BULETINUL

INSTITUTULUI POLITEHNIC BUCUREŞTI

SERIA INGINERIE ELECTRICĂ

TOMUL LIII, NR. 1–2, ANUL 1991

EXTRAS

M. COLAJANNI, S. TUCCI

APPROXIMATE SOLUTION OF FORK-JOIN QUEUEING SYSTEM

APPROXIMATE SOLUTION OF FORK-JOIN QUEUEING SYSTEM

M. COLAJANNI, S. TUCCI *

In this paper we propose a new approach for the solution of Fork-Join queueing systems. The solution is obtained by replacing the Fork-Join subsystem with an equivalent state dependent server. The service rate of this center is determined exploiting the isomorphism between the state diagram of the short-circuited subsystem and the one of a cyclic closed queueing network.

Systems with two and three parallel servers have been analysed. A large number of simulation runs have shown a high accuracy of the analytical results.

1. Introduction

Fork-Join models arise in many interesting applications. In effect different areas present activities which develop partly in sequential and partly in parallel way. Similar processes have points of splitting, in which sequential jobs fork into two or more concurrent operations and points of synchronization in which the activity becomes sequential.

The diffusion of multiprocessor architectures, of concurrent programs with constructs like PARBEGIN/PAREND of parallel operations (like WRITE) in replicated distributed database, makes actual and interesting the performance evaluation tools for such models.

But the requirement of parallel activities doesn't appear only in the computer science area. Many modern factories, in fact, have flexible manifacturing areas with activities of disassembly (Fork), parallel production and points of assembly (Join).

We consider the classic Fork-Join model shown in Figure 1 and described in the next paragraph.

We define job the sequential process which arrives to point of Fork and splits into tasks, parallel operations independently executable on different processors. After completion tasks have to wait, on respective join-queue, that all their brothers have been served. We assume Poissonian arrival process of jobs and independent and exponentially distributed with same mean service times.

The analysis of such system is quite difficult; in fact only few results have appeared in the literature.

In 1982 Heidelberger and Trivedi have solved the closed model with asyncronous (Fork without Join) [7] and synchronous task (Fork with Join) [8] for two and three parallel servers.

* Dept. of Electronics Engineering, Università di Roma „TOR VERGATA"

Only recently are appeared results for the classic open Fork-Join system.

Flatto et al. [6] have computed the generating function of the state probabilities for the model with two exponential servers. Baccelli et al. [1] have obtained lower and upper bounds on the response time for Fork-Join systems with n parallel servers and general distributions service times. Simple approximations of the mean response time for n exponential server system has been proposed by Nelson et al. [10]. Duda et al. [5] introduced three variants of classic Fork-Join system, solving them analitically for $n = 2$ and using a markovian solver for $n > 2$.

Our method starts from approximate analysis of Duda et al. [5] in which the Fork-Join system is represented by a flow-equivalent node with state dependent rate of service. This one is computed analysing the markovian process which models the Fork-Join subsystem short-circuited.

The most important result of our paper consists in showing the existing isomorphism between the state diagram of such process and the one of a cyclic closed queueing network with state dependent service rates using an opportune ordering of the states. Next, defined the service rate of the flow-equivalent node, we obtained the state stationary probabilities for an open Fork-Join system with two and there parallel processors, and thus we compute performance parameters as Mean response time and Mean number of customers.

The accuracy of method is proved comparing our analytical results with simulation ones. A large number of tests demonstrate that our approach is valid and it is sensitive to system congestion rate only.

2. The Model

We consider a classic Fork-Join system (Figure 1) consisting of a point of Fork, n identical parallel nodes, each formed by a single server and a FIFO queue, n join-queues with Global-FIFO discipline and a point of Join. We define job the sequential process which arrives to point of Fork and task the whole operation parallely executable on each processor.

