Analyse syntaxique non déterministe utilisant un modèle de n-grams

Christine Sinoquet, Jacques Nicolas

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Analyse syntaxique non déterministe utilisant un modèle de n-grams

Christine Sinoquet - Jacques Nicolas

N° 3262

septembre 1997

THÈME 3

ISSN 0249-6399

Analyse syntaxique non déterministe utilisant un modèle de n-grams

Christine Sinoquet - Jacques Nicolas *

Thème 3 — Interaction homme-machine,
images, données, connaissances
Projet Repco

Rapport de recherche n 3262 — septembre 1997 — 43 pages

Résumé : Nous étudions le problème de l'analyse non déterministe de séquences. Dans l'objectif de produire les meilleures solutions, nous proposons un modèle de couplage entre analyseur syntaxique et analyseur statistique.

Du point de vue syntaxique, nous nous inspirons du formalisme SVG (String Variable Grammars) proposé par D.Searls. Ce dernier introduit un nouveau type d'objet en analyse syntaxique, les variables de type chaîne, et définit une classe de transformations non déterministes (substitutions) applicables à ces variables. Du point de vue statistique, nous utilisons un modèle de n-grams.

Nous appliquons le principe de recherche des n meilleures solutions au problème de la traduction *reverse* d'une séquence protéique. Nos expérimentations sont menées sur le génome de *Escherichia coli*. Nous évaluons d'abord l'impact de l'augmentation de la taille du contexte pris en compte pour le choix d'un codon, puis la qualité de prédiction obtenue sur une sous-séquence la plus déterminée de la séquence protéique à traduire. Nous examinons également la qualité de prédiction obtenue lorsque les modèles sont calculés sur un ensemble de protéines homologues à la protéine testée. Nous complétons nos investigations par la mise en œuvre de deux protocoles expérimentaux destinés à comparer les qualités de prédiction obtenues selon deux modalités différentes (prise en compte du contexte, choix aléatoire).

Mots-clé : analyse syntaxique,n-grams, symbolique numérique, grammaires logiques, morphisme, apprentissage, SVG, traduction reverse, protéines, ADN

(Abstract: pto)

* sinoquet@irisa.fr jnicolas.@irisa.fr

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : 02 99 84 71 00 - International : +33 2 99 84 71 00
Télécopie : 02 99 84 71 71 - International : +33 2 99 84 71 71

Non-deterministic syntactical analysis with n-grams model

Abstract: We study non-deterministic sequences parsing. We propose a model coupling both a syntactical parser and a statistical analyser, in order to select the best solutions. With regard to syntactical aspect, we base our work on D.Searls'String Variable Grammars formalism (SVG). This formalism introduces new objects in syntactical parsing, string variables, and defines a class of suitable transformations which consist in the specification of left-to-right (vs opposite) variables processing on the one hand and a non-deterministic morphism (substitution) on the other hand. To guide the choice of best solutions, a statistical analyser grounded on a n-grams model classifies the alternative solutions. This model lays on the hypothesis that the probability to observe a given p-uplet, given the $n-1$ preceding p-uplets, is estimated with the relative frequence of this p-uplet, computed from a learning set. As an application, we deal with the problem of reverse translation of proteic sequences. We experiment on *Escherichia coli* genome, implementing dynamic programming. First we evaluate whether the increase of the context size is significant or not in codons prediction quality. Then we focus on reverse translation of a most deterministic sub-sequence of the protein. We also evaluate the prediction quality obtained with an n-gram model computed from a special learning set : homologous proteins with the protein to be translated. Finally, we complete our investigations with an experimental protocol aiming at comparison of prediction qualities obtained from two approaches : the first one relies on the hypothesis of a natural translation driven by context knowledge, the second one consists in random codons choice, the only concern being respect of genomic codons table ratios. In all cases, prediction quality remains relatively modest, which shows predominance of other factors than context in evolution processes.

Key-words: syntactical analysis, n-grams, numerical symbolics, logical grammars, morphism, learning, SVG, reverse translation, proteins, DNA .

Table des matières

Table des figures

Liste des tableaux

1 Introduction

Nous nous intéressons à l'analyse syntaxique de séquences à l'aide de grammaires logiques. Nous supposons le lecteur familier avec les notions de base de la programmation logique, et en particulier avec Prolog [CKVC83]. La gestion du non-déterminisme suppose la définition de mécanismes permettant d'explorer les différentes alternatives d'analyse. Lorsqu'il existe de nombreuses alternatives, on doit introduire des contraintes supplémentaires qui vont permettre de trier et filtrer ces solutions. Si ces contraintes ne sont pas disponibles, on peut essayer d'en inférer sur des corpus d'apprentissage où les "bonnes" analyses sont connues. Dans le cadre de notre étude, le corpus d'apprentissage va permettre de calculer une fonction de pondération, reflet de la qualité des différentes analyses possibles.

Plus particulièrement, nous construisons un analyseur de séquences, basé à la fois sur une analyse syntaxique non déterministe et sur une analyse statistique de corpus, et spécialisé dans la recherche des n meilleures solutions. Pour la partie syntaxique, nous utilisons un formalisme particulier, les grammaires à variables de type chaîne, grammaires dérivées des DCG [PW80]. Pour la partie statistique, nous faisons l'hypothèse que la pondération associée à l'une des alternatives d'analyse reflète le contexte local d'analyse. Cette hypothèse est en général vérifiée pour des domaines "spécialisés"où la distribution des diverses alternatives ne suit pas une loi uniforme. La pondération représente alors la probabilité d'occurrence de la séquence pour laquelle elle a été calculée. Le calcul des pondérations peut être effectué sur la base des connaissances compilées à partir d'un corpus de séquences connues du domaine. Nous utilisons ici le modèle statistique le plus simple pour prendre en compte le contexte d'une analyse : le modèle des n-grams.

La présente étude a donné lieu à une application dans le domaine des séquences biologiques. Le problème traité est celui de la traduction reverse d'une séquence protéique, problème de nature non-déterministe. En effet, étant donnée une séquence protéique, il existe de nombreuses séquences d'acides nucléiques qui en sont toutes une possibilité de traduction reverse. Il s'agit d'identifier les séquences les plus pertinentes, sur la base d'un corpus de séquences réellement observées dans le génome étudié.

La section 2 présente d'abord le formalisme des grammaires à variables de type chaîne, proposé par D. Searls (SVG, String Variable Grammar). Nos travaux consistent à incorporer à ce formalisme un mécanisme de prise en compte du contexte local d'analyse, destiné à guider l'analyse vers le meilleur choix. Après le rappel relatif au formalisme SVG, nous étudions le couplage d'un analyseur syntaxique et d'un analyseur statistique fondé sur un

modèle de n-grams. Nous posons alors les fondements théoriques de ce couplage. La section 3 présente la manière de mettre en œuvre l'analyseur non- déterministe avec modèle de n-grams à l'aide d'une technique de programmation dynamique. La section 4 présente enfin une application à la traduction reverse d'une séquence protéique.

2 Couplage formalisme SVG-modèle de n-grams

2.1 Formalisme SVG

Confronté à l'objectif d'étendre les grammaires hors-contexte [Cho55] de façon à rendre compte des dépendances intramoléculaires présentées par les séquences biologiques (acides nucléiques et protéines, en l'occurrence), D. searls s'est inspiré des DCG [PW80]. Sa contribution consiste en l'introduction de variables, au sens logique du terme, au côté des termes terminaux et non terminaux classiques [Sea89] [Sea93a] [Sea93b]. En cours d'analyse, la rencontre de la première occurrence d'une variable donnée déclenche le renseignement (l'instanciation) de cette variable. Lors de la rencontre de toute autre occurrence de cette variable, il y a confrontation entre le contenu de cette variable et le reste non encore analysé de la séquence. Ainsi un tel formalisme permet-il l'écriture de règles aussi concises que $tandem\_copies \rightarrow XX$, par exemple, ce qui représente une simplification de spécification par rapport à un autre formalisme hors-contexte étendu, utilisé pour décrire des références croisées : les grammaires indexées. Plus généralement, une règle SVG peut utiliser un symbole de type ${}^{s}f(X)$ où la transformation ${}^{s}f$ appliquée à la variable X met en œuvre un morphisme f (ou substitution élémentaire) de la manière suivante :

f, substitution élémentaire, associe à tout caractère d'un alphabet $\mathcal{A}_1$ un ensemble de chaînes de caractères formées sur un alphabet $\mathcal{A}_2$. Si, pour chaque élément de $\mathcal{A}_1$, l'ensemble des images par f est réduit à un singleton, f est déterministe.

Le symbole s indique la façon dont est générée toute image morphique ${}^{s}f(X)$, à partir de la chaîne X.

Ainsi, ${}^{+}f$ désigne une transformation directe:

$$\forall a \in \mathcal{A}_1 \quad \forall \omega \in \mathcal{A}_1^* \quad {}^{+}f(a.\omega) = f(a).{}^{+}f(\omega).$$

En revanche, ${}^{-}f$ désigne une transformation inverse:

$$\forall a \in \mathcal{A}_1 \quad \forall \omega \in \mathcal{A}_1^* \quad {}^{-}f(a.\omega) = {}^{-}f(\omega).f(a).$$

Le point désigne la concaténation entre chaînes.

Dans le cas de l'application identité de $\mathcal{A}_1$ dans $\mathcal{A}_1$, ${}^{+}id(X)$ spécifie la répétition d'un motif X [1], ${}^{-}id(X)$ désigne la séquence inverse de X.

Selon les besoins, on peut devoir modéliser une association de motifs au moyen d'une grammaire SVG ($tandem\_copies \rightarrow {}^{+}id(X)$), ${}^{+}id(X)$ permettra d'analyser *abab* par exemple; $palindrome \rightarrow {}^{+}id(X), {}^{-}id(X)$ sera le modèle utilisé pour l'analyse de *abba*, ...). Mais dans d'autres cas, pour X fixée, on peut vouloir rechercher les meilleures solutions ${}^{s}f(X)$. C'est cette génération des meilleures solutions engendrées par un analyseur non-déterministe qui constitue la motivation de nos travaux : il s'agit de guider au mieux le choix des possibilités de substitution.

2.2 Fondements théoriques du couplage analyseur syntaxique-analyseur statistique

Un exemple illustre d'abord le principe que nous souhaitons mettre en œuvre pour guider un analyseur non-déterministe à l'aide d'un modèle de n-grams [Jel90] [Jel83]. Nous établissons ensuite les bases du calcul du poids associé à une solution.

Supposons par exemple qu'un modèle de n-grams compile les informations relatives aux couples de caractères xy, soit les fréquences d'utilisation de $f(x)$, $f(y)$, $f(x).f(y)$ dans les séquences du langage disponibles. Nous supposons pour simplifier que toutes les chaînes $f(x)$ formées à partir du caractère x ont la même longueur, et ce quel que soit le caractère x. Nous appelons n cette constante. Pour tout mot $x.y.z.t.\omega$, nous avons : ${}^{+}f(x.y.z.t.\omega) = f_2(x.y).f_y(z).f_z(t).{}^{+}f(\omega)$. La notation $f_y(z)$ indique que l'ordre de qualité des choix de z dépend des choix effectués sur y et rappelle que $f_2(y.z)$ est pris en compte pour déterminer $f_y(z)$.

De façon plus générale, on peut considérer que la substitution f permet d'obtenir, à partir d'un mot m formé sur un alphabet $\mathcal{A}_1$, un certain nombre de mots à l'aide d'un alphabet $\mathcal{A}_2$. Soit m un mot formé sur $\mathcal{A}_1$ comportant p caractères. $m_{i,l}$ désigne la sous-chaîne de m débutant au caractère i (compris) et comportant l caractères. Considérons $s = {}^{\pm}f(m)$. s comporte $n \times p$ caractères. Par substitution, à chaque caractère de m correspond une séquence de n caractères de s. Appelons n-uplet une telle séquence. On note $s_{i,l}$ la sous-chaîne de la chaîne s débutant par le n-uplet numéro i, et de longueur $n \times l$. Par exemple, avec cette notation, le n-uplet numéro 1 du mot s est noté $s_{1,1}$.

