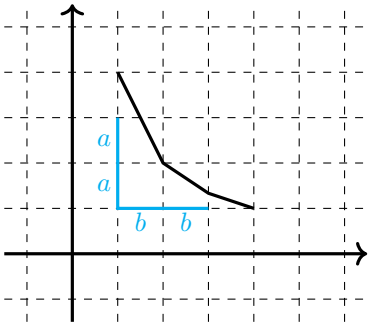
Hyperboles et mots

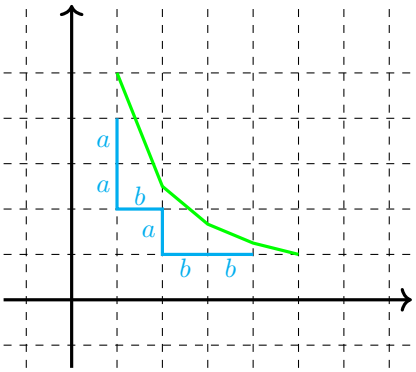
Denise Vella-Chemla

16.2.17

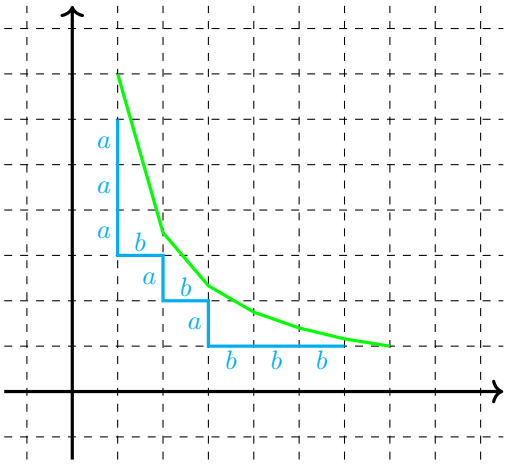
Un mot de Christoffel [1] est un mot sur un alphabet de deux lettres qui "colle au plus près" à une courbe par segments liant des points discrets (par le dessus ou par le dessous). Voyons des exemples :
- *mot de l'hyperbole discrétisée associé à* $n = 4$ *: aabb.*



