High-Bandwidth Low-Latency Approximate Interconnection Networks

ABSTRACT

Computational applications are subject to various kinds of numerical errors, ranging from deterministic round-off errors to soft errors caused by non-deterministic bit flips, which do not lead to application failure but corrupt application results. Non-deterministic bit flips are typically mitigated in hardware using various error correcting codes (ECC). But in practice, due to performance and cost concerns, these techniques do not guarantee error-free execution. On large-scale computing platforms, soft errors occur with non-negligible probability in RAM and on the CPU, and it has become clear that applications must tolerate them. For some applications, this tolerance is intrinsic as result quality can remain acceptable even in the presence of soft errors (e.g., data analysis applications, multimedia applications). Tolerance can also be built into the application, resolving data corruptions in software during application execution. By contrast, today's optical networks hold on to a rigid error-free standard, which imposes limits on network performance scalability. In this work we propose high-bandwidth, low-latency approximate networks with the following three features: (1) Optical links that exploit multi-level quadrature amplitude modulation (QAM) for achieving high bandwidth; (2) Avoidance of forward error correction (FEC), which makes optical link error-prone but affords lower latency; and (3) The use of symbol mapping coding between bit sequence and QAM to ensure data integrity that is sufficient for practical soft-error-tolerant applications. Discrete-event simulation results for application benchmarks show that approx networks achieve speedups up to 2.94 when compared to conventional networks.

1. INTRODUCTION

Computational applications are subject to various kinds of numerical errors. Some of these errors are well-known and deterministic, due to the fact that computers encode approximate numerical values to represent application data (e.g., decimal number 0.1 is approximated as 0.00011001100_{binary}). Computers are also subject to non-deterministic errors [1], which can be caused by external factors, including cosmic radiations or packaging pollution, or due to the use of DVFS to reduce energy consumption [2]. Because error correcting code (ECC) techniques are not full proof (due to cost and performance concerns), the expectation is that current computers do not compute in an error-free manner as they have unreliable memory and processors. When computing on large-scale platforms that aggregate tens to hundreds of thousands of compute nodes, the probability that an application execution is subject to non-deterministic numerical errors that can corrupt application results (without causing an actual failure) is no longer negligible. There is thus a large literature on how to cope with these errors. One approach, termed "approximate computing," consists in designing applications that can tolerate soft errors and for which there are known trade-offs between error rates and result quality [3]. Another approach is to recover from soft errors at runtime, e.g., using redundant computation [4], domain-specific algorithm-based fault tolerance (ABFT) [5–8], or domain-specific result verification and re-execution [9, 10],

By contrast with computational resources, communication standards, such as InfiniBand and Ethernet, still guarantee virtually error-free data transfer with extremely low bit-error rate (BER), such as 10^{-12} for Ethernet and InfiniBand. Given the high-bandwidth and low-latency goals of next-generation interconnects, the performance cost of ensuring these extremely low BER will become significant. For instance, due to the pervasive use of forward error correction (FEC) (instead of, say, cyclic redundancy check (CRC) with retransmission on error), it is expected that switch delay will become relatively higher if current BERs are to be maintained. It is reported that 25 Gbps high-reliability high-end optical interconnects require at least 84 ns for storing 2,112 bit encoding blocks used in Base-R FEC computation [11]. Although the Base-R FEC has lower overhead than some alternatives (e.g., Reed-Solomon coding), one can still expect about 100ns FEC overhead per packet at every switch. Since current switch delay ranges from 40ns [12] to 100ns in InfiniBand QDR, the per-hop delay will increase up to 3-fold due to FEC computation. The FEC processing delay becomes the same (100ns) even for 100Gbps (25Gbps×4) optical links (e.g., 100GBASE-SR4). Although some 100Gbps cables, e.g., 100GBASE-LR4 and those on BlackWidow [13], successfully avoid FEC, most of over-100Gbps cables will have to use FEC so as to meet current low BER requirements.

In this work we propose approximate-computing interconnection networks, which we call "approx networks." Approx networks have higher BER than conventional interconnects, meaning that they require applications to tolerate soft errors (bit flips) that can occur in transferred data. Our rationale is that in many applications such tolerance is already expected for computation, as discussed earlier. Consequently, providing link BER on par with soft error rates expected on compute resources should be sufficient. In other words, provided overly conservative error correction for communication is not worthwhile for applications that already tolerates soft errors for computation. Due to the lowered BER requirement, approx networks can achieve high bandwidth and low latency as follows. At the link layer, we use multi-level quadrature amplitude modulation (QAM) for high-bandwidth purpose, but we avoid FEC so as to lower communication latency (but also reliability). In the network layer, we use a low-hop network topology so that each packet traverses a small number of (error-prone) optical links; thus reducing the probability of experiencing bit flips. In the application layer, we provide APIs that manipulate symbol-map coding between bit sequence and signal (i.e., QAM), to reduce the numerical error range (which benefits approximate-computing applications). More specifically, our main contributions in this work are:

- We propose a network design, approx networks, that achieves higher bandwidth than current network interconnects and low latency for applications that can tolerate soft errors. This approach relies on the codesign of different network layers. (Section 3)
- We propose a symbol mapping technique that makes it possible to trade off higher bandwidth for lower BER, so as to meet the requirements of various applications. For instance, assuming that bandwidth B can be achieved with a nominal BER of 10^{-5} without special symbol mapping, a reliable symbol mapping scheme achieves bandwidth $B/4$ with a lower BER of 10^{-16} (which is almost the same quality as the BER of current conventional networks). (Section 4)
- Discrete-event simulation results for approximate-computing and ABFT applications on approx networks show speedups up to 2.94 when compared to conventional networks, showing that approx network can achieve good performance/reliability tradeoffs in practice. (Section 5)

Background information and related work are discussed in Section 2. Section 6 discuss relevant issues and limitations of approx networks. Section 7 concludes with a summary of our findings and perspectives on future work.

2. BACKGROUND AND RELATED WORK

2.1 Baseline Interconnection Networks

Conventional packet networks used in datacenters and in the HPC domain consist of electric switches and copper cables. In recent years, active optical cables (AOC) have been adopted for higher link rates. The bandwidth of cheap commodity AOCs is now 40 or 56 Gbps, and is expected to reach 100 Gbps (25 Gbps $\times$ 4). As bandwidth increases, so do reliability concerns. Consequently, it is expected that high-bandwidth links will use FEC to ensure reliability. For instance, 100GBASE-SR4/CR4 (25 Gbps $\times$ 4) uses Reed-Solomon FEC. We note that 100GBASE-LR/ER4 (25 Gbps $\times$ 4) does not, but 400GBASE-FR8/LR8, which is currently being standardized, is expected to use Reed-Solomon FEC. Current electric switches provide reliable transfer via non-blocking crossbars that form bases for lossless interconnection networks. Electric switch chips have enjoyed a 100-fold aggregate bandwidth improvement over the last 10 years [14], and recent products, such as IBM PERCS, reach 10Tbps bandwidth per router node. By contrast, commodity cable bandwidth has only moderately increased. These trends are expected to continue.

A range of network topologies of switches, including tori, fat trees, and Dragonfly [15], are used to interconnect compute nodes in datacenters and HPC systems. Since electric switching chips have high aggregate bandwidth, these topologies can be used to interconnect high-radix switches. In addition to these traditional topologies, random topologies with arbitrary degree have been proposed that achieve low diameter, low average shortest path length, and thus competitively low end-to-end network latencies when compared to traditional topologies for given degree and network size specifications [16].

In this work we consider networks of electric switches interconnect via AOCs as our baselines, in accordance with current implementation of high-end datacenters and HPC systems.

2.2 Optical Switches in Interconnection Networks

In comparison to popular interconnection networks based on electric switches, optical switching devices may provide an attractive alternative for future HPC networks.

Installing optical circuit switches between top of racks (ToRs) has been proposed to provide reconfigurable interconnection networks. Adapting network topologies to traffic patterns have been demonstrated to lead to 2$\times$ performance improvements [17]. The reconfiguration time of such networks was measured as 1.2 to 2.2 seconds, including reconfiguring optical switches and updating forwarding rules [18]. Introducing wavelength division multiplexing (WDM) transmission techniques along with wavelength switching capability has also been considered to increase link capacity and flexibility in the optical domain. Previous work has pointed to possibly significant reductions of transceivers, cables, and power consumption of ToRs [19].

