

## HYDRO

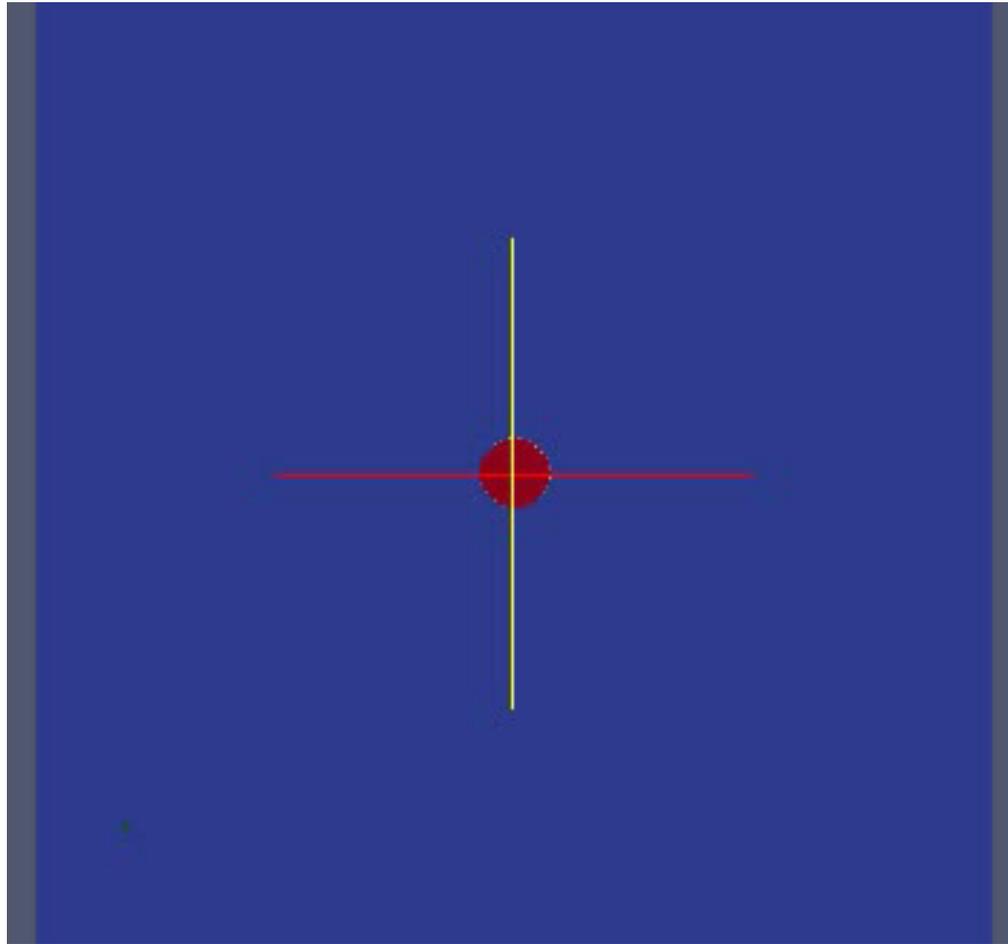
De la version séquentielle à la version hybride MPI+OpenMP



# Présentation du code HYDRO

- HYDRO est une version simplifiée du code d'astrophysique RAMSES.
- Code de mécanique des fluides (CFD), qui résout les équations d'Euler compressible de l'hydrodynamique en 2D.
- Méthode volume finis utilisant un schéma de Godunov d'ordre 2, avec résolution d'un problème de Riemann à chaque interface sur une grille régulière cartésienne 2D.
- 1500 lignes pour la version séquentielle F90. Disponible en Fortran ou en C.
- Facilement customizable (taille du domaine, nombre d'itération en temps, fréquence des IO, etc.), via un fichier d'input ASCII contenant des couples clés/valeurs.
- Par défaut, on va simuler une explosion de Sedov et suivre son évolution au cours du temps.

