A Novel Approach to Improve the Performance of Interconnection Networks with Hot-Spots*

J. M. García and A. Flores
Dept. de Informática y Sistemas
University of Murcia
Campus de Espinardo, s/n 30080 Murcia (Spain)
{jmgarcia, aflores}@dif.um.es

Abstract

Congestion in interconnection networks due to the presence of hot spots is an important and difficult problem that occurs in parallel machines. This problem has been studied in depth and different solutions for the case of multiprocessors with shared memory have been proposed. Current trends point towards the implementation of systems with physically distributed memory, either based on message passing (multicomputers) or on a single shared memory address space (multiprocessors). Our paper is developed in this context. Up to now, proposals to improve the throughput of networks with hot-spots have focused on using virtual channels or adaptive algorithms. We present a novel solution based on reconfigurable networks. A reconfigurable network is one in which nodes can change their position depending on the communication pattern in order to diminish the congestion produced in the network and, therefore, increase its throughput. We studied this problem in two-dimensional k-ary n-cube networks using a deterministic routing algorithm and wormhole routing. In this paper the main features of a reconfigurable network are presented and the results obtained by simulation are shown. These results confirm that this technique as a very interesting one for systems with distributed memory, with applications to a great variety of problems.

1. Introduction

The growing demand for high processing power in various scientific and engineering applications has made multiprocessor architectures increasingly popular. This is exemplified by the proliferation of a variety of parallel machines with different design philosophies. All these architectures rely on an efficient interconnection network. The interconnection network is often the critical component of a large parallel computer because performance is very sensitive to network latency and throughput and because the network accounts for a large fraction of the cost and power dissipation of the machine. An efficient communication network for high-performance parallel computers must provide low latency message transmission. An interconnection network is described by its topology, routing algorithm and flow control. The topology of a network is the arrangement of its nodes and channels into a graph. Routing algorithm decides a path chosen by a message in this graph. Flow control deals with the allocation of channel and buffer resources to a message as it travels along this path. Contemporary distributed memory systems use wormhole routing mainly due to its high transmission efficiency and reduced buffer size requirements. The blocking due to using wormhole routing, can considerably reduce the performance of the network. In large parallel processing systems there are applications in which blocking situations can occur more easily. Therefore, it is necessary to study this problem deeply and search for adequate solutions. The case of hot-spots is very common in large parallel systems, whether they have shared-memory or use message-passing. This situation is produced when many processors request access to the same data item, or service by the same processor, at the same time. Such hot-spot contention creates congestion in the system. In shared-memory multiprocessor systems, the memory module containing the hot-spot data item may become saturated. In distributed-memory multicomputer systems, communication links leading to the hot-spot processor may become saturated. In the past, hot-spot conten-

*This work was supported in part by the Spanish CICYT under Grant TIC94-0510-C02-02

tion has been widely studied in the context of multiprocessor systems that utilize buffered multistage interconnections networks (an indirect network) to connect processors to the shared memory modules [16]. It has been observed that the access times for all memory references may be severely degraded, not just the references to a hot-spot location, due to a phenomenon called tree saturation [12]. It is easy to see that hot-spots can also exist in multicomputer systems. For example, synchronization may be required in the applications that are executed in those machines. In this situation, the processor which is coordinating the synchronization activity may become a hot-spot node. More recently studies have been carried out in direct networks like binary hypercube networks [6].

