In search of a bijection between β-normal 3-indecomposable planar lambda terms and $\beta(0, 1)$-trees

Katarzyna Grygiel
Institute of Theoretical Computer Science
Jagiellonian University in Kraków

Guan-Ru Yu
Institute of Statistical Science
Academia Sinica in Taipei

Computational Logic and Applications 2020

Last year in Versailles (CLA 2019)

Some topological properties of planar lambda terms
by Noam Zeilberger and Jason Reed

results & questions

3-indecomposable planar terms are counted by A000260, which also counts β-normal 2-indecomposable (= unitless) planar terms. Indeed, 3-indecomposable planar terms admit a direct inductive characterization...

$$t ::= x \mid C\{t\}$$
$$C ::= \lambda x.C \mid \bullet\, u$$

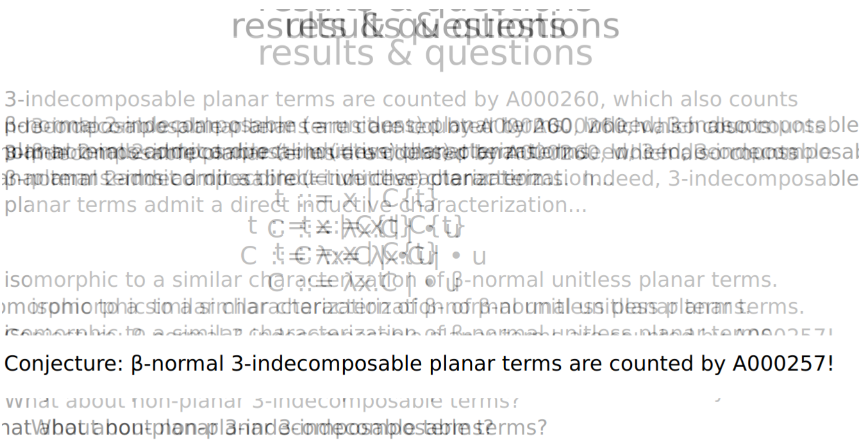
isomorphic to a similar characterization of β-normal unitless planar terms.

Conjecture: β-normal 3-indecomposable planar terms are counted by A000257!

What about non-planar 3-indecomposable terms?

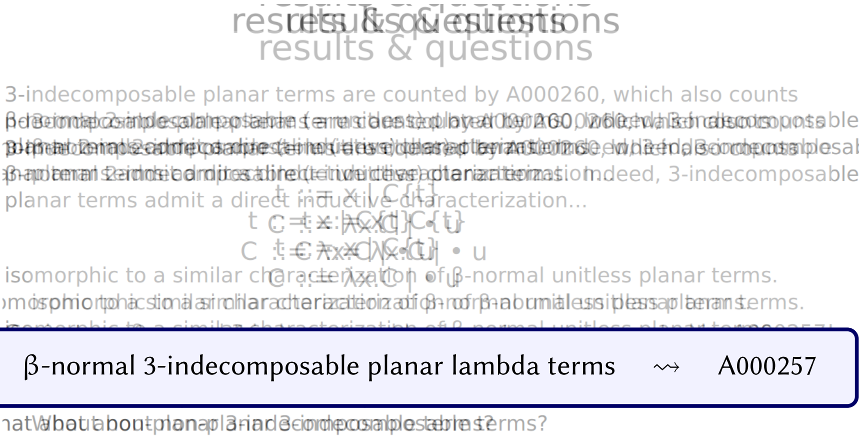
Last year in Versailles (CLA 2019)

Some topological properties of planar lambda terms
by Noam Zeilberger and Jason Reed



Last year in Versailles (CLA 2019)

Some topological properties of planar lambda terms
by Noam Zeilberger and Jason Reed



β-normal 3-indecomposable planar lambda terms ⇝ A000257

A000257

THE ON-LINE ENCYCLOPEDIA OF INTEGER SEQUENCES ®

founded in 1964 by N. J. A. Sloane

Search Hints

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

A000257 Number of rooted bicubic maps: a(n) = (8n-4)*a(n-1)/(n+2). 21
(Formerly M2927 N1175)

1, 1, 3, 12, 56, 288, 1584, 9152, 54912, 339456, 2149888, 13891584, 91287552, 608583680, 4107939840, 28030648320, 193100021760, 1341536993280, 9390758952960, 66182491668480, 469294031831040, 3346270487838720, 23981605162844160, 172667557172477952 (list; graph; refs; listen; history; text; internal format)

OFFSET 0,3

COMMENTS
Number of rooted Eulerian planar maps with n edges. - Valery A. Liskovets, Apr 07 2002
Number of indecomposable 1342-avoiding permutations of length n.
Also counts rooted planar 2-constellations with n digons. - Valery A. Liskovets, Dec 01 2003
a(n) is also the number of rooted planar hypermaps with n darts (darts are semi-edges in the particular case of ordinary maps). - Valery A. Liskovets, Apr 13 2006
Number of "new" intervals in Tamari lattices of size n (see Chapoton paper). - Ralf Stephan, May 08 2007

A000257

THE ON-LINE ENCYCLOPEDIA OF INTEGER SEQUENCES ®

founded in 1964 by N. J. A. Sloane

Search Hints

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

A000257 Number of rooted bicubic maps: a(n) = (8n-4)*a(n-1)/(n+2). 21
(Formerly M2927 N1175)

1, 1, 3, 12, 56, 288, 1584, 9152, 54912, 339456, 2149888, 13891584, 91287552, 608583680, 4107939840, 28030648320, 193100021760, 1341536993280, 9390758952960, 66182491668480, 469294031831040, 3346270487838720, 23981605162844160, 172667557172477952 (list; graph; refs; listen; history; text; internal format)

OFFSET 0,3

COMMENTS
Number of rooted Eulerian planar maps with n edges. - Valery A. Liskovets, Apr 07 2002
Number of indecomposable 1342-avoiding permutations of length n.
Also counts rooted planar 2-constellations with n digons. - Valery A. Liskovets, Dec 01 2003
a(n) is also the number of rooted planar hypermaps with n darts (darts are semi-edges in the particular case of ordinary maps). - Valery A. Liskovets, Apr 13 2006
Number of "new" intervals in Tamari lattices of size n (see Chapoton paper). - Ralf Stephan, May 08 2007

A000257

THE ON–LINE ENCYCLOPEDIA OF INTEGER SEQUENCES ®

founded in 1964 by N. J. A. Sloane

Guan-Ru!

Search Hints

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

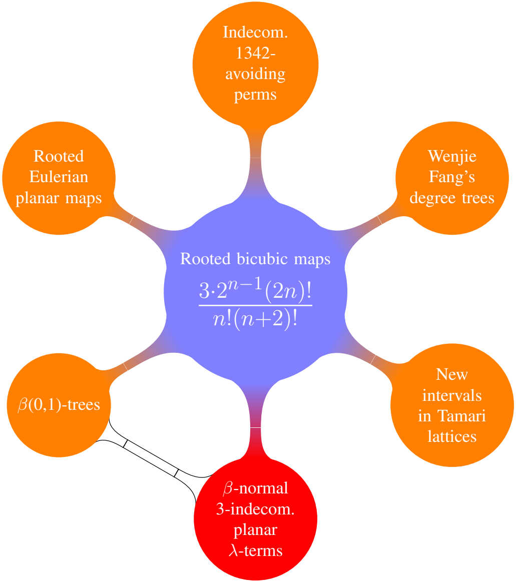
A000257 Number of rooted bicubic maps: a(n) = (8n-4)*a(n-1)/(n+2). 21

(Formerly M2927 N1175)

1, 1, 3, 12, 56, 288, 1584, 9152, 54912, 339456, 2149888, 13891584, 91287552, 608583680, 4107939840, 28030648320, 193100021760, 1341536993280, 9390758952960, 66182491668480, 469294031831040, 3346270487838720, 23981605162844160, 172667557172477952 (list; graph; refs; listen; history; text; internal format)

OFFSET	0,3
COMMENTS	Number of rooted Eulerian planar maps with n edges. - Valery A. Liskovets, Apr 07 2002 Number of indecomposable 1342-avoiding permutations of length n. Also counts rooted planar 2-constellations with n digons. - Valery A. Liskovets, Dec 01 2003 a(n) is also the number of rooted planar hypermaps with n darts (darts are semi-edges in the particular case of ordinary maps). - Valery A. Liskovets, Apr 13 2006 Number of "new" intervals in Tamari lattices of size n (see Chapoton paper). - Ralf Stephan, May 08 2007

OEIS A000257



Lambda terms

Let $\mathcal{V}$ be a countable set of variables. **Lambda terms** are defined by the following grammar:

$$\mathcal{T} ::= \mathcal{V} \mid \lambda \mathcal{V}.\mathcal{T} \mid \mathcal{T}\ \mathcal{T}$$

Lambda terms

Let $\mathcal{V}$ be a countable set of variables. **Lambda terms** are defined by the following grammar:

$$\mathcal{T} ::= \mathcal{V} \mid \lambda\mathcal{V}.\mathcal{T} \mid \mathcal{T}\,\mathcal{T}$$

Example.

$$\lambda a.\,\big(\lambda b.a\big(\lambda c.(c(\lambda d.b)b)\big)\big)\;\big(a\big(\lambda e.\big(\lambda f.\lambda g.e\big)\big(\lambda h.\big((\lambda i.eh)e\big)\big(\lambda j.\lambda k.kj\big)\big)\big)\big)$$

