International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 02 Issue: 05 | August-2015 www.irjet.net p-ISSN: 2395-0072

# Dynamic Network Reconfiguration for Switch-based Networks

**Ms. Deepti Metri[1], Prof. A. V. Mophare[2]**

[1]*Student, Computer Science and Engineering, N. B. N. Sinhgad College of Engineering, Maharashtra, India*
[2]*Professor, Computer Science and Engineering, N. B. N. Sinhgad College of Engineering, Maharashtra, India*

---

**Abstract** - ***In today's world of networking, communication subsystems play a key role and have become bottleneck in systems due to significantly increase in computer performance. To counter this problem, current high-performance distributed systems employ switch-based interconnection networks. These subsystems should not only ensure high levels of performance, but must also be highly robust and easily available. In this scenario, after the occurrence of a topological change, a management mechanism must reestablish connectivity between network devices. This requires performing a network reconfiguration, which consists in updating the routing function [8] [13]. The main challenge involved in network reconfiguration is the reduction of performance degradation during the change assimilation process. Unfortunately, in most cases, user traffic is stopped during the reconfiguration process to avoid deadlock. These strategies are called static reconfiguration techniques. Static reconfiguration techniques significantly reduce network service since the application traffic is temporally stopped in order to avoid deadlocks but has negative impact on network service availability [4]. So we prefer the dynamic reconfiguration technique in which injection of data traffic is allowed while the routing function is being updated. [1].***

***Key Words:****dynamic reconfiguration, network management, deadlock avoidance, and performance*

## 1. INTRODUCTION

Computer performance has significantly increased in recent years and, consequently, communication subsystems have become bottlenecks within systems.To counter this problem, current high-performance distributed systems employ switch-based interconnection networks. This subsystem is usually based on a point-to-point switch-based technology, and should not only ensure high levels of performance, but must also be highly robust and easily available.

In recent years, some different technologies have emerged, based on distributed routing (as InfiniBand [8]) or source routing (as Myrinet 2000 [13]). In such subsystems, after the occurrence of a topological change such as the failure or aggregation of a network component, a new routing function, which is appropriate to the resulting topology, must be obtained and uploaded in the routing elements. The change assimilation process is typically performed by a centralized management entity, which is called mapper in Myrinet, subnet manager in InfiniBand, and fabric manager in ASI. More specifically, once the manager detects the change, it first has to collect information about all active devices—endpoints and switches—physically connected to the fabric. After that, a new set of paths for packet delivery among endpoints must be established. The last phase involved in the change management process consists in replacing the old routing function by the new one[1].

The process of replacing a routing function with another one is traditionally referred to as network reconfiguration. It is well known that, even when both routing functions are by themselves deadlock-free, updating the routing paths in an uncontrolled way may leads to deadlock situations due to the introduction of temporary dependencies among network resources [7], [11]. This means packets that belong to one of the routing functions may take turns that are not allowed in the routing function.

Traditional reconfiguration mechanisms (designed for Autonet and Myrinet [4] networks) solved this problem by emptying the network of packets before replacing the routing function. Such a simple approach is referred to as static reconfiguration, and has a negative impact on the network service availability, since for a period of time the network cannot accept packets [1].

Several distributed real-time applications have strict communications requirements. Distributed multimedia applications have similar, although less strict, quality if service (QoS) requirements with rigorous limitations on CDV (Cell delay variation), CTD (maximum cell transfer delay), and CLR (cell loss ratio) parameters. Now-a-days, many distributed multimedia applications, such as real time video compression and decompression, video-on-demand servers, distributed databases, etc., require computing power beyond that available in current uniprocessors. These application require very high network bandwidth, which can be provided by high speed LAN [5].

When multimedia applications are executed on a local area switch-based network, topology changes (for example, due to components failure) may affect their behaviour. If static reconfiguration is used, user traffic is stopped and the average packet latency increases

