Nationaal Lucht- en Ruimtevaartlaboratorium

National Aerospace Laboratory NLR

NLR-TP-98198

An algorithm to check the topological validity of multi-block domain decompositions

S.P. Spekreijse and J.W. Boerstoel

Nationaal Lucht- en Ruimtevaartlaboratorium

National Aerospace Laboratory NLR

NLR-TP-98198

An algorithm to check the topological validity of multi-block domain decompositions

S.P. Spekreijse and J.W. Boerstoel

This report is based on a presentation to be held at the 6th International Conference on Numerical Grid Generation in Computational Field Simulations, University of Greenwich, London, England, 6th-9th July 1998.

This investigation has been carried out under a contract awarded by the Netherlands Agency for Aerospace Programs, contract number 01105N.

Division:	Informatics/Fluid Dynamics
Issued:	April 1998
Classification of title:	Unclassified

Summary

An algorithm is presented which answers the question whether it is possible to construct a C^0-continuous multi-block grid in a given, arbitrary multi-block domain decomposition. This answer is defined by the topological connectivity relations of a multi-block domain. The algorithm is very useful to reduce problem turnaround times when solving CFD problems.

Contents

(18 pages in total)

1 Introduction

Nowadays, multi-block structured grids are widely used for solving Reynolds-averaged Navier-Stokes (RANS) equations for steady flows over complex configurations. Three types of multi-block grids are commonly used. In overlapped (overset) grids, adjacent blocks form an overlapped region instead of a clearly identifiable zonal interface. In patched grids, the adjoining blocks form a common interface, but the point distribution on the block boundaries that form the interface is different. In C^0-continuous grids, the adjoining blocks that form the interface have a point-to-point match at the zonal interface. With regard to flow solvers, C^0-continuous grids are the simplest and most convenient to deal with. One important advantage of C^0-continuous grids is that conservation is obtained across block interfaces. Therefore, C^0-continuous grids are most often used in practice.

In practice, two types of multi-block domain decompositions are used to construct C^0-continuous grids (Ref.1). The simplest one requires complete block boundary interfacing, in which the blocks are packed block-face-to-block-face. However, this is very restrictive and leads to many blocks in domains about complex configurations (like a complete aircraft). Therefore, most often, partial block boundary interfacing is allowed, in which a face of a block can be adjacent to more than one other block. An illustration of a complete block boundary interfacing domain decomposition is shown in Fig.2, which is a simple 2D domain decomposition of a part of the harbour of Rotterdam. A partial block boundary interfacing domain decomposition, which requires much less blocks, is shown in Fig.3. The corresponding multi-block grid is shown in Fig.4.

In this paper we consider partial block boundary interfacing multi-block domain decompositions to be used for the construction of C^0-continuous grids. A complicating effect of the use of partial block boundary interfacing is that it may not always be possible to compute a C^0-continuous grid.

Domain decompositions for complex configurations may consist of many blocks (typically hundreds of blocks may be needed for viscous flow simulations about complete aircraft). A CFD engineer who is constructing the blocks (domain decomposition), with the purpose to generate, at a later time, a C^0-continuous grid, may not be aware that the constructed blocks are not suitable to construct a C^0-continuous grid. Detection of this shortcoming during the later stages of the grid generation process is inefficient, because then the CFD engineer has to go back to the domain decomposition process and must re-do (partially) the domain decomposition. Thus it is very useful that a CFD engineer can always check, during the domain decomposition process, whether the current block decomposition can be used to generate a C^0-continuous grid.

In this paper we will describe an algorithm which answers the question whether a given, arbitrary, partial block boundary interfacing, multi-block domain decomposition can be used to construct a C^0-continuous grid.

The algorithm has been implemented in the domain decomposer ENDOMO, which is a graphical interactive program for the domain decomposition of arbitrary flow domains in blocks (Ref.2). ENDOMO is part of NLR's Euler/Navier-Stokes flow calculation system ENFLOW (see Fig.1), (Ref.3). The grid generator ENGRID is used to compute C^0-continuous grids in a multi-block domain decomposition constructed with ENDOMO (Ref.4). Given a C^0-continuous grid, the flow solver ENSOLV computes the solution of the 3D flow equations based on RANS (Ref.5).

2 Topological elements and their connectivity relations

The topology of a multi-block domain decomposition consists of blocks,faces,edges and vertices together with connectivity relations between these elements. As a preparation to the algorithm to be presented, a data structure for these connectivity relations will be introduced in this section.

