MOTIVATION TO OPTICAL MULTISTAGE INTERCONNECTION NETWORKS

[1]SindhuLakshmi Manchikanti, [2]Gayathri Korrapati

[1]Department of Computer Science, Sree Vidyanikethan Institute of Management,
Tirupathi, A.Rangampet-517102, Andhra Pradesh, India

[2]Department of Computer Science, Sree Vidyanikethan Institute of Management,
Tirupathi, A.Rangampet-517102, Andhra Pradesh, India

Abstract

The advancement in the optical technology have drawn the idea of optical implementation of MINs as an important optical switching topology to meet the ever increasing demands of high performance computing communication applications for high channel bandwidth and low communication latency. However, dealing with electro-optic switches instead of electronic switches held its own challenges introduced by optics itself. Limited by the properties of optical signals, optical MINs (OMINs) introduce optical crosstalk, as a result of coupling two signals within each switching element. Therefore, it is not possible to route more than one message simultaneously, without optical crosstalk, over a switching element in an OMIN. Reducing the effect of optical crosstalk has been a challenging issue considering trade-offs between performance and hardware and software complexity. To solve optical crosstalk, many scheduling algorithms have been proposed for routing in OMIN based on a solution called the time domain approach.

Keywords, *of the abstract: Omega network, Multistage Interconnection networks, time domain approach, Scheduling algorithms*

1. Introduction

Advances in electro-optic technologies have made optical communication a promising networking alternative to meet the ever increasing demands of high-performance computing communication applications for high channel bandwidth, low communication latency and parallel processing as well. Optical Multistage Interconnection Network (OMIN) is popular in switching and communication applications and has been studied extensively as an important interconnecting scheme for communication and parallel computing systems. The OMIN is frequently proposed as connections in multiprocessor systems or in high bandwidth network switches [1]. A major problem in OMIN is optical crosstalk. It is caused by coupling two signals within a Switching Element (SE). Crosstalk problem in a switch is the most prominent factor, which reduces the signal-to-noise ratio and restricts the size of a network. Various methods to decrease the undesirable effect of crosstalk have been proposed, that apply the concept of dilation in either the space, time or wavelength domains. With the space domain approach, additional SE(s) and links are used to certify that at most only one input and one output of every SE will be active at any given time. With the time domain approach, two connections will be activated at different time slots if they share the same SE in any stage of the network. The last approach, the wavelength domain, different wavelengths are used for routing active connections by ensuring two wavelengths entering an SE to be far apart by routing or using wavelength converters. Whenever the limitation of the network size is reached, the time domain method may be used as a feasible way to trade the maximal bandwidth available to each particular input and output pair for enhanced connectivity. Again, it is useful when future technology let the transmission rate to expand faster than the network size or when the cost of expanding the bandwidth of each connection becomes as "cheap" as the cost of building a network of twice its original size.

2. Omega Network

An Optical Omega Network (OON) topology has altogether N inputs, N outputs and n stages where $n = \log_2 N$. Each stage has $N/2$ SEs with each SE has two inputs and two outputs connected in a certain pattern[2]. The inter-stage connection pattern in an Omega network is of shuffle-exchange connection pattern To connect the source address to the destination address, the address is shifted one bit to the left circularly in each connection such as source to the first stage, one stage to the next stage. For instance, to connect between each stage in an 8 x 8 optical Omega network, each connection is shuffle-exchanged as shown in

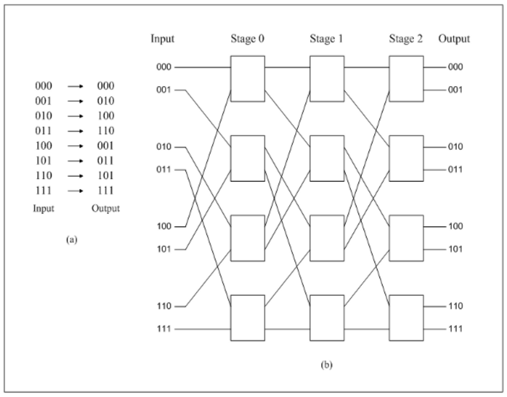


Fig 1. a) Shuffle-Exchange Inter-Stage Connection Pattern, and (b) An 8 x 8 Optical Omega Network.

The shuffle-exchange connections have to be considered when scheduling a permutation for routing in the OON. The inter-stage connection pattern determines the routing mechanism in a network. It also limits the number of messages that can be routed simultaneously in a single time slot or pass, since no two signals are allowed to share an SE at any given time or crosstalk will occur. Figure 1(b)illustrates the general layout of the Omega network topology. OON is topologically equivalent to many other topologies such as the Baseline, Butterfly and Cube networks and. Since many other topologies are equivalent to the Omega network topology, performance results obtained for the Omega network are also applicable to other OMIN topologies.

