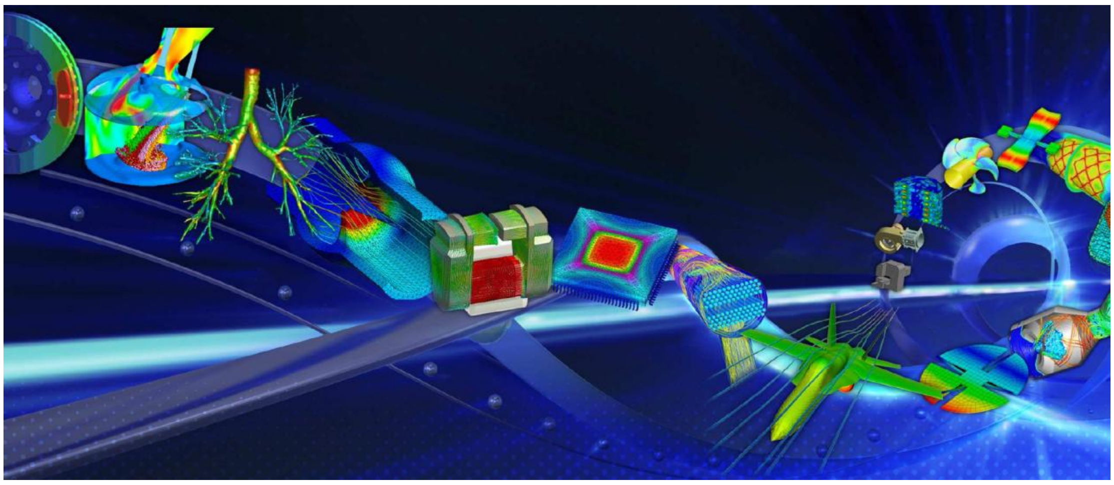


AUTODYN® Parallel Processing Tutorial Version 14.0

ANSYS, Inc.
Southpointe
275 Technology Drive
Canonsburg, PA 15317
ansysinfo@ansys.com
http://www.ansys.com
(T) 724-746-3304
(F) 724-514-9494

Release 14.0
November 2011

Table of Contents

Preface

AUTODYN Tutorial Manuals

AUTODYN tutorial manuals provide detailed tuition on particular features available in the program. The manuals assume that you are proficient in setting up, reviewing, executing, and post processing data for simple problems such as those presented in the AUTODYN-2D or AUTODYN-3D demonstration manuals. If you have worked through the problems in the demonstration manual, you should have no difficulty following the tutorials.
Most tutorials are interactive and you are expected to have access to AUTODYN while working through the tutorial. Some tutorials have associated files, which contain sample calculations used in the tutorial.
Existing manuals are continuously updated and new manuals are created as the need arises, so you should contact ANSYS if you cannot find the information that you need.

1. Introduction

AUTODYN is a general-purpose computer code that has been developed specifically for analyzing non-linear, dynamic events such as impacts and blast loading of structures and components. The program offers users a variety of numerical techniques with which to solve their problems. These include Lagrange, Shell, Euler, ALE (Arbitrary Lagrange Euler) and SPH (Smooth Particle Hydrodynamics) solvers. As reliance on computational simulations becomes accepted, the complexity of the problems to be solved increases in size and resolution. However, the practical computation of these very large simulations has been restrained by the lack of performance of available computers. Problems requiring millions of elements and run-times that can run many weeks are not uncommon. Even the fastest single CPUs cannot easily cope with these larger problems. One approach to overcoming these limitations is to utilize parallel systems. Parallel algorithms have been implemented in AUTODYN to take advantage of parallel systems that allow simultaneous use of multiple CPUs either on a single multi-processor machine or over a distributed network of computers. This tutorial describes the method used by AUTODYN to process problems in parallel and explains how the user sets up and runs a calculation using parallel processing. Currently AUTODYN supports the use of a maximum of 127 tasks (slave processes) in a single parallel analysis.

Users set up problems for parallel processing in exactly the same way as they do for serial processing, and the processing of results (plotting, saving, etc.) are also performed in the usual way.

At the current time, the structured as well as unstructured Part calculations of all 3D solvers have been parallelized:

- Lagrange (with joins)
- ALE (with joins)
- Shell (with joins)
- SPH (with joins)
- Euler Ideal Gas
- Euler Multi-Material
- Beams (with joins)

Note:

- Although the joins between unstructured parts have been parallelized it is strongly advised to avoid the use of unstructured parts in combination with joins in parallel analyses.
  In AUTODYN a new option is available that will merge joined unstructured nodes that reside at the same physical location in the model into one single unstructured node. The option is available under the Join menu and will increase robustness in many applications involving joins.

Interaction calculations have been parallelized between Lagrange, ALE, Shell and SPH solvers, using the Gap and Trajectory contact algorithm.

Coupled calculations have been parallelized between the Euler Ideal Gas and Euler multi-material solver and structured and unstructured Lagrange and Shell solvers.

Parallel simulations using Trajectory contact or bonded connections are only available as a beta option currently.
Simulations will run in parallel when trajectory contact is selected with the following restrictions:

- Will only run on one of 2,4,8,16,32,64 slaves
- Parallel efficiencies for models in which significant erosion takes place are likely to show low speed-ups compared with serial
- Models containing rigid bodies and / or bonded connections may have low parallel efficiencies
- Trajectory contact currently cannot be run in parallel when an SPH part is present in the model.

Improving the efficiency of parallel trajectory contact is an active area of development. Contact ANSYS for the latest updates on developments in this area.

2. Parallelization of Structured Parts, SPH, Unstructured Parts and Interaction

2.1. Parallelization of Structured Part Calculations

Domain Decomposition is used to parallelize the structured Part computations performed by all solvers which utilize an IJK mesh. Using this method, each part is divided along index planes in the I, J and K directions (J and K directions for Shell parts) to form smaller parts called sub-domains. At the present time, users must define the domain decomposition manually to be used for each problem they wish to run in parallel.

These sub-domains are distributed amongst the CPUs of the parallel machine using an algorithm that attempts to minimize inter-CPU communications and balance the computational load on each machine. Each sub-domain is processed in parallel as if it were a standard part in serial processing. This importantly allows most of the source code for serial processing to be used without modification.

Because the part structure does not normally change during simulations, a static decomposition of the entire index space is usually sufficient to achieve good parallel performance.

Efficient parallel processing of the Part calculations requires not only good load-balancing of the computation, but also the efficient exchange of data at sub-domain boundaries. This is best achieved by users clearly understanding the process involved and choosing their domain decompositions to suit the specific host configuration they intend to use.

Automatic decomposition can be used to decompose Euler-FCT Parts using the recursive bi-section method described below.

2.2. Parallelization of SPH Calculations

One of the main benefits of the SPH solver is its ability to evaluate the governing variables of a nonlinear dynamic analysis without utilizing a mesh or grid structure, thus allowing for very high levels of deformation. This benefit means that the domain cannot be decomposed by choosing grid lines and intersections as occurs with grid calculations.

In order to decompose the SPH calculation we utilize the virtual work units which are set up to facilitate the searching algorithm which is integral to the SPH solver. Each work unit is a cube and together they encompass the entire computational domain with a grid structure. The number of SPH nodes within a work unit is evaluated and a similar algorithm to that used to decompose the grid calculations is used to decompose the work units and therefore SPH nodes in order to minimize inter-processor communication. At present this is a static decomposition; each SPH node stays on the same processor for the length of the calculation.

2.3. Parallelization of Unstructured Calculations

As the numbering of nodes/elements of the unstructured solvers does not adhere to the same restrictions as the structured solvers, the parts cannot be simply decomposed by I, J, K index lines. The decomposition is done in a similar manner to SPH calculations. Virtual work units are applied to the model and the work done in each work unit is evaluated. A recursive bi-section method is then applied in order to effectively load balance the model. The amount of communication between processors is minimized but this is a secondary consideration compared to the load balancing. This method is completely automatic and requires a minimal amount of information from the user.

2.4. Parallelization of Contact Interactions

To understand how contact interactions are parallelized, we must first understand how the algorithm works in serial calculations.

Contact logic is used when one surface element of a part attempts to penetrate another surface element of the same or a different part. This requires a global search of Cartesian space to find possible contacts. Using the same sub-domain decomposition used for the grid calculation is not an efficient way to parallelize contact interactions. This is because the contact surfaces that need to be processed come from elements that lie on the surface of a part and thus comprise only a subset of the total elements in the part. It is easy to see that some sub-domains might contain many surface elements, while others none at all. Moreover, if a calculation allows the erosion or removal of elements as penetration occurs, the actual surface elements of a part will change as the calculation proceeds. We are therefore forced to use a second, dynamic domain decomposition of Cartesian space for the contact calculations.

Generally, any two surface elements anywhere in the simulation can come in contact with each other during some time step, even those that belong to the same object (self-interaction). Checking for all such contacts requires a global search in Cartesian space that in practice can take up to 50% of the overall CPU time. For efficiency, the contact nodes and faces are spatially sorted to speed this computation and to avoid unnecessary tests of distant elements. Thus, the contact algorithm used in AUTODYN can be considered in two parts. Firstly, a calculation is performed to identify neighboring nodes/faces that require to be checked for interaction. Secondly, a detailed interaction calculation is performed for all these identified nodes/faces.

Determining which nodes/faces require to be checked for interactions is achieved with a bucket-sort. A grid of virtual work units is defined in Cartesian space. Each work unit is a cube, with sides twice the smallest face dimension of all interacting faces. In tests, this cube size was found to not only yield the most efficient computing times (due to the fine sort), but also to generate sufficient work units to allow efficient load-balancing for the parallelization. These work units are virtual because storage for a particular work unit is only allocated when it is determined that the work unit contains nodes/faces that are to be tested for interaction.

The bucket-sort loops over all the surface nodes and faces of a problem, and constructs a list of the actual work units required for a particular analysis. The sort is performed in two passes, in which all the nodes are sorted, and then the faces are sorted. First, each surface node is added to the work unit, which contains it. Next, each surface face is added to all work units, which contain nodes that might interact with the face. This is determined by checking each node of the face to see if it is contained within, or is in close proximity to, a work unit's domain. At this stage, only work units that already contain surface nodes are considered. The proximity test is based on the size of the contact detection zone used for the interaction logic and the amount of "slack" allowed enabling the calculations described here to be performed less frequently than every cycle.

Finally, the node and face tables built for each work unit in the list are examined to determine the total number of node/face interactions that will be required to be computed for the work unit (this number is used to facilitate load-balancing in the parallel version). In general, this will equal the total number of nodes in the work unit times the total number of faces. However, this can be reduced if, for example, self-interaction of a part with itself is not permitted, or two parts have been specified not to interact with each other. If the total number of interactions required to be computed for a work unit is found to be zero, then the work unit is removed from the list.

At the end of this procedure a compact group of work units has been generated, each containing a list of surface nodes and faces that require testing for interaction. Each node has been uniquely assigned to a particular work unit. Faces have been assigned to multiple work units, as required. These lists may be valid for a number of computational cycles, depending on the proximity test used to determine potential node-face interactions and on changes in surface definitions (if an element is eroded or removed, surfaces need to be redefined).

Within each work unit, detailed interaction calculations are performed between the nodes and faces in each work unit list. The calculation is very robust in that every impact is detected and dealt with correctly regardless of the deformation and relative movement of bodies or changes in surface definitions

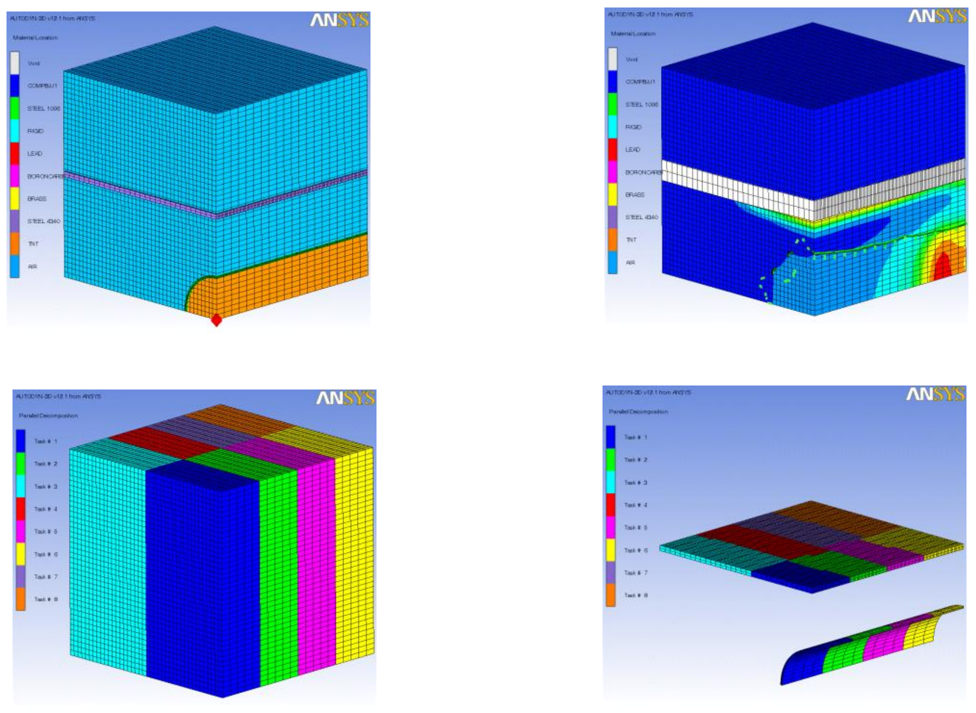
Parallelization of the contact algorithm described above is fairly straightforward. Once the work units containing the node and face lists to be tested for impact have been generated, a load-balancing algorithm efficiently distributes them amongst the available CPUs, assuming each CPU has either the same speed or a pre-determined relative speed provided by the user. The data decomposition used for the contact interaction calculation is different from the one used for the grid calculation, so the load-balancing algorithm attempts to minimize the inter-CPU communications required between these two decompositions. Although a static decomposition is used for the grid calculation, a dynamic decomposition has to be used for the contact calculation. Consequently, load balancing of the newly formed work units is performed for each cycle on which a sort is carried out. This allows contact calculations to remain well load-balanced even when surfaces are reconfigured as the simulation progresses, or during the erosion (removal) of elements.

Results have shown that the contact algorithm generates sufficient work units during the sort phase to allow efficient load balancing for parallel processing. Furthermore, the scheme uses simpler communication patterns than those that rely on recursive coordinate bisection (RCB) to assign equal amounts of nodes to all CPUs, and adapts well to heterogeneous systems where CPUs may have different speeds and workloads that may vary with time.

2.5. Parallelization of Euler-Lagrange Coupling Interactions

Aside from handling the decomposition of unstructured parts, the automatic parallel decomposition algorithm can also handle the decomposition of Euler parts, Euler Ideal Gas as well as Euler Multi-material. This facilitates the decomposition of complicated Euler/Lagrange coupled models; the user only needs to define the number of tasks over which the model should be assigned and AUTODYN will automatically produce a decomposition configuration with good load balancing qualities and minimal inter-processor communication.

To further enhance the efficiency of coupled calculations the sub-domains of the FE structure will be placed on the same processor as the Euler sub-domains located in the same geometric space. This decreases the necessary inter-processor communication for the coupling calculations.



Example: 8 CPU automatic decomposition of a casing fragmentation analysis.

Notes

- The multi-material Euler part should not be joined to other Euler Parts

3. Establishing a Parallel Processing Environment

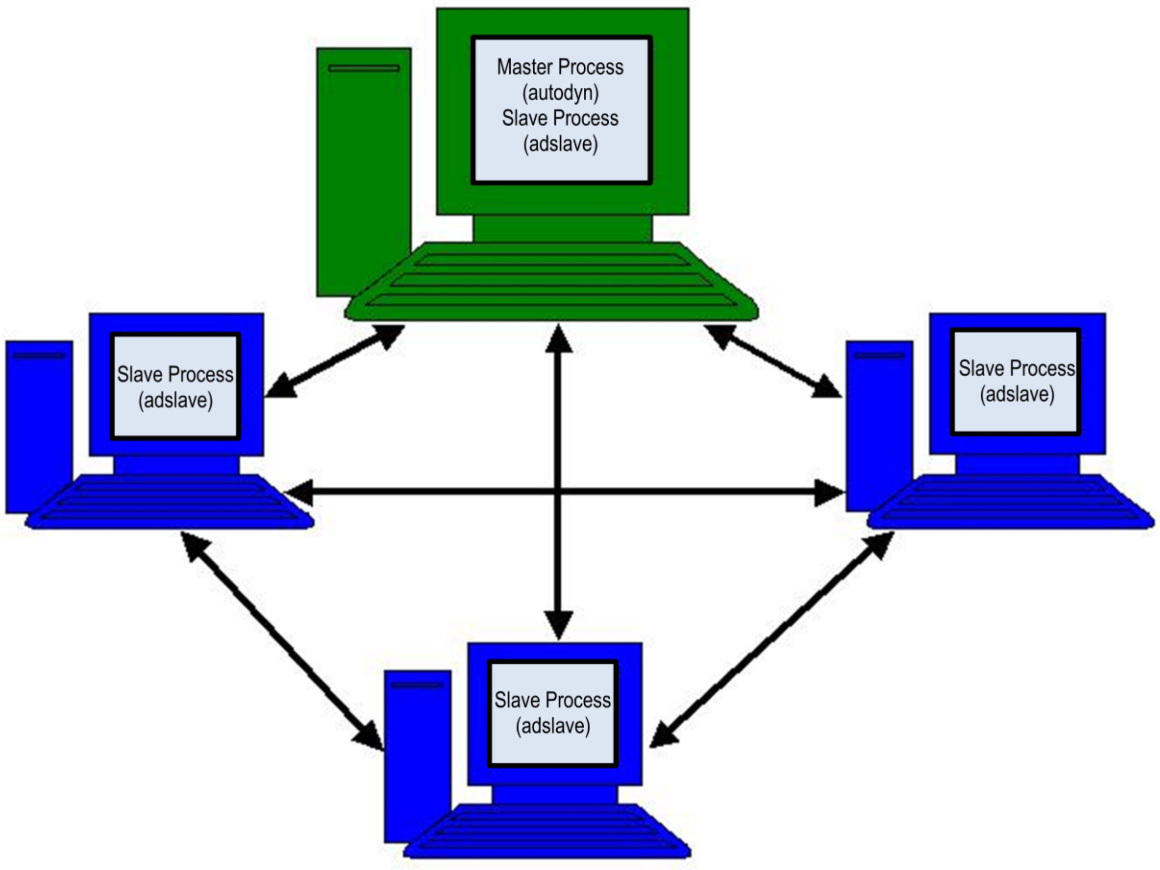
AUTODYN has been designed for parallel processing on a variety of systems ranging from a Massively Parallel Processor (MPP) using shared memory to heterogeneous distributed networks of computers.

