Les automates finis en mathématiques

par Jean-Paul Allouche

[NDLR : Le pliage de papier et les automates ont déjà inspirés des travaux MATh.en.JEANS. Le pliage de papier et le mot infini décrivant la suite de “virages” correspondante donne naissance à une courbe fractale célèbre appelée courbe du dragon (voir MATh.en.JEANS 1991, actes du congrès de Strasbourg, pp. 97-98, et MATh.en.JEANS 1994, actes du 5ème congrès, pp. 105-112 et 113-118).

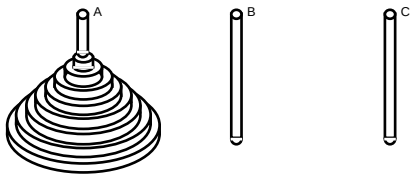
Le point de vue des automates finis permet l'étude “grammaticale” des suites de lettres, c'est-à-dire des règles de formation de *mots*. et établit une liaison intéressante avec la théorie des nombres. (voir aussi, plus loin, dans l'article « les autres sujets du congrès », une présentation des automates finis par Jean-Pierre Ressayre).]

Nous nous proposons de montrer comment les *automates finis* introduits à l'origine en théorie des langages formels, un peu à la croisée de l'informatique théorique et de la linguistique, permettent de construire des suites ayant de nombreuses propriétés mathématiques. Le lecteur désirant lire des survols plus complets peut se reporter à [4] et [1].

tours de Hanoï

Le jeu des tours de Hanoï inventé par Lucas (sous l'anagramme Claus) se compose de trois tiges verticales et de N disques de tailles toutes différentes, disons de tailles respectives 1, 2, …, N, percés d'un trou en leur centre.

Au départ les disques sont tous empilés sur la première tige, le disque N au-dessous, puis le disque N-1, … jusqu'au disque 1 au sommet de la tige. Le jeu consiste, à chaque étape, à prendre le disque situé au sommet de l'une des tiges et à le transporter sur une autre tige, en respectant la règle : *un disque ne peut pas reposer sur un disque de numéro strictement inférieur*.

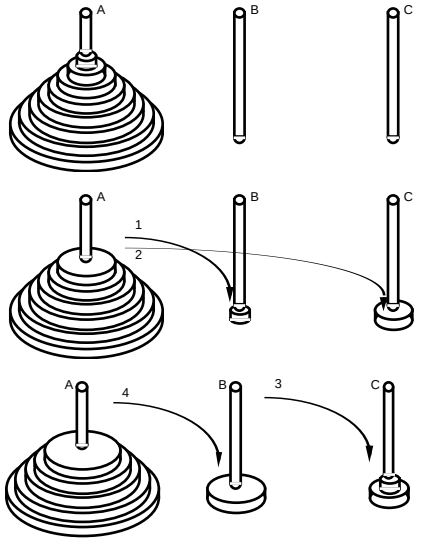


Le but est d'empiler tous les disques sur une autre tige. Remarquons qu'à cause de la règle ci-dessus ils seront nécessairement empilés dans le même ordre qu'au départ, c'est-à-dire le grand disque au-dessous et par ordre décroissant de numéros.

L'algorithme classique, qui donne d'ailleurs le nombre minimal de mouvements, consiste à remarquer que pour pouvoir déplacer le disque N, il faut avoir d'une part enlevé les N–1 disques qui reposaient sur lui, d'autre part avoir une tige libre (car ce disque ne peut reposer sur aucun des autres disques), donc il est nécessaire et suffisant d'avoir déplacé ces N–1 autres disques vers une autre tige, (autre-

ment dit d'avoir déjà résolu la question pour N–1 disques), puis de transporter le disque N sur la tige vide et … de transporter à nouveau les N–1 disques de la tige où ils attendaient sur celle où l'on a posé le disque N.

Le lecteur peut s'arrêter un instant et tenter de jouer par lui-même ou regarder la figure.



Un tel algorithme est dit *récursif*, on voit qu'on définit ainsi une procédure N en appelant deux fois la procédure N–1 etc. La procédure 1, qui consiste à jouer avec un seul disque n'est pas bien difficile ! Un tel algorithme se programme fort aisément dans un langage qui permet la récursivité, comme *Pascal*. On voit aussi qu'il est nécessaire de garder en mémoire tout ce qui se trouve sur chaque tige pour calculer le(s) coup(s) suivant(s).

Convenons de choisir comme tige d'arrivée la deuxième si N est impair et la troisième si N est pair. Un instant de réflexion montre alors que, lors de l'utilisation de l'algorithme précédent, on peut définir une suite infinie de mouvements dont les “préfixes” de longueur $2^N - 1$ donnent exactement les déplacements nécessaires pour transférer N disques.

