Fujitsu's Activities in Improving Performance of LS-DYNA Nonlinear Finite Element Analysis Software

● Kenshiro Kondo

The LS-DYNA nonlinear finite element analysis software package developed for structural analysis by the Livermore Software Technology Corporation (LSTC) is widely used by the automobile, aerospace, construction, military, manufacturing, and bioengineering industries. Fujitsu has been a partner with LSTC since 1996, supporting customers in Japan. A common application of LS-DYNA is car crash simulation. One way to improve the accuracy of the simulation results is to increase the number of elements in the analytical model. However, this increases the amount of computation, resulting in longer computation times, which goes against user expectations of quicker job turnaround when using high-performance computing systems. We report Fujitsu's activities in supporting higher speeds in a hybrid version of LS-DYNA applicable to large-scale parallel processing on the K computer and in improving the performance of the LS-DYNA package for car crash simulation.

1. Introduction

LS-DYNA nonlinear finite element analysis software[1)] is a general-purpose multiphysics simulation software package developed and sold by Livermore Software Technology Corporation (LSTC) in the United States. One of its major features is its ability to analyze large deformation behavior in structures over time. Car crash analysis is a prime target of LS-DYNA as reflected by its popularity with many automobile companies throughout the world. To evaluate the safety of their products, automobile companies use LS-DYNA to analyze structural deformation and harm to passengers when a car collides with an object or another vehicle.

Fujitsu has been selling and supporting LS-DYNA in Japan as a sales representative since 1996 and has been supporting the development of parallel-processing versions of LS-DYNA toward high-performance computing (HPC) as an LSTC partner. Fujitsu currently sells three parallelization versions of LS-DYNA:

1) Shared memory parallel (SMP) version for shared-memory-type parallel computers (thread parallelization using OpenMP language)
2) Massively parallel processing (MPP) version for distributed-memory-type parallel computers (process parallelization using the Message Passing Interface [MPI] language)
3) Hybrid version combining SMP and MPP

To support the ever increasing scale of computing, Fujitsu is currently providing technology to support higher speeds in LS-DYNA using the Hybrid version, which is deemed especially suitable for massively parallel computing.

In this paper, we introduce Fujitsu's efforts in increasing processing speed in LS-DYNA, focusing on the Hybrid version of parallel processing with car collision analysis as the target application.

2. Current state of car crash analysis and associated issues

An automobile company must satisfy a large number of laws and regulations governing collision safety when bringing a new model to market. In this process, performing crash experiments repeatedly with expensive experimental vehicles not only requires a hefty development budget but is also highly inefficient since valuable time is needed to manufacture, upgrade, and test experimental vehicles. Automobile companies have consequently come to use computer-based car crash analysis with the aims of decreasing the number of experimental vehicles that have to be made and of reducing development time. Increasing the accuracy of car crash analysis can decrease the number of experiments with actual experimental vehicles and reduce development time, so it is not surprising that improving accuracy has become of particular importance to automobile companies. Analysis accuracy can typically be improved by enhancing LS-DYNA and/or increasing the level of detail in the analysis model. In this paper, we take up only the latter method for raising accuracy. The following gives examples of achieving a more detailed analysis model.

1) Increasing level of detail in barrier model

Kojima et al. of the Toyota Technical Development Corporation have reported an example of changing the barrier model of aluminum-honeycomb material used in car crash experiments from cuboid-like solid elements to oblong shell elements.[2] According to this report, modeling by solid elements to simulate the macroscopic characteristics of aluminum-honeycomb material results in inappropriate deformation in the form of an hourglass mode that generates no strain energy. This is thought to be one reason for discrepancies between experimental and analytical values. However, modeling in more detail using shell elements—though increasing computational time by about 1.4 times—can significantly reduce the occurrence of hourglass mode and obtain analytical results much closer to experimental values.

Furthermore, when applying shell elements to the barrier model provided by LSTC, it has been found that a more detailed model than the solid-element model can be obtained although the number of elements increases about 300%.[3]

In other words, changing from solid to shell elements, while improving the accuracy of analysis owing to a more detailed model, increases the number of elements, thereby increasing computational complexity.

