Partitioning Effects on MPI LS-DYNA Performance

Jeffrey G. Zais
IBM
1308 Third Street
Hudson, WI 54016-1225
zais@us.ibm.com

Abbreviations:
MPI – message-passing interface
RISC - reduced instruction set computing
SMP - shared memory parallel
MPP – massively parallel processing

Keywords:
Crash Simulation, LS-DYNA, Metal Stamping, Parallel Computing, Performance, Workstation

ABSTRACT

The MPI version of LS-DYNA includes several options for decomposition of the finite element model. In this paper, the use of these options will be explored, for both metalforming and automotive crash simulations for input decks with size ranging from small to very large. The effect on elapsed time performance and scalability will be measured for different partitioning options. In addition, performance characteristics of a workstation cluster will be evaluated.

INTRODUCTION

The MPI version of LS-DYNA has been developed by LSTC throughout the 1990's . A primary goal for the code has been superior scalability, enabling many processors to work together efficiently in the execution of LS-DYNA.

Some important aspects which affect performance of the MPI version are processor speed, model characteristics (size and type of elements), communication, and load balance. Processor performance has a very direct impact on MPI LS-DYNA elapsed time, but for the purposes of this study, the type of processors will be considered fixed, so the resulting variations in performance will not be considered.

Conversely, communication and load balance can be somewhat controlled by the end user. During the initialization phase of the MPI code, the model is partitioned into several domains, and the arrangement of these domains will cause changes in how much communication is required between the domains, and will also influence the load imbalance between the domains. This study demonstrates, for several representative input decks, how various partitioning options affect overall performance.

SMP PARALLELISM IN LS-DYNA

The first LS-DYNA multi-processor version was an SMP (shared memory parallel) implementation available on the CRAY Y-MP in 1989. Subsequently, SMP Parallel LS-DYNA has been ported to many computer platforms and architectures, ranging from high-end vector supercomputers to RISC-based servers to machines based on low-cost commodity-based microprocessors. For all of these computers, SMP parallelism is achieved in basically the same fashion. Compilers use directives inserted into the LS-DYNA source code to generate machine instructions which divide the work for key loops among the available processors specified by the user. Only loops which are "safe" - those that will produce correct results when run in parallel - have these parallel constructs.

In SMP parallelism, there are several barriers which cause less than ideal scaling. Chief among them is the fact that only a certain number of the loops in the code are able to run in parallel. Even for those loops that are parallel, there is startup time where the work is divided among the various processors, adding to system overhead. Finally, even though the work is scheduled as evenly as possible, load imbalance among the processors means that some processors will complete their portion of the work in a loop and must wait for the others, before the machine can advance to the next section of the LS-DYNA code.

Because of all these factors, SMP parallel performance is limited. While experts from the LSTC development team and the various computer hardware vendors continually work on

improving SMP parallel performance, parallel scalability beyond 4 or 6 processors is not very effective, and is not used very often in a production environment.

MPI PARALLELISM IN LS-DYNA

Characteristics of MPI LS-DYNA

The limits on SMP parallelism, not just in LS-DYNA, but in many codes, have led to the development of domain decomposition parallel methods. The chief advantage is that with domain decomposition methods, a far greater percentage of the instructions can be run in parallel, so the parallel efficiency is greater.

In the domain decomposition version of LS-DYNA, the geometry is divided into several domains, one for each processor. During every time step, each processor advances the solution for its own domain to the end of that time step. This process is independent of all other domains, so it is highly parallel. However, before work on the next time step can begin, communication must occur to relate information on the state of the solution to neighboring domains. Once this communication is complete, the solution phase of the next time step begins.

The communication in LS-DYNA takes place according to MPI (the message-passing interface), a standard communication protocol. This is a portable set of communication calls available on all popular computer systems. Therefore, the code is referred to either the MPI version of LS-DYNA, or the domain decomposition version of LS-DYNA. Another common name is "MPP-DYNA." The MPP moniker was originally associated with the "massively parallel processing" machines of the mid-1990's, but as parallel computer architectures evolved, "massively parallel" became something of a misnomer. Today MPP-DYNA more accurately refers to the "message-passing parallel" version of the code, as opposed to the "shared memory parallel" version of LS-DYNA.

The MPP machines were envisioned with thousands of processors applied to execution of the same binary. Parallel speedup of computers is limited by Amdahl's Law. Figure 1 shows how a code can scale as a function of percentage of time spent in parallel routines. In order to use hundreds (or thousands) of processors a code must be more than 99.9% parallel. For a full-featured industrial code like LS-DYNA, this is very difficult. Fortunately, LSTC has been able to make the MPI version contain enough parallel content so that in cases it is more than 98% parallel and can scale efficiently up to 64 or more processors. Since microprocessors have evolved to be very powerful in recent years, this level of scalability coupled with fast processors allows the user to solve large LS-DYNA simulations in a reasonable elapsed times.

It is apparent that a well-selected set of domains could influence MPI LS-DYNA performance. Load balance between the domains is achieved by insuring that each domain has an equal amount of work required at each time step. This is, of course, more complicated than just dividing the total number of elements in the model into groups with the same number of elements. The computational cost of different types of elements and materials is one factor which complicates the load balance. In addition, time spent in contact also has a great influence on computational cost. The time spend in contact will also change during the solution, so that a domain which initially produces good load balance may end up with much worse load balance characteristics.

The relative cost for communication is also important. If the domains are established incorrectly, great amounts of communication could be required between domains.

Because of these reasons, selection of the proper domain decomposition does influence performance.

