LS-DYNA® HYBRID Studies using the LS-DYNA® Aerospace Working Group Generic Fan Rig Model

Gunther Blankenhorn and Jason Wang
Livermore Software Technology Cooperation

Gilbert Queitzsch
Federal Aviation Administration

Cing-Dao Kan
Center for Collision Safety and Analysis, George Mason University

Kivanc Sengoz
National Crash Analysis Center, George Washington University

Thomas J. Vasko
Central Connecticut State University

Abstract

In addition to the well-known parallel versions of LS-DYNA, the symmetric multiprocessing (SMP) version and massively parallel processor (MPP) version, LSTC offers an LS-DYNA HYBRID version that combines these two parallel programming models into a single code. The development of LS-DYNA HYBRID, which started in 2011, is focused on obtaining high code performance on large cluster environments.

The intent of the current study is to investigate the LS-DYNA HYBRID performance, scalability, and output consistency using a modified LS-DYNA Aerospace Working Group Generic Fan Rig Model. The original model is an outcome of a Federal Aviation Administration (FAA) funded university project and it is used as a test case for the LS-DYNA Aerospace Working Group Test Case Suite (http://awg.lstc.com).

Introduction

Since the LS-DYNA code was first introduced in the early nineties, continuous improvements have been made to the code to meet the needs of engineers who are using the code for increasingly complex applications. As this complexity has increased from purely structural

problems to multi-physics problems and modeling details have raised the number of elements and solution variables, the performance of the code on multi-processor computers and computer clusters has become even more important. Calculation times have also risen with the use of more complex material models, multiple contacts, coupling between different modeling approaches, smaller element sizes, and more expensive element formulations. Contemporaneous to the increasing complexity of features and models, the increases in processor performance and the design of sophisticated clusters and networks have opened new possibilities for the solution of complex analysis models in a more acceptable timeframe. In order for software to take advantage of these increasingly common computational environments with multiple cores and compute clusters, parallel programming methods that enable the software to calculate portions of the model in parallel on several processors are required.

LS-DYNA offers two parallel programming models. SMP (Symmetric Multi-Processing), which originated from the serial code and uses OpenMP® [3] directives to split the threads thereby enabling them to be run on multiple cores; and MPP (Massively Parallel Processing), which uses a message passing protocol to exchange information between the cores on a board or over the network. The SMP parallel programming model runs on computers with multiple identical cores with the cores and memory connected via a bus that is shared between all cores. The MPP programming model first performs a decomposition of the problem (domain decomposition) and then distributes the sub-domains to different cores using MPI (Message Passing Interface) protocol for communications between the subdomains during calculation. These sub-domains use their own exclusive part of memory and exchange data that is needed on other cores via MPI protocol. The MPP model is, therefore, not restricted to hardware where all cores have access to the same memory. This is important when using compute clusters, where an arbitrary number of cores are gathered together and connected via a network. For the MPI protocol, messages can be passed between cores on one motherboard or over the network, which provides the opportunity to build clusters of single compute nodes connected via a network.

The LS-DYNA MPP version is scalable over a wide range of core counts and can reduce calculation wall clock times dramatically. While the MPI protocol connects all cores, increasing the number of cores rapidly increases the network load, which has a negative impact on the performance and scalability. The LS-DYNA HYBRID [4] version takes advantage of a combined SMP and MPP programming model to address this issue.

In this study a fan rig model, developed by the National Crash Analysis Center (NCAC) and funded by the Federal Aviation Administration (FAA) is used. The original model is part of the LS-DYNA Aerospace Working Group (AWG) Test Case Suite [1] and is a natural candidate for an investigation of an aerospace application with LS-DYNA HYBRID. Some modifications to this model were, however, made for the current study and they are described in a later section.

In the following sections, the merits and limits of the LS-DYNA HYBRID version are discussed and the fan rig model is described. In addition, the hardware and computational environmental setup used in the study are specified, and the performance, scalability, and output consistency are documented and conclusions, taken from these observations, are presented.

LS-DYNA HYBRID Version

In general, LS-DYNA HYBRID combines SMP and MPP parallel programming models to reduce wall clock time when using an increasing number of cores. Additionally, the LS-DYNA HYBRID version can be used (in a specific setup) to obtain consistent outputs for different core counts. “Output” is meant here to be the various result quantities obtained from binary or ASCII format result files, e.g., the value of specific node displacements for a node in a specific coordinate direction at a specific time or the values of global and part specific energies at a certain time.
In the following section, performance and scalability as well as output consistency are discussed for the LS-DYNA HYBRID version. This includes the discussion of basic characteristics of the LS-DYNA SMP and LS-DYNA MPP versions since the LS-DYNA HYBRID version is a combination of these two programming models.

Performance and Scalability

One of the main measures of performance is the elapsed wall clock time $T_{elapsed}$, which can be split up as follows:

for LS-DYNA SMP: $T_{elapsed} = T_{computation} + T_{IO} + T_{overhead}$

for LS-DYNA MPP: $T_{elapsed} = T_{computation} + T_{communication} + T_{IO}$

for LS-DYNA HYBRID: $T_{elapsed} = T_{computation} + T_{communication} + T_{IO} + T_{overhead}$

where $T_{computation}$ is the actually time the cores do operations on the problem and T_{IO} is the time needed to perform the Input / Output operations to the hard disk. $T_{overhead}$ is related to the OpenMP® thread overhead in the SMP programming model. This thread overhead is related to the time needed to manage and create threads in the SMP programming model that also includes managing the memory for these threads and delays due to mismatches in computational time between threads. $T_{communication}$ is the time which is required by the MPI protocol to exchange data between different cores.

If the number of cores is increased for an MPP application, the performance and scalability may not also increase as would be expected. In MPP applications, communication is done between all cores used for the calculation. If the core count increases, the time needed for the communication $T_{communication}$ increases. If the amount of data exchanged via MPI protocol hits the bandwidth of the network connections, communication speed decreases rapidly, and when the time used for communication becomes dominant, the analysis fails to scale.