A block has six faces, twelve edges, and eight vertices as topological boundary elements. A face has four edges and four vertices as topological boundary elements. An edge has two vertices as topological boundary elements.

A block, face, edge and vertex is identified by a label B, F, E, V, respectively, from sets $\{B\}$, $\{F\}$, $\{E\}$, and $\{V\}$, which are so-called ”label sets”. In practice, positive integers are used as labels.

The way how blocks, faces, edges, and vertices are connected to each other over common boundary elements defines the topology of a domain decomposition. Three types of basic connectivity relations are needed to define the topology of a complete block boundary interfacing multi-block domain decomposition. Two additional basic connectivity relations are needed to allow partial block boundary interfacing mentioned in the introduction.

The first type of basic connectivity relations concern block-to-face connectivities.These are mappings of the form

$$\forall\, B \in \{B\} : \; B \mapsto (F_1, F_2, F_3, F_4, F_5, F_6), \tag{1}$$

with F_i the labels of the six faces of block B. The face labels are assumed to be ordered such that (F_1, F_2),(F_3, F_4) and (F_5, F_6) are pairs of opposite faces in block B. Two blocks will be connected to each other block-face-to-block-face, if they have at least one common face label in their block-to-face relations.

The second type of basic connectivity relations concern face-to-edge connectivities. These are mappings of the form

$$\forall\, F \in \{F\} : \; F \mapsto (E_1, E_2, E_3, E_4), \tag{2}$$

with E_i the labels of the four edges of face F. Here, too, the edge labels are assumed to be ordered such that (E_1, E_2) and (E_3, E_4) are pairs of opposite edges in face F. Two faces will be connected to each other face-edge-to-face-edge, if they have at least one common edge label in their face-to-edge relations.

The third type of basic connectivity relations concern edge-to-vertex connectivities. These are mappings of the form

$$\forall\ E \in \{E\} :\ E \mapsto (V_1, V_2), \tag{3}$$

with V_i the labels of the two vertices of edge E. Two edges will be connected to each other edge-vertex-to-edge-vertex, if they have at least one common vertex in their edge-to-vertex relations.

Derived connectivity relations, such as block-to-edge, block-to-vertex and face-to-vertex, can be computed from the basic connectivity relations.

Compound edges and compound faces are introduced to allow partial block boundary interfacing.

A compound face is defined as a face consisting of two subfaces that are joined together at a common edge. Each subface of a compound face is also allowed to be compound again. A face which is not compound is elementary. The set of compound and elementary faces are denoted as $\{F^c\}$ and $\{F^e\}$. Thus $\{F\} = \{F^e\} \cup \{F^c\}$.

A compound edge is an edge consisting of two subedges, that are joined together at a common vertex. Each subedge of a compound edge is also allowed to be compound again. An edge which is not compound is elementary. The set of compound and elementary edges are denoted as $\{E^c\}$ and $\{E^e\}$. Thus $\{E\} = \{E^e\} \cup \{E^c\}$.

The fourth type of basic connectivity relations concern compound-face-to-subface connectivities. These are mappings of the form

$$\forall\ F \in \{F^c\} :\ F \mapsto (F_1, F_2), \tag{4}$$

giving the two subfaces F_1 and F_2 of compound face F.

The fifth and last type of basic connectivity relations concern compound-edge-to-subedge connectivities. These are mappings of the form

$$\forall\, E \in \{E^c\} : \; E \mapsto (E_1, E_2). \tag{5}$$

An elementary edge of which the two vertices in the edge-to-vertex connectivity relation are the same, thus $E \mapsto (V_1, V_2)$ and $V_1 = V_2$, is degenerated to a vertex and called collapsed.

Let the topology of a partial block boundary interfacing multi-block domain decomposition be defined by the above specified basic connectivity relations. Then the question becomes: *'is it possible to construct a C^0-continuous multi-block grid ?'*

Let the grid dimension of an edge be the number of grid cells along that edge. Each pair of opposite edges in a face must have the same grid dimension, and each four opposite edges in a block must have the same grid dimension Due to the constraining effect of the requirement that each grid line in each block must be continuous over any face that the block has in common with any adjacent block (C^0 continuous grid), it is possible to subdivide the set of edges $\{E\}$ into disjunct subsets (called groups) with the property that the grid dimension of all edges in the same group must be the same, while the grid dimensions of two edges in different groups is generally different. Thus each edge (elementary or compound) belongs to exactly one group. Furthermore, simple sum relations between the grid dimensions of different groups may exist due to the existence of compound edges. If, for instance, a compound edge E with subedges E_1, E_2 belong to the groups G and G_1, G_2, respectively, then the grid dimension of group G is equal to the sum of the grid dimensions of groups G_1 and G_2. Denote by $\mathcal{G}$ the grid dimension of group G, i.e. $\mathcal{G} =$ dim G, then $\mathcal{G} = \mathcal{G}_1 + \mathcal{G}_2$.

