FEA Information International News

www.feainformation.com

Issue August, 2001

A free e-mail monthly publication

FEA Information International News:

Managing Editor
Trent Eggleston
Feature Editor
Marsha Victory
Technical Writer
David Benson
Art Director:
Wayne Mindle

DO NOT OPEN e-mail from:
anonymous@feainformation.com

FEA Information Inc. DOES NOT USE the above address.

FEA Information Participants

Multimaterial Eulerian Hydrodynamics

Part 2: The Eulerian Step (Pure Elements)

Author: David J. Benson

The Eulerian step accounts for the material transport between adjacent elements. It does considerably more than just calculate the volume of material transported, namely it has to account for the effect of material entering and exiting has on the values of all the state variables in the calculation. At the very minimum, this includes the three velocity components, the density, and the internal energy. Most calculations require the full stress tensor for modeling the material strength, so the stress tensor (six components for the symmetric Cauchy stress) and the history variables associated with the material strength model (e.g., the equivalent plastic strain) are also transported. In total, the typical Eulerian step is transporting at least ten variables for each material in addition to the momentum. The number of transport variables rapidly increases if the material models are complicated (e.g., the back stress for kinematic hardening adds six variables) or if the stress rate requires the deformation gradient (nine variables if ***F***is transported, possibly more if the polar decomposition is transported). Unsurprisingly, the Eulerian step is substantially more expensive than the Lagrangian step.

Although the theoretical genesis of the Eulerian step is the convective term in the material derivative, the transport algorithms are based on projecting the solution from a slightly distorted Lagrangian mesh on to the spatially fixed Eulerian mesh. The Eulerian step therefore is closely related to the general remap problem faced by rezoning and adaptive meshing algorithms. A general remap is many times more expensive than a typical transport algorithm because any element in the Lagrangian mesh can contribute to the Eulerian mesh. Eulerian transport algorithms assume that an element can contribute material only to its adjacent neighbors, greatly simplifying the algorithms. The simplification comes at the price of stability: the time step is restricted to the shortest length of time required for material to flow across an element. In most calculations, the wave speed is much higher than the material velocity, and the Lagrangian step determines the stable time step. However, for some problems, such as shaped charges, the mateial velocity can exceed the wave speed in the material and the Eulerian step determines the time step.

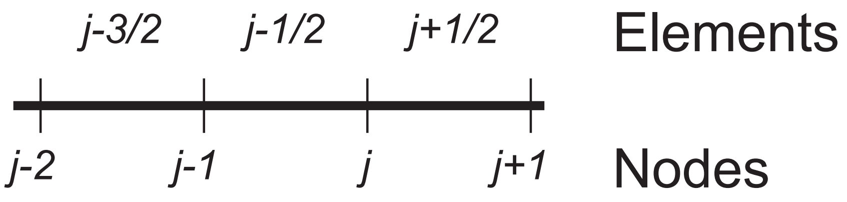


Figure 1. One-dimensional mesh.

For the moment, the discussion will be restricted to transport in one dimension. This is not much of a restriction since most transport algorithms originally started as one-dimensional algorithms and then extended to multi-dimensional problems. The nodes are numbered sequentially from left to right (see Figure 1) and pairs of adjacent nodes define the elements. Assume that the velocity is centered at the nodes, and the material variables are located at the centroids of the elements (Godunov methods have all the variables located at the element centroids). For convenience, the element defined by nodes *j* and *j*+1 is *j*+1/2. The general update formula for a variable ϕ in element *j*+1/2 is

$$\phi_{j+1/2}^{n+1} = \left[\phi_{j+1/2}^{n} V_{j+1/2}^{n} + \phi_{j}^{n} v_{j} - \phi_{j+1}^{n} v_{j+1}\right] / V_{j+1/2}^{n+1}$$

where V is the element volume, v is the transport volume between adjacent elements, the superscript refers to the time step number and the subscript is the location in the mesh. The calculation of the transport volumes is essentially geometrical; in one dimension, the transport volume is just the incremental displacement between time step n and n+1. In general, the transport volume is just the volume swept out by the edge or face of an element during a time step.

The values of ϕ in the transport volumes, ϕ_j^n and ϕ_{j+1}^n , are calculated by the transport algorithm. The simplest stable algorithm is the donor cell method. When the material flows from left to right, the value of ϕ_j^n is $\phi_{j-1/2}^n$, and for flow from right to left, it's $\phi_{j+1/2}^n$. The donor cell method is a first order accurate, monotonic, upwind method. First order accuracy means that the solution smears out too rapidly to be acceptable for most applications. One of the problems with early transport algorithms was they introduced oscillations into the solution, which could lead to physically wrong solutions such as elements having negative plastic strain. Monotonic transport methods are guaranteed not to introduce any wiggles into the solution. Mathematical variations on the theme of monotonic methods are total variation diminishing (TVD) and essentially non-oscillatory (ENO) methods. Upwind methods use information upstream from the transport volume to calculate the value of ϕ in the transport volume. This idea has a lot of physical appeal and is crucial for the stability.

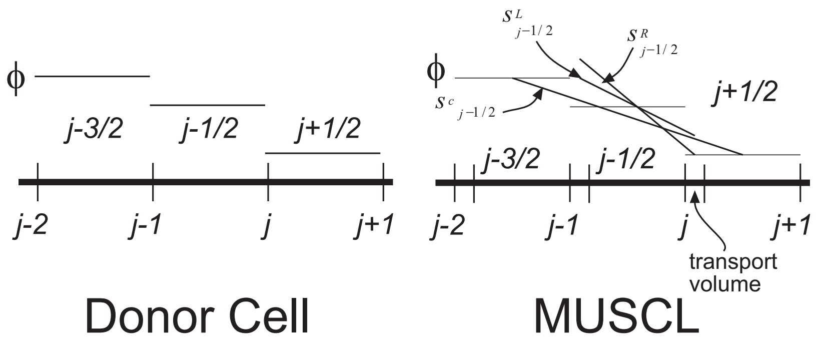


Figure 2. The donor cell and MUSCL algorithms.

