Projet individuel d'algorithmique-programmation
AP1 :
groupe 1.1

octobre 2010

1 Informations générales

1.1 Travail à rendre

Le projet est à réaliser en OCaml **individuellement**. Il sera accompagné d'un **dossier** contenant impérativement la description des choix faits, la description des types et des fonctions. Pour chaque fonction, on donnera impérativement l'interface complète (dans le code en commentaire et dans le rapport pour les fonctions présentées).

Même si le sujet est décomposé en questions, il est possible qu'une question se résolve par l'écriture d'une ou plusieurs fonctions intermédiaires. Celles-ci doivent comporter une interface également.

Le dossier fournira également des cas de tests accompagnés des résultats attendus et retournés.

Sur le site du cours figure un petit document sur ce que l'on attend dans un rapport. Consultez-le !

1.2 Calendrier et procédure de remise

Le projet (dossier + copie du listing de code) est à rendre au secrétariat **le 22 novembre à 16h au plus tard**. Le numéro du groupe, ainsi que le nom du chargé de TD, devront figurer en gros, et en rouge sur la page de garde.

Le fichier .ml contenant votre code devra être déposé électroniquement au plus tard **le 22 novembre à minuit**.

Les soutenances seront organisées dés la semaine suivante. Il vous faudra consulter les panneaux d'affichage et votre courrier electronique pour obtenir l'ordre de passage.

Enfin n'attendez pas pour vous mettre au travail ! Un projet se travaille dès la remise du sujet afin d'avoir le temps de laisser murir la solution et de poser des questions au client (dans votre cas, votre chargé de TP).

1.3 Procédure de dépot

Pour déposer le code du projet, consultez les informations sur les machines de l'école, tout figure.

Ne vous inquiétez pas, l'interface fabrique un nom de projet déposé à partir de votre login. Votre projet ne sera pas confondu avec celui d'un autre.

Indiquez quand même en commentaire dans votre fichier de code Ocaml votre nom et votre groupe. Ce sera plus facile pour le correcteur.

Vous pouvez déposer successivement plusieurs versions, le dernier dépot écrase le précédent, seul le dernier dépot est pris en compte

Enfin tout cela peut se faire de l'extérieur.

Le code à déposer est un fichier.ml commenté **avec les interfaces des fonctions**. **Le fichier doit pouvoir être compilé (sans aucune intervention humaine - lecture du buffer complet) sur les machines Yaka de l'école**, n'oubliez pas de vérifier que tout fonctionne sur les machines de l'école avant de le dép oser - si vous l'avez développé sous un autre système d'exploitation.

2 Énoncé 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 bidimentionnel en subdivisant l'espace récursivement en 4 zones appelées quadrants (quadrant Nord-Ouest NO, quadrant NE, SO et SE)

Les quadtrees sont en particulier 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 une adaptation de la notion d'arbre binaire pour représenter des points dans un espace à deux dimensions.

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, appelée 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'arete supérieure)
- bottom (coordonnée 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 subdivisé 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.

Question 1

Écrire une fonction `pappartient` qui retourne true si le point est présent et false sinon.

Question 2

Écrire une fonction `pchemin` qui retourne le chemin pour atteindre ce point si 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.