# Simulation d'une explosion avec HYDRO



# Équations et variables

Équations d'Euler pour un gaz idéal :

$$\partial_t U + \partial_x F(U) + \partial_y G(U) = 0$$

Avec  $U$  le vecteur des variables conservatives

$${}^tU = (\rho, \rho u, \rho v, E)$$

$\rho$  la densité

$u$  la vitesse suivant la première dimension  $x$

$v$  la vitesse suivant la deuxième dimension  $y$

$E$  l'énergie totale

# Schéma numérique

Après discrétisation et intégration par la méthode des volumes finis, on obtient :

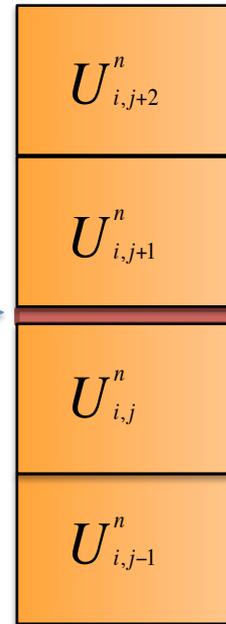
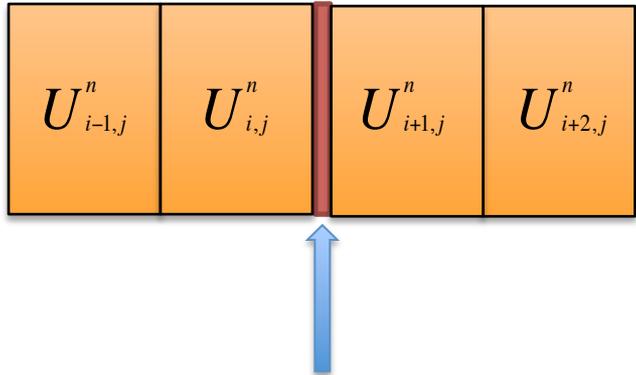
$$U_{i,j}^{n+1} = U_{i,j}^n + \frac{\Delta t}{\Delta x} \left( F_{i+1/2,j}^{n+1/2} - F_{i-1/2,j}^{n+1/2} \right) + \frac{\Delta t}{\Delta y} \left( G_{i,j+1/2}^{n+1/2} - G_{i,j-1/2}^{n+1/2} \right)$$

Où F et G sont les vecteurs flux suivant les deux dimensions.

Dans le code, le domaine discrétisé au n<sup>ieme</sup> pas de temps est stocké dans le tableau `uold(1:nx,1:ny,1:nvar)`

