IUG Journal of Natural and Engineering Studies **Vol 23, No 1, 2015, pp 1-6, ISSN 2409-4587**
Peer-reviewed Journal of Islamic University-Gaza **IUGNES**

A Mathematical Model for Cluster Applications*

Mohamed I. Riffi[1,]**

[1]Faculty of Science, Islamic University of Gaza, Gaza Strip, State of Palestine

Received on (17-6-2014) Accepted on (23-9-2014)

Abstract
In this paper, we use probabilistic modeling and techniques to derive formulas in order to determine the number of computer nodes needed to execute applications on a cluster of computers so that application response time can be satisfied. Our probabilistic model is basically an *M/G/1/K* queueing system. In this model, we account for the workload conditions (in terms of the number of applications or jobs being received per unit time) as well as the processing power of each node. Finally, We use simulations to present a numerical example showing how our probabilistic model can be used.

Keywords Network performance, Rule-based servers, Queueing systems, Cluster computers, Probabilistic models.

نموذج رياضي للتطبيقات العنقودية

ملخص
في هذا البحث، نستخدم نمذجة وأساليب احتمالية لاشتقاق معادلات من أجل تحديد عدد عقد الحاسوب المطلوبة فتنفيذ تطبيقات على حاسبات عنقودية حتى يكون زمن الاستجابة للتطبيق مرضية. النموذج الاحتمالي هنا هو بشكل أساسي نظام طابور *M/G/1/K*. في هذا النموذج، نأخذ في الحسبان أحوال تحميل العمل (وفق عدد التطبيقات أو الوظائف التي تستقبل في كل وحدة زمن) وأيضا قوة المعالجة لكل عقد. وفي النهاية، نستخدم المحاكاة لعرض مثل عددي يظهر كيفية استخدام نموذجنا الاحتمالي.

كلمات مفتاحية: أداء الشبكة، الخوادم المبنية على القاعدة، أنظمة الطابور، الحاسبات العنقودية، النماذج الاحتمالية.

*This research was supported by a research grant from the Scientific Research Council of the Palestinian Ministry of Higher Education.
**Corresponding author e-mail address: riffim@gmail.com

1. Introduction:

Many business and scientific applications (such as pharmaceuticals, oil and gas, healthcare, financial services manufacturing, physics, chemistry, biology, computer science, etc.) require cluster computers to run. In a computer cluster, multiple computers or nodes are allocated to run one application [1, 2]. The application or job computation is divided and then distributed to run in parallel on multiple cluster (or worker) nodes or machines. After the computation is finished, a collector node collects the results of computation on all the cluster nodes.

One of the main challenges of the scheduler node is to determine the correct number of the cluster or worker nodes needed to finish the computation. This mainly depends on the response time needed to finish the application. Naturally, the more worker nodes allocated, the smaller the response time. However, the allocating more nodes than needed will result to extra cost and usage of cluster machines. So the idea is to allocate the correct minimal number of worker nodes while satisfying the response time specified by the user.

Therefore, the main objective of this paper is to devise a mathematical model that can be used to determine the minimal number of cluster nodes needed to satisfy a given response time. We use queueing theory to achieve this. We give numerical examples and show how this model can be used to determine the correct minimal number of cluster worker nodes to satisfy the required response time.

The rest of the paper is organized as follows. Section 2 presents our mathematical queuing model to capture the dynamism of executing cluster applications. Section 3 presents numerical examples to show how the mathematical model can be used in determining the response time. Finally, Section 4 concludes the study and identifies future work.

2. Background:

Our analytical model throughout the analysis in this paper assumes the packet arrivals are Poisson with fixed packet sizes, and the service times all have exponential distribution. For certain types of network traffic, assuming Poisson arrivals is adequate. In [4], it was concluded that modeling the voice traffic as Poisson with fixed-size packets gives adequate approximation, especially when the voice traffic is high. However, for some other traffic types, especially that of Ethernet, network packets are not fixed in size, and their arrivals do not always follow a Poisson process but rather bursty [5]. Also in reality, service times are not necessarily always exponential. An analytical solution becomes intractable and not a trivial task when considering variable-size packets and non-Poisson arrivals, and when also considering general service times. The impact of having bursty traffic and having general distribution for packet sizes and service times can be best modeled and studied using DES (Discrete Event Simulation) [6].

