Suites infinies, complexité et géométrie

TAPSOBA T.

RÉSUMÉ

Par une démarche très simple, nous montrons que la définition «classique» de la fonction de complexité ne permet pas vraiment de décrire comment une suite est ... «complexe».

Mots clés : Suites infinies, suites ultimement périodiques, substitutions, complexité.

ABSTRACT

Using a simple approach we show that the «classical» definition of complexity function does not allow to describe how a sequence is ... «complex».

Key-words : Infinite sequences, ultimately periodic sequences, substitutions, complexity.

I. INTRODUCTION

1.1 Mots

Soit A un ensemble fini appelé alphabet dont les éléments sont appelés lettres et soit n un entier strictement positif. Un mot u de longueur n (on note alors $|u| = n$) sur l'alphabet A est par définition u_n n-uplet $(u_1, \ldots, u_n)$ de lettres de A. L'ensemble des mots sur l'alphabet A est noté A*. Par convention, on appelle mot vide le mot de longueur 0 (souvent noté e).

Sur A* on définit une importante opération appelée concaténation qui associe à deux mots u et v de A*, un mot w noté uv défini comme suit:

- Si u = e, alors uv = v,
- Si v = e, alors uv=u,
- Si $u = (u_1, \ldots, u_n)$ et $v = (v_1, \ldots, v_k)$, alors $u_v = (u_1, \ldots, u_n, v_1, \ldots, v_k)$.

On confère ainsi une structure de monoïde à A* où le mot vide est l'élément neutre. Comme la concaténation est associative, on se passe de parenthèses pour écrire tout simplement le mot uvw à la place de u(vw) ou (uv)w.

1.2 Facteurs

Soit un mot de A*. Le mot v de A* est dit factaur de u, s'il existe deux mots w et w' de A* (éventuellement vide(s)) tels que u=wvw'. Avec cette notation, si w (resp. w') est le mot vide, v est dit préfixe (resp. suffixe) de u. On désigne respectivement par Fact(u), Préf(u), Suf(u) les ensembles des facteurs, préfixes et suffixes de u.

1.3 Langages

Un ensemble de mots sur un alphabet donné est appelé langage sur cet alphabet. Trois groupes principaux d'opérations peuvent être définis sur les langages :

- les opérations booléennes ; il s'agit là des opérations ordinaires de la théorie des ensembles : union, intersection, complémentation.
- le produit et les puissances ; si L et M sont deux langages sur l'alphabet A, le produit L.M de ces langages est l'ensembles des mots w vérifiant w=uv où uvL et vvM. La n-ième puissance de L, notée Ln est définie par récurrence par $L^0=e$, $L^{n+1}=L.L^n=L^n.L$.
- l'itération; l'itérée du langage L, notée L*, est l'union de toutes les puissances de L.

1.4 Remarque

Le lecteur intéressé pourra se référer à [14] pour plus de détails sur toutes les notions ci-dessus.

II. SUITES INFINIES

2.1 Suites ultimement périodiques

Une suite $(x_n)_{n \vee IN}$ est dite ultimement périodique s'il existe un entier non nul τ et un entier $_{n0}$ tels que $x_{n+\tau}=x_n$ pour tout $_{n\geq 0}$.

2.2 Une approche

Lançons «indéfiniment» une pièce de monnaie et regardons la suite :

PFFFPPFFPPFPFF... des «pile» et «face» obtenue. Cette suite étant construite au hasard, les lois statistiques seront observées ; ainsi la fréquence d'apparition d'un mot donné, par exemple PF, sera très proche de sa probabilité (dans notre cas) et cela sera certainement vrai pour tout mot.

Plus généralement, une suite sur un alphabet de deux lettres est dite normale si pour tout entier strictement positif p, tout mot de longueur p apparaît dans la suite avec la fréquence $\frac{1}{2^p}$.

Il existe des algorithmes pour construire des suites normales. La plus connue de ces suites est celle de Champernowne : on écrit la suite des entiers naturels 0, 1, 2,... en base deux, ce qui donne la suite 011011100101110111...

Si les procédés pour construire des suites infinies sont nombreux, l'auto-référence est certainement la méthode la plus simple pour obtenir des suites non ultimement périodiques.

