Crash Simulation of Large-Number-of-Elements Car Model by LS-DYNA on Highly Parallel Computers

● Kenshiro Kondo ● Mitsuhiro Makino

(Manuscript received March 18, 2008)

The analysis of car crashes by computer simulation has become an indispensible tool for shortening automobile development time and lowering costs. To shorten development time even further, researchers have endeavored to shorten the time needed for creating analysis models, and it is now becoming possible to create analysis models in a short time without manual work. However, to obtain a practical level of accuracy in analysis when creating an analysis model automatically, one needs a level of detail more than ten times that of analysis models currently being used by automobile companies, but the time taken to perform such an analysis is impractical. To resolve this problem, we investigated the possibility of performing practical analysis by highly parallel computing on a parallel computer. This paper explains the need for large-number-of-elements crash analysis at automobile companies and describes the highly parallel execution with up to 512 processors of a 10-million-element analysis model using the nonlinear dynamic structured analysis program LS-DYNA. The technical information obtained from these simulations, measures for improving processing speed, and future research issues are presented.

1. Introduction

LS-DYNA[1)] is a nonlinear dynamic structure analysis program developed by Livermore Software Technology Corp. (LSTC) in the USA. It can analyze large deformation behavior in structures by the explicit time integration method, and it is widely used in diverse fields from car crash analysis to dropped cell-phone impact analysis. In addition to being an LS-DYNA distributer, Fujitsu has partnered with LSTC to help develop various versions of LS-DYNA, including a vector-supercomputer version, a shared memory parallel (SMP) version (parallel processing using shared memory) using the OpenMP language, and a massively parallel processing (MPP) version (parallel processing using distributed memory) using the message passing interface (MPI) language.

The most important application field of LS-DYNA is the analysis of car behavior at the time of a crash. In this field, analyzing car crashes by computer simulation helps to reduce the number of experimental vehicles that must be built, shortens development time, and lowers development costs. From here on, to shorten development time even further, it will be necessary to shorten the analysis time and the analysis-model creation time.

In this paper, in light of the rapid increase in the cost performance of PC clusters, which is making highly parallel processing feasible, we describe the present state of car-crash analysis and introduce the following related research issues.

- Current state of crash analysis and analysis technology in automobile companies
- Shortening of time required for creating large-number-of-elements crash analysis

models and its effect
- Evaluation of analysis time of large-number-of-elements crash analysis models
- Increase in simulation speed of large-number-of-elements crash analysis models
- Future research issues

2. Current state of crash analysis and analysis technology in automobile companies

In this section, we describe the current size of crash analysis models, the state of LS-DYNA parallelization, and the state of associated hardware in automobile companies.

2.1 Scale of analysis models

Japanese automobile manufacturers are working to improve vehicle structure to ensure crash safety based on safety standards developed by each company such as Toyota's "GOA" and Nissan's "Zone Body". As reflected by statistical data presented in a traffic safety white paper,[2] a crash-safe body structure in conjunction with seat belts and air bags can reduce the number of fatalities. Future research will aim to lower the level of bodily harm suffered by injured passengers.

In computer simulations, two ways of improving the accuracy of crash safety analysis of a vehicle structure are to increase the number of components in the analysis model and to represent vehicle geometry more accurately. For this reason, the number of elements in analysis models has been increasing steadily from 10 000 elements in the latter half of the 1980s to 100 000 elements[3] in the mid-1990s and 1 million elements in recent years. This increase in the number of elements in analysis models became possible through improvements in software and enhanced processing capabilities of computers.

