Advanced System-Scale and Chip-Scale Interconnection Networks for Ultrascale Systems

John Shalf

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master's of Science
in
Electrical Engineering

Peter Athanas, Chair
Wu-Chun Feng
Scott Midkiff

December 14, 2010
Blacksburg, Virginia

Keywords: Interconnects, Exascale, Manycore, Energy Efficiency, Photonics

Advanced System Scale and Chip-Scale Interconnection Networks for Ultrascale Systems

John Shalf

(ABSTRACT)

The path towards realizing next-generation petascale and exascale computing is increasingly dependent on building supercomputers with unprecedented numbers of processors. Given the rise of multicore processors, the number of network endpoints both on-chip and off-chip is growing exponentially, with systems in 2018 anticipated to contain thousands of processing elements on-chip and billions of processing elements system-wide. To prevent the interconnect from dominating the overall cost of future systems, there is a critical need for scalable interconnects that capture the communication requirements of target ultrascale applications. It is therefore essential to understand high-end application communication characteristics across a broad spectrum of computational methods, and utilize that insight to tailor interconnect designs to the specific requirements of the underlying codes. This work makes several unique contributions towards attaining that goal. First, the communication traces for a number of high-end application communication requirements, whose computational methods include: finite-difference, lattice-Boltzmann, particle-in-cell, sparse linear algebra, particle mesh ewald and FFT-based solvers.

This thesis presents an introduction to the *fit-tree* approach for designing network infrastructure that is tailored to application requirements. A fit-tree minimizes the component count of an interconnect without impacting application performance compared to a fully connected network. The last section introduces a methodology for reconfigurable networks to implement fit-tree solutions called Hybrid Flexibly Assignable Switch Topology (HFAST). HFAST uses both passive (circuit) and active (packet) commodity switch components in a unique way to dynamically reconfigure interconnect wiring to suit the topological requirements of scientific applications. Overall the exploration points to several promising directions for practically addressing both the on-chip and off-chip interconnect requirements of future ultrascale systems.

This work was supported by the Office of Advanced Scientific Computing Research in the Department of Energy Office of Science under contract number DE-AC02-05CH11231.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Computing technology has been a significant and pervasive driving force in the global technology market over the past two decades. It affects nearly every aspect of life from education, entertainment, transportation and personal communication to the basic infrastructure in our economy, medicine, engineering and science. Society has come to depend not just on computing but on the increases in computing capability that have been available each year for given cost and power budget. However, for the first time in decades, the advances in computing technology are now threatened, because while transistor density is projected to increase with Moores Law, the energy efficiency of silicon is not keeping pace. HPC system architectures are expected to change dramatically in the next decade as power and cooling constraints are limiting increases in microprocessor clock speeds. Consequently computer companies are dramatically increasing on-chip parallelism to improve performance. The traditional doubling of clock speeds every 18-24 months is being replaced by a doubling of cores or other parallelism mechanisms. During the next decade the amount of parallelism on a single microprocessor will rival the number of nodes in the first massively parallel supercomputers that were built in the 1980s. Applications and algorithms will need to change and adapt as node architectures evolve. Future generation consumer electronics devices, which are limited by battery life, would not support any new features that rely on increased computing. The iPhone, Google, simulation-based medical procedures, and our understanding of climate change would not have been possible without these increases in computing performance. If computing performance stalls at todays levels the Information Technology industry will shift from a growth industry to a replacement industry, and future societal impacts of computing will be limited to what can be done on todays machines.

The next major milestone in High Performance Computing, an exascale system, would be impractical at hundreds megawatts. Computing technology is rapidly approaching a *power wall*, which will limit future growth in computing capability. Overcoming this *power wall* will require fundamental advances in component technologies using advanced nanomaterials to enable transformational changes in the power, performance, and programmability of fu-

ture computing devices. The path towards realizing next-generation petascale and exascale computing is increasingly dependent on building supercomputers with unprecedented numbers of processors. To prevent the interconnect from dominating the overall cost of these ultra-scale systems, there is a critical need for scalable interconnects that capture the communication requirements of ultrascale applications. Future computing systems must rely on development of interconnect topologies that efficiently support the underlying applications' communication characteristics. It is therefore essential to understand high-end application communication characteristics across a broad spectrum of computational methods, and utilize that insight to tailor interconnect designs to the specific requirements of the underlying codes.

As scientific computing matures, the demands for computational resources are growing at a rapid rate. It is estimated that by the end of this decade, numerous grand-challenge applications will have computational requirements that are at least two orders of magnitude larger than current levels [34, 48, 54]. However, as the pace of processor clock rate improvements continues to slow [2], the path towards realizing ultrascale computing is increasingly dependent on scaling up the number of processors to unprecedented levels. To prevent the interconnect architecture from dominating the overall cost of such systems, there is a critical need to effectively build and utilize network topology solutions with costs that scale linearly with system size.

Among the many issues affecting scalability of future system is the scaling of high bandwidth interconnects, designed for both on-chip and off-chip communication to memory and to other computational devices. Future computing systems, whether based on traditional circuits or the proposal nanotechnology devices will rely on parallelism to keep power budgets manageable while increasing performance. At the macro-scale, interconnects must keep costs, efficiency, and power consumption under control in the face of exponential growth in system parallelism. At the chip-level, nanophotonic interconnects can exploit extremely high-capacity and low-power interconnection supported by inherent parallelism of optics. This document will describe requirements for future manycore processors with massively parallel nanophotonic and nanoelectronic interconnects for a new generation of logic elements. Contemporary computing systems do not have sufficient memory and communication performance to balance their computation rates primarily due to limited I/O throughput of the off-chip electrical links. Optics provide ultra-high throughput, minimal access latencies, and low power dissipation that remains independent of capacity and distance that would enable I/O bandwidth to be uniformly plentiful across a system. Multi-wavelength operation can bring massive parallelism to the computing system to enable construction of systems that attack grand-challenge scientific problems such as the study of global climate change, and support continued growth in the data processing capabilities of commercial datacenters, which are estimated to double every 18 months. Massively parallel nanoelectronic interconnection offers low-power short distance interconnection between many cores.

This thesis will present a deep analysis of the requirements of ultrascale applications in order to better understand the demands on hardware at the system scale and chip scale in the

face of massive growth in parallelism. The application requirements for system-scale will be distilled into requirements for multi-tiered networks using the high-level abstraction of fit-trees. The next chapter will introduce the Hybrid Flexibly ASsignable Topology (HFAST) approach to implementing a dynamically reconfigurable topology to enable a physical realization of the fit-tree method. Then this thesis revisits the application communication pattern analysis with an eye towards understanding chip-scale interconnect requirements for Networks on Chip (NoCs) that must offer scalable performance for interconnecting hundreds or even thousands of cores over the next decade. Finally, the performance study compares a number of different NoC topologies that include both electrical packet switch and optical circuit switch components.

1.1 Contributions of this Work

High performance computing (HPC) systems implementing *fully-connected networks* (FCNs) such as fat-trees and crossbars have proven popular due to their excellent bisection bandwidth and ease of application mapping for arbitrary communication topologies. However, as supercomputing systems move towards tens or even hundreds of thousands of processors, FCNs quickly become infeasibly expensive. This is true for extreme-scale systems that are anticipated to contain hundreds of thousands of nodes as well as individual manycore chips containing hundreds or even thousands of processing elements. These trends have renewed interest in networks with a lower topological degree, such as mesh and torus interconnects (like those used in the IBM BlueGene and Cray XT series), whose costs rise linearly with system scale. Indeed, the number of systems using lower degree interconnects such as the BG/L and Cray Torus interconnects has increased from 6 systems in the November 2004 list to 28 systems in the more recent Top500 list of June 2007 [58] . However, it is unclear what portion of scientific computations have communication patterns that can be efficiently embedded onto these types of networks.

The quality of an interconnect should be measured by how well it captures the communication requirements of a target application, as opposed to theoretical metrics such as diameter and bisection bandwidth, since such metrics depend only on the interconnect topology, ignoring the communication topologies of target applications. For this proposed approach, it is essential to understand scientific application communication requirements across a broad spectrum of computational methods. Once this information is derived, the interconnect design can be tailored for the specific communication requirements of the underlying applications in terms of cost and performance effectiveness, while exploring how new technologies can be adopted for breakthroughs in interconnect solutions.

This work demonstrates a new application-driven approach to interconnect design and presents several unique contributions. Chapter 3 examines the communication requirements of high-end applications that demand the largest scale computing resources. The selected applications represent a broad array of scientific domains and computational methods, which

include: finite-difference, lattice-Boltzmann, particle-in-cell, sparse linear algebra, particle mesh ewald, and FFT-based solvers. The IPM (Integrated Performance Monitoring) profiling layer was used to gather detailed messaging statistics with minimal impact to code performance. Chapter 4 introduces the concept of a "fit tree" to understand how adaptive interconnect topologies can be constructed dynamically to efficiently support the underlying applications' communication characteristics. Chapter 4 also presents a novel approach to using optical circuit switches to rewire multitiered communication networks to optimizing interconnect wiring topologies called Hybrid Flexibly Assignable Switch Topology (HFAST). HFAST allows the implementation of interconnect topologies that are specifically tailored to application requirements, via the proposed fit-tree approach or other mapping strategies. Finally, it will present data demonstrating that the hybrid interconnect design is able to meet application communication requirements using only a fraction of the resources required by conventional fat-tree or CLOS interconnects. Chapter 5 applies the same design principles to understanding interconnect requirements for future Network-on-Chip designs that can scale to support manycore chips containing hundreds of processing elements. The chapter compares a number of competing 2D planar on-chip interconnect topologies that include both electronic and silicon-photonic components.

These studies show that the diverse communication requirements of different applications force interconnect designers toward a conservative approach that over-provisions resources to avoid congestion across all possible application classes. This limitation can be overcome within a dynamically reconfigurable interconnect infrastructure that relies on optical circuit switching to optimize the wiring topology of the system both at chip-scale and system-scale. For system-scale interconnects, the rewiring minimizes expensive optical-electrical-optical and energy-inefficient transitions. For NoC designs, the optical circuit switching bypasses expensive buffering for electronic packet routers as well as costly re-amplification of the electrical signal for long paths on chip. The key to the overall approach is using circuit switching to dynamically provision high-bandwidth network pathways (on-chip and off-chip) to match sustained application communication flows that are identified by runtime communication statistics.

Overall results lead to a promising approach for addressing the interconnect requirements of future exascale computing systems. Although the three research thrusts — HPC communication characterization, system-scale interconnect design, and NoC design — work closely in concert, each of these components could also be considered as independent contributions, which advance the state-of-the art in their respective areas.

Chapter 2

Background

Over the past forty years, progress in supercomputing has consistently benefitted from improvements in integrated circuit scaling according to Moores law, which has yielded exponential improvements in peak system-level floating-point performance. For as long as the Top500 list [58] has been in existence, HPC system measured LINPACK performance has consistently increased by a factor of 1000x every 11 years. Moore's law has supplied 100x of that improvement, but the extra 10x has been delivered through innovations that are specific to the leading-edge HPC architectural space. In particular, the burden of this extra 10x scaling has fallen largely to advanced interconnect architectures that enable large arrays of commodity components function together as an integrated, scalable HPC system.

However, changes in device physics threaten further sustained progress of extreme-scale HPC systems. For the first time in decades, the advances in computing technology are now threatened, because while transistor density on silicon is projected to increase with Moores Law, the energy efficiency of silicon is not. Power has rapidly become the leading design constraint for future HPC systems systems. Numerous studies conducted by DOE-ASCR [55], DOE-NNSA [56] and DARPA [20] have concluded that given these new constraints, the current approach to designing leading-edge HPC systems is unsustainable leading to machines consuming upwards of 60 megawatts. New approaches will not emerge from evolutionary changes in processor speed and scale from todays petascale systems, but will require fundamental breakthroughs in hardware technology, programming models, algorithms, and software at both the system and application level. The Top500 list predicts emergence of an Exaflop-scale computing system by 2019. Given these daunting technology challenges, continuation of historical growth rates in HPC are by no means certain.

15 years ago at the advent of the Top500 list, FLOPs were the most expensive component in an HPC system design. However, systems today and in the future are increasingly bound by their communications infrastructure and the power dissipation associated with high-bandwidth information exchange across the vastly growing number of computing nodes. In fact, the limitations on power dissipation imposed by packaging constraints have become

so paramount that performance metrics are now typically measured per unit power. Optical interconnect technology is inherently different from electronics in its unique capability to route multi-wavelength signals transparently and propagate over long distance spans without need for regeneration. Thus, fundamentally, optical interconnection networks seamlessly cross over traditional electronic system boundaries from on-chip to off-chip and beyond. In this context, optical interconnection networks offer a fundamentally disruptive technology solution and the possibility of creating an extraordinarily energy efficient communications infrastructure that seamlessly spans across traditional electronic boundaries and can deliver uniformly high-bandwidths over the entire system. Recent advances in 3D Integration CMOS technology, the possibility for realizing hybrid photonic-electronic networks-on-chip. Nanoscale silicon photonics devices offer the possibility of creating highly power efficient platforms that break through current barriers to achieving globally high system bandwidth on a fixed power budget. However, making use of those capabilities requires a fundamental reconsideration of interconnect architecture at both chip-scale and system scale. This research on advanced interconnect technology addresses the technology challenges of interconnect design that are essential for maintaining the historical 1000x growth rate in HPC performance within the next decade.

This chapter will set the context for the advanced interconnect designs by diving into details of historical sources of performance improvement, which are ending. It will then cover new technology options that can carry us through the next decade.

2.1 Emerging Technology Trends

One important discontinuity that has developed in system architecture is motivated by changes in device physics below the 90nm scale. The changes at the chip level create a cascade of design choices that affect every aspect of system design and result in some of the most daunting challenges for software design on future trans-petaflop systems. The most important result of these changes is that we can no longer depend on exponential scaling of the serial performance of microprocessors to derive future performance improvements, leading to a explosive growth in chip-level parallelism. Furthermore, the energy cost of moving data is on track to exceed the cost of computation. The industry is set to move towards massive parallelism at all levels, with power-constrained data movement. All of these trends put increased strain on interconnect design both at the chip-level and system level to maintain performance while fitting into a constrained budgets for power and component cost.

This chapter will first walk through the factors behind the clock-frequency stall in microprocessor designs, and then move on to characterize the cost of data movement in response to future scaling trends.

2.1.1 The End of Clock Frequency Scaling

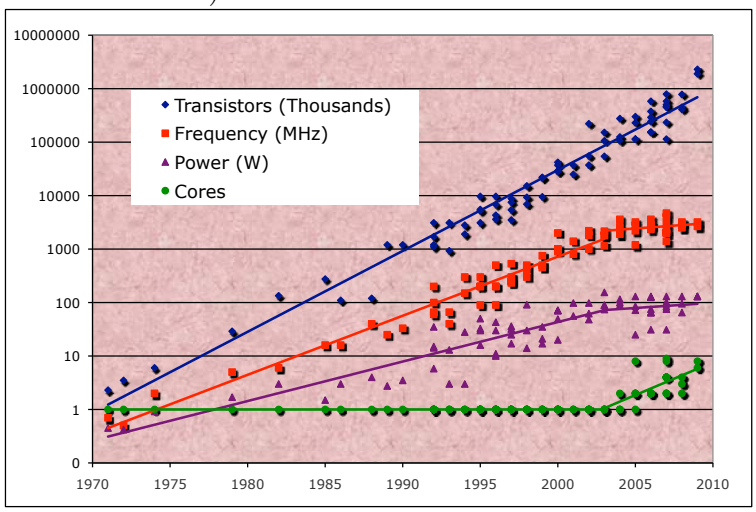
Figure 2.1 shows that Moores law, which states that you can integrate twice as many components onto an integrated circuit every 18 months at fixed cost, still holds. However, the traditional sources of performance improvements such as instruction level parallelism (ILP) and clock frequency scaling have been flattening since 2003. Figure 2 shows the improvements in processor performance as measured by the SPEC benchmark over the period from 1975 to present. Since 1986 performance has improved by 52 percent per year with remarkable consistency. During that period, as process geometries scaled according to Moore's law, the active capacitance of circuits scaled down accordingly. This effect is referred to Dennard scaling after the scaling theory advanced by Robert Dennard of IBM Research in 1974 [49]. Dennard first described his theory at a time when transistors gate lengths were on the order of 1mm, and amazingly they have continued to hold true all the way down to the 1 micron chip lithography scale. The *Device Dimension* in Table 2.1 refers to feature scaling with chip lithography improvements, which have improved by a factor of 2x every 18 months according to Moore's law. Dennard's scaling rules provided guidelines for the industry to reliably scale down logic devices while maintaining a constant power density (which is designed to stay constant as shown in Table 2.1).

Table 2.1: Technology principles according to Dennard scaling (from Dennard's original paper [49]).

Device or Circuit Parameter	Scaling Factor
Device Dimension $L \times W = Area$	$1/k$
Doping Concentration Na	k
Voltage V	$1/k$
Current I	$1/k$
Capacitance $Area/t$	$1/k$
Delay time/circuit VC/I	$1/k$
Power dissipation/circuit VI	$1/k^2$
Power Density VI/A	1

As a consequence of Dennard scaling, supply voltages could be kept constant or even dropped modestly in order to allow manufacturers to increase clock-speeds. This application of the Dennard scaling parameters, known as *constant electric field frequency scaling* [6] fed the relentless increases in CPU clock-rates over the past decade and a half. However, below the 90nm scale for silicon lithography, this technique began to hit its limits because manufacturers could no longer scale down voltage supplies at historical rates as shown in Figure 2.3(a). Consequently, the static power dissipation from leakage current began to surpass dynamic power dissipation from circuit switching as shown if Figure 2.3(b). With the end of Dennard scaling, power density has now become the dominant constraint in the design of new processing elements, and ultimately limits clock-frequency growth for future microprocessors. The

Figure 2.1: This graph shows that Moores law is alive and well, but the traditional sources of performance improvements (ILP and clock frequencies) have all been flattening. The performance of processors as measured by SpecINT has grown 52% per year with remarkable consistency, but improvements tapered off around 2003 due to the end of Dennard scaling rules. (*This figure is based on original data from Kunle Olukotun and Herb Sutter, but updated with more recent data.*)

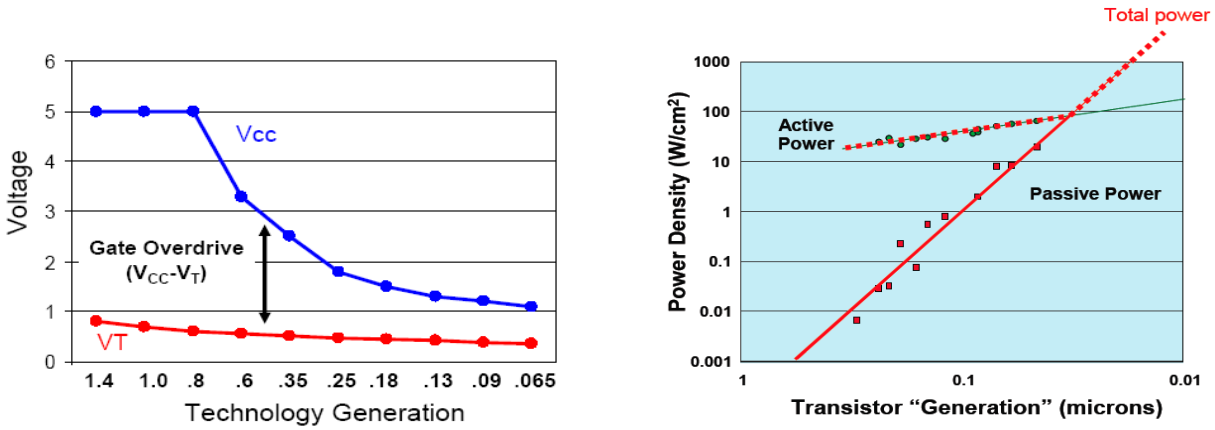


direct result of power constraints has been a stall in clock frequency that is reflected in the flattening of the performance growth rates starting in 2002 as shown in Figure 2.1. In 2006, individual processor cores are nearly a factor of three slower than if progress had continued at the historical rate of the preceding decade. Other approaches for extracting more performance such as Instruction Level Parallelism (ILP) and out-of-order instruction processing have also delivered diminishing returns as shown in Figure 2.1. Having exhausted other well-understood avenues to extract more performance from a uniprocessor, the mainstream microprocessor industry has responded by halting further improvements in clock frequency and increasing the number of cores on the chip. Patterson and Hennessy [26] estimate the number of cores per chip is likely to double every 18-24 months henceforth.

2.1.2 Move to Massive Parallelism

The stall in clock frequencies leaves few options for maintaining historical exponential trends in clock frequency performance. Shifting from exponential increases in clock frequency to exponential increases in processor cores is a relatively straightforward response, but the desire for more elegant solutions has also reinvigorated study of more radical alternative approaches to computing such as Field Programmable Gate Arrays (FPGAs), Graphics Processing Units (GPU), and even dataflow-like tiled array architectures such as TRIPS [9]. The principle

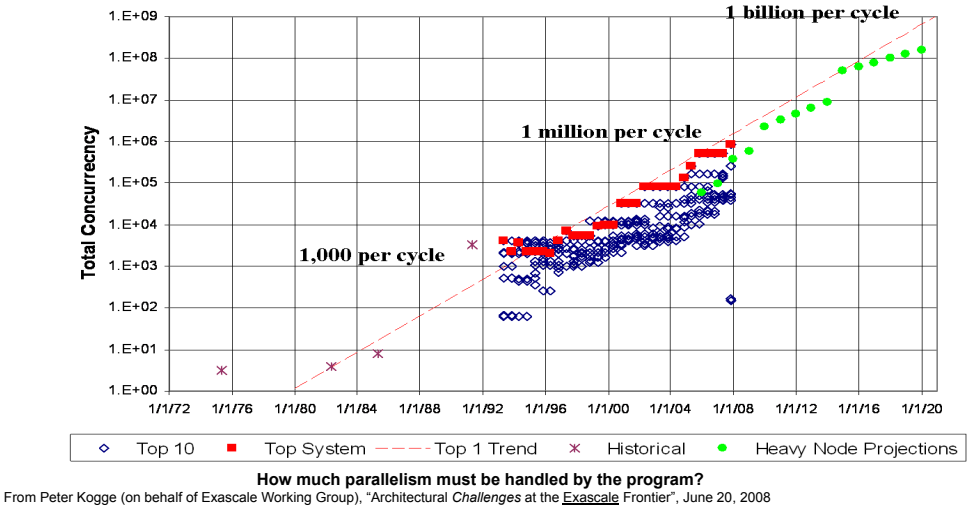
Figure 2.2: Manufacturers are no longer able to scale-down the threshold voltage V_T because of leakage current, they can no longer scale down the supply voltage V_{cc} as aggressively. Consequently, the industry has departed from Dennard's scaling formula to maintain constant power density. (*Source: P. Packan, Intel, 2007 IEDM Short Course*)



(a) Suppply Voltage Trends

(b) Static vs. Dynamic Power

Figure 2.3: With flat clock rates, all performance improvements must come from parallelism. Exascale computing systems are anticipated to contain millions or even billions of FPUs (*source DARPA Exascale Report [20]*)



impediment to adapting a more radical approach to hardware architecture is that we know even less about how to program efficiently such devices for diverse applications than we

do parallel machines composed of multiple CPU cores. Until a clearer alternative emerges, multicore continues to be the most likely approach to continued performance improvements in the face of a fixed power budget.

