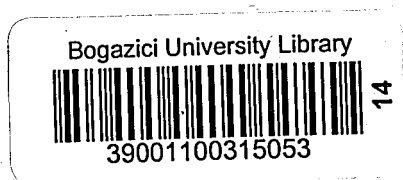


**SIMULTANEOUS OPTIMIZATION OF
NETWORK STRUCTURE AND CIRCUIT ROUTING
IN TRANSMISSION NETWORKS**

by

Ayşe Tülin Aktin

B.S. in I.E., Boğaziçi University, 1981



Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of

Master of Science
in
Industrial Engineering

Boğaziçi University

1984

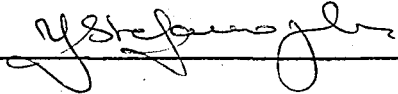
We hereby recommend that the thesis entitled "Simultaneous Optimization of Network Structure and Circuit Routing in Transmission Networks" submitted by Ayşe Tülin Aktin be accepted in partial fulfillment of the requirements for the degree of Master of Science in Industrial Engineering in the Institute for Graduate Studies in Science and Engineering, Boğaziçi University.

EXAMINING COMMITTEE

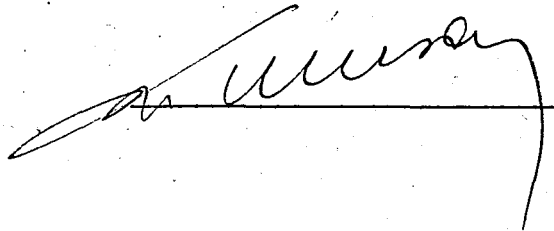
Öğr.Gör. Çetin EVRANUZ
(Thesis Advisor)



Doç.Dr. Yorgo ISTEфанOPULOS

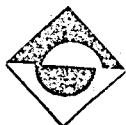


Doç. Dr. Gündüz ULUSOY



DATE:

181710



ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my thesis supervisor Mr. Çetin Evranuz for his continuous help and support during all the stages of this study.

I would also like to express my sincere gratitude to Doç.Dr. Gündüz Ulusoy for his valuable comments and suggestions.

Thanks are also due to Doç.Dr. Yorgo Istefanopulos for all his contributions.

I am also deeply grateful to the members of the Operational Research Division of Marmara Scientific and Industrial Research Institute for their understanding and help especially during the computer programming stage of this study.

I also sincerely wish to thank Mr. Erdal Yazgaç for the excellent drawings and for his boundless understanding and tolerance.

ABSTRACT

A telecommunications network can be separated into a switching and a transmission network. Using the idea of this separation, the planning and optimization of telecommunication networks is divided into two distinct but interlinked stages as the switching network optimization and the transmission network optimization. Furthermore, a modular approach is chosen to solve these sub-problems. In this study, the first two modules of the transmission network optimization problem, namely, the network structure and circuit routing optimization modules, will be handled. Unlike the algorithms contained in literature, a simultaneous optimization procedure is adopted in this study.

The developed solution procedure considers this problem both as a development (no existing transmission media) and as a capacity expansion (taking account of the existing network) problem.

One of the most important features of this problem is the existence of economies of scale in the link cost functions for installing transmission systems. These link cost functions can be decomposed into a fixed and a variable cost component. In the case of using alternate systems, these functions become piecewise concave, that is, concave in the range covered by any single technology.

Some of the major characteristics of the developed algorithm are as follows: it can take into account the existing network, consider mixed technology and alternate transmission systems, handle economies of scale and fixed charges present in the problem, together with the capacity limits and the parallel links in the network.

Ö Z E T

Bir telekomünikasyon şebekesi iki ana bileşenden oluşmaktadır: bir santral şebekesi ve bir iletişim şebekesi. Bu ayırımdan dolayı telekomünikasyon şebekelerinin planlanması ve eniyilenmesi problemi de iki aşamada gerçekleşmektedir. Önce santral şebekesinin eniyilenmesi problemi çözülmekte, daha sonra bu aşamanın çıktıları girdi olarak kullanılarak iletişim şebekesinin eniyilenmesi problemi ele alınmaktadır. Bu farklı, fakat birbirleriyle yakın ilişkisi bulunan problemlere modüler bir yaklaşım uygulanarak, bunların kendi aralarında da bazı altproblemlere ayrışmaları sağlanmaktadır. Bu çalışmanın konusu, iletişim şebekesinin eniyilenmesi probleminde ilk iki altproblem olarak geçen serim yapısının belirlenmesi ve devre yönlendirilmesinin eniyilenmesi problemleridir. Bu iki altproblem genellikle ayrı ayrı eniyilenmektedir. Çalışmada probleme yeni bir boyut getirilmiş ve bu problemlerin eşzamanlı eniyilenmesi düşünülmüştür.

Geliştirilen çözüm yordamında bu problem hem mevcut iletişim sistemlerinin gözönüne alınmadığı bir geliştirme problemi olarak, hem de mevcut şebekenin hesaba katıldığı bir kapasite artırımı problemi olarak ele alınmaktadır.

Problemin en önemli özelliklerinden biri, iletişim sistemlerinde ölçek ekonomisinin geçerli olması ve maliyet işlevinin bir sabit bileşenle, devre sayısına bağlı bir değişken bileşene ayrıştırılabilesidir. Ancak, alternatif

sistemlerin kullanılması söz konusu olduğunda, maliyet işlevi parçalı sürekli içbükey bir işlev olmaktadır.

Geliştirilen algoritmanın en önemli özellikleri, mevcut şebekeyi gözönüne alabilmesi, birden fazla teknolojinin, alternatif iletişim sistemlerinin, ölçek ekonomisinin, sabit maliyetlerin, şebekedeki paralel ve kapasite kısıtlı ayrıtların içerilmiş olmasıdır.

TABLE OF CONTENTS

	<u>Page</u>
ACKNOWLEDGEMENTS	iii
ABSTRACT	iv
ÖZET	vi
LIST OF FIGURES	xi
LIST OF TABLES	xiii
I. INTRODUCTION	1
II. SOME RELATED CONCEPTS OF TELECOMMUNICATION NETWORKS	6
2.1 Basic Definitions	6
2.2 Nature of Telephone Traffic	10
2.3 Technological Aspects of Transmission Systems	12
III. PROBLEM DEFINITION	15
3.1 The General Telecommunications Network Optimization Problem	15
3.2 The Transmission Network Optimization	19
3.2.1 The Network Structure Optimization	23
3.2.2 The Circuit Routing Optimization	25
IV. LITERATURE SURVEY	29
4.1 A General Survey	29
4.2 A Survey on the Network Structure and Circuit Routing Optimization Problems	44

	<u>Page</u>
V. A SURVEY ON SHORTEST PATH ALGORITHMS AND A SPECIAL IMPLEMENTATION FOR CAPACITATED NETWORKS	50
5.1 Literature Survey on Shortest Path Algorithms	50
5.2 Network Representation	52
5.2.1 Terminology	53
5.2.2 Data Storage Problem in Large-Scale Networks	53
5.2.2.1 The Matrix Representation	54
5.2.2.2 The Ladder Representation	55
5.2.2.3 The Forward Star Representation	56
5.3 Basic Labeling Methods	57
5.3.1 The Label-Correcting Method	59
5.3.2 The Label-Setting Method	59
5.4 A Special Implementation of Dijkstra's Shortest Path Algorithm for Capacitated Network Problems	60
5.4.1 The Characteristics of the Developed Algorithm	61
5.4.2 Flowchart of the Algorithm	62
5.4.3 A Performance Comparison of the Developed Algorithm and Some Other Algorithms	65
VI. PROBLEM FORMULATION AND THE SOLUTION PROCEDURE	68
6.1 Characteristics of the Problem	68
6.2 Assumptions of the Model	74
6.3 The Solution Procedure	75
6.3.1 The Development Problem	82
6.3.2 The Capacity Expansion Problem	89
6.3.3 The Minimum Cost Flow Algorithm Used in the Procedure	91
VII. NUMERICAL RESULTS	94
7.1 The Test Network	94
7.2 Results of the Test Network	98
7.2.1 Results of the Development Problem	99
7.2.2 Results of the Capacity Expansion Problem	108
7.3 Analysis of the Results	111

VIII. CONCLUSION AND EXTENSIONS FOR FUTURE WORK	113
APPENDIX A. DESCRIPTION OF THE INPUT DATA	118
APPENDIX B. EXPLANATION OF THE COMPUTER PROGRAM	119
APPENDIX C. LISTING OF THE COMPUTER PROGRAM	125
BIBLIOGRAPHY	175

LIST OF FIGURES

	<u>Page</u>	
FIGURE 1.1	The General Description of the Procedure	2
FIGURE 2.1	A Telecommunications Network	6
FIGURE 2.2	An Example of Switching Nodes and Links	8
FIGURE 2.3	The Hierarchy of Concepts	9
FIGURE 2.4	Routing Plan for Origin Destination Pair (A,D)	11
FIGURE 3.1	The Overall Procedure	17
FIGURE 3.2	Routing of Trunk Group AB on the Transmission Network	18
FIGURE 3.3	Flowchart of the Transmission Network Optimization Problem	22
FIGURE 3.4	Flowchart of the NSO Module	26
FIGURE 3.5	Handling Mixed Transmission Media	27
FIGURE 3.6	Flowchart of the CRO Module	28
FIGURE 4.1	A Typical Capacity Expansion Cost Function	37
FIGURE 4.2	Cost Function for Telpak Renting	44
FIGURE 4.3	A Failing Case for the Connectivity Measure on Node Degrees	45
FIGURE 4.4	An Example of Three Edge - Disjoint Paths for the Node Pair (A,B)	46
FIGURE 5.1	Example Network	54
FIGURE 5.2	Flowchart of the Shortest Path Algorithm Used in the Solution Procedure	62

	<u>Page</u>
FIGURE 6.1 The Cost Function of the Three Alternate Transmission Systems	71
FIGURE 6.2 A General Glance at the Problem	76
FIGURE 6.3 The Approximate Link Cost Function	77
FIGURE 6.4 A Failing Case of Rerouting Between the Endpoints of a Link	79
FIGURE 6.5 A Better Way of Rerouting (a)	80
FIGURE 6.6 A Better Way of Rerouting (b)	81
FIGURE 6.7 Flowchart of the Solution Procedure	83
FIGURE 6.8 Calculating Remaining Capacities in the Development Problem	88
FIGURE 6.9 Calculating Remaining Capacities in the Capacity Expansion Problem	90
FIGURE 7.1 Switching Network of the Test Network	95
FIGURE 7.2 Transmission Network of the Test Network	96
FIGURE 8.1 An Example of Multirouting	114
FIGURE 8.2 Updated Flow Values	115

LIST OF TABLES

	<u>Page</u>	
TABLE 5.1	Matrix Representation of the Example Network	55
TABLE 5.2	Ladder Representation of the Example Network	55
TABLE 5.3	Forward Star Representation of the Example Network	57
TABLE 5.4	Results of the Timing Experiments for the Three Algorithms	67
TABLE 6.1	The Cost Components and Capacity of the Alternate Systems Used in the Model	71
TABLE 7.1	Originating and Terminating Nodes of the Links and Their Distances	95
TABLE 7.2	Originating and Terminating Switching Nodes of Each Trunk Group and Their Number of Circuits (Set 1)	97
TABLE 7.3	Originating and Terminating Switching Nodes of Each Trunk Group and Their Number of Circuits (Set 2)	98
TABLE 7.4	Demand Data Set 1; Demands in Descending Order; Criterion V_1	100
TABLE 7.5	Demand Data Set 1; Demands in Ascending Order; Criterion V_1	101
TABLE 7.6	Demand Data Set 2; Demands in Descending Order; Criterion V_1	102
TABLE 7.7	Demand Data Set 2; Demands in Descending Order; Criterion V_2	103

TABLE 7.8	Demand Data Set 2; Demands in Descending Order; Criterion V_3	104
TABLE 7.9	Demand Data Set 2; Demands in Ascending Order; Criterion V_1	105
TABLE 7.10	Demand Data Set 2; Demands Unsorted; Criterion V_1	106
TABLE 7.11	Demand Data Set 2; Demands Unsorted; Criterion V_3	107
TABLE 7.12	Demand Data Set 2; Demands in Descending Order; Criterion V_2	108
TABLE 7.13	Demand Data Set 2; Demands in Ascending Order; Criterion V_2	109
TABLE 7.14	Demand Data Set 2; Demands Unsorted; Criterion V_2	110

I. INTRODUCTION

A telecommunications network is the means of interconnecting telephone customers on demand. This network is in fact a collection of a number of telephone exchanges (or switching nodes) interconnected by links (transmission systems). The basic elements of a transmission system are the circuits which are fixed means of conveying one conversation or other communication in both directions between two specified nodes [1].

The optimization of telecommunication networks is a rather complex problem. This complexity is due to the trade-off between the investments in equipment to meet the demand from the customers and the quality of service that will be given. It is essential to enable the customers to make successful calls to other customers in an economic manner, but of course together with a satisfactory quality of service. This prescribed standard of service is evaluated in terms of the proportion of successful calls during the busy hour of the day (known as Grade of Service or blocking). Some of the factors that can degrade the quality of service perceived by the customer are caused by equipment failures, and congestion due to unforeseen surges of traffic due to customer behaviour. On the other hand, the financial return will be insufficient if the investments in equipment are too high. Likewise, if they are too

small the quality of service perceived by the customers will be poor because of the insufficient plant to carry the telephone traffic. Hence, to resolve the trade-off between cost and service, the investments in equipment must be minimized subject to a set of service level constraints [2].

A telecommunications network comprises two major component networks: a switching network and a transmission network. A switching network consists of switching nodes interconnected by groups of circuits (trunk groups) over which the telephone traffic flows. A transmission network consists of the transmission systems which carry the trunk groups between switching nodes. One of the traditional transmission facilities is the cable consisting of a large number of wires. Recently, the use of radios, satellites and fiber optics have been initiated.

The optimization of telecommunication networks is divided into two distinct but interlinked stages as the Switching Network Optimization Problem (SNOP) and the Transmission Network Optimization Problem (TNOP) [2,3,4,5].

The general optimization procedure can be illustrated as follows:

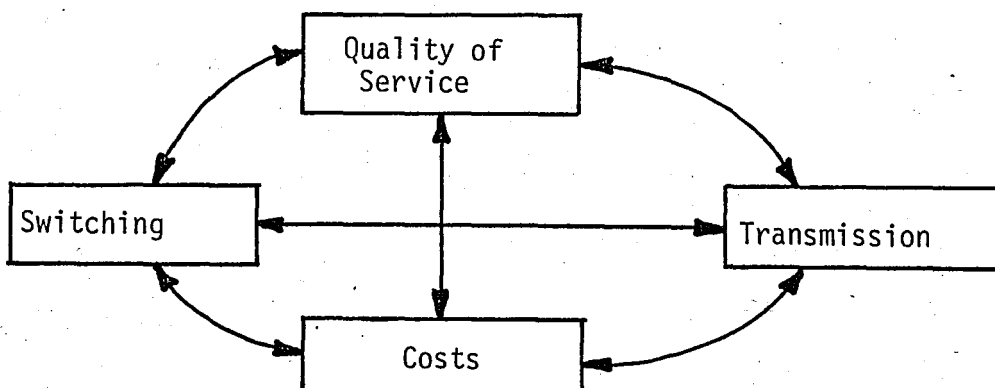


Figure 1.1 - The general description of the procedure.

The left part of the figure represents the Switching Network Optimization Problem in which the circuit demands are determined from the optimization of traffic routings in the switching network. Evranuz, et.al., [6] have conducted a study on SNOP. In another study, Mısırlı [7] presents an algorithm for SNOP together with some different approaches to this problem.

The right part of the figure stands for the Transmission Network Optimization Problem. In this problem, these circuit demands will be routed optimally in the transmission network. Baybars and Kortanek [8], Evranuz [9], Evranuz and Miraboğlu ([10, 11, 12]), Evranuz and Aktin [13] have studies on the optimization of the transmission facilities for telecommunication networks.

Finally, the overall optimization process, as enabled by the algorithms developed, takes place in between the opposing poles of Quality of Service and Costs.

The transmission network optimization, as mentioned by Nivert and Noort [14], is carried out in a number of logical steps. First of all, the network structure has to be designed in such a way that a certain degree of structural reliability is achieved, and that the routing of circuits will not require major changes in the already existing parts of the network. Secondly, the circuit demand is routed in an optimal way, taking into account the diversification requirements.

In order to determine whether the service requirements are met, the network is then analysed in a rather approximate manner. It is on the basis of this analysis that the requirements on a stand-by protection network can be established, which is to be optimally routed on

the remaining capacities in the network. The analysis may also serve in defining the appropriate degree of overdimensioning of particular trunk groups. Cavellero and Tonietti [15], Kaptanoğlu and Evranuz [16] and Kaptanoğlu [17] have studies on the determination of stand-by requirements in telecommunication network.

The general optimization procedure uses a modular approach in solving its related problems. The switching network optimization problem is divided into eight main modules, while the transmission network optimization problem into five main modules [4].

The switching network optimization problem and the last three modules of the transmission network optimization problem (which are the Grade of Service Evaluation, Stand-by Requirements Determination, and Stand-by Optimization modules) are totally out of the scope of this study. Only the first two modules of TNOP, namely, the network structure and circuit routing optimization problems are analyzed in the thesis. But rather than optimizing them separately as it is done by the members of the COST 201 Project [2,4], a simultaneous optimization procedure is proposed. This procedure considers the problem both as a development (no existing media), and as a capacity expansion (taking account of the existing network) problem.

After this introductory presentation of the problem, Chapter II gives some of the related concepts of telecommunication networks.

With the aim of defining the problem clearly, Chapter III provides a more exact description of the telecommunications network optimization

problem, and states the two problems, namely, the network structure and the circuit routing that will be optimized simultaneously in this thesis. It also includes the input requirements and the resulting outputs.

A brief review of the related literature is given in Chapter IV.

Chapter V presents a survey on shortest path algorithms and introduces the special implementation which was developed especially for the capacitated networks involved in the transmission network optimization problem.

Chapter VI starts with the characteristics of the problem and the main assumptions of the developed model, and describes the formulation of the problem. At the end of the chapter, flowcharts of the solution procedure applied are included.

In Chapter VII, the test network is introduced, and the numerical results of the developed algorithm are discussed.

The listing and explanation of the computer program appear in the appendices.

II. SOME RELATED CONCEPTS OF TELECOMMUNICATION NETWORKS

This chapter presents some of the basic concepts that are commonly used in telecommunication networks, and also includes some technological aspects of transmission networks. The material presented in this chapter will provide a better understanding of the problem definition.

2.1 BASIC DEFINITIONS

A telecommunications network is a collection of junctions (or points) some or all of which are joined by direct communication links. Such a network can be illustrated by a graph in which "vertices" and "edges" correspond to the "points" and "links" of the network, respectively. An example of a real-life telecommunications network with 8 points and 15 direct links are shown in Figure 2.1 [8].

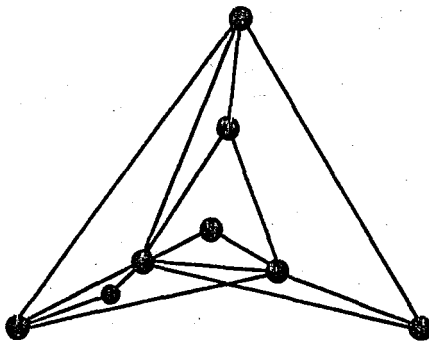


Figure 2.1 - A telecommunications network

A link is a collection of facilities known as transmission equipment which when taken together comprise various transmission systems.

Telephone traffic, which will be referred to simply as traffic, is defined as the aggregate of telephone calls over a group of circuits (trunks) with regard to the duration and the number of calls [18]. Traffic intensity is the traffic volume (total occupancy) occurring during a specified period of time divided by the duration of the period, both quantities being expressed in the same time unit [1]. The resultant quantity is fundamentally dimensionless and is expressed in erlangs, after the Danish mathematician A.K. Erlang, who is the founder of the telephone traffic theory, or in terms of hundred call seconds per hour (CCS). Since there are 3600 seconds in an hour, 36 CCS equals one erlang.

As mentioned in Chapter I, a telecommunications network is separated into a switching and a transmission network.

A switching network is a collection of switching nodes and their interconnecting switching links without regard to the transmission media on which the switching links are carried. A switching node is a switching machine at a specified location. A switching link is the total number of trunks connecting any two specified switching nodes irrespective of their grouping into trunk groups and direction of operation. A trunk group is a set of circuits treated as an entity for dimensioning purposes and provided to carry a specified amount of traffic between the same two switching nodes. Figure 2.2 illustrates the above definitions [1].

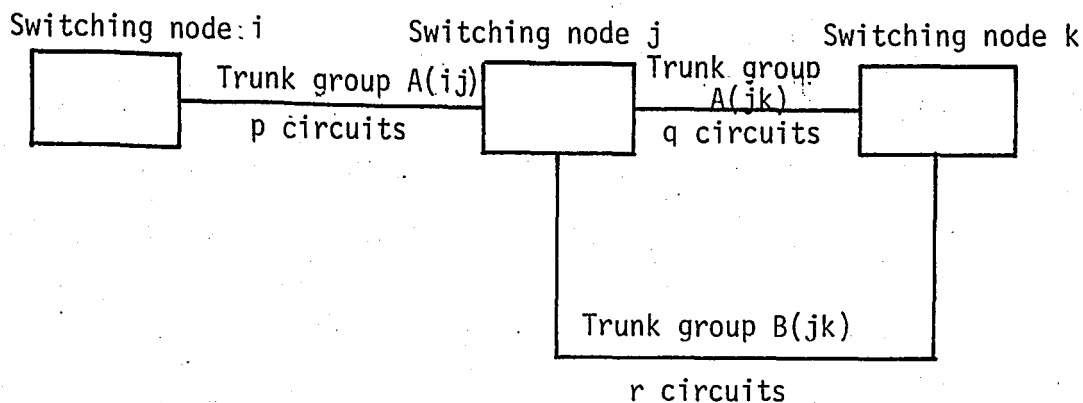


Figure 2.2 - An example of switching nodes and links.

In the figure given above,

Switching link $ij = A(ij) = p$ circuits

Switching link $jk = A(jk) + B(jk) = q + r$ circuits

A transmission network, on the other hand, is a collection of transmission nodes and their interconnecting transmission sections. A transmission node is a location in the transmission network where a transmission section terminates and which provides multiplexing, demultiplexing, or analogue/digital conversion for the interconnection of transmission sections. A transmission link is the total transmission capability between any two specified transmission nodes comprising one transmission section or a number of interconnected transmission sections. Finally, a transmission section is a transmission medium at any specified level of multiplexing

between two transmission nodes such that no intermediate nodes are involved and all circuits are carried on common physical equipment at some point between the nodes.

The hierarchy of concepts in relation to the switching and transmission networks is illustrated below [1].

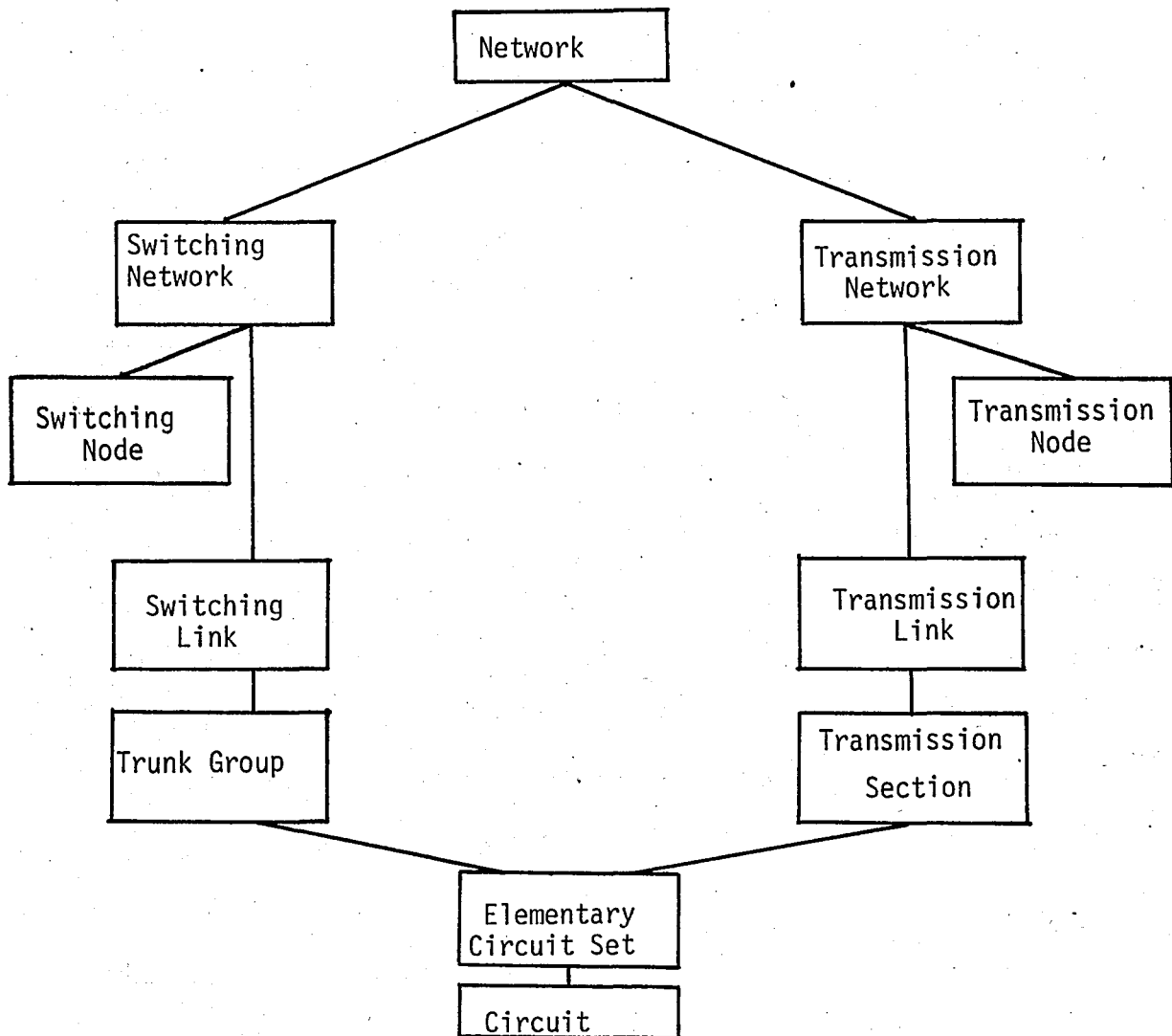


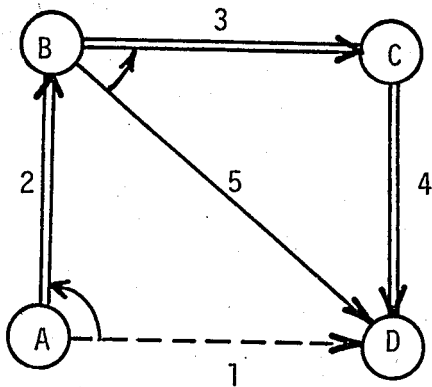
Figure 2.3 - The hierarchy of concepts.

2.2 NATURE OF TELEPHONE TRAFFIC

In real-life telephone networks, not all pairs of points are connected directly by transmission facility links. That is, in graph theoretic terms, telephone networks are typically non-complete graphs [8]. However, since the graph itself is connected, it is possible to dial any point from any other specific point. Traffic, in the form of voice telephone calls, originate at a junction A, such as a city, to be transmitted to another junction B, termed the destination. If there is a direct link in the network connecting points A and B then the call is transmitted through that link as long as not all the circuits of that link are busy at a given time. However, if all the A-to-B lines are busy, or if there is no direct link joining A and B, then the call can be transmitted from A to B through a sequence of links (with the assumption that there are switching facilities at point A to switch the A-to-B traffic to some other link). Depending on how dense (in terms of the number of links) the network is, there may be just a few or perhaps many such sequences of links which could carry the traffic of a specified link. Such sequences are referred to as alternate routes [8].

Furthermore, if the customer demand for some pair (A,B) exceeds the capacity of the direct link between A and B, then the excess demand can be switched to an alternate route. Hence, in peak traffic conditions when the traffic offered to a first choice direct trunk group encounters a blocking condition where all circuits are engaged, it can overflow to a second choice trunk group and so on, until the last and final traffic routing is reached, which is generally the basic route.

Figure 2.4 illustrates an alternate routing plan [2].



First choice route = {1}
 Second choice route = {2,5}
 Final choice route = {2,3,4}
 Direct route = {1}
 Final links = {2,3,4}
 High usage links = {1,5}

Figure 2.4 - Routing plan for origin destination pair (A,D).

In the example above, traffic routing from A to D is first offered to the first choice direct route A-D. If all circuits in this trunk group are already carrying calls, then the traffic overflows to the second choice route via B, A-B-D. If this traffic route is engaged, the traffic flows via the final routing A-B-C-D. There are two possibilities to establish a hierarchy of the links in the network, where a link is termed high-usage if its traffic can be switched to an alternate route (s); and is termed a final choice link if traffic overflows onto it. Thus, the excess traffic of final links are lost. Furthermore, a high-usage link can only carry its own traffic, whereas a final link can carry the traffic for other pairs of points in the network [19].

Routing implies that more than one set of circuits can be installed on a link to meet the requirements of several relations. However, because of the high fixed cost of installing a system, and of the economies of scale involved in installing a larger system, it will often be less

expensive to route them than otherwise.

It is important to differentiate between the offered and the carried traffic in order to clarify the nature of telephone traffic in a network. The carried traffic, which is a measurable quantity, is the amount of traffic actually handled by the system. The offered traffic, on the other hand, is usually greater than the carried traffic by the amount of lost or blocked traffic. This blocked traffic will either be lost and cleared away from the system, or overflow to an alternate route, if there exists any. Kaptanoğlu [17] gives in detail the theories and methodologies utilized in the calculation of blocking probabilities which are of great importance for the optimization procedure.

2.3 TECHNOLOGICAL ASPECTS OF TRANSMISSION SYSTEMS

Transmission systems provide the circuits for transmitting speech and other signals between the nodes of a telecommunications network. These circuits convey the signals in both directions. If a circuit uses a separate transmission path for each direction, then each of these unidirectional paths is called a channel. In general, a complete channel consists of sending equipment at a terminal station, a transmission link which may contain repeaters at intermediate stations, and receiving equipment at another terminal station [20].

Both transmission channels and the signals they convey may be classified in two classes: analogue and digital. An analogue signal is a continuous function of time; at any instant it may have any value between limits set by the maximum power that can be transmitted. Speech signals

are an example of analogue signals. On the other hand, a digital signal can only have discrete values. The most common digital signal is a binary signal, having only two values, "1" and "0". A telegraph signal is an example of a digital signal. A television waveform is a mixture of analogue and digital signals, since it transmits both the picture contents and synchronizing pulses.

Recently, digital techniques have been introduced into both switching and transmission. By this means, speech signals are sampled and converted into a series of coded pulses, each occupying a discrete and recurring time interval, and the time periods between these slots can be occupied by other conversations. This enables a large number of trunk groups to be time-division multiplexed onto two pairs of wires or coaxial tubes [3].

In order to transmit an analogue signal without error, the channel must be a linear system. Any deviation from linearity will cause non-linear distortion of an analogue signal. The most common examples of analogue channels are the cable systems and the radio-relay systems equipped with linear amplifiers. A digital channel does not require to be linear, since its output provides a number of discrete conditions corresponding to the input signal. A telegraph circuit, whose output signal is provided by the operation of a relay is an example of a digital channel.

It is not necessary to transmit analogue signals over analogue channels, and digital signals over digital channels. Data communication and voice-frequency telegraphy over telephone lines are examples of transmitting digital signals over analogue channels. Analogue signals may be coded for transmission over digital channels by means of analogue-to-digital converters. An example is the transmission of speech by means of

pulse-code modulation over lines equipped with regenerators.

If a link can provide adequate transmission over a band of frequencies which is wider than that of the signals to be sent, it can be used to provide a number of channels. At the sending terminal, the signals of different channels are combined to form a composite signal of wider bandwidth. At the receiving terminal, the signals are separated and retransmitted over separate channels. This process is known as multiplexing. It is used in order to maximize the number of traffic groups that can be carried on the transmission media. The separate channels that enter and leave the terminal stations are called baseband channels. The transmission link which carries the multiplex signal is called a broadband channel or a bearer channel [10].

The transmission network comprises transmission nodes interconnected by transmission links. Some types of transmission media are:

- analogue coaxial cable
- digital coaxial cable
- digital PCM dedicated cable
- analogue microwave radio
- digital microwave radio
- optical fibre cable
- submarine cable.

These different transmission media are made up of modules of circuits multiplexed to form the necessary frequency range (for analogue systems), or bit range (for digital systems). The transmission nodes provide multiplexing and demultiplexing facilities and also form flexibility points for the interconnection of transmission links or modules of circuits between transmission systems [2].

III. PROBLEM DEFINITION

The first section of this chapter includes a detailed description of the overall optimization problem. This will be helpful in understanding the relationship between the two distinct but interlinked stages of the telecommunications network optimization, that is, the switching network and the transmission network optimization problems, clearly. The second section of the chapter briefly describes the transmission network optimization problem, and gives more exact definitions of the network structure and circuit routing optimization problems which will be handled in the thesis simultaneously.

3.1 THE GENERAL TELECOMMUNICATIONS NETWORK OPTIMIZATION PROBLEM

A major characteristic of the overall optimization procedure is that the switching and transmission networks are treated separately in such a way that a consistent result will be obtained. The procedure has to produce minimum cost routings of traffic flow in the switching network, and the minimum cost routings of circuits in the transmission network. These minimum cost flows are constrained by the quality of service

parameters. A problem exists as how to achieve the quality of service required, and in order to handle this problem various possible measures are considered. One is the overdimensioning of trunk groups, in which the trunk group capacities are increased. A second measure is the diversification (or multirouting) of trunk groups, and a third measure is the application of redundancy in the transmission network [4,14].

The basic network elements to be treated by the procedure are switches, trunk groups, transmission stations and transmission sections. Of these elements, the most relevant attributes (location, hierarchy, technology, cost factors, capacity constraints, length, service requirements, failure data, etc.) are specified. In addition, items like the traffic demands, the routing strategy are input data. The output of the procedure essentially consists of a dimensioned switching network (trunk group sizes) and a designed transmission network (capacities of transmission sections, trunk group routes and stand-by capacities). Moreover, the relation between quality and costs will be qualified in terms of a number of parameters.

A flowchart of the overall procedure is given in Figure 3.1 [2].

As it can be seen from the figure below, the major inputs to the switching network optimization problem are the matrix of traffic demand between all switching nodes, hierarchy, minimum blockings, maximal routing scheme, and the mean circuit cost coefficients. The major output of this procedure is a circuit demand matrix (being a translation of the traffic demand matrix) which is passed forward to the transmission network optimization problem.

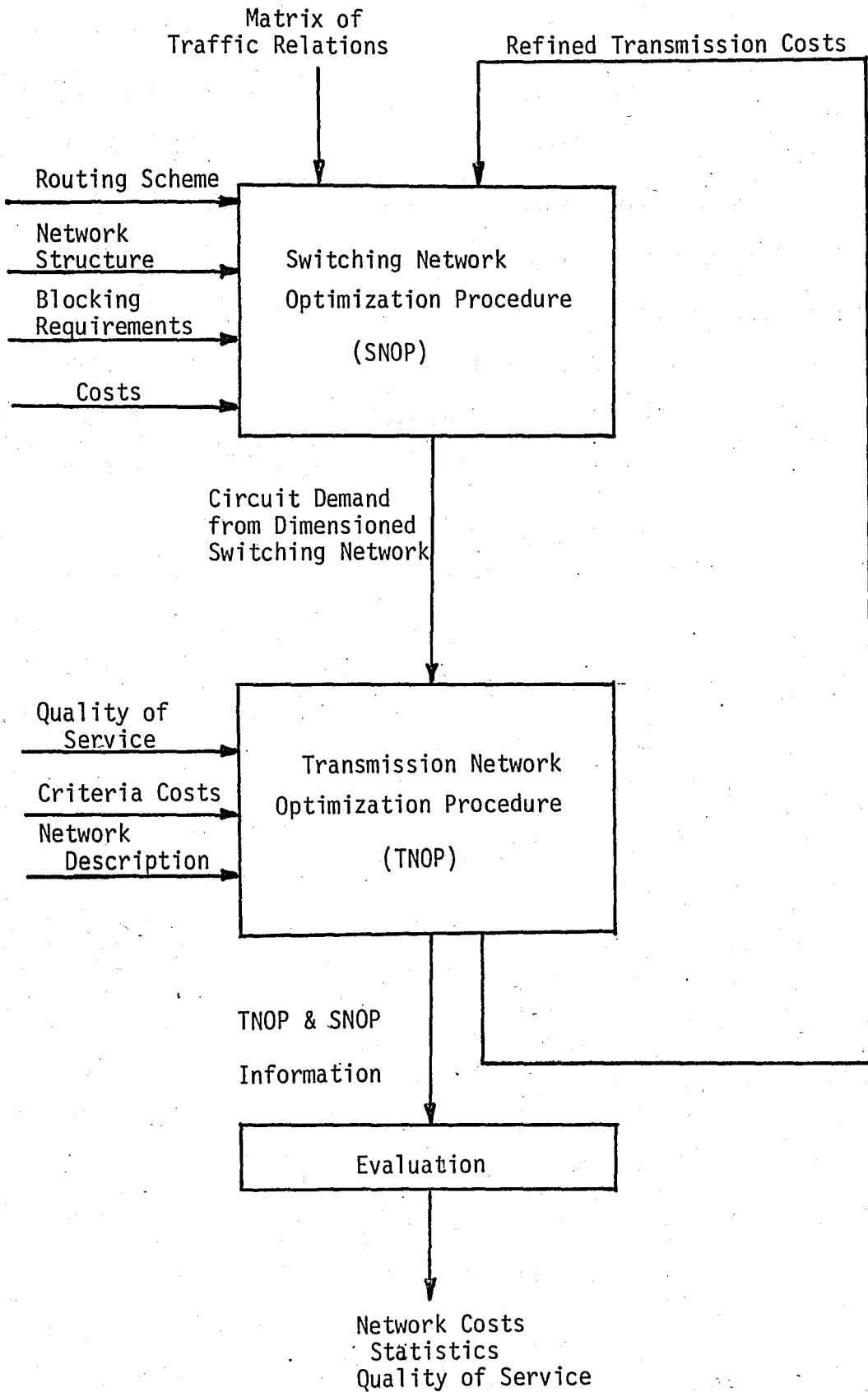


Figure 3.1 - The overall procedure.

In addition to the circuit demand comprising a matrix of trunk groups received from SNOP, the other main inputs to the transmission network optimization problem are the maximal graph of transmission nodes and media, cost figures and service requirements in failure conditions. The output consists of a detailed description of the resulting analogue-digital network, including the routing of all circuits, the proposed stand-by capacity per link together with the best routing of it, and a summary of the quality of the network.

The relationship between the switching and transmission networks can be made evident by the following example [1].

