AN ALGORITHM FOR NON CONFLICT SCHEDULE WITH DIAGONAL ACTIVATION OF JOINT SUB MATRICES

Kiril Kolchakov, Vladimir Monov
Institute of Information and Communication Technologies, Bulgarian Academy of Sciences
kkolchakov@abv.bg, vmonov@iit.bas.bg

In this paper, we have developed an algorithm for a non-conflict schedule obtained through the activation of joint diagonal sub matrices in switching nodes of type Crossbar. A software model is eleborated with the purpose of studying the algorithm. The size of sub matrices in the Crossbar is optimized and a finite automat is designed for a sub matrix control. A comparison with known algorithms with diagonal activation of connection matrix is done.

Key words: *Network nodes, Message switching Node traffic, Crossbar switch, Conflict elimination, Packet messages, Sparse matrix.*

1. Introduction

The traffic via Crossbar switching nodes is casual and depends on the users. The formulation of the conflict issue during operation of the switching nodes is as follows: the dimensions of the switches in the switching nodes are N x N, where N sources of packet massages are connected to N receivers via the switch of the switching node. The switching node traffic is random by nature and depends on the users. Conflicts are available in the following two cases:

- When one source of massage requests communication to two or more massage receivers
- When one massage receiver receives communication requests from two or more massage sources.

The evasion of conflicts is directly related to the switching node performance. The status of the switch of the switching node is represented with the so called connection matrix. For N x N dimensional switch the dimension of the connection matrix T is N x N also, where every element $T_{ij} = 1$ if the connection request from i-source to j- receiver exists. In the opposite case $T_{ij} = 0$.

A conflict situation arises if any row of the connection matrix has more than a single 1, which corresponds to the case when one source requests a connection with more than one receiver. The presence of more than a single 1 in any column of the matrix T also indicates a conflict situation, it means that two or more sources have requested a connection with the same receiver [1].

The problem of conflict situations is solved by compiling a non-conflict schedule. Algorithms using different approaches to obtain non-conflict schedules are

described in [5]. Algorithms with diagonal activation of connections matrix to obtain non-conflict schedule are described in [1,2,3]. Software models to obtain a non-conflict schedule through the sparse matrix-masks are described in [7].

2. Description of the algorithm

The connections matrix T with N x N size, where N is being the degree of two, is divided into sub matrices (S) with dimension n x n, (n also is a degree of two), i.e:
T = [S i j] , i = 1- n, j = 1- n
The sets of sub matrices located along the main diagonal are processed simultaneously in each of the diagonals.
For submatritces in diagonals parallel to the main one, the principle of reconciliation is used, [1].

The idea of synthesis of the algorithm **ADAJS** (Algorithm with diagonal activations of joint sub-switching matrices) is based on the knowledge that the diagonal sub matrices with requests for service in the matrix T are non-conflict in the diagonal where they are located.
There are diagonals with sub matrices of requests that are non-conflict to one another. Figure 1 shows joint couples of non-conflict diagonals with sub matrices of requests for service and the main diagonal of sub matrices that can not be jointed with anyone else.

The whole process of the implementation of **ADAJS** algorithm for obtaining a non-conflict schedule is divided into steps. The first step refers to the main diagonal sub matrices processed simultaneously and without conflict. The next steps are related to the reconciliation of the diagonals parallel to the main diagonal by pairs (Figure 1).

The analytical description of the steps shown on Figure 1 is as follows:

Step1 : $S_{11}, S_{22}, S_{33}, S_{44}$ **Step3 :** $S_{21}, S_{32}, S_{43}, S_{14}$

Step2 : $S_{41}, S_{12}, S_{23}, S_{34}$ **Step4 :** $S_{31}, S_{42}, S_{13}, S_{24}$

$T = [S_{ij}]$, i = 1 - 4, j = 1 - 4

The size (n) of the sub matrix determines the number of steps (**I**) as follows

$$I = N/n. \qquad (1)$$

For **N** = const., **I = f (n),** where **$1 < n \leq N / 2$.**

The software model **SMADAJS**, describing the algorithm **ADAJS** (Algorithm with diagonal activations of joint sub-switching matrices) is written in MATLAB programming language. Our study of **SMADAJS** software model is performed on Dell Precision Workstation 420.

