TD langages algébriques — Corrigé

David A. Madore

23 janvier 2017

INF105

Git : `d851d50 Mon Jan 23 17:10:23 2017 +0100`

Exercice 1.

Considérons le fragment simplifié suivant de la grammaire d'un langage de programmation hypothétique :

$$\begin{aligned}
\mathit{Instruction} &\to \texttt{foo} \mid \texttt{bar} \mid \texttt{qux} \mid \mathit{Conditional} \\
&\mid \texttt{begin}\ \mathit{InstrList}\ \texttt{end} \\
\mathit{Conditional} &\to \texttt{if}\ \mathit{Expression}\ \texttt{then}\ \mathit{Instruction}\ \texttt{else}\ \mathit{Instruction} \\
&\mid \texttt{if}\ \mathit{Expression}\ \texttt{then}\ \mathit{Instruction} \\
\mathit{InstrList} &\to \mathit{Instruction} \mid \mathit{Instruction}\ \mathit{InstrList} \\
\mathit{Expression} &\to \texttt{true} \mid \texttt{false} \mid \texttt{happy} \mid \texttt{trippy}
\end{aligned}$$

(Ici, les « lettres » ou tokens ont été écrits comme des mots, par exemple `foo` est une « lettre » : les terminaux sont écrits en police à espacement fixe tandis que les nonterminaux sont en italique et commencent par une majuscule. On prendra *Instruction* pour axiome.)

(1) Donner l'arbre d'analyse de : `if happy then if trippy then foo else bar else qux` ; expliquer brièvement pourquoi il n'y en a qu'un.

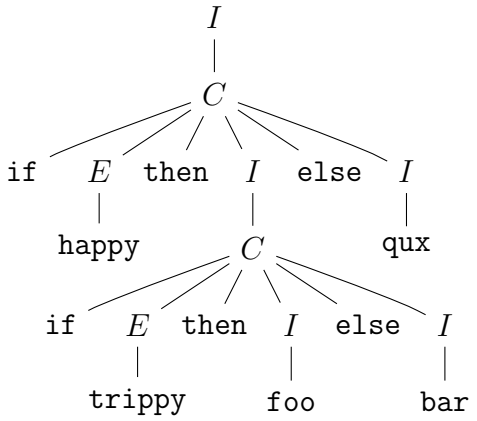
(2) Donner deux arbres d'analyse distincts de : `if happy then if trippy then foo else bar`. Que peut-on dire de la grammaire présentée ?

(3) En supposant que, dans ce langage, `begin` I `end` (où I est une instruction) a le même effet que I seul, comment un programmeur peut-il réécrire l'instruction considérée en (2) pour obtenir un comportant équivalent à l'une ou l'autre des deux interprétations ?

(4) Modifier légèrement la grammaire proposée de manière à obtenir une grammaire faiblement équivalente dans laquelle seul l'un des arbres d'analyse

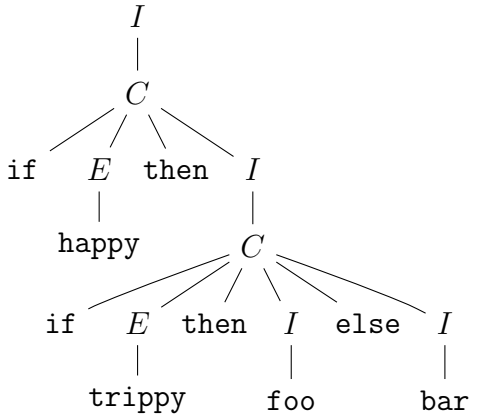
obtenus en (2) est possible (i.e., une grammaire qui force cette interprétation-là par défaut) ; on pourra être amené à introduire des nouveaux nonterminaux pour des variantes de *Instruction* et *Conditional* qui interdisent récursivement les conditionnelles sans `else`.