The new industry buzzword multicore captures the plan of doubling the number of standard cores per die with every semiconductor process generation starting with a single processor. Multicore will obviously help multiprogrammed workloads, which contain a mix of independent sequential tasks, and prevents further degradation of individual task performance. But switching from sequential to modestly parallel computing will make programming much more difficult without rewarding this greater effort with a dramatic improvement in power-performance. The alternative approach moving forward is to adopt the manycore trajectory, which employs simpler cores running at modestly lower clock frequencies. Rather than progressing from 2 to 4, to 8 cores with the multicore approach, a manycore design would start with hundreds of cores and progress geometrically to thousands of cores over time. Figure 2.4 shows that moving to a simpler core design results in modestly lower clock frequencies, but has enormous benefits in power consumption and chip surface area. Even if you presume that the simpler core will offer only 1/3 the computational efficiency of the more complex out-of-order cores, a manycore design would still be an order of magnitude more power and area efficient in terms of sustained performance.

Figure 2.4: The diagram shows the relative size and peak power dissipation of different CPU core architectures at 65nm chip lithography scale. Simpler processor cores require far less surface area and power with only a modest drop in clock frequency. 5-9 stage pipelines have been shown to be optimal for energy per operation design point in [2]. Even if measured by sustained performance on applications, the power efficiency and performance per unit area is significantly better when using the simpler cores. (*source: sandpile.org*)

The manycore approach has been adopted very rapidly in the consumer electronics and embedded world, and will likely emerge in the HPC space in the press towards exascale computing. Parallelism with concurrencies that have formerly been associated with HPC applications are already emerging in mainstream embedded applications. The Cisco Metro chip in new CRS-1 router contains 188 general-purpose Tensilica cores, and has supplanted Ciscos

previous approach of employing custom Application Specific Integrated Circuits (ASICs) for the same purpose [18]. Surprisingly, the performance and energy efficiency of the Metro are competitive with their full custom logic design. The next-generation of smart phones from both Apple and the Android platform are anticipated to contain multicore embedded processor designs in future systems.

Another early adopter of the manycore design paradigm are GPUs. The NVidia Fermi (CUDA) Graphical Processing Unit (GPU) replaces the semi-custom pipelines of previous generation GPUs with hundreds of general-purpose processing elements called Streaming Multiprocessors (SM's) that are arranged into groups called *warps*. Fermi, in particular, heralds the convergence of manycore with mainstream computing applications. Whereas traditional General Purpose GPUs (GPGPUs) have a remarkably obtuse programming model involving drawing an image of your data to the frame-buffer (the screen), Fermis more general purpose cores can be programmed using more conventional CUDA code and will soon support IEEE standard double-precision arithmetic. Both Intel and AMD roadmaps indicate that tighter integration between GPUs and CPUs is the likely path toward introducing manycore processing to mainstream consumer applications on desktop and laptop computers.

Each of these cases push towards chip architectures containing hundreds or even thousands of computational elements within the next decade. The interconnection network requirements on chip will be on par with the scalable networks required for today's supercomputing systems even for single-chip systems. However, the challenges of getting to an exaflop will push the limits on both chip-scale and system scale interconnection networks.

2.2 Emerging Challenges for Interconnects

In an ideal world, system implementations would never subject applications to any performance constraints. However, power and cost of different components of an HPC system force system architects to consider difficult trade-offs that balance the actual cost of system components against their effect on application performance. For example, if doubling floating-point execution rate nets a 10% gain in overall application performance, but only increases system costs by 5%, then it is a net benefit despite the counter-intuitive effect on system balance. It is important to have an open dialog to fully understand the cost impacts of key design choices so that they can be evaluated against their benefit to the application space.

2.2.1 The Cost of Power

Even with the least expensive power available in the US, the cost of electricity to power supercomputing systems is a substantial part of the Total Cost of Ownership (TCO). When burdened with cooling and power distribution overheads, even the least expensive power in

the U.S. (< 5cents/KWH) ultimately costs $1M per Megawatt per year to operate a system. To keep the TCO manageable DOEs Exascale Initiative Steering Committee adopted 20MW as the upper limit for a reasonable system design [20,55]. This limit is movable, but at great cost and design risk.

2.2.2 The Cost of a FLOP

Floating point used to be the most costly component of a system both in terms of design cost and power. However, today, FPUs consume a very small fraction of the area of a modern chip design and a much smaller fraction of the power consumption. On modern systems, a double-precision FMA (fused multiply add) consumes 100 picoJoules. By contrast, reading the double-precision operands from DRAM costs about 2000 pJ. By 2018 floating-point operations will consume about 10.6pJ/op on 11nm lithography technology[3], and the cost of reading from DRAM will only improve modestly to 1000pJ unless more energy-efficient memory technology is developed.

With these figures of merit, it would only consume 100W to put 10 Teraflops on a chip, which is easily achievable. However, it would require 2000W of power required to supply memory bandwidth to those floating-point units at a modest memory bandwidth to floating-point ratio of 0.2. The consequence is that we can engineer far more floating-point capability onto a chip than can reasonably be used by an application. Engineering FLOPs is not a design constraint; data movement presents the most daunting engineering and computer architecture challenge.

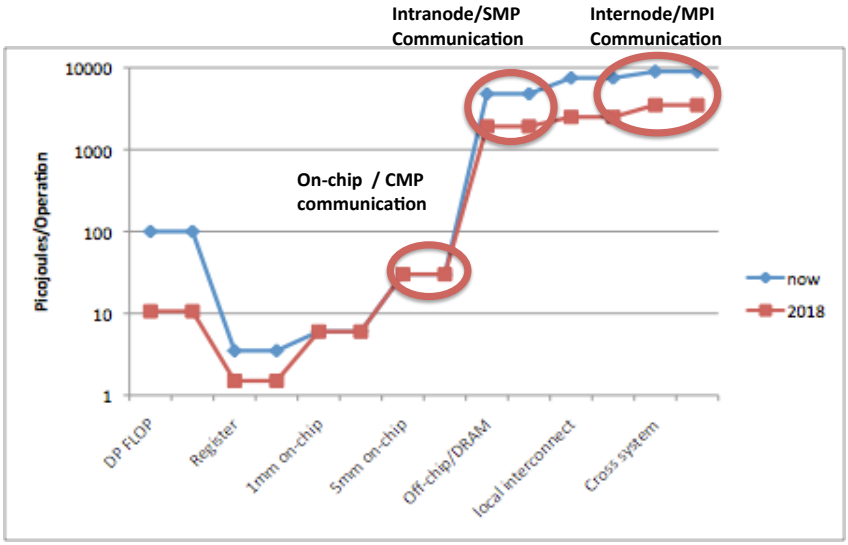
2.2.3 Cost of Moving Data

Memory interfaces and communication links on modern computing systems are currently dominated by electrical/copper technology. However, wires are rapidly being subsumed by optical technology because of the limits of bit rate scaling as we shrink wires length scales as observed by David A. B. Miller of Stanford [40, 41]. Miller observes that for a conventional electrical line (without repeaters or equalization) can be modeled as a simple RC circuit by virtue of the simplified Telegraphers equation for lossy transmission line. The wire must be charged and discharged at a rate governed by the RC time constant, which is given by Equation 2.1 where R_l is the resistance of the wire, C_l is the capacitance and l is the length of the wire. As the wire length increases, the risetime (given by the RC time constant) increases by the square of the length thereby reducing the bit-rate.

$$Risetime \approx R_l C_l length^2 \tag{2.1}$$

Miller observes that if you shrink the wire proportionally in all dimensions by a factor of s, the resistance (R_l) increases proportionally to the reduced wire aspect ratio, which reduces

Figure 2.5: With new scaling rules and massive growth in parallelism, data locality is increasingly important. This diagram shows the cost of a double-precision multiply-add at different levels of the memory hierarchy (from the cost of performing the flop to the movement of data operands from registers, different distances on-chip, and distances off-chip.)



by a factor of s^2 , but capacitance (C_l) remains the same. The consequence is that for constant voltage, the bit-rate carrying capacity of an RC line scales proportional to $B \approx \frac{A}{l^2}$, where B is the bandwidth of the wire and A is the cross-sectional area of the wire and l^2 is the length of the wire. The consequence of this observation is that natural bit rate capacity of the wire depends on the aspect ratio of the line, which is the ratio of the length to the cross-sectional area for a constant input voltage and does not improve as we shrink the wires down with smaller lithographic processes. We can push to a higher bitrate by increasing the drive voltage to the wire, but this also increases power consumption. These effects are summarized in Equation 2.3, which assumes a simple RC model of the wire and no re-amplification *(long-haul wires on-chip are normally re-amplified at regular intervals to maintain a linear power profile as a function of length, but at a cost of more power consumption).*

$$B \approx \frac{A}{l^2} \quad (for\ fixed\ voltage\ swing) \tag{2.2}$$

$$Power \approx B \times \frac{l^2}{A} \tag{2.3}$$

This has the following consequences to system design [3, 28];

- Power consumed increases proportionally to the bit-rate, so as we move to ultra-high-

bandwidth links, the power requirements will become an increasing concern

- Power consumption is highly distance-dependent (quadratically with wire length without re-amplification), so bandwidth is likely to become increasingly localized as power becomes a more difficult problem.
- Improvements in chip lithography (making smaller wires) will not improve the energy efficiency or data carrying capacity of electrical wires.

In contrast, optical technology does not have significant distance-dependent energy consumption. It costs nearly the same amount of energy to transmit an optical signal 1 cm as it does to transmit it to the other end of a room. Also, signaling rate does not strongly affect the energy required for optical data transmission. Rather, the fixed cost of the laser package for optical systems and the absorption of light to receive a signal are the dominant power costs for optical solutions.

As the cost and complexity of moving data over copper will become more difficult over time, the cross-over point where optical technology becomes more cost-effective than electrical signaling has been edging closer to the board and chip package at a steady pace for the past two decades. Contemporary short-distance copper links consume about 10-20 pJ/bit, but could be improved to 2pJ/bit for short-haul 1 cm length links by 2018. However, the efficiency and/or data carrying capacity of the copper links will fall off rapidly with distance (as per equation 2) that may force a movement to optical links. Contemporary optical links consume about 30-60pJ/bit, but solutions that consume as little as 2.5pJ/bit have been demonstrated in the lab [28, 64]. In the 2018 timeframe optical links are likely to operate at 10pJ/bit efficiency. Moreover, silicon photonics offers the promise of breaking through the limited bandwidth and packaging constraints of organic carriers using electrical pins.

Another serious barrier to future performance growth is cost of signals that go off-chip as we rapidly approach pin-limited bandwidth. Due to the skin effect [24], and overheads of more complex signal equalization, it is estimated that 10-15GHz is likely the maximum feasible signaling rate for off-chip differential links that are 1-2cm in length. A chip with 4000 pins would be a very aggressive, but feasible design point for 2018. If you consider that half of those pins (2000) are power and ground, while the remaining 2000 pins are differential pairs, then the maximum feasible off-chip bandwidth would be 1000 $\times$ 10GHz, which comes to approximately 1 Terabyte/second (10 Terabits/sec with 8/10 encoding). Breaking through this 1 TB/s barrier would require either more expensive, exotic packaging technology (ceramics rather than organic packages), or migration to on-chip optics, such as silicon-photonic ring-resonator technology [21, 25] that will be covered in more detail in Chapter 5 of this thesis.

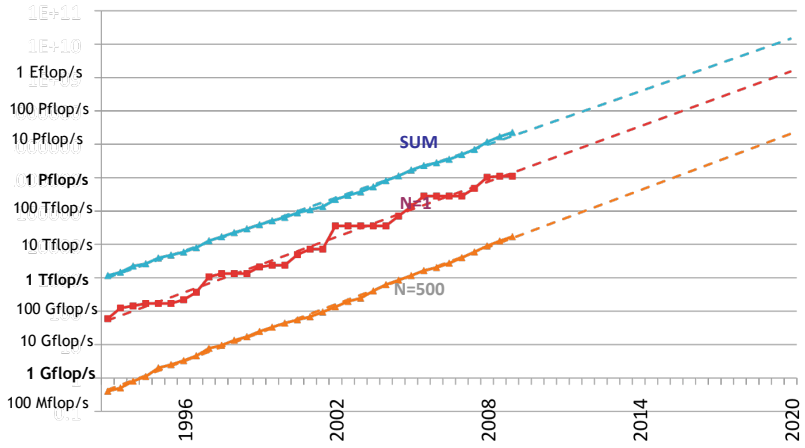
Without major breakthroughs in packaging technology or photonics, it will not be feasible to support globally flat bandwidth across a system. Algorithms, system software, and applications will need to aware of data locality. The programming environment must enable

algorithm designers to express and control data locality more carefully. The system must have sufficient information and control to make decisions that maximally exploit information about communication topology and locality. Flat models of parallelism (e.g. flat MPI or shared memory/PRAM models) will not map well to future node architectures.

2.3 Exascale Computing Challenges

Over the past thirty years, progress in supercomputing has consistently benefitted from improvements in integrated circuit scaling according to Moores law, which has yielded exponential improvements in peak system-level floating-point performance. For as long as the Top500 list has been in existence, HPC system measured LINPACK performance has consistently increased by a factor of 1000x every 11 years as shown in Figure 2.6. Moore's law has supplied 100x of that improvement, but the extra 10x has been delivered through innovations that are specific to the leading-edge HPC architectural space.

Figure 2.6: Historically leading-edge HPC system performance on the Top500 list has improved by a factor of 1000x every 11 years with remarkable consistency. The *red* line in the center is the LINPACK performance of the top ranked system in the Top500 while the orange line on the bottom is the performance of the last system on the Top500 list. Finally, the blue line on the top is the sum of the performance of all of the systems on the Top500 list. The consistency of the historical performance improvements at all scales is truly remarkable. *(source: Top500.org)* [58]



As a result of the stall in clock frequencies, future HPC architectures will be forced to exponential scaling of system parallelism to maintain future improvements in system performance. According to DARPA projections shown in Figure 2.3, leading-edge systems are expected to contain on the order of 1 billion computational elements by the time exascale

computing systems debut in 2019. Therefore, new algorithms and programming models will need to stay ahead of a wave of exponentially increasing system concurrency a tsunami of parallelism shown in Figure 2.3. The primary area of growth in parallelism is explicit parallelism on-chip. Whereas the number of nodes in an exascale system is expected to grow by a factor of 10x over the next decade, the parallelism on-chip is expected to grow by a factor of 100x. This requires reconsideration of on-chip organization of CPU cores, and the semantics of inter-processor communication. With parallelism is exploding both at chip level and at system level, data movement at both at chip-level and system level, have moved from a peripheral concern to a central design challenge for future systems of all scales. For billion processor systems, the interaction between the macro-scale (systemwide networks) and micro-scale (on-chip NoCs) is increasingly important. Moreover, the move to silicon photonic technology for on-chip networks stands to bridge the gap between on chip communication and off-chip communication design as optical technology moves on-chip, to build interconnects that offer sustained global bandwidth necessary to maintain efficient computational performance.

2.3.1 Ultrascale Interconnects

Future computing systems, whether based on traditional circuits or the proposal nanotechnology devices will rely on parallelism to keep power budgets manageable while increasing performance. The correct design point identified above will entail massively parallel chip designs that require interconnection networks with millions or even billions of endpoints as evidenced by the scaling in Figure 2.3. With current technology scaling trends, computing systems will not have the memory and communication performance to balance their computation rates. However, optical communications opens the possibility of ultra-high throughput, minimal access latencies, and low power dissipation that remains independent of capacity and distance that would enable I/O bandwidth to be uniformly plentiful across a system. Multi-wavelength operation can bring massive parallelism to the computing system to enable construction of systems that attack grand-challenge scientific problems such as the study of global climate change, and support continued growth in the data processing capabilities of commercial datacenters, which are estimated to double every 18 months. Massively parallel silicon photonic interconnection fabrics offer a low-power short distance interconnection between cores within a chip for massively parallel *manycore* chip designs.

The primary challenge of optical networks is keeping them all-optical. Optical-Electronc-Optical (OEO) conversions dominate the cost and power consumed by these networks, and undercut the energy and performance advantages that might be derived from moving to optical technology. Modern packet-switched networks with optical links perform all of their switching functionality in the electronic domain because packet-switching requires some form of temporary storage to buffer up parts of the packet to provide sufficient time for logic to make a routing decision. For example, in a typical source routed network, you must read the address bits at the start of the packet to determine which direction to send the packet.

However, the development of cost-effective optical buffering technology has been elusive. IBM and Corning spent five years on the OSMOSIS project to create an all-optical packet switch. The switching component was done entirely in the optical domain, but packet header still had to be converted into an electrical signal because the switching logic could not be fully implemented in the optical domain [1]. Therefore, buffering and routing decisions for optical networks as we know them, must still be done using electronics, which requires expensive optical-electrical conversions.

Circuit-switching offers an approach to interconnection networks that do not contain any switches, and therefore are amenable to all-optical networks. Technologies such as Micro Electro-Mechanical mirror Systems (MEMS), similar to the technology used for Digital Light Projector (DLP) technology, and silicon photonic switches, offer technologies that can be used to build energy-efficient interconnects. However, circuit switches are oblivious to packet boundaries, so it requires a different way of thinking about interconnect architecture. A network built exclusively of unbuffered circuit switches can only realize their efficiency for sustained traffic flows where they change state infrequently. However, we can derive huge The challenge is to develop an architecture that minimizes the number of OEO conversions employed to make an end-to-end connection.

2.4 Approach

This document describes the design of an interconnect architecture that fits within these constraints, and delivers scalable performance for future massively parallel chip and system architectures. Chapter 3 develops a detailed understanding of the application communication requirements in detail to guide architectural decisions for the interconnect implementation. Chapter 4 describes an efficient strategy for mapping communication to a hybrid optical network in a manner to that minimizes OEO conversions at system level. Finally, Chapter 5 applies this strategy to intra-chip silicon photonic networks, which also use a circuit-switched approach for the optical plane. Chapter 6 concludes with an analysis of the energy efficiency and performance benefits of these technologies, and implications for the future of scalable computing systems.

Chapter 3

Application Requirements

In order to quantify HPC interconnect requirements and study efficient implementation approaches, one must first develop an understanding of the communication characteristics for realistic scientific applications. Several studies have observed that many applications display communication topology requirements that are far less than the total connectivity provided by fully-connected networks. For instance, the application study by Vetter et al. [62, 63] indicates that the applications that scale most efficiently to large numbers of processors tend to depend on point-to-point communication patterns where each processor's average *topological degree of communication* (TDC) is 3–7 distinct destinations, or neighbors. This provides strong evidence that many application communication topologies exercise a small fraction of the resources provided by fully-connected networks.

This section expands on previous studies by exploring detailed communication profiles across a broad set of representative parallel algorithms. The IPM profiling layer is used to quantify the type and frequency of application-issued MPI calls, as well as identify the buffer sizes utilized for both point-to-point and collective communications. Finally, the communication topology of each application is analyzed to determine the average and maximum TDC for bandwidth-limited messaging.

3.1 Data Collection Tools and Methods

3.1.1 IPM: Low-overhead MPI profiling

Integrated Performance Monitoring (IPM) [31] tool was used to profile the communication characteristics of the scientific applications in this study — an application profiling layer that allows non-invasive collection of the communication characteristics of these codes as they run in a production environment. IPM brings together multiple sources of performance metrics into a single profile that characterizes the overall performance and resource usage of

the application. It maintains low overhead by using a unique hashing approach that allows a fixed memory footprint and minimal CPU usage. IPM is open source, relies on portable software technologies, and is scalable to thousands of tasks.

The core idea of IPM is to provide an easy to use and scalable means of collecting performance data from HPC codes in a production environment. On most HPC platforms there is no parallel aware layer for collection aggregation and reporting of communication and HPM statistics. IPM adds such a layer providing job level performance profiles with little to no effort required by the end user running their application. For instance collecting an IPM profile only requires the user to set an environment variable in their batch script in order to collect a IPM profile of the application exactly as it is run in a production setting. This approach avoids perturbations both to user and code caused by profiling a code within a application performance tool or otherwise non production environment. IPM is scalable to thousands of tasks with extremely low overhead. This overhead is quantified later in the paper when presenting application results.

Since the evaluated workloads expresses distributed memory parallelism via MPI (Message Passing Interface), the IPM implementation focuses on name-shifted profiling interface to MPI. The use of the profiling interface to MPI is of widely recognized value in profiling MPI codes [47, 62]. The name-shifted or PMPI interface allows each MPI call to be wrapped by profiling code that collects communication performance information.

IPM collects a wide variety of communication information using a very low-overhead hashing technique, which allows us to non-invasively instrument full-scale application codes without dramatically affecting their performance. The data collection for this work principally utilize information that encodes the number and timing of each MPI call. The communication information is gathered on each task about each MPI call with a unique set of arguments. Arguments to MPI calls contain message buffer size, as well as source and destination information. In some cases IPM also tracks information from the `MPI_Status` structure. For instance, in the case of `MPI_Send`, IPM keeps track of each unique buffer size and destination, the number of such calls, as well as the total, minimum and maximum runtimes to complete the call. IPM also allows code regions to be defined, enabling separation of application initialization from steady state computation and communication patterns, because the analysis is primarily concerned with the communication topology for the application in its post-initialization steady state. Experiments were run on a variety of Department of Energy supercomputing systems; the data collected depends on the concurrency, application code, and input— no machine-dependent characteristics are collected or analyzed in this study.

3.1.2 Message Size Thresholding

This study focuses primarily on network topology optiimizations to reduce contention for bandwidth-bound messages. Therefore, the analysis of the TDC for this application suite requires a criteria for choosing the thresholding size for messages that are bandwidth limited.

Table 3.1: Overview of scientific applications evaluated.

Name	Lines	Discipline	Problem and Method
BBeam3D [46]	28,000	Particle Physics	Vlasov-Poisson using Particle in Cell and FFT
Cactus [10]	84,000	Astrophysics	Einstein's Theory of GR using PDE solve on Structured Grid
GTC [39]	5,000	Magnetic Fusion	Vlasov-Poisson using Particle in Cell Method
LBCFD [43]	3,000	Fluid Dynamics	Navier-Stokes using Lattice Boltzmann Method
MADbench [7]	5,000	Cosmology	CMB Analysis using Newton-Raphson on Dense Matrix
PARATEC [11]	50,000	Materials Science	Electronic Structure using FFT and Dense Linear Algebra
PMEMD [27]	37,000	Life Sciences	Molecular Dynamics using Particle Mesh Ewald
SuperLU [38]	42,000	Linear Algebra	Sparse Solve using LU Decomposition

Otherwise, the analysis may mistakenly presume that a given application has a high TDC even if trivially small (latency-bound) messages are sent to majority of its neighbors.

The product of the message bandwidth and the delay (latency) for a given point-to-point connection provides a good critera for appropriate threshold for the analysis. The *bandwidth-delay product* describes precisely how many bytes must be "in-flight" to fully utilize available link bandwidth. This can also be thought of as the minimum size required for a non-pipelined message to fully utilize available link bandwidth. Vendors commonly refer to an $N_{1/2}$ metric, which describes the message size below which you will get only 1/2 of the peak link performance; the $N_{1/2}$ metric is typically half the bandwidth-delay product. The thresholding criteria selects messages that are above the minimum message size that can theoretically saturate the link, i.e. those messages that are larger than the bandwidth-delay product.

Table 3.2 shows the bandwidth-delay products for a number of leading-edge interconnect implementations, where the best performance hovers close to 2 KB. Therefore a 2 KB threshold

Table 3.2: Bandwidth-delay products for several high performance interconnect technologies. This is the effective peak unidirectional bandwidth delivered per CPU (not per link).

