History-Aware Adaptive Routing Algorithm For Energy Saving in Interconnection Networks

Hai Nguyen, Gonzalo Zarza, Daniel Franco and Emilio Luque
Computer Architecture & Operating Systems Department
Universitat Autònoma de Barcelona,08193 Bellaterra, Spain
hai.nguyen@caos.uab.es, gonzalo.zarza@uab.es, daniel.franco@uab.es, emilio.luque@uab.es

Abstract— *The increase of link speeds in the interconnection networks is evident both inside and outside of a datacenter. Thus it contributes an increasing portion of the power budget of the interconnection system. Link power management has been receiving more attention and many mechanisms were proposed. The emerging bit-serial link technology allows the links to work with different numbers of lanes & speeds. When the traffic load is slight, links are put in low-speed mode and consume less energy. However, links working in the low speed mode result in the increase in serialization latency. We propose a routing algorithm that takes into account the history usage of the links to focus network traffic in a small subset of high-speed links. It keeps high-speed links busier and leaves low-speed links with more idle time. Thus the mechanism saves energy and reduces the incurred serialization latency.*

Keywords: energy saving, history-aware, routing algorithm, interconnection networks

1. Introduction

Network components consume 10-20% of the total power of an interconnected system [1]. The energy consumption and heat dissipation problem for interconnection systems make the need for a more efficient networking become more evident. Network links contribute a major portion of the energy required for the network, around 58% [2]. However the link utilization in the interconnection system is low. We have conducted simulations with 64 processing nodes arranged in fat tree topology [3]. The traffic patterns are imported from two benchmarks modeling Black-Scholes partial differential equation and Fluid Animate Particle Simulation using Smoothed Particle Hydrodynamics [4]. For both applications the average link utilization is lower than 5%. The energy consumption for links is almost insensitive to the fluctuation of the traffic intensity, thus they burn the same amount of energy while working very little. The average link utilization will be less in the future with the ever-increasing link speed [5]. Besides, for a particular traffic pattern, the link utilization is not spatially uniform. There are some links that are almost idle and others that are much busier. Less energy consumption for those almost idle links is a desired behavior. Many studies have been focusing on a better link power management. Researchers have proposed many mechanisms for the better use of links with different approaches.

The first approach is dynamically turning a number of links on and off as the function of traffic [6], [7], [8], [9], [10]. In this approach, a link activation request can be sent from a sending node by inherent system events to reduce the link re-activation time overhead [11]. When applying this approach the path diversity is greatly reduced and the deadlock avoidance becomes an issue. Another approach is the Dynamic Voltage Scaling (DVS) mechanism to dynamically adjust link frequency and voltage with the history-based policy of the link utilization [2]. This approach has the potential to save a significant amount of energy in the link components even though it introduces more complexity in the hardware design. Another approach for the link power management control is to judiciously adjust the width of the link [5], [12], [13]. With the use of bit-serial link technology, where every link consists of multiple lanes, the link width control mechanism works naturally. Links in PCI-Express are available in up to 16-lane configuration (denoted as $x16$). Similarly, Infiniband has made available the multi-lane links with $x4$ and $x12$. Our work focuses on the last approach.

To date, the dynamic link width mechanism is applied with a history-oblivious routing algorithm. Traffic is spreaded through many links to prevent and alleviate congestion situation. However, in light traffic load scenarios, some fraction of the traffic is routed through low-speed links while other high-speed links still do not work at full capacity. Thus the mechanism incurs an additional serialization latency. To solve this issue, we propose a history-aware adaptive routing algorithm that prioritizes the route of traffic on a small subset of links and focuses the traffic on that subset. This subset of links has a high value of link utilization and thus will be kept at high speed, consequently the serialization latency is reduced. Moreover, as a side effect the routing algorithm at the same time leaves more links idle and thus gives them more opportunity to be adjusted to the low-speed mode and save more energy.

This paper makes a contribution in proposing a history-

The 2013 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'13)

aware adaptive routing algorithm. It makes the comparison about the latency behavior and the energy consumption between history-aware and history-oblivious routing algorithms. It also conducts experiments, analyzes results for both synthetic traffic and traffic imported from trace files.