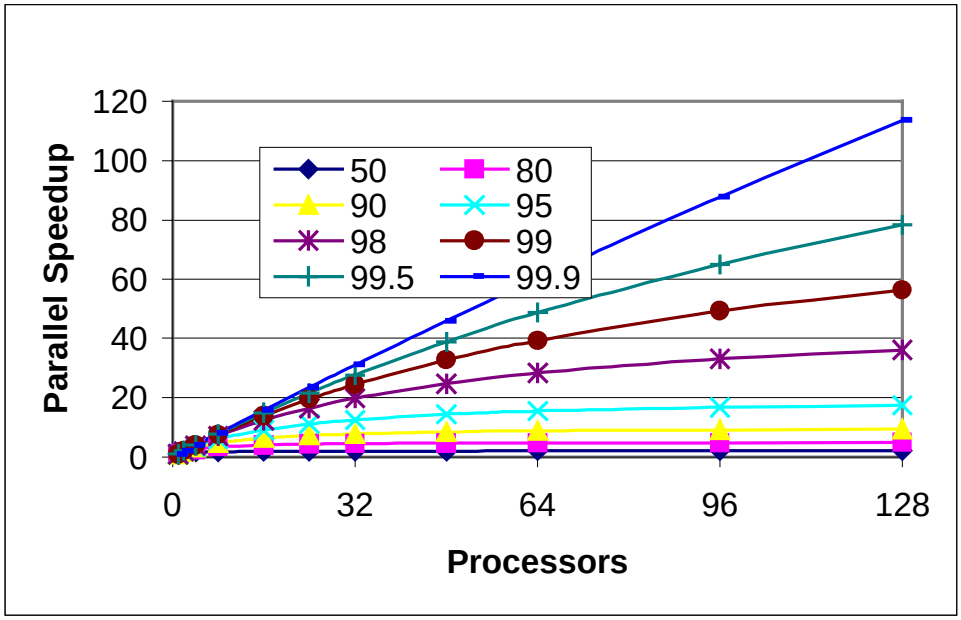

Figure 1. Amdahl's law for parallel speedup. Each line corresponds to speedup for a particular percentage of parallel instructions, from 50% to 99.9% parallel.

Default Partitioning

There are several options available for partitioning the LS-DYNA model. These are documented in the LS-DYNA Users' manual. The default method used for partitioning is RCB (recursive coordinate bisection). The other methods available are RSB (recursive spectral bisection) and "greedy." The default RCB method usually produces the shortest elapsed time, so the others will not be investigated here. The partitioning method can be specified in the partitioning file, an optional file used by the MPI LS-DYNA binary, where the user can specify several options relating to partitioning.

A useful tool for investigating the partitioning is the *show* command in the partitioning file. Enabling this option will cause LS-DYNA to halt just after initialization, with the partitioning information graphically contained in the D3PLOT file. Figure 2 shows the NCAC Taurus model partitioned into four domains.

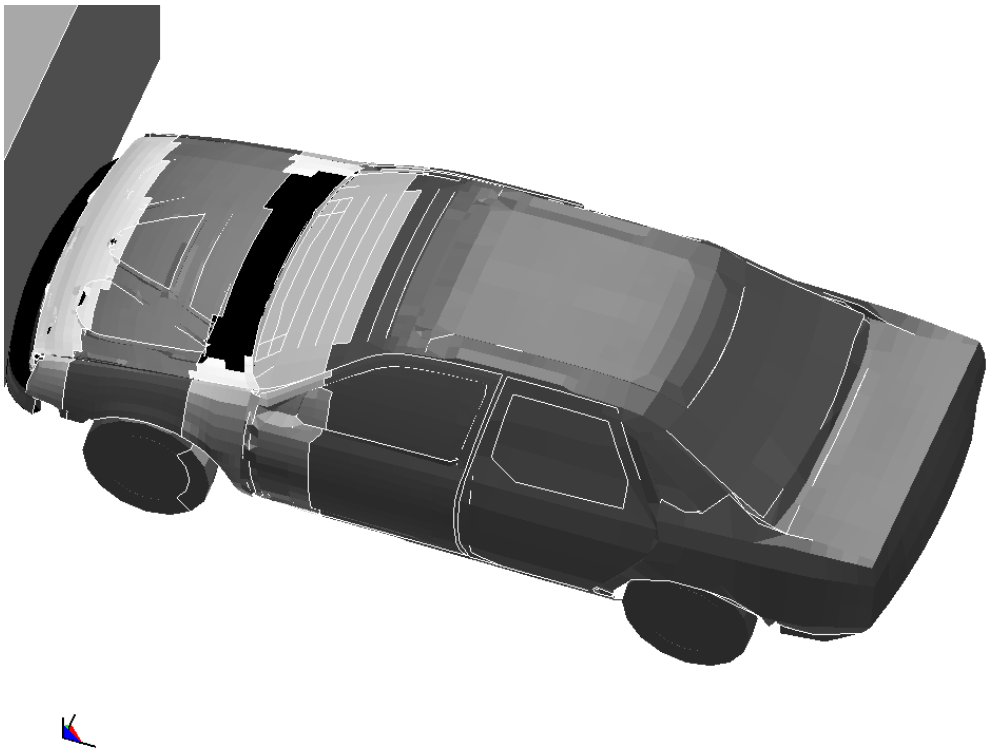

Figure 2. Default partitioning of the NCAC Taurus model - 4 domains.

Geometric Partitioning

While using RCB, the partitioning can be controlled using both the geometry and the contact surfaces of the model. This is useful because the default partitioning is sometimes less than ideal. For instance, the NCAC Taurus model of Figure 2 is involved in a front crash, so that most of the contact searching will take place in the front of the vehicle. Therefore, the two domains at the front of the vehicle will require more computation than those at the rear of the vehicle, leading to load imbalance. A better domain decomposition is something like that which is displayed in Figure 3, where most domains runs from the front of the vehicle to the back, so that all domains have similar contact characteristics, and better overall load balancing.

Figure 3. Customized partitioning of the NCAC Taurus model - 8 domains.

This type of domain decomposition can be achieved using the geometry options *expdir* and *expsf*. These options will expand the model in a certain direction (expdir) by a certain scale factor (expsf) before the domain decomposition occurs. A typical usage is

decomposition (expdir 2 expsf 10)

which will stretch the model out by a factor of 10 in the Y direction. Once this is done, the regular partitioning algorithm will generally partition the model in long strips from the front to the rear of the vehicle, as desired.