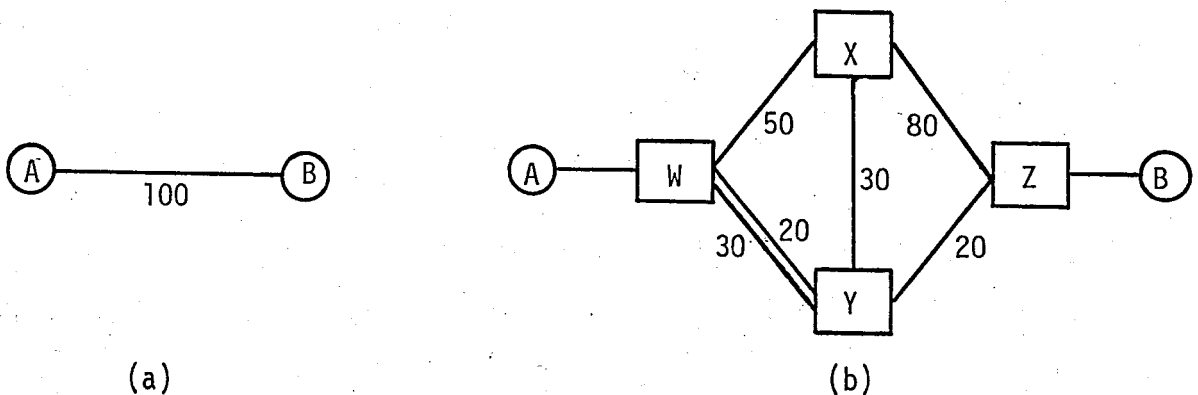


Figure 3.2 - Routing of trunk group AB on the transmission network.

Figure 3.2(a) shows a trunk group of capacity 100 between the switching nodes A and B. In Figure 3.2(b), these circuits are routed in the transmission network where W,X,Y,Z are the transmission nodes. These 100 circuits are allocated to three transmission paths in the following manner:

A → WYZ → B	20 circuits
A → WYXZ → B	30 circuits
A → WXZ → B	50 circuits

As it can be seen from Figure 3.2(b), there may be more than one transmission sections between two transmission nodes. This example also makes it clear that all the circuits of a trunk group may not be affected by the failure of a transmission media.

Taking advantage of certain network characteristics, the two main subproblems, that is, SNOP and TNOP, are further broken down, thus allowing for a modular and flexible procedure to be constructed [14]. In the overall optimization procedure, SNOP is handled in eight main modules, while TNOP in five main modules.

3.2 THE TRANSMISSION NETWORK OPTIMIZATION PROBLEM

The transmission network planning procedure involves examination of the matrix of trunk groups comprising the switching (for functional) network and grouping them to form the physical network of transmission nodes and links. It requires optimization to [2]:

- a. minimize the cost by minimizing the length of trunk groups, maximizing trunk groups sharing the same transmission link to achieve economies of scale; and selection of the most economic transmission media,
- b. meet security criteria (against failure of transmission media) by, for example,

- i) route diversity (or multirouting) where circuits making up individual trunk groups are split between two or more physically separate transmission paths,
- ii) a stand-by network where back-up systems are provided which can be used to duplicate working systems if they fail,
- iii) overdimensioning which is more relevant to the traffic relations, because it protects these relations rather than a particular trunk group or transmission section [3].

The transmission network is described as a multigraph. Each edge of the graph represents a transmission medium or the part of it using the same analogue or digital technique; so each transmission medium is represented by one or a pair of edges. In the last case, when a medium is in failure condition, the corresponding pair of edges fails at the same time.

The failures considered are single failures of transmission media. Multiple failures and node failures are not contained in the failure model.

Capacity limits and modularities of transmission media are taken into account in the procedure.

As mentioned at the end of Section 3.1, the transmission network optimization problem is divided into five main modules [4] which are:

NSO - Network Structure Optimization

CRO - Circuit Routing Optimization

GOS - Grade of Service Evaluation

SBR - Stand-by Requirements Determination

SBO - Stand-by Optimization.

Nivert and Noort [14] and Lindberg, et.al [21] who are members of the COST 201 Project describe the inter-relation between these modules by the following flowchart.

The first module which is the Network Structure Optimization (NSO), determines the basic structure of the transmission network from a maximal set of existing or proposed media. To each link, a capacity limit and a cost function, comprising a fixed and a variable component, are associated. As a result of the optimization procedure, the unnecessary links, subject to certain connectivity criteria being met, are deleted. In order to guarantee the connectivity of the network, a minimum node degree will be maintained for the transmission stations. Different connectivity criteria are discussed in Section 4.1.

This new network or the available network is passed to the Circuit Routing Optimization (CRO) module which performs the optimal routing of the trunk group demands upon it. If required, this module will also route the trunk groups on diverse paths (multirouting), and through the use of penalty functions will ensure that the maximum capacity of a media is not exceeded.

In parallel, the Grade of Service Evaluation (GOS) module prepares the traffic data for the Stand-by Requirements Determination (SBR) module by transforming the trunk group data into equivalent circuits. The SBR module is then able to examine the effect of a failure on each of the transmission media and give the number of stand-by circuits which are

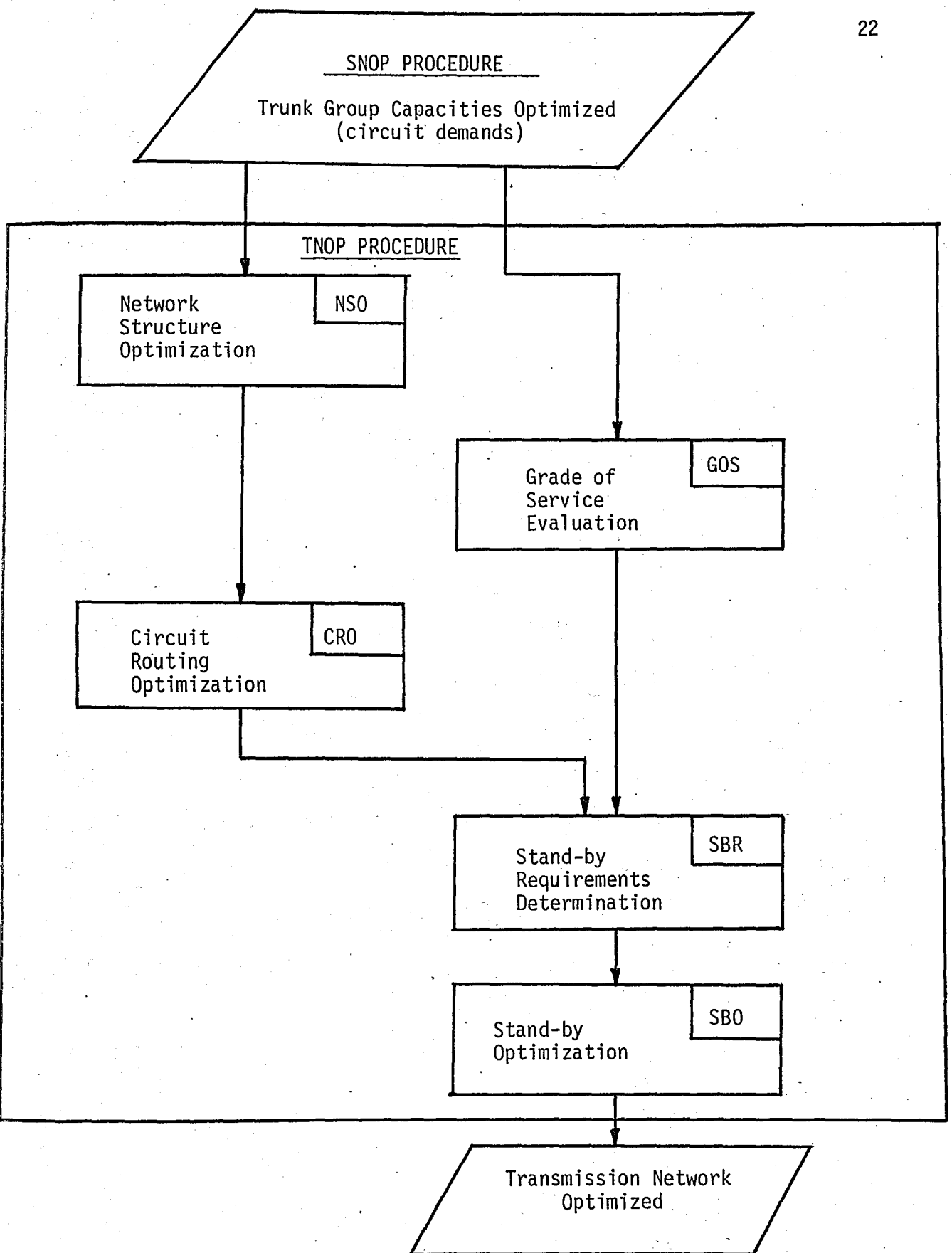


Figure 3.3 - Flowchart of the Transmission Network Optimization Problem.

required to maintain the grade of service in failure conditions. The studies by Cavellero and Tonietti [15], Kaptanoğlu and Evranuz [16] and Kaptanoğlu [17] analyze the problem of stand-by requirements determination.

The final module, which is the Stand-by Optimization (SBO) module, routes the stand-by requirements in the transmission network using the available capacities of the media, and if necessary, using transfer of technique at a transmission node. The requirements are met at minimum cost, considering all single failures of the media.

In this thesis, instead of optimizing the NSO and CRO modules independently, a simultaneous optimization procedure is adopted. The problem is to specify the links on which different transmission systems are to be installed, together with the type of the systems and the number of circuits to be installed on each such link. The objective is to minimize the cost of routing while meeting the point-pair circuit demands. This problem can be solved both as a development (no existing media) and as a capacity expansion (taking account of the existing network) problem. In the following sections, the optimization of the NSO and CRO modules are presented separately. This will bring a better insight to the solution procedure given in Chapter IV.

3.2.1 The Network Structure Optimization

The objective of the Network Structure Optimization (NSO) module is to determine the structure of the transmission network. This is achieved by routing the circuit demand between the switching nodes on the maximal transmission network, and deleting the transmission media

which are either under-utilized or expensive. The deletion of a transmission medium is only carried out if the connectivity degree of the nodes is not violated [4].

The resulting available transmission network (which is a subset of the inputted maximal network) is passed to the circuit routing optimization module for more accurate routing of the trunk group circuit demands upon it.

The major inputs of the NSO module are [4]:

- i) the maximal graph which comprises media that are either existing or proposed by the planner (each medium is represented by a link in the maximal graph),
- ii) the cost coefficients of each transmission medium,
- iii) the capacity limit of each medium,
- iv) the technology of each medium, whether analogue or digital,
- v) the threshold value for the minimum number of circuits on a medium,
- vi) the minimum required degree of each node. This should, in most cases, guarantee the connectivity of the graph. Node failures are considered to be negligible in comparison to media failure.

The main outputs of this module are:

- i) the available transmission media,
- ii) the capacities of the available media.

The members of the COST 201 Project have proposed an algorithm for the NSO module which uses the inputs stated above. The flowchart of this algorithm [4] is given in Figure 3.4 for a better understanding of the NSO module.

3.2.2 The Circuit Routing Optimization

The objective of the Circuit Routing Optimization (CRO) module is to determine the optimal routing of trunk groups in the transmission network. To increase the flexibility of the network against failures trunk groups can be multirouted, that is, more than one transmission path can be given to each trunk group. This is only performed when the associated cost is reasonable [21].

The links of the network in this module can have transmission media with mixed technology, that is, both analogue and digital. Such media are represented by a pair of links having common failure characteristics. The change of technique in a transmission node is represented by splitting the node into an analogue and digital node, and introducing a transfer link in between. The advantage of this representation is that only the cost coefficients for the links are required. The cost for transfer of technique in a node is described by a cost coefficient on the transfer link. The process of handling mixed technology is illustrated in Figure 3.5 [4,14].

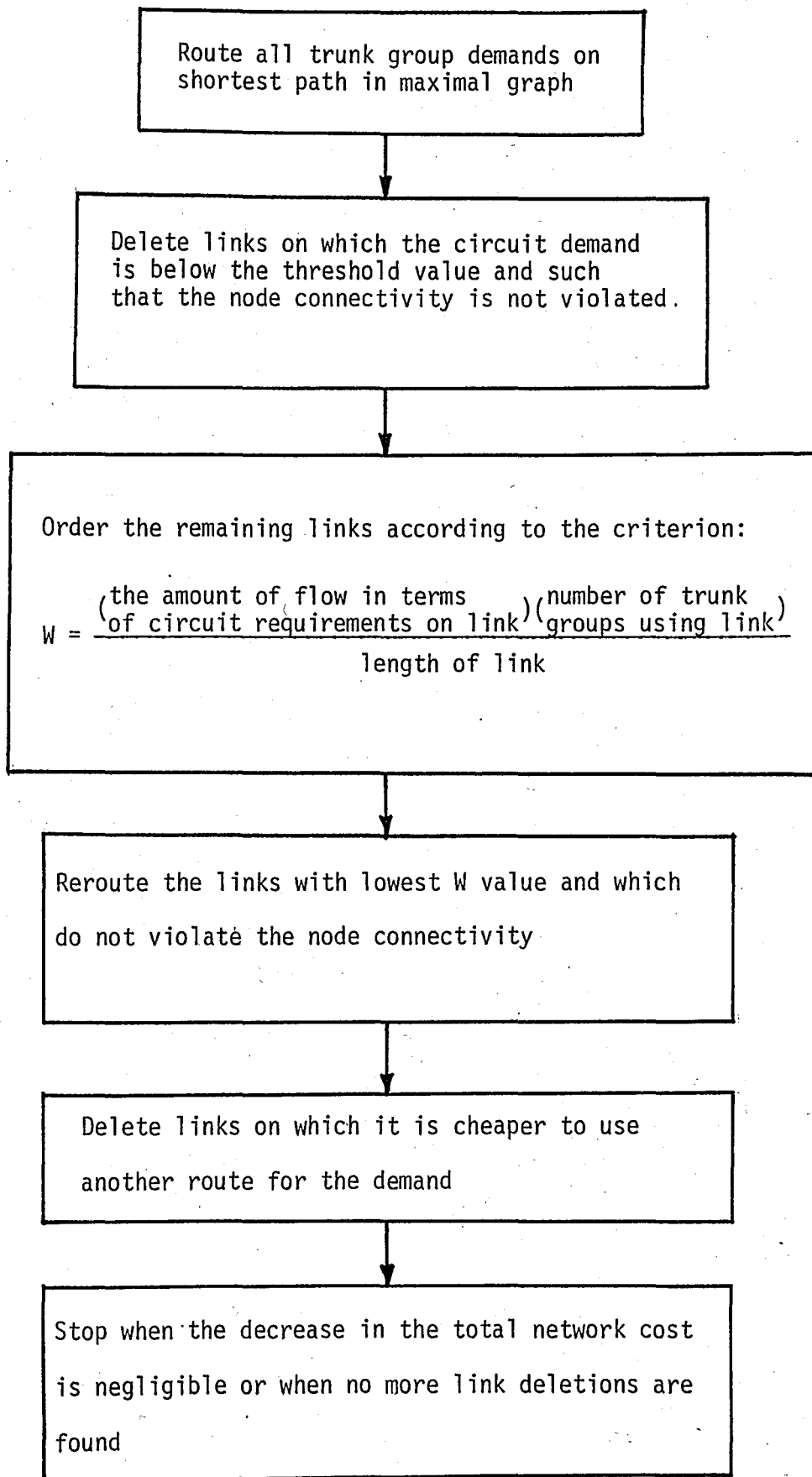
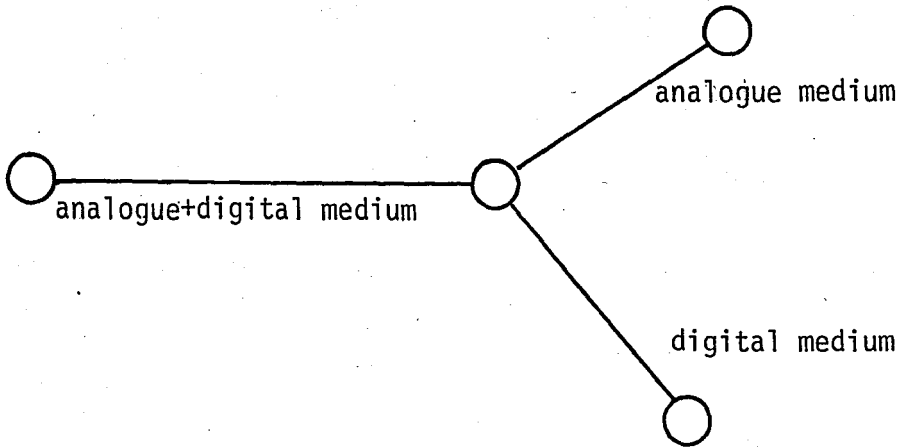
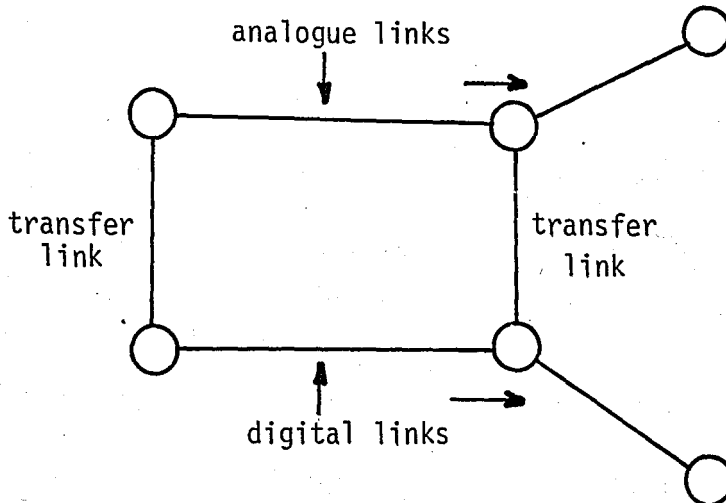


Figure 3.4 - Flowchart of the NSO Module.



a) The transmission network



b) The corresponding network model

Figure 3.5 - Handling mixed transmission media.

The flowchart of the CRO module proposed by the members of the COST 201 Project [4] is introduced in Figure 3.6 to give a better insight.

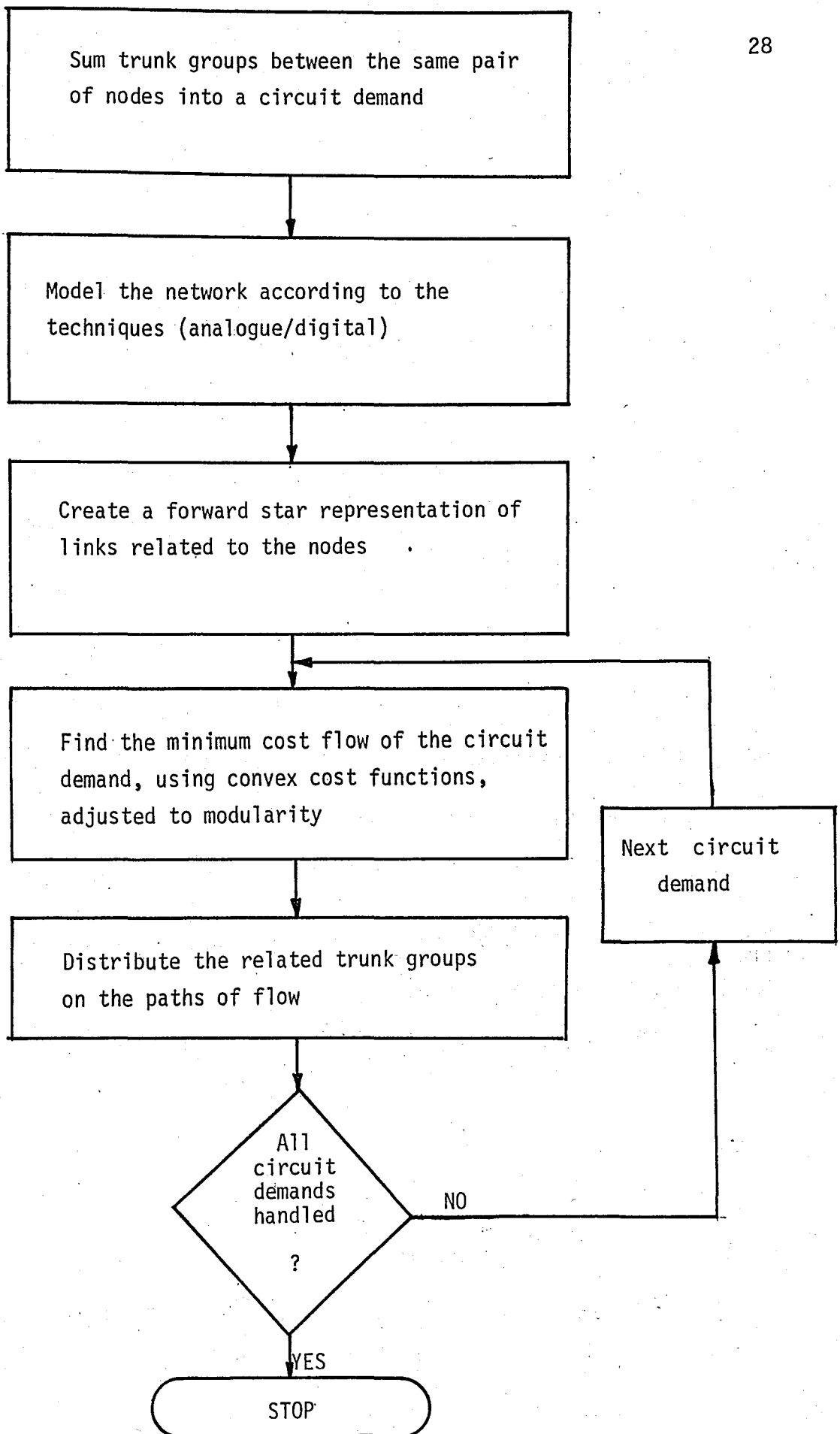


Figure 3.6 - Flowchart of the CRO Module.

IV. LITERATURE SURVEY

The first section of this chapter includes a general literature survey on the following topics:

- i. The multicommodity network flow problem (especially the minimum cost multicommodity flow problem),
- ii. Economies of scale in networks,
- iii. Planning and capacity expansion of telecommunication networks,
- iv. Static and dynamic design problems in planning and optimal routing in telecommunication networks.

The second section gives a more specific survey in the sense that, the different studies on the Network Structure and Circuit Routing Optimization modules within the framework of the COST 201 Project and presented.

4.1 A GENERAL SURVEY

Mathematical planning of telecommunication networks is typically concerned with the so-called multicommodity network flow problems. Some of the papers in literature concerned with multicommodity flows deal with

the problem of determining a minimum cost flow, while some others deal with a maximum flow problem. In both problems, there is a capacitated network and discrete commodities from certain sources are sent to certain sinks along the links of this network. This means that the flow on each link is a set of individual and interchangeable commodities.

The objective of the minimum cost flow problem is to minimize the total cost of carrying a specified amount of the flow from each source to its correspondent sink for all commodities.

On the other hand, maximum flow problems are basically concerned with how to get the largest amount of flow consisting of numerous commodities through a capacity limited network structure. Each commodity is defined by its unique source and sink with the resultant flows competing for capacity within the links of the network [22].

The multicommodity network flow problem can be formulated as below [23]:

Consider a network which has N nodes $\{1,2,\dots,N\}$ and M directed arcs $\{a_1,a_2,\dots,a_M\}$. Arcs a_1,a_2,\dots,a_ℓ ($\ell \leq M$) have capacities b_1,b_2,\dots,b_ℓ . Let there be K commodities and define x_{km} as the flow of commodity k in arc a_m . A source node s_k and a sink node t_k is associated with each commodity k . The constraints are:

1. Flows are nonnegative

$$x_{km} \geq 0 \quad , \quad \text{for all } k \text{ and } m$$

2. Capacity restrictions on arc a_m

$$\sum_{k=1}^K x_{km} \leq b_m \quad , \quad 1 \leq m \leq \ell$$

3. Flow conservation for commodity k at node n

$$\sum_{a_m \in B_n} x_{km} - \sum_{a_m \in A_n} x_{km} = \begin{cases} -f_k, & \text{if } n = s_k \\ 0, & \text{if } n \neq s_k \text{ and } n \neq t_t \\ f_k, & \text{if } n = t_t \end{cases}$$

where

f_k : the amount of flow of commodity k in the network,

B_n : the set of arcs terminating at node n ,

A_n : the set of arcs originating at node n .

For the minimum cost problem, the flows f_k are given and the objective is to minimize the total cost

$$\min z_1 = \sum_{k,m} c_{km} x_{km}$$

The maximum flow problem views the f_k as variables and has the objective

$$\max z_2 = \sum_k f_k$$

A general approach to solve the multicommodity flow problem is by decomposing it into several single commodity flow problems. For this reason, some papers on the single commodity flow problem are also given below.

The main computational procedures developed for solving the minimum cost flow problem are the primal-dual type algorithms of Ford and Fulkerson [24], Busacker and Gowen (described in [25]). These are dual methods in which feasible flows become available when the computations terminate. Fulkerson's "Out-of-Kilter" algorithm (described in [24]) is essentially

a primal method in that it can be started with a feasible flow or one becomes available at an early stage.

Another primal method for solving the minimum cost flow problems is given by Klein [26]. In the simple procedure he proposes, the feasible flows are maintained throughout. He also gives primal algorithms for the assignment and transportation problems. Klein uses the maximum flow routine of Ford and Fulkerson to find an initial flow and constructs a special network. He then uses a matrix multiplication method to find the shortest routes between every pair of vertices. There is also a negative cycle tracing routine in the algorithm he proposes.

One of the methods for solving the minimum cost flow problem is to find cycles of negative length in a marginal cost network [27]. These negative cycles indicate a change in the flows around the cycle, which will result in a reduction in the total cost. The detection and flow change around the negative cycles are the topics of Bennington's [27] paper. He claims that it is always possible to detect any negative cycle in the marginal cost network by using a shortest path algorithm from one node to all other nodes. He proposes an algorithm that finds negative cycles in this manner and which can be used in Klein's [26] method for solving the minimum cost flow problem. He presents also the computational results comparing this algorithm with the out-of-kilter algorithm.

Tomlin [28] uses the Dantzig-Wolfe decomposition principle to solve the minimum cost multicommodity flow problem. He states this problem as follows:

Consider the network $[N,A]$ with nodes $i = 1,2,\dots,n$ and directed arcs. Assign each arc a capacity $b_{ij} \geq 0$, and associate with each arc a cost $c_{ij} \geq 0$ per unit flow. For each commodity $k = 1,2,\dots,q$, denote the source s_k and the sink t_k , and let the required flow of commodity k be r_k (assuming a single source and sink for each commodity). The problem then becomes to meet the flow requirements and capacity constraints over an existing network at minimum cost.

Tomlin [28] formulates the minimum cost multicommodity network flow problem in both node-arc (where each commodity must satisfy the Kirchoff node conservation requirements) and arc-chain form, leading to very large linear programs.

Chen and De Wald [22] use the Dantzig-Wolfe decomposition principle in solving the maximum flow multicommodity flow problem. They first formulate this problem in node-arc form as a large-scale linear program. Then taking advantage of the special structure, the Dantzig-Wolfe decomposition principle is utilized to partition this problem into a master program and a set of sub-programs. Each sub-program is solved as a shortest route (chain) problem based on the original network with the arc distances being the dual variables of the master program. A shortest route can be determined by an efficient algorithm. This chain effectively labels the chain over which a positive amount of flow is to be directed. Finally, an appropriate amount of flow is transmitted over the chain in a fashion analogous to (and probably a generalization of) the single commodity flow augmentation procedure. The procedure utilized in this process for this purpose is referred to as the Backtracking Method, since it often requires the backtracking and reduction of previously allocated

chain flow. A criterion is also developed to test the flow at each iteration for optimality.

Hartman and Lasdon [23] present an algorithm for solving minimum cost or maximum flow multicommodity flow problems. By using the special structure of any basis matrix, the simplex method can be performed while maintaining the inverse of a working basis whose dimension is only the number of currently saturated arcs. Aside from multiplication by its inverse, all other simplex computations are performed using addition or graph theoretic operations. The algorithm is a specialization of the generalized upper bounding method for block angular problems.

An important feature of the telecommunications network planning problem is that the cost functions associated with installing transmission systems are concave, reflecting economies of scale [5,6,8]. These functions may be approximately decomposed into a fixed charge and a linear cost part. The fixed charge represents the initial investment cost of installing a transmission system (e.g., cable) on a link. The linear cost part, on the other hand, represents the cost of installing the circuits (e.g., wires in cables) of that system. Furthermore, it is also assumed that both of these costs depend on the length of the individual links (i.e., the actual distance between the two points joined by that link).

The paper by Yaged [29] presents a foundation for examining network economy of scale effects. The point-to-point nature of circuit demands and the dynamic nature of an evolving communications network are the important properties of the models studied in the paper. The effects of these two properties on measurements of network economy of scale effects are investigated.

Yaged states that a natural measure of the network scale effect is the network cost elasticity e , which is defined as the ratio of the fractional decrease in network average cost per circuit mile to the fractional increase in total circuit miles carried by the network. It is also noted that the average cost per circuit mile on a link decreases as the number of circuits increases. Hence, the maximum scale effect is given by $e = 1$, while $e = 0$ signifies no scale effect.

Yaged [29] uses a simple example to show that static studies, while providing useful insight, have little relevance to issues concerned with a communications network evolving over time. The paper describes static network models and relates network scale effects to transmission facility economies of scale. A dynamic model of network evaluation is also presented, which includes the point-to-point nature of demand and the sequential nature of facility installation decisions. This model can be used to evaluate the network scale effects arising from the availability of high capacity low average cost (gross investment) transmission facilities.

Like in many applications within the private and public sectors, planning for the expansion of capacity is of vital importance in communication networks [30]. The complexity of this expansion policy results from the strong dependency between the optimal routing of the demand and the optimal capacity expansion plan. The optimal routing at each period depends on the existing link capacities, and the expansion policy for any link depends on the routing decisions.

Luss [30] presents a literature survey on capacity expansion problems. He defines the basic capacity expansion problem as the problem which consists of determining the sizes of facilities to be added and

the associated times at which they should be added, as well as the types of facilities to be installed so that the present worth cost of all expansions is minimized.

The capacity expansion cost is usually concave, exhibiting economies of scale, that is, the average cost per capacity unit decreases with the expansion size [5,6]. Some commonly used cost functions are [30]:

i. the power cost function

$$f(x) = Kx^\alpha \quad \text{where } 0 < \alpha < 1, \quad x \geq 0,$$

ii. the fixed charge cost function

$$f(x) = \begin{cases} 0 & , \quad \text{if } x = 0 \\ A+Bx & , \quad \text{if } x > 0 \end{cases} ,$$

iii. or some combination of the two.

However in some applications, the expansion cost function is not concave. Suppose that different technologies are used to add facilities where

$$f(x) = \begin{cases} 0 & , \quad \text{if } x = 0 \\ A_1+B_1x & , \quad \text{if } 0 < x \leq a \\ A_2+B_2(x-a) & , \quad \text{if } a < x \end{cases}$$

and $A_2 \geq A_1+B_1a$. In that case, the cost function is piecewise concave, that is, concave in the range covered by any single technology. In fact, this is the case in telecommunication networks. Figure 4.1 [30] illustrates such a typical cost function.

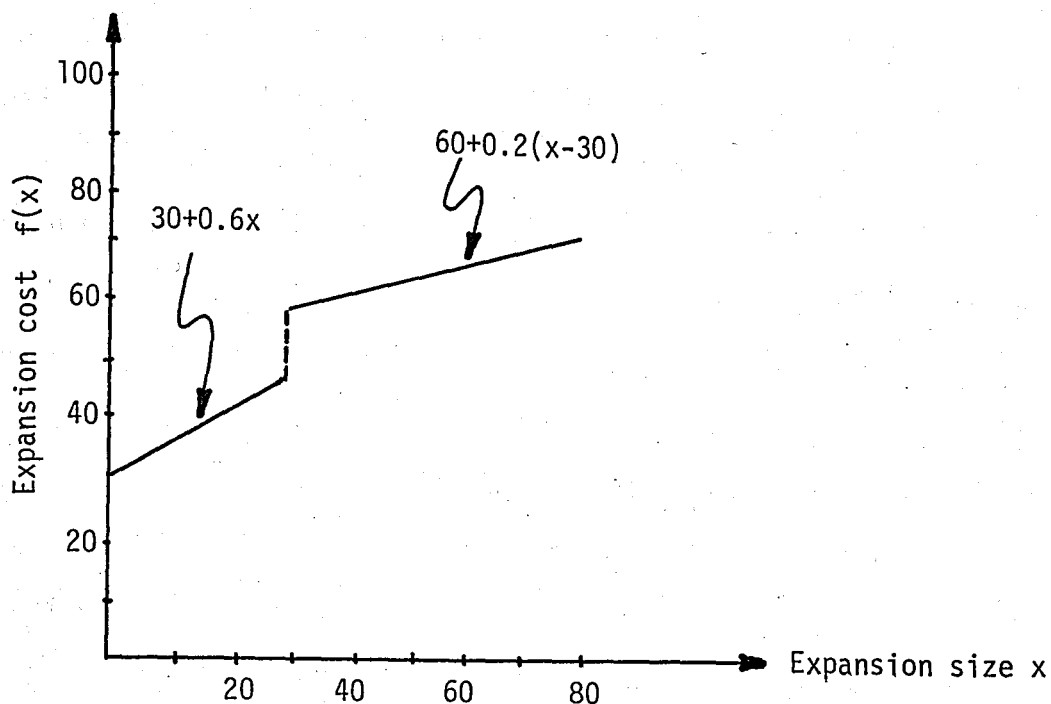


Figure 4.1 - A typical capacity expansion cost function.

In the papers by Yaged [31], Baybars and Kortanek [8], Ulusoy [32], Evranuz and Miraboğlu [10,11,12], and Evranuz and Aktin [13], economies of scale is taken into account while developing the algorithms.

Literature contains several papers on the planning and capacity expansion of telecommunication networks. Opposite to the operational planning methods, the model by Claus and Krätzig [33] allows a global cost-optimum network to be obtained, which is subject to a certain set of constraints.

The planning problem of Claus and Krätzig, which is taken from the work of the "Network Planning Department of the Telecommunication

Administration of the Deutsche Bundespost", is how to utilize and to extend the capacity of an existing buried cable network in such a way that all (future) traffic requirements are met by minimum costs. Hence, the decision variables in the model are the discrete digital systems (PCM) set up on existing cable lines and new cable links to be installed in the future.

This cost-optimum planning and capacity expansion problem is solved by using mixed-integer programming. Furthermore, the formulation takes account of the circuit capacity of the system, and path diversification required for reliability reasons.

In another study conducted by Hackbarth [34], a cost-optimum network structure for the planning of telecommunication transmission networks is calculated which provides sufficient capacity for the trunk groups required between the exchanges. For reasons of network reliability, each trunk group is split up depending on its size, and the parts are routed over several edge-disjoint paths.

It has been shown that this problem can be described by a multi-commodity network flow problem with a non-linear objective function. Furthermore, it was shown that, in long-term planning of telecommunication transmission networks, the objective function can be represented by a sum of continuous monotone increasing concave functions [34].

For the resulting concave minimization problem, Hackbarth shows that local minima can be calculated by a sequence of solutions to linear problems, and that a corresponding procedure can be described by a fixed point algorithm. The linear problem is solved in such a way that for each trunk group, the corresponding edge-disjoint paths having minimum

total lengths are calculated. The fixed point algorithm is used in a heuristic procedure to calculate a sequence of local minima with decreasing values of the objective function. For this purpose, a densely meshed initial network is systematically reduced by prohibition of edges.

In the network model Hackbarth [34] chooses an undirected graph, because the sum of all circuits required between two exchanges is taken as the demand, and the direction is unimportant in a transmission link [31].

One particular approach to communications network planning problem is to formulate it as a linear program as done by Mc Callum [35].

Mc Callum considers a communications network in which the nodes might be interpreted as switching nodes and the arcs as transmission links. The forecasts of circuit requirements between specified pairs of points are given. The requirements are called demands, and must be met by routing circuits along certain circuit paths or designs in the network. If an arc has insufficient capacity to provide for these demands, then its capacity can be augmented by the construction of new facilities.

Hence, the problem attacked by Mc Callum [35] is to determine the routing of circuits on designs, and the construction of new capacity so as to meet the demands at minimum cost. Cost is defined here as the sum of routing costs and construction costs.

Mc Callum considers the single-time-period version of this problem formulated as a linear program in arc-chain form, and applies the generalized upper bounding technique of Dantzig and Van Slyke.

In the planning and capacity expansion of telecommunication networks, the nature of the design problem becomes another important feature. Some of the papers contained in literature concentrate on the static design problem, while some others on the dynamic design problem.

In the static design problem, demands are taken to be constant over the time horizon of interest. This implies that all the investment decisions are executed at the beginning of the time horizon.

However, in the dynamic design problem, demands change over the time horizon. This leads to finding an optimal investment policy in which the time and capacity of each investment decision made over the time horizon is specified. The study of this problem is obviously more complicated and would require a heuristic approach.

The paper by Baybars and Kortanek [8], analyzes such a dynamic model for the facilities design problem in telecommunication networks.

Baybars and Kortanek [8] define the transmission facilities planning problem in telecommunication networks as a fixed charge multi-commodity flow problem, and state this problem as follows:

Given point-pair circuit requirements for each year in the planning horizon, find a minimum present value cost facility installation plan by specifying the type of transmission systems and the links themselves on which the systems are to be installed, as well as the number of circuits to be installed on each such link in each period of the finite planning horizon.

However, this transmission network optimization problem is considered on the switching network, instead of the transmission network itself in Baybars and Kortanek's [8] study. Therefore to increase the

grade of service, alternate routing is considered in place of multi-routing in transmission networks. The formulation also takes into account alternate transmission systems with different capacities, and the cost functions associated with installing transmission systems are taken as concave, reflecting economies of scale. The links are specified as high-usage and final choice in their formulation. But since in transmission networks a link may carry high-usage and final choice trunk groups simultaneously, it is insignificant to make such a distinction.

This transmission facilities planning problem is formulated as a mixed integer program, and a heuristic method is presented to solve this model.

In order to ensure system reliability (i.e., to avoid dependency on a specific transmission system), Baybars and Kortanek [8] assume that not all the transmission facilities on specific links will be of the same type.

In another paper by Baybars and Kortanek [19], again a heuristic approach to the transmission facility planning in telecommunication networks is presented. A procedure is developed for obtaining approximate optimal solutions over a three period planning horizon. It is assumed that the transmission supplies are unlimited, and there is only one alternate route for circuit assignment.

On the other hand, Evranuz and Mirabog̃lu [10,11,12] formulate the optimal planning of transmission facilities in telecommunication networks as a static design problem. Alternate transmission media, multirouting and economies of scale are also considered in their formulation. Three different models are developed and tested for comparison against each other.

The minimum cost routing in multicommodity network flow problems is formulated for static network models in the paper by Yaged [31], and an iterative technique for finding a local minimum to the problem is presented.