Suppose an n-bit binary numbers from 0 to $N - 1$ (where $n=\log_2 N$ and N is the network size) is used to label the addresses of N input or output ports from top to bottom of the OON, the shuffle-exchange interconnection connects output port $s_0s_1s_2 \ldots s_{n-1}$ from stage i to the input port $s_1s_2 \ldots s_{n-1}s_0$ of stage $i + 1$, $0 \leq i < n - 1$. Every stage of switches in the OON is preceded by the shuffle-exchange interconnection including the N source inputs connected to the switches of the first stage. The switching connections in each SE can be of either straight or cross connection.

To route a message in an OON, the destination tag which is binary equivalent of the destination address, ($d_{n-1}d_{n-2} \ldots d_1d_0$) is used. The i th bit d_i is used to control the routing at the i th stage counted from the right with $0 \leq i \leq n - 1$. If $d_i = 0$, the input is connected to the upper output. Otherwise, if $d_i = 1$, it is connected to the lower output. In other words, message routing can be achieved simply by relaying messages to either the upper switch output link or the lower output link of the SEs according to the destination address. This unique characteristic of the OON are often referred to as self-routing.

3. Multistage Interconnection Network

Multistage Interconnection Network (MIN) are a class of dynamic interconnection network that connects input devices to output devices through a number of switch stages, each stage consists of a set of SEs arranged in cascaded order, where each switch is a crossbar network[3]. Frequently proposed as interconnection schemes in multiprocessor systems or in high bandwidth network switches, MIN has assumed importance in recent times, because of their cost-effectiveness. While crossbar networks have the advantage of establishing connections between every input port to any free output port, it requires N^2 switches to construct the network where N is the network size. MIN requires only $N(\log_2 N)/2$ switches for the same N..

4. Crosstalk in Optical Omega Network

In the event of optical crosstalk occurrence, a small fraction of the input signal power may be detected at another output disregard of the actual signal injected to the appropriate output port. Consequently, the input signal will be distorted at the output due to loss and crosstalk accumulated along the connection path.

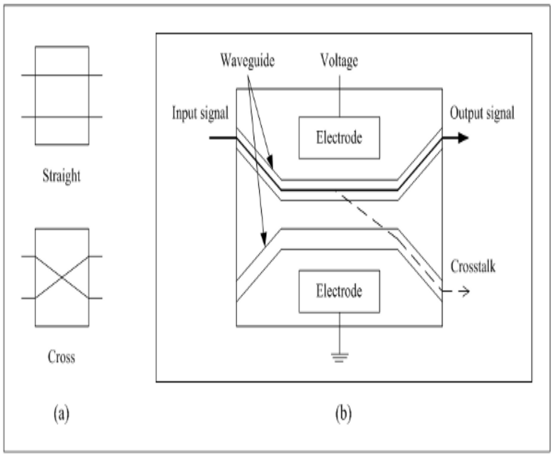


Fig 2. a) Straight or Cross Logic State of a 2x2 SE, and (b) Optical Crosstalk Effect in an Electro-Optic SE.

Because routing in OON make use of both SE configuration shown in Figure 2(a), optical crosstalk has been the major drawback in achieving the most of network performance when routing permutations simultaneously[4]. Therefore, it is not possible to route more than one message simultaneously, without optical crosstalk, over an SE in OON. Reducing the effect of optical crosstalk has been a challenging issue considering trade-offs between performance and hardware and software complexity. To solve optical crosstalk, many scheduling algorithms have been proposed for routing in OMIN based on a solution called the time domain approach, which divides the *N* optical inputs into several groups such that crosstalk-free connections can be established. In this chapter, we propose a solution that can further optimize and improve the performance of message scheduling for routing in OON using the time domain approach.

5. Time domain Approach

In order to avoid crosstalk in OONs, several approaches based on network dilation have been proposed. The three approaches include the the space domain, time domain and wavelength domain dilation. Space domain approach duplicates and combines a MIN to avoid crosstalk within individual SE. Using this approach, an *N* x *N* network is dilated into a network that is essentially equivalent to a *2N* x *2N* network, but only half of the input and output ports used for routing. Based on this approach, a dilated Benes network has been proposed where up to *N* connections can be established without sharing any SE. However, it uses more than double of the number of switches required for the same connectivity. A set of permutation connection is partitioned into several scheduling groups called semi-permutations in such a way that the entries within each group are crosstalk-free and Each group is routed to its corresponding destination independent of the other groups in a different time slot. The main advantage of the time domain approach is that it does not involve additional cost of having more SEs as well as the cost for wavelength conversion as does the space and wavelength domain approaches.

5.1. Time domain approach framework

Because routing messages simultaneously across the OON causes crosstalk, it is important to make sure a permutation is decomposed and scheduled in crosstalk-free order for routing messages. The general framework of the time domain approach consists of two phases including permutation decomposition in the first stage and message scheduling in the second as illustrated in Figure 3.

