Devoir d'automates d'arbres
À remettre au plus tard le 25 octobre 2013

Partie 1: contraintes ensemblistes

Dans toute la suite, $\mathcal{F}$ est un alphabet fini de symboles de fonction (à arité fixée). $T(\mathcal{F})$ est l'ensemble des arbres finis étiquetés par $\mathcal{F}$ et $2^{T(\mathcal{F})}$ est l'ensemble des sous-ensembles (ou langages) de $T(\mathcal{F})$.

$\mathcal{S}(\mathcal{X})$, ensemble des *expression ensemblistes* sur les variables $\mathcal{X}$ est le plus petit ensemble tel que:

- $\mathcal{X} \subseteq \mathcal{S}(\mathcal{X})$
- $\bot, \top \in \mathcal{S}(\mathcal{X})$
- Si $e_1, e_2 \in \mathcal{S}(\mathcal{X})$, alors $e_1 \cap e_2, e_1 \cup e_2 \in \mathcal{S}(\mathcal{X})$
- Si $f \in \mathcal{F}$ est d'arité n et $e_1, \ldots, e_n \in \mathcal{S}(\mathcal{X})$, alors $f(e_1, \ldots, e_n) \in \mathcal{S}(\mathcal{X})$
- Si $f \in \mathcal{F}$ est d'arité n, $i \in \{1, \ldots, n\}$ et $e \in \mathcal{S}(\mathcal{X})$, alors $f_i^{-1}(e) \in \mathcal{S}(\mathcal{X})$.

Si σ est une application de $\mathcal{X}$ dans $2^{T(\mathcal{F})}$, et $e \in \mathcal{S}(\mathcal{X})$, $[\![e]\!]_\sigma$ est défini par récurrence:

- $[\![\top]\!]_\sigma = T(\mathcal{F})$, $[\![\bot]\!]_\sigma = \emptyset$
- $[\![X]\!]_\sigma = \sigma(X)$ si $X \in \mathcal{X}$
- $[\![e_1 \cap e_2]\!]_\sigma = [\![e_1]\!]_\sigma \cap [\![e_2]\!]_\sigma$, $[\![e_1 \cup e_2]\!]_\sigma = [\![e_1]\!]_\sigma \cup [\![e_2]\!]_\sigma$
- $[\![f(e_1, \ldots, e_n)]\!]_\sigma = \{f(t_1, \ldots, t_n) \mid t_1 \in [\![e_1]\!]_\sigma, \ldots, t_n \in [\![e_n]\!]_\sigma\}$
- $[\![f_i^{-1}(e)]\!]_\sigma = \{t \in T(\mathcal{F}) \mid \exists t_1, \ldots, t_n \in T(\mathcal{F}), f(t_1, \ldots, t_{i-1}, t, t_{i+1}, \ldots, t_n) \in [\![e]\!]_\sigma\}$

Une *contrainte ensembliste* est une formule $\exists X_1, \ldots, \exists X_m.\ e_1 \subseteq e'_1 \wedge \ldots e_n \subseteq e'_n$ où $e_1, \ldots, e_n, e'_1, \ldots, e'_n \in \mathcal{S}(\mathcal{X})$. Une *solution* d'une telle contrainte est une application σ des variables libres de la formule (i.e., les variables distinctes de $X_1, \ldots, X_m$) dans $2^{T(\mathcal{F})}$ telle qu'il existe une application θ de $\{X_1, \ldots, X_m\}$ dans $2^{T(\mathcal{F})}$, telle que, pour tout $i = 1, ..., n$, $[\![e_i]\!]_{\sigma \cup \theta} \subseteq [\![e'_i]\!]_{\sigma \cup \theta}$.

Une contrainte $\exists X_1, \ldots, \exists X_n.\ e_1 \subseteq e'_1 \wedge \ldots e_n \subseteq e'_n$ est *définie* si les expressions $e'_1, \ldots e'_n$ ne contiennent aucun symbole $\cup$ et aucun symbole f_i^{-1}.

Si σ, θ sont deux solutions d'une contrainte ensembliste, $\sigma \leq \theta$ si, pour tout $X \in \mathcal{X}$, $\sigma(X) \subseteq \theta(X)$.

Une solution σ de la contrainte ensembliste ϕ, de variables libres $Z_1, \ldots, Z_n$ est *reconnaissable* s'il existe un automate d'arbre $\mathcal{A}$ qui accepte respectivement dans des états $q_{Z_1}, \ldots, q_{Z_n}$ les langages $\sigma(Z_1), \ldots, \sigma(Z_n)$.

