Recent Developments in LS-DYNA S-ALE

Hao Chen

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.

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

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. Hence we introduced a new input keyword *ALE_STRUCTURED_MESH_TRIM to perform the trimming operation at the initialization phase. This keyword has two cards and could be used multiple times. In case of multiple times, each * ALE_STRUCTURED_MESH _TRIM keyword represents one “trim” or “untrim” process; and each is processed at the order of appearance in the input deck.

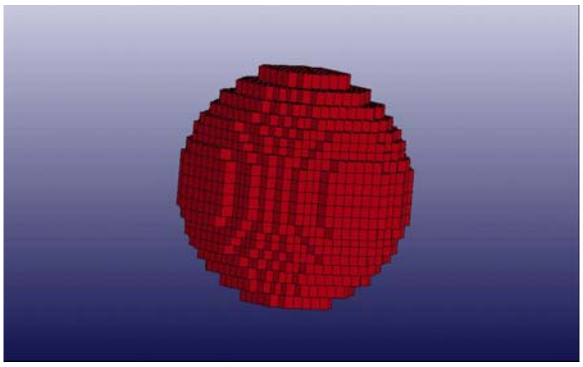
The field “oper” means “operation” and could take either “trim” or “keep”. “trim” is to trim the elements inside or outside of certain geometry. The geometry could be either simply geometry such as plane, box, cylinder, sphere, or some complex shape represented by a list of segments or shell parts. Contrarily, “keep” is to add the elements into the mesh. Those elements might or might not be trimmed by previous commands from the mesh; but are wanted in the final mesh. By combining these two operations, users could enjoy quite some flexibility in the mesh construction process.

There are currently six trimming commands. 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		
$ cptx	cpty	cptz	nid0	lcsid	
1001	1001	1001	1	234	

*ALE_STRUCTURED_MESH_TRIM					
$ mshid	command	oper	flip	nid	radius
1	SPHERE			5	0.1
$ mshid	command	oper	flip	psetid	offset
1	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 keyword *ALE_STRUCTURED_MESH_MOTION. Currently it only has one option, "FOLLOW_GC" which makes the mesh to follow mass center's motion. More mesh motion options are expected to be added per user's request; very possibly on the mesh expansion/contraction.

Below is an example using *ALE_STRUCTURED_MESH_MOTION. The input deck is available at http://ftp.lstc.com/anonymous/outgoing/hao/sale/models/meshtrim/2bagstrim.tar.

*ALE_STRUCTURED_MESH					
$ mshid	pid	nbid	ebid		
1	1	10001	10001		
$ cptx	cpty	cptz	nid0	lcsid	

1001	1001	1001	4001	234	

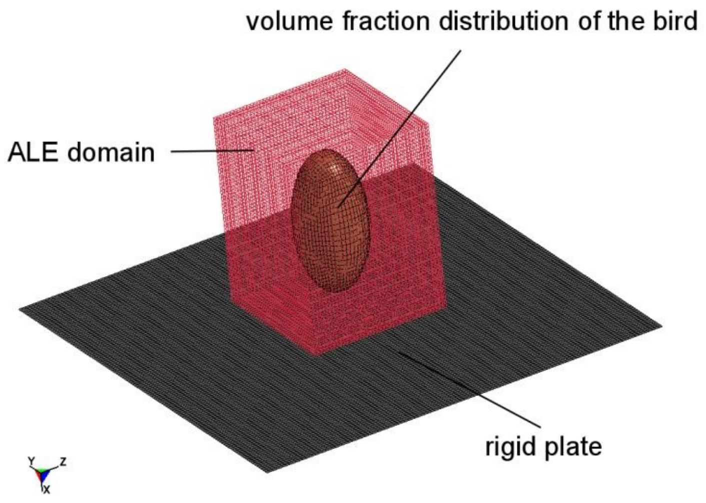
*ALE_STRUCTURED_MESH_TRIM					
$ mshid	command	oper	flip	psetid	offset
1	PARTSET			3	0.03

*ALE_STRUCTURED_MESH_MOTION					
$ mshid	option	AMMGSET			
1	FOLLOW_GC	1			

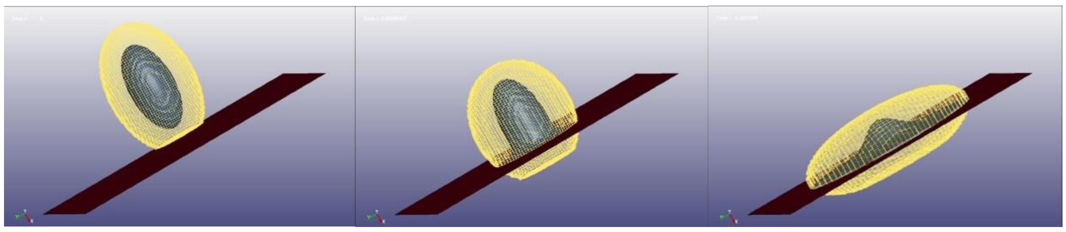
Comparison between ALE, S-ALE and trimmed S-ALE

Now we use a bird hitting plate example to show how we use trimming and mesh motion features. Also, we do a comparison on the results and running time for the three cases: ALE solver, S-ALE solver and S-ALE solver with trimmed mesh. The input deck is at http://ftp.lstc.com/anonymous/outgoing/hao/sale/models/awgbirdstrike/.

