Algorithmes pour Grammaires Probabilistes

Bernard Merialdo
Institut EURECOM
BP 193
06904 Sophia-Antipolis
merialdo@eurecom.fr

Résumé

Dans cette article nous souhaitons faire une présentation des différents algorithmes permettant de mettre en oeuvre des grammaires probabilistes pour le traitement du langage naturel. Nous détaillons en particulier des travaux récents concernant les grammaires hors-contexte. Les algorithmes concernés couvrent quatre questions: comment estimer les probabilités des règles, comment trouver la meilleure analyse, comment calculer la probabilité d'une phrase, et enfin comment calculer la probabilité d'un début de phrase. Les réponses à ces questions font intervenir des variantes probabilistes des algorithmes classiques d'analyse, selon des stratégies qui peuvent être ascendantes ou descendantes. Finalement, nous mentionnons quelques travaux concernant d'autres aspects des grammaires probabilistes.

I. Introduction

L'utilisation de statistiques dans le traitement du langage naturel fait l'objet de recherches de plus en plus nombreuses. L'augmentation de la puissance de calcul et des capacités de stockage des ordinateurs, la disponibilité croissante de grands volumes de textes sous forme électronique, la construction de bases de données linguistiques (dictionnaires, textes étiquetés ou analysés) sont autant de facteurs positifs pour cette évolution. Ces statistiques peuvent être faites à différents niveaux: fréquences de mots, de suites de catégories grammaticales, co-occurrences etc... et servir à diverses tâches: sélection de vocabulaire, correspondance pour traduction, construction de modèles de langage...

L'approche statistique intervient très fortement dans les modèles de langage basés sur des modèles de Markov, c'est-à-dire des automates finis probabilistes, qui sont très largement utilisés pour différentes phases de la reconnaissance de la parole. Les algorithmes pour utiliser ces modèles sont maintenant bien connus et maîtrisés (Charniak, 1993), que ce soit pour la phase d'apprentissage des probabilités à partir des données (algorithme de Baum-Welch), ou pour la phase de calcul lors de l'utilisation du modèle pour la tâche concernée (algorithme de Viterbi ou bien Maximum de Vraisemblance).

Les automates à états finis présentant quelques limitations, certains travaux se sont intéressés à des modèles plus complexes, en particulier les grammaires hors-contexte. Les algorithmes concernés deviennent alors plus délicats à concevoir et à mettre en oeuvre. Ils se fondent sur des extensions des algorithmes classiques: Baum-Welch, Viterbi, CYK, LR et Earley. Dans cette présentation, nous souhaitons faire un rapide survol des principaux algorithmes qui ont été proposés récemment, en particulier dans les travaux de (Jelinek et al., 1992) et (Stolcke, 1993).

II. Grammaires hors-contexte probabilistes

Les grammaires hors-contexte sont bien connues dans le cadre du langage naturel. L'introduction des probabilités s'effectue en associant une probabilité à chaque règle. Ces

quantités doivent former une distribution pour toutes les règles dont le terme de gauche est un non-terminal donné: si X est un non-terminal, la somme des probabilités de toutes les règles $X \rightarrow \lambda$ doit être 1(λ est une suite de terminaux et non-terminaux).

$$\sum_{X \rightarrow \lambda} P(X \rightarrow \lambda) = 1$$

Voici un exemple simple de grammaire hors-contexte probabiliste:

$S \rightarrow AB$	$p = 0.7$
$S \rightarrow BA$	$p = 0.3$
$A \rightarrow a$	$p = 0.6$
$A \rightarrow b$	$p = 0.4$
$B \rightarrow b$	$p = 1.0$

On peut alors définir la probabilité d'une analyse d'une phrase:

$$p(S \rightarrow AB \rightarrow bB \rightarrow bb) = 0.7 \times 0.4 \times 1.0 = 0.28$$

ainsi que la probabilité que la grammaire produise une phrase donnée:

$$p(S \rightarrow bb) = p(S \rightarrow AB \rightarrow bB \rightarrow bb) + p(S \rightarrow BA \rightarrow bA \rightarrow bb)$$
$$= 0.7 \times 0.4 \times 1.0 + 0.3 \times 1.0 \times 0.4 = 0.4$$

