Evaluating the Performance of Shared Memory Parallel Computer System Using Recursive Models

O.E. Oguike[1] and D.U. Ebem[2]
M.N. Agu[3] and S.C.Echezona[4]

Department of Computer Science
University of Nigeria
Nsukka, Enugu State
Nigeria

e-mail:
[1] osondu.oguike@unn.edu.ng
[2]deborah.ebem@unn.edu.ng
[3]monica.agu@unn.edu.ng
[4]stevenson.echezona@unn.edu.ng

H.O.D. Longe[5]
O. Abass[6]

Department of Computer Sciences
University of Lagos
Lagos, Nigeria

e-mail:
[5]hodlonge@yahoo.co.uk
[6]olaabass@unilag.edu.ng

***Abstract*— Using a queuing approach to describe a shared memory parallel computer system, it can be considered as a parallel computer system with a shared single ready queue. This can depict a single queue parallel server queuing model. Some models that are based on probability density function have been used to model the performance of the queuing system. This paper uses recursive models to evaluate the performance of a single queue parallel server queuing model of compute intensive applications of a parallel computer system. The recursive models that this paper uses are efficient models because each recursive model makes one recursive call.**

Keywords-parallel computer queuing system; recursive models; compute intensive applicationr; performance metrics

I. INTRODUCTION

Consider a parallel computer system with a finite single ready queue [1,2,3,4]. The assumptions to be made is that arrival of processes into the finite single ready queue is according to poisson distribution and the service time is according to exponential distribution.[5,6].

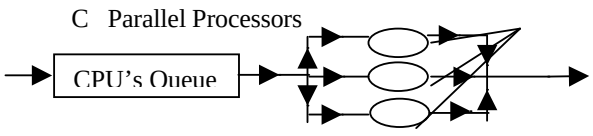


Figure 1 Model of a Single Queue, Parallel Processor Queuing System

The diagram in Figure 1, illustrates a parallel computer queuing system, having a finite single ready queue, and C number of processors.

II. STATEMENT OF THE PROBLEM

Jobs that arrive in the shared single ready queue of the parallel processors are split among the various processors for concurrent execution. Furthermore for compute intensive applications, jobs are arriving more than they are departing. It implies that the parallel processors are always busy. Therefore, there is the need to redefine the service rate of the various processors and use efficient recursive models to evaluate the various performance metrics of the system. The service/departure rate for a single queue parallel server queuing system has been modeled in [7] as

$$\mu_x = \begin{cases} x\mu, & 0 <= x < C \\ C\mu, & C <= x <= X \\ 0, & otherwise \end{cases} \quad (1)$$

X denotes the maximum number of processes that can be in the system. The above service/execution rate does not truly represent the service rate or execution rate of compute intensive applications of a parallel computer system, where all the processors are busy every time, executing a part of a program concurrently. There is the need to redefine the service rate of compute intensive applications of a parallel computer [8]. Some of the non-recursive models for compute intensive applications of parallel computer system, which have been developed in [8] are undefined when the arrival rate is equal to the service rate of the various parallel processors. There is therefore the need to use recursive models to develop thorough models that will solve the problem of undefined case as seen in the non-recursive

models whenever the arrival rate is equal to the service rate. Furthermore, it is important that recursive models be used to evaluate the performance of the system because recursive models are efficient models, especially when a recursive model makes only one recursive call [9], as seen in the recursive models used in this paper. This paper extends the models developed in [18] by using recursive models to model the standard error, which assesses the accuracy of the models developed in this paper.

III. METHODOLOGY

We aim at using recursive models to evaluate the performance of parallel computer system. We have achieved this aim by using computer queuing approach [10], with a finite single queue. The parallel processors depict parallel servers, and we have used all the related laws that are based on queuing system, like Little's formulae as stated in [7]. We have used statistical method of probability density function to develop the probability density function of the number of processes that join the queuing system. We have used mathematical method of deriving recursive models that determines the xth term and important functions of a given mathematical sequence to develop the recursive models. We have used the recursive models to develop the various performance metrics that evaluate the performance of parallel computer system. We have simulated the models, using Java programming language and we have used statistical regression/trend line analysis to analyze the results of the simulation [11].

IV. LITERATURE REVIEW AND LIMITATION OF CURRENT TECHNIQUE

Recursive models have been used extensively in the literature to model the performance of single finite queue parallel server queuing system [18, 20, 21]. However, the main contribution of this paper is that it redefines and represents an appropriate arrival rate of processes based on the way processes are split and assigned to the various parallel processors and for compute intensive application of parallel computer queuing system. Some of the benefits of using recursive models is that it is easy to simulate on the computer, and it is very efficient, especially when the recursive model contains one recursive call [19]. However, one of the limitations of using recursive models is that it is not very efficient when the recursive model contains more than one recursive call [19].