2.2 LS-DYNA parallelization

In conjunction with this increase in the number of elements in analysis models, LS-DYNA has increased processing speed by introducing parallel processing. Initially, vector processing was introduced as an extension of serial processing, but processing power was limited because of single-CPU vector processing. It was decided to execute in parallel the sections of reiterative processing, that is, the DO loop in the FORTRAN language version of a program. This was done on a computer in which multiple CPUs share memory using an SMP version of LS-DYNA, which was the first version of LS-DYNA to appear as a parallel-processing program. In this method, since multiple CPUs are accessing shared memory simultaneously, data supplied from memory to the CPUs is more likely to be late when the degree of parallelism increases. As a result, no improvement in performance can be expected for ten or more processors. The MPP version of LS-DYNA, which was developed to solve this problem, makes use of a computer in which each CPU has its own memory and where all CPUs are interconnected in a network. In this version, prior to computer processing, a process called domain decomposition is performed. Here, the model is decomposed into sub-domains, each of which uses memory dedicated to a CPU. This scheme makes for high parallel-processing performance even when the number of processors is more than ten. Fujitsu has collaborated with LSTC to develop a commercial MPP version of LS-DYNA to expand sales of its scalable cluster computers. In the mid-1990s, the time required to analyze an 80 000-element analysis model with a vector supercomputer was 30 to 40 hours. Toward the end of the 1990s, the time required to analyze a 180 000-element analysis model with an SMP supercomputer was about 80 hours. At present, it takes 20 to 40 hours to analyze a million-element analysis model with an MPP computer having from 16 to 32 processors.[3]

2.3 Hardware

The platform of choice for crash analysis by LS-DYNA has changed from vector machines and scalable machines using shared memory to PC clusters. In a PC cluster (such as PRIMERGY RX200 S2/RX200 S3), individual clusternodes are interlinked by a high-speed InfiniBand interconnect.

For a certain analysis model of crash analysis, an improvement in cost performance of about 200 times was observed when running eight RX200 S3 servers in parallel, in which each server has two Intel dual-core CPUs and the servers are connected by InfiniBand, compared with executing the same process on a VPP5000 vector computer. For the future, we can expect improvements in the performance of single CPUs to slow down and the development of multi-core CPUs (in which the CPU contains two or more cores) to become mainstream. Cost performance should improve all the more if multi-core CPUs can be used efficiently. At present, the use of 16 to 32 parallel cores per job is considered practical in car crash analysis, but in the near future, we foresee the introduction of large-scale PC cluster systems exceeding 1000 cores even in private enterprises and the coming of practical large-scale analysis in excess of 100 parallel cores per job.

3. Shortening of time required for creating large-number-of-elements crash analysis models and its effect

To shorten development time, we need to think not only about shortening the time it takes to execute analysis by computer but also about shortening the time it takes to create an analysis model for use by Computer-Aided Engineering (CAE). The creation of an analysis model has traditionally required a designer to accurately represent the geometry of components included in the model by hand, but if the number of components is large in order to get more accurate results, a period of several months might be needed. Against this background, research into software for creating analysis models has been progressing, and software development companies like Altair Engineering, Inc.[4] and Engineering Technology Associates (eta), Inc.[5] that specialize in CAE preprocessing functions have come to provide BatchMeshing[6] functions that are making the automatic creation of analysis models a possibility. However, when a practical analysis model with good analysis accuracy is created automatically without manual work, the number of elements must be at least ten times that of manually prepared analysis models currently in use by automobile companies, as will be explained below.

Du Bois et al.[7] compared the accuracy of analysis models prepared by BatchMeshing and ones prepared manually. These analysis models consisted mostly of quadrilateral elements. Specifically, three analysis models of each type were prepared using three sizes of quadrilateral elements: 10 mm × 10 mm, 5 mm × 5 mm, and 2 mm × 2 mm. For the analysis models using 10 mm × 10 mm quadrilateral elements, the hand-meshed model was far more accurate than the batch-meshed model. The disparity in accuracy was smaller for the 5 mm × 5 mm models, but the batch-meshed model here was still below the level deemed practical. The 2 mm × 2 mm hand-meshed and batch-meshed models obtained equivalent levels of accuracy. In short, analysis accuracy the same as that of a manually prepared analysis model can be achieved if the analysis model is prepared using BatchMeshing with 2 mm × 2 mm quadrilateral elements.