This paper is concerned with hot-spot contention in distributed memory systems with a 2-D torus network. We propose a reconfiguration mechanism to improve the performance of these systems. The goal of a reconfigurable network is to increase the performance by minimizing the congestion of messages in the network. Therefore, when a zone of the network is highly congested due to the amount of messages, the reconfiguration mechanism will try to put the nodes in the most convenient positions in the network to alleviate this situation. That is, with the reconfiguration of the network we aim at reducing the congestion by means of changing the position of the nodes that receive a large number of messages and that cause a large collapse in the network due to the deterministic routing. In this paper, we are going to study the improvement produced by a reconfiguration mechanism in a reconfigurable network with hot-spots. This study can be applied to message-passing systems as well as shared-memory systems, though we have mainly developed it for multicomputers (message-passing system). Thus, we use messages with large size in our simulations. The rest of the paper is organized as follows: in the next section, we introduce the hot-spot problem and we review the main solutions up to now. In section 3, we present the reconfigurable network and we describe some interesting properties used in our approach. The evaluation of the reconfigurable multicomputer and the simulation results are described in section 4. Finally, some conclusions and ways for future work are drawn.

2. The Hot-Spot Problem

Congestion in the interconnection network arises when the network cannot sustain the flow of input messages. Instead, a hot-spot arises if too many messages are routed through a small subset of the interconnection network. Therefore, a hot-spot [12] causes a large percentage of the messages routed to be transmitted through the same link. Congestion and hot-spots can severely degrade the network performance and, hence, the performance of the overall system in wormhole networks. In these networks, the time to route a message is fairly insensitive to the distance between nodes but, on the other hand, these networks are fairly sensitive to conflicts on the same link. Once a hot-spot is formed, most messages in the network would be blocked, and the network throughput may drop to nearly zero. A congested area is composed of a congested link and those links that have forward paths to the congested link. In a congested area, both non-hot and hot messages are being blocked for a very long time, and therefore, the throughput of the area is very low. Avoiding hot-spots is even more important in the presence of non local communications, since the simultaneous presence of all these phenomena can result in a very low performance. Furthermore, while a proper hardware support (wormhole routing) can improve the latency of non local communications, congestion and hot-spots mostly depend upon the communication patterns of the parallel algorithm adopted.

In multiprocessors, many different approaches have been suggested to eliminate the hot-spot problem. The basic idea of these schemes is to incorporate some hardware in the interconnection network to trap and combine data accesses if they are directed at the same memory location. Combining requests reduces communication traffic that leads to a lower average network delay time. However, the hardware required for such schemes is extremely expensive [12]. Nowadays, there are several studies to improve the throughput with hot-spots, as in [16], [15] or [11].

In multicomputers, several routing and flow control mechanisms have been proposed to reduce or avoid congestion, such as virtual channels, random routing or adaptive routing. In virtual channels [4] the delay experienced by a message depends on the number of messages that cross the same link but not upon the size of these messages. However, with virtual channels the congestion is reduced only if the number of conflicts is low. Recent studies [3] have shown that virtual channels are expensive, increasing node delay considerably. Random routing helps reducing the contention produced by hot-spots. However, the main disadvantage of random routing is that it does not exploit the locality in the user program [2]. Adaptive routing reduces network latency and increases network throughput, but this technique is more complex than deterministic routing, usually slowing down clock frecuency. Another proposed approach is restricted the packet injection in order to reduce the congestion in the network

[14]. The node selection can be done at regular time intervals (congestion avoidance) or on-demand (congestion detection and recovery). This scheme improves the node delay and so, the network congestion, but the network throughput is not improved. Finally, in [6] the impact of hot-spot contention in multicomputers using two types of static interconnection networks, the standard binary hypercube network and a hierarchical network (the BH/BH network) is considered. This paper introduces a new deterministic routing algorithm to improve the performance of these networks.

Our work is developed in the context of distributed memory systems. Until now, little work has been done about hot-spots in this context. However, current trends in parallel machines aim at physically distributed systems, whether they implement the message passing model or the shared memory model. Hot-spots in multicomputers can occur for many reasons. For example, a typical case for a hot-spot is the result of a global synchronization operation. In our paper, as in the Pfister and Norton work [12], we assume an open model, in which processors generate requests continuously without being blocked by responses being returned. Other authors have studied the hot-spot problem with a model where processors have a maximum of Nout requests outstanding before they are required to be blocked by a reply [1]. However, in this situation only high fractions of hot-spot traffic cause significant performance degradation. Open system models present a more realistic situation in hot-spot studies.

