Introduction to LS-DYNA MPP

The Arup Campus, Blythe Gate, Blythe Valley Park, Solihull, West Midlands, B90 8AE

tel: +44 (0) 121 213 3399

email: dyna.support@arup.com

2018

Outline

- Introduction to MPP LS-DYNA
- MPP decomposition methods
- Visualising decompositions
- Load balancing information
- MPP contacts
- Restart analysis

SMP (Symmetric Multi-Processing)

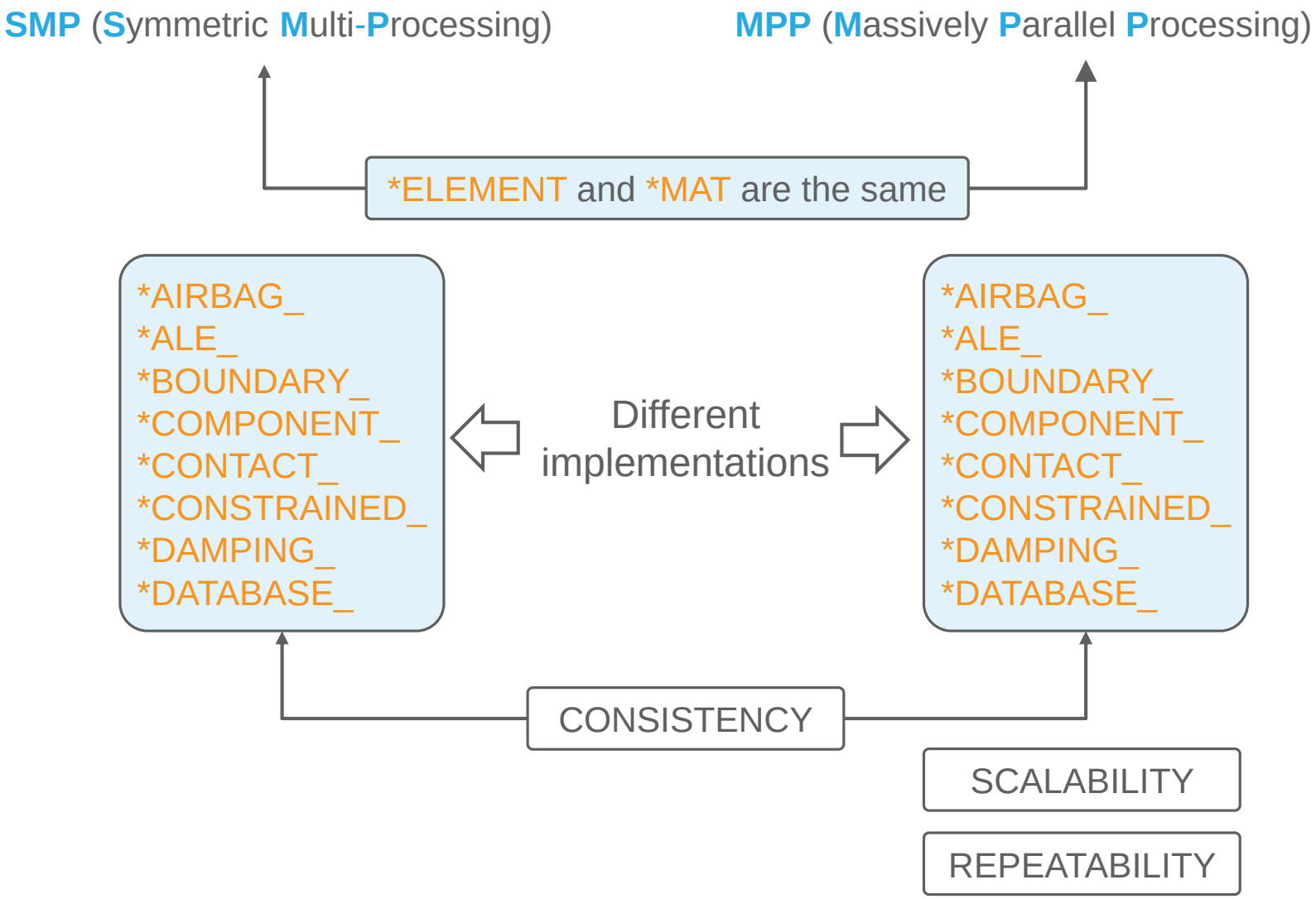
- Originated from the serial code
- Uses OpenMP® directives to split tasks into parallel threads
- Runs on computers with multiple identical cores with the cores and memory connected via a shared data bus
- **Consistent results with different cores (consistency flag turned on!)**
- Scalable up to ~ 8 CPUs

$$T_{elapsed} = T_{cpu} + T_{sys} + T_{omp}$$

MPP (Massively Parallel Processing)

- Uses a message passing protocol to exchange information between the cores on a board or over a network
- MPP solver performs a domain decomposition of the problem…
- …. and distributes the sub-domains to different cores using MPI protocols for communication between the subdomains during analysis
- **Results change with different cores**
- Scalable >> 16 CPUs

$$T_{elapsed} = T_{cpu} + T_{sys} + T_{mpp}$$



MPP domain decomposition involves dividing the model into several domains, which are done by the primary processor, and assigning each domain to a core.

The elements and nodes on the boundary of each domain transfer information to those in the other domain over a network connection using message passing protocols.

Factors that affect parallel performance:

Load Balance:

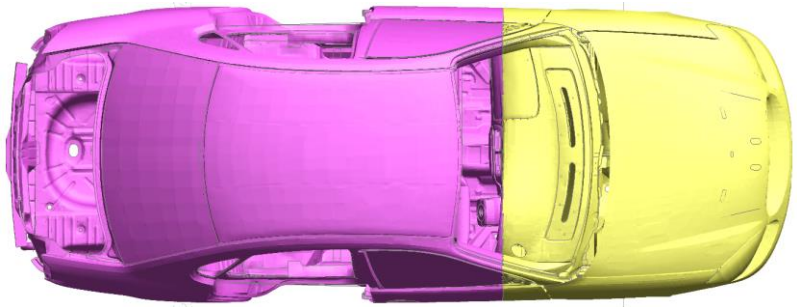
- Boundaries of the decomposed domains
- Variations between different element formulations and material models
- Treatment of contacts
- Special features used in the modelling

Communication:

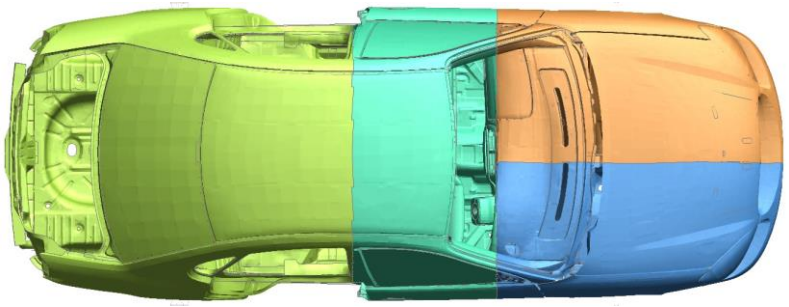
- Memory/Cache system
- Interconnections
- MPI libraries
- Fortran compiler

Recursive Coordinate Bisection (RCB)

- Recursively bisects the model about a plane (one of three axes) perpendicular to the longest dimension
- Method tends to generate cube shaped domains aligned along the coordinates axes



