Identification des langages rationnels à résiduels k-disjoints

Alain Terlutte, Fabien Torre

Identification des langages rationnels à résiduels k-disjoints

Alain Terlutte, Fabien Torre

LIFL Grappa et INRIA Mostrare

Résumé : Nous définissons les familles de *langages rationnels à résiduels k-disjoints* et nous présentons les possibilités d'identification de ces familles. Chaque famille de langages rationnels à résiduels k-disjoints contient la famille correspondante de langages k-réversibles. L'union des familles, pour $k \in \mathbb{N}$, couvre tous les langages rationnels. Nous montrons que chacune de ces familles est identifiable à la limite, en temps et données polynômiaux à partir d'exemples positifs, en les représentant par des AFD.

Mots-clés : inférence grammaticale, identification à la limite, langages rationnels, langages résiduels.

1 Introduction

Cet article traite d'Inférence Grammaticale et nous nous plaçons en particulier dans le modèle d'*identification à la limite* défini par (Gold, 1967).

Il s'agit de reconnaître des langages à partir d'exemples. Un langage est un ensemble de mots et appartient à une famille de langages. Une famille de langages $\mathcal{L}$ est définie par une classe de représentations $\mathcal{H}_{\mathcal{L}}$. Pour décrire une même famille de langages, il peut exister plusieurs classes de représentations. Par exemple, les langages rationnels peuvent être représentés par des AFN (Automates Finis Non déterministes), des AFD, des expressions rationnelles, des grammaires régulières, mais aussi des couples (langage local, morphisme strictement alphabétique), etc.

Ensuite, il existe plusieurs types de présentations des exemples. On peut ne présenter que des exemples positifs. Dans ce cas, une présentation S d'un langage L sera une séquence infinie de mots w_i appartenant au langage. On peut aussi présenter des exemples positifs et négatifs. Dans ce cas, une présentation S sera une séquence infinie de couples (w_i, b_i), dans lesquels b_i est un booléen indiquant si le mot appartient au langage. On pourrait aussi avoir une séquence de mots associés à une probabilité d'appartenir au langage, etc. Les n premiers mots de la présentation sont notés S_n.

Un algorithme d'apprentissage est alors une application d'une présentation d'exemples (ou d'un sous-ensemble de cette présentation) vers une représentation d'un langage. Différents modèles ont été proposés pour introduire des contraintes de polynômialité (Pitt, 1989; de la Higuera, 1997). La polynômialité peut porter sur le temps, sur le nombre d'erreurs de prédiction, sur la taille de l'échantillon caractéristique, etc.

Lorsque l'on fixe des contraintes de polynômialité, le choix de la classe de représentations $\mathcal{H}_{\mathcal{L}}$ prend tout son sens. En effet le polynôme aura pour paramètre la taille de la représentation et ces tailles varient, pour un même langage, selon la représentation choisie.

Dans ces modèles, la famille des langages reconnaissables, représentée par des AFD, est identifiable à la limite en utilisant des exemples positifs et négatifs (Oncina & García, 1992). Avec une présentation d'exemples positifs, aucune famille contenant tous les langages finis et au moins un langage infini ne peut être identifiée. Ainsi la famille des langages rationnels n'est pas identifiable par présentation d'exemples positifs seuls.

Par contre, d'autres familles, représentées par des AFD, disposent de résultats d'apprenabilité à la limite en utilisant des exemples positifs uniquement. C'est le cas pour les familles de langages k-testables (García & Vidal, 1990) et pour les familles de langages k-réversibles (Angluin, 1982).

Angluin (1982) a défini les langages k-réversibles par une propriété de l'automate déterministe minimal qui reconnaît ce langage : l'automate miroir de cet automate déterministe minimal est déterministe avec une anticipation de k lettres. Elle a montré que chaque famille de langages k-réversibles était identifiable à la limite par présentation d'exemples positifs, et ceci en temps polynômial et avec un échantillon polynômial. Ce résultat est un des résultats fondamentaux de l'inférence grammaticale.

Mais des hiérarchies telles que les k-réversibles ou les k-testables ne couvrent pas la totalité des langages rationnels. Par exemple, les langages $L_1 = (a+b)^*a(a+b)^*$ et $L_2 = (a+b)^*(a+c)^*$ ne sont réversibles pour aucun k. Par conséquent, les langages rationnels restent hors d'atteinte des algorithmes d'apprentissage.

Nous cherchons dans cet article à dépasser ces résultats négatifs sur les rationnels par l'étude des langages résiduels.

