appears in: Proceedings of the ACM conference on High Performance Computer Architecture (HPCA3), February 1–5, 1997, San Antonio, TX.

Global Address Space, Non-Uniform Bandwidth: A Memory System Performance Characterization of Parallel Systems

T. Stricker[1] and T. Gross[1,2]

[1]School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

[2]Institut für Computer Systeme
ETH Zürich
CH-8092 Zürich

Abstract

Many parallel systems offer a simple view of memory: all storage cells are addressed uniformly. Despite a uniform view of the memory, the machines differ significantly in their memory system performance (and may offer slightly different consistency models). *Cached* and *local* memory accesses are much faster than *remote read* accesses to data generated by another processor or *remote write* to data intentionally pushed to memories close to another processor. The bandwidth from/to cache and local memory can be an order of magnitude (or more) higher than the bandwidth to/from remote memory. The situation is further complicated by the heavy influence of the *access pattern* (i.e. the spatial locality of reference) on both the local and the remote memory system bandwidth. In these modern machines, a compiler for a parallel system is faced with a number of options to accomplish a data transfer most efficiently. The decision for the best option requires a cost benefit model, obtained in an *empirically* evaluation of the memory system performance. We evaluate three DEC Alpha based parallel systems, to demonstrate the practicality of this approach. The common DEC-Alpha processor architecture facilitates a direct comparison of memory system performance. These systems are the DEC 8400, the Cray T3D, and the Cray T3E. The three systems differ in their clock speed, their scalability and in the amount of coherency they provide. The DEC 8400 is a shared memory, symmetric multiprocessor based on a high speed bus offering sequential consistency; the Cray T3D and T3E are scalable multicomputers based on a scalable 3D torus interconnect and either do not cache remote accesses at all (T3E) or provide only partial memory consistency within a node (T3D) and therefore typically leave consistency to the application or compiler.

Our performance characterization shows that although the clock rate of the DEC 8400 doubled compared to the Cray T3D, the DEC 8400 offers only modest improvements in the performance of remote memory operations over the Cray T3D. The local and remote memory system performance of the Cray T3E matches the doubled clock speed of the processor.

This research was sponsored in part by the Advanced Research Projects Agency (ITO) monitored by SPAWAR under contract N00039-93-C-0152. T. Stricker's current address: Institut für Computer Systeme, ETH Zürich, Switzerland.

1 Introduction

For many applications, it is the memory system of a computer that determines the performance. The MFlop/s (million of floating point operations per second) rating of a system may be impressive, but if the memory system cannot supply operands at the right speed, application performance may be quite disappointing. This topic is especially important for high-end, parallel systems, which contain multiple processors to boost peak computation performance.

Many parallel systems provide the model of uniform memory access (all memory cells can be addressed by any processor); however the term "uniformity" applies only to the access model, not to performance. The actual performance of the memory system depends on parameters such as the working set size, the stride, and whether the processor attempts to read/write local or remote data. To obtain the best performance for applications, the compiler writer or application developer must understand the performance implications of different types of memory accesses. Whereas private-memory machines (e.g. a network of workstations) force a programmer or compiler to optimize the program for locality and efficient memory access just to obtain correctness, shared-memory systems (providing a uniform view of memory) do not require such an optimization. But to obtain respectable performance, it is advisable on shared-memory machines to move data close to the processor that operates on them and to adjust the granularity of access so that false sharing is eliminated. Even after all possible optimizations are done, communication (data transfers between different parts of the machine) remains an important part of the execution time in many parallel programs.

The designers of memory systems have a number of options to bridge the performance differences between local and remote memory accesses. The design of a memory hierarchy is a well-known strategy, and recent processors include on-chip caches to bring the lower levels of the memory hierarchy closer to the processor. However, even in the presence of on-chip caches, there are many design options left. To get a better understanding of how such memory systems behave in practice, we investigate the memory system performance of three parallel systems.

The Cray T3D, the Cray T3E, and the DEC 8400 are three recent systems that have some features in common yet include completely different memory systems. The DEC 8400 is a shared memory, symmetric multiprocessor based on a high speed bus. The Cray T3D and T3E are scalable multiprocessors based on a 3D-torus interconnect. All systems have in common that (i) they are based on the same processor architecture, (ii) each processor can access the complete memory system, and (iii) each processor contains at least a small on-chip cache. However, there exists a big difference in the access mechanisms: in case of the Cray T3D and T3E, the processor distinguishes between local and remote memory, and the L1/L2 caches of different processing elements do not cache all global memory or are non-coherent among each other. On the T3D the caches may or may not be coherent with local memory or the communication interface handling remote accesses (selection of the policy at load time). Consequently, either a sophisticated compiler or a highly skilled programmer is required to map an application onto this system. In the case of the DEC 8400, all memory locations are accessed using the same instructions, and the bus based hardware maintains coherence of a global multi-level memory hierarchy with memory banks and caches. This organization puts the burden to maintain coherency on the hardware and simplifies the programmer's or compiler's task. Furthermore, the DEC 8400 employs an off-chip L3 cache, whereas the Cray systems instead employ a streaming unit to improve memory bandwidth for strided accesses.

Given the complexity of the memory system design of modern high-performance systems, and the absence of a simple analytical model, compiler and application developers need guidance on how to implement the basic data transfers. In the experimental study reported here, we present measurements of key performance parameters, which can then be combined to obtain a realistic model of memory system performance. The experimental framework (and the environments that may apply such performance data) are sketched in Section 2. The key differences in the memory designs of these systems are presented in Section 3. Then we discuss performance for local and remote accesses in Sections 4 and 5. Section 7 extends the discussion from micro-benchmarks to a complete application kernel.

2 Experimental setup

We empirically evaluate the local and remote memory system performance of three parallel systems. To reduce architectural difference to a minimum and allow for a maximum of comparisons, all systems are based on implementations of the DEC Alpha processor architecture. These systems and their processors are the DEC 8400, 300MHz 21164 [EV-5], the Cray T3D, 150MHz 21064 [EV-4] and the Cray T3E, 300MHz 21164 [EV-5].

The DEC 8400 is a cache coherent, shared memory, symmetric multiprocessor (SMP) based on a high speed bus offering a cache coherency model close to sequential consistency. We used a four processor system and also repeated some measurements on an eight processor system. The Cray T3D and T3E are scalable multicomputers based on a 3D torus high speed interconnect. For the Cray T3D measurements we used a four processor partition out of a system with 512 processors. For the Cray T3E data we used four nodes out of the first 256 node production system shortly after its installation at the PSC. Some minor improvements of the measured data can be expected the communication software matures.

For readers not familiar with the three architectures we discuss the important parameters relevant to the memory and communication system interface in Section 3. In addition to the technical reference material of the vendors (DEC 8400 [8, 6, 9, 7], Cray T3D [8, 1, 3], Cray T3E [12, 4]), other research groups evaluated some aspects of these machines [2, 10, 15]. An empirical study comparing the two Alpha processors based on standard benchmarks provides useful insights using performance metrics not related to the memory system [5].

Our goal is to measure and compare the memory systems of these modern parallel systems. Since it is nowadays usual to use a compiler to map applications onto a parallel system, we are particularity interested in an investigation that pays attention to the access patterns encountered in compiled code. Typical workloads for machines of this class at the Los Alamos and Livermore National Laboratories point to the practical importance of vectorizable memory-intensive workloads, and when adapting such workloads to the cache oriented memory systems of microprocessors, difficulties may arise [18].

2.1 Parallel machines as compiler target

This investigation is part of the Fx Fortran compiler project; Fx [17] is a dialect of High Performance Fortran (HPF) [11], and the Fx compiler has been re-targeted to a number of parallel systems. For many applications, the key to getting good performance is to generate efficient communication code; communication code is any code that moves data from one memory zone to another. For a private-memory system (like the Paragon, iWarp or network of workstations), communication code may involve explicit "send/receive" operations and is required for correctness; for a shared-memory system like the DEC 8400, such code may involve extra memory copy operations that may be advantageous to improve performance (but are no longer required for correctness). On SMPs like DEC 8400 we call a transfer *remote*, if one processor writes data to memory and another processor reads it, and *local*, if the same processor reads and writes the data.

To tune the performance of the Fx compiler, we first measure the basic performance for key operations of the "copy-transfer-model" [15] to obtain performance figures for local and remote transfers. These micro-benchmarks allow the compiler writer, the compiler or the runtime-system to pick the least expensive way to move data in the system.

