TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No. FHWA/TX-89/1153-6F	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle IMPLEMENTATION OF AN EQUILIBRIUM CAPACITY RESTRAINT MODEL IN THE TEXAS LARGE NETWORK ASSIGNMENT PACKAGE		5. Report Date November 1992 6. Performing Organization Code
7. Author(s) Jimmie D. Benson, George B. Dresser, Charles E. Bell, and J. Michael Heath		8. Performing Organization Report No. Research Report 1153-6F
9. Performing Organization Name and Address Texas Transportation Institute The Texas A&M University System College Station, Texas 77843-3135		10. Work Unit No. 11. Contract or Grant No. 2-10-89-1153
12. Sponsoring Agency Name and Address Texas Department of Transportation Division of Transportation Planning P.O. Box 5051 Austin, Texas 78763		13. Type of Report and Period Covered Final - September 1989 - August 1992 14. Sponsoring Agency Code

15. Supplementary Notes

Research performed in cooperation with the U.S. Department of Transportation, Federal Highway Administration.
Research Study Title: Improving the Efficiency, Effectiveness, and Responsiveness of the Traffic Assignment Process.

16. Abstract

One of the major objectives of Study 2-10-89-1153 has been to improve the accuracy of the assignment models implemented in the Texas Travel Demand Package. Three areas for improvements of the Texas capacity restraint model were identified: improved techniques for estimating iteration weights using an equilibrium approach; user-supplied impedance adjustment functions which can vary by functional class; and an option for computing link impedances which provide for variable weighting of time and distance.

While never implemented in the Texas Large Network Assignment Package (Texas Package), the equilibrium assignment procedure has become a widely used procedure in operational studies over the past 10 years. The principal difference between the equilibrium model and the capacity restraint procedure implemented in the Texas Package in 1979 is in the iteration weights. In the current Texas capacity restraint procedure, the analyst specifies the weights to be used in combining the iterations. In equilibrium assignments, these iteration weights are computed to minimize the impedance for each trip.

An equilibrium capacity restraint assignment procedure, therefore, was implemented in the Texas Package. The procedure was implemented as an option in both the ASSIGN SELF-BALANCING and PEAK CAPACITY RESTRAINT routines. The documentation manuals for these routines were also revised.

17. Key Words Traffic Assignment, Capacity Restraint, Equilibrium Assignment, Equilibrium Procedure		18. Distribution Statement No restrictions. This document is available to the public through the National Technical Information Service 5285 Port Royal Road Springfield, Virginia 22161.	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 153	22. Price

Form DOT F 1700.7 (8-69)

IMPLEMENTATION OF AN EQUILIBRIUM CAPACITY RESTRAINT MODEL IN THE TEXAS LARGE NETWORK ASSIGNMENT PACKAGE

by

Charles E. Bell
Systems Analyst

Jimmie D. Benson
Associate Research Engineer

J. Michael Heath
Engineering Research Associate

and

George B. Dresser
Research Statistician

Research Report 1153-6F

Study 2-10-89-1153
Improving the Efficiency, Effectiveness, and Responsiveness
of the Traffic Assignment Process

Sponsored by

Texas Department of Transportation

in cooperation with the
U.S. Department of Transportation
Federal Highway Administration

Texas Transportation Institute
The Texas A&M University System
College Station, Texas 77843-3135

November 1992

METRIC (SI*) CONVERSION FACTORS

APPROXIMATE CONVERSIONS TO SI UNITS

Symbol	When You Know	Multiply By	To Find	Symbol
		LENGTH		
in	inches	2.54	centimeters	cm
ft	feet	0.3048	meters	m
yd	yards	0.914	meters	m
mi	miles	1.61	kilometers	km
		AREA		
in^2	square inches	6.452	centimeters squared	cm^2
ft^2	square feet	0.0929	meters squared	m^2
yd^2	square yards	0.836	meters squared	m^2
mi^2	square miles	2.59	kilometers squared	km^2
ac	acres	0.395	hectares	ha
		MASS (weight)		
oz	ounces	28.35	grams	g
lb	pounds	0.454	kilograms	kg
T	short tons (2000 lb)	0.907	megagrams	Mg
		VOLUME		
fl oz	fluid ounces	29.57	millimeters	mL
gal	gallons	3.785	liters	L
ft^3	cubic feet	0.0328	meters cubed	m^3
yd^3	cubic yards	0.765	meters cubed	m^3

Note: Volumes greater than 1000 L shall be shown in m^3.

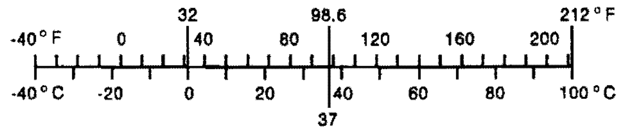
Symbol	When You Know	Multiply By	To Find	Symbol
		TEMPERATURE (exact)		
°F	Fahrenheit temperature	5/9 (after subtracting 32)	Celsius temperature	°C

These factors conform to the requirement of FHWA Order 5190.1A

*SI is the symbol for the International System of Measurements

APPROXIMATE CONVERSIONS TO SI UNITS

Symbol	When You Know	Multiply By	To Find	Symbol
		LENGTH		
mm	millimeters	0.039	inches	in
m	meters	3.28	feet	ft
yd	meters	1.09	yards	yd
km	kilometers	0.621	miles	mi
		AREA		
mm^2	millimeters squared	0.0016	square inches	in^2
m^2	meters squared	10.764	square feet	ft^2
yd^2	kilometers squared	0.39	square miles	mi^2
ha	hectares (10,000 m^2)	2.53	acres	ac
		MASS (weight)		
g	grams	0.0353	ounces	oz
kg	kilograms	2.205	pounds	lb
Mg	megagrams (1000 kg)	1.103	short tons	T
		VOLUME		
mL	millimeters	0.034	fluid ounces	fl oz
L	liters	0.264	gallons	gal
m^3	meters cubed	35.315	cubic feet	ft^3
m^3	meters cubed	1.308	cubic yards	yd^3
		TEMPERATURE (exact)		
°C	Celsius temperature	9/5 (then add 32)	Fahrenheit temperature	°F



ABSTRACT

One of the major objectives of Study 2-10-89-1153 has been to improve the accuracy of the assignment models implemented in the Texas Travel Demand Package. Three areas for improvements of the Texas capacity restraint model were identified: improved techniques for estimating iteration weights using an equilibrium approach; user-supplied impedance adjustment functions which can vary by functional class; and an option for computing link impedances which provide for variable weighting of time and distance.

While never implemented in the Texas Large Network Assignment Package (Texas Package), the equilibrium assignment procedure has become a widely used procedure in operational studies over the past 10 years. The principal difference between the equilibrium model and the capacity restraint procedure implemented in the Texas Package in 1979 is in the iteration weights. In the current Texas capacity restraint procedure, the analyst specifies the weights to be used in combining the iterations. In equilibrium assignments, these iteration weights are computed to minimize the impedance for each trip.

An equilibrium capacity restraint assignment procedure, therefore, was implemented in the Texas Package. The procedure was implemented as an option in both the ASSIGN SELF-BALANCING and PEAK CAPACITY RESTRAINT routines. The documentation manuals for these routines were also revised.

DISCLAIMER

EXECUTIVE SUMMARY

The equilibrium assignment has become a widely used procedure in operational studies over the past 10 years. It is a procedure which the U.S. Department of Transportation encourages. The use of equilibrium assignments is specified in the EPA reports, "Quality Review Guidelines for Base Year Emission Inventories" (September 1991) and "Section 187 VMT Forecasting and Tracking Guidance" (January 1992), and is expected to be included in future EPA documents affecting mobile sources. An equilibrium assignment procedure was therefore implemented in the Texas Package. The procedure was implemented as an option in both the ASSIGN SELF-BALANCING and PEAK CAPACITY RESTRAINT routines. The documentation manuals for these routines were also revised. These changes should insure that TxDOT's traffic assignment process is in compliance with federal guidelines.

The equilibrium assignment technique is an iterative capacity restraint assignment technique. Operationally, it is similar to the Texas Capacity Restraint Procedure with one major difference: The iteration weights are determined as a part of the iterative process. The method used for determining the iteration weights is the heart of the equilibrium process.

The theoretical constructs for equilibrium assignments are summarized as follows:

> The problem of trip assignment in the sequential urban travel-forecasting process is how to assign (or allocate) a specified number of vehicles to the paths taken from each origin to each destination. The path chosen by each traveler is generally assumed to be the path that minimizes his or her journey time, or some combination of time and cost. All travelers are assumed to have identical perceptions of travel time and cost. If the network is congested, that is, if each link's travel time depends of the flow of vehicles on that link, then the following equilibrium problem results: Finding the assignment of vehicle to links such that no traveler can reduce his or her travel time from origin to destination by switching to another path.

In other words, a traveler first selects the path along the route that requires the least amount of time. But other travelers also use parts of the path and the time increases. The traveler then shifts to a different path (as do other travelers). Then that path becomes congested, and another path is selected (as other travelers select different paths as well).

Eventually, the traveler cannot find a faster path, and the travel time on the final path is about the same as it would be on the original, congested path. At that point, the system is close to equilibrium.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

Highway traffic assignments are used by the Texas Department of Transportation (TxDOT) for a variety of purposes. Project ranking and Transportation Commission decisions, project development, analysis of highway alternatives, corridor analysis, geometric design, pavement design, and environmental analysis, are probably the most frequent applications. Traffic assignments are also used by TxDOT in conjunction with the Metropolitan Planning Organizations (MPOs) to develop urban area transportation plans. Assignments may be used as input to traffic simulation programs and for reconstruction management.

Traffic assignments are the product of either the traditional four-step urban transportation planning process (trip generation, trip distribution, mode split, and traffic assignment); or, for smaller urban areas where the mode choice analyses are not needed, the traditional three-step urban transportation planning process (trip generation, trip distribution, and traffic assignment). The highway assignment results are probably the most visible and widely used part of the modeling process. In recent years there has been a growing perception that the traffic assignment models in the Texas Package are not state-of-the-art since there have been no major modifications or improvements to the assignment models since 1979. Recognizing this, one of the major objectives of Study 2-10-89-1153 has been to improve the accuracy of the assignment models implemented in the Texas Travel Demand Package. Three areas for improvements of the Texas capacity restraint model were identified: improved techniques for estimating iteration weights using an equilibrium approach; user-supplied impedance adjustment functions which can vary by functional class; and an option for computing link impedances which provides for variable weighting of time and distance.

While never implemented in the Texas Large Network Assignment Package (Texas Package), the equilibrium assignment procedure has become a widely used procedure in operational studies over the past 10 years. It is a procedure which the U.S. Department of Transportation generally encourages. The principal difference between the equilibrium model and the capacity restraint procedure implemented in the Texas Package in 1979 is in the iteration weights. In the current Texas capacity restraint procedure, the analyst specifies the weights to be used in combining the iterations. In equilibrium assignments, these iteration weights are computed to minimize the impedance for each trip. The use of equilibrium assignments is specified in the EPA report "Quality Review Guidelines for Base Year Emission Inventories," (September 1991) and the EPA report "Section 187 VMT Forecasting and Tracking Guidance," (January 1992) and is expected to be included in future EPA documents affecting mobile sources as they are issued.

An equilibrium capacity restraint assignment procedure, therefore, was implemented in the Texas Package. The procedure was implemented as an option in both the ASSIGN SELF-BALANCING and PEAK CAPACITY RESTRAINT routines. The documentation manuals for these routines were also revised. These changes should insure that TxDOT's traffic assignment process is in compliance with federal guidelines.

II. OVERVIEW OF THE EQUILIBRIUM ASSIGNMENT TECHNIQUE

The equilibrium assignment technique is basically an iterative capacity restraint assignment technique. Operationally, it is similar to the Texas Capacity Restraint Procedure implemented in 1979 with one major difference: The iteration weights are determined as a part of the iterative process. The method used for determining the iteration weights is the heart of the equilibrium process. The purpose of this chapter is to provide an overview of the assignment technique. The information presented focuses on the theoretical constructs and mathematical description of the technique, where the equilibrium computations are performed in the iterative process, and the algorithm is used to estimate the equilibrium solution.

THEORETICAL CONSTRUCTS

The theoretical constructs for equilibrium assignments are commonly referred to as the Wardrop (5) conditions. In the paper entitled "Equilibrium Trip Assignment: Advantages and Implications for Practice" (1), the Wardrop conditions are summarized as follows:

> The problem of trip assignment in the sequential urban travel-forecasting process is how to assign (or allocate) a specified number of vehicles (or persons) to the paths taken from each origin to each destination. The path chosen by each traveler is generally assumed to be the path that minimizes his or her journey time, or some combination of time and cost. All travelers are assumed to have identical perceptions of travel time and cost. If the network is congested, that is, if each link's travel time depends of the flow of vehicles on that link, then the following equilibrium problem results: Find the assignment of vehicle to links such that no traveler can reduce his or her travel time from origin to destination by switching to another path.

In other words, a traveler first selects his path along the route he believes to be the minimum time. But other travelers also use parts of his path and the time increases. He then shifts to a different path (as do other travelers). Then that path gets congested, and he selects another path (as others may do, also). Eventually, he cannot find a faster path, and the travel time on the final path is about the same as it would be on the congested original path. At that point, the system is close to equilibrium.

MATHEMATICAL DESCRIPTION

Perhaps the most commonly cited mathematical description of the user-equilibrium problem is provided in the paper "Equilibrium Trip Assignment: Advantages and Implications for Practice" by Eash, Janson, and Boyce (1). The following mathematical description was based on this paper but uses somewhat revised notation. If the trip matrix (T_{ij}) is given, the equilibrium assignment of trips to links may be found by solving the following nonlinear programming problem:

Find the value of λ_k to:

$$Minimize: \sum_a \int_0^{V_{ak}} S_a(x)dx$$

Subject to:

$$V_{ak} = (1-\lambda_k)V_{a(k-1)} + \lambda_k W_a$$

$$V_{ak} = \sum_i \sum_j \sum_r \delta_{arij} X_{rij}$$

$$\sum_r X_{rij} = T_{ij}$$

$$X_{rij} \geq 0$$

$$0 \leq \lambda_k \leq 1$$

Where:

λ_k	=	the "lambda" value estimated in the optimization process for iteration *k*.
V_{ak}	=	the weighted average volume for link *a* for iteration *k*.
W_{ak}	=	the all-or-nothing assigned volume on link *a* of the network for iteration *k*.

$S_a(x)$	=	the average impedance (normal travel time) on link *a* with a flow volume of *x; the impedance with flow x.* Typically a capacity restraint impedance adjustment function is used to estimate the impedances that would occur for the various volumes on the link .
δ_{arij}	=	Delta variable will have a value of either 0 or 1: If link "a" belongs to path "r" from zone "i" to zone "j", then $\delta_{arij} = 1$; otherwise, $\delta_{arij} = 0$.
X_{rij}	=	number of vehicles of zone *i* to zone *j* on path *r*.

For all links *a* in the networks: $i = 1,...N$; $j = 1,...N$; and N = number of zones.

The link flows for which the objective function of the equilibrium assignment achieves its minimum value are those that satisfy the equilibrium conditions stated by Wardrop. Hence, the solution that minimizes the sum of the integral of the congestion functions for all links is the equilibrium solution.

In the objective function, the impedance values $S_a(x)$ for the values of x from 0 to V_{ak} for link *a* create a curve in which the impedances increase with volume. The integral in the objective function finds the area under this curve for volumes from 0 to V_{ak}. Eash, Janson and Boyce (1) note that:

> . . . The area under curve S_a has no (known) interpretation. Why, then, should we be interested in minimizing the sum of these areas over all links? The answer to the question is conceptually simple. The link flows for which this objective function achieves its minimum value are those that satisfy the equilibrium conditions stated by Wardrop.

ALTERNATIVE OBJECTIVE FUNCTION

In the implementation of the equilibrium procedure in the Texas Package, an option to use an alternative objective function was provided. The alternative objective function is:

Find the value of λ_k to:

$$\textit{Minimize}: \sum_a V_{ak} S_a(V_{ak})$$

Subject to the same constraints as the previous objective function.

This alternative objective function is the objective function used in the Equilibrium Assignment in TRANPLAN Version 7.0 (9). Since TxDOT has adopted TRANPLAN for its micro-computer applications, the mainframe assignment package provides an option for using the same objective function. Perhaps the most salient features of the alternative objective function are that it is easier to compute, and the objective function has a real interpretation (unlike the classic objective function). Using the alternative objective function and impedances representing average travel time per vehicle, the optimal λ values are estimated, minimizing the total vehicle hours of travel on the network links with a specified capacity. Using the alternative objective function and impedances representing average travel costs per vehicle (instead of travel time), the optimal λ values are estimated to minimize the total travel costs on the network links with specified capacities.

EQUILIBRIUM ASSIGNMENT ALGORITHM

The equilibrium assignment is basically an iterative capacity restraint assignment procedure. Given a network with congestion functions for each link, a trip matrix to be assigned, and a current solution for the link loadings (i.e., the resulting weighted average volume from iteration *k* noted as $V_{a(k-1)}$), the following five steps are performed for iteration *k*:

1. Compute the travel time on each link $S_a(V_{a(k-1)})$ that corresponds to the flow $V_{a(k-1)}$ in the current solution. In other words, the weighted average volume from the preceding iteration is used to compute the link impedances for the new all-or-nothing assignment for this iteration.
2. Trace minimum path trees from each origin to all destinations using the link impedances (i.e., travel times) from Step 1.
3. Assign all trips from each origin to each destination for the minimum path

(all-or-nothing assignment). Call this link loading W_{ak}.

4. Combine the current solution ($V_{a(k-1)}$) and the new assignment (W_a) to obtain a new current solution (V_{ak}) by using a value λ_k. The λ value for iteration k (λ_k) is estimated in this step. A non-linear programming technique is used to estimate the lambda value, which minimizes one of the two objective functions (which is selected by the user):

   Either:

$$Minimize: \sum_a \int_0^{V_{ak}} S_a(x)dx$$

   or alternatively:

$$Minimize: \sum_a V_{ak} S_a(V_{ak})$$

   Where:

$$V_{ak} = (1-\lambda_k)V_{a(k-1)} + \lambda_k W_a$$

5. If this is the last iteration, the procedure terminates and the V_{ak} link volumes are the final results. Otherwise, the next iteration is performed (i.e., return to Step 1).

Figure II-1 provides a graphical description of the iterative process for an equilibrium assignment using five iterations.

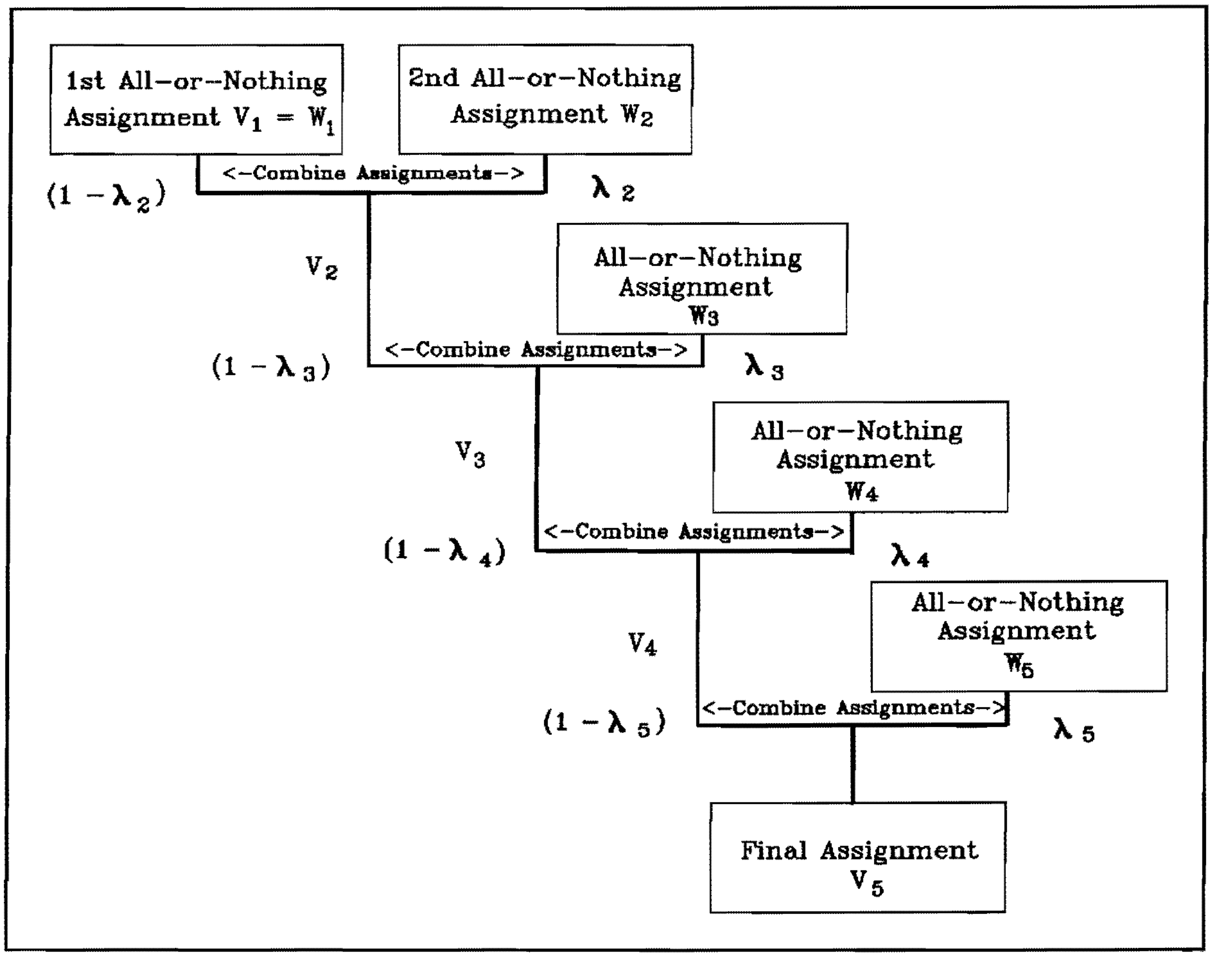


Figure II-1. Iterative Process for a Five Iteration Equilibrium Assignment

ALGORITHM FOR ESTIMATING LAMBDA

The value of λ is estimated for combining the new all-or-nothing assignment with the current solution

$$V_{ak}=(1-\lambda_k)V_{a(k-1)}+\lambda_k W_a$$

such that the new assignment will minimize the objective function. Figure II-2 illustrates a typical relationship between potential lambda values and the objective function results. The problem is to find the lambda value which provides the minimum value of the objective. The following step-by-step description of the algorithm used to estimate lambda minimizes the objective function value:

1. Set $\lambda_{Low} = 0$ and compute the objective function.
2. Set $\lambda_{High} = 1$ and compute the objective function.
3. Set $\lambda_{trial} = ((\lambda_{High} + \lambda_{Low})/2)$.
4. Calculate the objective function for $\lambda_{Trial.}$
5. If the objective function for λ_{High} is greater than the objective function for λ_{Low}, then replace λ_{High} and its respective objective function with λ_{Trial} and its objective function.
6. If the objective function for λ_{Low} is greater than or equal to the objective function for λ_{High}, then replace λ_{Low} and its respective objective function with λ_{Trial} and its objective function.
7. Steps 3 through 6 are repeated 17 times to ensure that the λ which was found effectively approximates the minimum objective function.

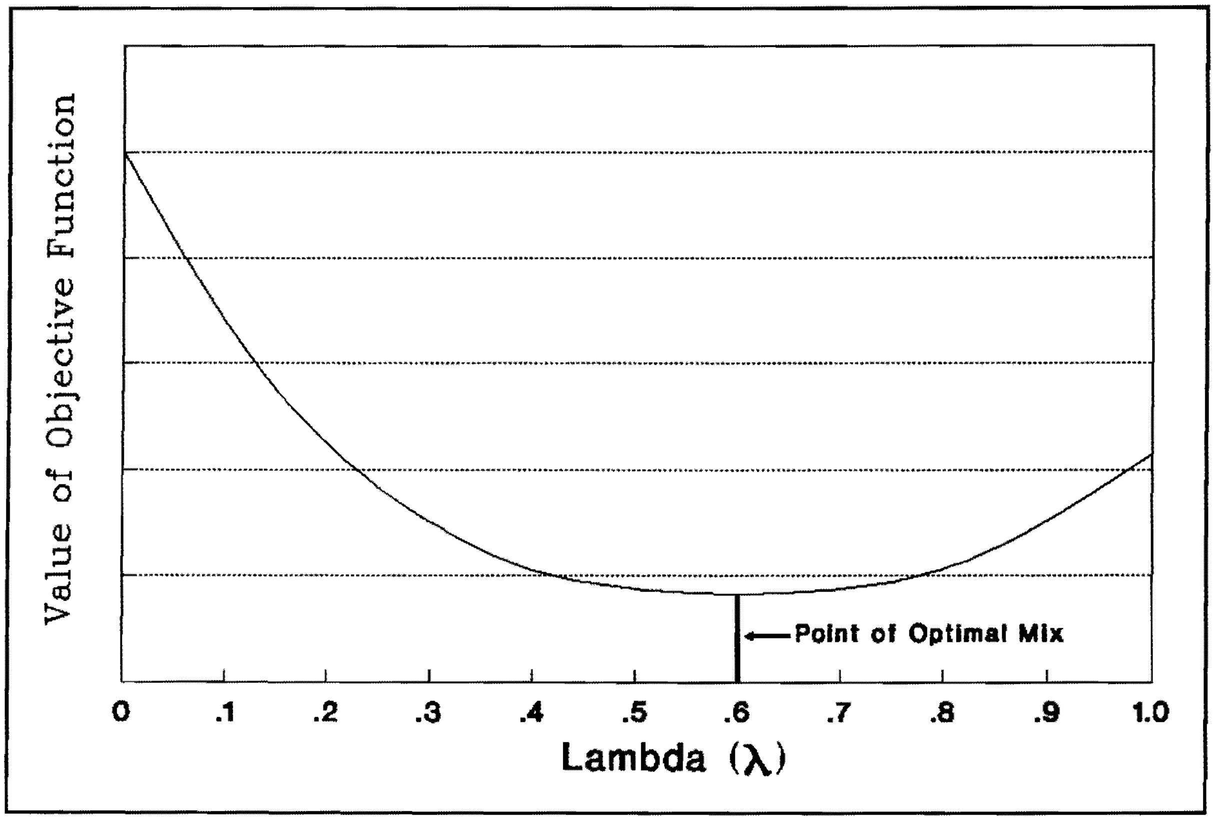


FIGURE II-2. Relationship between Objective Function Values and Potential Lambda Values

III. SOFTWARE MODIFICATIONS

The Texas Large Network Assignment Package has two routines which perform capacity restraint assignments: the ASSIGN SELF-BALANCING routine and the PEAK CAPACITY RESTRAINT routine. The ASSIGN SELF-BALANCING routine is used to perform 24-hour capacity restraint assignments and the PEAK CAPACITY RESTRAINT is used to perform peak period (or time-of-day) assignments.

CURRENT 24-HOUR CAPACITY RESTRAINT ROUTINE

The ASSIGN SELF BALANCING routine is somewhat unique (relative to other software packages) in its specific orientation to 24-hour assignments. The user-supplied link data (which is input via the ASSEMBLE NETWORK routine) specify the nondirectional link capacities and speeds. In applying the capacity restraint, a link's V/C ratio is computed by dividing the nondirectional link volume (i.e., the sum of the link's A-to-B volume and B-to-A volume) by the link's nondirectional capacity (specified in the link data).

The routine currently provides two capacity restraint options: the old capacity restraint procedure originally implemented in the package in 1970 (an option no longer used) and the improved procedure implemented in the routine in 1979. Under the old original 1970 routine, a link's travel time/speed was adjusted only if it's V/C ratio exceeded 1.0 for one or more iterations. In relatively uncongested networks, the travel time/speed on only a few links might be adjusted. This was a relatively weak capacity restraint procedure which would no longer be considered desirable. The 1970 version also used a regression technique to estimate iteration weights. Early experience with the iteration weights model found that it provided unreliable and questionable results. The routine was quickly modified to allow the user to specify the iteration weights. In some respects, the implementation of equilibrium techniques is actually an implementation of a much improved version of a 1970 feature.

The option implemented in the routine in 1979 represented a major improvement. This improved option offered several very salient features:

- Under the 1979 option, the impedance adjustment was applied to all links with specified capacities after each iteration (not just those links which have a V/C ratio greater than 1.0 for one or more iterations).

- Under the 1979 option, a modified version of the Bureau of Public Roads (BPR) impedance adjustment function was implemented. The original BPR function assumed that the input speed was a free-flow speed. In the Texas applications, the input link speeds were 24-hour speeds and not free-flow speeds.
- The routine continued to allow the user to specify the iteration weights.
- Under the 1979 option, the V/C ration computation for the application of the impedance adjustment function used the weighted average of the preceding iterations. The user-specified iteration weights were used to define the relative weights during the iterative process. This was a major departure from the previous practice of using only the volume from the preceding iteration.

CURRENT PEAK CAPACITY RESTRAINT ROUTINE

The PEAK CAPACITY RESTRAINT routine and the PEAK ASSEMBLE NETWORK routine were developed for the Texas Large Network Assignment Package in 1988. These routines are actually modified versions of the ASSIGN SELF-BALANCING routine and the ASSEMBLE NETWORK routine. The new routines developed in 1988 provide for the handling of directional speeds and capacities rather than the nondirectional speeds and capacities of their predecessors. These routines were needed to facilitate the assignment of peak period trips to a peak period network and to reflect the directional imbalances (both in terms of volumes and speeds) typically observed during peak periods on most facilities.

EQUILIBRIUM OPTION

Both the ASSIGN SELF-BALANCING routine and the PEAK CAPACITY RESTRAINT routine were modified to provide the option of performing an equilibrium assignment. From a user perspective, the basic difference is that the equilibrium assignment procedure will compute the iteration weights; and, hence, the user will not need to supply these weights.

EQUILIBRIUM OPTIMIZATION OPTIONS

With the equilibrium assignment procedure implemented, there are two objective functions that can be selected for optimization. The first is the classic objective function as described in the paper entitled "Equilibrium Trip Assignment: Advantages and Implications for Practice" (1). This was implemented in the Texas Package software as the default objective function. The alternative (or second) objective function estimates the iteration weights to minimize the total vehicular impedance (i.e., vehicle hours of travel). The software was developed to allow the user to optionally select this alternative objective function for use in the equilibrium process. The mathematical descriptions of both objective functions are provided in Chapter II of this report.

TRAVEL TIME/SPEED ADJUSTMENT OPTIONS

The capacity restraint procedure implemented in 1979 used a modified version of the widely used BPR impedance adjustment function. The BPR function assumes the impedance is based on a free-flow (or "zero volume") link speed. Since the Texas highway networks have traditionally been coded using a 24-hour speed rather than free-flow speeds, a modified version of the BPR impedance adjustment function was implemented in 1979. The formula for the Texas impedance adjustment function is:

$$I_{n+1} = [0.92 + 0.15\,(V/C)^4]\,I_0$$

Where:

I_0	=	the initial link impedance (travel time) using the input speed.
I_{n+1}	=	the link impedance (travel time) for iteration n+1.
V	=	the weighted average assigned link volume from iterations 1 to n.
C	=	the link capacity.

A constraint is imposed to limit the magnitude of the impedance (travel time) adjustment. The maximum impedance adjustment varies by iteration. Following the initial assignment, the maximum impedance adjustment factor is 2 (i.e., essentially reducing the speed by one half). The maximum impedance adjustment factor is increased by 1 for each succeeding iteration. To reflect this constraint, the equation could be expressed as:

$$I_{n+1} = \min\{ [0.92 + 0.15 (V/C)^4]\, I_0 \;, [n+1]\, I_0 \}$$

This formulation of the equation directly reflects the constraint. In the ASSIGN SELF-BALANCING routine, the impedance adjustment is applied using the weighted average nondirectional volume and capacity. In the PEAK CAPACITY RESTRAINT routine, the impedance adjustment is applied directionally to each link.

Two Travel Time/Speed Adjustment Options Implemented

Both the ASSIGN SELF BALANCING and the PEAK CAPACITY RESTRAINT routines have been modified to allow the user to specify different travel time or speed adjustment functions by functional classification. The following describes the two options provided in the new version of these routines:

Option 1 - Variable BPR Form Functions

The first option implemented allows the user to specify different variations of the traditional BPR function by functional classification code. The general form of the impedance adjustment function by functional classification is:

$$I_{n+1} = [A_f + B_f\, (V/C)^{D_f}]\, I_0$$

Where:

I_0	=	the initial link impedance (travel time) using the input speed.
I_{n+1}	=	the link impedance (travel time) for iteration n+1.
V	=	the weighted average assigned link volume from iterations 1 to n.
C	=	the link capacity.
A_f	=	the user-supplied constant term for functional classification f.
B_f	=	the user-supplied coefficient term for functional classification f.
D_f	=	the user-supplied exponent for functional classification f.
f	=	the functional classification code for the link to which the adjustment is to be applied. The functional classification codes are a hexadecimal digit whose value can be 0, 1, 2, 3, 4, 5, 6, 7,

8, 9, A, B, C, D, E, or F.

As with the current model, a constraint is imposed to limit the magnitude of the impedance (travel time) adjustment. The maximum impedance adjustment varies by iteration. Following the initial assignment, the maximum impedance adjustment factor is 2 (i.e., essentially reducing the speed by one half). The maximum impedance adjustment factor is increased by 1 for each of the succeeding iterations. To reflect this constraint, the equation could be expressed as:

$$I_{n+1} = \min\{\ [A_f + B_f\,(V/C)^{D_f}]\ I_0\ \ ,\ [n+1]\ I_0\ \}$$

This version of the generalized BPR formulation directly reflects the constraint.

