Recent Developments in LS-DYNA® S-ALE

Hao Chen, Ian Do
Livermore Software Technology Corporation

Abstract

The LS-DYNA ALE/FSI package is widely used in studying structures under blast loading. Generally, the ALE mesh is necessarily unstructured to accommodate complex geometries; however, for simple rectilinear geometries, a structured, logically regular, mesh can be utilized. Recognition of this latter case leads to algorithmic simplifications, memory reductions, and performance enhancements, which are impossible in unstructured mesh geometries.

In 2015, LSTC introduced a new structured ALE (S-ALE) solver option dedicated to solve the subset of ALE problems where a structured mesh is appropriate. As expected, recognizing the logical regularity of the mesh brought a reduced simulation time for the case of identical structured and unstructured mesh definitions.

*The new S-ALE solver is easy to use, especially for users acquainted with the ALE solver. Only two new keywords are introduced: *ALE_STRUCTURED_MESH and *ALE_STRUCTURED_MESH_CONTROL_POINTS. The former is used to generate the mesh and invokes S-ALE solver, and the latter is to provide mesh spacing information along each local directions. Other ALE keywords remain the same.*

In this presentation we will introduce the new developments and enhancements in LS-DYNA S-ALE for the past two years.

Converting ALE models to S-ALE models

Mesh in S-ALE models are generated automatically. *ALE_STRUCTURED_MESH card has three fields, CPIDX, CPIDY, CPIDZ. Each corresponds to a *ALE_STRUCTURED_MESH_CONTROL_POINTS card which specifies mesh spacing in x,y and z directions. The coordinates used in the _CONTROL_POINTS card are taken as local coordinates. In case mesh origin (NID0) and/or local coordinate system (LCSID) are specified, global coordinates are calculated by using the following formula.

In the past two years, we observed a growing trend of needs to convert existing ALE models to use the S-ALE solver. It is not a difficult job, but tedious. Coordinates needs to be extract out; local coordinate system has to be generated. For unevenly spaced mesh, it requires more user efforts. We implemented an automated converter to help.

To use that, one simply add an *ALE_STRUCTURED_MESH card with CPIDX setting to either 0 or -1 into the existing ALE model. Both CPIDX=0 and CPIDX=-1 convert the ALE model to S-ALE model and output an ASCII file named “saleconvrt.inc” containing the converted S-ALE keywords. The difference is CPIDX=-1 continues the run with S-ALE solver; CPIDX=0 ALE solver. The part id (DPID), node start ID (NBID), element start ID (EBID) are optional.

This conversion does the three following things. First, it extracts the geometric information from ALE input decks and then rebuilds the mesh using S-ALE keywords. Secondly, in case of multiple mesh parts are used in the ALE model, special care needs to be taken to combine these multiple mesh parts into one mesh part and in the new mesh, fill the volume by different fluids represented by different mesh parts. Thirdly, if ambient parts are used, we delete such parts and convert them into "ambient elements" using *BOUNDARY_AMBIENT cards.

The second and third thing might be a little hard to understand. To know what it really does, we need to explain the key difference in ALE and S-ALE setup – the role of *PART definition. In a structure analysis, the *PART definition contains three things: mesh, material and integration rule. Mesh by part ID which is cross-referenced in *ELEMENT_ cards; material by MAT ID+ EOS ID + HOURGLASS ID; integration rule or element formulation by SECTION ID.

One could immediately notice this definition does not fit ALE well. In ALE models, typically there is a mesh which multiple fluids (materials) flow in it. In Lagrange simulations mesh and material belong to one integral part and are not separable. However, in ALE simulations they are totally irrelevant.

ALE setup inherited the PART definition from Lagrange model setup. This brought considerably confusions as the PART in an ALE model can refer to mesh, can refer to material or sometimes both. One could easily be lost when filling out the PID fields in all those ALE keywords.

When designing the S-ALE setup, we decided to separate the *PART usage into two categories: mesh part and material part. An ALE part refers either mesh or material. It can not be both. Material part wraps up a set of MAT+EOS+HOURGLASS. Its ID is only used once in the *ALE_MULTI-MATERIAL_GROUP card. All other usages are illegal. Mesh part ID (DPID in *ALE_STRUCTURED_MESH) is a collection of solid elements and nodes. It is used in all other ALE keywords to refer mesh.

Back to the above mention item 2 and item 3. In case an ALE setup contains multiple parts, by default, each part's mesh are filled by the material referred in the part definition. Such cases need to be picked out and processed. For item 3, "ambient parts" definitely contain both mesh and material and can not live with S-ALE part definition. So we divided them into two concepts and listed them in *BOUNDARY_AMBIENT. Mesh wise we converted a part into a solid element set; material wise we specify it by using ALE_MULTI-MATERIAL_GROUP ID (AMMG ID).

Deleting S-ALE mesh in the run

ALE is mostly used to solve engineering problems of large momentum transfer during a short period of time. Most of times that momentum transfer happens at the beginning of simulation and its effects, if not finish soon, fades away very fast. A way to bypass the unnecessary ALE element calculation, advection and fluid structure interaction (FSI) becomes intriguing in such cases.

