INFORMATIQUE : LANGAGE C - 1A - EOST - 2012/2013

TP – Multi-Resolution Grid

Application du langage C à la tomographie sismique :
construction d'une paramétrisation du modèle
adaptée à la couverture spatiale des données

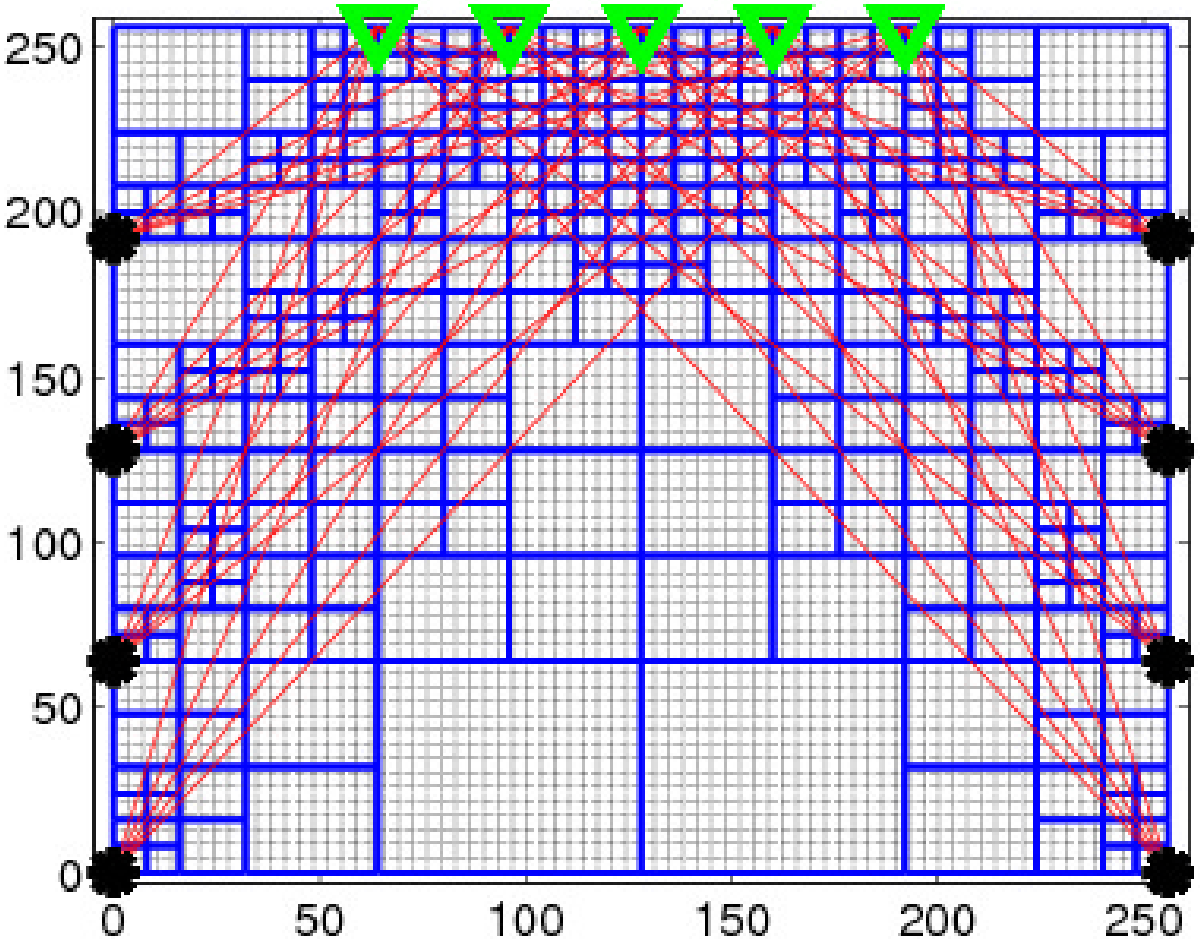

FIGURE 1 – Multi-Resolution Grid

```c
/* Multi Resolution Grid (MRG) : application to seismic tomography*/
/*                        1A EOST                                 */
/*                 Christophe Zaroli (2012)                       */
//--------------------------------------------------------------------------
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<time.h>
#define BUFFMAX 80              // taille buffer lecture fichier donnees
//--------------------------------------------------------------------------
/* structure noeud pour creation arbre */
typedef struct noeud
       {
        int x; //abscisse coin bas gauche de la cellule
        int y; //ordonnee coin bas gauche de la cellule
        struct noeud *fils1; //subdivision sens trigo
        struct noeud *fils2;
        struct noeud *fils3;
        struct noeud *fils4;
       }Noeud;
//--------------------------------------------------------------------------
/* variables globales */
int g_N;
int g_M;
int g_l; // l=2^N : largeur de la cellule initiale (unite=pixel)
int g_L; // L=2^M : longeur de la cellule initiale (unite=pixel)
int g_nb;  // nb=l*L
int *g_tab_ray; // tableau des nombres de rais par pixel
int g_seuil; // seuil pour subdivision d'une cellule
int g_pmax; // profondeur de l'arbre maximale : min(N,M)
//--------------------------------------------------------------------------
/* prototypes des fonctions */
void aff_arbre(Noeud *, int);
void maj_tab_ray(int, int, int, int);
float projection_x(int , int , int, int , int );
float projection_y(int , int , int, int , int );
float ProduitScalaire(float [2], float [2]);
void lecture_data( char *);
int densite (int , int , int );
Noeud * allouerNoeud(int , int );
void developper(Noeud *, int );
//--------------------------------------------------------------------------
/* abscisse de la projection d'un point sur la droite (rai) */
float projection_x(int a, int b, int c, int x, int y)
{
// *** A COMPLETER ***
}