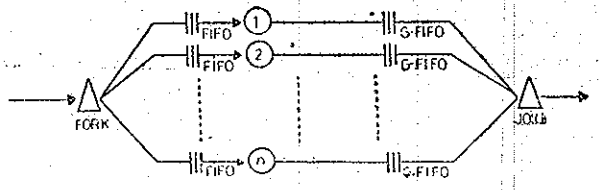

Fig. 1 — The classic Fork-Join model

The jobs arriving at this system are split into n brother tasks, which are independently executed on the n servers and then have to wait, on the respective join-queue, the completion of all their brothers...

We assume Poissonian with mean $1/\lambda$ the distribution of arrival process of jobs and independent and exponentially distributed with mean

$1/\mu$, for each node the service times. These assumptions simplify the analysis of the system but don't limit much the adherence of the model to real systems.

3. Approximate Analysis of the Model

In this paper the Fork-Join model is solved for $n = 2$ and $n = 3$ parallel servers. The extension of the method to larger number of servers is cumbersome due to the dimension explosion of the state transition diagram of the analysed process; however, at present, models with $n > 3$ are in examination.

The presence in the model of a structure like the primitive of synchronization Join leads at a queueing network which doesn't satisfy the product form conditions. So we make an approximate analysis, which takes the method of Duda-Czachorski [5] as starting point of the solution.

The first step is an application of the Norton's theorem of the flow-equivalent server [3]: the subsystem with the Fork-Join primitives (Figures 2a) is replaced by an equivalent server with state-dependent rate of service. In this way we obtain an open M/M/1 known system (Figure 2b). But in order to obtain the equivalence it is necessary to compute the rate of service $\mu(k)$ in a suitable way.

This is possible solving the short-circuited subsystem (Figure 2c) with nk tasks, for every possible k.

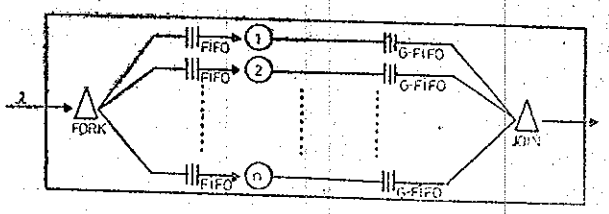

Fig. 2a — The extracted subsystem

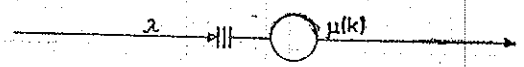

Fig. 2b — The equivalent system obtained

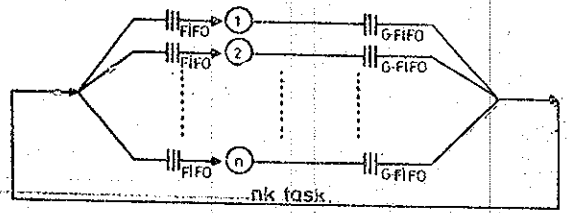

Fig. 2c — The short-circuited subsystem

Getting the equivalent rate of service we can solve the system, knowing the state probability distributions of a M/M/1 [9]:

$$P_0 = \text{Prob}\,(n=0) = \frac{1}{1+\sum_{j=1}^{\infty}\prod_{i=0}^{j-1}\frac{\lambda}{\mu(i+1)}} \tag{1}$$

$$P_k = \text{Prob}(n=k) = P_0 \prod_{i=0}^{k-1}\frac{\lambda}{\mu(i+1)} \quad k=1,2,\ldots \tag{2}$$

By means of these it is possible to obtain all performance parameters of the Fork-Join system: in particular the mean number of customers and the mean response time.

So the problems is led to suitable computation of the service rate $\mu(k)$. To this purpose it is necessary to solve the short-circuited subsystem (Figure 2c) with nk tasks, where $k = 1,2,\ldots$ is the number of job of the Fork-Join system and n is the number of parallel processors. The state dependent service rate $\mu(k)$ is set to the value of throughput of this closed subsystem.

We solve the short-circuited Fork-Join by the analysis of associated markovian process.