$$U_{i,j}^n \approx uold(i,j,1:4)$$

# Calcul du flux



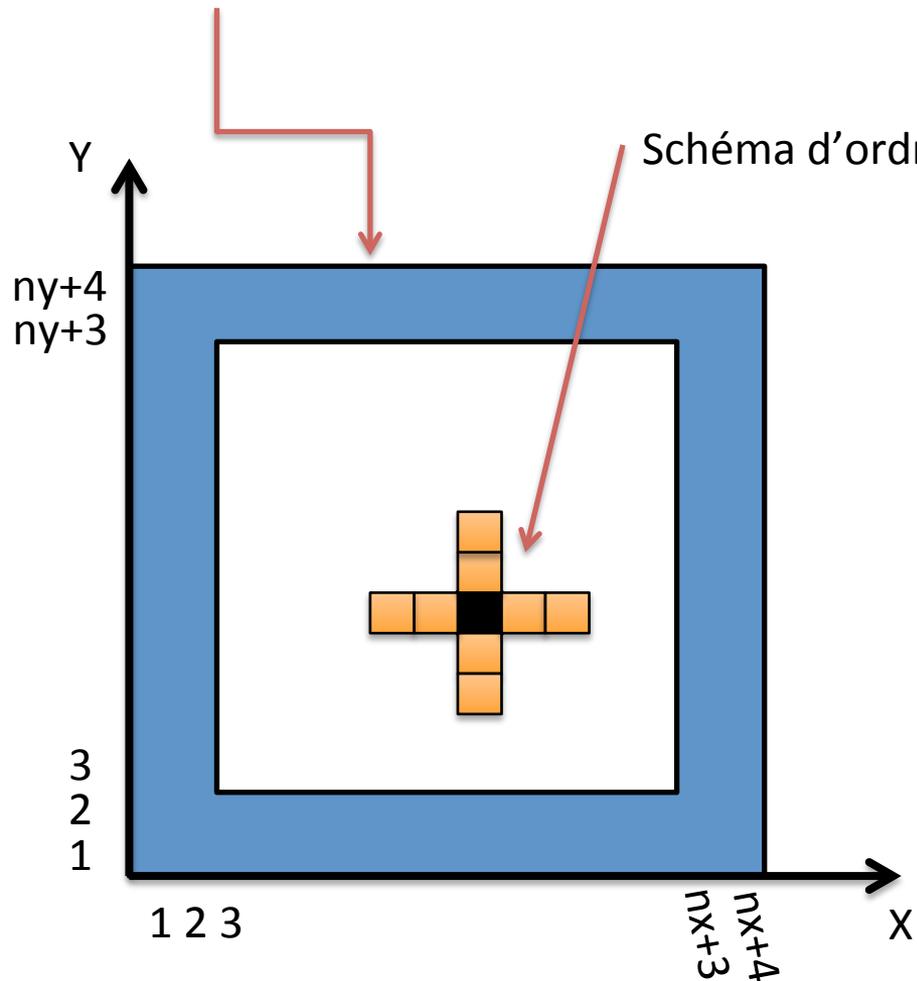
$$F_{i+1/2,j}^{n+1/2} = \text{function}(U_{i-1,j}^n, U_{i,j}^n, U_{i+1,j}^n, U_{i+2,j}^n)$$

$$G_{i,j+1/2}^{n+1/2} = \text{function}(U_{i,j-1}^n, U_{i,j}^n, U_{i,j+1}^n, U_{i,j+2}^n)$$

Dans le code, l'enchaînement des sous-routines *constoprism*, *trace*, *riemann* et *cmpflx* appelées depuis *godunov* permet de calculer ce flux...

# HYDRO version séquentielle

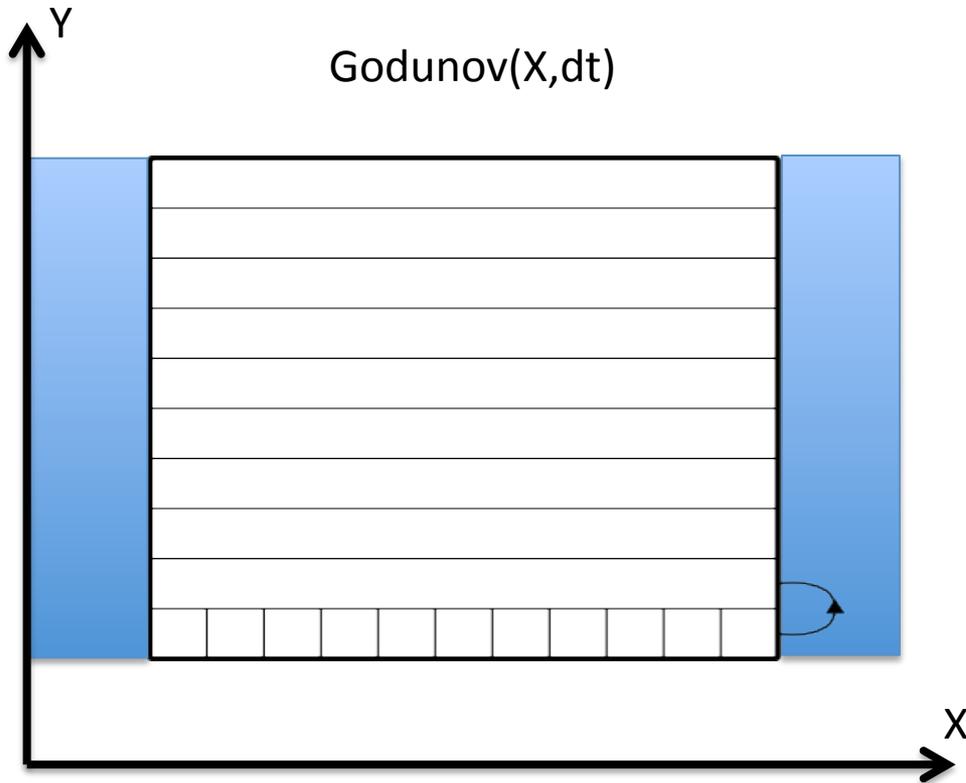
Mailles fantômes pour gérer les CL du domaine physique