Plus précisément notons a le mouvement qui consiste à prendre le disque situé au sommet de la première tige pour le mettre au sommet de la deuxième, b celui qui transfère le disque du sommet de la deuxième tige sur la troisième tige, et c celui qui transfère le disque du sommet de la troisième tige vers la première. Appelons aussi $\overline{a}$, $\overline{b}$, et $\overline{c}$ les *inverses* des mouvements a, b et c respectivement (par exemple $\overline{c}$ transfère le disque au sommet de la première tige vers la troisième). La suite infinie de mouvements commence alors par :

$$a\ \overline{c}\ b\ a\ c\ \overline{b}\ a\ \overline{c}\ b\ \overline{a}\ c\ b\ a\ \overline{c}\ b\ a\ \ldots$$

[ci-contre, le début de ce début : $a\ \overline{c}\ b\ a$.]

On vérifie que le préfixe de longueur $2^1 - 1 = 1$, qui est a donne la solution du déplacement d'un disque, que le préfixe de longueur $2^2 - 1 = 3$ donne la solution du déplacement de deux disques …

Comment obtenir la suite infinie ci-dessus sur les six lettres a, b, c, $\overline{a}$, $\overline{b}$, $\overline{c}$, de façon plus “simple” ? Une première réponse est la suivante : considérons sur l'*alphabet* (ensemble fini de lettres) $\{a, b, c, \overline{a}, \overline{b}, \overline{c}\}$ le *morphisme* ([= l'application], on dit aussi la *substitution*) défini par :

$$\begin{aligned}
a &\mapsto a\ \overline{c}\\
b &\mapsto c\ \overline{b}\\
c &\mapsto b\ \overline{a}\\
\overline{a} &\mapsto a\ c\\
\overline{b} &\mapsto c\ b\\
\overline{c} &\mapsto b\ a
\end{aligned}$$

L'application ci-dessus associe à chaque lettre un *mot* de deux lettres. Elle est étendue aux mots en définissant l'*image* d'un mot [= le transformé par l'application] comme la *concaténation* des images des lettres dans le même ordre, par exemple :

$$a\ b\ \overline{c} \mapsto (a\ \overline{c})(c\ \overline{b})(b\ a)$$

qu'on écrit sans les parenthèses :

$$a\ b\ \overline{c} \mapsto a\ \overline{c}\ c\ \overline{b}\ b\ a$$

Si l'on utilise le terme “morphisme”, c'est que, si on considère l'ensemble des mots sur l'alphabet $\{a, b, c, \overline{a}, \overline{b}, \overline{c}\}$, c'est-à-dire l'en-

semble des suites finies de symboles [lettres] sur cet alphabet, on peut définir une opération sur les mots appelée “concaténation”, qui consiste simplement à écrire les mots bout à bout.

Par exemple :

$$a\ b\ b\ c\ .\ \overline{b}\ a\ \overline{c} = a\ b\ b\ c\ \overline{b}\ a\ \overline{c}$$

[le mot $a\ b\ b\ c\ \overline{b}\ a\ \overline{c}$ est le *concaténé* du mot $a\ b\ b\ c$ et du mot $\overline{b}\ a\ \overline{c}$.]

On obtient ainsi une loi de composition interne évidemment associative et d'élément neutre le mot vide. L'application ci-dessus vérifie alors clairement la propriété : l'image du concaténé de deux mots est le concaténé des images.

Itérons [= répétons] alors cette application en partant de a. Autrement dit, calculons les images successives de a, de l'image de a, de l'image de l'image de a … :

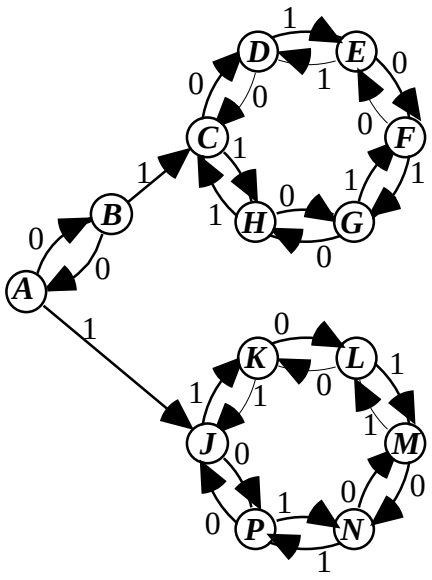
$a \mapsto a\ \overline{c} \mapsto a\ \overline{c}\ b\ a \mapsto a\ \overline{c}\ b\ a\ c\ \overline{b}\ a\ \overline{c}$ …

Deux choses apparaissent (et peuvent être démontrées).

D'une part chaque “nouveau” mot commence par le mot précédent. Cette “compatibilité” permet en itérant une infinité de fois d'obtenir une “limite”, c'est-à-dire une suite infinie dont chacun de ces mots est un préfixe [= un début]. Remarquons aussi que la suite infinie obtenue est invariante [= ne change pas] si on lui applique le morphisme ci-dessus (quitte à étendre la définition de ce morphisme pour pouvoir l'appliquer aux suites infinies).