Lambda terms

Let $\mathcal{V}$ be a countable set of variables. **Lambda terms** are defined by the following grammar:

$$\mathcal{T} ::= \mathcal{V} \mid \lambda\mathcal{V}.\mathcal{T} \mid \mathcal{T}\ \mathcal{T}$$

Example.

$$\lambda a.\ (\lambda b.a(\lambda c.(c(\lambda d.b)b))\ (a(\lambda e.(\lambda f.\lambda g.e)(\lambda h.((\lambda i.eh)e)(\lambda j.\lambda k.kj))))$$

The above term is closed. It means that every variable is bound by some lambda.

$$\lambda.\ (\lambda.\bullet(\lambda.(\bullet(\lambda.\bullet)\bullet))\ (\bullet(\lambda.(\lambda.\lambda.\bullet)(\lambda.((\lambda.\bullet\bullet)\bullet)(\lambda.\lambda.\bullet\bullet))))$$

Lambda terms

Let $\mathcal{V}$ be a countable set of variables. **Lambda terms** are defined by the following grammar:

$$\mathcal{T} ::= \mathcal{V} \mid \lambda\mathcal{V}.\mathcal{T} \mid \mathcal{T}\ \mathcal{T}$$

Example.

$$\lambda a.\ \big(\lambda b.a\big(\lambda c.(c(\lambda d.b)b)\big)\big)\ \big(a\big(\lambda e.\big(\lambda f.\lambda g.e\big)\big(\lambda h.\big((\lambda i.eh)e\big)\big(\lambda j.\lambda k.kj\big)\big)\big)\big)$$

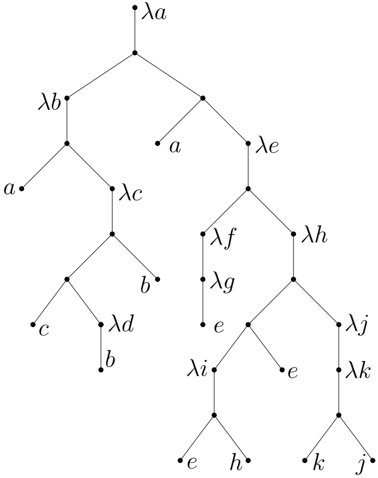
The above term is closed. It means that every variable is bound by some lambda.

$$\textcolor{blue}{\lambda}.\ \big(\textcolor{green}{\lambda}.\textcolor{blue}{\bullet}\big(\textcolor{red}{\lambda}.(\textcolor{red}{\bullet}(\textcolor{brown}{\lambda}.\textcolor{green}{\bullet})\textcolor{green}{\bullet})\big)\big)\ \big(\textcolor{blue}{\bullet}\big(\textcolor{green}{\lambda}.\big(\textcolor{red}{\lambda}.\textcolor{brown}{\lambda}.\textcolor{green}{\bullet}\big)\big(\textcolor{red}{\lambda}.\big((\textcolor{brown}{\lambda}.\textcolor{green}{\bullet}\textcolor{red}{\bullet})\textcolor{green}{\bullet}\big)\big(\textcolor{red}{\lambda}.\textcolor{brown}{\lambda}.\textcolor{brown}{\bullet}\textcolor{red}{\bullet}\big)\big)\big)\big)$$

We will be interested in closed terms only.

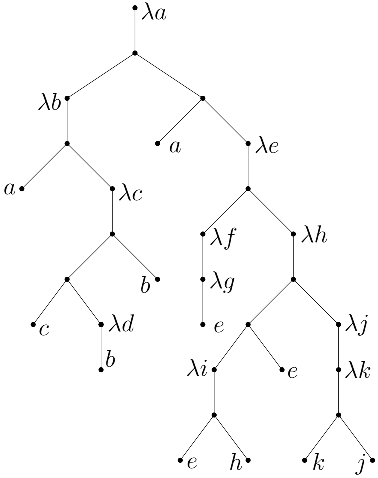
Lambda terms as trees

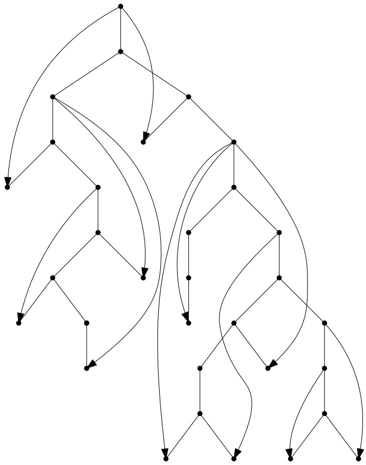
$$\lambda a.\left(\lambda b.a\left(\lambda c.(c(\lambda d.b)b)\right)\right)\left(a\left(\lambda e.(\lambda f.\lambda g.e)\left(\lambda h.((\lambda i.eh)e)(\lambda j.\lambda k.kj)\right)\right)\right)$$



Lambda terms as trees

$$\lambda a.\,\big(\lambda b.a\big(\lambda c.(c(\lambda d.b)b)\big)\big)\;\Big(a\big(\lambda e.(\lambda f.\lambda g.e)\big(\lambda h.((\lambda i.eh)e)(\lambda j.\lambda k.kj)\big)\big)\Big)$$





Planar lambda terms

A lambda term is **linear** iff each variable has exactly one occurrence (*i.e.*, there is a one-to-one correspondence between unary nodes and leaves). The **size** of a linear term is defines as the number of applications (binary nodes).

Planar lambda terms

A lambda term is **linear** iff each variable has exactly one occurrence (*i.e.*, there is a one-to-one correspondence between unary nodes and leaves). The **size** of a linear term is defines as the number of applications (binary nodes).

Let us consider unary-binary trees with the same number of leaves and unary nodes.

Now, in the right-to-left order on leaves, we add a pointer to each leaf linking it with the closest unary node above that has not been chosen yet. The resulting structure has a planar representation on the plane.

Planar lambda terms

A lambda term is **linear** iff each variable has exactly one occurrence (*i.e.*, there is a one-to-one correspondence between unary nodes and leaves). The **size** of a linear term is defines as the number of applications (binary nodes).

Let us consider unary-binary trees with the same number of leaves and unary nodes.

Now, in the right-to-left order on leaves, we add a pointer to each leaf linking it with the closest unary node above that has not been chosen yet. The resulting structure has a planar representation on the plane.

Linear terms that are obtained this way are called **planar**.

Planar lambda terms

A lambda term is **linear** iff each variable has exactly one occurrence (*i.e.*, there is a one-to-one correspondence between unary nodes and leaves). The **size** of a linear term is defines as the number of applications (binary nodes).

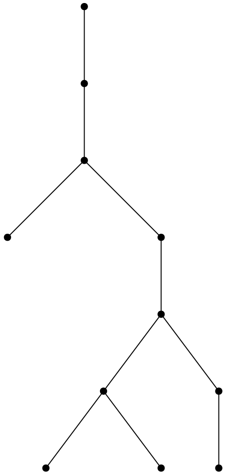
Let us consider unary-binary trees with the same number of leaves and unary nodes.

Now, in the right-to-left order on leaves, we add a pointer to each leaf linking it with the closest unary node above that has not been chosen yet. The resulting structure has a planar representation on the plane.

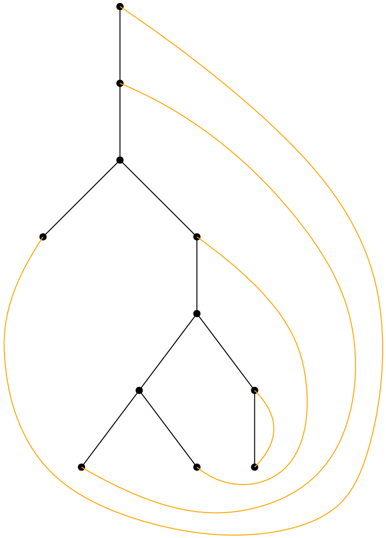
Linear terms that are obtained this way are called **planar**.

Due to planarity, we no longer need to bother about the pointers, as every unary-binary tree corresponds to at most one planar term.

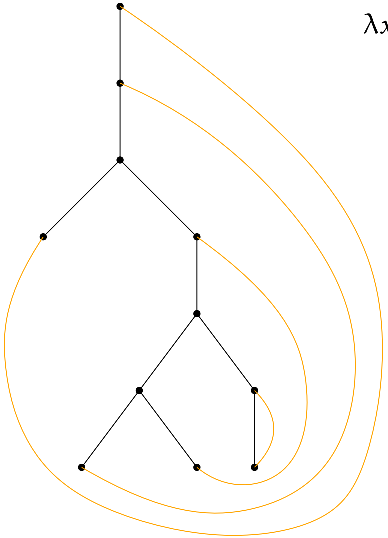
Example of a planar lambda term



Example of a planar lambda term

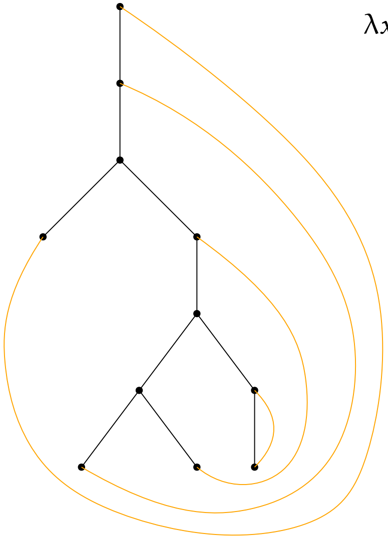


Example of a planar lambda term



$$\lambda xy.x\big(\lambda z.(zy)(\lambda w.w)\big)$$

Example of a planar lambda term



$\lambda xy.x\big(\lambda z.(zy)(\lambda w.w)\big)$

size $= 3$

β-normal 3-indecomposable planar lambda terms

A lambda term is **β-normal** iff it does not contain any subterm of the form $\lambda x.M$. Equivalently, the corresponding tree does not have a left child that is unary.

