Evaluating Energy-Saving Strategies on Torus, K-Ary N-Tree, and Dragonfly

Felix Zahn
Heidelberg University
Institute of Computer Engineering
Mannheim, Germany
felix.zahn@ziti.uni-heidelberg.de

Armin Schäffer
Heidelberg University
Institute of Computer Engineering
Mannheim, Germany
a.schaeffer@stud.uni-heidelberg.de

Holger Fröning
Heidelberg University
Institute of Computer Engineering
Mannheim, Germany
holger.froening@ziti.uni-heidelberg.de

Abstract—**Energy is one of the most crucial factors in the design of large-scale computing systems, especially high-performance computing. While exascale systems could be built with current hardware solutions, the required funding exceeds the budget of most institutions. Since a system is never fully utilized, energy-proportional components can save a substantial amount of energy. However, current interconnect technologies still operate at a fixed power consumption rate. Therefore, network power consumption becomes increasingly important as its contribution to overall power consumption is increasing.**

Energy-proportional interconnection networks is a research area that is still emerging. In this work, we analyze the effects of different topology characteristics on power consumption and potential energy savings of interconnection networks. We compare the differences in the design of common topologies and the related impact to energy savings. In particular, we analyze the power consumption of torus, k-ary n-tree, and dragonfly. We also use existing topology-independent power-saving policies to derive potential energy savings for each topology and compare the policies to other work which is specific to topology hardware features. The comparison concludes that topology-independent policies are superior for energy savings and the other work is superior for execution time.

I. Introduction

The U.S. Department of Energy (DoE) aims to build exascale systems within a power budget of 20 megawatts [1]. In order to reach this ambitious goal, supercomputer must increase their energy efficiency at all levels. Using components with a fixed utilization to power ratio is one of the most promising approaches. While there is a good progress in energy proportional processors, most interconnection networks still operate at constant power. Previous analyses have shown that interconnects are in particular suited for this approach since they show a rather low average utilization [2]. Although the contribution of interconnection networks to the overall power consumption is rather small, multiple analyses show that their share will increase in the near future up to 30% [3] [4]. Additionally, the International Technology Roadmap for Semiconductors (ITRS) report predicts that data center power consumed for networking and switching will exceed power consumed by both storage and cooling by 2019 [5].

In contrast to the rather small share of the switch core logic, link ports and especially serializers consume the major share of the total power consumption of a switch. While the core logic is mainly designed using CMOS technology, Current Mode Logic (CML) is the underlying technology for serial links. Unlike CMOS, which dissipates power every time when switching, CML consumes power constantly during operation due to its current-driven character. Hence, frequency scaling has a small impact on power consumption, while reducing link width is more suitable for energy savings. In order to change link width, single parallel lanes inside a link can be switched on and off individually but may cause a certain downtime for the entire link in order to reconfigure.

In recent work, we have shown that even rather simple approaches enable energy-saving in direct non-hierarchical interconnection networks of up to 80% [6]. Although indirect and hierarchical networks differ from direct, flat networks in multiple factors, such as radix, routing, or link utilization, energy-saving in both type of networks follows the same fundamental approach. In this work, we analyze the impact of these factors on performance and energy-saving possibilities remains unclear.

In particular, we make the following contributions:

- How effective are our existing power-saving policies for other topologies, such as k-ary n-trees and dragonflys and how do they influence the execution time of different benchmarks?
- How do these power-saving policies compare to related work, which is tailored to certain topologies?
- How do different topologies compare in terms of link energy consumption?

The remainder of this paper is structured as follows: In Section 2 we give some short background information about serial links and power consumption of different topologies from an analytical view. In Section 3 we introduce our concept of energy-saving in interconnection networks. This is followed by the methodology of our experiments, including parameter sets for used traces, policies, and topologies. In Section 4 we provide the results and a short evaluation of our experiments. Section 5 concludes the evaluation part with a short discussion. Next, a brief overview of related work is given in Section 6. Finally, Section 7 finishes this work with a short conclusion.

II. Background

The network topology describes the physical layout of the various network elements such as switches and nodes.

The choice of the appropriate topology is an important task during the design process because many significant properties of the network are directly related to the chosen topology, such as bisection bandwidth and diameter. In this section, we are going to derive a simple analytical power model for topologies relevant for this study, which are 3D-Torus, k-ary n-tree and dragonfly, to showcase the difference in power-saving potential.

A. Serial links as common transmission technique

Serialization is a common practice in interconnection networks. This technique allows to convert wide internal data paths running at a given frequency $f_{parallel}$ into fewer, possibly only one, serial data path(s) running at higher frequency f_{serial}, which results in slimmer physical links. A serial link usually has no dedicated clock signal, instead, the clock is embedded in the data signals. As a result, a serial link offers the possibility to transmit data using fewer physical data paths, in the following called link lanes, thereby reducing the number of pins. However, serial links require a link training to ensure that the embedded clock is locked on the receiving side and that both transmitter and receiver agree on a common view of word boundaries. In terms of power saving, several aspects of a serial link are relevant:

1) A link training is required any time the frequency of a lane is changed, resulting in a time period in which the link is on but cannot transmit data. We will refer to this as transition time. Note that turning a lane off does not result in a transition time.
2) Even in the case of no traffic, a serial link has to continue exhibiting signaling transitions to maintain clock locking and word alignment.
3) Serializers and de-serializers are the dominant factors regarding overall power consumption of a network switch, as shown in previous work [2].

