Memory Performance and SPEC OpenMP Scalability on Quad-Socket x86_64 Systems

Daniel Molka, Robert Schöne, Daniel Hackenberg, and Matthias S. Müller

Center for Information Services and High Performance Computing (ZIH)
Technische Universität Dresden, 01062 Dresden, Germany
{`daniel.molka`, `robert.schoene`, `daniel.hackenberg`, `matthias.mueller`}
`@tu-dresden.de`

Abstract. Because of the continuous trend towards higher core counts, parallelization is mandatory for many application domains beyond the traditional HPC sector. Current commodity servers comprise up to 48 processor cores in configurations with only four sockets. Those shared memory systems have distinct NUMA characteristics. The exact location of data within the memory system significantly affects both access latency and bandwidth. Therefore, NUMA aware memory allocation and scheduling are highly performance relevant issues. In this paper we use low-level microbenchmarks to compare two state-of-the-art quad-socket systems with x86_64 processors from AMD and Intel. We then investigate the performance of the application based OpenMP benchmark suite SPEC OMPM2001. Our analysis shows how these benchmarks scale on shared memory systems with up to 48 cores and how scalability correlates with the previously determined characteristics of the memory hierarchy. Furthermore, we demonstrate how the processor interconnects influence the benchmark results.

1 Introduction

The performance demands of applications continuously grow and to face this challenge, increasing core counts have become the dominant factor in microprocessor development. For server and HPC workloads with high degrees of parallelism, it is common to use multiple processors to further improve performance. Such multi-socket, multi-core systems are usually implemented as cache coherent shared memory systems. They provide a global view on the available memory, allowing multiple threads to share a single address space and exchange data via the jointly used memory. Based on cache coherent shared memory, the language extension OpenMP enables developers to easily parallelize applications written in C/C++ or Fortran. OpenMP provides an API that allows programmers for example to define parallel regions and to parallelize loops. Within a parallel region, multiple threads are executed concurrently and the workload can be distributed among them. The number of threads is determined at runtime. Therefore, one executable can be flexibly used on systems of different scale and still utilize all available cores. However, this simple programming model does not consider any hardware specifics. The shared memory can be physically distributed among the sockets in the system. The non-uniform memory access (NUMA) results in different memory latencies and bandwidths depending on the location of the data in the system. Therefore, NUMA optimized memory allocation and the placement of threads that access the data are strongly performance relevant issues. Additionally, data can be cached at different locations and resources are shared by multiple cores what increases the complexity even further. In this paper we therefore use sophisticated low-level memory benchmarks to characterize the performance of memory accesses and data transfers between caches on two quad-socket NUMA systems using multi-core x86_64 processors from AMD and Intel. Furthermore, we use SPEC OMPM2001 to investigate how these performance characteristics influence the performance and scalability of typical shared memory high performance computing application.

2 Related Work

The microbenchmarks we use to examine the latencies and bandwidths between the different memory components were first introduced in [6]. They have been further adapted to support additional coherence states and to collect fundamental performance data of two socket x86_64 servers in [4]. SPEC OMP2001 is a well-established shared memory benchmark suite. Saito et al. introduce these benchmarks and present a scalability analysis for up to 128 threads running SPEC OMP benchmarks on medium and large datasets [8]. Aslot et al. use a quad-processor UltraSPARC II system to gather performance related information for specific code sections of the SPEC OMP benchmark applications [1]. They list the most time consuming code regions and discuss scalability issues. Moreover, Müller et al. present scalability and performance results for SPEC OMPL2001 and SPEC HPC2002 benchmarks [7]. Fürlinger et al. analyze the scalability of SPEC OMPM2001/OMPL2001 benchmarks on a 32 processor Itanium system and use the profiling tool `ompP` to break scalability issues down to reasons like thread management, imbalances, and synchronization [3]. All these analyses have been performed on systems with single-core processors and therefore do not provide performance insights for systems consisting of multi-core processors with shared L3 cache and integrated memory controllers.

3 Test Systems

We analyze two quad-socket cache coherent NUMA systems with processors from Intel and AMD. Table 1 summarizes the hardware configuration.

