1981

PMVA - Purdue Mean Value Analysis Program User's Guide

Jeff Brumfield

Report Number:
81-383

PMVA

Purdue Mean Value Analysis Program

User's Guide

Jeff Brumfield

Department of Computer Sciences
Purdue University
West Lafayette, Indiana 47907

CSD-TR-383

April 1981

Abstract. PMVA is a program for analyzing closed product-form queueing network models. A model is input to the program using a keyword oriented network description language. Performance measures can be computed by invoking one of several analytic algorithms based on mean value analysis. Characteristics of a network may be selectively changed and the network reevaluated.

CONTENTS

1. INTRODUCTION

PMVA is a program for analyzing closed product-form queueing network models. A queueing network is input to the program using a network description language. This keyword oriented language allows concise, yet readable, descriptions of networks. Once specified, a network may be analyzed for various customer populations using one or more solution techniques. Characteristics of a network may be selectively changed and the network reevaluated.

The program consists of an input routine, several solution routines, and an output routine. The input and output routines communicate with the solution algorithms through a standard interface. This allows additional algorithms to be implemented as new commands which use the existing input/output routines.

Syntax of Input Data

All input is free format. That is, any number of blanks may be used between data items to improve readability. End-of-line characters are ignored. Four types of data items are read by the program: keywords, symbolic names, numbers, and special characters.

Keywords are alphabetic strings which have a special meaning to the program. Many keywords are the initial letters of common queueing theory terms (e.g., FCFS for First Come First Serve). A list of all keywords recognized by the program appears in Appendix A. All letters in keywords are upper case.

User-defined symbolic names may be of any length and may contain any characters except delimiters. Keywords should not be used as symbolic names. Only the first ten characters of a symbolic name will be stored by the program.

Because numbers are read using standard Pascal input routines, all integer and real numbers must conform to Pascal syntax. In particular, at least one digit must precede the decimal point in a real number. Since negative numbers are never valid input, all numbers should be unsigned. Wherever a real number is valid in the input, a fraction of the form "num1/num2" may appear.

Special characters, along with the blank and end-of-line characters, serve as delimiters. Two of the special characters have a common use throughout the input stream. The semicolon terminates each command and each section of the network description. The comma is used to separate the items of a list.

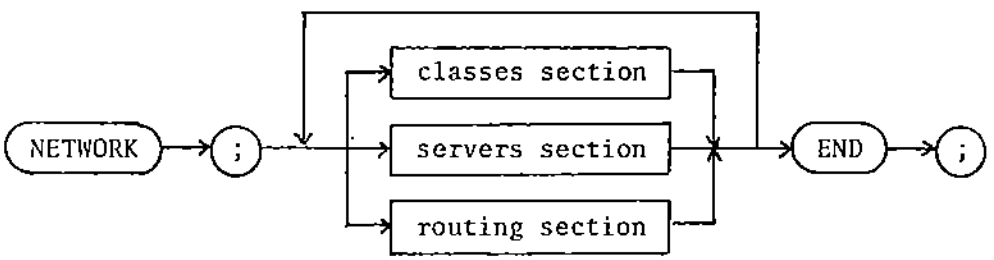
Comments may appear between any two data items. A comment is any string of characters except a double quote enclosed within double quotes.

2. DEFINING A QUEUEING NETWORK MODEL

2.1 The NETWORK Command

The NETWORK command is used to define a queuing network model. The network description is validated and converted to an internal representation for use by the various solution algorithms. In the current implementation, only one network may be stored at any time. When a second NETWORK command is read, the first network description is lost.

Syntax. The syntax of the NETWORK command is:



Detailed descriptions of the three sections appear in the following pages.

Order of sections. There may be multiple occurrences of each section. However, since all customer class and service center names must be defined before they are referenced in other sections, it is most natural to specify the classes section first, the servers section second, and the routing section last.

Limitations on complexity of network. The complexity of networks which can be analyzed (i.e., the maximum number of service centers, customer classes, etc.) is determined by the amount of storage allocated in the program. If, as the network description is being read, any limit is exceeded a diagnostic message containing the value of the limit will be issued.

Since static storage allocation is used, the program must be recompiled to handle more complex networks. Appendix B lists each limit, its value as currently specified in the program, and

a brief description of its use.

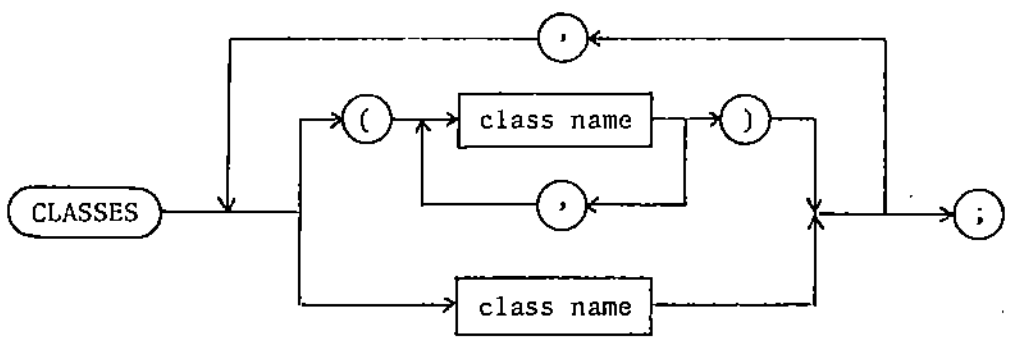
Tabular description of network. After the network is validated and some preprocessing is performed, the network description is presented in tabular form. Although the exact format of the tables depends on the characteristics of the network, all parameters which were read are included in the tables.