V. DEVELOPING THE MODELS

As a result of the use of the above methodologies, the following models have been developed.

A. Probability Density Function of the Number of Processes on the Parallel Queueing System.

Let X denotes the maximum number of processes that can be in the finite parallel processor queuing system at any time [12, 13, 14], and let C denotes the number of processors on the parallel computer. Suppose the arrival rate, λx when x processes are on the queuing system of the parallel processors can be described as follows:

$$\lambda_x = \begin{cases} \lambda, & x = 0,1,2,,3,...X-1 \\ 0, & otherwise \end{cases} \quad (2)$$

With good load balancing and homogeneous parallel processors, the departure/service rate, μ_x or rate for the next CPU burst time when x processes are on the queuing system of the parallel computer can be described as follows:

$$\mu_x = \begin{cases} c\mu, & x = 1,2,3,4,..., X \\ 0, & otherwise \end{cases} \quad (3)$$

We have used a constant model for the service time distribution, $c\mu$, rather than the usual distribution seen in most literature [7,15] as stated in equation (1). This is because of the way jobs are serviced/executed on a parallel computer system. Jobs that arrive at the common ready queue are split and assigned to the various processors for concurrent executions. The aim is to keep all the processors busy, we assume that under the steady state, the service rate/execution rate of a parallel computer will be the same, irrespective of the number of jobs in the common ready queue. Using the steady state probability as stated in [7,16] the probability that x processes will be on the queueing system is

$$P_x = \begin{cases} \rho^x P_0, & x \le X \\ 0, & otherwise \end{cases} \quad (4)$$

P_0 can be obtained as we sum all the probabilities and equate it to 1. This implies that:

$$\sum_{x=0}^{X} P_x = 1. \quad (5)$$

From equations (4) and (5), it implies that:

$$P_0 + \rho P_0 + \rho^2 P_0 + \rho^3 P_0 + \rho^4 P_0 + \ldots + \rho^X P_0 = 1. \quad (6)$$

Factorizing equation (6), it implies that

$$P_0 (1 + \rho + \rho^2 + \rho^3 + \ldots + \rho^X) = 1. \quad (7)$$

Using Mathematical method of deriving recursive model for the xth term of the sequence, ρ^0 , ρ^1, ρ^2 , ρ^3 ,…, ρ^X , the xth term of the sequence is given as

$$\text{Term1}(X, \rho) = \begin{cases} 1, & X = 0 \\ \rho * \text{Term1}(X-1, \rho), & X \neq 0 \end{cases} \quad (8)$$

Therefore, the sum of the first xth term of the sequence is given as

$$\text{Sum1}(X, \rho) = \begin{cases} 1, & X = 0 \\ \text{Term1}(X, \rho) + \text{Sum1}(X-1, \rho), & X \neq 0 \end{cases} \quad (9)$$

Therefore, using equation (9) in equation (7), it implies that

$$\text{Po}*\text{Sum1}(X, \rho) = 1 \quad (10)$$

Solving for Po in equation (10), we have

$$\text{Po} = \frac{1}{Sum1(X, \rho)} \quad (11)$$

Using equation (11) in equation (4), we obtain the probability density function of the number of processes that arrive at the parallel queuing system as

$$P_x = \begin{cases} \dfrac{\rho^x}{Sum\ 1(X, \rho)}, & x \leq X \\ 0, & otherwise \end{cases} \quad (12)$$

Equation (12) is the probability density function that models the probability that x processes will be in the parallel queuing system.

B. Average Number of Processes in the Parallel Queuing System.

Furthermore, the average number of processes in the queuing system can be described statistically as expectation of x , where x is the random variable that denotes the number of processes in the system. This can be written as

$$E(x) = \sum_{x=0}^{X} xP_x . \quad (13)$$

Using equation (12) in equation (13), we obtain

$$E(x) = \sum_{x=0}^{X} x\frac{\rho^x}{Sum1(X, \rho)} \quad (14)$$

Simplifying equation (14), we obtain

$$E(x) = \frac{1}{Sum1(X, \rho)} \sum_{x=0}^{X} x\rho^x \quad (15)$$

Expanding equation (15), we obtain

$$E(x) = \frac{1}{Sum1(X, \rho)} \left(\rho + 2\rho^2 + 3\rho^3 + 4\rho^4 + \ldots + X\rho^X\right) \quad (16)$$

