On the Performance of Up*/Down* Routing*

Francisco J. Alfaro[1], Aurelio Bermúdez[2], Rafael Casado[2],
José Duato[3], Francisco J. Quiles[2], José L. Sánchez[2]

[1] Department of Computer Engineering and Technology
University of Murcia
30071- Murcia, Spain
falfaro@ditec.um.es
[2] Department of Computer Science
University of Castilla-La Mancha
02071- Albacete, Spain
{abermu, rcasado, paco, jsanchez}@info-ab.uclm.es
[3] Department of Information Systems and Computer Architecture
Technical University of Valencia
46071- Valencia, Spain
jduato@gap.upv.es

Abstract. Networks of Workstations (*NOWs*) are usually arranged as a set of interconnected switches with hosts connected to switch ports through interface cards. Several commercial interconnects for high-speed NOWs use up*/down* routing. Every time the network is powered on or the topology is changed, a configuration algorithm is executed, which provides information about the topology and generates a directed graph. Routing tables are computed from this directed graph. There are several ways to obtain the directed graph. The most frequent way is by means of algorithms based on minimum-depth spanning-trees (*MDST*) or propagation-order spanning-trees (*POST*). This paper shows that, for most networks, graphs obtained by means of these methods can be improved in order to achieve higher network performance.

1 Introduction

High-speed NOWs, like Autonet [14], Myrinet [1] and Servernet [9], provide an inexpensive alternative for building parallel computers with high communication bandwidth and reduced latency. These networks usually consist of a set of switches connected by point-to-point links, and of hosts linked to switches through a network interface card (NIC).

Up*/down* routing is broadly used for routing in high-speed NOWs [14, 1]. However, it is well known that the performance of up*/down* routing is strongly dependent on the network topology [10]. The generation of up*/down* routing tables requires the construction of a directed graph by assigning directions to the

* This work was partly supported by the Spanish CICYT under Grant TIC97-0897-C04, and Caja Castilla-La Mancha

operational links in the network. As we will see later, this direction assignment has a strong impact on network throughput.

There are several ways of assigning link direction through the construction of a directed graph: minimum-depth spanning-tree (MDST) [14], propagation-order spanning-tree (POST) [12], etc. This paper shows that these methods do not produce the best graph, and that it is possible to find a link direction assignment that makes the network achieve a higher throughput.

Our main goal is to compare the behavior of directed graphs generated by traditional methods with the behavior of the best directed graph that we can obtain.

In Section 2 we describe the up*/down* routing algorithm used by Autonet and Myrinet as well as two methods used by Autonet to obtain the directed graph: MDST and POST. Section 3 presents an example that shows how performance can be improved and presents a genetic algorithm that searches for the best possible directed graph for a given network. Section 4 compares the performance of the network when using the traditional approach and when using the optimized directed graph.

2 Background

2.1 Up*/down* Routing and Correct Graphs

Up*/down* routing [14] is a partially adaptive deadlock-free routing algorithm suitable for regular and irregular topologies [7]. Up*/down* routing requires assigning a direction to all the links of the network. In every link, a direction is named *up* and the opposite one is named *down*. Legal routes never use a link in up direction after having used one in down direction, in order to avoid deadlocks. Messages can cross zero or more links in up direction, followed by zero or more links in down direction. By doing so, cycles in the channel dependency graph [5] are eliminated and deadlock is avoided.

In order to use up*/down* routing, the directed graph which represents the link direction assignment must satisfy some properties [3, 2]. A graph satisfying those properties will be referred to as a *correct graph*. Those properties are the following:

1. A *root node* is a node in a directed graph that is not the source of any arc. The up*/down* routing algorithm requires the existence of a single root node in the graph. The reason is that there are no up*/down* legal routes between two root nodes, since each possible route requires down to up transitions. This restriction is required for the routing algorithm to provide paths between any pair of nodes.
2. The graph must be acyclic to avoid deadlock.

Thus, every link direction assignment leading to an acyclic directed graph that contains a single root node is a *correct graph*. It is important to remark that no more restrictions are imposed on the directed graph. Therefore, several correct directed graphs can be obtained starting from the same topology.

2.2 MDST and POST Directed Graphs

As mentioned above, the generation of up*/down* routing tables requires the construction of a correct directed graph of the network in which nodes are the active network switches and arcs are the links among switches. Two of the methods used for obtaining this directed graph are minimum-depth spanning-tree (MDST) [14] and propagation-order spanning-tree (POST) [12].

