Calcul parallèle haute performance dans CAST3M

Jean-Yves COGNARD

Pierre VERPEAUX

1 – Problématique

2 – Résolution parallèle de problèmes non linéaires d'évolution

3 – Environnement de programmation parallèle

4 – Résolution des problèmes linéaires de grande taille

5 – Problèmes non linéaires de grande taille

6 – Conclusions

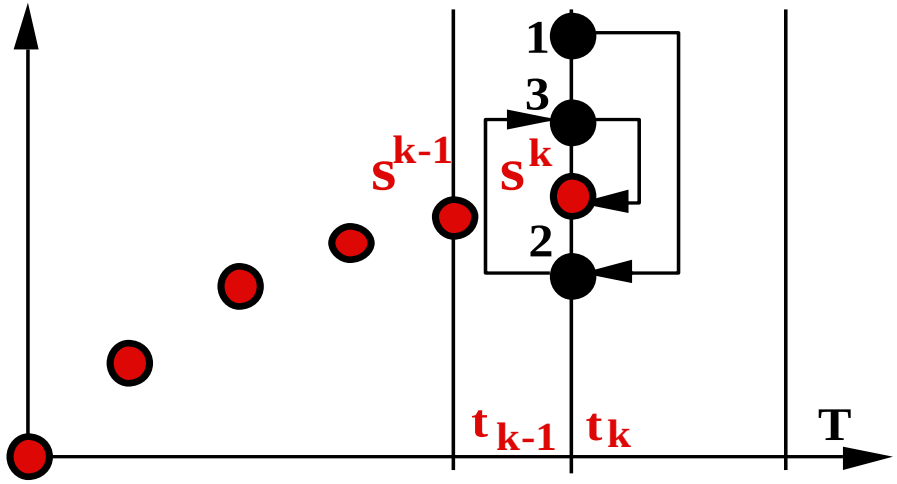
Développement d'applications parallèles

- **Ordinateurs puissants Architecture parallèle (INTEL, AMD ...)**
  - Développements complexes
  - Performance / Portabilité
  - Différentes architectures : Langage de programmation ?
  - Utiliser les codes existants
- **Développement d'applications parallèles**
  - Algorithmes parallèles "Mécanique"
    - Bonnes performances & Bon équilibrage des charges
  - Environnement de programmation parallèle
    - Système indépendant du type d'ordinateur
    - Assurer la cohérence des données
    - Compatibilité des versions séquentielle et parallèle
- **Le but est de résoudre « rapidement » des problèmes de grande taille**
  - Large classe de problèmes non linéaire en quasi-statique

Problème à résoudre

- **Le problème à résoudre est non linéaire et d'évolution**
  - **Découpage de l'intervalle de temps en incréments**

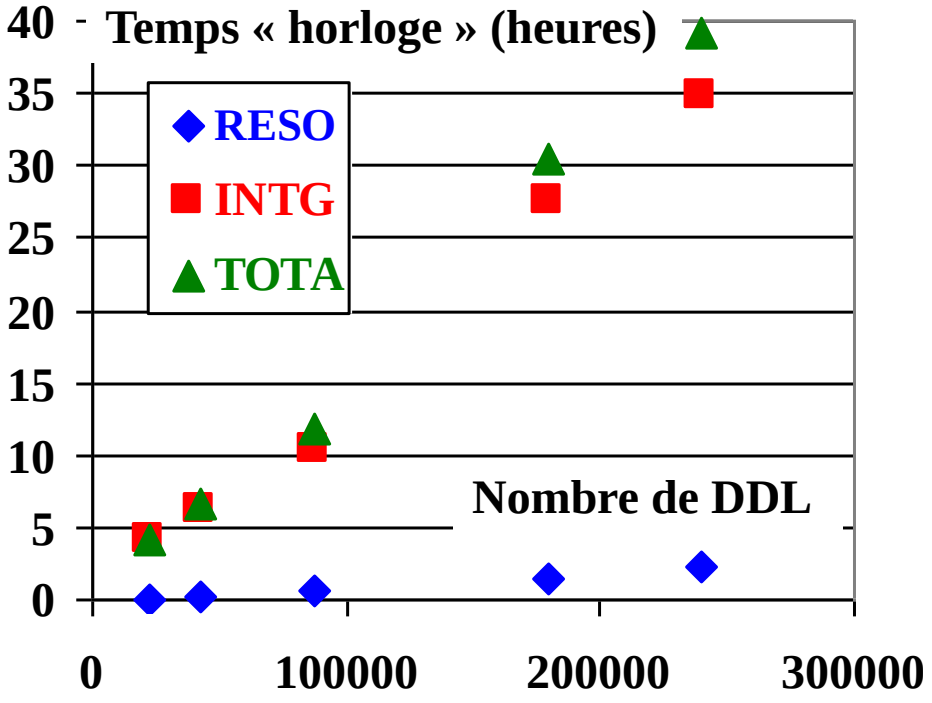
- **Problème non linéaire et stationnaire par incrément**
  - **Algorithme de type NEWTON**
    - **Procédure à deux étapes**
  - **Etape 1: Intégration de la loi de comportement**
    - **Problème non-linéaire local**
      (point d'intégration)
  - **Etape 2: Correction en déplacement**
    - **Problème linéaire global**
      (Ki δqi = Ri)
  - **Test de convergence**



Cout des simulations non linéaires

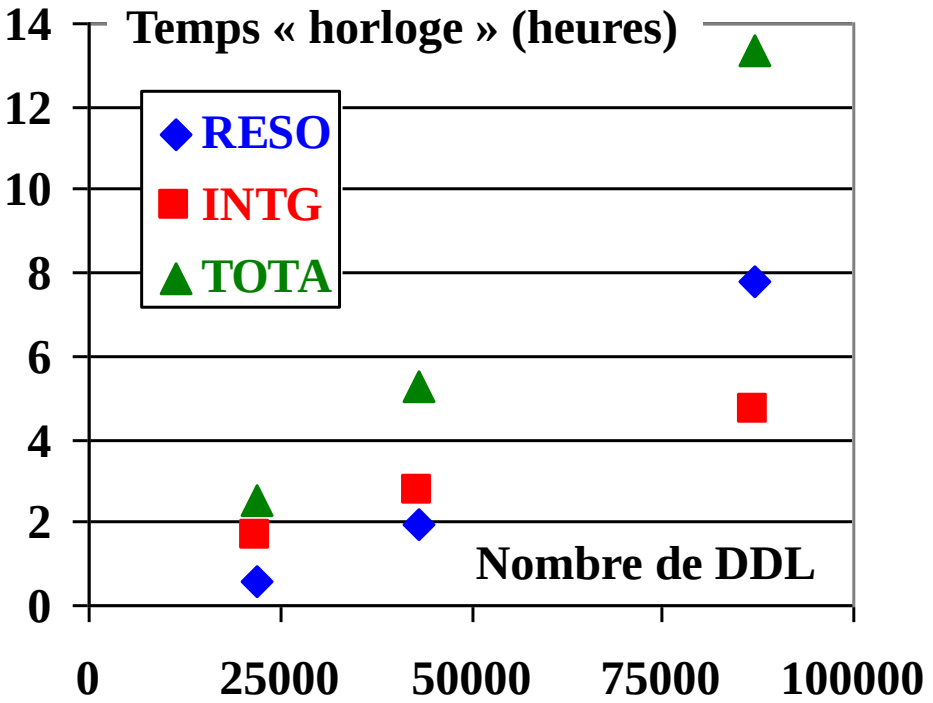
Non-linéarités matériaux

- 1 cycle de chargement
- 50 incréments
- 280 itérations (Newton)
- 1 factorisation



Non-linéarités matériaux & géométriques

- Chargement monotone
- 80 incréments
- 270 itérations (Newton)
- 80 factorisations



Résolution des problèmes globaux linéaires & Intégration de la loi de comportement

Résolution parallèle de problèmes non linéaires d'évolution

- **Approche Mécanique : 2 principaux points**
  - **Résolution des problèmes linéaires globaux**
    - **Techniques de type décomposition de domaine (choix?)**
    - **Algorithmes de type Quasi-Newton pour réduire le coût numérique**

      Utilisation des déplacements précédents → une meilleure solution

      déplacements vérifiant « bien » les conditions cinématiques

      Choix : résolution directe du problème condensé

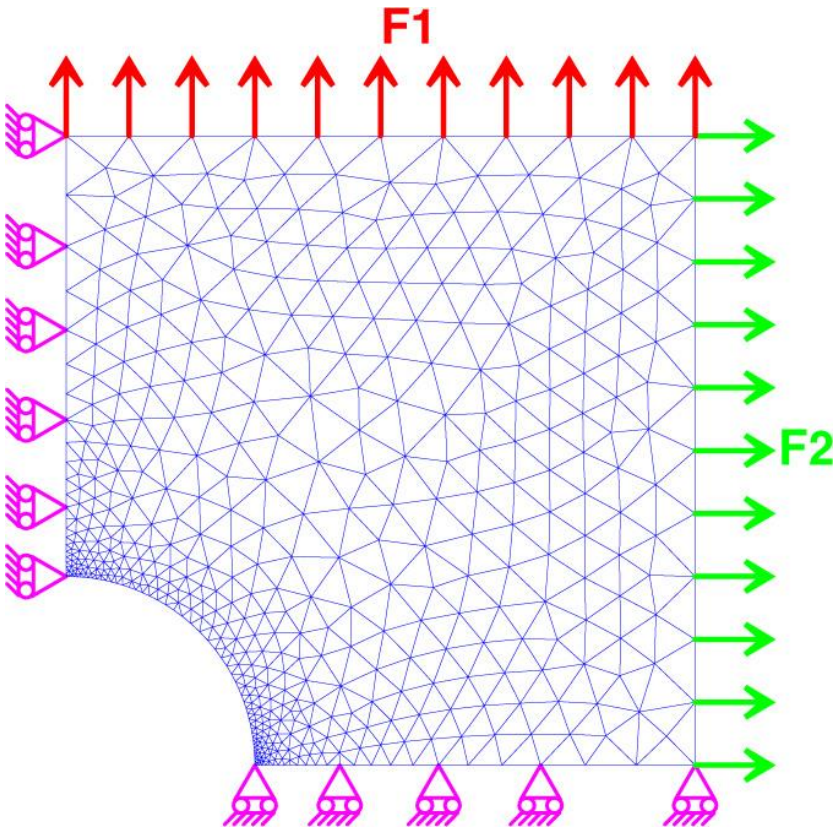
  - **Intégration de la relation de comportement (problème local non linéaire)**
    - **Bon équilibrage des calculs**
    - **Limiter les rééquilibrages dynamiques**
    - **Limiter les communications**