1. Dans ce formalisme, la spécification X correspond à la spécification sous-jacente ${}^{+}id(X)$.

Le processus d'analyse que nous proposons met en jeu un paramètre FR, facteur de regroupement. Pour simplifier l'exposé, nous raisonnons dans un premier temps comme si nous ne souhaitions identifier qu'une unique solution s. Le principe consiste à progresser caractère par caractère, dans le mot m, et à retenir la substitution $f(c)$ la plus pertinente pour le caractère courant c en tenant compte de la substitution ${}^{\pm}f(u)$ opérée pour les $FR-1$ caractères précédents (séquence u). En définitive, bien que la progression opère caractère par caractère, à chaque étape, est examinée la substitution ${}^{\pm}f(u.c)$ d'une séquence de FR caractères $(u.c)$,${}^{\pm}f(u)$ étant déjà figée. Formellement, nous posons ${}^{\pm}f(u.c) = {}^{\pm}f(u).f(c) = f_{FR,u}(c)$.En ce sens, nous utilisons un modèle de FR-n-grams.

Notons alors qu'à une progression dans m de caractère en caractère se superpose une progression par sous-mots de FR caractères avec chevauchement entre ces sous-mots (sur une longueur de $FR-1$ caractères). De façon corollaire, il y a chevauchement entre les deux FR-nuplets[2] successifs des étapes i et $i+1$, chacun de ces FR-$n$uplets étant terminé par le n-uplet traité aux étapes i et $i+1$ respectivement. Plus particulièrement, pour un facteur de regroupement égal à 1, aucun contexte n'est pris en compte.

Considérons M et S deux variables aléatoires à valeurs dans les vocabulaires des mots de p caractères de $\mathcal{A}_1$ et des mots de $n \times p$ caractères de $\mathcal{A}_2$. De la même façon que nous avions défini $m_{i,l}$ et $s_{i,l}$, nous définissons une relation similaire de projection entre les variables M et $M_{i,l}$ d'une part, S et $S_{i,l}$.

Notons $E_{i,l}$ l'événement $(S_{i,l} = s_{i,l})/(M_{i,l} = m_{i,l})$.

L'hypothèse de prise en compte du contexte s'exprime par la relation suivante, où pour le choix du n-uplet i $(i > FR)$, il est tenu compte des $FR-1$ n-uplets précédemment choisis:

$(S = s/M = m) = (E_{1,FR} \cap_{i=FR+1}^{p} (E_{i,1}/E_{i-FR+1,FR-1})$.

$E_{i,1}$ fait référence à la transformation de la sous-chaîne de m débutant au caractère numéro i, et de longueur 1, c'est-à-dire que $E_{i,1}$ concerne le caractère numéro i de m, et de façon corollaire le n-uplet numéro i de s. ($E_{i-FR+1,FR-1}$ a trait à la transformation de la sous-chaîne de m, de longueur $FR-1$, et précédant le caractère numéro i.).

Avec l'hypothèse d'indépendance sur les événements considérés, une probabilité est calculée comme suit:

$$p(S = s/M = m) = p(E_{1,FR}) \ \ \Pi_{i=FR+1}^{p} \ \ p(E_{i,1}/E_{i-FR+1,FR-1}).$$

2. Un FR-$n$uplet est une séquence de $FR \times n$ caractères de $\mathcal{A}_2$. Intuitivement, c'est le résultat de la transformation opérée à partir d'un sous-mot de FR caractères de $\mathcal{A}_1$.

Or

$$p(E_{i,1}/E_{i-FR+1,FR-1}) = \frac{p(E_{i,1} \cap E_{i-FR+1,FR-1})}{p(E_{i-FR+1,FR-1})}.$$

Rappelons que $E_{i,1}$ concerne le n-uplet numéro i et $E_{i-FR+1,FR-1}$ concerne les $FR-1$ n-uplets précédents.

$E_{i,1} \cap E_{i-FR+1,FR-1}$ n'est autre que $E_{i-FR+1,FR}$.

Donc,

$$p(S = s/M = m) = p(E_{1,FR}) \ \ \Pi_{i=FR+1}^{n} \ \ \frac{p(E_{i-FR+1,FR})}{p(E_{i-FR+1,FR-1})}.$$

3 Construction des n meilleures solutions

De manière à produire un algorithme efficace, nous adoptons la technique de la programmation dynamique au problème du guidage d'un analyseur non-déterministe par un modèle de n-grams. Dans cette section, nous donnons la méthode de parcours dirigé, et en largeur d'abord, de l'arbre des solutions. Nous illustrons ensuite cette méthode à l'aide d'un exemple.

3.1 Programmation dynamique

Pour calculer les X meilleures solutions de longueur $n \times p$ caractères (la séquence à traduire étant de longueur p), nous appliquons la récurrence suivante, et ce pour chaque possibilité de traduction du $i$ème caractère (en une séquence de $n \times i$ caractères $s_{i,1}$ selon les notations de la section 2):

$\mathcal{E}_{i,s_{i,1}} = Max_X(e_{i-1} + score_{i,s_{i,1}}, e_{i-1} \in \mathcal{E}_{i-1}), i > FR$

$\mathcal{E}_{FR,s_{1,FR}} = Max_X(score_{FR,s_{1,FR}})$.

L'étape $i (i > FR)$ est celle de la traduction du $i$ème caractère en une séquence de $n \times i$ caractères. La première étape concerne la traduction "en bloc" des FR premiers caractères. $score_{i,s_{i,1}}$ traduit la prise en compte du contexte dans la traduction du $i$ème caractère, la traduction des $i-1$ premiers caractères étant établie ($\mathcal{E}_{i-1}$).

En utilisant les notations et les bases théoriques de la section 2, nous considérons $s_{i-FR+1,FR}$ séquence de longueur $n \times FR$ caractères se terminant par $s_{i,1}$. Nous considérons également $s_{i-FR+1,FR-1}$ séquence de longueur $n \times (FR-1)$ caractères, suffixe de $s_{1,i-1}$. Nous exprimons alors:

$score_{i,s_{i,1}} = g(n_2(p((E_{i-FR+1,FR})), n_2(p(E_{i-FR+1,FR-1})),$

$score_{FR,s_{1,FR}} = n_2(p(E_{1,FR}))$,

n_2 étant une fonction de $[0,1]$ dans $\mathcal{R}$ et g une fonction de $\mathcal{R} \times \mathcal{R}$ dans $\mathcal{R}$ (n_2 et g définissent une normalisation.).

Le modèle de n-grams recourt aux informations concernant les FR-$n$uplets d'une part, et les $FR-1$-$n$uplets d'autre part. Il utilise les transformations pondérées f_{FR} et f_{FR-1}. Une transformation pondérée f_l est définie de ${\mathcal{A}_1}^l$ dans ${\mathcal{A}_2}^{l \times n} \times \mathcal{R}$; elle associe à un sous-mot $m_{i,l}$ un couple $s_{i,l}, n_2(p(E_{i,l}))$.

3.2 Exemple

Nous présentons maintenant les trois étapes de l'algorithme à l'aide d'un court exemple établi sur des données réelles (transformation du mot IEK, pour $FR = 2$ (facteur de regroupement), $p = 3$ (longueur du mot), $n = 3$ (longueur de substitution) et $X = 2$ (nombre de solutions retenues) et la table d'usage des sextuplets presentée en tableau 1 ($\mathcal{A}_1 = \{I, E, K\}$ et $\mathcal{A}_2 = \{a, c, g, t\}$).

IE	att gaa	-0.933468
	atc gaa	-1.267814
	att gag	-1.759712
	atc gag	-2.412037
	ata gaa	-3.125387
	ata gag	-3.926165
EK	gaa aaa	-0.663344
	gag aaa	-1.351609
	gaa aag	-2.073298
	gag aag	-2.299788

TAB. 1 – *Table d'usage des sextuplets pour la traduction des sous-mots IE et EK*

Les valeurs négatives proviennent de la normalisation: sont en effet considérés les logarithmes de probabilités (n_2 est la fonction logarithme et g est l'addition).

La première étape du processus est initiée de la manière suivante: la substitution de IE offre 2 possibilités de traduction (*gaa* ou *gag*) pour le caractère E (triplet numéro 2). Pour chacune de ces possibilités, sont retenues les 2 meilleures solutions (poids P_i):

triplet 2 = gaa
attgaa − 0.933468
atcgaa − 1.267814

triplet 2 = gag
attgag − 1.759712
atc gag -2.412037

À l'étape suivante (étape), il y a 2 possibilités de traduction (*aaa* ou *aag*) pour K (triplet numéro 3). Pour chacune d'entre elles, sont retenues de la même façon les 2 meilleures solutions. triplet 3 = aaa
attgaaaaa -1.2291 *attgaaaaa*
atcgaaaaa -1.5634 *atcgaaaaa*
attgagaaa -1.9327
atcgagaaa -2.5850

triplet 3 = aag
attgaaaag -2.6390 *attgaaaag*
atcgaaaag -2.9734
attgagaag -2.8809 *attgagaag*
atcgagaag -3.5332

À l'étape finale, parmi toutes les traductions obtenues (*attgaaaaa*, *atcgaaaaa*, *attgaaaag*, *attgagaag*), les 2 meilleures solutions sont: *attgaaaaa* (-1.2291) et *atcgaaaaa* (-1.5634).

4 Application à la traduction reverse d'une séquence protéique

Nous précisons tout d'abord en quoi consiste la traduction reverse. Nous montrons ensuite comment un analyseur non-déterministe utilisant un modèle de n-grams peut être utilisé afin de produire les n meilleures "sondes nucléiques", résultats de la traduction reverse d'une séquence protéique donnée. Puis, nous présentons les principaux composants de l'analyseur implémenté et donnons le calcul de la complexité de l'algorithme implémenté en Prolog. Nous terminons cette section en exposant les résultats obtenus.

4.1 Traduction reverse

L'ADN contient les informations codifiant la synthèse des protéines.Une molécule d'ADN est une séquence de nucléotides. Un nucléotide est caractérisé en particulier par la nature de la base qu'il contient. Le langage de l'ADN est-il codé au moyen d'un alphabet à 4 lettres que sont a,c,g et t. Par ailleurs, une molécule protéique est une séquence d'acides aminés. Le langage des protéines est codé à l'aide de 20 caractères correspondant aux 20 acides aminés existants.Il existe une correspondance entre bases et acides aminés, qui permet la traduction déterministe d'une séquence nucléique vers une séquence protéique. Cette correspondance, appelée code génétique, associe un codon à un acide aminé. Un codon correspond à 3 bases. Sur les 64 codons possibles, 61 sont utilisés pour le codage d'un acide aminé. Comme il existe seulement 20 acides aminés, le code génétique est dégénéré, c'est-à-dire qu'il existe des codons synonymes. Le tableau 8 présente le code génétique pour *Escherichia coli*, qui est un code à peu près universel.

La conséquence de la dégénérescence du code génétique est que la traduction reverse (d'une séquence protéique vers une séquence d'ADN) est une opération non-déterministe. Nous cherchons à rendre ce non-déterminisme le plus faible possible en utilisant les spécificités d'usage des codons par rapport à leur contexte. Les applications biologiques concernées sont la production de sondes d'ADN et la recherche de gènes dans des banques.

4.2 Adéquation de l'algorithme précédent à la production de sondes nucléiques

Nous avons signalé le non déterminisme de la traduction reverse. Le nombre de codons synonymes que peut présenter un acide aminé est 1,2,3,4 ou 6. Si l'on adopte une moyenne de 4 synonymes par acide aminé, une séquence protéique de 10 acides aminés admet 4^{10}, soit environ un million de séquences codantes différentes. Une amélioration peut consister à choisir une fraction de la protéine étudiée qui comporte des acides aminés codés par un unique codon. Mais ces acides aminés sont rares (Méthionine (M), Tryptophane (W)). Par ailleurs, la sonde doit être spécifique, c'est-à-dire qu'elle ne doit s'hybrider qu'avec les séquences nucléiques cibles.

