Binomial^3

coefficients, equivalence, complexity...

Michel Rigo

http://www.discmath.ulg.ac.be/
joint work with Marie Lejeune and Matthieu Rosenfeld

One World Seminar on Combinatorics on Words
13th July 2020

The *binomial coefficient* of two finite words $x = x_1 \cdots x_p$ and $y = y_1 \cdots y_q$ counts occurrences of subsequences

$$\binom{x}{y} = \#\{(j_1, \ldots, j_q) \mid 1 \leq j_1 < \cdots < j_q \leq p \wedge x_{j_1} \cdots x_{j_q} = y\}.$$

$$\binom{011010}{010} =$$

Over a 1-letter alphabet

$$\binom{a^p}{a^q} = \binom{p}{q}, \quad p, q \in \mathbb{N}.$$

Background

The *binomial coefficient* of two finite words $x = x_1 \cdots x_p$ and $y = y_1 \cdots y_q$ counts occurrences of subsequences

$$\binom{x}{y} = \#\{(j_1, \dots, j_q) \mid 1 \leq j_1 < \cdots < j_q \leq p \wedge x_{j_1} \cdots x_{j_q} = y\}.$$

$$\binom{011010}{010} =$$

Over a 1-letter alphabet

$$\binom{\mathtt{a}^p}{\mathtt{a}^q} = \binom{p}{q}, \quad p, q \in \mathbb{N}.$$

Background

The *binomial coefficient* of two finite words $x = x_1 \cdots x_p$ and $y = y_1 \cdots y_q$ counts occurrences of subsequences

$$\binom{x}{y} = \#\{(j_1, \ldots, j_q) \mid 1 \leq j_1 < \cdots < j_q \leq p \wedge x_{j_1} \cdots x_{j_q} = y\}.$$

$$\binom{\mathtt{011010}}{\mathtt{010}} = 6$$

Over a 1-letter alphabet

$$\binom{\mathtt{a}^p}{\mathtt{a}^q} = \binom{p}{q}, \quad p, q \in \mathbb{N}.$$

Background

Binomial coefficients of words have a long fascinating history:

- in Lothaire's book, Sakarovitch and Simon's chapter
- reconstruction problem: Let $k, n \in \mathbb{N}$. Words of length n are k-reconstructible whenever the multiset of scattered factors of length k (or k-deck) uniquely determines any word of length n [Kalashnik, Schützenberger 1973, Krasikov–Roditty 1997, Dudik–Schulman 2003,...]
- appear inside Parikh matrices
- link with piecewise testable languages [Simon 1975]
- noncommutative extension of Mahler's theorem on interpolation series [Pin–Silva 2014]
- generalized Pascal triangles [Leroy–R.–Stipulanti 2016]

Background

- Abelian equivalence (Erdős 1957)

$$astronomers \sim moonstarers \sim nomorestars^1$$

$$\Psi(0110100) = \begin{pmatrix} 4 \\ 3 \end{pmatrix} = \Psi(0101010).$$

- Karhumäki 1980 : *Generalized Parikh mappings and homomorphisms*
- k-abelian equivalence counts factors of length up to k

	0	1	00	01	10	11
0110100	4	3	1	2	2	1
0101010	4	3	0	3	3	0

[Huova, Karhumäki, Saarela, Whiteland, Zamboni, . . .]

[1] `wordsmith.org` Internet Anagram Server

Definitions

Let $k \geq 1$. Two finite words x, y are *k-binomially equivalent* if

$$x \sim_k y : \qquad \binom{x}{u} = \binom{y}{u}, \quad \forall u \in A^{\leq k}.$$

They have the same k-spectrum (formal polynomial introduced by Salomaa).

[Dudik–Schulman 2003]

$$\text{if } |x| \geq k \geq |u|, \quad \binom{|x|-|u|}{k-|u|}\binom{x}{u} = \sum_{t \in A^k} \binom{x}{t}\binom{t}{u}.$$

<u>Corollary</u>: Let $x, y \in A^{\geq k}$, $x \sim_k y$ if and only if

$$\binom{x}{u} = \binom{y}{u}, \quad \forall u \in A^k.$$

Definitions

- $x \sim_1 y$ iff x and y are abelian equivalent
- consecutive refinements: $x \sim_{k+1} y$ implies $x \sim_k y$

Let $\mathbf{w}$ be an infinite word and $\mathrm{Fac}_n(\mathbf{w})$ be its set of factors of length n. The *k-binomial complexity function* is

$$\mathsf{b}_{k,\mathbf{w}} : n \mapsto \# \left(\mathrm{Fac}_n(\mathbf{w}) / \sim_k\right)$$

$$\mathsf{b}_{1,\mathbf{w}}(n) \leq \cdots \leq \mathsf{b}_{k,\mathbf{w}}(n) \leq \mathsf{b}_{k+1,\mathbf{w}}(n) \leq \cdots \leq \mathsf{p}_{\mathbf{w}}(n)$$

An example

The twelve factors of length 5 of the Thue–Morse word:

$$\binom{u}{\mathtt{aa}} = \binom{|u|_{\mathtt{a}}}{2},$$

	$\binom{\cdot}{0}$	$\binom{\cdot}{1}$	$\binom{\cdot}{01}$	$\binom{\cdot}{10}$
11010	2	3	1	5
10110	2	3	2	4
11001	2	3	2	4
01101	2	3	4	2
10011	2	3	4	2
01011	2	3	5	1
10100	3	2	1	5
01100	3	2	2	4
10010	3	2	2	4
00110	3	2	4	2
01001	3	2	4	2
00101	3	2	5	1

$$\mathsf{b}_{2,\mathbf{t}}(5) = 8 < \mathsf{p}_{2,\mathbf{t}}(5) = 12.$$

Some results on binomial complexity

R.–Salimov TCS 2015

- Let $\mathbf{s}$ be a Sturmian word, then

$$\mathsf{b}_{2,\mathbf{s}}(n) = n + 1, \quad \forall n \geq 0.$$

Hence, $\mathsf{b}_{k,\mathbf{s}}(n) = n + 1$ for all $k \geq 2$ and all $n \geq 0$.

