LS-DYNA® Performance Enhancement of Fan Blade Off Simulation on Cray XC40

Ting-Ting Zhu, *Cray Inc.*
Jason Wang, *LSTC*
Brian Wainscott, *LSTC*

Abstract

This work uses LS-DYNA to enhance the performance of engine FEA simulation of fan blade off containment test on Cray XC40 supercomputers. Blade off containment test is a specific form of air safety test required by the Federal Aviation Administration and other safety agencies, which involves the intentional release of a fan blade when engine is running at full power. The released blade must be contained within the fan cases during the impact and imbalanced rotation. The simulation of fan blade off containment test is technically challenging and computationally intensive. To enhance the performance of fan blade off containment simulation, some improvements are made in surface to surface erosion contact in LS-DYNA version R8.0.0. In this study, the performance of fan blade off simulation using LS-DYNA version R8.0.0 is compared with that using LS-DYNA version R7.1.2 on a Cray XC40 supercomputer. In addition, the MPI communication patterns and load balance among the MPI processes in the fan blade off containment simulation is also analyzed. Finally, a very large model of fan blade off simulation with more than 80 million elements, which used to be a daunting task that took a modern MPP computer more than a month to complete, is carried out using LS-DYNA version R8.0.0 on the Cray XC40 computer.

1 Introduction

The failure of a fan or a compressor blade in a jet engine at high rotating speeds, due to fatigue, blade detachment, overheating, bird strike, material defect, etc., is a major hazard. The failed rotating fan blade can release high-energy fragments that could perforate the engine case and damage fuel tanks, and lead to plane crash. In order to ensure the safety of passengers and crews, Federal Aviation Administration and other safety agencies require that all blade fragments should be contained within the engine cases [2]. Thus fan blade off containment study becomes very important research topic in aerospace industry [3] [4].

In this study, we use explicit LS-DYNA [1], a commercially available finite element program, to simulate the fan blade off containment on a Cray XC40 supercomputer. This type of simulation is very computationally intensive due to the complex nature of the analysis. During our initial study, we found that a large fan blade off containment simulation with problem size of 80.6 million elements took more than a month when LS-DYNA R7.1.2 was used. The ideal time-to-solution for this type of simulation is within one day. In order to reduce the simulation time to less than a day, some enhancements was made in surface to surface erosion contact in LS-DYNA R8.0.0. In this paper, we first summarize the enhancements made in surface to surface erosion contact. Then we use a medium size model to conduct performance comparison of fan blade off containment simulation between LS-DYNA versions R8.0.0 and R7.1.2. In addition to that, the MPI communication pattern and load balance among the MPI processes are also analyzed. Finally a large model of fan blade off containment simulation with 80.6 million elements is carried out using LS-DYNA R8.0.0.

2 LS-DYNA Code Optimization for Fan Blade off Containment Simulation

The contacts between the failed blade and engine case as well as between the failed blade and others blades are simulated using LS-DYNA surface-to-surface erosion contact. The surface-to-surface erosion contact is the most computationally expensive part in fan blade off simulation. Thus the key to enhance the fan blade off containment simulation is to improve the performance of surface-to-surface erosion contact in LS-DYNA.

To optimize surface-to-surface erosion contact, Cray performance profiling tool called CrayPat was used to identify the most time consuming subroutines in LS-DYNA. Then these routines were optimized and tested on Cray XC40. The final enhanced code was implemented in LS-DYNA R8.0.0 and newer versions. The first improvement made in erosion contact is the 30% reduction of memory required for storing erosion contact surfaces. The second improvement is made in erosion contact initialization through removing some redundant erosion calculations. The third improvement is made in exterior surface calculation. Previously, the new exterior surface was fully recomputed, which was very time consuming and a major bottleneck for parallel scaling. The new method keeps most of the existing surface data untouched, and only modifies the portion where elements erode. This significantly speeds up the computation of surface-to-surface erosion contact; as a result, it improves the overall code performance of fan blade off simulation by many folds.