Pour choisir, parmi le nombre de combinaisons codantes, celles qui constitueront les meilleures sondes, on utilise la table d'usage des codons, propre à une espèce donnée. Pour un acide aminé donné, cette table indique la fréquence d'utilisation de chacun des codons synonymes, dans le génome.

acide aminé	**codons synonymes**
M	atg
W	tgg
C	tgc, tgt
D	gat, gac
E	gaa, gag
F	ttt, ttc
H	cat, cac
K	aaa, aag
N	aac, aat
Q	cag, caa
Y	tat, tac
I	att, atc, ata
A	gcg, gcc, gca, gct
G	ggc, ggt, ggg, gga
P	ccg, cca, cct, ccc
T	acc, acg, act, aca
V	gtg, gtt, gtc, gta
L	ctg, ttg, tta, ctc, ctt, cta
R	cgc, cgt, cgg, cga, aga, agg
S	agc, tcc, tct, tcg, agt, tca

TAB. 2 – *Code génétique pour Escherichia coli*

Le choix d'un formalisme inspiré des SVGs guidé par un modèle de n-grams, destiné à la production des n meilleures sondes nucléiques est justifié par les arguments suivants :

- Un problème de traduction reverse revient, au niveau élémentaire (substitution f), à associer un codon à un acide aminé. Le code génétique est naturellement traductible à l'aide d'une grammaire modélisant f. Il est ensuite possible de construire un mécanisme de progression dans une séquence protéique X, qui permette la génération simultanée de ${}^{+}f(X)$, séquence de codons - ou 3-uplets -codant pour X.
- Le mécanisme Prolog sous-jacent, non-déterministe, permet d'obtenir l'ensemble des séquences codantes ${}^{+}f(X)$ pour la séquence X.
- le modèle de n-grams met en jeu :
  - les alphabets $\mathcal{A}_1 = \{A, C, D, \ldots\}$ (ensemble des 20 symboles représentant les acides aminés) et $\mathcal{A}_2 = \{a, c, g, t\}$ (ensemble des 4 bases),
  - le paramètre n égal à 3 (À un acide aminé correspond un 3-uplet ou codon.),
  - les transformations pondérées f_{FR} et f_{FR-1}, qui compilent les informations sur la contexte (à gauche) des codons. Elles correspondent aux tables d'usage des FR-$n$uplets et $FR-1$-$n$uplets, c'est-à-dire à la traduction des séquences de FR et $FR-1$ acides aminés respectivement. Le cas particulier où FR est égal à 1 revient à la seule table d'usage des codons, ce qui revient à ne prendre en compte aucun chevauchement.

4.3 Présentation de l'application

Réalisée en Prolog, l'application se compose de 4 principaux modules:

- un générateur automatique de tables d'usage des nuplets (grammaires probabilisées),
- un générateur d'analyseurs spécifiés au moyen de grammaires SVG,
- un module spécialisé dans la production des meilleures traductions reverse,
- un module d'édition des résultats.

4.3.1 Générateur automatique de tables d'usage

L'application utilise deux paramètres, l'identificateur commun aux fichiers comportant des informations sur les locci des gènes (ECcontigs.born) et les contigs (ECcontigs.seq), et le facteur de regroupement (FR), qui dirige le recensement des séquences de FR-*n*uplets dans chacun des gènes. Le programme vérifie la validité des gènes (Un gène valide débute par atg, gtg ou ttg.), lit les deux sens du brin d'ADN et restitue le brin complémentaire s'il y a lieu. Il accepte la présence de bases inconnues (codées par n ou x) dans les séquences.

Le générateur de tables d'usage procède en trois étapes:

- calcul de la fréquence[3] d'apparition de chacun des 64^{FR} FR-nuplets[4],
- production des règles de la grammaire (dictionnaire) par fréquences décroissantes (préparation à l'élagage de l'arbre des combinaisons possibles),
- normalisation des fréquences.

La normalisation nous conduit à raisonner en termes de probabilités (conditionnelles) (division de la fréquence d'un synonyme donné par la somme des fréquences de tous les synonymes).

En adaptant le vocabulaire utilisé en section 2, c'est-à-dire en considérant les mots $prot$, adn (au lieu de m, s) et les variables aléatoires $PROT$, ADN (au lieu de M, S), on obtient la formule de calcul de poids suivante : fournie pour $E_{i,l}$, nous écrivons :

$$p(ADN = adn/PROT = prot) = p(E_{1,FR}) \;\; \Pi_{i=FR+1}^{p} \;\; \frac{p(E_{i-FR+1,FR})}{p(E_{i-FR+1,FR-1})}.$$

Une normalisation ultérieure allègera le calcul du poids d'une solution. En considérant $N(E_{i,l}) = log(p(E_{i,l}))$, nous écrivons plus simplement:

3. La fréquence d'un nuplet est le nombre d'occurrences de ce nuplet divisé par la somme des nombres d'occurrences de tous les nuplets, dans la fraction codante du génome.

4. La cohérence entre le module de traduction reverse et le générateur de tables d'usage est assurée par le fait que le contexte est pris en compte de manière identique dans les deux modules. Ainsi, avec un facteur de regroupement FR, une occurrence de codon sera recensée FR fois dans le contexte de FR séquences de FR codons, respectivement en positions numéros FR, $FR-1$, ... ,1 de chacune des séquences respectives. À ce stade, une structure de données provisoire (dictionnaire des triplets, sextuplets ...) permet de stocker les informations.

$$N(ADN = adn/PROT = prot) = N(E_{1,FR}) \ +\Sigma_{i=FR+1}^{p} \ N(E_{i-FR+1,FR}) - N(E_{i-FR+1,FR-1}).$$

Cette formule, relative à des $n$uplets de longueurs FR et $FR-1$ respectivement, impose l'utilisation conjointe de deux dictionnaires. Par exemple, un facteur de regroupement égal à 3 nécessite l'utilisation des tables d'usage des 9-uplets d'une part, des sextuplets d'autre part.

4.3.2 Générateur d'analyseurs associés à des grammaires SVG

Nous mentionnions *supra* que le langage Prolog est particulièrement bien adapté à la spécification de grammaires, donc à l'analyse syntaxique. Il est possible d'écrire directement en Prolog l'analyseur correspondant à une grammaire donnée. Les manipulations sur la chaîne en cours d'analyse sont alors gérées par le programmeur (progression dans la chaîne à la faveur de la satisfaction d'une règle de la grammaire). La plupart des versions de Prolog comportent un traducteur permettant de générer, à partir d'une grammaire DCG[5] [PW80], l'ensemble des règles Prolog implémentant l'analyseur associé à cette grammaire. Nous avons ainsi obtenu dynamiquement un analyseur adapté à une grammaire SVG donnée.

Nous avons donc écrit un traducteur de grammaires SVG en règles Prolog. Ce traducteur étant général, nous sommes aussi en mesure de confirmer ou d'infirmer la présence d'un tandem de copies, ou d'un palindrome biologique, par exemple, au sein d'une molécule biologique. Dans le cas de la génération automatique des 10 meilleures sondes pour une séquence protéique donnée, il a cependant fallu spécialiser l'algorithme général:

- premièrement, pour traiter la prise en compte du contexte (les $FR-1$ codons précédemment choisis) dans le choix du codon courant,
- ensuite, pour progresser de front dans l'arbre des solutions en cours de construction (programmation dynamique), selon les principes théoriques exposés *supra*.

4.3.3 Complexité

Le mot m dont il s'agit de fournir les X meilleures substitutions comporte p caractères. L'analyse est réalisée caractère par caractère, avec mémorisation d'un contexte (gauche)

5. Definite Clause Grammar

de $FR-1$ caractères. Ainsi, le processus d'analyse joue du caractère $FR+1$ au caractère p(p-FR-2 étapes intermédiaires, selon le terme utilisé lors de la présentation de l'exemple).

Par définition de la substitution élémentaire f, à chaque caractère de $\mathcal{A}_1$ est associé un ensemble de chaînes de caractères de $\mathcal{A}_2$, chacune de longueur m. Si N_1 et N_2 sont respectivement les cardinaux de $\mathcal{A}_1$ et $\mathcal{A}_2$, pour un mot de FR caractères, il existe $(\frac{N_2^n}{N_1})^{FR}$ substitutions possibles en moyenne.

Lors de l'étape initiale, l'ensemble des substitutions (associées à un poids) correspondant au mot $m_{1,FR}$ est identifié. Chaque substitution est rangée dans une liste indexée par clé. Chaque substitution est donc insérée dans une sous-liste dont tous les éléments (des substitutions) sont caractérisés par l'identité de leurs n-uplets suffixes (cette valeur commune est la clé). L'indexation par clé permet un accès à la sous-liste adéquate en $O(log(\frac{N_2^n}{N_1}))$ (Il y a $\frac{N_2^n}{N_1}$ codages sur $\mathcal{A}_2$ pour un caractère de $\mathcal{A}_1$ en moyenne). Chaque sous-liste est triée par ordre croissant des poids associés aux substitutions. Lorsque le cardinal d'une sous-liste de clé K est strictement inférieur à X, toute substitution de suffixe K est insérée dans la sous-liste. Dès que ce cardinal est égal à X, une substitution de suffixe K est soit rejetée parce que non améliorante, soit insérée dans la sous-liste après que la tête de liste ait été éliminée. Le traitement d'une substitution nécessite donc un accès indexé ($O(log(\frac{N_2^n}{N_1}))$ et au pire un parcours de longueur X dans la sous-liste adéquate. La complexité maximale de l'étape initiale est donc $(0(log(\frac{{N_2}^n}{N_1}) + X) \times (\frac{N_2^n}{N_1})^{FR}$.

A l'étape intermédiaire numéro $i(i > FR)$, nous disposons alors d'une liste L_{i-1} indexée de substitutions $s_{1,i-1}$ organisée en sous-listes de substitutions à n-uplet suffixe commun.

Pour chacune des $(\frac{N_2^n}{N_1})^{FR}$ possibilités de substitutions $s_{i-FR+1,FR}$, le $FR-1$ n-uplet préfixe C est identifié, ainsi que l'avant-dernier n-uplet K. Il s'agit alors d'accéder à la sous-liste indexée par K, dans la liste L_{i-1}. Parmi les X (au plus) substitutions $s_{1,i-1}$ rangées dans cette sous-liste, seules sont intéressantes celles qui se terminent par le suffixe C. Le chevauchement étant alors possible, des substitutions $s_{1,i}$ peuvent être construites par concaténation des $s_{1,i-1}$ et du dernier n-uplet K' de $s_{i-FR+1,FR}$. Le poids d'une substitution $s_{1,i}$ est la somme du poids de la substitution $s_{1,i-1}$ et d'une fonction des poids de $s_{i-FR+1,FR}$ et $s_{i-FR+1,FR-1}$. Il y a au pire X substitutions $s_{1,i-1}$ présentant un chevauchement de $(FR-1) \times n$ caractères avec une substitution $s_{i-FR+1,FR}$. Pour chaque cas de chevauchement valide, est reconduit le schéma de rejet ou d'insertion déjà évoqué lors de la présentation de l'étape initiale.

Les nouvelles substitutions $s_{1,i}$ sont rangées dans la sous-liste indexée par K' dans la liste $L_i(O(log(\frac{{N_2}^n}{N_1})))$. En définitive, la complexité dans le pire des cas, pour chacune des p-FR-2 étapes intermédiaires, est égale à $(O(log(\frac{N_2^n}{N_1}) + 2X) \times (\frac{{N_2}^n}{N_1})^{FR}$.

L'étape finale consiste à obtenir une liste classique triée par ordre décroissant à partir de la liste indexée par clés, L_p. Une primitive Prolog offre cette possibilité. Il s'agit ensuite de ne retenir que les X premiers éléments de cette liste (L_p comportait au pire $X \times \frac{{N_2}^n}{N_1}$ éléments). La complexité de l'opération est $O(log(\frac{N_2^n}{N_1}))$.

En bilan, la complexité de l'algorithme est évaluée à

$$[(O(log(\frac{N_2^n}{N_1})) + X) + (O(log(\frac{N_2^n}{N_1})) + 2X) \times (p - FR - 2)] \times (\frac{{N_2}^{n\,FR}}{N_1}) + O(log(\frac{{N_2}^n}{N_1}))$$

Si $log(\frac{{N_2}^n}{N_1})$ est petit devant X, l'ordre de grandeur de la complexité est $O(Xp(\frac{{N_2}^n}{N_1})^{FR})$. X et $(\frac{{N_2}^n}{N_1})^{FR}$ étant des constantes, l'algorithme est linéaire en p.

