DK 621.394.343:621.394.01

# Queueing Networks with Degenerate Exponential Servers*

Bernhard Walke

## Introduction and Summary

In literature, results of measuring a large number of variables suitable for describing computer-center operations have been published, e. g. [1]. Such variables can be used in simulation studies and for analytical treatment of computer models. Simulation models are frequently highly sophisticated whereas analytical models are often much simpler. The former show a great similarity to real systems whereas the latter tend to lack credibility. The reason for this is that the Markovian assumptions are the only ones where a mathematical treatment has a chance of success. Nevertheless computational results of analytical models often agree very well with simulation output [1], so that the drastic simplification seems to be justified. Two typical simplifications are the approximation of highly skewed distribution functions (d.f.) by negative exponential d.f.'s and the substitution of complicated CPU[1]) scheduling algorithms by simple **f**irst-**c**ome-**f**irst-**s**erve (FCFS) or **p**rocessor-**s**haring (PS) servicing.

We intend here to show that with the help of a trick it is also possible to apply the analytical results of Markov models (with exponential service-time d. f.) to distribution functions with greater variance. The trick involves the scheduling algorithm of the servers and does not affect the analytical computation. Instead of FCFS or PS scheduling we use a preemptive discipline similar to those used in real systems. In so doing one can explain why simulation results from realistic models with realistic scheduling algorithms correspond so well to analytic results from a Markov model. The idea is presented by way of an example: Computation of the equilibrium distribution of customers in a closed queueing network with one degenerate exponential server.

*) This research was partly supported by the 2nd EDV-Program of the German Federal Republic.

1) Central Processing Unit

First we consider the piecewise exponential d.f. as a good approximation to measured compute-time d.f. and explain the well-known **p**reemptive-**p**riority-**c**lass-**p**rocessing (PPCP) scheduling algorithm which is optimal for maximum throughput and minimum mean response-time. Then we introduce the degenerate exponential d.f. as a somewhat simpler approximation and recognize PPCP scheduling to degenerate in this case to LCFS-P scheduling. By this we show the computability of queueing networks with LCFS-P scheduling suddenly to be very attractive also for operating systems.

We demonstrate how to use the well-known computational algorithms for exponential servers also in the case of degenerate exponential servers. An example shows that computational results for a simple queueing network with a degenerate exponential server and simulation results for a better (piecewise exponential) approximation to measured compute-time d.f. do, infact, agree very well. This demonstrates that such networks, together with a suitable interpretation, can also be applied to compute steady state parameters under realistic service-time d.f. and optimal scheduling.

## 1. The model

For a large variety of queueing networks with nearly arbitrarily connected queues and servers, computational algorithms are known ([2, 3] and ref. there). To illustrate the new idea we use the well-known central server network [3] (Fig. 1). More complicated networks can be modified in the same way, the only difference being the greater number of parameters. Closed queueing networks are well suited for representing the overall behavior of multiprogrammed computer systems. Such networks can be used to study the behavior of fixed-partition systems in batch mode. Here there are $n$ programs running concurrently on the system and $M$ queues, each of which is in front of a server. Program behavior is characterized by an alternating sequence of compute and I/O intervals where each interval is possibly preceded by a queueing delay. A terminated program is instantly replaced by an identical job. A given job may not use the CPU and I/O service stations concurrently. Throughput can be computed from the probability that the CPU is idle.

## 2. Processing times

From measurements of cumulative-frequency d.f.'s of compute time $t_R$ per interval (the total amount of compute time between consecutive page or segment faults), it is known that an exponential distribution function

$$P(t_R \leq t) = 1 - e^{-\mu_1 t} \tag{1}$$

with mean $E(t_R) = 1/\mu_1$

and variance $\sigma^2(t_R) = 1/\mu_1^2$

is only a rough approximation. Figure 2 shows this for measurements made on the Telefunken Computer TR 440 (compare curves 1 and 3).