Question 1

Donner une solution (sous la forme d'un automate d'arbres) de la contrainte ensembliste:

$$\begin{cases} a \cup f_1^{-1}(X) & \subseteq & X \\ g(a) & \subseteq & Y \\ f(Y,X) & \subseteq & X \cap Y \\ Y \cap a & \subseteq & \bot \end{cases}$$

sur $\mathcal{F} = \{f(2), g(1), a(0)\}$.

Donner une solution non-reconnaissable de cette contrainte.

Question 2

1. Montrer que, si $e \in S(\mathcal{X})$ et $\sigma \leq \theta$ sont des applications de $\mathcal{X}$ dans $2^{T(\mathcal{F})}$, alors $[\![e]\!]_\sigma \subseteq [\![e]\!]_\theta$.
2. Montrer que toute contrainte ensembliste définie ou bien n'a pas de solution, ou bien a une solution minimale (pour $\geq$).
3. Donner un exemple de contrainte ensembliste qui a au moins une solution, mais pas de solution minimale.

Question 3

Montrer que, pour toute contrainte ensembliste définie ϕ, on peut calculer (en temps polynomial) une contrainte ensembliste ψ qui a le même ensemble de solutions que ϕ et telle que les formules atomiques de ψ sont de l'une des formes:

- $X \subseteq Y$
- $X \subseteq \bot$
- $X \cap Y \subseteq Z$
- $\top \subseteq X$
- $f(X_1, \ldots, X_n) \subseteq X$
- $f_i^{-1}(X) \subseteq Y$
- $X \subseteq f(\top, \ldots, \top)$

Une telle contrainte sera dite *simple*.

Question 4

Soit ϕ une contrainte ensembliste simple, qui ne comporte aucune inclusion $X \subseteq \bot$ et aucune inclusion $X \subseteq f(\top, \ldots, \top)$. Montrer comment construire (en temps polynomial) un automate d'arbres alternant bidirectionnel A_ϕ qui accepte la solution minimale de ϕ.

Question 5

Déduire des questions précédentes un algorithme (exponentiel) qui étant donnée une contrainte ensembliste définie, décide si celle-ci possède une solution et, si oui, calcule un automate d'arbres qui accepte la plus petite solution.

Partie 2: automates ascendants d'arbres à arbre (aaaaa)

On considère dans cette partie une généralisation des classiques automates à pile de mots: d'une part nos automates reconnaitront des arbres et, d'autre part, nous utiliserons des arbres au lieu des piles. Enfin, nous n'utiliserons pas seulement des accès à la mémoire qui correspondent aux accès dans les piles (empiler, dépiler, tester le sommet de pile) mais aussi des tests d'égalité entre les arbres synthétisés aux fils d'un même noeud.

Si Γ est un ensemble de symboles de fonction (avec arité), on note $U_\Gamma = \bigcup_{n=0}^{+\infty} U_\Gamma^n$ le plus petit ensemble de fonctions de $T(\Gamma)^n$ dans $T(\Gamma)$ tel que:

- Les fonctions (notées $\lambda x_1 \cdots x_n.x_i$) qui à $x_1, \ldots, x_n$ associent x_i sont dans U_Γ
- Les fonctions (notées $\lambda x_1, \ldots, x_n.f(x_{i_1}, \ldots, x_{i_k})$) qui à $x_1, \ldots, x_n$ associent $f(x_{i_1}, \ldots, x_{i_k})$ où $i_1, \ldots i_k \in \{1, \ldots, n\}$ sont des indices distincts et f un symbole d'arité k sont dans U_Γ
- Les fonctions (notées $\lambda f(x_1, \ldots, x_n).x_i$) qui à $f(x_1, \ldots, x_n)$ associent x_i et ne sont pas définies sur les autres arbres, sont dans U_Γ.
- U_Γ est clos par composition

Un aaaaa est donné par deux alphabets $\mathcal{F}, \Gamma$ de symboles de fonction, un ensemble fini d'états, un sous-ensemble d'états finaux et une relation de transition définie par un ensemble fini de règles:

$$f(q_1, \ldots, q_n) \xrightarrow[h]{\sim} q$$

où $q_1, \ldots, q_n, q \in Q$, $\sim$ est une relation d'équivalence sur $\{1, \ldots, n\}$ et $h \in U_\Gamma$ est une fonction de $T(\Gamma)^m$ dans $T(\Gamma)$ où m est le nombre de classes d'équivalence de c.

