ANALYSIS OF BUFFERED MULTISTAGE INTERCONNECTION NETWORK FOR PARALLEL PROCESSORS

INFORMATICA 4/87

Dušan Fajfar
Institute for Teleinformatics
ISKRA Telematika, Kranj
Matija Lokar
University E. K. Ljubljana, Department of Mathematics

UDK 618.3.02

Abstract In the paper a method for calculating distribution law on the number of memory requests in the finite buffer in multistage interconnection network of parallel processors is given. A modified delta network is used for the connecting processors with memory modules. Memory requests on each processor are generated randomly and independently. Two cases of traffic flow are discussed: the constant average rates of requests and the time dependent average rates.

Povzetek V članku analiziramo večnivojsko povezovalno mrežo med procesorji in pomnilniškimi moduli, ki jo sestavljajo elementi z vmesnimi pomnilniki omejene kapacitete. Podana je metoda za izračun porazdelitvenega zakona števila zahtev po podatkih iz pomnilnika v posameznem elementu mreže. Vsak procesor generira zahteve po pomnilniških modulih naključno in neodvisno od ostalih. Obravnavana sta dva primera: časovno konstantno in s časom spreminjajoče se povprečno število zahtev.

Keywords Buffered network, delta network, multistage interconnection network, buffer length, queueing theory.

I. Introduction

In the recent years a lot of new multiprocessor architectures have been proposed. The main problem of any multiprocessor system is its interconnection network. A typical configuration of such system is illustrated in Fig. 1. Many identical processors are connected via an interconnection network to identical memory modules. Each processor should have access to each memory module with requests generated randomly and independently at each processor.

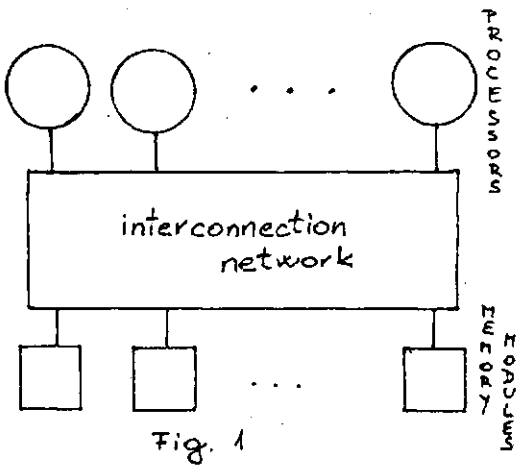

Fig. 1

We have several possible organizations of such processors - memory interconnection. For example, a shared bus in an unexpensive, but admits very low transfer rate. On the other hand maximum transfer rate is attained by the full crossbar switch (with n×m switches where n is number of processors and m number of memory modules). However it is far too complicated and expensive for practical application. Thus we have to use interconnection networks with less then n×m switches. One of them is multistage interconnection network [1]. A lot of them have been presented in the literature ([2],[3],[4],[5],[6]). In this paper we study a delta network with some modifications in the first and the last stage (input process from processors and output process to memory). In section II description of network is given. The network consists of three types of switches. For more detailed presentation see [6]. In sections III, IV and V the distribution law of buffer length for each type of switch is calculated. In section VI results and conclusions are given.

II. The network model and system operation

We discuss a multistage network for connection N processors with N memory modules, where $N = 2^n$. The model for N = 16 is illustrated in Fig. 2.

The network consists of $\log_2 N + 1$ stages. Each stage has N identical switches. With regard to the number of input and output links there are three types of switches. Switches at the first stage have one input and two output

links, switches at the last stage have two input links and only one output link. Switches at all other stages have two input and two output links. To achieve higher transfer rate and to avoid blocking there is a finite buffer at each output link, where the incoming requests wait to be processed further.

As we have finite buffers we propose that in the case when the buffer is full, the incoming requests are lost.