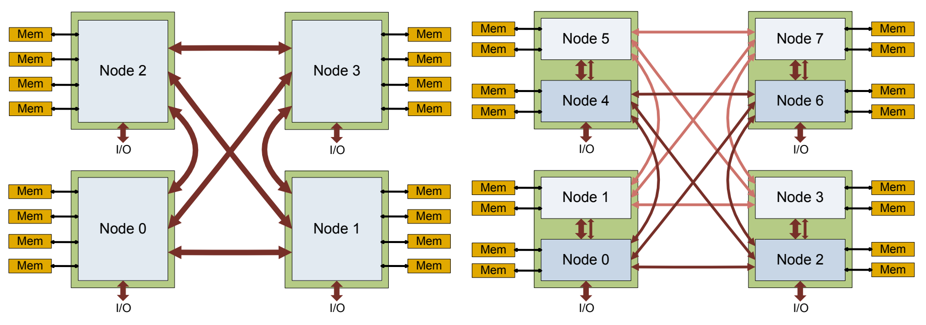
The 8-core Intel Xeon 7500 series processors consist of a monolithic die and feature 24 MiB of shared L3 cache. The two integrated memory controllers each provide two scalable memory interface (SMI) channels. Connected to each SMI channel is a scalable memory buffer (SMB) that controls the DDR3 memory. Each processor features 4 QuickPath Interconnect (QPI) links. One link is used

Table 1: Hardware configuration of test systems

System	Nehalem-EX	Magny Cours
Processors	4x Intel Xeon X7560	4x AMD Opteron 6172
Cores	32 (SMT disabled)	48
Core clock	2.266 GHz (w/o Turbo Boost)	2.100 GHz
Cache	2x 32 KiB L1, 256 KiB L2 per core 24 MiB L3 per processor	2x 64 KiB L1, 512 KiB L2 per core 2x 6 MiB L3 per processor
Interconnect	6.4 GT/s QPI (25.6 GB/s)	6.4 GT/s HT 3.0 (25.6 GB/s)
Memory configuration	256 GiB DDR3-1066 4x SMI per socket	64 GiB DDR3-1333 4x DDR3 per socket
OS	Red Hat EL6 2.6.32-71.el6.x86_64	Ubuntu 10.10 2.6.35-22-server
Compiler	gcc 4.4.4, icc 11.1 (20091130)	gcc 4.4.5, icc 11.1 (20091130)

to connect to the chipset, three links can be used to connect to other processors. The 12-core AMD Opteron 6100 series processors are multi-chip-modules (MCM). They consist of two six-core dies that are internally connected via HyperTransport (HT) links. Each die includes 6 MiB shared L3 cache and a dual-channel DDR3 controller for a total of 12 MiB L3 and four DDR3 channels per processor. Each socket supports four 16-Bit HT 3.0 links, one for communication with the chipset and three to connect to other sockets.

The topologies of the test systems are depicted in Figure 1. The four sockets in the Intel system are fully connected via QPI links. Each link provides 25.6 GB/s of raw bandwidth (12.8 GB/s per direction). Data is transferred in 64 Byte packages each with an 8 Byte header [5, 10]. This protocol overhead limits the achievable bandwidth to 11.37 GB/s per direction. The AMD system has four sockets as well, however it consists of eight NUMA nodes. The three 16-Bit links per socket that connect the processors are actually used as six 8-Bit links that provide 12.8 GB/s (6.4 GB/s per direction) each. The dies within a MCM are connected via one 16-Bit and one 8-Bit link [2]. Every die is directly connected



(a) four socket Xeon 7500 (b) four socket Opteron 6100

Fig. 1: System topology comparison

to three dies in other sockets. There are two sets of four fully connected nodes: {0,2,4,6} and {1,3,5,7}. Data transfers between those groups require two HT hops if the nodes are not in the same MCM.

Both systems use snooping based protocols to maintain cache coherence. The additional messages reduce the effective bandwidth. AMD implements a probe filter, called HT Assist, that uses a portion of the L3 cache as directory in order to filter unnecessary messages [2]. However, that reduces the L3 capacity and the L3 bandwidth has to be shared between data accesses and probe filter accesses. Furthermore, only 16 MiB of a node's memory can be cached somewhere in the system as the directory size is limited.

