Partie I. Méthodes symboliques
Alexandre DIKOVSKY

Cours 4. Autres conditions de P-apprenabilité.

1 Condition de densité finie bornée

1.1 Densité finie bornée

T.Shinohara [*New Generation Computing*, 1990, 8(4)] a proposé une condition suffisante de P-apprenabilité plus forte que celle de l'épaisseur finie. Sa condition s'applique aux systèmes formels d'une *taille bornée*. Pour ne pas se limiter à une classe particulière de systèmes formels, nous allons étendre la condition de Shinohara à une classe de ressources très riche suivante.

Définition 1. *Soit une famille indexée récursive* $\mathcal{L} = \{L(G) \mid G \in \mathcal{G}\}$. *Nous allons dire que* $\mathcal{L}$ *est une* famille indexée de grammaires, *si :*
1. Il y a une fonction Φ *qui affecte à tout* $G \in \mathcal{G}$ *un* ensemble de structures $\Phi(G) \neq \emptyset$,
2. $(\mathcal{G}, \subset)$ *est un treillis par l'inclusion* $G_i \subset G_j$ *et l'opérateur* Φ *est* monotone *sur* $\mathcal{G}$ *:*

$$G_i \subseteq G_j \Rightarrow \Phi(G_i) \subseteq \Phi(G_j).$$

3. Il y a une fonction de taille *définie sur* $\mathcal{G}$ *:*
(i) $|G| \in \mathbf{N} - \{0\}$,
(ii) $G_i \subset G_j \Rightarrow |G_i| < |G_j|$.
4. Pour un $n > 0$, *soit* $\mathcal{G}^n =_{df} \{G \in \mathcal{G} \mid |G| \leq n\}$ *et* $\mathcal{L}^n =_{df} \{L(G) \mid G \in \mathcal{G}^n\}$. *A*

Définition 2. *Soit une famille indexée de grammaires* $\mathcal{L} = \{L(G) \mid G \in \mathcal{G}\}$.
1. Une grammaire $G \in \mathcal{G}$ *est réduite par rapport à un ensemble fini de structures* I, *si*

$$I \subseteq \Phi(G) \text{ et } I \not\subseteq \Phi(G') \text{ pour toute } G' \subset G.$$

2. $\mathcal{L}$ *a une* densité finie bornée, *si quel que soit un ensemble fini de structures* I, *la classe des réssources* $G \in \mathcal{G}$ *réduites par rapport à* I *est finie.*

Remarque 1. *Il y a infiniment beaucoup de CDG différentes dans* $\mathbf{GCD}^2$ *et de grammaires contextuelles dans* $\mathbf{GCt}^2$. *Par ailleurs, les familles* $\mathbf{GCD}^n$ *et* $\mathbf{GCt}^n$ *ont une densité finie bornée pour tout* n. *Expliquons le par exemple pour* $\mathbf{GCt}^n$.

Proposition 1. *Toute famille* $\mathbf{GCt}^n$ *a une densité finie bornée.*

Preuve. Soit un ensemble fini de motifs D, sa taille $d = |D|$ et une grammaire $G \in \mathbf{GCt}^n$ réduite par rapport à D. Soit $drvR(G, D)$: l'ensemble des dérivations $S \Rightarrow_G^* x, x \in D$, avec pas plud d'une répétition de motifs intermédiaires et $rules(G, D)$: le sous ensemble de G qui contient toutes les règles utilsées dans $drvR(G, D)$. Selon l'hypothèse, $|rules(G, D)| \leq n$. G étant contextuelle, $|r| \leq l_D = max\{|x| \mid x \in D\}$ pour toute règle $r \in rules(G, D)$. Finalement, le nombre des terminaux dans les règles dans $rules(G, D)$ est borné par la taille d de D et le nombre des nonterminaux dans ces règles est borné par $l_D \times |rules(G, D)| \leq l_D \times n$. Par conséquent, le nombre des grammaires contextuelles deux-à-deux non isomorphes [1] réduites par rapport à D est borné par $(d + l_D \times n)^{l_D \times n}$. □

Théorème 1 (Shinohara, ibid). *Si une famille indexée de grammaires* $\mathcal{L}$ *a une densité finie bornée, alors toute famille* $\mathcal{L}^n, n \geq 1$, *a une élasticité finie.*

Preuve. Nous allons prouver le théorème par l'induction sur n.
1. L'élasticité finie de $\mathcal{L}^1$ est une conséquence évidente du point 4 de la définition 1.
2. Soit établie l'élasticité finie de toute famille $\mathcal{L}^n$, $n < i$ $(i \geq 2)$. Supposons que la famille $\mathcal{L}^i$ a une élasticité infinie. Dans ce cas il existent deux séquences infinies

$$\sigma = s_0, s_1, s_2, \ldots \tag{1}$$

et

$$G_0, G_1, G_2, \ldots \text{dans } \mathcal{G}^i \tag{2}$$

telles que

$$\{s_0, \ldots, s_{k-1}\} \subseteq \Phi(G_k) \text{ mais } s_k \notin \Phi(G_k) \text{ pour tous } k \geq 1.$$

