PUC

Series: Monografias em Ciência da Computação
Nº 9/78

A HEAVY TRAFFIC MODEL OF A MULTIPROGRAMMING SYSTEM

by

Gunter R.H. Bolch

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 – CEP-22453
RIO DE JANEIRO – BRASIL

PUC/RJ - DEPARTAMENTO DE INFORMÁTICA

Series: Monografias em Ciência da Computação
Nº 9/78

Series Editor: Michael F. Challis May, 1978

A HEAVY TRAFFIC MODEL OF A MULTIPROGRAMMING SYSTEM *

by

Gunter R. H. Bolch

* This work has been partially sponsored by FINEP.

For copies contact:

Rosane T.L.Castilho
Head, Setor de Documentação e Informação
Depto. de Informática - PUC/RJ
Rua Marques de São Vicente, 209 - Gávea
22435 - Rio de Janeiro - RJ - Brasil

ABSTRACT -

In the present paper we consider an analytical queneing network model of a multiprogramming system with symmetric processor, several peripheral devices and an abstraviles distributed number of programs in the system.

In order to find a simple model for this system we assume that the processor are always ackive and with this "heavy traffic assumption". We obtain a much simple model than the models which are derived with the wellknown methods of Gordon/Newell [1] and Busen [2].

We also give a relation to check whether the heavy traffic condition is satisfied.

KEYWORDS -

Multiprogramming, multiprocessor system, queeing network model, responsetime, shroughpart, utilisation, queuelenght.

RESUMO -

No presente artigo considera-se um modelo analítico de redes de filas deum sistema de multiprogramação com processadores simé - tricos, diversos periféricos e programas com distribuição qualquer.

Para obter um modelo simples deste sistema, supõe-se que os processadores estão sempre ativos e com esta suposição de tráfego peado obtém-se um modelo muito mais simples do que os derivados dos conhecidos métodos de Gordon/Newell [1] e Buzen [2].

Dá-se também uma relação para testar se a condição de tráfego pesado é satisfeita.

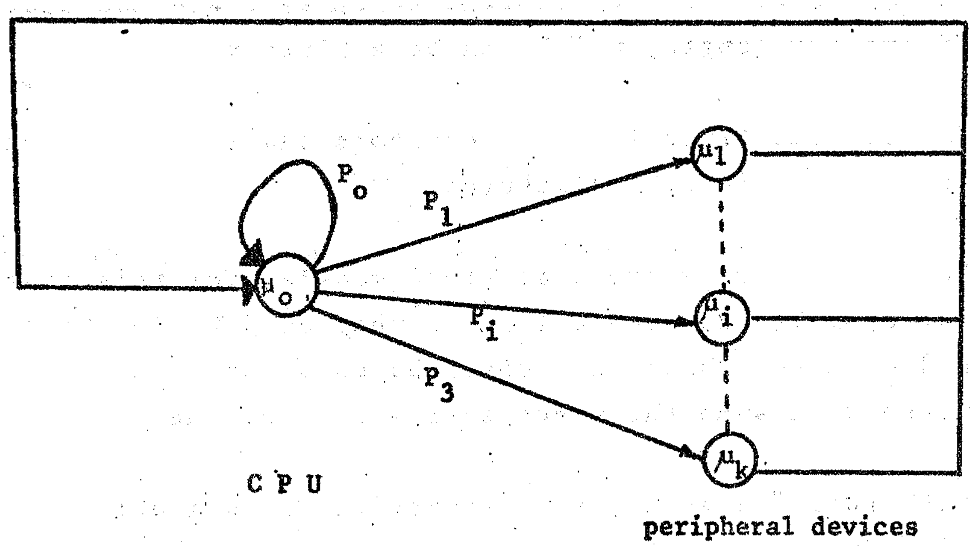
PALAVRAS CHAVES

Multiprogramação, sistema de multiprocessadores, modelos de redes de filas, tempo de resposta, vasão, utilização, comprimento de fila.

CONTENTS

1 - The Multiprogramming System

The multiprogramming system we consider is composed of a CPU with several symmetric processors, an arbitrary number of peripheral devices and a main memory with finite storage capacity (Fig. 1.1).



m_o number of processors
k number of peripheral devices
μ_i service rate
P_i transition probability

Figure 1.1 The multiprogramming system

After being served in the CPU a job leaves the system (and the main memory) with the probability P_o or needs a peripheral device with the probability P_i and is then served in the CPU again etc. The released storage region is allocated to one or move jobs which then join the queue in front of the CPU. In front of the main memory a great number (with meanr) of jobs is waiting to find a free storage region.

The transition probabilities P_i an the service rates μ_i are known, as well as the storage capacity H and the distribution of program length, which can be arbitrary.

The service times in the processors and the peripheral devices are exponentially distributed.

There already exists analytical models for this system: for example [2] with a fixed number of jobs in the system and [3] with an arbitrarily distributed number. In the present paper we assume the processors are always active.

