PAPER: Influence of deterministic customers in time sharing scheduler

Patricia Pascal        Thierry Monteil
LAAS - CNRS
7 avenue du Colonel Roche 31077 Toulouse Cedex 4 France
{ppascal,monteil}@laas.fr

Abstract

This paper presents an approach to study the influence of deterministic customers in queueing systems. This is used to make real time decisions to manage resources. This method is applied to the scheduler of multiprocessor machines with round robin model. The transient probabilities which take care of the deterministic events are given. They are used to express a relation for the variation of the expectation of the number of processes. This relation is simplified with a general method called fluid approximation and used to predict the execution time of the deterministic events.

Keywords: queueing system, Markov chain, scheduler, round robin.

1 Introduction

Many phenomena are studied with stochastic models in telecommunications [1] and in telecommunication networks (networks structures and packet switching) [2] and in computers systems, operating systems) [3] [4]. The goal of those studies is to extract information from the system in stationary state and sometimes in transitory state. Another aspect is the study of the systems with finite queues [5] or after a specific event (failure of a server). In this paper the special event studied is the arrival of a perfectly well known work (deterministic) in the stochastic system. The goal is to evaluate the response time for this special event. The system studied in this paper is the scheduler of multiprocessor machines with time sharing policies (round robin multiservers queueing model) and the incoming of well known processes (regular parallel applications). The multiprocessor workstation model is first described, then the problem with its motivation is explained, in a third part the Markov chain integrating the deterministic process is introduced. To finish approximations and validation are presented.

2 Multiprocessor workstation model

To study the influence of deterministic process in stochastic system on multiprocessor machines analysis of the load evolution is necessary and so the model of scheduling on multiprocessor machines must be found. In [6] the comparison of different multiprocessor schedulers (Solaris 2.6 and IRIX 6.5) has been done. All algorithms are time sharing: a time slice is given to a process during a quantum and the process at the head of the queue will access to the first free processor during a quantum. At the end of the quantum the process will leave the system with the probability σ or will come back to the queue with the probability 1 − σ. x is defined as the number of processes in the system. Figure 1 displays the model chosen for multiprocessor machines.
The probability $g_i$ for a process to take the processor $i$ times and the average $E(T)$ are:

$$g_i = \sigma (i-1) (1 - \sigma), \quad E(T) = \sum_{i=1}^{\infty} (iQ)g_i = \frac{Q}{1 - \sigma} = \frac{1}{\mu}$$

3 Presentation of the problem and motivation

The goal of this study was to realize an application mapping on multiprocessor stations with resources utilization optimization. Supposed the processor time required by the machine for this application is a deterministic one whereas the other applications already running on the machines are unknown and constitute the stochastic load of the machines. The execution time of deterministic processes has to be predicted. Machines behaviour after the insertion of the deterministic processes was observed in order to know if there is an influence of the deterministic processes on the load. The number of processes in the system with and without creation of deterministic processes has been compared. Deterministic processes and processes already on the machines are scheduled with the round robin policy with a quantum $\mu$ like any process running on the multiprocessor workstation model (round robin model without priorities and multiservers). Stochastic processes are created with a Poisson rate, they use the processors during an exponential time. At a fixed date the deterministic processes are inserted. A list of “jobs” constitutes the load of the machine. Processes are virtually created: no “fork()” is done but a process creation consists on adding a new process in the list. The deterministic processes are marked. An average of the number of processes in the system $X(t)$ from several simulations with different initialisations of the stochastic load generator is depicted on figure (2) . Two cases are presented in the table 1 : the incoming rate $\lambda$, the service rate $\mu$, the number of deterministic processes $d$, the number of servers $C$, the utilization rate $\rho$. The deterministic processes are inserted at time $t=500$, they require 500 units of computing time.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Case 1</th>
<th>Case 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda$</td>
<td>0.006</td>
<td>0.02</td>
</tr>
<tr>
<td>$\mu$</td>
<td>0.03333</td>
<td>0.03333</td>
</tr>
<tr>
<td>$\rho$</td>
<td>0.09</td>
<td>0.3</td>
</tr>
<tr>
<td>$d$</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$C$</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
In the first case, as the utilization rate of the machine is small the influence of deterministic processes on the stochastic ones is not very important. But, in the second case the influence is more important. This phenomenon increases with the utilization rate and it can be concluded that it can not be neglected. In the next part, the markov chain chain of the model integrated the deterministic processes is presented.