Bram van Leer developed the MUSCL algorithm, which is currently the most popular method for second order accurate, monotonic transport in hydrocodes. Like the donor cell method, it uses upstream information. Instead of a piecewise constant distribution of ϕ in the element, the MUSCL algorithm approximates ϕ with a piecewise linear function (see Figure 2).

$$\phi(x) = s_{j-1/2}(x - x_{j-1/2}^n) + \phi_{j-1/2}^n$$

Second order accuracy is achieved in the smooth parts of the solution by using a second order approximation of the slope, and monotonicity is guaranteed by limiting the slope to prevent it from creating a new maximum or minimum. For a uniform mesh, a simple second order approximation of the slope is

$$s^c{}_{j-1/2} = \left(\frac{\phi_{j+1/2}^n - \phi_{j-3/2}^n}{x_{j+1/2}^n - x_{j-3/2}^n} \right)$$

and it is limited by looking at the maximum slopes it can have to the left and right without introducing a new extremum. The value of ϕ is evaluated in the center of the transport volume, which is located at $\tilde{x}_j = x_j^n - 1/2\Delta t u_j^{n-1/2}$, so the limiting slopes for $s_{j-1/2}$ are

$$s_{j-1/2}^L = \frac{\phi_{j-1/2}^n - \phi_{j-3/2}^n}{x_{j-1/2}^n - \tilde{x}_{j-1}^n}$$

$$s_{j-1/2}^R = \frac{\phi_{j+1/2}^n - \phi_{j-1/2}^n}{\tilde{x}_j^n - x_{j-1/2}^n}.$$

The final slope is

$$s_{j-1/2} = \tfrac{1}{2}\left[\operatorname{sgn}(s_{j-1/2}^L) + \operatorname{sgn}(s_{j-1/2}^R)\right]\min\left(\left|s_{j-1/2}^L\right|, \left|s_{j-1/2}^c\right|, \left|s_{j-1/2}^R\right|\right).$$

The slope chosen in Figure 2 would be $s_{j-1/2}^c$ because it has the shallowest slope. Note that although the line used to define the second order, centered approximation of the slope doesn't pass through $\left(x_{j-1/2}^n, \phi_{j-1/2}^n\right)$, the interpolation function for ϕ in the element does, and therefore the integrated average of ϕ over the element equals $\phi_{j-1/2}^n$.

Operator splitting extends the one-dimensional transport algorithm to logically regular meshes. One-dimensional sweeps are made along the mesh lines in a sequence of directions each time step, and the sequence is reversed the following time step to prevent aliasing errors from developing. In addition to being a simple method for extending a one-dimensional method to multi-dimensions, operator splitting also introduces corner coupling into the transport. Consider a fluid moving along a diagonal direction of a two-dimensional, logically regular mesh. Fluid moves from one element to a diagonal neighbor, and they share a node, but not a face. Clearly some material flows from one to the other even though the cross-sectional area connecting them is zero, and this flow is the corner coupling.

Sweeps along the mesh directions are possible with unstructured meshes. In most cases, the transport is calculated in all directions at once. Applying a one-dimensional method in each direction independently results in no corner coupling. Material tends to spread out in the direction perpendicular to the diagonal flow. One way to counteract this problem with unstructured meshes is to generate a mesh such that material doesn't flow along the mesh diagonals. This approach is sometimes not feasible with a logically structured mesh, but it usually is feasible with unstructured meshes. Corner coupling is, therefore, much more important for codes restricted to logically regular meshes.

The third part of this series will discuss reconstructing and tracking the material interfaces in an Eulerian hydrocode.

1960 Batch Computing

Batch Computing is a style of computer usage that was popular in the early 1960's. During this era computer input was made by means of an "input deck" of punch cards.

The state of the art of this era was a card based input system, and users had to line up one at a time to use the computer. Each program "deck" was fed at the computer in a rather serial manner, and when the output was complete, the user was notified and came to the computer center to pick up the output. The original Finite Element Programs were developed in this style, and were run this way for many years on departmental computers. These same computers running the Finite Element Programs were also being used for billing and payroll. This sharing of departmental computers for two different purposes brought about some positive, and not so positive, characteristics of this computing style. The computer input was provided using card decks and if the decks were dropped they required a complete hand resort of the order, or to make changes in the program, new cards required sequential position within the card input deck. Although these tasks were time consuming and didn't provide an efficient solution, from a system management point of view, these systems worked very well. The positive aspect of this type of computer system was that the systems administrator had complete control over which jobs ran, and at what time. Therefore, systems resources where easy to manage and balance.

1970 Time Shared Computing for Engineers

In the early 1970's, commercial implementations of "time sharing" began to become popular on smaller "departmental" computers such as the Digital Equipment Corporation's PDP line, and the Data General Nova line. These systems differed from the earlier "batch" style computers because engineers could now sit in front of a "character based" terminal and interact with the computer in real-time. The word "time-shared" came from the fact that the computer's operating system was designed to provide each engineer with a small portion (slice) of time presenting the appearance of a dedicated system. Although the operating system appeared dedicated it was, in fact, being used by many concurrent users. This concept effectively replaced the older "batch" method.

Throughout the 1970's time-sharing was the undisputed method for providing cost effective computer systems and applications to engineers. They now had an improved mechanism for interacting with the computer that precluded the need for cumbersome input card decks. They could create input files via the computer terminal, and submit the application to run using a queuing mechanism. However, as more and more engineers logged on to the single system, the ability for its central processor to present an "interactive session" became strained. This led to a period of discontent, and bred a culture of nocturnal programmers and engineers who would work at night when interactive computer response was best.

Despite the minor inconveniences associated with response time, the time-sharing method became a defacto standard in the 1970's for computer operating systems including the very popular Unix operating system developed at Bell Labs. Most of the advantages of a centralized file and operating system that the older “batch” style of computing were retained during the time sharing era. This centralized approach allowed engineers to share files, information, and other resources such as system memory and disk without the need for direct user intervention, providing a high degree of collaboration among engineers.