Remarquons que selon le point 3 de la définition 1 l'ordre de l'inclusion $\subset$ sur $\mathcal{G}$ est bien fondé. Cela implique que quel que soient un ensemble fini D de motifs et une grammaire $G \in \mathcal{G}$ il existe une grammaire $G' \subseteq G$ réduite par rapport à D. Effectivement, au cas contraire d'après la définition 2 il existe une suite infinie de sous grammaires $G = G^{(0)} \supset G^{(1)} \supset G^{(2)} \supset \ldots$ toute vérifiant l'inclusion : $D \subseteq L(G^{(i)})$. Notons G'_k la sous grammaire de G_k réduite par rapport à $\{s_0, \ldots, s_{k-1}\}$.

Soit la fonction

$$h(k) =_{df} min\{j \mid G'_k \text{ est réduite par rapport à } \{s_0, \ldots, s_j\}\}. \tag{3}$$

[1] Deux grammaires sont *isomorphes* : $G_1 \equiv G_2$ si elles sont identiques à un renommage des nonterminaux près.

Remarque 2.
(*i*) $h(k) < k$ *car* $a_k \notin \Phi(G'_k)$.
(*ii*) La fonction h n'est bornée par aucune constante c. *Effectivement, au cas contraire toute* G_m, $m > c$, *dans* (2) *est réduite par rapport à* $\{s_0, \ldots, s_c\}$, *se qui contredit à la densité finie bornée de* $\mathcal{L}$.

Alors, il existe une suite infinie ascendante

$$1 < h(k_1) < h(k_2) < \ldots \text{ telle que } k_1 < k_2 < \ldots \tag{4}$$

Soit deux préfixes consécutifs de (1) :

$$\sigma[h(k_j) - 1] = \{s_0, \ldots, s_{h(k_j)-1}\} \text{ et } \sigma[h(k_j)] = \{s_0, \ldots, s_{h(k_j)}\}.$$

Par définition de h, G'_{k_j} est réduite par rapport à $\sigma[h(k_j)]$. Selon (3), G_{k_j} n'est pas réduite par rapport à $\sigma[h(k_j) - 1]$. C'est-à-dire, il y a une sous grammaire $G''_{k_j} \subset G'_{k_j}$ telle que $\sigma[h(k_j) - 1] = \{s_0, \ldots, s_{h(k_j)-1}\} \subseteq \Phi(G''_{k_j}) \subseteq \Phi(G'_{k_j}) \subseteq \Phi(G_{k_j})$. Dans ce cas, $|G''_{k_j}| < i$, car $G''_{k_j} \subset G_{k_j}$ implique $|G''_{k_j}| < |G_{k_j}| \leq i$. En même temps, $s_{h(k_j)} \notin \Phi(G''_{k_j})$ par définition de h parce que k_j est l'index minimal tel que $G'_{k_j} \subset G_{k_j}$ est réduite par rapport à $\{s_0, \ldots, s_{h(k_j)}\}$.

Alors, on a trouvé deux suites

$$s_0, s_{h(k_1)}, s_{h(k_2)} \ldots \text{ et } \{G''_{k_1}, G''_{k_2}, \ldots\} \in \mathcal{L}^{i-1}$$

qui contredisent l'élasticité finie de $\mathcal{L}^{i-1}$. □

Corollaire 1. *Si une famille indexée de grammaires* $\mathcal{L}$ *a une densité finie bornée, alors toute famille* $\mathcal{L}^n, n \geq 1$, *est P-apprenable.*

Corollaire 2. *Les familles indexées* $\mathbf{GCt}^n$ *des grammaires contextuelles à au plus de n règles sont toutes P-apprenables.*

1.2 Apprentissage des GCD rigides et k-valuées

[Béchet & Dikovsky & Foret & Moreau, *9th Conf. on Formal Grammars'2004*]

Pour la raison des types itérés les GCD s'avèrent non-apprenables à partir de chaînes.

Construction des points limites.

Définition 3. *Soit* S, A, B *trois catégories élémentaires. On définit les GCD rigides suivantes :*
$C_0 = S$
$C_{n+1} = (C_n/A^*)/B^*$
$G_0 = \{a \mapsto A, b \mapsto B, c \mapsto C_0\}$
$G_n = \{a \mapsto A, b \mapsto B, c \mapsto [C_n]\}$
$G_* = \{a \mapsto A, b \mapsto A, c \mapsto [S/A^*]\}$

Ces constructions produisent le point limite suivant.

