Interconnection Networks: Architectural Challenges for Utility Computing Data Centers

Olav Lysne, Sven-Arne Reinemo, Tor Skeie, Åshild Gronstad Solheim, and Thomas Sødring
Simula Research Laboratory, University of Oslo

Lars Paul Huse and Bjørn Dag Johnsen, Sun Microsystems

Advancing research will enable an interconnection network to support the same seamless virtualization found in other parts of hardware, such as CPUs. Such a network thus poses particular challenges as well as opportunities for a utility computing data center.

Years ago, the vision of a computational grid received considerable attention. The idea that computational power should be as readily available over the Internet as electrical power is over the power grid is appealing, and it inspired researchers worldwide to launch many new research projects and programs addressing different issues. Even though academic interest in the computational grid appears to have peaked, it spawned awareness of a new operational mode for computational data centers, often called utility computing. In this mode, the system assigns compute resources to customers on demand, who request a subset of the resources in a data center for a defined period. Typically, customers pay only for the resources they require and only for the period they require them. To capitalize on this model, vendors have provided various solutions to utility computing, such as Sun’s N1, HP’s Adaptive Enterprise, and IBM’s E-Business On Demand. Recent examples of utility computing services now being offered include the Sun Grid Compute Utility and the Amazon Elastic Compute Cloud (www.amazon.com/ec2).

A utility computing data center (UCDC) dynamically creates virtual servers containing a subset of the available resources to fulfill user demands. The jobs or services running in a UCDC typically have diverse characteristics, such as different resource requirements, running times, software quality, security requirements, and importance. In addition to the classic high-performance computing applications, a job might be, for example, an ad hoc Web service set up to host a World Cup football championship for two months. While present-day solutions to utility computing mainly consist of software running atop existing architectural designs, we argue that to get the full benefit of UCDCs in the future, the underlying architecture should evolve. Recent innovations include the development of virtualization support in hardware, such as in CPUs from major vendors.

In research on interconnection networks, the common assumption has been that the system is executing only one job or class of jobs concurrently, and that the interconnection network’s task is to maximize overall performance. A UCDC’s diverse requirements therefore generate a set of problems that researchers have only marginally studied before.

INTERCONNECTION NETWORK RESEARCH
Interconnection networks stipulate particularly high demands in terms of bandwidth, delay, and delivery over short distances. Typical sample interconnection application areas include multiprocessors, computing clusters, storage servers, computer I/O devices, processor-memory interconnects, networks-on-a-chip, and...
the internal network of Internet Protocol routers. Two decades ago, interconnection networks were mainly bus-based systems. However, such architectures suffer from both serial access and arbitration overheads, and in that respect they scale poorly. To keep pace with computer evolution and the increased burden imposed on data centers, application processing, and enterprise computing, interconnection network technologies have migrated into point-to-point switch-based networks over the past 10 to 15 years. These technologies benefit from the parallelism offered by nonblocking switches capable of forwarding packets at full link speed concurrently between input and output ports if no group of packets competes for the same output port. Today, most state-of-the-art interconnection network technologies like InfiniBand (www.infinibandta.org) and HyperTransport (www.hypertransport.org) support switch-based principles.

Interconnection networks typically use techniques like virtual cut-through or wormhole switching. In both these techniques, when a packet header arrives at a switch, the switch immediately routes it onward, which reduces packet latency in large interconnection networks. The difference between virtual cut-through and wormhole switching is that the former assumes a maximum packet size and will buffer the packet completely in one switch if the requested output port is busy. The latter, on the other hand, does not assume a maximum packet size, and a switch may only have sufficient buffer space for a portion of a packet. This makes a packet stretch out in the network during congestion, causing a movement toward its destination that resembles a worm wriggling through a heap of sand—hence the term wormhole switching.

A distinguishing factor when comparing interconnection networks to traditional computer networks is that no packet loss occurs in an interconnection network. Because the system does not let a switch discard noncorrupted packets, it must exercise flow control on the links. Packets need sufficient credits in the next switch to be forwarded, which guarantees the buffer space to receive it at the other end. Such lossless networks are susceptible to deadlocks because buffer resources of one channel will have direct and indirect dependencies on buffer resources of other channels. As such, the system must choose routing strategies carefully to avoid deadlocks.

Figure 1 depicts a classic example of a small network that can deadlock. If all switches forward their packets counterclockwise, deadlock occurs when all switches simultaneously try to forward streams of packets to the switch located diagonally opposite. The packets will need to make two hops to reach their destination, but are blocked because they need access to the buffers in the next switch. However, packets being blocked for the same reason occupy those buffers, and so forth. Thus, the routing function causes a cyclic dependency between the channels’ buffer resources.