Les algorithmes classiques d'analyse doivent être modifiés pour intégrer cet aspect probabiliste. Les utilisations possibles des grammaires probabilistes conduisent à quatre types de questions (Jelinek et al., 1992):

1. si les règles d'une grammaire sont données, comment estimer leurs probabilités à partir d'exemples de phrases produites?
2. quelle est l'analyse la plus probable d'une chaîne x selon une grammaire G?
3. quelle est la probabilité qu'une chaîne x soit produite par une grammaire G?
4. quelle est la probabilité que la chaîne x soit le préfixe d'une chaîne engendrée par G (c'est-à-dire le début d'une phrase valide)?

La question 1 est celle de l'apprentissage: à partir d'un ensemble de textes (phrases valides), comment estimer les probabilités des règles. Cela serait facile si on disposait des analyses correspondantes (il n'y aurait alors qu'à compter le nombre de fois qu'une règle est appliquée), mais ce n'est qu'exceptionnellement le cas.

La question 2 est celle de la désambiguïsation des analyses multiples: lorsque la grammaire autorise plusieurs analyses, chacune possède une probabilité, et l'analyse optimale (au sens du modèle) est celle de plus forte probabilité.

Les questions 3 et 4 interviennent par exemple en reconnaissance de parole (souvent réalisée en reconnaissant la phrase de gauche à droite) lorsque plusieurs hypothèses (phrases partielles) sont proposées par le niveau acoustique et doivent être comparées par le niveau linguistique. Ce même problème se retrouve aussi en reconnaissance de l'écriture, ainsi qu'en traduction automatique.

Pour introduire progressivement les méthodes qui interviennent dans le traitement de ces questions, nous les traiterons dans l'ordre 3, 4, 2, 1.

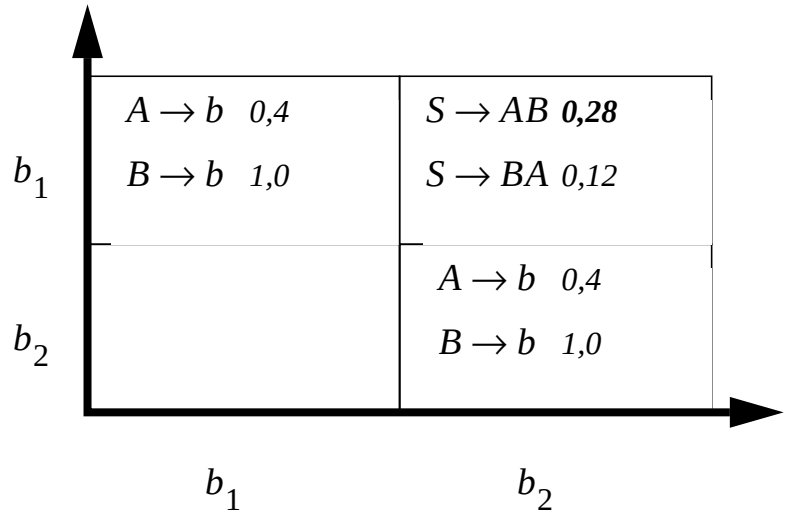
III. Probabilité d'une phrase

La probabilité d'une phrase selon une grammaire probabiliste est la somme des probabilités de ses analyses. Pour la calculer, il suffit donc de construire toutes ces analyses, puis de calculer leurs probabilités et de les ajouter. Cette méthode est toutefois inefficace si le nombre d'analyses est élevé. Les algorithmes d'analyse peuvent être modifiés pour calculer des probabilités intermédiaires qui permettent de factoriser les calculs en fonction des ressemblances de certaines analyses, et donc d'arriver au même résultat plus rapidement.

Nous décrivons les modifications de deux algorithmes classiques: l'algorithme CYK (Cocke-Younger-Kasami) basé sur une analyse ascendante, et l'algorithme d'Earley basé sur une analyse descendante.