In this chapter, we outline the procedure for establishing a parallel processing environment for AUTODYN. The procedure should be similar for most platforms. Platform specific procedures for configuring the parallel processing environment are given in referenced appendices.

When using AUTODYN for parallel processing, data must be exchanged between cooperating tasks, and some message passing protocol has to be used to achieve this. We currently use MPI (Message Passing Interface) to allow a heterogeneous collection of computers networked together to be viewed by AUTODYN as a single parallel computer. The component computers can be single or multiple processor machines, including MPPs (Massively Parallel Processors).

If you wish to use the parallel processing options described in this tutorial, you must install the Platform MPI message passing protocol on all the machines you intend to use.

Platform MPI is a part of the unified installation packages on both Linux and Windows platforms.



Schematic of the Parallel Processing Environment

Whether you intend to run a calculation in serial mode or parallel mode, AUTODYN is always started by activating a single process on one machine. This is done by starting the **autodyn** process.

If a serial calculation is being performed, this is the only process that is run. It performs all aspects of the problem (setup, execution, I/O, post processing etc.)

If a parallel calculation is being performed, this process is considered to be the **master process** and additional **slave processes** are started on each CPU used by the parallel system (including the CPU running the master process). These slave processes are spawned automatically by the master process, which issues the required system commands to start the **adslave** process once on each CPU. This parallel processing environment is shown schematically in the figure above.

In what follows, we assume that the configuration we are implementing comprises of **n** hosts, each having **m** CPU's, and that the hostname of the nth host is **hostn**. **Host1** is special, in that it runs the master process (**autodyn**) and is usually the system on which the program files are stored.

When running AUTODYN in parallel on a system of networked computers, the program executables and libraries must be accessible on each of the computers. They can either be installed on all computers in the system or installed on one computer (usually the one running the master process) and shared with the others using NFS mounting or Windows Sharing. We recommend sharing as it simplifies the handling of the files. Setup is further simplified if the shared mount points/folders are the same on each host (including the master).

3.1. Windows Systems using Platform MPI

AUTODYN is made available on Windows operating systems using Platform MPI as the parallel message passing protocol. Dynamic spawning of slave processes from the AUTODYN component system is not possible. Therefore the number of slave tasks is specified before starting the AUTODYN program using 'mpirun'. The process for this is outlined below. Once the AUTODYN executable is started the user would have to restart their AUTODYN session should they require a different number of slave tasks.

To establish the parallel environment Platform MPI must be installed on each machine in the cluster following the installation instructions for Platform MPI as provided in ANSYS online documentation. Users may find it necessary to set the MPI password on each machine in their cluster. To do this they can call mpirun with the '-cache' argument the first time they run mpirun, entering their user password when prompted. This argument should be removed for subsequent parallel runs.

The path environment variable should be updated to include the $MPI_ROOT\bin directory.

3.1.1. Running AUTODYN with MPIRUN

Create a file called 'applfile' in the C:\Documents and Settings\<username>\Application Data\Ansys\v140\AUTODYN directory. In this file enter the following text, making the relevant changes for your parallel model, machine names, and installation path should the installation not be in the default location:

```
-e MPI_FLAGS=y0 -h machine1  -np 1 "C:\Program Files\ANSYS Inc\v140\AISOL\AUTODYN\winx64\autodyn.exe"
-h machine1  -np 2 "C:\Program Files\ANSYS Inc\v140\AISOL\AUTODYN\winx64\adslave.exe"
-h machine2  -np 2 "C:\Program Files\ANSYS Inc\v140\AISOL\AUTODYN\winx64\adslave.exe"
```

The above applfile will launch 1 master process, 2 slaves on machine1 and 2 further slaves on machine2. If using Windows XP 32-bit winx64 should be replaced by Intel.

AUTODYN is then started using mpirun. To do this the user could call mpirun from a command prompt, but it is suggested to create a batch file in the same application data directory as the applfile file and execute that. For example, create a blank file called 'autodyn_mpi.bat'. In this file type:

```
  mpirun -e MPI_WORKDIR="C:\Program Files\ANSYS Inc\v140\AISOL\AUTODYN\winx64" -f applfile
```

The MPI_WORKDIR text should be modified if the user's installation of AUTODYN is not in the default location.

Once the batch file is executed the AUTODYN program starts along with the number of slaves specified in the applfile. The user can now create their model or load an existing one as usual.

3.2. Linux Systems using Platform MPI

In what follows, we assume that AUTODYN and Platform MPI files are installed on host1, in a directory named **/usr/ansys_inc/**, and that this directory is NFS mounted on all other hosts (with the same mount point). To NFS mount this directory, add the following line to the file **/etc/fstab** on each host (except **host1**):

```
host1: /usr/ansys_inc/    /usr/ansys_inc/   nfs   defaults   0   0
```

/etc/fstab may have a different name for other Linux systems

Following the normal installation procedures for AUTODYN products, all AUTODYN files will be installed in **/usr/ansys_inc/v140/autodyn/**. In particular, the AUTODYN master and slave executables (**autodyn** and **adslave**) will reside in the directory **/usr/ansys_inc/v140/autodyn/bin/<platform>,** where **<platform>** is one of: linia32, linop64, linem64t or linia64

After installing AUTODYN and Platform MPI on all the cluster machines, the environment variable MPI_ROOT can be set to be the directory where Platform MPI is installed and the path to the MPI_ROOT/bin directory added to the *path* environment variable. These changes are not required to run AUTODYN in parallel, but when set they will override the defaults used by AUTODYN.

3.2.1. Host Configuration Set to Run AUTODYN in Parallel

To run AUTODYN in parallel, you must define the host machine configuration you wish to use. AUTODYN allows you to define up to ten host configuration sets and select one of these sets as the active configuration.

Host configuration set data is saved in an external file called **"parallel.cfg"**, located in your running directory. This is a text file that must be edited manually.

Here is an example of the typical contents of this file.

```
1. #@EPDEF=\\host1\autodyn
2. #@PPDEF office
3. #@PPCFG office
4. host1 sp=1000
5. #@ mem=128 cpu=1 task=1
6. host2 sp=500 ep=\\host2\autodyn
7. #@ mem=256 cpu=2 task=2
```

Line 1: Defines the default path for the AUTODYN executable to be **\\host1\autodyn**. Where **autodyn** is a shared directory on **host1**.

Line 2: Defines the configuration set name that is to be used (**office**)

Line 3:	Indicates that the following lines, until the next #@**PPCFG** statement define configuration set **office**. (There is only one configuration set defined in this example)
Line 4:	Adds **host1** to configuration set “**office**”. It has a relative speed of **1000** and uses the default path for the AUTODYN executable (since no **ep** parameter is defined)
Line 5:	host1 has **128** Mb of memory, **1** CPU, and is to have **1** task (slave process) started on it.
Line 6:	Adds **host2** to configuration set “**office**”. It has a relative speed of **500** and **/autodyn** as the path for the AUTODYN executable.
Line 7:	Host2 has **256** Mb of memory, **2** CPUs, and is to have **2** tasks (slave processes) started on it. (The operating system will automatically allocate 1 task to each CPU).

parallel.cfg is an external file that is read whenever a problem is executed in AUTODYN. Host configuration sets are therefore not problem dependent (unlike the domain decomposition sets described in the next chapter). When you create or modify a host configuration set while working on a particular problem, that set becomes available for all problems.

When an AUTODYN analysis is started using the command script **autodyn130**, the host configuration set defined in the **parallel.cfg** file is read and fully automatic the applfile is generated and mpirun is started (similar as is done on Windows operating systems).

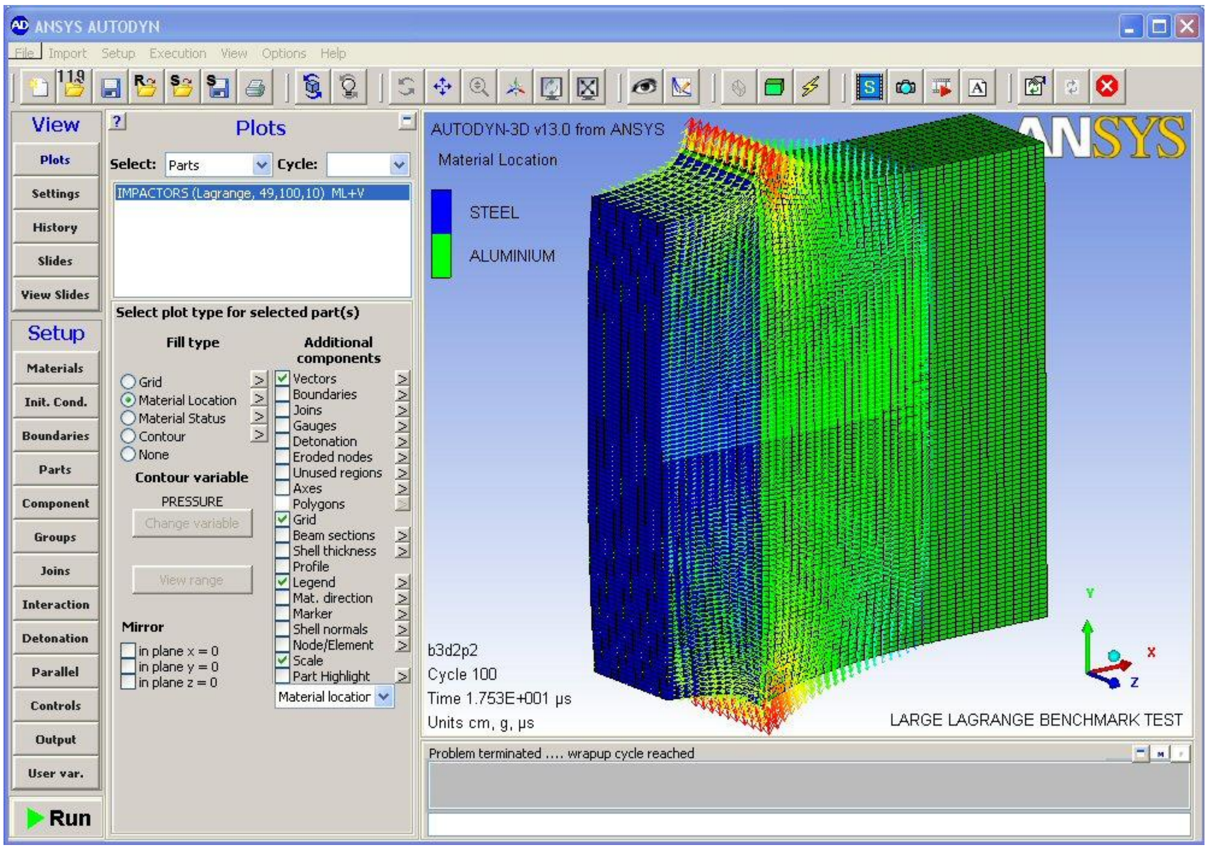
4. Creating Domain Decomposition Sets

To enable parallel processing the problem must be decomposed and the various sub-domains assigned to the available processors. AUTODYN offers two ways of decomposition generation:

- Manual
  The user creates manually domain decomposition sets that are associated with a particular model and these are stored along with all other problem data whenever the model is saved. This Chapter 4 describes how to manually setup domain decomposition sets for different models.
- Fully automatic.
  This option is only available for models containing any combination of Euler and unstructured solvers and no further input is required by the user in terms of model set-up. This option is described in Chapter 5.

4.1. Decomposing Structured Calculations

In the following example, you will create a domain decomposition set for a Lagrange impact benchmark calculation. Load in the model or ident **B3D2P2** in the samples directory of the Workbench installation which most probably is C:\Program Files\ANSYS Inc\v140\AISOL\Samples\Autodyn. Though this example specifically applies to a Lagrange process, the method shown is appropriate to decompose any grid calculation, including Euler processes.



Select: Parts

Cycle:

IMPACTORS (Lagrange, 49,100,10) ML+V

Fill type: Material Location

Additional components: Vectors ✓, Grid ✓, Legend ✓, Scale ✓

Material location

AUTODYN-3D v13.0 from ANSYS

Material Location

b3d2p2
Cycle 100
Time 1.753E+001 μs
Units cm, g, μs

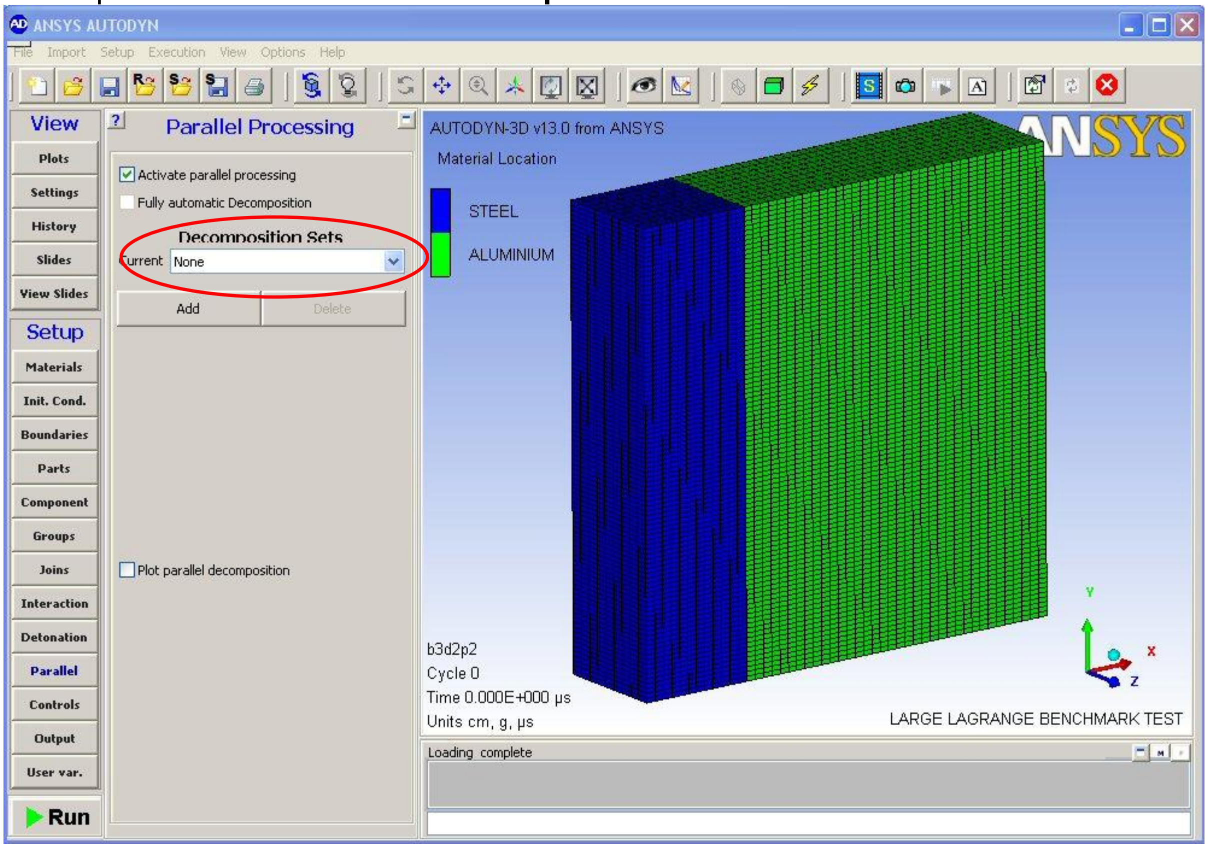
LARGE LAGRANGE BENCHMARK TEST

Problem terminated .... wrapup cycle reached

The problem consists of a single Lagrange part (49x100x10 cells) filled with two materials. The steel has an initial z-velocity of 0.06 cm/microsecond, while the aluminum is initially at rest.