Corrigé. (1) L'arbre d'analyse de `if happy then if trippy then foo else bar else qux` est le suivant (en notant I, C et E pour *Instruction*, *Condition* et *Expression* respectivement) :

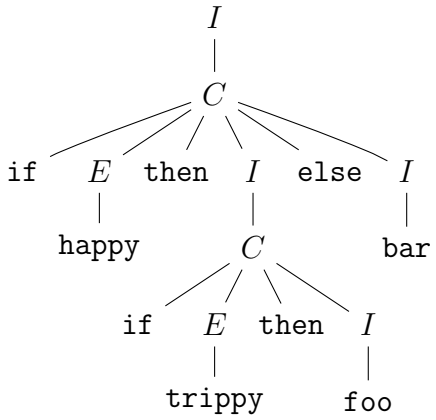


Il est le seul possible car une fois acquis que les deux `if` comportent chacun un `else`, il se construit ensuite en descendant de façon unique (l'instruction est forcément une condition, qui s'analyse en `if` E `then` I `else` I de façon unique, et chacun des morceaux s'analyse de nouveau de façon unique).

(2) Un arbre d'analyse possible consiste à associer le `else bar` avec `if trippy then foo` :



un autre consiste à associer le `else bar` avec `if happy then` ... :



La grammaire présentée est donc ambiguë.

(3) Pour forcer la première interprétation (le `else bar` se rapporte au `if trippy`), on peut écrire : `if happy then begin if trippy then foo else bar end`.

Pour forcer la seconde interprétation (le `else bar` se rapporte au `if happy`), on peut écrire : `if happy then begin if trippy then foo end else bar`.

(4) Pour forcer la première interprétation (le `else` se rapporte au `if` le plus proche possible), on peut modifier la grammaire comme suit :

$$
\begin{aligned}
\mathit{Instruction} &\rightarrow \texttt{foo} \mid \texttt{bar} \mid \texttt{qux} \mid \mathit{Conditional} \\
&\mid \texttt{begin}\ \mathit{InstrList}\ \texttt{end} \\
\mathit{InstrNoSC} &\rightarrow \texttt{foo} \mid \texttt{bar} \mid \texttt{qux} \mid \mathit{CondNoSC} \\
&\mid \texttt{begin}\ \mathit{InstrList}\ \texttt{end} \\
\mathit{Conditional} &\rightarrow \texttt{if}\ \mathit{Expression}\ \texttt{then}\ \mathit{InstrNoSC}\ \texttt{else}\ \mathit{Instruction} \\
&\mid \texttt{if}\ \mathit{Expression}\ \texttt{then}\ \mathit{Instruction} \\
\mathit{CondNoSC} &\rightarrow \texttt{if}\ \mathit{Expression}\ \texttt{then}\ \mathit{InstrNoSC}\ \texttt{else}\ \mathit{InstrNoSC} \\
\mathit{InstrList} &\rightarrow \mathit{Instruction} \mid \mathit{Instruction}\ \mathit{InstrList} \\
\mathit{Expression} &\rightarrow \texttt{true} \mid \texttt{false} \mid \texttt{happy} \mid \texttt{trippy}
\end{aligned}
$$

L'idée est d'obliger une instruction conditionnelle qui apparaîtrait après le `then` d'une conditionnelle complète à être elle-même complète (elle ne peut pas être courte, car alors le `else` devrait se rattacher à elle), et ce, récursivement. On peut montrer que la grammaire ci-dessus est inambiguë et faiblement équivalente à celle de départ.

On peut aussi fabriquer une grammaire inambiguë, faiblement équivalente à celle de départ, qui force l'autre interprétation (le `else` se rapporte au `if` le plus lointain possible), mais c'est nettement plus complexe (l'idée générale pour

apparier un `else` avec un `if...else` dans cette logoque est de demander que *soit* le `else` n'est suivi d'aucun autre `else`, *soit* toute instruction conditionnelle entre le `then` et le `else` est elle-même complète). Contrairement à la grammaire précédente, cette grammaire, bien qu'inambiguë, est probablement impossible à analyser avec un analyseur LR (ou même, déterministe). ✓

Exercice 2.