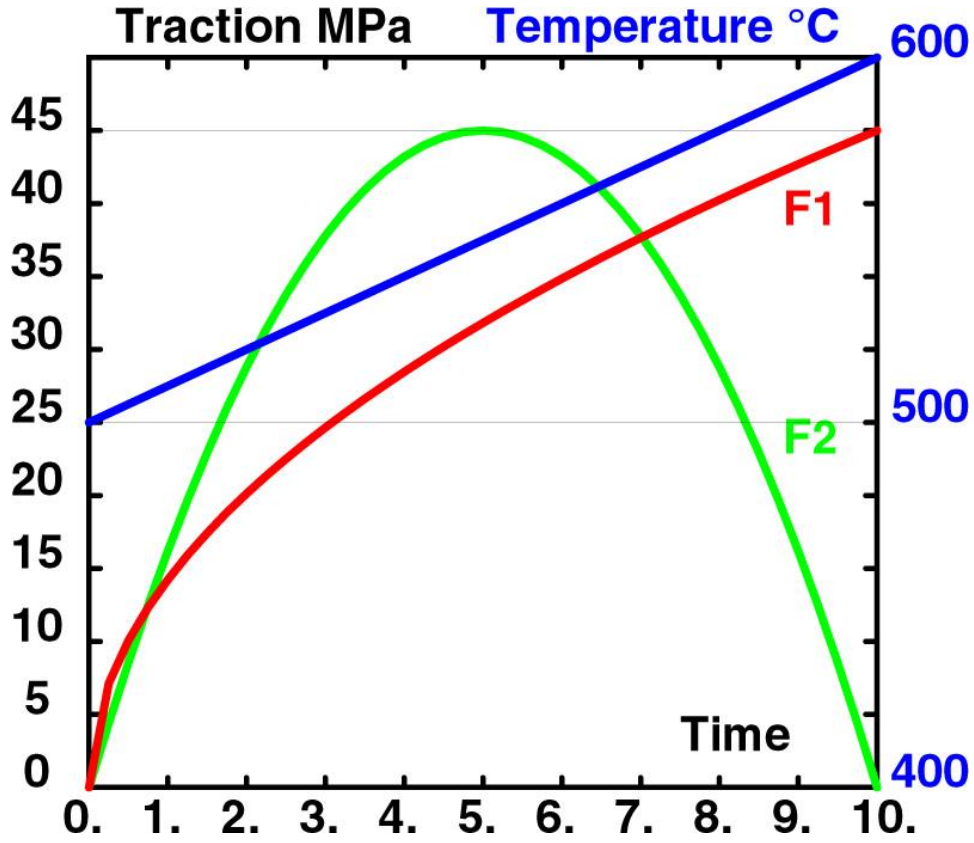
      Choix : utilisation d'un second découpage