Personal Computing

In the early 1980’s a number of technologies became available at low enough price points to consider a new model for the computer. The key technologies during this period included the microprocessor, (developed by companies such as Motorola with the 68000 and Intel with the x86 line) the local area network (LAN), and the lower cost higher density disk and memory components. These developments enabled systems engineers to build a personal computer system that provide each user with their own central processor and storage system, connected to other systems over a high speed network. These developments happened in two primary areas simultaneously, the Engineering Workstation, powered by the Motorola 68000 and Unix operating system, and the Personal Computer (PC) powered by the Intel X86 microprocessor and the DOS operating system.

The emergence of the Engineering Workstation represented a significant inflection point in the computer aided design and development market. With its bit mapped graphics capabilities, the Engineering Workstation could not only handle the computational and storage needs of heavy applications such as ANSYS, but it also had the ability to display a graphical output far superior to the traditional numerical table output common on the centralized systems of the period. Furthermore, the Engineering Workstation was a personal device providing better interactive response than the time-shared systems. With the development of the low cost microprocessor, Engineering Workstations companies were popping up all over the world including companies such as Apollo Computer, Sun Microsystems, and Silicon Graphics.

It was during this period that the ANSYS application began to evolve to meet this new architectural opportunity. In both the time-shared and batch modes, ANSYS was used as a “offline” computational engine for providing results. The input, solution, and output were treated as three distinct entities. The user would create an input file offline, submit it to the central queuing facility for computation at a later time, and wait for the output to be created. With the introduction of the Engineering Workstation, this model began to evolve to a more ‘transaction oriented’ type of operation. Users would interact rather dynamically with their own “local” copy of the application during the input, solution, and output stages. This led to a much tighter coupling of the ANSYS application in an effort to provide users with a complete interactive simulation environment at their desktop.

However the Engineering Workstation and the Personal Computer also have their limitations. Although the central processor for these systems is local, and typically provides much better interactive response than a shared system, the memory and disk also remains local, and thus isolated from the rest of the network. Although the computers are attached via a LAN, the ability for one system to view all storage as one entity has been elusive at best. As a result, when either memory or disk storage is purchased for the Personal Computer or Engineering Workstation, the resource is of primary use to the individual user of that system. With applications like ANSYS that consume large amounts of computational and storage capacity, the cost of these personal systems can be quite high, especially when purchasing one for each user of a large engineering department. The result is often to constrain the desktop configuration in an effort to contain costs. A popular method for configuring desktop systems is to use the systems requirements for the CAD package as a guideline. This approach inevitably results in an under configured system for complex simulations.

Paul Bemis, is the author of this series article and the manager of The ANSYS, Inc. e-CAE.com ASP Program that provides:

- A mechanism for running **ANSYS** simulations and/or **LS-DYNA** simulations on large parallel compute servers, at a remote data center site using the internet.
- The system has been developed to allow engineers the ability to run remote simulations with specific controls on job execution parameters.
- The solution uses “state of the art” security and systems infrastructure technology from providers including Sun, Hewlett Packard, Silicon Graphics, Cisco, and others.
- The service is ideal for engineers and companies requiring occasional “surge” capacity for time critical simulations, or periodic simulations of large models.

Key e-CAE.com seccurity features:

- HTTP or HTTPS (Secure) access
- Strict account & file controls.
- Full data communication encryption.
- Secure Socket Layer (SSL).
- Continuous monitoring on all operations.

Key e-CAE.com benefits include:

- Large problem simulations "on demand" via the Web.
- Fast, predictable results on new high performance compute servers.
- Easy-to-use via web browser.
- Optional VPN and dedicated connections available.
- Efficient, secure systems management with full back-up and archival capability.
- Reduce cost and improve reliability of total engineering simulation solution.
- Elimination of long procurement processes and cost justifications.
- Reduced “cost of entry” into engineering simulation.
- Easy and prompt ANSYS application versions upgrades.
- Improved user integration and collaboration across geographies and divisions.

Part II: Computing in the 90’s, Client Server, Distributed Web Based
Part III: FAQ on Web Based Distributed Simulation

The ANSYS, Inc. e-CAE.com ASP Program - A mechanism for running **ANSYS** simulations and/or **LS-DYNA** simulations on large parallel compute servers, at a remote data center site using the internet. For Complete Information Contact: Paul Bemis **– [paul.bemis@ansys.com]**

Contact Modeling in LS-DYNA- Some Recommendations – Part 1

1.0 Introduction

Contact treatment forms an integral part of many large-deformation problems. Accurate modeling of contact interfaces between bodies is crucial to the prediction capability of the finite element simulations. LS-DYNA offers a large number of contact types. Some types are for specific applications, and others are suitable for more general use. Many of the older contact types are rarely used but are still retained to enable older models to run as they did in the past. Users are faced with numerous choices in modeling contact. This document is designed to provide an overview of contact treatment in LS-DYNA and to serve as a guide for choosing appropriate contact types and parameters.

2.0 How Contact Works

In LS-DYNA, a contact is defined by identifying (via parts, part sets, segment sets, and/or node sets) what locations are to be checked for potential penetration of a “slave” node through a “master” segment. A search for penetrations, using any of a number of different algorithms, is made every time step. In the case of a penalty-based contact, when a penetration is found a force proportional to the penetration depth is applied to resist, and ultimately eliminate, the penetration. Unless otherwise stated, the contacts discussed here are penalty-based contacts as opposed to constraint-based contacts. Rigid bodies may be included in any penalty-based contact but in order that contact force is realistically distributed, it is recommended that the mesh defining any rigid body be as fine as that of a deformable body.

Though sometimes it is convenient and effective to define a single contact that will handle any potential contact situation in a model, it is permissible to define any number of contacts in a single model. It is generally recommended that redundant contact, i.e., two or more contacts producing forces due to the same penetration, be avoided by the user as this can lead to numerical instabilities.