The cost of the network is modeled as the sum of the cost of providing a given number of channels on each link. Each link cost is assumed to be a concave function of the link size. This assumption, like in many other papers, is again based upon the fact that the transmission systems which can be installed on a link display economies of scale.

Yaged [31] shows that an optimal solution to the concave routing problem is a shortest path routing. Multirouting and the technologies of transmission systems on the same link are not considered in the formulation.

Another paper in which the optimal routing in networks is formulated as a static design problem is by Ulusoy [32]. Given the set of nodes and the demand between each pair of nodes, the problem is to select a set of arcs to route the flows through the network such that the demands are satisfied and the resulting total cost is minimum. Fixed charge and economies of scale are also taken into consideration in the developed heuristic algorithm which is based on the shortest path algorithm. Since the shortest path algorithms assume a linear cost function and the cost functions in the paper are taken to be concave with fixed charge allowed, four linear approximation techniques to the cost function are suggested so that a unit cost can be assigned to each arc. The arc cost functions f_{ij} are assumed to have the form:

$$f_{ij} = \begin{cases} 1.5x_{ij}L_{ij} & , \quad 0 \leq x_{ij} \leq 140 \\ 1.2x_{ij}L_{ij} + 42L_{ij} & , \quad 140 < x_{ij} \leq 220 \\ 0.6x_{ij}L_{ij} + 174L_{ij} & , \quad 220 < x_{ij} \end{cases}$$

where x_{ij} are the arc flows and L_{ij} are the distances of arcs.

Like in Yaged's paper [31], multirouting is not considered and only a single technology is taken into account.

A different approach to a large-scale network routing problem with nonlinear cost functions is described by Claus and Kleitman [36]. This approach involves a multistage construction process.

The specific problem that is considered in the paper concerns the optimal location of rented telephone lines in order to minimize the rental costs, given the properties of the rental rate structure. This is known as the "telpaking problem".

The problem under consideration takes the following form:

Given the set of requirements, each of which involves a pair of locations between which a communication line is desired, construct a network of telpaks and private lines which can accommodate the requirements at minimum cost.

This problem is one in which concentration of use along paths provides savings up to the capacity of telpaks [36]. The general pattern is that of routing many simultaneous requirements over routes on which costs are of a step function nature in their dependence on usage. Claus and Kleitman illustrate the step function nature of costs by the following graph.

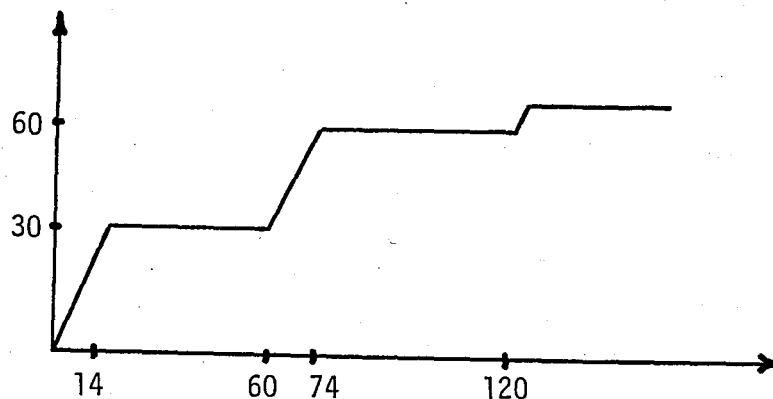


Figure 4.2 - Cost function for telpak renting.

According to Claus and Kleitman [36], an exact, minimum cost solution does not appear to be a feasible possibility for problems in the size range of interest here. Their approach consists of a number of heuristic steps, each of which is geared to rectifying the worst failings of the previous steps. They claim that the combined effect of all of these steps seems to be considerably more effective than any of them alone.

4.2 A SURVEY ON THE NETWORK STRUCTURE AND CIRCUIT ROUTING OPTIMIZATION PROBLEMS

As mentioned in Chapter III, the telecommunications network optimization problem is first divided into two sub-problems which are further

partitioned into modules. Two of these, namely, the Network Structure Optimization (NSO) and Circuit Routing Optimization (CRO) modules, constitute the topic of this thesis.

The methods for planning and optimization of telecommunication networks are developed within the COST (European Co-operation in Scientific and Technical Research) 201 Project which is conducted by 11 European countries, including Turkey. The basic objective of the COST 201 Project is to develop computer-based planning procedures to optimize the dimensioning of telecommunication networks by minimizing the total cost under specified constraints of quality of service [3,4].

One of the major problems that arise during the transmission network optimization is to find an efficient connectivity measure for the network. For this purpose, three different measures are proposed.

The first one is to consider the node degrees, that is, the number of edges coming in and going out the nodes [4]. But this is rather a weak measure, and there is no way to guarantee the connectiveness of the network. Consider the following example in which degree of node A is six, and degree of node B is seven.

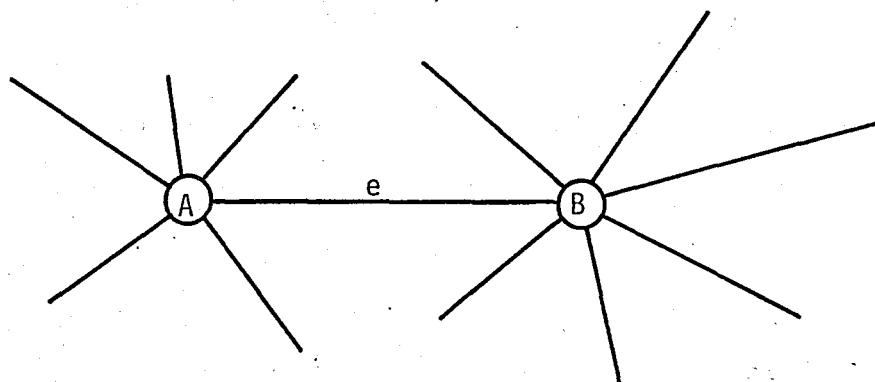


Figure 4.3 - A failing case for the connectivity measure on node degrees.

Suppose that a minimum node degree of three is used as a connectivity measure. If edge e is the proposed edge for deletion and the network is checked for connectivity, it will be seen that A will remain with a node degree of five and B with a node degree of six. Since they are both greater than the minimum node degree, e will be deleted, but the network itself will become disconnected.

For this reason, another protection measure is developed, which is to route the flow between the same node pair over several edge-disjoint paths [37] as illustrated in Figure 4.4.

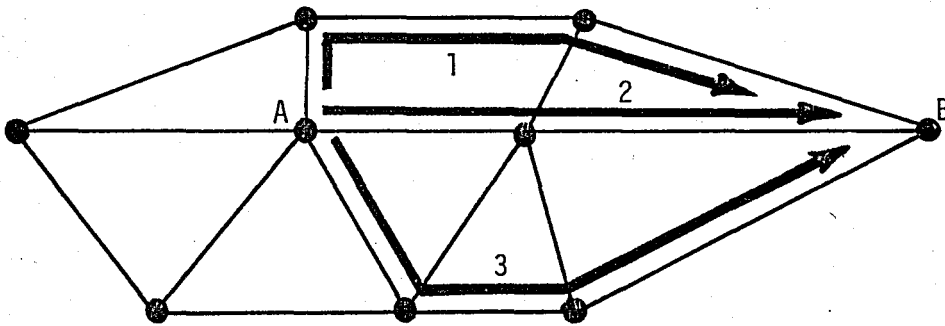


Figure 4.4 - An example of three edge-disjoint paths for the node pair (A,B).

A third connectivity measure is to use a flow augmenting algorithm which is equivalent to route a flow in a capacitated network. For this reason, a maximum flow algorithm is used between the nodes for which the connectivity will be checked. Each edge is assumed to have a capacity of unity [9].

Nivert and Noort [14], propose an algorithm for the network structure optimization problem which first routes the circuit demands on the shortest paths in the initial graph. Those edges with a flow below a threshold are deleted, unless node degree requirements are violated. Next, the media are ordered according to a criterion of efficiency. The least efficient links are subsequently rerouted onto cheaper paths, if available, and the rerouted links themselves are deleted. The algorithm terminates either when no more deletions are found, or when the network cost does not decrease significantly any more.

Evranuz has two papers on this problem. In his first paper [37], he gives an algorithm for solving the network structure optimization problem in which the trunk group requirements are routed in k shortest paths.

In his second paper [9], Evranuz proposes a heuristic algorithm in which the fixed charges and capacity limits on the links, and the existing network are taken into consideration. Also, different technologies are presented by parallel links in the network.

The circuit routing optimization problem is formulated as a multi-commodity flow problem with capacity constraints. For several reasons, this multicommodity flow problem is separated into a number of single commodity flow optimizations [14,21].

Nivert and Noort [14] introduce an algorithm in which both analogue and digital transmission media are considered. The algorithm also handles multirouting using a piecewise linear convex cost function.

Lindberg, et.al [21] take into account the maximum capacity constraint on each transmission medium, and reduce this capacity by an input

factor in order to always have some remaining capacity for the construction of the stand-by network.

The capacity constraints are handled using Lagrangean relaxation, and the problem is decomposed into several single commodity minimum cost flow sub-problems. A feasible solution is obtained by iterations where the Lagrange multipliers are changed by a subgradient technique. This means that if a capacity constraint was violated at one iteration, this link will get a higher cost coefficient at the next. However Lindberg, et.al. claim that, this alone is not sufficient to produce feasible solutions. Therefore, they use also a penalty cost which is given as input. As soon as the total flow on a link has reached the capacity limit, this penalty cost is added to the link.

Each sub-problem consists of routing one circuit demand. A shortest augmenting path technique is chosen to solve the minimum cost flow problem in this capacitated network. With this technique, it is necessary to use a shortest path algorithm a few times for each circuit demand. Since this is the innermost loop of CRO, an efficient algorithm is needed. Lindberg, et.al. [21] have chosen two of the shortest path algorithms from the paper by Dial, et.al. [38] for this purpose.

To increase the flexibility of the network against failures, Lindberg, et.al. also consider multirouting by using a piecewise linear convex cost function as in the paper by Nivert and Noort [14]. The first part of this function represents the real cost, while the slightly increasing second part serves to find a reasonable second path, and the third part is a penalty cost forcing multirouting.

Unlike the papers mentioned above, the study by Evranuz and Aktin [13] brings a new dimension to the transmission network optimization problem in the sense that, the network structure and circuit routing modules are optimized simultaneously. The study presents an approximate algorithm that can be used for large transmission networks, and which comprises more than one technology, parallel links, alternate transmission systems, the existing network, fixed charges, and economies of scale.

V. A SURVEY ON SHORTEST PATH ALGORITHMS AND A SPECIAL IMPLEMENTATION FOR CAPACITATED NETWORKS

This chapter has two main purposes. The first purpose is to present a brief survey on the different shortest path algorithms contained in literature, and to give some methods for storing large-scale networks in the computer. The second purpose is to introduce a special implementation of Dijkstra's shortest path algorithm for capacitated and parallel-edged network problems. This developed algorithm will be used in the solution procedure described in Chapter VI.

5.1 LITERATURE SURVEY ON SHORTEST PATH ALGORITHMS

In the analysis of transportation and communication systems, one major problem that arises naturally is to find the shortest, cheapest or fastest route between two points in the network. In other words, a routing which is minimum according to some criterion is required. To solve this problem, one needs an algorithm in which the shortest path between those specified points is found. Therefore, the shortest path algorithm is extensively used in the design of transportation and communication networks [39].

In most of the applications, the networks are very large and efficient algorithms are thus required. The literature contains several algorithms for the calculation of shortest paths in large networks.

Gilsinn and Witzgall [39] analyze and compare a set of labeling algorithms in which the shortest paths from one node to all other nodes in the network are calculated. The paper points out the importance of computer implementation technology and investigates different data handling (more specifically, list processing) techniques which can be used to improve the basic algorithms in this class.

The paper by Dial, et.al. [38] further extends the work of Gilsinn and Witzgall [39], emphasizing the fact that alternative list structures and labeling methods indeed exert a remarkably powerful influence on solution efficiency, and that the identity of the best of these methods depends upon the topology of the network and the range of the arc length coefficients. An additional significant result of the study is that the label-setting algorithm previously documented as the most efficient is dominated for all problem structures examined by the new methods.

The survey paper written by Dreyfus [40] treats five discrete shortest path problems which are:

- i. determining the shortest path between two specified nodes of a network,
- ii. determining the shortest paths between all pairs of nodes of a network,
- iii. determining the second, third, etc., shortest paths,

- iv. determining the fastest path through a network with travel times depending on the departure time, and
- v. finding the shortest path between specified endpoints that passes through specified intermediate nodes.

Furthermore, it gives theoretical computational bounds for each class. Also, some algorithms are modified to yield efficient procedures.

Dantzig [41] presents an effective method for obtaining the shortest route from a given origin to all other nodes in a network or to a particular destination point. The method starts by fanning out from the origin. Its special feature is that this fanning out is done one point at a time and the distance assigned is final.

Glover, et.al. [43,44] develop a new hybrid solution algorithm which integrates the features of label-setting and label-correcting methods in an effective manner. It is also reported that this new algorithm has proved notably superior to the best label-setting and the best label-correcting algorithms on all problem topologies tested.

5.2 NETWORK REPRESENTATION

The first sub-section of this section contains some definitions of the terms that are commonly used in describing the shortest path algorithms. Data storage problem in large-scale network is analyzed in the second sub-section. The terminology adopted in this section is taken from the studies by Gilsinn and Witzgall [39], and Dial, et.al. [38].

5.2.1 Terminology

A network consists of a finite set N of nodes and a finite set E of arcs, and can be denoted as $G(N,E)$. An ordered pair (v,w) of nodes corresponds to each arc $e \in E$, where v is the starting or beginning node and w is the terminating or ending node.

A directed path, or simply a path, is a finite sequence of arcs $P = \{e_1, e_2, \dots, e_n\}$ such that for each $i = 2, 3, \dots, n$, arc e_i begins at the end of arc e_{i-1} . P is called a path from node v to node w if arc e_1 starts at v and arc e_n terminates at w . A path P from v to w is called a circuit if $v = w$, that is, if the beginning and ending nodes of P are the same. A path for which $e_i \neq e_j$ whenever $i \neq j$ is called arc-simple. Hence, in such a path no arc appears more than once.

Assume that to each arc $e = (v,w)$ in $G(N,E)$ there corresponds a non-negative length denoted by $\lambda(e)$ or $\lambda(v,w)$. Then, a path which is defined as

$$d(P) = \sum_{i=1}^n \lambda(e_i)$$

can be assigned to each path P . If $d(P)$ is the minimum length of any path between two specified nodes, then P is called a shortest path.

5.2.2 Data Storage Problem in Large-Scale Networks

A network having M arcs and N nodes may be represented in the computer in several ways. Since this representation directly affects the performance of the developed algorithms applied to the network, efficient techniques have to be developed.

5.2.2.1 The Matrix Representation

One way of representing a network in the computer is by considering all possible node pairs (v,w) . This is called the matrix representation. Corresponding to each pair, the following information is stored:

$$l(v,w) = \begin{cases} l(e) & , \text{ if nodes } v \text{ and } w \text{ are connected by arc } e \\ 0 & , \text{ if } v = w \\ \infty & , \text{ if no arc exists between } v \text{ and } w \end{cases}$$

where $v = 1,2,\dots,N$ and $w = 1,2,\dots,N$. Hence an N by N matrix is required for storage. But if N is quite large and M is small compared to N^2 which is the case in many applications, most of the entries in the matrix will be infinity. Therefore, storing the entire matrix will be inefficient or even impossible - if N is large - in this case.

Consider the network in Figure 5.1.

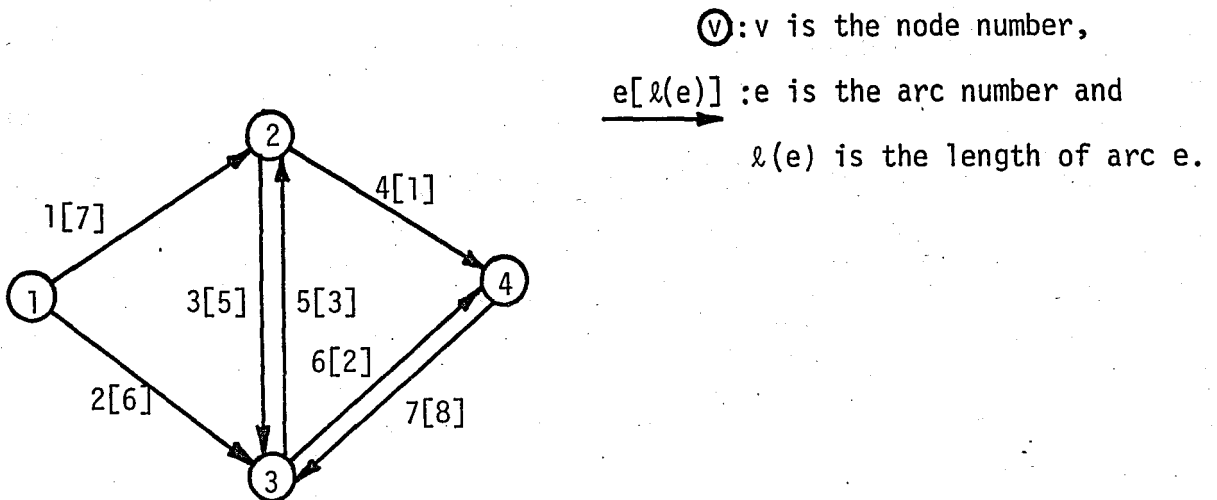


Figure 5.1 - Example network.

This network may be represented in matrix form in the following manner:

Table 5.1 - Matrix Representation of the Example Network

Node No. \ Node No.	1	2	3	4
1	0	7	6	∞
2	∞	0	5	1
3	∞	3	0	2
4	∞	∞	8	0

5.2.2.2 The Ladder Representation

Another way, which is the ladder representation, requires a list of all arcs in the network. This list must contain the beginning node, the ending node, and the length of each arc $e \in E$. Thus, $3M$ storage locations are required.

The network in Figure 5.1 may be represented in ladder form by the following lists:

Table 5.2 - Ladder Representation of the Example Network

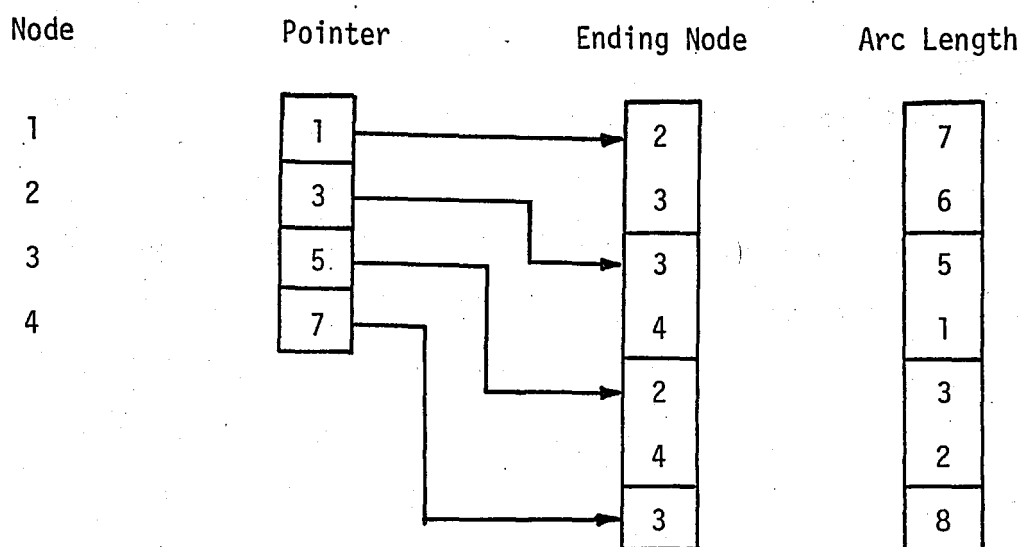
Arc No.	Beginning Node	Ending Node	Arc Length
1	1	2	7
2	1	3	6
3	2	3	5
4	2	4	1
5	3	2	3
6	3	4	2
7	4	3	8

5.2.2.3 The Forward Star Representation

The most popular way of storing a network is to use a linked list structure [50]. In this method, the arcs in the network are ordered by their starting nodes so that all of the arcs which begin at the same node appear together. Therefore, only the ending node and the length of each arc have to be stored. Also, to indicate the block of computer memory locations for the arcs beginning at each node, a pointer is kept. Thus, the pointer has N entries; each shows the beginning position of node v in the arrays containing the information about the arcs. In other words, each entry indicates the storage location of the first arc starting at each node. Hence, $N + 2M$ units of memory are required. If the nodes are numbered sequentially from 1 to N , then one can determine the last arc starting from v as the arc immediately preceding the first arc starting at node $v+1$. (If the nodes do not naturally appear in such a fashion, it will be necessary to convert them to this form, since efficient operation of the algorithms requires that nodes be numbered sequentially.) The representation of a network in which all the arcs starting at the same node appear together is called the forward star form, and the forward star of a node v , denoted by $FS(v)$ consists of all the arcs starting at v , that is, $FS(v) = \{(v,j):(v,j) \in E\}$. Furthermore, if the arcs that belong to the forward star of each node are sorted by ascending length, then this will be called the sorted forward star form.

To store the example network in Figure 5.1 in forward star form, the following lists are required:

Table 5.3 - Forward Star Representation of the Example Network



Note that, the lists headed "ending node" and "arc length" are the same in both ladder and forward star forms. The third list (labeled "beginning node") required by the ladder representation is replaced in the forward star form by the list labeled "pointer".

5.3 BASIC LABELING METHODS

The two fundamental ways of storing a network in the computer were described in the previous section. Corresponding to these, there are two basic methods for treating shortest path problems: matrix methods which use the matrix representation of the network, and labeling methods which utilize ladder or forward star representations [39].

Matrix methods yield the shortest distances between all pairs of nodes simultaneously. Since they require large computer storage locations, their application is restricted to relatively small networks. The computational

effort of matrix methods depends only on the number of nodes and is independent of the actual number of arcs present in the network. Therefore, if N is quite large and M is small compared to N^2 , these methods tend to be less efficient than labeling methods. For this reason, in handling large and sparse networks labeling methods are preferred.

Labeling methods for computing shortest paths can be divided into two general classes as label-correcting and label-setting which will be explained in the following sub-sections. A detailed description of these algorithms can be found in the studies by Aktin and Evranuz [45,46].

Some additional definitions have to be given before going into the labeling methods.

In the context of directed networks, a rooted tree, or simply a tree, is a network $T(N_T, E_T)$ together with a node r (called the root), such that each node $v \in N_T$, except r , is accessible from r by a unique arc-simple path in T . Alternatively, the network $T(N_T, E_T)$ is a tree if:

- i. every node $v \in N_T - \{r\}$ is the end of exactly one arc in T ,
- ii. r is not the end of any arc in T ,
- iii. there are no circuits in T .

A rooted tree T is called a minimum tree or shortest path tree (in the larger network $G(N, E)$ under discussion) if T contains all nodes of G accessible from r , and if for each node $v \in N_T$, the unique path in T from r to v is a shortest path from r to v in the network G [38,39].

Each step of the labeling algorithms can be characterized by a tree T rooted at node r . Therefore when the algorithms terminate, the shortest

paths from r to all nodes accessible from r are obtained. Since the manner in which T is handled directly affects the performance of the algorithms, efficient techniques have to be developed.

Gilsinn and Witzgall [39], and Dial, et.al. [38] present various implementation techniques which will improve the efficiency of the basic labeling algorithms.

5.3.1 The Label-Correcting Method

The typical label-correcting method starts with any tree T routed at r and "corrects" T until no further improvement or enlargement is possible. That is, the arcs in E_T are exchanged, augmented, or updated in such a manner that the unique path from r to v in T is replaced or shortened. But until termination, there is no guarantee that this new path is a shortest path.

Label-correcting methods work for negative arc lengths as long as there are no circuits of negative length in the network $G(N,E)$.

5.3.2 The Label-Setting Method

The label-setting method, on the other hand, starts out with the tree T consisting of r alone, and at each iteration augments N_T by one node $v \in N$, and E_T by one arc $(u,v) \in E$ in such a manner that $u \in N_T$, $v \in N_T$, and the unique path from r to v in T is a shortest path. Hence at each step, T is a minimum tree for all nodes in T . A label-setting method terminates when all arcs in E which have their starting endpoints in N_T also have their ending endpoints in N_T , or - if the goal was to determine the shortest paths only to a subset S of nodes - when all nodes in S are in the tree.

It should be noted that label-setting methods work only for non-negative arc lengths.

5.4 A SPECIAL IMPLEMENTATION OF DIJKSTRA'S SHORTEST PATH ALGORITHM FOR CAPACITATED NETWORK PROBLEMS

A telecommunications network, being capacitated and parallel-edged, has quite an interesting structure. Since it is capacitated, a flow value which has to be updated from time to time is associated with each edge. For this reason in handling such a network, the edges become more important than the nodes [45,46]. One other problem arises from the parallel edges in the network. Although the algorithms that are presented in the studies by Gilsinn and Witzgall [39] and Dial, et.al. [38] calculate correct path lengths when multiple edges exist for the same node pair, they are unsuitable for this problem since they calculate shortest paths over the nodes in the network. Therefore by using the algorithms given in these papers, it will be hard to keep track of the edges on the shortest path, and unless a new array is kept to store this information, it will be impossible to understand by which edge a certain node on the shortest path was reached. But keeping a new array is undesirable because of its extra memory requirement. Therefore, an edge number processing algorithm is more suitable for a capacitated and parallel-edged network problem in which the edges are more important than the nodes [45,46]. For this reason, an algorithm which is in fact a special implementation of Dijkstra's shortest path algorithm was developed.

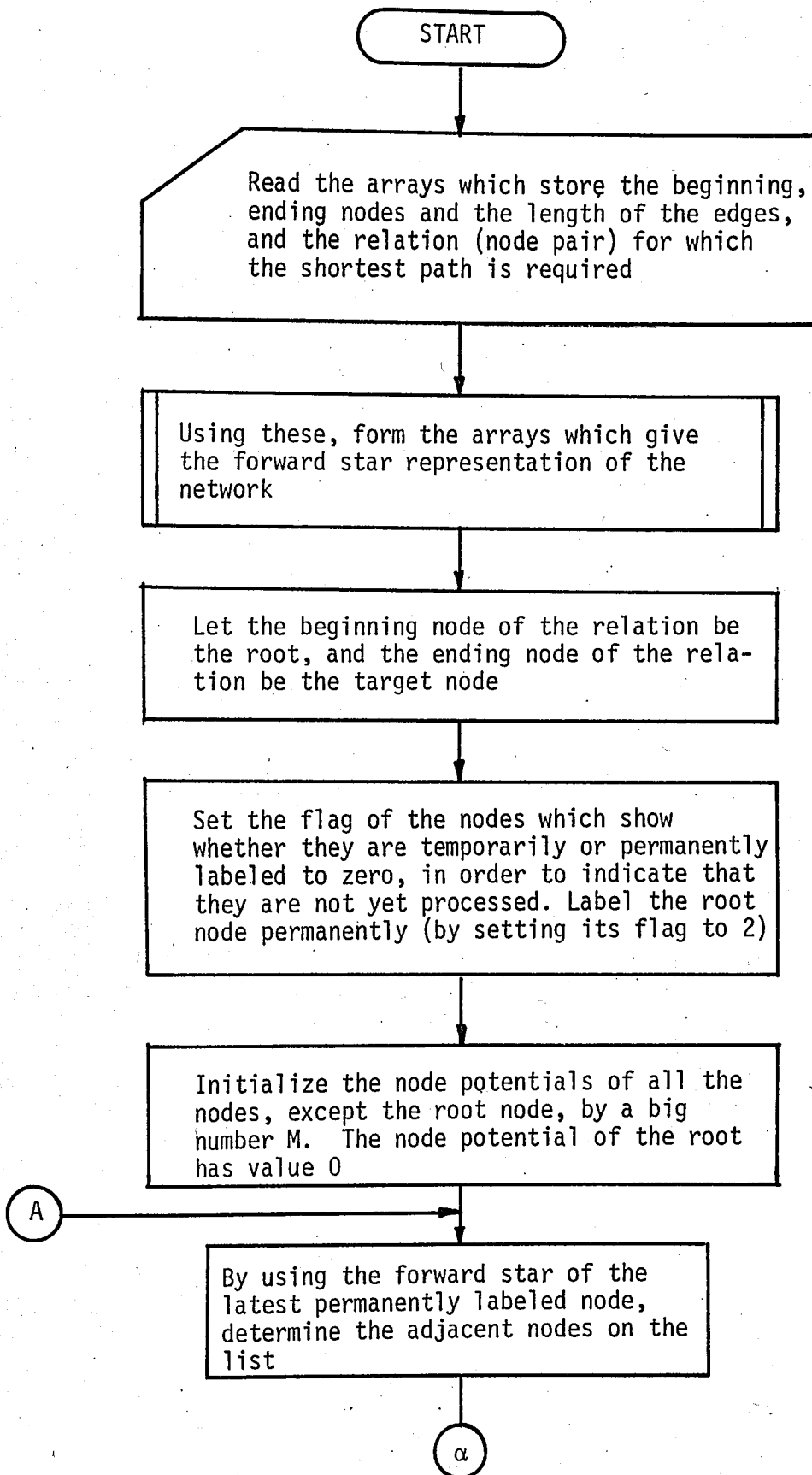
5.4.1 The Characteristics of the Developed Algorithm

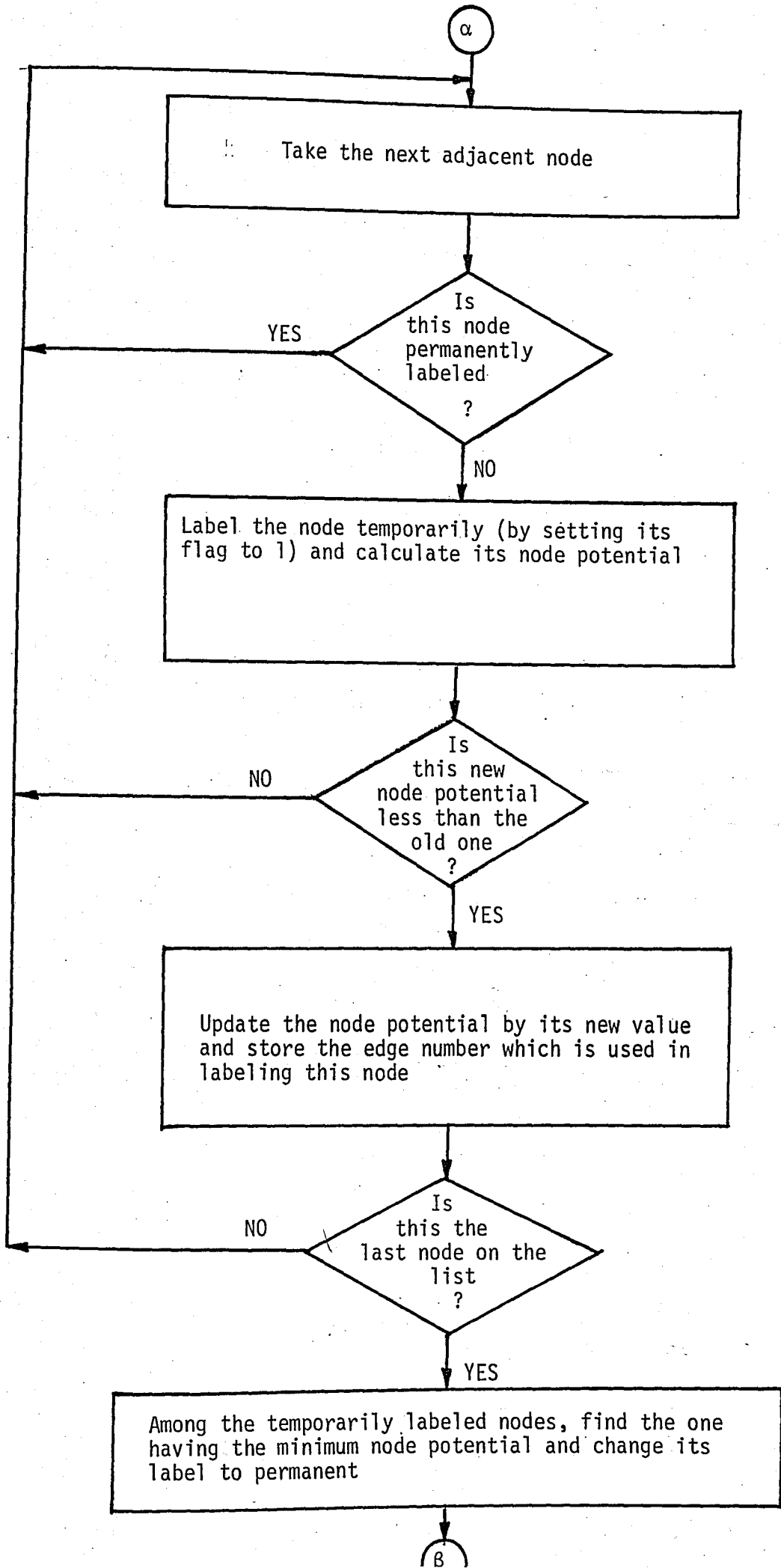
The characteristics of the developed algorithm can be summarized as follows [45,46]:

- i. By processing the edge numbers, it will be easy to handle capacitated edges,
- ii. Since for each intermediate node on the shortest path one will be able to understand by which edge that certain node was reached, parallel edges won't cause any trouble,
- iii. Multirouting and circuit routing will be handled very efficiently,
- iv. Since only the shortest path between two given nodes is required, the algorithm includes a stopping criterion. Also, a warning mechanism is developed for informing the user whenever the network becomes disconnected,
- v. Forward star representation of a network is employed also in this new implementation.

The program of the developed algorithm written in Fortran language appears in Appendix C.

Note that this algorithm can be used in any capacitated, parallel-edged network.





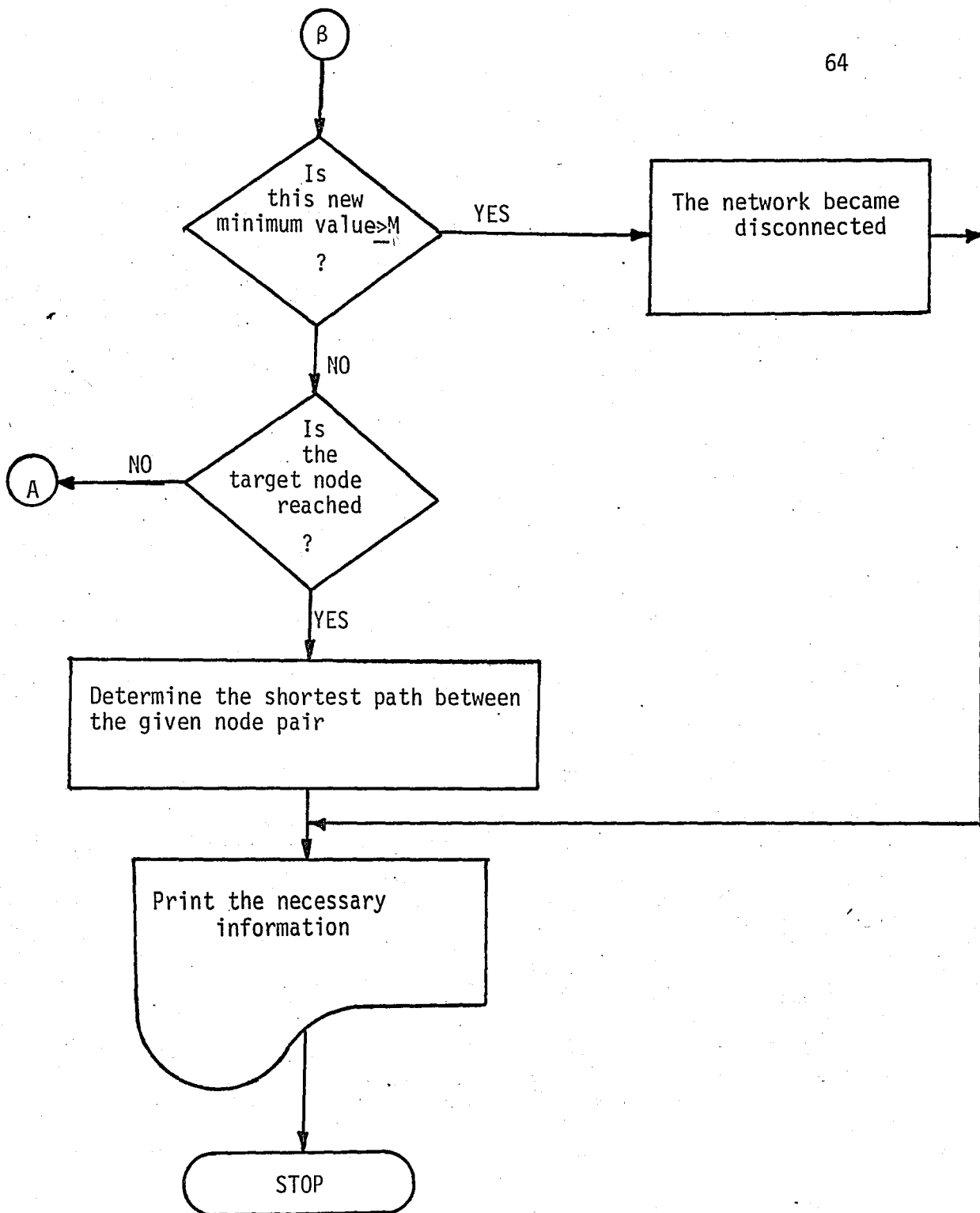


Figure 5.2 - Flowchart of the shortest path algorithm used in the solution procedure.

5.4.3 A Performance Comparison of the Developed Algorithm and Some Other Algorithms

The developed algorithm was compared with two of the different implementation techniques given in the Gilsinn and Witzgall study [39]. However, because of the reasons stated below, this comparison does not have a general quality. It was just performed to determine roughly the place of the developed algorithm among some others contained in the literature. The reasons can be stated as follows [45,46]:

- i. The structures of the developed algorithm and the algorithms given in Gilsinn and Witzgall [39] are different. The first one processes edge numbers, while the others process node numbers during the calculation of shortest paths,
- ii. The aim of the developed algorithm is different than that of the others given in the Gilsinn and Witzgall study [39]. In the developed algorithm, only the shortest path between the given node pair is needed. However in the others, the calculation of a shortest path tree whose root node is specified by the user is required. That is, finding the shortest paths from one node to all other nodes in the network are intended,
- iii. It was not possible to separate the time required for printing the outputs from the computer times taken. Because of the differences mentioned in (i) and (ii), for the algorithms given in the Gilsinn and Witzgall study [39], the arrays containing the predecessor node of each node, and the distance of each

node from the root have to be printed. However, for the developed algorithm, it is sufficient to print the matrix which stores the edges in the shortest paths of the given relations. For this reason, an actual timing comparison cannot be made.

