

Combinatorics of Lyndon words

Amy Glen

School of Chemical & Mathematical Sciences
Murdoch University, Perth, Australia

amy.glen@gmail.com
<http://amyglen.wordpress.com>

Mini-Conference

“Combinatorics, representations, and structure of Lie type”

The University of Melbourne

February 16–17, 2012

Words

By a *word*, I mean a *finite or infinite sequence* of symbols (*letters*) taken from a non-empty finite set $\mathcal{A}$ (*alphabet*).

Examples:

- 001
- $(001)^\infty = 001001001001001001001001001001 \dots$
- 1100111100011011101111001101110010111111101...
- 100102110122220102110021111102212222201112012...
- 1121212121212...

Words

By a *word*, I mean a finite or infinite sequence of symbols (*letters*) taken from a non-empty finite set $\mathcal{A}$ (*alphabet*).

Examples:

- 001
- $(001)^\infty = 0.\underset{\uparrow}{0}1001001001001001001001001001\dots$
- 1100111100011011101111001101110010111111101...
- 100102110122220102110021111102212222201112012...
- 1121212121212...

Words

By a *word*, I mean a finite or infinite sequence of symbols (*letters*) taken from a non-empty finite set $\mathcal{A}$ (*alphabet*).

Examples:

- 001
- $(001)^\infty = 0.01001001001001001001001001001 \dots = (2/7)_2$
- 1100111100011011101111001101110010111111101...
- 100102110122220102110021111102212222201112012...
- 1121212121212...

Words

By a *word*, I mean a finite or infinite sequence of symbols (*letters*) taken from a non-empty finite set $\mathcal{A}$ (*alphabet*).

Examples:

- 001
- $(001)^\infty = 0.01001001001001001001001001001 \dots = (2/7)_2$
- $1.\underset{\uparrow}{100111100011011101111001101110010} \dots$
- 100102110122220102110021111102212222201112012 $\dots$
- 1121212121212 $\dots$

Words

By a *word*, I mean a finite or infinite sequence of symbols (*letters*) taken from a non-empty finite set $\mathcal{A}$ (*alphabet*).

Examples:

- 001
- $(001)^\infty = 0.01001001001001001001001001001 \dots = (2/7)_2$
- $1.100111100011011101111001101110010 \dots = ((1 + \sqrt{5})/2)_2$
- 100102110122220102110021111102212222201112012...
- 1121212121212...

Words

By a *word*, I mean a finite or infinite sequence of symbols (*letters*) taken from a non-empty finite set $\mathcal{A}$ (*alphabet*).

Examples:

- 001
- $(001)^\infty = 0.01001001001001001001001001001 \dots = (2/7)_2$
- $1.100111100011011101111001101110010 \dots = ((1 + \sqrt{5})/2)_2$
- 10.0102110122220102110021111102212222201112012 ...
 ↑
- 1121212121212 ...

Words

By a *word*, I mean a finite or infinite sequence of symbols (*letters*) taken from a non-empty finite set $\mathcal{A}$ (*alphabet*).

Examples:

- 001
- $(001)^\infty = 0.01001001001001001001001001001 \dots = (2/7)_2$
- $1.100111100011011101111001101110010 \dots = ((1 + \sqrt{5})/2)_2$
- $10.0102110122220102110021111102212222201112012 \dots = (\pi)_3$
- 1121212121212...

Words

By a *word*, I mean a finite or infinite sequence of symbols (*letters*) taken from a non-empty finite set $\mathcal{A}$ (*alphabet*).

Examples:

- 001
- $(001)^\infty = 0.01001001001001001001001001001 \dots = (2/7)_2$
- $1.100111100011011101111001101110010 \dots = ((1 + \sqrt{5})/2)_2$
- $10.0102110122220102110021111102212222201112012 \dots = (\pi)_3$
- $[1; 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, \dots] = \sqrt{3}$

Words

By a *word*, I mean a *finite or infinite sequence* of symbols (*letters*) taken from a non-empty finite set $\mathcal{A}$ (*alphabet*).

Examples:

- 001
- $(001)^\infty = 0.01001001001001001001001001 \dots = (2/7)_2$
- $1.100111100011011101111001101110010 \dots = ((1 + \sqrt{5})/2)_2$
- $10.0102110122220102110021111102212222201112012 \dots = (\pi)_3$
- $[1; 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, \dots] = \sqrt{3}$

The study of combinatorial properties of words, known as **combinatorics on words** (MSC: 68R15), has connections to many modern, as well as classical, fields of mathematics with applications in areas ranging from *theoretical computer science* (from the algorithmic point of view) to *molecular biology* (DNA sequences).

Combinatorics on Words

- In particular, connections with algebra are very deep.

Combinatorics on Words

- In particular, connections with algebra are very deep.
Indeed, a natural environment of a finite word is a free monoid.

Combinatorics on Words

- In particular, connections with algebra are very deep.
Indeed, a natural environment of a finite word is a free monoid.
- Formally, under the operation of **concatenation**, the set $\mathcal{A}^*$ of all **finite words over $\mathcal{A}$** is a **free monoid** with identity element ε (the **empty word**) and set of generators $\mathcal{A}$.

Combinatorics on Words

- In particular, connections with algebra are very deep.
Indeed, a natural environment of a finite word is a free monoid.
- Formally, under the operation of **concatenation**, the set $\mathcal{A}^*$ of all **finite words over $\mathcal{A}$** is a **free monoid** with identity element ε (the **empty word**) and set of generators $\mathcal{A}$.
- The **set of non-empty finite words** over $\mathcal{A}$ is the **free semigroup** $\mathcal{A}^+ := \mathcal{A}^* \setminus \{\varepsilon\}$.

Combinatorics on Words

- In particular, connections with algebra are very deep.
Indeed, a natural environment of a finite word is a free monoid.
- Formally, under the operation of **concatenation**, the set $\mathcal{A}^*$ of all **finite words over $\mathcal{A}$** is a **free monoid** with identity element ε (the **empty word**) and set of generators $\mathcal{A}$.
- The **set of non-empty finite words** over $\mathcal{A}$ is the **free semigroup** $\mathcal{A}^+ := \mathcal{A}^* \setminus \{\varepsilon\}$.
- It follows immediately that the mathematical research of words exploits two features: **discreteness** and **non-commutativity**.

Combinatorics on Words

- In particular, connections with algebra are very deep.
Indeed, a natural environment of a finite word is a free monoid.
- Formally, under the operation of **concatenation**, the set $\mathcal{A}^*$ of all **finite words over $\mathcal{A}$** is a **free monoid** with identity element ε (the **empty word**) and set of generators $\mathcal{A}$.
- The **set of non-empty finite words** over $\mathcal{A}$ is the **free semigroup** $\mathcal{A}^+ := \mathcal{A}^* \setminus \{\varepsilon\}$.
- It follows immediately that the mathematical research of words exploits two features: **discreteness** and **non-commutativity**.
The latter aspect is what makes this field a very challenging one.

Combinatorics on Words

- In particular, connections with algebra are very deep.
Indeed, a natural environment of a finite word is a free monoid.
- Formally, under the operation of **concatenation**, the set $\mathcal{A}^*$ of all **finite words over $\mathcal{A}$** is a **free monoid** with identity element ε (the **empty word**) and set of generators $\mathcal{A}$.
- The **set of non-empty finite words** over $\mathcal{A}$ is the **free semigroup** $\mathcal{A}^+ := \mathcal{A}^* \setminus \{\varepsilon\}$.
- It follows immediately that the mathematical research of words exploits two features: **discreteness** and **non-commutativity**.
The latter aspect is what makes this field a very challenging one.
Many easily formulated problems are difficult to solve ...

Combinatorics on Words

- In particular, connections with algebra are very deep.
Indeed, a natural environment of a finite word is a free monoid.
- Formally, under the operation of **concatenation**, the set $\mathcal{A}^*$ of all **finite words over $\mathcal{A}$** is a **free monoid** with identity element ε (the **empty word**) and set of generators $\mathcal{A}$.
- The **set of non-empty finite words** over $\mathcal{A}$ is the **free semigroup** $\mathcal{A}^+ := \mathcal{A}^* \setminus \{\varepsilon\}$.
- It follows immediately that the mathematical research of words exploits two features: **discreteness** and **non-commutativity**.

The latter aspect is what makes this field a very challenging one.

Many easily formulated problems are difficult to solve . . . mainly because of the limited availability of mathematical tools to deal with non-commutative structures compared to commutative ones.

Combinatorics on Words . . .

- There have been important contributions on “words” dating as far back as the beginning of the last century.

Combinatorics on Words . . .

- There have been important contributions on “words” dating as far back as the beginning of the last century.
- Many of the earliest contributions were typically needed as tools to achieve some other goals in mathematics

Combinatorics on Words . . .

- There have been important contributions on “words” dating as far back as the beginning of the last century.
- Many of the earliest contributions were typically needed as tools to achieve some other goals in mathematics, with the exception of **combinatorial group theory** which studies combinatorial problems on words as representing group elements.

Combinatorics on Words . . .

- There have been important contributions on “words” dating as far back as the beginning of the last century.
- Many of the earliest contributions were typically needed as tools to achieve some other goals in mathematics, with the exception of **combinatorial group theory** which studies combinatorial problems on words as representing group elements.
- **Early 1900's**: First investigations by **Axel Thue** (repetitions in words)

Combinatorics on Words . . .

- There have been important contributions on “words” dating as far back as the beginning of the last century.
- Many of the earliest contributions were typically needed as tools to achieve some other goals in mathematics, with the exception of **combinatorial group theory** which studies combinatorial problems on words as representing group elements.
- **Early 1900's**: First investigations by **Axel Thue** (repetitions in words)
- **1938**: **Marston Morse & Gustav Hedlund**
Initiated the formal development of **symbolic dynamics**.

Combinatorics on Words . . .

- There have been important contributions on “words” dating as far back as the beginning of the last century.
- Many of the earliest contributions were typically needed as tools to achieve some other goals in mathematics, with the exception of **combinatorial group theory** which studies combinatorial problems on words as representing group elements.
- **Early 1900's**: First investigations by **Axel Thue** (repetitions in words)
- **1938**: **Marston Morse & Gustav Hedlund**
Initiated the formal development of **symbolic dynamics**.
This work marked the beginning of the formal study of words.

Combinatorics on Words . . .

- There have been important contributions on “words” dating as far back as the beginning of the last century.
- Many of the earliest contributions were typically needed as tools to achieve some other goals in mathematics, with the exception of **combinatorial group theory** which studies combinatorial problems on words as representing group elements.
- **Early 1900's**: First investigations by **Axel Thue** (repetitions in words)
- **1938**: **Marston Morse & Gustav Hedlund**
Initiated the formal development of **symbolic dynamics**.
This work marked the beginning of the formal study of words.
- **1960's**: Systematic study of words initiated by **M.P. Schützenberger**.

Combinatorics on Words . . .

- There have been important contributions on “words” dating as far back as the beginning of the last century.
- Many of the earliest contributions were typically needed as tools to achieve some other goals in mathematics, with the exception of **combinatorial group theory** which studies combinatorial problems on words as representing group elements.
- **Early 1900's**: First investigations by **Axel Thue** (repetitions in words)
- **1938**: **Marston Morse & Gustav Hedlund**
Initiated the formal development of **symbolic dynamics**.
This work marked the beginning of the formal study of words.
- **1960's**: Systematic study of words initiated by **M.P. Schützenberger**.
- Recent developments in **combinatorics on words** have culminated in the publication of three books by a collection of authors, under the pseudonym of M. Lothaire . . .

Combinatorics on Words . . .

Lothaire Books

- ① “Combinatorics on words”, 1983 (reprinted 1997)
- ② “Algebraic combinatorics on words”, 2002
- ③ “Applied combinatorics on words”, 2005

Combinatorics on Words . . .

Lothaire Books

- ① “Combinatorics on words”, 1983 (reprinted 1997)
- ② “Algebraic combinatorics on words”, 2002
- ③ “Applied combinatorics on words”, 2005

In the introduction to the first edition, Roger Lyndon stated

“This is the first book devoted to broad study of the combinatorics of words, that is to say, of sequences of symbols called letters. This subject is in fact very ancient and has cropped up repeatedly in a wide variety of subjects.”

Combinatorics on Words . . .

Lothaire Books

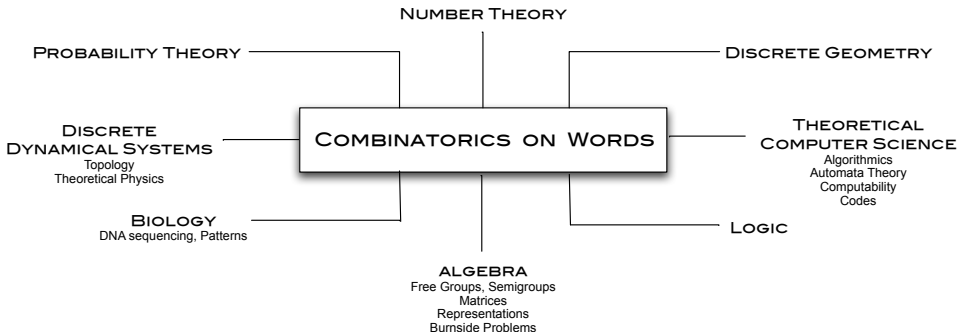
- ① “Combinatorics on words”, 1983 (reprinted 1997)
- ② “Algebraic combinatorics on words”, 2002
- ③ “Applied combinatorics on words”, 2005

In the introduction to the first edition, Roger Lyndon stated

“This is the first book devoted to broad study of the combinatorics of words, that is to say, of sequences of symbols called letters. This subject is in fact very ancient and has cropped up repeatedly in a wide variety of subjects.”

Combinatorics on words has now become a very active and challenging field in its own right.

Combinatorics on Words ...



Basic Definitions: Rotations

- Given a word $w = x_1x_2 \cdots x_n$ (with x_i letters), the first **rotation** of w is $R(w) = x_2 \cdots x_nx_1$.

Basic Definitions: Rotations

- Given a word $w = x_1x_2 \cdots x_n$ (with x_i letters), the first **rotation** of w is $R(w) = x_2 \cdots x_nx_1$.
- So $R(w)$ shifts w by one position to the left, $R^2(w)$ shifts by two positions to the left, and so on ...

Basic Definitions: Rotations

- Given a word $w = x_1x_2 \cdots x_n$ (with x_i letters), the first **rotation** of w is $R(w) = x_2 \cdots x_nx_1$.
- So $R(w)$ shifts w by one position to the left, $R^2(w)$ shifts by two positions to the left, and so on ...
- By convention, $w = R^0(w)$, the **0-th (trivial) rotation** of itself.

Basic Definitions: Rotations

- Given a word $w = x_1x_2 \cdots x_n$ (with x_i letters), the first **rotation** of w is $R(w) = x_2 \cdots x_nx_1$.
- So $R(w)$ shifts w by one position to the left, $R^2(w)$ shifts by two positions to the left, and so on ...
- By convention, $w = R^0(w)$, the **0-th (trivial) rotation** of itself.

