# Initiation à la combinatoire des mots - GMIN314

Pascal Ochem, Gwenaël Richomme, Patrice Séébold

gwenael.richomme@lirmm.fr

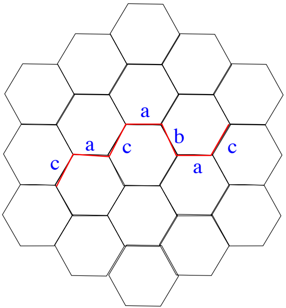
Cours 1

# Les transparents de la réunion de rentrée

# Contexte ?

Mot = un objet de base en modélisation
utile dans de nombreux domaines :

- langages formels, automates,
- compilateurs,
- compression,
- algo du texte et du génôme,
- gestion de file d'attente,
- placement de robots,
- théorie des nombres,
- ...




Chemin dans un graphe

# Contexte ? (suite)

L'étude des mots peut permettre :

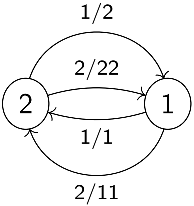
- une meilleure compréhension des objets modélisés,
- une aide à l'analyse d'algorithmes,
- principe pour de nouveaux algorithmes,
- ...

| | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Texte | a | n | e | a | n | a | n | u | a | n | a | n | a | n | a | s | a | n | e |
| Motif | | | | | | a | n | a | n | a | s | | | | | | | | |
| | | | | | | o | o | x | | | | | | | | | | | |

# Contexte ? (fin)

Questions nous intéressant plus particulièrement :

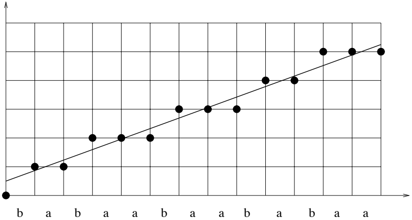
- études de familles de mots aux propriétés extrêmales,
- mécanismes pour engendrer des mots et étude des mots obtenus,
- existence de mots avec des propriétés particulières notamment évitant des motifs,
- résolution de problèmes ouverts,
- ...




Kolakoski 221121221 ...
n'a pas livré tous ses secrets

# Contenu prévisionnel du cours

1. Exemples de liaisons avec d'autres domaines
2. Résultats de base sur les mots finis
3. Mots infinis
    - mécanismes pour les engendrer : morphismes, transducteurs, ...
    - classification par le nombre de facteurs
    - mots sturmiens



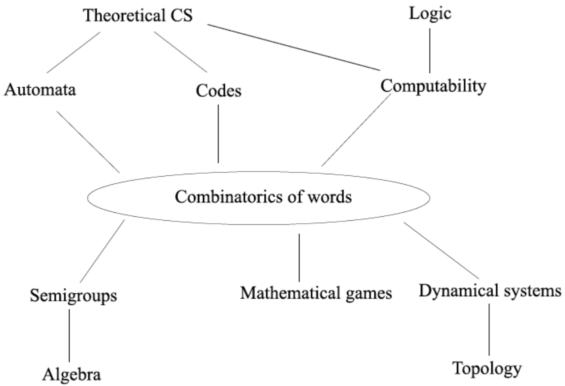

4. Mots et évitabilité de motifs

# Organisation

- Équipe pédagogique : P. Ochem, G. Richomme, P. Séébold
- Chaque séance : cours + exos
- MCC : examen (14 points) + devoirs (6 points)