The million-element analysis model commonly in use today consists of 6 mm × 6 mm quadrilateral elements prepared manually. In contrast, there are about 10 million elements in an analysis model prepared with 2 mm × 2 mm quadrilateral elements using BatchMeshing. Here, computational time is determined by the number of elements and Δt (computer time step size), and as explained below, the computational time of the 10-million-element model increases to 27 times that of the 1 million element model.

Because the elements used here are quadrilateral in shape, the increase in the required number of elements when changing from one element size to another can be calculated by squaring the element-side length for each element size and taking the ratio of those squares. In particular, the increase in number of elements = (6 mm × 6 mm) ÷ (2 mm × 2 mm) = 9 times. Now, according to the Courant condition[8] for determining stability in the solution of an explicit time integration method, the magnitude of Δt is inversely proportional to element length. (Δt decrease = 2 mm ÷ 6 mm = 1/3.) To calculate up to the same physical time, we must increase the number of Δt repetitions by three times when changing from a quadrilateral-element length of 6 mm to one of 2 mm.

Increase in computational time =

increase in no. of elements × (1/Δt decrease) = 27

At present, on a 32-CPU PRIMEPOWER HPC2500, it would take 20 to 40 hours to compute 1 million elements and 500 to 1000 hours to compute 10 million elements. Five hundred to one thousand hours of elapsed time is impractical even if the time needed for preparing the analysis model can be shortened. It is therefore necessary to increase the number of processors in order to increase the processing speed by 27 times. Here, we should focus our attention on the possibility that computational cost could become cheaper than the labor cost for mesh creation because the cost performance of PC clusters is improving.

4. Evaluation of analysis time of large-number-of-elements crash analysis models

Since there are no results for analyzing 10-million-element analysis models at present, no evaluation data exists. We therefore created 10-million-element evaluation data for the case of a 5-million-element car body colliding with another 5-million-element car body based on publicly available Caravan 300 000-element data.[9),10)] To investigate problems associated with highly parallel execution of a 10-million-element analysis model, we performed joint research with the Information Technology Center of Nagoya University.[11)]

Measurement environment
(installed at Information Technology Center, Nagoya University)
Hardware: PRIMEPOWER HPC2500
CPU: SPARC64V, 2.08 GHz
Operating system: Solaris9
Application: LS-DYNA MPP970R6763
Input data: Caravan 10 million elements
(contact definition: SOFT = 1)

Since executing 120 ms of physical time in a crash analysis would make the elapsed time excessively long, and because LS-DYNA repeats the same calculations for each time step, we carried out measurements for 10 ms of physical time and estimated the elapsed time for 120 ms. We found that the computational time was 192 hours for 64 processors, 70 hours for 265 processors, and 91 hours for 512 processors, indicating that performance drops when the parallelism is increased from 256 to 512.

A graph of parallel-processing efficiency for a 10-million-element crash analysis model is shown in **Figure 1**. The horizontal axis represents the number of processors and the vertical axis shows parallel processing efficiency with a parallelism of 16 taken to be “1”. Computation in LS-DYNA can be classified into element calculation, contact calculation, and other types of calculation. As shown by the broken line in the graph, the parallel-processing performance of element calculation improves as the number of processors increases, but that of contact calculation drops after 96 processors. As a result, the total processing time for 256 processors is only 10.7 times versus the ideal value of 16 times, and for 512 processors, it drops to as low as 9.2 times the ideal value of 32. These results reveal the importance of improving the speed of contact

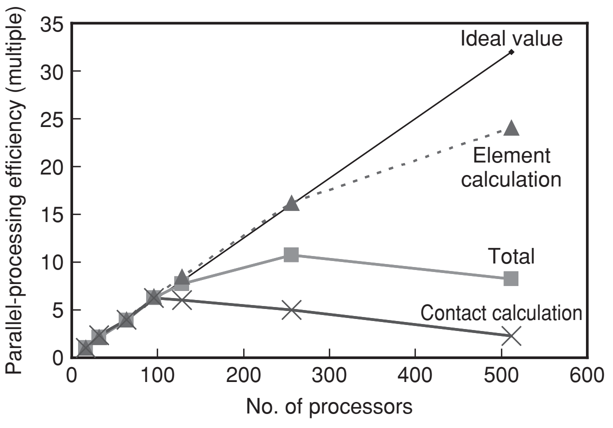

Figure 1
Parallel-processing efficiency for a 10-million-element model (before performance improvements).