Un *calcul* γ de l'aaaaaa sur un terme t est un arbre, ayant les mêmes positions que celles de t, mais étiqueté par $\mathcal{F} \times Q \times T(\Gamma)$ et tel que, pour toute position p de t,

> si $p \cdot 1, \ldots p \cdot n$ sont les positions de t de longueur $|p|+1$, si $\gamma(p) = \langle f, q, u \rangle$ et $\gamma(p \cdot i) = \langle g_i, q_i, u_i \rangle$ pour $i = 1, \ldots, n$,
> alors il existe une transition $f(q_1, \ldots, q_n) \xrightarrow[h]{\sim} q$ de l'automate, telle que
>
> - $u_i = u_j$ si $i \sim j$
> - Si $i_1, \ldots, i_m$ sont des représentants des classes d'équivalence de $\sim$ tels que $i_1 < \ldots < i_m$, alors h est définie sur $u_{i_1} \ldots, u_{i_m}$ et $h(u_{i_1}, \ldots, u_{i_m}) = u$.

Un calcul sur t est acceptant si la racine est étiquetée par un état final. Le langage accepté par un automate est l'ensemble des t tels qu'il existe un calcul acceptant de l'automate sur t.

Question 1

Soit $\mathcal{F} = \{f(2), g(1), a(1)\}$ et $\Gamma = \{s(1), 0(0)\}$ Soit l'automate donné par les règles suivantes (tous les états sont finaux):

$$a \xrightarrow[0]{} q_0 \qquad g(q_0) \xrightarrow[\lambda x_1.s(x_1)]{} q_0$$

$$f(q_0, q_0) \xrightarrow[\lambda x_1.s(0)]{1\sim 2} q_1 \qquad f(q_0, q_1) \xrightarrow[\lambda x_1,x_2.s(x_2)]{} q_1$$

$$f(q_1, q_0) \xrightarrow[\lambda x_1,x_2.s(x_1)]{} q_1 \qquad f(q_1, q_1) \xrightarrow[\lambda x_1.s(x_1)]{1\sim 2} q_1$$

$$g(q_1) \xrightarrow[\lambda x_1.s(0)]{} q_0$$

Parmi les termes suivants, lesquels sont acceptés par $\mathcal{A}$? (Pour ceux qui sont acceptés, le justifier en donnant un calcul de $\mathcal{A}$)

1. $f(a, g(a))$
2. $g(f(g(a), g(a)))$
3. $f(f(a, a), f(g(a), g(a)))$

Question 2

Donner un exemple de langage d'arbres qui est accepté par un aaaaa mais par aucun aaa.

Question 3

Les mots sur un alphabet Σ peuvent être considérés comme des arbres sur l'alphabet $\Sigma \cup \{0\}$ où tous les symboles de Σ sont unaires: si $w \in \Sigma^*$, $\widehat{w}$ est défini par: $\widehat{\epsilon} = 0$, $\widehat{w \cdot a} = a(\widehat{w})$.

Montrer que, si $L \subseteq \Sigma^*$ est accepté (par état final) par un automate à pile sans ϵ-transition, alors $\{\widehat{w} \mid w \in L\}$ est accepté par un aaaaa.

Question 4

Montrer que, étant donné un aaaaa, on peut construire une contrainte ensembliste définie, de variables libres $\{X_q, q \in Q\}$ et dont la plus petite solution σ est telle que $\sigma(X_q) = \{u \mid$ il existe un calcul ρ de $\mathcal{A}$ et $f \in \mathcal{F}$ tels que $\rho(\epsilon) = \langle f, q, u \rangle\}$

Question 5

Déduire de la question précédente et de la première partie que le vide des aaaaa est décidable en temps exponentiel.

Question 6

Quelles sont les propriétés de clôture satisfaites par la classe des langages reconnus par des aaaaa ? Justifier.