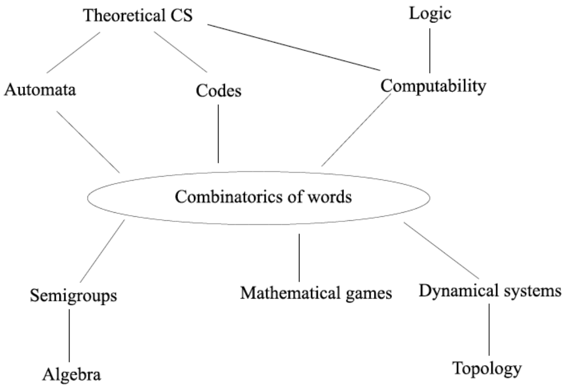
# Quelques liens avec d'autres domaines

# Des liens



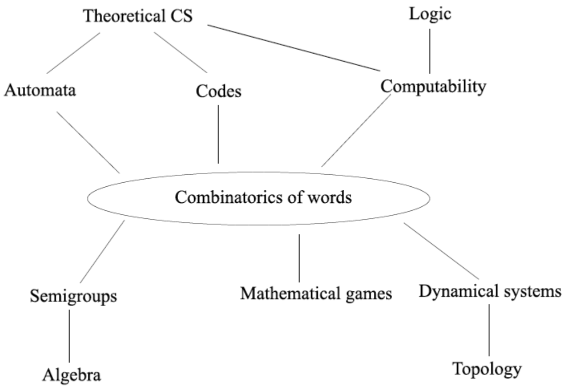

J. Karhumäki, Combinatorics of words
notes de cours, vue sur le web le 16/09/2014
http://www.utu.fi/en/units/sci/units/math/staff/Pages/karhumak.aspx

# Des liens



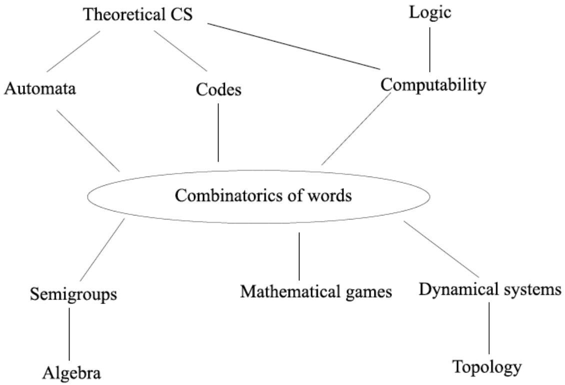

"Automata" : plus généralement théorie des langages formels
et ses applications (compilateurs)

# Des liens



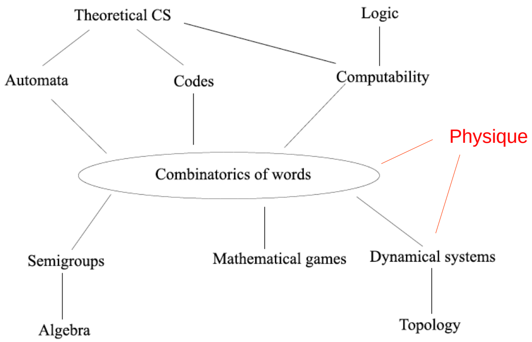

"Codes" : codes à longueurs variables
Morphismes injectifs = actions de codage

# Des liens



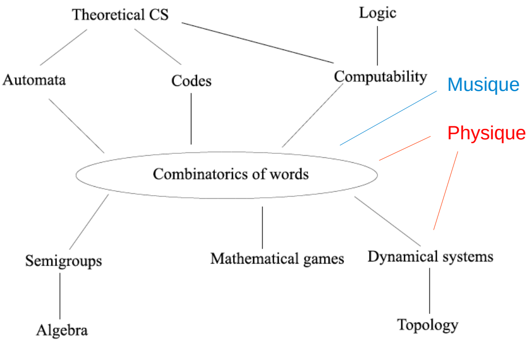

Algebra : exemple célèbre "Problème de Burnside"
*If S is a finitely generated semigroup and if each of its elements generate a finite semigroup, is G itself finite ?*
Non ! Preuve utilise mots sans carré.