//--------------------------------------------------------------------------
/* ordonnee de la projection d'un point sur la droite (rai) */

float projection_y(int a, int b, int c, int x, int y)
{
// *** A COMPLETER ***
}
//--------------------------------------------------------------------------
/* produit scalaire entre deux vecteurs de dim 2 */

float ProduitScalaire(float X[2], float Y[2])
{
// *** A COMPLETER ***
}
//--------------------------------------------------------------------------
/* lecture des donnees du fichier en entree (couples source-station) */
/* et affectation des variables globales g_N et g_M, et construction du tableau g_tab_ray */
void lecture_data( char * filename)
{
// *** A COMPLETER ***
}
//--------------------------------------------------------------------------
/* mise a jour du tableau global g_tab_ray */

void maj_tab_ray(int x_s, int y_s, int x_r, int y_r)
```



```c
{
// *** A COMPLETER ***
}
//--------------------------------------------------------------------------
/* Calcule et retourne le nombre de rais (densite) passant par une cellule definie
 par son point bas gauche (x,y) et le niveau de profondeur (p) dans l'arbre */

int densite (int x, int y, int p)
{
// *** A COMPLETER ***
}
//--------------------------------------------------------------------------
/* alloue un nouveau noeud et initialise les coordonnes du point bas gauche de la cellule */

Noeud * allouerNoeud(int x, int y)
{
// *** A COMPLETER ***
}
//--------------------------------------------------------------------------
/* construit l'arbre de maillage de la cellule definie par le noeud a et par la profondeur p */
void developper(Noeud *a, int p)
{
// *** A COMPLETER ***
}
//--------------------------------------------------------------------------
/* affichage des feuilles de l'arbre */
void aff_arbre(Noeud *a, int niv)
{
// *** A COMPLETER ***
}
//--------------------------------------------------------------------------
// programme principal
void main(int argc, char **argv)
{
Noeud *arbre=allouerNoeud(0,0); // arbre initial correspondant a une seule cellule
g_seuil = atoi(argv[2]);
lecture_data(argv[1]);
developper(arbre,0);
aff_arbre(arbre,0);
}
//--------------------------------------------------------------------------
```