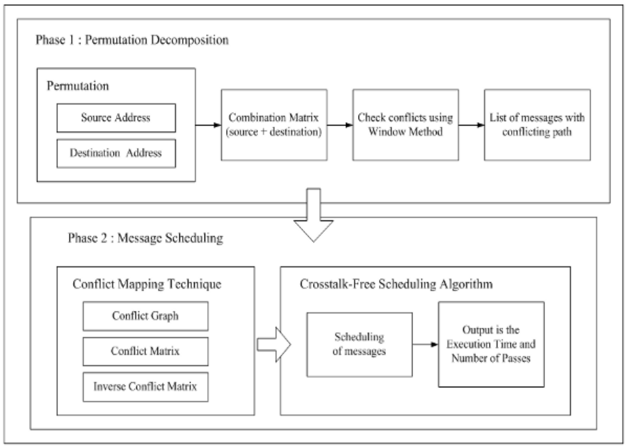

Phase 1 : Permutation Decomposition
Permutation
Source Address
Destination Address
Combination Matrix (source + destination)
Check conflicts using Window Method
List of messages with conflicting path
Phase 2 : Message Scheduling
Conflict Mapping Technique
Conflict Graph
Conflict Matrix
Inverse Conflict Matrix
Crosstalk-Free Scheduling Algorithm
Scheduling of messages
Output is the Execution Time and Number of Passes

Fig 3. Time Domain Approach Framework.

5.2. Permutation Generation

Before a permutation can be divided into its crosstalk-free subsets, the source and destination addresses of the permutation are randomly generated. A permutation refers to a one-to-one mapping from a source node to a destination node in the OON. The network size, *N* is defined as a base-2 integer, 2^{n} where $n=log_2N$, ranging from the smallest size 4 to the largest size 1024 that represents the number of source nodes and destination nodes of the network.

5.3. The Combination Matrix

To build the combination matrix, each source and destination address pair of a permutation will be represented separately in their *n*-digit binary structure, where $n = \log_2 N$. Then, both source and destination addresses from the pair are combined; with the source address put on the left followed by the destination address on the right

5.4. Conflict Discovery

5.4.1. Window method

Based on the combination matrix, conflict patterns are checked using some pattern-checking method. Window Method (WM) is one example of a pattern-checking method where the combination matrix is divided into windows of the same size; and if any two messages have the same bit pattern between them in any of the windows, then it implies conflict between the message pair. Thus, the two messages must not be scheduled in the same group. In WM, an optical window of size $m - 1$ where $m=\log_2 N$ and *N* is the size of the optical network is applied to the columns of the combination matrix, from left to right excluding the first and the last columns. For each optical window, the bit pattern for each message is compared for similarity with the bit pattern of the rest of the other $N – 1$ message(s) sequentially starting from message 0 to $N – 1$. If the bit pattern is the same, it will be mapped into the array of conflict pattern.

5.4.2. Improved Window Method

Sequentially comparing bit patterns among all messages in each optical window was found to be time consuming especially when the network size, N is large and the number of optical window increases. To reduce the execution time contributed by the WM, the Improved WM (IWM) was proposed that eliminates checking for conflicts in the first optical window[5].This is because the first optical window has the same conflict pattern where the first $N/2$ inputs in sequence uses the same SEs as the second half of the other $N/2$ inputs. Therefore, inputs 0 to ($N/2$ – 1) will have conflict with inputs $N/2$ to (N – 1), which is always true for any size of network, N.

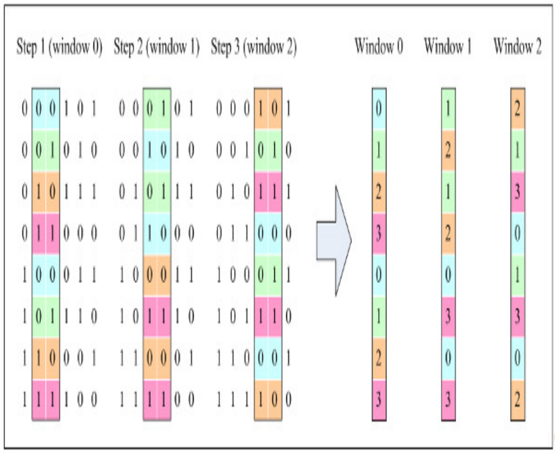
5.4.3. Bitwise Window method

Based on comparative analysis performed in it was concluded that the time spent for identifying conflicts is very high compared to routing the messages. Table 1 shows the execution time of WM compared to the time executed for scheduling and routing.

Table 1: WM Execution Time (*ms*)

Network Size	Routing + WM	WM	Routing
8	0.032	0.031	0.001
16	0.078	0.063	0.015
32	0.219	0.204	0.015
64	1.031	1.000	0.031
128	4.797	4.656	0.141
256	25.329	24.187	1.142
512	110.750	108.906	1.844
1024	519.922	499.046	20.876

Based on the analysis, then proposed the Bitwise Window Method (BWM) that significantly reduces the execution time of the WM. In the new BWM, each (n - 1)-bit binary optical window of the standard WM where $n=\log_2 N$ and N is the network size, be transformed into its equivalent decimal representation using bitwise operations. As a result, the number of columns used to compare each message for similar bit pattern is reduced to n, instead of ($n^2 – n$) for an N x N Omega network[6]. Figure 4 illustrates the transformation steps for each optical window in BWM implementation.