# Des liens

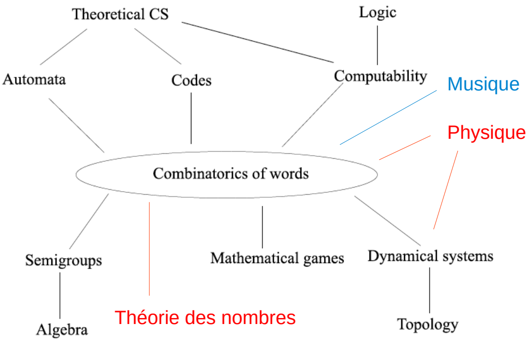



Physique : liens entre quasi-cristaux et mots sturmiens

# Des liens

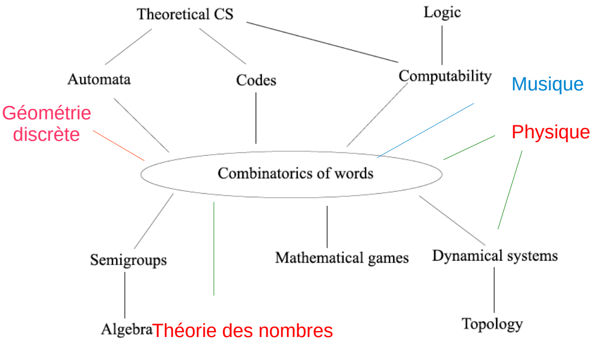

Theoretical CS
Logic
Automata
Codes
Computability
Musique
Physique
Combinatorics of words
Semigroups
Mathematical games
Dynamical systems
Algebra
Topology

# Des liens




Théorie des nombres, systèmes de numération
Nombres rationnels = mots dont le développement décimal est ultimement périodique

# Des liens



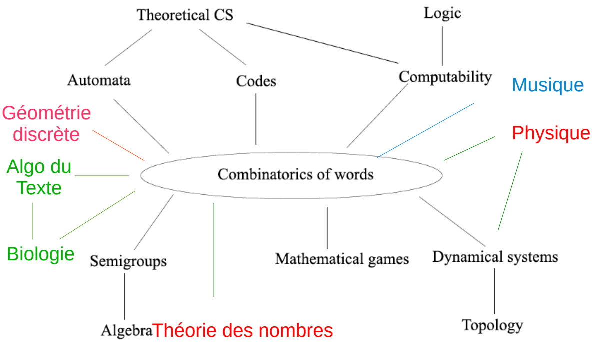

Géométrie discrète

Exemple historique : codage de segments = mots équilibrés ;

codage de droite = mots sturmiens

Autres approches récentes : codage de plan, mots tangents, ...

# Des liens




Biologie :

- génôme,
- modélisation croissance plantes (L-systèmes),
- . . . ,

# Au sujet de l'algorithmique du texte

J. Berstel, école jeunes chercheurs Marne la Vallée 2004 :
« l'algorithmique du texte est le premier étage d'une fusée qui en comporte 3, le deuxième étant l'algorithmique du texte, le troisième étant constitué par les applications (biologie, traitement de texte (XML, grep, ...)) »

# Au sujet de l'algorithmique du texte

J. Berstel, école jeunes chercheurs Marne la Vallée 2004 :
« l'algorithmique du texte est le premier étage d'une fusée qui en comporte 3, le deuxième étant l'algorithmique du texte, le troisième étant constitué par les applications (biologie, traitement de texte (XML, grep, ...)) »

- principes d'algorithmes
  Exemple : bords disjoints dans Knuth, Morris, Pratt
- analyse d'algorithmes
  Exemple : les mots de Fibonacci permettent d'obtenir le délai maximal entre deux décalages de fenêtre dans KMP

# Quelques exemples survolés

# Mot de Thue-Morse et carrés magiques (1/2)

Mot de Thue-Morse : point fixe de

$$\mu : \begin{cases} 0 \rightarrow 01 \\ 1 \rightarrow 10 \end{cases}$$

$$0110100110010110100101100110 1001 \ldots$$

Multiples propriétés :

- pas de chevauchement
- nème terme = parité du nombre de 1 dans écriture binaire de n

# Mot de Thue-Morse et carrés magiques (2/2)

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | **2** | **3** | 4 | **5** | 6 | 7 | **8** | **9** | 10 | 11 | **12** | 13 | **14** | **15** | 16 |

# Mot de Thue-Morse et carrés magiques (2/2)

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | **2** | **3** | 4 | **5** | 6 | 7 | **8** | **9** | 10 | 11 | **12** | 13 | **14** | **15** | 16 |

| | | | |
|---|---|---|---|
| | 2 | 3 | |
| 5 | | | 8 |
| 9 | | | 12 |
| | 14 | 15 | |