To enable flexibility for the user in modeling contact, LS-DYNA presents a number of contact types and a number of parameters that control various aspects of the contact treatment. In the following sections, contact types are first discussed with recommendations regarding their application. A description of the contact parameters then presented.

3.0 Contact Types

In crash analysis, the deformations can be very large and predetermination of where and how contact will take place may be difficult or impossible. For this reason, the automatic contact options are recommended as these contacts are non-oriented, meaning they can detect penetration coming from either side of a shell element. Automatic contact types in LS-DYNA are identifiable by the occurrence of the word “AUTOMATIC” in the *CONTACT command. The contact search algorithms employed by automatic contacts make them better-suited than older contact types to handling disjoint meshes. In the case of shell elements, automatic contact types determine the contact surfaces by projecting normally from the shell mid-plane a distance equal to one-half the ‘contact thickness’. Further, at the exterior edge of a shell surface, the contact surface wraps around the shell edge with a radius equal to one-half the contact thickness thus forming a continuous contact surface. We sometimes refer to this offsetting of the contact surfaces from shell mid-planes as considering “shell thickness offsets”. The ‘contact thickness' can be specified directly or scaled by the user using optional parameters in the contact definition. If the contact thickness is not specified by the user, the contact thickness is equal to the shell thickness (or, in the case of single surface contacts, the minimum of the shell thickness and element edge length). In like fashion, the contact surface for beam elements (where beam contact is considered) is offset from the beam centerline by the equivalent radius of the beam cross-section. Because contact surfaces are offset from shell mid-planes and from beam centerlines, it is **extremely important** that appropriate gaps between shell and beam parts be modeled in the finite element geometry in order to account for shell thickness and beam cross-section dimensions. Not doing so will result in initial penetrations in the contact surfaces. LS-DYNA will make one pass to eliminate any detected initial penetrations by moving the penetrating slave nodes to the master surface. Not all initial penetrations will necessarily be removed and this can lead to nonphysical contact behavior. Time taken in setting up an accurate initial geometry is always time well spent.

Most contact types in LS-DYNA place a limit on the maximum penetration depth that is allowed before the slave node is released and its contact forces are set to zero. This is done mainly in automatic contact types to prevent large contact forces from developing in the opposite sense should the slave node pass through a shell mid-plane. This maximum penetration depth is tabulated for various contact types in Table 6.1 of the Version 960 User's Manual. Sometimes automatic contact interfaces appear not to work because this contact threshold is reached early in the simulation. This often occurs if extremely thin shell elements are included in the contact surface. In these cases, contact failure can usually be prevented by scaling up the default contact thickness or setting the contact thickness to a value larger than the shell thickness. Alternately, setting SOFT=1 (discussed later) will often correct the problem.

3.1 One-Way Treatment of Contact

One-way contact types allow for compression loads to be transferred between the slave nodes and the master segments. Tangential loads are also transmitted if relative sliding occurs when contact friction is active. A Coulomb friction formulation is used with an exponential interpolation function to transition from static to dynamic friction. This transition requires that a decay coefficient be defined and that the static friction coefficient be larger than the dynamic friction coefficient. The "one-way" term in one-way contact is used to indicate that only the user-specified slave nodes are checked for penetration of the master segments. One-way contacts may be appropriate when the master side is a rigid body, e.g., a punch or die in a metal stamping simulation. A situation where one-way contact may be appropriate for deformable bodies is where a relatively fine mesh (slave) encounters a relatively smooth, coarse mesh (master). Other common applications are beam-to-surface or shell-edge-to-surface scenarios where the beam nodes or the shell edge nodes, respectively, are given as the slave node set. There are a number of keyword options that activate one-way contact.

For contact between an airbag (slave) and segmented rigid dummy model (master), one of the following two contact types are often employed:

*CONTACT_AUTOMATIC_NODES_TO_SURFACE (a5)
*CONTACT_AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE (a10)

For metal stamping, special one-way forming contacts are recommended with the workpiece defined on the slave side:

*CONTACT_FORMING_NODES_TO_SURFACE (m 5)
*CONTACT_FORMING_ONE_WAY_SURFACE_TO_SURFACE (m 10)

Orientation is automatic with forming contacts. The rigid tooling surface can be constructed from disjoint element patches where contiguous nodal points are sometimes merged out, but not always. These patches are not assumed to be consistently oriented; consequently, during initialization, the reorientation of these disjoint element patches is performed. Forming contact tracks the nodal points of the blank as they move between the disjoint element patches of the tooling surface. Penalty forces are used to limit penetrations. Generally the ONE_WAY_SURFACE_TO_SURFACE option is recommended since the penetration of master nodes through the slave surface is considered in adaptive remeshing. Without this feature, adaptive remeshing may fail to adequately refine the mesh of the blank to capture sharp details in the master surface, and the master surface will protrude through the blank.

When the surface orientations are known throughout the analysis, the following non-automatic contact types may be effective:

*CONTACT_NODES_TO_SURFACE (5)
*CONTACT_ONE_WAY_SURFACE_TO_SURFACE (10)
*CONTACT_CONSTRAINT_NODES_TO_SURFACE (18)
*CONTACT_ERODING_NODES_TO_SURFACE (16)

If there is a possibility that the nodes of the slave surface can physically end up behind the master surface, these contact types should be avoided. Shell thickness offsets may or may not be considered with these non-automatic contact types (see SHLTHK in *CONTROL_ CONTACT). If shell thickness offsets are inactive (default), then the old node-to-surface contact treatment from public domain DYNA3D is used for contact types 5 and 10 above where incremental searching is used to locate potential master segments for any given slave node. This searching technique uses segment connectivity; therefore, the master surface must not be disjoint. If the geometry of the surfaces have sharp angles or if the segments are very badly shaped, the searching algorithm can fail to find the proper master segment. If the shell thickness offsets are active, SHLTHK>0, the master surface is projected based on nodal normal vectors, and the location of the slave node on a master segment is determined by using global segment-based bucket sorting; therefore, the master surface can be disjoint and sharp edges and bad element shapes do not create significant problems in the searching. The use of nodal normal vectors to project the master surface is quite expensive in CPU costs, but has an advantage that the projected master surface is continuous even for convex surfaces. Until the FORMING contact types were developed, types 5 and 10 contacts with shell thickness offsets were often the contact of choice for sheet metal stamping.

