

Dynamic Network Reconfiguration Algorithm

Khalid Kh. Sharaf^{1,*}, AbuBaker ElHussein²

¹Computer Science Department, Alaqsa University, Gaza, Palestine

²Computer Engineering, Africa university, Khartoum, Sudan

Abstract Quality of service(QoS) is a term widely used in the last recent years in the area of networks. The QoS generally describes the assurance of sufficiently low delay and packet loss for certain types of applications or traffic, many distributed applications executed on networks of workstation require the interconnection network to provide some QoS support. These network must be able to support topology changes due to any failures occur(component failure, hot replacement, hot expansion) without stopping traffic, the dynamic network reconfiguration(DNR) satisfy the QoS requirement. The dynamic reconfiguration of the interconnection network is advanced feature of some multicomputer to reduce the communication overhead. The DNR is defined as the process of changing from one routing function to another while in real time. In this paper we expound in depth the reconfiguration algorithm and many terms related to it.

Keywords Computer Networking, Quality of Service, Dynamic Network Reconfiguration, Protocols

1. Introduction

The quality of service (QoS), availability, reliability, and predictability are becoming increasingly important as system size and demand increase. This is especially true for high-end servers (web, database, data centers, video-stream, etc.), which are currently based on clusters of PCs and/or highly parallel computers. In these systems, the network interconnecting plays a very important role to achieving high system availability. have been developed over recent years that enable increased network dependability. These techniques are in common use today in system area networks (SANs) and local area networks (LANs) and even in recent Ethernet specifications[16]. The local area network (LAN) may change their topology due to switches being turned on or off, link remapping, over load, component failures and removal or addition of components. These situation normally requires the network routing algorithm to be reconfigured(to update routing table) in order to (re) establish full network connectivity among the attached nodes. In transitioning between the old and new routing functions during network reconfiguration, additional dependencies among network resources may be introduced as cyclic dependencies (deadlock)[1][2][3]. Many techniques processing this situation through static reconfiguration meaning that user traffic is stopped and the average packet latency increases dramatically during the reconfiguration. It should be noted that traffic is stopped in the entire network. So it will not be possible to guarantee

the required QoS. Other techniques. processing this situation is dynamic reconfiguration, The application of a dynamic reconfiguration mechanism will reduce the negative effect of the reconfiguration process. The dynamic reconfiguration provide the system availability and especially suitable to support many application(web, database, data centers, multimedia, etc),which require a guaranteed QoS. But the dynamic reconfiguration by it itself not provide QoS guarantee. However, the opposite is true, without implementing the dynamic reconfiguration it will be impossible to guarantee QoS, because the user traffic will be stopped for many time. With dynamic reconfiguration, the idea is to allow user traffic to continue uninterruptedly during the time that the network is reconfigured, thus reducing the number of packets that miss their real-time/QoS deadline[9]. It is important to take into account several concepts regarding the dynamic reconfiguration as local and global reconfiguration, equivalent topology, Reconfiguration triggering, strategies of change and Reconfiguration control. First, local reconfiguration only affects the requesting node and any other node but the global reconfiguration is a modification in the links of multiple nodes at the same time, global reconfiguration could produce deadlock problems. Second, equivalent topology. Reconfiguration can be made in such a way that the new network has the same topology (only some nodes have exchanged their places) or allowing any arbitrary topology, also, it is not necessary to modify the routing algorithm. Alternatively, any arbitrary topology may be allowed even irregular topologies. In the last case, we must take into account deadlock problems[18]. Third, reconfiguration triggering. When the change in the topology occurs, the reconfiguration algorithm updating the routing table in host and switches. Forth, There are two strategies to

* Corresponding author:

ekhsharaf@hotmail.com (Khalid Kh. Sharaf)