- A Parikh constant morphism f is such that

$$\Psi(f(\mathtt{a})) = \Psi(f(\mathtt{b})) \text{ for all letters } \mathtt{a}, \mathtt{b}.$$

Let $k \geq 1$. If $\mathbf{w}$ is a fixed point of f, then there exists a constant C_k such that

$$\mathsf{b}_{k,\mathbf{w}}(n) \leq C_k, \quad \forall n \geq 0.$$

- This is one of the few cases, with arithmetical complexity, where Sturmian words don't have minimal complexity among aperiodic words.

Some results on binomial complexity

Lejeune–Leroy–R. JCTA 2020

- For the Thue–Morse word $\mathbf{t}$, we know the constant C_k (as a function of k). Let $k \geq 1$.

  Short factors. For all $n \leq 2^k - 1$, we have

  $$\mathsf{b}_{k,\mathbf{t}}(n) = \mathsf{p}_{\mathbf{t}}(n).$$

  Longer factors. For all $n \geq 2^k$, we have

  $$\mathsf{b}_{k,\mathbf{t}}(n) = \begin{cases} 3 \cdot 2^k - 3, & \text{if } n \equiv 0 \pmod{2^k}; \\ 3 \cdot 2^k - 4, & \text{otherwise.} \end{cases}$$

Example : $\mathsf{b}_{2,\mathbf{t}}(5) = 8$.

$f^k(0) \sim_k f^k(1)$ but $f^k(0) \not\sim_{k+1} f^k(1)$ [Ochsenschläger 1981]

Some results on binomial complexity

Lejeune–R.–Rosenfeld AAM 2020

- Let $\mathbf{T}$ be the Tribonacci word `010201001` $\cdots$ then

$$\mathsf{b}_{2,\mathbf{T}}(n) = 2n + 1, \quad \forall n \geq 0.$$

  Hence, $\mathsf{b}_{k,\mathbf{T}}(n) = 2n + 1$ for all $k \geq 2$ and all $n \geq 0$.

We adapt a notion of *template* and *ancestor* [Aberkane, Currie, Rampersad, ...]

Equivalence classes

From the paper *The binomial equivalence classes of finite words*, Lejeune–R.–Rosenfeld, IJAC 2020, arXiv:2001.11732.

Take a finite alphabet A, what can be said about $A^*/\sim_k$? How look like the k-binomial equivalence classes ?

R.–Salimov, for a binary alphabet:

$$\#\left(\{0,1\}^n/\sim_2\right) = \frac{n^3+5n+6}{6} = \binom{n+1}{3} + n + 1$$

$$\texttt{A000125} = 1, 2, 4, 8, 15, 26, 42, 64, 93, 130, 176, \ldots$$

Cake numbers: maximal number of pieces resulting from n planar cuts through a cube

and, for an arbitrary k: polynomial growth of the number of classes

$$\#\left(\{0,1\}^n/\sim_k\right) \in \mathcal{O}(n^{2((k-1)2^k+1)})$$

Equivalence classes

an equivalence class: $[w]_\sim = \{u \in A^* \mid u \sim w\}$

In Whiteland's thesis, for k-abelian equivalence, study of

- The language made of *lexicographically least element* of each equivalence class

$$\boxed{\mathsf{LL}(\sim, A) = \{w \in A^* \mid \forall u \in [w]_\sim : w \leq_{lex} u\}\,.}$$

Note that $\underbrace{\#(\mathsf{LL}(\sim, A) \cap A^n)}_{\text{pick one word of each class}} = \#(A^n/\!\sim).$

- The language made of *singleton classes*

$$\boxed{\mathsf{Sing}(\sim, A) = \{w \in A^* \mid \#[w]_\sim = 1\}\,.}$$

Equivalence classes

[Whiteland's thesis] Let $k \geq 1$. For the k-abelian equivalence, $\mathsf{LL}(\sim_{k,ab}, A)$ and $\mathsf{Sing}(\sim_{k,ab}, A)$ are regular languages.

[Karhumäki–Puzynina–Rao–Whiteland TCS 2017]
Study of singleton k-abelian classes: connections with cycle decompositions of the de Bruijn graph, necklaces and Gray codes.

What can we learn for k-binomial equivalence?

2-binomial equivalence over a binary alphabet

Example, for $A = \{0, 1\}$ and $k = 2$:
Among the 32 words of length 5 in $\{0, 1\}^*$

- 20 give rise to a singleton class and,
- there are 6 classes of size 2 for the 2-binomial equivalence :

$$\{10110; 11001\},\ \{01110; 10101\},\ \{01101; 10011\},$$

$$\{01100; 10010\},\ \{01010; 10001\},\ \{00110; 01001\}.$$

It is easy to see that $x01y10z \sim_2 x10y01z$.

So

$$\#(\mathsf{Sing}(\sim_2, \{0, 1\}) \cap \{0, 1\}^5) = 20$$

and

$$\#(\mathsf{LL}(\sim_2, \{0, 1\}) \cap \{0, 1\}^5) = 26.$$

2-BINOMIAL EQUIVALENCE OVER A BINARY ALPHABET

From a result of Fossé and Richomme (2004):

They introduced a *switch (equivalence) relation* $\equiv$ such that $x01y10z \equiv x10y01z$ and its reflexive and transitive closure $\equiv^\star$.