B. Analytical power model based on topology characteristics

A first approximation of a power model, independent of the chosen topology, identifies the port count and thereby the number of (de-)serializers used. Furthermore, we assume the contribution of switch core logic, including among others routing logic, crossbar or similar switches and arbiters, to be constant. Although the power consumption of the switch core logic correlates with the number of ports, we believe these variations to be neglectable since overall power consumption is dominated by the link port power. Therefore, a first approximation of the total required power for a network is

$$P_{network} = L \cdot P_{port} \tag{1}$$

where L denotes the port count in the network. Calculating L proves itself to be not straight-forward since there is no universal rule that is valid for all topologies. However, it is feasible to derive topology-dependent formulas as shown in the following, as shown in Table I with N denoting network size and R the switch radix.

Topology	Port count L
3D-Torus	$6N$
k-ary n-tree	$N \cdot \left(\frac{2 \cdot \log(N)}{\log(\frac{R}{2})} + 1\right)$
Dragonfly	$N \cdot \left(\frac{4R}{R+1} + 1\right)$

TABLE I: Port count for different topologies, with network size N and switch radix R.

In order to visualize the scaling of the port count with network size, Table II shows the corresponding port count for selected network configurations. As we assume the overall power to be dominated by the port count, this data also creates an initial assumption on the overall power-saving potential for a given topology. In particular, we observe that the dragonfly topology has a low absolute port count, but also shows nice scalability when increasing network size. Contrary, the k-ary n-tree indicates both high port counts and a poor scalability. However, note that actual power consumption is also highly dependent on network latency, link utilization, and link idle times, requiring more detailed studies than this simple analytical model.

# Nodes	# Ports			
	3D Torus	k-ary n-tree		Dragonfly
72	432	564	(radix 7)	324
1,056	6,336	8,354	(radix 15)	5,016
16,512	99,072	133,529	(radix 31)	80,496

TABLE II: Port count L for different topologies and networks sizes.

III. Concept

For most components, a common approach to save energy and power is to adapt their power consumption to their utilization by switching between discrete power states. These power states trade off performance for reduced power consumption, since the component is not fully utilized. Usually, frequency scaling and switching on and off single components or modules are the most applicable techniques to design discrete power states.

A. Energy-Proportional Networks

Energy saving in interconnection networks is based on similar techniques, however, with some subtle differences. In recent work, we have shown that actual exascale benchmarks and other scientific application utilize links in a 3D torus network only up to 6% for Graph500 and NAMD [2].

Unlike many components, such as processors, that are mostly based on CMOS logic, frequency scaling is not applicable for interconnection networks. Our analysis shows that serial off-chip links contribute about 70% to the power consumption of a NIC, while the core logic only contributes little to the overall power consumption (1̃5%). In contrast to the CMOS-based core logic, serialization technology, and thereby the serial links is dominated by the CML standard. The way these two technologies consume power is fundamentally different. In CMOS technology, power is dissipated every time when switching, which results in a highly frequency depending power consumption. Contrary to CMOS, CML is a current-driven technology, which consumes power constantly while operating. Therefore, CML power consumption is independent of the operating frequency. This suggests scaling link width to be the most promising approach for power and energy saving in interconnection networks.

In most of today's interconnection technologies, serial links already consist of multiple lanes, such as Ethernet, Infiniband, Omnipath, EXTOLL, or PCIe. Switching these single lanes inside a link independently on and off enables adapting bandwidth to the current utilization. However, every time a lane is switched on or a link changes its operating frequency, the link has to perform a re-training. Also, power saving in interconnection networks is also a trade-off between power and performance. Therefore, it must be decided what performance decrease is tolerable in order to improve energy efficiency.

B. Policies

In recent work, we introduced several policies to switch between power states and, thereby, save energy in direct interconnects [6]. These policies do not take any specific feature of direct networks into account; hence, they can also be used for other topologies without any additional adaption. In order to analyze the impact of topology specific features, we compare our strategies with one introduced by Alonso et al. [7], which is designed for k-ary n-tree networks. In the following, there is a short overview of the used policies.

On/Off

This first policy switches links off if they are idling and back on if data has to be transmitted on the link. In order to decide whether a link should be on or off, there is a timer in every linkport, which is reset every time data is transmitted on this link. If the timer finishes without new data arriving, the link is switched off and remains in this state until a new packet or credit arrives in the output buffer. Once a link is switched on again, it is unable to transfer any data, until it has performed the re-training (transition time). Since an off-link cannot transmit any data, both directions in a bidirectional link are switched off simultaneously, because the credit based flow control requires a handshake.