Example

Consider the word $w = aabac$. This word has five distinct rotations:

Basic Definitions: Rotations

- Given a word $w = x_1x_2 \cdots x_n$ (with x_i letters), the first **rotation** of w is $R(w) = x_2 \cdots x_nx_1$.
- So $R(w)$ shifts w by one position to the left, $R^2(w)$ shifts by two positions to the left, and so on ...
- By convention, $w = R^0(w)$, the **0-th (trivial) rotation** of itself.

Example

Consider the word $w = aabac$. This word has five distinct rotations:

$$R^0(w) = aabac$$

Basic Definitions: Rotations

- Given a word $w = x_1x_2 \cdots x_n$ (with x_i letters), the first **rotation** of w is $R(w) = x_2 \cdots x_nx_1$.
- So $R(w)$ shifts w by one position to the left, $R^2(w)$ shifts by two positions to the left, and so on ...
- By convention, $w = R^0(w)$, the **0-th (trivial) rotation** of itself.

Example

Consider the word $w = aabac$. This word has five distinct rotations:

$$R^0(w) = aabac$$

$$R^1(w) = abaca$$

Basic Definitions: Rotations

- Given a word $w = x_1x_2 \cdots x_n$ (with x_i letters), the first **rotation** of w is $R(w) = x_2 \cdots x_nx_1$.
- So $R(w)$ shifts w by one position to the left, $R^2(w)$ shifts by two positions to the left, and so on ...
- By convention, $w = R^0(w)$, the **0-th (trivial) rotation** of itself.

Example

Consider the word $w = aabac$. This word has five distinct rotations:

$$R^0(w) = aabac$$

$$R^1(w) = abaca$$

$$R^2(w) = bacaa$$

Basic Definitions: Rotations

- Given a word $w = x_1x_2 \cdots x_n$ (with x_i letters), the first **rotation** of w is $R(w) = x_2 \cdots x_n x_1$.
- So $R(w)$ shifts w by one position to the left, $R^2(w)$ shifts by two positions to the left, and so on ...
- By convention, $w = R^0(w)$, the **0-th (trivial) rotation** of itself.

Example

Consider the word $w = aabac$. This word has five distinct rotations:

$$R^0(w) = aabac$$

$$R^1(w) = abaca$$

$$R^2(w) = bacaa$$

$$R^3(w) = acaab$$

Basic Definitions: Rotations

- Given a word $w = x_1x_2 \cdots x_n$ (with x_i letters), the first **rotation** of w is $R(w) = x_2 \cdots x_n x_1$.
- So $R(w)$ shifts w by one position to the left, $R^2(w)$ shifts by two positions to the left, and so on ...
- By convention, $w = R^0(w)$, the **0-th (trivial) rotation** of itself.

Example

Consider the word $w = aabac$. This word has five distinct rotations:

$$R^0(w) = aabac$$

$$R^1(w) = abaca$$

$$R^2(w) = bacaa$$

$$R^3(w) = acaab$$

$$R^4(w) = caaba$$

Basic Definitions: Rotations ...

- A word w of length $|w|$ has at most $|w|$ distinct rotations.

Basic Definitions: Rotations ...

- A word w of **length** $|w|$ has at most $|w|$ distinct rotations.
- More precisely, any word w can be uniquely expressed in the form

$$w = z^p = \underbrace{zz \cdots z}_{p \text{ times}}$$

where $p \geq 1$ and z is a **primitive word** (i.e., not a positive integer power of a shorter word).

Basic Definitions: Rotations ...

- A word w of **length** $|w|$ has at most $|w|$ distinct rotations.
- More precisely, any word w can be uniquely expressed in the form

$$w = z^p = \underbrace{zz \cdots z}_{p \text{ times}}$$

where $p \geq 1$ and z is a **primitive word** (i.e., not a positive integer power of a shorter word).

- Expressing a given word w in this way, it is easy to see that w has exactly $|w|/p$ ($= |z|$) distinct rotations

Basic Definitions: Rotations ...

- A word w of length $|w|$ has at most $|w|$ distinct rotations.
- More precisely, any word w can be uniquely expressed in the form

$$w = z^p = \underbrace{zz \cdots z}_{p \text{ times}}$$

where $p \geq 1$ and z is a **primitive word** (i.e., not a positive integer power of a shorter word).

- Expressing a given word w in this way, it is easy to see that w has exactly $|w|/p$ ($= |z|$) distinct rotations (the length of its **primitive root** z).

Basic Definitions: Rotations ...

- A word w of **length** $|w|$ has at most $|w|$ distinct rotations.
- More precisely, any word w can be uniquely expressed in the form

$$w = z^p = \underbrace{zz \cdots z}_{p \text{ times}}$$

where $p \geq 1$ and z is a **primitive word** (i.e., not a positive integer power of a shorter word).

- Expressing a given word w in this way, it is easy to see that w has exactly $|w|/p$ ($= |z|$) distinct rotations (the length of its **primitive root** z).

In particular, we note that **any primitive word** w has exactly $|w|$ distinct rotations.

Basic Definitions: Rotations ...

Examples

- 1 From the previous example, *aabac* is a primitive word of length 5 having 5 distinct rotations.

Basic Definitions: Rotations ...

Examples

- 1 From the previous example, $aabac$ is a primitive word of length 5 having 5 distinct rotations.
- 2 The word $v = abcabc = (abc)^2$ has $|v|/2 = 3$ distinct rotations:

Basic Definitions: Rotations ...

Examples

- 1 From the previous example, $aabac$ is a primitive word of length 5 having 5 distinct rotations.
- 2 The word $v = abcabc = (abc)^2$ has $|v|/2 = 3$ distinct rotations:

$$R^0(v) = abcabc$$

$$R^1(v) = bcabca$$

$$R^2(v) = cabcab$$

Basic Definitions: Rotations ...

Examples

- 1 From the previous example, $aabac$ is a primitive word of length 5 having 5 distinct rotations.
- 2 The word $v = abcabc = (abc)^2$ has $|v|/2 = 3$ distinct rotations:

$$R^0(v) = abcabc$$

$$R^1(v) = bcabca$$

$$R^2(v) = cabcab$$

- The **rotations** of a word w are also known as its **circular shifts** or **conjugates**.

Basic Definitions: Rotations ...

Examples

- 1 From the previous example, $aabac$ is a primitive word of length 5 having 5 distinct rotations.
- 2 The word $v = abcabc = (abc)^2$ has $|v|/2 = 3$ distinct rotations:

$$R^0(v) = abcabc$$

$$R^1(v) = bcabca$$

$$R^2(v) = cabcab$$

- The **rotations** of a word w are also known as its **circular shifts** or **conjugates**.
- Two words x and y over $\mathcal{A}$ are said to be **conjugate** if there exist finite words u and v such that $x = uv$ and $y = vu$.

Basic Definitions: Rotations ...

Examples

- 1 From the previous example, $aabac$ is a primitive word of length 5 having 5 distinct rotations.
- 2 The word $v = abcabc = (abc)^2$ has $|v|/2 = 3$ distinct rotations:

$$R^0(v) = abcabc$$

$$R^1(v) = bcabca$$

$$R^2(v) = cabcab$$

- The **rotations** of a word w are also known as its **circular shifts** or **conjugates**.
- Two words x and y over $\mathcal{A}$ are said to be **conjugate** if there exist finite words u and v such that $x = uv$ and $y = vu$.

Equivalently, x and y are conjugate if and only if there exists a word z such that $xz = zy$

Basic Definitions: Rotations ...

Examples

- 1 From the previous example, $aabac$ is a primitive word of length 5 having 5 distinct rotations.
- 2 The word $v = abcabc = (abc)^2$ has $|v|/2 = 3$ distinct rotations:

$$R^0(v) = abcabc$$

$$R^1(v) = bcabca$$

$$R^2(v) = cabcab$$

- The **rotations** of a word w are also known as its **circular shifts** or **conjugates**.
- Two words x and y over $\mathcal{A}$ are said to be **conjugate** if there exist finite words u and v such that $x = uv$ and $y = vu$.

Equivalently, x and y are conjugate if and only if there exists a word z such that $xz = zy$ (that is, $y = z^{-1}xz$).

Basic Definitions: Rotations ...

Examples

- 1 From the previous example, $aabac$ is a primitive word of length 5 having 5 distinct rotations.
- 2 The word $v = abcabc = (abc)^2$ has $|v|/2 = 3$ distinct rotations:

$$R^0(v) = abcabc$$

$$R^1(v) = bcabca$$

$$R^2(v) = cabcab$$

- The **rotations** of a word w are also known as its **circular shifts** or **conjugates**.
- Two words x and y over $\mathcal{A}$ are said to be **conjugate** if there exist finite words u and v such that $x = uv$ and $y = vu$.

Equivalently, x and y are conjugate if and only if there exists a word z such that $xz = zy$ (that is, $y = z^{-1}xz$).

- This is an **equivalence relation** on $\mathcal{A}^*$ since x is conjugate to y if and only if y can be obtained by a **cyclic permutation** (rotation) of the letters of x .

Basic Definitions: Necklaces

The set of all conjugates (rotations) of a given word is called its **conjugacy class**.

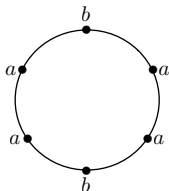
Basic Definitions: Necklaces

The set of all conjugates (rotations) of a given word is called its **conjugacy class**.

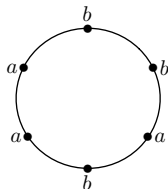
The conjugacy class of a word can be represented by a **necklace** (also known as a **circular word**).

Basic Definitions: Necklaces

The set of all conjugates (rotations) of a given word is called its **conjugacy class**. The conjugacy class of a word can be represented by a **necklace** (also known as a **circular word**).



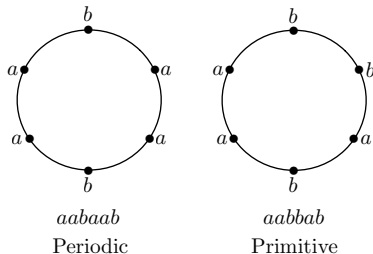
aabaab
Periodic



aabbab
Primitive

Basic Definitions: Necklaces

The set of all conjugates (rotations) of a given word is called its **conjugacy class**. The conjugacy class of a word can be represented by a **necklace** (also known as a **circular word**).

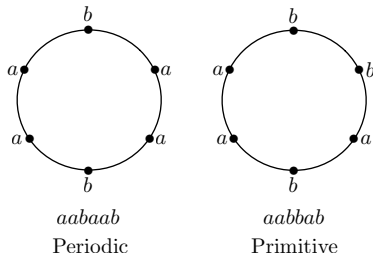


Note:

A necklace of length n over a k -letter alphabet can be thought of as n circularly connected beads of up to k different colours.

Basic Definitions: Necklaces

The set of all conjugates (rotations) of a given word is called its **conjugacy class**. The conjugacy class of a word can be represented by a **necklace** (also known as a **circular word**).



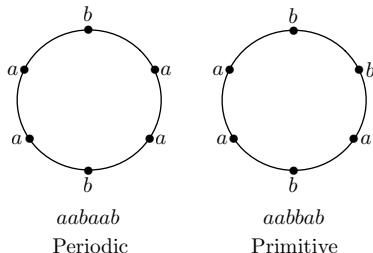
Note:

A necklace of length n over a k -letter alphabet can be thought of as n circularly connected beads of up to k different colours.

A necklace can also be classified as an orbit of the action of the cyclic group on words of length n .

Basic Definitions: Necklaces

The set of all conjugates (rotations) of a given word is called its **conjugacy class**. The conjugacy class of a word can be represented by a **necklace** (also known as a **circular word**).



Note:

A necklace of length n over a k -letter alphabet can be thought of as n circularly connected beads of up to k different colours.

A necklace can also be classified as an orbit of the action of the cyclic group on words of length n .

Each aperiodic (primitive) necklace can be uniquely represented by a so-called **Lyndon word** ...

Basic Definitions: Lyndon Words

Suppose the alphabet $\mathcal{A}$ is totally ordered by the relation $\prec$.

Basic Definitions: Lyndon Words

Suppose the alphabet $\mathcal{A}$ is totally ordered by the relation $\prec$.

Then we can totally order the semigroup $\mathcal{A}^+$ by the **lexicographical order** $\preceq$ (which is the usual **alphabetical order** in a dictionary) induced by the total order $\prec$ on $\mathcal{A}$.

Basic Definitions: Lyndon Words

Suppose the alphabet $\mathcal{A}$ is totally ordered by the relation $\prec$.

Then we can totally order the semigroup $\mathcal{A}^+$ by the **lexicographical order** $\preceq$ (which is the usual **alphabetical order** in a dictionary) induced by the total order $\prec$ on $\mathcal{A}$.

Definition

A finite word $w \in \mathcal{A}^+$ is a **Lyndon word** if w is **strictly smaller in lexicographical order** than all of its non-trivial **rotations** for the given total order $\prec$ on $\mathcal{A}$.

Basic Definitions: Lyndon Words

Suppose the alphabet $\mathcal{A}$ is totally ordered by the relation $\prec$.

Then we can totally order the semigroup $\mathcal{A}^+$ by the **lexicographical order** $\preceq$ (which is the usual **alphabetical order** in a dictionary) induced by the total order $\prec$ on $\mathcal{A}$.

Definition

A finite word $w \in \mathcal{A}^+$ is a **Lyndon word** if w is **strictly smaller in lexicographical order** than all of its non-trivial **rotations** for the given total order $\prec$ on $\mathcal{A}$.

Lyndon words are named after mathematician **Roger Lyndon**, who introduced them in 1954 under the name of **standard lexicographic sequences**.

Basic Definitions: Lyndon Words

Suppose the alphabet $\mathcal{A}$ is totally ordered by the relation $\prec$.

Then we can totally order the semigroup $\mathcal{A}^+$ by the **lexicographical order** $\preceq$ (which is the usual **alphabetical order** in a dictionary) induced by the total order $\prec$ on $\mathcal{A}$.

Definition

A finite word $w \in \mathcal{A}^+$ is a **Lyndon word** if w is **strictly smaller in lexicographical order** than all of its non-trivial **rotations** for the given total order $\prec$ on $\mathcal{A}$.

Lyndon words are named after mathematician **Roger Lyndon**, who introduced them in 1954 under the name of **standard lexicographic sequences**.

Being the **singularly** lexicographically smallest rotation implies that a Lyndon word is **different** from all of its non-trivial rotations, and is therefore **primitive**.

Basic Definitions: Lyndon Words

Suppose the alphabet $\mathcal{A}$ is totally ordered by the relation $\prec$.

Then we can totally order the semigroup $\mathcal{A}^+$ by the **lexicographical order** $\preceq$ (which is the usual **alphabetical order** in a dictionary) induced by the total order $\prec$ on $\mathcal{A}$.