Si L est un langage et u un mot, le langage résiduel $u^{-1}L$ est l'ensemble des mots w tels que uw appartient à L. Cette notion de langages résiduels est utilisée dans le cadre de l'inférence grammaticale mais sans être mise en évidence. Elle correspond à la question : « *le début u d'un mot étant donné, quels sont les fins de mots qui peuvent suivre ce u ?* ». En particulier, trouver les fins de mots qui peuvent suivre le mot vide ε, c'est à dire trouver le résiduel $\varepsilon^{-1}L$, résout l'identification du langage. La recherche des résiduels est d'autant plus pertinente pour la classe des langages rationnels que cette classe est la seule pour laquelle l'ensemble des résiduels est fini.

Pour les langages rationnels, la notion de résiduel est étroitement liée au langage reconnu à partir d'un état. Si un mot u atteint un état, le langage reconnu à partir de cet état est un sous-ensemble du résiduel $u^{-1}L$. Dans les automates déterministes, puisque chaque mot u atteint au plus un état, le langage reconnu à partir de cet état est exactement le résiduel $u^{-1}L$. Et passer d'un état à un autre par une lettre x équivaut à passer d'un résiduel $u^{-1}L$ au résiduel $(ux)^{-1}L$.

Le problème de l'identification des langages a souvent été abordé en utilisant le déterminisme. Dans ce cas, chaque mot u atteint une unique configuration (état, terminal ou non, etc.) et cette configuration définit l'ensemble de mots qui peuvent suivre u, c'est à dire le résiduel. Le déterminisme possède de très bonnes propriétés ; beaucoup de problèmes sont décidables grâce à ce déterminisme. Pour les langages ration-

nels, il existe un unique automate déterministe minimal reconnaissant chaque langage. Dans cette étude, nous privilégions la notion de résiduels et nous considérons que le déterminisme est une propriété permettant d'obtenir les résiduels.

Dans son article sur les k-réversibles, Angluin a donné une caractérisation des langages k-réversibles en termes de langages résiduels. Il s'agit donc pour nous de partir de cette définition pour définir une classe plus générale, celle des *langages rationnels à résiduels k-disjoints* avec le double objectif d'être capable d'identifier cette classe et d'atteindre les rationnels par clôture.

Après avoir présenté les notions nécessaires et avoir rappelé le cadre d'apprentissage, nous définissons les langages rationnels à résiduels k-disjoints. Nous vérifions que la hiérarchie des familles de langages rationnels k-disjoints couvre la totalité des langages rationnels. Nous montrons que chaque famille de langages rationnels à résiduels k-disjoints est identifiable à la limite, en temps et données polynomiaux, à partir d'exemples positifs. Nous présentons succinctement une restriction des langages rationnels à résiduels k-disjoints pour laquelle l'équivalence de deux résiduels est plus simple à tester.

2 Notations, définitions et rappels

2.1 Définitions classiques

Dans cette section, nous rappelons quelques définitions et notations concernant les automates finis. Pour plus de détails, nous invitons le lecteur à se reporter à (Hopcroft & Ullman, 1979; Yu, 1997).

Soit Σ un alphabet fini. L'ensemble Σ^* représente l'ensemble de tous les mots sur Σ. Pour un entier n donné, Σ^n représente l'ensemble des mots de longueur n et $\Sigma^{\leq n}$ l'ensemble de ceux de longueur inférieure ou égale à n. Nous notons ε pour le mot vide. La longueur d'un mot $u \in \Sigma^*$ est notée $|u|$. Un *langage* est un sous-ensemble de Σ^*. La cardinalité d'un ensemble E est le nombre de ses éléments ; elle est notée $\|E\|$. Nous utiliserons aussi une autre mesure de la taille d'un ensemble : $|[E]|$ représentera la somme des longueurs de ses éléments.

Pour énumérer les mots, nous utiliserons un ordre strict, noté $<$, sur l'alphabet Σ. L'ordre militaire sur Σ^* est noté $\prec_{ab}$ et il est défini par : $\forall u, v \in \Sigma^*$, nous avons $u \prec_{ab} v$ si et seulement si $|u| < |v|$ ou il existe des mots $w, u', v' \in \Sigma^*$ et deux lettres $x < y \in \Sigma$ tels que $|u'| = |v'|$ et $u = wxu'$, $v = wyv'$.

2.2 Langages reconnaissables

Un *automate fini non déterministe* (AFN) est un quintuplet $A = \langle \Sigma, Q, Q_I, Q_F, \delta \rangle$ où Q est un ensemble fini d'états, $Q_I \subseteq Q$ est l'ensemble des états initiaux, $Q_F \subseteq Q$ est l'ensemble des états finals, δ est la *fonction de transition*. La fonction de transition est un sous-ensemble $Q \times \Sigma \times Q$. De façon équivalente, on peut la voir comme une fonction d'un sous-ensemble de $Q \times \Sigma$ dans 2^Q.

