APPRENTISSAGE D'AUTOMATES PROBABILISTES DÉTERMINISTES

Franck Thollard* Alexander Clark†

* EURISE, Université de Saint-Etienne, France
thollard@univ-st-etienne.fr

† ISSCO/TIM, Université de Genève, Suisse
alex.clark@issco.unige.ch

Résumé

Nous présentons dans ce papier une preuve constructive de l'identification à la limite avec probabilité 1 de la classe des automates probabilistes déterministes (PDFA) à coefficients rationnels. L'algorithme utilisé est polynomial en le nombre de chaînes de l'ensemble d'apprentissage. Nous montrons par ailleurs que sans information supplémentaire, la classe des PDFA ne peut pas être apprenable dans le cadre KL-PPAC, cadre PAC adapté au cas probabiliste qui utilise la divergence de Kullback-Leibler. Nous discutons également l'utilisation d'autres distances.

Mots-clés : Automates probabilistes, Apprentissage PAC, Identification à la limite.

1 INTRODUCTION

Les automates finis déterministes probabilistes (PDFA) sont largement utilisés dans un nombre important de domaines de l'informatique et particulièrement en traitement automatique de la langue naturelle. Il y a donc un besoin pratique de fournir des algorithmes permettant l'apprentissage de ces modèles à partir d'ensembles d'apprentissage. Ces algorithmes se doivent de plus d'être efficaces tant en termes de quantité de données nécessaires à leur convergence que dans leurs temps de réponse.

Du point de vue théorique, il est intéressant de trouver des algorithmes dont on peut prouver la convergence. Cela a été fait pour certaines sous-classes des PDFA (Ron *et al.*, 1995).

Il existe plusieurs paradigmes d'apprentissage qui peuvent être utilisés pour ce problème. Dans ce papier, nous utiliserons le concept d'identification à la limite avec probabilité 1 où les algorithmes doivent trouver exactement la réponse après avoir perpétré seulement un nombre fini d'erreur. Nous nous intéresserons aussi au cadre "Probablement Approximativement Correct" (PAC) introduit par Valiant (1984), et étendu par Ron et *al.* (1995). Dans la suite, nous nous intéresserons uniquement au cas où on ne dispose que d'exemples positifs distribués selon une certaines densité de probabilité que l'algorithme doit retrouver (exactement ou approximativement). En effet, dans beaucoup d'applications, par exemple dans le domaine du traitement automatique de la langue naturelle, les exemples négatifs sont particulièrement difficiles à obtenir et, s'ils existent, sont souvent artificiels.

Nous donnons ici un algorithme qui identifie à la limite avec probabilité 1 la classe des PDFA à coefficients rationnels. La restriction aux coefficients rationnels, ou à d'autres sous-ensembles énumérables des réels est évidemment nécessaire et incontournable. Nous prouvons par la ensuite qu'il n'est pas possible d'approximer (dans un cadre dérivé du cadre PAC) la classe des PDFA sans information supplémentaire qui restreindrait d'une certaine manière la classe des automates cibles possibles.

La classe des PDFA à coefficients rationnels est clairement une classe énumérable de langages et peut donc, en tant que telle, être identifiée par des méthodes énumératives (Horning, 1972; Angluin, 1988; Kapur & Bilardi, 1992). Ces algorithmes sont cependant d'une complexité calculatoire trop importante et sont en pratiques impraticables sauf pour des problèmes triviaux.

L'algorithme que nous présentons ici est lié aux algorithmes de *fusion d'états*. Il procède en trois phases :

1. Construction d'une représentation des données sous la forme de l'arbre accepteur des préfixes probabilistes (PPTA)
2. Détermination de l'ensemble des classes d'équivalence des états qui doivent être fusionnés ou qui, en d'autres termes, sont des représentations distinctes du même état de l'automate cible. Deux états sont équivalents si leur fréquence est supérieure à un certain seuil d'une part et si les distributions qu'ils représentent sont suffisamment proches. Les états équivalents sont alors identifiés et fusionnés ce qui permet d'obtenir un automate isomorphe à la cible.
3. La technique proposée dans (de la Higuera & Thollard, 2000) est alors appliquée pour identifier les probabilités de l'automate.

Cet algorithme est polynomial en le nombre de symboles de l'ensemble d'apprentissage, mais nécessite, dans le pire des cas, un nombre exponentiel de données pour identifier l'automate cible.

Le reste du papier est organisé comme suit : la section 2 présente quelques définitions; les sections 3 et 4 décrivent l'algorithme et présentent la preuve de sa convergence. La section 5 définit le cadre KL-PPAC, adaptation du cadre PAC au cadre probabiliste, et montre que la classe des PDFA ne peut pas être appris selon ce critère. Nous concluons par une discussion en section 6.