Partitioning According to Contact Surfaces

For some crash input decks, the model can be efficiently partitioned according to the contact surfaces (the sliding interfaces) through use of the *silist* option in the partitioning file. A typical front crash model may have one or perhaps a few sliding interfaces which involve the elements toward the front end of the vehicle, where the contact occurs. Overall computational cost is lessened by not making elements towards the rear of the vehicle members of these sliding interface sets. Some less important sliding interfaces may exist in order to handle particular cases of contact. By assigning the more important sliding interfaces to the silist, MPI LS-DYNA will first partition the elements associated with those interfaces, and then follow with the remainder of the elements. This helps prevent load balance problems, since the primary contact surfaces are generally evenly distributed among the processors.

LS-DYNA PERFORMANCE IN STAMPING SIMULTATIONS

Even though it is a general-purpose code, LS-DYNA has historically been used primarily for crash simulation and metal stamping, so those applications will be examined in this study. This portion of the study examines some of the performance characteristics of stamping simulations.

General Scalability of Stamping Input Decks
The scalability of MPI LS-DYNA increases as the number of elements in each model grows. In order to assess the practical limits of scalability, several metalstamping input decks of various sizes were provided by Volvo Car Company

Model Identifier	*Size (elements)*
Stamp-141	141,000
Stamp-257	257,000
Stamp-655	655,000
Stamp-1090	1,090,000

Each input deck was run on a range of processors, between 1 and 64, and the timing information was recorded from the D3HSP file. For each input deck, both the initialization phase plus the simulation phase elapsed times decrease as more processors are used for the analysis. Figure 4 shows an example of such data.

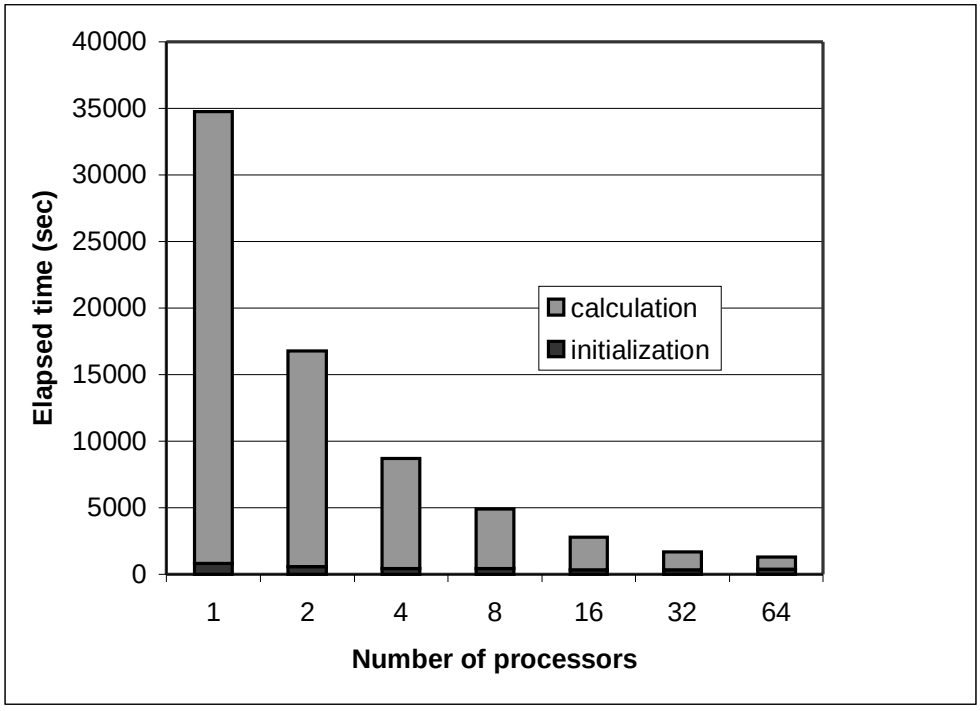

Figure 4. Parallel speedup for the Stamp-655 input deck.

Data for all four input decks is summarized in Figure 5. In this plot, the vertical axis represents the elapsed time speedup which occurs as the number of processors is doubled. Two effects are obvious:

1. As more and more processors are used, the gain from parallelism decreases. However, for the larger models it is still reasonable to use between 32 and 64 processors efficiently.
2. Better parallelism is observed for the larger models. For the smallest model used in this study, the elapsed time actually increased as the number of processors went from 32 to 64. Smaller models generate too much overhead which can't be overcome with parallelism at these processor counts.

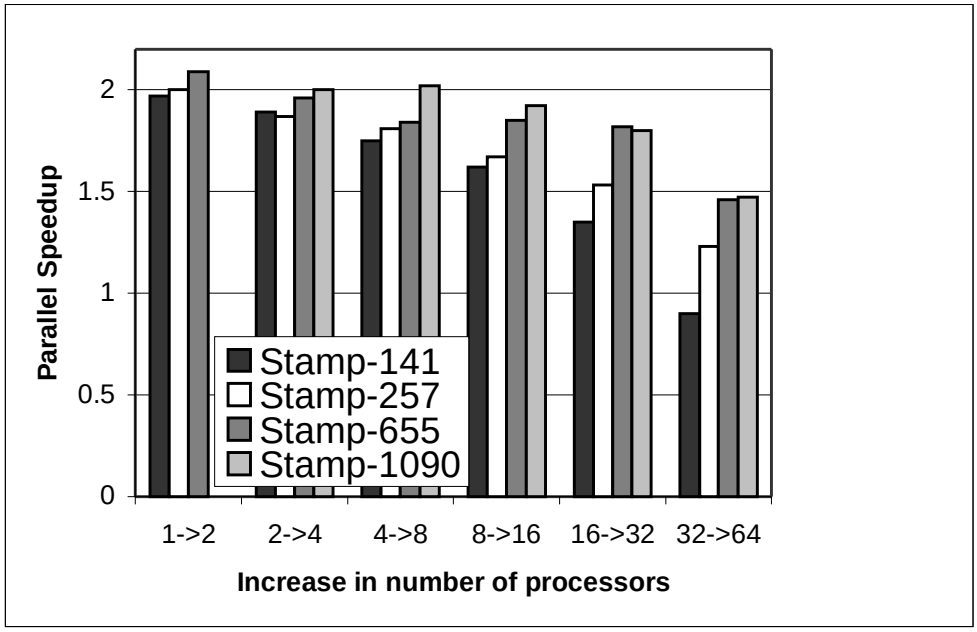

Figure 5. Parallel speedup of all Stamping input decks.