Nous noterons δ_* la fonction étendue, définie d'un sous-ensemble de $2^Q \times \Sigma^*$ dans 2^Q, par :

$\delta_*(\{q\}, \varepsilon) = \{q\}$, où $q \in Q$,
$\delta_*(\{q\}, x) = \delta(q, x)$, où $x \in \Sigma$, $q \in Q$,
$\delta_*(\{q\}, xu) = \delta_*(\delta(q, x), u)$ où $x \in \Sigma$, $q \in Q$ et $u \in \Sigma^*$,
$\delta_*(Q', u) = \cup_{q \in Q'} \delta_*(\{q\}, u)$, où $Q' \subseteq Q$ et $u \in \Sigma^*$.

Un mot $u \in \Sigma^*$ est reconnu par un AFN $A = \langle \Sigma, Q, Q_I, Q_F, \delta \rangle$ si $\delta_*(Q_I, u) \cap Q_F \neq \emptyset$ et le langage reconnu par A est $L_A = \{u \mid \delta_*(Q_I, u) \cap Q_F \neq \emptyset\}$. Nous notons $Rec(\Sigma^*)$ la famille des langages reconnaissables, qui est aussi la famille des langages rationnels.

Un AFN est *émondé* si et seulement si $\forall q \in Q$, $\exists w_1 \in \Sigma^*$, $q \in \delta_*(Q_I, w_1)$ et $\exists w_2 \in \Sigma^*$, $\delta_*(q, w_2) \cap Q_F \neq \emptyset$. Un état q est *accessible* par le mot u si $q \in \delta_*(Q_I, u)$.

Un AFN est *déterministe* (AFD) si Q_I contient un seul élément q_0 et si $\forall q \in Q$, $\forall x \in \Sigma$, $\|\delta(q, x)\| \leq 1$. Tout langage reconnaissable peut être reconnu par un AFD. Il existe un unique AFD minimal qui reconnaît un langage donné ; minimal signifiant *qui a le nombre minimum d'états* et l'unicité se faisant à un isomorphisme près.

Soit $A = \langle \Sigma, Q, Q_I, Q_F, \delta \rangle$ un AFN et soit $q \in Q$. Nous notons D_q l'ensemble des mots qui sont reconnus à partir de l'état q, c'est à dire $D_q = \{v \mid \delta_*(q, v) \cap Q_F \neq \emptyset\}$. Nous étendons ces définitions aux ensembles d'états $Q' \subseteq Q$, par $D_{Q'} = \{v \mid \exists q \in Q' \text{ tel que } \delta_*(q, v) \cap Q_F \neq \emptyset\}$.

Pour lever des ambiguïtés, nous écrirons parfois $D_{A,q}$ pour préciser l'automate.

2.3 Résiduels d'un langage

Soit L un langage sur Σ^* et soient $u, v \in \Sigma^*$. Le *résiduel* de L par rapport à u est défini par $u^{-1}L = \{v \in \Sigma^* \mid uv \in L\}$. Le langage D est un *résiduel du langage* L s'il existe un mot $u \in \Sigma^*$ tel que $D = u^{-1}L$.

Nous dirons que u est un *préfixe caractéristique* du résiduel $D = u^{-1}L$. Le mot u est un *préfixe représentatif* du résiduel $D = u^{-1}L$ si c'est le plus petit préfixe caractéristique de D en respectant l'ordre $\prec_{\alpha b}$ sur Σ^*.

Le théorème de Myhill-Nerode (Myhill, 1957; Nerode, 1958) montre que l'ensemble des résiduels distincts d'un langage L est fini si et seulement si L est reconnaissable par un AFN.

Si A est l'AFD minimal reconnaissant un langage L, la correspondance entre états de A et résiduels non vides de L est bijective. Il y a aussi une bijection entre l'ensemble des résiduels et l'ensemble des préfixes représentatifs. L'ensemble des préfixes représentatifs constitue un arbre : si $\alpha_1\alpha_2$ est un préfixe représentatif, alors α_1 est aussi un préfixe représentatif.

3 Langages rationnels avec résiduels k-disjoints

3.1 Définitions

Angluin (1982) a défini les langages k-réversibles par une propriété de l'automate déterministe minimal qui reconnaît le langage (le miroir de l'automate déterministe minimal est déterministe avec anticipation de k lettres). Ceci est la définition usuelle de la famille des langages k-réversibles, notée $Rev_k(\Sigma^*)$. Dans (Angluin, 1982), le théorème 14 caractérise les langages k-réversibles en termes de résiduels.

Définition 1
Un langage rationnel L possède des résiduels k-anticipés disjoints si et seulement si, pour tout α, $\alpha' \in \Sigma^*$, pour tout $u \in \Sigma^k$, nous avons

$$(\alpha u)^{-1}L \cap (\alpha' u)^{-1}L \neq \emptyset \quad \Longrightarrow \quad (\alpha u)^{-1}L = (\alpha' u)^{-1}L$$

Le théorème 14 de (Angluin, 1982) peut alors être réécrit de la façon suivante.