Option 2 - Graphical Description of Impedance Adjustment Functions

In some applications, the user may wish to try an impedance adjustment function that is not in the form of the BPR function. To provide for such user-specified impedance adjustment functions, a second option was included which allows the user to describe the desired function graphically via a series of coordinates describing the function. Under Option 2, the impedance adjustment function and the constraints which are applied by the Texas Package software can be expressed as follows:

$$I_{n+1} = \min\{\ [U_f\,(V/C)]\ I_0\ \ ,\ [n+1]\ I_0\ \}$$

Where:

I_0	=	the initial link impedance (travel time) using the input speed.
I_{n+1}	=	the link impedance (travel time) for iteration n+1.
V	=	the weighted average assigned link volume from iterations 1 to n.
C	=	the link capacity.
$U_f(V/C)$	=	the adjustment factor estimated from the graphical description of the user-supplied impedance adjustment function for functional classification f.

To graphically describe the function for a given functional classification, the user will input the impedance adjustment factor (i.e., the $U_f(V/C)$ values) for various V/C ratios from 0.0 to 4.0. The user can use from two points up to 400 points to describe this curve for a given functional classification. The user must provide the impedance adjustment factors for V/C ratio of 0.0. If a factor is not specified for the V/C ratio of 4.0, then the factor for the largest V/C ratio supplied will be used for the factor at a V/C of 4.0. For V/C ratios that fall between the user-supplied points, simple linear interpolation is used to estimate the factor from the points provided on either side of the V/C ratio. If a V/C ratio greater than 4.0 is encountered, the impedance adjustment factor for the V/C of 4.0 is used.

Applications Perspective

While it is possible to specify 16 different adjustment functions, it is unlikely that more than two or three different functions would be used for a given area. For example, a freeway adjustment function might be defined. If functional codes 1, 2, A, and B are used for freeways, then the same user-supplied values would be entered for each of the freeway functional classification codes. By allowing the user to specify the adjustment by functional code, there is no restriction in the choice of codes to use for a given type of facility. In other words, the software was designed for flexibility. Its design does not contemplate that the user would elect to use 16 different impedance adjustment functions.

REVISED TABULAR SUMMARY OF ITERATION RESULTS

TABLE L1 of both ASSIGN SELF-BALANCING and PEAK CAPACITY RESTRAINT summarizes the assignment results and assignment speeds by iteration for each link. A revised format for Table L1 is used under the equilibrium option. Table III-1 illustrates the Table L1 format as produced by ASSIGN SELF-BALANCING for six example links when the equilibrium option is not selected (i.e., the old format prior to the equilibrium revisions). The hypothetical assignment being summarized was a six iteration assignment with iteration weights of 10, 10, 20, 20, 20, and 20 percent which are the equivalent of lambda values of $\lambda_1 = 1.0$, $\lambda_2 = 0.5$, $\lambda_3 = 0.5$, $\lambda_4 = 0.3333333$, $\lambda_5 = 0.25$, and $\lambda_5 = 0.2$. In Table III-1, VOL 1 through VOL 6 are the nondirectional all-or-nothing assignment results from the six iterations. The VOL 7 results are the weighted averages of the six all-or-nothing assignments (i.e., the final capacity restrained assignment results). In

Table III-1, SPD 1 through SPD 6 are estimates of the nondirectional speeds used in each of the six all-or-nothing assignments. It should be noted that these speed estimates are computed using the link's impedance in integer hundredths of a minute and the link's distance in integer hundredths of a mile. The SPD 7 value is a somewhat unusual value. SPD 7 is determined by computing the weighted average link impedance (using the same iteration weights used in the volume computations) and the link's distance. In essence, this is a weighted average speed but is not the weighted average speed of the traffic on the link. A weighted average speed computed in this manner is of little value and should be used with caution. A better way to compute the weighted speed would be to divide the weighted average vehicle miles of travel (VMT) on the link by the weighted average vehicle hours of travel (VHT) on the link.

The V/C ratios are also displayed as shown in Table III-1. For the first six iterations, the V/C ratio data displayed are the all-or-nothing assignment V/C ratio results. The seventh V/C ratio is the weighted average volume divided by the link capacity. To facilitate identification of links with high V/C ratios in Tables III-1 and III-4, symbols are also provided to indicate the V/C ratio range:

	V/C Ratio Range	
Symbol	From	To
"."	0.0	1.24
"+"	1.24	1.49
"#"	1.49	1.99
"*"	1.99	and above

The TABLE L1 format was revised for equilibrium assignments to display the weighted average volume at the end of the iteration rather than the individual all-or-nothing results. Table III-2 summarizes the same assignment results for the same six links using the revised table format for equilibrium assignments (i.e., assuming an equilibrium assignment was performed and happened to have computed lambda values of $\lambda_1 = 1.0$, $\lambda_2 = 0.5$, $\lambda_3 = 0.5$, $\lambda_4 = 0.3333333$, $\lambda_5 = 0.25$, and $\lambda_5 = 0.2$). At the end of the iteration, the all-or-nothing assignment results are, of course, the weighted average results (i.e., the VOL 1 results are the same in Tables III-1 and III-2). The VOL 2 results are the weighted averages of the all-

or-nothing results from iterations 1 and 2 (in this example, the weighted average was computed using $\lambda_2 = 0.5$). Similarly, the VOL 3 results are effectively the weighted average of the all-or-nothing results from iterations 1, 2, and 3. The VOL 3 results in the Table III-2 examples can be computed using the VOL 2 results in Table III-2 (i.e., the weighted average of assignments 1 and 2 which can be represented by V_2), the VOL 3 results displayed in Table III-1 (i.e., the all-or-nothing results for iteration 3 which can be represented by W_3), and $\lambda_3 = 0.5$ (in this example the weighted average was computed using $\lambda_2 = 0.5$). The following general formula can be used for this computation:

$$V_k=(1-\lambda_k)V_{k-1}+\lambda_k W_k$$

VOL 4, VOL 5, and VOL 6 results can likewise be computed using:

$$\boldsymbol{\lambda_4 = 0.3333333, \lambda_5 = 0.25, \text{ and } \lambda_5 = 0.2.}$$

As may be noted, the VOL 6 results in Table III-2 are the weighted averages of the six all-or-nothing assignments and, hence, correspond to the results displayed under VOL 7 in Table III-1. The V/C ratio summaries were also change to correspond to the weighted average volumes rather than the all-or-nothing assignment results.

The PEAK CAPACITY RESTRAINT routine also produces a TABLE L1 summary. The TABLE L1 produced by the PEAK CAPACITY RESTRAINT routine summarizes the directional volumes and speeds by link. Table III-3 illustrates the format used for non-equilibrium assignment using results from a hypothetical peak-hour assignment (for the same six links shown in Tables III-1 and III-2). Again this hypothetical assignment was a six-iteration assignment with iteration weights of 10, 10, 20, 20, 20, and 20 percent (which are the equivalent of lambda values of: $\lambda_1 = 1.0$, $\lambda_2 = 0.5$, $\lambda_3 = 0.5$, $\lambda_4 = 0.3333333$, $\lambda_5 = 0.25$, and $\lambda_5 = 0.2$).

The TABLE L1 format from PEAK CAPACITY RESTRAINT was also revised for equilibrium assignments to display the weighted average volume at the end of the iteration rather than the individual all-or-nothing results (non-equilibrium assignments will continue to use the old format). Table III-4 summarizes the same assignment results for the same six links shown in Table III-3 using the revised table format for equilibrium assignments (i.e., assuming an equilibrium assignment was performed and happened to have computed lambda values of: $\lambda_1 = 1.0$, $\lambda_2 = 0.5$, $\lambda_3 = 0.5$, $\lambda_4 = 0.3333333$, $\lambda_5 = 0.25$, and $\lambda_5 = 0.2$).

The revised format for equilibrium assignments display the convergence characteristics of the capacity restraint assignment better. Consideration should be given to implementing this format for all capacity restraint assignments and not just for equilibrium assignments.

USER MANUAL REVISIONS

The two routines in the Texas Large Network Assignment Package that were revised under this study were the ASSIGN SELF-BALANCING routine and the PEAK CAPACITY RESTRAINT routine. These changes require revisions in the operating manual. The revised operating instructions for the ASSIGN SELF-BALANCING routine are provided in Appendix A. The revised operating instructions for the PEAK CAPACITY RESTRAINT routine are provided in Appendix B.

TABLE III-1
Old Tabular Summary for Six Sample Links
from the ASSIGN SELF-BALANCING Routine

```
                1995 24-HOUR ASSIGNMENT FOR EXAMPLE, TEXAS                        WEIGHTED      NOV 3, 1992

 TABLE L1                                LIST OF VOLUMES AND SPEEDS FOR UPDATED LINKS

                                     VOL 1  VOL 2  VOL 3  VOL 4  VOL 5  VOL 6  VOL 7          V/C V/C V/C V/C V/C V/C V/C
  ANODE BNODE  DIST LN CAPACITY     SPD 1  SPD 2  SPD 3  SPD 4  SPD 5  SPD 6  SPD 7           1   2   3   4   5   6   7
  ----- ----- ----- -- ---------  ------ ------ ------ ------ ------ ------ ------          --- --- --- --- --- --- ---
   3236  3237  1.20  3     95000 108649  84598 100753  98620  96086 100538  98524           .   .   .   .   .   .   .
                                   55.0   46.8   51.1   50.3   50.3   50.7   50.6          1.1  .9 1.1 1.0 1.0 1.1 1.0

   3281  3282  1.12  3     95000 124050  79716 112964  88013 118493  93317 102934           +   .   .   .   .   .   .
                                   55.1   40.7   49.4   47.3   49.8   48.0   48.2          1.3  .8 1.2  .9 1.2 1.0 1.1

   3313  3314  1.10  4    123330 117749 140313 116183 135704 113737 126330 124197           .   .   .   .   .   .   .
                                   55.0   52.8   50.4   52.0   50.8   51.6   51.7          1.0 1.1  .9 1.1  .9 1.0 1.0

   4135  5399   .39  2     21660  44628  12064  36771  10704  62113  15352  30657           *   .   #   .   *   .   +
                                   34.9   17.5   26.0   20.9   29.3   18.7   23.1          2.1  .6 1.7  .5 2.9  .7 1.4

   4211  4212   .50  3     28330  32602  24041  38395  41516  26594  51115  37188           .   .   +   #   .   #   +
                                   30.0   25.4   28.3   25.0   22.9   24.6   25.5          1.2  .8 1.4 1.5  .9 1.8 1.3

   5151  5152   .41  1      9990  17798   4344  13052  10313  11800  12632  11774           #   .   +   .   .   +   .
                                   28.0   14.0   24.6   22.8   23.9   23.7   22.5          1.8  .4 1.3 1.0 1.2 1.3 1.2
```

TABLE III-2:
Revised Tabular Summary for Six Sample Links
from the ASSIGN SELF-BALANCING Routine

```
              1995 24-HOUR ASSIGNMENT FOR EXAMPLE, TEXAS                  WEIGHTED      NOV 3, 1992


TABLE L1                              LIST OF SPEEDS AND WEIGHTED VOLUMES FOR UPDATED LINKS

                                    VOL 1  VOL 2  VOL 3  VOL 4  VOL 5  VOL 6                 V/C V/C V/C V/C V/C V/C
 ANODE BNODE  DIST LN CAPACITY  SPD 1  SPD 2  SPD 3  SPD 4  SPD 5  SPD 6                   1   2   3   4   5   6
 ----- ----- ----- -- -------- ------ ------ ------ ------ ------ ------                 --- --- --- --- --- ---
  3236  3237  1.20  3    95000 108649  96624  98688  98666  98021  98524                   .   .   .   .   .   .
                                 55.0   46.8   51.1   50.3   50.3   50.7                 1.1 1.0 1.0 1.0 1.0 1.0

  3281  3282  1.12  3    95000 124050 101883 107424 100953 105338 102934                   +   .   .   .   .   .
                                 55.1   40.7   49.4   47.3   49.8   48.0                 1.3 1.1 1.1 1.1 1.1 1.1

  3313  3314  1.10  4   123330 117749 129031 122607 126973 123664 124197                   .   .   .   .   .   .
                                 55.0   52.8   50.4   52.0   50.8   51.6                 1.0 1.0 1.0 1.0 1.0 1.0

  4135  5399   .39  2    21660  44628  28346  32559  25274  34484  30657                   *   +   #   .   #   +
                                 34.9   17.5   26.0   20.9   29.3   18.7                 2.1 1.3 1.5 1.2 1.6 1.4

  4211  4212   .50  3    28330  32602  28322  33358  36078  33707  37188                   .   .   .   +   .   +
                                 30.0   25.4   28.3   25.0   22.9   24.6                 1.2 1.0 1.2 1.3 1.2 1.3

  5151  5152   .41  1     9990  17798  11071  12062  11479  11559  11774                   #   .   .   .   .   .
                                 28.0   14.0   24.6   22.8   23.9   23.7                 1.8 1.1 1.2 1.1 1.2 1.2
```

TABLE III-3:
Old Tabular Summary for Six Sample Links
from the PEAK CAPACITY RESTRAINT Routine

```
              1995 MORNING PEAK HOUR ASSIGNMENT FOR EXAMPLE, TEXAS                WEIGHTED      NOV 3, 1992


TABLE L1                                   LIST OF VOLUMES AND SPEEDS FOR UPDATED DIRECTIONAL LINKS

                                  VOL 1  VOL 2  VOL 3  VOL 4  VOL 5  VOL 6  VOL 7        V/C V/C V/C V/C V/C V/C V/C
 ANODE BNODE  DIST FC LN CAPACITY SPD 1  SPD 2  SPD 3  SPD 4  SPD 5  SPD 6  SPD 7         1   2   3   4   5   6   7
 ----- ----- ----- -- -- -------- ------ ------ ------ ------ ------ ------ ------       --- --- --- --- --- --- ---
  3236  3237  1.20  1  3     5700   6671   4189   6175   5692   5712   5813   5764         .   .   .   .   .   .   .
                                    55.0   45.9   52.9   51.1   51.4   51.4   51.4       1.2  .7 1.1 1.0 1.0 1.0 1.0
     (B-A)    1.20  1  3     5700   4433   4457   4122   4387   4108   4462   4305         .   .   .   .   .   .   .
                                    55.0   56.7   56.7   57.1   57.1   57.1   56.8        .8  .8  .7  .8  .7  .8  .8

  3281  3282  1.12  1  3     5700   8793   4022   7899   4663   8101   5571   6528         #   .   +   .   +   .   .
                                    55.1   31.1   47.7   42.8   48.0   45.4   44.5       1.5  .7 1.4  .8 1.4 1.0 1.1
     (B-A)    1.12  1  3     5700   3885   4125   3646   4332   4009   3966   3992         .   .   .   .   .   .   .
                                    55.1   57.9   57.9   57.9   57.9   57.9   57.6        .7  .7  .6  .8  .7  .7  .7

  3313  3314  1.10  1  4     7400   6433   8629   6175   8114   5991   7219   7006         .   .   .   .   .   .   .
                                    55.0   55.0   51.2   53.7   52.0   53.2   53.0        .9 1.2  .8 1.1  .8 1.0  .9
     (B-A)    1.10  1  4     7400   5601   5711   5699   5755   5633   5692   5687         .   .   .   .   .   .   .
                                    55.0   56.9   56.9   56.9   56.9   56.9   56.7        .8  .8  .8  .8  .8  .8  .8

  4135  5399   .39  3  2     1300   3094     51   2977    102   5240    100   1998         *   .   *   .   *   .   #
                                    34.9   17.5   28.2   15.1   28.9   12.2   19.1       2.4  .0 2.3  .1 4.0  .1 1.5
     (B-A)     .39  3  2     1300   1467   1182    781    992   1108   1469   1135         .   .   .   .   .   .   .
                                    34.9   30.4   32.5   36.0   36.0   36.0   34.5       1.1  .9  .6  .8  .9 1.1  .9

  4211  4212   .50  4  3     1700   1059    972   1330   1301   1481   1239   1273         .   .   .   .   .   .   .
                                    30.0   31.9   32.3   31.6   31.6   31.3   31.5        .6  .6  .8  .8  .9  .7  .7
     (B-A)     .50  4  3     1700   2273   1485   2594   2942   1237   3985   2527         +   .   #   #   .   *   #
                                    30.0   21.6   26.3   22.1   18.9   22.9   22.7       1.3  .9 1.5 1.7  .7 2.3 1.5

  5151  5152   .41  5  1      600   1397      0    928    632    755    802    763         *   .   #   .   +   +   +
                                    28.0   14.0   23.4   19.7   21.8   21.8   20.9       2.3  .0 1.5 1.1 1.3 1.3 1.3
     (B-A)     .41  5  1      600    422    444    406    422    451    489    440         .   .   .   .   .   .   .
                                    28.0   29.3   29.3   29.6   29.6   29.3   29.3        .7  .7  .7  .7  .8  .8  .7
```

TABLE III-4:
Revised Tabular Summary for Six Sample Links
from the PEAK CAPACITY RESTRAINT Routine

```
          1995 MORNING PEAK HOUR ASSIGNMENT FOR EXAMPLE, TEXAS                   WEIGHTED      NOV 3, 1992

TABLE L1                         LIST OF SPEEDS AND WEIGHTED VOLUMES FOR UPDATED DIRECTIONAL LINKS

                                     VOL 1  VOL 2  VOL 3  VOL 4  VOL 5  VOL 6                    V/C V/C V/C V/C V/C V/C
 ANODE BNODE  DIST FC LN CAPACITY    SPD 1  SPD 2  SPD 3  SPD 4  SPD 5  SPD 6                     1   2   3   4   5   6
 ----- ----- ----- -- -- -------- ------ ------ ------ ------ ------ ------                    --- --- --- --- --- ---
  3236  3237  1.20  1  3     5700   6671   5430   5803   5766   5752   5764                     .   .   .   .   .   .
                                    55.0   45.9   52.9   51.1   51.4   51.4                    1.2 1.0 1.0 1.0 1.0 1.0
    (B-A)     1.20  1  3     5700   4433   4445   4284   4318   4266   4305                     .   .   .   .   .   .
                                    55.0   56.7   56.7   57.1   57.1   57.1                     .8  .8  .8  .8  .7  .8

  3281  3282  1.12  1  3     5700   8793   6408   7153   6323   6768   6528                     #   .   +   .   .   .
                                    55.1   31.1   47.7   42.8   48.0   45.4                    1.5 1.1 1.3 1.1 1.2 1.1
    (B-A)     1.12  1  3     5700   3885   4005   3826   3994   3998   3992                     .   .   .   .   .   .
                                    55.1   57.9   57.9   57.9   57.9   57.9                     .7  .7  .7  .7  .7  .7

  3313  3314  1.10  1  4     7400   6433   7531   6853   7273   6953   7006                     .   .   .   .   .   .
                                    55.0   55.0   51.2   53.7   52.0   53.2                     .9 1.0  .9 1.0  .9  .9
    (B-A)     1.10  1  4     7400   5601   5656   5678   5703   5686   5687                     .   .   .   .   .   .
                                    55.0   56.9   56.9   56.9   56.9   56.9                     .8  .8  .8  .8  .8  .8

  4135  5399   .39  3  2     1300   3094   1573   2275   1551   2473   1998                     *   .   #   .   #   #
                                    34.9   17.5   28.2   15.1   28.9   12.2                    2.4 1.2 1.8 1.2 1.9 1.5
    (B-A)      .39  3  2     1300   1467   1325   1053   1033   1051   1135                     .   .   .   .   .   .
                                    34.9   30.4   32.5   36.0   36.0   36.0                    1.1 1.0  .8  .8  .8  .9

  4211  4212   .50  4  3     1700   1059   1016   1173   1216   1282   1273                     .   .   .   .   .   .
                                    30.0   31.9   32.3   31.6   31.6   31.3                     .6  .6  .7  .7  .8  .7
    (B-A)      .50  4  3     1700   2273   1879   2237   2472   2163   2527                     +   .   +   #   +   #
                                    30.0   21.6   26.3   22.1   18.9   22.9                    1.3 1.1 1.3 1.5 1.3 1.5

  5151  5152   .41  5  1      600   1397    699    813    753    753    763                     *   .   #   .   +   +
                                    28.0   14.0   23.4   19.7   21.8   21.8                    2.3 1.2 1.4 1.3 1.3 1.3
    (B-A)      .41  5  1      600    422    433    420    420    428    440                     .   .   .   .   .   .
                                    28.0   29.3   29.3   29.6   29.6   29.3                     .7  .7  .7  .7  .7  .7
```

IV. TEST RESULTS

The final step in implementing the equilibrium assignment procedure is software testing. The preliminary phases of this testing simply focused on whether the software is functioning properly and all the options are working. The preliminary software testing was performed using a small hypothetical seven-zone network and trip table. The intermediate testing used the network and trip table for one of the smaller urban areas. The final tests were performed using the network and zone system for a large urban area. The purpose of this chapter is to briefly summarize these final equilibrium assignment test results and to compare these equilibrium results to those obtained using the current Texas capacity restraint procedure.

DATA BASE

The 1985 networks and trip tables for the Houston-Galveston region were selected as the data base for the final test applications. The networks and trip tables were developed in a cooperative effort between the Houston-Galveston Area Council (H-GAC), TxDOT, the Harris County Metropolitan Transit Authority (METRO), and the Texas Transportation Institute (TTI). The Houston-Galveston study area is an eight-county area (i.e., Harris County and the seven adjacent counties) of 7,809 square miles with a 1985 population of 3,579,797. The current Houston-Galveston travel models were developed and calibrated for the 1985 base year. The highway analysis zone structure for the region consists of 2,598 zones and 2,643 external stations (i.e., 2,643 centroids). The 1985 highway network consists of 9,025 link data cards (excluding centroid connectors) representing 5,378.1 centerline miles of highway system comprising 15,320.4 lane miles of highways. The network and zone structure essentially focus on Harris County. Harris County is 1,723 square miles with a 1985 population of 2,723,888. Harris County is represented by 1,539 highway analyses zones. The Harris County portion of the 1985 highway network is 5,880 links representing 2,499.5 centerline miles of system comprising 8,461.2 lane miles.

24-HOUR ASSIGNMENT RESULTS

The 1985 Houston-Galveston network has 24-hour counted volume estimates on all links except centroid connectors. With this data available, both the equilibrium assignment results and the current capacity restraint results can be compared to count data. The ASSIGN SELF-BALANCING routine was used to perform two 1985 24-hour equilibrium assignments for the Houston-Galveston region. Two equilibrium assignments were performed because the software provides the option of selecting one of two objective functions. The first application used the default objective function which was described mathematically in Chapter II as:

Find the value of λ_k to:

$$Minimize{:}\sum_a \int_0^{V_{ak}} S_a(x)dx$$

Where:

$$V_{ak}=(1-\lambda_k)V_{a(k-1)}+\lambda_k W_a$$

Subject to various constraints (described in Chapter II)

This default objective function will be referred to as "Objective Function 1" in subsequent discussions. The second application used the alternative objective function which is described mathematically in Chapter II as:

Find the value of λ_k to:

$$Minimize{:}\sum_a V_{ak}S_a(V_{ak})$$

Where:

$$V_{ak}=(1-\lambda_k)V_{a(k-1)}+\lambda_k W_a$$

Subject to various constraints (described in Chapter II)

This alternative objective function used in the second equilibrium application will be referred to "Objective Function 2" in the subsequent discussions.

Both equilibrium applications used six iterations so that their results could be compared to the capacity restraint results using the current Texas capacity restraint procedure. Like the Texas Model, the equilibrium assignment tests used the impedance adjustment function implemented in the Texas capacity restraint procedure in 1979. Since all three applications used the same network, trip table, impedance adjustment function, and number of iterations, their results relative to observed estimates are readily comparable. With all these parameters held constant, the differences in the assignment results can be attributed directly to the differences in the iteration weights (i.e., the user-supplied weights versus the iteration weights computed by the equilibrium technique).

In the current capacity restraint applications of the Houston-Galveston region, the following iteration weights are used: 10, 10, 20, 20, 20, and 20 percent. Table IV-1 summarizes the lambda values estimated as a part of the two equilibrium applications. These resulting lambda values can be used to estimate the equivalent iteration weights. Table IV-2 summarizes the equivalent iteration weights used for the three assignments. As previously noted, the differences in the iterations weights account for the differences in the assignment results for the three assignments.

Table IV-3 summarizes the results of the first equilibrium assignment which used Objective Function 1 (i.e., the default objective function). Table IV-4 summarizes the results of the second equilibrium assignment which used Objective Function 2 (i.e., the alternative objective function). Finally, Table IV-5 summarizes the of the current Texas capacity restraint application using iteration weights of: 10, 10, 20, 20, 20, and 20 percent. As may be observed, all three applications produced assignment results which compare favorably to the counted volume data. Based on previous research (7, 8), it is not surprising that the equilibrium did not substantially improve the assignment results relative to counts.

PEAK-PERIOD ASSIGNMENT RESULTS

None of the peak-period models implemented in Texas have a network with extensive count volumes for comparison. The assignment results using the new equilibrium option in the PEAK CAPACITY RESTRAINT routine can be compared only to results obtained using the current capacity restraint in the PEAK CAPACITY RESTRAINT routine. The 1985 morning peak-hour trip table and peak-hour network were selected for the final tests of the equilibrium option in the PEAK CAPACITY RESTRAINT routine. As with the 24-hour assignments, two equilibrium assignments were performed using the two available objective functions.

Both peak-hour equilibrium applications used six iterations so that their results could be compared to the peak-hour capacity restraint results using the current Texas Procedure. The equilibrium assignment test applications used the impedance adjustment function implemented in the Texas procedure in 1979 to facilitate the comparisons. Since all three applications used the same network, trip table, impedance adjustment function, and number of iterations, the differences in the peak-hour assignment results can be attributed directly to the differences in the iteration weights (i.e., the user-supplied weights versus the iteration weights computed by the equilibrium technique).

In the normal peak-hour capacity restraint applications of the Houston-Galveston region, the following iteration weights are used: 10, 10, 20, 20, 20, and 20 percent. Table IV-6 summarizes the lambda values estimated as a part of the two equilibrium applications. These resulting lambda values can be used to estimate the equivalent iteration weights. Table IV-7 summarizes the equivalent iteration weights used for the three assignments. As previously noted, the differences in the iteration weights account for the differences in the assignment results for the three assignments.

Tables IV-8 and IV-9 summarize results from the equilibrium peak-hour assignments using Objective Functions 1 and 2. The results using the current Texas Capacity Restraint Assignment are summarized in Table IV-10 for comparison. As expected, all three provide similar results.

CONCLUSIONS AND RECOMMENDATIONS

The equilibrium assignment procedure implemented in the Texas Package appears to provide reasonable capacity restraint results using either objective. The equilibrium results were similar to the current Texas Model results. The results suggest that the equilibrium technique works well with congested networks; in view of emerging EPA requirements, it is the recommended technique for the larger areas in Texas. The equilibrium procedure should probably be applied for the base year data (with counts) for the smaller areas and the results compared with the current Texas capacity restraint assignment results. If equivalent, then the equilibrium assignment procedure would be recommended.

The test results provide an opportunity to observe the impact of the two objective functions. Both functions seemed to provide good results relative to the counts. Objective Function 2 is easier to explain and is consistent with the TRANPLAN software. For these reasons the Objective Function 2 was considered the more desirable of the two.

TABLE IV-1
Estimated Lambda Values from the
Two 24-Hour Equilibrium Applications

Lambda	Lambda Estimates	
	Objective Function 1	Objective Function 2
λ_2	0.41630	0.40835
λ_3	0.21759	0.21412
λ_4	0.28597	0.27683
λ_5	0.17132	0.15905
λ_6	0.22204	0.20964

TABLE IV-2
Equivalent Iteration Weights for the
Three 24-Hour Assignments

Iteration Number	Current Texas Application	Equlibrium Assignments	
		Objective Function 1	Objective Function 2
1	10%	21.022%	22.349%
2	10%	14.993%	15.425%
3	20%	10.016%	10.292%
4	20%	18.436%	18.399%
5	20%	13.328%	12.570%
6	20%	22.204%	20.964%

TABLE IV-3
Summary of 24-Hour Equilibrium Assignment Results
Using Objective Function 1

	HARRIS COUNTY	SURROUNDING SEVEN COUNTIES	EIGHT COUNTY REGION
FREEWAYS AND EXPRESSWAYS			
NONDIRECTIONAL LINKS	407.	146.	553.
CENTERLINE MILES	226.07	166.96	393.03
ASSIGNED VMT	27306512.4	5748276.8	33054789.2
% OF COUNTED VMT	104.3%	99.1%	103.4%
AVG. LINK VOLUME	129328.1	36697.7	104872.3
AVG. DIFFERENCE	6514.1	-231.1	4733.2
AVG. % DIFFERENCE	5.0%	-0.6%	4.5%
PERCENT RMSE	16.0%	14.0%	17.1%
PRINCIPAL ARTERIALS			
NONDIRECTIONAL LINKS	1264.	525.	1789.
CENTERLINE MILES	448.95	301.59	750.54
ASSIGNED VMT	10496715.7	3691042.5	14187758.2
% OF COUNTED VMT	99.8%	87.0%	96.1%
AVG. LINK VOLUME	21598.6	13871.7	19331.1
AVG. DIFFERENCE	-484.8	-2433.8	-1056.8
AVG. % DIFFERENCE	-2.2%	-17.5%	-5.5%
PERCENT RMSE	40.6%	34.8%	39.9%
MINOR ARTERIALS			
NONDIRECTIONAL LINKS	3368.	1062.	4430.
CENTERLINE MILES	1345.85	673.64	2019.49
ASSIGNED VMT	16868314.3	4112132.3	20980446.6
% OF COUNTED VMT	101.4%	94.4%	100.0%
AVG. LINK VOLUME	13540.2	6660.4	11890.9
AVG. DIFFERENCE	745.6	-423.0	465.5
AVG. % DIFFERENCE	5.5%	-6.4%	3.9%
PERCENT RMSE	52.0%	61.8%	54.2%
COLLECTORS AND OTHER			
NONDIRECTIONAL LINKS	841.	1412.	2253.
CENTERLINE MILES	479.21	1736.49	2215.70
ASSIGNED VMT	1693488.3	3648661.3	5342149.6
% OF COUNTED VMT	100.3%	98.8%	99.2%
AVG. LINK VOLUME	4268.2	2557.9	3196.3
AVG. DIFFERENCE	99.8	-47.9	7.2
AVG. % DIFFERENCE	2.3%	-1.9%	0.2%
PERCENT RMSE	87.9%	76.3%	85.8%
ALL LINKS			
NONDIRECTIONAL LINKS	5880.	3145.	9025.
CENTERLINE MILES	2500.08	2878.68	5378.76
ASSIGNED VMT	56365030.7	17200113.0	73565143.7
% OF COUNTED VMT	102.5%	95.1%	100.6%
AVG. LINK VOLUME	21960.9	7416.7	16892.6
AVG. DIFFERENCE	788.0	-581.3	310.8
AVG. % DIFFERENCE	3.6%	-7.8%	1.8%
PERCENT RMSE	39.9%	48.2%	43.3%

TABLE IV-4
Summary of 24-Hour Equilibrium Assignment Results
Using Objective Function 2

	HARRIS COUNTY	SURROUNDING SEVEN COUNTIES	EIGHT COUNTY REGION
FREEWAYS AND EXPRESSWAYS			
NONDIRECTIONAL LINKS	407.	146.	553.
CENTERLINE MILES	226.07	166.96	393.03
ASSIGNED VMT	27306570.9	5751068.1	33057639.1
% OF COUNTED VMT	104.3%	99.1%	103.4%
AVG. LINK VOLUME	129306.1	36720.6	104862.2
AVG. DIFFERENCE	6488.0	-208.1	4720.1
AVG. % DIFFERENCE	5.0%	-0.6%	4.5%
PERCENT RMSE	16.0%	13.9%	17.0%
PRINCIPAL ARTERIALS			
NONDIRECTIONAL LINKS	1264.	525.	1789.
CENTERLINE MILES	448.95	301.59	750.54
ASSIGNED VMT	10514791.1	3689348.9	14204140.0
% OF COUNTED VMT	99.9%	86.9%	96.2%
AVG. LINK VOLUME	21635.2	13868.0	19355.8
AVG. DIFFERENCE	-448.2	-2437.5	-1032.0
AVG. % DIFFERENCE	-2.1%	-17.6%	-5.3%
PERCENT RMSE	40.7%	34.9%	40.0%
MINOR ARTERIALS			
NONDIRECTIONAL LINKS	3368.	1062.	4430.
CENTERLINE MILES	1345.85	673.64	2019.49
ASSIGNED VMT	16853761.9	4112336.3	20966098.2
% OF COUNTED VMT	101.4%	94.4%	99.9%
AVG. LINK VOLUME	13531.1	6661.4	11884.2
AVG. DIFFERENCE	736.8	-421.8	459.1
AVG. % DIFFERENCE	5.4%	-6.3%	3.9%
PERCENT RMSE	52.0%	61.8%	54.1%
COLLECTORS AND OTHER			
NONDIRECTIONAL LINKS	841.	1412.	2253.
CENTERLINE MILES	479.21	1736.49	2215.70
ASSIGNED VMT	1694490.8	3647090.6	5341581.5
% OF COUNTED VMT	100.3%	98.7%	99.2%
AVG. LINK VOLUME	4271.6	2557.0	3197.0
AVG. DIFFERENCE	101.4	-48.9	7.2
AVG. % DIFFERENCE	2.4%	-1.9%	0.2%
PERCENT RMSE	87.7%	76.4%	85.7%
ALL LINKS			
NONDIRECTIONAL LINKS	5880.	3145.	9025.
CENTERLINE MILES	2500.08	2878.68	5378.76
ASSIGNED VMT	56369614.8	17199843.9	73569458.7
% OF COUNTED VMT	102.5%	95.1%	100.6%
AVG. LINK VOLUME	21962.5	7417.1	16893.8
AVG. DIFFERENCE	789.3	-580.9	311.8
AVG. % DIFFERENCE	3.6%	-7.8%	1.8%
PERCENT RMSE	39.9%	48.2%	43.3%

TABLE IV-5
Summary of 24-Hour Assignment Results
Using Current Texas Capacity Restraint Model

