

Pattern avoidance on trees coming from moduli of curves

Andrew Reimer-Berg

Permutation Patterns 2026
July 24, 2026

Pattern Avoidance with words

- Extend classical pattern avoidance to words: $\underline{2466}53\underline{47}$ contains 1221.

Pattern Avoidance with words

- Extend classical pattern avoidance to words: $\underline{2466}53\underline{47}$ contains 1221 .
- Let $\underline{k} = (k_1, \dots, k_n)$ be a composition of n . ($k_i \in \mathbb{Z}_{\geq 0}$ and $\sum_i k_i = n$)

Pattern Avoidance with words

- Extend classical pattern avoidance to words: 24665347 contains 1221 .
- Let $\underline{k} = (k_1, \dots, k_n)$ be a composition of n . ($k_i \in \mathbb{Z}_{\geq 0}$ and $\sum_i k_i = n$)
- **Notation:** For $\mathcal{P}$ a set of patterns, write $\text{Av}_{\underline{k}}(\mathcal{P})$ to denote the set of words with content $\underline{k}$ that avoid every pattern $\tau \in \mathcal{P}$.

Pattern Avoidance with words

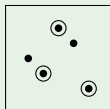
- Extend classical pattern avoidance to words: $24\overline{66}53\overline{4}7$ contains 1221 .
- Let $\underline{k} = (k_1, \dots, k_n)$ be a composition of n . ($k_i \in \mathbb{Z}_{\geq 0}$ and $\sum_i k_i = n$)
- **Notation:** For $\mathcal{P}$ a set of patterns, write $Av_{\underline{k}}(\mathcal{P})$ to denote the set of words with content $\underline{k}$ that avoid every pattern $\tau \in \mathcal{P}$.
- **Vincular patterns:** Use a dash to denote entries of τ that need not be adjacent to each other in σ . Entries without a dash between them **must** be adjacent to each other in σ .

Pattern Avoidance with words

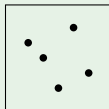
- Extend classical pattern avoidance to words: $24\underline{66}53\underline{4}7$ contains 1221 .
- Let $\underline{k} = (k_1, \dots, k_n)$ be a composition of n . ($k_i \in \mathbb{Z}_{\geq 0}$ and $\sum_i k_i = n$)
- **Notation:** For $\mathcal{P}$ a set of patterns, write $Av_{\underline{k}}(\mathcal{P})$ to denote the set of words with content $\underline{k}$ that avoid every pattern $\tau \in \mathcal{P}$.
- **Vincular patterns:** Use a dash to denote entries of τ that need not be adjacent to each other in σ . Entries without a dash between them **must** be adjacent to each other in σ .

Example

$\sigma = 3\underline{2541}\underline{1}$ contains $\tau = 23-1$:



$\sigma = 43152$ avoids $\tau = 23-1$:



Barred patterns

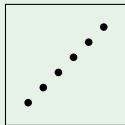
- **Barred patterns:** We now let some entries of τ be barred. For σ to contain τ , it must have a subword with the relative order of the non-barred portion of τ that is **not** a subword of all of τ .

Barred patterns

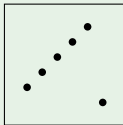
- **Barred patterns:** We now let some entries of τ be barred. For σ to contain τ , it must have a subword with the relative order of the non-barred portion of τ that is **not** a subword of all of τ .

Example

$\sigma = 123456$ contains $\tau = 23\bar{1}$:



$\sigma = 234561$ avoids $\tau = 23\bar{1}$:

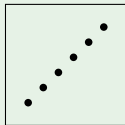


Barred patterns

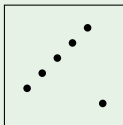
- **Barred patterns:** We now let some entries of τ be barred. For σ to contain τ , it must have a subword with the relative order of the non-barred portion of τ that is **not** a subword of all of τ .

Example

$\sigma = 123456$ contains $\tau = 23\bar{1}$:



$\sigma = 234561$ avoids $\tau = 23\bar{1}$:



- In this talk, we will combine all three of these ideas.