A better approximation is possible using a hyperexponential function

$$P(t_R \leq t) = 1 - \sum_{i=1}^{j} w_i \cdot e^{-\mu_i t} \tag{2}$$

$$\mu_i > 0, \quad w_i > 0, \quad \sum_{i=1}^{j} w_i = 1$$

with mean $E(t_R) = \sum_{i=1}^{j} w_i/\mu_i$ (3)

and variance $\sigma^2(t_R) = \sum_{i=1}^{j} 2\, w_i/\mu_i^2 - (E(t_R))^2$. (4)

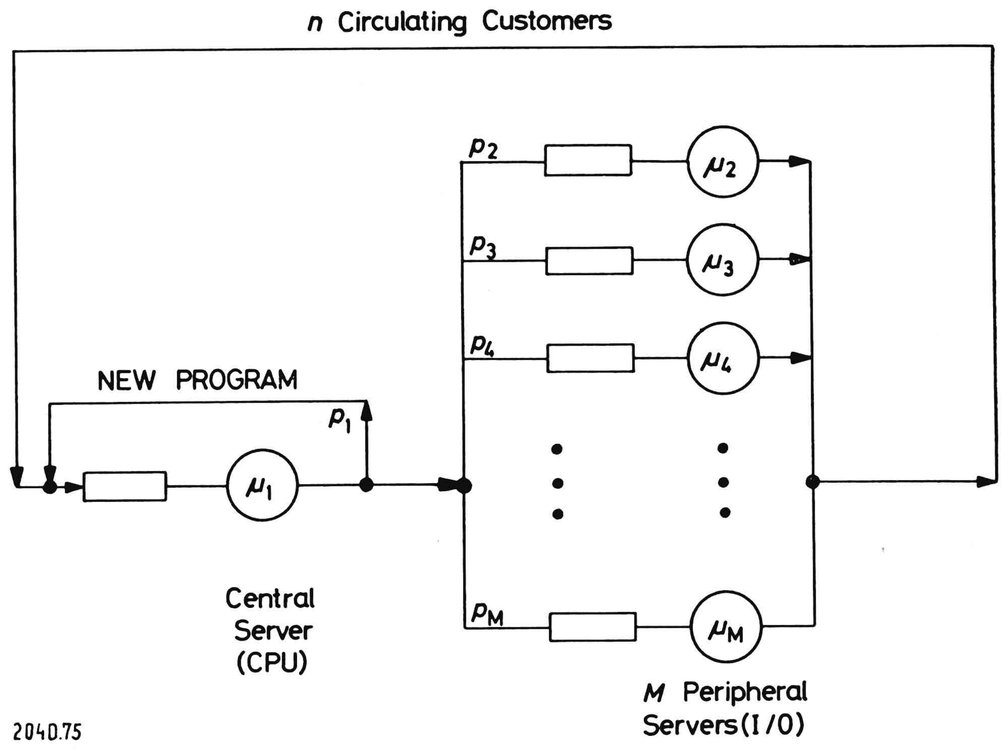


Fig. 1. Central server model of multiprogramming [3]

queue

The coefficient of variance

$$C = \sigma(t_R)/E(t_R) \tag{5}$$

is $C = 1$ for exponential functions and $C \geq 1$ for hyperexponential functions. A two-phase hyperexponential d.f. (curve 2 of Figure 2) approximates the experimental data much better than an exponential function.

Treatment of queueing networks with hyperexponential processing-time d.f. is possible [4], but throughput greatly depends on the scheduling algorithm. If *overhead* is considered, no optimal algorithm is known. *Without* overhead, throughput-optimal scheduling of hyperexponential processing time must be such that all waiting intervals always have consumed the same processing time [5]. Consequently, infinitesimal time slices must be used, which is not acceptable for small overhead either.

