Advanced MPP Decomposition of a SPH Model

Vincent LAPOUJADE[1], Christian MOUTELIERE[1], Dr. Antoine MILLECAMPS[2]
[1]*DynaS+, 5 avenue Didier DAURAT, F-31400, TOULOUSE*
[2]*SNECMA, Rond Point René RAVAUD – Réau, F-77550 MOISSY CRAMAYEL*

Abstract

SPH, Smoothed Particle Hydrodynamics, is a very efficient tool to model industrial problems where large deformations occur. However, one disadvantage of the SPH technique is the relative expensive cpu cost compared to standard Finite Elements. Using the MPP version of LS-DYNA® allows users to handle larger problems (up to more than millions of particles) in a reasonable time.

Due to the mesh free nature of the SPH method, standard decompositions used for finite elements can sometimes lead to very bad speed-up of the code. Users have to be aware of some options and rules to define customized decompositions in order to minimize communications between processors and get very good load balancing.

Two classes of models are presented for addressing all possible situations with respect to optimizing MPP decomposition of a calculation based in whole or in part on the SPH technology. The first one is a pure SPH model of a high velocity impact of a sphere on a plate. The second one is a coupled FE-SPH model of a bird impacting a set of fan blades of an engine. Two versions of the same problem will be studied: for the first, shell elements are used for the modeling of fan blades whereas for the second, solid elements are used.

Keywords

SPH, MPP decomposition

Introduction

With the dramatic increase of the performance to cost ratio of modern high performance computers, the sizes of numerical models have been constantly increasing in recent years. It is now common to deal with several hundred thousand element models and the threshold of several million elements is more and more often reached.

Consequently the need for parallel computation and a scalable solution becomes more and more important. However, it is known that the loop level parallelism of SMP binaries shows significantly limited scaling behavior for any number of processor above 8. On the other hand, the massively parallel processing (MPP) binary of LS-DYNA® takes advantage of better scaling properties through domain decomposition technique and message passing interface (MPI) programming. There has been, therefore, an increasing deployment of LS-DYNA MPP as a production version at many LS-DYNA user sites in past decade.

Since LS-DYNA MPP is a domain decomposition based software, data partitioning algorithms play a key role in the efficiency and scalability of the code. We will show in this paper that for pure SPH and coupled FE-SPH models, special data partitioning techniques may significantly improve the efficiency.

This paper can be considered as an update of the paper "MPP Decomposition of a SPH Model - 6th European LS-DYNA Users' Conference, Gothenburg - SWEDEN, 2007" to take into account both changes from an hardware point of view but also in the nature of the models used. The original paper has lead to conclusions for models with a mix of classic FE and SPH particles in a framework where the number of particles was very significantly higher than that of FE. The evolution of practices, especially in SNECMA, led to keep a relatively similar number of SPH particles to model the bird but to increase greatly the number of finite elements. The last test case presented in this paper will therefore allow to check whether the initial conclusions are still applicable in this new context.

MPP Parallelism in LS-DYNA – SPH Specificities

General Points

Unlike the SMP version of LS-DYNA, which exploits parallelism in loop level, the MPP LS-DYNA uses the domain decomposition technique to partition the model into sub-domains. With the domain decomposition version of LS-DYNA, the geometry is divided into several domains, one per processor.

In a standard finite element explicit analysis performed with LS-DYNA MPP, during each time step, each processor advances the solution for its own domain to the end of that time step. This process is mainly independent of all others domains, so it is highly parallel. Each processor only needs to communicate with other processors at two points during each time step : when contact forces are calculated and when the nodal displacements are being updated. Once this communication is complete, the solutions phase of the next time step begins.

The communication in LS-DYNA takes place according to MPI (the Message Passing Interface), a standard communication protocol. Depending on the processor type and the Os different MPI implementations can be used (such as OpenMPI, MPICH, Platform-MPI, Intel-MPI or MS-MPI…).

It is clear that a well selected domain decomposition could influence MPP LS-DYNA performance. Local balance between the domains is achieved by insuring that each domain has an equal amount of work required at each time step. Variation of the computational cost between different types of elements and material models is clearly a key factor, which complicates the load balance. In addition, time spent in the treatment of contact, which will change during simulation, also has a great influence on computational cost.

There are several options for partitioning the LS-DYNA model. These are documented in appendix O “LS-DYNA MPP User Guide” of the LS-DYNA Users’ manual. The standard method for partitioning is RCB (Recursive coordinate bisection). Basically, the RCB algorithm will split the model into 2 equal domains. The split plane cuts the largest domain dimension (x, y or z) at a location that roughly balances the number of elements in each half. Half of the processors are then allocated to each domain. The remaining portions will be then be split along their largest dimensions and a portion allocated to each processor until each domain has roughly the same number of elements and is assigned to one processor.

Because the domain decomposition can have a huge impact on how quickly the model is solved, the user has some control over how this calculation proceeds. The most common options are used to scale up or down, for the purpose of partitioning only, the model in one or several directions. The partitioning method and options relating to partitioning can be specified in the partitioning file “pfile”, an optional file used by the LS-DYNA MPP binary or through appropriate keywords.

MPP SPH COMPUTATIONS

The major difference between standard finite elements used in LS-DYNA and SPH, from an MPP point of view, is the amount of communication needed between processors. In the part of the explicit loop which is specific to SPH, there are three new points where communication between processors is needed. The first point is during the SPH sorting to determine the neighbors that are in the sphere of influence of each single particle. The second point is the computation of the strain rate. The third one is the computation of the internal forces.

For lagrangian finite elements, the MPP efficiency is governed by the number of nodes located at the boundaries between decomposition sub-domains. For SPH, all particles having at least one neighbor in another sub-domain will affect the efficiency. This means all particles located less than twice the smoothing length from their sub-domain boundary will create a need for communication between processors. The communication between processors for SPH MPP will be therefore not only more frequent but also more important in terms of the volume of data to be exchanged.

The second point is the relative cost of SPH elements compared to other LS-DYNA elements. For finite elements, the relative cost between different element types is fairly fixed. The cost of an SPH element can vary during the simulation mainly due to an increasing number of particles being taken into consideration in each sphere of influence. This will affect both the sorting efficiency and the evaluation of the strain rate and forces cpu cost. It clearly opens the way to new types of bottleneck for MPP binary efficiency.

Last, due to the extra amount and frequency of communication for MPP SPH, user should take care to use SPH MPP with a sufficient number of particles assigned for each processor (it is difficult to fix a clear limit but several ten thousand is, in the authors opinion, a good starting point). Otherwise, communication cost will be so important that MPP will be of no interest compared to SMP.

CASE #1 : HIGH VELOCITY IMPACT ON A SINGLE PLATE

MODEL PRESENTATION

The model used for case #1 is a standard aluminum sphere impacting an aluminum plate.

The sphere has a 5 mm diameter. It is associated with a classical Johnson-Cook aluminum model and the corresponding Gruneisen EOS. The impact occurs at 5 km/s. The inter-particular distance is 0.2 mm. The total number of particles in the impacting sphere is therefore 8217.

The plate is 60 x 60 mm with a thickness of 2 mm. The square plate is associated with the previous classical Johnson-Cook aluminum model and the previous Gruneisen EOS. The inter-particular distance is 0.2 mm (the plate therefore has 10 particles through the thickness). The number of particles in the square plate is therefore 906010.

The total number of SPH elements for case #1 is therefore 914 227.

The termination time is 10 µs.

The hardware system used to run this model (and also for case #2 and case #3) was a Dual eight cores Intel ® Xeon ® CPU E5-2687W V1 @ 3.10GHz with 128 Gb Memory under Linux64 CentOS 6.4.For the paper, the LS-DYNA MPP double precision binary version R7.0 with PlatformMPI was used on the 16 cores.

DECOMPOSITION

Due to the fact that 2 dimensions in the model are almost an order of magnitude larger than the third one, the default decomposition leads to a “patchwork” like decomposition (see Table 1 below).

Six other decompositions were tested. They are also presented in Table 1 below. As Case #1 is a pure SPH model the default decomposition is equivalent to the one obtain with “Sphdist” option. SphDist is a specific decomposition option to be used in “pfile” developed for SPH (also activated using *CONTROL_MPP_DECOMPOSITION_DISTRIBUTE_SPH_ELEMENTS). This option leads to distribute all SPH particles to all processors (cores). SPH elements usually have higher computational cost than other type of elements and it is supposed to be more efficient to distribute them to all CPU for better load balance.