Théorème 1 (Angluin, 1982)
Un langage rationnel L est k-réversible si et seulement si L possède des résiduels k-anticipés disjoints.

Soit L un langage rationnel possédant des résiduels k-anticipés disjoints. Il est facile de vérifier que

$$\begin{gathered} \alpha^{-1}L = {\alpha'}^{-1}L \\ \Updownarrow \\ \alpha^{-1}L \cap \Sigma^{<k} = {\alpha'}^{-1}L \cap \Sigma^{<k} \\ \text{et} \\ \forall u \in \Sigma^k, \left((\alpha u)^{-1}L = \emptyset \text{ et } (\alpha' u)^{-1}L = \emptyset\right) \text{ ou } (\alpha u)^{-1}L \cap (\alpha' u)^{-1}L \neq \emptyset \end{gathered} \tag{1}$$

En effet, il est clair que l'égalité $\alpha^{-1}L = {\alpha'}^{-1}L$ implique les conditions (1).

Supposons que les conditions (1) soient vérifiées mais que nous n'ayons pas $\alpha^{-1}L = {\alpha'}^{-1}L$. Alors il existe un mot $w \in \alpha^{-1}L \setminus {\alpha'}^{-1}L$ (ou $w \in {\alpha'}^{-1}L \setminus \alpha^{-1}L$). Ce mot w est plus grand que k, sinon l'égalité $\alpha^{-1}L \cap \Sigma^{<k} = {\alpha'}^{-1}L \cap \Sigma^{<k}$ n'est pas vérifiée Alors le mot w est égal à uw_1 avec $u \in \Sigma^k$. Dans ce cas l'égalité $(\alpha u)^{-1}L = \emptyset$ n'est pas vérifiée ; donc nous devons avoir $(\alpha u)^{-1}L \cap (\alpha' u)^{-1}L \neq \emptyset$. Puisque le langage possède des résiduels k-anticipés disjoints, cela signifie que $(\alpha u)^{-1}L = (\alpha' u)^{-1}L$. Donc le mot w_1 appartient à $(\alpha u)^{-1}L = (\alpha' u)^{-1}L$ et $uw_1 = w$ appartient à ${\alpha'}^{-1}L$ ce qui contredit l'hypothèse.

Nous définissons alors les langages rationnels k-disjoints par cette propriété, ou plutôt par sa contraposée plus lisible.

Définition 2
Un langage rationnel L possède des résiduels k-disjoints si et seulement si, pour tout α, $\alpha' \in \Sigma^*$, nous avons

$$\alpha^{-1}L \neq \alpha'^{-1}L$$
$$\Updownarrow$$
$$\alpha^{-1}L \cap \Sigma^{<k} \neq \alpha'^{-1}L \cap \Sigma^{<k}$$
ou
$$\exists u \in \Sigma^k, \big((\alpha u)^{-1}L \neq \emptyset \text{ ou } (\alpha' u)^{-1}L \neq \emptyset\big) \text{ et } (\alpha u)^{-1}L \cap (\alpha' u)^{-1}L = \emptyset$$

Nous dirons simplement que le langage est *résiduel k-disjoint* plutôt que rationnel possédant des résiduels k-disjoints.

Nous noterons $LRD_k(\Sigma^*)$ la famille des langages résiduels k-disjoints sur un alphabet Σ. La famille $LRD_*(\Sigma^*)$ est l'union de toutes les familles $LRD_k(\Sigma^*)$, pour $k \geq 0$.

D'après les définitions, nous pouvons vérifier les inclusions suivantes.

Propriété 1
Nous avons $Rev_0(\Sigma^*)$ est égale à $LRD_0(\Sigma^*)$.
Pour tout $k > 0$, nous avons $Rev_k(\Sigma^*)$ est strictement incluse dans $LRD_k(\Sigma^*)$.
Pour tout $k \geq 0$, nous avons $LRD_k(\Sigma^*)$ est strictement incluse dans $LRD_{k+1}(\Sigma^*)$.

Preuve : Nous avons vu que, si L est un langage k-réversible, alors L est résiduel k-disjoint. Donc, pour tout $k \geq 0$, nous avons $Rev_k(\Sigma^*)$ est inclus dans $LRD_k(\Sigma^*)$.

Si $k = 0$ dans les conditions (1), nous avons $\alpha^{-1}L = \alpha'^{-1}L$ si et seulement si $\big(\alpha^{-1}L = \alpha'^{-1}L = \emptyset$ ou $\alpha^{-1}L \cap \alpha'^{-1}L \neq \emptyset\big)$. Ainsi $\alpha^{-1}L \cap \alpha'^{-1}L \neq \emptyset \Longrightarrow \alpha^{-1}L = \alpha'^{-1}L$ ce qui est la définition de résiduel 0-anticipé disjoint.

