Devoir maison n°1bis - À rendre le 02 Novembre 2025 (pour les 5/2 surtout)

Exercice 1 : Arbres de construction de mots

Dans tout cet exercice on considère des expressions régulières et des mots sur un alphabet Σ fixé, qui sera $\{\alpha, \beta\}$ dans les exemples. En guise d'exemple on utilisera les expressions régulières suivantes.

$$f \stackrel{\text{déf}}{=} (\alpha \cdot \varepsilon)^\star | ((\beta \cdot \alpha) | \emptyset) \;\text{ et }\; e_{\mathsf{ex}} \stackrel{\text{déf}}{=} f^\star$$

Le mot $w = \alpha\alpha\beta\alpha$ est dans le langage de e_{ex} en tant que concaténation de trois mots dans le langage de f : $\alpha\alpha$, $\beta\alpha$ et ε. On remarque que w est aussi dans le langage de e_{ex} en tant que concaténation de trois autres mots dans le langage de f : α, α et $\beta\alpha$.

Ainsi, pour une expression régulière donnée, un même mot admet potentiellement plusieurs “constructions” qui justifient qu'il est dans le langage de cette expression régulière. Dans cet exercice on introduit la notion d'arbres de construction permettant de représenter la manière dont un mot est construit selon une expression régulière (comprendre : quels choix ont été faits lors de la lecture de l'expression régulière pour obtenir ce mot). On étudie ensuite des propriétés de ces arbres pour retrouver le lemme de l'étoile sans passer par les automates.

On définit par induction structurelle les **arbres de construction**, notés $\mathcal{A}$, de la manière suivante :

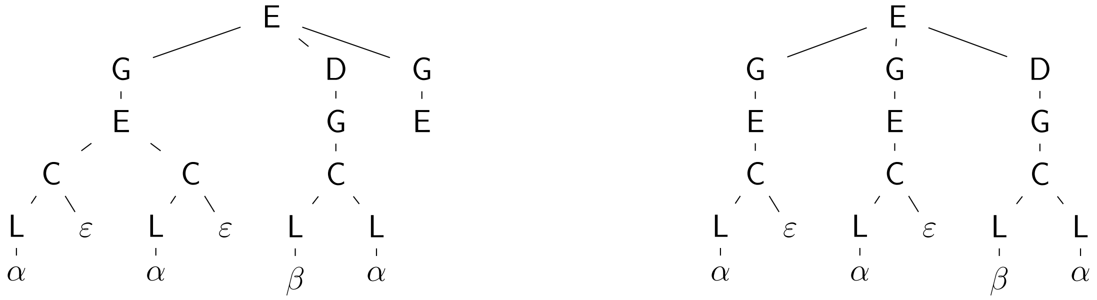
- $\varepsilon \in \mathcal{A}$;
- $l \in \mathcal{A}$, pour $l \in \Sigma$;
- Si $a \in \mathcal{A}$ alors $\mathsf{G}(a) \in \mathcal{A}$ et $\mathsf{D}(a) \in \mathcal{A}$;
- Si $a \in \mathcal{A}$ et $b \in \mathcal{A}$ alors $\mathsf{C}(a, b) \in \mathcal{A}$;
- Si $n \in \mathbb{N}$ et $(a_1, a_2, \dots, a_n) \in \mathcal{A}^n$ alors $\mathsf{E}(a_1, a_2, \dots, a_n) \in \mathcal{A}$.

G et D ont été choisis pour désigner Gauche et Droite ; E pour désigner Étoile, C pour désigner Concaténation.

Définitions OCaml. Un fichier `compagnon_arbres_construction.ml` est fourni avec cet exercice. On y trouvera les définitions des types mentionnés dans ce sujet ainsi que les exemples de l'énoncé. Un arbre de construction est représenté en OCaml au moyen du type ci-dessous.

```
type a_cons =
  | Eps                      (* ε *)
  | L of char                (* Une lettre de Σ *)
  | G of a_cons              (* G(a) *)
  | D of a_cons              (* D(a) *)
  | C of a_cons * a_cons     (* C(a,b) *)
  | E of a_cons list         (* E(a1,a2,...,an) *)
```

Les figures 1a et 1b représentent deux arbres de construction sur l'alphabet $\Sigma = \{\alpha, \beta\}$. Ces deux arbres de construction correspondent aux deux constructions données en début d'exercice pour justifier que $\alpha\alpha\beta\alpha$ est dans le langage de e_{ex}.



