Résolution de l'équation de Laplace par méthode de Monte-Carlo

I. Contexte et explication de la méthode de résolution.

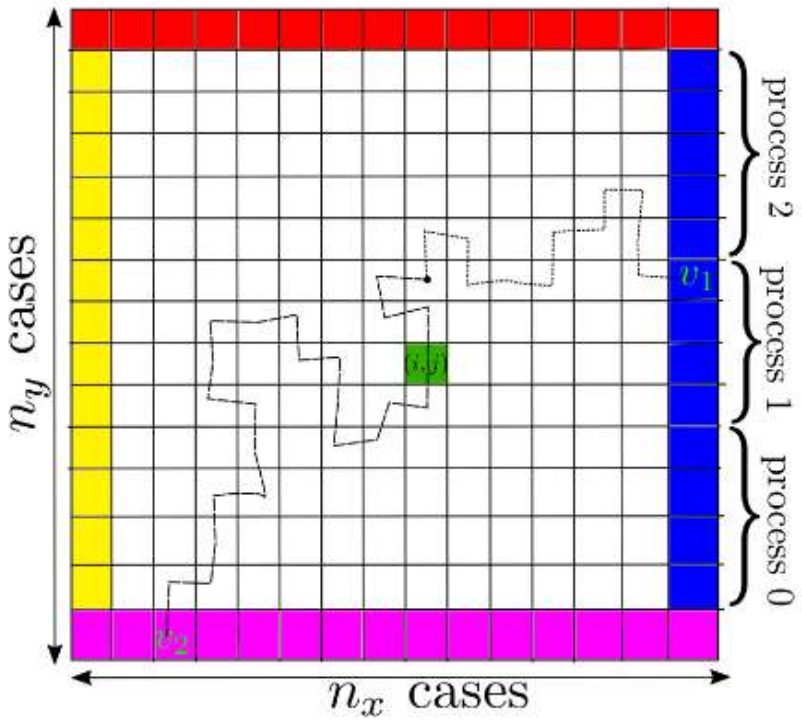
Nous voulons résoudre l'équation de Laplace :

$$\Delta u = 0 \qquad (1)$$

sur une grille de calcul comportant n_x cases selon x et n_y selon y. Chaque case est représentée par un couple d'entiers (i, j), avec $0 \leq i < n_x$ et $0 \leq j < n_y$.

La solution u doit satisfaire des conditions aux bords en valeurs imposées, à savoir u est imposée à une certaine valeur sur les cases du bord.

Nous utilisons une méthode dite stochastique pour approcher la solution de (1) sur, successivement, toutes les cases intérieures de la grille.



Ainsi, pour calculer la valeur u_{ij} approchant u sur la case (i, j), nous simulons la marche aléatoire d'une particule qui, partant de (i, j), peut se déplacer aléatoirement selon une loi uniforme de case en case, jusqu'à atteindre une case du bord correspondant à une valeur imposée de u.

On montre ainsément qu'en répétant ce procédé un grand nombre de fois, la moyenne des valeurs aux bords atteintes par les particules tend vers u_{ij}.

Cet exemple est représentatif de toute une classe de problèmes nécessitant d'exécuter un grand nombre de fois le même calcul mais avec différents jeux de données. C'est typiquement le cas d'algorithmes d'optimisation qui impliquent l'exécution de n simulations numériques différentes et indépendantes où seul va changer le paramètre dont on cherche la valeur minimisant une certaine fonction « coût ».

Nous réalisons un solveur de l'équation de Laplace par méthode de Monte-Carlo. Pour nx=ny=50 et N_tirage=60, nous avec des conditions de bords de Dirichlet visibles sur le tracé la solution suivante :