The time a link needs to remain inactive before it is switched off depends on the two parameters: transition time and maximum relative performance loss. This approach was inspired by the work of A. Venkatesh et al [8] and allows to cap the maximum performance loss that is tolerated while saving energy. Since this algorithm is designed originally for the MPI layer, this maximum performance loss is just from a NIC view and can sum up along paths in the network. Therefore, we decided to focus on two different inactivity periods, after which a link is switched off: A rather conservative approach, which should cause a small performance loss but also less energy savings and a more aggressive approach, which accept potential higher performance loss in order to save more energy. The different parameter and their impact are explained more detailed in [6].

High/Low

In order to improve performance, we introduced a second policy, which switches links into a lower power state instead of switching them completely off. In this lower power state, the link consumes less power but is still able to transmit data with slower speed. In particular, only one of twelve parallel lanes is active in this power state. Analogous to bandwidth, power consumption decreases by a factor of twelve.

Single packets or credits can be instantly transmitted without switching to a higher power state. If the associated link buffer is fill up to a certain level, it is better to switch the link back into a high power state, even though during the transition time no packet forwarding is possible. The corresponding threshold of the buffer's fill level is described in equation 2:

$$\frac{data}{BW_{low}} > \frac{data}{BW_{high}} + t_t => data > \frac{t_t}{\frac{1}{BW_{low}} - \frac{1}{BW_{high}}} \quad (2)$$

While this policy aims to decrease the performance loss for power saving, it thereby reduces the maximum amount of power that can be saved, since links are not completely switched off but put in a state which still allows to transfer data at a lower speed.

Alonso

Alonso et al. introduced a power-saving policy particular suited for k-ary n-trees [7], [9]. This policy switches links on and off depending on the current router utilization. The corresponding dynamic threshold is periodically adjusted, depending on the number of active outgoing links. In order to maintain connectivity in the network, a subset of links and switches is defined, which cannot be powered down. This minimal tree defines an upper bound for the maximal power savings. K-ary n-tree specific properties are taken into account by adapting the conditions for powering links on or off depending on the link direction (up or down) and the level of the corresponding switch in the network. Analogously to our policies, we used a conservative and an aggressive threshold. A detailed overview of this approach is provided in [7], [9].

Note that we adjusted this policy by switching links into the lowest power state, instead of switching them completely off in order to adapt it to our simulator. Bidirectional links are mandatory for the handshake of our credit-based flow control protocol. However, links in this low power state are only used by the flow control and cannot transmit packets. Additionally, we adjusted the DESTRO routing algorithm for k-ary n-trees, because this policy requires adaptive routing. If the routing algorithm selects a switched-off port, the packet is routed to another random valid port.

IV. Methodology

Since implementing new experimental energy-saving features in hardware is not feasible, we use a network simulator for our experiments. The following section provides a short overview of this simulator and the parameters we used for our experiments.

A. Traces

Energy-saving techniques in interconnection network exploit gaps between communication phases to reduce bandwidth in under-utilized links. However, these techniques fail with generated traffic due to its randomness. Therefore, we decided to use traces in the VEF [10] format. In this format, all MPI events are recorded in the traces and the periods in between are assumed as computation time. For our simulations, we selected the following real scientific applications, executed by 512 MPI ranks:

LULESH (Livermore Unstructured Explicit Shock Hydrodynamics): is one of the proxy exascale applications provided by the United States Department of Energy (DOE). It is a hydrodynamic simulation, which uses a stencil code to calculate the physical forces. The traces are generated with a problem size of 100 and 50 iterations. This results in about one million elements per rank.

NAMD (Nanoscale Molecular Dynamics program): performs a simulation of dynamic biomolecular systems with underlying n-body particle calculations. We select the Satellite Tobacco Mosaic Virus (STMV) molecule as input data, which is consists of about one million atoms and is frequently used.

Graph500: is a benchmark with a data-driven communication pattern. This breadth-first search (BFS) graph traversal is also the Graph500 list's benchmark, which is an alternative to the TOP500 list. The traces are generated with replicated-CSR implementation, a scale factor of 20, and an edge factor of 16.

WRF (Weather Research and Forecasting): uses numerical simulations for operational weather forecasting and climate research. This benchmark is used for comparing various aspects, including CPUs, Interconnects, and MPI library performance.

HPL (High-Performance Linpack): (High-Performance Linpack) is widely used as a benchmark for the Top500 list. It solves a dense N N system of linear equations. The traces are generated with the following parameters: row-mapping, N = 9984, P = 16, Q = 32, threshold = 16, NBs = 192.

B. Network Simulator

As a simulation environment, we extended the OMNeT++-based SAURON simulator [11] with multiple power measurement features and implemented the introduced energy-saving policies [2]. This simulator models a detailed NIC with all components of a commodity NIC that can be customized and adjusted by the user.