1.1 Travaux connexes

Cette section discute des travaux traitant de l'apprentissage d'automates probabilistes à états finis. Le premier résultat viens de Horning (1972) qui montre que toute classe énumérable de langage peut être identifiée à la limite avec probabilité 1. Le problème de cette approche et de celles du même type (Angluin, 1988; Kapur & Bilardi, 1992) est qu'elles ne fournissent pas de preuve constructive.

Gillman et *al.* (1994) étudient pour leur part le problème de *l'inférence exacte* de modèles de Markov Cachés ergodiques définis sur un alphabet binaire. Leur cadre d'apprentissage permet à l'algorithme de lancer des requêtes à un *oracle probabiliste* pour obtenir la probabilité à long terme d'une chaîne binaire arbitraire. Ils prouvent que l'inférence est *difficile* : tout algorithme doit faire appel à l'oracle un nombre exponentiel de fois. Leur preuve provient de la théorie de l'information et ne dépend d'aucun pré-requis concernant la séparabilité de certaines classes d'hypothèses.

Plus récemment, des travaux ont été proposés dans le domaine des preuves constructives pour l'identification à la limite avec probabilité 1 de la structure de PDFA (Carrasco & Oncina, 1999). Des problèmes persistent (comme nous le verrons plus loin) et la première partie de notre papier se propose de les régler et de clarifier certains points. Par ailleurs, l'identification à la limite de variables aléatoires à valeurs rationnelles a été prouvée (de la Higuera & Thollard, 2000). La combinaison de ces deux résultats amène une preuve constructive de l'identification à la limite de la classe des PDFA à coefficients rationnels.

L'autre cadre d'apprentissage qui a été massivement étudié est le cadre PAC (PAC pour Probablement Approximativement Correct). Dans ce cadre, l'objectif est d'inférer une bonne solution (*i.e.* une solution proche de la cible) avec une grande probabilité. Dans ce cadre, les résultats sont quelques peu différents suivant les objets que l'on veut inférer ou ce que l'on en sait. Abe et Warmuth (1992) montrent que les automates non déterministes acycliques qui définissent une distribution de probabilités sur Σ^n, avec n et Σ connus, peuvent être appris en temps polynomial. Ils montrent par ailleurs que la polynomialité de l'apprentissage est perdue lorsque la taille de l'alphabet fait partie des données.

Kearns et *al.* (1994) montrent pour leur part, que le modèle ne peut être approximé si la distribution de probabilité peut être représentée par un automate fini probabiliste défini sur $\{0, 1\}$. Plus précisément, ils réduisent le problème de l'inférence à celui du problème des fonctions de parité : il existe une constante $0 < \eta < \frac{1}{2}$ telle que il n'existe pas d'algorithme efficace pour apprendre des fonctions de parité sous la distribution uniforme dans le modèle PAC avec un bruit de classification η.

En conséquence, connaître la classe des objets que l'on cherche à inférer permet de guider les algorithmes étant donné que les objets traités par Abe et Warmuth (1992) sont plus complexes que ceux proposés par Kearns et *al.*. Suivant ce principe, Ron *et al.* (1994; 1995) proposent l'apprentissage d'une classe particulière d'automates probabilistes. Ils proposent dans un premier temps un algorithme qui apprend des arbres de suffixes probabilistes et montrent leur convergence dans un modèle à la PAC. Ils prouvent ensuite le même genre de résultat avec des automates acycliques déterministes.

Nous montrons dans la seconde partie de cet article que les automates cycliques probabilistes ne peuvent pas être inférés sans information additionnelle concernant la cible, comme par exemple une borne sur l'espérance de la longueur des chaînes où sur l'entropie de la distribution de la cible.

2 DÉFINITIONS

2.1 Distributions de probabilités discrètes

Σ est un alphabet fini de symboles. $\Sigma^\star$ est le monoïde libre généré par Σ et λ est l'identité de $\Sigma^\star$, la chaîne vide. La longueur d'une chaîne s est notée $|s|$. Une distribution D sur $\Sigma^\star$ est une fonction de $\Sigma^\star \rightarrow [0,1]$ telle que $\sum_{\sigma \in \Sigma^\star} D(\sigma) = 1$.

2.2 Identification à la limite avec probabilité un

Nous traitons ici de l'identification à la limite avec probabilité un que nous noterons en abrégé ILP1. Nous considérons une séquence infinie S de chaînes tirées indépendamment conformément à une distribution inconnue D. Nous notons S_n l'ensemble constitué des n premiers éléments de S. Notre algorithme A produit une hypothèse H_n à partir de S_n.