System	Technology	MPI Latency	Peak Bandwidth	Bandwidth Delay Product
SGI Altix	NUMAlink-4	1.1us	1.9 GB/s	2 KB
Cray XT4	Seastar 2	7.3us	1.2 GB/s	8.8 KB
NEC SX-9	IXS Super-Switch	3us	16GB/s	48 KB
AMD Commodity Cluster	IB4x DDR	2.3us	950MB/s	2.2 KB

is chosen as the target bandwidth-limiting messaging threshold. This reflects the state-of-the-art in current switch technology and an aggressive goal for future leading-edge switch technologies. There is an implicit assumption that below that threshold, the latency-bound messages would not benefit from a dedicated point-to-point circuit. Such messages are only affected by topology when it comes to the number of links traversed, and cannot be sped up by increasing available bandwidth. Such messages would be routed over multiple links or a lower-bandwidth interconnect that is used for collectives. Therefore, in addition to a high-bandwidth hybrid interconnect, there is likely a need for a second low-latency low-bandwidth interconnect for handling collective communications with small payloads. A tree network, similar to the one used in the IBM BlueGene/L, does not incur a large additional cost because it is designed to handle low-bandwidth messages and can therefore employ considerably less expensive hardware components. This network could also carry small point-to-point messages that do not benefit from the high-bandwidth hybrid interconnect. However, such messages could also be routed over the high-bandwidth links without provisioning a dedicated path.

3.2 Evaluated Scientific Applications

This section will highlight the salient features of the eight applications studied that cover a broad range of communication requirements. A high level overview of the codes and methods is presented in Table 3.1. Each of these applications is actively run at multiple supercomputing centers, consuming a sizable amount of computational resources. Descriptions of the algorithms and scientific impacts of these codes have been extensively detailed elsewhere [7, 10, 11, 27, 38, 39, 43, 46]; but will be presented in brief below;

3.2.1 BBeam3D

BBeam3D [46] models the collision process of two counter-rotating charged particle beams moving at close to the speed of light. The application is a 3D particle-in-cell computation that contains multiple models (weak-strong, strong-strong) and multiple collision geometries (head-on, long-range, crossing angle), with collisions calculated self-consistently by solving the Vlasov-Poisson equation using Hockney's FFT method. Thus the code exhibits communication characteristics that reflect the combined requirements of the PIC method and the 3D FFT for the Poisson solver.

3.2.2 Cactus

Cactus [10] is an astrophysics computational toolkit designed to solve the challenging coupled nonlinear hyperbolic and elliptic equations that arise from Einstein's Theory of General Relativity. Consisting of thousands of terms when fully expanded, these partial differential equations (PDEs) are solved using finite differences on a block domain-decomposed regular grid distributed over the processors. The Cactus communication characteristics reflect the requirements of a broad variety of PDE solvers on non-adaptive block-structured grids.

3.2.3 GTC

The Gyrokinetic Toroidal Code (GTC) is a 3D particle-in-cell (PIC) application developed to study turbulent transport in magnetic confinement fusion [39]. GTC solves the non-linear gyrophase-averaged Vlasov-Poisson equations [37] in a geometry characteristic of toroidal fusion devices. By using the particle-in-cell (PIC) method, the non-linear PDE describing particle motion becomes a simple set of ordinary differential equations (ODEs) that can be easily solved in the Lagrangian coordinates. Unlike BB3D, GTC's Poisson solver is localized to individual processors, so the communication requirements only reflect the needs of the PIC core.

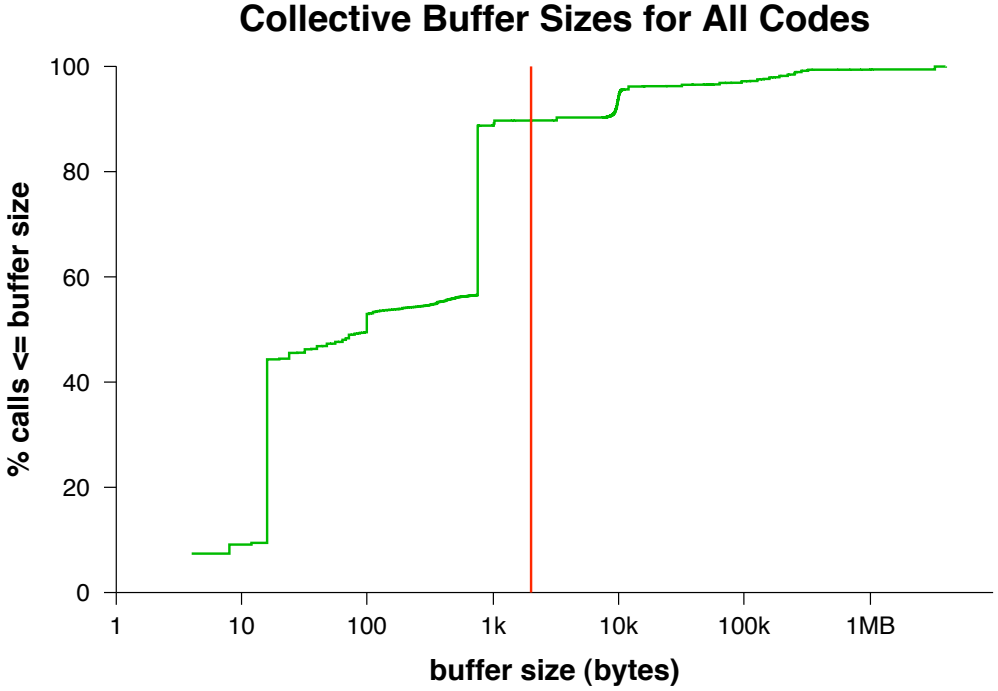
3.2.4 LBCFD

LBCFD [43] utilizes an explicit Lattice-Boltzmann method to simulate fluid flows and to model fluid dynamics. The basic idea is to develop a simplified kinetic model that incorporates the essential physics, and reproduces correct macroscopic averaged properties. LBCFD models 3D simulations under periodic boundary conditions, with the spatial grid and phase space velocity lattice overlaying each other, distributed with a 3D domain decomposition.

3.2.5 MadBench

Based on the MADspec cosmology code that calculates the maximum likelihood angular power spectrum of the cosmic microwave background (CMB), MADbench [7] is a simplified benchmark that inherits the characteristics of the application without requiring massive input data files. MADbench tests the overall performance of the subsystems of real massively-parallel architectures by retaining the communication and computational complexity of MADspec and integrating a dataset generator that ensures realistic input data. Much of the computational load of this application is due to its use of dense linear algebra, which is reflective of the requirements of a broader array of dense linear algebra codes in the scientific workload.

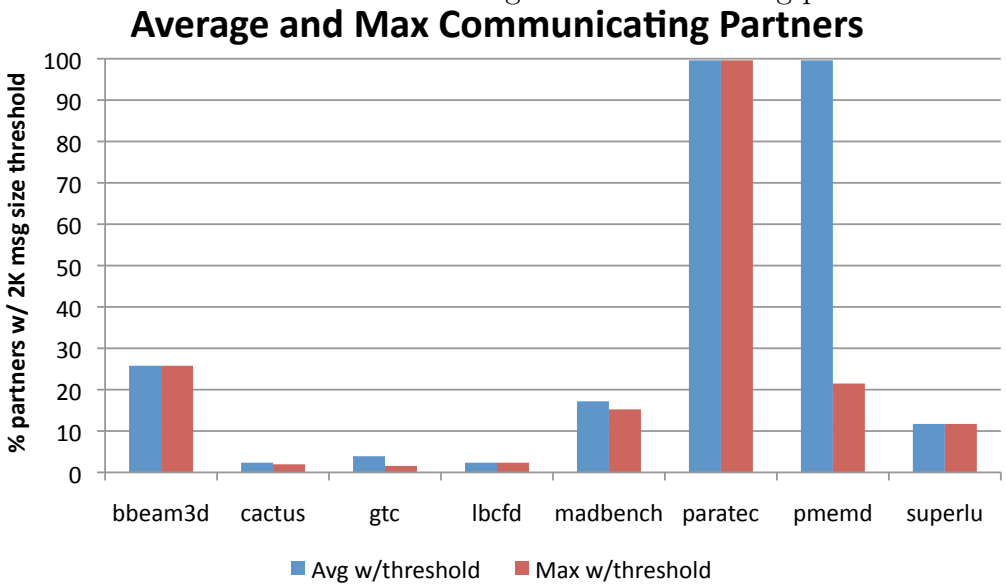
Figure 3.1: Buffer sizes distribution for collective communication for all codes. The pink line demarcates the 2 KB bandwidth-delay product.



3.2.6 ParaTEC

ParaTEC (PARAllel Total Energy Code [11]) performs ab-initio quantum-mechanical total energy calculations using pseudopotentials and a plane wave basis set. In solving the Kohn-Sham equations using a plane wave basis, part of the calculation is carried out in real space and the remainder in Fourier space using specialized parallel 3D FFTs to transform the wavefunctions. The communication involved in these FFTs is the most demanding portion of ParaTEC's communication characteristics. A workload analysis at the National Energy

Figure 3.2: Average and maximum communicating partners for the studied applications at $P = 256$, thresholded by the 2KB bandwidth-delay product. Communications smaller than the threshold are not considered in calculating the communicating partners.



Research Scientific Computing Center (NERSC) [67] has shown that Density Functional Theory (DFT) codes, which include ParaTEC, QBox, and VASP, account for more than 3/4 of the materials science workload.

3.2.7 PMEMD

PMEMD (Particle Mesh Ewald Molecular Dynamics [27]) is an application that performs molecular dynamics simulations and minimizations. The force evaluation is performed in an efficiently-parallel manner using state of the art numerical and communication methodologies. PMEMD uses a highly asynchronous approach to communication for the purposes of achieving a high degree of parallelism. PMEMD represents the requirements of a broader variety of molecular dynamics codes employed in chemistry and bioinformatics applications.

3.2.8 SuperLU

SuperLU [38] is a general purpose library for the direct solution of large, sparse, nonsymmetric systems of linear equations on high performance machines. The library routines perform an LU decomposition with partial pivoting as well as a triangular system solve through forward and back substitution. This application relies on sparse linear algebra of various kinds for its main computational kernels, ranging from a simple vector scale to a large triangular

solve. Sparse methods are becoming increasingly common in the scientific workload because they apply work only to non-zero entries of the matrix in order to improve time-to-solution for large scale problems.

Together, this collection of numerical methods spans the characteristics of a great many more applications, especially with respect to communication patterns. For example the core algorithm of the PARATEC code studied here, has the communication characteristics of many other important plane-wave density functional theory (DFT) calculations. Likewise a large number of finite difference and particle-mesh codes exhibit similar communication patterns to Cactus and PMEMD. Note that certain quantities relevant to the present study, such as communication degree, are largely dictated by the scientific problem solved and algorithmic methodology. For instance, in the case of Cactus where finite differencing is performed using a regular grid, the number of neighbors is determined by the dimensionality of the problem and the stencil size. Profiling a greater number of applications would of course improve the coverage of this study; however, the eight applications detailed here broadly represent a wide range of scientific disciplines and modern parallel algorithms under realistic computational demands.

Table 3.3: Breakdown of MPI communication calls, percentage of point-to-point (PTP) messaging, maximum and average TDC thresholded by 2 KB, and FCN utilization (thresholded by 2 KB) for evaluated application on 256 processors.

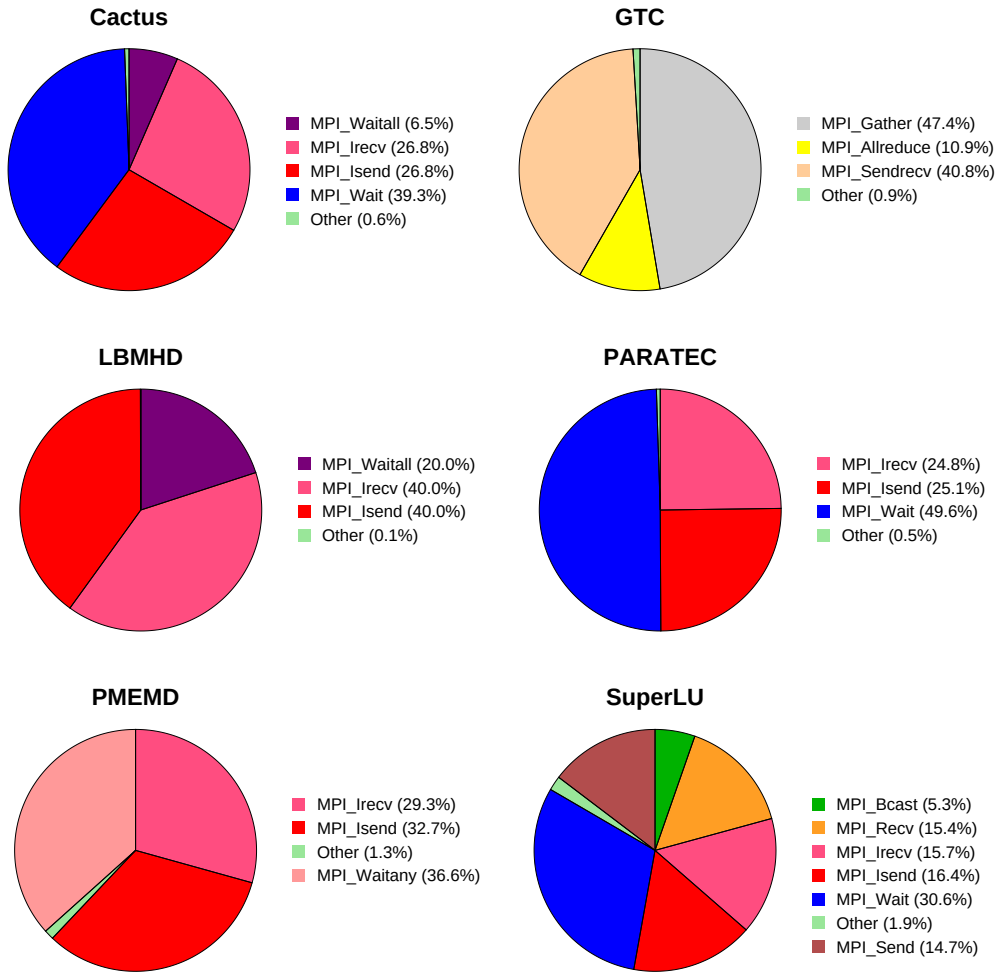
Function	BB3D	Cactus	GTC	LBCFD	MAD bench	PARA TEC	PMEMD	Super LU
Isend	0%	26.8%	0%	40.0%	5.3%	25.1%	32.7%	16.4%
Irecv	33.1%	26.8%	0%	40.0%	0%	24.8%	29.3%	15.7%
Wait	33.1%	39.3%	0%	0%	0%	49.6%	0%	30.6%
Waitall	0%	6.5%	0%	20.0%	0%	0.1%	0.6%	0%
Waitany	0%	0%	0%	0%	0%	0%	36.6%	0%
Sendrecv	0%	0%	40.8%	0%	30.1%	0%	0%	0%
Send	33.1%	0%	0%	0%	32.2%	0%	0%	14.7%
Gather	0%	0%	47.4%	0%	0%	0.02%	0%	0%
Reduce	0.5%	0.5%	11.7%	0.02%	13.6%	0%	0.7%	1.9%
Bcast	0.02%	0%	0.04%	0.08%	6.8%	0.03%	0%	5.3%
PTP Calls	99.2%	98.0%	40.8%	99.8%	66.5%	99.8%	97.7%	81.0%
TDC (max,avg)	66,66	6,5	10,4	6,6	44,39	255,255	255,55	30,30
FCN Utilization	25.8%	2.0%	1.6%	2.3%	15.3%	99.6%	21.4%	11.7%

3.3 Communication Characteristics

The communication characteristics of the studied applications will be analyzed by quantifying the MPI call count distributions, collective and point-to-point buffer sizes, and topological connectivity.

3.3.1 Call Counts

Figure 3.3: Relative number of MPI communication calls for each of the codes.



The breakdown of MPI communication call types is shown in Table 3.3, for each of the studied applications. The analysis only considers calls dealing with communication and synchronization, and do not analyze other types of MPI functions which do not initiate or

complete message traffic. Notice that overall, these applications utilize only a small subset of the entire MPI library. Figure 3.3 shows that most codes use a small variety of MPI calls, and utilize mostly point-to-point communication functions (over 90% of all MPI calls), except GTC, which relies heavily on `MPI_Gather`. Observe also that non-blocking communication is the predominant point-to-point communication model for these codes.

3.3.2 Buffer Sizes for Collectives

Figure 3.1 presents a cumulative histogram of buffer sizes for collective communication (that is, communication that involves all of the processors), across all eight applications. Observe that relatively small buffer sizes are predominantly used; in fact, about 90% of the collective messages are 2 KB or less (shown as the bandwidth-delay product by the pink line), while almost half of all collective calls use buffers less than 100 bytes. These results are consistent with previous studies [62,63] and validate IBM's architectural decision to dedicate a separate lower-bandwidth network on their BlueGene machines for collective operations. For this broad class of applications, collective messages are mostly constrained by the latency of the interconnect, regardless of the topological interconnectivity.

3.3.3 Point-to-Point Buffer Sizes

A cumulative histogram of buffer sizes for point-to-point communication is shown in Figure 3.4 for each of the applications; once again the 2 KB bandwidth-delay product is shown by the pink vertical lines. A wide range of communication characteristics are observed across the applications. Cactus, LBCFD, and BBeam3D use a relatively small *number* of distinct buffer sizes, but each of these buffers is relatively large. GTC employs some small communication buffers, but over 80% of the messaging occurs with 1 MB or larger data transfers. In addition, it can be seen that SuperLU, PMEMD, MADbench, and PARATEC use many different buffer sizes, ranging from a few bytes to over a megabyte in some cases. Overall, Figure 3.4 demonstrates that unlike collectives (Figure 3.1), point-to-point messaging in these applications uses a wide range of buffers, as well as large message sizes. In fact, for all but two of the codes, buffer sizes larger than the 2 KB bandwidth-delay product account for $> 75\%$ of the overall point-to-point message sizes.

3.3.4 Topological Connectivity

This section explores the topological connectivity for each application by representing the volume and pattern of message exchanges between all tasks. By recording statistics on these message exchanges we form an undirected graph that describes the topological connectivity required by each application. Note that this graph is undirected as we assume that most

modern switch links are bi-directional; as a result, the topologies shown are always symmetric about the diagonal. From this topology graph we then calculate the quantities that describe communication patterns at a coarse level. Such reduced metrics are important in allowing us to make direct comparisons between applications. In particular the maximum and average TDC (connectivity) of each code is examined because it is a key metric for evaluating the potential of lower-degree and non-traditional interconnects. The analysis shows the max and average connectivity using a thresholding heuristic based on the bandwidth-delay product (see Section 3.1.2) that disregards smaller latency-bound messages. In many cases, this thresholding lowers the average and maximum TDC substantially. An analysis of these results in the context of topological network designs are presented in Section 3.4.

Figure 3.5(a) shows the topological connectivity of BBeam3D for $P = 256$ as well as the effect of eliminating smaller (latency-bound) messages on the number of partners. Observe the high TDC for this charge density calculation due to its reliance on data transposes during the 3D FFTs. For this code, the maximum and average TDC is 66 neighbors; both of these are insensitive to thresholding lower than 64 KB. BBeam3D thus represents an application class that exhibits a TDC smaller than the full connectivity of a fat tree, with little sensitivity to bandwidth-limited message thresholding.

Figure 3.5(b) shows the ghost-zone exchanges (halo exchange) of Cactus result in communications with "neighboring" nodes, represented by diagonal bands. In fact, each node communicates with at most six neighbors due to the regular computational structure of this 3D stencil code. On average, the TDC is 5, because some nodes are on the boundary and therefore have fewer communication partners. The maximum TDC is independent of run size (as can be seen by the similarity of the $P = 64$ and $P = 256$ lines) and is insensitive to thresholding, which suggests that no pattern of latency-bound messages can be excluded. Note however that the low TDC indicates limited utilization of an FCN architecture.

As shown in Figure 3.5(c), GTC exhibits a regular communication structure typical of a particle-in-cell calculation that uses a one-dimensional domain decomposition. Each processor exchanges data with its two neighbors as particles cross the left and right boundaries. Additionally, there is a particle decomposition within each toroidal partition, resulting in an average TDC of 4 with a maximum of 17 for the $P = 256$ test case. This maximum TDC is further reduced to 10 when using the 2 KB bandwidth-delay product message size threshold. These small TDC requirements clearly indicate that most links on an FCN are not being utilized by the GTC simulation.

The connectivity of LBCFD is shown in Figure 3.5(d). Structurally, the communication occurs in several diagonal bands, just like Cactus. Note that although LBCFD streams the data in 27 directions (due to the 3D decomposition), the code is optimized to reduce the number of communicating neighbors to 6, as seen in Figure 3.5(d). This degree of connectivity is insensitive to the concurrency level. The maximum TDC is insensitive to thresholding, showing that the communications of this application use larger message sizes.

MADbench's communication topology characteristics are shown in Figure 3.5(e). Each pro-

cessor communicates with 38 neighbors on average, dropping to 36 if we eliminate messages smaller than 2 KB. The communication is relatively regular due to the underlying dense linear algebra calculation, with an average and maximum TDC that are almost identical. MADbench is another example of a code whose overall TDC is greater than the connectivity of a mesh/torus interconnect, but still significantly less than the number of links provided by a fat-tree.

Figure 3.5(f) shows the complex structure of communication of the PMEMD particle mesh ewald calculation. Here the maximum and average TDC is equal to P and the degree of connectivity is a function of concurrency. For the spatial decomposition used in this algorithm, the communication intensity between two tasks drops as their spatial regions become more distant. The rate of this drop off depends strongly on the molecule(s) in the simulation. Observe that for $P = 256$, thresholding at 2 KB reduces the average connectivity to 55, even though the maximum TDC remains at 256. This application class exhibits a large disparity between the maximum and average TDC.

Figure 3.5(g) shows the communication requirements of PARATEC. This communication-intensive code relies on global data transposes during its 3D FFT calculations, resulting in large, global message traffic [11]. Here the maximum and average TDC is equal to P, and the connectivity is insensitive to thresholding. Thus, PARATEC represents the class of codes that make use of the bisection bandwidth that an FCN configuration provides.

Finally, Figure 3.5(h) shows the connectivity and TDC for SuperLU. The complex communication structure of this computation results in many point-to-point message transmissions: in fact, without thresholding the connectivity is equal to P. However, by removing the latency-bound messages by thresholding at 2 KB, the average and maximum TDC is reduced to 30 for the 256 processor test case. Also, note that the connectivity of SuperLU is a function of concurrency, scaling proportionally to $\sqrt{P}$ (see [38]).

The following section measures the topological connectivities in the context of interconnect requirements.

3.4 Communication Connectivity Analysis

Based on the topological connectivities of our applications, the codes are categorized as follows: Applications with communication patterns such that the maximum TDC is less than the connectivity of the interconnection network (*case i*) can be perfectly embedded into the network, albeit at the cost of having some connections be wasted/idle. If the TDC is equal to that of the underlying interconnect and the communication is isomorphic to the network architecture, then the communication can also be embedded (*case ii*). However, if the TDC is equal and the the communication is non-isomporphic to the interconnect (*case iii*) or if the TDC is higher than the underlying network (*case iv*), there is no embedding without sharing some links for messaging, which can lead to message contention for bandwidth bound

messages.

3.4.1 Collectives

Consistent with the hypotheses presented in the previous subsection, Figure 3.1 shows that nearly all of the collective communication payload sizes fall below 2 KB. This result is consistent with previous research [62] and validates IBM's architectural decision to dedicate a separate lower-bandwidth network on BG/L for collective operations. One could imagine computing a minimum-latency routing pattern that is overlaid on the high-bandwidth interconnect topology, but the complexity of such an algorithm is out of the scope of this paper. This traffic can be carried over a lower-bandwidth, latency-oriented dedicated-tree network, similar to the one in IBM Blue Gene that carries collective messages and possibly small-payload point-to-point messages, and focus the remaining analysis on accelerating large payload, bandwidth-bound, point-to-point messages. This secondary low-latency low-bandwidth network can play a central role in managing circuit switch configurations, which will be discussed in more detail in the next chapter.