3. Reconfigurable Networks

In this section, we briefly describe what an interconnection network with reconfigurable topology consists of. As we will see, this class of network is a valid alternative to resolve the problem of hot-spots in a system with distributed memory. Interconnection networks with a wormhole switching mechanism are insensitive to the communication distance, but they are fairly sensitive to conflicts on the same link, that is, the congestion problem. A reconfigurable network is a very adequate technique to solve this problem and reduce the communication cost. Basically, this technique consists of placing the different processors in those positions in the network which, at each computational moment and according to the existing communication pattern among processors, are more adequate for the development of such a communication pattern. There are two types of reconfiguration: static and dynamic. The first approach is based on a static switching topology. A program is divided into several phases, where each phase requires a different topology. Before a new phase starts its execution, a new topology is selected by means of a software reconfiguration point. This approach is quite simple, but the flexibility of reconfiguration is limited. The second approach consists of changing the topology arbitrarily at runtime. In this paper, we focus on dynamic reconfiguration.

3.1. An overview

In this section, we present a reconfigurable network as a good solution to the problem of network contention. This technique offers more advantages than other solutions such virtual channels or adaptive algorithms. We can see the network contention as a condition under which the communication delay of non-hot messages becomes very large because of excessive demands of hot messages for the hot nodes of the network. When the blockage of a message is produced in a network with wormhole switching, some links remain occupied by the message but there is no flow of information through these links. Since the flits of this message cannot be mixed with flits of other messages, the bigger the contention of the network, the larger the blockage, and therefore, most links of the network will be out of use during a certain time. This fact means a loss of bandwidth in the network and, therefore, a decrease in the throughput that the network could reach. Thus, we define the contention that the message has supported in the network as the number of links that the message maintained occupied without sending information through them, multiplied by the time that it occupied them.

Contention is related to message delay. However, contention also means the number of links that a message maintain occupied. It is not the same if a message supports a given delay having reserved only one link, as if having reserved various links. In the second case, the negative effect caused in the network is greater even though the blocking time is the same. According to this definition, message contention will be measured in busy links times clock cycles that this message has occupied those links. Obviously, message and network size will influence in the contention. In an ideal network, or a lightly loaded network, contention will be reduced or almost null. On the opposite side, for a more loaded network or one with a difficult communication pattern (that is, that causes many messages going through the same zone of the network), the contention of each message will grow and could become very high. One of the communication patterns that is going to cause more contention in the network is that of hot-spots. Because of this, this pattern has been chosen to show the suitability of dynamic reconfiguration as a solution to the

problem of network contention. As we will show later, a reconfigurable network offers more advantages than other solutions used until now, such as virtual channels or adaptive routing.

With dynamic reconfiguration, an attempt is made to physically move the nodes from a congested zone in the network to a less congested zone. In this way, the messages that arrive at this node will support less contention and, therefore, the network will have a greater throughput. Movement of nodes in the network is possible thanks to the fact that the nodes are connected through a crossbar, and in the crossbar the links between the nodes can be modified. In the next section, we show a feasible implementation of this structure. The idea of a reconfigurable network is the following: when messages arriving through a given channel to their destination node have supported a large contention (and, therefore, a long delay), the reconfiguration algorithm will try to move this destination node to another place that produces less contention, by exchanging its position with its neighbour node in that direction. The different details of the reconfiguration algorithm can be found in [9]. In what follows, we will concisely describe how it operates so that the range of results that we are going to present here can be properly evaluated. The first aspect to take into account is that the nodes are only permitted to change with one of their neighbours, so in this way, a great disturbance in the network is not produced, and the results obtained are much better. This implies that for a node to reach a determined position in the network, on occasions various changes to arrive at that location are necessary.