Définition 1 (ILP1)
A identifie une classe de distributions $\mathcal{D}$ à la limite avec probabilité 1 (ILP1) si et seulement si pour tout $D \in \mathcal{D}$, $Pr(\exists n \, t.q. \forall k > n \, H_k(S_k) = D) = 1$

Nous utiliserons par la suite l'abréviation *presque toujours* en lieu et place de la phrase *avec probabilité un pour toutes sauf un nombre fini de valeurs de n.*

2.3 Métrique sur les distributions

Soit deux distributions D_1, D_2 définies sur $\Sigma^\star$, nous définissons

Définition 2 (distance d_*)

$d_*(D_1, D_2) = \max_{s \in \Sigma^*} |D_1(s) - D_2(s)|$

d_* est une métrique. Soit une séquence de mots S tirés indépendamment conformément à la distribution D. La distribution empirique des n premiers éléments de S (notée $\hat{D_n}$) sera proche de la distribution D et ce avec une grande probabilité :

Angluin (1988) montre que $Pr(d_*(D, \hat{D}_n) > \sqrt{\frac{6a \log n}{n}}) < 4n^{-a}, a > 1$. Nous adopterons par la suite la notation $\epsilon_n^a = \sqrt{\frac{6a \log n}{n}}, a > 1$

Le modèle que nous allons étudier ici diffère des modèles généralement utilisés dans la littérature en ce sens où il définit une distribution de probabilités sur $\Sigma^\star$. D'autres travaux (Kearns *et al.*, 1994; Abe & Warmuth, 1992; Ron *et al.*, 1995) traitent de distributions de probabilités définies sur un ensemble fini.

Nous donnons maintenant la définition formelle de notre modèle.

Définition 3 (PDFA)
Un Automate Fini Déterministe Probabiliste *(PDFA) est un 5-tuple* $(\Sigma, Q, q_I, \xi, \delta, \gamma, F)$ *où* Σ *est l'alphabet,* Q *est l'ensemble des états,* $q_I \in Q$ *est l'état* initial, $\xi \subset Q \times \Sigma \times Q \times (0,1]$ *est un ensemble de transitions probabilistes.* F*:* $Q \rightarrow [0,1]$ *est la probabilité que l'automate termine dans un état particulier. Les deux fonctions* δ *et* γ*, de* $Q \times \Sigma$ *dans* Q *et* $(0,1]$ *respectivement, sont définies comme :*

$$\begin{array}{rcl} \delta(q_i, \sigma) & = & q_j \text{ ssi } \exists\ p \in (0,1]\ : (q_i, \sigma, q_j, p) \in \xi \\ \gamma(q_i, \sigma) & = & p \text{ ssi } \exists\ q_j \in Q\ : (q_i, \sigma, q_j, p) \in \xi \end{array}$$

Ces fonctions s'étendent naturellement à $Q \times \Sigma^{\star}$.

Nous imposons que pour chaque état q, $\sum_{\sigma} \gamma(q, \sigma) + F^A(q) = 1$, que chaque état est accessible à partir de l'état initial avec une probabilité non nulle et peut générer au moins une chaîne de probabilité non nulle. Sous ces conditions, l'automate définit une distribution de probabilités sur $\Sigma^{\star}$.

$\gamma^A(q_I, x) \times F^A(\delta(q_I, x))$ représentera la probabilité de x selon l'automate A. Cette probabilité sera noté $Pr_A(x)$. Par ailleurs, pour $q_j \in Q$, $\Gamma(q_j)$, définit la distribution de probabilités obtenue en considérant q_j comme état initial.

Soit I_+ un *ensemble positif*, *i.e.* un ensemble de chaînes appartenant au langage probabiliste que l'on veut modéliser. Soit $PTA(I_+)$ l'*arbre accepteur des préfixes* construit à partir de I_+. L'*arbre accepteur des préfixes* est un automate qui accepte uniquement les chaînes de I_+ et dans lequel les préfixes communs ont été fusionnés, engendrant une structure d'arbre. Le $PPTA(I_+)$ est l'extension probabiliste du $PTA(I_+)$ dans laquelle chaque transition est associée à une probabilité dépendant du nombre de fois où elle a été utilisé lors de la génération (ou de manière équivalente l'analyse) de l'ensemble I_+

Notons $n(q)$ le compte (ou l'effectif) de l'état q, c'est à dire, le nombre de fois où l'état q a été utilisée pour générer I_+ à partir du $PPTA(I_+)$, et $n(q, \#)$ le nombre de fois où l'analyse d'une chaîne de I_+ finit en q. De même, $n(q, \sigma)$ représentera le compte de la transition $(q, \sigma, \bullet, \bullet)$ dans le $PPTA(I_+)$. Le $PPTA(I_+)$ est l'estimation du maximum de vraisemblance de la distribution empirique construite à partir de I_+. En particulier, pour le $PPTA(I_+)$ l'estimation de la probabilité d'une transition s'exprime comme : $\hat{\gamma}(q, \sigma) = \frac{n(q,\sigma)}{n(q)}$, $a \in \Sigma \cup \{\#\}$.

Un exemple de $PPTA$ est présenté en figure 1.

