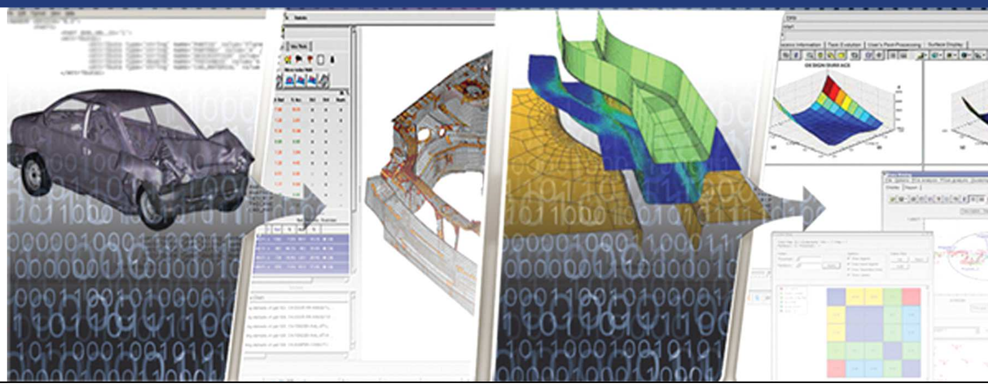


RADIOSS THEORY MANUAL

Version 2017 – January 2017

Large Displacement Finite Element Analysis

Chapter 12

Altair

Altair Engineering, Inc., World Headquarters: 1820 E. Big Beaver Rd., Troy MI 48083-2031 USA
Phone: +1.248.614.2400 • Fax: +1.248.614.2411 • www.altair.com • info@altair.com

CONTENTS

Chapter 12

RADIOSS PARALLELIZATION

12.0 RADIOSS PARALLELIZATION

The performance criterion in the computation was always an essential point in the architectural conception of RADIOSS. At first, the program has been largely optimized for the vectored super-calculators like CRAY. Then, a first parallel version SMP made possible the exploration of shared memory on processors.

In the case of analysis of systems with high number of d.o.f., the use of shared memory parallel machine architectures is common. In RADIOSS, there are two models of parallel programming:

- SMP : Shared Memory Processors,
- SPMD : Single Program Multiple Data.

In this chapter, the principle of RADIOSS parallelization is described.

12.1 Measure of performance of a parallel application

The Speed-Up is the ratio of sequential time T(1) and the parallel time on P processors T(P):

$$S(P) = T(1) / T(P) \qquad \text{EQ. 12.1.0.1}$$

The efficiency is defined as:

$$E(P) = S(P) / P \;; \qquad E(P) \le 1 \qquad \text{EQ. 12.1.0.2}$$

The Amdahl's law for multitasking is used to determine the speed-up:

$$S(P) = \frac{\left(T_{Seq} + T_{Par}\right)}{\left(T_{Seq} + {T_{Par}}/{P}\right)} \qquad \text{EQ. 12.1.0.3}$$

where T_{Par} and T_{Seq} are the computation times respectively related to parallel and non-parallel parts. As $T_{Seq} + T_{Par} = 1$, write:

$$S(P) = \frac{1}{\left(T_{Seq} + {T_{Par}}/{P}\right)} \qquad \text{EQ. 12.1.0.4}$$

The limit value can be obtained when the process number tends to infinite:

$$S(\infty) = \frac{1}{T_{Seq}} \qquad \text{EQ. 12.1.0.5}$$

Table 12.1.1 gives the Speed-Up in function of number of processors and the rate of parallelization in the program. It can be seen that if the rate of parallelization is less 95%, the computation acceleration will not be greater than 20 however the number of processors. This means that to obtain a good scalability of a code, at least 99% of the program must be parallel.

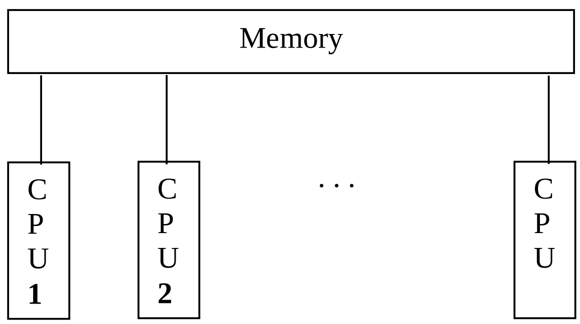
Table 12.1.1 Speed-up in function of process number and parallelization