Geometry background

- $\overline{M}_{0,n+3}$: moduli space of stable genus 0 curves with $n + 3$ marked points.

Geometry background

- $\overline{M}_{0,n+3}$: moduli space of stable genus 0 curves with $n + 3$ marked points.
- We can compute intersection products of cohomology classes $\omega_1, \dots, \omega_n$, whose answers are nonnegative integers.

Geometry background

- $\overline{M}_{0,n+3}$: moduli space of stable genus 0 curves with $n + 3$ marked points.
- We can compute intersection products of cohomology classes $\omega_1, \dots, \omega_n$, whose answers are nonnegative integers.

Theorem (Gillespie, Griffin, Levinson, 2023)

$$\int_{\overline{M}_{0,n+3}} \omega^{\underline{k}} = \left\langle \begin{matrix} n \\ \underline{k} \end{matrix} \right\rangle = |\text{Slide}^\omega(\underline{k})| = |\text{Tour}(\underline{k})|$$

Geometry background

- $\overline{M}_{0,n+3}$: moduli space of stable genus 0 curves with $n + 3$ marked points.
- We can compute intersection products of cohomology classes $\omega_1, \dots, \omega_n$, whose answers are nonnegative integers.

Theorem (Gillespie, Griffin, Levinson, 2023)

$$\int_{\overline{M}_{0,n+3}} \omega^{\underline{k}} = \left\langle \begin{matrix} n \\ \underline{k} \end{matrix} \right\rangle = |\text{Slide}^\omega(\underline{k})| = |\text{Tour}(\underline{k})|$$

- The numbers $\left\langle \begin{matrix} n \\ \underline{k} \end{matrix} \right\rangle$ are the *asymmetric multinomial coefficients*, and $\text{Tour}(\underline{k})$ and $\text{Slide}^\omega(\underline{k})$ are two classes of trivalent trees.

Geometry background

- $\overline{M}_{0,n+3}$: moduli space of stable genus 0 curves with $n + 3$ marked points.
- We can compute intersection products of cohomology classes $\omega_1, \dots, \omega_n$, whose answers are nonnegative integers.

Theorem (Gillespie, Griffin, Levinson, 2023)

$$\int_{\overline{M}_{0,n+3}} \omega^{\underline{k}} = \left\langle \begin{matrix} n \\ \underline{k} \end{matrix} \right\rangle = |\text{Slide}^\omega(\underline{k})| = |\text{Tour}(\underline{k})|$$

- The numbers $\left\langle \begin{matrix} n \\ \underline{k} \end{matrix} \right\rangle$ are the *asymmetric multinomial coefficients*, and $\text{Tour}(\underline{k})$ and $\text{Slide}^\omega(\underline{k})$ are two classes of trivalent trees.
- Dissertation project: Found bijection $\text{Tour}(\underline{k}) \leftrightarrow \text{Slide}^\omega(\underline{k})$

Geometry background

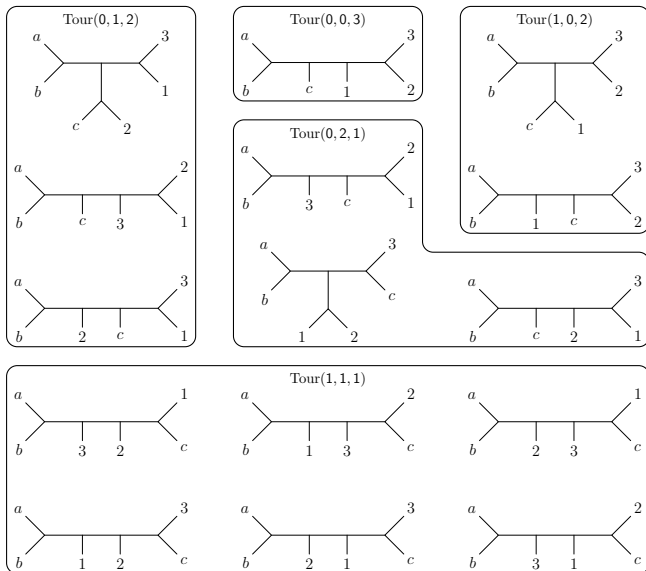
- $\overline{M}_{0,n+3}$: moduli space of stable genus 0 curves with $n + 3$ marked points.
- We can compute intersection products of cohomology classes $\omega_1, \dots, \omega_n$, whose answers are nonnegative integers.