If $X_i(t)$ is the number of tasks in the i-th join-queue at time t, we have a Markov chain with space state formed by integers: $j = 0, 1, \ldots, k$ and dimension $s_i = k+1$. Therefore it is possible to characterize the Markov chain corresponding to the operation of the whole subsystem by the number of tasks in each join-queue:

$$[X_1(t),\ X_2(t),\ldots,\ X_n(t)]$$

with:

$$0 \leq X_i(t) \leq k \text{ for: } i = 1,\ldots,n \text{ number of parallel processors}$$

The dimension of this state space should be $(k+1)^n$, but some observation and the assumption made in §2 reduce this value and the analysis becomes more tractable. In fact:

- it is necessary to exclude the contemporary presence of tasks on all join-queues, because this occurence is equivalent to a new arrival at the fork node; in fact it means that all the brother tasks are completed and can leave the join-queue. Hence the markovian process is characterized by $n-1$ dimensional states, because on value is always zero.
- the assumption of parallel serves with the same rate of service allows two simplifications: states like $[i, j]$ and $[j, i]$ (for instance with $n = 3$) should not be regarded as different; moreover it is equivalent to consider null any one of n elements of each state (we suppose always the n-th one).

We can obtain now the state diagram of short-circuited subsystem and demonstrate that, with a convenient ordering of the states, it results

isomorphic to that one of a cyclic closed network with n nodes and particular rates of service.

The following two subsections analyse the cases of Fork-Join systems with two and three parallel processors.

3.1. *The case of 2 Processors* ($n = 2$)

The state diagram of the short-circuited subsystem, described in Figure 2c, is easily obtainable. In this case the state of markovian process $X_1(t)$ are characterized only by the number of tasks in one of two join-queues (we assume the first).

For $k = 1$ (Figure 3a) in the subsystem there are two tasks and two possible situations:

State 0: both tasks are in service; there are no tasks in the join-queue;

State 1: one task have been executed and is in join-queue waiting for the completion of the brother task.

For $k = 2$ (Figure 3b) in the subsystem there are four customers and three possible configurations.

State 0: two tasks are in processing and two are waiting in the queues of servers; there are no tasks in the join-queue;
State 1: one tasks is in the join-queue;
State 2: two tasks are waiting in the join-queue.

Generalization to $k = i$ is straightforward (Figure 3c).

In Figure 4 the state diagram of a cyclic closed network with $k = i$ customers and two identical nodes with FIFO queues and exponentially distributed servers with rate μ is drawn.

The analogy between the two diagrams is clear: they differ only in the rate of transition from state 0 to state 1. This results is possible owing to the following state ordering:

— the states of the markovian process of the cyclic network are characterized by the number of customers in the first and in the second node; therefore we have made this choice:

$$0:(i,0) \quad 1:(i\text{-}1,1) \quad 2:(i\text{-}2,2) \ldots i\text{-}1:(1,i\text{-}1) \quad i:(0,i)$$

— the states of the markovian process of the short circuited subsystem are determinated by the number of tasks waiting in the first join-queue; so we have chosen the increasing ordering:

$$0:[0] \quad 1:[1] \quad 2:[2] \ldots i-1\ [i-1] \quad i:[i]$$

The only difference between the two diagrams is overcome assigning to the first node of the cyclic network a a queue-dependent rate of service:

$$\mu_1(i) = \begin{matrix} \mu & i < k \\ 2\mu & i = k \end{matrix}$$

The second one keeps the fixed rate of service :

$$\mu_2(k) = \mu$$

As a consequence of the total isomorphism between the two state diagrams we can say that the two subsystems have the same state probability and the same performances. Therefore the throughput $\lambda(k)$ of the subsystem may be obtained simply solving a product form network with two nodes; one with fixed and the other with queue, dependent rate of service.