# Mot de Thue-Morse et carrés magiques (2/2)

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | **2** | **3** | 4 | **5** | 6 | 7 | **8** | **9** | 10 | 11 | **12** | 13 | **14** | **15** | 16 |

| | | | |
|---|---|---|---|
| | 2 | 3 | |
| 5 | | | 8 |
| 9 | | | 12 |
| | 14 | 15 | |

| | | | |
|---|---|---|---|
| 16 | 2 | 3 | 13 |
| 5 | 11 | 10 | 8 |
| 9 | 7 | 6 | 12 |
| 4 | 14 | 15 | 1 |

# Mot de Thue-Morse et carrés magiques (2/2)

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | **2** | **3** | 4 | **5** | 6 | 7 | **8** | **9** | 10 | 11 | **12** | 13 | **14** | **15** | 16 |

| | | | |
|---|---|---|---|
| | 2 | 3 | |
| 5 | | | 8 |
| 9 | | | 12 |
| | 14 | 15 | |

| | | | |
|---|---|---|---|
| 16 | 2 | 3 | 13 |
| 5 | 11 | 10 | 8 |
| 9 | 7 | 6 | 12 |
| 4 | 14 | 15 | 1 |

Fonctionne pour les carrés magiques $2^n \times 2^n$ avec $n \geq 2$

# Mot de Thue-Morse et carrés magiques (2/2)

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | **2** | **3** | 4 | **5** | 6 | 7 | **8** | **9** | 10 | 11 | **12** | 13 | **14** | **15** | 16 |

| | | | |
|---|---|---|---|
| | 2 | 3 | |
| 5 | | | 8 |
| 9 | | | 12 |
| | 14 | 15 | |

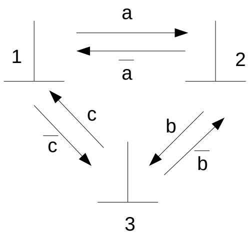
| | | | |
|---|---|---|---|
| 16 | 2 | 3 | 13 |
| 5 | 11 | 10 | 8 |
| 9 | 7 | 6 | 12 |
| 4 | 14 | 15 | 1 |

Fonctionne pour les carrés magiques $2^n \times 2^n$ avec $n \geq 2$


A. Adler and S.-Y. R. Li, Magic cubes and Prouhet sequences
Amer. Math. Monthly 84 (8), p618-627 (1977)

J. Berstel, A. Lauve, C. Reutenauer, F. V. Saliola
Combinatorics on Words, CRM Monograph Series 27, 2009

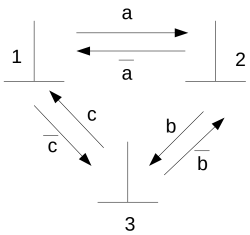
# Tours de Hanoï (1/6)




Exemple : déplacement de 3 disques de 1 vers 2 va être codé par

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

# Tours de Hanoï (1/6)

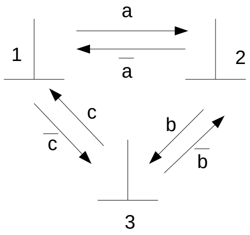



Exemple : déplacement de 3 disques de 1 vers 2 va être codé par

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

Connu : déplacement optimal en $2^n - 1$ déplacement pour $n$ disques

# Tours de Hanoï (2/6)




$H_n =$ suite de déplacements optimal pour $n$ disques à partir de 1 (vers 2 si $n$ pair, vers 3 si $n$ impair)

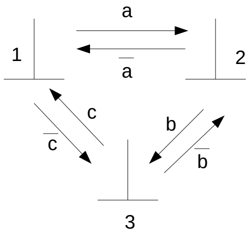
$H_0 = \varepsilon$ (mot vide)

$H_1 = a\bar{c}b$

$H_2 = a\bar{c}bac\bar{b}a$

$H_3 = a\bar{c}bac\bar{b}a\bar{c}b\bar{a}cba\bar{c}b$

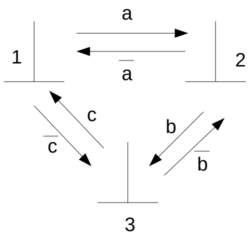
# Tours de Hanoï (3/6)




Pour déplacement à partir de 2 (vers 3 ou 1 selon parité) : $\sigma(H_n)$

$$\text{où } \sigma \begin{cases} \sigma(a) = b & \sigma(\overline{a}) = \overline{b} \\ \sigma(b) = c & \sigma(\overline{b}) = \overline{c} \\ \sigma(c) = a & \sigma(\overline{c}) = \overline{a} \end{cases}$$