2.2 The CLASSES Section

In order to accurately represent a real system, it is often necessary to divide the customer population into classes. All customers of the same class are modeled using an identical set of characteristics. Customers of different classes may have different service times at most types of service centers and different routing probabilities. The CLASSES section is used to define the names of the different customer classes. For simple models in which all customers are alike, the CLASSES section may be omitted.

If the routing of customers through the network is not specified for a multi-class model in which class switching occurs, then the CLASSES section is also used to group the classes into chains. A chain is a set of customer classes such that a customer of any class in the chain may become a customer of any other class in the chain by switching classes.

Syntax. The syntax of the CLASSES section is:



Class names. The names of the customer classes must be unique in their first 10 characters. The order in which class names are specified determines the order in which performance measures are reported by class. If the CLASSES section is omitted, the model is assumed to have a single, unnamed class.

Chains. If no parentheses appear in the list of class names, each customer class is assumed to belong to a unique chain (i.e., there is no class switching). If two or more classes are

members of a chain, their names must be specified within the same set of parentheses. If the ROUTING section is included as part of the network description, the chains will always be determined from the routing.

Examples. A model having three customer classes representing batch, time-sharing, and transaction processing workloads could contain the following statement:

```
CLASSES  BATCH, TSO, IMS;
```

Unless the ROUTING section specifies differently, each class will be in a separate chain (i.e., there will be no class switching).

In the next two examples we assume that the ROUTING section will be omitted and the CLASSES section will be used to specify the chains. If there are three customer classes and there is no class switching, then the following two specifications are equivalent:

```
CLASSES  A, B, C;

CLASSES  (A), (B), (C);
```

The following CLASSES section specifies six customer classes grouped into three chains. The first is comprised of the single class A, the second consists of the three classes B, C, and D, and the third has two classes, E and F:

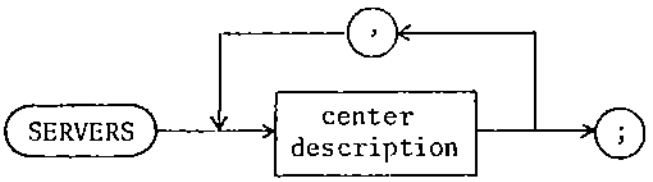
```
CLASSES  A, (B, C, D), (E, F);
```

Customers of class A never change class. Customers of classes B, C, and D may switch among any of these three classes but may not switch into classes A, E, or F. Finally, customers of classes E and F may switch only between these two classes.

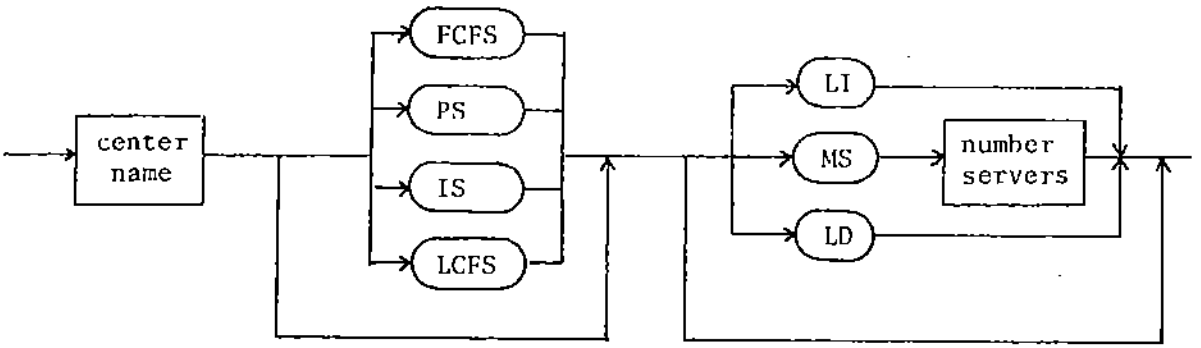
2.3 The SERVERS Section

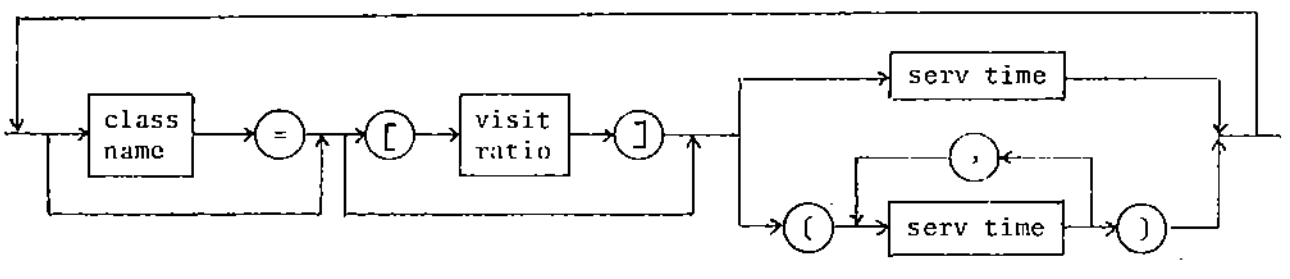
The servers section is used to name and describe each service center in the network. Included in the description are the queuing discipline, service type, and service time information. If visit ratios are known, they may also be specified in this section.