The contact type:

*CONTACT_CONSTRAINT_NODES_TO_SURFACE (18)

is similar in treatment to *CONTACT_ NODES_TO_SURFACE with shell thickness offsets. Being constraint-based rather than penalty-based, type 18 contact cannot be used with rigid bodies. The forces are computed to keep the slave nodes exactly on the master surface (zero penetration). In general, this contact has never been as stable as the penalty-based contacts and is, therefore, not recommended.

Eroding contact types are recommended whenever solid elements involved in the contact definition are subject to erosion (element deletion) due to material failure criteria. These eroding contacts contain logic which allow the contact surface to be updated as exterior elements are deleted. In *CONTACT_ERODING_NODES_TO_SURFACE, the slave side of the contact should be defined using a node set containing all the nodes (not just nodes on the outer suface) of the slave side part(s).

3.2 Two-Way Treatment of Contact

This contact works essentially the same way as the corresponding one-way treatments described above, except that the subroutines which check the slaves nodes for penetration, are called a second time to check the master nodes for penetration through the slave segments. In other words, the treatment is symmetric and the definition of the slave surface and master surface is arbitrary since the results will be the same. There is an increased cost of approximately a factor of two due to the extra subroutine calls.

In crash analysis, the contact type …

*CONTACT_AUTOMATIC_SURFACE_TO_SURFACE (a3)

is a recommended contact type since, in crash simulations, the orientation of parts relative to each other cannot always be anticipated as the model undergoes large deformations. As mentioned before, automatic contacts check for penetration on either side of a shell element.

For metal forming simulations, the contact type …

*CONTACT_FORMING_SURFACE_TO_SURFACE (m 3)

is available but is generally not used in favor of the one-way forming contacts.

The two-way (symmetric) counterparts to the previously discussed contact types 5, 18, and 16 are:

*CONTACT_SURFACE_TO_SURFACE (3)
*CONTACT_CONSTRAINT_SURFACE_TO_SURFACE (17)
*CONTACT_ERODING_SURFACE_TO_SURFACE (14).

3.3 Tied Contact (Translational DOF only, No Failure, No Offset)

In tied contact types, the slave nodes are constrained to move with the master surface. At the beginning of the simulation, the nearest master segment for each slave node is located based on an orthogonal projection of the slave node to the master segment. If the slave node is deemed ‘close’ to the master segment based on established criteria, the slave node is moved to the master surface. In this way, the initial geometry may be slightly altered without invoking any stresses. It is always recommended that tied contacts NOT be defined by part Ids but rather by node/segment sets. In this way, the user has more direct control over what gets tied to what and thus can prevent unintended constraints. As the simulation progresses, the isoparametric position of the slave node with respect to its master segment is held fixed using kinematic constraint equations. Examples of this contact type are:

*CONTACT_TIED_NODES_TO_SURFACE (6)
*CONTACT_TIED_SURFACE_TO_SURFACE (2)

These contact types should generally only be used with solid elements since rotational degrees-of-freedom of the slave node are not constrained. The use of this contact type for shell elements may produce unrealistically soft behavior. Contact types 2 and 6 differ only in the input format (slave segments vs. slave nodes); the numerical treatment is the same.

In general, when using tied interfaces between similar materials, the master surface should be the more coarsely meshed side since these constraints are not applied symmetrically. However, if one material is significantly softer, the master side should be the stiffest material.

Constraint-based tied contacts such as types 2 and 6 cannot be used to tie a rigid body to a deformable body or to another rigid body. Nodes of deformable bodies that the user wishes to be tied to a rigid body can be included as extra nodes for the rigid body using the *constrained_extra_nodes command. Alternately, the OFFSET option can be used for tied contacts involving rigid bodies (see below).

3.4 Tied Contact (Translational DOF only, No Failure, With Offset)

This contact types works the same as above but an offset distance between the master segment and the slave node is permitted. Offset tied contacts use a penalty-based formulation and thus can be used to tie rigid bodies. Examples of this contact type are:

*CONTACT_TIED_NODES_TO_SURFACE_OFFSET (o 6)
*CONTACT_TIED_SURFACE_TO_SURFACE_OFFSET (o 2)

This contact type works best if the surfaces are very close, since moments that develop due to the offset are not taken into account. Not accounting for the moment transmission due to offsets can impose rotational constraints on the structure. With the penalty approach this is not too much of a problem, however, with the constraint method, the results can be completely wrong.

To account for the moment transmission between the offset surfaces, two methods are available. The first, based on a penalty formulation, uses beam-like spring elements to transmit the moments:

*CONTACT_TIED_NODES_TO_SURFACE_BEAM_OFFSET (o 6)
*CONTACT_TIED_SURFACE_TO_SURFACE_BEAM_OFFSET (o 2)

and the second uses constraint equations:

*CONTACT_TIED_NODES_TO_SURFACE_CONSTRAINT_OFFSET (c 6)
*CONTACT_TIED_SURFACE_TO_SURFACE_CONSTRAINT_OFFSET (c 2)

The offsets can be reasonably large with the BEAM and CONSTRAINT options. However, since rotational degrees-of-freedom are not affected, the offset contacts should not be used with structural elements like beams and shells. The offset contacts that transmit moments were added to the first release of version 960 after the manual was published.