To validate the observations based on micro-benchmarks, we discuss in this paper the performance for an application kernel, a 2D-FFT. On the Cray T3D and T3E, the compiler is responsible to generate explicit communication, so the 2D-FFT is done as a sequence of four steps: 1D-FFT, transpose, 1D-FFT, transpose. Transfers are realized with a customized primitive similar to shmem_put on the T3D and with shmem_iput on the T3E[3]. On the DEC 8400, we could chose a different sequence without explicit transpose, but without establishing locality of reference, performance is dismal. So to obtain performance on the DEC 8400, we must find a similar structure for the application, and we re-targeted the Fx compiler to generate such code.

The Fx compilers for the DEC 8400 and the Cray T3D are based on the Catacomb compiler back-end [13], which generates highly optimized code for explicit data transfers. This compiler does not rely on any hardware-support for coherent shared memory and carefully separates synchronization operations from data transfers, according to the direct deposit model (see next section). For most communication steps, the Catacomb back-end produces low-level C code, which is compiled separately and linked to the Fortran application code. Catacomb provides a general way of generating communication code for all array assignment statements and array distributions, not just for transposes of two dimensional, block distributed data.

2.2 Direct deposit/fetch models

The Fx parallelizing compiler uses the *direct deposit* model [14, 16, 17], which captures the style of communication relying on remote load/stores as primitives to transfer data. Coherency is only established at explicit synchronization points. These operations form the basis for the implementation of array assignment statements with distributed arrays (as defined by HPF).

In the deposit model—or its dual counterpart, the fetch model—only one of the two node processors (sender, receiver) actively participates in a data transfer. For deposits, the sender "drops" the data into the address space of the receiver, without participation of the receiver process. Both models allow a clean separation of control and data transfer. In the deposit model, control messages, hardware barriers, or system semaphores are used to deal with explicit synchronization, and data messages are sent only when the receiver has signaled its willingness to accept them. Similarly, in the fetch model, control messages establish when data is ready to be fetched, and then the data transfer can take place. The DEC 8400 shared memory multiprocessor implements fetch through its coherency mechanism; the Cray T3D implements fetch and deposit data transfers directly in the network interface, while in the T3E a combination between hardware (the E-registers) and system software (shmem_iput) performs fetch and deposit.

3 Differences in memory systems design

3.1 DEC 8400

The memory system design of a node in the DEC 8400 is based on a 3-level hierarchy of caches (L1, L2, L3) on the processor board and a shared dynamic memory (DRAM) on separate boards. The first two levels of caches are integrated on the DEC Alpha 21164 processor chip. The L1 cache is just 8KByte in size; it is a simple, data-only, direct mapped, write through cache, but the access latency is only two clocks (i.e. 6.6ns). The L1 cache can supply two operands per cycle, yielding a peak read bandwidth of 4.8 GByte/s. The L2 on-chip cache is 96KByte large, it is a 3-way associative unified instruction/data cache and has a write-back latency of 6 clocks (a peak bandwidth of 2.4 GByte/s). The write-back L3 cache comprises 4MB of fast SRAM (10ns) and is located on each processor board. According to specifications in [7] a 20 ns latency and a 915 MB/s bandwidth could be realized.

The DRAM memory of a DEC 8400 is built from memory modules, which are two-way interleaved. With four memory modules, a maximal interleaving of 8 is possible. Like a workstation, the DEC 8400 supports virtual memory. The vendor lists 176ns–928ns as an average latency for load operations from main memory, depending on how many memory-modules (i.e., memory banks) are installed and how many processors compete for memory access. The system used for these measurements contains four memory modules with a total of 4GByte. For large contiguous transfers, the DRAM memory is faster than the L3 caches of another processor; this fact indicates that the memory system includes modest stream support for large contiguous transfers.

In a symmetric multiprocessor both memory accesses and inter-processor data transfers involve the bus-based communication system. The DEC 8400 is built around a high speed system bus with 40-bit address and 256-bit data path. This bus is clocked at 75 MHz, a quarter of the clock frequency of the microprocessor, yielding a peak transfer-rate of 2.4 GByte/s across the system bus. This limit is reduced to a peak of 1.6 GByte/s under the best burst transfer protocol [9, 7]. A bus system puts limitations on scalability (fixed number of slots for processor and memory cards) but provides free broadcast, which significantly simplifies the implementation of globally coherent caches.

3.2 Cray T3D

In the Cray T3D, the caches play a much smaller role than in the DEC 8400. The processing node of the T3D consists of a DEC 21064 processor with just an on-chip L1 cache; the memory system includes a DRAM based memory system on the same board which is interfaced with fast ECL external support circuitry to the processor as well as to the communication system. The on-chip L1 cache is 8KB in size, data-only, direct mapped and write-through, read-allocate. The write path contains an on-chip write-back queue that buffers the high rate processor writes and coalesces them into 32 bytes entities if they are contiguous. The external circuitry supports contiguous reads with a read-ahead logic that can be turned on/off at program load time. With its completely different read and write paths and the external read-ahead logic, this memory system supports streamed access to large amounts of data. DRAM accesses within the same DRAM page are accelerated; see the technical data sheets or [1] for details regarding different speeds and bandwidths.

Remote accesses are performed to a network interface, which is also built from ECL gate arrays. Remote stores are directly captured from the write back queues, while remote loads can be performed in a transparent blocking manner at minimal speed, or somewhat faster through an external FIFO pre-fetch queue located in the support circuitry. Simultaneous communication is limited to one (or a few) communication partner(s), and there is a "per message" overhead for switching partners. The switching fabric is arranged as a 3D-torus of fast ($>$ 200MB/s) links[1].

Every node has some fetch/deposit circuitry that handles incoming remote operations (loads and stores) with their memory accesses on behalf of the communication system. These accesses can happen without involvement of the processor at the receiver node (i.e., there is no requirement to generate an interrupt). This circuitry can store incoming data words directly into the user space of the processing element, since both address and data are sent over the network. The on-chip cache of the main processor can be invalidated line by line as data is stored into local memory or can be invalidated entirely when the program reaches a synchronization point.

3.3 Cray T3E

The design of the memory system of the Cray T3E is similar to the T3D in the sense that it too includes support circuitry for non-local operations, stream support, and carefully tuned DRAM performance. However, this memory system inherits its cache structure from the DEC 21164 processor. The first two levels of caches (L1, L2) are integrated as on the DEC 8400, since the T3E is also based on the DEC Alpha 21164 processor chip. There is no L3 cache, but the memory system includes support for memory streams. For a complete a complete discussion about the design of the streaming units and the E- registers see [12].

Remote stores and remote loads are performed through a set of external E-registers located in the support circuitry around the DEC Alpha processor. For the moment, we rely on a first implementation of the shmem_iput and shmem_iget communication primitives provided by Cray Research for the Cray T3E. The communication network is similar to the Cray T3D but only faster and in addition every processor has its own network access; consequently, the raw link throughput improves significantly over the T3D.

4 Performance metrics and benchmarks

Several prototypes of early shared memory computers lead to the definition of a model that characterized communication and local memory system performance with a single parameter, the

[1]The actual implementation pairs two processing nodes with a single network access. Therefore the effective link speed seen by each of the two processors falls back to 70 MByte/s, but the largest machine size that can route AAPC permutations without congestion increases to 1024 nodes.

access latency. This model is known as the non-uniform memory access model (NUMA) and a large number of papers (e.g., [19]) analyze algorithm performance under that model. The NUMA model does not distinguish between local and remote memory and describes memory just as fast and slow memory in terms of access latency. The problem with the basic model is that it takes the reciprocal value of latency for bandwidth and fails to capture cases of pre-fetched data, pipelined transfers, or differences due to access patterns, i.e., contiguous, strided, and indexed accesses.

4.1 Memory and communication system bandwidth

For the communication and memory operations generated by a compiler for a data parallel language (e.g., HPF) the actual transfer bandwidth is more important than the access latency. Compilers like the Fx compiler have a good knowledge of the access pattern and are well able to use pipelining and arrange large transfers in an optimal manner. Often the only issue that counts is the rate at which a given amount of data can be moved for a particular case, and the latencies of a single memory access are irrelevant for most large scale scientific or data intensive applications.

The *copy transfer* model provides a bandwidth-oriented characterization of a memory system [15] that pays attention to memory access patterns. In this model each communication step is seen as a composition of basic copy transfers with known performance characteristics. If a given platform allows more than one way to implement a communication step, the modeled bandwidth metric is used to determine the best way to implement this communication step. The model used in [15] is asymptotic and applies only to large memory–to–memory transfers. We extend the copy transfer model by a working set parameter to capture the *temporal locality* of computation and communication methods that can be blocked for caches. Although we characterize local memory accesses, remote memory accesses, and entire communication operations such as transposes in the same framework, we make a simple distinction between local and remote copy transfers for an easy reference without loosing the generality to cover multiple levels of remote memory (e.g., in clusters of SMPs).