The paper is organized as follows: In section 2, the basics of the dynamic link speed mechanism based on dynamic link width is presented. It involves the Monitoring & Decision Making phase. Section 3 describes the history-aware adaptive routing algorithm. Section 4 illustrates and explains the experimental results. Finally, we draw conclusions in section 5.

2. Dynamic Link Speed Mechanism Basics

The dynamic speed behavior of a link is achieved by varying its width according to the fluctuation of traffic on it. The main process of the mechanism is *Monitoring & Decision Making*, where the link activities are monitored to decide whether to change the link width.

The monitoring and decision making process involves detecting when to change the link speed. Link Utilization (LU) is monitored at the port basis. The mathematical definition of LU is presented in the equation 1.

$$\textbf{LU} = \frac{\sum_{t=1}^{H} A(t)}{H} \tag{1}$$

Where A(t) = $\begin{cases} 1 \text{ if traffic passes in cycle } t \\ 0 \text{ if no traffic passes in cycle } t \end{cases}$ and H is a sliding history window size.

The value of LU is less than 1, and it directly reflects how frequent a link was used. The larger the value of LU, the busier the link is. When the value of LU drops below the threshold value th_low and the link is not at its minimum width, the mechanism triggers the link to reduce its width one level. To simplify the routing algorithm, avoid deadlock avoidance issue and reduce the re-activation time overhead, a link is never turned off and it never has the width of 0. In contrast, when LU exceeds the threshold th_high and the link is not at its maximum width, the mechanism increases the link width one level.

Another criterion for the mechanism is the input buffer occupancy of the next router. This information is available for the mechanism based on the credit-based flow control of the router. If the input buffer occupancy at the far end of the link is higher than the threshold $th\_buffer\_occupancy$, it is the signal indicating that the network is congested at the far end of the link. The movement of packets in that situation is restricted by the availability of the buffer space, not the link bandwidth. Thus the mechanism can reduce the link width more aggressively without sacrificing the serialization latency.

Algorithm 1 Changing Link Speed Decision

```
**Monitoring the next input buffer occupancy**
**if** $buffer\_occupancy > th\_buffer\_occupancy$ **then**
    $th\_low$= $th\_low\ for\ congestion$
    $th\_high$=$th\_high\ for\ congestion$
**else**
    $th\_low$=$th\_low\ for\ non-congestion$
    $th\_high$=$th\_high\ for\ non-congestion$
**end if**
**Monitoring the $LU$ value of the link**
**if** ($LU$ of the link $< th\_low$) **and** (The link is not at minimum width) **then**
    Decreasing The Link Width
**end if**
**if** ($LU$ of the link $> th\_high$) **and** (The link is not at maximum width) **then**
    Increasing The Link Width
**end if**
```

The decision making process is made every period of time T. The pseudo code for this mechanism is described in Algorithm 1.

The values of the thresholds are configurable and they are configured depending on the objective of the network. The higher the values of th_low and th_high the more agressive the mechanism triggers links to reduce their width to save more energy. The difference between th_low and th_high also should be carefully selected to avoid the link flip flop when traffic fluctuates often.

3. History-Aware Adaptive Routing Algorithm

With the typical path redundancy of network configurations (to facilitate the load balancing and fault tolerance), at every intermediate router, there might be several output ports a packet can take to make the progress towards its destination. For example, for k-ary n-cube network topology, every router has n productive ports for packets to come closer their destination.

For a network applied the Dynamic Link Width mechanism, any given router connects with its set of links that are at different speeds. Any port p_i couples with a link with the link width value of W_i, the input buffer occupancy at the far end of the link has the value of $Buffer\_Occupancy(p_i)$ as illustrated in Fig. 1. In this figure, the link couples with port p_1 is at a high link width level and thus has a higher bandwidth compared with the links coupled with the other ports p_2,...,p_k. It is preferred that packets move on the link connected with port p_1 to have less serialization latency.