All-optical intra-datacenter network interconnects have also been considered. Spectral selective switches (SSS) were installed as ToRs and top-of-cluster switches, and, depending on distance, optical link latencies between 29

ns and 648 ns were achieved [20]. In [21], a hierarchical arrayed waveguide grating router (AWGR)-based direct optical interconnect architecture was proposed that can accommodate more than 1,000,000 compute nodes with a diameter of 7. Optical packet switching technologies have also been studied [22, 23]. Although the packet switching mechanism can provide high compatibility with existing electrical packet-based interconnect networks, significant issues such as economical viability, compact optical buffering, and high-radix nanosecond-scale optical switches must be resolved.

Although our proposed approx networks would be amenable to these optical packet switching and circuit switching technologies, a quantitative evaluation is beyond the scope of this paper.

2.3 Modulation Techniques For High-bandwidth Optical Communication

Advanced modulation formats such as quadrature phase shift keying (QPSK) or M-QAM have been investigated in modern optical communication systems, particularly for long-haul systems. It has been experimentally demonstrated that advanced modulation formats can achieve higher spectral efficiency than simple on-off-keying (OOK), for instance, 428-Gbps dual polarization (DP) QPSK (107-GBd, 3.3-b/s/Hz) [24], 528-Gbps DP-16QAM (66-GBd, 6-b/s/Hz) [25], 516-Gbps DP-64QAM (43-GBd, 8-b/s/Hz) [26], 256-Gbps DP-256QAM (16-GBd) [27], and 50.53-Gbps PD-1024QAM-OFDM (3-GBd, 11.7-b/s/Hz) [28]. They use FEC techniques with 7%–31% overhead in throughput to compensate for signal degradation and to avoid intermediate signal regeneration during long (hundreds of kilometers) transmissions. In this study we use M-QAM but we do not use FEC (see Section 3).

2.4 Developing Applications for Unreliable Computers

Silent errors can be tolerated via a standard checkpoint-rollback-recovery approach, provided that failure detectors are available [29, 30]. A well-known drawback of checkpoint-rollback-recovery is the performance loss due to checkpointing overhead. Algorithm-based fault tolerance (ABFT) [5–7] is a domain-specific technique to detect and correct silent errors with low overhead (relying on checksums). Interestingly, a proposed application of ABFT is to reduce power consumption by disabling ECC hardware during memory accesses [31]. Although the goal of these methods is to recover from silent errors, another approach often termed "approximate computing" consists in developing applications that inherently tolerate silent errors [32], or in bounding their impact on application result quality. Regardless of the approach, the onus of tolerating silent errors is placed on the application and a large body of recent work provides techniques for designing applications that run effectively in silent error prone platforms.

Given that applications can tolerate silent errors, it is possible to relax the notion of correctness for hardware components, so as to aggressively pursue higher

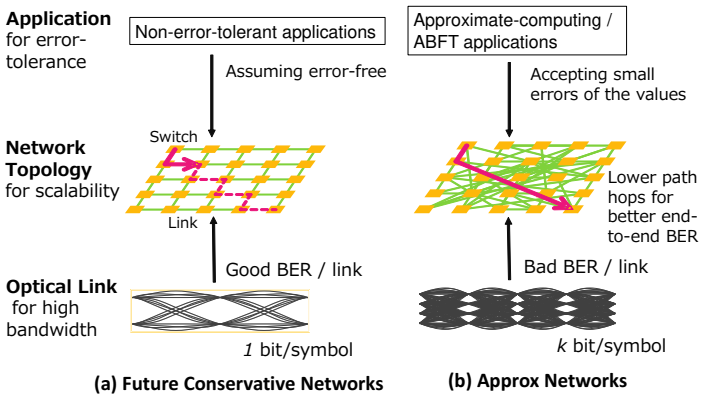

Figure 1: Baseline and approx network components.

performance and/or lower power consumption. Such "approximate" components have been investigated for RAM [31] and SSDs [33]. However, to the best of our knowledge, approximate networks have been considered only for wireless local area networks for H.264 data transfers [34] and for consumer applications on the Internet [35]. In this work, we propose to use the "approximate philosophy" for interconnection networks in datacenters or HPC systems.

3. APPROX NETWORKS

We propose approx networks that consist of electric switches interconnected via AOCs. A typical configuration of conventional and approx networks is compared in Figure 1. Each key element of the approx network approach is explained hereafter.

3.1 Optics

Conventional short-reach optical communication uses on-off-keying (OOK) or binary phase-shift keying (BPSK) as digital modulation. To achieve up to 10× link bandwidth improvements, approx networks use M-QAM ($16 \leq M \leq 1024$) as digital modulation for each optical link. When using 1,024-QAM, 10-bit per symbol can be transferred, while OOK and BPSK transfer 1-bit per symbol. QAM technologies used in long-haul optical communication network can be used for our proposed interconnects, as explained in Section 2.

Despite being a long-haul commodity, a 16-QAM transceiver that works without DSP can achieve a relatively low 32.5 pJ/bit power consumption (Finisar FTLC3321x3NL). Power consumption can be reduced by optimizing for short-haul use. The power consumption of DSP, which is required for advanced modulations, can also be reduced thanks to advances in LSI technology. Overall, current products and technology trends point to future power consumption sufficiently low to accommodate HPC systems, which will then benefit from features that AOC technology does not provide (e.g., patch panels). Generally, when compared to an equivalent FEC-protected network, the power consumption decreases in our proposed approx networks is

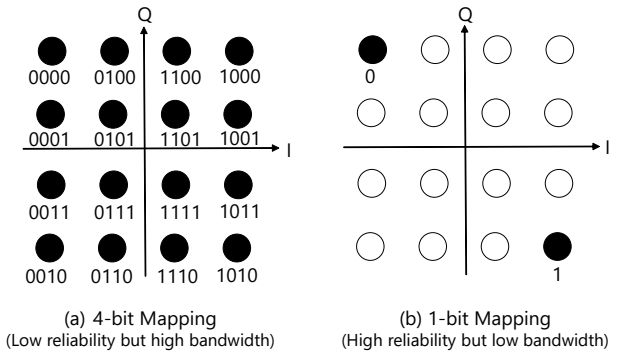

Figure 2: Bit mappings on 16-QAM.

in proportion to the power cost of FEC computation. Appropriate values of M for M-QAM can be selected to satisfy particular power consumption constraints.

M-QAM makes the margin to a neighboring symbol small. In long-haul network, FEC rather than CRC-based ARQ (Automatic Repeat-Request) is used in order to achieve good BER and avoid intermediate regeneration, as described in Section 2. Although CRC-ARQ has been used in interconnection networks [13], unreliable networks incur a large number of re-transmissions between switches.

Our proposed approx networks bypass FEC so as to avoid the incurred latency. As a result, they do not guarantee error-free communication. To reduce the effect of symbol errors, the conversion between signal and bit sequence uses the Gray code method to have a Hamming distance of 1 bit between neighbors, as shown in Figure 2(a). Figure 2(b) shows a alternative that has low error (due to larger inter-symbol distances), but that achieves bandwidth 4 times lower.

3.2 Network Topology

To maintain low BER of end-to-end communication, a network topology of switches would ideally have low diameter and low average shortest path length, both measured in numbers of hops. This is because bit flips may be introduced at every optical link. It is reasonable to consider a network topology that has a diameter at most 10. This stems from the equation of end-to-end BER, $1-(1-P_b)^n$, where P_b denotes the per-link BER and n the hop-count. For example, for $P_b = 10^{-5}$ and $n > 10$, the end-to-end BER is $> 10^{-4}$. In other terms, the end-to-end BER is less than an order of magnitude than the per-link BER when the diameter is at most 10. This low diameter requirement precludes the use of many traditional topologies at scale. Recently proposed random topologies [16] provide an attractive solution to build approx networks with low end-to-end BER. Similarly, high-radix network topologies such as Dragonfly and Slimfly [36] has low diameter and are also good candidates. These topologies make it possible to use approx networks in large-scale HPC platforms.