Définition 4 (d_{min})
Pour un PDFA, avec Q *comme ensemble d'états,* $d_{min} = \min_{q_i, q_j \in Q, i \neq j} d_*(\Gamma(q_i), \Gamma(q_j))$

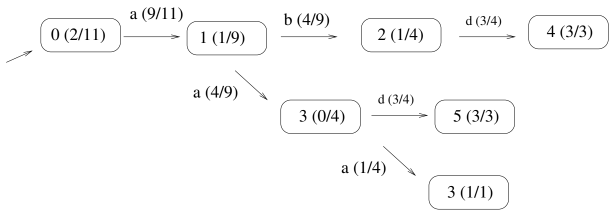

FIG. 1 – *PPTA construit avec $I+$ = {λ, aac, aac, abd, aac, aac, abd, abd, λ, a, ab }*

Si $d_{min} = 0$, l'automate peut être simplifié sans changer la distribution qu'il engendre en fusionnant l'ensemble des états tel que $d_*(\Gamma(q_i), \Gamma(q_j)) = 0$. Nous supposerons donc par la suite que $d_{min} > 0$.

Définition 5 (ϕ)
Étant donné un automate C, et un PPTA construit sur un ensemble de séquences générées à partir de C et q un état du PPTA, nous définissons $\phi(q)$ comme l'état de C qui correspond à l'état q. Plus formelement, $\phi(q_I^{PPTA}) = q_I^C$ et $\forall q, q' \in Q^{PPTA}$, $\forall \sigma \in \Sigma$ si $\delta^{PPTA}(q, a) = q'$ alors $\delta^C(\phi(q), a) = \phi(q')$.

Définition 6 (Etats équivalents)
Soit C un automate, et un PPTA construit sur un ensemble de séquences générées à partir de C, deux états du PPTA seront dit équivalents *si ils correspondent au même état dans C,* i.e. *q_i est équivalent à q_j ssi $\phi(q_i) = \phi(q_j)$.*

3 L'ALGORITHME D'INFÉRENCE

Nous pouvons maintenant décrire l'algorithme qui identifie à la limite avec probabilité 1 la classe des PDFA à coefficients rationels.

Algorithme 1: L'algorithme d'apprentissage

Data : S_n l'ensemble d'apprentissage et n sa taille, $a > 2$
Result : Un PDFA
A ← `Construire_PPTA` (S_n) ;
pour $(q_i, q_j) \in Q^A \times Q^A$ **faire**
  si $d_*(\Gamma(q_i), \Gamma(q_j)) < \epsilon^a_{n(q_i)} + \epsilon^a_{n(q_j)}, \quad n(q_i) > \sqrt{n}, n(q_j) > \sqrt{n}$ **alors**
    `A_Fusionner` ← (q_i, q_j) ;
pour $(q_i, q_j) \in$ *`A_Fusionner`* **faire**
  `Fusion` (A, q_i, q_j) ;
`Identifie_Proba` (A) ;

L'algorithme procède comme suit : pour chaque n nous avons S_n les n premiers éléments de l'ensemble S. Nous construisons le PPTA de S_n et comptons le nombre de chaînes de S_n qui traversent chaque état.

Nous déterminons ensuite l'ensemble des états qui doivent être fusionnés, états dont les distributions des suffixes sont similaires et dont les comptes sont suffisamment importants. Nous fusionnons ensuite ces états et déterminisons récursivement le résultat. Nous utilisons enfin l'algorithme d'identification des coefficients des rationnels (de la Higuera & Thollard, 2000). La différence principale entre cet algorithme et celui présenté par Carrasco & Oncina (1999) est l'introduction des contraintes sur les effectifs des états ($n(q_i) > \sqrt{n}$). Elles nous permettent de satisfaire les conditions d'utilisation des tests statistiques.

Nous définissons maintenant plus formellement l'algorithme.

Définition 7 (Etats similaires)
Deux états du PPTA q_i *et* q_j *sont* similaires *ssi* $d_*(\Gamma(q_i), \Gamma(q_j)) < \epsilon^a_{n(q_i)} + \epsilon^a_{n(q_j)}$

Définition 8 (Chemins simples et semi-simples)
Étant donné un PDFA, un chemin simple *(resp.* semi-simple*) est une séquence de transitions qui part de l'état initial et dans laquelle chaque transition apparaît au plus une fois (resp. au plus de deux fois).*

Définition 9 (Etats pleins, PPTA plein)
Un état q_i *d'un PPTA est* plein *si* $n(q_i) > f(n)$ *où* $f(n)$ *est une fonction de* n *sub-linéaire non-bornée. Le PPTA est* plein *si pour chaque chemin semi-simple de la cible, il existe un chemin correspondant dans le PPTA dont chaque état est* plein.