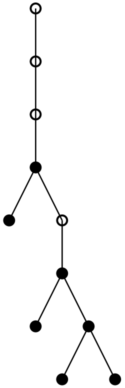
β-normal 3-indecomposable planar lambda terms

A lambda term is **β-normal** iff it does not contain any subterm of the form $\lambda x.M$. Equivalently, the corresponding tree does not have a left child that is unary.

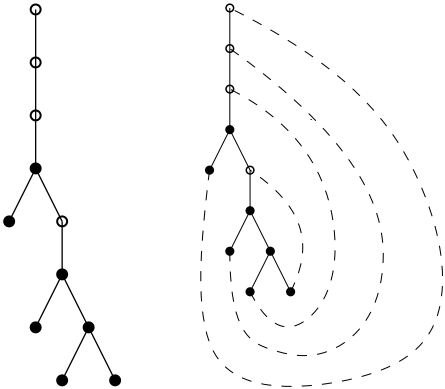
In the obtained structure, let us remove every vertex that corresponds to some leaf by contracting its two incident edges.

If the resulting map is internally 3-connected, *i.e.*, removing any two edges but the two incident to the root does not make the map disconnected, then we say that the original planar term is **3-indecomposable**.

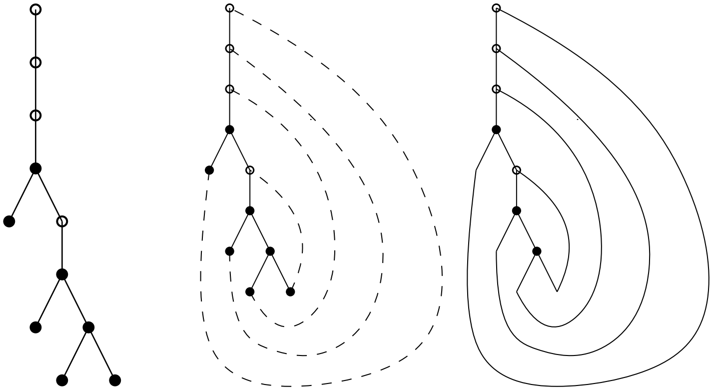
β-normal 3-indecomposable planar lambda terms



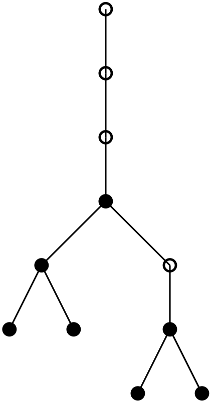
β-normal 3-indecomposable planar lambda terms



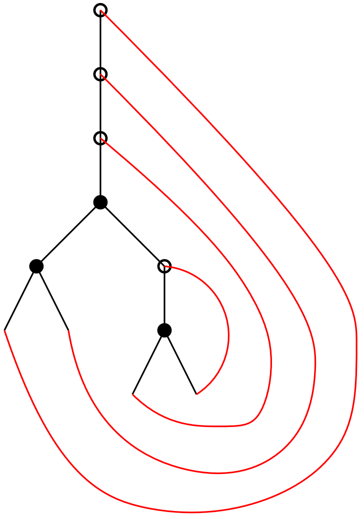
β-normal 3-indecomposable planar lambda terms



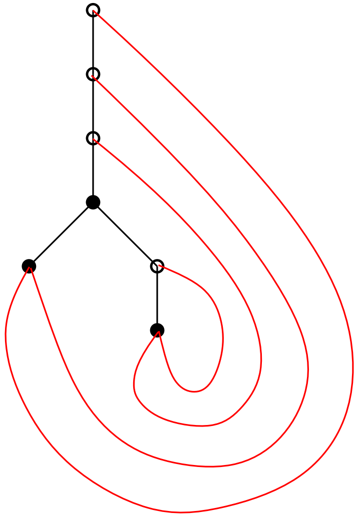
Term violating 3-indecomposability



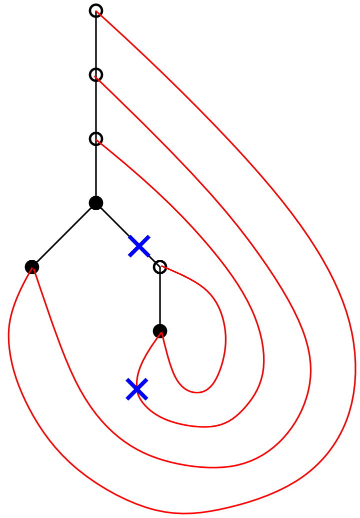
Term violating 3-indecomposability



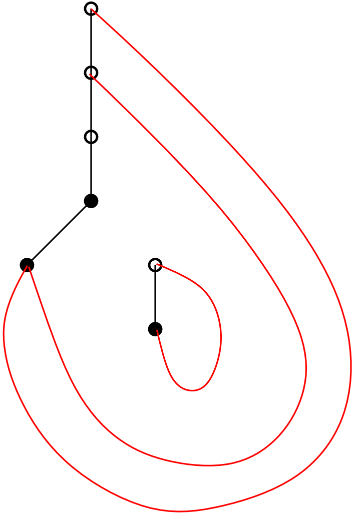
Term violating 3-indecomposability



Term violating 3-indecomposability

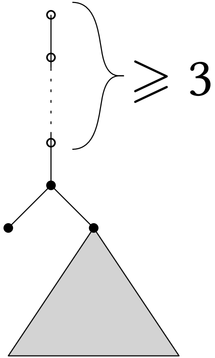


Term violating 3-indecomposability



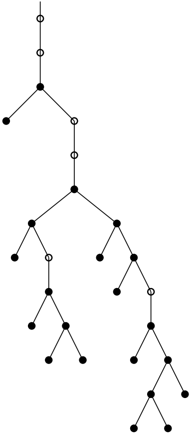
Properties

- Every term of size at least 3 has at least three head lambdas.
- The leftmost maximal binary subtree is just a single leaf.



Properties

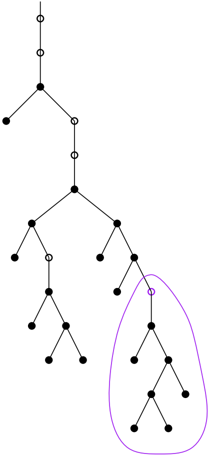
- In every non-trivial maximal unary-binary subtree starting with a unary node the right subtree rooted at the highest binary node has more leaves than the number of its unary nodes increased by the number of head unary nodes.



# 'right' unary nodes	# 'right' leaves

Properties

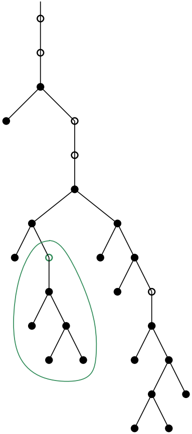
- In every non-trivial maximal unary-binary subtree starting with a unary node the right subtree rooted at the highest binary node has more leaves than the number of its unary nodes increased by the number of head unary nodes.



# 'right' unary nodes	# 'right' leaves
1	3

Properties

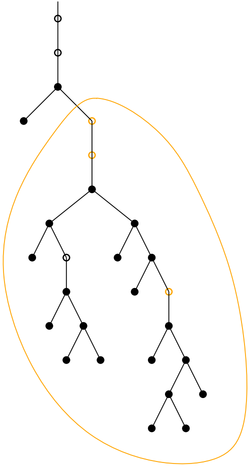
- In every non-trivial maximal unary-binary subtree starting with a unary node the right subtree rooted at the highest binary node has more leaves than the number of its unary nodes increased by the number of head unary nodes.



# ‘right’ unary nodes	# ‘right’ leaves
1	2

Properties

- In every non-trivial maximal unary-binary subtree starting with a unary node the right subtree rooted at the highest binary node has more leaves than the number of its unary nodes increased by the number of head unary nodes.

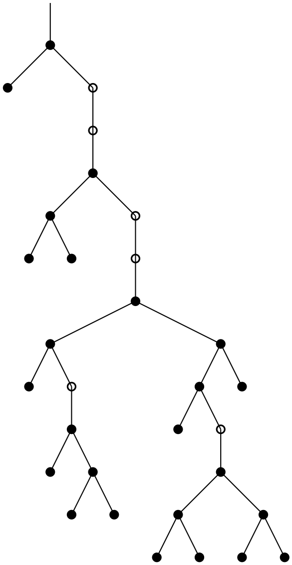


# 'right' unary nodes	# 'right' leaves
3	6

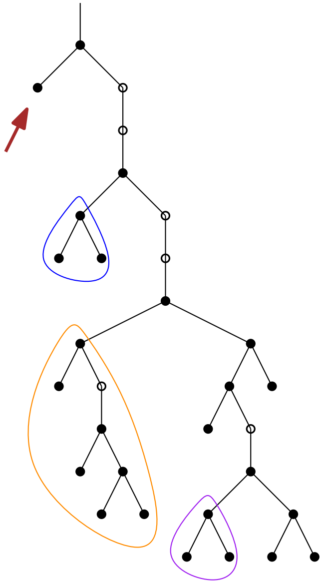
Sending gifts and skipping top unary nodes

Without any loss of information, we perform two operations on the unary-binary trees of our interest.

- Skipping all head unary nodes.
- Sending gifts (as depicted).



