Parallélisation du code Mistral, solveur par éléments finis d'un problème de MHD bifluide tridimensionnel

A. Orriols,

J.-F. Gerbeau, C. Le Bris et T. Lelièvre

orriols@cermics.enpc.fr

CERMICS - École Nationale des Ponts et Chaussées

9 février 2005

Plan de l'exposé

- Introduction
- Le code *Mistral*
- La librairie *Aztec*
- Parallélisation du code *Mistral* avec *Aztec*
- Conclusion

La nécessité de la parallélisation

Le modèle: Navier-Stokes + Maxwell + loi d'Ohm $\longrightarrow$ MHD

La nécessité de la parallélisation

Le modèle: Navier-Stokes + Maxwell + loi d'Ohm $\rightarrow$ MHD

$$
\left\{
\begin{array}{rcl}
\textbf{sur } \Omega \textbf{ :} & & \\
\partial_t \rho + \operatorname{div}(\rho u) & = & 0 \\
\partial_t (\rho u) + \rho u \cdot \nabla u - \operatorname{div}(2\nu\, d(u)) + \nabla p & = & \rho g + \dfrac{1}{\mu_0} \operatorname{rot} B \times B \\
\operatorname{div} u & = & 0 \\
\partial_t B + \dfrac{1}{\mu_0} \operatorname{rot}\left(\dfrac{1}{\sigma} \operatorname{rot} B\right) & = & \operatorname{rot}(u \times B) \\
\operatorname{div} B & = & 0 \\
\textbf{sur } \Gamma \textbf{ :} & & \\
u & = & 0 \\
\dfrac{1}{\mu_0 \sigma} \operatorname{rot} B \times n & = & k \times n \\
B \cdot n & = & q
\end{array}
\right.
$$

La nécessité de la parallélisation

Le modèle: Navier-Stokes + Maxwell + loi d'Ohm $\rightarrow$ MHD

- 3D, 7 inconnues scalaires (u, B, p)
- 3 non-linéarités (advection, force de Lorentz, loi d'Ohm)
- problème d'évolution
- contrôle optimal (pour la suite)

La nécessité de la parallélisation

Le modèle: Navier-Stokes + Maxwell + loi d'Ohm $\rightarrow$ MHD

- 3D, 7 inconnues scalaires (u, B, p)
  - nombre élevé de degrés de liberté
- 3 non-linéarités (advection, force de Lorentz, loi d'Ohm)
- problème d'évolution
- contrôle optimal (pour la suite)

La nécessité de la parallélisation

Le modèle: Navier-Stokes + Maxwell + loi d'Ohm $\rightarrow$ MHD

- 3D, 7 inconnues scalaires (u, B, p)
  - nombre élevé de degrés de liberté
- 3 non-linéarités (advection, force de Lorentz, loi d'Ohm)
  - GMRES + point fi xe ou Newton-Raphson
- problème d'évolution

- contrôle optimal (pour la suite)

La nécessité de la parallélisation

Le modèle: Navier-Stokes + Maxwell + loi d'Ohm $\rightarrow$ MHD

- 3D, 7 inconnues scalaires (u, B, p)
  - nombre élevé de degrés de liberté
- 3 non-linéarités (advection, force de Lorentz, loi d'Ohm)
  - GMRES + point fi xe ou Newton-Raphson
- problème d'évolution
  - schéma d'Euler semi-implicite
- contrôle optimal (pour la suite)

La nécessité de la parallélisation

Le modèle: Navier-Stokes + Maxwell + loi d'Ohm $\rightarrow$ MHD

- 3D, 7 inconnues scalaires (u, B, p)
  - nombre élevé de degrés de liberté
- 3 non-linéarités (advection, force de Lorentz, loi d'Ohm)
  - GMRES + point fi xe ou Newton-Raphson
- problème d'évolution
  - schéma d'Euler semi-implicite
- contrôle optimal (pour la suite)
  - algorithme de gradient, état adjoint

Type de parallèlisme retenu

Classifi cation de Flynn (1966) : 2 catégories principales :

- SIMD (Single Instruction, Multiple Data) : synchrone (ex : machines vectorielles)

- MIMD (Multiple Instruction, Multiple Data) : asynchrone (ex : MPI sur réseau CERMICS)

Type de parallèlisme retenu

Classifi cation de Flynn (1966) : 2 catégories principales :

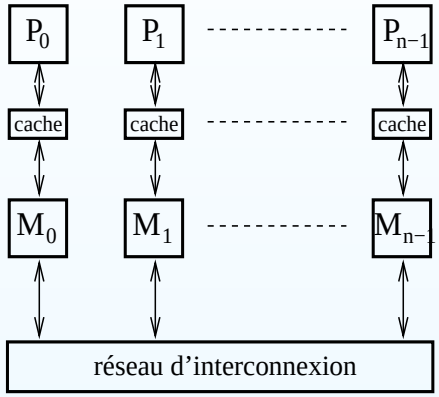
- SIMD (Single Instruction, Multiple Data) : synchrone
(ex : machines vectorielles)

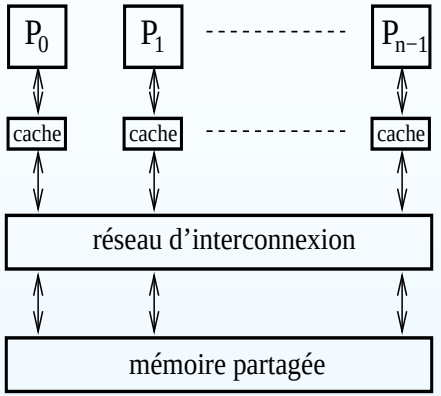
- MIMD (Multiple Instruction, Multiple Data) : asynchrone
(ex : MPI sur réseau CERMICS)

  - SPMD (Single Program, Multiple Data) : exécution asynchrone d'une même instruction sur des données différentes, variante de MIMD

  Avantages : souplesse d'utilisation (clusters, réseaux),
  unique code source,
  architecture la plus répandue

Mémoire distribuée / mémoire partagée





(P: processeur, M: mémoire)

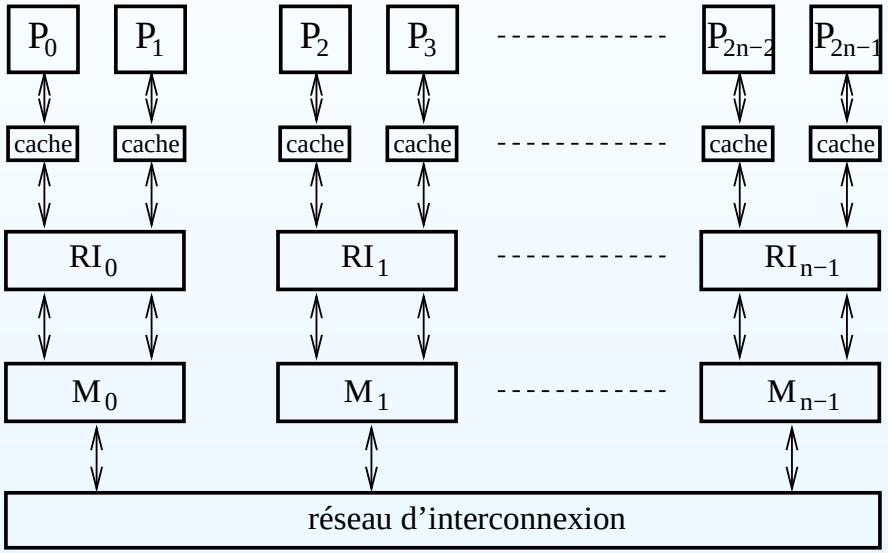
Avantage de la mémoire distribuée : accès rapide à la mémoire

Inconvénient : communications lentes avec les autres machines

Mémoire partagée : c'est l'inverse

Le matériel du CERMICS

Système hybride :



CLUMPS (cluster de SMP, Symmetric Multiprocessor)