Factorizing equation (16), we obtain

$$E(x) = \frac{1}{Sum1(X, \rho)} \rho\left(1 + 2\rho + 3\rho^2 + 4\rho^3 + \ldots + X\rho^{X-1}\right) \quad (17)$$

A recursive model can be used to determine the convergence of this series,

$$1 + 2\rho + 3\rho^2 + 4\rho^3 + \ldots + X\rho^{X-1}$$

as seen in equation (17). In order to determine the recursive model, we first determine the xth term of the sequence, 1, $2\rho^1$, $3\rho^2$, $4\rho^3$,…, $X\rho^{X-1}$, which can be determined by considering the sequence as two sequences. The first sequence is: 1, 2, 3, 4, …., X , while the second sequence is: ρ^0 , ρ^1, ρ^2 , ρ^3 ,…, ρ^{X-1} . Therefore the recursive model for the xth term of the first sequence is given as follows:

$$\text{Term2}(X) = \begin{cases} 1, & X = 1 \\ 1 + \text{term2}(X-1), & X \neq 1 \end{cases} \quad (18)$$

The recursive model for the xth term of the second sequence has been developed earlier in equation (8). Therefore, combining the two models in equations (8) and (18), the series, $1 + 2\rho + 3\rho^2 + 4\rho^3 + \ldots + X\rho^{X-1}$ converges to this recursive model,

$$\text{Sum2}(X, \rho) = \begin{cases} 1, & X = 1 \\ \text{Term2}(X)*\text{term1}(X-1, \rho) + \text{Sum2}(X-1, \rho), & X \neq 1 \end{cases} \quad (19)$$

Therefore, using equation (19) in equation (17), we obtain

$$E(x) = \frac{\rho}{Sum1(X,\rho)} \text{Sum2}(X, \rho) \quad (20)$$

$E(x)$, which is the average number of processes that is in the queuing system can be denoted as Ls. It implies that

$$Ls = \frac{\rho}{Sum\ 1(X,\rho)} \text{Sum2}(X, \rho) \quad (21)$$

C. *Effective Arrival Rate, λ_{eff}*

The effective arrival rate, λ_{eff} models the rate of arrival of processes that actually joins the single ready queue. The reason for this is because the single ready queue of parallel processor queuing system is finite; this means that there is a limit to the number of processes that the single ready queue can admit. Therefore, a process that arrives to join the queue can actually join the queue or be lost on arrival. Assuming that λ is the rate of arrival of processes into the single ready queue, and λ_{lost} is the rate of arrival of processes that will not be able to join the queue on arrival, because the finite single ready queue is full. It implies that:

$$\lambda_{eff} = \lambda - \lambda_{lost} \quad (22)$$

Since the queuing system allows a maximum of X processes, the probability that a process will not be able to join the queuing system on arrival will be the probability that X processes will be on the queuing system on arrival, P_x Therefore, λ_{lost}, the rate of arrival of processes that will not be able to join the common ready queue will be λP_X . It implies that

$$\lambda_{lost} = \lambda P_X \quad (23)$$

Using equation (23) in equation (22), we obtain

$$\lambda_{eff} = \lambda - \lambda P_X \quad (24)$$

Factorizing equation (24), we obtain

$$\lambda_{eff} = \lambda(1-P_X) \quad (25)$$

D. *Average Waiting Time on the Queuing System.*

Using Little's formulae as stated in [7], the average length of the queuing system is directly proportional to the average waiting time on the queuing system. This can be expressed as follows,

$$L_s \alpha W_s \quad (26)$$

L_s denotes the average length of the queuing system and W_s denotes the average waiting time on the queuing system. Applying the constant of proportionality in equation (26), we obtain

$$L_s = \lambda_{eff} W_s, \quad (27)$$

where λ_{eff} is the constant of proportionality.