Both methods are based on the execution of a distributed algorithm that builds a spanning-tree from the network graph. Starting from this spanning-tree, the algorithm assigns directions to all the links of the network. In particular, the "up" end of each link is defined as: 1) the end whose switch is closer to the root in the spanning-tree; 2) the end whose switch has the lower identifier (*id*), if both ends are at switches at the same tree level. The MDST and POST methodologies are based on a breadth-first searching methodology [12].

On the one hand, MDST produces the minimum-depth spanning-tree of the network, where the root node is the switch with the lowest *id* and the rest of switches are assigned positions in the tree that provide the shortest distance to the root. If there are several positions that provide the shortest distance to the root, then the one provided by the switch with the lowest *id* is selected. Thus, it is possible to guarantee that there is only one possible minimum-depth spanning-tree for a network topology and an accompanying identifier assignment.

Figure 1 shows a network topology and the *id* for each switch. The resulting directed graph using MDST is shown in Figure 2. Note that the link between Node 14 and Node 10 is the only one that does not belong to the spanning-tree.

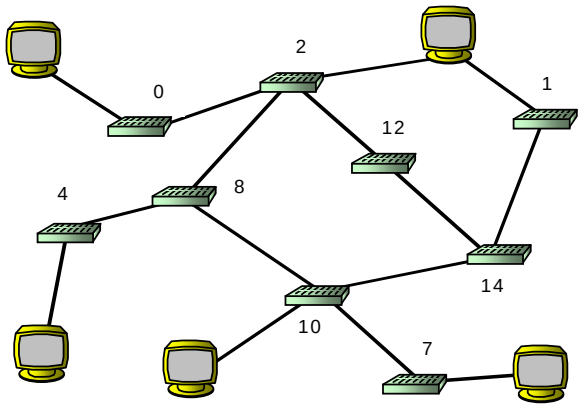


Fig. 1. An example of irregular network

On the other hand, POST chooses as the root node of the spanning-tree the switch that detects the change in the topology and starts the reconfiguration process. This node is called the *initiator*. The initiator sends messages to its

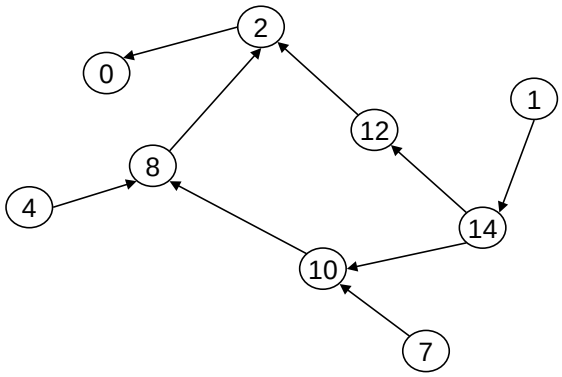


Fig. 2. Directed graph for the network in Figure 1 obtained by using MDST method

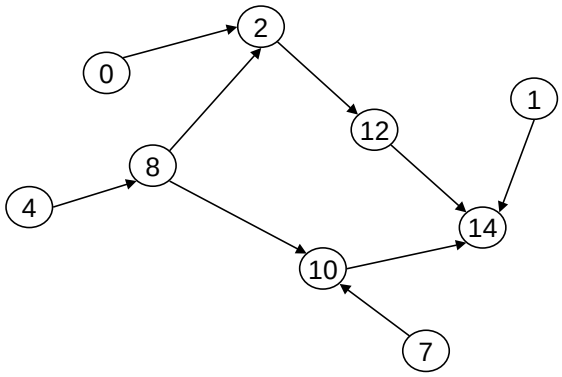


Fig. 3. Directed graph for the network in Figure 1 obtained by using POST method

neighboring switches. When a switch receives a control message, it join the tree and propagates new control messages to its own neighboring switches (the switch that sent it the incoming message is omitted). In this way, the switches will join the tree as soon as they receive messages from other switches.

Note that the resulting spanning-tree depends on the initiator switch, but also on the propagation order of the reconfiguration process through the network.

Figure 3 shows a possible directed graph for the network shown in Figure 1 using POST and assuming that the initiator is Node 14.

3 Improving Network Performance

As we will see in Section 4, network throughput has a direct relationship with the directed graph used to build the up*/down* routing tables. In other words, network throughput will be determined by the direction assigned to the network links.