Definition

A finite word $w \in \mathcal{A}^+$ is a **Lyndon word** if w is **strictly smaller in lexicographical order** than all of its non-trivial **rotations** for the given total order $\prec$ on $\mathcal{A}$.

Lyndon words are named after mathematician **Roger Lyndon**, who introduced them in 1954 under the name of **standard lexicographic sequences**.

Being the **singularly** lexicographically smallest rotation implies that a Lyndon word is **different** from all of its non-trivial rotations, and is therefore **primitive**.

So a Lyndon word w is a **primitive word that is the lexicographically smallest word in its conjugacy class**

Basic Definitions: Lyndon Words

Suppose the alphabet $\mathcal{A}$ is totally ordered by the relation $\prec$.

Then we can totally order the semigroup $\mathcal{A}^+$ by the **lexicographical order** $\preceq$ (which is the usual **alphabetical order** in a dictionary) induced by the total order $\prec$ on $\mathcal{A}$.

Definition

A finite word $w \in \mathcal{A}^+$ is a **Lyndon word** if w is **strictly smaller in lexicographical order** than all of its non-trivial **rotations** for the given total order $\prec$ on $\mathcal{A}$.

Lyndon words are named after mathematician **Roger Lyndon**, who introduced them in 1954 under the name of **standard lexicographic sequences**.

Being the **singularly** lexicographically smallest rotation implies that a Lyndon word is **different** from all of its non-trivial rotations, and is therefore **primitive**.

So a Lyndon word w is a **primitive word that is the lexicographically smallest word in its conjugacy class**, i.e., $w \in \mathcal{A}$ or $w \prec vu$ for all words u, v such that $w = uv$.

Basic Definitions: Lyndon Words

Suppose the alphabet $\mathcal{A}$ is totally ordered by the relation $\prec$.

Then we can totally order the semigroup $\mathcal{A}^+$ by the **lexicographical order** $\preceq$ (which is the usual **alphabetical order** in a dictionary) induced by the total order $\prec$ on $\mathcal{A}$.

Definition

A finite word $w \in \mathcal{A}^+$ is a **Lyndon word** if w is **strictly smaller in lexicographical order** than all of its non-trivial **rotations** for the given total order $\prec$ on $\mathcal{A}$.

Lyndon words are named after mathematician **Roger Lyndon**, who introduced them in 1954 under the name of **standard lexicographic sequences**.

Being the **singularly** lexicographically smallest rotation implies that a Lyndon word is **different** from all of its non-trivial rotations, and is therefore **primitive**.

So a Lyndon word w is a **primitive word that is the lexicographically smallest word in its conjugacy class**, i.e., $w \in \mathcal{A}$ or $w \prec vu$ for all words u, v such that $w = uv$.

It follows that **$w \in \mathcal{A}^+$ is a Lyndon word iff $w \in \mathcal{A}$ or $w \prec v$ for all proper suffixes v of w .**

Examples

Example 1: $w = aabac$ with alphabet $\mathcal{A} = \{a, b, c\}$

Examples

Example 1: $w = aabac$ with alphabet $\mathcal{A} = \{a, b, c\}$

w is a Lyndon word for the orders $a \prec b \prec c$ and $a \prec c \prec b$

Examples

Example 1: $w = aabac$ with alphabet $\mathcal{A} = \{a, b, c\}$

w is a Lyndon word for the orders $a \prec b \prec c$ and $a \prec c \prec b$
(i.e., a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = a$).

Examples

Example 1: $w = aabac$ with alphabet $\mathcal{A} = \{a, b, c\}$

w is a Lyndon word for the orders $a \prec b \prec c$ and $a \prec c \prec b$
(i.e., a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = a$).

$R^2(w) = bacaa$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = b$.

Examples

Example 1: $w = aabac$ with alphabet $\mathcal{A} = \{a, b, c\}$

w is a Lyndon word for the orders $a \prec b \prec c$ and $a \prec c \prec b$
(i.e., a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = a$).

$R^2(w) = bacaa$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = b$.

$R^4(w) = caaba$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = c$.

Examples

Example 1: $w = aabac$ with alphabet $\mathcal{A} = \{a, b, c\}$

w is a Lyndon word for the orders $a \prec b \prec c$ and $a \prec c \prec b$
(i.e., a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = a$).

$R^2(w) = bacaa$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = b$.

$R^4(w) = caaba$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = c$.

Example 2

For the 2-letter alphabet $\{a, b\}$ with $a \prec b$, the Lyndon words up to length five are as follows (sorted lexicographically for each length):

$a, b, ab, aab, abb, aaab, aabb, abbb, aaaab, aaabb, aabab, aabbb, abbbb, \dots$

Examples

Example 1: $w = aabac$ with alphabet $\mathcal{A} = \{a, b, c\}$

w is a Lyndon word for the orders $a \prec b \prec c$ and $a \prec c \prec b$ (i.e., a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = a$).

$R^2(w) = bacaa$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = b$.

$R^4(w) = caaba$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = c$.

Example 2

For the 2-letter alphabet $\{a, b\}$ with $a \prec b$, the Lyndon words up to length five are as follows (sorted lexicographically for each length):

$\underbrace{a, b}_2, \underbrace{ab}_1, \underbrace{aab, abb}_2, \underbrace{aaab, aabb, abbb}_3, \underbrace{aaaaab, aaabb, aabab, aabbb, ababb, abbbb}_{6}, \dots$

Examples

Example 1: $w = aabac$ with alphabet $\mathcal{A} = \{a, b, c\}$

w is a Lyndon word for the orders $a \prec b \prec c$ and $a \prec c \prec b$ (i.e., a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = a$).

$R^2(w) = bacaa$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = b$.

$R^4(w) = caaba$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = c$.

Example 2

For the 2-letter alphabet $\{a, b\}$ with $a \prec b$, the Lyndon words up to length five are as follows (sorted lexicographically for each length):

$\underbrace{a, b}_2, \underbrace{ab}_1, \underbrace{aab, abb}_2, \underbrace{aaab, aabb, abbb}_3, \underbrace{aaaaab, aaabb, aabab, aabbb, ababb, abbbb}_{6}, \dots$

How do we generate them?

Examples

Example 1: $w = aabac$ with alphabet $\mathcal{A} = \{a, b, c\}$

w is a Lyndon word for the orders $a \prec b \prec c$ and $a \prec c \prec b$ (i.e., a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = a$).

$R^2(w) = bacaa$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = b$.

$R^4(w) = caaba$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = c$.

Example 2

For the 2-letter alphabet $\{a, b\}$ with $a \prec b$, the Lyndon words up to length five are as follows (sorted lexicographically for each length):

$\underbrace{a, b}_2, \underbrace{ab}_1, \underbrace{aab, abb}_2, \underbrace{aaab, aabb, abbb}_3, \underbrace{aaaaab, aaabb, aabab, aabbb, ababb, abbbb}_{6}, \dots$

How do we generate them? How do we count them?

Examples

Example 1: $w = aabac$ with alphabet $\mathcal{A} = \{a, b, c\}$

w is a Lyndon word for the orders $a \prec b \prec c$ and $a \prec c \prec b$ (i.e., a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = a$).

$R^2(w) = bacaa$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = b$.

$R^4(w) = caaba$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = c$.

Example 2

For the 2-letter alphabet $\{a, b\}$ with $a \prec b$, the Lyndon words up to length five are as follows (sorted lexicographically for each length):

$\underbrace{a, b}_2, \underbrace{ab}_1, \underbrace{aab, abb}_2, \underbrace{aaab, aabb, abbb}_3, \underbrace{aaaab, aaabb, aabab, aabbb, ababb, abbbb}_{6}, \dots$

How do we generate them? How do we count them?

These questions will be answered in a moment

Examples

Example 1: $w = aabac$ with alphabet $\mathcal{A} = \{a, b, c\}$

w is a Lyndon word for the orders $a \prec b \prec c$ and $a \prec c \prec b$ (i.e., a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = a$).

$R^2(w) = bacaa$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = b$.

$R^4(w) = caaba$ is a Lyndon word for the orders on $\mathcal{A}$ with $\min(\mathcal{A}) = c$.

Example 2

For the 2-letter alphabet $\{a, b\}$ with $a \prec b$, the Lyndon words up to length five are as follows (sorted lexicographically for each length):

$\underbrace{a, b}_2, \underbrace{ab}_1, \underbrace{aab, abb}_2, \underbrace{aaab, aabb, abbb}_3, \underbrace{aaaab, aaabb, aabab, aabbb, ababb, abbbb}_{6}, \dots$

How do we generate them? How do we count them?

These questions will be answered in a moment, but first ...

Some Applications of Interest

- There are many and varied applications of Lyndon words in algebra and combinatorics – far too many to mention here.

Some Applications of Interest

- There are many and varied applications of Lyndon words in algebra and combinatorics – far too many to mention here.
- For instance, Lyndon words have an application to the description of free Lie algebras in constructing bases.

Some Applications of Interest

- There are many and varied applications of Lyndon words in algebra and combinatorics – far too many to mention here.
- For instance, Lyndon words have an application to the description of free Lie algebras in constructing bases.

This was, in fact, Lyndon's original motivation for introducing these words.

Some Applications of Interest

- There are many and varied applications of Lyndon words in algebra and combinatorics – far too many to mention here.
- For instance, Lyndon words have an application to the description of free Lie algebras in constructing bases.

This was, in fact, Lyndon's original motivation for introducing these words.

- Lyndon words may also be understood as a special case of Hall sets.
[C. Reutenauer, *Free Lie Algebras*, Oxford University Press, 1993]

Some Applications of Interest

- There are many and varied applications of Lyndon words in algebra and combinatorics – far too many to mention here.
- For instance, Lyndon words have an application to the description of free Lie algebras in constructing bases.

This was, in fact, Lyndon's original motivation for introducing these words.

- Lyndon words may also be understood as a special case of Hall sets. [C. Reutenauer, *Free Lie Algebras*, Oxford University Press, 1993]
- Lyndon words also have applications to semigroups, pattern matching, and representation theory of certain algebras (cf. recent work of Ram *et al.*).

Some Applications of Interest . . .

All such applications make use of **combinatorial properties of Lyndon words**, particularly factorisation theorems and lexicographical properties.

Some Applications of Interest . . .

All such applications make use of **combinatorial properties of Lyndon words**, particularly factorisation theorems and lexicographical properties.

The rest of this talk will be concerned with some of the **most important combinatorial properties** of Lyndon words, which you will hopefully find interesting and which may be useful in your own work.

Some Applications of Interest . . .

All such applications make use of **combinatorial properties of Lyndon words**, particularly factorisation theorems and lexicographical properties.

The rest of this talk will be concerned with some of the **most important combinatorial properties** of Lyndon words, which you will hopefully find interesting and which may be useful in your own work.

In particular, we'll learn about the following in regards to Lyndon words:

Some Applications of Interest . . .

All such applications make use of **combinatorial properties of Lyndon words**, particularly factorisation theorems and lexicographical properties.

The rest of this talk will be concerned with some of the **most important combinatorial properties** of Lyndon words, which you will hopefully find interesting and which may be useful in your own work.

In particular, we'll learn about the following in regards to Lyndon words:

- Enumeration

Some Applications of Interest . . .

All such applications make use of **combinatorial properties of Lyndon words**, particularly factorisation theorems and lexicographical properties.

The rest of this talk will be concerned with some of the **most important combinatorial properties** of Lyndon words, which you will hopefully find interesting and which may be useful in your own work.

In particular, we'll learn about the following in regards to Lyndon words:

- Enumeration
- Generation

Some Applications of Interest . . .

All such applications make use of **combinatorial properties of Lyndon words**, particularly factorisation theorems and lexicographical properties.

The rest of this talk will be concerned with some of the **most important combinatorial properties** of Lyndon words, which you will hopefully find interesting and which may be useful in your own work.

In particular, we'll learn about the following in regards to Lyndon words:

- Enumeration
- Generation
- Factorisation

Some Applications of Interest . . .

All such applications make use of **combinatorial properties of Lyndon words**, particularly factorisation theorems and lexicographical properties.

The rest of this talk will be concerned with some of the **most important combinatorial properties** of Lyndon words, which you will hopefully find interesting and which may be useful in your own work.

In particular, we'll learn about the following in regards to Lyndon words:

- Enumeration
- Generation
- Factorisation

I'll return to some of the aforementioned applications along the way . . .

Counting Lyndon Words

From now on, we consider words over a totally ordered finite alphabet $\mathcal{A}$ consisting of at least two distinct letters, unless stated otherwise.

Counting Lyndon Words

From now on, we consider words over a totally ordered finite alphabet $\mathcal{A}$ consisting of at least two distinct letters, unless stated otherwise.

- We know that the conjugacy class of any primitive word contains a single Lyndon word.

Counting Lyndon Words

From now on, we consider words over a totally ordered finite alphabet $\mathcal{A}$ consisting of at least two distinct letters, unless stated otherwise.

- We know that the conjugacy class of any primitive word contains a single Lyndon word.

That is, each primitive word is a conjugate (rotation) of a unique Lyndon word.

Counting Lyndon Words

From now on, we consider words over a totally ordered finite alphabet $\mathcal{A}$ consisting of at least two distinct letters, unless stated otherwise.

- We know that the conjugacy class of any primitive word contains a single Lyndon word.

That is, each primitive word is a conjugate (rotation) of a unique Lyndon word.

- So Lyndon words form representatives of conjugacy classes of primitive words.

Counting Lyndon Words

From now on, we consider words over a totally ordered finite alphabet $\mathcal{A}$ consisting of at least two distinct letters, unless stated otherwise.

- We know that the conjugacy class of any primitive word contains a single Lyndon word.

That is, each primitive word is a conjugate (rotation) of a unique Lyndon word.

- So Lyndon words form representatives of conjugacy classes of primitive words.
- Hence the number of Lyndon words of length n over $\mathcal{A}$ is precisely the number of aperiodic (primitive) necklaces of length n over $\mathcal{A}$.

Counting Lyndon Words

From now on, we consider words over a totally ordered finite alphabet $\mathcal{A}$ consisting of at least two distinct letters, unless stated otherwise.

- We know that the conjugacy class of any primitive word contains a single Lyndon word.

That is, each primitive word is a conjugate (rotation) of a unique Lyndon word.

- So Lyndon words form representatives of conjugacy classes of primitive words.
- Hence the number of Lyndon words of length n over $\mathcal{A}$ is precisely the number of aperiodic (primitive) necklaces of length n over $\mathcal{A}$.

How do we count these?

Counting Lyndon Words . . .

- Suppose $\mathcal{A}$ has size $|\mathcal{A}| = k$.

Counting Lyndon Words . . .

- Suppose $\mathcal{A}$ has size $|\mathcal{A}| = k$.
- Let $N_k(n)$ denote the number of primitive necklaces of length n on $\mathcal{A}$.

Counting Lyndon Words . . .

