AN ANALYSIS OF DEADLOCK RISK DURING CENTRALIZED NETWORK MAPPING *

P.J. GARCIA, F.J. QUILES, F.J. ALFARO, J.L. SANCHEZ
Dept. de Informática
Universidad de Castilla-La Mancha
Escuela Politécnica Superior - 02071- Albacete, Spain
Tel.: (+34) 967 599200 - Fax: (+34) 967 599224
e-mail: {pgarcia, paco, falfaro, jsanchez}@info-ab.uclm.es

J. DUATO
Dept. de Informática de Sistemas y Computadores
Universidad Politécnica de Valencia
Camino de Vera, s/n - 46071- Valencia, Spain
e-mail: jduato@gap.upv.es

ABSTRACT
Modern high-performance interconnection networks implement mechanisms for obtaining explicit topology information. One of these mechanisms is based on a host that explores the network in a centralized way, using "scouting" messages, for discovering the network topology. In this paper, we analyze the risk of deadlock due to such centralized mapping processes in a source-routing network, also evaluating the probability of deadlock configurations and the impact of those deadlocks on network performance.

KEY WORDS
Interconnection Networks, Topology Detection, Mapping, Deadlock

1 Introduction

A key issue for the performance of switch-based interconnection networks is the technique used to route the packets. In these networks, packets must follow specific paths, crossing several point-to-point links and switches to travel from their source host to their destination.

Many routing algorithms have been proposed for establishing the paths packets should follow in switch-based networks [1, 2, 3]. A desirable feature in these algorithms is to guarantee deadlock freedom. This property assures that packets will not be blocked forever in the network. Usually, deadlock freedom is achieved by imposing routing restrictions [4, 5]. So, for every source-destination pair, only a subset of all the possible paths is allowed. The implementation of routing criteria and restrictions varies depending on the network characteristics. In networks with distributed routing, switches make routing decisions, but if source routing is used, these decisions are made by the hosts [6]. In the last case, the source host determines every output channel of the route and each switch must only forward the packet to the port indicated by its header. Therefore, hosts need information about the network topology. For regular networks, it is not needed much information because the topology is defined by the connection pattern and dimensions of the network, but in the case of irregular networks, an explicit knowledge of the topology is needed.

Most modern high-performance interconnection networks are switch-based irregular networks. So, the mechanism used to obtain topology information becomes an essential matter in their design. In fact, some of the most popular high-performance networks implement "self-configuration" (or reconfiguration) mechanisms [6, 7, 8]. In these cases, the network can detect its own topology without external help and uses this information to compute, according to some algorithm, the routes that guarantee the proper routing of packets. So, the "auto-detection" mechanism is the initial phase of reconfiguration processes. Specifically, we have analyzed an auto-detection mechanism proposed for source-routing networks with "dumb" switches that was implemented in Myrinet [6, 9]. When this mechanism is used, the topology is detected by means of centralized mapping processes. Such processes are based on control messages that explore the whole network. We have found that this mapping may cause deadlock configurations because "scouting" messages are not subject to the routing restrictions imposed to user traffic.

In this paper, we explain how deadlock may occur due to centralized mapping processes. We present simulation-based results that show the probability of deadlock for different network sizes and traffic loads. We also present results showing the impact of deadlocks on the network performance when deadlock recovery techniques are used for "breaking" those deadlocks. The rest of the paper is organized as follows: In Section 2 we review the basics of centralized mapping mechanisms. In Section 3 we explain why deadlock configurations are possible during the mapping process and the consequences of such deadlocks. The simulation methodology and results are presented in Section 4. Finally, some conclusions are given in Section 5.

2 Centralized Mapping Mechanism

Although the mapping mechanism described in the following paragraphs was proposed for being implemented

*This work was partly supported by the Spanish CICYT under Grant TIC2000-1151-C07 and by Junta de Comunidades de Castilla-La Mancha under Grant PBC-02-008.

on Myrinet networks [6], it could be used on any network with similar characteristics. So, we assume that the mapping mechanism will be applied in a network that uses source routing, wormhole switching [4] and "dumb" switches (they have neither processors nor routing tables). We also assume that the topology of the network can be irregular and may change at any time, so the routing tables can not be generated[1] (or updated) without detailed information about the specific network topology. Therefore, some mechanism is needed for discovering the topology when required. To do that, a centralized mapping mechanism based on scouting messages can be implemented.