The first of the techniques chosen from Gilsinn and Witzgall study [39], namely C2, has been applied to a label-correcting method, and employs a FIFO sequence list on which nodes appear at most once and which is treated in a rotating manner through the use of two pointers u and v . Pointer u points to the entry whose forward star is to be examined next and v is the position of the last node added. As u moves down the list, there is unused space at the top of the list above u which may be used for new nodes when space at the bottom of the list is exhausted.

The second technique, S2, is applied to a label-setting method, and contains a list which is partially ordered in a tree-sorted form. The elements of such a list may be considered as nodes in a binary tree in which the label of the predecessor of any node never exceeds the label of the node, that is, $d(p(v)) \leq d(v)$. The labels of nodes in any path in the binary tree are linearly ordered, but nodes on different paths are non-commensurable.

Both techniques, C2 and S2, employ the forward star representation of a network.

The comparison was made for three different test networks. The first of these has 10 nodes and 16 edges, the second 10 nodes and 21 edges, while the third has 77 nodes and 126 edges. The last two networks are real transmission networks. Table 5.4 gives the timings for the

three algorithms as applied to these different networks. From the figures given in the table, SHPTH, the developed algorithm, was found to be superior among the others, since it is the one that fits to the solution procedure presented in Chapter VI in the most effective manner.

Table 5.4 - Results of the Timing Experiments for the Three Algorithms

Network	Timing (seconds)		
	C2	S2	SHPTH
1 (N=10,M=16)	10 $\sum_{i=1} t_i = 13$	10 $\sum_{i=1} t_i = 13$	8 $\sum_{j=1} m_j = 7$
2 (N=10,M=21)	10 $\sum_{i=1} t_i = 15$	10 $\sum_{i=1} t_i = 14$	15 $\sum_{j=1} m_j = 9$
3 (N=77,M=126)	$t_1 = 74^*$	$t_1 = 34^*$	181 $\sum_{j=1} m_j = 111$

Note that,

t_i = the time required to calculate the shortest path tree whose root node is specified as i ,

m_j = the time required to calculate the shortest path between the given node pair (specified as j).

* Due to the size of the problem, it is the time obtained only by taking $r = 1$.

VI. PROBLEM FORMULATION AND THE SOLUTION PROCEDURE

This chapter states the characteristics of the problem together with the main assumptions of the model used. It also provides a detailed description of the solution procedure, and presents the minimum cost flow algorithm which plays an important role within the framework of the procedure.

6.1 CHARACTERISTICS OF THE PROBLEM

The characteristics of the problem can be stated as follows:

- i. The problem of simultaneously optimizing the network structure and the circuit routing in transmission networks can be formulated as a capacity expansion problem.

The transmission facilities planning problem, as stated by Baybars, and Kortanek [19], is to find a minimum cost facility installation plan by specifying the type of transmission systems, and the links themselves on which the systems are to be installed, as well as the number of circuits to be installed on each such link. The point-pair circuit requirements are given as input.

The decision variables of the problem considered here are the different transmission systems set up on existing links, and the new transmission media to be installed in the future. The point-pair circuit requirements (demands) must be met by routing circuits along certain circuit paths, and if a link has insufficient capacity to carry these demands, then its capacity can be augmented by the construction of new facilities. Hence, cost is defined here as the sum of routing costs and construction costs.

In this thesis, besides a capacity expansion problem, a development problem is also considered. In the development problem, the existing network is not taken into account, and a transmission network is developed throughout the solution procedure which starts with a maximum possible network. In fact, these two problems are closely interrelated, and consideration of the existing network with fixed cost components of existing media equaling to zero, changes the development problem to a capacity expansion problem.

- ii. The optimal routing of circuit demands is treated as a multi-commodity network flow problem.

It is known that message flows have definite origins and destinations, and a demand (in number of circuits) is associated with each flow between these node pairs. The problem is to send these discrete items through a capacity limited network structure. The property that each message flow can be defined by its unique source and sink makes the routing problem a multi-commodity flow problem, in which the resultant flows all share

the same channels and compete for capacity within the links of the network [22,47]. In this specific transmission network problem, a commodity corresponds to a message flow between a node pair.

Furthermore, the objective, being to minimize the total network cost while meeting these circuit requirements, distinguishes the problem as a minimum cost multicommodity network flow problem.

- iii. Fixed costs are taken into consideration during the solution procedure.
- iv. As mentioned by Evranuz and Aktin [13], one of the most important features of the problem is that the link cost functions associated with installing transmission systems are piecewise concave, that is, concave in the range covered by any single technology. This is an indicator of the validity and importance of economies of scale in the problem. Hence, as capacity increases, the average cost per capacity unit decreases [5,6].
Furthermore, the cost function of a link can be decomposed into a fixed cost component and a variable cost component. The fixed cost component depends on the length of the link, whereas the variable cost component depends both on the length and the number of circuits in the transmission media [8,13].
- v. The model considers also alternate transmission systems for each link. Table 6.1 which is taken from Baybars and Kortanek [8] gives the installation costs of three alternate transmission

systems, as well as the installation costs of their respective circuits, and the circuit capacity of each system.

Table 6.1 - The Cost Components and Capacity of the Alternate Systems Used in the Model

System	Fixed Cost (dollars)	Variable Cost (dollars)	Capacity (no. of circuits)
1	530,000	3,100	30
2	870,000	1,070	90
3	1,400,000	277	270

Figure 6.1 illustrates the behaviour of this cost function.

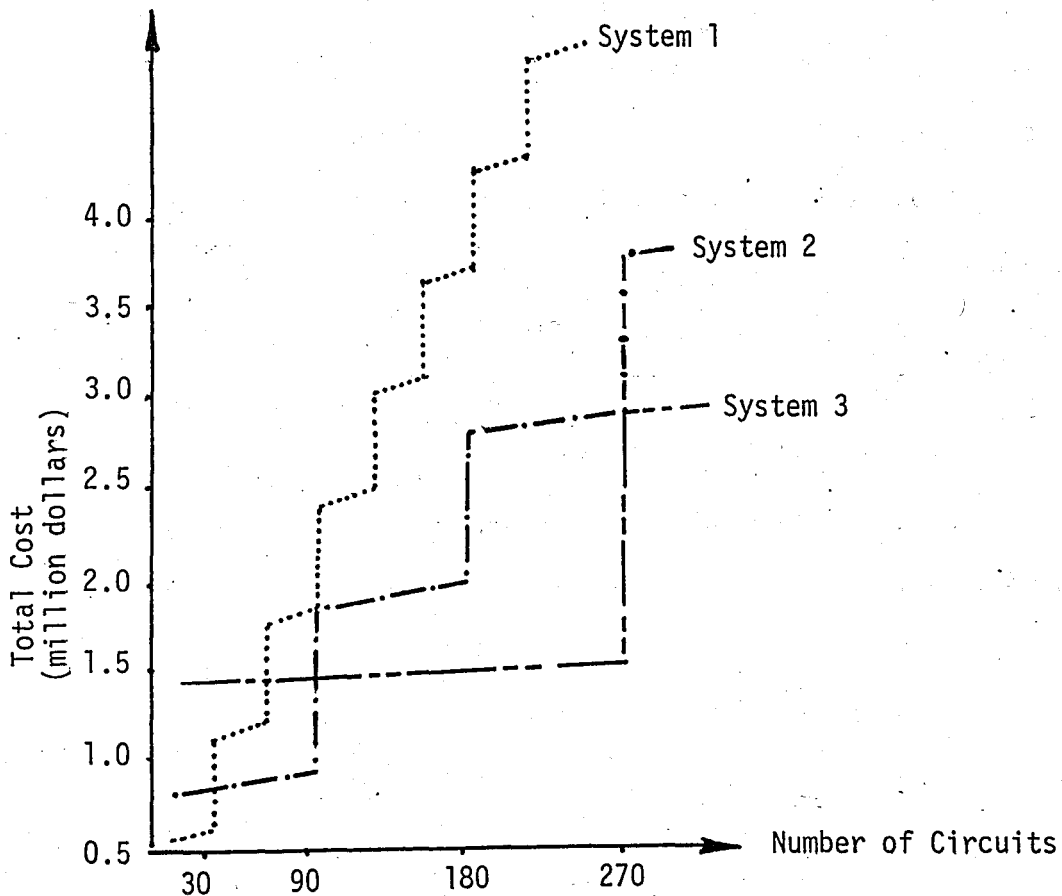


Figure 6.1 - The cost function of three alternate transmission systems.

Consider a link where there is no existing media. Suppose that a flow of 40 is assigned to this link. The question is whether to install two systems of capacity 30, or one system of capacity 90 on this link.

As it can be seen from Figure 6.1, the cost of installing two systems of capacity 30 and therefore using five units of one of these systems is much higher than selecting a system with capacity 90. Looking at the figure, it is seen that until and including 30 units, system 1 is cheaper. In the range between 31 and 90 units system 2 is the cheapest, and from 91 units system 3 is the most economic one.

The data in Table 6.1 will be used in the developed model.

vi. The costs of links calculated during the solution procedure are in the most general form, that is, there is a linear relationship between the cost and length of a link. This is the case if the transmission media are cables. What happens if other media, such as microwave radios or satellites are used?

In the case of a microwave radio, intermediate stations are required at certain locations on a link. For example, if there is a link of hundred kilometers, there may be intermediate stations at every twenty kilometers on this link. But specifying the location of these stations are rather complex, and the decision of location highly depends on the power of the station and the topology. Therefore in this case, the cost of a link doesn't exhibit a linear relationship with the distance and there will be jumps in the cost function, whenever a station is located.

In the case of a satellite, on the other hand, from the point the signal reaches the ground station, costs may have a linear relationship with the distance of the links. For example, suppose that Washington and Ankara are connected through a satellite. Then, Istanbul will be connected through a cable or microwave system to Ankara in order to have a contact with Washington, and this will bring a linear relationship to the cost function of the link between Istanbul and Ankara if a cable system is used.

Note that any type of link cost function can be used in the model.

- vii. The technology of a transmission medium can be analogue, digital or mixed. These different technologies are also considered in the model.
- viii. Since each link cost function has three discrete system capacities (30-90-270) associated with it, the model takes also modularity into account. Modularity, which is to take the size of trunk groups in multiples of a fixed module size, can be performed both in analogue and digital media [4].
- ix. The model can handle the parallel edges in the network.
- x. There are capacity limits on the links.
- xi. Since message flows can pass in either direction, the links of the network are considered as undirected.
- xii. The problem is modeled as static, which means that the projected demands will be met by the target network.

6.2 ASSUMPTIONS OF THE MODEL

The main assumptions of the model can be stated as follows:

- i. There is no analogue/digital conversion, meaning that the changes of A/D techniques in a transmission node are not taken into account.
- ii. Since a maximum possible network is taken, there is only link deletion. Adding links is not considered in the model.
- iii. The developed model does not concern multirouting, which is to route the circuit demands over diverse paths. But the reliability of the network is considered implicitly by the nature of the developed algorithm such that, even if a media on the path of a certain relation fails, the relation can be connected through a sequence of links which carry the circuit requirements of other relations.
- iv. In the solution procedure, once a link is fixed, it is not further considered as a candidate link for lowering the system on it, or for deletion.
- v. In the minimum cost flow algorithm, if the remaining capacity of a link is zero, or if the capacity of a link is all used, the capacity of this link is not highered to the capacity of the next system. In these cases, the network may become disconnected which will be traced by the algorithm, and rerouting may not be realized.

6.3 THE SOLUTION PROCEDURE

The problems of optimizing the network structure and routing the circuit demands on this network were separately explained in Section 3.2. As stated in COST 201 Project Report [4], the general trend is to optimize these problems individually. However, the procedure presented in this section will perform a simultaneous optimization.

As mentioned in Section 6.1, this optimization problem is formulated both as a development and as a capacity expansion problem. These two problems are explained in detail in Sections 6.3.1 and 6.3.2, respectively. However, since one is a special case of the other, a general procedure is introduced in this section.

The general procedure of simultaneously optimizing the network structure and circuit routing in transmission networks is illustrated in Figure 6.2.

The multicommodity flow nature of the problem was explained in Section 6.1. In the solution procedure, this multicommodity network flow problem is decomposed into several single commodity flow problems, and each of them handles one circuit routing. That is, in each subproblem, the circuit demand of only one relation (node pair) is routed.

The solution procedure consists of two major steps. In the first step, the link cost function which was given in Figure 6.1 is approximated as illustrated in Figure 6.3, and the circuit requirements are routed with the assumption that the capacity of transmission systems are used in certain ratios [13]. Since multirouting is not taken into consideration, this ratio is taken as one. However, if there is multirouting,

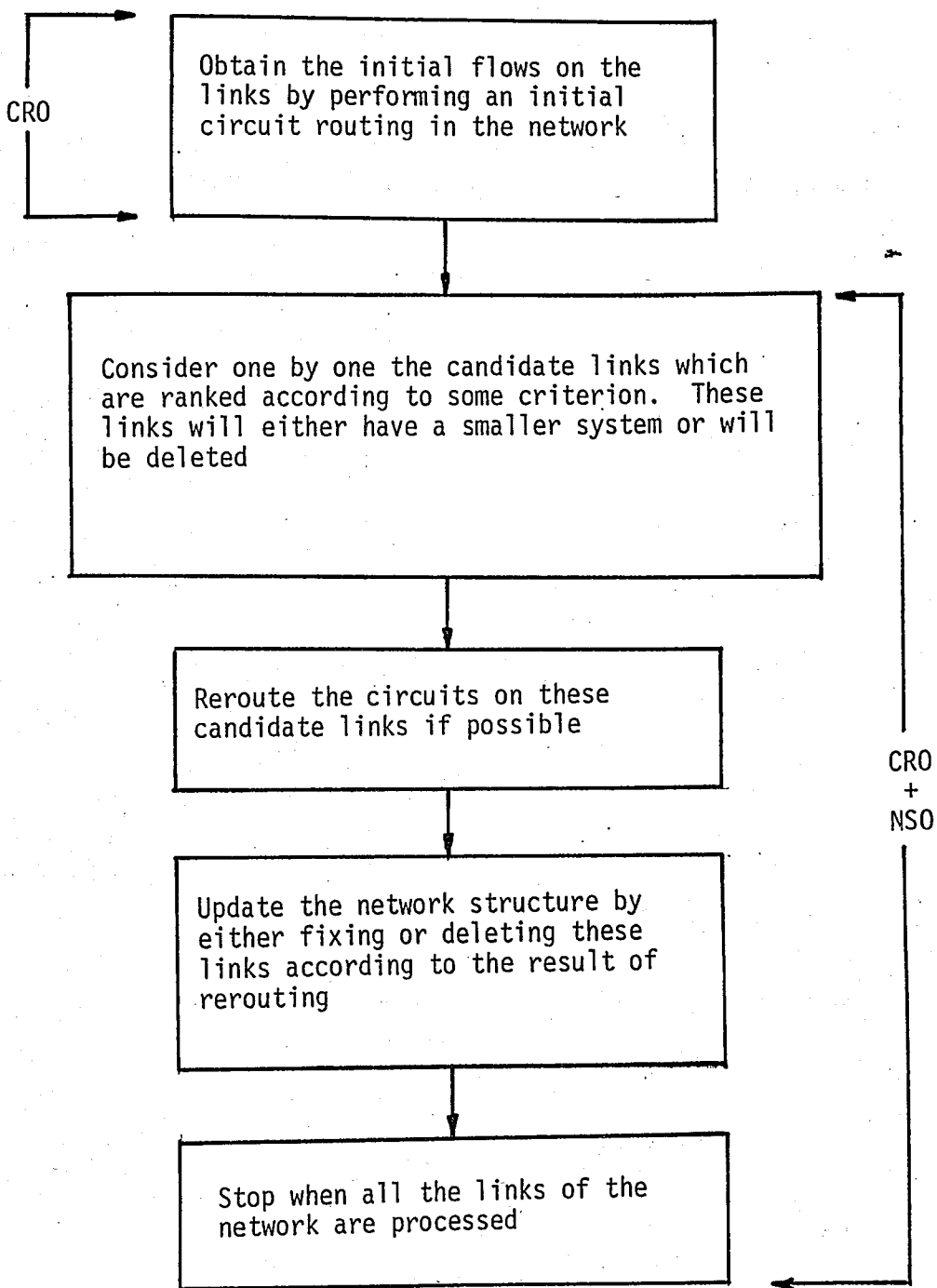


Figure 6.2 - A general glance at the problem.

different ratios can be given to each diverse path such that the sum of all ratios equals to one.

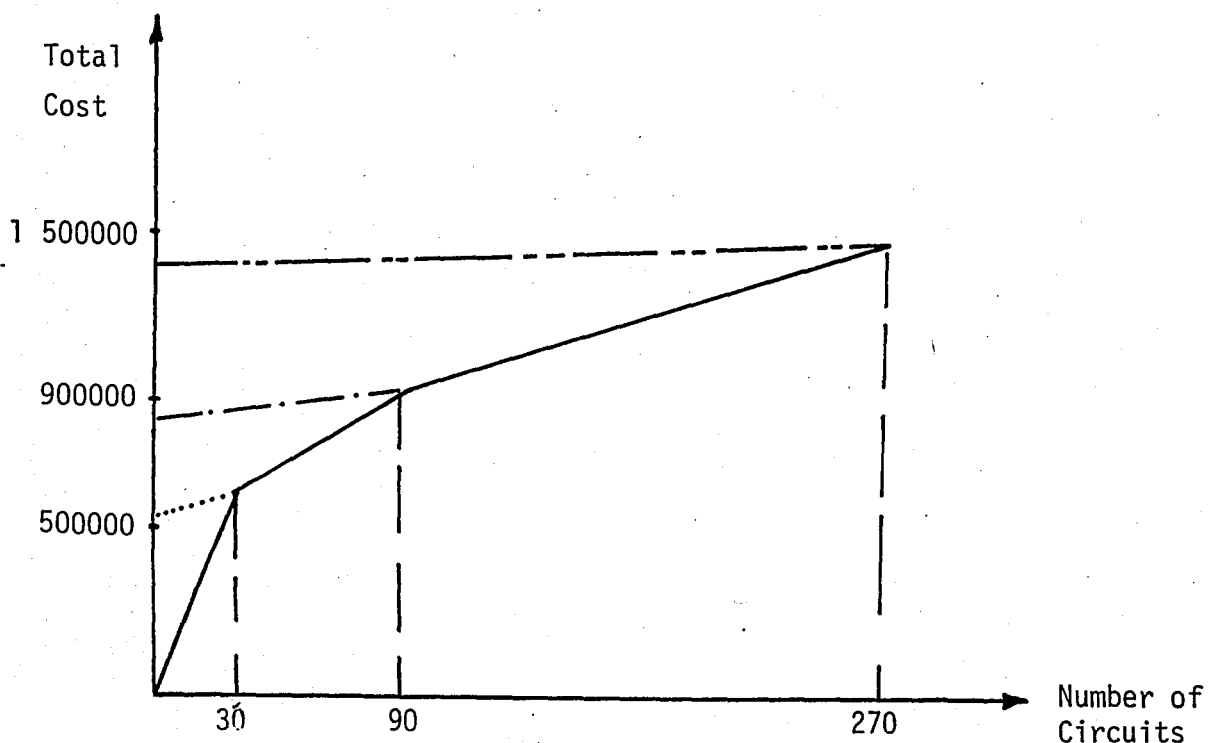


Figure 6.3 - The approximate link cost function.

These initial routings are obtained by using the shortest path algorithm presented in Section 5.4, which is a special implementation of Dijkstra's shortest path algorithm.

However in this algorithm, instead of using the actual distances between the node pairs, the approximated link cost values are used. These cost values are in fact obtained as a result of a first order linear interpolation. Using the cost values is necessary in order to consider alternate transmission media, since by employing the length of links which are constants, different systems cannot be considered

during the routing. These approximate cost values are calculated from the slopes in Figure 6.3 as follows:

$$\text{COST}_i = \left[\frac{\text{CVT}_{\ell_s} - \text{CVT}_{\ell_s-1}}{\text{CAP}_{\ell_s} - \text{CAP}_{\ell_s-1}} \right] \times \text{MD}_i \quad (6.1)$$

where

$$\text{CVT}_{\ell_s} = \text{FCOST}_{\ell_s} + [\text{UVC}_{\ell_s} \times \text{CAP}_{\ell_s}] \quad (6.2)$$

Here,

- ℓ_s shows the type of the current system installed on link i ,
- MD_i is the length of link i ,
- CAP_{ℓ_s} is the capacity of system ℓ_s ,
- FCOST_{ℓ_s} is the fixed cost of installing system ℓ_s ,
- UVC_{ℓ_s} is the unit variable cost of system ℓ_s , and
- CVT_{ℓ_s} is the total cost of the current system installed on link i .

The circuit demands which are sorted in descending order are routed by the use of this shortest path algorithm. During the routing process, if the flow on a link exceeds the capacity of the current system on that link, the next greater system is installed on the link.

The first step ends when all the circuit demands are routed. Then, the initial flows and the real cost values associated with each link are calculated. The real total cost of link i is computed as:

$$\text{RECOST}_i = [\text{FCOST}_{\ell_s} + [\text{UVC}_{\ell_s} \times \text{XFLW}_i]] \times \text{MD}_i \quad (6.3)$$

where

- XFLW_i is the total amount of flow on link i .

The initial total network cost is also calculated.

In the second major step, by using a criterion for handling the links, a more realistic cost comparison is made [13]. Associated with each link i , the following criterion is calculated:

$$V_i = \left[\frac{\text{FCOST}_{\ell_s} + [\text{UVC}_{\ell_s} \times \text{XFLW}_i]}{\text{XFLW}_i} \right] \times \text{MD}_i \quad (6.4)$$

The V_i values are sorted in descending order, and the link having the maximum V value is chosen. This is the candidate link which either will have a smaller system, or will be deleted from the network. That is, if a feasible flow pattern having a smaller total network cost is obtained after rerouting the total circuit demands on this link, the link will either have system $\ell_s - 1$ if $\ell_s \neq 1$, or will be deleted if $\ell_s = 1$. The connectivity of the network is also checked during the rerouting process. The V value of a fix or deleted link is set to infinity so that it won't be chosen later on.

For the rerouting process, there are two alternatives. The first one, which is proposed by the COST 201 Project members [3], is to reroute all the flow between the endpoints of the candidate link. This may be easier and may require less computer time, however a routing which is far from being minimum may be obtained. The following example illustrates such a case:

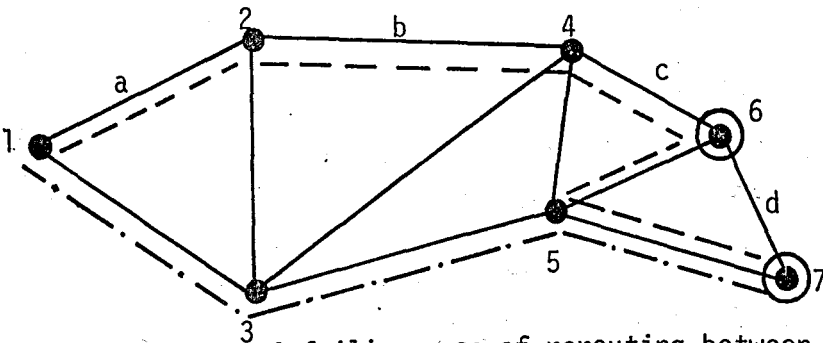


Figure 6.4 - A failing case of rerouting between the endpoints of a link.

Suppose link d is the candidate link, and it is on the path of relation $(1,7)$. That is, the path of node pair $(1,7)$ is $\{a,b,c,d\}$. And suppose that after rerouting the total flow on link d between the endpoints $(6,7)$, path --- is obtained for relation $(1,7)$. However, if the relation was rerouted as a whole, path ---- which has a smaller cost could have been obtained. But, by rerouting between endpoints $(6,7)$, this minimum path can never be taken into account.

Therefore, a better way of rerouting is to take relation by relation, that is, to perform this process between the source and sink nodes of relations using the candidate link [9,37]. Consider another example:

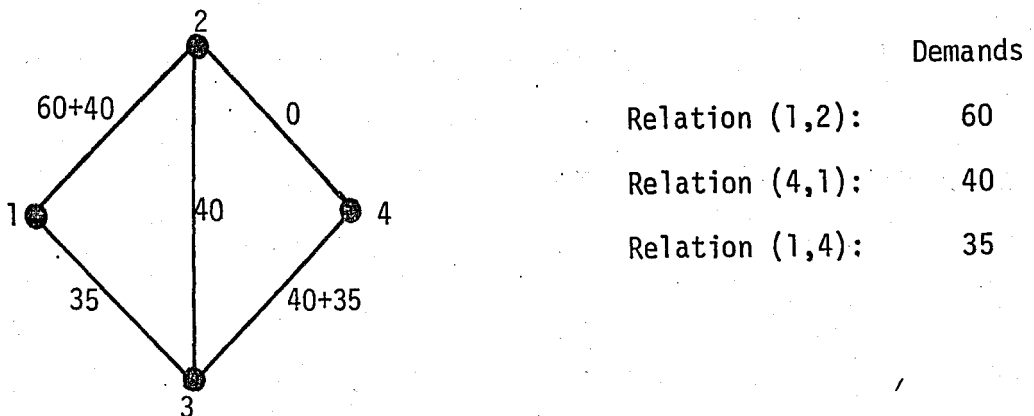


Figure 6.5 - A better way of rerouting (a).

Suppose relation $(1,2)$ has the path $\{(1,2)\}$, relation $(4,1)$ has the path $\{(4,3),(3,2),(2,1)\}$, and relation $(1,4)$ has $\{(1,3),(3,4)\}$, and the initial routing is as shown. Now, suppose link $(4,3)$ is the candidate link. Then, the relations using that link will be found, and the corresponding flows will be decreased from the links on the path of those relations as shown:

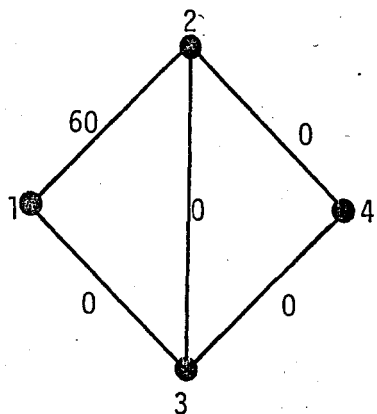


Figure 6.6 - A better way of rerouting (b).

After the flows on the links are decreased, the remaining capacities are calculated and the network is ready for rerouting. In the example above, relations (1,4) and (4,1) will be rerouted.

The rerouting of circuit demands is performed by the minimum cost flow algorithm of Busacker and Gowen (described in [48], and [49]). A detailed explanation of this algorithm is given in Section 6.3.3. Within this algorithm, again the shortest path algorithm of Section 5.4 is used.

After the rerouting process, the total network cost is calculated. If this new cost is less than the old cost, then the rerouting is valid and the system on the candidate link is updated. Of course, all the cost figures and flow values, and the systems on all the other links will also be revised. This process continues until all the links of the network are processed, and no further improvement can be made.

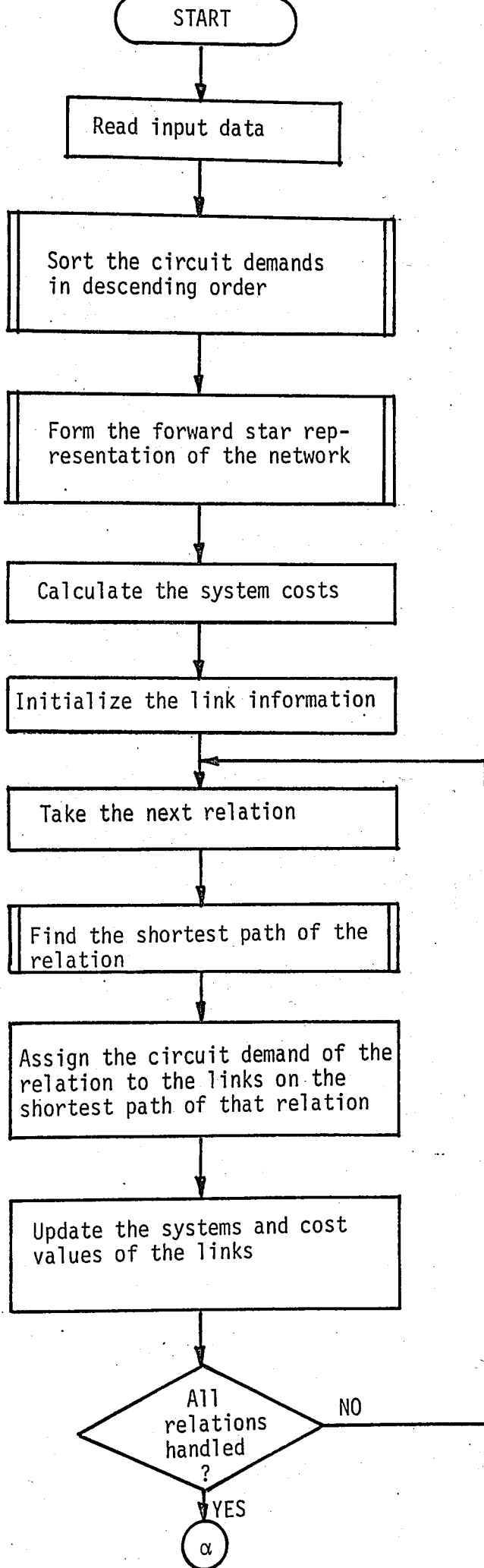
The flowchart of the solution procedure is given in Figure 6.7.

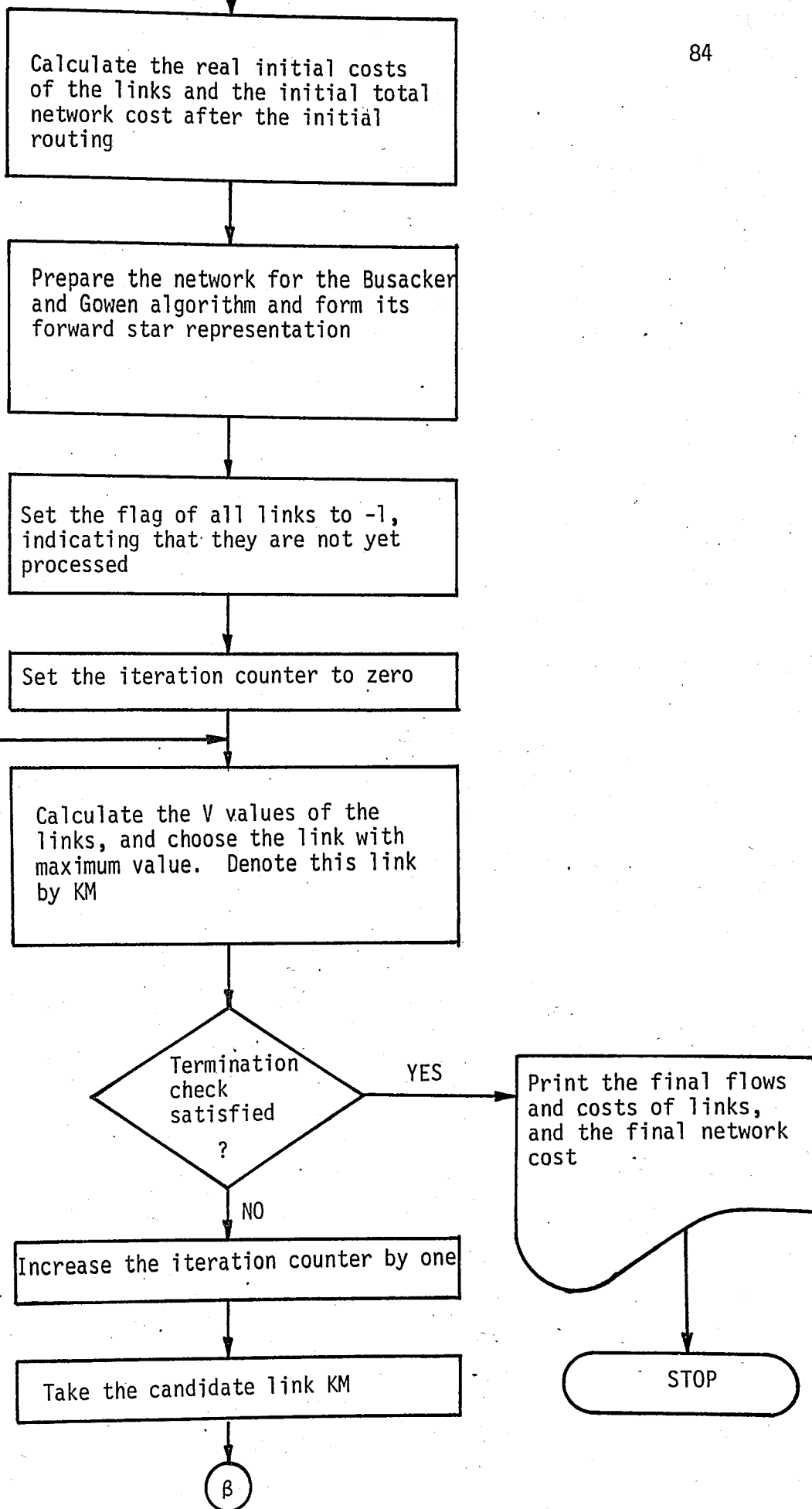
6.3.1 The Development Problem

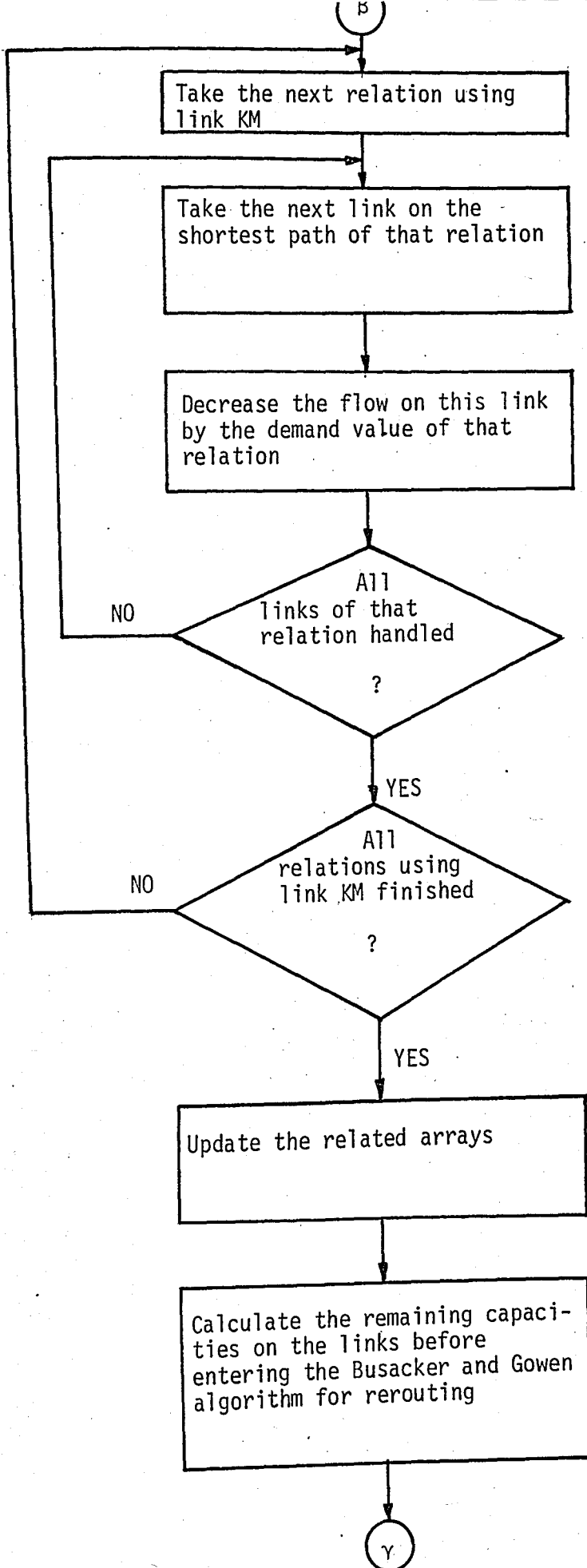
The major difference between the development and capacity expansion problems is in the initial network which is given as input. The first problem considers a maximum network in which there are no existing media. The second problem, on the other hand, takes into account the existing network. The flowchart in Figure 6.7 is the general solution procedure which is applied to both problems. However, as it is stated in Section 6.3.2, the capacity expansion problem requires some variations in this procedure.

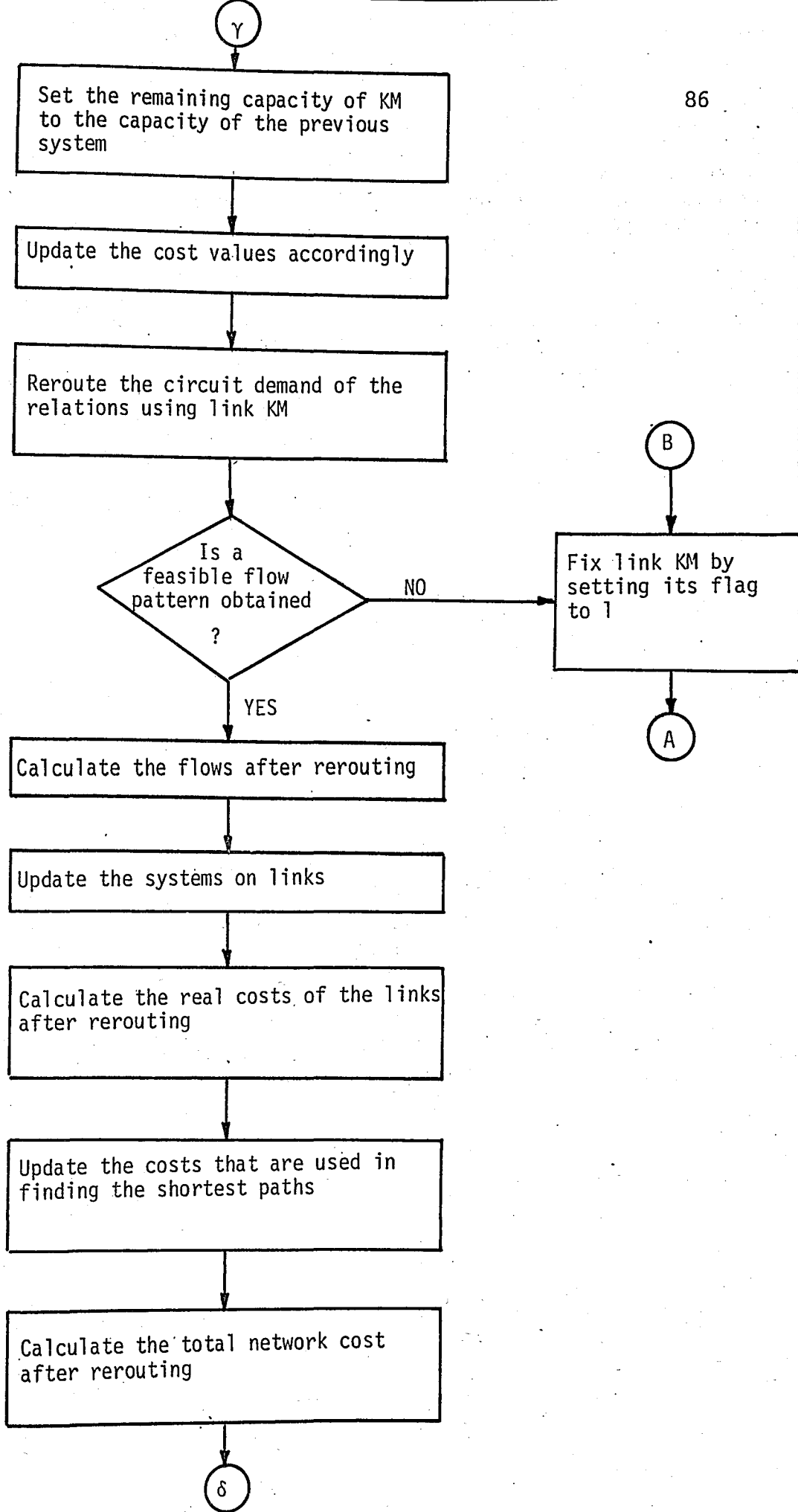
The procedure for the development problem starts with a network in which the existing capacities on all links are taken as zero. In the initialization stage of the algorithm, it is assumed that system 1 is installed on all links. During the initial routing process, if the flow on a link exceeds this initial capacity, a higher system which can carry this flow value is installed on the link. After the routing process, the links with zero flow value have still system 1, while the others have been updated accordingly.