- Suppose $\mathcal{A}$ has **size** $|\mathcal{A}| = k$.
- Let $N_k(n)$ denote the **number of primitive necklaces of length n** on $\mathcal{A}$.
- If $|w| = n$ and $w = v^p = \underbrace{vv \cdots v}_{p \text{ times}}$ where v is a primitive word,

Counting Lyndon Words . . .

- Suppose $\mathcal{A}$ has **size** $|\mathcal{A}| = k$.
- Let $N_k(n)$ denote the **number of primitive necklaces of length n** on $\mathcal{A}$.
- If $|w| = n$ and $w = v^p = \underbrace{vv \cdots v}_{p \text{ times}}$ where v is a primitive word, then
 $n = pd$ where $d = |v|$

Counting Lyndon Words . . .

- Suppose $\mathcal{A}$ has **size** $|\mathcal{A}| = k$.
- Let $N_k(n)$ denote the **number of primitive necklaces of length n** on $\mathcal{A}$.
- If $|w| = n$ and $w = v^p = \underbrace{vv \cdots v}_{p \text{ times}}$ where v is a primitive word, then $n = pd$ where $d = |v|$ and the number of distinct conjugates of w is exactly d .

Counting Lyndon Words . . .

- Suppose $\mathcal{A}$ has **size** $|\mathcal{A}| = k$.
- Let $N_k(n)$ denote the **number of primitive necklaces of length n** on $\mathcal{A}$.
- If $|w| = n$ and $w = v^p = \underbrace{vv \cdots v}_{p \text{ times}}$ where v is a primitive word, then $n = pd$ where $d = |v|$ and the number of distinct conjugates of w is exactly d .
- Hence, the total number of different words of length n on $\mathcal{A}$ is

$$k^n = \sum_{d|n} d \cdot N_k(d)$$

where the sum is over all positive divisors of n .

Counting Lyndon Words . . .

Now, by the well-known **Möbius inversion formula**, we have

$$N_k(n) = \frac{1}{n} \sum_{d|n} \mu(d) \cdot k^{n/d}$$

where μ is the **Möbius function** defined on $\mathbb{N}^+$ as follows:

$$\mu(n) = \begin{cases} 1 & \text{if } n = 1, \\ (-1)^i & \text{if } n = p_1 \cdots p_i \text{ where the } p_i \text{ are distinct primes,} \\ 0 & \text{if } n \text{ is divisible by a square.} \end{cases}$$

Counting Lyndon Words . . .

Now, by the well-known **Möbius inversion formula**, we have

$$N_k(n) = \frac{1}{n} \sum_{d|n} \mu(d) \cdot k^{n/d}$$

where μ is the **Möbius function** defined on $\mathbb{N}^+$ as follows:

$$\mu(n) = \begin{cases} 1 & \text{if } n = 1, \\ (-1)^i & \text{if } n = p_1 \cdots p_i \text{ where the } p_i \text{ are distinct primes,} \\ 0 & \text{if } n \text{ is divisible by a square.} \end{cases}$$

This “**necklace-counting formula**” (often called **Witt’s formula**) was proved by E. Witt in 1937 in connection with the theorem on **free Lie algebras** now called the **Poincaré–Birkhoff–Witt Theorem**.

Counting Lyndon Words . . .

By Witt's formula, the numbers of binary Lyndon words of each length, starting with length 0 (empty word), form the integer sequence

$$1, 2, 1, 2, 3, 6, 9, 18, 30, 56, 99, 186, 335, \dots$$

Counting Lyndon Words . . .

By Witt's formula, the numbers of binary Lyndon words of each length, starting with length 0 (empty word), form the integer sequence

1, 2, 1, 2, 3, 6, 9, 18, 30, 56, 99, 186, 335, . . . [[sequence A001037 in OEIS](#)].

Counting Lyndon Words . . .

By Witt's formula, the numbers of binary Lyndon words of each length, starting with length 0 (empty word), form the integer sequence

1, 2, 1, 2, 3, 6, 9, 18, 30, 56, 99, 186, 335, . . . [[sequence A001037 in OEIS](#)].

Note

- There are as many Lyndon words of length n over an alphabet of size p (a prime) as there are irreducible monic polynomials of degree n over a finite field of characteristic p . [[Gauss, 1900](#)]

Counting Lyndon Words . . .

By Witt's formula, the numbers of binary Lyndon words of each length, starting with length 0 (empty word), form the integer sequence

1, 2, 1, 2, 3, 6, 9, 18, 30, 56, 99, 186, 335, ... [sequence A001037 in OEIS].

Note

- There are as many Lyndon words of length n over an alphabet of size p (a prime) as there are irreducible monic polynomials of degree n over a finite field of characteristic p . [Gauss, 1900]
- However, there is no known bijection between such irreducible polynomials and Lyndon words.

Counting Lyndon Words . . .

By Witt's formula, the numbers of binary Lyndon words of each length, starting with length 0 (empty word), form the integer sequence

1, 2, 1, 2, 3, 6, 9, 18, 30, 56, 99, 186, 335, . . . [sequence A001037 in OEIS].

Note

- There are as many Lyndon words of length n over an alphabet of size p (a prime) as there are irreducible monic polynomials of degree n over a finite field of characteristic p . [Gauss, 1900]
- However, there is no known bijection between such irreducible polynomials and Lyndon words.

We've just seen how to count the number of Lyndon words of each length

Counting Lyndon Words . . .

By Witt's formula, the numbers of binary Lyndon words of each length, starting with length 0 (empty word), form the integer sequence

1, 2, 1, 2, 3, 6, 9, 18, 30, 56, 99, 186, 335, . . . [sequence A001037 in OEIS].

Note

- There are as many Lyndon words of length n over an alphabet of size p (a prime) as there are irreducible monic polynomials of degree n over a finite field of characteristic p . [Gauss, 1900]
- However, there is no known bijection between such irreducible polynomials and Lyndon words.

We've just seen how to count the number of Lyndon words of each length, but how do we generate them?

Counting Lyndon Words . . .

By Witt's formula, the numbers of binary Lyndon words of each length, starting with length 0 (empty word), form the integer sequence

1, 2, 1, 2, 3, 6, 9, 18, 30, 56, 99, 186, 335, . . . [[sequence A001037 in OEIS](#)].

Note

- There are as many Lyndon words of length n over an alphabet of size p (a prime) as there are irreducible monic polynomials of degree n over a finite field of characteristic p . [[Gauss, 1900](#)]
- However, there is **no known bijection between such irreducible polynomials and Lyndon words**.

We've just seen how to count the number of Lyndon words of each length, **but how do we generate them?**

Duval (1988) gave a beautiful, and clever, recursive process that generates Lyndon words of bounded length over a totally ordered finite alphabet.

Counting Lyndon Words . . .

By Witt's formula, the **numbers of binary Lyndon words of each length**, starting with length 0 (empty word), form the integer sequence

1, 2, 1, 2, 3, 6, 9, 18, 30, 56, 99, 186, 335, . . . [[sequence A001037 in OEIS](#)].

Note

- There are as many Lyndon words of length n over an alphabet of size p (a prime) as there are irreducible monic polynomials of degree n over a finite field of characteristic p . [[Gauss, 1900](#)]
- However, there is **no known bijection between such irreducible polynomials and Lyndon words**.

We've just seen how to count the number of Lyndon words of each length, **but how do we generate them?**

Duval (1988) gave a beautiful, and clever, recursive process that generates Lyndon words of bounded length over a totally ordered finite alphabet.

At the core of Duval's efficient algorithm is the following important factorisation theorem . . .

Factorisation of Lyndon words

Theorem (Lyndon)

A word $w \in \mathcal{A}^+$ is a Lyndon word if and only if $w \in \mathcal{A}$ or there exists two Lyndon words u and v such that $w = uv$ and $u \prec v$.

Factorisation of Lyndon words

Theorem (Lyndon)

A word $w \in \mathcal{A}^+$ is a Lyndon word if and only if $w \in \mathcal{A}$ or there exists two Lyndon words u and v such that $w = uv$ and $u \prec v$.

In general, this factorisation is not unique since, for example, $w = aabac$ has two such factorisations:

$$w = (a)(abac) \quad \text{and} \quad w = (aab)(ac).$$

Factorisation of Lyndon words

Theorem (Lyndon)

A word $w \in \mathcal{A}^+$ is a Lyndon word if and only if $w \in \mathcal{A}$ or there exists two Lyndon words u and v such that $w = uv$ and $u \prec v$.

In general, this factorisation is not unique since, for example, $w = aabac$ has two such factorisations:

$$w = (a)(abac) \quad \text{and} \quad w = (aab)(ac).$$

However, there is a unique factorisation of a given Lyndon word as a product uv of two Lyndon words u, v with $u \prec v$, called the **standard factorisation**.

Factorisation of Lyndon words

Theorem (Lyndon)

A word $w \in \mathcal{A}^+$ is a Lyndon word if and only if $w \in \mathcal{A}$ or there exists two Lyndon words u and v such that $w = uv$ and $u \prec v$.

In general, this factorisation is not unique since, for example, $w = aabac$ has two such factorisations:

$$w = (a)(abac) \quad \text{and} \quad w = (aab)(ac).$$

However, there is a unique factorisation of a given Lyndon word as a product uv of two Lyndon words u, v with $u \prec v$, called the standard factorisation.

Theorem (Chen-Fox-Lyndon 1958)

If $w = uv$ is a Lyndon word with v its lexicographically smallest proper suffix, then u and v are also Lyndon words and $u \prec v$.

Factorisation of Lyndon words

Theorem (Lyndon)

A word $w \in \mathcal{A}^+$ is a Lyndon word if and only if $w \in \mathcal{A}$ or there exists two Lyndon words u and v such that $w = uv$ and $u \prec v$.

In general, this factorisation is not unique since, for example, $w = aabac$ has two such factorisations:

$$w = (a)(abac) \quad \text{and} \quad w = (aab)(ac).$$

However, there is a unique factorisation of a given Lyndon word as a product uv of two Lyndon words u, v with $u \prec v$, called the standard factorisation.

Theorem (Chen-Fox-Lyndon 1958)

If $w = uv$ is a Lyndon word with v its lexicographically smallest proper suffix, then u and v are also Lyndon words and $u \prec v$.

So the standard factorisation of a Lyndon word $w = uv$ is obtained by choosing v to be the lexicographically least proper suffix of w

Factorisation of Lyndon words

Theorem (Lyndon)

A word $w \in \mathcal{A}^+$ is a Lyndon word if and only if $w \in \mathcal{A}$ or there exists two Lyndon words u and v such that $w = uv$ and $u \prec v$.

In general, this factorisation is not unique since, for example, $w = aabac$ has two such factorisations:

$$w = (a)(abac) \quad \text{and} \quad w = (aab)(ac).$$

However, there is a unique factorisation of a given Lyndon word as a product uv of two Lyndon words u, v with $u \prec v$, called the standard factorisation.

Theorem (Chen-Fox-Lyndon 1958)

If $w = uv$ is a Lyndon word with v its lexicographically smallest proper suffix, then u and v are also Lyndon words and $u \prec v$.

So the standard factorisation of a Lyndon word $w = uv$ is obtained by choosing v to be the lexicographically least proper suffix of w , which also happens to be the longest proper suffix of w that is Lyndon.

Factorisation of Lyndon words

Theorem (Lyndon)

A word $w \in \mathcal{A}^+$ is a Lyndon word if and only if $w \in \mathcal{A}$ or there exists two Lyndon words u and v such that $w = uv$ and $u \prec v$.

In general, this factorisation is not unique since, for example, $w = aabac$ has two such factorisations:

$$w = (a)(abac) \quad \text{and} \quad w = (aab)(ac).$$

However, there is a unique factorisation of a given Lyndon word as a product uv of two Lyndon words u, v with $u \prec v$, called the standard factorisation.

Theorem (Chen-Fox-Lyndon 1958)

If $w = uv$ is a Lyndon word with v its lexicographically smallest proper suffix, then u and v are also Lyndon words and $u \prec v$.

So the standard factorisation of a Lyndon word $w = uv$ is obtained by choosing v to be the lexicographically least proper suffix of w , which also happens to be the longest proper suffix of w that is Lyndon.

Example: $w = aabac$ has standard factorisation $w = (a)(abac)$.

An Application in Algebra

- In algebraic settings, Lyndon words give rise to **commutators** using **standard factorisation** iteratively.

An Application in Algebra

- In algebraic settings, Lyndon words give rise to **commutators** using **standard factorisation** iteratively.
- For example, the Lyndon word $aababb$ with standard factorisation $(a)(ababb)$ gives rise to the commutator $[a, [[a, b], [[a, b], b]]]$.

An Application in Algebra

- In algebraic settings, Lyndon words give rise to **commutators** using **standard factorisation** iteratively.
- For example, the Lyndon word $aababb$ with standard factorisation $(a)(ababb)$ gives rise to the commutator $[a, [[a, b], [[a, b], b]]]$.
- These commutators can be viewed either as elements of the **free group** with $[x, y] = xyx^{-1}y^{-1}$, or as elements of the **free Lie algebra** with $[x, y] = xy - yx$.

An Application in Algebra

- In algebraic settings, Lyndon words give rise to **commutators** using **standard factorisation** iteratively.
- For example, the Lyndon word $aababb$ with standard factorisation $(a)(ababb)$ gives rise to the commutator $[a, [[a, b], [[a, b], b]]$.
- These commutators can be viewed either as elements of the **free group** with $[x, y] = xyx^{-1}y^{-1}$, or as elements of the **free Lie algebra** with $[x, y] = xy - yx$.

In either case, Lyndon words give rise to a basis of some algebra.

An Application in Algebra

- In algebraic settings, Lyndon words give rise to **commutators** using **standard factorisation** iteratively.
- For example, the Lyndon word $aababb$ with standard factorisation $(a)(ababb)$ gives rise to the commutator $[a, [[a, b], [[a, b], b]]]$.
- These commutators can be viewed either as elements of the **free group** with $[x, y] = xyx^{-1}y^{-1}$, or as elements of the **free Lie algebra** with $[x, y] = xy - yx$.

In either case, Lyndon words give rise to a basis of some algebra.

- For specific details, see [C. Reutenauer, *Free Lie Algebras*, 1993] and [M. Lothaire, *Combinatorics on words*, 1983].

Generation of Lyndon Words

Duval's Algorithm: Generates the Lyndon words over $\mathcal{A}$ of length at most n ($n \geq 2$).

Generation of Lyndon Words

Duval's Algorithm: Generates the Lyndon words over $\mathcal{A}$ of length at most n ($n \geq 2$).

If w is one of the words in the list of Lyndon words up to length n (not equal to $\max(\mathcal{A})$), then the next Lyndon word after w can be found by the following steps:

Generation of Lyndon Words

Duval's Algorithm: Generates the Lyndon words over $\mathcal{A}$ of length at most n ($n \geq 2$).

If w is one of the words in the list of Lyndon words up to length n (not equal to $\max(\mathcal{A})$), then the next Lyndon word after w can be found by the following steps:

- 1 Repeat the letters from w to form a new word x of length exactly n , where the i -th letter of x is the same as the letter at position $i \pmod{|w|}$ in w .