(a) arbre de construction a^1_{ex} (b) arbre de construction a^2_{ex}

FIGURE 1 – Exemples d'arbres de construction

On définit, le **mot associé** à un arbre de construction a, noté[♣] $p(a)$, par induction sur les arbres de construction :
- $p(\varepsilon) \stackrel{\text{déf}}{=} \varepsilon$;
- $p(l) \stackrel{\text{déf}}{=} l$ pour $l \in \Sigma$;
- $p(\mathsf{G}(a)) \stackrel{\text{déf}}{=} p(\mathsf{D}(a)) \stackrel{\text{déf}}{=} p(a)$ pour $a \in \mathcal{A}$;
- $p(\mathsf{C}(a, b)) \stackrel{\text{déf}}{=} p(a) \cdot p(b)$ pour $a \in \mathcal{A}$ et $b \in \mathcal{A}$;
- $p(\mathsf{E}(a_1, a_2, \ldots, a_n)) \stackrel{\text{déf}}{=} p(a_1) \cdot p(a_2) \cdot \ldots p(a_n)$ pour $n \in \mathbb{N}$ et $(a_1, a_2, \ldots, a_n) \in \mathcal{A}^n$[♡].

Par exemple, le mot associé à l'arbre de construction a^1_{ex} de la figure 1a est $p(a^1_{\mathsf{ex}}) = \alpha\alpha\beta\alpha$, et c'est aussi celui associé à a^2_{ex}.

Définitions OCaml. On représente les mots au moyen du type `type mot = char list`.

Q. 1 Définir une fonction `mot : a_cons -> mot` prenant en paramètre un arbre de construction et retournant le mot qui lui est associé. On assurera une complexité algorithmique linéaire en la taille de l'arbre de construction passé en paramètre.

On définit le fait qu'un arbre de construction $a \in \mathcal{A}$ soit un **modèle** d'une expression régulière $e \in \mathcal{E}_{reg}$, ce que l'on note $a \models e$, par induction sur les expressions régulières :
- $\varepsilon \models e$ si et seulement $e = \varepsilon$;
- $l \models e$ si et seulement $e = l$;
- $\mathsf{G}(b) \models e$ si et seulement si e est de la forme $f|g$ et $b \models f$;
- $\mathsf{D}(b) \models e$ si et seulement si e est de la forme $f|g$ et $b \models g$;
- $\mathsf{C}(b, c) \models e$ si et seulement si e est de la forme $f \cdot g$ et $b \models f$ et $c \models g$;
- $\mathsf{E}(a_1, a_2, \ldots, a_n) \models e$ si et seulement si e est de la forme $f^\star$ et $\forall i \in [\![1, n]\!], a_i \models f$.

Par exemple, l'arbre de construction a^1_{ex} de la figure 1a est modèle de l'expression régulière e_{ex}, mais ce n'est pas un modèle de l'expression régulière $((\alpha \cdot \varepsilon)^\star | (\beta \cdot \alpha))^\star$

Q. 2 Définir une fonction `est_modele : a_cons -> regexp -> bool` prenant en paramètres un arbre de construction a et une expression régulière e et testant si $a \models e$.

Dans la suite, lorsque e est une expression régulière, on note $\mathcal{M}(e)$ **l'ensemble de ses modèles**, à savoir :

$$\mathcal{M}(e) \stackrel{\text{déf}}{=} \{a \in \mathcal{A} \mid a \models e\}.$$

♣. p comme production

♡. Avec le cas particulier, pour $n = 0$, que $p(a_1) \cdot p(a_2) \cdot \ldots p(a_n) = \varepsilon$

Q. 3 Pour f et g deux expressions régulières, exprimer $\mathcal{M}(f\cdot g)$, $\mathcal{M}(f|g)$ et $\mathcal{M}(f^\star)$ en fonction de $\mathcal{M}(f)$ et $\mathcal{M}(g)$.