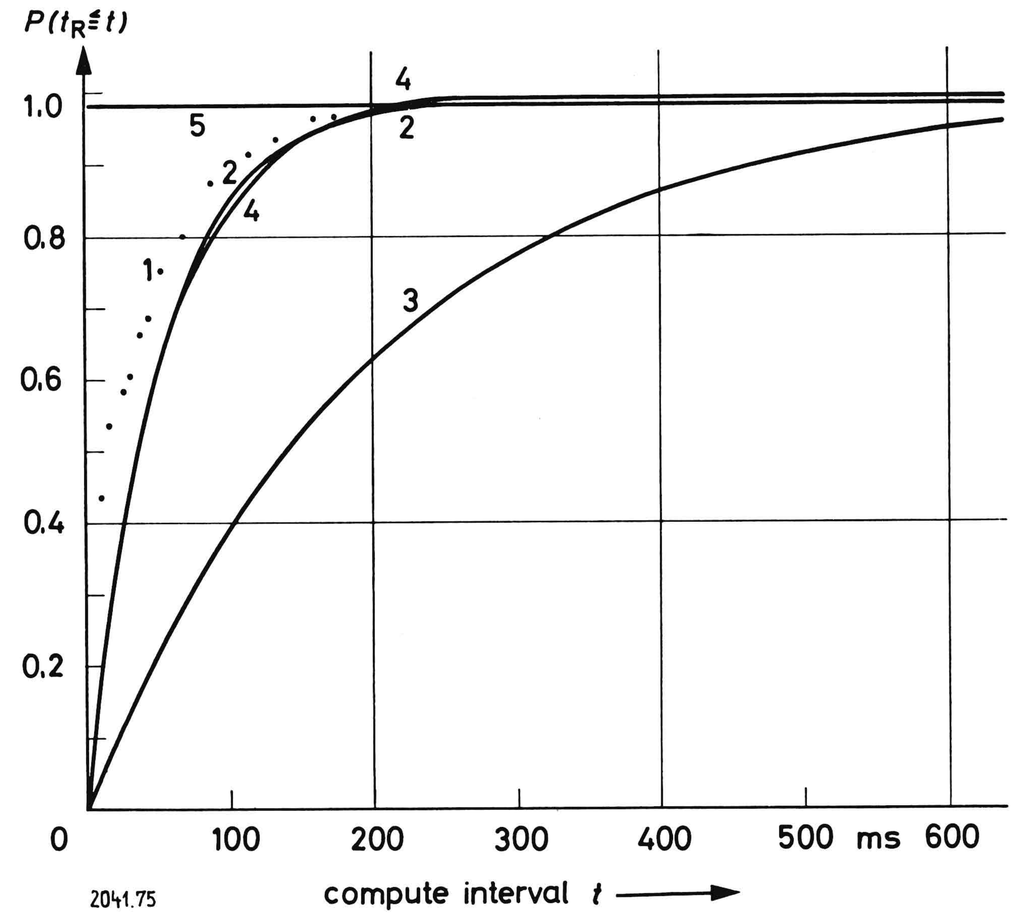


Fig. 2. Experimental data for the length of compute time intervals $t_R$ (TC—TR 440, batch mode) and different approximations with the same mean and (curves 2, 4, 5) also the same variance

1 measured; mean = 0.2 s, variance = 4.5 $s^2$
2 hyperexponential;
$\mu_1 = 19.7\ s^{-1}$, $\mu_2 = 0.067\ s^{-1}$, $w_1 = 0.99$
3 exponential
4 piecewise exponential;
$\mu_1 = 18.4\ s^{-1}$, $\mu_2 = 0.067\ s^{-1}$, $t_{g1} = 0.25$ s
5 degenerate exponential;
$p_g = 0.983$, $\mu = 0.085\ s^{-1}$

Recent work [7, 8] has shown that another approximation using a piecewise exponential function of the form

$$P(t_R \leq t) = 1 - \exp\left\{-\sum_{i=1}^{j} t_{gi}(\mu_i - \mu_{i+1}) - \mu_{j+1} \cdot t\right\} \qquad (6)$$

$$t_{gj} \leq t \leq t_{gj+1};\ j = 0, 1, \ldots K;$$

$$0 = t_{g0} < t_{g1} < \ldots < t_{gK} < t_{gK+1} = \infty$$

$$\text{with} \quad \mu_1 < \mu_2 < \ldots \mu_i \ldots < \mu_{K+1} \qquad (7)$$

has some advantages: The throughput-optimal scheduling algorithm for zero overhead is known [5, 7] very simple, and also "plausibly optimal" for (small) nonzero overhead. In most cases two exponential pieces are sufficient (Fig. 2, curve 4). Little is gained by using more than two pieces [8]. For two pieces, the mean and variance are given by

$$E(t_R) = 1/\mu_1 + (1/\mu_2 - 1/\mu_1)\, e^{-\mu_1 t_{g1}} \qquad (8)$$

and

$$\sigma^2(t_R) = \mu_1^{-2} + (2/\mu_2 - 2/\mu_1)\, e^{-\mu_1 t_{g1}} \times$$

$$\times \{t_{g1} + 1/\mu_2 + (1/\mu_2 - 1/\mu_1)\, e^{-\mu_1 t_{g1}}\}. \qquad (9)$$

A similar, empirically determined distribution function has already been used by Rehmann et al. [9] to discribe the compute load of large time sharing systems.