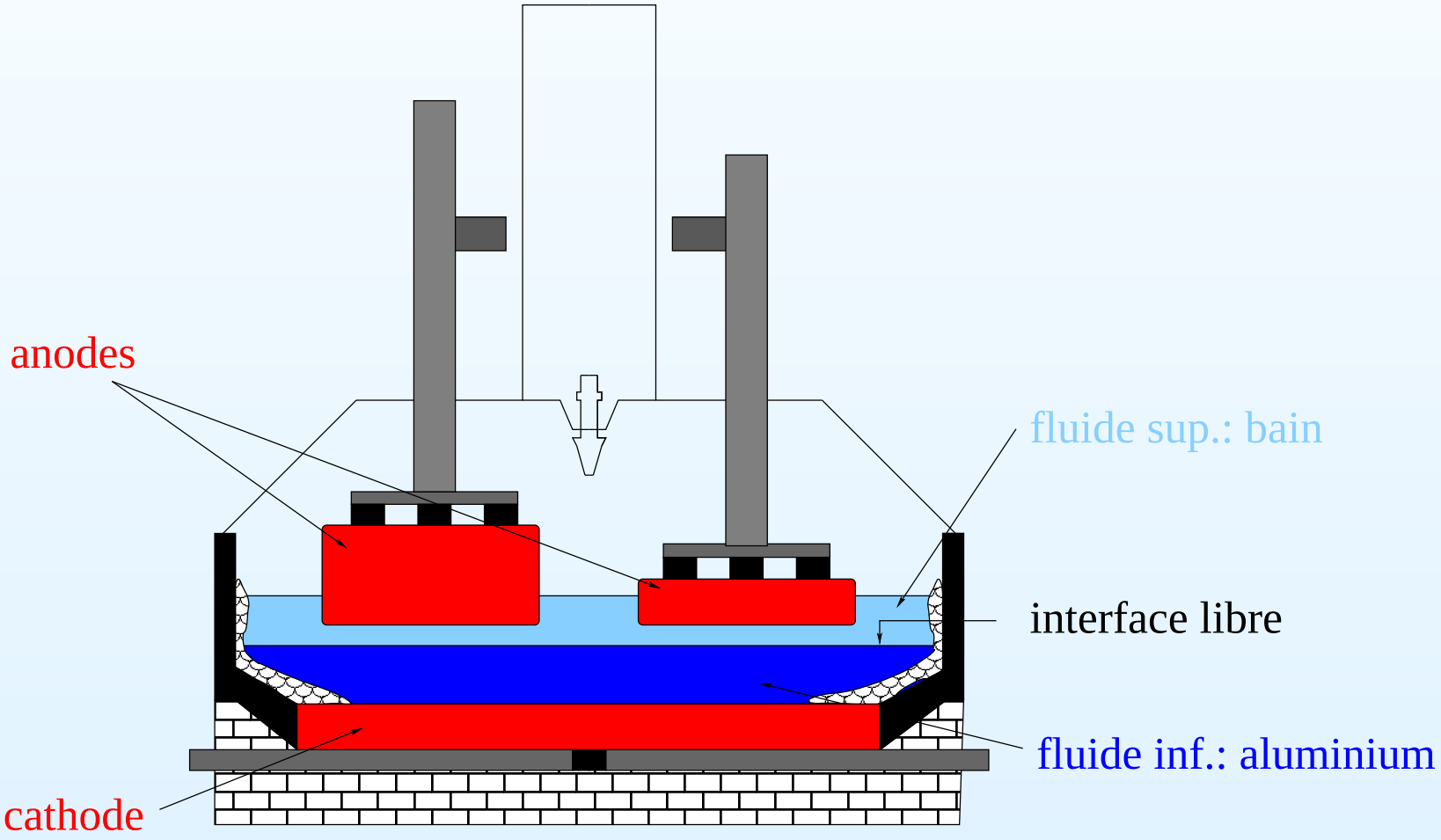
Le CLUMPS du CERMICS est actuellement constitué de 6 biprocesseurs

I - Le code *Mistral*

- Présentation
- Chargement du maillage
- Initialisation des éléments fi nis
- Construction du profi l creux de la matrice
- Construction du système linéaire
- Problèmes non linéaires et / ou d'évolution
- Phénomènes bifluides : formulation ALE
- Exemple

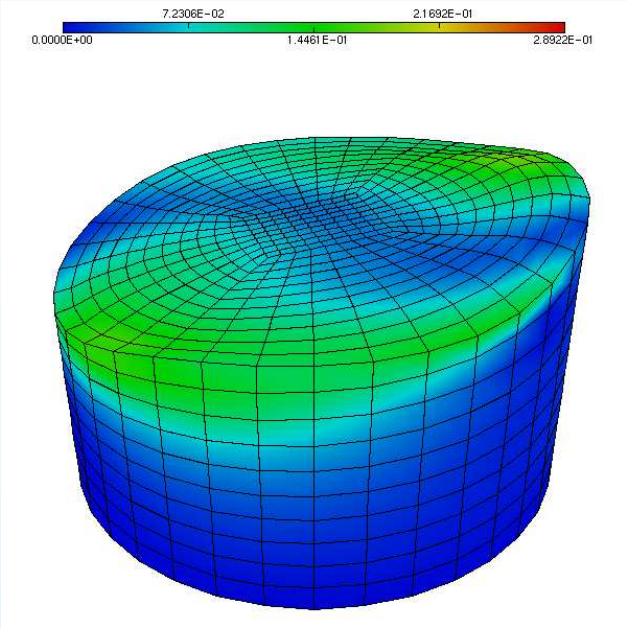
I.1 - Présentation

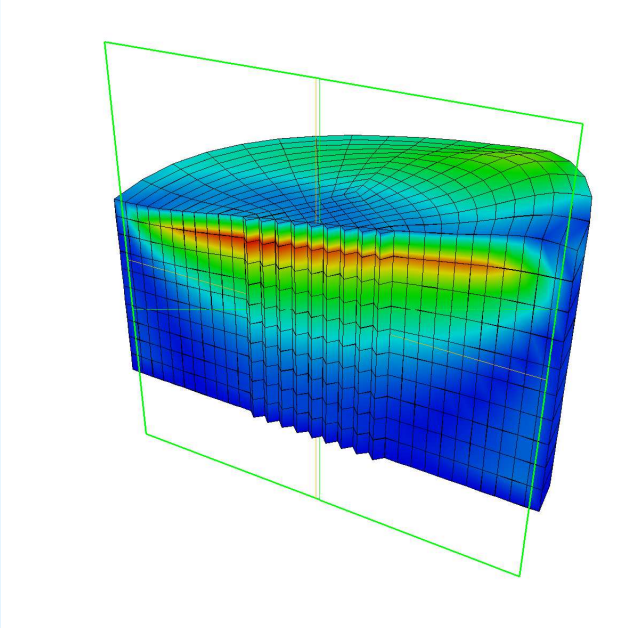
- Le phénomène physique :
  modélisation MHD d'une réaction d'électrolyse



I.1 - Présentation

- Auteurs : J.-F. Gerbeau (INRIA) et T. Lelièvre (CERMICS)
- Langage de programmation : C++
- Éléments fi nis Q^1 stabilisés
- 2 fluides séparés par une interface mobile $\longrightarrow$ formulation ALE

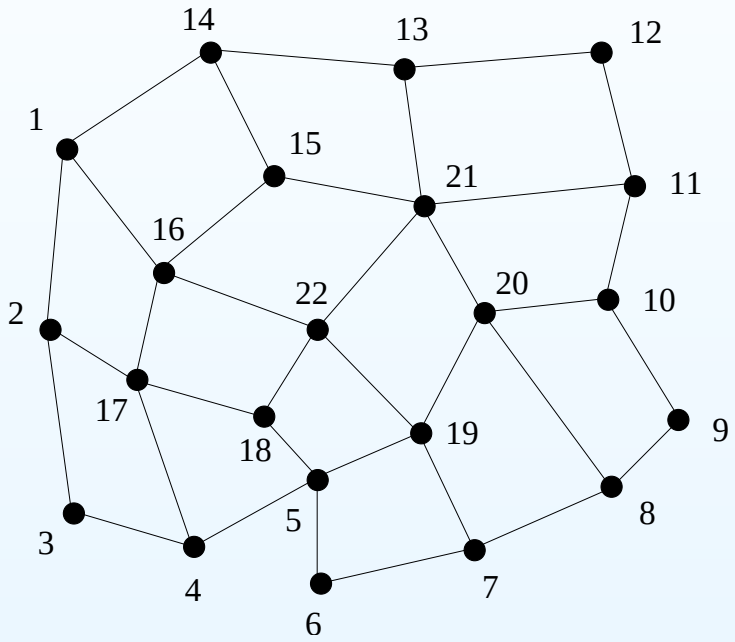




Le cas test du “rolling”

I.2 - Chargement du maillage

- Fichier d'entrée *Fidap Neutral File* $\longrightarrow$ objet *Geometrie* en mémoire



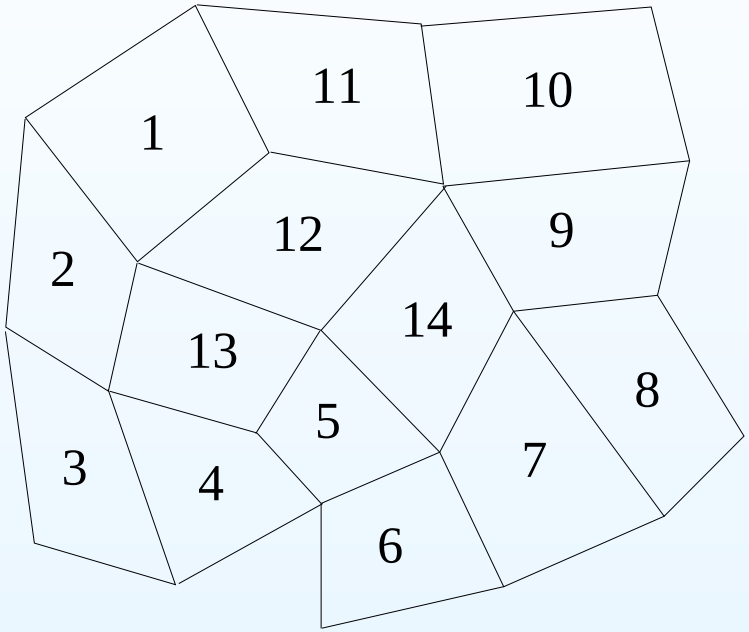
Liste de nœuds

Structure de données : *Geometrie* $= \bigcup$ *Figure* $= \bigcup\bigcup$ *Elemgeo*