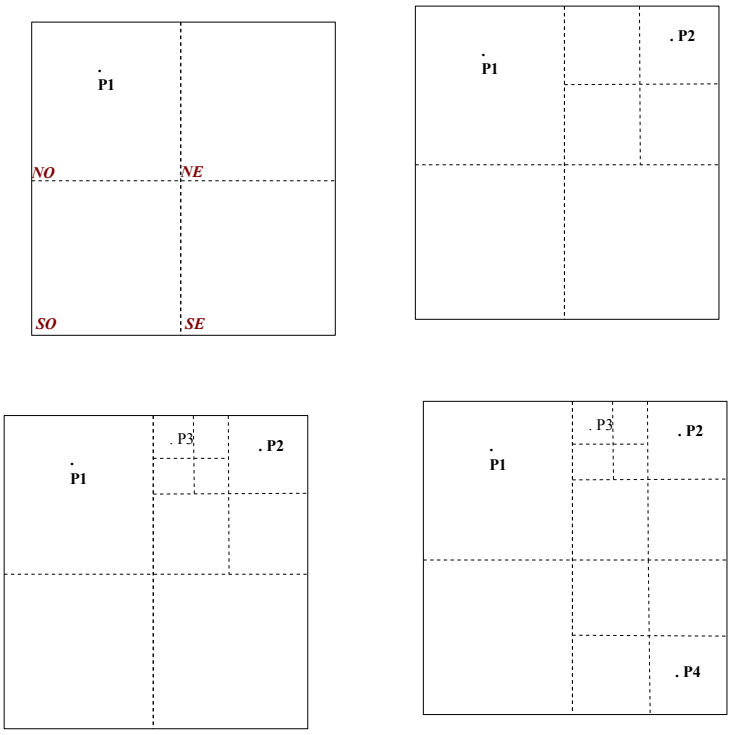

FIGURE 1 – Pquadtrees àprès l'insertion successive des points P1, P2, P3 et P4

Question 3

Écrire une fonction `insère` qui insère un point P dans un Pquadtree q. Elle retourne un nouveau Pquadtree contenant les points de q et P.

Question 4

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

Question 5

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).

Des notes concernant la manipulation du mode graphique et des fonctions nécessaires de la bibliothèque seront publiées ultérieurement. Vous pouvez également consulter l'ouvrage *Apprentissage de la programmation avec OCaml, C. Dubois, V. Ménissier, Hermès, 2004*, présent à la bibliothèque.

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 considérera 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 $2^n \times 2^n$ est ainsi subdivisée itérativement en quadrants.

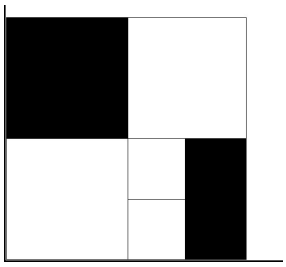

FIGURE 2 – Une image composite

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, 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

Question 6

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

Question 7

Écrire 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.

Question 8

Écrire une fonction `intersection` qui calcule l'image intersection de deux images. Les pixels noirs de l'image intersection sont noirs sur les deux images.

Question 9

Écrire une fonction `union` qui calcule l'image union de deux images. Les pixels noirs de l'image résultante sont noirs sur l'une ou 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`.

Question 10

Écrire 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 nœud interne (RQ) est codé par un 0, toute feuille (Uni) est codée par un 1. Chaque 1 est suivi du codage de la couleur : 0 pour Blanc, 1 pour Noir.

Question 11
Écrire 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`.

Question 12
Écrire la fonction inverse `décoder` qui prend en paramère 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 représenttaion 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 patie, par ses 4 coordonnées (left, right, top, bottom), le type `rect` est encore utilisé ici.

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.

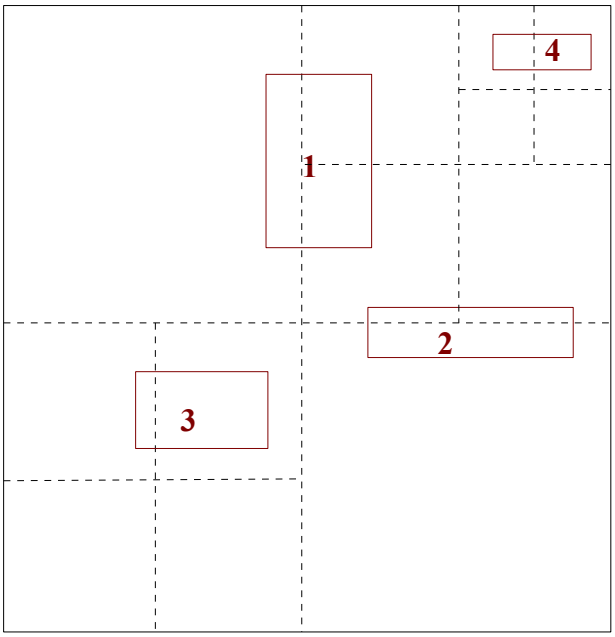

FIGURE 3 – Un quadtree de rectangles

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

Visualisation

Question 13

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

Insertion d'un rectangle dans un quadtree

Question 14

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 associées 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.