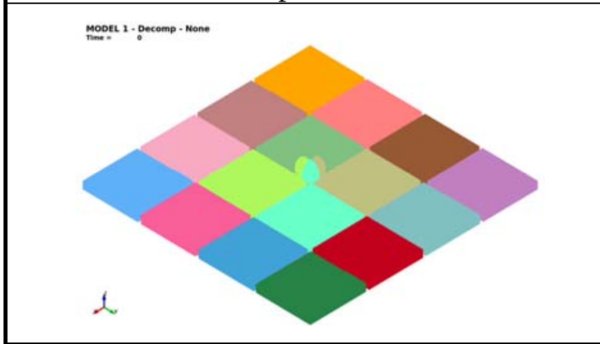
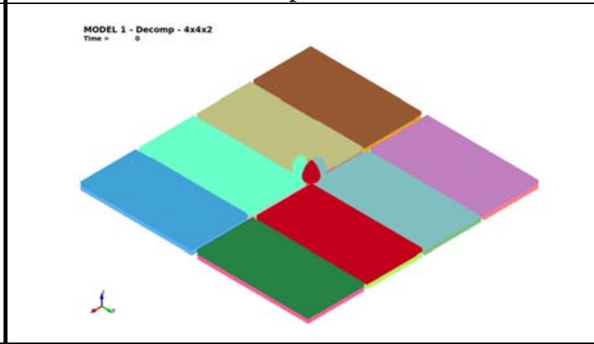
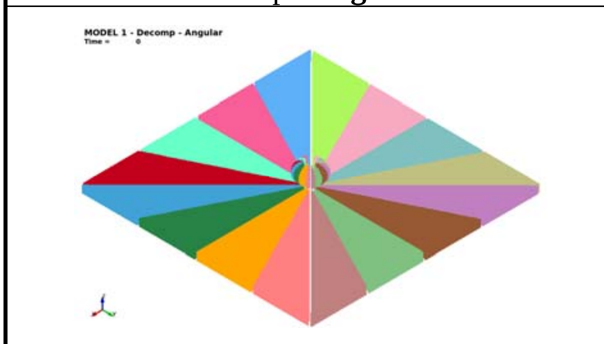
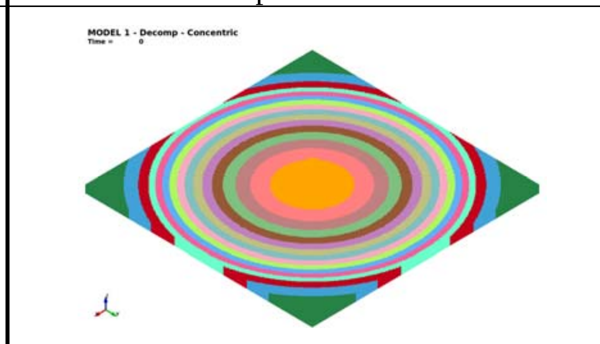
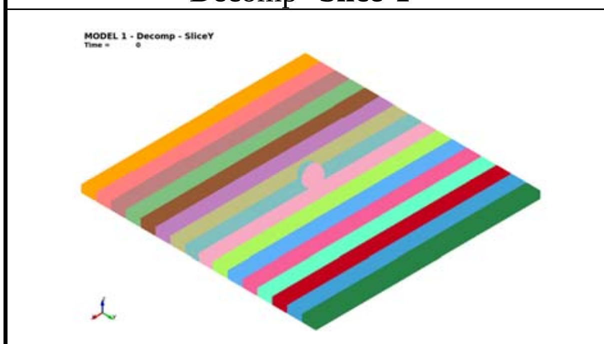
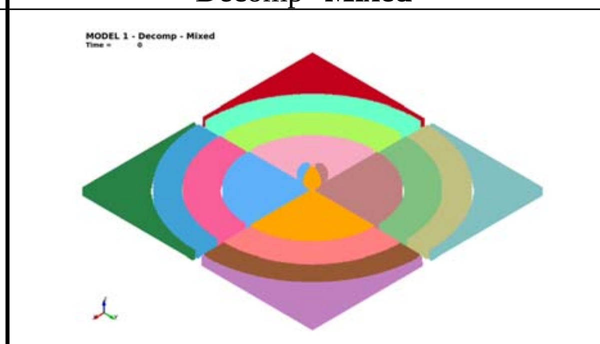
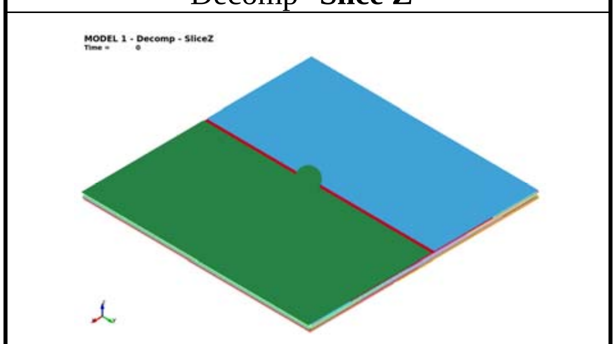
Decomp "**Default**"	Decomp "**4x2x2**"
	
none	*"decomp {sz 8.5 sx 2 }"*
Decomp "**Angular**"	Decomp "**Concentric**"
	
"decomp {c2r 0 0 0 0 0 1 1 0 0 sy 5000} "	*"decomp {c2r 0 0 0 0 0 1 0 1 0 sx 5000}"*
Decomp "**Slice Y**"	Decomp "**Mixed**"
	
"decomp {sy 5000}"	*"decomp {c2r 0 0 0 0 0 1 1 0 0}"*
Decomp "**Slice Z**"	
	
"decomp { sz 1000}"	

Table 1: Decomposition & pfile for Case #1

DISCUSSION OF RESULTS

Model at start of simulation Model at end of simulation time (10 µs)

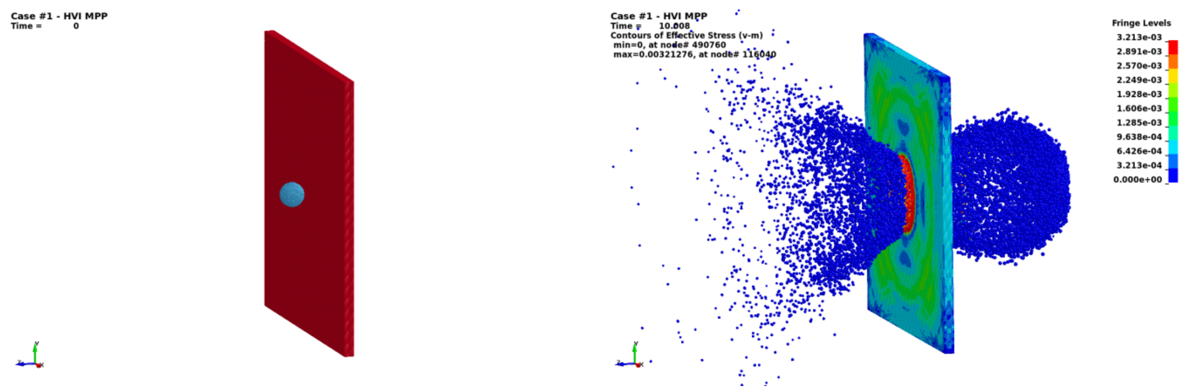


Figure 1: Case #1 – Results of simulation

The default decomposition model is set as the reference. The elapsed time for this decomposition model is 1380 seconds. The relative efficiency is shown in Figure 2.