3.3 Applications

Approx networks are designed assuming that bit flips can occur with non-negligible probability. Target applications for these networks include approximate computing applications that can tolerate some bit flips to some extent and still produce acceptable results, and ABFT applications that autonomously cope with soft errors in their algorithm. The latter applications need to be executed with BER within acceptable range to complete the execution, which allows all bit flips to be handled automatically. By contrast, for the former, not all bit flips are equal and some could lead to dramatic data corruptions that would lead to unacceptable results. In typical scientific computing applications, for instance, floating point numbers encoded according to IEEE standards are being communicated. The IEEE 754 standard for floating-point numbers consists of sign, exponent and fraction bits. A bit flip in the sign and the exponent has likely a high impact on the numerical value of the floating point number. Instead, a bit flip in the fraction bit, and in particular in its least significant bits, has a smaller impact on the numerical value. We propose a "symbol-mapping" method to map bit sequences to numerical values in a way that reduces the numerical impact of bit flips. This method essentially enforces different BERs for different parts of the communicated data, so as to strike a good compromise between expected error on numerical values and network performance.

To flexibly adjust BER according to the bit rank, we use redundant mapping between bit sequences and symbols on QAM, so that larger distance between symbols can offer better symbol recognition accuracy, and thus data fidelity. In the example of Figure 2(b), a single bit is sent, thus maintaining an equivalent or better BER than that of OOK, but giving up the bandwidth improvement of QAM. We describe our bit protection mechanism in more detail, along with theoretical models, in the next section.

Hardware implementation of variable coding on QAM system is feasible in practice, because it only requires a code translator logic between (de)modulator and network interfaces. For an M-QAM system, the code translator logic receives a $(\log_2 M)$-bit symbol from demodulator and converts it to a 1- to $(\log_2 M)$-bit symbol referring a code mapping table for network interfaces. Then, NICs receive bit-protection metadata for data to be sent/received from commanding program to manipulate the different classes of (de)modulated data, i.e., protected/unprotected. Importantly, in this framework the transceivers and switches control the same symbols and bit sequences in payload as those processed in an M-QAM system, meaning that switch-by-switch type awareness is not required. We assume this approach is integrated into Ethernet MAC/PHY controllers, and we have verified its feasibility with FPGA-based network interfaces [37]. Details and evaluation are provided in Section 6.1.

4. ANALYTICAL EVALUATION

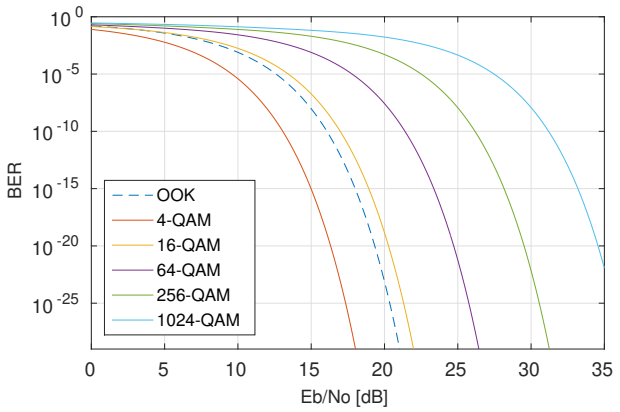

Figure 3: BER versus Eb/No for optical modulation.

In this section we analyze the relationship between transmission noise, numerical data error and link BER. The results are used to instantiate discrete-event simulation experiments presented in Section 5.

4.1 Bit Error Rate vs. E_b/N_0

We analyze the energy per bit to noise power spectral density ratio (E_b/N_0). We assume that the additive noise follows the Gaussian probability distribution function $\mathcal{N}(0, N_0/2)$. The probability of symbol error, P_s, for the Additive White Gaussian Noise (AWGN) channel is known to be:

$$P_s = \frac{1}{2}a(M)\text{erfc}\left(\frac{d(M,E_b)}{\sqrt{4N_0}}\right), \quad (1)$$

where the function $a(M)$ denotes the average number of the nearest neighbors in the constellation and the function $d(M, E_b)$ is the minimum distance between symbols [38].

The theoretical BER for OOK, $P_{b,\text{OOK}}$, is thus given by:

$$P_{b,\text{OOK}} = \frac{1}{2}\text{erfc}\left(\sqrt{\frac{E_b}{2N_0}}\right). \quad (2)$$

By contrast, M-QAM exploits the orthogonality property of two carrier signals, and its bit error rate ($P_{bc,\text{M-QAM}}$) and symbol error rate ($P_{sc,\text{M-QAM}}$) can be calculated as:

$$\begin{aligned} P_{sc,\text{M-QAM}} &= 2\left(1-\frac{1}{\sqrt{M}}\right)\text{erfc}\left(\sqrt{\frac{6\log_2(M)E_b}{M-1}\frac{1}{4N_0}}\right) \\ P_{bc,\text{M-QAM}} &= \frac{1}{\log_2 M}P_{sc,\text{M-QAM}}. \end{aligned} \quad (3)$$

Considering two identical probability spaces, the bit error rate of M-QAM system $P_{b,\text{M-QAM}}$ is given as

$$P_{b,\text{M-QAM}} = 1-(1-P_{bc,\text{M-QAM}})^2. \quad (4)$$

Figure 3 plots the required E_b/N_0 values to obtain given BER values on raw optical links. The value of E_b/N_0 is usually related to the signal-to-noise ratio (SNR). As expected, 1,024-QAM requires better E_b/N_0 than OOK to obtain a given BER. Although testbed measurements for 428-Gbps DP-QPSK have shown 10^{-5} BER [24], higher-order QAM, such as 64-QAM, have lower BER [25,28]. Note that it is expected for 16-1,024 QAM to become practical as the technology scaling improves.

4.2 Symbol Error Rate Exploration

4.2.1 *Model*

In the given modulation systems, bit flips occurrences in the links and transceivers are analogous to a product of simple Additive White Gaussian Noise (AWGN). This is because we assumes optical networks over short distances, meaning that the waveform quality deterioration caused by optical modes, which cause waveform deformation, is small enough to be ignored. Instead, errors due to device limitations, such as transceivers and optical lasers, take on more significance.

An error that occurs in a symbol of M-QAM is expressed by error vector $\boldsymbol{e} = \boldsymbol{w}-\boldsymbol{v}$, which is the difference vector of the measured symbol vector $\boldsymbol{v}$ and the ideal symbol vector $\boldsymbol{w}$. The error vector has i-component and q-component, which correspond to two coordinates on the IQ plane, each with their own probability space. The main point is that the probability of multiple bit flips in a symbol differs from that of one bit flip, and it would affect both our error model and symbol mapping strategy due to possible anomalous erring behavior. Given that the distribution of each error vector component follows a normal distribution independently, let us determine the probability that more than one bit flip occur in a symbol, which, combined with the bit-error model used in the following section, defines the overall BER.

Let $\mathcal{G}_{(0,\sigma^2)}(x)$ be the cumulative distribution function (CDF) of the normal distribution $\mathcal{N}(0,\sigma^2)$.

$$\mathcal{G}_{(0,\sigma^2)}(x) = \frac{1}{2}\left(1+\text{erf}\left(\frac{x}{2\sigma^2}\right)\right) \quad (5)$$

The general symbol error rate can be computed as follows. First, the probability of one component in the error vector to be out of range from a symbol in a set I, which we defined as those not at the periphery of the constellation, is expressed as

$$p_{ces\in I} = 2\,\mathcal{G}_{(0,N_0/2)}\left(\frac{d(M,E_b)}{2}\right), \quad (6)$$

given the same assumption as the previous section. Then, the symbol error rate is calculated from the following equation

$$p_{es\in I} = 1-(1-p_{ces\in I})^2. \quad (7)$$

In the same way, probability of error at the symbol in corners (set K) and the rest at the periphery (set J) can be calculated from $p_{es\in K} = 1-(1-p_{ces\in I})(1-p_{ces\in I}/2)$

and $p_{ces\in J} = 1-(1-p_{ces\in I}/2)^2$, and the overall symbol error rate is given as

$$P_s = \sum_{s\in I} p_{es\in I} + \sum_{s\in J} p_{es\in J} + \sum_{s\in K} p_{es\in K}. \quad (8)$$

Usually, the derivation of the symbol error rate involves many approximations, thus only symbol errors that cause single bit errors are considered in the general formula, such as one given in Equation 3.