The first important breakthrough in deadlock avoidance considered only the store-and-forward paradigm. Later, researchers extended this result to wormhole switching and adaptive routing techniques. They based all these results on the notion of channel dependency graphs, which uses conventional graph theory to examine whether a routing function is deadlock-free. Along with the migration to switch-based networks, most academic research shifted focus to propose scalable regular network topologies—where meshes, hypercubes, tori, and multistage interconnects are most prominent—together with deadlock-free routing algorithms that could benefit from the regular structure of these topologies.

In the late 1990s, the high-performance computing community started building cluster computers connected by off-the-shelf switches. Such computer systems might also be structured as irregular topologies, leading
to the need for topology-agnostic routing strategies. Academia focused on developing such routing algorithms and, recently, researchers have introduced algorithms that perform almost as well as regular topology-specific algorithms, in addition to the flexibility that generic algorithms provide.

Recently, there has been a shift toward an increased focus on the network’s ability to provide sustained service if a network component fails. A plethora of work has addressed this problem area, most of which relies on a static fault model that assumes the entire network and accompanying applications remain shut down during the network’s reconfiguration. The current trend is to handle fault situations on the fly, assuming a dynamic fault model. This poses new challenges related to the deadlock problem because rerouting packets to a new fault-free path causes new channel dependencies in the resulting routing function.

Researchers have also focused on quality-of-service (QoS) in interconnection networks. Along with the introduction of new application areas related to multimedia and other time-sensitive applications, it became evident that interconnection networks must offer traffic differentiation, as increases in bandwidth do not perpetuate, and thus developers cannot solely rely on overprovisioning in the future. The research community has taken note of this, and today most interconnection network standards offer QoS mechanisms, while academia has been researching how to best utilize these mechanisms.

Recently, both standardization bodies and academia have studied congestion management. Since interconnection networks use flow control, congestion affects overall performance, severely limiting the network’s ability to meet its QoS guarantees. When congestion occurs, a depletion of credits begins, with the result that buffers start filling up. The situation spreads upstream through the network, building up congestion trees. To remedy this problem, the interconnection networking community has proposed various congestion-management techniques that span simple forward/backward explicit notifications to more advanced solutions that perform congestion-tree housekeeping.

SYSTEM MODEL

We view a UCDC as a large collection of resources in which each resource falls into one of the following categories:

- compute nodes (CPUs with memory),
- storage nodes (disks or tapes), and
- access nodes (gateways to external networks).

An interconnection network links these resources. In principle, the interconnection technology used is not fixed, but we assume it is based on a point-to-point link technology like InfiniBand. At least one link connects each resource to the interconnection network. Both capacity and resilience can prompt the need for connecting a resource to the interconnection network via more than one link.

Figure 2 shows our UCDC model. End users for such a system simply lease resources as required, which gives them the flexibility to offload management of the physical resources to the UCDC provider. The UCDC provider benefits by sharing a single set of physical resources among several different users.

The UCDC system can group a network’s resources into logical subnetworks to support virtualization. In this case, each logical subnetwork, which has its own virtual interface, is referred to as a virtual server. The benefits of virtualization include increased flexibility, availability, and scalability. The flexibility arises from the ability to change the allocated resources’ setup during runtime. The ability to move the virtual server from one set of physical resources to another, if required, provides availability. Increased capability, such as an increase in available compute power, yields scalability by allowing an allocation of additional resources to the virtual server.

However, virtualization does introduce concerns relating to security, anonymity, and QoS. An allocated partition within the network should not be allowed to interfere with another partition. Interference can take the form of unauthorized access to resources, applica-
tions, and data—including interference on the interconnect in the form of malicious or accidental cross-traffic generation. Anonymity can be an important aspect within such a domain as an end user might want to prevent a third party from obtaining knowledge about the traffic traversing both shared and private links. This kind of anonymity also extends to whether the partition is idle or not. Allocated partitions should ensure an implementation of privacy and security where the typical sandbox approach to security might apply. QoS exists on many levels, such as between applications in one operating system instance, between several operating system instances on one CPU, between several CPUs in one node, between nodes in the same partition, and between different partitions.