D'autre part la suite infinie obtenue est exactement “la suite de Hanoï” : ses préfixes de longueur $2^N–1$ sont les mouvements qui permettent de déplacer N disques.

Une autre méthode pour obtenir “simplement” cette suite est ce qu'on appelle un *automate fini*..



Le schéma qui précède est un exemple d'automate fini et plus précisément de 2-automate : un 2-automate est composé d'un ensemble fini d'*états*, (l'un des états est appelé *état initial*), de deux familles de *flèches* étiquetées 0 et 1 qui sont des applications de l'ensemble des états dans lui-même et d'une *fonction de sortie* de l'ensemble des états dans un ensemble fini (sur le dessin cette fonction a été écrite directement sur les états eux-mêmes) :

$\varphi(A)$ = quelconque,		$\varphi(B)$ = quelconque,			
$\varphi(C)$	$= \overline{c}$	$\varphi(D)$	$= \overline{c}$	$\varphi(E)$	$= \overline{a}$
$\varphi(F)$	$= \overline{a}$	$\varphi(G)$	$= \overline{b}$	$\varphi(H)$	$= \overline{b}$
$\varphi(J)$	$= a$	$\varphi(K)$	$= b$	$\varphi(L)$	$= b$
$\varphi(M)$	$= c$	$\varphi(N)$	$= c$	$\varphi(P)$	$= a$

Cet automate fonctionne comme suit : pour un entier (par exemple treize), on écrit treize en base 2 (treize = 1101) puis on nourrit l'automate des chiffres 1101 en commençant par la droite, partant de l'état initial et suivant les flèches. Ici on obtient l'état P, puis , en appliquant φ, la lettre a.

Si l'on fait la même chose successivement pour tous les entiers, on obtient :

un	= 1	∞	a	deux	= 10	∞	$\overline{c}$
trois	= 11	∞	b	quatre	= 100	∞	a
cinq	= 101	∞	c	six	= 110	∞	$\overline{b}$

…

Le lecteur attentif aura reconnu (et peut-être même prouvé) que la suite de lettres obtenue est notre suite de Hanoï).

Plus de détails sont donnés dans [2], remarquons simplement que, dans cette manière de calculer les mouvements, on n'a jamais besoin de retenir quels sont les disques qui se trouvent sur telle ou telle pile, et que pour calculer le *n*-ème coup, il suffit de connaître le développement binaire de *n*. On calcule alors, presque les yeux fermés, le coup en question (à condition bien sûr que les coups précédents aient été joués correctement).

le pliage de papier

Plions une feuille de papier, la partie gauche par dessus la partie droite. Recommençons l'opération en pliant toujours verticalement de la même façon (c'est-à-dire la partie gauche par dessus la partie droite).

En dépliant la feuille de manière à avoir des angles droits, on peut imaginer un petit personnage se promenant sur le bord supérieur de la feuille de papier (disons *de la gauche vers la droite*), qui va tourner à gauche (*G*) ou à droite (*D*) suivant les plis qu'il rencontre. Au début, il n'y a qu'un pli et il va tourner à droite (*D*). Après une deuxième opération on a cette fois-ci trois plis et notre personnage va tourner à droite, puis encore à droite, puis à gauche. En itérant cette procédure, on obtient successivement :

D
D D G
D D G D D G G
…

On constate que chaque mot obtenu commence par le mot précédent et qu'on obtient ainsi, après un nombre infini d'opérations, une suite infinie de D et de G.

D D G D D G G D D D G G D G G…

Le lecteur est invité à se munir d'une feuille de papier et à la plier comme indiqué.

Comment construire cette suite sans plier effectivement du papier ? Un résultat de “physique amusante” affirme d'ailleurs que, quelle que soit la taille du papier initial, on ne peut le plier plus de … sept fois !

Une première façon de procéder est la suivante. On considère l'alphabet à quatre lettres $\{a, b, c, d\}$ et on définit un morphisme par :

$a \mapsto a\,b$
$b \mapsto c\,b$
$c \mapsto a\,d$
$d \mapsto c\,d$

Comme plus haut, on peut itérer ce morphisme en partant de la lettre *a*. On obtient successivement :

a
a b
a b c b
a b c b a d c b
…

d'où une suite infinie

a b c b a d c b a b c d …

qui est un “point fixe” du morphisme (c'est-à-dire qui est invariante quand on lui applique le morphisme).