Theorem (Gillespie, Griffin, Levinson, 2023)

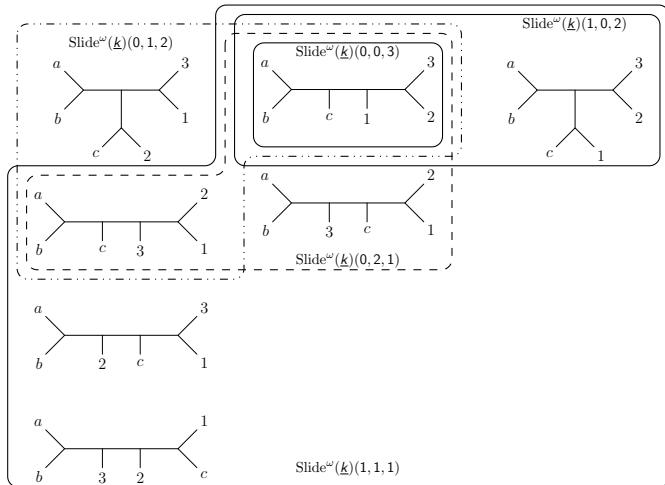
$$\int_{\overline{M}_{0,n+3}} \omega^{\underline{k}} = \left\langle \begin{matrix} n \\ \underline{k} \end{matrix} \right\rangle = |\text{Slide}^\omega(\underline{k})| = |\text{Tour}(\underline{k})|$$

- The numbers $\left\langle \begin{matrix} n \\ \underline{k} \end{matrix} \right\rangle$ are the *asymmetric multinomial coefficients*, and $\text{Tour}(\underline{k})$ and $\text{Slide}^\omega(\underline{k})$ are two classes of trivalent trees.
- Dissertation project: Found bijection $\text{Tour}(\underline{k}) \leftrightarrow \text{Slide}^\omega(\underline{k})$
- Difficulty: $\text{Tour}(\underline{k})$ and $\text{Slide}^\omega(\underline{k})$ behave very differently

Tournament sets are disjoint



Slide sets are heavily overlapping



Slide trees

- $\text{Slide}^\omega(\underline{k})$ is a set of trivalent trees, with leaves labeled by $a, b, c, 1, 2, \dots, n$, and a, b adjacent to the same vertex.

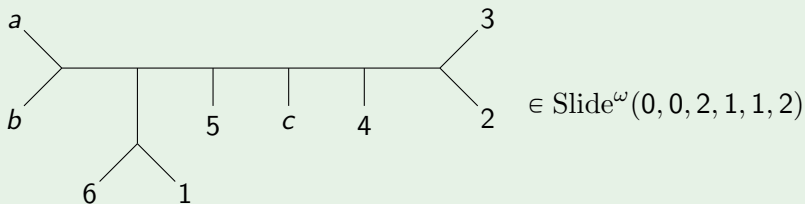
Slide trees

- $\text{Slide}^\omega(\underline{k})$ is a set of trivalent trees, with leaves labeled by $a, b, c, 1, 2, \dots, n$, and a, b adjacent to the same vertex.
- Impose an ordering on our alphabet: $a < b < c < 1 < 2 < \dots < n$

Slide trees

- $\text{Slide}^\omega(\underline{k})$ is a set of trivalent trees, with leaves labeled by $a, b, c, 1, 2, \dots, n$, and a, b adjacent to the same vertex.
- Impose an ordering on our alphabet: $a < b < c < 1 < 2 < \dots < n$