Local accesses are memory accesses that go to local caches or to DRAM memory in the same processing node or on the same processor board. Accesses that cause data to traverse a processor board boundary may be local or remote accesses. If the reading processor (consumer) and the writing processor (producer) of a copy transfer are the same, we call such an access local, despite the fact that it might cross the bus or a switch to reach a memory module of the memory system. In contrast, if the reading processor and writing processor of are different for a copy transfer, the memory accesses of that transfer serve the purpose of inter-processor communication and are therefore considered to be *remote accesses*. In the rest of the paper we sometimes refer to the bandwidth of remote loads and stores as communication performance. The bandwidth of copy loops with remote accesses are often called the throughput of a remote copy transfer.

Our investigation of transfer performance deemphasizes the different cache coherency models used in the symmetric multiprocessor (DEC 8400) and the distributed memory machines (T3D and T3E). The coherency models reflect only the difference between explicit and implicit communication in either a message passing or a shared memory machine. The coherency model might be responsible for differences in the instruction streams performing a given copy transfer, but not for the performance of the *most efficient* way to transfer data. On the T3D and the T3E there is no support for global cache coherence; in some cases it is better to turn even local coherence (within a node) off for a particular transfer operation and establish consistency later at a synchronization point.

4.2 Micro-benchmarks for the memory hierarchy

A few small, but highly optimized, memory system benchmark program measure the memory copy bandwidth of strided memory accesses for different working sets. The same micro-benchmark programs are used on the DEC 8400, the Cray T3D, and the T3E. Two different basic memory operations are examined, all of them operate on 64 bit double words.

Load Sum A load operation and an add-summing operation. All elements of the working set are used to accumulate a sum. [2]

Load/Store copy All data of the working set is copied by either loading it with a fixed stride and storing it contiguously, or by loading it contiguously and storing it with a fixed stride. Such copy transfers are common in transpose operations.

A third "Store Constant" benchmark was written as a dual to the "Load Sum" benchmark to evaluate store performance [14]. The resulting graphs did not add enough insight to the picture to warrant the space in a short conference paper. The store benchmarks confirmed the specified, default write-back policies of the caches and the proper function of the write back queues.

The Fx compiler produces low level C code for all interprocessor communication. Therefore we generated our benchmarks with the vendors' production C compilers rather than in assembly. However they are carefully crafted C routines, inspected for well scheduled machine code.

On the DEC 8400 and the T3E node, a memory bandwidth of 2400 MByte/s corresponds to the delivery of one 64-bit operand (8 Bytes) per CPU clock cycle (300 MHz). On the T3D a peak of 1200 MByte/s could be achieved when one operand is delivered per clock cycle (150 MHz). Both CPUs are supposed to reach their peak bandwidth when accessing data out of L1 cache. Unfortunately, not even the vendors' own compilers can generate the necessary instruction schedules for such a memory system benchmark. With a lot of careful C-code tuning and much handholding, we measured about half of the peak bandwidth for loads out of L1 cache with compiler generated benchmarks. The other levels of the memory hierarchy are much slower and not affected by this compiler problem.

Unless noted otherwise, the memory system benchmarks are executed on either a single processor of the shared memory machine (other processors idle), or on a single node partition of the distributed memory system.

5 Characterization of local and remote accesses

We use the two classes of micro-benchmarks to probe the multilevel memory hierarchy of the two machines in consideration. In particular we test the support of the memory system for *temporal locality* and *spatial locality* separately. The performance of the memory system is measured in access bandwidth for different strides and different working sets. The stride parameter shows how well caches and external stream logic help with read ahead and other means of improving bandwidth for accesses with spatial locality. The working set parameter shows how the memory

[2] Because modern processors often detect meaningless, artificial instruction streams, it is important that the operand of a load is actually used. Our loop is sufficiently unrolled to hide the latencies of the loads and floating point operations where they can be hidden. This transformation may effect some L1 cache numbers, because of issue bottlenecks, but allows us to report true, achievable bandwidth numbers for all other parts of the memory hierarchy.

hierarchy supports temporal locality, i.e. the effect of cache hits through reuse of recently accessed data. Our micro-benchmarks access all locations of the working set exactly once, but start with a primed cache for exactly that working set.

5.1 DEC 8400: Local memory system performance

Figure 1 shows the measured load bandwidth with a single processor of an otherwise unloaded DEC 8400. This figure depicts a comprehensive picture of the DEC 8400 memory hierarchy with bandwidth data for all levels of the memory hierarchy. The horizontal plateaus at 700 MByte/s, 120 MByte/s and 28 MByte/s show the level of memory system performance for different sizes of working sets. The grey bands in the graph indicate the largest working set in the graph with the characteristic performance of (i) L2 cache accesses, (ii) L3 cache accesses, or (iii) DRAM memory accesses. A slope of increasing performance marks the end of the access pattern axis toward lower strides. Its steepness indicates improved bandwidth for loads with contiguous accesses and accesses with small strides. A selection of even, odd, and prime strides permits to detect performance gains and losses due to a banked memory system. The flat performance for large working sets shows that the interleaved DRAM memory is handled internally and not visible — except for the cases along the border of the characteristic working sets for different cache levels.

Maximum memory performance for loads is approximately 1100 MByte/s in small working sets that fit entirely into the L1 cache. There is no penalty for higher strides, but the bandwidth becomes difficult to measure due to loop overhead and other constant overheads in the micro-benchmark and the performance ridge in the stride/working-set diagram falls off without immediate reason. In this zone the diagram rather reflects what is achievable by a compiler than what the hardware can do in theory. An application may experience a bandwidth of 1100 MByte/s out of L1 cache and 750 MByte/s out of L2 cache for large strides even if these bandwidths cannot be measured with a micro-benchmark. For loads out of L3 cache, we experience the peak of 600 MByte/s for contiguous accesses only, while strided accesses fall down to 120 MByte/s. This behavior is caused by the large cache lines at that level and by the read-ahead logic of the L2 cache which consumes load bandwidth unnecessarily to read-allocate the whole cache line, although only a single word is used.

The measurements in Figure 1 show the memory system performance of a single processor while other processors are idle. We also ran the same micro-benchmark with all four processors accessing local caches and DRAM memory independently at the same time. The performance results had a much larger variability but behaved as expected. Because the DEC 8400 has a shared memory system, a decrease in bandwidth for accesses to the shared DRAM memory is expected; the local caches continue working at full speed. The bandwidth for the L1, L2 and L3 cache stay almost the same, while the bandwidth for strided accesses to the DRAM memory decreases by about 8% for contiguous accesses and 25% for strided accesses under full load on all four processors.

5.2 DEC 8400: Remote accesses - memory system performance

To measure communication performance, these memory micro-benchmarks include remote memory accesses. On a symmetric shared memory multiprocessor, this mode of operation is achieved when one processor is producing data by storing it while another processor retrieves the same data elements. To ensure race-free behavior, reading takes place after the two processors reached a synchronization point. We measure the transfer bandwidth of the second processor while it is pulling the data over.

Figure 2 shows the remote memory performance for the DEC 8400. We see a multi-tiered performance picture depending on whether the working set can be held in fast SRAM cache or slower DRAM main memory. However, notice the entirely different scale for the access bandwidth! The maximal performance for remote memory accesses is down to 140 MByte/s from 1100 MByte/s for local accesses. For strided accesses out of DRAM, performance is about 22 MByte/s. These results are not surprising since remote accesses do not only cross the chip boundaries of processors but travel across the bus of the shared system and involve coherency protocols. The coherency mechanism detects misses on shared data and pulls the necessary cache lines over from a DRAM memory bank or from the caches of a remote processor board. The DEC 8400 does not have support for pushing data into memory or caches of a remote processor.

5.3 T3D: Local memory system performance

The T3D exhibits a much simpler picture; the performance of local memory accesses is shown in Figure 3. With distributed memories, the per-node performance of the local memory accesses looks exactly the same, whether just one or all 512 processors of an entire machine execute programs. The simpler node architecture with a two-tiered memory hierarchy exposes a sharp performance drop when we exceed the working set of the L1 cache and hit upon DRAM memory. An important characteristic of the Cray T3D node design is an external read-head logic that contributes to the steep slope of higher performance for contiguous DRAM accesses. Figure 3 reflects this design optimization: Contiguous loads from local DRAM memory on the Cray T3D are about 30% faster than in the DEC 8400 — despite the T3D's older design and slower clock rate.