Soit $\Sigma = \{a, b\}$. On considère le langage M des mots qui *ne s'écrivent pas* sous la forme ww avec $w \in \Sigma^*$ (c'est-à-dire sous la forme d'un carré ; autrement dit, le langage M est le *complémentaire* du langage Q des carrés considéré dans l'exercice 3) : par exemple, M contient les mots a, b, ab, aab et $aabb$ mais pas ε, aa, $abab$ ni $abaaba$.

(0) Expliquer pourquoi tout mot sur Σ de longueur impaire est dans M, et pourquoi un mot $x_1 \cdots x_{2n}$ de longueur paire $2n$ est dans M si et seulement si il existe i tel que $x_i \neq x_{n+i}$.

On considère par ailleurs la grammaire hors contexte G (d'axiome S)

$$\begin{aligned} S &\to A \mid B \mid AB \mid BA \\ A &\to a \mid aAa \mid aAb \mid bAa \mid bAb \\ B &\to b \mid aBa \mid aBb \mid bBa \mid bBb \end{aligned}$$

(1) Décrire le langage $L(G, A)$ des mots dérivant de A dans la grammaire G (autrement dit, le langage engendré par la grammaire identique à G mais ayant pour axiome A). Décrire de même $L(G, B)$.

(2) Montrer que tout mot de longueur impaire est dans le langage $L(G)$ engendré par G.

(3) Montrer que tout mot $t \in M$ de longueur paire est dans $L(G)$. (Indication : si $t = x_1 \cdots x_{2n}$ est de longueur paire $2n$ et que $x_i \neq x_{n+i}$, on peut considérer la factorisation de t en $x_1 \cdots x_{2i-1}$ et $x_{2i} \cdots x_{2n}$.)

(4) Montrer que tout mot de $L(G)$ de longueur paire est dans M.

(5) En déduire que M est algébrique.

Corrigé. (0) En remarquant que si $n = |w|$ alors $|ww| = 2n$, on constate que tout mot de la forme ww est de longueur paire, et de plus, que pour un mot de longueur $2n$, être de la forme ww signifie que son préfixe de longueur n soit égal à son suffixe de longueur n ; c'est-à-dire, si $t = x_1 \cdots x_{2n}$, que $x_1 \cdots x_n = x_{n+1} \cdots x_{2n}$, ce qui signifie exactement $x_i = x_{n+i}$ pour tout $1 \leq i \leq n$.

(1) La règle $A \to a \mid aAa \mid aAb \mid bAa \mid bAb$ permet de faire à partir de A une dérivation qui ajoute un nombre quelconque de fois une lettre (a ou b) de chaque part de A, et finalement remplace ce A par a. On obtient donc ainsi exactement les mots de longueur impaire ayant un a comme lettre centrale : $L(G, A) = \{w_1 a w_2 : |w_1| = |w_2|\}$. De même, $L(G, B) = \{w_1 b w_2 : |w_1| = |w_2|\}$.

(2) Tout mot de longueur impaire est soit dans $L(G, A)$ soit dans $L(G, B)$ selon que sa lettre centrale est un a ou un b. Il est donc dans $L(G)$ en vertu des règles $S \to A$ et $S \to B$.

(3) Soit $t = x_1 \cdots x_{2n}$ un mot de M de longueur paire $2n$: d'après (0), il existe i tel que $x_i \neq x_{n+i}$. Posons alors $u = x_1 \cdots x_{2i-1}$ et $v = x_{2i} \cdots x_{2n}$. Chacun de u et de v est de longueur impaire. De plus, leurs lettres centrales sont respectivement x_i et x_{n+i}, et elles sont différentes : l'une est donc un a et l'autre un b ; mettons sans perte de généralité que $x_i = a$ et $x_{n+i} = b$. Alors $u \in L(G, A)$ d'après (1) et $v \in L(G, B)$: le mot $t = uv$ s'obtient donc par la règle $S \to AB$ (suivie de dérivations de u à partir de A et de v a partir de B) : ceci montre bien $t \in L(G)$.

