Hot spot contention and message combining in the Simple Serial Synchronized Multistage Interconnection Network

Toshihiro Hanawa Takashi Fujiwara Hideharu Amano
Department of Computer Science, Keio University
3-14-1, Hiyoshi Yokohama 223 Japan
{hanawa, fujiwara, hunga}@aa.cs.keio.ac.jp

Abstract

Simple Serial Synchronized (SSS) Multistage Interconnection Network (MIN) is a novel MIN architecture for connecting processors and memory modules in multiprocessors. Synchronized bit-serial communication simplifies the structure/control, and permits the bit-serial message combining mechanism.

From the theoretical analysis and empirical results, it appears that the influence of the hot spot contention is not catastrophic in the small size SSS-MIN (16 $\times$ 16) when some parallel programs from SPLASH benchmark run. In this situation, the effect of the message combining is small.

However, in a large system (256 processors), the pass through ratio improves by 11% when a parallel program with relatively large access ratio (r=0.4) runs. Since the additional hardware for the message combining is only 20% in the SSS-MIN, the message combining is useful in large systems.

1. Introduction

Multistage Interconnection Network(MIN)[2] is often used for the connection between processors and memory modules on medium-scale multiprocessors. Most of them are blocking networks like the omega network[2], and packets are transferred between switching elements in the store-and-forward or worm-hole manner through bit-parallel (8 – 64 bits) lines. In blocking networks, packets may collide with each other in the network even if their destinations are different. In this case, one of the conflicting packets is stored in the packet buffer.

However, the implementation of large scale MINs has been a cumbersome job because of their complicated structure and pin-limitation problem. Because of the store-and-forward operation, a buffer storage and control mechanism are required in each switching element. The pin-limitation problem caused by bit-parallel communication makes high density implementation difficult. Since a large number of chips requires a large area for implementation, the clock frequency is limited at relatively low level (10-50MHz).

As a solution to these problems, we proposed a novel MIN architecture called Simple Serial Synchronized (SSS)-MIN[1] and network topologies for the MIN[5]. Two different LSI chips for the SSS-MIN were implemented [10] [6], and a prototype multiprocessor called SNAIL (Sss-Network Architecture ImpLementation) is available [10].

In the SSS-MIN, packets are transferred in the serial and synchronized manner. Synchronized bit-serial communication simplifies the structure/control, and also solves the pin-limitation problem. The simple structure permits the use of a high frequency clock and a high throughput network topologies[5]. To enhance the performance, *pipelined circuit switching* and *stage hopping* are introduced.

One of the important characteristics of the SSS-MIN is that it permits the message combining with a small increase of hardware by the best use of the synchronous serial packet transfer. This technique is called the *bit serial message combining*. Unlike the message combining mechanism in the traditional MIN, it only requires about 20% increase of the hardware, and this function is available on the SNAIL.

In this paper, the effect of the bit-serial message combining in the SSS-MIN is evaluated with theoretical model, simulation, and the real multiprocessor.

2. The SSS-MIN

2.1. Basic operation of the SSS-MIN

The basic operation of the SSS-MIN is illustrated in Figure 1. Like the Asynchronous Transfer Mode (ATM)-based packet switching systems for telecommunication, all packets are inserted into the SSS-MIN serially synchronized with a common frame clock from input buffers prepared between processors and the MIN. Since each switching element stores only one bit (or a few bits) of the packet, the SSS-MIN behaves like a set of shift registers with the switching capability. After the delay for passing through all stages, the head of packets come out at the outlet of the MIN.

When a conflict occurs, one of the conflicting packets is routed to the incorrect direction since the SSS-MIN provides no packet buffer in each switching element. The conflict bit in the routing tag is set when the conflicting packet is routed to the incorrect direction. Such a packet is treated as a dead packet, and never interferes with other packets.

When the top of an address packet reaches the outlet of the MIN, all switching elements are set to be straight or exchange, and a path is formed from the input to the output of the MIN. Here, this path is called the *trace*.

When the trace is set, it can be judged whether the packet is correctly routed or not by referring the conflict bit at the outlet of the MIN. An acknowledge signal (ACK: ACKnowledge or NAK: Not AcKnowledge) is returned to the processor which issues the access request through this trace without being stored at each switching element, but only with the delay of wires and multiplexors. If the NAK signal is returned, the packet is inserted again in the next frame from the input buffer.