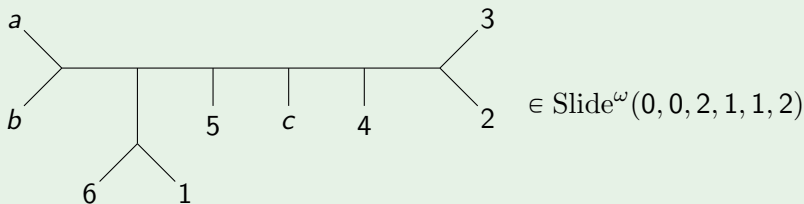
Example



Slide trees

- $\text{Slide}^\omega(\underline{k})$ is a set of trivalent trees, with leaves labeled by $a, b, c, 1, 2, \dots, n$, and a, b adjacent to the same vertex.
- Impose an ordering on our alphabet: $a < b < c < 1 < 2 < \dots < n$

Example



- Can determine if tree T is in set $\text{Slide}^\omega(\underline{k})$ by **Slide labeling algorithm**.

Slide trees

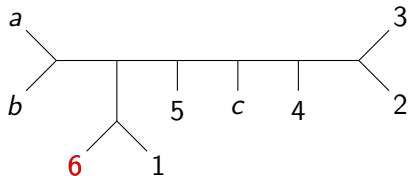
Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.

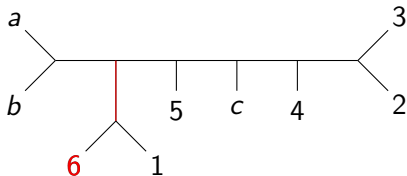


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)

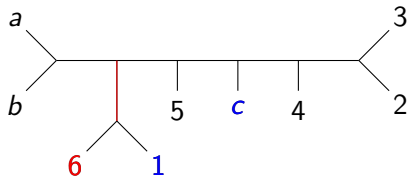


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.

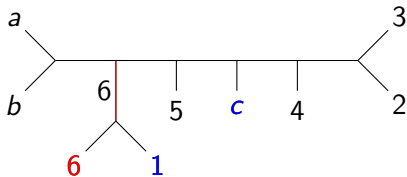


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.

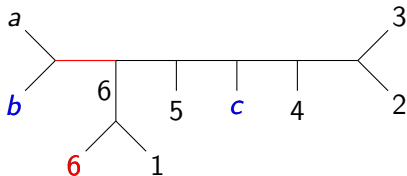


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.
- 3 **Iterate:** If ℓ has labeled k_ℓ edges, decrement ℓ . If $\ell = 0$, we're done.

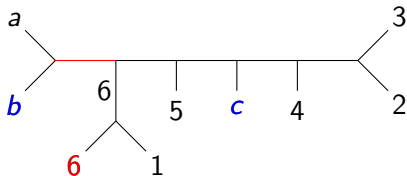


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.
- 3 **Iterate:** If ℓ has labeled k_ℓ edges, decrement ℓ . If $\ell = 0$, we're done.
- 4 **Contract labeled edges.**

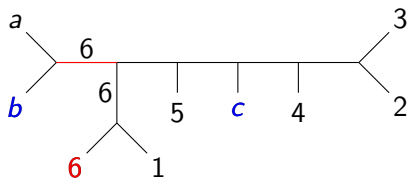


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.
- 3 **Iterate:** If ℓ has labeled k_ℓ edges, decrement ℓ . If $\ell = 0$, we're done.
- 4 **Contract labeled edges.**

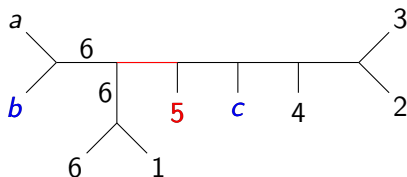


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.
- 3 **Iterate:** If ℓ has labeled k_ℓ edges, decrement ℓ . If $\ell = 0$, we're done.
- 4 **Contract labeled edges.**

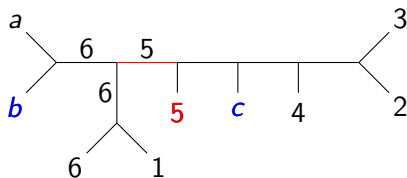


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.
- 3 **Iterate:** If ℓ has labeled k_ℓ edges, decrement ℓ . If $\ell = 0$, we're done.
- 4 **Contract labeled edges.**