Generation of Lyndon Words

Duval's Algorithm: Generates the Lyndon words over $\mathcal{A}$ of length at most n ($n \geq 2$).

If w is one of the words in the list of Lyndon words up to length n (not equal to $\max(\mathcal{A})$), then the next Lyndon word after w can be found by the following steps:

- 1 Repeat the letters from w to form a new word x of length exactly n , where the i -th letter of x is the same as the letter at position $i \pmod{|w|}$ in w .
- 2 If the last letter of x is $\max(\mathcal{A})$ for the given order on $\mathcal{A}$, remove it, producing a shorter word, and take this to be the new x .

Generation of Lyndon Words

Duval's Algorithm: Generates the Lyndon words over $\mathcal{A}$ of length at most n ($n \geq 2$).

If w is one of the words in the list of Lyndon words up to length n (not equal to $\max(\mathcal{A})$), then the next Lyndon word after w can be found by the following steps:

- 1 Repeat the letters from w to form a new word x of length exactly n , where the i -th letter of x is the same as the letter at position $i \pmod{|w|}$ in w .
- 2 If the last letter of x is $\max(\mathcal{A})$ for the given order on $\mathcal{A}$, remove it, producing a shorter word, and take this to be the new x .
- 3 If the last letter of x is $\max(\mathcal{A})$, then **repeat Step 2**.

Generation of Lyndon Words

Duval's Algorithm: Generates the Lyndon words over $\mathcal{A}$ of length at most n ($n \geq 2$).

If w is one of the words in the list of Lyndon words up to length n (not equal to $\max(\mathcal{A})$), then the next Lyndon word after w can be found by the following steps:

- 1 Repeat the letters from w to form a new word x of length exactly n , where the i -th letter of x is the same as the letter at position $i \pmod{|w|}$ in w .
- 2 If the last letter of x is $\max(\mathcal{A})$ for the given order on $\mathcal{A}$, remove it, producing a shorter word, and take this to be the new x .
- 3 If the last letter of x is $\max(\mathcal{A})$, then **repeat Step 2**. Otherwise, replace the last letter of x by its successor in the sorted ordering of the alphabet $\mathcal{A}$.

Generation of Lyndon Words

Duval's Algorithm: Generates the Lyndon words over $\mathcal{A}$ of length at most n ($n \geq 2$).

If w is one of the words in the list of Lyndon words up to length n (not equal to $\max(\mathcal{A})$), then the next Lyndon word after w can be found by the following steps:

- 1 Repeat the letters from w to form a new word x of length exactly n , where the i -th letter of x is the same as the letter at position $i \pmod{|w|}$ in w .
- 2 If the last letter of x is $\max(\mathcal{A})$ for the given order on $\mathcal{A}$, remove it, producing a shorter word, and take this to be the new x .
- 3 If the last letter of x is $\max(\mathcal{A})$, then **repeat Step 2**. Otherwise, replace the last letter of x by its successor in the sorted ordering of the alphabet $\mathcal{A}$.

Note: Since, in general, the factorisation of a Lyndon word as a product uv of two Lyndon words u, v with $u \prec v$ is not unique, Duval's algorithm may produce the same Lyndon word more than once.

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b .

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

a

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\boxed{a} \longrightarrow aa$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\boxed{a} \longrightarrow aa \longrightarrow \boxed{ab}$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{c} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \end{array}$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \end{array}$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \longrightarrow b \end{array}$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \longrightarrow b \longrightarrow \end{array}$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \longrightarrow b \longrightarrow \varepsilon \end{array}$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \longrightarrow b \longrightarrow \epsilon \end{array}$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \longrightarrow b \longrightarrow \epsilon \end{array}$$

Updated List: $\mathcal{L} = \{a, b, ab, \dots\}$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \longrightarrow b \longrightarrow \epsilon \end{array}$$

Updated List: $\mathcal{L} = \{a, b, ab, \dots\}$

Lyndon words of length at most 3:

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \longrightarrow b \longrightarrow \epsilon \end{array}$$

Updated List: $\mathcal{L} = \{a, b, ab, \dots\}$

Lyndon words of length at most 3:

$$\boxed{a}$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \longrightarrow b \longrightarrow \epsilon \end{array}$$

Updated List: $\mathcal{L} = \{a, b, ab, \dots\}$

Lyndon words of length at most 3:

$$\boxed{a} \longrightarrow aaa$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \longrightarrow b \longrightarrow \epsilon \end{array}$$

Updated List: $\mathcal{L} = \{a, b, ab, \dots\}$

Lyndon words of length at most 3:

$$\boxed{a} \longrightarrow aaa \longrightarrow \boxed{aab}$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \longrightarrow b \longrightarrow \epsilon \end{array}$$

Updated List: $\mathcal{L} = \{a, b, ab, \dots\}$

Lyndon words of length at most 3:

$$\begin{array}{l} \boxed{a} \longrightarrow aaa \longrightarrow \boxed{aab} \\ \boxed{ab} \end{array}$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \longrightarrow b \longrightarrow \epsilon \end{array}$$

Updated List: $\mathcal{L} = \{a, b, ab, \dots\}$

Lyndon words of length at most 3:

$$\begin{array}{l} \boxed{a} \longrightarrow aaa \longrightarrow \boxed{aab} \\ \boxed{ab} \longrightarrow aba \end{array}$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \longrightarrow b \longrightarrow \epsilon \end{array}$$

Updated List: $\mathcal{L} = \{a, b, ab, \dots\}$

Lyndon words of length at most 3:

$$\begin{array}{l} \boxed{a} \longrightarrow aaa \longrightarrow \boxed{aab} \\ \boxed{ab} \longrightarrow aba \longrightarrow \boxed{abb} \end{array}$$

Duval's Algorithm in Action

Let's use the algorithm to generate all the Lyndon words up to length 4 over the alphabet $\{a, b\}$ with $a \prec b$.

We begin with the Lyndon words of length 1: a and b . List: $\mathcal{L} = \{a, b, \dots\}$

Lyndon words of length at most 2:

$$\begin{array}{l} \boxed{a} \longrightarrow aa \longrightarrow \boxed{ab} \\ \boxed{b} \longrightarrow bb \longrightarrow b \longrightarrow \epsilon \end{array}$$

Updated List: $\mathcal{L} = \{a, b, ab, \dots\}$

Lyndon words of length at most 3:

$$\begin{array}{l} \boxed{a} \longrightarrow aaa \longrightarrow \boxed{aab} \\ \boxed{ab} \longrightarrow aba \longrightarrow \boxed{abb} \end{array}$$

Updated List: $\mathcal{L} = \{a, b, ab, aab, abb, \dots\}$

Duval's Algorithm in Action . . .

Lyndon words of length at most 4:

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

a

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$$\boxed{a} \longrightarrow aaaa$$

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$$\boxed{a} \longrightarrow aaaa \longrightarrow \boxed{aaab}$$

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$$\begin{array}{l} \boxed{a} \longrightarrow aaaa \longrightarrow \boxed{aaab} \\ \boxed{ab} \end{array}$$

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$$\begin{array}{l} \boxed{a} \longrightarrow aaaa \longrightarrow \boxed{aaab} \\ \boxed{ab} \longrightarrow abab \end{array}$$

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$$\begin{array}{l} \boxed{a} \longrightarrow aaaa \longrightarrow \boxed{aaab} \\ \boxed{ab} \longrightarrow abab \longrightarrow \cancel{abb} \end{array}$$

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$$\boxed{a} \longrightarrow aaaa \longrightarrow \boxed{aaab}$$

$$\boxed{ab} \longrightarrow abab \longrightarrow \cancel{abb} \text{ (repeat)}$$

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$$\boxed{a} \longrightarrow aaaa \longrightarrow \boxed{aaab}$$

$$\boxed{ab} \longrightarrow abab \longrightarrow \cancel{abb} \text{ (repeat)}$$

$$\boxed{aab}$$

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$$\boxed{a} \longrightarrow aaaa \longrightarrow \boxed{aaab}$$

$$\boxed{ab} \longrightarrow abab \longrightarrow \cancel{abb} \text{ (repeat)}$$

$$\boxed{aab} \longrightarrow aaba$$

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$$\boxed{a} \longrightarrow aaaa \longrightarrow \boxed{aaab}$$

$$\boxed{ab} \longrightarrow abab \longrightarrow \cancel{abb} \text{ (repeat)}$$

$$\boxed{aab} \longrightarrow aaba \longrightarrow \boxed{aabb}$$

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$\boxed{a} \longrightarrow aaaa \longrightarrow \boxed{aaab}$

$\boxed{ab} \longrightarrow abab \longrightarrow \cancel{abb} \text{ (repeat)}$

$\boxed{aab} \longrightarrow aaba \longrightarrow \boxed{aabb}$

$\boxed{abb}$

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$\boxed{a} \longrightarrow aaaa \longrightarrow \boxed{aaab}$

$\boxed{ab} \longrightarrow abab \longrightarrow \cancel{abb} \text{ (repeat)}$

$\boxed{aab} \longrightarrow aaba \longrightarrow \boxed{aabb}$

$\boxed{abb} \longrightarrow abba$

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$\boxed{a} \longrightarrow aaaa \longrightarrow \boxed{aaab}$

$\boxed{ab} \longrightarrow abab \longrightarrow \cancel{abb} \text{ (repeat)}$

$\boxed{aab} \longrightarrow aaba \longrightarrow \boxed{aabb}$

$\boxed{abb} \longrightarrow abba \longrightarrow \boxed{abbb}$

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$$\boxed{a} \longrightarrow aaaa \longrightarrow \boxed{aaab}$$

$$\boxed{ab} \longrightarrow abab \longrightarrow \cancel{abb} \text{ (repeat)}$$

$$\boxed{aab} \longrightarrow aaba \longrightarrow \boxed{aabb}$$

$$\boxed{abb} \longrightarrow abba \longrightarrow \boxed{abbb}$$

Updated List: $\mathcal{L} = \{a, b, ab, aab, abb, aaab, aabb, abbb, \dots\}$

Duval's Algorithm in Action ...

Lyndon words of length at most 4:

$$\boxed{a} \longrightarrow aaaa \longrightarrow \boxed{aaab}$$

$$\boxed{ab} \longrightarrow abab \longrightarrow \cancel{abb} \text{ (repeat)}$$

$$\boxed{aab} \longrightarrow aaba \longrightarrow \boxed{aabb}$$

$$\boxed{abb} \longrightarrow abba \longrightarrow \boxed{abbb}$$

Updated List: $\mathcal{L} = \{a, b, ab, aab, abb, aaab, aabb, abbb, \dots\}$

And on it goes ...

Remarks on Duval's Algorithm

- The worst-case time to generate a successor of a Lyndon word w by Duval's procedure is $O(n)$.

Remarks on Duval's Algorithm

- The worst-case time to generate a successor of a Lyndon word w by Duval's procedure is $O(n)$.
- This can be improved to constant time if the generated words are stored in an array of length n and the construction of x from w is performed by appending letters to w instead of making a new copy of w .

Remarks on Duval's Algorithm

- The worst-case time to generate a successor of a Lyndon word w by Duval's procedure is $O(n)$.
- This can be improved to constant time if the generated words are stored in an array of length n and the construction of x from w is performed by appending letters to w instead of making a new copy of w . [Berstel-Pocchiola 1994]

Remarks on Duval's Algorithm

- The worst-case time to generate a successor of a Lyndon word w by Duval's procedure is $O(n)$.
- This can be improved to constant time if the generated words are stored in an array of length n and the construction of x from w is performed by appending letters to w instead of making a new copy of w . [Berstel-Pocchiola 1994]
- Duval (1983) also developed an algorithm for standard factorisation that runs in linear time and constant space.

Back to Factorisations of Lyndon Words

A famous theorem concerning Lyndon words asserts that every word w can be uniquely factorised as a non-increasing product of Lyndon words ...

Back to Factorisations of Lyndon Words

A famous theorem concerning Lyndon words asserts that every word w can be uniquely factorised as a non-increasing product of Lyndon words ...

Theorem

Any word $w \in \mathcal{A}^+$ may be uniquely written as a non-increasing product of Lyndon words, i.e.,

$$w = \ell_1 \ell_2 \cdots \ell_n$$

where the ℓ_i are Lyndon words such that $\ell_1 \succeq \ell_2 \succeq \cdots \succeq \ell_n$

Back to Factorisations of Lyndon Words

A famous theorem concerning Lyndon words asserts that every word w can be uniquely factorised as a non-increasing product of Lyndon words ...

Theorem

Any word $w \in \mathcal{A}^+$ may be uniquely written as a non-increasing product of Lyndon words, i.e.,

$$w = \ell_1 \ell_2 \cdots \ell_n$$

where the ℓ_i are Lyndon words such that $\ell_1 \succeq \ell_2 \succeq \cdots \succeq \ell_n$, called the **Lyndon factorisation**.

Back to Factorisations of Lyndon Words

A famous theorem concerning Lyndon words asserts that every word w can be uniquely factorised as a non-increasing product of Lyndon words ...

Theorem

Any word $w \in \mathcal{A}^+$ may be uniquely written as a non-increasing product of Lyndon words, i.e.,

$$w = \ell_1 \ell_2 \cdots \ell_n$$

where the ℓ_i are Lyndon words such that $\ell_1 \succeq \ell_2 \succeq \cdots \succeq \ell_n$, called the **Lyndon factorisation**.

Example: *abaacaab*

Back to Factorisations of Lyndon Words

A famous theorem concerning Lyndon words asserts that every word w can be uniquely factorised as a non-increasing product of Lyndon words ...

Theorem

Any word $w \in \mathcal{A}^+$ may be uniquely written as a non-increasing product of Lyndon words, i.e.,

$$w = \ell_1 \ell_2 \cdots \ell_n$$

where the ℓ_i are Lyndon words such that $\ell_1 \succeq \ell_2 \succeq \cdots \succeq \ell_n$, called the **Lyndon factorisation**.

Example: $abaacaab = (ab)(aac)(aab)$

Back to Factorisations of Lyndon Words

A famous theorem concerning Lyndon words asserts that every word w can be uniquely factorised as a non-increasing product of Lyndon words ...

Theorem

Any word $w \in \mathcal{A}^+$ may be uniquely written as a non-increasing product of Lyndon words, i.e.,

$$w = \ell_1 \ell_2 \cdots \ell_n$$

where the ℓ_i are Lyndon words such that $\ell_1 \succeq \ell_2 \succeq \cdots \succeq \ell_n$, called the **Lyndon factorisation**.

Example: $abaacaab = (ab)(aac)(aab)$

The origin of the theorem is unclear

Back to Factorisations of Lyndon Words

A famous theorem concerning Lyndon words asserts that every word w can be uniquely factorised as a non-increasing product of Lyndon words ...