For the time unit we choose the system cycle time. At each cycle only one request can be transmitted through the same link except at the first stage where more than one request can come by input links from each processor. In one time unit only the first request in the buffer can travel from the stage i to stage i + 1. As we can see on Fig. 2. there are no links between the switches of the same stage. The input links of stage i are the output links of the stage i-1. The first stage has input links from processors and the last stage has output links to memory modules. This regularity of the network gives the possibility to analyze it stage by stage instead of the whole network at once. In the next section we give the analysis of the first stage in the section IV we give analysis of the stages indexed from 2 to $\log_2 N$ and in the section V the last stage is analyzed.

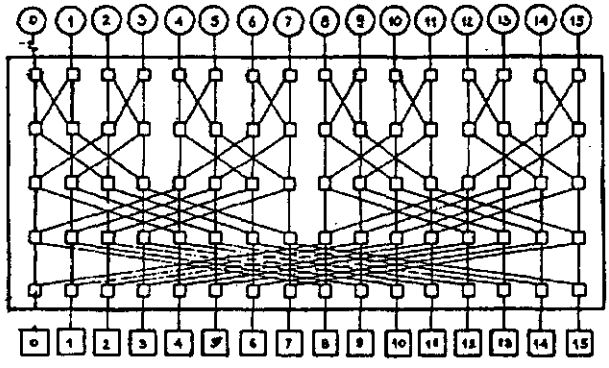

Fig. 2

III. Switches with one input and two output links

Switches with one input and two output links can be found only in the first stage. The input links come from processors and the output links are connected to switches of the second stage. Each processor is connected to only one switch. There are no links between the switches of the same stage, so we have N identical systems. Each system consists of a processor that sends the requests to switch and two output links, each with finite buffer.

As the memory requests are time independent ([6]), we use the Poisson distribution with a given average rate a. The average rate can be different for each processor. Let p^k denote the probability that k requests are sent from processor in one cycle. By Poisson law we get

$$p^k = \frac{a^k e^{-a}}{k!} \qquad (1)$$

We analyze the case where a is constant and the case where a is time dependent, denoted by a(t).

Each request coming from processor is switched in one of two output buffers. As requests are uniformly distributed between memory modules ([6]) there is an equal probability that the incoming request joins the first or the second buffer. By p_m^k we denote the probability that m of k requests enter the first buffer (and k-m requests enter the second buffer). Thus we get

$$p_m^k = \binom{k}{m} (0.5)^k \qquad (2)$$

Let ρ_m denote the probability that m requests enter the first buffer in one cycle. Then

$$\rho_m = \sum_{k=m}^{\infty} p^k p_m^k = \sum_{k=m}^{\infty} a^k e^{-a} / k! \binom{k}{m} (0.5)^k = a^m e^{-a/2} / (2^m m!) \qquad (3)$$

The output process is very simple if we compare it with the input process. In each cycle only the first request from each buffer leaves system.

Let us first analyze stationary system with constant average rate of input process. Since events on both buffers are equal we could analyze only one of them. Let bl denote the buffer length (we propose that all buffers have the same length but we do not have any difficulties when buffers have different length. We just have to calculate the distribution for all switches). The balance diagram is shown in Fig 3.

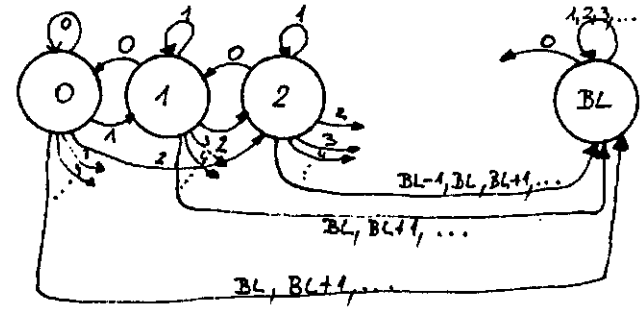

Fig. 3

The nodes in the diagram denote the number of requests in the buffer. If we look at in- and out-coming arcs to each node we get the system of balance equations

where $\Omega(s)$ denotes the probability that we have s requests in the buffer.