- *mot de l'hyperbole discrétisée associé à* $n = 5$ *: aababb.*

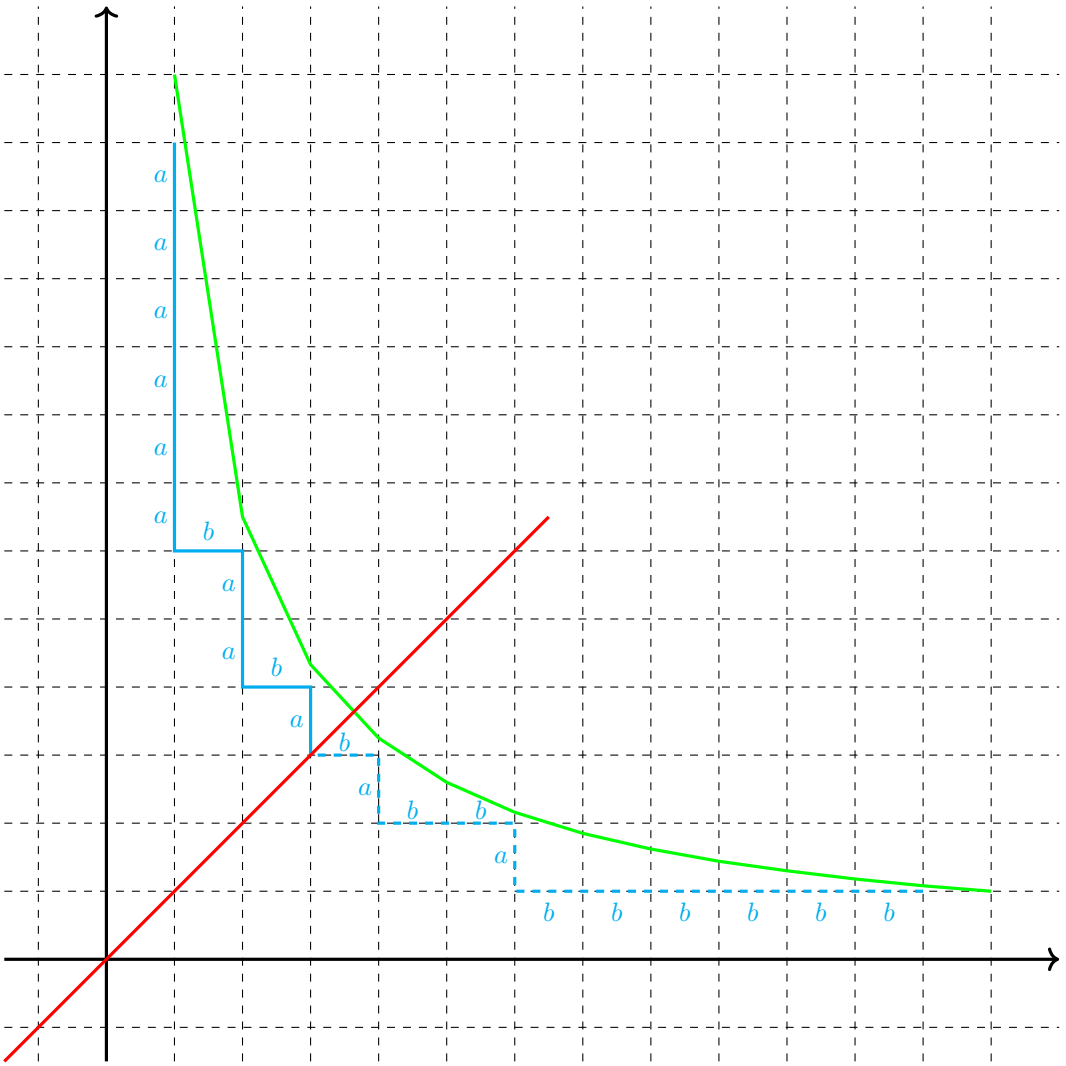


- *mot de l'hyperbole discrétisée associé à* $n = 7$ *: aaababababb.*



Du fait de la symétrie des hyperboles, qui se traduit en mot par le fait que les mots sont de longueur paire et ont leur fin symétrique de leur début à inversion des lettres près, on ne conserve comme mot associé à un nombre que le début des mots présentés ici (un mot de $n-2$ lettres pour caractériser n ; on montre cette symétrie sur le dernier graphique ci-dessous par la diagonale rouge et la fin du mot en pointillé).

- *mot de l'hyperbole discrétisée associé à* $n = 13$ *: aaaaaabaaba.*



On calcule par programme les mots des 500 premiers entiers naturels, à la recherche de régularités. On fournit ci-dessous les mots des nombres jusqu'à 50.

3	a
4	aa
5	aab
6	$aaab$
7	$aaaba$
8	$aaaaba$
9	$aaaabaa$
10	$aaaaabab$
11	$aaaaabaab$
12	$aaaaaabaab$
13	$aaaaaabaaba$
14	$aaaaaaabaaba$
15	$aaaaaaabaaaba$
16	$aaaaaaaabaabaa$
17	$aaaaaaaabaaabab$
18	$aaaaaaaaabaaabab$
19	$aaaaaaaaabaaabaab$
20	$aaaaaaaaaabaaabaab$
21	$aaaaaaaaaabaaaababa$
22	$aaaaaaaaaaabaaabaaba$
23	$aaaaaaaaaaabaaaabaaba$
24	$aaaaaaaaaaaabaaaabaaba$
25	$aaaaaaaaaaaabaaaabaabaa$
26	$aaaaaaaaaaaaabaaaabaabab$
27	$aaaaaaaaaaaaabaaaaabaabab$
28	$aaaaaaaaaaaaaabaaaabaaabab$
29	$aaaaaaaaaaaaaabaaaaabaabaab$
30	$aaaaaaaaaaaaaaabaaaaabaabaab$
31	$aaaaaaaaaaaaaaabaaaaabaaababa$
32	$aaaaaaaaaaaaaaaabaaaaabaaababa$
33	$aaaaaaaaaaaaaaaabaaaaaabaabaaba$
34	$aaaaaaaaaaaaaaaaabaaaaabaaabaaba$
35	$aaaaaaaaaaaaaaaaabaaaaaabaaabaaba$
36	$aaaaaaaaaaaaaaaaaabaaaaaabaaababaa$
37	$aaaaaaaaaaaaaaaaaabaaaaaabaaabaabab$
38	$aaaaaaaaaaaaaaaaaaabaaaaaabaaabaabab$
39	$aaaaaaaaaaaaaaaaaaabaaaaaaabaaabaabab$
40	$aaaaaaaaaaaaaaaaaaaabaaaaaabaaaabaabab$
41	$aaaaaaaaaaaaaaaaaaaabaaaaaaabaaabaabaab$
42	$aaaaaaaaaaaaaaaaaaaaabaaaaaaabaaabaabaab$
43	$aaaaaaaaaaaaaaaaaaaaabaaaaaaabaaaabaababa$
44	$aaaaaaaaaaaaaaaaaaaaaabaaaaaaabaaaabaababa$
45	$aaaaaaaaaaaaaaaaaaaaaabaaaaaaaabaaabaaababa$
46	$aaaaaaaaaaaaaaaaaaaaaaabaaaaaaabaaaabaabaaba$
47	$aaaaaaaaaaaaaaaaaaaaaaabaaaaaaaabaaaabaabaaba$
48	$aaaaaaaaaaaaaaaaaaaaaaaabaaaaaaaabaaaabaabaaba$
49	$aaaaaaaaaaaaaaaaaaaaaaaabaaaaaaaabaaaabaaababaa$
50	$aaaaaaaaaaaaaaaaaaaaaaaabaaaaaaaabaaaabaaababab$