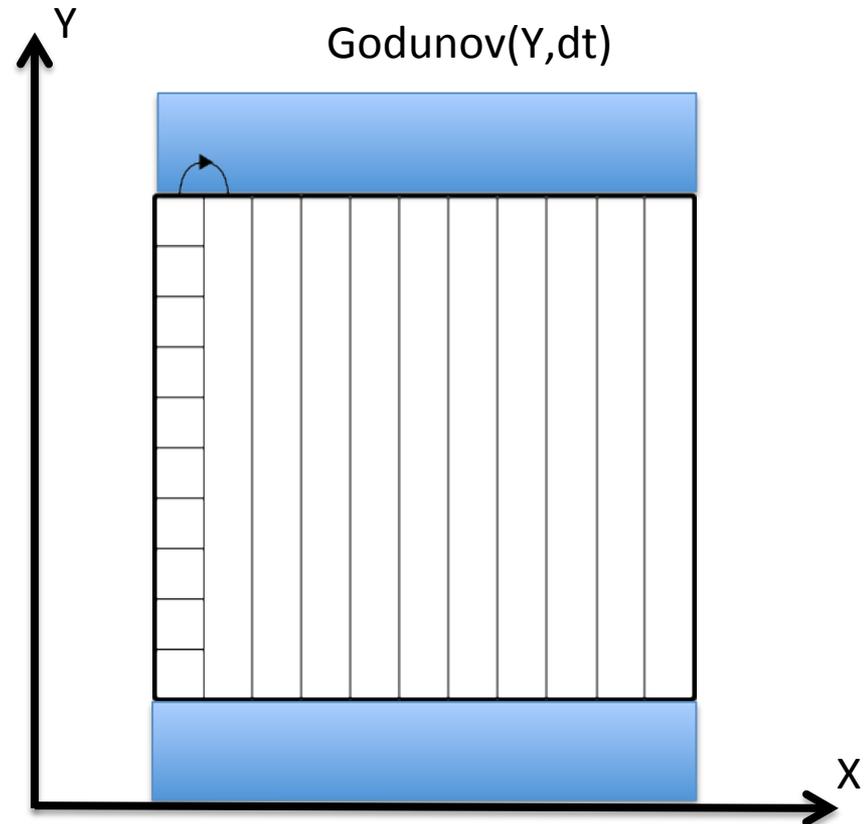
Algorithme HYDRO

```
initialize uold(:, :, :)  
while  $t < tend$  do  
  dt = cmpdt(); // compute time step  
  if  $n \% 2 == 0$  step  
  then  
    Godunov(X, dt); Godunov(Y, dt); // update uold  
  else  
    Godunov(Y, dt); Godunov(X, dt); // at t+dt  
  end if  
  t = t + dt  
end while
```

