Projet individuel IPF ENSIIE-2A 2016-2017

1. Informations générales

1. Travail à rendre

Le projet est à réaliser individuellement. Il sera accompagné d'un dossier décrivant l'ensemble des choix faits, la description des types et des fonctions. Le dossier devra également fournir des cas de tests accompagnés des résultats attendus et ceux effectivement obtenus.

Le projet (y compris le code) ainsi que le rapport sont à rendre le jour de la soutenance. La date des soutenances est encore à définir.

2. Enoncé du projet : Quadtrees

Un **quadtree** est une structure de données arborescente où chaque noeud a 4 fils. Elle est couramment utilisée pour partitionner un espace bidimensionnel en subdivisant l'espace récursivement en 4 zones appelées quadrants (Nord-Ouest, Nord-Est, Sud-Ouest, Sud-Est). Les quadtrees sont utilisés pour représenter des images, ce qui est le contexte de ce projet.

Le projet consiste en la création et la manipulation de variantes de quadtrees. On étudiera le moyen de représenter des points dans une image, de représenter une image composée de pixels et enfin une collection de rectangles.

3. Des points dans une image

On utilise ici la notion de Point Quadtree, abrégée dans la suite en Pquadtree. Un Pquadtree est soit vide, soit composée d'une racine contenant un nombre fixé de points (ici 1 point) et 4 quadrants qui sont eux-mêmes des Pquadtrees. Pour plus de facilité, dans un premier temps, on associe également à chaque quadrant la surface qu'il occupe, appelée support dans la suite. Le support est un rectangle de manière générale, un carré ici.

Au départ, le quadtree couvre un rectangle de base, appelé support ou feuille, de taille une puissance de 2. L'insertion du premier point partitionnera donc ce support en 4 quadrants d'aires identiques. On travaille dans le plan muni d'un repère et donc de deux axes : l'axe des x et l'axe des y. Et plus particulièrement dans un rectangle englobant : un carré de côté 2^n.

Un rectangle est caractérisé par les 4 valeurs suivantes :

- top (coordonnée en y de l'arête supérieure)
- bottom (coordonnéee en y de l'arete inférieure)
- left (coordonnée en x de l'arete de gauche)
- right (coordonnée en x de l'arete de droite)

Le rectangle englobant, la feuille, est telle que top vaut 2^n, bottom vaut 0, left vaut 0 et right vaut 2^n. On pourra utiliser un type enregistrement pour définir le type des rectangles (voir le cours en ligne fichier enregistrement.pdf) ou un quadruplet. Dans la suite ce type est nommé rect.

```
type pquadtree =
PEmpty
| PNoeud of point*rect*pquadtree*pquadtree*pquadtree*pquadtree
```

NPnoeud(p, r, q1, q2, q3, q4) représente le Pquadtree contenant le point p couvrant le support défini par r, et subdivisée en 4 quadrants (qui sont des Pquadtrees) q1, q2, q3 et q4 (dans l'ordre le cadrant NO, NE, SO, SE).
Deux opérations nous intéressent particulièrement : la recherche d'un point et l'insertion d'un rectangle dans un Pquadtree.

Interrogation

Il s'agit de déterminer si un point (décrit par ses coordonnées) est présent dans une image représentée par un Pquadtree.

Q1 : Ecrire une fonction **pappartient** qui retourne **true** si le point est présent et **false** sinon.
Q2 : Ecrire une fonction **pchemin** qui retourne le chemin pour atteindre ce point s'il est présent et échoue sinon. Ce chemin prendra la forme d'une liste de noms de quadrants. Par exemple, [NO;SO;SE] qui indique que ce point se trouve dans le quadrant SE du quadrant SO du quadrant NO de la feuille initiale.

Insertion d'un point

Un point est inséré dans le quadrant approprié. L'insertion d'un point dans un quadrant vide provoque sa subdivision en 4 parties égales. Et ce à tout niveau. Par exemple, la figure 1 illustre les Pquadtrees obtenus après l'insertion successive des points P1,P2,P3 et P4.