$$\Omega(0) = \sum_{m=1}^{\infty} \rho_m = \Omega(1) * \rho_0$$

$$\Omega(s) = [\rho_0 + \sum_{m=2}^{\infty} \rho_m] =$$

$$= \Omega(0) * p_s + \sum_{k=1}^{s-1} \Omega(k) * p_{s-k+1} + \Omega(s+1)\rho_0$$

$$s = 1, 2, \ldots, bl - 1 \quad (4)$$

$$\Omega(bl) = [1 - \rho_0]$$

$$= \Omega(0) * \rho_{bl} + \sum_{k=1}^{bl-1} \Omega(k) * p_{bl-k+1}$$

If we pot ρ_m expressed by (3) in the left side of equations (4) we get the following system of linear equations.

$$\Omega(1) = (e^{-a/2} - 1) * \Omega(0)$$

$$\Omega(s) = [1 - (a/2)e^{-a/2}] =$$

$$= \Omega(0) * p_s + \sum_{k=1}^{s-1} \Omega(k) * p_{s-k+1} + \Omega(s+1)\rho_0$$

$$s = 1, 2, \ldots, bl - 1 \quad (4')$$

$$\Omega(bl) = e^{-a/2}$$

$$= \Omega(0) * \rho_{bl} + \sum_{k=1}^{bl-1} \Omega(k) * p_{bl-k+1}$$

As the system (4') is trivialy solved by putting all $\Omega(s)$ to 0, we add the normalization equation (5).

$$\sum_{s=0}^{\infty} p(s) = 1 \quad (5)$$

Now the system can be solved by any of the well-known methods for solving linear system of equations.

For the system where average rate of input process is time dependent we could not use the same approach. Because the system works in cycles we consider the discrete time. Instead of probabilities $\Omega(s)$ we have $\Omega(s,t)$ where the second variable denotes the cycle counter. With v_i^t we denote the probability that i requests came on input line to the buffer at the time t and is expressed by the following equation

$$v_i^t = \sum_{k=i}^{\infty} p_i^k \cdot p^k \quad (6)$$

A simple calculation gives

$$v_i^t = a(t)^m e^{-a(t)/2} / (2^m m!) \quad (7)$$

With a given buffer length bl we get the recurrence relation for the number of requests in the buffer at the time t (denoted by q_t) if in this moment V_n requests will join the buffer.

$$\begin{aligned} q_{t+1} &= \min \{bl, q_t + V_{t+1} - 1\} \quad q_t \neq 0 \\ q_{t+1} &= \min \{bl, V_{t+1}\} \quad q_t = 0 \end{aligned} \quad (8)$$

So $\Omega(s,t)$ is expressed by

$$\Omega(s,t) = \sum_{i=1}^{s+1} \Omega(i,t-1)\, v_{s-i+1}^t + p(0,t-1)\, v_s^t \quad (9)$$

if s is less than buffer length bl. In that case some of the requests are rejected. The equation is almost the same as (9), we must only replace v_i^t with $\bar{v}_i^t$, where

$$\bar{v}_i^t = 1 - \sum_{j=0}^{i-1} v_j^t \quad (10)$$

is the probability that more than i-1 requests are joining the buffer at the time t.

$$\Omega(s,t) = \sum_{i=1}^{s+1} \Omega(i,t-1)\, v_{s-i+1}^t + p(0,t-1)\, \bar{v}_s^t \quad (11)$$

We start with the distribution

$$p(0,0) = 1, \quad p(s,0) = 0$$

$$s = 1,2,\ldots,bl$$

and repeatedly calculate the probabilities.

The same approach we could use in the case where average input rate is time constant. But as equations (9) and (10) can not be simplified, the calculation with this approach is time consuming.

IV. Switches with two input and two output links

Switches with two input and two output links we find in stages form 2 to $\log_2 N$. Two input links are connected to two switches of the previous stage and the output links are connected to two switches of the next stage.