Syntax. The syntax of the SERVERS section is:



The syntax of each center description is:





Service center names. The names of the service centers must be unique in their first 10 characters. The order in which service centers are specified determines the order in which performance measures are reported by service center.

Queuing disciplines. The queuing disciplines supported are first come first serve (FCFS), processor sharing (PS), delay or infinite server (IS), and last come first serve preempt/resume (LCFS). If the queueing discipline is omitted, FCFS is assumed.

Service types. A service center may contain one or more servers. The mean service times of customers may or may not depend on the number of customers at the service center. The

following three types of service are allowed.

A load independent (LI) center contains a single server; customer service times are independent of the queue length. A multi-server (MS) center contains two or more identical load independent servers. A load dependent (LD) center contains a single load dependent server. The service times of customers at a load dependent service center depend on the total number of customers at the center.

In another variation of load dependency, a customer's service time depends on the number of customers of the same class present at the service center. This type of load dependent service center is not currently supported.

Although a multiple server service center may be simulated by a single load dependent server, it notationally simpler and algorithmically more efficient to use the MS feature. If the service center type is omitted, LI is assumed.

Restriction. Infinite servers must be load independent. The multi-server feature is not valid for an infinite server.

Specification by class. Service time information (including the optional visit ratios) is specified by customer class. All customer class names must have been defined previously in the CLASSES section. If the CLASSES section was omitted (indicating that there is only one customer class), the class name and equals sign are also omitted.

If the service time information is identical for all customer classes which visit the service center, the keyword ALL may be used in place of the class name. The service time information which follows the equals sign is then applied to all classes.

If some, but not all, of the classes have the same service time information, a default set of values may be defined by specifying the keyword DEF in place of the class name in the first class specification. The service time information in this specification is initially assigned to all classes. Other class specifications may follow to redefine the values for particular classes.

Visit ratios. Visit ratios must be specified for all service centers and classes or for none of them. If visit ratios are not specified, they will be computed from the routing description.

Service times. The type of service time information which must be specified depends on the service center type. For load independent and multi-server service centers, the mean service time of the customer class is specified. For multi-server service centers, this is interpreted as the mean service time at one

of the servers.

For a load dependent service center, the mean service times when there are 1, 2, ... customers at the center must be specified. If there are more customers at the center than the number of service times specified, the service time is assumed to remain constant at the last value specified. The dependency of the mean service times upon the load must be proportional for all classes. That is, the ratio of the mean service time when there are n customers present to the mean service time when there is one customer present must be the same for all classes at a load dependent service center.

At a FCFS service center, the service time information must be the same for all classes which visit the center. If service rates are known instead of service times, the service times can be specified as 1.0/rate.

Examples. The following service center descriptions might appear in a single class model of a terminal driven computer system:

```
TERMINALS   IS     LI   4.000,
CPU         PS     LI   0.025,
DISK        FCFS   LI   0.050,
TAPE        FCFS   LI   0.150;
```

In order to model a dual processor in a system with two workloads, we might use the following service center description:

```
CPU   PS   MS 2   BATCH=0.064   TIMESHARE=0.037
```

Because of an efficient head scheduling algorithm, the mean service time of requests at a disk drive may decrease with increased load. This could be incorporated into a model by representing the disk as a load dependent FCFS server:

```
DISK FCFS LD ALL=(0.0250, 0.0225, 0.0214, 0.0208, 0.0203, 0.0200)
```

Note that each of the preceding descriptions would be terminated by a comma (or perhaps a semicolon) if it were included as part of a SERVERS section.

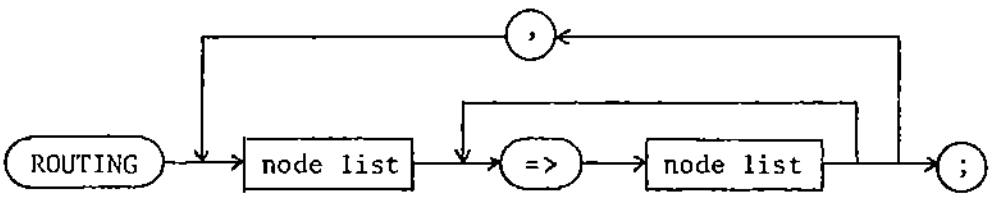
2.4 The ROUTING Section

The ROUTING section describes how customers move among the service centers. After departing from a service center, a customer may move to any service center (including the one he just left) with some fixed probability. If there is more than one customer class, the routing probabilities may be different for each class. Also, a customer may change from one class to

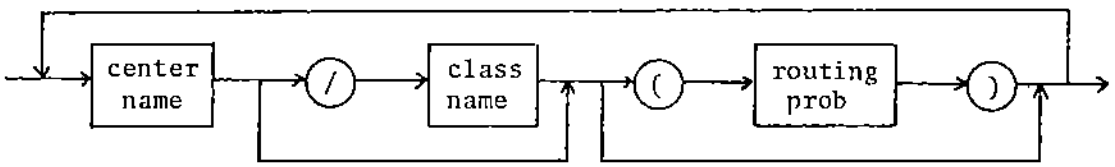
another when moving between two service centers.