The mapping mechanism is centralized because at a given time only one host will be performing this task. This host is called "mapper" because it must elaborate a "map" reflecting the network topology. In order to obtain the topology, the mapper explores the network sending scouting messages and receiving replies to them. Basically, a scouting message is sent for exploring a port if the mapper does not know which component is connected to this port.

First, the mapper will send to the unexplored port a message for detecting a host. If the "unknown" element connected to the port is really a host, it will receive the message. Once processed this message, the "detected" host will send a reply (containing its identifier) to the mapper[2]. If the mapper receives the reply, the host will be considered as detected and added to the map, where the received identifier will allow to distinguish this host from formerly detected hosts. The mapper will wait for a reply until a fixed timeout expires. If this happens and the reply has not been received, the mapper will send the same "host-detecting" message again, and so until the mapper has waited a fixed number of times for the reply. Only then the mapper will assume that the "unknown" element is not a host.

In this case, the mapper will send to the port a message for detecting a switch. These messages are different from the "host-detecting" ones because a "dumb" switch can not generate messages, so it can not reply to a scouting one. To solve this problem, a "switch-detecting" message would follow a cyclic route. So, the message will follow a path that leads to the port to explore and, if there is a switch connected to this port, the message will cross it in order to come back to the mapper following the reverse path. If the mapper receives a "switch-detecting" message, the switch will be added to the map. It will be identified by a switch number assigned by the mapper. Like "host-detecting" messages, "switch-detecting" messages will be sent again (after a timeout period and a maximum of times) if they do not return to the mapper. If no element is detected, the exploration of the port will finish and it will be considered disconnected. When a switch is detected, all its ports (excepting the port where the scouting message has gone in and out) are marked in the map as connected to "unknown" elements, and so they must be explored later. The ports of a same switch are explored simultaneously. When all the ports of a switch have been completely explored, the unexplored ports of another switch will be explored.

The mapping process will continue until there are no ports to explore. Then, the network map is considered finished. The whole mapping process is repeated periodically in order to "refresh" the map. Every new map is completely redone, in such a way that the mapper builds it from scratch[3]. Therefore, an "old" map does not constrain further mapping processes, and changes in the network can be detected by comparing the resulting new map and the previous one. If both maps differ, the mapper will send the new map to other hosts in order to update their routing tables.

3 Deadlock During Centralized Mapping

After explaining the basics of the mechanism, we can focus on the problem of deadlock during mapping. First, we will explain why mapping can lead to deadlock. Later, we will analyze how such deadlocks affect network performance.

3.1 Deadlocks Due to Centralized Mapping

Whatever deadlock-free routing algorithm the network uses, the information contained in the network map will be used to compute deadlock-free routes to fill the routing tables. Deadlock-free routing algorithms guarantee deadlock-free routes by imposing routing restrictions, in such a way that not every possible path between two hosts will be allowed. So, given a topology, the application of the routing algorithm will make some possible routes "illegal", and so they will not be included in the routing tables.

As explained above, the mapper performs every mapping process independently of the former map of the network. In fact, the mapper sends scouting messages following a procedure that does not use the mapper own routing table. Therefore, these messages are sent without any routing restriction, and they can follow routes that are illegal according to the routing algorithm. This is not a problem when the network is explored by the first time, because routing tables are still empty, and no user traffic exists in the network. But, as mapping is repeated, further explorations will be performed under a different network status. In these cases, there could be messages in the network following valid routes (user traffic), but there could also be messages following illegal routes (mapping traffic). In such conditions, no deadlock freedom guarantee can be provided. For instance, deadlock can exist during the mapping of the network whose diagram[4] is shown in Figure 1.

[1] Myrinet uses the Up*/Down* algorithm to compute routes, but we do not specify the routing algorithm used; we just assume it is deadlock-free.

[2] "Host-detecting" messages contain the route from the tried port to the mapper, so the host does not need to access its routing table for replying.

[3] So, when a mapping process begins, the only port to explore is the mapper own port.