The following assertions are equivalent:

- $u, v \in \{0,1\}^*$ are 2-binomially equivalent, $u \sim_2 v$,
- u, v have the same Parikh matrix,
- $u \equiv^\star v$.

Corollary: $\mathrm{Sing}(\sim_2, \{0,1\})$ is a regular language

$$0^*1^* + 1^*0^* + 0^*10^* + 1^*01^* + 0^*101^* + 1^*010^*$$

and, from a DFA, we can easily find the growth function of this language (and thus $\#(\{0,1\}^n/\sim_2)$).

2-binomial equivalence over larger alphabets

It's more complicated over a larger alphabet:

$$1223312 \sim_2 2311223$$

but there is no sequence of "switches" from one word to the other. Otherwise stated

$$u \equiv^\star v \Rightarrow u \sim_2 v \text{ but the converse does not hold.}$$

We have computed the first few values of

$$\#(\{1,2,3\}^n/\sim_2)$$

$$\texttt{A140348} = 1, 3, 9, 27, 78, 216, 568, 1410, \ldots.$$

THE ON-LINE ENCYCLOPEDIA OF INTEGER SEQUENCES ®

founded in 1964 by N. J. A. Sloane

1, 3, 9, 27, 78, 216, 568, 1410 Search Hints

(Greetings from The On-Line Encyclopedia of Integer Sequences!)

Search: **seq:1,3,9,27,78,216,568,1410**

Displaying 1-1 of 1 result found. page 1

Sort: relevance | references | number | modified | created Format: long | short | data

A140348 Growth function for the submonoid generated by the generators of the free nil-2 group on three generators. +30 1

1, 3, 9, 27, 78, 216, 568, 1410, 3309, 7307, 15303 (list; graph; refs; listen; history; text; internal format)

OFFSET 0,2

COMMENTS

```
The process of expressing a word in generators as a sorted word in generators
  and commutators is Marshall Hall's 'collection process'.
Since this monoid 'lives in' a nilpotent group, it inherits the growth
  restriction of a nilpotent group. So according to a result of Bass, a(n) = O(
  n^8).
It seems this is the correct growth rate. This sequence may well have a rational
  generating function, though, according to a result of M Stoll, the growth
  function of a nilpotent group need not be rational, or even algebraic.
Computations on a free nilpotent group, or on submonoids, may be aided by using
  matricies. I. D. MacDonald describes how to do this in an American
  Mathematical Monthly article and he gives a recipe explicitly for the nil-2,
  3 generator case.
```

Nil-2 group

Let $(G, \cdot)$ be a multiplicative group.

The commutator of 2 elements : $[x, y] = x^{-1}y^{-1}xy$

$$xy = yx[x, y] \quad \forall x, y \in G.$$

Note that $[x, y]^{-1} = [y, x]$.

A *nil-2 group*: the commutators belong to the center $Z(G)$, i.e.,

$$(\bullet): \qquad [x, y]z = z[x, y] \quad \forall x, y, z \in G.$$

Let $\Sigma = \{1, \ldots, m\}$ be a set of m generators. The *free nil-2 group* on Σ is the quotient of the free monoid $(\Sigma \cup \Sigma^{-1})^*$ under the relations $xx^{-1} = \varepsilon$ and $(\bullet)$.

$$12321 = (12[2, 1])[1, 2]321 = 21[1, 2]321 = 213(21[1, 2]) = 21312.$$

natural projection on the quotient: $\pi(12321) = \pi(21312)$.

NIL-2 GROUP

<u>Theorem:</u>
Let $\Sigma = \{1, \ldots, m\}$. The monoid $\Sigma^*/\sim_2$ is isomorphic to the submonoid, generated by Σ, of the nil-2 group $N_2(\Sigma)$.

Otherwise stated, if $r \in N_2(\Sigma)$, $\pi^{-1}(r) \cap \Sigma^*$ is an equivalence class for $\sim_2$; and conversely.

Generating the $\sim_2$-class of a word

Two possible questions:

- Given two words u, v, decide whether or not $u \sim_k v$ (Freydenberger, Gawrychowski, Karhumäki, Manea, Rytter 2015)
  - deterministic polynomial time algorithm (based on NFA)
  - Monte-Carlo algorithm with running time $\mathcal{O}(|u|k^2 + k^4)$
- Given a word u, list the words in $[u]_{\sim_k}$

Here, we explain how to list words in $[u]_{\sim_2}$ for an arbitrary alphabet

Generating the $\sim_2$-class of a word

A "switching" algorithm on words:

Input: a finite word $w = 1223312$
Output: a particular sequence of words $\ell_0, \ell_1, \ldots, w$

- starting from the lexicographically least element $\ell_0 = 1122233$ in the abelian class of w
- at each step, perform a single switch $ab \mapsto ba$, with $a < b$
- the longest common prefix with w is non-decreasing:

$$|\ell_i \wedge w| \leq |\ell_{i+1} \wedge w|$$

- we have $\ell_i = pcx$ and $w = pdy$ with $c < d$;
  consider the leftmost occurrence of d in x: $cx = \underline{cud}v$ and proceed to $|u| + 1$ switches to bring d in front.