Partitioning Options for Stamping

The LS-DYNA Users' Manual recommends the following partitioning file options for stamping input decks (assuming that the direction of travel for the punch is in the Z direction):

```
decomposition ( expdir 3 expsf 0 )
```

This essentially flattens out the input deck geometry before partitioning, and the model is partitioned in a series of columns in the Z direction.

Partitioning Study with a Metal Stamping Input Deck

Two of the Volvo input decks were run to completion as a test of the partitioning options. Each deck was run according to the LSTC recommendation, plus according to the other two directions, as a comparison. Results of this experiment are shown in Figures 6 and 7. These figures verify that the recommendations in the Users' Manual are correct, demonstrating that partitioning in the Z direction can substantially improve elapsed time performance. In addition, the recommended solution appears more robust, since partitioning in the other

directions sometimes results in the simulation stopping at a point just before the planned completion of the analysis.

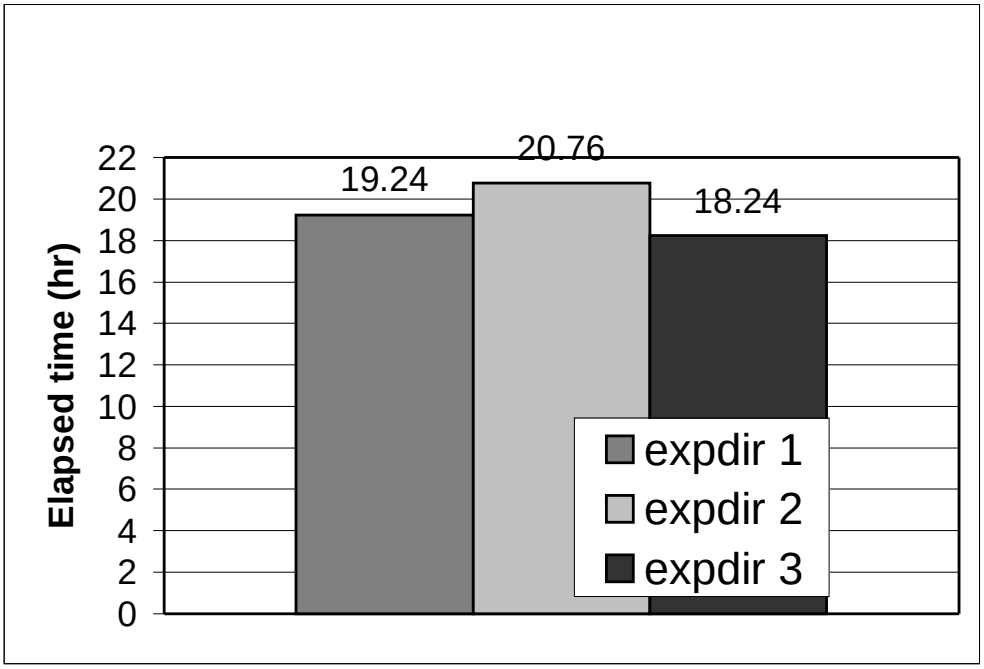

Figure 6. Partitioning results for the Stamp-257 model.

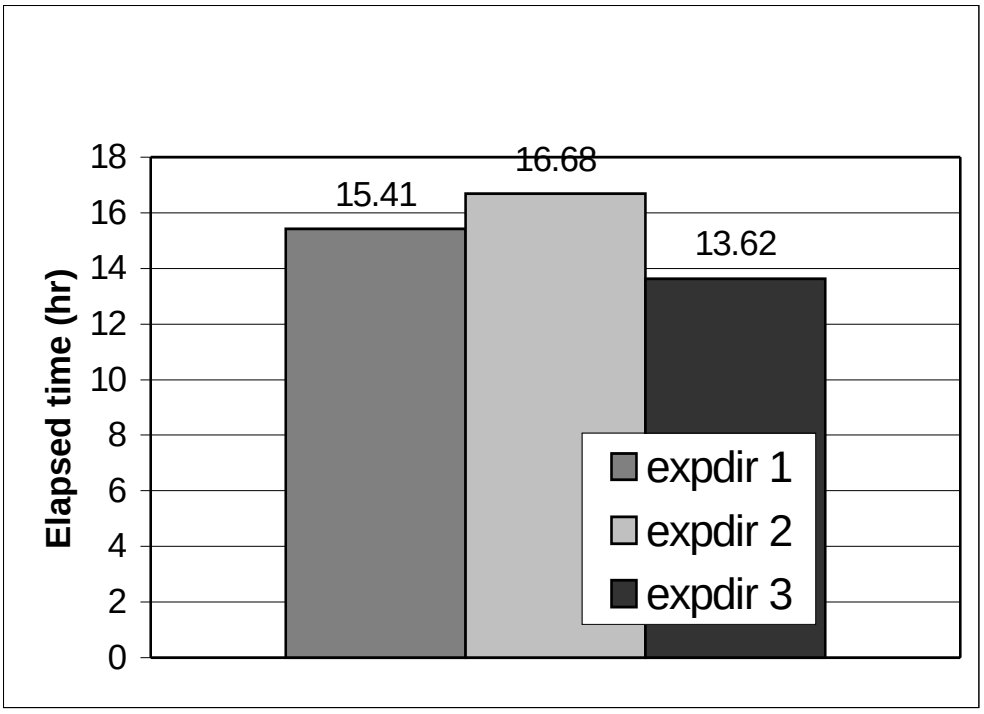

Figure 7. Partitioning results for the Stamp-141 model.