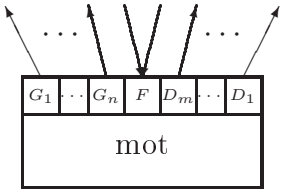
Théorème 2. *On a* $\mathcal{L}(G_n) = \{c(b^*a^*)^k \mid k \leq n\}$ *et* $\mathcal{L}(G_*) = c\{b, a\}^*$.

Corollaire 3. *Ce point limite montre la non-apprenabilité à partir de chaînes des classes de GCD rigides autorisant les types itérés* (A^*).

Réseaux de dépendances (typés / non-typées). Comme c'etait le cas des GC classiques, on peut trouver pour les GCD sans types itérés une structure qui, d'un côté, représente (complètement ou partiellement) les preuves et les structures de dépendances et, d'un autre côté, garantit la densité finie bornée.

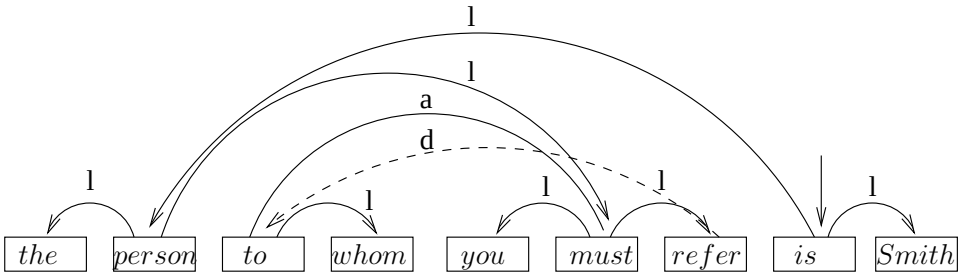
Définition 4. *Un nœud non-typé est une liste de sommets appelés* **cases** *associés à un mot. Un nœud est un nœud non-typé dont chaque case correspond à une catégorie élémentaire de l'une des catégories associées au mot.*

Voici le nœud qui correspond à la catégorie $[G_1 \backslash \cdots \backslash G_n \backslash F / D_m / \cdots / D_1]$:

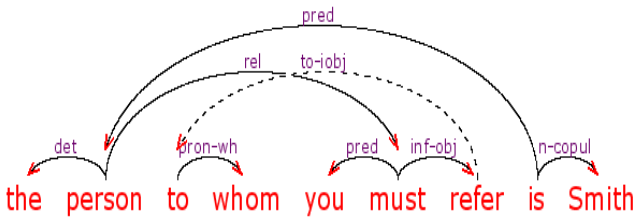


Définition 5. *Un* réseau de dépendances *(non-typées) est une liste de nœuds (non-typés) connectés entre eux par des dépendances locales (l), ancrées (a) et distantes (d) qui correspondent à l'analyse de la phrase.*

Exemple 2. *Par exemple, le réseau suivant :*



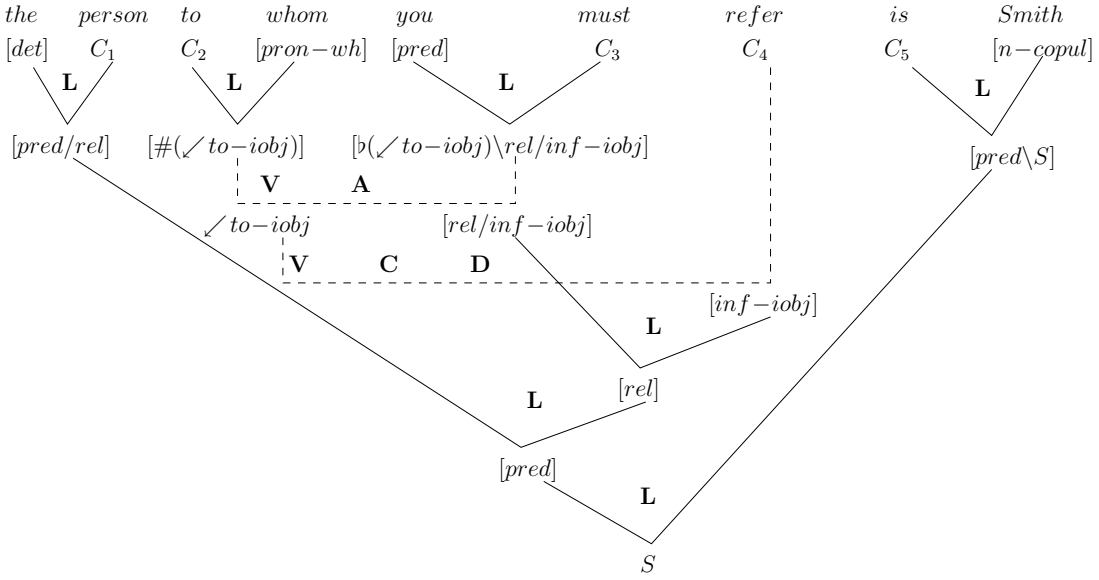
dont l'une des cases n'est pas connectée [2] *représente partiellement l'arbre de dépendances :*