To keep the core count constant, but reduce the MPI communication, the LS-DYNA HYBRID version combines SMP and MPP parallel programming models. In the LS-DYNA HYBRID programming model the MPP cores are replaced by an SMP process (the SMP process contains all SMP threads). The SMP calculations can use several cores, while the MPI protocol communicates only between the SMP processes, which reduce the amount of communication when compared to a pure MPP run.

Output Consistency

The drawback of having different parallel programming models is that some of the code subroutines for LS-DYNA keywords [2] such as the *CONTACT, *ALE, *AIRBAG, *BOUNDARY, *CONSTRAINED keywords are different, while others such as the *ELEMENT and *MAT keywords are, for the most part, the same code. This discrepancy leads to different computed output for LS-DYNA SMP, LS-DYNA MPP and LS-DYNA HYBRID versions compared to each other.

Additional differences can be seen for all three parallel programming methods when using different core counts. The order of summation of result vectors depends on the number of cores used to compute them. Due to round off errors, the result of this summation is order dependent, which shows up as different output for different core counts. Methods to avoid or suppress these differences in output are discussed for the three parallel programming models in the next paragraphs.

When running LS-DYNA SMP versions, there is a consistency flag available that enforces the order of the summation of certain result vectors, thereby maintaining output numerical consistency when using different core counts. There is, however, a time penalty of about 10-15% of the wall clock time for this LS-DYNA SMP consistency option.

When running LS-DYNA MPP versions, there are numerical variations due to round off errors during the summation of certain result vectors. Using double precision, a finer mesh, or avoiding instabilities in the model, may help reduce these numerical variations. Unfortunately, a consistency flag option for MPP would increase wall clock time. Certain MPI protocols sum up the result vectors on the cores that belong to one compute node first and then sum up all results from the compute nodes connected via the network. If the compute nodes have different core counts, this could end up with different results even for constant core counts. To avoid this effect, the “lstc_reduce” option in pfile [2] should be set.

The LS-DYNA HYBRID version inherits the merits and limits from both programming models, SMP and MPP that it was derived from. Using LS-DYNA HYBRID without the LS-DYNA SMP consistency flag would show differences in output when using different core counts for the SMP threads, as would different MPP processors counts show differences in the computed output. However, for the LS-DYNA HYBRID version, using the merits of the consistency flag for SMP threads and keeping the MPP processors constant, output consistency is maintained for varying SMP thread counts. This enables a consistent output if the number of MPP processors is kept constant and the SMP thread count varies.

LS-DYNA AWG Generic Fan Rig Model

The Generic Fan Rig Model was developed at the National Crash Analysis Center at George Washington University. The development was funded by the Federal Aviation Administration. The original model is available at the LS-DYNA Aerospace Working Group web site (http://awg.lstc.com) and is used as a Test Case in the Engine Related Impact and Failure Test Case Suite.

Physical Model Description

The generic fan rig model (see Figure 1) contains geometry for the major components of a fan rig, as well as generic material data describing the material behavior. The model includes the strut, mounts, turbine cases, shaft, blades, containment cases, and associated links. The fan diameter is 40 inches and the fan contains 20 wide chord blades (integrally bladed disk) made from Ti6-4. The solid wall containment case is made from AL-2024 and the hollow fan shaft with wall thickness of 0.2 inches is made from SS-304. The model contains three bearings. In the front of the shaft, a ball bearing reacts thrust and radial loads, while two roller bearings (see Figure 1) react radial loads. The model assesses initial containment and post containment interactions between the released blade, trailing blades, and the containment case.

Finite Element Model Description

The containment case and the fan blades are discretized with solid elements. The material model used for the Ti6-4 fan blades is *MAT_JOHNSON_COOK, while the AL-2024 solid wall containment case uses *MAT_TABULATED_JOHNSON_COOK. The disk hub is assumed rigid and the shaft is discretized with shell elements using *MAT_ELASTIC. The shaft is connected to the structure via *CONSTAINED_JOINT_SPHERICAL and *CONSTRAINED_JOINT_CYLINDRICAL with two rigid rings attached to each of the joints (see Figure 1). The surrounding structure and the mounts are mostly discretized with shell elements. Details on material properties and modelling approach can be found at http://awg.lstc.com.

