amU Faculté des sciences Aix Marseille Université

Année 2024-25

Domaine Sciences et Technologies
Licence mathématiques-informatique
Compilation : TP 2
Codes UE : SIN6U64L
TP 2 : ANTLR (suite)

1 Introduction

Dans ce TP, on va implémenter les grammaires (légèrement modifiées) du TD 2. Vous utiliserez pour cela le dépôt que vous avez forké pour le TP 1 dans lequel vous rajouterez un package `matrix` pour la première partie du TP et un package `quadtree` pour la deuxième partie du TP.

2 Matrices

On considère des matrices d'entiers, c'est-à-dire un tableau bidimensionnel d'entiers avec des lignes et des colonnes. Un tel tableau peut être représenté sous la forme d'une expression parenthésée. L'expression `(1,2,3),(4,5),(),(6,7,8)` par exemple, représente une matrice dont la première ligne est composée des entiers `1`, `2` et `3`. Lorsqu'une ligne se termine par des `0`, ceux-ci peuvent être omis, comme c'est le cas pour la seconde ligne de notre matrice, qui est composée des chiffres `4`, `5` et `0`. Si une ligne n'est composée que de 0, alors elle est représentée par une paire de parenthèses, comme c'est le cas pour la troisième ligne de notre exemple. Afin que chaque représentation corresponde à une seule matrice, on considérera que la longueur de la suite d'entiers la plus longue correspondant à une ligne donne le nombre de colonnes de la matrice. Par exemple, la représentation $(0,0,0),(),()$ correspond à la matrice suivante :

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

On acceptera que les matrices ayant au moins une ligne (pas de mot vide). Par contre, une matrice n'ayant que des lignes vides est possible.

2.1 Questions

1. Écrivez un fichier `Matrix.g4` qui permet de faire l'analyse des représentations de matrices telles que décrites ci-dessous.
2. Utilisez soit un attribut synthétisé ou bien un visiteur pour calculer le nombre de **lignes** d'une matrice à partir de sa représentation.
3. Utilisez soit un attribut synthétisé ou bien un visiteur pour calculer le nombre de **colonnes** d'une matrice à partir de sa représentation.
4. Utilisez soit des attributs synthétisés ou bien un visiteur pour construire un tableau de tableaux d'entiers (de type `int [][]`) correspondant à la matrice (la case `t[i][j]` du tableau `t` retourné devant correspondre à la valeur à la ligne i et à la colonne j de la matrice).

3 Quadtree

On s'intéresse ici à la représentation d'images par des *quadtrees*, utilisés pour l'analyse, la compression et la synthèse d'images. Une image est dite *simple* si elle est de couleur uniforme. Étant donné un medium graphique (écran, imprimante, etc.) carré, qui mesure 1024×1024 pixels, on peut représenter une image par une structure d'arbre, appelé *quadtree*, dans laquelle chaque nœud a exactement 0 ou 4 fils. L'idée de base est la suivante : si une image associée à un nœud n'est pas simple, on la découpe en 4 morceaux de même taille et les fils du nœud correspondant sont les *quadtrees* associés aux 4 sous-images. Si au contraire l'image est simple, alors elle est caractérisée par une information unique, sa couleur ; dans ce cas, le nœud porte cette information et n'a pas de fils. Par exemple, si on parcourt les 4 quarts d'une image en décrivant un Z (donc en commençant par le quart en haut à gauche, puis le quart en haut à droite, puis le quart en bas à gauche pour finir avec le quart en bas à droite), et si on suppose que les dix régions en lesquelles on a découpé le carré de la figure 1 sont simples, de couleurs `r`, `g` et `b`, alors cette figure peut être représentée par le *quadtree* de la figure 2. Un *quadtree* peut alors être représenté sous la forme d'une expression parenthésée qui décrit la structure de l'arbre correspondant. Le *quadtree* de la figure 2 est représenté sous la forme suivante : `(rbg((bbrg)grb))`