Figure, *Elemgeo* : surfaciques ou volumiques

I.2 - Chargement du maillage

- Fichier d'entrée *Fidap Neutral File* $\longrightarrow$ objet *Geometrie* en mémoire



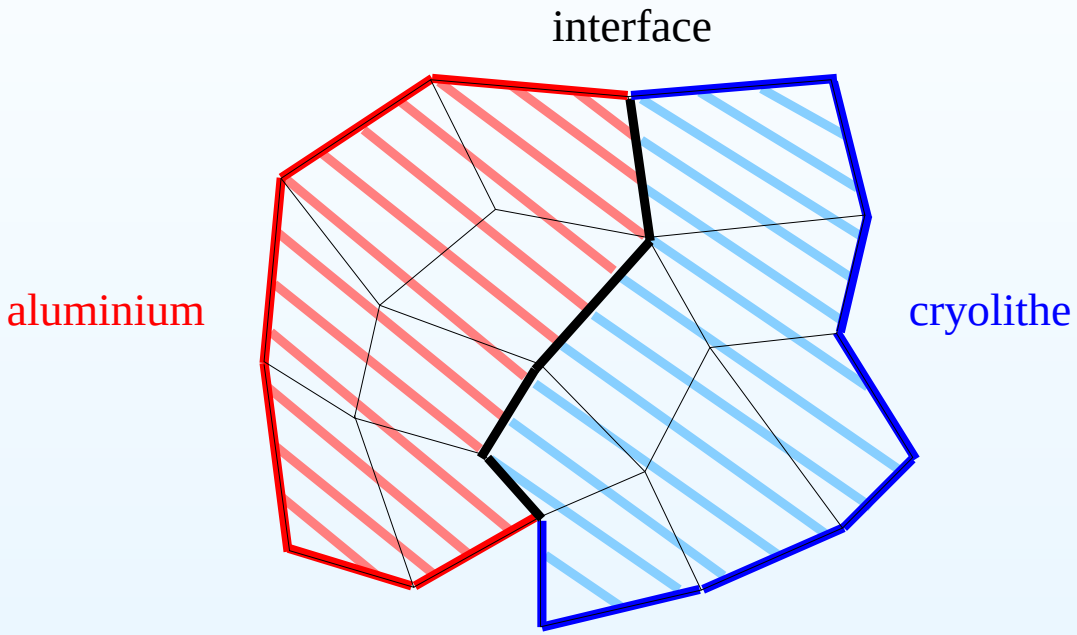
Liste de nœuds

Structure de données : *Geometrie* $= \bigcup$ *Figure* $= \bigcup\bigcup$ *Elemgeo*

Figure, *Elemgeo* : surfaciques ou volumiques

I.2 - Chargement du maillage

- Fichier d'entrée *Fidap Neutral File* $\longrightarrow$ objet *Geometrie* en mémoire



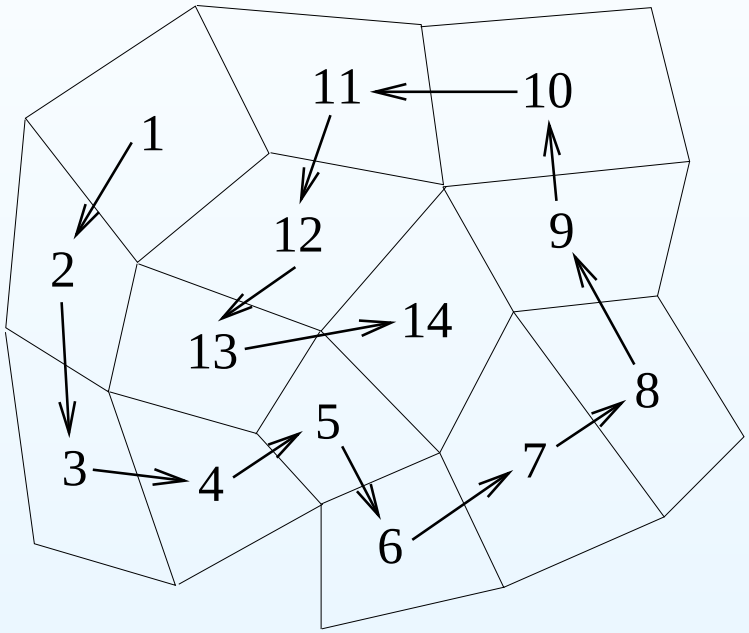
Liste de nœuds

Structure de données : *Geometrie* $= \bigcup$ *Figure* $= \bigcup\bigcup$ *Elemgeo*

Figure, *Elemgeo* : surfaciques ou volumiques

I.3 - Initialisation des éléments fi nis

Principe : parcours des éléments géométriques :



Pour chacun d'entre eux : création d'un élément fi ni associé

- Affectation de 2 jeux de fonctions de formes

$\Longrightarrow$ 2 types d'interpolation (pour les problèmes mixtes)

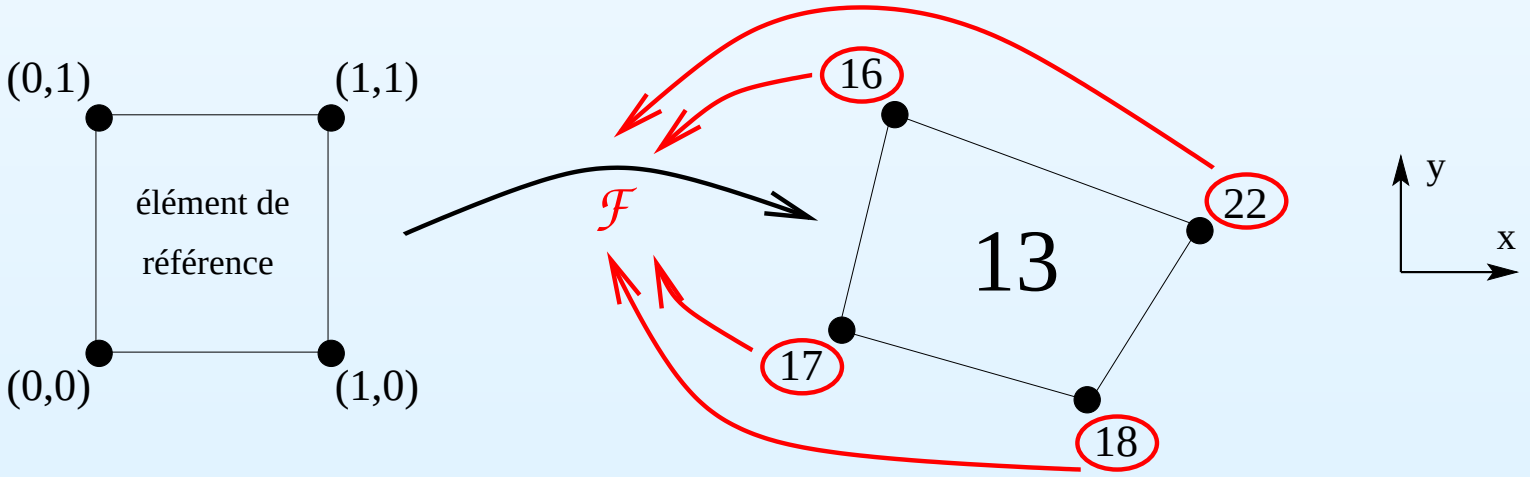
I.3 - Initialisation des éléments fi nis (exemple)

Tout élément fi ni(objet *Elemfi*) est muni de l'ensemble des opérateurs variationnels discrets implémentés à ce jour.

Par exemple, dans le cas du laplacien, il peut à ce stade calculer sa **matrice élémentaire** :

$$\mathbb{A} = (\mathbb{A}_{ij})_{1\leq i,j\leq 4} = \Big(\int_K \nabla\tau_i \, \nabla\tau_j \Big)_{1\leq i,j\leq 4} .$$

grâce aux coordonnées de ses sommets :



I.4 - Construction du profil creux de la matrice

Principe de *Mistral* : implémentation classique des méthodes d'éléments finis = calcul élémentaire + assemblage + résolution

- Profil de matrice creuse à déterminer (format *SparseLib++*)

  ($\Rightarrow$ économies de mémoire)

1) Numérotation des degrés de liberté

   - 1 ddl = combinaison entre 1 nœud et 1 inconnue scalaire
   - Conditions de Dirichlet imposées par élimination

     $\Rightarrow$ suppression de certains degrés de liberté

2) Établissement de la connectivité du système linéaire

I.4 - Construction du profil creux de la matrice

Élimination des conditions de Dirichlet : le système linéaire :

$$\left[\begin{array}{c|c} I & 0 \\ \hline A_D & A \end{array}\right] \left[\begin{array}{c} \\ X \\ \\ \end{array}\right] \left[\begin{array}{c} X_D \\ - \\ \\ Y \end{array}\right]$$

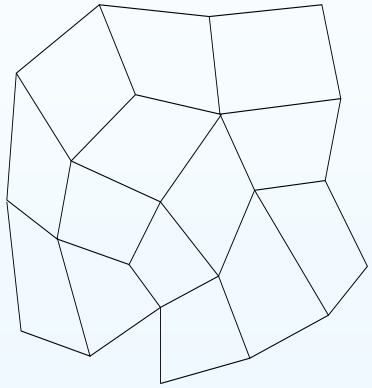
devient :

$$\left[\begin{array}{c} \\ A \\ \\ \end{array}\right] \left[\begin{array}{c} \\ X_{\overline{D}} \\ \\ \end{array}\right] = \left[\begin{array}{c} \\ Y - A_D\, X_D \\ \\ \end{array}\right] .$$