L'algorithme CYK

L'algorithme CYK suppose que la grammaire a été mise sous la forme normale de Chomsky (où les seuls types de règles sont de la forme $X \rightarrow AB$ ou $X \rightarrow a$). Il réalise une analyse ascendante en construisant un tableau à deux entrées dont la case en ligne *i* et colonne *j* contient les analyses de la sous-chaîne $m_i m_{i+1} \ldots m_j$. Le tableau est construit en commençant par la première diagonale, correspondant à la génération des terminaux, puis en remplissant progressivement les diagonales supérieures. Par exemple, l'analyse de la chaîne bb selon la grammaire citée précédemment donnera le tableau:



Le rajout d'informations dans une case du tableau correspond toujours à l'application d'une règle de la grammaire, on peut donc tenir à jour la probabilité correspondante. Lorsqu'une case du tableau est remplie, on ne conserve que les non-terminaux de gauche qui y apparaissent, affectés de la somme des probabilités des hypothèses qui les ont introduits. Par exemple, dans le tableau précédent, on ne gardera dans la case (1,2) que le non-terminal *S* avec une probabilité de *0,4*. Le fait de sommer ces probabilités dans une case conduit à une factorisation des calculs pour les différentes analyses possibles contenant ce non-terminal à cet endroit du tableau. Il faut également remarquer que la probabilité ainsi associée à un non-terminal dans une case correspond à la probabilité que ce non-terminal génère la sous-chaîne $m_i m_{i+1} \ldots m_j$. En particulier, dans la case (1,*l*) on trouvera donc la probabilité que la racine de la grammaire engendre toute la phrase.

L'algorithme d'Earley

L'algorithme d'Earley est un exemple d'algorithme utilisant une stratégie d'analyse descendante. Il définit des états, $i:_k X \to A \cdot B$, dont la signification est que l'on a traité les *i* premiers symboles de la phrase, et que depuis le symbole en position *k*, on essaie d'analyser selon la règle $X \to AB$ dont on a déjà fait coïncider la partie *A*. L'algorithme opère de gauche à droite, à partir de l'état initial $0:_0 \to \cdot S$ en appliquant trois types d'opérations:

- **prédiction**: pour tout état $i:_k X \to \lambda \cdot Y\mu$ et toute règle $Y \to \upsilon$, on rajoute l'état

$$i:_i Y \to \cdot \upsilon$$

- **analyse**: pour tout état $i:_k X \to \lambda \cdot a\mu$, si a est un non-terminal qui coïncide avec le prochain symbole à analyser, on rajoute l'état

$$i+1:_k X \to \lambda a \cdot \mu$$

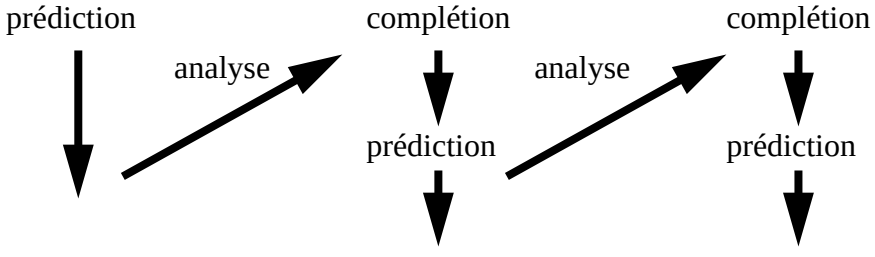
- **complétion**: pour tout état complet $i:_j Y \to \upsilon \cdot$ et pour tout état précédant $j:_k X \to \lambda \cdot Y\mu$, on ajoute l'état

$$i:_k X \to \lambda Y \cdot \mu$$

Le processus se poursuit jusqu'à l'obtention de l'état final $l:_0 \to S \cdot$. Par exemple, l'analyse de la phrase bb donnerait:

		b		b	
0:	$_0 \to \cdot S$	1:	$_0A \to b \cdot$	2:	$_1A \to b \cdot$
0:	$_0S \to \cdot AB$	1:	$_0B \to b \cdot$	2:	$_1B \to b \cdot$
0:	$_0S \to \cdot BA$	1:	$_0S \to A \cdot B$	2:	$_0S \to AB \cdot$
0:	$_0A \to \cdot a$	1:	$_0S \to B \cdot A$	2:	$_0S \to BA \cdot$
0:	$_0A \to \cdot b$	1:	$_1A \to \cdot a$	2:	$_0 \to S \cdot$
0:	$_0B \to \cdot b$	1:	$_1A \to \cdot b$		
		1:	$_1B \to \cdot b$		