If you run this problem for 100 cycles and then view a velocity vector plot, you will see the screen above, showing the impact of the steel on the aluminum.

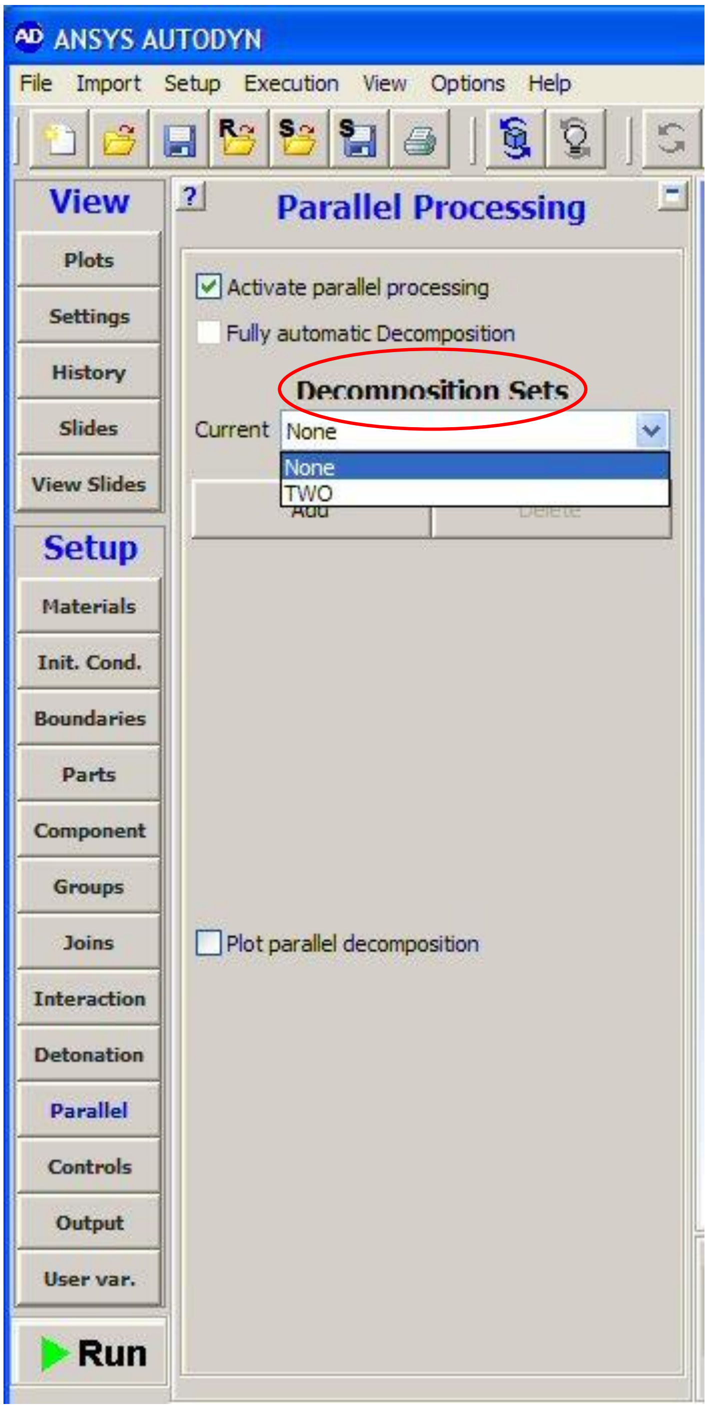
This model already contains **Decomposition Set** information. The current decomposition is shown in the **Decomposition Set** text box.



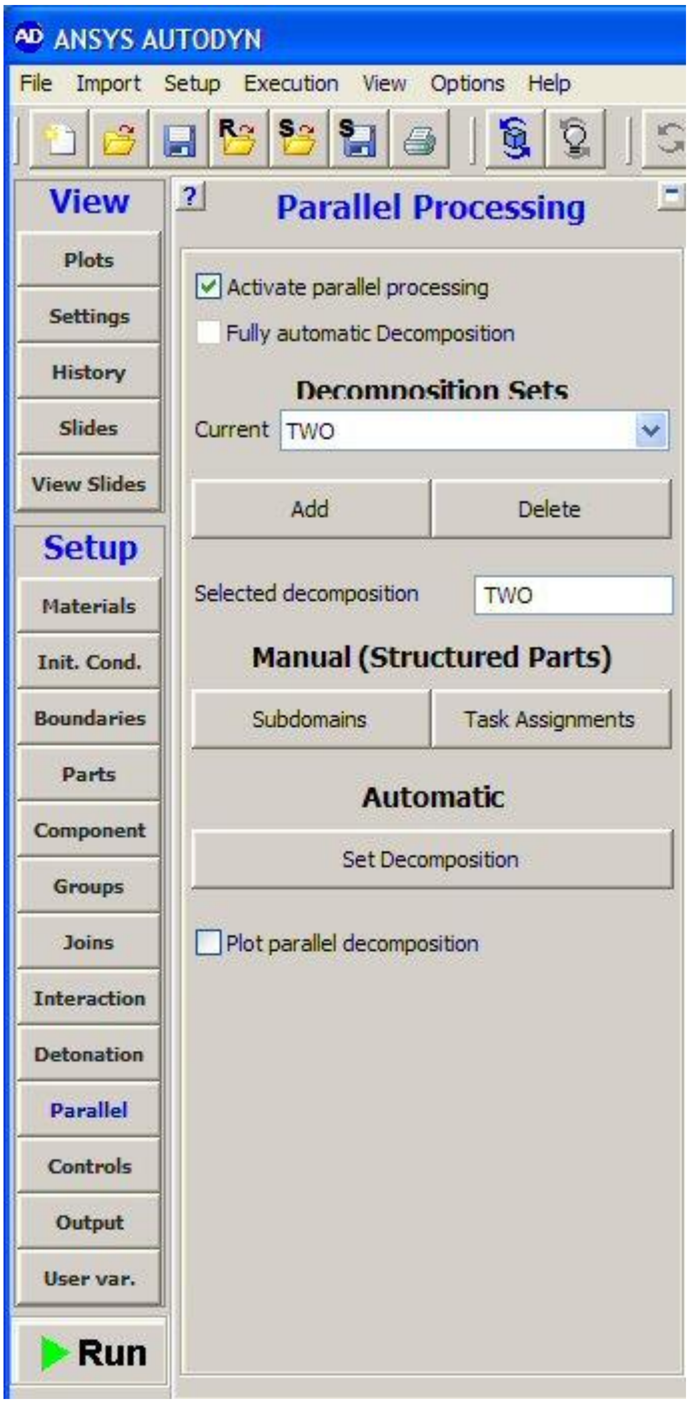
Activate parallel processing [✓]
Fully automatic Decomposition [ ]
Decomposition Sets
Current: None
Plot parallel decomposition [ ]

AUTODYN-3D v13.0 from ANSYS
Material Location
b3d2p2
Cycle 0
Time 0.000E+000 μs
Units cm, g, μs
LARGE LAGRANGE BENCHMARK TEST
Loading complete

Presently the current decomposition set is set to “**None**”. The current decomposition set can be changed by using the drop down list activated by pressing the list button on the right of the text box. The image below shows a close up of the **Parallel** panel.



Choose the decomposition set denoted by "**TWO**". Further buttons allowing the modification of the decomposition set will appear.



☑ Activate parallel processing

☐ Fully automatic Decomposition

Decomposition Sets

Current: TWO

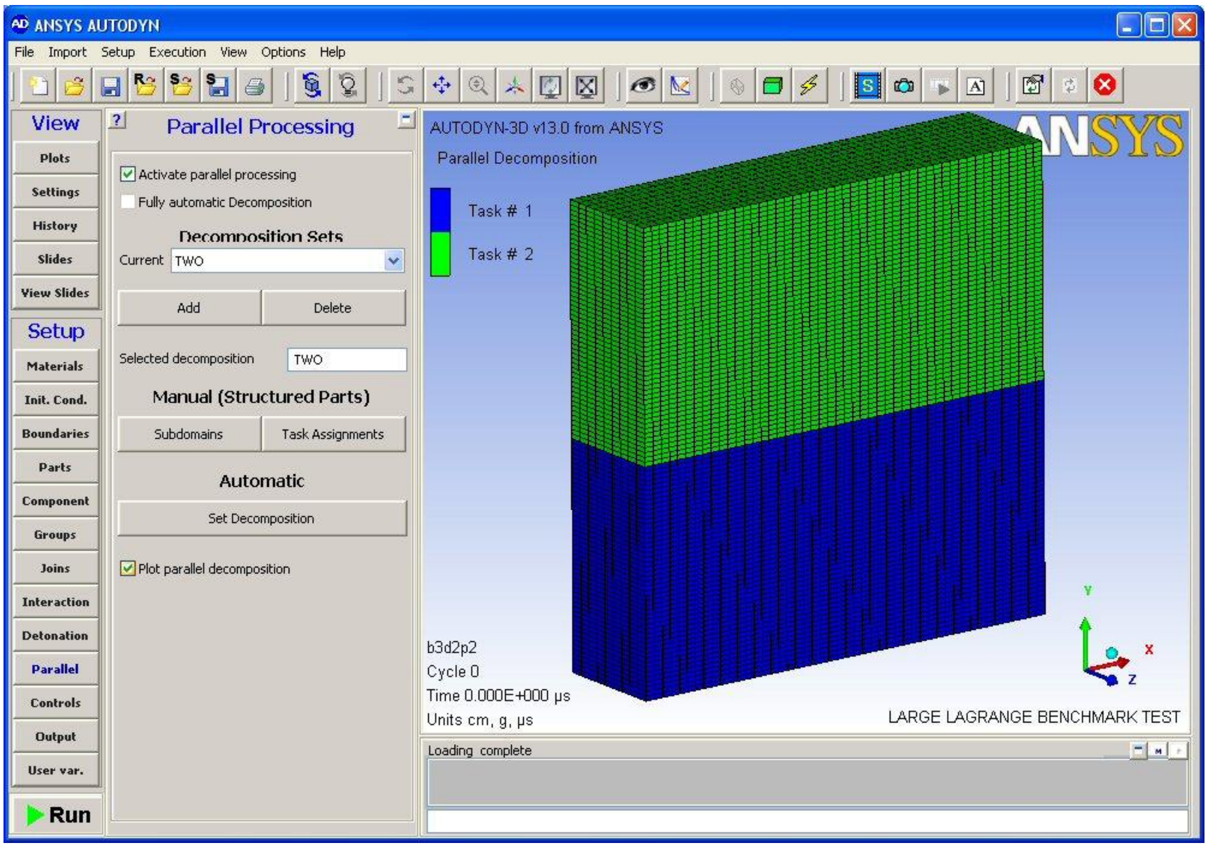
Selected decomposition: TWO

Manual (Structured Parts)

Automatic

☐ Plot parallel decomposition

Check the tick box marked **Plot parallel decomposition**. The decomposition of the model will now be displayed (you may need to manually refresh the screen if automatic refresh is not activated):



Activate parallel processing: ✓
Fully automatic Decomposition: ☐

Decomposition Sets
Current: TWO

Selected decomposition: TWO

Manual (Structured Parts)

Automatic

Plot parallel decomposition: ✓

Parallel Decomposition

🟦	Task # 1
🟩	Task # 2

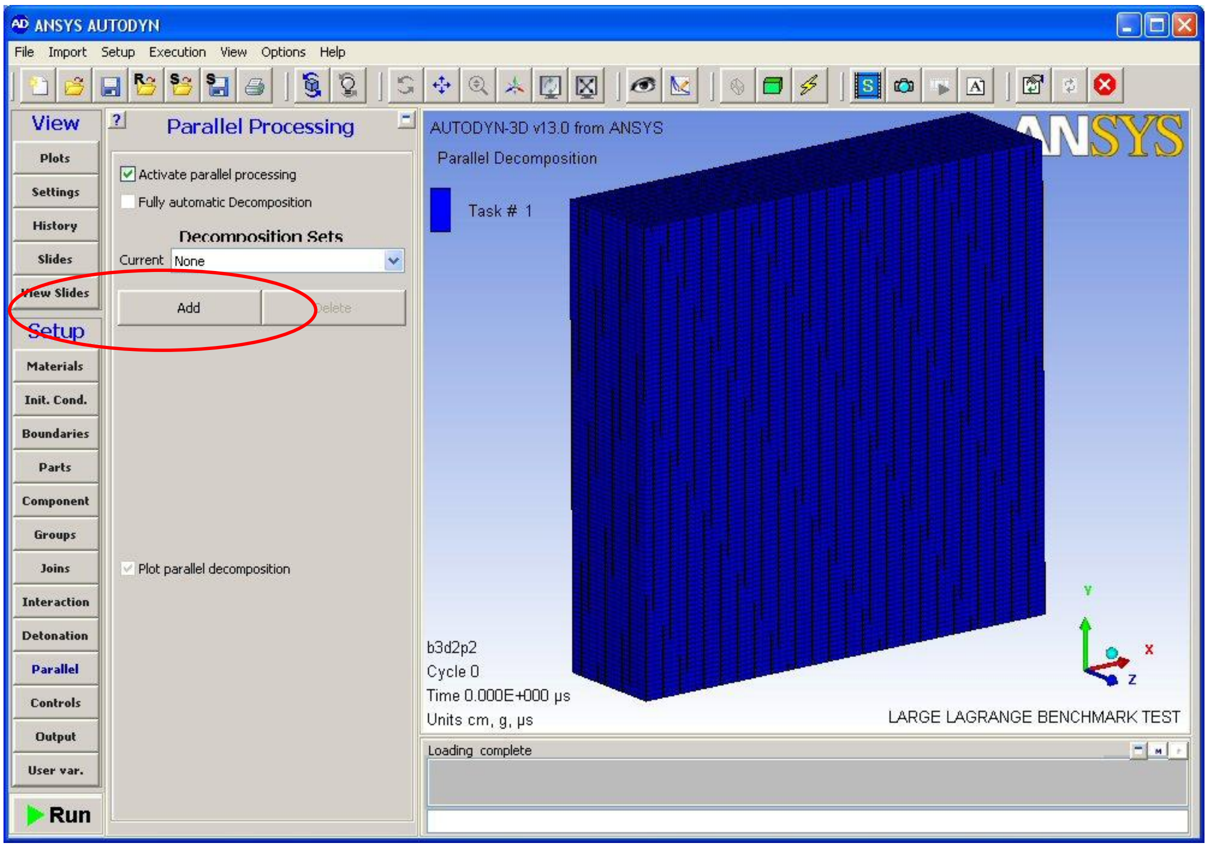
b3d2p2
Cycle 0
Time 0.000E+000 μs
Units cm, g, μs

Loading complete

This image shows the decomposition of the model into its constitutive sub-domains. The scale illustrates which sub-domain is assigned to which task. Here the bottom sub-domain is assigned to task 1 and the top sub-domain assigned to task 2. This is the optimum decomposition for efficient parallel computing over two tasks. The model has been decomposed in the J direction. As the J direction contains the most elements, this decomposition will result in the smallest amount of information being passed between tasks.
There are 490 elements on each sub-domain boundary; 49 in the I direction and 10 in the K direction. If we had decomposed the part in the I direction, there would have been 1000 elements on each sub-domain boundary; 100 in the J direction and 10 in the K direction. Obviously there is a computational overhead associated with the amount of information passed between tasks and therefore each element on the sub-domain boundaries increases the amount of communicated information and therefore computational expense.

Before proceeding press the **Delete** button to discard the “**TWO**” decomposition. The instructions to enable you to replace it are detailed below.

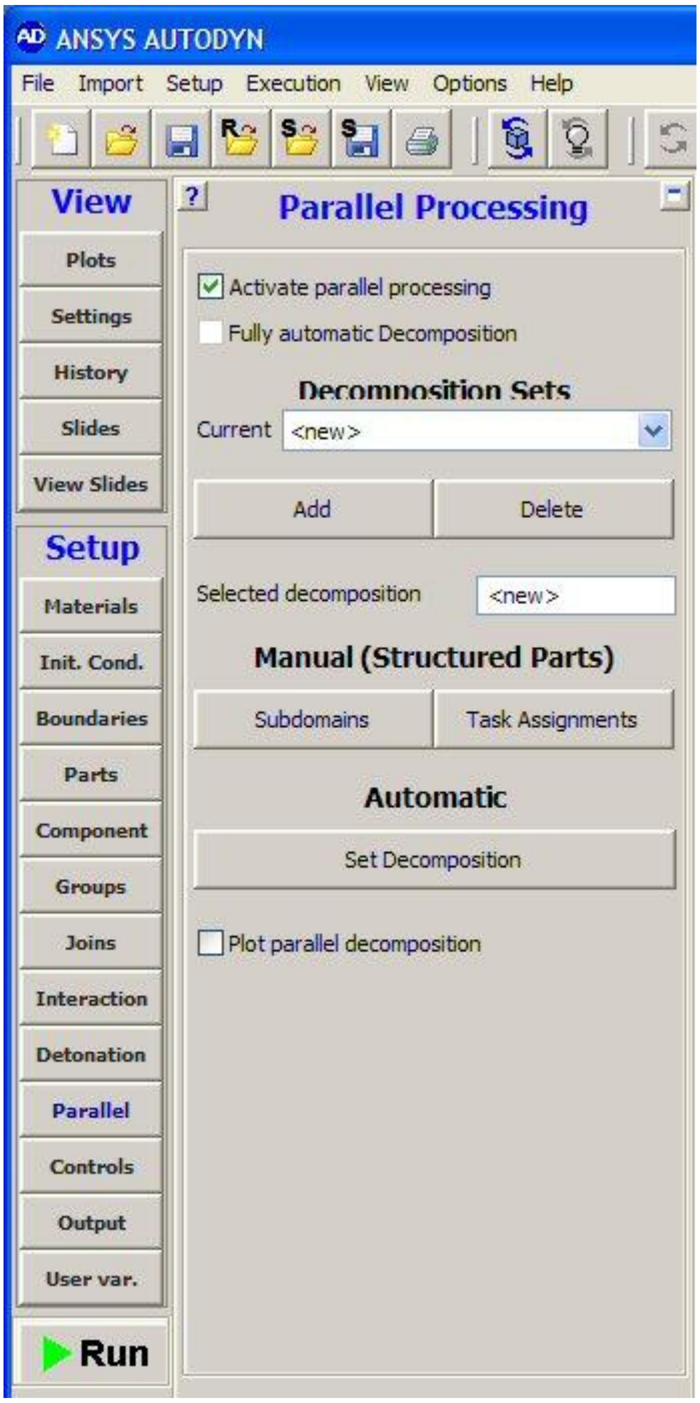
Press the **Add** button in the **Decomposition Set** section of the **Parallel** panel.