(D = ensemble des degrés éliminés)

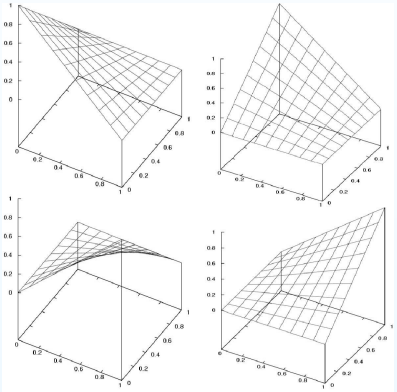
Résumé

Les 3 étapes précédentes = INITIALISATION :



Geometrie, Figure, Elemgeo

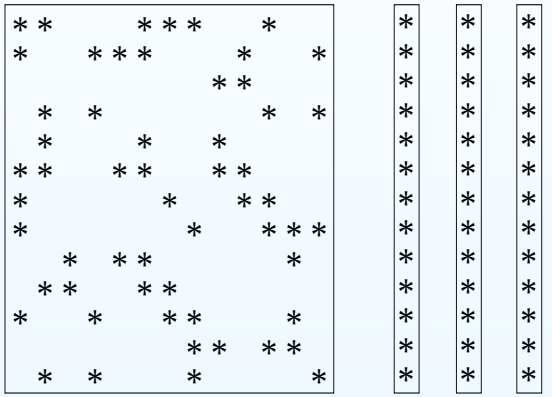
+



Interpol, Fctform

dirichlet()

def()



Elemfi, DegLib, Matrice, Vecteur

Cette phase d'initialisation, qui traite donc 8 types d'objets, se fait dans le constructeur de l'objet modélisant le problème, en lisant un fi chier d'entréedans lequel sont contenues les informations nécessaires à cette construction :

- nom du fi chier de maillage
- interpolation
- paramètres numériques, etc...

I.5 - Construction du système linéaire (assemblage)

Sur chaque élément fi ni :

- calculer la matrice élémentaire
- $\forall$ combinaison “équation + nœud sur l’élément”
  - si c’est un DDL (ligne)
    - $\forall$ combinaison “inconnue + nœud sur l’élément”
      - si c’est un DDL (colonne)
        - assembler action sur $X_{\overline{D}}$ dans A
      - sinon
        - assembler action sur X_D dans Y
    - assembler source dans Y

I.6 - Résolution

Librairie $SparseLib++$: préconditionneurs

- ID
- DIAG
- ILU(k)
- ...

Librairie $IML++$: solveurs de type Krylov

- gradient conjugué
- GMRES
- BiCGSstab
- ...

Exemples

- problème de Stokes : ID + gradient conjugué
- problème de MHD : ILU(0) + GMRES

I.7 - Problèmes non linéaires et / ou instationnaires

Exemple : Navier-Stokes stationnaire

$$u \cdot \nabla u - \nu \, \Delta u + \nabla p = f \, ,$$
$$\text{div}\, u = 0 \, ,$$

- Algorithme de point fi xe :

$$u^n \cdot \nabla u^{n+1} - \nu \, \Delta u^{n+1} + \nabla p^{n+1} = f \, ,$$
$$\text{div}\, u^{n+1} = 0 \, ,$$

ou éventuellement de type Newton-Raphson

Assemblage et résolution sont englobés dans une boucle gérée par la fonction $avance()$ de l'objet $Nonlin$ ($u^n \leftarrow u^{n+1}$, $A \leftarrow 0$).

I.7 - Problèmes non linéaires et / ou instationnaires

Exemple : Navier-Stokes instationnaire

$$\partial_t u + u \cdot \nabla u - \nu \, \Delta u + \nabla p = f \, ,$$
$$\text{div}\, u = 0 \, ,$$

- Algorithme de point fi xe relaxé:

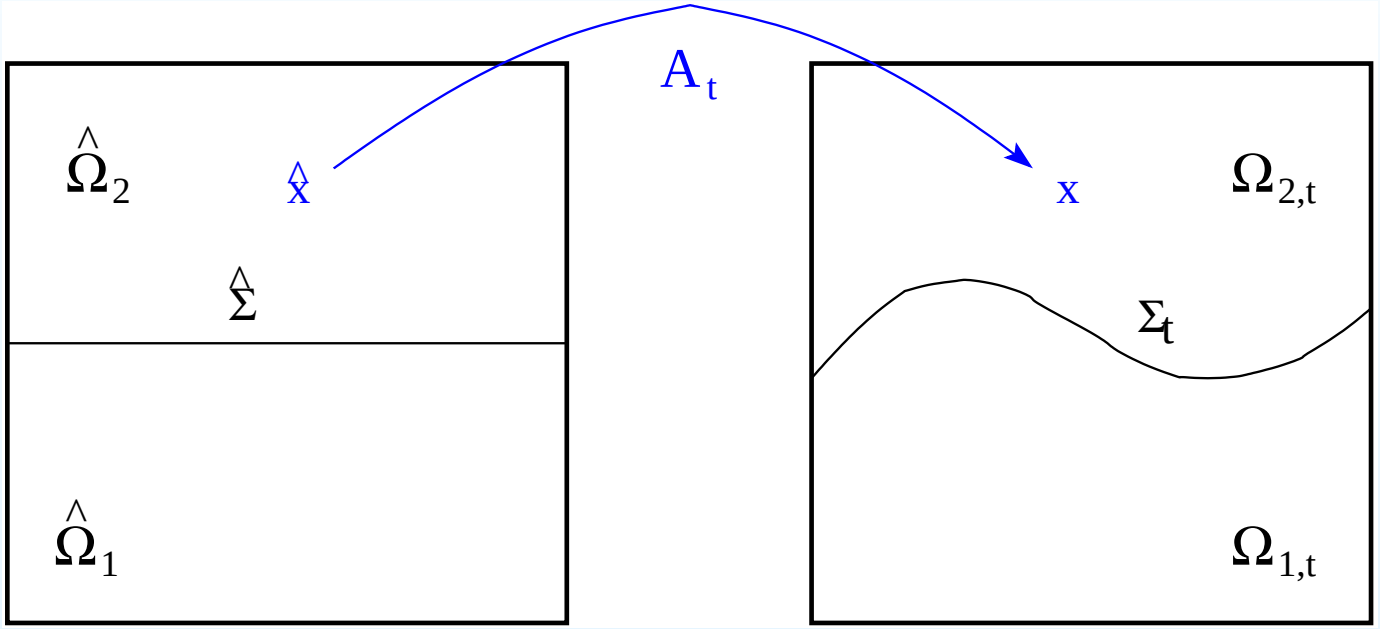
$$\frac{u^{n+1} - u^n}{\delta t} + u^n \cdot \nabla u^{n+1} - \nu \, \Delta u^{n+1} + \nabla p^{n+1} = f \, ,$$
$$\text{div}\, u^{n+1} = 0 \, ,$$

(schéma d'Euler semi-implicite)

Assemblage et résolution sont englobés dans une boucle gérée par la fonction $avance()$ de l'objet $Transit$ ($u^n \leftarrow u^{n+1}$, $A \leftarrow 0$).

I.8 - Modèles bifluides : formulation ALE

- même principe que pour les problèmes transitoires
- déplacement du maillage à chaque pas de temps
- 2 assemblages : avant et après déplacement du maillage



Vitesse du domaine : $w(x,t) = \partial_t A_t(\hat{x})$, où $\hat{x} = A_t^{-1}(x)$

I.8 - Modèles bifluides : formulation ALE

1) Résolution du déplacement du maillage

- les frontières du domaine restent fixes $\Rightarrow w \cdot \nu = 0$ sur $\partial\Omega$
- l'interface sépare les 2 fluides $\Rightarrow w \cdot \nu = u \cdot \nu$ sur Σ

(ν = normale sortante)

Ces conditions ne déterminent pas complètement w.

I.8 - Modèles bifluides : formulation ALE

1) Résolution du déplacement du maillage

- les frontières du domaine restent fixes $\Rightarrow w \cdot \nu = 0$ sur $\partial\Omega$
- l'interface sépare les 2 fluides $\Rightarrow w \cdot \nu = u \cdot \nu$ sur Σ

(ν = normale sortante)

Ces conditions ne déterminent pas complètement w.
Alors on cherche $w = (0, 0, w_z)$, tel que :

$$\begin{cases} -\Delta w_z = 0 & \text{sur } \Omega \\ w_z = u \cdot \nu / \nu_z & \text{sur } \Sigma \\ \partial_\nu w_z = 0 & \text{sur } \partial\Omega \end{cases}$$

Un système linéaire indépendant du problème principal est implémenté pour traiter ce problème

I.8 - Modèles bifluides : formulation ALE

2) Prise en compte de la contrainte de masse : $\int_\Sigma u \cdot \nu = 0$

$$\Rightarrow \int_{\Omega_1^n} \text{div}\, u_h \, dx = 0 \quad \text{(car} \quad \int_{\partial\Omega_1^n} u_h \cdot \nu_h \, dx = 0 \quad \text{).}$$

I.8 - Modèles bifluides : formulation ALE

2) Prise en compte de la contrainte de masse : $\int_\Sigma u \cdot \nu = 0$

$$\Rightarrow \int_{\Omega_1^n} \text{div}\, u_h \, dx = 0 \quad \text{(car} \quad \int_{\partial\Omega_1^n} u_h \cdot \nu_h \, dx = 0 \quad \text{).}$$

Non vérifi ée pour les éléments fi nis continus en pression !