Step 1 (window 0)	Step 2 (window 1)	Step 3 (window 2)	Window 0	Window 1	Window 2
0 0 0 1 0 1	0 0 0 1 0 1	0 0 0 1 0 1	0	1	2
0 0 1 0 1 0	0 0 1 0 1 0	0 0 1 0 1 0	1	2	1
0 1 0 1 1 1	0 1 0 1 1 1	0 1 0 1 1 1	2	1	3
0 1 1 0 0 0	0 1 1 0 0 0	0 1 1 0 0 0	3	2	0
1 0 0 0 1 1	1 0 0 0 1 1	1 0 0 0 1 1	0	0	1
1 0 1 1 1 0	1 0 1 1 1 0	1 0 1 1 1 0	1	3	3
1 1 0 0 0 1	1 1 0 0 0 1	1 1 0 0 0 1	2	0	0
1 1 1 1 0 0	1 1 1 1 0 0	1 1 1 1 0 0	3	3	2

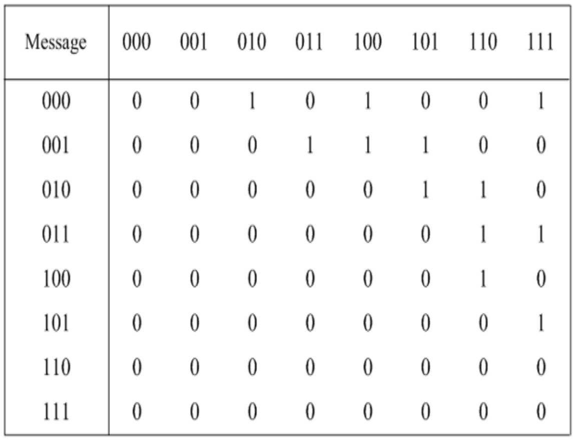
Fig 4.Optical Window Transformation in BWM.

5.5. Conflict Graph

The conflict graph is one of the foremost technique proposed to map conflicts discovered using WM. By definition, the conflict graph of an N-permutation π (where N is the network size) is the graph $G(V, E)$ where V is a set of vertices $\{v_0v_1v_2 \ldots v_{N-1}\}$ and E is a set of edges $\{(v_0, v_1), \ldots, (v_i, v_j), \ldots, (v_{N-2}, v_{N-1})\}$. Each vertex, $V = \{v_0v_1 \ldots v_{N-1}\}$ in the conflict graph represents a source node's address i.e. v_0 for source 000, v_1 for source 001 and so on for all nodes in the network. In the conflict graph, any two vertices v_i and v_j are connected by an edge, E to indicate conflict, if and only if they share a common SE at certain stage of the network.

5.5.1. Conflict Matrix

Another conflict-mapping technique that can be used to map conflict pattern identified using WM is called the conflict matrix.. The conflict matrix is defined as a square matrix, M with matrix size of N x N where N is the network size. The conflict matrix is illustrated in Figure 5. Since the message 000 has conflict with messages 010, 100 and 111, elements $M_{000,010}$, $M_{000,100}$ and $M_{000,111}$ are set to the value 1 to indicate conflict in the conflict matrix. The rest of the intersections for message 000 i.e. the intersections between message 000 and messages 001, 011, 101 and 110 are set to 0 value, which means that these messages will not cause crosstalk with the message 000 during routing in the network.



Message	000	001	010	011	100	101	110	111
000	0	0	1	0	1	0	0	1
001	0	0	0	1	1	1	0	0
010	0	0	0	0	0	1	1	0
011	0	0	0	0	0	0	1	1
100	0	0	0	0	0	0	1	0
101	0	0	0	0	0	0	0	1
110	0	0	0	0	0	0	0	0
111	0	0	0	0	0	0	0	0

Fig 5. The Conflict Matrix.

6. Scheduling Algorithms

To perform scheduling of the messages into crosstalk-free groups for routing in OON include the standard four Heuristic algorithms; Sequential Increasing, Sequential Decreasing, Degree Ascending and Degree Descending algorithm, Simulated Annealing (SA) algorithm, Genetic Algorithm (GA), Ant Colony Optimization (ACO) algorithm, Remove Last Pass (RLP) algorithm, Zero algorithm, Improved Zero (IZero) algorithm and Bitwise-Based algorithm. To evaluate the performance of the time domain scheduling algorithm, researchers have used two main parameters; the total execution time for scheduling permutations [7].

- The ACO algorithm successfully reduces the number of passes when limited crosstalk is allowed in the network. Unfortunately, when zero crosstalk is concerned, the number of passes is higher than the rest of the other algorithms.
- RLP algorithm gives the best result when the number of passes is considered. However, the algorithm consumes longer execution time than other time domain algorithms. Apart from the algorithm's dependency to other algorithm to obtain the initial solution, the RLP algorithm also involves complex procedures when making scheduling decisions.
- Improved the weaknesses found in the original Zero algorithm, IZero algorithm performed slightly higher in terms of its execution time for scheduling permutations compared to the original algorithm while maintaining the same result in the total number of passes to route a permutation All Bitwise-Based algorithms have shown to successfully reduce the execution time of the original algorithm , except that the number of passes obtained by the new algorithm is the same as before it is implemented using the Bitwise approach7.