3.2. The reconfiguration algorithm

A reconfigurable network is controlled by a reconfiguration algorithm. We show an updated version of the algorithm presented in [9]. A node determines the convenience of changing itself taking into account the information that it receives about contention in the network. This information is obtain of the messages arriving to it. This algorithm is controlled by a cost function that measures contention suffered by messages that arrive at that node. The reconfiguration algorithm that we have developed has four thresholds which control a greater or smaller number of the changes. The thresholds depend on various factors, among which are the size of the network, the size of the internal buffers of the router, etc. To handle the network reconfiguration, we have developed a reconfiguration protocol among the nodes and the control node. The control node is responsible to modify the network topology, adapting it to the new circumstances. Before it, the control node must to check several details in order to perform a safe change. Since the reconfiguration algorithm is distributed, it is possible that, at the same time, various reconfigurations in different places are produced in the network. Although, theoretically this is possible, since in practice the number of reconfigurations is small, this fact is only produced on very rare occasions.

Another detail to take into account is the following. When a pair of nodes interchange their positions, messages cannot pass through them because a mixture of flits from different messages would be produced. It is necessary to create a security zone around the nodes that are going to make the change, that comprises the neighbouring nodes. This security zone will not allow any message to enter the zone that is going to be reconfigured. With this, the mixture of flits from different messages and the loss of information is avoided. As it can be appreciated, this security zone will block other messages during the time that the reconfiguration takes. Because the number of changes is small, this fact does not affect the network, noticing, on the other hand, the large positive effect of a node having changed its position. Logically, all those messages that are circulating through other zones of the network not affected by the change, continue their normal course while the change is being made. For the messages that are already on their route, it is possible that in some cases the problem of deadlock could occur, even though the routing algorithm is deadlock-free (dimension-order increasing routing). Due to the change of position of the nodes, cycles in the messages can be produced. The solution adopted for those messages is the following: in the intermediate node in which the header of the message remains blocked, we see if the message needs to go through another lower dimension again, through which it has already passed. Then, and to avoid the possibility of deadlock, all the message is stored in this node and removed from the network. Once stored, it is routed again, guaranteeing then that there is no possibility of deadlock by the routing algorithm itself. In the case that the changing in the network does not mean that the message has to go through to a lower dimension again, it is not stored and continues its normal course. In this situation, the change effected has not introduced any possibility of deadlock. As it is logical, the action of a change implies a cost in time and resources of the network. In this paper, the cost has not been taken into account, considering the ideal change with zero cost.

A reconfigurable network is very well suited for parallel applications where communication pattern varies over time. An example is the triangularization of sparse

matrices by means of fast Givens rotations. In [13] we show the improvement of the performance of the Givens rotations algorithm for sparse matrices using a reconfigurable multicomputer. This algorithm is used very much in many scientific applications, such as lineal system resolution or eigenvalue problems.

4. Performance Evaluation of a Reconfigurable Multicomputer

In this section, we are going to show the results obtained in a distributed memory system with reconfigurable topology for the problem of hot-spots. The results have been obtained by simulation, by means of a tool that we have developed named Pepe [10]. To continue, we will detail the main features of Pepe and the parameters that we have taken into account in our simulations.

4.1. Programming and simulation environment

Pepe takes a parallel program as input and generates an intermediate code for its execution on a distributed memory system. The most important parameters of this system can be varied by the user. Pepe generates performance estimates and quality measurements for the interconnection network. Pepe has two main phases and several modules within it. The first phase is more language-oriented, and it allows us to code, simulate and optimize a parallel program. The second phase has several tools for mapping and evaluating the reconfigurable architecture. We can vary several parameters such as different interconnection topologies or routing algorithms. The last module is properly the network simulator. It is an improved version of a previous simulator [7] that supports network reconfiguration. It can simulate at the flit level different topologies and network sizes up to 16K nodes. Our simulator allows us to choose the time that the router needs to route a message header, the time for transferring a flit through the switch and the bandwidth of physical channels. The network reconfiguration is transparent to the user, being handled by the reconfiguration algorithm executed as part of the run-time kernel of each node. The reconfiguration algorithm decides when a change must be carried out by means of a cost function. The network reconfiguration is carried out in a decentralized way, that is, each node is responsible for trying to find its best position in the network depending on the model of communication. Also, reconfiguration is limited, preserving the original topology. Finally, we want to have a small number of changes to keep the reconfiguration cost low. There are several different parameters [9] that can be varied to adjust how the reconfiguration is performed. The quality of each reconfiguration is measured by the simulator.