An exception is made for collapsed edges. Collapsed edges do not belong to a group because their grid dimension is not unique. This is illustrated in Fig.5 where edge E is a collapsed edge. In this figure, edge E_4 is opposite to edge E in face F_1, edge E_5 is opposite to edge E in face F_2, and edge E_6 is opposite to edge E in face F_3. Thus if it was required that the grid dimension of collapsed edge E is unique then this would imply that edges E_4, E_5, E_6 have the same grid dimension and consequently belong to the same group. This is of course unacceptable.

The sum relations between the groups can be used to detect whether it is possible to construct a C^0-continuous grid in a multi-block domain decomposition. As an illustration, consider the two-dimensional domain decomposition as shown in Fig.6. Define group $G_1 = \{E_1\}$. Edges E_2 and

E_4 belong to the same group because they are opposite edges in face F_2. Furthermore, compound edge E_3 also belongs to this group because it is an opposite edge of E_4 in face F_1. Define group $G_2 = \{E_2, E_3, E_4\}$. Edge E_7 and edge E_5 also belong to the same group because they are opposite edges in face F_1, and edge E_6 and edge E_8 belong to the same group because they are opposite edges in face F_2. Define groups $G_3 = \{E_5, E_7\}$ and $G_4 = \{E_6, E_8\}$. Thus four groups are detected. Because of the compound edge E_3, we have dim $E_1 +$ dim $E_2 =$ dim E_3, and consequently $\mathcal{G}_1 + \mathcal{G}_2 = \mathcal{G}_2$. From this relation it follows that the grid dimension of group G_1 (and thus of edge E_1) must be equal to zero. This means that it is not possible to generate a C^0-continuous grid for this case.

We will generalize this method for arbitrary, partially boundary interfacing, multi-block domain decompositions in the next section.

3 The algorithm

An arbitrary multi-block domain decomposition has n groups. Each group consists of edges with the same grid dimension. These groups are automatically generated from the topological connectivity relations. If partial block boundary interfacing is applied, then also m sum relations between the grid dimensions of the groups exist. Also the sum relation are automatically generated from the topological connectivity relations. Denote by $\mathcal{G}_i = \dim G_i$, the grid dimension of group G_i. Thus m sum relations of the form

$$\mathcal{G}_i + \mathcal{G}_j = \mathcal{G}_k \ , \ i, j, k \in \{1 \ldots n\}. \tag{6}$$

exist. This system of equations is of course underdetermined because otherwise the grid dimensions would be determined by the topological connectivity relations. In practice $m \ll n$. Thus the grid dimensions $\mathcal{G}_i$ are in general not defined by Eq.(6). However, like in the example of Fig.6 discussed above, the structure of this system of equations can be such that one or more grid dimensions $\mathcal{G}_i$ are defined and equal to zero. In that case, the topology of the multi-block domain decomposition is incorrect and no C^0-continuous multi-block grid can be generated. We will use the following theorem to determine if one or more $\mathcal{G}_i$ is defined and equal to zero.

Consider a general matrix equation $A\vec{x} = 0$ where A is a real $m \times n$ matrix and $\vec{x} \in \mathcal{R}^n$. Furthermore, $m < n$ so that the system of equations is underdetermined. Denote by $\vec{a}_1 \ldots \vec{a}_m$ the row vectors of the matrix A. Thus $\vec{a}_k \in \mathcal{R}^n$ and $A\vec{x} = 0 <=> (\vec{a}_k, \vec{x}) = 0$ for all $k = \{1 \ldots m\}$.

Define for all $i \in \{1 \ldots n\}$, the matrices

$$\begin{aligned} B_i &= (\vec{a}_1, \ldots, \vec{a}_m, \vec{e}_i), && (7) \\ C_i &= B_i^T B_i, && (8) \end{aligned}$$

where $\vec{e}_i \in \mathcal{R}^n$ is the unit vector in i-direction. Notice that C_i is a symmetric semi-positive-definite $(m+1) \times (m+1)$ matrix.