	HARRIS COUNTY	SURROUNDING SEVEN COUNTIES	EIGHT COUNTY REGION
FREEWAYS AND EXPRESSWAYS			
NONDIRECTIONAL LINKS	407.	146.	553.
CENTERLINE MILES	226.07	166.96	393.03
ASSIGNED VMT	27205417.6	5745845.0	32951262.5
% OF COUNTED VMT	103.9%	99.0%	103.0%
AVG. LINK VOLUME	128805.3	36684.9	104484.2
AVG. DIFFERENCE	5961.2	-243.9	4322.9
AVG. % DIFFERENCE	4.6%	-0.7%	4.1%
PERCENT RMSE	15.9%	14.2%	17.0%
PRINCIPAL ARTERIALS			
NONDIRECTIONAL LINKS	1264.	525.	1789.
CENTERLINE MILES	448.95	301.59	750.54
ASSIGNED VMT	10518224.5	3685175.9	14203400.4
% OF COUNTED VMT	100.0%	86.8%	96.2%
AVG. LINK VOLUME	21622.2	13796.0	19325.5
AVG. DIFFERENCE	-461.5	-2509.5	-1062.5
AVG. % DIFFERENCE	-2.1%	-18.2%	-5.5%
PERCENT RMSE	40.7%	34.7%	40.0%
MINOR ARTERIALS			
NONDIRECTIONAL LINKS	3368.	1062.	4430.
CENTERLINE MILES	1345.85	673.64	2019.49
ASSIGNED VMT	16923390.3	4116115.5	21039505.8
% OF COUNTED VMT	101.8%	94.5%	100.3%
AVG. LINK VOLUME	13570.2	6661.9	11914.1
AVG. DIFFERENCE	776.3	-421.1	489.2
AVG. % DIFFERENCE	5.7%	-6.3%	4.1%
PERCENT RMSE	51.6%	62.3%	53.8%
COLLECTORS AND OTHER			
NONDIRECTIONAL LINKS	841.	1412.	2253.
CENTERLINE MILES	479.21	1736.49	2215.70
ASSIGNED VMT	1699908.4	3662221.7	5362130.2
% OF COUNTED VMT	100.7%	99.2%	99.6%
AVG. LINK VOLUME	4282.8	2571.1	3210.0
AVG. DIFFERENCE	118.9	-34.6	22.7
AVG. % DIFFERENCE	2.8%	-1.3%	0.7%
PERCENT RMSE	88.8%	76.5%	86.4%
ALL LINKS			
NONDIRECTIONAL LINKS	5880.	3145.	9025.
CENTERLINE MILES	2500.08	2878.68	5378.76
ASSIGNED VMT	56346940.8	17209358.1	73556298.9
% OF COUNTED VMT	102.4%	95.1%	100.6%
AVG. LINK VOLUME	21949.1	7409.9	16882.5
AVG. DIFFERENCE	775.1	-588.0	300.1
AVG. % DIFFERENCE	3.5%	-7.9%	1.8%
PERCENT RMSE	39.7%	48.4%	43.2%

TABLE IV-6
Estimated Lambda Values from the Two Peak-Hour Equilibrium Applications

Lambda	Lambda Estimates	
	Objective Function 1	Objective Function 2
λ_2	0.12500	0.12500
λ_3	0.25000	0.25000
λ_4	0.24594	0.23867
λ_5	0.11872	0.12279
λ_6	0.22588	0.20365

TABLE IV-7
Equivalent Iteration Weights for the Three Peak-Hour Assignments

Iteration Number	Current Texas Application	Equlibrium Assignments	
		Objective Function 1	Objective Function 2
1	10%	33.760%	34.902%
2	10%	4.986%	4.986%
3	20%	12.861%	13.296%
4	20%	16.779%	16.672%
5	20%	9.190%	9.779%
6	20%	22.588%	20.365%

TABLE IV-8
Summary of Peak-Hour Equilibrium Assignment Results
Using Objective Function 1

	HARRIS COUNTY	SURROUNDING SEVEN COUNTIES	EIGHT COUNTY REGION
FREEWAYS AND EXPRESSWAYS			
DIRECTIONAL LINKS:	814.	292.	1106.
CENTERLINE MILES:	226.07	166.96	393.03
ASSIGNED VMT:	2733336.2	499723.1	3233059.2
AVG. LINK VOLUME:	6416.0	1662.5	5161.0
PRINCIPAL ARTERIALS			
DIRECTIONAL LINKS:	2392.	1030.	3422.
CENTERLINE MILES:	448.95	301.59	750.54
ASSIGNED VMT:	1016644.9	352224.3	1368869.2
AVG. LINK VOLUME:	1090.5	652.2	958.6
MINOR ARTERIALS			
DIRECTIONAL LINKS:	6735.	2097.	8832.
CENTERLINE MILES:	1345.85	673.64	2019.49
ASSIGNED VMT:	1665375.0	394298.2	2059673.2
AVG. LINK VOLUME:	658.1	319.5	577.7
COLLECTORS AND OTHER			
DIRECTIONAL LINKS:	1682.	2824.	4506.
CENTERLINE MILES:	479.21	1736.49	2215.70
ASSIGNED VMT:	191661.7	365985.2	557646.8
AVG. LINK VOLUME:	228.2	128.2	165.5
ALL LINKS			
DIRECTIONAL LINKS:	11623.	6243.	17866.
CENTERLINE MILES:	2500.08	2878.68	5378.76
ASSIGNED VMT:	5607017.7	1612230.7	7219248.4
AVG. LINK VOLUME:	1088.1	350.7	830.4

TABLE IV-9
Summary of Peak-Hour Equilibrium Assignment Results
Using Objective Function 2

	HARRIS COUNTY	SURROUNDING SEVEN COUNTIES	EIGHT COUNTY REGION
FREEWAYS AND EXPRESSWAYS			
DIRECTIONAL LINKS:	814.	292.	1106.
CENTERLINE MILES:	226.07	166.96	393.03
ASSIGNED VMT:	2746072.9	500066.9	3246139.8
AVG. LINK VOLUME:	6442.9	1663.7	5181.1
PRINCIPAL ARTERIALS			
DIRECTIONAL LINKS:	2392.	1030.	3422.
CENTERLINE MILES:	448.95	301.59	750.54
ASSIGNED VMT:	1019448.8	352821.6	1372270.4
AVG. LINK VOLUME:	1093.8	652.8	961.1
MINOR ARTERIALS			
DIRECTIONAL LINKS:	6735.	2097.	8832.
CENTERLINE MILES:	1345.85	673.64	2019.49
ASSIGNED VMT:	1653434.6	393538.6	2046973.2
AVG. LINK VOLUME:	653.2	318.4	573.7
COLLECTORS AND OTHER			
DIRECTIONAL LINKS:	1682.	2824.	4506.
CENTERLINE MILES:	479.21	1736.49	2215.70
ASSIGNED VMT:	191776.5	365391.5	557168.1
AVG. LINK VOLUME:	227.6	128.0	165.2
ALL LINKS			
DIRECTIONAL LINKS:	11623.	6243.	17866.
CENTERLINE MILES:	2500.08	2878.68	5378.76
ASSIGNED VMT:	5610732.9	1611818.6	7222551.5
AVG. LINK VOLUME:	1087.8	350.4	830.1

TABLE IV-10
Summary of Peak-Hour Assignment Results
Using Current Texas Capacity Restraint Model

	HARRIS COUNTY	SURROUNDING SEVEN COUNTIES	EIGHT COUNTY REGION
FREEWAYS AND EXPRESSWAYS			
DIRECTIONAL LINKS:	814.	292.	1106.
CENTERLINE MILES:	226.07	166.96	393.03
ASSIGNED VMT:	2662740.9	501584.7	3164325.7
AVG. LINK VOLUME:	6247.8	1667.1	5038.4
PRINCIPAL ARTERIALS			
DIRECTIONAL LINKS:	2392.	1030.	3422.
CENTERLINE MILES:	448.95	301.59	750.54
ASSIGNED VMT:	1006962.3	345867.8	1352830.1
AVG. LINK VOLUME:	1092.0	639.6	955.8
MINOR ARTERIALS			
DIRECTIONAL LINKS:	6735.	2097.	8832.
CENTERLINE MILES:	1345.85	673.64	2019.49
ASSIGNED VMT:	1807543.6	396340.2	2203883.8
AVG. LINK VOLUME:	715.6	324.0	622.6
COLLECTORS AND OTHER			
DIRECTIONAL LINKS:	1682.	2824.	4506.
CENTERLINE MILES:	479.21	1736.49	2215.70
ASSIGNED VMT:	190463.5	355163.0	545626.6
AVG. LINK VOLUME:	234.1	125.3	165.9
ALL LINKS			
DIRECTIONAL LINKS:	11623.	6243.	17866.
CENTERLINE MILES:	2500.08	2878.68	5378.76
ASSIGNED VMT:	5667710.4	1598955.8	7266666.1
AVG. LINK VOLUME:	1110.8	349.0	844.6

REFERENCES

1. Eash, R.W., B.N. Janson and D.E. Boyce. Equilibrium Trip Assignment: Advantages and Implications for Practice. Transportation Research Board, Transportation Research Record 728, 1979, pp. 1-78.
2. UTPS Reference Manual. Urban Mass Transportation Administration; Federal Highway Administration, 1977.
3. TRANPLAN User Manual - Version 7.0. Distributed by the Urban Analysis Group, Danville, CA.
4. Benson, J.D. and W.D. Cunagin. Development and Implementation of a New Impedance Adjustment Function for Capacity Restraint Traffic Assignment. Texas Transportation Institute, College Station, TX, February 1980.
5. Wardrop, J.G. Some Theoretical Aspects of Road Traffic Research. Proc., Institution of Civil Engineers, Part 2, Vol. 1, 1952, pp. 325-378.
6. Bell. C.E. and A. Horton. Program Documentation Manual for the Texas Large Network Assignment Models. Texas Transportation Institute, College Station, TX, August 1986.
7. Benson, J.D. and G.B. Dresser. A Comparative Analysis of Two Capacity Restraint Assignment Models Used in Texas. Research Report 1153-5. Texas Transportation Institute, College Station, TX, November 1992.
8. Chang, D.M. and G. B. Dresser. A Comparison of Traffic Assignment Techniques. Research Report 1153-3. Texas Transportation Institute, College Station TX, August 1990.
9. TRANPLAN Version 7.0 Source Code. Developed and distributed by the Urban Analysis Group, Danville, CA.

APPENDIX A: NEW OPERATING INSTRUCTIONS FOR THE ASSIGN SELF-BALANCING ROUTINE

PEAK CAPACITY RESTRAINT: Program Function

The function of this program is to produce a multi-path assignment in which the assigned volumes are in relative balance with traffic counts or else conform to link directional capacities. This is an iterative technique which adjusts directional link impedances to obtain the desired balance. There are two methods of obtaining the iteration weights. The methods are user-input weights or the equilibrium assignment method of weighting iterations. The program produces several cross-classification tables and comparison tables to indicate how well the above function is being achieved.

Data Set References

Input:	$NETWORK	(1)	Directional Network (contains network, lanes, previous assignments and iteration assignments) [1]
	$CTVOUT	(8)	Trip Matrix [1]
		(49)	Trees for COPY option
		(50)	Trees for SKIP option
		(3)	Directional assigned volumes and turn volumes from previous iterations [2]
Scratch:		(48)	Table look-up for impedance update functions and integrals by functional class
		(83)	Sort input
		(84)	Sort output
		SORTWK1	Sort work unit
		SORTWK2	Sort work unit
		SORTWK3	Sort work unit
Output:		SORTMSG	Sort message print file
	$NEWNET	(9)	Directional Network (updated for next iter.) [1]

[1] Default unit number

[2] Required as input when the First Iteration Number is not 1.

$SEPARAT	(20)	Separation Matrix [1]
$ROUTE	(25)	Route Profile [1]
	(SELTRP)	Selected interchanges [3]
	(3)	Directional assigned volumes and turn volumes from all iterations run
	(50)	Trees
	(85)	Nondirectional Network Data Set
	(4)	Scratch [4]
	(4)	Save Parameter cards
	(11)	Save Parameter data
$SUBAREA	(0)	Subarea Network Abstract (optional)[4]

Program Call Card

Column	Contents
1-24	$PEAK CAPACITY RESTRAINT
25-32	blank
33-80	ignored

Parameter Cards and Data Cards

Weights Card

This card provides the user the opportunity of electing one of the following options:

[3] DDName, this data set is written only if an iteration is run which has its iteration number equal to the Selected Links Iteration number.

[4] If Subarea Assignment is used, this unit must be defined.

1. Specifying the iteration weightings (i.e., percent each iteration is to be weighted) for the computation of the final weighted (or combined) assignment.

2. Allowing the program to compute the iteration weightings using the equilibrium technique.

If option 1 is elected, an iteration weight (i.e., an integer percent) must be specified for each iteration to be executed (i.e., Iteration 1 through the Maximum Iteration Number) such that the sum of the iteration weights is equal to exactly 100. If option 2 is selected, the iteration weight fields are ignored.

It is best to specify percents to load, or one or two iterations can be dropped from the weighted assignment. For 5 iterations, 15, 15, 20, 20, 30 percent are recommended.

Column	Contents
1-4	*WGT
3-6	blank
7-12	Percent weight for iteration 1
13-18	Percent weight for iteration 2
19-24	Percent weight for iteration 3
25-30	Percent weight for iteration 4
31-36	Percent weight for iteration 5
37-42	Percent weight for iteration 6
43-48	Percent weight for iteration 7
49-54	Percent weight for iteration 8
55-80	ignored

Turn Penalty Card

This card not only specifies the amount of time to be added as a penalty for each turn but provides the user the opportunity to elect a number of options available in the program. These options include:

- Specify the use of capacities rather than counts in the balancing process.
- Limit or extend the maximum number of iterations to be performed (default value is 5 iterations).
- Stop and restart the program between iterations (or between the list iteration and the weighted assignment) thereby allowing the program to be executed as a series of jobs.
- Selected Link output may be obtained from one iteration during a run. NOTE: if the preceding multiple run option is used, Selected Link output may be obtained from each run).
- Restart the program in case a machine problem is encountered during the iteration process.
- Select the equilibrium option for producing iteration weights.
- Allows the user the option of obtaining an additional all-or-nothing assignment using weight impedances.

The format for the turn penalty card is:

Column	Contents
1-5	*TURN
6	blank
7-12	Turn penalty in hundredths of a minute (a decimal point is assumed between Columns 10 and 11). Not used if *TRN card image is included.
13-16	Optional field which may be used to take advantage of a partially built TREES data set. If the field contains the word "COPY", the partially built TREES data set will be read from Unit 49 and copied to Unit 50. Missing trees and trees containing invalid codes will be rebuilt and

	included in the TREES data set on Unit 50. (NOTE: It is important that the First Iteration Number parameter in Columns 39-42 specifies the iteration number in which the partial TREES data set on Unit 49 was built.)
17-30	blank
31-34	<u>CAP</u> - to perform a capacity restraint assignment using the old impedance function.
	<u>NCAP</u> - to perform a capacity restraint assignment using the new impedance function.
	<u>EQLB</u> - to perform an Equilibrium capacity restraint assignment. The additional cards *EQLIB, *FCLCRV, *FCLPARM, and an *END card are read. The values on the *WGT card are ignored.
	blank - to perform a volume restraint assignment using input traffic count data.
35-37	WGT - to produce an all-or-nothing traffic assignment using weighted impedances; otherwise, this assignment is not produced.
38	blank
39-42	First Iteration number (default value = 1).
43-48	Stop Iteration number (default value = 0).
49-54	Minimum Iteration Number (default value = 3).
55-60	Maximum Iteration Number (default value = 5).
61-66	Selected Links Iteration Number (default value = 0).
67-68	blank
69-72	SKIP to skip building trees on the First Iteration Number and use the trees input on Unit 50 for this iteration. These trees will be destroyed if another iteration or the second weighted assignment is run. <u>This</u>

	option cannot be used with a Subarea PEAK CAPACITY RESTRAINT run.
73-80	blank

The parameters in Columns 39-66 are used as follows:

First Iteration Number: This parameter is used when a PEAK CAPACITY RESTRAINT run is to be continued from another job. This parameter should be set to one more than the previous number of iterations run. If it is greater than the Maximum iteration number, then the weighted iteration will be the next one run. If it is zero or blank, it will default to 1. If this parameter is greater than 1, then the last updated NETWORK data set and the data set from Unit 3 are required from the previous PEAK CAPACITY RESTRAINT run.

Stop Iteration Number: This parameter can be used to stop the ASSIGN SELF-BALANCING process at the end of any iteration except the weighted iteration. If it is left blank or zero, the process will run to completion.

Minimum Iteration Number: If the user elects the option whereby the iteration weights are specified on the Weights Card, the value parameter will be automatically set equal to the Maximum Iteration Number (regardless of the number in Columns 49-54) and subsequently ignored. If the user elects the option whereby the iteration weights are computed using a regression technique, the actual number of iterations performed may vary between the Minimum and Maximum Iteration Numbers. (NOTE: Normally the maximum number of iterations will be performed regardless of this parameter since the termination of the iterative process prior to the Maximum Iteration Number requires the T value, computed from the regression analysis, to be less than or equal to 1.96 .) Under the regression weighting option, the parameter defaults to 3 if the field in the card is blank or zero.

Maximum Iteration Number: This is the maximum iteration number which will be run. If it is zero or blank, it defaults to 5. This parameter must not be greater than 8.

Selected Links Iteration Number: This is the iteration number on which the selected links parameter cards following the last *TREE card will be read, and the SELTRP data set will be written. If this output is not desired, then this field should be left blank and the selected links parameter cards will not be read.

New Turn Penalties

This card image specifies 16 turn penalties or turn prohibits. This card image specifies all combinations of the new link direction codes. The turn penalties must be in the range of 0 to 10.23 minutes. A turn prohibit is specified by -1 or any other negative value. The *TURN card image must still be present because it has several optional flags which are still examined. This card image is optional and should only be used if the 16 turn penalty/prohibit scheme is being used.

Column	Contents
1-4	*TRN
5-8	blank
9-12	Turn penalty from W to W in hundredths
13-16	Turn penalty from W to X in hundredths
17-20	Turn penalty from W to Y in hundredths
21-24	Turn penalty from W to Z in hundredths
25-28	Turn penalty from X to W in hundredths
29-32	Turn penalty from X to X in hundredths
33-36	Turn penalty from X to Y in hundredths
37-40	Turn penalty from X to Z in hundredths
41-44	Turn penalty from Y to W in hundredths
45-48	Turn penalty from Y to X in hundredths
49-52	Turn penalty from Y to Y in hundredths
53-56	Turn penalty from Y to Z in hundredths
57-60	Turn penalty from Z to W in hundredths
61-64	Turn penalty from Z to X in hundredths
65-68	Turn penalty from Z to Y in hundredths

69-72	Turn penalty from Z to Z in hundredths

Tree Selection Card [5]

This card image specifies those centroids from which trees will be built. One Tree Selection Card per subnetwork is required; a minimum of one to a maximum of 4,800 centroids is permissible. The Tree Selection Cards may specify various groups of one or more trees in any order, but proper functioning of the load process requires that the trees be built in a serial sequence order. In the format of the Tree Selection Card the six character fields are each composed of two subfields, A and B, of five characters and one character, respectively:

Column	Contents	
1-5	TREE [5]	
6-7	blank	
8-12	Subfield A	First selection field
13	Subfield B	
14-18	Subfield A	Second selection field
19	Subfield B	
20-24	Subfield A	Third selection field
25	Subfield B	
26-30	Subfield A	Fourth selection field
31	Subfield B	
32-36	Subfield A	Fifth selection field
37	Subfield B	
38-42	Subfield A	Sixth selection field
43	Subfield B	

[5] Tree Selection Cards are not read if unit $SUBAREA is used.

44-48	Subfield A	Seventh selection field
49	Subfield B	
50-54	Subfield A	Eighth selection field
55	Subfield B	
56-60	Subfield A	Ninth selection field
61	Subfield B	
62-66	Subfield A	Tenth selection field
67	Subfield B	
68-72	Subfield A	Eleventh selection field
73	Subfield B	

NOTE: A comma (,) may be entered in Column 73 if desired when subfield 68-72 is used. However, Columns 73-80 are not read; the program places the comma when entries are made in Columns 68-72.

Subfield A may contain any valid centroid number in the network. Subfield B functions as a delimiter and may contain a blank, comma, or period. Any other character will give an error message. A period used as a delimiter causes all trees built within the control range to have BCD Minimum Path Descriptions printed for inspection. This does not affect the tree building process otherwise, except it causes a delay due to writing the output.

In processing selection fields from left to right, the occurrence of two consecutive A subfields separated by a blank B subfield will initiate a control setup for inclusive tree building, beginning with the centroid specified by the first A subfield and ending with the centroid specified by the second A subfield. A comma in the second B subfield is optional for this situation, since the starting and ending centroids have been found for a search group. The occurrence of two successive B subfields containing either commas (may be implied as mentioned above) or periods, causes a single centroid to be specified; i.e., a control setup for inclusive tree building beginning and ending with the centroid specified in the intermediate A subfield.

For example, to build trees 1 through 90, with BCD Minimum Path Descriptions of trees 1 and 50, the following Tree Selection Card would be required:

Column	Contents
1-5	*TREE
6-7	bb
8-13	bbbb1.
14-19	bbbb2b
20-25	bbb49b
26-31	bbb50.
32-37	bbb51b
38-43	bbb90b

In the above example, lower case b's are used to indicate blanks.

Turn Suppression Option

The Turn Suppression Card denotes the assigned network output that is desired. Columns 7-80 are ignored.

Columns 1-6	Action
*ALL	Write link volumes, turn volumes, and the vehicle-hour and vehicle-mile summary.
*LINKS	Write link volumes and the vehicle-hour and vehicle-mile summary only.

The LIST Card, LINKS Cards, EQUAL Cards, and END Card (for EQUALS Cards not for Selected Link Cards) are read only if $SUBAREA is used.

If the SUBAREA file number is set to nonzero by a $SUBAREA, XX card, then a set of cards (i.e., LIST cards, DIRECT cards, and optional LINKS cards) which define the portion of the network within the subarea are read. These cards are:

List Card

This card (or cards) specifies a continuous ring of links (and/or centroid connectors) which completely surrounds the subarea. The LIST cards must be chained to the next

LIST card but must specify one complete ring of links. For error checking, it is convenient to keep the LIST cards in order. Each card may contain up to twelve node (or centroid) numbers. A link is defined between the first and the second number on this card; between the second and third number; etc. The first zero or blank field on this card ends the data on this card, which means that a LIST card can define from one to 11 links to be in the ring.

Column	Contents
1-4	LIST
5-6	blank
7-12	First node (or centroid) number
13-18	Second node (or centroid) number
19-24	Third node (or centroid) number
25-30	Fourth node (or centroid) number
31-36	Fifth node (or centroid) number
37-42	Sixth node (or centroid) number
43-48	Seventh node (or centroid) number
49-54	Eighth node (or centroid) number
55-60	Ninth node (or centroid) number
61-66	Tenth node (or centroid) number
67-72	Eleventh node (or centroid) number
73-78	Twelfth node (or centroid) number

Direct Card

This card specifies on which side of the ring of links the subarea lies.

Columns	Contents
1-6	DIRECT

7-12	Node on the subarea ring of links
13-18	Node that is inside the subarea is connected to the first node in Columns 7-12, and is further connected to another node in the subarea.

Links Card

These cards may define other links to be included in the summaries produced after the assigned network listing of the assignment output from a subarea assignment. This card will be necessary for any part of the network inside the ring of links which is isolated from the other part of the network inside the ring of links. The most likely use for this card is for centroid connectors where the centroid only connects to the nodes (or centroids) in the ring of links. This card also may define links which cannot be surrounded by the ring of links.

Column	Contents
1-5	LINKS
6	blank
7-12	First node (or centroid) number
13-18	Second node (or centroid) number
19-24	Third node (or centroid) number
25-30	Fourth node (or centroid) number
31-36	Fifth node (or centroid) number
37-42	Sixth node (or centroid) number
43-48	Seventh node (or centroid) number
49-54	Eighth node (or centroid) number
55-60	Ninth node (or centroid) number
61-66	Tenth node (or centroid) number
67-72	Eleventh node (or centroid) number

73-78	Twelfth node (or centroid) number

Last Centroid Card

Column	Contents
1-6	*LASTC
7-12	Last (i.e., highest) centroid number for trees
13-80	ignored

Equal Cards

These cards define the sector centroids and also the centroids which are equated to the sector centroids. Trees will not be built for centroids equated to sector centroids. The exception to this is that trees will be built for all zones equated to the same zone (sector centroids). An entry is generated for each sector centroid and if it is additionally equated to itself, a warning message will be printed. The numbers to the right of the EQUAL on these cards may specify ranges of centroids by placing the beginning of the range on the card in one field and immediately following this field by a field with the last centroid of the range with a minus sign in front of the centroid number. A range must all be specified on one EQUAL card.

Column	Contents
1-4	Sector Centroid Number
5	blank
6-10	EQUAL
11-80	14 fields for centroids or centroid ranges (Format: 14I5)

End Card

This card defines the end of the EQUAL cards.

Columns	Contents
1-5	blank

6-8	END
9-80	blank

Selected Links Parameter Cards

The following parameter cards are required only if the Selected Links Iteration Number is set equal to an iteration that is run.

Selected Link Cards

A Selected Link Card must be provided for each desired link. This card also limits the number of interchanges to print. If any of the limits are omitted or zero, it will be set for the maximum number permitted. None of the options should be set greater than the allowable maximum. After any one of the three limits has been reached, the output will be terminated.

Column	Contents
1-4	*SEL
5-6	blank
7-12	A-node
13-18	B-node
19-24	Percentage of total volume to be included (range 1 to 100)
25-30	Minimum two-way volume to be allowed (range 1 to 32,767)
31-36	Number of zone pairs to be included (range 1 to 32,767)
37-80	ignored

End Selected Links

Column	Contents
1-4	*END
5-80	blank

Equilibrium Optimization Option

Column	Contents	
1-6	*EQLIB	
8-11	IMPD	This option specifies that the optimization function is the sum of the link impedance multiplied by the assigned volumes for the links which have capacities. The default optimization function is the integral of the link impedance over the volume range of zero to the assigned link volume for links with capacities.

Functional Class Impedance Update Function Card

Column	Contents
1-8	*FCLPARM
11-15	Functional Class Number 0 to 15
21-30	"A" constant for this functional class; default is 0.92
31-40	"B" constant for this functional class; default is 0.15
41-50	"C" constant for this functional class; default is 4.0 and the maximum is 30

The default data or the parameters read are used to build the impedance update function value W for each directional link with an assigned volume using the following equation:

$$W_f = A_f + B_f[\frac{V}{C}]^{D_f}$$

The default impedance update function is:

$$W_f = 0.92 + 0.15[\frac{V}{C}]^4$$

Where:

W_f	=	the value multiplied by the initial impedance for functional class *f*.
V	=	the directional iteration weighted link volume.
C	=	the directional link capacity.

Functional Class Impedance Update Values Header Card

Column	Contents
1-7	*FCLCRV
11-15	Functional Class Number 0 to 15

This card defines the beginning of a table of values of V/C ratio and impedance update ratio. There can be from 2 to 400 input data cards. The input data cards must be in an increasing V/C ratio order and they must define an impedance update function which has increasing or equal values as the V/C ratio increases. The range of the V/C ratio is from 0.0 to 4.0.

Functional Class Impedance Update Values Header Card

Column	Contents
1-10	Volume to capacity ratio (0.0 to 4.0); must include a decimal point
11-20	Impedance update value (greater than zero)

End of Impedance Update Functions

Column	Contents
1-4	*END
5-80	blank

This data card is required after the end of the last impedance update function but is not required between impedance update functions.

Normal Operation

The Weights Card, the Turn Penalty Card, optionally the *TRN card with 16 turn penalties/prohibits, and the Tree Selection Cards are read and interpreted. The following messages are printed if these cards are correct:

THE TREE CARDS HAVE ESTABLISHED THE FOLLOWING PARAMETERS

TURN PENALTY = ---.---

LINK COUNTS WILL BE USED IN THE LINK IMPEDANCE FUNCTION

or LINK CAPACITIES WILL BE USED IN THE OLD LINK IMPEDANCE FUNCTION

or LINK CAPACITIES WILL BE USED IN THE NEW LINK IMPEDANCE FUNCTION

AN INCOMPLETE TREE DATA SET WILL BE READ FROM UNIT 49 AND MISSING TREES OR TREES WITH INVALID CODES WILL BE BUILT [6]

A SECOND WEIGHTED ASSIGNMENT USING THE WEIGHTED IMPEDANCES WILL BE PRODUCED [6]

FIRST ITERATION = -

STOP AFTER ITERATION = -

MINIMUM NUMBER OF ITERATIONS = -

[6] Optional

MAXIMUM NUMBER OF ITERATIONS = -

ITERATION TO GET SELECTED LINKS OUTPUT -

PERCENTS TO LOAD ARE -- -- -- -- --
THE PERCENTS CALCULATED FROM A REGRESSION WILL BE USED AS THE PERCENTS TO LOAD*

TREE BUILDING WILL BE SKIPPED AND A SET OF TREES PREVIOUSLY BUILT WILL BE USED TO ASSIGN ITERATION -

FOR SUBNETWORK - SEARCH MINIMUM PATHS FROM ZONES------ TO ------ INCLUSIVE AND OUTPUT
or SUPPRESS OUTPUT

If the equilibrium assignment option is read on the *TURN card, then the following message will be printed.

EQUILIBRIUM WILL MINIMIZE INTEGRAL VHT FUNCTION

or EQUILIBRIUM WILL MINIMIZE VEHICLE HOURS OF TRAVEL

The appropriate portions of the messages are repeated as often as necessary to describe the various control ranges for the tree search. The steps described below are repeated for each of three to five iterations.

The link impedances are updated using directional link volumes. Directional capacities and ground counts are also used. The initial network data set must be from an ASSEMBLE PEAK NETWORK.

The NETWORK data set is read. Minimum path trees are traced according to the Turn Penalty Card and Tree Selection Card, and the Trip Matrix is assigned to the minimum paths. When the iteration number is one, the entire assignment is printed. This will correspond to a minimum time path assignment if impedances have been calculated from speed and distance.

The variable C is set to either directional traffic counts or directional capacities, depending upon which of three options are entered on the Turn Penalty Card (Columns 31-34). If C is zero for a given link (again dependent on the option selected), the impedance for that link is not changed.

The five options available for updating link impedances are:

Traffic Counts

I(n + 1) = (0.75 + 0.25*(V/C))*I(n)

NOTE: If I(n - 1)/I(n) > 1.43, I(n + 1) is set to 1.43*I(n)

Capacities - OLD Impedance Function (CAP)

The link impedance is not changed in this option unless the link volume has exceeded capacity for one or more iterations. When the link impedance is updated, the following function is used:

I(n + 1) = (0.75 + 0.25*(V/C))*I(n)

NOTE: If I(n + 1)/I(n) > 1.43, I(n + 1) is set to 1.43*I(n)

Capacities - NEW Impedance Function (NCAP)

I(n + 1) = (0.92 + 0.15 (V(n)/C)**4)*I(1)

NOTE: If I(n + 1)/I(n) > (n + 1), I(n + 1) is set to (n + 1)*I(n)

The following constraints are applied for all updated impedances:

If I(n + 1) > 10.23 - turn penalty, I(n + 1) is set to 10.23 - turn penalty

If I(n + 1) = 0.0 and I(n) = 0.0, then I(n + 1) is set to 0.01

Capacities - Equilibrium Formula (EQLB)

I(n + 1) = (A(fc) + B(fc) (V(n)/C)**D(fc))*I(1)

Where:

A(fc)	=	a value read from the *FCLPARM card for functional class "fc" or a default value of 0.92.
B(fc)	=	a value read from the *FCLPARM card for functional class "fc" or a default value of 0.15.
D(fc)	=	a value read from the *FCLPARM card for functional class "fc" or a default value of 4.

NOTE: If I(n + 1)/I(n) > (n + 1), I(n + 1) is set to (n + 1)*I(n) the following constraints are applied for all updated impedances:

If I(n + 1) > 10.23 - turn penalty, I(n + 1) is set to 10.23 - turn penalty

If I(n + 1) = 0.0 and I(n) = 0.0, then I(n + 1) is set to 0.01

Capacities - Equilibrium Table Lookup (EQLB)

The new impedance value is obtained from interpolating a value from the link V/C ratio between two points from the table look up values input after a *FCLCRV table of impedance update factors. The new impedance is obtained by multiplying the table look-up value by the initial impedance.

NOTE: If I(n + 1)/I(n) > (n + 1), I(n + 1) is set to (n + 1)*I(n) the following constraints are applied for all updated impedances:

If I(n + 1) > 10.23 - turn penalty, I(n + 1) is set to 10.23 - turn penalty

If I(n + 1) = 0.0 and I(n) = 0.0, then I(n + 1) is set to 0.01

The new network containing the revised impedances is written on the NEWNET data set. The route profile and separation matrix data sets also are written. Data will also be included in the NEWNET data set from the last assignment (and all previous assignments up to a maximum of 19) describing directional link volumes and the corresponding directional link impedances. The following tables and summaries also are produced:

CROSS-CLASSIFICATION OF V/C FREQUENCIES FROM LAST TWO ASSIGNMENTS.[7]

CROSS-CLASSIFICATION OF LINK COUNTS BY V/C RATIO FROM LAST TWO ASSIGNMENTS.[7]

JURISDICTION SUMMARY (This table is not printed if there are functional class codes on the link data).

JURISDICTIONAL/FUNCTIONAL CROSS-CLASSIFICATION OF ASSIGNED VOLUMES (This table is not written if more than 95 percent of the links have no functional class code).

[7] Directional Table

JURISDICTIONAL/FUNCTIONAL CROSS-CLASSIFICATION OF COUNTED VOLUMES (This table is not written if more than 95 percent of the links have no functional class code or if all locations in the table are zero).

JURISDICTIONAL/FUNCTIONAL CROSS-CLASSIFICATION OF LINK CAPACITIES (This table is not written if more than 95 percent of the links have no functional class code or if all locations in the table are zero).