3.4.2 Point-to-Point Traffic

This section examines the communication traces for each of the applications and considers the class of network best suited for its communication requirements. First, the four codes exhibiting the most regularity in their communication exchanges are examined: Cactus, GTC, LBCFD, and MADbench. Cactus displays a bounded TDC independent of run size, with a communication topology that isomorphically maps to a regular mesh; thus a fixed 3D mesh/torus would be sufficient to accommodate these types of stencil codes, although an adaptive network (see Section 4.2) would also fulfill Cactus's requirements (i.e. consistent with *case i*). LBCFD and MADbench also display a low degree of connectivity; however, while their communication pattern is isotropic, their respective structures are not isomorphic to a regular mesh, thereby corresponding to *case iii* classification. Although GTC's primary communication pattern is isomorphic to a regular mesh, it has a maximum TDC that is higher than the average due to important connections that are *not* isomorphic to a mesh (*case iv*). Thus a fixed mesh/torus topology would be not well suited for this class of computation.

BBeam3d, SuperLU and PMEMD all exhibit anisotropic communication patterns with a TDC that scales with the number of processors. Additionally, PMEMD has widely differing maximum and average TDC. However, with thresholding, the proportion of processors that have messages that would benefit from the dedicated links is large but stays bounded to far less than the number of processors involved in the calculation (consistent with *case iii*). Thus a regular mesh or torus would be inappropriate for this class of computation, while an FCN remains underutilized.

Finally, PARATEC represents the communications requirements for a large class of important chemistry and fluids problems where part of the problem is solved in Fourier space. It requires large global communications involving large messages that fully utilize the FCN and are therefore consistent with *case iv*. PARATEC's large global communications are a result of the 3D FFTs used in the calculation, which require two stages of global 3D transposes. The first transpose is non-local and involves communications of messages of similar sizes between all the processors, resulting in the uniform background of 32 KB messages. In the second transpose, processors only communicate with neighboring processors, resulting in additional message traffic along the diagonal of the graph. PARATEC's large global communication requirements can only be effectively provisioned with an FCN network.

In summary, only one of the eight codes studied (Cactus) offered a communication pattern that maps isomorphically to a 3D mesh network topology (*case i*). This indicates that mesh/torus interconnects may be insufficient for a diverse scientific workloads. Additionally, only PARATEC fully utilizes the FCN at large scales (*case iv*); thereby undercutting the motivation for using FCNs across a broad range of computational domains. The underutilization of FCN for our codes can be clearly seen in the last row of Table 3.3. Thus, for a wide range of applications (cases *ii* and *iii*), we believe there is space to explore alternative interconnect architectures that contain fewer switch ports than a fat-tree but greater connectivity than mesh/tori networks; such interconnects are explored further in Section 4.1.

Figure 3.4: Buffer sizes distribution for point-to-point communication. The pink lines demarcate the 2 KB bandwidth-delay product.

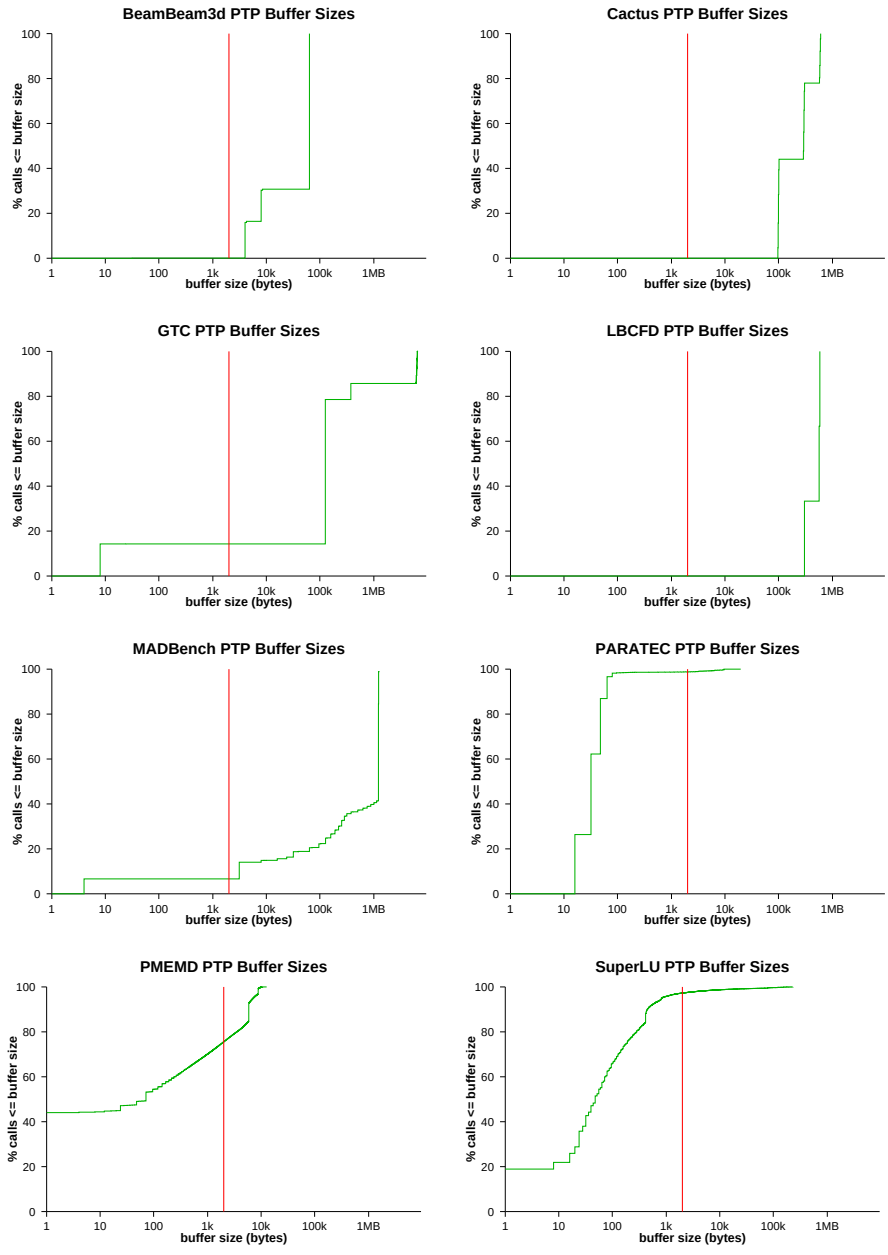
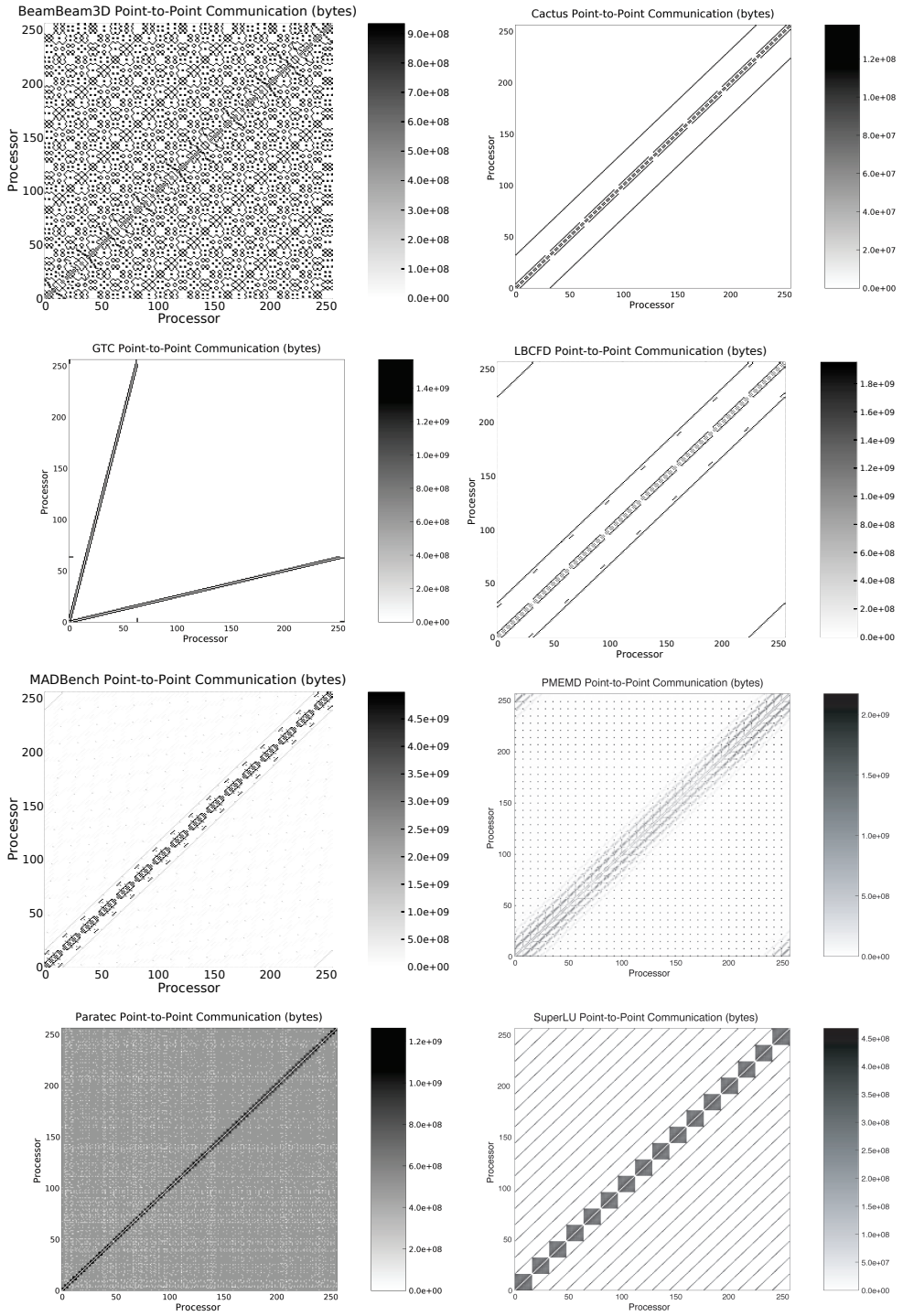


Figure 3.5: Topological connectivity of each of the studied applications, showing volume of communication at P=256.



Chapter 4

Developing an Optimized Topology for System Scale Interconnects

The previous chapter performed a deep analysis of application requirements to guide the design of future interconnects. This section reprocesses the raw communication data to understand how it utilizes a multi-stage CLOS network to expose opportunities for reducing the component count of an interconnect to more closely match application requirements. It then introduces the concept of a fit-tree, which is a CLOS topology that has been optimized to eliminate components that are otherwise not utilized by the network. The chapter closes with a discussion of a hybrid approach to interconnect design called *Hybrid Flexibly Assignable Switch Topology* (HFAST) infrastructure, which allows the implementation of interconnect topologies that are specifically tailored to application requirements, via the proposed fit-tree approach or other mapping strategies.

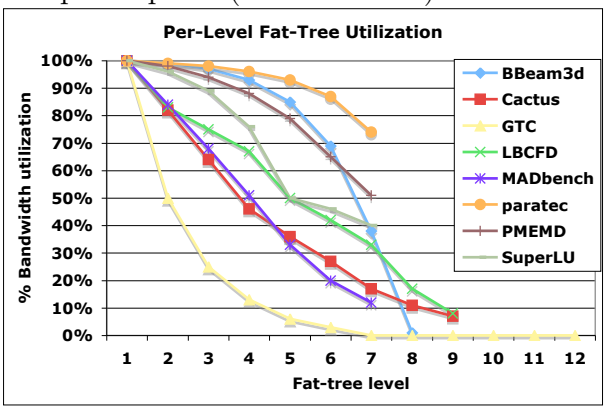
4.1 Fit-Tree Interconnect Analysis

The analysis in the previous chapter showed that the communication patterns of most applications have irregular patterns, evincing the limitations of 3D mesh interconnects. At the same time, most communication patterns are sparse, revealing that the large bandwidth of a FCN is not necessary and have good locality, showing that an intelligent task-to-processor assignment can significantly decrease the load on the network. This section demonstrates how statistics about communication patterns of target applications can be adopted to build interconnects that are more effective and cost-efficient. Specifically, we start with a fat-tree topology, and then develop the concept of a *fit-tree*, which allows comparable performance on target applications at a fraction of the interconnect resources of a fat-tree. While the science-driven approach is examined in the context of fat-trees in this article, the same analysis may be applied to other popular topologies; the choice of fat-trees is motivated by their

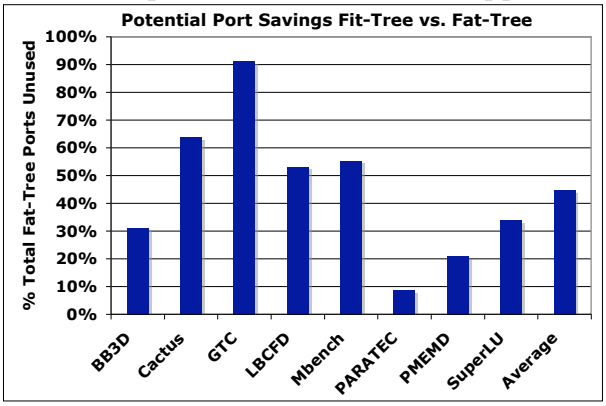
high popularity, as evidenced by their strong presence in TOP500 list.

This analysis starts with a review of the fat-tree topology and its resource requirements in Section 4.1.1, and then examine how well this topology corresponds to the application communication requirements in Section 4.1.2. Establishing the under-utilization of fat-tree network resources, motivates the novel fit-tree methodology described in Section 4.1.3.

Figure 4.1: (a) Underutilization of fat-tree bandwidth for the examined application suite. Level 1 refers to the bottom of the tree closest to the processors. The vertical axis represents percentage of the bandwidth utilized at each level. (b) The potential savings in the number of required ports (and thus cost) for an ideal fit-tree compared with the fat-tree approach.



(a)



(b)

4.1.1 Fat-Tree Resource Requirements

Conceptually, a fat-tree is a k-ary tree with processors on the bottom-most level, where the thicknesses (capacities) of the edges increase at higher levels of the tree. Here, k is defined by the $k \times k$ switch block size used to implement the network. That is, 2×2 switches will yield a binary tree, 4×4 switches yield a 4-ary tree, etc. In a conventional fat-tree, the total bandwidth is constant for each level of the tree; thus the thickness of an edge at level $i + 1$ is k times the thickness at level i. Messages can travel up the tree and back down to traverse from a processor to any other processor without being constrained by bandwidth limitations; this structure can be thought of as a "folded" Benes network [14].

The relation between the number of levels, number of processors and the number of switch boxes is now quantified. A fat-tree with L levels built with $k \times k$ switches can have up to $2k^L$ processors, since the number of nodes is multiplied by k at each level from the root down to the tree's bottom. Conversely, the depth of a fat-tree for P processors built with $k \times k$ switches is $\log_k P - \log_k 2$. The corrective term of 2 is due to the root level of the fat-tree, where all switch ports are available for the lower level, unlike intermediate levels, where half

of the ports are used for connections with the higher level. Since the total bandwidth at each level is constant, so are the number of switch ports per level. As a result, the bottom level of the fat-tree, which connects the processors to the network, requires $\lceil \frac{P}{k} \rceil$ switches; thus a fat-tree with L levels built with $k \times k$ switches requires $\frac{LP}{k} = 2Lk^{L-1}$ switches. Conversely, building a fat tree for P processors requires $(\log_k P - \log_k 2)\lceil \frac{P}{k} \rceil$ of $k \times k$ switches.

Constructing fat-trees where the network bandwidth is preserved at all levels is extremely challenging for thousands of processors, and simply infeasible for the next-generation of ultrascale computing systems with tens or hundreds of thousands of processors. Besides the construction complexity, the performance of a fat-tree network degrades while the cost inflates sharply with increasing processor count. From the performance perspective, as the depth of the tree increases with larger concurrencies, the number of hops per message increases, corresponding to larger message latencies. While latency due to the interconnection network may not be significant for small to medium number of processors, it can dominate the message transmission cost at very high concurrencies. Additionally, the cost of a fat-tree grows superlinearly with larger parallel systems, since fat-tree construction depends on the number of switching blocks as well as the number of cables employed. These factors eliminate fat-tree topologies as a practical interconnection paradigm for next generation supercomputers.

4.1.2 Fat-Tree Utilization

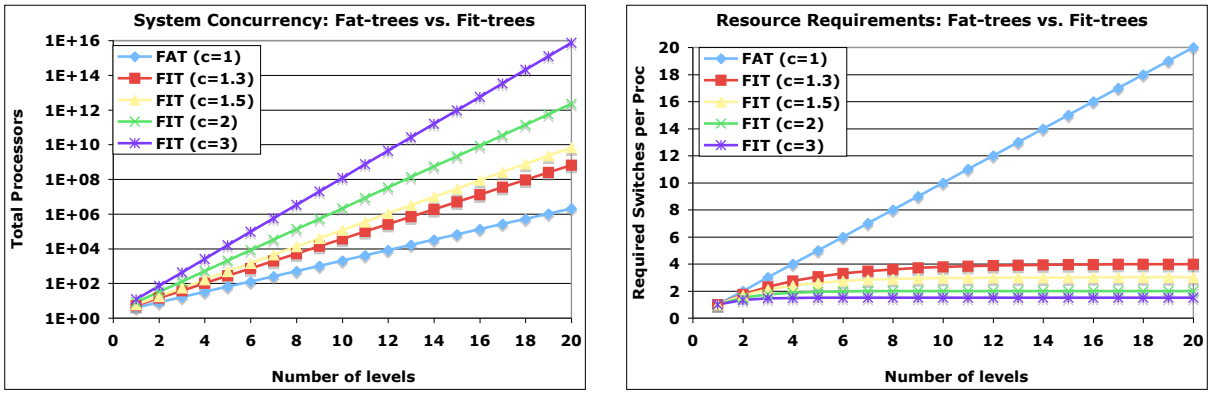
This section analyzes the fraction of available fat-tree bandwidth that is utilized by the selected applications. In previous work [32], we employed two methods to assign tasks to processors: one that assigns processors based on the natural ordering of the tasks, and a second method that aims to minimize the average number of hops for each message using a heuristic based on graph partitioning. For the analysis here, the tasks are assigned to processors using the heuristic methodology.

The application communication patterns presented in Section 5.4 are used to create an *instance* of communication. For a given instance, a processor sends a message to one of its communicating partners chosen at random. About $10P$ instances of communication for each application are created to approximate the communication overhead as the messages are routed over the interconnect. The pathway of the messages are recorded as they reach each level of the fat-tree. Using this estimation strategy, the behavior of each application is simulated to determine the communication load on the network.

Figure 4.1(a) displays the results for bandwidth utilization of a fat-tree built with 2×2 switches. In this figure, the horizontal axis corresponds to the the fit-tree level starting with the leaf nodes (i.e. the processors). The vertical axis correspond to bandwidth utilization, which is computed by counting the number of messages that reach a given level, and comparing this number with the level's total available bandwidth (P for a fat-tree). The results show that the bandwidth utilization drops sharply as the tree level increases. For GTC, this

number drops to 0 at level seven, indicating that the highest six levels of the fat-tree are not used at all. A similar trend is seen in all examined applications. Even for PARATEC, which uses all-to-all-communication in its FFT, bandwidth utilization goes down to 74% at the top level even though P is only 256 processors. These results clearly show that fat-tree bandwidth is underutilized for most applications, especially for those that can scale up to thousands of processors. The next section will use this observation to propose an alternative interconnection topology.

Figure 4.2: Comparison of fat-tree and fit-tree scalabilities in terms of (a) potential system concurrency for a fixed number of tree levels and (b) the required number of switches per processor.



(a) (b)

4.1.3 Fit-Tree Approach

The motivation for the *fit-tree* topology comes from the observation that the available bandwidth of a fat-tree is not utilized at all levels — especially the higher ones — of a fat-tree network for many scientific computing applications. The fit-tree topology is an improvement on fat-trees, that exploits the locality of bandwidth utilization in fat-trees to provide a design options with better scalability in terms of performance and cost.

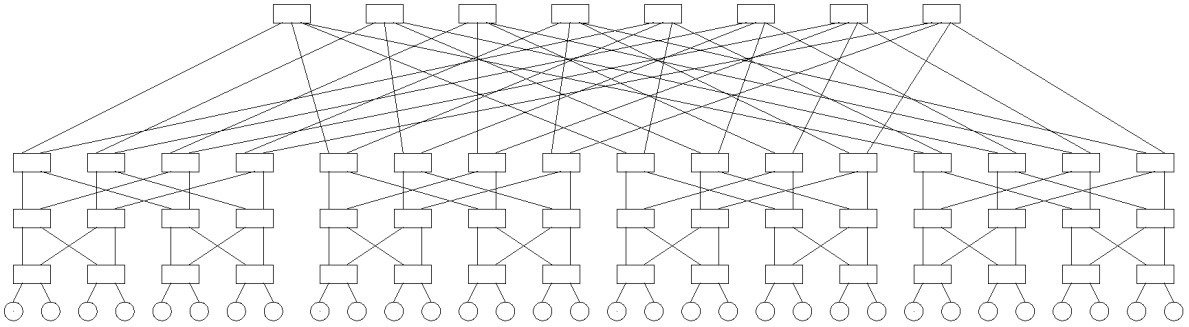
Consider an intermediate node in a fat-tree that is the root of a sub-tree of P' processors. In a conventional fat-tree, this node corresponds to a $P' \times P'$ switch, whose ports are assigned so that P' of them are connected to the lower level and P' are connected to the higher level. This provides P' different communication channels for P' processors. Since some of this bandwidth is redundant (Section 4.1.1), eliminating a portion of the connections within the higher level will not degrade performance. This kind of redundancy can be useful for algorithms that exhibit a more randomized messaging pattern such as graph algorithms. However, the mix of scientific applications extracted from the DOE workload do not demonstrate such random access patterns, so the upper-tiers of the interconnect are underutilized for those

cases. Future work may consider a deeper analysis of the communication requirements for graph algorithms. Note that although this network design optimization decreases cabling requirements, it does not improve switch costs or overall performance.

In the proposed fit-tree design, the number of ports used for connections to the higher levels of the tree is less than the number of ports used for connections to the lower levels. This approach leverages otherwise un-utilized switch ports to increase the number of connected nodes at lower tree levels, allowing an increase in the number of processors rooted at a node (at the same level). Thus the fit-tree design has a $c:1$ ratio between the number of ports that go down and up (respectively) for each intermediate level, where $c > 1$ is the *fitness ratio*. Conversely, a conventional fat-tree has a $1:1$ ($c = 1$) ratio between bandwidth down and up at each intermediate level.

The fit-tree methodology enables building larger systems for a fixed number of levels in the interconnect tree. A direct comparison with fat-trees can be made in two ways. If the total bandwidth is preserved at each level of the tree, a fat-tree is built using k children per node. However, a fit-tree's node has ck children, where c is the fitness ratio. This translates to an exponential advantage in the number of processors the interconnect can support, as a fit-tree of L levels built with $k \times k$ switches and a $c:1$ ratio will contain $2(ck)^L$ processors as opposed to $2k^L$ for a fat-tree. Conversely, the depth of a fit-tree for a fixed number of processors P built with $k \times k$ switches and a $c:1$ ratio is $\log_{ck} P - \log_{ck} 2$. This reduced number of levels for a fixed number of processors translates to a reduction in switch count. Overall, this results in a substantial reduction in the port counts and consequent wiring complexity required to implement a fit tree that offers the same performance as a fat tree. Figure 4.1.2(a) shows the advantage of the fit-tree approach for potential system concurrency given a fixed number of levels. Alternatively, one can consider fixing the number of tree levels while decreasing the

Figure 4.3: A four-level fat-tree built from $2x2$ switches. The fit-tree approach "trims" links at the upper levels if the extra bandwidth is unneeded and packs the resulting necessary links into as few switch blocks as possible.



total fit-tree bandwidth at the higher levels. For a fat-tree the total bandwidth provisioned

is computed as:

$$\sum_{i=1}^{L-1} \frac{P}{kc^{i-1}} = \frac{P(\frac{1}{c^L} - 1)}{k(\frac{1}{c} - 1)} < \frac{Pc}{k(c-1)}$$

In a fit-tree, however, the bandwidth can be reduced by c at each level of the tree. It is worth noting that the total bandwidth, and thus the number of switches, scales linearly with P, which provides perfect scalability for fit-trees. For example, given $c = 2$, the total number of required switches will be no more than two times the number of switches at the first level. Figure 4.1.2(b) highlights the fit-tree advantage, by showing a comparison of the required number of switch components required for fat- and fit-trees using varying fitness ratios.