With this "heavy traffic assumption" we are able to find a simple, (but for most applications sufficiently exact) model to calculate all interesting system variables in a short time without a computer. In order to check the heavy traffic condition and to estimate the error if we make the heavy traffic assumption, we derive special relations from sensitivity and accuracy considerations.

2 - The Number of Programs in the System

The number of programs in the system depends on the storage capacity H of the main memory (which is constant), and on the program length with the arbitrary distribution Fg(x). We are interested in the probability q_n

$$q_n = P[n \text{ programs in the main memory}]$$

for which we find solutions for example in [3] or in [4]

We have

$$q_n = q_{n\,min} - q_{(n+1)min} \qquad (2.1)$$

and the mean number $\bar{N}$ of jobs in the system

$$\bar{N} = E[N] = \sum_{n=1}^{\infty} q_{n\,min} \qquad (2.2)$$

with

$$q_{n\,min} = P[\text{at least } n \text{ programs in the main memory}]$$

We obtain $q_{n\,min}$ with the p d f fng(x) of n program lengths, which is the convolution of n p d f's fg(x) of one program length

$$fng(x) = fg(x)\ (*)\ fg(x)\ (*)\ fg(x)\ (*)\ \ldots \qquad (2.3)$$

with the Laplace-Transform:

$$F^*ng(s) = (F^*g(s))^n \qquad (2.5)$$

$$q_{n\,min} = \int_{-\infty}^{H} fng(x)\,d_x$$

<u>Example 1:</u> Exponential distribution

$$Fg(x) = 1 - e^{-\frac{x}{\xi}}$$

ξ = mean program-length

$$q_{n\ min} = 1 - \sum_{i=0}^{n-1} \frac{H^i}{i!\xi^i} \cdot e^{-\frac{H}{\xi}} \qquad (2.6)$$

<u>Example 2:</u> Normal distribution

$$Fg(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot \int_{-\infty}^{x} e^{-\frac{1}{2}\left(\frac{y-\xi}{\sigma}\right)^2} dy$$

σ^2 = variance

$$q_{n\ min} = \frac{1}{\sigma\sqrt{2n\pi}} \cdot \int_{-\infty}^{H} e^{-\frac{1}{2}\left(\frac{y-n\xi}{\sigma\sqrt{n}}\right)^2} dy \qquad (2.7)$$

3 - The Heavy Traffic Approximation

If there are many jobs in the system, we may assume that processor queue never empties. Then for the utilization of the processors we have:

$$\rho_o = 1$$

This is our heavy traffic assumption. ρ_o cannot be greater than 1, for in this case the queue length would grow to infinity, which is not possible, since the mean number of jobs in the system is constant $\bar{N}$.

The condition for this assumption is that there are a sufficient number of jobs in the system. This can only be the case if the main memory is big enough and if there is a large number of jobs waiting infront of the main memory.

We make further comments about this matter in chapter 5.

With this heavy traffic assumption it is now very easy to calculate most of the interesting variables of the system.

With $\rho_o = \frac{\lambda o}{mo.\mu}$ and $\rho_o = 1$

we get the arrivalrate at the CPU

$$\lambda_o = m_o . \mu \qquad (3.1)$$

and the throughput

$$\lambda = p_o \cdot \lambda_o \qquad (3.2)$$

since

p_o = P[Job leaves the system after being served in the CPU]

= P[Job in the processor queue is a newly-arrived job]

λ_o is also the departurate from the CPU and therefore the arrival rate at the peripheral devices and for the arrivalrates at the single peripheral devices we get:

$$\lambda_i = P_i \cdot \lambda_o \tag{3.3}$$

and for the utilization ρ_i.

$$\rho_i = \frac{\lambda_i}{\mu_i} \tag{3.4}$$

With utilization ρ_i we are now able to calculate all interesting variables of the peripheral devices with the wellknown relation for M/M/1 queueing systems, see for example [5]:

Discrete distribution of number of jobs in the devices

$$P_i(n_i) = P[n_i \text{ jobs in the devices}] = (1-\rho_i)\, \rho_i^{n_i} \tag{3.5}$$

Mean number of jobs in the device:

$$\bar{N}_i = \frac{\rho_i}{1-\rho_i} \tag{3.6}$$

Queue length in front of the device:

$$\bar{N}_{q_i} = \bar{N}_i - \rho_i \tag{3.7}$$

System time in the device:

$$T_i = \frac{1}{\lambda_i} \cdot \bar{N}_i \tag{3.8}$$

and the waiting time:

$$W_i = \frac{1}{\lambda_i} \cdot \bar{N}_{q_i} \tag{3.9}$$

If we have several peripheral devices of the same type, for example three drums or two printers, then we have only one queue infront of these devices and have to use then the formulas of M/M/m queueing systems. [5]

For the CPU, using equation (2.2) and (3.6) we get the following expressions for the the mean number of jobs:

$$\bar{N}_o = \bar{N} - \sum_{i=1}^{k} N_i \qquad (3.10)$$

and the queue length

$$\bar{N}_{q_o} = \bar{N}_o - m_o \qquad (3.11)$$

and with Little's result [6]

$$T_o = \frac{1}{\lambda_o} \cdot \bar{N}_o \qquad (3.12)$$

and

$$W_o = \frac{1}{\lambda_o} \cdot \bar{N}_{q_o} \qquad (3.13)$$

A very important variable for computer systems is the response time T, is is the time from the instant a jobs joins the system until it is completely served.

