EE482, Spring 1999
Research Paper Report

Deadlock Recovery Schemes

Jinyung Namkoong **Nuwan Jayasena**
Mohammed Haque **Manman Ren**

May 18, 1999

Introduction

The selected papers address the problems of deadlock, which is a critical issue in interconnection networks. It occurs as a result of cyclic resource (channel or buffer) dependency. Since all the packets involved in a deadlocked configuration are blocked forever, any interconnection network design should address this problem properly. The two alternatives for dealing with deadlocks are prevention (avoidance) and recovery. Prevention is the traditional approach to deadlocks. Prevention based routing algorithms enforce certain routing restrictions in order to avoid deadlocks. So, in exchange for some loss in adaptivity, (in the form of dedicated virtual channels or routing options reserved for escape from deadlocks,) the network is guaranteed never to enter the deadlock condition. On the other hand, recovery-based approach allows packets to use all the virtual channels, and don't put any restrictions on routing. Consequently, the degree of adaptivity is much higher for recovery-based algorithm compared to prevention-based algorithm with the same number of virtual channels. Which of these two alternatives works the best chiefly depends on the frequency of deadlock situations, as well as many other network parameters such as packet length, buffer configuration, and flow control.

Both prevention and recovery schemes achieve the same thing: namely to prevent deadlocks from stalling the system indefinitely. The constraint is really the cost and the performance, as is with any engineering problem. Since there are many variables involved, it's important to set up a standard measure for comparison. Paper [1] gives one such option for comparison. In this paper, the total number of blocked messages in the network, potential for number of cycles in the network, as well as probability of deadlock are used to compare different routing schemes. The performance of different routing schemes is ultimately measure by the load at which the network saturates. The conclusion of this paper is first that deadlock situations occur very infrequently in most interconnection networks. And since this probability of deadlock decreases exponentially with routing adaptivity, it's best to make the common case faster and deal with deadlocks as special cases. This calls for fully adaptive routing for all virtual channels, rather than only on subset of virtual channels as in deadlock prevention scheme. In this report, Two approaches to deadlock recovery routing scheme are presented. One is DISHA which uses a central buffer space for special lane for deadlocked packet [2], and CR which uses the compressionless property of wormhole routing to guarantee deadlock-free network [3].

The key issues in analyzing and comparing these different routing strategies are performance as measured by saturation load (the fraction of network capacity, at which the latency-load curve makes a sharp turn upward), and hardware complexity, which determine the router delay as well as cost.

Two Recovery Schemes: DISHA and CR

DISHA Recovery Scheme

DISHA is designed to provide a simple, cost effective, and efficient routing strategy which will ensure an optimized performance in the absence of deadlock by dedicating all the available virtual channel (VC) at each node to the packets. In this true fully adaptive routing (TFAR), deadlock condition may occur. To break from this, one special buffer per router is used for escape path. Basically, the only difference compared to Duato's algorithm is that this escape buffer, which may be thought of as a special kind of VC, is central to all the input ports, whereas in Duato's, there are certain number of VCs per input port which have restricted routing. Thus, at some added cost of implementing this special kind of buffer, all the VCs associated with input ports can be used in fully adaptive routing. [2] claims that the router design is very simple (compared to prevention-based scheme with the same number of fully adaptive VCs,) and that this router can be used for any topology.

As with any recovery-based approach, the detection scheme has a significant impact on the performance. Since accurate detection of deadlock can be costly and inefficient, DISHA uses simple timer-based detection scheme. Basically, if a packet header is blocked for longer than a pre-defined time, the router assumes that this packet is potentially deadlocked and takes appropriate actions. Though simple to implement, this can cause many wrong detection (packet that's merely waiting for a busy channel,) or choose a wrong packet that's affected by deadlock but not actually a part of the deadlock cycle. Therefore, the choice of the timer limit is essential.