The finite queueing system with an N-stage service can be modeled as an M/G/1/K queueing system with incoming packets following a Poisson arrival λ and service times with general distribution. The service time is a random variable consisting of the sum of N independent distributed random variables. In other words, incoming packets are served sequentially in multiple stages. New packets enter Stage 1 only after the previous packet has departed the queueing system, i.e. left Stage N. The overall service time would thus be the sum of the random service times of the N stages. For rule-based servers, in practice, $r > \mu$, where μ denotes the mean service rate of the scheduler node and r denotes the mean service rate for the result collector node, since the service time of IP processing involving device driver handling and network processing is on average much larger than that of processing individual rules.

We follow the approach presented in [7] to analyze the M/G/1 and M/G/1/K queueing system, with a queue size of K inclusive of the one in service. The approach is based on computing first the steady-state probabilities at the departure instants of packets from the queue, to find subsequently the queue size distribution at an arbitrary time.

3. Mathematical Queueing Model:

In this section, we present a queueing model for executing cluster applications with the aim of deriving formulas for the mean response time. As shown in Figure 1, an arriving application gets first queued in a buffer of size *K*-1 and then gets serviced sequentially in three stages by: (1) the scheduler node, (2) the worker nodes which work in parallel, and (3) the result collector node. We assume independent exponentially distributed service times for all nodes. We also assume a Poisson arrival for incoming application.

Let λ denote the workload which is the mean arrival rate of the cluster application, μ denote the mean service rate of the scheduler node, β denote the mean service rate of a worker node, r denote the mean service rate for the result collector node, N denote the number of worker nodes, K denote the capacity of the queueing system inclusive of the one in service, and γ denote the throughput or the departure rate.

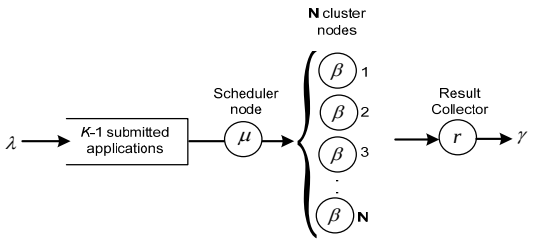

Figure 1 *Queueing model for servicing cluster applications*

The queueing system shown in Figure 1 is basically an *M/G/1/K* queueing system, with a Poisson arrival λ, generally distributed service times, and a system capacity *K*. The challenge in analyzing such a queueing system is to compute $b(x)$ which is the PDF of the generally distributed random variable *X* representing the service times, and subsequently deriving a closed-form solution to α_k which is the probability of having k application arrivals during this service time. α_k is expressed in terms of $b(x)$ as

$$\alpha_k = \int_0^\infty \frac{(\lambda x)^k}{k!} e^{-\lambda x} b(x) dx \qquad (1)$$

Effectively, $b(x)$ is the PDF of a random variable X which is the sum of three independent random variables representing the service times at the three service stages; namely, the scheduler node, the worker nodes, and the collector node. The first stage random service time has an exponential distribution with a mean $1/\mu$ and a PDF $f(t) = \mu e^{-\mu t}$. The third stage random service time has also exponential distribution with a mean $1/r$ and a PDF $g(t) = re^{-rt}$.

For the second stage, we assume the service times for each worker node are independent and exponentially distributed with a mean $1/\beta$. Then, the second stage random service time B for these N parallel workers can be expressed as $B = \max\{B_1, B_2, ..., B_N\}$. In a way, the second stage service time depends on the *slowest* of these N parallel workers, whereby the result collector will start executing after the slowest worker node has finished. Therefore, the CDF of the random variable *B* can be expressed as

$$F_B(t) := P(B \le t) = \prod_{i=1}^{N} P(B_i \le t) = (1 - e^{-\beta t})^N \qquad (2)$$

The PDF of B can be obtained by differentiating the above equation with respect to *t* as

$$f_B(t) = N\beta(1 - e^{-\beta t})^{N-1} e^{-\beta t}. \qquad (3)$$

Then, $b(x)$ is the convolution of the three PDFs: $f(t)$, $f_B(t)$, and $g(t)$. Since the convolution is a linear operator, we compute first the convolution of $f(t)$ and $g(t)$, and

then compute the convolution of the resulting function $h(t)$ with $f_B(t)$.