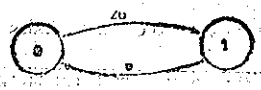

Fig. 3 a — The state diagram of subsystem for $k = 1$

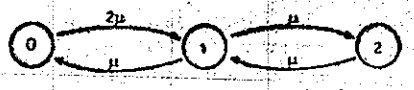

Fig. 3b — The state diagram of subsystem for $k = 2$

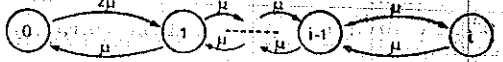

Fig. 3c — The state diagram of subsystem for $k = i$

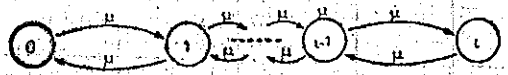

Fig. 4 — The state diagram of the cyclic network

Using [2] we get the following result :.

$$\lambda(k) = \lambda \frac{2k}{2k+1}.$$

Setting the rate of service of the equivalent server to the value of this throughput :

$$\mu(k) = \lambda(k)$$

we can rewrite the expression (1) and (2) of the stationary-state probability in the following form :

$$P_0 = (1-p)^{3/2}$$

$$P_k = P_0 \rho^k (2k+1)!!/(2k)!!$$

in which :

$$(x)!! = \begin{cases} x \cdot (x-2) \ldots 3 \cdot 1 & \text{if } x \text{ is odd} \\ x \cdot (x-2) \ldots 4 \cdot 2 & \text{if } x \text{ is even} \end{cases}$$

and with $\rho = \lambda/\mu$, utilization parameter, which has to be smaller than 1, so that the series in (1) is convergent.

Now it is possible to compute the performance parameters of the Fork-Join system. The most interesting ones are the Mean number of customers ($E[n]$) and the Mean response time ($E[R]$) :

$$E[n] = \sum_{n=1}^{\infty} n \cdot P_n \quad E[R] = E[n]/\lambda$$

Therefore in a Fork-Join system with two parallel processors we have :

$$E[n] = 3/2 \frac{\rho}{(1-p)}$$

$$E[R] = \frac{3/2}{\mu \cdot (1-p)}.$$

The precision of this formula is tested in § 4.

3.2. The Case of 3 Proccessors ($n = 3$).

The analysis of a Fork-Join system with three parallel processors is similar to the preceding one :

- The Fork-Join subsystem is replaced by a flow-equivalent server.
- The queue-dependent rate $\mu(k)$ is found computing the throughput of the short-circuited subsystem.
- We observe that the state diagram of the subsystem and that one of a cyclic closed network with three nodes (Figures 5) are similar.

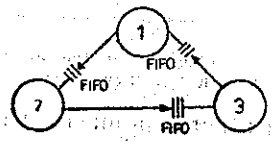

Fig. 5 — The cyclic closed network

The states of the markovian process of the Fork-Join subsystem are now characterized by a couple :

$$[X_1(t),\ X_2(t)]$$

where $X_i(t)$ is the number of tasks waiting in the i-th join-queue. So we have :

$$c = (k+1)(k+2)/2$$

admissible states. Ordering them in the following way :

0 : (0,0)	4 : (2,1)	8 : (3,2)
1 : (1,0)	5 : (2,2)	...
2 : (1,1)	6 : (3,0)	$c-2$: $(k, k-1)$
3 : (2,0)	7 : (3,1)	$c-1$: (k, k)

we obtain (for $k = 3$) the state diagram of Figure 6a. The diagram (Figure 6b) of the markovian process of a cyclic closed network with three

nodes has the same number of states and the same structure if we utilize the follwing arrangement :

$0 : (N, 0, 0)$	$4 : (N-2, 1, 1)$	$8 : (N-3, 1, 2)$
$1 : (N-1, 1, 0)$	$5 : (N-2, 0, 2)$	...
$2 : (N-1, 0, 1)$	$6 : (N-3, 3, 0)$	$c-2 : (0, 1, N-1)$
$3 : (N-2, 2, 0)$	$7 : N-3, 2, 1)$	$c-1 : (0, 0, N)$