COMPARISON OF ASSIGNED VOLUMES WITH COUNTED VOLUMES[7]

COMPARISON OF ASSIGNED VOLUMES WITH LINK CAPACITIES[7]

COMPARISON OF ASSIGNED VOLUMES (from last assignment) WITH ASSIGNED VOLUMES FROM (assignment before last)[7]

ITERATION WEIGHTING - MULTIPLE REGRESSION ANALYSIS

The iteration number is printed in the Heading record of each of the above tables. Then the unit numbers of the NETWORK and NEWNET data sets are switched. If the calculated T value for the last entry in the ITERATION WEIGHTING-MULTIPLE REGRESSION ANALYSIS table is less than 1.96, and the minimum Iteration Number of iterations have been completed, the repetitions are terminated. If an iteration is run which is equal to the stop iteration number, the process stops at the end of that iteration. The maximum iterations allowed is the Maximum Iteration Number.

After the iterative process terminates, the B values from the final ITERATION WEIGHTING-MULTIPLE REGRESSION ANALYSIS table are multiplied by 100 (B(0) is disregarded). Any negative values are set to zero, and the integer portion of the other B values is selected after scaling the sum to 10 percent. If the *WGT card contained weight, these are used instead of the ones calculated. These values are then printed in a table entitled ITERATION WEIGHTS APPLIED.

A weighted assignment is calculated by applying the iteration weights (percentages) to their respective assigned volumes and summing. An updated Network data set is prepared which includes the weighted assigned volumes, and the weighted assignment is printed with all assigned volumes and turning movements. A set of printed tables similar to those already described for the individual iterations is written for the weighted assignment.

Using the same iteration weights applied with the assigned volumes, a set of directional weighted impedances is calculated in an analogous manner. If WGT option is specified on the *TURN Card, the weighted impedances are used for one final assignment, and everything written for the calculated weighted assignment is also produced for the weighted impedance assignment. All assigned volumes and turning movements are printed for the weighted impedance assignment. Finally, the Route Profile and Corridor Intercept tables are printed, followed by a table of assigned volumes and impedances from all iterations (including the weighted assignments). The last assignment produces new data sets on the NETWORK or NEWNET, SEPARAT, and ROUTE units. A message is written at the end indicating the unit number on which the final NETWORK data set has been written:

THE FINAL LOADED NETWORK IS ON --

If the PEAK CAPACITY RESTRAINT run is to be restarted, this is the data set which must be Unit NETWORK on the restart run.

When a machine malfunction or an Abnormal Program Termination occurs, the following things should be considered when setting up a restart. The Unit 3 data set, the Network data set and the First Iteration Number used in a restart of PEAK CAPACITY RESTRAINT must be consistent in respect to the number of iterations previously completed. The data for an iteration are written on Unit 3 before the assigned network is printed or if it is not printed, then before the Network data set is updated. The Unit 3 data set is closed and reopened after being written so that it will be complete. When PEAK CAPACITY RESTRAINT is restarted, Unit 3 is positioned to the end of the First Iteration Number -1 and partially checked for errors. If there are additional iterations or records written on this data set, they are ignored and written over.

The data written on a Network data set are written concurrently with the printing of the assigned network, or if the printing is skipped, then it is written before the first output following the assignment of the network is printed. The writing of the directional Network data set is an update process and there is an old Network data set and a new Network data set for each iteration. Each new Network data set is closed immediately after being written. The first Network data set used by the iteration indicated by First Iteration Number parameter is Unit NETWORK. At the end of this iteration, Unit NEWNET becomes the new Network data set. Units NETWORK and NEWNET alternately become the Network data sets used on succeeding iterations. When the PEAK CAPACITY RESTRAINT process ends through the Stop Iteration Number option, a message "THE FINAL LOADED NETWORK IS ON X" gives the Unit number of the new directional Network data set for the last iteration run. If the program ends abnormally, the Unit to use as the Network data set on a restart must be determined by the number of iterations which have completed the assignment process. If there is any doubt of which data set to use, NETWORK or NEWNET, then one or both of the data sets can be printed with the OUTPUT NETWORK program and the header records examined to determine the appropriate data set. A check should also be made to see that all links for the last node in the Network are printed. Also a network with nondirectional volumes is written to Unit 85. This data set is written when table L1 is produced.

Error Messages

INVALID TURN PENALTY OR TREE CARD READ

The program prints this line under the card which has an error in its identification field (Columns 1-6). The correct contents for Columns 1-6 for the two cards are *TURN and *TREE. The program stops with a STOP 0 after examining both cards for errors.

ILLEGAL FIELD SEPARATION CHARACTER IN TREE CARD

This message is printed if a field separation character is found which is not a comma, a period, or a blank. The program stops with a STOP 0 after it examines the rest of the Tree Card.

---- ERROR(S) DETECTED IN ABOVE PARAMETER CARDS, EXECUTION TERMINATED

The program stops with a STOP 0.

NO VALID OPTIONS ON *EQLIB CARD

Columns 8-11 of the *EQLIB card do not contain "EQLB". This counts as one error. The program will end with a STOP 9 after examining other parameter cards.

FOR FUNCTIONAL CLASS -- A CONSTANT IS TOO SMALL = --------.-----

The "A" impedance update value on the *FCLPARM card is less than or equal zero. The program will end with a STOP 9 after examining other parameter cards.

FOR FUNCTIONAL CLASS -- D CONSTANT IS TOO SMALL = --------.-----

The "D" impedance update value on the *FCLPARM card is less than or equal to zero. The program will end with a STOP 9 after examining other parameter cards.

FOR FUNCTIONAL CLASS -- D CONSTANT IS TOO LARGE = ------.-----

The "D" impedance update value on the *FCLPARM card is larger than 30. Values of 30 and above can produce overflows in calculating the power of the V/C ratio. The program will end with a STOP 9 after examining other parameter cards.

ERROR, FUNCTIONAL CLASS = -- IS INVALID ON "--------------------------------"

The functional class on a *FCLCRV data card is less than 0 or greater than 15. The program will end with a STOP 9 after examining other parameter cards.

FUNCTIONAL CLASS -- IMPEDANCE UPDATE FUNCTION ERROR
----- NON-INCREASING ERRORS FOR THE VCR INPUT

The V/C ratios on data cards following a *FCLCRV card are not in an increasing order. The program will end with a STOP 9 after examining other parameter cards.

FUNCTIONAL CLASS -- IMPEDANCE UPDATE FUNCTION ERROR
----- NON-INCREASING ERRORS FOR THE VALUE INPUT

The impedance multiplier on data cards following a *FCLCRV card are not in an increasing order. The program will end with a STOP 9 after examining other parameter cards.

------ ERRORS IN EQUILIBRIUM INPUT DATA CARDS, STOP 9, EXECUTION TERMINATED

This message gives a count of errors on the equilibrium data cards and the program executes a STOP 9.

**** TURN TYPE FOR NODE -----, ILLEGAL, = ----------, SET TO 28

This message should be printed only if an input error has occurred in reading the NETWORK data set. It indicated that no turning movements will be saved for the node in the message.

NODES .GT. 16000, = ------

This message is self-explanatory. The program stops with a STOP 0 at this point.

NUMBER OF SUBNETS FROM BINARY NETWORK = ---

NUMBER OF SUBNETS FROM VOLUME FILE = ---

NUMBER OF SUBNETS FROM TREE FILE = --- ARE NOT EQUAL, LOAD DELETED.

There is a conflict in the number of subnetworks in one of the three data sets. The program does not load the network, and ends with a STOP 7.

FIRST NODE, LAST CENTROID INFORMATION FROM TREE FILE AND VOLUME

FILES DOES NOT AGREE, OR ONE IS GREATER THAN ------ OR THE NUMBERS ARE NOT INCREASING ORDER LOAD DELETED.

```
FILE FSTND(1) LSTND(1) FSTND(2) LSTND(2) FSTND(3) LSTND(3) FSTND(4)
LSTND(4)

LOAD   -----    -----    -----    -----    -----    -----    -----    -----

TREE   -----    -----    -----    -----    -----    -----    -----    -----
```

The first number printed is the last centroid number of the last subnetwork from the Network data set. No centroid number from the trip matrix or Path data set can be larger than this number. Another condition for which the message is written is when the first node number for a subnetwork in the Trip Matrix data set is not equal to the corresponding first node number from the Network data set. These conditions both indicate that the Path, Trip Matrix, and Network data sets are not for the same network. The program continues after this message is printed.

```
NUMBER OF TURNING MOVEMENTS TO SAVE EXCEED 60000, = -----
```

This message indicates that more than 60,000 locations in the turning movement table are needed to load the network. The actual number of locations needed is printed after the comma. The program ends with a STOP 50 after this message. One way to bypass this problem is to use the LOAD SELECTED LINKS program with the option to output the loaded network without the turning movements.

```
I/O ERROR ON TREE FILE AFTER A NODE -----
```

This indicates that there was a possible error on reading a path record. The program continues and attempts to use the path record.

```
I/O ERROR ON TRIP FILE AFTER A NODE -----
```

The program skips the Trip Matrix record and continues.

```
ABEND U101
```

This abend code indicates that one of the turn type code numbers, which has a valid range of from 1 to 28, has been destroyed during the load process. The job abended without a dump. The probable cause of this error would be an I/O error from the Path data set.

```
ABEND U102
```

This error code could indicate several things, some of which are: (1) the three data sets which are input to this program are not for an identical network, (2) there has been an I/O error in reading one of the data sets, or (3) a CPU error has occurred. The job is abended without a dump.

TRMV ERR

This error message is printed in the loaded network output if one or more of the turning movements cannot be calculated. This message indicates that the network has not been loaded correctly. This message could occur if a network which contains errors is assigned.

MORE THAN TWO LINKS FROM NODE -----

This message occurs during the processing of the Route Profiles and indicates that a node has three or more links with the same route code. Only two of the links will be retained. This will cause links to be lost in the Route Profile output for this route.

ROUTE -- HAS NO ENDS, A LINK WILL BE CHOSEN AS A STARTING POINT.

The links included in this route form a closed loop. The link with the lowest node number will be chosen as the starting point for the Route Profile output.

ERROR ON UNIT 3, EXECUTION TERMINATED WITH STOP 21.

There was an error in reading the parameter record from Unit 3 when the First Iteration Number on the *TURN card was not 1. The program is terminated with STOP 21.

ERROR IN PARAMETER RECORD FROM UNIT 3, EXECUTION TERMINATED WITH STOP 21 PARAMETER RECORD.

There was an error in the values in the parameter record. The values in the parameter record are printed. The program execution is terminated with STOP 21.

ERROR, MISSING RECORDS OR I/O ERROR ON UNIT 3, ------ RECORDS READ ------ RECORDS WERE TO BE READ EXECUTION TERMINATED WITH STOP 21.

PARAMETER RECORD

The number of records to skip position Unit 3 is determined from the values of the parameter record and the number of iterations previously run. Then the program tries to skip this number of records on Unit 3. If the end of data set condition or an I/O error is encountered, then this message is printed. This message will be printed when the First Iteration Number is set larger than one more than the previous number of iteration run.

INVALID *WGT CARD READ

The card following the program call card does not contain *WGT in Columns 1-4.

ERROR IN ABOVE PERCENTS, SUM IS ----

The percents in the *WGT card for iterations 1 to the Maximum Iteration Number do not sum to 100. The sum is printed.

REGRESSION CALCULATION OF ITERATION WEIGHTINGS IS NOT ALLOWED; USE THE EQUILIBRIUM OPTION

The old option of calculating the iteration weighting percents has been removed. The user should use the new equilibrium option for calculating the weightings.

SORT ERROR; LINK READ FROM NET IS ----- ----- LINK READ FROM SORT IS ----- -----

The Anode-Bnode of the link read from the sort did not match the expected Anode-Bnode of the link being processed. This is probably a sort error or a JCL error. The Anode and Bnode of the expected link are printed in the first two dash fields. The Anode and Bnode of the link from the sort are printed in the third and fourth dash fields. Up to three sort records are used for each link. The first sort record contains assignments 1 through 6 (including the lanes as the first assignment).

SORT ERROR IN SECOND SORT RECORD; LINK READ FROM NET IS ----- ----- LINK READ FROM SORT IS ----- -----

The Anode-Bnode of the link read from the sort did not match the expected Anode-Bnode of the link being processed. This is probably a sort error or a JCL error. The Anode and Bnode of the expected link are printed in the first two dash fields. The Anode and Bnode of the link from the sort are printed in the third and fourth dash fields. Up to three sort records are used for each link. The second sort record is written when 7 assignments have been done (including the lanes as the first assignment).

SORT ERROR IN THIRD SORT RECORD; LINK READ FROM NET IS ----- ----- LINK READ FROM SORT IS ----- -----

The Anode-Bnode of the link read from the sort did not match the expected Anode-Bnode of the link being processed. This is probably a sort error or a JCL error. The Anode and Bnode of the expected link are printed in the first two dash fields. The Anode and Bnode of the link from the sort are printed in the third and fourth dash fields. Up to three sort records are used for each link. The third sort record is written when 15 assignments have been done (including the lanes as the first assignment).

STOPPING FOR 100 OR MORE SORT ERRORS WITH STOP 9

This error message indicates that there have been 100 SORT error messages printed while printing table L1.

<u>Order of Input Card Images for a Non Subarea Run</u>

```
$PEAK CAPACITY RESTRAINT

*WGT

*TURN

*TRN                       (only if 16 turn penalty/prohibits)

*ALL or *LINKS

*SEL                       (only if selected links iteration)
      .
      .
      .

*END                       (only if selected links iteration)
```

<u>Order of Input Card Images for a Non-Subarea Run with the Equilibrium Option</u>

```
$PEAK CAPACITY RESTRAINT

*WGT

*TURN            EQLB           -   -
```

*TRN	(only if 16 turn penalty/prohibits)
*ALL or *LINKS	
*SEL	(only if selected links iteration)
.	
.	
.	
*END	(only if selected links iteration)
*EQLIB IMPD	(only if the optimization function is the sum of link impedance times the assigned volume for links with capacities)

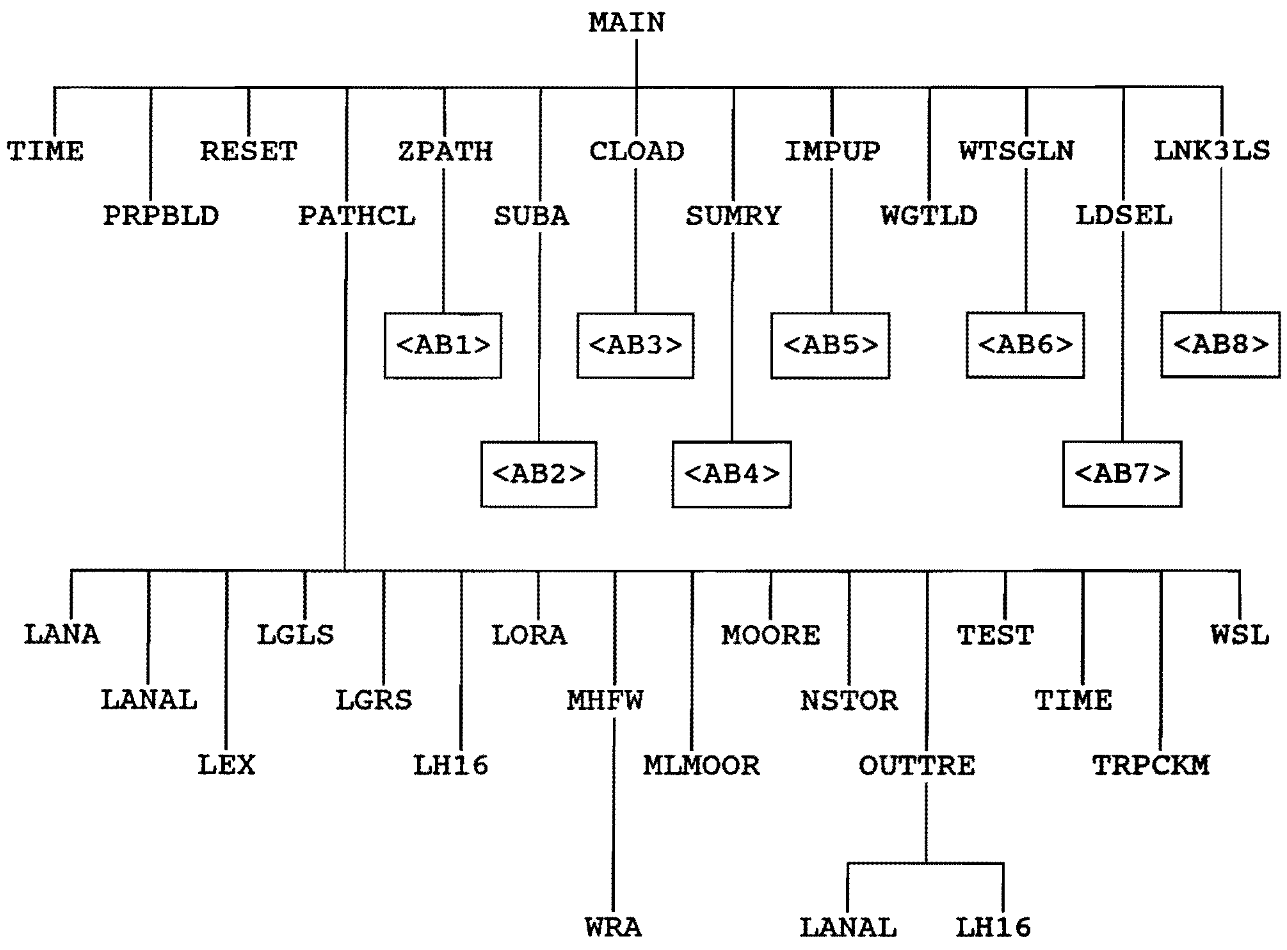
```
*FCLCRV      5
0.0     1.0
0.5     1.0
0.7     1.5
1.5     3.0
2.5     30.0
3.9     785.
*FCLCRV      6
0.0     1.0
4.0     9000.
*FCLCRV      7
0.0     1.0
1.0     1.0
2.0     20.
3.0     400.
```

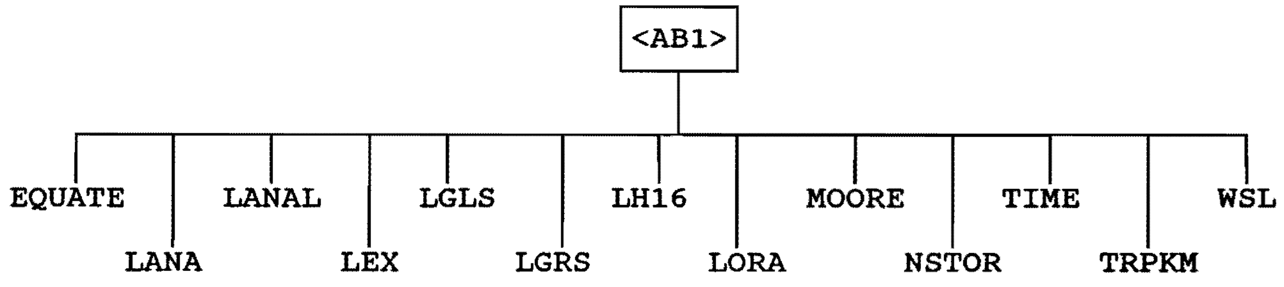
```
4.0     4000.

*END
```

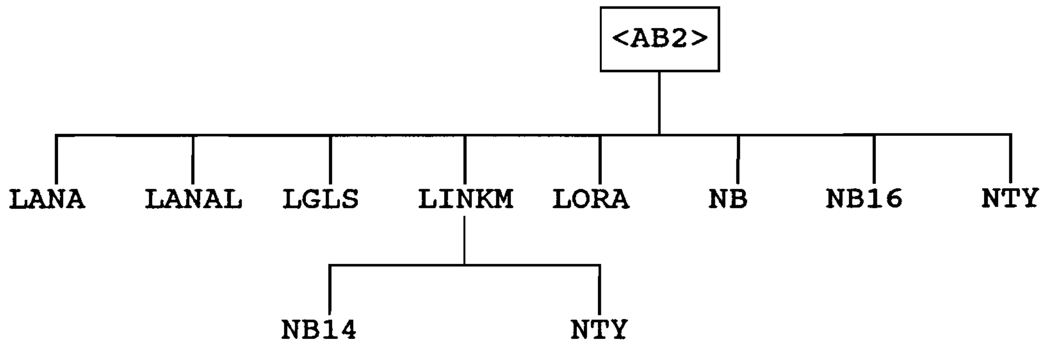
Order of Input Card Images for a Subarea Run

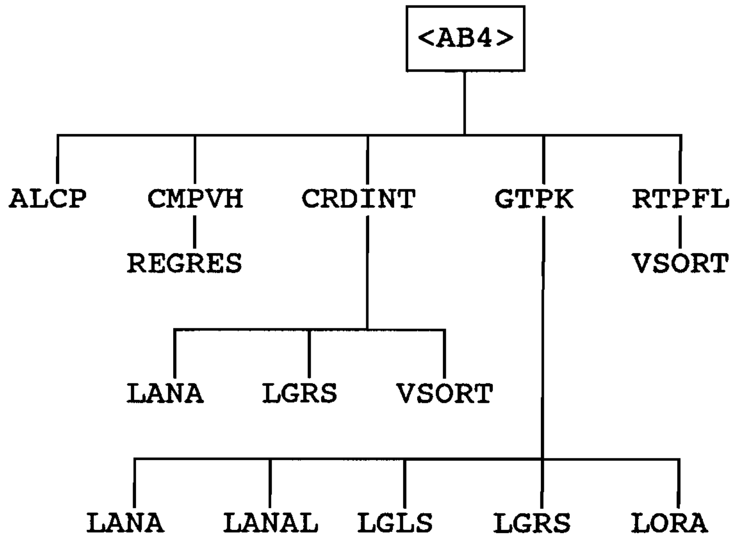
```
$SUBAREA, xx

$PEAK CAPACITY RESTRAINT

*WGT

*TURN

*TRN                          (only if 16 turn penalty/prohibits)

*ALL or *LINKS

*LASTC

XXXX EQUAL

END

LIST
    .
    .
    .

DIRECT

LINKS                         (optional)
    .
    .
    .

END

*SEL                          (only if selected links iteration)
    .
    .
    .

*END                          (only if selected links iteration)
```

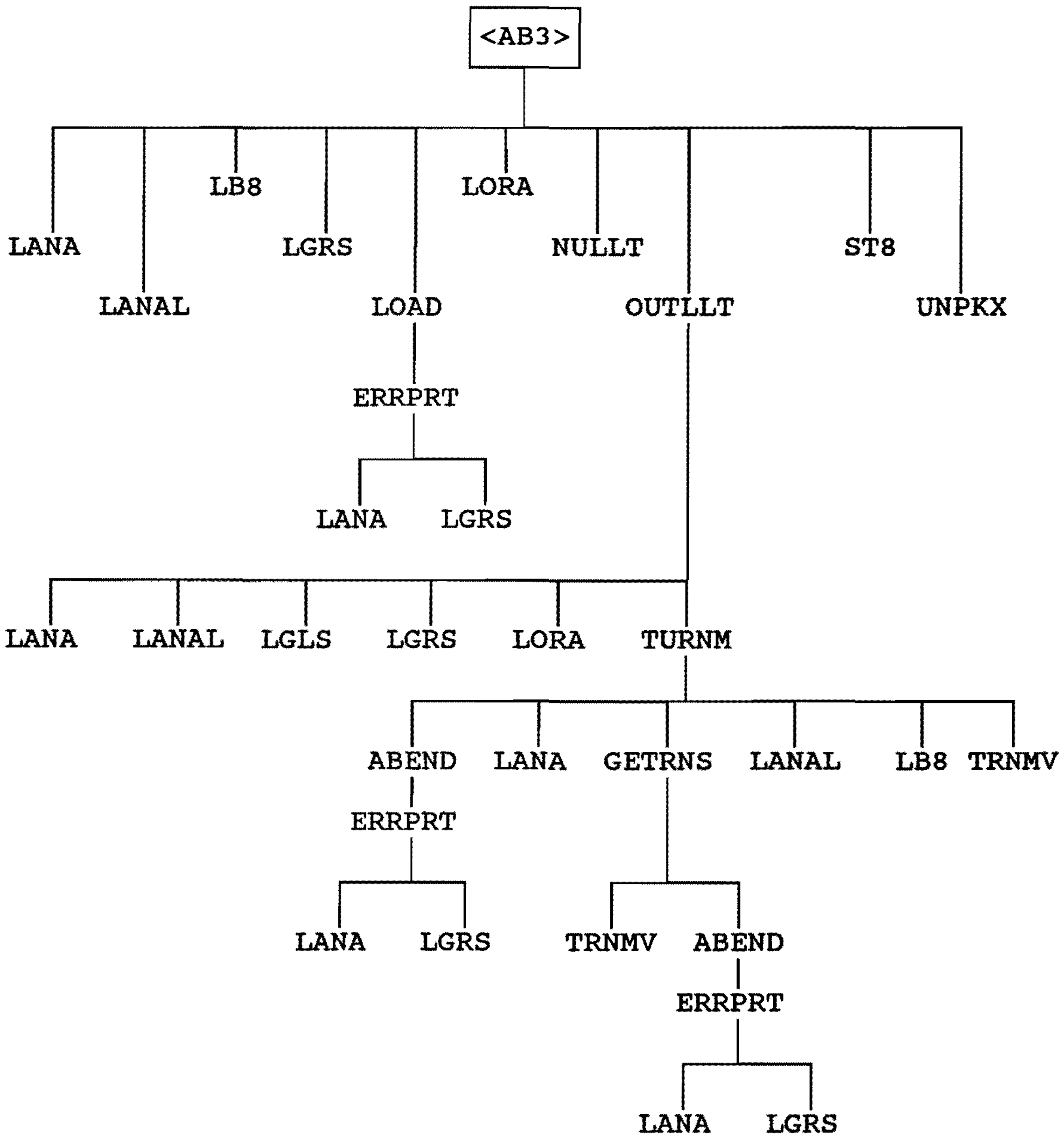
Sequence of Subroutines Called If Not Equilibrium

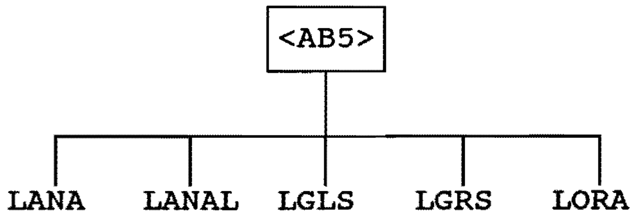


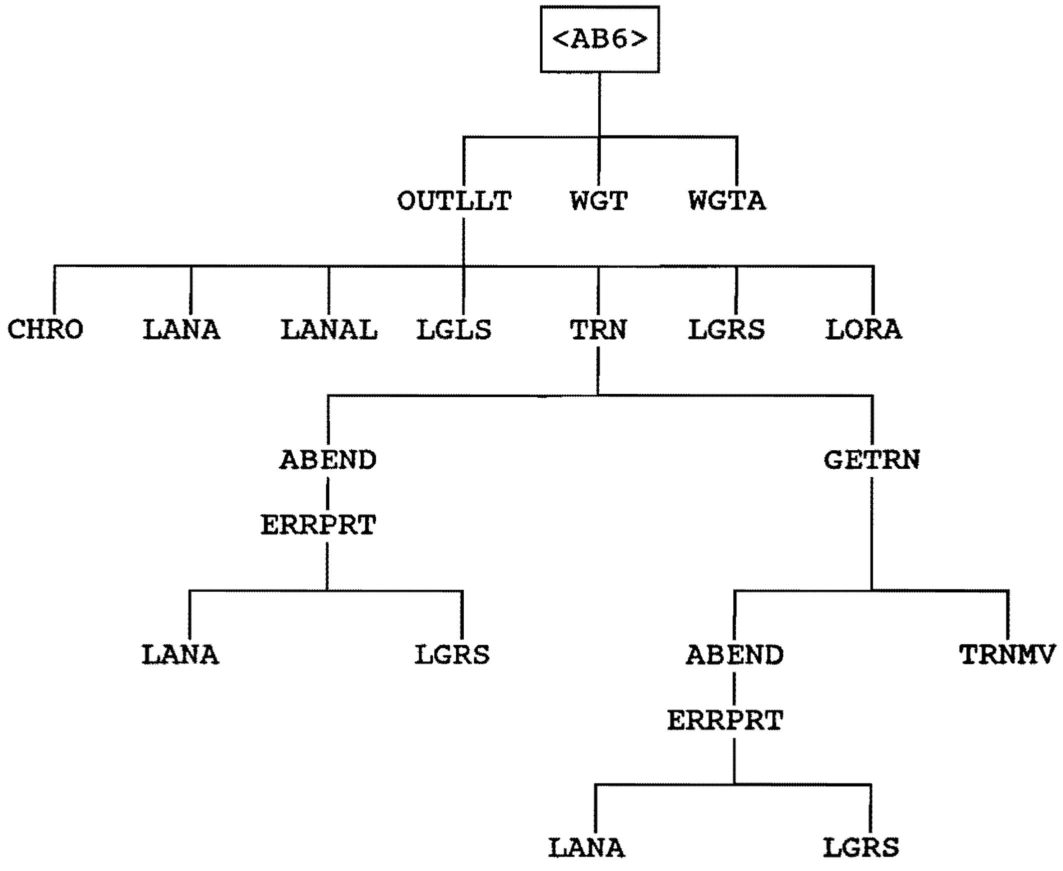


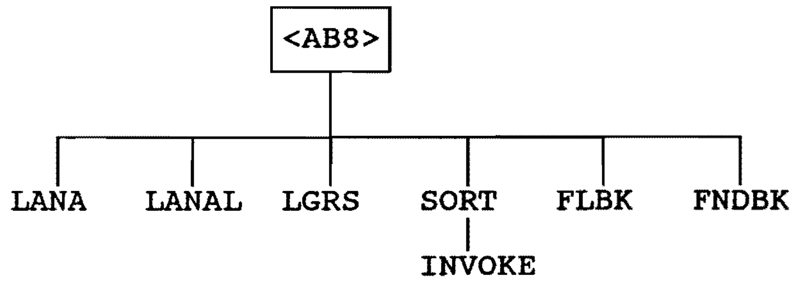

<AB2>
LANA
LANAL
LGLS
LINKM
LORA
NB
NB16
NTY
NB14
NTY

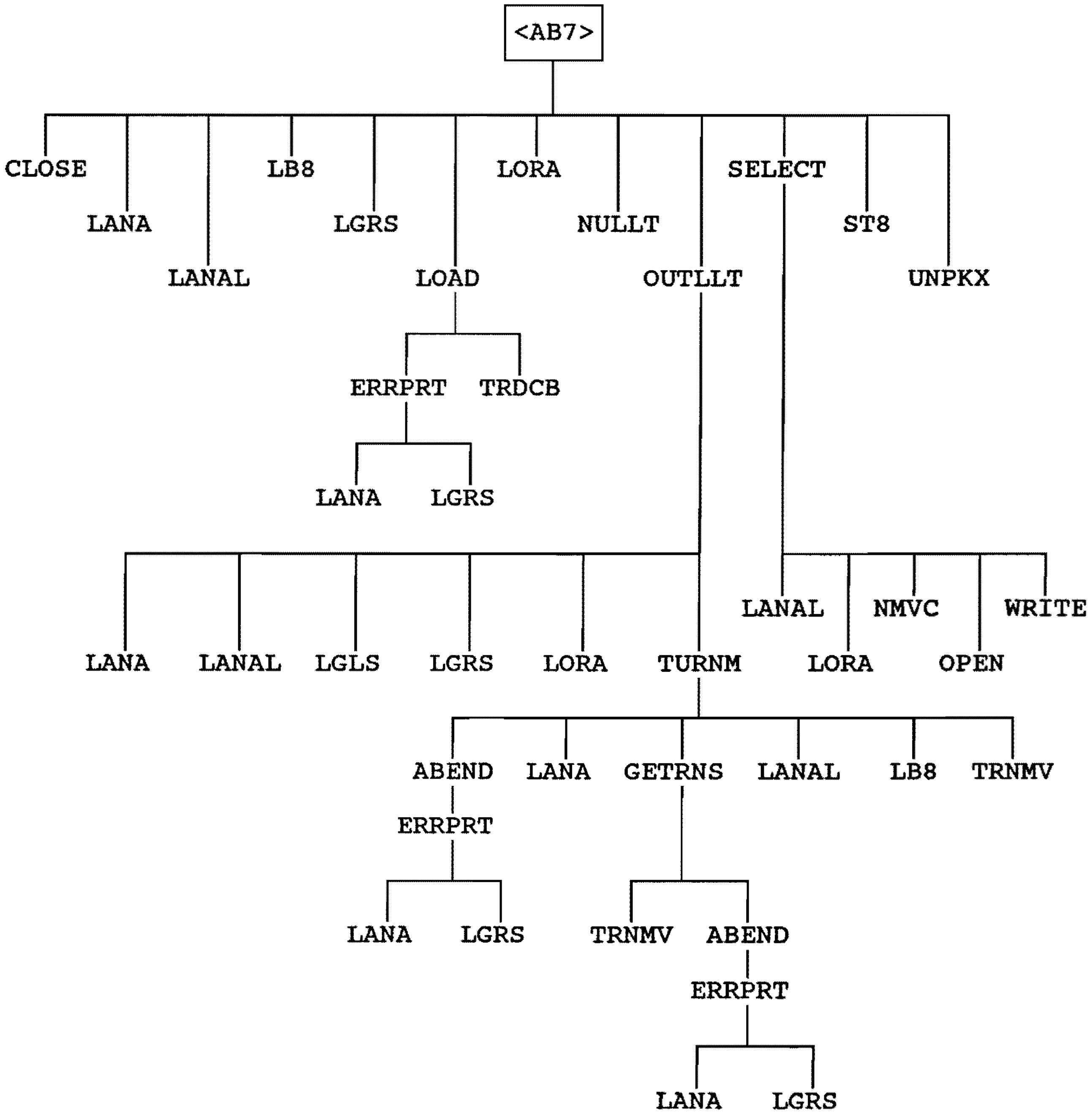

<AB4>
ALCP
CMPVH
CRDINT
GTPK
RTPFL
REGRES
VSORT
LANA
LGRS
VSORT
LANA
LANAL
LGLS
LGRS
LORA


<AB3>
LANA
LANAL
LB8
LGRS
LOAD
LORA
NULLT
OUTLLT
ST8
UNPKX
ERRPRT
LANA
LGRS
LANA
LANAL
LGLS
LGRS
LORA
TURNM
ABEND
LANA
GETRNS
LANAL
LB8
TRNMV
ERRPRT
LANA
LGRS
TRNMV
ABEND
ERRPRT
LANA
LGRS