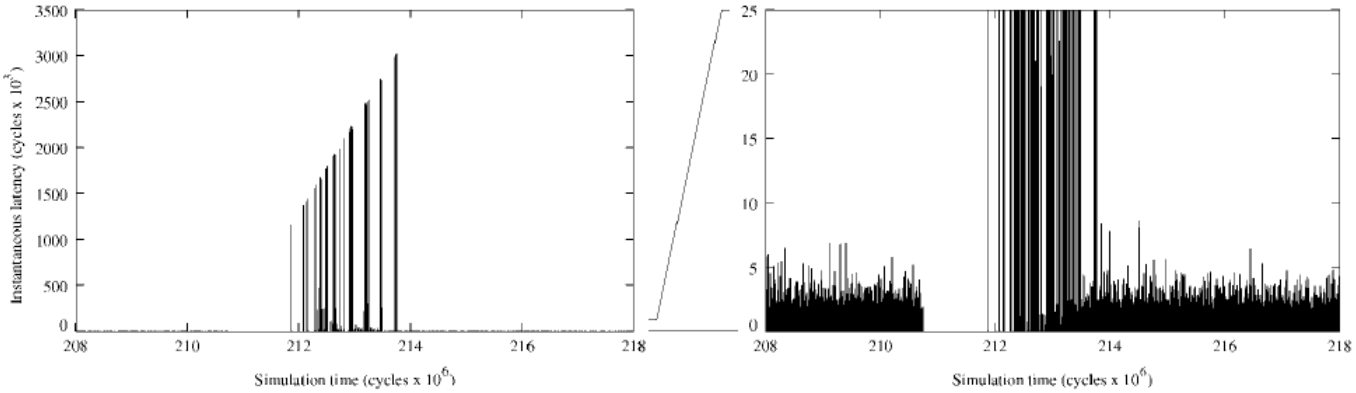


**Fig.-1:** Consequences of static reconfiguration in Auto net.

dramatically during the reconfiguration [5].This has negative impact on network service availability, since for a period of time the network cannot accept packets.

Therefore we tackle reconfiguration of interconnection network from dynamic point of view- by performing network reconfiguration without stopping the transmission of user packets. In order to guarantee deadlock freedom, dynamic reconfiguration schemes are more elaborate than static reconfiguration schemes [1].

## 2. RELATED WORK

Depending on network characteristics, there are several approaches to implement deadlock-free dynamic reconfiguration. The network characteristics that mainly determine the set of choices for reconfiguration are the existence of extra resources for routing the way the routing algorithm is implemented, and the intelligence ”available” in the routers and network interface cards.

The existence of extra resources significantly simplifies deadlock avoidance during dynamic reconfiguration. On the other hand, when no extra resources are available, it is necessary to guarantee deadlock freedom at all times. This can be done by avoiding cyclic dependencies between network resources, including the interaction between the old and old routing functions. Fortunately, not all deadlocks are equally dangerous. Some deadlocks are due to the current interaction between the old and new routing algorithms and will disappear when all the routing tables have been completely updated. These deadlocks are referred to as *transient deadlocks.* Some other deadlocks will remain after finishing reconfiguration. These deadlocks are much more dangerous and will be referred to as *permanent deadlocks.* Therefore, if cyclic dependencies are allowed during dynamic reconfiguration, it is necessary to guarantee that no permanent deadlock will form.

In summary, there exist three basic approaches to implement dynamic reconfiguration:

1. Avoiding cyclic dependencies between network
2. Allowing cyclic dependencies between network resources while avoiding permanent deadlocks.
3. Splitting network resources into two sets and statically reconfiguring those sets in sequence.

A common problem that arises in all the approaches is that some messages may become *unroutable* due to component failures and / or changes in routing tables. Unroutable messages can be easily identified: The routing algorithm does not offer any path for them to reach their destination. Basically, there exist three solutions for the problem:

1. Discard unroutable messages
2. Buffer unroutable messages until they become routable.
3. Allow especial routing options for unroutable messages.

Solution 1 obviously solves the problem at the cost of having to retransmit messages. However, this solution works in all kinds of networks. Solution 2 is only guaranteed to work if enough buffer space is available, distributed routing is used, and the final routing algorithm does not consider previous message history (e.g., the input channel) when computing the output channel. Finally, solution 3 also requires distributed routing [5].

## 3. EXISTING SYSTEM

In the literature, the process of replacing a routing function with another one is traditionally referred to as network reconfiguration. It is well known that, even when both routing functions are by themselves deadlock-free, updating the routing paths in an uncontrolled way may lead to deadlock situations due to the introduction of temporary dependencies among network resources [7], [11]. This means packets that belong to one of the routing functions may take turns that are not allowed in the routing function.

Traditional reconfiguration mechanisms (designed for Autonet and Myrinet [4] networks) solved this problem by emptying the network of packets before replacing the routing function. Such a simple approach is referred to as static reconfiguration, and has a negative impact on the network service availability, since for a period of time the network cannot accept packets [1].

Fig. 1 shows the consequences of static reconfiguration in Autonet. The plot on the right shows a zoom of the left plot. Under normal conditions, network latency varies from 2,000 to 4,000 cycles. However, during reconfiguration, it reaches from one to three million cycles. Moreover traffic is stopped for longer than 1 million of cycles.

