	miles	1.6	kilometers	km
		AREA		
in^2	square inches	6.5	square centimeters	cm^2
ft^2	square feet	0.09	square meters	m^2
yd^2	square yards	0.8	square meters	m^2
mi^2	square miles	2.6	square kilometers	km^2
	acres	0.4	hectares	ha
		MASS (weight)		
oz	ounces	28	grams	g
lb	pounds	0.45	kilograms	kg
	short tons (2000 lb)	0.9	tonnes	t
		VOLUME		
tsp	teaspoons	5	milliliters	ml
Tbsp	tablespoons	15	milliliters	ml
fl oz	fluid ounces	30	milliliters	ml
c	cups	0.24	liters	l
pt	pints	0.47	liters	l
qt	quarts	0.95	liters	l
gal	gallons	3.8	liters	l
ft^3	cubic feet	0.03	cubic meters	m^3
yd^3	cubic yards	0.76	cubic meters	m^3
		TEMPERATURE (exact)		
°F	Fahrenheit temperature	5/9 (after subtracting 32)	Celsius temperature	°C

Approximate Conversions from Metric Measures

Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
mm	millimeters	0.04	inches	in
cm	centimeters	0.4	inches	in
m	meters	3.3	feet	ft
m	meters	1.1	yards	yd
km	kilometers	0.6	miles	mi
		AREA		
cm^2	square centimeters	0.16	square inches	in^2
m^2	square meters	1.2	square yards	yd^2
km^2	square kilometers	0.4	square miles	mi^2
ha	hectares (10,000 m^2)	2.5	acres	
		MASS (weight)		
g	grams	0.035	ounces	oz
kg	kilograms	2.2	pounds	lb
t	tonnes (1000 kg)	1.1	short tons	
		VOLUME		
ml	milliliters	0.03	fluid ounces	fl oz
l	liters	2.1	pints	pt
l	liters	1.06	quarts	qt
l	liters	0.26	gallons	gal
m^3	cubic meters	35	cubic feet	ft^3
m^3	cubic meters	1.3	cubic yards	yd^3
		TEMPERATURE (exact)		
°C	Celsius temperature	9/5 (then add 32)	Fahrenheit temperature	°F

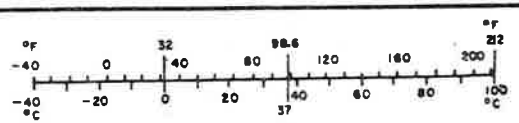


TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES (Cont.)

LIST OF TABLES

AN OVERVIEW OF MATHTECH RESEARCH IN TRANSPORTATION NETWORK AGGREGATION

Introduction

This report completes three years of research by Mathtech, Inc., on Aggregation of Transportation Networks under the auspices of Project TARP, U. S. Department of Transportation, Office of the Secretary. While its primary purpose is to report results of the final year under contract DOT-TSC-1443, this overview also contains, in the following sections, a summary of results from the first two years under contracts DOT-TSC-883 and DOT-TSC-1232.

The purpose of the research has been to bring together the network aggregation practices of transportation planners and the related theory and computational results of mathematical programming. In so doing, it was anticipated by both the sponsors and researchers that improvements in the day-to-day methodology would result. It is further hoped that other researchers will be attracted to the models and methods developed here and offer additional results. This overview concludes with specific research and development recommendations.

Research under Contract DOT-TSC-883

The first year of Mathtech research in network aggregation is documented in reference [6]. The central objective of this work was to identify, extend, and evaluate methods of network aggregation in transportation planning models. Toward this end, an annotated bibliography of over 200 articles and books was compiled from the transportation science and operations research literature. This literature survey revealed, somewhat surprisingly, very few theoretical results for, or practical examples of, transportation network aggregation. (The reference section of this overview includes some of the bibliography which has been useful in the Mathtech research.)

From the available references, however, it was possible to identify network aggregation methods as being of two types - hierarchical and intrinsic.

Hierarchical aggregation may involve either an extraction of network elements (links, nodes, etc.) or an abstraction of elements into pseudo or dummy elements which replace the original ones. The result is a smaller network which topologically resembles the original. Intrinsic aggregation, on the other hand, involves the construction of an alternative model - not necessarily a network - which will produce the results desired. For example, reference [5] describes an attempt to determine traffic equilibrium by formulating area-wide supply and demand curves. This methodology completely avoids the use of a transportation network model.

The literature survey also revealed that four main classes of network models, or problems, were likely to be of use in transportation

planning. These are (a) traffic assignment type flow problems, (b) shortest path problems, (c) maximum flow problems, and (d) Hitchcock-Koopmans type problems. As pointed out in the report, these classes are not disjoint. The importance of each of the particular problems, however, justified the four types.

For definitions we paraphrase from the report:

> Traffic Assignment Models: Problems in which link costs to a user are nondecreasing functions of flows on each link and the constraints specify origin - destination flows and (possibly) capacity bounds on some links. This category includes both the user optimized (equilibrium) and system optimized models so often used in flowing highway networks.
>
> Shortest-Path Problem: A fundamental network problem - for given constants associated with each link, determine shortest paths between specified nodes.
>
> Maximum Flow Problems: This model asks for maximum flow between a fixed origin and destination in a network with capacities on the individual links.
>
> Hitchcock-Koopmans Transportation Problems: Another basic network problem which, in its original form, asks for the flows from m origins to n destination along direct links which satisfy given demands at minimum (linear) cost.

Table 1 illustrates the possible combinations of aggregation methods and problems. Entries in the chart are representative references, exclusive of this report.

TABLE 1. AGGREGATION METHODS AND PROBLEMS

Network Aggregation Method		Traffic Assignment	Shortest Path	Maximum Flow	Hitchcock-Koopmans
Hierarchical	Extraction	7, 8, 19	25	2	
	Abstraction	3, 4, 5, 6, 17, 18			1, 6, 10, 13, 15, 16
Intrinsic		22		21	

The literature survey and above classifications formed a basis from which mathematical and computational research could be conducted. Principal results of the first year were presented in a collection of papers in the final report. Two of these papers, entitled "Bounding Aggregation Error in Network Models," and "Bounding Aggregation Error in the Equilibrium Model," have the common theme of using the duality theory of mathematical programming to bound aggregation error. Specifically, the first paper suggests that if a problem P is aggregated to some form $\overline{P}$, then solving $\overline{P}$ implies that its dual $\overline{D}$ is also solved. If solutions for P and D are constructed from $\overline{P}$ and $\overline{D}$, then the relation

$$\text{Objective value for } P \geqq \text{Optimal value for } P \geqq \text{Objective value for } D \quad (1)$$

will hold. This relation assumes that P is a minimization problem (i.e., any optimization problem), that the dual problems exist, and that there exists some mechanism, called "lifting," for obtaining disaggregated solutions from aggregated solutions. To demonstrate the idea, it is applied to the Hitchcock-Koopmans problem. Solution of a 3 source, 100 destination numerical example shows the power of the idea - a "natural" clustering of the destinations into four aggregate destinations produces an easily solved (3 x 4) problem whose solution may be lifted to a solution of the disaggregated problem. Application of (1) shows that the result is within 8 percent of optimality. Even more interestingly, it is shown that solutions of the aggregated dual ($\overline{D}$) can be used to improve the aggregation.

These results are limited to problems of the Hitchcock-Koopmans type, i.e., single commodity, linear problems where the destination (or origin) nodes are aggregated.* However, the second paper points out that the equilibrium traffic assignment problem is a convex optimization problem

---

*Concurrent and independent research by Geoffrion[15,16] and Zipkin [12] has produced equivalent results.

and that nonlinear duality theory exists as a tool for bounding aggregation error. In particular, it is shown that equilibrium models with linear link costs are quadratic programs, for which explicit duals exist. Therefore, the relation (1), is valid, at least in theory, for the equilibrium model.

Other results from [6] include:

- A fixed point algorithm for the equilibrium traffic assignment problem. This method was later named PATHFIX, coded, and tested computationally. The method is currently limited to problems where paths between origin-destination pairs are known. Its outstanding feature is the precision of the solutions obtained. Separate, self-contained documentation of the theory and results is available [14].

- Theoretical results on the effect of extracting or inserting a traversal link between two chains joining one fixed origin-destination pair in the traffic assignment model. The main result is the precise evaluation of the cost change per individual traveler brought about by the simplest non-trivial aggregation: the extraction of a single link.

- Empirical results from a computational study aimed at determining the range of error induced by aggregation of a highway network.

- Bounds and estimates for average speed per path, and estimates of computational savings from aggregation.

Research under contract DOT-TSC-1232

The statement of work for the second year of research called for a survey of aggregation practices to be conducted by interviewing transportation network users in D. O. T. and other transportation planners as might be required. The objective of this survey was to learn what aggregation practices and problems were most prevalent from among the possible combinations (Table 1). The emphasis was on obtaining detailed evidence, including actual network data, of a real world study which could be examined and simulated.

As an outgrowth of this experience it was anticipated that specializations of the mathematical models and algorithms could be developed which would improve aggregation practices.

Part I* of the final report [12] documents the results of this survey which consisted of 20 interviews of D.O.T. personnel, representing all transportation modes, as well as several transportation planners and consultants in the Washington, D. C. area. It contains detailed descriptions of four aggregation practices encountered at UMTA, FRA, the Washington Transportation Planning Board, and at JHK Associates, a consulting firm. These four, as well as one dissertation from the literature [18] on a study in Phoenix, Arizona, were shown in the report to fit a common pattern of extraction aggregation. With respect to Figure 1, the FRA model involved a shortest path problem; the other four were traffic assignment problems.

The pattern of extraction aggregation employed is simple. Given a large network and matrix of flow demands (trip table in the traffic assignment problem), it involves the extraction of a subnetwork containing links of particular interest, the transfer of some portion of the demand matrix to the nodes of the subnetwork, and a subsequent flowing of the subnetwork to satisfy the transferred demand. In each case documented, the specifics of the pattern varied greatly; e.g., the subnetwork could be the entire original network or just a few links of the original.

Table 2, adapted from the final report, shows the pattern of extraction aggregation and how the various studies fit this pattern. (For details, the reader should consult the report.)

---

*Part II [14] is documentation of the PATHFIX algorithm.

TABLE 2. EXTRACTION AGGREGATION PRACTICES

Extraction Aggregation Steps	JHK Shirley Study	Dial's Origin Aggregation (UMTA)	Wilson's Load Node (Phoenix)	Mann's Long Trip Loop (Washington TPB)	Mann's Short Trip Loop (Washington TPB)	FRA Circuity Study
1. Given Network and Trip Table	Virginia and D.C. Zone Network	8,000 Node Experimental Network	Phoenix Network	Metropolitan Washington Zone Network (including District centroids)	Same	FRA Network
2. Identify Links	Shirley Highway	Arbitrary Subnetwork	All Links	All Links	Same	All Links
3. Extract Subnetwork	Major DC and Va. arteries	Entire Network	Entire Network	Entire Network	Same	Entire Network (Segments)
4. Select Pseudo Centroids	Access & Egress Points on the Arteries	District Centers and Super Centers	Intersection Nodes Surrounding Zones	District Centroids	Zone Centroids	Nodes with Rail Yards
5. Transfer Trip Table	Manually	Nearest Center	Inversely Proportional to Straight Line Distance	Transfer is made Dynamically in Step 6	Trip Table is Fixed	Near Yard
6. Flow Subnetwork	All-or-Nothing (Manual)	Equilibrium Assignment	Capacity Restraint Assignment	Incremental Assignment of Long Trips	Incremental Assignment of Short Trips	Shortest Path

The terms "trip table" and "centroid" are traffic assignment terms; more generally, these might be replaced with "demand matrix" and "origin/destination nodes". It is assumed, of course, that the links identified in step 2 are a subset of the extracted subnetwork, and that the pseudo-centroids are nodes of the subnetwork. Step 6, flowing the subnetwork, is accomplished by some algorithm appropriate for the particular problem. It is applied to the subnetwork and transferred trip table in lieu of the original network and trip table. The resulting computational savings may be modest or large. The Phoenix aggregation resulted in computer time savings of 40 percent to 60 percent, while the JHK aggregation in a study of Interstate 95 (Shirley Highway), south of Washington, involved a subnetwork so small that potential savings in computation effort were estimated at over 99 percent.

In Figure 1, the MEASURE step has been added to illustrate the need for feedback in the process. None of the studies surveyed had such a step; instead, the flows obtained in Step 6 were used in the planning process.

The second phase of work under the contract was a simulation of the Shirley study. The objectives were to (a) gain insight into actual aggregation practices, (b) determine what steps of the extraction aggregation model seemed the most critical, (c) test the idea of duality theory as a tool for the MEASURE step, and (d) determine whether estimates of computational savings (which are derived in [12]) are accurate.

In this simulation, a given network of over 9000 links formed the basis from which a subnetwork of 483 links was extracted. The transfer of the demands in the original 700 x 700 trip table to the subnetwork became the object of several heuristic methods, because, not surprisingly,

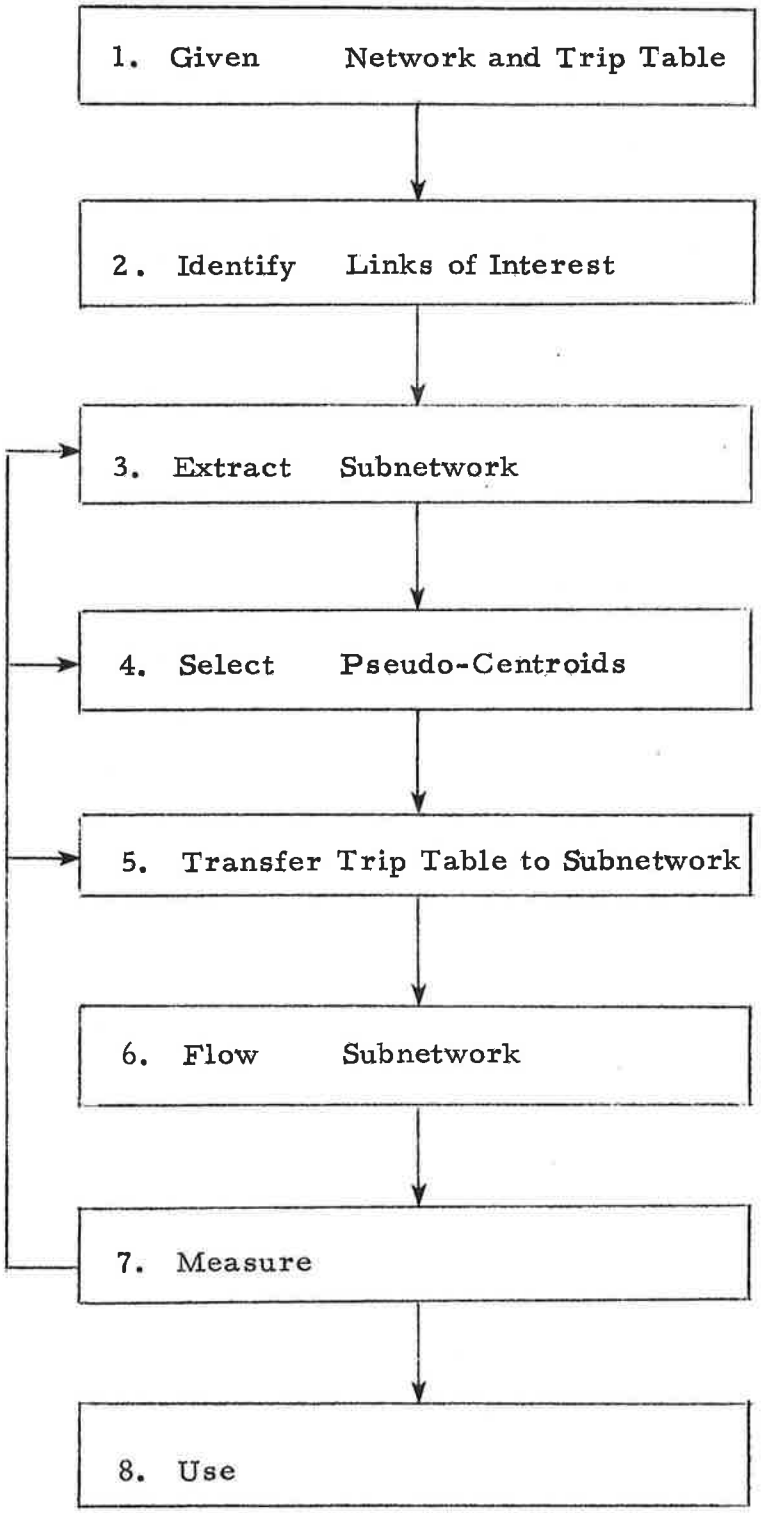


FIGURE 1. EXTRACTION AGGREGATION WITH FEEDBACK

the model showed high sensitivity to this step. Other conclusions of the computational study were that the formulas for predicting savings were very accurate, and that the heuristics employed to construct primal and dual solutions (see (1)) yielded very weak bounds.

The final chapters of Part I of the report address the extraction aggregation model from the viewpoint of mathematical programming. Decomposition theory is cited as a primary tool for making Steps 5 to 7 of the model precise and guiding further heuristic development. Theoretical proofs were given to show that various known error measures for the traffic assignment model are equivalent to bounding error by nonlinear duality theory.

Current Research Summary (Contract DOT-TSC-1443)

Research for the current year has focused on the extraction aggregation model. The emphasis has been on obtaining new theoretical results and algorithms, testing ideas computationally, and the design and testing of an improved heuristic. Particular stress has been placed on defining the extent to which nonlinear decomposition theory applies to steps 5 to 7 of the model (see Figure 1), and the extent to which duality theory is useful as a tool in step 7.

This research has produced six independent papers which form the body of this report. Each adopts a style and technical level deemed appropriate for the particular problem considered. For numerical examples and computational experiments, the equilibrium traffic assignment problem is often used. However, virtually every result applies to any convex multicommodity flow problem. The remainder of this section

is a nontechnical summary of the papers.

Paper 1. Transfer Decomposition

This paper is an expository development of a new method for network decomposition which is particularly well-suited for problems where the subnetwork is a set of major links. (The Shirley highway study is one example.) Its name derives from the fact that the communication between master and subproblem is via the subnetwork trip table of the TRANSFER step. In other words, the master problem constructs different (trial) trip tables which are used by the subproblem to flow the subnetwork. The process is a convergent one which, in theory, will ultimately produce a correct subnetwork trip table. By "correct," we mean the trip table will, upon solution of the subproblem, produce the same subnetwork flows obtainable if the full problem is solved.

There is, of course, communication from the subproblem to the master problem after each trial flowing of the subnetwork. With respect to Figure 1, the steps 5 to 7 are repeated and the inner feedback loop is present.

Transfer Decomposition has several appealing features not present in other decompositions. They include:

a) No preliminary decomposition by commodity (origin) is required.

b) Both master and subproblems are network flow problems. Therefore existing computer codes can easily be employed to implement the method.

c) Computational experiments with variations of the basic method suggest that computer time and space requirements can be competitive with existing methods for solving the full problem.

The paper includes a numerical example, heuristic and precise variations of the method, and the results of computational experiments.

Paper 2. Subset Decomposition

This paper presents an interesting alternative decomposition which is appealing for particular problems. The method of Wilson [18], for example, involved a heuristic removal of centroid connectors in a highway network and a transfer of the trip table to actual intersections of the network. Viewed as a decomposition, all of the deleted links (centroid connectors) have similar characteristics, e.g., constant travel times. Subset Decomposition exploits such special structure.

In this method, the master problem is shown to be minimization of a convex function subject to just one linear constraint, and a simple method for maintaining feasibility is given. The subproblems for each subnetwork (there may be many) are convex flow problems of the same form as the original problem. For the traffic assignment problem, a necessary modification of the Frank-Wolfe method, which is commonly used for such problems, is given.

Computational testing of the method is proposed for future research.

Paper 3. Minimizing the Gap Function in Certain Convex Programs

This paper is related to the bounding formula (1) on page 4. It presents a new idea for direct minimization of the duality gap; i.e., the difference between the upper and lower bounds of (1). This gap is expressed as a function of primal variables only (flow variables in a network problem), and methods are presented for minimizing it to achieve the optimal value of zero.

The idea has intuitive appeal in the equilibrium traffic assignment problem because the duality gap is (as shown in [12]) the total time for all trips in the network minus the time if all trips are via shortest paths. In the usual algorithms for the equilibrium problem, this quantity approaches zero indirectly because the algorithm is designed to minimize a different objective function.

Results in the paper include sufficient conditions for the duality gap to be a convex function. Unfortunately, however, these conditions are not normally met for equilibrium problems unless the link travel times are linear. The computational results reported are mixed - in some cases direct minimization of the gap produced solutions very quickly, but in others convergence is slow. It is our feeling that this remains an interesting research problem, but for the short term, the most useful result of this investigation is the correction steps for improving error bounds (see Paper 4).

Paper 4. The MEASURE Step and Nonlinear Duality

As already mentioned, theoretical results in [12] prove that all methods commonly used for bounding error in traffic assignment problems are equivalent to the use of nonlinear duality theory (1). Further, computational results have verified additional analysis showing that the bounds are often very weak.

Paper 4 shows some inherent limitations of duality theory for bounding convex cost flow problems. Specifically, it is shown that flows for the entire network are required in order to evaluate the dual objective value. This requirement is not present in linear problems such as the Hitchcock-Koopman problem, and means that lifting of aggregated dual

solutions requires not just dual variables, but primal (flow) variables as well. An important implication is that nontrivial bounds require the solution of at least one linear program. This discussion includes both graphical and network interpretations of the dual problem.

Paper 4 also contains the results of some computational experiments in which error bounds are generated from infeasible network flows. These results suggest that bounds in the range of 10 percent to 50 percent can often be obtained from infeasible flows (e.g., all flow variables equal to zero). An interesting, and still experimental, idea of "correction steps" which can improve these bounds is briefly explored. It is pointed out that such correction steps could easily be added to existing D.O.T. software such as UROAD, but that the cost of the improved bounds may not be attractive.

Paper 5. Equivalent Flows for Extracted Subnetworks

This paper explores the theoretical relationships between flows obtained from flowing a given network and those obtained from flowing an extracted subnetwork. Questions such as the existence of a subnetwork trip table which will replicate optimal flows on the subnetwork are addressed. One interesting result investigated is the theorem of Nguyen's [17] which gives a formulation of the equilibrium assignment problem that does not require a trip table. More precisely, if origin to destination path times are known, Nguyen's formulation will replicate flows in the network equivalent to those one would obtain if the trip table were known. However, several examples show that the unpredictable nature of the subnetwork trip table limits the usefulness of the theorem in step 5 of the extraction aggregation model.

Paper 6. Experiments with a TRANSFER and FLOW Heuristic

This paper summarizes some experiments with a heuristic method suggested in the prior research [12]. From this prior work, it seemed that, for the simulated Shirley study (see above), the flows of the subnetwork were highly sensitive to the number of trips transferred, but not very sensitive to other parameters of the extraction aggregation model. It was conjectured that any intelligently designed transfer of the correct number of trips to the subnetwork would produce good flows on the subnetwork links.

The specifics of the heuristic (known as M1) involve connecting, via pseudo links, the original 700 centroids to the 483 link subnetwork of the Shirley study. The pseudo links are an example of abstraction aggregation because they are constructed from deleted links of the original network in an intuitively appealing way. The original (700 x 700) trip table is scaled by a fraction which is an estimate of the proportion of trips using the extracted subnetwork. It is noted in the paper that M1 is a generalization of the procedure which must be implemented in the design of a computer network model of urban highways because the pseudo links are generalizations of centroid connectors.

Computational experiments with M1 were limited to two choices for percentage of trips transferred and three choices for number of pseudo-links per centroid. The results for aggregate measures, such as average error in links flows, etc., are within 15 percent of the correct values obtained from a benchmark flowing of the entire network.

Summary and Recommendations

Our research in transportation network aggregation has been of an exploratory nature, in keeping with the philosophy of the TARP Project. Throughout, emphasis has been on development of new methodology relevant to the various possible combinations of aggregations and models (Table 1). We feel that the inner loop (Figure 1) of the extraction aggregation model has been thoroughly studied as an application of decomposition theory. Sufficient computational testing has been done to demonstrate the potential of the proposed methods. Other types of aggregation, especially abstraction, remain open questions except in very special cases. Primary accomplishments of our research are given below:

Development of an Aggregation method for the Hitchcock-Koopmans problem [6].

Development and testing of the PATHFIX Algorithm [14].

Survey of Extraction Aggregation Methods [12].

Development and testing of TRANSFER-FLOW-MEASURE decompositions [Papers 1 and 2].

Exploration of Nonlinear Duality for Bounding Error in Convex Flow Problems [12, Papers 3 and 4).

Development of a Large-Scale Data base for testing Extraction Aggregation Methods [12].