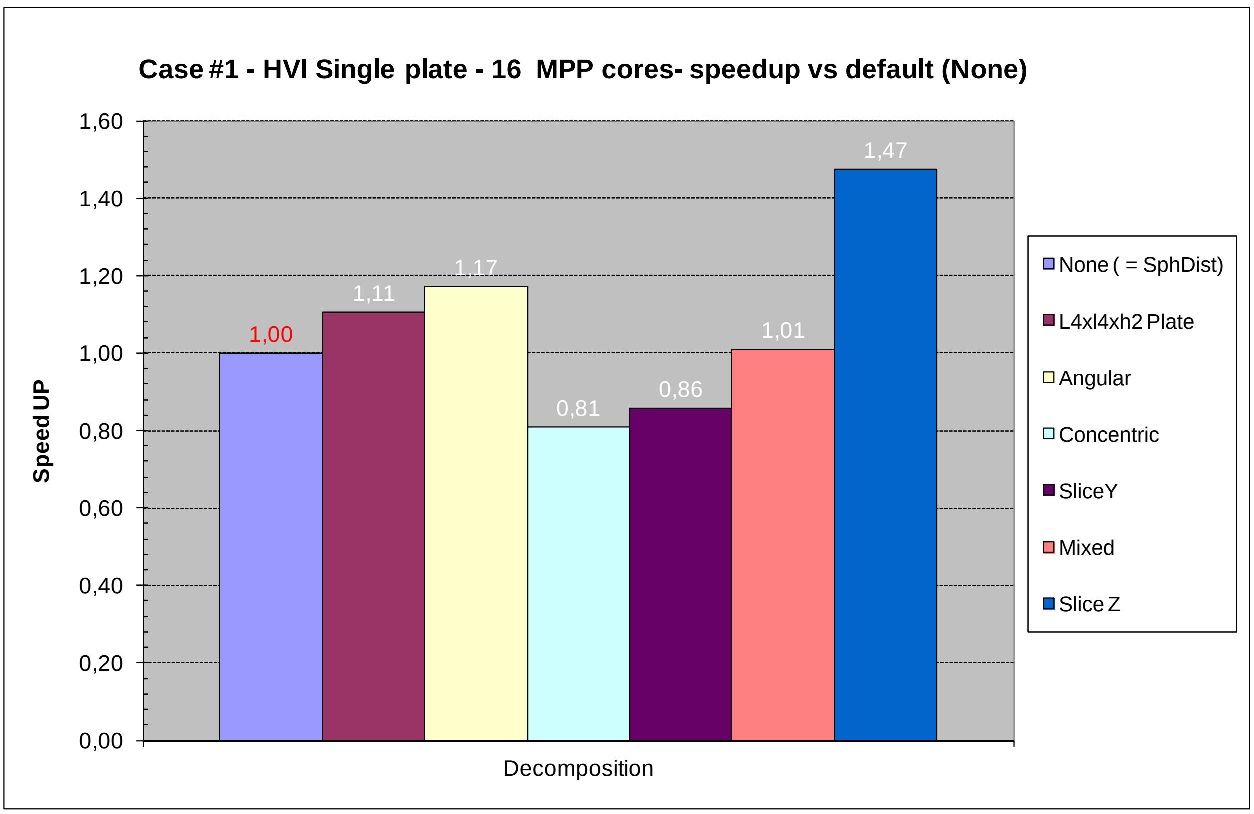


Figure 2: Case #1 – Speedup vs. default

This first case is a pure SPH model. There is therefore no contact, which is well known in MPP as a source of load balance problem. The decomposition efficiency will be therefore directly linked with the SPH computation.

The "Slice Z" decomposition logically leads to a 1.47 ratio. This of course is directly linked with the fact that for each single particle in the plate, the sphere of influence of each particle is split over several domains in the Z direction. The "Angular" decomposition is the second worst with a ratio of 1.17. For this decomposition, the great majority of spheres of influence especially in the initial impact area are split in multiple sub domains due, compare to 2007 paper, to the fact that

16 cores are used leading to smaller domains with more boundaries and consequently more communications needed. The “4x2x2” decomposition is the third worst with a ratio of 1.11. For this decomposition, the great majority of spheres of influence are split at least in 2 sub domains due to the Z slices.

For all other decompositions, we have a great majority of particles whose spheres of influence are completely enclosed within a single sub-domain. The communication costs are therefore dramatically reduced compare to the third previous decompositions. Analyzing the figures more precisely, the “mixed” decomposition leads to a ratio of 1.01. The “mixed” decomposition has a topology closer to the default. A large number of spheres of influence are entirely enclosed in a single domain. For the domain boundaries, spheres of influence are mainly split over only 2 sub domains just as in the default decomposition. The major difference is the number of junctions including four domains: 13compared to 9 in the default decomposition. The result is a slightly higher ratio for the “mixed” decomposition.

Last, there are two decompositions whose ratios are less than one “SliceY” & “Concentric”. For both, spheres of influence are never split across more than 2 sub domains. And the best, with a ratio of 0.81, is the “Concentric” decomposition, for which the sphere impactor is not spread across two sub-domains.

PARTIAL SYNTHESIS

The conclusions that can be drawn from previous data and the best decomposition leading to the lowest computation time differ compared to the 2007 study on the same model.

This is mainly due to the fact that the number of processing cores has doubled leading to a number of particles twice lower for each core. The total number of particles being the same, this results in requiring more particles have a need for inter core communication.

In the case of decomposition "SliceY", which was the optimal decomposition for 8 cores, the number of frontiers between two domains was generally multiplied by 2 for 16 cores and for each core the number of particle inducing a communication need with another domain is greatly increased (22 to 43%). This very significant extra cost explains the relative loss of efficiency for the corresponding decomposition.

In contrast decomposition "concentric" which is the optimum of 16 hearts, has the great advantage of keeping the impactor on a single core significantly reducing the need for communication in this area of the model.

To summarize in the conclusions from 2007, the part relating to the fact that the most effective decompositions are those for which the frontiers between more than two domains are as limited as possible remains a priori correct. However with the increase in the number of cores and decrease in particles number per core, the fact to minimize the area of frontiers is not necessarily a guarantee of optimum efficiency.

CASE 2: BIRD IMPACT ON FAN BLADES (Shell)

MODEL PRESENTATION

The model used for case #2 is a 2.5 lb bird impacting titanium fan blades modeled with shell elements. This model has been provided by SNECMA. It is representative of a real input deck.

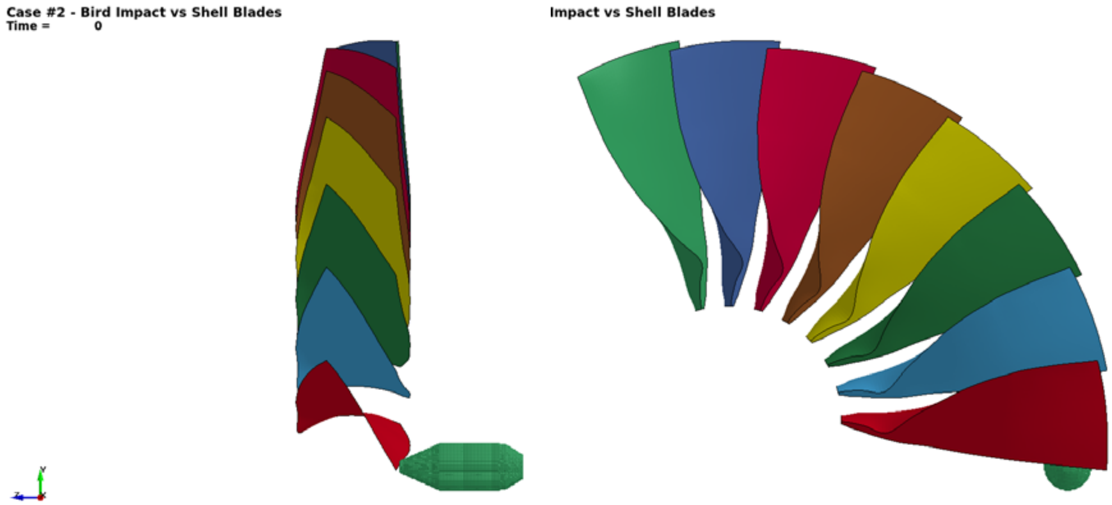


Figure 5: Case #2 – Mesh Views

The bird has a global mass of 2.5 lb. It is associated with a null material law model and a tabulated EOS. The impact occurs at 94 m/s. The inter-particular distance is 2 mm. The number of SPH elements in the bird is therefore 157 596.

Eight titanium fan blades are included in the model. A Johnson-Cook model is used for titanium. Each fan blade is meshed with 9240 type 16 fully integrated shell elements. The total number of shell elements is therefore 73920. The engine is rotating at cruise speed. A contact of the "Node_to_Surface" type is defined between the bird and the blades.

DECOMPOSITION

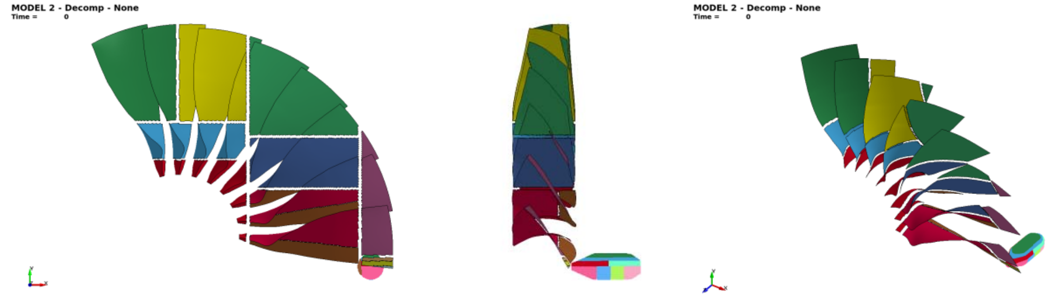
The RCB default decomposition leads to mixed domains of pure SPH, pure finite elements and mixed element types.

A first attempt to optimize the decomposition is the "SphDist" decomposition, where the Bird is decomposed over the 16 cpu.

Based on results obtain in the 2007 paper, a third model "BirdZ" was studied. For this model the bird is decomposed on the 16 CPU but in slice along Z direction. For the four following decompositions ("BirdZ + Concentric", "BirdZ + Angular", "BirdZ + Angular2" & "BirdZ + Blade Z"), the bird is still decomposed over the 16 cpu in a slice Z type decomposition and effort is also applied to the blades decomposition.

