Performance Evaluation of Operating Systems Using Approximate Analytical Methods

Gunter Bolch and Stefan Greiner
Institute for Mathematical Machines and Data Processing IV
University Erlangen - Nürnberg
e-mail: bolch@informatik.uni-erlangen.de

November 20, 1995

Abstract

The motivation for the following paper were three models for a variant of a UNIX based multiprocessor operating system that was developed at the Institute for mathematical machines and dataprocessing IV [Jung 91]. These models were formerly analyzed by Markovian methods to get performance measures and the measurements of the real system were also available. This models had the following problems :

- Class switching is allowed.
  There are more job classes in the system available which can switch their class. The problem then is that the number of jobs in each class is not longer constant but depends on the time when you examine the system.
- Jobs have priorities with a mixed priority strategy.
  That means that some job classes can be preempted but others not.

To reduce the time to analyze the system a new approximate technique is developed. The technique is based on the well known 'Mean Value Analysis' (MVA). This new technique can be applied to open and closed queueing networks.

1 Description of the System

In this section three models of a UNIX-based multiprocessor operating systems [Jung91] are presented. For these models we developed a new technique to compute the performance measures as these models can not be analyzed with standard techniques.

1.1 System assumptions

- the network is closed and there are K jobs in the system.
- there are several job classes in the system and class switching is allowed.
- the service times are exponentially distributed.
- the peripheral device system consists of three magnetic tapes.
- the system has three job classes *user, kern, driv* with the priority order driv > kern > user.
- jobs in the context *user* can always be preempted by *driv* and *kern* jobs. Other preemptions are not possible.
- the system is in steady state.

- the standard parameters are

K = 10 No.Apu = 2	$p_{io} = 0.05$ $p_{done} = 0.01/0.005$ $p_{drivdone} = 0.4$	$s_{user} = 0.25...20.0$ $s_{kern} = 1.0$ $s_{driv} = 0.5$

- model of a UNIX-job [Jung91]

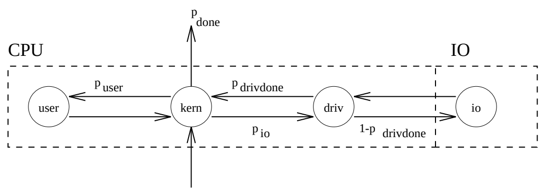

Figure 1: The monoprocessor model

1.2 The Monoprocessor-Model

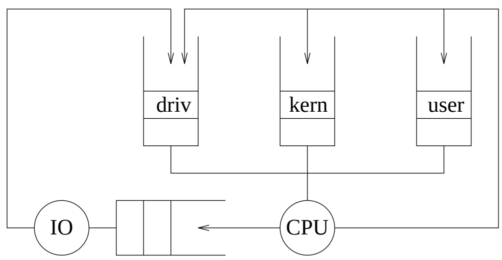

Figure 2: The monoprocessor model

The model shown in the figure above is called monoprocessor model. In an abstract way we have a closed queueing network with three job classes where class switching is allowed. At the CPU we have a mixture of a preemptive- and nonpreemptive service strategy. A condition for this system is that at any time at most one job can be active at the CPU.

1.3 The Master-Slave-Model

For this model, shown in the next picture, we have the same assumptions as in the monoprocessor-model but now we have two so called APU's. These are additional coprocessors that work together with the main CPU. In addition to the monoprocessor model it is assumed that only jobs in the context *user* can be processed on the APU. If APU and CPU are both free and a user job is in the queue then it is decided randomly on which processor (CPU or APU) the job is serviced.