In the next session, we will demonstrate the performance enhancement of fan blade off simulation of LS-DYNA version R8.0.0 on Cray XC40.

3 Performance Enhancement of Fan Blade off Simulation Using LS-DYNA R8.0.0 on Cray XC40

For quick turnaround time, a medium size model of fan blade off containment simulation is first used in this study. To demonstrate the performance enhancement of fan blade off simulation, which is a result of the optimization of surface-to-surface erosion contact, we compare the performance of the medium size model between LS-DYNA versions R8.0.0 and R7.1.2 on a Cray XC40 computer. Additionally, to further understand the MPI performance of LS-DYNA R8.0.0, Cray MPI profiling tool called Profiler is used to study the MPI communication patterns and load balance.

3.1 Medium Size Model of Fan Blade off Containment Simulation

The medium size model of fan blade off containment simulation consists of 26.5 million nodal points, and 24 million solid elements. In this model, an engine fan is represented by three fan blades rotating in a stationary engine case, as shown in Figure 1. Initially, the fan blades rotates at a constant speed of 5,800 revolution per minute (RPM) for 1 millisecond, which is so called dynamic relaxation phase. During this phase, the initial stress state of the fan blades are computed by LS-DYNA using the constant angular velocity, as shown in Figure 2. After dynamic relaxation phase, the red rotating blade is intentionally released to the engine case for 1 millisecond to simulate the fan blade off containment test, which is so called transient phase. During the transient phase, due to centrifugal force (or its initial stress), the released blade collides with the engine case and the other two blades. The collisions between the released blade and engine case as well as between the released blade and the other two rotating blades are modeled using surface-to-surface erosion contact.