Unless they are fixed or deleted, all the links can be a candidate for further examination. After decreasing the flows on each link as explained in Section 6.3, the remaining capacities are calculated. Figure 6.8 illustrates an example (The system data given in Section 6.1 is used).









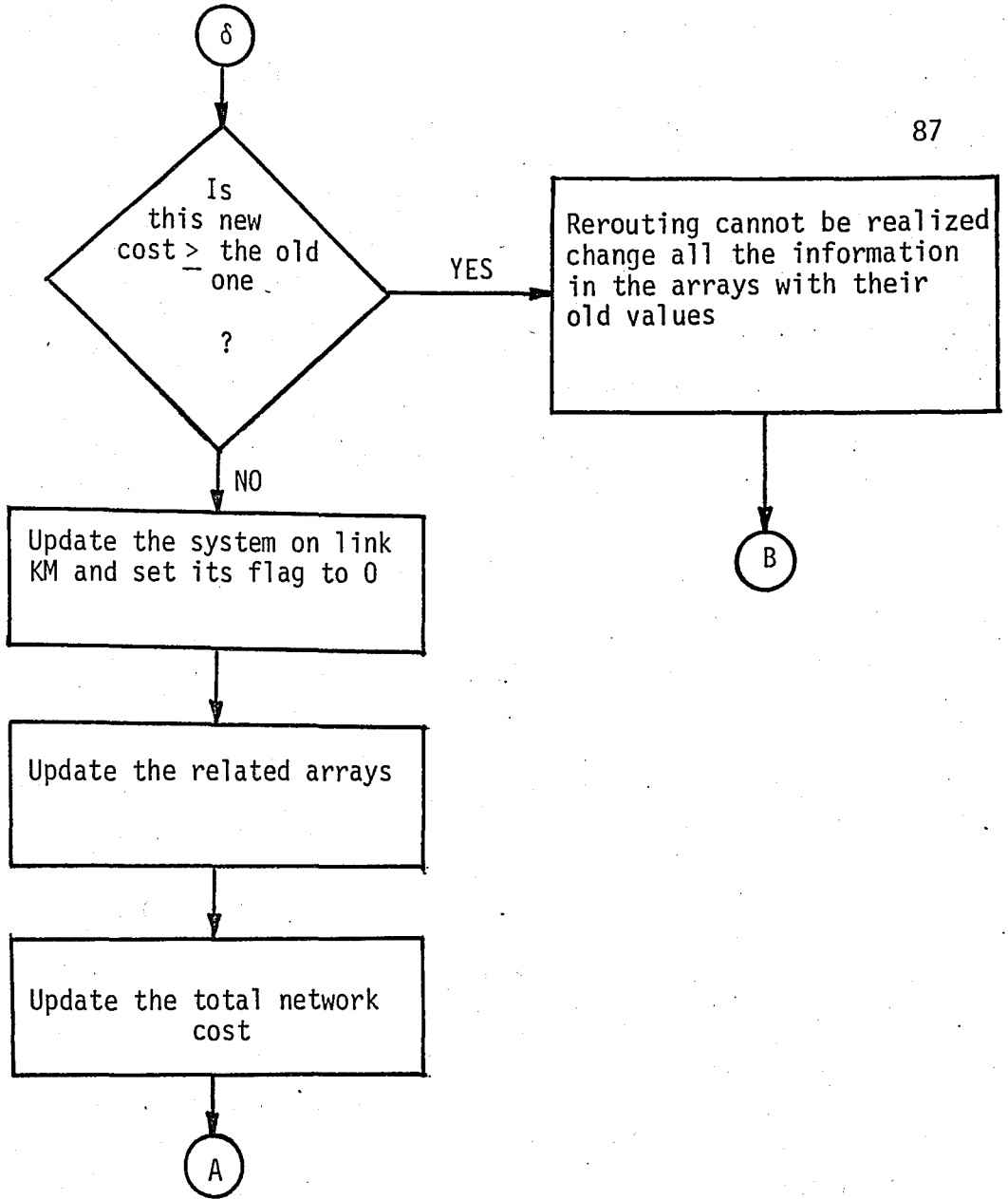
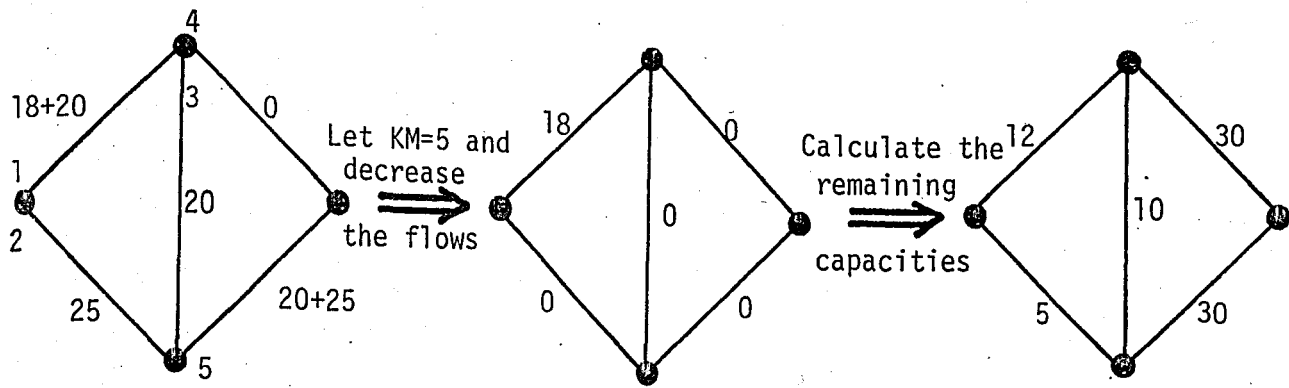


Figure 6.7 - Flowchart of the solution procedure.



Systems on Links:

Link 1 - System 2

Link 2 - System 1

Link 3 - System 1

Link 4 - System 1

Link 5 - System 2

Figure 6.8 - Calculating remaining capacities in the development problem.

Note that after decreasing the flows, link 1 has 18 units of flow left. Since this is a development problem and there is no existing (fixed) media on link 1, it can be lowered to system 1 with a remaining capacity of 12. Link 5, which is the candidate link, is also lowered to system 1 from system 2 with a remaining capacity of 30.

After the rerouting process, the flows and systems on links are updated, and if the new total network cost is less than the old one, these new systems will become the current systems on the links. The process continues in this manner until no further improvement can be made.

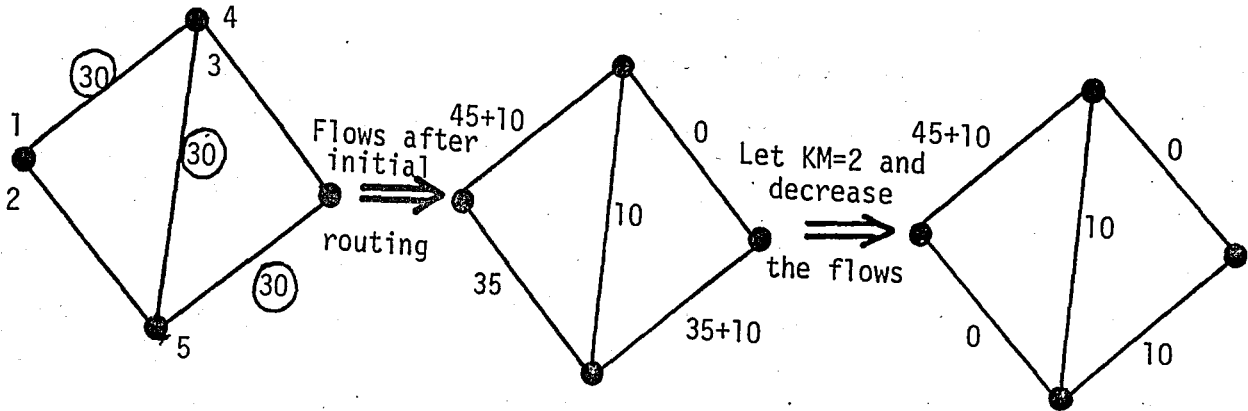
6.3.2 The Capacity Expansion Problem

In the capacity expansion problem there is an existing network, that is, some links have existing media which must be taken into account. During the solution procedure, the links with no existing media are treated in the same manner as in the development problem. Therefore, there is no need in repeating the treatment of such links.

In the initialization stage of the algorithm, the cost of the links with existing media are calculated according to the current systems on them. The fixed cost component of existing media are taken as zero. During the initial routing process, if the flow on such a link exceeds its existing capacity, another system will be installed near the current one such that, they together will be able to carry the total amount of flow on the link. For example, assume that the existing capacity of link i is 30, and a flow of 40 units is assigned to this link. Since there is already an existing media with 30 units, the problem is how to carry the extra ten units. For this reason, a second system with 30 units is installed on link i .

It is important not to delete links with existing capacity. Therefore, such links are fixed at the beginning of the algorithm to avoid deletion.

The remaining capacities of these links are also calculated taking into account the existing media. An example is given in Figure 6.9 (Again the system data in Section 6.1 is used).



Existing media:

Link 1 - System 1

Link 3 - System 1

Link 5 - System 1

Systems on Links:

Link 1 - 0 + 1

Link 2 - 2

Link 3 - 0

Link 4 - 1

Link 5 - 0 + 1

Calculate the remaining



capacities

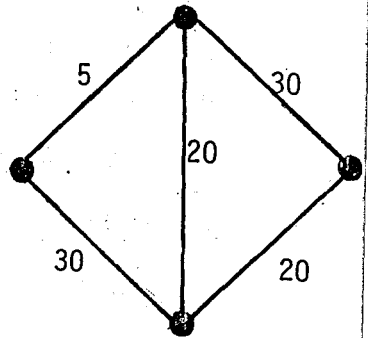


Figure 6.9 - Calculating remaining capacities in the capacity expansion problem.

Note that after decreasing the flows, link 5 is left with 10 units of flow. There were two systems of capacity 30 on this link; one being the existing medium, and the other being the installed medium. However, after the decreasing process, the second system with 30 units is not required any more. Therefore, the remaining capacity of link 5

is 20, and there is only the existing medium left on it.

After the rerouting process, the solution procedure continues as it is explained in the development problem.

6.3.3 The Minimum Cost Flow Algorithm Used in the Procedure

The rerouting of circuit demands is one of the major steps in the solution procedure. This step is performed by applying a minimum cost flow algorithm between the endpoints of the relations using the candidate link KM. The problem is to find a flow of a given value (the circuit demand of the relation) between the endpoints of a relation so that the total cost of the flow is minimized. This process is repeated for all the relations using KM.

The "out-of-kilter" algorithm of Ford and Fulkerson [24] is the best-known procedure for the minimum cost flow problem. However, a conceptually simpler algorithm was preferred in the solution procedure. For this reason, the minimum cost flow algorithm of Busacker and Gowen [48] was chosen.

The minimum cost flow algorithm of Busacker and Gowen is a dual algorithm. The reason is that, the algorithm involves the building up of the minimum cost flow pattern with the given value by starting with a minimum cost flow pattern having some value less than this given value, and adding extra flow in the network to obtain a minimum cost flow having a value greater than the starting value, etc. until the minimum cost flow pattern with the given value is reached. Since the zero flow is a minimum cost flow of value zero, a starting minimum cost flow pattern is always available.

A different approach is the primal approach which is used in the minimum cost flow algorithm of Klein[26,48]. The aim of this approach is first to obtain a feasible flow pattern and then improve this flow pattern until the minimum cost flow is reached. Klein [26] uses the maximum flow algorithm in finding an initial feasible flow pattern. His algorithm is based on negative circuit determination.

The algorithm of Busacker and Gowen requires the links of the network to be directed. However, the transmission network that is used in the procedure is undirected. Therefore, each link is separated to two arcs, one in each direction, thus increasing the number of edges in the network.

The next step of the algorithm is to construct an incremental graph. In this graph, associated with each arc there is a "dashed" arc in the opposite direction which carries the flow passing through its "solid" arc. The cost of this dashed arc is the negative of the cost of its solid arc. The construction of this incremental graph increases the number of links to $4x$ (the actual number of links in the network). Therefore, the network is prepared for this incremental graph before entering the minimum cost flow algorithm, and at the end, these arcs are again converted to the original links of the network.

The algorithm then finds the shortest path between the endpoints of a relation. However, since there are negative costs on the arcs, either a shortest path algorithm working with general cost matrices, for example, the algorithm due to Ford (described in [48]), must be utilized, or the negative costs must be removed. It was preferable to use a negative cost removal algorithm, because there is already a shortest

path algorithm used in the solution procedure. For this reason, the algorithm described in Bazaraa and Jarvis [49] was used in removing the negative costs, and the shortest path algorithm described in Section 5.4 was later on applied to this network.

The minimum cost flow algorithm then calculates the largest amount of flow that can be sent along this shortest path, and allocates this flow value on the arcs of the shortest path. This procedure continues until the given flow value is reached.

Since the optimal routing of circuit demands is treated as a multi-commodity network flow problem, after rerouting a relation, the flows on the "dashed" arcs which show the actual amount of flow used in rerouting that relation, are all set to zero so that those flows won't be reused during the rerouting of the next relation.

VII. NUMERICAL RESULTS

The solution procedure presented in Chapter VI is applied to a test network which is commonly used by 11 European countries in the context of COST (European Co-operation in Scientific and Technical Research) 201 Project: "Methods for Planning and Optimization of Telecommunication Networks". This test network is a part of the Irish telecommunications network.

The first section introduces the test network and tabulates the input data. Then, the results of the development and capacity expansion problems are listed. Finally, an analysis of the results is presented.

7.1 THE TEST NETWORK

The switching network of the test network is given in Figure 7.1. It is a small test network with 6 switching nodes and 15 trunk groups. The transmission network of this test network is illustrated in Figure 7.2. This network consists of 10 transmission nodes and 21 transmission media. The nodes numbered from 1 to 6 are the switching nodes, and the ones from 7 to 10 constitute the transmission nodes. The switching

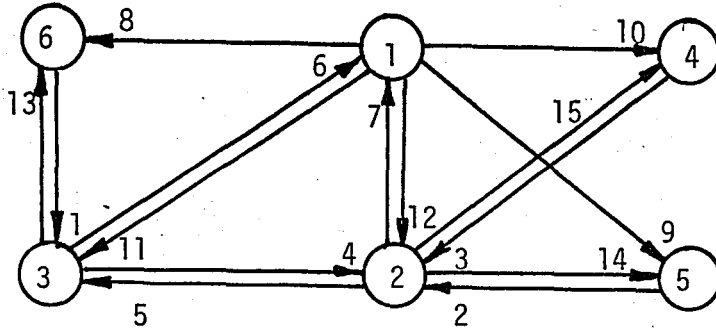


Figure 7.1 - Switching network of the test network.

nodes are the originating and terminating nodes of trunk groups, whereas the transmission nodes are the junctions or multiplexing nodes over which the trunk groups are routed.

The originating and terminating nodes of the links in Figure 7.2, together with their distances are presented in Table 7.1.

Table 7.1 - Originating and Terminating Nodes of the Links and Their Distances

Link No.	Originating Node	Terminating Node	Distance (km)
1	1	7	11
2	1	7	12
3	1	8	64
4	1	9	26
5	2	3	78
6	2	5	144
7	2	3	75
8	2	9	20
9	2	10	14
10	3	6	15
11	3	9	46
12	3	5	40
13	4	5	30
14	4	10	30
15	5	10	30
16	6	7	90
17	6	8	75
18	8	9	98

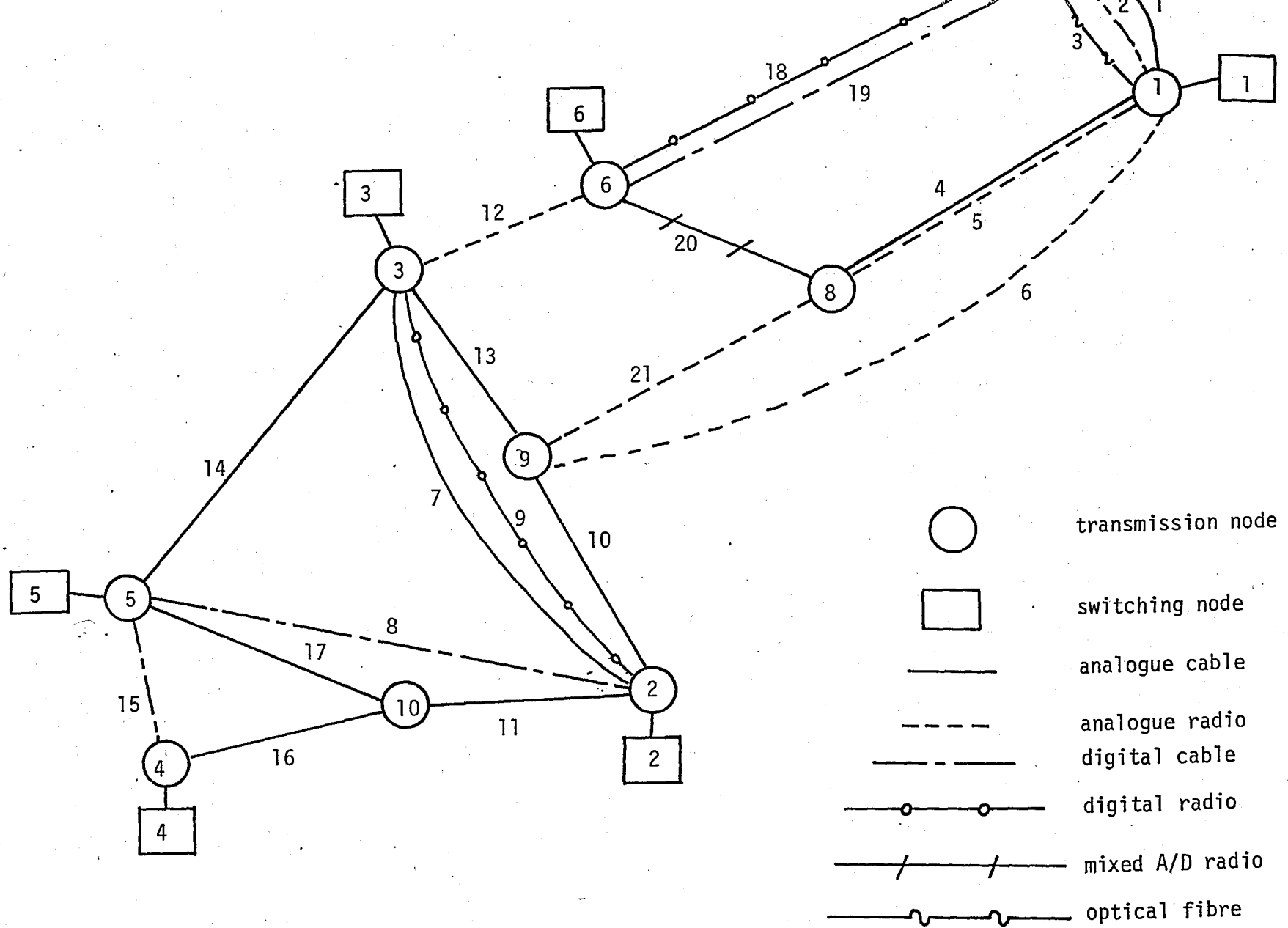


Figure 7.2 - Transmission network of the test network.

Note that while the original transmission network has 21 transmission media, only 18 of them were taken. This is due to the fact that, some links having common endpoints are also equal in length. For example, there are two links of length 11 between node pair (1,7). Therefore, only one of them was taken into consideration.

The circuit demands of 15 relations are given in Table 7.2.

Table 7.2 - Originating and Terminating Switching Nodes of Each Trunk Group and Their Number of Circuits (Set 1)

Relation No.	Originating SW Node	Terminating SW Node	Number of Circuits
1	6	3	12
2	5	2	14
3	4	2	17
4	3	2	5
5	2	3	5
6	3	1	30
7	2	1	59
8	1	6	9
9	1	5	10
10	1	4	11
11	1	3	17
12	1	2	26
13	3	6	3
14	2	5	2
15	2	4	3

This table tabulates the originating and terminating switching nodes of each trunk group (relation) and the number of circuits (demand) required between them. There is also another set of circuit demands consisting of 14 relations and their corresponding information which

was used in the same test network. This set is given in Table 7.3.

Table 7.3 - Originating and Terminating Switching Nodes of Each Trunk Group and Their Number of Circuits (Set 2)

Relation No.	Originating SW Node	Terminating SW Node	Number of Circuits
1	2	5	8
2	2	6	17
3	3	4	33
4	3	5	6
5	2	4	27
6	3	6	7
7	1	2	25
8	1	4	18
9	1	5	23
10	2	3	35
11	1	3	22
12	4	5	13
13	4	6	17
14	5	6	22

All input data are taken from COST Project 201 except for the system capacities and related cost figures. These figures are taken from the paper by Baybars and Kortanek [8], and were tabulated in Section 6.1. Three alternate transmission systems are considered in the model.

7.2 RESULTS OF THE TEST NETWORK

The algorithm presented in Section 6.3 was programmed in FORTRAN language, and was run on the IBM 4331 system at the Marmara Scientific

and Industrial Research Institute. The first part of this section presents the results of the development problem together with a sensitivity analysis. In the same manner, the results of the capacity expansion problem are given in the second part.

7.2.1 Results of the Development Problem

The data set of circuit demands, and the different cases analyzed are specified at the heading of the tables.

For the calculation of criterion V, three different formulas are used. The one given in Section 6.1 is denoted by V_1 , while

$$V_i = \frac{XFLW_i}{MD_i} \quad (7.1)$$

is denoted by V_2 , and

$$V_i = \left[\frac{REALCT - PRECT}{DELFL_i} \right] \times MD_i \quad (7.2)$$

where

$$DELFL_i = XFLW_i - CAP_{\ell S-1} \quad (7.3)$$

$$REALCT = FCOST_{\ell S} + [UVC_{\ell S} \times DELFL_i] \quad (7.4)$$

$$PRECT = FCOST_{\ell S-1} + [UVC_{\ell S-1} \times CAP_{\ell S-1}] \quad (7.5)$$

is denoted by V_3 .

If criterion V_2 is used, the candidate link will be the one having the minimum V_2 value. Criterion V_3 , on the other hand, is treated like V_1 , that is, the link having the maximum V_3 value will be the candidate link.

Table 7.4 - Demand Data Set 1; Demands in Descending Order;
Criterion V_1

Link No.	System Chosen (initial values)				System Chosen (final values)				
	Type	Capacity	Used Capacity	Real Total Cost (\$)	Type	Capacity	Used Capacity	Real Total Cost (\$)	
1	-	-	-	-	-	-	-	-	
2	-	-	-	-	-	-	-	-	
3	-	-	-	-	-	-	-	-	
4	3	270	162	37,566,720	3	270	162	37,566,720	
5	-	-	-	-	-	-	-	-	
6	-	-	-	-	-	-	-	-	
7	-	-	-	-	-	-	-	-	
8	3	270	116	28,642,640	3	270	116	28,642,640	
9	2	90	57	13,033,860	2	90	57	13,033,860	
10	1	30	24	9,066,000	1	30	24	9,066,000	
11	2	90	66	43,268,512	2	90	66	43,268,512	
12	-	-	-	-	-	-	-	-	
13	-	-	-	-	-	-	-	-	
14	2	90	31	27,095,088	2	90	31	27,095,088	
15	1	30	26	18,318,000	1	30	26	18,318,000	
16	-	-	-	-	-	-	-	-	
17	-	-	-	-	-	-	-	-	
18	-	-	-	-	-	-	-	-	
Total Network Cost				176,990,816	Total Network Cost				176,990,816

No improvement could be obtained (7 iterations).

Table 7.5 - Demand Data Set 1; Demands in Ascending Order;
Criterion V_1

Link No.	System Chosen (initial values)				System Chosen (final values)				
	Type	Capacity	Used Capacity	Real Total Cost (\$)	Type	Capacity	Used Capacity	Real Total Cost (\$)	
1	-	-	-	-	-	-	-	-	
2	-	-	-	-	-	-	-	-	
3	-	-	-	-	-	-	-	-	
4	3	270	162	37,566,720	3	270	162	37,566,720	
5	-	-	-	-	-	-	-	-	
6	-	-	-	-	-	-	-	-	
7	-	-	-	-	-	-	-	-	
8	3	270	116	28,642,640	3	270	116	28,642,640	
9	2	90	57	13,033,860	2	90	57	13,033,860	
10	1	30	24	9,066,000	1	30	24	9,066,000	
11	2	90	66	43,268,512	2	90	66	43,268,512	
12	-	-	-	-	-	-	-	-	
13	-	-	-	-	-	-	-	-	
14	2	90	31	27,095,088	2	90	31	27,095,088	
15	1	30	26	18,318,000	1	30	26	18,318,000	
16	-	-	-	-	-	-	-	-	
17	-	-	-	-	-	-	-	-	
18	-	-	-	-	-	-	-	-	
Total Network Cost				176,990,816	Total Network Cost				176,990,816

No improvement could be obtained (7 iterations).

Table 7.6 - Demand Data Set 2; Demands in Descending Order;
Criterion V_1

Link No.	System Chosen (initial values)				System Chosen (final values)				
	Type	Capacity	Used Capacity	Real Total Cost (\$)	Type	Capacity	Used Capacity	Real Total Cost (\$)	
1	-	-	-	-	-	-	-	-	
2	-	-	-	-	-	-	-	-	
3	-	-	-	-	-	-	-	-	
4	2	90	88	25,068,160	2	90	88	25,068,160	
5	-	-	-	-	-	-	-	-	
6	-	-	-	-	-	-	-	-	
7	-	-	-	-	-	-	-	-	
8	3	270	174	28,963,952	3	270	196	29,085,840	
9	3	270	132	20,111,888	3	270	154	20,197,200	
10	2	90	63	14,061,150	2	90	63	14,061,150	
11	3	270	130	66,056,448	3	270	152	66,336,784	
12	1	30	22	23,928,000	-	-	-	-	
13	1	30	13	17,108,992	-	-	-	-	
14	3	270	95	42,789,440	3	270	108	42,897,472	
15	2	90	37	27,287,696	2	90	72	28,411,200	
16	-	-	-	-	-	-	-	-	
17	-	-	-	-	-	-	-	-	
18	-	-	-	-	-	-	-	-	
Total Network Cost				265,375,712	Total Network Cost				226,057,792

Links 12 and 13 have been deleted (9 iterations).

Table 7.7 - Demand Data Set 2; Demands in Descending Order;
Criterion V_2

Link No.	System Chosen (initial values)				System Chosen (final values)				
	Type	Capacity	Used Capacity	Real Total Cost (\$)	Type	Capacity	Used Capacity	Real Total Cost (\$)	
1	-	-	-	-	-	-	-	-	
2	-	-	-	-	-	-	-	-	
3	-	-	-	-	-	-	-	-	
4	2	90	88	25,068,160	2	90	88	25,068,160	
5	-	-	-	-	-	-	-	-	
6	-	-	-	-	-	-	-	-	
7	-	-	-	-	-	-	-	-	
8	3	270	174	28,963,952	3	270	196	29,085,840	
9	3	270	132	20,111,888	3	270	154	20,197,200	
10	2	90	63	14,061,150	2	90	63	14,061,150	
11	3	270	130	66,056,448	3	270	152	66,336,784	
12	1	30	22	23,928,000	-	-	-	-	
13	1	30	13	17,108,992	-	-	-	-	
14	3	270	95	42,789,440	3	270	108	42,897,472	
15	2	90	37	27,287,969	2	90	72	28,411,200	
16	-	-	-	-	-	-	-	-	
17	-	-	-	-	-	-	-	-	
18	-	-	-	-	-	-	-	-	
Total Network Cost				265,375,712	Total Network Cost				226,057,792

Links 12 and 13 have been deleted (9 iterations).

Table 7.8 - Demand Data Set 2; Demands in Descending Order;
Criterion V_3

Link No.	System Chosen (initial values)				System Chosen (final values)				
	Type	Capacity	Used Capacity	Real Total Cost (\$)	Type	Capacity	Used Capacity	Real Total Cost (\$)	
1	-	-	-	-	-	-	-	-	
2	-	-	-	-	-	-	-	-	
3	-	-	-	-	-	-	-	-	
4	2	90	88	25,068,160	2	90	88	25,068,160	
5	-	-	-	-	-	-	-	-	
6	-	-	-	-	-	-	-	-	
7	-	-	-	-	-	-	-	-	
8	3	270	174	28,963,952	3	270	196	29,085,840	
9	3	270	132	20,111,888	3	270	154	20,197,200	
10	2	90	63	14,061,150	2	90	63	14,061,150	
11	3	270	130	66,056,448	3	270	152	66,336,784	
12	1	30	22	23,928,000	-	-	-	-	
13	1	30	13	17,108,992	-	-	-	-	
14	3	270	95	42,789,440	3	270	108	42,897,472	
15	2	90	37	27,287,696	2	90	72	28,411,200	
16	-	-	-	-	-	-	-	-	
17	-	-	-	-	-	-	-	-	
18	-	-	-	-	-	-	-	-	
Total Network Cost				265,375,712	Total Network Cost				226,057,792

Links 12 and 13 have been deleted (9 iterations).

Table 7.9 - Demand Data Set 2; Demands in Ascending Order;
Criterion V_1

Link No.	System Chosen (initial flows)				System Chosen (final flows)				
	Type	Capacity	Used Capacity	Real Total Cost (\$)	Type	Capacity	Used Capacity	Real Total Cost (\$)	
1	-	-	-	-	-	-	-	-	
2	-	-	-	-	-	-	-	-	
3	-	-	-	-	-	-	-	-	
4	2	90	88	25,068,160	-	-	-	-	
5	-	-	-	-	-	-	-	-	
6	-	-	-	-	-	-	-	-	
7	-	-	-	-	-	-	-	-	
8	3	270	128	28,709,120	3	270	136	28,753,440	
9	2	90	86	13,468,280	2	90	78	13,348,440	
10	2	90	63	14,061,150	2	90	63	14,061,150	
11	3	270	130	66,056,448	3	270	138	66,158,384	
12	2	90	68	37,710,400	2	90	76	38,052,800	
13	1	30	30	18,690,000	1	30	30	18,690,000	
14	2	90	78	28,603,792	2	90	78	28,603,792	
15	1	30	8	16,644,000	-	-	-	-	
16	-	-	-	-	-	-	-	-	
17	-	-	-	-	-	-	-	-	
18	-	-	-	-	-	-	-	-	
Total Network Cost				249,011,328	Total Network Cost				232,736,144

Link 15 has been deleted (9 iterations).

Table 7.10 - Demand Data Set 2; Demands Unsorted; Criterion V_1

Link No.	System Chosen (initial flows)				System Chosen (final flows)				
	Type	Capacity	Used Capacity	Real Total Cost (\$)	Type	Capacity	Used Capacity	Real Total Cost (\$)	
1	-	-	-	-	-	-	-	-	
2	-	-	-	-	-	-	-	-	
3	-	-	-	-	-	-	-	-	
4	2	90	88	25,068,160	2	90	88	25,068,160	
5	-	-	-	-	-	-	-	-	
6	-	-	-	-	-	-	-	-	
7	-	-	-	-	-	-	-	-	
8	3	270	105	28,581,696	2	90	88	19,283,200	
9	3	270	133	20,115,760	3	270	150	20,181,696	
10	2	90	63	14,061,150	2	90	63	14,061,150	
11	1	30	17	26,804,192	-	-	-	-	
12	3	270	135	57,495,792	3	270	152	57,684,160	
13	3	270	143	43,188,320	3	270	168	43,396,080	
14	3	270	125	43,038,736	3	270	150	43,246,496	
15	1	30	8	16,644,000	-	-	-	-	
16	-	-	-	-	-	-	-	-	
17	-	-	-	-	-	-	-	-	
18	-	-	-	-	-	-	-	-	
Total Network Cost				274,997,760	Total Network Cost				222,920,928

Links 11 and 15 have been deleted (9 iterations).

Table 7.11 - Demand Data Set 2; Demands Unsorted; Criterion V_3

Link No.	System Chosen (initial flows)				System Chosen (final flows)				
	Type	Capacity	Used Capacity	Real Total Cost (\$)	Type	Capacity	Used Capacity	Real Total Cost (\$)	
1	-	-	-	-	-	-	-	-	
2	-	-	-	-	-	-	-	-	
3	-	-	-	-	-	-	-	-	
4	2	90	88	25,068,160	2	90	88	25,068,160	
5	-	-	-	-	-	-	-	-	
6	-	-	-	-	-	-	-	-	
7	-	-	-	-	-	-	-	-	
8	3	270	105	28,581,696	2	90	88	19,283,200	
9	3	270	133	20,115,760	3	270	150	20,181,696	
10	2	90	63	14,061,150	2	90	63	14,061,150	
11	1	30	17	26,804,192	-	-	-	-	
12	3	270	135	57,495,792	3	270	152	57,684,160	
13	3	270	143	43,188,320	3	270	168	43,396,080	
14	3	270	125	43,038,736	3	270	150	43,246,496	
15	1	30	8	16,644,000	-	-	-	-	
16	-	-	-	-	-	-	-	-	
17	-	-	-	-	-	-	-	-	
18	-	-	-	-	-	-	-	-	
Total Network Cost				274,997,760	Total Network Cost				222,920,928

Links 11 and 15 have been deleted (9 iterations).

7.2.2 Results of the Capacity Expansion Problem

The results of the capacity expansion problem are tabulated in the same manner as in Section 7.2.1. However, this time the existing capacities are taken into account during the calculations.

Table 7.12 - Demand Data Set 2; Demands in Descending Order;
Criterion V_2

Link No.	System Chosen (initial values)					System Chosen (final values)					
	Types	Ex. Cap.	Total Cap.	Used Cap.	Real Total Cost (\$)	Types	Ex. Cap.	Total Cap.	Used Cap.	Real Total Cost (\$)	
1	1	30	30	-	-	1	30	30	-	-	
2	-	-	-	-	-	-	-	-	-	-	
3	-	-	-	-	-	-	-	-	-	-	
4	2	90	90	88	2,448,160	2	90	90	88	2,448,160	
5	1,1	30	60	35	49,802,992	1,1	30	60	35	49,802,992	
6	1	30	30	-	-	1	30	30	25	11,160,000	
7	-	-	-	-	-	-	-	-	-	-	
8	2	-	90	50	18,470,000	2	-	90	52	18,512,800	
9	1	-	30	27	8,591,800	-	-	-	-	-	
10	1,1,1	30	90	63	18,829,488	1,2	30	120	63	14,974,650	
11	2	90	90	88	4,331,360	2	90	90	90	4,429,800	
12	3	-	270	127	57,407,152	3	-	270	129	57,429,312	
13	1,1,1	30	90	81	39,332,992	1,2	30	120	108	31,393,792	
14	1	-	30	27	18,410,992	-	-	-	-	-	
15	-	-	-	-	-	-	-	-	-	-	
16	-	-	-	-	-	-	-	-	-	-	
17	-	-	-	-	-	-	-	-	-	-	
18	-	-	-	-	-	-	-	-	-	-	
Total Network Cost					217,624,928	Total Network Cost					190,151,488

link 14 has been deleted (3 iterations)

Table 7.13 - Demand Data Set 2; Demands in Ascending Order;
Criterion V_2

Link No.	System Chosen (initial values)					System Chosen (final values)					
	Types	Ex. Cap.	Total Cap.	Used Cap.	Real Total Cost (\$)	Types	Ex. Cap.	Total Cap.	Used Cap.	Real Total Cost (\$)	
1	1	30	30	-	-	1	30	30	-	-	
2	-	-	-	-	-	-	-	-	-	-	
3	-	-	-	-	-	-	-	-	-	-	
4	2	90	90	88	2,448,160	2	90	90	88	2,448,160	
5	1,1	30	60	40	51,012,000	1,1	30	60	40	51,012,000	
6	1,1	30	60	31	90,158,400	1,1	30	60	31	90,158,400	
7	-	-	-	-	-	-	-	-	-	-	
8	2	-	90	60	18,684,000	2	-	90	60	18,684,000	
9	1	-	30	27	8,591,800	1	-	30	27	8,591,800	
10	1,1,1	30	90	63	18,829,488	1,2	30	120	63	14,974,650	
11	2,1	90	120	98	29,950,592	2,1	90	120	98	29,950,592	
12	3	-	270	96	57,063,680	3	-	270	96	57,063,680	
13	1,1,1	30	90	81	39,332,992	1,2	30	120	108	31,393,792	
14	1	-	30	27	18,410,992	-	-	-	-	-	
15	-	-	-	-	-	1	-	30	27	18,410,992	
16	-	-	-	-	-	-	-	-	-	-	
17	-	-	-	-	-	-	-	-	-	-	
18	-	-	-	-	-	-	-	-	-	-	
Total Network Cost					334,481,664	Total Network Cost					322,687,744

Link 14 has been deleted (5 iterations).

Table 7.14 - Demand Data Set 2; Demands Unsorted, Criterion V_2

Link No.	System Chosen (initial values)					System Chosen (final values)					
	Types	Ex. Cap.	Total Cap.	Used Cap.	Real Total Cost (\$)	Types	Ex. Cap.	Total Cap.	Used Cap.	Real Total Cost (\$)	
1	1	30	30	-	-	1	30	30	-	-	
2	-	-	-	-	-	-	-	-	-	-	
3	-	-	-	-	-	-	-	-	-	-	
4	2	90	90	88	2,448,160	2	90	90	88	2,448,160	
5	1,1	30	60	50	53,430,000	1,1	30	60	50	53,430,000	
6	1,1	30	60	41	94,622,400	1,1	30	60	41	94,622,400	
7	-	-	-	-	-	-	-	-	-	-	
8	3	-	270	140	28,775,600	3	-	270	146	28,808,832	
9	3	-	270	107	20,014,944	3	-	270	113	20,038,208	
10	1,1,1	30	90	63	18,829,488	1,2	30	120	63	14,974,650	
11	2,1	90	120	96	29,665,392	2,1	90	120	102	30,520,992	
12	1	-	30	6	21,944,000	-	-	-	-	-	
13	1,1	30	60	46	20,178,000	1,1	30	60	46	20,178,000	
14	2	-	90	62	28,090,192	2	-	90	62	28,090,192	
15	2	-	90	45	27,544,496	2	-	90	51	27,737,088	
16	-	-	-	-	-	-	-	-	-	-	
17	-	-	-	-	-	-	-	-	-	-	
18	-	-	-	-	-	-	-	-	-	-	
Total Network Cost					345,542,144	Total Network Cost					320,848,384

Link 12 has been deleted (5 iterations).