In our cases T is the sum of the waiting time T_W a job has to wait until it may enter the main memory and the system time Ts a job is in the main memory

$$T = T_W + T_s$$

T_W we get immediately from the throughput λ (equation (3.2)) and the mean number of jobs waiting in front of the main memory:

$$T_W = \frac{r}{\lambda}$$

and T_s from the mean number $\bar{N}$ of jobs in the system and Little's result [6]

$$T_s = \frac{1}{\lambda} \cdot \bar{N}$$

and finally:

$$T = \frac{1}{\lambda} (r+\bar{N}) \qquad (3.14)$$

The calculation of the discrete distribution of the number of jobs in the CPU is a bitmore complicated and is there fore for the next chapter.

4 - The Number of Jobs in the CPU

To calculate the discrete distribution of the number of jobs in the CPU we use the known distribution of the number of jobs in the whole system and in the peripheral devices.

With

$$P(i,j) = P[i \text{ jobs in the peripheral devices, } j \text{ jobs in the system}]$$

we get

$$P_o(n_o) = \sum_{i=n_o}^{\infty} P(i-n_o, i) \qquad (4.1)$$

since there must be $(i-n_o)$ jobs in the peripheral devices, if there are n_o jobs in the CPU and i jobs in the whole system.

The probabilities

$$q_n = P[n \text{ jobs in the main memory}]$$

and P_n = P[n jobs in the pefipherak devices]

are independent.

q_n depends only on the storage capacity and the distribution of the program length, and P_n only on the distribution of the servi ce times and the transition probabilities.

There fore we have:

$$P(i,j) = P_i \cdot q_j \qquad (4.2)$$

and

$$P_o(n_o) = \sum_{i=n_o}^{\infty} P_{i-n_o} \cdot q_i \qquad (4.3)$$

In many practical cases q_n is greater than 0 only in a small interval and then we have to sum only a few steps.

(see the example in chapter 6).

If there is a constant number N of jobs in the system we have:

$$q_N = 1$$
$$q_n = 0 \qquad n \neq N$$

and we get in this case

$$P_o(n_o) = P_{N-n_o} \qquad (4.4)$$

Often it is sufficient to assume a constant number $\bar{N}$ of jobs in the system also when we have an arbitrary distribution of this number and we then get the following approximation for $P_o(n_o)$

$$P_o(n_o) = P_{\bar{N}-n_o} \qquad (4.5)$$

The accuracy of this approximation is sufficient in many cases (for example the system in chapter 6) and the cost is much less.

We may calculate the probability P_n with the probabilities

$P_i(n_i)$ in the peripheral devices. Since these probabilities are independent P_n is the convolution of all $P_i(n_i)$. But this is a very lengthy operation and so we replace the k peripheral devices with different utilizations by k peripheral devices with the same utilization ρ. We obtain this ρ with the condition that the mean number of jobs in the devices is the same in both cases:

$$\sum_{i=1}^{k} \bar{N}_i = k.\frac{\rho}{1-\rho}$$

$$\rho = \frac{\sum\limits_{i=1}^{k}}{k+\sum\limits_{i=1}^{k}\bar{N}_i} \qquad (4.6)$$

The discrete distribution $P^*(n)$ of the number of jobs in one of these replaced peripheral devices is

$$P^*(n) = (1-\rho)\rho^n \qquad (4.7)$$

P_n is the convolution of k of these probabilities

$$P_n = P^*(n) \circledast P^*(n) \circledast P^*(n) \circledast \ldots\ldots \circledast P^*(n)$$

With the z-Transform of P_n

$$P(z) = \frac{(1-\rho)^k}{(1-\rho_z)^k}$$

we finally get:

$$P_n = \frac{1}{(k-1)!} .(1-\rho)^k.\rho^n.\sum_{i=1}^{k-1} n+i \qquad (4.8)$$

5 - Accuracy and sensitivity considerations

5.1 - The sensitivity

Because of the approximation the value of the CPU utilization ρ_o is a little too high (for example 1 instead of 0,98) and therefore the values of the CPU arrivalrate λ_o, the through - put λ and the utilization ρ_i of the peripheral devices are too high by the same factor (see equations 3.1 - 3.4).

These values are the upper limits for the respective variables.