5.4 T3D: Remote accesses - communication system performance

The memory on the T3D is truly distributed among the node processors. Despite the logical model of a global address space, performance considerations force the T3D compiler to move the data from remote memory to local memory before computing on it. The explicit transfers from local to remote memory in a distributed memory machine have the advantage that there is a choice between fetching data elements from remote memory vs. depositing them to remote memory.

We found that for compiler generated code, deposits based on remote stores are preferable for performance reasons. Naive, compiler generated remote loads are possible, but they result in communication performance that is an order of magnitude below the network bandwidth — unless the pre-fetch pipeline is used properly. We do not know of a node compiler that produces code for these loads, and programming all compiler communication primitives in assembly language seems too hard to be worthwhile. We concentrate on deposits in Figure 5, but for completeness we show some fetch data based on Cray's hand-coded shmem_iget primitives in Figure 4. (The Z axis of these two figures is scaled to allow easy comparison with Figure 2.)

In Section 6.2 we describe all copy transfers based on deposits to characterize communication performance of the Cray T3D.

5.5 T3E: Local memory access performance

Figure 6 depicts the memory bandwidth for the T3E. The simple node architecture is reflected in the graph: there are three

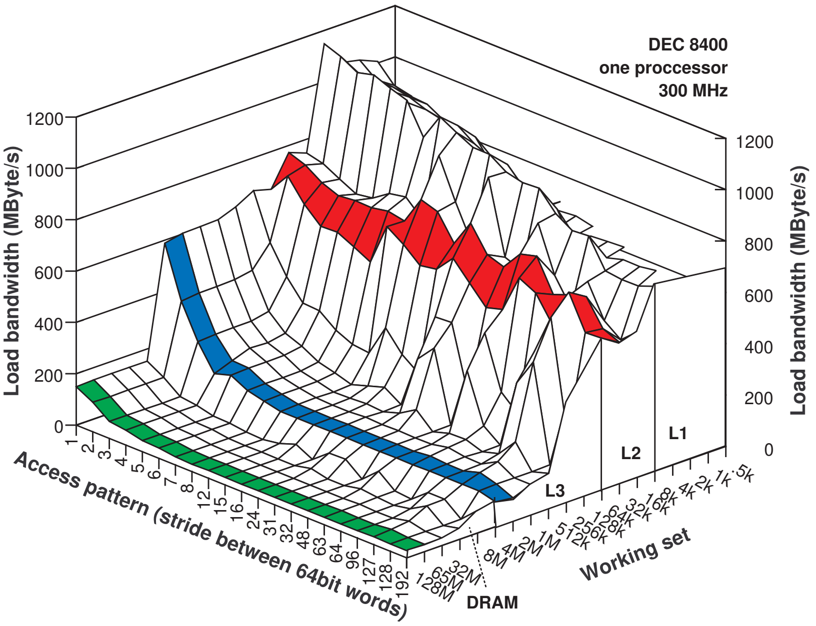

Figure 1: Bandwidth of loads for different access patterns (strides) and different working sets on the DEC 8400. One processor runs memory benchmark, other three processors are idle.

performance levels, exposing a sharp drop from cache access to DRAM memory access when we exceed the working set of the L1/L2 caches. Not surprisingly, the local memory access performance of the T3E resembles the picture of the DEC 8400 in the performance of its L1 and L2 caches. Any differences for small working sets are due to idiosynchrocies in the measurement environment, i.e. the lack of inlined timer functions, and the (still) larger loop overhead inflicted on programs by the Cray compiler.

The important differences to the DEC 8400 are (i) the lack of L3 caches (ii) and the better support for contiguous access streams from main memory. While the DEC 8400 achieves just 150 MByte/s for contiguous loads out of DRAM main memory, the T3E node is capable of load transfers of up to 430 MByte/s. In other words, the memory system bandwidth of DRAM on the T3E compares well with the memory system performance out of the local L3 cache on the DEC 8400 (which is about 600 MByte/s). Again this performance is due to the support hardware for streaming on the T3E [3].

Between the two generations of Cray machines, the clock rate double. Yet although we observe an improvement of close to a factor of two for access to the L1/L2 cache and streamed access to DRAM, we see no improvement for strided accesses out of DRAM. These accesses seem stuck at about 42 MByte/s on the T3E (43 MByte/s on the T3D). Although this performance still compares favorably to the DEC 8400 performance of 28 MByte/s, it may nevertheless spell disappointments for some applications that move from the T3D to the T3E.

5.6 T3E: Remote memory system performance

The T3E represents a further step from a distributed memory machine to a coherent shared memory multi-processor. Its global address space can now map the entire memory in the machine, but

[3]The importance of the streaming support was also demonstrated by an earlier test-vehicle that disabled streaming support. On this machine, the measured bandwidth is about 120 MByte/s.

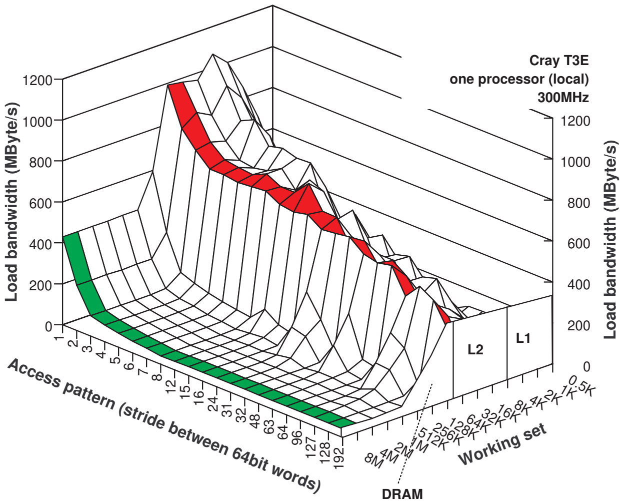

Figure 6: Cray T3E: Load bandwidth for different access patterns (strides) and different working sets. One processor runs memory benchmark, other three processors idle.

a compiler is still forced to copy the data from remote memory to local memory or at least the 512 E-registers before computing on it. The characterizations reported in this section are based on the *shmem_iget* and *shmem_iput* routines provided by Cray which we treated as black box building blocks.

Unlike on the T3D, the deposit model (i.e., using remote stores) enjoys no performance advantages over the fetch model (i.e., using remote loads). Figures 7 and 8 illustrate this point. Both modes of operation perform impressively at 350 MByte/sec for contiguous data transfers (stride 1). This is more than four times the bandwidth in the Cray T3D and twice the bandwidth in the DEC 8400. Part of the impressive improvement over

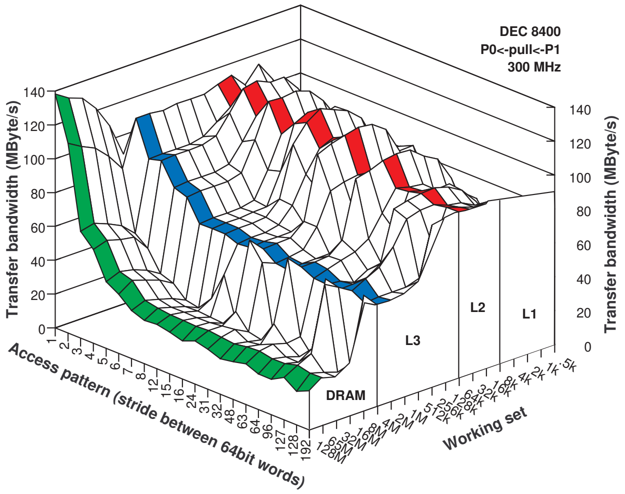

Figure 2: DEC 8400: Remote load bandwidth for different access patterns (strides) and different working sets. Throughput for transfers from processor P1 to processor P0.