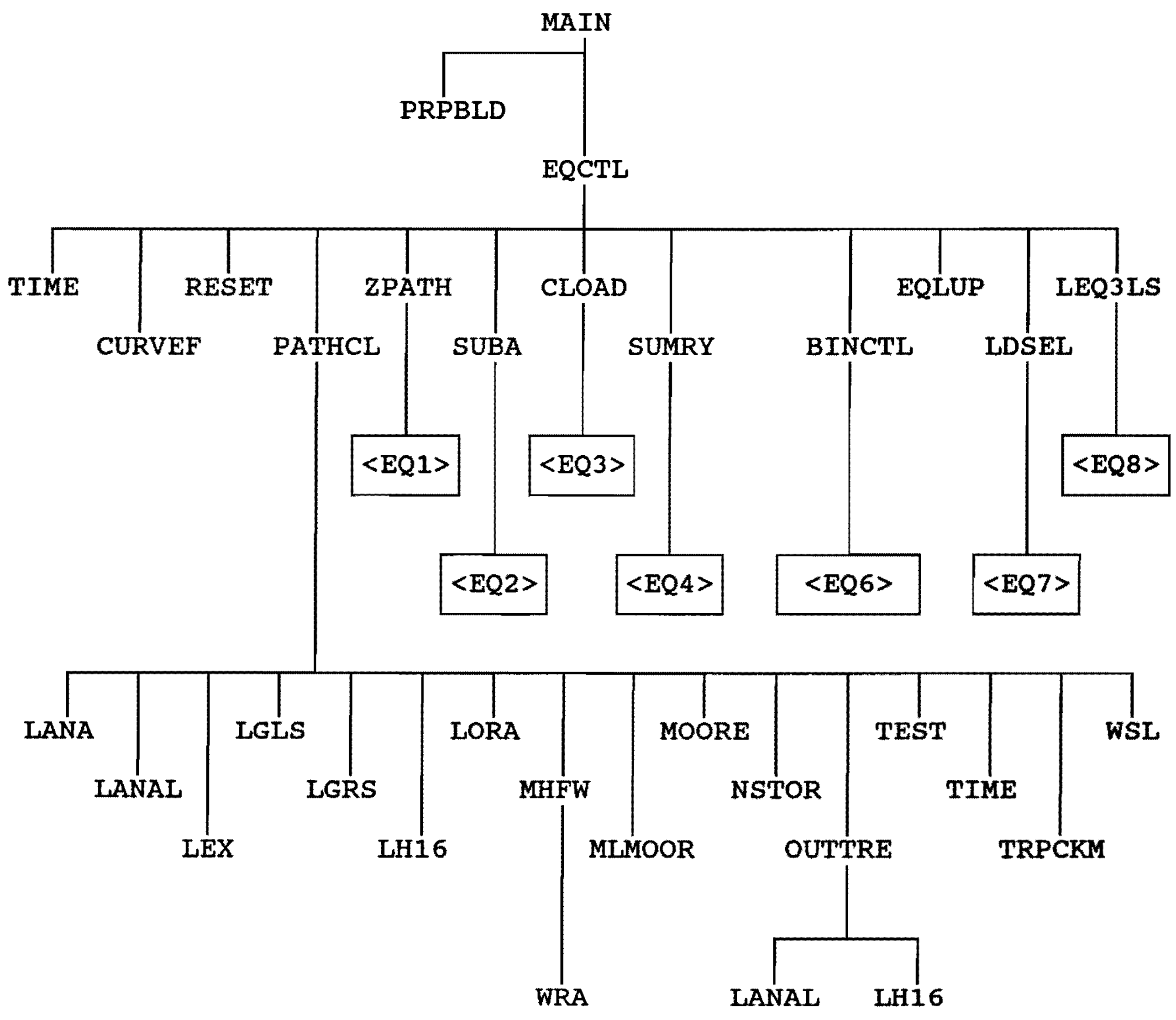

<AB5>
LANA
LANAL
LGLS
LGRS
LORA

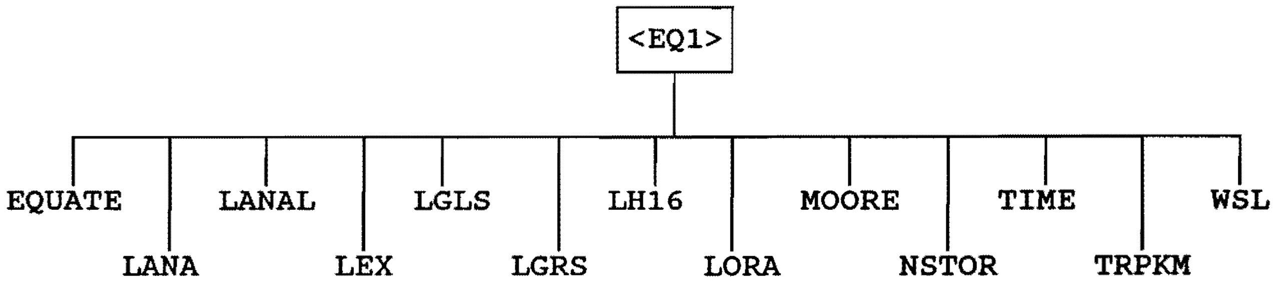

<AB6>
OUTLLT
WGT
WGTA
CHRO
LANA
LANAL
LGLS
TRN
LGRS
LORA
ABEND
GETRN
ERRPRT
LANA
LGRS
ABEND
TRNMV
ERRPRT
LANA
LGRS

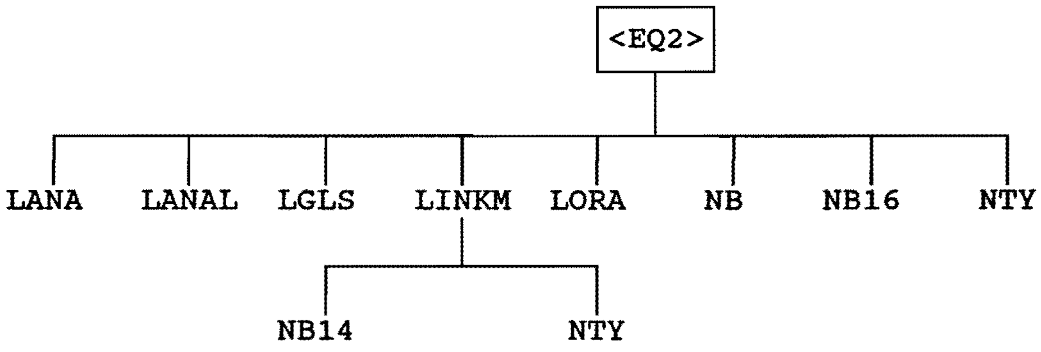

<AB8>
LANA
LANAL
LGRS
SORT
FLBK
FNDBK
INVOKE

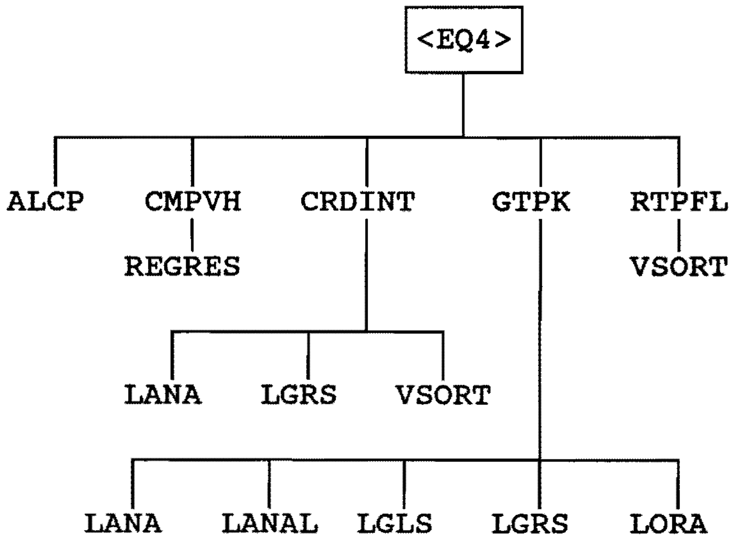

<AB7>
CLOSE
LANA
LANAL
LB8
LGRS
LOAD
LORA
NULLT
OUTLLT
SELECT
ST8
UNPKX
ERRPRT
TRDCB
LANA
LGRS
LANA
LANAL
LGLS
LGRS
LORA
TURNM
LANAL
LORA
NMVC
OPEN
WRITE
ABEND
LANA
GETRNS
LANAL
LB8
TRNMV
ERRPRT
LANA
LGRS
TRNMV
ABEND
ERRPRT
LANA
LGRS

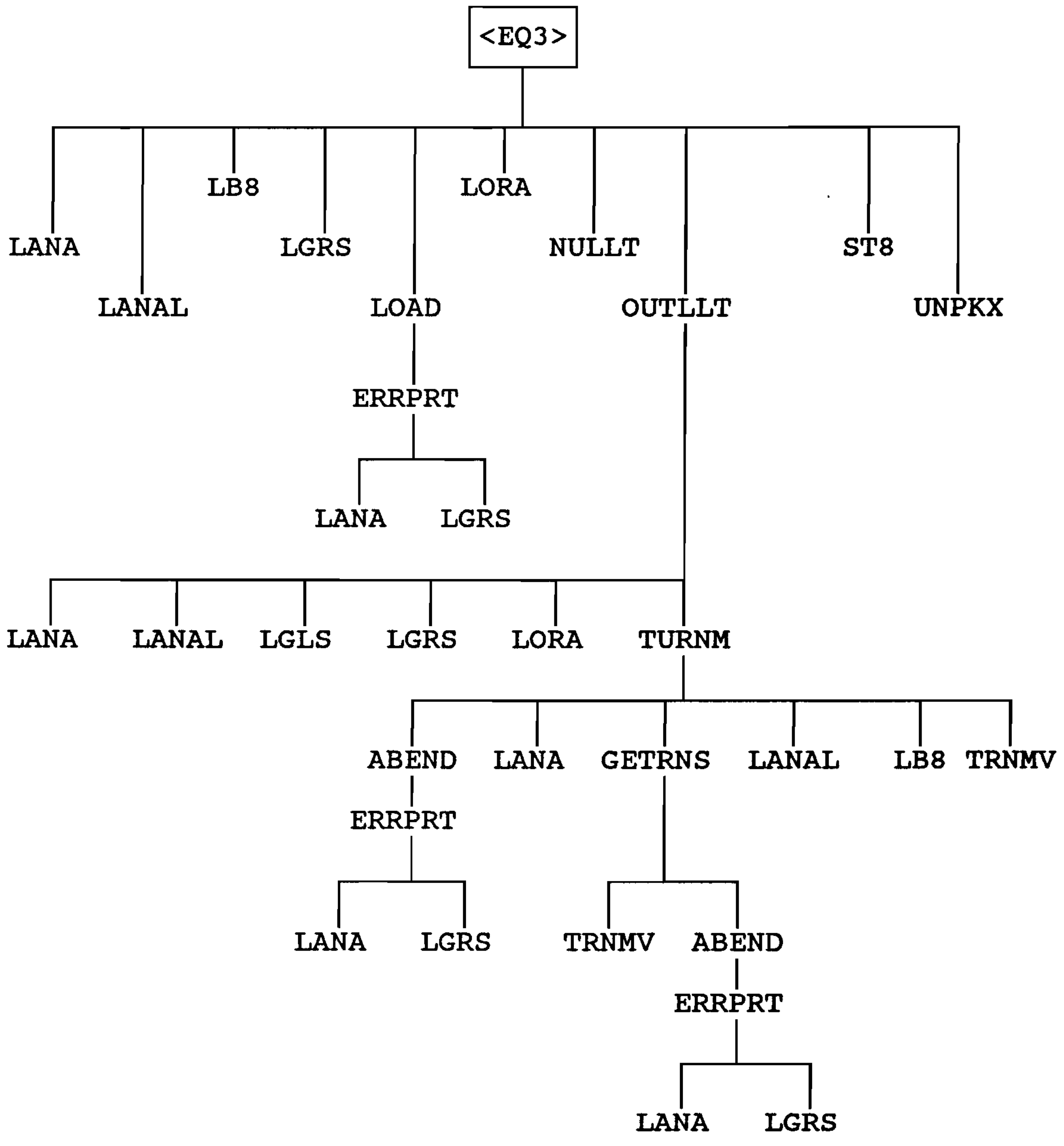
Sequence of Subroutines Called If Equilibrium

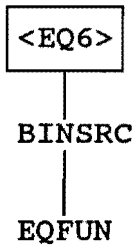



<EQ1>
EQUATE
LANA
LANAL
LEX
LGLS
LGRS
LH16
LORA
MOORE
NSTOR
TIME
TRPKM
WSL

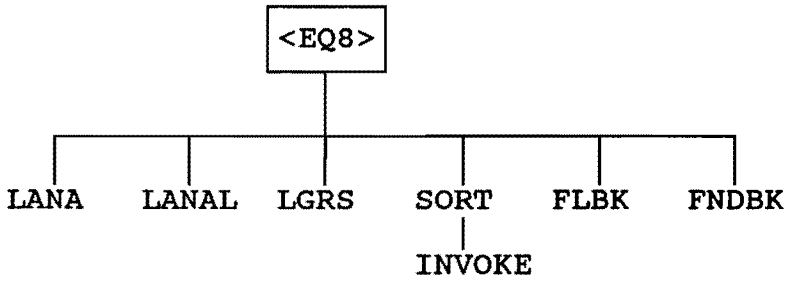

<EQ2>
LANA
LANAL
LGLS
LINKM
LORA
NB
NB16
NTY
NB14
NTY


<EQ4>
ALCP
CMPVH
CRDINT
GTPK
RTPFL
REGRES
VSORT
LANA
LGRS
VSORT
LANA
LANAL
LGLS
LGRS
LORA

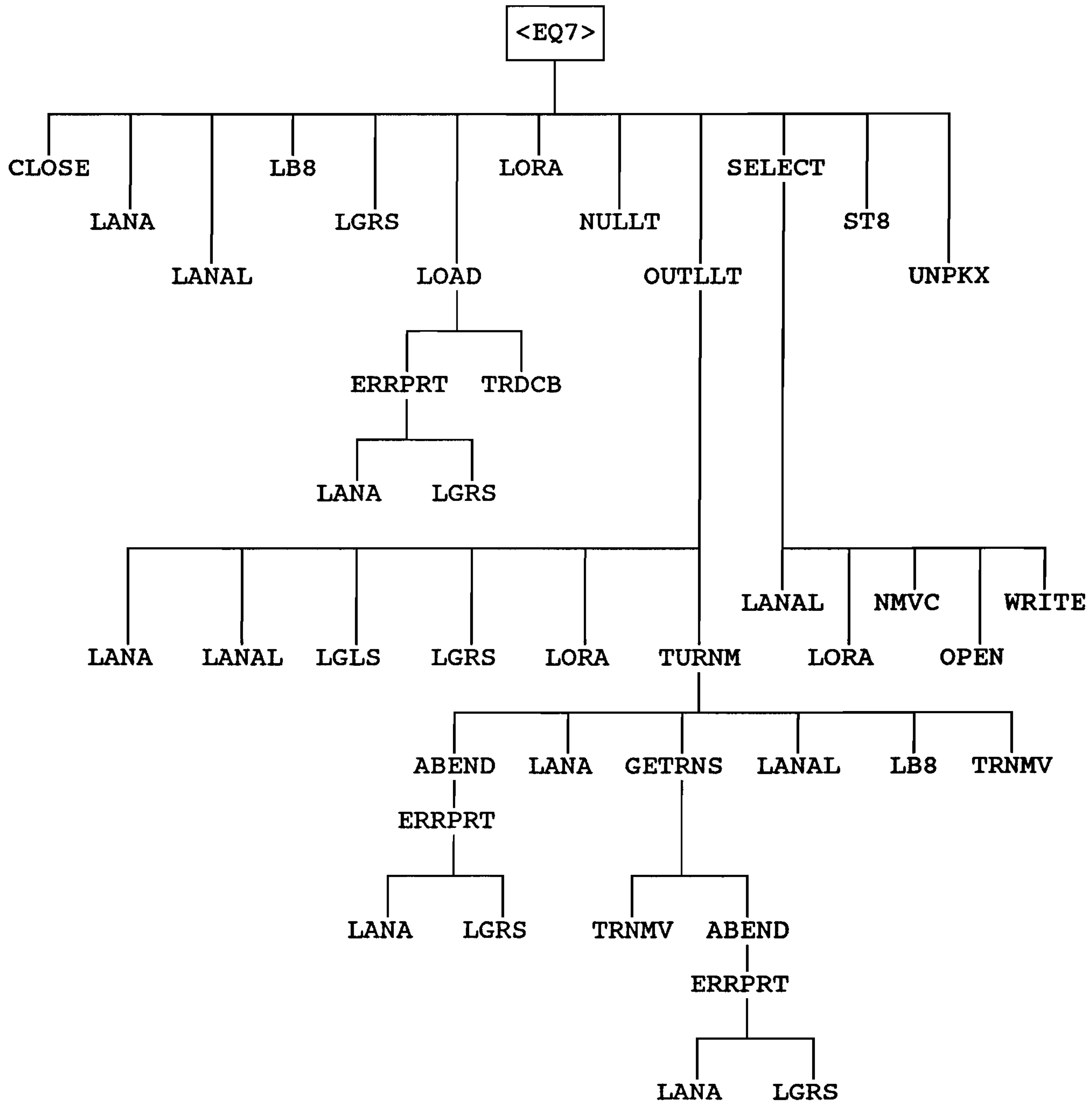

<EQ3>
LB8
LORA
LANA
LGRS
NULLT
ST8
LANAL
LOAD
OUTLLT
UNPKX
ERRPRT
LANA
LGRS
LANA
LANAL
LGLS
LGRS
LORA
TURNM
ABEND
LANA
GETRNS
LANAL
LB8
TRNMV
ERRPRT
LANA
LGRS
TRNMV
ABEND
ERRPRT
LANA
LGRS



<EQ6>
BINSRC
EQFUN



<EQ8>
LANA
LANAL
LGRS
SORT
FLBK
FNDBK
INVOKE


<EQ7>
CLOSE
LANA
LANAL
LB8
LGRS
LOAD
LORA
NULLT
OUTLLT
SELECT
ST8
UNPKX
ERRPRT
TRDCB
LANA
LGRS
LANA
LANAL
LGLS
LGRS
LORA
TURNM
LANAL
LORA
NMVC
OPEN
WRITE
ABEND
LANA
GETRNS
LANAL
LB8
TRNMV
ERRPRT
LANA
LGRS
TRNMV
ABEND
ERRPRT
LANA
LGRS

Summary of Individual Subroutines

MAIN: In the execution of PEAK CAPACITY RESTRAINT without the Equilibrium option, the MAIN program performs the following steps:

1. The logical variable PEAK in the PK labeled common is set to true to allow subroutines ALCP, CLOAD and IMPUP to work using a directional capacity restraint.

2. The subroutine PRPBLD is called to read the *WGT, *TURN, *TREE and loaded network output specification card images.

3. If this is a restart of PEAK CAPACITY RESTRAINT then subroutine RESET is called to position FORTRAN Unit 3 correctly.

4. If the SUBAREA data set is zero then the subroutine PATHCL is called to build minimum path trees for this iteration; otherwise the subroutine ZPATH is called to build trees if the subarea option is used with PEAK CAPACITY RESTRAINT.

5. If this is not the iteration for selected links, then the subroutine CLOAD is called to load the trees built for this iteration. If this is the selected link iteration, the subroutine LDSEL is called instead to load the trees.

6. If the SUBAREA data set is not zero, then the subroutine SUBA is called to prepare the network data set for summaries.

7. The subroutine SUMRY is called to produce summary tables for this iteration.

8. If no weights were input on the *WGT card image and this is the new impedance update, then subroutine IMPUP is called to update the link impedances.

9. Steps 3 through 7 are repeated for the number of iterations.

10. Subroutine WGTLD is called.

11. Subroutine WTSGLN is called.

12. If an extra weighted iteration is specified, then subroutines PATHCL and then CLOAD are called.

13. If the SUBAREA data set is not zero, then the subroutine SUBA is called to prepare the network data set for subroutine LNKLST.

14. Subroutine SUMRY is called to produce summaries.

15. Subroutine LNK3LS is called to produce a directional summary of the links with updated link impedances. The summary lists both directions of each link together but does not sum to the directional links. As both directions are listed the first directional link listed has its Anode less than the Bnode. The second directional link is labeled (B-A). This subroutine writes all directional links to Unit 83. It then calls subroutine SORT to sort these links into Bnode-Anode order. The sorted links are then read back in from Unit 84. This requires a special set of JCL which is also used by the ASSEMBLE PEAK NETWORK program. As the sorted links are read back in a nondirectional network data set is written to Unit 85. It is anticipated that this nondirectional network data set would be used to plot nondirectional volumes.

MAIN: In the execution of PEAK CAPACITY RESTRAINT with the Equilibrium option, the MAIN program performs the following steps:

1. The logical variable PEAK in the PK labeled common is set to true to allow subroutines ALCP, CLOAD and IMPUP to work using a directional capacity restraint.

2. The subroutine PRPBLD is called to read the *WGT, *TURN, *TREE and loaded network output specification card images.

3. The subroutine EQCTL is called.

ABEND: This subroutine builds a message which contains the FORTRAN internal statement number of the line that called it. Next the subroutine calls subroutine ERRPRT. Then the subroutine abends with a user code of 100. This subroutine is written in Assembly language and is a CSECT named ABEND which is contained in the deck with subroutine GETVOL.

ALCP: This subroutine calculates a multiple regression analysis to determine the iteration weighting for a PEAK CAPACITY RESTRAINT process. It then prints the results of this analysis. The summations for the regressions are done in subroutine GTPK. Only the links with a non-zero count (or capacity depending on which is specified) are considered, and centroid connectors are ignored. The count (or capacity) is the dependent variable, and the assigned directional volumes from each of the iterations are the independent variables in the analysis. This subroutine is called by subroutine SUMRY. This subroutine contains the labelled common HEADER, CAPRES and TBL. This subroutine is written in FORTRAN IV.

BINCTL: This subroutine reads the network data set and obtains the previous weighted assignment directional link volumes, the last assignment directional link volumes, the first link impedances, and the link capacity. The subroutine then calls BINSRC to do a binary search. The subroutine then calculates and saves the percents to produce the weighted link volumes for this iteration.

BINSRC: This subroutine performs a binary search to find the optimum combination of the last iteration to the previous weighted iteration volumes. The minimization function is either the sum of the new link impedances times the assigned volume or the sum of the integral of the link impedance update function over the assigned volume. The impedance used in this minimization function is determined from the new trial weighting being used in BINSRC.

The subroutine initially calculates the optimization function for the end points of 0.0 and 1.0 (which are 0 percent of the previous weighted and 100 percent, respectively). It splits this range and calculates the optimization function for a lambda of 0.5 (50 percent). It then throws away the end point with the highest optimization value, and the point of 0.5 replaces that end point. The process is repeated 17 times to develop the estimated lambda value.

CHRO: This subroutine moves character data from non-character variables to character variables. The first argument is the destination of the move. The second argument is the origin of the move. The third argument is the length of the move. This function is an entry point in the Assembly language CSECT NMVC.

CLOAD: It reads the network from Unit NETWORK and modifies this to the format needed for the LOAD subroutine. It initializes the directional link volume array and the turn volume array to zero. It calls subroutine SELECT if it is a LOAD SELECTED LINKS run. It reads the trip matrix and the paths data sets. These are assumed to be in sort on the origin zones. It calls subroutine LOAD to load trips in the network if there is both a tree record and one or more trip records for an origin zone. After the network is loaded, CLOAD calls subroutine SVLOAD to save the loaded network on Unit 3 if a PEAK CAPACITY RESTRAINT run is in iterations 1 through 5. CLOAD then calls subroutine OUTLLT to print the loaded network.

CLOSE: This subroutine closes data set SELTRP.

CMPVH: This subroutine prints the Jurisdiction Summary or the Jurisdictional/FUNCTIONAL Cross-classification Tables and the three

Comparison of Assigned Volumes with link volumes, Counted volumes, and Capacities.

CRDINT: This subroutine calls VSORT (which sorts the corridor intercept records) and prints the corridor intercept tables.

EQCTL: This subroutine controls the equilibrium option of PEAK CAPACITY RESTRAINT and ASSIGN SELF-BALANCING.

1. The subroutine CURVEF is called to read impedance function update parameters or user impedance function curves by functional class. The user also can input an option to use an optimization function of the sum of the vehicle hours of travel for links with capacities.

2. If this is a restart of PEAK CAPACITY RESTRAINT, then subroutine RESET is called to position FORTRAN Unit 3 correctly.

3. If the SUBAREA data set is zero then the subroutine PATHCL is called to build minimum path trees for this iteration; otherwise the subroutine ZPATH is called to build trees if the subarea option is used with PEAK CAPACITY RESTRAINT.

4. If this is not the iteration for selected links, then the subroutine CLOAD is called to load the trees built for this iteration. If this is the selected link iteration, the subroutine LDSEL is called, instead, to load the trees.

5. If the SUBAREA data set is not zero, then the subroutine SUBA is called to prepare the network data set for summaries.

6. The subroutine SUMRY is called to produce summary tables for this iteration.

7. If this is iteration 2 or later, then the BINCTL subroutine is called to find a new optimum lambda for combining the present iteration with the previous weighted iteration.

8. The subroutine EQLUP is called to put the new impedances in the network data set which will be used for the next iteration assignment.

9. Steps 3 through 8 are repeated for the number of iterations.

10. Subroutine SUMRY is called to produce summaries.

11. Subroutine LEQ3LS is called to produce a directional summary of the links with updated link impedances. The summary lists both directions of

each link together but does not sum them to directional links. As both directions are listed, the first directional link listed has its Anode less than the Bnode. The second directional link is labeled (B-A). This subroutine writes all directional links to Unit 83. It then calls subroutine SORT to sort these links into Bnode-Anode order. The sorted links are then read back in from Unit 84. This requires a special set of JCL which is also used by the ASSEMBLE PEAK NETWORK program. As the sorted links are read back in, a nondirectional network data set is written to Unit 85. It is anticipated that this nondirectional network data set would be used to plot nondirectional volumes.

EQFUN: This subroutine calculates the equilibrium minimization function.

EQLUP: This subroutine updates the link impedances using the new weighted assignment volumes.

EQUATE: The primary function of this subroutine is to establish zone to sector zone equivalences. EQUAL card images containing the sector zone numbers and the zone numbers which are in those sectors are read, and an array is established containing the zone to sector zone equivalences. If any zone in the network has not been equivalenced to a sector, that zone is equivalenced to itself which then indicates it is in the subarea window or the boundary surrounding it. If no EQUALS cards are encountered, all zones are equivalenced to sector one. This subroutine is written in FORTRAN IV VS 77.

ERRPRT: This subroutine prints error messages from assembly language subroutines. The first argument is an array of length 17. The first index of the first argument is the address where ERRPRT was called from. The next 16 elements of this array are the 16 general purpose registers at the time ERRPRT was called. The second argument is the home zone if it applies, otherwise it is zero. The third argument is a character variable of length 50. This subroutine is written in FORTRAN IV VS.

FLBK: This subroutine reads link data from SORTOUT (Unit 84) into a buffer for one Anode. One record from the next Anode is read and also saved.

FNDBK: This subroutine finds a specific reversed link from the data read in by FLBK. If the Anode is past the last complete Anode data read by FLBK then FLBK is called to read the next Anode link data.

GETRN: This subroutine gets the weighted turn volumes which were saved and places them in the turn volume matrix.

GETRNS:	This subroutine gets the turn volumes which were saved and places in the turn volume matrix.
GTPK:	This subroutine prints the V/C cross-classification table if there are two or more assignments on Unit NETWORK. It computes the summations necessary for the tables printed by subroutine CMPVH and for the curve fit printed by subroutine ALCP. It saves corridor intercept information in core in labeled common CD. It writes route profile records on Unit ROUTE. If logical variable SUM is true, GTPK calculates weighted directional volumes and updates the network data set writing it on Unit NETWORK. All comparisons and tables are made from the weighted directional volumes if SUM is true.
IMPUP:	This subroutine uses directional volumes when run with PEAK CAPACITY RESTRAINT and nondirectional volumes when run with ASSIGN SELF-BALANCING. This subroutine implements the new capacity restraint function when no weights are input. When no weights are input for PEAK CAPACITY RESTRAINT with the new impedance update function, then the percents to load are calculated by regression by subroutines GTPK and ALCP at the end of each iteration. This subroutine then reads Unit NET, which is the first argument of IMPUP, and updates the link impedances using the percents calculated from the regression for calculating the weighted link volumes. The updated network is written to Unit NNET, which is the second argument of IMPUP.
LANA:	This integer function does a logical 'and' of the first argument and the second argument. This function is an entry point in the Assembly language CSECT LOPS.
LANAL:	This integer function does a logical 'and' of the first argument and a fixed mask of X'00003FFF'. This extracts 14-bit integers in the range of 0 to 16,383 from other data. This function is an entry point in the Assembly language CSECT LOPS.
LB8:	This integer function indexes into an array of 8-bit integers and returns the value. The first argument is the array and the second argument is the index. This function is an entry point in the Assembly language CSECT NMVC.
LDSEL:	This is an entry point in subroutine CLOAD. It sets the Selected Links option to true, then loads a network with selected links output to data set SELTRP. This subroutine is written in FORTRAN I VS 77.
LEX:	This integer function does a logical exclusive 'or' of the first argument and the second argument. This function is an entry point in the Assembly language CSECT LOPS.

LGLS:	This integer function does a logical left shift. The first argument is the 32-bit data to be shifted and the second argument is the number of bits to shift left. This function is an entry point in the Assembly language CSECT LOPS.
LGRS:	This integer function does a logical right shift. The first argument is the 32-bit data to be shifted, and the second argument is the number of bits to shift right. This function is an entry point in the Assembly language CSECT LOPS.
LH16:	This integer function does a shift left logical 16 followed by a shift right logical of 16 on the half word argument. This extracts a 16-bit integer in the range of 0 to 65,535 from the half word. This function is an entry point in the Assembly language CSECT NMVC.
LINKM:	This subroutine reads the card images which describe a subarea for focusing. It checks that the links on these card images are actually in the network. It then flags the links as BOUNDARY, AREA, or ADDED which are in these card images. This subroutine is written in FORTRAN IV VS 77.
LNK3LS:	This subroutine lists the data by directional link for the 1 to 8 iterations of PEAK CAPACITY RESTRAINT and outputs Table L1. This subroutine calls subroutine SORT to sort the links into Bnode-Anode order so that it can print Anode-Bnode directional volumes and also Bnode-Anode directional volumes on the next line. This subroutine prints A-node, B-node, Distance, Functional Class by link. Also by iteration the directional Link Volume, the Volume to capacity ratio code (a single character), the link impedance and the link speed. This subroutine also writes a nondirectional network data set on Unit 85. This subroutine is written in FORTRAN IV VS 77.
SORT:	This subroutine calls subroutine INVOKE to sort the records written to Unit 83.
INVOKE:	This subroutine links dynamically to the system SORT package. It also places the number of sort input records on the sort statement which it passes to the sort package.
LOAD:	This subroutine loads a trip record by adding each trip interchange volume to all of the directional link volumes in the path connected between the origin and destination zones of the trip interchange. Some turn volumes are also summed in this process. This subroutine also writes a record on Unit SELTRP for each selected link crossed in loading each trip interchange volume.

LORA:	This integer function does a logical 'or' of the first argument and the second argument. This function is an entry point in the Assembly language CSECT LOPS.
MHFW:	This subroutine moves an array of half words to an array of full words. The subroutine changes -1 inputs to an output value of X'OOFFFFFF'. Other values which were not -1 are divided by 2. The subroutine then calls subroutine WRA to write the resulting full word array. This subroutine is written in Assembly language and is the only subroutine in this deck.
MLMOOR:	This subroutine builds one minimum time path tree each time it is called. It also sums the distance along this minimum time path. This subroutine is written in Assembly language and is the only control section in this deck.
MOOR:	This control section builds one minimum path tree each time it is called. Its entry point is MOORE.
NB:	This subroutine extracts the node number and flags from a link. The first argument is the link contained in a full word. The second argument returned by NB is the node number of the link. The third argument is also returned by NB and is one or more flags indicating if the link is in the area. This subroutine is written in Assembly language and is the CSECT name.
NB14:	This integer function does a logical 'and' of a half word argument and a fixed mask of X'00003FFF'. This extracts a 14-bit integer in the range of 0 to 16,383 from the half word. This function is an entry point in the Assembly language CSECT NB.
NB16:	This integer function does a logical 'and' of a half word argument and a fixed mask of X'0000FFFF'. This extracts a 16-bit integer in the range of 0 to 65,535 from the half word. This function is an entry point in the Assembly language CSECT NB.
NMVC:	This subroutine is a character move subroutine. The first argument is the destination character variable. The second argument is the destination offset in bytes. The third argument is the origin offset in bytes. The fourth argument is the length of the move in bytes. The fifth argument is the origin character variable. Arguments 2, 3, and 4 are full word arguments while the first and fifth arguments can have any alignment. This subroutine is written in Assembly language and is a CSECT.
NSTOR:	This subroutine stores a value into a half word array. This first argument is the half word array. The second argument is an index into the array. Bits 6 through 31 of the third argument are stored into the indexed half word array.

This subroutine is written in Assembly language and is an entry point in CSECT NMVC.