Actual recovery from deadlock can be done either sequentially or concurrently. In sequential recovery scheme, the router need to have exclusive access to the deadlock buffer before it can place a packet on it. Once access is gained, it places any one of the packets from the deadlock cycle into the buffer (since there is no order in which packet from the deadlock cycle can be chosen) and asserts the status line to indicate to the subsequent router that this packet should be placed on the deadlock buffer. The router that has been implemented in [2] follows this strategy and routes the packet to its destination through a minimal routing path. On the other hand, concurrent routing doesn't require the routers to have exclusive access to the Deadlock buffer. This allows parallel recovery from single deadlock cycles and simultaneous parallel recovery from multiple cycles, which may speed up the recovery process, but needs to be verified deadlock-free.

Implementing DISHA:

Deadlock Detection:

Deadlock detection can be done using a time-dependent selection function. An elapsed time $T_{elapsed}$, which is the number of cycles that a router fails to send out the header, can

be compared to a predetermined upper bound T_{out}, which is the maximum number of cycles that a packet can experience congestion. Once $T_{elapsed}$ reaches T_{out}, the network is considered to be deadlocked.

Deadlock Recovery with DISHA:

Deadlock recovery with DISHA can be done either by flit-by-flit router switch (crossbar) allocation policy where the crossbar is reconfigured for each flit or by packet-by-packet allocation policy where reconfiguration of crossbar is done for each packet.

While employing the flit-by-flit allocation policy, a packet, after waiting at an input port of a router T_{out} cycles, is placed on the deadlock buffer with appropriate status line asserted. The following router then places this packet on the deadlock buffer, and it continues until it reaches the destination. At each router the output port is assigned to the flit on the deadlock buffer by the arbitration and decision logic that the router implements. But in the case of packet-by-packet policy, reconfiguration of crossbar is different. In a Deadlock situation it may appear that a Deadlocked packet on the deadlock buffer needs to go to the same output port as is currently used by some other packet. Since reconfiguration is not done for each flit, the decision logic needs to remember the current status of the crossbar allocation before tearing it down to accommodate for the deadlocked packet. After the deadlock packet is cleared, it will then revert the configuration to its saved state after the deadlock is cleared.

Advantages of DISHA:

Advantage that DISHA offers are that it provides a true fully adaptive wormwhole routing by implementing deadlock recovery scheme rather than deadlock prevention scheme where the routing suffers loss of adaptivity and performance by restricting routing in some particular directions.

DISHA also doesn't require any dedicated virtual channel to implement its deadlock recovery scheme. This yields faster and simpler router architecture. For example, if, in DISHA, 4 virtual channels are allocated for each input port, then for a router, that dedicates 2 virtual channels for deadlock recovery, a total of 6 virtual channels are needed per port to achieve the same performance as DISHA. This would result in a more complicated router architecture for that particular router as compared to DISHA. This also means that DISHA is more cost effective, since more VC would result in a more expensive router.

Since DISHA does not employ abort-and-retry scheme as opposed to Compressionless routing, it need not store the flit after it has been served from the buffer, which also implies that no work that has been done towards the routing of a flit is wasted. Moreover, since in Compressionless routing, abort-and-retry packets are killed, they suffer more latency compared to the ones that are not killed. But in the case of DISHA, this excess latency is not incurred.

In DISHA the buffers for VCs are only 2 flit wide, therefore, it burdens the router with a lot less overhead. Moreover, if the number of virtual channels is not a power of 2, then, without any additional cost, the leftover states can be utilized to assert the status line when packet needs to be routed through the deadlock recovery buffer.

Disadvantages of DISHA:

There are some weak points in DISHA. One is the extra logic required to implement the token logic. The deadlock buffer is central to the whole router. If more than one input port receives packets destined to the deadlock buffer, there arises a need to arbitrate and assign a token to one of the packets to have access to the deadlock buffer. This requires an extra logic implementation in the router, which results in more complicated router architecture than CR routers, and needs more hardware.

As stated above, DISHA implements either flit-by-flit or packet-by-packet port assignment for the input and output ports of the router. If the packet-by-packet scheme is employed, the switching logic becomes more complex, and requires more hardware since it has to save the switching status before it reorganizes the input-output connection in order to route through the deadlock-buffer.