To derive the formula for the mean service time $E[B]$ of the second stage, we use Equation (3) to express $E[B]$ as

$$E[B] = \int_0^\infty t f_B(t)dt = \int_0^\infty N\beta t(1-e^{-\beta t})^{N-1}e^{-\beta t}dt$$

To evaluate this integral, we let $y = 1-e^{-\beta t}$. Then $dy = \beta t e^{-\beta t}dt$ and $t = -\log(1-y)/\beta$. Therefore, the integrals becomes of the form

$$E[B] = -\frac{N}{\beta}\int_0^1 \log(1-y)y^{N-1}dy,$$

which evaluates to $\frac{1}{\beta}H(N)$, where

$H(N) = \sum_{i=1}^{N}\frac{1}{i}$ is the N[th] harmonic number.

Therefore,

$$E[B] = \frac{1}{\beta}\sum_{i=1}^{N}\frac{1}{i}. \quad (4)$$

In [3], it has been shown that the convolution of two density functions of two random service times with means of $1/r$ and $1/\mu$, can be expressed as

$$h(t) = \begin{cases} \frac{r\mu}{\mu-r}\left(e^{-rt}-e^{-\mu t}\right) & \mu \neq r \\ \mu^2 t e^{-\mu t} & \mu = r \end{cases} \quad (5)$$

Let us consider first the case where $\mu \neq r$. $b(x)$ is the convolution of $h(t)$ and $f_B(t)$ which can be expressed as

$$b(x) = \frac{N\eta\mu\beta}{\mu-r}\left(\int_0^x (1-e^{-\beta t})^{N-1}e^{-\beta t}e^{-r(x-t)}dt - \int_0^x (1-e^{-\beta t})^{N-1}e^{-\beta t}e^{-\mu(x-t)}dt\right) = \frac{N\eta\mu\beta}{\mu-r}(I_1-I_2), \quad (6)$$

where

$$I_1 = \int_0^x (1-e^{-\beta t})^{N-1}e^{-\beta t}e^{-r(x-t)}dt, \text{ and}$$

$$I_2 = \int_0^x (1-e^{-\beta t})^{N-1}e^{-\beta t}e^{-\mu(x-t)}dt$$

Integrating by parts both I_1 and I_2, we get

$$I_1 = e^{Nx\beta}\sum_{i=1}^{N}\frac{C_{i,N}e^{-(i-N)x\beta}}{(r-i\beta)} - \frac{(N-1)!e^{-rx}\beta^{N-1}}{\prod_{i=1}^{N}(r-i\beta)}, \text{and}$$

$$I_2 = e^{Nx\beta}\sum_{i=1}^{N}\frac{C_{i,N}e^{-(i-N)x\beta}}{(\mu-i\beta)} - \frac{(N-1)!e^{-\mu x}\beta^{N-1}}{\prod_{i=1}^{N}(\mu-i\beta)},$$

where $C_{i,N} = (-1)^{i+1}\binom{n-1}{i-1}$.

From Equation (1), $\alpha_k = \frac{\lambda^k}{k!}\int_0^\infty x^k e^{-\lambda x}b(x)dx$,

and by substitution, we get

$$\frac{\mu-r}{N\eta\mu\beta}\alpha_k = \frac{\lambda^k}{k!}\sum_{i=1}^{N}\frac{C_{i,N}}{(r-i\beta)}\int_0^\infty x^k e^{-x\lambda}e^{-Nx\beta}e^{-(i-N)x\beta}dx - \frac{(N-1)!\beta^{N-1}}{\prod_{i=1}^{N}(r-i\beta)}\frac{\lambda^k}{k!}\int_0^\infty x^k e^{-x\lambda}e^{-rx}dx$$
$$-\frac{\lambda^k}{k!}\sum_{i=1}^{N}\frac{C_{i,N}}{(\mu-i\beta)}\int_0^\infty x^k e^{-x\lambda}e^{Nx\beta}e^{-(i-N)x\beta}dx + \frac{(N-1)!\beta^{N-1}}{\prod_{i=1}^{N}(\mu-i\beta)}\frac{\lambda^k}{k!}\int_0^\infty x^k e^{-x\lambda}e^{-\mu x}dx$$