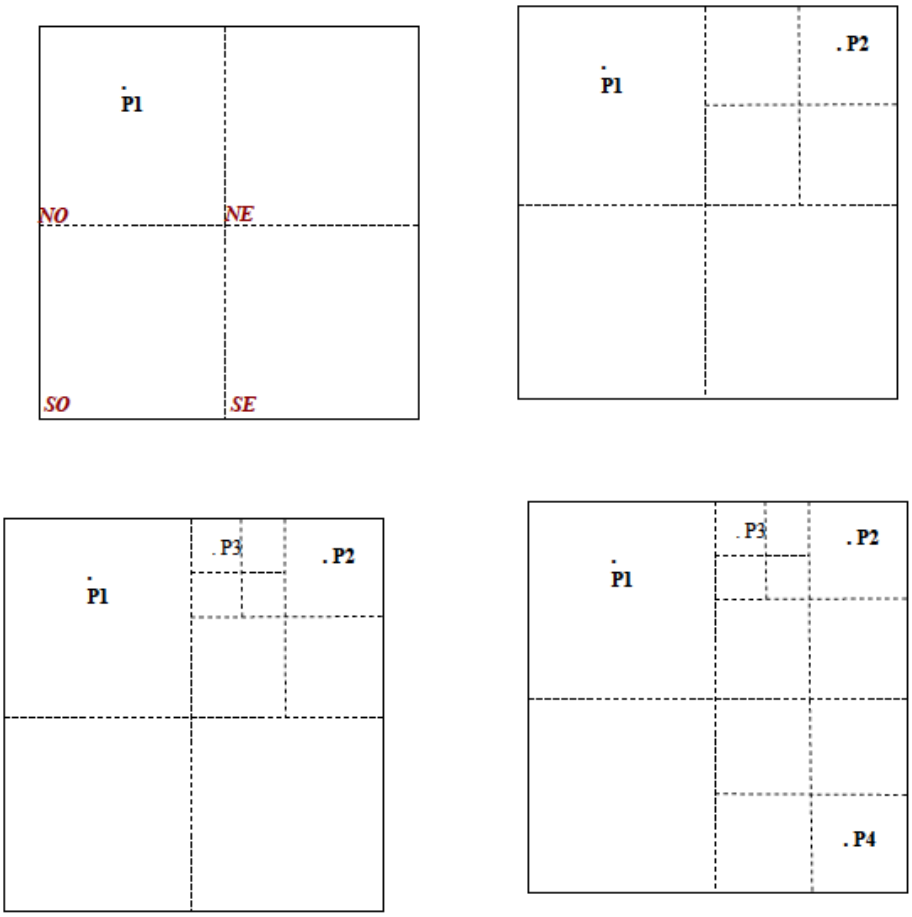

Figure 1 : Pquadtrees après l'insertion successive des points P1,P2,P3 et P4.

Q3: Ecrire une fonction **insère** qui insère un point P dans un Pquadtree q. Elle retourne un nouveau

Pquadtree contenant tous les points de q ainsi que P.
Q4 : Construire une scène par ajouts successifs de points. Montrer par un ou plusieurs jeux de test que la forme du Pquadtree dépend de l'ordre d'insertion des points.

Visualisation
Q5 : En utilisant la bibliothèque graphique de Ocaml, écrire une fonction qui permet de visualiser graphiquement un Pquadtree. Vous dessinerez les bords du support du quadtree ainsi que les bords de tous les quadrants qui le composent (on dessinera les carrés supports).

4. Représentation de régions uniformes

Dans cette partie, on veut manipuler des images constituées de pixels coloriés disposés sur une surface carrée dont le côté a 2^n pixels où n est un entier naturel. Ici on ne considèrera que deux couleurs, noir et blanc.