Sending gifts and skipping top unary nodes



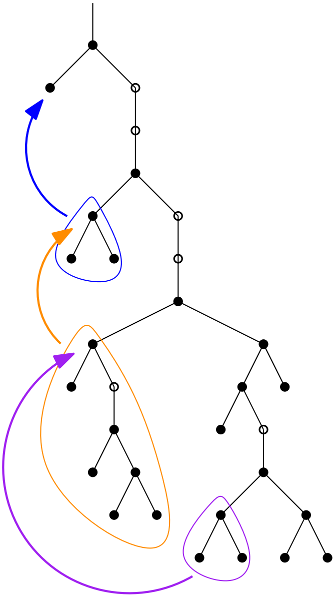
Without any loss of information, we perform two operations on the unary-binary trees of our interest.

- Skipping all head unary nodes.
- Sending gifts (as depicted).

Sending gifts and skipping top unary nodes

Without any loss of information, we perform two operations on the unary-binary trees of our interest.

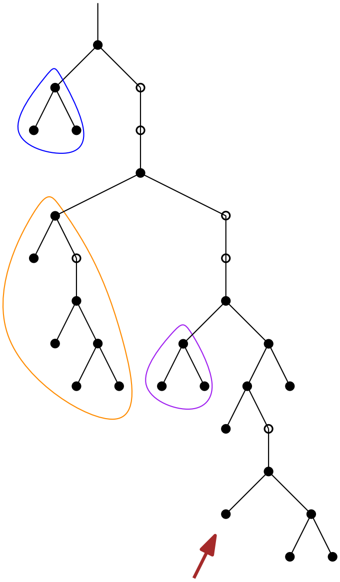
- Skipping all head unary nodes.
- Sending gifts (as depicted).



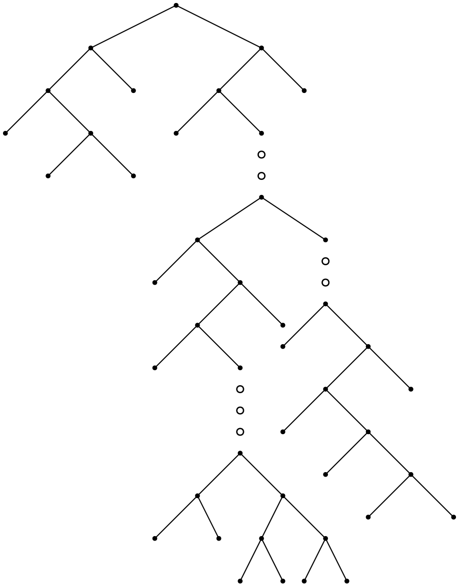
Sending gifts and skipping top unary nodes

Without any loss of information, we perform two operations on the unary-binary trees of our interest.

- Skipping all head unary nodes.
- Sending gifts (as depicted).



Decomposition of trees after pre-processing



Bicubic maps and $\beta(0,1)$-trees

A planar rooted map is **bicubic** iff it is bipartite and cubic.

Bicubic maps and $\beta(0, 1)$-trees

A planar rooted map is **bicubic** iff it is bipartite and cubic.

A **$\beta(0, 1)$-tree** is a rooted plane tree whose nodes are labeled with non-negative integers in the following way:

- leaves have label 0;
- the label of the root is one more than the sum of its children's labels;
- the label of any other node exceeds the sum of its children's labels by at most one.

Bicubic maps and $\beta(0,1)$-trees

A planar rooted map is **bicubic** iff it is bipartite and cubic.

A $\beta(0,1)$**-tree** is a rooted plane tree whose nodes are labeled with non-negative integers in the following way:

- leaves have label 0;
- the label of the root is one more than the sum of its children's labels;
- the label of any other node exceeds the sum of its children's labels by at most one.

Theorem (Claesson, Kitaev, and de Mier)

Bicubic maps are in bijection with $\beta(0,1)$-trees.

Bicubic maps and $\beta(0,1)$-trees

A planar rooted map is **bicubic** iff it is bipartite and cubic.

A $\beta(0,1)$-**tree** is a rooted plane tree whose nodes are labeled with non-negative integers in the following way:

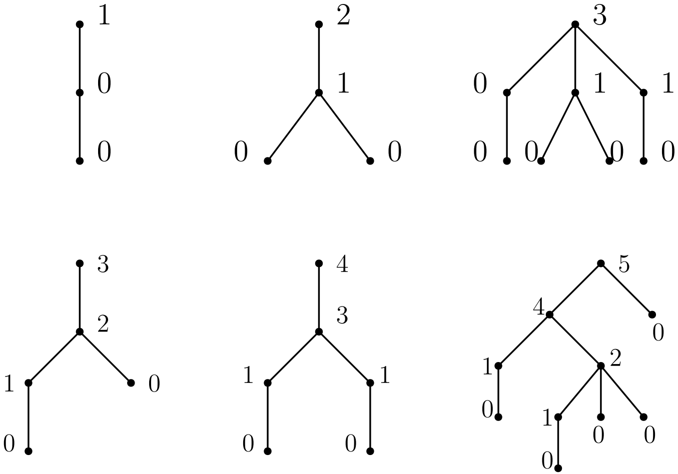
- leaves have label 0;
- the label of the root is one more than the sum of its children's labels;
- the label of any other node exceeds the sum of its children's labels by at most one.

Theorem (Claesson, Kitaev, and de Mier)

Bicubic maps are in bijection with $\beta(0,1)$-trees.

So let's play with $\beta(0,1)$-trees!

Examples of $\beta(0,1)$-trees

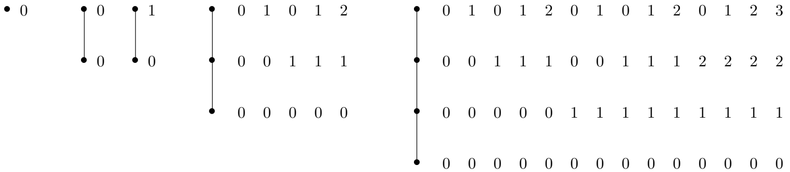


$\beta(0,1)$-bricks

By a **β(0,1)-brick** (or **brick** for brevity) we mean a $\beta(0,1)$-tree with no binary nodes and with its root removed.

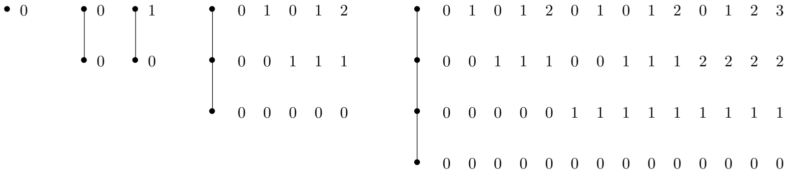
β(0,1)-bricks

By a **β(0,1)-brick** (or **brick** for brevity) we mean a $\beta(0,1)$-tree with no binary nodes and with its root removed.



β(0,1)-bricks

By a **β(0,1)-brick** (or **brick** for brevity) we mean a $\beta(0,1)$-tree with no binary nodes and with its root removed.



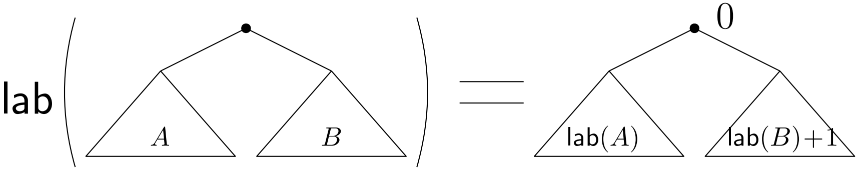
Bricks are enumerated by Catalan numbers.

Labelling binary trees

We label internal nodes in a binary tree by assigning to each of them the number of right-leaning edges on the path from the the root to the node being labelled.

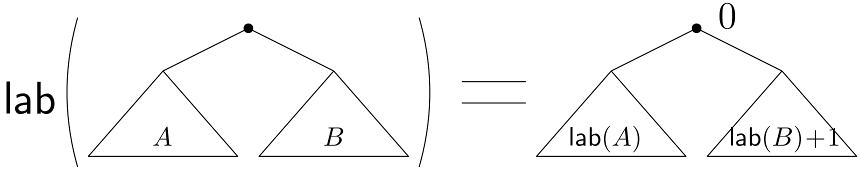
Labelling binary trees

We label internal nodes in a binary tree by assigning to each of them the number of right-leaning edges on the path from the the root to the node being labelled.



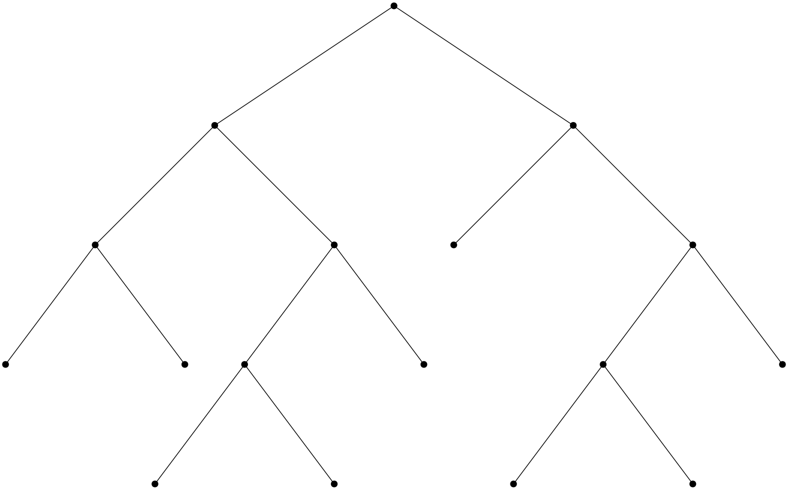
Labelling binary trees

We label internal nodes in a binary tree by assigning to each of them the number of right-leaning edges on the path from the the root to the node being labelled.