C. Simulation Parameters

The used network simulator provides multiple parameter sets that can be set up by the user. For parameters that have a minor relevance for energy saving, we configure a common HPC network. All parameters concerning the focus of this work are provided here:

3D Torus: We use an 8x8x8 configuration with 512 nodes in total and an X-Y-Z dimension order routing.

k-ary n-tree: We examine two different configurations for k-ary n-tree to gain insights about the effects of hierarchies in networks regarding power savings: three stages with a switch radix of 16 resulting in 512 nodes and two stages with a switch radix of 64 resulting in 1024 nodes. For both configurations, DESTRO Routing [12] is used.

Dragonfly: The parameters of the dragonfly are closely orientated at [13]. We use a switch radix of 28 ports and a fully interconnection, which results in 756 nodes. We select a dragonfly specific minimal routing [13].

V. Results

With our experiments, we aim to gain insights about the efficiency of our prior introduced energy-saving policies, and how features of different topologies affect execution time (also referred to as performance), and energy consumption.

For all policies we choose a conservative and an aggressive setting, which results in a power-down timer of $t_{down} = 1ms$ for the conservative setting and $t_{down} = 0.1ms$ for the aggressive setting. The impact of these parameters is discussed in more detail in [14]. Additionally, we use similar settings for Alonso's policy with a conservative utilization threshold for powering a link on of 0.15 and an aggressive one of 0.9. Last, we set the transition time to $100\mu s$, which we believe to be representative for today's hardware.

Note that we take only link power into account since we are not aware of a reliable switch core power model. We leave such analysis to our future work.

A. 3D Torus

The first topology we analyze is the 3D Torus. Tori are flat, direct networks, which are widely used due to their good scaling behavior. Figure 1 depicts the relative link energy consumption of our policies, normalized to the energy consumption of the same network, running the same workload without any energy-saving policy.

All evaluated policies substantially reduce link energy consumption, up to as low as about 1% of the original energy consumption. In particular, we observe that the on/off policy with an aggressive setting performs better than the high/low policy for all workloads. For the conservative setting, high/low provides better results for NAMD and HPL. This is caused by a lot of small messages and idling periods, which prevent the on/off policy to switch off links. Except for one configuration, all combinations enable link energy savings of more than 90%.

In Figure 2, the corresponding relative performance loss normalized to the execution time without energy savings is shown. The high/low policy performs better for all workloads and keeps the performance loss within one percent for LULESH, NAMD and WRF. This is not surprising since this policy was designed to trade-off some energy saving for a better and more reliable performance. Also as expected, the conservative

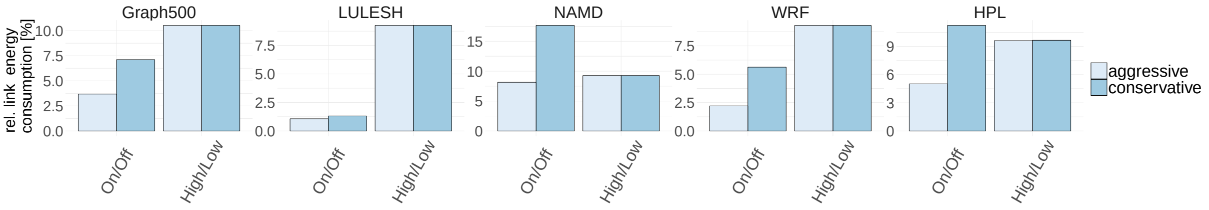

Fig. 1: 3D Torus: relative link energy consumption, normalized to energy consumption without power savings

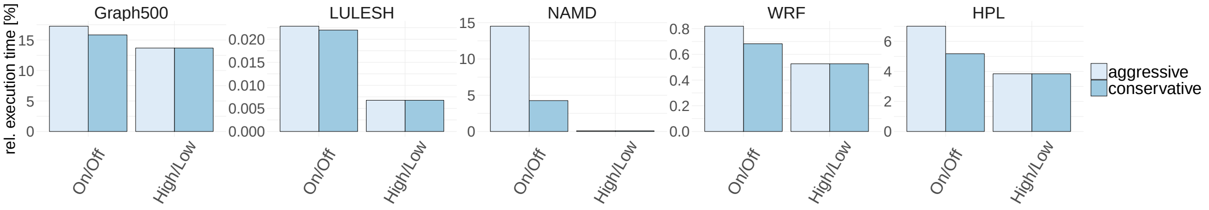

Fig. 2: 3D Torus: relative execution time increase, normalized to execution time without power savings

setting for on/off shows better performance than the aggressive switching off one. Furthermore, it is notable that the high/low policy does not show any significant differences between the aggressive and conservative setting in both energy savings and performance.

B. K-Ary N-Tree

This widely used topology received more attention in past regarding power and energy saving. We implemented another policy introduced by Alonso et al. [9] in order to compare our policies to one tailored to k-ary n-trees.