Q. 4 Démontrer que pour toute expression régulière $e \in \mathscr{E}_{reg}$, $\mathcal{L}(e) = \{p(a) \mid a \in \mathcal{M}(e)\}$.

On dit d'un arbre de construction $a \in \mathcal{A}$ qu'il est **sans étoile productive**, dès lors que pour tout sous-arbre b de l'arbre a, s'il existe $n \in \mathbb{N}$, tel que $b = \mathsf{E}(a_1, a_2, \ldots, a_n)$ alors $\forall i \in [\![1, n]\!], p(a_i) = \varepsilon$. On rappelle que la **taille** d'une expression régulière $e \in \mathscr{E}_{reg}$, notée $\|e\|$ dans cet exercice, est le nombre de ses connecteurs, constantes et lettres comptées avec multiplicité.

Q. 5 Montrer que pour toute expression régulière $e \in \mathscr{E}_{reg}$, pour tout arbre de construction $a \in \mathcal{A}$ qui est un modèle de e, si a est sans étoile productive, alors $|p(a)| \leqslant \|e\|$.

Q. 6 Montrer que pour tout arbre de construction $a \in \mathcal{A}$ et tout sous-arbre b de a, il existe deux mots x et y tels que :

(i) $p(a) = x\cdot p(b)\cdot y$;

(ii) pour tout arbre de construction b', l'arbre a' obtenu en remplaçant, dans a, le sous-arbre b par b' vérifie $p(a') = x\cdot p(b')\cdot y$.

De la même manière on peut montrer le résultat suivant, que l'on admettra dans la suite.

Proposition 0.1

Si a est modèle d'une expression régulière e et si $b = \mathsf{E}(a_1, a_2, \ldots, a_n)$ est un sous-arbre de a, alors en notant $b' = \mathsf{E}(a_1, a_2, \ldots, a_{i-1}, \underbrace{a_i, a_i, \ldots, a_i}_{k \text{ fois}}, \ldots, a_n)$, et a' l'arbre obtenu en remplaçant, dans a, le sous-arbre b par b', a' est modèle de e. Précisons que ceci est vrai pour tout $i \in [\![1, n]\!]$ et pour tout $k \in \mathbb{N}$, y compris 0.

Q. 7 Définir une fonction `iter_etoile : a_cons -> int -> int -> a_cons option` prenant en paramètres un arbre a, un entier i et un entier k et retournant l'arbre a' obtenu en remplaçant le premier (dans l'ordre préfixe) sous-arbre de a de la forme $\mathsf{E}(a_1, a_2, \ldots, a_n)$ par l'arbre $\mathsf{E}(a_1, a_2, \ldots, a_{i-1}, \underbrace{a_i, a_i, \ldots, a_i}_{k \text{ fois}}, \ldots, a_n)$ (on peut supposer que $i \leqslant n$). Si aucun sous-arbre de l'arbre a n'est de la forme $\mathsf{E}(\ldots)$, la fonction devra retourner `None`.

Q. 8 En utilisant la fonction `iter_etoile` de la **Q. 7**, proposer un jeu de tests permettant de se persuader de la véracité de la propriété 0.1.

Q. 9 Déduire de la propriété 0.1 que si L est un langage régulier, alors il existe $N \in \mathbb{N}$, tel que pour tout mot $w \in L$ de longueur strictement supérieure N, il existe trois mots $(x, y, z) \in (\Sigma^\star)^3$ tels que :

$a)$ $x\cdot y\cdot z = w$;

$b)$ $y \neq \varepsilon$;

$c)$ $\forall k \in \mathbb{N}, x\cdot y^k\cdot z \in L$.

Q. 10 Définir une fonction `trouve_xyz : a_cons -> (mot * mot * mot) option` prenant en paramètre un arbre de construction a et retournant un triplet de mots (x, y, z) (avec $y \neq \varepsilon$) tel que pour tout $k \in \mathbb{N}$, xy^kz est un mot du langage de tout expression régulière dont a est un modèle. Si aucun tel triplet n'existe, la fonction devra renvoyer `None`.

Q. 11 Déduire de la question **Q. 9** que le langage $L \overset{\text{déf}}{=} \{\alpha^{n^2} \mid n \in \mathbb{N}\}$ n'est pas un langage régulier.