Observing the two figures 6 it is possible to note that the diagrams are isomorphic except some rates of transition. The complete equality is obtained choosing particular rates of services for the nodes of the cyclic network :

$$\mu_1(n_1, n_2) = \begin{cases} 3\mu & n_1 = N \\ 2\mu & n_1 <> N \quad n_2 = 0 \\ \mu & \text{otherwise} \end{cases}$$

$$\mu_2(n_3) = \begin{cases} 2\mu & n_3 = 0 \\ \mu & n_3 <> 0 \end{cases}$$

$$\mu_3 = \mu$$

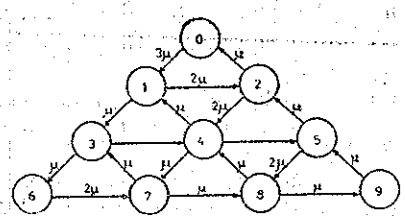

Fig. 6a — The state diagram of the Fork-Join subsystem with $k = 3$ jobs

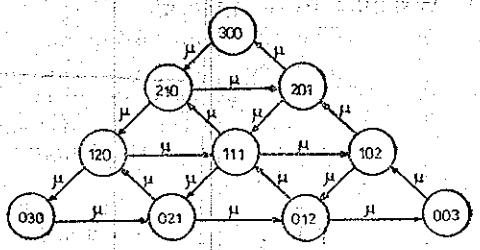

Fig. 6b — The state diagram of the cyclic closed network with $N = 3$ customers

The cyclic network with these rates of service is not in product form, therefore we solve it through the approximate method described in [4].

The expression obtained for the throughput is :

$$\lambda(k) = \mu \frac{2k}{2k+1} \cdot \frac{\Omega(k)}{1/3 + \Omega(k)}$$

where :

$$\Omega(k) = \sum_{i=1}^{k} \cdot \frac{(2i+1)!!}{(2i)!!} .$$

Fixing $\mu(k) = \lambda(k)$, the state probabilities (1) and (2) of M/M/1 system become :

$$P_0 = \frac{1}{1 + \sum_{j=1}^{\alpha} \rho^j \frac{(2j+1)!!}{(2j)!!} \prod_{i=1}^{j} \left(1 + \frac{1}{3 \sum_{k=1}^{i} \frac{(2k+1)!!}{(2k)!!}} \right)}$$

$$P_k = P_0 \rho^k \frac{(2k+1)!!}{(2k)!!} \prod_{i=1}^{k} \left(1 + \frac{1}{3 \sum_{k=1}^{i} \frac{(2j+1)!!}{(2j)!!}} \right)$$

The evaluation of the limit of this series has been very cumbersome; therefore we have decided to interrupt the computation so that further values don't affect the 15-th number after point. We made it for different values of the utilization coefficient ($\rho = 0.1, 0.2, \ldots, 0.9$).

Analogous approach has been followed for the evaluation of the performance parameters :

$$E[n] = P_0 \left[\sum_{k=1}^{\alpha} n \rho^n \frac{(2n+1)!!}{(2n)!!} \prod_{j=1}^{n} \left(1 + \frac{1}{3 \sum_{j=1}^{i} \frac{(2j+1)!!}{(2j)!!}} \right) \right]$$

$$E[R] = E[n]/\lambda .$$

4. Validation of the Approximation

The precedent analysis is an approximation. The method accuracy has been verified comparing the Mean number of customers and the Mean response time obtained by our formulas with simulation results. Both the systems (with 2 and 3 parallel processors) have been simulated with different values for the arrival and service rates : it has been observed that the method is sensitive only to the changes of ρ.

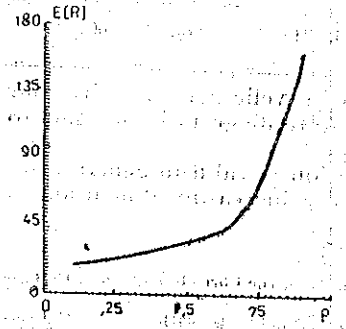

Fig. 7 — Mean response time for $n = 3$

For the Mean response time, the accuracy is very good until ρ is sufficiently less than 1 the system is not much congested). More exactly, like Figure 7 shows, the method tends to overstiming simulation results when $\rho > 0.8$.