Generating the $\sim_2$-class of a word

$w = 1223312$

$\ell_0 = 1122233$ common prefix with w: 1 ; $c = 1 < d = 2$

perform a switch $12 \mapsto 21$

$w = 1223312$

$\ell_1 = 1212233$ common prefix with w: 12 ; $c = 1 < d = 2$

perform a switch $12 \mapsto 21$

$w = 1223312$

$\ell_2 = 1221233$ common prefix with w: 122 ; ; $c = 1 < d = 3$

perform two switches $23 \mapsto 32$ and $13 \mapsto 31$

$w = 1223312$

$\ell_3 = 1221323$

$\ell_4 = 1223123$, common prefix with w: 1223 ; $c = 1 < d = 3$

perform two switches $23 \mapsto 32$ and $13 \mapsto 31$

$\ell_5 = 1223132$

$\ell_6 = 1223312 = w$

Generating the $\sim_2$-class of a word

$w = 1223312$

$\ell_0 = 1122233$ common prefix with w: 1 ; $c = 1 < d = 2$

perform a switch $12 \mapsto 21$

$w = 1223312$

$\ell_1 = 1212233$ common prefix with w: 12 ; $c = 1 < d = 2$

perform a switch $12 \mapsto 21$

$w = 1223312$

$\ell_2 = 1221233$ common prefix with w: 122 ; ; $c = 1 < d = 3$

perform two switches $23 \mapsto 32$ and $13 \mapsto 31$

$w = 1223312$

$\ell_3 = 1221323$

$\ell_4 = 1223123$, common prefix with w: 1223 ; $c = 1 < d = 3$

perform two switches $23 \mapsto 32$ and $13 \mapsto 31$

$\ell_5 = 1223132$

$\ell_6 = 1223312 = w$

Generating the $\sim_2$-class of a word

$w = 1223312$

$\ell_0 = 1122233$ common prefix with w: 1 ; $c = 1 < d = 2$

perform a switch $12 \mapsto 21$

$w = 1223312$

$\ell_1 = 1212233$ common prefix with w: 12 ; $c = 1 < d = 2$

perform a switch $12 \mapsto 21$

$w = 1223312$

$\ell_2 = 1221233$ common prefix with w: 122 ; ; $c = 1 < d = 3$

perform two switches $23 \mapsto 32$ and $13 \mapsto 31$

$w = 1223312$

$\ell_3 = 1221323$

$\ell_4 = 1223123$, common prefix with w: 1223 ; $c = 1 < d = 3$

perform two switches $23 \mapsto 32$ and $13 \mapsto 31$

$\ell_5 = 1223132$

$\ell_6 = 1223312 = w$

Generating the $\sim_2$-class of a word

$w = 1223312$

$\ell_0 = 1122233$ common prefix with w: 1 ; $c = 1 < d = 2$

perform a switch $12 \mapsto 21$

$w = 1223312$

$\ell_1 = 1212233$ common prefix with w: 12 ; $c = 1 < d = 2$

perform a switch $12 \mapsto 21$

$w = 1223312$

$\ell_2 = 1221233$ common prefix with w: 122 ; ; $c = 1 < d = 3$

perform two switches $23 \mapsto 32$ and $13 \mapsto 31$

$w = 1223312$

$\ell_3 = 1221323$

$\ell_4 = 1223123$,   common prefix with w: 1223 ; $c = 1 < d = 3$

perform two switches $23 \mapsto 32$ and $13 \mapsto 31$

$\ell_5 = 1223132$

$\ell_6 = 1223312 = w$

Generating the $\sim_2$-class of a word

$w = 1223312$
$\ell_0 = 1122233$ common prefix with w: 1 ; $c = 1 < d = 2$

perform a switch $12 \mapsto 21$
$w = 1223312$
$\ell_1 = 1212233$ common prefix with w: 12 ; $c = 1 < d = 2$

perform a switch $12 \mapsto 21$
$w = 1223312$
$\ell_2 = 1221233$ common prefix with w: 122 ; ; $c = 1 < d = 3$

perform two switches $23 \mapsto 32$ and $13 \mapsto 31$
$w = 1223312$
$\ell_3 = 1221323$
$\ell_4 = 1223123$, common prefix with w: 1223 ; $c = 1 < d = 3$

perform two switches $23 \mapsto 32$ and $13 \mapsto 31$
$\ell_5 = 1223132$
$\ell_6 = 1223312 = w$

Generating the $\sim_2$-class of a word

The lexicographically least element has the largest vector

$$\left(\binom{\ell_0}{12} \quad \binom{\ell_0}{13} \quad \binom{\ell_0}{23}\right)$$

(for lexicographic order on $\mathbb{N}^3$)

ℓ_i	$\binom{\cdot}{12}$	$\binom{\cdot}{13}$	$\binom{\cdot}{23}$
$1\underline{12}2233$	6	4	6
$12\underline{12}233$	5	4	6
$1221\underline{23}3$	4	4	6
$122\underline{13}23$	4	4	5
$12231\underline{23}$	4	3	5
$1223\underline{13}2$	4	3	4
1223312	4	2	4

⚠ A switch $ab \mapsto ba$ $(a < b)$ decreases by one the value of $\binom{u}{ab}$

Generating the $\sim_2$-class of a word

Some remarks:

- The $\sim_2$-equivalence class of a word u is completely determined by

$$\left(\binom{w}{1},\binom{w}{2},\binom{w}{3},\binom{w}{12},\binom{w}{13},\binom{w}{23}\right).$$

- $u \sim_2 v$ implies that u, v are abelian equivalent

- In particular, if two words are abelian equivalent, they are 2-binomially equivalent if they agree on

$$\left(\binom{\cdot}{12},\binom{\cdot}{13},\binom{\cdot}{23}\right).$$

Generating the $\sim_2$-class of a word

Two abelian equivalent words are 2-binomially equivalent if and only if *the total number of exchanges of $ab \mapsto ba$ $(a < b)$ when applying the algorithm, is the same*.