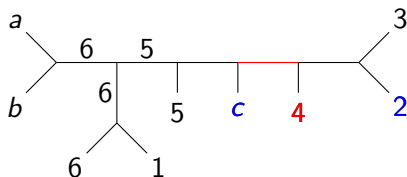


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.
- 3 **Iterate:** If ℓ has labeled k_ℓ edges, decrement ℓ . If $\ell = 0$, we're done.
- 4 **Contract labeled edges.**

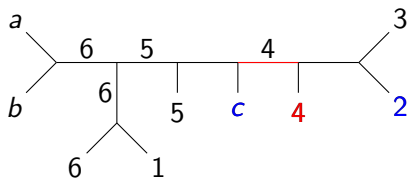


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.
- 3 **Iterate:** If ℓ has labeled k_ℓ edges, decrement ℓ . If $\ell = 0$, we're done.
- 4 **Contract labeled edges.**

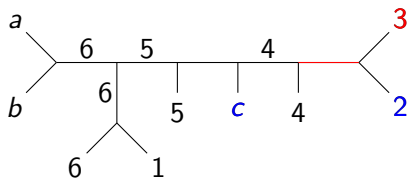


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.
- 3 **Iterate:** If ℓ has labeled k_ℓ edges, decrement ℓ . If $\ell = 0$, we're done.
- 4 **Contract labeled edges.**

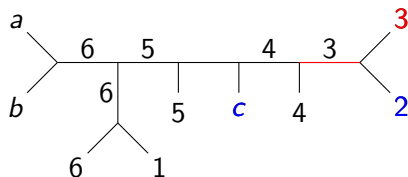


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.
- 3 **Iterate:** If ℓ has labeled k_ℓ edges, decrement ℓ . If $\ell = 0$, we're done.
- 4 **Contract labeled edges.**

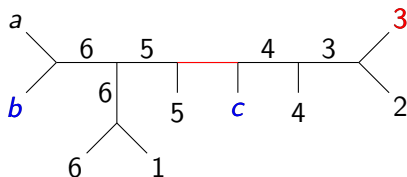


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.
- 3 **Iterate:** If ℓ has labeled k_ℓ edges, decrement ℓ . If $\ell = 0$, we're done.
- 4 **Contract labeled edges.**

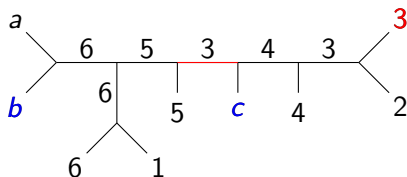


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.
- 3 **Iterate:** If ℓ has labeled k_ℓ edges, decrement ℓ . If $\ell = 0$, we're done.
- 4 **Contract labeled edges.**

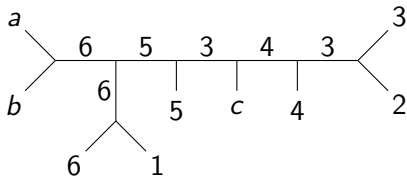


$\text{Slide}^\omega(0, 0, 2, 1, 1, 2)$

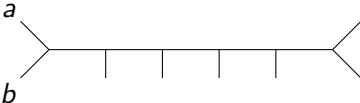
Slide trees

Definition (Slide labeling algorithm)

- 0 Start with $\ell = n$.
- 1 **Choose next edge to label:** Let e be the first unlabeled internal edge on path from leaf ℓ to a . (If none exist, then labeling fails.)
- 2 **Verify that label is valid:** Let m_1 be smallest leaf label on the same side of e as ℓ , and m_2 the smallest on the same side of e as a , excluding the branch containing a itself. If $\ell \geq m_1 \geq m_2$, then label e with ℓ . Else, terminate.
- 3 **Iterate:** If ℓ has labeled k_ℓ edges, decrement ℓ . If $\ell = 0$, we're done.
- 4 **Contract labeled edges.**


$$\in \text{Slide}^\omega(0, 0, 2, 1, 1, 2)$$

Caterpillar Trees

- A tree of the form  is called a caterpillar tree. Denote the set of caterpillars in $\text{Slide}^\omega(\underline{k})$ as $\text{Cat}^\omega(\underline{k})$.