2 MPI ranks



4 MPI ranks

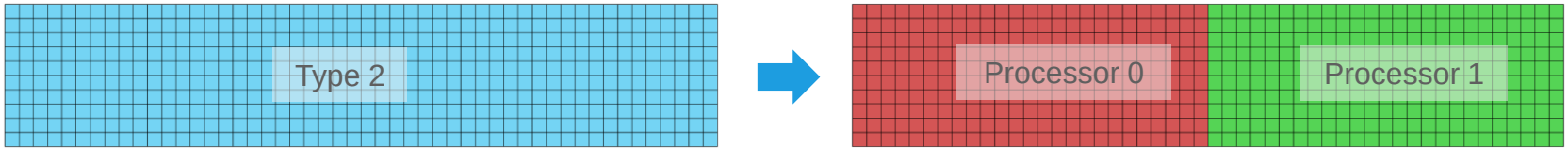
8 MPI ranks

16 MPI ranks

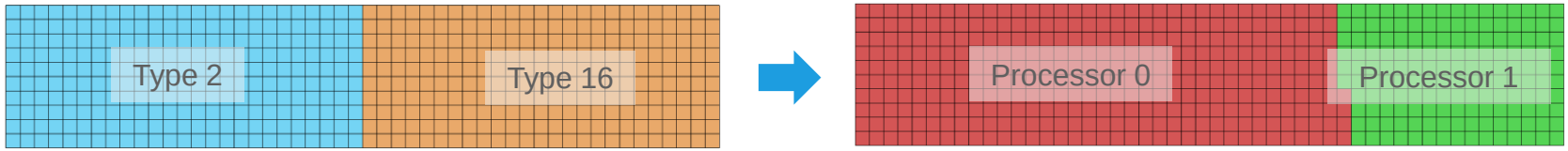
Decomposition methods in LS-DYNA

To improve the load balance LS-DYNA has some additional built-in intelligence

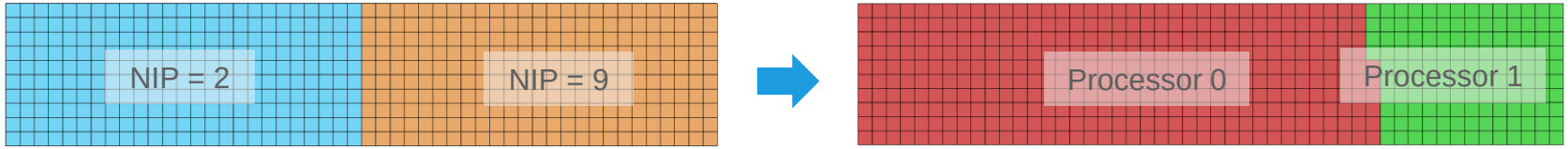
1) Initial Geometry



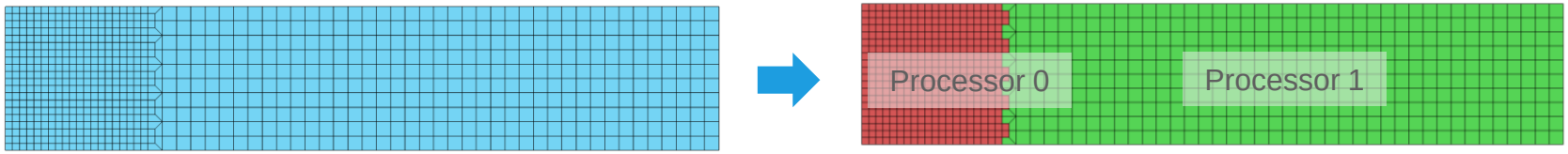
2) Element Formulation



3) NIP – Number of Integration Points



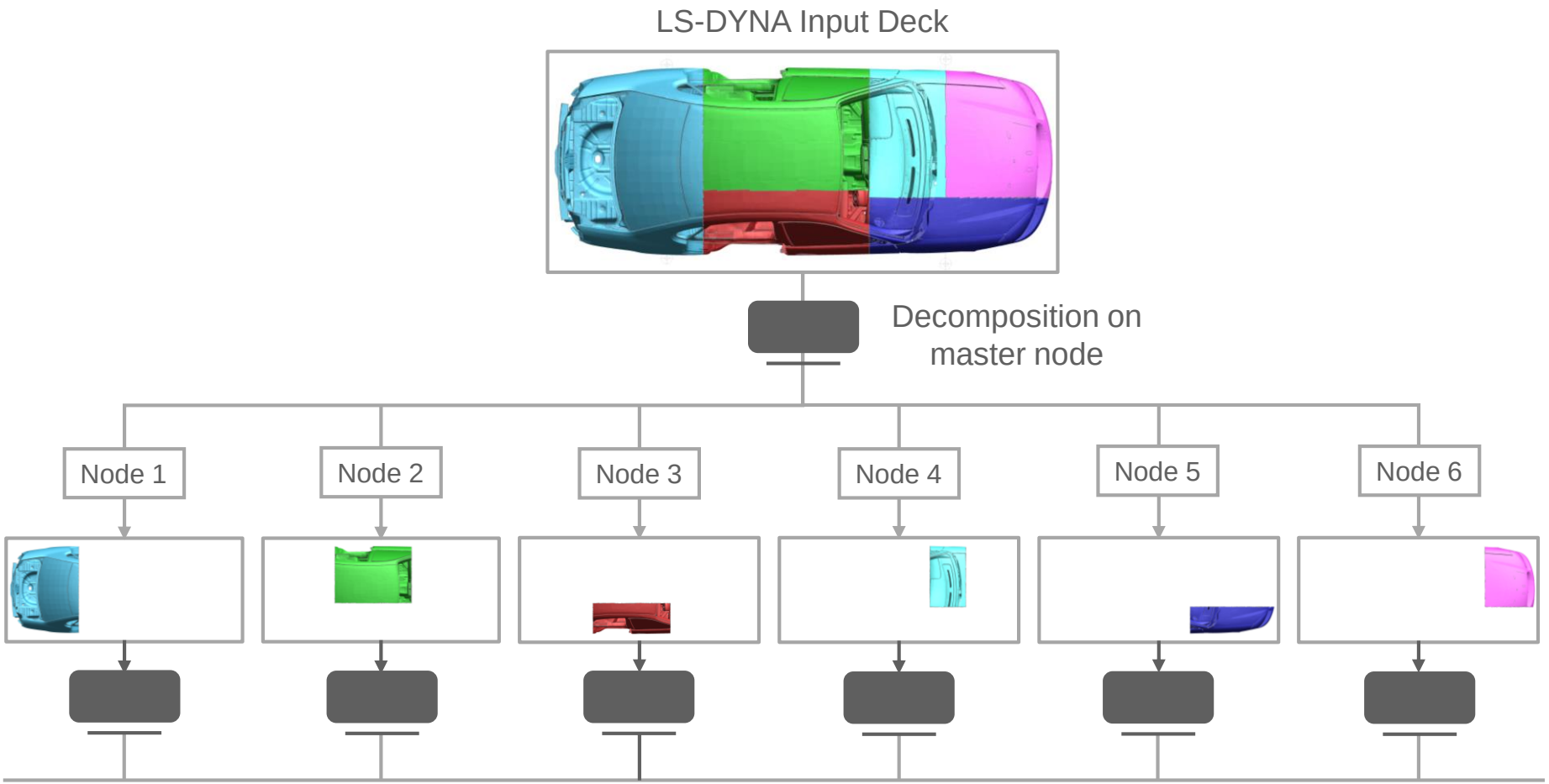
4) Element count weighting



No weighting is given to material formulation

http://www.d3view.com/2008/02/rcb-decomposition-and-ls-dyna-built-in-intelligence/

MPP Domain Decomposition

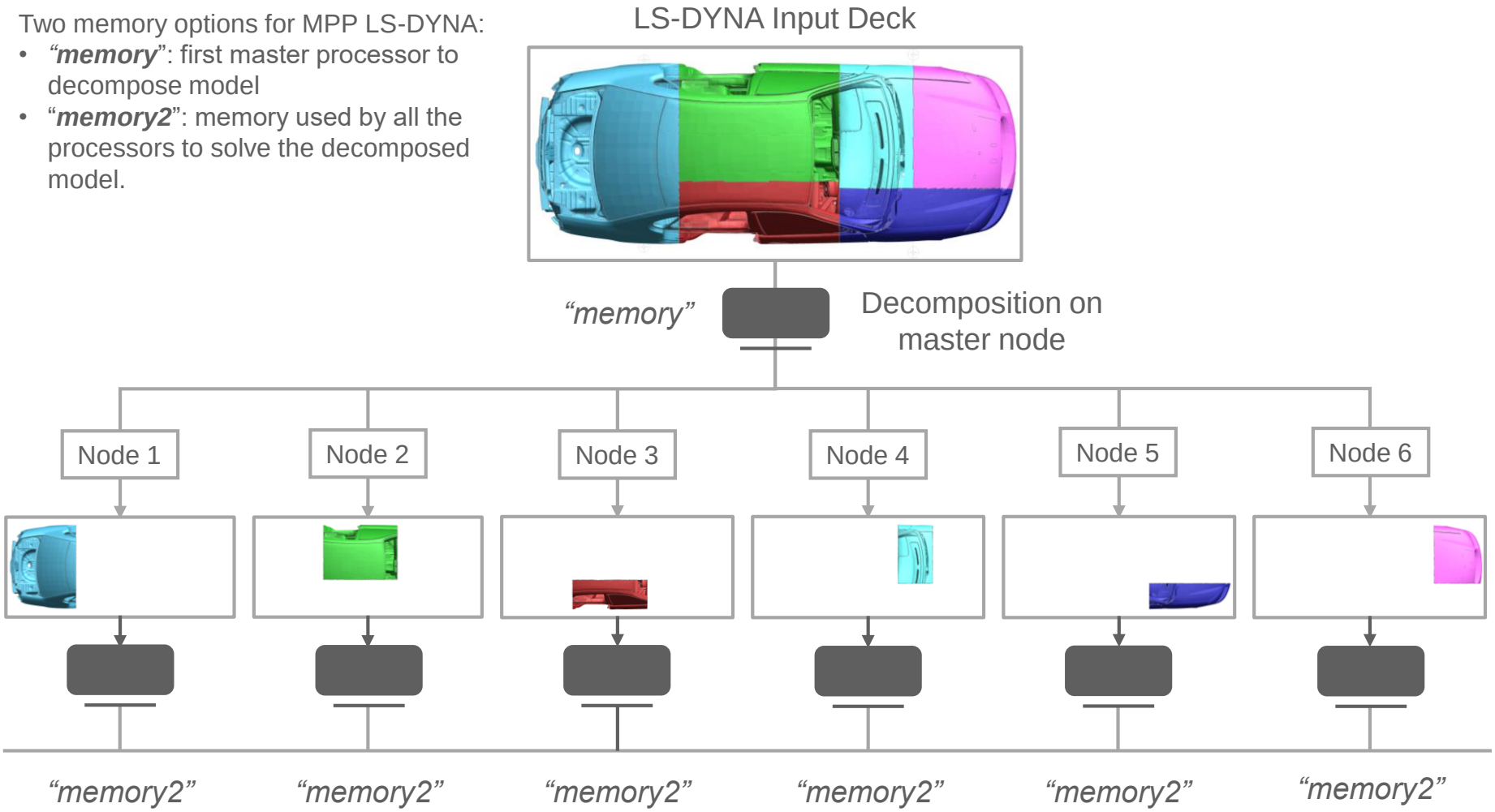


http://www.d3view.com/2008/02/rcb-decomposition-and-ls-dyna-built-in-intelligence/

Memory Settings for MPP LS-DYNA

Two memory options for MPP LS-DYNA:

- *"**memory**"*: first master processor to decompose model
- "***memory2***": memory used by all the processors to solve the decomposed model.