Published online at <http://journal.sapub.org/computer>

Copyright © 2013 Scientific & Academic Publishing. All Rights Reserved

choose possible changes, first, Small alteration in the network. A node can only exchange its position with one of its neighbor nodes, second, Large alteration in the network. A node can change to any other place in the network. To keep the same topology easily, it is convenient for the changed node to take the position left by the node which requested the change, last, Reconfiguration control. There are two ways for controlling the dynamic reconfiguration of the network: a centralized control (a node -the system controller- is responsible to doing network reconfiguration) or a distributed control (each node controls its own reconfiguration). In a centralized control, the system controller is connected to all the nodes of the system through a control bus. This type of control could be a bottleneck for the system, since all the nodes must reconfiguration requests to the system controller.. An alternative is to use a distributed control. In this case, several network reconfigurations can be carried out simultaneously, without bottleneck problems. Furthermore, distributed control is well suited for fault tolerant operation as it does not require any central facility and can easily bypass faulty nodes or links. However, the deadlock problem could appear. Another problem is that every node must communicate with each other to make a change, increasing the message traffic in the network. In both schemes there exists a considerable overhead to carry out a reconfiguration[3]. The rest of the paper is organized as follow: section 2 discusses the motivations for the need reconfiguration algorithm, section 3 discuss the routing and dynamic network reconfiguration, section 4 discuss the important terms in the dynamic reconfiguration, section 5 present related work, section 6 present the suggested algorithm, section 7 discuss the verification of the algorithm, section 8 discuss the evaluation and expected result and section 9 is the conclusion.

2. Motivation

Like what we mentioned earlier in the introduction, there are many motivation for the dynamic network reconfiguration, although there are different ways to deal with component failures, once a failure is occur the reconfiguration process discovers the new topology and computes the new routing tables. During the reconfiguration process the traffic is not stopped thus reducing the impact of the topology change on the application performance. From our point of view, the mechanisms proposed up to now do not achieve all the goals because of the way they guarantee deadlock freedom during the reconfiguration process. All of them are based on deadlock-avoidance techniques to guarantee the network does not block. In particular, a deadlock may occur when cyclic channel dependencies are formed by packets (Fig 1 illustrate the cycle in a dynamic network reconfiguration). Thus, the reconfiguration mechanisms avoid these cyclic dependencies from being formed.

The network will be deadlocked if a cyclic channel

dependency occurs and the resources used (buffers) along the cycle become full. However, if buffers are not full, then packets can still advance and thus deadlock will not form.

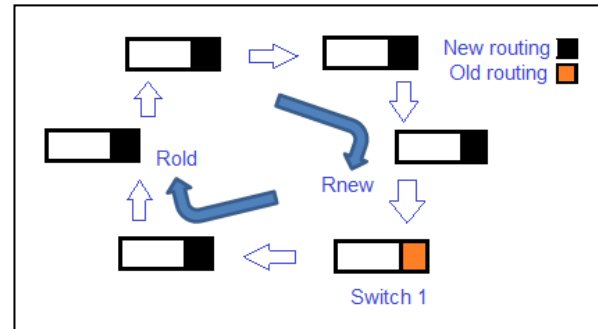


Figure 1. Cycle in a dynamic network reconfiguration

Sum objective must account when designing a dynamic reconfiguration mechanism, The first, is to guarantee that the network will not block during the reconfiguration phase.

Although both routing algorithms (the new and the old) will be deadlock-free, interactions between both routing algorithms could lead to deadlocks. Second, is to reduce the number of dropped packets during the reconfiguration process. Packets may be dropped when a reconfiguration mechanism is triggered because a component has failed. And when the reconfiguration mechanism drops packets in order to guarantee deadlock freedom. Third, is to become transparent to the running applications. That is, the performance of the network (throughput, latency and QoS levels) should not be altered due to a reconfiguration process. Forth, should be its simplicity, the mechanism should work for any routing algorithm and any topology.

3. A routing and Dynamic Network Reconfiguration

Firstly, Routing is the process of moving a packet of data from one network to another network based on the destination IP address. The Internet uses routing to move data from your computer, across several networks, to reach a final destination, like a website. Specialized computer devices that perform this routing function are referred to as routers. Routers use the information contained in a route to make decisions about which network interface to forward a packet through in order to reach the destination address in the packet. Routers maintain a list of routes which is often referred to as a routing table.

Routers look up routes in the routing table to figure out how to move data from one network to another network. Routes are simply the signposts that tell a router which network interface to forward a packet through in order to reach the packet's intended destination.