The ROUTING section may be omitted only if visit ratios have been specified in the SERVERS section and equivalence classes have been specified in the CLASSES section (if class switching occurs).

Syntax. The syntax of the ROUTING section is:



The syntax of each node list is:



Customers are routed from each service center/class combination on the left hand side of an arrow (=>) to each service center/class combination on the right hand side of the arrow according to the probability associated with the destination service center/class.

Service center names. All service center names appearing in the ROUTING section must have been defined previously in the SERVERS section.

Routing probabilities. Each routing probability must be a real number in the range 0.0 to 1.0. For each customer class, the routing probabilities from each service center which that class visits must sum to 1.0. All routing combinations not described in the ROUTING section are assumed to occur with zero probability. Routing probabilities whose decimal representation contains many digits can often be specified more concisely in fractional form (i.e., num1/num2).

Routing probabilities must be specified for every service center/class combination in a node list or for none of them. If no routing probabilities are specified, the routing is assumed to be uniform. That is, each routing probability defaults to the reciprocal of the number of items in the list. Although specifying routing probabilities on the left hand side of the first

arrow in a routing descriptor is syntactically correct, the numbers have no meaning and are ignored by the program.

Customer class names. All customer class names appearing in the ROUTING section must have been defined previously in the CLASSES section. If not specified, the class associated with the first service center in a routing descriptor defaults to the first class name listed in the CLASSES section. If no classes were specified, the class defaults to the single unnamed class.

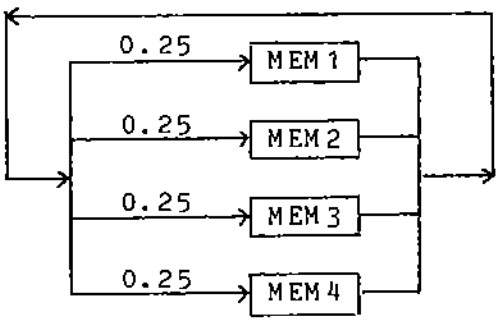
The class name associated with any other service center in a list defaults to the class associated with the previous service center in the list. If the service center is the first in a list, its class defaults to the class associated with the first service center in the previous list.

Examples. Default values for customer classes and routing probabilities have been chosen to simplify the specification of routing. For example, in single class networks class names never need to be specified. In multi-class networks with no class switching, a class name needs to be specified only for the first service center in each statement. For example,

```
CPU/BATCH => DRUM (0.3) DISK (0.5) TAPE (0.2) => CPU
```

might describe the routing of batch jobs in a central server model of a computer system.

The default values allow routing probabilities to be omitted when customers are routed uniformly among a set of service centers. In the following simple model of a memory controller, processors (the customers) are assumed to require access to memory locations in one of four memory banks (the servers) with equal probability.



The routing statement

```
MEM1 MEM2 MEM3 MEM4 => MEM1 MEM2 MEM3 MEM4
```

would describe this.

The default values also allow routing probabilities to be omitted when the routing is deterministic. The routing of customers in a cyclic network with four service centers could be described by the single statement:

```
SERV1 => SERV2 => SERV3 => SERV4 => SERV1
```

2.5 Error Detection and Recovery

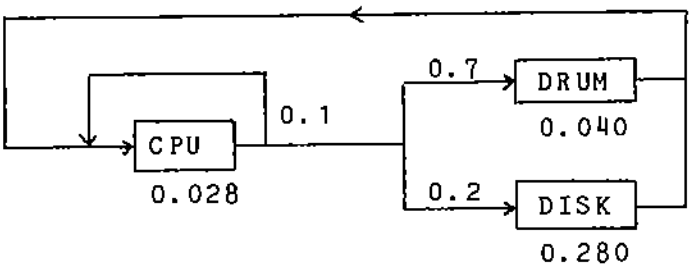
The program attempts to detect all errors which may occur when describing a network. Simple syntax errors are detected as the input is parsed; consistency errors can be detected only by examining the entire network description. Appendix D lists all diagnostic messages which may be issued.

If an error is detected as the network description is being read, all input is ignored up to the next comma or semicolon. If the program is being used interactively, the line containing the error can usually be reentered. If an error is detected after the END keyword has been entered, the CHANGE command must be used to correct it.

2.6 Complete Examples

This section presents three examples of complete network definitions. The input of any of these descriptions to the program would cause all the parameters to be listed in tabular form. The network could then be analyzed at various customer populations using any of the available algorithms.

Example 1. This example of a single class central server model of a computer system is taken from BUZE73. All devices are modeled as first come first serve, load independent service centers. Routing probabilities and mean service times are given in the following diagram.