Comme nous le verrons dans la section 3.2, le langage $ab^{k-1}b^* + db^{k-1}(b+c)^*$ est résiduel k-disjoint mais n'est pas résiduel k-anticipé disjoint. Le langage $ab^kb^* + db^k(b+c)^*$ est résiduel $(k+1)$-disjoint mais n'est pas résiduel k-disjoint. □

Il est facile de vérifier que chaque langage reconnaissable L appartient à une famille de langages résiduels k-disjoints, pour un certain k.

Propriété 2
$Rec(\Sigma^*)$ est égal à $LRD_*(\Sigma^*)$.

Preuve : Soit L un langage reconnaissable. Il a un ensemble fini de résiduels.

Pour chaque paire α, α' de résiduels distincts, il existe une longueur $k_{\alpha,\alpha'}$ telle que $\alpha^{-1}L \cap \Sigma^{<k_{\alpha,\alpha'}} \neq \alpha'^{-1}L \cap \Sigma^{<k_{\alpha,\alpha'}}$. Soit k_L la plus grande valeur des $k_{\alpha,\alpha'}$. Pour chaque paire α, α' de résiduels distincts, nous avons $\alpha^{-1}L \cap \Sigma^{<k_L} \neq \alpha'^{-1}L \cap \Sigma^{<k_L}$. Donc le langage L est résiduel k_L-disjoint. □

Propriété 3
Décider si un automate déterministe minimal reconnaît un langage résiduel k-disjoint s'effectue en temps $\mathcal{O}(\|A\|^4)$.

Preuve : Soit A_{min} un automate déterministe minimal qui reconnaît un langage L. Pour décider si le langage est résiduel k-disjoint, il faut, pour chaque couple d'états distincts $(\alpha^{-1}L, \alpha'^{-1}L)$, vérifier que l'une des deux propriétés suivantes est vérifiée.

Premièrement, les langages reconnus à partir de ces états peuvent être distincts pour les mots de longueur inférieure à k ; c'est à dire $\alpha^{-1}L \cap \Sigma^{<k} \neq \alpha'^{-1}L \cap \Sigma^{<k}$. Cette vérification s'effectue en temps constant... bien que la constante soit importante puisqu'elle est d'ordre $\|\Sigma^{<k}\|$.

Deuxièmement, il faut vérifier s'il existe un $u \in \Sigma^k$, tel que $\big((\alpha u)^{-1}L \neq \emptyset$ ou $(\alpha' u)^{-1}L \neq \emptyset\big)$ et $(\alpha u)^{-1}L \cap (\alpha' u)^{-1}L = \emptyset$. Au pire, cette vérification est faite en réalisant $\|\Sigma^k\|$ intersections et en vérifiant si ces intersections sont vides. Cette vérification s'effectue donc en temps $\mathcal{O}(\|A_{min}\|^4)$. Là encore, la puissance a un facteur constant important de l'ordre de $\|\Sigma^k\|$. □

Nous pouvons aussi essayer de distinguer les résiduels en examinant seulement les débuts de mots du résiduel. Dans la définition 2, seule l'égalité des ensembles $\alpha^{-1}L \cap \Sigma^{<k}$ serait requise pour avoir l'égalité de deux résiduels. Cela définirait une autre hiérarchie de langages, que nous appelons langages résiduels k-disjoints stricts. Cette restriction a pour conséquence que chaque famille de langages résiduels k-disjoints stricts est finie. Mais cette autre hiérarchie couvre encore les langages rationnels.

3.2 Exemples

Quand $k \geq 1$, les différences entre langages k-réversibles et langages résiduels k-disjoints apparaissent. Par exemple, le langage $L_1 = ab^* + d(b+c)^*$ n'est pas 1-réversible ; cependant il est résiduel 1-disjoint.

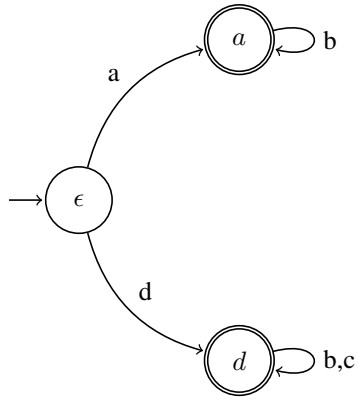

FIG. 1 – L'AFD minimal reconnaissant $L_1 = ab^* + d(b+c)^*$.

Le langage L_1 n'est pas 1-réversible parce que

$$(ab)^{-1}L \cap (db)^{-1}L \neq \emptyset \qquad \text{mais} \qquad (ab)^{-1}L \neq (db)^{-1}L$$

Remarquons que le langage L_1 n'est k-réversible pour aucun $k \in \mathbb{N}$.