4.4 Résultats obtenus

Nous comparons d'abord les résultats obtenus, pour une même séquence protéique, lorsque nous modifions le facteur de regroupement (FR successivement égal à 1, 2, 3). Ces premiers résultats nous conduisent ensuite à examiner la qualité de prédiction obtenue sur une sous-séquence la plus déterministe de la séquence protéique à traduire. Nous examinons également la qualité de prédiction obtenue lorsque les modèles sont calculés sur un ensemble de protéines homologues à la protéine testée. Nous terminons en comparant systématiquement les qualités de prédiction obtenues selon les deux modalités suivantes : la première modalité adopte l'hypothèse d'un processus naturel tenant compte de la notion de contexte, la seconde modalité consiste à choisir les codons au hasard, avec le seul souci de respecter les proportions fournies par la table d'usage des codons.

4.4.1 Influence du facteur de regroupement

Tester l'application consiste à confronter les 10 meilleures solutions obtenues avec la séquence nucléique (la cible) codant réellement cette séquence protéique. La comparaison des poids d'une sonde et de la cible d'une part, le calcul d'un taux de discordance entre sonde et cible d'autre part sont réalisés. En effet, l'alignement d'une sonde et de la cible

met en évidence les bases discordantes. Le taux de discordance est le rapport du nombre de bases discordantes au nombre total de bases de la sonde (ou de la cible).

Il est vérifié que si l'on réduit le génome à la cible associée à une certaine séquence protéique, cette cible apparaît en première solution. Nous avons réalisé ce test sur la fraction de gène $atgaacaaaaacagagggtttacgcctctg$ codant MNKNRGFTPL (gène *acrA* de *Escherichia coli*). Par la suite, c'est le génome de *Escherichia coli* qui a fourni matière à nos tests. Cependant, pour les tests réalisés sur le génome entier, nous n'attendions pas que la cible apparaisse nécessairement parmi les 10 meilleures solutions. En effet, l'existence naturelle de la cible n'est pas antinomique de la faible fréquence d'apparition de ses composants dans le génome. La sélection de cibles est réalisée au moyen d'un fichier contenant des séquences protéiques et leur code ADN.

Le tableau 3 indique l'ordre de grandeur du taux de discordance (défini *supra*) obtenu successivement avec les valeurs du facteur de regroupement 1, 2 et 3, pour des séquences protéiques de longueur comprise entre 10 et 280. La description des séquences protéiques testées figure en annexe A (*cf* tableau 8). Ceci ne peut être considéré que comme des exemples illustratifs. Une approche plus systématique s'impose, qui doit considérer un plus grand nombre de séquences.

L'amélioration apportée par la prise en compte du contexte est relativement générale (bien que limitée), dès lors que la longueur considérée est assez grande(au-delà de 40). Le gain maximal sur le pourcentage de concordance entre sonde et cible avoisine 5%. C'est plutôt un gain de 1% environ qui est observé pour les séquences de longueur 10 présentant une amélioration. Nous nous attendions à ce que l'impact de la prise en compte du contexte s'exprime nettement avec des séquences relativement longues. De fait, même si la séquence de longueur 40 dément ce principe, ou encore que le pourcentage stagne, avec et sans contexte, pour la séquence de longueur 50, c'est sur les séquences de longueurs 44, 46, 125 et 280 que l'amélioration atteint respectivement 3, 5, 1.5 et 3%. Quant aux séquences de longueur 148 et 273, elles se caractérisaient déjà par un pourcentage de discordance relativement faible (environ 20% et 16%) pour une valeur 1 de FR. Dans ces deux derniers cas, l'amélioration qui a été constatée avec prise en compte du contexte est négligeable. Ce cas contraste avec celui de la $7i$ème séquence de longueur 10 du tableau 3, pour laquelle le taux diminue de 11% à 9% environ.

Pour les séquences de longueur 10, une représentation graphique des discordances entre les 10 sondes solutions et la cible est proposée. Par exemple, pour la traduction reverse de MNKNRGFTPL, codée naturellement par

$$atgaacaaaaacagagggtttacgcctctg,$$

séquence protéique	**taille**	**FR = 1**	**FR = 2**	**FR = 3**
1	10	0.187	0.160	0.163
2	10	0.277	0.267	0.300 -
3	10	0.157	0.153	0.240 -
4	10	0.160	0.160	0.193 -
5	10	0.180	0.233	0.170 +
6	10	0.217	0.237	0.183 +
7	10	0.117	0.090	0.110
8	10	0.290	0.260	0.300 -
9	10	0.257	0.270	0.240 +
10	20	0.227	0.217	0.237 -
11	30	0.238	0.234	0.259 -
12	40	0.328	0.334	0.364 -
13	44	0.237	0.208	0.208
14	46	0.257	0.211	0.242 -
15	50	0.303	0.307	0.337 -
16	125	0.202	0.185	0.175 +
17	148	0.196	0.192	0.187 +
18	273	0.164	0.160	
19	280	0.162	0.130	

TAB. 3 – *Moyenne des taux de discordance pour les 10 meilleures solutions avec ou sans prise en compte du contexte ($FR = 1 FR = 2 FR = 3$) Légende : effet négatif(-), positif(+) de l'augmentation de FR de 2 à 3*

nous générons automatiquement la mise en évidence des discordances entre une solution et la cible (cf tableau 4)(Nous faisons de même pour identifier rapidement la modification d'une solution par rapport à une autre.).

```
                    a a  g       g    t
                    | |  |       |    |
atg  aac  aaa  aac  cgt  ggc  ttt  acc  ccg  ctg
```

TAB. 4 – *Mise en exergue des discordances entre sonde et cible*

C'est ainsi qu'apparaît nettement le fait que les ensembles de solutions sélectionnées avec les valeurs 1 et 2 de FR sont en général disjoints.

Enfin, notons que sur les exemples étudiés, nous avons toujours constaté que le poids de la cible reflétait la faible présence dans le génome de ses composants. Le poids calculé pour une cible a toujours été situé nettement à l'extérieur de l'intervalle des poids des 10 solutions. Encore la notion d'intervalle est-elle toute relative. C'est ainsi que sur l'exemple relatif à la longueur 280, cet intervalle est $[-188.8023, -188.7857]$, alors que le poids de la cible est -272.2381.

D'une manière générale, nous constatons qu'aucune tendance à l'amélioration ni à la dégradation ne peut être décelée, concommitamment à la croissance de FR (de 1 à 3), pour une même séquence donnée. Ainsi, la séquence de taille 46, dont le taux de discordance s'est amélioré (de 5% environ) pour un passage de 1 à 2 de la valeur de FR retrouve cependant un taux voisin du taux calculé pour la valeur 1, lorsque FR vaut 3. En revanche, là où nous observions une dégradation (resp. une amélioration), en passant de la valeur 1 à la valeur 2 de FR, nous constatons que le sens de cette évolution est maintenue au cours du passage de la valeur 1 à la valeur 3. C'est ainsi qu'une amélioration de 1,5 % est encore observée pour la séquence de taille 46, lors du passage de la valeur 1 à la valeur 3 pour FR.

4.4.2 Traduction sur une sous-séquence la plus déterministe

La traduction reverse est limitée à la sous-séquence la plus déterministe de la séquence protéique d'origine. Cette sous-séquence est composée des acides aminés possédant peu de codons synonymes. Le tableau 5 regroupe quelques résultats obtenus à partir de séquences protéiques de diverses longueurs dont est extraite, pour chacune d'entre elles, la sous-séquence la plus déterministe de taille 45.

séquence protéique	**taille**	**FR = 2** séquence intégrale	**FR = 2** séquence déterministe	**impact**	**FR = 3** séquence intégrale	**FR = 3** séquence déterministe	**impact**
15	50	0.307	0.327	-	0.337	0.361	-
16	125	0.185	0.127	+	0.175	0.128	+
17	148	0.192	0.148	+	0.187	0.155	+
18	273	0.160	0.115				
19	280	0.130	0.156	-			

TAB. 5 – *Moyenne des taux de discordance pour les 10 meilleures solutions (FR = 2, FR = 3) pour la sous-séquence la plus déterministe de longueur 45*

Les tests réalisés sur 5 séquences protéiques illustrent le fait qu'une dégradation de performance, lorsqu'elle est constatée, reste limitée (moins de 3%). En revanche, lorsqu'une amélioration se produit, elle est relativement nette (jusqu'à 5%)

4.4.3 Adaptation du modèle de n-grams (homologie)

Une autre expérimentation consiste, étant donnée une séquence protéique à analyser, à construire les tables d'usage des $FR-1$-nuplets et des FR-nuplets au moyen des seules séquences protéiques du génome présentant une homologie avec la séquence à analyser. Deux protéines sont dites homologues lorsque le taux de discordance[6] ne dépasse pas un seuil fixé (30 % en général). Comme, plus la séquence protéique à analyser est longue, moindre sera le nombre de séquences du génome utilisées pour établir les statistiques, et par conséquent d'autant moins discordantes avec la séquence protéique initiale seront les 10 solutions trouvées, il est impératif de limiter la longueur de la séquence protéique à analyser. Le tableau 6 présente les résultats obtenus pour des homologies à 30% et à 20% et pour un facteur de regroupement FR égal à 2. Sur l'ensemble des tests *a priori* significatifs[7], la moitié s'est révélée sans effet net par rapport au calcul classique, un tiers des résultats étant moins bon et le reste étant nettement amélioré.

6. La définition du taux de discordance, jusqu'alors utilisée pour les séquences nucléiques, s'applique aussi aux séquences protéiques.

7. Le nombre de séquences sélectionnées dans le génome n'est pas réduit à un singleton.

séquence protéique	**taille**	**FR = 2** génome entier	**FR = 2** séquences homologues à 30%	**FR = 2** séquences homologues à 20%
1	10	0.160	0.153 (9676, 9.01%)	
2	10	0.267	0.273 - (21025, 19.58%)	
3	10	0.153	0.157 (13132, 12.23%)	
4	10	0.160	0.150 (13583, 12.65%)	
5	10	0.233	0.237 (9953, 9.27%)	
6	10	0.237	0.233 (13534, 12.61%)	
7	10	0.090	0.083 - (10700, 9.97%)	
8	10	0.260	0.270 - (9047, 8.43%)	
9	10	0.270	0.270 (21902, 20.40%)	
10	20	0.217	0.230 - (1007, 1.88%)	0.212 (16899, 31.48%)
11	30	0.234	0.186 ++ (61, 0.17%)	
12	40	0.334	0.172 ++ (120, 0.0037%)	
13	44	0.208	0.019 (**2**, 0.0082%)	0.158 ++ (564, 2.31%)
14	46	0.211	0.009 (**1**, 0.0043%)	0.191 + (811, 3.48%)
15	50	0.307	0.145 (**1**, 0.0047%)	0.327 - (700, 3.26%)
17	148			

TAB. 6 – *Comparaison entre les moyennes des taux de discordance obtenus pour les 10 meilleures solutions (FR = 2 - calculs avec génome entier et génome restreint aux séquences "homologues" à la séquence à analyser (Légende : effet positif (+), négatif (-)) (N.B.: Le couple figurant entre parenthèses donne le nombre de séquences homologues et le pourcentage génome homologue / génome entier)*

4.4.4 Prédictions au hasard et selon le contexte

Le protocole expérimental choisi compare systématiquement les qualités de prédiction obtenues selon les deux modalités suivantes : la première modalité adopte l'hypothèse d'un processus naturel tenant compte de la notion de contexte, la seconde modalité consiste à choisir les codons au hasard. Selon ce protocole, nous comptabilisons les discordances par acide aminé, au lieu de comptabiliser les discordances globalement (et par bases nucléotidiques), sur la séquence protéique. Nous rappelons en tableau 7 le nombre de codons caractéristique de chaque acide aminé.

nombre de codons synonymes	**acide aminé**
1	M,W
2	C,D,E, F,H,K, N,Q,Y
3	I
4	A,G,P,T,V
6	L,R,S

TAB. 7 – *Nombre de codons caractéristique de chacun des 20 acides aminés*

A priori, les acides aminés dont le codage est le moins déterministe sont le plus susceptibles de présenter une discordance, dans leur traduction reverse, avec la séquence d'ADN de référence. Comme par ailleurs ces mêmes acides aminés sont aussi ceux qui sont les plus fréquemment rencontrés dans le génome, il est davantage probable que certaines de leurs traductions reverse ne concorderont pas avec les codons de la séquence d'ADN de référence.