LS-DYNA PERFORMANCE IN AUTOMOTIVE CRASH SIMULTATIONS

General Scalability of Automotive Crash Input Decks
General scalability of MPI LS-DYNA for automotive crash was measured by testing a variety of input decks:

Elements	*Model Identifier*
5,500	WPI Rigid Pole
28,000	NCAC Taurus
100,000	Customer Front Offset
275,000	NCAC Neon

Figures 8-11 show scalability results for these input decks, showing how the elapsed time decreases as the number of processors increases. In general, performance of these input decks is very similar to the tendencies observed in the stamping scalability study.

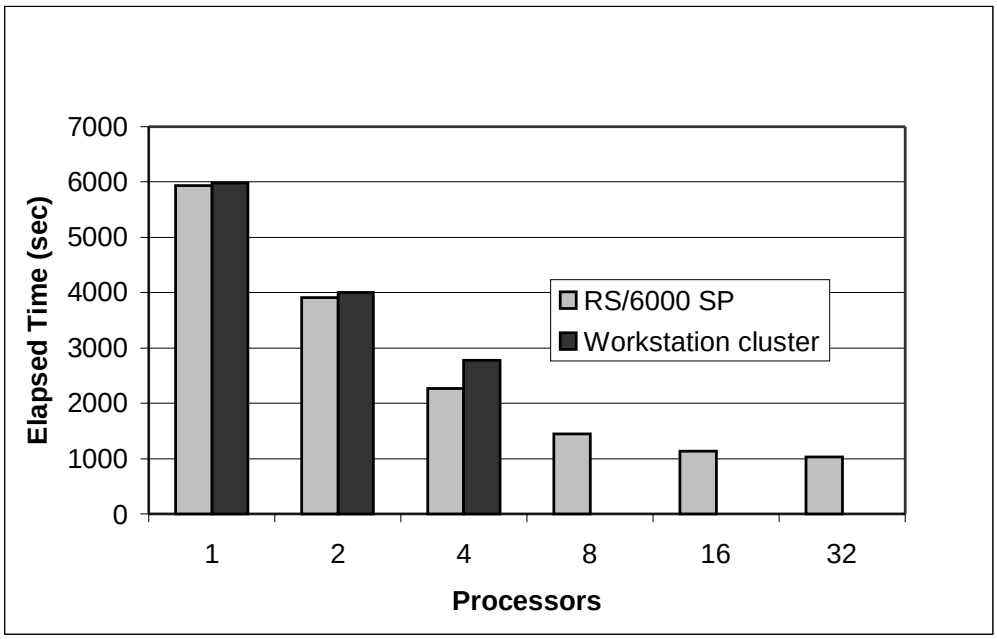

Figure 8. WPI Rigid Pole scalability.

Scalability on a workstation cluster
Use of the domain decomposition version of LS-DYNA allows users to run simulations on a network of computers. As a first step in evaluating the efficiency of that concept, all of the automotive crash input decks and one of the stamping input decks were run on a small network of workstations in addition to the IBM RS/6000 SP computer. This network of workstations consisted of two IBM 43P model 260 workstations, each with a pair of 200 MHz POWER3 processors, identical to those found in the RS/6000 SP system. Therefore, the only significant difference in the two configurations was the interconnection:

System	RS/6000 SP	RS/6000 workstation cluster
Communication	switch	100T Ethernet
Latency (microsec)	24	high
Bandwidth (MB/sec)	133	12.5

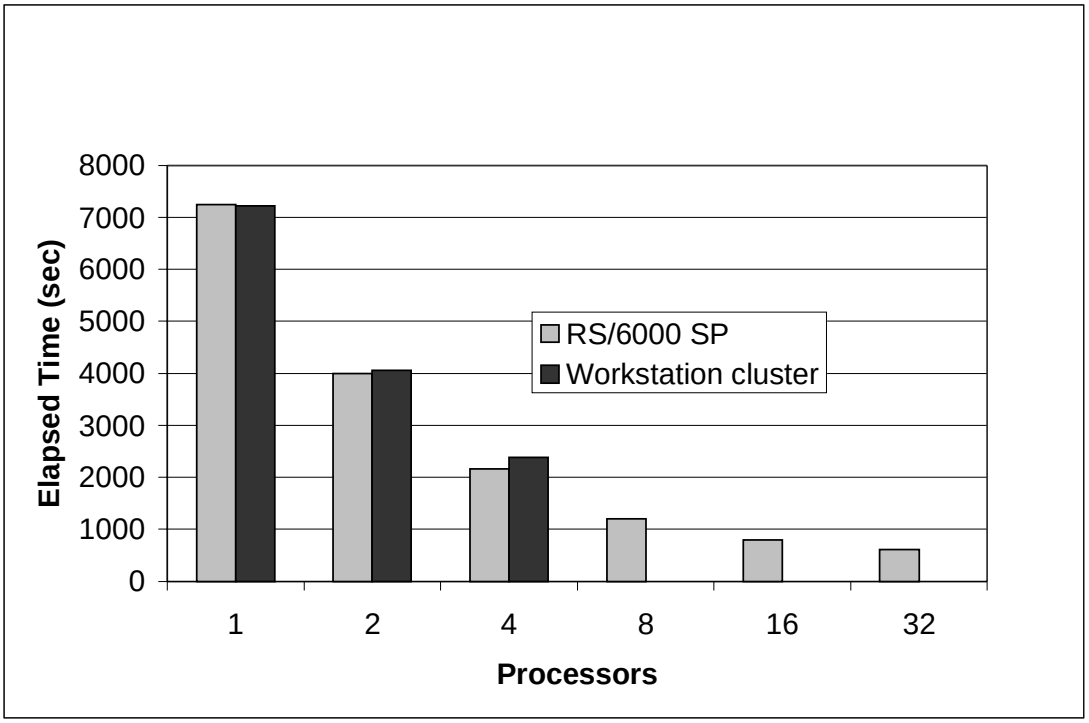

Figure 9. NCAC Taurus scalability.