4 The Markov chain

d is the number of deterministic inserted processes and C the number of processors. The time required by the deterministic processes is known. The state of the system at time t is given by the number of processes in the system. A deterministic process requires \( t_0 \) units of computing time. The state of the system at time t is given by the number of processes n means that there are \( x = n \) processes in the system. The time quantum \( Q \) is the number of processes that can be served simultaneously in the system at time t + Q only depends on the state at time t and on the probability \( \lambda \) of having a departure or an arrival. So the system is a Markovian one.

There can not be less processes than d in the system after the arrival of the deterministic processes until they leave. So, states below d in the Markov chain can not be reachable. The probability to have a new process during a quantum Q is equal to \( \lambda Q \). If there are less processes than processors then all processes are running on a processor. If there are more processes than processors then \( l \) processes are running on a processor and \( (n - C) \) processes are waiting. The process at the head of the queue will be executed by the first server which will become free.

Let consider a state n with \( n < C \) then all processes have a server and no process can leave the system. So, the probability that one of the n processes (excepted the deterministic ones) leaves the system is equal to \( (n - d) * \mu Q \). Let now consider a state n with \( n > C \) then only the stochastic processes are running on a processor and \( (n - C) \) processes are waiting. As there are \( n - d \) stochastic processes in the whole, the probability to have a stochastic process leaving the system is equal to \( (n - d) * \mu Q \) and so the probability to reach the state n-1 is equal to \( C * ((n - d)/n) * \mu Q \). On figure 3 the case \( d < C \) is depicted and on figure 4 the case \( d \geq C \) is depicted.
The Markov chain represents the evolution of the number of processes in the system after the insertion of the deterministic processes. The transient probabilities could be calculated...
Let $P_i(t)$ be the probability to have $i$ processes in the system at time $t$ and $P_i(t)$. Equation (2) expresses the case $d < C$ and equation (3) the case $d \geq C$.

$$
\begin{align*}
\text{for } d < i < C: & \quad P_i'(t) = \frac{\partial P_i(t)}{\partial t} = \lambda P_{i-1}(t) + (i + 1 - d)\mu_i \\
& \quad - \left(\lambda + (i - d)\mu\right)P_i(t) \\
\text{for } i = d: & \quad P_d'(t) = \frac{\partial P_d(t)}{\partial t} = \mu P_{d+1}(t) - \lambda P_d(t) \\
\text{for } i \geq C: & \quad P_i'(t) = \frac{\partial P_i(t)}{\partial t} = \lambda P_{i-1}(t) + C\mu \frac{(i + 1 - d)}{i + 1} \\
& \quad - \left(\lambda + C\mu \frac{i - d}{i}\right)P_i(t)
\end{align*}
$$

$$
\begin{align*}
\text{for } d < i: & \quad P_i'(t) = \frac{\partial P_i(t)}{\partial t} = \lambda P_{i-1}(t) + C\mu \frac{(i + 1 - d)}{i + 1} \\
& \quad - \left(\lambda + C\mu \frac{i - d}{i}\right)P_i(t) \\
\text{for } i = d: & \quad P_d'(t) = \frac{\partial P_d(t)}{\partial t} = \frac{C}{d + 1} \mu P_{d+1}(t) - \lambda P_d(t)
\end{align*}
$$

A fictitious stationnary state is defined as an infinite state where the deterministic processes are still there. It is like a system in which the deterministic processes never leave their execution time that must be predicted the behaviour of the machine is only interessant when the deterministic processes are still in the system. So the number of deterministic processes must be considered in the markov chain stability condition.

The markov chain is irreducible because every state can be reached from every other state. So the limiting probabilities always exist and are independent of the initial distribution. The states are recurrent non-null states if after an infinite time the state is not an infinite one. The condition is that the incoming rate is lower than the departure rate which could be written as:

$$
f(i) = \frac{\lambda}{C\mu \left(\frac{i + 1 - d}{i + 1}\right)} < 1
$$
f(i) is defined for states \( i \geq d \), it decreases and the maximum is for \( i = d \) 
\[ \frac{\lambda(d + 1)}{C \mu} \]
So the condition is satisfied if this relation (5) is correct:
\[ \frac{\lambda(d + 1)}{C \mu} < 1 \]

### 4.1 Expectation of the Number of Processes

\( X(t) \) is defined as the expected number of processes in the system and \( X^\infty \) as the expectation number of processes at the stationary case. \( X^\infty \) can be calculated from the Markov chain theory \( [7] [8] \) will be used. It consists in finding an equation with \( X(t) \) and its derivative.