Prenons alors l'image lettre à lettre de cette suite infinie par l'application

$a \mapsto D$
$b \mapsto D$
$c \mapsto G$
$d \mapsto G$

nous obtenons la suite

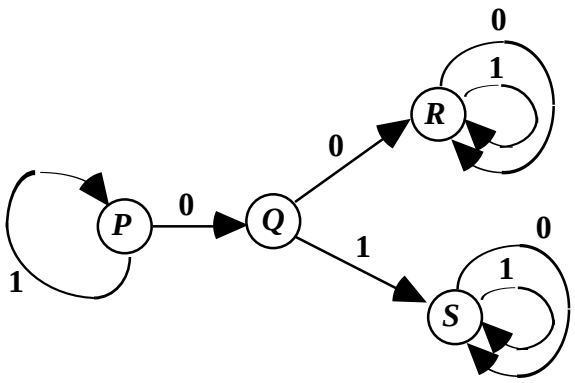
D D G D D G G D D D G G …

qui n'est autre, comme on peut le constater, mais aussi le démontrer, que la suite de pliage de papier.

Une autre manière d'engendrer cette suite est la suivante. On considère l'automate ci-dessous. Les états sont {*P*, *Q*, *R*, *S*), l'état initial étant *P*, les flèches sont étiquetées 0 et 1 et la fonction de sortie φ est donnée par :

$$\varphi(P) = \varphi(Q) = \varphi(R) = D$$
$$\varphi(S) = G$$

Notons qu'ici la suite des *G* et *D* est indexée par 0, 1, 2, … et non par 1, 2, … comme dans l'exemple précédent, de sorte que le terme d'indice 0 est *D*, celui d'indice 1 est *D*, celui d'indice 2 est *G*, etc.



Lorsque l'on nourrit l'automate avec le chiffre *n* en base 2 comme ci-dessus, on obtient le terme d'indice *n* de la suite de pliage de papier.

les suites infinies sans carré ou sans cube

Au début du siècle Thue [5], [6] posa la question suivante : est-il possible de construire une suite infinie sur deux lettres qui ne contienne pas de *carré*, c'est-à-dire qui ne contienne pas deux blocs consécutifs identiques. Par exemple 0011100111 est un carré (le carré de 00111).

La réponse est négative et il est très facile de s'en convaincre : soient 0 et 1 les deux lettres sur lesquelles on construit la suite.

Aux notations près on peut supposer que l'on commence par 0. Mais alors la suite ne peut commencer par 00 qui est un carré (le carré de 0), elle commence donc par 01.

L'élément suivant ne peut être un 1, car 011 contient le carré 11. La suite commence donc par 010.

Mais on aboutit alors à une impossibilité car si l'élément suivant est un 0, autrement dit si la suite commence par 0100, elle contient le carré 00, et si c'est un 1, la suite commence par 0101 qui est un carré.

En fait nous venons de prouver que, sur un alphabet à deux lettres, tout mot de longueur supérieure ou égale à quatre contient un carré.

Naturellement Thue ne s'en tient pas là et se demande s'il est possible de construire une suite sans carré sur trois lettres ou une suite sans cube sur deux lettres. Il ajoute, et l'on voit là la marque d'un vrai chercheur, que ce problème ne lui paraît pas susceptible d'applications immédiates, mais que sa difficulté même laisse espérer des retombées et qu'en tout état de cause il est intéressant de résoudre de tels problèmes même sans … arrière-pensée.

La suite sans cube sur deux lettres que Thue a construite et que nous décrivons ci-dessous fut retrouvée dans les années 20 par Morse dans un autre contexte et avait en fait été utilisée en 1851 par Prouhet comme nous l'expliquerons plus bas. Aussi cette suite est-elle maintenant appelée la suite de Prouhet-Thue-Morse.

Considérons l'alphabet {0, 1} et le morphisme défini par :

$$0 \mapsto 0\,1$$
$$1 \mapsto 1\,0$$

En itérant ce morphisme à partir de 0, on obtient successivement :

0
0 1
0 1 1 0
0 1 1 0 1 0 0 1
…

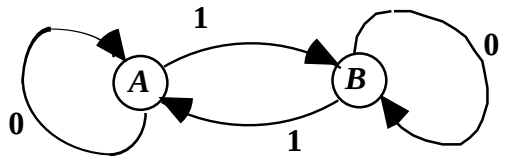
d'où une suite infinie, point fixe du morphisme :

0 1 1 0 1 0 0 1 1 0 0 1 0 1 1 0 1 0 0 1 0 1 1 0 …

Il est assez facile de montrer que cette suite ne contient ni 0 0 0 ni 1 1 1 ; il est plus difficile de prouver comme Thue que cette suite ne contient en fait aucun cube.

D'ailleurs Thue a prouvé un peu plus, il a montré que cette suite ne contient aucun *chevauchement*, c'est-à-dire aucun mot de la forme *wwx*, où *w* est un mot (fini) et *x* la première lettre de *w*. Le mot *ananas* par exemple contient le chevauchement *anana*, qui s'écrit *wwx* avec *w* = *an* et *x* = *a*. Il est clair qu'une suite infinie sans chevauchement ne contient *a fortiori* pas de cube.