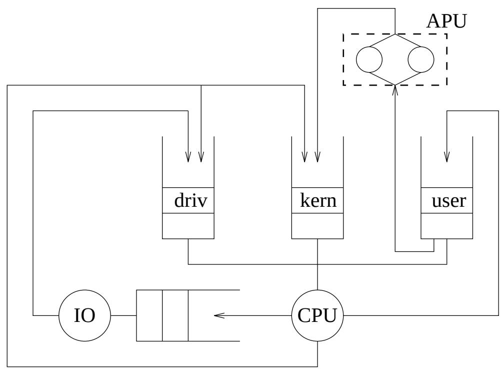

Figure 3: The master slave model

1.4 The Associated-Processor-Model

The assumptions are the same as in the master-slave-model but now kernel jobs are allowed to be processed at the APU. Because of the fact that kernel jobs have higher priority than user jobs, preemption is also possible at the APU.

1.5 Problems of the Models

The models of this operating systems have the following characteristics :

- class switching is allowed.
- there is a mixed priority strategy (preemptive and nonpreemptive job classes)

Because of these characteristics it is not possible to analyze the models with standard analysis techniques like mean value analysis.

2 Computation Time

Because of the fact that one can not use standard product-form analysis techniques, it seems to be necessary to use simulation or Markovian analysis to get performance measures.

2.1 Simulation

At first we tried to get performance measures by simulation. The problem was that even with such a simple queueing model it took about two days to get performance measures for the standard parameter set and $s_{user} = 0.25$. But this is too long if you want to analyze the system in many different scenarios or if you want to optimize the system.

2.2 Markovian Analysis

The next step was to analyze the models with Markovian techniques. For this we used the Markov analyzer **MOSES** [BGJZ94]. This is a tool where one can specify the system very easily. The tool then generates the underlying Markov chain and solves the set of equations. With this tool it was possible to reduce the computation time considerably compared with simulation. The computation time for the monoprocessor model with the same set of parameters as in the simulation model was about one minute and the computation time for the associated processor model was at about 25 minutes. This is short compared to simulation but still very long if one wants to analyze the system for a wide set of parameter values.

2.3 Approximate Analysis of the Models

Because of the long computation time of Markovian analysis and simulation, we developed a new approximate technique that allows a very fast evaluation of each of the three systems. The nice thing about this new technique is that it is used together with existing analysis techniques like mean value analysis but the still existing analysis technique need not be changed. We do not extend MVA to solve the system but we change the model of the system so that it can be solved by MVA.
The principle of this new technique is shown in the following picture.

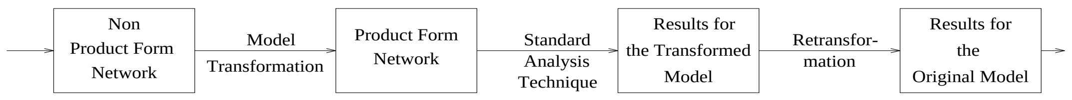

Figure 4: Principle of the analysis technique

In the rest of the paper we show how to extend a standard analysis technique to analyze the operating system models. As standard analysis technique, we choose the mean value analysis but our new technique is not only restricted to this analysis technique and can also be used for other analysis techniques like SCAT and convolution.

3 Development of the new Technique

3.1 Networks with Class Switching

The first problem that has to be solved is class switching. If class switching is allowed the number of jobs in a class is not constant. To solve this problem the concept of chains [Munt73] is used. A chain is a set of classes in which a job can not switch from one chain to another chain. Every class of a chain can be reached by every other class in that chain but not from classes of other chains. If all classes can be reached by every other class we have only one chain. If class switching is not allowed then the number of chains is the number of classes. Therefore the number of jobs in each chain is constant and the only thing one has to do is to transform the class parameters into chain parameters. Then you can analyze the system on chain measures and retransform the chain performance measures into class performance measures after the computation is done.

3.2 Networks with Class Switching and Mixed Priority Strategy

The key idea to solve the problem of the mixed priority strategy is the technique of the shadow server [Sevc77] which is originally applicable to networks with a pure preemptive priority strategy. The idea of this technique is to split the R-class server into R single-class servers (one service station per job-class).

