LS-DYNA® Performance on 64-Bit Intel® Xeon® Processor-Based Clusters

Tim Prince, PhD ME
Hisaki Ohara, MS IS
Nick Meng, MS EE
Intel® Software and Solutions Group

Abstract

Benchmark performance of a cluster based on the newly introduced 64-Bit Intel® Xeon® Procesessor-based clusters is presented. Car2car and 3cars benchmarks from topcrunch.org are evaluated, using OpenIB based interconnects. Effect of shared memory options under Intel® MPI is evaluated. Improvement from optimizing 3cars P-file decomposition is demonstrated. Intel® Cluster Tools profiler is discussed.

Summary

Performance of the 3cars and new car2cars benchmarks from topcrunch.org is evaluated on a cluster of Intel Xeon dual-core servers, running Intel® MPI and OpenIB. Elapsed computation time is shown to be affected significantly by choice of message passing communication device and by optimizing LS-DYNA decomp parameters. LS971, still in beta test at the time of writing, shows significantly improved performance.

Cluster Specification

Cluster performance is evaluated running Red Hat EL4* update 2 Linux* x86-64 operating system. CPUs on the pre-production units tested run at 3.2 GHz, vs. 3.73 GHz announced as supported at product release. Each node has 2 dual-core packages, a total of 4 cores, with a dual Front Side Bus, giving a separate memory bus path for each package. Memory installed consists of 8GB RAM per node. LS-DYNA release is ls970.6763 for Intel® MPI V.2 running on Intel® EM64T. Testing was performed with 4 nodes (total 16 cores, 16 MPI processes) and 8 nodes (32 cores, 32 MPI processes).

CPU: Intel® Xeon® Processor, 3.2GHz Dual Core (each core has 2MB L2 cache)
Memory: 8GB FB-DIMM (667MHz) 1Gx8
InfiniBand: Mellanox* HCA (MHEL-CF128-T)
    PCI-Express x8
OS: RHEL4 U2 for EM64T
OpenIB
    kernel: 2.6.9.OpenIB.4507.EL.rootsmp (backported kernel for RHEL4)
    Usermode: openib-usermode-4507-1

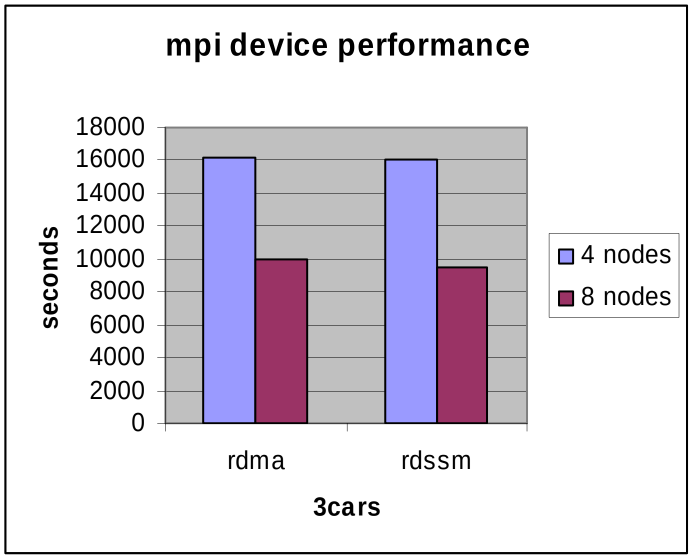
MPP-DYNA Command Line Parameters

We stumbled over the command line parameters for the car2car benchmark. Experience has shown that it may be necessary to use memory=700M memory2=400M when running 2 processes, or memory=600M memory2=300M for 4 processes, and memory2=120M for 16 processes. Perplexingly poor results with shared memory communication at 16 processes were resolved by adjusting the memory2 parameter.