To evaluate the method it is important to know something about the sensitivity of the interesting variable with respect to the error in the utilization ρ_i

We consider the mean number $\bar{N}_i$ of jobs in the peripheral devices and define the sensitivity E_i as follows:

$$\frac{\Delta \bar{N}_i}{\bar{N}_i} = E_i \frac{\Delta \rho_i}{\rho_i} \qquad (5.1)$$

und get with

$$\Delta \bar{N}_i = \frac{\partial \bar{N}_i}{\partial \rho_i} . \Delta \rho_i \qquad (5.2)$$

and equation (3.6)

$$E_i = \frac{1}{1-\rho_i} \qquad (5.3)$$

Figure 5.1 shows how this sensitivity depends on the utilization ρ_i

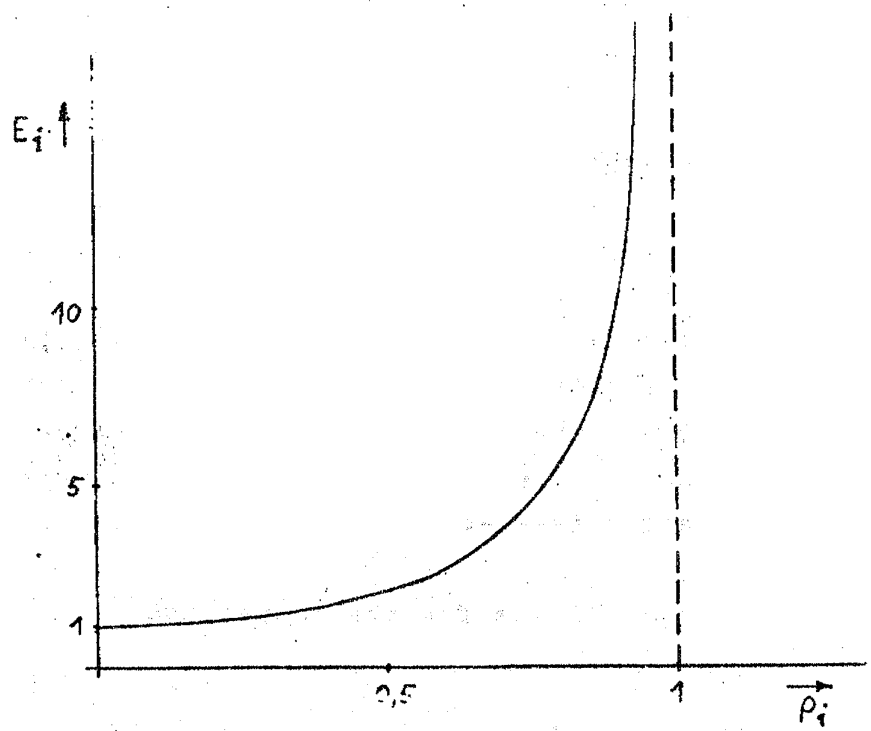


Fig. 5.1 Sensitivity of the mean number of jobs in the peripheral devices.

From equations (5.3) and Figure(5.1) we see that for ρ_i close to 1 errors have a much greater influence than for ρ_i close to 0. If for example ρ_i has an error of 1% this yields error of 5% for $\rho_i = 0,8$ and only 1,43% for $\rho_i = 0,3$

Similar for the probilities $P_i(n_i)$ we get

$$\frac{\Delta P_i(n_i)}{P_i(n_i)} = [n_i - \bar{N}_i] \frac{\Delta \rho_i}{\rho_i} \qquad (5.4)$$

We recognize from (5.4) that values close to the mean $\bar{N}_i$ will be accurate and that the sensivitity grows linearly by with respect to the distance from the mean $\bar{N}_i$.

Example:

We consider the system of chapter 6 and calculate the value of the mean $\bar{N}_i$ at some values of ρ_o (see table 5.1). Our approximate model has three general properties, which are also apparent from table 5.1.

1. The values from the approximate model (ρ_o=1) are upper limits for the real values.

2. The errors are greaters the greater the values of ρ_i

3. We still get acceptable values, when $\rho_i \leq 0,8$ and $\rho_o \geq 0,95$

ρ_o	$\bar{N}_1$ ρ_1=0,37	$\bar{N}_2$ ρ_2=0,51	$\bar{N}_3$ ρ_3=0,66	$\bar{N}_4$ ρ_4=0,46	$\bar{N}_5$ ρ_5=0,81
1	0,60	1,04	1,92	0,84	4,21
0,99	0,59	1,02	1,87	0,82	4,00
0,95	0,55	0,94	1,67	0,76	3,30
0,90	0,51	0,85	1,45	0,69	2,67

Table 5.1 Mean number $\bar{N}_i$ of jobs in the peripheral devices for some values of the CPU utilization ρ_o

5.2 - The Heavy Traffic Condition

To apply our approximate method it is necessary to know whether the heavy traffic condition ($\rho_o \approx 1$) is satisfied. This can only be the case if there are a sufficient number of programs in the system or in the main memory. Our problem is there fore to find the minimum storage capacity H min. We have the following relation between the storage capacity H and the CPU utilization ρ_o, see Figure 5.2 [3]