Optimal scheduling of customers with the d.f. given by eq. 6 requires preemptive priority for first-class compute intervals (with $t_R \leq t_{g1}$) over all other classes, for second-class intervals ($t_{g1} \leq t_R < t_{g2}$) over all higher-order classes (3., 4., ..., $K + 1$), and so on. The order within each class is arbitrary (e.g. FCFS) [5]). We call this algorithm **p**reemptive-**p**riority-**c**lass-**p**rocessing (PPCP). One advantage of using a piecewise approximation rather than a hyperexponential approximation is that optimal scheduling becomes trivial (e.g. [9]).

Another advantage is that this approximation is the starting point for a tricky but very powerful simplification: Our experience with piecewise exponential functions, using two pieces to approximate different measurements for compute-intervals, is that first-class intervals contribute only a small portion of the CPU load, the main load coming from the second class. This finding encouraged Marte [10] to formulate a new distribution function

$$P(t_R \leq t) = 1 - (1 - p_g)\, e^{-\mu t} \qquad (10)$$

which is called *degenerate exponential*. From Figure 2 (curve 5) it can be seen that, although only a rough approximation, it is nevertheless much better than an exponential function because not only the first but also the second moment of measured data is taken into account. Mean and variance are given by

$$E(t_R) = (1 - p_g)/\mu \qquad (11)$$

$$\sigma^2(t_R) = (1 - p_g^2)/\mu^2 \qquad (12)$$

Using the new function (eq. 10) short service times are approximated by the time $t_R = 0$ and the remaining service time by an exponential d.f. (Figure 3). These short service times appear in eq. (10) with probability $p_g$. The degenerate exponential distribution function can be considered as a special two-phase hyperexponential d.f., or a special case of the piecewise exponential d.f. and last but not least as a generalization of an exponential d.f.