Solution

Partie 1

Question 1

Solution par automate d'arbre. Une solution reconnaissable σ est donnée par l'automate dont les transitions sont :

$$\begin{array}{rcl} a & \to & q_X \\ g(q_X) & \to & q_X \\ f(q_Y, q_X) & \to & q_Y \end{array} \qquad \begin{array}{rcl} g(q_X) & \to & q_Y \\ f(q_Y, q_X) & \to & q_X \end{array}$$

- On prouve d'abord que $\sigma(Y) \subseteq \sigma(X)$ par exemple par récurrence sur la taille des termes de $\sigma(Y)$, ce qui conduit à montrer que la première contrainte est bien vérifiée.
- La deuxième contrainte est aisément vérifiée par construction de l'automate, il suffit de donner un run acceptant de $g(a)$.
- La troisième contrainte se montre également par récurrence simple sur la hauteur des termes.
- La dernière contrainte est immédiate après analyse des transitions de l'automate.

Solution non-reconnaissable Une solution non reconnaissable est donnée par :

$$\sigma(X) = \{g^{n^2}(a) \mid n \in \mathbb{N}\} \cup \{f(t,t) \mid t \in T(\mathcal{F})\}$$
$$\sigma(Y) = \sigma(X) \setminus \{a\}$$

La non-reconnaissabilité de σ se prouve par l'absurde en utilisant le lemme de pompage. Il faut également vérifier que chacune des quatre contraintes est bien vérifiée par cette solution.

Question 2

1. La preuve se fait par récurrence sur la hauteur des termes. Les trois cas de base utilisent directement la définition de $[\![.]\!]$ et la propriété $\sigma \leq \theta$. Les quatre cas de récurrence se montrent en utilisant l'hypothèse de récurrence sur les sous termes des membres droits de la définition de $[\![.]\!]$ Par exemple, si $e = f_i^{-1}(e')$, pour tout $t \in [\![e]\!]_\sigma$, il existe $t_1, \ldots, t_n \in T(\mathcal{F})$ tels que $f(t_1, \ldots, t_{i-1}, t, t_{i+1}, \ldots, t_n) \in [\![e']\!]_\sigma$. Grâce à l'hypothèse de récurrence on a $[\![e']\!]_\sigma \subseteq [\![e']\!]_\theta$, d'où $f(t_1, \ldots, t_{i-1}, t, t_{i+1}, \ldots, t_n) \in [\![e']\!]_\theta$ soit $t \in [\![e]\!]_\theta$.

2. Il suffit de montrer que l'ensemble des solutions d'une contrainte ensembliste est stable par intersection. si σ_1 et σ_2 sont des solutions de ϕ, soit σ définie par $\sigma(X) = \sigma_1(X) \cap \sigma_2(X)$. On peut, sans perte de généralité se restreindre au cas où ϕ n'a pas de variable liée (il suffit de considérer les substitutions $\sigma_i \cup \theta_i$ au lieu de σ_i). Il suffit ensuite de montrer la propriété pour une contrainte atomique $e \subseteq e'$. (si σ est solution de toutes les contraintes atomiques, alors elle est solution de la contrainte ensembliste).

   Comme $\sigma \leq \sigma_1$ et $\sigma \leq \sigma_2$, d'après 2.1, pour toute expression ensembliste e, $[\![e]\!]_\sigma \subseteq [\![e]\!]_{\sigma_1}$ et $[\![e]\!]_\sigma \subseteq [\![e]\!]_{\sigma_2}$. Par conséquent, $[\![e]\!]_\sigma \subseteq [\![e]\!]_{\sigma_1} \cap [\![e]\!]_{\sigma_2}$. Comme, par hypothèse, $[\![e]\!]_{\sigma_1} \subseteq [\![e']\!]_{\sigma_1}$ et $[\![e]\!]_{\sigma_2} \subseteq [\![e']\!]_{\sigma_2}$, il en résulte que $[\![e]\!]_\sigma \subseteq [\![e']\!]_{\sigma_1} \cap [\![e']\!]_{\sigma_2}$. On montre alors, par

récurrence sur e' que $[\![e']\!]_{\sigma_1} \cap [\![e']\!]_{\sigma_2} = [\![e']\!]_\sigma$ dans le cas où e' est construit à partir de $\mathcal{X}, \mathcal{F}, \top, \bot, \cap$ seulement (i.e., quand la contrainte est définie).

Pour $\top, \bot$ c'est immédiat. Pour $X \in \mathcal{X}$ c'est la définition de σ.

Si $e' = f(e_1, \ldots, e_n)$, $[\![f(e_1, \ldots, e_n)]\!]_{\sigma_1} \cap [\![f(e_1, \ldots, e_n)]\!]_{\sigma_2} = f([\![e_1]\!]_{\sigma_1} \cap [\![e_1]\!]_{\sigma_2}, \ldots [\![e_n]\!]_{\sigma_1}$ et on utilise l'hypothèse de récurrence. Dans le cas de l'intersection, c'est aussi immédiat par l'hypothèse de récurrence.

