

High Performance and Scalable Communication and I/O Subsystems for Next-Generation Clusters with Manycore Architectures

Dhabaleswar K. (DK) Panda
Department of Computer Science and Engineering
The Ohio State University, 2015 Neil Avenue, Columbus, OH 43210
panda@cse.ohio-state.edu

1 Motivation

Modern clusters are being used in different configurations (high performance computing clusters, parallel file systems and multi-tier data-centers). These systems are allowing scientists and engineers to solve grand challenge problems in their respective domains and make significant contributions to their fields. Recent progress in processing, memory, I/O and networking technologies has enabled scientists in various domains (including weather prediction, nanoscience modeling, multiscale and multiphysics modeling, biological computations, scientific, engineering, bio-medical and financial) to aim for problems that had never been considered before. Many of these applications operate on multi-terabytes of data and also generate multi-terabytes of data as output. The execution modes for many of these applications are also moving away from batch-modes to interactive modes (large-scale data-centers) where users not only want to operate on large-scale data, but also want to carry out interactive processing, visualization and data-mining on their original data and the results. Thus, clustered systems which provide dedicated nodes to handle file I/O and data analysis together with the traditional compute nodes are becoming a common scenario.

With such advances in clusters, and the upcoming even more powerful clusters (e.g., IBM Roadrunner supercomputer [7]), several interesting challenges become evident with respect to the capabilities of various systems software, including message passing libraries, file systems and data-center system software, to match the capabilities of such systems. Manycore/Multicore architectures, which have been recently gaining popularity due to their low-cost per processing element, are of particular importance in the growing scales and capabilities of these clustered systems.

Currently dual-core architectures (two cores per die) are widely available from various industry leaders including Intel, AMD, Sun and IBM. The negligible cost associated with placing an extra processing core on the same die has allowed these architectures to increase the capabilities of applications significantly. Quad-core (four cores per die) from Intel is already available. Such quad-core processors from AMD are also expected to be introduced late this year. Recently, Intel has announced that it will be introducing an 80-core die [6] within the next five years. Other industries are expected to follow this trend. Such multicore/manycore processors lead to significant challenges in designing the communication and I/O subsystems for next generation clusters.

2 Emerging Multicore/Manycore Architectures

While the main idea of all these emerging architectures is to provide multiple processing elements to function in parallel, each of these architectures differs in several key characteristics (e.g., whether they have a second-level cache, whether they share the second-level cache, the connecting topology between the cores, homogeneity of the processing power of cores, etc.). Therefore, it is important that the systems software understand these architectural differences and provide appropriate approaches to make most advantage of the benefits of each architecture while being least affected by the limitations.

Broadly, these variations in the architectures can be broken down into different classes and their impact can be studied along three dimensions: (i) on communication within the chip-level multiprocessor (intra-CMP), (ii) on communication between chip-level multiprocessors on a single system (inter-CMP) and (iii) on communication across different physical systems (inter-node). Figure 1 illustrates the three different dimensions in which the architecture of the processing cores can impact.

Intra-CMP: As shown in Figure 1, multiple different intra-CMP architectures exist with different key characteristics. For example, the Intel multicore architecture [2] provides a shared second-level cache between the cores. Further, the different cores on the chip might be directly connected with a single hop or through multiple hops. The AMD multicore architecture [3] differs from the Intel architecture in that each core has its own local cache (instead of a shared cache). Again, the connection between the cores could be single hop or multiple hops. The third significantly different architecture is that of the IBM Cell processor [8]. The key difference of this architecture is its usage of heterogeneous processing elements, where one processing