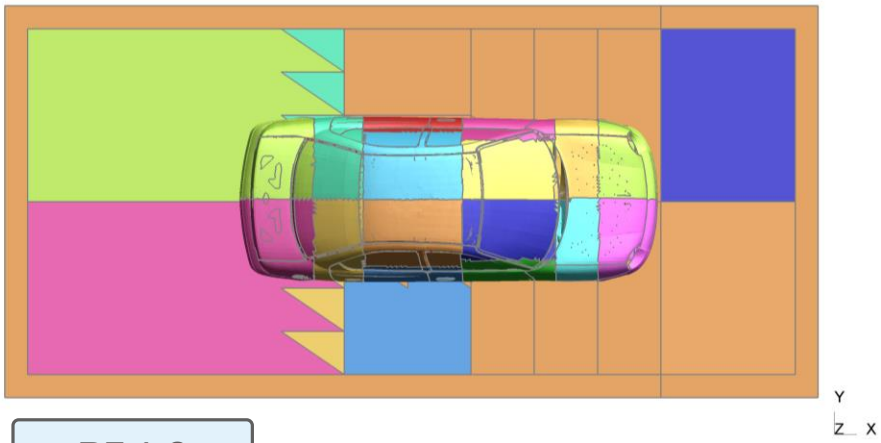


mpirun –np 6 mpp971 i=test.k memory=80m memory2=20m

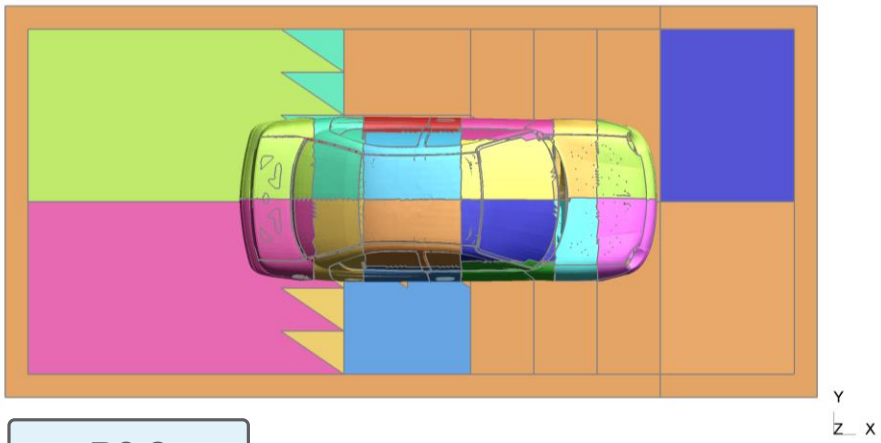
http://www.d3view.com/2008/02/rcb-decomposition-and-ls-dyna-built-in-intelligence/

Decomposition in LS-DYNA

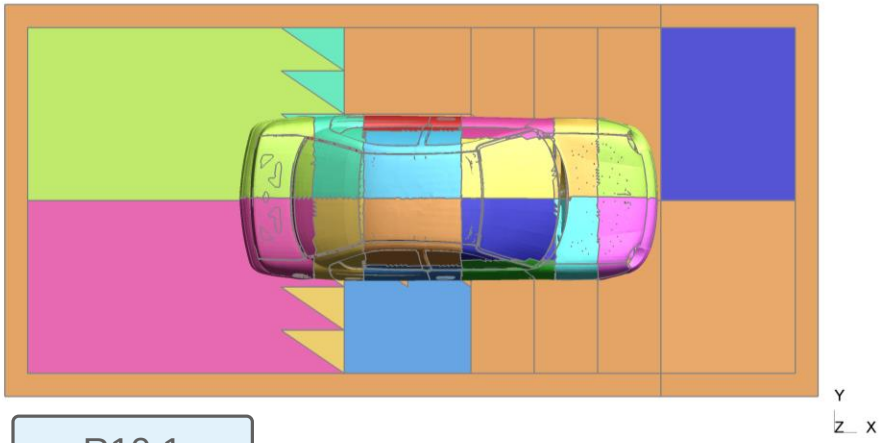
Note: default rcb decomposition method pretty consistent from one LS-DYNA solver to another



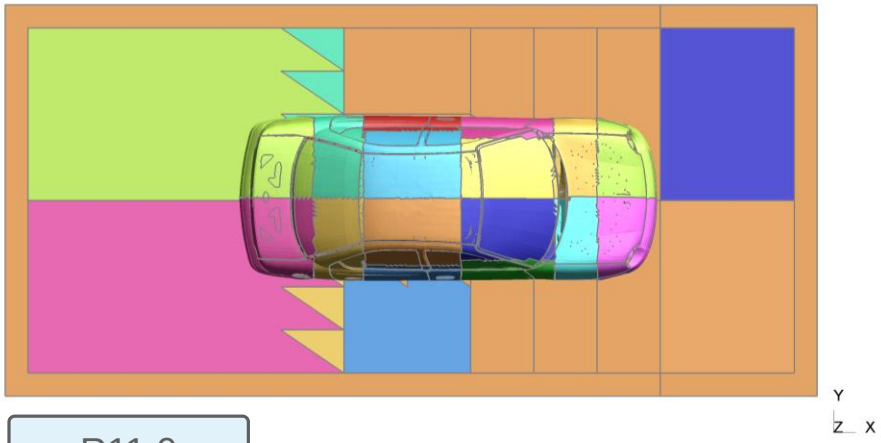
R7.1.3



R9.3



R10.1



R11.0

What is a PFILE?

pfile_neon_refined_revised:

```
general { nodump nofull nod3dump nofail nobeamout }
decomp { sy 2 }
dir { local /local _ shm _ dir/neon _ refined _ revised }
```

- pfile contains MPP specific parameters that effect the execution of the program
- The file is split into sections, with several options in each section:
  - **directory, decomposition, contact, general**
- The file is case insensitive and free format input
- Can be used as a separate file or via the *CONTROL_MPP_PFILE keyword
- Full list of options can be found in Appendix O and *CONTROL_MPP card of LS-DYNA Keyword User's Manual Vol. I

Additional comments:

- Check the output of PFILE directives in the .otf/d3hsp file – useful when learning pfile syntax
- From **R10.1** – use of parameters (defined via *PARAMETER) in *CONTROL_MPP_PFILE

Special Decompositions

- If the default decomposition algorithm is not desired, it is possible for the user to provide a set of coordinate transformation functions which are applied to the model before it is decomposed (Appendix O: LS-DYNA MPP User Guide).
- **General form** for a special decomposition would look like this in pfile:

```
decomposition {
region { <region specifiers> <transformation> <grouping> }
region { <region specifiers> <transformation> <grouping> }
<transformation>
}
```

<region specifiers> are:
- box
- sphere
- cylinder
- parts
- partsets
- silist

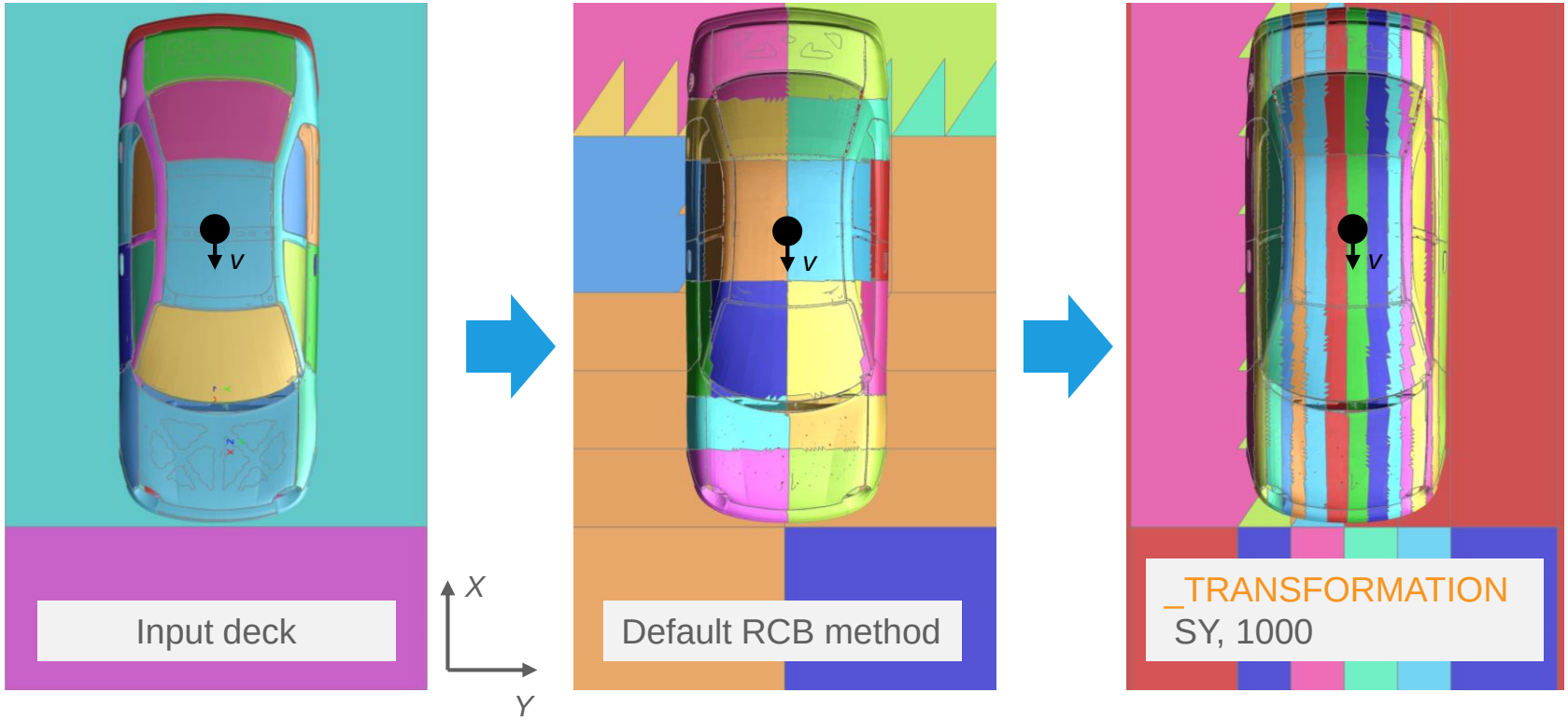
<transformation specifiers> are:
- local
- sx t, sy t, sz t
- rx t, ry t, rz t
- txyz x y z
- mat
- 3vec
- C2R
- S2R

<grouping specifiers> are:
- lumped
- together (R11.0)
- nproc *n frstp* (R11.0)

Special Decompositions

- If the default decomposition algorithm is not desired, it is possible for the user to provide a set of coordinate transformation functions which are applied to the model before it is decomposed (Appendix O: LS-DYNA MPP User Guide).