Theorem

The matrix equation $A\vec{x} = 0$ implies $x_i = 0$ if and only if the matrix C_i has a zero eigenvalue and the $(m+1)$th component of the corresponding eigenvector is non-zero.

Proof

Suppose that C_i has a zero eigenvalue. Then an eigenvector $\vec{\alpha} = (\alpha_1 \dots \alpha_{m+1})^T$ exists such that $C_i\vec{\alpha} = 0$. Thus $B_i^T B_i \vec{\alpha} = 0$ so that $(B_i^T B_i \vec{\alpha}, \vec{\alpha}) = 0 <=> (B_i\vec{\alpha}, B_i\vec{\alpha}) = 0 <=> B_i\vec{\alpha} = 0$. Thus we have found that coefficients $\alpha_1 \dots \alpha_{m+1}$ exist such that

$$\alpha_1\vec{a}_1 + \dots + \alpha_m\vec{a}_m + \alpha_{m+1}\vec{e}_i = 0. \tag{9}$$

From this relation and using $A\vec{x} = 0$ it follows that $\alpha_{m+1}(\vec{x}, \vec{e}_i) = \alpha_{m+1}x_i = 0$. Under the assumption that $\alpha_{m+1} \neq 0$ we find that $x_i = 0$.

If, on the other hand, $A\vec{x} = 0$ implies that $x_i = 0$, then coefficients $\alpha_1 \dots \alpha_m$ exist such that $\alpha_1\vec{a}_1 + \dots + \alpha_m\vec{a}_m = \vec{e}_i$. Then the vector $\vec{\alpha} = (\alpha_1, \dots, \alpha_m, -1)^T$ is an eigenvector of C_i with eigenvalue zero. $\square$

Notice that if the row vectors $\vec{a}_1 \dots \vec{a}_m$ of matrix A are linearly independent, then the coefficient α_{m+1} cannot be zero in Eq.(9). Thus we also have

Corollary

If the row vectors of matrix A are linearly independent then the matrix equation $A\vec{x} = 0$ implies $x_i = 0$ if and only if the matrix C_i has a zero eigenvalue. $\square$

However, we will not assume that the row vectors are linearly independent.

The sum relations given by Eq.(6) may have a structure such that one or more grid dimension $\mathcal{G}_i$ of the groups is defined and equal to zero. To check this, the sum relations given by Eq.(6) are assembled into a matrix equation $A\vec{\mathcal{G}} = 0$ where A is a real $m \times n$ matrix and $\vec{\mathcal{G}} = (\mathcal{G}_1, \dots, \mathcal{G}_n)^T$. Next, n matrices C_i are computed according to Eq.(8). Each matrix C_i is a symmetric semi-positive-definite $(m+1) \times (m+1)$ matrix. If a matrix C_i has a zero eigenvalue and the last coefficient of the corresponding eigenvector is none zero, then the grid dimension of group G_i must be zero so that it is not possible to construct a C^0-continuous multi-block grid.

Thus we need to compute the eigenvalues and eigenvectors of n symmetric semi-positive-definite $(m+1) \times (m+1)$ matrices. For this purpose we use the algorithm RSP from the package SEISPACK retrieved from NETLIB which can be found at the Internet address `http://www.netlib.org/seispack/index.html`.

The following two examples are illustrations of the algorithm. In example 2, the row vectors of matrix A are linearly dependent.

Example 1

$n = 3$ and $m = 2$ and the sum relations are given as

$$\mathcal{G}_1 + \mathcal{G}_2 = \mathcal{G}_3 \tag{10}$$

$$\mathcal{G}_2 + \mathcal{G}_3 = \mathcal{G}_1 \tag{11}$$

The matrix A becomes

$$A = \begin{pmatrix} 1 & 1 & -1 \\ -1 & 1 & 1 \end{pmatrix}, \tag{12}$$

and it can be easily verified that the matrix C_2 is

$$C_2 = \begin{pmatrix} 3 & -1 & 1 \\ -1 & 3 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \tag{13}$$

so that C_2 has a zero eigenvalue with corresponding eigenvector $(1, 1, -2)$. Thus $\mathcal{G}_2 = 0$. A multi-block domain decomposition that will lead to the sum relations given by Eq.(10) and Eq.(11) is shown in Fig.7. Thus a C^0-continuous multi-block grid can not be generated in this domain decomposition.