In practice it is possible to build hybrid topologies, where each node has more than k children, thus allowing the bandwidth to reduce gradually. This allows fit-tree designers to trade off between cost savings and performance. A hybrid optical/electrical interconnect solution would allow fit-tree designs that could be dynamically reconfigured to the requirements of the underlying application. The potential advantage of a fit-tree architecture is examined for the evaluated set of scientific codes.

Table 4.1: Fitness ratios for (top) each applications across all levels and (bottom) each level across all applications

Code	BB3D	Cactus	GTC	LBCFD	MAD bench	PARA TEC	PMEMD PMEMD	Super LU
Min	1.01	1.22	1.92	1.11	1.19	1.01	1.02	1.04
Avg	1.60	1.40	3.01	1.41	1.44	1.05	1.12	1.17
Max	4.00	1.59	4.00	1.94	1.67	1.18	1.27	1.52
Med	1.09	1.36	3.00	1.24	1.44	1.03	1.09	1.12

Level	1	2	3	4	5	6	7	8
Min	1.01	1.01	1.02	1.03	1.07	1.15	1.54	1.57
Avg	1.21	1.22	1.26	1.39	1.35	1.62	2.87	2.57
Max	2.00	2.00	1.92	2.17	2.00	3.00	4.00	4.00
Med	1.12	1.09	1.15	1.31	1.22	1.43	2.97	2.13

4.1.4 Fit-Tree Evaluation

The previous section showed how fit-trees can significantly improve the cost and scalability of fat-trees, while preserving performance. The critical question is therefore determining the appropriate fitness ratio for a given computation. In this section, we investigate the fitness ratio requirements of the selected applications. These experiments use the same experimental setup as in Section 4.1.2, and compute the fitness ratio of level $i + 1$ as the ratio between

the bandwidth utilization at level $i+1$ and i respectively. To reduce the effect of outliers, we consider 4 to be the highest allowable fitness ratio.

Table 4.1(top) presents the fitness ratios of each examined application. For clarity of presentation, only the minimum, average, maximum, and median across all fit-tree levels is shown. Results show that (as expected) fitness ratios are higher for applications with sparse communication: GTC, BB3D, LBCFD, MADbench, Cactus, and SuperLU. Note that these applications are known to exhibit better scalability compared to communication-intensive computations such as PARATEC and PMEMD. However, it is remarkable that even PARATEC, which require global 3D FFTs, has a fitness ratio of 1.18 at its top level.

Table 4.1(bottom) presents fitness ratios for each level of the fit-tree across all studied applications. Again for clarity of presentation, only the minimum, average, maximum, and median values are displayed. Results show that, while the fitness ratios are low at the lowest levels, they increase with increasing fit-tree levels. This is expected, as the number of nodes rooted at a node is doubled at each level of the fat-tree, creating room for locality where the percentage of local communication increases.

Based on Table 4.1 it is difficult to decide on a single "ideal" fitness ratio, but the data show strong quantitative support for the fit-tree concept. After all, even the minimum fitness ratio at level six is 1.15. It is worth repeating that the main motivation is interconnect designs for next-generation petascale and exascale systems, which are expected to have hundreds of thousands of processors. Therefore, even a fitness ratio of 1.15 will translate to enormous savings in costs and improvements in performance as displayed in Figure 4.1.2. The potential savings in switch ports versus a fat-tree for the examined applications is shown in Figure 4.1(b). Even for the moderate concurrency levels explored here, the hybrid fit-tree approach can reduce the port count requirements by up to 44% (on average).

The fit-tree methodology provides guidance on how to construct multi-tiered interconnect topologies that have just enough components to meet the application's messaging requirements. The next chapter will propose a a hardware solution that uses this information to dynamically constructing fit-trees appropriate for each application, thus building the best-performing interconnect at the lowest cost in terms of switch ports.

4.2 HFAST: A Reconfigurable Interconnect Architecture

As computing technology moves towards building clustered computing systems containing tens (or hundreds) of thousands of processors, building fully-connected interconnects quickly becomes infeasible due to the superlinear cost of this network design approach. Moreover, the analysis of large-scale scientific applications in Section 5.4 quantified the under-utilization of FCNs for a wide array of numerical methods. Section 4.1 presented a fit-tree methodology

and demonstrated that it has the potential to improve network performance while reducing component costs.

The fit-tree analysis in the previous section demonstrated the tremendous savings that can be achieved by designing interconnects by taking application requirements into account. In particular, the analysis demonstrated that for many applications, fit-trees can be as effective as fat-trees in meeting application requirements, but exhibit linearly scaling costs. However, the caveat is the communication requirements of application can vary significantly. Thus guaranteeing high performance for all code classes requires designing each part of the interconnect for the maximum demand among all target applications — resulting in over-provisioning and degraded efficiency. The remedy is in reconfigurability, which would allow construction of an interconnect topology dynamically for each application to achieve maximum performance with minimum resources.

This chapter presents a methodology for dynamically creating interconnects. This approach would allow fit-trees to be constructured with variable fitness ratios as well as arbitrary network configurations. The proposed technology, called HFAST (Hybrid Flexibly Assignable Switch Topology), uses passive/circuit switches to dynamically provision active/packet switch blocks — allowing the customization of interconnect resources for application-specific requirements. The next section examines recent trends in the high-speed optical wide area networking community, which has developed cost-effective solutions to similar challenges in order to understand the motivation for this approach.

4.2.1 Circuit Switch Technology

Packet switches, such as Ethernet, Infiniband, and Myrinet, are the most commonly used interconnect technology for large-scale parallel computing platforms. A packet switch must read the header of each incoming packet in order to determine on which port to send the outgoing message. As bit rates increase, it becomes increasingly difficult and expensive to make switching decisions at line rate. Most modern switches depend on ASICs or some other form of semi-custom logic to keep up with cutting-edge data rates. Fiber optic links have become increasingly popular for cluster interconnects because they can achieve higher data rates and lower bit-error-rates over long cables than is possible using low-voltage differential signaling over copper wire. However, optical links require a transceiver that converts from the optical signal to electrical so the silicon circuits can perform their switching decisions. The Optical Electrical Optical (OEO) conversions further add to the cost and power consumption of switches. Fully-optical switches that do not require an OEO conversion can eliminate the costly transceivers, but per-port costs will likely be higher than an OEO switch due to the need to use exotic optical materials in the implementation [1].

Circuit switches, in contrast, create hard-circuits between endpoints in response to an external control plane – just like an old telephone system operator's patch panel, obviating the need to make switching decisions at line speed. As such, they have considerably lower com-

Figure 4.4: Optical Circuit Switching elements. (a) A micro-electromechanical mirror is the central component of the Movaz optical circuit switch (OCS) module shown in (b). (c) A combination of eight ring resonators allows the construction of a 4×4 nonblocking optical switch based on silicon photonic ring resonator technology developed at Cornell and Columbia University.

(a) MEMS Switch Element

(b) Movaz OCS Module

(c) SiPhotonic Ring Resonator Switch

plexity and consequently lower cost per port. Circuit switches enable considerable power and cost savings as they do not require expensive (and power-hungry) optical/electrical transceivers required by the active packet switches. Also, because non-regenerative circuit switches create hard-circuits instead of dynamically routed virtual circuits, they contribute almost no latency to the switching path aside from propagation delay.

One such technology for all optical interconnects are micro-electro-mechanical mirror (MEMS) based optical circuit switches. MEMS based optical switches, such as those produced by Lucent, Calient and Glimmerglass, are common in the telecommunications industry and the prices are dropping rapidly as the market for the technology grows larger and more competitive. MEMS based technologies have been deployed broadly for wide-area telecommunications applications to create circuit-switched network fabrics, such as the NSF National Lambda Rail and Canada's CANARIEnet high performance national network [16, 57].

Another technology that has emerged more recently are solid-state silicon photonic technology that has seen rapid advances in the past 5 years. In particular, ring-resonator and Mach-Zehnder optical switch technologies are capable of routing a single path from any source to any destination using Photonic Switching Elements, shown in Figure 4.4, which are simple structures that, when inactive, consume little power and simply pass optical data through. Switching a PSE uses a tiny amount of power, and the PSE consumes a small active power while switched to bend the beam of light 90 degrees, causing the message to turn. Mach-Zehnder based optical circuit switches have seen recent commercial application in Luxtera active optical cables. It is anticipated that such silicon photonic optical circuit switches will rapidly supplant directly modulated lasers, and see further integration with conventional silicon lithography technology. The silicon photonic architecture will be described

in more detail in Chapter 5, which will cover optical *Network on Chip* (NoC) architecture.

4.2.2 Relationship to Wide Area Networks

Networking providers need to over-provision network resources in order to minimize resource contention – otherwise data transfer performance for the most demanding applications suffers greatly and guaranteed quality-of-service becomes all but impossible. However, network providers have noted that the applications that drive the most demanding bandwidth requirements for wide area networks tend to establish a limited number of high-performance point-to-point connections. While packet switches are capable of inspecting and routing each packet that arrives on an interface, the capability of that resource is wasted when the majority of packets are associated with the same source and destination addresses. Given the topological requirements of the most demanding applications, and the fact that per-port cost of a full-crossbar circuit switch is a fraction of that of an equivalent packet switch, high-performance networking has been rapidly moving towards a hybrid packet-switched/circuit-switched infrastructure.

Circuit switches have been a central component of the telecommunications infrastructure since the very beginning of the telephone network. While their presence has typically been hidden from wide area networking, a number of wide-area network service providers have begun to deploy light-paths and circuit-switched networks that offer dedicated circuits to the most demanding applications. These dedicated circuits, known as "light-paths" or "switched lambdas", provide performance guarantees that are far better than can be offered by the usual best-effort packet-switched networks. Robust control-plane technology like GMPLS [50] is able to control the light paths created by the passive circuit switches and the routed virtual circuits created by the active packets switches in tandem—providing transparent control of the network topology. The novel part of new trends in the networking community is the software that allows user applications to announce their requirements to the control-plane using protocols like UCLP (User Controlled Light Paths) pioneered by CANARIENet and StarLight in recent years [16].

The resulting architecture offers much better performance guarantees than can be offered by a mere best-effort packet-switched network and does so at a lower cost. Packet switches and routers can be enormously expensive hardware components. A high-performance router suitable for wide-area networking, capable of managing multiple OC-192/SONET (10 gigabit) connections, can easily cost upwards of half-a-million dollars. A typical GMPLS-capable MEMS-based optical circuit switch costs a fraction as much per port. As optical circuit switch technology matures in the telecommunications world, the cost per port is rapidly becoming competitive with local area networking packet switches and may well offer a cost-effective alternative to the packet switches employed in supercomputer interconnects. There is a significant number of research efforts that attempts to exploit the cost-advantages of circuit switched interconnects.

HFAST shares many technologies that were developed to serve these wide- area networking applications, but presents a new approach to reconfigurable hybrid interconnects for scientific computing, which utilizes both passive and active switch components that are available on the commodity market.

4.2.3 Related Work

Circuit switches have long been recognized as a cost-effective alternative to packet switches, but it has proven difficult to exploit the technology for use in cluster interconnects because the switches do not understand message or packet boundaries. It takes on the order of milliseconds to reconfigure an optical path through the switch, and one must be certain that no message traffic is propagating through the light path when the reconfiguration occurs. In comparison, a packet-switched network can trivially multiplex and demultiplex messages destined for multiple hosts without requiring any configuration changes.

The most straightforward approach is to completely eliminate the packet switch and rely entirely on a circuit switch. A number of projects, including the OptIPuter [16] transcontinental optically-interconnected cluster, use this approach for at least one of their switch planes. The OptIPuter nodes use Glimmerglass MEMS-based optical circuit switches to interconnect components of the local cluster, as well as to form transcontinental light paths which connect the University of Illinois half of the cluster to the UC San Diego half. One problem that arises with this approach is multiplexing messages that arrive simultaneously from different sources. Given that the circuit switch does not respect packet boundaries and that switch reconfiguration latencies are on the order of milliseconds, either the message traffic must be carefully coordinated with the switch state or multiple communication cards must be employed per node so that the node's backplane effectively becomes the message multiplexor; the OptIPuter cluster uses a combination of these two techniques. The single-adapter approach leads to impractical message-coordination requirements in order avoid switch reconfiguration latency penalties, whereas the multi-adapter approach suffers from increased component costs due to the increased number of network adapters per host and the larger number of ports required in the circuit switch.

There are a number of similar examples of pure circuit-switched networks, such as KLAT-2 [23], that employ multiple cards per node that are connected in to a circuit switch thereby eliminating the need for any layer-2 packet switches. The resulting system is essentially an pure circuit switched network, but the node now acts as the Layer-2 switch rather than having a dedicated switch component. It requires development of heuristic clique-mapping techniques to improve the quality of the mapping algorithm because the underlying problem is NP-complete. For example genetic programming approaches have been used for optimizing the fixed switch topology of the Flat Neighborhood Networks [23] to optimize the embedding. However, genetic algorithms can be expensive and difficult to tune. As a system is scaled up, the likelihood of arriving at a reasonable solution diminishes due to combinatorial explosion

in the size of the search space.

One proposed solution, the ICN (Interconnection Cached Network) [22], recognizes the essential role that packet switches play in multiplexing messages from multiple sources at line rate. The ICN consists of processing elements that are organized into blocks of size k which are interconnected with small crossbars capable of switching individual messages at line rate (much like a packet switch). These k-blocks are then organized into a larger system via a $k * N_{blocks}$ ported circuit switch. The ICN can embed communication graphs that have a consistently bounded topological degree of communication (TDC) less than k. The jobs must be scheduled in such a way that the bounded contraction of the communication topology (that is, the topological degree of every subset of vertices) is less than k. This is an NP-complete problem for general graphs when $k > 2$, although such contractions can be found algorithmically for regular topologies like meshes, hypercubes, and trees. If the communication topology has nodes with degree greater than k, some of the messages will need to take more than one path over the circuit switch and therefore share a path with other message traffic. Consequently the bandwidth along that path is reduced if more than one message must contend for the same link on the network. Job placement also plays a role in finding an optimal graph embedding. Runtime reconfiguration of the communication topology on an ICN may require task migration in order to maintain an optimal embedding for the communication graph. The HFAST approach detailed in this work has no such restriction to regular topologies and needs no task migration. This class of network solution is referred to as as bounded-degree hybrid interconnects (BDHI).

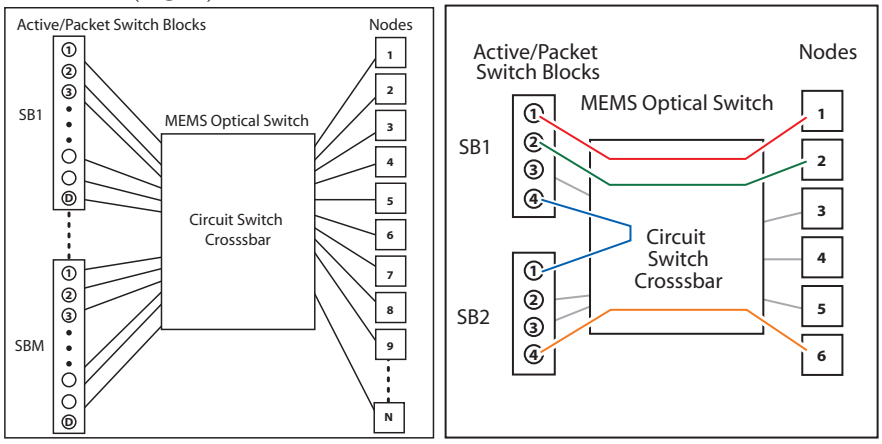
Adaptive routing (AR) offers an alternative approach to reducing link contention in low-degree interconnects. However, the additional logic required for AR greatly increases hardware complexity to achieve the same goal as the HFAST approach. HFAST reduces interconnect link contention by reconfiguring the wiring using simpler circuit switches, whereas adaptive routing makes contention-avoiding such decisions on a packet-by-packet basis. This study make use of a broad array of HPC applications to demonstrate that routing decisions made on a longer timescale, which is amenable to the circuit switch reconfiguration times, offer an efficient approach to reducing hot-spots in a lower-degree interconnect. Overall, HFAST offers lower design complexity and hence a more cost-effective approach to achieving the same capabilities for hot-spot avoidance as AR.

Finally, there are a number of hybrid approaches that use combination packet/circuit switch blocks. Here each switching unit consists of a low bandwidth dynamically-routed network that is used to carry smaller messages and coordinate the switch states for a high-bandwidth circuit switched network that follows the same physical path. Some examples include Gemini [12], and Sun Microsystems Clint [19]. Each of these uses the low-bandwidth packet-switched network to set up a path for large-payload bulk traffic through the circuit switch hierarchy. While the circuit switch path is unaware of packet boundaries, the lower-speed packet network is fast enough to mediate potential conflicts along the circuit path. This overcomes the problems with coordinating message traffic for switch reconfiguration exhibited by the purely circuit-switched approach. While promising, this architecture suffers from

the need to use custom-designed switch components for a very special-purpose use. In the short term, such a specialized switch architecture will have difficulty reaching a production volume that can amortize the initial development and manufacturing costs. The target is to make use of readily available commodity components in the design of an interconnect in order to keep costs under control.

4.2.4 HFAST: Hybrid Flexibly Assignable Switch Topology

Figure 4.5: General layout of HFAST (left) and example configuration for 6 nodes and active switch blocks of size 4 (right).



HFAST is a novel approach to overcoming these obstacles that were outlined in the previous subsection, by using (Layer-1) passive/circuit switches to dynamically provision (Layer-2) active/packet switch blocks at runtime. This arrangement leverages the less expensive circuit switches to connect processing elements together into optimal communication topologies using far fewer packet switches than would be required for an equivalent fat-tree network composed of packet switches. For instance, packet switch blocks can be arranged in a single-level hierarchy when provisioned by the circuit switches to implement a simpler topology like a 3D torus, whereas a fat-tree implementation would require traversal of many layers of packet switches for larger systems – contributing latency at each layer of the switching hierarchy. Therefore this hybrid interconnection fabric can reduce fabric latency by reducing the number of packet switch blocks that must be traversed by a worse-case message route.

Using less-expensive circuit switches, one can emulate many different interconnect topologies that would otherwise require fat-tree networks. The topology can be incrementally adjusted to match the communication topology requirements of a code at runtime. Initially, the circuit switches can be used to provision densely-packed 3D mesh communication topologies

for processes. As runtime data about messaging patterns is measured by the system, the interconnect topology can be adjusted at discrete synchronization points to better match the measured communication requirements and thereby dynamically optimize code performance. MPI topology directives can be used to speed the runtime topology optimization process. There is also considerable research opportunities available for studying compile-time instrumentation of codes to infer communication topology requirements at compile-time. In particular, languages like *Unified Parallel C* (UPC) [59] offer a high-level approach for exposing communication requirements at compile-time. Similarly, the compiler can automatically insert the necessary synchronization points that allow the circuit switches time to reconfigure since the Layer-1 switches do not otherwise respect packet boundaries for in-flight messages.

HFAST differs from the bounded-degree ICN approach in that the fully-connected passive circuit switch is placed between the nodes and the active (packet) switches. This supports a more flexible formation of communication topologies without any job placement requirements. Codes that exhibit non-uniform degree of communication (e.g. just one or few process(es) must communicate with a large number of neighbors) can be supported by assigning additional packet switching resources to the processes with greater communication demands. Unlike the ICN and OptIPuter, HFAST is able to treat the packet switches as a flexibly assignable pool of resources. In a sense, the HFAST approach is precisely the inverse of the ICN – the processors are connected to the packet switch via the circuit switch, whereas the ICN uses processors that are connected to the circuit switch via an intervening packet switch.

Figure 4.5 shows the general HFAST interconnection between the nodes, circuit switch and active switch blocks. The diagram on the right shows an example with six nodes and active switch blocks of size 4. In this example, Node 1 can communicate with Node 2 by sending a message through the circuit switch in switch block 1 (SB1) via the red circuit path, and back again through the circuit switch (green circuit path) to Node 2. This shows that the minimum message overhead will require crossing the circuit switch two times. If the TDC of Node 1 is greater than the available degree of the active SB, multiple SBs can be connected together (via a myriad of interconnection options). For the example in Figure 4.5, if Node 1 was to communicate with Node 6, the message would first arrive at SB1 (red), then be transferred to SB2 (blue), and finally sent to Node 6 (orange) — thus requiring 3 traversals of the circuit switch crossbar and two active SB hops.

The HFAST approach holds a clear advantage to statically built interconnects, since additional packet switch resources can dynamically be assigned to the subset of nodes with higher communication requirements. HFAST allows the effective utilization of interconnect resources for the specific requirements of the underlying scientific applications. This methodology can therefore satisfy the topological connectivity of applications categorized in *cases i-iii* (defined in Section 3.4). Additionally, HFAST could be used to dynamically create fit-trees with static or variable fitness rations. Furthermore, because the circuit switches have allocated a network that matches the application, the network can avoid elaborate dynamic routing approaches that result in greater router complexity and slower routing speed. This

approach avoids job fragmentation, since "migration" is essentially a circuit switch configuration that can be performed at a barrier in milleseconds. Finally, the HFAST strategy could even iteratively reconfigure the interconnect between communication phases of a dynamically adapting application [32]. Future work will continue to explore the potential of the HFAST in the context of demanding scientific applications.

4.2.5 HFAST Baseline Cost Model

Fat-tree and CLOS networks are built in layers of N-port switches such that L layers can be used to create a fully connected network for P processors where $P = 2 * (N/2)^L$. However, the number of switch ports in the interconnection network per processor grows at a rate of $(1 + 2(L - 1))$. So, for instance, a 6-layer fat-tree composed of 8-port switches requires 11 switch ports for each processor for a network of 2048 processors! Messages must traverse up to 21 layers of packet switches to reach their destination. While state-of-the-art packet switches typically contribute less than 50ns to the message latency, traversing 21 layers of them can become a significant component of the end-to-end latency.

With the HFAST solution, the number of ports required for the passive circuit switch grows by the same proportion as a full FCN. However, the cost per port for the circuit switch is far less than the cost per port for a packet switch using a leading-edge technology. Packet switches, the most expensive component per-port, can be scaled linearly with the number of processors used in a given system design. So unlike a fixed topology mesh, hypercube, or torus interconnect, the cost of HFAST is not entirely linearly-proportional to the number of processors because of the cost of the fully connected circuit switch. However, the cost of the most *expensive* component, the packet switches and network interface cards for the hosts, scales proportionally with the number of processors.

A simple cost function is introduced below that represents the applicability of HFAST given the TDC of each node in the computation. To simplify the analysis an upper-bound is presented that does not use any sophisticated graph-theoretic methods to optimize mappings. In addition, homogenous active switch block size of 16 ports is assumed for reference.

Generally, the cost $Cost_{HFAST}$ is given by

$$N_{active} * Cost_{active} + Cost_{passive} + Cost_{collective},$$

where N_{active} is the number of active switch blocks required, and $Cost_{active}$, $Cost_{passive}$, and $Cost_{collective}$ are the respective costs of a single active switch block, the passive switch, and the collective network. HFAST is effective if $Cost_{HFAST} < Cost_{fat-tree}$.