## 4. PROPOSED SYSTEM

Now, we describe a complete management mechanism which supports dynamic routing reconfiguration based on close graph properties. Fig.2 shows all steps involved in the change assimilation process. First, in order to detect topological changes, the manager may periodically poll the status of fabric devices, similar to the sweeping mechanism included in InfiniBand [8]. On the other hand, when a fabric device detects a change in the status of a local port, it may notify this event to the manager by means of a specific management packet. We consider this detection method in this work. Once the change is detected, the manager must compute a new routing graph. This phase consists of two sequential tasks: topology discovery (such as using [16]) and assignment of up*/down* directions.

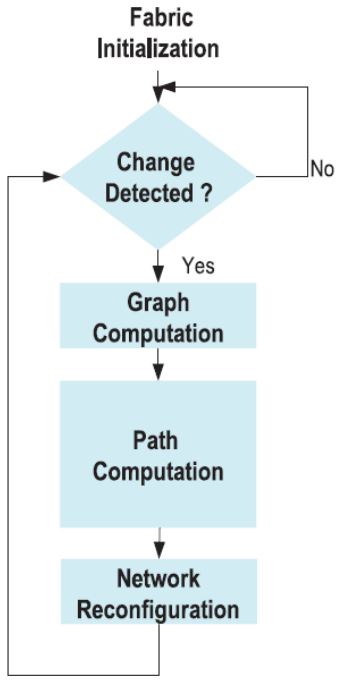


**Fig-2:** Fabric management mechanism phases

Starting from the discovered topology graph, our mechanism obtains a new up*/down* directed graph. After that, the manager performs the path computation phase. The last phase involved in the change assimilation process consists in replacing the old routing function by the new one. The manager sends the new paths to the network endpoints without considering any special sequence or synchronization. Each endpoint replaces the obsolete paths by the new ones as soon as they are received.

### 4.1. Up*/Down*

Up*/Down* is a well-known routing algorithm for irregular networks, initially proposed in Autonet, and later used in Myrinet. Up*/Down* routing allows for deterministic or adaptive routing as well as a choice of minimal or non-minimal routing. The core of Up*/Down* routing is to assign \Up" (and implicitly a \Down" the opposite way) directions to links, so that no cycle of Up links can be found. To avoid deadlock packets can travel zero or more Up links, then zero or more Down links, but no further transition to Up links are allowed. In calculating the routing table's further restrictions may be applied, e.g. ensuring minimal routing [10].

*Definition*: *A consistent Up*/Down* graph is a directed graph which has no cycles of Up links, no cycles of Down links, the root can reach all other nodes through exclusively through Down links, and all nodes can reach the root exclusively through Up links.*

There are many ways to assign the directions; a popular method is to find aspanning tree, assigning the Up direction towards the root of the spanning tree. For non-spanning tree links, the Up direction is assigned towards the node with the smallest spanning tree depth. In case of a tie, Up direction can be assigned towards the lower ID node. In Fig. 3, the node A has been chosen as the root. The black arrows of the figure are the links which belong to the spanning tree found from node A. The Up direction ofthese links are all towards the root node. The gray arrows are the links which do not belong to the spanning tree. For the link between F and G, the node F has a lower depth in the spanning tree than G, so the Up direction must be towards F. For the other gray link, between D and E, both nodes have the same depth. We assume alphabetical order which gives D a lower ID, thus the Up direction of the link is towards node D.

