UNIVERSITÉ D'ORLÉANS **Programmation Parallèle**

M1 Info Année **2019 - 2020**

Série de Travaux Dirigés : **7** *- Remplissage d'un MNT*

Un Modèle Numérique de Terrain (MNT) est une matrice représentant une parcelle de terrain et dont chaque élément représente la hauteur en un point dit géo-référencé du terrain.

Sur un terrain réel, l'eau qui tombe sur le sol s'écoule en grande partie vers le cours d'eau le plus proche. Pour calculer cet écoulement sur un MNT ce n'est pas facile car un terrain est composé d'une multitude de petites cuvettes dans lesquelles l'eau va se stocker.Sur le MNT brut si on souhaite calculer l' écoulement principal vers les cours d'eau, on peut réaliser un calcul à partir du terrain brut et on va ainsi calculer l'écoulement dans les cuvettes avant de pouvoir avoir l'écoulement principal. On peut aussi modifier le MNT pour supprimer ces cuvettes et ensuite utiliser le MNT modifié pour calculer l' écoulement principal.

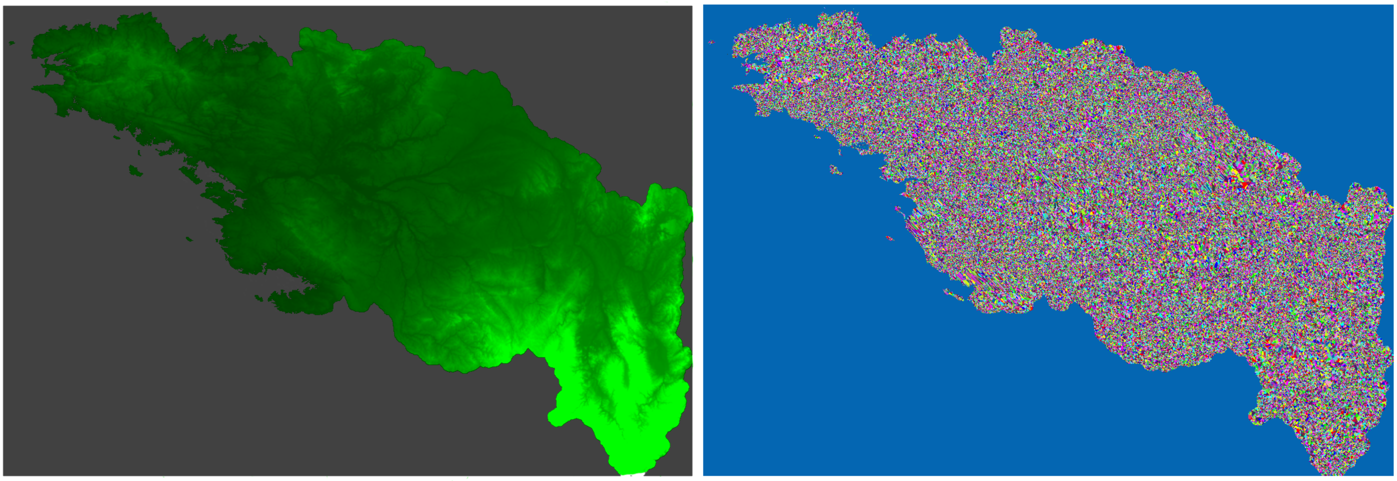

Figure 1: Le bassin Loire-Bretagne (MNT et Cuvettes)

L' intérêt du pré calcul pour remplir les cuvettes est illustré par la figure avec à gauche le MNT sur lequel on voit les cours d'eau et à droite l'ensemble des cuvettes du terrain brut. L'objectif de ce sujet est de paralléliser un algorithme de remplissage des petites cuvettes qui va consister à travailler sur un MNT intermédiaire dans lequel on a tout rempli à partir de la hauteur max du terrain et qu'on va progressivement abaisser vers le MNT initial mais en gardant les cuvettes remplies.