[4] For the sake of simplicity only a minimal number of switch ports ($P_0, P_1, ..$) and hosts ($A_1, B_1, ..$) have been included in the diagram.

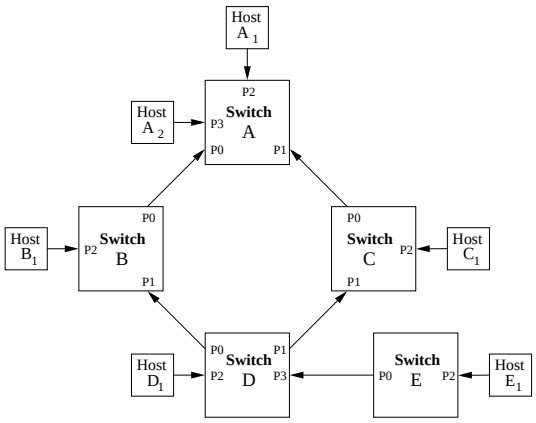


Figure 1. Network diagram for the example.

For this example, we assume that the well-known Up*/Down* routing algorithm is used, so every link in the diagram is an "arrow" of the network Up*/Down* graph. Taking into account the Up*/Down* rule[5] and the network graph, an user packet can not cross the switch D from port P_0 to port P_1 (and vice versa). But scouting messages can make this cross if it is needed during the mapping process.

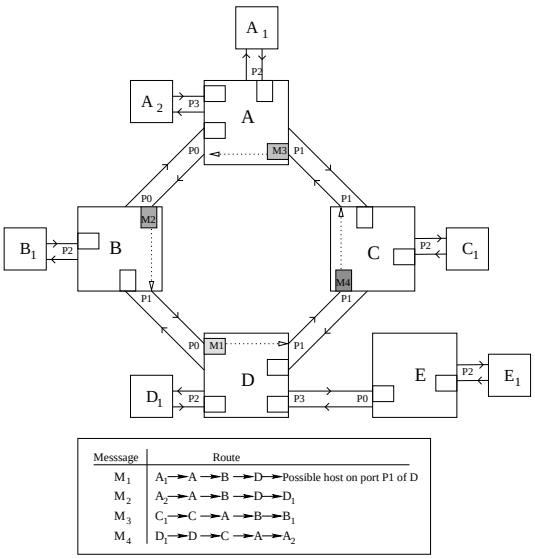
Figure 2 shows an example of a possible deadlock configuration[6] during the network mapping. In the diagram, every link has been split into its two opposing channels. At every switch port, input buffers are shown as small squares, the shaded ones containing messages ($M_1, M_2, ..$) that follow routes shown in the table below the diagram. We assume that every message fills its buffer more or less completely, in such a way that flow control prevents the reception of more flits from the corresponding input channel. Every dotted arrow begins at an input buffer that stores a message and points at the output channel requested by this message. We assume that host A_1 is the mapper, and that M_1 is the only scouting message in the network. It could be a "host-detecting" message exploring port P_1 of switch D. Other messages ($M_2, M_3, ..$) are user traffic and therefore, they follow valid Up*/Down* routes. However, M_1 follows an illegal route, and this fact causes deadlock.

We have used Up*/Down* in the example, but deadlock risk would exist for other deadlock-free routing algorithms. This is because, whatever routing restrictions the algorithm imposes, they will be ignored by scouting messages. Note that user traffic could never cause deadlock without at least one scouting message following an illegal route. However, not all the scouting messages follow illegal routes. So, the presence of these messages in the network makes possible, but does not necessarily imply, the existence of deadlock. Anyway, this possibility is a risk that should be considered[7].

[5] Although it is widely known, we remind the Up*/Down* rule: No packet would cross a link along the "up" direction (arrow direction) after having crossed one in the "down" direction (opposite direction).

[6] Although we assume that wormhole switching is used, this configuration would be also deadlocked if virtual cut-through switching is used.

[7] Assuming that the network topology contains at least one cycle.