4.2. Performance measures and description of our problem

The most important performance measures obtained with our environment are delay, latency, throughput and contention. Delay is the additional latency required to transfer a message with respect to an idle network. An idle network means a network without message traffic and, thus, without channel multiplexing. It is measured in clock cycles. The message latency lasts since the message is injected into the network until the last flit is received at the destination node. Throughput is usually defined as the maximum amount of information delivered per time unit and per node. It is usually measured in flits per clock cycles and per node. Node contention is the sum of the contention of all messages that have arrived at this node. It is measured in links times clock cycles. This value gives an idea of the amount of links that have been maintained occupied without sending information, due to blocked messages. These parameters will help us evaluate the behaviour of a reconfigurable network for the problem of hot-spots in a system with distributed memory. The situation that is going to be simulated is the following: Let us consider a multicomputer with a uniform distribution of message destinations. In this message pattern, message destinations are randomly chosen among all the nodes with the same probability. At a given moment, and with the network in steady state, the communication pattern changes, and a small number of hot-spots appear in the network. The network is divided into as many zones as there are hot-spots which have appeared in the network. All the nodes in a given zone start sending some proportion of messages to the corresponding hot-spot. The rest of the messages that they send continue being distributed uniformly among all the nodes of the network. Let α be the proportion of messages with uniform destination among all the nodes of the network, and β be the proportion of messages whose destination is the corresponding hot-spot of the zone to which this node belongs. Initially, α is 100% and β is zero, and later the value of β increases to a value between 15% and 30%. As it has been mentioned before, the change in the communications pattern could be due to, for example, the necessity of synchronization among the different nodes that are working on the same problem. As it is easy to imagine, enormous congestion is produced in the network, due to the great

number of messages that want to reach the hot-spot node. This causes the appearance of contentions in the network and the consequent delays, therefore degrading the performance of the network. Our objective with a reconfigurable network is to improve the performance and increase the throughput of the network.

4.3. Parameters of simulations

In the simulations that will be presented next, the following parameters have been used: the network topology is 2-D torus with 256 nodes (16 in each dimension). The proportion of messages at the hot-spot (β) has been fixed at 20%. For each simulation run, we have considered that message generation rate is constant and the same for all the nodes. Each simulation was run until the network reached steady state, that is, until a further increase in simulated network cycles did not change the measured results appreciably. Once the network has reached a steady state, the flit generation rate is equal to the flit reception rate (traffic). The number of hot-spots taken for this number of nodes has been 2 ("a" and "b"), dividing therefore the network into two halves, each one sending messages to one of the hot-spots. The size of the messages that circulate through the network is the normal in multicomputers, that is, long messages [2] fixed and equal to 256 flits. Two hot-spots could situate themselves together in any randomly determined position because their position in the network has no significant influence. From now on, the figures have been plotted for the two hot-spots together, spot "a" in position 127 and spot "b" in position 128. The nodes in the upper half of the network (initial positions 0-127) send their proportion of messages to hot-spot "a", and the lower half (the rest of the nodes) to hot-spot "b". Our network uses the dimension order deterministic algorithm proposed in [5] for the k-ary n-cube, modified so that it uses bidirectional channels with two virtual channels per physical channel. Dimension order routing routes a packet successively in each dimension, until the distance in that dimension is zero, then proceeds to the next dimension. Another selected parameters are the following. The router takes one clock cycle to compute the output channel, the switch takes one clock cycle to transfer a flit through the crossbar and the bandwidth of physical channels is equal to size messages (in flits) per clock cycle.