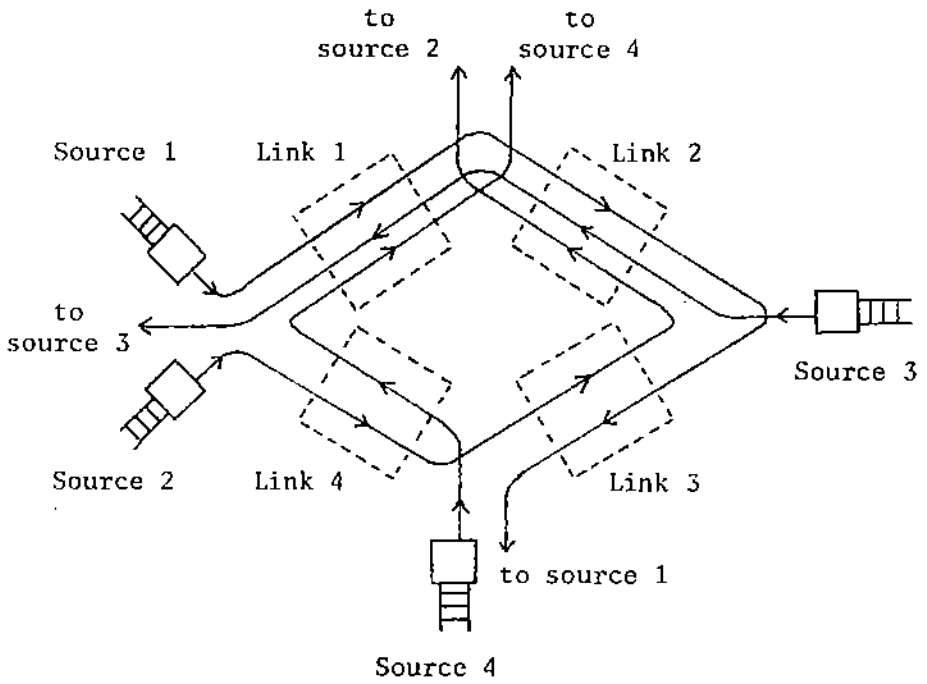


The following PMVA input describes this model:

```
NETWORK;
SERVERS
   CPU   FCFS  LI  0.028,
   DRUM  FCFS  LI  0.040,
   DISK  FCFS  LI  0.280;
ROUTING
   CPU => DRUM (0.7) DISK (0.2) => CPU,
   CPU => CPU (0.1);
END;
```

Example 2. This example of a communication network is taken from REIS79. The model has four customer classes with no class switching. Four of the eight service centers model half-duplex links; the other four model message sources. The mean service times are given in the table below. The topology of the network is shown in the figure on the next page.

	CLASS1	CLASS2	CLASS3	CLASS4
SOURCE1	2.0	-	-	-
SOURCE2	-	2.0	-	-
SOURCE3	-	-	3.0	-
SOURCE4	-	-	-	5.0
LINK1	2.0	-	4.0	1.0
LINK2	2.0	0.5	4.0	-
LINK3	2.0	0.5	-	-
LINK4	-	0.5	-	1.0

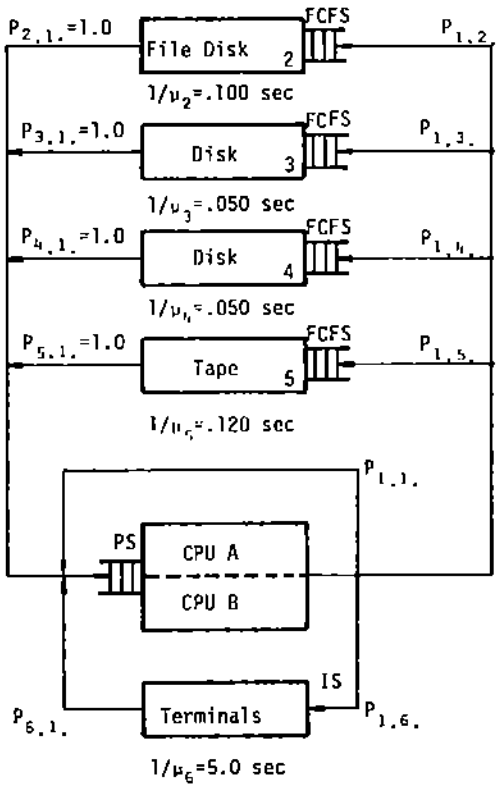


The following PMVA input describes this model:

```
NETWORK;
CLASSES
   CLASS1, CLASS2, CLASS3, CLASS4;
SERVERS
   SOURCE1  FCFS  LI  CLASS1=2.0,
   SOURCE2  FCFS  LI  CLASS2=2.0,
   SOURCE3  FCFS  LI  CLASS3=3.0,
   SOURCE4  FCFS  LI  CLASS4=5.0,
   LINK1    PS    LI  CLASS1=2.0  CLASS3=4.0  CLASS4=1.0,
   LINK2    PS    LI  CLASS1=2.0  CLASS2=0.5  CLASS3=4.0,
   LINK3    PS    LI  CLASS1=2.0  CLASS2=0.5,
   LINK4    PS    LI  CLASS2=0.5  CLASS4=1.0;
ROUTING
   SOURCE1/CLASS1 => LINK1 => LINK2 => LINK3 => SOURCE1,
   SOURCE2/CLASS2 => LINK4 => LINK3 => LINK2 => SOURCE2,
   SOURCE3/CLASS3 => LINK2 => LINK1 => SOURCE3,
   SOURCE4/CLASS4 => LINK4 => LINK1 => SOURCE4;
END;
```

Example 3. This example of a multi-class model with class switching is taken from BALB77. In this terminal driven system, each distinct type of job behavior is modeled by a different customer class. After passing through a short initial phase in which files and programs are copied from a permanent file disk to faster disks, a job either enters a long I/O bound phase in which it accesses the tape channel and the two fast disks or it enters a CPU bound phase in which it accesses only the fast disks. Jobs then return to the terminals and emerge as new jobs after a 5 second delay.