There are two basic kinds of routes:

1. Static routing.
2. Dynamic routing.

First one, Static routing is not really a routing protocol. Static routing is simply the process of manually entering

routes into a device's routing table via a configuration file that is loaded when the routing device starts up. As an alternative, these routes can be entered by a network administrator who configures the routes manually. Since these manually configured routes don't change after they are configured (unless a human changes them) they are called 'static' routes. Static routing is the simplest form of routing, but it is a manual process. Use static routing when you have very few devices to configure and when you know the routes will probably never change.

Static routing also does not handle failures in external networks well because any route that is configured manually must be updated or reconfigured manually to fix or repair any lost connectivity. Second one, Dynamic routing protocols are supported by software applications running on the routing device (the router) which dynamically learn network destinations and how to get to them and also advertise those destinations to other routers. This advertisement function allows all the routers to learn about all the destination networks that exist and how to get to those networks. A router using dynamic routing will 'learn' the routes to all networks that are directly connected to the device. Next, the router will learn routes from other routers that run the same routing protocol (RIP, RIP2, EIGRP, OSPF, IS-IS, BGP etc). Each router will then sort through its list of routes and select one or more 'best' routes for each network destination the router knows or has learned. Dynamic routing protocols will then distribute this 'best route' information to other routers running the same routing protocol, thereby extending the information on what networks exist and can be reached. This gives dynamic routing protocols the ability to adapt to logical network topology changes, equipment failures or network outages 'on the fly'.

Secondly, Dynamic network reconfiguration is described as the process of replacing one routing function with another while the network keeps running. The main challenge is avoiding deadlock anomalies while keeping limitations on packet injection and forwarding minimal. a deadlock may occur when cyclic channel dependencies are formed by packets, when the topology of the network changes, either involuntarily due to failing/faulty components or voluntarily due to hot removal or addition of components. This normally requires the network routing algorithm to be reconfigured (routing function) in order to (re)establish full network connectivity among the attached nodes. so we will presenting a simple, fast and efficient mechanism for dynamic network reconfiguration.

4. Some Concepts

Duo to network configuration, there are several way to implement dynamic reconfiguration and avoiding deadlock, Determine the group of choice for reconfiguration are the existence of extra resources for routing depending on the network characteristics , the way the routing algorithm implemented ,advanced routers(intelligence) and network

interface card[10]. When the extra resources are unavailable, it is necessary to guarantee deadlock freedom at all time among entire network, by avoiding cyclic dependencies between network resources we can guarantee deadlock freedom[15], including the interaction between the old routing function and the new routing function, fortunately, there are more than one type of deadlock, some deadlocks are due to the current interaction between the old and new routing algorithm and will disappear when the routing tables completely updated, these deadlock named transient deadlock and it is not dangerous, on the other hand, some other deadlocks will remain after updating the all routing table, these named permanent deadlock and it is more dangerous, if cyclic dependencies occurred during reconfiguration, it is important to guarantee that no permanent deadlocks will form. Thus, there are three way to implement dynamic reconfiguration, first, avoiding cyclic dependencies between the existence network resources at all-time[16], it used in all the networks, including the networks with source routing and/or networks without virtual channels. Second, avoiding permanent deadlock when allowing cyclic dependencies between network resources, this approach requires distributed routing. Third , Divide the network resources into two sets and statically reconfiguration in sequence, this approach requires distributed routing and extra resources(virtual channel)[18]. A common problem that arises in all approaches is unroutable messages, and to solve this problem, there are three way, first, discard unroutable messages , the cost of having retransmit message is problem , so some hardware support also required, second, buffer unroutable messages until they become routable, it is possible to work if enough buffer space is available, third, allow especial routing option for unroutable messages, in this solution must guarantee avoiding permanent deadlock.

5. Related Work

In some situations, however, the premises on which the routing algorithm and/or network topology are defined may break, which affects the network's dependability. This can happen, for example, when the topology of the network changes, either involuntarily due to failing/faulty components or voluntarily due to hot removal or addition of components. This normally requires the network routing algorithm to be reconfigured (routing function) in order to (re) establish full network connectivity among the attached nodes. In transitioning between the old and new routing functions during network reconfiguration, additional dependencies among network resources may be introduced, causing what is referred to as reconfiguration-induced deadlock. Current techniques typically handle this situation through static reconfiguration, meaning that application traffic is stopped and, usually, dropped from the network during the reconfiguration process. While this approach guarantees the prevention of reconfiguration induced