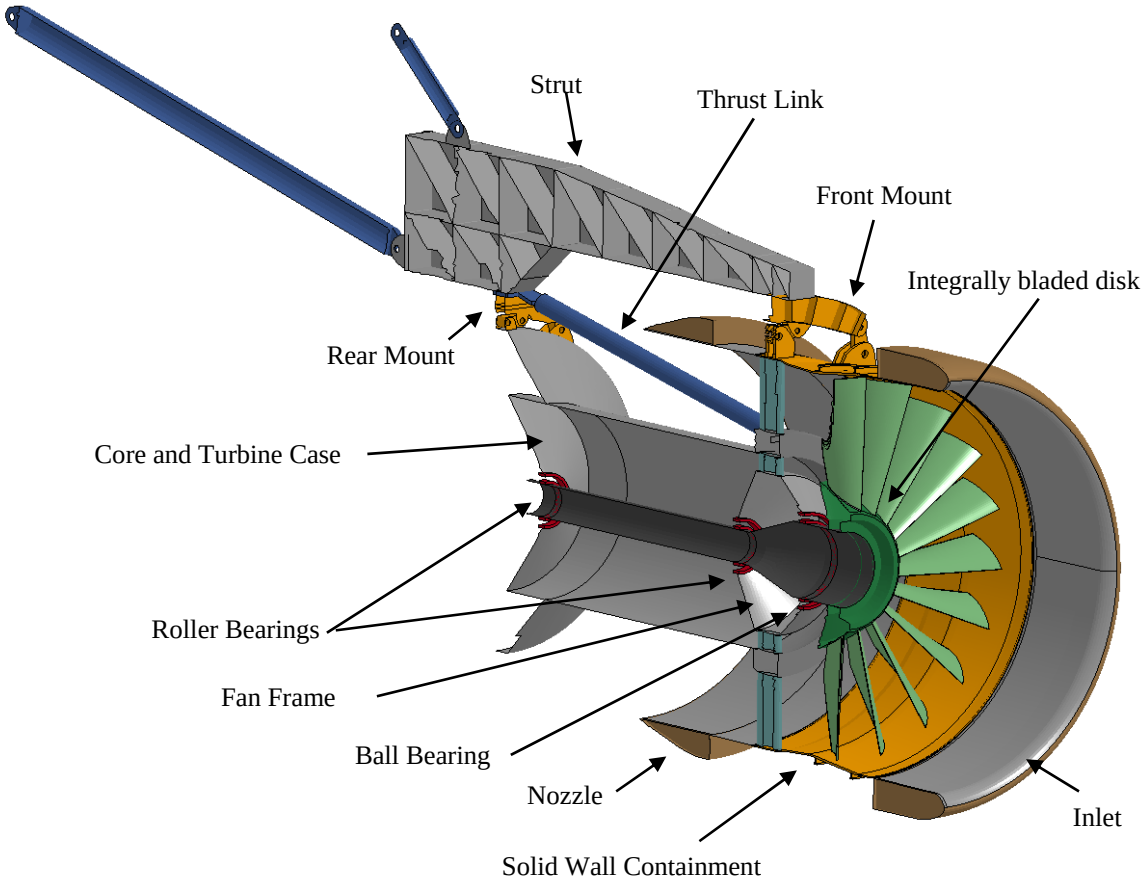

Figure 1: Fan Rig Model Components

The rotating blades and the shaft are initially pre-stressed. The simulation time is set to 0.08 seconds, which is approximately the time for one revolution of the shaft and the blades.
One fan blade is released at the start of the analysis and the interaction between the released and trailing fan blades along with containment is investigated.

Study Setup

This study is focused on performance, scalability, and output consistency. Studies of network connection types or different hardware or MPI protocols were not performed. Factors which influence output consistency are consistency flags in the SMP version and a fixed processor count in the MPP version. Factors which influence scalability and performance are the decomposition of the model, the performance of a single core, the communication characteristics of the network connection, file systems, MPI product and versions, memory and cache system, and the numerical model.
In the following sub-sections, the core counts used, the hardware and compute environments, and the decomposition and the command line are specified.

LS-DYNA HYBRID and LS-DYNA MPP core counts

This study is performed with 12 MPP processors and 1/2/3/6/12 SMP threads (see Table 1). This reflects a model setup which runs on a single node on the cluster, 12 MPP processors and 1 SMP thread, and a scale up to more cores in production, e.g. up to 12 MPP processors and 12 SMP threads. The details follow.

Total core count	Hybrid		MPP processors
	MPP processors	SMP Threads	
12	12	1	12
24	12	2	24
36	12	3	36
72	12	6	72
144	12	12	144

Table 1: Core counts for LS-DYNA HYBRID and LS-DYNA MPP in this study

Hardware and Compute Environment

All runs are performed on one compute cluster. Every node of the cluster has two sockets for an Intel®, Xeon® CPU E5645 (6 cores) running on 2.40GHz, which sums up to 12 cores per node. Operating system on the nodes is a CentOS release 5.6.

The nodes are connected to each other via an InfiniBand from Mellanox Technologies MT26428 [ConnectX VPI PCIe 2.0 5GT/s - IB QDR / 10GigE] (rev b0).

The MPI protocol is a Platform MPI version 08.01.01.00 [9535] Linux x86-64.

The LS-DYNA HYBRID and LS-DYNA MPP versions used for the study are R7.1 beta versions, revision 86844, double precision.

Decomposition und Command Line

The decomposition divides the model into sub-domains. This is done by the primary processor in a single thread and, therefore, is independent of the number of cores specified. Ideally, the computational costs for each sub-domain should be equal and, in this case, the load balance should also be equal for each core.

In cases where the standard decomposition in LS-DYNA is not suited to the model, the decomposition can be steered via a 'pfile' [2]. This was indeed the case for this model where the decomposition is done via the pfile entry:

```
decomp {  numproc 12 region { parts 33 35 37 39 43 46 47 48 49 50 c2r 0 0 0 1 0 0 0 1 0 sy 10000 } }
```

This entry splits the blades and the hub in a radial scheme as shown in Figure 2.

An additional pfile entry used is `general { lstc_reduce }`, which was already discussed in the section "Output Consistency".

LS-DYNA HYBRID versions are started via the following command line (using Platform MPI):

mpirun –np <number of mpp processors> <hybrid executable name> ncpu=-<number of smp threads> <lsdyna command line options>

This command line contains options for both the number of MPP processors and the number of the SMP threads. It should be noted that the SMP thread specification should be negative to turn on the LS-DYNA SMP consistency option.