Compressionless Routing

Compressionless Routing (CR) [3] is another adaptive routing framework that utilizes detection and recovery from deadlock instead of dedicating hardware resources to prevent this infrequent phenomenon..

CR is based on the observation that deadlock cannot occur after the head of a packet reaches its destination. If the packet being transmitted is long enough to guarantee that its head reaches the destination before the last flit enters the network, the source can monitor the progress of the packet and detect if it blocks. The minimum required length of a packet to guarantee the above is (B_{cap} . $(D + 2 . M_{mis})$), where B_{cap} is the depth of a channel buffer in flits per channel, D is the distance in hops from source to destination, and M_{mis} is the maximum number of misroutes allowed. Any packet that is too short can be padded by the sender to meet the above requirement. Figure 1 illustrates the flow through the network for long and short packets. If a packet remains blocked beyond a particular predetermined interval, the sender can determine that a potential deadlock condition exists and take corrective action. This simple methodology is used in Compressionless Routing to guarantee freedom from deadlocks. Once a potential deadlock is detected, the source sends a *KILL* signal (on a separate wire) down the path allocated for the blocked packet. This signal will free the resources occupied by the deadlock-suspect packet at each node until it “catches up” with the head of the packet, at which point it will terminate. The source then attempts to re-transmit after a delay. It is straightforward to extend the CR paradigm to provide fault tolerance by padding all packets to ensure the last flit (not just the head) reaches the destination before the sender

finishes transmitting. This aspect, however, is not discussed here as our focus is deadlock recovery.

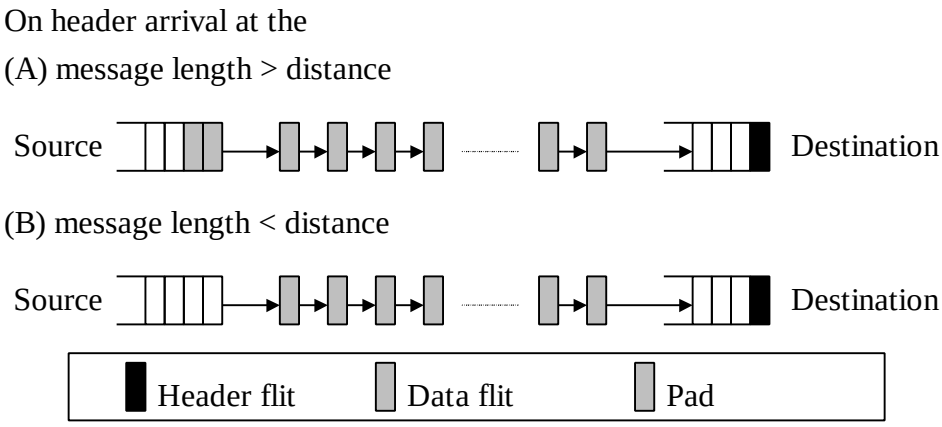

Figure 1: Flow through the network for (A) long and (B) short

Implementation of CR

One of the main advantages of CR is the simplicity of its implementation. Backpressure mechanisms already present in wormhole routing can easily be used to detect when a message blocks. The only extra logic required at the sender is the ability to calculate the distance to a given destination and accordingly pad short packets, and time-out when no progress is made by a packet for a certain period and issue a *KILL* signal. Each router also needs to be able to act according to a *KILL* signal it may receive, and forward it on as appropriate. The receiver needs the ability to drop the packet currently being received if a *KILL* signal is received on that channel.

Performance of CR

The authors of [3] provide performance measures based on register-transfer level simulations to compare CR with dimension order routing (DOR) and to study the effects of varying the time a packet is allowed to remain blocked before assuming the formation of deadlock. The simulations are for 16x16 and 8x8x8 torus networks using 32 flits per packet random traffic. The results show that CR performance is optimal for a time-out equal to the message length. The authors argue that this time-out period is probably inadequate to identify true deadlocks, and therefore, packets blocked due to channel congestion are also being killed and re-transmitted on different paths contributing to improved performance. When compared to DOR with 2 virtual channels (VCs), CR with a single virtual channel performs as well or better in most cases, particularly in the 3D network. CR with two virtual channels significantly outperforms DOR with 2 VCs.