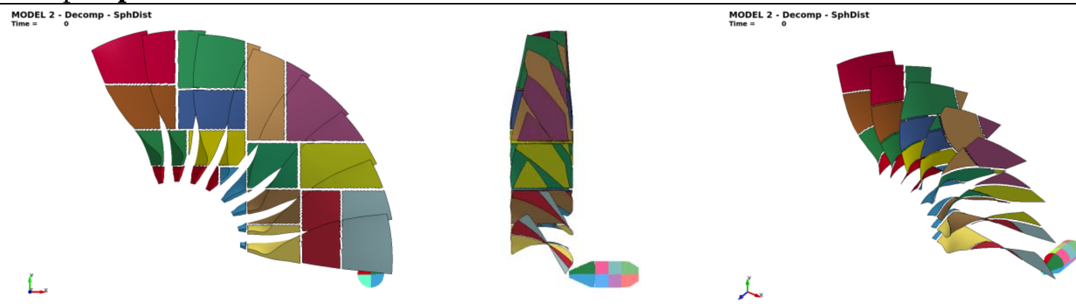
After analyzing the results, three new decompositions were tested ("SphDist + Concentric", "SphDist + Angular", "SphDist + Angular2". For these three decompositions, bird is simply distributed over all cores (used of SphDist for SPH) and a special effort is made on the decomposition of finite elements used for the blades.

Decomp "**Default**"



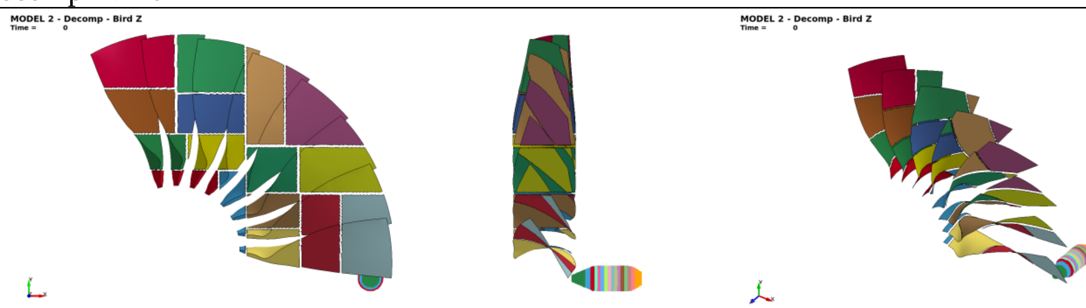
none

Decomp "**SphDist**"



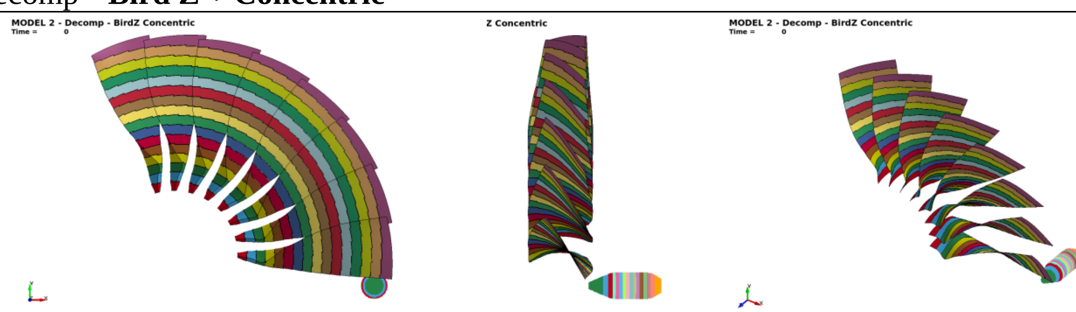
decomp { sphdist }

Decomp "**Bird Z**"



decomp { region {parts 100000 sz 5000 } }

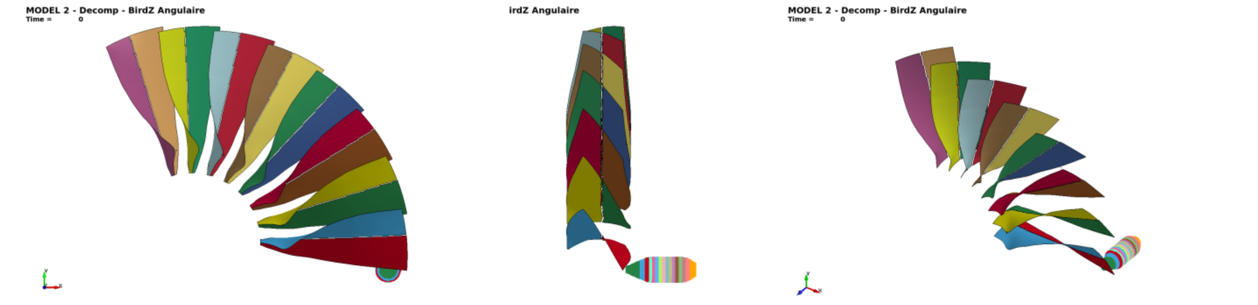
Decomp " **Bird Z + Concentric** "



decomp { region {parts 100000 sz 5000 } }
decomp { region {parts 1 2 3 4 5 6 7 8 c2r 0 0 0 0 0 1 1 -1 0 sx 5000 } }

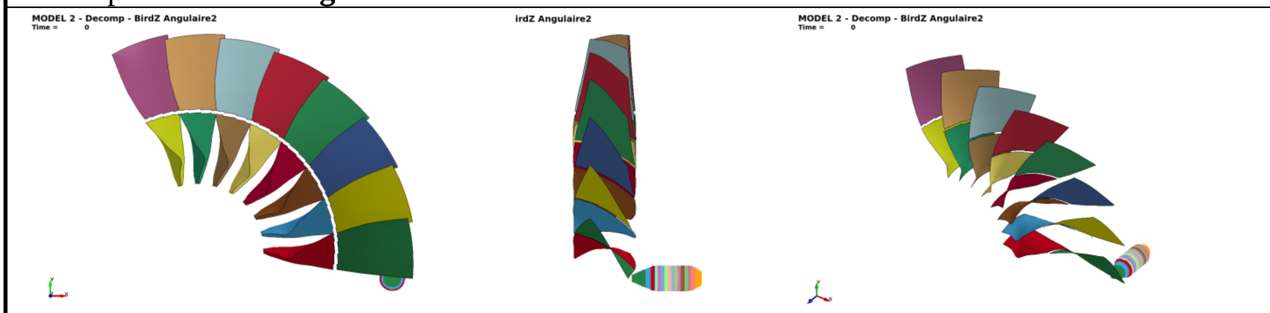
Table 2A: Decomposition & pfile for Case #2

Decomp " **Bird Z + Angular** "



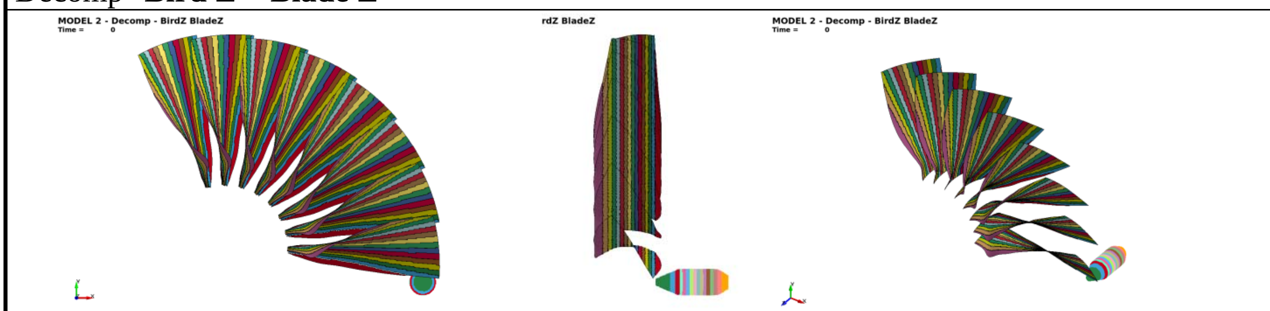
decomp { region {parts 100000 sz 5000 } }
decomp { region {parts 1 2 3 4 5 6 7 8 c2r 0 0 0 0 0 1 1 -1 0 rx 14.485 sy 50 sz 50} }

Decomp " **Bird Z + Angular 2**"