Cet arbre de dépendances est engendré par la GCD :

[2] Il s'agit de la conclusion principale du réseau de dépendances : la catégorie élémentaire S. Celle-ci est désignée sur la figure par la flèche sans origine sur "is".

$the \mapsto [det]$
$person \mapsto [det\backslash pred/rel] = C_1$
$to \mapsto [\#^l(\swarrow to{-}iobj)/pron{-}wh] = C_2$
$whom \mapsto [pron{-}wh]$
$you \mapsto [pred]$
$must \mapsto [pred\backslash\flat^l(\swarrow to{-}iobj)\backslash rel/inf{-}obj] = C_3$
$refer \mapsto [\nwarrow to{-}iobj\backslash inf{-}obj] = C_4$
$is \mapsto [pred\backslash S/n{-}copul] = C_5$
$Smith \mapsto [n{-}copul]$
En voici une preuve :



Définition 6. *Un réseau de dépendances (non-typées) N est correct pour une GCD G sans types itérés s'il existe une analyse de la séquence des mots décrite par N qui correspond aux nœuds (non-typés) et aux dépendances de N.*[3]
Le langage de réseaux de dépendances de G, noté $\mathcal{NL}(G)$, est l'ensemble des réseaux de dépendances corrects pour G.
Le langage de réseaux de dépendances non-typées de G, noté $\mathcal{UNL}(G)$, est l'ensemble des réseaux de dépendances non-typées corrects pour G.

Pour montrer la P-apprenabilité des GCD rigides sans types itérés, nous allons considérer la famille indexée $\mathcal{UNL}(GCD)$, où $\Phi(G)=_{df}\mathcal{UNL}(G)$.

Lemme 1. *Pour tout ensemble fini $X \subseteq \mathcal{UNL}(G)$, l'ensemble des langages de réseaux de dépendances non-typées générés par des GCD rigides sans types itérés réduites par rapport à X est fini.*

Preuve. Si une grammaire rigide de réseaux de dépendances non-typées $G = (W, \mathbf{C}, S, g)$ utilisant les types Tp est réduite par rapport à X, alors tout mot n'apparaissant pas dans l'un des réseaux contenus dans X est associé à travers g à l'ensemble vide. Tous les autres

[3]Une structure de dépendances correspond à un réseau de dépendances si le type de chaque dépendance (locale, ancrée ou distante) du réseau est identique à celui du mot subordonné de la dépendance correspondante dans la structure.

mots sont nécessairement associés à exactement un type de Tp (car la grammaire est rigide). Le nombre de cases gauches et droites est déterminé par les occurences du mot dans les réseaux de dépendances non-typées, et doit être identique pour toutes les occurences du fait que le langage qu'on doit apprendre correspond à une grammaire rigide. Si le nombre total de cases (gauches, droites plus une case pour la catégorie principale) de l'ensemble des mots de X est borné par m, et si n est le nombre de mots dans X, alors la taille de l'ensemble des grammaires (non équivalentes[4] entre elles) réduites par rapport à X est bornée par le nombre de partitions d'un ensemble à $n \times m$ éléments. □

Corollaire 4. *La famille indexée $\mathcal{UNL}(GCD^{rigides})$ a une densité finie bornée et donc a une élasticité finie.*

Ainsi les GCD rigides sans types itérés s'avèrent P-apprenables à partir de réseaux de dépendances non-typées. De plus, on peut adapter l'algorithme RG aux réseaux de dépendances.

Algorithme RGD d'apprentissage à partir de réseaux :

1. Les occurences de chaque mot sont collectés en même temps que ses arités gauches et droites dans la séquence en entrée. L'algorithme échoue si un mot est utilisé avec des arités différentes.
2. A chaque mot w on associe $n+m+1$ variables, qui correspondent aux n types arguments à gauche, aux m types arguments à droite, et au type tête : $X^w_{-n}, \ldots, X^w_{-1}, X^w_0, X^w_1, \ldots, X^w_m$ (où X^w_0 est le type tête).
3. Ensuite on construit un système d'équations sur les variables, basé sur les dépendances qui apparaissent dans les réseaux de dépendances non-typées en entrée : une dépendance correspond aux deux variables présentes au début et à la fin de celle-ci. Ces deux variables sont unifiées. De plus, les dépendances des arguments (resp. du type tête) doivent être orientées depuis le nœud courant vers un autre (resp. depuis un autre nœud vers celui-ci)[5].
4. Le système d'équations résultant est résolu et un type élémentaire est associée à chaque classe d'équivalence des variables, notée $\overline{X^w_i}$.
5. Pour chaque mot w associé aux variables $X^w_{-n}, \ldots, X^w_{-1}, X^w_0, X^w_1, \ldots, X^w_m$, la GCD produite en sortie associe le type