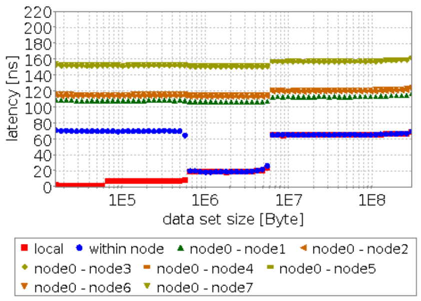
4 Benchmarks

4.1 Microbenchmarks

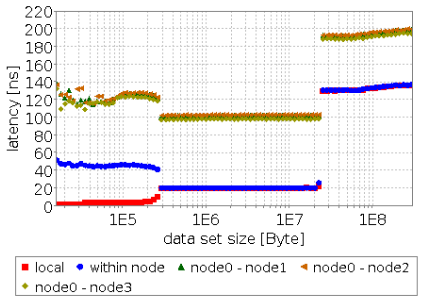
We use a set of open source microbenchmarks [6, 4] to perform a low-level analysis of each system's memory performance. Highly optimized assembler routines and time stamp counter based timers enable precise performance measurements of data accesses in cache coherent NUMA systems with a 64 Bit x86 processors. The benchmarks are parallelized using pthreads and individual threads are pinned to single cores using `sched_setaffinity()`. Coordinated data access sequences are performed in consideration of the coherence protocol to transfer data into a selected cache in a well-defined coherence state. Latencies and bandwidths of accesses to any core's caches and any socket's memory can be measured as well as the aggregated bandwidth of shared caches and memory controllers. The benchmarks support different data allocation schemes: `localalloc` results in all threads using memory local to their NUMA node while `globalalloc` forces all memory to be allocated at the first NUMA node.

4.2 SPEC OMPM2001

To show the impact of the different hardware characteristic of the two platforms on application performance we use the SPEC OMP Benchmark suite V3.2. The 11 codes from SPEC OMPM2001 cover a wide range of applications and use different OpenMP constructs for the parallelization. The philosophy to use real applications results in relatively complex performance properties. More details about the benchmark and its performance properties can be found in [8, 1, 7, 3]. We use the Intel Compiler Suite in version 11.1 on both systems to create the benchmark executables. The same optimization flags are used for all benchmarks, namely `-O3 -ipo -openmp`. To consider the different SIMD extensions we additionally use `-msse3` on the AMD and `-xSSE4.2` on the Intel system. `Likwid-pin` [9] is used to avoid thread migration.



(a) Opteron 6172



(b) Xeon X7560

Fig. 2: Access latencies: Thread on core 0 accesses local or other core's memory.

5 Results

5.1 Memory Latency

The microbenchmark results of cache and memory latencies are depicted in Figure 2 and summarized in Table 2.

Figure 2a depicts the latencies in the quad-socket Opteron 6172 system. Accesses to local caches require 1.4 ns (3 cycles) for the L1, 7.1 ns (15 cycles) for the L2, and 19 ns (40 cycles) for the L3 cache. Reading data from the L1 or L2 cache of another core on the same die requires 70 ns. This is significantly higher than measured on older Opteron models without HT Assist [4] as probing other cores is delayed until the HT Assist directory has been checked. With 109 ns the latency between the two dies within a MCM is only marginally lower than the 113 ns for accessing directly connected dies in other sockets. A second HT hop adds another 40 ns to the latency. Memory requests have a latency of 65 ns for local memory, 113-119 ns for memory that can be reached with one HT hop, and 159 ns if two hops are necessary. Without HT Assist, responses from memory would be delayed until the farthest nodes reply to the probe message. This would require approximately 133 ns (152 ns for 2-hop L3 accesses includes HT Assist lookup of about 19 ns). Thus, the HT Assist reduces the local memory latency by more than 50%.