2) Increasing level of detail in human model

Version 4 of the THUMS[4] virtual human model developed by Toyota Motor Corporation features detailed models of internal human organs, which had not been included in previous versions. This makes it possible to evaluate injuries to internal organs caused by automobile collisions. The amount of information in this new model is about 14 times that in the previous version, which greatly increases computational complexity.

As reflected by the above two examples, improving analysis accuracy generally increases computational complexity. Without getting too theoretical, computational complexity in a program like LS-DYNA using an explicit method is determined by the number of elements and the time step size (Δt). Increasing the number of elements in the automobile model from one to ten million increases computational complexity about 27 times.[5] To deal with such an increase in computational complexity, LSTC and Fujitsu are working together to speed up calculations through parallel processing.

The SMP version of LS-DYNA, which runs with a degree of parallelism of 4 to 16, was widely used from the second half of the 1990s through the first half of the 2000s. The MPP version then became popular with a parallelism of several 10s to 100. Looking forward, however, we can foresee a need for even higher degrees of parallelism as the level of model detail and computational complexity continues to increase. The MPP version, though, incurs greater overhead associated with inter-process communication as parallelism increases, which means that completing analysis in a realistic period of time will be difficult on the basis of MPP alone. Against this background, the Hybrid version of parallel LS-DYNA, combining the SMP and MPP versions, was developed as a means of coping with the demand for even higher processing speeds.

For the above reasons, Fujitsu is now working to speed up LS-DYNA processing by applying the Hybrid version, but, at the same time, it is also studying other means of increasing processing speed. The following section summarizes the approaches being taken.

3. Increasing LS-DYNA processing speed

Fujitsu's is taking three main approaches to increasing the processing speed of LS-DYNA: increasing speed by using the Hybrid version, achieving even higher speeds by using the Hybrid version with a hardware barrier, and improving computational load balance through visualization.

3.1 Increasing speed using Hybrid version

Compared to the MPP version, the Hybrid version can significantly reduce the amount of message data and frequency of message sending in communication between processes. The Hybrid and MPP versions were compared in terms of communication using a two-million-element car crash model[6] available on the Web. The Hybrid version was tested with the number of operating threads per process set to 4. This means that running the Hybrid version with a parallelism of 256 equates to 64 processes × 4 threads. Running the MPP version with a parallelism of 256 equates to 256 processes.

A bar chart comparing the amount of message data in all processes between the Hybrid and MPP versions is shown in **Figure 1**. For a parallelism of 256, message data with the Hybrid version was about one-third that with the MPP version, and for a parallelism of 1024, message data with the Hybrid version was about one-half that with the MPP version. A bar chart comparing the number of times messages were sent in all processes between the Hybrid and MPP versions is shown in **Figure 2**. For both a parallelism of 256 and 1024, the number of messages sent with the Hybrid version was less than one-fifth that with the MPP version.[7]

We compared the computational performance of the Hybrid and MPP versions in terms of elapsed times when they were used

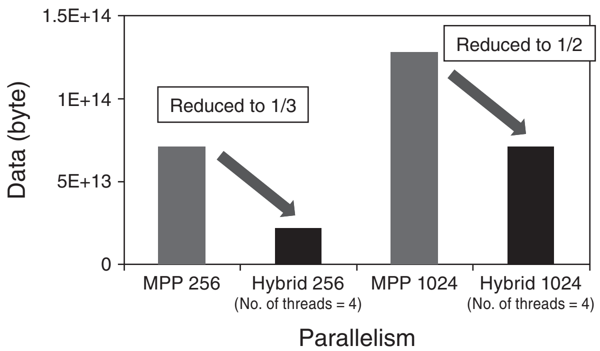


Figure 1
Amount of message data in two-million-element model.