Les $n/2$ premières lettres sont toutes des a. Dans la suite, les lettres b servent de séparateurs. On écrit un nouveau programme qui compte les nombres de lettres a dans les paquets de a successifs. Le résultat de ce programme est fourni pour les nombres jusqu'à 100 en annexe 2. Ci-dessous, les nombres de a pour les mots du tableau précédent (en deuxième colonne dans chaque sous-partie du tableau, p signifie premier et c composé).

			11	p	2	21	c	4 1 1	31	p	5 3 1 1	41	p	7 3 2 2
			12	c	2	22	c	3 2 1	32	c	5 3 1 1	42	c	7 3 2 2
3	p		13	p	2 1	23	p	4 2 1	33	c	6 2 2 1	43	p	7 4 2 1 1
4	c		14	c	2 1	24	c	4 2 1	34	c	5 3 2 1	44	c	7 4 2 1 1
5	p		15	c	3 1	25	c	4 2 2	35	c	6 3 2 1	45	c	8 3 3 1 1
6	c		16	c	2 2	26	c	4 2 1	36	c	6 3 1 2	46	c	7 4 2 2 1
7	p	1	17	p	3 1	27	c	5 2 1	37	p	6 3 2 1	47	p	8 4 2 2 1
8	c	1	18	c	3 1	28	c	4 3 1	38	c	6 3 2 1	48	c	8 4 2 2 1
9	c	2	19	p	3 2	29	p	5 2 2	39	c	7 3 2 1	49	c	8 4 3 1 2
10	c	1	20	c	3 2	30	c	5 2 2	40	c	6 4 2 1	50	c	8 4 3 1 1

On découvre que les nombres de a du mot d'un nombre impair premier sont tous identiques aux nombres de a du mot de son successeur, et que cela n'est pas le cas pour les nombres de a du mot d'un nombre impair composé. Cela est dû au fait que l'hyperbole d'un nombre premier ne passe que par des points non-entiers du réseau (hormis les points triviaux $(1, n)$ et $(n, 1)$). L'hyperbole du successeur d'un nombre premier, quant à elle, passe par des points entiers du réseau mais cela ne suffit pas à modifier le mot de Christoffel qui ne se voit qu'ajouter un a au tout début (dans la première partie de $n/2$ lettres qu'on a choisi de négliger). Le comptage des lettres fait sauter cela aux yeux alors qu'il est plus difficile de le constater directement sur les suites confuses de a et b du tableau en page précédente.

On est ainsi à nouveau (cf le travail effectué autour de la conjecture de Goldbach) en train de compter des nombres d'assertions logiques : ici, elles sont de la forme : $n \leqslant (x+1)y$.