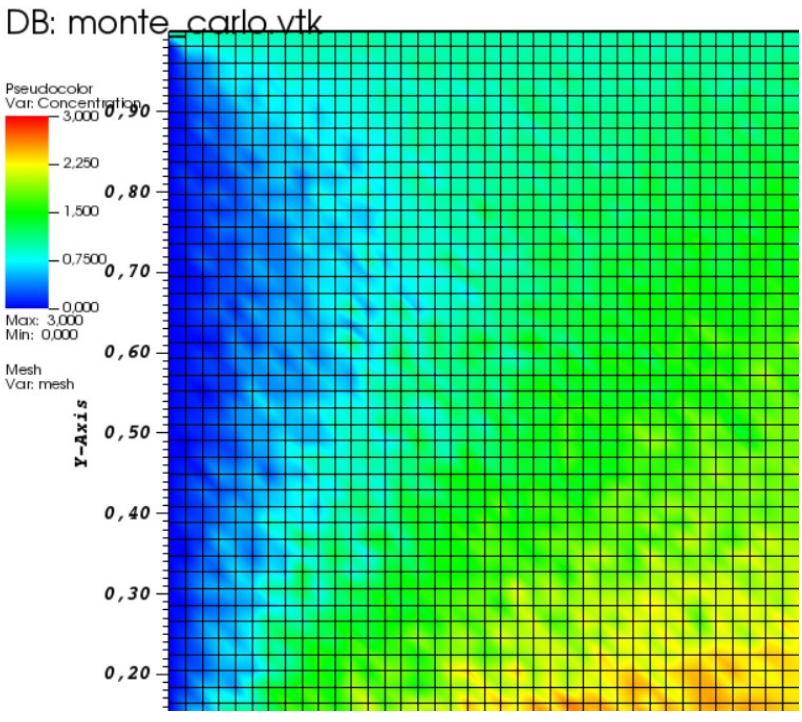


Figure 1 : Tracé de la solution de l'équation de Laplace avec l'algorithme séquentiel.

II. Description de la démarche de parallélisation.

Nous parallélisons l'algorithme en affectant au processus de rang p uniquement un certain nombre de lignes et non toute la grille, comme représenté sur la figure suivante :

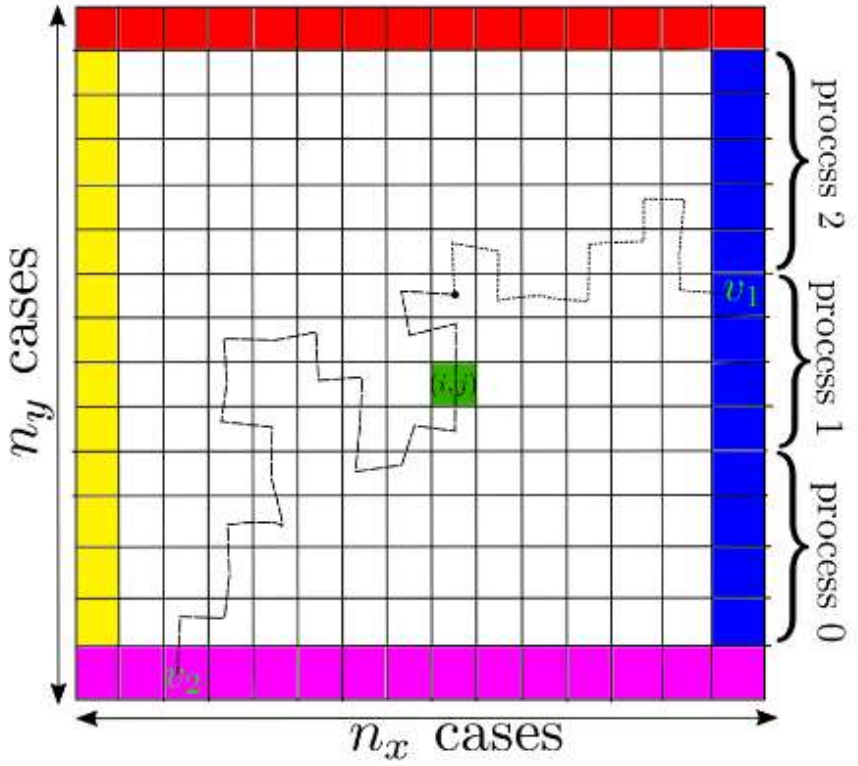


Figure 2 : Schéma représentant la procédure de parallélisation de l'algorithme.

Nous découpons notre grille uniformément en p morceaux et chaque processus se voit attribuer un morceau de même taille ny/p . En passant à la partie entière et en faisant attention aux indices des bords, ce découpage réalise bien ce qui est demandé. Ceci est réalisé par la fonction ***partitionnement*** qui retourne les indices ***deb*** et ***fin*** des lignes prises en charge par le processus.

Chaque processus calcule alors en parallèle son morceau de grille, et le communique ensuite au processus 0. Ce processus 0 se charge alors de reconstituer la grille et de l'écrire sur le fichier graphique.