NTY:	This integer function obtains a half word argument which it then shifts right by 14 bits. Then the shifted argument is logical anded[8] with the value X'00000003'. This extracts a 2 bit integer in the range of 0 to 3 from the half word. This function is an entry point in the Assembly language CSECT NB.
NULLT:	This subroutine changes the turn codes in the first argument for the number of zones set in the second argument. The turn code is saved in bits 3 through 7 of the high order byte of each full word. The turn code is set to the value of 28 which indicates that no turns should be saved for this node zone. The name of this subroutine is derived from null turns. This function is an entry point in the Assembly language CSECT LOPS.
OPEN:	See ASORT.
OUTLLT:	This subroutine controls the printing of the loaded network. It prints page headings, calls subroutine TURNM to get the link volumes and turn volumes for a node and formats the directional link volume, nondirectional link volumes, and turn volumes.
OUTTRE:	This subroutine prints one tree each time it is called.
OUTWLT:	This subroutine prints the loaded network for one segment of the loaded network. It calls subroutine TRN to calculate the turn volumes for one node. It reads the node numbers and the node names from Unit NETWORK, and it writes the updated Network data set with the weighted assignment volumes added on Unit NEWNET.
WGT:	This subroutine multiplies a group of volumes by an integer percent and places the results in another array.
PATHCL:	This is the control subroutine for building trees. It defines arrays used by subroutines called from this division. It reads the network into core from Unit NETWORK and changes it to the form used by the tree builder subroutine. It controls the building of trees, the printing of trees, the packing of the paths, and writes the paths data set and the separation matrix data sets and the MILESEP separation matrix.

[8] Boolean algebra operation.

PRPBLD: This subroutine reads the *TURN card and the *TREE cards which specify the turn penalty and the trees to be built and printed. The COPY parameter is also specified on the *TURN card if it is used.

REGRES: This subroutine performs a linear regression analysis and prints the results of this analysis.

RESET: This subroutine reposition FORTRAN Unit 3 to the correct place to do a restart of PEAK CAPACITY RESTRAINT. This subroutine is written in FORTRAN IV VS 77.

RTPFL: This subroutine reads the route profiles from Unit ROUTE and prints the route profile tables.

SELECT: This subroutine reads the parameter cards of LOAD SELECTED LINKS. For each *SEL card it writes one record on Unit SELTRP, and it marks both of the one-way directional links as selected. This subroutine also reads one of the following parameter cards: *ALL, *LINKS, or *NONE. If the *LINKS card is read, this subroutine sets all turn codes in memory to 28. If the *NONE card is read, a logical variable is set to specify that the loaded network will not be printed.

ST8: This subroutine saves the last byte of a full word in the address accessed by the first argument. The location where the byte is saved is indexed by the second argument. The data saved is the fourth byte of the third argument. This function is an entry point in the Assembly language CSECT NMVC.

SUBA: This subroutine reads the card images that describe a subarea of a network. Then this subroutine extracts the subarea part of the network to the Unit SUBAREA data set. This subroutine is written in FORTRAN IV VS 77.

SUMRY: This is the control program for the summaries produced after an assignment. The subroutines called by SUMRY are determined by 3 logical variables. One of the logical variables, SUM, if true causes GTPK to produce a weighted assignment on Unit NETWORK and produce all tables and comparisons from this weighted assignment. Subroutine ALCP is called only if logical variable RES is true. If logical variable RTP is false, then the corridor intercept and route profile tables are skipped.

TEST: This subroutine tests whether the data contained in a path record contains any values of 6. A path record contains relative link indices of the path back to the origin. There are only six ways out of each node maximum. The relative index starts at 0 for the first link. Only values between 0 and 5 will be used for link indices. A value of 7 is used to indicate the origin zone. The third argument is set to a count of the number of 6 path indices found; a count of

	zero indicates no error. This subroutine is written in Assembly language and is an entry point in the TRPCKM CSECT.
TIME:	This subroutine returns the time of day in units of 0.01 of a second.
TRDCB:	This is the DCB (data control block) for the SELTRP data set. It is defined as an entry point in CSECT CLOSE which is part of the LOAD deck. It is written in Assembly language.
TRN:	This subroutine gets the weighted directional, volumes the weighted nondirectional volumes, and the weighted turn volumes saved. It also calculates the other weighted turn volumes and marks which turn volumes should not be printed because of one-way links.
TRNMV:	This subroutine adds 2 indices together and gets the assigned volumes indexed by the sum from a full word array.
TRPCKM:	This subroutine packs an array of path indices from 16-bit integers to ten 3-bit integers per word. The control section also contains the entry point TEST which checks to see that an array of packed path contains no indices of 6.
TURNM:	This subroutine gets the directional volumes, nondirectional volumes, and turn volumes saved. It also calculates the other turn volumes and marks which turn volumes should not be printed because of one-way links.
UNPKX:	This subroutine unpacks the path indices and places them in full words.
VSORT:	This subroutine sorts records in core. The first argument in the calling sequence is the address of the array of records to be sorted. The second argument is the number of records. The third argument is the length of each record in bytes (must be between 1 and 256 bytes). The fourth argument is the length of the sort key in bytes (must be between 1 and 256 bytes) which cannot be longer than the record length. The sort key starts at the first byte of the record. The sort key is treated as an unsigned binary number and the records are sorted into ascending order on the sort keys.
WGTA:	This subroutine multiplies a group of volumes by an integer percent and adds the results into another array.
WGTLD:	This subroutine converts the positive weights output from a regression of the iteration volumes from PEAK CAPACITY RESTRAINT into a set of percents which sum to 100. This subroutine is written in FORTRAN IV VS 77.

WRA:	This subroutine writes a record of separations for one tree for the centroids only. This subroutine is called by subroutine MHFW.
WRITE:	This subroutine writes one record to the data set whose DDname is SELTRP. This subroutine has one argument which is the address of a location containing 18 bytes of data to be written as the next record. This subroutine is written in Assembly language, and it is an entry point in the CLOSE CSECT which is part of the LOAD deck.
WSL:	This subroutine writes a record of separations for one tree for the centroids only.
WTSGLN:	This subroutine reads Unit 3 and, using the weights for each iteration, sums up the weighted directional link volume, reverses directional link volumes, and turn volumes for one segment in memory. It then rewinds Unit 3 and calls subroutine OUTWLT to print this segment of the loaded network. It repeats the above steps for other segments. The line counter used by subroutine OUTWLT to print page headings is only initialized for the first call to OUTWLT.
ZPATH:	This subroutine controls building trees for BUILD AREA TREES. This subroutine calls subroutine EQUATE, and then it builds trees for sector centroids not equated to sector centroids. This subroutine is written in FORTRAN IV VS 77.

ADDITIONAL JCL FOR PEAK CAPACITY RESTRAINT

```
//FT83F001   DD  UNIT=SYSDA,DSN=&INSORT,SPACE=(6320,(1600,400)),
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=6320)
//SORTIN     DD  DSN=&INSORT,VOL=REF=*.FT83F001,DISP=(OLD,PASS)
//SORTMSG    DD  SYSOUT=A
//SORTWK01   DD  UNIT=SYSDA,SPACE=(6320,(1200),,CONTIG),SEP=SORTIN
//SORTWK02   DD  UNIT=SYSDA,SPACE=(6320,(1200),,CONTIG),SEP=SORTWK01
//SORTWK03   DD  UNIT=SYSDA,SPACE=(6320,(1200),,CONTIG),SEP=SORTWK02
//SORTOUT    DD  DSN=&OUTSORT,UNIT=SYSDA,SEP=SORTWK03,
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=6320),SPACE=(6320,(1600,400))
//FT84F001   DD  DSN=&OUTSORT,VOL=REF=*.SORTOUT,DISP=(OLD,PASS)
```

Note that the placement of the FT83F001 and SORTIN statements must be in the listed order. Also the SORTOUT and FT84F001 statements must also be in the order listed. The reason that these statements must be in the order listed is that the first statement allocates the data set for the SORT package, and the second statement allows it to be read by the

FORTRAN program. The space allocations above are enough for 41,000 one way links with 19 assignments.

APPENDIX B: NEW OPERATING INSTRUCTIONS FOR THE PEAK CAPACITY RESTRAINT ROUTINE

ASSIGN SELF-BALANCING: Program Function

The function of this program is to produce a multi-path assignment in which the assigned volumes are in relative balance with traffic counts or else conform to link capacities. This is an iterative technique which adjusts link impedances to obtain the desired balance. There are two methods of obtaining the iteration weights. The methods are user-input weights and the equilibrium assignment method of weighting iterations. The program produces several cross-classification tables and comparison tables to indicate how well the above function is being achieved.

Data Set References

Input:	$NETWORK	(1)	Network (contains network, lanes, previous assignments and iteration assignments) [9]
	$CTVOUT	(8)	Trip Matrix [1]
		(49)	Trees for COPY option
		(50)	Trees for SKIP option
		(3)	Nondirectional assigned volumes and turn volumes from previous iterations [10]
Scratch:		(48)	Table look-up for impedance update functions and integrals by functional class.
Output:	$NEWNET	(9)	Network (updated for next iter.) [1]
	$SEPARAT	(20)	Separation Matrix [1]
	$ROUTE	(25)	Route Profile [1]
		(SELTRP)	Selected interchanges [11]
		(3)	Directional assigned volumes and turn volumes from all iterations run

[9] Default unit number

[10] Required as input when the First Iteration Number is not 1.

[11] DDName, this data set is written only if an iteration is run which has its iteration number equal to the Selected Links Iteration number.

	(50)	Trees
	(4)	Scratch [12]
	(4)	Save Parameter cards
	(11)	Save Parameter data
$SUBAREA	(0)	Subarea Network Abstract (optional) [4]

Program Call Card

Column	Contents
1-22	$ASSIGN SELF-BALANCING
23-32	blank
33-80	ignored

OR

Column	Contents
1-22	$ASSIGN SELF-DIVERTING
23-32	blank
33-80	ignored

[12] If Subarea Assignment is used, this unit must be defined.

Parameter Cards and Data Cards

Weights Card

This card provides the user the opportunity of electing one of the following options:

1. Specifying the iteration weightings (i.e., percent each iteration is to be weighted) for the computation of the final weighted (or combined) assignment.

2. Allowing the program to compute the iteration weightings using the equilibrium technique.

If option 1 is elected, an iteration weight (i.e., an integer percent) must be specified for each iteration to be executed (i.e., Iteration 1 through the Maximum Iteration Number) such that the sum of the iteration weights is equal to exactly 100. If option 2 is selected, the iteration weights are ignored.

It is best to specify percents to load, or one or two iterations can be dropped from the weighted assignment. For 5 iterations, 15, 15, 20, 20, 30 percent are recommended.

Column	Contents
1-4	*WGT
3-6	blank
7-12	Percent weight for iteration 1
13-18	Percent weight for iteration 2
19-24	Percent weight for iteration 3
25-30	Percent weight for iteration 4
31-36	Percent weight for iteration 5
37-42	Percent weight for iteration 6
43-48	Percent weight for iteration 7
49-54	Percent weight for iteration 8
55-80	ignored

Turn Penalty Card

This card not only specifies the amount of time to be added as a penalty for each turn but provides the user the opportunity to elect a number of options available in the program. These options include:

- Specify the use of capacities rather than counts in the balancing process.
- Limit or extend the maximum number of iterations to be performed (default value is 5 iterations).
- Stop and restart the program between iterations (or between the list iteration and the weighted assignment) thereby allowing the program to be executed as a series of jobs.
- Selected Link output may be obtained from one iteration during a run. (NOTE: if the preceding multiple run option is used, Selected Link output may be obtained from each run).
- Restart the program in case a machine problem is encountered during the iteration process.
- Select the equilibrium option for producing iteration weights.
- Allows the user the option of obtaining an additional all-or-nothing assignment using weight impedances.

The format for the turn penalty card is:

Column	Contents
1-5	*TURN
6	blank
7-12	Turn penalty in hundredths of a minute (a decimal point is assumed between Columns 10 and 11). Not used if *TRN card image is included.
13-16	Optional field which may be used to take advantage of a partially built TREES data set. If the field contains the word "COPY", the partially built TREES data set will be read from Unit 49 and copied to Unit 50. Missing trees and trees containing invalid codes will be rebuilt and included in the TREES data set on Unit 50. (NOTE:

	It is important that the First Iteration Number parameter in Columns 39-42 specifies the iteration number in which the partial TREES data set on Unit 49 was built.)
17-30	blank
31-34	<u>CAP</u> - to perform a capacity restraint assignment using the old impedance function. <u>NCAP</u> - to perform a capacity restraint assignment using the new impedance function. <u>EQLB</u> - to perform an equilibrium capacity restraint assignment. The additional cards *EQLIB, *FCLCRV, *FCLPARM, and an *END card are read. The values on the *WGT card are ignored. blank - to perform a volume restraint assignment using input traffic count data.
35-37	WGT - to produce an all-or-nothing traffic assignment using weighted impedances; otherwise, this assignment is not produced.
38	blank
39-42	First Iteration number (default value = 1).
43-48	Stop Iteration number (default value = 0).
49-54	Minimum Iteration Number (default value = 3).
55-60	Maximum Iteration Number (default value = 5).
61-66	Selected Links Iteration Number (default value = 0).
67-68	blank
69-72	SKIP to skip building trees on the First Iteration Number and use the trees input on Unit 50 for this iteration. These trees will be destroyed if another iteration or the second weighted assignment is run. <u>This option cannot be used with a Subarea ASSIGN SELF-BALANCING run</u>.

73-80	blank

The parameters in Columns 39-66 are used as follows:

First Iteration Number: This parameter is used when an ASSIGN SELF-BALANCING run is to be continued from another job. This parameter should be set to one more than the previous number of iterations run. If it is greater than the Maximum iteration number, then the weighted iteration will be the next one run. If it is zero or blank, it will default to 1. If this parameter is greater than 1, then the last updated NETWORK data set and the data set from Unit 3 are required from the previous ASSIGN SELF-BALANCING run.

Stop Iteration Number: This parameter can be used to stop the ASSIGN SELF-BALANCING process at the end of any iteration except the weighted iteration. If it is left blank or zero, the process will run to completion.

Minimum Iteration Number: This parameter must be less than or equal to the maximum iteration number. The parameter performs no function.

Maximum Iteration Number: This is the maximum iteration number which will be run. If it is zero or blank, it defaults to 5. This parameter must not be greater than 8.

Selected Links Iteration Number: This is the iteration number on which the selected links parameter cards following the last *TREE card will be read, and the SELTRP data set will be written. If this output is not desired, then this field should be left blank and the selected links parameter cards will not be read.

New Turn Penalties

This card image specifies 16 turn penalties or turn prohibits. This card image specifies all combinations of the new link direction codes. The turn penalties must be in the range of 0 to 10.23 minutes. A turn prohibit is specified by -1 or any other negative value. The *TURN card image must still be present because it has several optional flags which are still examined. This card image is optional and should only be used if the 16 turn penalty/prohibit scheme is being used.

Column	Contents
1-4	*TRN
5-8	blank
9-12	Turn penalty from W to W in hundredths

13-16	Turn penalty from W to X in hundredths
17-20	Turn penalty from W to Y in hundredths
21-24	Turn penalty from W to Z in hundredths
25-28	Turn penalty from X to W in hundredths
29-32	Turn penalty from X to X in hundredths
33-36	Turn penalty from X to Y in hundredths
37-40	Turn penalty from X to Z in hundredths
41-44	Turn penalty from Y to W in hundredths
45-48	Turn penalty from Y to X in hundredths
49-52	Turn penalty from Y to Y in hundredths
53-56	Turn penalty from Y to Z in hundredths
57-60	Turn penalty from Z to W in hundredths
61-64	Turn penalty from Z to X in hundredths
65-68	Turn penalty from Z to Y in hundredths
69-72	Turn penalty from Z to Z in hundredths

Tree Selection Card [13]

This card image specifies those centroids from which trees will be built. One Tree Selection Card per subnetwork is required; a minimum of one to a maximum of 4,800 centroids is permissible. The Tree Selection Cards may specify various groups of one or more trees in any order, but proper functioning of the load process requires that the trees be built in a serial sequence order. In the format of the Tree Selection Card the six character fields are each composed of two subfields, A and B, of five characters and one character, respectively:

[13] Tree Selection Cards are not read if unit $SUBAREA is used.

Column	Contents	
1-5	TREE [5]	
6-7	blank	
8-12	Subfield A	First selection field
13	Subfield B	
14-18	Subfield A	Second selection field
19	Subfield B	
20-24	Subfield A	Third selection field
25	Subfield B	
26-30	Subfield A	Fourth selection field
31	Subfield B	
32-36	Subfield A	Fifth selection field
37	Subfield B	
38-42	Subfield A	Sixth selection field
43	Subfield B	
44-48	Subfield A	Seventh selection field
49	Subfield B	
50-54	Subfield A	Eighth selection field
55	Subfield B	
56-60	Subfield A	Ninth selection field
61	Subfield B	
62-66	Subfield A	Tenth selection field
67	Subfield B	
68-72	Subfield A	Eleventh selection field
73	Subfield B	

NOTE: A comma (,) may be entered in Column 73 if desired when subfield 68-72 is used. However, Columns 73-80 are not read; the program places the comma when entries are made in Columns 68-72.

Subfield A may contain any valid centroid number in the network. Subfield B functions as a delimiter and may contain a blank, comma, or period. Any other character will give an error message. A period used as a delimiter causes all trees built within the control range to have BCD Minimum Path Descriptions printed for inspection. This does not affect the tree building process otherwise, except it causes a delay due to writing the output.

In processing selection fields from left to right, the occurrence of two consecutive A subfields separated by a blank B subfield will initiate a control setup for inclusive tree building, beginning with the centroid specified by the first A subfield and ending with the centroid specified by the second A subfield. A comma in the second B subfield is optional for this situation, since the starting and ending centroids have been found for a search group. The occurrence of two successive B subfields containing either commas (may be implied as mentioned above) or periods, causes a single centroid to be specified; i.e., a control setup for inclusive tree building beginning and ending with the centroid specified in the intermediate A subfield.

For example, to build trees 1 through 90, with BCD Minimum Path Descriptions of trees 1 and 50, the following Tree Selection Card would be required:

Column	Contents
1-5	*TREE
6-7	bb
8-13	bbbb1.
14-19	bbbb2b
20-25	bbb49b
26-31	bbb50.
32-37	bbb51b
38-43	bbb90b

In the above example, lower case b's are used to indicate blanks.

Turn Suppression Option

The Turn Suppression Card denotes the assigned network output that is desired. Columns 7-80 are ignored.

<u>Columns 1-6</u> <u>Action</u>

*ALL	Write link volumes, turn volumes, and the vehicle-hour and vehicle-mile summary.
*LINKS	Write link volumes and the vehicle-hour and vehicle-mile summary only.

The LIST Card, LINKS Cards, EQUAL Cards, and END Card (for EQUALS Cards, not for Selected Link Cards) are read only if $SUBAREA is used.

If the SUBAREA file number is set to nonzero by a $SUBAREA, XX card, then a set of cards (i.e., LIST cards, DIRECT cards, and optional LINKS cards) which define the portion of the network within the subarea are read. These cards are:

List Card

This card (or cards) specifies a continuous ring of links (and/or centroid connectors) which completely surrounds the subarea. The LIST cards must be chained to the next LIST card but must specify one complete ring of links. For error checking, it is convenient to keep the LIST cards in order. Each card may contain up to twelve node (or centroid) numbers. A link is defined between the first and the second number on this card; between the second and third number; etc. The first zero or blank field on this card ends the data on this card, which means that a LIST card can define from one to eleven links to be in the ring.

Column	Contents
1-4	LIST
5-6	blank
7-12	First node (or centroid) number
13-18	Second node (or centroid) number
19-24	Third node (or centroid) number
25-30	Fourth node (or centroid) number
31-36	Fifth node (or centroid) number
37-42	Sixth node (or centroid) number
43-48	Seventh node (or centroid) number

49-54	Eighth node (or centroid) number
55-60	Ninth node (or centroid) number
61-66	Tenth node (or centroid) number
67-72	Eleventh node (or centroid) number
73-78	Twelfth node (or centroid) number

Direct Card

This card specifies on which side of the ring of links the subarea lies.

Columns	Contents
1-6	DIRECT
7-12	Node on the subarea ring of links
13-18	Node that is inside the subarea is connected to the first node in Columns 7-12, and is further connected to another node in the subarea.

Links Card

These cards may define other links to be included in the summaries produced after the assigned network listing of the assignment output from a subarea assignment. This card will be necessary for any part of the network inside the ring of links which is isolated from the other part of the network inside the ring of links. The most likely use for this card is for centroid connectors where the centroid only connects to the nodes (or centroids) in the ring of links. This card also may define links which cannot be surrounded by the ring of links.

Column	Contents
1-5	LINKS
6	blank
7-12	First node (or centroid) number
13-18	Second node (or centroid) number

19-24	Third node (or centroid) number
25-30	Fourth node (or centroid) number
31-36	Fifth node (or centroid) number
37-42	Sixth node (or centroid) number
43-48	Seventh node (or centroid) number
49-54	Eighth node (or centroid) number
55-60	Ninth node (or centroid) number
61-66	Tenth node (or centroid) number
67-72	Eleventh node (or centroid) number
73-78	Twelfth node (or centroid) number

Last Centroid Card

Column	Contents
1-6	*LASTC
7-12	Last (i.e. highest) centroid number for trees
13-80	ignored

Equal Cards

These cards define the sector centroids and also the centroids which are equated to the sector centroids. Trees will not be built for centroids equated to sector centroids. The exception to this is that trees will be built for all zones equated to the same zone (sector centroids). An entry is generated for each sector centroid and if it is additionally equated to itself, a warning message will be printed. The numbers to the right of the EQUAL on these cards may specify ranges of centroids by placing the beginning of the range on the card in one field and immediately following this field by a field with the last centroid of the range with a minus sign in front of the centroid number. A range must all be specified on one EQUAL card.

Column	Contents
1-4	Sector Centroid Number
5	blank
6-10	EQUAL
11-80	14 fields for centroids or centroid ranges (Format: 14I5)

End Card

This card defines the end of the EQUAL cards.

Columns	Contents
1-5	blank
6-8	END
9-80	blank

Selected Links Parameter Cards

The following parameter cards are required only if the Selected Links Iteration Number is set equal to an iteration that is run.

Selected Link Cards

A Selected Link Card must be provided for each desired link. This card also limits the number of interchanges to print. If any of the limits are omitted or zero, it will be set for the maximum number permitted. None of the options should be set greater than the allowable maximum. After any one of the three limits has been reached, the output will be terminated.

Column	Contents
1-4	*SEL
5-6	blank

7-12	A-node
13-18	B-node
19-24	Percentage of total volume to be included (range 1 to 100)
25-30	Minimum two-way volume to be allowed (range 1 to 32,767)
31-36	Number of zone pairs to be included (range 1 to 32,767)
37-80	ignored

End Selected Links

Column	Contents
1-4	*END
5-80	blank

Equilibrium Optimization Option

Column	Contents	
1-6	*EQLIB	
8-11	IMPD	This option specifies that the optimization function is the sum of the link impedance multiplied by the assigned volumes for the links which have capacities. The default optimization function is the integral of the link impedance over the volume range of zero to the assigned link volume for links with capacities.

Functional Class Impedance Update Function Card

Column	Contents
1-8	*FCLPARM
11-15	Functional Class Number 0 to 15
21-30	"A" constant for this functional class; default is 0.92
31-40	"B" constant for this functional class; default is 0.15
41-50	"C" constant for this functional class; default is 4.0 and the maximum is 30.

The default data or the parameters read are used to build the impedance update function value W for each nondirectional link with an assigned volume using the following equation:

$$W_f = A_f + B_f[\frac{V}{C}]^{D_f}$$

The default impedance update function is:

$$W_f = 0.92 + 0.15[\frac{V}{C}]^4$$

Where:

W_f	=	the value multiplied by the initial impedance for functional class *f*.
V	=	the nondirectional iteration weighted link volume.
C	=	the nondirectional link capacity.

Functional Class Impedance Update Values Header Card

Column	Contents
1-7	*FCLCRV
11-15	Functional Class Number 0 to 15

This card defines the beginning of a table of values of V/C ratio and impedance update ratio. There can be from 2 to 400 input data cards. The input data cards must be in an increasing V/C ratio order, and they must define an impedance update function which has increasing or equal values as the V/C ratio increases. The range of the V/C ratio is from 0.0 to 4.0.

Functional Class Impedance Update Values Header Card

Column	Contents
1-10	V/C ratio (0.0 to 4.0, must include a decimal point)
11-20	Impedance update value (greater than zero)

End of Impedance Update Functions

Column	Contents
1-4	*END
5-80	blank

This data card is required after the end of the last impedance update function but is not required between impedance update functions.

Normal Operation

The Weights Card, the Turn Penalty Card, optionally the *TRN card with 16 turn penalties/prohibits, and the Tree Selection Cards are read and interpreted. The following messages are printed if these cards are correct:

THE TREE CARDS HAVE ESTABLISHED THE FOLLOWING PARAMETERS

TURN PENALTY = ---.---

LINK COUNTS WILL BE USED IN THE LINK IMPEDANCE FUNCTION

or LINK CAPACITIES WILL BE USED IN THE OLD LINK IMPEDANCE FUNCTION

or LINK CAPACITIES WILL BE USED IN THE NEW LINK IMPEDANCE FUNCTION

or

EQUILIBRIUM WILL MINIMIZE VEHICLE HOURS OF TRAVEL

or

EQUILIBRIUM WILL MINIMIZE INTEGRAL VHT FUNCTION

AN INCOMPLETE TREE DATA SET WILL BE READ FROM UNIT 49 AND MISSING TREES OR TREES WITH INVALID CODES WILL BE BUILT [14]

A SECOND WEIGHTED ASSIGNMENT USING THE WEIGHTED IMPEDANCES WILL BE PRODUCED [6]

FIRST ITERATION = -

STOP AFTER ITERATION = -

MINIMUM NUMBER OF ITERATIONS = -

MAXIMUM NUMBER OF ITERATIONS = -

ITERATION TO GET SELECTED LINKS OUTPUT -

PERCENTS TO LOAD ARE -- -- -- -- --

[14] Optional

TREE BUILDING WILL BE SKIPPED AND A SET OF TREES PREVIOUSLY BUILT WILL BE USED TO ASSIGN ITERATION -

FOR SUBNETWORK - SEARCH MINIMUM PATHS FROM ZONES------ TO ------ INCLUSIVE AND OUTPUT
or
SUPPRESS OUTPUT

The appropriate portions of the messages are repeated as often as necessary to describe the various control ranges for the tree search. The steps described below are repeated for each of three to five iterations.

The link impedances are updated using nondirectional link volumes. Nondirectional capacities and ground counts are also used.

If the run is an equilibrium run, then *EQLIB, *FCLCRV, *FCLPARM, and an *END card are read after the *TURN card. If default options are used only an *END card will read for the equilibrium control and data cards.

The NETWORK data set is read. Minimum path trees are traced according to the Turn Penalty Card and Tree Selection Card, and the Trip Matrix is assigned to the minimum paths. When the iteration number is one, the entire assignment is printed. This will correspond to a minimum time path assignment if impedances have been calculated from speed and distance.

The variable C is set to either nondirectional traffic counts or nondirectional capacities depending upon which of three options is entered on the Turn Penalty Card (Columns 31-34). If C is zero for a given link (again dependent on the option selected), the impedance for that link is not changed.

The five options available for updating link impedances are:

<u>Traffic Counts</u>

$$I(n + 1) = (0.75 + 0.25*(V/C))*I(n)$$

NOTE: If $I(n - 1)/I(n) > 1.43$, $I(n + 1)$ is set to $1.43*I(n)$

<u>Capacities - OLD Impedance Function</u> (<u>CAP</u>)

The link impedance is not changed in this option unless the link volume has exceeded capacity for one or more iterations. When the link impedance is updated, the following function is used:

$$I(n + 1) = (0.75 + 0.25*(V/C))*I(n)$$

NOTE: If $I(n + 1)/I(n) > 1.43$, $I(n + 1)$ is set to $1.43*I(n)$

Capacities - NEW Impedance Function (NCAP)

$$I(n + 1) = (0.92 + 0.15\ (V(n)/C)**4)*I(1)$$

NOTE: If $I(n + 1)/I(n) > (n + 1)$, $I(n + 1)$ is set to $(n + 1)*I(n)$

The following constraints are applied for all updated impedances:

If $I(n + 1) > 10.23$ - turn penalty, $I(n + 1)$ is set to 10.23 - turn penalty

If $I(n + 1) = 0.0$ and $I(n) = 0.0$, then $I(n + 1)$ is set to 0.01

Capacities - Equilibrium Formula (EQLB)

$$I(n + 1) = (A(fc) + B(fc)\ (V(n)/C)**D(fc))*I(1)$$

Where:

A(fc)	=	a value read from the *FCLPARM card for functional class "fc" or a default value of 0.92.
B(fc)	=	a value read from the *FCLPARM card for functional class "fc" or a default value of 0.15.
D(fc)	=	a value read from the *FCLPARM card for functional class "fc" or a default value of 4.

NOTE: If $I(n + 1)/I(n) > (n + 1)$, $I(n + 1)$ is set to $(n + 1)*I(n)$

The following constraints are applied for all updated impedances:

If $I(n + 1) > 10.23$ - turn penalty, $I(n + 1)$ is set to 10.23 - turn penalty

If $I(n + 1) = 0.0$ and $I(n) = 0.0$, then $I(n + 1)$ is set to 0.01

Capacities - Equilibrium Table Look-up (EQLB)

The new impedance value is obtained from interpolating a value from the link V/C ratio between two points from the table look up values input after a *FCLCRV table of impedance update factors. The new impedance is obtained by multiplying the table look-up value by the initial impedance.

NOTE: If $I(n + 1)/I(n) > (n + 1)$, $I(n + 1)$ is set to $(n + 1)*I(n)$

The following constraints are applied for all updated impedances:

If $I(n + 1) > 10.23$ - turn penalty, $I(n + 1)$ is set to 10.23 - turn penalty

If $I(n + 1) = 0.0$ and $I(n) = 0.0$, then $I(n + 1)$ is set to 0.01

The new network containing the revised impedances is written on the NEWNET data set. The route profile and separation matrix data sets also are written. Data will also be included in the NEWNET data set from the last assignment (and all previous assignments up to a maximum of 19) describing nondirectional link volumes and the corresponding nondirectional link impedances. The following tables and summaries also are produced:

CROSS-CLASSIFICATION OF V/C FREQUENCIES FROM LAST TWO ASSIGNMENTS.[15]

CROSS-CLASSIFICATION OF LINK COUNTS BY V/C RATIO FROM LAST TWO ASSIGNMENTS.[7]

JURISDICTION SUMMARY (This table is not printed if there are functional class codes on the link data).

JURISDICTIONAL/FUNCTIONAL CROSS-CLASSIFICATION OF ASSIGNED VOLUMES (This table is not written if more than 95 percent of the links have no functional class code).

JURISDICTIONAL/FUNCTIONAL CROSS-CLASSIFICATION OF COUNTED VOLUMES (This table is not written if more than 95 percent of the links have no functional class code or if all locations in the table are zero).

JURISDICTIONAL/FUNCTIONAL CROSS-CLASSIFICATION OF LINK CAPACITIES (This table is not written if more than 95 percent of the links have no functional class code or if all locations in the table are zero).

[15] Directional Table

COMPARISON OF ASSIGNED VOLUMES WITH COUNTED VOLUMES[7]

COMPARISON OF ASSIGNED VOLUMES WITH LINK CAPACITIES[7]

COMPARISON OF ASSIGNED VOLUMES (from last assignment) WITH ASSIGNED VOLUMES FROM (assignment before last)[7]

ITERATION WEIGHTING - MULTIPLE REGRESSION ANALYSIS

The iteration number is printed in the Heading record of each of the above tables. Then the unit numbers of the NETWORK and NEWNET data sets are switched. If an iteration is run which is equal to the stop iteration number, the process stops at the end of that iteration. The maximum iterations allowed is the Maximum Iteration Number.

The percents from the *WGT card are then printed in a table entitled ITERATION WEIGHTS APPLIED. If the equilibrium option is run, the percents for each iteration are calculated by this process. The process calculates a lambda value from 0 to 1 which determines how much of the current iteration to weight with the previous weighted link volumes. One minus lambda is multiplied by the previous weighted iteration. The program calculates these for all iterations and calculates the resulting percents from each iteration.

A weighted assignment is calculated by applying the iteration weights (percentages) to their respective assigned volumes and summing. An updated Network data set is prepared which includes the weighted assigned volumes, and the weighted assignment is printed with all assigned volumes and turning movements. A set of printed tables similar to those already described for the individual iterations is written for the weighted assignment.

Using the same iteration weights applied with the assigned volumes, a set of nondirectional weighted impedances is calculated in an analogous manner. If WGT option is specified on the *TURN Card, the weighted impedances are used for one final assignment, and everything written for the calculated weighted assignment is also produced for the weighted impedance assignment. All assigned volumes and turning movements are printed for the weighted impedance assignment. Finally, the Route Profile and Corridor Intercept tables are printed, followed by a table of assigned volumes and impedances from all iterations (including the weighted assignments). The last assignment produces new data sets on the NETWORK or NEWNET, SEPARAT, and ROUTE units. A message is written at the end indicating the unit number on which the final NETWORK data set has been written:

THE FINAL LOADED NETWORK IS ON --

If the ASSIGN SELF-BALANCING run is to be restarted, this is the data set which must be Unit NETWORK on the restart run.

When a machine malfunction or an Abnormal Program Termination occurs, the following things should be considered when setting up a restart. The Unit 3 data set, the Network data set, and the First Iteration Number used in a restart of ASSIGN SELF-BALANCING must be consistent in respect to the number of iterations previously completed. The data for an iteration are written on Unit 3 before the assigned network is printed or if it is not printed, then before the Network data set is updated. The Unit 3 data set is closed and reopened after being written so that it will be complete. When ASSIGN SELF-BALANCING is restarted, Unit 3 is positioned to the end of the First Iteration Number -1 and partially checked for errors. If there are additional iterations or records written on this data set, they are ignored and written over.

The data written on a Network data set is written concurrently with the printing of the assigned network or if the printing is skipped, then it is written before the first output following the assignment of the network is printed. The writing of the nondirectional Network data set is an update process, and there is an old Network data set and a new Network data set for each iteration. Each new Network data set is closed immediately after being written. The first Network data set used by the iteration indicated by First Iteration Number parameter is Unit NETWORK. At the end of this iteration, Unit NEWNET becomes the new Network data set. Units NETWORK and NEWNET alternately become the Network data sets used on succeeding iterations. When the ASSIGN SELF-BALANCING process ends through the Stop Iteration Number option, a message "THE FINAL LOADED NETWORK IS ON X" gives the Unit number of the new nondirectional Network data set for the last iteration run. If the program ends abnormally, the Unit to use as the Network data set on a restart must be determined by the number of iterations which have completed the assignment process. If there is any doubt of which data set to use, NETWORK or NEWNET, then one or both of the data sets can be printed with the OUTPUT NETWORK program and the header records examined to determine the appropriate data set. A check should also be made to see that all links for the last node in the Network are printed. Also a network with nondirectional volumes is written to Unit 85. This data set is written when Table L1 is produced.