3.5 Tied Contact (Translational DOF and Rotational DOF, With Failure, No Offset)

This contact interface uses a kinematic type constraint method to tie the slave nodes to the master segments and treats both translational and the rotational degrees-of-freedom. Additionally, failure can be specified when combined with beam elements of material type, *MAT_SPOTWELD, when modeling spot welds. Examples of this contact type are:

*CONTACT_TIED_SHELL_EDGE_TO_SURFACE (7)
*CONTACT_SPOTWELD (7)
*CONTACT_SPOTWELD_WITH_TORSION (s 7)

With the above types the nodes are projected to lie on the master segment. This is quite important for *CONTACT_SPOTWELD, since the beams that model the spot welds need to be as long as possible to minimize the mass scaling that is necessary to allow the calculation to have a reasonable time step size. With the TORSION option, the torsional forces in the beam, which models the spot weld, are transmitted as equivalent forces to the surrounding nodes of the master surface. The rotational constraint about the axis of the beam is then enforced. The nonlinear shell elements in LS-DYNA have a zero stiffness drilling degree-of-freedom at each node, so it is necessary to carry the torsional forces through the membrane behavior of the shell.

3.6 Tied Contact (Translational DOF and Rotational DOF, With Failure, With Offset)

These contact interface options uses either a kinematic or penalty type constraint method to tie offset slave nodes to the master segments:

*CONTACT_TIED_SHELL_EDGE_TO_SURFACE_OFFSET
*CONTACT_TIED_SHELL_EDGE_TO_SURFACE_BEAM_OFFSET
*CONTACT_TIED_SHELL_EDGE_TO_SURFACE_CONSTRAINT_OFFSET

With the BEAM and CONSTRAINT option, the moments that develop from the offsets are computed and used in the update of the master surface.   The nodes involved should belong to deformable elements. The CONSTRAINT option cannot be used with rigid bodies. The difficulty with using even the penalty option with rigid bodies is related to the nodal masses of the rigid body. If the nodal masses are accurate then the penalty method is okay. If the masses are nonsense, as is often the case if the rigid body geometry is accurate but the inertial properties are defined independently of the mesh, then the penalty method may break down since the nodal masses of the rigid body are used to set the penalties that are used in the rotational constraints.

3.7 Tied Contact (Translational DOF, With Failure, With Offset)

The following penalty based contact types allow for the definition of failure parameters. It is extremely important to have the contact segment orientation aligned appropriately as it determines the tensile and compression direction. Failure can be based on the forces or stress along the normal (tensile) and shear directions. Examples of this contact type are:

*CONTACT_TIEBREAK_NODES_TO_SURFACE (8)
*CONTACT_TIEBREAK_NODES_ONLY
*CONTACT_TIEBREAK_SURFACE_TO_SURFACE (9)

When offsets are used, there is no option to distribute the moments created by the offsets to the master surface. Originally, this contact was developed to work without offsets. Effort is under way to provide alternatives to these options in the next release of LS-DYNA.

3.7 Single Surface

These contact types are the most widely used contact options in LS-DYNA, especially for crashworthiness applications. With these types, the slave surface is typically defined as a list of part ID's. No master surface is defined. Contact is considered between all the parts in the slave list, including self-contact of each part. If the model is accurately defined, these contact types are very reliable and accurate. However, if there is a lot of interpenetrations in the initial configuration, energy balances may show either a growth or decay of energy as the calculation proceeds.

For crash analysis, the contact type…

*CONTACT_AUTOMATIC_SINGLE_SURFACE (13)

is recommended. This contact has improved from version to version of LS-DYNA and is the most popular contact option.

The older single surface contact type…

*CONTACT_SINGLE_SURFACE (4)

should be avoided since it has not undergone improvement. It eventually will be removed or recoded. The differences between CONTACT_SINGLE_SURFACE and CONTACT_AUTOMATIC_ SINGLE_SURFACE are twofold. First, the older method uses nodal based bucket sorting where closest nodes are found that do not share common segments. This nodal based searching can break down if the segments vary appreciably in size and shape, especially, if aspect ratios are large. Secondly, the older method uses segment projection to determine the contact surface. This requires the calculation of nodal normal vectors that are area weighted by the segments that share the node, which in turns creates further difficulties for T-intersections and other geometric complications. The calculation of the vectors can require 25% of the total CPU required.

For modeling the deployment of airbags the following contact option is recommended:

*CONTACT_AIRBAG_SINGLE_SURFACE (a13)

With AIRBAG_SINGLE_SURFACE, contact between nodes and multiple segments is considered. Much more searching is done than in the normal contact option and, consequently, this contact option is much more expensive. During the past several years, the soft constraint option, on optional card A, in the contact definition, set to 2 has proved to deploy airbags very accurately. We current recommend this option for airbag deployment. The latter option is currently being implemented for MPP usage.

The final contact is:

*CONTACT_AUTOMATIC_GENERAL (26)

The contact treatment with this option was similar to type 13 through the 950c release of LS-DYNA. The main difference was that three possible contact segments, rather than just two, were stored for each slave node. With 950d and later versions, type 13 was substantially improved and now type 13 is frequently more accurate. The main feature of the GENERAL option is that shell edge-to-edge and beam-to-beam contact is treated automatically. All free edges of the shells and all beam elements are checked for contact with other free edges and beams. Unlike type 13 contact, type 26 contact checks for contact along the entire length of beams and exterior shell edges, not just at the nodes. There is a new option in 960 to also check internal shell edges (INTERIOR option). This is quite expensive, however, and is not usually needed. We plan to update this contact type in version 970 of LS-DYNA to include all the recent improvement in the AUTOMATIC_SINGLE_SURFACE contact.