Figure 4a shows a network with a link direction assignment provided by the MDST method. The up*/down* routing rules lead to a bottleneck at Node 0 (the root node of the tree), because messages are not allowed to cross Node 3 in order to move from the right subtree to the left one, and vice versa. If we change the direction of link 1 (see Figure 4b), we might achieve an improvement in network throughput of about 100%. The reason is that now messages can also cross from one subnetwork to the other through Node 3.

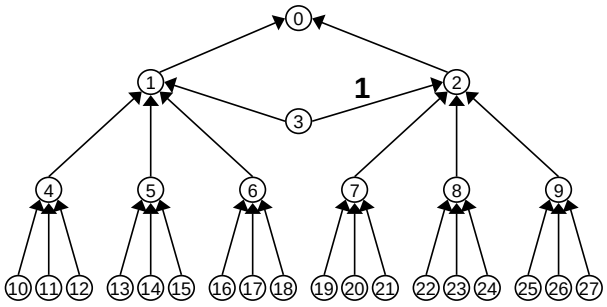
If we study the directed graph generated by Autonet, we will see that in most cases it is not the one that achieves the highest network throughput. Our purpose is to determine how suboptimal the directed graph obtained by MDST and POST is, comparing the resulting network performance with that obtained by using the best possible directed graph.

Each possible graph can be obtained from another graph by changing the direction of some links of the network. Therefore, the number of possible variations for a certain network is $N_g = 2^{links}$, where *links* is the number of links in the network and the number of correct graphs is $N_c = 2^{links}$ - {incorrect graphs}. The number of incorrect graphs is much smaller than the number of correct ones. Thus, even for a relatively small network with a few links, the number of directed graphs to study is too high to recommend the use of an exhaustive search method.

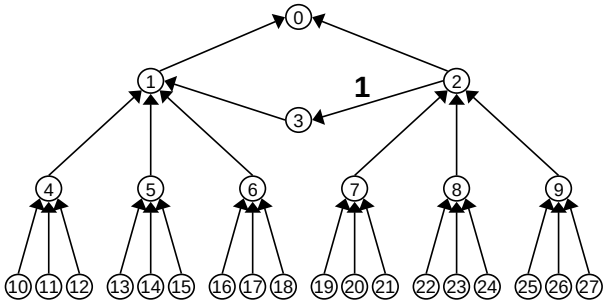
3.1 Genetic Algorithms

Genetic Algorithms (GAs) [8] are adaptive methods used to solve search and optimization problems in wide or complex spaces. They are based on the genetic behavior of the living beings and the natural evolution and the adaptation of these beings alongside successive generations. We will make use of GAs to search for the best (or near-optimal) possible directed graph.

GAs work with a population of t individuals, where each one represents a feasible solution to the given problem. An adaptation value is associated to each



(a)



(b)

Fig. 4. An example of throughput improvement by changing only the direction of link 1

individual, which indicates its capacity to survive. The individuals of a generation that better adapt to the requirements of the problem will have a higher probability of surviving and reproducing in the following generation than the worse adapted individuals. This reproduction leads to the perpetuation of their genes in substitution for the genes of the weakest individuals. Thus, the individuals of next generations will be better and better adapted to the conditions of the problem.

Each individual is represented by means of a chain of values, or *chromosome*, which contains the necessary information to encode the values for the important problem parameters. When the reproduction of the best adapted individuals for a generation takes place, a series of genetic operators are applied to their chromosomes. The most usual genetic operators are: *cross*, which picks up two

parents and mixes their chromosomes starting from a randomly generated gene or position, and *mutation*, which consists of randomly alterating some of the parent's genes. The individuals that are born in generation i are evaluated together with the survivors of generation $i-1$ and, amongst them, the best adapted ones will reproduce to give rise to generation $i+1$.

3.2 GAs and NOWs

In our research, we will consider each possible correct directed graph as individuals belonging to the same population. Individuals of the same population will then be distinguished from each other according to the direction of their links.

The form in which a directed graph is encoded in the chromosome for its later treatment has a great importance, and it must be unique. We represent a link between nodes i and j ($i < j$) with a gene in the chromosome, being equal to 1 if the link from node i to node j is in the up direction, and equal to 0 if it is in the down direction. The place that each link occupies in the chromosome is called a *locus* and it is defined by assigning a number to the links, beginning from port 0 of Node 0 and continuing in increasing order. Note that links are numbered only the first time they are evaluated. Therefore, their locus is fixed. Figure 5 shows an example of link numbering and the associated chromosome.