Theorem

Any word $w \in \mathcal{A}^+$ may be uniquely written as a non-increasing product of Lyndon words, i.e.,

$$w = \ell_1 \ell_2 \cdots \ell_n$$

where the ℓ_i are Lyndon words such that $\ell_1 \succeq \ell_2 \succeq \cdots \succeq \ell_n$, called the **Lyndon factorisation**.

Example: $abaacaab = (ab)(aac)(aab)$

The origin of the theorem is unclear – it is usually credited to **Chen-Fox-Lyndon (1958)**, whose paper does not explicitly contain the statement, but it can be recovered from their results.

Back to Factorisations of Lyndon Words

A famous theorem concerning Lyndon words asserts that every word w can be uniquely factorised as a non-increasing product of Lyndon words ...

Theorem

Any word $w \in \mathcal{A}^+$ may be uniquely written as a non-increasing product of Lyndon words, i.e.,

$$w = \ell_1 \ell_2 \cdots \ell_n$$

where the ℓ_i are Lyndon words such that $\ell_1 \succeq \ell_2 \succeq \cdots \succeq \ell_n$, called the **Lyndon factorisation**.

Example: $abaacaab = (ab)(aac)(aab)$

The origin of the theorem is unclear – it is usually credited to **Chen-Fox-Lyndon (1958)**, whose paper does not explicitly contain the statement, but it can be recovered from their results.

We note that the Lyndon factorisation of a word can be computed in linear time [Duval 1983].

Applications of Lyndon Factorisations

- Used as part of a bijective variant of the **Burrows-Wheeler transform** for **data compression**. [Burrow-Wheeler 1994, Gil-Scott 2009, Kufleitner 2009]

Applications of Lyndon Factorisations

- Used as part of a bijective variant of the **Burrows-Wheeler transform** for **data compression**. [Burrow-Wheeler 1994, Gil-Scott 2009, Kufleitner 2009]
- Used in algorithms for **digital geometry**. [Brlek et al. 2009]

Applications of Lyndon Factorisations

- Used as part of a bijective variant of the **Burrows-Wheeler transform** for **data compression**. [Burrow-Wheeler 1994, Gil-Scott 2009, Kufleitner 2009]
- Used in algorithms for **digital geometry**. [Brlek et al. 2009]
- **Connection to de Bruijn sequences**: If one concatenates together, in lexicographical order, all the Lyndon words over an alphabet of size k that have length dividing a given number n , then the result is a **de Bruijn sequence** (1975)

Applications of Lyndon Factorisations

- Used as part of a bijective variant of the **Burrows-Wheeler transform** for **data compression**. [Burrow-Wheeler 1994, Gil-Scott 2009, Kufleitner 2009]
- Used in algorithms for **digital geometry**. [Brlek et al. 2009]
- **Connection to de Bruijn sequences**: If one concatenates together, in lexicographical order, all the Lyndon words over an alphabet of size k that have length dividing a given number n , then the result is a **de Bruijn sequence** (1975) – a **necklace** of length k^n on an alphabet of size k in which each possible word of length n appears exactly once as one of its **factors** (contiguous subwords).

Applications of Lyndon Factorisations

- Used as part of a bijective variant of the **Burrows-Wheeler transform** for **data compression**. [Burrow-Wheeler 1994, Gil-Scott 2009, Kufleitner 2009]
- Used in algorithms for **digital geometry**. [Brlek et al. 2009]
- **Connection to de Bruijn sequences**: If one concatenates together, in lexicographical order, all the Lyndon words over an alphabet of size k that have length dividing a given number n , then the result is a **de Bruijn sequence** (1975) – a **necklace** of length k^n on an alphabet of size k in which each possible word of length n appears exactly once as one of its **factors** (contiguous subwords).
- For example, the concatenation, in lexicographical order, of the Lyndon words on two letters with lengths dividing 4 yields the de Bruijn word $B(2, 4)$:

$$a \cdot aaab \cdot aabb \cdot ab \cdot abbb \cdot b$$

Applications of Lyndon Factorisations

- Used as part of a bijective variant of the **Burrows-Wheeler transform** for **data compression**. [Burrow-Wheeler 1994, Gil-Scott 2009, Kufleitner 2009]
- Used in algorithms for **digital geometry**. [Brlek et al. 2009]
- **Connection to de Bruijn sequences**: If one concatenates together, in lexicographical order, all the Lyndon words over an alphabet of size k that have length dividing a given number n , then the result is a **de Bruijn sequence** (1975) – a **necklace** of length k^n on an alphabet of size k in which each possible word of length n appears exactly once as one of its **factors** (contiguous subwords).
- For example, the concatenation, in lexicographical order, of the Lyndon words on two letters with lengths dividing 4 yields the de Bruijn word $B(2, 4)$:

$$a \cdot aaab \cdot aabb \cdot ab \cdot abbb \cdot b$$

- This fact was discovered by **Fredericksen and Maiorana** in 1978.

Applications of Lyndon Factorisations

- Used as part of a bijective variant of the **Burrows-Wheeler transform** for **data compression**. [Burrow-Wheeler 1994, Gil-Scott 2009, Kufleitner 2009]
- Used in algorithms for **digital geometry**. [Brlek et al. 2009]
- **Connection to de Bruijn sequences**: If one concatenates together, in lexicographical order, all the Lyndon words over an alphabet of size k that have length dividing a given number n , then the result is a **de Bruijn sequence** (1975) – a **necklace** of length k^n on an alphabet of size k in which each possible word of length n appears exactly once as one of its **factors** (contiguous subwords).
- For example, the concatenation, in lexicographical order, of the Lyndon words on two letters with lengths dividing 4 yields the de Bruijn word $B(2, 4)$:

$$a \cdot aaab \cdot aabb \cdot ab \cdot abbb \cdot b$$

- This fact was discovered by **Fredericksen and Maiorana** in 1978.
- Interestingly, there is a relationship between Lyndon words, shift-register sequences, and de Bruijn words [Knuth 2005].

Some Applications of de Bruijn Sequences

- A de Bruijn sequence can be used to shorten a brute-force attack on a PIN-like code lock that does not have an “enter” key and accepts the last n digits entered.

Some Applications of de Bruijn Sequences

- A de Bruijn sequence can be used to shorten a brute-force attack on a PIN-like code lock that does not have an “enter” key and accepts the last n digits entered.

Example

Consider such a digital door lock with a 3-digit code.

Some Applications of de Bruijn Sequences

- A de Bruijn sequence can be used to shorten a brute-force attack on a PIN-like code lock that does not have an “enter” key and accepts the last n digits entered.

Example

Consider such a digital door lock with a 3-digit code.

All possible solution codes are contained exactly once in $B(10, 3)$ — the de Bruijn sequence with alphabet $\{0, 1, \dots, 9\}$ that contains each word of length 3 exactly once.

Some Applications of de Bruijn Sequences

- A de Bruijn sequence can be used to shorten a brute-force attack on a PIN-like code lock that does not have an “enter” key and accepts the last n digits entered.

Example

Consider such a digital door lock with a 3-digit code.

All possible solution codes are contained exactly once in $B(10, 3)$ — the de Bruijn sequence with alphabet $\{0, 1, \dots, 9\}$ that contains each word of length 3 exactly once.

$B(10, 3)$ has length $10^3 = 1000$

Some Applications of de Bruijn Sequences

- A de Bruijn sequence can be used to shorten a brute-force attack on a PIN-like code lock that does not have an “enter” key and accepts the last n digits entered.

Example

Consider such a digital door lock with a 3-digit code.

All possible solution codes are contained exactly once in $B(10, 3)$ — the de Bruijn sequence with alphabet $\{0, 1, \dots, 9\}$ that contains each word of length 3 exactly once.

$B(10, 3)$ has length $10^3 = 1000$, so the number of presses required to open the lock would be at most $1000 + 2 = 1002$ (since the solutions are cyclic)

Some Applications of de Bruijn Sequences

- A de Bruijn sequence can be used to shorten a brute-force attack on a PIN-like code lock that does not have an “enter” key and accepts the last n digits entered.

Example

Consider such a digital door lock with a 3-digit code.

All possible solution codes are contained exactly once in $B(10, 3)$ — the de Bruijn sequence with alphabet $\{0, 1, \dots, 9\}$ that contains each word of length 3 exactly once.

$B(10, 3)$ has length $10^3 = 1000$, so the number of presses required to open the lock would be at most $1000 + 2 = 1002$ (since the solutions are cyclic), compared to up to 4000 presses if all possible codes were tried separately.

Some Applications of de Bruijn Sequences

- A de Bruijn sequence can be used to shorten a brute-force attack on a PIN-like code lock that does not have an “enter” key and accepts the last n digits entered.

Example

Consider such a digital door lock with a 3-digit code.

All possible solution codes are contained exactly once in $B(10, 3)$ — the de Bruijn sequence with alphabet $\{0, 1, \dots, 9\}$ that contains each word of length 3 exactly once.

$B(10, 3)$ has length $10^3 = 1000$, so the number of presses required to open the lock would be at most $1000 + 2 = 1002$ (since the solutions are cyclic), compared to up to 4000 presses if all possible codes were tried separately.

- de Bruijn sequences are also of general use in neuroscience and psychology experiments, and even robotics . . .

Some Applications of de Bruijn Sequences

- A de Bruijn sequence can be used to shorten a brute-force attack on a PIN-like code lock that does not have an “enter” key and accepts the last n digits entered.

Example

Consider such a digital door lock with a 3-digit code.

All possible solution codes are contained exactly once in $B(10, 3)$ — the de Bruijn sequence with alphabet $\{0, 1, \dots, 9\}$ that contains each word of length 3 exactly once.

$B(10, 3)$ has length $10^3 = 1000$, so the number of presses required to open the lock would be at most $1000 + 2 = 1002$ (since the solutions are cyclic), compared to up to 4000 presses if all possible codes were tried separately.

- de Bruijn sequences are also of general use in neuroscience and psychology experiments, and even robotics . . .
- See [Wikipedia](#) for more information and other interesting applications.

Standard Factorisations

- Recall that if $w = uv$ is a Lyndon word with v the longest proper Lyndon suffix of w , then u is also a Lyndon word and $u \prec v$.

Standard Factorisations

- Recall that if $w = uv$ is a Lyndon word with v the longest proper Lyndon suffix of w , then u is also a Lyndon word and $u \prec v$.
- This factorisation of w as an increasing product of two Lyndon words is called the **standard** or **right-standard factorisation** of w .

Standard Factorisations

- Recall that if $w = uv$ is a Lyndon word with v the longest proper Lyndon suffix of w , then u is also a Lyndon word and $u \prec v$.
- This factorisation of w as an increasing product of two Lyndon words is called the **standard** or **right-standard factorisation** of w .
- An alternative **left-standard factorisation** is due to Shirshov (1962) and Viennot (1978).

Standard Factorisations

- Recall that if $w = uv$ is a Lyndon word with v the longest proper Lyndon suffix of w , then u is also a Lyndon word and $u \prec v$.
- This factorisation of w as an increasing product of two Lyndon words is called the **standard** or **right-standard factorisation** of w .
- An alternative **left-standard factorisation** is due to Shirshov (1962) and Viennot (1978).

Theorem (Shirshov 1962, Viennot 1978)

If $w = uv$ is a Lyndon word in $\mathcal{A}^+$ with u the longest proper Lyndon prefix of w , then v is also a Lyndon word and $u \prec v$.

Standard Factorisations

- Recall that if $w = uv$ is a Lyndon word with v the longest proper Lyndon suffix of w , then u is also a Lyndon word and $u \prec v$.
- This factorisation of w as an increasing product of two Lyndon words is called the **standard** or **right-standard factorisation** of w .
- An alternative **left-standard factorisation** is due to Shirshov (1962) and Viennot (1978).

Theorem (Shirshov 1962, Viennot 1978)

If $w = uv$ is a Lyndon word in $\mathcal{A}^+$ with u the longest proper Lyndon prefix of w , then v is also a Lyndon word and $u \prec v$.

Example

The left-standard and right-standard factorisations of $aabaacab$ are:

$$(aabaac)(ab) \quad \text{and} \quad (aab)(aacab).$$

Standard Factorisations . . .

The left-standard and right-standard factorisations of a Lyndon word sometimes coincide.

Standard Factorisations . . .

The left-standard and right-standard factorisations of a Lyndon word sometimes coincide.

Example

The Lyndon word $aabaabab$ has coincidental left-standard and right-standard factorisations: $(aab)(aabab)$.

Standard Factorisations . . .

The left-standard and right-standard factorisations of a Lyndon word sometimes coincide.

Example

The Lyndon word $aabaabab$ has coincidental left-standard and right-standard factorisations: $(aab)(aabab)$.

Let's take a closer look at some “coincidental Lyndon words” over $\{a, b\}$ with $a \prec b$. . .

Standard Factorisations . . .

The left-standard and right-standard factorisations of a Lyndon word sometimes coincide.

Example

The Lyndon word $aabaabab$ has coincidental left-standard and right-standard factorisations: $(aab)(aabab)$.

Let's take a closer look at some “coincidental Lyndon words” over $\{a, b\}$ with $a \prec b$. . .

$$a, b, ab, ab, abb, aaab, abbb, aaaab, aabab, ababb, abbbb, \dots$$

Do you notice any structural property that holds for all these words?

Standard Factorisations . . .

The left-standard and right-standard factorisations of a Lyndon word sometimes coincide.

Example

The Lyndon word $aabaabab$ has coincidental left-standard and right-standard factorisations: $(aab)(aabab)$.

Let's take a closer look at some “coincidental Lyndon words” over $\{a, b\}$ with $a \prec b$. . .

$a, b, ab, ab, a\underline{b}b, a\underline{a}ab, a\underline{b}bb, a\underline{a}aab, a\underline{a}ba, a\underline{b}abb, a\underline{b}bbb, \dots$

How about now?

Standard Factorisations . . .

The left-standard and right-standard factorisations of a Lyndon word sometimes coincide.

Example

The Lyndon word $aabaabab$ has coincidental left-standard and right-standard factorisations: $(aab)(aabab)$.

Let's take a closer look at some “coincidental Lyndon words” over $\{a, b\}$ with $a \prec b$. . .

$a, b, ab, ab, a\underline{b}b, a\underline{a}ab, a\underline{b}bb, a\underline{a}aa, a\underline{a}ba, a\underline{b}abb, a\underline{b}bbb, \dots$

All such words take the form aub where u is a palindrome.

Standard Factorisations . . .

The left-standard and right-standard factorisations of a Lyndon word sometimes coincide.

Example

The Lyndon word $aabaabab$ has coincidental left-standard and right-standard factorisations: $(aab)(aabab)$.

Let's take a closer look at some “coincidental Lyndon words” over $\{a, b\}$ with $a \prec b$. . .

$a, b, ab, ab, a\underline{b}b, a\underline{a}ab, a\underline{b}bb, a\underline{a}aa, a\underline{a}ba, a\underline{bab}b, a\underline{bbb}b, \dots$

All such words take the form aub where u is a palindrome.