Integrating by parts, we get

$$\int_0^\infty x^k e^{-x\lambda}e^{-Nx\beta}e^{-(i-N)x\beta}dx = k!(\lambda+i\beta)^{-k-1}$$

$$\int_0^\infty x^k e^{-x\lambda}e^{-rx}dx = \frac{k!(\lambda+r)^{-k}}{\lambda+r} = k!(\lambda+r)^{-k-1}$$

$$\int_0^\infty x^k e^{-x\lambda}e^{-\mu x}dx = \frac{k!(\lambda+\mu)^{-k}}{\lambda+\mu} = k!(\lambda+\mu)^{-k-1}$$

Therefore, we have

$$\frac{\mu - r}{Nr\,\mu\beta}\alpha_k = \sum_{i=1}^{N}\frac{C_{i,N}\lambda^k(\lambda + i\beta)^{-k-1}}{(r - i\beta)} - \sum_{i=1}^{N}\frac{(N-1)!\,\lambda^k\beta^{N-1}(\lambda + r)^{-k-1}}{\prod_{i=1}^{N}(r - i\beta)} - \sum_{i=1}^{N}\frac{C_{i,N}\lambda^k(\lambda + i\beta)^{-k-1}}{(\mu - i\beta)} + \sum_{i=1}^{N}\frac{(N-1)!\,\beta^{N-1}\lambda^k\beta^{N-1}(\lambda + \mu)^{-k-1}}{\prod_{i=1}^{N}(\mu - i\beta)}.$$

Or, equivalently,

$$\begin{aligned}\alpha_k = &\sum_{i=1}^{N}\frac{nr\mu\beta C_{i,N}\lambda^k(\lambda+i\beta)^{-k-1}}{(\mu-r)(r-i\beta)} - \sum_{i=1}^{N}\frac{nr\mu(N-1)!\lambda^k\beta^N(\lambda+r)^{-k-1}}{(\mu-r)\prod_{i=1}^{N}(r-i\beta)} \\ &- \sum_{i=1}^{N}\frac{nr\mu\beta C_{i,N}\lambda^k(\lambda+i\beta)^{-k-1}}{(\mu-r)(\mu-i\beta)} + \sum_{i=1}^{N}\frac{nr\mu(N-1)!\beta^{N-1}\lambda^k\beta^N(\lambda+\mu)^{-k-1}}{(\mu-r)\prod_{i=1}^{N}(\mu-i\beta)}\end{aligned} \quad (7)$$

Let us consider now the case where $\mu = r$. Then $b(x)$ can be expressed as

$$\begin{aligned}b(x) &= \mu^2\int_0^x N\beta(1-e^{-\beta t})^{N-1}e^{-\beta t}(x-t)e^{-\mu(x-t)}dt \\ &= I_1 - I_2,\end{aligned} \quad (8)$$

where

$$I_1 = N\beta\mu^2 xe^{-\mu x}\int_0^x (1-e^{-\beta t})^{N-1}e^{-(\beta-\mu)t}dt, \text{ and}$$

$$I_2 = N\beta\mu^2 e^{-\mu x}\int_0^x (1-e^{-\beta t})^{N-1}te^{-(\beta-\mu)t}dt.$$

Integrating by parts, we get

$$I_1 = N\beta\mu^2 xe^{-\mu x}\sum_{i=1}^{N}C_{N,i}(1-e^{-x(i\beta-\mu)})/(i\beta-\mu), \text{ and}$$

$$I_2 = N\beta\mu^2 e^{-\mu x}\sum_{i=1}^{N}C_{N,i}\left(1-(1+ix\beta-x\mu)e^{-x(i\beta-\mu)}\right)/(i\beta-\mu)^2.$$

From Equation (1), we get

$$\alpha_k = N\beta\mu^2\frac{\lambda^k}{k!}\sum_{i=1}^{N}C_{N,i}\int_0^\infty xx^k e^{-\mu x}e^{-\lambda x}(1-e^{-x(i\beta-\mu)})/(i\beta-\mu)dx - N\beta\mu^2\frac{\lambda^k}{k!}\sum_{i=1}^{N}C_{N,i}\int_0^\infty x^k e^{-\mu x}e^{-\lambda x}(1-(1+ix\beta-x\mu)e^{-x(i\beta-\mu)})/(i\beta-\mu)^2dx$$