Traditionally, processor architectures include two modes of operation: user (or unprivileged) and supervisor (or privileged). The operating system uses the supervisor mode, in which the executing code has access to all instructions and control registers. Since the operating system, including device drivers, controls the partition’s operation, the UCDC provider must either trust the operating system on the compute nodes to behave correctly, or the network implementation (including the network interface cards) must be able to enforce critical policies for access control and traffic separation. Hence, in UCDC environments where a job specification may include operating systems and drivers that the UCDC owner does not control, the compute nodes become inherently untrusted, and the requirements on the network implementation become stricter.

Recent developments in CPU technology include multicore CPUs. The use of multicore CPUs will profoundly affect the programming paradigm, which will also be reflected in future UCDCs as application software will be written and rewritten to take advantage of this parallelism. Hypervisors must take into account both multicore CPUs and how virtualization should extend from a single-core to as few jobs as possible, and the set of jobs the fault terminates should, to the greatest extent possible, be controlled from a job-importance perspective. A fault-tolerance solution should allow unaffected jobs to run without interruption.

This list’s properties are not new. All are easily recognizable from the requirements that first faced timesharing mainframes decades ago and that led to solutions such as CPU scheduling, virtual memory, and disk scheduling. For the system interconnect, however, the preceding list presents the following yet-to-be-addressed challenges.

Flexible partitioning. Because several jobs will run concurrently and the software’s quality in these jobs will vary, a misbehaving job must not consume bandwidth in the interconnect to the extent that other jobs suffer. Moreover, a typical job will come with a set of requirements relating to the number of compute nodes, storage nodes, and access nodes it demands. This requires partitioning the network as a set of, possibly noncontiguous, regions to maximize resource utilization.

Fault tolerance. The effect of a faulty component in the interconnection network should be constrained to as few jobs as possible, and the set of jobs the fault terminates should, to the greatest extent possible, be controlled from a job-importance perspective. A fault-tolerance solution should allow unaffected jobs to run without interruption.

Predictable service. With regard to partitioning and multiple jobs, it should be possible to guarantee a specific portion of the interconnect network capacity to each partition or job. Further, it should be possible to differentiate between jobs based on importance.

 RESOURCE PARTITIONING

Partitioning describes the mapping of a request for compute resources to a subset of the physical architecture. Taking a snapshot of an operational UCDC,
we expect to see a set of jobs running, each utilizing a subset of the assigned resources, and each possibly belonging to different owners. The set of resources assigned to a job forms a partition (or virtual server). In the interconnection network, we only observe communication between two resources if they are part of the same partition.

Figure 3 illustrates three different approaches to resource partitioning. In the case where all partitions are disjoint and can be assumed to utilize disjoint parts of the interconnection network, the problem is relatively easy. It is equivalent to the traditional problem of contiguously allocating processors and has been studied extensively, particularly for meshes, but also for tori and hypercube topologies.

In the two-dimensional case, these topologies are rectangular, and resources (CPUs) have typically been allocated in nonoverlapping rectangular partitions of suitable size. The preferred routing methods for these topologies automatically let the resources in each subrectangle communicate internally without sending packets outside its own partition. Thus, there is no link contention between the separate partitions. This can be advantageous for communication-intensive applications.

Fragmentation limits a system’s resource utilization and is a well-known problem for traditional contiguous processor-allocation schemes. External fragmentation means that there are enough resources available to serve an incoming job request, but the shape of the available resources is not as required. Internal fragmentation occurs when some restriction causes the allocation of more resources than requested by a job. One possible restriction could be that the allocated resources must constitute a square in which the length of each side is a power of 2. Noncontiguous allocation methods solve the problem of fragmentation at the expense of increased link contention between the different partitions. Researchers have studied and assessed several such allocation algorithms for various communication patterns. Hybrid schemes have also been proposed.

For several reasons, however, previous studies of processor allocation did not solve the problems facing a UCDC provider. First, these studies considered only allocation of compute nodes. In addition to these CPUs, a job running on a UCDC typically also requires access to other resource types such as storage and access nodes. Thus, our system model for a UCDC poses new challenges to the resource partitioning problem.

Second, traditional approaches to contiguous resource allocation assumed that the resources needed for each partition can be made disjoint. This assumption might not be sustainable for a provider of virtual servers due to resource utilization requirements. Moreover, utilization of storage and access nodes could imply overlapping partitions and shared links.

Third, the predominant trend in network-based computing uses multistage topologies like Clos networks and fat trees. These topologies have characteristics that differ from meshes, tori, and hypercubes, making the existing studies inapplicable.