$$[\overline{Y^w_{-n}}\backslash_{-n} \ldots \backslash_{-2}\overline{Y^w_{-1}}, \backslash_{-1}\overline{Y^w_0}/_1\overline{Y^w_1}/_2 \ldots /_m\overline{Y^w_m}]$$

à w, où :
   - Y^w_i est X^w_i si la dépendance est locale et qu'il y a toujours une seule dépendance entrant (ou sortant) de cette case. Si $i \neq 0$, $\backslash_i$ est $\backslash$ (de même $/_i$ est $/$ dans le cas d'un argument droit).
   - Sinon, Y^w_i est $\#(\swarrow X^w_i)$ (resp. $\#(\searrow X^w_i)$) si la case argument gauche (resp. argument droit) est la fin d'une dépendance ancrée.

[4]Deux grammaires sont équivalentes si leurs langages sont identiques. Le fait qu'une bijection existe entre les ensembles des catégories élémentaires des deux grammaires est une condition suffisante à cela.

[5]L'algorithme renvoie un échec si cette condition n'est pas remplie.

– Y_i^w est $\nwarrow X_i^w$ (resp. $\nearrow X_i^w$) si la case argument gauche (resp. argument droit) est le début d'une dépendance distante. $\backslash_i$ est $\backslash$ (de même $/_i$ est $/$).
– Y_0^w est $\swarrow X_0^w$ (resp. $\searrow X_0^w$) si la case argument gauche (resp. argument droit) est la fin d'une dépendance distante.

Comme dans le cas des GC, on peut prouver le théorème

Théorème 3. *L'algorithme RGD apprend les grammaires rigides de réseaux de dépendances non-typées.*

Finalement, en utilisant la réduction de Kanazawa par une relation F-valuée, on obtient

Corollaire 5. *Les GCD rigides et k-valuées sans types itérés sont P-apprenables à partir de chaînes.*

2 Condition de réversibilité

2.1 Automates finis d'arbres réversibles

[J. Besombes, J.-Y.Marion, *TAL*, 2003, 44(1)]

Nous n'allons pas distinguer entre les arbres et les termes. Pour tout ensemble de symboles $\mathcal{F}$ et tout ensemble de variables $\mathcal{X}$, les arbres sont définis par : $f \in \mathcal{F}$ est un arbre et si $f \in \mathcal{F}$ et $t_1, \ldots, t_k$ sont des arbres, alors $f(t_1, \ldots, t_k)$ est un arbre. L'ensemble des arbres est noté $\mathcal{T}(\mathcal{F}, \mathcal{X})$. $\mathcal{T}(\mathcal{F}) =_{df} \mathcal{T}(\mathcal{F}, \emptyset)$ est l'ensemble des arbres terminaux. $sub(t)$ note l'ensemble des sous-arbres de t, $sub(E) =_{df} \bigcup_{t \in E} sub(t)$.

Un contexte est un arbre $c[\diamond]$ à une seule occurrence de la variable spéciale $\diamond$ distinguée par les crochets. Le résultat de composition $c[\diamond](\diamond \leftarrow u)$ est noté $c[u]$.

Définition 7. *Un* automate fini d'arbres (AFA) $A = (\mathcal{F}, \mathcal{Q}, \mathcal{Q}_F, \rightarrow_A)$ *a les transitions de la forme*

$$f(q_1, \ldots, q_n) \rightarrow_A q, \text{ où } q, q_1, \ldots, q_n \in \mathcal{Q}, f \in \mathcal{F}, n \geq 0 \tag{5}$$

A est déterministe (AFDA) *si* $\rightarrow_A$ *est une fonction.*
Les transitions (5) *sont étendues aux arbres à états :* $t[f(q_1, \ldots, q_n)] \rightarrow_A t[q]$. *Le* langage d'arbres reconnu par *A est l'ensemble* $\mathcal{L}_A =_{df} \{t \in \mathcal{T}(\mathcal{F}) \mid t \rightarrow_A^* q_f \in \mathcal{Q}_F\}$[6].