Intégration en parallèle de la loi de comportement

➢ Modèle viscoplastique anisotherme

- ❑ Modèle de Chaboche avec écrouissage cinématique et isotrope

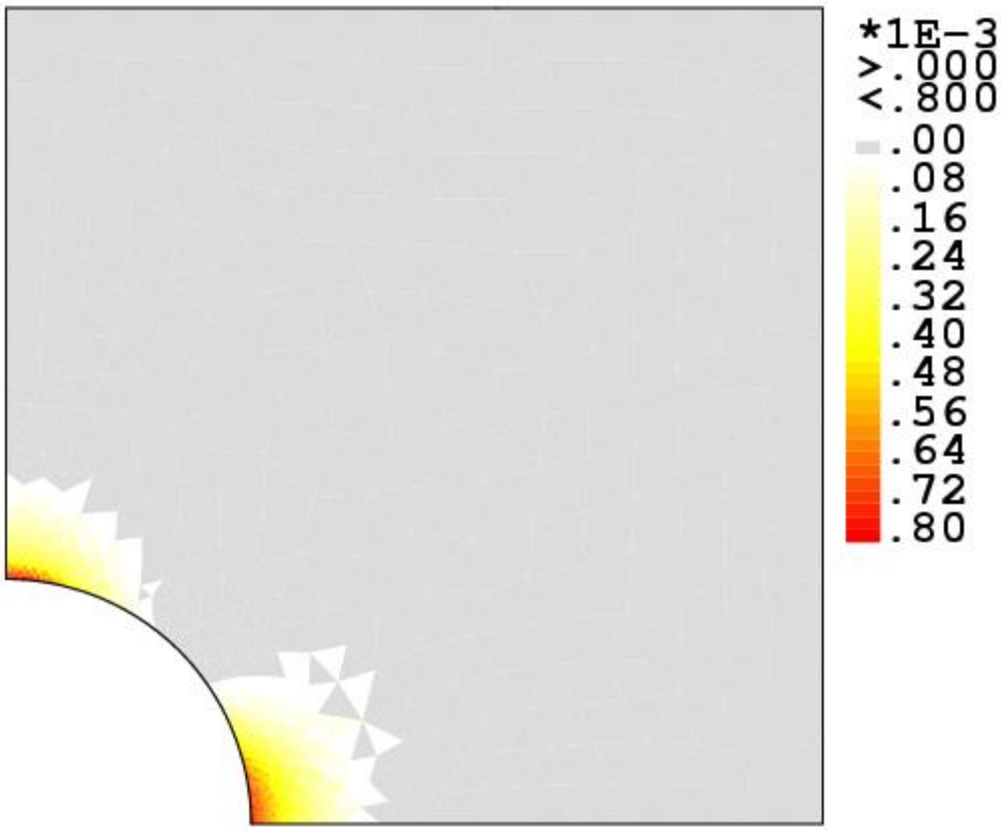




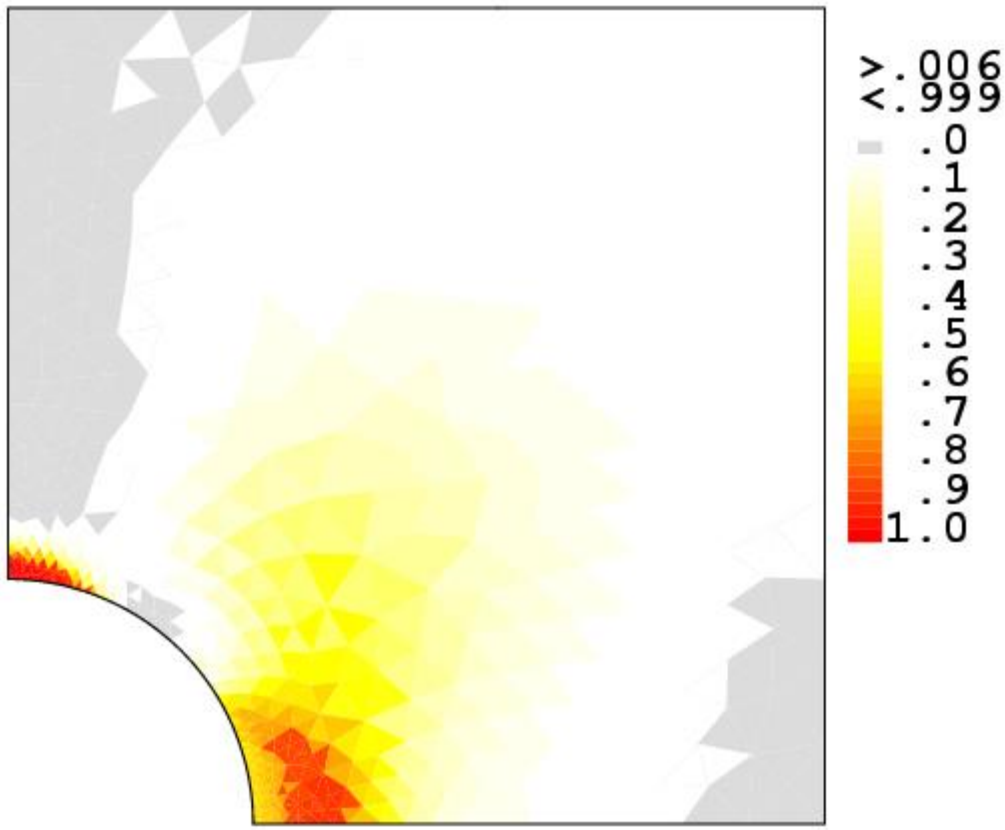
- ❑ Chargement découpé en 10 incréments

Intégration en parallèle de la loi de comportement

➢ **Déformation plastique cumulée & temps CPU par élément (par incrément)**



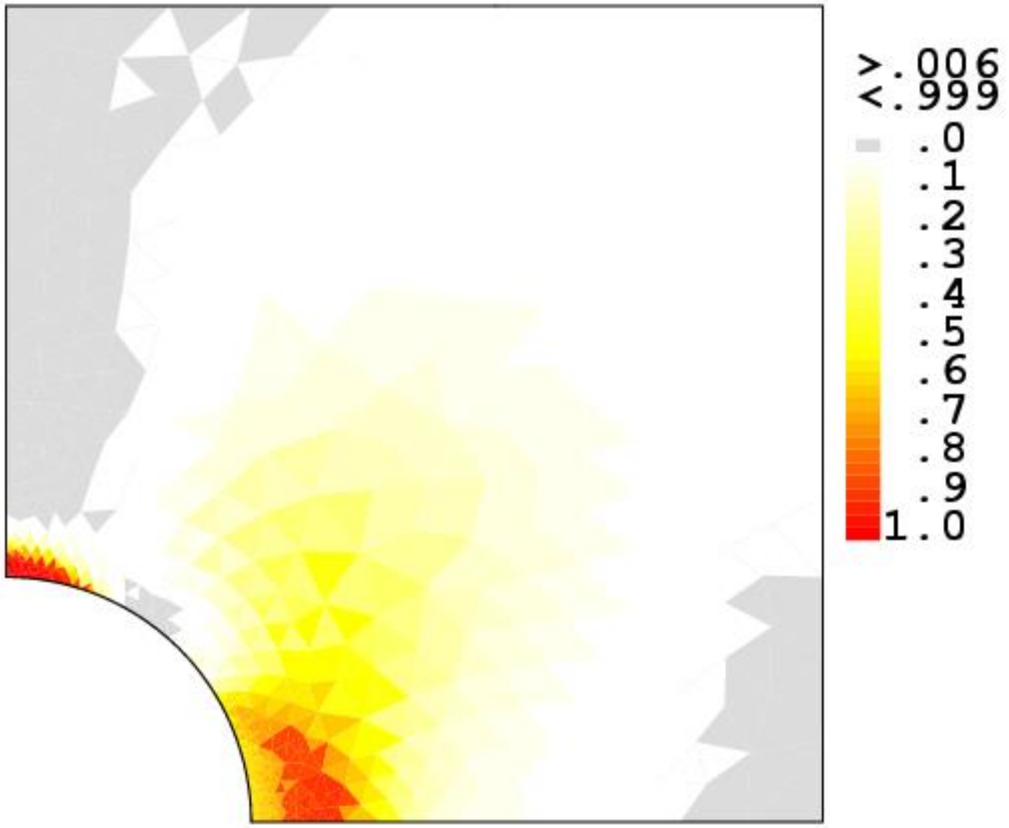
Plastic cumulated strain at t = 10



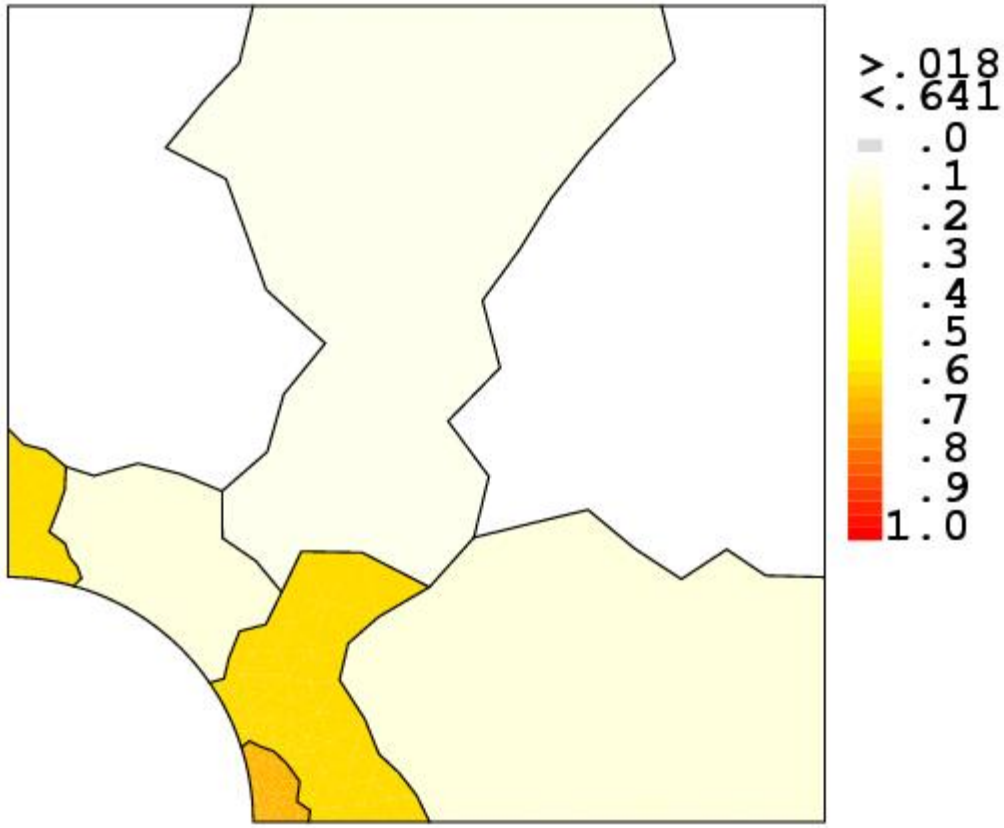
CPU time per element at t = 10

Intégration en parallèle de la loi de comportement

➢ **Temps CPU par élément & temps CPU par sous structure (par incrément)**



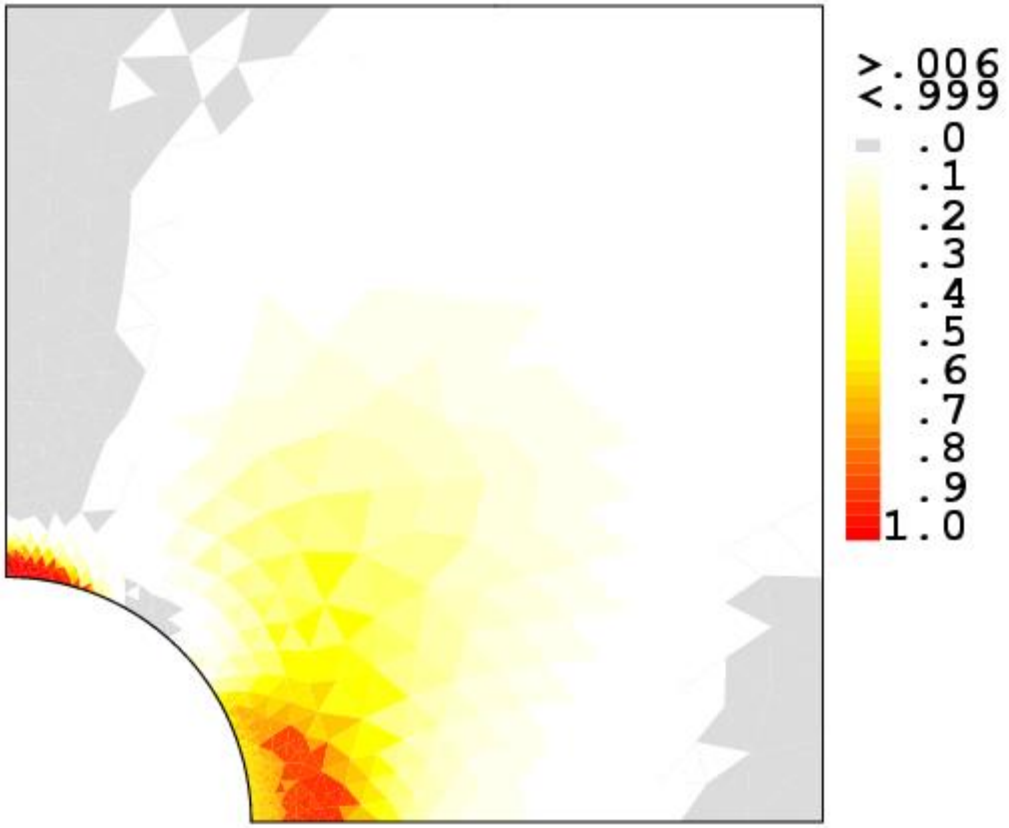
CPU time per element at t = 10



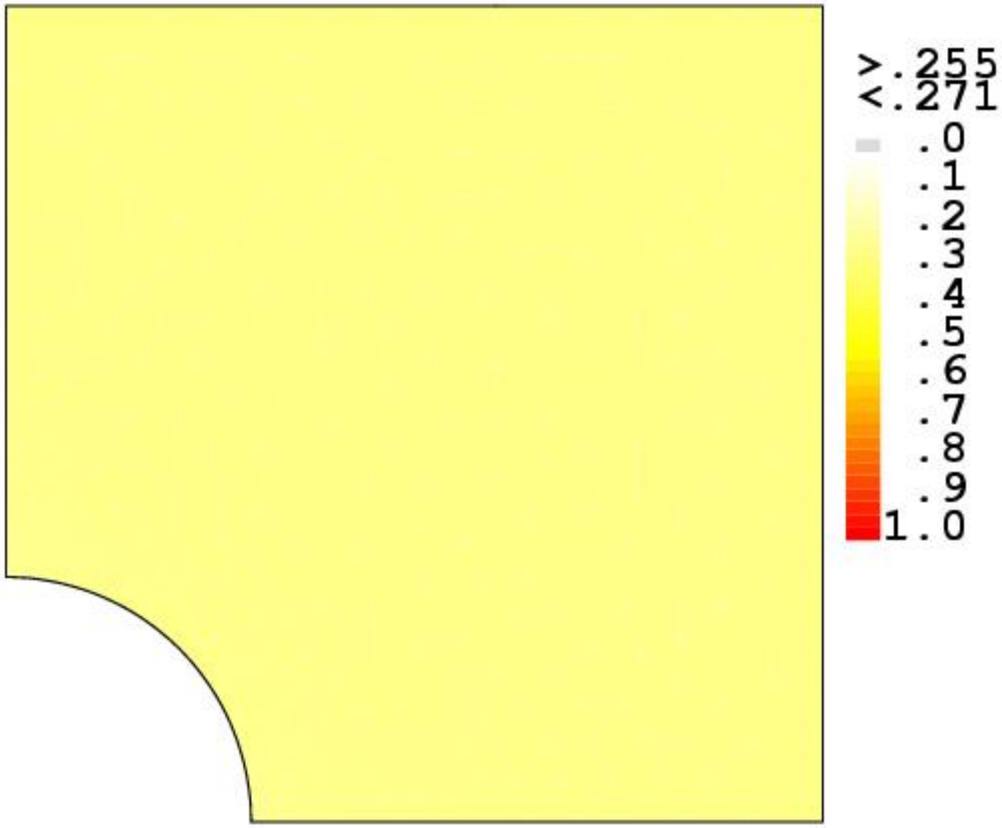
CPU time per sub-domain at t = 10

Intégration en parallèle de la loi de comportement

➢ **Temps CPU par élément & temps CPU par blocs d'éléments (par incrément)**



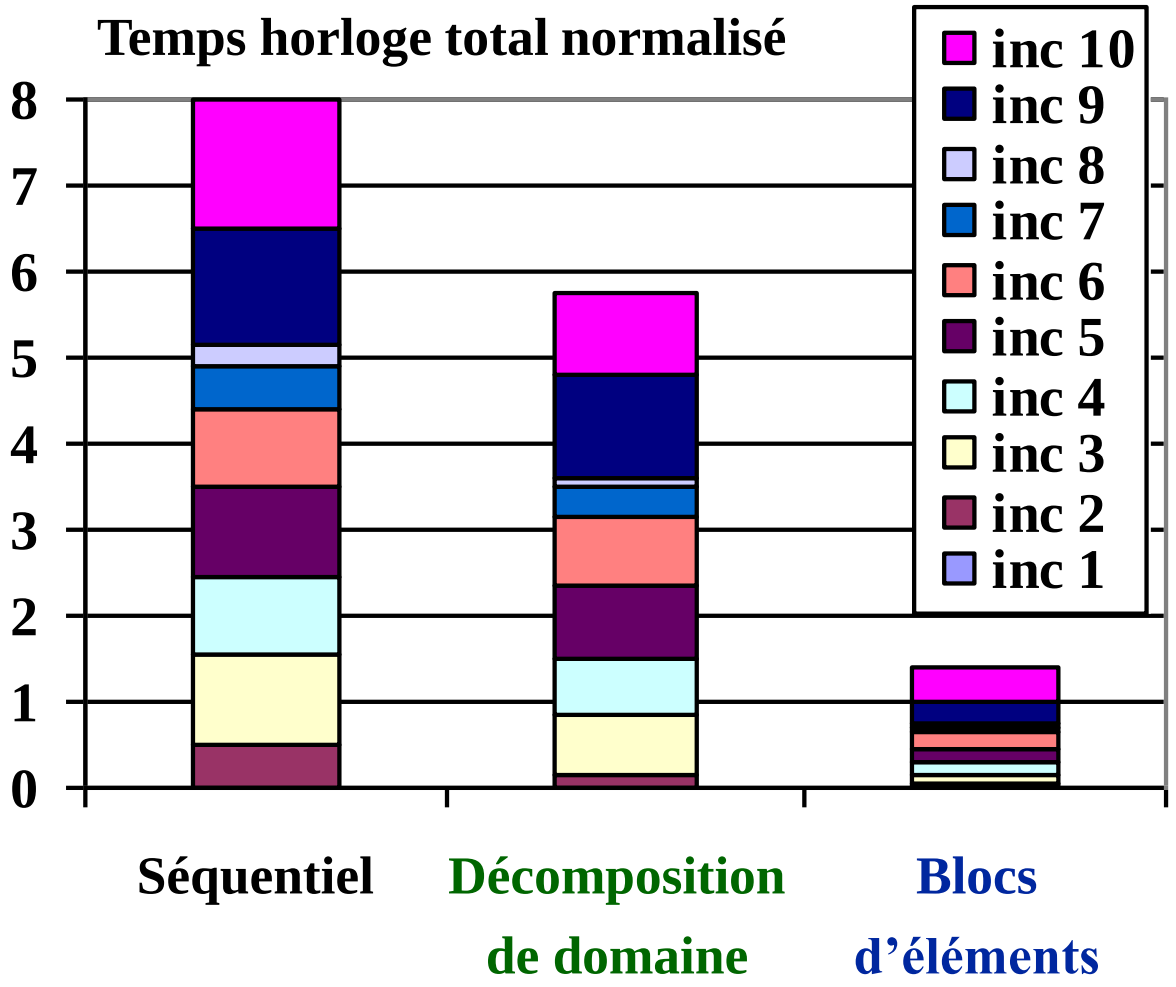
CPU time per element at t = 10



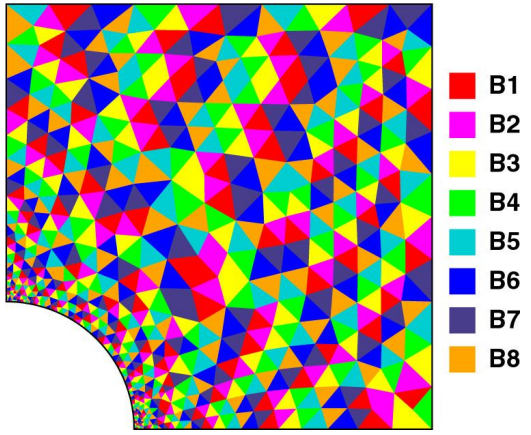
CPU time per bloc at t = 10

Intégration en parallèle de la loi de comportement