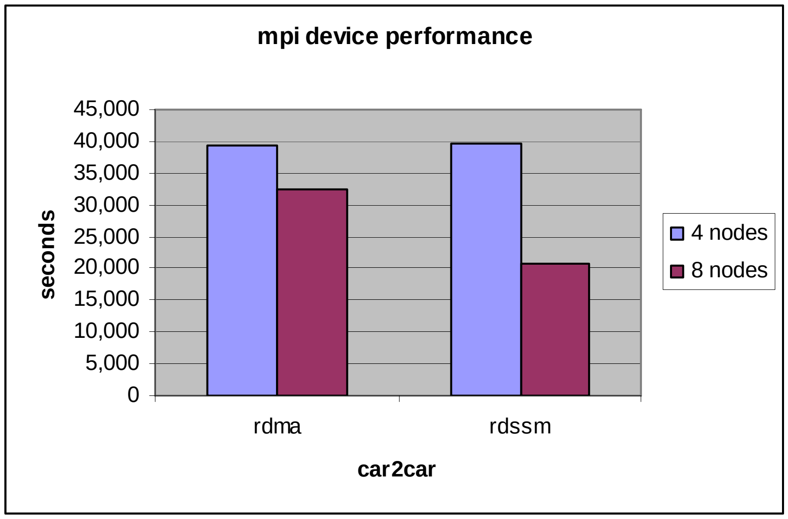
Communication Device

Intel MPI supports changing communication device by setting environment variable or mpiexec command line parameter, with the same LS-DYNA binary, without reboot. We tested communication by OpenIB Infiniband alone (rdma device), and by combining Infiniband communication between nodes with shared memory communication within the node (rdssm device). As might be expected, best performance on the 8-node cluster was achieved with the combined shared memory and OpenIB communication.

Somewhat unexpectedly, the car2car benchmark on the 4-node cluster performed poorly at first, with shared memory. The problem was resolved by adjusting the memory2 command line parameter. Then, a performance increase is provided by adding shared memory communication, increasing with number of nodes:



The new caravan benchmark got a slight decrease in performance on 4 nodes with combine OpenIB and shared memory communication, while it scaled well to 8 nodes only with the combined rdssm device:

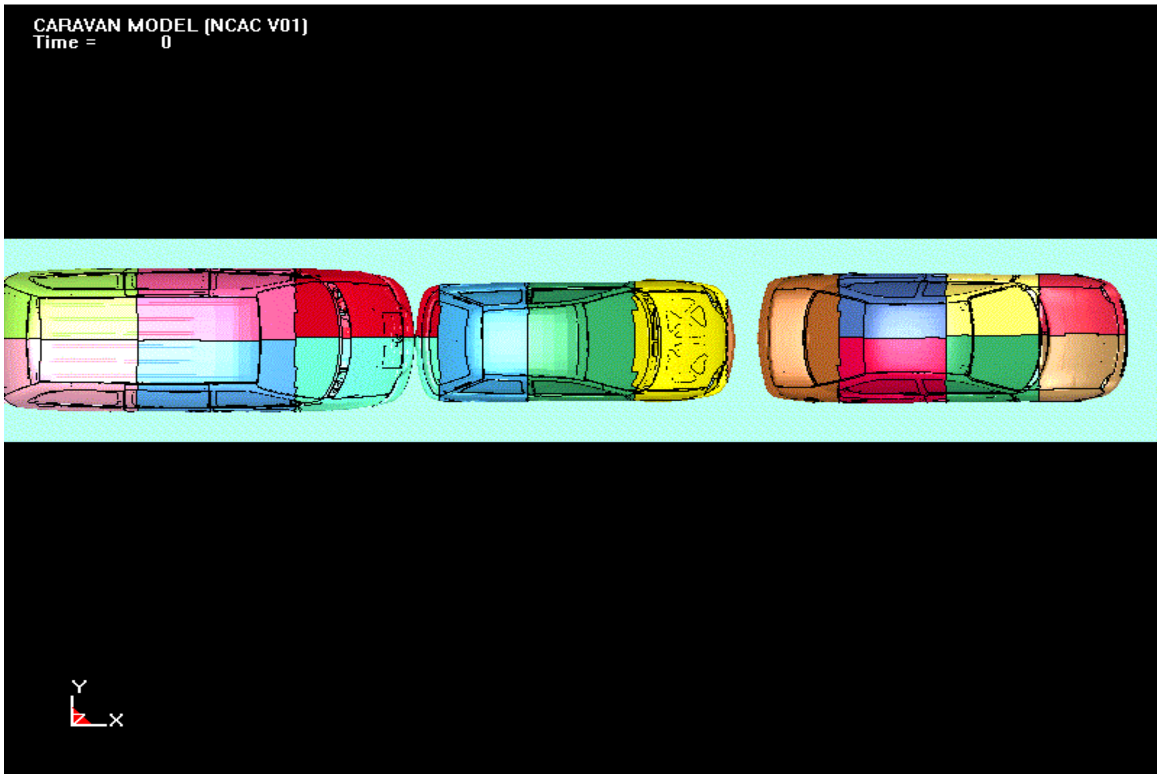


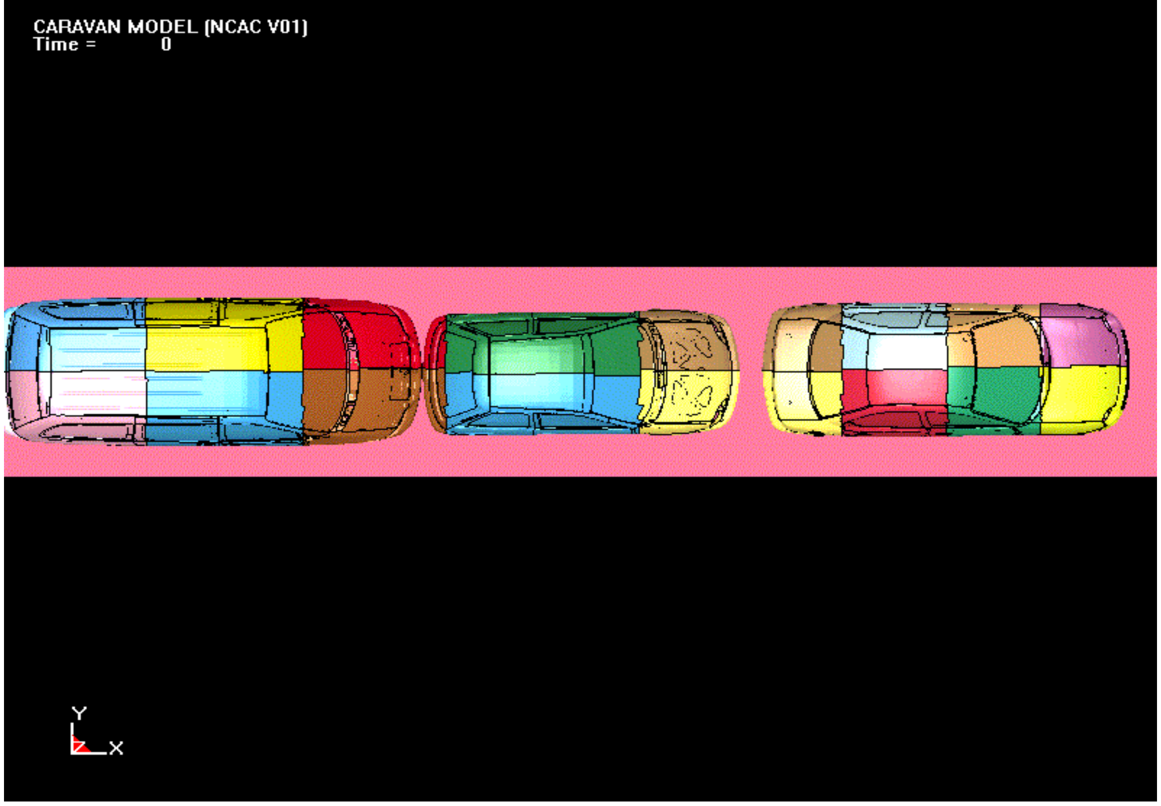
Shared Memory Communication Affects Cache Behavior

Single-node profiling has shown that shared memory communication can produce a favorable effect on cache performance. Both gprof and Intel VTune™ profilers were used to confirm a significant reduction in cache misses on the 3cars benchmark, between no shared memory and shared memory communication. So, it appears that performance can be enhanced by keeping arriving messages in cache. This is in addition to the possible benefit of keeping local communication off the interconnect. Past CPU models have not had adequate cache to perform well with shared memory communication. Improved cache in newer models translates to better performance of shared memory communicators.

Performance Tuning by pfile decomp

Some time after topcrunch posting of 3cars benchmark results began, a rule was made that pfile decomp parameters should not be changed from those supplied on topcrunch. However, it is not generally possible to get effective scaling of 3cars performance to 32 or more processes, except by optimizing the decomp parameters. For 16 processes running on 4 nodes, we found elapsed time reduced from 16006 with topcrunch decomp, to 14346 with decomp { silist 6 sy 2}.