Nous faisons varier la taille des séquences protéiques à traduire, et la longueur du contexte local (facteur de regroupement). Nous avons mis en œuvre l'algorithme de traduction reverse sur 1000 séquences protéiques de taille 10, puis 44, avec un facteur de regroupement (FR) égal à 2, et enfin sur 100 séquences de taille 300, en augmentant la taille du contexte local ($FR = 3$). Pour chacune d'entre elles, la solution naturelle (ou observée) est connue. Nous avons choisi désormais de ne retenir que la première solution calculée (la prédiction) par notre algorithme. De cette manière, nous établissons des statistiques, pour chaque type d'acide aminé, quant à la distribution entre codons. Nous avons par ailleurs effectué une traduction en choisissant au hasard le codon correspon-

dant à un acide aminé, mais en respectant les distributions théoriques. Nous avons réalisé cette dernière expérimentation pour l'ensemble des 1000 séquences de longueur 44 acides aminés.

Des calculs statistiques sur la fraction codante du génome nous fournissent les distributions théoriques. Le comptage des fréquences des codons présents dans les séquences observées (naturelles) donne les distributions observées. Un même type de comptage, réalisé sur les prédictions donne les distributions prédites.

Les résultats relatifs à la corrélation distribution théorique / distribution observée, par codon, sont donnés en annexe B en figures 1, 2 et 3 respectivement pour les 1000 séquences protéiques de longueur 10, 44 et 300. La table des codes utilisés pour désigner les codons est fournie en annexe B (cf tableau 9). Ces résultats révèlent une bonne corrélation en moyenne, excepté pour les codons les plus fréquents (dans la nature) des acides aminés les moins déterminés. Ces codons sont sur-représentés. Comme attendu, avec les séquences les plus longues, l'adéquation entre distribution théorique (qui concerne le génome entier) et distribution observée est la meilleure (44, 300).

Les résultats relatifs à la corrélation entre distribution prédite / distribution observée, par codon, sont donnés en annexe B en figures 4, 5, 6 et 7 respectivement pour les 1000 séquences protéiques de longueur 10 et 44, avec prédiction guidée par contexte local, pour les 1000 séquences protéiques de longueur 44 avec choix aléatoire du codon (mais respectant les distributions théoriques), et pour les 100 séquences de longueur 300. Une autre présentation des résultats est fournie en figures 8, 9, 10 et 11 ((10, FR=2), (44, FR=2), (44, choix aléatoire), (300, FR=3)). Y est calculé le pourcentage moyen d'erreur relative, pour chaque codon, sur l'ensemble des séquences ((nombre de prédits - nombre de prédits corrects)/nombre de prédits).

La comparaison des prédictions guidées par contexte local met en exergue, pour les longueurs 10 et 44, 12 codons pour lesquels la corrélation prédiction/observation est mauvaise (trop forts pourcentages prédits). Par voie de conséquence, certains codons sont sous-représentés, voire absents, de la prédiction (même sur un millier de séquences). Les acides aminés concernés sont : A,C,G,K,N,Q,R,S,T,L,P,V. À part C, K , N et Q qui possèdent 2 codons, les autres acides aminés cités présentent 4 ou 6 codons. Inversement, aucun acide aminé présentant 4 ou 6 codons n'est traduit de façon satisfaisante.

La situation est totalement différente pour ce qui concerne la prédiction avec choix aléatoire. Pour les acides aminés à plus de 4 synonymes, les codons les moins fréquemment représentés dans le génome ne sont pas systématiquement éliminés comme c'était le cas précédemment.

La qualité de prédiction est évaluée par le pourcentage moyen de codons discordants (tous codons confondus), calculé sur l'ensemble des séquences à traduire. Ce pourcentage est 51,239 pour la taille 10 (FR=2); 50,317 pour la taille 44 (FR=2). Bien que la corrélation entre distributions soit la meilleure lorsque le choix des codons est effectué au hasard, cette méthode conduit à un faible score : 59,340. Lorsque sont augmentées la taille de la séquence protéique et la taille du contexte local d'analyse (respectivement 300, FR=3), nous observons que la qualité de prédiction est sensiblement accrue : 40,152.

5 Bilan - perspectives

Nous avons étudié le problème de la prédiction des meilleures traductions reverse d'une séquence protéique sur la base de statistiques établies sur le génome (prise en compte du contexte). L'originalité de notre contribution consiste à réaliser un couplage entre un analyseur syntaxique et un analyseur statistique. Du point de vue syntaxique, nous nous sommes inspirés du formalisme SVG proposé par D.Searls. Ce dernier introduit un nouveau type d'objet en analyse syntaxique, les variables de type chaîne, et définit une classe de transformations applicables à ces variables. Cette vision transformationnelle semble d'ailleurs particulièrement adaptée à la réalité biologique (répétitions, répétitions inversées, palindromes, substitutions, mutations...). Pour diriger le choix des meilleures solutions, il faut alors être capable de classer ces alternatives en fonction de leur probabilité de présence dans la nature. Pour ce faire, nous avons utilisé un modèle de n-grams.

Dans tous les cas, les qualités de prédiction restent relativement faibles, ce qui met en évidence l'importance d'autres facteurs que la notion de contexte local dans les processus d'évolution naturelle.

En particulier, nous n'avons pas pris en compte les caractéristiques physico-chimiques des bases nucléotidiques. Nous nous en sommes tenus à une approche syntaxique, axée sur l'utilisation de l'outil de modélisation que représente le formalisme SVG. Dans ce domaine linguistique, une amélioration à apporter consisterait à exprimer des contraintes supplémentaires au moyen de spécifications SVG. Par exemple, nous pourrions contraindre la séquence nucléotidique à ne pas être un palindrome biologique (de façon qu'elle ne s'hybride pas avec elle-même). Nous pourrions également interdire toute possibilité d'hybridation de l'une quelconque de ses extrémités avec cette même extrémité (L'extrémité ne doit pas être un palindrome biologique non plus.).

Références

[Cho55] N. Chomsky. *The logical structure of linguistic theory.* The University of Chicago Press, Chicago, 1975, 1955.

[CKVC83] A. Colmerauer, H. Kanaoui, and M. Van Caneghem. Prolog ii, bases théoriques et développements actuels. *Technique et science informatiques*, 2(4), 1983.

[Jel83] F. Jelinek. A maximum likelihood approach to continuous speech recognition. IEEE *transactions on pattern analysis and machine intelligence*, 5(2):179–190, march 1983.

[Jel90] F. Jelinek. *Self-organized language modeling for speech recognition*, pages 450–506. Morgan Kaufmann, San Mateo Calif., 1990.

[PW80] F.C.N. Pereira and D.H.D. Warren. Definite clause grammars for language analysis - a survey of the formalism and a comparison with augmented transition networks. *Artificial intelligence*, (13):231–278, 1980.

[Sea89] D.B. Searls. Investigating the linguistics of DNA with definite clause grammars. In E. Lusk and R. Overbeek, editors, *Logic programming: proceedings of the North American Conference*, pages 189–208. MIT Press, 1989.

[Sea93a] D.B. Searls. *The computational linguistics of biological sequences*, pages 47–120. AAAI/MIT Press, 1993.

[Sea93b] D.B. Searls. String variable grammar: a logic grammar formalism for the biological language of DNA. *Journal of logic programming*, (12):1–29, 1993.

A appendix a

numéro	séquence protéique
1	MNKNRGFTPL
2	AVVLMLSGSL
3	QITTELPGRT
4	FKEGSDIEAG
5	QQGGQQMPAV
6	QVSGIILKRN
7	FAWVIAIIIM
8	HTVFNQPIPL
9	VGAGGAGLRA
10	MNKNRGFTPL AVVLMLSGSL
11	MNKNRGFTPL AVVLMLSGSL ALTGCDDKQA
12	MNKNRGFTPL AVVLMLSGSL ALTGCDDKQA QQGGQQMPAV
13	MFKTTLCALL ITASCSTFAA PQQINDIVHR TITPLIEQQK IPGM
14	FYTNDLSMAS LGVAAVAIAV LAVLNLCGAR RTGVYILVGV VLWTAG
15	MNKNRGFTPL AVVLMLSGSL ALTGCDDKQA QQGGQQMPAV GVVTVKTEPL
16	MINSPRVCIQ VQSVYIEAQS SPDNERYVFA YTVTIRNLGR APVQLLGRYW LITNGNGRET EVQGEGVVGV QPLIAPGEEY EVQGEGVVGV QPLIAPGEEY QYTSGAIIET PLGTMQGHYE QYTSGAIIET PLGTMQGHYE MIDENGVPFS IDIPVFRLAV PTLIH
17	MIMANSGATS GWYHDFLETP VQLRVGSLEI NKNMLLWIND ALMAVFFLLV GLEVKRELMQ GSLASLRQAA FPVIAAIGGM IVPALLYLAF NYADPITREG WAIPAATDIA FALGVLALLG SRVPLALKIF LMALAIIDDL GAIIIIAL
18	MNNRVHQGHL ARKRFGQNFL NDQFVIDSIV SAINPQKGQA MVEIGPGLAA LTEPVGERLD QLTVIELDRD LAARLQTHPF LGPKLTIYQQ DAMTFNFGEL AEKMGQPLRV FGNLPYNIST PLMFHLFSYT DAIADMHFML QKEVVNRLVA GPNSKAYGRL SVMAQYYCNV IPVLEVPPSA FTPPPKVDSA VVRLVPHATM PHPVKDVRVL SRITTEAFNQ RRKTIRNSLG NLFSVEVLTG MGIDPAMRAE NISVAQYCQM ANYLAENAPL QES
19	MATYLIGDVH GCYDELIALL HKVEFTPGKD TLWLTGDLVA RGPGSLDVLR YVKSLGDSVR LVLGNHDLHL LAVFAGISRN KPLDRLTPLL EAPDADELLN WLRRQPLLQI DEEKKLVMAH AGITPQWDLQ TAKECARDVE AVLSSDSYPF FLDAMYGDMP NNWSPELRGL GRLRFITNAF TRMRFCFPNG QLDMYSKESP EEAPAPLKPW FAIPGPVAEE YSIAFGHWAS LEGKGTPEGI YALDTGCCWG GTLTCLRWED KQYFVQPSNR HKDLGEAAAS

TAB. 8 – *Description des séquences protéiques*

B appendix b

code utilisé	codon	code utilisé	codon
A_1	gcg	N_1	aac
A_2	gcc	N_2	aat
A_3	gca	---	---
A_4	gct	P_1	ccg
---	---	P_2	cca
C_1	tgc	P_3	cct
C_2	tgt	P_4	ccc
---	---	---	---
D_1	gat	Q_1	cag
D_2	gac	Q_2	caa
---	---	---	---
E_1	gaa	R_1	cgc
E_2	gag	R_2	cgt
---	---	R_3	cgg
F_1	ttt	R_4	cga
F_2	ttc	R_5	aga
---	---	R_6	agg
G_1	ggc	---	---
G_2	ggt	S_1	agc
G_3	ggg	S_2	tcc
G_4	gga	S_3	tct
---	---	S_4	tcg
H_1	cat	S_5	agt
H_2	cac	S_6	tca
---	---	---	---
I_1	att	T_1	acc
I_2	atc	T_2	acg
I_3	ata	T_3	act
---	---	T_4	aca
K_1	aaa	---	---
K_2	aag	V_1	gtg
---	---	V_2	gtt
L_1	ctg	V_3	gtc
L_2	ttg	V_4	gta
L_3	tta	---	---
L_4	ctc	W	tgg
L_5	ctt	---	---
L_6	cta	Y_1	tat
---	---	Y_2	tac
M	atg		