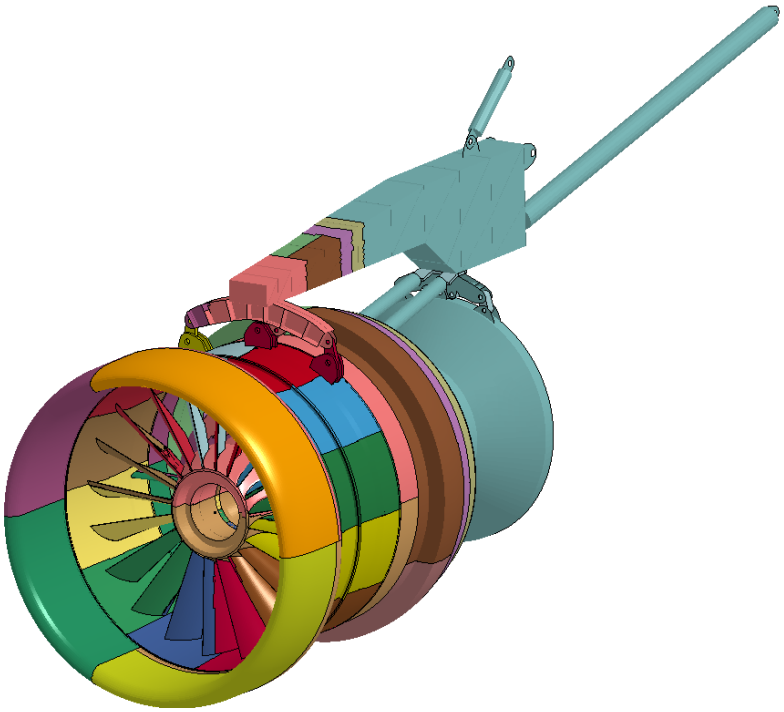

Figure 2 : Decomposition - each color refers to the core where the calculation is performed

Study Results

Results of the study are presented in the following sub-section. The performance, scalability, feature time distribution, and output consistency are reported.
In the figures for this section, the LS-DYNA HYBRID version runs are labeled “HYBRID MPP <X> / SMP <Y>” with <X> being the number of MPP processors and <Y> being the number of SMP threads. The total number of cores used in a LS-DYNA HYBRID version is the product of these two numbers. The LS-DYNA MPP version runs are labeled “MPP <X> proc.” with <X> being the number of MPP processors, which is equal to the number of cores used for the calculation. Results should be compared for equal core counts.

LS-DYNA HYBRID Performance

The performance herein is meant to be the elapsed total wall clock time compared for different core counts and parallel programming models (see Figure 3).These results give a first impression of the performance and the actual wall clock time, since the time the calculation needed to finish, is shown..
Comparing these total wall clock times, it can be seen, that the LS-DYNA HYBRID versions and LS-DYNA MPP version for 12 cores differ with a slight increase in time for the LS-DYNA HYBRID version. For core counts of 24, 36, 72 and 144, both versions show little differences in total wall clock time. The speed penalty for using a LS-DYNA HYBRID version vanishes for this application for core counts greater than 24 cores.

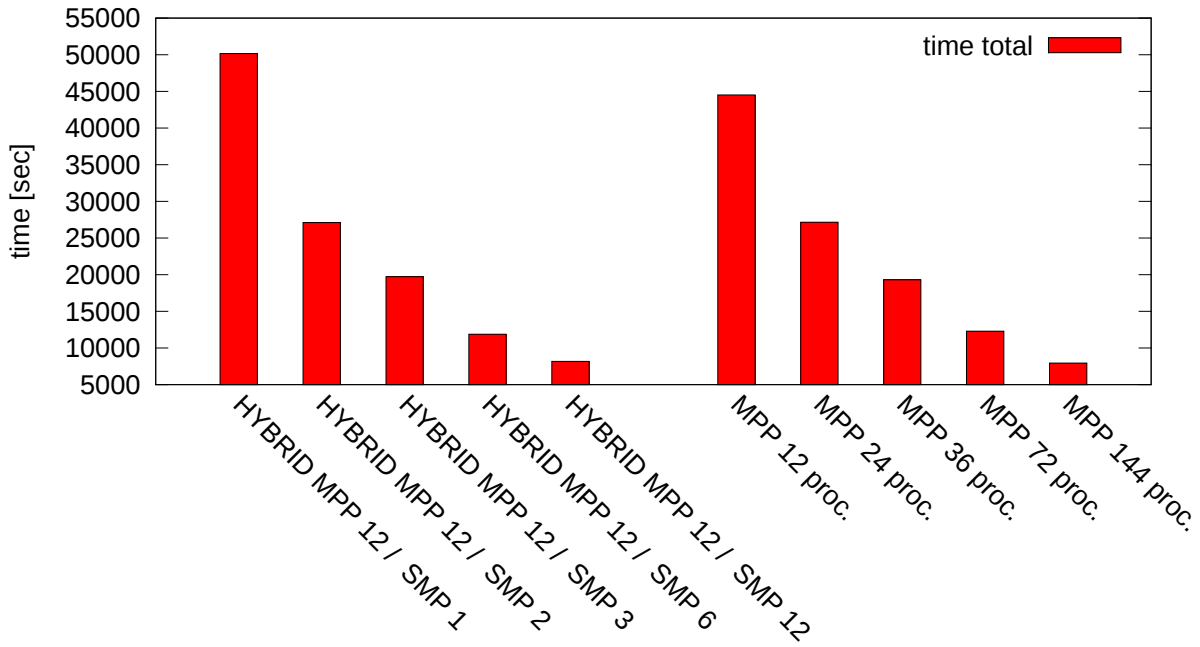

Figure 3: Wall clock time for different LS-DYNA HYBRID and LS-DYNA MPP runs

LS-DYNA HYBRID Scalability and Speedup

Scalability is defined here as the elapsed wall clock time divided by the time the LS-DYNA MPP version needed to finish on 12 cores. The speedup is defined as the reciprocal value of the scalability.

Figure 4 shows the scalability with respect to the total wall clock time. The speed penalty for the LS-DYNA HYBRID version for calculations using 12 cores can be quantified to approximately 12%, and for higher core counts, no speed penalty can be observed. Figure 5 shows the speedup for the different calculation variations. The speedup decreases with growing core counts for both parallel applications.