L'analyse s'effectue en suivant toujours la même succession d'opérations:



Lorsqu'on veut calculer la probabilité d'une analyse, on regarde la suite d'opérations

qui la composent, et on multiplie les probabilités des règles utilisées dans toutes les phases de prédiction. Pour mettre en oeuvre de façon efficace ce calcul, on est conduit à une variante probabiliste de l'algorithme faisant intervenir deux probabilités intermédiaires pour chaque état (Stolcke, 1993):

- $\alpha_i({}_k X \to \lambda \cdot \mu)$ probabilité d'arriver à cet état après avoir généré les i premiers symboles,
- $\gamma_i({}_k X \to \lambda \cdot \mu)$ probabilité d'arriver à cet état après avoir généré les i-k derniers symboles depuis le non-terminal X selon cette règle.

Pendant l'analyse, on met à jour ces probabilités de façon adaptée pour chaque opération (la notation += désigne le cumul de ces valeurs pour toutes les façons possibles d'arriver au même état):

- **prédiction**: $i: {}_k X \to \lambda \cdot Y\mu \quad [\alpha, \gamma]$ donne $i: {}_i Y \to \cdot \upsilon \quad [\alpha', \gamma']$ avec les relations:

$$\alpha' \mathrel{+}= \alpha \times P(Y \to \upsilon)$$
$$\gamma' = P(Y \to \upsilon)$$

- **analyse**: $i: {}_k X \to \lambda \cdot a\mu \quad [\alpha, \gamma]$ donne $i+1: {}_k X \to \lambda a \cdot \mu \quad [\alpha', \gamma']$ avec les relations:

$$\alpha' = \alpha$$
$$\gamma' = \gamma$$

- **complétion**: $i: {}_j Y \to \upsilon \cdot \quad [\alpha'', \gamma'']$ et $j: {}_k X \to \lambda \cdot Y\mu \quad [\alpha, \gamma]$ donnent

$$i: {}_k X \to \lambda Y \cdot \mu \quad [\alpha', \gamma']$$

avec les relations:

$$\alpha' \mathrel{+}= \alpha \times \gamma''$$
$$\gamma' \mathrel{+}= \gamma \times \gamma''$$

(remarquons que c'est dans cette étape que se justifie le besoin de la probabilité γ).

Lorsque ce calcul a été mené sur l'intégralité de l'analyse, la probabilité de la chaîne se trouve alors comme:

$$\alpha_l({}_0 \to S \cdot) = \gamma_l({}_0 \to S \cdot)$$

Récursion

Le raisonnement précédent n'est pas complètement exact. Une difficulté technique provient de la possibilité de récursion dans les règles de la grammaire. Si l'on a par exemple une règle de la forme $X \to X\mu$ de probabilité p, le processus de prédiction probabiliste devrait faire les opérations suivantes:

$i: \quad {}_i X \to \cdot X\mu \quad p$

$i: \quad {}_i X \to \cdot X\mu \quad p \times p$

$i: \quad {}_i X \to \cdot X\mu \quad p \times p \times p$

Dans le cas non-probabiliste, l'état est identique et la récursion s'arrête

immédiatement, par contre, dans le cas probabiliste, les états sont différents parce que la probabilité associée est différente. En théorie, il faudrait définir une infinité d'états, ce qui n'est évidemment pas réalisable. La situation peut se compliquer encore lorsque la récursion n'est pas apparente sur une règle unique, mais sur une succession de règles qui retournent au même non-terminal, comme dans $X \to Y\lambda \qquad Y \to X\mu$. (c'est un problème similaire à celui des boucles vides dans les modèles de Markov).