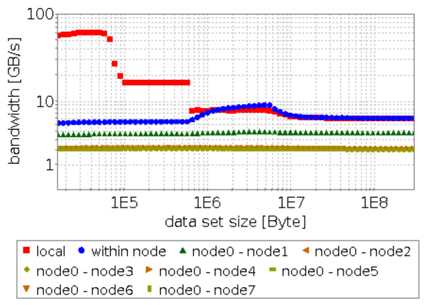
Table 2: Access latencies in ns, cache lines in state modified

Processor	Opteron 6172					Xeon X7560		
Source	local	within socket		other socket		local	within socket	other socket
		on-die	2nd die	1 hop	2 hops			
L1	1.4	70.4	109.5	113.3	153.3	1.8	48.2	126.0
L2	7.1	70.4	109.5	113.3	153.3	4.4	46.4	128.2
L3	19.0	19.0	107.6	111.9	152.4	20.3	20.3	103.0
RAM	65.7	65.7	114.3	119.0	159.0	130.4	130.4	192.8

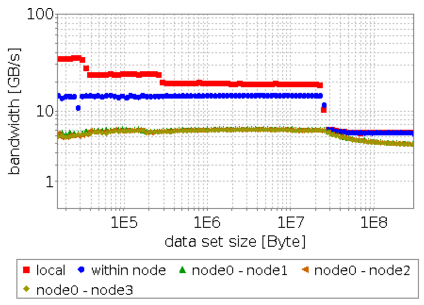
Figure 2b shows the latencies on the quad-socket Xeon X7560 system. With 1.8 ns (4 cycles), the L1 cache is slightly slower than the AMD implementation. The L2 is faster (4.4 ns; 10 cycles) and the L3 has almost the same latency (around 20 ns). With around 47 ns on-die transfers from other L1 and L2 caches are notably faster. As can be expected from a fully connected system, the latencies for all other sockets are almost identical. However, the single QPI hop adds about 62 ns to the latency compared to 40 ns per HT hop. Memory latencies are 130 ns for the local memory and 192 ns for memory at other sockets which is much more than on the AMD system. It is also significantly more than the 103 ns latency of remote L3 cache accesses that provide an estimate for the snoop responses. The high latency is therefore not a problem of the coherence mechanism but can be attributed to the SMBs that translate from the processor's SMI interface to DDR3.

5.2 Memory Bandwidth

Figure 3 depicts the read bandwidths that we measured for a single thread. A summary that also includes write bandwidths can be found in Table 3. While the Opteron's L1 caches have two 128-Bit read ports (32 Byte per cycle), the Xeon's L1 has only one 128-Bit read port. The L1 write bandwidth is 16 Byte per cycle on both systems. On the Intel system, the L2 and L3 caches provide more bandwidth. Data exchanges between cores on a single die achieve higher bandwidths as well. The single threaded bandwidth from/to main memory is relatively low on both systems. Remote accesses again show the more complex topology of the AMD system. Data can only be read with about 3.8 GB/s from caches or memory of the processor's second die. The bandwidth drops further down to 2.1 GB/s for reading from other sockets. On the Intel system, data can be read with 6.3 GB/s from caches in other processors and 3.9 GB/s from remote memory. These transfer rates are limited by the interconnect as well as by the outstanding requests supported by a single core.



(a) Opteron 6172



(b) Xeon X7560

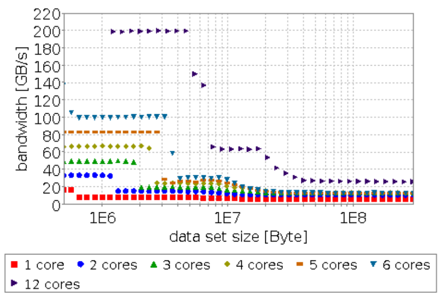
Fig. 3: Memory read bandwidth: Thread on core 0 accesses other core's memory

Table 3: Single thread read (write) bandwidth in GB/s. Write bandwidth listed only for local caches and memory as it cannot be written into other cores' caches.