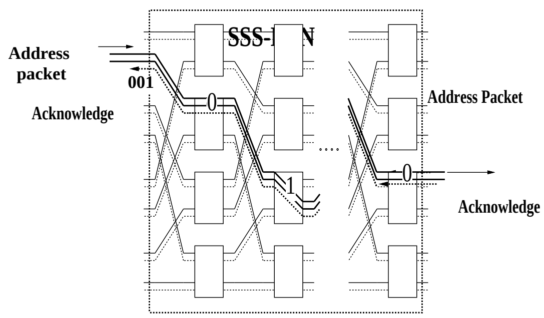

Figure 1. Structure of the SSS-MIN

Once the trace is established, the rest of the address packet and the data packet are not required to be stored in each switching element. Thus, the flip-flop can be omitted in some stages, and packets can hop several stages. This technique is called the *stage hopping*. The latency is much reduced by this mechanism.

2.2. Pipelined circuit switching

The trace is utilized not only for the acknowledge signal but also for the transfer of data packets. The address transfer, acknowledge signal, and data transfer can be performed in the overlapped manner. Figure 2 shows a timing diagram of these transfers. In frame i, an address packet is transferred. When the head of the address packet arrives at the outlet of the MIN, the trace is established and the acknowledge signal is transferred immediately. If the ACK signal is returned, the data packet is also transferred using the same trace in frame $i + 1$. If the pass through time of the MIN is large, the pipeline can be formed for these transfers. We call this operation overlapped or pipelined circuit switching.

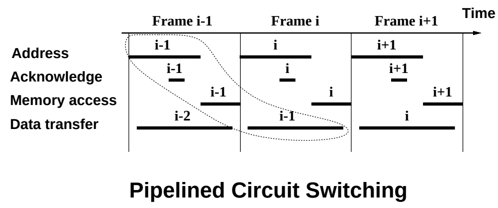

Figure 2. Timing diagram of packets/signal transfers in the SSS-MIN

In practice, the depth of the pipeline is two since the pass through time of a packet is reduced by the stage hopping mechanism. By sharing the trace, the structure of the SSS-MIN becomes simpler than the conventional method which requires two independent MINs for bi-directional transfer.

2.3. Bit serial message combining

The message combining[4] is a technique which merges identical references into a composite combined reference at each stage of the network. Though a cost effective switch architecture with combining has been researched and implemented for conventional MIN[13], it still requires a large amount of additional hardware.

In the SSS-MIN, the message combining is easily realized by a bit-by-bit comparison during the address packet transfer. If the addresses in both packets are equal, the switching element is set to be the backward broadcast mode as shown in Figure 3. After these address packets pass through, the tree-like trace remains in switching elements, and the returning acknowledge and data packets are multicast through the tree. This technique is also useful for the write-read combining.

To combine Test&Set operations, a simple additional hardware is required. The similar tree-like trace for data transfer is formed .However, the data is not multicast but one of the returned values is forced to be '1' in each element. At the same time, the Memory Interface Unit (MIU) which is provided between the MIN and memory modules writes all-1 into the target address of the memory module. As a result, only one processor can read out the data in the target address, and other processors get all-1. This mechanism is called the *bit-serial message combining*.

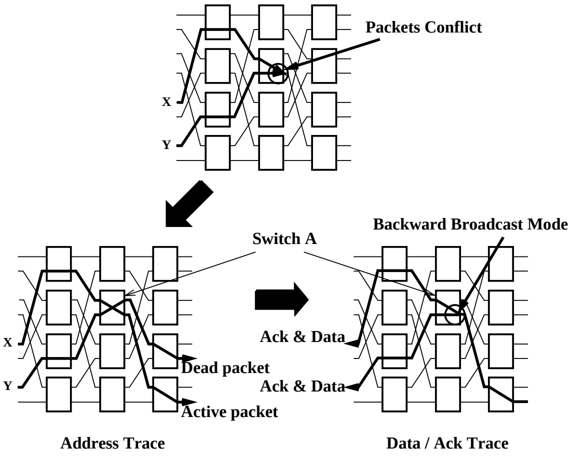

Figure 3. Message combining

3. Analysis model for the hot spot and message combining

Here, probabilistic models for analyzing the throughput of the SSS-MIN under the traffic with the hot spot are proposed.