# HYDRO version séquentielle



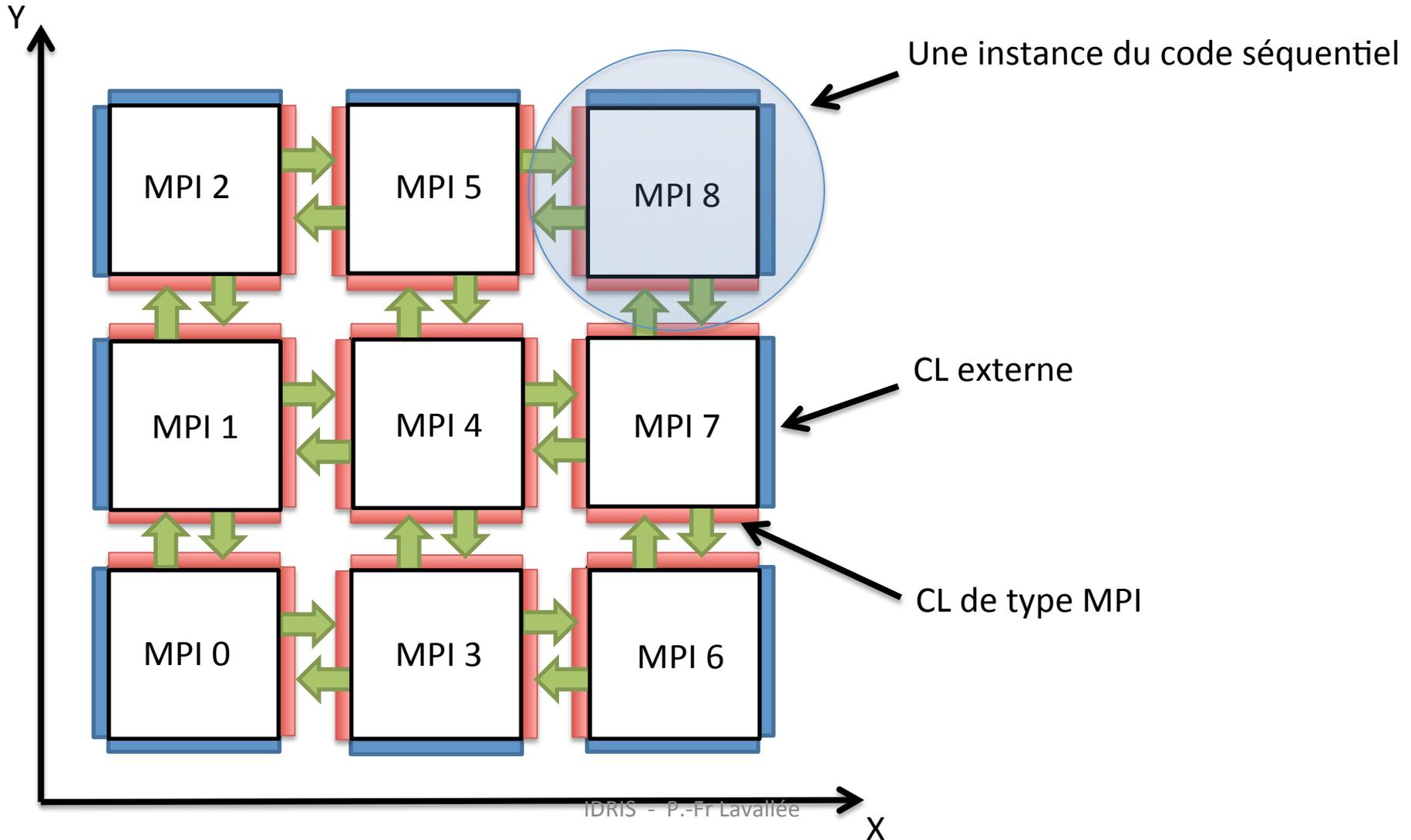
1. MAJ des CL pour les mailles fantômes verticales
2. Contribution des flux des interfaces verticales pour la MAJ de  $uold(:, :, :)$ , ligne par ligne