On conclut alors que $[\![e]\!]_\sigma \subseteq [\![e']\!]_\sigma$: σ est bien solution de $e \subseteq e'$.

3. On considère l'alphabet $\mathcal{F}$ qui contient l'unique symbole (constant) a et la contrainte $a \subseteq X \cup Y$. $\sigma_1(X) = \{a\}, \sigma_1(Y) = \emptyset$ d'une part et $\sigma_2(X) = \emptyset, \sigma_2(Y) = \{a\}$ d'autre part, définissent deux solutions de la contrainte. Elles sont toutes deux minimales puisque $\sigma(X) = \sigma(Y) = \emptyset$ n'est pas solution.

Question 3

On commence par abstraire les sous-expressions non variables:

$$e[e'] \subseteq e'' \to \exists X.e[X] \subseteq e'' \wedge e' \subseteq X \qquad e \subseteq e'[e''] \to \exists X.e \subseteq e'[X] \wedge X \subseteq e''$$

On montre d'après la question 2.1 et par récurrence structurelle sur les expressions ensemblistes que ces transformations préservent l'ensemble des solutions et, par ailleurs elles terminent si elles ne sont appliquées que lorsque e' (resp. e'') n'est pas une variable (démontré par décroissance stricte d'une fonction des termes vers les entiers positifs mesurant la "taille" d'un terme). Les expressions ensemblistes qui interviennent dans la contrainte peuvent donc désormais être supposées "plates" (ou de profondeur au plus 1), c'est-à dire être ou bien des variables ou bien des symboles de construction des expressions appliqués à des variables. Cette étape requiert un temps linéaire.

On utilise ensuite les transformations:

1.	$e \subseteq X \cap Y$	$\to$	$e \subseteq X \wedge e \subseteq Y$
2.	$X \cup Y \subseteq e$	$\to$	$X \subseteq e \wedge Y \subseteq e$
3.	$\bot \subseteq e \wedge C$	$\to$	C
4.	$e \subseteq \top \wedge C$	$\to$	C
5.	$e \subseteq f(X_1, \ldots, X_n)$	$\to$	$\exists X.e \subseteq X \wedge (\bigwedge_{i=1}^n f_i^{-1}(X) \subseteq X_i) \wedge X \subseteq f(\top, \ldots, \top)$
6.	$f_i^{-1}(X) \subseteq \bot$	$\to$	$\exists Y.f_i^{-1}(X) \subseteq Y \wedge Y \subseteq \bot$
7.	$f(X_1, \ldots, X_n) \subseteq \bot$	$\to$	$X_1 \subseteq \bot \wedge \ldots \wedge X_n \subseteq \bot$
8.	$X \cap Y \subseteq \bot$	$\to$	$\exists Z.X \cap Y \subseteq Z \wedge Z \subseteq \bot$

1. Elimine toutes les intersections à droite
2. Élimine toutes les unions à gauche
3. élimine tous les $\bot$ à gauche
4. Élimine tous les $\top$ à droite
5. Remplace tous les $f(X_1, \ldots, X_n)$ à droite par des $f(\top, \ldots, \top)$.
6. Les règles suivantes éliminent tous les cas où $\bot$ est à droite et ce n'est pas une variable à gauche.

Chacune des règles produit une contrainte atomique simple (ou supprime une contrainte atomique). Cette deuxième étape permet donc d'obtenir une contrainte simple en temps linéaire.

Il reste enfin à montrer que les règles ci-dessus préservent l'ensemble des solutions. C'est immédiat pour la plupart d'entre elles. Nous examinons seulement la règle 5, montrons qu'il existe une solution σ telle que $[\![e]\!]_\sigma \subseteq [\![f(X_1, \ldots, X_n)]\!]_\sigma$ si et seulement si il existe $\theta : X \mapsto \theta(X)$ telle que $[\![e]\!]_{\sigma\cup\theta} \subseteq [\![X]\!]_{\sigma\cup\theta}$ et pour tout $i \leq n$, $[\![f_i^{-1}(X)]\!]_{\sigma\cup\theta} \subseteq [\![X_i]\!]_{\sigma\cup\theta}$ ainsi que $[\![X]\!]_{\sigma\cup\theta} \subseteq [\![f(\top, \ldots, \top)]\!]_{\sigma\cup\theta}$. Pour prouver l'implication gauche vers droite, on pose $\theta(X) = [\![e]\!]_\sigma$ et on montre que chaque inclusion est vérifiée. Pour l'autre sens, il suffit de montrer que les conditions impliquent $[\![X]\!]_{\sigma\cup\theta} \subseteq [\![f(X_1, \ldots, X_n)]\!]_\sigma$.