Development and Testing of Heuristic Methods for Extraction Aggregation [12, Paper 6].

Further research and development ideas relevant to transportation networks which we feel important are:

a. Resolution of Decomposition

Virtually every result obtained in our research and the research of others [7, 16] has either employed decomposition theory,

or may be related to decomposition theory [24]. It is our belief that the theory will continually be the basis for large-scale network methodology development. The TARP project has produced various decompositions in coded form, and a variety of networks of all sizes. Therefore, we recommend that a thorough testing of the methods be undertaken with an emphasis on the derivation of prediction formulas which can be used to guide the choice of method to employ on a network of given parameters. (It is worth noting that this study would include the well-known Frank-Wolfe method because it is also a decomposition.)

b. Heuristic Development

Although decomposition is the theoretical key to solving large network problems precisely, all such methods have two serious drawbacks--they converge slowly and they flow the entire network. Many transportation studies require only flowing a subnetwork. From the experiments with heuristics in our research, we feel that it is possible to develop sharp versions which will serve many users. If the ideas of the Papers 1, 3, and 6 in this report are employed, it is reasonable to expect that such heuristics would produce solutions provably within 10 to 15 percent of optimality for a fraction of the cost of flowing a full network to the same percent relative error.

c. Minimax Code for Networks

A problem which repeatedly occurs in the solving of nonlinear network flow problems is that of minimizing the maximum of a collection of linear functions subject to network constraints. Two examples are in bounding error and in solving decomposition master problems. Although this is an example of the convex cost

network flow problem, very little attention has been given to the possibility of an algorithm exploiting the special structure of this problem.

d. Testing of Subset Decomposition

The theory of this method is developed in Paper 2. It has particular appeal for problems such as Wilson's [18] which has, as one subset, arcs with linear costs. The potential of this method needs to be explored computationally.

e. Extraction Aggregation Model

Steps 5 to 7 of this model have been thoroughly explored and the relationship to nonlinear decomposition methods illuminated. No work has been done on the interesting possibilities suggested by the outer feedback loops in Figure 1. Specifically, we have not developed mathematical programming techniques which would guide the variation of either pseudo-centroids or the choice of subnetwork links. Almost surely, methods devised in this area would be combinatorial.

REFERENCES

1. Balas, E., "Solution of Large-Scale Transportation Problems Through Aggregation," Operations Research, Vol. 13, No. 1, 1965.

2. Chan, Y., Follansbee, K.G., Manheim, M. and Mumford, J.R., "Aggregation in Transport Networks: An Application of Hierarchical Structure," Department of Civil Engineering Research Report R68-47, M.I.T., July 1968.

3. Chan, Y., "Optimal Travel Time Reduction in a Transport Network: An Application of Network Aggregation and Branch Bound Techniques," Department of Civil Engineering Research Report R69-39, M.I.T., July 1969.

4. Chan, Y., "A Method to Simplify Network Representation in Transportation Planning," Transportation Research, Vol. 10, 1976.

5. Chan, Y., et al., "Review and Compilation of Demand Forecasting Experiences: An Aggregation of Estimation Procedures," Technical Report, Pennsylvania State University, June 1977.

6. Cullen, D.E., Kuhn, H.W. and Frank, M., "Aggregation in Network Models for Transportation Planning," Mathtech Final Report, DOT-TSC-RSPD-78-7, February 1978.

7. Dantzig, G.B., Maier, S.F., Lansdowne, Z.F., "The Application of Decomposition to Transportation Network Analysis," Final Report DOT-TSC-OST-76-26, October 1976.

8. Geoffrion, A.M., "Using an Auxiliary Model to Guide the Design of a Very Large Logistic Planning Model," Management Science Study Center Discussion Paper No. 60, U.C.L.A., April 1976.

9. Geoffrion, A.M., "A Priori Error Bounds for Procurement Commodity Aggregation in Logistics Planning Models," Management Science Study Center Working Paper No. 251, U.C.L.A., June 1976.

10. Geoffrion, A.M., "Customer Aggregation in Distribution Modeling," Western Management Science Institute Working Paper No. 259, U.C.L.A., October 1976.

11. Gomory, R.E. and Hu, T.C., "Multi-Terminal Network Flows," J. SIAM, Vol. 9, No. 4, p. 551-70, December 1961.

12. Hearn, D.W., "Network Aggregation in Transportation Planning, Part I," Mathtech Final Report, DOT-TSC-RSPD-78-8, I, April 1978.

13. Hu, T.C., "A Decomposition Algorithm for Shortest Paths in a Network," J. ORSA, Vol. 16, No. 1, p. 91-102, January 1968.

14. Kuhn, H.W., "Network Aggregation in Transportation Planning, Part II," Mathtech Final Report, DOT-TSC-RSPD-78-8, II, April 1968.

15. Long, C.D. and Stover, V.G., "The Effect of Network Detail on Traffic Assignment Results," Texas Transportation Institute Research Report No. 60-11, Texas A&M University, August 1967.

16. Magnanti, T.L. and Simpson, R.W., "Transportation Network Analysis and Decomposition Methods," Final Report DOT-TSC-RSPD-78-6, Massachusetts Institute of Technology, March, 1978.

17. Nguyen, S., "Estimating an OD Matrix from Network Data: A Network Equilibrium Approach," Centre de recherche sur les transports, Report #60, University of Montreal, February 1977.

18. Wilson, E.N., "The Load Node Concept of Traffic Assignment for Urban Areas," Ph.D. Dissertaion, Arizona State University, September 1972.

19. Wollmer, R., "Removing Arcs from a Network," Operations Research, Vol. 12, 1964.

20. Woods, D.L. and Stover, V.G., "The Effect of Zone Size on Traffic Assignment," Texas Transportation Institute Research Report 60-8, Texas A&M University, August 1967.

21. Zipkin, P.H., "A Priori Bounds for Aggregated Linear Programs with Fixed-Weight Disaggregation," School of Organization and Management Technical Report #86, Yale University, 1976.

22. Zipkin, P.H., "Error Bounds for Aggregated Convex Transportation Problems," Graduate School of Business Research Paper 93A, Columbia University, April 1978.

23. Zipkin, P.H., "Bounds on the Effect of Aggregating Variables in Linear Programs," Graduate School of Business Research Paper 211, Columbia University, October 1977.

24. Zipkin, P.H., "Aggregation in Linear Programming," Ph.D. Thesis, Yale, 1977.

25. Zipkin, P.H., "Bounds for Aggregating Nodes in Network Problems," Graduate School of Business Research Paper 79A, Columbia University, March 1978.

1. TRANSFER DECOMPOSITION

Donald W. Hearn

Introduction

The Extraction Aggregation Model [2]* has at its core the problem of determining a demand matrix for a given extracted subnetwork (TRANSFER), flowing the subnetwork (FLOW), and comparison of the solution obtained with the "correct" solution which one would obtain with no extraction (MEASURE). In this paper we show that these steps may be viewed in the context of a nonlinear decomposition of the entire network which we call Transfer Decomposition because it illuminates the TRANSFER step of the Extraction Aggregation Model. As a consequence, of course, flows on all links are obtained and a lower bound is generated. Thus Transfer Decomposition accomplishes all three of the steps. As will be seen, it is a very flexible algorithm which can be varied to suit alternative needs such as different choices of subnetwork and of subnetwork demand matrix. The final section contains specific suggestions for the development of production computer codes.

TRANSFER Decomposition

In keeping with the expository style of this paper, Transfer Decomposition will be explained for the example of Figure 2. We assume the problem to be a multicommodity convex cost flow problem (such as the traffic assignment problem). The arcs of the network are partitioned

---

*See the overview section of this report.

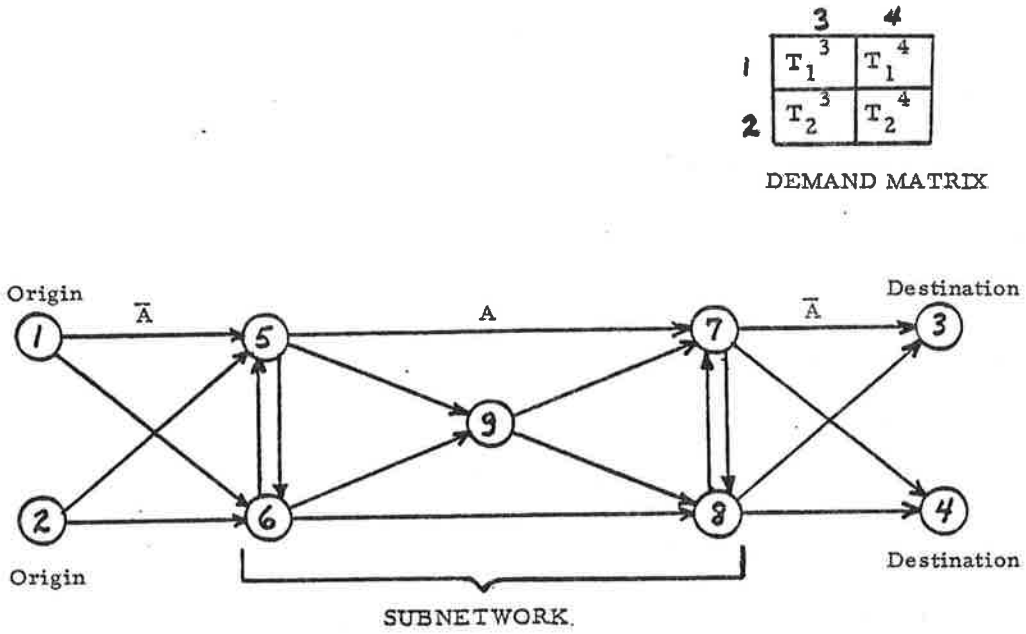


FIGURE 2. EXAMPLE PROBLEM

into sets $\overline{A}$ and A where subnetwork arcs are in A and all others are in $\overline{A}$. Components of A and $\overline{A}$ are of the form ij where i and j are nodes of the network. Thus,

$$A = \{56,\ 65,\ 57,\ 59,\ 68,\ 69,\ 97,\ 98,\ 78,\ 87\},$$

and

$$\overline{A} = \{15,\ 16,\ 25,\ 26,\ 73,\ 74,\ 83,\ 84\}.$$

Individual commodity flows, representing all combinations of origin to destination flows, are summed over commodity into total arc flows y_{ij} for $ij \in \overline{A}$ and x_{ij} for $ij \in A$. There is a convex function f_{ij} for each arc which represents the "cost" (or "time") of traversing the arc as a function of total flow. Therefore the problem is

$$\text{Minimize} \sum_{ij \in A} f_{ij}(y_{ij}) + \sum_{ij \in \overline{A}} f_{ij}(x_{ij}), \qquad (P)$$

subject to constraints which conserve flow by commodity with respect to the given demand matrix (known as the trip table in the traffic assignment problem). Note that the subnetwork of the example problem does not contain either origin or destination points (centroids). This assumption simplifies the present development, but also it reflects the conditions which often occur in extraction aggregation practices (see [2], especially the Shirley Highway study and Wilson's Load Node scheme). In a more general treatment, this assumption may be relaxed.

Assume, now, that the flows y_{ij}, $ij \in \overline{A}$ are fixed at values feasible to the flow conservation conditions, and let the x_{ij}, $ij \in A$ be variable. Then there exists * an induced demand matrix (trip table) for the subnetwork, and, owing to the additive nature of (P), the optimal set of x_{ij}

---

*The existence of an induced trip table is intuitively clear. For a rigorous and detailed treatment see Paper 5 of this report.

for the fixed y_{ij} may be obtained by solving

$$\text{Minimize} \sum_{ij \epsilon A} f_{ij}(x_{ij}) \qquad \text{(S)}$$

subject to conservation of flow constraints on the subnetwork with respect to the demand matrix induced by the fixed y_{ij}. The demand matrix of the subproblem (S) does not need to retain the original commodity identification of the problem (P). For example, flows of a commodity from node 1 to node 4 through nodes 5, 9, and 8 in problem (P) are merely represented in problem (S) as flows from 5 to 8, combined with all other of the original commodities flowing from node 5 to 8, such as flow from 2 to 3 via nodes 5, 7 and 8.

Based on this observation, we reformulate problem (P) as two network flow problems: an outer problem and an inner problem (see Figure 3). The inner network flow problem is (S) which has optimal value h(z) where

$$h(z) = h\left(z_{57}, z_{58}, z_{67}, z_{68}\right) = \min \sum_{ij \epsilon A} f_{ij}(x_{ij}) ,$$

subject to conservation of flow on the subnetwork where $z = \left(z_{57}, z_{58}, z_{69}, z_{68}\right)$ is the vector of components of the subnetwork demand matrix induced by the y_{ij}, $ij \epsilon \overline{A}$.

The function h(z) may be thought of as the minimum "cost" or "time" to traverse the extracted subnetwork given the subnetwork demand matrix.

The outer (or master) problem is

$$\text{Minimize} \sum_{ij \epsilon \overline{A}} f_{ij}(y_{ij}) + h(z) , \qquad \text{(M)}$$

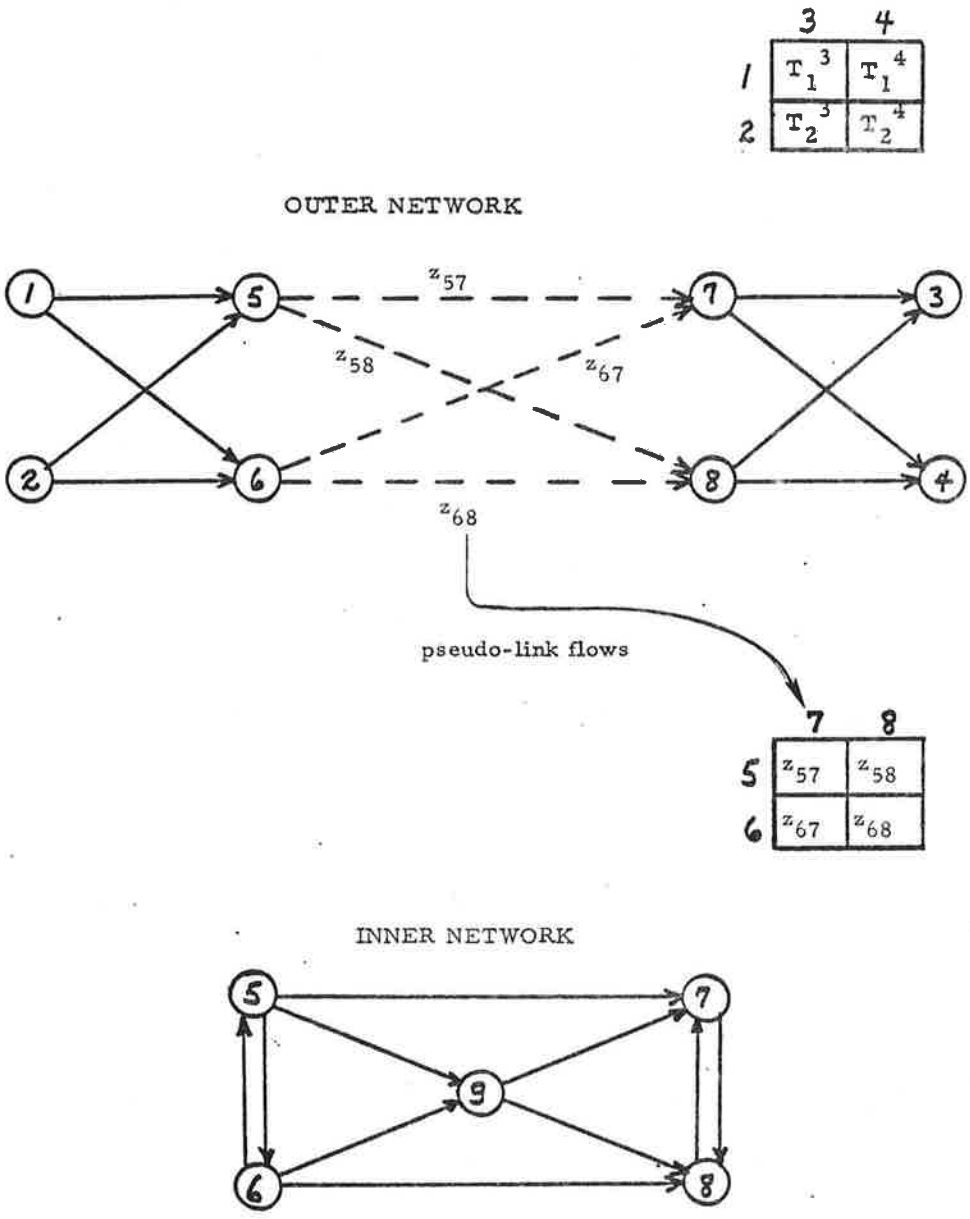


FIGURE 3. OUTER AND INNER NETWORKS OF TRANSFER DECOMPOSITION

subject to flow conservation for the outer network of Figure 3. Note in Figure 3 that the induced demand matrix for the inner problem is obtained directly from flows on "pseudo-links" of the outer network. This provides communication from the problem (M) to the subproblem (S). To complete the loop, the subproblem (S) supplies the master problem pseudo-link "times" or "costs" which are obtained from the optimal multipliers of the subproblem.

Given these observations, we state without proof the following properties of h, and the relationships of (M), (S) and (P) upon which Transfer Decomposition is based. Proofs follow, for the most part, from known results in decomposition theory as expounded by Lasdon [3] and Geoffrion [1].

Property 1. h(z) is a convex function for any z feasible to problem (M).

Property 2. For given z feasible to (M), the set of optimal multipliers of (S) is the set of derivatives of h. h is differentiable if and only if the set of multipliers is unique.

Property 3. The optimal solution value of problem (M) is equal to the solution value of problem (P). If the solution of problem (M) is within ϵ_1 of optimality and the corresponding solution of problem (S) is within ϵ_2 of optimality then the combined solution is within $\epsilon_1 + \epsilon_2$ of optimality for problem (P).

Property 1 guarantees that the objective of problem (M) has just one optimal solution value. Property 3 insures that solving (M) and the subproblem (S) does yield a solution to (P) to within measurable tolerance.

It is clear that both problems (M) and (S) are convex multicommodity flow problems, just as (P) is. Therefore any algorithm which will solve (P) will also solve both (M) and (S). Only one qualification to this statement must be made. The function h is not differentiable over its domain and therefore an algorithm which required differentiability of the objective function for its convergence proof might not converge when applied to (M). However, h can be shown to be differentiable "almost everywhere" in its domain, and at nondiffentiable points, subderivatives of h always exist. Therefore, one may heuristically employ an algorithm such as Frank-Wolfe to solve (M). Our computational experiments bear evidence that this works without convergence difficulties. Of course, no such problem arises for (S). The objective of (S) will be differentiable as a function of the x_{ij} variables if the f_{ij} are differentiable.

When (P) is a traffic assignment problem, both (M) and (S) are traffic assignment problems. Therefore, the well known Frank-Wolfe method may be applied to both. We summarize by stating the Transfer Decomposition algorithm for the problem of Figure 2, assuming given tolerances ϵ_1 and ϵ_2.

TRANSFER DECOMPOSITION ALGORITHM

Step 0. Construct a feasible flow for problem (M), e.g., determine the all-or-nothing assignment with zero times on the pseudo-links.

Step 1. Construct the trip table for problem (S) from the pseudo-link volumes.

Step 2. Solve problem (S) to within some tolerance ϵ_2. Travel times of problem (S) are returned to the master problem as pseudo-link travel times.

Step 3. Given the new travel times determine a lower bound for problem (M) by solving the usual Frank-Wolfe subproblem. If problem (M) is within ϵ_1 of optimality, terminate. Otherwise load the outer network trips along shortest paths determined by the new travel times.

Step 4. Perform a line search minimization between the new outer network flows and the prior flows. (Note that each point of the line search requires solving problem (S)). The final iteration of the line search produces new link volumes for the outer network. Go to 1.

Of course, particular care must be exercised in step 4, the line search. Since each iteration requires a resolving of problem (S), the method is only practical if just a few points on the line need be evaluated.

A heuristic variation performs the line search, not on the objective of (M), but on a function $\ell(y, z)$ which is defined to be the maximum of all tangent planes generated in prior iterations. The information required to evaluate $\ell(y, z)$ is easily obtained by employing Property 2. It must be stored, so the heuristic requires more core, but the trade-off with time required to solve the subproblems can be dramatic.

This heuristic does not produce monotonically decreasing objective values for (M). However, even if the objective increases, the method continues to the next iterate defined by minimizing $\ell(y, z)$. Since $\ell(y, z)$ is defined by the maximum of tangent planes and the line search is a minimization, we

refer to the method as a Minimax Heuristic. Below are the modifications required to the Transfer Decomposition Algorithm.

MINIMAX HEURISTIC

Step 0. No change

Step 1. No change

Step 2. No change

Step 3. (Add to the previous Step 3.) Construct a tangent plane (from the multipliers) for the objective of (M) and save.

Step 4. Perform a line search on the function $\ell(y, z)$, the maximum of all tangent planes. The final iteration of the line seach produces new link volumes for the outer network. Retain as the current objective of (M) the minimum of the previous value and the value produced by the line search. Go to 1.

An Example

It is instructive to consider a numerical example of the transfer decomposition algorithm of the previous section. In Figure 4 traffic assignment data for the network already introduced is given. The "volume delay" formula, typical of FHWA and UMTA models, represents the time to traverse each link as a function of the link volume, or flow. The constant T_0 is the uncongested travel time for the link, and the "Capacity" term is capacity only in a penalty sense. (Both T_0 and Capacity values were randomly generated). The optimal objective value of 1854 is for the user equilibrium model of traffic flow. The fixed trip table requires that a total of 100 trips traverse the network from origins 1 and 2 to destinations 3 and 4. The

number beside each link represents an optimal solution in terms of total link flow. These values were obtained by applying the Frank-Wolfe algorithm to the full problem.

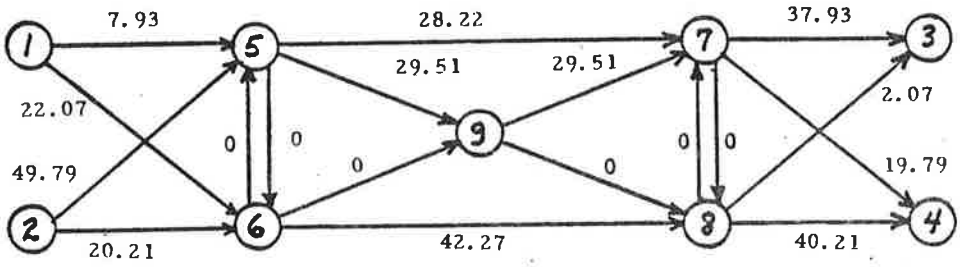
The initial iterations of Transfer Decomposition on this problem are illustrated in Figure 5. The subnetwork trip tables generated as flows on the pseudo-links of problem (M) in steps 0 and 4 are in the second column. In effect, these are extreme points of the domain of (M) with respect to the variables (z_{57}, z_{58}, z_{67}, z_{68}). Step 4 of the algorithm, the line search, transfers a convex combination of all past trip tables to the subproblem. These are shown in the third column. Step 2 of the algorithm solves the subnetwork traffic assignment problem with the trip table of column 3 and produces the multipliers (travel times) shown in the fourth column. Successive values of the objective for (M) are shown in the final column.

In theory, an infinite number of iterations will produce the optimal solution values for the subnetwork trip table and the multipliers shown in the final row with objective value 1836. Note, however, that in only three iterations the subnetwork trip table, a convex combination of the three trip tables of column two, produces a solution within 8 percent of the optimal value. Equally impressive is the fact that the travel times (multipliers) for the traversal of the subnetwork are quite close to their optimal values, differing by percentages which range from less than 3 percent to at most 12 percent.