Hereafter we first assess the probability of double bit flips in a symbol to show that their effect can be converged by our model, and then propose possible design spaces for bit coding for QAM systems. The case of double bit flips can be classified into two factors, (1) mis-demodulation to the nearest symbols in the diagonal line, and (2) mis-demodulation to the 2nd-nearest symbols in the horizontal/vertical line. To calculate the probability of the diagonal mis-demodulation (case(1)) $P_{e_\mathbf{d}}$, we use the error vector model again:

$$\begin{aligned} p_{ce_\mathbf{d}s\in I} &= 2\,\mathcal{G}_{(0,N_0/2)}\left(\frac{d(M,E_b)}{\sqrt{2}}\right), \\ p_{e_\mathbf{d}s\in I} &= p^2_{ce_\mathbf{d}s\in I}. \end{aligned} \quad (9)$$

This is the product of the probabilities that each of the error vector components is out-ranged. Probabilities $p_{e_\mathbf{d}s\in J}$ and $p_{e_\mathbf{d}s\in K}$ can be also calculated in the same way, and we can estimate $P_{e_\mathbf{d}}$ as in Equation 8. Our calculation yields that, given a 1024-QAM system with BER 10^{-5}, which means $SER = 10^{-4}$, the diagonal symbol error rate $P_{e_\mathbf{d}}$ is 6.25×10^{-10}.

Likewise, the probability of 2-hop mis-demodulation (case(2)) $P_{e_\mathbf{2h}}$ is given by

$$\begin{aligned} p_{ce_\mathbf{2h}s\in I} &= 2\,\mathcal{G}_{(0,N_0/2)}\left(\frac{3d(M,E_b)}{2}\right), \\ p_{e_\mathbf{2h}s\in I} &= 1-(1-p_{ce_\mathbf{2h}s\in I})^2, \end{aligned} \quad (10)$$

and is accordingly calculated as $P_{e_\mathbf{2h}} = 6.12\times10^{-36}$ with the same assumptions and derivations.

The above implies that a device that can manipulate M-QAM can obtain better BER by limiting the symbols and retain more distance between symbols. For example, M/2 symbol mapping, as shown in Figure 4(a), can lead to almost the same BER as $P_{e_\mathbf{d}}$, and M/4 symbol mapping, as shown in Figure 4(b) can lead to better BER, as calculated by double the distance function in Equation 7. Assuming nominal BER 10^{-5}, our calculations show that M/2 symbol mapping leads to a BER of 6.25×10^{-10}, and M/4 symbol mapping leads to BER of 1.52×10^{-16}, while the bandwidth is reduced by a factor 2 and 4, respectively. Combining these mapping policy, we can flexibly adjust BER in accordance with bit priorities determined by bit positions in relevant data types.

4.2.2 *Numerical Error Range*

Our bit protection mechanism allows developers to determine to what extent they want to protect data fidelity. Although the wide range of error propagation behavior in different algorithms makes a comprehensive and general error estimation impossible, errors in one single value of a double precision floating point number can be statically calculated and can serve as a useful case study. We implemented a simple parallel application, using MPI (Message Passing Interface), in which processes exchange 10 million double precision floating point numbers through (simulated) approximate channels. We then calculated the *mean error*, i.e., the total accumulated error divided by total sent data. In the simulation we define the *protection level* as a metric of bit protection, indicating the number of bits to be protected counting from the most significant bit. We assumed that the BER is 10^{-5}.

Figure 5 plots the mean error versus the protection level. The 12-bit header comprises the sign bit and the exponent bits. As expected, the numerical error is very large when these first 12 bits are not protected, and the error then decreases steadily once these crucial bits are protected. In the next section, for various case studies, we choose protection levels that strike good compromises between data integrity and network performance.

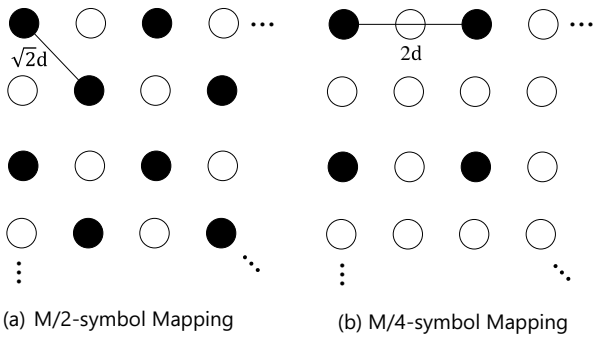

Figure 4: M/2-symbol and M/4-symbol mappings.

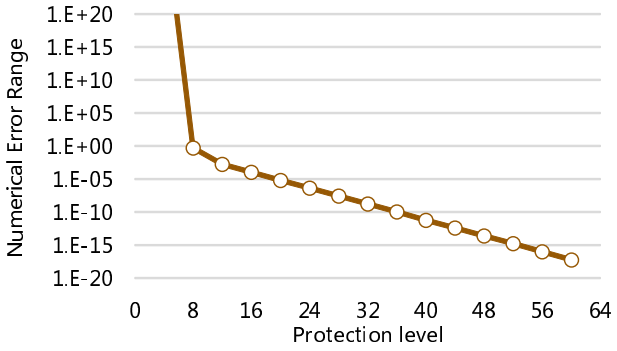

Figure 5: Numerical error range vs. protection level (bit rank).

5. EVALUATION

In this section we use discrete-event simulation to evaluate the performance of parallel applications and

benchmarks when executed on approx networks and on conventional networks. Specifically, we use the SimGrid simulation framework (v3.12) [39]. SimGrid implements validated simulation models, is scalable, and makes it possible to simulate the execution of unmodified parallel applications that use the Message Passing Interface (MPI) [40]. We implemented the bit-error model of optical links described in the previous section and our proposed bit protection mechanisms within the SimGrid simulation models.

5.1 Methodology

5.1.1 Network Configuration

We compare a conventional interconnection network as expected in the near future (Conv. NW), a high-bandwidth implementation of this same network using FEC (FEC Perf. NW), and an approx network (Approx. NW). Networks characteristics are shown in Table 1. Link bandwidth is set to 100Gbps (25Gbps $\times 4$) in the conventional interconnection network, while the approx network has up to 1Tbps (250Gbps $\times 4$). Note that the bandwidth for approx networks varies according to the bit protection mechanism, as detailed in the next section.

We use three different network topologies: 3D torus, Dragonfly [15] and random topologies. For these topologies we use a shortest path routing scheme using Dijkstra's algorithm. To avoid deadlocks between paths we use topology-agnostic virtual-channel transition routing, a well-studied approach for irregular bi-directional networks [41]. Each switch has a 60 nsec delay and cable length is assumed to be 5m, for a link delay of 25ns. Each host has 4 cores, for an overall computation speed of 500 GFlops. We configure SimGrid to utilize its built-in version of the MVAPICH2 implementation of MPI collective communications [42]. Since a BER of 10^{-5} has been measured in 400Gbps single carrier DP-QPSK [24], we consider that high bandwidth (256Gbps - 1Tbps) using four lanes will be feasible at that BER. We emphasize that our approach is applicable to various kinds of QAM, i.e., 4, 16, 64, 256-QAM (200, 400, 600 and 800Gbps link used in the simulation, resp.), which can be chosen based on particular power consumption constraints.

5.1.2 Data Protection Mechanisms

To ensure perfect data integrity for designated parts of the communicated data, we assume the use of FEC or of reliable modulation schemes. We implemented the impact on latency and bandwidth due to these mechanisms in the simulation models used by SimGrid, allowing us to flexibly switch the network configuration in the simulation, i.e., bandwidth, link/switching latency, and BER.

Likewise, we implemented an interface that allows software to specify the relative importance of bits in data types used in MPI communication. While the network handles transactions with perfect data integrity for normal MPI data types, as specified by the data type argument passed on MPI send functions, we also define *Approximate Data Types* that can be used by the MPI application developer. Note that these approximate data types could offer cooperation with annotation protocols of other approximate computing works such as [43]. The relative importance of bits in these data types is defined as error masks by the developer ahead of time, making it possible to protect certain bits via the reliable communication methods described above. In this work, we assume two levels of bit importance, i.e., protected and unprotected.