ℓ_i	$\binom{\cdot}{12}$	$\binom{\cdot}{13}$	$\binom{\cdot}{23}$
1122233	6	4	6
1212233	5	4	6
2112233	4	4	6
2112323	4	4	5
2113223	4	4	4
2131223	4	3	4
2311223	4	2	4

ℓ_i	$\binom{\cdot}{12}$	$\binom{\cdot}{13}$	$\binom{\cdot}{23}$
1122233	6	4	6
1212233	5	4	6
1221233	4	4	6
1221323	4	4	5
1223123	4	3	5
1223132	4	3	4
1223312	4	2	4

2 switches of each of the three types

Generating the $\sim_2$-class of a word

To determine all the words in $[1223312]_{\sim_2}$, we have to

- list all the words that can be obtained from 1122233
- when applying 2 switches of each of the three types $12 \mapsto 21$, $13 \mapsto 31$ and $23 \mapsto 32$.

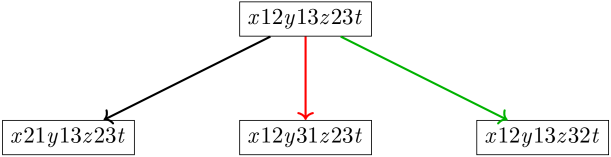
Remark:
The number of switches $ab \mapsto ba$, $a < b$, is given by

$$\binom{\ell_0}{ab} - \binom{w}{ab} = \binom{w}{ba}.$$

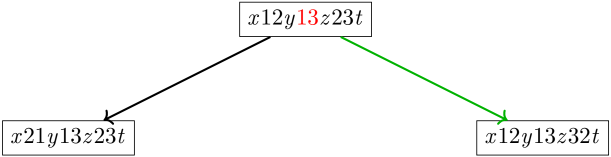
$$\binom{1223312}{21} = \binom{1223312}{31} = \binom{1223312}{32} = 2.$$

Generating the $\sim_2$-class of a word

- edges black : $12 \mapsto 21$; red : $13 \mapsto 31$; green $23 \mapsto 32$

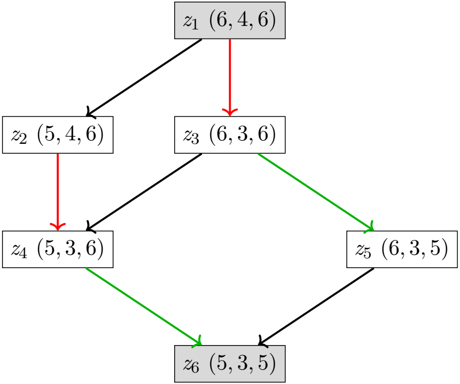


- Since w is given, limited number of edges of any given color. For instance, if no more red edge is available:



Generating the $\sim_2$-class of a word

- Two paths with the same origin and destination must use the same number of edges of any given color.

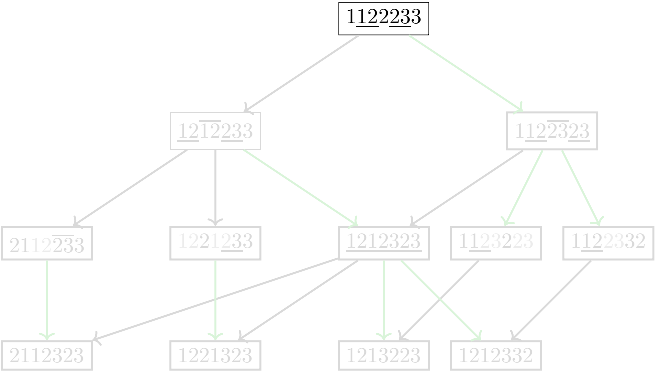


- There is always the path coming from the algorithm.

Generating the $\sim_2$-class of a word

Ex. cont. Building a graph (then reduced to a tree) with edges in black : $12 \mapsto 21$; red : $13 \mapsto 31$; green $23 \mapsto 32$

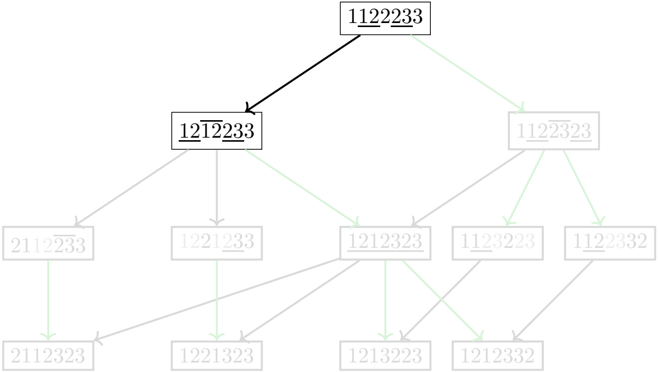
no more than 2 black/red/green edges on each path going downwards



first four levels

Generating the $\sim_2$-class of a word

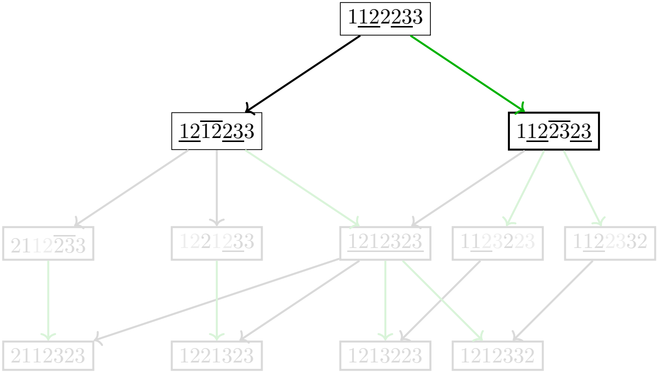
Ex. cont. Building a graph (then reduced to a tree) with edges in black : $12 \mapsto 21$; red : $13 \mapsto 31$; green $23 \mapsto 32$
no more than 2 black/red/green edges on each path going downwards



first four levels

Generating the $\sim_2$-class of a word

Ex. cont. Building a graph (then reduced to a tree) with edges in black : $12 \mapsto 21$; red : $13 \mapsto 31$; green $23 \mapsto 32$
no more than 2 black/red/green edges on each path going downwards