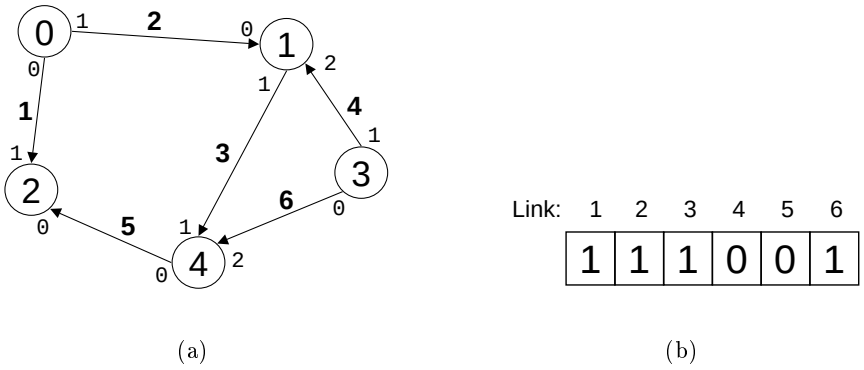


Fig. 5. An example encoding of the network (a) in the chromosome (b). Small numbers next to each node represent port numbers

The way in which we measure the adaptation of the individuals of a generation is by using the throughput achieved by the routing algorithm that they represent. Throughput is defined as the maximum amount of information that the network is able to transmit per unit of time. We obtain this value by simulation.

Two important parameters for the GA are the number of individuals in a generation, and how many of them die in order to give way to the following generation. We have seen that the best results are obtained by considering a population of 12 individuals in which the four worst adapted die. Four new individuals will be born in the next generation by applying the genetic operators cross and mutation to the two best individuals of the current generation. Obviously, these new individuals must represent correct graphs.

4 Performance Evaluation

In this section, we show the results of our comparison. We have evaluated the performance of MDST and POST directed graphs and the performance of the directed graph obtained using the GA described in the previous section through the evaluation of the associated up*/down* routing algorithm.

GAs have an inherent non-determinism. For this reason, we have evaluated how stable the results provided by them are. To achieve it, we have obtained the entire set of possible correct directed graphs for several small networks (with 8 switches). For each topology, the directed graph provided by the GA obtained the best throughput.

Simulation was used to evaluate network performance. Our simulators model the network at the phit[1] level. The evaluation methodology used is based on the one proposed in [6].

4.1 Network Model

The network consists of a set of switches as well as hosts connected to them. We used a set of irregular randomly generated topologies. Also, we have evaluated hierarchical topologies like the example in Figure 4. We have evaluated networks with 8, 16, 24, 32, 48, and 64 switches.

4.2 Packet Generation

We have considered uniform traffic. Message destination is randomly chosen amongst all the hosts in the network, and the average generation rate is constant and the same for all the hosts. For packet length, 16-byte and 64-byte packets were considered. Nevertheless, we have found that packet length does not significantly affect the relative performance of the routing algorithms computed from different directed graphs. Thus, we will only show the results for short packets.

[1] Unit of information that can be transferred across a physical channel in a single step or cycle.

4.3 Simulation Results

As indicated above, we use network throughput to evaluate the optimality of the directed graph. Tables 1a, 1b, 1c, and 1d show the throughput achieved by several networks of size 8, 16, 24, and 32 switches, respectively. First, we can see that the throughput obtained when using our GA is always equal to or greater than that obtained by using the other methods. For networks with 8 switches, there is no difference. However, for larger networks, we can observe increments of 50%, 60%, and even 80% in network throughput. For networks with 48 switches we have obtained increments of up to 24% with respect to MDST and up to 80% with respect to POST, and for networks with 64 switches the increment reaches 58% with respect to MDST and 78% with respect to POST.

We have also evaluated the average message latency versus traffic for some networks with 24, 32, and 64 switches. Figures 6, 7, and 8 show the results. As can be observed, in all the cases the routing algorithm obtained using GAs reduces latency and increases throughput with respect to the MDST and POST methods.