➢ Coût numérique pour 8 processeurs



Décomposition de domaine

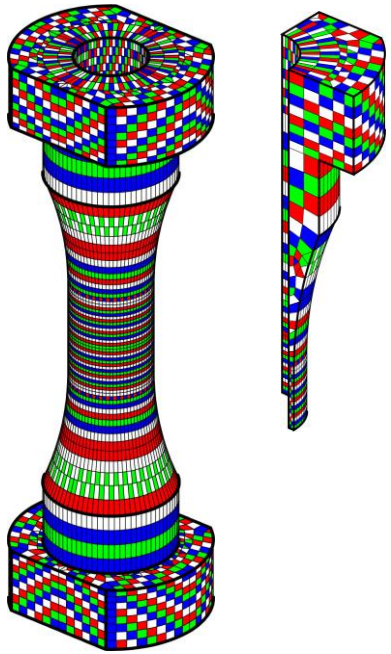


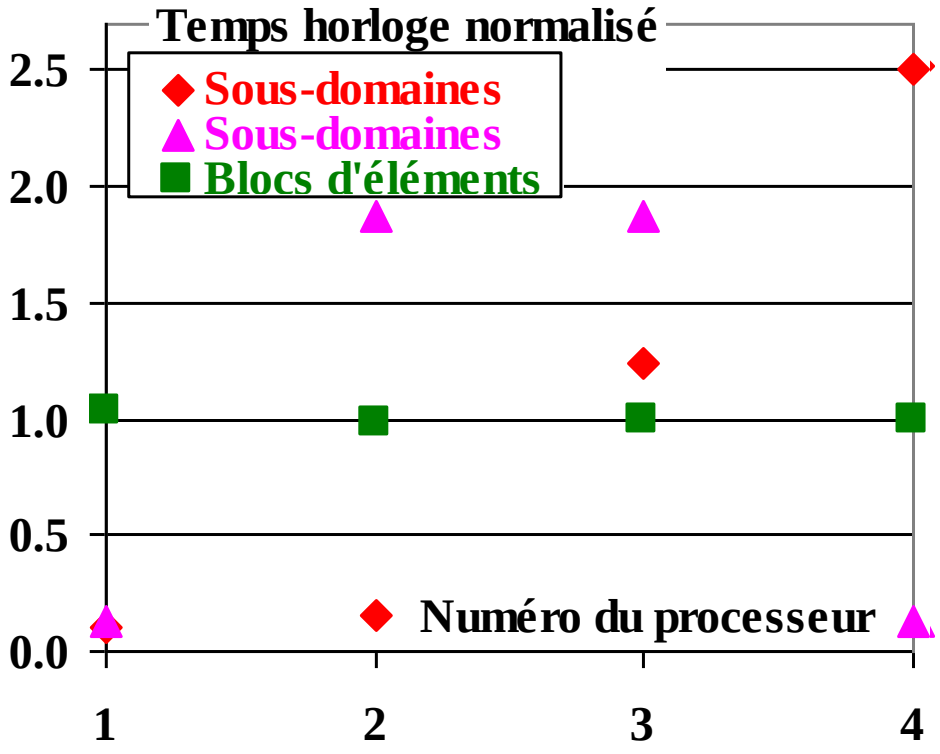
Blocs d'éléments

Intégration en parallèle de la loi de comportement

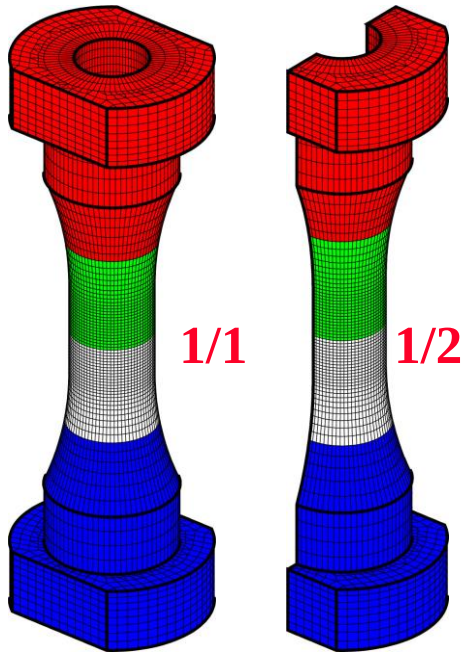
- **Problème 3D**
  - Décomposeur automatique
  - 4 processeurs
- **Equilibrage des calculs ?**

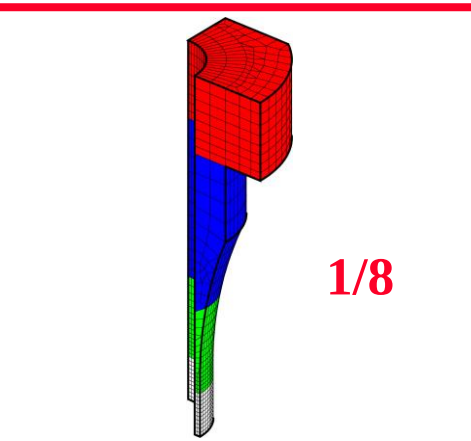
Blocs d'éléments





Utilisation d'un second découpage : Blocs d'éléments





Code Eléments Finis CAST3M

- **Code Eléments Finis orienté « objets »**
  - Différentes applications : Structure, Fluide…
  - But : Faciliter le développement de nouveaux algorithmes
- **Entités**
  - Objets dans la base de données → NOM, TYPE, VALEUR

    MAILLAGE, MODELE, CHAMPS (Nœuds, Eléments), ...
  - OPERATEURS → +, -, SURF, RESO, TRAC, …

    RESULTATS = OPERATEUR OPERANDES ;
  - PROCEDURES → fonctions utilisées comme les OPERATEURS

    Regroupement d'opérations sur les objets
- **Interface utilisateur**
  - Langage de programmation : ESOPE ← Parallélisation
    - Fortran 77 + Extensions + Gestion de la mémoire
  - Langage de commande : GIBIANE ← Parallélisation
    - Boucles, tests, procédures