A est réversible *s'il est déterministe dans les deux sens :*
1) *A est déterministe,*
2) $\mathcal{Q}_F = \{q_f\}$ *est singleton et*
3) *si* $f(q_1, \ldots, p_1, \ldots, q_n) \rightarrow_A q$ *et* $f(q_1, \ldots, p_2, \ldots, q_n) \rightarrow$

Exemple 3. *L'AFDA A_1 à état final q_1 :*

$true \rightarrow_{A_1} q_1$	$false \rightarrow_{A_1} q_0$
$Not(q_0) \rightarrow_{A_1} q_1$	$Not(q_1) \rightarrow_{A_1} q_0$
$Xor(q_0, q_0) \rightarrow_{A_1} q_0$	$Xor(q_0, q_1) \rightarrow_{A_1} q_1$
$Xor(q_1, q_0) \rightarrow_{A_1} q_1$	$Xor(q_1, q_1) \rightarrow_{A_1} q_0$

est réversible et il reconnaît le langage réversible d'arbres qui consiste des expressions booléennes vraies.

Soit un AFDA A. $\equiv_A$ note la congruence sur les arbres induite par A :

$$t \equiv_A u =_{df} \exists q \in \mathcal{Q}\ (t \rightarrow_A^* q\ \wedge\ u \rightarrow_A^* q).$$

Pour tout langage d'arbres $\mathcal{L}$, sa *congruence principale* est définie par :

$$t \equiv_{\mathcal{L}} u =_{df} \forall c[\diamond]\ (c[t] \in \mathcal{L} \Leftrightarrow c[u] \in \mathcal{L}).$$

Le théorème classique de Myhill-Nerode est vrai pour les langages d'arbres :

Théorème 4. *Les propriétés suivant d'un langage d'arbres $\mathcal{L}$ sont équivalents :*
a. La congruence $\equiv_{\mathcal{L}}$ a un rang (le nombre des classes) fini ;
b. L'automate $A_{\mathcal{L}}$ avec les états $\mathcal{Q} =_{df} \{[t]_{\mathcal{L}} \mid t \in \mathcal{T}(\mathcal{F})\}$, avec les états finaux $\mathcal{Q} =_{df} \{[t]_{\mathcal{L}} \mid t \in \mathcal{L}\}$ et avec les transitions $f([t_1]_{\mathcal{L}}, \ldots, [t_n]_{\mathcal{L}}) \rightarrow_{\mathcal{A}_{\mathcal{L}}} [f(t_1, \ldots, t_n)]_{\mathcal{L}}$ est l'AFDA minimal qui reconnaît $\mathcal{L}$.

Lemme 2 (principal). *Soit un AFDA $\mathcal{A}$ minimal et réversible.*
Si $c[t], c[u] \in \mathcal{L}_{\mathcal{A}}$, alors $t \equiv_{\mathcal{A}} u$.

Preuve. 1. Si $c = \diamond$, alors $t, u \in \mathcal{L}_{\mathcal{A}}$. Etant donné qu'il y a un seul état final q_f, $t \rightarrow_{\mathcal{A}} q_f$ et $u \rightarrow_{\mathcal{A}} q_f$. Alors $t \equiv_{\mathcal{A}} u$.
2. Soit $c[\diamond] = c'[f(t_1, \ldots, \diamond, \ldots, t_n)]$ et $c[t], c[u] \in \mathcal{L}_{\mathcal{A}}$. Alors $|c'[\diamond]| < |c[\diamond]|$. Selon l'hypothèse, $f(t_1, \ldots, t, \ldots, t_n) \rightarrow_{\mathcal{A}} q$ et $f(t_1, \ldots, u, \ldots, t_n) \rightarrow_{\mathcal{A}} q$ pour un même q. $\mathcal{A}$ est déterministe. Il existent donc, des états uniques $q_1, \ldots, q_n$ tels que $t_1 \rightarrow_{\mathcal{A}} q_1, \ldots, t_n \rightarrow_{\mathcal{A}} q_n$. Ainsi, selon la réversibilité de $\mathcal{A}$, $t \rightarrow_{\mathcal{A}} p$ et $u \rightarrow_{\mathcal{A}} p$ pour un même état unique p. □

Corollaire 6 (principal). *Un langage d'arbres $\mathcal{L}$ est réversible ssi pour tous arbres t, u, s'ils partagent un contexte quelconque $c[\diamond]$ dans $\mathcal{L}$ (c'est-à-dire $c[t], c[u] \in \mathcal{L}$), alors ils sont équivalents (partagent tous les contextes) dans $\mathcal{L}$.*

Preuve. Appliquer le lemme principal à l'AFDA minimal $\mathcal{A}_{\mathcal{L}}$ (qui est réversible). □

2.2 Apprentissage des automates finis d'arbres réversibles

Ensembles caractéristiques. Soit un langage d'arbres réversible $\mathcal{L}$ et son AFDA minimal $\mathcal{A}_{\mathcal{L}}$. Pour tout son état q il existent :
1) un arbre minimal (relativement à la taille) $rt(q)$, tel que $rt(q) \rightarrow_{\mathcal{A}_{\mathcal{L}}} q$ et
2) un contexte minimal $C_q[\diamond]$, tel que $C_q[q] \rightarrow_{\mathcal{A}_{\mathcal{L}}} q_f$.