Par sub-linéaire, nous stipulons que le ratio $f(n)/n$ tend vers 0 quand n tend vers l'infini. Pour fixer les idées, on pourra considérer que $f(n) = \sqrt{n}$.

Définition 10 (Bon PPTA)
Soit q_i *et* q_j *deux états pleins du PPTA. Le PPTA est* bon *si* q_i *et* q_j *sont similaires si et seulement si ils sont équivalents.*

4 CONVERGENCE DE L'ALGORITHME

Notre algorithme fusionne toutes les paires d'états du PPTA qui sont à la fois pleins et similaires. Nous prouvons dans un premier temps que le PPTA sera *presque toujours* plein, et qu'il sera *presque toujours* bon.

LE PPTA EST *presque toujours* PLEIN

Notons dans un premier temps qu'il n'existe qu'un nombre fini de chemins semi-simple – en particulier, chaque état ne pouvant apparaître qu'au plus deux fois, il y a moins de $(|Q| \times |\Sigma|)^{2|Q|}$ chemins semi-simples. Chaque chemin aura une probabilité non nulle qui est le produit des probabilités des transisions qui le constitue, multiplié par la probabilité d'arrêt du dernier état d'analyse. Si nous considérons le chemin avec la plus petite probabilité (disons ϵ), le compte (ou l'effectif) de ce chemin, pour de grandes valeurs de n sera proche de $n\epsilon$, qui deviendra plus grand que $f(n)$ pour des valeurs suffisamment grandes de n. Ceci peut être prouvé formellement en utilisant par exemple les bornes de Chernov (Feller, 1950).

LE PPTA EST *presque toujours* BON

La somme des comptes des états du PPTA sera la somme des longueurs de toutes les chaînes de l'ensemble d'apprentissage (incluant les redondances). L'espérance de cette valeur est simplement n fois $E[|s|]+1$, valeur qui est finie. En conséquence, la somme des comptes du PPTA sera presque toujours inférieure à disons $(2E[|s|]+1)n$, et donc, par un argument de comptage, le nombre d'états pleins sera presque toujours inférieur à $(2E[|s|]+1)\sqrt{n}$. Par conséquent, le nombre de comparaisons que nous allons faire sera inférieur à $4n(E[|s|]+1)^2 = bn$, qui est borné linéairement.

Nous montrons dans un premier temps que deux états pleins et similaires seront équivalents pour des valeurs suffisamment grandes de n. q_i, q_j sont similaires donc, $d_*(\Gamma(q_i), \Gamma(q_j)) < \epsilon^a_{n(q_i)} + \epsilon^a_{n(q_j)}$ et donc, puisqu'ils sont pleins, $d_*(\Gamma(q_i), \Gamma(q_j)) < 2\epsilon^a_{\sqrt{n}}$. Nous savons par ailleurs que ces distributions seront proches des distributions des états originaux. Donc, avec grande probabilité $d_*(\Gamma(q_i), \Gamma(\phi(q_i))) < \epsilon^a_{\sqrt{n}}$; de même, pour q_j. En utilisant l'inégalité triangulaire, il vient $d_*(\Gamma(\phi(q_i)), \Gamma(\phi(q_j))) < 4\epsilon^a_{\sqrt{n}}$ avec une probabilité plus grande que $1 - 12n^{-a}$. Vu que dans la cible $d_{min} > 0$, on a, pour toutes sauf pour un nombre fini de valeurs de n, $d_{min} > 4\epsilon^a_{\sqrt{n}}$. Si nous prenons $a > 2$, et que nous remarquons que le nombre de paires d'états pleins est borné linéairement, nous pouvons voir que la probabilité qu'il y ait un erreur avec n états est bornée par $bn12n^{-a}$. Or la série $bn12n^{-a}$ converge, et donc, par application du lemme Borel-Cantelli, il n'y aura qu'un nombre fini d'erreurs.

Réciproquement, nous voulons montrer que les états équivalents pleins seront similaires ; étant donné que les états sont pleins, nous savons que les deux distributions seront toutes deux proches de la vrai distribution. Un raisonnement similaire mais plus simple nous amène à montrer qu'avec probabilité un, il n'y aura qu'un nombre fini d'erreurs de cette sorte.

LE RÉSULTAT EST STRUCTURELLEMENT CORRECT

Nous avons montré jusqu'ici que le PPTA sera presque toujours bon. Nous montrons maintenant que si le PPTA est bon et plein, alors le résultat sera structurellement isomorphe à la cible.

Lorsque les états similaires sont définis, nous les fusionnons. Si deux états à fusionner ont des transitions étiquetées par le même symbole, nous fusionnons récursivement les états cibles de ces transitions. Nous montrons maintenant que le résultat de cette procédure est isomorphe à la cible si le PPTA est plein et bon.