☑ Activate parallel processing
☐ Fully automatic Decomposition
Current: None
☑ Plot parallel decomposition

AUTODYN-3D v13.0 from ANSYS
Parallel Decomposition
b3d2p2
Cycle 0
Time 0.000E+000 μs
Units cm, g, μs
LARGE LAGRANGE BENCHMARK TEST
Loading complete

Additional buttons and information will then appear in the **Parallel** panel:



[✓] Activate parallel processing
[ ] Fully automatic Decomposition

Decomposition Sets

Current <new>

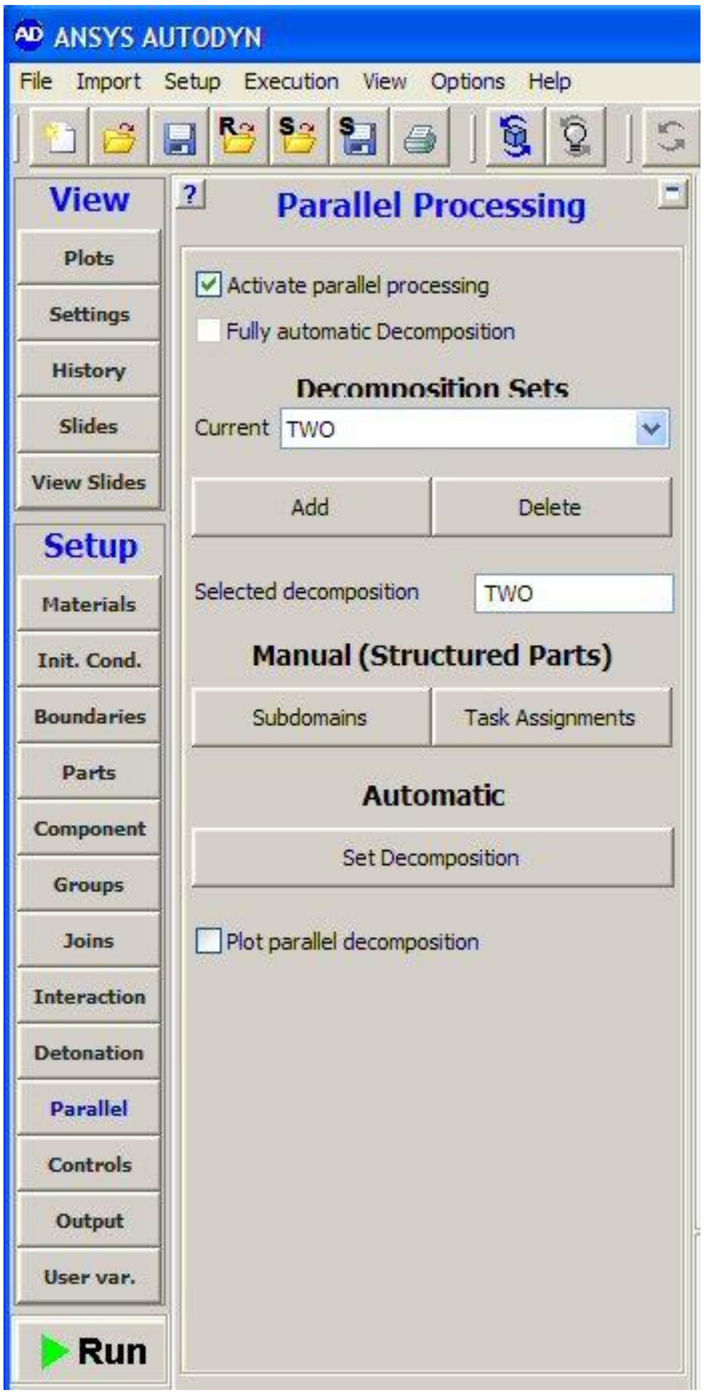
Selected decomposition <new>

Manual (Structured Parts)

Automatic

[ ] Plot parallel decomposition

Type “**TWO**” in the **Selected decomposition** text box. This will be the name of the decomposition set that will be defined. This name is also displayed in the **Current** text box:



[✓] Activate parallel processing

[ ] Fully automatic Decomposition

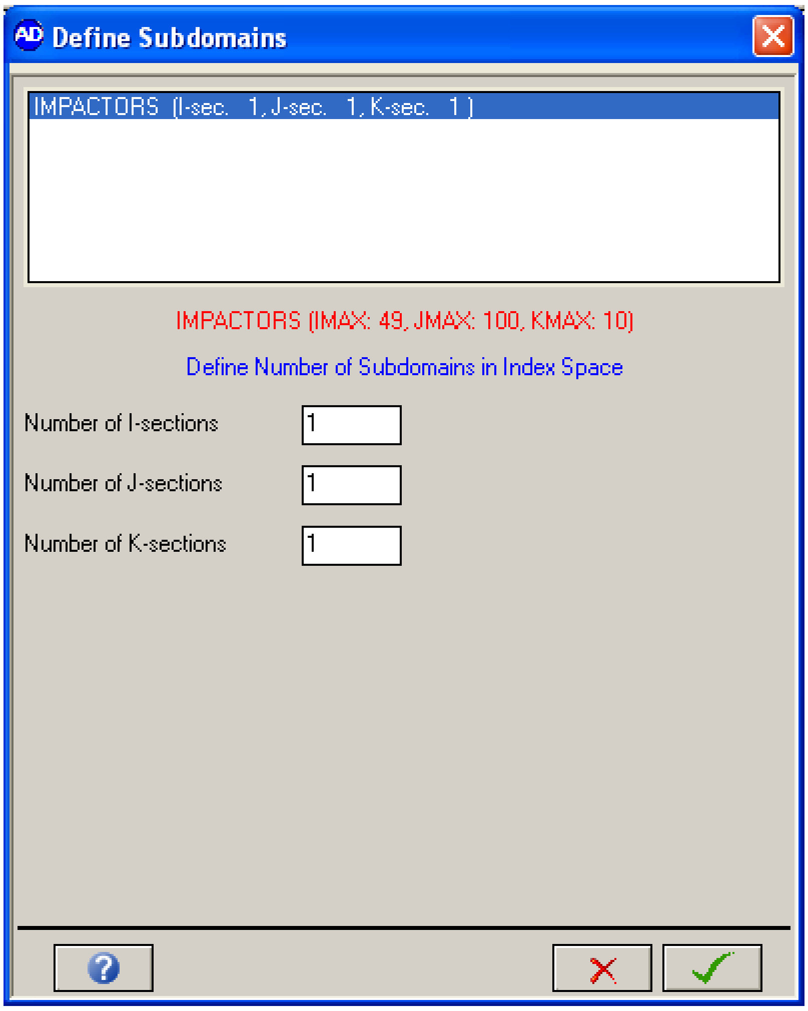
Decomposition Sets

Current: TWO

Selected decomposition: TWO

[ ] Plot parallel decomposition

Press the **Subdomains** button to activate a popup window in which the decomposition information can be entered. The following window will appear:



Define Subdomains

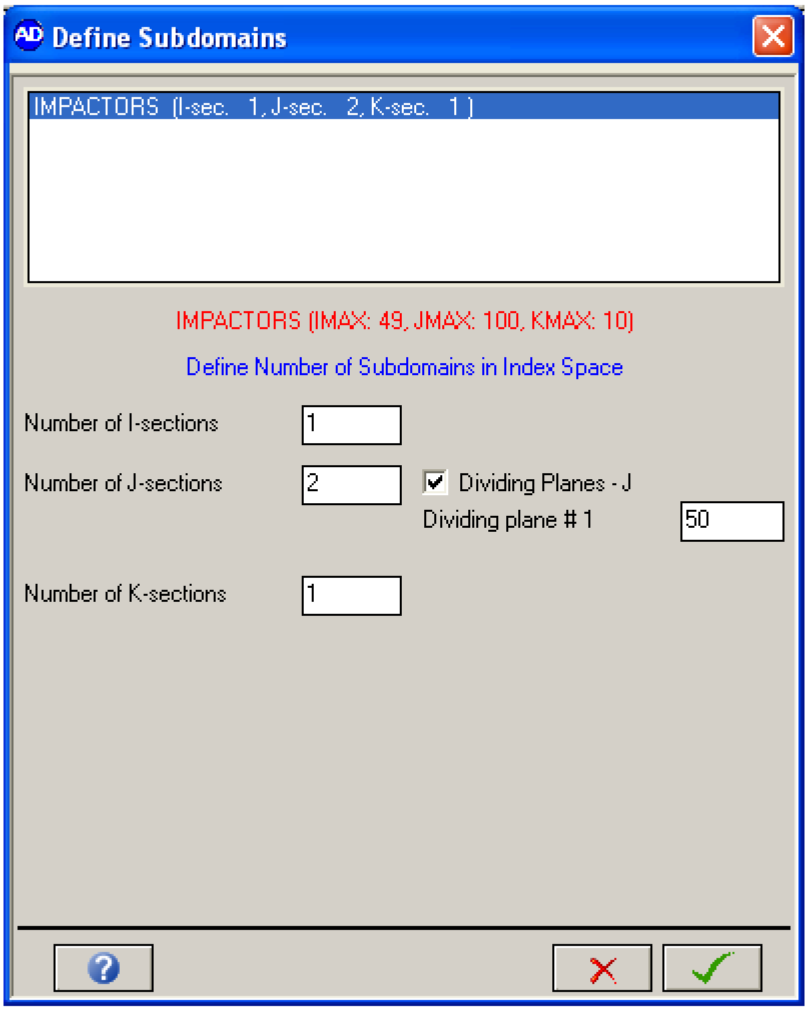
IMPACTORS (I-sec. 1, J-sec. 1, K-sec. 1)

IMPACTORS (IMAX: 49, JMAX: 100, KMAX: 10)

Define Number of Subdomains in Index Space

Number of I-sections	1
Number of J-sections	1
Number of K-sections	1

As we wish to decompose the part into two sub-domains in the J direction, enter "**2**" in the **Number of J-sections** text box. AUTODYN will automatically evaluate the optimum dividing plane for the decomposition and enter it in the **Dividing plane #1** text box:


Define Subdomains

IMPACTORS (I-sec. 1, J-sec. 2, K-sec. 1)

IMPACTORS (IMAX: 49, JMAX: 100, KMAX: 10)

Define Number of Subdomains in Index Space

Number of I-sections: 1

Number of J-sections: 2 [✓ Dividing Planes - J]

Dividing plane # 1: 50

Number of K-sections: 1

You may alter the J value of the dividing plane by changing the number within the text box. There may be situations where this would be advantageous, but this would involve more complicated models.

Press the OK button to store this information.

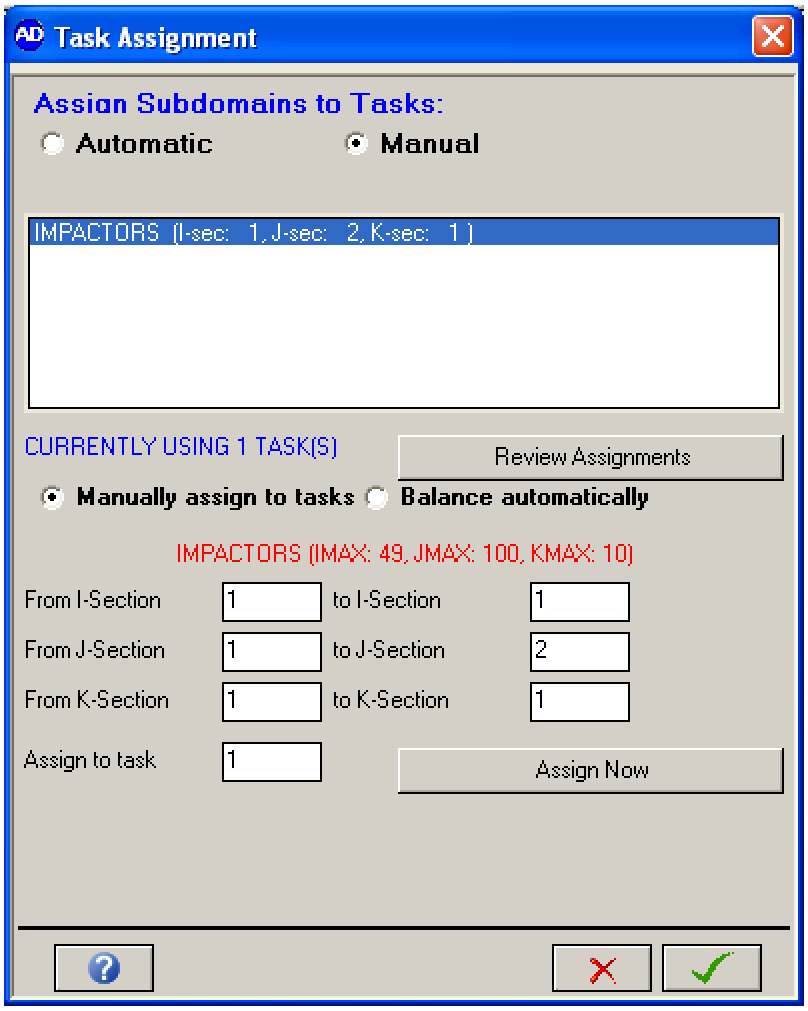
In this case the divisions can be chosen such that the resulting two sub-domains are identical in size. This is the optimum way to set up a parallel processing calculation for excellent load-balancing and most efficient execution.

This may not always be possible. You may have multiple parts, each of different dimensions, making it more difficult to decompose each part so that one sub-domain of equal size is created for each processor.

- For the current host configuration, if you have two parts of differing size, it may be possible to subdivide each part into two equal size sub-domains, so that you have a total of four sub-domains. These sub-domains could then be allocated one to each processor (one from each part) to allow perfect load balancing. There would be a small amount of overhead, due to processing two sub-domains instead of one on each processor, but this is likely to be a small price to pay for perfect load-balancing.

- Taking this principal a step further, for multiple parts that do not offer very uniform decomposition, you can decompose the parts into many more sub-domains than there are processors and let AUTODYN automatically load-balance the sub-domains over all processors (a discussion of automatic load-balancing follows). The more sub-domains AUTODYN has to distribute, the easier it will be to effectively load-balance a problem. However, the more sub-domains created, the more overhead will be involved to exchange data at sub-domain boundaries. You may have to experiment to find a good balance, but our experience so far indicates that parallel calculation are generally computation intensive, so it is likely that good load-balancing is more important than minimizing data communication between sub-domains. Most of our benchmarking has been on well load-balanced problems with minimal communication overhead, so we cannot offer more detailed guidelines at present.