Définition 8. L'ensemble d'exemples caractéristiques *pour $\mathcal{L}$, noté $CS(\mathcal{L})$, est un ensemble minimal (relativement à la cardinalité) qui contient :*
(i) l'arbre $C_q[rt(q)]$ pour tout état $q \in \mathcal{Q}_{\mathcal{L}}$,
(ii) l'arbre $C_q[f(rt(q_1), \ldots, rt(q_n)]$ pour toute transition $f(q_1, \ldots, q_n) \rightarrow_{\mathcal{A}_{\mathcal{L}}} q$.

Remarque évidente : $CS(\mathcal{L}) \subseteq \mathcal{L}$.

Exemple 4. *Pour l'automate* A_1 *de l'exemple 3 nous pouvons choisir :* $rt(q_1) = true$, $rt(q_0) = false$, $C_{q_1}[\diamond] = Xor(\diamond, false)$ *et* $C_{q_0}[\diamond] = Not(\diamond)$. *Alors, l'ensemble caractéristique correspondant* $CS(\mathcal{L}_{A_1})$ *est*
$\{Not(false),\ Xor(true, false),\ Xor(Not(false), false),\ Not(Not(true)),$
$Not(Xor(false, false)),\ Xor(Xor(false, true), false),\ Xor(Xor(true, false), false),$
$Not(Xor(true, true))\}$.

Théorème 5. *Soit deux langages réversibles d'arbres* $\mathcal{L}_1, \mathcal{L}_2$ *sur* $\mathcal{T}(\mathcal{F})$. *Si* $CS(\mathcal{L}_1) \subseteq \mathcal{L}_2$, *alors* $\mathcal{L}_1 \subseteq \mathcal{L}_2$.

Preuve. Il suffit de prouver pour tout état $q \in \mathcal{Q}_1$ et pour tout arbre t que

$$\text{(6)} \qquad \textit{si } C_q[t] \in \mathcal{L}_1, \textit{ alors } C_q[t] \in \mathcal{L}_2.$$

Effectivement, si (6) est vrai pour q_f, alors $C_{q_f}[t] = t \in \mathcal{L}_1$ implique $t \in \mathcal{L}_2$. (6) est prouvé par induction sur $|t|$. □

Corollaire 7. *La famille* $\mathcal{L}(LAR)$ *des langages réversibles est P-apprenable.*

Preuve. C'est une conséquence directe du théorème d'Angluin. Effectivement, $\mathcal{L}(LAR)$ est une famille indexée, dans laquelle on peut énumérer $CS(\mathcal{L})$ pour tout AFDA réversible $\mathcal{A}_{\mathcal{L}}$. Il reste de remarquer que selon théorème 5, $CS(\mathcal{L}_1) \subseteq \mathcal{L}_2$ exclut $\mathcal{L}_2 \subsetneq \mathcal{L}_1$. Ainsi les ensembles $CS(\mathcal{L})$ sont révélateurs pour $\mathcal{L} \in \mathcal{L}(LAR)$. □

Algorithme d'apprentissage efficace BM.

Soit une suite d'exemples $E \in \mathcal{T}(\mathcal{F})$.
CONSTRUIRE l'AFDA $\mathcal{A}_0(E)$ suivant :
Etats : un état (noté $[t]$) pour tout sous-arbre $t \in sub(E)$;
Etats finaux : $[t]$ pour $t \in E$;
Transitions : $f([t_1], \ldots, [t_n]) \to_0 [f(t_1, \ldots, t_n)]$ pour tout sous-arbre $f(t_1, \ldots, t_n) \in sub(E)$.
Remarque : $\mathcal{L}_{\mathcal{A}_0(E)} = E$.
$i := 0$;
TANT QUE $\mathcal{A}_i(E)$ n'est pas réversible
SELECTIONNER une règle R parmi :

R₁	si $f(p_1, \ldots, p_m, q, p_{m+1}, \ldots, p_n) \to_i p$ et $f(p_1, \ldots, p_m, q', p_{m+1}, \ldots, p_n) \to_i p$, alors $q' := q$
R₂	si $f(p_1, \ldots, p_n) \to_i p$ et $f(p_1, \ldots, p_n) \to_i p'$, alors $p' := p$
R₃	si q_{f1}, q_{f2} sont deux états finaux de $\mathcal{A}_i(E)$, alors $q_{f2} := q_{f1}$