Network	MDST	POST	GA
A	0.469	0.468	0.469
B	0.469	0.468	0.469
C	0.469	0.468	0.469
D	0.328	0.328	0.328
E	0.328	0.328	0.328

(a) 8 switches

Network	MDST	POST	GA
A	0.180	0.183	0.203
B	0.180	0.175	0.180
C	0.469	0.412	0.469
D	0.406	0.329	0.469
E	0.328	0.295	0.359
F	0.203	0.223	0.328
G	0.203	0.199	0.234

(b) 16 switches

Network	MDST	POST	GA
A	0.164	0.185	0.297
B	0.148	0.148	0.234
C	0.180	0.161	0.203
D	0.164	0.141	0.164
E	0.117	0.117	0.133
F	0.133	0.122	0.133
G	0.117	0.117	0.117
H	0.203	0.157	0.234
I	0.133	0.151	0.203

(c) 24 switches

Network	MDST	POST	GA
A	0.102	0.096	0.102
B	0.102	0.102	0.133
C	0.102	0.100	0.102
D	0.148	0.190	0.328
E	0.234	0.228	0.469
F	0.180	0.245	0.406
G	0.180	0.226	0.359
H	0.297	0.249	0.406
I	0.234	0.252	0.406
J	0.203	0.237	0.359

(d) 32 switches

Table 1. Throughput for several irregular networks with 8, 16, 24, and 32 switches

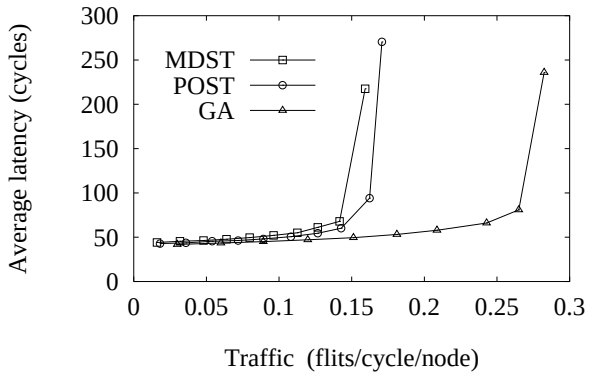


Fig. 6. Average message latency versus traffic for an irregular network with 24 switches

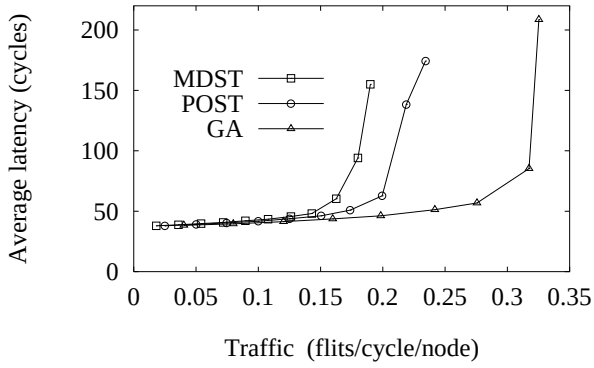


Fig. 7. Average message latency versus traffic for an irregular network with 32 switches

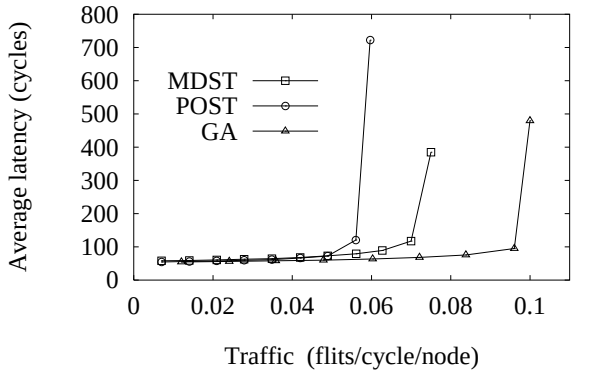


Fig. 8. Average message latency versus traffic for an irregular network with 64 switches

We have analyzed the directed graphs obtained by MDST, POST and GAs. The performance improvement obtained by using GAs comes from a reduction in the number of routing restrictions required to avoid deadlock. In those cases that does not exist any better set of routing restrictions, all the strategies obtained the same results.

5 Related Work

Two approaches that improve performance by obtaining a better directed graph are proposed in [10, 13]. The former is suitable for regular topologies and is based on a previous appropriate assignment of switch identifiers (*id*). The latter is suitable for general topologies including irregular ones. In this case, the directed graph requires the previous generation of a deep-first tree (DFT), instead of a breadth-first spanning-tree, like MDST or POST.