The part has now been decomposed into sub-domains. Though, if you view the parallel decomposition you will see that both of the sub-domains have been automatically allocated to one task. Therefore, we need to assign these sub-domains to the available tasks in order to produce an efficient calculation. This process is initiated be pressing the **Task Assignments** button. The following popup window will appear:


Task Assignment

Assign Subdomains to Tasks:

Automatic [✓ Manual]

IMPACTORS (I-sec: 1, J-sec: 2, K-sec: 1)

CURRENTLY USING 1 TASK(S)

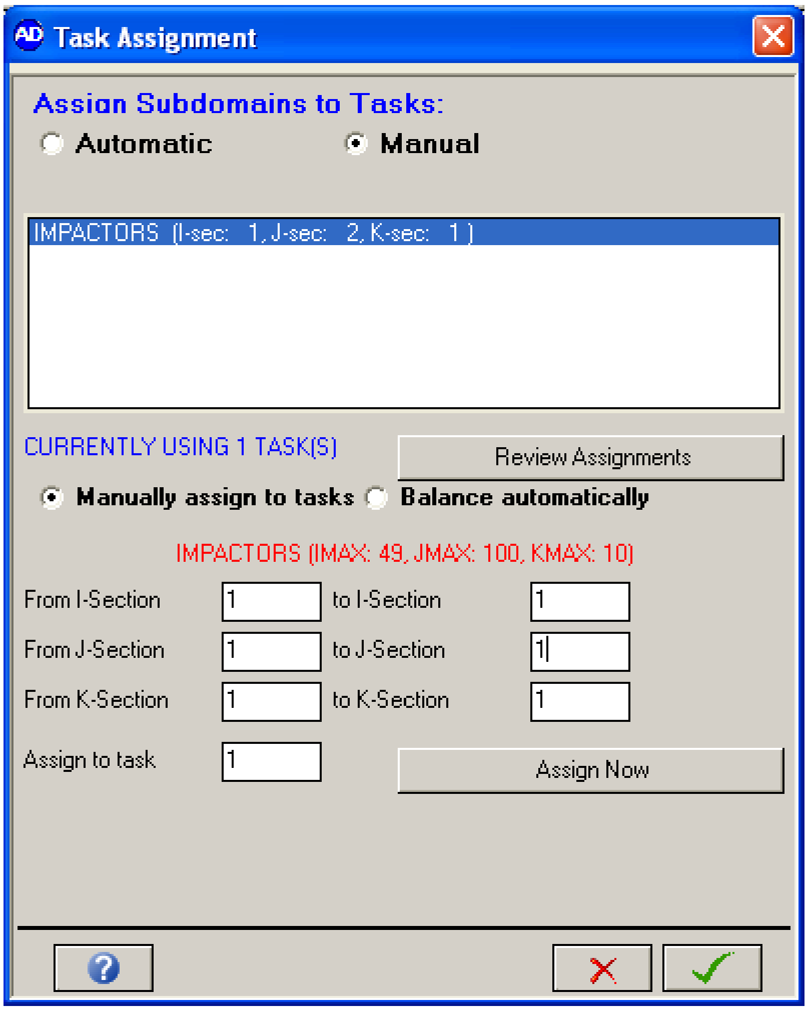
[✓ Manually assign to tasks] Balance automatically

IMPACTORS (IMAX: 49, JMAX: 100, KMAX: 10)

From I-Section	1	to I-Section	1
From J-Section	1	to J-Section	2
From K-Section	1	to K-Section	1
Assign to task	1		

The sub-domains can be assigned manually or automatically. If the **Automatic** radio box is checked then the sub-domains will be assigned at run time. If the model is a very simple one then this course of action could be taken. Here we will manually assign the sub-domains to the available tasks in order to become familiar with the process.

We wish to assign the two J-sections of the model to two different tasks. To do this, enter 1 into the **to J-Section** text box and press the **Assign Now** button:



Task Assignment

Assign Subdomains to Tasks:
Automatic [✓ Manual]

IMPACTORS (I-sec: 1, J-sec: 2, K-sec: 1)

CURRENTLY USING 1 TASK(S)
Review Assignments

[✓ Manually assign to tasks] Balance automatically

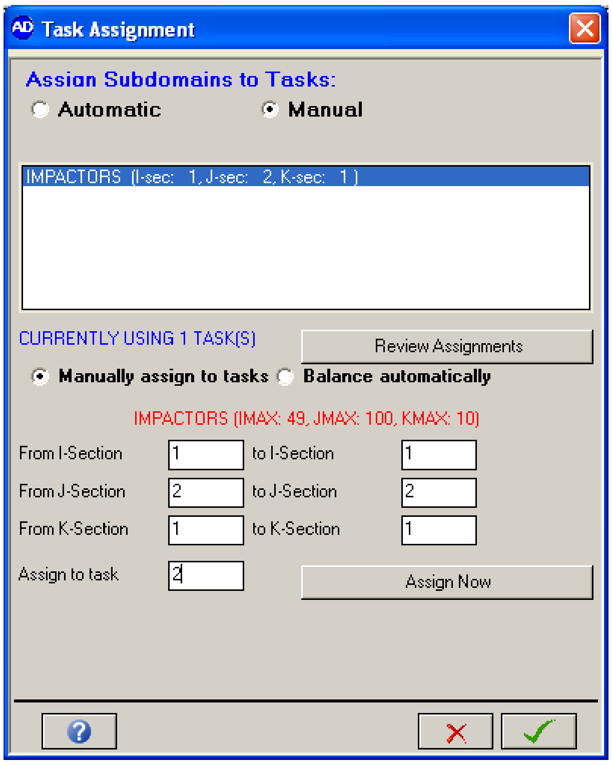
IMPACTORS (IMAX: 49, JMAX: 100, KMAX: 10)

From I-Section	1	to I-Section	1
From J-Section	1	to J-Section	1
From K-Section	1	to K-Section	1
Assign to task	1		

Assign Now

These actions have assigned the first J-section sub-domain to the first task.

To assign the second J-section sub-domain to the second task enter "**2**" into the **From J-section**, **to J-section** and **Assign to task** text boxes and press the **Assign Now** button.



Task Assignment

Assign Subdomains to Tasks:
Automatic [✓ Manual]

IMPACTORS (I-sec: 1, J-sec: 2, K-sec: 1)

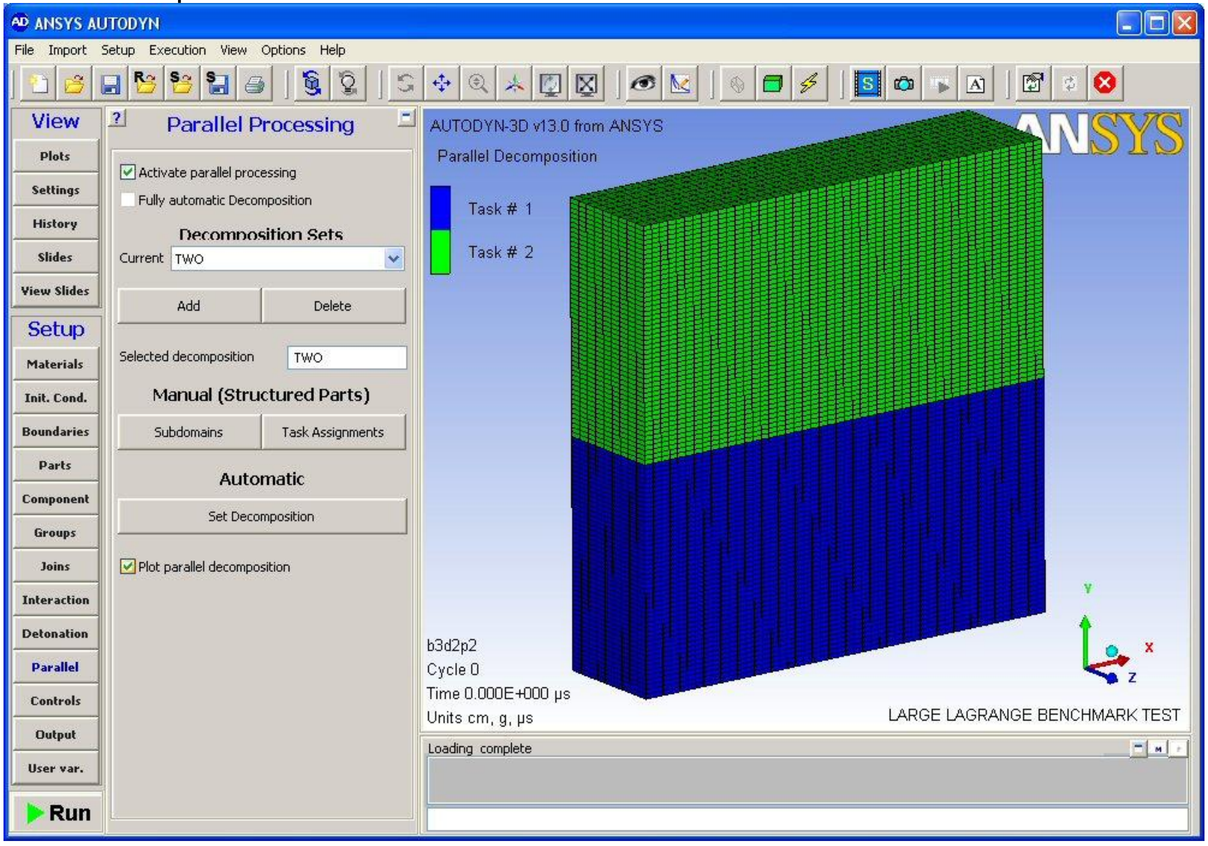
CURRENTLY USING 1 TASK(S)

[✓ Manually assign to tasks] Balance automatically

IMPACTORS (IMAX: 49, JMAX: 100, KMAX: 10)

From I-Section	1	to I-Section	1
From J-Section	2	to J-Section	2
From K-Section	1	to K-Section	1
Assign to task	2		

If you have automatic refresh activated the image of the model on screen should have been updated to show the two sub-domains on two different tasks:



ANSYS AUTODYN

Parallel Processing

[✓] Activate parallel processing
[ ] Fully automatic Decomposition

Decomposition Sets

Current: TWO

Selected decomposition: TWO

Manual (Structured Parts)

Automatic

[✓] Plot parallel decomposition

AUTODYN-3D v13.0 from ANSYS

Parallel Decomposition

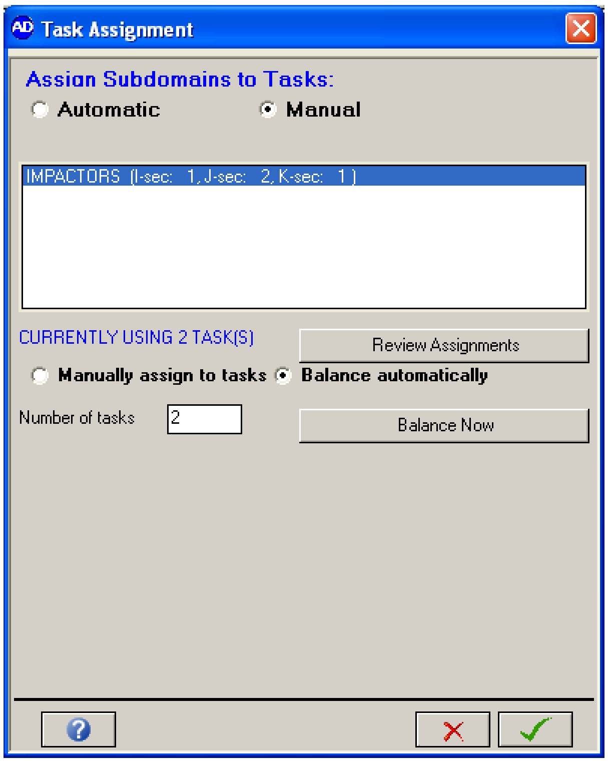
🟦 Task # 1
🟩 Task # 2

b3d2p2
Cycle 0
Time 0.000E+000 μs
Units cm, g, μs

LARGE LAGRANGE BENCHMARK TEST

Loading complete

We could have balanced the task assignments automatically and viewed the results instantly by checking the **Balance automatically** radio box in the **Task Assignment** window. This results in a change to the configuration of the window:


Task Assignment
Assign Subdomains to Tasks:
Automatic [✓ Manual]
IMPACTORS (I-sec: 1, J-sec: 2, K-sec: 1)
CURRENTLY USING 2 TASK(S)
Review Assignments
Manually assign to tasks [✓ Balance automatically]
Number of tasks 2
Balance Now

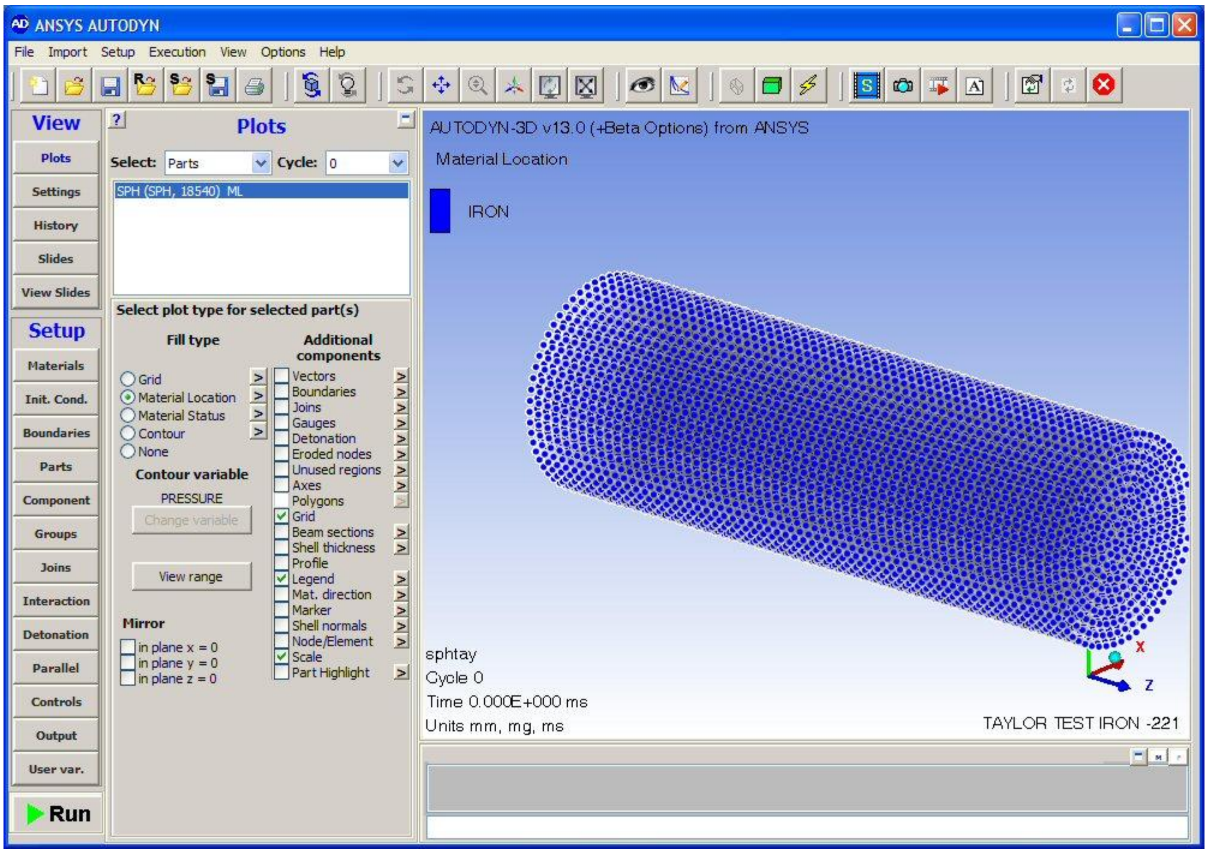
By pressing the **Balance Now** button the task assignments will be immediately processed automatically. If you view the domain decomposition image then you will see that the decomposition is identical to that which we defined manually.

The model is now ready to be run in parallel. Press the **RUN** button to execute the calculation.

4.2. Decomposing SPH Calculations

Decomposing SPH parts adopts a similar approach to that applied for grid calculations, though there is no manual option to decompose the problem.

Load in the SPH problem **SPHTAY** from the samples directory within the AUTODYN distribution.