Example 2

$n = 3$ and $m = 3$ and the sum relations are given as

$$2\mathcal{G}_1 = \mathcal{G}_2 \tag{14}$$

$$2\mathcal{G}_3 = \mathcal{G}_2 \tag{15}$$

$$\mathcal{G}_1 + \mathcal{G}_3 = \mathcal{G}_2 \tag{16}$$

The matrix A becomes

$$A = \begin{pmatrix} 2 & -1 & 0 \\ 0 & -1 & 2 \\ 1 & -1 & 1 \end{pmatrix}, \tag{17}$$

and it can be easily verified that, for example, matrix C_1 is

$$C_1 = \begin{pmatrix} 5 & 1 & 3 & 2 \\ 1 & 5 & 3 & 0 \\ 3 & 3 & 3 & 1 \\ 2 & 0 & 1 & 1 \end{pmatrix}, \tag{18}$$

so that C_1 has a zero eigenvalue with corresponding eigenvector $(1, 1, -2, 0)$. Thus the last component of the eigenvector is zero so that $\mathcal{G}_1$ is undefined.

Remark 1

The set of edges is subdivided into disjunct subsets which we call groups. In the same way, it is also possible to subdivide the set of groups into disjunct subsets which we call families. The grid dimensions of the groups are only related to each other, by the sum relations of Eq.(6), within the same family. This property characterizes a family. Thus the grid dimensions of two groups in different families are completely independent. Each group belongs to exactly one family.

The algorithm to check if a grid dimension of a group equals zero can be applied on each family separately instead of on the complete set of groups. This makes the algorithm computationally much more efficient because it is more efficient to compute eigenvalues and eigenvectors of small matrices C_i for each family separately than to compute the eigenvalues and eigenvectors of large matrices C_i for the complete set of groups.

Remark 2

The following numbers give an impression about the number of blocks, groups (n), sum relations (m), and families for a standard application. This standard application concerns a multi-block domain decomposition about a wing-body-pylon-nacelle configuration to be used for Navier-Stokes flow simulations. The domain decomposition consists of 239 blocks. The total number of groups is 98 and the total number of sum relations between the groups is 80. The groups could be subdivided into 8 families. The largest family contains 50 groups with 39 sum relations. It took less than one minute on a Silicon Graphics Indy workstation to verify that the topology of the domain decomposition was correct.

Remark 3

Notice that the presented algorithm to detect whether the grid dimension of a group equals zero, does not make use of the binary tree structure of the compound edges and the related corresponding form of the sum relations in Eq.(6) (the matrix A is a general $m \times n$ matrix). Therefore, the algorithm can also be used for partial block boundary interfacing multi-block domain decompositions described by other topological rules.

Remark 4

The presented algorithm detects from the topology whether a group exist with a grid dimension equal to zero. When this occurs then no C^0-continuous multi-block grid can be computed. It is clear that this is a necessary condition. It is likely that it is also a sufficient condition in the sense that if the algorithm detects no group with zero grid dimension then it will always be possible to generate a C^0-continuous multi-block grid. However, we do not have a proof for this conjecture.

4 Conclusions

A domain decomposition of a 3D flow domain about a complex configuration may consist of many blocks and can be very complicated. A CFD engineer who is constructing the blocks, with the purpose to generate, at a later time, a C^0-continuous grid, may not always be aware that the domain decomposition is not suitable to construct a C^0-continuous grid. Especially, this may happen for complicated geometries. Detection of this shortcoming during the later stages of the grid generation process is inefficient, because then the CFD engineer has to go back to the domain decomposition process and must re-do (partially) the domain decomposition. Thus it is very useful that a CFD engineer can always check, during the domain decomposition process, whether a current block decomposition can be used to generate a C^0-continuous grid. The presented algorithm can be used for this purpose, and appears to be an important tool in the practice of multi-block domain decomposition to reduce problem turnaround times when solving CFD problems.