In Figure 6 the speedup for the contact and rigid body features are displayed. It can be observed, that the speedup for the LS-DYNA MPP versions are lower than the speedup for the LS-DYNA HYBRID versions. This is due to the data exchange for the contacts and rigid body features between the different cores, which increases the load on the network. For the LS-DYNA HYBRID version, this network load is constant for different core counts because the message passing did not increase with the constant 12 MPP processors used throughout the LS-DYNA HYBRID runs. The speedup for the element processing is displayed in Figure 7. This feature, qualitatively observed, has a lower impact on message passing than the contact and rigid body feature. Therefore, the speedup for LS-DYNA MPP versions is close to a linear increasing function.

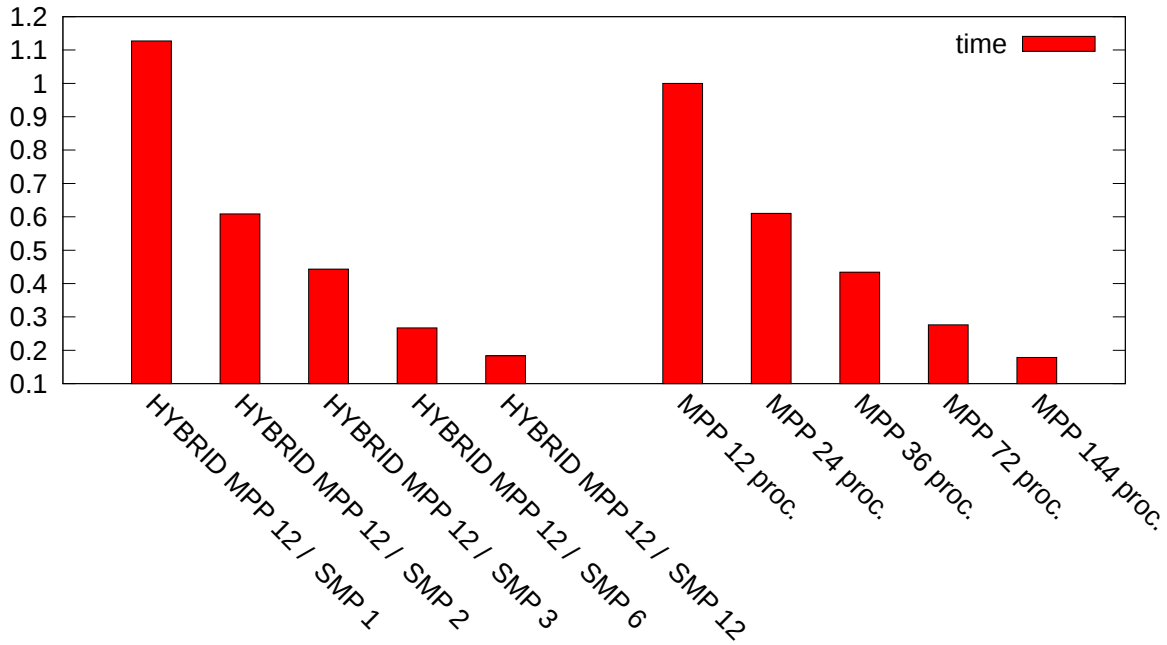

Figure 4: Scalability of wall time with respect to LS-DYNA MPP run with 12 processors