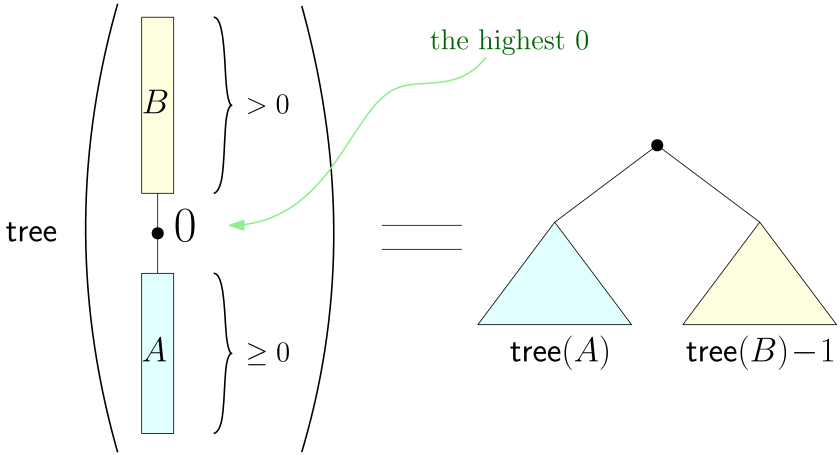


We use the notions ‘binary tree’ and ‘labelled binary tree’ interchangeably.

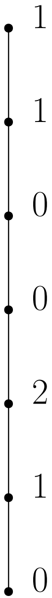
Labelling binary trees



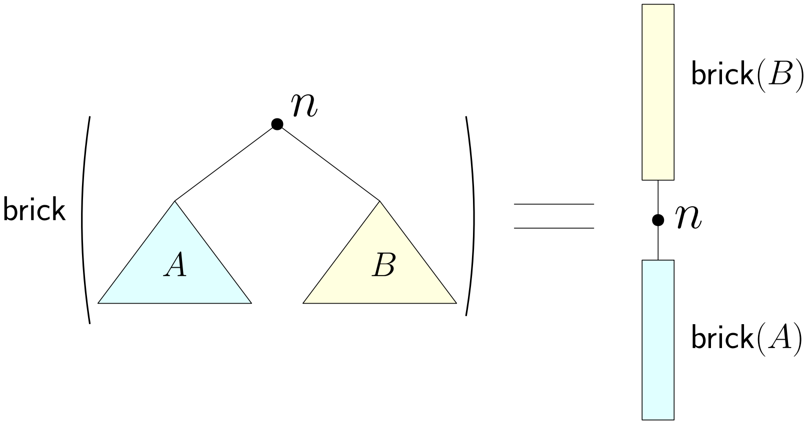
Bricks to trees



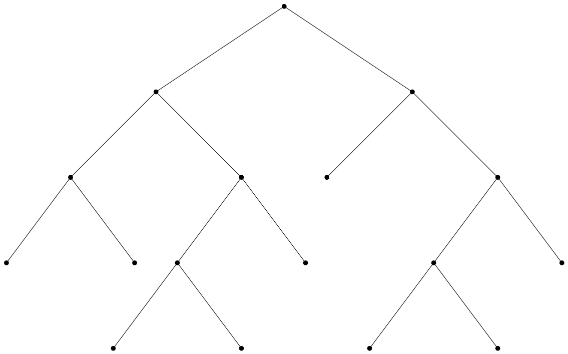
Bricks to trees



Trees to bricks



Trees to bricks



The ultimate goal

- $\beta(0,1)$-trees $\longrightarrow$ (several adjectives) lambda terms

  How to decompose $\beta(0,1)$-trees into bricks and how to combine the corresponding binary trees together in order to obtain lambda terms?

The ultimate goal

- $\beta(0,1)$-trees $\longrightarrow$ (several adjectives) lambda terms

  How to decompose $\beta(0,1)$-trees into bricks and how to combine the corresponding binary trees together in order to obtain lambda terms?

- (several adjectives) lambda terms $\longrightarrow$ $\beta(0,1)$-trees

  How to encode bricks corresponding to the binary trees from the decomposition of lambda terms and how to glue them together in order to obtain $\beta(0,1)$-trees?

The ultimate goal

- $\beta(0,1)$-trees $\longrightarrow$ (several adjectives) lambda terms

  How to decompose $\beta(0,1)$-trees into bricks and how to combine the corresponding binary trees together in order to obtain lambda terms?

- (several adjectives) lambda terms $\longrightarrow$ $\beta(0,1)$-trees

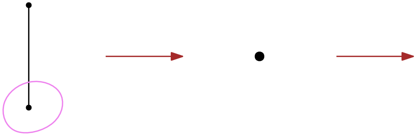
  How to encode bricks corresponding to the binary trees from the decomposition of lambda terms and how to glue them together in order to obtain $\beta(0,1)$-trees?

In the case of our candidate for a bijection…

…the number of maximal binary trees in the decomposition of lambda trees corresponds to the number of leaves in $\beta(0,1)$-trees.

Translation: trivial cases

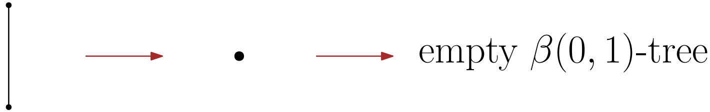
$\lambda x.x$



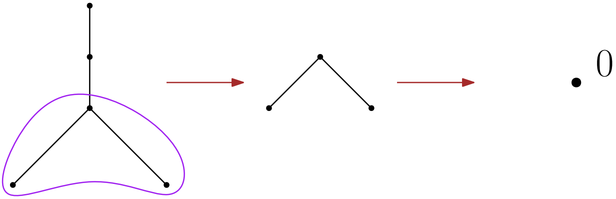
empty $\beta(0,1)$-tree

Translation: trivial cases

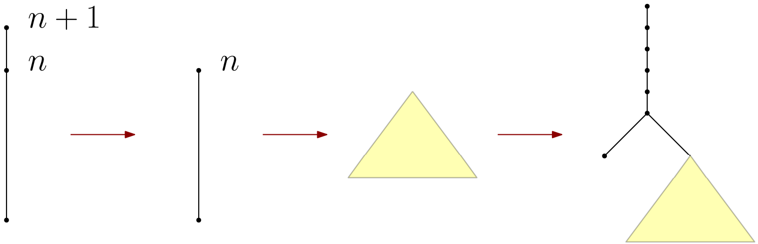
$\lambda x.x$



$\lambda xy.xy$



Translation: β(0, 1)-trees with one leaf



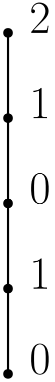
$\beta(0,1)$-tree with 1 leaf

brick

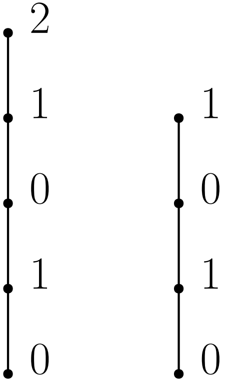
binary tree

lambda term with 1 part

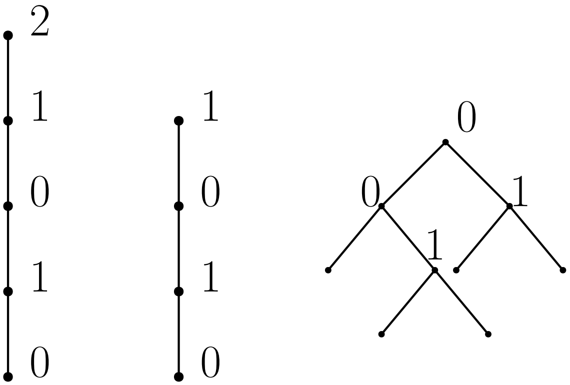
Example: $\beta(0,1)$-trees with one leaf



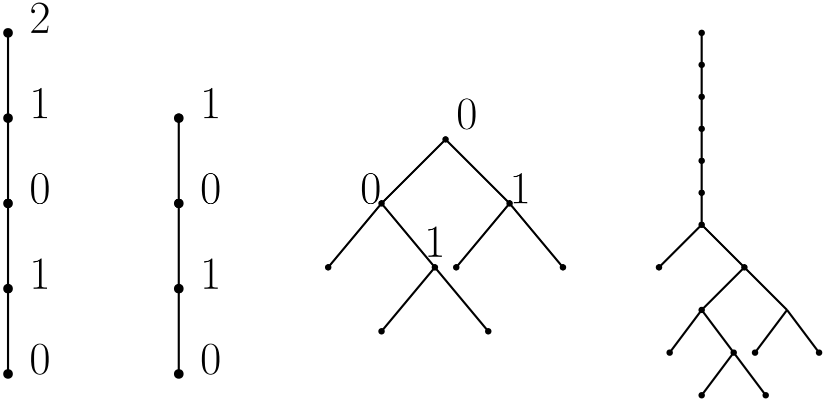
Example: $\beta(0,1)$-trees with one leaf



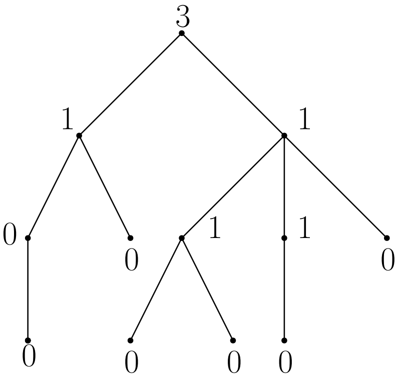
Example: $\beta(0, 1)$-trees with one leaf