7.3 ANALYSIS OF THE RESULTS

Looking at the tables given in the previous section, it is seen that the number of iterations in the capacity expansion problems are less than those in the development problems. This is due to the fact that the V_i values of the existing media are set to very large (if the minimum is chosen) or very small (if the maximum is chosen) numbers such that they won't be considered as candidate links for deletion. In other words, such links are fixed at the beginning of the algorithm. Since the number of iterations is directly related to the number of the candidate links, there are less iterations in capacity expansion problems. Of course, this highly depends on the input data of existing media.

In all of the different cases given in Section 7.2, the links having the smallest flow values are deleted (since they are already having system 1, before entering the minimum cost flow algorithm for rerouting, their remaining capacities become zero). The V values of such links indicate that they are under-utilized or expensive, and must be deleted from the network.

A set of interesting results are given in Tables 7.6, 7.9 and 7.10. A better result was obtained in the case where the demands are unsorted. Although the initial total network cost in Table 7.10 is higher than in the others, a smaller final total network cost was obtained in the same number of iterations. Table 7.9 starts with the smallest initial total cost, but doesn't improve much. Therefore, the order of the circuit demands exert a remarkable influence on the results.

Finally it should be noted that, the different V criteria all yielded the same results; choosing the same candidate links for deletion, although the order of the choices may be different at the intermediate steps. Other V criteria can also be used in the solution procedure.

VIII. CONCLUSION AND EXTENSIONS FOR FUTURE WORK

In this study, the network structure and circuit routing in transmission networks are optimized simultaneously. The developed algorithm specifies the type of transmission systems and the links themselves on which the systems are to be installed, as well as the number of circuits to be installed on each such link. While meeting the point-pair circuit demands, a minimum cost routing is obtained.

As a result of the existence of economies of scale, and the consideration of alternate transmission media, the link cost functions become piecewise concave.

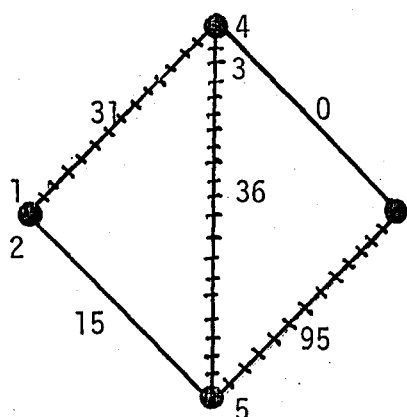
In this problem, the variable costs of the media are very small compared to the fixed costs [9]. Therefore in all of the cost comparisons, the fixed costs are also taken into account.

Two different but interrelated problems are presented in the solution procedure. One is the development problem, in which there are no existing transmission media in the network. The other is the capacity expansion problem, which takes the existing network into account. For this problem, the fixed cost of the existing media are taken as zero during the solution procedure.

The developed model also takes different technologies into account. That is, a transmission media can have analogue, digital or mixed technology. However, no analogue/digital conversion is considered. This can be achieved by using transfer links which is explained in Section 3.2.2.

In order to increase the flexibility of the network against transmission media failures, the circuit demands can be multirouted. That is, diverse transmission paths can be assigned to each relation. However, in the solution procedure presented in Chapter VI, multirouting is not taken into account.

One way to consider multirouting is as follows:



Systems on Links:

Link 1 - System 2

Link 2 - System 1

Link 3 - System 2

Link 4 - System 1

Link 5 - System 3

Figure 8.1 - An example of multirouting.

Suppose $KM = 1$. Then, the remaining capacities (given by the numbers on the links) are calculated before entering the Busacker and Gowen algorithm for rerouting.

The first step is to consider the relations using link 1, and their respective flow values which constitute the flow on link 1:

$$31 = 16(\text{Relation 1}) + 7(\text{Relation 2}) + 8(\text{Relation 3}).$$

Then, the relation having the smallest flow value is chosen. In

this case, relation 2 with value seven is the smallest. The shortest path of relation 2 (as shown by ||||| in Figure 8.1) is taken, and the flows on the links of this shortest path are analyzed:

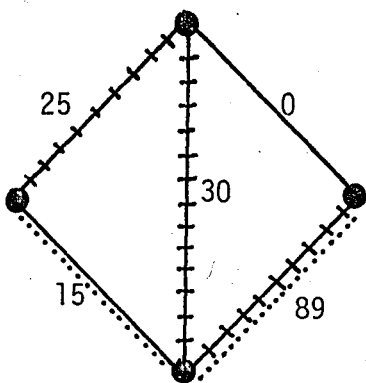
Link 3 has 36 units. Hence, in order to have a smaller system (system 1) on this link, $36 - 30 = 6$ units must be taken out.

Similarly, link 5 has 95 units. In order to have system 2 on this link, $95 - 90 = 5$ units must be taken out.

The next step is to choose the maximum excess flow which is smaller than the value of relation 2:

$$\max\{5,6\} = 6 < 7.$$

Hence, the flow of relation 2, which is seven, can be separated as six plus one. The maximum value is then subtracted from all the links on the shortest path of relation 2. The updated flows are illustrated in Figure 8.2.



Updated Systems on Links:

Link 1 - System 1

Link 2 - System 1

Link 3 - System 1

Link 4 - System 1

Link 5 - System 2

Figure 8.2 - Updated flow values.

As a result, the systems of the links on the shortest path have decreased. At this stage, the second shortest path of relation 2 (not necessarily link-disjoint) can be calculated (suppose path in Figure 8.2).

This is necessary in rerouting the maximum flow value which was decreased from all the links on the first shortest path. Then, the cost of this new path can be compared with the cost of the old path to see whether it is economic to use this path. Multirouting can be realized if this result is positive.

A quite different and more realistic approach to this optimization procedure may be to consider it as a dynamic design problem.

Finally, it will be interesting to compare the results of the simultaneous and independent optimizations of the NSO and CRO modules.

APPENDICES

APPENDIX A

DESCRIPTION OF THE INPUT DATA

The parameters and arrays which denote the basic inputs to the computer program are as follows:

CAP(L)	: The circuit capacity of transmission system L.
CCL(KR)	: The demand (in number of circuits) of relation (node pair) KR.
CON(KR)	: The beginning node of relation KR.
CTN(KR)	: The terminating node of relation KR.
EXCAP(K)	: The existing capacity (in number of circuits) of edge K.
FCOST(L)	: The fixed cost of installing transmission system L.
LCN	: The number of relations (i.e., the number of node-pair circuit demands).
LLL	: The maximum possible number of edges in the shortest path of a relation.
MD(K)	: The length (in kilometers) of edge K.
MG	: The number of edges in the network.
MON(K)	: The originating node of edge K.
MTN(K)	: The terminating node of edge K.
NN	: The number of transmission nodes in the network.
UVC(L)	: The unit variable cost associated with transmission system L.

APPENDIX B

EXPLANATION OF THE COMPUTER PROGRAM

The variables which denote the basic outputs of the computer program are as follows:

- ISYS (K,.) : The matrix which contains the transmission systems on edge K.
- RE COST(K) : The final cost of edge K.
- XFLW(K) : The total amount of flow (in number of circuits) on edge K.

It may be helpful to include the definitions for some additional variables used within the computer program.

- COST(K) : The approximated cost value of edge K (this value is used during the calculation of the shortest paths).
- COSTAR(K) : The real total cost of edge K after rerouting.
- COSTBR(K) : The initial real total cost of edge K.
- CURCAP(K) : The current capacity (in number of circuits) of edge K.
- FLNUM(K,.) : The matrix which stores the circuit demand of the relations using edge K.
- ICONCH : The variable which indicates whether the network became disconnected or not.
- ICONCH = 1 , if the network became disconnected.
- ICONCH = 0 , otherwise.

- IDASH(K) : The dashed edge of edge K.
- $$\left. \begin{array}{l} \text{IDASH}(K) = K+1 \\ \text{IDASH}(K+1) = K \\ \text{IDASH}(K+2) = K+3 \\ \text{IDASH}(K+3) = K+2 \end{array} \right\} \text{ for } K = 1, 5, 9, \dots, 4MG$$
- IDELI(KR,..) : The matrix which contains the edge numbers in the shortest path of relation KR.
- INDEX(K) : The index which shows the number of transmission systems on edge K.
- INODP(I) : The pointer which shows the beginning position of node I in the array NOEDGE(.).
- INUM(K) : The index which shows the number of relations using edge K.
- IPER(I) : The flag of node I in the shortest path algorithm.
- $$\begin{array}{l} \text{IPER}(I) = 0 \text{ , if the node is not yet processed.} \\ \text{IPER}(I) = 1 \text{ , if the node is temporarily labeled.} \\ \text{IPER}(I) = 2 \text{ , if the node is permanently labeled.} \end{array}$$
- ITEMP(.) : The array which contains the temporarily labeled nodes in the shortest path algorithm.
- ITWIN(K) : The twin edge of edge K.
- $$\left. \begin{array}{l} \text{ITWIN}(K) = K+2 \\ \text{ITWIN}(K+2) = K \end{array} \right\} \text{ for } K = 1, 5, 9, \dots, 4MG$$
- KFLAG(K) : The flag of edge K.
- $$\begin{array}{l} \text{KFLAG}(K) = -1, \text{ if the edge is not yet processed.} \\ \text{KFLAG}(K) = 0, \text{ if the edge is deleted.} \\ \text{KFLAG}(K) = 1, \text{ if the edge is fixed.} \end{array}$$
- KM : The candidate edge for deletion.
- KNUM(K,..) : The matrix which stores the relations using edge K.
- LABL(I) : The current node potential of node I in the shortest path algorithm.
- LCIN(K) : The current transmission system on edge K (for the edges having no existing media on them).
- LSAVE(I) : The edge number which is used in labeling node I in the shortest path algorithm.

- NEDGE(I) : The total number of edges which go out of node I.
- NNN(KR) : The actual number of edges in the shortest path of relation KR.
- NOEDGE(.) : The array which stores the edge numbers that go out of each node in the network.
- REMCAP(K) : The remaining capacity (in number of circuits) left on edge K before entering the minimum-cost flow algorithm.
- RESULT : The variable which indicates the existence of a negative circuit in the network.
- RESULT = 1 , if there is a negative circuit in the network.
- RESULT = 0 , otherwise.
- SUMFLW(K) : The amount of flow (in number of circuits) on edge K obtained as a result of the rerouting.
- V(K) : The deletion criterion calculated for edge K.

In the above list of the additional variables, the arrays IDASH(K) and ITWIN(K) need further explanation. As it is mentioned in Chapter VI, the number of edges in the network are increased before entering the minimum-cost flow algorithm of Busacker and Gowen, and an incremental graph is constructed. In this graph, the edges numbered as 1,3,5,7,... are referred to as "solid" edges, while the ones numbered as 2,4,6,8,... are referred to as "dashed" edges. The solid edges are the actual edges of the network which are assigned a direction now. The dashed edges are necessary in order to keep track of the actual amount of flow passing through a solid edge. A solid edge has a "twin" solid edge in the opposite direction and a dashed edge, whereas a dashed edge has only a solid edge.

The objectives of the subroutines with their input requirements and issuing outputs are given in the following paragraphs.

SUBROUTINE FSTAR

This subroutine determines the forward star representation of the network. Its outputs are the arrays $INODP(I)$, $I = 1, 2, \dots, NN$, and $NOEDGE(.)$. The inputs to this subroutine are the arrays $MON(K)$ and $MTN(K)$ which contain the originating and terminating nodes of edge K , for $K = 1, 2, \dots, MG$.

SUBROUTINE FSTAR2

The objective of this subroutine is exactly the same as FSTAR, but the network used in this subroutine is the one that will be employed in the minimum-cost flow algorithm. That is, the network has $4MG$ edges. The inputs and outputs are also the same, but of course have different names.

SUBROUTINE MINCTF

This subroutine calculates the minimum-cost flow pattern from $CON(KR)$ to $CTN(KR)$ with the specified flow value $CCL(KR)$ for all the relations using the candidate edge KM . In other words, it performs the rerouting process. The outputs of this subroutine are the updated arrays of $IDEL1(KR,.)$, $FLNUM(K,.)$ and $KNUM(K,.)$. The major inputs are the $COST(K)$ and $REMCAP(K)$ values, for $K = 1, 2, \dots, 4MG$.

SUBROUTINE M2

This subroutine is in fact the main program of MINCTF. Its main objective is to prepare the network for the minimum-cost flow algorithm by assigning the related capacity and cost values to each arc (note that there are $4MG$ arcs in the network). Its output is the $SUMFLW(K)$, $K = 1, 2, \dots, MG$, array. The inputs are the original capacity and cost figures of the network.

SUBROUTINE REMOVE

The objective of this subroutine is to convert the network with negative costs to an equivalent network having non-negative costs. The output consists of this non-negative cost values. The inputs to this subroutine are the $INODP(I)$, for $I = 1, 2, \dots, NN$, and the $NOEDGE(.)$ arrays, and the cost figures on each arc.

SUBROUTINE SHPTH

This subroutine determines the shortest path of relation KR by an edge number processing algorithm. Its output is the $IDEL1(KR,.)$ matrix. The main inputs to this subroutine are the $CON(KR)$, $CTN(KR)$, $INODP(I)$, $I = 1, 2, \dots, NN$, $NOEDGE(.)$ and the $COST(K)$, $K = 1, 2, \dots, MG$, arrays.

SUBROUTINE SHPTH2

The objective of this subroutine is exactly the same as SHPTH,

but the network that is used here is the one having 4MG edges. The inputs and outputs are also the same, but the arrays have different names.

SUBROUTINE SORT

This subroutine sorts the circuit demand of the relations either in descending or in ascending order. The output consists of the sorted arrays, while the inputs are the CON(KR), CTN(KR) and CCL(KR) arrays, for $KR = 1, 2, \dots, LCN$.

SUBROUTINE STAR2

This subroutine prepares the network for the minimum-cost flow algorithm by increasing the number of edges and forming the IDASH(K) and ITWIN(K), $K = 1, 5, 9, \dots, 4MG$, arrays. It is in fact the main program of FSTAR2 and its output consists of the related arrays which show the forward star representation of the network having 4MG edges. The main inputs are the MON(K) and MTN(K), $K = 1, 2, \dots, MG$, arrays.

SUBROUTINE VCAL

This subroutine calculates the criterion V for all the edges in the network. The outputs are the V(K), $K = 1, 2, \dots, MG$, array and the candidate edge KM. The input consists of the arrays FCOST(.), UVC(.), XFLW(K), MD(K), $K = 1, 2, \dots, MG$.

APPENDIX C

LISTING OF THE COMPUTER PROGRAM

```

*****
MAIN PROGRAM
*****
COMMON /AAA/NN, MG, LLL, MON(300), MTN(300), REMCAP(300), ITWIN(300),
& IDASH(300), ICONCH
COMMON /XXX/CON(150), CTN(150), CCL(150), LCN
COMMON /XYZ/INODP(100), NOEDGE(300), IDEL1(150, 20)
COMMON /ZZZ/INODP2(100), NOEDG2(300), IDEL2(150, 20), RESULT
COMMON /EEE/NDUM(150), IDELDM(150, 20)
COMMON /YYY/FCOST(3), UVC(3), CAP(3), LCIN(300), CURCAP(300), XFLW(300)
COMMON /IJK/STOMD(300), SUMFLW(300)
COMMON /VVV/V(300), KM, INUM(300), KNUM(100, 20), FLNUM(100, 20),
& KFLAG(300)
COMMON /TTT/NOE(150), I2NUM(300), K2NUM(100, 20), FL2NUM(100, 20)
DIMENSION CVT(3), MD(300), COST(300), COSTBR(300), COSTAR(300),
& NNN(150), STJRE(300), STOFLW(300), IDUM(300), KDUM(100, 20),
& FLDUM(100, 20), STOCAP(300), STOCIN(300), SBR(300)
INTEGER CON, CTN
REAL MD
MR=5
MW=6
READ(MR, 1) LLL
FORMAT(I2)
READ(MR, 2) MG, NN
FORMAT(2(2X, I4))
WRITE(MW, 27) MG, NN
FORMAT(///, 15X, 'MG=', I4, /, 15X, 'NN=', I4)
READ(MR, 3) LCN
FORMAT(2X, I4)
NK=MG
NKR=LCN
WRITE(MW, 4)
FORMAT(///, 15X, 'MON', 5X, 'MTN', 5X, 'MD', /)
DO 5 K=1, NK
READ(MR, 6) MON(K), MTN(K), MD(K)
FORMAT(2I4, 2X, F5.0)
WRITE(MW, 7) MON(K), MTN(K), MD(K)
FORMAT(14X, I4, 4X, I4, 2X, F5.0)
CONTINUE
WRITE(MW, 8)
FORMAT(///, 15X, 'CON', 5X, 'CTN', 5X, 'CCL', /)
DO 9 IC=1, NKR
READ(MR, 10) CON(IC), CTN(IC), CCL(IC)
FORMAT(2(2X, I4), 1X, F6.0)
CONTINUE
=====
SORT THE CIRCUIT DEMANDS IN DESCENDING ORDER
=====
CALL SORT
DO 59 IC=1, NKR
WRITE(MW, 11) CON(IC), CTN(IC), CCL(IC)
FORMAT(14X, I4, 4X, I4, 4X, F4.0)
CONTINUE
WRITE(MW, 86)
FORMAT(///, 15X, 'SYSTEM', 5X, 'FCOST', 5X, 'UVC', 12X, 'CAP', /)

```

```

DO 22 I=1,3
READ(MR,89) FCOST(I),UVC(I),CAP(I)
FORMAT(2X,F8.0,2X,F9.1,1X,F9.0)
WRITE(MW,99) I,FCOST(I),UVC(I),CAP(I)
FORMAT(17X,I1,5X,F8.0,2X,F6.1,5X,F10.0)
CONTINUE
=====
FORM THE FORWARD STAR REPRESENTATION OF THE NETWORK
=====
CALL FSTAR
WRITE(MW,190)
FORMAT(4(/),12X,'++ FSTAR OF THE NETWORK ++')
WRITE(MW,29) (INODP(I),I=1,NN)
FORMAT(/,15X,'INODP(I)',/,9X,10I4)
NK2=NK*2
WRITE(MW,49) (NOEDGE(J),J=1,NK2)
FORMAT(/,15X,'NOEDGE(J)',5(/,9X,20I4))
=====
CALCULATE THE SYSTEM COSTS
=====
DO 51 J=1,3
CVT(J)=FCOST(J)+UVC(J)*CAP(J)
CONTINUE
=====
INITIALIZE THE INFORMATION ABOUT THE EDGES
=====
DO 50 K=1,NK
XFLW(K)=0.
CURCAP(K)=CAP(1)
LCIN(K)=1
COST(K)=CVT(1)*MD(K)/CAP(1)
INUM(K)=0
IDUM(K)=0
CONTINUE
DO 100 I=1,100
DO 101 J=1,20
KNUM(I,J)=0
FLNUM(I,J)=0.
KDUM(I,J)=0
FLDUM(I,J)=0.
CONTINUE
CONTINUE
=====
CALCULATE THE INITIAL FLOWS ON THE EDGES
=====
DO 12 KR=1,NKR
WRITE(MW,15) KR,CON(KR),CTN(KR)
FORMAT(/////15X,'THE RESULTS FOR RELATION',I4,1X,'(FROM NODE',I3,
&' TO NODE',I3,')',/,14X,56('*'))
-----
FIND THE SHORTEST PATH OF RELATION KR
-----
CALL SHPTH(KR,IELN,COST)
IF(ICONCH.EQ.1) GO TO 12
WRITE(MW,25) (IDEL1(KR,L),L=1,IELN)

```

```
FORMAT(///,15X,'IDEL1 ENTRIES',/,14X,10I4)
```

```
-----
ASSIGN THE FLOW CCL(KR) TO THE EDGES ON THE SHORTEST PATH OF
RELATION KR
-----
```

```
NNN(KR)=IELN
DO 52 L=1,IELN
  K=IDEL1(KR,L)
  INUM(K)=INUM(K)+1
  IL=INUM(K)
  KNUM(K,IL)=KR
  FLNUM(K,IL)=CCL(KR)
  XFLW(K)=XFLW(K)+CCL(KR)
  IF(XFLW(K).LE.CURCAP(K)) GO TO 52
-----
```

```
UPDATE THE SYSTEMS ON THE EDGES
-----
```

```
LCIN(K)=LCIN(K)+1
LS=LCIN(K)
CURCAP(K)=CAP(LS)
IF(XFLW(K).GT.CURCAP(K)) GO TO 33
-----
```

```
UPDATE THE COSTS THAT ARE USED IN FINDING THE SHORTEST PATHS
-----
```

```
COST(K)=(CVT(LS)-CVT(LS-1))*MD(K)/(CAP(LS)-CAP(LS-1))
CONTINUE
WRITE(MW,55)
FORMAT(///,15X,'EDGE',3X,'XFLW',8X,'CURCAP',8X,'COST',/)
DO 53 J=1,NK
  WRITE(MW,54) J,XFLW(J),CURCAP(J),COST(J)
  FORMAT(14X,I4,5X,F8.0,4X,F10.0,4X,F8.0)
CONTINUE
CONTINUE
WRITE(MW,36)
```

```
FORMAT(////,15X,'_KNUM MATRIX_',/)
WRITE(MW,23) ((KNUM(K,IL),IL=1,20),K=1,NK)
FORMAT(9X,20I4)
WRITE(MW,39)
FORMAT(////,15X,'_FLNUM MATRIX_',/)
WRITE(MW,24) ((FLNUM(K,IL),IL=1,20),K=1,NK)
FORMAT(9X,20F4.0)
DO 38 I=1,NK
  STOC(I)=COST(I)
  STOFLW(I)=XFLW(I)
  STOCAP(I)=CURCAP(I)
  STOCIN(I)=LCIN(I)
CONTINUE
-----
```

```
CALCULATE THE REAL INITIAL COSTS OF THE EDGES AND
THE INITIAL TOTAL COST OF THE NETWORK
-----
```

```
WRITE(MW,70)
FORMAT(5(/),15X,'INITIAL FLOWS ON EDGES',/,15X,'_EDGE_',8X,'_XFLW_',
8X,'_COSTBR_',/)
DO 71 L=1,NK
```

```

LS=LCIN(L)
IF(XFLW(L).NE.0.) GO TO 300
COSTBR(L)=0.
GO TO 301
COSTBR(L)=(FCOST(LS)+UVC(LS)*XFLW(L))*MD(L)
WRITE(MW,72) L,XFLW(L),COSTBR(L)
FORMAT(14X,I4,5X,F8.0,3X,F11.0)
CONTINUE
TOTC=0.
DO 73 K=1,NK
TOTC=TOTC+COSTBR(K)
CONTINUE
DO 900 K=1,NK
SBR(K)=COSTBR(K)
CONTINUE
WRITE(MW,74) TOTC
FORMAT(///,15X,_INITIAL TOTAL COST=',F11.0)
=====
PREPARE THE NETWORK FOR THE B&G ALGORITHM BY FORMING ITS
FORWARD STAR REPRESENTATION
=====
CALL STAR2
=====
INITIALIZE THE FLAGS OF THE EDGES

KFLAG DENOTES THE FLAG OF THE EDGE
-1      IF THE EDGE IS NOT YET PROCESSED
 0      IF THE EDGE IS DELETED
 1      IF THE EDGE IS FIXED
=====
DO 16 K=1,NK
KFLAG(K)=-1
CONTINUE
ITT=0
+++++
START THE ALGORITHM
+++++
WRITE(MW,160)
FORMAT(10(/),15X,'CURRENT FLOWS ON EDGES',//,15X,'EDGE',8X,'XFLW',
s/)
DO 162 J=1,NK
WRITE(MW,161) J,XFLW(J)
FORMAT(14X,I4,5X,F8.0)
CONTINUE
DO 102 I=1,NK
DO 103 J=1,20
KDUM(I,J)=KNUM(I,J)
FLDUM(I,J)=FLNUM(I,J)
CONTINUE
IDUM(I)=INUM(I)
CONTINUE
NU=0
=====
CALCULATE THE V(K) VALUES
=====

```

```

CALL VCAL(MD)
WRITE(MW,170) (KFLAG(J),J=1,NK)
FORMAT(///,15X,'FLAGS OF EDGES',/,9X,20I4)
=====
CHECK FOR TERMINATION
=====
DO 83 J=1,NK
IF(KFLAG(J).NE.-1) GO TO 83
IF(V(J).NE.-555555555.) GO TO 46
CONTINUE
=====
PRINT THE FINAL FLOWS AND THE FINAL NETWORK COST
=====
WRITE(MW,191)
FORMAT(4(/),12X,'** NO FURTHER IMPROVEMENT CAN BE MADE, THEREFORE
STOP **')
IF(TCOST.EQ.0..OR.TCOST.GE.TOTC) GO TO 119
WRITE(MW,105)
FORMAT(4(/),15X,'FINAL FLOWS ON EDGES',//,15X,'EDGE',8X,'XFLW',8X,
'RECOST',/)
DO 106 L=1,NK
WRITE(MW,107) L,XFLW(L),SBR(L)
FORMAT(14X,I4,5X,F8.0,3X,F11.0)
CONTINUE
WRITE(MW,104) TCOST
FORMAT(///,15X,'FINAL NETWORK COST=',F11.0)
GO TO 91
WRITE(MW,120)
FORMAT(4(/),15X,'FINAL FLOWS ON EDGES',//,15X,'EDGE',8X,'XFLW',8X,
'RECOST',/)
DO 121 L=1,NK
WRITE(MW,122) L,XFLW(L),SBR(L)
FORMAT(14X,I4,5X,F8.0,3X,F11.0)
CONTINUE
WRITE(MW,123) TOTC
FORMAT(///,15X,'FINAL NETWORK COST=',F11.0)
GO TO 91
=====
INCREASE THE NUMBER OF ITERATIONS BY ONE
=====
ITT=ITT+1
WRITE(MW,192) ITT
FORMAT(///,15X,'NO OF ITERATIONS=',I4)
WRITE(MW,36)
WRITE(MW,23) ((KNUM(K,IL),IL=1,20),K=1,NK)
WRITE(MW,39)
WRITE(MW,24) ((FLNUM(K,IL),IL=1,20),K=1,NK)
WRITE(MW,108)
FORMAT(////,15X,'CURRENT IDEL1 MATRIX',/)
WRITE(MW,109) ((IDEL1(I,J),J=1,20),I=1,NKR)
FORMAT(9X,20I4)
WRITE(MW,223) (INUM(K),K=1,NK)
FORMAT(////,15X,'INUM VALUES',/,9X,20I4)
=====
KM IS THE CANDIDATE EDGE

```



```

-----
IW=INUM(KM)
DO 60 IL=1,IW

```

```

-----
TAKE THE RELATIONS USING EDGE KM ONE BY ONE

```

```

-----
LR=KNUM(KM,IL)
IELN=NNN(LR)
DO 61 IE=1,IELN

```

```

-----
TAKE THE EDGES USING RELATION LR ONE BY ONE

```

```

-----
K1=IDEL1(LR,IE)
JD=INUM(K1)
DO 113 J=1,JD
IF(KNUM(K1,J).NE.LR) GO TO 113
IQ=J
GO TO 114
CONTINUE

```

```

-----
DECREASE THE FLOW ON THESE EDGES BY THE AMOUNT FLW ( OF LR )

```

```

-----
FLW=FLNUM(K1,IQ)
XFLW(K1)=ABS(XFLW(K1)-FLW)
CONTINUE
CONTINUE

```

```

-----
UPDATE THE RELATED ARRAYS

```

```

-----
IW=INUM(KM)
DO 84 I=1,IW
LR=KNUM(KM,I)
DO 80 K=1,NK
MC=INUM(K)
DO 82 J=1,MC
IF(KDUM(K,J).NE.LR) GO TO 82
IDUM(K)=IDUM(K)-1
KDUM(K,J)=0
FLDUM(K,J)=0.
CONTINUE
CONTINUE
CONTINUE
WRITE(MW,163) (IDUM(K),K=1,NK)
FORMAT(///,15X,'IDUM VALUES',/,9X,20I4)
WRITE(MW,193)
FORMAT(4(/),15X,'_DECREASED FLOW VALUES',//,15X,'EDGE',8X,'XFLW',
&/)
DO 194 J=1,NK
WRITE(MW,195) J,XFLW(J)
FORMAT(14X,I4,5X,F8.0)
CONTINUE
DO 81 K=1,NK
IF(K.EQ.KM) GO TO 81
JT=INUM(K)
I=0

```

```

DO 85 J=1,JT
IF(KDUM(K,J).NE.0) GO TO 131
KDUM(K,J)=KDUM(K,J+1)
KDUM(K,J+1)=0
FLDUM(K,J)=FLDUM(K,J+1)
FLDUM(K,J+1)=0.
GO TO 85
I=I+1
CONTINUE
IF(I.NE.IDUM(K)) GO TO 130
CONTINUE

```

CALCULATE THE REMAINING CAPACITIES ON THE EDGES BEFORE ENTERING
THE B&G ALGORITHM FOR REROUTING

```

DO 30 K=1,NK
IF(K.EQ.KM) GO TO 31
IF(XFLW(K).GT.CAP(1)) GO TO 401
REMCAP(K)=CAP(1)-XFLW(K)
IF(REMCAP(K).EQ.0.) GO TO 405
CURCAP(K)=CAP(1)
LCIN(K)=1
COST(K)=CVT(1)*MD(K)/CAP(1)
GO TO 30
CURCAP(K)=CAP(2)
LCIN(K)=2
REMCAP(K)=CAP(2)-XFLW(K)
COST(K)=(CVT(2)-CVT(1))*MD(K)/(CAP(2)-CAP(1))
GO TO 30
IF(XFLW(K).GT.CAP(2)) GO TO 402
REMCAP(K)=CAP(2)-XFLW(K)
IF(REMCAP(K).EQ.0.) GO TO 406
CURCAP(K)=CAP(2)
LCIN(K)=2
COST(K)=(CVT(2)-CVT(1))*MD(K)/(CAP(2)-CAP(1))
GO TO 30
CURCAP(K)=CAP(3)
LCIN(K)=3
REMCAP(K)=CAP(3)-XFLW(K)
COST(K)=(CVT(3)-CVT(2))*MD(K)/(CAP(3)-CAP(2))
GO TO 30
CURCAP(K)=CAP(3)
LCIN(K)=3
REMCAP(K)=CAP(3)-XFLW(K)
COST(K)=(CVT(3)-CVT(2))*MD(K)/(CAP(3)-CAP(2))
GO TO 30
LS=LCIN(K)
IF(LS.EQ.1) GO TO 500
REMCAP(K)=CAP(LS-1)
IF(LS.EQ.2) GO TO 501
COST(K)=(CVT(2)-CVT(1))*MD(K)/(CAP(2)-CAP(1))
GO TO 30
COST(K)=CVT(1)*MD(K)/CAP(1)
GO TO 30
REMCAP(K)=0.

```

```

COST(K)=999999999.
CONTINUE
DO 930 J=1,NK
IF(KFLAG(J).NE.0) GO TO 930
COST(J)=999999999.
CONTINUE
=====
REROUTE THE CIRCUIT DEMAND OF THE RELATIONS USING EDGE KM
=====
CALL M2(COST)
IF(RESULT.EQ.1.) GO TO 91
-----
IS A FEASIBLE FLOW PATTERN OBTAINED ? IF NO, FIX THE SYSTEM
ON EDGE KM
-----
IF(ICUNCH.EQ.1) GO TO 65
WRITE(MW,21)
FORMAT(////,15X,'K2NUM MATRIX',/)
WRITE(MW,28) ((K2NUM(K,IL),IL=1,20),K=1,NK)
FORMAT(9X,20I4)
WRITE(MW,35)
FORMAT(////,15X,'FL2NUM MATRIX',/)
WRITE(MW,37) ((FL2NUM(K,IL),IL=1,20),K=1,NK)
FORMAT(9X,20F4.0)
WRITE(MW,164) (I2NUM(K),K=1,NK)
FORMAT(////,15X,'I2NUM VALUES',/,9X,20I4)
WRITE(MW,165)
FORMAT(6(/),15X,'FLOWS OBTAINED AFTER APPLYING THE B&G ALGORITHM_',
/,15X,'EDGE',8X,'SUMFLW',/)
DO 166 KI=1,MG
WRITE(MW,167) KI,SUMFLW(KI)
FORMAT(14X,I4,7X,F8.0)
CONTINUE
-----
CALCULATE THE FLOWS AFTER REROUTING
-----
DO 66 I=1,MG
SUMFLW(I)=SUMFLW(I)+XFLW(I)
CONTINUE
WRITE(MW,41)
FORMAT(5(/),15X,'FLOWS AFTER REROUTING',/,15X,'EDGE',8X,'XFLW',
8X,'COSTAR',/)
-----
UPDATE THE SYSTEMS ON THE EDGES
-----
DO 40 K=1,MG
IF(SUMFLW(K).GT.CAP(1)) GO TO 43
LCIN(K)=1
CURCAP(K)=CAP(1)
GO TO 44
IF(SUMFLW(K).GT.CAP(2)) GO TO 45
LCIN(K)=2
CURCAP(K)=CAP(2)
GO TO 44
LCIN(K)=3

```

CURCAP(K)=CAP(3)

CALCULATE THE REAL COSTS OF THE EDGES AFTER REROUTING

LS=LCIN(K)
IF(SUMFLW(K).NE.0.) GO TO 302
COSTAR(K)=0.
GO TO 303
COSTAR(K)=(FCOST(LS)+UVC(LS)*SUMFLW(K))*MD(K)
WRITE(MW,42) K,SUMFLW(K),COSTAR(K)
FORMAT(14X,I4,5X,F8.0,3X,F11.0)
IF(KFLAG(K).NE.0) GO TO 198
COST(K)=999999999.
GO TO 40

UPDATE THE COSTS THAT ARE USED IN FINDING THE SHORTEST PATHS

IF(LS.EQ.1) GO TO 197
COST(K)=(CVT(LS)-CVT(LS-1))*MD(K)/(CAP(LS)-CAP(LS-1))
GO TO 40
COST(K)=CVT(1)*MD(K)/CAP(1)
CONTINUE
IF(SUMFLW(KM).NE.0.) GO TO 503
COST(KM)=999999999.

CALCULATE THE TOTAL NETWORK COST AFTER REROUTING

TCOST=0.
DO 56 K=1,MG
TCOST=TCOST+COSTAR(K)
CONTINUE
WRITE(MW,26) TCOST
FORMAT(///,15X,'TOTAL COST AFTER REROUTING=',F11.0)

=====

COST COMPARISON
=====

IF(TCOST.LT.TOTC) GO TO 115
NO=1
GO TO 65
=====

EDGE KM CAN BE DELETED
=====

DO 48 K=1,MG
XFLW(K)=SUMFLW(K)
STOFLW(K)=XFLW(K)
STORE(K)=COST(K)
STOCAP(K)=CURCAP(K)
STOCIN(K)=LCIN(K)
SBR(K)=COSTAR(K)
CONTINUE

UPDATE THE RELATED ARRAYS

DO 20 K=1,MG
JF=IDUM(K)

```

DO 88 J=1,10
IF(K.EQ.KM) GO TO 87
JJ=JF+J
KDUM(K,JJ)=K2NUM(K,J)
FLDUM(K,JJ)=FL2NUM(K,J)
GO TO 88
KDUM(K,J)=K2NUM(K,J)
FLDUM(K,J)=FL2NUM(K,J)
CONTINUE
CONTINUE

```

```

-----
UPDATE THE IDEL1 MATRIX
-----

```

```

IV=INUM(KM)
DO 110 L=1,IV
IR=KNUM(KM,L)
NC=NNN(IR)
DO 112 LL=1,NC
IDEL1(IR,LL)=0
CONTINUE
JE=NDUM(IR)
NNN(IR)=JE
DO 111 N=1,JE
IDEL1(IR,N)=IDELDM(IR,N)
CONTINUE
CONTINUE
DO 92 K=1,MG
DO 93 L=1,20
KNUM(K,L)=KDUM(K,L)
FLNUM(K,L)=FLDUM(K,L)
CONTINUE
INUM(K)=IDUM(K)+I2NUM(K)
CONTINUE

```

```

-----
SINCE KM CAN BE DELETED, SET ITS FLAG TO 0
-----

```

```

KFLAG(KM)=0
-----

```

```

UPDATE THE TOTAL COST OF THE NETWORK
-----

```

```

TOTC=0.
TOTC=TCOST
WRITE(MW,34) KM
FORMAT(///,12X,'** SINCE A FEASIBLE FLOW PATTERN HAS BEEN FOUND, A
&ND THE TOTAL COST AFTER REROUTING',/,15X,'IS LESS THAN THE PREVIOUS
&S COST, EDGE',I3,' CAN BE DELETED **')
GO TO 90
=====
EDGE KM CANNOT BE DELETED, THEREFORE FIX KM BY SETTING ITS
FLAG TO 1
=====
IF(NO.NE.1) GO TO 116
WRITE(MW,117) KM
FORMAT(///,12X,'** A FEASIBLE FLOW PATTERN HAVING A TOTAL COST GRE
&ATER THAN THE PREVIOUS ONE',/,15X,'HAS BEEN OBTAINED, THEREFORE ED

```

```
GE_,I3, _ CANNOT BE DELETED **')  
KFLAG(KM)=1  
DO 17 J=1, MG  
COST(J)=STORE(J)  
XFLW(J)=STOFLW(J)  
CURCAP(J)=STOCAP(J)  
LCIN(J)=STOCIN(J)  
CONTINUE  
GO TO 90  
STOP  
END
```

SUBROUTINE SORT

```
*****  
SORTING THE CIRCUIT DEMANDS IN DESCENDING ORDER  
*****  
COMMON /XXX/CON(150),CTN(150),CCL(150),LCN  
INTEGER CON,CTN  
I=0  
DO 1 K=1,LCN-1  
IF(CCL(K).GE.CCL(K+1)) GO TO 1  
DM=CCL(K)  
CCL(K)=CCL(K+1)  
CCL(K+1)=DM  
NC1=CON(K)  
CON(K)=CON(K+1)  
CON(K+1)=NC1  
NC2=CTN(K)  
CTN(K)=CTN(K+1)  
CTN(K+1)=NC2  
I=I+1  
CONTINUE  
IF(I.NE.0) GO TO 2  
RETURN  
END
```

SUBROUTINE FSTAR

THE FORWARD STAR REPRESENTATION OF A NETWORK

COMMON /AAA/MN, MG, LLL, MON(300), MTN(300), REMCAP(300), ITWIN(300),

LDASH(300), ICONCH

COMMON /XYZ/INODP(100), NOEDGE(300), IDEL1(150, 20)

DIMENSION NEDGE(100)

NK=MG

DO 45 JI=1, NN

NEDGE(JI)=0

INODP(JI)=0

CONTINUE

DO 47 IO=1, NK

NOEDGE(IO)=0

CONTINUE

=====

CALCULATE THE OUTGOING DEGREE OF THE NODES

=====

DO 1 K=1, NK

LOR=MON(K)

NEDGE(LOR)=NEDGE(LOR)+1

LOR=MTN(K)

NEDGE(LOR)=NEDGE(LOR)+1

CONTINUE

=====

FORM THE POINTER ARRAY

=====

INODP(1)=1

DO 2 IN=1, NN

INODP(IN+1)=INODP(IN)+NEDGE(IN)

CONTINUE

=====

ASSIGN THE EDGES TO THE POINTER ARRAY

=====

DO 3 K=1, NK

LOR=MON(K)

IZ=INODP(LOR)

NOEDGE(IZ)=K

INODP(LOR)=INODP(LOR)+1

LOR=MTN(K)

IZ=INODP(LOR)

NOEDGE(IZ)=K

INODP(LOR)=INODP(LOR)+1

CONTINUE

DO 4 I=1, NN

INODP(I)=INODP(I)-NEDGE(I)

CONTINUE

RETURN

END


```

SUBROUTINE SHPTH(KR,LM,MD)
*****
THE SHORTEST PATH ALGORITHM
*****
COMMON /AAA/NN, MG, LLL, MON(300), MTN(300), REMCAP(300), ITWIN(300),
IDASH(300), ICONCH
COMMON /XXX/CON(150), CTN(150), CCL(150), LCN
COMMON /XYZ/INODP(100), NOEDGE(300), IDEL1(150,20)
DIMENSION LABL(100), IPER(100), ITEMP(100), LSAVE(100), MD(300)
INTEGER CON, CTN
REAL MD, M, MIN, LABL, LS
MR=5
MW=6
NK=MG
NII=NN
ICONCH=0

```

INITIALIZATION: ITEMP(K) IS THE TEMPORARY FILE WHICH CONTAINS THE
NODES TEMPORARILY LABELED

```

DO 36 K=1, NN
ITEMP(K)=0
CONTINUE

```

INITIALIZATION: IDEL1(KR, LI) IS THE MATRIX WHICH STORES THE EDGE
NUMBERS IN THE SHORTEST PATH OF TRAFFIC RELATION KR

```

DO 37 LI=1, LLL
IDEL1(KR, LI)=0
CONTINUE
LM=0
M=99999999.