La résolution de ce problème passe par un calcul matriciel permettant de calculer directement la probabilité à associer à l'état. On définit:

- la relation du "coin-gauche" par

$$X \to_L Y \text{ si et seulement si il existe une règle } X \to Y\lambda$$

- P_L la matrice des probabilités $P(X \to_L Y) = \sum_{X \to Y\lambda} P(X \to Y\lambda)$
- $X \Rightarrow_L Y$ est la fermeture transitive réflexive de la relation $\to_L$
- et R_L la matrice des probabilités $R(X \Rightarrow_L Y)$

R_L vérifie la relation de récurrence:

$$\boxed{R(X \Rightarrow_L Y) = \delta(X,Y) + \sum_Z P(X \to_L Z) \times R(Z \Rightarrow_L Y)}$$

soit sous forme matricielle: $R_L = I + P_L R_L$, ce qui permet de la calculer directement par la relation:

$$\boxed{R_L = (I - P_L)^{-1}}$$

Avec ces définitions, la formulation correcte de l'étape de prédiction devient:

- prédiction: i: $_k X \to \lambda \cdot Z\mu \quad [\alpha, \gamma]$ donne i: $_i Y \to \cdot \upsilon \quad [\alpha', \gamma']$ avec les relations:

$$\boxed{\begin{aligned} \alpha' + &= \alpha \times R(Z \Rightarrow_L Y) \times P(Y \to \upsilon) \\ \gamma' &= P(Y \to \upsilon) \end{aligned}}$$

Des modifications analogues sont à apporter aux étapes de complétion, qui peuvent elles aussi donner lieu à des applications infinies de règles dans le cas probabiliste. Par contre, il n'y a pas de problème pour les étapes d'analyse, qui ne peuvent être infinies car elles correspondent à l'émission d'un symbole.

IV. Probabilité d'un préfixe

Il est parfois important de savoir si une suite de mots $m_1 m_2 \ldots m_i$ constitue un bon début de phrase, c'est-à-dire de calculer la probabilité qu'une phrase de la grammaire commence par cette suite (c'est alors le préfixe de la phrase). Cette probabilité est la somme des probabilités de ces phrases, qui peuvent malheureusement être en nombre infini.

L'algorithme CYK ne permet pas de calculer directement la probabilité d'un préfixe. Toutefois, une méthode efficace pour ce calcul est présentée dans (Jelinek and Lafferty, 1991). Elle utilise une approche similaire à celle du "coin gauche" précédemment exposée, en remplaçant le calcul d'une somme infinie par celui de l'inverse d'une matrice.

Dans l'algorithme d'Earley, la probabilité d'un préfixe $m_1 m_2 \ldots m_i$ apparaît de façon plus naturelle à partir des probabilités α. Elle peut se calculer comme:

$$\sum_{{}_kX \to \lambda m_i \cdot \mu} \alpha_i\left({}_kX \to \lambda m_i \cdot \mu\right)$$

V. Meilleure analyse d'une phrase

Les probabilités servent de façon naturelle à lever les ambiguïtés. Lorsqu'une phrase possède plusieurs analyses, le modèle probabiliste dit que l'analyse optimale est celle ayant la probabilité la plus élevée. Le calcul de cette analyse de plus forte probabilité, aussi appelée analyse de Viterbi, s'effectue par des algorithmes très proches de ceux utilisés pour le calcul de la probabilité d'une phrase. Le processus est identique, à cela près qu'au lieu d'ajouter les probabilités d'hypothèses comparables, on prend le maximum pour ne garder que la meilleure, suivant en cela le principe classique de programmation dynamique.

Dans l'algorithme CYK, on ne garde que la meilleure façon d'obtenir un non-terminal dans une case du tableau. Dans l'algorithme d'Earley, on effectue les mêmes étapes de prédiction, analyse et complétion, mais on utilise un maximum aux endroits où l'on sommait des probabilités.

La recherche de la meilleure analyse par ces méthodes sera souvent assez rapide. Remarquons qu'on peut parfois modifier légèrement ces algorithmes pour recherches les N meilleures analyses, ou bien utiliser un seuil pour ne produire que celles ayant une bonne probabilité.

VI. Apprentissage

Nous nous plaçons dans le cas où l'on souhaite ajouter des probabilités à une grammaire pré-existante. Les règles sont en général écrites manuellement pour fournir des analyses possibles de phrases, et les probabilités serviront à ordonner ces analyses par probabilité décroissante.