Cette suite peut en fait être aussi engendrée par l'automate suivant, avec la fonction de sortie qui à l'état *A* associe 0 et à l'état *B* associe 1.



le problème de Prouhet-Tarry-Escott

Le problème de théorie des nombres abordé par Prouhet en 1851, et redécouvert par Tarry et Escott au début du siècle est le suivant.

On se donne un entier *N* supérieur à 1 et on considère l'ensemble d'entiers naturels :

$$A = \{1, 2, 3, \ldots, 2^N\}.$$

On cherche alors à partager A en deux sous-ensembles, notés I et **J**, qui ont les propriétés suivantes.

• Les deux ensembles I et **J** forment une partition de A : ils sont non vides, disjoints et leur réunion est l'ensemble A tout entier.
• Les ensembles I et **J** ont le même nombre d'éléments.
• La somme des éléments de I est égale à la somme des éléments de **J**.
• La somme des carrés des éléments de I est égale à la somme des carrés des éléments de **J**.
• La somme des cubes des éléments de I est égale à la somme des cubes des éléments de **J**.
• …
• La somme des puissances (N–1)-èmes des éléments de I est égale à la somme des puissances (N–1)-èmes des éléments de **J**.

En d'autres termes, les ensembles I et **J** doivent vérifier :

$$I \neq \emptyset,\ \mathbf{J} \neq \emptyset,\ I \cap \mathbf{J} = \emptyset,\ I \cup \mathbf{J} = A$$

$$\forall k = 0, 1, \ldots, N-1$$
$$\sum_{i \in I} i^k = \sum_{j \in J} j^k$$

Il est amusant que la suite de Morse donne une solution de ce problème, de la façon suivante. Prenons par exemple $N = 3$, donc $A = \{1, 2, \ldots, 8\}$.

Écrivons les huit premiers termes de la suite de Prouhet-Thue-Morse : 0 1 1 0 1 0 0 1, numérotons-les de un à huit.
Notons I l'ensemble des numéros en face de 0 et **J** l'ensemble des numéros en face de 1 :

0 1 1 0 1 0 0 1
1 2 3 4 5 6 7 8

de sorte que

$I = \{1, 4, 6, 7\}$ et $\mathbf{J} = \{2, 3, 5, 8\}$.

On vérifie alors aisément que I et **J** forment une partition de $A = \{1, 2, \ldots, 8\}$, puisque l'on a :

$$\text{Card } I = \text{Card } \mathbf{J} = 4$$
$$1 + 4 + 6 + 7 = 2 + 3 + 5 + 8 = 18$$
$$1^2 + 4^2 + 6^2 + 7^2 = 2^2 + 3^2 + 5^2 + 8^2 = 102$$

Notons qu'on conjecture qu'à l'échange près des ensembles I et **J** la solution décrite ci-dessus pour $N = 3$ (et dont le principe est valable pour N quelconque) est *unique* et que ce résultat semble difficile à obtenir.

automates et transcendance

Un joli résultat qui date de 1980 et est dû à Christol, Kamae, Mendès France et Rauzy [3] montre qu'une suite est engendrée par un automate fini, comme les suites décrites plus haut, si et seulement si la série formelle associée est algébrique sur le corps des fractions rationnelles à une indéterminée sur un corps fini. Décrire ce résultat plus précisément nous mènerait trop loin, mais nous allons ici aborder un autre résultat de transcendance concernant les suites automatiques.

Rappelons d'abord qu'un nombre réel est dit *algébrique* s'il est racine d'un polynôme à coefficients entiers, (c'est-à-dire s'il existe un polynôme à coefficients entiers qui s'annule lorsque l'on remplace la variable par ce nombre réel).

> Ainsi $\sqrt{2}$ est algébrique car c'est une racine du polynôme $X^2 - 2$. [c'est-à-dire $(\sqrt{2})^2 - 2 = 0$; si l'on remplace X par $\sqrt{2}$, $X^2 - 2$ devient nul.]
>
> De même $\sqrt{2} + \sqrt{3}$ est racine du polynôme $X^4 - 10X^2 + 1$.
>
> Naturellement les nombres rationnels (les fractions) sont algébriques ; par exemple 22/7 est racine du polynôme $7X - 22$.

Un nombre qui n'est pas algébrique est dit *transcendant*. Ainsi, il a été démontré que le nombre π et le nombre e (base des logarithmes népériens) sont transcendants. C'est d'ailleurs parce que π est transcendant que la quadrature du cercle est impossible.