The adherence of the results for the Mean number of customers to the simulation is remarkable for every ρ when $n = 3$, whereas there is a tendency to overestimate the parameter when $n = 2$.

In conclusion this approach may be considered particularly efficient because in most cases the results of the method fall into the intervals of confidence or are very close to them (Table 1 and 2).

5. Conclusive Remarks

We have solved Fork-Join systems with two and three parallel processors using a new approach. With regard to accuracy, the method is sensitive to the system congestion rate only; the comparison of our analytical results with the simulation ones demonstrates that the approach is efficacious. The decrease in the accuracy for $\rho > 0.8$ doesn't undermine its validity, because systems so congested are unsual in reality.

Besides there aren't theoretic limitations that prevent us from utilizing this method to Fork-Join systems with more than three parallel servers and, in fact, at present we are studying them. We have observed

Mean Number of Customers *Table 1*

	$n = 2$		$n = 3$	
	METHOD	SIMULATION	METHOD	SIMULATION
$\rho = 0.1$	0.1667	[0.1552, 0.1693]	0.2019	[0.1982, 0.2075]
$\rho = 0.3$	0.6428	[0.6139, 0.6475]	0.7642	[0.7494, 0.7754]
$\rho = 0.5$	1.5000	[1.3802, 1.4723]	1.7452	[1.6780, 1.7445]
$\rho = 0.7$	3.5000	[3.0578, 3.2547]	3.9640	[3.8327, 4.1195]
$\rho = 0.9$	13.500	[12.506, 14.033]	14.646	[13.907, 16.071]

Mean Response Time *Table 2*

	$n = 2$		$n = 3$	
	METHOD	SIMULATION	METHOD	SIMULATION
$\rho = 0.1$	16.667	[16.365, 16.416]	20.191	[19.709, 20.182]
$\rho = 0.3$	21.428	[20.653, 21.032]	25.475	[24.543, 25.707]
$\rho = 0.5$	30.000	[28.271, 29.457]	34.905	[33.031, 35.374]
$\rho = 0.7$	50.000	[46.270, 48.922]	56.629	[52.415, 56.377]
$\rho = 0.9$	150.000	[134.89, 146.62]	162.74	[149.47, 160.19]

that for $n = 4$ too it's possible to find a closed cyclic network with four nodes and appropriate service rates, with a state diagram isomorphic to that of Fork-Join system.

We are induced to think that this relation is valid in general, but the analysis is difficult in consequence of the increasing dimension of state diagram.

REFERENCES

1. *F. Baccelli, A. M. Makowski*, "Simple computable bounds for the Fork-Join queue". INRIA-Rapporte de recherche n. 394 (4/1985)
2. *J. Buzen*, "Computational algorithms for closed queueing networks with exponential servers". Communication ACM, 16 (1973).
3. *K. M. Chandy, U. Herzog, L. Woo* "Parametric analysis of queueing networks". IBM Journal of Res. & Devel., 19 (1/1975).
4. *M. Colajanni, S. Tucci*, "Un metodo aggregativo per la soluzione di reti con tassi di servizio network state dependent" will appear.
5. *A. Duda, T. Czachorski*, "Performance evaluation of the Fork & Join synchronization primitives", Acta informatica, 24 (1987).
6. *L. Flatto, S. Hahn*, "Two parallel queues created by arrivals with two demands". SIAM Journal Applied Math., 44,5 (1984).
7. *P. Heidelberger, K. Trivedi*, "Queueing network models for parallel processing with asynchronous tasks". IBM Research Report-RC 9102 (1982).
8. *P. Heidelberger, K. Trivedi*, "Analytic queueing models for programs with internal concurrency". IBM Research Report-RC 9194 (1982).
9. *S. S. Lavenberg, C. Sauer*, "Computer performance modeling handbook" Academic Press (1983).
10. *R. Nelson, A. N. Tantawi*, "Approximate analysis of Fork/Join synchronization in parallel queues". IBM Research Report-RC 11481 (1985).