*CONTROL_MPP_DECOMPOSITION_TRANSFORMATION
Purpose: specifies transformations to apply to modify the decomposition



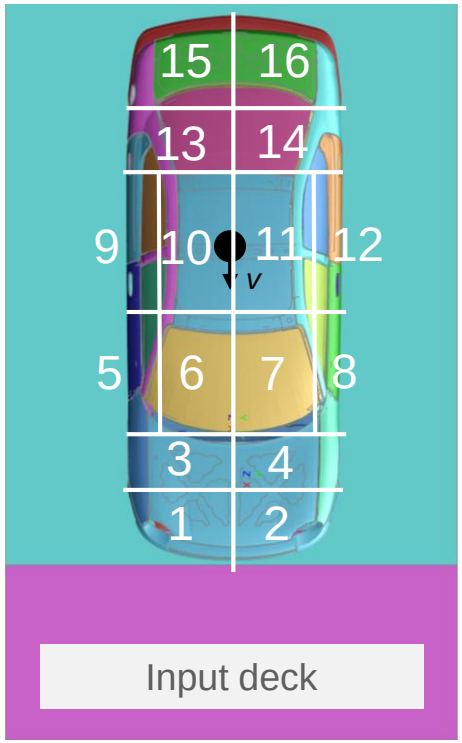
http://www.d3view.com/2006/09/simulation-model-decomposition-using-recursive-bisection-method-rcb/

Computationally expensive region decomposed across all processors

Special Decompositions

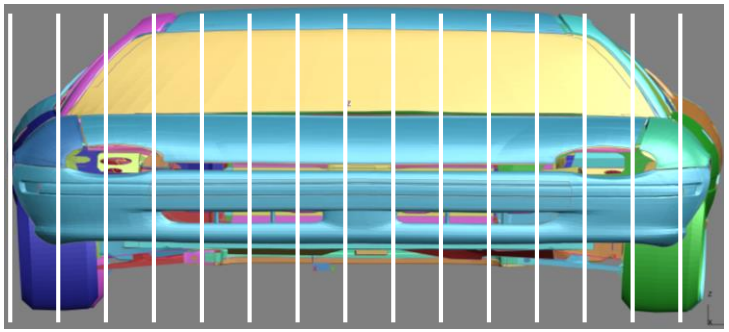
- If the default decomposition algorithm is not desired, it is possible for the user to provide a set of coordinate transformation functions which are applied to the model before it is decomposed (Appendix O: LS-DYNA MPP User Guide).

*CONTROL_MPP_DECOMPOSITION_TRANSFORMATION
Purpose: specifies transformations to apply to modify the decomposition

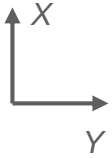


Input deck

In this example, the following scales the Y axis of a model by a factor of 1000



For certain load cases this method decomposition may be more efficient

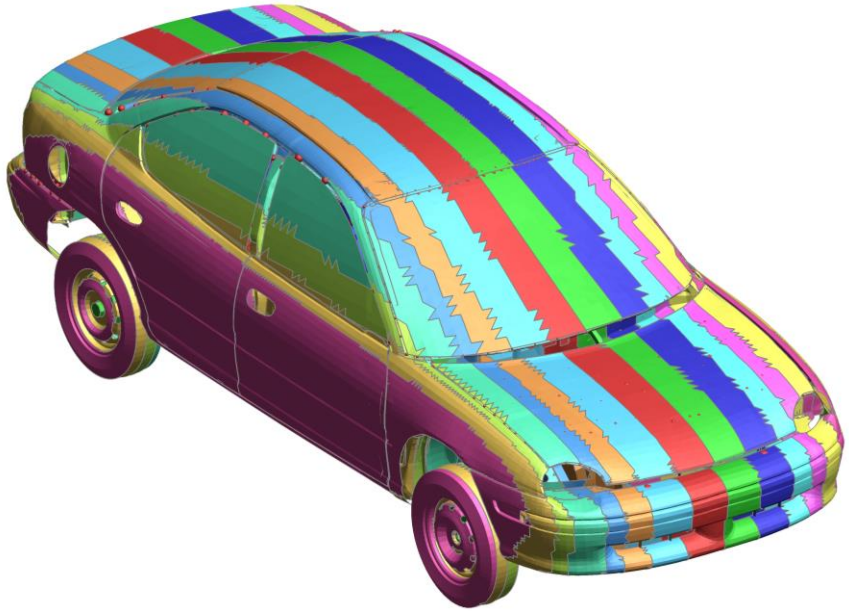


_TRANSFORMATION
SY, 1000

_TRANSFORMATION
SY, 1000

http://www.d3view.com/2006/09/simulation-model-decomposition-using-recursive-bisection-method-rcb/

*CONTROL_MPP_PFILE
decomp {sy 200.00}



sy 200.0: Scale the current y coordinates by 200.00

11 minute run time on 16 CPU

*CONTROL_MPP_PFILE
decomp { silist 1 sy 200.00}

SILIST = *CONTROL_MPP_DECOMPOSITION_CONTACT_DISTRIBUTE

silist 1: All elements involved in a contact interface 1 are included in the region.

7 minute run time on 16 CPU

Problem:

- MPP decomposition is based on averaging the computational across the processors. If a model has been modified or refined, the cost profile will change and model will decompose in a different way. This may change numerical results, particularly for sensitive models that exhibit material/element failure. In such models, it would be difficult to distinguish between 'real' changes due to design updates, and changes due to the code.

RCBLOG

keyword:

*CONTROL_MPP_DECOMPOSITION_RCBLOG

pfile:

decomposition{ rcblog file_rcblog}

In the first job run, LS-DYNA will store all the cut information and also retain all other options in the pfile into 'file_rcblog'.

In subsequent runs, replace p=pfile to p=file_rcblog and LS-DYNA will decompose the model based on the preserved cut lines.

Visualising decompositions

Decomposition of element domains can be visualised by:

- *CONTROL_MPP_DECOMPOSITION_SHOW
  - Outputs one plot state then terminates the analysis.
  - Each part correspond to the group of solids, shells, beams, thick shells, or SPH particles assigned to a particular processor
- *CONTROL_MPP_DECOMPOSITION_OUTDECOMP
  - Does not terminate the analysis early. Instructs LS-DYNA to output a settings file that contours elements according to processor ID.
    - ITYPE EQ.1: database in LS-PrePost format:

      decomp_parts.lsprepost (binary)
    - ITYPE EQ.2: database in animator format:

      decomp_parts.ses (ASCII)
  - *When ITYPE EQ. 1, the elements assigned to any particular core can be viewed and animate by LS-PrePost by (1) reading the d3plot data, and then (2) selecting Models > Views > MPP > Load > decomp_parts.lsprepost*

Command in partition file (pfile): OUTDECOMP ITYPE

*CONTROL_MPP DECOMPOSITION_OUTDECOMP (ITYPE EQ. 2)

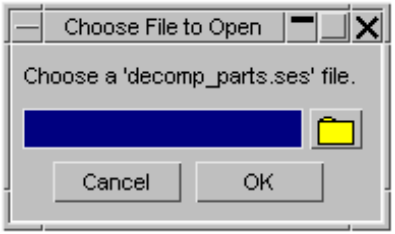
Run LS-DYNA job

LS-DYNA®

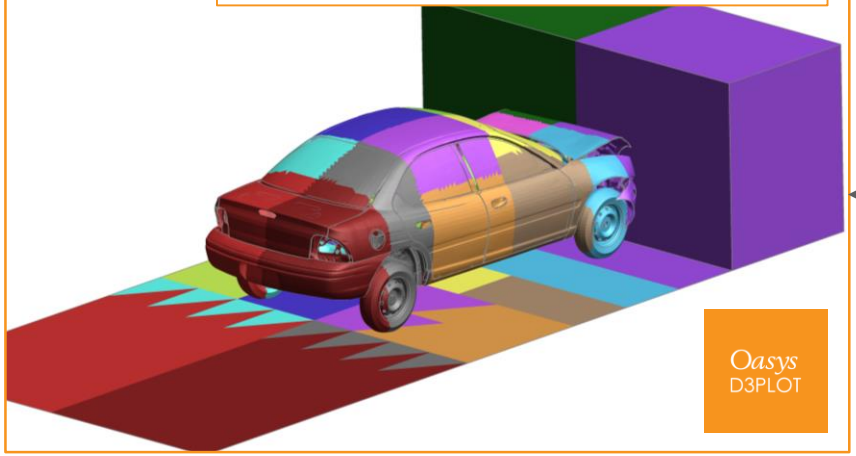
Output:
decomp_parts.ses

Oasys PRIMER

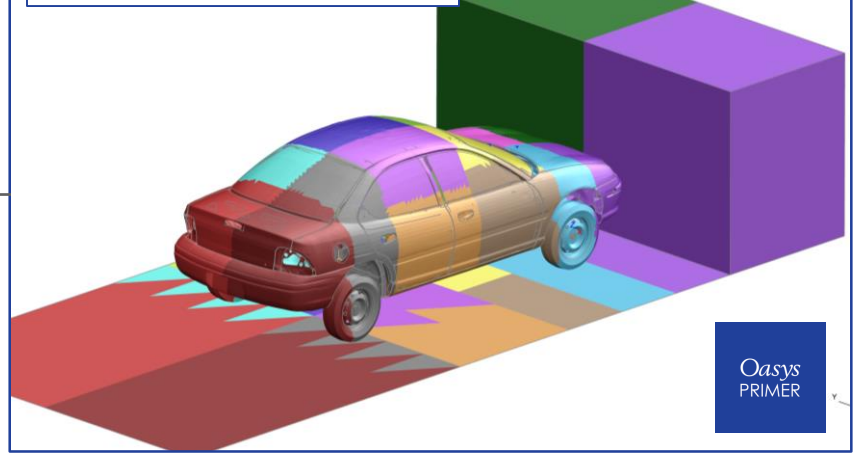
1. Tools > Script > decomposition_script.js.
2. Press Run.
3. PRIMER should compile a user interface:



Read .ptf data and .prp file into D3PLOT



Write out property (.prp) file



Load balancing information

Information during execution

Whenever you have an LS-DYNA job running on a cluster node, you can access that node using SSH. One advantage of this is that it allows you to look at the processor core and memory usage reported by that specific system (compute node). This can useful for troubleshooting and for other purposes.

Here is an MPP LS-DYNA job running on our remote cluster vdgcls01:

```
vdgcls01 gmohamed 101% qstat -u gmohamed | grep neon.refined
223865.vdgcls01.global  gmohamed    dyna     neon.refined.rev  11314     2     16       --  2476:40:3 R  00:00:25
```

From that output, we can see that the job ID number is 223865:

```
vdgcls01 gmohamed 106% qstat -f 223865 | grep host
    exec_host = atrnode29/0-15
    submit_host = vdgcls01.global.arup.com
```

The LS-DYNA job is running on node atrnode29 and we've requested 16 (0-15) cores on that node. If we want to look at the processor core and memory utilisation on these nodes:

```
vdgcls01 gmohamed 108% ssh atrnode29
vdgcls01 gmohamed 108% top -u gmohamed
```

Load-balancing information

Information during execution

Use the ‘top’ command to check the available memory in the system

```
atrnode2 gmohamed 101% top -u gmohamed
top - 14:55:08 up 65 days, 23:17,  1 user,  load average: 15.44, 9.90, 7.51
Tasks: 600 total,  17 running, 583 sleeping,   0 stopped,   0 zombie
Cpu(s): 66.7%us,  0.1%sy,  0.0%ni, 33.2%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  132030628k total, 93124324k used, 38906304k free,   162844k buffers

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
11603 gmohamed  20   0  761m 516m  17m R 100.0  1.4   3:22.29 ls-dyna_mpp_s_R
11604 gmohamed  20   0  680m 428m  13m R 100.0  1.3   3:24.08 ls-dyna_mpp_s_R
11607 gmohamed  20   0  701m 449m  12m R 100.0  1.3   3:23.74 ls-dyna_mpp_s_R
11610 gmohamed  20   0  704m 452m  12m R 100.0  1.4   3:23.84 ls-dyna_mpp_s_R
11615 gmohamed  20   0  701m 450m  13m R 100.0  1.3   3:23.71 ls-dyna_mpp_s_R
11605 gmohamed  20   0  677m 426m  12m R 99.4  80.3   3:23.93 ls-dyna_mpp_s_R
11606 gmohamed  20   0  681m 428m  13m R 99.4  1.3   3:23.82 ls-dyna_mpp_s_R
11608 gmohamed  20   0  687m 436m  13m R 99.4  1.3   3:24.08 ls-dyna_mpp_s_R
11609 gmohamed  20   0  686m 434m  13m R 99.4  1.3   3:23.95 ls-dyna_mpp_s_R
11611 gmohamed  20   0  702m 451m  13m R 99.4  1.3   3:23.73 ls-dyna_mpp_s_R
11612 gmohamed  20   0  698m 447m  12m R 99.4  1.3   3:24.06 ls-dyna_mpp_s_R
11613 gmohamed  20   0  704m 456m  13m R 99.4  1.4   3:23.82 ls-dyna_mpp_s_R
11614 gmohamed  20   0  702m 454m  13m R 99.4  1.4   3:23.96 ls-dyna_mpp_s_R
11616 gmohamed  20   0  699m 448m  12m R 99.4  1.3   3:24.05 ls-dyna_mpp_s_R
11617 gmohamed  20   0  704m 456m  13m R 99.4  1.4   3:23.69 ls-dyna_mpp_s_R
11618 gmohamed  20   0  709m 461m  13m R 99.4  1.4   3:23.92 ls-dyna_mpp_s_R
```

Load-balancing information

Information during execution

Use the ‘top’ command to check the available memory in the system.

You DO NOT want your job using swap space.
Often characterised by a job appear to be hanging for prolonged periods of time.
See d3hsp/otf extract below:

```
  23149 t 2.5000E-02 dt 1.08E-06 write d3plot file            10/26/18 11:23:16
 node  number       7348 deleted at time 2.50020E-02     shell element      7120  failed at time  2.5002E-02
shell element        7154  failed at time  2.5002E-02     shell element      7201  failed at time  2.5016E-02
shell element        7458  failed at time  2.5032E-02     shell element      1480  failed at time  2.5037E-02
shell element        1504  failed at time  2.5041E-02     shell element      7288  failed at time  2.5048E-02
shell element        9232  failed at time  2.5051E-02     shell element      7107  failed at time  2.5059E-02
shell element       25408  failed at time  2.5068E-02     shell element      9259  failed at time  2.5070E-02
shell element        7227  failed at time  2.5077E-02     shell element     20330  failed at time  2.5078E-02
shell element        9181  failed at time  2.5095E-02     shell element      7100  failed at time  2.5096E-02
shell element        7103  failed at time  2.5096E-02     shell element      7108  failed at time  2.5096E-02
shell element        7125  failed at time  2.5096E-02     shell element      7130  failed at time  2.5097E-02
shell element        7255  failed at time  2.5098E-02     shell element      7277  failed at time  2.5101E-02
shell element        7310  failed at time  2.5101E-02     shell element      7122  failed at time  2.5104E-02
shell element       25792  failed at time  2.5107E-02     shell element      1526  failed at time  2.5112E-02
 24075 t 2.6000E-02 dt 1.08E-06 write d3plot file            10/26/18 12:24:11
```

Load-balancing information


```
T i m i n g   i n f o r m a t i o n
                        CPU(seconds)    %CPU  Clock(seconds) %Clock
  ------------------------------------------------------------------
  Keyword Processing ... 1.9453E+00    0.04     1.9692E+00    0.04
  MPP Decomposition .... 2.5644E+00    0.05     2.6758E+00    0.05
    Init Proc ......... 9.8150E-01    0.02     9.8608E-01    0.02
    Decomposition ..... 1.1864E+00    0.02     1.1884E+00    0.02
    Translation ....... 3.9653E-01    0.01     5.0131E-01    0.01
  Initialization ..... 9.8191E-01    0.02     1.0404E+00    0.02
    Init Proc Phase 1 .. 6.9641E-01    0.01     7.1895E-01    0.01
    Init Proc Phase 2 .. 6.5679E-02    0.00     8.5949E-02    0.00
  Element processing ... 4.4645E+02    8.41     4.4728E+02    8.40
    Shells ............ 4.4602E+02    8.40     4.4672E+02    8.39
  Binary databases ..... 5.7473E+01    1.08     5.7575E+01    1.08
  ASCII database ....... 1.2730E+00    0.02     1.5664E+00    0.03
  Contact algorithm .... 7.8894E+02   14.86     7.9040E+02   14.85
    Interf. ID       1 4.8851E+02    9.20     4.8940E+02    9.20
    Interf. ID       2 1.3022E+02    2.45     1.3048E+02    2.45
    Interf. ID    3000 3.8772E+01    0.73     3.8860E+01    0.73
    Interf. ID    3001 8.4251E+01    1.59     8.4395E+01    1.59
    Interf. ID    3002 4.4200E+01    0.83     4.4283E+01    0.83
  Particle Algorithm ... 3.8919E+03   73.28     3.8994E+03   73.27
    P-structure ....... 2.5996E+03   48.95     2.6048E+03   48.95
    Particle collision . 1.2921E+03   24.33     1.2945E+03   24.32
  Rigid Bodies ......... 1.3857E+01    0.26     1.3838E+01    0.26
  Time step size ....... 1.6579E+01    0.31     1.6568E+01    0.31
  Rigid wall ........... 1.0820E-01    0.00     9.6764E-02    0.00
  Group force file ..... 1.1662E-01    0.00     1.3421E-01    0.00
  Others ............... 4.8768E+01    0.92     4.8825E+01    0.92
  Misc. 1 .............. 1.2468E+01    0.23     1.2602E+01    0.24
  Misc. 2 .............. 4.0869E-01    0.01     6.0941E-01    0.01
  Misc. 3 .............. 1.7122E+01    0.32     1.7227E+01    0.32
  Misc. 4 .............. 9.9219E+00    0.19     9.9172E+00    0.19
  ------------------------------------------------------------------
  T o t a l s           5.3108E+03  100.00     5.3218E+03  100.00

 Problem time       =    1.0000E-01
 Problem cycle      =    111112
 Total CPU time     =      5311 seconds (  1 hours 28 minutes 31
seconds)
 CPU time per zone cycle  =       396 nanoseconds
 Clock time per zone cycle=       397 nanoseconds
```