Another deadlock-free routing schemes have been proposed for NOWs, like the adaptive-trail routing algorithm [11], the minimal adaptive routing [15], and the smart-routing [4].

6 Conclusions

In this paper we have shown that changing the direction of some links in the directed graph used for up*/down* routing has a significant impact on the throughput achieved by the network. We conclude that there are correct graphs that offer a higher network throughput than the graphs generated by usual methods, such as MDST and POST, which are used in Autonet and Myrinet. Also, differences in network performance for GA, MDST and POST increase with network size.

In order to obtain the best possible directed graph, we have used a search and optimization tool based on genetic algorithms, which are a very good choice when the search space is wide and complex. We have defined a representation of the NOWs for the GA. Also, we have used throughput as the adaptation metric.

7 Future Work

Obviously, GAs are not suitable for computing routing tables during a reconfiguration process due to the very high computational cost. As future work, we plan to study the best networks obtained, and try to characterize them. We will attempt to propose an efficient design methodology that obtains the best graph when a reconfiguration takes place, thus maximizing network throughput.

8 Acknowledgements

The authors thank *Timothy M. Pinkston*, at the University of Southern California, for his suggestions and comments on drafts of this paper. We also thank *Juan Luis García-Navarro* and *José A. Gámez-Martín* for their collaboration with genetic algorithm simulations.

References

1. N.J. Boden, D. Cohen, R.E. Felderman: Myrinet – a gigabit per second local area network. IEEE Micro, pages 29–36, February 1995.
2. R. Casado, A. Bermúdez, F.J. Quiles, J.L. Sánchez, J. Duato: Performance evaluation of dynamic reconfiguration in high-speed local area networks. In Proceedings of the sixth Symposium on High Performance Computer Architecture (HPCA-6), January 2000.
3. R. Casado, F.J. Quiles, J.L. Sánchez, J. Duato: Deadlock-free routing in irregular networks with dynamic reconfiguration. In Lecture Notes in Computer Science, vol. 1602, pp. 165–180. Springer, January 1999, Proceedings of the CANPC'99.
4. L. Cherkasova, V. Kotov, T. Rockicki: Fibre channel fabrics: evaluation and design. In Proceedings of 29th Hawaii International Conference on System Sciences, February 1995.
5. W.J. Dally, C.L. Seitz: Deadlock-free message routing in multiprocessor interconnection networks. IEEE Transactions on Computers, C-36(5), May 1987.
6. J. Duato, A. Robles, F. Silla, R. Beivide: A comparison of router architectures for virtual cut-through and wormhole switching in a NOW environment. In Proceedings of the 13th International Parallel Processing Symposium and 10th Symposium on Parallel and Distributed Processing (IPPS/SPDP'99), April 1999.
7. J. Duato, S. Yalamanchili, L. Ni: Interconnection networks. An engineering approach. IEEE Computer Society, 1997.
8. D. E. Goldberg: Genetic Algorithms in search, optimization and machine learning. Addison-Wesley, 1989.
9. R. W. Horst: Tnet: A reliable system area network. IEEE Micro, February 1995.
10. S. Owicki, A. R. Karlin: Factors in the performance of the AN1 computer network. Technical Report 88, Digital Equipment Corporation Systems Research Center, Palo Alto, CA, June 1992.
11. W. Qiao, L.M. Ni: Adaptive routing in irregular networks using cut-through switches. In Proceedings of the 1996 International Conference on Parallel Processing (ICPP'96), August 1996.
12. T.L. Rodeheffer, M.D. Schroeder: Automatic reconfiguration in Autonet. Technical Report 77, Systems Research Center of Digital Equipment Corporation, September 1991.
13. J.C. Sancho, A. Robles, J. Duato: A new methodoly to compute deadlock-free routing tables for irregular networks. In Proceedings of the WorkShop on Communication and Architectural Support for Network-Parallel Computing'00 (CANPC'00), January 2000.
14. M.D. Schroeder, A.D. Birrell, M. Burrows, H. Murray, R.M.Needham, T.L. Rodeheffer, E.H.Satterthwate, C.P.Thacker: Autonet: a high-speed, self-configuring local area network using point-to-point links. Technical Report 59, Systems Research Center of Digital Equipment Corporation, 1990.
15. F. Silla, J. Duato: Improving the efficiency of adaptive routing in networks with irregular topology. In Proceedings of the 1997 Int. Conference on High Performance Computing (HiPC'97), December 1997.