A nouveau, on adoptera une vision hiérarchique des images : une image est soit de couleur uniforme, soit formée de 4 quadrants NO, NE, SO et SE. Une image 2^n x 2^n est ainsi subdivisée itérativement en quadrants.

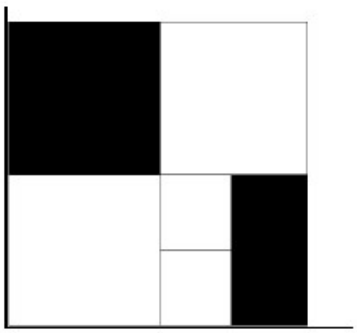

Figure 2 : Une image composite

Par exemple, l'image de la figure 2 peut se décomposer en un Rquadtree comportant dans l'ordre (Nord-Ouest, Nord-Est, Sud-Ouest, Sud-Est) : une image noire, une image blanche, un quadtree comportant dans l'ordre une image blanche, une image noire, une image blanche, une image noire. Selon ce schéma, les images sont représentées par des Rquadtrees du type suivant :

type couleur = Blanc | Noir ;;
type rquadtree = Uni of couleur | RQ of rquadtree * rquadtree * rquadtree * rquadtree;;

Visualisation

Q6 : En utilisant à nouveau la bibliothèque graphique, écrire une fonction qui permet de visualiser un Rquadtree. Cette fois, la fonction prendra en paramètre les dimensions du support (2^n). Les dimensions des supports des quadrants seront calculées par la fonction.

Manipulation d'images représentées par des Rquadtrees

Q7 : Ecrire une fonction qui réalise l'inverse vidéo d'une image. Elle prend un Rquadtree et retourne un Rquadtree obtenu en remplaçant les pixels noirs par des pixels blancs et vice-versa.

Q8 : Ecrire une fonction intersection qui calcule l'image intersection de deux images. Les pixels noirs de l'image intersection sont noirs sur les 2 images.

Q9 : Ecrire une fonction union qui calcule l'image union de deux images. Les pixels noirs de l'image résultante sont noirs sur l'une et l'autre des deux images. L'union peut donner lieu à des valeurs de la forme RQ(Uni Noir, Uni Noir, Uni, Noir, Uni Noir) qu'il conviendra de normaliser en le Rquadtree Uni Noir.

Q10 : Ecrire une fonction qui réalise une symétrie par rapport à la médiane verticale et une fonction qui réalise une symétrie par rapport à la médiane horizontale.

Codage et décodage

Dans le but de sauvegarder les images dans des fichiers, on se pose le problème de transformer un Rquadtree en une liste de 0 et de 1, ainsi que le problème réciproque. Ici on ne s'intéresse qu'au codage et au décodage de l'image ; l'écriture dans un fichier n'est pas traitée. Le codage se fait en réalisant un parcours infixe du Rquadtree. Tout noeud interne (RQ) est codé par un 0, toute feuille (Uni) est codé par un 1. Chaque 1 est suivi du codage de la couleur : 0 pour Blanc, 1 pour Noir.
Q11 : Ecrire une fonction **coder** qui prend un Rquadtree et le transforme en une liste de bits codée comme une chaîne de caractères ou une liste de valeurs du type bit défini par type bit = Zero | Un.
Q12 : Ecrire la fonction inverse **decoder** qui prend en paramètre une chaîne de bits et retourne le Rquadtree correspondant (et échoue si la chaîne de bits n'est pas bien formée).

5. Représentation d'une collection de rectangles

Cette partie est une extension de la première (Pquadtrees) et s'intéresse à la représentation d'un ensemble de rectangles dans une feuille. La collection de rectangles est représentée par une structure de quadtree similaire à celle des Pquadtrees, appelée quadtree dans cette partie.

Un rectangle est représenté comme dans la première partie, par ses 4 coordonnées (left, right, top, bottom); on réutilise pour cela le type **rect**.