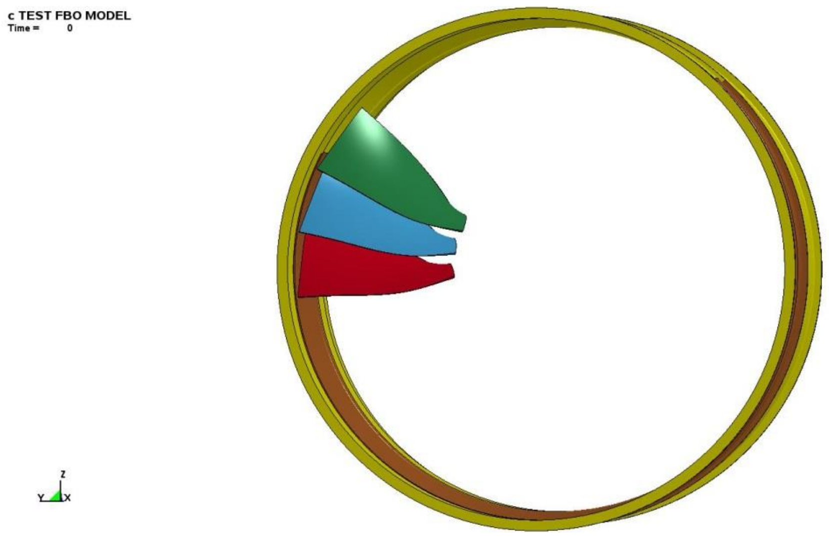

Figure 1 Medium size model of fan blade off simulation with three blades

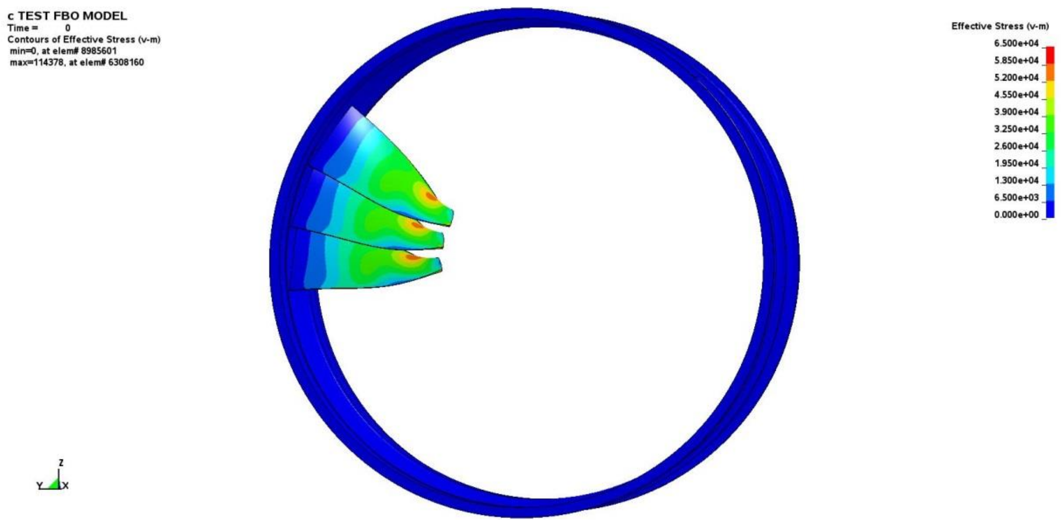

Figure 2 Explicit LS-DYNA uses a constant rotation speed to set the initial stress state of the blades prior to the fan blade off simulation

3.2 CRAY XC40 Software and Hardware Configurations

The following Cray XC40 hardware and software configurations are used in this simulation:

Table 1: Summary of hardware and software configurations

System type	Cray XC40
Processor architecture	2.3 GHZ Intel Haswell
sockets/node	2 sockets per node
Number of cores per socket	16 cores per socket or 32 core per node
Network	Aries Interconnect with Dragonfly Topology
Memory size per node	128 GB/node DDR4-2400
OS	CRAY CLE5.2
LS-DYNA versions	LS-DYNA R8.0.0 and R7.1.2
Compiler version	Intel/14.0.4.211
MPI version	CRAY-MPICH/7.3.2

3.3 Performance comparison of Fan Blade Simulation between Versions R8.0.0 and R7.1.2

In this test case, the performance of LS-DYNA versions R8.0.0 and R7.1.2, before erosion contact starts, is very similar. Since we are only interested in comparing the performance differences after the released fan starts colliding with the engine case and other two blades, we break the simulation into two parts, non-erosion part and erosion part. The non-erosion part is the portion of simulation that does not involve erosion contact computation, which starts at the beginning of dynamic relaxation phase and ends at about 0.2ms into transient phase. The erosion part is the portion of simulation that involves erosion contact computation, which starts at about 0.2ms into transient phase. For comparison purpose, the erosion portion of simulation is only computed during transient phase of 0.2ms to 0.35ms.

Figure 3 shows the performance comparison of fan blade off simulation of the medium size model between versions R8.0.0 and R7.1.2. . At 256 cores, the total elapsed time of fan blade off simulation using R8.0.0 is about 1.9 times faster than that using version R7.1.2. This speedup increases as core count increases. At 2048 cores, version R8.0.0 is about 2.7 times faster than version R7.1.2.