1. MAJ des CL pour les mailles fantômes horizontales
2. Contribution des flux des interfaces horizontales pour la MAJ de  $uold(:, :, :)$ , colonne par colonne

# HYDRO version parallèle MPI

On reprend la version séquentielle qu'on duplique plusieurs fois et on ajoute un nouveau type de CL : la CL MPI (échange de valeurs avec ses voisins)

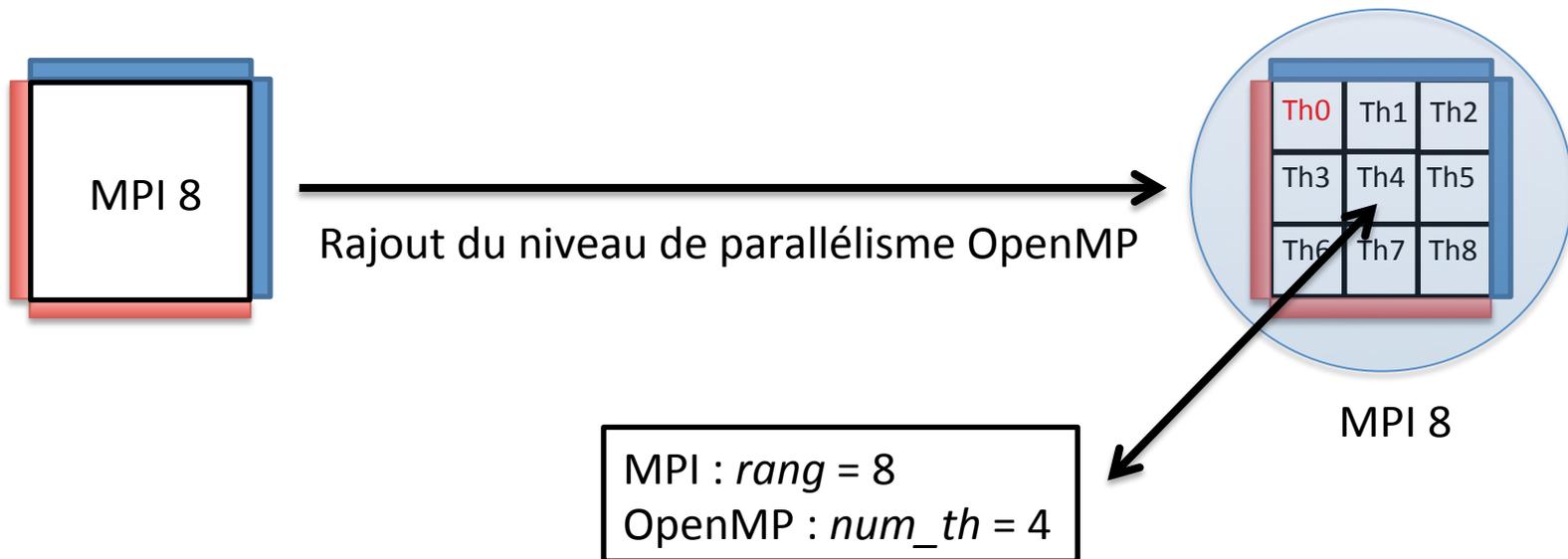


# HYDRO version parallèle MPI

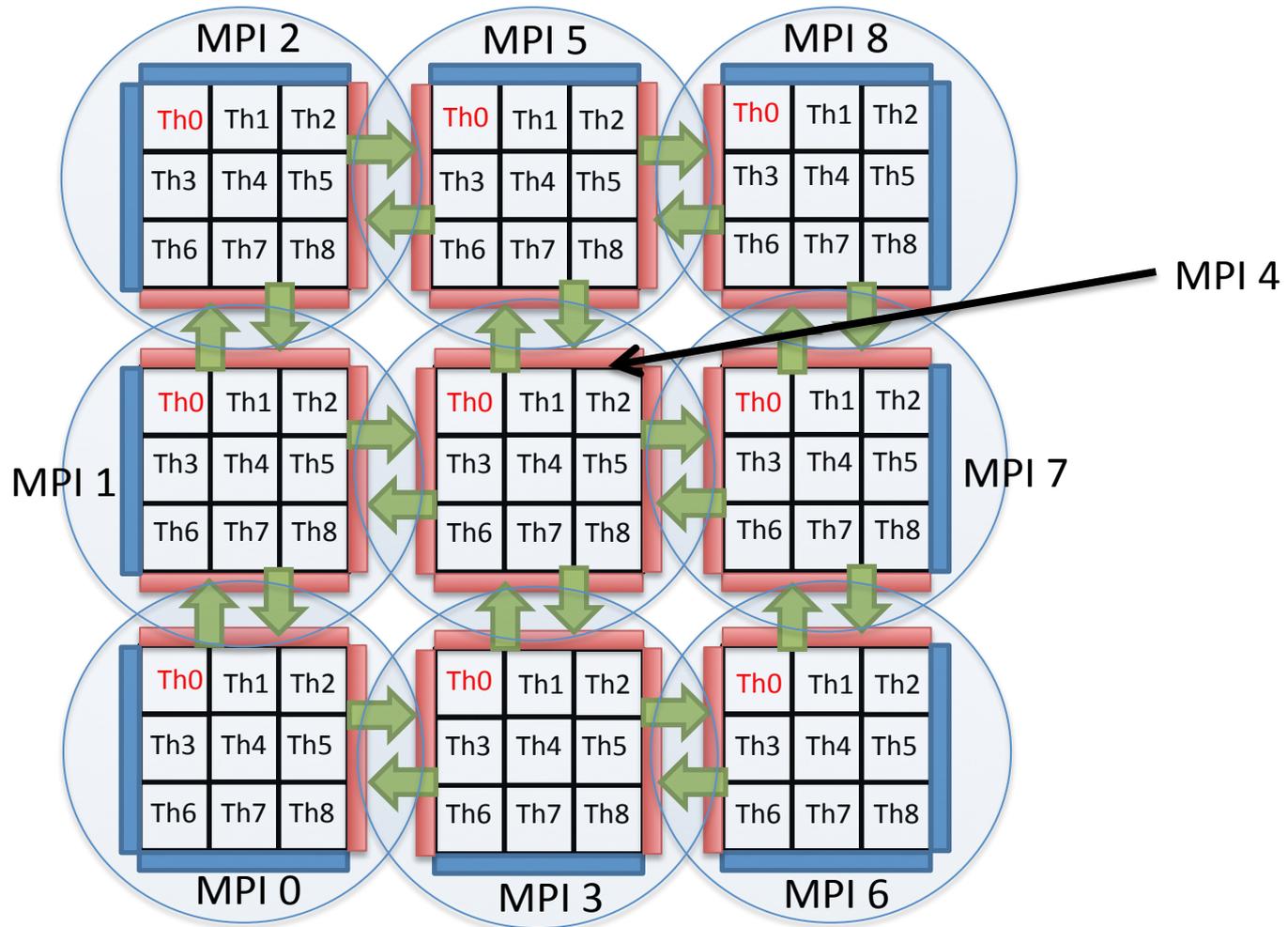
- Utilisation d'une topologie MPI 2D pour répartir les processus MPI dans chacune des directions et calculer la taille locale des sous-domaines associés à chaque processus MPI
- Détermination des voisins à chaque processus MPI (pour savoir à/de qui envoyer/recevoir !)
- Création de deux types dérivés MPI (un dans chaque direction) pour ne faire qu'un seul envoi/réception par voisin
- Le calcul du pas de temps nécessite une réduction pour calculer un maximum global sur la totalité des sous-domaines MPI

