

# Parallélisation d'un code de diffusion de la chaleur en deux dimension

Dufumier & Valade  
Projet d'AMS301

# But du TP

## Code séquentiel

Créer un code séquentiel qui résolve l'équation de la chaleur sur une grille 2D sur un seul processeur en utilisant la méthode des différences finies spatiales et temporelles.

# But du TP

## Code séquentiel

Créer un code séquentiel qui résolve l'équation de la chaleur sur une grille 2D sur un seul processeur en utilisant la méthode des différences finies spatiales et temporelles.

## Code parallèle

Paralleliser ce code sur une grille structurée.

# But du TP

## Code séquentiel

Créer un code séquentiel qui résolve l'équation de la chaleur sur une grille 2D sur un seul processeur en utilisant la méthode des différences finies spatiales et temporelles.

## Code parallèle

Paralleliser ce code sur une grille structurée.

## Outils

Utilisation du langage C et de la bibliothèque MPI. On fera tourner le code sur la machine gin pour étudier la scalabilité faible et forte.

# But du développement séquentiel préalable

## Buts principaux

- ▶ Créer des outils réutilisables dans le code parallèle
- ▶ Avoir une méthodologie de vérification de la justesse du code. On utilise une solution exacte en espace et en temps et on calcul de l'écart de la solution approximée à cette solution exacte.

# But du développement séquentiel préalable

## Buts principaux

- ▶ Créer des outils réutilisables dans le code parallèle
- ▶ Avoir une méthodologie de vérification de la justesse du code. On utilise une solution exacte en espace et en temps et on calcul de l'écart de la solution approximée à cette solution exacte.

## On s'attachera à vérifier que

- ▶ Les arguments sont bien passés en début de fonction
- ▶ La solution exacte est bonne
- ▶ L'équation locale bien implémentée
- ▶ La norme  $L^2$  fonctionne

# Structure du programme

- Récupération des arguments
- Allocation de la mémoire (grilles `read`, `write`, `exact_sol`)
- Calcul de la CFL (sortie si trop grande)
- Initialisation de `read` avec la solution exacte

Pour  $t \in [1, N_{steps}]$

Pour  $i, j \in [1, N_{pts} - 2]^2$

- Mettre à jour la grille `write` au point  $(i, j)$  à partir de `read`

Fin Pour

- Comparer `write` avec `exact_sol` avec la norme  $L^2$
- Échanger les pointeurs de `read` et `write`

Fin Pour

- Afficher les résultats

# Résultats



# Problème et méthode

**On procède à un découpage structuré de la grille en deux dimensions.**

**On dispose des valeurs suivantes**

- ▶ Une grille globale de  $N_x \times N_y$  points
- ▶  $P$  processeurs

# Problème et méthode

**On procède à un découpage structuré de la grille en deux dimensions.**

**On dispose des valeurs suivantes**

- ▶ Une grille globale de  $N_x \times N_y$  points
- ▶  $P$  processeurs

**On cherche à connaître**

- ▶ La répartition des processeurs sur une grille  $P_x \times P_y$
- ▶ La taille de la grille locale  $N_x^p \times N_y^p$  pour chaque grille locale (sauf sur les bords)

# Problème et méthode

**On procède à un découpage structuré de la grille en deux dimensions.**

**On dispose des valeurs suivantes**

- ▶ Une grille globale de  $N_x \times N_y$  points
- ▶  $P$  processeurs

**On cherche à connaître**

- ▶ La répartition des processeurs sur une grille  $P_x \times P_y$
- ▶ La taille de la grille locale  $N_x^p \times N_y^p$  pour chaque grille locale (sauf sur les bords)

## Méthode

- ▶ On choisit de faire que les grilles locales soient homotétiques à la grille globale
- ▶ Pour maximiser le nombre de processeurs utilisés, on ajoute une fonction heuristique

# Constuction de la grille de processeurs

1. Premier calcul des  $P_x, P_y$  :

$$P_x = \left\lfloor \sqrt{P \frac{N_x}{N_y}} \right\rfloor, \quad P_y = \left\lfloor \frac{P}{P_x} \right\rfloor$$

2. Heuristique de modification des  $P_x, P_y$  :

$$P_x, P_y = \underset{i, j \in V}{\operatorname{argmin}} (P - i \times j), \quad V = \{P_x \pm 1, P_y \pm 1\}$$

3. Calcul des  $N_x^p, N_y^p$  :

$$N_x^p = \left\lfloor \frac{N_x}{P_x} \right\rfloor, \quad N_y^p = \left\lfloor \frac{N_y}{P_y} \right\rfloor$$

# Une structure non triviale

## Intérêts

- ▶ Accéder sans difficulté aux informations copiées depuis les processeurs voisins
- ▶ Avoir des contenants facilitant les communications entre processeurs