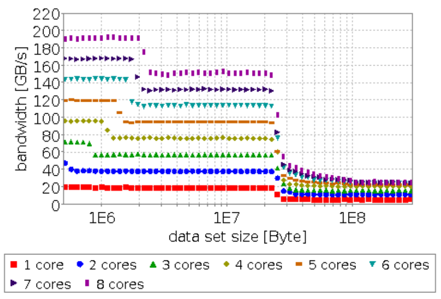
Processor	Opteron 6172					Xeon X7560		
Source	local	within socket		other socket		local	within socket	other socket
		on-die	2nd die	1 hop	2 hops			
L1	62.9 (31.8)	5.4	3.7	2.1	2.1	35.2 (35.2)	14.3	6.3
L2	16.7 (10.0)	5.4	3.7	2.1	2.1	24.1 (22.3)	14.3	6.3
L3	7.8 (7.0)	9.0	3.9	2.1	2.1	19.2 (12.7)	14.3	6.3
RAM	6.1 (4.4)	6.1 (4.4)	3.8 (3.2)	2.1 (2.0)	2.1 (2.0)	5.5 (4.9)	5.5 (4.9)	3.9 (3.5)

By using eight concurrent threads , 11.0 GB/s can be transferred over one QPI link. This is very close to the expected peak performance of 11.37 GB/s (see Section 3). Six cores on one die of the Opteron 6172 processor can read data with 5.3 GB/s from the second die in the MCM which is much lower than expected. However, no more than 2.1 GB/s can be read from dies in other sockets via the 8-Bit links even with multiple cores reading in parallel. Thus, the bandwidth limit is not caused by too few outstanding requests per core but by the width of the HT links. The low performance indicates that only one link is used to exchange data between two dies in different sockets, even though two links are available between the sockets.

The scaling behavior of the L3 and main memory read bandwidth is depicted in Figure 4 for single NUMA nodes. The results as well as the corresponding write bandwidths are summarized in Table 4. On the AMD system the six cores of one die share an aggregate L3 bandwidth of 30.9 GB/s which is identical for reading and writing. Each dual-channel memory controller delivers a read and write bandwidth of 13.2 GB/s and 7.1 GB/s, respectively. On the Intel system, L3 read and write bandwidths scale linearly with the number of cores and reach a total of 152 and 101 GB/s, respectively. Thus, the Xeon's shared



(a) Opteron 6172



(b) Xeon X7560

Fig. 4: Memory read bandwidth scaling for concurrent accesses of multiple cores

Table 4: Aggregate read (write) bandwidth per socket in GB/s.

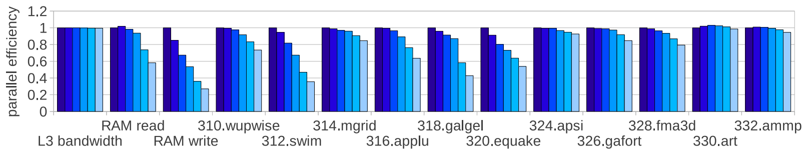
Source		1 core	2 cores	4 cores	6 cores	8 cores	12 cores
Opteron	L3	7.8 (7.1)	15.2 (14.1)	24.1 (24.5)	30.8 (30.9)	n/a	63.7 (61.8)
6172	RAM	6.1 (5.1)	10.7 (6.6)	13.2 (7.1)	13.2 (7.1)	n/a	26.2 (14.0)
Xeon	L3	19.2 (12.7)	38.3 (25.4)	76.6 (50.8)	114 (76.0)	152 (101)	n/a
X7560	RAM	5.5 (4.9)	11.4 (8.5)	20.6 (10.8)	24.5 (10.9)	25.7 (10.9)	n/a

L3 cache provides significantly more bandwidth than both of the Opteron's L3 partitions combined while being larger and shared by all cores. The memory read bandwidth per socket is almost identical on the two systems (AMD: 26.2 GB/s, Intel: 25,7 GB/s) while the AMD system provides higher write bandwidth (AMD: 14.0 GB/s, Intel: 10,9 GB/s).