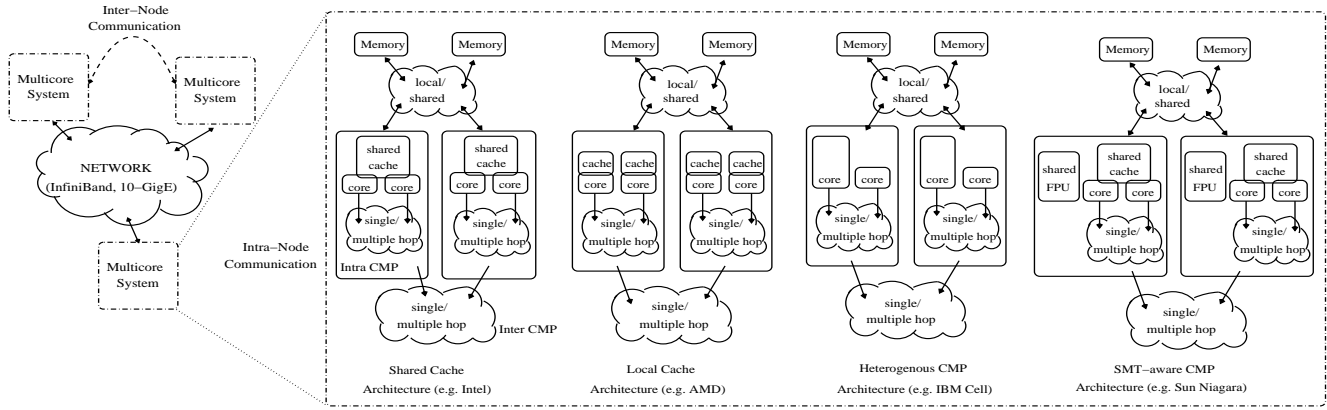


Figure 1. Emerging Multicore/Manycore Architectures with Multi-level Communication Hierarchy

element (referred to as the power processing element) is more powerful than the others (synergistic processing elements). Finally, the fourth architecture we consider is the Sun Niagara architecture [9]. This architecture is specifically tuned to efficiently support SMT (up to 4 threads on current cores). The disadvantage, however, is that all cores share an FPU unit.

Inter-CMP: With respect to the inter-CMP dimension, again multiple different architectural characteristics exist. These differences in the architectural characteristics are primarily based on whether there exists a third-level cache or not, the connection topology between different chips (e.g., single hop or multiple hop, dedicated or shared buses, etc.) and the memory architecture (e.g., if the memory is local for each chip or shared between chips with UMA or NUMA access modes).

Inter-Node: Within the inter-node dimension, though the actual communication relies on the characteristics of the high-speed network used (e.g., InfiniBand, 10 Gigabit Ethernet/iWARP), the protocol processing for inter-node communication heavily relies on the specific intra-CMP and inter-CMP architecture (e.g., closeness of the processing core memory to the network) and how the protocol processing is done (i.e., using the same core as the computation or using a dedicated set of cores).

As mentioned above, different multicore/manycore architectures differ in their characteristics significantly. Thus, this leads to a fundamental challenge of how the next generation communication and I/O subsystems can be designed by taking advantage of these emerging multicore/manycore architectural features with their three-levels of communication hierarchy.

3 Our Vision and Position Statement

As described in Section 2, several different multicore architectures have been introduced recently and more are expected to be available soon. With computer systems increasingly relying on multicore/manycore architectures to expand in capabilities, it is important that the upper-level system software be able to match and exploit these architectural variances. Thus, our objective is to investigate approaches in which the upper-level systems software can be designed to be aware of the multicore/manycore architecture and tune itself to take most benefits of the architecture while being minimally affected by its limitations. We focus on three major components, namely (i) Multicore-aware MPI, (ii) Multicore-aware I/O and file systems and (iii) Multicore-aware multi-tier data centers.

3.1 Message Passing Interface (MPI)

In this section we point out the directions how MPI libraries can be optimized on multicore/manycore clusters.

Intra-CMP and Inter-CMP Communication: With the advent of multicore/manycore processor technology, the volume of MPI communication within one node is significantly improved. Therefore, designing an MPI library with optimized intra-node communication support is critical. High performance MPI intra-node communication requires efficient buffer management, intelligent process-to-processor scheduling, and reduced data copies. Our group has conducted preliminary evaluation and analysis on multicore clusters [4] and proposed optimization schemes for MPI intra-node communication [5].

Inter-node Communication: MPI libraries over High-speed networks, such as InfiniBand [1] utilize internal buffers for performing efficient inter-node communication. The major challenge to improve inter-node communication performance in a multicore/manycore cluster would be to design *Optimized Buffer Management* mechanisms which provide very good performance in a scalable manner.

Scalable Collective Communication: Collective communication is an important component of MPI which stress the memory-cache subsystem. Designing scalable collectives on multicore/manycore systems require investigation of multi-level-aware and NUMA-aware collective algorithms.