Un quadtree peut être vide, ou contenir 4 quadrants eux-mêmes quadtrees. Il contient également la description de son rectangle support ainsi que la liste des rectangles qui traversent une de ses médianes. Attention, les points qui se trouvent sur une médiane ne sont dans aucun des quadrants.

Un quadtree est représenté par le type quadtree défini par :
type quadtree =
Empty | Q of rect * (rect list) * (rect list) * quadtree * quadtree * quadtree * quadtree;;

Ainsi un quadtree non vide est de la forme **Q(s, lv, lh, q1, q2, q3, q4)**
avec :

- s, la description du support (sous forme d'une valeur de type rect),
- lv, la liste des rectangles qui contiennent des points de la médiane verticale du support s,
- lh, la liste des rectangles qui contiennent des points de la médiane horizontale du support s,
- q1, q2, q3 et q4, respectivement les quadtrees NE, NO, SE, et SO.

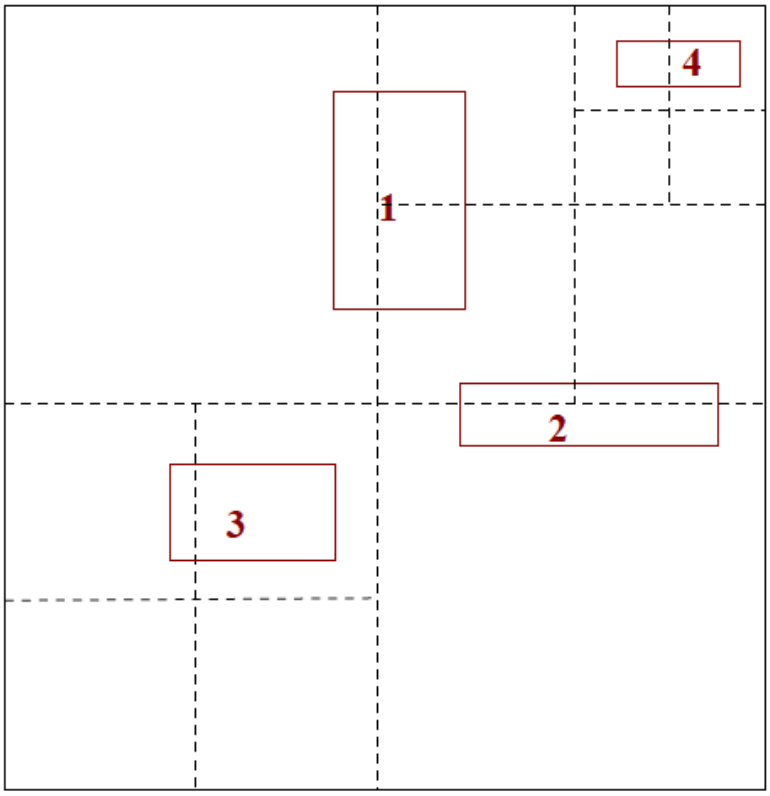

Figure 3 : Un quadtree de rectangles

Le quadtree représenté dans la figure 3 a pour support le carré le plus grand, il contient directement les rectangles 1 et 2 (i.e. le rectangle 1 dans sa propre liste lv et le rectangle 2 dans sa liste lh). Son quadrant SO contient le rectangle 3 (dans sa liste lv) et le quadrant NE – NE contient le rectangle 4 (dans la liste lv). Les rectangles ont été insérés dans l'ordre 1,2,3 et 4.

Visualisation

Q13 : En utilisant à nouveau la bibilothèque graphique, écrire une fonction qui permet de visualiser un quadtree de rectangles.