Messsage	Route
M_1	$A_1 \rightarrow A \rightarrow B \rightarrow D \rightarrow$ Possible host on port P1 of D
M_2	$A_2 \rightarrow A \rightarrow B \rightarrow D \rightarrow D_1$
M_3	$C_1 \rightarrow C \rightarrow A \rightarrow B \rightarrow B_1$
M_4	$D_1 \rightarrow D \rightarrow C \rightarrow A \rightarrow A_2$

Figure 2. Deadlock caused by a network mapping process.

3.2 Deadlock Impact on Performance

Now it's time to analyze the impact of deadlocks caused by mapping on network performance. It's obvious that a permanent deadlock in the network will lead to a disastrous degradation of network performance. So, the question is: Will the packets involved in a deadlock be blocked forever?. We must remind that mapping is not the only cause of deadlock. For instance, even if deadlock-free routing is used, message communication is never free of transmission errors that may cause deadlock. Some networks implement mechanisms for solving this problem. For instance, in Myrinet, the time a packet can remain partially or completely stored in a buffer is limited by a fixed timeout[8]. When this timeout expires, timed-out packets are dropped[9]. So, the packets involved in a deadlock will be dropped sooner or later and will not be blocked forever. This mechanism is a deadlock recovery technique: Deadlocks may happen, but some mechanism detects and "breaks" them.

However, the implementation of such timeout mechanism does not turn harmless deadlocks that may occur during mapping. First, we must remind that the timeout mechanism "removes" deadlocks, but does not avoid them. This means that a deadlock will exist until it is detected and removed. So, it will last for the timeout period. During this period, a portion of the network will not be available[10]. This fact will affect message latency. Of course, for the sake of a faster detection of deadlocks, it is possible to reduce the timeout period. But, in this case, it will be difficult to distinguish real deadlocks from network congestion.

Another drawback of the timeout mechanism is the

[8] The Myrinet blocked-packet timeout is on the order of 25 ms.

[9] This mechanism also solves the problem, common under heavy traffic loads, of having packets stopped for a very long time due to flow control.

[10] This portion will be at least the one involved in the deadlock, but the whole network may become unavailable soon after deadlock happens.

fact that blocked messages must be dropped for removing the deadlock. In general, dropped user messages should be sent again from their source hosts, and this requires time-consuming processing and large buffers in the hosts, and produces an increase in network contention. However, it could be more dangerous the dropping of scouting messages[11]. A dropped scouting message will be a non-acknowledged scouting message, and the mapper will send it again, but it will only make this up to a limited number of retries. Deadlocks are possible during all the mapping process, so a scouting message may be dropped every time the mapper sends it to the network. In this case, although the port "scouted" by this message has not been really explored, the mapper will consider that the port is not connected to the device existing there. Therefore, the correct detection of the network topology can not be guaranteed.

To sum up, the possibility of deadlocks during mapping is a risk of degradation or malfunction of several aspects of the network performance, even if the network implements some mechanism for removing those deadlocks. In the next section, we study if this theoretical deadlock risk and impact could noticeably affect network performance.

4 Evaluation of Performance Degradation

In order to quantify the degradation of network performance due to deadlocks caused by mapping, we have evaluated the following metrics when mapping is performed:

- Probability of deadlock situations.
- Probability of failed mapping process.
- Average message latency from generation.

We have used a flit-level event-driven simulator that models the behavior of a Myrinet network. In the following sections, we will detail the network model and the evaluation criteria we have used. Later, we will present the results for each one of the three metrics specified above.

4.1 Network Model

The network is composed of a set of switches and hosts, all of them interconnected by links. Network topology is irregular and has been randomly generated taking into account some restrictions: There are exactly 4 hosts attached to each switch, all the switches have exactly 8 ports, and two neighboring switches are connected by a single link [12]. We have evaluated networks with 8, 16, and 32 switches.

To model the links, we have assumed short Myrinet LAN cables to interconnect switches and hosts. These cables offer a bandwidth of 160 MB/s, and have a delay of 4.92 ns/m. Flits and physical links are one byte wide. To model the switch, we have assumed that each switch has a simple routing control unit that removes the first flit of the packet header and uses it to select the output link. A crossbar inside the switch allows multiple packets to traverse it simultaneously. Each output port can process only one packet header at a time. When a packet gets the routing control unit, but it cannot be routed because the requested output link is busy, it must wait in the input buffer until its next turn. The blocked-packet timeout mechanism has been modeled on each switch buffer: A packet waiting in a buffer for more than 25 ms will be dropped.