Solving for W_s in equation (27), we obtain

$$W_s = \frac{L_s}{\lambda_{eff}} \quad (28)$$

Using equation (21) in equation (28), we obtain

$$W_s = \frac{\rho}{\lambda_{eff} * Sum1(X,\rho)} * \text{Sum2}(X, \rho) \quad (29)$$

E. *Average Waiting Time on the Ready Queue.*

The average waiting time on the queuing system, which will be denoted as Ws has been defined in [17], as average time that a process waits on the single queue together with the average service time. Because the distribution of the service time is exponential with parameter, μ, therefore, it implies that

$$W_s = W_q + 1/\mu \quad (30)$$

Solving for W_q in equation (30) and using equation (29) in equation (30), we obtain

$$W_q = \frac{\rho}{\lambda_{eff} * Sum1(X,\rho)} * \text{Sum2}(X, \rho) - \frac{1}{\mu} \quad (31)$$

F. *Average Number of Processes on the Queue.*

Using another version of Little's formulae as stated in [7], the average length of the queue is directly proportional to the average waiting time on the queue. This can be expressed as

$$L_q \alpha W_q \quad (32)$$

L_q denotes the average length of the queue and W_q denotes the average waiting time on the queue. Applying the constant of proportionality in equation (32), we obtain,

$$L_q = \lambda_{eff} W_q. \quad (33)$$

Using equation (31) in equation (33), we obtain,

$$L_q = \frac{\rho * Sum2(X,\rho)}{Sum1(X,\rho)} - \frac{\lambda_{eff}}{\mu} \quad (34)$$

Using equation (21) in equation (34), we obtain that

$$L_q = L_s - \frac{\lambda_{eff}}{\mu} \quad (35)$$

Simplifying equation (35), we obtain,

$$L_s - L_q = \frac{\lambda_{eff}}{\mu} = \bar{c} \quad (36)$$

$\bar{c}$ denotes the average number of busy processors. The percentage of utilization of the parallel processors is:

$$((L_s-L_q)/C)*100 = ((\frac{\lambda eff}{\mu})/C)*100 \quad (37)$$

VI. Estimating the Standard Error of the Models

Since all the models that have been developed are statistical models that are based on probability density function, it is necessary that we assess the accuracy of the models. One of the best ways of assessing the accuracy of probabilistic models is to estimate the standard error of the models. Statistically, the standard error of the random variable, X, which is number of processes in the queue of the parallel computer is defined as

$$SE(X) = \sqrt{VAR(X)} \quad (38)$$

VAR(X) is defined as

$$VAR(X) = E(X^2)-(E(X))^2 \quad (39)$$

$E(X^2)$ in equation (39) can be simplified as follows:

$$E(X^2) = \sum_{x=0}^{X} x^2 P_x \quad (40)$$

Using equation (12) in equation (40), we obtain the following:

$$E(X^2) = \sum_{x=0}^{X} x^2 \frac{\rho^x}{Sum1(X,\rho)} \quad (41)$$

Simplifying equation (41), we obtain,

$$E(X^2) = \frac{1}{Sum1(X,\rho)} \sum_{x=0}^{X} x^2 \rho^x \quad (42)$$

Simplifying equation (42) further, we obtain,

$$E(X^2) = \frac{1}{Sum1(X,\rho)} \left(\rho + 2^2\rho^2 + 3^2\rho^3 + 4^2\rho^4 + ... + X^2\rho^X\right) \quad (43)$$

Factorizing equation (43), we obtain

$$E(X^2) = \frac{1}{Sum1(X,\rho)} \rho\left(1 + 2^2\rho + 3^2\rho^2 + 4^2\rho^3 + ... + X^2\rho^{X-1}\right) \quad (44)$$

The series in equation (44) is made up of two sequences, which are:

$$\text{Sequence3} = \rho^0, \rho^1, \rho^2, \rho^3, ..., \rho^{X-1} \quad (45)$$

$$\text{Sequence4} = 1, 4, 9, 16, ..., X^2 \quad (46)$$

The recursive model that determines the xth term of the sequence in equation (46) has been stated in equation (17). Similarly, the recursive model that determines the xth term of the sequence in equation (46) is given as:

$$\text{Term3}(X) = \begin{cases} 1, & X = 1 \\ (2*X-1) + \text{Term3}(X-1), & X \neq 0 \end{cases} \quad (47)$$

Combining the two sequences, the recursive model for the xth term of this series in equation (44), which is shown below,

$$\left(1 + 2^2\rho + 3^2\rho^2 + 4^2\rho^3 + ... + X^2\rho^{X-1}\right)$$

is given as

Sum3(X, ρ), which is equivalent to:

$$\begin{cases} 1, & X = 1 \\ \text{Term1}(X-1, \rho)*\text{Term3}(X) + \text{Sum3}(X-1, \rho), & X \neq 0 \end{cases} \quad (48)$$