Insertion d'un rectangle dans un quadtree

Q14 : Soit un quadtree couvrant une feuille de côté 2^n contenant une collection de rectangles. On veut écrire une opération qui insère un nouveau rectangle décrit par ses 4 coordonnées.
L'insertion suit le principe suivant :
Un rectangle est inséré dans la liste lh ou lv associéees au quadtree dont le support a une médiane qui traverse le rectangle. Si le rectangle contient des points d'une des médianes du support du quadtree, il est inséré dans une des deux listes lv ou lh. Sinon, il s'agit de choisir le quadtree NO, NE, SO ou SE qui convient et de l'insérer récursivement dans ce quadtree. Voir la figure 4 qui illustre étape par étape la construction du quadtree de la figure 3 par insertion successive des rectangles 1, 2, 3 et 4.

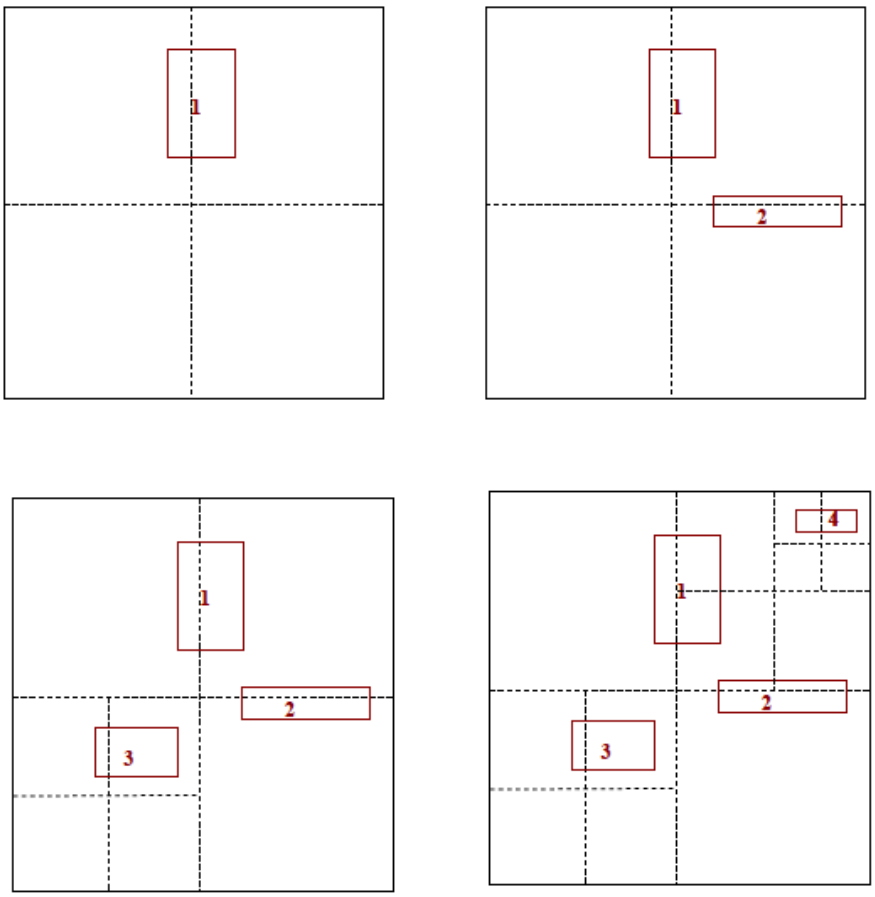

Figure 4 : Etapes de construction

Pour réaliser cet algorithme, il conviendra de définir un certain nombre de fonctions dites géométriques comme par exemple tester si un point de coordonnées x et y est dans le rectangle ou non, calculer l'intersection avec une des médianes etc.

Q15 : Construire une scène par ajouts successifs de rectangles.

Interrogation

Il s'agit ici de déterminer pour un point donné P à quels rectangles d'un quadtree il appartient.

Q16 : Ecrire une fonction qui prend en paramètre un point P représenté par son couple de coordonnées (x,y) et un quadtree q. On veut retourner la liste des rectangles contenus dans q qui contiennent le point P. La recherche de ce point consiste à d'abord collecter les rectangles de lv et de lh qui contiennet P, puis de continuer récursivement avec chacun des quadrants de Q.