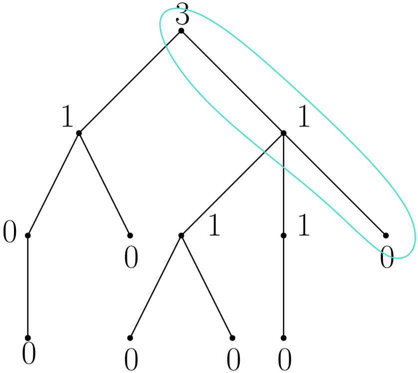
Example: $\beta(0,1)$-trees with one leaf



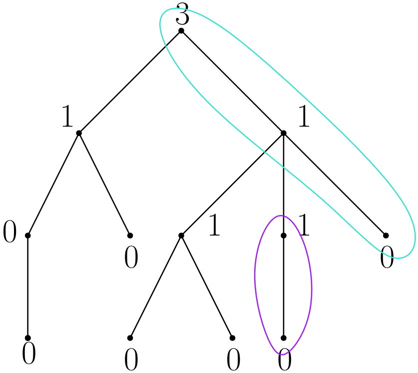
Translation: $\beta(0,1)$-trees with no jumps



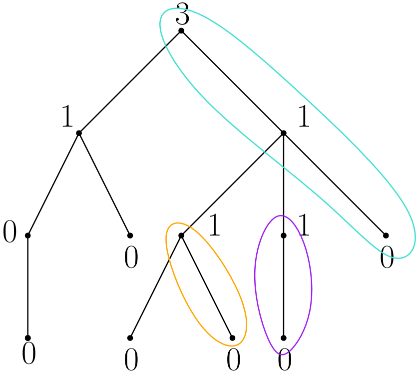
Translation: $\beta(0,1)$-trees with no jumps



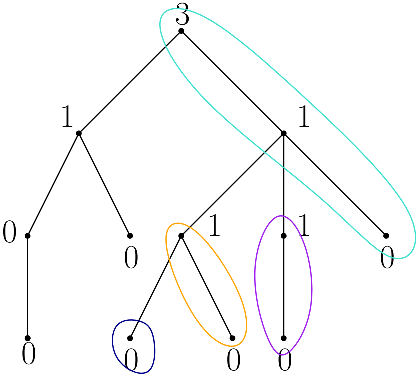
Translation: $\beta(0,1)$-trees with no jumps



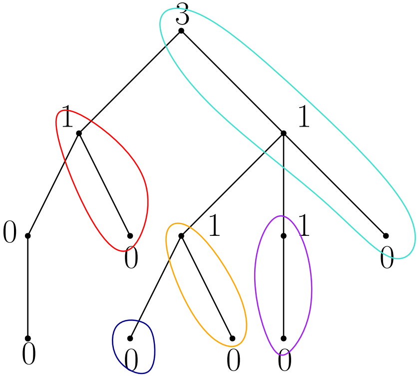
Translation: $\beta(0,1)$-trees with no jumps



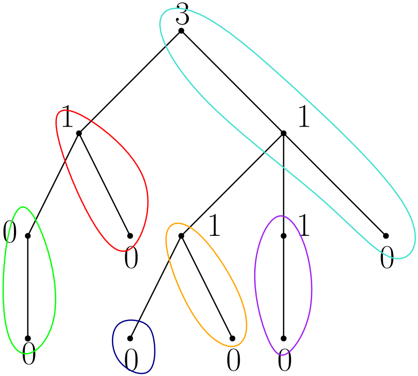
Translation: β(0, 1)-trees with no jumps



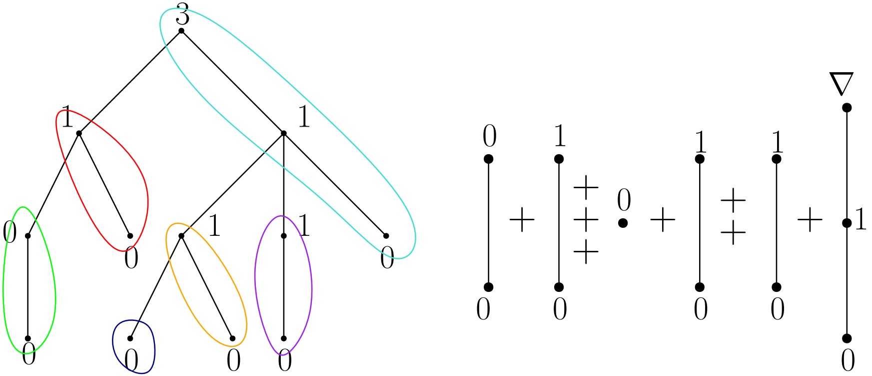
Translation: $\beta(0,1)$-trees with no jumps



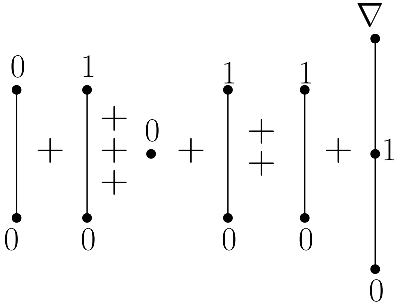
Translation: $\beta(0,1)$-trees with no jumps



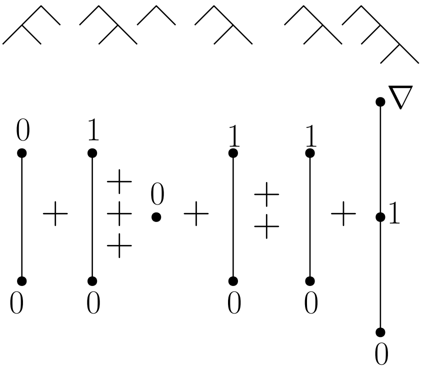
Translation: $\beta(0,1)$-trees with no jumps



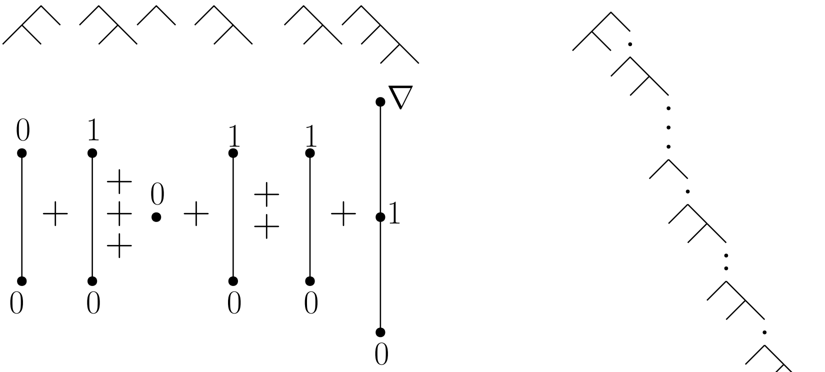
Translation: $\beta(0,1)$-trees with no jumps



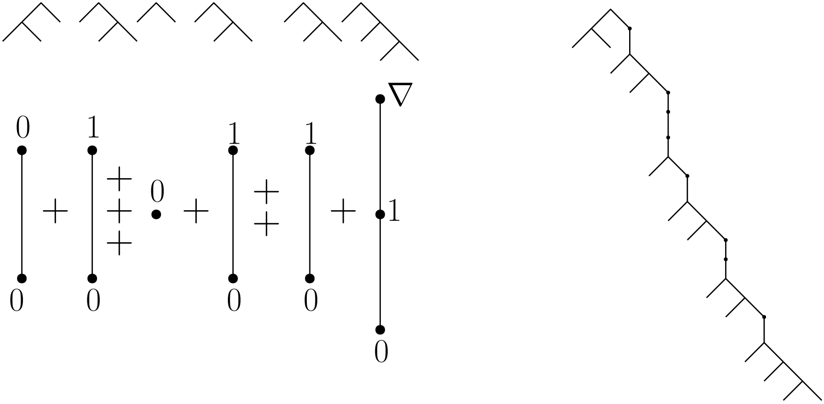
Translation: $\beta(0,1)$-trees with no jumps



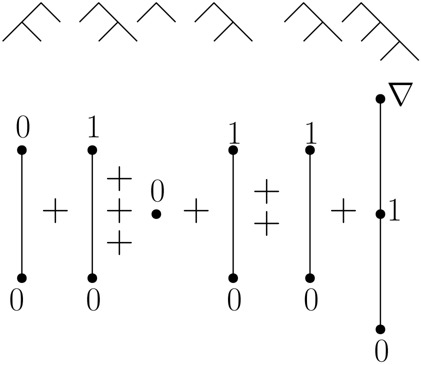
Translation: β(0, 1)-trees with no jumps

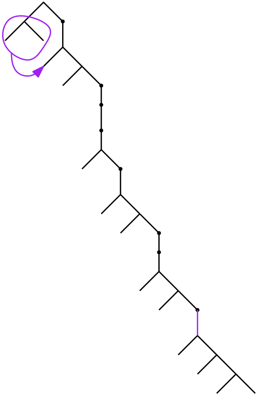


Translation: $\beta(0,1)$-trees with no jumps

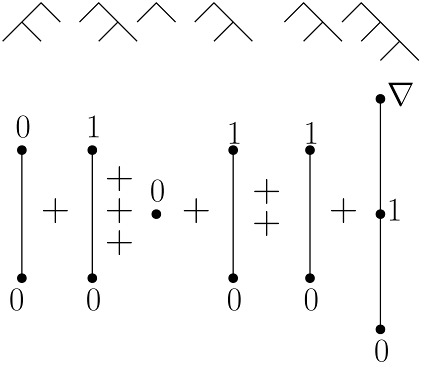


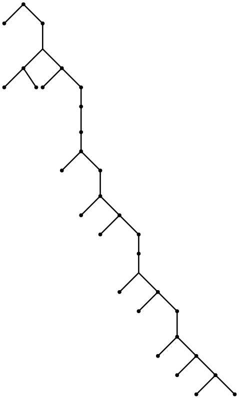
Translation: $\beta(0,1)$-trees with no jumps