Voyons ces assertions logiques sur des exemples : 7, nombre premier, a le même mot constitué d'une seule lettre a que son successeur 8 parce que $7 \leqslant (2+1).3$ est une inégalité de même sens que $8 \leqslant (2+1).3$.

De même, 11, nombre premier, a le même mot constitué de 2 lettres a que son successeur 12 parce que les inégalités $11 \leqslant (2+1).5$ et $12 \leqslant (2+1).5$ sont de même sens, de même que les inégalités $11 \leqslant (2+1).4$ et $12 \leqslant (2+1).4$.

Par contre, pour ne prendre qu'un exemple, 27 est composé car son mot est différent du mot de son successeur. Il y a passage d'une lettre a à une lettre b par exemple du fait de l'inversion de sens des 2 inégalités $27 \leqslant (2+1).9$ et $28 > (2+1).9$.

Les scans de la première et la dernière page de l'exécution du programme fourni en annexe présentent :
- la découverte de l'égalité des mots des nombres premiers inférieurs à 60 et du mot de leur successeur[1] ;
- la détection des inversions de lettres pour les nombres composés de 450 à 500[2].

La méthode proposée procède en trois étapes :
- la première consiste à calculer les assertions logiques pour un nombre donné ainsi que pour son successeur ;
- la seconde compte les assertions positives successives (les longueurs des paquets de lettres a) ;
- la troisième consiste à comparer les nombres obtenus.

La première étape est le calcul de l'application de $\mathbb{N}^3$ dans $\mathbb{B}$ qui associe à tout triplet de nombres (n, x, y) la valeur booléenne de l'inégalité $n \leqslant (x+1)y$.
La comparaison du sens des inégalités (i.e. de deux booléens) est réalisée sur l'espace entier par une application de $\mathbb{B} \times \mathbb{B}$ dans $\mathbb{B}$.
L'opération d'égalité $b1 = b2$ qui associe à 2 booléens $b1$ et $b2$ un troisième booléen b qui vaut 1 si $b1$ et $b2$ sont tous deux égaux à 1 ou bien tous deux égaux à 0 s'écrit :

$$(b1 = b2) = (\neg b1 \vee b2) \wedge (\neg b2 \vee b1).$$

Le comptage des longueurs des séquences de a de la seconde étape s'effectue classiquement (arithmétique de Peano). Aux nombres compris entre $n^2 + n + 1$ et $n^2 + 3n + 2$ (on avait remarqué que le nombre de paquets de a était augmenté de 1 selon des cycles de plus en plus longs, liés à la suite des nombres pairs successifs, d'où ces formules), sont associés $n - 1$ nombres qui sont les tailles des paquets successifs de lettres a.

La comparaison de la troisième étape est le calcul d'une application de $\mathbb{N}^2$ dans $\mathbb{B}$ qui associe à tout couple de nombres (x, y) la valeur booléenne de l'égalité $x = y$.

On pense avoir fourni ici tous les ingrédients nécessaires à une manière plutôt "syntaxique" de tester la primalité des entiers.

Bibliographie :

[1] Jean Berstel, Aaron Lauve, Christophe Reutenauer, Franco Saliola, *Combinatorics on Words : Christoffel Words and Repetitions in Words*, 2008.

[1] http://denise.vella.chemla.free.fr/2017fev15-1.jpg
[2] http://denise.vella.chemla.free.fr/2017fev15-2.jpg

Annexe 1 : programme de calcul des mots

```
1
2  #include <iostream>
3  #include <stdio.h>
4
5  int prime(int atester) {
6    bool pastrouve = true;
7    unsigned long k = 2;
8
9    if (atester == 1) return 0;
10   if (atester == 2) return 1;
11   if (atester == 3) return 1;
12   if (atester == 5) return 1;
13   if (atester == 7) return 1;
14   while (pastrouve) {
15     if ((k * k) > atester) return 1;
16     else
17       if ((atester % k) == 0) {
18         return 0 ;
19       }
20       else k++;
21   }
22 }
23
24 int main(int argc, char* argv[]) {
25   int n, i, xcourant, ycourant, xa, xb ;
26   float res ;
27
28   for (n = 3 ; n <= 500 ; ++n) {
29       printf("%5d : ", n) ;
30       if (prime(n)) std::cout << "(p) " ; else std::cout << "(c) " ;
31       xcourant = 1 ; ycourant = n-1 ;
32       xa = 0 ; xb = 0 ;
33       for (i = 1 ; i <= n-1 ; ++i) {
34          res = (float) n-(((float) xcourant + 1.0) * (float) ycourant) ;
35          if (res > 0.0) {
36            std::cout << "b" ;
37            if (i > n/2+1) std::cout << xa << " " ;
38            xcourant = xcourant+1 ;
39            xa = 0 ; xb = xb+1 ;
40          }
41          else {
42            if (i > n/2) std::cout << "a" ;
43            ycourant = ycourant-1 ;
44            xb = 0 ; xa = xa+1 ;
45          }
46       }
47       std::cout << "\n" ;
48   }
49 }
```