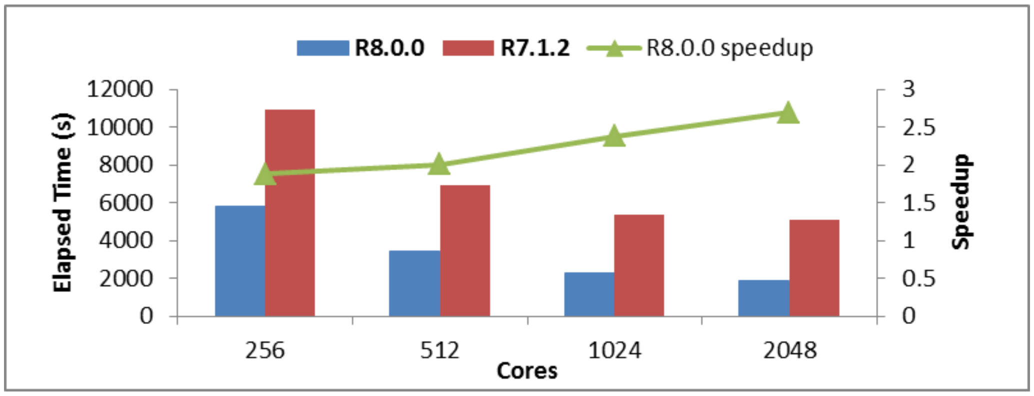

Figure 3 Total elapsed time comparison of fan blade off simulation between LS-DYNA R8.0.0 and R7.1.2 for the medium size model during transient phase of 0.2ms to 0.35ms on CRAY XC40

To find out where the speedup is coming from, in the next paragraph, we will break down the total elapsed time based on functionalities into “element processing”, “contact”, and “other miscellaneous”, and then compare each of them between versions R8.0.0 and R7.1.2.

3.4 Performance Comparison of Element Processing, Contact and Miscellaneous
Upon termination, LS-DYNA provides detailed timing information of several major functions at the end of its log file. For fan blade off simulation, the functions of “element processing”, ”contact algorithm”, and “other miscellaneous” consume more than 95% of total wall clock time. Now let’s compare the performance difference for each of these functions between versions R8.0.0 and R7.1.2.

The elapsed time of “element processing” shown in Figure 4 is almost inversely proportional to core counts in both versions R8.0.0 and R7.1.2, in other words, the parallel scaling of “element processing” is nearly ideal. More importantly, the performance differences of “element processing” between these two versions are negligible.

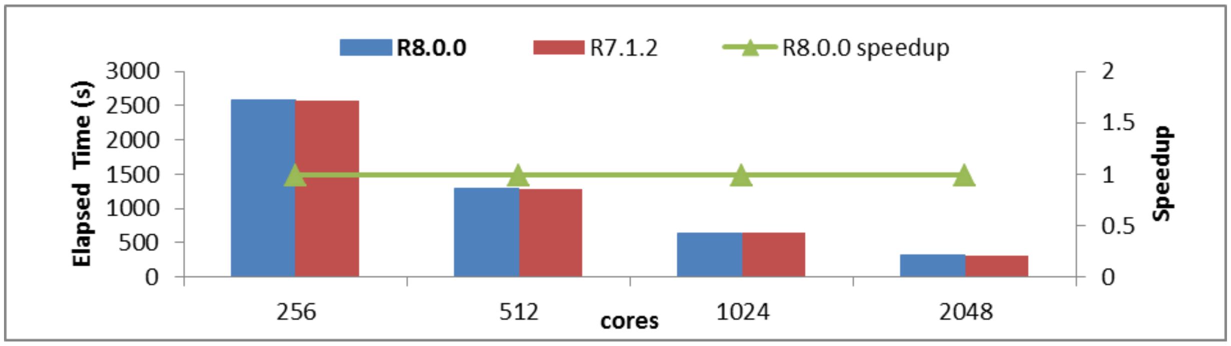

Figure 4 Element processing comparison between R8.0.0 and R7.1.2 in the medium size model during transient phase of 0.2ms – 0.35ms

Figure 5 shows that “contact” of R8.0.0 is about 3.2 to 3.5 times faster than that of R7.1.2. Additionally, it also shows that “contact” of both versions scales reasonably well for core counts from 256 to 512, but the scaling levels off after 1024 cores. This means that “contact” scalability is one of the limiting factors for the parallelization of fan blade off simulation of the medium size model at core counts above
1024.

