HPC - M1 CSMI

Convection 2D sur grille régulière

On considère l'équation de convection en deux dimensions sur le carré $[0,1]^2$:

$$\frac{\partial u}{\partial t} + \mathbf{v} \cdot \nabla u = 0,$$

où :

- $u = u(x, y, t)$ est l'inconnue,
- $\mathbf{v} = (v_x, v_y)$ est un champ de vitesse constant. On suppose que les deux composantes de la vitesse sont > 0. Vous pouvez supposer que $v_x, v_y = 1$.

On impose des conditions de Dirichlet homogènes : $u = 0$ sur les bords du domaine gauche et inférieur. La condition initiale est la fonction indicatrice du disque de centre $(1/2, 1/2)$ et de rayon $1/8$.

On discrétise en espace à l'aide d'une grille régulière de taille $n \times n$ (espacement $h = 1/(n-1)$, $(x_i, y_j) = (ih, jh)$, $0 \leq i, j < n$) et en temps par un schéma **implicite** décentré à pas de temps Δt.

Questions :

1. Calculer la solution exacte du problème à l'instant $T = 1/4$.
2. Écrire le schéma de façon détaillée. Dans quel ordre faut il faire les calculs pour passer du pas de temps k au pas de temps $k+1$ sans inverser de matrice ? (indication: partir de la cellule en bas à gauche, puis balayer le maillage selon des diagonales de la grille).
3. Montrer que l'on peut avancer en temps ce schéma en ne stockant qu'un seul vecteur (au lieu de 2 pour le schéma explicite classique).
4. Montrer que le schéma est inconditionnellement stable en norme L^∞. Dans la suite, on prendra $\Delta t = T/100$ indépendamment de la valeur de n.

5. Donner la complexité asymptotique du schéma en fonction de n pour atteindre un temps final $T = 1/4$ fixé (nombre d'opérations flottantes en fonction de n, ne pas compter les opérations de lecture/écriture en mémoire).
6. **(Modèle de temps de calcul)**
   (a) Estimer le temps d'exécution en supposant que tous les accès mémoire sont instantanés (ne comptabiliser que les opérations flottantes). On notera t_c le temps (supposé constant) d'une opération de virgule flottante (addition, multiplication, etc.).
   (b) Refaire l'estimation en supposant que chaque accès mémoire (lecture ou écriture) prend un temps fixe t_s.
   (c) Proposer un modèle simplifié avec cache :
      i. Mémoire principale de grande capacité, accès lents, de temps unitaire t_s.
      ii. Cache unique de taille C, avec des accès rapides, de temps unitaire t_f.
      iii. Décrire comment la présence d'un cache change le temps d'exécution attendu quand le maillage est balayé diagonale par diagonale. Pour simplifier les calculs, on supposera que n est grand devant C (une ligne de la grille ne rentre pas dans le cache). On négligera les variations de performances liées aux bords ainsi qu'au début et à la fin du calcul. On fera l'hypothèse que la mémoire cache est gérée par le système de façon très simple : quand le cache est plein, la dernière donnée lue remplace la plus ancienne.
7. **(Programmation)**
   (a) Implémenter le schéma implicite en `C++`. Comment vérifiez-vous la validité du code ?
   (b) Implémenter une version parallèle du code avec 2 threads OpenMP. Pour cela, lors du balayage des diagonales, attribuer environ une moitié de la diagonale à chaque thread. Expliquez en quoi cette approche est correcte. Mesurer l'accélération apportée par le nouvel algorithme sur des grilles de taille $n = 200$, $n = 400$, $n = 800$. Conclusion ?

Notes:

- Supposer que $v_x, v_y = 1$.
- Vous pouvez utiliser ce code comme point de départ :
  https://irma.math.unistra.fr/~helluy/hpc/grid2d.tgz
  Commande de compilation : `g++ *.cpp`
- Pour observer un **speedup** significatif avec la version parallèle, compilez sans optimisation : `g++ -O0 *.cpp -lm -fopenmp` .
- Vous pouvez supposer que le maillage est assez grand (n=400 ou n=800) pour que le surcoût de création des threads soit négligeable.