We have used the routing algorithm really used in Myrinet: Up*/Down* based on BFS. Regarding traffic pattern, we have considered a uniform message destination distribution[13] . This distribution states that every host sends each message to any of the other hosts with equal probability. For each simulation run, we have considered that the packet generation rate is constant and the same for all the hosts. In order to evaluate deadlock probability as a function of traffic load, we have run simulations using different packet generation rates. For each network evaluated, the values of these rates have been obtained as equally-spaced fractions of the saturation rate during normal operation[14]. We have used a constant payload for user packets: 64 bytes[15]. "Host-detecting" messages have a fixed payload of 60 bytes and "switch-detecting" messages a fixed payload of 12 bytes. The size of each user or control packet is obtained by adding its variable header size to its payload.

We have modeled in the simulator every aspect of the Myrinet mapping mechanism described in Section 2. So, a host will act as network mapper, sending scouting messages and waiting (during 5 ms) for replies. A non-acknowledged message will be sent again up to three times.

4.2 Evaluation Criteria

In each simulation run, a mapping process begins after the network has reached a steady state. When the mapping process ends, the simulation finishes, regardless of whether the network topology has been correctly detected or not. If the mapping process causes deadlock, the simulation is considered "deadlocked"; otherwise it is considered "non-deadlocked". A "deadlocked" simulation will not stop the mapping process because blocked messages will be dropped after the corresponding timeout. If this dropping causes that a network component is not detected, the mapping process ends, and the simulation is considered a "failed-mapping" simulation. If the mapping process ends and the topology has been correctly detected, the simulation is considered a "correct-mapping" simulation[16].

[11] In fact, there will always be at least one scouting message involved in every deadlock that will be dropped.

[12] These assumptions are quite realistic and have already been considered in other similar studies [10, 11, 3, 12].

[13] Although we have used this distribution, we think the user traffic distribution has no influence in the results, and similar results would be obtained for other distributions like hot-spot, random, bit-reversal, etc..

[14] This traffic rate can be seen in Figure 5.

[15] This size was chosen because it is often used in tests and in order to avoid favoring deadlock with longer messages occupying several buffers.

[16] So, a "correct-mapping" simulation can be "deadlocked" or "non-deadlocked", but "failed-mapping" simulations are always "deadlocked".

For each network evaluated and packet generation rate used, we have run one hundred simulations, varying on each of them the random seed used to generate traffic. So, the probability of deadlock and failed mapping for each case can be estimated as the percentage of "deadlocked" simulations and "failed-mapping" simulations found once the corresponding simulations have finished.

Also, for each simulation, we have measured the average message latency in order to quantify the impact of deadlocks on such metric. This latency has been measured from message generation, so it is the time a message exists in the network, since it is generated at its source until it is completely received at its destination. In each simulation, the latency is considered only after the mapping process begins. For each network evaluated and packet generation rate used, we have estimated a single value of average message latency as the average of the measures obtained in the corresponding one hundred simulations.

4.3 Deadlock Probability Results

Figure 3 shows, for each network size used (8, 16, and 32 switches), the percentage of "deadlocked" simulations as a function of the traffic generation rate. As stated above, these results are used as a measure of deadlock probability.

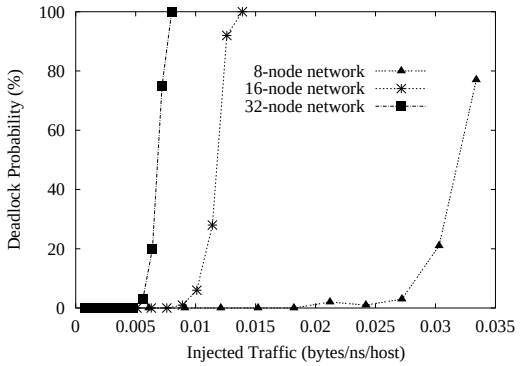


Figure 3. Percentage of "deadlocked" simulations for network sizes of 8, 16, and 32 nodes.