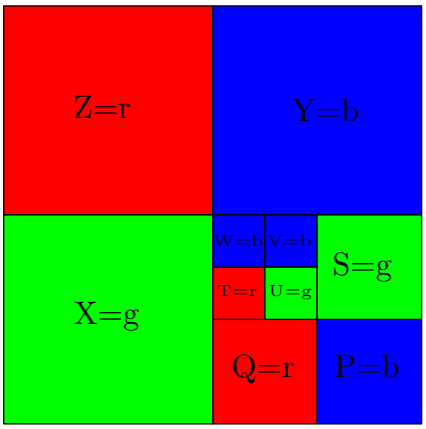


Figure 1

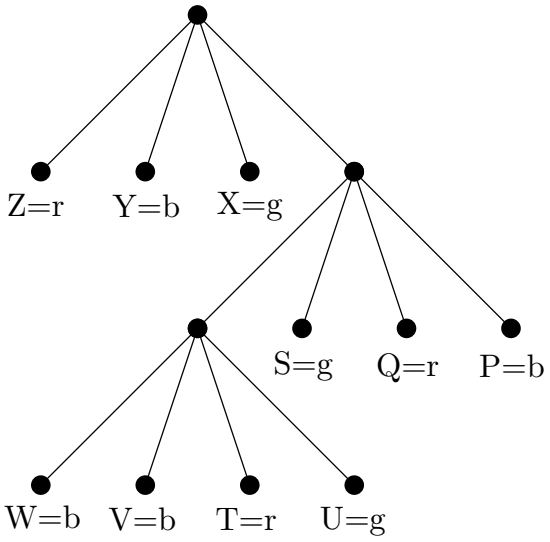


Figure 2

1. Écrivez un fichier `Quadtree.g4` qui permet de faire l'analyse des représentations de *quadtrees* telles que décrites ci-dessous.
2. Rajoutez dans la grammaire définie dans le fichier `Quadtree.g4` les attributs hérités `x`, `y` et `size` correspondant respectivement aux coordonnées x et y du coin supérieur gauche du carré et la taille *size* de ses côtés.

Le but du reste du TP est de produire une image au format SVG à partir d'une représentation de *quadtree*. Le fichier devra donc être au format suivant :

```
<?xml version="1.0" encoding="utf-8"?>
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" width="size" height="size">

 <!-- Carrés composant le quadtree -->

</svg>
```

La valeur de `"size"` devra correspondre à la taille de l'image (`"1024"` dans notre exemple). Pour définir les carrés de couleurs composant le *quadtree* on utilisera la balise `<rect>` qui permet de définir des rectangles. Le format pour définir un rectangle est le suivant :