The routing of jobs and the mean service times are shown in the following figure and table.



Class 1 (Permanent Files)	Class 2 (Tape Access)	Class 3 (Compute-bound)
$1/\mu_{11}=.005$ sec.	$1/\mu_{12}=.020$ sec.	$1/\mu_{13}=.500$ sec.
$P_{1111}=.4500$	$P_{1232}=.2495$	$P_{1333}=.4950$
$P_{1121}=.2250$	$P_{1242}=.2500$	$P_{1343}=.4950$
$P_{1131}=.2250$	$P_{1252}=.5000$	$P_{1361}=.0100$
$P_{1141}=.0500$	$P_{1261}=.0005$	
$P_{1112}=.0500$		
$P_{1113}=.0500$		

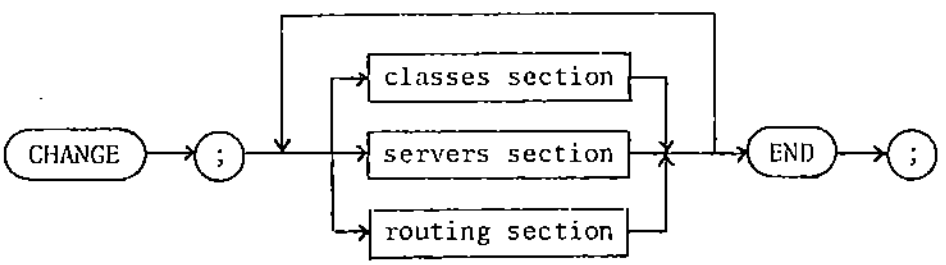
The following PMVA input describes this model:

```
NETWORK;
CLASSES
   INIT, IOBOUND, CPUBOUND;
SERVERS
   TERMINALS   IS     LI     ALL=5.0,
   CPU         PS     MS 2   INIT=0.005  IOBOUND=0.02  CPUBOUND=0.5,
   FILEDISK    FCFS   LI     ALL=0.1,
   DISK1       FCFS   LI     ALL=0.05,
   DISK2       FCFS   LI     ALL=0.05,
   TAPE        FCFS   LI     ALL=0.12;
ROUTING
   CPU/INIT => FILEDISK (0.45) DISK1 (0.225) DISK2 (0.225) => CPU,
   CPU/INIT => CPU/CPUBOUND (0.05) CPU/IOBOUND (0.05),
   CPU/IOBOUND => DISK1 (0.2495) DISK2 (0.25) TAPE (0.5) => CPU,
   CPU/IOBOUND => TERMINALS/INIT (0.0005),
   CPU/CPUBOUND => DISK1 (0.495) DISK2 (0.495) => CPU,
   CPU/CPUBOUND => TERMINALS/INIT (0.01),
   TERMINALS/INIT => CPU;
END;
```

3. MODIFYING A QUEUEING NETWORK MODEL

The CHANGE command is used to modify one or more parameters of a queueing network which was specified using the NETWORK command. When analyzing several similar queueing networks, a new network can be defined as a set of changes to the previous network. In an interactive environment, the CHANGE command can also be used to correct mistakes made when entering a network description.

Syntax. The syntax of the CHANGE command is:



The syntax of each section is the same as was specified for the NETWORK command.

When using the CHANGE command, only those parameters to be changed must be specified. All other parameters will keep their previous values. After the changed network is validated, the network description is presented in tabular form.

Limitations. In the current implementation, the names of customer classes and service centers cannot be changed. Also, the service type of a service center (LI, MS, or LD) cannot be changed. If class switching was specified by grouping class names in the CLASSES section, a new class cannot be added to an existing chain.

Deleting classes and service centers. A customer class can be effectively deleted either by specifying zero customers when analyzing the network or by setting the service time of that class at every service center to zero. A service center can be effectively deleted by setting the service times for all classes at that center to zero.

Adding classes and service centers. New customer classes and service centers can be added to a network using the CHANGE command. Remember that all routing probabilities previously specified will remain the same unless explicitly changed.

Examples. In the following examples we will modify the networks defined in section 2.6. To change the service time of the disk in Example 1, we could use the command:

```
CHANGE; SERVERS DISK 0.32; END;
```

The following command could be used to modify the routing of customers in Example 3 so that jobs in their I/O bound phase access the file disk:

```
CHANGE;
ROUTING
   CPU/IOBOUND => DISK2 (0.15) FILEDISK (0.10) => CPU;
END;
```

Note that the mean service time of I/O bound jobs at the file disk would be 0.1 since the keyword ALL was used in the definition of the device.