Regarding the deadlock probability, it grows when traffic load increases, regardless of network size. This is due to the fact that, at higher traffic generation rates, it is easier that cyclic arrangements of messages (so, deadlock) can occur, just because there are more messages simultaneously in the network. Figure 3 shows that deadlock probability also grows with network size. The results of "deadlocked" simulations for the 8-node network do not reach 100%, but for the 16- and 32-node networks, a 100% is reached. This is due to the fact that larger networks need more scouting messages to be mapped, and therefore it is easier that one of these messages makes a "down-up" cross and causes deadlock. Moreover, the topology of larger networks could have more cycles where cyclic arrangements of messages can occur. Although the results for 16- and 32-node networks are similar, the 100% of deadlocked simulations is reached for the 32-node network at a lower load.

4.4 Failed-Mapping Probability Results

Figure 4 shows, for the network sizes used, the percentage of "failed-mapping" simulations as a function of traffic generation rate. These results are considered a measure of the probability of incorrect topology detection. Note that the direct cause of failed mapping processes is the blocked-packet timeout mechanism used for removing deadlocks.

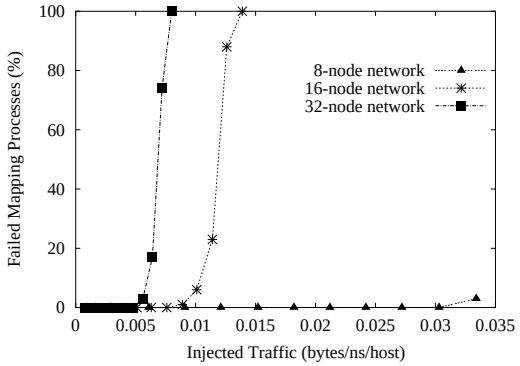


Figure 4. Percentage of "failed-mapping" simulations for network sizes of 8, 16, and 32 nodes.

As can be seen in Figure 4, given a network size and a traffic rate, the probability of failed mapping is always smaller than or equal to the deadlock probability shown in Figure 3. This is due to the fact that not every "deadlocked" simulation is a "failed-mapping" simulation. This is specially true in the case of the 8-node network. In this case, the failed-mapping probability reaches a very low maximum, whereas the deadlock probability maximum reaches 80%. For larger network sizes, the probability of a failed mapping reaches 100%. This is due to the fact that the number of scouting messages used in the mapping process is far greater than in networks of small size. In these cases, it is easier that a scouting message would be dropped three times and considered "lost" by the mapper.

4.5 Average Message Latency Results

The average message latency results shown in Figure 5 have been obtained for the same traffic loads and network sizes considered in previous sections. This figure compares the results obtained during mapping and the results obtained during the normal operation of the network[17].

These results show that average message latency during mapping begins to grow dramatically at traffic rates lower than the normal saturation point. This increase in latency during mapping is directly related to deadlock probability. Given a network size and traffic rate, if the deadlock probability is zero, the latency during mapping approaches

[17] The results for deadlock probability or failed mappings during normal operation of the network will always be zero.