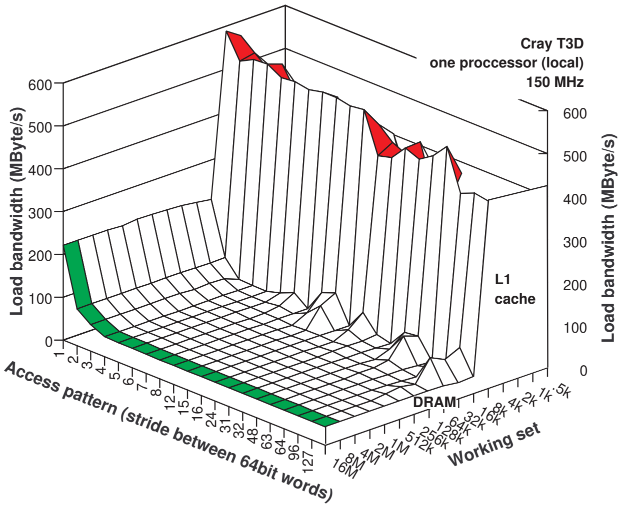

Figure 3: Cray T3D: Load bandwidth for different access patterns (strides) and different working sets. One processor runs memory benchmark, other three processors idle.

the T3D is that in a T3E every processor has a network node, while in the T3D two processor share a network node. However, the gap between contiguous streams and strided accesses has widened to over a factor of two. The ripples in Figure 8 indicate that the memory system at the destination node has difficulties storing data at full network speed if the same bank is hit in consecutive receives. This observation indicates that fetches are more advantageous for even strides than deposits. Therefore the back-end of the Fx compiler should generate fetch code for the T3E while sticking with deposit code for the T3D.

6 Comparison of copy transfer performance

Scientific application codes for parallel computers handle large arrays of distributed data. We encounter a few characteristic communication patterns when arrays are re-distributed. For many distributions, every processor must exchange data with every other processor. These "all-to-all personalized communication" (AAPC) operations have received considerable interest by researchers. Transposes are one example of an AAPC; since the actual execution of these operation does not depend on the data values, a transpose is a good generic test case to assess the joint strengths and weaknesses of the communication and memory systems. The performance of array transposes is largely determined by the ability of the DRAM memory system to handle local and remote copy transfers with strides (dense matrices) or indices (sparse matrices). Therefore, this characterization of the memory system goes beyond pure loads and stores bandwidth to measure the copy bandwidth. Since end-to-end transfers in compiled parallel programs often involve strides, we graph access patterns for various strides, from contiguous up to constant strides of 64.

In many computation and communication steps the amount of data (re)distributed, called the communication working set, is much larger than the total amount of cache memory in the machine. With a largest cache size of 8KByte on the T3D and 4 MByte on the DEC 8400, a working set of 65 MByte per processor is sufficient to force every copy operation to go from DRAM memory to DRAM memory. In this section we select a few key working sets from our general memory characterizations in Figures 1 to 6 and discuss the memory system performance in more detail. The characterizations are according to the basic copy transfer model [15] (no working set parameter, full copy operation - read and write) and focus on large copy transfers with no reuse of data and without temporal locality in the caches.

6.1 Copy transfers in local memory

Figures 9 to 11 depict the measured throughput of a local memory copy with either strided loads or strided stores. On the DEC 8400 and the T3E, there is little difference in performance between local copies using strided loads versus local copies using strided stores. The caches help equally with read-ahead and write-behind to accelerate transfers with low strides. A DEC 8400 can copy contiguous blocks at about 57 MByte/s and strided data at about 18 MByte/s. Given the high clock rate and the high nominal performance of the bus system of the DEC 8400, these memory bandwidth numbers are rather disappointing. On the T3D we see a significant improvement in bandwidth for contiguous loads due to the read-ahead logic. The T3D node architecture is able to copy contiguous memory blocks at a 100 MByte/s despite its lower clock rate of 150MHz. Furthermore, well pipelined writes through a write-back queue allow strided stores at up to 70 MByte/s, which is almost three times the speed of the DEC 8400. The T3E has an impressive copy bandwidth of 200 MByte/s for contiguous blocks, but unfortunately the picture for strided access resembles more the DEC 8400 than the T3D. The write-back caches prohibit efficient strided stores. This observation holds for compiler generated copy loops and may not be accurate for assembler generated programs that manage to work around the default write-back cache policy. The good performance of some communication libraries suggests that assembly loops with adequate prefetching can further activate the streaming support and perform much better than a C program compiled with the vendor's compiler.

For the DEC 8400 the bandwidth results for large transfers from and to DRAM memory (basic copy transfer model) show only part of the picture. This machine has a distinct four-level memory hierarchy; if a transpose operation can keep its working set in the a given level of the memory hierarchy, big performance advantages may be obtained. The L1 and L2 caches are too

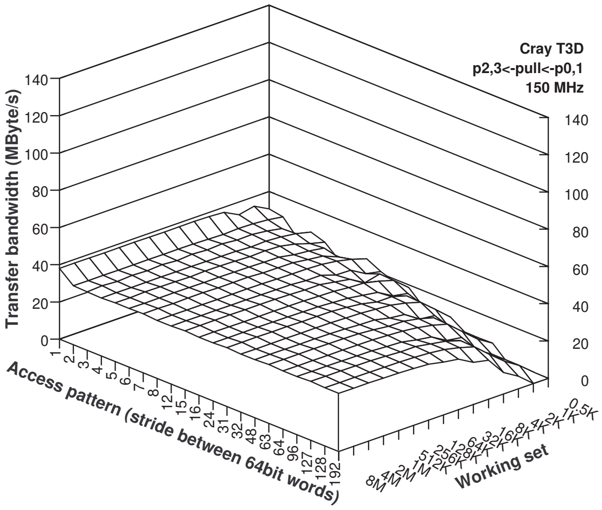

Figure 4: Cray T3D under fetch model: Transfer bandwidth for different access patterns (strides) and different working sets obtained through remote loads.

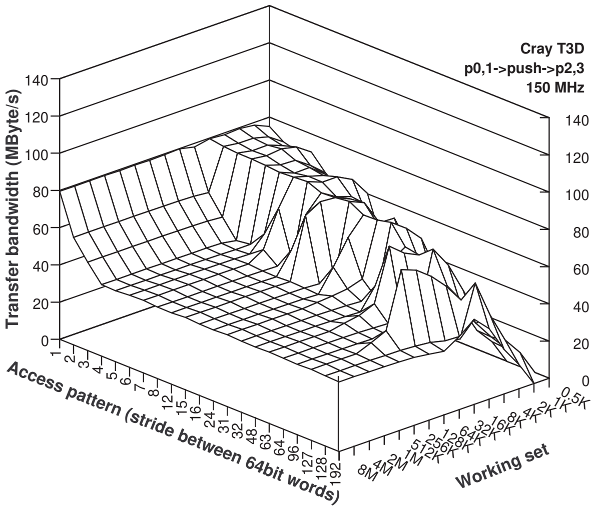

Figure 5: Cray T3D under deposit model: Transfer bandwidth for different access patterns (strides) and different working sets, obtained through remote stores.

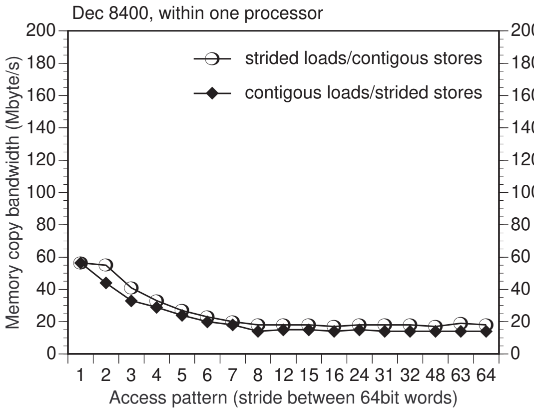

Figure 9: Measured performance of the local memory system for large transfer, with either strided loads or strided stores for the DEC 8400.

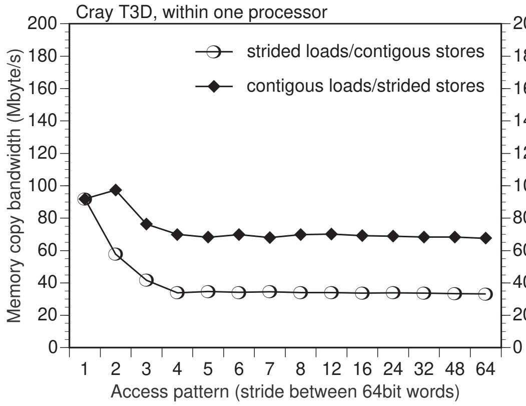

Figure 10: Measured performance of the local memory system for large transfer, with either strided loads or strided stores for the Cray T3D.

small to make this option attractive, but blocked algorithms for the L3 caches could yield interesting performance numbers. The characterization in the extended model (Figure 1 in Section 5.1) contains the data to assess the performance gain of such blocked algorithms. Just follow the marked band for the L3 working set (4 MByte) in the figure and see the potential performance for blocked "all cache" communication.