ESOPE : Mémoire Virtuelle Programmée

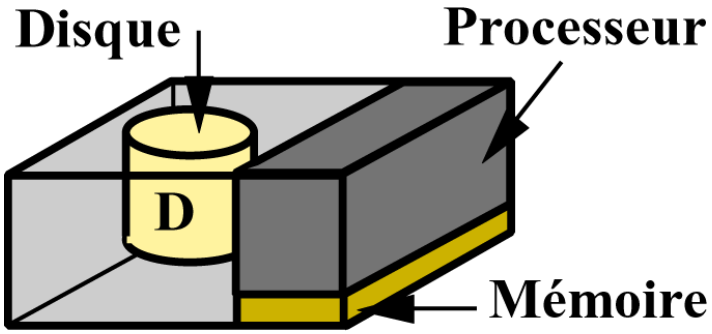
- **Structuration des données (C, F90, …) : FORTRAN 77 + Extensions …**

```
SEGMENT PSEG
      INTEGER LIST(N1)
      REAL*8   TAB(N1,N2)
END SEGMENT
```

Allocation Pseudo-dynamique de la mémoire

SEGMENT : 1 pointeur

N1 = 10 ; N2 = 15



SEGINI PSEG	initialisation
LIST(1) = 0	(PSEG.LIST(1)=0)
SEGDES PSEG	mode « inactif »
SEGACT PSEG	mode « actif »
SEGSUP PSEG	libération de la mémoire

- **Peut déplacer un SEGMENT non utilisé de la Mémoire vers le Disque**

  (mode « inactif » : SEGDES)

- **Rappel des SEGMENTS si nécessaire ➔ sans demande explicite de l'utilisateur**

  (activation de l'objet : SEGACT)

ESOPE Parallèle : Mémoire Virtuelle Partagée Programmée

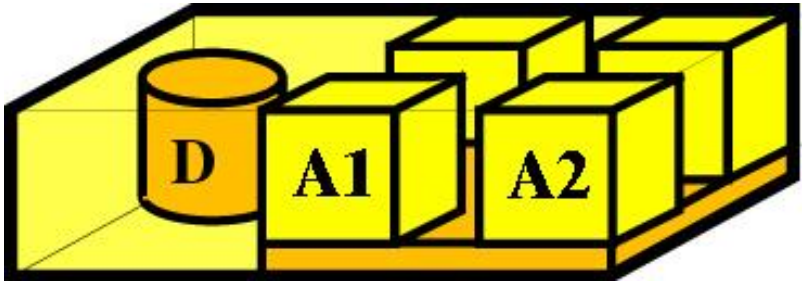
- **Ré-utilisation directe du code séquentiel (ESOPE)**
  - Tous les segments sont partagés
  - Limiter le nombre de nouvelles instructions
- **Limiter les phases d'attente et d'éventuels blocages**
  - SEGACT PSEG mode « lecture seule »
  - SEGACT PSEG*MOD mode « écriture »
- **Implémentation de dialogues complexes entre applications**
  - SEGACT PSEG*(MOD,R=N1,W=N2) activations conditionnelles

    possible après N1 SEGACT PSEG

    N2 SEGACT PSEG*MOD par une autre application
- **Assurer la cohérence des données & Cacher le transfert des données**
  - Utiliser les propriétés des Segments en mode « lecture seule » et « écriture »
    - Segment en mode « écriture » → une seule copie
    - Segment en mode « lecture seule » → plusieurs copies
  - Transfert en une seule opération du bloc de données demandé

ESOPE 2000 : Mémoire Partagée (SMP)

- **Les applications partagent la même mémoire**
  - **Synchronisation en utilisant les « pthreads »**
- **Exemple : requête en mode « écriture »**



Application A1		Application A2
SEGACT PSEG*MOD ➔ Mode « attente »	➔ (PSEG : A2 « écriture ») surveillance de PSEG Travail… …	SEGACT PSEG*MOD (PSEG utilisé par A2) Travail…
Propriétaire de PSEG ➔ Travail…	Etude PSEG queue ➔ ➔ A1 peut continuer Travail…	➔ SEGDES PSEG

Résolution de problèmes linéaires de grande taille

- **Limiter le coût numérique lorsque le nombre de DDL augmente (plus de 10^5)**
  - Limiter le remplissage de la matrice de rigidité lors de la factorisation
  - Encombrement de la matrice factorisée pour un cube avec N nœuds

    $N^{1.67}$ pour une méthode de type Cuthill Mac Kee

    $N^{1.34}$ pour une méthode de type Nested dissection

- **Nested dissection**
  - Partition récursive du maillage en 2 zones

    Numérotation des nœuds : zone 1, zone 2, interface
  - Minimiser l'encombrement de la matrice :

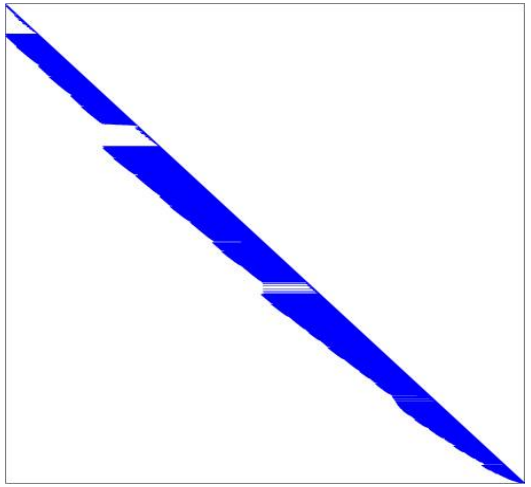
    Equilibrage des zones & minimiser la taille des interfaces

    Problème d'optimisation résolu par une méthode de Monte Carlo
- **Similaire à une technique de décomposition de domaine**
- **Solveur direct parallèle :**

  version "Multithreads" en mémoire partagée

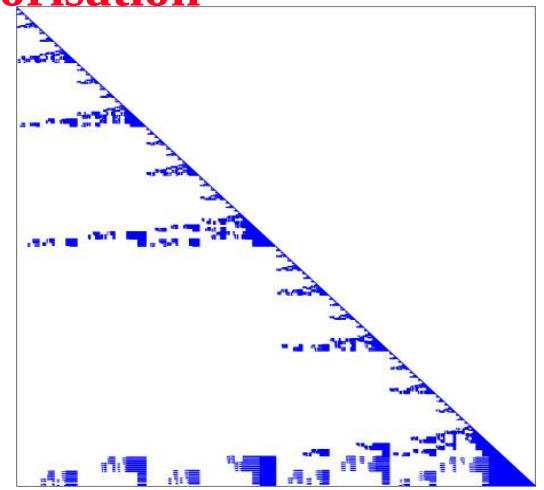
Le cube

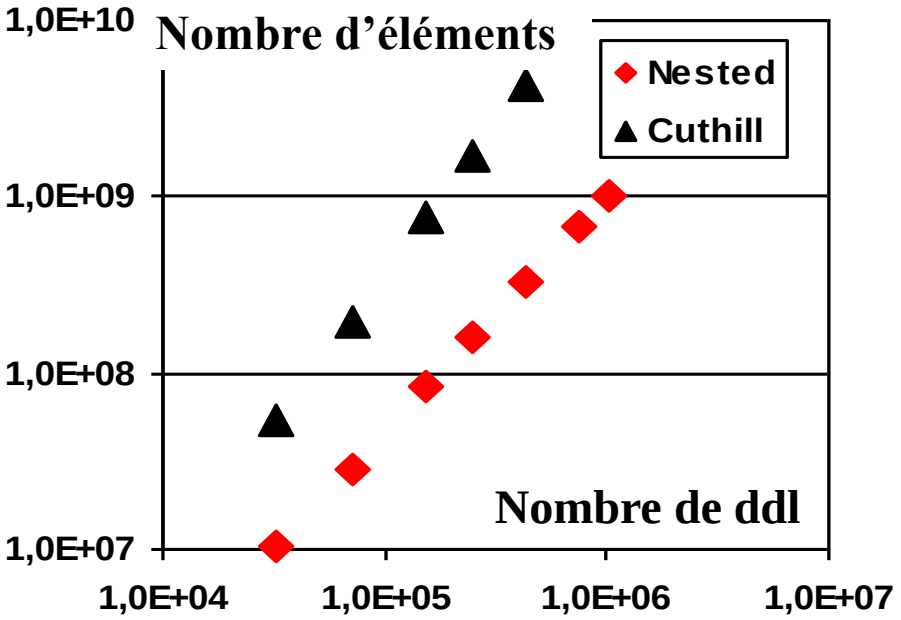
Remplissage de la matrice factorisée & Coût de la factorisation