Discussion of CR

In addition to requiring minimal additional logic to implement, Compressionless Routing has several other advantages. Unlike static or dynamic prevention techniques [5, 7, 4, 6], CR does not inherently require virtual channels to guarantee deadlock freedom. Single VC implementations can significantly reduce the complexity of routing decision logic and channel control. At the same time, CR also does not limit routing freedom as done by the Turn Model [8], thereby providing full routing freedom. Ordered message delivery can also be guaranteed in the absence of multiple network ports per node since the sender does not transmit another message until the recipient has started receiving the previous one. CR is also applicable to a variety of topologies and can easily be extended to provide fault tolerance.

The simplistic scheme used to detect deadlock in CR also introduces some drawbacks. Many of these drawbacks arise from having to pad short packets leading to lower channel utilization for useful data. For networks with shallow channel buffers and short diameters, the padding overhead may not be significant. If, however, the network has deep buffers or a large diameter, the padding overhead may increases significantly. The degree to which deep buffers degrade performance cannot be ascertained as no performance measures are provided for channel depths greater than 2 flits per node. While the authors argue that larger buffers can be accommodated by using virtual channels, this nullifies a major advantage of CR -- the ability to provide deadlock free routing without the complexity of multiple virtual channels. The need for shallow buffers also means that this technique can only be applied to networks employing wormhole flow control. The amount of padding needed also grows linearly with the number of misroutes allowed, imposing a practical limitation on the freedom to misroute as well. Another disadvantage of CR is the abort-and-retry approach utilized in cases of potential deadlock, which wastes the work done in routing the messages thus far. The retransmissions also imply a need to buffer the entire message at the source at least until the head reaches the destination. This could lead to significant increases in buffer space requirements, particularly considering that long message sizes are required in CR to reduce padding overhead.

Performance Comparison of CR and DISHA routing schemes

The papers [2] and [3] have some latency vs. load-rate graphs for comparing the performance of deadlock recovery schemes (CR and DISHA, respectively) with other deadlock prevention schemes. Sure enough, these recovery schemes perform better than prevention schemes with similar buffer requirements. Though there can be other measures of comparison, (such as deadlock frequency and number of blocked packets,) the most relevant and simple metric seems to be the load at which the network saturates. This saturation load is defined to be the fraction of maximum network capacity at which the latency curve approaches vertical asymptotic line.

All of these performance measurements depend heavily on the number of VC/PC and the traffic type, as well as many other network parameters, which makes comparisons from different papers somewhat difficult. Fortunately, [2] and [3] had similar experiments for random traffic on 16x16 torus network, with packet size of 32 flits and VC buffer depth of 2 flits. According to [2], CR with two VC/PC saturates at load-rate of ~0.35, whereas DOR (dimension order routing) with two VC/PC saturates at ~0.22. Now, according to [3], DISHA with four VC/PC saturates at ~0.8 when up to three mis-routes are allowed, and ~1.0 when only minimal paths are allowed. It also marks DOR with four VC/PC at ~0.5. If these separate results can be directly compared, since the load rate for DOR has increased by ~2.5 when VC/PC is changed from two to four, it can be surmised that the same should be true for CR as well. In that case, the load rate for CR with four VC/PC would be ~0.85, which brings it very close to the performance of DISHA. Of course, it seems obvious that CR cannot outperform DISHA, simply because it uses abort-and-retry method.

Summary and Discussion

The choice between deadlock recovery and deadlock prevention schemes is certainly a valid question, which profoundly affects the performance of interconnection network. Making the common case fast has proven to be the right approach for many aspects of computer architecture. In this case also, the frequency of deadlock instances has been shown to be low enough to justify recovery approach. Since this approach treats deadlocks as special cases, the same router can be applied to many different topologies as long as the assumption about infrequency of deadlocks still holds true, thus freeing the designers from worrying about cumbersome deadlock issues while creating or expanding a network.

The critical issues for dealing with deadlocks is, as in most engineering problems, cost and performance. Papers [2] and [3] respectively addresses such advantages of CR and DISHA over other prevention schemes. Both claim to achieve better performance while using a very simple routers.