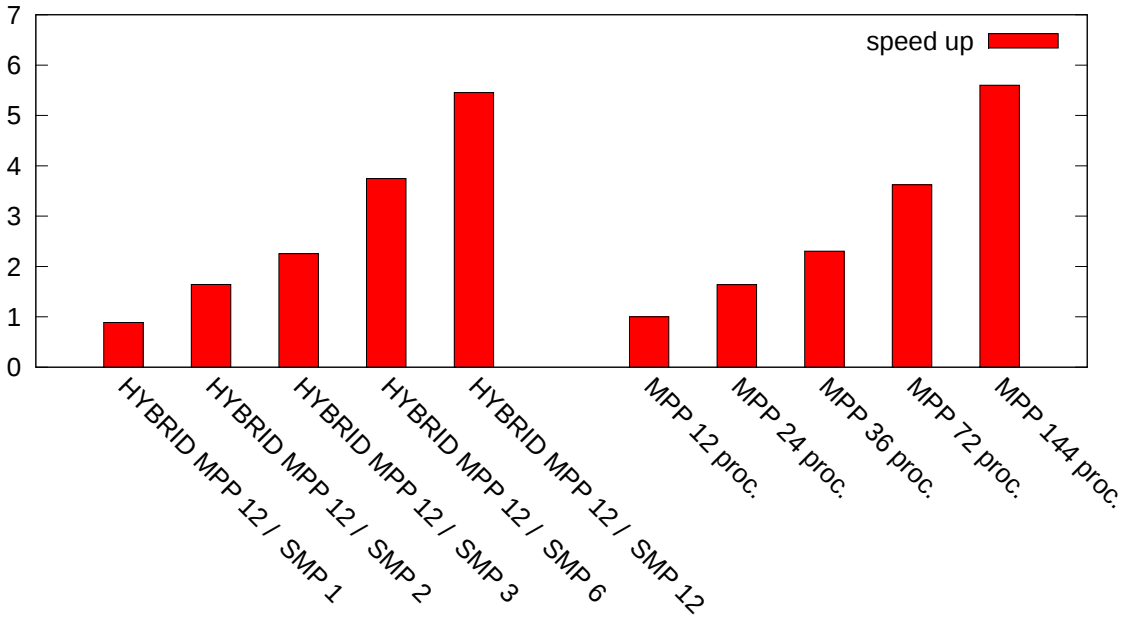

Figure 5: Wall time speedup with respect to LS-DYNA MPP run with 12 processors

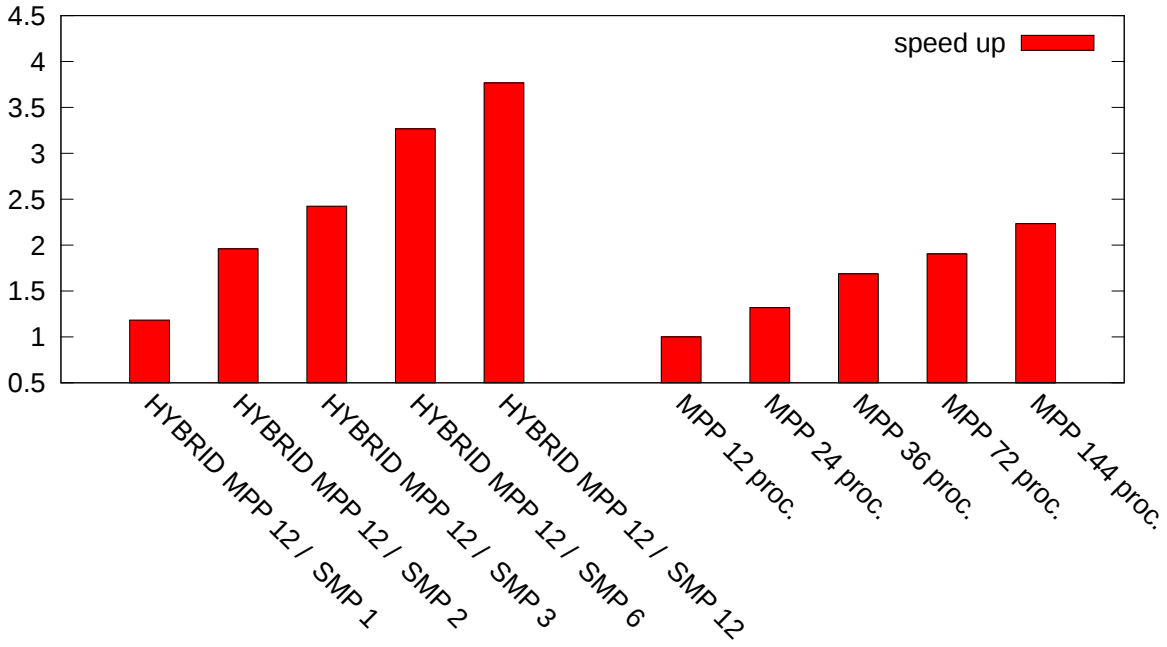

Figure 6: Contacts and Rigid Body speedup with respect to LS-DYNA MPP run with 12 processors

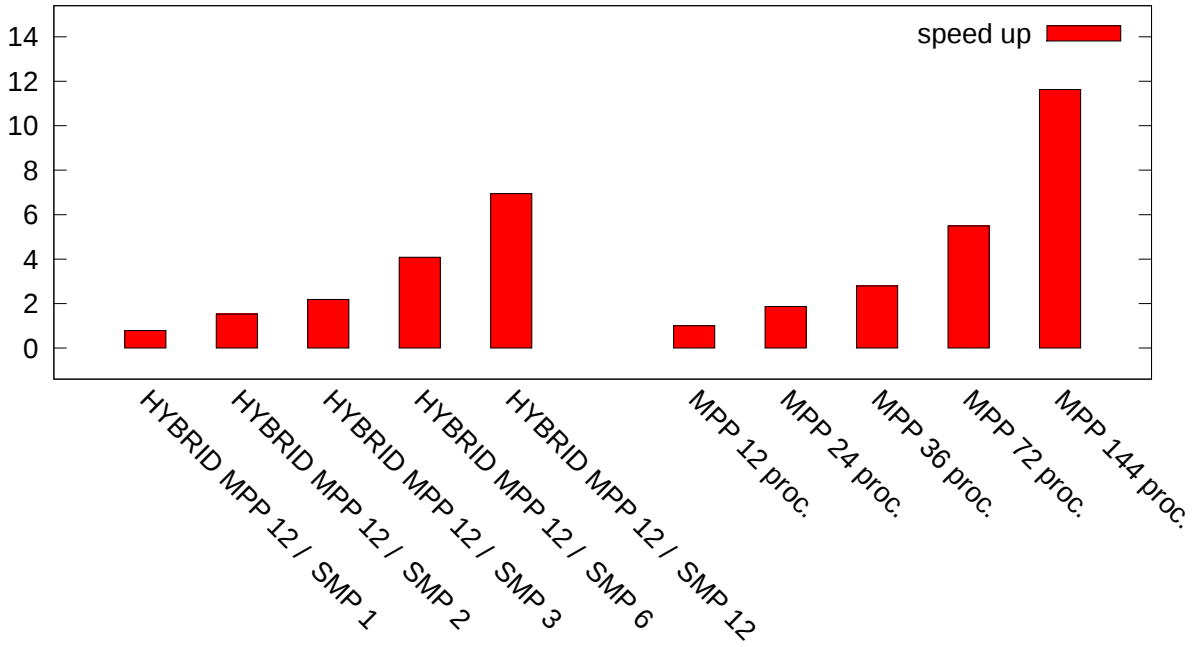

Figure 7: Element Speedup with respect to LS-DYNA MPP run with 12 processors

The speedup for the element feature in the LS-DYNA HYBRID version is qualitatively lower than the speedup for the LS-DYNA MPP version. This is assumed to be a result of overhead for the OpenMP® thread management and the additional part, $T_{overhead}$, respectively. The SMP part in the LS-DYNA HYBRID version did not scale in the same way as the LS-DYNA MPP version.

LS-DYNA HYBRID Feature Time Distribution

In Figure 8, details of the time distribution for different features are displayed. It can be seen that the three features, element processing, contact and rigid bodies and other (calculations) are dominant for all runs and the initial procedures like initialization, keyword processing, mpp decomposition, and disk I/O can be neglected.