Si l'on se donne une suite de 0 et de 1, on peut lui associer aisément un nombre réel en mettant une virgule, disons après le premier terme, et en considérant le nombre réel dont cette expression est le développement en base 2. Par exemple, en partant de la suite de Prouhet-Thue-Morse

0 1 1 0 1 0 0 1 1 0 0 1 0 1 1 0 …,

on obtient le nombre réel (écrit en base 2)

0,110100110010110…

On peut démontrer que ce nombre est transcendant. Plus généralement, il se produit le phénomène suivant : si on prend une suite engendrée par un automate comme ci-dessus,

ou bien cette suite est *ultimement périodique*, c'est-à-dire périodique à partir d'un certain rang, et le nombre réel associé est rationnel, ou bien elle n'est pas ultimement périodique et ce nombre est transcendant. Ceci est un résultat dû à Loxton et van der Poorten (en fait la preuve n'a pas encore été donnée dans le cas le plus général, mais il est très vraisemblable que ce résultat est vrai.)

En d'autres termes, plier du papier est une activité qui permet de fabriquer des nombres transcendants !

On peut aussi énoncer la contraposée du résultat de Loxton et van der Poorten : les chiffres d'un nombre algébrique irrationnel (c'est-à-dire qui n'est pas rationnel) comme $\sqrt{2}$ ne peuvent pas être engendrés par un automate fini.

On conjecture une propriété beaucoup plus forte sur $\sqrt{2}$, c'est que ce nombre est *normal* dans toute base : par exemple dire que $\sqrt{2}$ est normal en base 10 signifie que tout chiffre apparaît avec la fréquence 1/10, que tout bloc de deux chiffres (comme 41) apparaît avec la fréquence 1/100, tout bloc de trois chiffres avec la fréquence 1/1000, etc.

En d'autres termes aucun bloc n'est privilégié, un peu comme pour un nombre “au hasard”.

Pour donner une idée de la difficulté de la question par rapport aux résultats connus, non seulement on ne sait pas si, comme on le soupçonne, $\sqrt{2}$ est normal, mais on ne sait même pas si chaque chiffre apparaît avec la fréquence 1/10. En fait ceci impliquerait que chaque chiffre apparaît une infinité de fois et *on ne sait même pas démontrer que le chiffre* 4*, par exemple, apparaît une infinité de fois* !

références

[1] J.-P. Allouche, *Automates finis en théorie des nombres*, Expo. Math. **5**, 1987, 239–266.

[2] J.-P. Allouche et F. Dress, *Tours de Hanoï et automates*, RAIRO, Informatique Théorique et Applications **24**, **1**, 1990, 1-15. ##

[3] G. Christol, T. Kamae, M. Mendès France et G. Rauzy, *Suites algébriques, automates et substitutions*, Bull. Soc. math. France **108**, 1980, 401–419.

[4] M. Dekking, M. Mendès France and A. van der Poorten, *FOLDS !*, Math. Intell. **4**, 1982, 130–138, 173–181; 190–195.

[5] A. Thue, *Über unendliche Zeichenreihen*, Norske vid. Selsk. Skr., I. Mat. Nat. Kl., Christiana **1**, 1906, 1–22.

[6] A. Thue, *Über die gegenseitige Lage gleicher Teilegewisser Zeichenreihen* , Norske vid. Selsk. Skr., I. Mat. Nat. Kl., Christiana **7**, 1912, 1–67.

[NDLR : signalons également l'existence du « Mouvement Français des Plieurs de Papier », 56 rue Coriolis, 75012 Paris, qui édite la revue « Le pli ».]

Annexe — Jean-Pierre Ressayre — **description en 4 points d'un automate** (à lire en se reportant à l'exemple cité en bas, à gauche : automate *PARITÉ*)

1. Un automate A comporte un ruban divisé en cases, appelé son ENTRÉE ; dans chaque case, nous pouvons ***écrire*** un symbole (chiffre ou lettre), et A peut le ***lire***. Ces symboles appartiennent à un ensemble fini appelé ALPHABET de I, et noté S_A.

On appelle MOTS de S_A ***toutes*** les suites finies de symboles de S_A. Bref : sur l'entrée de A, nous écrirons un mot de S_A (de longueur quelconque) ; alors A lit ce mot (case par case, de la gauche vers la droite).

2. A comporte un ÉTAT INITIAL noté e_0 dans lequel se trouve A au départ ; mais en lisant son entrée, A peut passer dans d'autres états, qui forment un ensemble fini noté E_A.

3. A l'automate A est associée une fonction notée f_A, qui, à tout couple (e, s) où e est dans E_A, s dans S_A, fait correspondre un état $f_A(e, s)$ de E_A. Et le point essentiel du fonctionnement de A est :

(†) **Si A se trouve dans l'état e pendant qu'il lit le symbole s, alors A passe dans l'état $f_A(e, s)$ avant de lire le symbole suivant.**

Ainsi, quand A a lu tous les symboles d'un mot M, par application répétée de (†) à partir de l'état e_0, A parvient dans un état qu'on note $f_A(e_0, M)$.