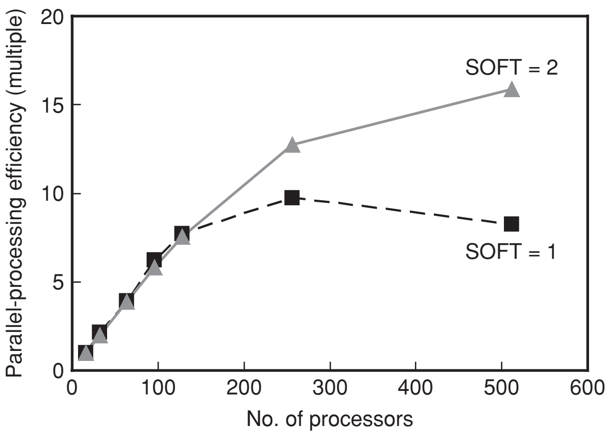

Figure 2
Parallel-processing efficiency for two types of contact definitions.

calculation for parallel processing with more than 96 processors.

5. Increase in simulation speed of large-number-of-elements crash analysis models

This section focuses on measures for increasing the simulation speed of large-number-of-elements crash analysis models. Factors that have a major influence on the performance of a parallel computer are contact calculation (as shown in Figure 1), load balance, and data communication among processors. Measures for improving performance in each of these areas are described below.

5.1 Performance improvement measures

1) Optimization of contact definition method

As shown in Figure 1, the efficiency of contact calculation decreases as the number of processors for parallel processing is increased. We investigated the reasons for this. In LS-DYNA, there are two types of contact processing methods called SOFT = 1 and SOFT = 2. In SOFT = 1, when two components make contact, one is treated as a node and the other as a segment in assessing the contact. Then, after the contact assessment has been completed, the roles of these components are reversed and contact-assessment processing is performed again. In SOFT = 2, both components are treated as segments in assessing contact. **Figure 2** shows how the parallel processing efficiency differs between SOFT = 1 and SOFT = 2. The horizontal axis represents the number of processors and the vertical axis shows the parallel-processing efficiency with a parallelism of 16 taken to be “1”. While no difference in efficiency can be observed between SOFT = 1 and SOFT = 2 up to 128 processors, SOFT = 2 becomes more efficient than SOFT = 1 for 256 and 512 processors. For SOFT = 1, we think that the factors impeding performance are performing node-segment-based contact assessment twice and transferring many small messages simultaneously as parallelism increases. SOFT = 2 can be seen to be an efficient contact processing method for highly parallel processing.

2) Improvement of load balance

The process wait time among CPUs can be reduced by making the amount of calculation performed by each CPU uniform. Domain decomposition in LS-DYNA makes the number of elements to be processed by each CPU uniform but does not take into account the amount of contact calculation. Contact calculation can therefore concentrate at a certain CPU, which

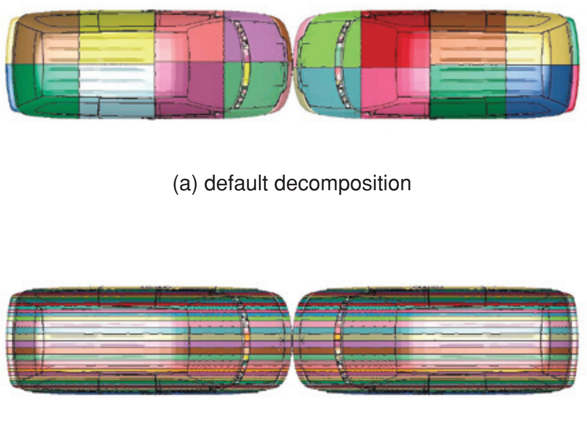
(a) default decomposition
(b) domain decomposition in direction of car motion

Figure 3
Comparison of methods of domain decomposition on 32 processors.[12)]

increases the load on that CPU and degrades the load balance. Thus, to make the load associated with contact calculation uniform, the user must choose appropriate settings.