Caterpillar Trees

- A tree of the form  is called a caterpillar tree. Denote the set of caterpillars in $\text{Slide}^\omega(\underline{k})$ as $\text{Cat}^\omega(\underline{k})$.
- Natural to read off edge labels of caterpillars from left to right to form a word. For a tree T , denote this word $w(T)$.

Caterpillar Trees

- A tree of the form  is called a caterpillar tree. Denote the set of caterpillars in $\text{Slide}^\omega(\underline{k})$ as $\text{Cat}^\omega(\underline{k})$.
- Natural to read off edge labels of caterpillars from left to right to form a word. For a tree T , denote this word $w(T)$.

Theorem (Gillespie, Griffin, Levinson, 2022)

$$w : \text{Cat}^\omega(1, 1, \dots, 1) \longleftrightarrow \text{Av}_n(23-1).$$

Caterpillar Trees

- A tree of the form  is called a caterpillar tree. Denote the set of caterpillars in $\text{Slide}^\omega(\underline{k})$ as $\text{Cat}^\omega(\underline{k})$.
- Natural to read off edge labels of caterpillars from left to right to form a word. For a tree T , denote this word $w(T)$.

Theorem (Gillespie, Griffin, Levinson, 2022)

$$w : \text{Cat}^\omega(1, 1, \dots, 1) \longleftrightarrow \text{Av}_n(23-1).$$

Example

$$w \left(\begin{array}{ccccccccccc} & a & & & & & & & & & 6 \\ & \diagdown & & 5 & & 2 & & 1 & & 4 & & 3 & & 6 \\ & \diagup & & | & & | & & | & & | & & | & & \diagup \\ b & & & 5 & & 2 & & c & & 4 & & 1 & & \diagdown \\ & & & & & & & & & & & & & 3 \end{array} \right) = 521436 \in \text{Av}_6(23-1).$$

The case $\underline{k} = (1, 1, \dots, 1)$

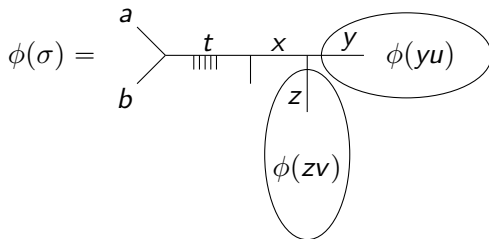
- **Goal:** Build full bijection $\phi : S_n \leftrightarrow \text{Slide}^\omega(1, 1, \dots, 1)$.

The case $\underline{k} = (1, 1, \dots, 1)$

- **Goal:** Build full bijection $\phi : S_n \leftrightarrow \text{Slide}^\omega(1, 1, \dots, 1)$.
- **Idea:** Given $\sigma \in S_n$, denote the values of the first instance of $23-1$ in σ by x, y, z (respectively). So, $\sigma = txyuzv$ for some words t, u , and v .
- From how we chose x, y, z , tx avoids $23-1$.

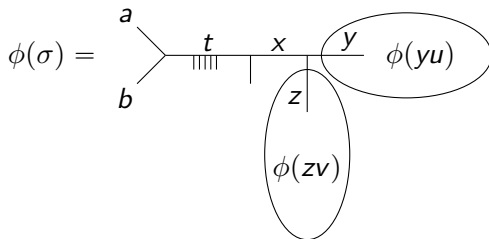
The case $\underline{k} = (1, 1, \dots, 1)$

- **Goal:** Build full bijection $\phi : S_n \leftrightarrow \text{Slide}^\omega(1, 1, \dots, 1)$.
- **Idea:** Given $\sigma \in S_n$, denote the values of the first instance of 23-1 in σ by x, y, z (respectively). So, $\sigma = txyuzv$ for some words t, u , and v .
- From how we chose x, y, z , tx avoids 23-1.
- So, can construct as follows:



The case $\underline{k} = (1, 1, \dots, 1)$

- **Goal:** Build full bijection $\phi : S_n \leftrightarrow \text{Slide}^\omega(1, 1, \dots, 1)$.
- **Idea:** Given $\sigma \in S_n$, denote the values of the first instance of 23-1 in σ by x, y, z (respectively). So, $\sigma = txyuzv$ for some words t, u , and v .
- From how we chose x, y, z , tx avoids 23-1.
- So, can construct as follows:



- To reverse, read off edges from root. When tree branches, read off entire branch off of the larger edge first.