4. Il y a un sous-ensemble fixé de E_A, dont les éléments sont appelés les ÉTATS ACCEPTANTS de A ; et on dit que A ACCEPTE le mot de S_A si $f_A(e_0, M)$ est l'un de ces états acceptants.

C'est tout. Donc un "automate" A ne fait que "lire" les mots de S_A et "dire" s'il les accepte ou non. Mais il y a des variantes qui vérifient (1) + (2) + (3), mais à la place de (4) ont un autre dispositif. Par exemple : ils ont un deuxième ruban appelé ***sortie*** de A, sur lequel ils ***écrivent***, au fur et à mesure qu'ils lisent leur entrée.

Exemple : A = l'automate appelé PARITÉ

1.— S_A comprend "0" et "1" ; voici A avec le mot M = "10010" sur son entrée :

au départ	***après avoir lu M***
e_0	$f_A(e_0, M)$
10010	10010

2.— E_A comprend deux états : "+" et "–" ; l'état initial e_0 est "+".

3.— $f_A(+, 0) = +$ et $f_A(–, 0) = –$; c'est-à-dire que si A lit un 0, alors A ne change ***pas*** d'état. $f_A(+, 1) = –$ et $f_A(–, 1) = +$; c'est-à-dire que si A lit un 1, alors A ***change*** d'état.

Lecture du mot M :

	étape 1	étape 2	étape 3	étape 4	étape 5	étape 6	éta 7	étape 8
A est dans l'état :	+	–	–	–	+	+		
il lit alors :	1	0	0	1	0			

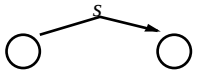
4.— A comporte + comme seul état acceptant ; donc puisque ci-dessus : $f_A(e_0, M) = +$, le mot M est accepté par A.

Remarque : on a une description **complète** de A et de ce qu'il fait, lorsqu'on connaît S_A, e_0, E_A, f_A, et l'ensemble $E_A^+ \subset E_A$ des états acceptants

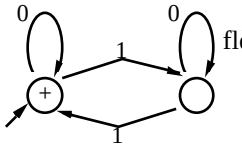
LE GRAPHE de *A*, *G*(*A*)

1.— C'est un dessin où chaque état $e \in E_A$ est figuré par un rond : O ; $G(A)$ a donc autant de ronds que E_A a d'éléments. Parmi tous ces ronds, celui qui représente l'état ***initial*** e_0 est distingué par une flèche : →O. Et on représente les états ***acceptants*** en mettant un "+" : ⊕.

2.— Pour représenter "$f_A(e, s) = e'$", $G(A)$ comporte une flèche qui ***part*** du rond représentant e qui ***arrive*** au rond représentant e', et qui est ***étiquetée*** par s :



Exemple : le graphe G(PARITÉ)

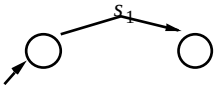


flèche qui représente $f_A(–, 0) = –$

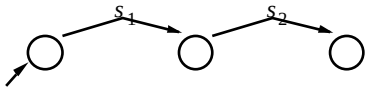
flèche qui représente $f_A(–, 1) = –$

3.— Voici le ***chemin dans G(A)*** associé au mot M de S_A :

• Ce chemin part de l'état initial →O, et si s_1 est le ***premier*** symbole de M, il emprunte la flèche étiquetée s_1 qui part de →O :



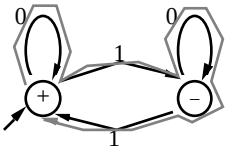
• Cette flèche arrive à un rond O ; si s_2 est le ***deuxième*** symbole de M, le chemin emprunte la flèche étiquetée s_2 qui part de ce O :



• etc … ! Ainsi, le chemin aboutit ***au rond qui représente*** $f_A(e_0, M)$.

4.— ***Moralité*** : si je vous donne $G(A)$ et que vous voulez savoir si un mot M est accepté par A, vous n'avez qu'à ***parcourir le chemin associé à M*** dans $G(A)$. Si ce chemin se termine à un rond qui contient un +, alors M est accepté …

Exemple : chemin associé à M = "0101" dans G(PARITÉ)

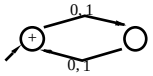


il aboutit à "+" donc : M accepté.

Nota Bene : si $f_A(e, s) = f_A(e, s') = e'$ dans A, alors la flèche qui va du rond e au rond e' dans $G(A)$ est étiquetée par les ***deux*** symboles s et s'.

QUESTIONS :

1) Quels sont tous les mots acceptés par PARITÉ ?