3.8 Contact Entity

This contact type is used for treating deformable nodes against “rigid” geometric surfaces. The analytical equations defining the geometry of the surface are used in the contact calculations. This is an improvement over the usual segmented surface as represented by a mesh. A penalty-based approach is used in calculating the forces that resist penetration. This contact type is widely used to couple LS-DYNA with rigid body dummies, which have surfaces approximated by nice geometric shapes such as ellipsoids. An automatic mesh generator is used to mesh the rigid surfaces to aid visualizing the results. The mesh is not used in the contact calculations. The analytical rigid surfaces can be of the following types

Flat
Sphere
Cylinder
Hyper-ellipsoid
Torus
Load curve defining the line
CAL3D/MADYMO plane
CAL3D/MADYMO ellipsoid
VDA surface (read from a file)
IGES surface (read from a file)

4.0 Contact Stiffness Calculation

Contact treatment is internally represented by linear springs between the slave nodes and the nearest master segments. The stiffness of these springs determines the force that will be applied to the slave nodes and the master nodes. There are currently two methods of calculating the contact spring stiffness and they are briefly discussed below.

4.1 Penalty-based approach (SOFT = 0 in Optional Card ‘A’ in *CONTACT_)

This method is the default method and uses the size of the contact segment and its material properties to determine the contact spring stiffness. As this method depends on the material constants and the size of the segments, it works effectively when the material stiffness parameters between the contacting surfaces are of the same order-of-magnitude. In cases where dissimilar materials come into contact, the contact might break down, as the stiffness, which is roughly the minimum of the slave and master stiffness, may be too small. This frequently happens with soft dense foams contact metal materials. Consequently, for crash analysis we do not recommend the option, SOFT=0, unless prior experience shows that no problems occur.

4.2 Soft Constraint-based approach (SOFT = 1& 2 on Optional Card ‘A’ in *CONTACT_)

This non-default method calculates the stiffness of the linear contact springs based on the nodal masses that come into contact and the global time step size. The resulting contact stiffness is independent of the material constants and is well suited for treating contact between bodies of dissimilar materials. The stiffness is found by taking the nodal mass divided by the square of the time step size with a scale factor to ensure stability. Generally, for the case of metals contacting metals the resulting penalty stiffness for SOFT=0 or SOFT=1 is similar. For the case where soft dense foams contact metal, the option, SOFT=1 often gives interface stiffness that are one or two orders-of-magnitude greater. The SOFT=1 option is recommended for impact analysis where dissimilar materials come into contact.

The SOFT=2 option uses mass and time step based penalty stiffness as in SOFT=1. SOFT=2 invokes a segment-based contact algorithm which has it origins in Pinball contact developed by Belytschko and his co-workers. With this contact algorithm, contact between segments is treated rather than using the usual node-to-segment treatment. When two 4-noded segments come into contact, forces are applied to eight nodes to resist segment penetration. This treatment has the effect of distributing forces more realistically and sometimes is quite effective for very stubborn contact problems. The SOFT=2 option is currently being ported for MPP calculations. Beam contact is not handled by SOFT=2 type contact. Further, SOFT=2 is available only for surface-to-surface and single surface contacts and not for nodes-to-surface contacts. The optional parameter EDGE on Optional Card A should be used cautiously when segment-edge-to-segment-edge contact is anticipated and SOFT is set to 2.

5.0 Contact Output

There are numerous output files pertaining to contact which can be written by LS-DYNA. LS-POST can read these output files and plot the results.

The most common contact-related output file, RCFORC, is produced by including a *DATABASE_RCFORC command in the input deck. RCFORC is an ASCII file containing resultant contact forces for the slave and master sides of each contact interface. The forces are written in the global coordinate system. Note that RCFORC data is not written for single surface contacts as all the contact forces from such a contact come from the slave side (there is no master side) and thus the net contact forces are zero. To obtain RCFORC data when single surface contacts are used, one or more ‘force transducers’ should be added via the *CONTACT_FORCE_TRANSDUCER_PENALTY command. A force transducer does not produce any contact forces and thus does not affect the results of the simulation. A force transducer simply measures contact forces produced by other contact interfaces defined in the model. One would typically assign a subset of the parts defined in a single surface contact to the slave side of a force transducer. No master side is defined. The RCFORC file would then report the resultant contact forces on that subset of parts.

The ASCII output file NCFORC reports contact forces at each node. The command *DATABASE_NCFORC is required in the input deck to produce such a file. Further, one or more contact print flags must be set (see SPR and MPR on Card 1 of *CONTACT). Only those surfaces whose print flag is set to a value of 1 will have their nodal contact force output to the NCFORC file.

By including a *DATABASE_SLEOUT command, contact interface energies are written to the ASCII ouput file SLEOUT. In cases where there are two or more contact interfaces in a model and the global statistics file (GLSTAT) indicates a problem with contact energy, such as a large negative value, the SLEOUT file is useful for isolating which contact interfaces are responsible. For general information on interpreting contact energies, see the LS-DYNA Theory Manual, Section 23.8.4.

In some cases, it can be very useful to visualize contact surfaces and produce fringe plots of contact stress both in directions normal and tangential to the contact surface. To do this, a binary interface file must be written by (1) including a *DATABASE_BINARY_INTFOR command in the input deck, (2) setting one or more contact print flags as detailed above, and (3) including the option “s=filename” on the LS-DYNA execution line where filename is the intended name of the binary database. The database can be postprocessed using LS-POST.

FEA Information Web Sites Summary

Each Monday the news page changes to showcase:
Software – Product - Participant's Headquarters or a software distributor

If you missed any of the following the information is archived on the News Page on FEA Information for two months:

July 2nd:

- **Software**: Oasys Ltd. – United Kindgom – **Oasys D3Plot** is a 3D visualization package for post-processing visually the results of LS-DYNA Analysis.
- **Participant**: **ANSYS, Inc**., headquartered in the USA. .