However, if the routing policy is oblivious about the history usage of the links and spreads packets through many links then all the links have a low average utilization values

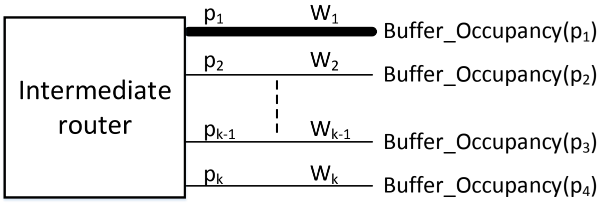


Fig. 1: Multi-port with different link speeds

and thus being put at a low speed. Consequently the average packet latency increases due to the serialization latency by moving in thin links. In the low load situation, a better routing policy that focuses traffic in a subset of high speed links while leaves other links idle is a desired behaviours.

At every router, with a set of compatible output ports, the history-aware routing policy gives the decision about which output to take with the preference for the most-recently-used port, unless there are a strong evidence not to do so but using a normal adaptive routing instead. A normal adaptive routing algorithm will be used if one of the following conditions hold:

- All the links of the router are at maximum speed. This is the situation when high traffic load is present in the network and the routing policy should balance the traffic by spreading packets to many links.
- The input buffer occupancy at the far end of the link coupled with the most-recently-used port is higher than the threshold value of $buffer\_threshold$. This is the signal that the link is over-utilized and the continuity of traffic injection to them can lead to congestion.

The proposed routing algorithm takes advantage of high bandwidth links in the case of low traffic load. In high traffic load situation when all links are in high utilization and thus in high-speed status, normal adaptive routing algorithm takes place. The routing decision is summarized in Algorithm 2.

Algorithm 2 History-Aware Adaptive Routing Algorithm

```
Getting the set of compatible output ports
Choosing the most recently used port as the $outport$
**if** All links are at maximum special **or** The buffer occupancy higher than $buffer\_threshold$ **then**
    $outport$=Normal_Routing_Algorithm()
**end if**
Exporting the $outport$
```

Since the topology of the network does not change, no special care for deadlock avoidance techniques is required. When taking into account the link history usage, it results in having some maximum-speed links carry a large fraction of the traffic load, while others links are mostly idle and put in low-speed mode. It is preferred to have a small fraction of links to work at full capacity and deliver the majority of traffic. With packets moving in a subset of high-speed links according to the proposed routing algorithm, the serialization latency is reduced. Besides, when the majority of traffic moves in the small fraction of high-speed links, the other fraction of links is almost idle and being put in a low-speed status, the energy consumption is further reduced.

4. Experimental Results

In this section, we evaluate the History-Aware Adaptive Routing Algorithm versus a History-Oblivious Routing Algorithm, both of them are applied in the interconnection networks with the dynamic link width mechanism to save energy.

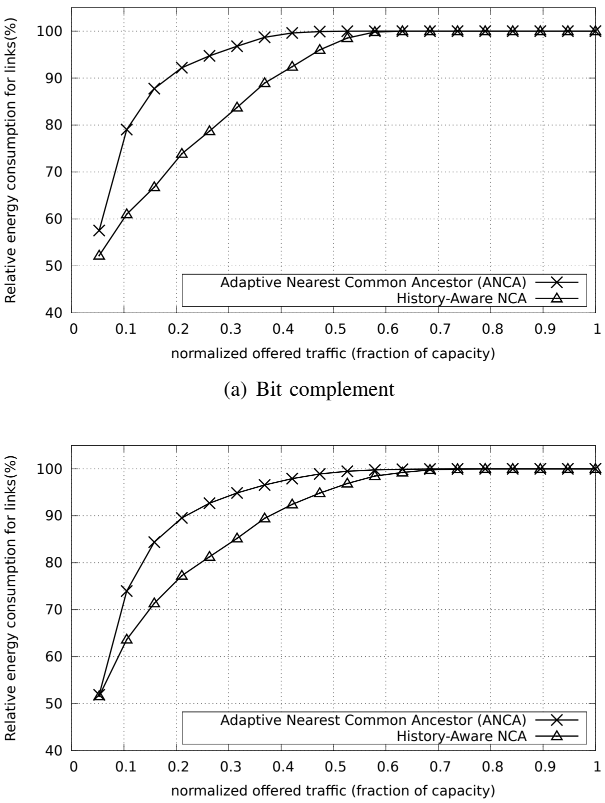
The framework for the simulation is the modified version of $booksim$ [14]. The interconnection network is configured with 64 processing nodes, arranged in the 4-ary 3-stage fat tree topology with virtual channel flow control. There is 16 virtual channel for each link, the virtual channel buffer size is 16 flits, a packet consists of 4 flits. To minimize the impact of the mechanism to the average packet latency, only links between routers are considered to adjust the width. Thus the communication between a processing node and its immediate connected router is performed with maximum speed.