Annexe 2 : résultat du programme de calcul des mots

			26	c	4 2 1	51	c	9 4 2 2 1	76	c	12 7 3 3 2 1 1
			27	c	5 2 1	52	c	8 5 2 2 1	77	c	13 6 4 3 2 1 1
3	p		28	c	4 3 1	53	p	9 4 3 2 1	78	c	13 6 4 3 1 2 1
4	c		29	p	5 2 2	54	c	9 4 3 2 1	79	p	13 7 4 2 2 2 1
5	p		30	c	5 2 2	55	c	9 5 3 1 2	80	c	13 7 4 2 2 2 1
6	c		31	p	5 3 1 1	56	c	9 5 2 2 2	81	c	14 6 4 3 2 1 2
7	p	1	32	c	5 3 1 1	57	c	10 4 3 2 1 1	82	c	13 7 4 3 2 1 1
8	c	1	33	c	6 2 2 1	58	c	9 5 3 2 1 1	83	p	14 7 4 3 2 1 1
9	c	2	34	c	5 3 2 1	59	p	10 5 3 2 1 1	84	c	14 7 4 3 2 1 1
10	c	1	35	c	6 3 2 1	60	c	10 5 3 2 1 1	85	c	14 7 5 2 2 2 1
11	p	2	36	c	6 3 1 2	61	p	10 5 3 2 2 1	86	c	14 7 4 3 2 2 1
12	c	2	37	p	6 3 2 1	62	c	10 5 3 2 2 1	87	c	15 7 4 3 2 2 1
13	p	2 1	38	c	6 3 2 1	63	c	11 5 3 2 2 1	88	c	14 8 4 3 2 2 1
14	c	2 1	39	c	7 3 2 1	64	c	10 6 3 2 1 2	89	p	15 7 5 3 2 1 2
15	c	3 1	40	c	6 4 2 1	65	c	11 5 4 2 1 1	90	c	15 7 5 3 2 1 2
16	c	2 2	41	p	7 3 2 2	66	c	11 5 3 3 1 1	91	c	15 8 4 3 3 1 1 1
17	p	3 1	42	c	7 3 2 2	67	p	11 6 3 2 2 1	92	c	15 8 4 3 2 2 1 1
18	c	3 1	43	p	7 4 2 1 1	68	c	11 6 3 2 2 1	93	c	16 7 5 3 2 2 1 1
19	p	3 2	44	c	7 4 2 1 1	69	c	12 5 4 2 2 1	94	c	15 8 5 3 2 2 1 1
20	c	3 2	45	c	8 3 3 1 1	70	c	11 6 4 2 2 1	95	c	16 8 5 3 2 2 1 1
21	c	4 1 1	46	c	7 4 2 2 1	71	p	12 6 3 3 1 2	96	c	16 8 4 4 2 2 1 1
22	c	3 2 1	47	p	8 4 2 2 1	72	c	12 6 3 3 1 2	97	p	16 8 5 3 3 1 2 1
23	p	4 2 1	48	c	8 4 2 2 1	73	p	12 6 4 2 2 1 1	98	c	16 8 5 3 3 1 2 1
24	c	4 2 1	49	c	8 4 3 1 2	74	c	12 6 4 2 2 1 1	99	c	17 8 5 3 2 2 2 1
25	c	4 2 2	50	c	8 4 3 1 1	75	c	13 6 4 2 2 1 1	100	c	16 9 5 3 2 2 1 2