Le langage L_1 est résiduel 1-disjoint parce que
- pour $\varepsilon^{-1}L$ et $a^{-1}L$, nous avons $\varepsilon^{-1}L \cap \Sigma^{<1} \neq a^{-1}L \cap \Sigma^{<1}$;
- pour $\varepsilon^{-1}L$ et $d^{-1}L$, nous avons $\varepsilon^{-1}L \cap \Sigma^{<1} \neq d^{-1}L \cap \Sigma^{<1}$;
- pour $a^{-1}L$ et $d^{-1}L$, nous avons $a^{-1}L \cap \Sigma^{<1} = \{\varepsilon\} = d^{-1}L \cap \Sigma^{<1}$ mais aussi $(dc)^{-1}L \neq \emptyset$ et $(ac)^{-1}L \cap (dc)^{-1}L = \emptyset$.

On peut généraliser cet exemple avec les langages $L_{k+1} = ab^kb^* + db^k(b+c)^*$, $k \geq 0$.

Pour chaque $k \geq 0$, le langage L_{k+1} n'est pas résiduel $(k+1)$-anticipé disjoint ; donc n'est pas $(k+1)$-réversible. Le langage L_{k+1} est résiduel $(k+1)$-disjoint. Le langage L_{k+1} n'est pas résiduel k-disjoint.

Le langage L_1 est aussi résiduel 1-disjoint strict.

4 Identification des langages résiduels k-disjoints

4.1 Présentation

Après le contexte donné en introduction, donnons la définition précise du modèle d'identification dans lequel nous nous plaçons.

Définition 3 (de la Higuera, 1997)
La famille de langages $\mathcal{L}$ est identifiable à la limite en temps et données polynômiales *à partir d'exemples positifs, en utilisant la classe de représentations $\mathcal{H}_\mathcal{L}$ si et seulement s'il existe un algorithme ϕ et deux polynômes $p()$ et $q()$ tels que :*

1. *pour tout échantillon S d'exemples positifs, $\phi(S)$ fournit une représentation H appartenant à $\mathcal{H}_\mathcal{L}$ telle que $S \subseteq L_H$ et le temps d'exécution de ϕ sur S est en $\mathcal{O}(p(|[S]|))$;*
2. *pour tout $L \in \mathcal{L}$, il existe un échantillon S_L (dit échantillon caractéristique) tel que $|[S_L]|$ est en $\mathcal{O}(q(\|H_{min,L}\|))$, où $\|H_{min,L}\|$ est la taille de la représentation minimale de L dans $\mathcal{H}_\mathcal{L}$, et si $S_L \subseteq S$, alors $L_{\phi(S)} = L$.*

4.2 Échantillon caractéristique

Puisque l'ensemble des mots représentatifs est un arbre, nous allons examiner les préfixes de l'échantillon et tester si les conditions de l'égalité de deux résiduels sont satisfaites. Il faut donc que l'échantillon contienne les informations pour décider de telles égalités.

Soit L un langage résiduel k-disjoint.

Soit $\mathcal{R}_c = \{\alpha \mid \alpha \textit{ est le préfixe représentatif de } \alpha^{-1}L \neq \emptyset\}$ l'ensemble des préfixes représentatifs des résiduels.

Soit $\mathcal{P}_c = \mathcal{R}_c \cup \{\alpha x \mid \alpha \in \mathcal{R}_c,\ x \in \Sigma\}$ l'ensemble des préfixes représentatifs, éventuellement suivis d'une lettre.

Nous définissons l'échantillon caractéristique S_{Lc} avec les caractéristiques suivantes :

- Pour chaque $\alpha \in \mathcal{P}_c$, l'ensemble $\alpha(\alpha^{-1}L \cap \Sigma^{<k})$ est inclus dans S_{Lc}.
- Pour chaque $\alpha \in \mathcal{P}_c$, pour chaque $u \in \Sigma^k$ si $(\alpha u)^{-1}L \neq \emptyset$, alors $\alpha u \gamma$ appartient à S_{Lc}, où γ est le plus petit mot de $(\alpha u)^{-1}L$.
- Pour chaque α et $\alpha' \in \mathcal{P}_c$, pour chaque $u \in \Sigma^k$ si $(\alpha u)^{-1}L \cap (\alpha' u)^{-1}L \neq \emptyset$, alors $\alpha u \gamma$ appartient à S_{Lc}, où γ est le plus petit mot de $(\alpha u)^{-1}L \cap (\alpha' u)^{-1}L$.

La valeur $\|\mathcal{R}_c\|$ est aussi le nombre d'états de A_{Ldet}, l'automate déterministe minimal reconnaissant L. La taille de S est la somme des longueurs de ses mots, notée $|[S]|$. La longueur des mots de l'échantillon caractéristique est bornée par $2\|A_{Ldet}\| + k$. Le nombre de mots de l'échantillon caractéristique est borné par $3\|\Sigma\|^k \ \|A_{Ldet}\|^2$. Pour chaque famille de langages résiduels k-disjoints, pour chaque alphabet Σ, la valeur $\|\Sigma\|^k$ est une valeur constante. Alors la taille de l'échantillon caractéristique est d'ordre $\mathcal{O}(\|A_{Ldet}\|^3)$.