Since there are different possibilities to build this topology, we choose two different configurations, two and three stages:

Two Stages: The two stages are built with a switch radix of 64, resulting in 1024 nodes. In order to ensure a fair comparison, we subtract the power of all additional links that are not utilized due to the mapping of the traces. The energy-saving potential and the according performance loss are shown in Figure 3 and Figure 4, respectively. Again, the energy consumption and runtime are normalized to the workloads executed on this topology without any power saving.

Concerning energy saving, almost all combination follow the same trend as when applied to the torus topology. Again, on/off enables more energy saving, especially the aggressive setting. The two different settings result in just slightly differences for the high/low policy. Only even the conservative setting for HPL in combination with the on/off policy provides better results than the high/low policy.

Alonso's policy, despite it was developed for this particular topology, results in a substantially higher energy consumption, yielding results in the range of 35-50% of the original link energy. For reference, the other policies reduce link energy down to 10% or less. These obtained energy savings are very close to an upper bound, which is determined by multiple links that are used to maintain full connectivity. Also, Alonso's policy shows an impressive performance in terms of execution time. Except for the Graph500 workload, which causes a bad performance for all policies, all benchmarks are basically seeing no increase in execution time

The performance of the two other policies is surprising, too. For Graph500, WRF, and HPL, high/low shows worse performance than on/off. In particular, this is remarkable since high/low was designed to trade-off some energy savings for an improved performance.

Three stages: Compared to an equally-sized 3D Torus, the k-ary n-tree with three stages requires more links. Since the traffic in the network remains the same, the average utilization per link decreases. However, the effect of this lower utilization on the energy savings are rather small. Figure 5 and Figure 6 depict the results for the three-staged k-ary n-tree.

There are just small differences in the energy-saving results for all topologies between the two-staged and three-staged version. Only for NAMD, the on/off policy provides better results for two stages than for three stages. Again, Alonso's policy reduces the link energy consumption to about 40% for all workloads.

The number of stages of the k-ary n-tree affects the execution time. While the performance of all workloads with the on/off policy decreases for three stages, the performance of the high/low policy slightly improves.

C. Dragonfly

The dragonfly topology is an example of a hierarchical direct network and is becoming more popular due to its low network diameter and high bisection bandwidth. The design rules for dragonfly do not allow for a size of 512 nodes, therefore, we used the closest possible size (756 nodes). Analogous to the two-stage k-ary n-tree, we remove the power consumed by the remaining links that connect to unused nodes. Still, the number of links used for dragonfly is slightly higher than for a 3D torus. The results for this topology are depicted in Figure 7 and Figure 8.

Surprisingly, the results of this topology follow the same trends as for the 3D torus, but dragonfly enables in average

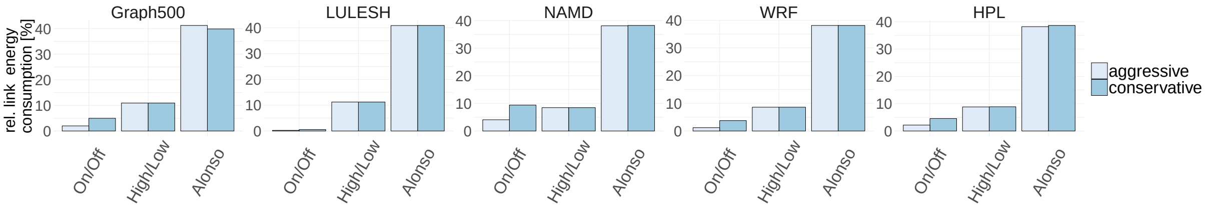

Fig. 3: k-ary n-tree (two stages): relative link energy consumption normalized to energy consumption without power savings

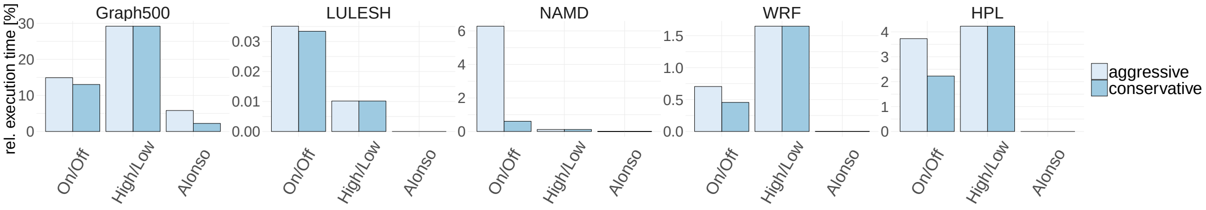

Fig. 4: k-ary n-tree (two stages): relative execution time increase, normalized to execution time without power savingse