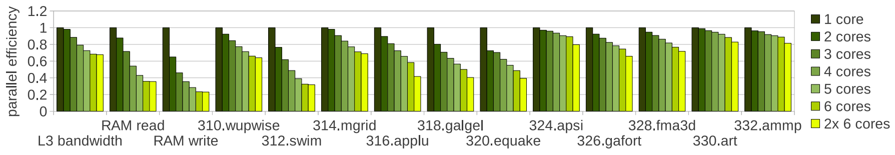
The L3 and memory bandwidths scale linearly with the number of sockets on both systems if memory is allocated by each thread (localalloc). However, if all memory is allocated and initialized by the master thread (globalalloc), memory bandwidth is reduced significantly. In this case performance is limited as only the memory controllers of the first NUMA node are used. The bandwidth limitations of the processor interconnects affect the performance as well. The Intel system provides 17.7 GB/s memory bandwidth if all four sockets use memory from node0. On the AMD system, the single socket bandwidth is already reduced to 7.5 GB/s as two of the processor's four memory channels remain unused and half the cores access memory of the second die via the intra-socket HT connection. If all sockets are used, the 8-Bit links that connect the individual dies become the limiting factor resulting in an even lower bandwidths of 6.6 GB/s for this worst-case scenario. Furthermore, the limited coverage of the HT Assist directory causes invalidations of cache lines if more than 16 MiB of node0's memory are cached somewhere in the system. The 48 L2 caches (24 MiB) already exceed that capacity, thus the effective L3 size is reduced to zero. While both systems require NUMA optimized memory allocation to achieve optimal memory performance, the low-level benchmarks strongly indicate that the AMD memory subsystem is much more sensitive to non-optimal memory allocation.

5.3 SPEC OMPM2001 Scaling with Multiple Cores

For our OpenMP scalability analysis we first examine the parallel efficiency of SPEC OMPM2001 applications running on a single socket of our quad-socket test systems (see Figure 5). Also depicted in Figure 5 are the L3 bandwidth as well as main memory read and write bandwidth. On the Intel system, the L3 cache design is remarkably powerful, allowing the L3 bandwidth to scale linearly. The L3 cache of the AMD processor does not scale linearly, neither does the main memory bandwidth of both systems. Especially the memory write bandwidth scales poorly with a growing number of cores. The limited scaling of shared resources (see Table 4) has a strong influence on the benchmark results, which results in significantly different scaling behavior on multi-core processors than



(a) Xeon X7560



(b) Opteron 6172

Fig. 5: SPEC OMPM2001 scaling with multiple cores of a single socket. A parallel efficiency of 1 indicates linear speedup.

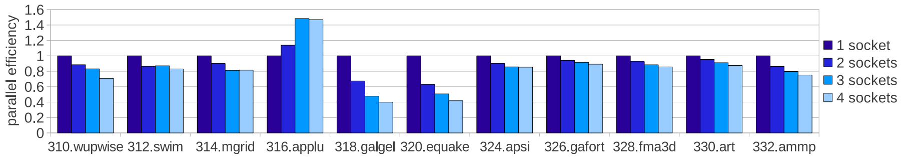
reported in [3] for single-core multi-socket systems. Based on the results shown in Figure 5 the benchmarks can be divided into three groups:

- *Group 1* includes mainly compute bound benchmarks. 324.apsi, 330.art, and 332.ammp show marginal memory influence on the Intel system as well as a minor L3 cache dependence.
- *Group 2* consists of 310.wupwise, 314.mgrid, 326.gafort, and 328.fma3d that are subject to a higher influence of L3 cache and main memory bandwidths.
- *Group 3* contains the strongly memory bound benchmarks 312.swim, 316.applu, 318.galgel, and 320.equake.

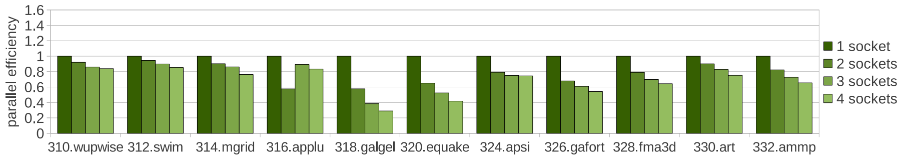
Figure 5b also shows the scalability effects on the Opteron 6172 processor when both dies (2x 6 cores) are being used. 310.wupwise, 312.swim, and 314.mgrid scale almost linearly. They are not affected by the limited interconnect bandwidths and increased access latencies. The remaining benchmarks show significantly reduced parallel efficiency if both NUMA nodes of a single socket are being used.