Une situation d'apprentissage commode est celle où l'on dispose d'un corpus de textes préalablement analysé (de façon correcte) selon cette grammaire. On peut alors estimer la probabilité d'une règle par sa fréquence relative d'apparition, calculée à partir du nombre d'occurrences de la règle et du nombre d'occurrences de sa tête (corps de gauche):

$$p\left(X \to A_1 A_2 \ldots A_n\right) \;=\; \frac{n\left(X \to A_1 A_2 \ldots A_n\right)}{n\left(X\right)}$$

On se trouve rarement dans cette situation favorable, car les corpus de textes analysés sont rares. L'attirail probabiliste fournit néanmoins un algorithme permettant d'estimer les valeurs des probabilités à partir du seul texte (sans analyse). Cet algorithme, appelé *'Inside-Outside'*, est le correspondant de l'algorithme de Baum-Welch (Baum et al., 1970) pour les

modèles de Markov cachés et a été proposé dans (Baker, 1979). Le principe en est simple: chaque phrase du texte fournit un certain nombre d'analyses possibles, chacune affectée d'une probabilité. On compte le nombre d'occurrences de chaque règle dans chaque analyse, et on les cumules en pondérant par les probabilités des analyses. Puis on cumule ces comptes pour toutes les phrases du corpus. A partir d'une valeur initiale des probabilités des règles (permettant de calculer les probabilités des analyses), on peut donc avoir une nouvelle estimation des nombres d'occurrences dans les analyses possibles, donc une nouvelle estimation des probabilités des règles. Un théorème nous garantit que ces nouvelles estimations des probabilités sont "meilleures" que les probabilités initiales, et que ce processus réitéré convergera vers un optimum local.

Une mise en oeuvre efficace de l'algorithme '*Inside-Outside*' est un peu plus délicate car elle demande de factoriser les calculs pour éviter une explosion combinatoire. Elle passe par le calcul des probabilités intermédiaires (justifiant le nom de l'algorithme):

$$p_{inside}(X, i, j) = p_I(X, i, j) = p(X \rightarrow m_i m_{i+1} \ldots m_j)$$

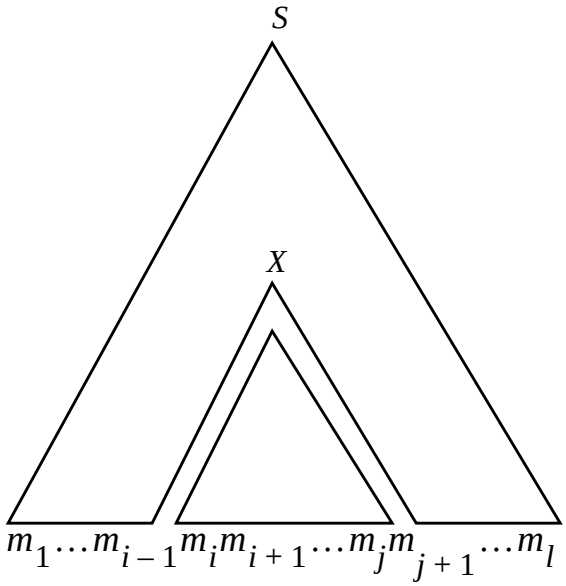
$$p_{outside}(X, i, j) = p_O(X, i, j) = p\left(S \rightarrow m_1 \ldots m_{i-1} X m_{j+1} \ldots m_l\right)$$

A partir de ces probabilités, on peut calculer le nombre de fois qu'une règle sera utilisée dans l'analyse d'une phrase:

$$C(X \rightarrow AB) = \frac{p(X \rightarrow AB) \sum_{i,j} p_O(X, i, j) \sum_{k=i}^{j-1} p_I(A, i, k) \times p_I(B, k+1, j)}{p_I(S, 1, l)}$$

ce qui, cumulé sur tout le corpus, permet d'avoir de nouvelles estimations des probabilités des règles.

Les probabilités *Inside* et *Outside* correspondent aux étapes de l'analyse montrées dans la figure suivante:



L'algorithme CYK peut servir à calculer ces probabilités. Une description détaillée en est donnée dans (Jelinek et al., 1992). Le même principe "*Inside-Outside*" peut également s'appliquer à l'algorithme d'Earley (Stolcke, 1993).