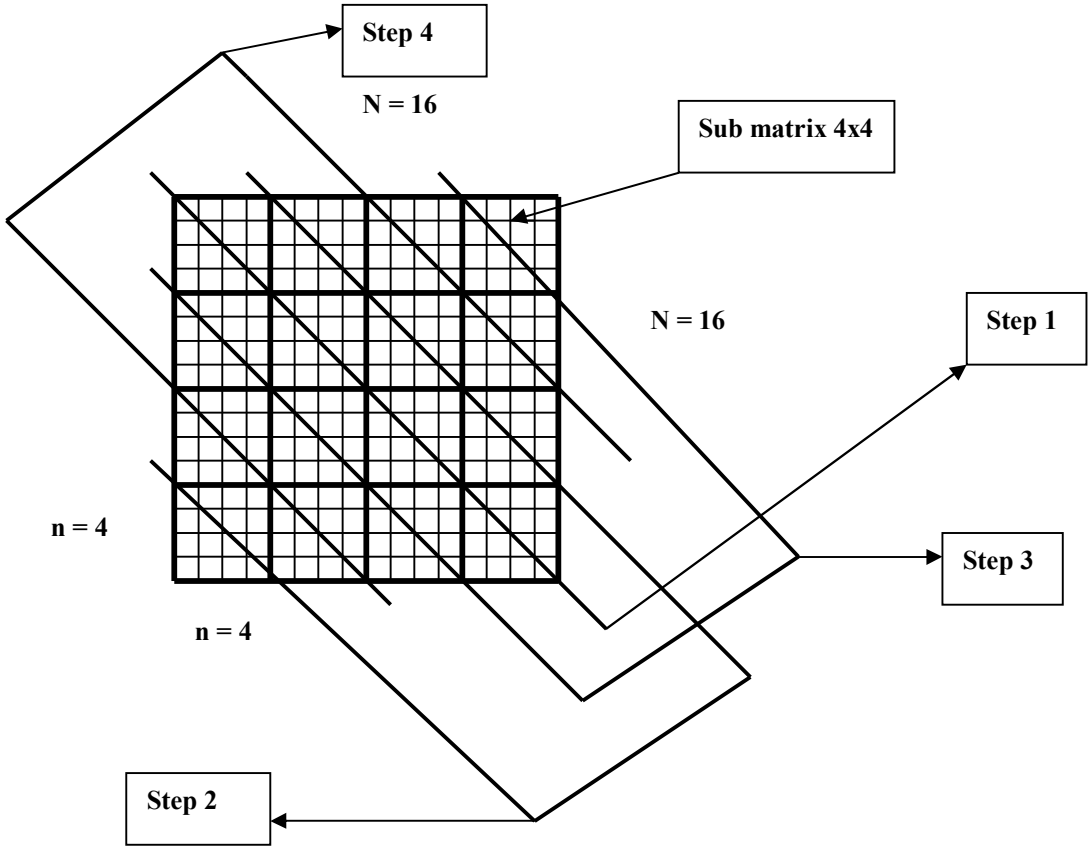

Figure 1 : Diagonal activation of joint sub matrices

3. Determination of the submatrix optimal size

The goal is to determine the optimal size of the sub matrix related to performance of the software model **SMADAJS**. We have studied the necessary time to work (**TW**) of the model for different values of n, where N = const. The results are shown graphically in Figures 2 and 3.