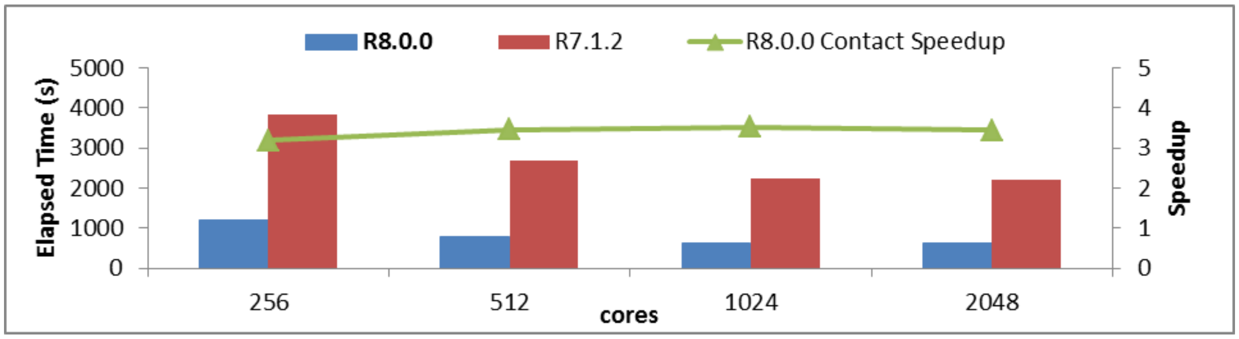

Figure 5 Contact comparison between R8.0.0 and R7.1.2 in the medium size model during transient phase of 0.2ms – 0.35ms

Figure 6 shows that the elapsed time of “other miscalleneous” decreases as core count increases. Besides it also shows that the elapsed time of “miscallaneous” using R8.0.0 is about 2.4 times to

3.3 times faster than that using R7.1.2. Additionlly, parallel performance of "miscallaneous" scales linearly for core counts from 256 to 512; however, the scaling starts to reverse after 1024 cores. That means that "miscallaneous" computation is another limiting factor for the parallel scalability of fan blade off simulation of this medium size model at higher core counts.

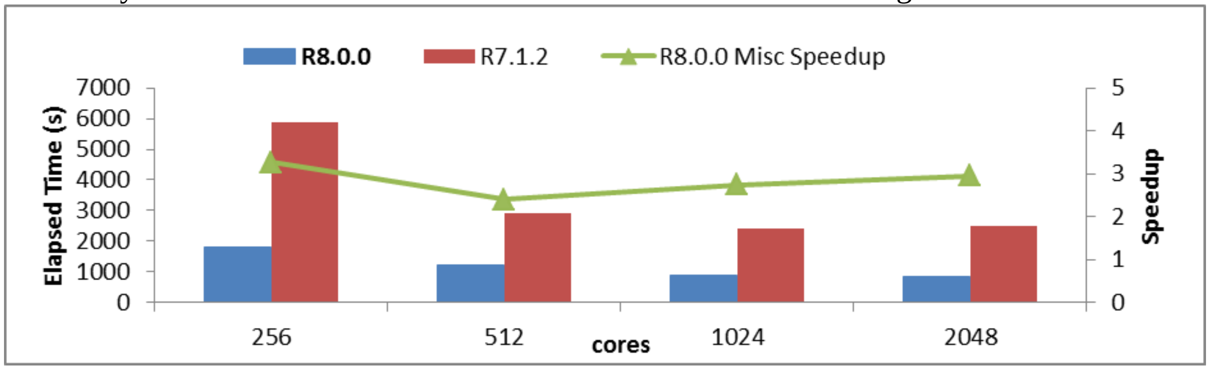

Figure 6 Miscellaneous time comparison between R8.0.0 and R7.1.2 in the medium size model during transient phase of 0.2ms – 0.35ms

By breaking down the LS-DYNA wall time based on its functionality, one can clearly see that the enhancement of fan blade off simulation in LS-DYNA R8.0.0 is coming from "contact" and "miscellaneous", which is the direct outcome of surface-to-surface erosion contact optimizations described in section 2.

In order to understand the parallelism of LS-DYNA fan blade off simulation, in the next section, we use Cray MPI profiling tool called "profiler" to reveal how much time is spent on MPI communication and which MPI calls are dominating the MPI time.