Table 1: Default Network Parameters

	Link BW	Switch Latency	BER
Conv. NW	100Gbps	60 ns	Error Free
FEC Perf. NW	1Tbps	160 ns(FEC inc)	Error Free
Approx NW	$\leq$ 1Tbps	60 ns	$\leq 10^{-5}$

The trade-off between BER and latency/bandwidth with respect to the corresponding symbol mapping policy is simulated by a manually adjusting a bandwidth scaling factor. The bandwidth for the simulated approx network is thus configured semi-automatically by the simulator according to the symbol mapping policy, protection level, and transferred data type. More formally, let t_p be the fraction of protected bits, f_p the bandwidth factor for the reliable symbols, and s the data size in bits. The average bandwidth $\tilde{B}$ is then:

$$\tilde{B} = \frac{s}{\frac{t_p s}{f_p f(s) B} + \frac{(1-t_p)s}{f(s)B}} = \frac{f_p}{t_p + f_p - t_p f_p} f(s) B,$$

where B denotes the original bandwidth of the approx network (e.g., 1Tbps) and $f(s)$ is another bandwidth factor calculated in SimGrid based on the data size.

Note that routing and flow control information are protected within the same framework we established for payload data. Therefore, the BER of such information becomes the same as that for non-approximate networks.

5.2 Case-Study Applications

5.2.1 Approximate-Computing Applications

We simulate the execution of the CG and FFT applications of the NAS Parallel Benchmarks (version 3.3.1, MPI versions) [44], and a K-means clustering algorithm. Table 2 gives salient characteristics of these applications.

CG: Conjugate Gradient (CG) solves a linear system $A.x = b$, where A is a symmetric positive-definite matrix. The metric for result quality is then average mean error when compared to the result produced by an error-free execution.

FFT: Fast Fourier Transform (FFT) is a popular numerical application that is well-suited to an approximate computing approach [45]. The metric for result quality is average mean error when compared to the result produced by an error-free execution.

Table 2: Applications used in the evaluation. [1] Number of exchanged messages for this data type; [2] Fraction of exchanged messages that are of this data type; [3] Protection Level in bits; [4] 16-bit real part and 16-bit imaginary part.

Application	Description	Approximate Data Type	Count[1]	Fraction[2]	PL[3]
CG	Conjugate Gradient	`MPI_DOUBLE`	399,360	0.02	16
FFT	Fast Fourier Transform	`MPI_DOUBLE_COMPLEX`	2,882,519	0.99	16+16 [4]
K-means	Clustering	`MPI_FLOAT`	95,805,592	0.99	12
FT-MM	ABFT Matrix Multiply	`MPI_DOUBLE`	178,753,735	0.99	0
FT-CG	ABFT Conjugate Gradient	`MPI_DOUBLE`	1,791,200	0.08	0

K-means: The K-means clustering algorithm is a popular non-hierarchical clustering algorithm for vector quantization, used widely in data mining. We use a 1GB dataset with 11,500,000 points in a 9-dimensional space. The metric of result quality is the number of mis-classified points.

5.2.2 *Algorithm-Based Fault-Tolerant Applications*

We simulate ABFT versions of the Matrix Multiplication (MM) and Conjugate Gradient (CG) applications:

FT-MM: We use the algorithm in [46].When calculating $C = A \times B$, the algorithm encodes matrices A and B in a way that includes checksums provided by a checksum vector e. This encoding allows the algorithm to detect and correct soft errors at every few iterations of the computation.

Since the algorithm implements its own soft error tolerance, we relax the bit protection mechanism in the approx network, only ensuring the documented BER that the algorithm requires to keep making progress.

FT-CG: We use the fault tolerant CG algorithm in [8]. Unlike FT-MM, FT-CG does not employ checksums, but instead detect and correct errors using certain numerical invariants.

5.3 Application Execution Time

In this section we measure application execution times and determine the impact of network characteristics (switch delay, link bandwidth, and network topology). Note that we define the execution time as the wall-clock time between the start of the execution until its completion (which thus encompasses all computation and communication operations).

Figure 6 shows execution times, normalized to that on the conventional network (Conv.), for our three approximate computing application and our two ABFT applications, using a Dragonfly topology with 256 switches and 1024 processors. Values below 1 denote improvement over the conventional network. Note that a speedup of s for the overall execution time of a parallel application implies a communication speedup greater than s. While the approx network (Approx) uses bit protection mechanisms to limit data corruption, the fully-approximated network (Full-approx) exposes all communicated bits to a high BER. We include Full-approx to show an upper-bound on the performance gain using our approach (understanding that application results are likely unacceptable with Full-approx).

The left side of the figure shows results for our three approximate computing applications. The most notable result is that the approx network reduces execution time significantly compared to the conventional network and the FEC-perfect network, except for K-means for which FEC-perfect is better than Approx. For FFT, the approx network with appropriate bit protection achieves a speedup of 2.94 when compared the conventional network, and 1.98 when compared to the FEC-perfect network. For CG, only 2% of the transferred messages allow soft errors to occurs, and yet the execution time on the approx network is close to that of the Full-approx network. This implies that using our approach even for only a small portion of the communication payload can alleviate the communication bottleneck and lead to significant performance improvements.

The only exception, K-means, is a case in which the approx network is not beneficial when compared to FEC-Perfect. This is because the K-means is one of the applications that are inherently non-latency-sensitive or that are good candidates for standard latency-hiding techniques. In fact, it has been reported that the latency of data transfer does not significantly impact the performance of some K-means implementations [47]. Note that our approach can coexist with FEC, and would be implemented in practice by overriding or bypassing existing FEC procedures. Therefore, before execution, an application could switch to using the network in a standard FEC mode. Nevertheless, our proposed approx networks should be beneficial for various classes of applications, as demonstrated for two of our three case-study applications.

The right side of Figure 6 shows results for our two ABFT applications. Recall that for these applications we did not use our selective bit protection mechanism since the algorithm can autonomously detect and correct bit flips provided the BER is sufficiently low. This is why the Approx and Full-Approx results are identical in the figure. The trade-off is between error protection and the redundant computation for checksums, not the bandwidth limitation. In our proposed approach, the whole BER can be flexibly tuned by the symbol mapping as described in Section 4.2. Here we assume the network can provide BER 10^{-7} (the impact of bandwidth restriction is evaluated in Section 5.5). Our results demonstrate that the ABFT applications achieve significant performance improvements when using approx networks.

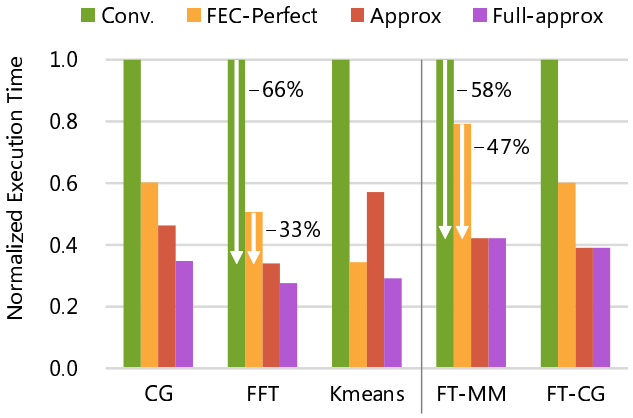

Figure 6: Application execution time normalized to that on a conventional network (Dragonfly, 256 switches).

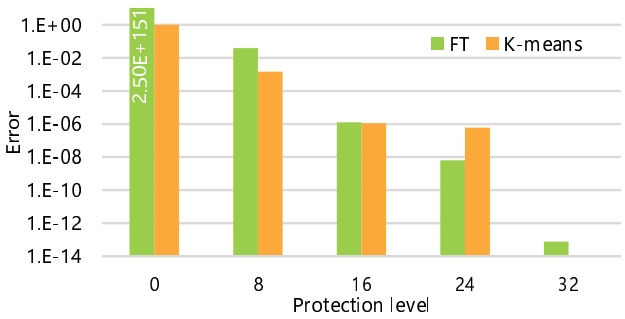

Figure 7: QoR trade-off (FFT and Kmeans).