# HYDRO version parallèle hybride

- Au sein de chaque processus MPI, on rajoute un niveau de parallélisme OpenMP (approche Coarse Grain de type décomposition de domaine)
- Support du multi-threading de type `MPI_THREAD_FUNNELED`, les communications sont faites par le thread maître (celui de rang 0) pendant que les autres threads attendent sur une barrière de synchronisation
- Chaque thread est identifié de façon unique par son numéro de processus MPI (*rang*) et par son numéro de thread (*num\_th*)

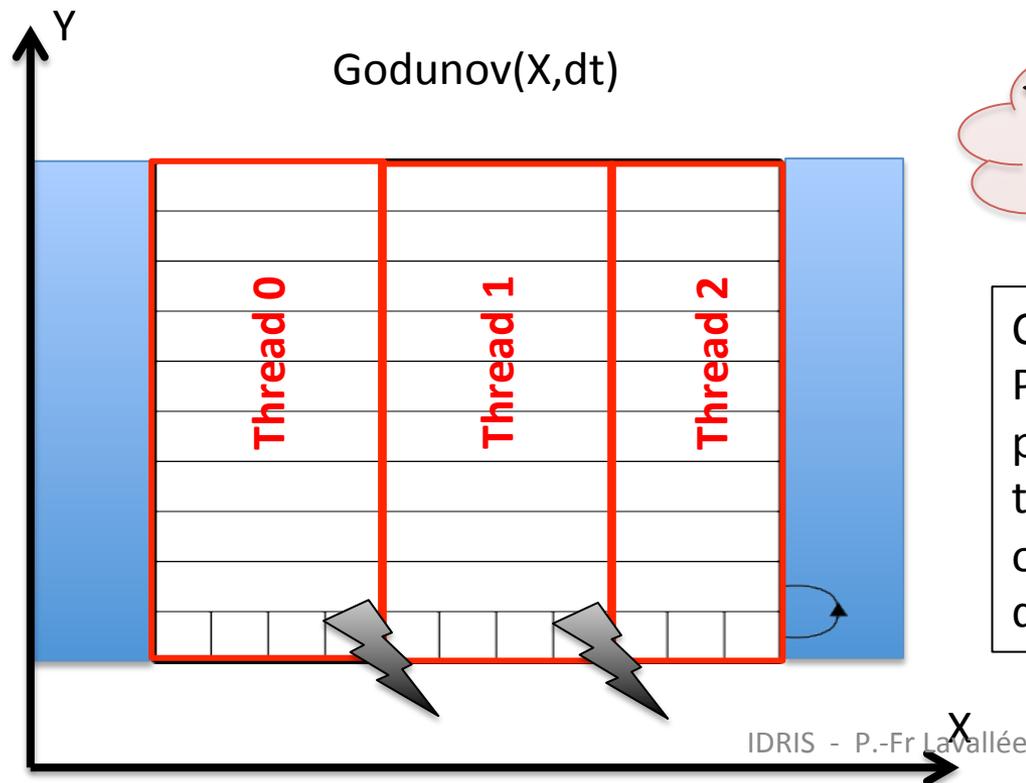


# HYDRO version parallèle hybride



# HYDRO version parallèle hybride

- Attention à la gestion des synchronisations (MPI et OpenMP) pour ne pas casser les dépendances et conserver ainsi la sémantique du code (i.e. obtenir les mêmes résultats que la versions séquentielle au erreurs d'arrondis près...)
- Synchronisation de type exclusion mutuelle des threads pour calculer le pas de temps sur un sous-domaine MPI
- Synchronisation (MPI+OpenMP) avant de commencer la MAJ de uold(:, :, :)
- Synchronisation « fine » OpenMP nécessaire lors de la MAJ du vecteur de travail (ligne ou colonne suivant la direction X ou Y)



Condition de dépendance :  
Pour une ligne donnée, le thread 0 ne peut écrire son résultat que lorsque le thread 1 à fini sa lecture. Au maximum, on ne pourra avoir qu'une ligne de décalage entre les différents threads !