TAB. 9 – *Description des codons (N.B: Le préfixe du code désigne l'acide aminé concerné.)*

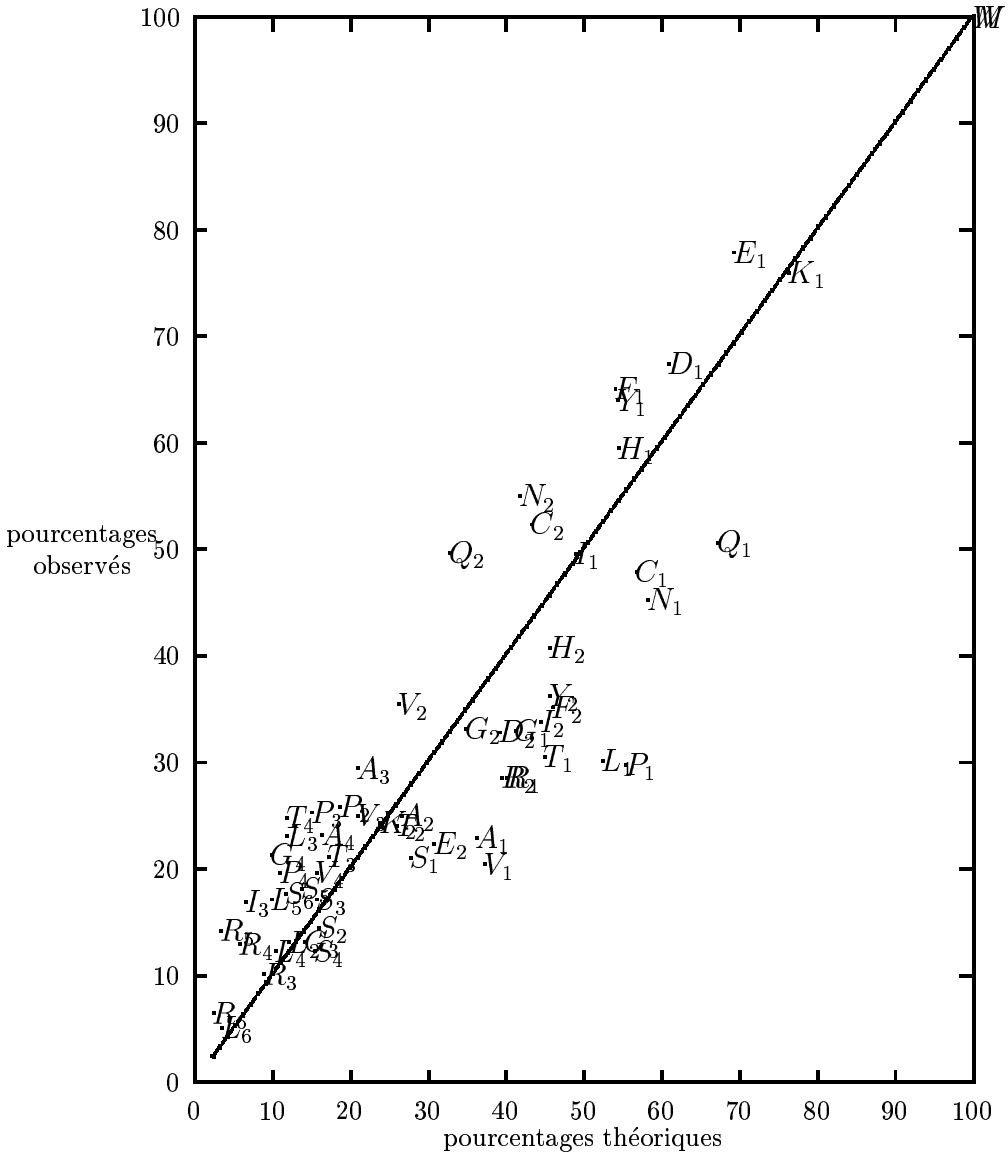

FIG. 1 – *Comparaison des distributions théorique et observée, par codon, pour 1000 séquences protéiques de longueur 10*

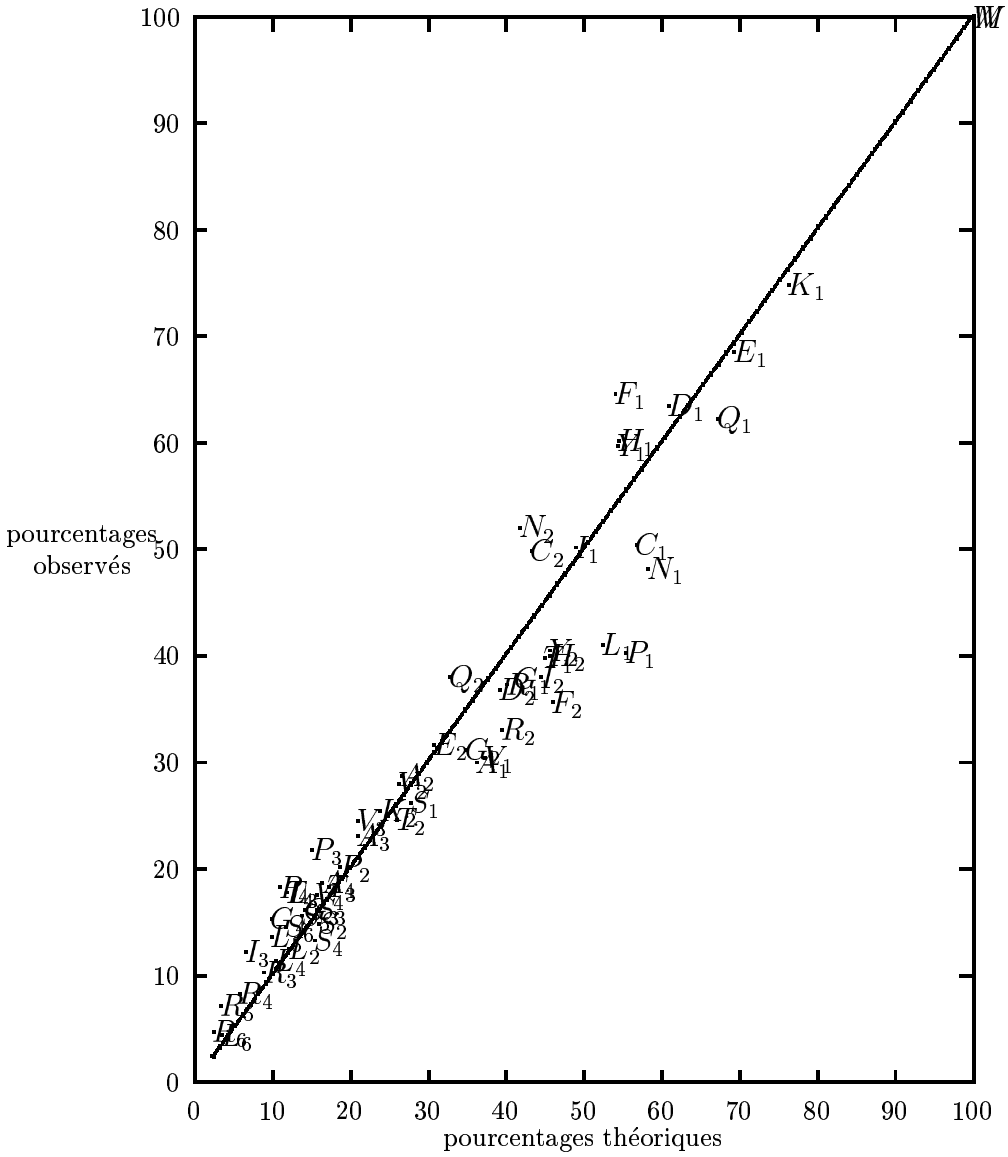

FIG. 2 – *Comparaison des distributions théorique et observée, par codon, pour 1000 séquences protéiques de longueur 44*

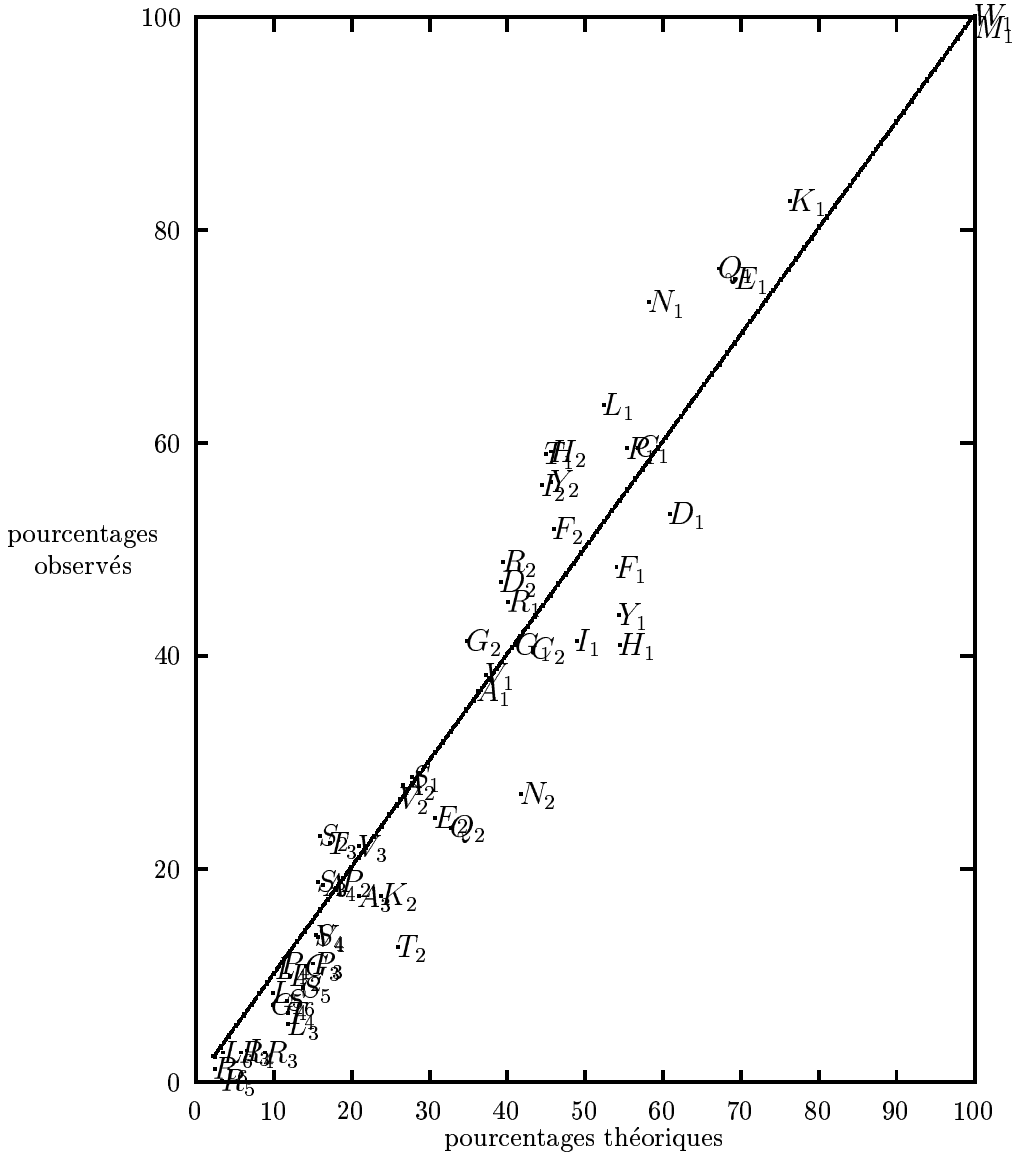

FIG. 3 – *Comparaison des distributions théorique et observée, par codon, pour 100 séquences protéiques de longueur 300*