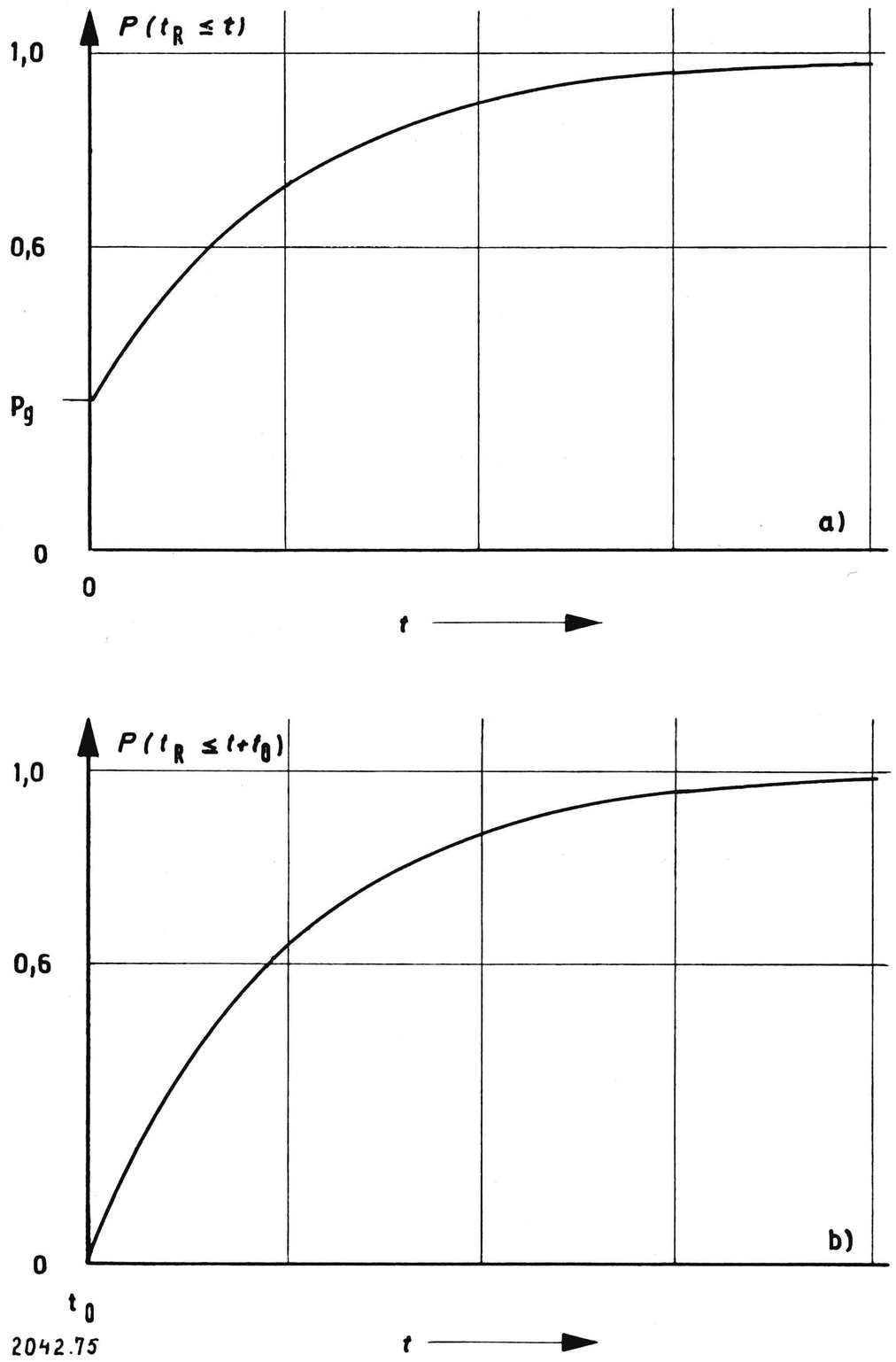


Fig. 3. Important attribute of the degenerate exponential function

a) Degenerate exponential compute-time d.f.
b) Distribution of compute time after a given consumed CPU time $t_0 \geq 0$.

The optimal scheduling algorithm remains the same as in the case for piecewise exponential functions with two pieces: First-class intervals (with $t_R = 0$) have preemptive priority over second-class intervals. In the case of degenerate exponential d.f. this algorithm PPCP is identical to the well-known **l**ast-**c**ome-**f**irst-**s**erve-**p**reemptive (LCFS-P) algorithm [11].

As will be shown in the next chapter, this d.f. (eq. 10) is well suited for analytical treatment of Markov models.

## 3. Networks with degenerate exponential service time d. f.

Following the solution of Gordon and Newell [6] for general closed queueing networks with exponential servers and a fixed number of circulating customers, Buzen [3] has found simple computational algorithms to solve for the basic equilibrium distributions. His 'central server network' (Fig. 1) has $M$ exponential servers and $n$ circulating customers. Customers leaving the 1 st (central) server proceed to the $i^{th}$ server with probability $p_i$ ($i = 1, \ldots, M$), and customers leaving one of the $M - 1$ peripheral servers proceed directly to the central server with probability one. $\mu_i$ is the mean service rate of the $i^{th}$ exponential server ($i = 1, 2, \ldots, M$) when it is busy (e.g. for the central server $\mu_1 = 1/E(t_R)$; eq. (1)).

In many cases the coefficient of variance of the central server's service-time d.f. is much greater than 1, in which case the degenerate exponential d.f. would be a better approximation than the exponential d.f. In many cases also the computer system to be modeled has some preemptive or time-slicing CPU