The case $\underline{k} = (1, 1, \dots, 1)$

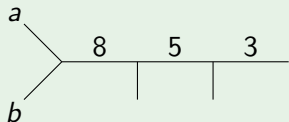
Example

Let $\sigma = 85\underline{3}7694\underline{2}1$. Then, $\phi(\sigma)$ is as follows:

The case $\underline{k} = (1, 1, \dots, 1)$

Example

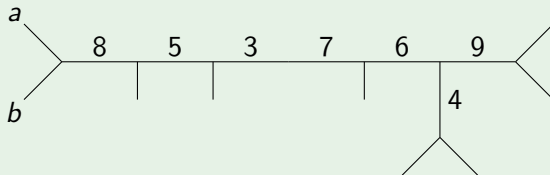
Let $\sigma = 85\underline{3}7694\underline{2}1$. Then, $\phi(\sigma)$ is as follows:



The case $\underline{k} = (1, 1, \dots, 1)$

Example

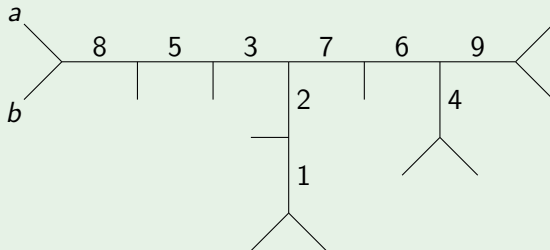
Let $\sigma = 85\underline{3}7694\underline{2}1$. Then, $\phi(\sigma)$ is as follows:



The case $\underline{k} = (1, 1, \dots, 1)$

Example

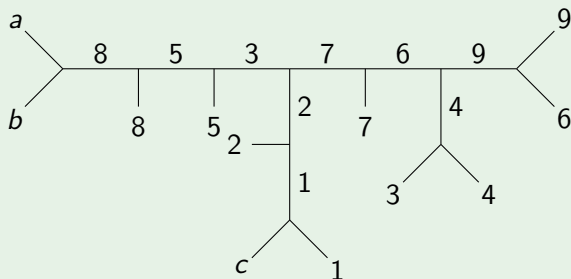
Let $\sigma = 85\underline{3}7694\underline{2}1$. Then, $\phi(\sigma)$ is as follows:



The case $\underline{k} = (1, 1, \dots, 1)$

Example

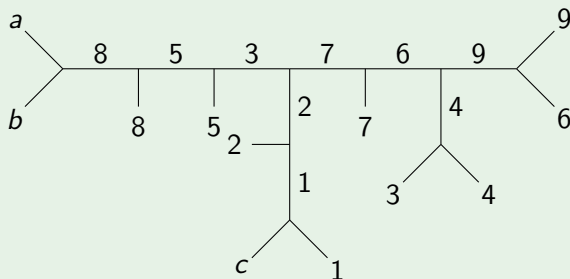
Let $\sigma = 85\underline{3}7694\underline{2}1$. Then, $\phi(\sigma)$ is as follows:



The case $\underline{k} = (1, 1, \dots, 1)$

Example

Let $\sigma = 853769421$. Then, $\phi(\sigma)$ is as follows:



Theorem (R-B.)

The map ϕ defines a bijection between S_n and $\text{Slide}^\omega(1, 1, \dots, 1)$.

Right-justified case

Definition

A composition $\underline{k}$ is *right-justified* if all entries that are 0 occur before all non-zero entries.

Right-justified case

Definition

A composition $\underline{k}$ is *right-justified* if all entries that are 0 occur before all non-zero entries.

Theorem (R-B.)

If $\underline{k}$ is right-justified, then $w : \text{Cat}^\omega(\underline{k}) \leftrightarrow \text{Av}_{\underline{k}}(2-1-2, 23-\bar{2}-1)$.