```

=====

```

IS=CON(KR)

```

=====

```

IT=CTN(KR)

```

=====

INITIALIZATION OF THE LABELS OF NODES BY A BIG NUMBER M

IPER DENOTES THE FLAG OF THE NODE
0 IF THE NODE IS NOT YET PROCESSED
1 IF THE NODE IS TEMPORARILY LABELED
2 IF THE NODE IS PERMANENTLY LABELED

```

DO 1 LN=1, NII
LABL(LN)=M
IPER(LN)=0
CONTINUE

```

=====

INITIALIZATION OF THE BEGINNING NODE
=====

```

LABL(IS)=0
IPER(IS)=2
I=IS
LL=0

```

START THE ALGORITHM

FIND THE BEGINNING AND ENDING POSITIONS OF NODE I

```

LBEG=INODP(I)
LEND=INODP(I+1)-1
DO 20 L=LBEG,LEND

```

=====

TAKE THE EDGES IN THE FORWARD STAR OF NODE I ONE BY ONE
=====

```

KE=NOEDGE(L)

```

COMPARE I WITH THE TERMINATING NODE OF THE EDGE UNDER
CONSIDERATION

```

IF(I.NE.MTN(KE)) J=MTN(KE)
IF(I.EQ.MTN(KE)) J=MON(KE)

```

SKIP THE NODE IF IT IS PERMANENTLY LABELED

```

IF(IPER(J).EQ.2) GO TO 20

```

CALCULATE THE TEMPORARY LABEL OF THE NODE

```

LS=LABL(I)+MD(KE)
IF(LS.GE.LABL(J)) GO TO 20
LABL(J)=LS

```

STORE THE EDGE NUMBER WHICH IS USED IN LABELING NODE J

```

LSAVE(J)=KE
IF(IPER(J).NE.0) GO TO 20

```

SET THE FLAG OF NODE J TO 1

```

IPER(J)=1
LL=LL+1

```

STORE THE NODE NUMBER IN THE TEMPORARY FILE

```

ITEMP(LL)=J
CONTINUE

```

=====

FIND THE NODE WITH THE MINIMUM TEMPORARY LABEL AND
MAKE ITS LABEL PERMANENT
=====

```

MIN=M
NEND=LL
IF(LL.EQ.0) GO TO 80

```

DD 25 L=1,NEND

 TAKE THE NODE NUMBER FROM THE TEMPORARY FILE

JJ=ITEMP(L)
 IF(IPER(JJ).EQ.2) GO TO 25
 IF(LABL(JJ).GE.MIN) GO TO 25
 MIN=LABL(JJ)
 NSAVE=JJ
 CONTINUE

=====

CHECK FOR CONNECTIVITY

=====

IF(MIN.LT.99999999.) GO TO 228
 WRITE(MW,222)
 FORMAT(///,12X,'** THE NETWORK BECAME DISCONNECTED **')
 ICONCH=1
 GO TO 30
 IPER(NSAVE)=2

=====

CHECK FOR TERMINATION

=====

IF(NSAVE.EQ.IT) GO TO 15
 I=NSAVE
 GO TO 10

=====

BACKTRACING THE SHORTEST PATH

=====

IJ=IT

 TAKE THE EDGE NUMBER WHICH IS USED IN LABELING NODE IJ

KK=LSAVE(IJ)
 LM=LM+1
 IF(LM.GT.LLL) GO TO 33

 INPUT THE EDGE NUMBER INTO THE IDEL1 MATRIX

IDEL1(KR,LM)=KK
 II=IJ
 IF(II.EQ.MTN(KK)) IJ=MON(KK)
 IF(II.EQ.MON(KK)) IJ=MTN(KK)
 IF(IJ.EQ.IS) GO TO 30
 GO TO 35

WRITE(MW,40)
 FORMAT(///,12X,'** INCREASE THE VALUE OF LLL AND THEN USE THE ALGO
 RITHM AGAIN **')
 ICONCH=1
 RETURN
 END

SUBROUTINE STAR2

THE MAIN PROGRAM OF MINCTF (-1-)

COMMON /AAA/NN, MG, LLL, MON(300), MTN(300), MMC(300), ITWIN(300),
IDASH(300), ICONCH

COMMON /ZZZ/INODP2(100), NOEDG2(300), IDEL2(150,20), RESULT

COMMON /CCC/IMON(300), IMTN(300), TMD(300), TMMC(300)

MW=6

NK=MG

NKR=LCN

NK=NK*4

THE NUMBER OF EDGES IN THE NETWORK IS INCREASED TO (NK*4)

IMON(1)=MON(1)

IMTN(1)=MTN(1)

MGM=MG-1

DO 6 K=1, MGM

KX=(K*4)+1

IMON(KX)=MON(K+1)

IMTN(KX)=MTN(K+1)

6 CONTINUE

=====

THE PROCEDURE BELOW PREPARES THE NETWORK FOR THE MIN-COST FLOW

ALGORITHM OF BUSACKER AND GOWEN

THE EDGES 1,3,5,7,... ARE REFERRED TO AS SOLID EDGES, AND THE
EDGES 2,4,6,8,... ARE REFERRED TO AS 'DASHED' EDGES. A SOLID EDGE
HAS A 'TWIN' SOLID EDGE AND A DASHED EDGE. A DASHED EDGE, ON THE
OTHER HAND, HAS ONLY A SOLID EDGE.

DO 5 K=1, NK, 4

IMON(K+1)=IMTN(K)

IMTN(K+1)=IMON(K)

IDASH(K)=K+1

IDASH(K+1)=K

IMON(K+2)=IMTN(K)

IMTN(K+2)=IMON(K)

ITWIN(K)=K+2

ITWIN(K+2)=K

IMON(K+3)=IMON(K)

IMTN(K+3)=IMTN(K)

IDASH(K+2)=K+3

IDASH(K+3)=K+2

5 CONTINUE

=====

FORM THE FORWARD STAR REPRESENTATION OF THE NETWORK

IN MINCTF (BY B&G)

=====

CALL FSTAR2

WRITE(MW,3)

3 FORMAT(3(/),12X,'++ FSTAR OF THE NETWORK IN MINCTF ++')

WRITE(MW,29) (INODP2(I), I=1, NN)

29 FORMAT(//,15X,'INODP2(I)',/,9X,10I4)

```
WRITE(MW,49) (NOEDG2(J),J=1,NK)  
9 FORMAT(///,15X,'NOEDG2(J)',5(/,9X,20I4))  
RETURN  
END
```

SUBROUTINE VCAL(MD)

CALCULATION OF THE V(K) VALUES

COMMON /AAA/NN, MG, LLL, MDN(300), MTN(300), REMCAP(300), ITWIN(300),

&IDASH(300), ICONCH

COMMON /YYY/FCOST(3), UVC(3), CAP(3), LCIN(300), CURCAP(300), XFLW(300)

COMMON /VVV/V(300), KM, INUM(300), KNUM(100,20), FLNUM(100,20),

&KFLAG(300)

DIMENSION MD(300), DELFL(300)

REAL MD

MW=6

NK=MG

DO 6 K=1, NK

V(K)=0.

6 CONTINUE

DO 1 K=1, NK

LS=LCIN(K)

IF(XFLW(K).EQ.0.) GO TO 3

REALCT=FCOST(LS)+UVC(LS)*XFLW(K)

V(K)=REALCT*MD(K)/XFLW(K)

GO TO 1

3 V(K)=-555555555.

1 CONTINUE

=====

THE V(K) VALUE OF A FIXED OR DELETED EDGE IS SET TO INFINITY

=====

DO 7 L=1, NK

IF(KFLAG(L).EQ.-1) GO TO 7

V(L)=-555555555.

7 CONTINUE

DO 10 I=1, NK

IF(V(I).NE.-555555555.) GO TO 11

10 CONTINUE

GO TO 12

=====

FIND THE MAXIMUM OF THE V(K) VALUES

=====

11 VMAX=-555555555.

DO 2 K=1, NK

IF(V(K).LE.VMAX) GO TO 2

VMAX=V(K)

KM=K

2 CONTINUE

WRITE(MW,8) (K, V(K), K=1, NK)

8 FORMAT(5(/), 15X, 'V(K) VALUES', //, 10(2X, 4(I3, '=>', F11.0, 2X), /))

WRITE(MW,9) KM

9 FORMAT(//, 15X, 'KM=', I3, 2X, '(THE MAXIMUM V(K) VALUE)')

12 RETURN

END

SUBROUTINE M2(MD)

THE MAIN PROGRAM OF MINCTF (-2-)

COMMON /AAA/NN, MG, LLL, MON(300), MTN(300), MMC(300), ITWIN(300),
&IDASH(300), ICONCH

COMMON /XXX/CON(150), CTN(150), CCL(150), LCN

COMMON /ZZZ/INODP2(100), NOEDG2(300), IDEL2(150,20), RESULT

COMMON /EEE/NDUM(150), IDELDM(150,20)

COMMON /IJK/STOMD(300), SUMFLW(300)

COMMON /CCC/IMON(300), IMTN(300), TMD(300), TMMC(300)

COMMON /VVV/V(300), KM, INUM(300), KNUM(100,20), FLNUM(100,20),
&KFLAG(300)

COMMON /TTT/NOE(150), I2NUM(300), K2NUM(100,20), FL2NUM(100,20)

DIMENSION MD(300), XMD(300), XMMC(300)

INTEGER CON, CTN

REAL MD, MMC

MW=6

NK=MG

NKR=LCN

NK=NK*4

DO 41 I=1,150

NOE(I)=0

41 CONTINUE

DO 42 I=1,300

I2NUM(I)=0

42 CONTINUE

DO 43 I=1,100

DO 44 J=1,20

K2NUM(I,J)=0

FL2NUM(I,J)=0.

44 CONTINUE

43 CONTINUE

DO 4 K=1,NK

SUMFLW(K)=0.

4 CONTINUE

THE NUMBER OF EDGES IN THE NETWORK IS INCREASED TO (NK*4)

TMD(1)=MD(1)

TMMC(1)=MMC(1)

MGM=MG-1

DO 6 K=1,MGM

KX=(K*4)+1

TMD(KX)=MD(K+1)

TMMC(KX)=MMC(K+1)

6 CONTINUE

=====
THE PROCEDURE BELOW PREPARES THE NETWORK FOR THE MIN-COST FLOW
ALGORITHM OF BUSACKER AND GOWEN
=====

DO 5 K=1,NK,4

TMD(K+1)=999999999.

TMMC(K+1)=0.

TMD(K+2)=TMD(K)

```

TMMC(K+2)=TMMC(K)
TMD(K+3)=999999999.
TMMC(K+3)=0.
5 CONTINUE
IF(MD(1).NE.999999999.) GO TO 10
TMD(1)=999999999.
TMD(3)=999999999.
10 DO 9 L=1,MSM
IF(MD(L+1).NE.999999999.) GO TO 9
KV=(L*4)+1
TMD(KV)=999999999.
JER=ITWIN(KV)
TMD(JER)=999999999.
9 CONTINUE
DO 39 IE=1,NK
STMD(IE)=TMD(IE)
39 CONTINUE
DO 14 IJ=1,NK
IF(TMD(IJ).NE.999999999.) GO TO 14
TMMC(IJ)=0.
14 CONTINUE
DO 56 I=1,NK
XMMC(I)=0.
XMMC(I)=TMMC(I)
56 CONTINUE
WRITE(MW,18) (J,TMD(J),J=1,NK)
18 FORMAT(6(/),15X,'ARC COSTS',//,22(2X,4(I3,'=>',F11.0,2X),/))
=====
FIND THE RELATIONS USING THE CANDIDATE EDGE KM
=====
IK=INUM(KM)
DO 7 J=1,IK
NR=KNUM(KM,J)
WRITE(MW,15) NR,CON(NR),CTN(NR)
15 FORMAT(///// ,15X,'THE RESULTS FOR RELATION',I4,1X,'(FROM NODE',
&I3,' TO NODE_',I3,')',/,14X,56('*'))
DO 22 LP=1,20
IDELDM(NR,LP)=0
22 CONTINUE
-----
REROUTE THE CIRCUIT DEMAND OF RELATION NR
-----
CALL MINCTF(NR,TMD)
IF(RESULT.EQ.1.) GO TO 88
=====
CHECK IF A FEASIBLE FLOW PATTERN IS OBTAINED
=====
IF(ICONCH.EQ.1) GO TO 33
7 CONTINUE
=====
CALCULATE THE FLOW ON EACH EDGE OBTAINED AFTER APPLYING THE
B&G ALGORITHM
=====
DO 51 K=1,NG
NV=I2NUM(K)

```



```
DO 52 J=1,NV
SUMFLW(K)=SUMFLW(K)+FL2NUM(K,J)
52 CONTINUE
51 CONTINUE
GO TO 88
33 WRITE(MW,87) KM
87 FORMAT(///,12X,'** NO FEASIBLE FLOW PATTERN HAS BEEN FOUND, THEREF
ORE EDGE',I3,' CANNOT HAVE A SMALLER SYSTEM OR CANNOT BE DELETED *
**')
88 RETURN
END
```

SUBROUTINE FSTAR2

THE FORWARD STAR REPRESENTATION OF A NETWORK (-TO BE USED IN B&G-)

COMMON /AAA/NN, MG, LLL, MON(300), MTN(300), REMCAP(300), ITWIN(300),
&IDASH(300), ICONCH

COMMON /ZZZ/INODP2(100), NOEDG2(300), IDEL2(150,20), RESULT

COMMON /CCC/IMON(300), IMTN(300), TMD(300)

DIMENSION NEDGE2(100)

NK=MG

NK=NK*4

DO 45 JI=1, NN

NEDGE2(JI)=0

INODP2(JI)=0

45 CONTINUE

DO 47 IO=1, NK

NOEDG2(IO)=0

47 CONTINUE

=====
CALCULATE THE OUTGOING DEGREE OF THE NODES
=====

DO 1 K=1, NK, 4

LOR=IMON(K)

NEDGE2(LOR)=NEDGE2(LOR)+2

LOR=IMTN(K)

NEDGE2(LOR)=NEDGE2(LOR)+2

1 CONTINUE

=====
FORM THE POINTER ARRAY
=====

INODP2(1)=1

DO 2 IN=1, NN

INODP2(IN+1)=INODP2(IN)+NEDGE2(IN)

2 CONTINUE

=====
ASSIGN THE EDGES TO THE POINTER ARRAY
=====

DO 3 K=1, NK, 4

LOR=IMON(K)

IZ=INODP2(LOR)

NOEDG2(IZ)=K

KOR=ITWIN(K)

NOEDG2(IZ+1)=IDASH(KOR)

INODP2(LOR)=INODP2(LOR)+2

LOR=IMTN(K)

MOR=IDASH(K)

IZ=INODP2(LOR)

NOEDG2(IZ)=MOR

NOEDG2(IZ+1)=ITWIN(K)

INODP2(LOR)=INODP2(LOR)+2

3 CONTINUE

DO 4 I=1, NN

INODP2(I)=INODP2(I)-NEDGE2(I)

4 CONTINUE

RETURN

END

SUBROUTINE SHPTH2(KR,LM,ND)

THE SHORTEST PATH ALGORITHM (-TO BE USED IN 8&G-)

COMMON /AAA/NN, MG, LLL, MON(300), MTN(300), REMCAP(300), ITWIN(300),
IDASH(300), ICONCH

COMMON /XXX/CON(150), CTN(150), CCL(150), LCN

COMMON /ZZZ/INODP2(100), NOEDG2(300), IDEL2(150,20), RESULT

COMMON /CCC/IMON(300), IMTN(300), TMD(300), TMMC(300)

DIMENSION LABL(100), IPER(100), ITEMP(100), LSAVE(100), MD(300)

INTEGER CON, CTN

REAL MD, II, MIN, LABL, LS

NR=5

MW=6

NK=MG

NK=NK#4

NII=NN

ICONCH=0

INITIALIZATION: ITEMP(K) IS THE TEMPORARY FILE WHICH CONTAINS THE
NODES TEMPORARILY LABELED

DO 36 K=1, NN

ITEMP(K)=0

CONTINUE

INITIALIZATION: IDEL2(KR, LI) IS THE MATRIX WHICH STORES THE EDGE
NUMBERS IN THE SHORTEST PATH OF TRAFFIC RELATION KR

DO 37 LI=1, LLL

IDEL2(KR, LI)=0

CONTINUE

LM=0

M=99999999.

=====

IS IS THE BEGINNING NODE OF RELATION KR

IS=CON(KR)

=====

IT IS THE TERMINATING NODE OF RELATION KR

IT=CTN(KR)

=====

INITIALIZATION OF THE LABELS OF NODES BY A BIG NUMBER M

IPER DENOTES THE FLAG OF THE NODE

0 IF THE NODE IS NOT YET PROCESSED

1 IF THE NODE IS TEMPORARILY LABELED

2 IF THE NODE IS PERMANENTLY LABELED

=====

DO 1 LN=1, NII

LABL(LN)=M

IPER(LN)=0

CONTINUE

=====

1

INITIALIZATION OF THE BEGINNING NODE

```
=====
LABL(IS)=0
IPER(IS)=2
I=IS
LL=0
```

START THE ALGORITHM

FIND THE BEGINNING AND ENDING POSITIONS OF NODE I

```
10 LBEG=INODP2(I)
LEND=INODP2(I+1)-1
DO 20 L=LBEG,LEND
```

=====
TAKE THE EDGES IN THE FORWARD STAR OF NODE I ONE BY ONE
=====

```
KE=NOEDG2(L)
```

COMPARE I WITH THE TERMINATING NODE OF THE EDGE UNDER
CONSIDERATION

```
IF(I.NE.IMTN(KE)) J=IMTN(KE)
IF(I.EQ.IMTN(KE)) J=IMON(KE)
```

SKIP THE NODE IF IT IS PERMANENTLY LABELED

```
IF(IPER(J).EQ.2) GO TO 20
```

CALCULATE THE TEMPORARY LABEL OF THE NODE

```
LS=LABL(I)+ND(KE)
IF(LS.GE.LABL(J)) GO TO 20
LABL(J)=LS
```

STORE THE EDGE NUMBER WHICH IS USED IN LABELING NODE J

```
LSAVE(J)=KE
IF(IPER(J).NE.0) GO TO 20
```

SET THE FLAG OF NODE J TO 1

```
IPER(J)=1
LL=LL+1
```

STORE THE NODE NUMBER IN THE TEMPORARY FILE

```
ITEMP(LL)=J
```

```
20 CONTINUE
```

=====
FIND THE NODE WITH THE MINIMUM TEMPORARY LABEL AND
MAKE ITS LABEL PERMANENT
=====

```
MIN=M
```

```

NEND=LL
IF(LL.EQ.0) GO TO 80
DO 25 L=1,NEND
-----
TAKE THE NODE NUMBER FROM THE TEMPORARY FILE
-----
JJ=ITEMP(L)
IF(IPER(JJ).EQ.2) GO TO 25
IF(LABL(JJ).GE.MIN) GO TO 25
MIN=LABL(JJ)
NSAVE=JJ
CONTINUE
=====
CHECK FOR CONNECTIVITY
=====
IF(MIN.LT.99999999.) GO TO 228
WRITE(MW,222)
FORMAT(///,12X,'** THE NETWORK BECAME DISCONNECTED **')
ICONCH=1
GO TO 30
IPER(NSAVE)=2
=====
CHECK FOR TERMINATION
=====
IF(NSAVE.EQ.IT) GO TO 15
I=NSAVE
GO TO 10
=====
BACKTRACING THE SHORTEST PATH
=====
IJ=IT
-----
TAKE THE EDGE NUMBER WHICH IS USED IN LABELING NODE IJ
-----
KK=LSAVE(IJ)
LM=LM+1
IF(LM.GT.LLL) GO TO 33
-----
INPUT THE EDGE NUMBER INTO THE IDEL2 MATRIX
-----
IDEL2(KR,LM)=KK
II=IJ
IF(II.EQ.IMTN(KK)) IJ=IMON(KK)
IF(II.EQ.IMON(KK)) IJ=IMTN(KK)
IF(IJ.EQ.IS) GO TO 30
GO TO 35
WRITE(MW,40)
FORMAT(///,12X,'** INCREASE THE VALUE OF LLL AND THEN USE THE ALGO
RITHM AGAIN **')
ICONCH=1
RETURN
END

```

```

SUBROUTINE MINCTF(KR,XMD)
*****
THIS SUBROUTINE CALCULATES THE MINIMUM COST FLOW PATTERN FROM S TO
T WITH A SPECIFIED FLOW VALUE ( DUE TO BUSACKER & GOWEN )
*****

```

```

COMMON /AAA/NN, MG, LLL, MON(300), MTN(300), REMCAP(300), ITWIN(300),
&IDASH(300), ICONCH
COMMON /XXX/CON(150), CTN(150), CCL(150), LCN
COMMON /ZZZ/INODP2(100), NOEDG2(300), IDEL2(150,20), RESULT
COMMON /EEE/NDUM(150), IDELDM(150,20)
COMMON /IJK/STOMD(300), SUMFLW(300)
COMMON /CCC/IMON(300), IMTN(300), TMD(300), XMMC(300)
COMMON /TTT/NOE(150), I2NUM(300), K2NUM(100,20), FL2NUM(100,20)
DIMENSION FLOW(300), LFLAG(300), XMD(300), OLDMD(300)
INTEGER CON, CTN

```

```

MW=6
VALUE=0.
NK=MG
NK=NK*4
NDUM(KR)=0
DO 32 J=1, NK
XMD(J)=0.
XMD(J)=STOMD(J)
LFLAG(J)=0
CONTINUE
DO 50 K=1, NK, 2
IED=IDASH(K)
XMMC(IED)=0.
XMD(IED)=999999999.
CONTINUE
WRITE(MW,36) (J,XMMC(J),J=1,NK)
&FORMAT(///,15X,'INITIAL FLOW VALUES',//,11(8X,8(I5,'=>',F6.0),
&///))

```

```

*****
START THE ALGORITHM
*****
DO 55 I=1, NK
IF(XMMC(I).LT.0.0001E+00.OR.XMMC(I).GT.0.999999E+00) GO TO 55
XMMC(I)=0.0E+00
CONTINUE
DO 77 JR=1, NK, 2
IF(XMMC(JR).NE.0.) GO TO 76
XMD(JR)=999999999.
JFOR=IDASH(JR)
XMD(JFOR)=999999999.
GO TO 77
IFOR=IDASH(JR)
IF(XMMC(IFOR).EQ.0.) GO TO 77
XMD(IFOR)=-XMD(JR)
CONTINUE
DO 15 J=1, NK
OLDMD(J)=0.
OLDMD(J)=XMD(J)
CONTINUE
=====

```

REMOVE THE NEGATIVE COSTS

=====

CALL REMOVE(XMD)

IF(RESULT.EQ.1.) GO TO 10

=====

FIND THE SHORTEST PATH OF RELATION KR

=====

CALL SHPTH2(KR,LM,XMD)

IF(ICDNCH.EQ.1) GO TO 10

WRITE(MW,43) (IDEL2(KR,J),J=1,LM)

FORMAT(////,15X,'IDEL2 ENTRIES',/,14X,10I4)

DO 16 JK=1,NK

XMD(JK)=0.

XMD(JK)=OLDMD(JK)

CONTINUE

=====

CONSTRUCT THE INCREMENTAL GRAPH

CALCULATE THE LARGEST AMOUNT OF FLOW (FMIN) THAT CAN BE SENT
ALONG THE PATH

FMIN=99999999.

DO 2 L=1,LM

KX=IDEL2(KR,L)

IF(XMNC(KX).GE.FMIN) GO TO 2

FMIN=XMNC(KX)

CONTINUE

VALUE=VALUE+FMIN

WRITE(MW,57) FMIN,VALUE

FORMAT(////,15X,'INTERMEDIATE FLOW VALUE=',F6.0,/,15X,

&'CUMULATIVE FLOW VALUE =',F6.0)

CHECK IF THE GIVEN FLOW VALUE IS EXCEEDED

IF(VALUE.GT.CCL(KR)) GO TO 7

=====

UPDATE THE FLOWS ON THE EDGES

DO 3 L=1,LM

KL=IDEL2(KR,L)

FLOW(KL)=0.

FLOW(KL)=FMIN

IF(FLOW(KL).GT.XMNC(KL)) GO TO 3

XMNC(KL)=XMNC(KL)-FLOW(KL)

IF KL IS A SOLID EDGE, DECREASE THE FLOW ON ITS TWIN SOLID EDGE
AND INCREASE THE FLOW ON ITS DASHED EDGE BY THE AMOUNT FMIN

IF(KL/2*2.EQ.KL) GO TO 19

JDR=ITWIN(KL)

XMNC(JDR)=XMNC(JDR)-FLOW(KL)

IF KL IS A DASHED EDGE, INCREASE THE FLOWS ON ITS SOLID EDGE AND
ON THE TWIN OF ITS SOLID EDGE BY THE AMOUNT FMIN


```

-----
KOR=IDASH(KL)
XMMC(KOR)=XMMC(KOR)+FLOW(KL)
-----
IF KL IS A SOLID EDGE, GO TO 18
-----
IF(KOR/2*2.EQ.KOR) GO TO 18
MOR=ITWIN(KOR)
XMMC(MOR)=XMMC(MOR)+FLOW(KL)
GO TO 1
=====
UPDATE THE COSTS OF THE EDGES
=====
IF(LFLAG(KOR).EQ.1) GO TO 1
LFLAG(KOR)=1
XMD(KOR)=-XMD(KL)
IF(XMMC(KL).NE.0.) GO TO 3
IF(LFLAG(KL).EQ.1) GO TO 3
LFLAG(KL)=1
XMD(KL)=999999999.
IF(LFLAG(JOR).EQ.1) GO TO 3
LFLAG(JOR)=1
XMD(JOR)=999999999.
CONTINUE
WRITE(MW,73) (I,XMMC(I),I=1,NK)
FORMAT(////,15X,'UPDATED FLOW VALUES',//,11(3X,8(I5,'=>',F6.0),
//))
VDUM=VALUE
FDUM=FMIN
-----
STORE THE ABOVE INFORMATION IN I2NUM(.),K2NUM(.,.) AND FL2NUM(.,.)
-----
IF(VALUE.LE.CCL(KR)) GO TO 20
VDUM=FMIN-VALUE+CCL(KR)
FDUM=VDUM
NOE(KR)=LM
NUM=NDUM(KR)
NDUM(KR)=NDUM(KR)+LM
DO 40 L=1,LM
KA=IDEL2(KR,L)
-----
CONVERT THE EDGE NUMBERS TO THEIR ORIGINAL VALUES
-----
XKA=FLOAT(KA)/4.
TKA=XKA-INT(XKA)
IF(TKA.EQ.0.) GO TO 4
INEW=INT(XKA+1.)
GO TO 5
INEW=INT(XKA)
I2NUM(INEW)=I2NUM(INEW)+1
I2=I2NUM(INEW)
DO 25 J=1,I2
IF(K2NUM(INEW,J).EQ.KR) GO TO 35
CONTINUE
K2NUM(INEW,I2)=KR

```

IF(TKA.NE.0.) GO TO 81
FL2NUM(INEW,I2)=-FDUM
GO TO 82
FL2NUM(INEW,I2)=FDUM
KK=L+NUM

INPUT EDGE INEW INTO THE IDELDM(KR,.) MATRIX

DO 45 K=1, KK
IF(IDELDM(KR,K).EQ.INEW) GO TO 40
IDELDM(KR, KK)=INEW

CONTINUE

GO TO 40

I2NUM(INEW)=I2NUM(INEW)-1

IF(TKA.NE.0.) GO TO 83

FL2NUM(INEW,J)=FL2NUM(INEW,J)-FDUM

GO TO 40

FL2NUM(INEW,J)=FL2NUM(INEW,J)+FDUM

CONTINUE

NSUM=NDUM(KR)

DO 22 KD=1, NSUM

IF(IDELDM(KR, KD).NE.0) GO TO 22

NDUM(KR)=NDUM(KR)-1

CONTINUE

II=0

DO 52 KI=1, NSUM

IF(IDELDM(KR, KI).NE.0) GO TO 26

IDELDM(KR, KI)=IDELDM(KR, KI+1)

IDELDM(KR, KI+1)=0

GO TO 52

II=II+1

CONTINUE

IF(II.NE.NDUM(KR)) GO TO 51

=====

CHECK FOR TERMINATION

=====

IF(VALUE.GT.CCL(KR)) GO TO 21

IF(VALUE.LT.CCL(KR)) GO TO 8

GO TO 11

VALUE=VDUM

DO 12 L=1, LM

KL=IDEL2(KR, L)

XMMC(KL)=XMMC(KL)-VALUE

IF(KL/2*2.EQ.KL) GO TO 37

IX=IT*IN(KL)

XMMC(IX)=XMMC(IX)-VALUE

IR=IDASH(KL)

XMMC(IR)=XMMC(IR)+VALUE

IF(IR/2*2.EQ.IR) GO TO 12

IW=IT*IN(IR)

XMMC(IW)=XMMC(IW)+VALUE

CONTINUE

WRITE(MN, 9) CCL(KR), (J, XMMC(J), J=1, NK)

FORMAT(///, 15X, 'THE GIVEN FLOW VALUE=', F5.0, ///, 15X, 'FLOWS ON ARC
&S', //, 11(8X, 8(I5, '==>', F6.0), //))

```
GO TO 23
KS=NDUM(KK)
DO 24 M=1,KS
IDELDM(KR,M)=0
CONTINUE
RETURN
END
```

SUBROUTINE REMOVE(XMD)

CONVERSION OF THE NETWORK WITH NEGATIVE COSTS TO AN EQUIVALENT
NETWORK HAVING NON-NEGATIVE COSTS

COMMON /AAA/NN, MG, LLL, MON(300), MTN(300), REMCAP(300), ITWIN(300),
&IDASH(300), ICONCH

COMMON /ZZZ/INODP2(100), NOEDG2(300), IDEL2(150,20), RESULT
DIMENSION XMD(300), CBAR(100)

MW=6
NK=MG
NK=NK*4
RESULT=0.

=====
IF THE COSTS ARE ALREADY NON-NEGATIVE, RETURN
=====

DO 3 J=1, NK
IF(XMD(J).LT.0.) GO TO 11
CONTINUE
GO TO 10

START THE ALGORITHM

IT=1
ICT=0
DO 1 I=1, NN

AMONG THE EDGES IN THE FORWARD STAR OF NODE I, CHOOSE THE ONE
WITH THE MINIMUM COST

CBAR(I)=99999999.
LBEG=INODP2(I)
LEND=INODP2(I+1)-1
DO 2 L=LBEG, LEND
KE=NOEDG2(L)
IF(XMD(KE).GE.CBAR(I)) GO TO 2
CBAR(I)=XMD(KE)

CONTINUE
IF(CBAR(I).GE.0.) GO TO 6

UPDATE THE COSTS OF THE EDGES

DO 4 IL=LBEG, LEND
KE=NOEDG2(IL)
MOR=IDASH(KE)
XMD(KE)=XMD(KE)-CBAR(I)
XMD(MOR)=XMD(MOR)+CBAR(I)

CONTINUE
GO TO 1
ICT=ICT+1
CONTINUE

=====
CHECK FOR TERMINATION
=====

IF(ICT.EQ.NN) GO TO 10

```
IF(IT.GE.(NN+1)) GO TO 7  
IT=IT+1  
GO TO 8
```

```
=====
```

```
CHECK FOR NEGATIVE CIRCUIT
```

```
=====
```

```
IF(IT.NE.(NN+1)) GO TO 10  
WRITE(MW,9)  
FORMAT(///,12X,'** THERE IS A NEGATIVE CIRCUIT IN THE NETWORK **')  
RESULT=1.  
RETURN  
END
```

```

*****
MAIN PROGRAM
*****
COMMON /AAA/NN, MG, LLL, MON(300), MTN(300), REMCAP(300), ITWIN(300),
&IDASH(300), ICONCH
COMMON /XXX/CON(150), CTN(150), CCL(150), LCN
COMMON /XYZ/INODP(100), NOEDGE(300), IDEL1(150,20)
COMMON /ZZZ/INODP2(100), NOEDG2(300), IDEL2(150,20), RESULT
COMMON /EEE/NDUM(150), IDELDM(150,20)
COMMON /YYY/FCOST(3), UVC(3), CAP(3), LCIN(300), CURCAP(300), XFLW(300)
COMMON /IJK/STDMO(300), SUMFLW(300)
COMMON /VVV/V(300), KM, INUM(300), KNUM(100,20), FLNUM(100,20),
&KFLAG(300)
COMMON /TTT/IDE(150), I2NUM(300), K2NUM(100,20), FL2NUM(100,20)
DIMENSION CVT(3), MD(300), COST(300), COSTBR(300), COSTAR(300),
&NNN(150), STJRE(300), STOFLW(300), IDUM(300), KDUM(100,20),
&FLDUM(100,20), STOCAP(300), STOCIN(300), SBR(300), EXCAP(300),
&ISYS(300,20), INDEX(300), STODEX(300), ISTDS(300,20)
INTEGER CON, CTN
REAL MD
NR=5
NW=6
READ(MR,1) LLL
1 FORMAT(I2)
READ(MR,2) MG, NN
2 FORMAT(2(2X, I4))
WRITE(MW,27) MG, NN
7 FORMAT(///, 15X, 'MG=', I4, /, 15X, 'NN=', I4)
READ(MR,3) LCN
3 FORMAT(2X, I4)
NK=MG
NKR=LCN
WRITE(MW,4)
4 FORMAT(///, 15X, 'MON', 5X, 'MTN', 5X, 'MD', 5X, 'EXCAP', /)
DO 5 K=1, NK
READ(MR,6) MON(K), MTN(K), MD(K), EXCAP(K)
6 FORMAT(2I4, 2(2X, F5.0))
WRITE(MW,7) MON(K), MTN(K), MD(K), EXCAP(K)
7 FORMAT(14X, I4, 4X, I4, 2X, F5.0, 5X, F5.0)
5 CONTINUE
WRITE(MW,8)
8 FORMAT(///, 15X, 'CON', 5X, 'CTN', 5X, 'CCL', /)
DO 9 IC=1, NKR
READ(MR,10) CON(IC), CTN(IC), CCL(IC)
10 FORMAT(2(2X, I4), 1X, F6.0)
9 CONTINUE
DO 755 I=1, NK
DO 756 J=1, 20
ISYS(I, J)=0
6 CONTINUE
5 CONTINUE
=====
SORT THE CIRCUIT DEMANDS IN DESCENDING ORDER
=====
CALL SORT

```

```

CAP FOR A VM/SP.TUBITAK
DO 59 IC=1,NKR
WRITE(MW,11) CON(IC),CTN(IC),CCL(IC)
FORMAT(14X,I4,4X,I4,4X,F4.0)
CONTINUE
WRITE(MW,86)
FORMAT(///,15X,'SYSTEM',5X,'FCOST',5X,'UVC',12X,'CAP',/)
DO 22 I=1,3
READ(MR,89) FCOST(I),UVC(I),CAP(I)
FORMAT(2X,F8.0,2X,F9.1,1X,F9.0)
WRITE(MW,99) I,FCOST(I),UVC(I),CAP(I)
FORMAT(17X,I1,5X,F8.0,2X,F6.1,5X,F10.0)
CONTINUE
=====
FORM THE FORWARD STAR REPRESENTATION OF THE NETWORK
=====
CALL FSTAR
WRITE(MW,190)
FORMAT(4(/),12X,'++ FSTAR OF THE NETWORK ++')
WRITE(MW,29) (INODP(I),I=1,NN)
FORMAT(///,15X,'INODP(I)',/,9X,10I4)
NK2=NK*2
WRITE(MW,49) (NOEDGE(J),J=1,NK2)
FORMAT(///,15X,'NOEDGE(J)',5(/,9X,20I4))
=====
CALCULATE THE SYSTEM COSTS
=====
DO 51 J=1,3
CVT(J)=FCOST(J)+UVC(J)*CAP(J)
CONTINUE
=====
INITIALIZE THE INFORMATION ABOUT THE EDGES
=====
DO 50 K=1,NK
XFLW(K)=0.
IF(EXCAP(K).EQ.0.) GO TO 350
CURCAP(K)=EXCAP(K)
IF(EXCAP(K).GT.CAP(1)) GO TO 352
INDEX(K)=1
ISYS(K,1)=1
COST(K)=UVC(1)*MD(K)
GO TO 351
IF(EXCAP(K).GT.CAP(2)) GO TO 353
INDEX(K)=1
ISYS(K,1)=2
COST(K)=UVC(2)*MD(K)
GO TO 351
INDEX(K)=1
ISYS(K,1)=3
COST(K)=UVC(3)*MD(K)
GO TO 351
CURCAP(K)=CAP(1)
LCIN(K)=1
COST(K)=CVT(1)*MD(K)/CAP(1)
INUM(K)=0
IDUM(K)=0