4.4. Evaluation results

In this section, we present the results we have obtained with a reconfigurable network. Firstly, we show the improvement in the network contention for a determined generation rate of messages. Afterwards, we show the average message latency versus traffic when using a hot-spot pattern for message destination.

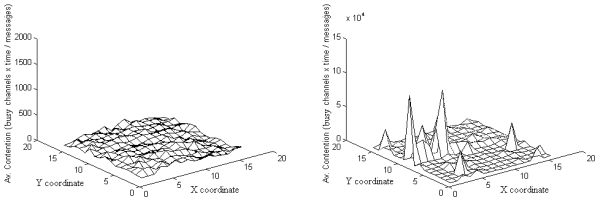

Figure 1. a) random uniform pattern b) hot-spot pattern

In figure 1, we can see how the contention of the network changes when the hot-spot communication model is introduced. In this case, a low generation rate of messages of the network capacity (25%) has been chosen. Initially (Fig. 1.a), the average contention (contention per total received messages) in each one of the nodes for the uniform communications model is shown. Since the load in the network is low, there is practically no contention in the network (note that the figures 1.a and 1.b have different scales).

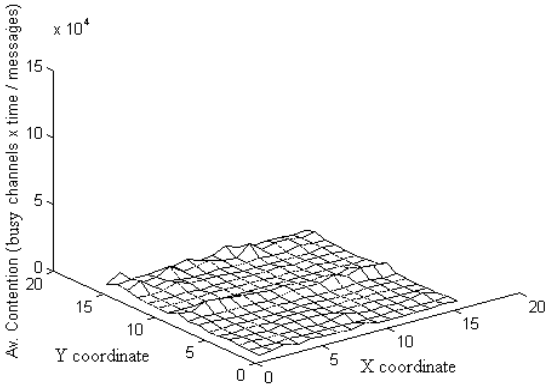

Figure 2. Contention for a reconfigurable network.

In figure 1.b a drastic increase in contention can be seen. The hot-spots and the nearest zones support high contention, predecing considerable delay in the delivery of the messages and a loss of throughput of the network (in the second case, the throughput decreases by 60%). In this figure, it can be appreciated how the contention not only affects the hot-spots, but also is extended along the network. An effect similar to *tree saturation*

is obtained as was already observed by Pfister [12] for multiprocessors. In figure 2, it can be appreciated how the contention for each one of the nodes using the reconfigurable network remains. As it can be seen, this figure presents a much better aspect than figure 1.b, being the results similar to those shown in figure 1.a, where hot-spots were not considered. We would like to emphasize there is a reduction in the contention of the network of 45%. This improvement is produced with a reduced number of changes in the network (here there have been approximately 60 changes). The final position of the hot-spots "a" and "b" search for the centre of the zone of the nodes that send them the messages. That is, the node "a" searches for position 64 in the network, while the node "b" searches for position 192.

Now, we are going to evaluate the average message latency versus traffic when using a hot-spot model for message destination. For this model, we evaluate the deterministic routing algorithm and the dynamic reconfiguration algorithm. Also, with comparative purposes, a reconfigurable network must be compared to known techniques such as virtual channels or adaptive algorithms.

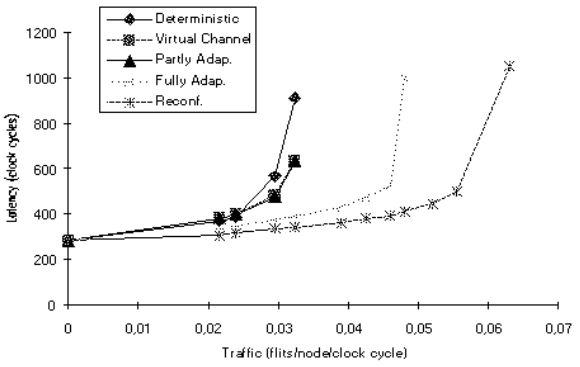

Figure 3. Average message latency versus traffic for hot-spot traffic pattern.