For a given code each node is examined in turn. For each node, if the TDC is less than the active switch block size (in our case 15), it is assigned to an active switch block. However, if the TDC is greater than 15, it is assigned the number of switch blocks needed to build a tree network large enough to communicate with all of the node's partners. This algorithm uses

potentially twice as many switch ports as an optimal embedding, but it has the advantage that it will complete in linear time.

As an example, we determine the cost for Cactus, a code that exhibits an average and maximum TDC of 6 per node. For each node, then, we assign a single active switch block, giving us $N_{active} = P$. That is, the number of active switch blocks required is equal to the number of processors in a run. For codes like PMEMD that exhibit a maximum TDC that is higher than the average, additional packet switch blocks can be provisioned (if available) to construct a higher-radix tree network to support the higher-degree communication pattern required by that subset of processors.

The procedure outlined above creates an efficient mapping when average TDC is less than the switch block size. However, the method yields a far less efficient mapping, relative to a Fat-tree or CLOS network, for codes with higher TDC. The mapping procedure uses the packet switches exclusively for fan-in and fan-out of connections between nodes, and therefore does not exercise the full internal bisection connectivity of these switch blocks.

The general problem of switch block assignment can be reduced to the clique-mapping problem where tightly interconnected cliques are mapped to switch blocks in order to maximize the utilization of internal switch connectivity. The optimal solution to the fully generalized clique-mapping problem is NP-complete [35]. However, the fact that the switch blocks are of finite size bounds the complexity of the problem to less than NP-complete, but it still involves a large search space. The fit-tree approach provides a much more optimal solutions in polynomial time.

4.3 Summary and Conclusions

There is a crisis looming in parallel computing driven by rapidly increasing concurrency and the non-linear scaling of switch costs. It is therefore imperative to investigate interconnect alternatives to ensure that future HPC systems can cost-effectively support the communication requirements of ultrascale applications across a broad range of scientific disciplines. Before such an analysis can be undertaken, one must first understand the communication requirements of large-scale HPC applications, which are the ultimate drivers for future interconnect technologies.

To this end, this chapter has presented one of the broadest studies to date of high-end communication requirements, across a broad spectrum of important scientific disciplines. Analysis of these data show that most applications do not utilize the full connectivity of traditional fully-connected network implementations. Based on these observations, a novel network analysis called a fit-tree was introduced The analysis reveals that fit-trees can significantly improve the cost and scalability of fat-trees, while preserving performance through reduced component count and lower wiring complexity. Finally, the HFAST infrastructure is described, which combines passive and active switch technology to create dynamically

reconfigurable network topologies, and could be used to create custom-tailored fit-tree configurations for specific application requirements. This approach meets the performance benefits of adaptive routing approaches while keeping component counts (and associated cost and power) bounded. Overall results lead to a promising approach for ultra-scale system interconnect design and analysis.

Future work will pursue two major thrusts. The first thrust will expand the scope of both the applications profiled and the data collected through the IPM profiling. The low overhead of IPM profiling opens up the possibility of the characterization of large and diverse application workloads. These studies will enable a more detailed performance data collection, including the analysis of full chronological communication traces. Studying the time dependence of communication topologies could expose opportunities to reconfigure an HFAST interconnect within a dynamically evolving computation. The studies will also have application to interconnect topologies and circuit provisioning for emerging chip multiprocessors (CMPs) that contain hundreds or thousands of cores per socket. The second thrust will continue the exploration of fit-tree solutions in the context of ultra-scale scientific computations. This portion of the investigation will require comparisons with alternative approaches such as high-radix routers, as well as examining the physical aspects of constructing reconfigurable fit-tree interconnects including issues of packaging and cable layout cost and energy models.

Chapter 5

Network on Chip (NoC) Design Study

In the continual drive toward improved computing performance, power efficiency has emerged as a prime design consideration. In fact, the limitations on power dissipation imposed by packaging constraints have become so paramount that performance metrics are now typically measured per unit power. At the chip scale, the trend toward multicore architectures and chip multiprocessors (CMPs) for driving performance-per-watt by increasing the number of parallel computational cores is dominating new commercial releases. With the future path clearly toward further multiplication of the on-chip processing cores, CMPs have began to essentially resemble highly parallel computing systems integrated on a single chip. In this context, the role of the interconnect and associated global communication infrastructure is becoming central to the chip performance. As with highly parallel systems, performance is increasingly tied to how efficiently information is exchanged and how well the growing number of computational resources are utilized. The realization of a scalable on-chip communication infrastructure faces critical challenges in meeting the large bandwidth capacities and stringent latency requirements demanded by CMPs in a power efficient fashion. With vastly increasing on-chip and off-chip communication bandwidths, the interconnect power consumption is widely seen as an acutely growing problem. It is unclear how conventional CMOS scaling of electronic interconnects and networks-on-chip (NoCs) will continue to satisfy future bandwidths and latency requirements within the CMP power budget. The insertion of photonics in the on-chip global interconnect structures for CMP can potentially leverage the unique advantages of optical communication and capitalize on the capacity, transparency, and fundamentally low energy consumption that have made photonics ubiquitous in long-haul transmission systems. The construction of photonic NoC could deliver performance-per-watt scaling that is simply not possible to reach with all-electronic interconnects.

The photonics opportunity is made possible now by recent advances in nanoscale silicon photonics and considerably improved photonic integration with commercial CMOS chip manufacturing. Unlike prior generations of photonic technologies, the remarkable capabilities

of nanoscale silicon photonics offer the possibility of creating highly integrated photonic platforms for generating and receiving optical signals with fundamentally superior power efficiencies. These tremendous gains in power efficiencies for optical modulators and receivers are driven by the nanoscale device footprints and corresponding capacitances, as well as by the tight proximity of electronic drivers enabled by the monolithic CMOS platform integration. Photonic NoCs can deliver a dramatic reduction in power expended on intra-chip global communications while satisfying the high bandwidths requirements of CMPs. Photonic NoCs change the rules of power scaling: as a result of low loss optical waveguides, once a photonic path is established, the data is transmitted end-to-end without the need for repeating, regeneration, or buffering. In electronic NoCs, on the other hand, a message is buffered, regenerated, and then transmitted on the inter-router links multiple times en route to its destination. Furthermore, the switching and regenerating elements in CMOS consume dynamic power that grows with the data rate. The power consumption of optical switching elements, conversely, is independent of the bit rate, so, once generated, high bandwidth messages do not consume additional dynamic power when routed. While photonic technology offers these potentially enormous advantages in terms of energy and bandwidth, there are fundamental limitations to which must be taken into consideration when designing photonic NoCs that can truly exploit these technology gains. Two necessary functions for packet switching NoCs, namely buffering and header processing, are very difficult to implement directly in the optical domain with optical devices. Therefore, new paradigms in networking architecture and circuitry design must be developed to fully exploit and drive future innovations in nanoscale photonic devices.

This chapter explores photonic networks-on-chip architectural solutions for high-performance CMP design which leverages the remarkable progress in silicon photonics to offer a major reduction in the power dissipated on intra-chip communications. The intra-chip photonic infrastructure can also offer seamless off-chip communications. The analysis examines the impact of innovative interconnect micro-architectures that leverage nanoscale silicon photonic and complementary devices developed in synergy with electronics. The interaction between the optical and electrical networks to create these hybrid designs shares many of the same features demonstrated by HFAST for the system-scale interconnection networks.

5.1 Background

The microprocessor industry is set to double the number of cores per chip every 18 months – leading to chips containing hundreds of processor cores in the next few years [2]. This path has been set by a number of conspiring forces, including complexity of logic design and verification, limits to instruction level parallelism and – most importantly – constraints on power dissipation. In this brave new world of ubiquitous chip multiprocessing (CMP), the on-chip interconnect will be a critical component to achieving good parallel performance. Unfortunately, a poorly designed network could easily consume significant power, thereby

nullifying the advantages of chip multiprocessing.

Consequently, there is an urgent need to develop communication architectures that can maintain performance growth under a fixed power budget. Current processor-manufacturing roadmaps point to simple mesh or torus networks-on-chip (NoC) via electrical routers as the medium-term solution; however, previous work [3] has shown that such architectures may not be best-suited for balancing performance and energy usage. This chapter investigates a promising alternative to electrical NoCs, namely architectures that exploit optics for some or all inter-processor communications.

According to the International Technology Roadmap for Semiconductors [30], three-dimensional chip stacking for three-dimensional integration (3DI) is a key focus area for improving latency and power dissipation, as well as for providing functionally diverse chip assemblies. Recent advances in 3DI CMOS technology [5] have paved the way for the integration of silicon-based nanophotonic devices with conventional CMOS electronics, with the premise of realizing hybrid photonic/electronic NoCs [53]. High density through-silicon-vias (TSVs), the critical enabling technology for 3DI, electrically connect wafer layers. One of the fundamental assumptions of this work is that 3D integrated chips will play an important role as the interconnect plane for future chip multiprocessors, whether the NoC is electrical or photonic, and that the TSVs have a minimal impact on the power dissipation for these chip implementations.

Extensive cycle-accurate simulations using custom software within the OMNeT++ framework [61] to evaluate the tradeoffs between the electrical and photonic network designs. This work differs from previous efforts through the use of a comprehensive event-driven simulation allowing us to model the low-level electronic and photonic details of the evaluated interconnect configurations. The modeling detail enables a comprehensive analysis of the energy, latency, and physical performance of the devices under more realistic application workloads. In addition to standard synthetic traffic models, this study utilizes traces of real parallel scientific applications to determine the potential benefits of the hybrid network for Single Program Multiple Data (SPMD) style algorithms.

The simulation environment is used to analyze interconnection networks of various types and configurations for performance and energy consumption. Reported metrics include the execution time of the benchmark/application, the total energy consumed therein, and the energy efficiency, a metric which emphasizes the network performance gained with each unit of energy spent. The performance of electronic mesh and torus topologies are simulated along with the photonic NoC studied in [44], known as a blocking torus (which is referred to as a *photonic torus*). In the photonic NoC, a photonic network and an electronic control network coordinate to provide the system with high bandwidth communications. The simulations show that the photonic interconnects studied here offer excellent power-efficiency for large messages, but are less advantageous for carrying small messages. The results show how different application characteristics can affect the overall performance of the network in ways that are not readily apparent in higher level analysis.

5.2 Related Work

Prior related works have made significant gains in the area of on-chip optical interconnects. Petracca *et al.* investigated Cooley-Tukey FFT traffic patterns on different photonic topologies in [44]. The photonic NoC is described as an electronic control network augmented with a photonic network made up of silicon waveguides and *photonic switching elements* (PSEs). Each PSE, shown in Figure 5.2, is composed of silicon micro-ring resonators that deflect light when polarized. These building blocks are extended to create a broadband circuit-switched 2D torus topology for on-chip communication.

Novel wavelength-routed architectures have also been proposed both for inter-core communications [60] and for off-chip communications [4]. These networks take advantage of wavelength-division multiplexing (WDM) to dedicate wavelengths to destinations in the network. Lower level modeling was performed in [8, 42], which is a good step towards achieving a comprehensive analysis of an architecture, but it has yet to be seen how these networks compare to other competing systems under real workloads.

For electronic CMPs, Dally *et al.* [3] compared several possible NoC topologies using detailed timing, area, and energy models for the network components. Of the explored networks, the best in terms of energy and communication time was a *Concentrated Mesh*, a type of mesh topology that uses larger-radix routers to cluster four processors at each mesh node and contains express channels around the perimeter of the network.

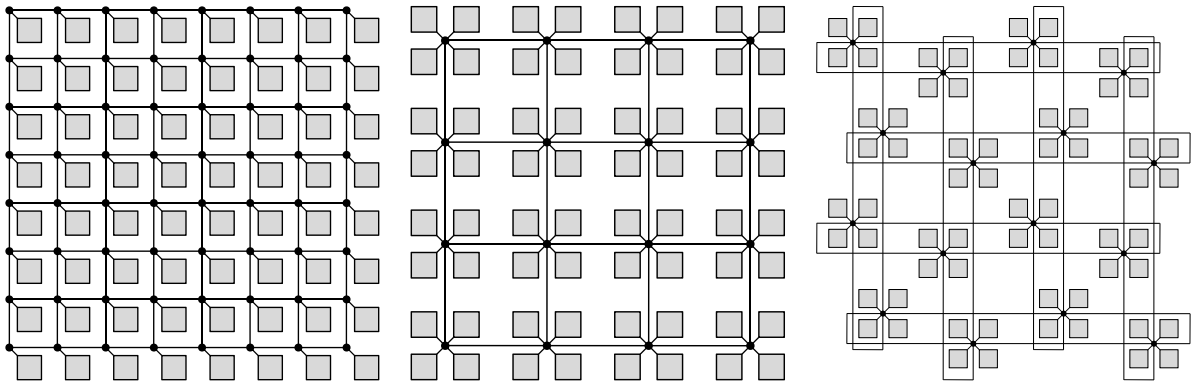
Other work proposing a hybrid interconnection network for multiple processor systems [33] characterized the inter-chip communication requirements for full scientific applications using similar measurement tools. The study found that fully connected network topologies are overprovisioned for most applications and their size grows exponentially with system concurrency. However, mapping application communication topologies onto simpler interconnect topologies such as meshes or tori leads to difficult topology mapping and resource scheduling problems. A hybrid approach that employs optical circuit switches to reconfigure the interconnect topology to match application requirements can retain the advantages of a fully connected network using far fewer components. No timing models were used in this study whose focus was on the mapping of the inter-chip communication topologies rather than performance.

5.3 Studied Network Architectures

This section describes the examined NoC architectures, which includes various networks for both conventional electronic networks and hybrid photonic-electronic networks.

3DI utilizing Thru-Silicon-Vias (TSVs) showcases inherently short interconnect paths with reduced resistance and capacitance, as well as lower power consumption. These characteris-

Figure 5.1: Mesh, concentrated mesh, and concentrated torus topology. The concentrated topologies require a larger-radix switch, but reduce the average hop count.

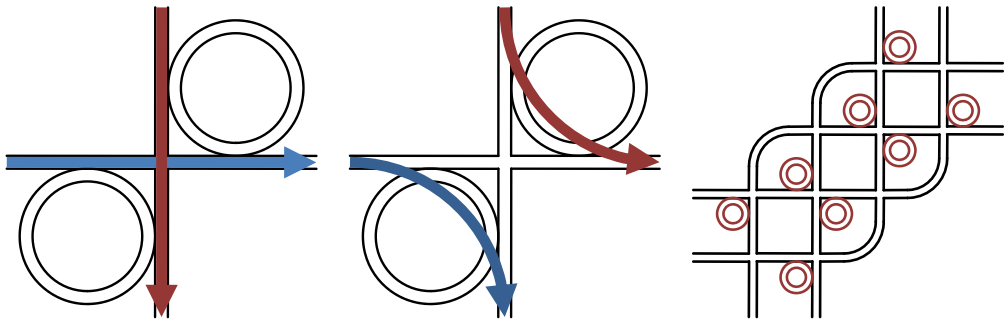


(a) Mesh (b) Concentrated Mesh (c) Concentrated Torus

tics enable the TSV's to enable the switching plane to be integrated on a separate plane of stacked silicon with very low power dissipation for the vias that connect between the planes. For the 32 nm technology node, the TSV is expected to scale to a 1.4 μm contact pitch, 0.7 μm diameter, almost 5×10^7 cm^{-2} maximum density, and 15 μm maximum layer thickness [30]. By stacking memory and interconnect resources on dedicated CMOS layers above the processors, it is possible to integrate larger memories and faster interconnects with future CMPs [53]. Silicon nanophotonic technology may alleviate the limitations of conventional electronic networks by using optics to deliver much higher bandwidth within the same power budget, however it has several inherent limitations, such as the inability to perform buffering and processing in the optical domain, which need to be circumvented in order to take the full advantage of this new technology.

Electrical NoC Architecture. The modeled CMP contains 64 processors arranged in a 2D planar fashion, which is based on the requirements of the Green Flash manycore chip design [17] but also matches emerging commercial manycore offerings including Intel's 48-core Single-Chip Cluster Computer chip [29], Intel's 54 core Knights Corner [51], and the 64-core Tilera [69] chips. Although the processors themselves are not simulated, the behavior is assumed to be consistent with simple in-order cores with local store memories. The individual core size is $1.5mm \times 2.0mm$; the cores are located on the lowest layer of the 3DI CMOS die. Above the bottom layer are multiple layers devoted to the local store, allowing the cores sufficient capacity to feed computational units. Lastly, the top layer is where the global NoC is found. This consists of the electronic routers, and for the systems that include a photonic NoC, silicon nanophotonic components.

Figure 5.2: Photonic Switching Element. (a) Message propagate straight through. (b) Light is coupled into the perpendicular path. (c) A combination of eight ring resonators allows the construction of a 4×4 nonblocking optical switch.



(a) Off state (b) On state (c) 4 × 4 switch

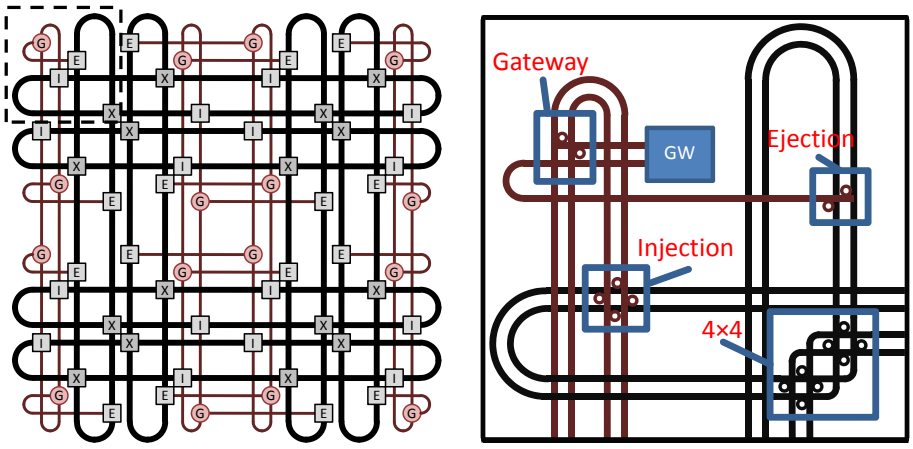
Photonic NoC Architectures. For the electrical network, the topologies shown in Figure 5.1 are modeled. The mesh topology is the baseline for the comparisons against all of the other studied networks. In comparison to more exotic electronic networks, the mesh is simple to implement due to its use of relatively low radix switches in a regular 2D planar layout.

This analysis incorporates the concept of concentrating processing cores at a network node, originally explored in [3]. For example, a full mesh would include an access point for each node, creating an 8×8 mesh. By concentrating a set of four nodes together, the size of the mesh can be reduced to 4×4 thereby reducing the average hop count each message must incur but increasing the radix of each router to accommodate the four node connections. The concentrated mesh and concentrated torus, shown in Figure 5.1 (b) and (c) are studied.

While the mesh network, which is used by Tilera for example, has the advantage of relatively simple routers compared to the CMesh, due to the latter's need for a larger radix switch (which can potentially consume more energy), the average number of links traversed in the CMesh is lower — leading to significantly better performance. However, the CMesh can also suffer from increased bandwidth contention and comparatively low bisection bandwidth. Each router is wormhole routed and the network supports virtual channels to eliminate deadlock and improve performance. For the implementation of the electrical NoC, there is no optical layer above the local memory layers on-chip. Recent work [3] also explores multiple electrical networks; for this paper, however, we assume a single electrical network. Also, concentrated topology is assumed to connect four processing cores, but without the express channels that are present in the CMesh in Dally's paper. So, unlike the concentrated networks in [3], the selected topologies do not contain express channels between non-adjacent switches.

The photonic NoC is composed of two layers on the top plane of the 3DI structure, a photonic

Figure 5.3: The photonic torus topology shown in (a) was developed by the Columbia University Lightwave Research Laboratory (LRL), and studied in [44]. Switch blocks are abbreviated: X - 4 × 4 nonblocking, I - injection, E - ejection, G - gateway. (b) is a zoom in of the dotted box in (a), which shows a single node in the photonic torus. The node(s) are connected to the gateway (GW) and the boxed areas represent switches used to control optical paths through the network.



(a) Photonic Torus (b) Switch Element

layer and an electronic control layer. The photonic layer provides a high bandwidth network for transmitting data and is constructed using silicon nanophotonic ring resonator structures that can be switched to control the propagation of optical signals (Figure 5.2). The electronic control layer is a secondary network used to transmit and act on control packets for the purpose of setting up and breaking down photonic links on the photonic layer. The control layer can also be provisioned as a low bandwidth network for transmitting small amounts of data.

Switching functionality on the photonic layer is derived from the use of ring resonator structures that act as PSEs, as in [44]. In Figure 5.2(a), the PSE is shown in the off-resonance state where messages propagate straight through the switch. Figure 5.2(b) shows the on-resonance state of the PSE, which bends the optical pathway implementing a turn. A control system is fabricated along with the switch to enable active switching of the device. The PSE models are implemented with the on-resonance state dormant, where no electrical current is applied, while the off-resonance state draws current to change the behavior of the device. By combining several PSEs together, functional network components such as the 4 × 4 nonblocking switch shown in Figure 5.2(c) can be created.

As described in [44], the main network structure of the topology is a folded torus shown as black lines in Figure 5.3(a). Included on the same topology is an additional set of waveguides and switches, shown as red lines, that are used to inject and eject optical messages into and

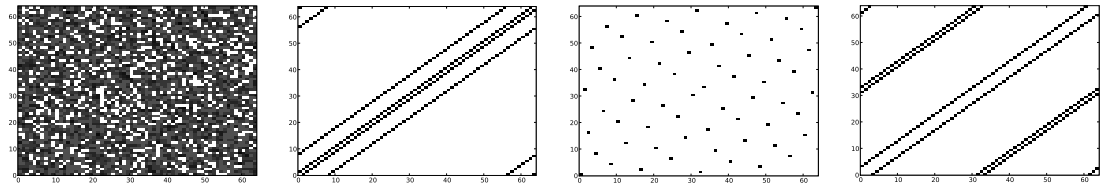
from the network. Typically, this network provides a single access point for each processing node; however, variations of this network are included with the with concentrated nodes, as previously described.

The transmission of data on the photonic network is enabled through the use of circuit switching, which requires the provisioning of an optical path before any data can be injected. The *path-setup phase* begins by sending a electronic setup control packet in the control layer, which travels through the network, establishing an optical path by configuring the appropriate PSEs. Once the setup packet reaches the destination node, the complete optical path has been allocated and an electronic acknowledgment is returned — allowing the source to begin data transmission upon receipt. The *breakdown phase* occurs upon complete transmission of data, where a breakdown control packet is sent along the network to release the optical path.

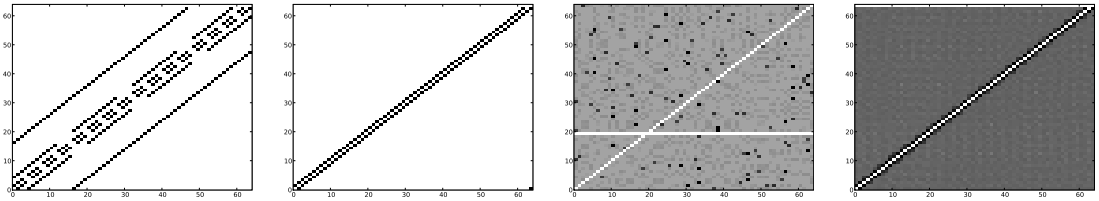
Figure 5.3(b) shows a detail view of the required photonic components needed to transmit and receive messages on the photonic NoC. The processing node (or nodes, for the concentrated configuration) injects messages electrically to the *gateway*, marked *GW*. Upon receiving an acknowledgement packet for a setup request, the gateway begins transmitting the message optically. The message first propagates through a *gateway switch*, which handles the routing of messages going to and from the gateway. Next, the message is directed towards the *injection switch* where it is switched into the torus network. The message then propagates through the torus (using dimension-ordered routing) until it reaches the correct turning point where it turns at a *4×4 nonblocking switch*. Once at the destination, the message exits the network via the *ejection switch*, and is directed to the gateway by the gateway switch where it is converted to an electronic signal and forwarded to the proper node.