Run time data taken from the d3hsp file as shown:

Element processing – material/element calculation

Contact algorithm – all contacts in the model

P-structure – particle-to-fabric collision calculation

Particle collision – particle-to-particle collision calculation

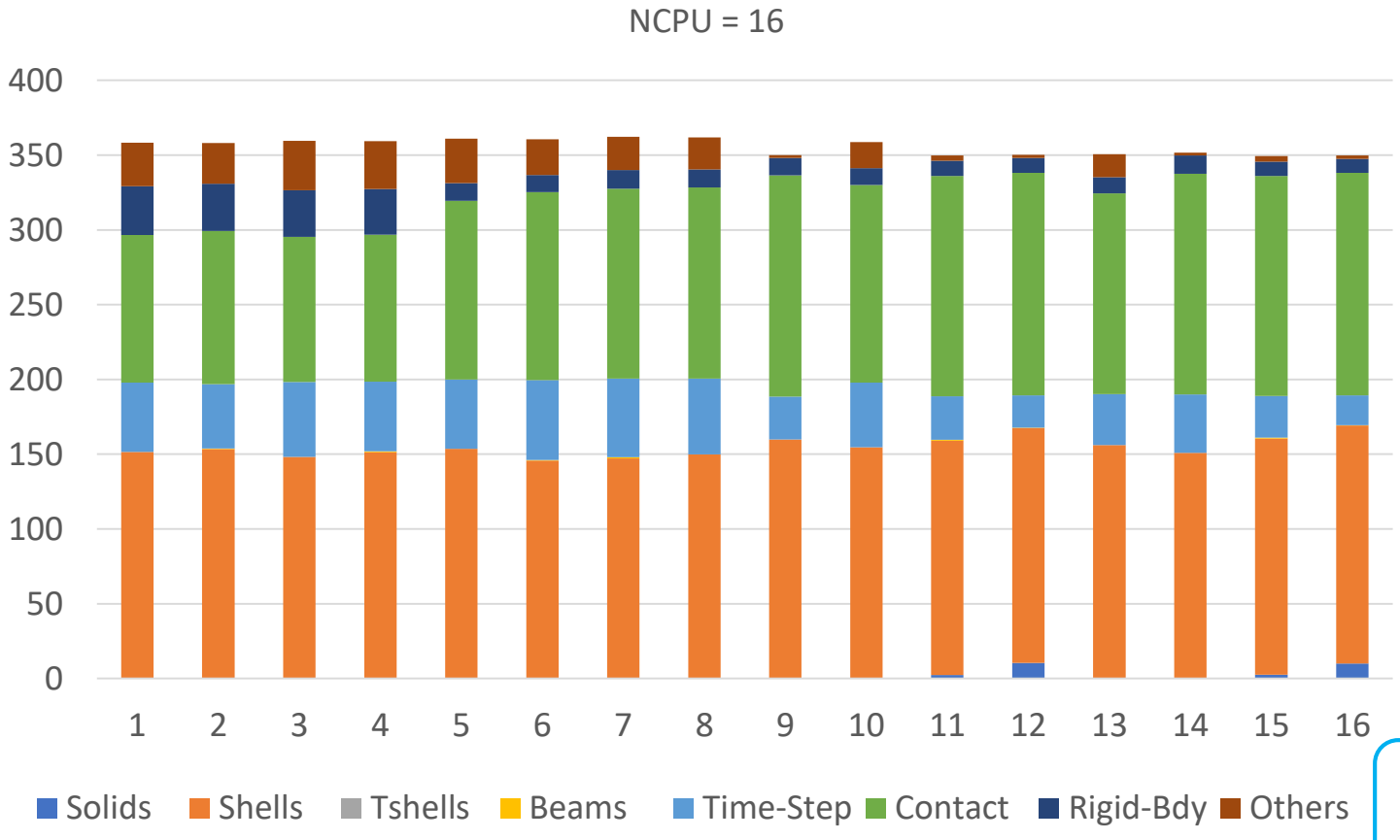
T o t a l s – total analysis run time (clock time)

Elapsed Time

Load-balancing information

Information after execution

- Processor load balance can be found in the '*load_profile.csv*' file

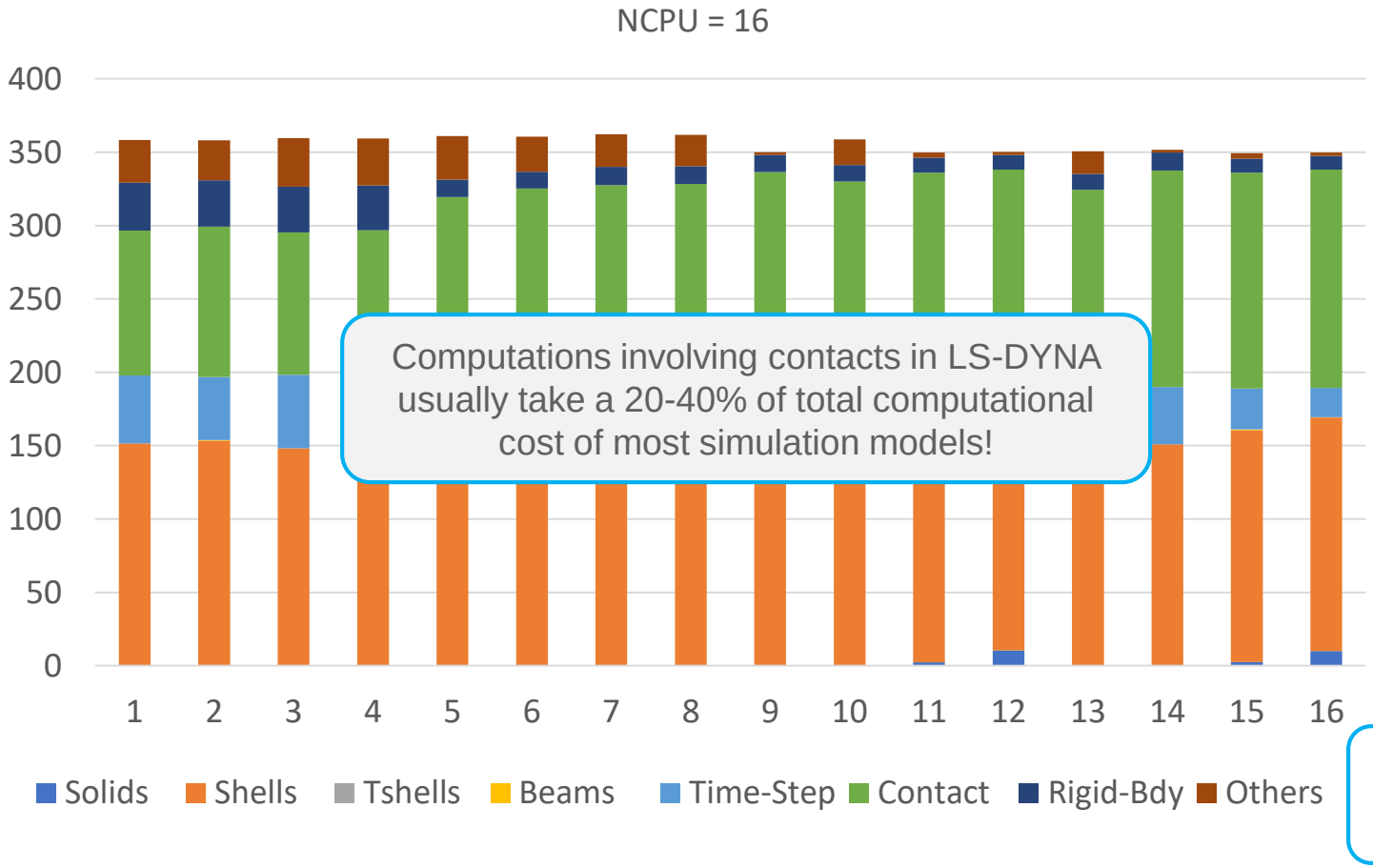


Y-axis: time (s)
X-axis: processor ID

Load-balancing information

Information after execution

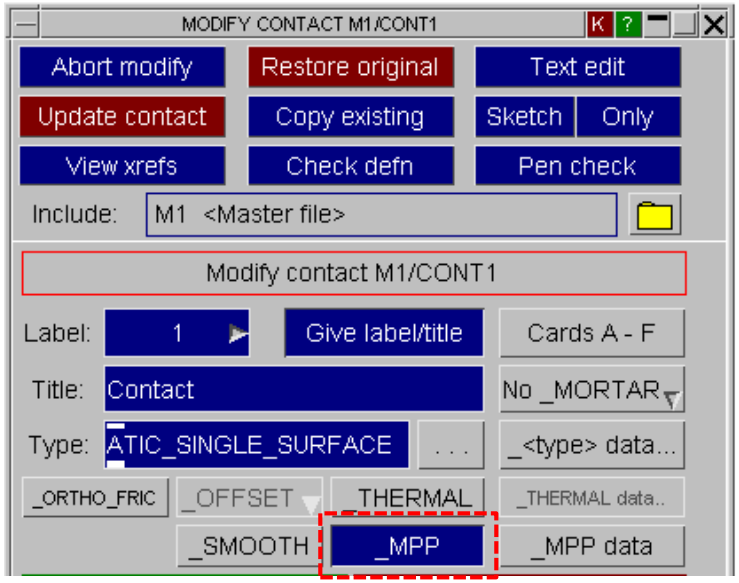
- Processor load balance can be found in the '*load_profile.csv*' file

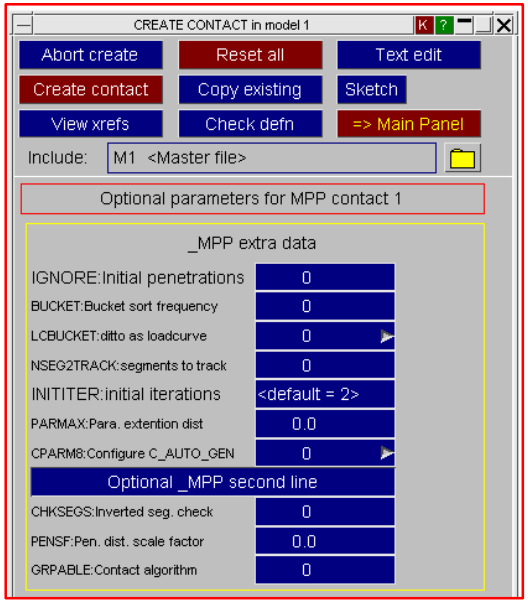


In simple terms, contacts comes down to the problem of comparing a single node *N* with a single segment *S* and applying forces as necessary to ensure *N* does not pass through *S*.