(4) On a vu en (1) que tout mot dérivant de A ou de B est de longueur impaire. Un mot t de $L(G)$ de longueur paire $2n$ dérive donc forcément de AB ou de BA. Sans perte de généralité, supposons qu'il dérive de AB, et on veut montrer qu'il appartient à M. Appelons u le facteur de t qui dérive de A et v le facteur de t qui dérive de B : on sait alors (toujours d'après (1)) que u s'écrit sous la forme $x_1 \cdots x_{2i-1}$ où la lettre centrale x_i vaut a, et que v s'écrit sous la forme (quitte à continuer la numérotation des indices) $x_{2i} \cdots x_{2n}$ où la lettre centrale x_{n+i} vaut b. Alors $x_{n+i} \neq x_i$ donc le mot t est dans M d'après (0).

(5) On a $M = L(G)$ car d'après les questions précédentes, tout mot de longueur impaire est dans les deux et qu'un mot de longueur paire est dans l'un si et seulement si il est dans l'autre. On a donc montré que M est algébrique. ✓

Exercice 3.

Soit $\Sigma = \{a, b\}$. Montrer que le langage $Q := \{ww : w \in \Sigma^*\}$ constitué des mots de la forme ww (autrement dit, des carrés ; par exemple, ε, aa, $abab$, $abaaba$ ou encore $aabbaabb$ sont dans Q) n'est pas algébrique. On pourra pour cela considérer son intersection avec le langage L_0 dénoté par l'expression rationnelle $a*b*a*b*$ et appliquer le lemme de pompage.

Corrigé. Supposons par l'absurde que Q soit algébrique : alors son intersection avec le langage rationnel $L_0 = \{a^m b^n a^{m'} b^{n'} : m, n, m', n' \in \mathbb{N}\}$ est encore algébrique. Or $Q \cap L_0 = \{a^m b^n a^m b^n : m, n \in \mathbb{N}\}$. On va maintenant utiliser le lemme de pompage pour arriver à une contradiction.

Appliquons le lemme de pompage pour les langages algébriques au langage $Q \cap L_0 = \{a^m b^n a^m b^n : m, n \in \mathbb{N}\}$ considéré : appelons k l'entier dont le lemme de pompage garantit l'existence. Considérons le mot $t := a^k b^k a^k b^k$: dans la suite de cette démonstration, on appellera « bloc » de t un des quatre facteurs a^k, b^k, a^k et b^k. D'après la propriété de k garantie par le lemme de pompage, il doit exister une factorisation $t = uvwxy$ pour laquelle on a (i) $|vx| \geq 1$, (ii) $|vwx| \leq k$ et (iii) $uv^i wx^i y \in Q \cap L_0$ pour tout $i \geq 0$.

Chacun de v et de x doit être contenu dans un seul bloc, i.e., doit être de la forme a^ℓ ou b^ℓ, sinon sa répétition (v^i ou x^i pour $i \geq 2$, qui appartient à L_0

d'après (iii)) ferait apparaître plus d'alternations entre a et b que le langage L_0 ne le permet. Par ailleurs, la propriété (ii) assure que le facteur vwx ne peut rencontrer qu'un ou deux blocs de t (pas plus). Autrement dit, v et x sont contenus dans deux blocs de t qui sont identiques ou bien consécutifs [1].

D'après la propriété (i), au moins l'un de v et de x n'est pas le mot vide. Si ce facteur non trivial est dans le premier bloc a^k, l'autre ne peut pas être dans l'autre bloc a^k d'après ce qui vient d'être dit : donc uv^iwx^iy est de la forme $a^{k'}b^na^kb^k$ avec $k' > k$ si $i > 1$, qui n'appartient pas à $Q \cap L_0$, une contradiction. De même, le facteur non trivial est dans le premier bloc b^k, l'autre ne peut pas être dans l'autre bloc b^k : donc uv^iwx^iy est de la forme $a^mb^{k'}a^{m'}b^k$ avec $k' > k$ si $i > 1$, qui n'appartient pas à $Q \cap L_0$, de nouveau une contradiction. Les deux autres cas sont analogues. ✓

Remarque : Les exercices 2 et 3 mis ensemble donnent un exemple explicite d'un langage M algébrique dont le complémentaire Q n'est pas algébrique.

1. La formulation est choisie pour avoir un sens même si v ou x est le mot vide (ce qui est possible *a priori*).