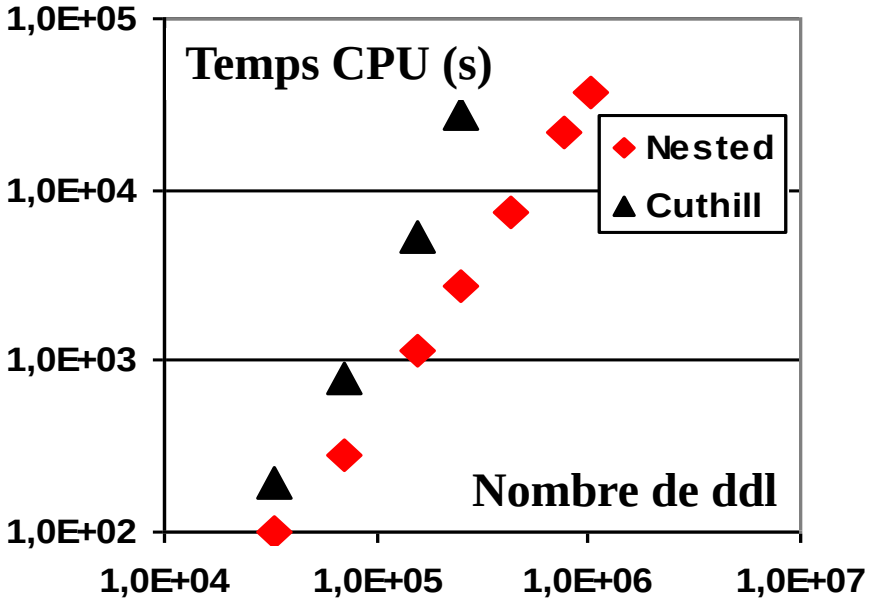


Cuthill-McKee : minimisation de la largeur de bande

Nested dissection : limiter le remplissage de la matrice

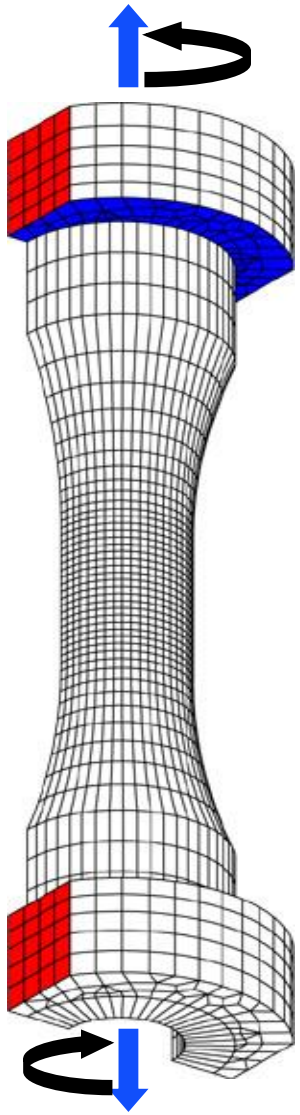


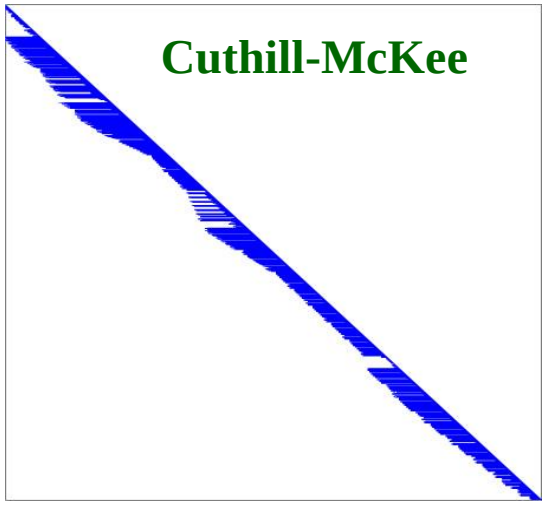


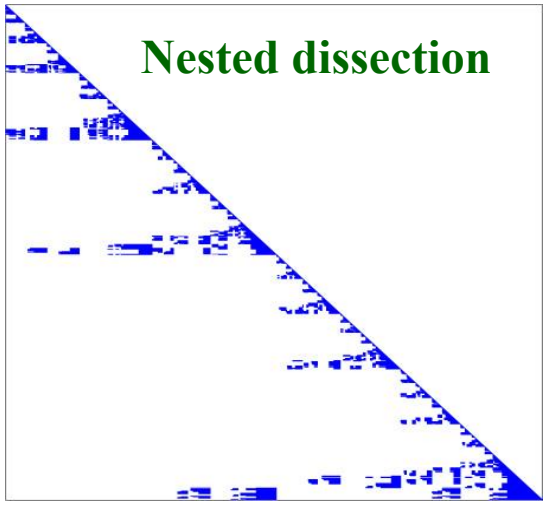


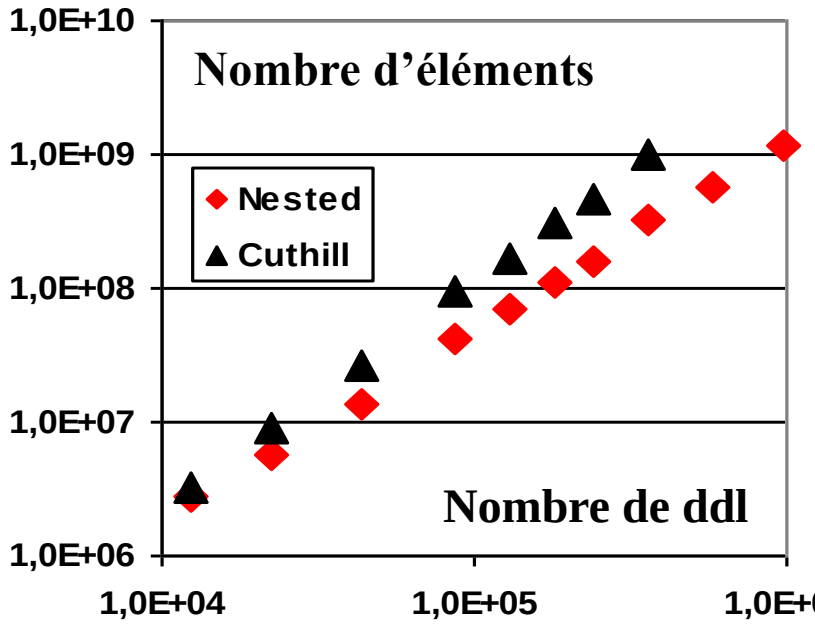
Eprouvette de traction-torsion

- Remplissage



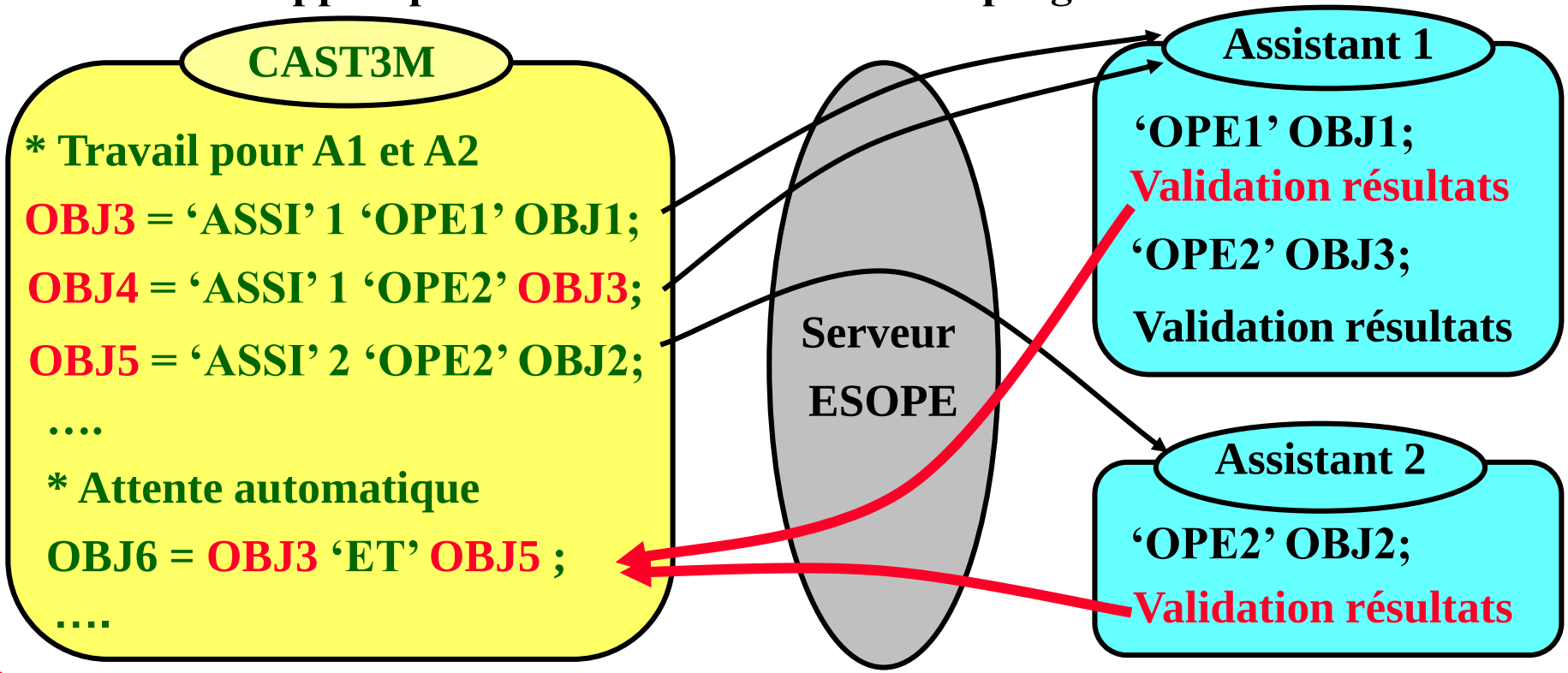






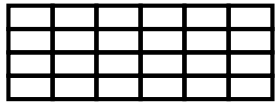
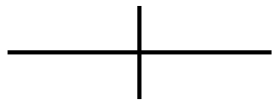
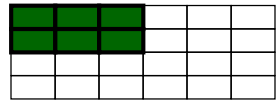
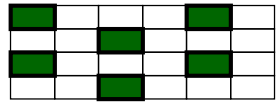
Distribution des calculs - GIBIANE

- **Opérateur ASSIstant (parallélisation au niveau utilisateur)**
  - Soumettre un travail « élémentaire » à un « ASSISTANT »
  - Assurer la cohérence des données & Limiter les phases d'attente
  - Développé à partir de l'environnement de programmation ESOPE



- **Mode « DataFlow » : Lancer l'opération – ne pas attendre le résultat**
- **Mode « Data Parallèle » : Option 'TOUS' (Sous structuration)**

Implémentation parallèle pour une itération

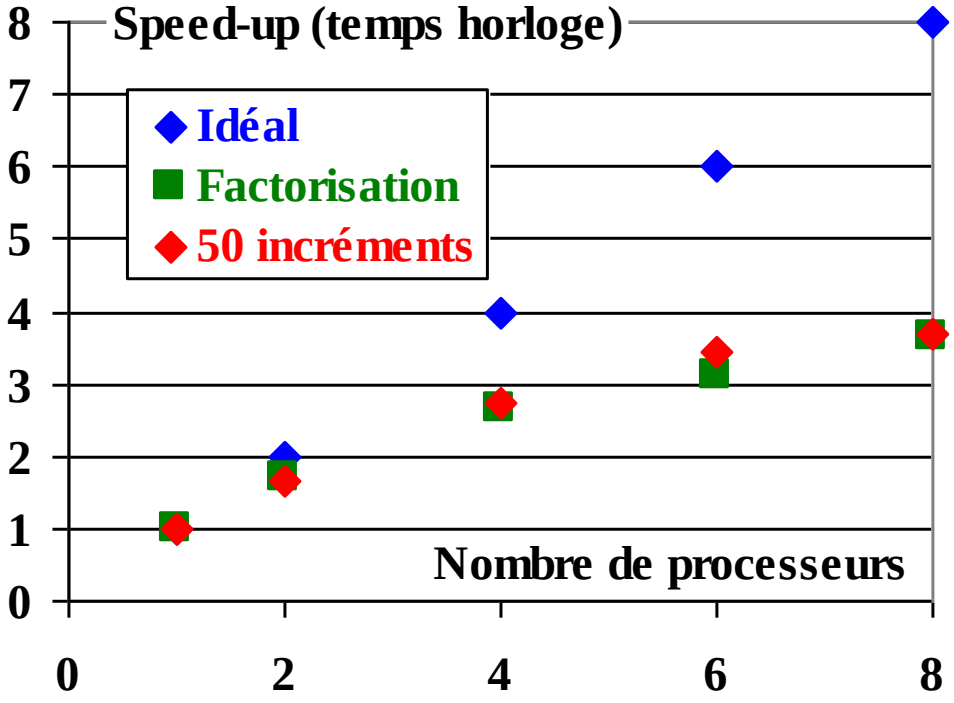
Maître - Séquentiel			Maître – Assistant (n° I1)	
Gestion des calculs	**Pb Complet**	**Pb Interface**	**Décomp Dom**	**Blocs Elém**
				