Traffic patterns directly impact to the efficiency of the energy saving mechanism and the history-aware routing algorithms. We have conducted the experiments with both synthetic traffic and traffic imported from trace files.

The synthetic traffic patterns generated are the $bit\ complement$ and $uniform$ patterns. The th_low, th_high in non-congested situation are 0.2 and 0.6, in congested situation are 0.35 and 0.75 accordingly. The $th\_buffer\_occupancy$ to detect congestion for the next immediate input buffer is 0.5. The number of lanes for every link is 12. Energy consumption for the link is assumed to be proportional to the number of active lanes. The relative link energy consumption in the results is the percentage of energy consumed for the link component of the network when they work with and without dynamic link width mechanism.

To see the impact of the history-aware adaptive routing algorithm in action, simulations were conducted with two different routing algorithms. We have used Nearest Common Ancestor (NCA)[15] and History-Aware Nearest Common Ancestor (HA-NCA) as specified in section 3.

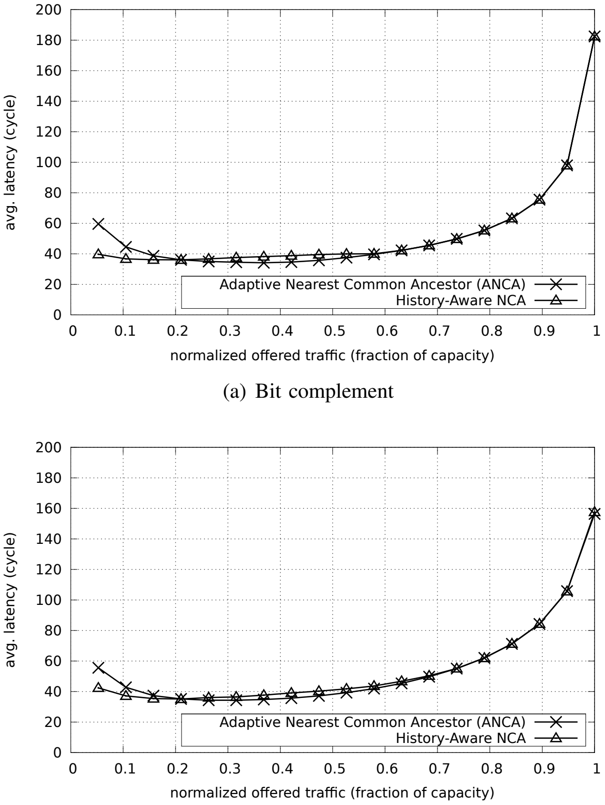
As we can see from Fig. 2 for both traffic patterns, when applying the dynamic link width mechanism the relative link energy consumption is proportional to the traffic load. With low traffic load a large fraction of links is put in low-speed mode and consume less energy. In contrast, in the high traffic situation almost all the links are at maximum width and speed, then the energy consumption is equal to the energy consumption when no saving mechanism is applied. The history-aware adaptive routing algorithm gains more energy



(a) Bit complement

(b) Uniform

Fig. 2: Relative Link Energy Consumption



(a) Bit complement

(b) Uniform

Fig. 3: Latency behavior comparison

saving because it makes a better use of a small fraction of links carrying the traffic and leave the others links idle.

Fig. 3 shows the latency behavior when applying the two routing algorithms. Even though the history-aware adaptive routing algorithm gains more energy saving, the latency behaviors for both of them are similar. Thus the proposed routing algorithm saves more energy without sacrificing the average packet latency.