Domain decompositions for the default decomposition method and for decomposition in the car's direction of motion are compared in **Figure 3**. On testing various types of domain decomposition, we found that domain decomposition in the direction of car motion was efficient for low parallelism of up to 32 processors. This is because—in a head-on crash—this type of domain decomposition uniformly distributes the amount of contact calculation within the engine compartment where complex components are located. Consequently, we decided that domain decomposition in the direction of car motion would be used here, even for highly parallel processing.

3) Improvement of data communication among processors

Contact calculation requires that data be transferred between nodes in order to handle contact between adjacent sub-domains. The characteristics of data communication in highly parallel processing were analyzed using a data-communication analysis tool created by Fujitsu and Fujitsu Laboratories. This analysis revealed the following characteristics of data communication in LS-DYNA.

- Many calls are made to the MPI_all_to_all() collective communication function. One call to this function performs $N_{CPU} \times N_{CPU}$ instances of communication, resulting in a high communications processing cost.
- One-to-one communication functions like MPI_Send()/MPI_Recv() can involve data communication in excess of 1 MB, while almost all calls to MPI_all_to_all() involve data communication under 1 KB.
- Contact calculation involves much waiting in MPI_all_to_all() processing.

On the basis of these findings, we enhanced the MPI_all_to_all() function in Fujitsu's MPI library to allow buffering so that communication data could be grouped together before transfer. This had the effect of cutting down on the number of data transfers and grouping together data items under 1 KB for batch transfer. This lowered the communication overhead and improved communication efficiency.

5.2 Results of performance improvements

A graph of parallel-processing efficiency for a 10-million-element crash analysis model is shown in **Figure 4**. The horizontal axis represents the number of processors and the vertical axis shows parallel processing efficiency with a parallelism of 16 taken to be "1". Before the performance improvements (Figure 1), contact calculation showed a parallel processing efficiency of 5 times for 256 processors and only 2.5 times for 512 processors. After the improvements, however, the parallel processing efficiency was more than 10 times in each case. **Figure 5** shows the results of estimated elapsed time for 120 ms of physical time, which is required for crash phenomena, from the execution of 10 ms of physical time in 10-million-elements crash analysis. Because of these improvements, the analysis time could be reduced from 91 hours to 43 hours, which enables the analysis to be achieved in a

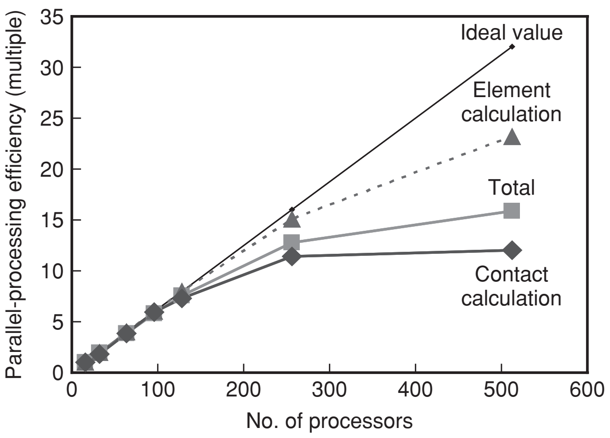

Figure 4
Parallel-processing efficiency for a 10-million-element model (after performance improvements).

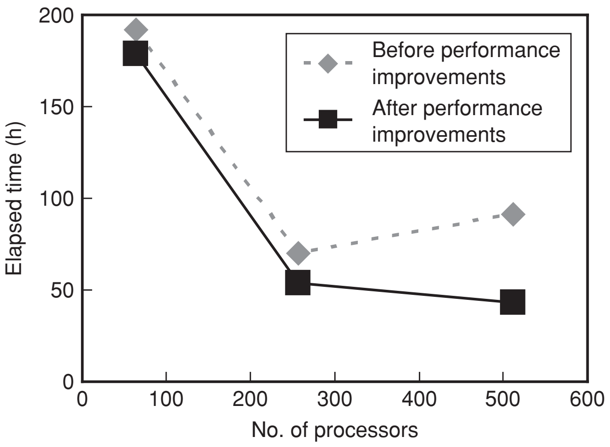

Figure 5
Estimation of elapsed time for a 10-million-element model (120 ms) before and after making performance improvements.

realistic timeframe.