For the conventional MIN, a lot of analysis results with simulation studies and theoretical models have been published [11][9][7][8][13]. The simplest equation for the worst case throughput R of conventional MINs was given by Phister[11]:

$$R = \frac{1}{1 - h + hN} \tag{1}$$

where N is the number of inputs (the number of processors) and h is the ratio of packets for the hot spot to the total number of packets (hot spot ratio). This equation represents the case that packets to the hot spot completely occupy all the buffers of switching elements. In this case, even if h is only 1%, the bandwidth is severely degraded to less than 10%. This comes from the assumption that all packets are blocked with the hot buffers.

Since conflicting packets never block other packets unlike conventional MINs, some of regular packets are not blocked in the SSS-MIN. However even in the SSS-MIN, regular packets to memory modules near the hot spot are severely blocked when packets to the hot spot increases. Since the probability of blocking depends on the location between the element and the hot spot, elements in the same stage are divided into some groups in the theoretical analysis.

3.1. Analysis model under the uniform traffic

First, the state of each switching element under uniform traffic is analyzed. In this model, all packets are sent to every memory module with the same probability.

Here, assume that a packet is inserted into an input of a switching element at a probability r. Then, the probability of the packet conflict is $0.25r^2$ because the conflict occurs if and only if packets are inserted from both inputs, and their destinations are the same. In the SSS-MIN, when conflict occurs, a misrouted packet becomes a “dead packet”, and never interferes with other packets in the later stages. These packets can be treated as being disappeared.

Therefore, when a packet is inserted into an input of a switching element in the i-th stage at a probability r_i, the probability for the next stage r_{i+1} is represented as follows:

$$r_{i+1} = r_i - 0.25{r_i}^2 \equiv X(r_i) \; . \tag{2}$$

The pass-through ratio (The probability that a packet arrives at the output) of a single banyan network with n stages is represented as follows:

$$r_n = X^n(r_0) \tag{3}$$

where r_0 represents the probability of the input packet for the MIN(that is, the traffic load of the network).

3.2. Analysis model with a hot spot

Here, we assume that a packet is inserted into an input of a switching element in the i-th stage at a probability r_i, and the packet is routed to the hot spot at a probability $r_i \times h_i$ (h_i is the hot spot ratio in the i-th stage). The output port of the element which is on the way to the hot spot is marked $O+$ while the other output is marked $O-$. Two input ports of a switching element (In1, In2) take one of the following three cases:

1. a packet for the hot spot arrives (H),
2. a regular packet arrives (E), and
3. no packets arrives (-).

The probability of these cases and the expected number of the packets at two outputs of the element are shown in Table 1. $O+(H)$ represents the expected number of packets which are sent for the hot spot ($O+$), and $O+(E)$ is the expected number of packets sent for the other destination. Since packets may be routed from both inputs, $O+(H)$, $O+(E)$, and $O-$ are less than 1.0 for each, and the sum of them is less than 2.0.

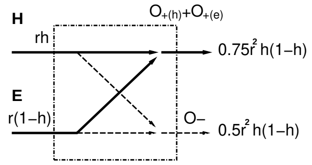

Figure 4. Model of an element

From this table, the probability of the packet arrival at the output (s_{O+}) of an element on the way to hot spot can be written:

$$s_{O+} = 2r_i h_i - 0.5r_i^2 h_i - 0.5r_i^2 h_i^2. \quad (4)$$

The probability of the packet arrival at the output of an element which is not on the way to hot spot (e_{O+}) is given by:

$$e_{O+} = r_i(1-h_i)(1-0.25r_i - 0.25r_i h_i). \quad (5)$$

The packet existing probabilities r_{i+1} and the hot spot ratio h_{i+1} at the input of the element on the way to hot spot in the next $(i+1)$ stage is represented as follows:

$$h_{i+1} = s_{O+}/(s_{O+} + e_{O+}) \equiv Zh(h_i, r_i) \quad (6)$$

$$r_{i+1} = s_{O+} + e_{O+} \equiv Zr(h_i, r_i). \quad (7)$$

On the other hand, at the input of the element which is not on the way to hot spot, the packet existing probability r_{i+1} is represented as follows:

$$r_{i+1} = r_i(1-h_i)(1-0.25r_i + 0.25r_i h_i) \equiv W(h_i, r_i). \quad (8)$$

Here, let the label of memory module including the hot spot be $m(m_{n-1}m_{n-2}\dots m_1 m_0)$. Since the packet arriving at the hot memory module passes through output port marked $O(+)$ in every stage, the probability that a packet arrives at the memory module can be written as:

$$\begin{aligned} & 2r_{n-1}h_{n-1} - r_{n-1}^2 h_{n-1}^2 \\ \equiv\ & Zr_{n-1}^{+}(h_{n-1}, r_{n-1}) \\ =\ & Zr_{n-1}^{+}(Zh^{n-1}(h_0, r_0), Zr^{n-1}(h_0, r_0)) \end{aligned} \quad (9)$$

(Memory 0 in Figure 5).

The arrival probability of a packet at the neighboring module of the hot memory $m_{n-1}m_{n-2}\dots\overline{m_0}$ is as follows:

$$\begin{aligned} & 2r_{n-1}(1-h_{n-1}) - r_{n-1}^2(1-h_{n-1})^2 \\ \equiv\ & Zr_{n-1}^{-}(h_{n-1}, r_{n-1}) \\ =\ & Zr_{n-1}^{-}(Zh^{n-1}(h_0, r_0), Zr^{n-1}(h_0, r_0)) \end{aligned} \quad (10)$$

Table 1. Packet arrival probability at the output of an element(without combining)

		Normal stage			Final stage (include hot spot)		Probability
In1	In2	$O+$ (H)	$O+$ (E)	$O-$	$O+$	$O-$	the event
–	–	0	0	0	0	0	$(1-r)^2$
–	H	1	0	0	1	0	$rh(1-r)$
H	–	1	0	0	1	0	$rh(1-r)$
–	E	0	0.5	0.5	0	1	r(1-h)(1-r)
E	–	0	0.5	0.5	0	1	r(1-h)(1-r)
H	E	0.5	0.25	0.5	1	1	$r^2(1-h)h$
E	H	0.5	0.25	0.5	1	1	$r^2(1-h)h$
E	E	0	0.75	0.75	0	1	$r^2(1-h)^2$
H	H	1	0	0	1	0	r^2h^2

(Memory 1 in Figure 5).

Similarly, the arrival probability of a packet at a memory module $m_{n-1}m_{n-2}\dots\overline{m_1}*$ ($*$ represents 0 or 1) is as follows:

$$X(W(Zh^{n-2}(h_0, r_0), Zr^{n-2}(h_0, r_0)) \quad (11)$$

(Memory 2 and 3 in Figure 5).

In general, an arrival packet at a memory module $m_{n-1}m_{n-2}\dots\overline{m_k} * \dots *$ is routed through output port $O+$ until the $(n-k-2)$-th stage, and routed through $O-$ from the $(n-k-1)$-th stage. Therefore, the packet arrival probability is given by:

$$X^k(W(Zh^{n-k-1}(h_0, r_0), Zr^{n-k-1}(h_0, r_0))) \quad (12)$$

where $Zh^0(h_0, r_0) \equiv h_0, Zr^0(h_0, r_0) \equiv r_0$.

The total pass through ratio of MIN is the sum of packet arrival probabilities for all memory modules, as follows:

$$\begin{aligned} R \ =\ & Zr_{n-1}^{+}(Zh^{n-1}(h_0, r_0), Zr^{n-1}(h_0, r_0)) \\ +\ & Zr_{n-1}^{-}(Zh^{n-1}(h_0, r_0), Zr^{n-1}(h_0, r_0)) \\ +\ & \sum_{k=1}^{n-1} 2^k \cdot X^k(W(Zh^{n-k-1}(h_0, r_0), Zr^{n-k-1}(h_0, r_0))), \\ & \text{where } Zh^0(h_0, r_0) \equiv h_0, Zr^0(h_0, r_0) \equiv r_0. \end{aligned} \quad (13)$$

3.3. Analysis model with message combining mechanism

In the SSS-MIN, packets combined by the bit serial message combining mechanism do not prevent the other packets

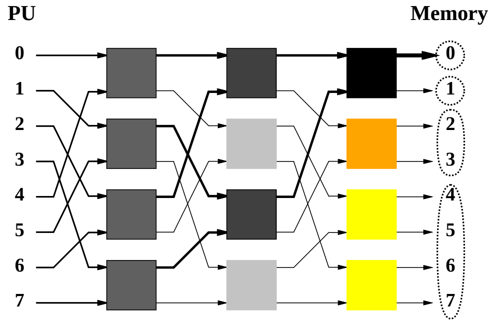

Figure 5. Analysis model under hot spot traffic

like dead packets(see Figure 3). Thus, the analysis model without message combining mechanism is useful in the case of message combining.