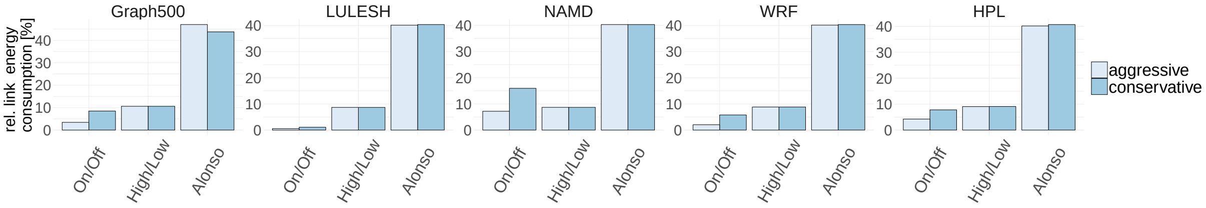

Fig. 5: k-ary n-tree (three stages): relative link energy consumption normalized to energy consumption without power savings

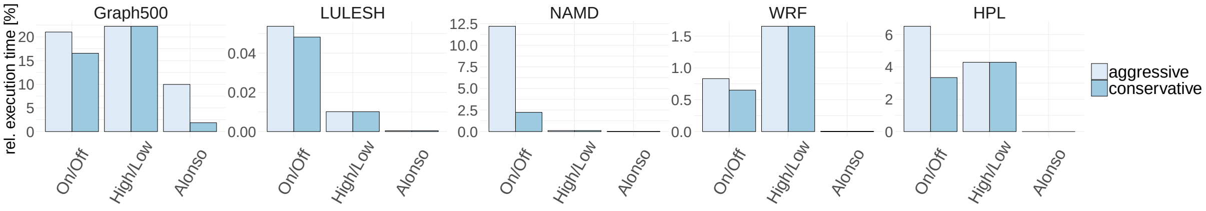

Fig. 6: k-ary n-tree (three stages): relative execution time increase, normalized to execution time without power savings

more power saving for the on/off policy, which correlates with the lower average utilization per link. Concerning execution time, dragonfly again shows effects similar to a 3D torus, except for the Graph500 workload, which performs better using the on/off policy than high/low.

Generally, both policies provide good results with link energy savings of more than 90% in most cases, within a reasonable performance loss. Only the Graph500 workload causes larger performance losses, which might not be tolerable in HPC settings. This is likely caused by the unpredictable character and the small message-based communication pattern of this workload.

VI. Discussion

In a 3D torus network, both policies show promising results. While on/off mostly enables more link power saving, high/low performs better in terms of execution time. Only for NAMD and HPL high/low consumes less link energy than on/off in a conservative setting. This is due to the workload's communication pattern: the idling periods between communication phases are too short for the conservative setting to switch links off, therefore, they remain in the fast but power-hungry high power state. However, once a link is in the low power state, the communication volume density is too small to trigger the high/low policy to switch back into a higher and faster power state. Comparing the aggressive to the conservative setting, the on/off policy shows results as expected: while an aggressive setting increases the energy savings, it also increases the execution time since links are switched more often, causing performance penalties in form of transition time. Surprisingly, there are almost no differences between these two settings for high/low, regarding both energy consumption and execution time. This indicates that most links remain in the lower power state, once they were switched to it. This corresponds to the overall number of transitions between power state for these workloads and policy.

For k-ary n-tree topologies there exist multiple design possibilities regarding switch radix and number of stages.

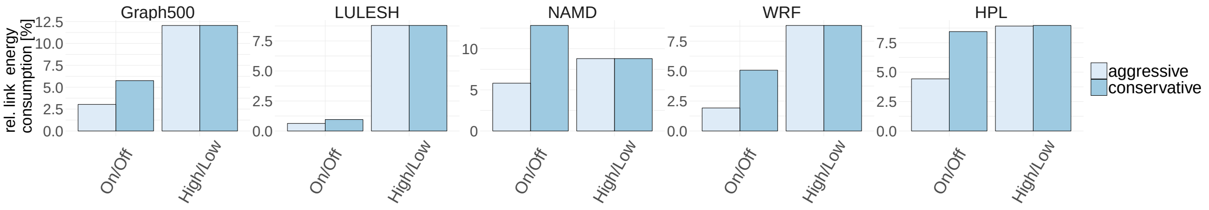

Fig. 7: Dragonfly: relative link energy consumption normalized to energy consumption without power savings

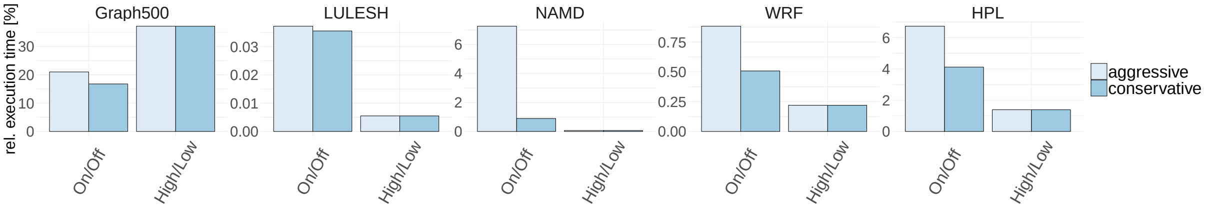

Fig. 8: Dragonfly: relative execution time increase, normalized to execution time without power savings