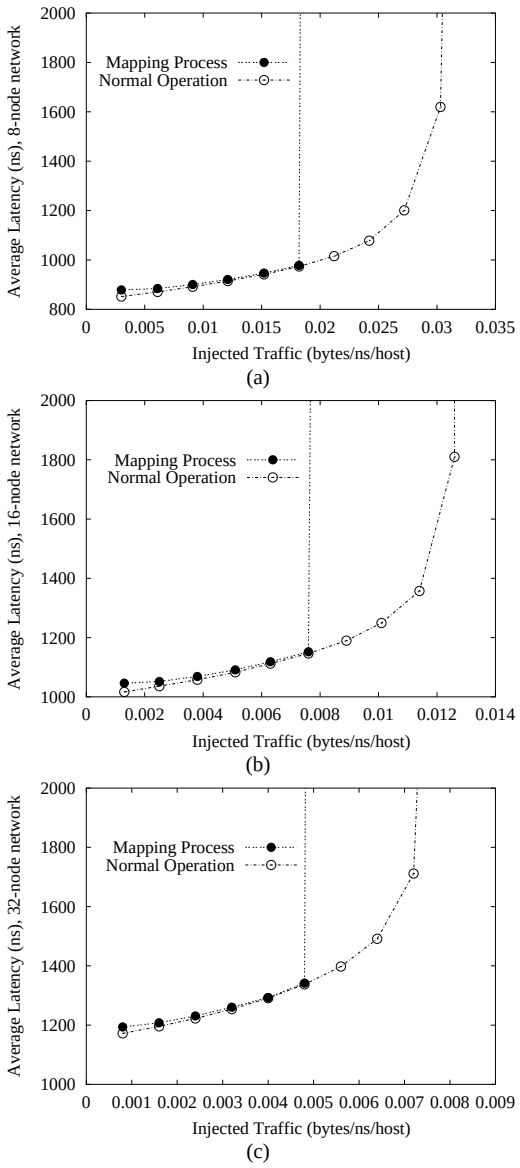


Figure 5. Average latency during mapping and during normal operation for networks of (a)8, (b)16, and (c)32 nodes.

the latency during normal operation[18]. For non-zero deadlock probabilities, the latency is enormous. This is due to the fact that deadlocks are detected and removed only after a timeout period. Again, the use of this deadlock recovery technique does not avoid the degradation of performance.

5 Conclusions

To sum up, we have detected and evaluated the risk and impact of deadlock during the centralized mapping of a network that uses source routing, "dumb" switches and deadlock recovery techniques. Deadlock may occur during mapping due to the use of scouting messages that follow routes not restricted by a deadlock-free routing algorithm.

[18]Note that this means that the additional traffic load due to scouting messages does not increase significantly the average message latency.

In this paper, we have shown the possibility of such deadlocks. We have also presented simulation-based results showing that deadlock probability is proportional to network size and traffic load. Although deadlock probability is smaller in networks of small size, deadlock-free guarantees can not be given during mapping in a network whose topology contains at least one cycle. We have also shown the degradation of network performance due to the blocked-packet timeout mechanism used for removing deadlocks. This mechanism causes an increase in average message latency and may cause mapping processes to detect the topology incorrectly. We have also shown results that measure this increase in latency during mapping and the probability of failed mapping processes. Both metrics are directly related to deadlock probability. According to our simulations results, 100% of mapping processes will cause deadlock and will not correctly detect the topology in networks of medium or large size for traffic loads approaching the network saturation point.

References

[1] J. Duato, S. Yalamanchili, and L. Ni. *Interconnection networks: An engineering approach* (Morgan Kaufmann Publishers, 2002).

[2] M. Schroeder, et al., Autonet: A high-speed, self-configuring local area network using point-to-point links. *IEEE J. on Selected Areas in Communications*, 9(8), 1991, 1318-1334.

[3] J. Sancho, A. Robles, and J. Duato. New methodology to compute deadlock-free routing tables for irregular networks. *Proc. CANPC*, Toulouse, France, 2000.

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

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

[6] N. Boden, D. Cohen, and R. Felderman. Myrinet- a gigabit per second local area network. *IEEE Micro*, 15(1), 1995, 29-36.

[7] T. Rodeheffer and M. Schroeder. Automatic reconfiguration in Autonet. Technical Report 77, Systems Research Center of Digital Equipment Corporation, Sept. 1991.

[8] J. Duato, R. Casado, F. Quiles, and J. Sánchez. Dynamic reconfiguration in high speed local area networks, in D.Avresky (Ed.) *Dependable Network Computing* (Kluwer Academic Publishers, 1999) 207-232.

[9] http://www.myri.com. Myricom home page. Technical report, Myricom Inc., 2001.

[10] P.J. García, M. Mora, F.J. Alfaro, J.L. Sánchez, and J. Flich. Evaluation of alternative arbitration policies for Myrinet switches. *Proc. CAC*, Fort Lauderdale, USA, 2002.

[11] J. Flich, M. Malumbres, P. López, and J. Duato. Improving routing performance in Myrinet networks. *Proc. IPDPS*, Cancun, Mexico, 2000.

[12] F. Silla and J. Duato. Improving the efficiency of adaptive routing in networks with irregular topology. *Proc. HiPC*, Bangalore, India, 1997.