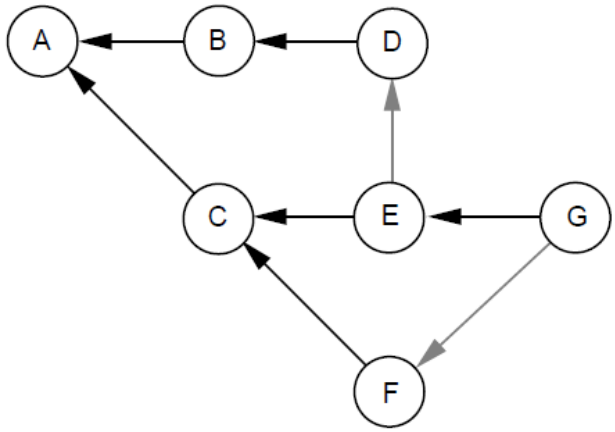

Fig-3: Up*/Down* link direction assignment

## 5. CONCLUSION

The process of replacing a routing function with another one is traditionally referred to as network reconfiguration. After the occurrence of a topological change such as the failure or aggregation of a network component, a new routing function, which is appropriate to the resulting topology, must be obtained and uploaded in the routing elements. even when both routing functions are by themselves deadlock-free, updating the routing paths in an uncontrolled way may lead to deadlock situations due to the introduction of temporary dependencies among network resources [7], [11]. This means packets that belong to one of the routing functions may take turns that are not allowed in another routing function. Static reconfiguration mechanisms solved this problem by emptying the network of packets before replacing the routing function. But it has a negative impact on the network service availability, since for a period of time the network cannot accept packets. So dynamic reconfiguration scheme is used in which data packets can be injected during the change assimilation process. Close up*/down* graphs are used to do this and while generating the new graph care should be taken that there are no cycles in newly generated graph to avoid deadlocks.

## REFERENCES

[1] Antonio Robles-Go´mez, Aurelio Bermu´dez and Rafael Casado, "A Deadlock-Free Dynamic Reconfiguration Scheme for Source Routing Networks Using Close Up*/Down* Graphs" IEEE Trans. Parallel and Distributed Systems, vol. 22, no. 10, pp. 1641-1652,Oct. 2011.

[2] D. Avresky and N. Natchev, "Dynamic Reconfiguration in
Computer Clusters with Irregular Topologies in the Presence of Multiple Node and Link Failures," IEEE Trans. Computers, vol. 54, no. 5, pp. 603-615, May 2005.

[3] A. Bermu´ dez, R. Casado, F.J. Quiles, and J. Duato, "Fast Routing Computation on InfiniBand Networks," IEEE Trans.
Parallel and Distributed Systems, vol. 17, no. 3, pp. 215-226, Mar. 2006.

[4] N.J. Boden, D. Cohen, R.E. Felderman, A.E. Kulawik, C.L. Seitz, J.N. Seizovic, and W. Su, "Myrinet: A Gigabit-per-Second Local Area Network," IEEE Micro, vol. 15, no. 1, pp. 29-36, Feb. 1995.

[5] R. Casado, A. Bermu´ dez, J. Duato, F.J. Quiles, and J.L. Sa´nchez, "A Protocol for Deadlock-Free Dynamic Reconfiguration in High- Speed Local Area Networks," IEEE Trans. Parallel and Distributed Systems, vol. 12, no. 2, pp. 115-132, Feb. 2001.

[6] W.J. Dally and C.L. Seitz, "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks," IEEE Trans. Computers, vol. C-36, no. 5, pp. 547-553, May 1987.

[7] J. Duato, O. Lysne, R. Pang, and T.M. Pinkston, "Part I: A Theory for Deadlock-Free Dynamic Network Reconfiguration," IEEE Trans. Parallel and Distributed Systems, vol. 16, no. 5, pp. 412-427, May 2005.

[8] InfiniBand Trade Assoc., InfiniBand Architecture Specification Release 1.2, http://www.infinibandta.org/, 2004.

[9] O. Lysne and J. Duato, "Fast Dynamic Reconfiguration in
Irregular Networks," Proc. IEEE Int'l Conf. Parallel Processing (ICPP '00), 2000.

[10] O. Lysne, J.M. Montan˜ ana, J. Flich, J. Duato, T.M. Pinkston, and T. Skeie, "An Efficient and Deadlock-Free Network Reconfiguration Protocol," IEEE Trans. Computers, vol. 57, no. 6, pp. 762-779, June 2008.

[11] O. Lysne, T. Pinkston, and J. Duato, "Part II: A Methodology for Developing Deadlock-Free Dynamic Network Reconfiguration Processes," IEEE Trans. Parallel and Distributed Systems, vol. 16, no. 5, pp. 428-443, May 2005.

[12] J.M. Montan˜ ana, J. Flich, and J. Duato, "Epoch-Based Reconfiguration: Fast, Simple, and Effective Dynamic Network Reconfiguration," Proc. Int'l Parallel and Distributed Processing Symp. (IPDPS '08), Apr. 2008.

[13] Myrinet, Guide to Myrinet-2000 Switches and Switch Networks, http://www.myri.com/, 2011.

**BIOGRAPHIES**

Ms. Deepti R. Metri, Student at N. B. Navale Sinhgad College of Engineering, Kegaon. She has completed his BE in Computer Science and Engineering.

Prof. Abhijeet V. Mophare, Assistant Professor at N. B. Navale Sinhgad College of Engineering, Kegaon. He has completed his ME in Computer Science and Engineering.