Correction du vecteur déplacement				
FEQ1 = ‘ASSI’ ‘SUPE’ … RES1 ;			➔ RES1 FEQ1 . i1	
Condensation du second membre				
FEQ2 = ‘ET’ FEQ1 ; DEP2 = ‘RESO’ RIG2 FEQ2 ;		← FEQ1 FEQ2 DEP2		
Montée - Descente				
DEP3 = ‘ASSI’ ‘SUPE’ … DEP2 ;			➔ DEP2 DEP3 . i1	
Remontée				
Accélération de convergence				
DEP4 = ‘ASSI’ ‘TOUS’ …. ;			DEP4 . i1	
DEP0 = DEP0 ‘+’ DEP4 ;	← DEP4 DEP0			
Déplacement				
Intégration de la loi de comportement & Test de convergence				
SIG1 EPS1= ‘ASSI’ ‘ECOU’ DEP0; FIN1 = ‘ASSI’ ‘BSIG’ SIG1 ;				➔ DEP0 EPS1 . i1 SIG1 . i1 FIN1 . i1
Contribution des forces internes				
FIN2 = ‘ET’ FIN1 ;	← FIN1 FIN2			
Assemblage des forces internes				
RES1 = ‘ASSI’ ‘–’ FEX1 FIN2 ;			➔ FIN2 RES1 . i1	
Contribution des forces résiduelles				

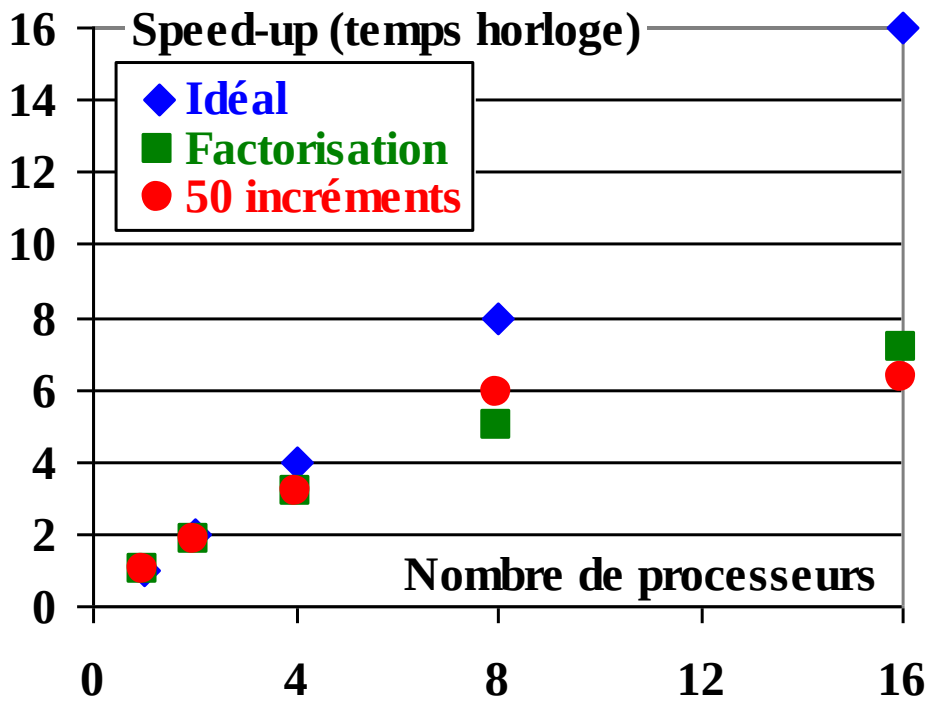
Eprouvette de traction-torsion – Chargement cyclique

➢ Comportement de la stratégie & Influence du type d'ordinateur

- ❑ PC
- ❑ 8 processeurs (quad-core)
- ❑ 16 Go mémoire partagée
- ❑ Linux

- ❑ IBM POWER5+
- ❑ 16 processeurs (dual-core)
- ❑ 64 Go mémoire partagée
- ❑ AIX (optimisation des paramètres)

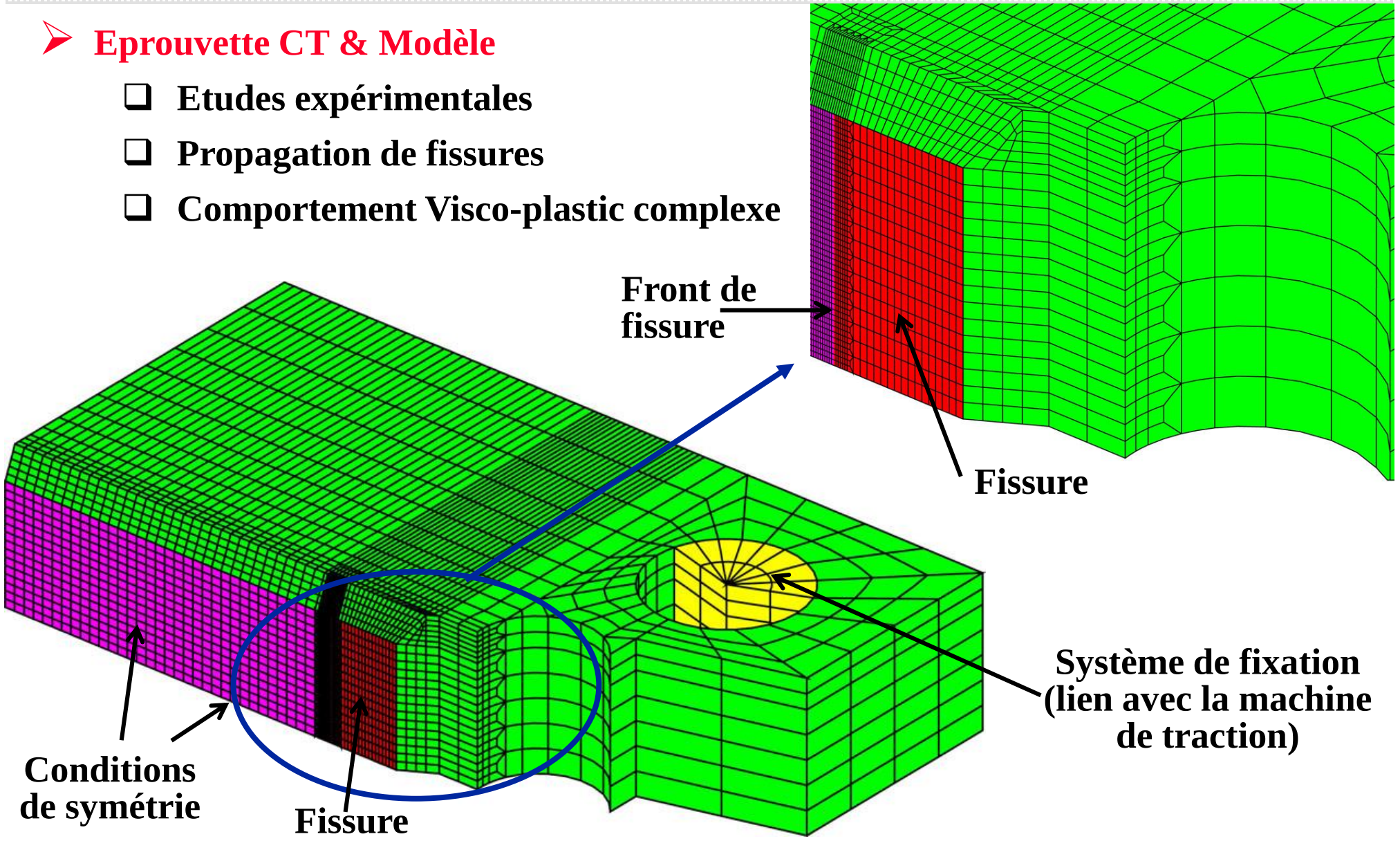




Eprouvette CT

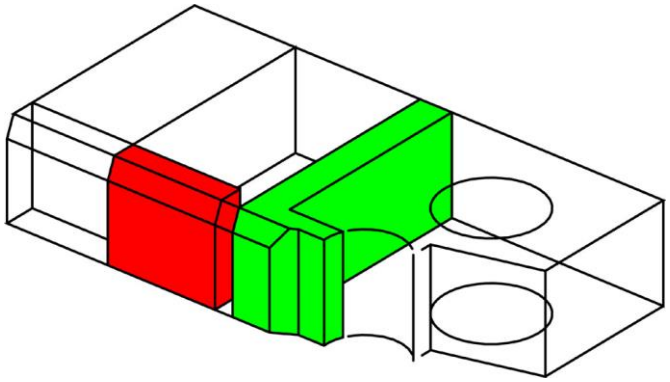
➢ Eprouvette CT & Modèle

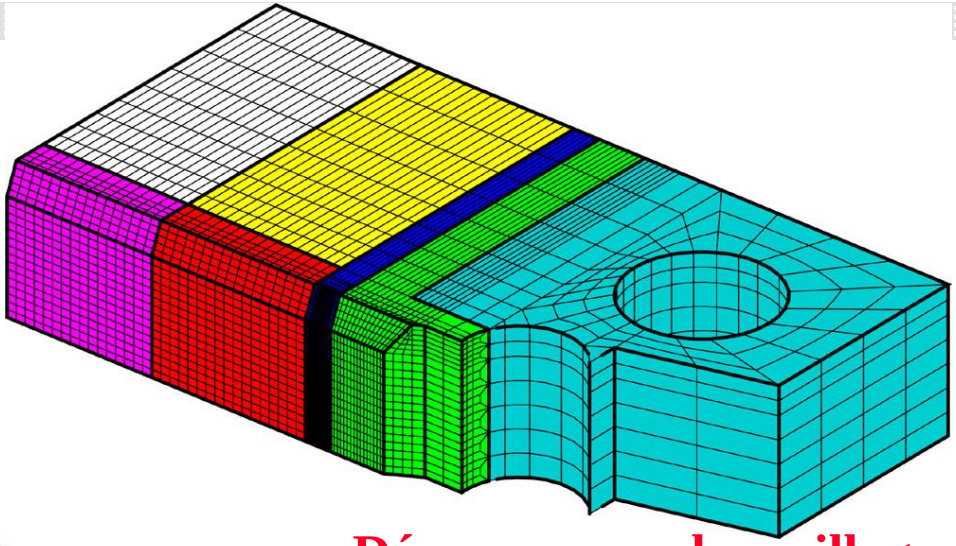
- ❑ Etudes expérimentales
- ❑ Propagation de fissures
- ❑ Comportement Visco-plastic complexe