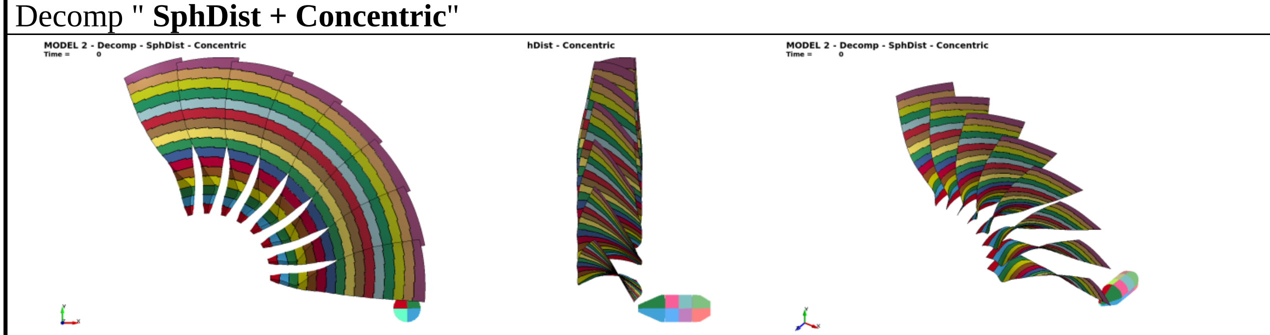
decomp { region {parts 100000 sz 5000 } }
decomp { region {parts 1 2 3 4 5 6 7 8 c2r 0 0 0 0 0 1 1 -1 0 rx 14.495 sy 50 sx 50} }

Decomp "**Bird Z + Blade Z**"



decomp { region {parts 100000 sz 5000 } }
decomp { region {parts 1 2 3 4 5 6 7 8 sz 5000 } }

Decomp " **SphDist + Concentric**"



decomp { sphdist } decomp { region {parts 1 2 3 4 5 6 7 8 c2r 0 0 0 0 0 1 1 -1 0 sx 5000 } }

Table 2B: Decomposition & pfile for Case #2

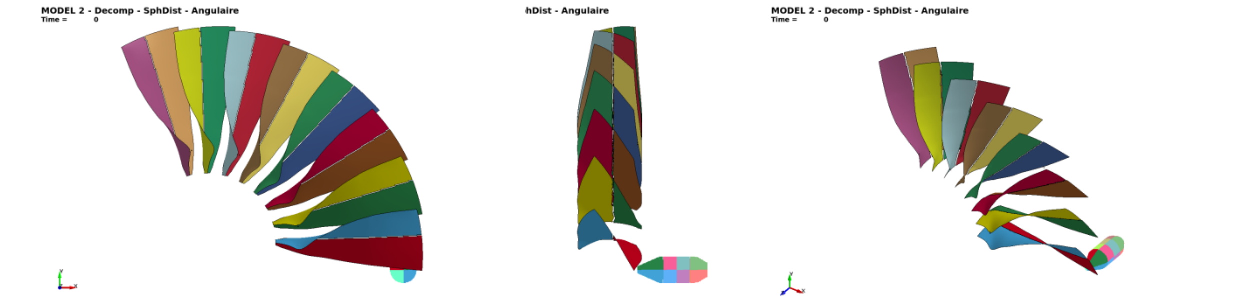
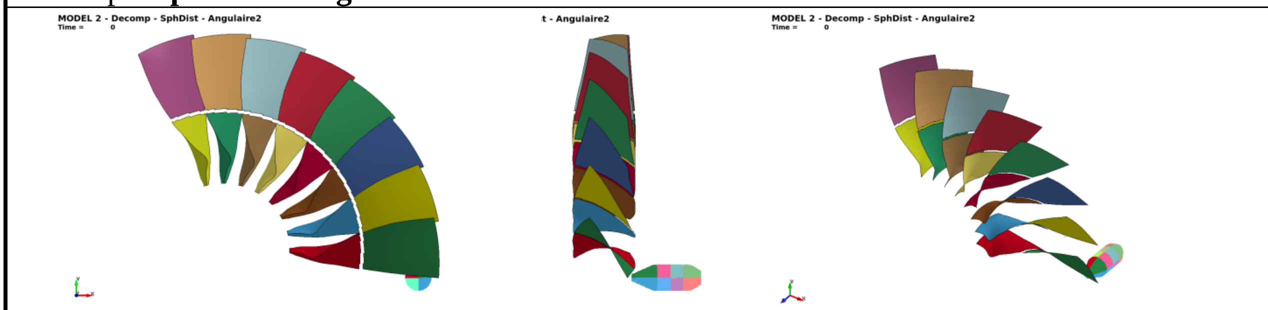
Decomp " **SphDist + Angular**"

decomp { sphdist } *decomp { region {parts 1 2 3 4 5 6 7 8 c2r 0 0 0 0 0 1 1 -1 0 rx 14.485 sy 50 sz 50} }*
Decomp " **SphDist + Angular 2**"

decomp { sphdist } *decomp { region {parts 1 2 3 4 5 6 7 8 c2r 0 0 0 0 0 1 1 -1 0 rx 14.495 sy 50 sx 50} }*