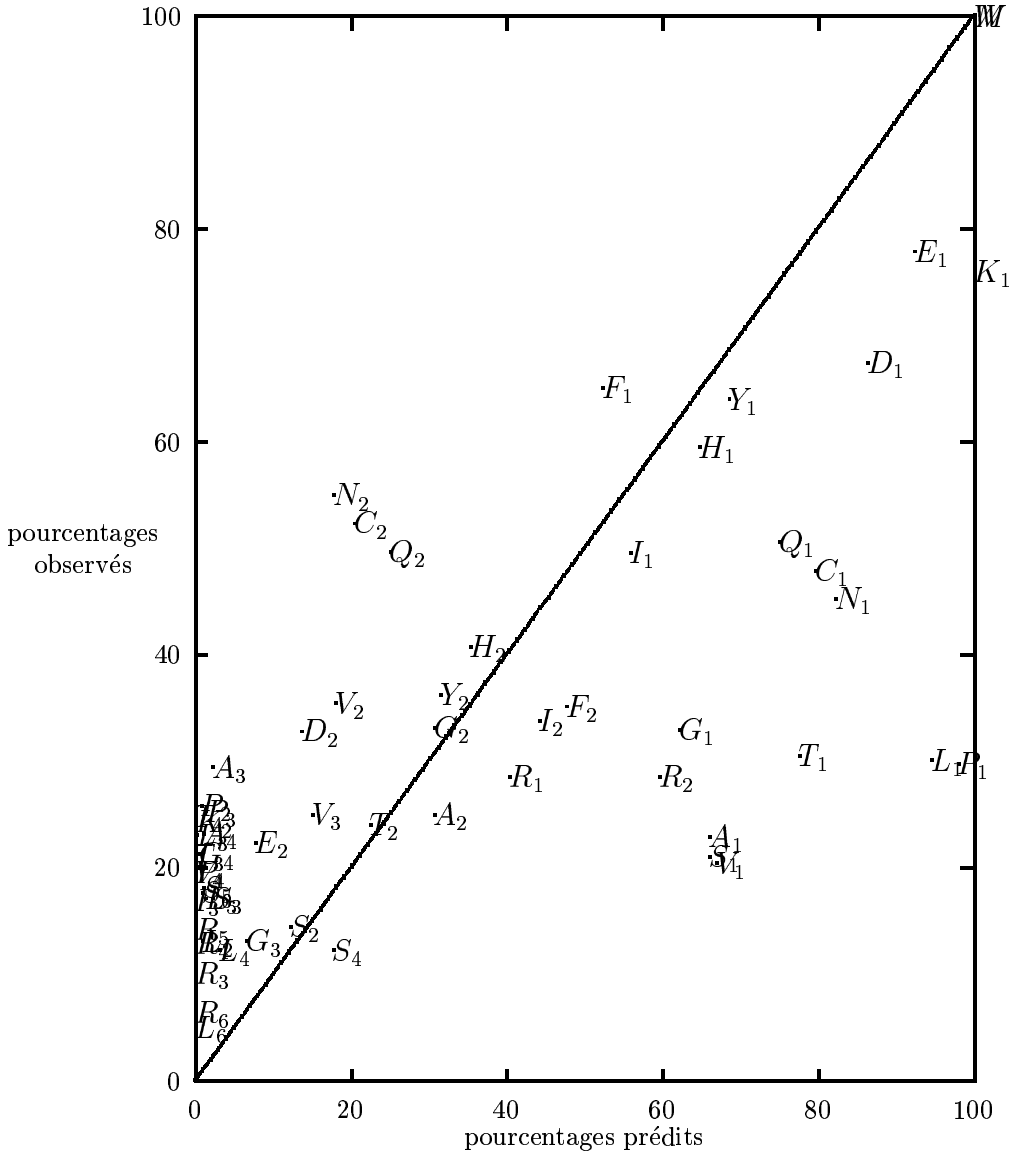

FIG. 4 – *Comparaison des distributions prédite et observée, par codon, pour 1000 séquences protéiques de longueur 10 (prédiction guidée par contexte local(FR=2))*

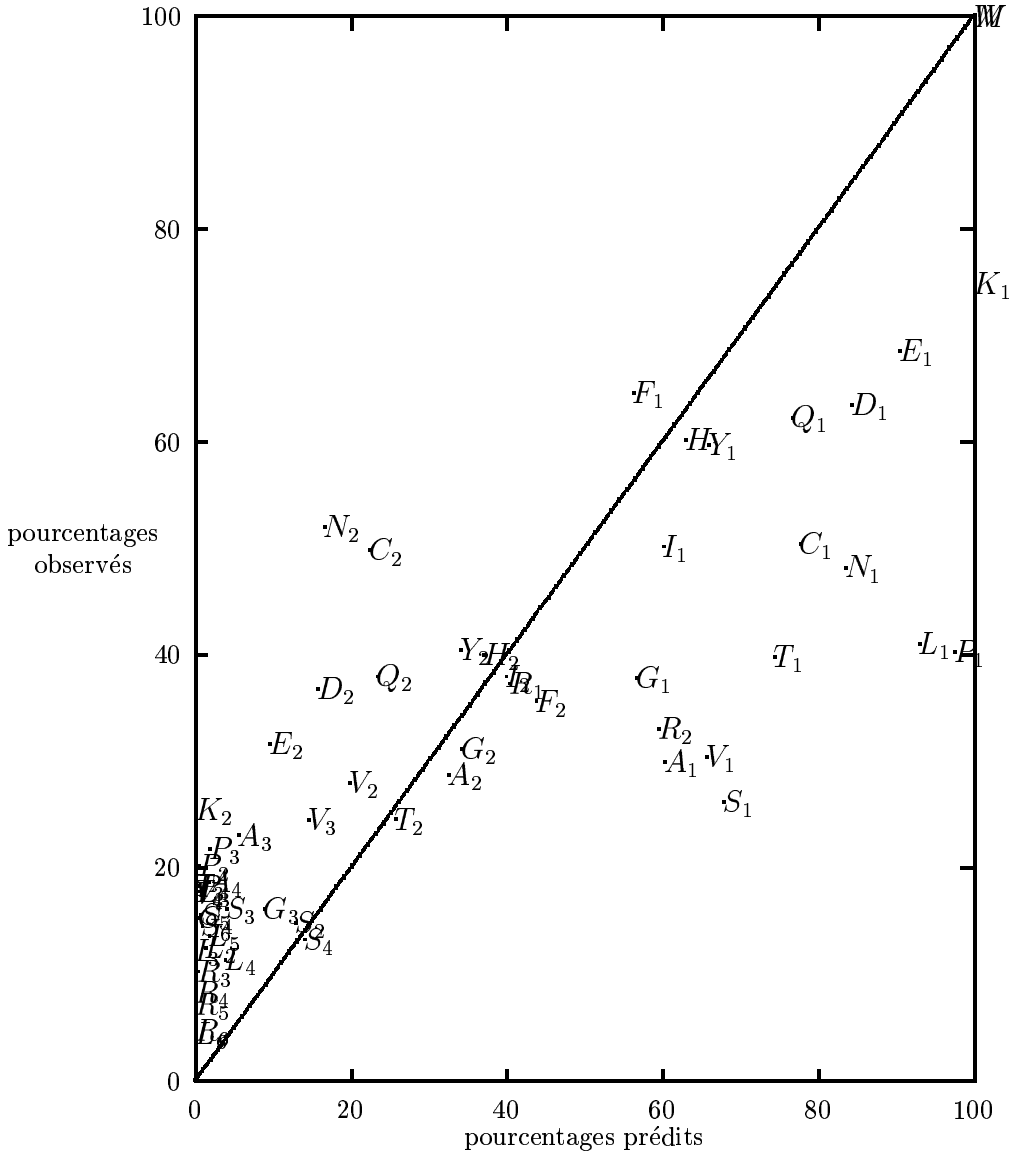

FIG. 5 – *Comparaison des distributions prédite et observée, par codon, pour 1000 séquences protéiques de longueur 44 (prédiction guidée par contexte local(FR=2))*

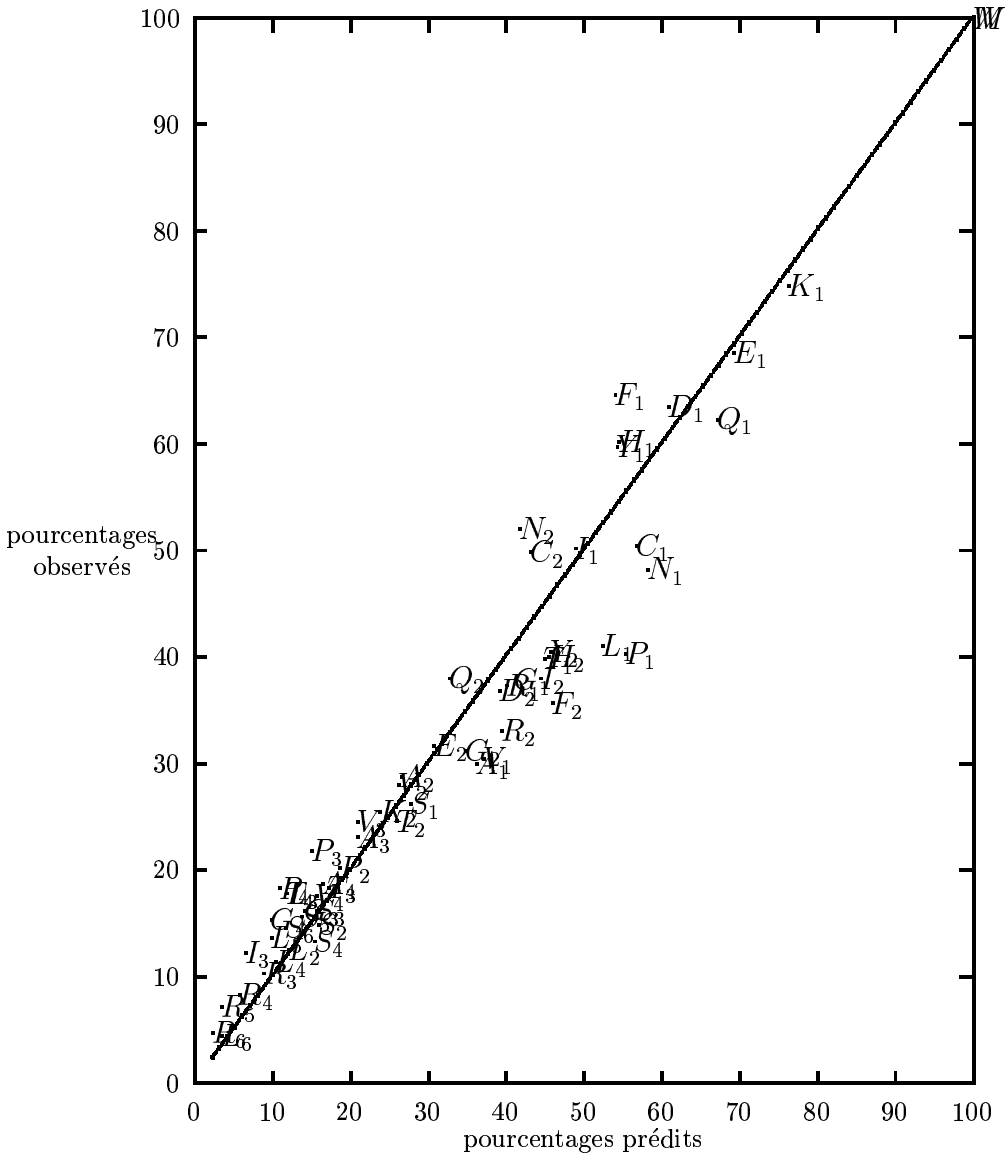

FIG. 6 – *Comparaison des distributions prédite et observée, par codon, pour 1000 séquences protéiques de longueur 44 (prédiction avec choix aléatoire du codon)*