Selective Transmission. Networks that transmit data exclusively on a photonic network ideally should favor large message sizes so that the path-setup overhead is sufficiently amortized over the transmission time of the entire message. Applications that send many small messages are subject to the full penalty of the path-setup overhead and will see substantially lower performance. This study also includes a *selective transmission* configuration of the photonic NoC that leverages the use of the electronic network as a low bandwidth data transmission medium. This configuration filters the packets using a size threshold, and transmits the data along the network that is most appropriate. A preliminary study using random traffic indicates a cross-over point of 256 bytes where transmitting smaller packets over the electronic control layer results in better performance and energy efficiency than using the photonic network alone.

Figure 5.4: Spyplots for the synthetic traces (top) and a selected subset of applications studied in Chapter 3 (bottom).



(a) Random (b) Neighbor (c) Bitreverse (d) Tornado



(e) Cactus (f) GTC (g) MADbench (h) PARATEC

5.4 Studied Benchmarks

This work extends related work by utilizing two sets of benchmarks: both standard synthetic traffic patterns and scientific application traces. Whereas the synthetic benchmarks help to identify the kinds of traffic best suited for each architecture, the application-based communication traces put real scientific workloads on the networks and test different mapping parameters. Figure 5.4 shows the spy plots of the eight benchmarks in this study. These plots illustrate the communication volume between each set of processors: a white square at the coordinate (p_i, p_j) in the plot represents no communication, while darker shades of gray represent increasing volumes of communication between two given processors. Details of the different benchmarks are given in Table 5.1.

Synthetic Benchmarks. The NoC testbeds are compared using four standard synthetic benchmarks from the literature [15], shown in the top of Figure 5.4. For each synthetic messaging pattern, two instances of the test are run: one with small messages and another with larger messages. Because of the restrictions of the hybrid interconnect studied, message transmissions are modeled as follows: each processor sends its messages as fast as possible, but blocks until receiving an acknowledgment from the destination processor before sending the next message.

In the *Random* test, each processor sends several messages to destinations chosen uniformly

at random, independently from the previous destinations. *Neighbor* is a standard test where each processor sends messages to its neighboring processors in the physical two-dimensional topology of the NoC. The last two synthetic messaging patterns are designed to stress two-dimensional NoC topologies: the communication of the *Bitreverse* pattern requires each processor to send a message to its corresponding bitreversed address, involving traversals to far regions of the network. Lastly, *Tornado* is a pattern designed to stress 2D meshes by having each processor communicate to its neighbor's neighbors; the idea is to "shift" the communication of the Neighbor pattern in an adversarial way.

Each of the synthetic benchmark traces are generated from their descriptions in the literature using Python scripts.

Application-Based Benchmarks. A novel contribution of this research is the use of actual application communication information for the simulation of network performance. The simulation trace data collection required a custom-designed profiling interface, used along with Linux's library preloading feature to overload the communication functions, thus keeping track of all function calls in an efficient, fixed-size array. At the end of application execution, the trace data was output to a separate file for each process, and the files are later combined to create the input data for the Omnet++ [61] network simulation. In order to accurately approximate communication behavior without including computation time, the trace tools order the communication into "phases" that are composed of sets of communications that must complete before further communication; essentially, the point-to-point synchronizations inherent in message passing were used to build an ordering of the communication.

This design study relies on a selected subset of the same SPMD applications used in Chapter 3 to understand macro-scale requirements for system-scale interconnection networks. For

Table 5.1: Benchmark Statistics

Benchmark	**Num Phases**	**Num Messages**	**Total Size (B)**	**Avg Msg Size (B)**
Random-Small	1	6400	614400	96
Random-Large	1	6400	819200000	128000
Neighbor-Small	1	6400	614400	96
Neighbor-Large	1	6400	819200000	128000
Bitreverse-Small	1	6400	614400	96
Bitreverse-Large	1	6400	819200000	128000
Tornado-Small	1	6400	614400	96
Tornado-Large	1	6400	819200000	128000
Cactus	2	285	7296000	25600
GTC	2	63	8177148	129796
MADbench	195	15414	86516544	5613
PARATEC	34	126059	5457332	43.3

this part, the traces were downsized to match expectations for chip-scale communication required for future manycore processors in 22nm technology. The parallelization style of these applications is an ideal starting point for this study, because of their easily understandable synchronous communication model and their wide use in the scientific programming community.

The applications used in this study are Cactus, GTC, PARATEC, and MadBench. Together, these four applications represent a broad subset of scientific codes with particular communication requirements both in terms of communication topology and volume of communication. For example, the nearest-neighbor Cactus communication represents components from a number of applications characterized by stencil-type behavior. Thus, the results of our study are applicable to a broad range of numerical computations.

5.5 Simulation Methodology

Columbia University developed a comprehensive simulation framework capable of capturing key low-level physical details of both optical and electronic components, while maintaining cycle-accurate functional modeling using event-driven execution to achieve low-overhead simulation. This simulation framework was used to conduct the simulation of the NoC performance and energy for this work [25, 44]. The core framework is implemented in the OMNeT++ environment [61], which consists of around 25k lines of code, many of which are dedicated to specifying the detailed layout of photonic devices. Though OMNeT++ enables a modular construction and hierarchical instantiation of components, subtle differences in spatial positioning and orientation require some manual configuration of each network.

The photonic hybrid networks under consideration here are all multi-wavelength circuit switched. Path setup messages are sent on the electronic network to establish end-to-end optical links between communicating pairs. Once optical transmission is complete, paths are torn down in the same fashion. Network topology and communication patterns therefore have a large effect on overall performance and power because of path setup congestion and blocking. Details of the implementation are discussed below.

5.5.1 Electronic Modeling

The electronic NoC, which is studied as a network for comparison, is modeled cycle-accurately. Electronic components, which pertain to both the electronic NoC and the electronic control plane of the photonic networks, are discussed below, followed by the photonic devices.

Processing Cores. Trace files captured from evaluated benchmarks (Section 5.4) are read into a processing core model that injects messages into the network. Messages are injected

as quickly as possible for each messaging phase, once the core is finished with previous communication. This simulates the bulk-synchronous style of communication employed by the studied applications. Likewise, the destination processors take flits out of the network as soon as they arrive, under the assumption that the processor is not busy performing other computation or communication. This methodology is used to stress the network, illustrating the effects of having many messages in-flight. The trace files keep track of individual messaging phases in the application. Explicit small synchronization messages are sent to and from a master core, which enforces barriers between application phases.

Note that the application communication patterns are highly dependent on process placement on the interconnect. Often, the canonical mapping of processes onto the NOC will result in anomalously poor communication performance. However, identifying the optimal mapping has proven difficult and unreliable [33]. Furthermore, in a real system that stochastically schedules and retires parallel applications on the NoC, it may not be possible to choose an ideal process mapping. Therefore, data was collected from a number of trials that use random process placements to build up a statistical view of the responsiveness of the interconnect to the application's communication requirements. The statistical approach was not applied to the synthetic benchmarks because they were intended to subject the interconnect to a carefully crafted set of exercises, and therefore statistical responses would not make sense.

In addition, communication elements are randomly assigned to cores in the network for the application data, to decrease the likelihood of a trace producing especially poor results by exploiting a single aspect of the network — a common artifact in real scientific computing. Each simulation is run fifty times with different mappings for each trace and topology, and the min, max, and average are subsequently collected. This randomization is not performed for the synthetic communication patterns because they are intended to stress specific aspects of the physical NoC layout. For convenience, the synthetic patterns were generated by the simulation rather than using trace data from an application.

Routers. The router model implements XY dimension ordered routing with bubble flow control [45] for deadlock prevention and to avoid overrunning downstream buffers. Additionally, the routers are fully pipelined with four virtual channels and can issue two grant requests in a single cycle. For power dissipation modeling, the ORION electronic router model [66] is integrated into the simulator, which provides detailed technology-specific modeling of router components such as buffers, crossbars, and arbiters. The technology point is specified as 32 nm. Buffer sizes, shown in Table 5.2, are determined through preliminary experiments that identify optimal power-performance tradeoffs for each implementation to enable a fair comparison between electronic and photonic networks. In general, purely electronic networks have larger buffers and channel widths to increase their performance. This involves an important tradeoff with power consumption, making it necessary to gauge efficiency and not merely performance or power, which will be discussed further in the analysis

Table 5.2: Electronic Router Parameters

Topology	Channel Width	Buffer Size (b)
Electronic Mesh	128	1024
Electronic Concentrated Mesh	128	2048
Electronic Concentrated Torus	128	2048
Photonic Torus	32	512
Selective Photonic Torus	64	1024
Photonic Concentrated Torus	32	1024
Selective Photonic Concentrated Torus	64	2048

of the results obtained. The concentrated networks also have larger buffers, presuming that this is appropriate given the smaller network size. Finally, the photonic networks using the *Selective* message filter have larger buffers to accommodate the electronic traffic that is allowed to travel on the interconnect.

Wires. The detailed wire model is based on data collected for various wire lengths with different numbers of repeaters, running at 5 GHz with double pumping. This allows us to optimally buffer wires for power dissipation (around 50 fJ/bit/mm), which dictates the wire latency. Individual wire lengths are calculated using core size, router area (calculated by ORION), number of routers, and topology.

Photonic Devices. Modeling of optical components is built on a detailed physical layer library that has been validated through the physical measurement of fabricated devices at Cornell and Columbia University [21, 52, 53]. The modeled components are primarily fabricated in silicon at the nano-scale, and include modulators, photodetectors, waveguides (straight, bending, crossing), filters, and PSEs consisting of ring resonators. These devices are characterized by attributes such as insertion loss, extinction ratio, delay, and power dissipation. Table 5.3 shows the optical parameters used [36, 65], excluding insertion loss and extinction ratio for brevity. Devices are sized appropriately and laid out into a network topology, which is controlled by the underlying electronic network.

5.5.2 Photonic Modeling

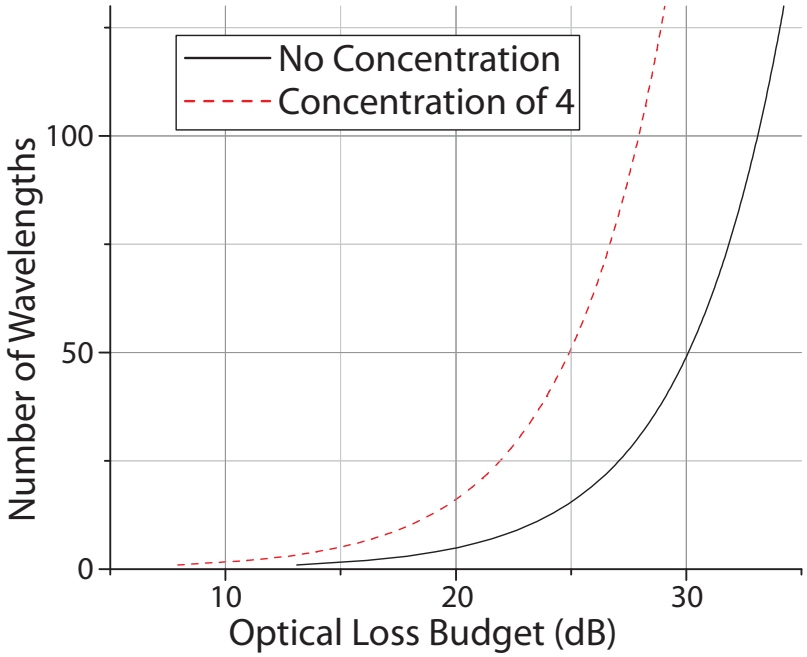
A key parameter for the photonic devices, which greatly affects network performance, is the number of allowable wavelengths. This number is ultimately constrained by network size, since larger networks will exhibit a greater network level insertion loss [13]. The upper limit on available source power is the non-linear threshold of the ring resonators, while the lower limit in received power is dictated by the sensitivity of the photodetectors. An important

Table 5.3: Optical Device Parameters

Sim Parameter	Value
Data rate (per wavelength)	10 Gb/sec
PSE dynamic energy	375 fJ[1]
PSE static (OFF) energy	400 uJ/sec[2]
Modulation switching energy	25 fJ/bit[3]
Modulation static energy (ON)	30 μW[4]
Detector energy	50 fJ/bit[5]
Wavelengths (8×8 network)	65
Wavelengths (4×4 conc. network)	128

advantage of the detailed simulator is the ability to perform this physical layer analysis, as shown in Figure 5.5, which determines the number of wavelengths available at different power budgets for a 64-core photonic torus. It has been determined empirically that 65 wavelengths can be used for the normal 8×8, and 150 for the 4×4 concentrated network for an optical power budget of 35 dB. The maximum number of wavelengths is limited to 128, considering space limitations on laser delivery to the modulators.

Figure 5.5: Insertion loss analysis of Photonic Torus topology.



5.6 Results

The performance characteristics of the selected NoC implementations are now evaluated using the synthetic and application traces. The synthetic benchmarks provide a high-level picture of the interconnect's responsiveness to different commonly-observed communication patterns, while the application traces give insight to performance under realistic scientific loads.

The reported metrics are as follows: (1) performance is analyzed via the execution time of the benchmark or application, (2) energy cost by the total energy spent in execution, and (3) energy efficiency by the performance gained from each unit energy. Note that while typical network comparisons use message latency as a performance metric, such analysis would underscore the true performance of the system by only examining the transmission speed of single streams of data. Because the execution times and energies of the benchmarks varies broadly, the results are normalized to the electronic mesh performance. The electronic mesh was selected as the baseline because it represents the most straightforward engineering approach to interconnecting cores for emerging manycore processor designs.

Recall that the scientific application experiments are conducted using fifty random process placements to develop a statistical view of the networks responsiveness to varying communication mappings (see Section 5.5). Application results are therefore shown using the average performance, with error bars indicating min and max behavior.

Network Speedup. Figure 5.7 presents the application execution time speedup achieved by the examined NoC architectures relative to the execution time of the baseline electronic mesh. Values start at one, which indicates even performance with the baseline. For the synthetic tests with small messages, which are shown in Figure 5.7 (a), the photonic networks without selective transmission do not show improved performance, because the setup messages result in increased latency that is not sufficiently amortized by the high bandwidth end-to-end transmission of the photonic network. The selective transmission shows improvement, but does not gain in speedup over the electronic mesh due to the increased number of routers in the hybrid network used for injection and ejection (see Figure 5.4(a)(b)). The synthetic tests with large messages, which are displayed in Figure 5.7 (b), show a significant improvement for the hybrid photonic networks, compared to what is observed for the experiments conducted on small messages. This illustrates the benefit of amortizing the setup overhead for purely circuit-switched photonic networks. Additionally, it is interesting

[1]Dynamic energy dissipation calculation based on carrier density, assuming 50-μm micro-ring diameter, 320-nm $\times$ 250-nm micro-ring waveguide cross-section, 75% waveguide volume exposure, 1-V forward bias.

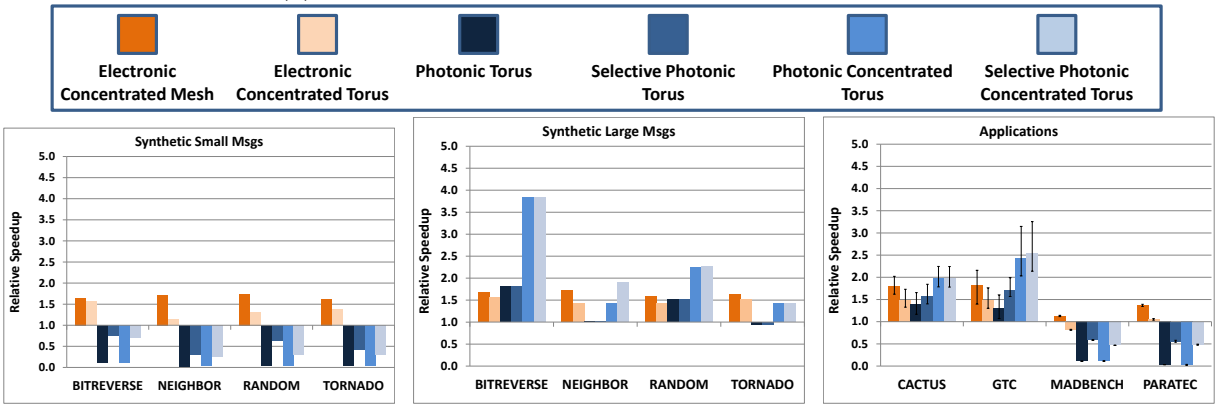
[2]Based on switching energy, including photon lifetime for re-injection.

[3]Same as [1], for a 3μm ring modulator.

[4]Based on experimental measurements in [68]. Calculated for half a 10GHz clock cycle, with 50% probability of a 1-bit.

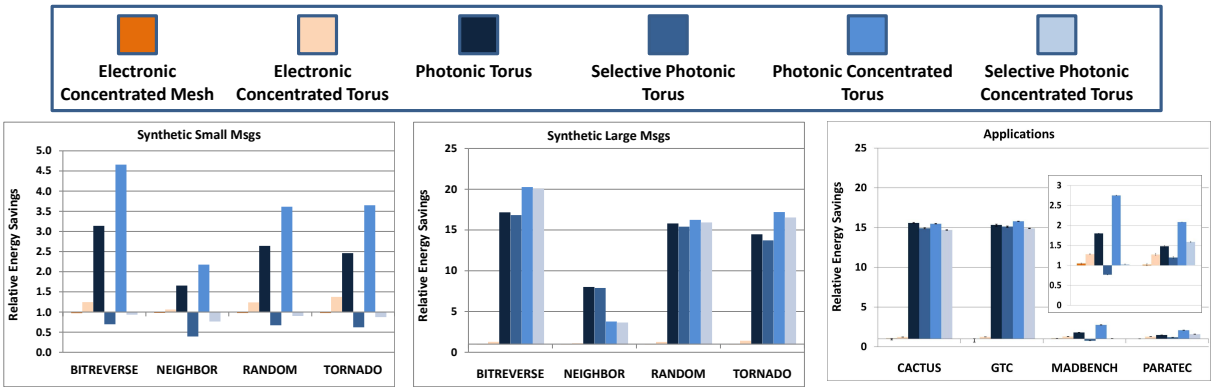
[5]Conservative approximation assuming femto-farad class receiverless SiGe detector with $C < 1fF$.

Figure 5.6: Energy savings relative to electronic mesh. MADbench and PARATEC shown in inset for clarity in (c).



(a) (b) (c)

Figure 5.7: **Network speedup relative to the electronic mesh.**

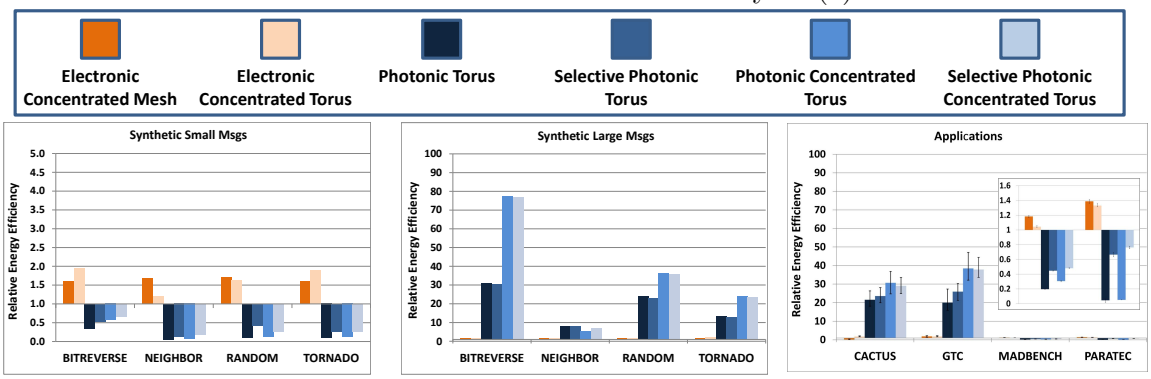


(a) (b) (c)

to note the improvement for the Bitreverse benchmark, which exhibits significantly longer communication patterns, in that circuit-switching directly improves the performance by mitigating contention on a one-time basis. Recall that the effective bandwidth of the photonic network only matches that of the electronic ones when the photonic network is concentrated ($128\lambda \times 10$Gbps vs. 128 channel width $\times$ 5GHz double pumped), which is why they perform significantly better than their full-network counterparts. However, the *Selective* method overcomes the limitations of the photonic NoC by adaptively choosing to route the smaller messages over the electronic network, resulting in speedups of up to 2.3×.

Figure 5.7 (c) shows the relative speedup of the real application traces. The concentrated photonic networks clearly outperform the other interconnect configurations for both Cactus and GTC, similar to the synthetic large-message traces. The photonic networks do not perform as well for the MADBench and PARATEC applications primarily because those benchmarks exhibit all-to-one and broadcast communication patterns, which are expected

Figure 5.8: Energy efficiency (network performance per unit energy) relative to the electronic mesh. MADbench and PARATEC shown in inset for clarity in (c).



(a) (b) (c)

to behave poorly in circuit-switched networks. For these types of applications, wavelength-routed inter-core networks would likely be more appropriate, and future work investigating the use of both circuit-switched and wavelength-routed photonics is under way. In addition, these two benchmarks use significantly smaller message sizes (see Table 5.1). The selective networks narrow the performance difference somewhat, but still do not achieve the nominal performance of the electronic mesh network, similar to the synthetic traces using small messages.

Energy Consumption. Figure 5.6 presents the results of the metric of total energy consumption; the plot shows the inverse of consumption (i.e. the energy savings), again relative to the electronic mesh baseline. The photonic networks are clear winners for most experiments — particularly the large-message synthetics as well as Cactus and GTC applications — showing over 10× improvement due to the decoupling of distance, bandwidth, and power during optical transmission. Since the circuit-switched photonic network does not consume power per-hop, the energy usage is much lower than the packet-switched electrical networks, which require energy consumption in order to make routing decisions at each hop. This point is particularly illustrated again in the Bitreverse benchmark. Because photonics is completely decoupled from distance travelled with repect to energy spent during transmission, it will provide higher benefits when communication pairs are further apart.

However, the photonic approach consumes more energy for the MADBench and PARATEC codes, as seen in Figure 5.6 (c), which is directly related to the use of very small messages. PARATEC, for example, requires many small messages to be sent to every core to implement a 3D transpose for an FFT – resulting in extensive blocking as path setups are repeatedly requested by a given node. Additionally, path blocked messages are sent back to the node when it is determined that a path is unavailable, which can lead to contention in

the electronic control network. This characteristic of the setup-block protocol is useful for preventing deadlock in a circuit-switched network, but in this case generates a substantial number of control messages. This messaging overhead ultimately dissipates a large amount of energy, making it increasingly difficult to overcome these overheads. As a result, the energy consumption can be twice as high even on the selective networks.

Another interesting result is shown in the discrepancy between the photonic networks with and without selective transmission for the traces with large message sizes. One would expect these two networks to perform the same in these conditions, since all messages are large enough to be selected for photonic transmission. Referring to Table 5.2, however, the electronic parameters for the selective networks are twice as large. This was done so that the selective networks could accommodate electronic traffic while not hindering path setup requests. This difference in buffer sizes, while saving us network performance for the small messages, ultimately causes more energy consumption by allowing more path setup congestion.

Performance for Energy Spent. Figure 5.8 shows the final metric: performance gained for every unit of energy spent, which is effectively a measure of a network's efficiency. This metric is calculated by multiplying the network execution time by the energy spent (plotted as the inverse so that values greater than 1 indicate a better performance per energy). The numbers are shown relative to the electronic mesh.