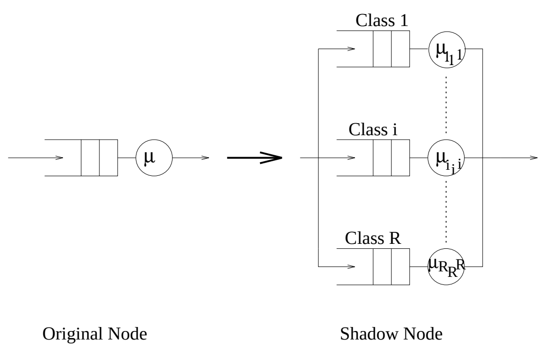

Figure 5: Principle of the shadow technique

To simulate the impact of preemption one has to increase iteratively the service times for jobs with lower priority. After the iteration has been finished the new service times are used to compute the performance measures with the mean value analysis. In [Kauf84] a modification of the shadow technique is introduced to get better results because the original shadow technique is not very exact. To deal with any mixed priority strategy the set ξ_r is defined as the set of jobs that can not be preempted by a class r job [BoGr94,Grei93]. Additionally it is assumed that $r \notin \xi_r$. With the help of Kaufman's extension of the shadow technique and the set ξ_r it is possible to formulate an algorithm for any mixed priority strategy with class switching [BoGr94,Grei93].

1. determine the chains in the network
2. transform the original model into the shadow model
3. set $\lambda_{i_j,r} = 0$
4. compute

$$\tilde{\rho}_{i_j,r} = \begin{cases} \lambda_{i_r,r} \cdot s_{i_r,r} & \textit{if } r = j \\ 0 & \textit{otherwise} \end{cases}$$

$$s_{i_j,r} = \begin{cases} \dfrac{s_{i,r}}{1 - \sum\limits_{s \in \xi_r} \frac{1}{\psi_{i_s,s}} \cdot \tilde{\rho}_{i_s,s}} & \textit{if } r = j \\ 0 & \textit{otherwise} \end{cases}$$

$$\psi_{i_j,r} = \begin{cases} \dfrac{\varrho(r)[1-\varrho(r)] + \beta(r) \cdot \varrho(r+1)}{\varrho(r)[1-\varrho(r)]^2 + \beta(r) \cdot \varrho_(i_r, r)} & if \quad r = j \\ 0 & otherwise \end{cases}$$

$$\beta(r) = \sum_{k \in \xi_r} \frac{\rho_{i_j,r}}{\omega_{r,k}}$$

$$\omega_{r,k} = \frac{s_{ik}}{s_{ir}}$$

$$\varrho(r) = \sum_{k \in \xi_r} \rho_{i_k,k}$$

   Here s_{ir} is the original service time of a a class r job at node i in the original network. For nodes with $s_{i_j,r} = 0$ the visit ratios must be zero. The rest of the visit ratios remain unchanged.

5. transform the class values into chain values (to differentiate between class values and chain values the chain values will be marked with *).
6. compute the performance measures of the transformed shadow model with the MVA.
7. retransform chain values into class values
   If the $\lambda^*_{i_j,r}$ (chain value) in two following steps differ less than ε stop the iteration. Otherwise go back to step 4.

3.3 Shadow Technique and Extended M/M/m Node

In the following we introduce a new type of node - the so called extended M/M/m-node. This new node type plays a major role in the UNIX multiprocessor models. The special feature of this node is that it has n job classes that are ordered in priority. The CPU can be entered by all job classes while the so called APU (associate processing unit) can only be entered by the job classes $v..n(1 \leq v \leq n)$. This means that the priority of the job depends on which service station it enters. This new node is shown in the following picture.