Integrating by parts, we get

$$\alpha_k = N\beta\mu^2\sum_{i=1}^{N}\frac{(k+2)C_{N,i}\lambda^k[(\lambda+\mu)^{-k-2}-(i\beta+\lambda)^{-k-2}]}{(i\beta-\mu)} - N\beta\mu^2\frac{\lambda^k}{k!}\sum_{i=1}^{N}\frac{C_{N,i}\lambda^k[(\lambda+\mu)^{-k-1}+(i\beta+\lambda)^{-k-2}((k+1)\mu-\lambda-i(k+2)\beta)]}{(i\beta-\mu)^2} \quad (9)$$

Following the approach in [3, 7], we now use α_k to compute the steady-state probabilities at the departure instants of applications from the queue. The mean time spent in the system by an application entering the queue can be expressed as

$$W = \frac{\overline{K}}{\gamma} = \frac{1}{\lambda}\sum_{k=1}^{K-1}k\pi_k + \frac{K}{\lambda}(\pi_0 + \rho - 1), \quad (10)$$

where $\rho = \lambda\overline{X}$ and $\overline{X} = 1/\mu + E[B] + 1/r$, and $\{\pi_{k;}\ 0 \le k \le K-1\}$ are the steady state probabilities at the departure instants which can be derived using the approach presented in [3,7].

4. Numerical Results:

In this section, we illustrate how our model can be used in determining the number of worker nodes needed to satisfy a given response time. For our numerical examples, we choose K=100, $1/r = 20$ ms, and $1/\beta = 250/N$ ms.

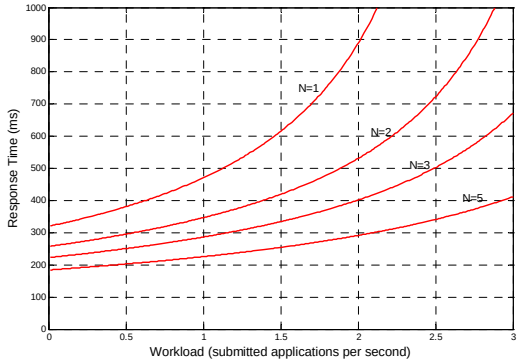

Figure 2 *Average response time curves in terms of number of worker nodes*

Figure 2 plots the average response time for different N in relation to the average workload λ. As shown in the figure, as the workload increases the response time increases with a small number of worker nodes. For example, to satisfy an average response time of 300 ms,

at a light workload ($\lambda < 1.5$), we require N to be smaller than 5. However at high workload ($\lambda > 2.5$), N has to be greater than 5.

Conclusion:

We have presented a mathematical queueing model based on queueing theory to determine the number of cluster nodes needed for executing a cluster application given the workload conditions and processing power of each cluster node. We showed a numerical example of how these formulas can be used to determine the number of cluster nodes needed to satisfy the mean response time for a cluster application.

References:

[1] Bader, D., and Pennington, R. *Cluster Computing: Applications* (1996, June). Retrieved July 13, 2007, from Georgia Tech College of Computing.

[2] William, W., Hargrove, and Forrest, M. H. *Cluster Computing: Linux Taken to the Extreme* (1999). Retrieved October 18, 2011, from Linux magazine.

[3] Salah, K. Analysis of a Two-Stage Network Server. *International Journal of Applied Mathematics and Computation*, Elsevier Science, ***217(23)***, (2011) 9634-9645.

[4] Karam, M., and Tobagi, F. Analysis of Delay and Delay Jitter of Voice Traffic in the Internet. *Computer Networks Magazine*, ***40(6)***, (2002) 711-726.

[5] Leland, W., Taqqu, M., Willinger, W., and Wilson, D. On the Self-Similar Nature of Ethernet Traffic. *IEEE/ACM Transaction on Networking*, ***2(1)***, (1994) 1-15.

[6] Jain, R. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Jonh Wiley & Sons, Inc., New York, USA (1991).

[7] Takagi, H. *Queueing Analysis*: *Finite Systems*. Vol. ***1***, North-Holland, Amsterdam, The Netherlands (1993).