NAME

SPNetwork − QNET stochastic processing network input file format

DESCRIPTION

SPNetwork (Stochastic Processing Network) files are created by users and are intended to be given as the input for the quadAlp and alp programs. They define the topology and event rates of any possible queuing network.

FORMAT

SPNetwork files consist of assignment statements, comments, and whitespace. Assignments have the form

tag = *value*

where *tag* is the name of a network parameter and *value* is a number of the appropriate type (integer or floating point). The *tag* may be subscripted if it represents the name of an array of values. It is not possible to assign an entire array with a single statement; each array entry must be assigned separately. Whitespace is ignored in tag names, so the following pairs of lines are identical:

```
server pools = 3
mu ( 1, 1 ) = 0.25

serverpools=3
mu(1,1)=0.25
```

Comments begin with the # symbol and continue until the end of the line. For example

```
# service rates
mu(1,1) = 0.25
mu(1,2) = 0.0  # could be omitted since default mu values are 0
mu(2,1) = 0.2
mu(2,2) = 0.5
```

REQUIRED PARAMETERS

There are several required elements of an SPNetwork file. All subscripted parameters are indexed from 1. The required parameters are:

classes the number of classes in the network; must be a positive integer value.

servers (or **pools** or **server pools**)
the number of servers (server pools) in the network; must be a positive integer value.

lambda(*i*)
the arrival rate at class *i*. If supplied this must be an integer or floating point value; assumed to be zero if not specified. *i* must be a positive integer.

mu(*i*,*j*) the rate at which server *i* serves jobs in class *j*, If supplied this must be an integer or floating point value; assumed to be zero if not specified. *i* and *j* must be positive integers.

c(*i*) the holding cost at class *i*; must be an integer or floating point value. *i* must be a positive integer.

Routing in a stochastic processing network may either be deterministic or probabilistic. Every SPNetwork file must contain routing information in *one or the other* of the following formats; they may not be mixed.

s(*i*) the successor class of class *i*. Unspecified values are assumed to be zero, indicating an exit from the system. This must be a nonnegative integer value and *i* must be a positive integer.

p(*i*,*j*) the probability that upon completion a job in class *i* moves to class *j*. Unspecified probabilities are assumed to be zero. If *j* is zero then the assigned value is the probability of an exit from the system. This must be a nonnegative floating point value, *i* must be a positive integer, and *j* must be a nonnegative integer.

ADDITIONAL PARAMETERS

There are several other parameters that may be specified for a stochastic processing network configuration.

K(*i*) the number of servers in server pool *i*. This specifies many jobs server *i* work on simultaneously. This must be a positive integer value and *i* must be a positive integer value.

Other parameters may also be listed in an SPNetwork file but will be ignored by the **SPNetwork**(3) class. Programs can use the **TaggedValues**(3) class may used to access these parameters. For example, some QNet programs can use a truncated state space. The input files for these cases use the parameter **N(*i*)** to specify the truncation for class *i*.

EXAMPLE

The following describes a Rybko-Stolyar network with deterministic routing.

```
# Rybko-Stolyar Network

# Number of Classes and Servers
classes = 4
servers = 2

# Arrival rates into each class, unspecified values default to zero
# format is lambda(<class>) = <rate>
lambda(1) = 0.08
lambda(3) = 0.08

# Service rates of each server for each class
# unspecified values default to zero
# format is mu(<server>,<class>) = <rate>
mu(1,1) = 0.12
mu(2,2) = 0.12
mu(2,3) = 0.28
mu(1,4) = 0.28

# Holding costs per job per unit time at each class
# format is c(<class>) = <cost per job per unit time>
c(1) = 1.0
c(2) = 1.0
c(3) = 1.0
c(4) = 1.0

# Successor class of each class (deterministic routing)
# 0 to exit system, unspecified values default to zero
# format is s(<class>) = <successor class>
s(1) = 2
s(3) = 4
# jobs leaving classes 2 and 4 exit the system
```

SEE ALSO

SPNetwork(3), **TaggedValues**(3)

AUTHOR

Christopher Pfohl