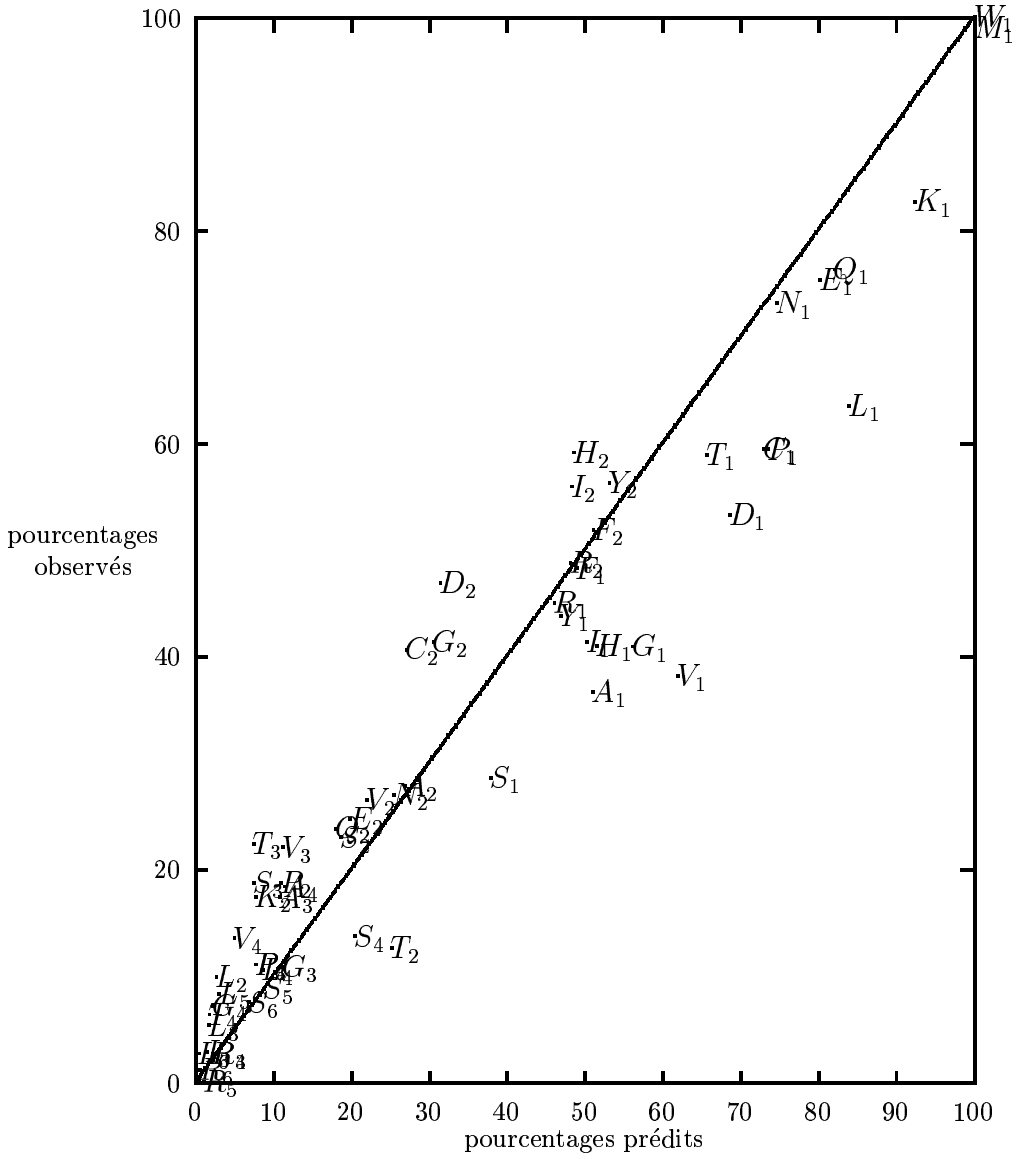

FIG. 7 – *Comparaison des distributions prédite et observée, par codon, pour 100 séquences protéiques de longueur 300 (prédiction guidée par contexte local(FR=3))*

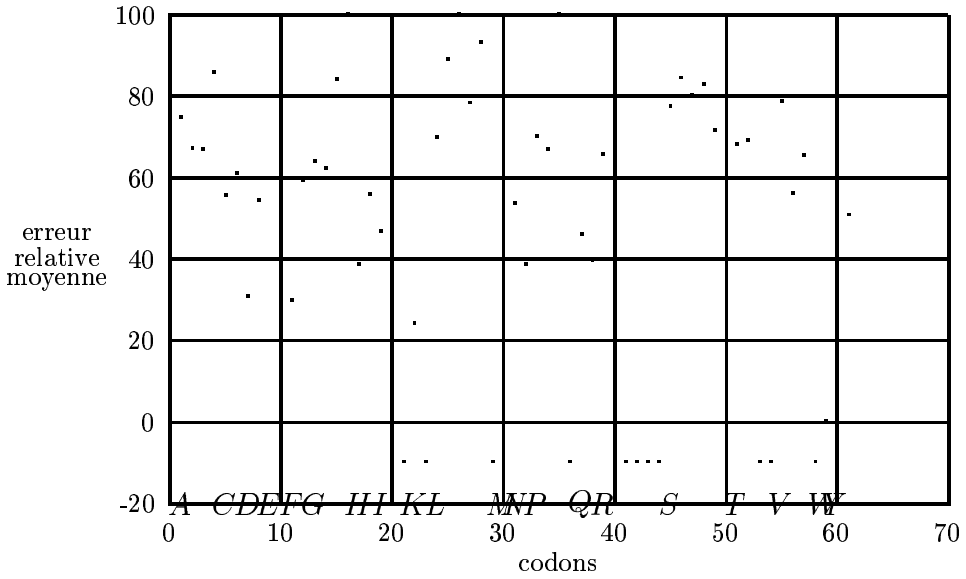

FIG. 8 – *Pourcentage moyen d'erreur relative (nb prédits - nb prédits corrects)/nb prédits, pour chaque codon, pour 1000 séquences protéiques de longueur 10 (prédiction guidée par contexte local(FR=2)) - N.B.: Un nombre négatif signifie l'absence du codon.*

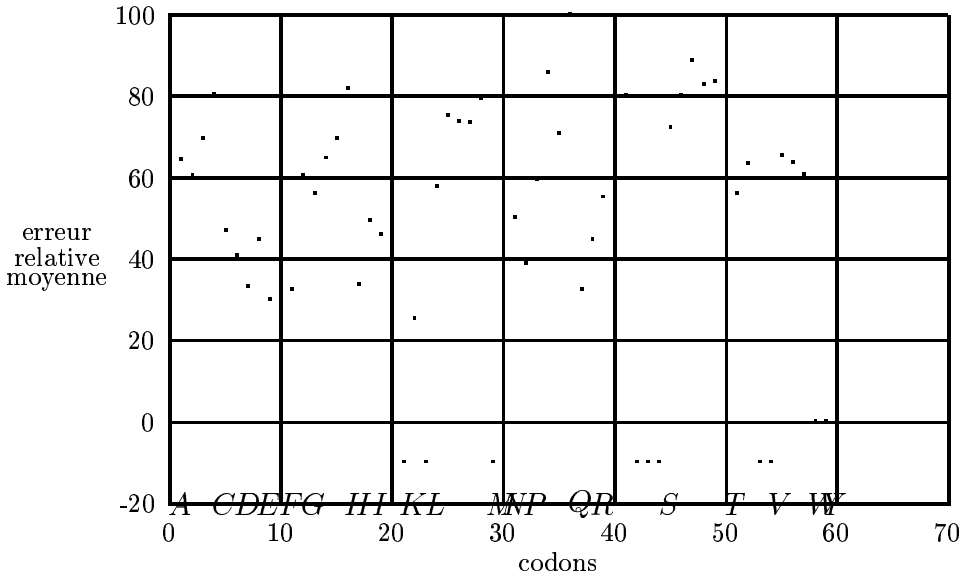

FIG. 9 – *Pourcentage moyen d'erreur relative (nb prédits - nb prédits corrects)/nb prédits, pour chaque codon, pour 1000 séquences protéiques de longueur 44 (prédiction guidée par contexte local(FR=2)) - N.B.: Un nombre négatif signifie l'absence du codon.*

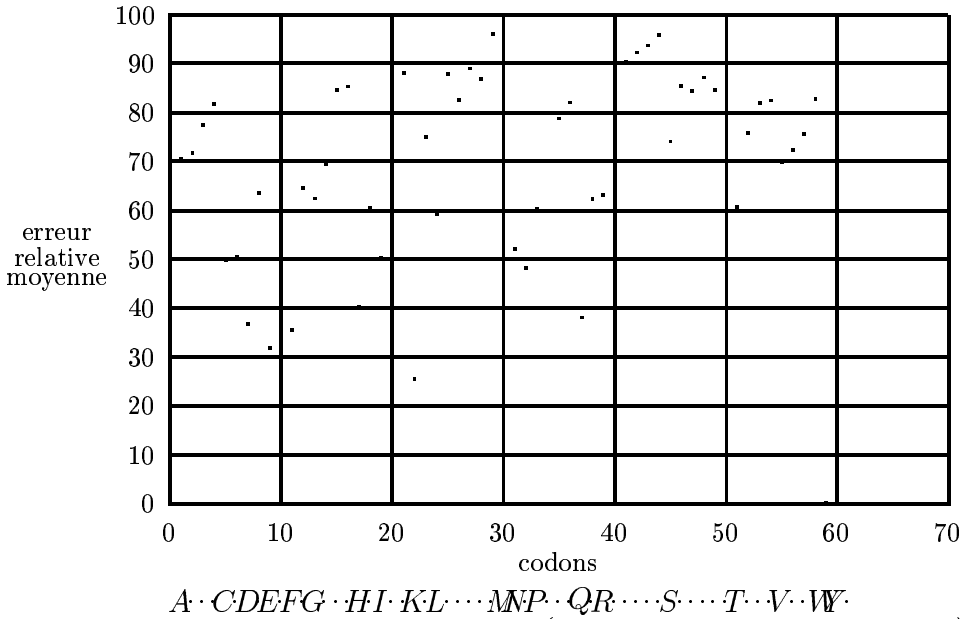

FIG. 10 – *Pourcentage moyen d'erreur relative (nb prédits - nb prédits corrects)/nb prédits, pour chaque codon, pour 1000 séquences protéiques de longueur 44 (prédiction avec choix aléatoire du codon)*

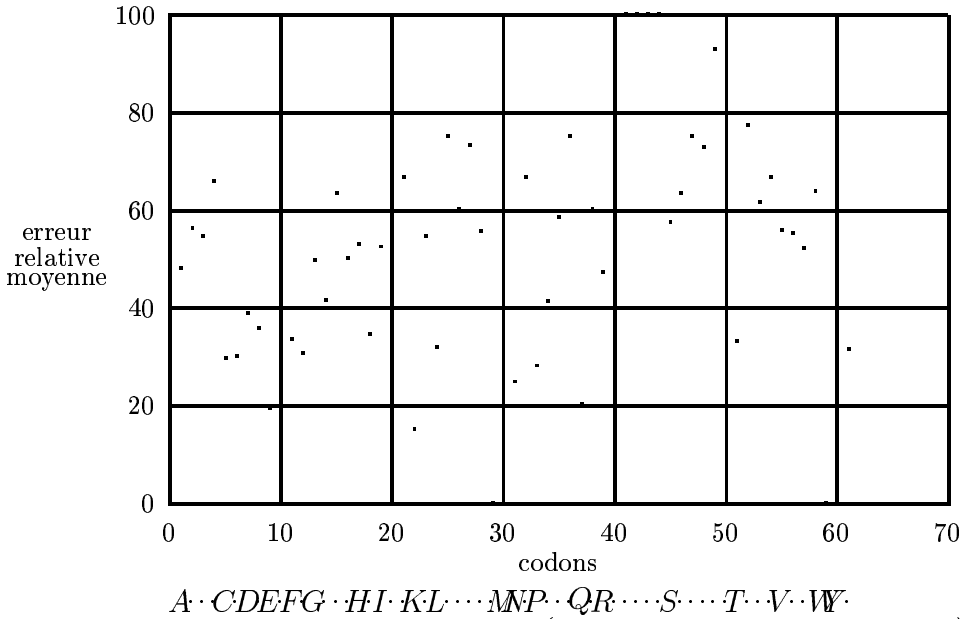

FIG. 11 – *Pourcentage moyen d'erreur relative (nb prédits - nb prédits corrects)/nb prédits, pour chaque codon, pour 100 séquences protéiques de longueur 300 (prédiction guidée par contexte local(FR=3)) - N.B.: Un nombre négatif signifie l'absence du codon.*

Unit´e de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unit´e de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unit´e de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unit´e de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unit´e de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399