The input process now depends on the output process of the previous stage. If the buffers that are connected to input links of the switch are not empty, we get a request out of them. As we have only two input links, we can not get more than two requests in one cycle. The output process is just the same as described in section III.

Again we first analyze the stationary case with constant average rate. As we could see from the previous section and as is shown in this, probabilities that buffers are not empty (in this case we get a request) are not changing with time. Let Π_1 and Π_2 be the

probabilities that the buffers from where input links come are empty. Then we have the following probabilities on the number of incoming requests

$$\begin{aligned} p^0 &= \Pi_1 * \Pi_2 \\ p^1 &= \Pi_1 (1 - \Pi_2) + \Pi_2 (1 - \Pi_1) \\ p^2 &= (1 - \Pi_1)(1 - \Pi_2) \\ p^k &= 0 \qquad k = 3, 4, \ldots \end{aligned} \tag{12}$$

As the path from the processor to the memory module is completely random, requests join each of the two buffers with equal probability. So we get

$$\begin{aligned} p_0^0 &= 1 \\ p_0^1 = p_1^1 &= p_1^2 = 0.5 \\ p_0^2 = p_2^2 &= 0.25 \end{aligned} \tag{13}$$

For the probability of m requests coming into buffer in one cycle we get

$$\rho_m = \sum_{k=m}^{2} p^k p_m^k \qquad m = 0,1,2 \tag{14}$$

The balance diagram for this type of switch is shown in Fig. 4.

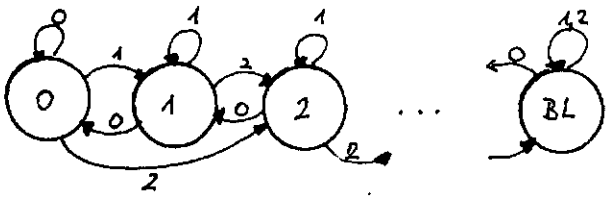

Fig. 4

The balance equatations which we obtain from the diagram are

$$\begin{aligned} \Omega(0)\ (\rho_1 + \rho_2) &= \Omega(1)\ \rho_0 \\ (\Omega(0) + \Omega(1))\ \rho_2 &= \Omega(2)\ \rho_0 \\ \Omega(s)\ \rho_2 &= \Omega(s+1)\ \rho_0 \qquad s = 2, 3, \ldots, bl - 1 \end{aligned} \tag{15}$$

Now the system is already linear. We just have to add the normalization equation (5) and solve it.

For the time dependent system we use a similar procedure as in the previous section. Consider that we analyze the stage i. Then we have

$$\Omega(0,t) = 1, \quad \Omega(s,t) = 0, \ s = 1, \ldots, bl, \quad t = 0, \ldots, i-1 \tag{16}$$

After i cycles the first request can come to the switch on the stage i. The number of requests in the buffer may grow maximaly by a request per cycle, because of maximaly two incoming requests and one outgoing request (except in the case where at the previous moment the buffer was empty. But we only have a jump from 0 to 2 and after that the length grows maximaly by one). So we get the following system

$$\begin{aligned} \Omega(0,t) &= [\Omega(0,t-1) + \Omega(1,t-1)]\ \rho_0 \\ &\quad \Omega(2,t-1)\ \rho_0 \\ \Omega(2,t) &= [\Omega(0,t-1) + \Omega(1,t-1)]\ \rho_2 + \\ &\quad \Omega(2,t-1)\ \rho_1 + \Omega(3,t-1)\ \rho_0 \\ \Omega(s,t) &= \sum_{k=s-1}^{s+1} \Omega(s-1,t-1)\rho_{s+1-k} \qquad s = 3, \ldots, bl - 1 \\ \Omega(bl,t) &= \Omega(bl,t-1)\ (\rho_1 + \rho_2) + \\ &\quad \Omega(bl-1,t-1)\ \rho_2 \end{aligned} \tag{17}$$

Equatations (17) are the same in the case where average input rate is time dependent and where is not.