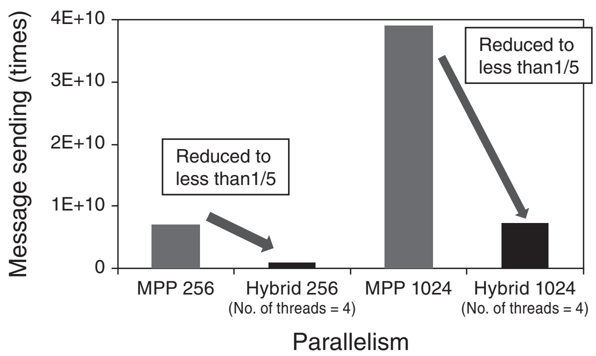


Figure 2
Message sending in two-million-element model.

to calculate phenomena having a physical time of 120 ms in the two-million-element car crash model. The simulations were run in the Information Technology Center at Nagoya University[8] with the environment configured as follows:

- Hardware: Fujitsu HX600 HPC server
- OS: Red Hat Enterprise Linux4
- Interconnects: InfiniBand DDR (2 GB/s) × 4
- Processor: Opteron 8380 (2.5 GHz) Quadcore
- No. of nodes (cores): 160 (2560)

As shown in **Figure 3**, the MPP version is faster for a parallelism of 128 and slightly faster for a parallelism of 256 and 512 while the Hybrid version is advantageous for a parallelism of 1024. In short, the Hybrid version is inferior in performance to the MPP version in the parallelism range of 128–512 but superior for highly parallel execution at a parallelism of 1024 and beyond.

Fujitsu is working jointly with LSTC to improve the Hybrid version of parallel LS-DYNA with the aim of achieving even faster processing by using massively parallel execution beyond a parallelism of 1000. This joint effort is using Fujitsu's Parallelnavi[9] software product to gather information for analyzing performance and to make improvements from the viewpoints of process parallelization and thread parallelization.

The work on process parallelization includes evaluating scalability, analyzing message size and MPI communication patterns, and studying domain decomposition processing (dividing the model into computational sub-domains corresponding to the parallel processes) applicable to massively parallel processing.

The work on thread parallelization includes classifying LS-DYNA computation into an element-calculation phase (which is costly), a contact-calculation phase, and a rigid-body calculation phase, embedding a measurement timer into costly subroutines, assessing the validity of measured values, and studying the feasibility of making improvements if room for improvement is found.

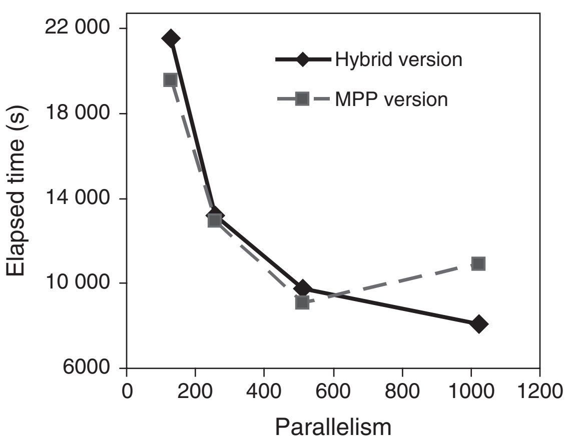


Figure 3
Performance comparison between Hybrid and MPP versions.

3.2 Achieving even higher speeds using Hybrid version with hardware barrier

Fujitsu's SPARC64 VII CPU on its FX1 high-end technical computing server and its SPARC64 VIIIfx CPU on its next-generation supercomputer (known as the K computer[note])[10),11)] are equipped with a hardware barrier function for achieving high-speed thread synchronization among cores within a CPU. In conjunction with Fujitsu's compiler technology, this hardware barrier enables the use of the Virtual Single Processor by Integrated Multi-core Parallel Architecture (VISIMPACT) facility for decreasing overhead in thread-based parallel processing among cores in hybrid parallel programming. The hardware barrier can perform high-speed inter-thread synchronization processing within a fixed amount of time regardless of the number of cores, and, as a result, can perform inter-thread synchronization processing about ten times

note) "K computer" is the English name that RIKEN has been using for the supercomputer of this project since July 2010. "K" comes from the Japanese word "Kei," which means ten peta or 10 to the 16th power.

faster than software-based synchronization processing using memory-data lock processing.