Premièrement, la fusion, et donc la fusion récursive, ne s'appliquera que sur des états équivalents. En conséquence, nous ne fusionnerons jamais incorrectement. Chaque état de la cible devant avoir au moins un état plein dans le PPTA, accessible par un chemin semi-simple, nous savons qu'il y aura au moins une copie de chaque état de la cible dans le résultat. De plus, chaque transition fera partie d'un chemin semi-simple et fera donc partie de la cible. En conséquence, il y aura dans le résultat une partie structurellement isomorphe à la cible; par ailleurs il ne peut pas y avoir d'autre état dans le résultat car sinon, nous aurions une transition qui mènerait en dehors de la partie isomorphique du résultat.

Une technique alternative consisterait à simplement supprimer les états du PPTA qui ne sont pas bons et pleins.

IDENTIFICATION DES COEFFICIENTS RATIONNELS

La structure de l'automate cible étant identifiable, il nous reste à identifier les valeurs des probabilités de l'automate. Ceci sera fait via de l'algorithme de (de la Higuera & Thollard, 2000) que nous présentons maintenant.

Dans une première étape, chaque probabilité est estimée par le maximum de vraisemblance, *i.e.* par le nombre de fois où une transition est utilisée normalisée correctement. Les auteurs utilisent un arbre de Stern-Brocot (Graham *et al.*, 1992) pour identifier l'ensemble des valeur rationnelles simples et proches de l'estimation. Ils choisissent alors la fraction a/b qui est la plus simple dans l'arbre de Stern-Brocot telle que

$$\left|\frac{m}{n} - \frac{a}{b}\right| < \sqrt{\frac{\log\log n}{n}} \tag{1}$$

Ils montrent alors que cette méthode identifie *presque toujours* les nombres rationnels. En effet, cet algorithme simplifie une estimation empirique comme 101/201 à une approximation de la forme 1/2.

Étant donné que nous ne ferons, avec probabilité un, qu'un nombre fini d'erreurs dans l'identification de la structure de l'automate, et étant donné que l'algorithme sus-décrit ne fera, avec probabilité un, qu'un nombre fini d'erreurs pour identifier les valeurs des transitions, nous concluons à l'identification à la limite avec probabilité un de la classe des PDFA.

ANALYSE DE COMPLEXITÉ

Supposons que nous ayons vu n chaînes avec un total de m lettres. Notre algorithme répondra, en un temps polynomial en $n + m$. La construction du PPTA est linéaire en m, et il aura au plus $m+n$ états. Il ne peut y avoir qu'au plus $(m+n)/\sqrt{n}$ états pleins dans le PPTA et donc nous ferons moins de $(m + n)^2/n$ comparaisons de distributions, chacune d'elles ayant une complexité bornée par le nombre d'états du PPTA. Fusionner une paire d'états et déterminiser récursivement le résultat est linéaire en m et nous ne devons faire cette opération qu'au plus m fois. Par conséquent, la complexité totale de la technique présentée est $O((m+n)^3/n)$.

On note tout de même que l'algorithme pourra avoir besoin d'un nombre exponentiel de données avant de pouvoir identifier la cible. Cette remarque nous a poussé à considérer un autre cadre d'apprentissage, le cadre PAC, que nous allons traiter dans la section suivante.

5 KL-PPAC APPRENABILITÉ

Dans le même esprit que le critère présenté dans (Ron *et al.*, 1995) nous proposons ici une adaptation du cadre PAC au cadre probabiliste

Définition 11 (Divergence de Kullback-Leibler)
La divergence de Kullback-Leibler (Cover & Thomas, 1991) entre deux distributions D_1, D_2 est définie comme $D(D_1||D_2) = \sum_\sigma D_1(\sigma) \log \frac{D_1(\sigma)}{D_2(\sigma)}$

Définition 12 (Apprentissage KL-PPAC)
Nous dirons qu'une classe de distributions $\mathcal{C}$ sur $\Sigma^\star$ est KL-PPAC apprenable par un algorithme A si pour chaque cible $C \in \mathcal{C}$, pour chaque paramètre de précision $\epsilon > 0$ pour chaque paramètre de confiance $\delta > 0$ et pour une mesure de la taille de la cible $|C|$, il existe une fonction $q(1/\epsilon, 1/\delta, |\Sigma|, |C|)$ telle que si la taille de l'ensemble d'apprentissage excède q l'algorithme A produit une hypothèse H telle que $Pr(D(C||H) > \epsilon) < \delta$.

Nous discuterons plus loin le choix de la divergence de Kullback-Leibler comme mesure de distance.

5.1 Les PDFA ne sont pas KL-PPAC apprenables

Nous prouvons dans cette section que les PDFA ne peuvent pas être appris dans le paradigme KL-PPAC. Supposons qu'il existe un algorithme permettant cet apprentissage. Nous choisissons une valeur particulière de ϵ et δ, et nous construisons une famille de PDFA pour lesquels, avec une probabilité supérieure à δ l'algorithme ne verra pas une portion de la cible dans les $q(1/\epsilon, 1/\delta, |\Sigma|, |C|)$ points de l'ensemble d'apprentissage dont il disposera. Nous montrons alors que la divergence de l'hypothèse avec la cible peut devenir aussi grande que l'on veut.