When \( d < C \) there is:
\[
X^\infty = \left( \sum_{i=d}^{C-1} \left( \frac{\lambda}{\mu} \right)^{(i-d)} \right) * \frac{i}{(i-d)!} + \sum_{i=C}^{\infty} \left( \frac{i}{C \mu} \right)^{(i-C)} * \frac{i!}{C!(i-d)!} * \left( \frac{\lambda}{\mu} \right)^{(C-d)}
\]

\[
P^\infty_d = \frac{\sum_{i=d}^{C-1} \left( \frac{\lambda}{\mu} \right)^{(i-d)} \frac{1}{(i-d)!} + \sum_{i=C}^{\infty} \left( \frac{i}{C \mu} \right)^{(i-C)} \frac{i!}{C!(i-d)!} * \left( \frac{\lambda}{\mu} \right)^{(C-d)}}{\sum_{i=d}^{\infty} \frac{\left( \frac{\lambda}{\mu} \right)^{(i-d)} i!}{d!(i-d)!}}
\]

When \( d \geq C \) there is:
\[
X^\infty = \sum_{i=d}^{\infty} \left( \frac{i}{C \mu} \right)^{(i-C)} * \frac{i!}{d!(i-d)!} \cdot P_d^\infty
\]

\[
P^\infty_d = \frac{\sum_{i=d}^{\infty} \left( \frac{i}{C \mu} \right)^{(i-d)} \frac{i!}{d!(i-d)!}}{\sum_{i=d}^{\infty} \frac{\left( \frac{i}{C \mu} \right)^{(i-d)} i!}{d!(i-d)!}}
\]

\( \dot{X}(t) \) is defined as the derivative of \( X(t) \), it is easy to show with equation (11) \( \dot{X}(t) \) obeys the fundamental following differential equation:
\[
\dot{X}(t) = \frac{\partial X(t)}{\partial t} = \sum_{i=0}^{+\infty} i \dot{P}_i(t)
\]
\[
\dot{X}(t) = \lambda - C \mu (1 - P_d(t)) + \mu \sum_{i=d}^{C-1} i P_{d+i-C-1}(t) + C \mu d \sum_{i=C+1}^{\infty} (P_i(t))
\]

When \( d \geq C \) it can be shown with equations (3) that \( \dot{X}(t) \) obeys the fundamental following differential equation:
\[
\dot{X}(t) = \lambda - C \mu (1 - P_d(t)) + C \mu d \sum_{i=C+1}^{\infty} (P_i(t))
\]
Equations (2) (11) or (3) (12) give the evolution of the system for \( t > t_0 \) but (11) and (12) are difficult to manipulate due to the transient probabilities in the formulas. In paragraph 5.1, an approximation will be proposed.

### 4.2 Prediction of execution time

It is assumed that the process arrives at time \( t_0 \). Let consider a deterministic process requiring \( t_c \) units of computing time. The deterministic process will depart from the system at time \( t_r \). \( x(t) \) is defined as the number of processes (the deterministic and the stochastic ones) in the system at time \( t \). As the number of processes in the system will not stay constant after time \( t_0 \) because the proposed model of UNIX system uses a time sharing policy with \( C_0 \), an approximation can be given.

\[
t_c = \int_{t_0}^{t_r} \frac{C}{x(t)} dt
\]

In next part the approximations and the validation are presented.