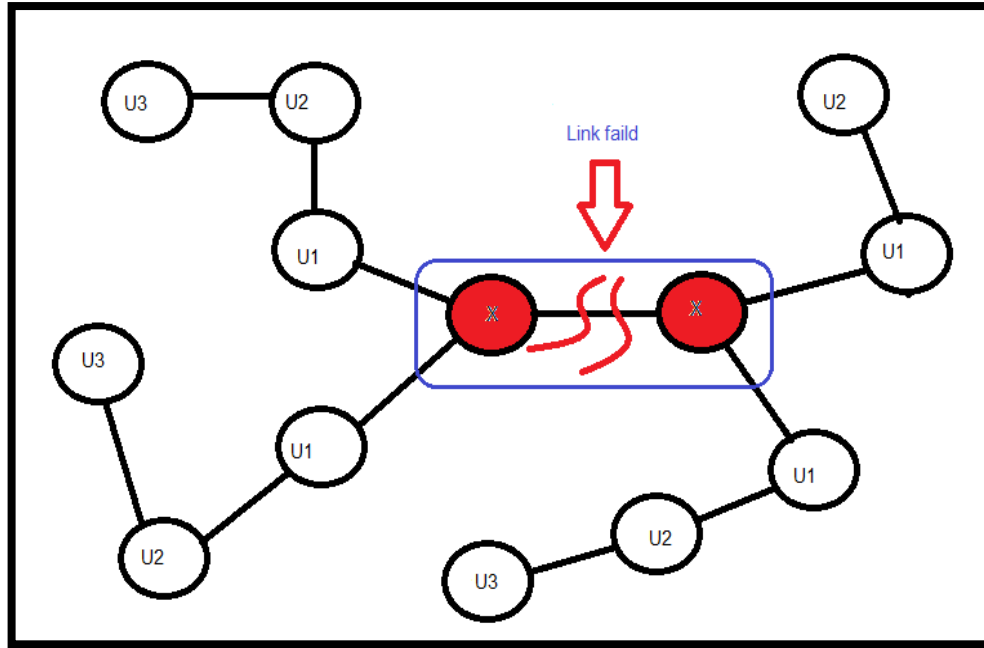


Figure 3. Case link failure

```

/* The all cases may occur when scheduling packet*/
If (Psched==0)
{
    If (Pcalc==Prout==0)
        The old packets will routed by the old
        routing function
    Else if (Pcalc==1 && Prout==0)
        The packets in transient situation
    Else if (Pcalc==Prout==1)
        Not occur
    Else if (Pcalc==0 && Prout==1)
        Not occur
}
Else if (Psched==1)
{
    If (Pcalc==Prout==1)
        The new packets will routed by the new
        routing function
    Else if (Pcalc==Prout==0)
        The packets are dropped
    Else if (Pcalc==0 && Prout==1)
        The packets are dropped
    Else if (Pcalc==1 && Prout==0)
        The packets are dropped
}

```

Figure 4. Cases may occur when scheduling packet

The switches receive the compute paths "new paths" it updates its routing table, and removing the old routing table. Thus, the packets will be routing with the new routing table. Notice that packets already routed and waiting in the queue of the switch will not be routed again. Fig 2 illustrates the order of the routing table updating in case of component

failure and Fig 3 illustrates the order of routing table updating in case link failure.

As shown in the Fig. 2, the "X" indicate to failed component, and "U1" indicate to first routing table will be updating, and so on.

As shown in the Fig. 3, illustrate the order of the routing table updating, in case link failed. the "U1" indicate to first routing table will be updating, and so on.