Therefore, this experiment was able to demonstrate the importance of the role of communication in the performance of the domain decomposition version of LS-DYNA.

As expected, the results for one and two processors (see Figures 8-12) are very close, since at this processor count all calculations reside on one node of the RS/6000 SP system or one workstation, and there is no communication between nodes.

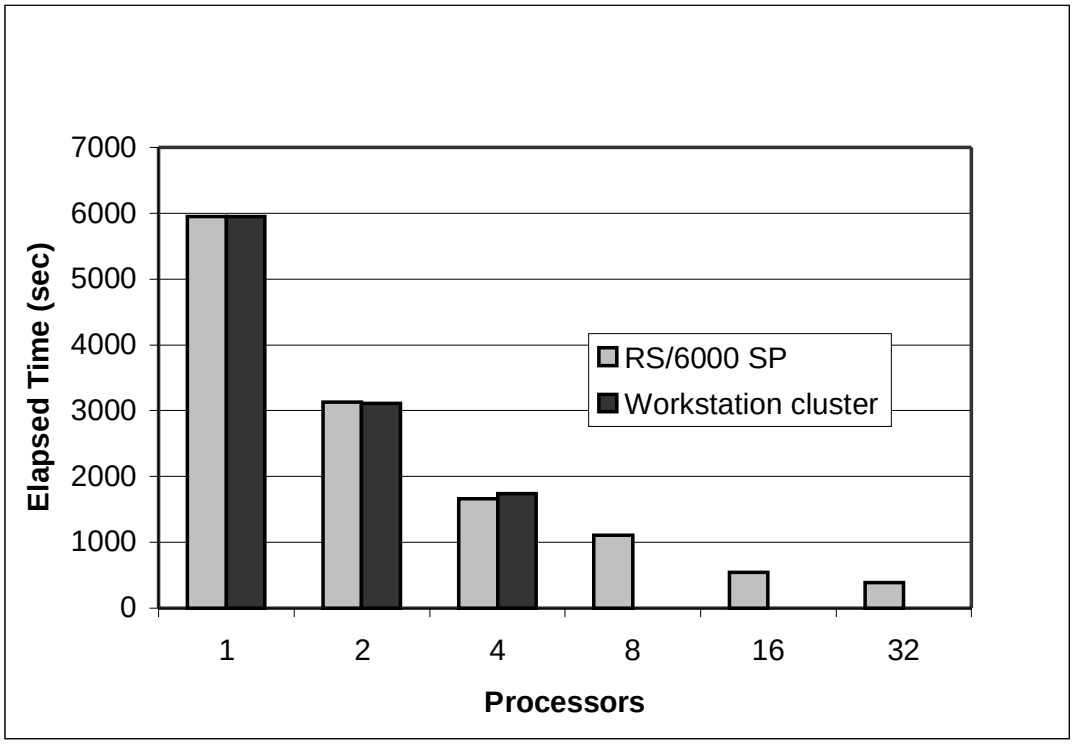

Figure 10. Customer 100,000 element model scalability.

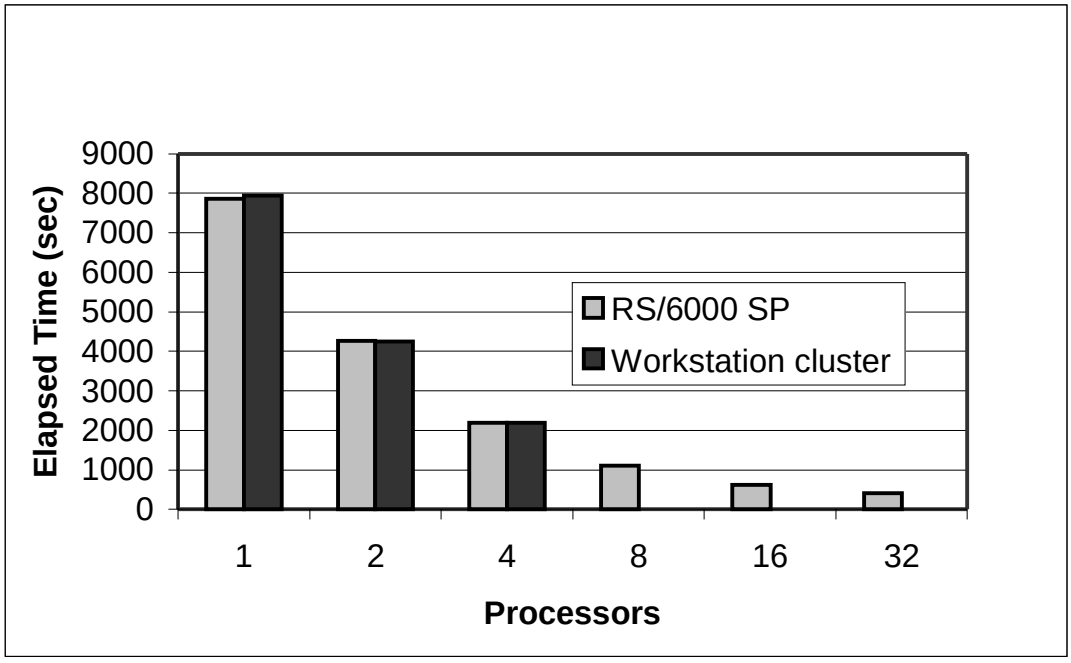

Figure 11. NCAC Neon scalability.