Question 4

On suppose sans perte de généralité que la contrainte ensembliste de comporte pas de variable liée. L'automate bidirectionnel alternant est obtenu par une traduction immédiate:

$$\begin{array}{rcl} X \subseteq Y & \rightsquigarrow & q_X(x) \to q_Y(x) \\ X \cap Y \subseteq Z & \rightsquigarrow & q_X(x), q_Y(x) \to q_Z(x) \\ \top \subseteq X & \rightsquigarrow & \to q_X(x) \\ f(X_1, \ldots, X_n) \subseteq X & \rightsquigarrow & q_{X_1}(x_1), \ldots, q_{X_n}(x_n) \to q_X(f(x_1, \ldots, x_n)) \\ f_i^{-1}(X) \subseteq Y & \rightsquigarrow & q_X(f(x_1, \ldots, x_n)) \to q_Y(x_i) \end{array}$$

Pour montrer que le langage accepté est solution de ϕ on analyse chacune des formes possibles que peut prendre ϕ puis on utilise l'une des règles de transitions de $\mathcal{A}_\phi$. On se sert ensuite de la minimalité du modèle de Herbrand dans lequel on définit le langage de $\mathcal{A}_\phi$ pour démontrer que le langage accepté est bien la solution minimale de ϕ.

Question 5

D'après la question 3. on se ramène en temps polynômial à des contraintes simples. Une contrainte simple d'écrit $\exists X_1, \ldots, X_n.\phi \wedge \psi$ où ψ ne contient que des inclusions de la forme $X \subseteq \bot$ ou $X \subseteq f(\top, \ldots, \top)$ et ϕ n'en contient aucune.

D'après la question 4, la solution minimale σ de ϕ est acceptée par un automate alternant bidirectionnel $\mathcal{A}_\phi$. On remarque alors que ou bien (la projection de) σ est la solution minimale de ψ ou bien la contrainte est insatisfaisable (on utilise la question 2.2 pour prouver ce résultat).

À partir de $\mathcal{A}_\phi$ on calcule en temps exponentiel un automate ascendant $\mathcal{B}_\phi$ qui accepte σ.

Pour tester la satisfaisabilité de la formule, il suffit de tester que $\mathcal{B}_\phi$ satisfait les contraintes $X \subseteq \bot$ (resp. $X \subseteq f(\top, \ldots, \top)$, ce qui se fait très simplement:

- $X \subseteq \bot$ ssi le langage accepté dans l'état q_X par $\mathcal{B}_\phi$ est vide (ce test est polynomial)
- $X \subseteq f(\top, \ldots, \top)$ ssi, pour toute règle $g(q_1, \ldots, q_m) \to q_X$ de $\mathcal{B}_\phi$, si $f \neq g$, alors il existe un indice i tel que le langage accepté par $\mathcal{B}_\phi$ dans l'état q_i est vide. Ce test est à nouveau polnomial.

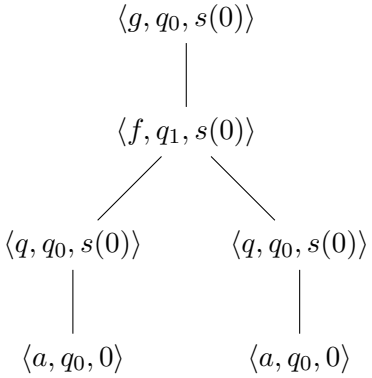
On a ainsi un test polynomial de l'existence d'une solution qui, au passage, calcule un automate $\mathcal{B}_\phi$ qui accepte la solution minimale quand elle existe.