```

```

CONTINUE
DO 100 I=1,100
DO 101 J=1,20
KNUM(I,J)=0
FLNUM(I,J)=0.
KDUM(I,J)=0
FLDUM(I,J)=0.
CONTINUE
CONTINUE

```

=====

CALCULATE THE INITIAL FLOWS ON THE EDGES

=====

```

DO 12 KR=1,NKR
WRITE(MW,15) KR,CON(KR),CTH(KR)
FORMAT(////,15X,'THE RESULTS FOR RELATION',I4,1X,'(FROM NODE',I3,
&' TO NODE',I3,')',/,14X,56(' '*'))

```

FIND THE SHORTEST PATH OF RELATION KR

```

CALL SHPTH(KR,IELN,COST)
IF(ICONCH.EQ.1) GO TO 12
WRITE(MW,25) (IDEL1(KR,L),L=1,IELN)
FORMAT(///,15X,'IDEL1 ENTRIES',/,14X,10I4)

```

ASSIGN THE FLOW CCL(KR) TO THE EDGES ON THE SHORTEST PATH OF RELATION KR

```

NNN(KR)=IELN
DO 52 L=1,IELN
K=IDEL1(KR,L)
INUM(K)=INUM(K)+1
IL=INUM(K)
KNUM(K,IL)=KR
FLNUM(K,IL)=CCL(KR)
XFLW(K)=XFLW(K)+CCL(KR)
IF(XFLW(K).LE.CURCAP(K)) GO TO 52

```

UPDATE THE SYSTEMS ON THE EDGES

```

IF(EXCAP(K).EQ.0.) GO TO 33
DIFF=XFLW(K)-CURCAP(K)
IF(DIFF.LE.0.) GO TO 52
IF(DIFF.GT.CAP(1)) GO TO 952
CURCAP(K)=CURCAP(K)+CAP(1)
INDEX(K)=INDEX(K)+1
IN=INDEX(K)
ISYS(K,IN)=1
COST(K)=CVT(1)*MD(K)/CAP(1)
GO TO 52
IF(DIFF.GT.CAP(2)) GO TO 953
CURCAP(K)=CURCAP(K)+CAP(2)
INDEX(K)=INDEX(K)+1
IN=INDEX(K)
ISYS(K,IN)=2
COST(K)=(CVT(2)-CVT(1))*MD(K)/(CAP(2)-CAP(1))

```



```

GO TO 52
3 CURCAP(K)=CURCAP(K)+CAP(3)
INDEX(K)=INDEX(K)+1
IN=INDEX(K)
ISYS(K,1N)=3
COST(K)=(CVT(3)-CVT(2))*MD(K)/(CAP(3)-CAP(2))
GO TO 52
3 LCIN(K)=LCIN(K)+1
LS=LCIN(K)
CURCAP(K)=CAP(LS)
IF(XFLW(K).GT.CURCAP(K)) GO TO 33

```

UPDATE THE COSTS THAT ARE USED IN FINDING THE SHORTEST PATHS

```

COST(K)=(CVT(LS)-CVT(LS-1))*MD(K)/(CAP(LS)-CAP(LS-1))
2 CONTINUE
WRITE(MW,55)
5 FORMAT(///,15X,'EDGE',8X,'XFLW',8X,'EXCAP',8X,'CURCAP',8X,'COST',
&/)
DO 53 J=1,NK
WRITE(MW,54) J,XFLW(J),EXCAP(J),CURCAP(J),COST(J)
4 FORMAT(14X,14,5X,F8.0,8X,F5.0,4X,F10.0,4X,F3.0)
3 CONTINUE
2 CONTINUE
DO 779 II=1,NK
IF(EXCAP(II).NE.0.) GO TO 779
IF(XFLW(II).EQ.0.) GO TO 779
LL=LCIN(II)
ISYS(II,2)=LL
9 CONTINUE
WRITE(MW,448)
WRITE(MW,449) ((ISYS(K,IL),IL=1,10),K=1,NK)
WRITE(MW,36)
6 FORMAT(////,15X,'KNUM MATRIX',/)
WRITE(MW,23) ((KNUM(K,IL),IL=1,20),K=1,NK)
3 FORMAT(9X,20I4)
WRITE(MW,39)
9 FORMAT(////,15X,'FLNUM MATRIX',/)
WRITE(MW,24) ((FLNUM(K,IL),IL=1,20),K=1,NK)
4 FORMAT(9X,20F4.0)
DO 38 I=1,NK
STORE(I)=COST(I)
STOFLW(I)=XFLW(I)
STOCAP(I)=CURCAP(I)
STOCIN(I)=LCIN(I)
STODEX(I)=INDEX(I)
8 CONTINUE
DO 1000 I=1,NK
DO 1001 J=1,20
ISTDS(I,J)=ISYS(I,J)
1 CONTINUE
0 CONTINUE

```

CALCULATE THE REAL INITIAL COSTS OF THE EDGES AND
THE INITIAL TOTAL COST OF THE NETWORK

```

=====
WRITE(MW,70)
FORMAT(5(/),15X,'INITIAL FLOWS ON EDGES',//,15X,'EDGE',8X,'XFLW',
&8X,'EXCAP',8X,'COSTBR',/)
DO 71 L=1,NK
LS=LCIN(L)
IF(XFLW(L).NE.0.) GO TO 300
COSTBR(L)=0.
GO TO 301
IF(EXCAP(L).EQ.0.) GO TO 955
IF(XFLW(L).GT.EXCAP(L)) GO TO 355
IZ=ISYS(L,1)
COSTBR(L)=UVC(IZ)*XFLW(L)*MD(L)
GO TO 301
COSTBR(L)=0.
IC=ISYS(L,1)
ID=INDEX(L)
DO 444 JJ=2,ID
IL=ISYS(L,JJ)
COSTBR(L)=COSTBR(L)+(FCOST(IL)+UVC(IL)*CAP(IL))*MD(L)
CONTINUE
CUSTBR(L)=COSTBR(L)+(UVC(IC)*CAP(IC)*MD(L))
DIF1=CURCAP(L)-EXCAP(L)
DIF2=DIF1-(XFLW(L)-EXCAP(L))
IF(DIF2.ST.CAP(1)) GO TO 666
COSTBR(L)=COSTBR(L)-(UVC(1)*DIF2*MD(L))
GO TO 301
IF(DIF2.GT.CAP(2)) GO TO 667
CUSTBR(L)=COSTBR(L)-(UVC(2)*DIF2*MD(L))
GO TO 301
CUSTBR(L)=COSTBR(L)-(UVC(3)*DIF2*MD(L))
GO TO 301
COSTBR(L)=(FCOST(LS)+UVC(LS)*XFLW(L))*MD(L)
WRITE(MW,72) L,XFLW(L),EXCAP(L),COSTBR(L)
FORMAT(14X,14,5X,F8.0,8X,F5.0,3X,F11.0)
CONTINUE
TOTC=0.
DO 73 K=1,NK
TOTC=TOTC+COSTBR(K)
CONTINUE
DO 900 K=1,NK
SBR(K)=COSTBR(K)
CONTINUE
WRITE(MW,74) TOTC
FORMAT(///,15X,'INITIAL TOTAL COST=',F11.0)
=====
PREPARE THE NETWORK FOR THE B&G ALGORITHM BY FORMING ITS
FORWARD STAR REPRESENTATION
=====
CALL STAR2
=====
INITIALIZE THE FLAGS OF THE EDGES--(FIX THE EDGES WITH EXISTING
CAPACITY)

```

KFLAG DENOTES THE FLAG OF THE EDGE

```

      -1      IF THE EDGE IS NOT YET PROCESSED
      0       IF THE EDGE IS DELETED
      1       IF THE EDGE IS FIXED
=====
DO 16 K=1,NK
IF(EXCAP(K).EQ.0.) GO TO 957
KFLAG(K)=1
GO TO 16
KFLAG(K)=-1
CONTINUE
ITT=0
*****
START THE ALGORITHM
*****
WRITE(MW,160)
FORMAT(10(/),15X,'CURRENT FLOWS ON EDGES',//,15X,'EDGE',8X,'XFLW',
&/)
DO 162 J=1,NK
WRITE(MW,161) J,XFLW(J)
FORMAT(14X,I4,5X,F8.0)
CONTINUE
DO 102 I=1,NK
DO 103 J=1,20
KDUM(I,J)=KNUM(I,J)
FLDUM(I,J)=FLNUM(I,J)
CONTINUE
IDUM(I)=INUM(I)
CONTINUE
NO=0
=====
CALCULATE THE V(K) VALUES
=====
CALL VCAL(MD)
WRITE(MW,170) (KFLAG(J),J=1,NK)
FORMAT(///,15X,'FLAGS OF EDGES',/,9X,20I4)
=====
CHECK FOR TERMINATION
=====
DO 83 J=1,NK
IF(KFLAG(J).NE.-1) GO TO 83
IF(V(J).NE.555555555.) GO TO 46
CONTINUE
=====
PRINT THE FINAL FLOWS AND THE FINAL NETWORK COST
=====
WRITE(MW,191)
FORMAT(4(/),12X,'** NO FURTHER IMPROVEMENT CAN BE MADE, THEREFORE
&STOP **')
WRITE(MW,448)
WRITE(MW,449) ((ISYS(K,IL),IL=1,10),K=1,NK)
IF(TCOST.EQ.0..OR.TCOST.GE.TOTC) GO TO 119
WRITE(MW,105)
FORMAT(4(/),15X,'FINAL FLOWS ON EDGES',//,15X,'EDGE',8X,'XFLW',8X,
&'EXCAP',8X,'REDCOST',/)
DO 106 L=1,NK

```

```

WRITE(MW,107) L,XFLW(L),EXCAP(L),SBR(L)
07 FORMAT(14X,I4,5X,F8.0,8X,F5.0,3X,F11.0)
06 CONTINUE
WRITE(MW,104) TCOST
04 FORMAT(///,15X,'FINAL NETWORK COST=',F11.0)
GO TO 91
09 WRITE(MW,120)
20 FORMAT(4(/),15X,'FINAL FLOWS ON EDGES',//,15X,'EDGE',8X,'XFLW',8X,
&'EXCAP',8X,'RECOST',/)
DO 121 L=1,NK
WRITE(MW,122) L,XFLW(L),EXCAP(L),SBR(L)
22 FORMAT(14X,I4,5X,F8.0,8X,F5.0,3X,F11.0)
21 CONTINUE
WRITE(MW,123) TOTC
23 FORMAT(///,15X,'FINAL NETWORK COST=',F11.0)
GO TO 91
=====
INCREASE THE NUMBER OF ITERATIONS BY ONE
=====
06 ITT=ITT+1
WRITE(MW,192) ITT
92 FORMAT(///,15X,'NO OF ITERATIONS=',I4)
WRITE(MW,36)
WRITE(MW,23) ((KNUM(K,IL),IL=1,20),K=1,NK)
WRITE(MW,39)
WRITE(MW,24) ((FLNUM(K,IL),IL=1,20),K=1,NK)
WRITE(MW,108)
08 FORMAT(////,15X,'CURRENT IDEL1 MATRIX',/)
WRITE(MW,109) ((IDEL1(I,J),J=1,20),I=1,NKR)
09 FORMAT(9X,20I4)
WRITE(MW,223) (INUM(K),K=1,NK)
23 FORMAT(////,15X,'INUM VALUES',/,9X,20I4)
-----
KM IS THE CANDIDATE EDGE
-----
IW=INUM(KM)
DO 60 IL=1,IW
-----
TAKE THE RELATIONS USING EDGE KM ONE BY ONE
-----
LR=KNUM(KM,IL)
IELN=NNN(LR)
DO 61 IE=1,IELN
-----
TAKE THE EDGES USING RELATION LR ONE BY ONE
-----
K1=IDEL1(LR,IE)
JD=INUM(K1)
DO 113 J=1,JD
IF(KNUM(K1,J).NE.LR) GO TO 113
IQ=J
GO TO 114
13 CONTINUE
-----
DECREASE THE FLOW ON THESE EDGES BY THE AMOUNT FLW ( OF LR )

```

```

-----
4 FLW=FLNUM(K1,IQ)
  XFLW(K1)=ABS(XFLW(K1)-FLW)
1 CONTINUE
0 CONTINUE

```

UPDATE THE RELATED ARRAYS

```

IW=INUM(KM)
DO 84 I=1,IW
  LR=KNUM(KM,I)
  DO 80 K=1,NK
    MC=INUM(K)
    DO 82 J=1,MC
      IF(KDUM(K,J).NE.LR) GO TO 82
      IDUM(K)=IDUM(K)-1
      KDUM(K,J)=0
      FLDUM(K,J)=0.
82 CONTINUE
80 CONTINUE
84 CONTINUE
WRITE(MW,193) (IDUM(K),K=1,NK)
93 FORMAT(///,15X,'IDUM VALUES',/,9X,20I4)
WRITE(MW,193)
93 FORMAT(4(/),15X,'DECREASED FLOW VALUES',/,15X,'EDGE',8X,'XFLW',
&/)
DO 194 J=1,NK
  WRITE(MW,195) J,XFLW(J)
95 FORMAT(14X,I4,5X,F8.0)
94 CONTINUE
DO 81 K=1,NK
  IF(K.EQ.KM) GO TO 81
  JT=INUM(K)
80 I=0
DO 85 J=1,JT
  IF(KDUM(K,J).NE.0) GO TO 131
  KDUM(K,J)=KDUM(K,J+1)
  KDUM(K,J+1)=0
  FLDUM(K,J)=FLDUM(K,J+1)
  FLDUM(K,J+1)=0.
GO TO 85
81 I=I+1
85 CONTINUE
IF(I.NE.IDUM(K)) GO TO 130
81 CONTINUE

```

CALCULATE THE REMAINING CAPACITIES ON THE EDGES BEFORE ENTERING
THE B&G ALGORITHM FOR REROUTING

```

DO 30 K=1,NK
  IF(K.EQ.KM) GO TO 31
  IF(EXCAP(K).EQ.0.) GO TO 958
  IF(XFLW(K).GT.EXCAP(K)) GO TO 959
  IW=INDEX(K)
DO 333 IY=2,IW

```

```

ISYS(K,1Y)=0
INDEX(K)=INDEX(K)-1
33 CONTINUE
REMCAP(K)=EXCAP(K)-XFLW(K)
CURCAP(K)=EXCAP(K)
IS=ISYS(K,1)
COST(K)=UVC(IS)*MD(K)
GO TO 30
59 DIF=XFLW(K)-EXCAP(K)
IF(DIF.GT.CAP(1)) GO TO 300
REMCAP(K)=CAP(1)-DIF
CURCAP(K)=EXCAP(K)+CAP(1)
COST(K)=CVT(1)*MD(K)/CAP(1)
GO TO 30
80 IF(DIF.GT.CAP(2)) GO TO 301
REMCAP(K)=CAP(2)-DIF
CURCAP(K)=EXCAP(K)+CAP(2)
COST(K)=(CVT(2)-CVT(1))*MD(K)/(CAP(2)-CAP(1))
GO TO 30
81 REMCAP(K)=CAP(3)-DIF
CURCAP(K)=EXCAP(K)+CAP(3)
COST(K)=(CVT(3)-CVT(2))*MD(K)/(CAP(3)-CAP(2))
GO TO 30
58 IF(XFLW(K).GT.CAP(1)) GO TO 401
REMCAP(K)=CAP(1)-XFLW(K)
IF(REMCAP(K).EQ.0.) GO TO 405
CURCAP(K)=CAP(1)
LCIN(K)=1
COST(K)=CVT(1)*MD(K)/CAP(1)
GO TO 30
05 CURCAP(K)=CAP(2)
LCIN(K)=2
REMCAP(K)=CAP(2)-XFLW(K)
COST(K)=(CVT(2)-CVT(1))*MD(K)/(CAP(2)-CAP(1))
GO TO 30
81 IF(XFLW(K).GT.CAP(2)) GO TO 402
REMCAP(K)=CAP(2)-XFLW(K)
IF(REMCAP(K).EQ.0.) GO TO 406
CURCAP(K)=CAP(2)
LCIN(K)=2
COST(K)=(CVT(2)-CVT(1))*MD(K)/(CAP(2)-CAP(1))
GO TO 30
06 CURCAP(K)=CAP(3)
LCIN(K)=3
REMCAP(K)=CAP(3)-XFLW(K)
COST(K)=(CVT(3)-CVT(2))*MD(K)/(CAP(3)-CAP(2))
GO TO 30
02 CURCAP(K)=CAP(3)
LCIN(K)=3
REMCAP(K)=CAP(3)-XFLW(K)
COST(K)=(CVT(3)-CVT(2))*MD(K)/(CAP(3)-CAP(2))
GO TO 30
31 LS=LCIN(K)
IF(LS.EQ.1) GO TO 500
REMCAP(K)=CAP(LS-1)

```

```

IF(LS.EQ.2) GO TO 501
COST(K)=(CVT(2)-CVT(1))*MD(K)/(CAP(2)-CAP(1))
GO TO 30
1 COST(K)=CVT(1)*MD(K)/CAP(1)
GO TO 30
2 REMCAP(K)=0.
COST(K)=999999999.
3 CONTINUE
DO 930 J=1,NK
IF(KFLAG(J).NE.0) GO TO 930
CUST(J)=999999999.
4 CONTINUE
DO 2222 I=1,NK
INDEX(I)=0
DO 2223 J=2,20
ISYS(I,J)=0
5 CONTINUE
2 CONTINUE
DO 2224 I=1,NK
IF(EXCAP(I).EQ.0.) GO TO 2225
IF(EXCAP(I).GT.XFLW(I)) GO TO 2224
IF(XFLW(I).EQ.0.) GO TO 2224
IF(REMCAP(I).GT.CAP(1)) GO TO 2226
INDEX(I)=2
ISYS(I,2)=1
GO TO 2224
6 IF(REMCAP(I).GT.CAP(2)) GO TO 2227
INDEX(I)=2
ISYS(I,2)=2
GO TO 2224
7 INDEX(I)=2
ISYS(I,2)=3
GO TO 2224
5 IF(REMCAP(I).GT.CAP(1)) GO TO 2228
INDEX(I)=2
ISYS(I,2)=1
GO TO 2224
8 IF(REMCAP(I).GT.CAP(2)) GO TO 2239
INDEX(I)=2
ISYS(I,2)=2
GO TO 2224
9 INDEX(I)=2
ISYS(I,2)=3
4 CONTINUE
IF(REMCAP(KM).NE.0.) GO TO 456
DO 433 J=1,20
ISYS(KM,J)=0
3 CONTINUE
INDEX(KM)=0
6 WRITE(MW,448)
8 FORMAT(///,15X,'SYSTEMS ON EDGES',/)
WRITE(MW,449) ((ISYS(K,IL),IL=1,10),K=1,NK)
9 FORMAT(9X,10I4)
=====
REROUTE THE CIRCUIT DEMAND OF THE RELATIONS USING EDGE KM

```

```
=====
CALL M2(CJST)
IF(RESULT.EQ.1.) GO TO 91
```

```
-----
IS A FEASIBLE FLOW PATTERN OBTAINED ? IF NO, FIX THE SYSTEM
ON EDGE KM
```

```
-----
IF(ICONCH.EQ.1) GO TO 65
WRITE(MW,21)
```

```
21 FORMAT(////,15X,'K2NUM MATRIX',/)
WRITE(MW,23) ((K2NUM(K,IL),IL=1,20),K=1,NK)
```

```
28 FORMAT(9X,20I4)
WRITE(MW,35)
```

```
35 FORMAT(////,15X,'FL2NUM MATRIX',/)
WRITE(MW,37) ((FL2NUM(K,IL),IL=1,20),K=1,NK)
```

```
37 FORMAT(9X,20F4.0)
WRITE(MW,164) (I2NUM(K),K=1,NK)
```

```
164 FORMAT(////,15X,'I2NUM VALUES',/,9X,20I4)
WRITE(MW,165)
```

```
165 FORMAT(6(//),15X,'FLOWS OBTAINED AFTER APPLYING THE BAG ALGORITHM',
& //,15X,'EDGE',8X,'SUMFLW',/)
```

```
DO 166 KI=1,MG
```

```
WRITE(MW,167) KI,SUMFLW(KI)
```

```
167 FORMAT(14X,I4,7X,F8.0)
```

```
166 CONTINUE
```

```
-----
CALCULATE THE FLOWS AFTER REROUTING.
```

```
DO 66 I=1,MG
```

```
SUMFLW(I)=SUMFLW(I)+XFLW(I)
```

```
66 CONTINUE
```

```
WRITE(MW,41)
```

```
41 FORMAT(5(//),15X,'FLOWS AFTER REROUTING',//,15X,'EDGE',8X,'XFLW',
& 8X,'COSTAR',/)
```

```
-----
UPDATE THE SYSTEMS ON THE EDGES
```

```
DO 40 K=1,MG
```

```
IF(EXCAP(K).EQ.0.) GO TO 952
```

```
IF(SUMFLW(K).LE.CURCAP(K)) GO TO 3020
```

```
DIFR=SUMFLW(K)-CURCAP(K)
```

```
IF(DIFR.GT.CAP(1)) GO TO 963
```

```
INDEX(K)=INDEX(K)+1
```

```
IN=INDEX(K)
```

```
ISYS(K,IN)=1
```

```
CURCAP(K)=CURCAP(K)+CAP(1)
```

```
COSTAR(K)=0.
```

```
DO 3021 LL=2,IN
```

```
IH=ISYS(K,LL)
```

```
COSTAR(K)=COSTAR(K)+(FCOST(IH)+UVC(IH)*CAP(IH))*MD(K)
```

```
3021 CONTINUE
```

```
KN=ISYS(K,1)
```

```
COSTAR(K)=COSTAR(K)+(UVC(KN)*DIFR*MD(K))
```

```
COST(K)=CVT(1)*MD(K)/CAP(1)
```

```
GO TO 303
```



```

3 IF(DIFR.GT.CAP(2)) GO TO 964
INDEX(K)=INDEX(K)+1
IN=INDEX(K)
ISYS(K,IN)=2
CURCAP(K)=CURCAP(K)+CAP(2)
COSTAR(K)=0.
DO 3022 LL=2,IN
IH=ISYS(K,LL)
COSTAR(K)=COSTAR(K)+(FCOST(IH)+UVC(IH)*CAP(IH))*MD(K)
22 CONTINUE
KN=ISYS(K,1)
COSTAR(K)=COSTAR(K)+(UVC(KN)*DIFR*MD(K))
COST(K)=(CVT(2)-CVT(1))*MD(K)/(CAP(2)-CAP(1))
GO TO 303
34 INDEX(K)=INDEX(K)+1
IN=INDEX(K)
ISYS(K,IN)=3
CURCAP(K)=CURCAP(K)+CAP(3)
COSTAR(K)=0.
DO 3023 LL=2,IN
IH=ISYS(K,LL)
COSTAR(K)=COSTAR(K)+(FCOST(IH)+UVC(IH)*CAP(IH))*MD(K)
23 CONTINUE
KN=ISYS(K,1)
COSTAR(K)=COSTAR(K)+(UVC(KN)*DIFR*MD(K))
COST(K)=(CVT(3)-CVT(2))*MD(K)/(CAP(3)-CAP(2))
GO TO 303
20 COSTAR(K)=0.
DR=SUMFLW(K)-EXCAP(K)
IF(DR.GT.0.) GO TO 4001
IP=ISYS(K,1)
COSTAR(K)=UVC(IP)*SUMFLW(K)*MD(K)
COST(K)=UVC(IP)*MD(K)
GO TO 303
01 IE=ISYS(K,1)
IS=INDEX(K)
DO 2229 IV=2,IS
KW=ISYS(K,IV)
COSTAR(K)=COSTAR(K)+(FCOST(KW)+UVC(KW)*CAP(KW))*MD(K)
29 CONTINUE
COSTAR(K)=COSTAR(K)+(UVC(IE)*CAP(IE)*MD(K))
D1=CURCAP(K)-EXCAP(K)
D2=D1-(SUMFLW(K)-EXCAP(K))
KWW=ISYS(K,IS)
COSTAR(K)=COSTAR(K)-(UVC(KWW)*D2*MD(K))
GO TO 303
52 IF(SUMFLW(K).GT.CAP(1)) GO TO 43
LCIN(K)=1
CURCAP(K)=CAP(1)
GO TO 44
43 IF(SUMFLW(K).GT.CAP(2)) GO TO 45
LCIN(K)=2
CURCAP(K)=CAP(2)
GO TO 44
45 LCIN(K)=3

```

CURCAP(K)=CAP(3)

 CALCULATE THE REAL COSTS OF THE EDGES AFTER REROUTING

```

44 LS=LCIN(K)
   IF(SUMFLW(K).NE.0.) GO TO 302
   COSTAR(K)=0.
   GO TO 303
302 COSTAR(K)=(FCOST(LS)+UVC(LS)*SUMFLW(K))*MD(K)
303 WRITE(MW,42) K,SUMFLW(K),COSTAR(K)
42  FORMAT(14X,14,5X,F8.0,3X,F11.0)
   IF(EXCAP(K).NE.0.) GO TO 40
   IF(KFLAG(K).NE.0) GO TO 193
   COST(K)=999999999.
   GO TO 40
    
```

 UPDATE THE COSTS THAT ARE USED IN FINDING THE SHORTEST PATHS

```

198 IF(LS.EQ.1) GO TO 197
   COST(K)=(CVT(LS)-CVT(LS-1))*MD(K)/(CAP(LS)-CAP(LS-1))
   GO TO 40
197 COST(K)=CVT(1)*MD(K)/CAP(1)
40  CONTINUE
   IF(SUMFLW(KM).NE.0.) GO TO 503
   COST(KM)=999999999.
    
```

 CALCULATE THE TOTAL NETWORK COST AFTER REROUTING

```

503 TCOST=0.
   DO 56 K=1,MG
   TCOST=TCOST+COSTAR(K)
56  CONTINUE
   WRITE(MW,26) TCOST
26  FORMAT(///,15X,'TOTAL COST AFTER REROUTING=',F11.0)
   WRITE(MW,448)
   WRITE(MW,449) ((ISYS(K,IL),IL=1,10),K=1,NK)
    
```

=====

COST COMPARISON

=====

IF(TCOST.LT.TOTC) GO TO 115

NO=1

GO TO 65

=====

EDGE KM CAN BE DELETED

=====

```

115 DO 48 K=1,MG
   XFLW(K)=SUMFLW(K)
   STDFLW(K)=XFLW(K)
   STDR(K)=COST(K)
   STDCAP(K)=CURCAP(K)
   STDCIN(K)=LCIN(K)
   SBR(K)=COSTAR(K)
   STDDEX(K)=INDEX(K)
48  CONTINUE
   DO 1002 I=1,NG
    
```

```

DU 1003 J=1,20
ISTUS(I,J)=ISYS(I,J)
1003 CONTINUE
1002 CONTINUE

```

UPDATE THE RELATED ARRAYS

```

DU 20 K=1,4G
JF=IDUM(K)
DU 38 J=1,10
IF(K.EQ.KM) GO TO 37
JJ=JF+J
KDUM(K,JJ)=K2NUM(K,J)
FLDUM(K,JJ)=FL2NUM(K,J)
GO TO 38
37 KDUM(K,J)=K2NUM(K,J)
FLDUM(K,J)=FL2NUM(K,J)
38 CONTINUE
20 CONTINUE

```

UPDATE THE IDELI MATRIX

```

IV=INUM(KM)
DU 110 L=1,IV
IR=KNUM(KM,L)
NC=NNN(IR)
DU 112 LL=1,NC
IDELI(IR,LL)=0
112 CONTINUE
JE=NDUM(IR)
NNN(IR)=JE
DU 111 N=1,JE
IDELI(IR,N)=IDELDI(IR,N)
111 CONTINUE
110 CONTINUE
DU 92 K=1,4G
DU 93 L=1,20
KNUM(K,L)=KDUM(K,L)
FLNUM(K,L)=FLDUM(K,L)
93 CONTINUE
INUM(K)=IDUM(K)+I2NUM(K)
92 CONTINUE

```

SINCE KM CAN BE DELETED, SET ITS FLAG TO 0

```
KFLAG(KM)=0
```

UPDATE THE TOTAL COST OF THE NETWORK

```

TOTC=0.
TOTC=TCOST
WRITE(MW,34) KM
34 FORMAT(///,12X,'** SINCE A FEASIBLE FLOW PATTERN HAS BEEN FOUND, A
&ND THE TOTAL COST AFTER REROUTING',/,15X,'IS LESS THAN THE PREVIOUS
&S COST, EDGE',I3,' CAN HAVE A SMALLER SYSTEM OR CAN BE DELETED **')

```

```
6)
GO TO 90
=====
EDGE KM CANNOT BE DELETED, THEREFORE FIX KM BY SETTING ITS
FLAG TO 1
=====
05 IF(NO.NE.1) GO TO 116
WRITE(MW,117) KM
17 FORMAT(///,12X,'** A FEASIBLE FLOW PATTERN HAVING A TOTAL COST GRE
ATER THAN THE PREVIOUS ONE',/,15X,'HAS BEEN OBTAINED, THEREFORE ED
GE',I3,' CANNOT HAVE A SMALLER SYSTEM OR CANNOT BE DELETED **')
16 KFLAG(KM)=1
DO 17 J=1,MG
COST(J)=STORE(J)
XFLW(J)=STOFLW(J)
CURCAP(J)=STOCAP(J)
LCIN(J)=STOLIN(J)
INDEX(J)=STODEX(J)
17 CONTINUE
DO 1004 I=1,MG
DO 1005 J=1,20
ISYS(I,J)=ISTOS(I,J)
05 CONTINUE
04 CONTINUE
GO TO 90
91 STOP
END
```

BIBLIOGRAPHY

1. COST Project 201, "Definitions and Terms", CNET, Paris, 1980.
2. COST Project 201, "Methods for Planning and Optimisation of Telecommunications Networks", Annual Progress Report, Part A Global Review, 1981/82, May 1982.
3. COST Project 201, "Methods for Planning and Optimisation of Telecommunications Network", Annual Progress Report, 1980/81, May 1981.
4. COST Project 201, "Methods for Planning and Optimisation of Telecommunications Networks", Annual Progress Report, Part B Technical Review, 1981/82, May 1982.
5. Evranuz, Ç., Ç. Mısırlı, A.İ. Dalgıç, T. Menlioğlu, "Telekomünikasyon Şebekelerinin Planlanması", A Paper presented at the Seventh National Operational Research Congress, İstanbul, September 1981.
6. Evranuz, Ç., Ç. Mısırlı, A.İ. Dalgıç, T. Menlioğlu, "COST 201: Telekomünikasyon Şebekelerinin Optimizasyonu ve Planlanması Projesi", Birinci Ara Rapor, Ekim 1981.
7. Mısırlı, Ç., "A Study on the Optimization of Switching Networks", Technical Report, Marmara Scientific and Industrial Research Institute, Operational Research Division, YA-82-03, January 1982.
8. Baybars, İ., K.O. Kortanek, "Transmission Facility Planning Models for Telecommunication Networks", Carnegie-Mellon University, Pittsburgh, USA, November 1981.
9. Evranuz, Ç., "A New Algorithm for Optimization of Transmission Network Structures", A Paper presented at the joint meeting of the Management Committee and Task Force of the COST Project 201, The Hague, 1982.
10. Evranuz, Ç., M. Miraboğlu, "Optimal Planning of Transmission Facilities for Telecommunications Networks", Technical Report, Marmara Scientific and Industrial Research Institute, Operational Research Division, YA-83-01, March 1983.

11. Evranuz, Ç., M. Miraboğlu, "Telekomünikasyon Şebekelerinde İletişim Sistemlerinin Eniyi Yatırım Planlaması", A Paper presented at the Eighth National Operational Research Congress, Ankara, June 1983.
12. Evranuz, Ç., M. Miraboğlu, "Telekomünikasyon Şebekelerinde İletişim Sistemlerinin Eniyi Yatırım Planlaması", Yöneylem Araştırması Dergisi, Yıl 2, Cilt 2, Aralık 1983.
13. Evranuz, Ç., T. Aktin, "Telekomünikasyon Şebekelerinin Serim Yapısı ile Devre Yönlendirilmesinin Eşzamanlı Eniyilenmesi", A Paper presented at the Eighth Operational Research Congress, Ankara, June 1983.
14. Nivert, K., N. Noort, "COST 201: A European Research Project: Methods for Planning and Optimization of Telecommunications Networks", A Paper presented at the ITC 10, Montreal, Canada, June 1983.
15. Cavellero, E., A. Tonietti, "Preliminary Description of Modules Grade of Service Evaluation and Determination of Stand-by Requirements in Transmission Network Optimization", CSELT International Report 81-08.245, September 1981.
16. Kaptanoğlu, D., Ç. Evranuz, "Telekomünikasyon Şebekelerinin Planlamasında Yedek Bulundurma Problemi", A Paper presented at the Eighth National Operational Research Congress, Ankara, June 1983.
17. Kaptanoğlu, D., "Determination of Stand-by Requirements in Telecommunication Networks", Technical Report, Marmara Scientific and Industrial Research Institute, Operational Research Division, 01 90 01 79 01, Nov. 1980.
18. Mina, R.R., "The Theory of Teletraffic Engineering", Part 1 of a Series on Telephone Traffic Engineering, April 1971, pp. 32-37.
19. Baybars, İ., K.O. Kortanek, "Transmission Facility Planning in Telecommunications Networks: A Heuristic Approach", European Journal of Operational Research, Vol. 16, 1984, pp. 59-83.
20. Flood, J.E., Telecommunications Networks, University of Aston, Birmingham, London, 1975.
21. Lindberg, P., U. Mocci, A. Tonietti, "COST 201: A European Research Project: A Procedure for Minimizing the Cost of a Transmission Network Under Service Availability Constraints in Failure Conditions", A paper presented at ITC 10, Montreal, Canada, June 1983.
22. Chen, H., C.G. De Wald, "A Generalized Chain Labelling Algorithm for Solving Multicommodity Flow Problems", Computers and Operations Research, Vol. 1, 1974, pp. 437-465.
23. Hartman, J.K., L.S. Lasdon, "A Generalized Upper Bounding Algorithm for Multicommodity Network Flow Problems", Networks, Vol. 1, pp. 333-354.

24. Ford, L.R., D.R. Fulkerson, Flows in Networks, Princeton University Press, Princeton, 1962.
25. Busacker, R.G., T.L. Saaty, Finite Graphs and Networks, McGraw-Hill, New York, 1965.
26. Klein, M., "A Primal Method for Minimal Cost Flows with Applications to the Assignment and Transportation Problem", Management Science, Vol. 14, No. 3, November 1967, pp. 205-220.
27. Bennington, G.E., "An Efficient Minimal Cost Flow Algorithm", Management Science, Vol. 19, No. 9, May 1973, pp. 1042-1051.
28. Tomlin, J.A., "Minimum Cost Multicommodity Network Flows", Operations Research, Vol. 14, 1966, pp. 45-51.
29. Yaged, Jr., B., "Economies of Scale, Networks, and Network Cost Elasticity", IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-5, No. 1, January 1975.
30. Luss, H., "Operations Research and Capacity Expansion Problems: A Survey", Operations Research, Vol. 30, No. 5, 1982, pp. 907-947.
31. Yaged, Jr., B., "Minimum Cost Routing for Static Network Models", Networks, Vol. 1, 1971, pp. 139-172.
32. Ulusoy, G., "Optimal Design of Routing Networks", Boğaziçi Üniversitesi Dergisi, Endüstri Mühendisliği, Vols. 8-9, No. 3100, 1981, pp. 183-199.
33. Claus, A., S. Krätzig, "Optimal Planning of Network Structures Within an Exchange Area", European Journal of Operational Research, Vol. 7, 1981, pp. 67-76.
34. Hackbarth, K.D., "A Heuristic Procedure for Calculating Telecommunication Transmission Networks in Consideration of the Network Reliability", A Paper presented at the International Workshop on Network Flow Optimization Theory and Practice, Pisa, March 1983.
35. Mc Callum, Jr., C.J., "A Generalized Upper Bounding Approach to a Communications Network Planning Problem", Networks, Vol. 7, 1977, pp. 1-23.
36. Claus, A., D.J. Kleitman, "Heuristic Methods for Solving Large Scale Network Routing Problems: The Telpaking Problem", Studies in Applied Mathematics, Vol. LIV, No. 1, March 1975, pp. 17-29.
37. Evranuz, Ç., "Network Structure Optimization of Transmission Networks", A Paper presented at the Joint Meeting of the Management Committee and Task Force of the COST Project 201, London, 1981.

38. Dial, R., F. Glover, D. Karney, D. Klingman, "A Computational Analysis of Alternative Algorithms and Labeling Techniques for Finding Shortest Path Trees", Networks, Vol. 9, 1979, pp. 215-248.
39. Gilsinn, J., C. Witzgall, "A Performance Comparison of Labeling Algorithms for Calculating Shortest Path Trees", NBS Technical Note 772, U.S. Department of Commerce, 1973.
40. Dreyfus, S., "An Appraisal of Some Shortest-Path Algorithms", Operations Research, Vol. 17, 1969, pp. 395-412.
41. Dantzig, G.B., "On the Shortest Route Through a Network", Management Science, Vol. 6, 1960, pp. 187-189.
42. Munro, J.I., R.J. Ramirez, "Reducing Space Requirements for Shortest Path Problems", Operations Research, Vol. 30, No. 5, 1982, pp. 1009-1013.
43. Glover, F., R. Glover, D. Klingman, "A Computer Study of Efficient Algorithms for Shortest Path Trees", Research Report CCS 399, Center for Cybernetic Studies, The University of Texas at Austin, June 1981.
44. Glover, F., R. Glover, D. Klingman, "Computational Study of an Improved Shortest Path Algorithm", Research Report CCS 419, Center for Cybernetic Studies, The University of Texas at Austin, June 1982.
45. Aktin, T., Ç. Evranuz, "En Kısa Yol Algoritmalarının Bilgisayara Uygulanış Yöntemleri ve İletişim Şebekelerinin Eniyilenmesinde Kullanılacak Bir Uyarılamanın Geliştirilmesi", A Paper presented at the Eighth National Operational Research Congress, Ankara, June 1983.
46. Aktin, T., Ç. Evranuz, "Shortest Path Algorithms and A Special Implementation for Capacitated Networks", Technical Report, Marmara Scientific and Industrial Research Institute, Operational Research Division, 01 90 01 79 01, 1983.
47. Hu, T.C., Integer Programming and Network Flows, Addison-Wesley, Reading MA, 1970.
48. Christofides, N., Graph Theory, Academic Press, London, 1975.
49. Bazaraa, M.S., J.J. Jarvis, Linear Programming and Network Flows, Wiley, New York, 1977.
50. Glover, F., D. Klingman, "Modeling and Solving Network Problems", A Paper presented at the NATO Advanced Study Institute on the Design and Implementation of Optimization Software, Urbino, Italy, 1977.