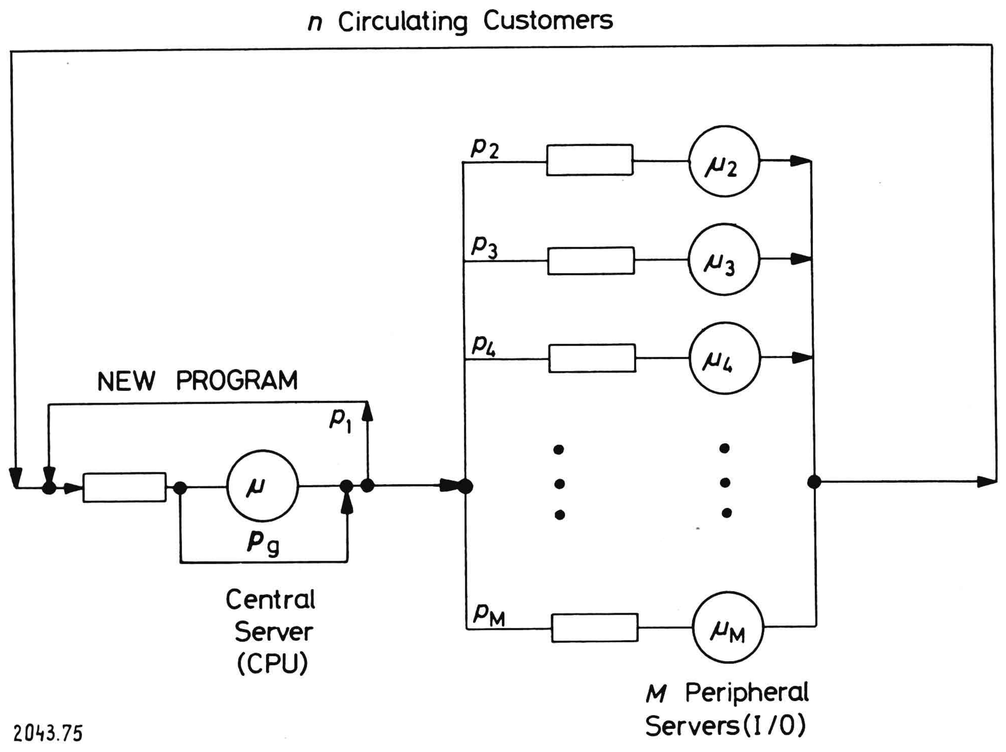


Fig. 4. Modified central server model of multiprogramming

queue

scheduling algorithm which should be modeled approximately. We take this into account by using LCFS-P scheduling [11], being sure to choose an optimal algorithm to serve compute-intervals which have a degenerate exponential distribution. The network (Fig. 1) must be slightly modified by adding a new path from the central server queue directly to the output line of the central server (Fig. 4). This path is used under LCFS-P scheduling with probability $p_g$ by zero length compute-intervals. The mean service rate of the central server must be altered from $\mu_1$ to $\mu$ (eq. 10)

$$\mu = \mu_1 (1 - p_g) \tag{13}$$

taking into account that the mean service rate of nonzero compute-intervals must be less than $\mu_1$. By doing this the mean service time $E(t_R)$ (eq. 11) for all compute-intervals together remains unchanged (eq. 1). Now we can proceed to compute the equilibrium distribution of customers in the network $P(n_1, n_2, \ldots, n_M)$, where $n_i$ is the number of customers present at the $i$-th facility, in the same way as is demonstrated in [2, 3]. We only have to replace $\mu_1$ in [2, 3] by $\mu/(1 - p_g)$ (eq. 13). The same is valid for the evaluation of the marginal distribution of exactly $k$ customers present at the $i^{th}$ facility. The same technique is appliable for the more general networks of Gordon and Newell [6] or Jackson [12]. Each server $i$, having a degenerate exponential instead of exponential service-time d.f., must be replaced by a server with LCFS-P scheduling and the modified service rate $\mu^{(i)} = \mu_i (1 - p_{gi})$, where $\mu^{(i)}$ is the service rate in eq. (10), $p_{gi}$ the probability for zero length service-time at the $i^{th}$ server, and $\mu_i$ the service rate for the $i^{th}$ server in the equations from Gordon and Newell for an exponential server $i$.

### *3.1 An example*

We consider a very simple network (Fig. 5) which we have also studied by simulation. This network is a strong simplification of the network in Fig. 4. It has only one peripheral server, and the

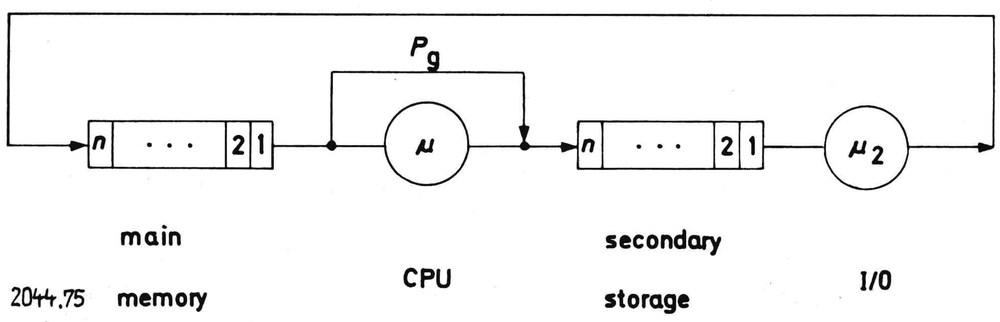


Fig. 5. Network with

1) degenerate exponential CPU time,
2) optimal CPU-scheduling LCFS-P, and
3) exponential I/O time

probability $p_1$ for a customer using the new program path is zero. Let $N$ be the number of customers in the CPU queue (including the job possibly being serviced) and $n - N$ the number in the I/O-server queue. Then one obtains from the stationary condition for an exponential central server [2, 3]