Joint research on ways to increase processing speed in massively parallel LS-DYNA computing was carried out with the Information Technology Center at Nagoya University. For this study, we used a practical ten-million-element model to compare performance among different LS-DYNA versions under a high degree of parallelism. While the ideal approach here would be to make calculations and compare performance for a physical time of 120 ms during which impact-induced deformation occurs, the large number of measurement patterns and the long measurement time for each compelled us to perform calculations and compare performance for a physical time of 10 ms. The simulations were again run in the Information Technology Center at Nagoya University[8] with the environment configured as follows:

- Hardware: Fujitsu FX1 technical computing server[12]
- OS: Open Solaris
- Interconnects: InfiniBand DDR (2 GB/s) × 1
- Processor: SPARC64 VII (2.5 GHz) Quadcore
- No. of nodes (cores): 768 (3072)

The LS-DYNA versions compared were the MPP version, the Hybrid version without VISIMPACT, and the Hybrid version with VISIMPACT. As shown in **Figure 4**, the MPP version exhibited behavior similar to that shown in Figure 3: as the degree of parallelism increases, performance rises, peaks at a parallelism of 1024, and then drops. A comparison of the Hybrid version without VISIMPACT with that with VISIMPACT showed found that performance was about the same for a parallelism of 512, but that performance with VISIMPACT was about 20%, 30%, and 40% better than that without VISIMPACT for a parallelism of 1024, 1536, and 2048, respectively. Given the slope of these plots for the Hybrid version, we can expect performance to continue to improve with a further increase in parallelism when using VISIMPACT.

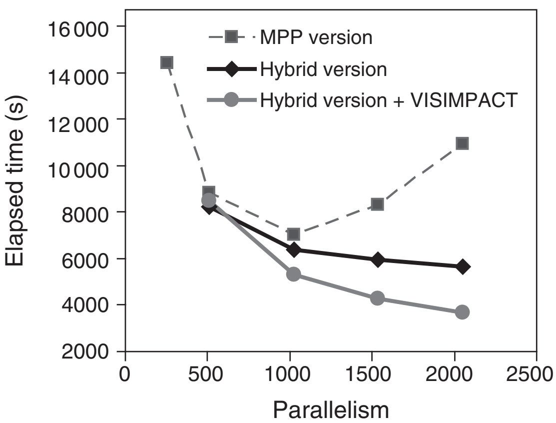


Figure 4
Elapsed time in ten-million-element model (10 ms).

This study tested the effect of using VISIMPACT when the simulation was run on the FX1 server. The FX1 features four cores per CPU, meaning four parallel threads per process. The K computer, on the other hand, features eight cores per CPU, which will make it possible to run eight parallel threads per process. This, in turn, will make it possible to perform parallel computing with a smaller number of processes, which should lead to even greater improvements in performance when using VISIMPACT.

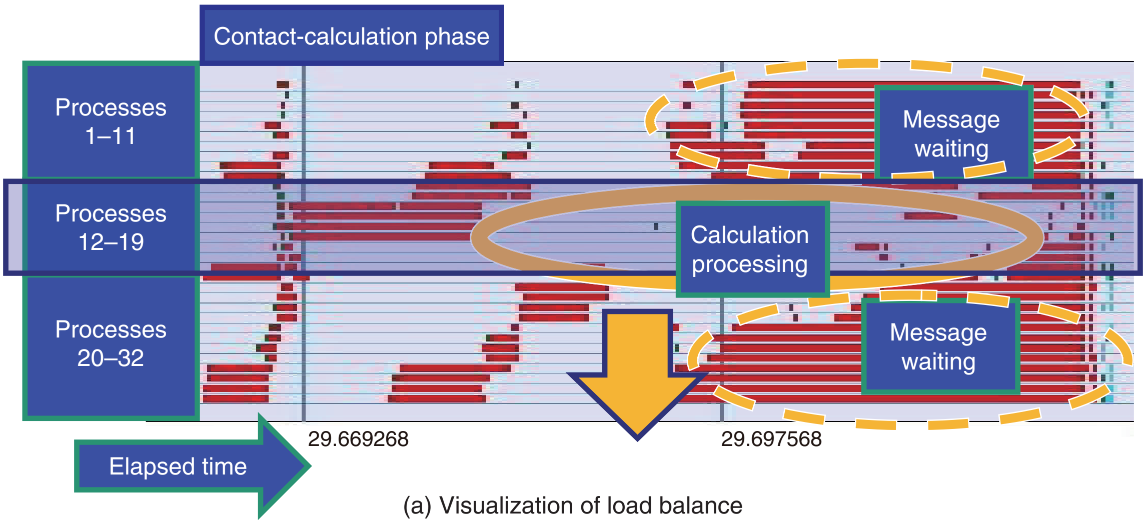
3.3 Improving computational load balance through visualization