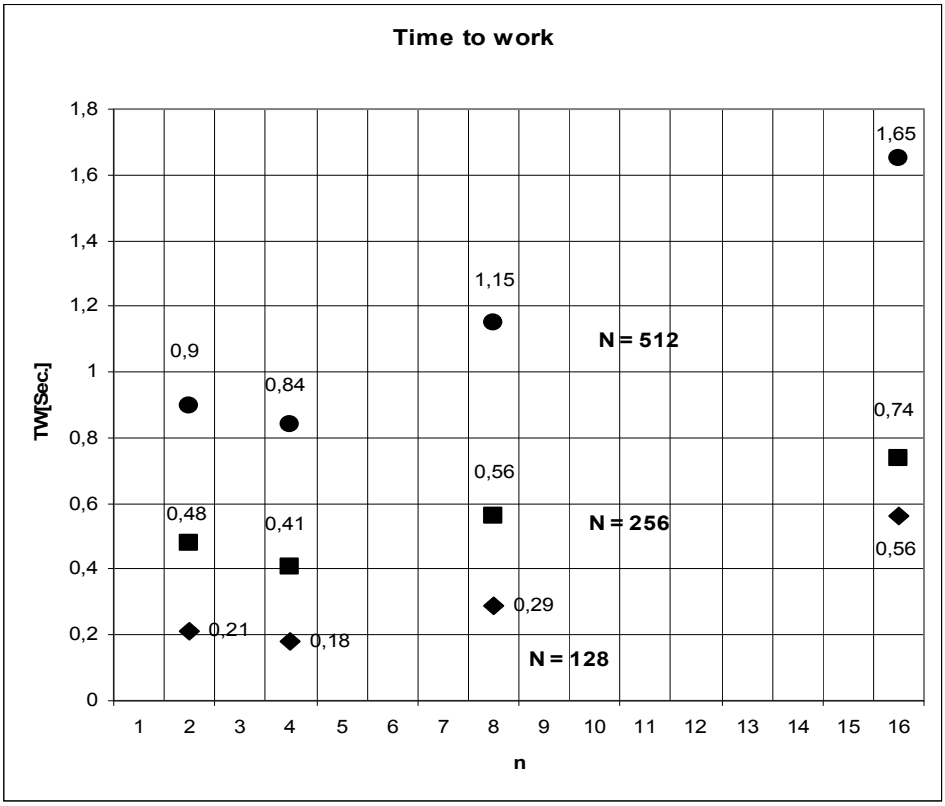

Figure 2 : Time to work for N = 128, N = 256 and N = 512 as a function of n.

A clear minimum of the operating time (**TW**) is seen for n = 4 for N = 128, 256, 512, 1024 and 2048. It can be concluded that for size N up to 2048, the optimal sub matrix size is $n_{opt} = 4$.

In order to reduce CPU work controlling the requests execution in a switching node of type Crossbar, it is appropriate to synthesize a finite automat. It will take care to handle the individual sub matrices. Then CPU will be limited to initialize the finite automat during the work of the algorithm **ADAJS** for the respective sub matrix only.

In our case the optimal size of the sub matrix is $n_{opt} = 4$. Therefore, the final automat shell replaces CPU for a sub matrix 4 x 4. Figure 4 shows the four steps of the algorithm for sub matrix 4 x 4 corresponding to the four states of the final automat - A_1 to A_4. A state A_0 should be available for the initial reboot of the final automat. In state A_0 all requests in the sub matrix are not allowed.

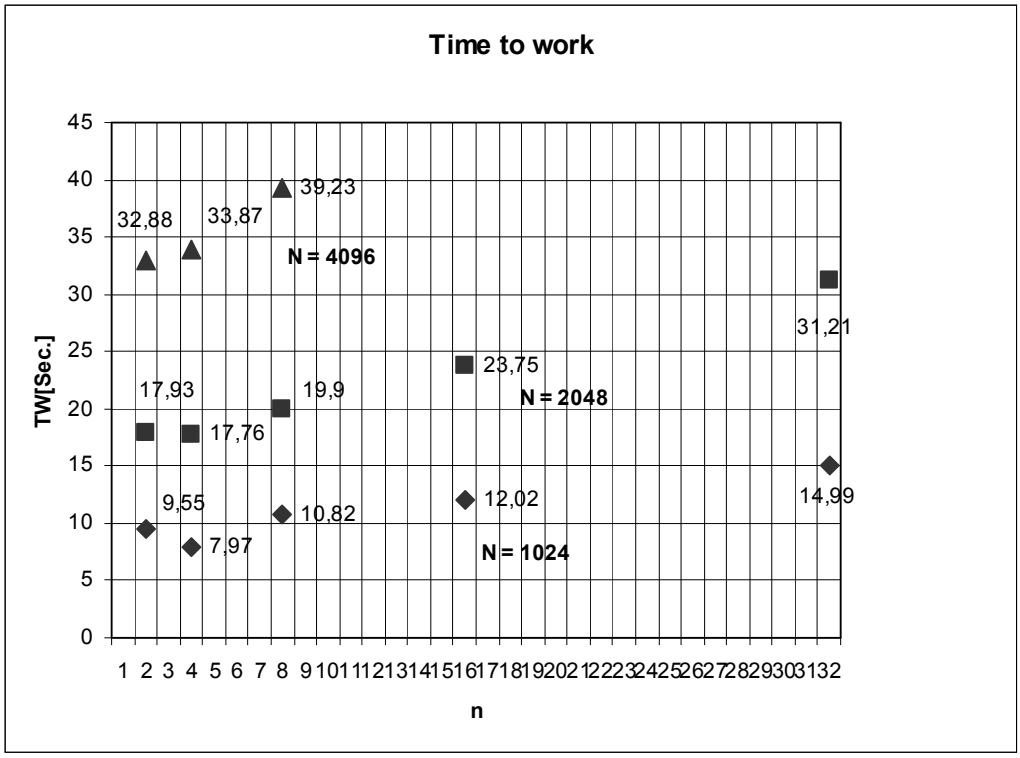

Figure 3 : Time to work for N = 1024, N = 2048 and N = 4096 as a function of n.