6.2 Copy transfers from/to remote memory

On the Cray T3D and the Cray T3E all remote store/load operations are explicitly programmed, while on the DEC 8400 the cache coherency protocols decide to pull/push data between processors for certain load and store operations. In this paper we focus solely on throughput for moving data from A to B, therefore we can easily compare the two architectures based on remote copy transfers. On the Cray T3D and the Cray T3E, we have the option of performing either strided loads or strided stores to implement a particular transpose operation. For a characterization most relevant to scientific applications with large distributed arrays, it is sufficient to compare transfers with big working sets. In Figures 12 and 14 we graph transfer performance for a 65 MByte working set on the DEC 8400, the Cray T3D and the T3E. The DEC 8400 and the Cray T3D handle contiguous data at about the same speed. For strided data however the T3D has a clear advantage as higher strides get involved. If copy transfers of transposes are properly optimized using strided stores on the T3D, they can be performed at about 55 MByte/s, while on a DEC 8400 the bandwidth of such transfers is limited to about 20 MByte/s. The performance on the T3E is a class on its own. It can transfer 350 MByte/s for contiguous blocks and falls down to 140 MByte/s or 70 MByte/s for strided accesses (depending on how the transfer is programmed). The ripples for odd strides suggest that the memory system at the destination node needs to avoid bank conflicts to keep up with the network speed. This is not surprising, since in the most recent generations of parallel systems interconnect the network performance has improved faster than memory system performance.

Similar to the local memory to memory copy performance, the remote copy transfer performance of the DEC 8400 depends on

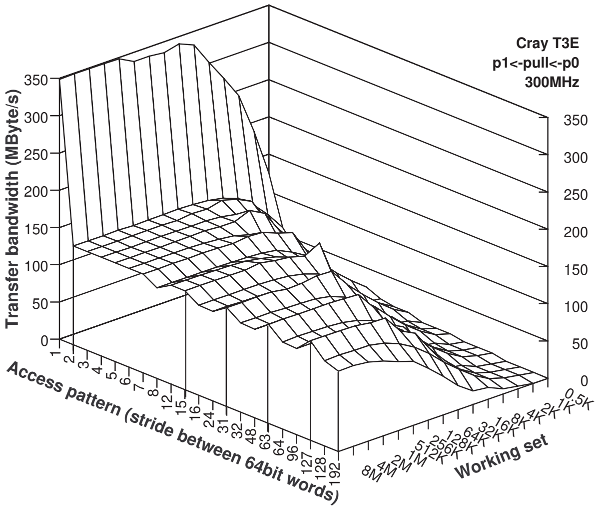

Figure 7: Cray T3E under fetch model: Transfer bandwidth for different access patterns (strides) and different working sets obtained through remote loads.

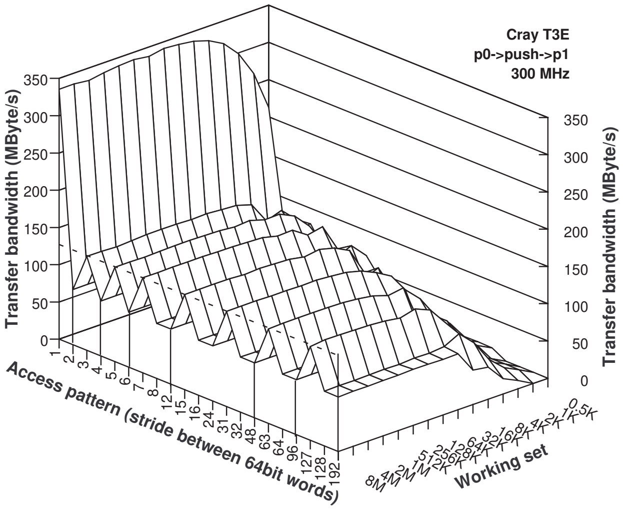

Figure 8: Cray T3E under deposit model: Transfer bandwidth for different access patterns (strides) and different working sets, obtained through remote stores.

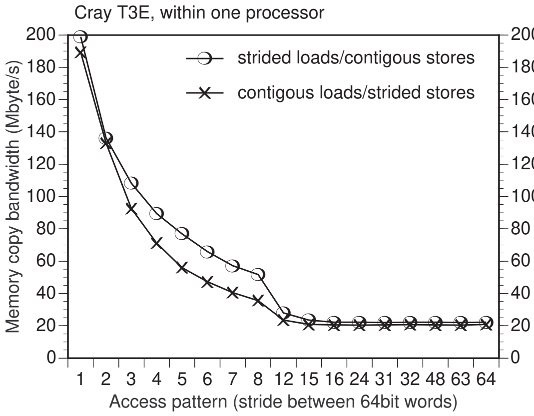

Figure 11: Measured performance of the local memory system for large transfer, with either strided loads or strided stores for the Cray T3E.

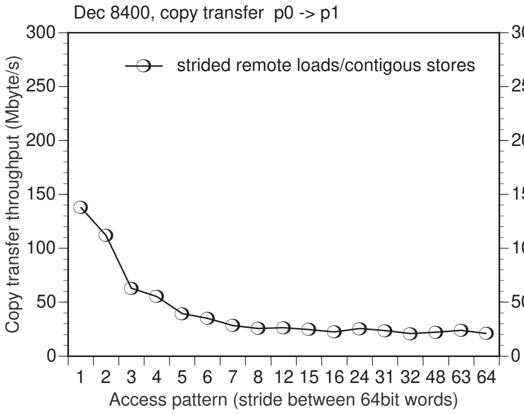

Figure 12: Measured performance of large remote copy transfers (i.e. between processors) on the DEC 8400, for different strides.

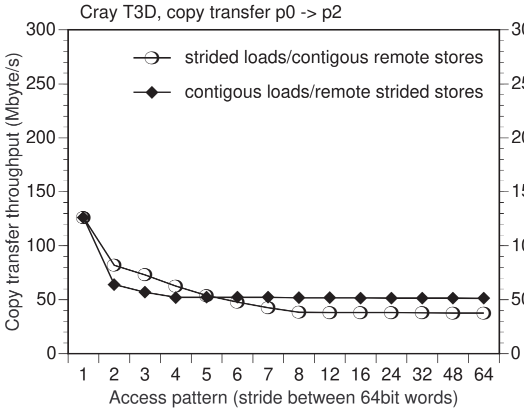

Figure 13: Measured performance of large remote copy transfers (i.e. between processors) on the Cray T3D, for different strides.

the characteristics of a memory hierarchy and delivers different performance for different working sets entirely within the L3 cache. Again the full characterizations with different working sets in Figure 2 reveal that strided remote transfers can be done faster from L3 cache if a global communication operation can be blocked. The characterization quantifies the advantage for this interesting compiler optimization.

Our measured performance for a remote copy transfer sets an upper bound on memory system performance in parallel applications since our simple measurements fail to capture some sharing of certain communication resources on both machines. On the T3D we used only one out of the two processors that are sharing a common network access (and with it a link). On the DEC 8400, we measured just one single processor or a single producer/consumer pair while copy transfers between two processors with no other workload executed on the machine. On the T3E there is no contention, and the remote copy transfer performance is expected to scale up to a 512 processor ($8 \times 8 \times 8$) torus, before bisection limits become visible in transposes (i.e., AAPC patterns).

7 Evaluation of an application kernel

For an evaluation of our memory characterization within an application generated by our compiler we coded an 2D-FFT appli-

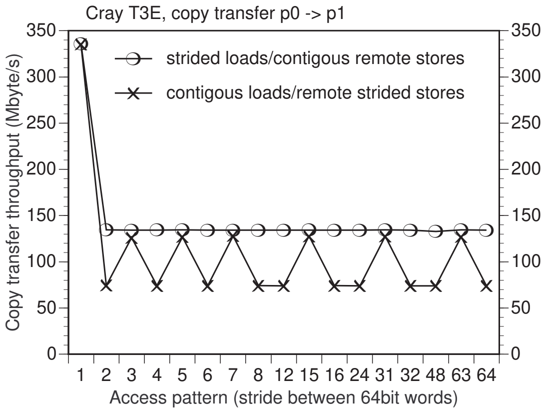

Figure 14: Measured performance of large remote copy transfers (i.e. between processors) on the Cray T3E for different strides.

cation kernel in Fortran and compiled it with the Fx compiler.

7.1 Implementation of 2D-FFT

Like all high performance multidimensional FFTs, the Fx code must achieve locality of reference in both computation steps, the row and column FFTs. Therefore our 2D-FFT has four characteristic steps: local row FFTs (1D), global row-column transpose, local column FFTs (1D), global column-row transpose. The transposes are indicated to the compiler by an assignment statement of two distributed arrays. The implementation operates on complex numbers represented as a pair of 64bit, double precision floating point numbers. Since Fortran allows to call external functions, we can rely on the best available library routine for a local 1D-FFT. Both vendors provide such an optimized FFT routine as part of their scientific library packages.