2) Prise en compte de la contrainte de masse : $\int_\Sigma u \cdot \nu = 0$

$$\Rightarrow \int_{\Omega_1^n} \text{div}\, u_h \, dx = 0 \quad \text{(car} \quad \int_{\partial\Omega_1^n} u_h \cdot \nu_h \, dx = 0 \quad \text{).}$$

Non vérifi ée pour les éléments fi nis continus en pression !

$\Rightarrow$ multiplicateur de Lagrange (λ) associé à cette contrainte :

$\Rightarrow$ système :

$$\left[\begin{array}{c|c} A & \Phi^T \\ \hline \Phi & 0 \end{array}\right] \left[\begin{array}{c} X_{\overline{D}} \\ \hline \lambda \end{array}\right] = \left[\begin{array}{c} Y - A_D X_D \\ \hline 0 \end{array}\right].$$

II - La librairie *Aztec*

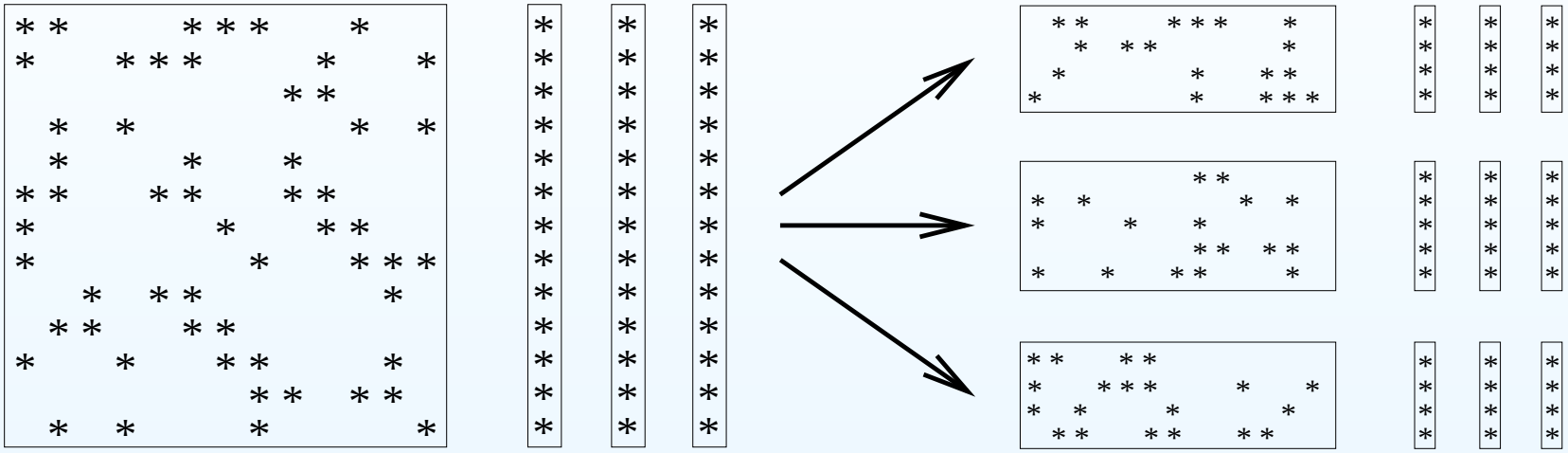
- Présentation
- Principe
- Exemple

II.1 - Présentation

- Surcouche de MPI $(Message\ Passing\ Interface)$
- Solveur parallèle de systèmes linéaires creux de grande taille
  - Méthodes de Krylov : CG, BiCGSstab, GMRES, ...
- Préconditionneur parallèle (Schwarz additif)
  - LU et ses variantes (ILU(k), ILUT, BILU, ...)
- Agrandissement et réducion de la matrice (“scaling”)
- Réutilisation d’un même préconditionnement sur plusieurs résolutions successives
- Format pour les sous-matrices : DMSR (Distributed MSR)

II.2 - Principe

1) Division du système en sous-matrices rectangulaires (lignes) :



Les indices des lignes affectées à chaque processeur sont dans le tableau $update$. Exemple :

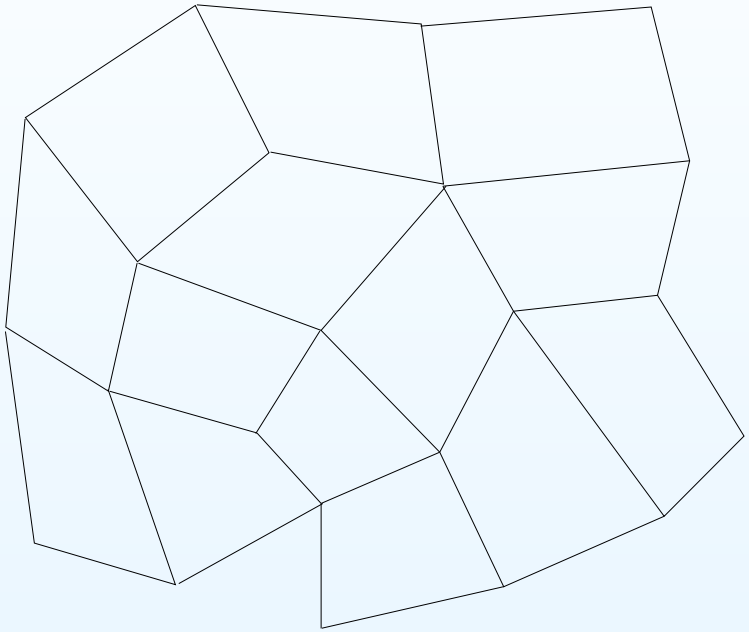
$update(0)$ = | 6 | 0 | 3 | 12 | ,

$update(1)$ = | 10 | 7 | 4 | 1 | 2 | ,

$update(2)$ = | 9 | 5 | 11 | 8 | .

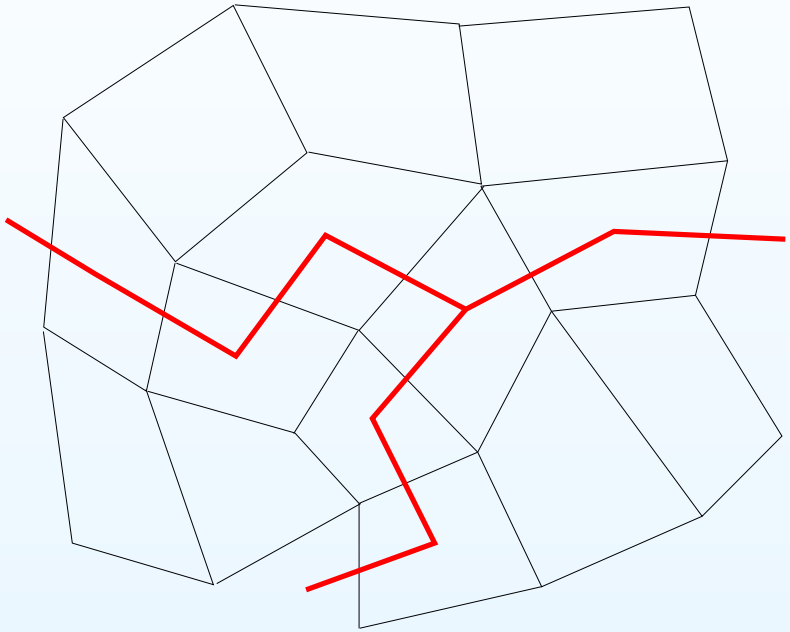
II.2 - Principe

2) Transformation des sous-matrices :



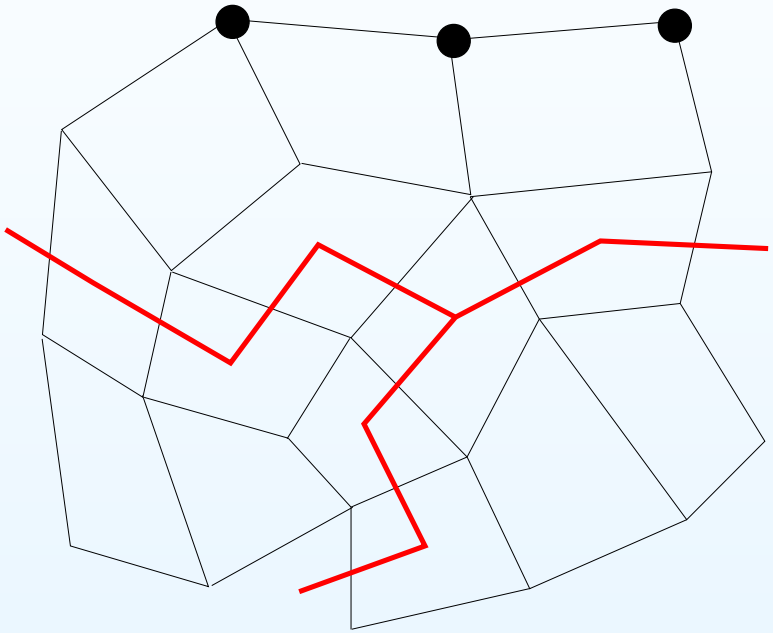
II.2 - Principe

2) Transformation des sous-matrices :