Enhanced MPI with Dedicated Communication Threads: Multicore/manycore architectures provide an excellent opportunity to utilize the additional processing capability on each node to allow for dedicating critical tasks in communication protocols to specific cores. Tasks that might be beneficial to be processed by dedicated cores include multi-threaded progress engine with advanced datatype processing, helper thread support in one sided operations, collective functions onloading, etc. **Reliability and Fault Tolerance:** Many functions to maintain reliability and fault tolerance in MPI can also be dedicated to specific cores, such as CRC calculation, check-pointing, etc.

3.2 I/O and File Systems

In this section we discuss the challenges to design I/O and file systems on manycore/multicore clusters.

High Performance MPI-IO Subsystem: Multicore/manycore architectures are bringing new challenges in designing a high performance I/O subsystem. Collective MPI-IO calls might be achieved in a multicore-aware manner with multi-level coordination. More complicated but accurate algorithms can also be utilized to have better collective MPI-IO performance.

Remote Memory and Caching Strategies: With many cores on one node, a lot of applications may run concurrently and might need to cache a large amount of data. Using remote memory as the file system cache is one interesting direction to explore, given the high speed of modern networks such as InfiniBand. It is also critical to explore new caching strategies while utilizing remote memory cache.

Asynchronous Read and Write: On a manycore system, one or more cores can be dedicated as I/O cores while others are used as compute cores. The compute cores can handover the I/O requests to the I/O cores and resume computation, which will lead to better computation and I/O overlap. Such asynchronous schemes require investigation.

Read-ahead Strategies: Read-ahead is a techniques to hide the latency of the storage. With more compute power on manycore systems, it is possible to detect complicated file access patterns, and optimize file read-ahead strategies.

3.3 Multi-Tier Data-Centers

In this section, we discuss how next generation data-centers can exploit the emerging manycore/multicore architectures.

CMP-aware Data Sharing Techniques: To increase the performance and scalability, current data-centers rely on advanced caching techniques for efficiently sharing the data across different tiers. With emerging manycore/multicore technology, data can be efficiently shared across the application thread and the caching thread using efficient buffer management techniques, cache-aware protocols and reduced data copies.

Asynchronous Memory Operations: With the emergence of several cores in a single system, one or more cores can be dedicated for performing memory operations such as copy, initialize and also complex operations such as encryption, decryption, etc. This can lead to significant overlap of computation with memory operations.

Enhanced Data-Center Primitives with Dedicated Communication Threads: Dedicating data-center primitives such as shared state, distributed lock manager, etc., to one or more cores can avoid the interference of the operating system for communicating between the application and the data-center primitives. Further, such cores can also handle advanced operations such as intra- and inter-node communication progress, monitoring the application's data accesses and applying appropriate pre-fetching techniques.

Fine-grained Data-Center Services: Manycore systems can enable current data-centers to adopt fine-grained resource management techniques such as frequent monitoring of resource usage of back-end nodes, intercepting incoming packets for fine-grained admission control, quality of service and prioritization, etc.

References

- [1] InfiniBand Trade Association. <http://www.infinibandta.com>.
- [2] Intel Multicore Architecture. <http://www.intel.com/technology/architecture/coremicro/>.
- [3] AMD. AMD Multicore Architecture. http://multicore.amd.com/GLOBAL/WhitePapers/Multi-Core_Processors_WhitePaper.pdf.
- [4] L. Chai, Q. Gao, and D. K. Panda. Understanding the Impact of Multi-Core Architecture in Cluster Computing: A Case Study with Intel Dual-Core System. In *International Symposium on Cluster Computing and the Grid*, 2007.
- [5] L. Chai, A. Hartono, and D. K. Panda. Designing High Performance and Scalable MPI Intra-node Communication Support for Clusters. In *The IEEE International Conference on Cluster Computing*, 2006.
- [6] Intel Corporation. <http://www.vnunet.com/vnunet/news/2165072/intel-unveils-tera-scale>, Sep 2006.
- [7] IBM. http://news.com.com/ibm+to+build+opteron-cell+hybrid+supercomputer/2100-1010_3-6112439.html, Sep 2006.
- [8] James A. Kahle. The cell processor architecture. In *MICRO*, page 3, 2005.
- [9] Poonacha Kongetira, Kathirgamar Aingaran, and Kunle Olukotun. Niagara: A 32-way multithreaded sparc processor. *IEEE Micro*, 25(2):21–29, 2005.