Lemme 1
Pour tout PDFA, H, pour tout $d > 0$ il existe un PDFA à un état C tel que $D(C||H) > d$.

S'il existe une chaîne s tel que $P_H(s) = 0$ alors, tout C qui génère s aura une divergence infinie. Nous supposons donc que H génère toutes les chaînes.

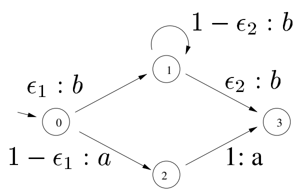

FIG. 2 – *Famille d'automate non apprenable dans le cadre KL-PPAC*

Prenons une lettre de l'alphabet et considérons le chemin dans H pour toutes les chaînes de la forme a^k. Il doit y avoir une valeur maximale de la transition de probabilité pour a dans cette séquence. Appelons cette valeur a_{max}. De même, il doit y avoir une valeur maximale pour la probabilité d'arrêt, disons t_{max}. Donc, $Pr_H(a^k) < a^k_{max} t_{max}$.

Considérons l'automate C, qui possède une seule transition et qui génère a avec la valeur $1-p$. On a donc $Pr_C(a^k) = (1-p)^k p$. Le calcul de la divergence entre C et H donne :

$$\begin{aligned} D(C||H) &= \sum_s Pr_C(q_I, s) \log \frac{Pr_C(s)}{\gamma_H(q_I, s)} = \sum_{k \geq 0} (1-p)^k p \log \frac{(1-p)^k p}{Pr_H(q_I, a^k)} \\ &\geq \sum_{k \geq 0} (1-p)^k p \log \frac{(1-p)^k p}{a^k_{max} t_{max}} \\ &\geq \frac{1-p}{p} \log \frac{1-p}{a_{max}} + \log \frac{p}{t_{max}} \end{aligned}$$

Lorsque p tend vers zéro la divergence se comporte comme $\log x + 1/x$, qui croît sans limite. Cela revient à augmenter l'espérance de la longueur des chaînes générées par l'automate et donc l'entropie de la distribution.

Cela signifie que si nous avons un ensemble de chaînes dont la somme des probabilités est très faible, la divergence relativement au sous langage peut être significative.

Forts de cet analyse, nous construisons maintenant une famille d'automates (cf figure 2) non apprenable dans le cadre KL-PPAC : nous considérons les automates qui avec la probabilité $1-\epsilon_1$ génèrent la chaîne a et avec la probabilité ϵ_1 génèrent b et rentre dans un sous automate dont l'espérance de la longueur des chaînes est très grande, nous ferons probablement une erreur significative. Si nous définissons ϵ_1 suffisamment petit pour que

$$\delta > 1 - (1-\epsilon_1)^{q(\epsilon, \delta)} \tag{2}$$

alors, avec probabilité plus grande que $1-\delta$ les données que l'algorithme verra seront une succession de a. Dans ce cas, nous ne pouvons pas savoir ce que le sous-automate contient après le b. Les automates inférés auront donc une divergence par rapport au sous-automate suppérieure à ϵ/ϵ_1. On a donc, avec une probabilité supérieure à δ, une hypothèse dont la divergence par rapport à la cible est plus grande que ϵ.

6 Discussion

Comme mentionné plus haut, la preuve proposée dans (Carrasco & Oncina, 1999) n'est pas complètement satisfaisante car elle ne prend pas en compte les faibles effectifs des états [1]. En effet, même avec de grands ensembles d'apprentissage, il existera certains états du PPTA qui auront de faibles comptes. Ces états ne satisferont pas alors les prérequis à l'utilisations des bornes de Hoeffding (Feller, 1950), bornes utilisées dans leur algorithme.

D'un autre coté, Ron et *al.* (1995) prouvent la KL-PPAC apprenabilité d'une sous-classe d'automates acycliques probabilistes en comparant explicitement les états qui ont des comptes supérieurs à un certain seuil. Clairement, nous avons besoin d'information supplémentaire sur la cible afin d'espérer pouvoir réaliser le KL-PPAC apprentissage des PDFA. L'extension du résultat de Ron et *al.* à des automates arbitraires nous semble raisonnable étant donné l'information qu'ils utilisent. Dans notre contexte, cela correspond à une borne supérieure sur le nombre d'états de l'automate cible, une borne inférieur sur la valeur de d_{min}, et une information supplémentaire sur l'entropie de la cible ou l'espérance de la taille de l'automate.