Ecole Supérieure d'Informatique
Univ. polytech. de Bobo-Dioulasso
01 BP 1091 Bobo-Dioulasso 01
Burkina Faso
Tél. : (+226) 70 26 05 14
E-Mail : theo_tapsoba@univ-ouaga.bf

2.3 Suites se référant à elles-mêmes

Une suite $(x_n)_{nvIN}$ est dite Markovienne si x_{n+1} ne dépend que de x_n. Par ce que nous appelons l'auto-référence nous obtenons des suites qui ne sont pas Markoviennes, bien que n'étant pas très éloignées. Une suite se référant à elle-même est une suite telle que lorsque les n premiers termes sont écrits, on sait écrire le (n+1)-ième terme en ne regardant que les m premiers termes, où m≤n. On essaye donc de créer un programme permettant d'écrire ses termes successifs. Ce programme peut lui aussi être écrit comme une suite sur le même alphabet. Plus cette dernière suite est courte, moins la suite donnée au départ est complexe.

Malheureusement cette définition classique de la complexité d'une suite due à Kolmogorov, Chaitin et Solomonoff [8] n'est pas effective ; en effet, pour une suite donnée, il n'existe pas toujours un algorithme permettant d'écrire le plus petit programme nécessaire pour la générer.

Dans ce qui va suivre, nous nous restreindrons à une auto-référence particulière nommée substitution, on obtient alors des suites infinies appelées suites substitutives ([13], [9], [5], ...).

2.4 Suites substitutives

Soit A= {0, 1} et f la fonction de A dans A* définie par f (0)=01, et f (1)=0. Nous étendons cette définition aux mots et aux suites par f (xy)=f(x) f(y).

En commençant du mot 0, on obtient successivement 01, 010, 01001, 01001010, ... Il est facile de voir que si u(n) est le mot obtenu à la n-ième étape, u(n) est préfixe de u(n+1). On construit ainsi une suite substitutive (notée Fib et appelée suite de Fibonacci ([4], [10], [7], [12], ...) vérifiant Fib=0100101001001... et qui débute par u(n) pour tout n. On a évidemment Fib=f (Fib) et c'est la seule suite ayant cette propriété.

Posons $\alpha = \dfrac{\sqrt{5}-1}{2}$ soit X = [0, 1[, X0 = [0, α[et X_1 = [α, 1[. Soit T : → X tel que T(x)=x+1-α si xvX_0 et T(x)=x-α si xvX_1. Désignons par χ(x) l'élément de {0, 1} tel que $xvX_{\chi(x)}$. La suite $(\chi(T^n(x)))_{nvIN}$ est appelée l'itinéraire de x et est notée it(x). On a alors (voir par exemple [11]) :

***Proposition 2.1** Les conditions suivantes sont équivalentes :*

(i) Pour tout x de X, l'orbite de x, c'est-à-dire la suite $(T^n(x))_{nvIN}$ est dense dans X.

(ii) Pour tout x de X, le langage Fact(it(x)) ne dépend pas de x. on le notera L.

(iii) Le nombre P(n) des éléments de L de longueur n est P(n)=n+1.

(iv) La suite Fib est l'itinéraire du point α^2.

Remarque 2.1 (i), (ii) et (iii) restent vraies pour tout irrationnel α de [0, 1[.

III. Complexité

3.1 La fonction de complexité

La suite de Fibonacci est peut-être l'exemple le plus «simple» de suite substitutive. Il est donc permis de penser que la notion de complexité prendra ce fait en compte... En effet, la proposition précédente semble indiquer d'une part que la fonction P(n) joue un rôle dans la description de la complexité et d'autre par que cette fonction peut facilement être calculée lorsque des propriétés géométriques sont connues.

D'où la définition suivante de la fonction de complexité (voir par exemple [2]) : soit x une suite infinie et L=Fact(x) le langage associé à x. La fonction de complexité de x, notée P(x, n) est le nombre de facteurs distincts de longueur n de L.

3.2 Complexité et prédiction

Soit x une suite sur un alphabet A et P(x, n) sa complexité. Supposons que l'on sache qu'un mot u de longueur n apparaît à un certain endroit dans x, c'est-à-dire qu'il existe k tel que x(k)x(k+1)... x(k+n-1)=u. Peut-on prédire alors la valeur de x(k+n)?

Posons λ(u)=Card{avA|uavFact(x)}. Si λ(u)=1, u se prolonge de manière unique (mettons ua) et on peut prédire avec exactitude que x(k+n)=a. Il y a donc deux cas extrêmes:

(i) « u, mot de longueur n, λ(u)= 1; (parfaite prédiction).