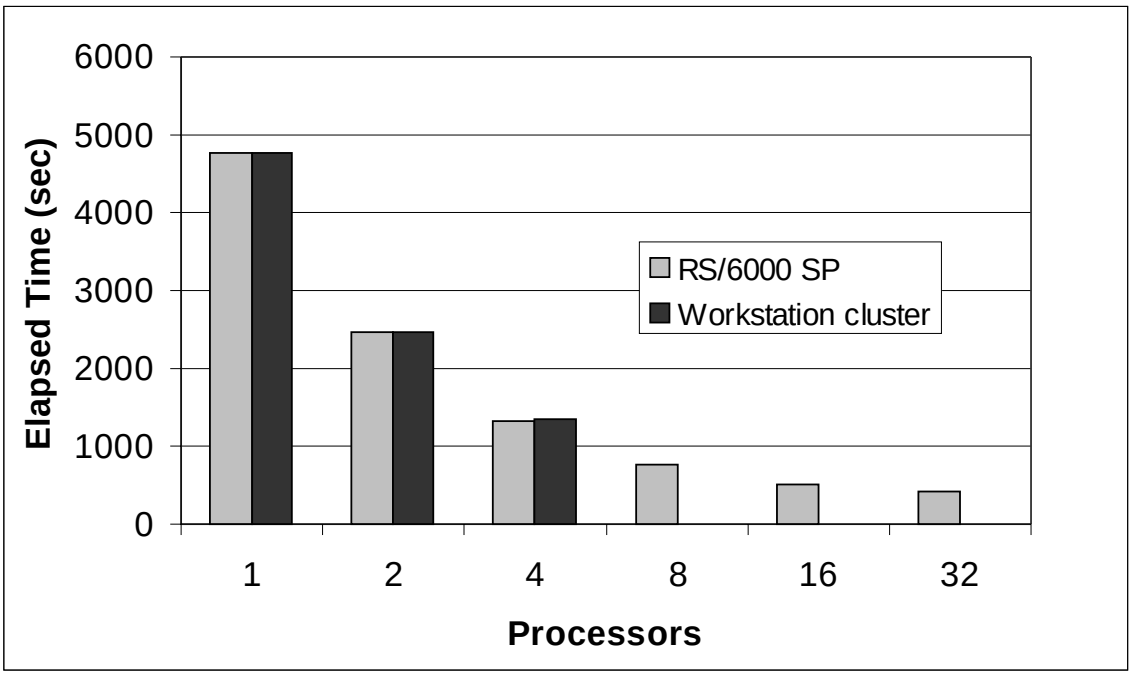

Figure 12. Stamp-141 scalability.

When using four processors, the slower ethernet communication causes a slight increase in the overall elapsed time of the jobs, but many users should find this an attractive and economical way to utilize their computer resources.

In order to gather data with convenient elapsed times, the larger simulations in this study were shortened in duration. One possible explanation of the results is that the communication characteristics would change as the simulation progresses, where contact plays a more important role. Therefore, the metalforming and 100,000 element crash simulations were both run to completion. A comparison of the elapsed times shows that even for the full simulations, the workstation cluster is only 3% slower than the RS/6000 SP system, so the partial simulation results are valid in comparing timings.

Partitioning Study with some Automotive Crash Input Decks
The three largest representative crash simulation input decks were run using various settings for partitioning, to determine the effect on performance. The results are summarized in Figures 13-15. These bar charts show how the elapsed times of the jobs change for various numbers of processors, for both default partitioning and partitioning using options such as:

decomposition (expdir 2 expsf 10.0 silist 1,2)

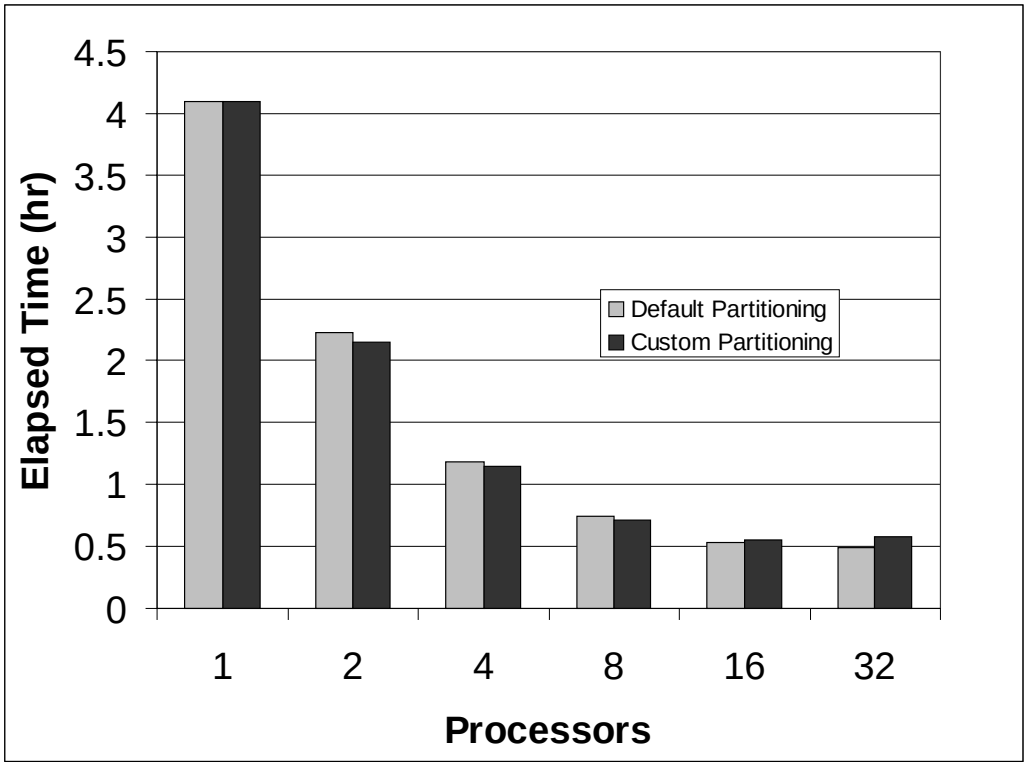

Figure 13. Partitioning effects on the NCAC Taurus model.