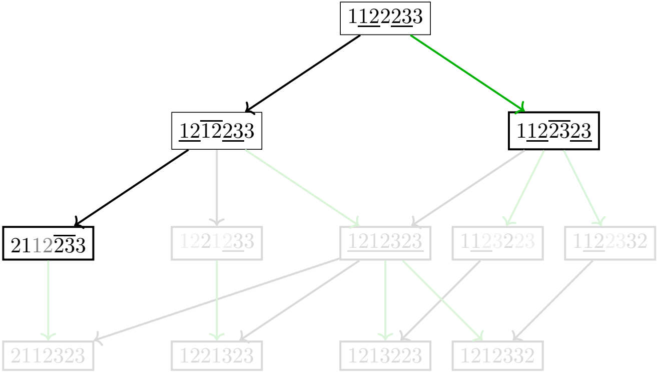


first four levels

Generating the $\sim_2$-class of a word

Ex. cont. Building a graph (then reduced to a tree) with edges in black : $12 \mapsto 21$; red : $13 \mapsto 31$; green $23 \mapsto 32$

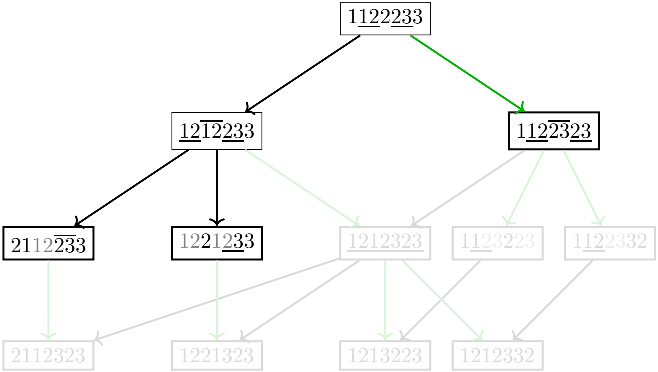
no more than 2 black/red/green edges on each path going downwards



first four levels

Generating the $\sim_2$-class of a word

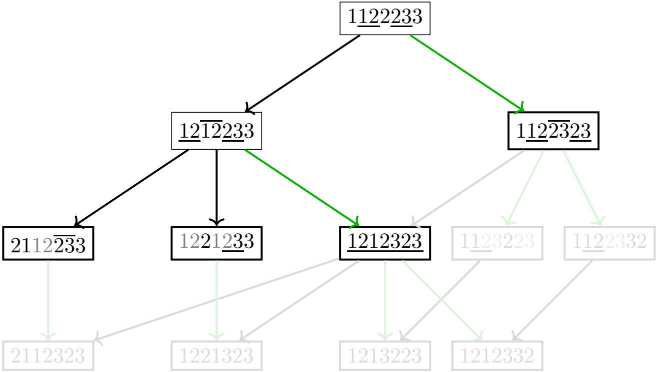
Ex. cont. Building a graph (then reduced to a tree) with edges in black : $12 \mapsto 21$; red : $13 \mapsto 31$; green $23 \mapsto 32$
no more than 2 black/red/green edges on each path going downwards



first four levels

Generating the $\sim_2$-class of a word

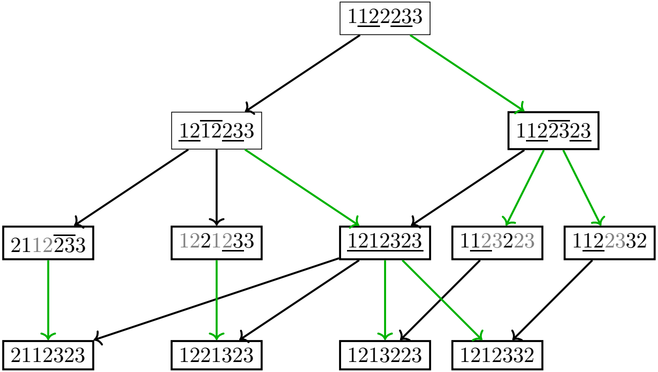
Ex. cont. Building a graph (then reduced to a tree) with edges in black : $12 \mapsto 21$; red : $13 \mapsto 31$; green $23 \mapsto 32$
no more than 2 black/red/green edges on each path going downwards



first four levels

Generating the $\sim_2$-class of a word

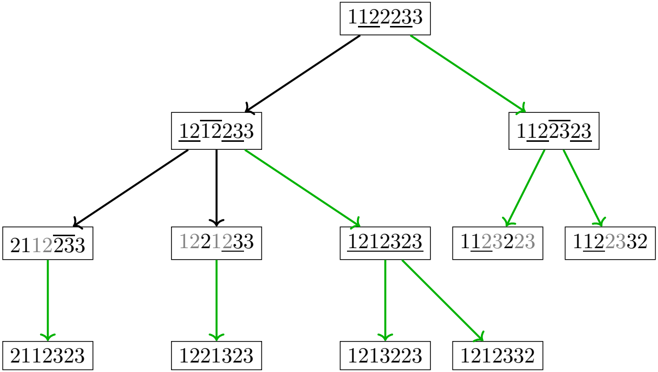
Ex. cont. Building a graph (then reduced to a tree) with edges in black : $12 \mapsto 21$; red : $13 \mapsto 31$; green $23 \mapsto 32$
no more than 2 black/red/green edges on each path going downwards



first four levels

Generating the $\sim_2$-class of a word

Ex. cont. Building a graph (then reduced to a tree) with edges in black : $12 \mapsto 21$; red : $13 \mapsto 31$; green $23 \mapsto 32$
If there are more than one path from the root to a vertex, keep the one corresponding to the algorithm.