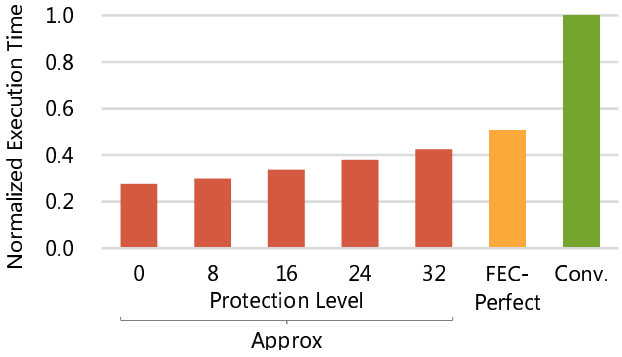

Figure 8: Normalized execution time (FFT).

5.4 Quality of Results

While most results in the previous section show significant performance improvements, a clear concern is that of the quality of the results produced by applications running on approx networks. The hope is that attractive trade-offs between performance and result quality can be achieved, e.g., by tuning the bit protection level. In this section we discuss the approximate computing applications (CG, FFT, and K-means). This is because the ABFT applications experience no results quality loss since they automatically correct soft errors in software.

Setting the bit protection level at 16 (see Section 5.2.1), we found that all executions of CG successfully converge to the same solution as that computed in an error-free environment.

Mean result error vs. bit protection level results are shown in Figure 7 for both FFT and K-means. These results demonstrate that protecting important bits (e.g., for FFT the sign and exponent bits for both real and imaginary part of complex double) is critical. Error decreases by several orders of magnitude once these bits are protected. For instance, see the drop by more than 4 orders of magnitude for FFT when going from protection level 8 to protection level 16. We mention that the error of K-means is almost the same for protection level 16 and 24. This is because we use the fraction of mis-classified points as the metric, thus one single error rises the error on the order of 10^{-7} as we use 11,500,000 points.

To highlight the quality-performance trade-off, Figure 8 plots the execution time normalized to that on the conventional network, showing approx network results for various bit protection levels, for FFT. The performance results in the previous section are for bit protection level 16, which leads to a mean error around 1.2×10^{-6}. Going to protection level 24 takes the error down 3 orders of magnitude, and only causes a small loss in performance. Even with bit protection level at 32, in which case the error is another 5 orders of magnitude lower, performance is better than that of conventional networks (and of the FEC-Perfect network).

Overall, our results suggest that approx networks provide attractive trade-offs between result quality and performance. They achieve the "flexibility of the quality control" discussed in the approximate computing literature, with the selection of the bit protection level being the "control knob."

5.5 Modulations and Link Bandwidth

Our expectation is that near-future optical network technology used in large-scale computing systems will be able to maintain 1Tbps link bandwidth with relatively high BER (10^{-5}). However, many factors are at play, including equipment cost, and it is thus likely that the upper bound on link bandwidth will be defined by the performance of installed market-driven network equipment. In this section, we investigate the impact of the upper-bound link bandwidth, which stems from the modulation technology in use, on the performance gains that can be achieved by approx networks.

We assume a network of links with bandwidth 100, 200, 400, 600, 800Gbps and 1Tbps, which corresponds to 4-QAM, 16-QAM, 64-QAM, 256-QAM, and 1024-QAM systems. Figure 9 shows execution times, normalized to that on a conventional network, vs. link bandwidth for the CG and FT-MM applications, for

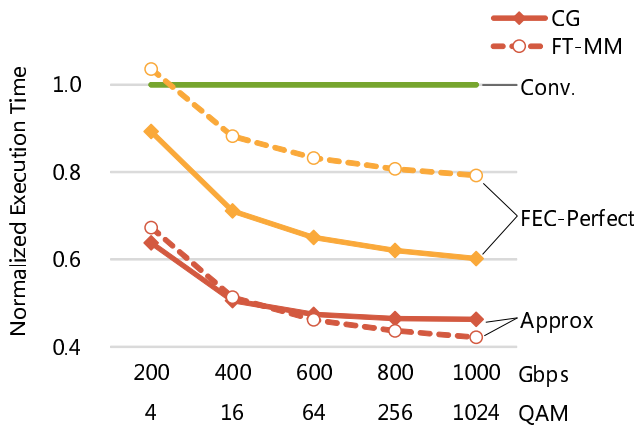

Figure 9: Normalized execution time vs. link bandwidth (CG and FT-MM).

conventional, FEC-perfect, and approx networks. We observed similar trends for our other benchmark applications (results not included). For the CG application, the important observation is that the performance saturates to a certain value, and that even when simple modulation technology is employed, such as 16-QAM and 64-QAM, performance gains due to approx networks are significant. Results are similar for the ABFT FT-MM application. ABFT applications benefit from approx networks despite the requirement of a moderately lowered BER, which is enabled by sacrificing some of the bandwidth gains.

5.6 Network Topology

In this section we present results for several topologies of 256 high-radix switches: Dragonfly topology [15] of degree 11 (df11); random topology of degree 11 (r11); random topology of degree 17 (r17); torus of degree 8 (tor8); random topology of degree 8 (r8). In addition we present results for two 64-switch topologies: torus of degree 6 (tor6); random topology of degree 6 (r6). In our simulations, the difference in average hop count did not lead to significant differences on application result quality, in part due to the limited problem size. However, its impact on application execution time is significant, as seen hereafter.

Figure 10 shows application execution times for the approximate computing application benchmarks FFT and CG, normalized to the execution time for the df11 topology (for 256-switch topologies) and that for the tor8 topology (for 64-switch topologies). Results are shown for conventional networks, FEC-Perfect networks, and approx networks. One observation, which confirms the findings in [48], is that random topologies are well-suited to applications with all-to-all communication patterns (such as FFT, for which conventional r11 outperforms conventional df11); while regular topologies like Dragonfly are well-suited to applications with regular communication patterns (such as CG, for which conventional df11 outperforms r11). Regardless, these results show that our proposed approx networks lead to significant performance improvements when compared to conventional networks (and when compared for FEC-Perfect networks in most cases) for 256-switch topologies. Large improvements are also achieved at lower scale for 64-switch topologies, but in some cases approx networks do not improve on FEC-Perfect networks.

Figure 10 also shows similar results for the ABFT FT-MM application. Approx networks leads to significant performance benefits across the board when compared to conventional networks and FEC-Perfect networks.

5.7 Switch Delay

In this section we quantify the impact of switch delay, which has been set to 60 nsec in all that precedes, on our results. Figure 11(a) shows execution time results for the approximate computing FFT application, for switch delays of 40 nsec, 60 nsec, and 100 nsec. Since the FEC is assumed to take the same processing time regardless of switch delay, i.e., 100nsec, its impact is larger as the switch delay is smaller. Results in the figure show that, as expected, for all three kinds of networks (conventional, FEC-Perfect, and approx) application execution time decreases as switch delay decreases. Importantly, the relative performance gain of approx networks also increases as switch delay decreases. For instance, improvement over the conventional network, resp. the FEC-Perfect network, is 2.45×, resp. 1.41×, for switch delays of 100nsec, but a higher 3.31×, resp. 1.54×, for switch delay of 40 nsec. Figure 11(b) shows similar results, with similar trends, for the ABFT FT-MM application. Overall, we conclude that our proposed approx networks will be even more attractive in the upcoming low-switch-delay era.

6. DISCUSSION

6.1 Implementation Complexity of Line Rate Control

A hardware implementation of the variable coding scheme on QAM systems requires simple code translator modules between modulator/demodulator and a network interface, referred as data-to-symbol and symbol-to-data translator modules. These translator modules are first notified by program about configuration of the mapping table associated with specific message type or data type, and they code/decode input data/symbols referring the mapping table.

We demonstrate feasibility of the hardware implementation. Since 1Tb/s Ethernet will not be available soon, we assume to aggregate ten FPGA-based 100Gb/s interfaces (e.g., NetFPGA-SUME [37]) to achieve 1Tb/s. The data-to-symbol and symbol-to-data translator modules are implemented at such network interfaces. To achieve 100Gb/s with a 1,024 QAM scheme, for example, the symbol-to-data translator module that translates 40 symbols into up to 400 bits depending on the mapping table is required to be operated at 250 MHz, and it is reasonable for state-of-the-art FPGA devices, such as Xilinx Virtex-7 690T used in NetFPGA-SUME.