2) Même question pour A tel que $G(A)$ = 

3) Soit G un dessin avec un ➚O, et avec des ronds joints par des flèches étiquetées par des symboles de l'ensemble S. Dans quel cas y-a-t-il un automate A tel que $G(A) = G$?

4) Trouver un automate A tel que $S_A = \{0, 1\}$ et E_A est l'ensemble de tous les mots qui ne contiennent ***pas*** la suite 10.

5) Même question en remplaçant 10 par 1100.

TROIS TYPES DE QUESTIONS :

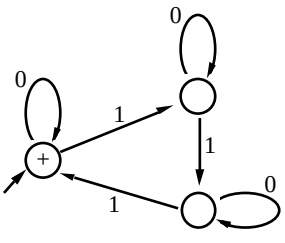
I— Je donne un automate A ; quels sont les mots qu'il accepte ?

II— Je donne un ensemble de mots L ; construisez un automate qui accepte précisément les mots de L.

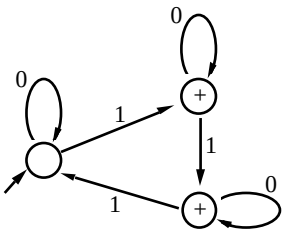
III— Je donne un automate A ; trouvez un automate B qui accepte les mêmes mots, mais qui est plus simple : E_B plus petit que E_A.

Questions de type I — Je donne un automate A ; quels sont les mots qu'il accepte ?

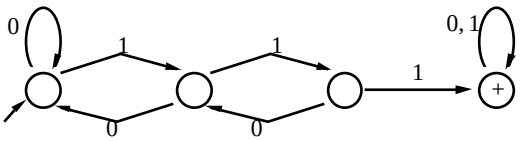
1. Si $G(A)$ =



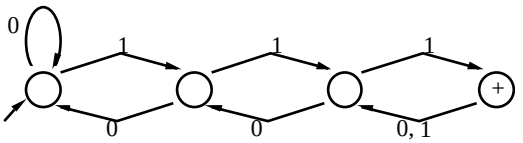
2. Si $G(A)$ =



3. Si $G(A)$ =



4. Si $G(A)$ =



Questions de type II — Je donne un ensemble de mots L ; construisez un automate qui accepte précisément les mots de L.

1. L = les mots contenant au moins trois "1".
2. L = les mots contenant au moins un "111" (noté : 1^3).
3. Questions 1, 2 en remplaçant "au moins" par "exactement".
4. L = les mots de la forme "$0^{n_1}10^{n_2}110^{n_3}111$", où n_1, n_2 et n_3 sont des entiers > 0.
5. L = les mots qui contiennent un "0101" ou un "1010", mais ***pas*** un "0110".
6. L = les mots M tels que ***si*** M contient un "10001", ***alors*** M ne contient ***pas*** de "101010".
7. L = si M contient 101110, ***alors*** M contient 01010 ; ***et de plus*** si M contient 00100, ***alors*** M contient 1^5.

CONCOURS N° 1

Vous vous posez à ***vous-même*** la question de types I ou II de votre choix ; vous la résolvez. Alors vous la posez ***aux autres*** (et ils essaient de trouver solution plus simple que vous).

CONCOURS N° 2

Répondre le premier aux questions (pas évidentes) qui suivent :

I.1.— Construire un automate A qui calcule ***le reste de la division par 2*** de l'entier :
$a_0 + a_1\,10 + a_2\,10^2 + ... + a_k\,10^k$ (cet entier est représenté sur l'entrée de A par le mot "$a_0 \ldots a_k$" de l'alphabet $\{0, 1, \ldots, 9\}$).

I.2.— Même question en remplaçant "2" par "3", en remplaçant "2" par "5".

I.3.— Mêmes questions en remplaçant la base 10 et l'alphabet $\{0, \ldots, 9\}$ par la base 2 et l'alphabet $\{0, 1\}$.

I.4.— Mêmes questions en écrivant : "$a_k\, a_{k-1} \ldots a_0$" au lieu de "$a_0\, a_1 \ldots a_k$" sur l'entrée.

II.— Montrez qu'il n'existe ***pas*** d'automate qui accepte précisément les mots 0^n1^n, où n est un entier > 0.

III.— A, B deux automates avec même alphabet : $S_A = S_B$;

> III.1.— Construire un automate C qui accepte les mots acceptés par A ***et*** par B (par A ***ou*** B).
> III.2 - Montrer que s'il existe un mot accepté par A mais pas par B, alors il en existe un de longueur $\leq n.p$; (n, p = nombre d'états de A, B). En déduire une méthode pour savoir si deux automates A_1, A_2 acceptent exactement les mêmes mots.

IV.— Méthode pour construire D qui accepte les mots "$s_1 \ldots s_n t_1 \ldots t_p$" ; où "$s_1 \ldots s_n$" est accepté par A et "$t_1 \ldots t_p$" par B.