Meeting future challenges requires new studies on resource partitioning, which must take the following into account:

- In addition to the allocation of compute nodes, other resources such as storage nodes and access nodes must be considered.
- Partitions should preferably be as disjoint as possible.
- Communication interference between partitions should be limited as far as possible.
- Multicore CPUs are now common and should be a design consideration.
- Sharing physical resources between partitions might raise security issues that should be carefully considered.

Figure 3. Three different approaches to resource partitioning. A new request for compute resources arrives when three jobs are running in the sample UCDC: The first job runs on entities 0, 1, 4, and 5; the second runs on entities 7 and 11; and the third runs on entities 12 through 15. The arrow indicates the number of resource entities requested, and the broken line frames the resulting allocation. (a) Allocations must be rectangular. (b) Allocations may be irregular. (c) Allocations may be irregular and overlapping.
• Currently, the most relevant network topologies are multistage networks, therefore these topologies should be in focus.

Generic routing is crucial for achieving flexible partitioning and high utilization of UCDCs.

**GENERIC ROUTING FOR FLEXIBLE PARTITIONING**

In previous work related to processor allocation in network-based computing, researchers generally assumed that the underlying routing function is fixed. For example, most mesh-based architectures rely on dimension-order routing (DOR) because the deadlock problem inherent in most interconnection networks has deprived these networks of the routing flexibility available in other network types, such as the Internet. For contiguous allocation, this has influenced the research community to partition in submeshes, where DOR is applicable. Also, for noncontiguous partitioning of mesh-based servers, DOR provides the underlying routing. This might cause the internal communication within a partition to pass through several other partitions and interfere with their behavior.

Two recent developments have opened up new possibilities. First, researchers have offered methods for dynamically reconfiguring interconnection networks. These methods let an interconnection network change from an old routing function to a new one while the network is up and running—without creating deadlocks during the transition phase. Second, researchers have significantly improved topology-agnostic routing algorithms, and their performance can now compete with special-purpose routing algorithms.

These developments allow for the dynamic creation of irregular subnetworks that serve a particular partition. Systems can use efficient topology-agnostic routing algorithms to keep traffic confined to a subnetwork, so that partitions remain separate from each other. Dynamic reconfiguration methods can change the routing function while the network is up and running to allow implementation of the routing function that a subnetwork needs for a new partition.

This poses the challenge of developing efficient partitioning methods that map jobs to computing resources by exploiting the flexibility that generic routing and dynamic reconfiguration provide. The new partitioning algorithms must ensure sufficient performance within each partition. In the model we have described, the scheduling problem is still prominent—the scheduler’s task is to select, from the set of available jobs, the next job to be served, taking into account the partitioning strategy. Job throughput is thus an important data center metric.

**FAULT TOLERANCE**

Fault tolerance is important to both the user and owner of a UCDC, but for different reasons. The user often embeds fault tolerance in the application, which lets it gracefully handle the occurrence of a fault in one or more resources. In this case, the most important feature required from the UCDC is redundancy, specifically the replacement of a faulty component such as a broken compute node to retain the amount of resources dedicated to the user. Thus, the user handles faults, while the owner makes sure that faulty components are replaced.

From the UCDC owner’s perspective, making sophisticated fault tolerance part of the UCDC architecture helps simplify system administration and improve utilization in the case of faults. This makes it possible to, for example, have a scheme that seamlessly preempts resources from a lower-priority partition whenever a resource in a higher-priority partition fails.

Over the past few decades, researchers have extensively studied the ability to maintain continued operation in the presence of faults in interconnection networks. This has resulted in a plethora of methods for addressing different fault models (link/node faults, static/dynamic/transient faults), different topologies (meshes/tori/hypercubes/multistage/irregular), and places of application (local/end-to-end rerouting).

Most methods developed for interconnection networks in general can also be applied in the system model we have described, but in our scenario new aspects that have not been considered previously become important. These are connected to the observation that failure of a node or link will most likely affect only a few—perhaps only one—of the current jobs or partitions. This means that most jobs could and should be allowed to continue as if nothing had happened.

Methods for fault tolerance that are limited to the resources set aside for one partition therefore become important. This means that methods for fault tolerance restricted to only a subpart of the network should be studied. Further, methods for rearranging resources between partitions of different priorities after a fault has occurred must be studied.

**PREDICTABLE SERVICE**

In many cases, guaranteeing completely disjoint physical resources to different partitions will be impractical. This means that sharing switches and links between virtual servers will be commonplace. Guaranteeing network service to a virtual server that shares resources with other virtual servers, even if the software running on a different virtual server misbehaves, requires QoS guarantees in the interconnection network.