$\mathcal{A}_{i+1}(E) := R(\mathcal{A}_i(E))$
END TQ
RENVOYER $\mathcal{A}_i(E)$

Exemple 5. *Soit* $E = \{f(a,b), f(g(a),b), f(g(g(a)),b)\}$.
1. $\mathcal{A}_0(E)$:
$\mathcal{Q}_0(E) = \{[a],[b],[f(a,b)],[g(a)],[f(g(a),b)],[g(g(a))],[f(g(g(a)),b)]\}$;
$\mathcal{Q}_{F_0}(E) = \{[f(a,b)],[f(g(a),b)],[f(g(g(a)),b)]\}$;
$\rightarrow_0$:

$$\begin{array}{cc} a \rightarrow_0 [a] & b \rightarrow_0 [b] \\ f([a],[b]) \rightarrow_0 [f(a,b)] & g([a]) \rightarrow_0 [g(a)] \\ f([g(a)],[b]) \rightarrow_0 [f(g(a),b)] & g([g(a)]) \rightarrow_0 [g(g(a))] \\ f([g(g(a))],[b]) \rightarrow_0 [f(g(g(a)),b)] & \end{array}$$

2. En appliquant trois fois $\mathbf{R_3}$, *on obtient :* $[f(a,b)] = [f(g(a),b)] = [f(g(g(a)),b)]$ *et le nouvel ensemble de transitions :*

$$\begin{array}{cc} a \rightarrow_3 [a] & b \rightarrow_3 [b] \\ f([a],[b]) \rightarrow_3 [f(a,b)] & g([a]) \rightarrow_3 [g(a)] \\ f([g(a)],[b]) \rightarrow_3 [f(a,b)] & g([g(a)]) \rightarrow_3 [g(g(a))] \\ f([g(g(a))],[b]) \rightarrow_3 [f(a,b)] & \end{array}$$

3. En appliquant deux fois $\mathbf{R_1}$, *on obtient :* $[a] = [g(a)] = [g(g(a))]$ *pour arriver au point de stabilisation :*
$\mathcal{A}_5(E)$:
$\mathcal{Q}_5(E) = \{[a],[b],[f(a,b)]\}$;
$\mathcal{Q}_{F_5}(E) = \{[f(a,b)]\}$;
$\rightarrow_5$: $a \rightarrow_5 [a], b \rightarrow_5 [b]$, $f([a],[b]) \rightarrow_5 [f(a,b)]$, $g([a]) \rightarrow_5 [a]$.

Lemme 3. *1.* **BM** *se termine après au plus* $n_E = |sub(E)| = \mathcal{O}(|E|^2)$ *étapes, car au moins un état est éliminé à chaque étape.*
2. L'AFDA $\mathbf{BM}(E) =_{df} \mathcal{A}_{n_E}(E))$ *est réversible.*
3. Pour tout langage réversible $\mathcal{L} \in \mathcal{T}(\mathcal{F})$ *si* $E \subseteq \mathcal{L}$, *alors* $\mathbf{BM}(E) \subseteq \mathcal{L}$.

Théorème 6. *L'algorithme* **BM** *P-apprend tout langage d'arbres réversible.*

Preuve. Si E est une séquence d'exemples du langage d'arbres cible $\mathcal{L}_c$, alors il existe un préfixe $E[K]$, tel que $CS(\mathcal{L}_c) \subseteq E[K]$. Alors selon lemme 3(3) $\mathbf{BM}(E[K]) \subseteq \mathcal{L}_c$ et selon lemme 3(2) $\mathbf{BM}(E[K])$ est réversible. Alors, selon théorème 5 $\mathcal{L}_c \subseteq \mathbf{BM}(E[K]) \subseteq \mathcal{L}_c$. □

Applications. Toutes les grammaires dont l'ensemble d'arbres de dérivations (ou de structures syntaxiques) est langage d'arbres rationnel (les grammaires hors contexte (HC), les grammaires de dépendance projectives lexicalisées (PL), les grammaires catégorielles classiques, etc.) sont susceptibles d'être contrainte par une condition de réversibilité. La sous-classe correspondante des grammaires réversibles va être P-apprenable. C'est ainsi qu'on peut prouver que les grammaires HC réversibles sont P-apprenables [Y.Sakakibara. *Inf. & Computation*, 1992 (97)], et que les grammaires de dépendance PL réversibles le sont aussi [J. Besombes, J.-Y.Marion, . *3d Workshop on Learning in Logic*, 2001].

Remarque 3. *L'ensemble d'arbres de dérivations (ou de structures syntaxiques) des GCD n'est pas un langage d'arbres rationnel. Alors, l'algorithme* **BM** *ne s'y applique pas.*