Process-parallelization in LS-DYNA is done by dividing the total computational domain in the model into sub-domains in accordance with the number of parallel processes as a form of preprocessing (domain decomposition). For example, if we were to use 32 parallel processes to perform the calculations needed to analyze a crash between two vehicles, we would divide the total computational domain (two vehicles) into 32 sub-domains so that each process performs calculations for a different sub-domain. Accordingly, balancing the load among these processes can shorten the wait time between processes and speed up overall processing.

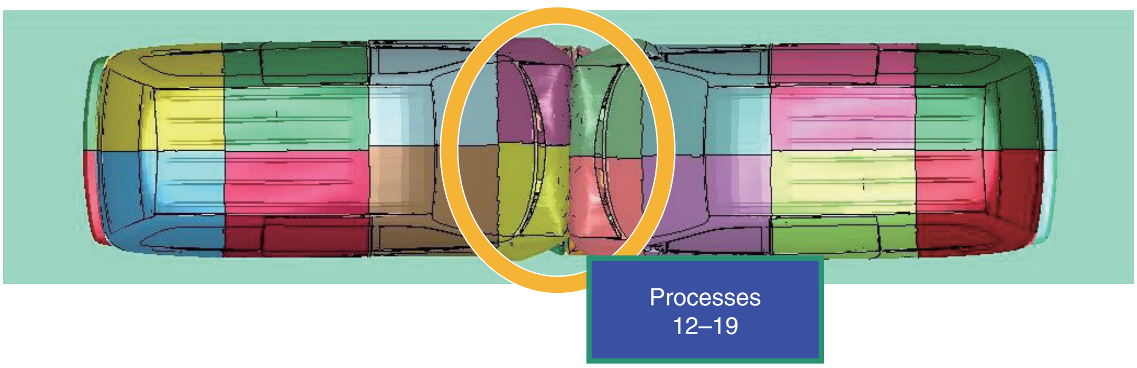
Load balance among processes can be checked using the visualization function in the Parallelnavi middleware for HPC. We describe an example of using this function to increase processing speed in the execution of a large-scale car crash model with 2 million elements on 32 processors.

The division of the total computational domain into sub-domains makes the number of elements to be processed by each process uniform, but there is no way to take into account the load related to contact calculations when this is done since object deformation cannot be predicted at the preprocessing stage. Consequently, if contact calculations should concentrate in certain processes, the computational load in those processes will be higher, degrading the load balance. To balance out the load caused by contact calculations, the user must devise an appropriate measure when creating the analysis model.

The results of visualizing the load balance in contact calculations by using the Parallelnavi visualization function at the 60-ms point are shown in **Figure 5 (a)**. Process numbers are shown along the vertical axis and elapsed time along the horizontal axis. Elapsed time in the figure represents the time from tool startup, which differs from the physical time of simulation. The horizontal lines represent