4.3 Algorithme par élagage

Soit S un échantillon.

Soient α et α' deux préfixes de l'échantillon. Nous devons comparer ces deux préfixes de façon à évaluer leurs langages résiduels.

Nous définissons l'*équivalence selon* S par : $\alpha \simeq_S \alpha'$ si et seulement si

$$\alpha^{-1}S \cap \Sigma^{<k} = \alpha'^{-1}S \cap \Sigma^{<k}$$
$$\text{et}$$
$$\forall u \in \Sigma^k, \left((\alpha u)^{-1}S = \emptyset \text{ et } (\alpha' u)^{-1}S = \emptyset\right) \text{ ou } (\alpha u)^{-1}S \cap (\alpha' u)^{-1}S \neq \emptyset$$

L'algorithme 1 va construire un automate déterministe.

Théorème 2
La famille des langages résiduels k-disjoints peut être identifiée à la limite, en temps et données polynômiaux à partir d'exemples positifs, en les représentant par des AFD. L'algorithme travaille en temps $\mathcal{O}(|[S]|^4)$. L'échantillon caractéristique a une taille en $\mathcal{O}(\|A_{Ldet}\|^3)$.

Preuve : Quand l'échantillon S contient l'échantillon caractéristique S_{Lc}, le test d'équivalence fournit le bon résultat pour tout couple de préfixes appartenant à $\mathcal{P}_c$. Puisque les préfixes sont étudiés dans l'ordre militaire, l'algorithme évalue correctement les préfixes représentatifs et les transitions suivant ces préfixes représentatifs. Ainsi l'algorithme construit l'automate déterministe minimal reconnaissant le langage L.

Nous avons vu que la taille de l'échantillon caractéristique est d'ordre $\mathcal{O}(\|A_{Ldet}\|^3)$.

Le test d'équivalence nécessite la comparaison de deux ensembles finis de taille $\leq \|\Sigma\|^k$ et l'évaluation d'au plus $\|\Sigma\|^k$ intersections de langages résiduels (ou une intersection et un test de $\|\Sigma\|^k$ sous-automates). Donc la complexité en temps du test d'équivalence est d'ordre $\mathcal{O}(|[S]|^2)$. Comme le test d'équivalence doit être fait pour chaque couple de préfixes, le complexité en temps de l'algorithme est d'ordre $\mathcal{O}(|[S]|^4)$.

Le test final pour savoir si A est minimal k-disjoint et consistant est aussi en $\mathcal{O}(|[S]|^4)$ puisqu'il est réalisé sur un automate déterministe de taille inférieure à $|[S]|$. □

Algorithm 1 Algorithme par élagage

```
% initialisation
Pref = {α ∈ Σ* | γ ∈ Σ*, αγ ∈ S}
Q = {q_ε}
q_0 = q_ε
si ε ∈ S alors Q_F = {q_ε} sinon Q_F = ∅
δ = ∅
supprimer ε de Pref

% construction de l'automate
A = ⟨Σ, Q, Q_I, Q_F, δ⟩
tant que A n'est pas consistant et Pref est non vide
  αx = premier mot (dans l'ordre militaire) de Pref
  s'il existe q_α' ∈ Q tel que αx ≃_S α'
    alors
      ajouter (α, x, α') à δ
      supprimer αxΣ* de Pref
    sinon
      ajouter q_αx à Q
      si αx ∈ S alors ajouter q_αx à Q_F
      ajouter (α, x, αx) à δ
      supprimer αx de Pref
  fin si
fin tant que

% appartenance à la classe et consistance
si A n'est pas k-disjoint minimal
      ou si A n'est pas consistant
  alors A = A_Σ*
fin si
```

L'algorithme par élagage permet de montrer que la famille est identifiable mais il n'est réellement efficace que pour un échantillon contenant l'échantillon caractéristique. Un algorithme par fusions d'états, sans suppression de transitions, restera consistant.

4.4 Exemple

Nous prenons $L_1 = ab^* + d(b+c)^*$ pour exemple, que l'on étudie en tant que langage 1-disjoint. Son échantillon caractéristique est $S_{L_1 c} = \{a, d, ab, db, dc, abb, dbb, dbc, dcb, dcc\}$.

Sur cet échantillon, l'algorithme ajoutera les états q_ε, puis q_a, puis q_d car les préfixes ε, a, d ne sont pas équivalents. Lorsque l'algorithme traite le préfixe ab, celui-ci est équivalent à a. L'algorithme va donc ajouter la boucle (q_a, b, q_a) et supprimer tous les mots abw de S. Le mot suivant sera db qui est équivalent à d et l'algorithme se poursuivra jusqu'à atteindre la cible.