VII. Travaux connexes

Les différents algorithmes utiles pour l'utilisation des grammaires probabilistes ne constituent qu'une facette du problème. De nombreux travaux examinent d'autres aspects de ces grammaires. Nous donnons ici, sans prétention d'être exhaustif, quelques références à des points de vue différents.

Analyse LR probabiliste

Une variante probabiliste de l'analyse LR a été proposée par (Wright, 1990). Elle ajoute des probabilités conditionnelles aux espaces d'états définis par l'algorithme. La préparation des tables d'analyse est plus complexe, en particulier parce que un même état non-probabiliste peuvent générer une infinité d'états probabilistes, si les probabilités associées peuvent prendre une infinité de valeurs. Pour remédier à ce problème, Wright suggère de fusionner les états probabilistes dont les probabilités diffèrent peu. L'algorithme LR, comme celui d'Earley, calcule facilement les probabilités de préfixe.

Utilisation de grands corpus

L'approche probabiliste va de pair avec un apprentissage sur de grands volumes de texte. Les travaux d'IBM/Lancaster (Black et al., 1993) contiennent une chaîne complète d'expérimentations, depuis la construction d'un corpus de phrases analysées, son utilisation pour le développement d'une grammaire complète de l'Anglais, et des considérations sur l'évaluation des performances d'analyses des grammaires. La construction de grands corpus est par ailleurs l'objet de grands travaux à l'Université de Pennsylvanie (Marcus et al., 1993).

Les grammaires probabilistes ont également été utilisées dans d'autres expérimentations, comme (Derouault and Merialdo, 1985), (Fujisaki et al., 1989), (Sharman et al., 1990).

Autres types de grammaires

(Briscoe and Carroll, 1993) reprennent et étendent l'analyse LR probabiliste. En affectant des probabilités directement à l'intérieur des tables d'analyse (au lieu des règles de la grammaire initiale), ils introduisent une certaine dépendance du contexte sur ces probabilités.

VIII. Conclusions

Cette présentation a montré les principaux algorithmes pouvant s'appliquer aux différentes utilisations des grammaires probabilistes, au travers de variantes probabilistes des algorithmes classiques. Devant l'intérêt croissant suscité par les méthodes statistiques, ces algorithmes devraient avoir une importance grandissante en traitement automatique du langage naturel en permettant d'utiliser des modèles plus puissants que les automates à états finis.

Références

Baker, J. K. (1979). Trainable grammars for speech recognition. In Wolf, D. K. J., editor, *Speech Communication Papers for the 97th Meeting of the ASA*, pages 547–550.

Baum, L., Petrie, T., Soules, G., and Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *Annals of Mathematical Statistics*, 41(1):164–171.

Black, E., Garside, R., and Leech, G. (1993). *Statistically-driven computer grammars of English: the IBM/Lancaster approach*. Rodopi, Atlanta, Ga.

Briscoe, T. and Carroll, J. (1993). Generalized probabilistic LR parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, 19(1):25–59.

Charniak, E. (1993). *Statistical Language learning*. MIT Press.

Derouault, A.-M. and Merialdo, B. (1985). Probabilistic grammar for phonetic to french transcription. *ICASSP*, 4:1577.

Fujisaki, T., Jelinek, F., Cocke, J., Black, E., and Nishino, T. (1989). A probabilistic method for sentence desambiguation. *Proceedings of the 1st International Workshop on Parsing Technologies*, pages 105–114.

Jelinek, F. and Lafferty, J. D. (1991). Computation of the probability of initial substring generation. *Computational Linguistics*, 17(3):315–323.

Jelinek, F., Lafferty, J. D., and Mercer, R. (1992). Basic methods of probabilistic context free grammars. In *NATO Advanced Studies Institute*, Italy.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313.

Sharman, R., Jelinek, F., and Mercer, R. (1990). Generating a grammar for statistical training. *DARPA Speech and Natural Language workshop*, pages 267–274.

Stolcke, A. (1993). An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. Technical Report TR-93-065, International Computer Science Institute, Berkeley, CA.

Wright, J. H. (1990). LR parsing of probabilistic grammars with input uncertainty for speech recognition. *Computer Speech and Language*, (4):297–323.