CR is an interesting way to deal with deadlock issue by using the property of wormhole flow control itself. It's like turning a bug into a feature, since the fact that wormhole routing spans many routers is not a good news for deadlock. It can potentially have large overhead as well as some limitations as listed in the previous section. However, due to its simple router design and small overhead in terms of hardware, it would be a viable solution for integrated router on a processor chip, with not too demanding performance requirement. Used in that context, it would use fewer chips per channel, which can help spread out a packet over multiple routers, thus resulting in smaller overhead. Also, in that case, since area is of a primary concern, it'd be a good idea not to buffer an entire packet at the source, but instead use the higher level software to deal with re-transmitting.

DISHA is a more general and efficient routing scheme. It derives its power from the fact [1] that the probability of deadlocks decreases exponentially with the routing freedom. Since the lower the frequency of deadlocks, the better the recovery strategy is, it's good

to have as much routing freedom as possible, which is precisely what DISHA achieves by using all VCs for possible routing. Deadlock prevention schemes with similar performance would require two more VCs per port, which can significantly increase the router complexity. Its efficiency is enhanced by the fact that upon detection, deadlock is resolved progressively, rather than regressively. That means that no packet is dropped and retransmitted.

Other Key Issues

An important part of deadlock recovery is the detection. Both CR and DISHA uses the timer based approach of detecting potential deadlocks. The authors state that the decision of this expiration time is a critical decision for network performance. Low time limit would raise a lot of false deadlock conditions, whereas high time limit would cause a deadlock to multiply through the network before it's detected and eliminated. [2] mentions that for its particular experiment with uniform traffic, timing limit equal to packet length gave the best performance, but it depends heavily on the type of traffic and topology. So, optimal, and possibly adaptive, decision of this timing limit is to be desired. Also, instead of relying on timer for inaccurate detection, one can look at alternative way of detection, without incurring too much overhead.

Another area requiring further research is to analyze exactly how the topology and traffic pattern affect the optimal choice for dealing with deadlocks. There are so many variables that significantly affect the performance so that it'd be helpful to come up with some kind of heuristic or analytical formula that will help designers choose the right deadlock method.

Neither of the papers address the issue of QoS (quality of service), which can be an important issue for real time applications. Since no priority information is encoded into packet, and no special channels are reserved for special packets, there's no guarantee in terms of latency. If QoS is to be provided, priority information needs to be encoded and the protocols expanded for both schemes.

Another issue is to prevent network saturation from occuring, by using some kind of throttling mechanism from the injectors. From [1], it can be seen that the network congestion (which is proportional to the number of blocked messages in the network) increases the probability of deadlocks as well as exponentially increasing the size of such deadlocks.

References

[1] T.M.Pinkston and S. Warnakulasuriya, On deadlocks in interconnection networks, *Proceedings of the 24th ISCA*, June 1997

[2] Anjan K. V. and T. M. Pinkston, An efficient fully adaptive deadlock recovery scheme: DISHA, *Proceedings of the 22nd ISCA*, pp.201-210, June 1995

[3] J. H. Kim, Z. Liu, and A. A. Chien. Compressionless routing: a framework for adaptive and fault-tolerant routing. *In Proceedings of the International Symposium on Computer Architecture*, pages 289-300, April 1994.

Additional references:

[4] A.A. Chien and J.H. Kim. Planar-adaptive Routing: Low cost adaptive networks for multiprocessors. In *Proceedings of the International Symposium on Computer Architecture*, pages 268-77, May 1992.

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

[6] J. Duato. On the design on deadlock-free adaptive routing algorithms for multicomputers: design methodologies. In *Proceedings of Parallel Architecture and Languages Europe*, 1991.

[7] D. Linder and J. Harden. An adaptive and fault tolerant wormhole routing strategy for k-ary n-cubes. *IEEE Transactions on Computers*, C-40(1):2-12, January 1991.

[8] L. Ni and C. Glass. The turn model for adaptive routing. In *Proceedings of the International Symposium on Computer Architecture*, 1992.