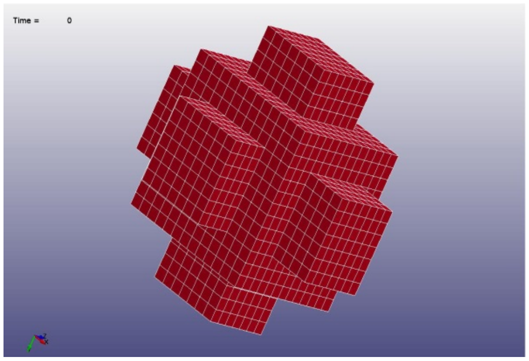
There is a recently added TDEATH field added in *ALE_STRUCTURED_MESH card. When the time reaches TDEATH, all ALE elements in this S-ALE mesh are deleted; advection bypassed; FSI stopped. Please note no extra effort needs to stop FSI. All FSI cards such as *CONSTRAINED_LAGRANGE_IN_SOLID, *ALE_COUPLING_NODAL_ are internally searched and those related to deleted mesh are picked out and terminated.

Merging S-ALE meshes

S-ALE allows for multiple meshes. Those meshes can be next to or far away from each other. They can occupy different spatial domain or overlap with each other. Each S-ALE mesh generated by *ALE_STRUCTURED_MESH card is independent. Calculations are done independently in each S-ALE mesh without knowing what is happening in other meshes.

Now there is an exception. Any two of such independent meshes can be jointed at their mesh interfaces, of course, provided, firstly their meshing have to be the same at the interface, secondly the distance between those two meshes are zero or negligible. To do this, one can simply setting the DPID field in the two or more *ALE_STRUCTURED_MESH cards to the same value. All nodes at the overlapping area at the mesh interface are merged and element connectivity rebuilt. This feature enables S-ALE solver to solve problems using mesh contains several boxes.

Below is a figure shows totally seven meshes merged. This example input deck could be found at http://ftp.lstc.com/anonymous/outgoing/hao/sale/models/meshmerge/salemerger.tar.



Mesh trimming

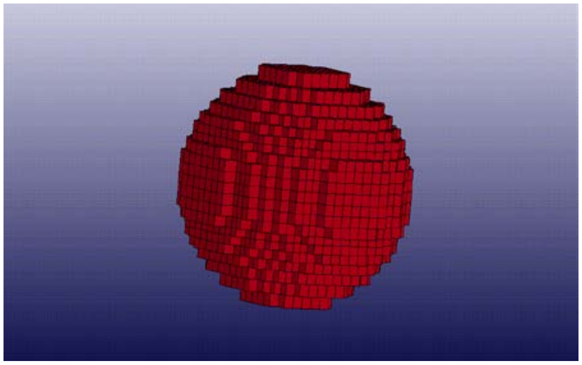
S-ALE supports limited types of meshes. The mesh is either of a rectangular box shape or a combination of connecting rectangular boxes (through mesh merging). This limitation is a tradeoff for the simplicity of geometry information. For certain cases, using a big box trying to cover up the structures could be quite wasteful. For example, let us say we build a model studying raindrops hitting on the windshield. On one hand, we need to make the mesh box large enough to cover the curved windshield; on the other hand, we know any element a few elements far away from the windshield is not necessary.

This motivated us to implement the mesh trimming feature. It is to trim the S-ALE mesh to bring the savings on running time and memory. To use this, we need to add the trimming subsection in *ALE_STRUCTURED_MESH card. The trimming subsection is inserted after the first two cards and marked by "_TRIM". A subsection can contains as many as cards and ends when another subsection is read in or when it reaches the end of keyword. There are now totally two subsections in *ALE_STRUCTURED_MESH card -- "_TRIM" and "_MOTION" which is described in the next section.

There are currently six commands in the _TRIM subsection. They are "PARTSET", "SEGSET", "PLANE", "CYLINDER", "BOX" and "SPHERE". Please refer to the LS-DYNA manual for exact usage.

```
    *ALE_STRUCTURED_MESH
$   mshid     pid    nbid     ebid
       1        1   200001   200001
$    nptx     npty     nptz     nid0    lcsid
     1001     1001     1001        1      234
_TRIM
$  command     oper     flip      nid    radius
SPHERE                             5      0.10
$  command     oper     flip   psetid    radius
$PARTSET                            3      0.03
```

Below is a figure shows a box mesh trimmed by "SPHERE". This example input deck could be found at http://ftp.lstc.com/anonymous/outgoing/hao/sale/models/meshtrim/saletrim.tar. For this example, out of total 9261 elements, 3614 elements are deleted. This brought a nearly 40% reduction on simulation time.



Mesh motion

S-ALE does not support *ALE_REFERENCE_SYSTEM_GROUP card which is used to move the ALE mesh. Instead it allows mesh to move and rotate by prescribing the motion on the origin node (NID0) and the three nodes used to define local coordinate system (LCSID). While it satisfies most user problems we have so far, it does not allow for mesh to follow the mass center of certain fluid. So we added a mesh motion subsection.

Currently it only has one command, “FOLLOW_GC” which makes the mesh to follow mass center's motion. The mesh motion subsection can contain multiple cards and started with “_MOTION”.

Below is an example on using the _MOTION subsection. The full example input deck could be found at http://ftp.lstc.com/anonymous/outgoing/hao/sale/models/meshtrim/2bagstrim.tar.

```
*ALE_STRUCTURED_MESH

$    mshid      pid      nbid      ebid

        1        1     10001     10001

$     nptx      npty      nptz      nid0     lcsid

     1001      1001      1001      4001       234

_TRIM

$  command     oper     flip

PARTSET                          3      0.03

_MOTION

$  command   ammgset

FOLLOW_GC         1
```

Conclusions

We introduced several notable new developments in LS-DYNA Structured ALE solver in this paper. Those features are added solely to reduce the simulation time and memory usage. In recent years, we observed a rapid growth in the ALE model size and the accompanying demands in speed and memory usage. Nowadays, models as large as 60 million elements are commonly used by the S-ALE users and we expect the trend to follow. The S-ALE developer at LSTC is committed to continually work with our users to improve.