	3	4
1	10	20
2	30	40

TRIP TABLE

OPTIMAL OBJECTIVE VALUE = 1836



$$\text{VOLUME DELAY} = T_0\left(1 + 0.15\left(\frac{\text{Flow}}{\text{Capacity}}\right)^4\right)$$

Link	T_0	Capacity
1-5	5	10
1-6	6	16
2-5	3	35
2-6	9	18
5-6	9	20
5-7	2	11
5-9	8	26
6-5	4	11
6-8	6	33
6-9	7	32
7-3	3	25
7-4	6	24
7-8	9	19
8-3	8	39
8-4	6	43
8-7	4	36
9-7	4	26
9-8	8	30

FIGURE 4. RANDOM PROBLEM

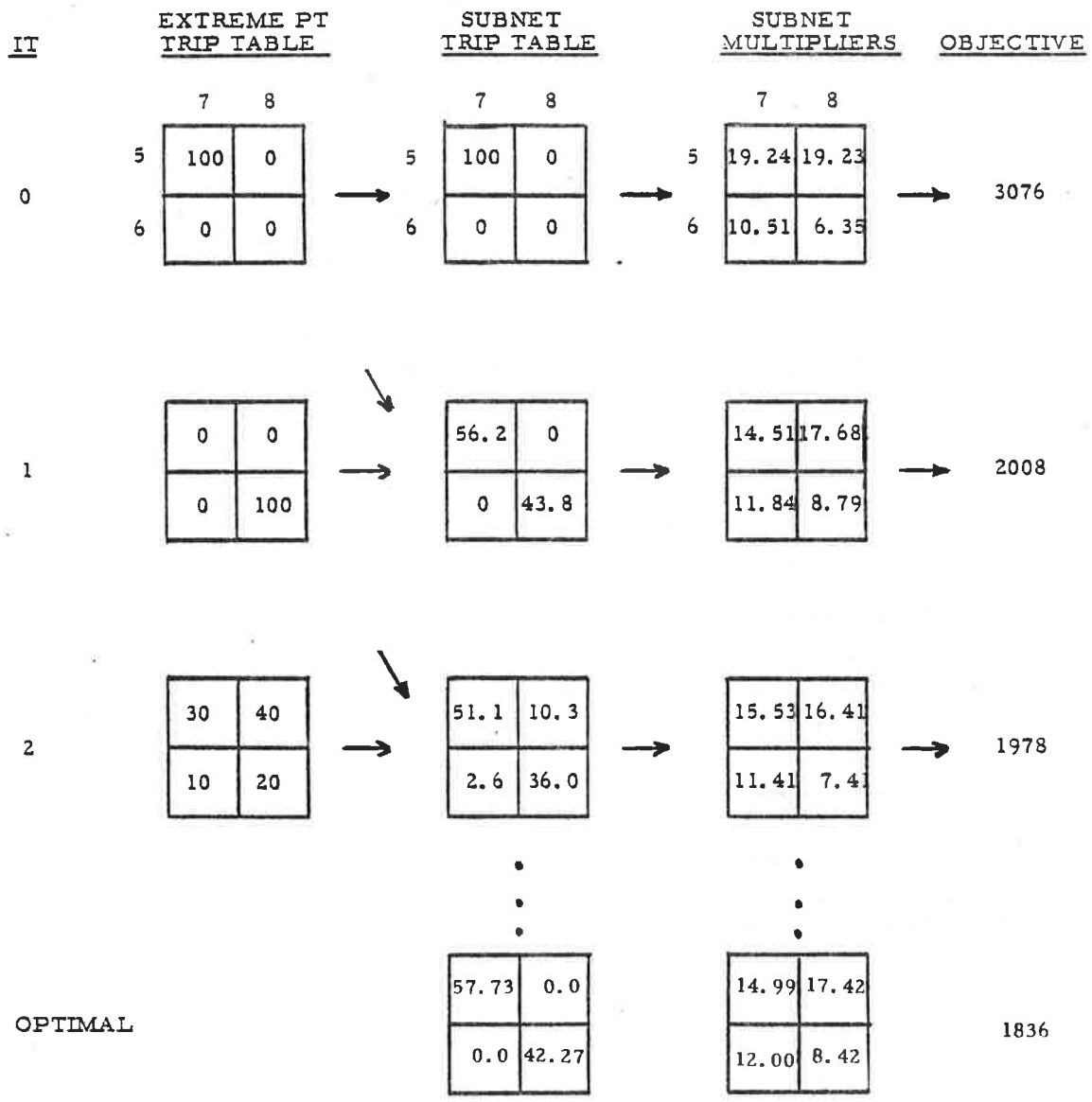


FIGURE 5. ITERATIONS OF TRANSFER DECOMPOSITION ON THE RANDOM PROBLEM

Alternative Transfer Decompositions

The subnetwork of the decomposition in Figure 2 is a "natural" one in that the network divides into disjoint segments and the pseudo-links have an easily visualized interpretation. Other choices of a subnetwork are possible, however. Consider Figure 6, where the same network as before has been decomposed so that the links (59, 97, 74) comprise the subnetwork. One advantage of this decomposition is that the subnetwork is a tree and therefore the execution of step 2, solution of subproblem (S), is trivial. The disadvantage is that the outer network is three links larger than the original network. Note, however, that the subnetwork trip table is not unique for this choice of decomposition. If links associated with z_{54}, and z_{94} were removed from the outer network, the subnetwork could be as shown and the trip table would only contain z_{59}, z_{97}, and z_{74}. The important difference is that the pseudo-links of the outer network always have "costs" which sum to h(z), even if the pseudo-link replaces an original link.

This example shows that Transfer Decomposition is quite general and can be varied to suit different needs. Computational results on the decomposition of Figure 6 are in the next section.

Computation Experiments

This section reports computation experiments on three small equilibrium assignment problems, all of which have the network topology of the example problem in Figure 2. The data for the three problems is given in Figure 4 and Figure 7. Problems E and H contain nonrandom data designed to make them easy and hard, respectively, in terms of the equilibrium model criteria. Thus the alternative origin to destination

TABLE 3. COMPUTATION RESULTS FOR RANDOM PROBLEM

Item	Frank-Wolfe			Transfer Decomposition (Fig. 2)				Minimax Heuristic				Alt. Transfer Decomposition (Fig. 5)			
	CPU Secs	Obj.	Bound	CPU Secs	Obj.	Bound	Number Subnet Calls	CPU Secs	Obj.	Bound	Number Subnet Calls	CPU Secs	Obj.	Bound	Number Subnet Calls
0	.04	42931	0	.23	3077	0	1	.22	3076	0	1	.05	10829	0	1
1	.11	2561	0	.66	2008	0	8	.33	2694	0	3	.12	1927	0	21
2	.17	2009	0	1.21	1973	1493	15	.43	2018	0	4	.20	1914	1561	41
3	.24	1982	1458	1.58	1952	1788	22	.54	2156*	1400	5	.28	1902	1640	61
4	.30	1932	1458	2.17	1911	1788	29	.66	2121	1400	6	.36	1899	1715	81
5	.36	1926	1636	2.67	1905	1788	36	.77	1964	1400	7	.43	1894	1753	101
6	.42	1916	1714	3.13	1896	1845+	43	.88	1981*	1859+	8	.51	1891	1753	121
7	.48	1908	1714	3.64	1882	1788	50	.99	1922	1610	9	.59	1889	1791	141
8	.55	1904	1746	4.11	1876	1793	57	1.10	1922	1610	10	.66	1889	1791	161
9	.61	1902	1746	4.61	1871	1802	64					.74	1886	1791	181
10	.68	1901	1756	5.10	1868	1814	71					.82	1884	1791	201
⋮												⋮			
15	1.00	1880	1790									1.21	1876	1804	301
⋮												⋮			
20	1.34	1872	1803	10.08	1850	1832	141					1.60	1871	1808	401
⋮															
30	1.93	1862	1813												
⋮															
40	2.54	1867	1818												
⋮															
50	3.18	1854	1821												

+Inaccurate Tangent Plane

*Heuristic retains minimum previous value

TABLE 4. COMPUTATION RESULTS FOR PROBLEM E

Item	Frank-Wolfe			Transfer Decomposition (Fig. 2)				Minimax Heuristic				Alt. Transfer Decomposition (Fig. 5)			
	CPU Secs	Obj.	Bound	CPU Secs	Obj.	Bound	Number Subnet Calls	CPU Secs	Obj.	Bound	Number Subnet Calls	CPU Secs	Obj.	Bound	Number Subnet Calls
0	.04	6091	0	.24	2736	0	1	.22	2736	0	1	.05	10431	0	1
1	.11	2024	0	.68	1886	0	8	.33	2655	0	3	.14	1906	0	21
2	.17	1794	351	1.38	1749	1193	15	.43	1889	0	4	.22	1763	1552	41
3	.24	1777	1585	2.06	1743	1665	22	.54	1917*	1194	5	.31	1737	1596	61
4	.30	1766	1586	2.56	1751†	1675	29	.66	1786	1222	6	.39	1729	1661	81
5	.36	1760	1649	3.17	1754†	1675	36	.77	1892*	1621	7	.47	1726	1682	101
6	.42	1755	1649	3.66	1740	1683	43	.88	1793	1621	8	.56	1725	1682	121
7	.48	1749	1649	4.36	1743	1695	50	.99	1776	1621	9	.64	1724	1683	141
8	.55	1746	1653	5.03	1732	1695	57	1.10	1748	1633	10	.72	1724	1697	161
9	.61	1744	1673	5.76	1729	1695	65					.81	1723	1697	181
10	.68	1742	1673	6.30	1750†	1695	73					.89	1723	1697	201
⋮												⋮			
15	1.00	1743	1688									1.31	1721	1699	301
⋮												⋮			
20	1.31	1729	1694									1.74	1720	1700	401
⋮															
30	1.93	1723	1696												
⋮															
40	2.54	1719	1707												
⋮															
50	3.18	1716	1707												

†Inaccurate line search

*Heuristic retains minimum previous value

TABLE 5. COMPUTATION RESULTS FOR PROBLEM H

	Frank-Wolfe			Transfer Decomposition (Fig. 2)				Minimax Heuristic				Alt. Transfer Decomposition (Fig. 6)			
Item	CPU Secs	Obj.	Bound	CPU Secs	Obj.	Bound	Number Subnet Calls	CPU Secs	Obj.	Bound	Number Subnet Calls	CPU Secs	Obj.	Bound	Number Subnet Calls
0	.04	6997	0	.22	6302	0	1	.22	6301	0	1	.05	6997	0	1
1	.11	1654	0	.61	1623	0	8	.33	2279	0	3	.12	1653	0	21
2	.17	1612	1153	1.06	1605	1375	15	.43	1535	0	4	.20	1606	1116	41
3	.24	1592	1187	1.55	1591	1375	22	.54	1751*	1501	5	.28	1590	1233	61
4	.30	1581	1381	2.00	1578	1375	29	.66	1603	1501	6	.36	1578	1373	81
5	.36	1571	1381	2.46	1570	1375	36	.77	1843*	1501	7	.42	1570	1373	101
6	.42	1567	1454	2.92	1566	1459	43	.88	1554	1501	8	.53	1565	1422	121
7	.48	1563	1454	3.39	1559	1459	50	.99	1536	1501	9	.61	1560	1422	141
8	.55	1558	1454	3.84	1557	1459	57	1.10	1532	1501	10	.70	1551	1444	161
9	.61	1557	1454	4.31	1552	1459	64					.78	1553	1444	181
10	.68	1552	1454	4.76	1549	1459	71					.86	1551	1465	201
⋮												⋮			
15	1.00	1543	1454									1.28	1543	1476	301
⋮															
20	1.31	1536	1478									1.68	1539	1491	401
⋮												⋮			
30	1.93	1531	1513												
⋮															
40	2.54	1529	1514												
⋮															
50	3.18	1529	1514												

*Heuristic retains prior minimum value

	Problem E		Problem H	
Link	T_0	Capacity	T_0	Capacity
1-5	5	10	5	10
1-6	6	16	6	16
2-5	3	35	3	35
2-6	9	18	9	18
5-6	5	25	1	50
5-7	5	25	5	25
5-9	5	25	2	35
6-5	5	25	1	50
6-8	5	25	5	25
6-9	5	25	2	35
7-3	3	25	3	25
7-4	6	24	6	24
7-8	5	25	1	50
8-3	8	39	8	39
8-4	6	43	6	43
8-7	5	25	1	50
9-7	5	25	2	35
9-8	5	25	2	25
Trip Tables				

	3	4
1	10	20
2	30	40

	3	4
1	25	25
2	25	25

FIGURE 7. DATA FOR PROBLEMS E AND H

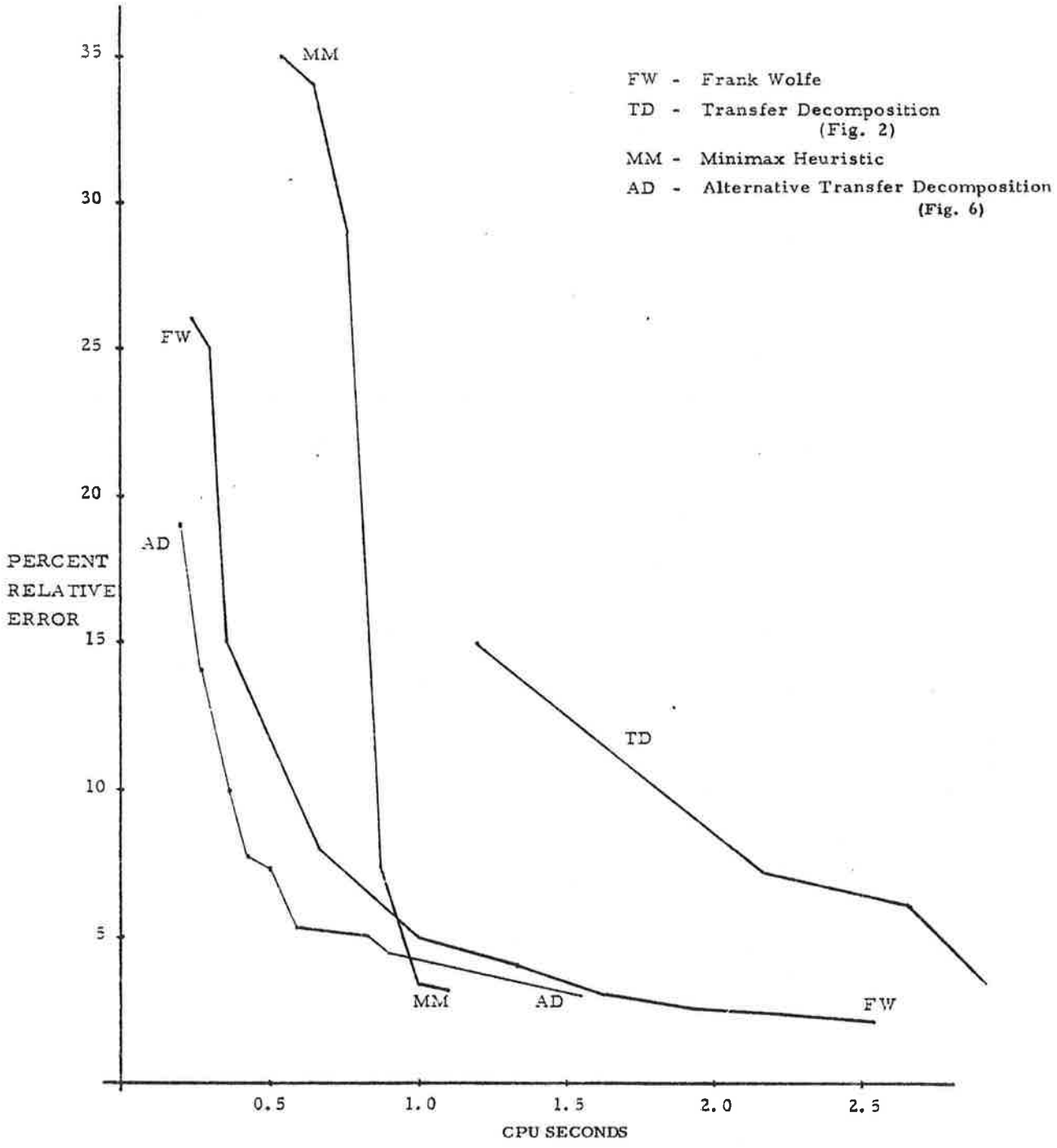


FIGURE 8. RANDOM PROBLEM -- RELATIVE ERROR COMPARISON

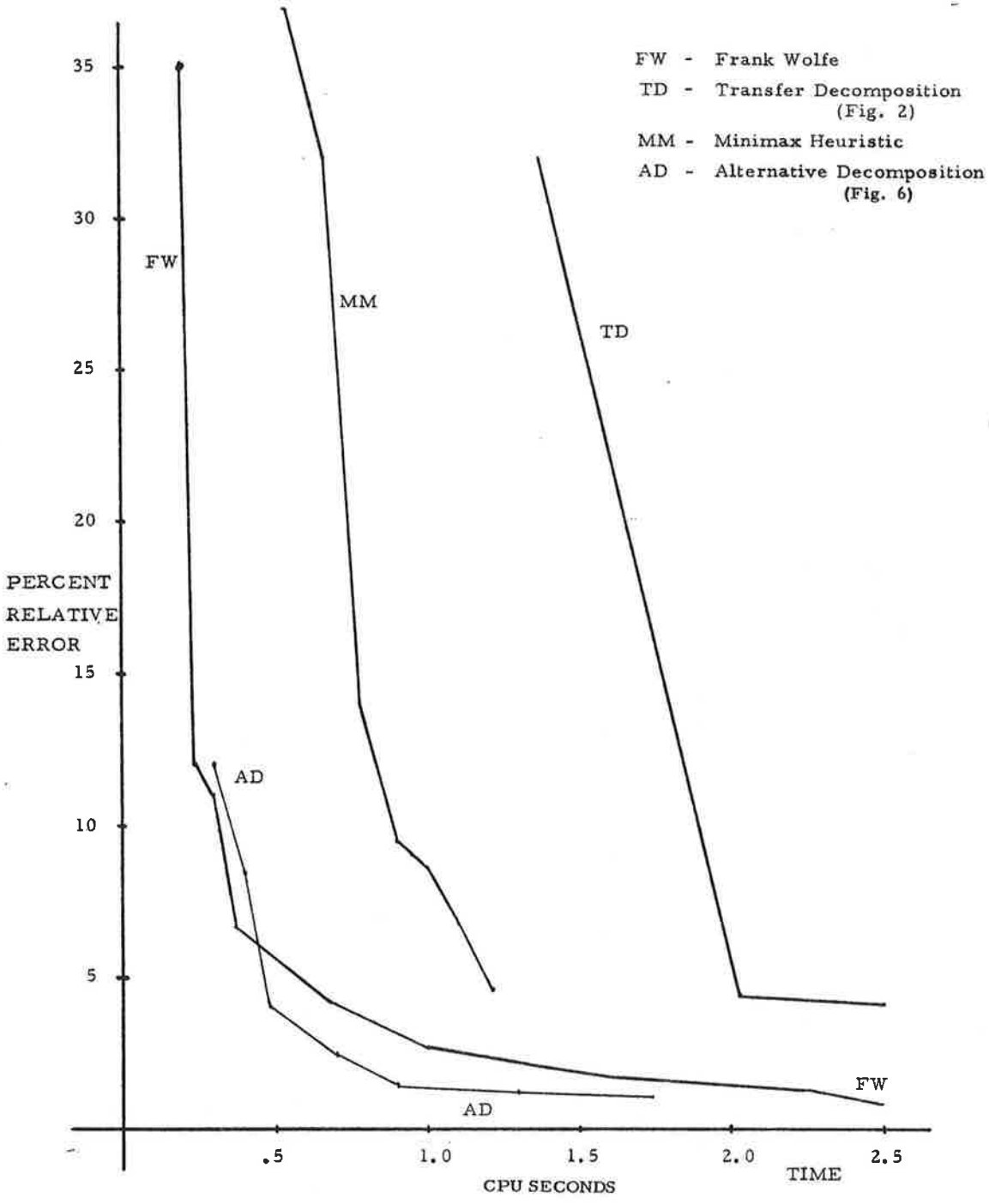


FIGURE 9. PROBLEM E -- RELATIVE ERROR COMPARISON

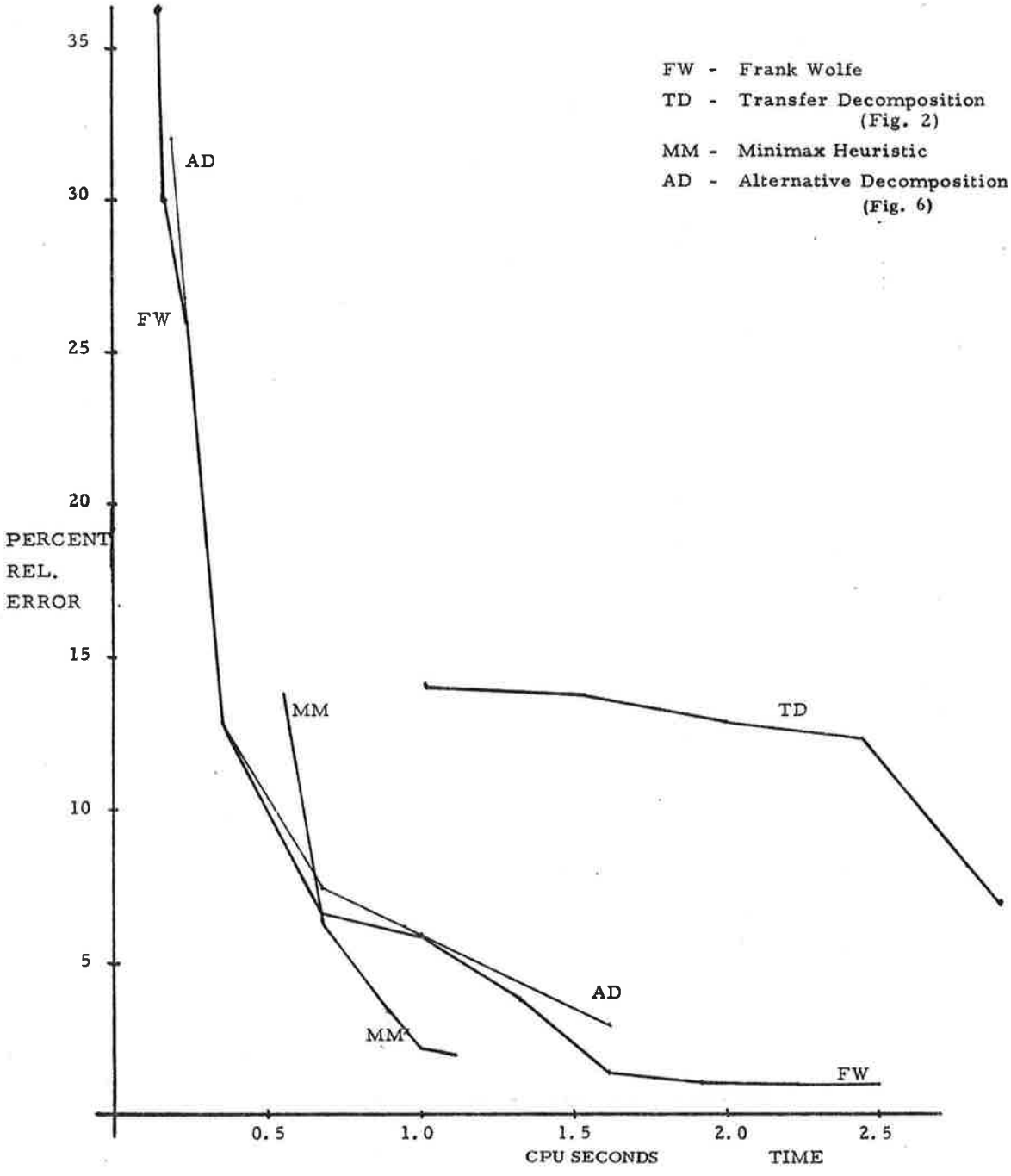


FIGURE 10. PROBLEM H -- RELATIVE ERROR COMPARISON

A production computer code of any of these methods would require extensive testing before determining all parameters.

Summary and Conclusions

The results of the previous section demonstrate that Transfer Decomposition is a viable method for solving convex multicommodity flow problems. Perhaps surprisingly, the experimental results with the alternative transfer decomposition of Figure 6 show that it can be competitive with the Frank-Wolfe method if the subnetwork is chosen in an appropriate manner. Intuitively, this requires that the subnetwork be small, so that problem (S) is easy to solve relative to the full network problem (P).

Transfer Decomposition has a number of advantages which make it appealing when compared with the Geographic Decomposition method of Maier and Robinson [4]. Chief among these are:

(a) Transfer Decomposition does not require, as a preliminary step, the decomposition of problem (P) by commodity. The automatic aggregation of the original commodities as flows on the pseudo-links is very important because, unless there are origin/destination points in the subnetwork, the subproblem (S) may be treated as a problem of assigning flows between access and egress points of the subnetwork.

(b) Both problem (M) and (S) are convex flow problems of the same form as (P). Therefore existing computer codes may be used to solve both problems, and, with current knowledge about computer time required to solve (P), it should be possible to predict the performance of Transfer Decomposition for given choices of (M) and (S).

The Geographic Decomposition method [4] was designed for large, loosely connected networks. In such cases there are certain to be many origin/destination points within each geographic region of the decomposed network. Therefore, it is appealing to decompose (P) by commodity prior to the decomposition by region. Transfer Decomposition, on the other hand, is designed for flowing subnetworks such as major arteries (expressways) in an urban area. In this case, there are no origin/destination points within the subnetwork. (The presence of such does not invalidate the method, however. The subnetwork of Figure 6 contains Node 4, a destination node.)