7. Fast Zero Algorithm

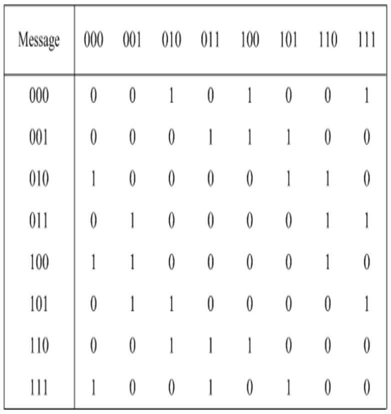
Fast Zero (FastZ) algorithm, is among the latest time domain scheduling algorithm proposed to optimally minimize the execution time of Zero-based algorithms. FastZ algorithm consist of three algorithms namely Fast ZeroX (FastZ_X), Fast ZeroY (FastZ_Y) and Fast ZeroXY (FastZ_XY) algorithms.

7.1. Permutation decomposition

Based on the time domain approach, scheduling depends very much on the pattern of conflicts among the messages. Conflict-mapping technique i.e. the conflict graph provides an easy access to refer conflicts between messages in the network before scheduling the messages. An efficient conflict-mapping technique affects the total execution time of an algorithm. Therefore, we proposed another technique called symmetric Conflict Matrix (sCM) to map conflicts in the network discovered using BWM. The new sCM is implemented in FastZ algorithm replacing the conflict matrix.

7.2. Symmetric Conflict Matrix

The sCM is defined as a square matrix, $S_{i,j}$ with matrix size of N x N where N is the network size. A great advantage using sCM compared to the conflict matrix is that the sCM provides a complete mapping of all possible conflicts in the network similar to the conflict graph. Scheduling algorithm can be simplified and more straightforward by comparing the intersection value of intersected messages to determine routability thus eliminates time-consuming procedures associated with multiple summation of the conflict matrix, finding intersections, and reducing the conflict matrix in Zero-based algorithms.



Message	000	001	010	011	100	101	110	111
000	0	0	1	0	1	0	0	1
001	0	0	0	1	1	1	0	0
010	1	0	0	0	0	1	1	0
011	0	1	0	0	0	0	1	1
100	1	1	0	0	0	0	1	0
101	0	1	1	0	0	0	0	1
110	0	0	1	1	1	0	0	0
111	1	0	0	1	0	1	0	0

Fig6. The Symmetric Conflict Matrix, sCM.

7.3. Message Scheduling

The basis of Zero-based algorithms lies in the Unique Case and Refine functions executed after obtaining the row or column summations of the conflict matrix. However, these procedures are time-consuming, thus contribute to longer execution time to schedule messages for routing in the network. Using sCM, scheduling of messages is more straightforward simply by checking through the intersections between the entries in the sCM, without prior row or column summation of the conflict matrix.

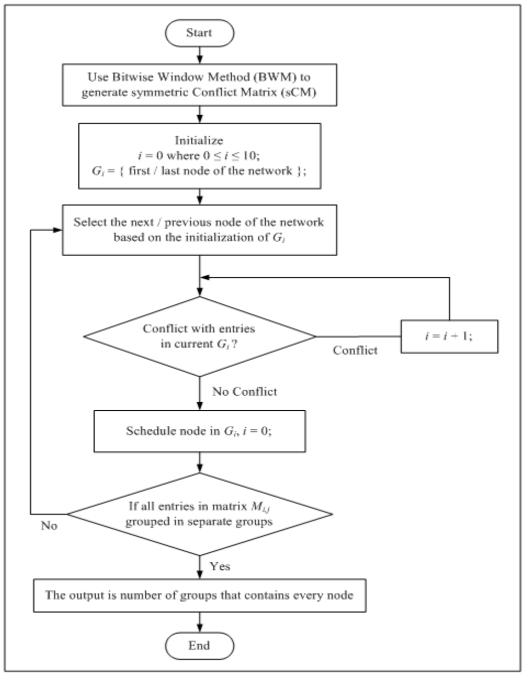

Start
Use Bitwise Window Method (BWM) to generate symmetric Conflict Matrix (sCM)
Initialize
$i = 0$ where $0 \leq i \leq 10$;
G_i = { first / last node of the network };
Select the next / previous node of the network based on the initialization of G_i
Conflict with entries in current G_i?
Conflict
$i = i + 1$;
No Conflict
Schedule node in G_i, $i = 0$;
If all entries in matrix $M_{i,j}$ grouped in separate groups
No
Yes
The output is number of groups that contains every node
End

Fig 7.General FastZ Algorithm Flowchart.