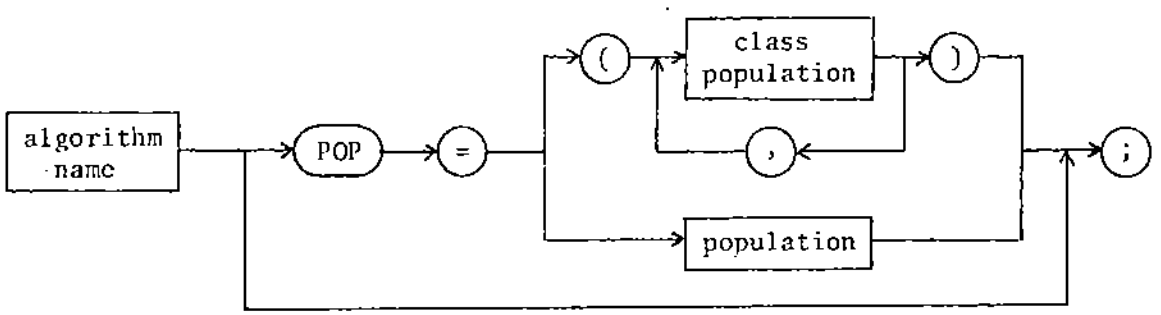
4. ANALYZING A QUEUEING NETWORK MODEL

Once a queueing network has been defined, one or more algorithms may be used to compute its performance characteristics. Although the solution algorithms differ greatly, the commands which invoke them have a common syntax and similar function.

4.1 Common Features

Each algorithm invocation command passes the internal representation of a queueing network to one of the solution routines. The routine computes a set of performance measures which is then passed to a common output routine.

Syntax. The syntax of the algorithm invocation commands is:



The valid algorithm names are MVA, ASYMP, SCHWEITZER, and LIN.

Customer population. The population of each customer class must be specified in the order that the classes were defined in the CLASSES section. If there is a single class, the population need not be enclosed in parentheses. If the customer population is omitted, the population from the previous algorithm invocation command will be used.

Output. Each invocation of an algorithm causes the following performance measures to be listed for each service center: throughput, mean waiting time, mean queue length, and utilization. For multiclass networks, both the global performance measures and the performance measures for each class are included. The global queue length distribution for each load dependent service center is also listed.

4.2 The MVA Command

The MVA command invokes the exact mean value analysis algorithm [REIS80]. The time required to compute the performance measures for a network depends on both the number of service centers and the customer population. The computing time increases as the product of the number of customers in each class.

Storage limitations. There is a limit on the total population (the sum of the populations of each class) only if there is a load dependent service center in the network. Otherwise, there is a limit only on the product of the populations of all classes excluding the class with the largest population. If either of these limits is exceeded a diagnostic message will be issued. The default values of the limits can be found in Appendix B.

4.3 The ASYMP and SCHWEITZER Commands

The ASYMP command invokes Bard's algorithm for large customer populations [BARD79]; the SCHWEITZER command invokes Schweitzer's algorithm [SCHW79]. Both algorithms are iterative versions of mean value analysis which require considerably less storage but produce only approximate results. These algorithms are particularly useful for analyzing models whose customer population would exceed the storage limits of the MVA command.

Storage limitations. There is no limitation on the total number of customers or the number of customers in any class.

Convergence of algorithms. A diagnostic message is printed if either algorithm fails to converge within a fixed number of iterations. The convergence criterion and the maximum number of iterations allowed are given in Appendix B.

Restrictions on networks which can be analyzed. Neither algorithm supports load dependent or multi-server service centers.

4.4 The LIN Command

The LIN command invokes the linearizer algorithm of Chandy and Neuse [CHAN80]. Linearizer is a mean value analysis based algorithm which gives high accuracy for any customer population. This algorithm should usually give better results than the algorithms used in the ASYMP and SCHWEITZER commands. For a single class network with very few customers, it may be more efficient to compute the exact solution using the MVA command than to

approximate the solution using the LIN command.

Storage limitations. There is no limitation on the total number of customers or the number of customers in any class.

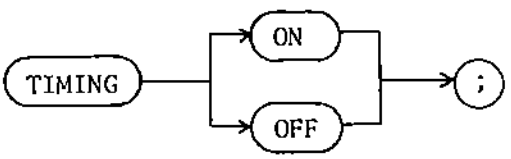
Restrictions on networks which can be analyzed. As currently implemented, this algorithm cannot be used to analyze networks with class switching or networks having load dependent or multi-server service centers.

5. MISCELLANEOUS COMMANDS

5.1 The TIMING Command

The TIMING command enables and disables the reporting of the CPU time required to execute each command.

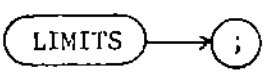
Syntax. The syntax of the TIMING command is:



5.2 The LIMITS Command

The LIMITS command lists the current values of all compile-time program limits. Appendix B contains the default values of these limits.

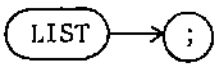
Syntax. The syntax of the LIMITS command is:



5.3 The LIST Command

The LIST command produces a tabular description of the current network.

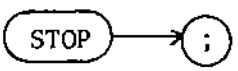
Syntax. The syntax of the LIST command is:



5.4 The STOP Command

The STOP command terminates execution of the program. This command should always be the last command in the input stream.

Syntax. The syntax of the STOP command is:



Bibliography

BALB77 Balbo, G., S. C. Bruell, and H. D. Schwetman, "Customer Classes and Closed Network Models - A Solution Technique," Information Processing 77, Proc. IFIP Congress 1977, B. Gilchrist, ed., North-Holland (1977), pp. 559-564.

BARD79 Bard, Y., "Some Extensions to Multiclass Queueing Network Analysis," Proceedings of the 4th International Symposium on Modelling and Performance Evaluation of Computer Systems, IFIP WG 7.3, North-Holland Publishing Company, Amsterdam, The Netherlands, February 1979.