The quality of our compiler-generated code was verified with a fine-tuned 2D-FFT application kernel written in C. This “best possible” implementation served as reference, and the Fx code performs within 3%.

7.2 2D-FFT overall application performance

The 2D-FFT application kernels are executed for different problem sizes on all three systems, the DEC 8400, the Cray T3D, and the Cray T3E. The performance measurements on four processors give a good impression of the impact of local and remote memory systems performance on the overall performance of the 2D-FFT.

Looking at the performance of the Fx program in Figure 15 we see that the DEC 8400 delivers a slightly higher performance than the Cray T3D on all different problem sizes of the 2D-FFT. For a 256×256 point 2D-FFT the Cray has an overall performance of 133 MFlop/s with four processors while the DEC 8400 peaks with about 220 MFlop/s, an improvement in performance of about 75%. This result is somewhat surprising, given that a processor of the DEC 8400 is clocked twice as fast, has 3 levels of caches, and can issue up to twice as many instructions per clock than the T3D processor. With its current communication library the T3E performs at 330 MFlop/s, about 50% above the DEC 8400.

7.3 2D-FFT computation and communication performance

For a more detailed analysis, Figures 16 and 17 depict the performance figures for computation and communication separately.

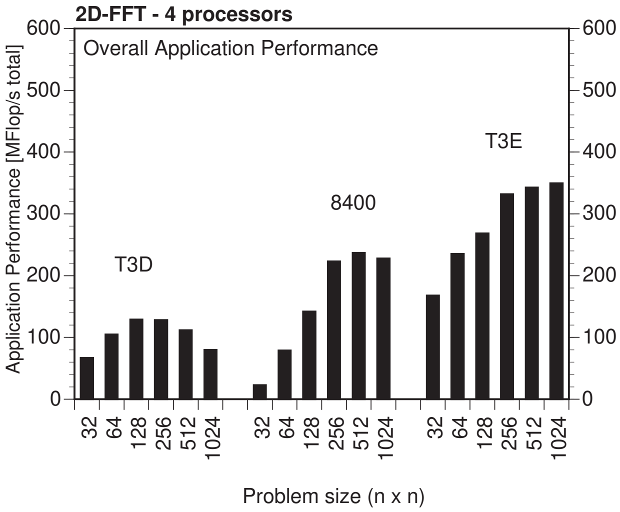

Figure 15: Total application performance of the 2D-FFT benchmark on 4 processors of a Cray T3D, a DEC 8400, and a Cray T3E.

For all problem sizes, the overall per processor performance depends on both factors, the local computation performance and the communication performance of the transpose. The parallel 2D-FFT application kernel is well balanced with regard to computation and communication work.

The computation performance in Figure 16 is heavily influenced by the memory hierarchy and the cache structure of the processor node. From the graphed local computation performance numbers (in MFlop/s) it is evident that the sum of local computation performance over all four processors is more than a factor 2.5 higher on the DEC 8400 than on the Cray T3D. Looking specifically at large problem sizes for the 2D-FFT (e.g., 1024×1024, a million elements), we realize that the performance on the T3D falls off with large problems, while the performance on the DEC 8400 stays nearly at the same level. This performance advantage of the DEC 8400 is due to the higher clock frequency and the better caches in its memory hierarchy. The large L2 and L3 caches allow the DEC 8400 to execute the row and column FFTs out of cache — rather than out of DRAM memory — for the problem sizes above 256×256 elements. The T3E can deliver even higher local performance (up to 200 MFlop/s per processor) possibly due to its better memory system with streaming units. Since we measured the routine in scientific library offered by the vendor as a black box, part of that improvement could also be attributed to better coding of the 1D-FFT primitive.

The benefit of a faster processor design on the DEC 8400 is reduced by a communication system that runs at approximately the same performance level as the communication system on the Cray T3D. The good performance for local 1D-FFTs in Figure 16 is overshadowed by the large communication overhead during the global transpose steps between row and column FFTs. To sustain the 2.5 fold performance improvement caused by its faster microprocessor and memory hierarchy, a similar performance improvement for the communication system would be required. This is not the case for a DEC 8400, which offers only lower or similar bandwidth for communication in transposes, thus limiting the overall performance to a factor below two over the T3D. From the local and remote memory performance characterizations (Sections 5 and 6) we anticipated that the T3E can fully utilize its better processor and communication system to achieve a performance improvement of at least a factor of 3 over

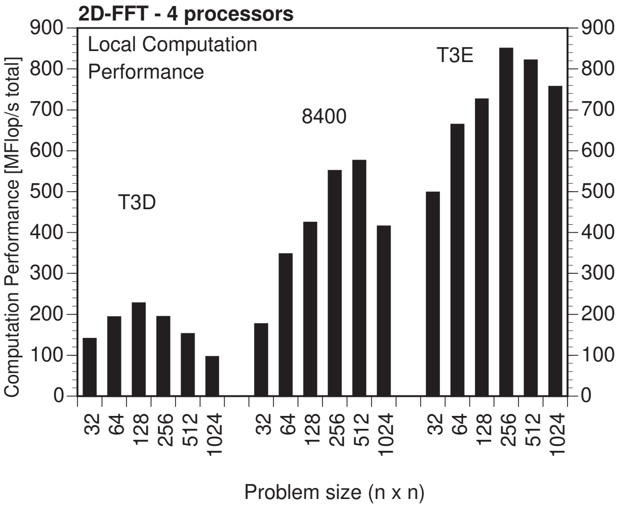

Figure 16: Local computation performance of the 2D-FFT benchmark on 4 processors of a Cray T3D, a DEC 8400, and a Cray T3E.

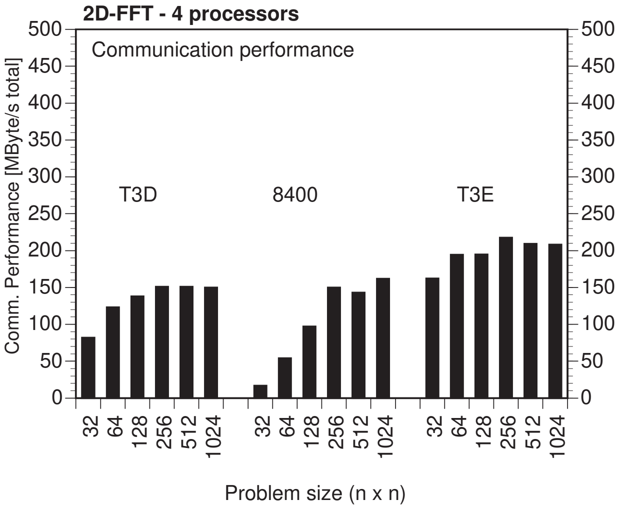

Figure 17: Communication performance in the transposes of the 2D-FFT benchmark on 4 processors of a Cray T3D, a DEC 8400, and a Cray T3E.

its predecessor, the T3D. Due to a mismatch between the required memory access patterns for the transpose of a distributed, two dimensional array of complex numbers and the simple capabilities of the shmem_iput primitive, the expected performance could not be achieved at this time. A rewrite of this crucial primitive is planned to address the performance problem.

8 Other factors

For our characterization we focus solely on the performance of the processing nodes in both machines, with a particular emphasis on the performance of the local and remote memory system. Other factors for a comparison include ease of use, scalability, and cost.

On the DEC 8400, a single OSF/1 UNIX operating system kernel manages all resources and schedules processes for all four processors. The processors are fully time-shared, and therefore the system has the "look and feel" of a big, fast workstation. This is the main difference to the Cray T3D, where considerable skills are required to manage job execution and processor allocation properly. The Cray T3D executes all required Unix services on a Cray C90 vector supercomputer serving as a front end. The processing nodes execute only a user program and a small run time kernel. In the Cray T3E, a micro-kernel executes on every node and some service nodes transparently provide all operating system services of UNICOS 8.0.

The Cray T3D and the T3E are built as highly scalable machines. Parallel programs with just four processors are only "modestly" parallel for a Cray T3D or a T3E, given the potential parallelism of executing them on a full size machine with 512 processors. On a T3D, the "massively" parallel performance of our compiler generated 2D-FFT written in Fx Fortran stays around 20 MFlop/s per processor and reaches a total performance of 8.75 GFlops when run on 512 processors. The code shows almost linear scalability from 16 to 512 nodes [16]. Based on our model of memory and communication system performance we expect to report similar scalability and a sustained aggregate performance for a 2D-FFT of about 20 GFlops, once we run the code on a full-size machine.