$$P(N)\ \mu_1 = P(N-1)\mu_2 \tag{14}$$

and with eq. (13) for a degenerate exponential central server (Fig. 5)

$$P(N)\ \frac{\mu}{1 - p_g} = P(N-1)\mu_2\,. \tag{15}$$

Our network is so simple that we need not use the computational algorithms of [3] to solve for $P(N)$. By recursion we obtain

$$P(N) = P(0) \left\{ \frac{\mu_2 (1 - p_g)}{\mu} \right\}^n \tag{16}$$

and with the normalizing condition

$$\sum_{i=0}^{n} P(i) = 1 \tag{17}$$

the throughput is

$$D = [1 - P(0)]\ \frac{\mu}{1 - p_g}\,. \tag{18}$$

From eq. (16, 17) we have

$$P(0) = \frac{1 - \dfrac{\mu_2 (1 - p_g)}{\mu}}{1 - \left\{ \dfrac{\mu_2 (1 - p_g)}{\mu} \right\}^{n+1}}\,. \tag{19}$$

The non-idle probability $1 - P(0) = D \cdot E(t_R)$ of the CPU is

$$D \cdot E(t_R) = \frac{\left\{ \dfrac{E(t_H)}{E(t_R)} \right\}^n - 1}{\left\{ \dfrac{E(t_H)}{E(t_R)} \right\}^{n+1} - 1} \tag{20}$$

expressed in terms of mean I/O service time $E(t_H) = 1/\mu_2$ and mean compute-interval time $E(t_R)$, (eq. 11).
If one knows $E(m)$, the mean number of compute-intervals per program, then it is possible to compute the program throughput

$$D_P = D/E(m)\,. \tag{21}$$

Reintroducing $p_1$ (Fig. 4) and thereby a geometric distribution for the mean number of compute-intervals per program, one can compute $E(m)$ from $p_1$

$$E(m) = \frac{1}{p_1}\,. \tag{22}$$

Remember that the normalized throughput eq. (20) does not depend on $p_g$, the probability of compute-intervals having length zero in eq. (10). This throughput is the same as for exponential compute-interval d.f. with an arbitrary scheduling algorithm. Throughput does not depend on $p_g$ because we have prescribed the special (and optimal) scheduling algorithm LCFS-P. For other algorithms (e.g. FCFS) the throughput depends on $p_g$.

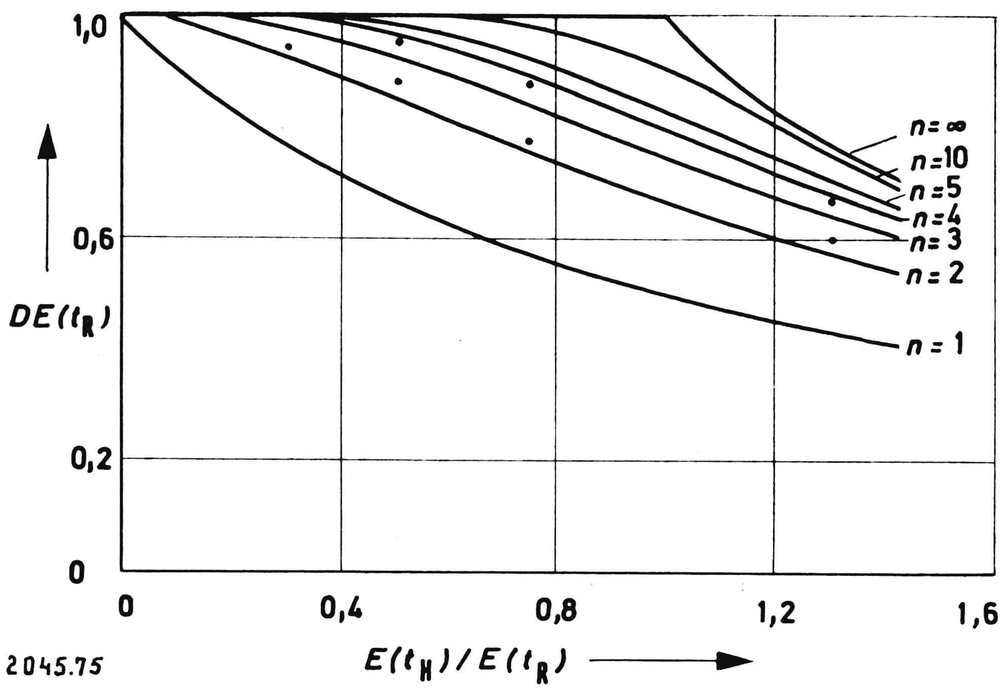


Fig. 6. Computational results and simulation for degenerate (curves) and piecewise exponential d.f. (dots). Simulation was carried out for $n = 2$ and 3