If both inputs of a switching element receive packets for the hot spot, these packets are combined. Therefore, the pass through ratio is represented as the sum of the results without combining and the probability of the message combining in every switching element.

As a result, an increase of the pass through ratio by the message combining is shown as follows:

$$\sum_{k=0}^{n-1} 2^k \cdot (Zh^{n-k-1}(h_0, r_0))^2 (Zr^{n-k-1}(h_0, r_0))^2. \quad (14)$$

Although a single banyan network is assumed here, this analysis model can be extended to high bandwidth MINs proposed for the SSS-MIN[5].

4. Analysis with a real multiprocessor: SNAIL

Here, the hot spot contention and message combining are analyzed with a real multiprocessor SNAIL. Since this machine consists of only 16 processors, the effect of the hot spot contention is not severe in usual parallel applications. However, through this analysis, parameters of the theoretical model can be obtained for evaluating larger systems.

4.1. Multiprocessor prototype SNAIL

SNAIL is a multiprocessor prototype which is developed for the performance evaluation of the SSS-MIN with practical application programs. Especially, SNAIL is the first multiprocessor system which implemented the message combining mechanism with more than 10 processors.

As shown in Figure 6, 16 processors each with a local memory are connected to the 16 shared memory modules with the SSS-MIN. One of processors has the interface of Ethernet, and it is connected the host workstation. The performance monitor and debugger system which works independently from SNAIL are offered , and they are connected to the input and the output of the MIN. All system modules share a single system clock (50MHz) and a frame clock (50/16=3.125MHz) for the synchronous operation.

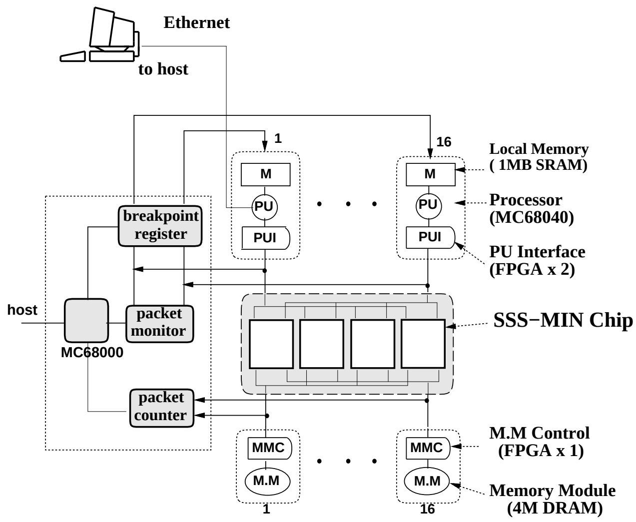

Figure 6. Diagram of a multiprocessor SNAIL

In multiprocessors using conventional MINs, it is almost impossible to analyze traffic for the shared memory by a hardware monitor since too many lines must be monitored. However, in the SSS-MIN, packets are transferred through a small number of wires, and it is possible to analyze the traffic by monitoring them.

The hardware detail of SNAIL is as follows:

- Processing Unit

  Motorola's MC68040 with 1Mbyte local memory is used. The clock used in the SSS-MIN (50MHz) is also used as a system clock to all processors.

  The PIU (Processor-MIN Interface Unit) manages parallel/serial conversion between processor buses and the SSS-MIN chips. 2-word address packet buffer and 2-word prefetch buffer are provided inside the PIU.

- Network

  An omega[3] network with 16-input/16-output is implemented in a single chip[1], and a high throughput network TBSF (Tandem Banyan Switching

[1]This chip was designed and fabricated with the cooperation of Kawasaki Steel Co. Ltd. Japan.

Fabrics)[14][5] can be simply realized by the tandem connection of them. The number of cells for the message combining is only about 20% of total number of required cells.