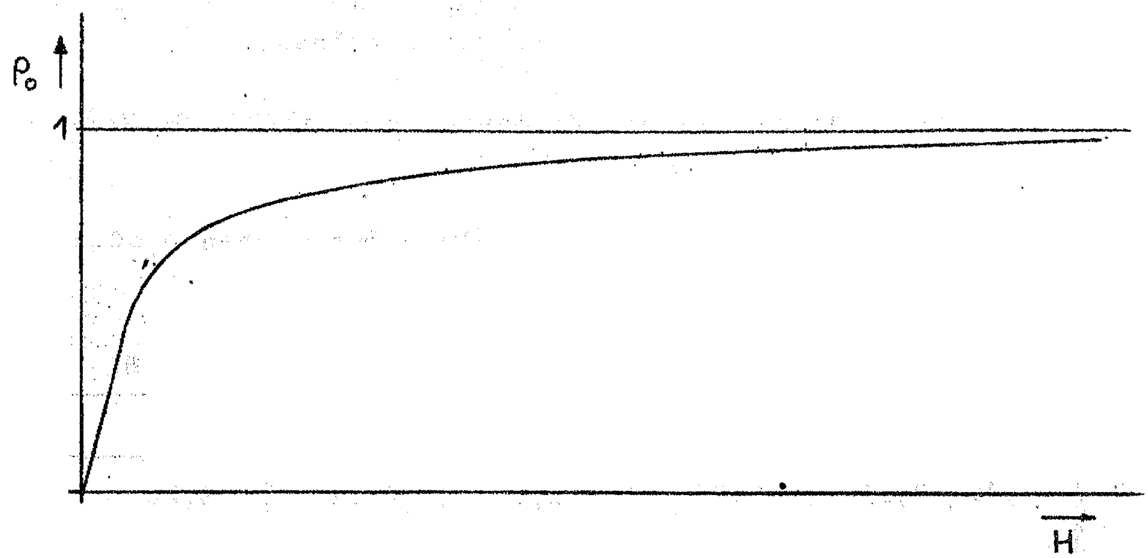


Figure 5.2 CPU utilization ρ_o versus storage capacity H.

For H = 0 we have $\rho_o = 0$, since there are no jobs in the system. At the beginning we have a linear slope.(if we double H we have the double number of jobs in the system and there fore the double value of ρ_o), and for $H \to \infty$ we have $\rho_o \to 1$.

This relation corresponds to

$$\rho_o(H) = 1 - e^{-\nu H} \qquad (5.5)$$

The unknown system parameter ν can be calculated if we assume that there is space for only one program in the main memory Then we have

$$H = \xi \quad (\xi \text{ program length})$$

During one cycle this job remains $1/\mu o$ secs in the CPU and $1/\mu$ secs in the peripheral devices.

$$\frac{1}{\mu} = \sum_{i=1}^{k} p_i \cdot \frac{1}{\mu_i} \qquad (5.6)$$

We consider in equation (5.6) that the service time in the peri pheral devices is 0 with the probability p_o (see Figure 1.1) Then in this case we get the CPU utilization ρ_o immediately

$$\rho_o(\xi) = \frac{1}{mo} \cdot \frac{\frac{1}{\mu_o}}{\frac{1}{\mu_o} + \frac{1}{\mu}} \qquad (5.7)$$

or

$$\rho(\xi) = \frac{x}{m_o} .$$

with

$$X = \frac{\text{servicetime in one processor}}{\text{servicetime during one cycle}}$$

and with equation (5.5)

$$\nu = - \frac{1}{\xi} \ln \left(1 - \frac{x}{m_o} \right) \qquad (5.8)$$

From equation(5.5) and (5.8) we can derive a nice formula to check whether the storage capacity H satisfies the heavy traffic condition:

$$\frac{Hmin}{\xi} = \frac{\ln (1 - \rho_{omin})}{\ln (1 - X/m_o)} \qquad (5.9)$$

We have to choose $\rho_o$min taking into account considerations of accuracy as we did in chapher 5 and we have to estimate or to calculate x with equations (5.6) and (5.7) and get an approximate value of the minimum storage capacity Hmin.
With the realistic value x = 1/2 we get the following values of $Hmin/\xi$.

ρ_o \ m_o	1	2	3
0,98	5,6	13,6	21,4
0,95	4,3	10,4	16,4
0,90	3,3	8,9	12,6

Table 5.2 Minim um storage capacity H_{min}/ξ with x= 1/2 and some values of CPU utilization ρ_o.

We see from table 5.2 that for the example in chapter 6 22 programs must find space in the main memory if x=1/2 and we demand ρ_o min =0,98.

6 - Example:

We shall now demonstrate the approximate model using a special 3 processor system.
We have as storage capacity H = 350 SU (storage units) and a normally distributed program length with mean ξ = 15 SU and standard deviation σ = 5 SU.

The mean service times $1/\mu_i$ and the transition probabilities are given in table 6.1.

Nr.	Device	$1/\mu_i$	p_i
0	Processor	71,8	0,093
1	Plotter	347,8	0,026
2	Drum	38,4	0,317
3	Disk	48,7	0,323
4	Card reader	160,3	0,068
5	Printer	111,8	0,173

Table 6.1 Mean servicetimes $1/\mu_i$ in time units TU and transition probabilities of the components of the system.