Problem:

- Complex models may contain several contact interfaces.
- The decomposition of such models can result in an uneven distribution of these contacts among the available processors.
- Some processors may have many contacts to handle, and others may have none. This variability can cause inefficiencies which adversely impact scalability.





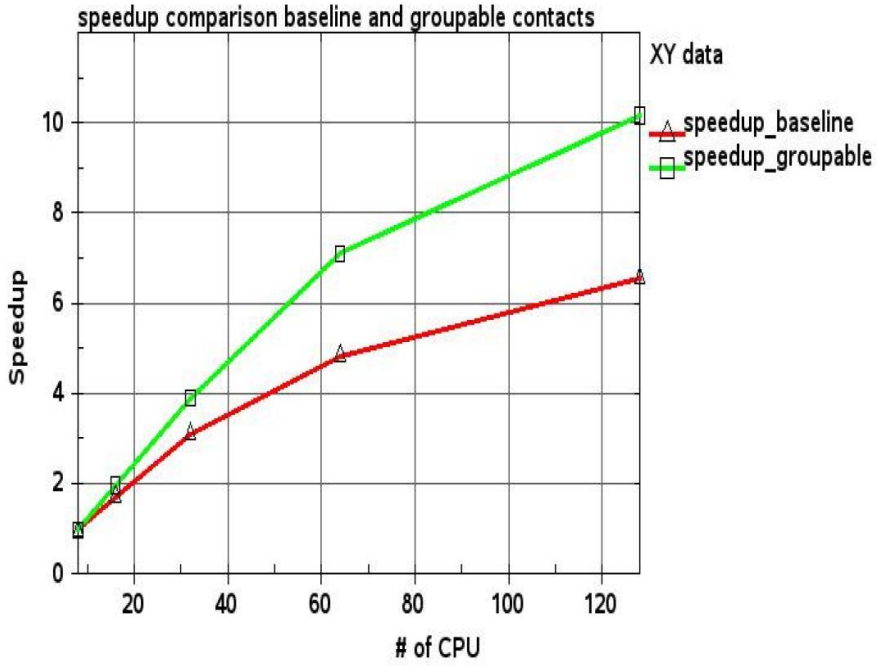
- IGNORE
- BUCKET
- LBUCKET
- NSEG2TRACK
- INITITER
- PARMAX
- **CPARM8**
- CHKSEGS
- PENSF
- **GRPABLE**

MPP Contacts: Groupable

- It is generally recommended to have as few contact definitions as possible – convert multiple contact definitions to a single surface with force transducers (*CONTACT_FORCE_TRANSDUCERS) and ‘FTALL = 1’ in *CONTROL_CONTACT.
- Where this approach is not desired by the user, the _GROUPABLE contact option is available.
- With this option turned on, contact definitions are internally combined in LS-DYNA to reduce communication.

Test conditions*
- 75 S2S contacts
- 1151856 nodes
- 1116160 shell elements

The “groupable” contact option is available for the following contact types:
- SINGLE_SURFACE
- NODE_TO_SURFACE
- SURFACE_TO_SURFACE
- ONE_WAY_SURFACE_TO_SURFACE
- AUTOMATIC_TIEBREAK
- TIED_*



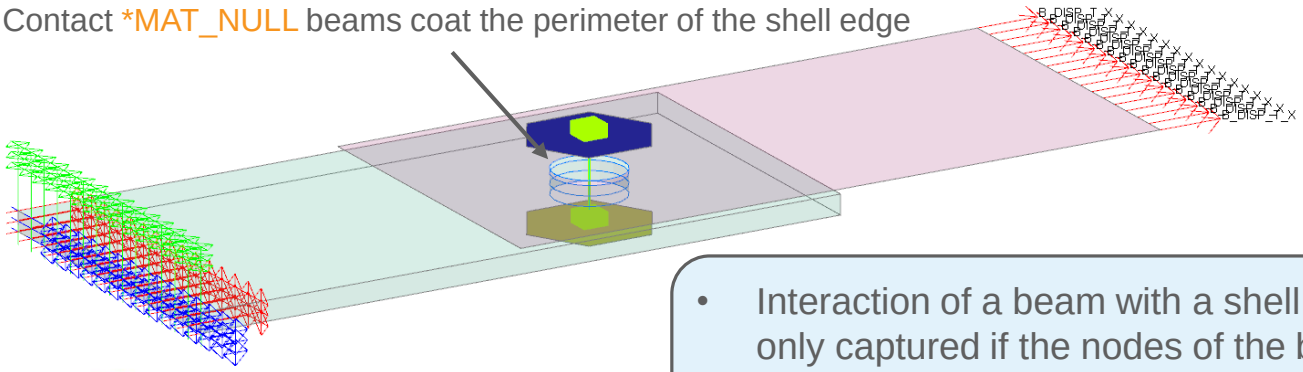
*MPP Class Handouts, Jason Wang, 2010

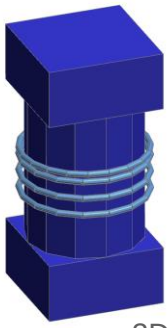
CPARM8:

1. Exclude beam to beam contact from the same part ID. This is for *CONTACT_AUTOMATIC_GENERAL.
2. Consider Spotweld beams in contact.

Example: Beam element for shank and shell elements for nut and head

Contact *MAT_NULL beams coat the perimeter of the shell edge





3D section view

- Interaction of a beam with a shell segment or a solid surface is only captured if the nodes of the beam get in contact with the shell segment.
- If nodes lay outside the segment, the contact is not taken into account.
- Deficiency is overcome by defining the numerically more expensive *CONTACT_AUTOMATIC_GENERAL_MPP.
- Spotweld beams (ELFORM = 9) are special beams and contact treatment is only permitted if CPARM8 =2

Parameters are set in a hierarchy:

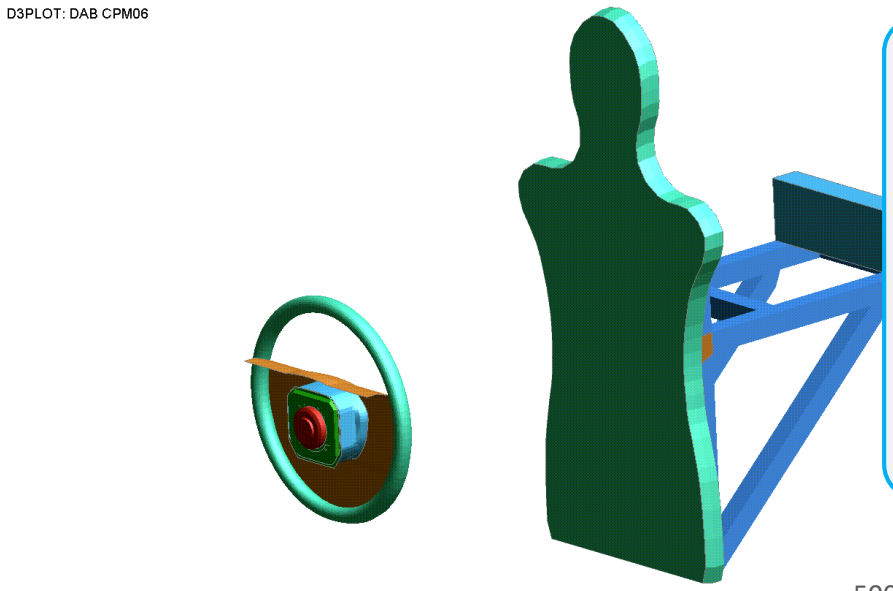
1. *CONTROL_CONTACT sets overall defaults,
2. *CONTACT_… sets parameters for that contact – *overrides 1 above*,
3. *PART_CONTACT sets parameters for that part in the contact – *overrides 1 & 2 above*.

Important parameters are:

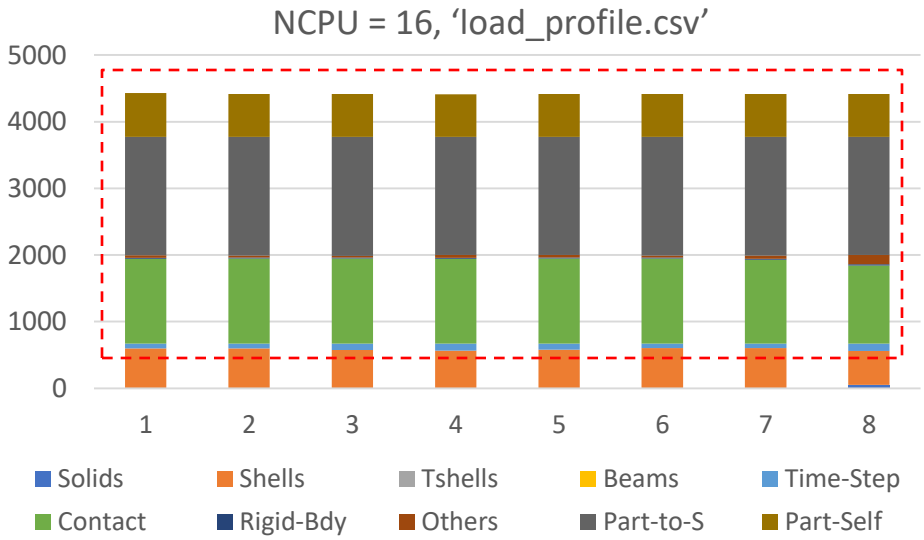
- Contact thickness and scaling factors
- Penalty stiffness scale factors or **SOFT** options
- Friction values
- Treatment of initial penetrations with **IGNORE** option

- Note: LS-DYNA will use the MIN(*SECTION_SHELL thickness, 0.4*element_length) for all parts unless *CONTROL_CONTACT SSTHK (=1) or *PART_CONTACT OPTT is activated.