- Memory modules

  Sixteen 4Mbyte-DRAM memory modules form a 64Mbyte-shared memory system which is interleaved in the line size of the on-chip cache (4 words).

  The MIU (Memory Interface Unit) manages parallel/serial conversion and other controls of memory modules including block transfer, and memory refresh. It also manages two synchronization operations, Test&Set and Fetch&Dec for every memory word.

The access time of memory systems without conflict is shown in Table 2.

Table 2. Memory access time

Frame period	320nsec
32bit data read/synchronization	360nsec
1line block transfer	800nsec

4.2. Traffic analysis with SNAIL

Here, hot spot ratio and traffic load of practical applications are evaluated with SNAIL. Four practical applications from SPLASH benchmark suite[12] (MP3D with 500 particles in 50 step, Water with 64 molecules in 2 step, Cholesky with 3466 dimension, and Barnes-Hut with 1024 bodies in 4 step) are used.

First of all, the speedup ratio of each application is shown in Figure 7. Since the number of processors is not so large in SNAIL, almost linear speedup is achieved except with Cholesky that requires relatively frequent access of the shared memory.

Figure 8 shows the network utilization for an input of MIN (r in the analysis model). On SNAIL $r = \frac{U}{A}$, where U is the number of frames in which a processor issues an access request and A is the total number of frames. From the result of analysis, r ranges from 10% to 35% and is almost independent from the number of processors. Considering the speedup ratio shown in Figure 7, congestion of the network slightly degrades the performance in Cholesky with maximum network utilization.

Figure 9 shows the combining ratio. In the SSS-MIN, since all packets to the hot spot in the same frame is completely combined, the combining ratio corresponds to the hot spot ratio h in the theoretical model. Because of the problem caused by the monitoring hardware, some applications cannot be analyzed with a large number of processors. From these results, the combining ratio is not so large (less than 4%) in these applications except with Cholesky but increases with the number of processors.

We also measured the performance improvement with the message combining in every application. However, only a little (1%) performance improvement can be observed even with 16 processors.

Figure 10 shows the pass through ratio with the theoretical model based on the parameters from SNAIL (r=10-40%, h=5-15%). These results show that the pass through ratio is not improved based on these parameters. Even with parameters from Cholesky(r=0.35, h=10%), only 3-4% improvement is observed from the theoretical model. This analysis result well matches the fact that the execution time improves only by 1% in SNAIL.

Especially in Cholesky, all processors often use the barrier synchronization on a variable in the shared memory. Most of message combining occurs for this variable. However, since processors must wait the result from the latest processor in barrier synchronization, message combining occurs just in the busy waiting and not so contribute to the performance improvement.

Thus, it appears that the combining mechanism in SNAIL does not work effectively in the current system size and applications. Next, the performance of larger systems are evaluated with the proposed theoretical model based on parameters analyzed in SNAIL.

4.3. Analysis of larger systems

According to the evaluation of SNAIL, traffic load is 35% in Cholesky, and 25% in Water. Here, it is assumed that the access load is 30%.

Figure 11 shows the relation between network size and pass through ratio when the access rate r is 30%. From this figure, the pass through ratio of the large scale MIN without message combining is severely degraded if the hot spot ratio is large.

However, the combining mechanism can save the degradation in a certain degree. Pass through ratio improves by 11% with 256 processors when the hot spot ratio is 15%. Note that the pass through ratio of the MIN with message combining mechanism under hot spot contention is larger than that of the MIN under the uniform traffic (h=0). Since all packets for the hot spot are combined, the traffic congestion of packets for non-hot spot is relaxed. Therefore, the larger the number of packets for the hot spot is, the better the pass through ratio of the MIN with message combining becomes.

Figure 12 shows the pass through ratio of a system with 256 processors with large r. From this figure, the pass