6. After all routing table updated to the new routing table, may be deadlock occur because the old packet Scheduled at the head of the switch queue. in this case, the NM require the deadlock recovery process. To break a deadlock some packets(scheduled at the head of the switch queue "old packet ") will be discarding . To do this, using a sign bit that specify the routing algorithm used to rout packet, "zero" value indicates the old routing function and "one" indicates the new routing function which means that each switch signed with the sign bit corresponding to the routing algorithm applied by its routing tables. Packets are also signed with the sign bit(sign bit injecting with an additional bit), when the packet arrival to the node (switch or router) and forwarded to the next node the sign bit of the packet is set to the sign of the reconfiguration algorithm used by the forwarded packet, and when receiving the new routing table the sign bit of the switch is changed accordingly. Because the routing and the packet forwarding not occurred at the same time, Some packets have not credit at the next switch, in this case, the routing algorithm routes packets(P_{calc}) to calculate the credit of the output port, and after the packet scheduled , the packet only no credits available and the packet could lead to deadlock will be dropping. Notes that whenever a new packets arrives to the switch(out o/p port is computed) and sign bit is union with the packet, the sign bit of the packet (P_{rout}) is equal the sign bit of the switch at the same time, thus deadlock recovery come to action when the packet reached

to the head of queue and being scheduled (P_{schd}).

The following analysis (Fig. 4) illustrate the all cases may occur when scheduling packet.

Note1: when the new routing table available to the switch they are used for the incoming packet, and just the packet is routed the output port and the sign bit of the switch are injecting to the packet until the packet arrived to the head of the queue.

Note2: when the credit of the packet are available, then the packet is forwarded. But if the packet haven't a credit at the request of the output port, then the switch may be drop the packet, To achieve this, the sign bit of the packet is compared with the current sign bit of the switch when the packet was routed, thus, some packet discarded as depicted in the above analysis.

7. Verification of the Algorithm

In this section we will discuss verification of the algorithm according to table 1.

Table 1. Illustrate the cases of the packet

Case	Values	Forward	Discard	Not occur
1	000	Yes		
2	001		Yes	
3	010			Yes
4	011		Yes	
5	100	Yes		
6	101		Yes	
7	110			Yes
8	111	Yes		

As shown in table 1 The values consist of three digit, first indicate the P_{calc} , second indicate P_{rout} and third indicate P_{schd} . As shown in the above table, Case 1 is a normal case ($P_{calc} = P_{rout} = P_{schd} = 1$), because the packet was routed with the old routing table and being scheduled (the packet is not discarded). Case 2 the packet will discard, because the old packet is routed with the old routing table but when being scheduled the routing table has been updated. Case 3 have no meaning since they represent the case when the switch downgrades from a new routing to the old routing. In case 4 the packet will discard, because it may happen that an old packet using a forbidden transition in the new routing algorithm (it could introduce a deadlock). Case 5 is a transient state, thus no need to discard any packet. Case 6 is like case 4, the packet will discard, because in this case the new packet is routed with the old routing table but when being scheduled the routing table has been updated. Case 7 like case 3 have no meaning since they represent the case when the switch downgrades from a new routing to the old routing. Case 8 is a normal case ($P_{calc} = P_{rout} = P_{schd} = 1$), because the packet was routed with the new routing table and being scheduled.

8. Evaluation and Expected Result

There are two way to evaluate the suggested mechanism, first, when sending new routing tablet to all switches in the network, second, when sending the difference between the old routing table and new routing table.

$$R_{diff} = R_{new} - R_{old}$$

In all the cases the NM monitors the network for changes, this is done by sending control packet to the all switches, upon reception of the control packet, the NM detecting any change through their links. In all reconfiguration mechanism, once a network topology occurred, the NM sends the new routing table Accompanied a control packet through the control virtual channel to all the switches. Now we compare the suggested mechanism with other mechanisms as Double Scheme mechanisms and Simple Reconfiguration. In case Simple Reconfiguration (SR), there are two versions evaluated. first one, named SR-PDA, this mechanism reduce the number of dropped packets (packet drop aware), due to issues the tokens that separate the old and new traffic at the same time tables are sent to switches[19]. Second one, named SR-LA, this mechanism tries to reduce the latency of packets (latency aware), due to first distributes the new routing tables and then sends the tokens to separate the traffic.. More details of DS, SR-PDA and SR-LA can be found in [19]. In case of Double Scheme mechanisms (DS) during the distribution of paths one virtual channel is drained. Once drained control packets are sent by the NM to restore normal operation (the reconfiguration has finished), in case suggested algorithm the all switches must be updated before using the new routing paths. This mechanism affect the number of dropped packets have a positive effect, means that the number of dropped packets less than dropped packets in other algorithms, but we note that, the suggested algorithm needs more time to reconfigure the network than DS and SR-PDA. This is due to the sequence used to reconfigure the switches. See figure 5, the time required for distributing four routing tables for two different cases.