Le résultat négatif présenté ici peut sembler en conflit avec le résultat de Abe et Warmuth (1992) qui prouvent que l'apprentissage au sens PAC peut être obtenu avec un ensemble d'apprentissage polynomial si l'on dispose d'une borne sur le nombre d'états de la cible et si l'on n'a pas de contrainte de complexité calculatoire. Ils considèrent cependant un type d'automate un peu différent qui produit des chaînes de taille finie et considère les distributions de probabilités sur Σ^n; nous considérons pour notre part des distributions sur $\Sigma^\star$, ce qui est plus difficile.

6.1 Mesures de distances

Pour mesurer la distance entre la cible et l'hypothèse, nous avons utilisé la divergence de Kullback-Leibler. Cette mesure est un choix naturel en raison de ses liens avec l'estimation du maximum de vraisemblance. De plus la KL divergence borne supérieurement plusieurs autres distances (par exemple la norme L_1). Par ailleurs, Kearns et *al.* (1994) rapellent que si nous considérons des distributions sur Σ^n, alors, pour toute cible C, $D(C||U) \leq n \log|\Sigma|$, où U est la distribution uniforme. Dans le cadre de distributions sur $\Sigma^\star$ cette borne n'existe plus. Ainsi, dans un certain sens, dans notre cas, la divergence de KL est trop contraignante.

Une question plus générale est donc la question de savoir si la tâche peut être possible si on utilise une autre mesure de distance. Les choix possibles concernant

1. Nous avons géré ce problème par l'introduction de la notion d'états plein.

la distance sont $\sum |D_1(x) - D_2(x)|$, la distance quadratique $\sum(D_1(x) - D_2(x))^2$ ou la distance de Hellinger $\sum(\sqrt{D_1(x)} - \sqrt{D_2(x)})^2$. D'autres mesures de distance sont cependant trop faciles étant donné que nous essayons de modéliser des distributions de probabilités discrètes. En particulier, l'utilisation de la distance d_* définie plus haut, affaiblirait trop le cadre : étant donné qu'il y a au plus $1/\epsilon$ chaînes de probabilité plus grande que ϵ, par l'utilisation des bornes de Hoeffding nous n'avons besoin que d'un ensemble d'apprentissage de taille n pour apprendre, avec $n > \frac{1}{2\epsilon^2} \log \frac{2}{\epsilon\delta}$ pour être sur que $P[d_*(H,T) > \epsilon] < \delta$

Remerciements :

Nous tenons à remercier Colin de la Higuera et Marc Sebban pour leurs commentaires d'une version préliminaire de ce papier.

RÉFÉRENCES

ABE N. & WARMUTH M. (1992). On the computational complexity of approximating distributions by probabilitic automata. *Machine Learning*, **9**, 205–260.

ANGLUIN D. (1988). *Identifying languages from stochastic examples*. Rapport interne YALEU/ DCS/RR-614, Yale University, Dept. of Computer Science, New Haven, CT.

CARRASCO R. C. & ONCINA J. (1999). Learning deterministic regular grammars from stochastic samples in polynomial time. *Theoretical Informatics and Applications*, **33**(1), 1–20.

COVER T. & THOMAS J. (1991). *Elements of Information Theory*. Wiley Interscience Publication.

DE LA HIGUERA C. & THOLLARD F. (2000). Identification in the limit with probability one of stochastic deterministic finite automata. In *ICGI-2000*, Lisbon.

FELLER W. (1950). *An introduction to probability theory and its applications*. New-York: John Wiley and Sons.

GILLMAN D. & SIPSER M. (1994). Inference and minimization of hidden markov chains. In *COLT'94*, p. 147–158, New Brunswick,NJ,USA.

GRAHAM R. L., KNUTH D. E. & PATASHNIK O. (1992). *Concrete Mathematics*. Addison Wesley Publishing Compagny.

HORNING J. J. (1972). A procedure for grammatical inference. *Information Processing*, **71**, 519–523.

KAPUR S. & BILARDI G. (1992). Language learning from stochastic input. In *Proceedings of the fifth conference on Computational Learning Theory*, p. 303–310, Pittsburgh.

KEARNS M., MANSOUR Y., RON D., RUBINFELD R., SCHAPIRE R. & SELLIE L. (1994). On the learnability of discrete distributions. In *Proc. of the 25th Annual ACM Symposium on Theory of Computing*, p. 273–282.

RON D., SINGER Y. & TISHBY N. (1994). Learning probabilistic automata with variable memory length. In *Seventh Conf. on COLT*, p. 35–46, New Brunswick: ACM Press.

RON D., SINGER Y. & TISHBY N. (1995). On the learnability and usage of acyclic probabilistic finite automata. In *ACM*, p. 31–40, Santa Cruz CA USA: COLT'95.

VALIANT L. G. (1984). A theory of the learnable. *Communications of the ACM*, **27**(11), 1134–1142.