Point (b), above, assumes that the method for solving convex flow problems will work when the objective value is convex, but not necessarily differentiable. Strictly speaking, methods such as Frank-Wolfe require differentiability for convergence. There are a variety of ways, including perturbation methods, which can be employed to avoid this difficulty, however, it is unlikely that such would be needed in practical computer codes. As already mentioned, the objective of (M) is differentiable almost everywhere.

All of the methods mentioned in this paper are decompositions in the mathematical programming sense, including the Frank-Wolfe method which is not usually thought of as such. This commonality means that they share two unfortunate idiosyncrasies of decompositions. First, they are slow to converge, i.e., the percent relative error in the objective value (see Figures 8 - 10) approaches zero asymptotically. No satisfactory explanation exists for this phenomenon and therefore there is little likelihood of significant improvement. For practical use, therefore, one usually must settle for relative error in the range of 2%-5%. Computational

experience with a variety of decompositions indicates that this range of error is attainable at reasonable computation cost. Attempting to attain error under 1 percent can often be prohibitively expensive.

The second feature of decompositions is a disadvantage with respect only to the Extraction Aggregation Model [2]. Transportation analysts are often interested only in flows on a subnetwork, and hence they employ this model as evidenced by the case studies in [2]. Decompositions, however, address the full problem (P) and thereby obtain flows for the entire network. Other than heuristics, there does not appear to be a way to obtain the correct flows for the subnetwork links without flowing the entire network. Even in heuristic methods, there is a theoretical requirement for at least one full set of flows for the network in order to measure the error in a set of heuristically derived subnetwork flows.*

The results presented here raise two interesting and potentially fruitful areas for computational development:

(a) Develop a heuristic for flowing highway subnetworks where the inputs are estimated extreme points of the subnetwork trip table space (cf. Figure 5). The master problem would be of the form

$$\begin{aligned} \min \quad & \bar{h}(z) && (\bar{M}) \\ \text{s.t.} \quad & z = \sum \lambda_i z^i \\ & \lambda_i \geq 0 \\ & \sum \lambda_i = 1 , \end{aligned}$$

*See Paper 3 of this volume for an elaboration of this point.

where each z^i is an extreme point, and $\bar{h}$ is defined to be the optimal extreme value of a subproblem having the same form as (S).

It would be relatively easy to gather z^i data for an existing subnetwork by having traffic engineers count flows at access and egress points. For design networks, the z^i could be estimated. Of course, the z^i need not be extreme points so long as their convex hull contains an optimal trip table.

(b) Perform sufficient computation experiments to develop prediction formulas which would estimate computer time and space requirements for each decomposition method proposed.* Some simple analysis would enhance the development of these formulas. For example, Transfer Decomposition involves a trade-off between the size of the original subnetwork and the number of pseudo-links.

Both of these suggestions build on the results of this paper and offer the potential for long term utilization by transportation planners.

---

*Barton has proposed another decomposition for the Extraction Aggregation Model. See Paper 2 of this report.

REFERENCES

1. Geoffrion, A. M., "Generalized Benders Decomposition," J. Optimization Theory and Appl. 10, No. 4 (1972).

2. Hearn, D. W., "Network Aggregation in Transportation Planning Models, Part I." Mathtech Final Report, DOT-TSC-RSPD-78-8, I, April, 1978.

3. Lasdon, L. S., Optimization Theory for Large Systems. The Macmillan Company, 1970.

4. Maier, S. F. and Robinson, D. C., "The Application of Geographic Decomposition to the Solution of Large-Scale Traffic Assignment Problems." G. S. B. A. Working Paper No. 213, Duke University, July 1977.

5. Nguyen, S. and James, L., "TRAFFIC - An Equilibrium Traffic Assignment Program," Centre de Recherche sur les Transports, Publication #17, University of Montreal, March 1975.

2. SUBSET DECOMPOSITION

Russell R. Barton

Introduction

In this paper we show that the aggregation procedure (steps 5 - 7)* given by Hearn (1977, p. 2-26) can be accomplished through use of another Bender's decomposition technique. Although in extraction aggregation the network is usually decomposed into two parts (extracted and non-extracted arcs), this subset decomposition technique can be used on networks decomposable into p parts. We will show how this technique leads to a particular algorithm that is especially interesting when one or more of the parts have constant cost functions (e.g., the Wilson problem).

Notation

We make four assumptions.

Assumption 1. The original network is connected.

Assumption 2. Each subnetwork is strongly connected, i.e., a path exists between every node pair of the subnetwork. (May be relaxed to strong connectedness of the split nodes of the subnetwork - see below).

Assumption 3. Nodes are adjacent to at most two of the subnetworks (trivial for $p = 2$; may be relaxed when $p > 2$).

Assumption 4. Nodes adjacent to more than one subnetwork are not centroids (may be relaxed).

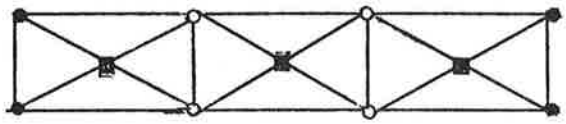
We define nodes adjacent to two subnetworks as "split" nodes (c.f. Maier & Robinson (1977)). Two legitimate decompositions for a particular network are shown in Figure 11.

---

*See the overview section of this report.

I. Regional Decomposition

Original Network:

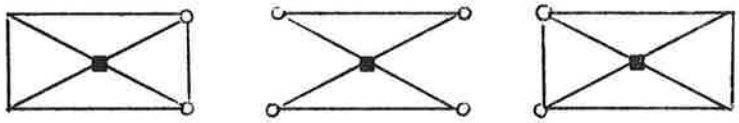


■ ≡ centroid

○ ≡ split node

● ≡ other node

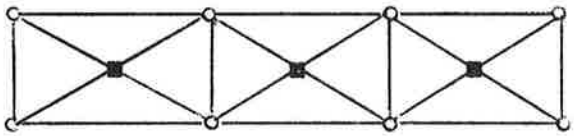
Decomposition:



Part 1 (region 1) | Part 2 (region 2) | Part 3 (region 3)

II. Aggregated (Extracted) Subnetwork Decomposition

Original Network:



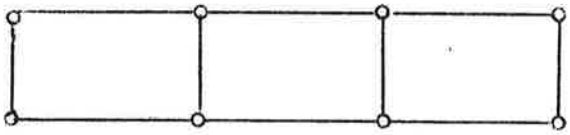
Decomposition:

■ ≡ centroid

○ ≡ split node

● ≡ other node (none here)

part 1



part 2

FIGURE 11. TWO ILLUSTRATIONS OF NETWORK DECOMPOSITION

Let commodity q be the demand between origin-destination pair q, and for split node i, let g_i^q represent the flow of commodity q passing between the two parts of the network incident to node i. Let δ_{ik} specify the relationship of g_i^q to the k^{th} subnetwork part:

$\delta_{ik} = -1$ if g_i^q denotes flow out of subnetwork k

$= 1$ if g_i^q denotes flow into subnetwork k

$= 0$ if split node i is not incident to subnetwork k.

Model Structure

We now present a particular subset decomposition model to simplify the discussion. We consider the case of two subnetwork parts (i.e., p=2). The extraction aggregation problem is an example of this case. If we let n_k^q be the vector whose components are $\delta_{ik} g_i^q$, then we may write the equilibrium traffic assignment problem as:

P1 $$\text{minimize} \sum_{ij \in A_o} \int_0^{f_{ij}} c_{ij}(u)du + \sum_{ij \in \bar{A}} \int_0^{f_{ij}} c_{ij}(u)du$$

s.t.

$$f_{ij} = \sum_q x_{ij}^q$$

$$\left(A \mid \bar{A}\right) \begin{pmatrix} x^q \\ \bar{x}^q \end{pmatrix} = \hat{b}^q \qquad \forall q$$

$$x \geq 0,$$

where x^q is the vector of q-commodity flows on links in A and $\bar{x}^q$ is the vector of q-commodity flows on links in $\bar{A}$.

Assumption 3 implies the following structure for the $\left(A \mid \overline{A}\right)$ matrix constraints.

$$\begin{array}{l} \text{nodes in part 1} \\ \text{split nodes} \\ \text{nodes in part 2} \end{array} \left(\begin{array}{c|c} A_{11} & 0 \\ A_{21} & A_{22} \\ 0 & A_{23} \end{array}\right) \begin{pmatrix} x^q \\ \overline{x}^q \end{pmatrix} = \begin{pmatrix} b^q \\ 0 \\ \overline{b}^q \end{pmatrix}.$$

Define $n^q \equiv A_{21} x^q$, $\overline{n}^q \equiv A_{22} \overline{x}^q$ and note that $n^q = -\overline{n}^q$. Then we can rewrite problem P1 in a separated fashion:

$$\boxed{P1'} \quad \text{minimize} \sum_{ij \in A} \int_0^{f_{ij}} c_{ij}(u)du + \sum_{ij \in \overline{A}} \int_0^{f_{ij}} c_{ij}(u)du$$

$$\text{s.t.} \qquad f_{ij} = \sum_q x_{ij}^q \quad ij \in A \qquad f_{ij} = \sum_q x_{ij}^q \quad ij \in \overline{A}$$

$$\forall q \left\{ \begin{array}{l} \begin{pmatrix} A_{11} \\ A_{21} \end{pmatrix} x^q = \begin{pmatrix} b^q \\ n^q \end{pmatrix} \qquad \begin{pmatrix} A_{22} \\ A_{23} \end{pmatrix} \overline{x}^q = \begin{pmatrix} -\overline{n}^q \\ \overline{b}^q \end{pmatrix} \\ x^q \geq 0 \qquad\qquad \overline{x}^q \geq 0 \\ \underbrace{\qquad\qquad\qquad\qquad\qquad\qquad}_{n^q \text{ feasible}} \end{array} \right. \qquad (*).$$

Thus P1' is separable except for the constraint (*) so we may write (P1') as

$$\boxed{P2} \quad \left. \begin{array}{ll} \text{minimize} & Z(n) + \overline{Z}(n) \\ \text{s.t.} & n^q \text{ feasible } \forall_q \end{array} \right\} \text{master problem}$$

where

$$Z(n) = \text{Min} \sum_{ij \in A} \int_0^{f_{ij}} c_{ij}(u)\, du$$

$$\text{s.t.} \quad f_{ij} = \sum_q x_{ij}^q \qquad ij \in A$$

$$\begin{pmatrix} A_{11} \\ A_{21} \end{pmatrix} x^q = \begin{pmatrix} b^q \\ n^q \end{pmatrix} \qquad \forall q$$

$$x^q \geq 0$$

$$\bar{Z}(n) = \text{Min} \sum_{ij \in \bar{A}} \int_0^{f_{ij}} c_{ij}(u)\, du$$

$$\text{s.t.} \quad f_{ij} = \sum_q \bar{x}_{ij}^q \qquad ij \in \bar{A}$$

$$\begin{pmatrix} A_{22} \\ A_{23} \end{pmatrix} \bar{x}^q = \begin{pmatrix} -\bar{n}^q \\ \bar{b}^q \end{pmatrix} \qquad \forall q$$

$$\bar{x}^q \geq 0$$

SUBPROBLEMS

The strong connectedness of each subnetwork (Assumption 2) implies that necessary and sufficient conditions on n^q for subnetwork feasibility are that:

$$b^q \cdot \vec{1} + n^q \cdot \vec{1} = 0 \qquad \forall q \quad (1)$$

$$\bar{b}^q \cdot \vec{1} - n^q \cdot \vec{1} = 0 \qquad \forall q \quad (2),$$

where $\vec{1}$ is a vector of ones. The feasibility of P1 implies that $b^q \cdot \vec{1} = -\bar{b}^q \cdot \vec{1}$, and so the n^q feasibility requirement (*) is simply that n^q lie in the subspace generated by (1) (equivalently (2)).

Thus we see that the master problem in P2 can be treated as an unconstrained problem if the gradients of $Z_1 + Z_2$ have components which sum to zero, forcing successive values of n^q to always lie in the subspace defined by (1) above.

The subproblem dual variables are the gradient components and, because the subproblem constraints have a redundancy, may be adjusted to meet this requirement. By defining the subproblem dual variables in

this way, we can sum (over all subproblems) the dual variables corresponding to the split node constraints to get a usuable master problem gradient. This process is described in more detail in the next section.

Solution Technique

A natural algorithm for solving P2 would be to apply a subgradient optimization technique for solving the unconstrained master problem, and apply the Frank-Wolfe algorithm to each of the multicommodity flow subproblems. Dual variable values for the split node flow constraints would be calculated from the link travel times, summed over the subproblems, and used as the subgradient for the next master step. The Frank-Wolfe algorithm used in traffic assignment codes (e.g. UROAD) must be modified to solve the subproblems of P2. This modification is explained in the appendix.

The algorithm is outlined below:

SUBSET DECOMPOSITION ALGORITHM

Step 0 Construct a feasible n vector. This can be done for each commodity q by defining $n_2^q, \ldots, n_p^q$ arbitrarily, and setting

$$n_1^q = -b^q \cdot \overset{>}{1} - \sum_{i=2} n_i^q \cdot$$

Step 1 Solve each multicommodity flow subproblem using the modified Frank-Wolfe algorithm (see Appendix). A lower bound on the overall objective function is given by the sum of the subproblem lower bounds. An upper bound is given by the sum of the subproblem optimal objective function values.

Solution of subproblem 2 yields dual variables for each node and commodity, u^q. Assume these average μ^q for the split nodes. Redefining the dual variables to be $u^q - \mu^q$ forces the split node sum to be zero as required. These adjusted values are then used to update the vectors n^q. This leads to two new subproblems and the process continues.

Alternate Algorithm

Maier and Robinson suggest solving the decomposed problem similar to P2. They explicitly include the n^q, $\bar{n}^q$ feasibility constraints in the master problem, and further decompose the problem by commodity. Thus P2 is solved iteratively for each commodity, holding the others fixed. The master problem incorporates inequality constraints that are relaxations of the Z, $\bar{Z}$ equalities defined by the subproblems. Thus their formulation involves a master problem which increases in size at each iteration. In Subset Decomposition, this does not occur and the master problem is effectively unconstrained.

APPENDIX

Frank-Wolfe Solution of the Subset Decomposition Subproblem

For the traditional traffic assignment problem, each commodity is associated with a unique origin, so that each set of commodity constraints has only one node with positive net flow (i.e., one component of b^q greater than zero). This property enables one to find extreme points (flows) by constructing minimal path trees rooted at origin q for each commodity, and assign all q-commodity flows to these paths. For the subproblems of P2, however, split nodes as well as origin q may have positive net flow for commodity q. Since, in this case, demands for commodity q can be met by a number of sources, one must do more than construct a minimal path q-rooted tree to solve the L.P. (transshipment problem) phase of the F-W algorithm.

One method for solving the L.P. is to transform the transshipment problem into a transportation problem by finding shortest paths. The L.P. phase of the F-W algorithm for the subproblems would then consist of three operations:

Subproblem Solution

L.P. Phase of Frank-Wolfe Algorithm

Step 1. Construct shortest path trees for the subproblem, one for each origin in the subnetwork and one for each split node that has positive net flow for some commodity. Record the shortest path distances for all origins (and positive split nodes) to all destinations (and split nodes with negative net flow).

Step 2. Using the distances from Step 1, along with trip demands defined by the b^q and n^q vectors, solve the transportation problem for all commodities simultaneously.

Step 3. Using the solution to Step 2, load appropriate flows onto the links in the shortest paths.

Note that we can solve Step 3 in one of two ways:

a) If the storage requirements are not too large, we can retain the minimal path trees found in Step 1 to assign the flows found in Step 2.

b) Alternatively, we can re-solve Step 1, assigning the flows of Step 2 as the minimal paths are found.

REFERENCES

1. Hearn, D. W. (1977), "Network Aggregation in Transportation Planning Models." Mathtech Final Report, DOT-TSC-RSPD-78-8, I, April 1978.

2. Maier, S., and Robinson, D. C. (1977), "The Application of Geographic Decomposition to the Solution of Large-Scale Traffic Assignment Problems." GSBA Working Paper No. 213, Duke University, July 1977.

3. MINIMIZING THE GAP FUNCTION IN CERTAIN CONVEX PROGRAMS

Donald W. Hearn

Introduction

In an earlier Mathtech report [2] we have pointed out the importance of the "gap" function (defined below) in certain convex programs such as the traffic assignment problem. This paper investigates the practicality of direct minimization of this function by the methods of nondifferentiable optimization, as opposed to the indirect minimization which occurs when the convex program is solved by the Frank-Wolfe method. Our preliminary results include an adaptation of a convergent method due to Polyak, and computational experiments which illuminate features of the method.

Consider the convex programming problem

$$\min \ f(x) \quad \text{s.t.} \quad x \in S = \{x \mid Ax = b, \ x \geqq 0\}, \qquad \text{(P0)}$$

for f convex and twice continuously differentiable. We assume throughout that S is bounded and thus there exists x^* which solves (P0).

Our approach is to solve (P0) indirectly by minimizing an auxiliary function, $G(x)$ defined as follows:

$$G(x) = \max_{y} \nabla f(x)(x - y) \quad \text{s.t.} \quad y \in S, \qquad \text{(P1)}$$

The function G has a number of interpretations; it is the negative of the directional derivative of f in the direction of some extreme point $y(x)$ which solves the maximization problem in (P1), and it can be considered as the difference between $f(x)$ and a lower bound on $f(x^*)$, i.e.,

$$f(x^*) \geqq f(x) - G(x) \quad x \in S$$

as is easily seen from the assumption that f is convex. Finally, as the following result shows, G(x) is exactly the duality gap for problem (P0). For this reason we refer to G as the "gap" function.

Lemma 1: For any x the dual problem is feasible and G(x) is the difference between f(x) and the dual objective value.

Proof: The dual of (P0) may be written

$$\max_{(x,\lambda,\mu)} \; L(x,\lambda,\mu) \qquad \text{(D0)}$$

$$\text{s.t.} \quad \nabla_x L(x,\lambda,\mu) = 0$$

$$\mu \geqq 0$$

$$\text{where} \quad L(x,\lambda,\mu) = f(x) - \lambda^T(Ax-b) - \mu^T x.$$

The constraints may be written $\lambda^T A \leqq \nabla f(x)$ which has a feasible solution λ if and only if there exists non-negative z such that $Az = 0$ and $\nabla f(x)^T z < 0$. For $x \neq x^*$, $y(x)$ exists since S is bounded, and $z = y(x)-x$ satisfies these conditions. For $x=x^*$, the existence of λ^* such that $(\lambda^*)^T A \leqq \nabla f(x^*)$ is guaranteed by the Kuhn-Tucker conditions for (P0). Hence (D0) is feasible for all $x \in S$.

Under the assumption of x fixed, the Lagrangian function, L, must be maximized in the dual variables (μ, λ) subject to the constraints. From the stationarity condition we have $\nabla f(x) - \lambda^T A - \mu^T = 0$ which implies $\nabla f(x)x - \lambda^T Ax = \mu^T x$.

Substitution in the Lagrangian function reduces (D0) to

$$\max_{\lambda} \quad f(x) - \nabla f(x)x + \lambda^T b$$

$$\text{s.t.} \quad \lambda^T A \leqq \nabla f(x)$$

but since

$$\begin{array}{lll} \max \ \lambda^T b & = & \min \ \nabla f(x) y \\ \text{s.t.} \ \ \lambda^T A \leqq \nabla f(x) & & y \in S, \end{array}$$

$G(x)$ is the duality gap.

In general, G will be nondifferentiable and nonconvex, but $G(x) \geqq 0$, $x \in S$, and $G(x^*) = 0$ if and only if x^* solves both (P0) and (P1), for otherwise $(y - x^*)$ is a feasible descent direction for f. When in addition, G is convex in x (f quadratic is one example), we show that one can solve (P1) by nondifferentiable methods. Small examples indicate that the nondifferentiability of G, rather than being a hindrance, may actually help in obtaining x^*.

The organization of this paper is as follows. In the next section we motivate the gap function using a small example due to P. Wolfe. Next are some results concerning the convexity of G and calculation of its subderivatives. We then state and prove convergence of an algorithm for solving (P1) assuming G is convex. Finally, the results of solving some small problems are presented.

Motivation

Part of the motivation for this investigation comes from the well-known Frank-Wolfe algorithm [1] of nonlinear programming. For problem (P0) the algorithm is

Step 0. Choose $x \in S$.

Step 1. Solve $\min_y \nabla f(x)y$ s.t. $y \in S$ and call the solution $\bar{y}$.

Step 2. Solve $\min_{0 \leq \lambda \leq 1} f(\lambda x + (1 - \lambda)\bar{y})$ and call the solution $\bar{\lambda}$.

Step 3. Replace x by $\bar{\lambda}x + (1 - \bar{\lambda})\bar{y}$.

Step 4. Go to 1.

Despite its elegant simplicity, and the fact that the process converges to x^* for any starting point [9], numerous applications have revealed that convergence is often quite slow. An article by Wolfe [7] explains this difficulty with a small example. In Figure 13, let the feasible region S be the triangle and the objective of problem (P0) be $f(x) = 1/2||x||^2$. Starting from x^0, the Frank-Wolfe algorithm produces iterates x^k which converge slowly to x^* at the midpoint of the base of the triangle. This illustration is indicative of many instances when the Frank-Wolfe iterates zig-zag toward the face of S which contains the optimal point. The phenomenon results from the fact that all movements from one iterate to the next are "toward" extreme points of S.

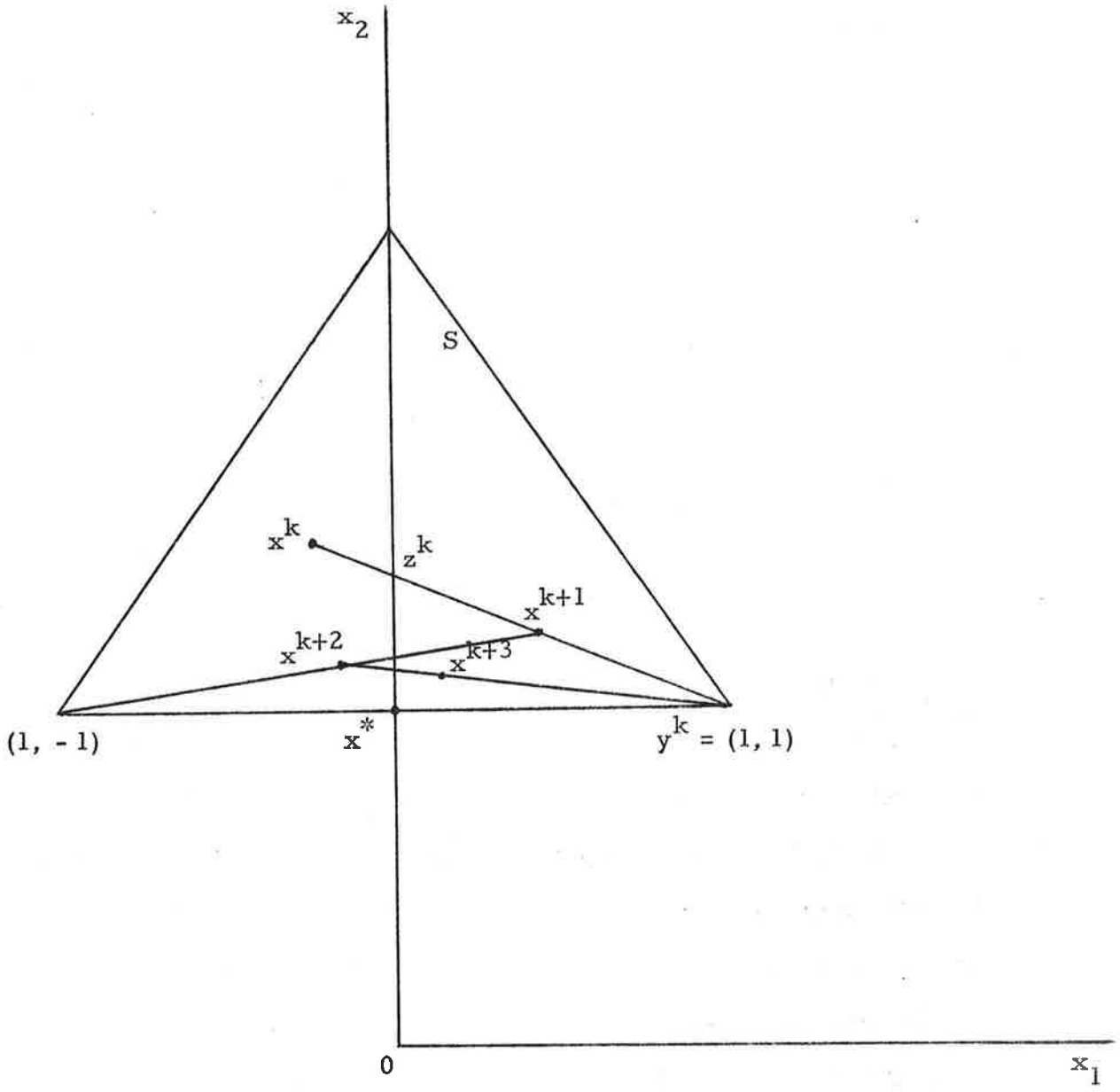


FIGURE 13. WOLFE'S EXAMPLE

Consideration of (P1) rather than (P0) leads to two interesting possibilities. First, as we will show, the search direction produced by the Frank-Wolfe algorithm ($\bar{y} - x$ in the above) is a descent direction for G as well as f. Thus, at points where G is differentiable, one can apply the Frank-Wolfe method to (P1). Furthermore, we will show how to compute feasible steepest descent directions when G is not differentiable. This approach solves the problem in Figure 1 in two iterations. To see this, note that

$$G(x) = \max_{y \in S} \nabla f(x)(x - y) = \|x\|^2 - \min_{y \in S} (x_1 y_1 + x_2 y_2)$$

$$= \|x\|^2 - x_2 + |x_1| \qquad \text{for all } x \in S.$$

From x^k, minimization of G along the direction $y^k - x^k$ yields the point z^k. Then, from z^k, the steepest descent direction is $x^* - z^k$.