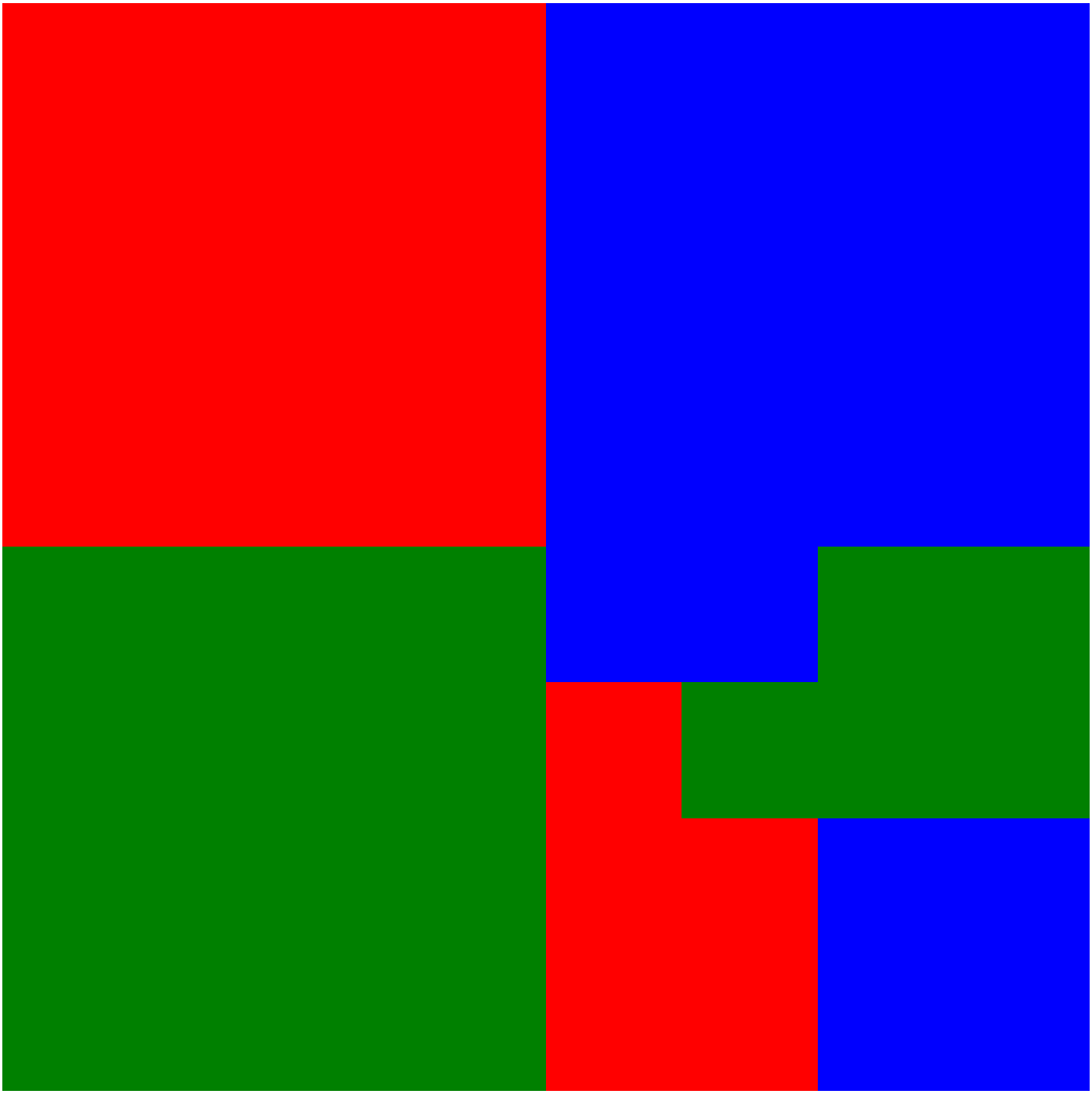
```
<rect width="size" height="size" x="x_value" y="y_value" fill="color"/>
```

La valeur `"size"` correspond à la taille du carré qui sera donc utilisé pour définir la largeur et hauteur du rectangle, les valeurs `"x_value"` et `"y_value"` correspondent respectivement aux coordonnées x et y du coin supérieur gauche du carré et la valeur `"color"` correspond à la couleur du carré qui est soit `"red"`, `"blue"` ou `"green"`.

Le fichier qui doit être produit à partir de l'entrée `(rbg((bbrg)grb))` avec une taille d'image 1014 devra être le suivant :

```
<?xml version="1.0" encoding="utf-8"?>
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" width="1024" height="1024">
    <rect width="512" height="512" x="0" y="0" fill="red" />
    <rect width="512" height="512" x="512" y="0" fill="blue" />
    <rect width="512" height="512" x="0" y="512" fill="green" />
    <rect width="128" height="128" x="512" y="512" fill="blue" />
    <rect width="128" height="128" x="640" y="512" fill="blue" />
    <rect width="128" height="128" x="512" y="640" fill="red" />
    <rect width="128" height="128" x="640" y="640" fill="green" />
    <rect width="256" height="256" x="768" y="512" fill="green" />
    <rect width="256" height="256" x="512" y="768" fill="red" />
    <rect width="256" height="256" x="768" y="768" fill="blue" />
</svg>
```

L'image aura l'affichage suivant :



3. Utilisez soit des actions sémantiques dans la grammaire `Quadtree.g4` ou bien un visiteur pour construire un fichier image `SVG` à partir d'une représentation de quadtree.
4. Créer une classe `QuadtreeConvert` ayant une méthode `main` qui permet de construire un fichier image `output.svg` de taille `size` à partir d'un fichier `input.quad` contenant la représentation d'un *quadtree* avec la ligne de commande `java QuadtreeConvert size input.quad output.svg`.