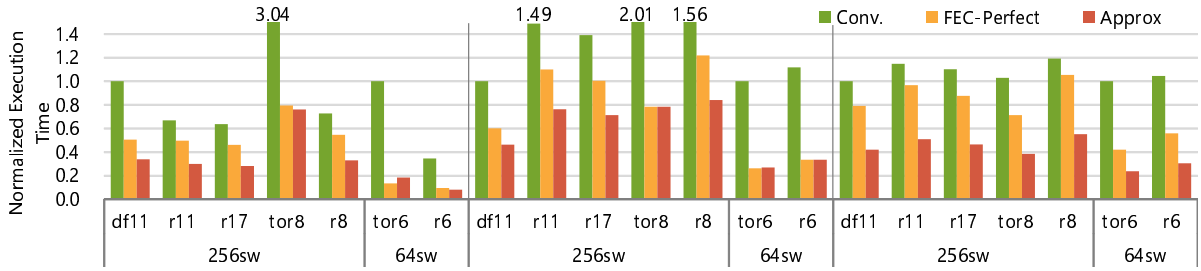

Figure 10: Performance comparison with different network topology (FFT, CG and FT-MM).

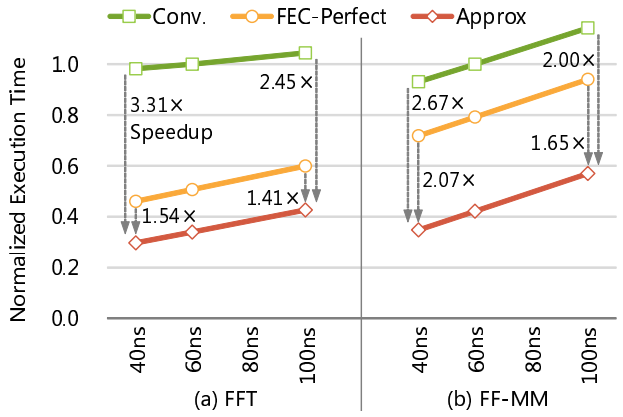

Figure 11: Normalized execution time vs. switch delay (FFT and FT-MM).

Table 3: Synthesis results of translator modules using Xilinx ISE 13.4 for Virtex-7 690T.

	Data-to-symbol	Symbol-to-data
# of Slice Reg	10,641 (1%)	10,640 (1%)
# of Slice LUTs	148,099 (34%)	147,699 (34%)
Clock period	3.232ns (309MHz)	2.897ns (345MHz)

Such 400-bit data-to-symbol and 40-symbol symbol-to-data translator modules with 10-kbit mapping tables were designed with Verilog HDL. Then they were synthesized with Xilinx ISE 13.4 for Virtex-7 690T. Table 3 shows the synthesis results. Although these modules consume substantial FPGA resources (e.g. 34% of total amount of slice LUTs) for supporting a symbol-level reconfigurability, both of them satisfy the 250MHz timing constraint, demonstrating the feasibility in performance.

6.2 Variable QAM vs. TMR

To send protected data on unreliable networks, it seems attractive to use alternative way using Triple Modular Redundancy (TMR) instead of employing variable QAM modulation, considering its overhead in redundancy. However, the overhead resides also in buffering time for TMR system, let alone FEC, as explained in Section 1. In fact, given the augmented bandwidth by QAM, the overhead of data size and transfer time is much lower than the overhead of buffering. This is because the QAM requires only line rate control whereas FEC requires data-block buffering, e.g., 2K.

6.3 Limitations

The protection scheme for compressed or scrambled data is a general argument in the field of approximate computing, and such approaches clearly require full protection. In this work we do not consider these approaches, because they usually require non-negligible latency overhead of data decompression. This is the same rationale for bypassing FEC latency overhead, one of the objectives of this work.

7. CONCLUSIONS

Computational applications are subject to various kinds of numerical errors, ranging from deterministic round-off errors to non-deterministic soft errors due to bit flips. In this context we proposed high-bandwidth, low-latency approximate networks. Our approach relies on the codesign of different network layers, i.e., QAM, the use of low hop-count topologies, and targets soft-error-tolerant applications.

Through our analysis of transmission noise, numerical data error and link BER, we found that given a baseline high BER of 10^{-5}, for an optical link, our proposed symbol mapping technique can lead to an improved BER of 10^{-16} (with a 4× reduction in bandwidth), or even more. As a result, our network makes it possible to enact different trade-offs between performance and reliability. Furthermore, we have proposed a bit protection scheme so that different trade-offs can be chosen for different bits in application data, depending on the importance of different bits for the numerical value of the application data. Using discrete-event simulation, we have evaluated our approach for two classes of applications: 1) approximate computing applications that can tolerate some data corruption and still produce valuable results; and 2) applications that employ Algorithm Based Fault Tolerance (ABFT) to detect and correct data corruption in software automatically. We have conducted extensive simulation experiments and analyses of the results. The main takeaway is that approx networks lead to significant performance improvements to applications, with speedups up to 2.94 when compared to the use of conventional networks.

8. REFERENCES

[1] F. Cappello, A. Geist, W. Gropp, S. Kale, B. Kramer, and M. Snir, "Toward exascale resilience: 2014 update," *Supercomputing frontiers and innovations*, vol. 1, no. 1, 2014.

[2] A. Dixit and A. Wood, "The impact of new technology on soft error rates," in *IEEE International on Reliability Physics Symposium (IRPS)*, pp. 5B.4.1–5B.4.7, 2011.

[3] S. Mittal, "A Survey of Techniques for Approximate Computing," *ACM Comput. Surv.*, vol. 48, pp. 62:1–62:33, Mar. 2016.

[4] D. Fiala, F. Mueller, C. Engelmann, R. Riesen, K. Ferreira, and R. Brightwell, "Detection and correction of silent data corruption for large-scale high-performance computing," in *Proc. of the ACM/IEEE SC*, 2012.

[5] K.-H. Huang and J. A. Abraham, "Algorithm-based fault tolerance for matrix operations," *IEEE Trans. Comput.*, vol. 33, no. 6, pp. 518–528, 1984.

[6] G. Bosilca, R. Delmas, J. Dongarra, and J. Langou, "Algorithm-based fault tolerance applied to high performance computing," *J. Parallel and Distributed Computing*, vol. 69, no. 4, pp. 410 –416, 2009.

[7] M. Shantharam, S. Srinivasmurthy, and P. Raghavan, "Fault tolerant preconditioned conjugate gradient for sparse linear system solution," in *Int. Conf. Supercomputing*, 2012.

[8] Z. Chen, "Online-ABFT: An Online Algorithm Based Fault Tolerance Scheme for Soft Error Detection in Iterative Methods," in *Proc. 18th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, pp. 167–176, 2013.

[9] A. R. Benson, S. Schmit, and R. Schreiber, "Silent error detection in numerical time-stepping schemes.," *CoRR*, vol. abs/1312.2674, 2013.

[10] P. Sao and R. Vuduc, "Self-stabilizing Iterative Solvers," in *Proc. ScalA'13*, ACM, 2013.

[11] M. Andrewartha, B. Booth, and C. Roth, "Feasibility and Rationale for 3m no-FEC server and switch DAC." `http://www.ieee802.org/3/by/public/Sept15/andrewartha_3by_01a_0915.pdf`, Sept. 2015.

[12] B. Towles, J. P. Grossman, B. Greskamp, and D. E. Shaw, "Unifying On-Chip and Inter-Node Switching Within the Anton 2 Network," in *ISCA*, pp. 1–12, June. 2014.

[13] S. Scott, D. Abts, J. Kim, and W. J. Dally, "The BlackWidow High-Radix Clos Network," in *ISCA*, pp. 16–28, 2006.

[14] D. Abts and J. Kim, *High Performance Datacenter Networks: Architectures, Algorithms, and Opportunities*. Morgan and Claypool Publishers, 2011.

[15] J. Kim, W. J. Dally, S. Scott, and D. Abts, "Technology-Driven, Highly-Scalable Dragonfly Topology," in *ISCA*, pp. 77–88, 2008.

[16] M. Koibuchi, H. Matsutani, H. Amano, D. F. Hsu, and H. Casanova, "A Case for Random Shortcut Topologies for HPC Interconnects," in *ISCA*, pp. 177–188, 2012.