The other possibility for solving (P1) is to apply the methods of nondifferentiable optimization. Chief among these is the method of Polyak [6] which is attractive here because the optimal value of (P1) is known to be zero.

Theory

To consider solving (P1), it is probably necessary that G be at least quasiconvex. We have not explored this possibility, but here assume rather strong conditions that imply G is convex. Under these conditions we also derive a formula for subderivatives of G. In addition, we show that the direction of the line search generated by the Frank-Wolfe algorithm for minimizing f is also a descent direction for G when G is differentiable at x. This result holds whether G is convex or not.

For convenience we first introduce the following notation:

$$W(z) = \min_{y \in S} z^T y = \max_{\text{s.t. } \lambda^T A \leqq z} \lambda^T b \qquad (3.1)$$

$$S(x) = W(\nabla f(x))$$

$$F(x) = \nabla f(x) x$$

$$Y(x) = \left\{ \bar{y} \in S \mid \nabla f(x) \bar{y} = \min_{y \in S} \nabla f(x) y \right\}.$$

Thus we have

$$S(x) = \nabla f(x) y \quad \forall y \in Y(x)$$

and $$G(x) = F(x) - S(x) .$$

Lemma 2. Let F(x) be convex and assume that each component of $\nabla f(x)$ is concave in x. Then G(x) is convex in x.

Proof: From (3.1) W is concave because the "min" part of its definition shows that it is the minimum of functions linear in z. From the "max"

part of its definition it is clear that it is also nondecreasing, i.e., $z_2 \geqq z_1$ implies $W(z_2) \geqq W(z_1)$. Therefore,

$$\begin{aligned} S(\lambda x_1 + (1-\lambda)x_2) &= W(\nabla f(\lambda x_1 + (1-\lambda)x_2)) \\ &\geqq W(\lambda \nabla f(x_1) + (1-\lambda)\nabla f(x_2)) \\ &\geqq \lambda W(\nabla f(x_1)) + (1-\lambda) W(\nabla f(x_2)) \\ &= \lambda S(x_1) + (1-\lambda) S(x_2), \end{aligned}$$

the first inequality uses the concavity of $\nabla f(x)$ and the monotonicity of W, while the second employs the concavity of W. The result follows since G is the difference between a convex and a concave function.

Obviously the conditions of the lemma are satisfied when $f(x)$ is a convex quadratic function, and they also apply, for example, when

$$f(x) = \sum_j \int_0^{x_j} \log(1+t)dt.$$

However, many simple convex functions such as scalar ones of the form $f(x) = x^k$, $k>2$, $x \geq 0$, do not satisfy the condition that f be convex with concave derivatives.

Lemma 3. If the components of $\nabla f(x)$ are concave, the subdifferential of $S(x)$ is given by the "chain rule:"

$$\begin{aligned} \partial S(x) &= \nabla^2 f(x)\, \partial W(\nabla f(x)) \\ &= \nabla^2 f(x) y \qquad \forall\, y \in Y(x). \end{aligned}$$

Proof: Since components of $\nabla f(x)$ are concave

$$\nabla f(x) + (z - x)\, \nabla^2 f(x) \geqq \nabla f(z), \qquad x, z \in S$$

and for $y \geqq 0$, this implies

$$(z - x)\, \nabla^2 f(x)\, y \geqq [\nabla f(z) - \nabla f(x)]\, y.$$

But if $y \in Y(x)$, then y is a subgradient of $W(\nabla f(x))$, hence

$$(z - x)\, \nabla^2 f(x)\, y \geqq W(\nabla f(z)) - W(\nabla f(x)) \qquad y \in Y(x).$$

$$= S(z) - S(x),$$

and the result follows.

Corollary $\quad \partial G(x) = \partial F(x) - \partial S(x)$

$$= \nabla f(x) + \nabla^2 f(x)(x - y) \qquad y \in Y(x).$$

Corollary G is differentiable at x if and only if $Y(x)$ is a singleton (i.e., the solution of the Frank-Wolfe subproblem is unique).

Corollary If G is differentiable at x then the Frank-Wolfe descent direction (for f) is also a descent direction for G.

Proof $\quad (y - x)\, \nabla G(x) = \nabla f(x)(y - x) - (y - x)\, \nabla^2 f(x)(y - x) < 0$, $y \in Y(x)$, because $(y - x)$ is a descent direction for f and because f convex implies $\nabla^2 f(x)$ positive semidefinite.

Poljak's Algorithm and Convergence

Poljak [6] has proposed an algorithm for minimizing a convex function which need not be differentiable. The key requirement of his method is that the optimal objective value be known. It is therefore attractive for this problem since $G(x^*) = 0$. Another attractive feature is that the method is a simple recursion. Each iteration a step is made in the direction of any negative subgradient of G; no line searches are required. The chief disadvantage of the method is that a projection on to S must be made to maintain feasibility. In one variable, Poljak's algorithm is the same as Newton's method for finding the root of a function.

We denote by $P_S(z)$ the projection of the point z on the set S. Poljak's method, applied to (P1) is

$$x^{k+1} = P_S(x^k - \rho_k t^k)$$

$$\rho_k = \lambda_k \frac{G(x^k)}{\|t^k\|^2} \qquad t^k \in \partial G(x^k)$$

$$0 \leqq \epsilon_1 < \lambda_k \leqq 2 - \epsilon_2 \qquad \epsilon_1, \epsilon_2 > 0.$$

Theorem (Polyak [5]): Let $\bar{x}$ be the limit of any convergent subsequence. Then $G(\bar{x}) = 0$.

Proof: Using the property that $\|P_S(z - w)\| \leqq \|z - w\|$ and the subgradient inequality: $G(x^*) \geqq G(x^k) + t^k(x^* - x^k)$ for $t^k \in \partial G(x^k)$, where x^* is a minimum point, we have

$$\begin{aligned} \|x^{k+1} - x^*\| &= \|P_S(x^k - \rho_k t^k - x^*)\|^2 \\ &\leqq \|x^k - \rho_k t^k - x^*\|^2 \\ &\leqq \|x^k - x^*\|^2 - 2(x^k - x^*)\rho_k t^k + \rho_k^2 \|t^k\|^2 \end{aligned}$$

$$\leq \|x^k - x^*\|^2 + \frac{\lambda_k^2}{\|t^k\|^2} G(x^k)^2 - 2\lambda_k \frac{G(x^k)}{\|t^k\|^2} (G(x^k) - G(x^*))$$

$$\leq \|x^k - x^*\|^2 - \epsilon_1 \epsilon_2 \frac{G(x^k)^2}{\|t^k\|^2} .$$

Hence $\|x^k - x^*\|$ is monotone decreasing, and thus $G(x^k)^2 / \|t^k\|^2$ tends to zero. From the previous section the formula for $\partial G(x)$ shows that the subgradients of G are uniformly bounded on S; therefore, the limit of $G(x^k) = G(\bar{x}) = 0$ since G is continuous. (It can also be shown that the entire sequence converges to $\bar{x}$.)

Experimentation

To test the two possibilities of minimizing the gap function, we have written an interactive program and solved some small problems. The user inputs the objective function choice, either f or G, the number of Frank-Wolfe iterations to be performed, and the total number of iterations (see Figure 14). The Frank-Wolfe iterations are performed as specified unless the objective value is changing too slowly, or the objective is G and a point of nondifferentiability is reached. In either of these cases, the code will automatically switch to Polyak iterations on G, as it will if neither occurs and the specified number of Frank-Wolfe iterations is exhausted.

First we experimented with Wolfe's example (see Figure 13). The results of five runs are given in Table 6. As expected, the usual Frank-Wolfe iterations converge slowly (Run 1). The most interesting result is the second run. The minimum of G along the Frank-Wolfe direction (y^0-x^0) where $y^0 = (1, 1)$ occurs at $x^1 = (0, 1.3333)$. (This is the point labeled z^k in Figure 1.) From this point the next descent direction is chosen from the set of subgradients of G at x^1. This set is parametrized by points on the lower edge of the triangle, and the (normalized) subgradient of minimum norm is (0, 1). Hence, the next step produces x^*. In effect, the minimization of G, rather than f, in the Frank-Wolfe direction produced a point on the "crease" of nondifferentiability of G. Since x^* is interior to a face of S, it lies at the intersection of this crease and the optimal face.

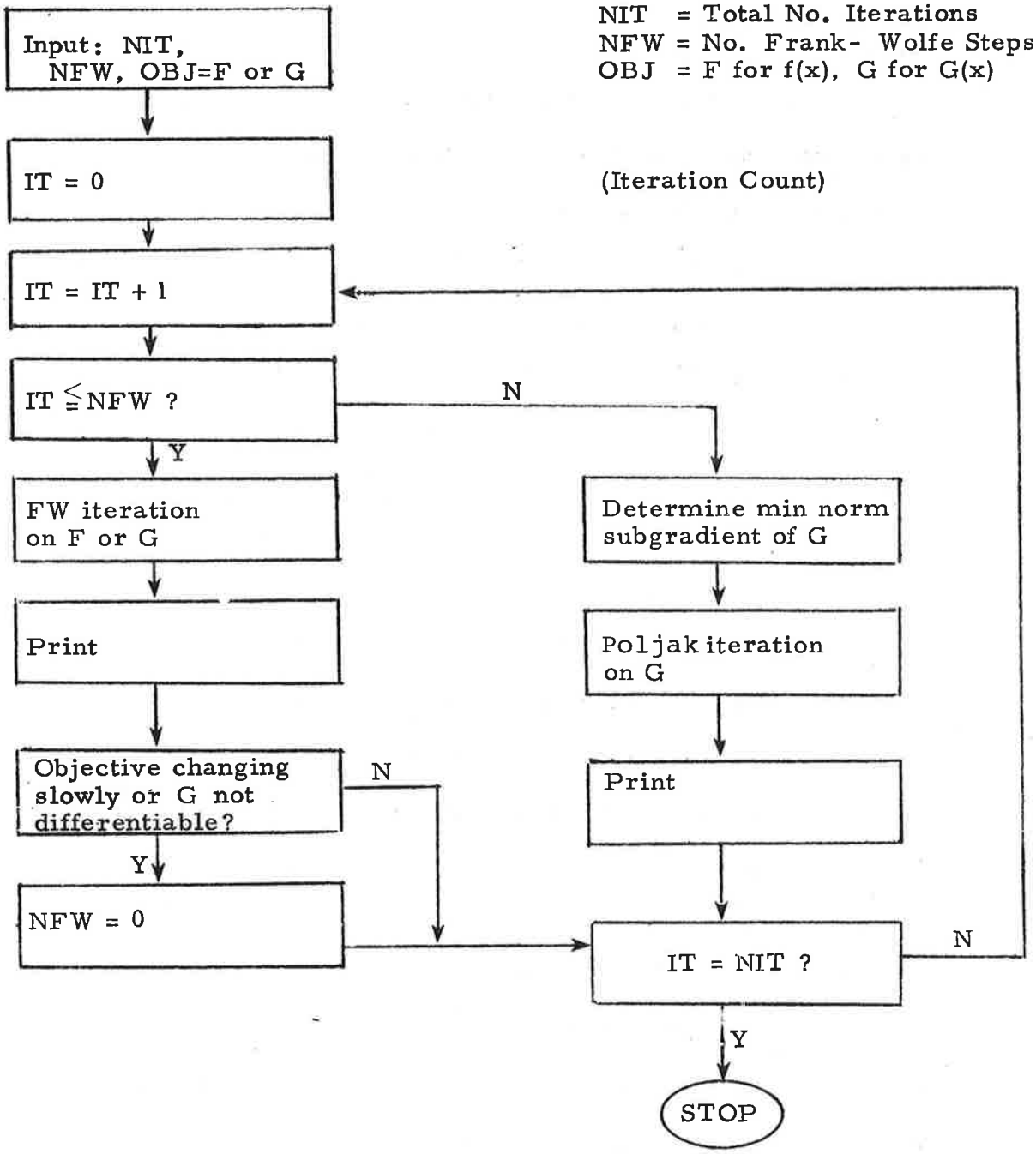


FIGURE 14. FRANK-WOLFE/POLJAK COMBINED FLOW CHART

TABLE 6. EXPERIMENT WITH WOLFE'S EXAMPLE

Run	x^0	Obj.	Method	Results
1	(-0.5, 1.5)	f	FW	Zig-Zags: $f(x^5) = 0.58189$ $x^5 = (-0.0886, 1.0751)$
2	(-0.5, 1.5)	G	FW	Solution in two iterations.
3	(-0.5, 1.5)	G	P	Solution in four iterations.
4	(1, 1)	G	FW	Solution in one iteration
5	(1, 1)	G	P	Near solution in nine iterations: $f(x^9) = 0.500001$ $x^9 = (0.0010, 1.0000)$

NOTES: P=Polyak
FW=Frank-Wolfe
$x^* = (0, 1)$
$f(x^*) = 0.5$

TABLE 7. EXPERIMENTS WITH PROBLEM TWO

Run	x^o	Obj	Method	Result
1	(1, 3)	F	FW	Solution in two iterations
2	(1, 3)	G	P	$f(x^9)$=. 4551585, x^9=(. 8485, . 1015) $G(x^9)$=. 0117842
3	(1, 0)	G	P	$f(x^8)$=. 477063 x^8=(. 9297, . 0550) $G(x^8)$=0. 02489
4	(1, 3)	G	FW/P	Automatic Switch to P after one iteration. $f(x^5)$=. 4545783 $x^5=x^*$, $G(x^5)$=0. 0000623
5	(3, 0)	G	FW/P	Automatic Switch to P after two iterations. $f(x^8)$=. 4545455, $G(x^8)$=0. 0001143 x^8=(0. 9090, 0. 0910)

NOTES: P=Polyak
FW=Frank-Wolfe
x^*=(. 909090. . . , . 090909. . .)
$f(x^*)$=0. 454545. . . .

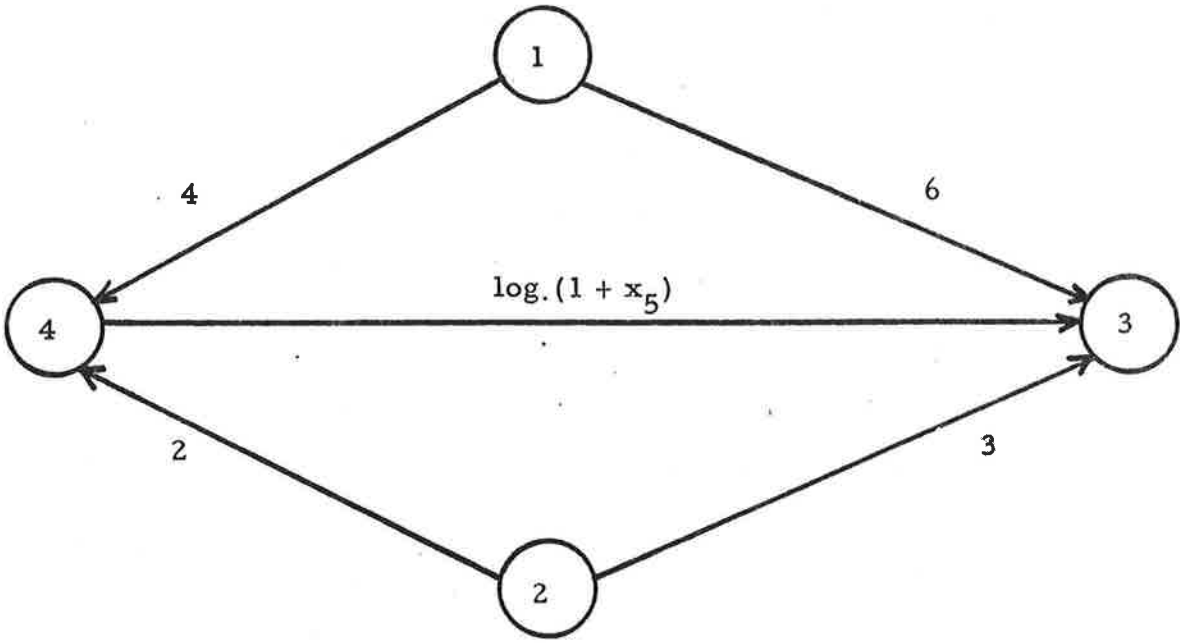
the Polyak algorithm. In Run 4 the crease was found in one Frank-Wolfe iteration, as before, but in Run 5 two iterations were required. In each case a line search along the crease, with either f or G as an objective, would have yielded x^* in just one more iteration.

The third test problem is a four-node traffic assignment problem. This is displayed graphically in Figure 16. The quantity beside each arc is the per unit travel time for the arc. This is constant for all arcs except for arc (4, 3) which has a cost of log $(1 + x_5)$ where x_5 is the total flow on the arc. Trip demands are 5 units from 1 to 3 and 8 units from 2 to 3. Let the arc flows be x_1, x_2, x_3 and x_4 on arcs (1, 4), (1, 3), (2, 4) and (2, 3) respectively. Then it is well known that the convex program which has optimal equilibrium flow is

$$\min \quad 4x_1 + 6x_2 + 2x_3 + 3x_4 + \int_0^{x_5} \log (1 + t)dt$$

$$\begin{aligned} \text{s.t.} \quad & x_1 + x_2 = 5 \\ & x_3 + x_4 = 8 \\ & x_1 + x_3 - x_5 = 0 \\ & x_2 + x_4 + x_5 = 13 \\ & x_1, x_2, x_3, x_4, x_5 \geqq 0. \end{aligned}$$

The optimal solution for this problem is $x^* = (5, 0, 4, 4, 9)$ and it is easily verified that this is an equilibrium solution. Test results for the problem are in Table 8. There are four extreme points for the constraint region of this problem:



Trip Table

	3
1	5
2	8

FIGURE 16. TRAFFIC ASSIGNMENT PROBLEM

TABLE 8. EXPERIMENTS WITH TRAFFIC ASSIGNMENT PROBLEM

Run	x^0	Obj.	Method	Result
1	(0, 5, 0, 8, 0)	F	FW	Solution in two iterations
2	(0, 5, 0, 8, 0)	G	FW/P	Automatic switch to P after two iterations. $G(x^{10}) = 0.039$
3	(5, 0, 8, 0, 13)	G	FW	Solution in one iteration
4	(5, 0, 0, 8, 5)	G	FW	Solution in one iteration
5	(0, 5, 8, 0, 8)	G	FW	Solution in two iterations
6	(0, 5, 0, 8, 0)	G	P	Very slow convergence, $G(x^{20}) = 3.5328$

NOTES: P = Polyak
FW = Frank-Wolfe
x* = (5, 0, 4, 4, 9)

(5, 0, 8, 0, 13)

(5, 0, 0, 8, 5)

(0, 5, 0, 8, 0)

(0, 5, 8, 0, 8),

and these were used as starting points for the six runs. Note that x^* is the midpoint of the line joining the first two extreme points. Because of this fact, runs 3 and 4 required just one iteration, and run 5 required only two. Also, run 1 demonstrates that from the "bad" extreme point, the usual Frank-Wolfe method found the solution in just two iterations. Runs 2 and 6 are the most interesting. In run 6, the pure Polyak method converged very slowly. This is because a projection was required at each iteration (cf. Table 1, Run 5). Run 2 demonstrates that, once again, the most powerful use of the gap function is to perform Frank-Wolfe steps until the crease of nondifferentiability of G is located. Here, two such steps were required to reach a point on the crease. From this point, a line search minimizing either f or G along the crease would produce the solution in one more iteration. In our implementation, minimization down the crease was accomplished by pure Polyak steps, which required a projection at each iteration, and hence yielded only a fair solution in eight more iterations.

Summary

Although the idea of converting a convex program to a minimax problem is hardly new [3], this appears to be the first examination of G(x) for computational purposes. A similar approach has been taken by Oettli [5] who defines a distance function of the primal and dual variables of a linear program. His algorithm, also an adaptation of Polyak's, is shown to have geometric convergence . Our adaptation of Polyak's method differs in that we consider arbitrary convex programs.

Our preliminary results here suggest a number of research possibilities. One is the development of an improved version of Polyak's method with better convergence. Another is to show how to find, in some efficient way, the optimal crease of nondifferentiability for the gap function in general problems. If this can be done, it opens new algorithmic possibilities where the optimization can be restricted to the intersection of this crease and the feasible set. The final product could be an algorithm which does not have the slow convergence properties of the Frank-Wolfe method. Finally, Paper 4 which follows shows that minimization of the gap function by Polyak steps offers the potential for sharply improving bounds in traffic assignment models.

REFERENCES

1. Frank, M. and Wolfe, P., "An Algorithm for Quadratic Programming," NRLQ 3, 95-110 (1956).

2. Hearn, D.W., "Network Aggregation in Transportation Planning Models," Mathtech Final Report, DOT-TSC-RSPD-78-8, I, April 1978.

3. Kuhn, H.W. and Tucker, A.W., "Nonlinear Programming" in Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability. University of California Press, Berkeley, California, 1951.

4. Luenberger, D., Introduction to Linear and Nonlinear Programming, Addison-Wesley, 1973.

5. Oettli, W., "An iterative method, having linear rate of convergence, for solving a pair of dual linear programs," Mathematical Programming, Vol. 3 (1972) No. 3.

6. Poljak, B.T., "Minimization of Unsmooth Functionals," U.S.S.R. Computational Mathematics and Mathematical Physics, Vol. 9, No. 3 (1969) pp 14-29.

7. Wolfe, P., "Convergence in Nonlinear Programming," in Integer and Nonlinear Programming, edited by J. Abadie, North-Holland, 1970.

8. Wolfe, P., "Algorithm for a Least Distance Programming Problem," Mathematical Programming Study 1 (1974) 190-205.

9. Zangwill, W., Nonlinear Programming, Prentice-Hall, 1969.

4. THE MEASURE STEP AND NONLINEAR DUALITY

Donald W. Hearn

Introduction

The Extraction Aggregation Model [1]* outlines the common practices of transportation planners who attempt to reduce computation time by extracting and flowing a subnetwork of a given larger network. The MEASURE step [1] is not implemented in practice, but was added to allow the possibility of feedback in the procedure. This paper addresses theoretical and computational aspects of the MEASURE step from the viewpoint of nonlinear programming. We point out that, if the original problem is a (nonlinear) convex minimization problem, it is necessary to obtain flows for the entire network in order to employ duality theory as a tool in measuring error. This result is a consequence of the nonlinearity of the problem and does not hold if all functions are linear. In addition, we report some computational experiments with obtaining error bounds from arbitrary flows and applying correction steps to those bounds.

Duality - Network Interpretation

We assume that the network flow problem of the Extraction Aggregation Model is of the form

$$(P) \qquad \min \quad f(x)$$
$$\text{s.t.} \quad Ax = b$$
$$x \geqq 0 ,$$

where f is a convex, differentiable function of a vector x, the components of which are flow variables. A and b are a matrix and vector, respec-

*See the overview section of this report.

The dual variables, λ_i^k have the interpretation of path cost or time from node i to destination node k. As indicated in the Figure 1, the feasibility requirement for (D) is that the λ_i^k be less than or equal to the minimum path cost from node i to destination k. This value, however, is a function of the flows along the path.

Therefore, to bound $f(\hat{x})$ by use of the weak duality condition, one must obtain flows $\bar{x}$ (where $\hat{x} = \bar{x}$, possibly) and then define $\bar{\lambda}$ such that the components $\bar{\lambda}_i^k$ are all less than or equal to the path costs defined by those flows.* In a linear problem, of course, the path costs are constants and the dual variables are independent of the flow variables.