6. Future research issues

To improve the scalability of highly parallel processing even more, we are undertaking the following measures in conjunction with LSTC, the developer of LS-DYNA, with the aim of implementing them in upcoming versions.

1) CONTACT_FORCE_TRANSDUCER function
   The use of many contact definitions in actual analysis models means that the load balance can easily deteriorate. To prevent this from happening, this function minimizes contact definitions and measures necessary contact forces.
2) Domain decomposition oriented to highly parallel processing taking into account the communication cost of contact calculation.
3) Optimization by grouping together data for batch transfer to make the most of the transfer performance of InfiniBand.

7. Conclusion

This paper introduced technology for analyzing large-number-of elements car crash analysis models by highly parallel processing. For parallel processing in excess of 100 processors, our examination of results based on the differences between the SOFT = 1 and 2 contact processing methods and on improvements made to MPI communication processing performance revealed that the performance of data communication among processors during contact calculation was of prime importance in raising parallel processing efficiency. We found that a 10-million-element car crash analysis model could be analyzed in a realistic period of time after making improvements to the communication performance.

The knowledge gained from this research on data communication performance will be used as feedback in the development of hardware and basic software with the hope of raising the speed of large-scale calculations in the future.

We would like to extend our deep appreciation to the personnel at the Information Technology Center, Nagoya University, who kindly gave their support in carrying out actual measurements.

References

1) Nonlinear dynamic structure analysis program LS-DYNA.
http://www.lstc.com
2) Cabinet Office Director-general for Policies on Cohesive Society: White Paper on Traffic Safety. (in Japanese).

http://www8.cao.go.jp/koutu/taisaku/h16kou_haku/genkyou/01010101.html

3) P. A. Du Bois: Crashworthiness Engineering Course Notes. Livermore Technology Corporation, I.5, January 2004.

4) Altair Engineering, Inc. *http://www.altair.com/Default.aspx*

5) Engineering Technology Associates, Inc. *http://www.eta.com/*

6) J. Seybold et al.: BatchMeshing and CAE Data Management as Key Technologies for Six Sigma Compliant CAE Processes for LS-DYNA Simulations. LS-DYNA Anwenderforum, Ulm2006. *http://www.dynamore.de/download/af06/papers/L-I-3.pdf*

7) P. A. Du Bois, et al.: A Study of Mesh Sensitivity for Crash Simulations and Batchmeshed Models. 4th DYNAmore LS-DYNA conference Bamberg 2005I-I-30, October 2005.

8) Courant Condition. *http://www.dynamore.de/download/lstc/nov_2001.pdf*

9) CARAVAN. *http://www.topcrunch.org*

10) National Crash Analysis Center. *http://www.ncac.gwu.edu/*

11) M. Makino: The Performance of Large Car Model by MPP Version of LS-DYNA on Fujitsu PRIMEPOWER. 9th International LS-DYNA Users Conference 2006I6-4, June 2006.

12) National Crash Analysis Center.: Benchmark Models. *http://www.ncac.gwu.edu/vml/models.html*

Kenshiro Kondo
Fujitsu Ltd.
Mr. Kondo joined Fujitsu Ltd., Tokyo, Japan in 1991 and has been engaged in high-performance computing and CAE solutions in the structural analysis area. Since 2005, he has been supporting the crash analysis program LS-DYNA.

Mitsuhiro Makino
Fujitsu Ltd.
Dr. Makino received the Ph.D. degree in Electrical Engineering from Nagoya University, Japan in 1981 and joined Fujitsu Ltd. in 1981. He has been engaged in computer simulation of fluids and structures and in high-performance computing. Since 1989, he has been developing and supporting Crashworthiness software.