Since the dimension order deterministic algorithm for 2-D torus needs 2 virtual channels per physical channel to be deadlock-free, the technique of virtual channels has been implemented with 4 virtual channels per physical channel. In the case of the adaptive algorithms, the methodology proposed by Duato [7] has been followed assuring that routing algorithms are deadlock-free, and the partially adaptive algorithm and the fully adaptive algorithm has been evaluated. The evaluation of these algorithms for both, uniform and non uniform message load can be found in [8]. In summary, these algorithms work in the following manner. The partially adaptive routing algorithm only requires two virtual channels per physical channel (as the same that the deterministic one). Channels can only be used crossing dimensions in ascending order, and a higher priority has been assigned to a one class of virtual channels in order to favour messages with less routing choices. The fully adaptive algorithm starts from the partly adaptive one adding a third virtual channel to each physical channel. This third virtual channel is used for fully adaptive routing, crossing dimensions in any order following a minimal path. This channel, inside each dimension, has higher priority than the original ones. Usually, adaptive algorithms requiere a lower clock frecuency [3] because of a more complex switch and router. However, in our simulations all the routing algorithms have the same clock rate.

Figure 3 shows the average message latency versus traffic. Figure 3 compares four usual routing algorithms and also the reconfigurable architecture. As it can be appreciated, the best results both for traffic and for latency are obtained in a dynamically reconfigurable network. These results are even better than those obtained with a fully adaptive algorithm, which gives an idea of the power of the improvement proposed. For the hot-spot model, the reconfigurable architecture increases throughput by 30% over the fully adaptive algorithm, and it almost doubles throughput over the rest of the routing algorithms. The reconfigurable architecture also achieves a reduction in message latency with respect to all the proposed routing algorithms for all the range of the traffic.

The number of changes necessary to reach these results is are very small. Actually, what happens is that there is some oscillation around the final position which causes some additional changes. Adjusting more carefully the reconfiguration algorithm, the same results with a lower number of changes should be obtained.

Finally, it must be noticed that the improvement for adding virtual channels is small. Then another possibility is that more virtual channels can be used in all the routing algorithms. But delays will increase accordingly. Thus, it is not interesting [8].

5. Conclusions and Future Work

In this paper, a new way to solve the problem of hot-spots in parallel machines has been shown. The study introduced here is centred on parallel machines with a direct interconnection network, specifically for the topology of 2-dimensional torus. This study examines the techniques for using a reconfigurable topology approach to improve network performance when hot-spot situations occur.

The problem of hot-spots has been solved by means of a dynamically reconfigurable network, that is, a network that allows the nodes to change position throughout the time, depending on the communication pattern that it has. These changes of position do not alter the topology of the network so that the same routing algorithm as for a static network can be used. In this paper the dimension-order deterministic algorithm has been used.

Simulation was used to evaluate the proposed techniques under certain assumptions about the execution environment and the network structure. With comparative purposes the results obtained with other techniques used to improve the efficiency of the network in hot-spots have been shown, such as virtual channels and adaptive routing algorithms. As it can be observed in the figures presented, the reconfigurable network obtains the best results with regard to the productivity of the network, without needing a high number of changes to reach these values. The results of these simulations were reported and discussed. Note that our approach is valid for a physically distributed memory system with a message passing model.