Table 2C: Decomposition & pfile for Case #2

DISCUSSION OF RESULTS

Model at start of simulation Model during bird slice

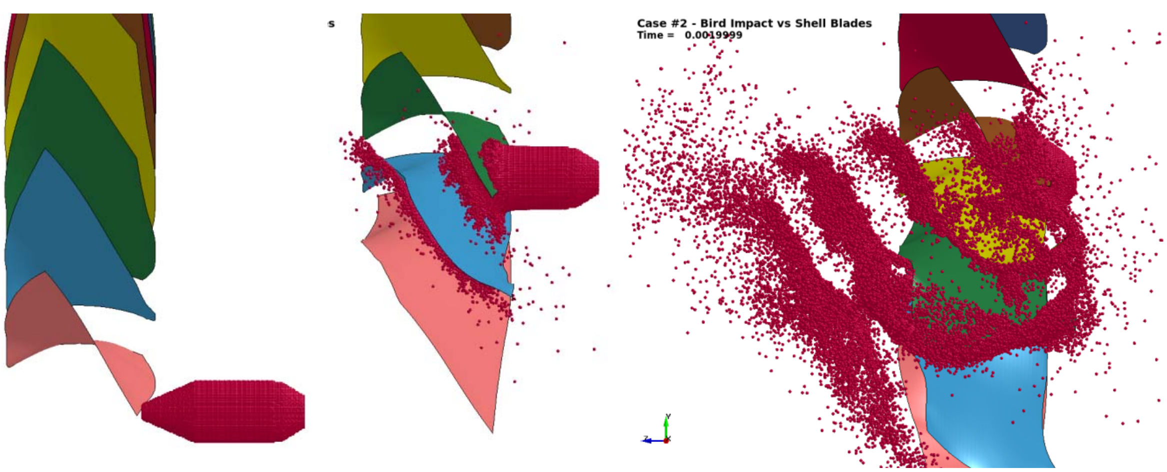


Figure 6: Case #2 – Results of simulation at various state

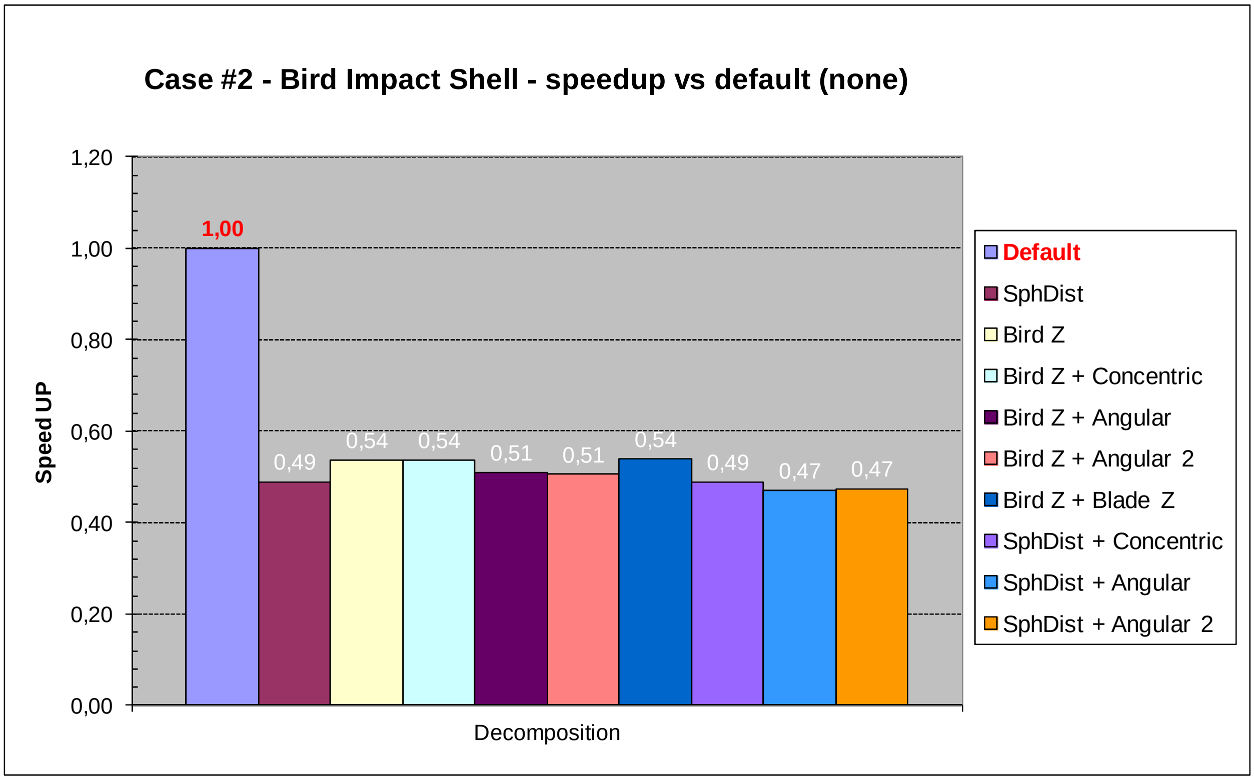


Figure 7: Case #2 – Speedup vs. default

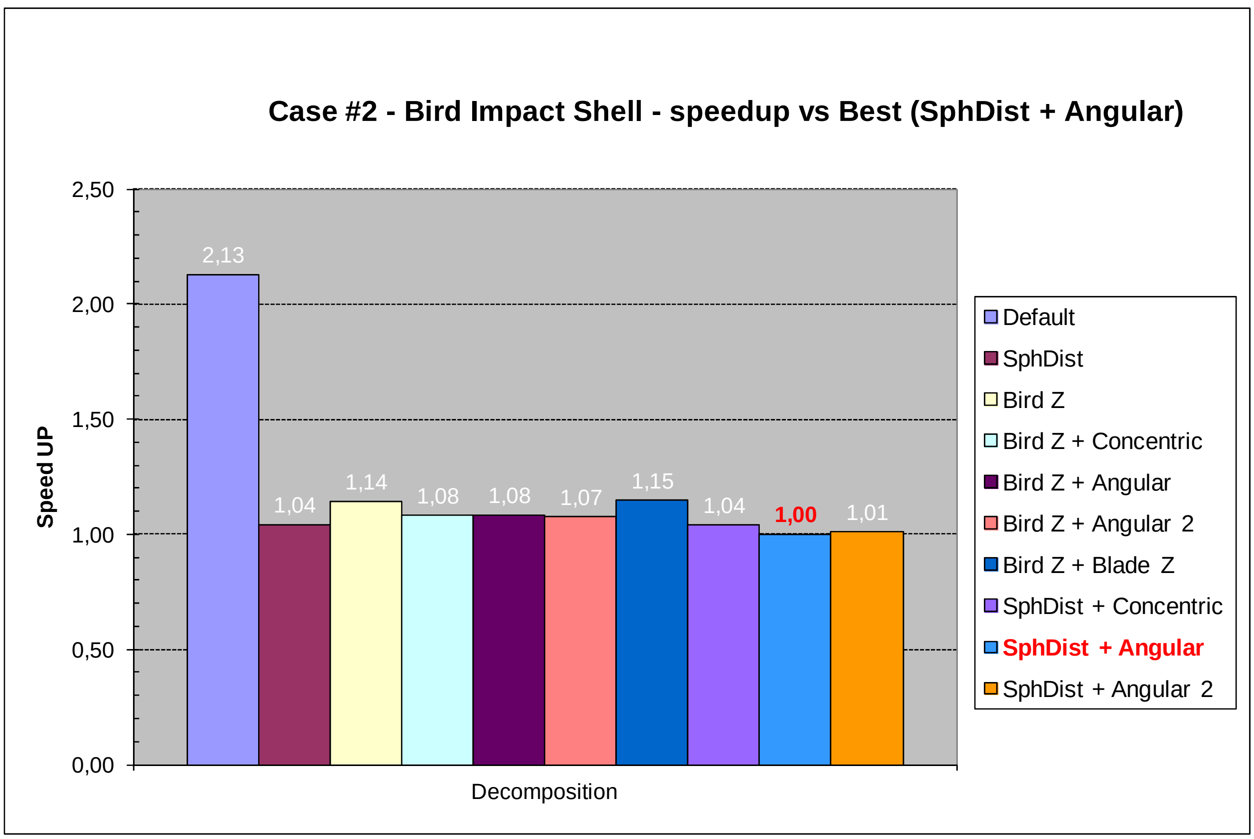


Figure 8: Case #2 – Speedup vs. best (SphDist + Angular)

For the default decomposition, the multiple boundary decomposition of the bird and the varying number of SPH elements in the sub-domains lead to a very poor efficiency of MPP LS-DYNA. This decomposition can be improved in three ways: the first one is to optimize the decomposition of the bird, the second one to optimize the decomposition of of finite elements used for the blades and third one to modify the relative weight of SPH elements in the decomposition when compared to finite elements.

The “SphDist” decomposition is the basic attempt to optimize bird decomposition. In this model, we have forced the SPH decomposition to be done on the 16 cores. The result is an elapsed time divided by more than 2. Yet, in this decomposition we have multiple boundaries between processors in the bird.

Based on results obtained in 2007, we therefore studied five new decompositions for which the bird is decomposed on the 16 cores in a slice pattern and tried different decompositions for the blades. The five decompositions “BirdZ”, “BirdZ + Concentric”, “BirdZ + Angular”, “BirdZ + Angular2” and “BirdZ + Blade Z” are all similar in term of efficiency. They all lead to a reduction roughly equal to 50% in elapsed time when compared to the default.

Yet, unlike observed on 8 cores in 2007, none led to better efficiency than the simple “SphDist” decomposition. As observed in case1, the increased number of cores led to a SPH subdomains with relatively small particles number. The gain obtained with a slice type decomposition on all cores for SPH is outweighed by the additional cost of communication induced by the fact that many more particles are close to the frontiers and induce a need for communication.

In this context, a more simple decomposition of distributing the SPH particles on every cores leads to a better compromise. The last three decompositions combine SphDist decomposition type for the bird and the different advanced decompositions tested previously for the blades. All three are roughly equivalent with optimal decomposition corresponding to the decomposition" SphDist + Angular" that optimizes the communication necessary for contacts.

PARTIAL SYNTHESIS

Case #2 confirms that with increasing n umber of core a slice decomposition for SPH parts is not necessary better. When mixing SPH elements and finite elements, the decomposition of the SPH over all processor leads to a good compromise. If it is not the optimum decomposition, at least it avoids a highly unbalanced decomposition due to relative position and topology of SPH and finite elements parts.

CASE 3: BIRD IMPACT ON FAN BLADES (Solid)

MODEL PRESENTATION

The model used for case #3 is a 1.5 lb bird impacting titanium fan blades modeled with solid elements. This model has been provided by SNECMA. It is representative of a real input deck.

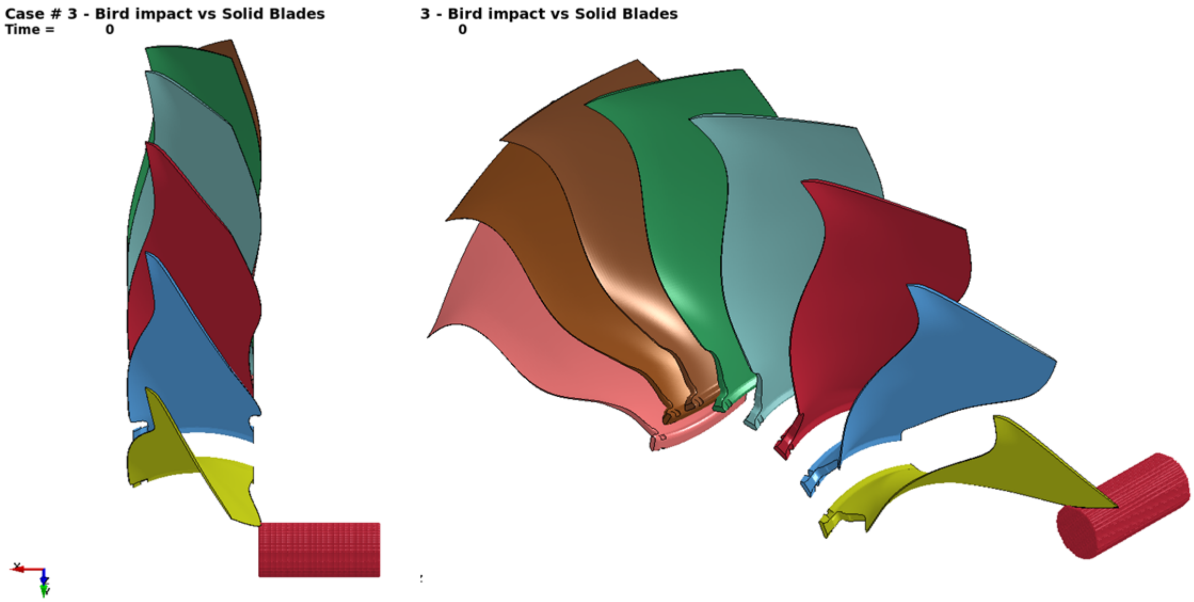


Figure 9: Case #3 – Mesh Views

The bird has a global mass of 1.5 lb. It is associated with a null material law model and a tabulated EOS. The impact occurs at 70 m/s. The inter-particular distance is 2.5 mm. The number of SPH elements in the bird is therefore 46 036.