5.4 SPEC OMPM2001 Scaling with Multiple Sockets or NUMA Nodes

Application scaling on multiple sockets is additionally influenced by the inter-socket connectivity via HT or QPI links. Higher latencies and lower bandwidths compared to the intra-socket case worsen multi-socket scalability. In addition to the coherence traffic, the links are stressed by accesses to distant memory due to NUMA unaware memory allocation. These considerations need to be taken into account for the evaluation of all performance results depicted in Figure 6. Unfortunately, the grouping with respect to memory boundedness does not apply here as the interconnect influence is different for the individual benchmarks. We



(a) Xeon X7560 (8 cores per socket)



(b) Opteron 6172 (12 cores per socket)

Fig. 6: SPEC OMPM2001 scaling with multiple sockets. A parallel efficiency of 1 indicates linear speedup.

observe good multi-socket scaling of 310.wupwise, 312.swim, and 314.mgrid on both systems. 316.applu shows strong super-linear speed-up on the Intel system in accordance to the findings of [8] and [3]. The data set of this benchmark apparently exceeds the available cache size of two AMD sockets and fits well into the caches of three or more sockets. However, the cache effect is alleviated by other effects that hinder multi-socket scalability. This behavior likely results from non-local memory usage, as e.g. the array initialization in `ssor.f:49` ff can provoke non-local memory accesses. The parallelized loop iterates over j from jst to $jend$ to initialize eight arrays that are later used for calculations in disregard of the locality information (e.g. in subroutine `blts`). 318.galgel and 320.equake scale poorly on both systems in accordance to [3]. A non-optimal memory allocation is likely a contributor to that behavior as single-socket scaling appears to be similar to memory bound benchmarks such as 312.swim. The remaining benchmarks scale significantly better on the Intel system, most likely due to the higher intra-socket bandwidths. This is particularly noticeable for 326.gafort, that accesses random indices of a computation matrix in one of the main loops (`shuffle-do#10`). This access pattern as well as the surrounding OMP locks lead to high inter-socket traffic.

5.5 SPEC OMPM2001 Performance Comparison

For a comparison of the overall performance of our test systems, it is important to note that the AMD system has an approx. 40% higher peak computing performance. This potentially benefits applications that are computationally bound. On the other hand, the Intel Xeon processor strongly benefits from its large L3 cache as well as the high L3 bandwidth, potentially improving the performance of memory bound applications. Figure 7 depicts the relative performance (AMD/Intel) of the two test systems. Note that the 'half socket' main mem-

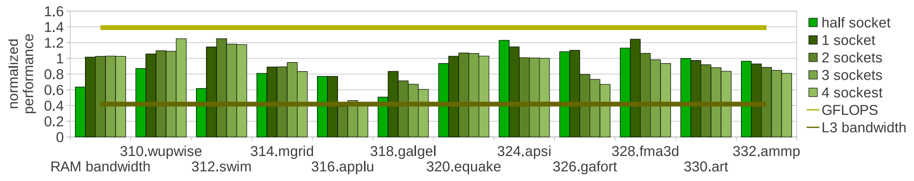


Fig. 7: SPEC OMPM2001 application performance of Opteron 6172 relative to Xeon X7560. Bars > 1 indicate a performance advantage for the Opteron.

ory bandwidth is much lower on AMD compared to Intel, as only two of the Opteron's four memory channels are used in this case. 312.swim and 318.galgel are particularly sensitive to this. The Intel system generally scales better over multiple sockets as illustrated by the decreasing relative performance of 318.galgel, 326.gafort, 328.fma3d, 330.art, and 332.ammp. One outlier is 316.applu, which shows super-linear speedup as stated in Section 5.4 and should therefore not be analyzed in detail at this point. As pointed out in Section 5.3, 310.wupwise, 312.swim, and 314.mgrid are not sensitive to inter-socket communication performance. The scaling behavior of these benchmarks is therefore competitive on the AMD system, with 310.wupwise and 312.swim showing superior overall performance. On average the scaling is worse on the Opteron system, despite AMD's HT Assist feature. This indicates that the unexpectedly low HyperTransport bandwidths (see Table 3) are significantly limiting the performance for benchmarks that perform remote cache or memory accesses. Therefore, the total SPEC OMPM2001 result is significantly higher on the Intel system while single-socket performance is almost identical.