Partie 2

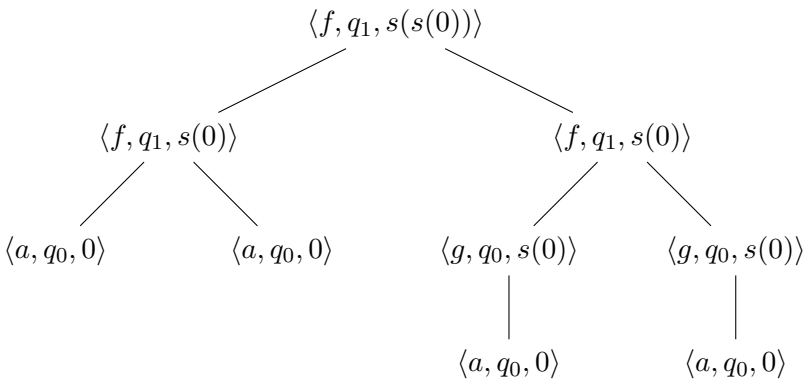
Question 1

1. $f(a, g(a))$ n'est pas accepté. En effet, il existe un seul calcul sur a: $\langle a, q_0, 0\rangle$ et un seul calcul sur $g(a)$: $\langle g, q_0, s(0)\rangle\,(\langle a, q_0, 0\rangle)$. La seule règle applicable pour étiqueter la racine est $f(q_0, q_0) \xrightarrow[\lambda x_1.s(0)]{1\sim 2} q_1$. Or pour appliquer cette règle, il faudrait que les arbres étiquetant la racine des calculs sur a et sur $g(a)$ soient identiques. Or ce n'est pas le cas ($0 \neq s(0)$).

2. $g(f(g(a), g(a)))$ est accepté par le calcul :



3. $f(f(a, a), f(g(a), g(a)))$ est accepté par le calcul :



Question 2

Le langage accepté par l'automate de la question précédente n'est pas reconnaissable.

S'il était reconnaissable, son intersection avec le langage des termes $\{f(g^n(a), g^m(a))|n, m \in \mathbb{N}\}$ serait aussi reconnaissable. Or les termes de cette forme acceptés par $\mathcal{A}$ sont les termes $L = \{f(g^n(a), g^n(a)) \mid n \in \mathbb{N}\}$, qui ne forment pas un langage reconnaissable, par le lemme de pompage. En effet par contradiction, si L est reconnaissable, il existe une constante k telle que pour tout terme $f(g^n(a), g^n(a))$ de L avec $n \geq k$, on ait $f(g^{n+m}(a), g^n(a))$ dans L avec $m \neq 0$. D'où contradiction.

Question 3

Soit $\mathcal{A} = (\Sigma, \Gamma, \delta, \mathcal{Q}, \mathcal{Q}_i, \mathcal{Q}_f)$ un automate à pile initialisé avec une pile vide et acceptant par pile vide avec $\delta \subseteq \mathcal{Q} \times \Sigma \times \Gamma \times \mathcal{Q} \times (\epsilon \cup \Gamma \cup \Gamma^2)$. On définit $\hat{\mathcal{A}} = (\Sigma \cup \{0\}, \hat{\Gamma}, \Delta, \mathcal{Q}, \mathcal{Q}_f)$ avec :

$$\hat{\Gamma} = \{z(1) \mid z \in \Gamma\} \uplus \{z_0(0)\} \ ,$$
$$\Delta = \{0 \xrightarrow[z_0]{} q_0\} \cup \left\{ a \xrightarrow[\lambda z(x).\hat{\gamma}(x)]{} q' \mid (q, a, z, q', \gamma) \in \delta \right\} \ ,$$

avec $\lambda z(x).\hat{\gamma}(x) \in U_{\hat{\Gamma}}$ par composition.

La preuve de $\mathcal{L}(\hat{\mathcal{A}}) = \{\hat{w} \mid w \in \mathcal{L}(\mathcal{A})\}$ se fait par récurrence sur la taille de w.

Question 4

Si $h \in U_\Gamma$ et $e_1, \ldots, e_n$ sont des expressions ensemblistes, on commence par définir l'expression ensembliste $h(e_1, \ldots, e_n)$:

- $(\lambda x_1 \cdots x_n.x_i)(e_1, \ldots, e_n) = f_i^{-1}(f(e_1, \ldots, e_n))$, avec f un symbole d'arité n
- $(\lambda x_1, \cdots x_n.f(x_1, \ldots, x_n))(e_1, \ldots, e_n) = f(e_1, \ldots, e_n)$
- $(\lambda f(x_1, \ldots, x_n).x_i)(e) = f_i^{-1}(e)$
- $(\lambda x_1, \ldots, x_m.h(h_1(x_1, \ldots, x_m), \ldots, h_n(x_1, \ldots, x_m)))(e_1, \ldots, e_m)$
  $= h(h_1(e_1, \ldots, e_m), \cdots, h_n(e_1, \ldots, e_m))$