(ii) « u, mot de longueur n, λ(u)= Card(A) ; (prédiction impossible).

Notons que pour |u|=n, la somme sur u des λ(u) vaut P(n+1), et donc que la somme des (λ(u)-1)=P(n+1)-P(n) (notée s(n)). Dans le cas (i), s(n)=0 et donc P(n+1)=P(n) ; dans le cas (ii), s(n)=P(n)(Card(A)-1) si bien que P(n+1)=P(n) Card(A).

Si (ii) est vraie pour tout n, on obtient $P(n)=(Card(A))^n$. Si (i) est vraie pour tout n, on montre facilement que pour |v|=n+1, la somme des (λ(u)-1)=0=s(n+1) et une récurrence donne P(m)=P(n) pour tout m≥ n. Il est clair (Voir néanmoins [6] par exemple) que pour que l'on ait cette dernière égalité, il faut et il suffit que x soit ultimement périodique. On en conclut de prime abord que la croissance de la suite $(P(n+1)-P(n))_{nvIN}$ mesure dans un certain sens la possibilité de prédire la suite x.

IV. Complexité et Géométrie

Nous avons dit plus haut que si pour une suite x, la fonction P(n) n'est pas strictement croissante alors x est ultimement périodique. Ainsi, avoir une suite non ultimement périodique suppose s(n)≥1 pour tout n. La complexité la plus faible envisageable est donc P(n)=n+1. Rappelons que (on peut le prouver par des arguments géométriques) la suite de Fibonacci vérifie cette propriété et on a (voir par exemple [3]) :

***Proposition 4.1** Si une suite x vérifie P(x)=n+1 alors elle est l'itinéraire d'un échange entre deux intervalles.*

Il vient alors cette question: Si pour une suite donnée, P(n) a une expression simple, a t-elle une interprétation géométrique? De nos jours, la question précédente n'est pas résolue dans sa généralité. Il est indispensable d'ajouter des conditions de type combinatoire (voir par exemple [1]) si on désire avoir des caractérisations nécessaires et suffisantes. Il apparaît donc que la fonction P(n) seule ne suffit pas pour vraiment décrire la complexité d'une suite...

RÉFÉRENCES

1- **ARNOUX P. et RAUZY G.,** Représentation géométrique de suites de complexité 2_{n+1}, Bull. Soc. Math. de France 119 (1991), 199-215.

2- **COBHAM A.,** Uniform tag sequences, Math. Syst. Theo. 6 (1972) 164-192.

3- **COVEN E. et HEDLUND G.A.,** Hedlund, Sequences with minimal block growth, Math. Systems Theory 7 (1973), 138-153.

4- **De LUCAS A.,** A combinatorial property of the Fibonacci word, Information processing. Letters 12 n° 4 (1981),193-195.

5- **DURAND F.,** Contribution à l'étude des suites et systèmes dynamiques substitutifs, Thèse de Doctorat, Université d'Aix-Marseille II (1996).

6- **HEDLUND G.A. et MORSE M.,** Symbolic dynamics, Amer. J. Math. 60 (1938), 815-866.

7- **KASA Z.,** Computing the d-complexty of words by Fibonacci-like sequences, Studia Univ.. Babes-Bolyai. Math. 35 (1990), 49-53.

8- **LI M. et VIANTANYI P.,** Kolmogorov complexity and its applications, in J. Van Leeuwen éditeur, Handbook of Theoretical Computer Science, Volume A : Algorithms and complexity, MIT Press (1990), 187-254.

9- **MOULINE J.,** Contribution à l'étude de la complexité des suites substitutives, Thèse de Doctorat, Université de Provence (1989).

10-**PANSIOT J.-J.,** Mots de Fibonacci et morphismes itérés, Lecture Notes in Computer Science 172 (1984), 380-389.

11-**QUEFFELEC M.,** Substitution dynamical systems, Spectral analysis, Lecture Note in Mathematics 1294 (1987), Pringer-Verlag Berlin.

12- **SÉÉBOLD P.,** Fibonacci morphisms and Sturmians words, Theoret. Comput. Sci. 88 (1991) 365-384.

13-**TAPSOBA T.,** Complexité de suites automatiques, Thèse de 3ème cycle, Université d'Aix-Marseille II (1987).

14-**TAPSOBA T.,** Contribution à l'étude des suites automatiques, Thèse de Doctorat d'Etat, Université de Ouagadougou (1999).