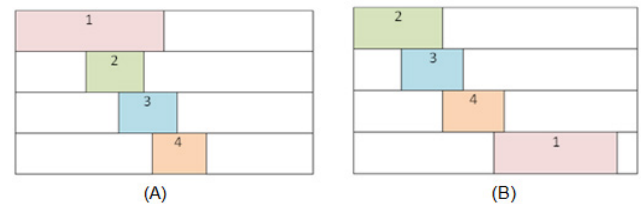


Figure 5. Example for required time for new routing

NM is sending sequentially the routing tables, each table requiring the same injection time, under assumption that all the tables have the same size, but each table requires different amount of time to arrive to their destination. As shown In the Figure, the total required time in case (A) is lower than in case (B), Because in case (A) the distribution of tables is highly overlapped. Therefore, total required time can be reduced, as shown in case (A) sending first tables that take more time to reach their destination. Thus, the order in

which tables are distributed affects the total reconfiguration time. Because the suggested algorithm sending the routing tables (All routing table) firstly to the switches surrounding the failure, and then to the remaining switches by increasing the distance to the failed component as shown in figure 2 and 3, the total required time for the reconfiguration may increase compared with the other reconfiguration mechanisms. But in case sending the difference between the new routing table and the old routing table, may greatly reduce the reconfiguration time. Finally fig.6 illustrate the simulation result,

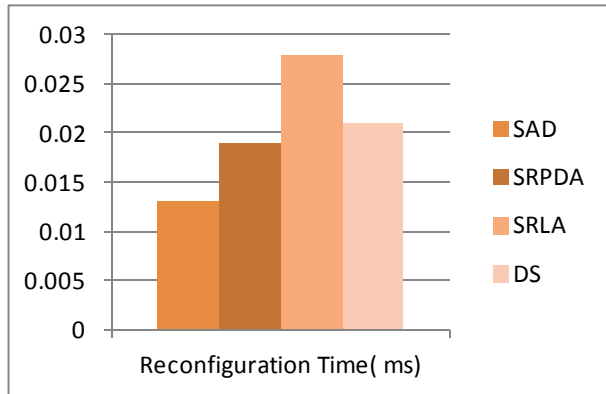


Figure 6. Reconfiguration time (ms)

As shown in figure 6, the suggested algorithm needs less time than other algorithms.

9. Conclusions

In this paper we have described a simple and efficient procedure for dynamic network reconfiguration called a suggested algorithm. Our method works for variety of topologies and operates with any pair of routing functions. a suggested algorithm working in two way, first one, when sending new routing tablet to the all switches in the network, and the second one, when sending the difference between the old routing table and the new routing table, and using sign bit for switch and packet. , the suggested algorithm gets more time to reconfigure the network than other algorithm, but guarantee deadlock free and low loss packet. To the best of our knowledge, this is the first practical way to deal with the deadlock when it happens in the dynamic network reconfiguration without the need for other resources.

As for future work, we are planning to present the network architecture that provide auto reconfiguration in dynamic network.

In the future we plan to present dynamic network architecture supports automatic reconfiguration, based on the directory enabled network management, which involves maintaining configuration information in a special repository known as directory.

ACKNOWLEDGEMENTS

We would like to thank Dr. AbuBaker ElHussein for their useful commands and help.