Eight titanium fan blades are included in the model. A Johnson-Cook model is used for titanium. Each fan blade is meshed with 89534 type 2 fully integrated solid elements. The total number of shell elements is therefore 716 272. The engine is rotating at cruise speed. A contact of the "Node_to_Surface" type is defined between the bird and the blades.

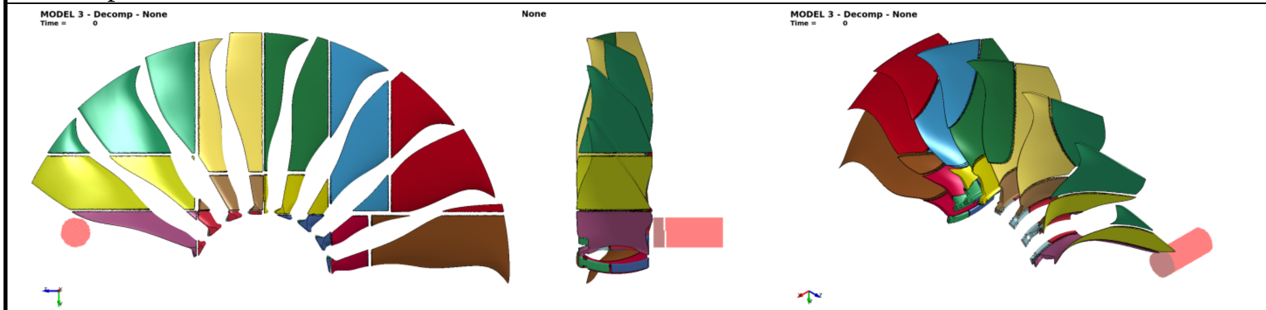
DECOMPOSITION

The RCB default decomposition leads to mixed domains of pure SPH, pure finite elements and mixed element types.

A first attempt to optimize the decomposition is the "SphDist" decomposition, where the Bird is decomposed over the 16 cpu.

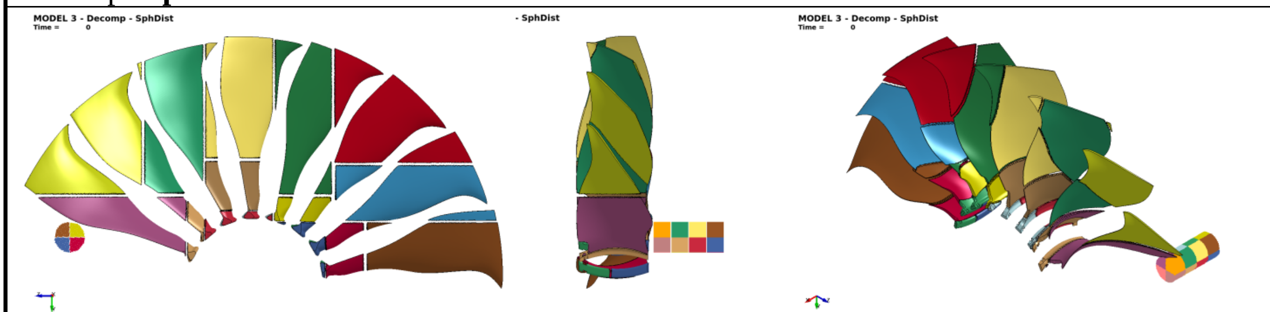
Based on results obtain in the 2007 paper, a third model "BirdX" was studied. For this model the bird is decomposed on the 16 CPU but in slice along X direction. For the two last decompositions ("BirdX + Concentric" & "BirdX + Blade X"), the bird is still decomposed over the 16 cpu in a slice X type decomposition and effort is also applied to the blades decomposition.

Decomp "**Default**"



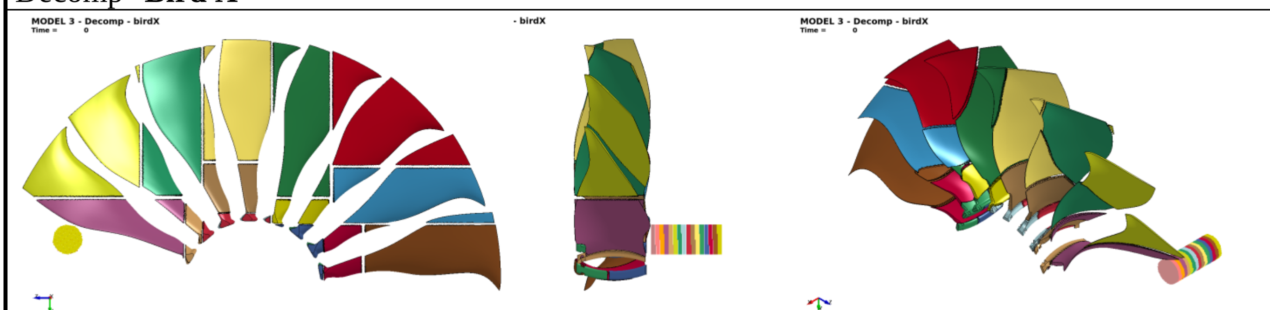
none

Decomp "**SphDist**"



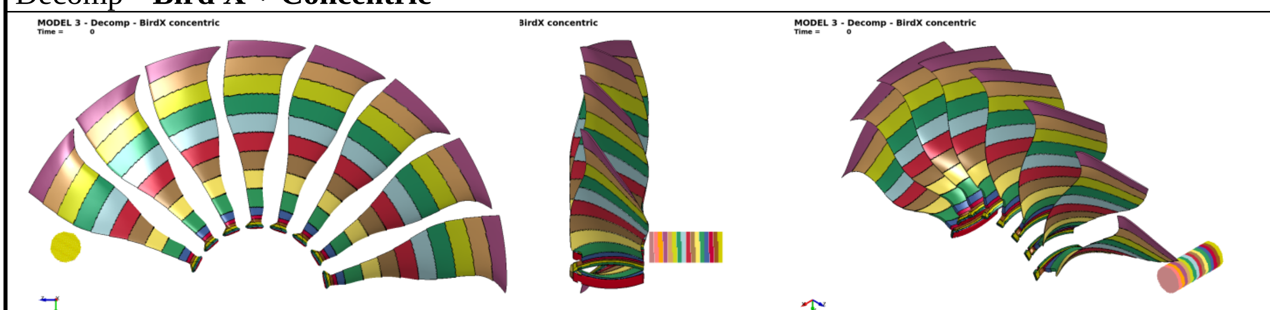
decomp { sphdist }

Decomp "**Bird X**"



decomp { region {parts 7000 sx 5000 } }

Decomp " **Bird X + Concentric** "



decomp { region {parts 7000 sx 5000 } }
decomp { region {parts 101 201 301 401 501 601 701 801 c2r 0 0 0 1 0 0 0 1 -1 sx 5000 } }

Table 3A: Decomposition & pfile for Case #3

Decomp "**Bird X + Blade X**"

decomp { region {parts 7000 sx 5000 } } *decomp { region {parts 101 201 301 401 501 601 701 801 sx 5000 } }*