Translation: $\beta(0,1)$-trees with no jumps





Jumps

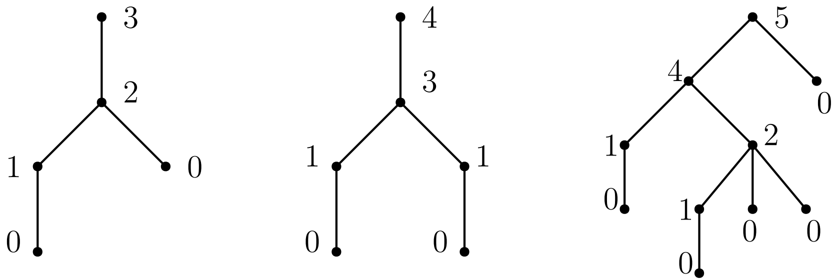
An internal node in a $\beta(0, 1)$-tree is a **jump** iff

- it is not the root and
- its label is by at least 2 greater than the label if its rightmost child.

Jumps

An internal node in a $\beta(0,1)$-tree is a **jump** iff

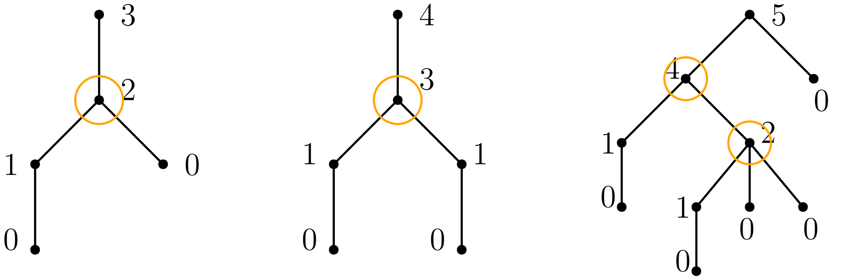
- it is not the root and
- its label is by at least 2 greater than the label if its rightmost child.



Jumps

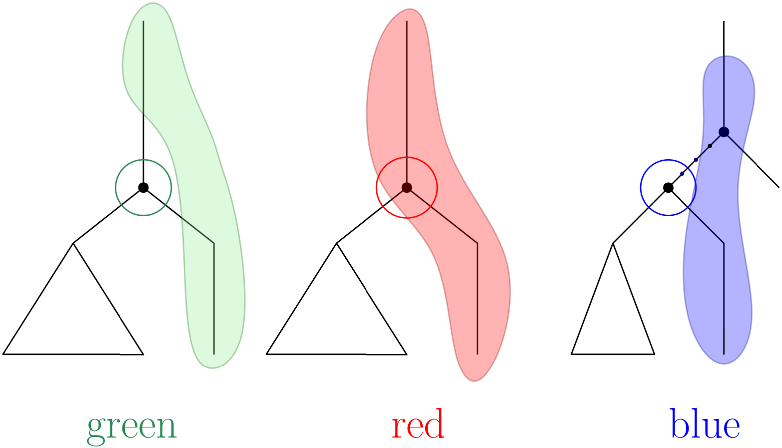
An internal node in a $\beta(0,1)$-tree is a **jump** iff

- it is not the root and
- its label is by at least 2 greater than the label if its rightmost child.

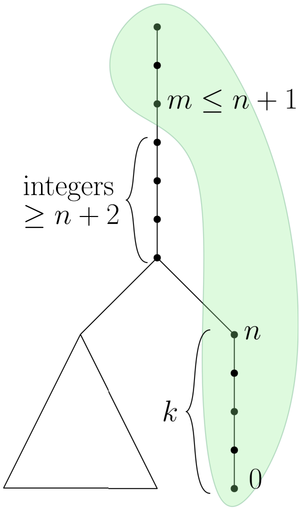


RGB jumps

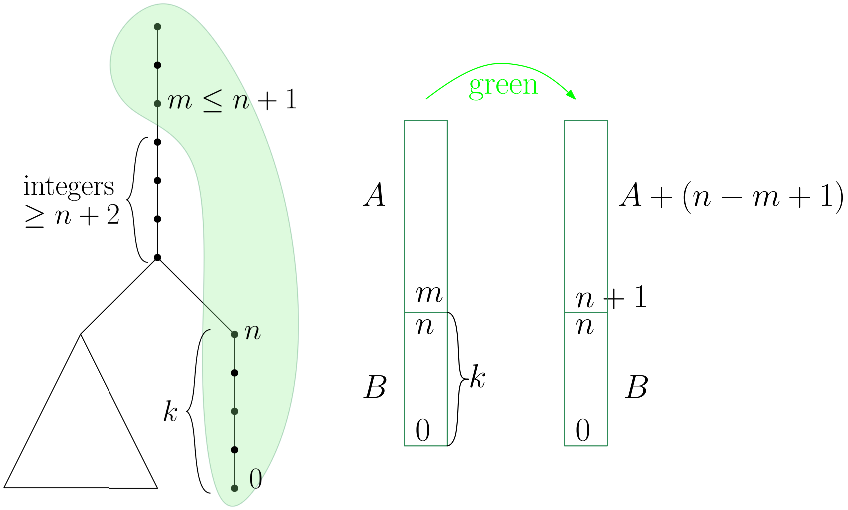
We distinguish three kinds of jumps related to the character of decomposition of $\beta(0, 1)$-trees.