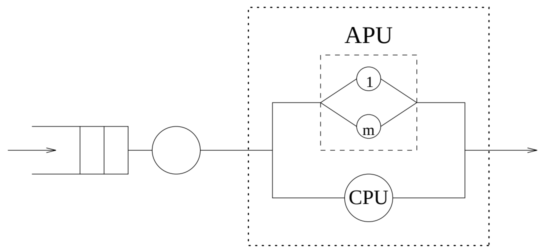

Figure 6: Extended M/M/m-node (original form)

Let ζ be the set which contains the highest priority of the job that can enter the service station (CPU, APU). In a general case if one has a so called *generalized extended M/M/m node* [Grei93] ζ contains normally more than two elements. In [BoGr94,Grei93] it is shown that in the shadow algorithm only the step for computing the mean service time $s_{i_j,r}$ has to be changed in the following way :

$$\forall \quad c \in \zeta \quad : \quad s_{i_j,r} = \begin{cases} \dfrac{s_{ir}}{1 - \sum\limits_{s \geq c, s \in \xi_r} \frac{1}{\psi_{i_s,s}} \cdot \tilde{\rho}_{i_s,s}} & \text{if } r = j \\ 0 & \text{otherwise} \end{cases}$$

4 Graphical representation of the results

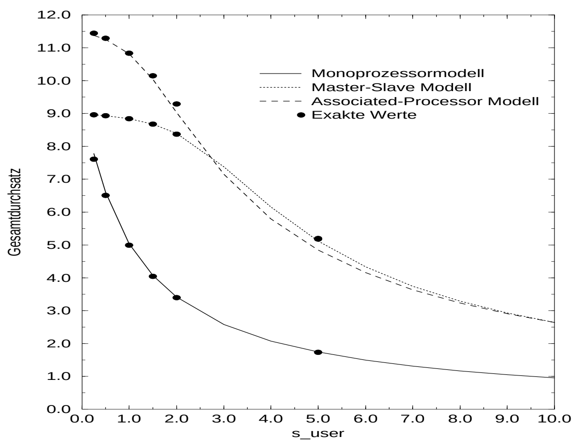

Figure 7: Comparison of ther results for the models

For this results, the standard parameter set with $p_{io} = 0.01$ is used. The computation time for each of the graphs was less than one second and as can be seen, the results of the new approximate technique are very close to the original results.

5 Conclusion

With the help of this new techniques it is possible to analyze the presented operating system models. The time to compute the performance measures for each of the three models was less than one second and the differences between the original results (which we got from measurements on the real system) and the approximated results were very small. Because of the very good results of these approximate formulas it is planned to develop queueing models for other operating systems (like MACH or BS2000) and to use this new technique to analyze the system.

References

[BGJZ94] Bolch G., Greiner S., Jung H., Zimmer R.: *The Markov Analyzer MOSES*, Technical Report TR-I4-10-94, Institute of Mathematical Machines and Data Processing IV University Erlangen-Nuernberg, June 1994

[BoGr94] Bolch, G. and Greiner, S.: *Approximate Analytical Performance Evaluation of a UNIX-Based Multiprocessor Operating System*, Technical Report TR-I4-01-94, Institute of Mathematical Machines and Data Processing IV University Erlangen-Nuernberg, Jan. 1994

[Bolc89] Bolch, G.: *Leistungsbewertung von Rechensystemen mittels analytischer Warteschlangenmodelle*, Teubner, Stuttgart, 1989.

[Grei93] Greiner, S.: *Leistungsbewertung des Betriebssystems UNIX mit Hilfe approximativer analytischer Verfahren*, Studienarbeit am IMMD IV, Erlangen 1993.

[Jung91] Jung, H.: *Leistungsbewertung UNIX-basierter Betriebssysteme für Multiprozessoren mit globalem Speicher*, Dissertation FAU Erlangen-Nürnberg, 1991.

[Kauf84] Kaufmann J.S.: *Approximation Methods for Networks of Queues with Prioritys* , Performance Evaluation Vol 4, pp. 183 - 198 , 1984.

[Munt73] Muntz, R.R.: *Poisson Departure Process and Queueing Networks*, Proc. of the 7th Annual Princeton Conf. on Information Sciences and Systems, Princeton University pp.435 - 440 , March 1973.

[Sevc77] Sevcik, K.C.: *Priority Scheduling Disciplines in Queueing Network Models of Computer Systems*, Proc. IFIP Congress, North Holland, pp.565 - 570, 1977.