II.2 - Principe

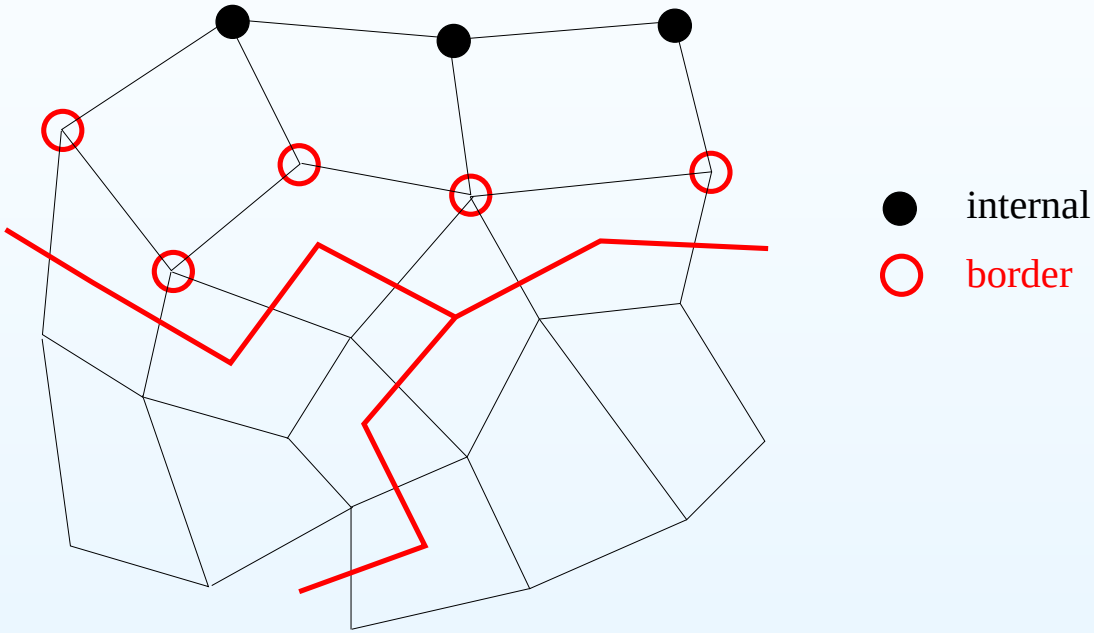
2) Transformation des sous-matrices :



internal

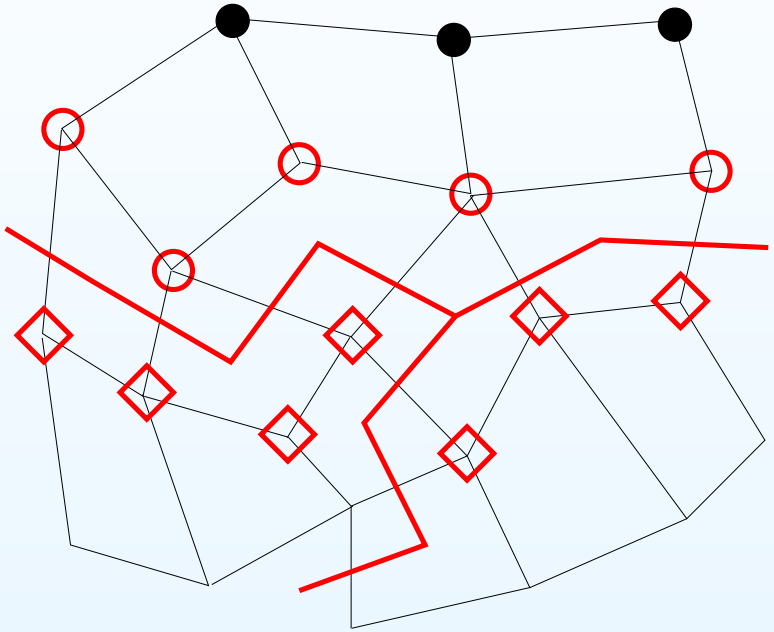
II.2 - Principe

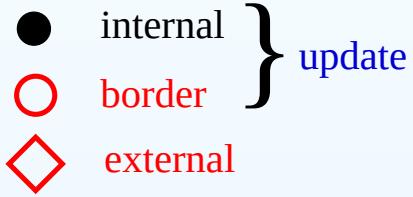
2) Transformation des sous-matrices :



II.2 - Principe

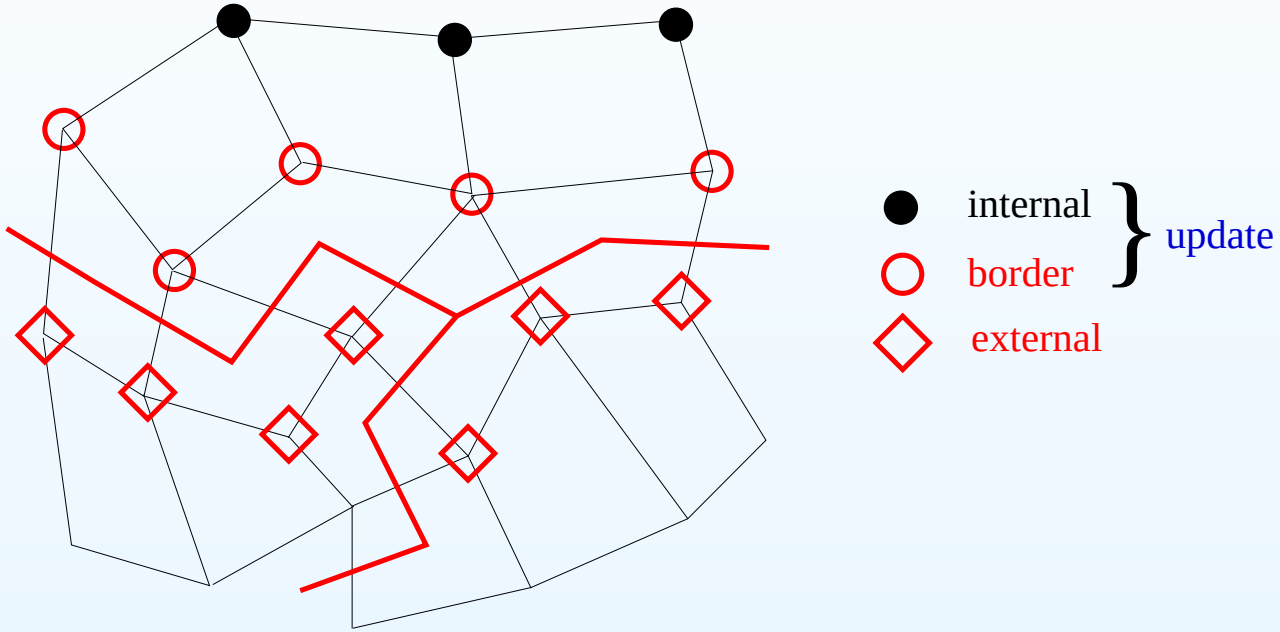
2) Transformation des sous-matrices :





II.2 - Principe

2) Transformation des sous-matrices :



$AZ\_transform()$ réindexe les sous-matrices dans l'ordre :

← *internal* →	← *border* →	← *external* →

et génère les tableaux de redirection $update\_index$ et $extern\_index$

II.3 - Exemple : problème de Poisson

$$\begin{matrix} -1 & 0 & 0 & 0 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & -1 \end{matrix}$$

- Instructions (principales) à écrire :

MPI_Init(...)

AZ_set_proc_config(...)

AZ_read_update(...)

create_matrix_row(update[i]) | $\forall \quad update[i]$

Si *update[i] = 0 : b[0]* $\leftarrow$ *1* |

Sinon : *b[update[i]]* $\leftarrow$ *0* |

AZ_transform(...)

AZ_reorder_vec(...)

AZ_solve(...) ($\leftarrow$ GMRES, ILU)

MPI_Finalize(...)