First we check with equation(5.9) whether the heavy traffic condition in satisfied. We get the mean service time in the peripheral devices from equation (5.6) and x from equation (5.7)

$$\frac{1}{\mu} = 74,0 \qquad\qquad x=0,49$$

and from equation (5.9) we finally obtain the minimum storage capacity H_{min} if we demand $\rho_o min=0,98$

$$Hmin = 21,9\ \xi = 328,5 < 350 = H$$

The heavy traffic condition is satisfied.

6.1 - Number of Jobs in the system.

From equations (2.5) and (2.2) we get the mean $\bar{N}$ and from equation(2.1) the discrete distribution q_n of the number of jobs in the system: see table 6.1 and figure 6.1

n	18	20	22	23	24	26	28
q_n	0,001	0,051	0,220	0,241	0,183	0,041	0,003

Table 6.1 Discrete distribution q_n of the number of jobs in the system. (E(N) = 22,89)

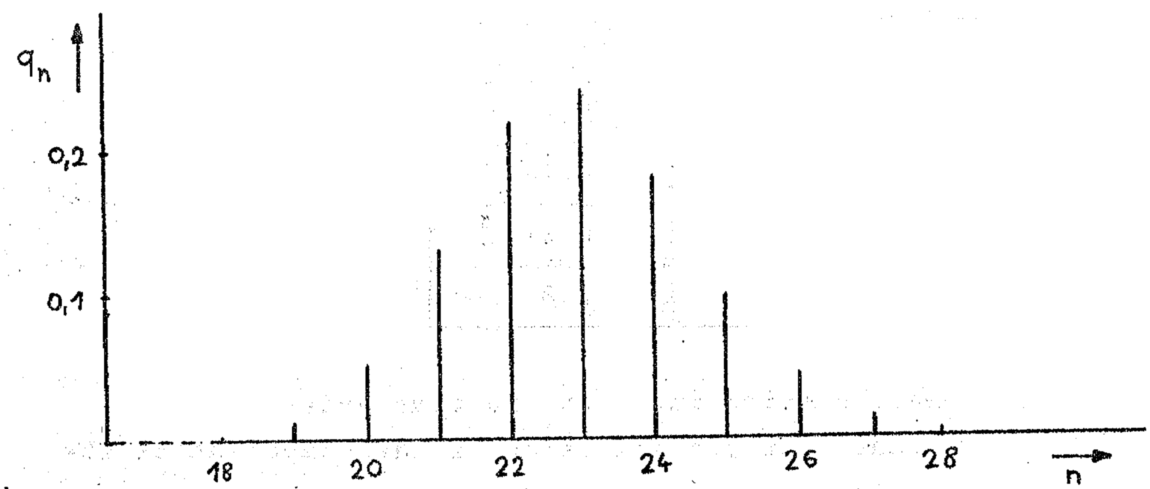


Figure 6.1 Discrete distribution q_n of the number of jobs in the system.

6.2 - Peripheral devices

In table 6.2 we find the utilization ρ_i and the mean number of jobs $\bar{N}_i$ for the peripheral devices from equation(3.4) and (3.6) and the exact values for this variable from [3]

	ρ_i	ρ_i ex.	$\bar{N}_i$	$\bar{N}_i$ ex.
1	0,373	0,371	0,595	0,589
2	0,510	0,506	1,039	1,019
3	0,658	0,653	1,924	1,858
4	0,455	0,452	0,836	0,822
5	0,808	0,802	4,197	3,772

Table 6.2 - Utilization ρ_i and mean number $\bar{N}_i$ for the peripheral devices compared with the exact values ρ_i ex. and $\bar{N}_{iex.}$

Only for the printer we have a noticeable deviation of $\bar{N}_i$. There are two reasons for this effect. First the sensivity which is greater for high values of ρ_i (see chapter 5.1); second the limited number of jobs in the system, which we didn't consider for the peripheral devices. In cases of long queues this limited number of jobs in the system has an influence not only on the CPU but also on the peripheral devices. For this case we compare the discrete distribution $P_5(n_5)$ of the number of jobs in the printer of our approximate model with the exact distribution [3] in figure 6.2