Seq / P \ Process number	**2**	**4**	**8**	**16**	**32**	**64**	**128**	**∝**
100%	2.0	4.0	8.0	16.0	32.0	64.0	128.	∞
99%	2.0	3.9	7.5	13.9	24.4	39.3	56.4	100.
98%	2.0	3.8	7.0	12.3	19.8	28.3	36.2	50.0
97%	1.9	3.7	6.6	11.0	16.5	22.1	26.6	33.3
96%	1.9	3.6	6.3	10.0	14.3	18.2	21.1	25.0
95%	1.9	3.5	5.9	9.1	12.5	15.4	17.4	20.0
90%	1.8	3.0	4.7	6.4	7.8	8.7	9.3	10.0
50%	1.3	1.6	1.8	1.9	1.9	2.0	2.0	2.0

12.2 SMP: Shared Memory Processors

RADIOSS SMP version is based on the concept of computers with shared memory as the architecture is described in Figure 12.2.1.

Figure 12.2.1 Architecture of shared memory



In this case, all processors can access to a common memory space. From programming point of view, each process called parallel task, reach to the entire memory space allocated by the program. It is necessary to manage properly the access to this shared memory by introducing barriers and locking mechanisms. The SMP model programming has the advantage to be managed easily. However, the performance of the method depends on the ratio between the memory access speed and the CPU speed.

The parallelism approach used in RADIOSS SMP is a multi-task programming type. The tasks are explicitly managed by the programmer. The computation tasks are attributed to the processors by a dynamic procedure as they are available. This approach is especially adapted to the super-computers used as computation server where the load of a given processor varies with respect to others. The SMP version is developed for computers with shared memory architecture and cannot be used efficiently on the super-computers with distributed memory or cluster structures.

The RADIOSS SMP version is constantly improved since the first release. However, the efficiency of SMP version has to be updated to take into account the evolution of computer architecture in memory management and CPU speed.

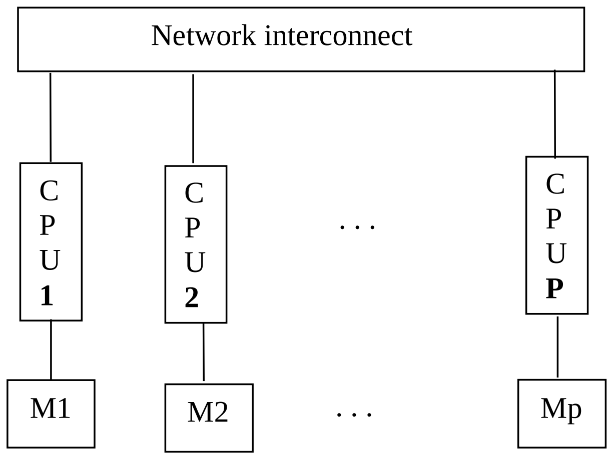
12.3 SPMD: Single Program Multiple Data

The development of the first SPMD version of RADIOSS is started in 1994. The version became a real alternative to SMP version after a long period of parallelization and optimization of the code. In fact, the scalability of the version is much better than the SPM version. The SPMD version allows using more processors with a better efficiency. It makes possible to use up to 64 processors. In addition, all RADIOSS Crash options are available in this version including “Arithmetic Parallel” option.

The principle of program is based on Single Program Multiple Data, where the same program runs with different data. RADIOSS Starter carries out domain decomposition. Then, RADIOSS Engine has just to send data to different processors in an initialization step. Thereafter, each program runs over each sub domain. It is necessary to communicate information between processors to manage data on the border of domains. This is carried out in RADIOSS using MPI (Message Passing Interface) library.

Figure 12.3.1 illustrates the architecture of multi-processor computers with distributed memory. The RADIOSS SPMD version runs independently to architecture of memory as well on the computers with shared memory or distributed memory or a set of work stations in a network.

Figure 12.3.1 Architecture with distributed memory



12.4 HMPP: Hybrid Massively Parallel Program

The Hybrid MPP version is a new parallel version developed in HyperWorks 11.0 which combines the best of the two previous parallel versions inside a unique code.

Benefits

This new approach allows reaching an impressive level of scalability. RADIOSS HMPP can scale up to 512 cores and more for a real performance breakthrough.