Therefore, using equation (48) in equation (44), we obtain,

$$E(X^2) = \frac{1}{Sum1(X,\rho)} \rho * \text{Sum3}(X, \rho) \quad (49)$$

Using equation (21) and equation (49) in equation (39), we obtain,

$$\text{Var}(X) = \frac{1}{Sum1(X,\rho)} \rho * \text{Sum3}(X, \rho) - (L_s)^2 \quad (50)$$

Therefore, using equation (50) in equation (38), we obtain,

$$SE(X) = \sqrt{Var(X)} \quad (51)$$

VII. PERFORMANCE ANALYSIS OF THE RECURSIVE MODELS

Performance analysis of the models aims at evaluating how parameter changes affect the performance metric that is under consideration. [10]. The performance metric that will be used to do the evaluation is the percentage of utilization of the parallel processors. The parameters that can be used to evaluate this performance metric under consideration are the maximum number of processes that can be in the system, X and λ, the arrival rate of processes into the single ready queue. As the value of X changes, we wish to find out how it affects the percentage of utilization of the parallel processors. Similarly, as the value of λ changes, we wish to find out how it affects the percentage of utilization of the parallel processors. After simulating the recursive models on the computer, using Java programming language, we obtain the various results of the performance metric, percentage of utilization of the parallel processors, for various values of X and specific value of λ and μ

After plotting the values of the percentage of utilization of the parallel processors against X as shown in figure 2a. The positive value of the slope in the trend line shows that as the number of processes that can be in the system increases, while other variables remain constant, the percentage of utilization of the processors increases. The interpretation of the result is that parallel computers realize high percentage of processors utilization for high memory capacity.

Similarly, after obtaining the various values of the performance metric, percentage of utilization of the parallel processors for fixed value of X = 25, C = 3, μ = 2 , and various values of λ. Figure 2b is used to illustrate the relationship between the performance metric, percentage of utilization of the parallel processors and λ. From figure 2b, the result shows that as the value of λ increases, the percentage of utilization of the processors increases while other variables, like X, C and μ remain constant. The interpretation of the result shows that as we use the parallel computer for compute intensive application, the percentage of utilization of the processors is bound to increase.

Furthermore, for fixed values of X = 25, which is the maximum number of processes to be admitted into the system, C = 3 parallel processors and arrival rate λ = 21, provided that ρ is not greater than 1, i.e. the system is used for non compute intensive applications, figure 2c shows the result of standard error against the departure rate. The result in figure 2c shows that the standard error of the models assumes a minimum value when fast processors are used on the parallel computer.

VIII. SUMMARY AND CONCLUSION

We have been able to evaluate the performance of parallel computer system, using recursive models, and assess the accuracy of the models by developing model that estimates the standard error. We have been able to simulate the models using Java programming language, and have been able to analyze the results of the simulation in order to establish how variable changes affect an important performance metric, percentage of utilization of the processors. Finally, we have been able to establish when the parallel computer assumes high performance and when the models assume high accuracy or low standard error.

ACKNOWLEDGMENT

We thank the Academic Board and the Postgraduate Board of the Department of Computer Science, University of Nigeria, Nsukka for giving us the opportunity to lecture these courses, Queuing Theory, Operating Systems, Algorithms and Structured Programming. The lectures that we delivered to both undergraduate and postgraduate students of university contributed to the development of this paper. We also thank the Postgraduate Board of the Department of Computer Sciences, University of Lagos, Nigeria for giving us the opportunity to carry out research in the area of parallel computing.

REFERENCES

[1] Henry H. Liu and Pat V. Crain, An Analytic Model for Predicting the Performance of SOA-Based Enterprise Software Applications, Proc. International Conference of Computer Measurement Group, (2004).

[2] S. Balsamo et al, A Review of Queueing Network Models with Finite Capacity Queues for Software Architecture Performance Prediction, (2002).

[3] Catalina M. Liado et al, A Performance Model Web Service, Proc. International Conference of Computer Measurement Group, (2005).

[4] Rosselio, J et al, A Web Service for Solving Queueing Network Models Using PMIF. www.perfeng.com/paperndx.htm, (2005).

[5] Cathy H. Xia, Zhen Liu., Queueing systems with long-range dependent input process and subexponential service time. Proc. ACM SIGMETRICS international conference on Measurement and modeling of computer systems,(2003).

[6] Shanti Subramanyam, Performance Modelling of a J2EE Application to meet Service Level s, Agreement, Proc. International Conference of Computer Measurement Group, (2005)

[7] Hamdy A. T.,. Operation Research: An Introduction, Prentice-Hall of India, (1999).