III - Parallèlisation du code $Mistral$

- Présentation
- Parallèlisation de la méthode itérative
- Parallèlisation de l'assemblage
- Formulation ALE
- Numérotation optimale des degrés de liberté
- Exemple récapitulatif
- Conclusion

III.1 - Parallélisation de la méthode itérative

- Construction du problème exemple d'$Aztec$ au format $SparseLib++$ (CSR, Compressed Sparse Row)
- Résolution avec $IML++$ (GMRES,ILU)
- Conversion CSR$\rightarrow$MSR + partitionnement
- Transformation + résolution par $Aztec$ (GMRES,ILU)

conversion partitionnement *AZ_transform()*

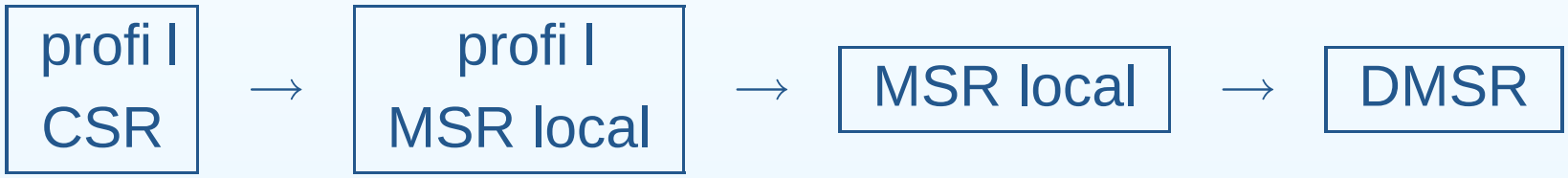
CSR $\rightarrow$ MSR global $\rightarrow$ MSR local $\rightarrow$ DMSR

$\Rightarrow$ Même résultat, même nombre d'itérations

III.2 - Parallélisation de l'assemblage

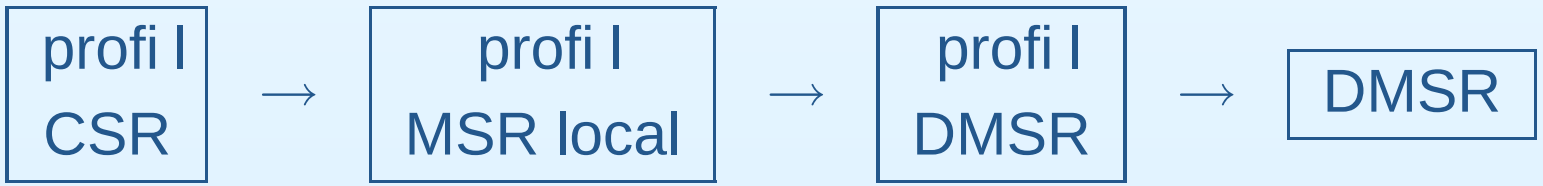
1) Problèmes stationnaires linéaires

conversion + partitionnement — assemblage — *AZ_transform()*



2) Problèmes à assemblages successifs

conversion + partitionnement — *AZ_transform()* — assemblage



III.3 - Formulation ALE

- Déplacement du maillage : non parallélisé (temps de calcul négligeable)

III.3 - Formulation ALE

▶ Déplacement du maillage : non parallélisé
(temps de calcul négligeable)

▶ Problème avec la contrainte de masse :
système à résoudre mal adapté au parallélisme :

$$\left[\begin{array}{c|c} A & \Phi^T \\ \hline \Phi & 0 \end{array}\right] \left[\begin{array}{c} X_{\overline{D}} \\ \hline \lambda \end{array}\right] = \left[\begin{array}{c} Y - A_D X_D \\ \hline 0 \end{array}\right].$$

III.3 - Formulation ALE

► Déplacement du maillage : non parallélisé
(temps de calcul négligeable)

► Problème avec la contrainte de masse :
système à résoudre mal adapté au parallélisme :

$$\left[\begin{array}{c|c} A & \Phi^T \\ \hline \Phi & 0 \end{array}\right] \left[\begin{array}{c} X_{\overline{D}} \\ \hline \lambda \end{array}\right] = \left[\begin{array}{c} Y - A_D X_D \\ \hline 0 \end{array}\right].$$

► 2 solutions :

1) Élimination de $X_{\overline{D}}$ par complément de Schur

2) Dédoublement des degrés de liberté en pression sur l'interface

III.3 - Formulation ALE

1) Résolution par complément de Schur

$$\left[\begin{array}{c|c} A & \Phi^T \\ \hline \Phi & 0 \end{array}\right] \left[\begin{array}{c} X_{\overline{D}} \\ \hline \lambda \end{array}\right] = \left[\begin{array}{c} Y - A_D X_D \\ \hline 0 \end{array}\right]$$

$$\Downarrow$$

$$X_{\overline{D}} = A^{-1}(Y - A_D X_D) - \lambda\, A^{-1}\Phi^T\,,$$

$$\Downarrow$$

$$\lambda = \frac{\Phi A^{-1}(Y - A_D X_D)}{\Phi A^{-1}\Phi^T}$$

III.3 - Formulation ALE

1) Résolution par complément de Schur

La procédure s'écrit, après assemblages de A et Φ :

1) Résoudre en parallèle le système $AX_1 = Y - A_D X_D$,
2) Résoudre en parallèle le système $AX_2 = \Phi^T$,
3) Calculer ΦX_1 et ΦX_2,
4) Diviser ΦX_1 par ΦX_2 pour obtenir λ,
5) Calculer $X_{\overline{D}}$ en retranchant λX_2 de X_1.

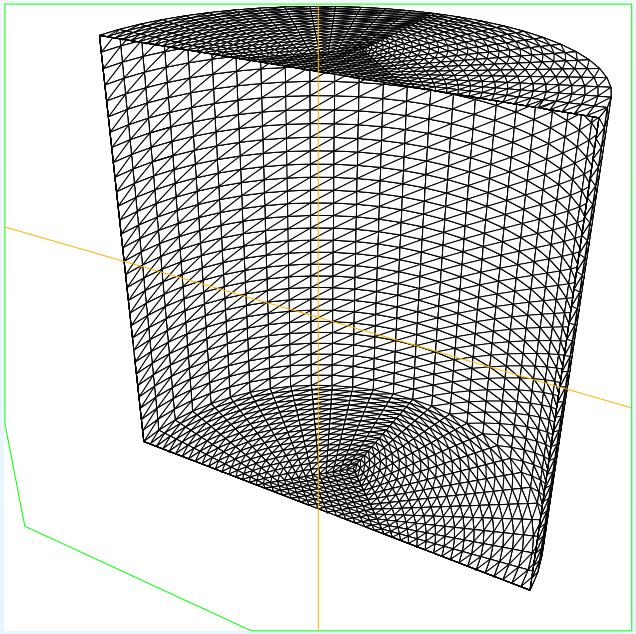
- 2 inversions de $A \Rightarrow$ temps de calcul * 1.5 !

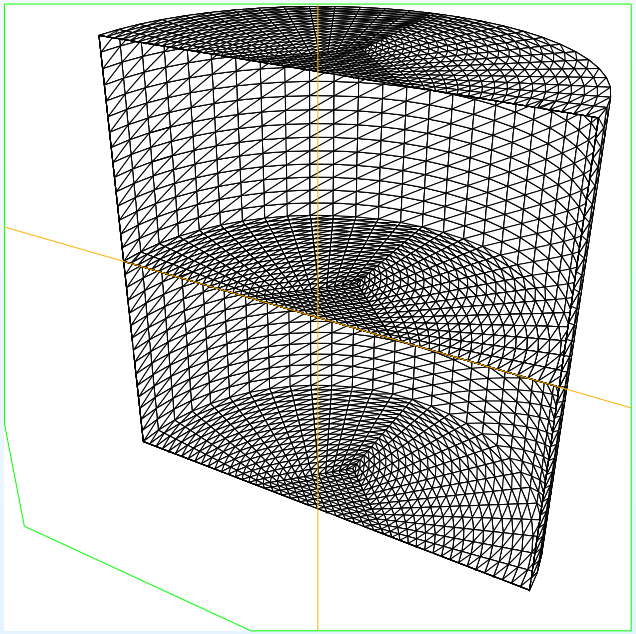
- on cherche à se passer de multiplicateur de Lagrange

III.3 - Formulation ALE

2) Dédoublement des nœuds du maillage sur l'interface

$\Longrightarrow$ Pression Q^1 dans chaque fluide avec possibilité de discontinuité à l'interface





Éléments volumiques visualisés avec $Medit^{®}$

III.3 - Formulation ALE

2) Dédoublement des nœuds du maillage sur l'interface

- Imposition d'une vitesse et d'un champ magnétique continus

  = "fusion" des degrés de liberté concernés à l'aide d'une variante de la fonction de défi nition de conditions périodiques $(write\_periodic\_file())$

- Problème de déplacement du maillage : imposition des mêmes conditions de Dirichlet sur les 2 "interfaces"

III.4 - Numérotation optimale des degrés de liberté

Parallèlisation d'un système linéaire = 2 freins à la minimisation du temps de calcul :

- ▶ communications trop importantes
- ▶ perte de la qualité du préconditionnement