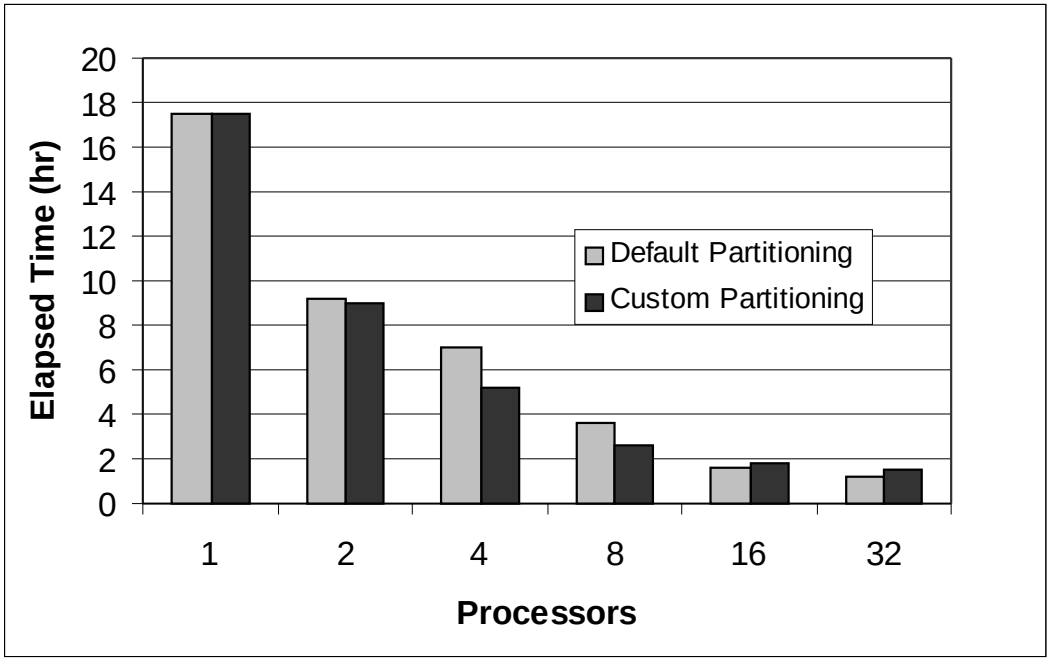

Figure 14. Partitioning effects on the 100,000 element customer model.

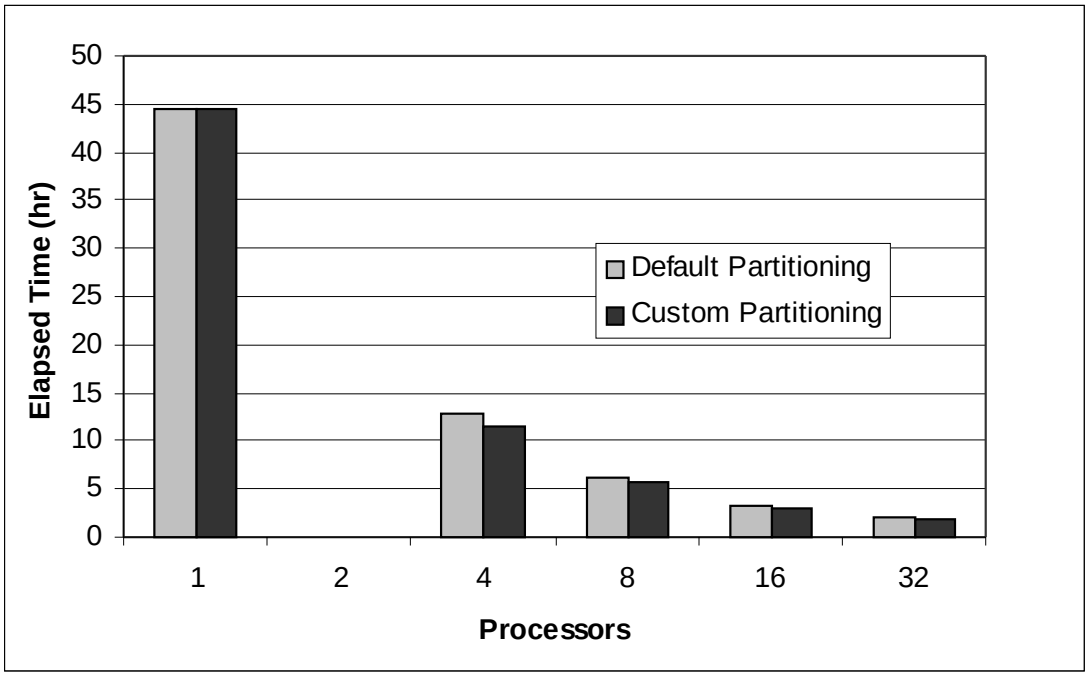

Figure 15. Partitioning effects on the NCAC Taurus input deck.

Two general trends concerning usage of these partitioning options can be observed from the performance data:

1. Customized partitioning does improve performance when using 4 or 8 processors.
2. Customized performance degrades performance when using 16 or more processors.

The most significant performance change is observed for the 100,000 element model, where customized partitioning with 4 processors improves the performance from 7.03 hours to 5.16 hours elapsed time, a gain of 36%.

SUMMARY

For stamping simulations, it is prudent to follow the recommended partitioning settings described in the LS-DYNA Users' Manual. This has been show to provide a faster and more robust solution. For automotive crash simulations using up to 8 processors, partitioning with the silist, expdir, and expsf options can improve performance, up to 36% for one of the examples studied here. For 16 or more processors, default partitioning provides better performance than customized partitioning.

ACKNOWLEDGEMENTS

Many people and organizations contributed to this effort. Particular thanks go to the following:

Volvo Car Corporation - Olofstrom stamping group, for use of the stamping input decks
NCAC - for use of the public domain Taurus and Neon crash simulation input decks
WPI - for use of the public domain rigid pole input deck
LSTC - for overall assistance, guidance, and provision of required binaries and license files