3.4 MPI Communication patterns in Fan Blade Off Simulation

Cray "profiler" is a library which uses wrapper routines to intercept calls to the MPI library. The wrapper routines forward the calls to the MPI library and also collect timing information of MPI calls and other statistics. A summary report of statistics is printed at the end of run when MPI_Finalize is called. The reported statistics includes MPI time, MPI communication time, and MPI synchronization time for the entire run and for each MPI call. Here the MPI communication time is the time that a process spends on communicating with other processes, which decreases with the interconnect speed. The MPI synchronization time is the time that a process spends on synchronizing with other processes, which increases with load imbalance in a program. The MPI time is the sum of the MPI communication time and synchronization time. However, for point to point communication calls, such as MPI_send and MPI_recv, the MPI synchronization time and communication time cannot be measured separately. So keep in mind that the measured MPI communication time in point to point MPI communication calls is actually the combined time of MPI synchronization and communication.

Figure 7 shows the MPI communication time, the MPI synchronization time, and the total MPI time in the fan blade off simulation of the medium size model during transient phase of 0.2ms to 0.35ms using LS-DYNA R8.0.0. The total MPI time decreases as core count increases at lower core counts of 256 to 1024, however, the trend reverses at higher core counts of 1024 to 2048. In addition, the MPI communication time is about 16% of the total MPI time at 256 cores and it gradually decreases to 6.5% at 2048 cores. In other words, the MPI synchronization time is 84% of the total MPI time at 256 cores and it increases to 93.5% at 2048 cores. This means that MPI time is dominated by MPI synchronization time in the fan blade off simulation, which is a strong indication of load imbalance.

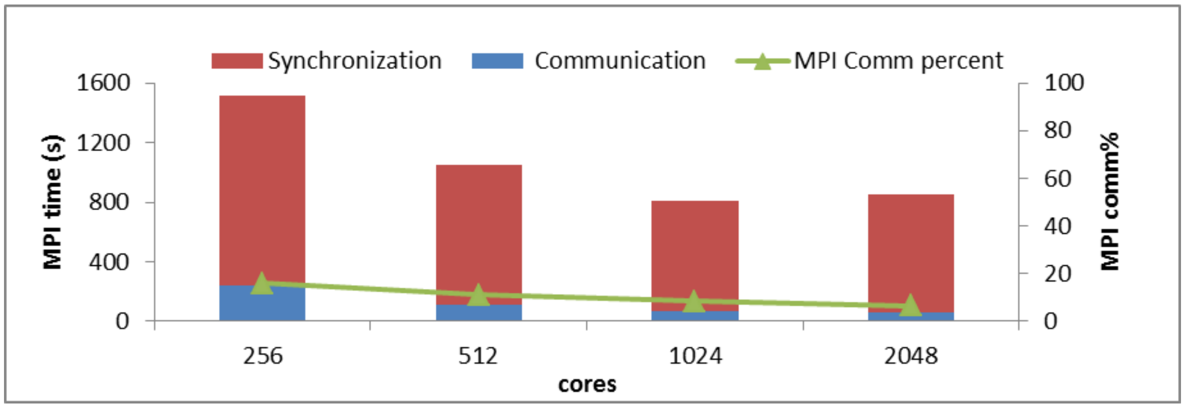

Figure 7 MPI time in the fan blade off simulation of the medium size model during transient phase of 0.2ms-0.35ms using LS-DYNA R8.0.0

Figure 8 shows the most time consuming MPI calls in the fan blade off simulation of the medium size mode during transient phase of 0.2ms to 0.35ms. Figure 7 indicates that the most MPI time is spent on MPI synchronization; and Figure 8 further reveals that most MPI synchronization time is spent on MPI_allgather , MPI_barrier, and MPI_alltoall. Please note that since most of MPI_allgather and MPI_alltoall time is spent on synchronization, we only plot the MPI_allgather and MPI_alltoall synchronization time on Figure 8. The MPI_allgather synchronization, the most costly one, decreases as core count increases from 256 to 1024; however, the tendency reverses from 1024 to 2048 cores. Additionally, the second most expensive MPI call is MPI_barrier. These MPI communication patterns clearly indicate that load imbalance is the bottleneck to the parallelization of the fan blade off simulation at high core counts.