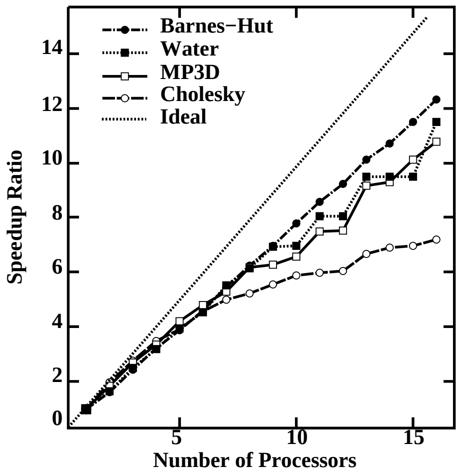

Figure 7. Speedup ratio

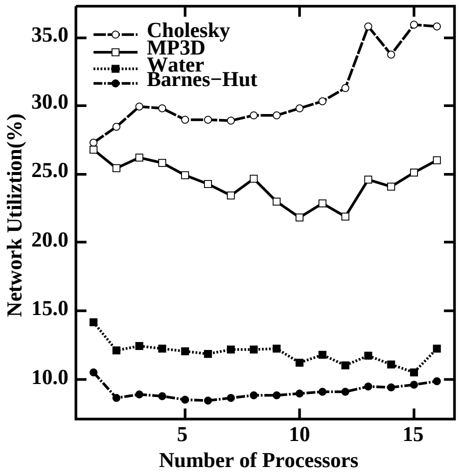

Figure 8. Network Utilization

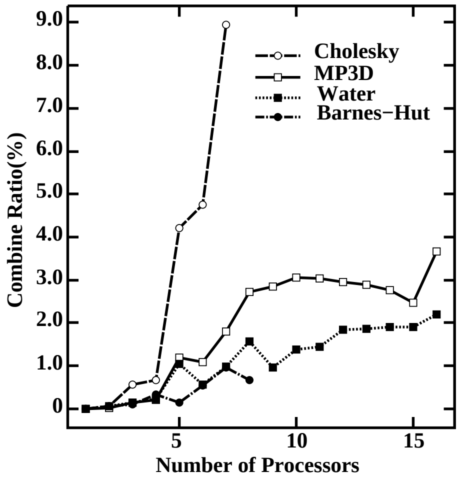

Figure 9. Combining ratio in whole packets

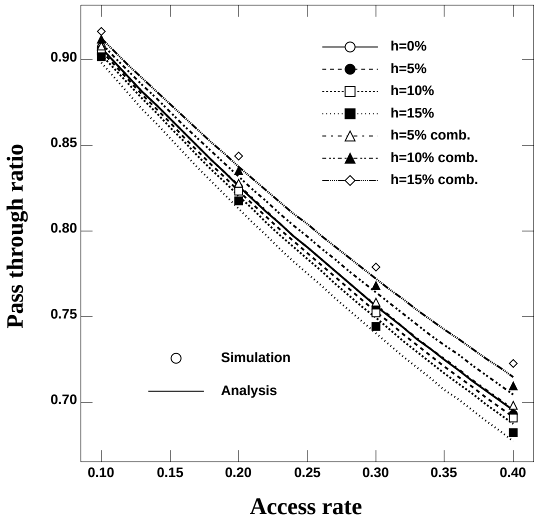

Figure 10. Relation between load and pass through ratio(16×16)

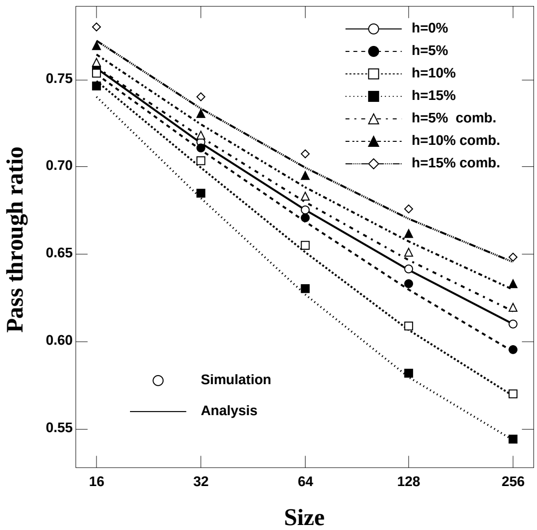

Figure 11. Relation between network size and pass through ratio(Access rate $= 30\%$)

through ratio is improved over 10% in any r when hot spot ratio is 15%.