Select: Parts
Cycle: 0
SPH (SPH, 18540) ML

Select plot type for selected part(s)

Fill type: Grid, [✓ Material Location], Material Status, Contour, None

Contour variable: PRESSURE

Additional components: Vectors, Boundaries, Joins, Gauges, Detonation, Eroded nodes, Unused regions, Axes, Polygons, [✓ Grid], Beam sections, Shell thickness, Profile, [✓ Legend], Mat. direction, Marker, Shell normals, Node/Element, [✓ Scale], Part Highlight

Mirror: in plane x = 0, in plane y = 0, in plane z = 0

AUTODYN-3D v13.0 (+Beta Options) from ANSYS

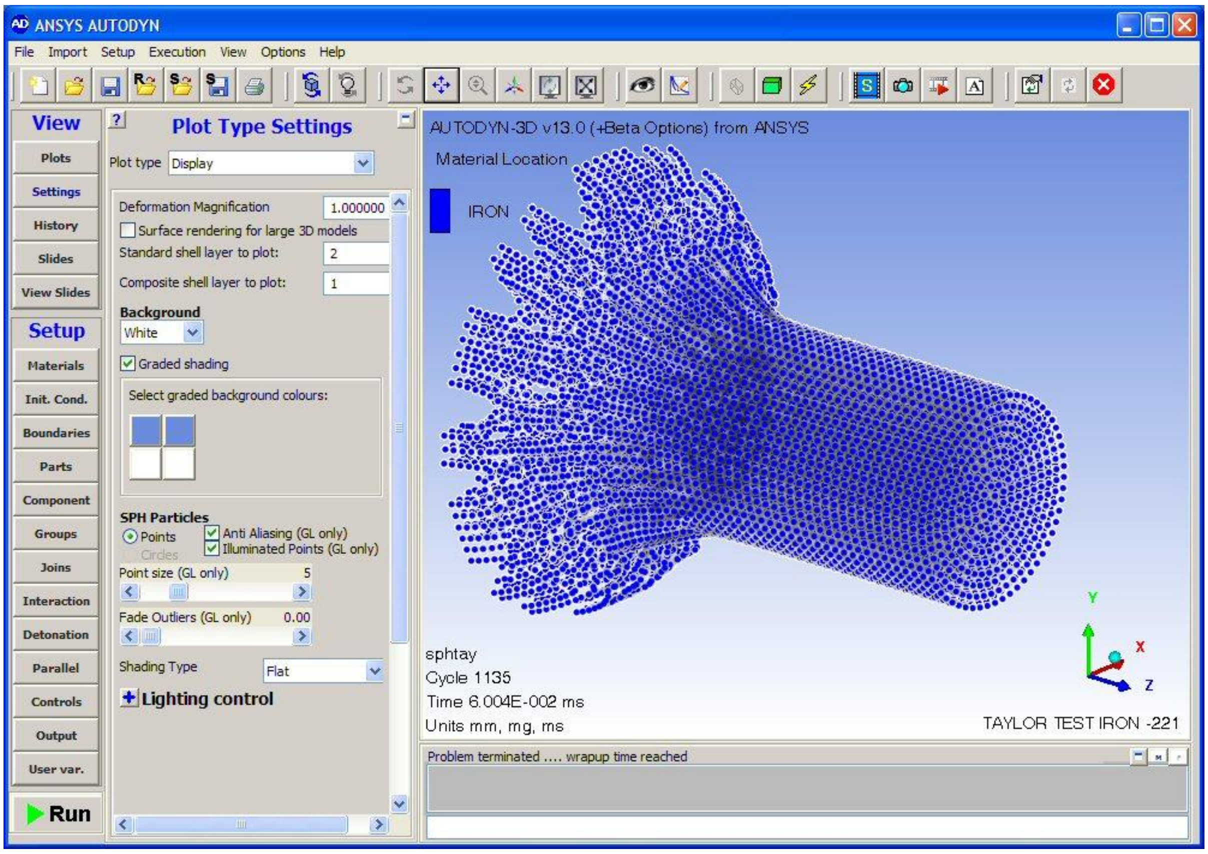
Material Location

IRON

sphtay
Cycle 0
Time 0.000E+000 ms
Units mm, mg, ms

TAYLOR TEST IRON -221

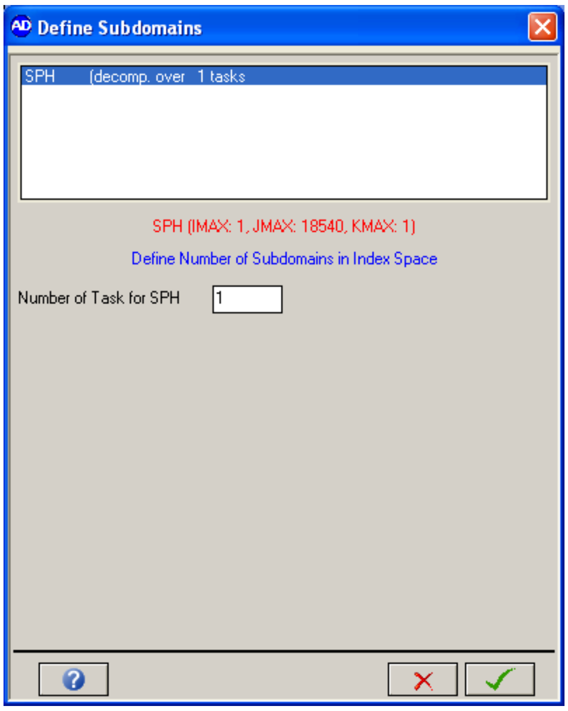
This is an iron Taylor Test problem where the deformation of a cylinder is studied after impacting a wall at 221 ms^{-1}. The model consists of a single SPH object made up of 18540 SPH nodes. Running the model to 0.06 ms will produce the following results:



Plot type Display
Deformation Magnification 1.000000
Surface rendering for large 3D models
Standard shell layer to plot: 2
Composite shell layer to plot: 1
Background White
✓ Graded shading
Select graded background colours:
SPH Particles
● Points
Circles
✓ Anti Aliasing (GL only)
✓ Illuminated Points (GL only)
Point size (GL only) 5
Fade Outliers (GL only) 0.00
Shading Type Flat

AUTODYN-3D v13.0 (+Beta Options) from ANSYS
Material Location
sphtay
Cycle 1135
Time 6.004E-002 ms
Units mm, mg, ms
TAYLOR TEST IRON -221
Problem terminated .... wrapup time reached

To decompose the model, press the **Parallel** button in the **Setup** toolbar and activate parallel processing by ticking the appropriate check box.
Following the same method as when decomposing the Lagrange calculation, press the **Add** button in the **Decomposition Sets** section of the **Parallel Processing** panel. Again enter "**TWO**" as the name of the decomposition and press the **Sub-domains** button, which will result in the following window being displayed.

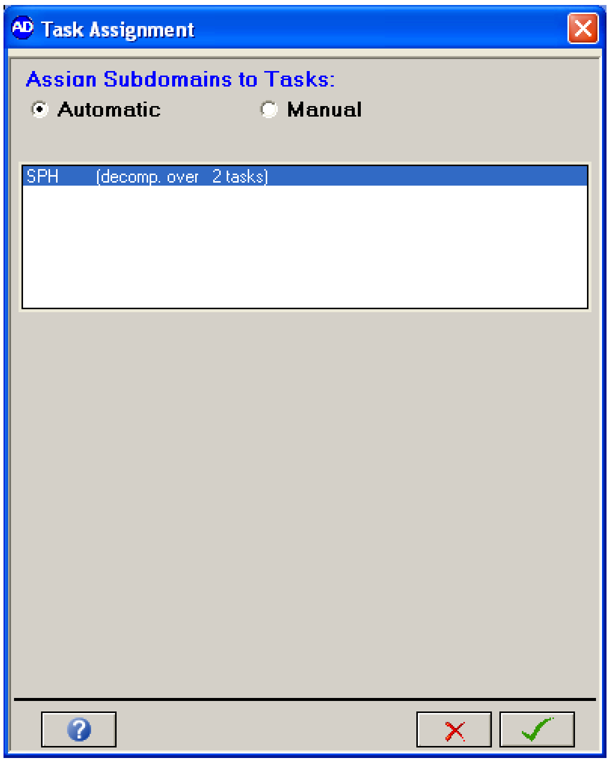


SPH (decomp. over 1 tasks

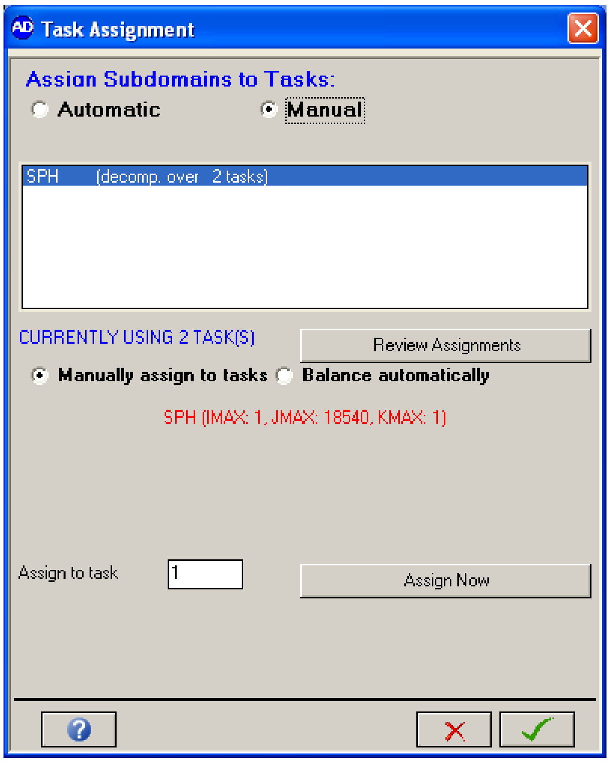
SPH (IMAX: 1, JMAX: 18540, KMAX: 1)
Define Number of Subdomains in Index Space
Number of Task for SPH 1

Enter "**2**" in the **Number of Task for SPH** text box and press the tick button.

Now, press the **Task Assignments** button on the main **Parallel Processing** panel. The following window will be displayed.

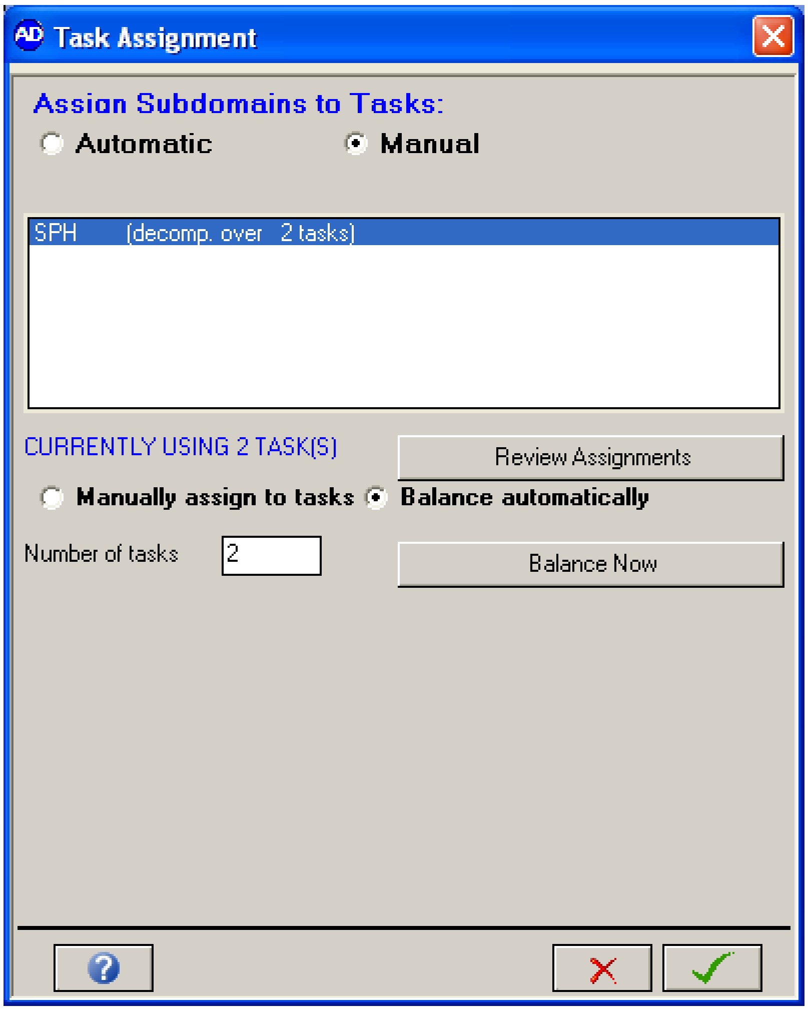


By default the **Automatic** radio box will be chosen. By leaving the default setting, the sub-domains would be assigned to tasks at runtime. As we wish to view the decomposition, check the **Manual** radio box, which will result in further options being displayed.


Task Assignment
Assign Subdomains to Tasks:
Automatic [✓ Manual]
SPH (decomp. over 2 tasks)
CURRENTLY USING 2 TASK(S)
Review Assignments
[✓ Manually assign to tasks] Balance automatically
SPH (IMAX: 1, JMAX: 18540, KMAX: 1)
Assign to task 1
Assign Now

If we wished to assign the entire SPH object to one task, which may be a viable option if the model contained multiple parts, we would enter the task number in the **Assign to task** text box and press the **Assign Now** button. As the model consists of a single SPH object we need to decompose the object onto the two available processors in order to efficiently compute the results.

Check the **Balance automatically** radio box and the available options will be modified.


Task Assignment

Assign Subdomains to Tasks:
Automatic [✓ Manual]

SPH (decomp. over 2 tasks)

CURRENTLY USING 2 TASK(S)

Manually assign to tasks [✓ Balance automatically]

Number of tasks: 2

The number of tasks will automatically be placed in the **Number of tasks** text box and therefore we only have to press the **Balance Now** button in order to assign the SPH nodes to the different tasks. AUTODYN will attempt to assign the nodes in the most efficient configuration possible for parallel processing. After doing this, press the tick button to change the focus back to the main AUTODYN window. Check the **Plot parallel decomposition** tick box and the decomposition will be displayed.

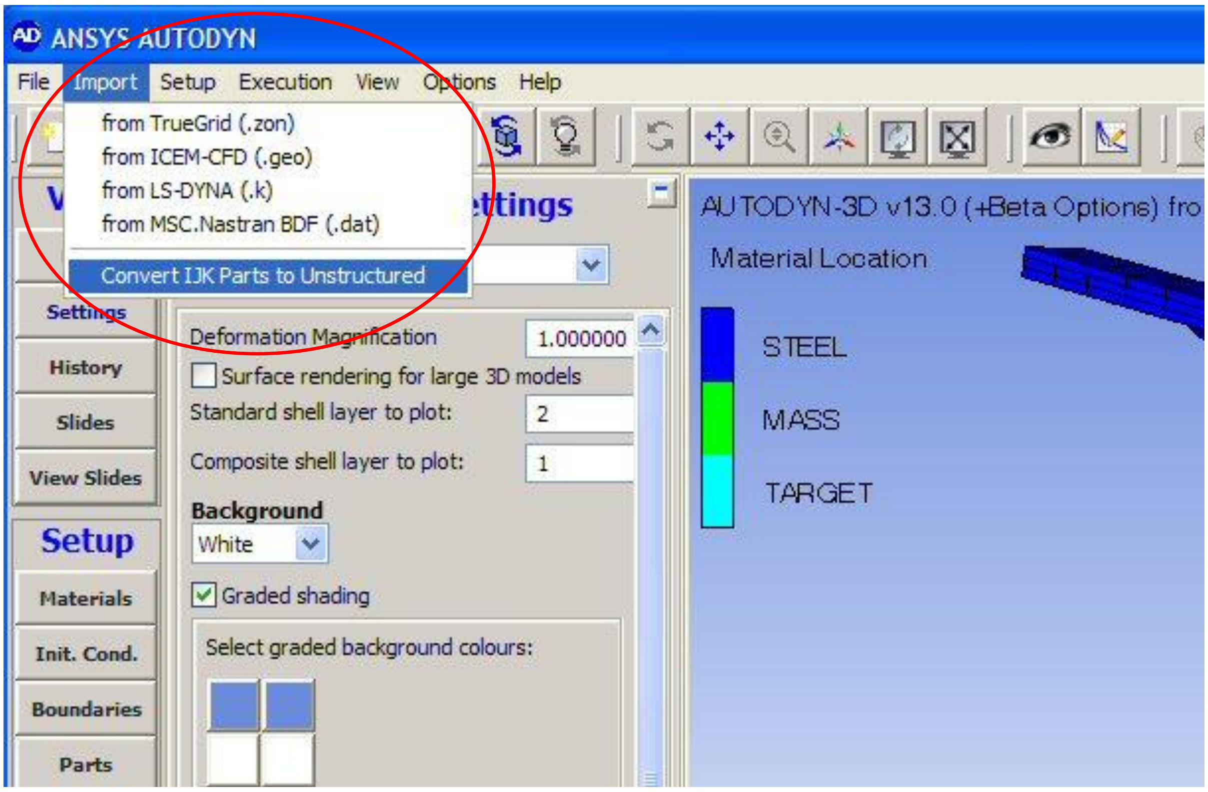
The model can be now efficiently run as a parallel computation.

4.3. Decomposing Unstructured Calculations

The decomposition of unstructured parts is applied automatically in a similar method to the decomposition of SPH grids.

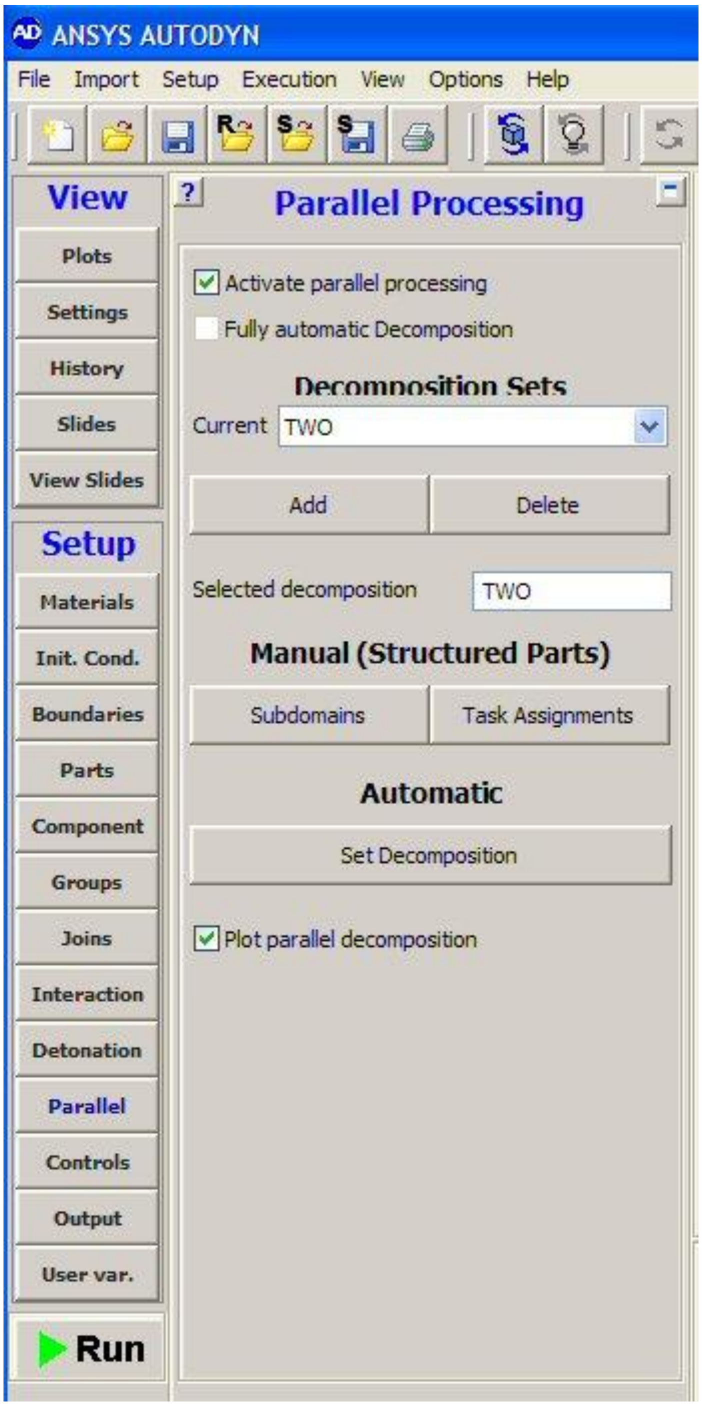
Load in CYRAIL from the samples directory within the Workbench distribution.