FastZ algorithm consists of three algorithms; FastZ_X, FastZ_Y and FastZ_XY algorithms. The difference between these algorithms is in the selection of which message to be added to the first scheduling group during group initialization. For initialization, FastZ_X algorithm selects the first message in the network to the first group. The rest of the messages are selected for scheduling ascendingly, starting from the second message based on the message's source address. On the contrary, FastZ_Y algorithm chooses the last message, *N* in the network for initialization. FastZ_Y algorithm schedules the other messages descendingly based on their addresses until all messages are scheduled into crosstalk-free groups. To schedule a permutation in the FastZ_XY algorithm, messages are scheduled using both FastZ_X and FastZ_Y algorithms sequentially. After both results are obtained, FastZ_XY algorithm compares both results and chooses whichever result that has the lowest total number of passes as its final result.

7.4. Fast Zero Algorithm with RLP

We introduced the latest development of Zero-based algorithms; the FastZ algorithm with RLP or FastRLP in short. The algorithm is designed by integrating FastZ algorithm with the RLP algorithm, with attempt to minimize the total number of passes for routing a given permutation. Based on analysis performed in, the RLP algorithm has shown to successfully schedule a permutation with less number of passes, than the Maximal Conflict Number (MCN) required for the permutation.

In FastRLP algorithm, messages are scheduled using FastZ algorithm to obtain initial scheduling groups called the initial solution. Depending on which algorithm used to obtain the initial solution, FastRLP algorithm can be divided into two algorithms. If the FastZ_X algorithm is used to obtain the initial solution, it is referred as the FastXRLP algorithm. Otherwise, if the FastZ_Y algorithm is used, then it is referred as the FastYRLP algorithm. After the initial solution is derived, RLP algorithm is used to remove the last pass by relaying messages to the unused paths of the previous passes. The RLP algorithm is executed if and only if the number of initial scheduling groups generated is more than two. This is because there is not a permutation that can be scheduled for routing in less than two groups without crosstalk in an OON regardless of the network size.

8. Numerical results and Discussions

Each of the algorithms is simulated 10,000 times for each execution on different network sizes, *N* and presented in average for comparative analysis. Performance evaluation will be based on two types of parameters; the execution time and number of passes.

The execution time is defined as the time elapsed between the beginning and the end of its execution on a sequential computer measured in milliseconds (*ms*). The execution time calculated for each algorithm includes the time taken to generate random permutation addresses, execute window transformation, check for conflicts between the messages, mapping conflicts into the sCM and finally schedule the messages into the crosstalk-free groups for each permutation set. Minimum execution time reflects better performance of an algorithm [8].

Based on the time domain approach, transferring messages from source nodes to the intended destination nodes without crosstalk involves dividing the messages into independent crosstalk-free groups called passes. These passes can be routed in one group at any given time. Less number of passes implies that more messages can be scheduled in the same pass for routing. Therefore, the number of passes obtained for an algorithm reflects the efficiency of the algorithm in terms of better scheduling strategy employed.

We divided and clustered the results of each algorithm into three categories; ZeroX, ZeroY and ZeroXY since they differ between each other in scheduling. Figure 8 and Figure 9 present the results for ZeroX algorithm, Figure 10 and Figure 11 present the result for ZeroY algorithm, while Figure 12 and Figure 13 present the result for ZeroXY algorithm in terms of the execution time and number of passes.

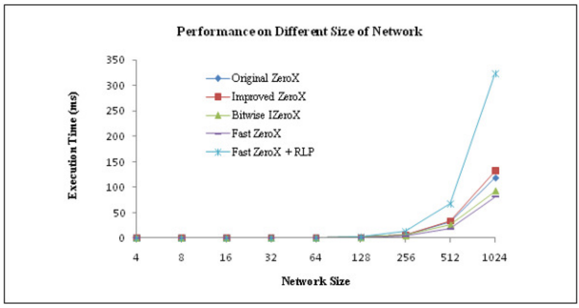


Fig 8. Execution Time vs. Network Size of the ZeroX Algorithm.

When the execution time is considered, it is evident that FastZ algorithm performs the best with the lowest average execution time consistently for all *N*, compared to all Zero-based algorithms (refer Figure 8, Figure 10 and Figure 12). Integrating FastZ and RLP algorithm result in higher execution time especially in large network, *N* = 1024 nodes. This is contributed by the RLP function embedded in the algorithm in order to reduce the number of passes.

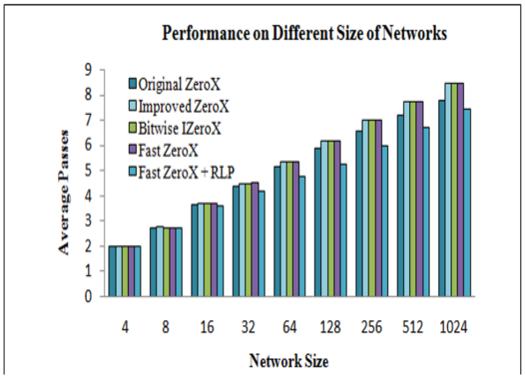


Fig 9. Number of Passes vs. Network Sizes of the ZeroXAlgorithm.