5 Conclusion

Nous avons proposé dans cet article la hiérarchie des *langages résiduels k-disjoints*. Cette hiérarchie couvre la famille des langages rationnels. Tout langage rationnel peut donc constituer une cible pour une identification exacte, par présentation de positifs seuls. Cette propriété a motivé notre travail et motivera la poursuite de l'étude de ces langages.

Signalons la hiérarchie de langages k-ORE (*k-occurence regular expressions*), définie par Bex *et al.* (2008), qui offre des résultats comparables : cette hiérarchie couvre elle aussi totalement la famille des langages rationnels. Mais pour cela les auteurs ont limité les occurrences de chaque lettre dans l'expression régulière cible, ce qui revient à limiter la taille de l'automate qui reconnaît le langage. Autrement dit, chaque famille de langages k-ORE est une famille finie. Les auteurs proposent un algorithme d'identification qui semble efficace mais, comme nous, ils sont confrontés aux mêmes problèmes de taille d'échantillon.

Nos algorithmes, même s'ils sont théoriquement polynomiaux, font appel à des ensembles finis qui peuvent être de très grande taille. Il faut aussi remarquer que, même si $LRD_k(\Sigma^*)$ est incluse dans $LRD_{k+1}(\Sigma^*)$, l'identification exacte d'un langage k-disjoint en tant que $k+1$-disjoint nécessite un échantillon plus grand.

Ces défauts peuvent être abordés de diverses façons. Nous avons évoqué les familles de langages résiduels k-disjoints stricts pour les pistes qu'elles amorcent.

On peut chercher à réduire la taille de l'échantillon caractéristique. Dans la définition de l'échantillon caractéristique, seuls les deux premiers items sont à retenir pour les k-disjoints stricts. Cela réduit l'échantillon mais cela ne change pas encore l'ordre de grandeur de l'échantillon caractéristique.

Ces familles présentent aussi l'avantage d'un test d'équivalence plus simple à évaluer. Si l'on travaille avec des automates émondés, cela revient à regarder les premières transitions sortant de chaque état. D'un point de vue formel, le test s'effectue en temps

constant pour chaque famille, mais avec une constante de l'ordre de $\|\Sigma^k\|$. Le test peut être réalisé sur un automate non déterministe. On sait que les automates finis à états résiduels, qui sont des automates non déterministes, peuvent être considérablement plus petits que les automates déterministes. Or la taille de l'échantillon caractéristique dépend de la taille de l'automate utilisé comme représentant du langage dans la famille.

Enfin, on peut aussi chercher une famille de langages pour laquelle l'absence d'un exemple serait moins pénalisant. Dans un automate reconnaissant un langage résiduel k-disjoint strict, on peut associer une sémantique aux états ; à partir de chaque état, on reconnaît des mots qui commencent par un certain ensemble de préfixes. L'algorithme d'identification revient alors à considérer comme équivalents des résiduels qui commencent par les mêmes préfixes. Le fait de pouvoir associer une sémantique aux résiduels permettra de faciliter la recherche d'approximations de ces résiduels.

Nous orienterons donc nos recherches vers les familles de langages k-disjoints stricts.

Références

ANGLUIN Dana. (1982). Inference of Reversible Languages. *JACM*, volume 29, number 3, pages 741–765.

BEX Geert Jan, GELADE Wouter, NEVEN Franck and VANSUMMEREN Stijn. (2008) Learning deterministic regular expressions pour the inference of schemas from XML data. *WWW '08 : Proceeding of the 17th international conference on World Wide Web*, pages 825–834.

GARCÍA Pedro and VIDAL Enrique. (1990). Inference of k-Testable Languages in the Strict Sense and Application to Syntactic Pattern Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, volume 12, number 9, pages 920–925.

GOLD E. Mark. (1967). Language identification in the limit. *Information and Control*, volume 10, number 5, pages 447–474.

DE LA HIGUERA Colin. (1997). Characteristic sets pour polynomial grammatical inference. *Machine Learning*, volume 27, pages 125–137.

HOPCROFT John E. and ULLMAN Jeffrey D. (1979). *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley.

MYHILL John. (1957). Finite automata and the representation of events. Technical Report 57-624, WADC.

NERODE Anil. (1958). Linear automaton transformation. In *Proc. American Mathematical Society*, volume 9, pages 541–544.

ONCINA José and GARCÍA Pedro. (1992). Inferring regular languages in polynomial update time. In *Pattern Recognition and Image Analysis*, pages 49–61. World Scientific.

PITT Leonard. (1989). Inductive Inference, DFAs, and Computational Complexity. In *Proceedings of International Workshop on Analogical and Inductive Inference*, volume 397 of *Lecture Notes in Computer Science*, pages 18–44. Springer-Verlag.

YU Sheng. (1997). *Handbook of Formal Languages, Regular Languages*, volume 1, chapter 2, pages 41–110. Springer Verlag.