The percentage of time spent in the contact and rigid body feature of the code increased for both programming models. Overall the increase in the LS-DYNA MPP version is qualitatively higher than in the LS-DYNA HYBRID version (see also Figure 6). As pointed out in the previous section, generally the contact and rigid body feature qualitatively need more time for communication than the element routines. With increasing core counts, the MPP version suffers from increasing $T_{communication}$ due to an increase of message passing for this feature.
For the contact and rigid body feature, the increase of computational time relative to the total time in the LS-DYNA HYBRID version is qualitatively higher since there is no additional message passing in this LS-DYNA HYBRID setup with a constant number of LS-DYNA MPP processors.

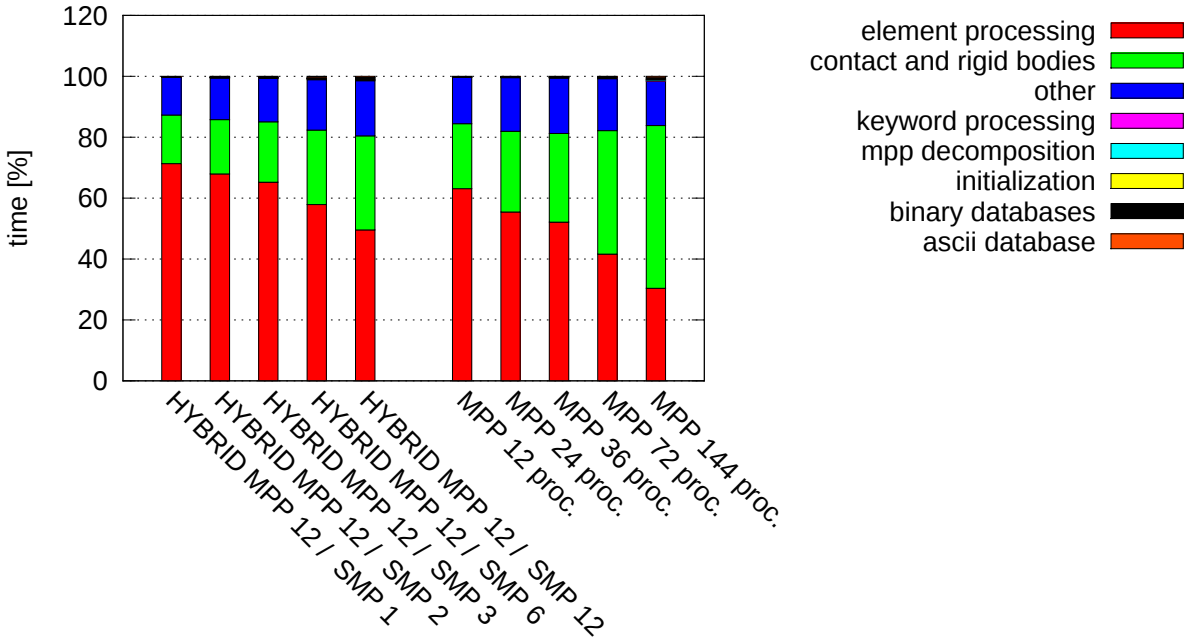

Figure 8: Wall clock details in percentage of total wall time

LS-DYNA HYBRID Output Consistency

Output consistency was tested over a wide range of outputs. They all showed the same characteristics, so only one output, the global internal energy, is presented here.

Figure 9 shows the global internal energy for the LS-DYNA HYBRID runs. The output for the five different combinations for the LS-DYNA HYBRID version, 12 MPP processors and 1, 2, 3, 6 and 12 SMP threads are equal and, therefore, it can be concluded that the results are consistent for all core counts in this study. This is the case for a variety of other outputs such as rigid body rotation of the fan hub, reaction forces in the joints, etc., which are not displayed here due to page count limitations.

In Figure 10, the output for LS-DYNA MPP versions is displayed. The output differs for different core counts. This is due to the aforementioned different summation order for internal vectors. As the output plots show, round off errors due to summation order become critical around time 0.0025 sec. At this time in the calculation, the released blade tip deforms inward and there is contact between the tailing blades and the released blade, as well as between the containment case and the released blade. With this combined contact and erosion scenario, minor changes in the summed vectors can change the contact configuration, i.e., contact is established at an earlier or later time step or erosion occurs at an earlier or later time step (compared to different core counts), which results in the scatter seen in Figure 10.