1 Les données

Le MNT se présente sous la forme d'un fichier texte constitué d'une entête suivie des valeurs de hauteur du MNT. L'entête contient les six lignes suivantes qui correspondent aux éléments explicités à droite

```
1025                                                        ncols
1025                                                        nrows
618360                                                      xllcorner
6754408                                                     yllcorner
1                                                           cellsize
-9999                                                       NODATA_VALUE
95.7 98.2579 102.195 104.46 107.86 106.635 105.86
```

Ainsi la première ligne donne le nombre de colonnes du MNT, la seconde donne le nombre de lignes et ces deux nombres peuvent être différents. Les deux lignes suivantes indiquent les coordonnées du coin supérieur gauche mais cette information ne sera pas utilisée comme la taille du point indiquée par la cinquième ligne. Enfin la dernière ligne de l'entête donne la valeur associée à l'absence d'information sur un point du MNT. Les lignes suivantes contiennent les valeurs des hauteurs. La phase initiale va consister à charger ce fichier en mémoire et à le distribuer sur les différents nœuds qui participent aux calculs.

2 L'algorithme séquentiel

A partir du fichier texte il est donc possible de construire une matrice de réels Z de taille $nrows \times ncols$ qui contient toutes les hauteurs du terrain correspondant. Ainsi Z_{ij} est la hauteur du point (i, j) de la parcelle de terrain représenté. L'algorithme de remplissage est un algorithme itératif qui construit une matrice W initialisée par

1. $W_{ij} = Z_{ij}$ pour $i = 0$ ou $j = 0$ ou $i = nrows - 1$ ou $j = ncols - 1$ (donc sur les bords)

2. $W_{ij} = Max$ sinon où Max désigne la plus haute hauteur du terrain

Les itérations peuvent se décrire par le pseudo code décrit par l'algorithme 1 où ϵ est une valeur représentant une petite pente pour l'écoulement de l'eau. Notons que pour effectuer ce calcul, il est nécessaire d'utiliser 3 matrices. La matrice Z du MNT initial et deux matrices W^{t-1} et W^t pour respectivement le remplissage calculé au temps précédent et pour le calcul courant.

Algorithm 1 Algorithme de Darboux pour le remplissage des cuvettes d'un MNT

modification=true
while modification **do** modification=false
    for all (i,j), $0 \leq i \leq nrows, 0 \leq j \leq ncols$ **do**
        if $W^{t-1}_{ij} > Z_{ij}$ **then**
            for all 8 neighbors $((i + k_1), (j + k_2)) = (n_1, n_2)$ $0 \leq k_1, k_2 \leq 1$ **do**
                if $Z_{n_1 n_2} \geq W^{t-1}_{n_1 n_2} + \epsilon$ **then**
                    $W^t_{ij} = Z_{ij}$
                    modification=true
                else if $W^{t-1}_{ij} > W^{t-1}_{n_1 n_2} + \epsilon$ **then**
                    $W^t_{ij} = W^{t-1}_{n_1 n_2} + \epsilon$
                    modification=true
                end if
            end for
        else
            $W^t_{ij} = W^{t-1}_{ij}$
        end if
    end for
end while

Exercice 1. Version OpenMP

A partir de l'analyse du programme séquentiel et des parties coûteuses des calculs proposez une version utilisant des directives OpenMP.

Étudiez l'accélération obtenue par rapport à la version séquentielle.

Exercice 2. Version MPI

Dorénavant vous souhaitez travailler sur un très large terrain dont les données doivent être distribuées pour pouvoir effectuer le calcul. Proposez une nouvelle version pour les architectures à mémoire distribuée avec MPI.

Avant de commencer il est utile de réfléchir aux points suivants

1. Si le fichier contenant le MNT initial est disponible sur un processeur *root* et un seul, comment peut-on distribuer le MNT initial sur l'ensemble des processeurs ? Est-il possible d'utiliser *MPI_Scatterv* en considérant qu'on souhaite travailler sur des MNT de très grande taille ?

2. En considérant uniquement la matrice Z comment va-t-on distribuer les données ? Est ce que les distribuer par bloc a un intérêt ? Et si on suppose qu'on a *nprocs* processeurs quelle sera la taille des données distribuées sur chaque processeur ?

3. Si on suppose la partie distribution terminée, est ce que l'algorithme nécessite des phases de communication entre les itérations ? Si oui, comment seront-elles effectuées ?

4. L'algorithme s'arrête lorsque au cours d'une itération aucune valeur de W^{t-1} n'a été modifiée ($W^t = W^{t-1}$). Comment gérer cette condition lorsque le calcul est réparti sur les *nprocs* processeurs ?