5 References

1. L.E. Eriksson, E. Orbekk,(1989), Algebraic Block-Structured Grid Generation based on a Macro-Block Concept. In: Applications of Mesh Generation to Complex 3-D Configurations, AGARD-CP-464.
2. S.P.Spekreijse,J.W. Boerstoel,(1996), Multiblock Grid Generation. Part 1: Elliptic Grid Generation Methods for Structured Grids. Part 2: Multiblock Aspects. Von Karman Institute for Fluid Dynamics (VKI). Lecture Series 1996-06. 27th Computational Fluid Dynamics Course. NLR-TP-96338, Part 1, pp.1-42, Part 2, pp. 1-51.
3. J.W. Boerstoel, A. Kassies, J.C. Kok, S.P. Spekreijse, (1996), ENFLOW, A Full-Functionality System of CFD Codes for Industrial Euler/Navier-Stokes Flow Computations. Paper IAST075, presented at the 2nd International Symposium on Aeronautical Science and Technology (IASTTI'96), Djakarta, Indonesia, 1996. NLR-TP-96286, pp.1-25.
4. S.P. Spekreijse, (1995), Elliptic Grid Generation Based on Laplace Equations and Algebraic Transformations, J. Comput. Phys. 118, 38-61. NLR-TP-94102.
5. J.C.Kok, J.W.Boerstoel, A. Kassies, S.P. Spekreijse, (1996), A Robust Multi-Block Navier-Stokes Flow Solver for Industrial Applications. Presented at the third ECCOMAS Computational Fluid Dynamics Conference, Paris, 1996. NLR-TP-96323, pp.1-11.

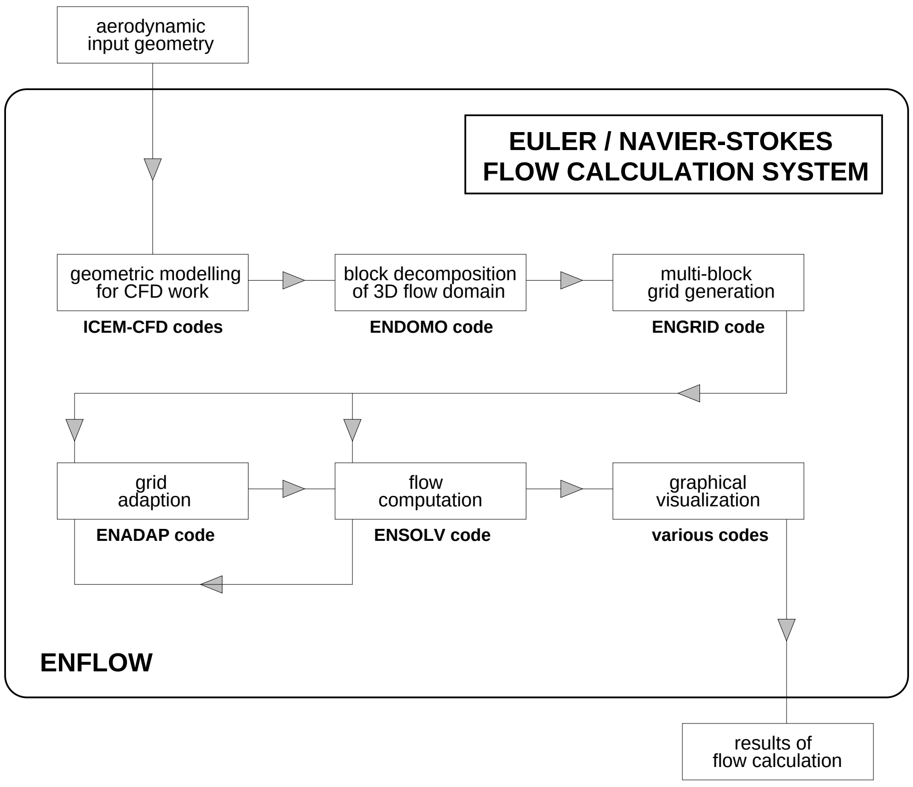
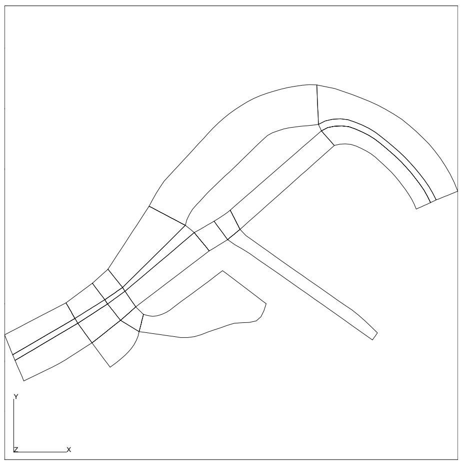

aerodynamic input geometry
EULER / NAVIER-STOKES FLOW CALCULATION SYSTEM
geometric modelling for CFD work
ICEM-CFD codes
block decomposition of 3D flow domain
ENDOMO code
multi-block grid generation
ENGRID code
grid adaption
ENADAP code
flow computation
ENSOLV code
graphical visualization
various codes
ENFLOW
results of flow calculation

Fig. 1 The NLR ENFLOW system.