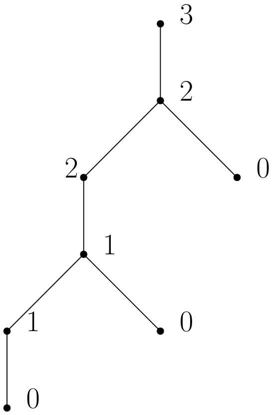
Green jump and green transformation



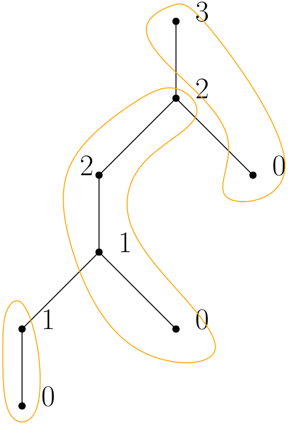
Green jump and green transformation



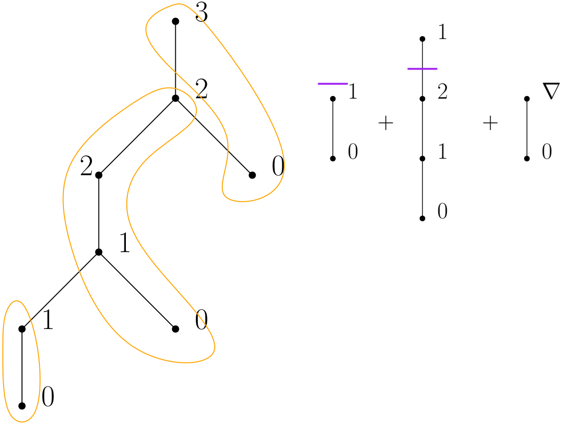
Green example 1



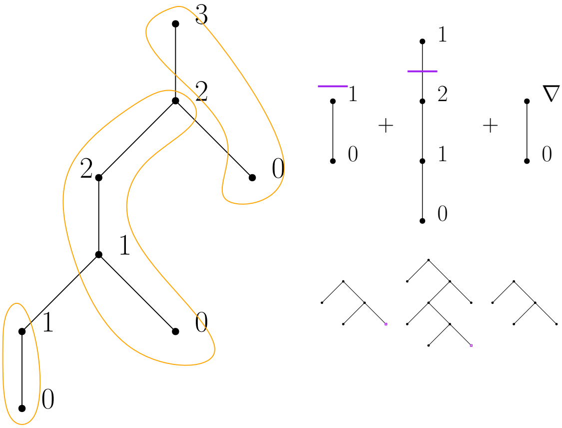
Green example 1



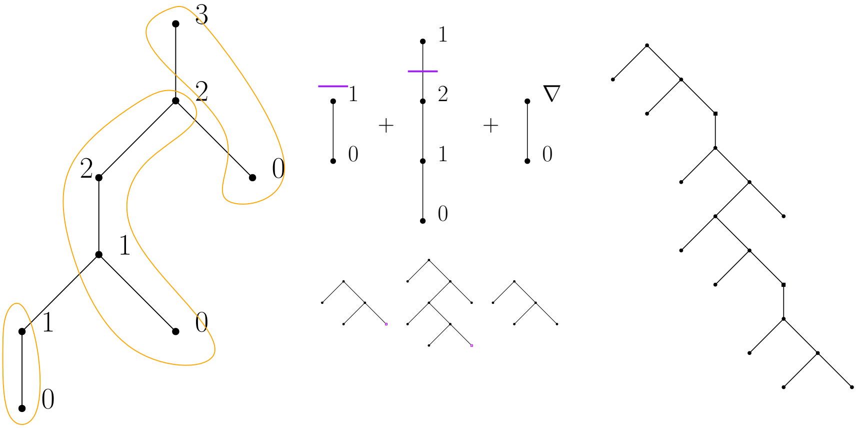
Green example 1



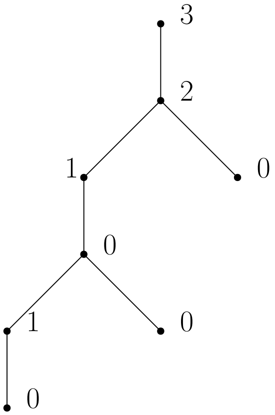
Green example 1



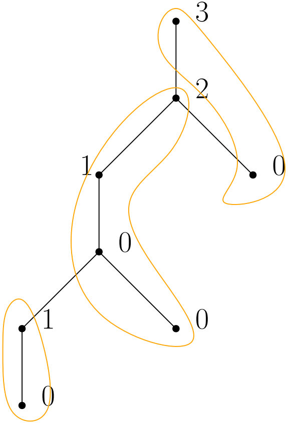
Green example 1



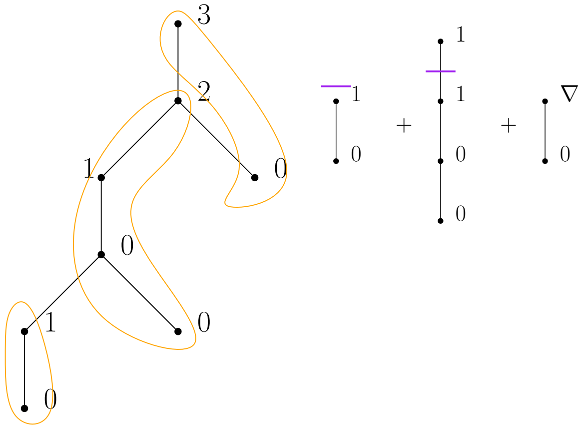
Green example 2



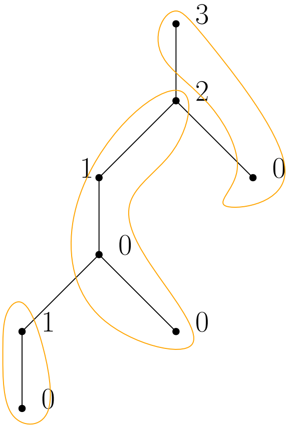
Green example 2

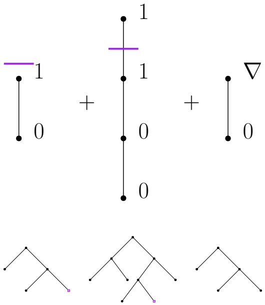


Green example 2

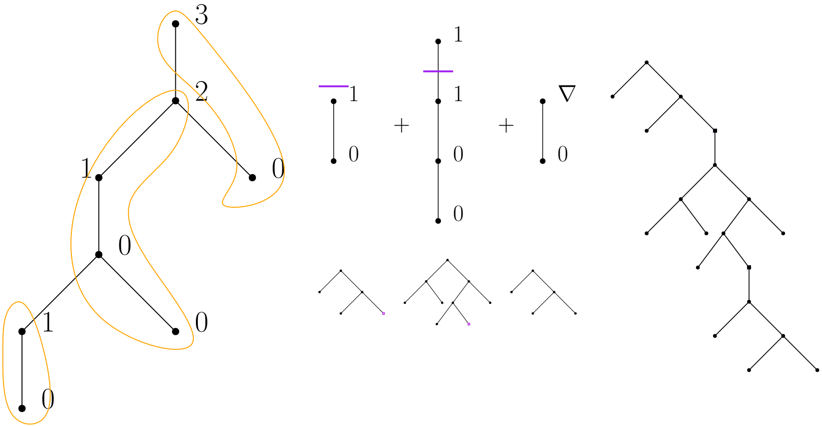


Green example 2

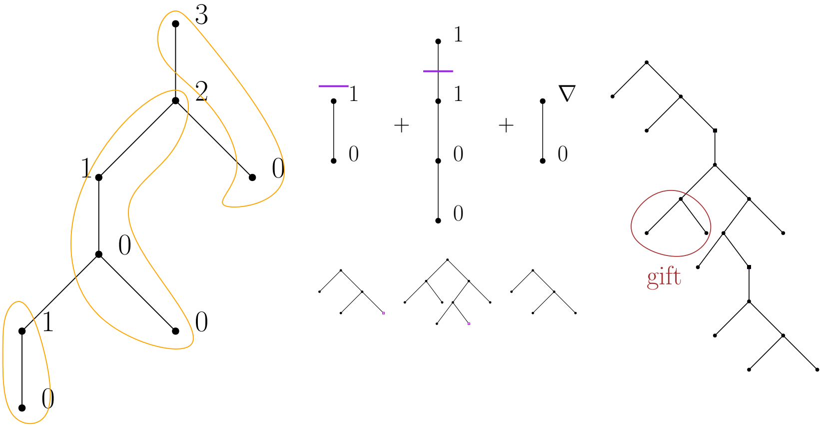




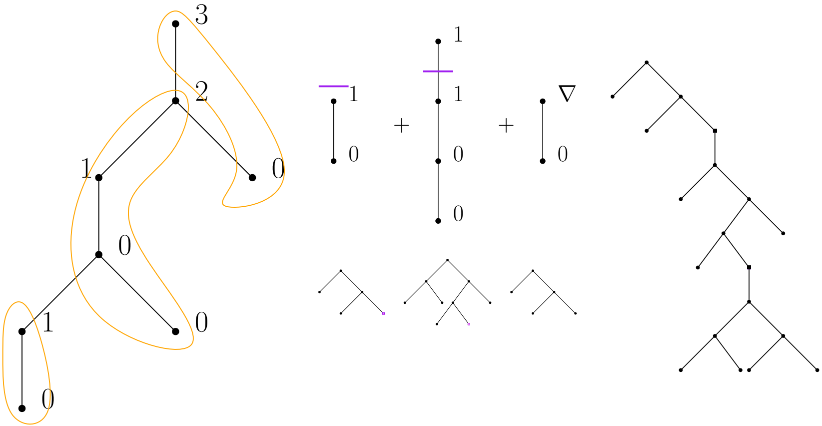
Green example 2



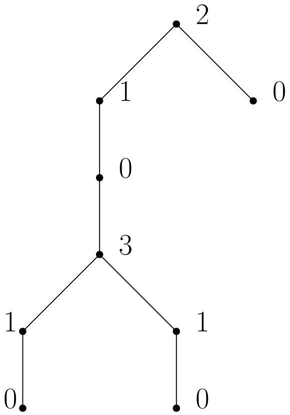
Green example 2



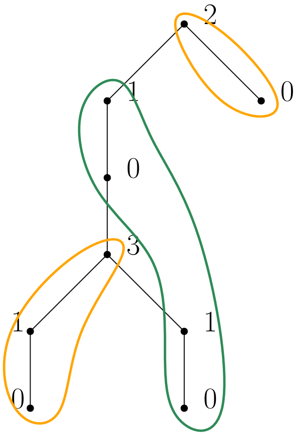
Green example 2



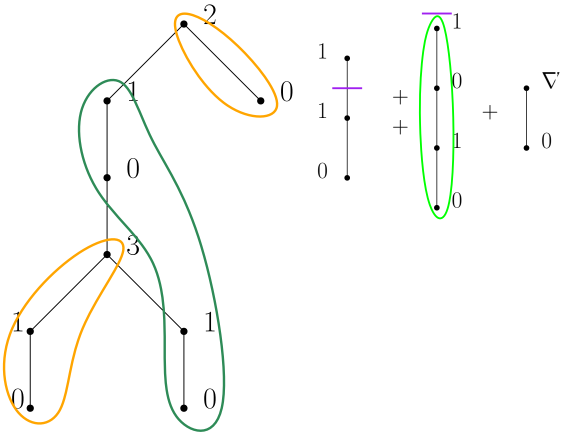
Green example 3



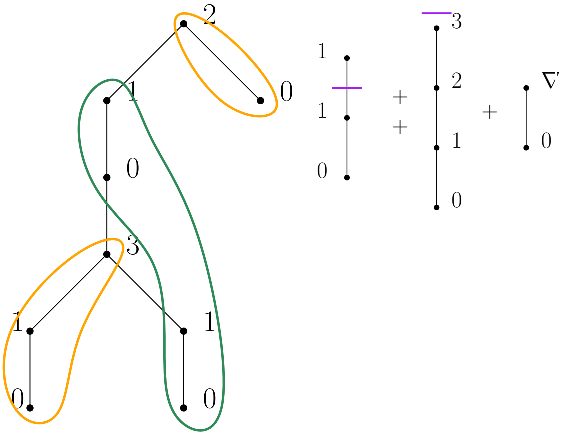
Green example 3



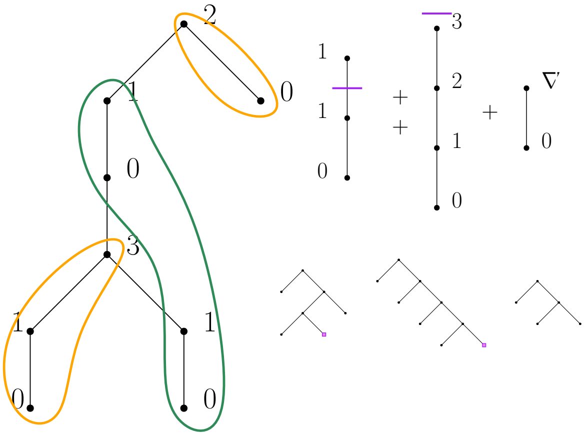
Green example 3



Green example 3



Green example 3



Green example 3