The benchmarks with small messages perform poorly on photonic networks, as seen in Figure 5.8 (a). Although network speedup is reasonable for some photonic networks in Figure 5.7, and energy gains are achieved for some photonic networks in Figure 5.6, the overall network performance is not improved over the electronic mesh when message sizes are small.

However, as shown in Figures 5.8 (b) and (c), the photonic networks' energy efficiency improvement over the electronic mesh for traces with large message sizes is amplified by the gains in both speedup and energy, resulting in improvements of over 20×. This benefit is realized over a variety of communication patterns, including two of the real applications, which demonstrates the possible appeal of on-chip photonics for many classes of applications.

5.7 Conclusions and Future Work

This work compares the performance and energy characteristics of electronic and photonic NoCs using a suite of synthetic communication benchmarks as well as traces from SPMD-style scientific applications on a detailed simulation framework. The analysis shows that a hybrid NoC has the potential to outperform electrical NoCs in terms of performance, while mitigating the power/energy issues that plague electronic NoCs when the communications are sufficiently large to amortize the increased message latency. For messaging patterns with

small messages and high connectivity, the current photonic network design does not perform as well as an electronic mesh, although parameter searches may mitigate this by sizing queues and message size cutoffs to enable better performance in the selective approach.

The comprehensive and detailed level of simulation as well as the range of applications and topologies investigated achieves interesting results that are not possible using a higher-level analysis. These observations will be important in guiding future CMP engineers who seek to design an interconnect architecture that does not become the bottleneck for performance or energy. As future architectures scale to even higher concurrencies, the power requirements and performance benefits of photonic interconnects will become increasingly attractive.

Although these results have addressed some questions about how different applications would behave on different NoCs, it also raises a number of concerns that will lead to important future studies. This work focuses completely on the interconnection network and does not account for data transfer onto the chip from DRAM, nor does it account for computing performance. Furthermore, it is not clear how the performance and energy consumption of the networks fit into overall system performance and energy, and how communication can be overlapped with computation more efficiently. These experiments are currently being pursued for future work.

Alternative topologies for both electronic and photonic networks must also be explored. Photonic network architectures that exhibit less blocking under heavy loads have been proposed in related work, and will be examined in detailed future studies. Many methods of improving electronic interconnect performance are also emerging that may substantially change the comparison between photonic and electronic NoCs.

A key contribution of this work was the focus on SPMD style applications found in the scientific community. Although many elements of these algorithms are finding their way into consumer applications such as realistic physics for games, and image processing kernels, future studies will also explore applications with more asynchronous communication models. Future work will provide a deeper examination of the differences between message passing and shared memory applications and how they interact with both photonic and electronic networks characteristics.

Chapter 6

Conclusions and Future Work

6.1 Summary

This research demonstrates a new application-driven approach to interconnect design that makes a number of unique contributions. To this end, we have presented one of the broadest studies to date of high-end communication requirements, across a broad spectrum of important scientific disciplines, whose computational methods include: finite-difference, lattice-Boltzmann, particle-in-cell, sparse linear algebra, particle mesh ewald, and FFT-based solvers to guide our design decisions for advanced ultrascale interconnects. Analysis of these data show that most applications present sparse messaging topologies that underutilize a fully connected network. Based on these observations, a novel network analysis called a fit-tree was introduced. The analysis reveals that fit-trees can significantly improve the cost and scalability of fat-trees, while preserving performance through reduced component count and lower wiring complexity. Finally, the HFAST infrastructure is described, which combines Optical Circuit Switches to reduce the number of Optical-Electrical-Optical (OEO) transitions for any given network path. HFAST creates dynamically reconfigurable network topologies are used to create custom-tailored fit-tree configurations for specific application requirements. This approach meets the performance benefits of adaptive routing approaches while keeping component counts (and associated cost and power) bounded. Overall results lead to a promising approach for ultra-scale system interconnect design and analysis for system scale interconnects. Finally, we take the same principles of mixed circuit-switched and packet-switched networks to the design of scalable silicon-photonic Network-on-Chip (NoC) implementations. The analysis shows that a hybrid NoC has the potential to outperform electrical NoCs in terms of performance, while mitigating the power/energy issues that plague electronic NoCs when the communications are sufficiently large to amortize the increased message latency. For messaging patterns with small messages and high connectivity, the current photonic network design does not perform as well as an electronic mesh, although parameter searches may mitigate this by sizing queues and message size cutoffs to

enable better performance in the selective approach. Overall, this research has demonstrated a common approach to developing effective interconnects that span the range from chip-scale to system-scale.

6.2 Future Work

There are a number of directions to expand the understanding of ultrascale interconnect technology. The first direction focuses on considering memory devices as peers on the global interconnect fabric rather than devices direct-attached to the node. The second focuses on developing new protocols for on-chip communication that take better advantage of the unique capabilities of optical NoC designs.

6.2.1 Unified Memory/Interconnect Fabric

Scalable optical interconnects based on OCS and packet switch components will likely revolutionize system scale networks, but the same component technologies will also play a central role in the future of memory technology. If the same links developed for connecting nodes to each other are employed to connect nodes to memory. Current electrical memory interfaces must be tightly integrated with the node due to the difficulty of implementing long-haul copper links, but the distance-independent bandwidth offered by optics enable us to reconsider the placement of memory relative to nodes and also the notion of whether off-chip links should be dedicated to one type of communication (memory vs. inter-node messaging), or if they should be treated as a unified *fabric* that can be re-provisioned based on instantaneous need using the HFAST/OCS approach.

The Network Interface Controller (NIC) is the gateway from the node to the system level network, and the NIC architecture can have a large impact on the efficiency with which communication models can be implemented. At a very high level, the network can be thought of as an extension of the memory subsystem; data movement across the network generally starts and ends in the memory subsystem. As such, the network can also benefit from the proposed advances to the memory architecture. Global Address Space communication models have unique challenges that are enforced by the NIC, but may be more efficiently provided by the memory system. For example, it is possible to provide support for atomic operations by including an atomic functional unit and cache on the NIC. However, this implies that local access to the atomic region much go through the NIC in order to maintain coherency. This is a case were it would be highly beneficial to have the memory system itself provide support for atomic operations. The question then becomes how to provide access to the capability to the NIC.

Future work will study the impact of the proposed memory operations designed for use by the local processors and determine if they can provide benefit to remote operations as well.

An important aspect of the research will be to determine how to expose these features to the NIC, as well as research enhancements to the NIC architecture that can work synergistically with the new memory operations to provide even more benefit. A central component of the research will be to compare the relative cost and benefits of different methods of integrating the NIC with the memory and processor. For example, traditionally a NIC is attached to the CPU via an I/O bus such as PCIe. System-on-Chip (SoC) packaging offers the possibility of attach the NIC via a processor bus, which would allow the NIC to be a peer in the node memory coherency protocol. With the Silicon Photonics options described in Chapter 5, the NIC could be organized to directly access to the memory subsystem through optical links. Each of these integration methods will enable a different set of architectural features, allowing a trade-off of complexity, performance and energy efficiency.

6.2.2 NoC Interprocessor Communication

Cache coherence protocols tend to require many broadcast operations and additional redundant data traffic, which works against the strengths of silicon photonics technology. In this research thrust we would consider alternative inter-processor communication protocols that would map to known high-level language semantics that would interact better with OCS technologies to achieve better performance and efficiency. Photonic NoC's prefer to see longer sustained data flows to achieve their best performance. Not only do cache-coherence protocols work against the expression of large sustained data flows, it is unclear if such protocols can scale in an energy-efficient manner to the extremely large core counts anticipated for future chip architectures.

The center of this research thrust is to consider alternative approaches to interprocessor communication that map to high-level semantics that can be expressed elegantly in high-level programming languages. One such option is to define hardware support for message passing interfaces between cores to enable more explicit application control over communication. The other approach is to apply the Partitioned Global Address Space (PGAS) model to support a more implicit model for inter-core communication that maps to existing PGAS languages such as Coarray Fortran (CAF) and Unified Parallel C (UPC). The PGAS approach offers the possibility of improved energy efficiency of the direct inter-core message queues with the convenience of implicit communication. The approach would require extra fences for establishing a consistent memory state. However, it can achieve substantial benefits over the cache-coherent approach because coherence, which expends extra energy to enforce memory consistency at all times – even when it is not necessary for the algorithm correctness.

Different choices for chip-scale interprocessor communication result in dramatically different costs in power, energy efficiency, and chip logic complexity. Like the previous studies, we will an application-driven approach to compare the effectiveness of these different approaches to on-chip interprocessor communication for a number of different interconnect topologies and optical/electrical design options.

6.3 Conclusion

Addressing the technology challenges discussed in this report and accelerating the pace of technology development will require focused investments to achieve exascale computing by 2018. Achieving an exascale level of performance by the end of the decade will require applications to exploit on the order of a billion-way parallelism provided by an envisioned exascale system. This is in sharp contrast to the approximately quarter million-way parallelism in todays petascale systems. Node architectures are expected to change dramatically in the next decade as power and cooling constraints limit increases in microprocessor clock speeds. Consequently computer companies are dramatically increasing on-chip parallelism to improve performance. The traditional doubling of clock speeds every 18-24 months is being replaced by a doubling of cores, threads or other parallelism mechanisms. Exascale systems will be designed to achieve the best performance within both power and cost constraints. In addition, hardware breakthroughs will be needed to achieve useful exascale computing later this decade, at least within any reasonable power budget. Applications and algorithms will need to change and adapt as node architectures evolve. They will need to manage locality and perhaps resilience to achieve high performance. A key element of the strategy as we move forward is the co-design of applications, architectures and programming environments in an application-driven design process.

In this work, we have demonstrated an application-driven codesign process to navigate this complex trade-space to developing effective interconnects for exascale computing systems to meet the requirements of demanding applications to enable transformational scientific breakthroughs over the next decade. Overall results lead to a promising approach for addressing the interconnect requirements of future exascale computing systems. We have taken the first steps towards technologies that can overcome the challenges of delivering scalable interconnects for systems containing millions or even billions of endpoints within a fixed cost envelope and fixed power budget. Moreover, the application-driven design process has guided the design process to ensure that all design trade-offs favor improved effectiveness for scientific application performance. There is an unprecedented opportunity for application and algorithm developers to influence the direction of future architectures to reinvent computing for the next decade.

Bibliography

[1] F. Abel, C. Minkenberg, R. Luijten, M. Gusat, I. Iliadis, R. Hemenway, R. Grzybowskiand C. Minkenberg, and R. Luijten. Optical-packet-switched interconnect for supercomputer applications. *OSA J. Opt. Network*, 3(12):900–913, Dec 2004.

[2] Krste Asanovic, Ras Bodik, Bryan Christopher Catanzaro, Joseph James Gebis, Parry Husbands, Kurt Keutzer, David A. Patterson, William Lester Plishker, John Shalf, Samuel Webb Williams, and Katherine A. Yelick. The landscape of parallel computing research: A view from Berkeley. Technical Report UCB/EECS-2006-183 (http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.html), EECS Department, University of California, Berkeley, December 2006.

[3] James Balfour and William Dally. Design tradeoffs for tiled CMP on-chip networks. In *International Conference on Supercomputing*, 2006.

[4] Christopher Batten et al. Building manycore processor-to-DRAM networks with monolithic silicon photonics. In *Proceedings of 16th IEEE Symposium on High Performance Interconnects*, Aug 2008.

[5] K. Bernstein et al. Interconnects in the third dimension: Design challenges for 3D ICs. In *Design Automation Conference*, 2007.

[6] Shekhar Borkar. Design challenges of technology scaling. *IEEE Micro*, 19(4):23–29, 1999.

[7] Julian Borrill, Jonathan Carter, Leonid Oliker, David Skinner, and R. Biswas. Integrated performance monitoring of a cosmology application on leading hec platforms. In *Proceedings of the International Conference on Parallel Processing (ICPP), to appear*, 2005.

[8] M. Briere et al. Heterogeneous modeling of an optical network-on-chip with SystemC. In *16th IEEE International Workshop on Rapid System Prototyping*, 2005.

[9] Doug Burger, Stephen W. Keckler, Kathryn S. McKinley, Mike Dahlin, Lizy K. John, Calvin Lin, Charles R. Moore, James Burrill, Robert G. McDonald, William Yoder,

and the TRIPS Team. Scaling to the end of silicon with edge architectures. *Computer*, 37(7):44–55, 2004.

[10] Cactus Homepage. `http://www.cactuscode.org`, 2004.

[11] A. Canning, L.W. Wang, A. Williamson, and A. Zunger. Parallel empirical pseudopotential electronic structure calculations for million atom systems. *J. Comput. Phys.*, 160:29, 2000.

[12] Roger D. Chamberlain, Ch'ng Shi Baw, and Mark A. Franklin. Gemini: An optical interconnection network for parallel processing. *IEEE Trans. on Parallel and Distributed Systems*, 13, October 2002.

[13] Johnnie Chan, Aleksandr Biberman, Benjamin G. Lee, and Keren Bergman. Insertion loss analysis in a photonic interconnection network for on-chip and off-chip communications. In *IEEE Lasers and Electro-Optics Society (LEOS)*, Nov. 2008.

[14] W. J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2004.

[15] William Dally and Brian Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers, 2004.

[16] T. DeFanti, M. Brown, J. Leigh, O. Yu, E. He, J. Mambretti, D. Lillethun, and J. Weinberger. Optical switching middleware for the optiputer. *IEICE Trans. FUNDAMENTALS/COMMUN./ELECTRON./INF. & SYST*, February 2003.

[17] David Donofrio, John Shalf, Leonid Oliker, Michael F. Wehner, Chris Rowen, Jens Krueger, Shoaib Kamil, and Marghoob Mohiyuddin. Energy-efficient computing for extreme-scale science. *Computer*, 42:62–71, 2009.

[18] W. Eatherton. The push of network processing to the top of the pyramid. In *keynote address at Symposium on Architectures for Networking and Communications Systems*, Oct. 2628 2005.

[19] Hans Eberle and Nils Gura. Separated high-bandwidth and low-latency communication in the cluster interconnect clint. In *Proceedings of the IEEE Conference on Supercomputing*, 2002.

[20] Peter Kogge et al. Exascale computing study: Technology challenges in achieving exascale systems. `http://users.ece.gatech.edu/~mrichard/ExascaleComputingStudyReports/exascale_final_report_100208.pdf`, 2008.

[21] Biswajeet Guha, Bernardo B. C. Kyotoku, and Michal Lipson. Cmos-compatible athermal silicon microring resonators. *Opt. Express*, 18(4):3487–3493, 2010.

[22] Vipul Gupta and Eugen Schenfeld. Performance analysis of a synchronous, circuit-switched interconnection cached network. In *ICS '94: Proceedings of the 8th international conference on Supercomputing*, pages 246–255, New York, NY, USA, 1994. ACM Press.

[23] Thomas Hauser, Timothy I. Mattox, Raymond P. LeBeau, Henry G. Dietz, and P. George Huang. High-cost cfd on a low-cost cluster. In *Proceedings of the IEEE Conference on Supercomputing, Dallas, Texas*, November 4-10 2000.

[24] W. Haydt and J. Buck. Engineering electromagnetics seventh edition. *New York: McGraw Hill*, 2006.

[25] Gilbert Hendry, Johnnie Chan, Shoaib Kamil, Lenny Oliker, John Shalf, Luca P. Carloni, and Keren Bergman. Silicon nanophotonic network-on-chip using tdm arbitration. *High-Performance Interconnects, Symposium on*, 0:88–95, 2010.

[26] John L. Hennessy and David A. Patterson. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[27] PMEMD Homepage. `http://amber.scripps.edu/pmemd-get.html`.

[28] Mark Horowitz, Chih-Kong Ken Yang, and Stefanos Sidiropoulos. High-speed electrical signaling: Overview and limitations. *IEEE Micro*, 18:12–24, 1998.

[29] Intel Single Chip Cluster Computer Homepage. `http://techresearch.intel.com/ProjectDetails.aspx?Id=1`.

[30] The international technology roadmap for semiconductors (ITRS). http://www.itrs.net.

[31] IPM Homepage. `http://www.nersc.gov/projects/ipm`, 2005.

[32] S. Kamil, A. Pinar, D. Gunter, M. Lijewski, L. Oliker, and J. Shalf. Reconfigurable hybrid interconnection for static and dynamic scientific applications. In *ACM International Conference on Computing Frontiers*, 2007.

[33] Shoaib Kamil, Ali Pinar, Daniel Gunter, Michael Lijewski, Leonid Oliker, and John Shalf. Reconfigurable hybrid interconnection for static and dynamic applications. In *ACM International Conference on Computing Frontiers*, 2007.

[34] D. Keyes. Science case for large-scale simulation. In *DOE Office of Science Workshop*, June 2003.

[35] L.T. Kou, L.J. Stockmeyer, and C.K. Wong. Covering edges by cliques with regard to keyword conflicts and intersection graphs. *Communications of the ACM*, 21:135–138, 1978.

[36] Benjamin G. Lee et al. High-speed 2×2 switch for multi-wavelength message routing in on-chip silicon photonic networks. In *European Conference on Optical Communication (ECOC)*, Sept. 2008.

[37] W. W. Lee. Gyrokinetic particle simulation model. *J. Comp. Phys.*, 72, 1987.

[38] Xiaoye S. Li and James W. Demmel. Superlu–dist: A scalable distributed-memory sparse direct solver for unsymmetric linear systems. *ACM Trans. Mathematical Software*, 29(2):110–140, June 2003.

[39] Z. Lin, S. Ethier, T.S. Hahm, and W.M. Tang. Size scaling of turbulent transport in magnetically confined plasmas. *Phys. Rev. Lett.*, 88, 2002.

[40] D. A. B. Miller and H. M. Ozaktas. Limit to the bit-rate capacity of electrical interconnects from the aspect ratio of the system architecture. *J. Parallel Distrib. Comput.*, 41(1):42–52, 1997.

[41] David A. B. Miller. Rationale and challenges for optical interconnects to electronic chips. In *Proc. IEEE*, pages 728–749, 2000.

[42] Ian O'Connor et al. Towards reconfigurable optical networks on chip. In *Reconfigurable Communication-centric Systems-on-Chip workshop*, June 2005.

[43] L. Oliker, A. Canning, J. Carter, et al. Scientific application performance on candidate petascale platforms. In *Proc. IEEE International Parallel & Distributed Processing Symposium (IPDPS)*, Long Beach, CA, Mar 26-30, 2007.

[44] Michele Petracca, Benjamin G. Lee, Keren Bergman, and Luca Carloni. Design exploration of optical interconnection networks for chip multiprocessors. In *16th IEEE Symposium on High Performance Interconnects*, Aug 2008.

[45] V. Puente, R. Beivide, J. A. Gregorio, J. M. Prellezo, J. Duato, and C. Izu. Adaptive bubble router: a design to improve performance in torus networks. In *Proc. Of International Conf. On Parallel Processing*, pages 58–67, 1999.

[46] J. Qiang, M. Furman, and R. Ryne. A parallel particle-in-cell model for beam-beam interactions in high energy ring collide rs. *J. Comp. Phys.*, 198, 2004.

[47] Rolf Rabenseifner. Automatic profiling of MPI applications with hardware performance counters. In *Proceedings of the 6th European PVM/MPI User's Group Meeting (EuroPVM/MPI)*, pages 35–42, September 1999.

[48] D.A. Reed. Workshop on the roadmap for the revitalization of high-end computing. In *Computing Research Association*, June 2003.

[49] V. L. Rideout, F. H. Gaensslen, and A. LeBlanc. Device design considerations for ion implanted n-channel mosfets. *IBM J. Res. Dev.*, 19(1):50–59, 1975.

[50] G. Rtvri, P. Fodor, J. Tapolcai, and T. Cinkler. Multi-layer traffic engineering schemes in gmpls networks. In *Proc. International Conference on Transparent Optical Networks (ICTON)*, Barcelona, Spain, July 2005.

[51] Larry Seiler, Doug Carmean, Eric Sprangle, Tom Forsyth, Michael Abrash, Pradeep Dubey, Stephen Junkins, Adam Lake, Jeremy Sugerman, Robert Cavin, Roger Espasa, Ed Grochowski, Toni Juan, and Pat Hanrahan. Larrabee: a many-core x86 architecture for visual computing. In *SIGGRAPH '08: ACM SIGGRAPH 2008 papers*, pages 1–15, New York, NY, USA, 2008. ACM.

[52] Assaf Shacham, Keren Bergman, and Luca Carloni. On the design of a photonic network-on-chip. In *First International Symposium on Networks-on-Chip*, 2007.

[53] Assaf Shacham, Keren Bergman, and Luca P. Carloni. Photonic networks-on-chip for future generations of chip multiprocessors. *IEEE Transactions on Computers*, 57(9):1246–1260, 2008.

[54] H.D. Simon, W.T. Kramer, W. Saphir, J. Shalf, D.H. Bailey, L. Oliker, M. Banda, C. W. McCurdy, J. Hules, A. Canning, M. Day, P. Colella, D. Serafini, M.F. Wehner, and P. Nugent. Science-driven system architecture: A new process for leadership class computing. *Journal of the Earth Simulator*, 2, January 2005.

[55] Horst Simon, Richard Stevens, and Thomas Zacharia. Modeling and simulation at the exascale for energy and the environment town hall meetings. `http://www.er.doe.gov/ascr/ProgramDocuments/Docs/TownHall.pdf`, 2008.

[56] Horst Simon, Richard Stevens, and Thomas Zacharia. A platform strategy for the advanced simulation and computing program, 2008.

[57] Larry Smarr, Joe Ford, Phil Papadopoulos, Shaya Fainman, Thomas DeFanti, Maxine Brown, and Jason Leigh. The optiputer, quartzite, and starlight projects: A campus to global-scale testbed for optical technologies enabling lambdagrid computing. In *Optical Fiber Communication Conference & Exposition and the National Fiber Optic Engineers Conference (OFC/NFOEC)*, March 2005.

[58] Top 500 supercomputer sites. `http://www.top500.org`, 2005.

[59] Berkeley UPC Homepage. `http://upc.lbl.gov`, 2002.

[60] D. Vantrease et al. Corona: System implications of emerging nanophotonic technology. In *Proceedings of 35th International Symposium on Computer Architecture*, Aug 2008.

[61] Andres Varga. The omnet++ discrete event simulation system. *Proceedings of the European Simulation Multiconference (ESM'2001)*, June 2001.

[62] Jeffery S. Vetter and Frank Mueller. Communication characteristics of large-scale scientific applications for contemporary cluster architectures. In *Proceedings of the 16th International Parallel and Distributed Processing Symposium (IPDPS)*, 2002.

[63] Jeffrey S. Vetter and Andy Yoo. An empirical performance evaluation of scalable scientific applications. In *Proceedings of the IEEE Conference on Supercomputing*, 2002.

[64] Jeffry Vetter and Scott Hemmert. Iaa interconnection network workshop. `http://www.csm.ornl.gov/workshops/IAA-IC-Workshop-08/`, San Jose, California, July 21-22, 2008.

[65] Y. Vlasov, W. M. J. Green, and F. Xia. High-throughput silicon nanophotonic wavelength-insensitive switch for on-chip optical networks. *Nature Photonics*, 2:242–246, April 2008.

[66] H. Wang et al. ORION: A power-performance simulator for interconnection networks. In *35th International Symposium on Microarchitecture*, 2002.

[67] Lin Wang Wang. A survey of codes and algorithms used in nersc materials science allocations. *LBNL Technical Report LBNL-16691*, 2006.

[68] M. R. Watts. Ultralow power silicon microdisk modulators and switches. In *5th Annual Conference on Group IV Photonics*, 2008.

[69] David Wentzlaff, Patrick Griffin, Henry Hoffmann, Liewei Bao, Bruce Edwards, Carl Ramey, Matthew Mattina, Chyi-Chang Miao, John F. Brown III, and Anant Agarwal. On-chip interconnection architecture of the tile processor. *IEEE Micro*, 27:15–31, 2007.