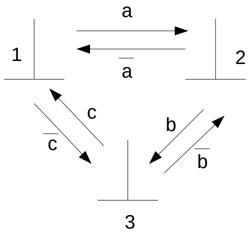
# Tours de Hanoï (3/6)




Pour déplacement à partir de 2 (vers 3 ou 1 selon parité) : $\sigma(H_n)$

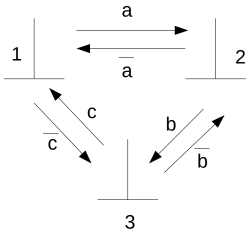
où $\sigma \begin{cases} \sigma(a) = b & \sigma(\overline{a}) = \overline{b} \\ \sigma(b) = c & \sigma(\overline{b}) = \overline{c} \\ \sigma(c) = a & \sigma(\overline{c}) = \overline{a} \end{cases}$

Pour déplacement de 3 vers 1 : $\sigma^2(H_n)$




Observation :

- $H_0 = \varepsilon$ (mot vide)
- $H_{2n+1} = H_{2n} a \sigma^{-1}(H_2 n)$ for $n \geq 0$
- $H_{2n} = H_{2n-1} \overline{c} \sigma(H_{2n-1})$ for $n \geq 1$



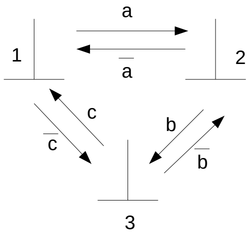

Résultat : for all $n \geq 0$ :

- $\varphi(H_{2n})a = H_{2n+1}$
- $\varphi(H_{2n+1})b = H_{2n+2}$

où

$$\varphi \begin{cases} a \to a\overline{c} & \overline{a} \to ac \\ b \to c\overline{b} & \overline{b} \to cb \\ c \to b\overline{a} & \overline{c} \to ba \end{cases}$$

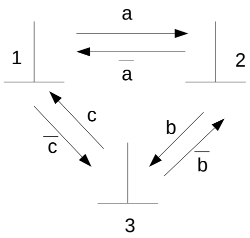
# Tours de Hanoï (6/6)




Conclusions :

- $H_n$ préfixe de longueur $2^n - 1$ du point fixe de $\varphi$ en $a$
- Analyse de ce point fixe : pas de carré

# Tours de Hanoï (6/6)




Conclusions :

- $H_n$ préfixe de longueur $2^n - 1$ du point fixe de $\varphi$ en $a$
- Analyse de ce point fixe : pas de carré


J.P. Allouche, D. Astorian, J. Randall, J. Shallit
Morphisms and Squarefree Strings, and the Tower of Hanoi Puzzle
Amer. Math. Monthly 101 (7), 1994.

# Mots et circularité (1/2)

B. Smyth
Computing Patterns in Strings, 2003

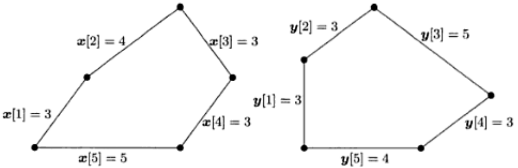



**Figure 1.1.** Two congruent polygons

En "computer graphics" et "computational geometry"

- représentation d'un polygone convexe par 2 chaînes
  - longueur des côtés dans ordre occurrence
  - angles dans même ordre
- problème : tester congruence
  (mêmes longueurs et mêmes angles)
- solution naïve : comparer un représentant d'un polygone avec tous les représentants de l'autre polygone

# Mots et circularité (2/2)

B. Smyth
Computing Patterns in Strings, 2003

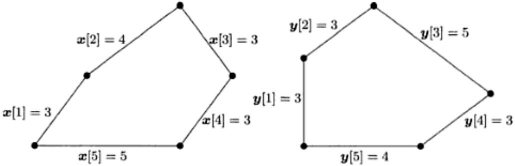



**Figure 1.1.** Two congruent polygons

Meilleure solution = choix d'un représentant
Mot de Lyndon = mot plus petit que ses suffixes propres dans l'ordre lexicographique.