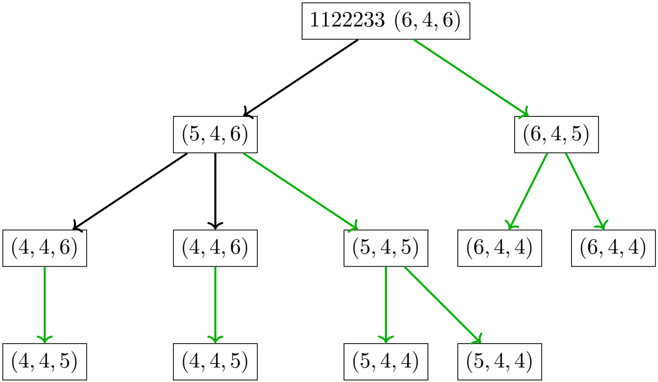
first four levels

Generating the $\sim_2$-class of a word

Ex. cont. Building a graph (then reduced to a tree) with edges in black : $12 \mapsto 21$; red : $13 \mapsto 31$; green $23 \mapsto 32$

We can keep track of the coefficients for 12, 13, 23

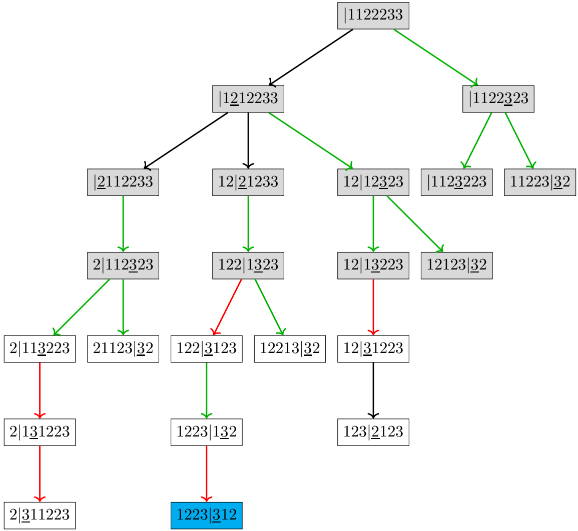
the total sum decreases by one on each level.



first four levels

Generating the $\sim_2$-class of a word

black : $12 \mapsto 21$; red : $13 \mapsto 31$; green $23 \mapsto 32$



$\binom{\cdot}{12}$	$\binom{\cdot}{13}$	$\binom{\cdot}{23}$
6	4	6
5	4	6
4	4	6
4	4	5
4	4	4
4	3	4
4	2	4
6	4	6
5	4	6
4	4	6
4	4	5
4	3	5
4	3	4
4	2	4

Generating the $\sim_2$-class of a word

To prove the result about the nil-2 group, we have introduced generalized binomial coefficients to the free group

For all words u over the alphabet $\Sigma \cup \Sigma^{-1}$ and $v \in \Sigma^t$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \sum_{(e_1,\dots,e_t)\in\{-1,1\}^t} \left(\prod_{i=1}^{t} e_i\right) \binom{u}{v_1^{e_1}\cdots v_t^{e_t}}.$$

Example:

$$\begin{bmatrix} aba^{-1}b \\ ab \end{bmatrix} = \underbrace{\binom{aba^{-1}b}{ab}}_{2} - \underbrace{\binom{aba^{-1}b}{a^{-1}b}}_{1} - \underbrace{\binom{aba^{-1}b}{ab^{-1}}}_{0} + \underbrace{\binom{aba^{-1}b}{a^{-1}b^{-1}}}_{0}.$$

Generating the $\sim_2$-class of a word

If two words u, v over $\Sigma \cup \Sigma^{-1}$ are such that $\pi(u) = \pi(v)$, i.e. they represent the same element of the nil-2 group, then

$$\begin{bmatrix} u \\ x \end{bmatrix} = \begin{bmatrix} v \\ x \end{bmatrix}$$

for $x = 1, 2, 3, 12, 13, 21, 23, 31, 32.$

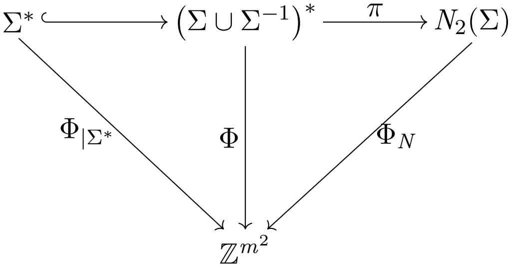
$$\Phi(w) := \left(\begin{bmatrix} w \\ 1 \end{bmatrix}, \begin{bmatrix} w \\ 2 \end{bmatrix} \begin{bmatrix} w \\ 3 \end{bmatrix}, \begin{bmatrix} w \\ 12 \end{bmatrix}, \begin{bmatrix} w \\ 13 \end{bmatrix}, \begin{bmatrix} w \\ 21 \end{bmatrix}, \begin{bmatrix} w \\ 23 \end{bmatrix}, \begin{bmatrix} w \\ 31 \end{bmatrix}, \begin{bmatrix} w \\ 32 \end{bmatrix} \right)$$

If $u, x \in \Sigma^*$, then

$$\begin{bmatrix} u \\ x \end{bmatrix} = \binom{u}{x}.$$

<u>Corollary</u>: if $u, v \in \Sigma^*$ are such that $\pi(u) = \pi(v)$, then $u \sim_2 v$.

Generating the $\sim_2$-class of a word



For the converse, if $u, v \in \Sigma^*$ are such that $u \sim_2 v$, we have to prove that $\pi(u) = \pi(v)$ $\rightsquigarrow$ we make use of the algorithm.