$D \cdot E(t_R)$ = CPU non-idle probability

## 4. Evaluation and assessment

We have seen that the piecewise exponential d.f. is well suited to approximate experimental data (Fig. 2). Using this d.f. and the PPCP discipline, CPU scheduling is similar to those algorithms used in real computer systems. The use of more time slices to schedule compute intervals (not jobs!) does not substantially influence throughput or mean response time (as is demonstrated in [8]).

A very interesting goal is to compute throughput under optimal CPU scheduling for d.f.'s with a coefficient of variance $C > 1$. For piecewise exponential d.f.'s we did not succeed, but for the rough approximation of a piecewise exponential function by a degenerate exponential function we did succeed. Figure 6 answers the question for our model (Fig. 5), how large an error is made by computing results for a degenerate exponential function instead of a piecewise exponential function with parameters shown in Figure 2. One can see (for $n = 2, 3$) that there is no essential difference between computational results for degenerate exponential d.f. (curves) and simulation results (dots) for piecewise exponential d.f. For $n = 1$ and $\infty$ there is obviously no difference in throughput which depends on the d.f. because only mean I/O service time and mean CPU time determine throughput in these cases. For other $n$ there is a difference which arises from the possibility of simultaneously servicing compute times of all first-class intervals of the piecewise exponential functions and I/O times. This is not possible for degenerate exponential compute times (where first-class intervals have zero length). The difference vanishes as the number $n$ of programs in the network increases.

Now we can see why comparisons of computational results for Markov models coincide so very well with simulation results, e.g. [1], although Markov models are mostly treated only for exponential service-time d.f. and simulation is carried out for highly skewed service time d.f.'s and some time slicing. Such comparisons are reasonable and show good agreement because, by using a well adapted scheduling algorithm, it is possible to reduce the effect on throughput for a coefficient of variance $C > 1$ to the same degree as if we had $C = 1$ and had used arbitrary scheduling.

ZE/FI-UL

## References

[1] A. L. Scherr: An analysis of time-shared computer systems MIT-Press, Cambridge, Mass., June 1967

[2] J. Buzen: Analysis of system bottlenecks using a queueing network model. ACM SIGOP Workshop, Apr. 1971, Harvard University

[3] J. P. Buzen: Computational algorithms for closed queueing networks with exponential servers, Comm. ACM, Vol. 16, No. 9, Sept. 1973, pp. 527—531

[4] J. C. Browne, K. M. Chandy, J. Hogarth and Ch. C. A. Lee: The effect on throughput of multiprocessing in a multiprogramming environment IEEE Trans. Comp., Vol. C 22, No. 8, Aug. 1973, pp. 728—735

[5] G. Olivier: Kostenminimale Prioritäten in Wartesystemen vom Typ M/G/1 Elektron. Rechenanl. 14 (1972), H. 6, S. 262—271 (German)

[6] W. J. Gordon and G. F. Newell: Closed queueing systems with exponential servers, Oper. Res. 15, 2 (Apr. 1967), pp. 254—265

[7] G. Marte: Optimal scheduling for time-shared computer systems with piecewise exponential computing time distribution, Int. Comp. Sympos. 1970, Bonn, May 21/22

[8] B. Walke und H. J. Küspert: Teilnehmerrechensysteme: mittlere Verweilzeiten bei optimaler Rechnerkernzuteilung. Elektron. Rechenanl. 13 (1971), H. 5, S. 193—199 (German)

[9] S. L. Rehmann and S. G. Gangwere, Jr.: A simulation study of resource management in a time-sharing system. AFIPS, FJCC 1968, pp. 1411—1429

[10] G. Marte: Zur Synthese von Teilnehmerrechensystemen, Wiss. Ber. AEG-TELEFUNKEN 44 (1971), H. 3, S. 114—123 (German)

[11] R. R. Muntz: Analytic models for computer system performance analysis. Lecture Notes in Computer Science, Vol. 8, NTG/GI-Symposium March 1974, Springer-Verlag, Berlin/Heidelberg/New York 1974, pp. 246—265

[12] J. R. Jackson: Networks of waiting lines, Oper. Res. 5, 4 (1957), pp. 518—521