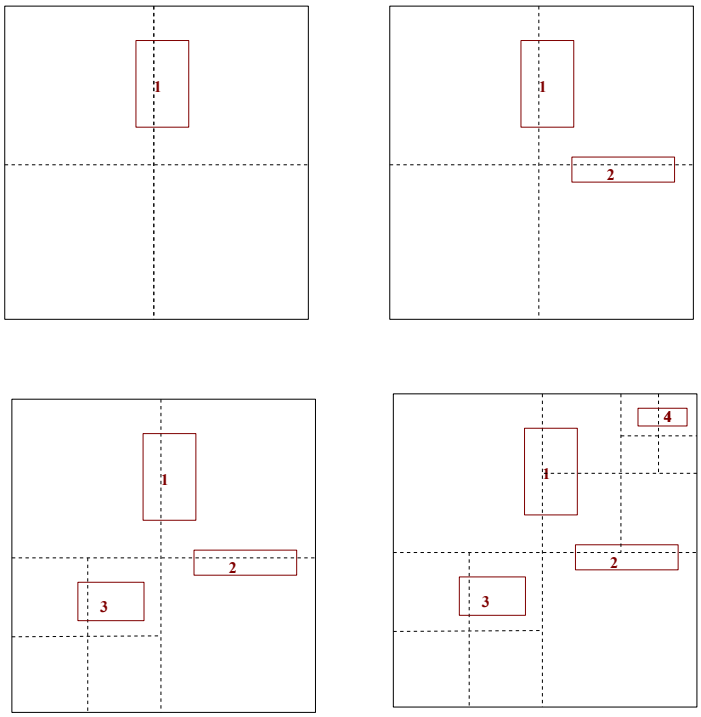

FIGURE 4 –

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.

Question 15
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. Par exemple dans la figure 5, le point P appartient aux rectangles 1 et 5.

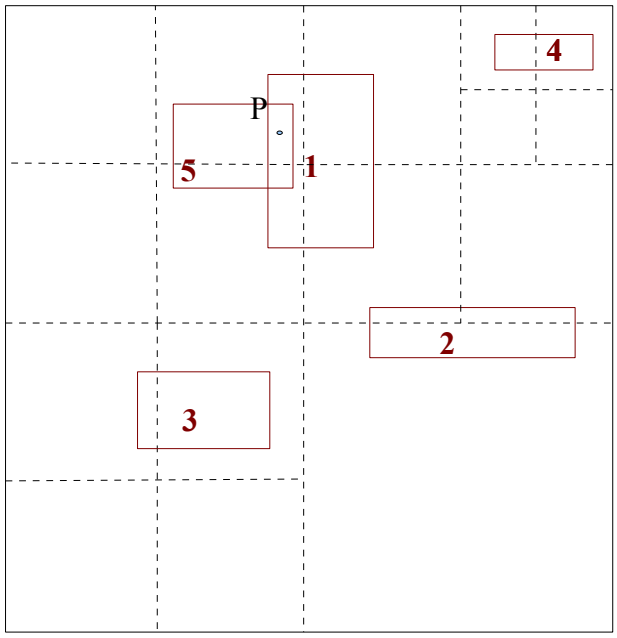

FIGURE 5 –

Question 16

Écrire 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 lh qui contiennent P puis de continuer récursivement avec chacun des quadrants de Q.