Convert this structured model into unstructured parts by using the "Import->Convert IJK Parts to Unstructured" option and ticking all of the available options:


ANSYS AUTODYN
File
Import
Setup
Execution
View
Options
Help
from TrueGrid (.zon)
from ICEM-CFD (.geo)
from LS-DYNA (.k)
from MSC.Nastran BDF (.dat)
Convert IJK Parts to Unstructured
Settings
History
Slides
View Slides
Setup
Materials
Init. Cond.
Boundaries
Parts
Deformation Magnification 1.000000
Surface rendering for large 3D models
Standard shell layer to plot: 2
Composite shell layer to plot: 1
Background
White
Graded shading
Select graded background colours:
AUTODYN-3D v13.0 (+Beta Options) fro
Material Location
STEEL
MASS
TARGET

This makes two unstructured parts called VOLUME 1 and VOLUME 2. Rename these parts to TARG and CYLINDER in the parts menu.

Go to the Parallel panel and tick the box to activate parallel processing. Add a decomposition set and call it TWO:



✓ Activate parallel processing

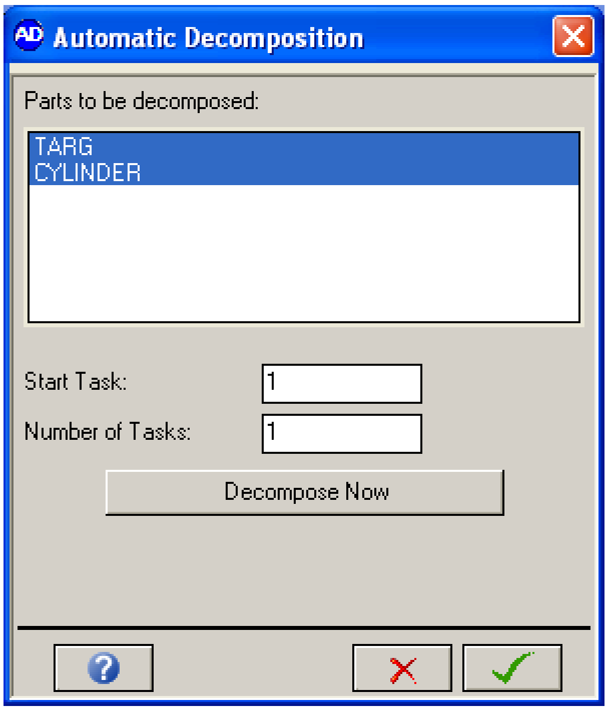
Fully automatic Decomposition

Current: TWO

Selected decomposition: TWO

✓ Plot parallel decomposition

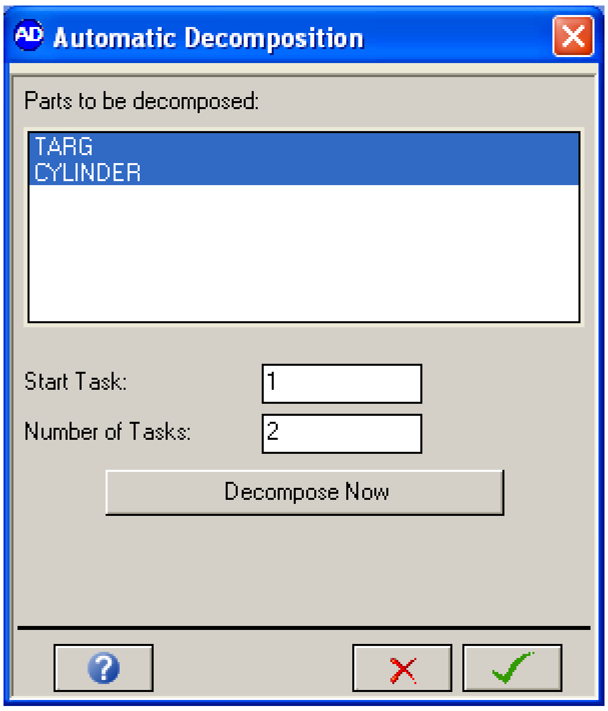
You will see that there are two sections within the Decomposition Sets area of the menu. For unstructured parts the second button marked Set Decomposition should be used. Pressing this button activates a window in which the information regarding the decomposition can be entered:



Parts to be decomposed:
TARG
CYLINDER

Start Task: 1
Number of Tasks: 1

All the unstructured parts within the model will in the Parts to be decomposed window and all are highlighted by default. To decompose the model over two tasks (task 1 and task 2) simply enter 2 into the Number of Tasks box and press the Decompose Now button:

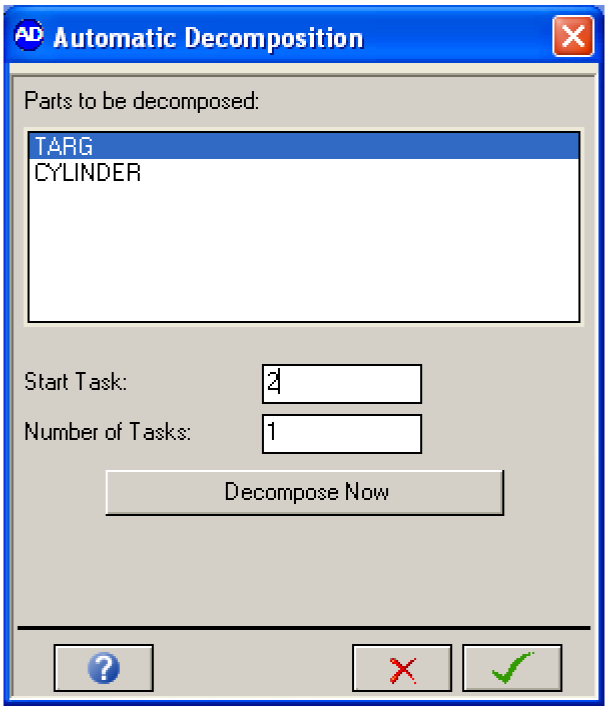


Parts to be decomposed:
TARG
CYLINDER

Start Task: 1
Number of Tasks: 2

Press the tick button and by choosing to plot the parallel decomposition, the following decomposition is seen:

If you wish to have the part TARG on Task 2, press the Set Decomposition again and highlight part TARG only, enter two into the Start Task box and press the Decompose Now button:


Automatic Decomposition

Parts to be decomposed:

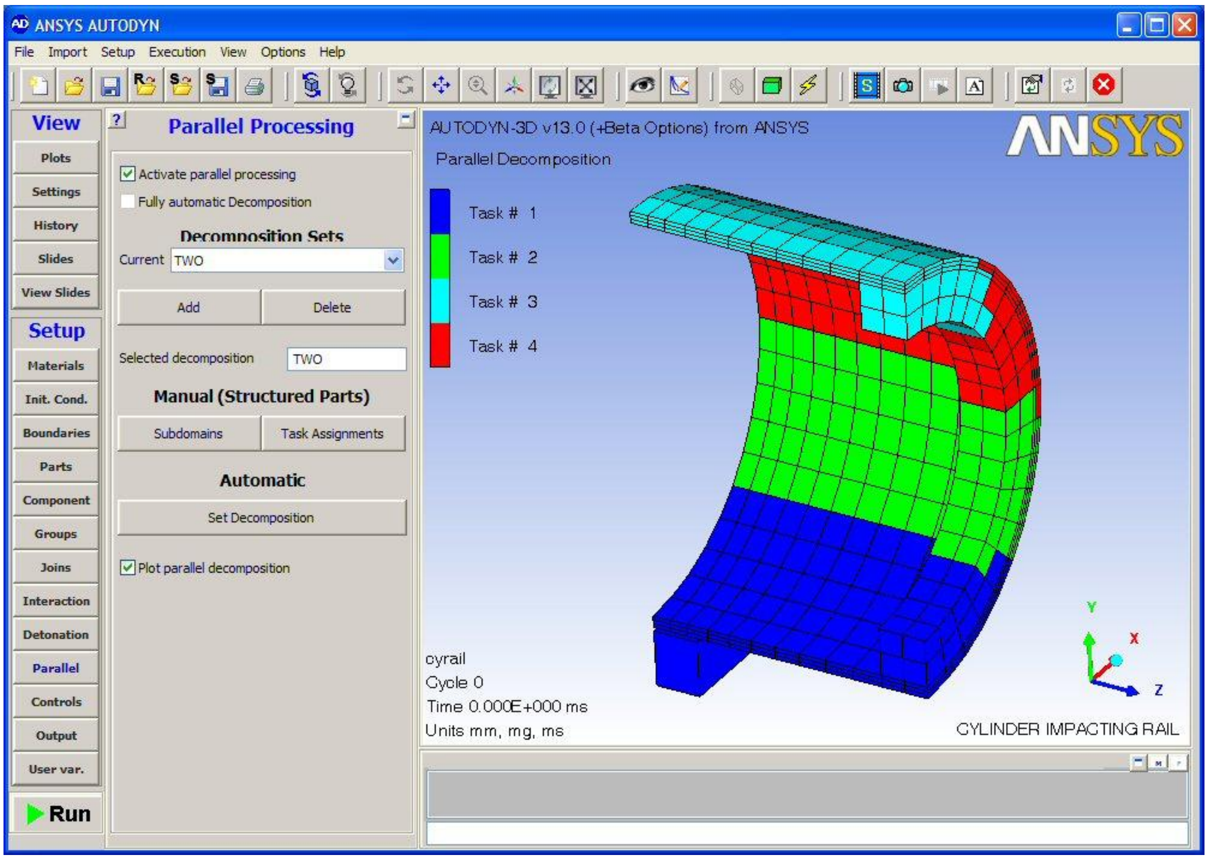
TARG
CYLINDER

Start Task: 2

Number of Tasks: 1

The part TARG will now be assigned to task 2:

In a similar manner the model can be decomposed over 4 slaves:



☑ Activate parallel processing
☐ Fully automatic Decomposition
Current: TWO
Selected decomposition: TWO
☑ Plot parallel decomposition

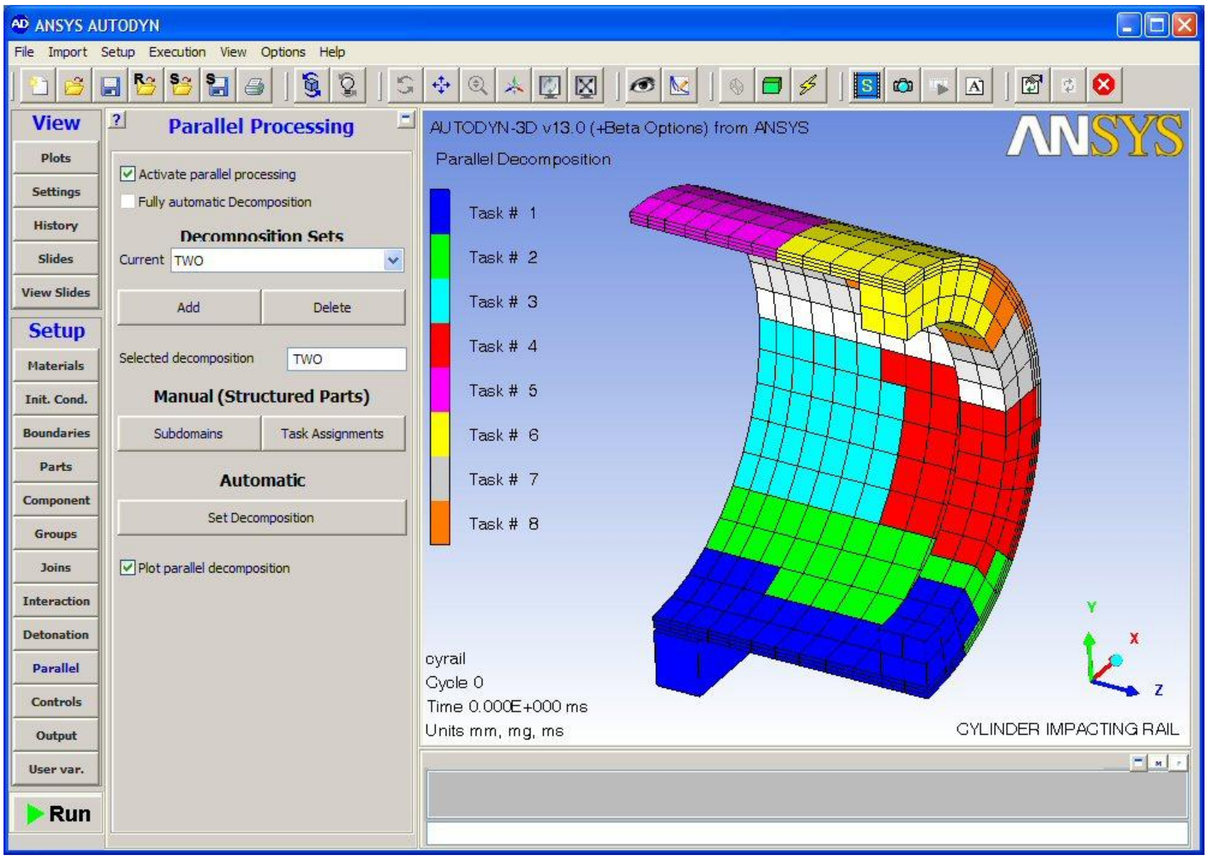
AUTODYN-3D v13.0 (+Beta Options) from ANSYS
Parallel Decomposition

🟦	Task # 1
🟩	Task # 2
🟦	Task # 3
🟥	Task # 4

cyrail
Cycle 0
Time 0.000E+000 ms
Units mm, mg, ms

CYLINDER IMPACTING RAIL

and over 8 slaves:



Parallel Processing

☑ Activate parallel processing
☐ Fully automatic Decomposition

Decomposition Sets

Current TWO

Selected decomposition TWO

Manual (Structured Parts)

Automatic

☑ Plot parallel decomposition

AUTODYN-3D v13.0 (+Beta Options) from ANSYS

Parallel Decomposition

🟦	Task # 1
🟩	Task # 2
🟦	Task # 3
🟥	Task # 4
🟪	Task # 5
🟨	Task # 6
⬜	Task # 7
🟧	Task # 8

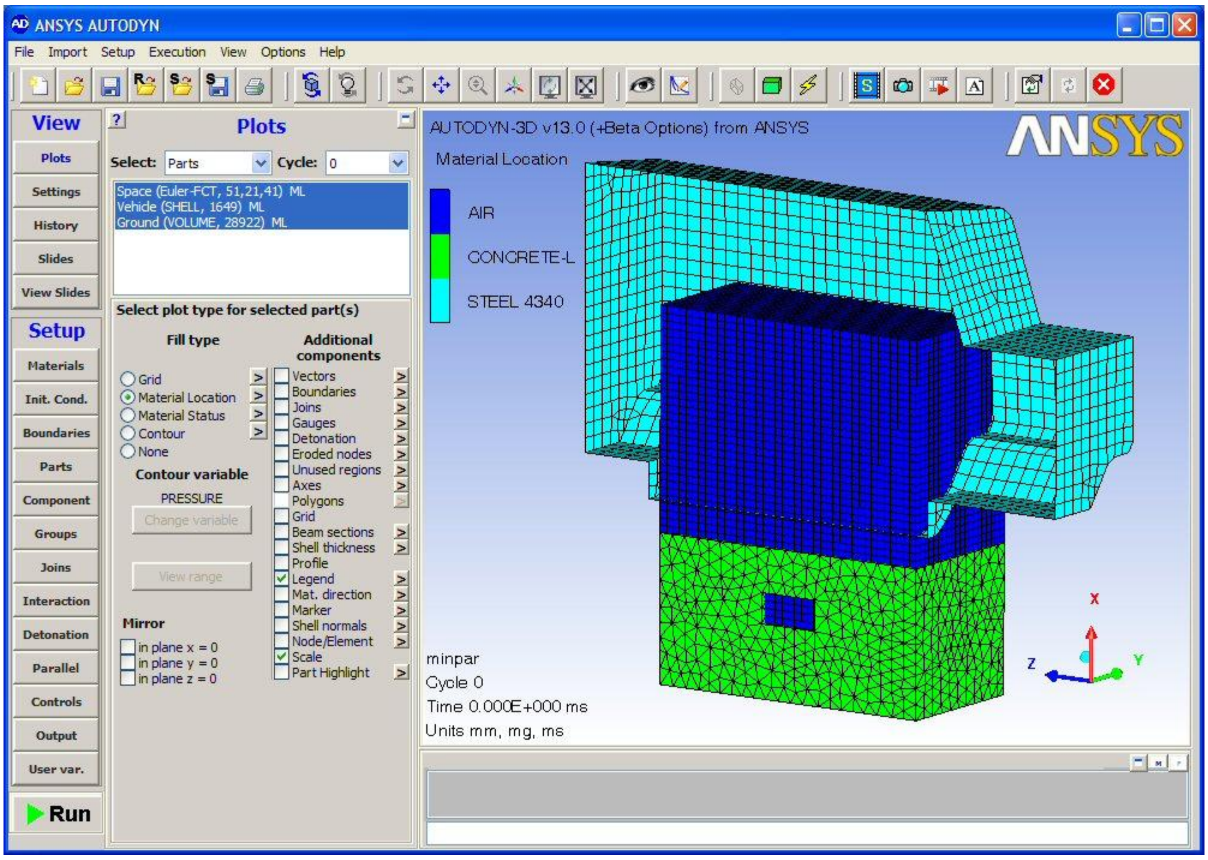
cyrail
Cycle 0
Time 0.000E+000 ms
Units mm, mg, ms

CYLINDER IMPACTING RAIL

4.4. Decomposing Euler-Lagrange Coupled Problems

In the following mine blast example, you will create a domain decomposition set for an analysis where an Euler Ideal Gas parts interacts with a vehicle structure and soil through Euler-Lagrange coupling.
The decomposition of models that include Euler parts interacting with unstructured parts can be applied automatically.