Scalability to a large number of processors was not a target for the DEC 8400 series of machines and did not constrain the design. According to [7] a DEC8400 is limited to 12 processors and/or 14 GBytes of memory. The maximal configurations are subject to the limitation of nine bus slots in the backplane with room for eight modules with either 2 CPUs or 2GByte of memory each.

9 Conclusions

The DEC 8400, the Cray T3D, and the Cray T3E support remote memory operations as means of inter-processor communication. Despite their differences in the support of global cache coherency and startup overheads for communication, the performance of these machines can be characterized accurately with the copy-transfer model, a simple bandwidth oriented model for memory system performance under different strides. The multi-level cache and memory hierarchies of machines like the DEC8400 make it necessary, however, to extend the basic copy transfer model with an additional working set parameter to capture the potential gains through operand reuse and blocking.

In this characterization and comparison of the local and remote memory system performance we observe a few interesting characteristics on three machines. These machines belong to different generations and were introduced over a 3-year period (Fall 1993 (T3D), Spring 1995 (DEC), Fall 1996 (T3E)), and looking at the data sheets, we notice a progression towards faster clock rates, larger caches, and faster bus or network interconnect speeds. However, these improvements do not translate into comparable across the board improvements in processing power, better memory systems performance, and better communication throughput with each generation. Comparing the T3D and T3E, a 2.5 fold increase in processing power is matched by a 2 fold improvement of local memory copy bandwidth and a 2.75 fold improvement in remote memory copy bandwidth. The DEC 8400, on the other hand, shows a significant improvement in local compute performance over the T3D (due to employing the next generation microprocessor with doubled clock rate), but its memory system performance for large local and remote copies remains still inferior to the older T3D. Large strided remote transfers achieve only 22 MByte/s per processor on the DEC 8400, a factor of 2.5 less than the 55 MByte/s measured in the T3D, or a factor of 6.5 less than the 140 MByte/s measured in the T3E. An exception to these performance differences are the contiguous accesses and small strides where T3D and DEC 8400 perform alike – but still a factor 2 below the T3E. We attribute those differences to the memory systems design philosophies, i.e. a cache focus on the DEC machine and a streams focus on the Cray machines.

Among the factors other than performance, we must mention ease of use (almost full coherency) and better entry level costs for bus based symmetric multiprocessors. We managed to re-target our parallelizing Fx compiler within an afternoon to the DEC 8400, while it took considerably longer to come up with a good T3D back-end for Fx, and we continue to refine the T3E back-end . However, looking at our memory system characterization and the performance of a simple 2D-FFT application kernel, we find the overall application performance on a DEC 8400 is only about a factor of 1.5 higher, on average, than on a Cray T3D partition with the same number of processors.

On the systems studied here, there exist big performance differences depending on how a memory access or communication operation is done by a compiler or a programmer. On all three machines, the straight remote memory copy bandwidth (or communication performance) is equal to or higher than the local copy performance. Therefore in communication steps like global array transposes, using local memory copies to rearrange access patterns, or pack communication buffers or blocks, never pays off. On the T3D, pulling data (fetch model) proves to be consistently inferior than pushing data (deposit model). On the T3E, pulling data seems to work equally well (odd strides) or better (even strides) than pushing data. On the DEC 8400, the implicit coherency mechanism limits the user to pulling; however there are large L3 caches that may support blocking, and if a global communication operation can be partitioned into sub-blocks, cache to cache transfers might perform better than remote memory copies.

The existence of these architectural tradeoffs emphasizes again that it is necessary to build better practical models of memory systems performance. These models can no longer be derived from the data sheets and hardware descriptions but require measurements of micro benchmarks to capture the combined effects of the streaming units, pre-fetch units, and other accelerators. Realistic models based on measurement provide the accurate understanding of memory system performance that is necessary to generate efficient code by compilation tools and application programmers.

Acknowledgments

We thank Ken Mortensen, Rich Graham, Sergiu Sanielevici and Steven Scott of Cray Research for their helpful hints as well as for their assistance in obtaining the performance data for the Cray T3E. We are also very grateful to George Cebulka, the system administrator for PSC's DEC 8400 at the time of our experiments, for promptly handling all our requests, to the reviewers for their comments and to Tryggve Fossum of DEC for clarifying some measurements on the DEC 8400.

Machine resources used for measurements included in this paper were provided by Cray Research and the Pittsburgh Supercomputing Center. We thank Monica Lam of Stanford University for providing us with access to a reference DEC8400 multiprocessor to repeat some measurements.

References

[1] D. Adams. Cray T3D system architecture overview. Technical report, Cray Research Inc., September 1993.

[2] R. Arpaci, D. Culler, A. Krishnamurthy, S. Steve Steinberg, and K. Yelick. Empirical evaluation of the Cray-T3D. In *Proc. 22nd Intl. Symposium on Computer Architecture*, pages 320–331, Santa Marguerita di Ligure, June 1995. ACM.

[3] R. Barriuso and Knies A. SHMEM user's guide for C. Technical report, Cray Research Inc., June 20 1994. Revision 2.1.

[4] F. Chism. Communication latency and bandwidth on the Cray T3E. In *Proc. 10th Intl. Parallel Processing Symposium*, Slides, Vendor Presentation, Honolulu, HI, April 1996. IEEE.

[5] Z. Cvetanovic and D. Bhandarkar. Performance characterization of the Alpha 21164 microprocessor using TP and SPEC workloads. In *Proc. 2nd High Performance Computer Architecture Conference*, pages 270–279, San Jose, Jan 1996. IEEE.

[6] Digital Equipment Corp., Maynard MA. *Alpha 21164 Microprocessor, Hardware Reference Manual*, 1995. EC-QAEQB-TE.

[7] Digital Equipment Corp., Maynard MA. *AlphaServer 8200 and AlphaServer 8400, Technical Summary*, 1995. BC-N446-10.

[8] Richard Sites (editor). *Alpha Architecture Reference Manual*. Digital Equipment Corp., Burlington MA, 1992. EY-L520E-DP.

[9] D. Fenwick, D. Foley, W. Gist, S. VanDoren, and D. Wissell. The AlphaServer 8000 series: High-end server platform development. *Digital Technical Journal Vol.*, 7(43), Spring 1995.

[10] V. Karamcheti and A. Chien. A comparison of architectural support for messaging on the TMC CM-5 and Cray T3D. In *Proc. 22nd Intl. Symposium on Computer Architecture*, pages 298–307, Santa Marguerita di Ligure, June 1995. ACM.

[11] C. Koelbel, D. Loveman, G. Steele, and M. Zosel. *The High Performance Fortran Handbook*. The MIT Press, Cambridge, MA, 1994.

[12] S. Scott. Synchronization and communication in the Cray T3E multiprocessor. In *Proc. 7th. International Conference on Architectural Support for Programming Languages and Operating Systems*, Boston, MA, Oct 1996. ACM.

[13] J. Stichnoth and T. Gross. A communication backend for parallel language compilers. In *Proceedings of 8th Workshop on Languages and Compilers for Parallel Computing*, pages 224–236, Columbus, Ohio, August 1995. Springer Verlag.

[14] T. Stricker. Direct deposit - A communication infrastructure for parallel and distributed programs. Ph.D. Thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, TR CMU-CS-96-xxx.

[15] T. Stricker and T. Gross. Optimizing memory system performance for communication in parallel computers. In *Proc. 22nd Intl. Symp. on Computer Architecture*, pages 308–319, Portofino, Italy, June 1995. ACM/IEEE.

[16] T. Stricker, J. Stichnoth, D. O'Hallaron, S. Hinrichs, and T. Gross. Decoupling synchronization and data transfer in message passing systems of parallel computers. In *Proc. Intl. Conf. on Supercomputing*, pages 1–10, Barcelona, July 1995. ACM.

[17] J. Subhlok, J. Stichnoth, D. O'Hallaron, and T. Gross. Programming task and data parallelism on a multicomputer. In *Proc. 4th ACM Symp. on Principles and Practice of Parallel Prog. (PPoPP)*, pages 13–22, May 1993.

[18] H. Wasserman. Benchmark tests on the Digital Equipment Corporation Alpha AXP 21164-based AlphaServer 8400. In *Proc. 1996 International Conference on Supercomputing*, pages 333–340, Philadelphia, May 1996. ACM.

[19] X. Zhang and X. Qin. Performance prediction and evaluation of parallel processing on a NUMA multiprocessor. *IEEE Transactions on Software Engineering*, 17(10):1059–68, Oct 1991.