V. Switches with two input and one output link

This kind of switches we find at the last stage. The input links come from the previous stage and the output links are connected to memory modules. From each switch we can reach only one memory module. The situation is almost the same as described in the section IV. The difference is only that we have just one buffer so all requests go in the same buffer. So we have to change the coefficients ρ_m defined by (12) and (14).

VI. Summary and conclusion

In the Table 1 probabilities on number of requests in the buffer for one switch on the first stage with different average input rates are given.

	average rate			
s	0.5	1	1.5	1.9
0	0.75000	0.50000	0.25085	0.07687
1	0.21302	0.32436	0.28020	0.12189
2	0.03277	0.12260	0.19489	0.12635
3	0.00385	0.03779	0.11706	0.11699
4	0.00037	0.01091	0.06787	0.10594
5	0.00004	0.00311	0.03915	0.09569
6	0.00000	0.00088	0.02258	0.08643
7	0.00000	0.00025	0.01302	0.07807
8	0.00000	0.00007	0.00751	0.07052
9	0.00000	0.00002	0.00433	0.06370
10	0.00000	0.00001	0.00250	0.05753

Table 1

	average rate			
s	0.5	1	1.5	1.9
0	0.75000	0.50000	0.25086	0.08057
1	0.22959	0.38889	0.39047	0.19734
2	0.01999	0.09876	0.23004	0.20422
3	0.00041	0.01097	0.08251	0.15007
4	0.00001	0.00122	0.08251	0.11028
5	0.00000	0.00014	0.02960	0.08104
6	0.00000	0.00002	0.01062	0.05955
7	0.00000	0.00000	0.00381	0.04376
8	0.00000	0.00000	0.00137	0.03216
9	0.00000	0.00000	0.00049	0.02363
10	0.00000	0.00000	0.00006	0.01737

Table 2

In the Table 3 the probabilities for the switches on the last stage are given. All requests are generated with equal average input rate.

	average rate			
s	0.5	1	1.5	1.9
0	0.50000	0.02500	0.00000	0.00000
1	0.38889	0.07500	0.00000	0.00000
2	0.09877	0.10000	0.00000	0.00000
3	0.01097	0.10000	0.00000	0.00000
4	0.00122	0.10000	0.00000	0.00000
5	0.00014	0.10000	0.00000	0.00000
6	0.00002	0.10000	0.00014	0.00000
7	0.00000	0.10000	0.00125	0.00000
8	0.00000	0.10000	0.01117	0.00008
9	0.00000	0.10000	0.09958	0.00897
10	0.00000	0.10000	0.88784	0.99094

Table 3

As we can see from the tables it is obvious that as the input average rates grow to the maximum output rate, which is 2, the buffers are more and more full, so also grows the number of rejected requests.

Acknowledgement

This work was partly supported by Iskra Delta Computers.

References

[1] Rettberg R., Thomas R., Contention is no obstacle to shared-memory multiprocessing, CACM, 29(1986), 12.

[2] Patel J.H., Performance Of Processor-Memory Interconnection for Multiprocessors, IEEE Trans. on Comp., Vol. C-30 (1981), 10.

[3]Kruskal C.P., Snir M., The Performance Of Multistage Interconnection Networks for Multiprocessors, IEEE Trans. on Comp., Vol. C-32 (1983), 12.

[4]Dias D.M., Jump J.R., Analysis and Simulation of Buffered Delta Networks, IEEE Trans. on Comp., Vol. C-30 (1981), 4.

[5]Thanawastien S., Nelson V.P., Interference Analysis of Shuffle/Exchange Networks, IEEE Trans. on Comp., Vol. C- 30(1981), 8.

[6]Brajak P., Designing a reconfigurable intelegent memory module (RIMM) for Performance Enhancement to Large Scale, General Purpose Parallel Processor, Informatica 11(1987), 1.

[7]Gelenbe E., Mitrani I. Analysis and Synthesis of Computer Systems, Academic Press, 1980 London.