(a) Visualization of load balance



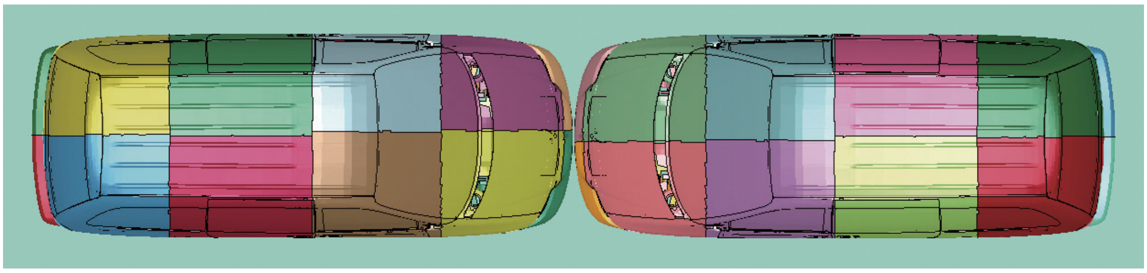
(b) Visualization of analysis results

Figure 5
Visualization of car crash model (60 ms).

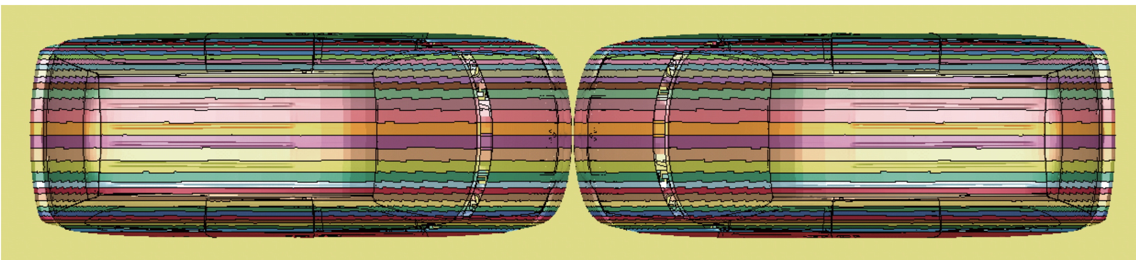
the calculation-processing, communication-processing, and message-waiting conditions for the corresponding processes. In this example, the ratio of communication processing is small, and message waiting in the figure includes message waiting and communication processing. The circled section in the block for processes 12–19 corresponds mostly to calculation processing while nearly the same time zone for processes 1–11 and 20–32 contains much message waiting. Thus, with one glance, it can be seen that the load for processes 12–19 is high and that the load for processes 1–11 and 20–32 is low, indicating poor load balance. Here, the time required for communication processing is extremely short compared with that for calculation processing and message waiting.

The state of impact between the two vehicles 60 ms after impact as obtained by LS-DYNA post-processing for the same physical time as that represented in Figure 5 (a) is shown in **Figure 5 (b)**. Each process is responsible for one of the quadrilateral sub-domains shown that divide up the vehicles from top to bottom and left to right. The circled section corresponds to processing by processes 12–19, which have already been shown to have a high load. At this location, deformation caused by the crash is progressing and the calculation load for determining element deformation and the extent of contact between the vehicles, between the vehicles' bodies and parts, and between parts is high. Figures 5 (a) and (b) can be used in combination to uncover processes with high load.

Reexamining how the model is decomposed into sub-domains can be effective in improving load balance. Specifically, to distribute the computational load in processes 12–19 and speed up processing in this example, domain decomposition can be performed in the direction of car motion. This has the effect of balancing out the amount of contact calculations within the engine compartment, in which complex components are concentrated. Default domain decomposition and domain decomposition in the direction of car motion are shown in **Figures 6 (a)** and **(b)**, respectively. Domain decomposition in (b) divides the total domain into 32 sub-domains



(a) Default decomposition



(b) Decomposition in direction of car motion

Figure 6
Comparison of domain decomposition in contact calculations for 32 processors.

in the direction of car motion. Each process here is responsible for a long and slender rectangular area. Domain decomposition in the direction of car motion speeds up computation compared to default decomposition about 1.2 times in the contact-calculation phase and about 1.1 times in elapsed time. Even a 10% improvement in elapsed time can have a significant effect when performing car crash analyses in the development of several new automobile models.

In general, performance-improvement effects depend largely on the scale of the model, the definition of the model, the load-balance conditions with domain decomposition, the number of parallel processors, etc. In this paper, we used a small-scale model as an example for the sake of understanding, but the same effects or better as those obtained here can be expected with a larger, practical model.