But not just any old palindrome . . .

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ =$$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

$$(\text{race})^+ =$$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

$$(\text{race})^+ = \text{racecar}$$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

$$(\text{race})^+ = \text{racecar}$$

- We define $\text{Pal}(\varepsilon) = \varepsilon$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

$$(\text{race})^+ = \text{racecar}$$

- We define $\text{Pal}(\varepsilon) = \varepsilon$, and for any word w and letter x ,

$$\text{Pal}(wx) = (\text{Pal}(w)x)^+.$$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

$$(\text{race})^+ = \text{racecar}$$

- We define $\text{Pal}(\varepsilon) = \varepsilon$, and for any word w and letter x ,

$$\text{Pal}(wx) = (\text{Pal}(w)x)^+.$$

For example:

$$\text{Pal}(\text{abab}) =$$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

$$(\text{race})^+ = \text{racecar}$$

- We define $\text{Pal}(\varepsilon) = \varepsilon$, and for any word w and letter x ,

$$\text{Pal}(wx) = (\text{Pal}(w)x)^+.$$

For example:

$$\text{Pal}(\text{abab}) = \underline{a}$$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

$$(\text{race})^+ = \text{racecar}$$

- We define $\text{Pal}(\varepsilon) = \varepsilon$, and for any word w and letter x ,

$$\text{Pal}(wx) = (\text{Pal}(w)x)^+.$$

For example:

$$\text{Pal}(\text{abab}) = \underline{\text{ab}}$$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

$$(\text{race})^+ = \text{racecar}$$

- We define $\text{Pal}(\varepsilon) = \varepsilon$, and for any word w and letter x ,

$$\text{Pal}(wx) = (\text{Pal}(w)x)^+.$$

For example:

$$\text{Pal}(\text{abab}) = \underline{\text{aba}}$$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

$$(\text{race})^+ = \text{racecar}$$

- We define $\text{Pal}(\varepsilon) = \varepsilon$, and for any word w and letter x ,

$$\text{Pal}(wx) = (\text{Pal}(w)x)^+.$$

For example:

$$\text{Pal}(\text{abab}) = \underline{\text{abaa}}$$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

$$(\text{race})^+ = \text{racecar}$$

- We define $\text{Pal}(\varepsilon) = \varepsilon$, and for any word w and letter x ,

$$\text{Pal}(wx) = (\text{Pal}(w)x)^+.$$

For example:

$$\text{Pal}(\text{abab}) = \underline{\text{a}}\underline{\text{b}}\underline{\text{a}}\underline{\text{a}}\underline{\text{b}}\underline{\text{a}}$$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

$$(\text{race})^+ = \text{racecar}$$

- We define $\text{Pal}(\varepsilon) = \varepsilon$, and for any word w and letter x ,

$$\text{Pal}(wx) = (\text{Pal}(w)x)^+.$$

For example:

$$\text{Pal}(\text{abab}) = \underline{\text{a}}\underline{\text{b}}\underline{\text{a}}\underline{\text{a}}\underline{\text{b}}\underline{\text{a}}\underline{\text{b}}$$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

$$(\text{race})^+ = \text{racecar}$$

- We define $\text{Pal}(\varepsilon) = \varepsilon$, and for any word w and letter x ,

$$\text{Pal}(wx) = (\text{Pal}(w)x)^+.$$

For example:

$$\text{Pal}(\text{abab}) = \underline{\text{a}}\underline{\text{b}}\underline{\text{a}}\underline{\text{a}}\underline{\text{b}}\underline{\text{a}}\underline{\text{a}}\underline{\text{b}}\underline{\text{a}}\underline{\text{a}}\underline{\text{b}}\underline{\text{a}}\underline{\text{a}}\underline{\text{b}}\underline{\text{a}}\underline{\text{a}}\underline{\text{b}}$$

Theorem (Melançon 1999, Berstel-de Luca 1997)

Suppose w is finite word over $\{a, b\}$. Then w is a Lyndon word with the property that its left and right standard factorisations coincide if and only if $w = aub$ (when $a \prec b$) or $w = bua$ (when $b \prec a$) where $u = \text{Pal}(v)$ for some word v over $\{a, b\}$.

- Pal is the **iterated palindromic closure operator** [Justin 2005] defined as follows.
- For a given word v , let v^+ denote the unique shortest palindrome beginning with v , called the **(right-)palindromic closure of v** .

For example:

$$(\text{glen})^+ = \text{glenelg}$$

$$(\text{race})^+ = \text{racecar}$$

- We define $\text{Pal}(\varepsilon) = \varepsilon$, and for any word w and letter x ,

$$\text{Pal}(wx) = (\text{Pal}(w)x)^+.$$

For example:

$$\text{Pal}(\text{abab}) = \underline{a}b\underline{a}a\underline{b}a\underline{b}a$$

$$\longrightarrow \text{coincidental Lyndon word: } a\text{Pal}(\text{abab})b = aabaabab \cdot aabab$$

The previous theorem says that the “coincidental Lyndon words” over $\{a, b\}$ with $a \prec b$ are precisely the words $a\text{Pal}(v)b$ where $v \in \{a, b\}^*$.

The previous theorem says that the “coincidental Lyndon words” over $\{a, b\}$ with $a \prec b$ are precisely the words $aPal(v)b$ where $v \in \{a, b\}^*$.

These words are known as (lower) Christoffel words (named after E. Christoffel 1800's)

The previous theorem says that the “coincidental Lyndon words” over $\{a, b\}$ with $a \prec b$ are precisely the words $aPal(v)b$ where $v \in \{a, b\}^*$.

These words are known as (lower) Christoffel words (named after E. Christoffel 1800's) – they can be constructed geometrically by the coding the horizontal and vertical steps of certain lattice paths ...

The previous theorem says that the “**coincidental Lyndon words**” over $\{a, b\}$ with $a \prec b$ are precisely the words $aPal(v)b$ where $v \in \{a, b\}^*$.

These words are known as (lower) **Christoffel words** (named after E. Christoffel 1800's) – they can be constructed geometrically by the coding the horizontal and vertical steps of certain **lattice paths** ...

- Consider a line (call it ℓ) of the form:

$$y = \frac{p}{q}x$$

where p, q are positive integers with $\gcd(p, q) = 1$.

The previous theorem says that the “**coincidental Lyndon words**” over $\{a, b\}$ with $a \prec b$ are precisely the words $aPal(v)b$ where $v \in \{a, b\}^*$.

These words are known as (lower) **Christoffel words** (named after E. Christoffel 1800's) – they can be constructed geometrically by the coding the horizontal and vertical steps of certain **lattice paths** ...

- Consider a line (call it ℓ) of the form:

$$y = \frac{p}{q}x$$

where p, q are positive integers with $\gcd(p, q) = 1$.

- Let $\mathcal{P}$ denote the path along the integer lattice below the line ℓ that starts at the point $(0, 0)$ and ends at the point (q, p) with the property that the region in the plane enclosed by $\mathcal{P}$ and ℓ contains no other points in $\mathbb{Z} \times \mathbb{Z}$ besides those of the path $\mathcal{P}$.

The previous theorem says that the “**coincidental Lyndon words**” over $\{a, b\}$ with $a \prec b$ are precisely the words $aPal(v)b$ where $v \in \{a, b\}^*$.

These words are known as (lower) **Christoffel words** (named after E. Christoffel 1800's) – they can be constructed geometrically by the coding the horizontal and vertical steps of certain **lattice paths** ...

- Consider a line (call it ℓ) of the form:

$$y = \frac{p}{q}x$$

where p, q are positive integers with $\gcd(p, q) = 1$.

- Let $\mathcal{P}$ denote the path along the integer lattice below the line ℓ that starts at the point $(0, 0)$ and ends at the point (q, p) with the property that the region in the plane enclosed by $\mathcal{P}$ and ℓ contains no other points in $\mathbb{Z} \times \mathbb{Z}$ besides those of the path $\mathcal{P}$.
- The so-called **lower Christoffel word of slope p/q** , denoted by $L(p, q)$, is obtained by coding the steps of the path $\mathcal{P}$.

The previous theorem says that the “**coincidental Lyndon words**” over $\{a, b\}$ with $a \prec b$ are precisely the words $aPal(v)b$ where $v \in \{a, b\}^*$.

These words are known as (lower) **Christoffel words** (named after E. Christoffel 1800's) – they can be constructed geometrically by the coding the horizontal and vertical steps of certain **lattice paths** ...

- Consider a line (call it ℓ) of the form:

$$y = \frac{p}{q}x$$

where p, q are positive integers with $\gcd(p, q) = 1$.

- Let $\mathcal{P}$ denote the path along the integer lattice below the line ℓ that starts at the point $(0, 0)$ and ends at the point (q, p) with the property that the region in the plane enclosed by $\mathcal{P}$ and ℓ contains no other points in $\mathbb{Z} \times \mathbb{Z}$ besides those of the path $\mathcal{P}$.
- The so-called **lower Christoffel word of slope p/q** , denoted by $L(p, q)$, is obtained by coding the steps of the path $\mathcal{P}$.
 - A horizontal step is denoted by the letter a .

The previous theorem says that the “**coincidental Lyndon words**” over $\{a, b\}$ with $a \prec b$ are precisely the words $aPal(v)b$ where $v \in \{a, b\}^*$.

These words are known as (lower) **Christoffel words** (named after E. Christoffel 1800's) – they can be constructed geometrically by the coding the horizontal and vertical steps of certain **lattice paths** ...

- Consider a line (call it ℓ) of the form:

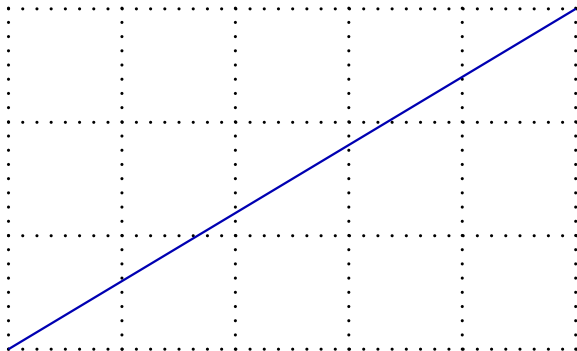
$$y = \frac{p}{q}x$$

where p, q are positive integers with $\gcd(p, q) = 1$.

- Let $\mathcal{P}$ denote the path along the integer lattice below the line ℓ that starts at the point $(0, 0)$ and ends at the point (q, p) with the property that the region in the plane enclosed by $\mathcal{P}$ and ℓ contains no other points in $\mathbb{Z} \times \mathbb{Z}$ besides those of the path $\mathcal{P}$.
- The so-called **lower Christoffel word of slope p/q** , denoted by $L(p, q)$, is obtained by coding the steps of the path $\mathcal{P}$.
 - A horizontal step is denoted by the letter a .
 - A vertical step is denoted by the letter b .

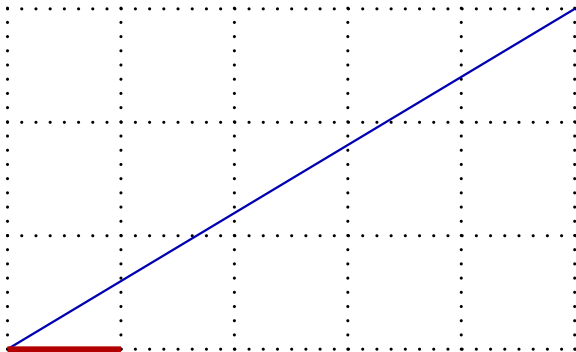
Christoffel Construction by Lattice Paths

Lower Christoffel word of slope $\frac{3}{5}$



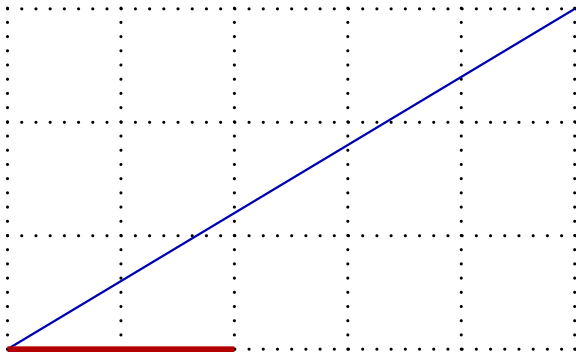
Christoffel Construction by Lattice Paths

Lower Christoffel word of slope $\frac{3}{5}$



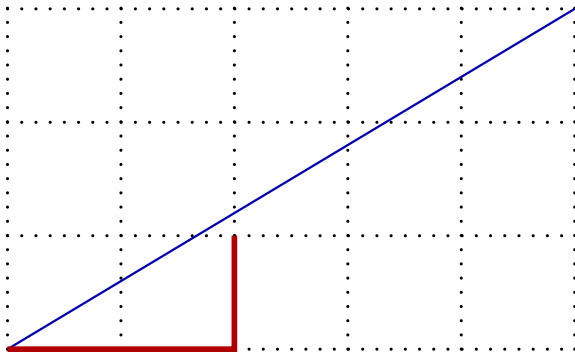
Christoffel Construction by Lattice Paths

Lower Christoffel word of slope $\frac{3}{5}$



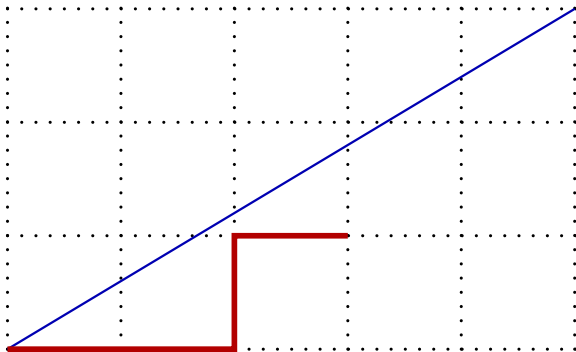
Christoffel Construction by Lattice Paths

Lower Christoffel word of slope $\frac{3}{5}$



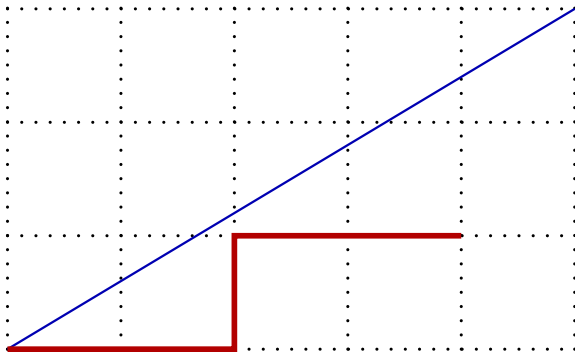
Christoffel Construction by Lattice Paths