For future work various interesting ways appear. On one hand, we would like to amplify our study over larger sized networks, such as 1024 or 2048 nodes. Besides, we would like to study the behaviour over other topologies of lower dimensions, such as mesh or 3-D torus. Lastly, we would like to study the behaviour of reconfigurable networks with other models of load and other communication patterns. As the communication pattern is highly application-dependent, we would like to evaluate another patterns with a large number of small messages. Also, the study of the behaviour of hot-spots for a network with a larger number of hot-spots remains open.

Finally, we would like to extend these results to the problem of hot-spots to shared memory machines and indirect networks. It seems to us that our results can be easily adapted to this other type of machines.

Acknowledgement

The authors are indebted to José L. Sánchez and F. J. Alfaro for their very useful comments and suggestions. The authors also thanks the anonymous reviewers for their thorough and helpful comments.

References

[1] V. S. Adve and M. K. Vernon. Performance analysis of mesh interconnection networks with deterministic routing. *IEEE Trans. on Parallel and Distributed Systems*, 5(3):225–246, March 1994.

[2] A. Agarwal. Limits on interconnection network performance. *IEEE Trans. on Parallel and Distributed Systems*, 2(4):398–412, October 1991.

[3] A. A. Chien. A cost and speed model for k-ary n-cube wormhole routers. In *Proc. Hot Interconnects '93*, August 1993.

[4] W. J. Dally. Virtual-channel flow control. In *Proc. of 17 Int. Symp. on Computer Architecture*, 1990.

[5] W. J. Dally and C. L. Seitz. Deadlock free message routing in multiprocessor interconnection networks. *IEEE Trans. on Computers*, C-36(5):547–553, May 1987.

[6] S. P. Dandamudi and D. L. Eager. Hot-spot contention in binary hypercube networks. *IEEE Trans. on Computers*, 41(2):239–244, February 1992.

[7] J. Duato. A new theory of deadlock-free adaptive routing in wormhole networks. *IEEE Trans. on Parallel and Distributed Systems*, 4(12):1320–1331, December 1993.

[8] J. Duato and P. López. Performance evaluation of adaptive routing algorithms for k-arys n-cubes. In K. Bolding and L. Snyder, editors, *LNCS 853*, pages 45–59. Springer-Verlag, 1994.

[9] J. M. García and J. Duato. Dynamic reconfiguration of multicomputers networks: Limitations and tradeoffs. In P. Milligan and A. Nuñez, editors, *Euromicro Workshop on Parallel and Distributed Processing*, pages 317–323. IEEE Press, 1993.

[10] J. M. García, J. L. Sánchez, J. Duato, and J. Fernández. Pepe: A trace-driven simulator to evaluate reconfigurable multicomputer architectures. Technical Report DIS TR 4-95, University of Murcia, March 1995.

[11] J. Liu, K. G. Shin, and C. C. Chang. Prevention of congestion in packet-switched multistage interconnection networks. *IEEE Trans. on Parallel and Distributed Systems*, 6(5):535–541, May 1995.

[12] G. F. Pfister and A. Norton. Hot spot contention and combining in multistage interconnection networks. *IEEE Trans. on Computers*, 34(10):943–948, October 1985.

[13] J. L. Sánchez, J. M. García, and J. Fernández. Improving the performance of parallel triangularization of a sparse matrix using a reconfigurable multicomputer. In *Proc. Workshop on Applied Parallel Computing in Physics, Chemistry and Engineering Science. LNCS 1041*, pages 495–502. Springer-Verlag, 1995.

[14] A. Smai. Congestion control in wormhole networks: first results. In *Proc. of Euro-Par '95*, August 1995.

[15] M. Wang, H. J. Siegel, M. A. Nichols, and S. Abraham. Using a multipath network for reducing the effects of hot spots. *IEEE Trans. on Parallel and Distributed Systems*, 6(3):252–268, March 1995.

[16] P. C. Yew, N. F. Tzeng, and D. H. Lawrie. Distributing hot-spot addressing in large scale multiprocessors. *IEEE Trans. on Computers*, C-36(4):388–395, April 1987.