APPENDIX Y: LS-DYNA HYBRID User Guide

This is a short user's guide for the hybrid version of LS-DYNA.

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® 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 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 version takes advantage of a combined SMP and MPP programming model to address this issue.

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.
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 actual 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 a portion of the MPP ranks 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 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 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.

SUPPORTED FEATURES

For a list of supported features please refer to "Appendix O MPP User Guide" of this manual.

CONTACT INTERFACES

LS-DYNA HYBRID uses the parallel contact algorithm of LS-DYNA MPP. For supported and unsupported contact options please refer to "Appendix O MPP User Guide" of this manual.

OUTPUT FILES AND POST-PROCESSING

For performance reasons, many of the ASCII output files normally created by LS-DYNA have been combined into a binary format. For further information on the available output files formats please refer to "Appendix O MPP User Guide" of this manual.

PARALLEL SPECIFIC OPTIONS

For memory options and restart capabilities please refer to "Appendix O MPP User Guide" of this manual

PFILE

LS-DYNA HYBRID uses the same pfile definitions as LS-DYNA MPP. For details on these definitions please refer to "Appendix O MPP User Guide" of this manual.

EXECUTION OF LS-DYNA HYBRID

LS-DYNA HYBRID, as well as LS-DYNA MPP, uses a message passing interface to communicate between several MPP ranks. The message passing interface is not part of the LS-DYNA implementation. A third party MPI product software is needed to execute LS-DYNA HYBRID when more than one MPP rank is used. There are several MPI products available and LSTC provides LS-DYNA HYBRID executables for all major implementations of the MPI standard.

APPENDIX Y

For the execution syntax of the MPI product, please refer to the user documentation of this product. In the following paragraphs <MPI-EXE> is used to identify the run command for the MPI product and "-np" is used as a flag to the MPI product executable to specify the number of MPP ranks.

The generic example for an execution syntax for LS-DYNA HYBRID is:

<MPI-EXE> -np < number of MPP ranks > <LS-DYNA HYBRID name > ncpu = - < number of OpenMP threads > <general LS-DYNA command line options >

The number of MPP ranks are specified in the same way as for LS-DYNA MPP versions. In addition to this specification, the OpenMP threads per MPP rank must specified as well. This is done in the same way as for the LS-DYNA SMP version, via the command line option "ncpu=". It is strongly recommended to use a negative number here to activate the SMP consistency flag.

APPENDIX Y