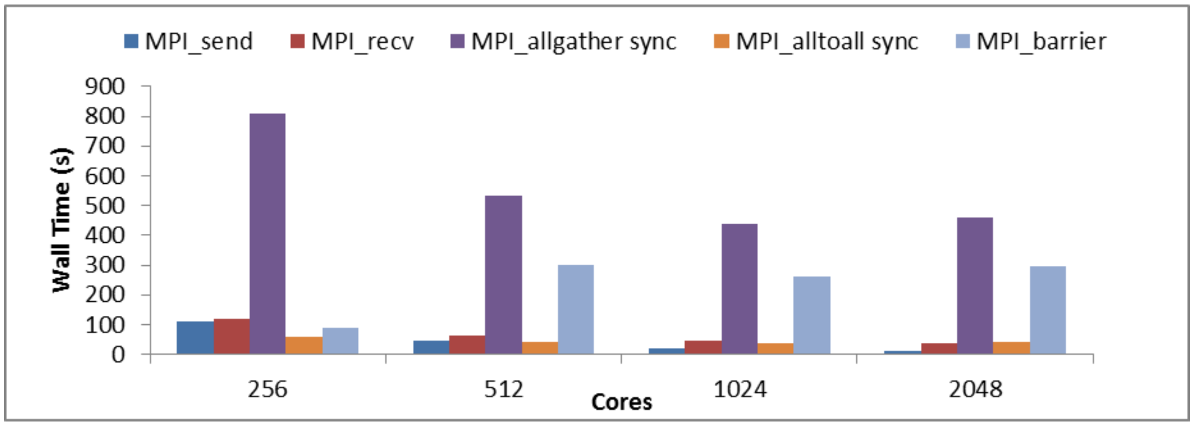

Figure 8 MPI communication patterns in the fan blade off simulation of the medium size model during transient phase of0.2ms – 0.35ms using LS-DYNA R8.0.0

Because of load imbalance, the parallel scaling of fan blade off simulation of the medium size model is limited to 1024 cores as shown in Figure 3. However, for a large size model of 80.6 million elements, fan blade off simulation can scale to 16 thousand cores, which will be demonstrated in the following section.

4. Performance of Large Fan Blade off Simulation

The large size model of fan blade off simulation is composed of 82 million nodal points, and 80.6 million solid elements. It is the refinement of the medium size model described in section 3.1. With version R7.1.2, this large model used to take more than a month to simulate 1 millisecond of transient phase using 16,384 cores on Cray XC40. Now with version R8.0.0, on the same system using the same number of cores, it only takes 21 hours, a speedup of 34 times. Figure 9 shows the parallel performance of the large model of fan blade off simulation during transient phase of 0.2ms to 0.3ms on Cray XC40. For performance purpose, hybrid LS-DYNA with 4 OpenMP threads per MPI rank is selected for this simulation. Thus the number of cores used in the calculation is equal to the number of MPI ranks times the number of OpenMPI threads. As shown in the Figure 9, the elapsed time of the transient phase scales almost linearly from 2048 to 4096 cores and the scaling levels off gradually after 8192 cores. For further understanding the performance behavior, the computational time spent on element processing, contact and other miscellaneous functions is also displayed in Figure 9. As one can see, the time spent on element processing scales linearly from 2048 to 16384 cores. The time spent on contact and other miscellaneous functions decreases from 2048 to 4096 cores, but increases from 4096 to 16384 cores. Therefore, contact and miscellaneous functions are the limiting factors that prevent this large model from scaling beyond 16,384 cores.