Growth order

- Salimov–R. bounds for binary alphabet
- In Lejeune's master thesis: $A = \{1, \ldots, m\}$ be an alphabet of size $m \geq 2$

$$\#(A^n/\sim_k) \in \mathcal{O}\left(n^{\frac{m}{(m-1)^2}(1+m^k(km-k-1))}\right).$$

- Let $A = \{1, \ldots, m\}$ be an alphabet of size $m \geq 2$ and $k \geq 1$

$$\#(A^n/\sim_k) \in \mathcal{O}\left(n^{k^2 m^k}\right)$$

$$\#(A^n/\sim_2) \in \Theta\left(n^{m^2-1}\right)$$

when n tends to infinity.

Non context-freeness

In comparison with Witheland's result, we get:

For any alphabet A of size at least 3 and for any $k \geq 2$, the languages $\mathsf{LL}(\sim_k, A)$ and $\mathsf{Sing}(\sim_k, A)$ are not context-free.

- From the previous slide, we have a polynomial bound

$$\#(\mathsf{Sing}(\sim_k, A) \cap A^n) \leq \#(\mathsf{LL}(\sim_k, A) \cap A^n) = \#(A^n / \sim_k) \leq P(n).$$

- [Ginsburg–Spanier]
A context-free language L is bounded, $L \subseteq w_1^* w_2^* \cdots w_\ell^*$, if and only if it has a polynomial growth, $\#(L \cap A^n) \leq Q(n)$.

$\rightsquigarrow$ it is enough to show that $\mathsf{LL}(\sim_k, A)$ and $\mathsf{Sing}(\sim_k, A)$ are not bounded.

Non context-freeness

If L is bounded and $M \subseteq L$, then M is bounded:

$$M \subseteq L \subseteq w_1^* w_2^* \cdots w_\ell^*$$

Hence, M not bounded implies L not bounded.

Strategy: define a particular (sub)family of singletons

$$\underbrace{\{\rho_{p,n} \mid p, n \in \mathbb{N}\}}_{\text{not bounded}} \subseteq \mathsf{Sing}(\sim_k, A) \subseteq \mathsf{LL}(\sim_k, A).$$

$$\rho_{p,n} := 1^p 2^{s_{n-1}} 3^{s_{n-2}} 1^{s_{n-3}} \cdots a^{s_1}$$

over $\{1, 2, 3\}$, where $a \equiv n \pmod 3$, and we take $s_n = 2 \times 8^{8^n}$.

Non context-freeness

If L is bounded and $M \subseteq L$, then M is bounded:

$$M \subseteq L \subseteq w_1^* w_2^* \cdots w_\ell^*$$

Hence, M not bounded implies L not bounded.

Strategy: define a particular (sub)family of singletons

$$\underbrace{\{\rho_{p,n} \mid p, n \in \mathbb{N}\}}_{\text{not bounded}} \subseteq \mathsf{Sing}(\sim_k, A) \subseteq \mathsf{LL}(\sim_k, A).$$

$$\rho_{p,n} := 1^p 2^{s_{n-1}} 3^{s_{n-2}} 1^{s_{n-3}} \cdots a^{s_1}$$

over $\{1, 2, 3\}$, where $a \equiv n \pmod 3$, and we take $s_n = 2 \times 8^{8^n}$.

Conclusions

k-binomial equivalence $\sim_k$

- $\#A = 2$, $k = 2$, switch equivalence — *everything is fine*
- $\#A \geq 3$, $k = 2$, algorithm and algebraic description of $\sim_2$-equivalence classes
- $\#A \geq 3$, $k = 2$, no simple operation corresponding to switch equivalence is known.
- $\#A \geq 3$, $k \geq 3$, extension of the above results?
- $\#A = 2$, $k = 2$, $\mathsf{LL}(\sim_2, A)$ is context-free
- $\#A \geq 3$, $k \geq 2$, $\mathsf{LL}(\sim_k, A)$ is not context-free, what about its descriptional complexity, automaticity?
- $\#A = 2$, $k \geq 3$, $\mathsf{LL}(\sim_k, A)$ — conjecture: not context-free, one needs to find an unbounded set of singletons...

Conclusions

Similar intricate “problems” for Parikh matrices/equivalence over larger alphabets ; see for instance A. C. Atanasiu, *Parikh Matrix Mapping and Amiability over a ternary alphabet*

Open question : give some (geometrical) interpretation of k-binomial equivalence/complexity

Some references

- J. Cassaigne, J. Karhumäki, S. Puzynina, and M. A. Whiteland, k-abelian equivalence and rationality, *Fund. Infor.* **154** (2017).
- S. Fossé, G. Richomme, Some characterizations of Parikh matrix equivalent binary words, *Inform. Process. Lett.* **92** (2004).
- D. D. Freydenberger, P. Gawrychowski, J. Karhumäki, F. Manea, W. Rytter, Testing k-binomial equivalence, in *Multidisciplinary Creativity: homage to Gheorghe Paun on his 65th birthday*, 239–248, Ed. Spandugino, Bucharest, Romania (2015).
- M. Lejeune, J. Leroy, M. Rigo, Computing the k-binomial complexity of the Thue–Morse word, *J. Comb. Theory, ser. A* **176** (2020).
- M. Lejeune, M. Rigo, M. Rosenfeld, Templates for the k-binomial complexity of the Tribonacci word, *Adv. Appl. Math.* **112** (2020).

Some references

- M. Lejeune, M. Rigo, M. Rosenfeld, On the binomial equivalence classes of finite words, to appear in IJAC, `arXiv:2001.11732`
- M. Rigo, P. Salimov, Another generalization of abelian equivalence: binomial complexity of infinite words, *Theoret. Comput. Sci.* **601** (2015).
- M. A. Whiteland, *On the k-Abelian Equivalence Relation of Finite Words*, Ph.D. Thesis, TUCS Dissertations **241**, Univ. of Turku (2019).