Error Messages

INVALID TURN PENALTY OR TREE CARD READ

The program prints this line under the card which has an error in its identification field (Columns 1-6). The correct contents for Columns 1-6 for the two cards are *TURN and *TREE. The program stops with a STOP 0 after examining both cards for errors.

ILLEGAL FIELD SEPARATION CHARACTER IN TREE CARD

This message is printed if a field separation character is found which is not a comma, a period, or a blank. The program stops with a STOP 0 after it examines the rest of the Tree Card.

---- ERROR(S) DETECTED IN ABOVE PARAMETER CARDS, EXECUTION TERMINATED

The program stops with a STOP 0.

NO VALID OPTIONS ON *EQLIB CARD

Columns 8-11 of the *EQLIB card do not contain "EQLB". This counts as one error. The program will end with a STOP 9 after examining other parameter cards.

FOR FUNCTIONAL CLASS -- A CONSTANT IS TOO SMALL = --------.-----

The "A" impedance update value on the *FCLPARM card is less than or equal zero. The program will end with a STOP 9 after examining other parameter cards.

FOR FUNCTIONAL CLASS -- D CONSTANT IS TOO SMALL = --------.-----

The "D" impedance update value on the *FCLPARM card is less than or equal to zero. The program will end with a STOP 9 after examining other parameter cards.

FOR FUNCTIONAL CLASS -- D CONSTANT IS TOO LARGE = ------.-----

The "D" impedance update value on the *FCLPARM card is larger than 30. Values of 30 and above can produce overflows in calculating the power of the V/C ratio. The program will end with a STOP 9 after examining other parameter cards.

ERROR, FUNCTIONAL CLASS = -- IS INVALID ON "--------------------------------"

The functional class on a *FCLCRV data card is less than 0 or greater than 15. The program will end with a STOP 9 after examining other parameter cards.

FUNCTIONAL CLASS -- IMPEDANCE UPDATE FUNCTION ERROR
----- NON-INCREASING ERRORS FOR THE VCR INPUT

The V/C ratios on data cards following a *FCLCRV card are not in an increasing order. The program will end with a STOP 9 after examining other parameter cards.

FUNCTIONAL CLASS -- IMPEDANCE UPDATE FUNCTION ERROR
----- NON INCREASING ERRORS FOR THE VALUE INPUT

The impedance multiplier on data cards following a *FCLCRV card are not in an increasing order. The program will end with a STOP 9 after examining other parameter cards.

------ ERRORS IN EQUILIBRIUM INPUT DATA CARDS, STOP 9, EXECUTION TERMINATED

This message gives a count of errors on the equilibrium data cards, and the program executes a STOP 9.

**** TURN TYPE FOR NODE -----, ILLEGAL, = -----------, SET TO 28

This message should be printed only if an input error has occurred in reading the NETWORK data set. It indicated that no turning movements will be saved for the node in the message.

NODES .GT. 16000, = ------

This message is self-explanatory. The program stops with a STOP 0 at this point.

NUMBER OF SUBNETS FROM BINARY NETWORK = ---

NUMBER OF SUBNETS FROM VOLUME FILE = ---

NUMBER OF SUBNETS FROM TREE FILE = --- ARE NOT EQUAL, LOAD DELETED.

There is a conflict in the number of subnetworks in one of the three data sets. The program does not load the network, and ends with a STOP 7.

FIRST NODE, LAST CENTROID INFORMATION FROM TREE FILE AND VOLUME FILES DOES NOT AGREE, OR ONE IS GREATER THAN ------ OR THE NUMBERS ARE NOT INCREASING ORDER LOAD DELETED.

```
FILE FSTND(1) LSTND(1) FSTND(2) LSTND(2) FSTND(3) LSTND(3) FSTND(4)
LSTND(4)

LOAD   -----   -----   -----   -----   -----   -----   -----   -----

TREE   -----   -----   -----   -----   -----   -----   -----   -----
```

The first number printed is the last centroid number of the last subnetwork from the Network data set. No centroid number from the trip matrix or Path data set can be larger than this number. Another condition for which the message is written is when the first node number for a subnetwork in the Trip Matrix data set is not equal to the corresponding first node number from the Network data set. These conditions both indicate that the Path, Trip Matrix, and Network data sets are not for the same network. The program continues after this message is printed.

NUMBER OF TURNING MOVEMENTS TO SAVE EXCEED 60000, = ------

This message indicates that more than 60,000 locations in the turning movement table are needed to load the network. The actual number of locations needed is printed after the comma. The program ends with a STOP 50 after this message. One way to bypass this problem is to use the LOAD SELECTED LINKS program with the option to output the loaded network without the turning movements.

I/O ERROR ON TREE FILE AFTER A NODE -----

This indicates that there was a possible error on reading a path record. The program continues and attempts to use the path record.

I/O ERROR ON TRIP FILE AFTER A NODE -----

The program skips the Trip Matrix record and continues.

ABEND U101

This abend code indicates that one of the turn type code numbers, which has a valid range of from 1 to 28, has been destroyed during the load process. The job abended without a dump. The probable cause of this error would be an I/O error from the Path data set.

ABEND U102

This error code could indicate several things, some of which are: (1) the three data sets which are input to this program are not for an identical network; (2) there has been an I/O error in reading one of the data sets; or (3) a CPU error has occurred. The job is abended without a dump.

TRMV ERR

This error message is printed in the loaded network output if one or more of the turning movements cannot be calculated. This message indicates that the network has not been loaded correctly. This message could occur if a network which contains errors is assigned.

MORE THAN TWO LINKS FROM NODE -----

This message occurs during the processing of the Route Profiles and indicates that a node has three or more links with the same route code. Only two of the links will be retained. This will cause links to be lost in the Route Profile output for this route.

ROUTE -- HAS NO ENDS, A LINK WILL BE CHOSEN AS A STARTING POINT.

The links included in this route form a closed loop. The link with the lowest node number will be chosen as the starting point for the Route Profile output.

ERROR ON UNIT 3, EXECUTION TERMINATED WITH STOP 21.

There was an error in reading the parameter record from Unit 3 when the First Iteration Number on the *TURN card was not 1. The program is terminated with STOP 21.

ERROR IN PARAMETER RECORD FROM UNIT 3, EXECUTION TERMINATED WITH STOP 21 PARAMETER RECORD.

There was an error in the values in the parameter record. The values in the parameter record are printed. The program execution is terminated with STOP 21.

ERROR, MISSING RECORDS OR I/O ERROR ON UNIT 3, ------ RECORDS READ ------ RECORDS WERE TO BE READ EXECUTION TERMINATED WITH STOP 21.

PARAMETER RECORD

The number of records to skip position Unit 3 is determined from the values of the parameter record and the number of iterations previously run. Then the program tries to skip this number of records on Unit 3. If the end of data set condition or an I/O error is encountered, then this message is printed. This message will be printed when the First Iteration Number is set larger than one more than the previous number of iteration run.

INVALID *WGT CARD READ

The card following the program call card does not contain *WGT in Columns 1-4.

ERROR IN ABOVE PERCENTS, SUM IS ----

The percents in the *WGT card for iterations 1 to the Maximum Iteration Number do not sum to 100. The sum is printed.

REGRESSION CALCULATION OF ITERATION WEIGHTINGS IS NOT ALLOWED; USE THE EQUILIBRIUM OPTION

The old option of calculating the iteration weighting percents has been removed. The user should use the new equilibrium option for calculating the weightings.

SORT ERROR; LINK READ FROM NET IS ----- ----- LINK READ FROM SORT IS ----- -----

The Anode-Bnode of the link read from the sort did not match the expected Anode-Bnode of the link being processed. This is probably a sort error or a JCL error. The Anode and Bnode of the expected link are printed in the first two dash fields. The Anode and Bnode of the link from the sort are printed in the third and fourth dash fields. Up to three sort records are used for each link. The first sort record contains assignments 1 through 6 (including the lanes as the first assignment).

SORT ERROR IN SECOND SORT RECORD; LINK READ FROM NET IS ----- ----- LINK READ FROM SORT IS ----- -----

The Anode-Bnode of the link read from the sort did not match the expected Anode-Bnode of the link being processed. This is probably a sort error or a JCL error. The Anode and Bnode of the expected link are printed in the first two dash fields. The Anode and Bnode of the link from the sort are printed in the third and fourth dash fields. Up to three sort records are used for each link. The second sort record is written when 7 assignments have been done (including the lanes as the first assignment).

SORT ERROR IN THIRD SORT RECORD; LINK READ FROM NET IS ----- ----- LINK READ FROM SORT IS ----- -----

The Anode-Bnode of the link read from the sort did not match the expected Anode-Bnode of the link being processed. This is probably a sort error or a JCL error. The Anode and Bnode of the expected link are printed in the first two dash fields. The Anode and Bnode of the link from the sort are printed in the third and fourth dash fields. Up to three sort records are used for each link. The third sort record is written when 15 assignments have been done (including the lanes as the first assignment).

STOPPING FOR 100 OR MORE SORT ERRORS WITH STOP 9

This error message indicates that there have been 100 SORT error messages printed while printing table L1.

Order of Input Card Images for a Non Subarea Run

$ASSIGN SELF-BALANCING	
*WGT	
*TURN	
*TRN	(only if 16 turn penalty/prohibits)
*ALL or *LINKS	
*SEL	(only if selected links iteration)
.	
.	
.	
*END	(only if selected links iteration)

Order of Input Card Images for a Non Subarea Run
with the Equilibrium Option

$ASSIGN SELF-BALANCING		
*WGT		
*TURN	EQLB	- -
*TRN		(only if 16 turn penalty/prohibits)
*ALL or *LINKS		
*SEL		(only if selected links iteration)
.		
.		
.		
*END		(only if selected links iteration)
*EQLIB IMPD		(only if the optimization function is the sum of link impedance times the assigned volume for links with capacities)

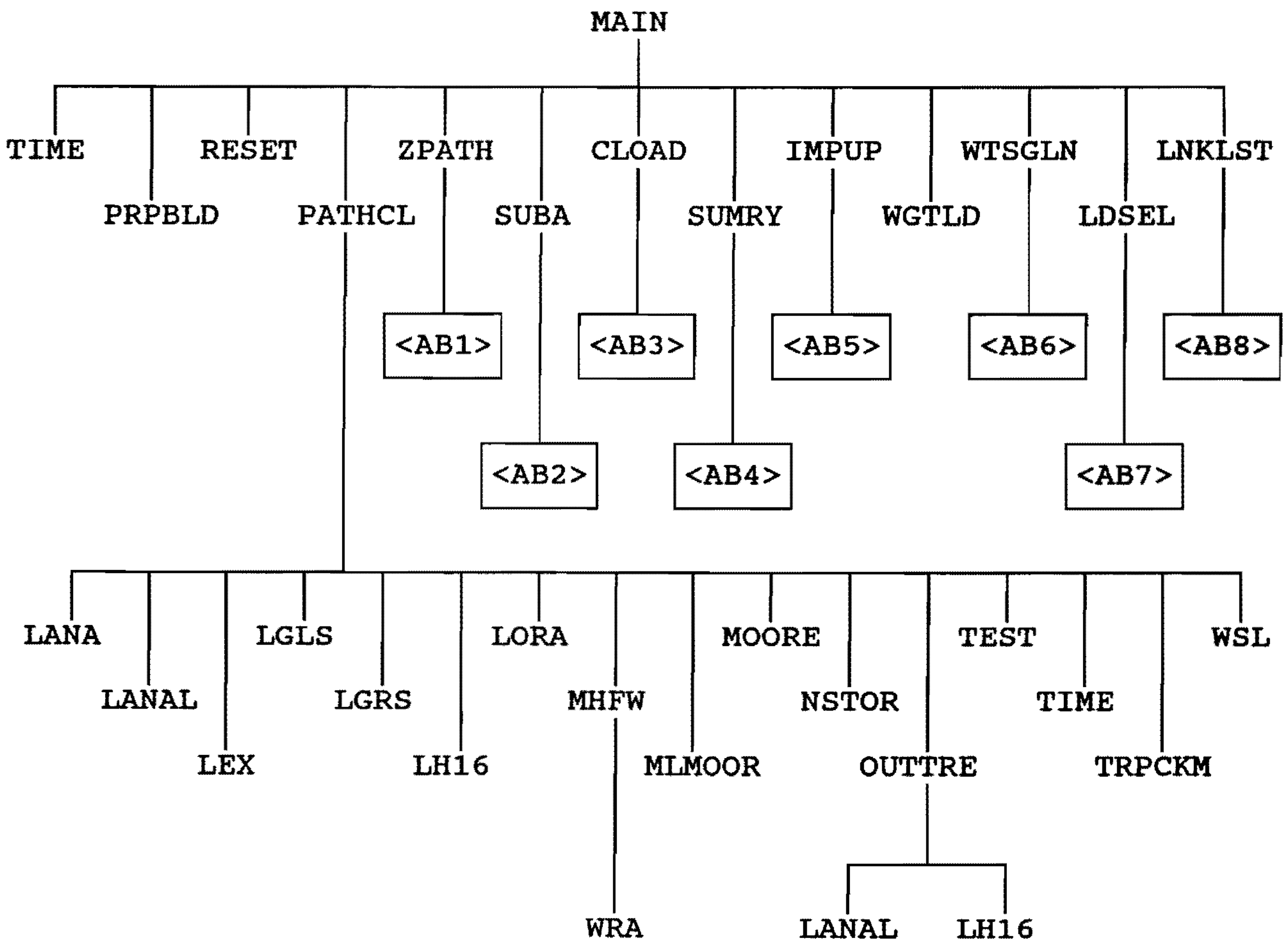
```
*FCLCRV     5
0.0      1.0
0.5      1.0
0.7      1.5
1.5      3.0
2.5      30.0
3.9      785.
*FCLCRV     6
0.0      1.0
4.0      9000.
*FCLCRV     7
0.0      1.0
1.0      1.0
2.0      20.
3.0      400.
4.0      4000.
*END
```

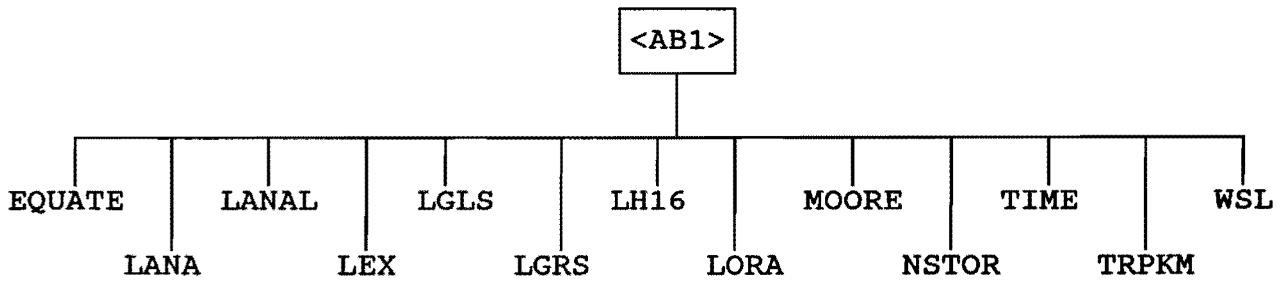
<u>Order of Input Card Images for a Subarea Run</u>

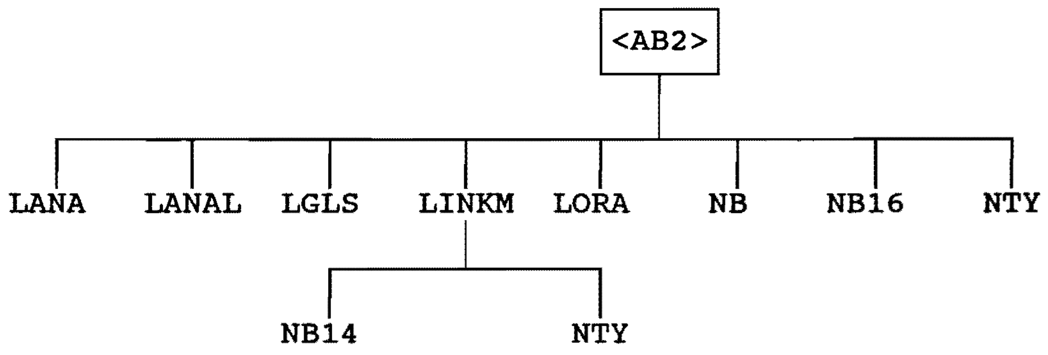
```
$SUBAREA, xx

$ASSIGN SELF-BALANCING

*WGT

*TURN

*TRN                          (only if 16 turn penalty/prohibits)

*ALL or *LINKS

*LASTC

XXXX EQUAL
       END

LIST

       .
       .
       .

DIRECT
```

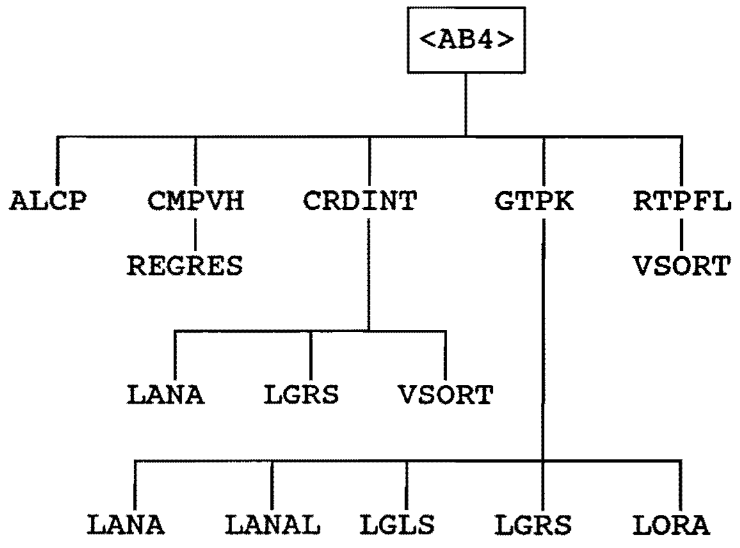
LINKS	(optional)
.	
.	
.	
END	
*SEL	(only if selected links iteration)
.	
.	
.	
*END	(only if selected links iteration)

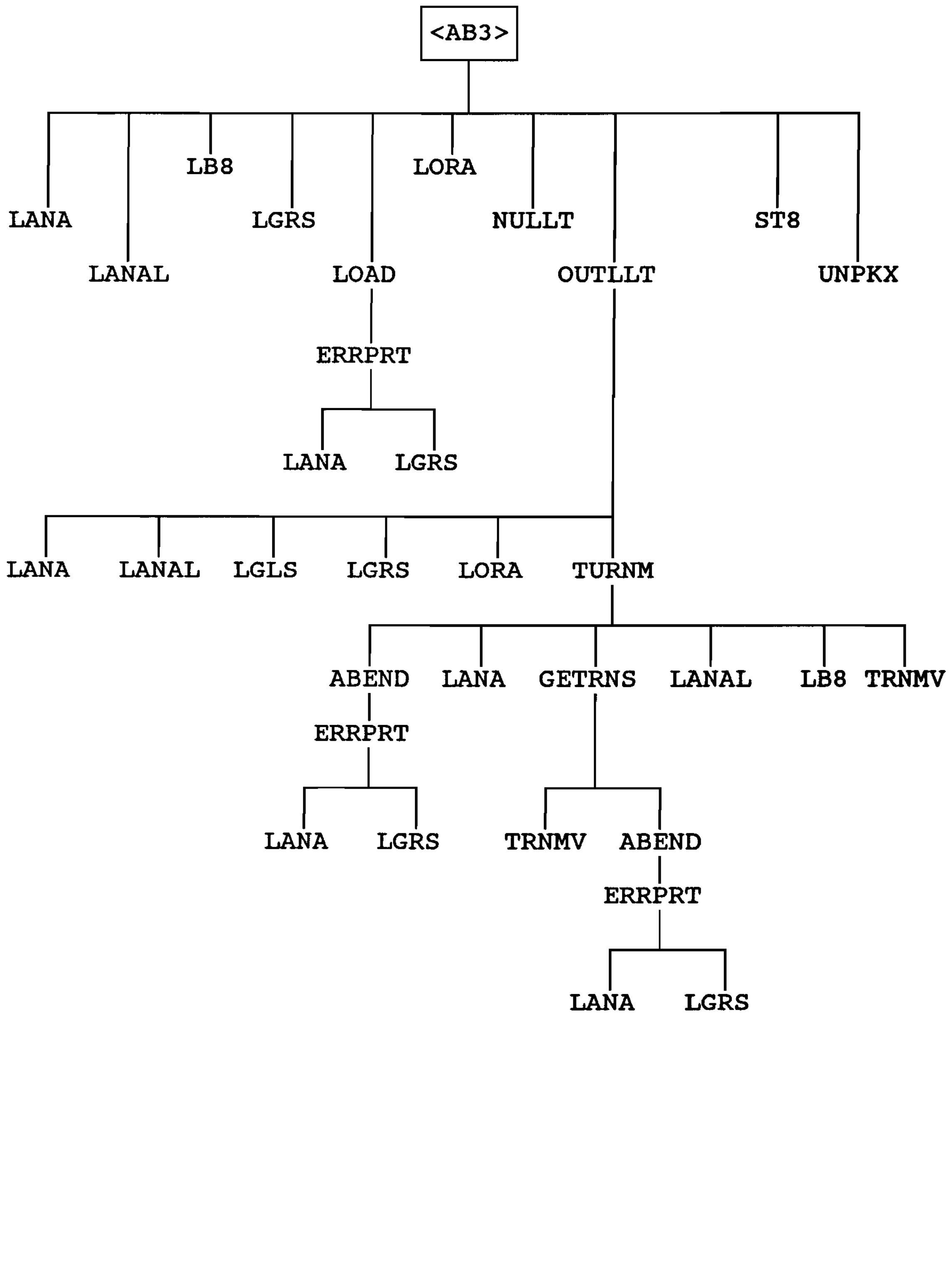
Sequence of Subroutines Called if not Equilibrium



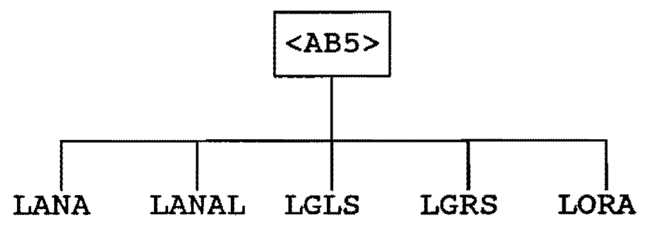

<AB1>
EQUATE
LANA
LANAL
LEX
LGLS
LGRS
LH16
LORA
MOORE
NSTOR
TIME
TRPKM
WSL

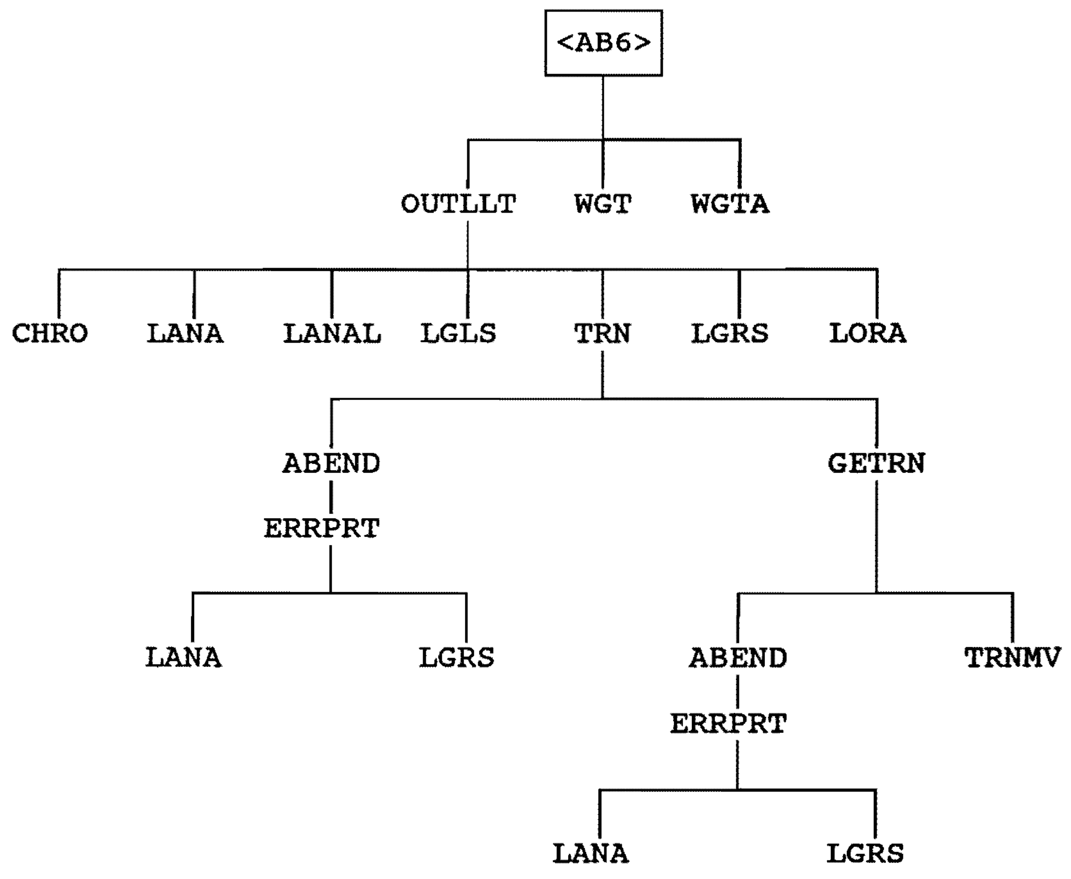

<AB2>
LANA
LANAL
LGLS
LINKM
LORA
NB
NB16
NTY
NB14
NTY

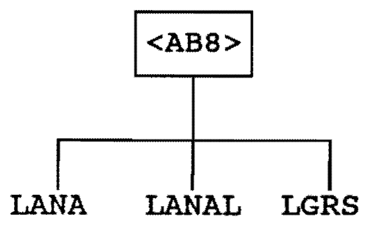

<AB4>
ALCP
CMPVH
CRDINT
GTPK
RTPFL
REGRES
VSORT
LANA
LGRS
VSORT
LANA
LANAL
LGLS
LGRS
LORA

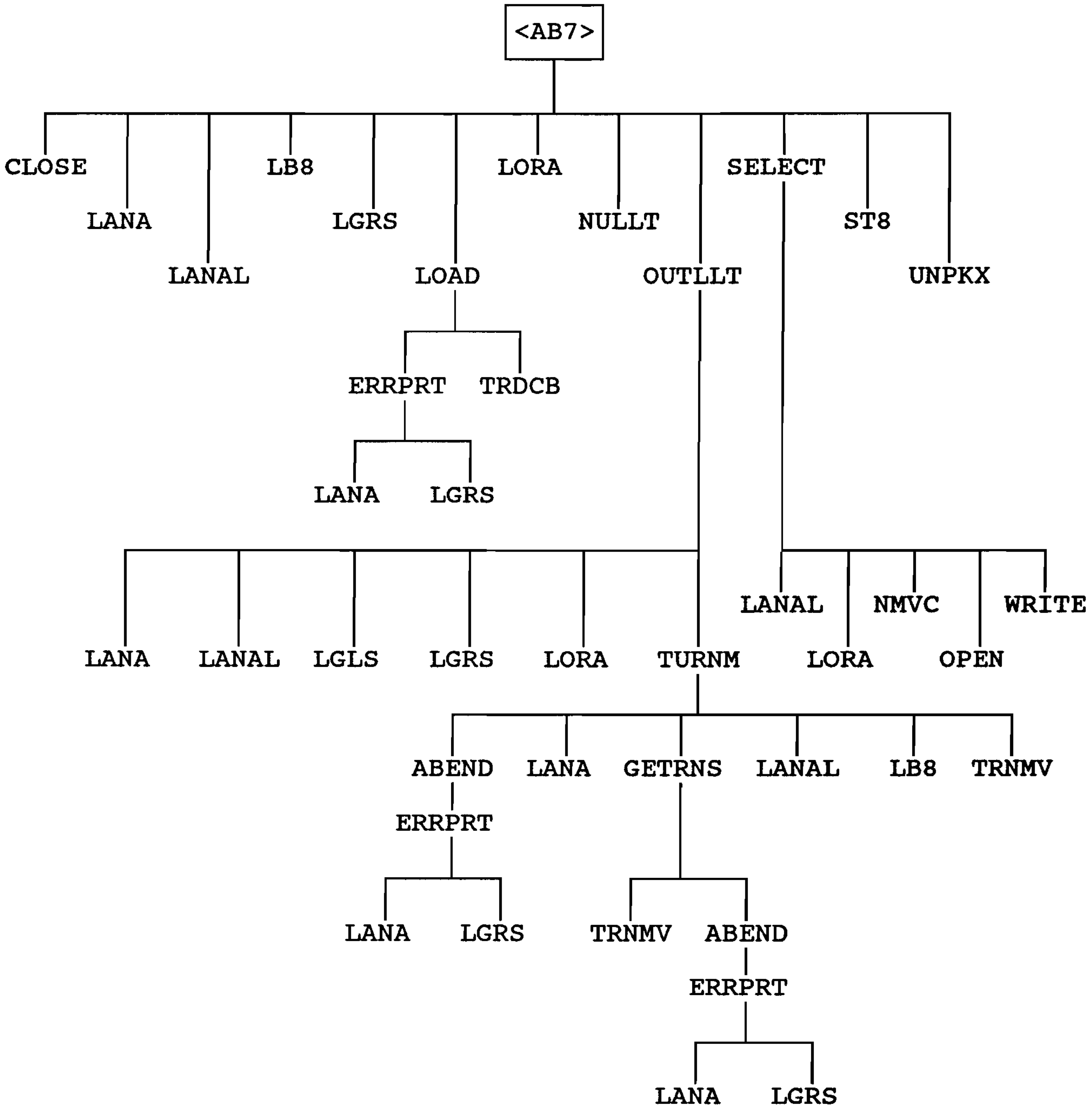


<AB3>
LANA
LANAL
LB8
LGRS
LOAD
LORA
NULLT
OUTLLT
ST8
UNPKX
ERRPRT
LANA
LGRS
LANA
LANAL
LGLS
LGRS
LORA
TURNM
ABEND
LANA
GETRNS
LANAL
LB8
TRNMV
ERRPRT
LANA
LGRS
TRNMV
ABEND
ERRPRT
LANA
LGRS