Load in the model MINPAR from the samples directory within the Workbench distribution.



Select: Parts Cycle: 0

Space (Euler-FCT, 51,21,41) ML
Vehicle (SHELL, 1649) ML
Ground (VOLUME, 28922) ML

Select plot type for selected part(s)

Fill type: ○ Grid [● Material Location] ○ Material Status ○ Contour ○ None

Contour variable: PRESSURE

Additional components: ☐ Vectors ☐ Boundaries ☐ Joins ☐ Gauges ☐ Detonation ☐ Eroded nodes ☐ Unused regions ☐ Axes ☐ Polygons ☐ Grid ☐ Beam sections ☐ Shell thickness ☐ Profile ☑ Legend ☐ Mat. direction ☐ Marker ☐ Shell normals ☐ Node/Element ☑ Scale ☐ Part Highlight

Mirror: ☐ in plane x = 0 ☐ in plane y = 0 ☐ in plane z = 0

AUTODYN-3D v13.0 (+Beta Options) from ANSYS

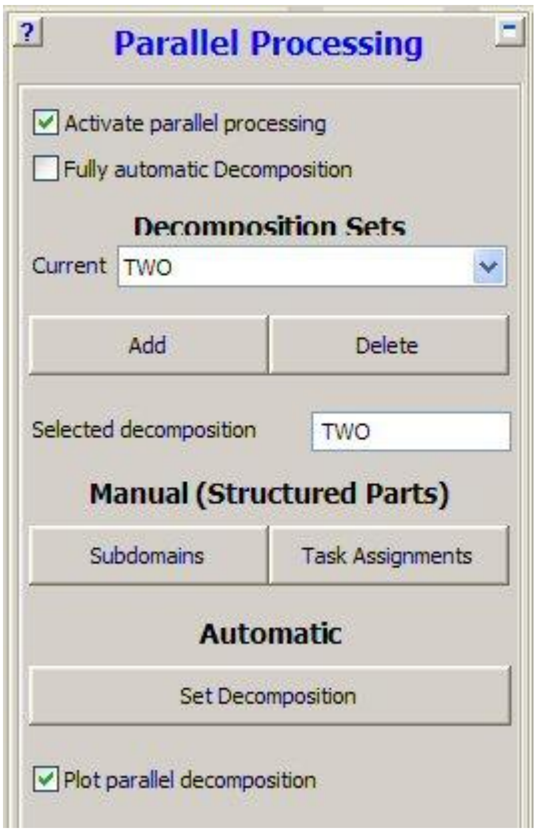
Material Location

🟦	AIR
🟩	CONCRETE-L
🟦	STEEL 4340

minpar
Cycle 0
Time 0.000E+000 ms
Units mm, mg, ms

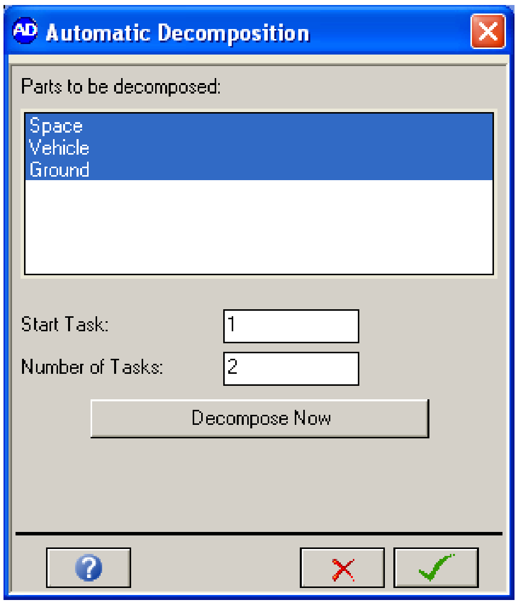
The model consists of an Euler Ideal Gas part (50x20x40) filled with air and two structural parts that model the soil (29.000 Tetra elements) and vehicle (1650 shell elements). The Euler part is filled with high pressurized gas in the region of the soil part where the mine is located and the vehicle will be hit by the blast wave and eroded nodes of the ejected soil. The blast wave interaction is calculated through Euler-Lagrange Interaction and the interaction of the ejected soil and vehicle is modeled through Lagrange-Lagrange interaction and erosion.
If this problem is run for 2 msec. and a combined pressure contour/velocity vector plot is requested, you will see the following screen.

Go to the Parallel panel and tick the box to activate parallel processing. Add a decomposition set and call it TWO:



☑ Activate parallel processing
☐ Fully automatic Decomposition
Current: TWO
Selected decomposition: TWO
☑ Plot parallel decomposition

Euler in combination with unstructured parts can automatically be decomposed. Select the button marked Set Decomposition and pressing this button activates a window in which the information regarding the decomposition can be entered:



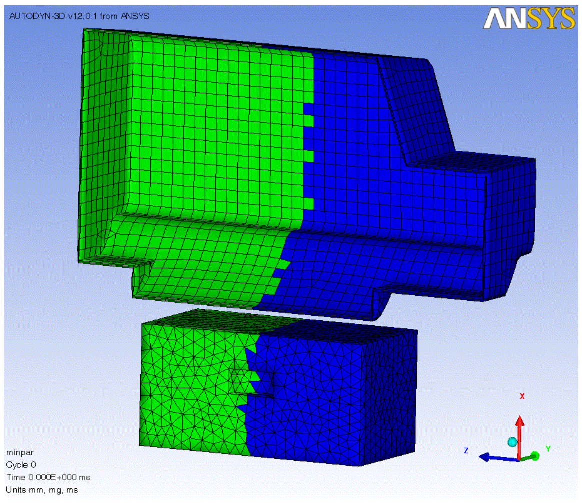
Parts to be decomposed:
Space
Vehicle
Ground

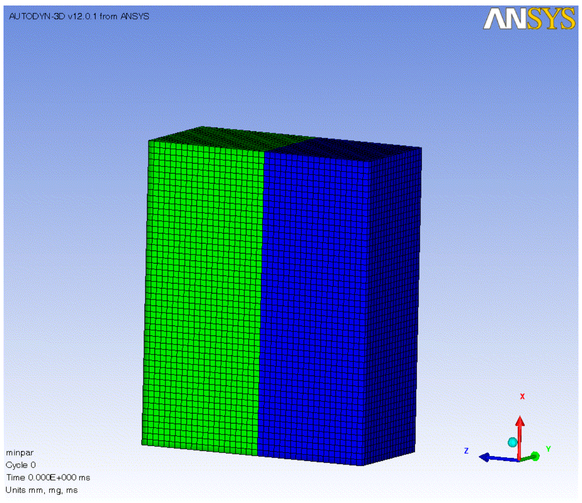
Start Task: 1
Number of Tasks: 2

The unstructured parts within the model and structured Euler Ideal Gas part will in the Parts to be decomposed window and all are highlighted by default. To decompose the model over two tasks (task 1 and task 2) simply enter 2 into the Number of Tasks box and press the Decompose Now button:

Press the tick button and by choosing to plot the parallel decomposition, the following decomposition is seen:

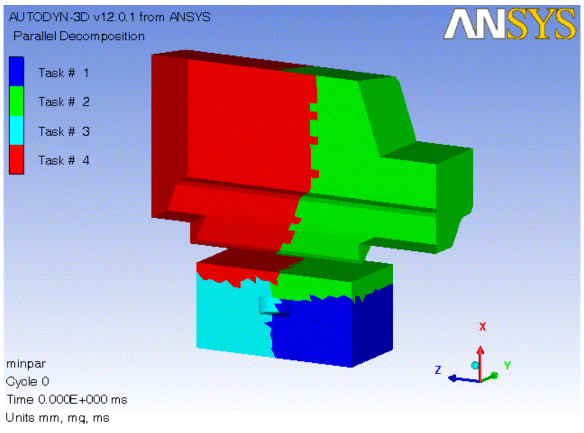
The decomposition algorithm automatically produces a decomposition configuration with good load balancing qualities and minimal inter-processor communication.
To further enhance the efficiency of the coupled calculation the sub-domains of the FE structure are placed on the same processor as the Euler Ideal Gas sub-domains located in the same geometric space. This decreases the necessary inter-processor communication for the coupling calculations, as can be seen below.

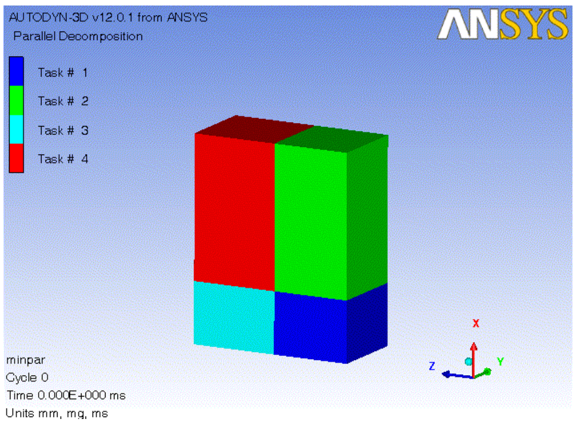




In a similar manner the model can be decomposed over 4 slaves:

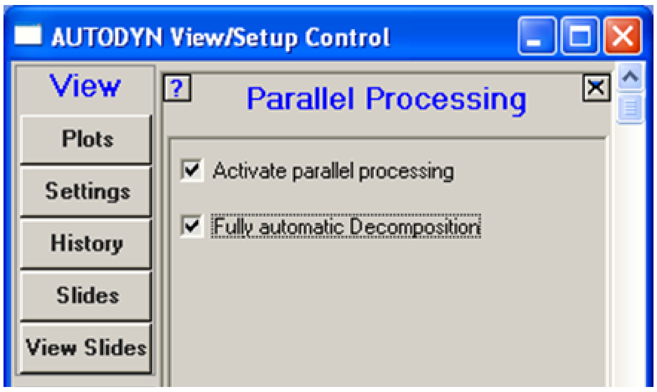
Again the sub-domains of the FE structure are placed on the same processor as the Euler Ideal Gas sub-domains located in the same geometric space to decrease the necessary inter-processor communication for the coupling calculations, as can be seen below.





5. Fully Automatic Decomposition on Windows

AUTODYN offers a fully automatic decomposition option for models containing any combination of Euler and unstructured solvers.

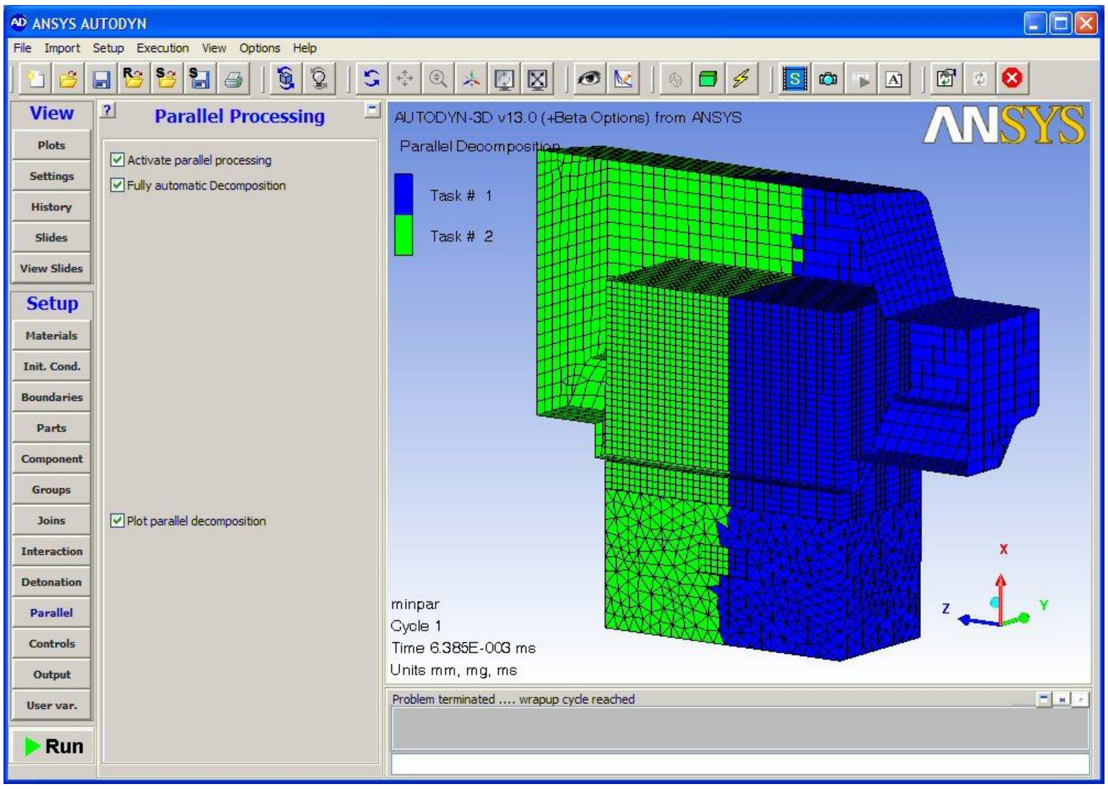


☑ Activate parallel processing

☑ Fully automatic Decomposition

When using this fully automatic option there is no further input required by the user in terms of model set-up. On execution the model is automatically decomposed over the number of tasks specified in the Platform MPI appl file. At any point after the model has been initialized the resultant decomposition can be viewed using the “Plot parallel decomposition” option also found in the parallel panel.

Load in the model MINPAR from the samples directory within the Workbench distribution.
Select the Fully automatic decomposition option and run the analysis for 1 cycle with 2 CPU’s defined in the Platform MPI appl file. The automatic domain decomposition that has been used can be seen in the picture below.



☑ Activate parallel processing

☑ Fully automatic Decomposition

☑ Plot parallel decomposition

minpar
Cycle 1
Time 6.385E-003 ms
Units mm, mg, ms

Problem terminated .... wrapup cycle reached

Appendix A. Overview of Parallelized Features and Known Limitations

Feature		Y	N	Limitation
Solvers	Structured	✓		
	Unstructured	✓		
	SPH	✓		
	NBS Tet		x	
	Rigid bodies	✓		
	Euler Ideal Gas	✓		
	Euler Multi-Material	✓		
Contact	Gap	✓		
	Trajectory	✓		Beta option • Only the Penalty option is parallelized • Will only run on 2,4,8,16,32 or 64 slaves • Parallel efficiencies are likely to show low speed-ups compared with serial for models: - with significant erosion - containing rigid bodies and/or bonded connections • Trajectory contact cannot be run in parallel when an SPH part is present in the model.
Connections	Bonded/Breakable	✓		Beta option See limitations of Trajectory Contact
	Spotwelds	✓		
	Reinforcement beams		x	
Joins	Structured/unstructured	✓		• Although the joins between unstructured parts have been parallelized it is strongly advised to avoid the use of unstructured parts in combination with joins in parallel analyses. • An option is available that will merge joined unstructured nodes that reside at the same physical location in the model into one single unstructured node. The option is available under the Join menu and will increase robustness in many applications involving joins.
	SPH-structured	✓		
	Euler Ideal Gas	✓		
	Euler Multi-Material	✓		• Joins cannot be used in combination with Euler/Lagrange coupling
Euler/Lagrange Interaction	Euler Ideal Gas	✓		
	Euler Multi-material	✓		• Joined Euler cannot be used in combination with Euler/Lagrange coupling
Boundary Conditions	Stress/Velocity/Bending Flow/Transmit	✓		
	Analytical Blast		x	
	Remote Displacement		x	
Materials	AUTODYN Materials	✓		
	Sesame Library		x	