Eprouvette CT

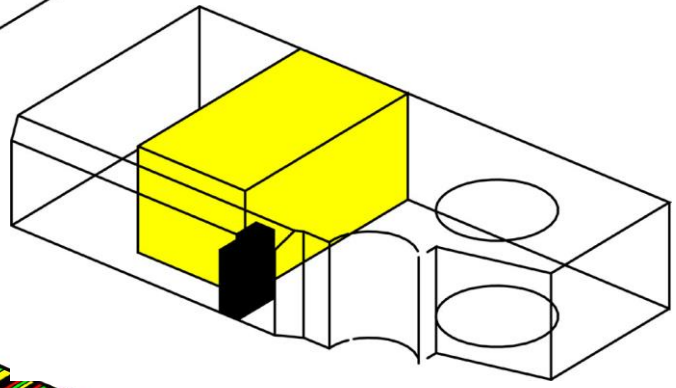
- Sous domaines

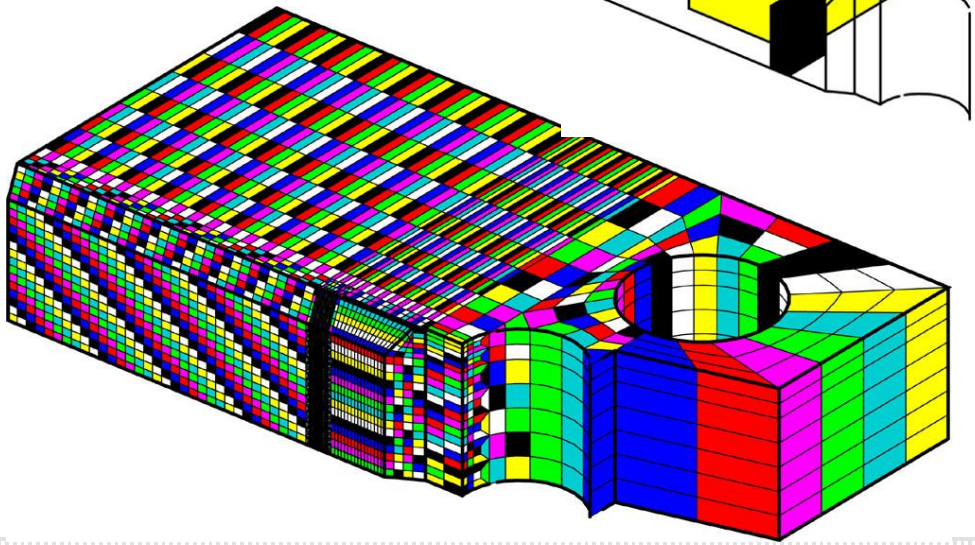


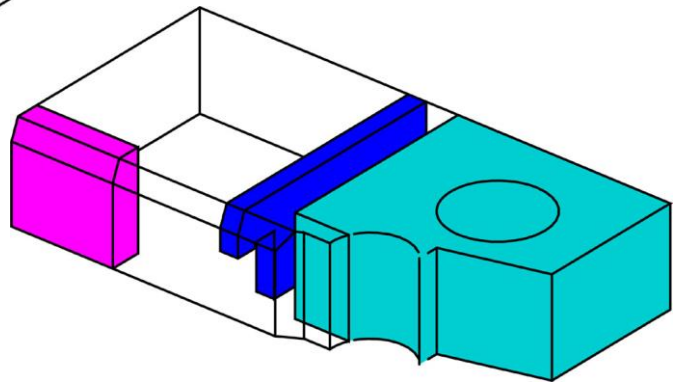


Décomposeur de maillage automatique (numérotation du solveur)

- Blocs d'éléments







Eprouvette CT

- Calculateur IBM
  - 16 processeurs (dual-core POWER5+)
  - 64 Go mémoire partagée

Speed up d'environ 10 pour la simulation non-linéaire complète avec 16 processeurs

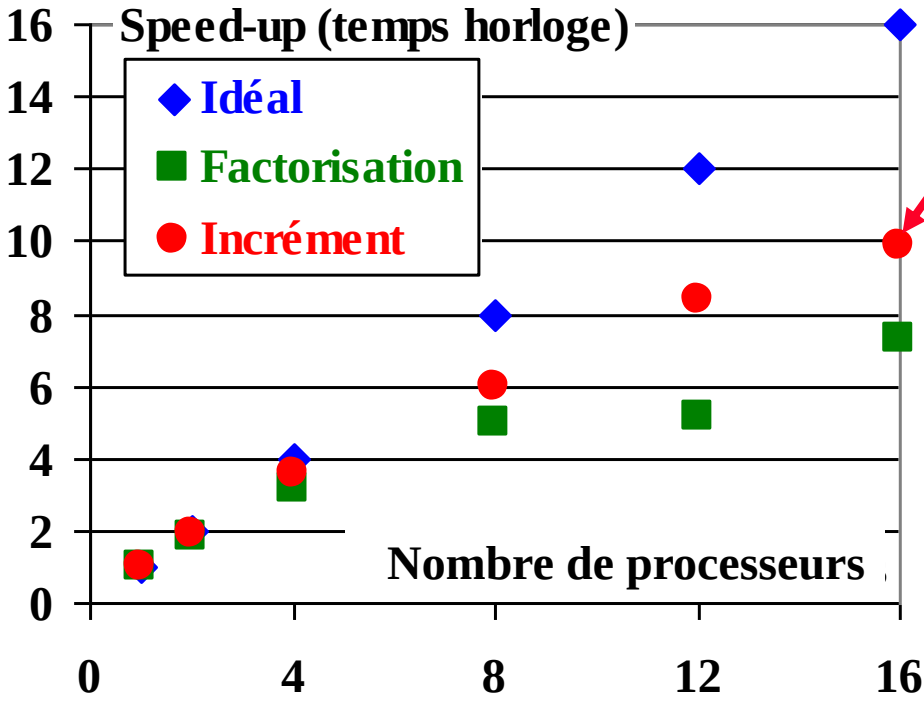
240 000 DDL - 18 000 éléments

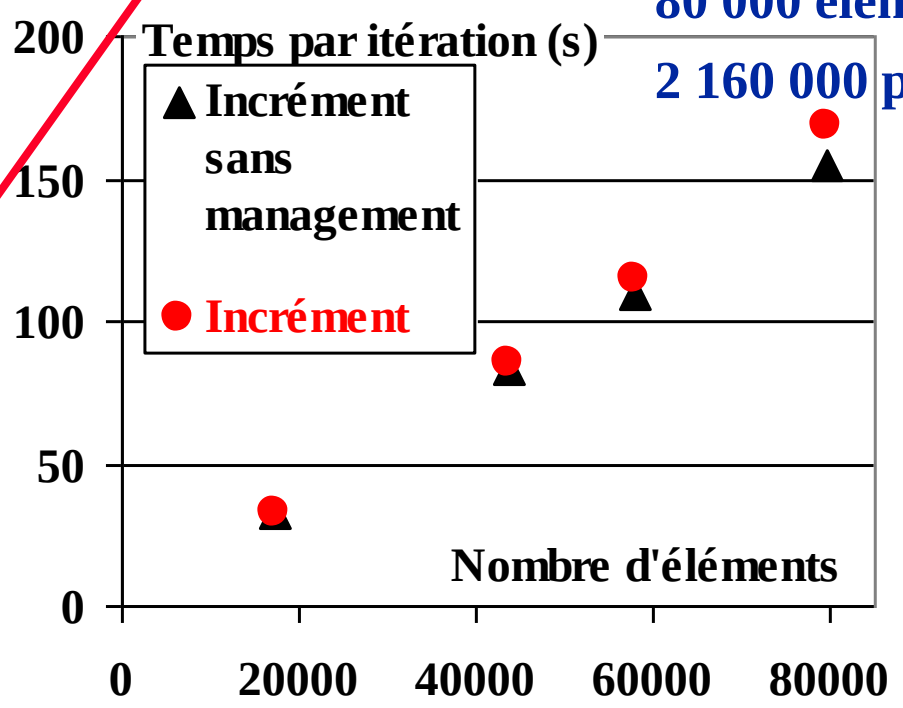
500 000 points d'intégration

Influence de la taille du modèle 1 100 000 DDL

80 000 éléments

2 160 000 pts int.





Conclusions

- **Réduction du coût numérique des simulations non linéaires en Eléments Finis**
  - **Environnement de programmation parallèle**
  - **Approche parallèle pour une large classe de problèmes non linéaires**
- **Environnement de programmation adapté au parallélisme**
  - **Cacher le transfert des données & Assurer la cohérence des données**
  - **Deux niveaux de développement (programmation & utilisation)**
  - **Le programmeur peut focaliser son attention l'organisation du programme**
  - **Système compatible avec l'architecture des ordinateurs modernes**
- **Approche parallèle pour les simulations non linéaires**
  - **L'utilisation de deux découpages donne un bon équilibrage**
  - **Résultats encourageants en 3D**

    Les performances de l'approche peuvent être augmentées

    Démarches utilisées en « Production » (version utilisateur de CAST3M)
- **Augmenter les performances et les possibilités du système**

  Réduction du coût de la gestion des objets (mémoire distribuée)