### 5 Approximations and validation

#### 5.1 Equations approximations

Equations (11) and (12) can not be used because \( P_i(t) \) must be known for \( t \in [t_0, t_r] \) and for each \( i \). Therefore there are infinite sums. It is not compatible with a real time resolution and an approximation must be done. Let \( X_a(t) \) be the approximation of \( X(t) \). In [9] a model for mono-processors machines was found (M/M/1 model), the prediction of the expectation number of processes was

\[
\dot{X}(t) = \lambda - \mu [1 - P_0(t)]
\]

The term \( P_0(t) \) was approximated with a fluid approximation : in the M/M/1 model relation (16) is true :

\[
1 - P_0^\infty = \frac{X^\infty}{1 + X^\infty}
\]

The fluid approximation consists of the extension of the relation in stationary case (17):

\[
1 - P_0(t) = \frac{X(t)}{1 + X(t)}
\]

So, the approximation of the expectation number of processes was (18):

\[
\dot{X}_a(t) = \lambda - \mu \frac{X_a(t)}{1 + X_a(t)}
\]
The relation (19) expresses that the variation of processes in the system is the difference between arrival and departure of processes. Load processes leave the system but deterministic processes do not leave the system and so are considered as re-introduced in the system to be sure they have the processor time requested. Equation (19) is the approximation:

$$\frac{\partial X_a(t)}{\partial t} = \lambda - C\mu\alpha\frac{X_a(t)}{1 + X_a(t)} + C\mu d, \quad \alpha = \frac{(\lambda + C\mu d)(1 - e^{-\lambda t})}{C\mu X(t)}$$

Equation (14) can not be used directly, the number of processes in the system is not known. With equation (19), it is the expectation of the number of processes that is calculated because it does not fluctuate a lot for a computer, it globally remains close to its expectation. An approximation of equation (14) is done in equation (20):

$$t_c = \int_{t_0}^{t_r} \frac{C}{X(t)} dt$$

This equation can be generalized for the prediction of the execution time of several deterministic processes inserted at the same time on the machine. Let consider N processes computing time request : $t_c^1 \leq t_c^2 \ldots \leq t_c^{N-1} \leq t_c^N$. Each process will leave the system at time $t_c^i$. Then equation (21) can be found (with $t_r^0 = t_0$).

$$t_c^j = \sum_{i=1}^{j} \left( \int_{t_r^{i-1}}^{t_r^i} \frac{C}{X(t)} dt \right)$$

In the last part, the validation will be presented.

### 5.2 Validation

To validate equation (19) two tests were done, it is explained in sections 5.2.1 tests, the same different cases were considered (case 1 to 6 depicted in table 1 with loads from 0.12 to 0.6 (group 1, group 2 and group 3). Utilization rate higher than 0.7 is not considered because it has been shown in section 3 that below this value the influence of the deterministic processes is minimal. For each utilization rate tests have been done when $d < C$ and when $d > C$ (case 3 and 4, case 5 and 6).

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Group 1</th>
<th>Group 2</th>
<th>Group 3</th>
<th>Group 4</th>
<th>Group 5</th>
<th>Group 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda$</td>
<td>0.02</td>
<td>0.02</td>
<td>0.02</td>
<td>0.02</td>
<td>0.04</td>
<td></td>
</tr>
<tr>
<td>$\mu$</td>
<td>0.03333</td>
<td>0.03333</td>
<td>0.03333</td>
<td>0.03333</td>
<td>0.03333</td>
<td>0.03333</td>
</tr>
<tr>
<td>$\rho$</td>
<td>0.12</td>
<td>0.12</td>
<td>0.3</td>
<td>0.3</td>
<td>0.6</td>
<td></td>
</tr>
</tbody>
</table>
5.2.1 Comparison of the Expectation number of processes given by the probabilities with the approximation.

The expectation of the number of processes was calculated with the probabilities distribution truncated at state 100 (equation (22)) and compared with the expectation of the number of processes calculated with the approximation equation (19).

\[ X(t) = \sum_{i=0}^{100} iP_t(i) \]

At time t=0 the deterministic processes are already in the system. The results are depicted on figures (5, 6).

Figure 5: Expectation number of processes in the system

Figure 6: Expectation number of processes in the system

So, this approximation is a good one in most cases. There is only a small difference in heavy load and when many deterministic processes are inserted.
5.2.2 Comparison of the Expectation number of processes given by the simulation with the approximation.

It has been controled that the influence described in section 3 was well predicted. The evolution of the expected number of processes after the introduction of deterministic processes and the prediction were compared. The number of processes was calculated with many simulations (average of 300 samples) and a deterministic process was introduced. The expectation of the number of processes for the same incoming rate and the same treatment rate was approximated with equation (19).

The results are depicted on figures (7, 8, 9).