Notes on Airbags



- In airbag analysis, the particle-to-structure collision calculation is the largest consumer of CPU.
  - Most affected by interconnect bottleneck
- Tasks that require a lot of processor communication have the worst scalability:
  - Particle-to-structure
  - Particle-to-particle contact

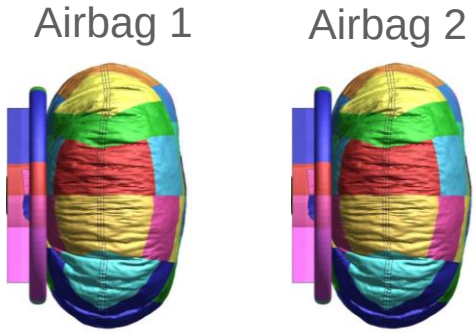


- Deployment and correlation of airbags can be sensitive to decomposition:
  *CONTROL_MPP_DECOMPOSITION_BAGREF
  *CONTROL_MPP_DECOMPOSITION_PARTS
  *CONTROL_MPP_DECOMPOSITION_PARTS_DISTRITBUTE

Notes on Airbag: Recommendations

- Users should benchmark their cluster performance and use the fastest processors per node (PPN) where possible.
- Run time of multiple airbags can be reduced by separating the CPU allocated to each CPM airbag using NPROC and FRSTP:
  - *CONTROL_MPP_DECOMPOSITION_ARRANGE_PARTS_{*OPTION*}
  - Users should try different ratios of NPROC based on the airbag requirements.
- To improve consistency, users should ensure airbag models are developed using a certain number of processors (NPROC), which is fixed throughout the life of the airbag model.

Decomposition procedure for a model with two CPM bags



1. Run model with airbag 1 active and airbag 2 removed.
   - Make a note of total time spent on CPM airbag 1 activity.
2. Run model with airbag 2 active and airbag 1 removed.
   - Make a note of total time spent on CPM airbag 2 activity.
3. Use NPROC to distribute the total number of CPUs between the two airbags in the same ratio as the time spent on each bag individually.

Multiple airbags: NPROC

*CONTROL_MPP_DECOMPOSITION_ARRANGE_PARTS_OPTION

Purpose: Allow users to distribute certain part(s) to all processors or to isolate certain part(s) in a single processor. This keyword supports multiple entries. Each entry is to be processed as a separate region for decomposition.

When this keyword is part of an included file and the LOCAL option is given, the decomposition will be done in the coordinate system of the included file, which be different from the global system, if the file is included using the *INCLUDE_TRANSFORM keyword

Card 1	1	2	3	4
Variable	ID	TYPE	NPROC	FRSTP
Type	I	I	I	I
Default	none	none	None	None

VARIABLE	DESCRIPTION
ID	Part ID/Part set ID
TYPE	EQ.0: Part ID to be distributed to all processors EQ.1: Part Set ID to be distributed to all processors EQ.10: Part ID to be lumped into one processor EQ.11: Part Set ID to be lumped into one processor.
NPROC	Used only for TYPE equal to 0 or 1. Evenly distributed Part ID/Part set ID to NPROC of processors.
FRSTP	Used only for TYPE equal to 0 or 1. Starting MPP rank ID.

MPP General Guidelines

- For consistency, use *CONTROL_MPP_IO_LSTC_REDUCE and _RCBLOG
- Merge small contact definitions into big ones (and use *CONTACT_FORCE_TRANSDUCERS for output).

- To improve load balancing, contact definitions should be equally distributed to all processors, use _GROUPABLE to handle smaller contact definitions.

- Distribute large contact area evenly among processor via pfile:
  - `decomp { silist 1,2,3}`
  - or in input deck
    *CONTROL_MPP_DECOMPOSITION_CONTACT_DISTRIBUTE

- To isolate any given contact to a single processor, use
  - *CONTROL_MPP_DECOMPOSITION_CONTACT_ISOLATE

- SMP and MPP contact algorithms are implemented differently:
  - The use of *CONTACT_AUTOMATIC_SURFACE_TO_SURFACE with SOFT=2 will be the contact that will give most similar results between MPP-DYNA and SMP.

- An MPP restart requires binary restart files called `d3dump##` and `d3full##`, where ## is a number
- Binary dump files contain a complete record of the model (stress, strain, deformation etc.) at a particular point in time
- There are three classes of restarts:
  - Simple restart: No changes to input deck.

    ```
    mpirun mpp971 -np 16 r=d3dump09
    ```

  - Small restart: Few small changes permitted, e.g. change termination time

    ```
    mpirun mpp971 -np 16 i=small.key r=d3dump09
    ```

    - Small restart file required

  - Full restart: Make significant changes to model.

    ```
    mpirun mpp971 -np 16 i=full.key n=d3full09
    ```

    - Full restart input deck required
    - *STRESS_INITIALISATION keyword required

Restart dump files

d3dump files:

- Permanent restart files which **accumulate** throughout the analysis.
- By default, binary d3dump file written at normal termination or crash of a run.
- Output frequency controlled with *DATABASE_BINARY_D3DUMP
  - A new restart file is created after each interval, CYL, thus a family of dump files is created and numbered sequentially, e.g. d3dump01, d3dump02 etc.

These files can be quite large and care should be taken with the d3dump files not to create too many.

Can be suppressed in MPP LS-DYNA:

- *CONTROL_MPP_IO_NOD3DUMP,
- *CONTROL_MPP_IO_NODUMP,
- *CONTROL_MPP_IO_NOFULL.

Switch Sensors

Sense switches allow you to control the behaviour of an analysis while it is running.

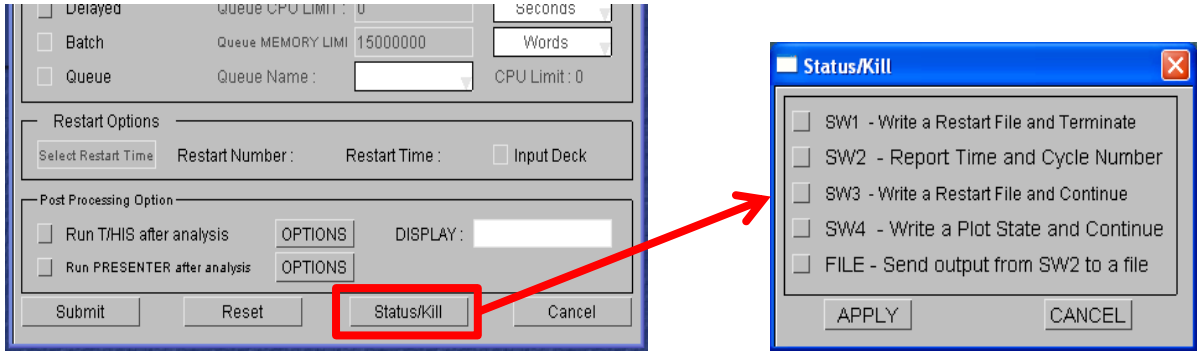
To activate one of the sense switch options a text file called *jobname*.kil needs to be created, containing one line of text, which is the sense switch you want to use:

e.g. sw1. (include the dot)

Some of the options are:

sw1. – A restart file is written and LS-DYNA terminates
sw2. – LS-DYNA responds with time & cycle info
sw3. – A restart file is written and LS-DYNA continues
sw4. – A plot state is written and LS-DYNA continues
swa. – Flush ASCII file buffers

The *jobname*.kil file can also be created using the Oasys SHELL:



Restart process

When restarting an LS-DYNA job:

- Use the same LS-DYNA executable as in the run that produced the dump file.
- Use same numbers of CPU's as in the run that produced the dump file.
- Use the same memory as in the run that produced the binary dump files.
- Run the analysis in the same directory.

Acknowledgements & References

- A Short course of LS-DYNA/MPP®, Jason Wang, 2010
- MPP Contact: Options and Recommendations, Brian Wainscott, 2015
- Appendix O: LS-DYNA MPP User Guide
- D3view blog (www.d3view.com), Suri Bala
- http://ftp.lstc.com/anonymous/outgoing/jday/restart.pdf

Contact Information

ARUP

www.arup.com/dyna

For more information please contact the following:

UK:

Arup

The Arup Campus
Blythe Valley Park
Solihull, West Midlands
B90 8AE
UK
T +44 (0)121 213 3399
F +44 (0)121 213 3302
dyna.support@arup.com

China:

Arup

39/F-41/F Huai Hai Plaza
Huai Hai Road (M)
Shanghai
China 200031

T +86 21 6126 2875
F +86 21 6126 2882
china.support@arup.com

India:

nHance Engineering Solutions Pvt. Ltd (Arup)

Ananth Info Park
HiTec City
Madhapur
Hyderabad - 500081
India
T +91 (0) 40 44369797 / 8
india.support@arup.com