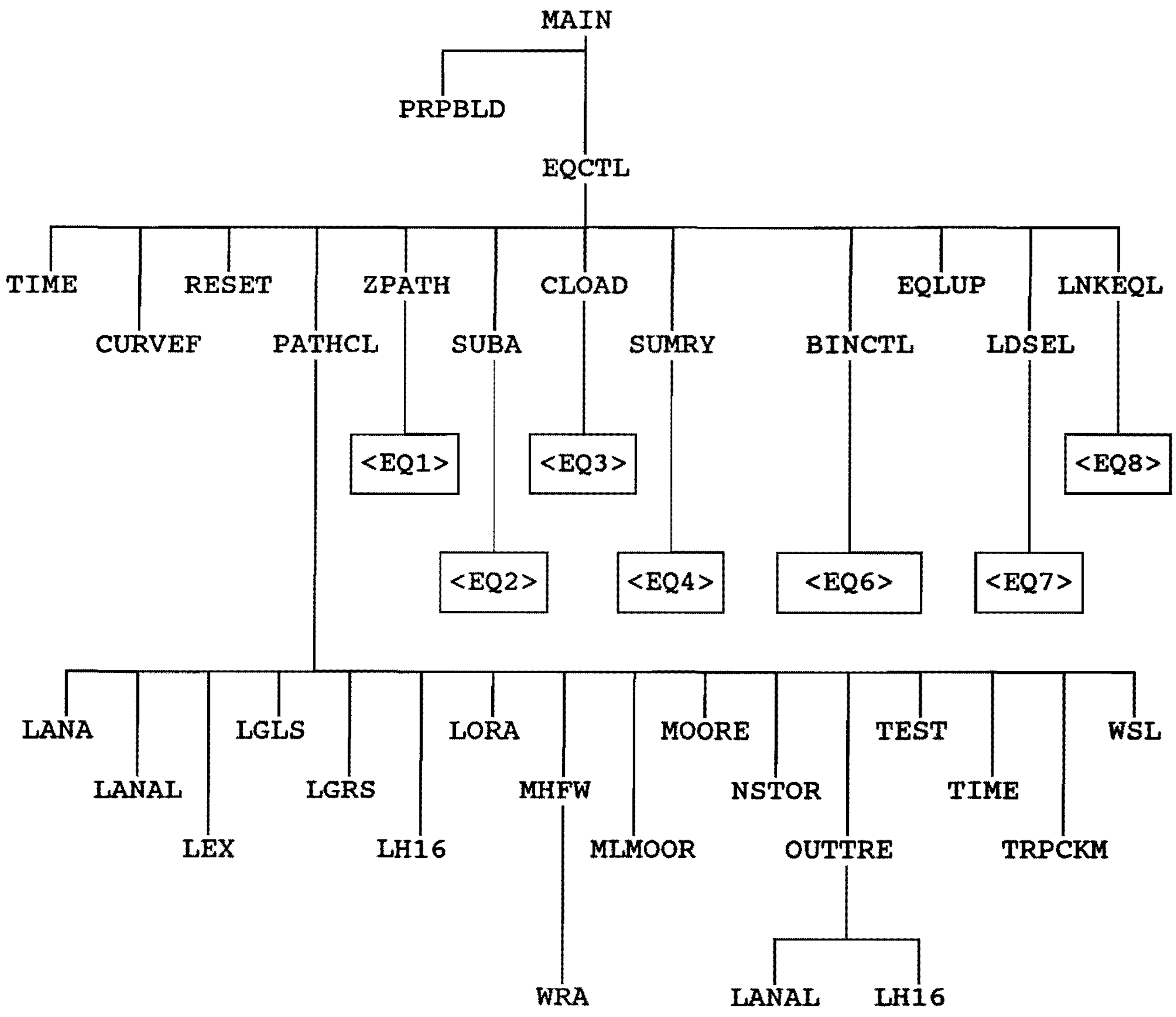

<AB5>
LANA
LANAL
LGLS
LGRS
LORA

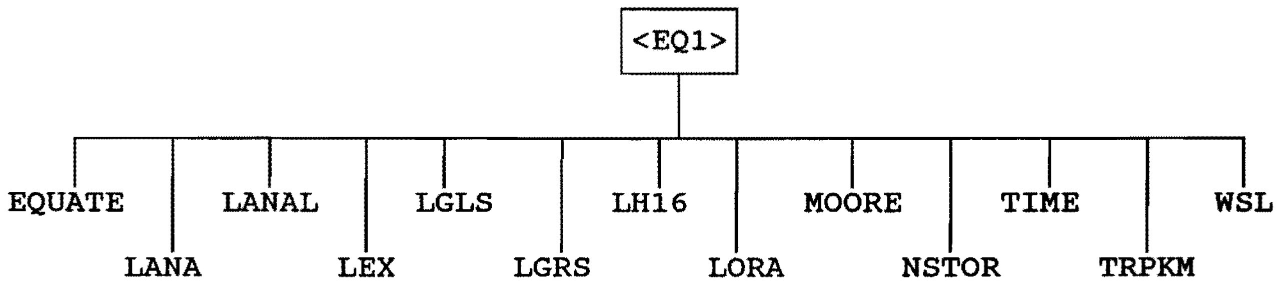

<AB6>
OUTLLT
WGT
WGTA
CHRO
LANA
LANAL
LGLS
TRN
LGRS
LORA
ABEND
GETRN
ERRPRT
LANA
LGRS
ABEND
TRNMV
ERRPRT
LANA
LGRS

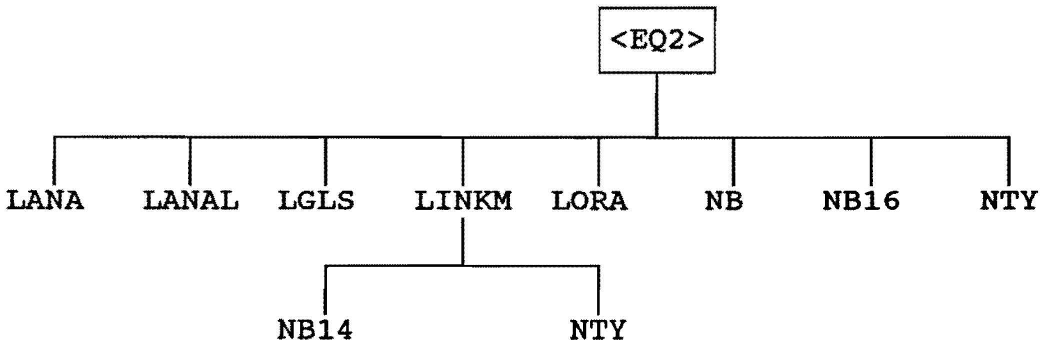

<AB8>
LANA
LANAL
LGRS

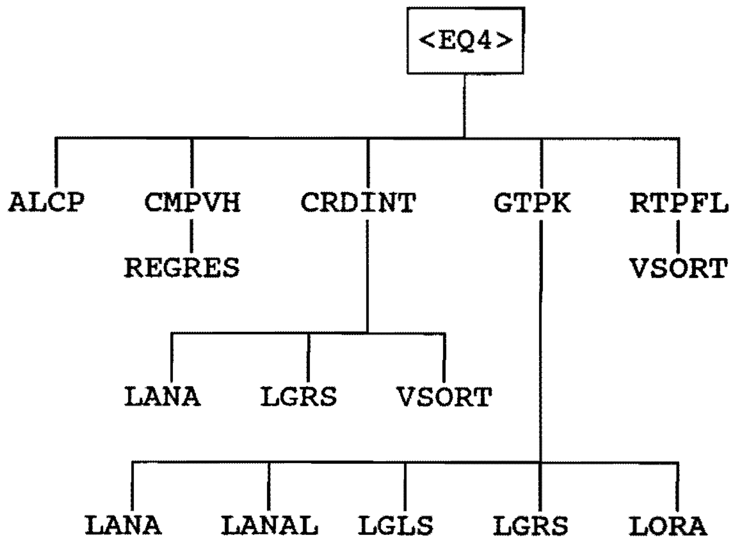

<AB7>
CLOSE
LANA
LANAL
LB8
LGRS
LOAD
ERRPRT
TRDCB
LANA
LGRS
LORA
NULLT
OUTLLT
LANA
LANAL
LGLS
LGRS
LORA
TURNM
ABEND
ERRPRT
LANA
LGRS
LANA
GETRNS
TRNMV
ABEND
ERRPRT
LANA
LGRS
LANAL
LB8
TRNMV
SELECT
LANAL
LORA
NMVC
OPEN
WRITE
ST8
UNPKX

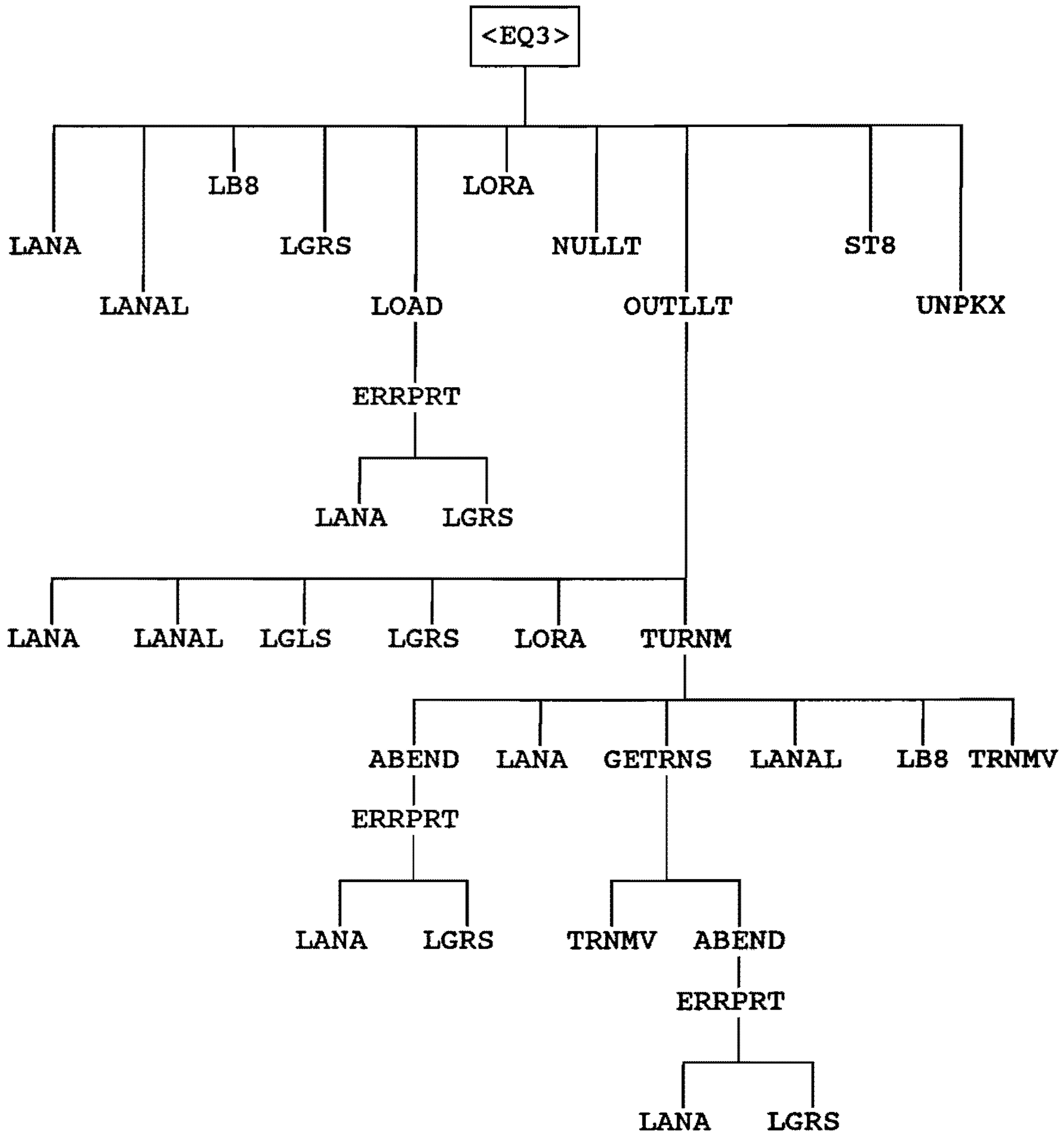
Sequence of Subroutines Called if Equilibrium

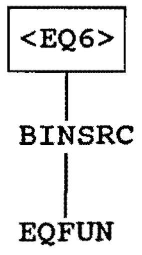


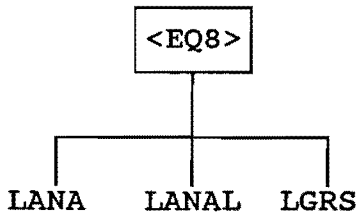

<EQ1>
EQUATE
LANA
LANAL
LEX
LGLS
LGRS
LH16
LORA
MOORE
NSTOR
TIME
TRPKM
WSL

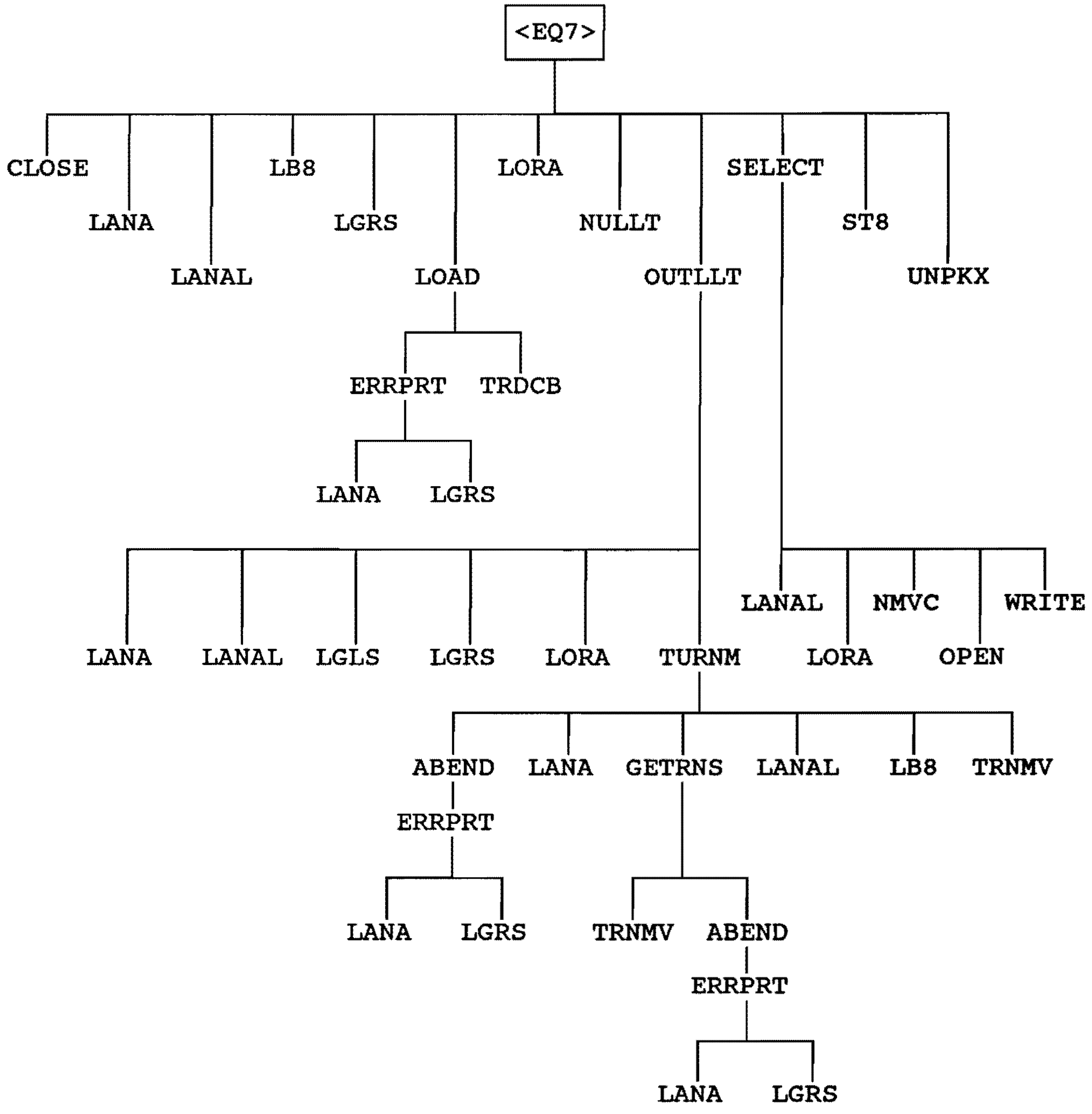

<EQ2>
LANA
LANAL
LGLS
LINKM
LORA
NB
NB16
NTY
NB14
NTY


<EQ4>
ALCP
CMPVH
CRDINT
GTPK
RTPFL
REGRES
VSORT
LANA
LGRS
VSORT
LANA
LANAL
LGLS
LGRS
LORA


<EQ3>
LANA
LANAL
LB8
LGRS
LOAD
LORA
NULLT
OUTLLT
ST8
UNPKX
ERRPRT
LANA
LGRS
LANA
LANAL
LGLS
LGRS
LORA
TURNM
ABEND
LANA
GETRNS
LANAL
LB8
TRNMV
ERRPRT
LANA
LGRS
TRNMV
ABEND
ERRPRT
LANA
LGRS


<EQ6>
BINSRC
EQFUN


<EQ8>
LANA
LANAL
LGRS


<EQ7>
CLOSE
LANA
LANAL
LB8
LGRS
LOAD
LORA
NULLT
OUTLLT
SELECT
ST8
UNPKX
ERRPRT
TRDCB
LANA
LGRS
LANA
LANAL
LGLS
LGRS
LORA
TURNM
LANAL
LORA
NMVC
OPEN
WRITE
ABEND
LANA
GETRNS
LANAL
LB8
TRNMV
ERRPRT
LANA
LGRS
TRNMV
ABEND
ERRPRT
LANA
LGRS

Summary of Individual Subroutines

MAIN: In the execution of ASSIGN SELF-BALANCING without the Equilibrium option, the MAIN program performs the following steps:

1. The logical variable PEAK in the PK labeled common is set to false to allow subroutines ALCP, CLOAD and IMPUP to work using a nondirectional capacity restraint.
2. The subroutine PRPBLD is called to read the *WGT, *TURN, *TREE and loaded network output specification card images.
3. If this is a restart of ASSIGN SELF-BALANCING, subroutine RESET is called to position FORTRAN Unit 3 correctly.
4. If the SUBAREA data set is zero, the subroutine PATHCL is called to build minimum path trees for this iteration otherwise the subroutine ZPATH is called to build trees if the subarea option is used with ASSIGN SELF-BALANCING.
5. If this is not the iteration for selected links, the subroutine CLOAD is called to load the trees built for this iteration. If this is the selected link iteration, the subroutine LDSEL is called, instead, to load the trees.
6. If the SUBAREA data set is not zero, the subroutine SUBA is called to prepare the network data set for summaries.
7. The subroutine SUMRY is called to produce summary tables for this iteration.
8. Steps 3 through 7 are repeated for the number of iterations.
9. Subroutine WGTLD is called.
10. Subroutine WTSGLN is called.
11. If an extra weighted iteration is specified, then subroutines PATHCL and then CLOAD are called.
12. If the SUBAREA data set is not zero, then the subroutine SUBA is called to prepare the network data set for subroutine LNKLST.
13. Subroutine SUMRY is called to produce summaries.

14. Subroutine LNKLST is called to produce a nondirectional summary of the links with updated link impedances.

MAIN: In the execution of ASSIGN SELF-BALANCING with the equilibrium option, the MAIN program performs the following steps:

1. The logical variable PEAK in the PK labeled common is set to false to allow subroutines ALCP, CLOAD and IMPUP to work using a nondirectional capacity restraint.

2. The subroutine PRPBLD is called to read the *WGT, *TURN, *TREE and loaded network output specification card images.

3. The subroutine EQCTL is called.

ABEND: This subroutine builds a message which contains the FORTRAN internal statement number of the line that called it. Next the subroutine calls subroutine ERRPRT. Then the subroutine abends with a user code of 100. This subroutine is written in Assembly language and is a CSECT named ABEND which is contained in the deck with subroutine GETVOL.

ALCP: This subroutine calculates a multiple regression analysis to determine the iteration weighting for an ASSIGN SELF-BALANCING process and prints the results of this analysis. The summations for the regressions are done in subroutine GTPK. Only the links with a non-zero count (or capacity depending on which is specified) are considered and centroid connectors are ignored. The count (or capacity) is the dependent variable and the assigned nondirectional volumes from each of the iterations are the independent variables in the analysis. This subroutine is called by subroutine SUMRY. This subroutine contains the labelled common HEADER, CAPRES and TBL. This subroutine is written in FORTRAN IV.

BINCTL: This subroutine reads the network data set and obtains the previous weighted assignment nondirectional link volumes, the last assignment nondirectional link volumes, the first link impedances, and the link capacity. The subroutine calls BINSRC to do a binary search. The subroutine then calculates the percents to produce the weighted link volumes for this iteration and saves them.

BINSRC: This subroutine performs a binary search to find the optimum combination of the last iteration to the previous weighted iteration volumes. The minimization function is either the sum of the new link impedances times the assigned volume or the sum of the integral of the link impedance update function over the assigned volume. The impedance used in this minimization function is determined from the new trial weighting being used in BINSRC.

The subroutine initially calculates the optimization function for the end points of 0.0 and 1.0 (which are 0 percent of the previous weighted and 100 percent respectively). It then splits this range and calculates the optimization function for a lambda of 0.5 (50 percent). It then throws away the end point with the highest optimization value and the point of 0.5 replaces that end point. The process is repeated 17 times to develop the estimated lambda value.

CHRO: This subroutine moves character data from non-character variables to character variables. The first argument is the destination of the move. The second argument is the origin of the move. The third argument is the length of the move. This function is an entry point in the Assembly language CSECT NMVC.

CLOAD: It reads the network from Unit NETWORK and modifies this to the format needed for the LOAD subroutine. It initializes the nondirectional link volume array and the turn volume array to zero. It calls subroutine SELECT if it is a LOAD SELECTED LINKS run. It reads the trip matrix and the paths data sets. These are assumed to be in sort on the origin zones. It calls subroutine LOAD to load trips in the network if there is both a tree record and one or more trip records for an origin zone. After the network is loaded, CLOAD calls subroutine SVLOAD to save the loaded network on Unit 3 if a ASSIGN SELF-BALANCING run is in iterations 1 through 5. CLOAD then calls subroutine OUTLLT to print the loaded network.

CLOSE: This subroutine closes data set SELTRP.

CMPVH: This subroutine prints the Jurisdiction Summary or the Jurisdictional/FUNCTIONAL Cross-classification Tables and the three Comparison of Assigned Volumes with link volumes, Counted volumes, and Capacities.

CRDINT: This subroutine calls VSORT (which sorts the corridor intercept records) and prints the corridor intercept tables.

EQCTL: This subroutine controls the equilibrium option of ASSIGN SELF-BALANCING.

1. The subroutine CURVEF is called to read impedance function update parameters or user impedance function curves by functional class. The user also can input an option to use an optimization function of the sum of the vehicle hours of travel for links with capacities.

2. If this is a restart of ASSIGN SELF-BALANCING, subroutine RESET is called to position FORTRAN Unit 3 correctly.

3. If the SUBAREA data set is zero then the subroutine PATHCL is called to build minimum path trees for this iteration: otherwise, the subroutine ZPATH is called to build trees if the subarea option is used with ASSIGN SELF-BALANCING.

4. If this is not the iteration for selected links then the subroutine CLOAD is called to load the trees built for this iteration. If this is the selected link iteration, the subroutine LDSEL is called instead to load the trees.

5. If the SUBAREA data set is not zero then the subroutine SUBA is called to prepare the network data set for summaries.

6. The subroutine SUMRY is called to produce summary tables for this iteration.

7. If this is iteration 2 or later, then the BINCTL subroutine is called to find a new optimum lambda for combining the present iteration with the previous weighted iteration.

8. The subroutine EQLUP is called to put the new impedances in the network data set which will be used for the next iteration assignment.

9. Steps 3 through 8 are repeated for the number of iterations.

10. Subroutine SUMRY is called to produce summaries.

11. Subroutine LNKEQL is called to produce a nondirectional summary of the links with updated link impedances. The subroutine prints the Anode, Bnode, Distance, and Functional Class for each link. Also by iteration the weighted nondirectional link volume. The final weighted volume is not listed as a separate value since it is already listed in the last iteration.

EQFUN: This subroutine calculates the equilibrium minimization function.

EQLUP: This subroutine updates the link impedances using the new weighted assignment volumes.

EQUATE: The primary function of this subroutine is to establish zone to sector zone equivalences. EQUAL card images containing the sector zone numbers and the zone numbers which are in those sectors are read and an array is established containing the zone to sector zone equivalences. If any zone in the network has not been equivalenced to a sector, that zone is equivalenced to itself which then indicates it is in the subarea window or the boundary surrounding it. If no EQUALS cards are encountered, all zones are equivalenced to sector one. This subroutine is written in FORTRAN IV VS 77.

ERRPRT:	This subroutine prints error messages from assembly language subroutines. The first argument is an array of length 17. The first index of the first argument is the address where ERRPRT was called from. The next 16 elements of this array are the 16 general purpose registers at the time ERRPRT was called. The second argument is the home zone if it applies, otherwise it is zero. The third argument is a character variable of length 50. This subroutine is written in FORTRAN IV VS.
GETRN:	This subroutine gets the weighted turn volumes which were saved and places them in the turn volume matrix.
GETRNS:	This subroutine gets the turn volumes which were saved and places in the turn volume matrix.
GTPK:	This subroutine prints the V/C cross-classification table if there are two or more assignments on Unit NETWORK. It computes the summations necessary for the tables printed by subroutine CMPVH and for the curve fit printed by subroutine ALCP. It saves corridor intercept information in core in labeled common CD. It writes route profile records on Unit ROUTE. If logical variable SUM is true, GTPK calculates weighted nondirectional volumes and updates the network data set writing it on Unit NETWORK. All comparisons and tables are made from the weighted nondirectional volumes if SUM is true.
IMPUP:	This subroutine uses nondirectional volumes when run with ASSIGN SELF-BALANCING. This subroutine implements the new capacity restraint function when no weights are input. When no weights are input for ASSIGN SELF-BALANCING with the new impedance update function, then the percents to load are calculated by regression by subroutines GTPK and ALCP at the end of each iteration. This subroutine then reads Unit NET, which is the first argument of IMPUP, and updates the link impedances using the percents calculated from the regression for calculating the weighted link volumes. The updated network is written to Unit NNET, which is the second argument of IMPUP.
LANA:	This integer function does a logical 'and' of the first argument and the second argument. This function is an entry point in the Assembly language CSECT LOPS.
LANAL:	This integer function does a logical 'and' of the first argument and a fixed mask of X'00003FFF'. This extracts 14-bit integers in the range of 0 to 16383

from other data. This function is an entry point in the Assembly language CSECT LOPS.

LB8: This integer function indexes into an array of 8-bit integers and returns the value. The first argument is the array and the second argument is the index. This function is an entry point in the Assembly language CSECT NMVC.

LDSEL: This is an entry point in subroutine CLOAD. It sets the Selected Links option to true, then loads a network with selected links output to data set SELTRP. This subroutine is written in FORTRAN I VS 77.

LEX: This integer function does a logical exclusive 'or' of the first argument and the second argument. This function is an entry point in the Assembly language CSECT LOPS.

LGLS: This integer function does a logical left shift. The first argument is the 32-bit data to be shifted and the second argument is the number of bits to shift left. This function is an entry point in the Assembly language CSECT LOPS.

LGRS: This integer function does a logical right shift. The first argument is the 32-bit data to be shifted and the second argument is the number of bits to shift right. This function is an entry point in the Assembly language CSECT LOPS.

LH16: This integer function does a shift left logical 16 followed by a shift right logical of 16 on the half word argument. This extracts a 16-bit integer in the range of 0 to 65,535 from the half word. This function is an entry point in the Assembly language CSECT NMVC.

LINKM: This subroutine reads the card images which describe a subarea for focusing. It checks that the links on these card images are actually in the network. It then flags the links as BOUNDARY, AREA, or ADDED which are in these card images. This subroutine is written in FORTRAN IV VS 77.

LNKEQL: This subroutine lists the data by nondirectional link for the 1 to 8 iterations of ASSIGN SELF-BALANCING and outputs Table L1. This subroutine prints A-node, B-node, Distance, Functional Class by link. Also by iteration the nondirectional Link Volume, the Volume to capacity ratio code (a single character), the link impedance and the link speed.

LOAD: This subroutine loads a trip record by adding each trip interchange volume to all of the directional link volumes in the path connected between the origin and destination zones of the trip interchange. Some turn volumes are also summed in this process. This subroutine also writes a record on Unit

	SELTRP for each selected link crossed in loading each trip interchange volume.
LORA:	This integer function does a logical 'or' of the first argument and the second argument. This function is an entry point in the Assembly language CSECT LOPS.
MHFW:	This subroutine moves an array of half words to an array of full words. The subroutine changes -1 inputs to an output value of X'OOFFFFFF'. Other values which were not -1 are divided by 2. The subroutine then calls subroutine WRA to write the resulting full word array. This subroutine is written in Assembly language and is the only subroutine in this deck.
MLMOOR:	This subroutine builds one minimum time path tree each time it is called. It also sums the distance along this minimum time path. This subroutine is written in Assembly language and is the only control section in this deck.
MOOR:	This control section builds one minimum path tree each time it is called. Its entry point is MOORE.
NB:	This subroutine extracts the node number and flags from a link. The first argument is the link contained in a full word. The second argument returned by NB is the node number of the link. The third argument is also returned by NB and is one or more flags indicating if the link is in the area. This subroutine is written in Assembly language and is the CSECT name.
NB14:	This integer function does a logical 'and' of a half word argument and a fixed mask of X'00003FFF'. This extracts a 14-bit integer in the range of 0 to 16,383 from the half word. This function is an entry point in the Assembly language CSECT NB.
NB16:	This integer function does a logical 'and' of a half word argument and a fixed mask of X'0000FFFF'. This extracts a 16-bit integer in the range of 0 to 65,535 from the half word. This function is an entry point in the Assembly language CSECT NB.
NMVC:	This subroutine is a character move subroutine. The first argument is the destination character variable. The second argument is the destination offset in bytes. The third argument is the origin offset in bytes. The fourth argument is the length of the move in bytes. The fifth argument is the origin character variable. Arguments 2, 3, and 4 are full word arguments while the first and fifth arguments can have any alignment. This subroutine is written in Assembly language and is a CSECT.

NSTOR:	This subroutine stores a value into a half word array. This first argument is the half word array. The second argument is an index into the array. Bits 6 through 31 of the third argument are stored into the indexed half word array. This subroutine is written in Assembly language and is an entry point in CSECT NMVC.
NTY:	This integer function obtains a half word argument which it then shifts right by 14 bits. Then the shifted argument is logical anded[16] with the value X'00000003'. This extracts a 2 bit integer in the range of 0 to 3 from the half word. This function is an entry point in the Assembly language CSECT NB.
NULLT:	This subroutine changes the turn codes in the first argument for the number of zones set in the second argument. The turn code is saved in bits 3 through 7 of the high order byte of each full word. The turn code is set to the value of 28 which indicates that no turns should be saved for this node zone. The name of this subroutine is derived from null turns. This function is an entry point in the Assembly language CSECT LOPS.
OPEN:	See ASORT.
OUTLLT:	This subroutine controls the printing of the loaded network. It prints page headings, calls subroutine TURNM to get the link volumes and turn volumes for a node, and formats the nondirectional link volume, nondirectional link volumes, and turn volumes.
OUTTRE:	This subroutine prints one tree each time it is called.
OUTWLT:	This subroutine prints the loaded network for one segment of the loaded network. It calls subroutine TRN to calculate the turn volumes for one node. It reads the node numbers and the node names from Unit NETWORK, and it writes the updated Network data set with the weighted assignment volumes added on Unit NEWNET.
WGT:	This subroutine multiplies a group of volumes by an integer percent and places the results in another array.
PATHCL:	This is the control subroutine for building trees. It defines arrays used by subroutines called from this division. It reads the network into core from Unit NETWORK and changes it to the form used by the tree builder subroutine. It controls the building of trees, the printing of trees, the packing of the paths, and writes the paths data set and the separation matrix data sets and the MILESEP separation matrix.

[16] Boolean algebra operation.

PRPBLD:	This subroutine reads the *TURN card and the *TREE cards which specify the turn penalty and the trees to be built and printed. The COPY parameter is also specified on the *TURN card if it is used.
REGRES:	This subroutine performs a linear regression analysis and prints the results of this analysis.
RESET:	This subroutine repositions FORTRAN Unit 3 to the correct place to do a restart of ASSIGN SELF-BALANCING. This subroutine is written in FORTRAN IV VS 77.
RTPFL:	This subroutine reads the route profiles from Unit ROUTE and prints the route profile tables.
SELECT:	This subroutine reads the parameter cards of LOAD SELECTED LINKS. For each *SEL card it writes one record on Unit SELTRP, and it marks both of the one-way directional links as selected. This subroutine also reads one of the following parameter cards: *ALL, *LINKS, or *NONE. If the *LINKS card is read, this subroutine sets all turn codes in memory to 28. If the *NONE card is read, a logical variable is set to specify that the loaded network will not be printed.
ST8:	This subroutine saves the last byte of a full word in the address accessed by the first argument. The location where the byte is saved is indexed by the second argument. The data saved is the fourth byte of the third argument. This function is an entry point in the Assembly language CSECT NMVC.
SUBA:	This subroutine reads the card images that describe a subarea of a network. Then this subroutine extracts the subarea part of the network to the Unit SUBAREA data set. This subroutine is written in FORTRAN IV VS 77.
SUMRY:	This is the control program for the summaries produced after an assignment. The subroutines called by SUMRY are determined by 3 logical variables. One of the logical variables, SUM, if true causes GTPK to produce a weighted assignment on Unit NETWORK and produce all tables and comparisons from this weighted assignment. Subroutine ALCP is called only if logical variable RES is true. If logical variable RTP is false, then the corridor intercept and route profile tables are skipped.
TEST:	This subroutine tests whether the data contained in a path record contains any values of 6. A path record contains relative link indices of the path back to the origin. There are only six ways out of each node maximum. The relative index starts at 0 for the first link. Only values between 0 and 5 will be used for link indices. A value of 7 is used to indicate the origin zone. The third argument is set to a count of the number of 6 path indices found; a count of

	zero indicates no error. This subroutine is written in Assembly language and is an entry point in the TRPCKM CSECT.
TIME:	This subroutine returns the time of day in units of 0.01 of a second.
TRDCB:	This is the DCB (data control block) for the SELTRP data set. It is defined as an entry point in CSECT CLOSE which is part of the LOAD deck. It is written in Assembly language.
TRN:	This subroutine gets the weighted nondirectional volumes, and the weighted turn volumes saved. It also calculates the other weighted turn volumes and marks which turn volumes should not be printed because of one-way links.
TRNMV:	This subroutine adds 2 indices together and gets the assigned volumes indexed by the sum from a full word array.
TRPCKM:	This subroutine packs an array of path indices from 16-bit integers to ten 3-bit integers per word. The control section also contains the entry point TEST which checks to see that an array of packed path contains no indices of 6.
TURNM:	This subroutine gets the nondirectional volumes and turn volumes saved. It also calculates the other turn volumes and marks which turn volumes should not be printed because of one-way links.
UNPKX:	This subroutine unpacks the path indices and places them in full words.
VSORT:	This subroutine sorts records in core. The first argument in the calling sequence is the address of the array of records to be sorted. The second argument is the number of records. The third argument is the length of each record in bytes (must be between 1 and 256 bytes). The fourth argument is the length of the sort key in bytes (must be between 1 and 256 bytes) which cannot be longer than the record length. The sort key starts at the first byte of the record. The sort key is treated as an unsigned binary number and the records are sorted into ascending order on the sort keys.
WGTA:	This subroutine multiplies a group of volumes by an integer percent and adds the results into another array.
WGTLD:	This subroutine converts the positive weights output from a regression of the iteration volumes from ASSIGN SELF-BALANCING into a set of percents which sum to 100. This subroutine is written in FORTRAN IV VS 77.
WRA:	This subroutine writes a record of separations for one tree for the centroids only. This subroutine is called by subroutine MHFW.

WRITE:	This subroutine writes one record to the data set whose DDname is SELTRP. This subroutine has one argument which is the address of a location containing 18 bytes of data to be written as the next record. This subroutine is written in Assembly language, and it is an entry point in the CLOSE CSECT which is part of the LOAD deck.
WSL:	This subroutine writes a record of separations for one tree for the centroids only.
WTSGLN:	This subroutine reads Unit 3 and, using the weights for each iteration, sums up the weighted nondirectional link volume and reverses nondirectional link volumes and turn volumes for one segment in memory. It then rewinds Unit 3 and calls subroutine OUTWLT to print this segment of the loaded network. It repeats the above steps for other segments. The line counter used by subroutine OUTWLT to print page headings is only initialized for the first call to OUTWLT.
ZPATH:	This subroutine controls building trees for BUILD AREA TREES. This subroutine calls subroutine EQUATE, and then it builds trees for sector centroids not equated to sector centroids. This subroutine is written in FORTRAN IV VS 77.