Le langage de l'énoncé est la plus petite solution de ϕ, l'ensemble de contraintes ensemblistes suivant: pour chaque règle $f(q_1, \ldots, q_n) \xrightarrow[h]{\sim} q$ on ajoute la contrainte :

$$h(\bigcap_{k \sim i_1} X_{q_k}, \ldots, \bigcap_{k \sim i_m} X_{q_k}) \subseteq X_q \ . \tag{1}$$

Soit $P(q) = \{u \mid \exists \rho, f.\ \rho(\epsilon) = \langle f, q, u \rangle\}$ et σ l'application telle que $\sigma(X_q) = P(q)$. Montrons que σ est solution de ϕ. Soit C une contrainte sous la forme de Eq. 1 et $f(q_1, \ldots, q_n) \xrightarrow[h]{\sim} q$ la transition associée. On montre en premier lieu par induction sur h que :

$$[\![h(e_1, \ldots, e_m)]\!]_\sigma = \{h(u_1, \ldots, u_m) \mid u_1 \in [\![e_1]\!]_\sigma, \ldots, u_m \in [\![e_m]\!]_\sigma\} \ . \tag{2}$$

On a donc $u \in [\![h(\bigcap_{k \sim i_1} X_{q_k}, \ldots, \bigcap_{k \sim i_m} X_{q_k})]\!]_\sigma$ ssi $\exists u_1 \in \bigcap_{k \sim i_1} P(q_k), \ldots, u_m \in \bigcap_{k \sim i_m} P(q_k)$ t.q. $u = h(u_1, \ldots, u_m)$. Pour tout $k \leq n$, par construction de P, il existe alors un calcul ρ_k et un symbole f_k tel que $\rho_k(\epsilon) = \langle f_k, q_k, u_{\text{rep}(k)} \rangle$ avec $\text{rep}(k)$ le représentant de la classe d'équivalence de k. On peut ainsi construire un calcul ρ de la façon suivante : $\rho(\epsilon) = \langle f, q, u \rangle$, et pour tout $k \leq n$ et toute position p dans ρ_k, $\rho(k.p) = \rho_k(p)$. Alors, C est vérifiée, $u \in [\![X_q]\!]_\sigma$.

Montrons que σ est la plus petite solution de ϕ, c'est à dire pour toute solution θ de ϕ et $q \in \mathcal{Q}$, si il existe un calcul ρ et un symbole f tels que $\rho(\epsilon) = \langle f, q, u \rangle$ alors $u \in \theta(X_q)$. Par récurrence sur la hauteur de ρ :

- si $\rho = \langle a, q, u \rangle$, alors $u \in [\![X_q]\!]_\theta$ par construction,
- si $\rho = \langle f, q, u \rangle\, (\rho_1, \ldots, \rho_n)$ avec $\rho_i(\epsilon) = \langle f_i, q_i, u_i \rangle$,
  alors il existe une transition $f(q_1, \ldots, q_n) \xrightarrow[h]{\sim} q$ dans l'automate avec $h(u_{i_1}, \ldots, u_{i_m}) = u$ et par hypothèse de récurrence, pour tout i, $u_i \in \theta(X_{q_i})$. On a donc par Eq. 2, $u \in [\![h(\bigcap_{k \sim i_1} X_{q_k}, \ldots, \bigcap_{k \sim i_m} X_{q_k})]\!]_\theta$, soit $u \in [\![X_q]\!]_\theta$ par construction.

Question 5

En partant d'un aaaaa $\mathcal{A}$, on peut construire en temps polynomial la contrainte ensembliste ϕ de la question précédente de taille polynomiale. On définit $\phi' = \phi \cup \bigcup_{q \in Q_f} X_q \subseteq \bot$. Par la question 1.5, on peut décider si ϕ et ϕ' sont satisfaisables en temps exponentiel. Alors, la langage de $\mathcal{A}$ est vide si et seulement si ϕ' est satisfaisable ou ϕ n'est pas satisfaisable.

Question 6

- La classe est trivialement close par union.
- Elle n'est pas close par intersection: par la question 3 , la cloture par intersection impliquerait que l'intersection de deux langages reconnus par des automates à pile sans ϵ transition est reconnue par un aaaaa. Or le vide d'une telle intersection n'est pas décidable, ce qui contredit la question 5.
- La classe n'est donc pas non plus close par complémentaire.