Table 3B: Decomposition & pfile for Case #3

DISCUSSION OF RESULTS

Model during bird slice

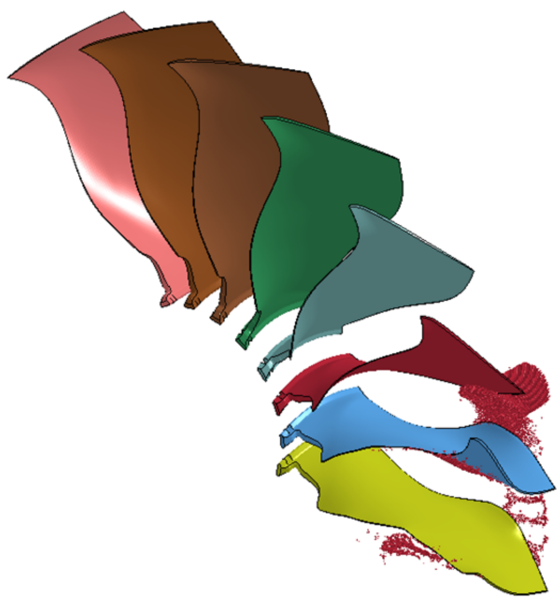


Figure 10: Case #3 – Results of simulation at various state

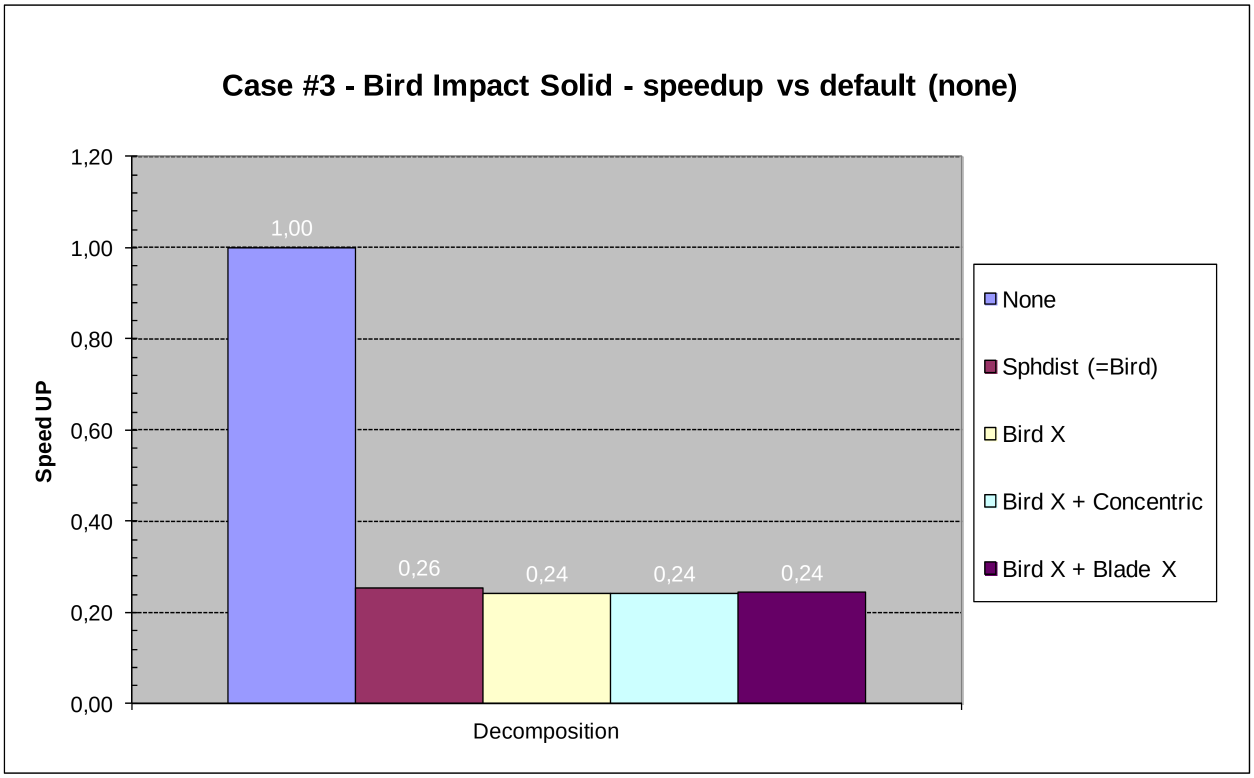


Figure 11: Case #3 – Speedup vs. default

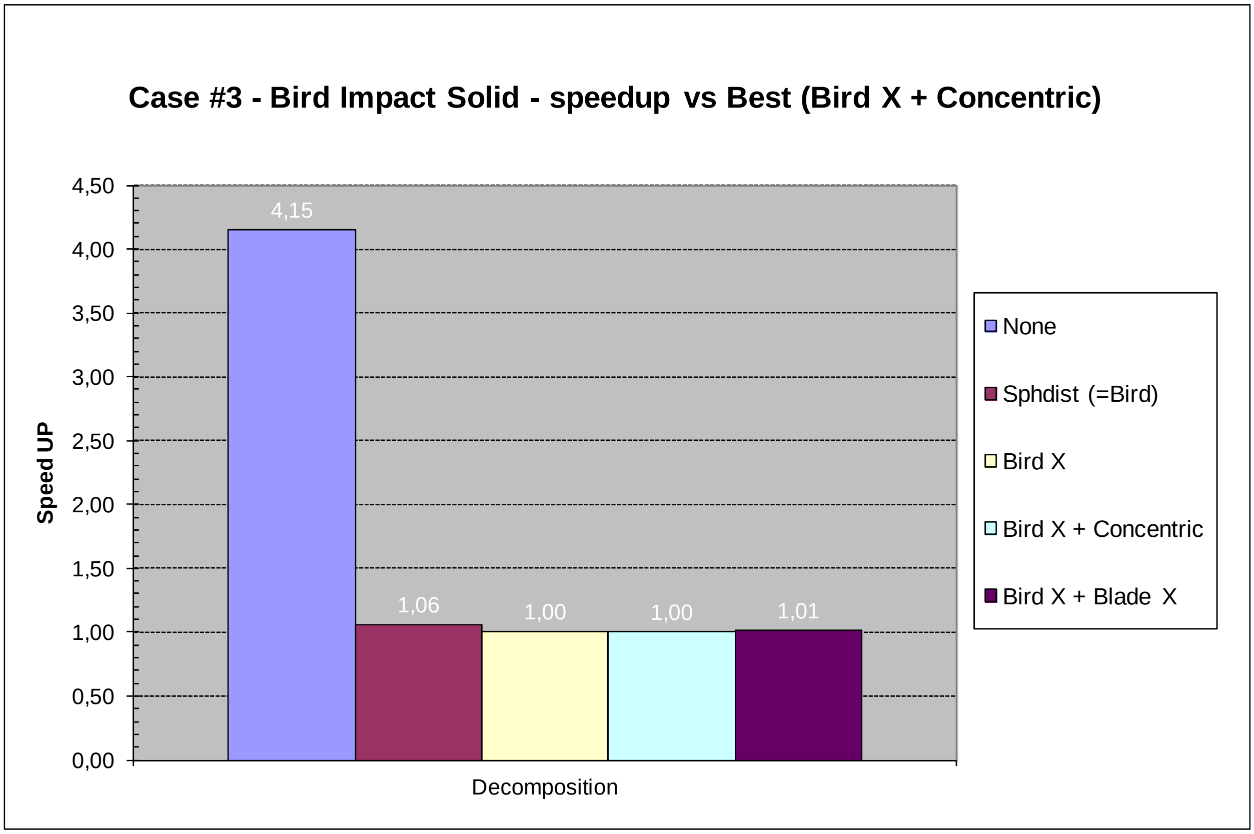


Figure 12: Case #3 – Speedup vs. best (BirdX + Concentric)

For the default decomposition, the multiple boundary decomposition of the bird and the varying number of SPH elements in the sub-domains lead to a very poor efficiency of MPP LS-DYNA.

The “SphDist” decomposition is the basic attempt to optimize bird decomposition. In this model, we have forced the SPH decomposition to be done on the 16 processors. The result is an elapsed time divided by more than 4.

Based on results obtained in 2007, we therefore studied three new decompositions for which the bird is decomposed on the 16 cores in a slice pattern and tried different decompositions for the blades. The five decompositions “BirdX”, “BirdX + Concentric” and “BirdX + Blade X” are all very similar in term of efficiency. They all lead to a reduction roughly equal to 75% in elapsed time when compared to the default and slightly better than the SphDist decomposition (6%).

PARTIAL SYNTHESIS

When mixing SPH elements and finite elements, the decomposition of the SPH over all processor leads to a good compromise and can induce significant speedup compare to default decomposition. If it is not the optimum decomposition, at least it avoids a highly unbalanced decomposition due to relative position and topology of SPH and finite elements parts.

In case 3, the very low number of particles per core almost automatically leads to a need for communication with another subdomain for all particles. In this case, the fact of decomposing bird in slice avoids having a need for communication between more than two cores and slightly improves the effectiveness of the decomposition.

SUMMARY AND CONCLUSIONS

Three models have been tested for this paper. For each model, it has been demonstrated that special data partitioning techniques can significantly improve the efficiency.

Generally speaking, for SPH parts, decompositions where boundaries between more than 2 sub-domains are reduced (or, better yet, suppressed if possible) are more efficient.

When mixing SPH elements and finite elements, the decomposition of the SPH over all processors leads to a good compromise. If it is not the optimum decomposition, at least it avoids highly unbalanced decomposition due to relative position and topology of SPH and finite elements parts.

With the increasing number of cores, the number of particles per core may be relatively low. In this case the communication needs can increase very fast way because too many particles are found on the frontiers.

ACKNOWLEDGMENT

Dr. A. MILLECAMPS of SNECMA is acknowledged for providing the bird impact models. Dr. J. Wang of LSTC is acknowledged for his helpful comments on the decomposition and results.

REFERENCES

1. LS-DYNA Keyword User's Manual, Version R7.0, LSTC, Livermore, February 2013.
2. V. LAPOUJADE, A. SUFFIS, J.L. LACOME “MPP Decomposition of a SPH Model”, 6th European LS-DYNA Users’ Conference, Gothenburg - SWEDEN, 2007.