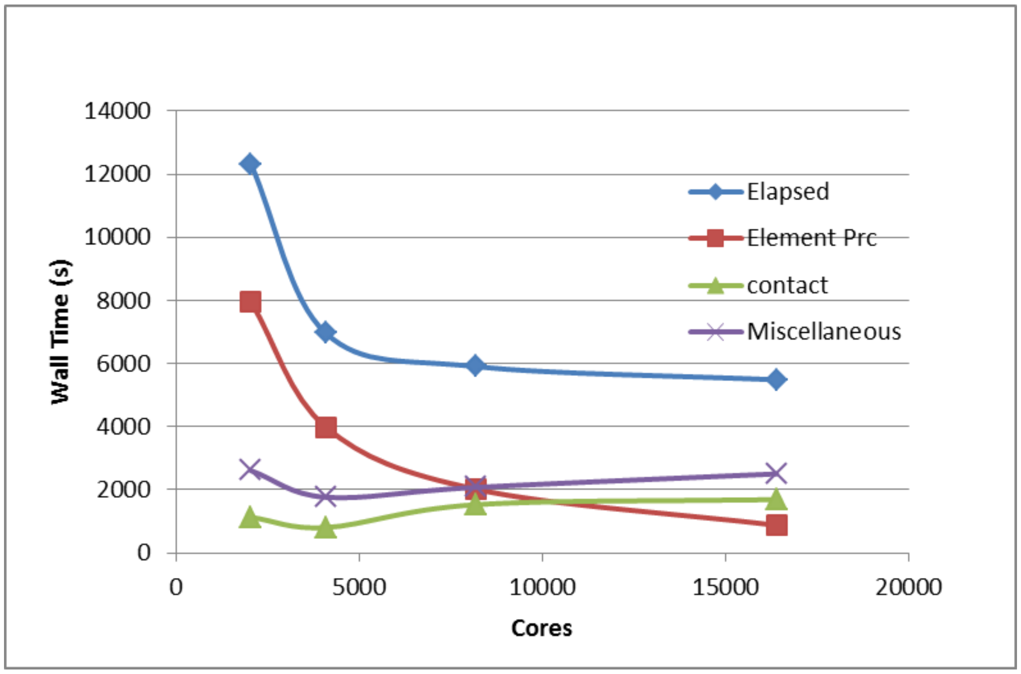

Figure 9 Parallel scaling of the large fan blade off model during transient phase of 0.2ms – 0.3ms on Cray XC40

5. Conclusion

The performance enhancement of fan blade off simulation has been implemented in LS-DYNA version R8.0.0 and newer versions. The performance comparisons of fan blade off simulation between LS-DYNA versions R7.1.2 and R8.0.0 are conducted on Cray XC40. In a medium size model of 24 million elements, LS-DYNA R8.0.0 is 1.9 to 2.7 times faster than R7.1.2. In a large size model of 80.6 million elements, L-DYNA R8.0.0 is 34 times faster than R7.1.2. Additionally, the MPI communication patterns of the fan blade off simulation of the medium size model are analyzed. It is found that most MPI time is spent on MPI synchronization of MPI_Algather and MPI_barrier, which indicates that load imbalance is the bottleneck for the parallelization of the fan blade off simulation at higher core counts.

References

[1] LS-DYNA USER'S MANUAL, Version R8.0, Livermore Software Technology Corporation, March 23, 2015.
[2] Federal Aviation Administration. Federal Aviation Regulation Part 33, Section 33.94. 1984.
[3] K.S. Carney, J.M. Pereira, et al, "Jet Engine Fan Blade Containment Using an Alternate Geometry", in International Journal of Impact Engineering 36 (2009) 720–728.
[4]Jason Burkley Husband, "Developing an Efficient FEM Structural Simulation of a Fan Blade Off test in a Turbofan jet engine", PhD dissertation.
[5] F. Gálvez, D. A. Cendón, et al, "Materials Behavior And Numerical Simulation Of a Turbine Blade-off Containment Analysis", 4th International Conference on Structures Under Shock and Impact.