Both investigated configurations with two and three stages show no significant differences regarding power saving or execution time. Minor better results for single configurations in the two-stage version indicate slight advantages regarding energy saving. However, our results only focus on link power consumption. Since fewer stages are required for higher radix switches, the rising core power consumption of these switches could exceed these benefits. Regarding aggressive and conservative settings, this topology behaves similarly to a 3D torus. In contrast to the original purpose and the results of the 3D torus, the high/low policy performs worse in terms of execution time compared to the on/off policy for three out of five workloads, likely caused by the selected routing algorithm. Since the DESTRO algorithm is designed to balance the load among all up links, single links are not utilized enough to switch back to the higher and faster power state, once they are in the low power state. This has particularly an impact for patterns that continuously send messages instead of bursts in a global communication phase. Compared to the policy of Alonso et al., our policies enable significant higher energy savings: while Alonso's policy achieves about 60% of energy consumption, high/low saves on average about 90% and on/off even more. However, it is impressive that Alonso's policy causes almost no performance loss, except for the Graph500 workload. We believe that small increases of execution time could possibly be hidden by an overlap of computation and communication, but larger delays will certainly impact energy consumption.

For the Dragonfly topology, both policies follow the same trends as for the 3D torus for most configurations, but Dragonfly shows even more improvements in terms of energy saving and execution time.Only for the Graph500 workload the 3D torus shows better results, possibly based on the all-to-all pattern. Except for this workload, high/low performs better than on/off in terms of execution time. This fact and the similar amount of energy saving suggest that high/low is a better fit for this topology.

VII. Related work

Various work on on-chip interconnection networks exists (an excellent summary can be found in [15]), but these designs adhere to completely different technical constraints such as signaling, physical distance, and topology. In the following, we concentrate on scalable (i.e., inter-node) interconnection networks.

The growing significance for more energy-proportional networks is shown in [3]. While the authors analyze power consumption analytically and show potential parameters for an improved energy-proportionality, they do not provide solutions.

An energy-efficient MPI runtime is introduced in [8]. Contrary, we rather optimize power consumption at link level and independent of a particular programming language, but we adapted the algorithm introduced in this work to calculate the threshold for switching network links into a low-power state.

The performance of Energy-Efficient Ethernet (EEE) is evaluated in [4], analyzing the behavior of EEE for HPC workloads regarding power consumption, bandwidth and latency. The authors also propose a concept in which power-downed links are regularly woken up to ensure continuous clock locking and word alignment. The overhead of this technique is analyzed in [16] and an extension is proposed for a deep-sleep state. In contrast to our work, this technique mainly relies on specific EEE features.

Hendry [17] proposes Asynchronous Circuit Programming (ACP), which allows the programmer to closely interact with the underlying hardware through a high-level interface to establish communication channels and additional features for HPC applications. Such an approach can guarantee near-optimal state selection as explicit knowledge about application behavior is present, however, the responsibility is shifted from architecture to the programmer with substantial implications on programming complexity.

A promising concept for obtaining deeper insights on the power behavior of interconnection networks is presented by

Wang et al. [18], [19], who developed an activity-based power model for fundamental switch components. As we only analyze link power consumption, a combination could yield comprehensive results on overall network power consumption.

VIII. Conclusion

While most components of HPC installations steadily achieve better energy proportionality, interconnection networks remain a constant factor in the system's overall power consumption. Therefore, their relative share of overall power consumption increases continuously. In order to achieve more energy-efficient interconnects, we have evaluated energy-saving policies applied to different topologies.

In general, these policies show good results, not only regarding energy saving but execution time. While the on/off strategy with an aggressive power-down policy is best with savings of more than 95% of the link energy for almost every configuration, it also causes the highest performance loss. Contrary, for 3D tori and Dragonflies, the high/low policy seems to be a good compromise as it still enables energy savings of more than 90% but results in less execution time increase. Seconding our results from an analytical model, Dragonfly shows the highest potential for link energy savings. This hierarchical topology has received rather little attention yet regarding energy saving, but our results suggest that it is a promising candidate for further studies. Another notable insight is the performance of Alonso's policy. Although it cannot keep up with the two other policies regarding energy savings, it outperforms both in terms of execution time.

Next, we plan to develop a switch core power model for a complete study of energy consumption in scalable interconnection networks, for a fairer and more precise comparison of different topologies. Furthermore, especially for indirect topologies adaptive routing seems promising due to a huge amount of redundancy in network paths. The results from this work also suggest that energy saving policies should take communication characteristics into account [20]. Last, investigate networks at a larger scale it is important to address simulation time, possibly by traffic models to generate synthetic but realistic traffic patterns.

Acknowledgments

This work is supported by the Carl-Zeiss-Foundation in the form of a scholarship. Furthermore, we thank Pedro Yebenes for his guidance and support regarding the network simulator, and Maximilian Thürmer for insightful discussions on switch power consumption.