In our case a Moore finite automat is suitable, because each state corresponds to an output signal that is not affected by the input word, as in the case of Mealy automat [9].

Figure 5 shows the state transition graph of the synthesized Moore automat for control of the sub matrix presented in Figure 4. Using Z_1 signal, the automat passes through the states from A_0 to A_4 and each state A_1 to A_4, corresponds to an output signal $St._1$ (Step1) to $St._4$ (Step4) which control the connection sub matrix directly. Using Z_2 signal, the automat goes to A_0 state with an output signal R which is an indication for CPU switching node that the automat is free and could be started at any time.

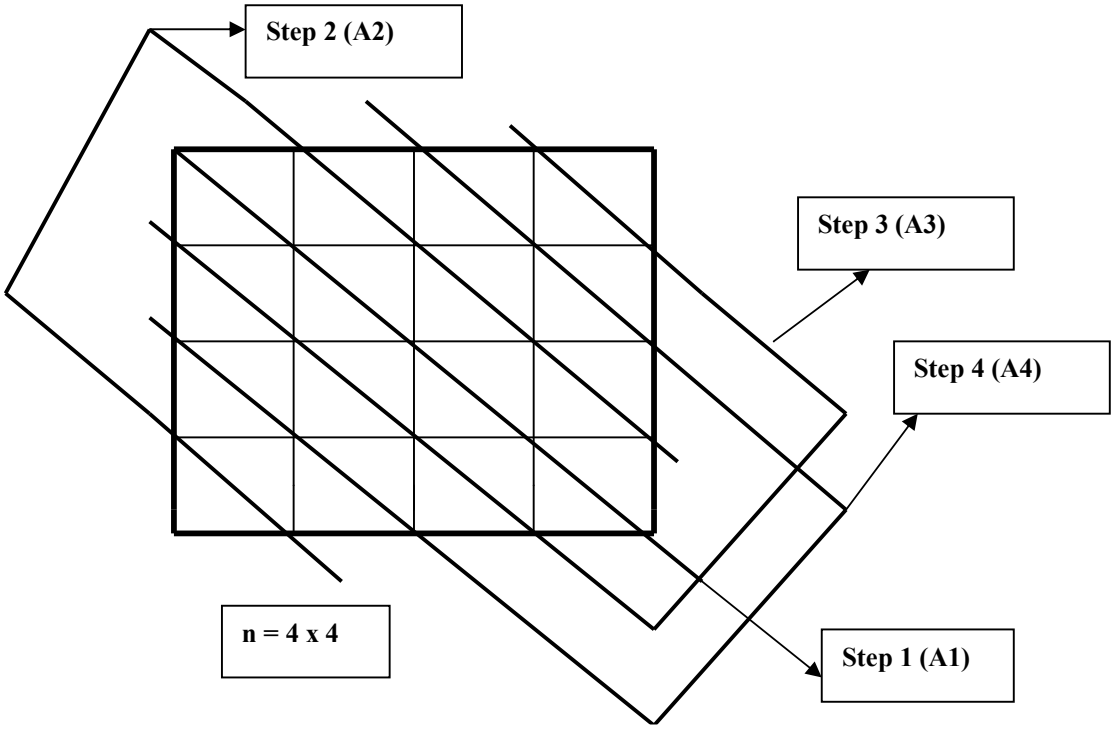

Figure 4 : Sub matrix 4 x 4.

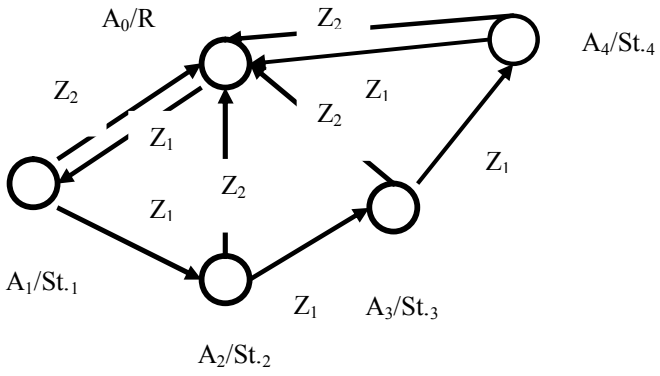

Figure 5 : Graph of the transitions.

4. A comparison of ADAJS with other algorithms with diagonal activation of the connection

The comparison is in terms of performance and necessary memory. There are two algorithms with a diagonal activated connections matrix:

1. **ADA (A**lgorithm with **D**iagonal **A**ctive) [3].
2. **AJDA** (**A**lgorithm with **J**oint **D**iagonals **A**ctivations) [2].