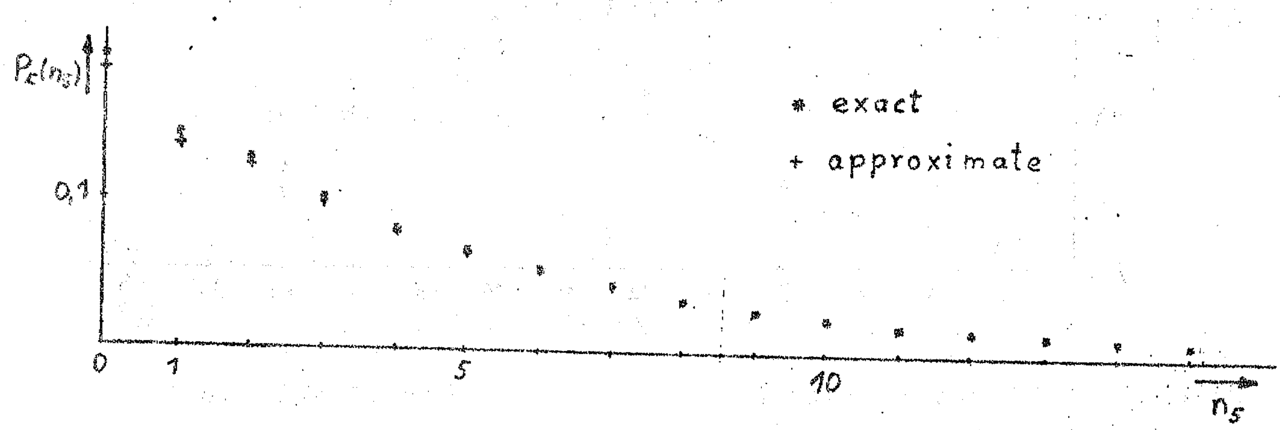


Figure 6.2 Discrete distribution of number of jobs in the printer

6.3 - <u>C P U</u>

The mean number of jobs in the CPU $\bar{N}_o$ is, from eq. (3.10):

$$\bar{N}_o = 14,30$$

and the exact value from [3]

$$\bar{N}_o \text{ ex} = 14,83$$

$\bar{N}_o$ is less than $\bar{N}_o$ ex, because the mean number $\bar{N}_i$ in the peripheral devices are a little toogreat and the number in the system is exact. The exact distribution for this case [3] is compared with our approximate equation (4.3) in Figure 6.3a and with our simplified approximate equation (4.4) in Figure 6.3b

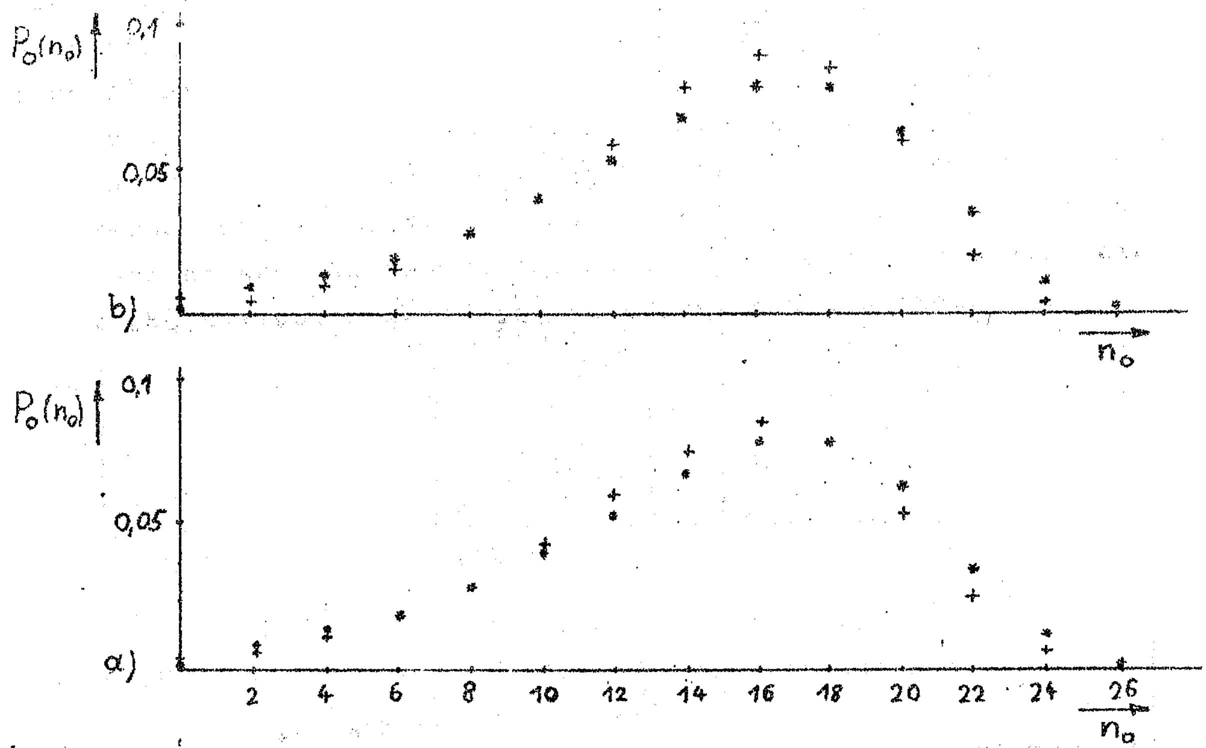


Figure 6.3 Discrete distribution of the number of jobs in the CPU

a) exact* and approximate+ values eq. (4.3)

b) exact* and approximate+ values eq. (4.4)

7 - Short Summary of the Modelling

We conclude this paper with a short summary of all the steps necessary to arrive at the approximate model of our multiprogramming system.