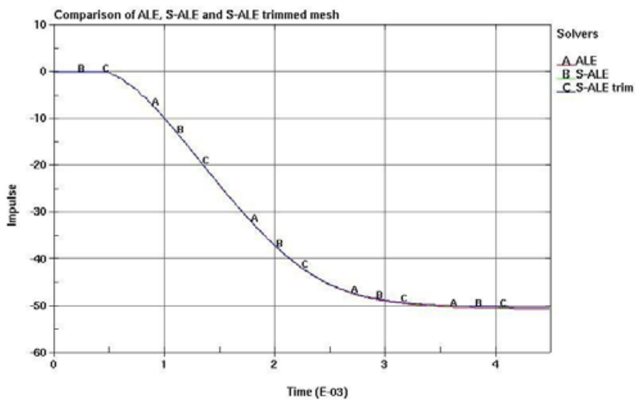
The following sketch shows the problem. A bird of ellipsoid shape is defined as ALE multi-material group (AMMG 1) in a box mesh. It is travelling at some initial speed and hit the rigid plate. In all three cases, the mesh moved the same way. We took the untrimmed ALE case as the base. And we constructed the second case with nothing changed but to use S-ALE solver. Then the third case was built by applying the trimming card.



The mesh motion for the trimmed S-ALE case could be seen in the following plots.



Below is the total impulse recorded on the rigid plate. We could see that for all three cases the curves are all on top of each other. This told us that neither the S-ALE solver nor the mesh trimming process affected the result.



Now let us look at the running time. From the following table, we could see S-ALE solver and the mesh trimming process brought huge time savings. Using S-ALE solve achieved a 40%-44% time reduction. And with mesh trimming the reduction became even bigger. It saved 65% of running time for both SMP 1core and MPP 4 cores.

METHODS	#of Elements	SMP 1 Core	SMP Savings	MPP 4 Core	MPP Savings
ALE	84800	1204 s		321 s	
S-ALE	84800	675 s	44%	191 s	40%
S-ALE trim	43219	426 s	65%	112 s	65%

Initial Volume Filling

A new keyword was added in S-ALE solver to do the volume filling in the initial S-ALE mesh -- *ALE_STRUCTURED_MESH_VOLUME_FILLING (ASMVF). It serves the same purpose as *INITIAL_VOLUME_FRACTION_GEOMETRY (IVFG) and has a quite similar format. However, it is a better and much more flexible alternative to the IVFG card.

It is different in a few ways. First, IVFG card could only appear once in the input file. Only the last one is honored and all previous appearances are simply ignored. While it might be enough for ALE simulations as ALE solver only supports one mesh, it becomes a concerning issue for S-ALE solver. We could have multiple meshes in one model but the IVFG card only allows us to perform volume filling on 1 part.

Secondly, IVFG has only 1 background material. It fills all element in the ALE mesh with that background material first. And then allows the user to put in multiple “switch” operations to switch that background material in certain geometry to certain different material. While it is sufficient as long as users know its limitation and perform the volume filling in some special modelspecific way, for new users or some difficult problems it could take quite some time and a few iterations before the volume filling could be performed correctly.

Thirdly, for MPP runs, IVFG does the volume filling at PHASE 1 and stores the volume filling result which is the volume fraction for each ALE multi-material group (AMMG) in each element. Later at PHASE 4, this database is read in and stored. This process is wasteful in two ways. First, at PHASE 1 only 1 core is active. For large models, the volume filling could be very time consuming. It could easily take hours before the run could even start. Secondly, writing out and reading back the volume fraction database is also time consuming as it is I/O intensive.

The *ALE_STRUCTURED_MESH_VOLUME_FILLING, is better in all the above mentioned 3 points. It supports multiple cards. And each card has a switch from/to AMMG pair so we do not have to always start from certain special background material. Also, it does the volume filling at PHASE 4 while all MPP cores are active. So each core does the volume filling on the S-ALE mesh it owns. So the new card provides better performance and more flexibility; requires less running time and resources. We recommend the S-ALE users to always use the new *ALE_STRUCTURED_MESH_VOLUME_FILLING card. As a matter of fact, in the current development version and the incoming release, the IVFG card will be internally converted to multiple ASMVF cards in S-ALE solver for better performance.

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.