NAME

Fmdn() − Virtual waiting time distribution function

IntFmdn() − Virtual waiting time distribution function for integral values of x

Mdn() − Calculates the state probabilities of a queue (queue length)

Fekdn() − Virtual waiting time distribution function

SYNOPSIS

#include <queuel.h>

double Fmdn(double x, double rho, int n);

double IntFmdn(int N, double rho, int n);

double *Mdn(double x, double rho, int n);

double Fekdn(double x, double rho, int k, int n);

DESCRIPTION

delim $$ These functions return the state probabilities or the virtual waiting time distribution of a $M/D/n$ queue. (And the virtual waiting time distribution of a ${E sub k}/D/n$ queue.)

Fmdn() is a model for the $M/D/n$ queuing system with Poisson arrivals and deterministic (constant) service time. Parameter *x* is the amount of unfinished work in the system. *Rho* is the load level of the system and *n* is the number of servers.

Fekdn() is a model for the ${E sub k}/D/n$ queuing system with Erlang-k arrivals and deterministic (constant) service time. Parameter *x* is the amount of unfinished work in the system. *Rho* is the load level of the system and *n* is the number of servers.

ALGORITHM

$M/D/n$ waiting time distribution is calculated using the following algorithm (Iversen):

P{$W <= t$}$ = sum from {i=0} to {n-1} sum from {j=0} to {i} P(j) sum from {{nu}=0} to {T} {{[A({nu}-t)] sup {{nu}n+{nu}-1-i}} over {[{nu}n+n-1-i]!}} e sup {A({nu}-t)},$

where $P(j)$ is a state probability.

For integral values of the waiting time we have

P{$W <= t$}$ = sum from {{nu}=0} to {n(t+1)-1} P({nu}).$

The state probabilities are calculated using the following procedure

− first we make an initial guess $(M/M/n)$:

${P sup {(1)}}(0)=({sum from {i=0} to {n-1} {({lambda h}) sup i} over {i!} + {({lambda h}) sup n} over {n!} {1} over {1 - {lambda h} over {n} } }) sup {-1}$

${P sup {(1)}}(i)=P(i-1) {lambda h} over i$, $i=1,2,...,n-1$

${P sup {(1)}}(i)=P(i-1) {lambda h} over n$, $i=n,n+1,...,I$ (until $P(i) < {epsilon}$)

${P sup {(1)}}(i)=P(i-1) {lambda h} over n$, $i=I+1,...,I+n$

−then we iterate until ${max sub {i<=I}} $|$P sup {(k)}(i)-P sup {(k-1)}(i)$|$<{epsilon}$:

$P sup {(k)}(i)=${$sum from {{nu}=0} to {n} P sup {(k-1)}({nu})$}$ P(i,h) + sum from {{nu}=n+1} to {n+i} P sup {(k-1)}({nu}) P(n+i-{nu},h)$, $i=0,1,...,I$

$P sup {(k)}(i)=P sup {(k)}(i-1) {{lambda h} over n}$, $i=I+1,...,I+n$

$S sup {(k)} = sum from {i=0} to {I+n} P sup {(k)} (i)$

$P sup {(k)}(i)={P sup {(k)}(i)} over {S sup {(k)}}$, $i=0,1,...,I+n$

$(P(i,h)={{({lambda h}) sup i} over {i!}} e sup {-({lambda h})})$

${E sub k}/D/n$ waiting time distribution is calculated using the $M/D/n$ algorithm:

${E sub k}/D/r$ (FIFO) is equivalent to $M/D/r*k$ (FIFO).

ERRORS

When $ rho $ is close to 1, these functions might give inaccurate results.

SEE ALSO

COST 224: Performance evaluation and design of multiservice networks