References

[1] "The opportunities and challenges of exascale computing," 2010. http://science.energy.gov/~/media/ascr/ascac/pdf/reports/exascale_subcommittee_report.pdf.

[2] F. Zahn, P. Yebenes, S. Lammel, P. J. Garcia, and H. Fröning, "Analyzing the energy (dis-) proportionality of scalable interconnection networks," in *2nd IEEE International Workshop on High-Performance Interconnection Networks in the Exascale and Big-Data Era (HiPINEB)*, pp. 25–32, March 2016.

[3] D. Abts, M. Marty, P. Wells, P. Klausler, and H. Liu, "Energy proportional datacenter networks," in *International Symposium on Computer Architecture*, pp. 338–347, 2010.

[4] K. P. Saravanan, P. M. Carpenter, and A. Ramirez, "Power/performance evaluation of energy efficient ethernet (eee) for high performance computing," in *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pp. 205–214, April 2013.

[5] "The international technology roadmap for semiconductors 2.0 - executive report," 2015. http://www.semiconductors.org/clientuploads/Research_Technology/ITRS/2015/0_2015%20ITRS%202.0%20Executive%20Report%20(1).pdf.

[6] F. Zahn, S. Lammel, and H. Fröning, "On link width scaling for energy-proportional direct interconnection networks," *To appear in Concurrency and Computation: Practice and Experience*, 2018.

[7] M. Alonso, S. Coll, V. Santonja, J. Martnez, P. Lopez, and J. Duato, "Power-aware fat-tree networks using on/off links," vol. 4782, pp. 472–483, 09 2007.

[8] A. Venkatesh, A. Vishnu, K. Hamidouche, N. Tallent, D. Panda, D. Kerbyson, and A. Hoisie, "A case for application-oblivious energy-efficient mpi runtime," in *SC15: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–12, Nov 2015.

[9] M. Alonso, S. Coll, J. M. Martinez, V. Santonja, P. Lopez, and J. Duato, "Dynamic power saving in fat-tree interconnection networks using on/off links," in *20th IEEE International Parallel Distributed Processing Symposium*, pp. 8 pp.–, April 2006.

[10] F. J. Andújar, J. A. Villar, J. L. Sánchez, F. J. Alfaro, and J. Escudero-Sahuquillo, "VEF Traces: A Framework for Modelling MPI Traffic in Interconnection Network Simulators," in *IEEE International Conference on Cluster Computing (CLUSTER)*, pp. 841–848, Sep 2015.

[11] P. Yebenes, J. Escudero-Sahuquillo, P. J. Garcia, and F. J. Quiles, "Towards modeling interconnection networks of exascale systems with omnet++," in *21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pp. 203–207, Feb 2013.

[12] C. G. Requene, *Low-Memory Techniques for Routing and Fault-Tolerance on the Fat-Tree Topology*. PhD thesis, Universidad Politécnica de Valencia, 2010. https://riunet.upv.es/bitstream/handle/10251/8856/tesisUPV3368.pdf.

[13] J. Kim, W. J. Dally, S. Scott, and D. Abts, "Technology-driven, highly-scalable dragonfly topology," in *International Symposium on Computer Architecture*, pp. 77–88, June 2008.

[14] F. Zahn, S. Lammel, and H. Fröning, "Early experiences with saving energy in direct interconnection networks," in *3rd IEEE International Workshop on High-Performance Interconnection Networks in the Exascale and Big-Data Era (HiPINEB)*, pp. 33–40, 2017.

[15] V. Soteriou and L. S. Peh, "Exploring the design space of self-regulating power-aware on/off interconnection networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, pp. 393–408, March 2007.

[16] P. Reviriego, J. A. Hernandez, D. Larrabeiti, and J. A. Maestro, "Performance evaluation of energy efficient ethernet," *IEEE Communications Letters*, vol. 13, pp. 697–699, Sept 2009.

[17] G. Hendry, "Decreasing network power with on-off links informed by scientific applications," in *IEEE International Symposium on Parallel Distributed Processing, Workshops and Phd Forum*, pp. 868–875, May 2013.

[18] H.-S. Wang, X. Zhu, L.-S. Peh, and S. Malik, "Orion: a power-performance simulator for interconnection networks," in *35th Annual IEEE/ACM International Symposium on Microarchitecture, 2002. (MICRO-35). Proceedings.*, pp. 294–305, 2002.

[19] H.-S. Wang, L.-S. Peh, and S. Malik, "A power model for routers: modeling alpha 21364 and infiniband routers," *IEEE Micro*, vol. 23, pp. 26–35, Jan 2003.

[20] B. Klenk and H. Fröning, "An overview of mpi characteristics of exascale proxy applications," in *High Performance Computing* (J. M. Kunkel, R. Yokota, P. Balaji, and D. Keyes, eds.), (Cham), pp. 217–236, Springer International Publishing, 2017.