BUZE73 Buzen, J., "Computational Algorithms for Closed Queueing Networks with Exponential Servers," Communications of the ACM, Vol. 16, No. 9, September 1973, pp. 527-531.

CHAN80 Chandy, K. and D. Neuse, "Fast Accurate Heuristic Algorithms for Queueing Network Models of Computing Systems," Technical Report TR-157, Department of Computer Science, University of Texas at Austin, Austin, TX, September 1980.

REIS79 Reiser, M., "A Queueing Network Analysis of Computer Communication Networks with Window Flow Control," IEEE Transactions on Communications, Vol. COM-27, No. 8, August 1979, pp. 1199-1209.

REIS80 Reiser, M. and S. S. Lavenberg, "Mean Value Analysis of Closed Multichain Queueing Networks," Journal of the ACM, Vol. 27, No. 2, April 1980, pp. 313-322.

SCHW79 Schweitzer, P., "Approximate Analysis of Multiclass Closed Networks of Queues," International Conference on Stochastic Control and Optimization, Amsterdam, The Netherlands, 1979.

Appendix A -- Keywords and Special Characters

Keywords

ALL	LCFS	NETWORKDESCRIPTION
ASYMP	LD	OFF
ASYMPTOTICANALYSIS	LI	ON
CHANGE	LIMITS	POP
CHANGENETWORK	LIN	POPULATION
CLASSES	LINEARIZER	PS
DEF	LIST	ROUTING
END	MEANVALUEANALYSIS	SCHWEITZER
FCFS	MS	SERVERS
IS	MVA	STOP
	NETWORK	TIMING

Special Characters

Special Character	Command/ Section	Use
;	all	Terminates all commands and network description sections
,	CLASSES	Separates class names
	SERVERS	Separates load dependent serv times
	SERVERS	Separates service center descriptions
	ROUTING	Separates routing descriptors
	invocation	Separates class populations
( )	CLASSES	Groups classes into chains
	SERVERS	Encloses load dependent service times
	ROUTING	Encloses routing probabilities
	invocation	Encloses class populations
[ ]	SERVERS	Encloses visit ratios
=	SERVERS	Associates class names with visit ratios and service times
=>	ROUTING	Indicates movement of customers
/	ROUTING	Separates service center name from customer class name
"	all	Delimits comments

Appendix B -- Compile-time Program Limits

Symbolic Name	Default Value	Description
MAXCLASSES	4	Maximum number of customer classes
MAXCHAINS	4	Maximum number of chains
MAXCENTERS	8	Maximum number of service centers
MAXSTAGES	32	Value of MAXCENTERS * MAXCLASSES
MAXLI	8	Maximum number of LI centers
MAXMS	3	Maximum number of MS centers
MAXLD	1	Maximum number of LD centers
MAXSRV	3	Maximum number of servers at any MS service center
MAXERRORS	10	Maximum number of errors reported per input line
MAXLISTSIZE	10	Maximum number of service center names between arrows in routing descriptor
LINEPS	1.0E-5	Singularity test used in solving visit ratio equations
MAXPOP	10	Maximum total number of customers if LD service centers in network (MVA only)
MAXSTATES	400	Maximum product of populations of all classes excluding the class with the largest population (MVA only)
EPSILON	1.0E-4	Maximum relative change allowed in global queue lengths for convergence (ASYMP and SCHWEITZER only)
MAXITER	100	Maximum number of iterations allowed (ASYMP and SCHWEITZER only)

Appendix C -- Environment Switches

Two Boolean constants in the program tailor the input/output routines to batch or interactive usage. These constants should be changed as needed and the program recompiled.

The constant ECHO determines whether characters read from the input file will be echoed to the output file. Although echoing of input characters is useful in a batch environment, it is provided automatically by most terminals or the component of the operating system which drives the terminals.

The constant PROMPT determines whether a prompt character will be written to the output file before a line of input is read. The default prompt character is a "greater than" sign (>). PROMPT should be assigned a value of TRUE if the software which drives the terminal does not provide a prompt character for input.

Constant	Batch Setting	Interactive Setting
ECHO	TRUE	FALSE
PROMPT	FALSE	TRUE or FALSE

Appendix D -- Diagnostic Messages

Number of customer classes must be <= nnn
First 10 characters of class name must be unique
Number of service centers must be <= nnn
Number of LI centers must be <= nnn
Number of MS centers must be <= nnn
Number of LD centers must be <= nnn
Infinite server must be load independent
Number of identical servers must be <= nnn
All serv times must be the same at a FCFS center
Branching probs must be in the range 0.0 to 1.0
Branching probabilities not previously specified
Parenthesized branching probability expected
"=>" expected
Service center name not previously defined
Customer class name not previously defined
Closing parenthesis expected
Unsigned number expected
Symbolic name expected
Unrecognized network description section keyword
Valid network description not available
Unrecognized command
Maximum customers at LD center is nnn
Number of center names between arrows must be <= nnn
Closing bracket expected
Load dependency factors must be same for all classes
Number of populations must be equal to number of classes
Unrecognized keyword

Routing matrix is singular
Routing probabilities for class xxx from service center yyy
    must sum to 1.0
Insufficient storage for queue length information
Total population exceeds maximum for networks with
    LD service centers
Approximation algorithms allow only LI service centers
Approximation failed to converge
Linearizer does not allow class switching or MS/LD centers