It is important to recognize that while $(\bar{\lambda},\bar{x})$ must be dual feasible to provide a lower bound for $f(\hat{x})$, it is not necessary that $\bar{x}$ be feasible to (P). In other words, the components of $\bar{x}$ need not satisfy the conservation of flow constraints. We make use of this fact in the computation experiments reported here.

Duality - Geometric Interpretation

Another useful interpretation of the weak duality result is given in Figure 18. This interpretation, which is independent of network topology, assumes that the problem (P) is a one variable problem, so the only constraints are that the variable x is bounded above and below. The feasible set is shown as a shaded line segment, and $\hat{x}$ is some particular value of x which need not be feasible. Since f is assumed differentiable, it is possible to construct a tangent for f at $\hat{x}$, as shown. Since f is convex, this tangent is below f(x) for all values of x. Thus, the tangent defines a linear function whose minimum value over the feasible set is a lower bound for f. This value is B(x) in the figure.

---

*Strictly speaking, there may be trivial dual solutions that do not require flowing the network, but they lead to very weak bounds. Setting all dual variables equal to a constant, for example, gives a lower bound of zero; valid when the link costs are non-negative.

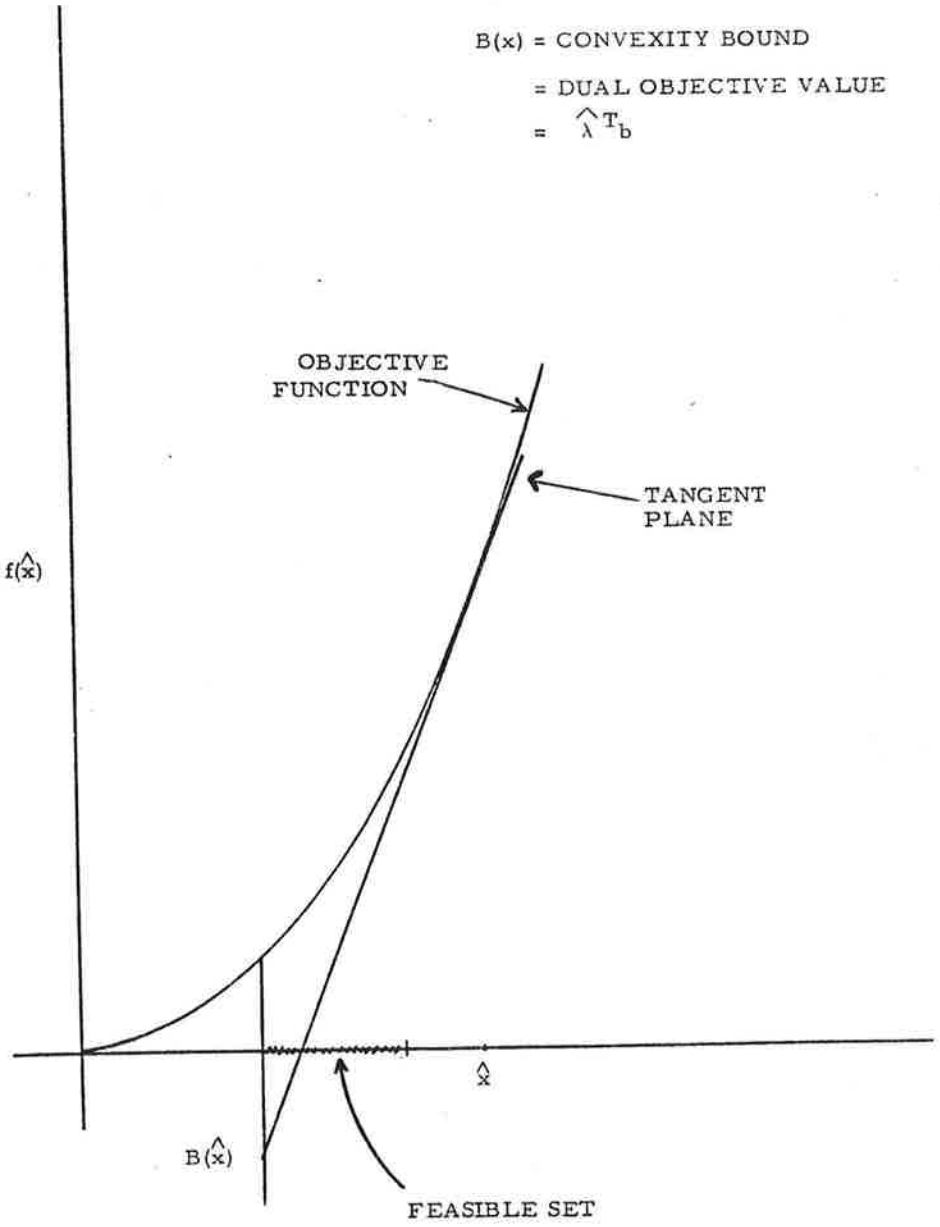


FIGURE 18. GEOMETRIC INTERPRETATION OF CONVEXITY BOUND

In [1], we have proven that the bound obtained by minimizing a tangent to f over the feasible set, which we call the convexity bound, is equal to the best bound obtainable by considering the dual (D) with $x = \hat{x}$. In other words,

$$B(\hat{x}) = f(\hat{x}) - \nabla f(\hat{x})\hat{x} + \max \lambda^T b$$
$$\text{s.t.} \quad \lambda^T A \leq \nabla f(\hat{x}).$$

It is also proven in [1] that the convexity bound is exactly the bound one obtains at each iteration when the Frank-Wolfe algorithm is applied to (P).

Bounding Flow Problems

We summarize the results of the previous sections by listing below several properties relevant to bounding flow problems of the form (P). Mathematical proofs are easily developed from the results in [1]. We assume (P) has a optimal solution x^*, and $f^* = f(x^*)$.

Property 1: If $\hat{x}$ is feasible to (P) and $(\bar{\lambda}, \bar{x})$ is feasible to (D), the inequalities

$$f(\hat{x}) \geq f^* \geq \bar{\lambda}^T \bar{x} + f(\bar{x}) - \nabla f(\bar{x})\bar{x}$$

always hold, regardless of whether $\bar{x}$ is feasible to (P).

Property 2: To obtain both upper and lower bounds on f^*, at least one set of flows feasible to (P) is required.

Property 3: The best (largest) lower bound for f^* obtainable from one set of flows, $\hat{x}$, is $B(\hat{x})$, the convexity bound. If $\hat{x}$ is feasible, $B(\hat{x})$ is also the Frank-Wolfe bound.

Property 4: If several flows, say $x^1, x^2, \cdots\cdots x^k$ are available, then the best available lower bound for f^* is obtained by minimizing the maximum of all tangent planes over the feasible set. (See [1]).

From a computational viewpoint, these results imply one important fact: obtaining a lower bound on f^* requires at a minimum the solution of one linear program (viz. the Frank-Wolfe subproblem). Fortunately, in network flow problems, this reduces to the calculation of shortest paths. In traffic assignment codes, however, it is well known that this is the major computational burden. Thus the price of a bound may be considerable.

Bounds from Infeasible Flows - Computational Experiments

Figure 19 summarizes the results of some computer tests designed to determine relative error in the convexity bounds obtained from arbitrary flows. For test problems, the three problems of Paper 1 in this report were used. These are all traffic assignment problems of the form (P). The link volume delay formulas for these problems are of the standard UMTA and FHWA format

$$T = T_o\left(1. + 0.15\left(\frac{\text{VOLUME}}{\text{CAPACITY}}\right)^4\right)$$

where T_o is the uncongested travel time. To obtain flows, the formula

$$\text{VOLUME} = \beta * \text{CAPACITY}$$

was used for $\beta = 0.0,\ 0.1,\ 0.2,\ \cdots\ 2.0$ and for each link of the network. The convexity bound and its relative error were then computed. Figure 3 shows the results as a function of β.

In the figure, note that the error relative to the optimal solution value for the three problems is approximately 42%, 22% and 19% when $\beta = 0$. Interestingly, these values are all approximately constant for $0 \leq \beta \leq .7$ and then the relative error increases rapidly for larger values of β. The experiments suggest the possibility that bounds under

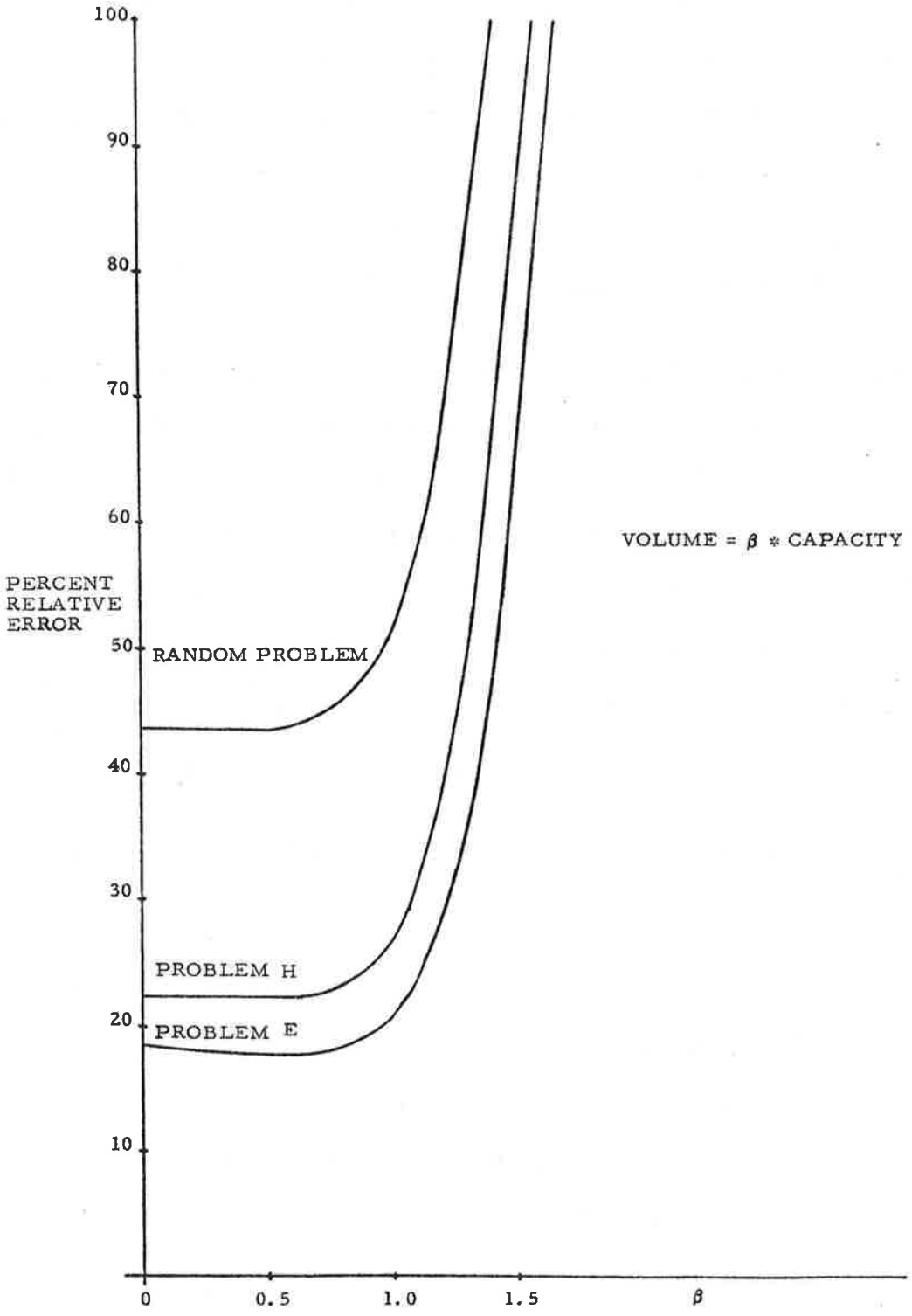


FIGURE 19. ERROR IN THE CONVEXITY BOUND

50% can be obtained by simply assigning zero flow to all network links and then solving the usual subproblem of the Frank-Wolfe algorithm to obtain the convexity bound. By contrast, the bounds produced in early iterations of the Frank-Wolfe method are usually negative, i.e., the relative error is greater than 100%.

Correction Steps for Improving Bounds

Paper 3 of this volume explores the possibility of direct minimization of the duality gap as an alternative method for solving (P). In the notation of this paper the duality gap is

$$G(x) = f(x) - B(x)$$

and has value zero at x^*, the optimal point. One method investigated in Paper 3 is the algorithm of Poljak, a recursion based on the formula

$$x^{k+1} = P_S\,(x^k - \rho_k\, t^k)$$

$$\rho_k = \gamma_k \frac{G(x^k)}{\|t^k\|^2}$$

$$0 < \gamma < 2$$

where P_S is the projection operator for the feasible set of problem (P) and where t^k is a subgradient of G at x^k. It can be proven that this recursion always generates a point x^{k+1} nearer the set of points where $G(x) = 0$ than the prior point x^k. This is represented symbolically in Figure 20. The feasible set is a pentagon and the locus of points for which $G(x) = 0$ is represented as a closed curve. These sets intersect at the optimal point, x^*. The point x^{k+1} represents a move in the direction t^k from the point x^k. Since t^k points toward the closed curve, x^{k+1} is nearer the curve (even without the projection onto S).

We have employed the recursion above, with projection only on the constraints $x \geqq 0$ of (P), to obtain "corrections" to the lower bounds

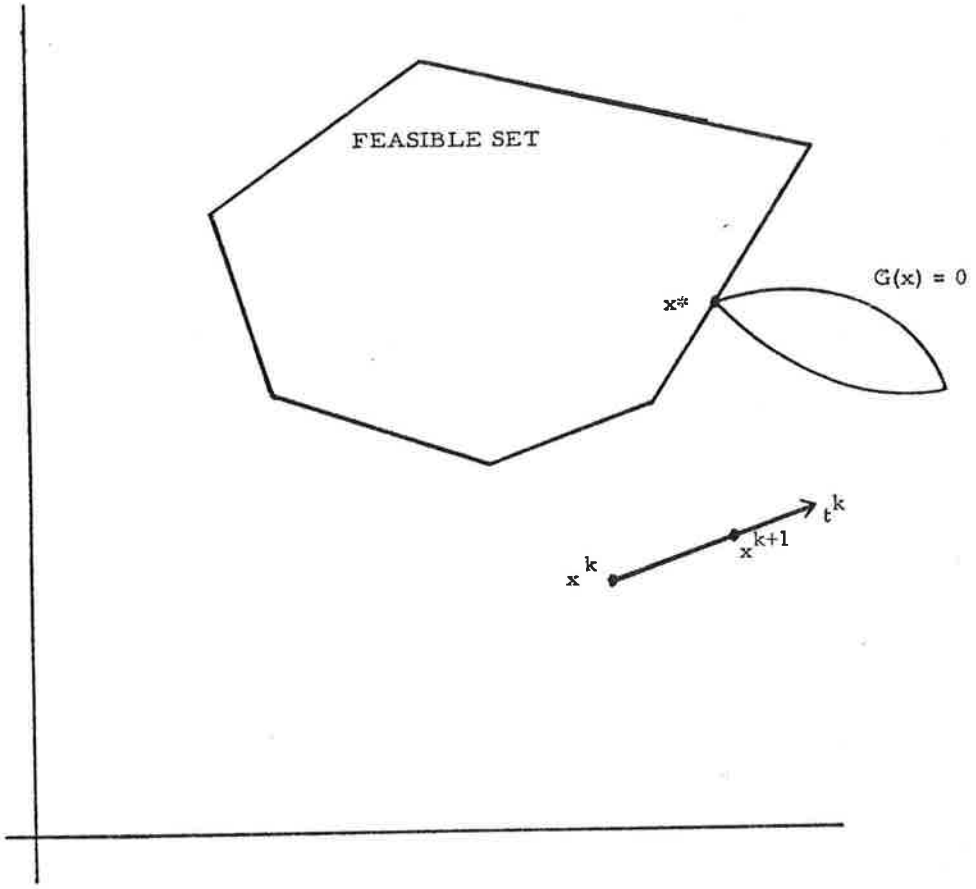


FIGURE 20. POLJAK CORRECTION STEP

obtained in the experiments of the previous section. These steps are corrections in the sense that a decreasing G can result in an increase in the bound B(x), if the value of f(x) does not decrease even more. The biggest question is what value to choose for γ , i.e., how far should the step in the direction of t be.

In Figure 21, the relative error in the convexity bound is shown after correction by the best value for γ in the range $0<\gamma \leqq 2$ (c f. Figure 18). Note that for small values of β, the correction step gives no improvement. However, for β near 1, there is a sharp improvement in the relative error when the best correction step is made. On all three test problems, a bound of 10%-15% is obtained.

To further test the validity of correction steps for improving the convexity bound, the correction steps were applied to the bounds normally produced by the Frank-Wolfe algorithm. Every iterate of this method produces feasible flows, but the correction steps, based on the recursion and projection just onto $x \geqq 0$, can produce infeasible flows to obtain the bound. A summary of the results for one of the problems is in Table 9. The optimal value for this problem is known to be between 1720 and 1700 (see Paper 1, Table 4) and the correction step produced a bound within 3% of this range at the third iteration, whereas the bound without correction did not achieve this error until the fifth iteration.

While the above test makes it appear reasonable to add correction steps to the Frank-Wolfe method, one must be aware that the computation cost of obtaining the correction is approximately the same as the cost of an iteration of the Frank-Wolfe method. Thus, seven subproblems are solved before obtaining the bound of 1657 (with corrections) while a better bound of 1649 (without corrections) is obtained after solving just six Frank-Wolfe subproblems.

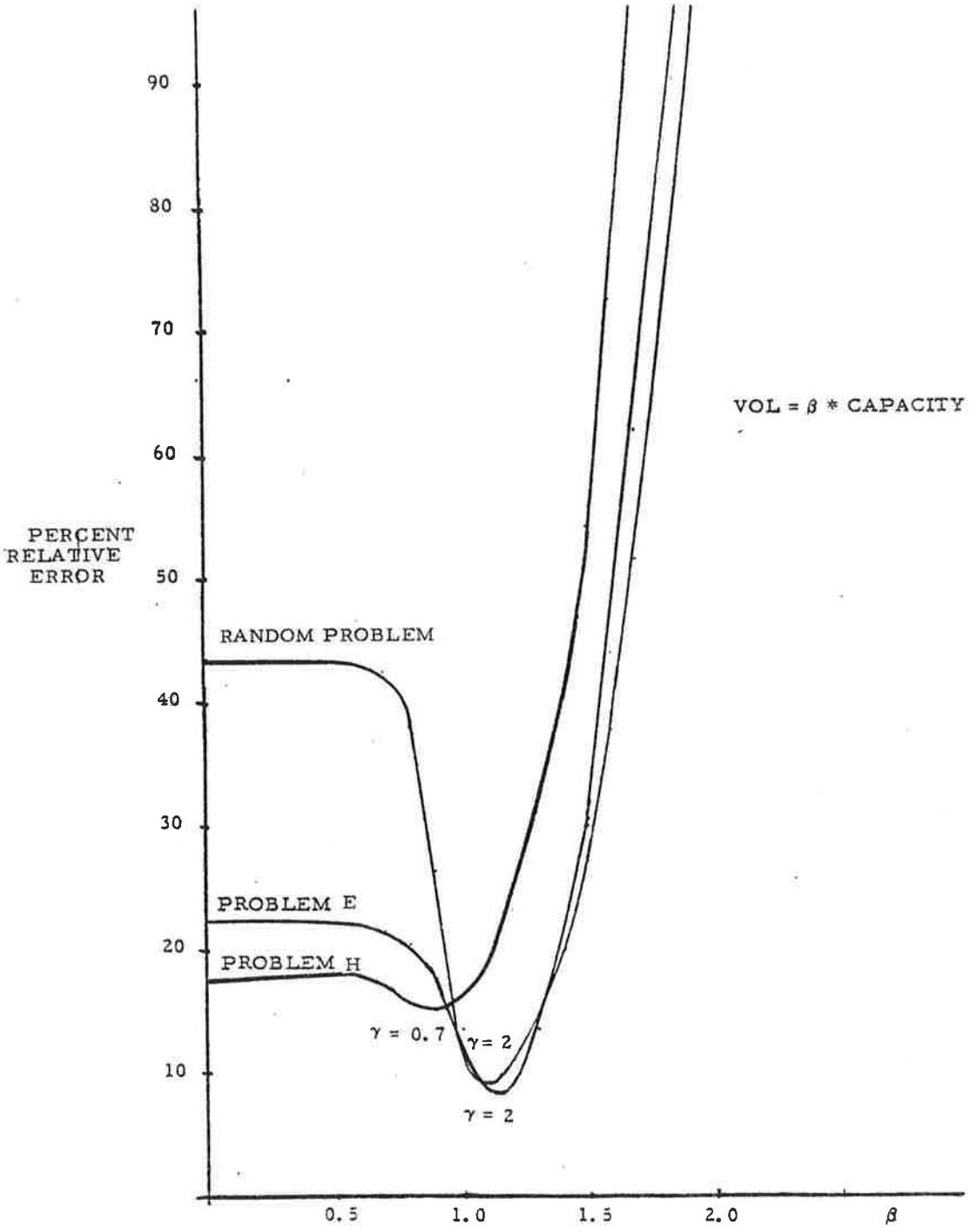


FIGURE 21. ERROR IN THE CONVEXITY BOUND AFTER BEST POLJAK CORRECTION STEP

TABLE 9. ADDITION OF POLJAK CORRECTION TO F W

(PROBLEM E)

IT.	BEST CONVEXITY BOUND -- NO CORRECTION	NO. FW SUBPROBLEMS SOLVED	BEST BOUND WITH POLYAK CORRECTION	NO. FW SUBPROBLEMS SOLVED
0	-	1	-	1
1	-17010	2	-368	3
2	351	3	1540	5
3	1585	4	1657	7
4	1586	5	1657	9
5	1649	6	1657	11
6	1649	7	1657	13
7	1649	8	1680	15
8	1653	9	1680	17
9	1673	10	1682	19
10	1673	11	1682	21
11	1677	12	1682	23
12	1677	13	1690	25
13	1677	14	1690	27
14	1677	15	1694	29
15	1688	16	1694	31
16	1688	17	1694	33
17	1688	18	1694	35
18	1688	19	1694	37
19	1693	20	1694	39
20	1693	21	1696	41

Note: $1720 \geq f^* \geq 1700$

Conclusions

This paper defines limits, in a certain sense, to which one might go in using nonlinear duality to bound error in the MEASURE step of the Extraction Aggregation Model. It complements and extends the results in (1) where we proved that various error measures for the traffic assignment problem are equivalent. To summarize, the results are:

a) At least one set of flows for the disaggregated problem (P) must be obtained in order to evaluate the dual objective value. This property results from the assumption that this problem (P) has a nonlinear objective function, and does not hold for linear problems.

b) The computation cost for obtaining a bound is at least the cost of solving a linear program with the same constraints as the disaggregated problem. For flow problems, this reduces to a sequence of one or more shortest path problems, but this may still be a nontrivial cost for large problems such as the traffic assignment problems.

c) Bounds may be obtained for arbitrary flows which are not feasible, and at the cost of additional computation, can be significantly improved by Poljak correction steps.

d) The correction steps can be added to standard production computer codes much as UROAD with ease, but the cost of computation per iteration doubles (approximately). Heuristic transfer and flow methods for the Extraction Aggregation model would likely involve finding flows feasible to the extracted subnetwork only. The above results imply that obtaining a bound would require assigning flows to the remainder of the network, but that they need not be feasible.

Of course any production use of these results would require testing to establish values of the parameters α, β and γ.

REFERENCE

1. Hearn, D.W., "Network Aggregation in Transportation Planning Models, Part I." Mathtech Final Report, DOT-TSC-RSPD-78-8,I, April, 1978.

5. EQUIVALENT FLOWS FOR EXTRACTED SUBNETWORKS

Russell R. Barton

Introduction

Transportation planners are frequently interested in estimating user equilibrium traffic flows for a large network $(\mathcal{N}, \mathcal{A})$, where link travel time is a convex function of link flow. These flows may be difficult, if not impossible, to find if the network is very large. Extraction aggregation* can be helpful in solving such problems for two reasons. First, the planner's main interest often lies in estimating flows on a subset $(\mathcal{N}, \mathcal{A})$ of the complete network. Second, Hearn [1977] showed that one may be able to get a good solution to the complete problem more quickly by first solving for flows on an extracted subnetwork. When the subnetwork is of special interest (i.e., the first reason) we would like to find a solution that matches the flow values one would obtain in solving the complete problem. If one is attempting to solve the complete problem quickly using an extracted network solution, it may or may not be advantageous to have the extracted solution match flows; perhaps some other characteristic of the extracted solution will be more important in finding a good solution to the complete problem.