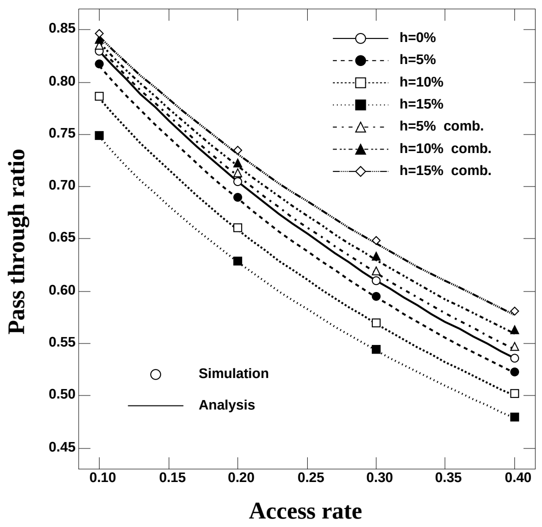

Figure 12. Relation between load and pass through ratio(256×256**)**

5. Conclusion

From the theoretical analysis and empirical results, it appears that the influence of the hot spot contention is not catastrophic in the small size SSS-MIN (16×16) when parallel programs from SPLASH benchmark run. In this situation, the effect of the message combining is small.

However, in a large system (256 processors), the pass through ratio improves by 11% when a parallel program with relatively large access ratio ($r = 0.4$) runs. Since the additional hardware for the message combining is only 20% in the SSS-MIN, the message combining is useful in large systems.

The simple and synchronized control of the SSS-MIN makes the software scheduling easy. If packet generation of each processor is well scheduled, the message combining mechanism is used for multicast/broadcast of data. This technique will be efficient for most applications with frequent data multicast. The co-operation of the message combining mechanism and software static scheduling is our future work.

References

[1] H. Amano, L. Zhou, and K. Gaye. SSS(Simple Serial Synchronized)-MIN: a novel multistage interconnection architecture for multiprocessors. In *Proc. of the IFIP 12th World Computer Congress, Vol.I*, pages 571–577, Sep. 1992.

[2] G. Broomell and J. Heath. Classification Categories and Historical Development of Circuit Switching Topologies. In *ACM Computing Surveys*, June 1983.

[3] D.H.Lawrie. Access and Alignment of Data in an Array Processor. *IEEE Trans. on Comput.*, c-24(12), dec 1975.

[4] A. Gottlieb, R. Grishman, C. P. Kruskal, K. Mcauliffe, L. Rudolf, and M. Snir. The NYU Ultracomputer - Designing an MIMD Shared Memory Parallel Computer. *IEEE Trans.on Comput.*, c-32(2), feb 1983.

[5] T. Hanawa, H. Amano, and Y. Fujikawa. Multistage Interconnection Networks with multiple outlets. In *Proc. of the International Conference on Parallel Processing*, volume I, Aug. 1994.

[6] T. Kamei, M. Sasahara, and H. Amano. An LSI implimentation of the Simple Serial Synchronized Multistage Interconnection Network. In *Proc. of the Synthesis and System Integration of Mixed Technologies(SASIMI'95)*, aug 1995.

[7] B. Kang, G. Lee, and R. Kain. Performance of multistage combining networks. In *Proc. of the International Conference on Parallel Processing*, volume I, aug 1991.

[8] L. Kurian and M. Thazhuthaveetil. Effect of hot spots on multiprocessor systems using circuit switched interconnection networks. In *Proc. of the International Conference on Parallel Processing*, volume I, pages 554–557, aug 1991.

[9] K. Lee and W. Hegazy. The Extra stage Gamma Network. In *International Symposium on Computer Architecture conference proceedings*, jun 1986.

[10] M.Sasahara et al. SNAIL: A Multiprocessor Based on the Simple Serial Synchronized Multistage Interconnection Network Architecture. In *Proc. of the 1994 International Conference on Parallel Processing, Vol.I*, pages I–117–I–120, Aug. 1994.

[11] G. Phister and V. Norton. Hot spot contention and combining in multistage interconnection networks. *IEEE Trans. on Comput.*, c-34(10):764–771, 1985.

[12] J. Singh, W. Weber, and A. Gupta. SPLASH: Stanford Parallel Applications for Shared-Memory. Technical report, Computer System Laboratory, Stanford University, 1992.

[13] S.R.Dickey and R.Kenner. Combining Switches for the NYU Ultracomputer. In *Proc on IEEE Frontiers to Massively Parallel Processing*, pages 521–523, nov 1992.

[14] F. Tobagi. Fast Packet Switch Architectures For Broadband Integrated Services Digital Networks. *Proceedings of the IEEE*, 78(1), 1990.