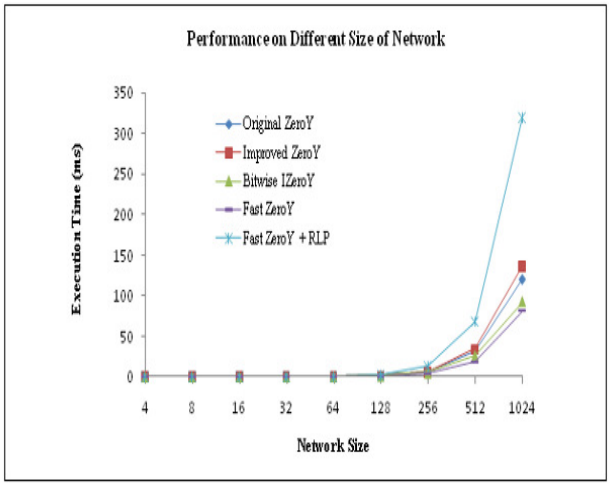


Fig 10. Execution Time vs. Network Sizes of the ZeroY Algorithm**.**

In terms of the number of passes generated for permutation routing, FastZ algorithm results closely to the IZero and BIZero algorithms. The increase in the number of passes of the FastZ algorithm compared to the original Zero algorithm was as expected. In terms of the elimination of crosstalk, the results in the number of passes are almost equal to that of the FastZ algorithm. This is mainly because the number of passes generated by FastZ algorithm may be the same as IZero and BIZero algorithms except which message(s) scheduled in each pass may be different when using FastZ algorithm.

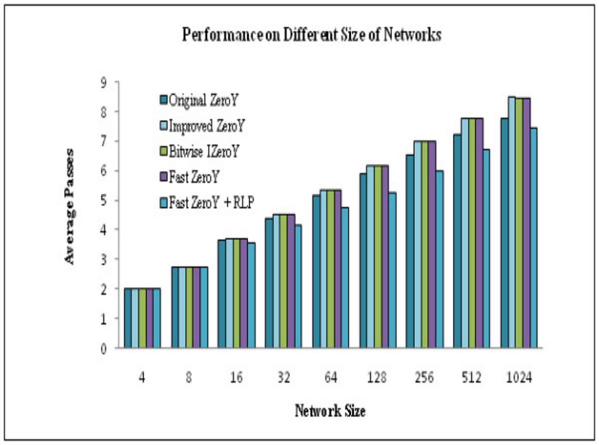


Fig11. Average Number of Passes vs. Network Sizes of the ZeroY Algorithm

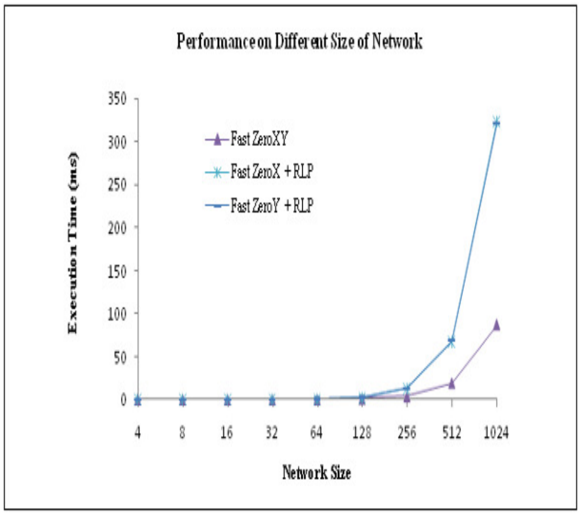


Fig 12. Execution Time vs. Network Sizes of the ZeroXY Algorithm.

Integrating the FastZ algorithm with RLP algorithm known as FastRLP algorithm has shown to successfully reduce the number of passes generated for a permutation starting from *N* = 16 onward (refer to Figure 9 and Figure 11). The results are not as significant for network size with small *N* (<16) because in the time domain approach no more than one input/output link can be active at any given time. Therefore, the minimum number of passes for these network ranges is limited to two passes for a permutation where in this case the RLP algorithm will not be executed at all.

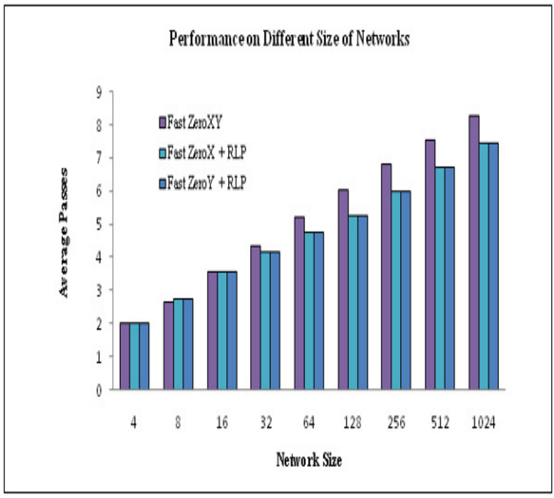


Fig13. Average Number of Passes vs. Network Sizes of the ZeroXY algorithm.