REFERENCES

- [1] R. Casado, A. Bermúdez, J. Duato, F. J. Quiles, and J. L. Sanchez, "A protocol for deadlock-free dynamic reconfiguration in high-speed local area networks", *IEEE Transactions on Parallel and Distributed Systems*, 12(2):115-132, 2001.
- [2] O. Lysne, T. M. Pinkston, and J. Duato, "A methodology for developing dynamic network reconfiguration processes", In 2003 International Conference on Parallel Processing (ICPP'03), IEEE, 2003, pp 77-86.
- [3] T. Pinkston, R. Pang, and J. Duato, "Deadlock-free dynamic reconfiguration schemes for increased network dependability", *IEEE Transactions on Parallel and Distributed Systems*, 14(8):780-794, 2003.
- [4] Olav Lysne, Timothy Mark Pinkston, and Jose Duato, "A Methodology for Developing deadlock-Free Dynamic Network Reconfiguration Processes", *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS*, 2005, 16(5), pp,428-443.
- [5] R. Casado, A. Bermúdez, J. Duato, F. J. Quiles, and J. L. S'anchez, "A protocol for deadlock-free dynamic reconfiguration in high-speed local area networks", *IEEE Transactions on Parallel and Distributed Systems*, 12:115-132, Feb 2001.
- [6] O. Lysne, J. M. Montañana, T. Pinkston, J. Duato, T. Skeie, and J. Flich, "Simple Deadlock-Free Dynamic Network Reconfiguration", In 11th International Conf. on High Performance Computing (HiPC), Bangalore (India), Dec 19-22, 2004.
- [7] Lysne, O., Pinkston, T.M., Duato, J., "Part II: A methodology for developing deadlock-free dynamic network reconfiguration on processes", *IEEE Trans. Parallel and Distributed Systems* 16(5), 428-443, 2005
- [8] Robles-Gómez, A., Bermúdez, A., Casado, R., Solheim, A.G., "Deadlock-free dynamic network reconfiguration based on close Up*/Down* graphs" In: Luque, E., Margalef, T., Benítez, D. (eds.) *Euro-Par 2008*. LNCS, vol. 5168, 2008, pp. 940-949. Springer, Heidelberg
- [9] Solheim, A.G., et al., "Efficient and deadlock-free reconfiguration for source routed networks", In: 9th Worksh. Communication Architecture for Clusters, 2009
- [10] Zafar, B., Pinkston, T.M., Bermúdez, A., Duato, J., "Deadlock-free dynamic reconfiguration over InfiniBand networks", *Int'l. Jnl. Parallel, Emergent and Distributed Systems* 19(2), 2004, pp.127-143.
- [11] Jose Duato, Olav Lysne, Ruoming Pang and Timothy M. Pinkston, "Part I: A Theory for Deadlock-Free Dynamic Network Reconfiguration", *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS*, 2005, 16(5), pp:412-427.
- [12] O. Lysne, J. Montanana, J. Flich, J. Duato, T. Pinkston, T. Skeie, "An efficient and deadlock-free network reconfiguration protocol, Computers", *IEEE Transactions on* 57, 2008, pp. 762-779.

- [13] R. Pang, T. M. Pinkston and J. Duato, "The Double Scheme: Deadlock-free Dynamic Reconfiguration of Cut-Through Networks", In The 2000 Intl' Conference on Parallel Processing, IEEE Computer Society, 2000, pp. 439-448..
- [14] R. Casado, A. Bermúdez, F. J. Quiles, J. L. Sánchez, and J. Duato, "Performance evaluation of dynamic reconfiguration in high-speed local area networks", In Proceedings of the Sixth International Symposium on High-Performance Computer Architecture, 2000.
- [15] R. Casado, A. Bermúdez, J. Duato, F. J. Quiles, and J. L. Sánchez, "A protocol for deadlock-free dynamic reconfiguration in high-speed local area networks". IEEE Transactions on Parallel and Distributed Systems, 12:115-132, Feb 2001
- [16] N. Natchev, D. Avresky, and V. Shurbanov, "Dynamic reconfiguration in high-speed computer clusters", In I. C. Society, editor, Proceedings of the International Conf. on Cluster Computing, Los Alamitos (USA), 2001, pp. 380-387.
- [17] R. Casado, A. Bermúdez, F. J. Quiles, J. L. Sánchez, and J. Duato, "Performance evaluation of dynamic reconfiguration in high-speed local area networks". In Proceedings of the Sixth International Symposium on High-Performance Computer Architecture, 2000.
- [18] O. Lysne, J. M. Montañana, T. Pinkston, J. Duato, T. Skeie, and J. Flich, "Simple Deadlock-Free Dynamic Network Reconfiguration", In 11th International Conf. on High Performance Computing (HiPC), Bangalore (India), 19-22 Dec 2000
- [19] J. Fernández, J. García, and J. Duato. A new approach to provide real-time services on high-speed local area networks. In Proceedings of the 15th International Parallel and Distributed Processing Symposium (IPDPS-01), pages 124-124, Los Alamitos, CA, April 23-27 2001. IEEE Computer Society.