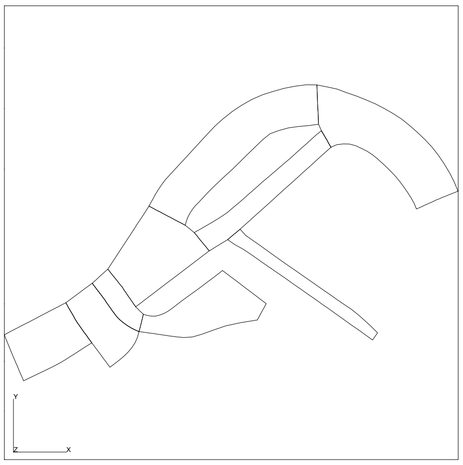


Fig. 2 Domain decomposition with complete block boundary interfacing.

Fig. 3 Domain decomposition with partial block boundary interfacing.

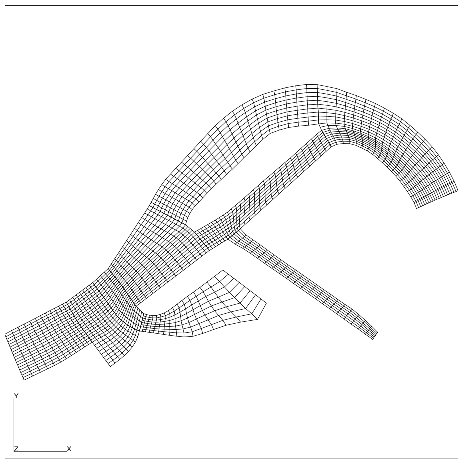


Fig. 4 C^0-continuous multi-block grid.

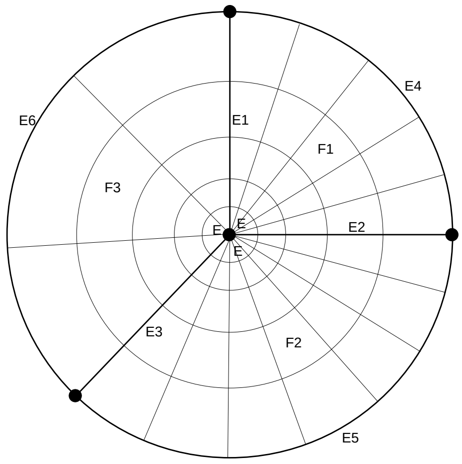


Fig. 5 The number of grid cells along a collapsed edge is not unique.

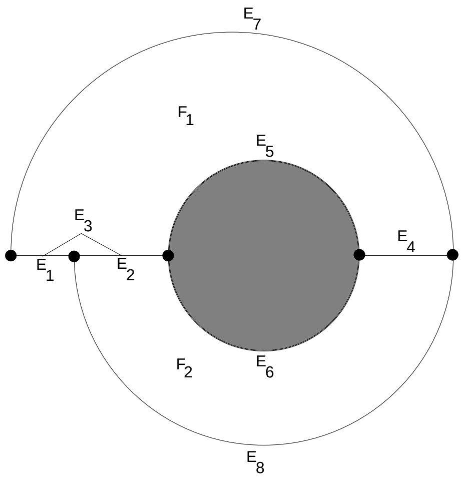


Fig. 6 A very simple example of a domain decomposition for which it is not possible to generate a C^0-continuous grid.

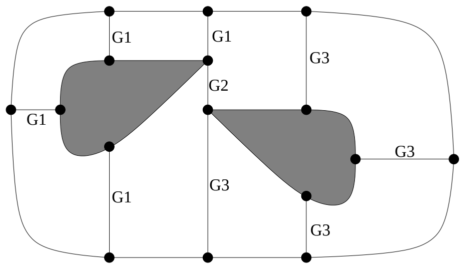


Fig. 7 Other example of a domain decomposition for which it is not possible to generate a C^0-continuous grid.