When compared to FastZ_XY algorithm, FastRLP algorithm reduced the number of passes only when *N*> 16 as shown in Figure 13. This is because in FastZ_XY algorithm, it compares and chooses the minimum number of passes generated between FastZ_X and FastZ_Y algorithms in each execution. It was also proven in that FastZ_XY algorithm has less number of passes compared to individual FastZ_X and FastZ_Y algorithm.

9. Conclusion and Future Works

We have presented the development of crosstalk-free scheduling algorithms for routing in OON, a type of OMIN topology. We also presented two latest developments in crosstalk-free scheduling algorithms; FastZ and FastRLP algorithms. Using the proposed sCM to map conflicts in the network, both algorithms have proven to improve scheduling in terms of the execution time as well as the number of passes. Through simulation technique, FastZ algorithm reduced the execution time by 32% compared to previous Zero, IZero and BIZero algorithms without much difference in the number of passes generated. On the other hand, FastRLP algorithm reduced the number of passes by 11% in average compared to all Zero-based algorithms despite significant increase shown in the algorithm's execution time.

In future, we would suggest that the execution time of FastRLP algorithms be reduced using bitwise operations. The idea of sCM can also be applied to any other time domain algorithms to map conflicts identified between the messages in the network. Next, FastZ and FastRLP algorithms can be implemented in parallel to achieve exponential improvement in the algorithm's execution time. Finally, it is worth to consider the design to support for multicast communication in the network. In this case, the multilayer architecture can be incorporated with the single layer design of the OON topology.

References

[1]Abdullah, M., M. Othman, H. Ibrahim, & S. Subramaniam. 2008. Simulated annealing algorithm for scheduling Divisible Load in large scale data grids. Proceedings of the International Conference on Computer and Communication Engineering, May 13-15, IEEE Xplore Press, KualaLumpur, pp: 1032-1036. DOI: 10.1109/ICCCE.2008.4580765.

[2]Abed, F. and M. Othman, 2007. Efficient window method in optical multistage interconnection networks. Proceedings of the IEEE International Conference on Telecommunications and Malaysia International Conference on Communications, May 14-17, IEEE Xplore Press, Penang, pp: 181-185. DOI: 10.1109/ICTMICC.2007.4448626.

[3] J.Sengupta, "Interconnection networks for parallel processing", Nutech Books, Deep and Deep publications,

New Delhi, (ISBN-81-7629-736-4), pp. 7-8, 15- 19, 42-45, 69-70, 2005.

[4] C. P. Kruskal, M. Shir, “Performance of multistage interconnection networks for multiprocessors”, IEEE transactions on computers, vol. C 32, December 1983.

[5] P. Dharma Aggarwal, “Testing and Fault Tolerance of Multistage Interconnection Networks”, IEEE Transactions on Computers, pp. 41-53, 1982.

[6] P.K Bansal, Kuldeep Singh, R.C Joshi , “On Fault tolerant Multistage Interconnection Network”, Conference on Computer Electrical
Engineering, vol. 20, no. 4, pp. 335-345, 1994.

[7] N. Laxmi Bhuyan, Qing Yang, P. Dharma Aggarwal, “Performance of Multiprocessor Interconnection Networks”, Proceeding of IEEE, pp. 25-37, 1989.

[8] N. Tzeng, P. Yew and C. Zhu, “A fault-tolerant for multistage interconnection networks”, 12th International Symposium on Computer Architecture, pp.368–375, 1985.

[9] A. Varma and C.S. Raghavendra, “Fault-tolerant routing in Multistage interconnection networks”, IEEE Transactions on Computers 385–393, 1989.

First Author

I am SindhuLakshmi Manchikanti completed my MCA at Viswaganga Institute of Technology in 2011. Completed my B.Sc.,ComputerScience in Sri Sai Baba National Degree College in 2008. Completed my Intermediate in Nalanda Junior College in 2005.Completed my SSC in Little Flower Montessori English Medium High School in 2003. After completion of my Post Graduation I worked for One year as a Software Trainer in NIIT at Anantapur, AndhraPradesh.
Currently I am working in Sree Vidyanikethan Institute of Management as an Assistant Professor at Tirupathi, Andhra Pradesh. I attended to the National Conference which is held at Sri Venkateswara College of Arts and Sciences at Tanjore, Tamilnadu and also the paper was published on Voice Recognization Technology and Applications.

Second Author

I am Gayathri Korrapati completed my MCA at Annamacharaya Institute of Technology in 2011.
After completion of my Postgraduation, presently I am working in Sree Vidyanikethan Institute of Management as an Assistant professor at Tirupathi, I had completed B.S.C in lekpakshi degree college at Proddatur, Kadapa district, A.P. I had completed intermeadiate in Bhavana junior college at Prodductur, Kadapa distirict, A.P. I attended to the National Conference which is held at Sri Venkateswara College of Arts and Sciences Tanjore, Tamilnadu and also the paper was published on voice Recognization Technology and Applications.