RADIOSS HMPP is independent of the computer architecture, more flexible and efficiently adapted to any hardware resources. It can run on distributed memory machines, shared memory machines, workstation cluster, or a high performance computation cluster. It can better exploit the inside power of highly multi-core machine and optimize the software according to the hardware.

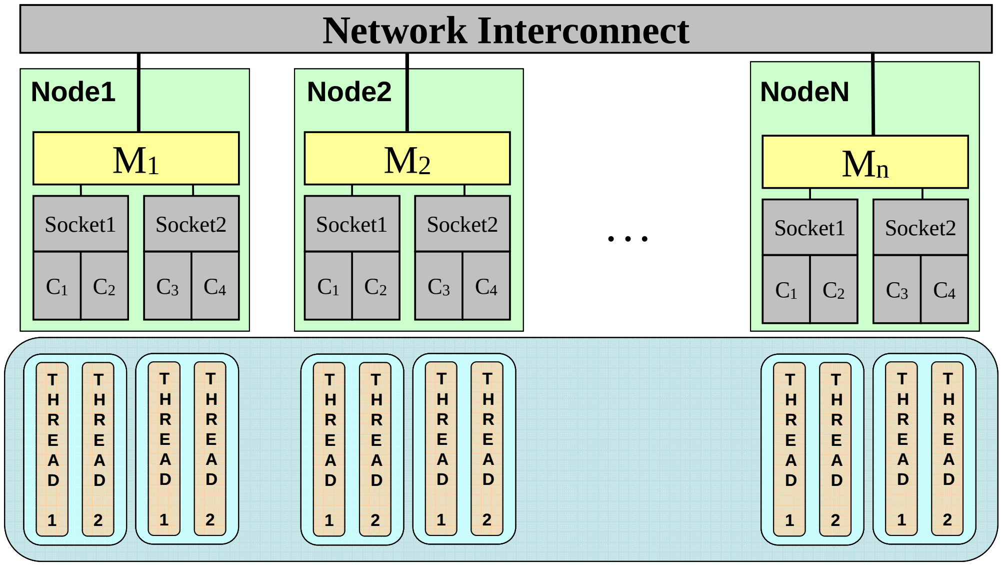
It decreases installation and maintenance costs by having a unique parallel version instead of two types of executables.

It improves the quality of the code in term of numerical results by having a full convergence between SMP and SPMD results. Parith/ON provides a unique answer independent of the number of SPMD domains and the number of SMP threads used.

Example of Hybrid usage

In the example below, the hardware is composed of N nodes of a cluster. Each node is a dual processor machine and each processor is a dual core. On each node the memory is shared between processors but the memory is distributed between the nodes.

Figure 12.3.2 Hybrid version run with 2 SMP threads per SPMD domains



With the Hybrid version, it is easy to optimize the software to run on such complex architecture. Figure 12.3.2 shows that each SPMD domain is computed by a different processor. The number of threads per domain is set to the number of cores per processor (two).

So, each processor computes a SPMD domain using two SMP threads, one per core.

How does it work?

For SPMD version, RADIOSS Starter divides the model into several SPMD domains and writes multiple RESTART files.

Then, mpirun command is used to start all the SPMD programs. Each program computes a SPMD domain using the number of SMP threads set. Indeed, each MPI program is a SMP parallel program. The management of computation at the frontiers of the domains remains and it is necessary to communicate some information between programs using MPI.

Example of execution

RADIOSS Starter is run from the command line. Here, the number of SPMD domains (Nspmd) with option -nspmd 16 is specified:

```
./s_11.0_linux64 –nspmd 16 –i ROOTNAME_0000.rad
```

The number of SMP threads (Nthread) is set through the use of environment variable OMP_NUM_THREADS:

```
setenv OMP_NUM_THREADS 2
```

Then, RADIOSS Engine is run using mpirun command:

```
mpirun –np 16 ./e_11.0_linux64_impi –i ROOTNAME_0001.rad
```

Notes:

- -np value of mpirun must match –nspmd value
- The total number of processes is equal to Nspmd*Nthread (32 in this example)

Recommended setup

As seen in the example above, a good rule is to set the number of SPMD domains equal to the number of sockets and to set the number of SMP threads equal to the number of cores per socket.

With a low number of cores (below 32), a pure SPMD run might be more effective but the performance gap should be small if the setup explained above is respected.

With a very high number of cores (1024), it is possible to increase the number of SMP threads up to the number of cores per node and set only one SPMD domain per node to maximize performance if the limit of scalability of the interconnect network is reached.