# Mots et circularité (2/2)

B. Smyth
Computing Patterns in Strings, 2003

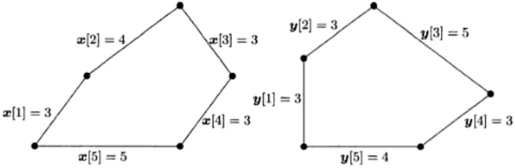



**Figure 1.1.** Two congruent polygons

Meilleure solution = choix d'un représentant
Mot de Lyndon = mot plus petit que ses suffixes propres dans l'ordre lexicographique.

*Solution du même type optée pour un problème de placement de robots (problème d'élection de leader dans un système distribué)*

Y. Dieudonné, F. Petit
Circle formation of weak robots and Lyndon words
Infor. Process. Letters 101, p156-162, 2007

# Au sujet algorithme de Burrows Wheeler (1/

S. Mantaci, A. Restivo, M. Sciortino
Burrows–Wheeler transform and Sturmian words
Infor. Process. Letters 86, p241–246, 2003

"The Burrows–Wheeler transform (BWT from now on) was introduced in 1994 by Burrows and Wheeler and it represents an extremely useful tool for textual lossless data compression.

# Au sujet algorithme de Burrows Wheeler (1/

S. Mantaci, A. Restivo, M. Sciortino
Burrows–Wheeler transform and Sturmian words
Infor. Process. Letters 86, p241–246, 2003

"The Burrows–Wheeler transform (BWT from now on) was introduced in 1994 by Burrows and Wheeler and it represents an extremely useful tool for textual lossless data compression.
The idea is to apply a reversible transformation in order to produce a permutation BWT(w) of an input string w, defined over an alphabet A, so that the string becomes easier to compress."

# Au sujet algorithme de Burrows Wheeler (1/

S. Mantaci, A. Restivo, M. Sciortino
Burrows–Wheeler transform and Sturmian words
Infor. Process. Letters 86, p241–246, 2003

"The Burrows–Wheeler transform (BWT from now on) was introduced in 1994 by Burrows and Wheeler and it represents an extremely useful tool for textual lossless data compression.
The idea is to apply a reversible transformation in order to produce a permutation BWT(w) of an input string w, defined over an alphabet A, so that the string becomes easier to compress."

Principe :

- Considérer les conjugués (permutations circulaires)
- Les trier
- BWT(w) = mot des dernières lettres

# Au sujet algorithme de Burrows Wheeler (2/)

Exemple : $w = abaababa$

Conjugués :

$abaababa$

$baababaa$

$aababaab$

$ababaaba$

$babaabaa$

$abaabaab$

$baabaaba$

$aabaabab$

# Au sujet algorithme de Burrows Wheeler (2/)

Exemple : $w = abaababa$

| Conjugués : | Triés : |
|---|---|
| $abaababa$ | $aabaabab$ |
| $baababaa$ | $aababaab$ |
| $aababaab$ | $abaabaab$ |
| $ababaaba$ | $abaababa$ |
| $babaabaa$ | $ababaaba$ |
| $abaabaab$ | $baabaaba$ |
| $baabaaba$ | $baababaa$ |
| $aabaabab$ | $babaabaa$ |

$$\mathrm{BWT}(abaababa) = bbbaaaaa = b^3a^5$$

Plus généralement : BWT(w) $= b^k a^\ell$ avec $pgcd(k, \ell) = 1 \iff w$ conjugué d'un mot standard

# Approfondissons !

# Qu'est-ce qu'un mot ?

- Alphabet : ensemble fini non vide de symboles
  Attention ! A la non-ambigüité des symboles !
- Lettre : élément d'un alphabet.
- Mot : suite de lettres.
  Notation : les lettres sont accolées.
  $(a, b, a, a, b)$ est noté $abaab$

  Formellement, on note : $a_1 \dots a_n$ un mot constitué dans l'ordre des lettres $a_1$ puis $a_2$ puis $\dots a_n$.
- Mot vide : suite vide de lettres : $\varepsilon$.

  Convention : $a_p \dots a_{p-1}$ désigne le mot vide.