(a) Case 1 (b) Case 2

Figure 7: Expectation number of processes in the system

(a) Case 3 (b) Case 4

Figure 8: Expectation number of processes in the system

(a) Case 5 (b) Case 6

Figure 9: Expectation number of processes in the system

It can be observed that the approximation represents correctly the evolution of the number of processes; for a high utilization rate a small error can be observed.
5.2.3 Validation of the execution time prediction.

Simulations of multiprocessor machines with round robin policy with different loads have validated the execution time prediction. Several tests have been done for different loads. Two load tests have been done when \( d < C \) and when \( d \geq C \).

At a fixed time the deterministic processes are inserted. The measured execution times were compared. When there is only one deterministic process it requires \( t_c \) of computing time. When three processes have been inserted the shortest process requires \( t^1_c = 1.5 \cdot t_c \) and the longest process requires \( t^3_c \).

The tests done with the average error rate for each process \( (E^1, E^2, \ldots) \) and the standard deviation error rate for each process \( (D^1, D^2, \ldots) \) are summed up on the table and depicted on figure (10) and (11).

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
<th>Case 4</th>
<th>Case 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \lambda )</td>
<td>0.006</td>
<td>0.006</td>
<td>0.02</td>
<td>0.02</td>
<td>0.02</td>
</tr>
<tr>
<td>( \mu )</td>
<td>0.03333</td>
<td>0.03333</td>
<td>0.03333</td>
<td>0.03333</td>
<td>0.03333</td>
</tr>
<tr>
<td>( \rho )</td>
<td>0.09</td>
<td>0.09</td>
<td>0.36</td>
<td>0.36</td>
<td>0.36</td>
</tr>
<tr>
<td>( d )</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>( C )</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>( t^1_c )</td>
<td>20 to 520</td>
<td>20 to 520</td>
<td>20 to 520</td>
<td>20 to 520</td>
<td>20 to 520</td>
</tr>
<tr>
<td>( t^2_c )</td>
<td>–</td>
<td>30 to 780</td>
<td>–</td>
<td>30 to 780</td>
<td>–</td>
</tr>
<tr>
<td>( t^3_c )</td>
<td>–</td>
<td>45 to 1170</td>
<td>–</td>
<td>45 to 1170</td>
<td>–</td>
</tr>
<tr>
<td>( E^1 )</td>
<td>1.6%</td>
<td>3%</td>
<td>6.1%</td>
<td>7.6%</td>
<td>16%</td>
</tr>
<tr>
<td>( E^2 )</td>
<td>–</td>
<td>1.6%</td>
<td>–</td>
<td>1.6%</td>
<td>–</td>
</tr>
<tr>
<td>( E^3 )</td>
<td>–</td>
<td>1.1%</td>
<td>–</td>
<td>2.5%</td>
<td>–</td>
</tr>
<tr>
<td>( D^1 )</td>
<td>2.9%</td>
<td>1%</td>
<td>2.3%</td>
<td>2.2%</td>
<td>6.7%</td>
</tr>
<tr>
<td>( D^2 )</td>
<td>–</td>
<td>2.1%</td>
<td>–</td>
<td>1.9%</td>
<td>–</td>
</tr>
<tr>
<td>( D^3 )</td>
<td>–</td>
<td>1.6%</td>
<td>–</td>
<td>1.4%</td>
<td>–</td>
</tr>
</tbody>
</table>

Table 3: Cases studied to validate the execution time prediction.

Figure 10: Approximation of execution time: case 1
when there are machines with small and medium load, when machines have a high load the mapping has less influence on the parallel application performance.

6 Conclusion

The influence of deterministic load on a stochastic one has been studied. The introduction of deterministic processes in multiprocessor machines has been considered. The phenomenon has been studied with a Markov chain integrating the deterministic load. The expectation of the number of processes and the expected execution time has been calculated. The novelty of this approach consists in calculating the transient probabilities influenced by the deterministic event; it allows to study stationary and transient state of the system. The method has been applied to networks with other queuing models. This method will be used to model high-speed networks (Myrinet, Giga Ethernet) and to deduce resources management policies. The clusters and parameters of the model (incoming rate, service time) are observed. If they depend on time, another algorithm is used to predict them with instantaneous and past average values knowledge [9]. This approach can be generalized for different queuing systems. The transient probabilities influenced by the deterministic event must be found. The variation of the expectation of the number of customers in the system (different models) can be approximated by using stationary properties [fluid approximation].

References


