1 Matrices

Une matrice est un tableau de *chiffres*. 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 chiffres `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. Une matrice vide contient une ou plusieurs lignes vides.

1. Écrire une grammaire non ambiguë G qui génère des matrices ayant un nombre de lignes et de colonnes quelconque.
2. Écrire l'arbre de dérivation de la matrice $(1, 2), (3, 4, 5)$
3. Écrire une grammaire attribuée pour G, fondée sur l'attribut synthétisé l qui permet de représenter combien la matrice possède de lignes. Si S est l'axiome de la grammaire, $S.l$ représente le nombre de lignes de la matrice générée par S.
4. Écrire une grammaire attribuée pour G, fondée sur l'attribut synthétisé c qui permet de représenter combien la matrice possède de colonnes. Si S est l'axiome de la grammaire, $S.c$ représente le nombre de colonnes de la matrice générée par S.
5. Écrire une grammaire attribuée pour G, fondée sur l'attribut synthétisé nz qui permet de représenter le nombre de cases non vides (différentes de zéro) dans cette matrice. Les cases contenant un zéro explicite (valeur `0`) sont considérées vides.
6. Écrire une grammaire attribuée pour G, fondée sur deux attributs hérités lh et ch et un attribut synthétisé v. Si l'on spécifie la valeur de lh et de ch pour l'axiome, l'attribut v vaut 1 si la matrice générée possède lh lignes et ch colonnes. Il vaut 0 sinon.
7. Écrire une grammaire attribuée pour G, fondée sur un attribut hérité cr et un attribut synthétisé p. Si l'on spécifie la valeur de cr pour l'axiome, l'attribut p vaut 1 si la matrice contient le chiffre recherché cr. Il vaut 0 sinon.
8. Écrire une grammaire attribuée pour G, fondée sur l'attribut synthétisé z qui permet de représenter le nombre de cases vides (égales à zéro) dans cette matrice. Cela comprend les cases contenant un zéro explicite (valeur `0`) et les zéros implicites omis à la fin de chaque ligne.

2 Expressions régulières

Les expressions régulières constituent une manière de décrire les langages réguliers. Voici la syntaxe des expressions régulières ainsi que le langage qu'elles dénotent, où X et Y sont des expressions régulières et x un symbole de l'alphabet. Pour éviter toute confusion, nous appellerons *eps* le symbole ε et *vide* le symbole $\emptyset$ quand ils font partie d'une expression régulière.

Expression régulière	Langage	Expression régulière	Langage
x	$\{x\}$	$X.Y$	$L(X).L(Y)$
eps (ε)	$\{\varepsilon\}$	$X+Y$	$L(X) \cup L(Y)$
vide ($\emptyset$)	$\emptyset$	X^*	$\bigcup_{n\geq 0} L(X)^n$
(X)	$L(X)$		

où $L.L' = \{w.w' \mid w \in L, w' \in L'\}$ et L^n est la concaténation de L avec lui-même n fois (avec $L^0 = \{\varepsilon\}$).

L'expression régulière $a + b.a^*$, par exemple, correspond au langage $\{a, b, ba, baa \ldots\}$

1. Écrire une grammaire non ambiguë G des expressions régulières sur l'alphabet $\{a, b\}$, en supposant que l'étoile est plus prioritaire que le produit, lui-même plus prioritaire que la somme.
2. Écrire un arbre de dérivation de l'expression $(a + b)^*.a$ [1]
3. On définit l'attribut synthétisé x qui représente l'expression, régulière correspondant aux nœuds de l'arbre de dérivation. Écrire une grammaire attribuée permettant de calculer la valeur de l'attribut x.

On aimerait savoir si le langage dénoté d'une expression régulière contient ε. Pour cela, on définit l'attribut booléen synthétisé e. Étant donné un nœud d'un arbre de dérivation, étiqueté X. $X.e$ vaut 1 si l'expression $\varepsilon \in L(X)$ et 0 sinon.

4. Écrire une grammaire attribuée permettant de calculer la valeur de l'attribut e.