The comparison is done by the software models SMADAJS, SMADA, SMAJDA of the algorithms. The software models SMADAJS, SMADA SMAJDA are written in MATLAB programming language. Studies were carried out on Dell Precision Workstation 420.

N	**TW** [Sec.] **SMADA**	**TW**[Sec.] **SMAJDA**	**TW**[Sec.] **SMADAJS**
500	26,80	34,39	2,98
1000	99,21	116,39	5,84
1500	223,61	252,61	9,03
2000	522,96	427,20	11,39
2500	617,37	588,35	13,42
3000	863,60	852,12	16,35
3500	1006,32	994,37	18,33
4000	1258,74	1360,68	21,61
4500	1835,4	2170,36	24,29

Table 1: Results from the software models investigation related to time to work

N	**M [KB] SMADA**	**M [KB] SMAJDA**	**M [KB] SMADAJS**
500	10 008, 008	10 008, 024	0,760
1000	40 016, 008	40 016, 024	0,768
1500	90 024, 008	90 024, 024	0,768
2000	160 032, 008	160 032, 024	0,752
2500	250 040, 008	250 040, 024	0,768
3000	360 048, 008	360 048, 024	0,768
3500	490 056, 008	490 056, 024	0,752
4000	640 064, 008	640 064, 024	0,760
4500	810 072, 008	810 072, 024	0,768

Table 2: Results from the software models investigation related to memory

From Table 1 it is seen that in terms of performance, **SMADAJS** is faster for some values of N up to 20 times more than **SMAJDA** and **SMADA**.

The memory required for **SMADAJS** is less than 1KB and it is almost constant for the whole range of the study and can be neglected in comparison with the memory needed for **SMAJDA** and **SMADA**.

5. Conclusion

As a main conclusion, our results show that the synthesised algorithm **ADAJS** (Algorithm with diagonal activations of joint sub switching matrices) for obtaining non-conflict schedule in Crossbar switching node is much faster than the **ADA** (Algorithm with Diagonal Active) and **AJDA** (Algorithm with Joint Diagonals Activations). Also, our algorithm requires a negligibly small amount of memory as compared to **ADA** and **AJDA**. Another important property of **ADAJS** is that it facilitates the CPU operation by local hardware control of sub matrices connections with finite automats.

REFERENCES

1. Kolchakov K., Taschev T. "An Algorithm of Non-conflict Schedule with Joint Diagonals Activation of Connectivity Matrix", Proceedings of the International Conference on Computer Systems – CompSysTech'12, 22-23 June 2012, Ruse, Bulgaria, ACM PRESS, ICPS, VOL.630, pp.245-250, ISBN 978-1-4503-1193-9.
2. Kolchakov K. Examination on Algorithms for Non Conflict Schedule with Diagonal Activation in Case of a Large Size Switching Matrix. International Conference Automatics and Informatics `12 03-05.10.2012 Sofia, Bulgaria. Proceedings CD: ISSN 1313 – 1869. pp.341-344.
3. Kolchakov K., "An Algorithm Synthesis of Non-Conflict Schedule by Diagonal Connectivity Matrix Activation" Proceedings of the International Conference AUTOMATICS AND INFORMATICS`11, John Atanasoff Society of Automatics and Informatics, Bulgaria, Sofia 03.10-07.10.2011., pp. B-247 – B251, Proceedings ISSN 1313-1850, CD ISSN 1313-1869.
4. Tashev T. Modelling throughput crossbar switch node with nonuniform load traffic. Proceedings of the International Conference "Distributed Computer and communication networks DCCN 2011", October 26-28, 2011, Moscow, Russia. R&D Company "Information and Networking technologies", Moscow, Russia, pp.96-102. (in russian)
6. Tashev T. Computering simulation of schedule algorithm for high performance packet switch node modelled by the apparatus of generalized nets. Proceedings of the 11th International Conference CompSysTech'2010, 17-18 June 2010, Sofia, Bulgaria. ACM ICPS, Vol.471, pp.240-245.
7. P. Wanjari, A. Chonhari, "Implementation of 4x4 crossbar switch for Network Processor", International Journal of Emerging Technology and Advanced Engineering, Website: www.ijetae.com (ISSN 2250 – 2459, Volume 1, Issue 2, December 2011).
8. D. Kim, K. Lee and H. Yoo, " A Reconfigurable Crossbar Switch with Adaptive Bandwidth Control for Networks-on-Chip", IEEE International Symposium on Circuits and Systems, 2005.
9. M.Schaller, K. Svozil, "Automaton Logic", International Journal of Theoretical Physics, Vol.35, No. 5, pp. 911-940, 1996, Plenum Publishing Corporation-Springer.