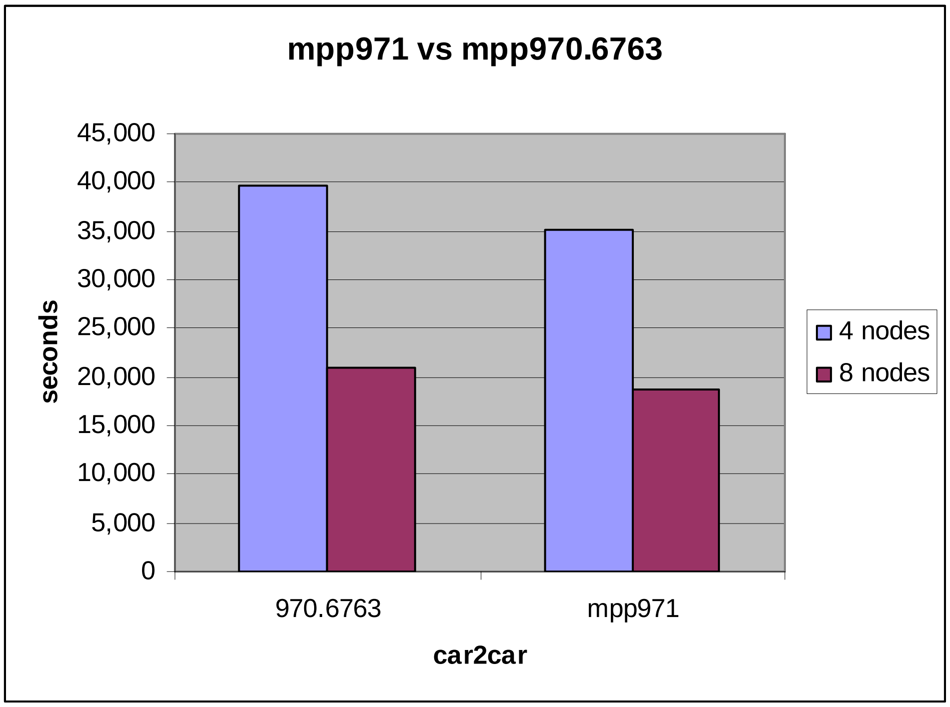


The lower picture shows the more effective 3cars decomp.

The author of the car2car benchmark made an effort to optimize decomposition for performance. The results were not expected to depend strongly on decomp variations, and the rule was stated clearly that such variations should not be undertaken.

LS971

LSTC has included additional optimization in LS971, to improve performance of current and future platforms. In addition, there is a possibility of upgrading to currently supported compilers, taking advantage of improved numerical stability and performance. We show a significant performance improvement for mpp971.6665 rdssm, built with Intel Fortran 9.1:



MPI Performance Profiling

The Intel® Cluster Tools component, the Intel® Trace Analyzer and Collector, can be used with a standard release of LS-DYNA MPP, by setting up shared libraries so as to substitute a profiling library. An Intel MPI or MPICH Intel platform cluster installation could obtain data to verify proper performance of a cluster, possibly for software support personnel to use at a remote location.

A potential use might be in optimizing communication by best distribution of processes to nodes. For example, certain groups of processes might communicate best by shared memory, and others by interconnect. We have evidence of asymmetries in LS-DYNA communication that might justify such an effort, but no practical conclusions. As demand for larger clusters increases, such analysis may be useful.

As a practical demonstration, we show improvement in performance of LS-DYNA collectives from mpp970.5434a Intel MPI 1 (private build with ifort 9.0) to mpp970.6763 iMPI 2 (LSTC released build with ifort 8.1) by this profiler. The time spent in MPI_Allreduce is cut in half, in spite of an increased number of calls, so the total time consumed by Intel MPI is reduced by 20 percent. The profile is for 1 ms of car2car simulation on 4 nodes (16 cores total).

Caravan	5434a_90-10		6763_81-20	
	Tself	#Calls	Tself	#Calls
Application	16,529		21,184	
MPI	3,711	5,185,369	3,026	4,648,365
MPI_Allreduce	2,077	388,138	949	444,284
MPI_Recv	1,389	899,134	1,674	682,553
MPI_Bcast	167	40,376		
MPI_Wait			213	1,368,132

Conclusion

We have demonstrated good performance on an OpenIB Intel MPI cluster made up of 8 nodes, including pairs of Dual-Core Intel Xeon processors - 32 cores total. Further improvements in performance may be expected from LS971. Profiling tools are available to assist customer installations, where there is a concern about computational performance.