As aforementioned, when applying dynamic link width mechanism the network incurs an additional serialization latency. Fig. 4 depicts the relationship between the latency behavior of the network with and without applying the saving mechanism. In both situations the history-aware routing algorithm was deployed. As we can see there is only a slight increase in average packet latency as opposed to a larger percentage of energy consuming reduction in Fig. 2.

To compare the impact of two routing algorithms with traffic imported from trace files. The traffic for the application Fluid Animate Particle Simulation using Smoothed Particle Hydrodynamics [4] was imported into the network with the same aforementioned network configuration using the Netrace framework [16]. Two routing algorithms are again put into comparison.

Without the energy saving mechanism the average packet latency is 37.81 cycles. The latency behaviors and the the relative link energy consumption when applying the dynamic link width mechanism with different routing algorithms is described in Table 1.

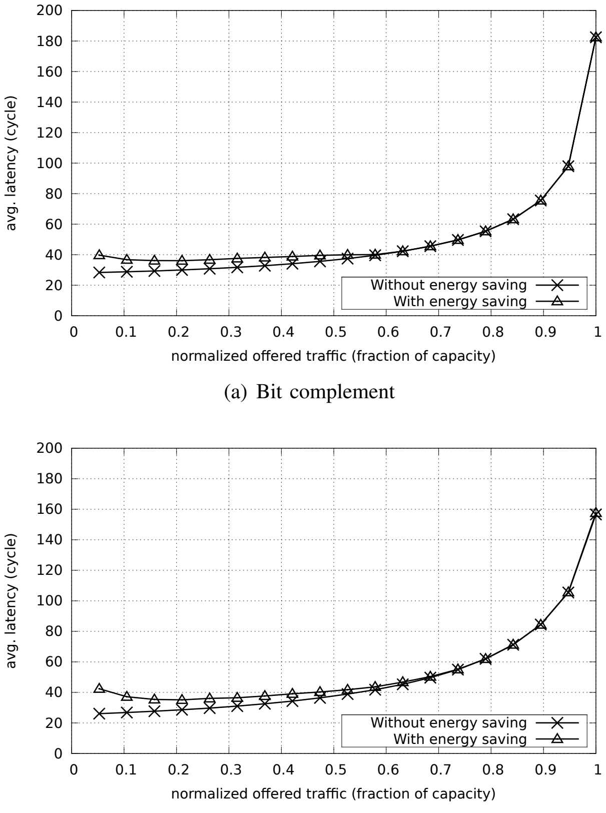
For this particular traffic pattern the relative link energy consumption is around 39.55% compared with the energy consumed by the links of the default system (without applying the mechanism). There is a typical trade-off where the energy consumption reduction comes with an increase in average packet latency. The energy saving gained by the dynamic link width mechanism is almost the same when applying History-Aware (HA-NCA) or History-Oblivious (NCA) Routing Algorithms. However with HA-NCA the increase in latency is lower than the NCA (29.60% as opposed to 56.75%).

5. Conclusions

We have proposed a history-aware adaptive routing algorithm to take the link history usage into account when making the routing decision. Our algorithm focuses traffic on a subset of high-speed links and put more links into the low-speed mode in the low load situation. The result is more energy saving is achieved with less additional average

Table 1: History Oblivious & Aware Routing Algorithm Comparison

	No saving mechanism applied	**Applied with NCA**	**Applied with HA-NCA**
Average Packet Latency	37.81 cycles	59.26 cycles	48.99 cycles
Percentage Latency Increase	0%	56.75%	29.60%
Relative Link Energy Consumption	100%	39.52%	39.58%



(a) Bit complement

(b) Uniform

Fig. 4: Latency behavior with/ without mechanism

packet latency. Our future work includes further applying this adaptive routing algorithm to the dynamic link width mechanism for energy saving.

Acknowledgments

This research has been supported by the MEC-Spain under contract TIN2011-24384.