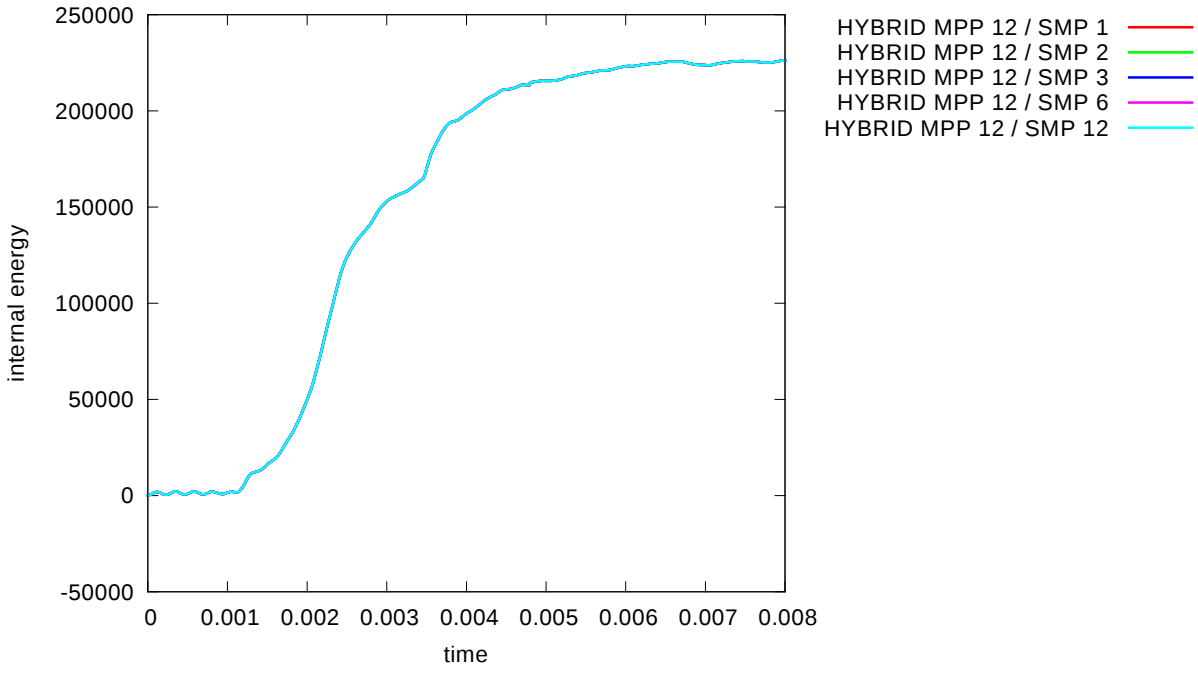

Figure 9: Output - internal energy - for LS-DYNA HYBRID runs

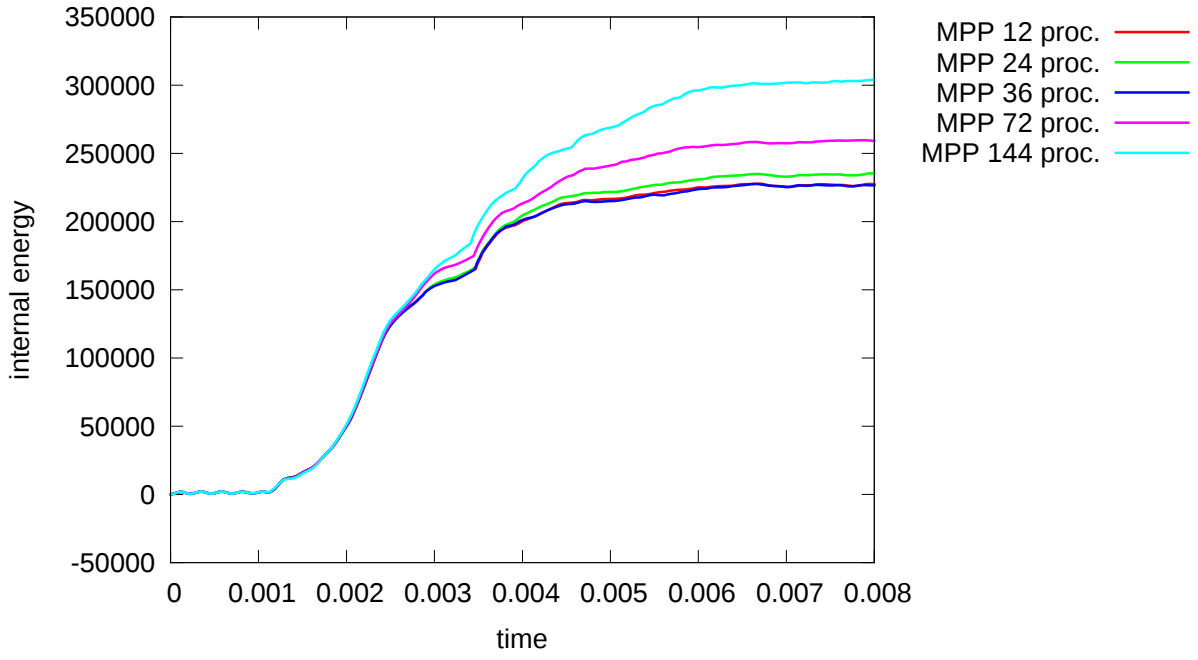

Figure 10: Output – internal energy – for LS-DYNA MPP runs

Conclusion

The considerations for using different parallel programming models are initially driven by a theoretical qualification of these models. In a production application, performance will depend on many factors such as hardware and compute environment set-up, the numerical model being used, and the decomposition of the model. General conclusions on performance, scalability, and output consistency are difficult to make and should only be made for a specific engineering application and numerical model.

The current study intended to perform such an evaluation on the Generic Fan Rig Model to better understand the performance, scalability, and output consistency for an aerospace application of a fan blade off event.

The following conclusions from the results of the LS-DYNA HYBRID and LS-DYNA MPP runs for five different core counts (12, 24, 36, 72, and 144) using a modified version of the AWG Generic Fan Rig model are summarized as follows:

- The speed penalty for the LS-DYNA HYBID version vanishes for core counts greater than 24 cores.
- For core counts of 24, 36, 72 and 144 the performance and speedup is similar for LS-DYNA HYBRID and LS-DYNA MPP versions.
- LS-DYNA MPP has an MPI communication overhead for the contact feature
- LS-DYNA HYBRID has an OpenMP® thread overhead for the element feature.
- The LS-DYNA HYBRID outputs are consistent for all five different core counts.
- The LS-DYNA MPP outputs scatter for different core counts.

As a final remark, it should also be stated that the speedup for the LS-DYNA HYBRID version due to reduced MPI protocol communication mentioned in the introduction section needs more cores to investigate.

Acknowledgments

The Authors wish to thank the Federal Aviation Administration (FAA) for funding and supporting the model development for the Fan Rig Model and the LS-DYNA Aerospace Working Group as well as Jim Day and Christoph Maurath for the valuable input on modelling approaches.

References

[1] AWG, LS-DYNA Aerospace Working Group, AWG ERIF Test Case Suite, at http://awg.lstc.com

[2] LSTC, LS-DYNA KEYWORD USER MANUAL, 7374 Las Positas Road, Livermore, CA, 94551, USA, Version R7.0 ed., February 2013.
[3] The OpenMP® API specification for parallel programming at http://www.openmp.org
[4] Ting-Ting Zhu, Jason Wang, "LS-DYNA Scalability Analysis on Cray Supercomputers", Proceedings 1st LS-DYNA China Conference, Dalian, 2013