[17] C. Minkenberg, G. Rodriguez, B. Prisacari, L. Schares, P. Heidelberger, D. Cheng, , and C. Stunkel, "Performance benefits of optical circuit switches for large-scale dragonfly networks," in *Optical Fiber Communication Conference (OFC)*, p. W3J.3, 2016.

[18] K. Christodoulopoulos, K. Katrinis, M. Russini, and D. O. Mahony., "Acceleratig HPC Workloads with Dynamic Adaptation of a Software-Defined Hybrid Electronic/Optical Interconnect," in *Optical Fiber Communication Conference (OFC)*, p. Th2A.11, Mar. 2014.

[19] Z. Zhu and S. Zhong, "Scalable and Topology Adaptive Intra-data Center Networking Enabled by Wavelength Selective Switching," in *Optical Fiber Communication Conference (OFC)*, p. Th2A.60, Mar. 2014.

[20] G. M. Saridis, E. Hugues-Salas, Y. Yan, S. Yan, S. Poole, G. Zervas, and D. Siomenidou, "DORIOS: Demonstration of an All-Optical Distributed CPU, Memory, Storage Intra DCN Interconnect," in *Optical Fiber Communication Conference (OFC)*, p. W1D.2, Mar. 2015.

[21] R. Proietti, Z. Cao, Y. Li, and S. J. B. Yoo, "Scalable and Distributed Optical Interconnect Architecture based on AWGR for HPC and Data Cetners," in *Optical Fiber Communication Conference (OFC)*, p. Th2A.59, Mar. 2014.

[22] W. Miao, F. Agraz, H. de Waardt, S. Spadaro, H. J. S. Dorren, and N. Calabretta, "1.3 um SDN-enabled Optical Packet Switch Architecture for High Performance and Programmable Data Center Network," in *Optical Fiber Communication Conference (OFC)*, p. Th2A.66, Mar. 2015.

[23] R. Takahashi, S. Ibrahim, T. Segawa, T. Nakahara, H. Ishikawa, Y. Suzaki, Y. Huang, K. Kitayama, and A. Hiramatsu, "A Torus Datacenter Network Based on OPS/OCS/VOCS Enabled by Smart Flow Management," in *Optical Fiber Communication Conference (OFC)*, p. W3D.4, Mar. 2015.

[24] G. Raybon, A. L. Adamiecki, P. Winzer, C. Xie, A. Konczykowska, F. Jorge, J.-Y. Dupuy, L. L. Buhl, S. D. S. Chandrashekhar, M. Grove, and K. Rush, "Single-Carrier 400G Interface and 10-Channel WDM Transmission over 4800 km using All-ETDM 107-Gbaud PDM-QPSK," in *Optical Fiber Communication Conference (OFC)*, p. PDP4A.5, 2013.

[25] R. Rios-Muller, J. Renaudier, P. Brindel, C. Simonneau, P. Tran, A. Ghazisaeidi, I. Fernadez, L. Sxhmalen, and G. Charlet, "Optimized Spectrally Efficient Transceiver for 400-Gb/s Single Carrier Transport," in *The European Conference on Optical Communication (ECOC)*, p. PD.4.2, 2014.

[26] J. R. O. Bertran-Pard and, H. Mardoyan, P. Tran, R. Rios-Muller, A. Konczykowska, J.-Y. Dupuy, F. Jorge, M. Riet, B. Duval, J. Godin, S. Randel, G. Charlet, and S. Bigo, "Transmission of 50-GHz-Spaced Single-Carrier Channels at 516Gb/s over 600km," in *Optical Fiber Communication Conference/National Fiber Optic Engineers Conference (OFC/NFOEC)*, p. OTh4E.2, 2013.

[27] H.-C. Chien, Z. Jia, and J. Yu, "256-Gb/s Single-Carrier PM-256QAM Implementation using Coordinated DD-LMS and CMA Equalization," in *The European Conference on Optical Communication (ECOC)*, p. Mo.3.3.2, 2015.

[28] M.-F. Huang, D. Qian, and E. Ip, "50.53-gb/s pdm-1024qam-ofdm transmission using pilot-based phase noise mitigation," in *Opto-Electronics and Communications Conference (OECC)*, pp. 752–753, July 2011.

[29] G. Bronevetsky and B. de Supinski, "Soft error vulnerability of iterative linear algebra methods," in *Int. Conf. Supercomputing*, pp. 155–164, 2008.

[30] T. Hl'SSrault and Y. Robert, eds., *Fault-Tolerance Techniques for High-Performance Computing*, Computer Communications and Networks, Springer Verlag, 2015.

[31] D. Li, Z. Chen, P. Wu, and J. S. Vetter, "Rethinking Algorithm-based Fault Tolerance with a Cooperative Software-hardware Approach," in *SC*, pp. 44:1–44:12, 2013.

[32] M. Heroux and M. Hoemmen, "Fault-tolerant iterative methods via selective reliability," Research report SAND2011-3915 C, Sandia National Laboratories, 2011.

[33] A. Sampson, J. Nelson, K. Strauss, and L. Ceze, "Approximate storage in solid-state memories," in *The 46th Annual IEEE/ACM International Symposium on Microarchitecture, (MICRO)*, pp. 25–36, 2013.

[34] S. Sen, S. Gilani, S. Srinath, S. Schmitt, and S. Banerjee, "Design and implementation of an "approximate" communication system for wireless media applications," in *Proceedings of the ACM SIGCOMM 2010 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 15–26, 2010.

[35] B. Ransford and L. Ceze, "SAP: an architecture for selectively approximate wireless commu nication," *CoRR*,

vol. abs/1510.03955, 2015.

[36] M. Besta and T. Hoefler, "Slim Fly: A Cost Effective Low-diameter Network Topology," in *SC*, pp. 348–359, 2014.

[37] "NetFPGA." `http://netfpga.org`.

[38] P. Vaidyanathan, S.-M. Phoong, and Y.-P. Lin, *Signal processing and optimization for transceiver systems*. Cambridge University Press, 2010.

[39] SimGrid: Versatile Simulation of Distributed Systems. `http://simgrid.gforge.inria.fr/`.

[40] H. Casanova, A. Giersch, A. Legrand, M. Quinson, and F. Suter, "Versatile, Scalable, and Accurate Simulation of Distributed Applications and Platforms," *Journal of Parallel and Distributed Computing*, vol. 74, no. 10, pp. 2899–2917, 2014.

[41] J. Flich, T. Skeie, A. Mejia, O. Lysne, P. Lopez, A. Robles, J. Duato, M. Koibuchi, T. Rokicki, and J. C. Sancho, "A Survey and Evaluation of Topology Agnostic Deterministic Routing Algorithms," *IEEE Trans. on Parallel and Distributed Systems*, vol. 23, no. 3, pp. 405–425, 2012.

[42] "MVAPICH: MPI over InfiniBand, 10GigE/iWARP and RoCE." `http://mvapich.cse.ohio-state.edu/`.

[43] A. Sampson, W. Dietl, E. Fortuna, D. Gnanapragasam, L. Ceze, and D. Grossman, "EnerJ: Approximate data types for safe and general low-power computation," in *ACM SIGPLAN Notices*, vol. 46, pp. 164–174, ACM, 2011.

[44] The NAS Parallel Benchmarks. `http://www.nas.nasa.gov/Software/NPB/`.

[45] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger, "Neural acceleration for general-purpose approximate programs," in *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 449–460, IEEE Computer Society, 2012.

[46] P. Wu, C. Ding, L. Chen, F. Gao, T. Davies, C. Karlsson, and Z. Chen, "Fault tolerant matrix-matrix multiplication: correcting soft errors on-line," in *Proceedings of the second workshop on Scalable algorithms for large-scale systems*, pp. 25–28, ACM, 2011.

[47] P. Kraj, A. Sharma, N. Garge, R. Podolsky, and R. A. McIndoe, "Parakmeans: Implementation of a parallelized k-means algorithm suitable for general laboratory use," *BMC Bioinformatics*, vol. 9, no. 1, pp. 1–13, 2008.

[48] F. Chaix, I. Fujiwara, and M. Koibuchi, "Suitability of the random topology for hpc applications," in *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, pp. 301–304, Feb 2016.