Lower Christoffel word of slope $\frac{3}{5}$



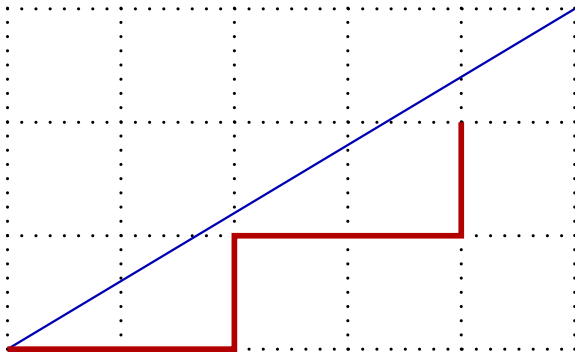
Christoffel Construction by Lattice Paths

Lower Christoffel word of slope $\frac{3}{5}$



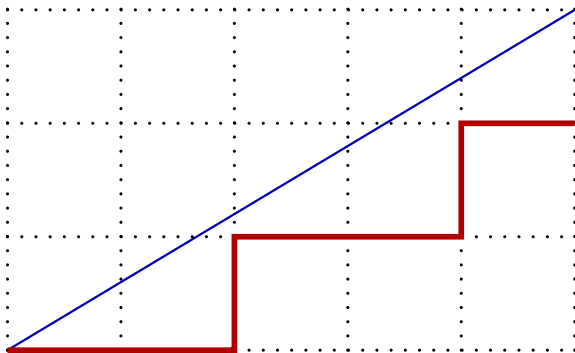
Christoffel Construction by Lattice Paths

Lower Christoffel word of slope $\frac{3}{5}$

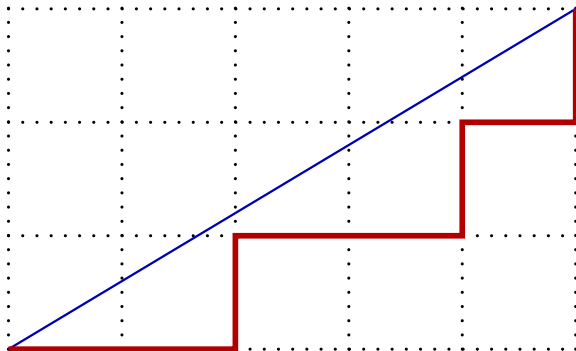


Christoffel Construction by Lattice Paths

Lower Christoffel word of slope $\frac{3}{5}$

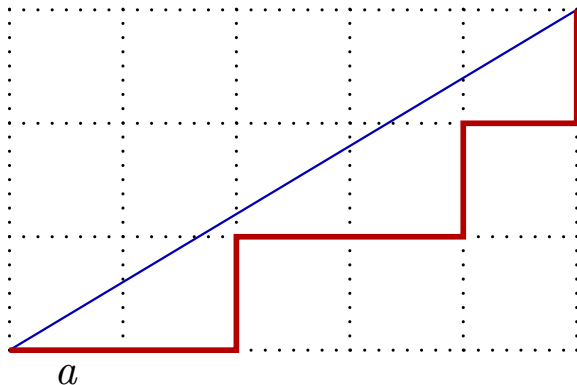


Lower Christoffel word of slope $\frac{3}{5}$



Christoffel Construction by Lattice Paths

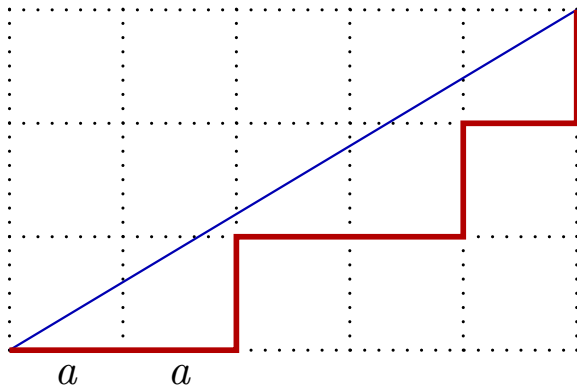
Lower Christoffel word of slope $\frac{3}{5}$



$$L(3, 5) = a$$

Christoffel Construction by Lattice Paths

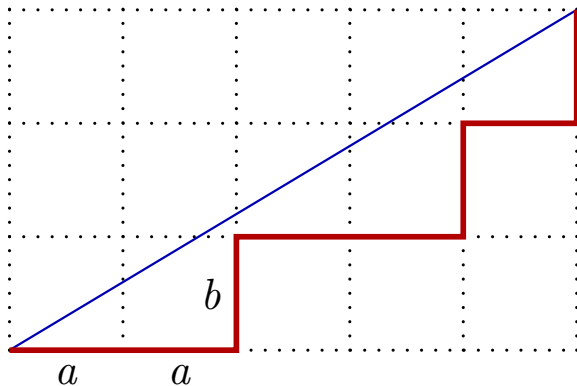
Lower Christoffel word of slope $\frac{3}{5}$



$$L(3, 5) = aa$$

Christoffel Construction by Lattice Paths

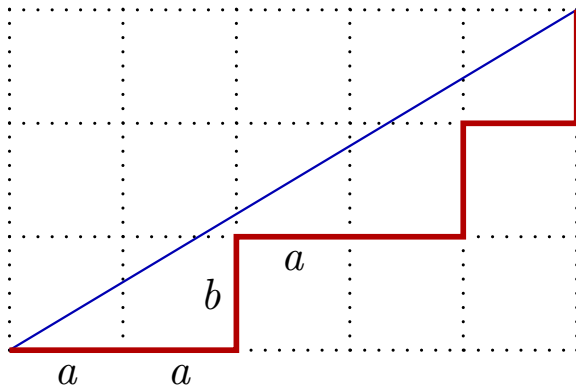
Lower Christoffel word of slope $\frac{3}{5}$



$$L(3, 5) = aab$$

Christoffel Construction by Lattice Paths

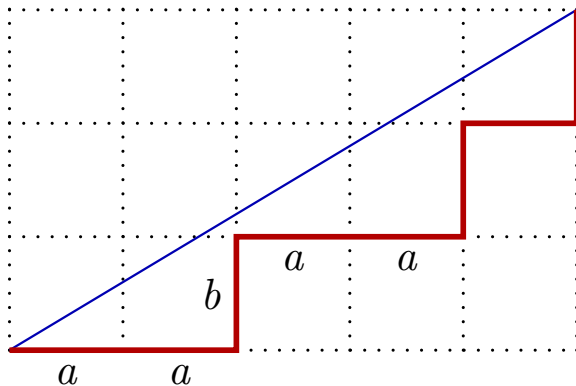
Lower Christoffel word of slope $\frac{3}{5}$



$$L(3, 5) = aaba$$

Christoffel Construction by Lattice Paths

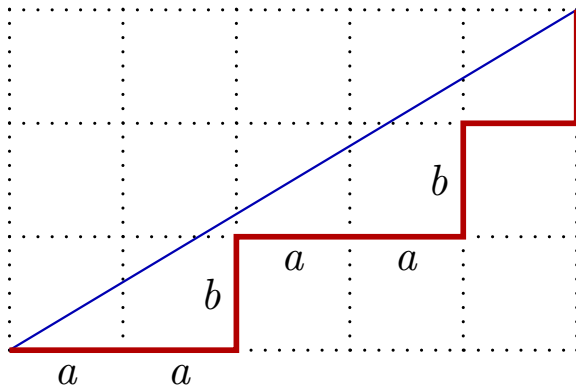
Lower Christoffel word of slope $\frac{3}{5}$



$$L(3, 5) = aabaa$$

Christoffel Construction by Lattice Paths

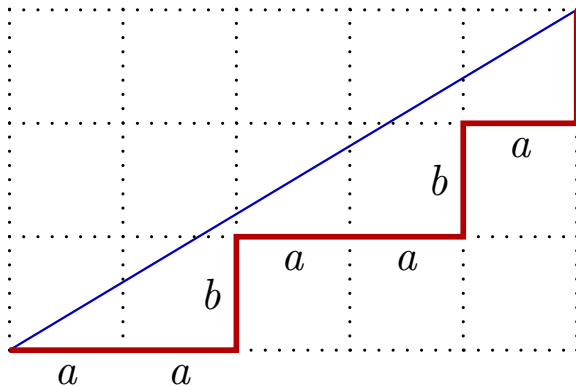
Lower Christoffel word of slope $\frac{3}{5}$



$$L(3, 5) = aabaab$$

Christoffel Construction by Lattice Paths

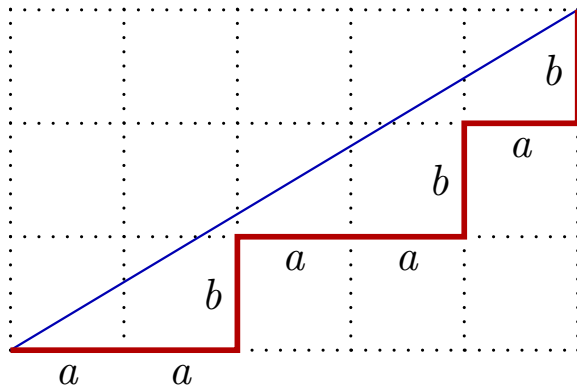
Lower Christoffel word of slope $\frac{3}{5}$



$$L(3, 5) = aabaaba$$

Christoffel Construction by Lattice Paths

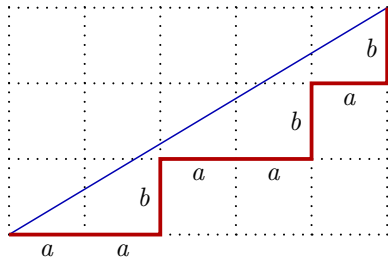
Lower Christoffel word of slope $\frac{3}{5}$



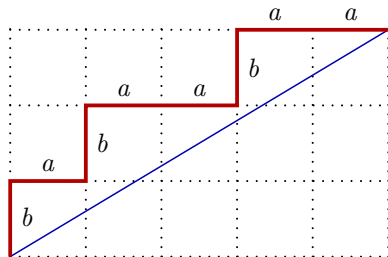
$$L(3, 5) = aabaabab$$

Christoffel Words

Lower & Upper Christoffel words of slope $\frac{3}{5}$



$$L(3, 5) = aabaabab$$



$$U(3, 5) = babaabaa$$

Remarks

- Christoffel words recently came in handy for obtaining a complete description of the minimal intervals containing all the fractional parts $\{x2^n\}$, $n \geq 0$, for some positive real number x .

Remarks

- Christoffel words recently came in handy for obtaining a complete description of the minimal intervals containing all the fractional parts $\{x2^n\}$, $n \geq 0$, for some positive real number x . [\[Allouche-Glen 2009\]](#)

Remarks

- Christoffel words recently came in handy for obtaining a complete description of the minimal intervals containing all the fractional parts $\{x2^n\}$, $n \geq 0$, for some positive real number x . [\[Allouche-Glen 2009\]](#)
- Christoffel words appeared in the literature as early as 1771 in Jean Bernoulli's study of continued fractions.

Remarks

- Christoffel words recently came in handy for obtaining a complete description of the minimal intervals containing all the fractional parts $\{x2^n\}$, $n \geq 0$, for some positive real number x . [\[Allouche-Glen 2009\]](#)
- Christoffel words appeared in the literature as early as 1771 in Jean Bernoulli's study of continued fractions.
- Since then, many relationships between these particular Lyndon words and other areas of mathematics have been revealed.

Remarks

- Christoffel words recently came in handy for obtaining a complete description of the minimal intervals containing all the fractional parts $\{x2^n\}$, $n \geq 0$, for some positive real number x . [Allouche-Glen 2009]
- Christoffel words appeared in the literature as early as 1771 in Jean Bernoulli's study of continued fractions.
- Since then, many relationships between these particular Lyndon words and other areas of mathematics have been revealed.
- For instance, the words in $\{a, b\}^*$ that are conjugates of Christoffel words are exactly the **positive primitive elements of the free group** $F_2 = \langle a, b \rangle$. [Kassel-Reutenauer 2007]

Remarks

- Christoffel words recently came in handy for obtaining a complete description of the minimal intervals containing all the fractional parts $\{x2^n\}$, $n \geq 0$, for some positive real number x . [Allouche-Glen 2009]
- Christoffel words appeared in the literature as early as 1771 in Jean Bernoulli's study of continued fractions.
- Since then, many relationships between these particular Lyndon words and other areas of mathematics have been revealed.
- For instance, the words in $\{a, b\}^*$ that are conjugates of Christoffel words are exactly the **positive primitive elements of the free group** $F_2 = \langle a, b \rangle$. [Kassel-Reutenauer 2007]
- Christoffel words also have connections to the theory of **continued fractions** and **Markoff numbers**.

Remarks

- Christoffel words recently came in handy for obtaining a complete description of the minimal intervals containing all the fractional parts $\{x2^n\}$, $n \geq 0$, for some positive real number x . [Allouche-Glen 2009]
- Christoffel words appeared in the literature as early as 1771 in Jean Bernoulli's study of continued fractions.
- Since then, many relationships between these particular Lyndon words and other areas of mathematics have been revealed.
- For instance, the words in $\{a, b\}^*$ that are conjugates of Christoffel words are exactly the **positive primitive elements of the free group** $F_2 = \langle a, b \rangle$. [Kassel-Reutenauer 2007]
- Christoffel words also have connections to the theory of **continued fractions** and **Markoff numbers**. [Markoff (1879, 1880), Cusick and Flahive (1989), Reutenauer (2006), Berstel et al. (2008)]

Remarks

- Christoffel words recently came in handy for obtaining a complete description of the minimal intervals containing all the fractional parts $\{x2^n\}$, $n \geq 0$, for some positive real number x . [Allouche-Glen 2009]
- Christoffel words appeared in the literature as early as 1771 in Jean Bernoulli's study of continued fractions.
- Since then, many relationships between these particular Lyndon words and other areas of mathematics have been revealed.
- For instance, the words in $\{a, b\}^*$ that are conjugates of Christoffel words are exactly the **positive primitive elements of the free group** $F_2 = \langle a, b \rangle$. [Kassel-Reutenauer 2007]
- Christoffel words also have connections to the theory of **continued fractions** and **Markoff numbers**. [Markoff (1879, 1880), Cusick and Flahive (1989), Reutenauer (2006), Berstel et al. (2008)]
- Nowadays they are studied in the context of **Sturmian words**

Remarks

- Christoffel words recently came in handy for obtaining a complete description of the minimal intervals containing all the fractional parts $\{x2^n\}$, $n \geq 0$, for some positive real number x . [Allouche-Glen 2009]
- Christoffel words appeared in the literature as early as 1771 in Jean Bernoulli's study of continued fractions.
- Since then, many relationships between these particular Lyndon words and other areas of mathematics have been revealed.
- For instance, the words in $\{a, b\}^*$ that are conjugates of Christoffel words are exactly the **positive primitive elements of the free group** $F_2 = \langle a, b \rangle$. [Kassel-Reutenauer 2007]
- Christoffel words also have connections to the theory of **continued fractions** and **Markoff numbers**. [Markoff (1879, 1880), Cusick and Flahive (1989), Reutenauer (2006), Berstel et al. (2008)]
- Nowadays they are studied in the context of **Sturmian words**, but that's a story for another day . . .

Thank You!