[8] O.E.Oguike et al,; Modelling the Performance of Compute Intensive Applications of a Parallel Computer System; Proc of. IEEE 2nd International Conference on Computational Intelligence, Modelling and Simulation; (2010).

[9] Ivan Stojmenovic; Recursive Algorithms in Computer Science Courses : Fibonacci Numbers and Binomial Coefficients; IEEE Transactions on Education; Vol. 48, No. 3

[10] Arjan J.C. van Gemund; Performance Modelling of Parallel Systems: An Introduction.

[11] Justyna Berlinska, The Statistical models of parallel applications, Annales UMCS Informatica, (2005).

[12] Arranchenkov, K.E., Vilchersky, N.O., Shevlyakor, G.L Priority queueing with finite buffer size and randomized push-out; mechanism. Proc. of ACM SIGMETRICS international conference on measurement and modeling of computer systems.; (2003).

[13] Abunday, B.D., and Khorram, E. The finite source queueing model for multiprogrammed computer systems with different CPU times and different I/O times. Acta Cybern. 8, 4 , (1998)

[14] J. Sztrik; Finite-Source Queueing Systems and their Applications: A Biliography;

[15] Trivedi K. Shridharbhai, Probability and Statistics with Reliability, Queuing and Computer Science Applications, John Wiley & Sons Inc., (2002).

[16] Per Brinch Hansen. Operating System Principles. Prentice-Hall of India Private Limited, (1990).

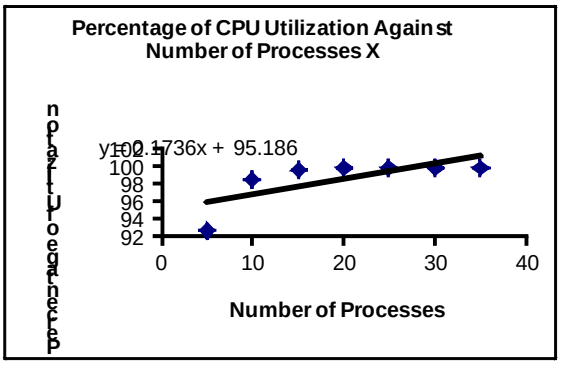
[17] Peterson, J. L, Silberschatz, A. Operating System Concepts, Addison-Wesley Publishing Company, (1985).

[18] Oguike, O.E et al.; Evaluating the Performance of Parallel Computer System Using Recursive Models; Proc of. IEEE 4th[nd] International Conference on Computer Modelling and Simulating; (2010).

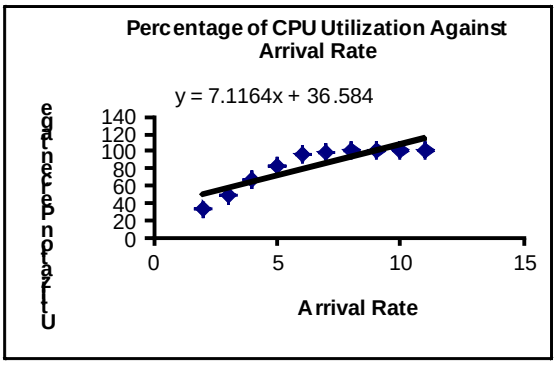
[19] Ivan Stojmenovic; Recursive Algorithms in Computer Science Courses : Fibonacci Numbers and Binomial Coefficients; IEEE Transactions on Education; Vol. 48, No. 3

[20] O.E. Oguike et al; Evaluating the Performance of Heterogeneous Distributed Memory Parallel Computer System Using Recursive; Proc. of Second International Conference on Intelligent Systems, Modelling and Simulation; (2011).

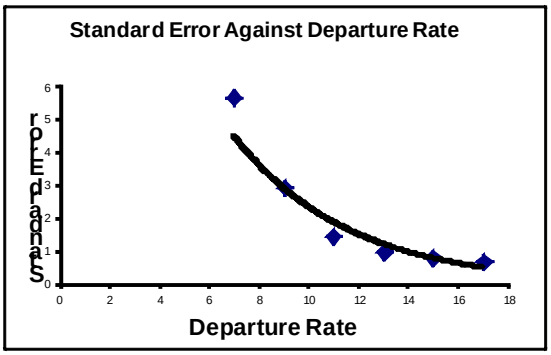
[21] J. Sztrik[a] and T. Gál A recursive solution of a queueing model for a multi-terminal system subject to breakdowns; Performance Evaluation Volume 11, Issue 1, Published by Elsevier, (1990).



a



b



c

Figure 2. Results of the Simulated Model