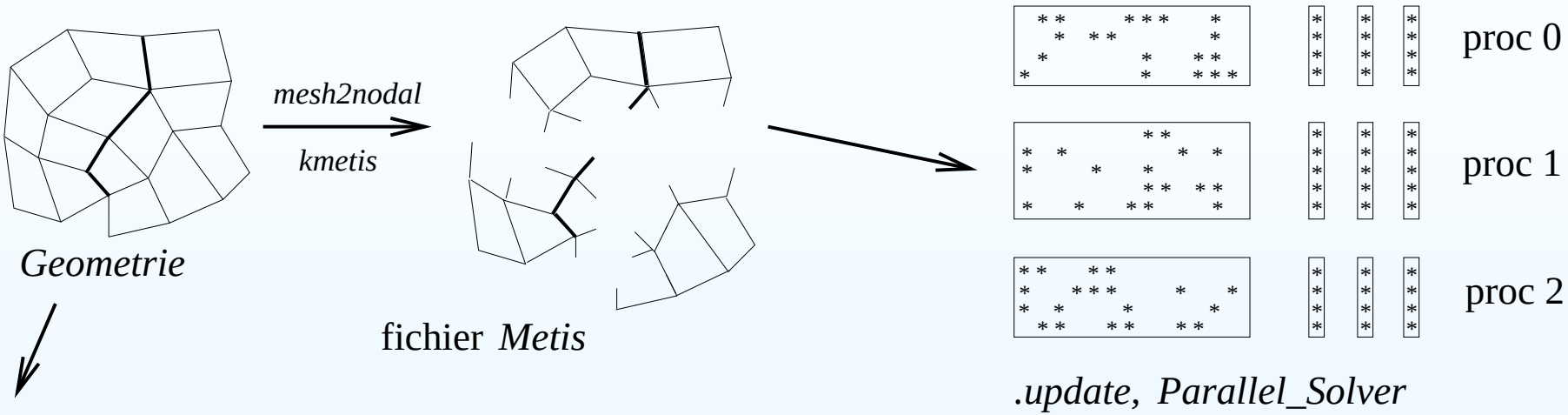
$\Rightarrow$ Partitionnement optimal du maillage avec $Metis^{®}$
= minimisation des "cassures" d'arêtes

Procédure :

1) conversion du maillage en graphe
2) partitionnement du graphe en sous-domaines de nœuds
3) Transposition en sous-domaines de degrés de liberté
   $\Rightarrow$ tableaux $update$

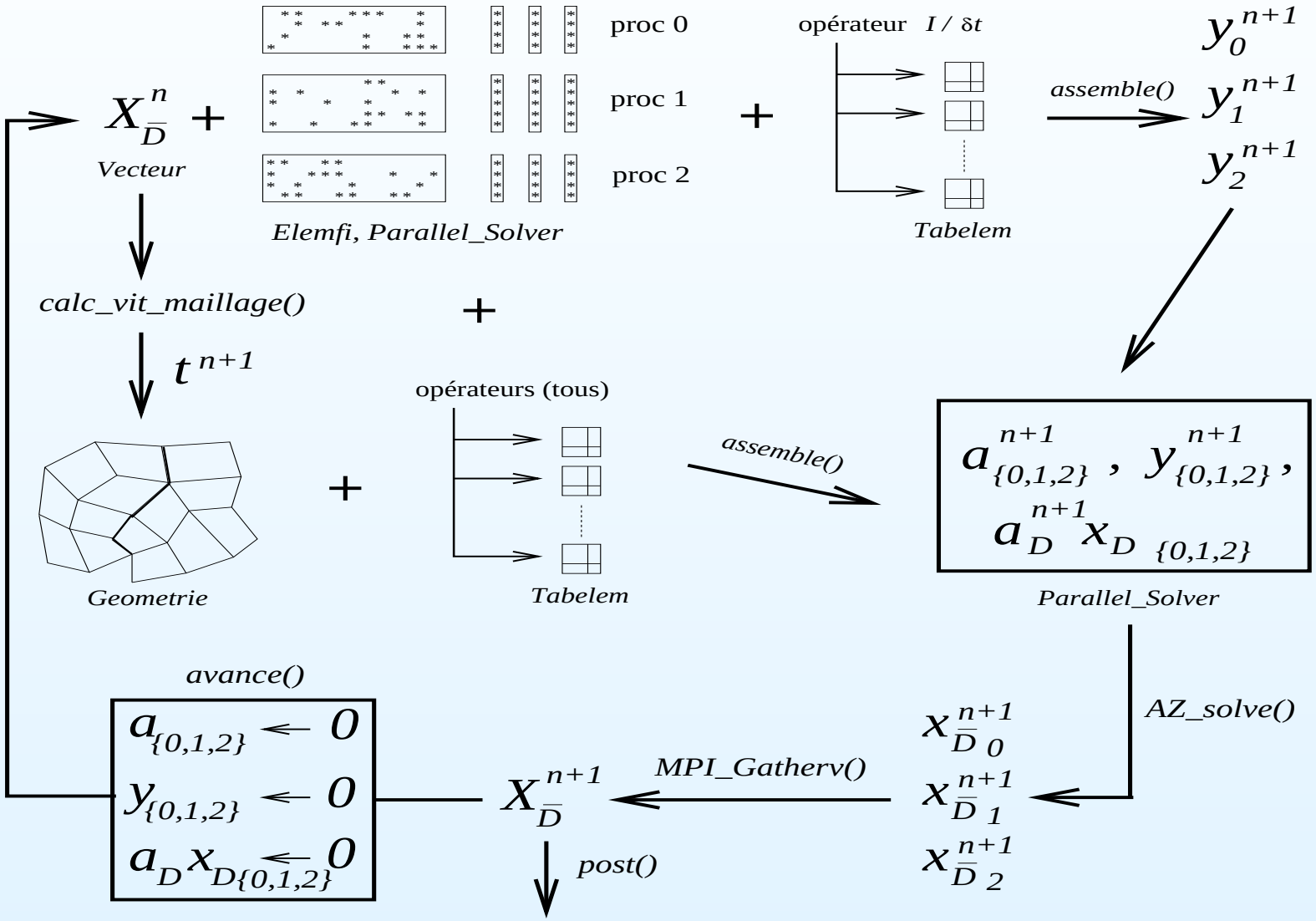
$\Rightarrow$ 20% d'éléments en moins dans $external$
$\Rightarrow$ 20% d'itérations en moins dans GMRES prec. ILU

III.5 - Exemple récapitulatif



Parallélisation de l'initialisation

assemblage, résolution et post-traitement



III.5 - Mesures

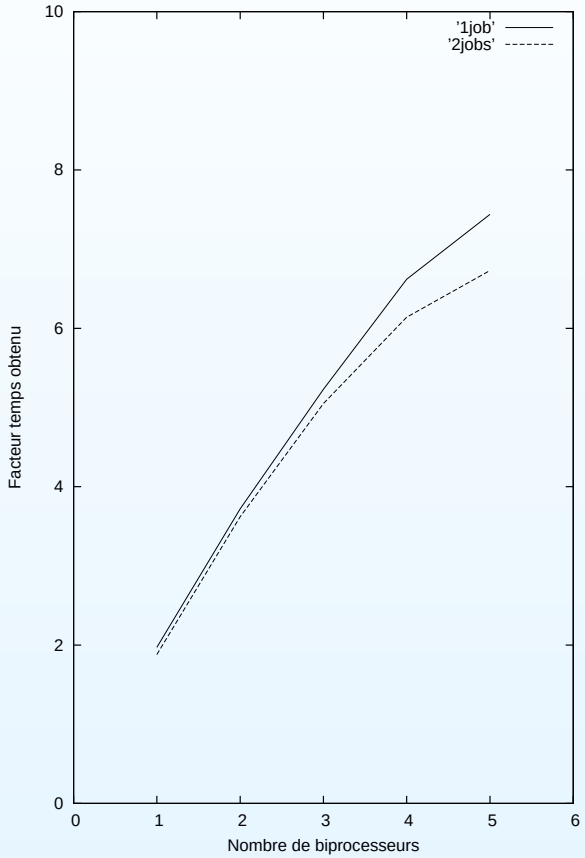
Programme séquentiel :

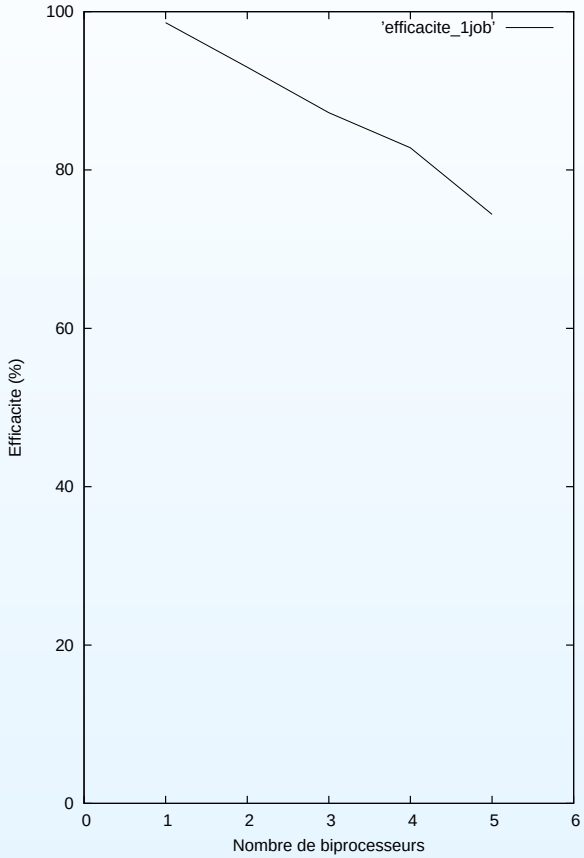
- initial : 7 h 23’
- avec dédoublement de l’interface : 7 h 04’

Programme parallèle :

Nombre de biprocesseurs	1 job par machine	2 jobs par machine
1	3 h 35’ (215’)	3 h 46’ (226’)
2	1 h 54’ (114’)	1 h 57’ (117’)
3	1 h 21’ (81’)	1 h 24’ (84’)
4	1 h 04’ (64’)	1 h 09’ (69’)
5	0 h 57’ (57’)	1 h 03’ (63’)

Conclusion





Décomposition de domaine ? Non encore connu en MHD...