Le *résiduel* d'un langage L par rapport à un symbole s, noté L/s est l'ensemble des mots de L ayant s pour préfixe, auxquels on a éliminé ce préfixe. En d'autres termes :

$$L/s = \{w \mid w \in \Sigma^*, sw \in L\}.$$

Exemple : si $L = \{a, abc, b\}$ alors $L/a = \{\varepsilon, bc\}$

Si E est une expression régulière correspondant au langage L et s est un symbole, alors le langage L/s peut aussi être décrit par une expression régulière, grâce aux règles suivantes.

$$\begin{array}{rclrcl}
x/y & = & \varepsilon \text{ si } x = y & x/y & = & \emptyset \text{ si } x \neq y \\
X.Y/x & = & (X/x).Y \text{ si } \varepsilon \notin L(X) & X.Y/x & = & (X/x).Y + Y/x \text{ si } \varepsilon \in L(X) \\
X + Y/x & = & (X/x) + (Y/x) & X^*/x & = & (X/x).X^* \\
\text{eps}/x & = & \emptyset & \text{vide}/x & = & \emptyset
\end{array}$$

où X, Y sont des expressions régulières sur Σ, et $x, y \in \Sigma$.

5. Donner des expressions régulières qui représentent les résiduels par rapport au symbole a des langages décrits par les expressions régulières suivantes : $(a.b + b.c)$, $a^*.b$ et $b.(a + b)^*$.

[1] Attention: dans les expressions régulières telles qu'elles sont définies ici, la concaténation est toujours explicite et requiert l'opérateur . entre chaque paire d'expression concaténée.

6. On définit un attribut hérité s et un attribut synthétisé r. s représente le symbole par rapport auquel on veut calculer un résiduel et r représente l'expression régulière dénotant le résiduel. Écrire une grammaire attribuée permettant de calculer les valeurs des deux attributs s et r.

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, 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))`

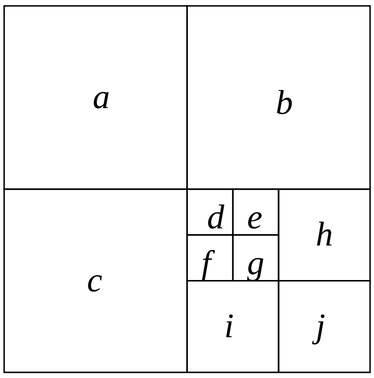

Figure 1

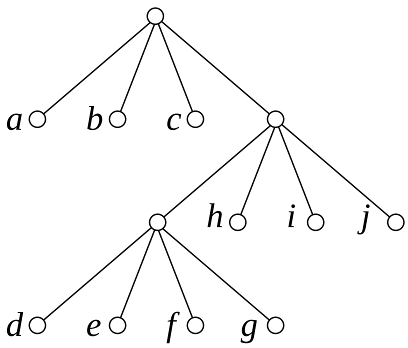

Figure 2

1. Écrivez une grammaire G des quadtrees considérant l'alphabet des couleurs $\{\mathtt{r}, \mathtt{g}, \mathtt{b}\}$.
2. Dessinez l'arbre de dérivation du quadtree `(((bbbr)(gbrg)rg)(bbrb)rb)`
3. Écrivez une grammaire attribuée pour compter le nombre s de sous-carrés simples qui composent un quadtree. La valeur de cet attribut à la racine de l'arbre de dérivation pour l'exemple de la figure 1 est $s = 10$.

On dispose par ailleurs d'un périphérique graphique qui reconnaît l'unique instruction élémentaire

`RECT` x_0 y_0 x_1 y_1 c

dans laquelle x_0, y_0, x_1 et y_1 sont des nombres entiers. L'exécution de cette instruction produit le remplissage du rectangle $R = \{(x, y) \mid x_0 \leq x < x_1 \quad y_0 \leq y < y_1\}$ avec la couleur c. Un programme pour notre périphérique, appelé *programme-machine*, est une suite de telles instructions.

4. En vous aidant de la figure 1, écrivez le programme-machine du *quadtree* `(rbg((bbrg)grb))`.
5. Écrivez une grammaire attribuée pour les attributs x_0, y_0 et t contenant les coordonnées x et y du coin supérieur gauche du carré et la taille t de ses côtés.
6. On souhaite connaître le nombre de pixels de chacune des trois couleurs. Pour cela on crée les attributs n_R, n_G et n_B qui indiquent le nombre de pixels de chaque couleur. Écrivez une grammaire attribuée pour les attributs n_R, n_G et n_B.