References

[1] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, “The cost of a cloud: research problems in data center networks,” *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 1, pp. 68–73, Dec. 2008. [Online]. Available: http://doi.acm.org/10.1145/1496091.1496103

[2] L. Shang, L.-S. Peh, and N. Jha, “Dynamic voltage scaling with links for power optimization of interconnection networks,” in *International Symposium on High-Performance Computer Architecture (HPCA-9 2003)*, feb. 2003, pp. 91 – 102.

[3] C. E. Leiserson, “Fat-trees: universal networks for hardware-efficient supercomputing,” *IEEE Trans. Comput.*, vol. 34, no. 10, pp. 892–901, Oct. 1985. [Online]. Available: http://dl.acm.org/citation.cfm?id=4492.4495

[4] C. Bienia, “Benchmarking modern multiprocessors,” Ph.D. dissertation, Princeton University, January 2011.

[5] K. Kant, “Power control of high speed network interconnects in data centers,” in *Proceedings of the 28th IEEE international conference on Computer Communications Workshops*, ser. INFOCOM’09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 145–150. [Online]. Available: http://dl.acm.org/citation.cfm?id=1719850.1719875

[6] V. Soteriou and L.-S. Peh, “Design-space exploration of power-aware on/off interconnection networks,” in *IEEE International Conference on Computer Design: VLSI in Computers and Processors (ICCD 2004)*, oct. 2004, pp. 510 – 517.

[7] M. Alonso, S. Coll, J. Martinez, V. Santonja, P. Lopez, and J. Duato, “Dynamic power saving in fat-tree interconnection networks using on/off links,” in *International Parallel and Distributed Processing Symposium (IPDPS 2006)*, april 2006, p. 8 pp.

[8] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown, “Elastictree: saving energy in data center networks,” in *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, ser. NSDI’10. Berkeley, CA, USA: USENIX Association, 2010, pp. 17–17. [Online]. Available: http://dl.acm.org/citation.cfm?id=1855711.1855728

[9] A. G. Savva, T. Theocharides, and V. Soteriou, “Intelligent on/off dynamic link management for on-chip networks,” *JECE*, vol. 2012, pp. 6:6–6:6, Jan. 2012. [Online]. Available: http://dx.doi.org/10.1155/2012/107821

[10] J. Yin, P. Zhou, A. Holey, S. S. Sapatnekar, and A. Zhai, “Energy-efficient non-minimal path on-chip interconnection network for heterogeneous systems,” in *Proceedings of the 2012 ACM/IEEE international symposium on Low power electronics and design*, ser. ISLPED ’12. New York, NY, USA: ACM, 2012, pp. 57–62. [Online]. Available: http://doi.acm.org/10.1145/2333660.2333675

[11] J. Li, W. Huang, C. Lefurgy, L. Zhang, W. E. Denzel, R. R. Treumann, and K. Wang, “Power shifting in thrifty interconnection network,” in *Proceedings of the 2011 IEEE 17th International Symposium on High Performance Computer Architecture*, ser. HPCA ’11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 156–167. [Online]. Available: http://dl.acm.org/citation.cfm?id=2014698.2014904

[12] K. Kant, “Multi-state power management of communication links,” in *Communication Systems and Networks (COMSNETS), 2011 Third International Conference on*, 2011, pp. 1–10.

[13] D. Abts, M. R. Marty, P. M. Wells, P. Klausler, and H. Liu, “Energy proportional datacenter networks,” *SIGARCH Comput. Archit. News*, vol. 38, no. 3, pp. 338–347, June 2010. [Online]. Available: http://doi.acm.org/10.1145/1816038.1816004

[14] G. M. J. B. B. T. J. K. Nan Jiang, Daniel U. Becker and W. J. Dally, “A detailed and flexible cycle-accurate network-on-chip simulator,” in *Proceedings of the 2013 IEEE International Symposium on Performance Analysis of Systems and Software*, 2013.

[15] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003.

[16] J. Hestness, B. Grot, and S. W. Keckler, “Netrace: dependency-driven trace-based network-on-chip simulation,” in *Proceedings of the Third International Workshop on Network on Chip Architectures*, ser. NoCArc ’10. New York, NY, USA: ACM, 2010, pp. 31–36. [Online]. Available: http://doi.acm.org/10.1145/1921249.1921258