July 9th:

- **Participant**: **LMS International**, headquartered in Belgium.
- **Software**: LMS International – Belgium – **LMS Pimento** combines the power of a professional lab system in a portable package. Ideal for the user who wants to pull an analyzer off the shelf, switch it on, start measuring, and produce a report.
- **Product**: Hewlett Packard – USA – **HP Workstations with Linux** from their workstation x2000 with Linux, moving up to the pL class and through the xL-class to the HP workstation x4000 with Linux.
- **Distributor**: We welcome a new software distributor to FEA Information Inc. websites: **DYNALIS,** headquartered in France

July 16th:

- **Product**: Fujitsu Ltd. – **LIFEBOOK X** series high-end notebook ideal for professional users who need industry-grade performance at home, in the office or on the road.
- **Distributor**: **Korean Simulations Technologies** – Korea

July 23rd:

- **FEA Information Announcement**: We have become members in the Society of Manufacturing Engineers, and the American Society of Mechanical Engineers.
- **Product**: **DesignSpace V. 6** capabilities include parametric simulation, fatigue, surface model support, nonlinear plus enhancements from Version 5
- **Distributor**: **Leading Engineering Analysis** (LEAP) – Australia
- **Software**: LSTC – USA – **LS-OPT** allows the user to structure the design process, explore the design space and compute optimal designs according to specified constraints and objectives.

July 30th

- **Distributor**: **Gisetta India** – headquartered in Chennai, India is India's premier consulting company in addition to local sales and software support.
- **Software**: The Japan Research Institute – **JMAG Studio** is a magnetic field analysis program that supports the development of electrical and electronic devices.
- **Operating System**: MSC Software – **MSC Linux** focuses on extreme performance computing in the Linux environment.

Courses and Events will be limited to 1 page
For further information visit the events page on FEA Information

Events

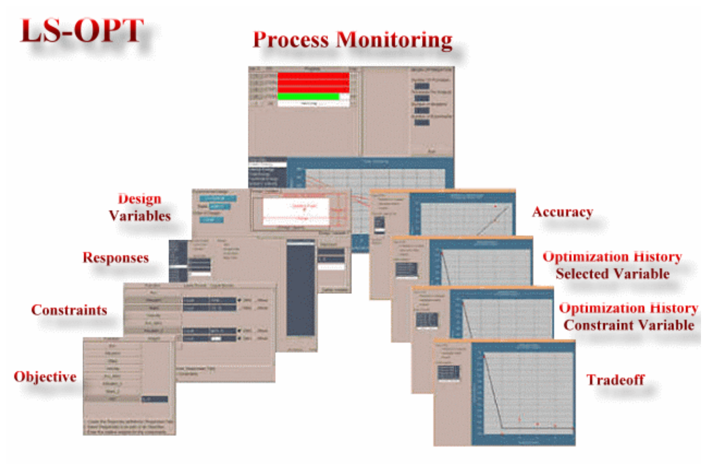
France	Sept 24-26	**Worldwide Aerospace Conference & Technology Showcase**, Toulouse Congress Center, Toulouse, France
USA	Sept 25-26	**LMS 2001 Conference for Physical and Virtual Prototyping**, Michigan State Management Education Center, Troy, MI.
Germany	Oct 17-19	**19th CAD-FEM Users' Meeting** - International Congress on FEM Technology will be held October 17-19, 2001 at the DORINT SANS SOUCI Hotel in Potsdam, near Berlin.
Japan	Oct. 30-31	**LS-DYNA Users Conference**, sponsored by Japanese Research Institute (JRI) - to be held at the Sheraton Grande Tokyo Bay Hotel.
France	Nov 13-14	**LMS 2001 Conference for Physical and Virtual Prototyping**, Hotel New York, Disnyeland Paris, Paris France
USA	May 19-21 2002	**7th International LS-DYNA User's Conference,** Hyatt Regency Hotel & Conference Center, Dearborn, MI

September Classes – Seminars

Sept	Country	Information	Class Title
4	US	LMS	(Troy, MI) LMS Time Waveform Replication Product Training
5	US	LMS	(Troy, MI) LMS CADA-X System Admin. Product Training
5-7	US	ANSYS Hdq.	High Frequency Electromagnetics
10-11	US	LSTC	Geomaterial Modeling with LS-DYNA
10-14	US	MSC Software	(Costa Mesa, CA) Introduction to MSC Patran Command Language
11	US	LMS	(Troy, MI) LMS Sysnoise Product Training
12-14	US	ANSYS Hdq.	Electromagnetic Analysis
12-14	KR	THEME	LS-DYNA Introductory Training
12-13	US	LSTC	LS-DYNA Implicit Training
17-18	US	ANSYS Hdq.	Dynamics
18-21	US	MSC Software	(Houston, TX) MSC Patran for Advanced Users
19	US	LMS	(Los Angeles, CA) Introduction to LMS DADS
20-21	US	ANSYS Hdq.	Explicit Dynamics with ANSYS/LS-DYNA
20-21	KR	THEME	eta/FEMB Introductory Training
24-25	US	ANSYS Hdq.	Introduction to ANSYS for MEMS
25	UK	Oaysis	Oasys Primer Version 8.1
25-28	US	MSC Software	(Bellevue, WA) MSC Patran for Advanced Users
26	UK	Oaysis	Oasys D3Plot & T/HIS Version 8.1
26-28	UK	Oaysis	Introductory Course
26-28	US	ANSYS Hdq.	Multiphysics Simulation for MEMS

FEA Information Showcase

Livermore Software Technology Corporation
[www.ls-dyna.com]
Optimization



Available Space To Showcase
In the meantime
Visit FEA Information Inc. Sites

Crash-Analysis
[crash-analysis]
Implicit FEA
[implicitfea.com]
Linux for PC
[linuxforpc.com]
Heat Transfer Analysis
[www.heattransferanalysis.com]
Warhead Analysis
[www.warheadanalysis.com]
Metal Forming Simulation
[www.metalformingsimulation.com]

MSC Software
[www.msclinux.com – software products]
Post-Processor

FEA Information Inc.
[www.feainformation.com]
On Line Store
Version 960 LS-DYNA Available

LS-DYNA Version 960 Keyword User's Manual Volume I
LS-DYNA Version 960 Keyword User's Manual Volume II
5th International LS-DYNA Users Conference 1998
6th International LS-DYNA Users Conference 2000
LS-DYNA Theoretical Manual
LS-DYNA Examples Manual
LS-OPT User's Manual Version 1

Product names referred herein are trademarks
of their respective companies