6 Conclusions

Today's shared memory x86 systems are complex cache coherent NUMA architectures. Multiple processors are connected via point-to-point interconnects and each processor features multiple cores. This results in different performance levels for data transfers depending on the exact location of the data source. Sophisticated low-level microbenchmarks that are tailored to investigate the performance of data transfers within shared memory systems are used to present precise results for latencies and bandwidth for on-die and off-die data transfers in two state-of-the-art quad-socket x86_64 systems. Considerable limitations of the inter-socket communication bandwidths are identified. We also provide detailed data for the scaling behavior of shared resources such as last level caches and main memory.

In our scalability analysis of SPEC OMPM2001 applications, both test systems show severe deficiencies in terms of overall parallel efficiency. Scaling is limited by shared resources within each NUMA node that do not scale linearly with the number of cores as well as non-linear scaling with the number of NUMA nodes. This shows that the bottlenecks identified by the microbenchmarks also limit the performance and scalability of real applications. For the majority of applications, the Intel test system performs better regarding both aspects. While the superior scaling for one processor is expected due to the single-die solution and the extremely well-performing L3 cache, the lower performance of the AMD system in multi-socket configurations is surprising. With respect to the tested workloads, the HT Assist feature does not prove to be sufficient to compensate for the disadvantages that come with the more complex topology and the comparatively low HyperTransport bandwidth.

Acknowledgment The authors would like to thank Intel Germany for providing us with the Intel Nehalem-EX test system. We also thank NEC Deutschland GmbH for granting us access to the AMD Magny Cours test system.

References

1. Vishal Aslot and Rudolf Eigenmann. Quantitative performance analysis of the SPEC OMPM2001 benchmarks. Sci. Program., 11:105–124, April 2003.
2. Pat Conway, Nathan Kalyanasundharam, Gregg Donley, Kevin Lepak, and Bill Hughes. Cache hierarchy and memory subsystem of the AMD Opteron processor. IEEE Micro, 30:16–29, March 2010.
3. Karl Fürlinger, Michael Gerndt, and Jack Dongarra. Scalability analysis of the SPEC OpenMP benchmarks on large-scale shared memory multiprocessors. In Yong Shi, Geert van Albada, Jack Dongarra, and Peter Sloot, editors, Computational Science – ICCS 2007, volume 4488 of Lecture Notes in Computer Science, pages 815–822. Springer Berlin / Heidelberg, 2007.
4. Daniel Hackenberg, Daniel Molka, and Wolfgang E. Nagel. Comparing cache architectures and coherency protocols on x86-64 multicore SMP systems. In MICRO 42: Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture, pages 413–422, New York, NY, USA, 2009. ACM.
5. Intel. An Introduction to the Intel QuickPath Interconnect, January 2009.
6. Daniel Molka, Daniel Hackenberg, Robert Schöne, and Matthias S. Müller. Memory performance and cache coherency effects on an Intel Nehalem multiprocessor system. In PACT '09: Proceedings of the 2009 18th International Conference on Parallel Architectures and Compilation Techniques, pages 261–270, Washington, DC, USA, 2009. IEEE Computer Society.
7. Matthias S. Muller, Kumaran Kalyanasundaram, Greg Gaertner, Wesley Jones, Rudolf Eigenmann, Ron Lieberman, Matthijs Van Waveren, and Brian Whitney. SPEC HPG benchmarks for high performance systems. Int. J. High Perform. Comput. Netw., 1:162–170, December 2004.
8. Hideki Saito, Greg Gaertner, Wesley Jones, Rudolf Eigenmann, Hidetoshi Iwashita, Ron Lieberman, Matthijs van Waveren, and Brian Whitney. Large system performance of SPEC OMP benchmark suites. International Journal of Parallel Programming, 31:197–209, 2003.
9. Jan Treibig, Georg Hager, and Gerhard Wellein. Likwid: A lightweight performance-oriented tool suite for x86 multicore environments. Parallel Processing Workshops, International Conference on, 0:207–216, 2010.
10. D. Ziakas, A. Baum, R.A. Maddox, and R.J. Safranek. Intel®quickpath interconnect architectural features supporting scalable system architectures. In High Performance Interconnects (HOTI), 2010 IEEE 18th Annual Symposium on, pages 1 –6, 2010.