# La concaténation

- Définition : la concaténation de deux mots $u$ et $v$ est le mot obtenu en mettant bout à bout dans l'ordre les lettres de $u$ puis les lettres de $v$.
- Notation : $uv$ ou $u.v$
- Exemples :
  $(aabac).(dab)$ vaut $aabacdab$
- Formellement, si $n \geq 0$ et $p \geq 0$ sont deux entiers, si $a_1, \ldots, a_n$, $b_1 \ldots, b_p$ sont des lettres, alors la concaténation des mots $a_1 \ldots a_n$ et $b_1 \ldots b_p$ est le mot $a_1 \ldots a_n b_1 \ldots b_p$.

# Monoïde

- Propriétés de la concaténation :
  - $u\varepsilon = u = \varepsilon u$
  - $(uv)w = u(vw)$
- Propriété : l'ensemble des mots munis de la concaténation forme un monoïde.
- Monoïde : ensemble muni d'une opération interne associative possédant un élément neutre.
- Autres exemples de monoïdes : $(\mathbb{N}, +)$, $(\mathbb{N}^+, \times)$
- Notation de l'ensemble des mots définis sur un alphabet $A$ :

$$A^*$$

# Monoïde libre

- Propriété fondamentale :
  Tout mot se décompose de manière unique sur les lettres.
  Plus formellement :
  si $a_1, \ldots, a_n, b_1, \ldots, b_p$ sont des lettres alors
  l'égalité $a_1 \ldots a_n = b_1 \ldots b_p$ implique $n = p$ et $a_i = b_i$ pour tout entier $i$, $1 \leq i \leq n$.
- Le monoïde $A^*$ est dit libre (de base $A$).

# Longueur

- Longueur d'un mot : nombre de lettres qui le composent.
- Notation : $|u|$ est la longueur de $u$.
- Exemple : $|abaab| = 5$
- Propriété :
  - $|\varepsilon| = 0$
  - $|uv| = |u| + |v|$

# Nombre d'occurrences

- $|u|_a$ : nombre d'occurrences de $a$ dans $u$
  ($a$ est une lettre, $u$ un mot)
- Exemple : $|abaab|_a = 3$
- Propriété :
  - $|\varepsilon|_a = 0$
  - $|uv|_a = |u|_a + |v|_a$

# Morphisme de Monoïdes

- La longueur des mots sur un alphabet $A$ est un *morphisme* de $(A^*, .)$ dans $(\mathbb{N}, +)$.
- Soient $(M_1, ._1)$ et $(M_2, ._2)$ deux monoïdes. Un morphisme $f$ de $M_1$ dans $M_2$ est une application de $M_1$ dans $M_2$ qui :
  - préserve le produit :
    pour tous $x, y$ dans $M_1$,
    $$f(x ._1 y) = f(x) ._2 f(y)$$
  - préserve l'élément neutre : l'image de l'élément neutre de $M_1$ est l'élément neutre de $M_2$.

# Morphisme de monoïdes libres

- Def. Morphisme entre deux monoïdes libres
  (on dira juste morphisme).
- Propriété : un morphisme de monoïdes libres est entièrement défini par l'image des lettres.
  - $f(\varepsilon) = \varepsilon$
  - $f(a_1 \dots a_n) = f(a_1) \dots f(a_n)$
- Exemple : morphisme de Fibonacci :

$$\varphi \begin{cases} a \mapsto ab \\ b \mapsto a \end{cases}$$

$$\varphi(aabab) = ababaaba$$

# Morphismes injectifs et codes

- Un morphisme $f$ est injectif si pour tous mots $u$ et $v$, $f(u) = f(v)$ implique $u = v$.
- Un ensemble de mots $X$ est un code si et seulement si pour tous entiers $n \geq 0$ et $p \geq 0$, pour touts mots, $x_1$, ..., $x_n$, $y_1$, ..., $y_p$, l'égalité $x_1 \ldots x_n = y_1 \ldots y_p$ implique $n = p$ et pour tout $i$ entre 1 et $p$, $x_i = y_i$.
- Propriété : un morphisme $f$ défini sur un alphabet $A$ est injectif si et seulement si $f(A)$ est un code.

# Morphismes. . .

. . . quelques utilisations

- Codage
- Construction simple de mots
- Explication de la structure de certains mots
- Formalisation de la notion d'équation