We consider a multiprogramming system with m_o symmetric processors and k peripheral devices corresponding to Figure 1.1.

The servicetimes are expontially distributed with means $1/\mu_i$ and the transition probabilities $p_o, \dots p_k$ are known. In front of the main memory of storage capacity H are waiting a large number of programs (mean number r) with arbitrary but known distribution of program length.

Step 1 heavy traffic condition

We have to check

$$\frac{H_{min}}{\xi} \leq \frac{H}{\xi} \qquad \xi \text{ mean program length}$$

with

$$\frac{H_{min}}{\xi} = \frac{\ln(1-\rho_o)}{\ln(1-X/m_o)}$$

$\rho_o \approx 0{,}98$ for good approximations (often $\rho_o = 0{,}95$ is sufficient)

$$X = \frac{1/\mu_o}{1/\mu_o + 1/\mu} \quad \text{with } \frac{1}{\mu} = \sum_{i=1}^{k} p_i \cdot \frac{1}{\mu_i}$$

Step 2 Number of jobs in the system

$$q_n = P[n \text{ jobs in the system}]$$

$$= q_{n\,min} - q_{(n+1)min}$$

Mean number:

$$E[N] = \bar{N} = \sum_{n=1}^{\infty} q_{n\,min}$$

with

$$q_{n\,min} = \frac{1}{\sigma\sqrt{2\pi n}} \int_{-\infty}^{H} e^{-\left(\frac{y-n\xi}{\sigma\sqrt{n}}\right)^2} dy$$

if the program length is normally distributed

and

$$q_{n\,min} = 1 - \sum_{i=0}^{n-1} \frac{H^i}{i!\,\xi^i} \cdot e^{-\frac{H}{\xi}}$$

for exponential distribution

with the mean ξ and the variance σ^2

and

$$q_{n\,min} = \int_{-\infty}^{H} fng(x)\, dx$$

with

$$fng(x) = fg(x) \circledast fg(x) \circledast \ldots\ldots$$

for arbitrarily distributed program length with the pdf fg(x)

Setp 3 <u>Throughput and responsetime</u>

Throughput

$$\lambda = p_o \cdot \lambda_o \text{ with } \lambda_o = m_o \cdot \mu_o$$

Response time

$$T = \frac{1}{\lambda} (\bar{N} + r)$$

Step 4 <u>Other interesting system variables</u>

The formulas to calculate the other interesting system variables are given in the following table:

	Peripheral Devices	CPU
Utilization	$\rho_i = \frac{\lambda i}{\mu i}$; $\lambda_i = p_i . \lambda_o$	1
Mean Number of Jobs	$\bar{N}_i = \frac{\rho_i}{1-\rho_i}$	$\bar{N}_o = \bar{N} - \sum_{i=1}^{k} \bar{N}_i$
Queue Length	$\bar{N}_{q_i} = \bar{N}_i - \rho_i$	$\bar{N}_{q_o} = \bar{N}_o - m_o$
Systemtime	$T_i = \frac{1}{\lambda_i} . \bar{N}_i$	
Waiting time	$W_i = \frac{1}{\lambda_i} . N_{q_i}$	
Distribution	$P_i(n_i) = (1-\rho_i)\rho_i^{\,n}$	see Step 5

Step 5 <u>Discrete distribution of the number of jobs in the CPU</u>

a) Expensive approximation

$$p_o(n_o) = \sum_{i=n_o}^{\infty} p_{i-n_o} \quad q_i$$

b) Simple approximation

$$p_o(n_o) = p_{\bar{N}-n_o}$$

$\bar{N}$ in this formula is the integer mean of the number jobs in the system.

with

$$p_n = \frac{1}{(k-1)!} (1-\rho)^k . \rho^n \prod_{i=1}^{k-1} n+1$$

and

$$\rho = \frac{\sum_{i=1}^{k} \bar{N}_i}{k + \sum_{i=1}^{k} \bar{N}_i}$$

REFERENCES

[1] Gordon.W.I. and Newell, G.F.: Closed "Queueing Systems with Expontial Servers", Operation Research 15 (1967) p 254-265

[2] Buzen,I.: "Queueing Network Models of Multiprogramming" Ph. D. Thesis Division of Engineering and Applied Science, Haward University, Cambridge, Mass. (1971)

[3] Bartsch,B., Bolch,G.: " Ein Analytisches Modell für Symmetrische Mehrprocessor an lagen" Informatik-Fachberichte Band 9 (1977) Springeverlag p 93-108

[4] Walke,B.: "Durchsatzberechnung für Rechen an lagen beiwählbarer Aufteilung des Arbeitsspeichers" Dissertation Stuttgart (1975)

[5] Kleinrock,L.: Queueing Systems, Volume 1: Theory, John Wiley & Sons, New York (1975)

[6] Little,J.D.C.: A Proof of the Queueing Formula $L=\lambda.W$, Operations Research Vol. 9 (1961), p. 383-387