Although researchers have studied QoS intensively
for mainstream packet-based networks over the years, beneficial results for interconnection networks have been scarce. The most important developments are the multimedia router,\textsuperscript{12} some work on reducing the resources needed for service differentiation,\textsuperscript{13} and some work specific to InfiniBand.\textsuperscript{14}

These works’ focus, however, is somewhat different from the UCDC’s needs. In particular, when studying QoS guarantees in the interconnection network, we must assume that applications might misbehave and attempt to use far more of the network resources than their allotted share. Further, the class-based differentiation that most methods suggest is insufficient for the level of differentiation required in the UCDC. The virtual view of the available resources requires a more fine-grained QoS scheme that reflects fault tolerance and partitioning in the compute and storage nodes and in the network.

The Internet community has conducted much research on algorithms that support the type of guarantees we seek, such as high granularity differentiation similar to IntServ (www.rfc-editor.org/rfc/rfc1633, txt). These results and methods could be ported to interconnection networks—a task more challenging than it might first appear. The naive approach would be to apply the mechanisms proposed for use for example in the Internet or asynchronous transfer mode to interconnection networks. Yet this approach must overcome the problem that researchers based these methods on the assumption that overutilization of one link will have no effect on the performance of any other.

This is not true for interconnection networks because the use of link-level flow control introduces congestion trees, which reduce the network’s overall performance and might even include flows not destined for the overutilized link. Therefore, we must study existing results from the Internet community but within the context of interconnection networks, as well as develop new solutions.

Three major topics that researchers must address to successfully implement the UCDC are high-granularity service differentiation, congestion control, and admission control. Recent work in the context of interconnection networks includes high granularity\textsuperscript{12} and class-based service differentiation schemes.\textsuperscript{14} Some researchers recently proposed congestion-control mechanisms,\textsuperscript{15,16} while other work supports a series of admission-control mechanisms.\textsuperscript{17} The question of how the suggested mechanisms impact partitioning and fault-tolerance schemes must be answered, and the Grand Challenge of combining all these features in the UCDC remains to be addressed.

\textbf{We believe that more research will enable an interconnection network to support the same seamless virtualization found in other parts of hardware, such as CPUs. The interconnection network thus poses particular challenges as well as opportunities for a UCDC.}$

\begin{thebibliography}{10}


\end{thebibliography}

**Olav Lysne** is a research director at Simula Research Laboratory and a professor of computer science at the University of Oslo. His research focuses on interconnects, specifically effective routing, fault tolerance, and quality of service. Lysne received an MS and a Dr.Scient, both from the University of Oslo. Contact him at olavly@simula.no.

**Svein-Arne Reinemo** is a postdoctoral researcher at Simula Research Laboratory. His research focuses on routing and quality of service in interconnection networks. Reinemo received a Cand.Scient. and a Dr.Scient. in computer science from the University of Oslo. Contact him at svenar@simula.no.

**Tor Skeie** is an associate professor at Simula Research Laboratory and the University of Oslo. His research focuses on scalability, effective routing, fault tolerance, and quality of service in switched network topologies. Skeie received an MS and a PhD in computer science from the University of Oslo. Contact him at tskeie@simula.no.

**Åshild Grønstad Solheim** is a PhD student at Simula Research Laboratory and the University of Oslo. Her research interests include resource partitioning, deadlock-free routing, and fault tolerance in interconnection networks. Solheim received an MS in computer science from the University of Oslo. Contact her at aashig@simula.no.

**Thomas Sødring** is a postdoctoral researcher at Simula Research Laboratory working on the ICON project. His research focuses on interconnects and the development of generic solutions for routing, fault tolerance, and quality of service. Sødring received a PhD in computer applications from Dublin City University. He is a member of the ACM. Contact him at tsodring@simula.no.

**Lars Paul Huse** is a senior staff engineer at Sun Microsystems, Oslo. His research focuses on crafting software to effectively utilize fast interconnects from hardware-close programming up to the user application interfacing. Huse received a PhD in computer science from the University of Oslo. Contact him at lars.paul.huse@sun.com.

**Bjørn Dag Johnsen** is a principal engineer at Sun Microsystems, Oslo. His research focuses on the implementation, configuration, and management of large-scale fabric and cluster systems. Johnsen received an MS in computer science from the University of Bergen, Institute of Informatics, Norway. Contact him at bjorn-dag.johnsen@sun.com.