In the following sections we consider the relationship between equilibrium solutions obtained for the extracted subnetwork and equilibrium solutions for the complete network. We will be particularly interested in solutions to the subnetwork that yield flows equivalent to the complete network flows on the extracted arcs. We show that, to solve this problem, one must flow the extracted network without knowing the trip table.

---

*See the overview section of full problem.

1. Problem Definition

The equilibrium traffic assignment problem can be described as a minimum cost multicommodity flow problem, where the commodity pq corresponds to trips from node p to node q. One in fact need only define separate commodities by destination nodes (or origin nodes) as is done below, but the fuller separation is sometimes useful. We assume that the complete network has 'n' nodes, 'a' directed arcs, and 'k' origin/destination centroids, while the extracted subnetwork has 'n' nodes, $\underline{a}$ arcs, and $\underline{k}$ origin/destination centroids. For simplicity in exposition (and as a necessity for the subnetwork that yields equivalent flows) we assume that all nodes are centroids, i.e., $n = k$ and $\underline{n} = \underline{k}$. This causes no loss in generality since positive link travel times imply zero optimal flows for commodities having no demand.

We index the nodes and arcs of the networks so that the extracted subsets consist of the first $\underline{n}$ and $\underline{a}$ indices. We will label arcs by their nodes so that arc ij is a directed arc from node i to node j. This label is in addition to the index of the arc. We let $P = \left\{pq \mid p \leq n,\ q \leq n\right\}$ and $\underline{P} = \left\{pq \mid p \leq \underline{n},\ q \leq \underline{n}\right\}$ be the index sets for the O-D pairs of the complete and extracted networks, respectively. Similarly we let $Q = \left\{q \mid q \leq n\right\}$ and $\underline{Q} = \left\{q \mid q \leq \underline{n}\right\}$ be the index sets for the destination centroids. Total demand for trips from p to q (in the complete problem) is given by z^o_{pq}.

The traversal time for link ij $\left(c_{ij}(\cdot)\right)$ is assumed to be a function of total link flow (f_{ij}) such that $\int_0^{f_{ij}} c_{ij}(u)du$ is convex in f_{ij}.

We can write the user equilibrium traffic assignment problem for the complete network as

P1 minimize $\sum_{ij \in \mathcal{A}} \int_0^{f_{ij}} c_{ij}(u)du$

s.t. $f_{ij} = \sum_{q \in Q} x_{ij}^q$

$Ax^q = b^q \qquad \forall q \in Q$

$x \geq 0$,

where A is the nxa node arc incidence matrix for $(\mathcal{N}, \mathcal{A})$. The b^q vector is a net flow n-tuple whose components sum to zero, and is defined by:

$$b_j^q = -\sum_{pq \in P} z_{pq}^o \qquad j = q$$

$$= z_{jq}^o \qquad \text{otherwise.}$$

The x^q vector is an a-tuple where x_{ij}^q gives flow of commodity q on arc ij.

By our indexing scheme we can write A as

$$A = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} \qquad (1.1)$$

where the partition is after the $\underline{n}^{th}$ row and $\underline{a}^{th}$ column, so that A_{11} is the node arc incidence matrix for the extracted subnetwork $(\underline{\mathcal{N}}, \underline{\mathcal{A}})$. The lower left submatrix is zero because we do not allow extracted arcs to be incident to deleted nodes. The upper right submatrix need not be zero, since deleted arcs may be incident to extracted nodes. Optimal flows are given by f_{ij}^*.

2. Extracted Network Equilibrium Flows

We write the user equilibrium traffic assignment problem for the extracted network as

$$\boxed{P2(\underline{z})} \quad \text{minimize} \quad \sum_{ij \in \underline{a}} \int_0^{\underline{f}_{ij}} c_{ij}(u)du$$

$$\text{s.t.} \quad \underline{f}_{ij} = \sum_{q \in \underline{Q}} \underline{z}_{ij}^q$$

$$A_{11}\, \underline{x}^q = \underline{b}^q \qquad \forall\, q \in \underline{Q}$$

$$\underline{x} \geq 0.$$

We are interested in conditions under which $\underline{f}_{ij}^* = f_{ij}^* \quad \forall_{ij} \in \underline{a}$. We have given the trip demands, $\underline{z}_{pq}$ for this problem as separate unknown variables. This is because, as the example below illustrates, setting $z_{pq} = z_{pq}^o$ does not guarantee generally that $\underline{f}_{ij}^* = f_{ij}^* \quad \forall_{ij} \in \underline{a}$.

Example 1.1

Consider the complete network below, where the extracted network is in bold ink.

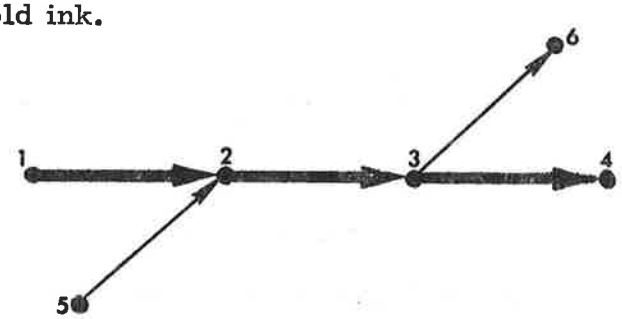


FIGURE 22. NETWORK FOR EXAMPLE 1.1

The nodes have been numbered in accordance with our assumptions above. If we let $z^o_{14} = d_1 > 0$, $z^o_{56} = d_2 > 0$ and all other z^o_{pq} values be zero, then

$$\underline{z}_{pq} = z^o_{pq} \Longrightarrow f^*_{23} \left(= d_1 + d_2\right) \neq f^*_{23} \left(= d_1\right)$$

In the next section we show that there exists at least one set of trip demands for the extracted problem (z^*_{ij}) that give $\underline{f}^*_{ij} = f^*_{ij} \quad \forall_{ij} \in \underline{a}$. As can be seen in Example 1.1, it may be necessary to allow new pairs of nodes to have nonzero trip demands, i.e., for $pq \in \underline{P}$:

Proposition 1.1. $\quad z_{pq} = 0 \not\Longrightarrow \underline{z}^*_{pq} = 0$

Furthermore, we have

Proposition 1.2. $\quad z_{pq} \neq 0 \not\Longrightarrow \underline{z}^*_{pq} \neq 0$

Proof. By the following example:

Example 1.2

For the network below, with bold lines for extracted links, we see that if $c_{23}(x) = c_{25}(x) + c_{53}(x)$, and if $z^o_{14} = d_1 > 0$, $z^o_{23} = d_1 > 0$ with all other z values zero, then we can get $f^*_{23} = \underline{f}^*_{23}$ with $\underline{z}_{14} = 0$ or $\underline{z}_{23} = 0$ (or both).

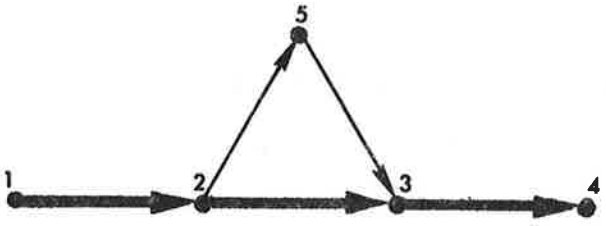


FIGURE 23. NETWORK FOR EXAMPLE 1.2

3. Existence of Equivalent Equilibrium Flows

We will show that, if one allows any node in $\underline{n}$ to be an origin/destination centroid for P2, there exists at least one set of $\underline{z}_{pq}$'s that give $\underline{f}^*_{ij} = f^*_{ij} \quad \forall_{ij} \in \underline{a}$. Our approach is to augment the components z^o_{pq}, $pq \in \underline{P}$ by P1 equilibrium flows corresponding to O-D pairs $p'q'$ not in $\underline{P}$. We will assume for the proof that once an equilibrium trip leaves the extracted subnetwork it does not return to the extracted subnetwork (e.g., example 1.2 is disallowed). The theorem can be proved without this restriction using a different construction for the $\underline{z}_{pq}$'s.[1]

We denote a P1-equilibrium flow on $ij \in \underline{a}$ corresponding to C1 (or C3) by $x^*_{ij}(q, C1)$ (or $x^*_{ij}(q, C3)$). The theorem on existence of flows will be proved by defining $\underline{z}^*$, $\underline{x}^*$ such that $\underline{f}^*_{ij} = f^*_{ij}$ and showing that $\underline{x}^*$ is feasible and optimal for P2($\underline{z}^*$).

We give

$$\underline{z}^*_{pq} = z^o_{pq} + \sum_{r \notin \underline{n}} \left(\begin{array}{l} \text{P1-equilibrium flows from p to r} \\ \text{that leave the subnetwork at node q} \end{array} \right) \qquad \text{(C1)}$$

$$+ \sum_{r \notin \underline{n}} \left(\begin{array}{l} \text{P1-equilibrium flows from r to q} \\ \text{that enter the subnetwork at p} \end{array} \right) \qquad \text{(C2)}$$

$$+ \sum_{r \notin \underline{n}} \sum_{s \notin \underline{n}} \left(\begin{array}{l} \text{P1-equilibrium flows from r} \\ \text{to s that enter the subnetwork} \\ \text{at p and leave at q} \end{array} \right). \qquad \text{(C3)}$$

The three cases leading to augmented flow are illustrated in Figure 1.3.

---

[1] For example, if we use $\underline{z}_{ij} = f^*_{ij} \; \forall_{ij} \in \underline{a}$, the existence of equivalent equilibrium flow is easy to prove. We feel, however, that this definition of subnetwork trip demands is less meaningful.

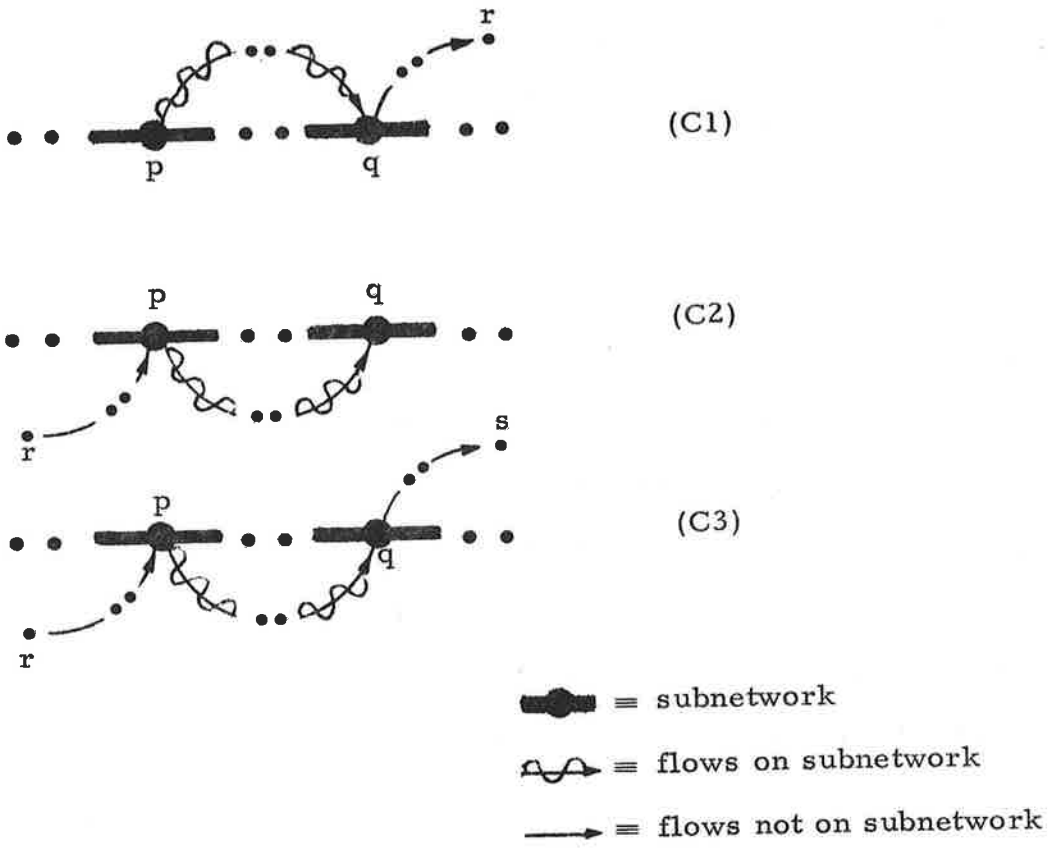


FIGURE 24. THREE TYPES OF EQUILIBRIUM FLOW THAT AUGMENT z^{o}_{pq} IN DEFINING z^{*}_{pq}

Theorem 1.1

There exists at least one set of trip demands $\underline{z}^{*}$ for P2($\underline{z}$) such that $f_{ij}^{*} = \underline{f}_{ij}^{*} \quad \forall_{ij} \in \underline{a}$.

Proof:

Choose $\underline{z}_{pq}^{*}$ as above and let

$$\underline{x}_{ij}^{*q} = x_{ij}^{*q} + x_{ij}^{*}(q, C1) + x_{ij}^{*}(q, C3), \tag{1.2}$$

for any x^{*} that is P1-optimal.

Feasibility of $\underline{x}_{ij}^{*q}$ for P ($\underline{z}^*$) is implied by the definition of $\underline{z}^*$ and the P1-feasibility of the original flows x^*. We see from (1, 2) that

$$\underline{f}_{ij}^{*} = \sum_{q \in Q} \underline{x}_{ij}^{*q} = \sum_{q \in Q} x_{ij}^{*q} + \sum_{s \in Q/\underline{Q}} x_{ij}^{*s}$$

$$= f_{ij}^{*}.$$

Now if $\underline{x}^*$ is not optimal for P2($\underline{z}^*$) then for some pqϵ $\underline{P}$ there exists a flow α>o on a non-shortest path ℓ from p to q. For some δ>o one can decrease the objective function of P2($\underline{z}^*$) by transferring δ percent of the flow (from p to q) on ℓ to a shortest path ℓ^*. By the construction of $\underline{x}^*$, a similar change can be made to the original flows, where for flows of type C1, C2, and C3 only the p - q portion of the path is rerouted. This operation implies that x^* is not P1-optimal, a contradiction, and so $\underline{x}^*$ is optimal for P2($\underline{z}^*$). □

The proof above applies to particular optimal solutions. When there are alternative optimal flows for the complete problem, the subnetwork optimal flows will match the flow vector f^* that was used to construct the $\underline{z}_{ij}^*$. It need not be true that there will be alternative optimal flows for the subproblem. Using the method above for constructing the $\underline{z}_{ij}^*$'s, alternative subnetwork optimal flows will imply that there are alternative complete network optimal flows. For different constructions of $\underline{z}_{ij}^*$ this need not be true. As the following example shows, it is possible that there is a $\underline{z}^*$ such that there are some P2 optimal solutions that are not P1 optimal, while all P1 optimal solutions are P2 optimal.

Example 1.3

We assume that the entire network below is extracted that $c_{13}(x) = c_{12}(x) + c_{23}(x) \quad \forall x \geq 0$, and that $z_{12}^{o} = z_{23}^{o} = 1$. If we let $\underline{z}_{13} = 1$ and all other $\underline{z}$ values be zero, then we get

$$\left\{ \begin{pmatrix} f_{12}^{*} = 1 \\ f_{23}^{*} = 1 \\ f_{13}^{*} = 0 \end{pmatrix} \right\} \subset \left\{ \begin{pmatrix} \underline{f}_{12}^{*} = 1 \\ \underline{f}_{23}^{*} = 1 \\ \underline{f}_{13}^{*} = 0 \end{pmatrix}, \begin{pmatrix} \underline{f}_{12}^{*} = 0 \\ \underline{f}_{23}^{*} = 0 \\ \underline{f}_{13}^{*} = 1 \end{pmatrix} \right\}.$$

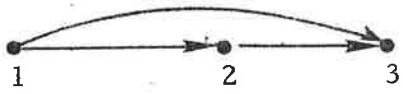


FIGURE 25. NETWORK FOR EXAMPLE 1.3

The above results show that, if we can assume no equilibrium flows return to the extracted subnetwork, that there is an intuitively appealing augmentation of z_{pq}^{o} that gives $f_{ij}^{*} = \underline{f}_{ij}^{*}$. As we have mentioned, this can be shown for another definition of $\underline{z}_{pq}^{*}$, allowing equilibrium flows to leave and return to the subnetwork.

4. Flowing the Extracted Network Without a Trip Table

Unfortunately, calculation of these $\underline{z}_{pq}^{*}$ vectors requires knowledge of the P1-optimal flows x_{ij}^{q*}. A recent theorem by Nguyen [1977] shows that one can calculate $\underline{x}_{ij}^{q*}$ and $\underline{z}_{pq}$ simultaneously that yield $f_{ij}^{*} = \underline{f}_{ij}^{*}$, provided one knows equilibrium travel times, μ_{pq}, for all pairs $pq \in \underline{P}$. This fact is actually a corollary of the main theorem, which shows that one can find equilibrium flows for a general network $(\mathcal{N},\mathcal{A})$ if one knows the equilibrium travel times, μ_{pq}, between all pairs $pq \in P$ for which $z_{pq}^{o} > 0$.

The form of the Nguyen problem, which is a special case of problem (M2) of Hearn [1977] is given below:

P3 $$\text{minimize}\left(\sum_{ij\in a}\int^{f_{ij}} c_{ij}(u)\,du\right) - \mu z$$

$$\text{s.t.} \quad f_{ij} = \sum_{q} x_{ij}^{q} \qquad (\eta) \qquad (1.3)$$

$$B_x = T_z \quad \forall\, q \in Q \qquad (\lambda) \qquad (1.4)$$

$$x, z \geq 0 \qquad (u, v) \qquad (1.5)$$

B is a block diagonal matrix of node arc incidence matrices, and T is a block diagonal matrix such that $Tz = b$. The q^{th} block, T^q, is an $n \times n-1$ matrix of the form:

$$T^q = \begin{bmatrix} 1 & & & & \\ & 1 & \ddots & 1 & \\ -1 & -1 & \cdots & -1 & -1 \cdots -1 \\ & & & 1 & \ddots 1 \end{bmatrix} \leftarrow q^{th} \text{ row}$$

We give two lemmas relating to the form of P3 that will be used to prove the theorem.

Lemma 1.1 $T^T T$ is full rank

Proof: T^q is $n \times n-1$ and clearly of rank n-1. Therefore, $T^{q\,T} T^q$ is $(n-1)\times(n-1)$ and of rank n-1 $\forall q$. This in turn implies $T^T T$ is full rank. □

Lemma 1.2 Suppose $F(x, y) = t(y) + g(x)$ where g is convex in x, and suppose that t is strictly convex in y. Suppose that y can be expressed as $y = \varphi(x)$. If φ is linear, then we can say that the space of points satisfying $y = \varphi(x)$ is a linear subspace. If C is a convex set in that subspace, then the problem

$$\min_{(x, y) \in C} \quad F(x, y),$$

has a unique solution in y, that is, $\exists\, y^*, \varphi(x) = y^*$ for any optimal x.

Proof: Suppose there are two optimal y values, y_1^* and y_2^*. Corresponding to these values we have two disjoint sets of x values, say X_1^* and X_2^*. They are disjoint because $y_1^* \neq y_2^*$ and $x_1 = x_2 \Longrightarrow \varphi(x_1) = \varphi(x_2)$. Choose an element arbitrarily from X_1^*, say x_1^*, and from X_2^*, say x_2^*. Since C is convex, $(x_1^*, y_1^*), (x_2^*, y_2^*) \in C \Longrightarrow (x_3, y_3) \in C$ for $(x_3, y_3) = \alpha(x_1^*, y_1^*) + (1-\alpha)(x_2^*, y_2^*)$, $0 < \alpha < 1$. Since (x_1^*, y_1^*) and (x_2^*, y_2^*) are optimal, we have:

$$F(x_3, y_3) \geq F(x_1^*, y_1^*) = F(x_2^*, y_2^*)$$

so $$F(\alpha x^* + (1-\alpha)x_2^*, y_1^* + (1-\alpha)y_2^*) \geq \alpha F(x_1^*, y_1^*) + (1-\alpha)F(x_1^*, y_1^*)$$

or $$t(\alpha y_1^* + (1-\alpha)y_2^*) + g(\alpha x_1^* + (1-\alpha)x_2^*) \geq$$
$$\alpha t(y_1^*) + (1-\alpha)\, t(y_2^*) + \alpha g(x_1^*) + (1-\alpha)\, g(x_2^*).$$

Thus one of the following must be true:

A: $$t(\alpha y_1^* + (1-\alpha)y_2^*) > \alpha t(y_1^*) + (1-\alpha)\, t(y_2^*)$$

B: $$g(\alpha x_1^* + (1-\alpha)x_2^*) > \alpha\, g(x_1^*) + (1-\alpha)\, g(x_2^*)$$

C: $$t(y_1^* + (1-\alpha)y_2^*) = t(y_1^*) + (1-\alpha)t(y_2^*) \text{ and } g(\alpha x_1^* + (1-\alpha)x_2^*) =$$
$$\alpha\, g(x_1^*) + (1-\alpha)\, g(x_2^*).$$

But

$A \Longrightarrow$ t strictly concave over the region between y_1^* and y_2^*

$B \Longrightarrow$ g strictly concave over the region between x_1^* and x_2^*

$C \Longrightarrow$ t not strictly convex over the region between y_1^* and y_2^*

□

We turn now to the theorem by Nguyen, which shows that one can find equilibrium flows without knowing the trip table $\left[z_{pq}\right]$ if one knows the equilibrium travel times, $\left[u_{pq}\right]$.

Theorem 1.2 (Nguyen [1977])

For P3, let μ_{pq} be the (P1) equilibrium travel time from p to q $\forall pq \in P$. Then if $\int_0^x c_{ij}(u)du$ is strictly convex in $x \forall ij \in A$, P1 and P3 have identical solutions in f_{ij} (i.e. $(f_{ij}^*) = (f_{ij}^*)$) and x^* is a solution to P1 $\Longrightarrow (x^*, z^o)$ is a solution to P3.

Proof: Nguyen's proof is for the arc-chain formulation of the user equilibrium traffic assignment problem. We give here an alternate proof based on the node arc (i.e. P1) formulation of the problem.

We first show that x^*, z^o satisfies the Kuhn-Tucker conditions for P3, which by the convexity of the objective function is sufficient for P3-optimality. We then show that by Lemmas 1.1 and 1.2, the P3-optimal value for f_{ij} is unique and thus identical to that for P1.

The Kuhn-Tucker conditions for P3 are:

$$\nabla f - B^T\lambda - D\gamma - u = 0 \qquad (1.6)$$

$$x \cdot u = 0 \qquad (1.7)$$

$$u \geq 0 \qquad (1.8)$$

$$\mu + T^T\lambda - v = 0 \qquad (1.9)$$

$$z \cdot v = 0 \qquad (1.10)$$

$$v \geq 0 \qquad (1.11)$$

along with feasibility conditions (1.3) - (1.5), where D is defined so that (1.3) reads f=Dx.

We see that $\mu^*, x^*, z^o, u^*, \lambda^*$ from P1 satisfy (1.3) - (1.8) since these are the Kuhn-Tucker conditions for P1. If we set

$$v_{pq} = 0 \text{ for } z_{pq}^{\,o} > 0$$

$$= \mu_{pq}^{\,*} + T_{pq} \cdot \lambda^* \text{ for } z_{pq}^{\,o} = 0$$

where T_{pq} is the column of T corresponding to z_{pq}, then (1.10) holds. Furthermore, by (4-16) of Hearn [1977] ($s_k^{\,i} \equiv \mu_{ki}$ $\lambda_k^{\,i} \equiv \lambda_{ki}$) we see that

$$T_{pq} \cdot \lambda = \lambda_{qq} - \lambda_{pq} \geq -\mu_{pq}$$

so that v_{ij} is nonnegative $\forall_{pq} \epsilon P$. It only remains to show that (1.9) holds for $z_{pq}^{\,o} > 0$. But this is just the requirement that

$$\lambda_{pq} - \lambda_{qq} = \mu_{pq}^{\,*} \qquad \forall_{pq} \ni z_{pq} > 0$$

which Hearn [1977] shows holds for λ^* for P1.