4. Future issues

As described above, the current Hybrid version of parallel LS-DYNA is not always faster than the MPP version, but it can have an effect on processing speed as the scale of computation increases. The following issues will be addressed to expand the use of the Hybrid version.

1) Increasing Hybrid speed for a parallelism of 128

At present, the MPP version is faster at a scale of 128 parallel processes, so a detailed analysis is needed to determine the reasons for this and to make improvements so that the Hybrid version can overtake the MPP version at this degree of parallelism.

2) Testing for massively parallel processing on a scale of 10 000 or higher

We are working to increase the speed of car crash analysis by LS-DYNA for the K computer with the aim of achieving relatively quick analysis at an ultra-high degree of parallelism of 10 000 or more for which almost no test results have been reported. For such massively parallel processing, communication processing can have a major effect even if using the Hybrid version, so, in addition to improving the LS-DYNA package, we will also study measures to reduce the amount of communication that can be taken when creating the analysis model.

5. Conclusion

In this paper, we introduced Fujitsu's efforts in increasing the speed of LS-DYNA processing, focusing on the use of the Hybrid version for parallel processing. Faced with an increasing level of detail in models used for analyzing car crashes, we showed that the Hybrid version is effective for speeding up analysis when using large-scale models. We also described how using a hardware barrier and improving load balance though visualization of computational load can also contribute to higher LS-DYNA processing speeds.

We would like to extend our deep appreciation to all concerned at the Information Technology Center, Nagoya University, for their invaluable support and cooperation in carrying out the measurements on the FX1 system.

References

1) Livermore Software Technology Corporation: LS-DYNA—Nonlinear Transient Dynamic Finite Element Analysis Software.
http://www.lstc.com/products/ls-dyna

2) Y. Takahira et al.: Development of ECE Side Impact MDB Modeling using Shell Elements. 6th German LS-DYNA Forum'07, B-III, 2007.
http://www.dynamore.de/en/training/conferences/past/07-forum/crash/development-of-ece-side-impact-mdb-model-using.

3) J. Hallquist: Recent Developments in LS-DYNA. 7th European LS-DYNA Conference, 2009.
http://www.dynalook.com/european-conf-2009/A-I-01.pdf

4) Toyota Motor Corporation: TMC Develops Advanced Version of Its Virtual Human Model.
http://www2.toyota.co.jp/en/news/10/05/0527.html

5) K. Kondo et al.: Crash Simulation of Large-Number-Of-Elements Car Model by LS-DYNA on Highly Parallel Computers. *Fujitsu Sci. Tech. J.*, Vol. 44, No. 4, pp. 467–474 (2008).
http://www.fujitsu.com/downloads/MAG/vol44-4/paper09.pdf

6) Top Crunch: 2-million Element Vehicle Collision Model.
http://www.topcrunch.org/

7) K. Kondo: The Performance of Car Crash Simulation by LS-DYNA Hybrid Parallel Version on Fujitsu FX1. 11th International LS-DYNA Users Conference 2010, I8-41-46, June 2006. *http://www.dynalook.com/international-conf-2010/Automotive-3-5.pdf*
8) Information Technology Center, Nagoya University. *http://www2.itc.nagoya-u.ac.jp/*
9) M. Tanaka et al.: Parallelnavi: A Development and Job Execution Environment for Parallel Programs on PRIMEPOWER. (in Japanese). *FUJITSU*, Vol. 52, No. 1, pp. 94–99 (2001). *http://img.jp.fujitsu.com/downloads/jp/jmag/vol52-1/paper20.pdf*
10) Fujitsu: K computer. *http://www.fujitsu.com/global/about/tech/k/whatis/processor/*
11) Fujitsu: CPU SPARC64 VIIIfx for Supercomputers. *http://www.fujitsu.com/global/about/tech/k/whatis/processor/*
12) Fujitsu: FX1: HPC High-end Technical Computing Server. *http://www.fujitsu.com/downloads/PR/2008/20080219-01a.pdf*

Kenshiro Kondo
Fujitsu Ltd.
Mr. Kondo is engaged in LS-DYNA technology support.