The uniqueness of $f_{ij}^{\,*}$ for P3 follows from lemmas 1.1 and 1.2. Lemma 1.1 says we can write

$$\lambda z = \lambda \left((T^T T)^{-1} T^T B \right) x \equiv g(x) \qquad (1.12),$$

which is convex (linear). We recall that

$$\sum_{ij \epsilon Q} \int_0^{f_{ij}} c_{ij}(u)\, du \equiv t(f) \qquad (1.13),$$

is strictly convex in f, and

$$f_{ij} = \sum_{q \epsilon Q} x_{ij}^{\,q} \equiv \gamma(x) \qquad (1.14),$$

where γ is a linear function. Then (1.12) - (1.14) imply that lemma 1.2 holds, and so f^* is unique for P3. □

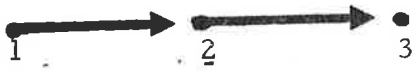
We note that the requirement μ_{pq}^{*} = Pl-optimal travel times $\forall_{pq} \epsilon P$ can be relaxed to μ_{pq}^{*} = Pl optimal travel times $\forall_{pq}$ such that $z_{pq}^{o} > 0$, provided the remaining μ_{pq}^{*} values are set to suitably large positive values for (1.9) and (1.11) to hold. This relaxation is not possible if one is looking at an extracted subnetwork, since $z_{pq}^{o} = 0 \not\Longrightarrow \underline{z}_{pq} = 0$. This problem limits the usefulness of Theorem 1.2 in finding equivalent flows for an extracted subnetwork.

A further problem arises from the fact that even if $\int_{o}^{f} c(u)du$ is strictly convex for each link, the optimal z_{pq} values need not be unique, as illustrated below:

Example 1.4 For the network below, any convex combination of the sets

$$\left\{ \begin{matrix} x_{12}^{*} = 2 & z_{12} = 2 \\ x_{13}^{*} = 2 & z_{23} = 2 \\ & z_{13} = 0 \end{matrix} \right\} \qquad \left\{ \begin{matrix} x_{12}^{*} = 2 & z_{12} = 0 \\ x_{13}^{*} = 2 & z_{23} = 0 \\ & z_{13} = 2 \end{matrix} \right\}$$

give optimal solutions to P3.



$c_{12}(x) = c_{23}(x) = x + 1$

$x_{12}^{*} = x_{23}^{*} = 2$

FIGURE 26. NETWORK FOR EXAMPLE 1.4

In the following example we find equilibrium flows for a network with linear travel time functions. The variable trip table problem P3 is solved to find equivalent flows for a subnetwork.

Example 1.5

For the network above, we have a linear arc travel time function:

$$c(u) = c_0 + c_1 u$$

where c_0 and c_1 are given for each arc as:

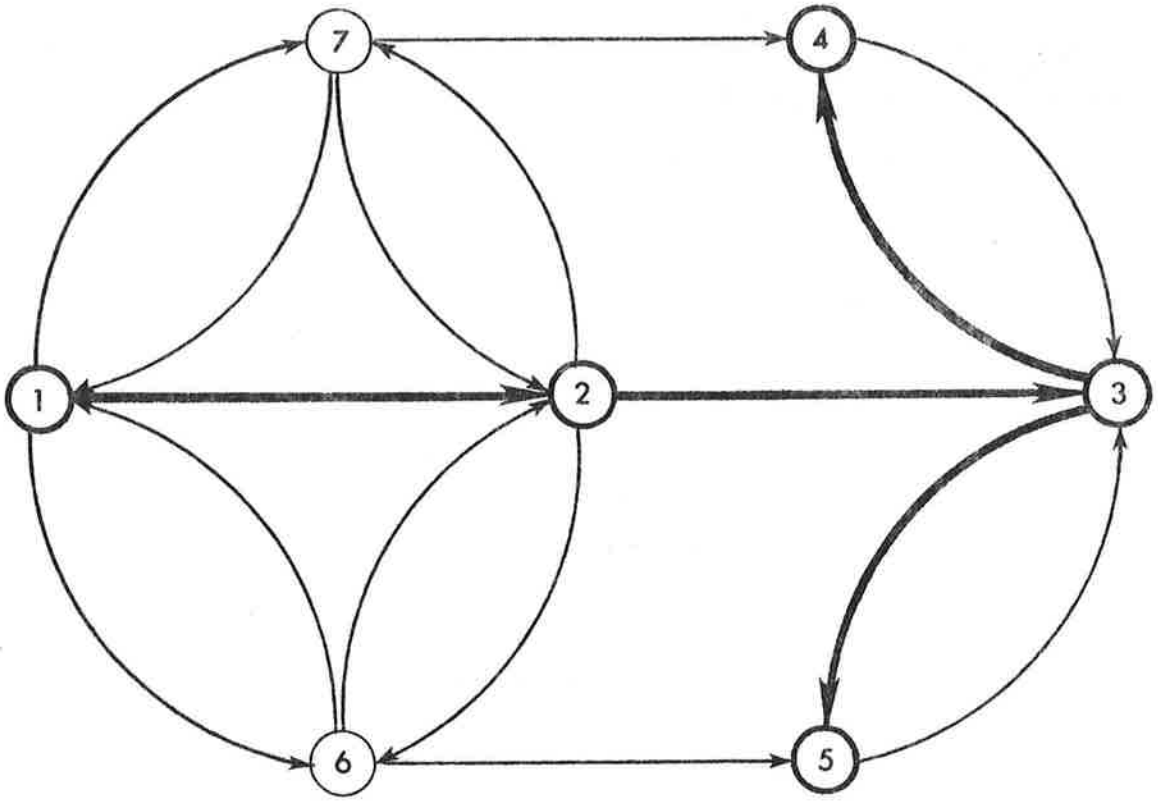
Arcs	c_0	c_1
12, 23, 34 35, 73, 53	1	1
65, 74, 26, 62, 72, 27, 16, 61, 71, 17.	1	4

The complete trip table (z^o), is defined as:

$[z^o]$ =

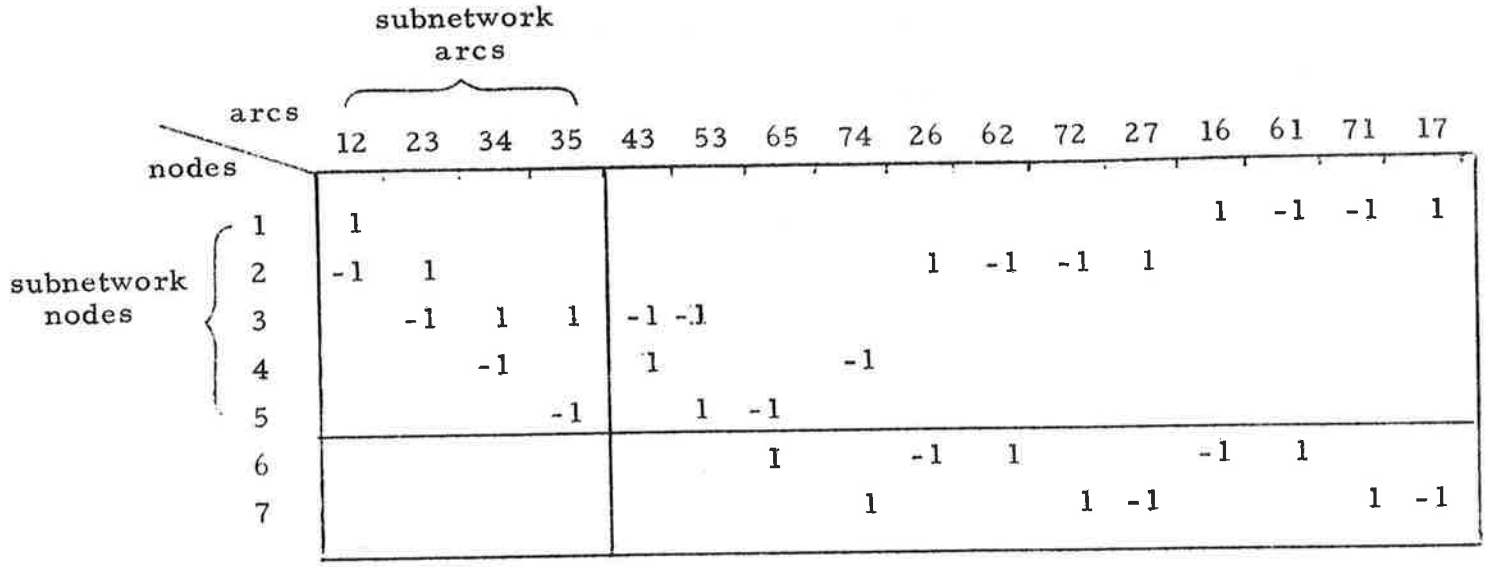
from \ to	4	5
1	6	6
2	9	9
6	9	5
7	5	9

The node-arc incidence matrix for the complete network is given below. The upper left portion is the n - a incidence matrix for the extracted subnetwork.



Note: extracted network in bold ink.

FIGURE 27. NETWORK FOR EXAMPLE 1.5



P1 for this problem is a quadratic program. Optimal flows and travel times are given below:

Arc	f_{ij}^*	Travel Time
12	8.9	9.9
23	33.6	34.6
34	19.4	20.4
35	19.4	20.4
43	2.6	3.6
53	2.6	3.6
65	12.2	49.8
74	12.2	49.8
26	1.1	5.2
62	4.4	18.8
72	4.4	18.8
27	1.1	5.2
16	3.5	15.1
61	2.0	8.9
71	2.0	8.9
17	3.5	15.1

The objective function value at optimality is approximately 1864.

To find equivalent flows on the extracted subnetwork, we solve P3, defining the μ vector by:

$$\mu_{12} = 9.9$$
$$\mu_{13} = 9.9 + 34.6 = 44.5$$
$$\mu_{14} = 9.9 + 34.6 + 20.4 = 64.9$$
$$\mu_{15} = 9.9 + 34.6 + 20.4 = 64.9$$
$$\mu_{23} = 34.6$$
$$\mu_{24} = 34.6 + 20.4 = 55.0$$
$$\mu_{25} = 34.6 + 20.4 = 55.0$$
$$\mu_{34} = 20.4$$
$$\mu_{35} = 20.4.$$

The structure of the extracted network allows elimination of μ_{21}, μ_{32}, μ_{31}, etc., since these node pairs are not connected by a directed path. This yields the following solution for flows and trip demands on the extracted network.

Arc	f_{ij}^*
12	8.9
23	33.6
34	19.4
35	19.4

$[\underline{z}^*]$ =

	4	5
1		8.9
2	14.2	10.5
3	5.2	

The objective function value at optimality is approximately -981. As mentioned above, the solution need not be unique in $\underline{z}^*$. Below is a set of trip tables that yield the same set of flows and P3-objective function, for $0 \leq \alpha \leq 10.5$:

$[\underline{z}^*]$ =

	3	4	5
1			8.9
2	α	14.2	$10.5-\alpha$
3			α

5. Sensitivity Results

To use Theorem 1.2 for solving the variable trip table traffic assignment problem one needs to know the equilibrium travel times μ_{pq}. We expect that these values will be unknown, and that one will solve P3 using estimates for μ_{pq}. It is important, in this case, to know how errors in these estimates affect the optimal values x^*, z^* for P3.

Basic results on the sensitivity of nonlinear program solutions to changes in problem parameters have been presented by Bigelow and Shapiro [1974], Robinson [1974], and Fiacco [1976]. In general terms, each of the three papers above is based on an application of the Implicit Function Theorem to the Kuhn-Tucker first order necessary conditions for optimality.

The Bigelow and Shapiro result is more general than the others in that strict complementary slackness is not required. Solutions to P3 need not satisfy strict complementary slackness, so the Bigelow and Shapiro results are more applicable for our problem.

If the conditions of the Kuhn-Tucker theorem are met, Bigelow and Shapiro's results imply the following conditions:

$$\begin{bmatrix} D & \vert & \underline{B}^T \\ & \vert & \underline{T}^T \end{bmatrix} \vert \; -I \; \vert \begin{pmatrix} \dot{x} \\ \dot{z} \\ \dot{\lambda} \\ \dot{u} \\ \dot{v} \end{pmatrix} = \begin{bmatrix} \geq 0 \\ \\ [\dot{\mu}_{pq}] \end{bmatrix}$$

$$\underline{B}^i \circ \dot{x} - \underline{T}^i \cdot \dot{z} \begin{cases} = 0 & \text{if } \lambda_i \neq 0 \\ \leq 0 & \text{if } \lambda_i = 0 \end{cases}$$

$$\dot{x}_i \begin{cases} = 0 & u_i > 0 \\ \geq 0 & u_i = x_i = 0 \end{cases}$$

$$\dot{z}_i \begin{cases} = 0 & v_i > 0 \\ \geq 0 & v_i = z_i = 0 \end{cases}$$

$$\dot{\lambda}_i \geq 0 \qquad \lambda_i = 0$$

$$\dot{u}_i \begin{cases} = 0 & x_i > 0 \\ \geq 0 & u_i = x_i = 0 \end{cases}$$

$$\dot{v}_i \begin{cases} = 0 & z_i > 0 \\ \geq 0 & v_i = z_i = 0 \end{cases}$$

$$\mathring{\lambda}_i \; \underline{B}^i \cdot \dot{x} - \underline{T}^i \cdot \dot{z} = 0 \qquad \lambda_i = 0$$

$$\mathring{u}_i \; \dot{x}_i = 0 \qquad u_i = x_i = 0$$

$$\dot{v}_i \; \mathring{z}_i = 0 \qquad v_i = z_i = 0,$$

where $\underline{B}^i$ and $\underline{T}^i$ represent the i^{th} rows of the block diagonal matrices having k (# destination centroids) blocks of A and T respectively. D is the Hessian matrix (with respect to both x and z) of the objective function of P3.

Bigelow and Shapiro show that these conditions can be solved by solving a certain quadratic program.

To illustrate these results for the subnetwork above, we consider the affects of changing u_{24} and u_{25} from 55.01 to 56.01. Thus $\dot{u}_{24} = \dot{u}_{25} = 1$, $\dot{u}_{pq} = 0$ for all other pairs. The changes in x and z, illustrated in Figure 28, are

$$\mathring{x}^4_{23} = 1 \qquad \dot{z}^{24} = 1 \qquad \dot{z}_{15} = -1$$

$$\dot{x}^5_{12} = -1 \qquad \mathring{z}^{34} = -1 \qquad \dot{z}_{25} = 1.$$

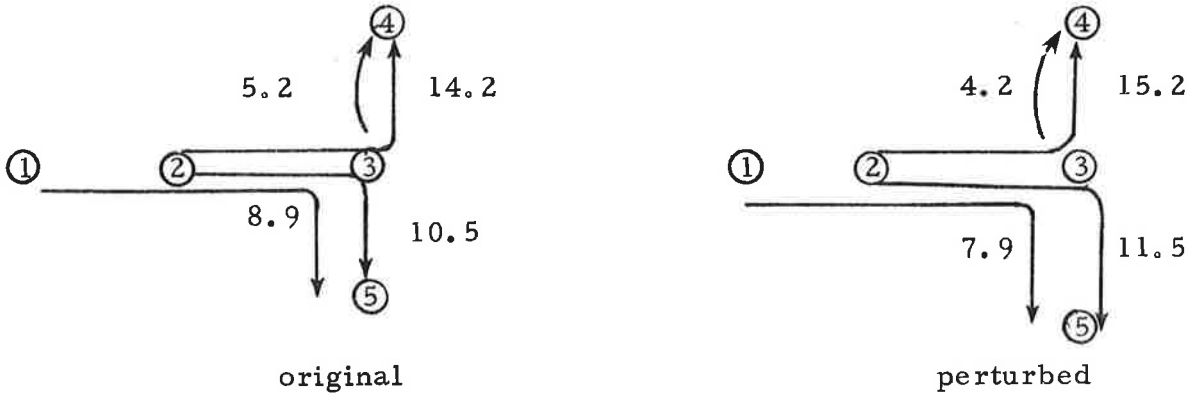


FIGURE 28. EFFECT ON OPTIMAL FLOWS OF $\dot{\mu}$

These changes, along with changes in the dual variables (u, v, λ) do indeed satisfy the above conditions. The solution is not unique, however, at least in terms of the $\dot{u}$, $\dot{v}$, and $\dot{\lambda}$ values.

Another problem in solving the combined distribution-assignment problem (P3) is that estimates of the μ_{pq} values may be inconsistent. For example, a planner may inadvertently estimate $\mu_{pq_1} < \mu_{pq_2}$ when the shortest pq_1 path passes through q_2. This could easily occur if the planner is asked to supply O-D travel times for a complex network. When the μ_{pq} values are inconsistent, the conditions of the Nguyen theorem are not met, and since the P3 optimal x and z values will be feasible, their implied travel times will not match the inconsistent μ vector. This is illustrated below.

Example 1.6 Consider the network below, with trip Table

$[z^o]$

	2	3
1	1	1
2		2

and link travel time function

$$c(u) = 1 + u$$

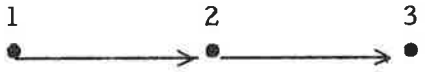


FIGURE 29. NETWORK FOR EXAMPLE 1.6

The optimal solution is given by:

$$x_{12}^2 = x_{12}^3 = 1,\ x_{23}^2 = 0,\ x_{23}^3 = 3,$$

and the resulting equilibrium times are:

$$\mu_{12} = 3$$

$$\mu_{23} = 4$$

$$\mu_{13} = 7 .$$

As expected, solution of P3 with these μ_{pq} estimates gives flows that are consistent with the p-q travel times. Now suppose one attempted to solve the variable trip table traffic assignment problem (P3) for this network using

$$\tilde{\mu}_{12} = 5$$

$$\tilde{\mu}_{23} = 3$$

$$\tilde{\mu}_{13} = 4 .$$

Solution of this problem gives the trip Table

$$[z] = \begin{array}{c|cc} & 2 & 3 \\ \hline 1 & 4 & \\ 2 & & 2 \end{array}$$

and flows: $x_{12}^2 = 4$, $x_{12}^3 = 0$, $x_{23}^3 = 2$. The resulting equilibrium travel times are

$$\bar{\mu}_{12} = 5 \quad = \tilde{\mu}_{12}$$

$$\bar{\mu}_{23} = 3 \quad = \tilde{\mu}_{23}$$

$$\bar{\mu}_{13} = 8 \quad \neq \tilde{\mu}_{13} .$$

(Since the $\bar{\mu}$ values are necessarily consistent with the optimal x and z values, they cannot match the input $\tilde{\mu}$ vector.)

6. EXPERIMENTS WITH A TRANSFER AND FLOW HEURISTIC

Russell R. Barton and Donald W. Hearn

Introduction

In the computation experiments of reference [1], it was suggested that a heuristic designed to transfer the correct percentage of the original trip table to the subnetwork of the Extraction Aggregation Model might be an economical alternative to flowing the entire network. This paper summarizes the results of experiments with such a heuristic applied to the network data base derived from the metropolitan D. C. network as described in [1].

The subnetwork, sketched in Figure 30, consists of major arteries of the Washington area. It has 483 links and 228 nodes, none of which are centroids, and is an extraction of the much larger (9, 386 links, 3, 027 nodes, and 700 centroids) VA/DC network as described in [1]. The trip table for the VA/DC network contained a total demand of 109, 000 trips between the 700 centroids. Of particular interest is the Shirley Highway. The experiments in [1] included the flowing of the entire VA/DC network to obtain benchmark flows for the links of the Shirley Highway, and heuristic attempts to replicate those flows by emulation of an actual study.

The heuristic of this study, which we have termed "M1", also attempts to replicate the benchmark flows. In the following sections, M1 is described and computation results reported.

M1 Heuristic

The heuristic assumes that there is a defined subnetwork of an existing given network and a given trip table for the full network.

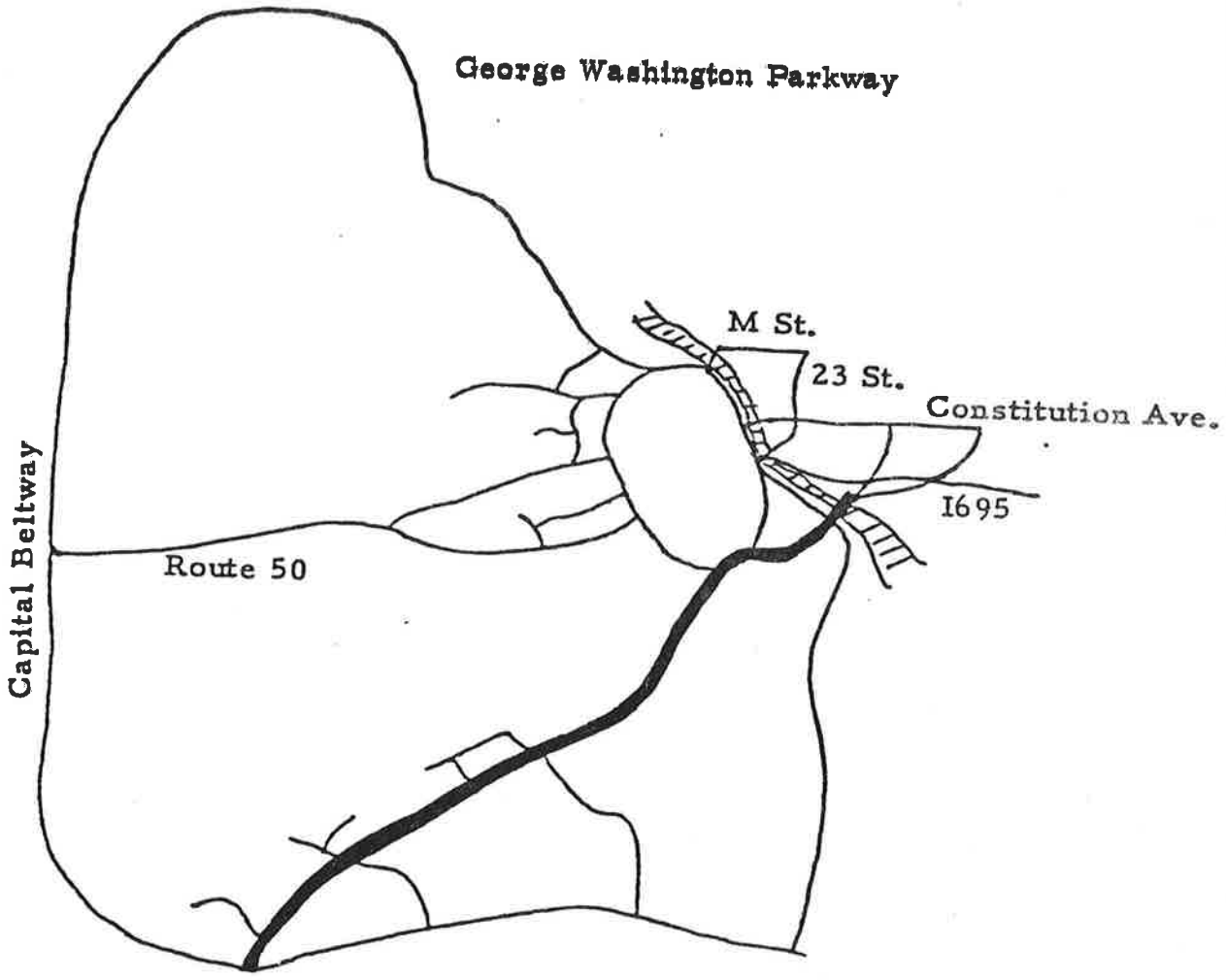


FIGURE 30. MODIFIED SHIRLEY STUDY NETWORK

TABLE 12. M1 ALGORITHM RESULTS

	F = 0.50		F = 0.65	
N_A	Shirley	Subnetwork	Shirley	Subnetwork
1	-534	196	816	736
2	-724	19	507	504
3	-867	-79	151	341

Average Per Link Bias

Table 12(a)

	F = 0.50		F = 0.65	
N_A	Shirley	Subnetwork	Shirley	Subnetwork
1	772	648	1245	973
2	780	545	1067	758
3	867	507	755	625

Average Per Link Error

Table 12(b)

N_A	No. Links	Time (Secs)	Percent
1	1883	62	28%
2	3249	75	34%
3	4593	82	37%
Full Network	9386	221	100%

CPU Time for Shortest Path Calculations

Table 12(c)

REFERENCE

1. Hearn, D. W., "Network Aggregation in Transportation Planning Models," Mathtech Final Report, DOT-TSC-RSPD-78-8,I, April, 1978.

APPENDIX - REPORT OF NEW TECHNOLOGY

This appendix certifies that no patentable inventions were developed in the course of this research. Principal new results of the research reported here are the methods of transfer and subset decomposition reported in papers 1 and 2, the theoretical results on the duality gap function in paper 3, and the application of those results to improved bounds in traffic assignment models reported in paper 4.

110 Copies

U.S. DEPARTMENT OF TRANSPORTATION
RESEARCH AND SPECIAL PROGRAMS ADMINISTRATION
TRANSPORTATION SYSTEMS CENTER
KENDALL SQUARE, CAMBRIDGE, MA. 02142

OFFICIAL BUSINESS
PENALTY FOR PRIVATE USE, $300

POSTAGE AND FEES PAID
U.S. DEPARTMENT OF TRANSPORTATION
513