Right-justified case

Definition

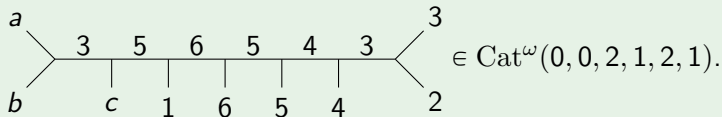
A composition $\underline{k}$ is *right-justified* if all entries that are 0 occur before all non-zero entries.

Theorem (R-B.)

If $\underline{k}$ is right-justified, then $w : \text{Cat}^\omega(\underline{k}) \leftrightarrow \text{Av}_{\underline{k}}(2-1-2, 23-\bar{2}-1)$.

Example

The word 356543 avoids $2-1-2$ and $23-\bar{2}-1$, and



General caterpillars

- Have full characterization for both $\text{Cat}^\omega(\underline{k})$ and $\text{Cat}^\psi(\underline{k})$

General caterpillars

- Have full characterization for both $\text{Cat}^\omega(\underline{k})$ and $\text{Cat}^\psi(\underline{k})$
- Not purely in terms of pattern avoidance

General caterpillars

- Have full characterization for both $\text{Cat}^\omega(\underline{k})$ and $\text{Cat}^\psi(\underline{k})$
- Not purely in terms of pattern avoidance

Definition

For a composition $\underline{k}$, define $z(i) := \#\{j > i \mid k_j = 0\}$.

General caterpillars

- Have full characterization for both $\text{Cat}^\omega(\underline{k})$ and $\text{Cat}^\psi(\underline{k})$
- Not purely in terms of pattern avoidance

Definition

For a composition $\underline{k}$, define $z(i) := \#\{j > i \mid k_j = 0\}$.

Definition

For a word w with content $\underline{k}$, let $k(i, j)$ denote the number of j 's in the subword of w right of the rightmost i . Define the number of repeats larger than i right of the rightmost i by

$$\text{BigRep}_w(i) = \sum_{j=i+1}^n \max((k(i, j) - 1), 0).$$

General caterpillars

Theorem (R.-B.)

Let $\underline{k}$ be a composition of n . Then, $T \in \text{Cat}^\omega(\underline{k})$ if and only if $w(T) \in \text{Av}_{\underline{k}}(2-1-2, 23-\bar{2}-1)$ and

$$\text{BigRep}_w(i) \geq z(i)$$

for all i where $k_i > 0$.

General caterpillars

Theorem (R.-B.)

Let $\underline{k}$ be a composition of n . Then, $T \in \text{Cat}^\omega(\underline{k})$ if and only if $w(T) \in \text{Av}_{\underline{k}}(2-1-2, 23-\bar{2}-1)$ and

$$\text{BigRep}_w(i) \geq z(i)$$

for all i where $k_i > 0$.

General caterpillars

Theorem (R.-B.)

Let $\underline{k}$ be a composition of n . Then, $T \in \text{Cat}^\omega(\underline{k})$ if and only if $w(T) \in \text{Av}_{\underline{k}}(2-1-2, 23-\bar{2}-1)$ and

$$\text{BigRep}_w(i) \geq z(i)$$

for all i where $k_i > 0$.

Definition

Let $\text{TotalRep}_w(i)$ be the *total* number of repeats right of the rightmost i , and $\ell_w(i)$ the number of i 's in the rightmost consecutive sequence of i 's.

General caterpillars

Theorem (R.-B.)

Let $\underline{k}$ be a composition of n . Then, $T \in \text{Cat}^\omega(\underline{k})$ if and only if $w(T) \in \text{Av}_{\underline{k}}(2-1-2, 23-\bar{2}-1)$ and

$$\text{BigRep}_w(i) \geq z(i)$$

for all i where $k_i > 0$.

Definition

Let $\text{TotalRep}_w(i)$ be the *total* number of repeats right of the rightmost i , and $\ell_w(i)$ the number of i 's in the rightmost consecutive sequence of i 's.

Theorem (R.-B.)

Similarly, $T \in \text{Cat}^\psi(\underline{k})$ if and only if $w(T) \in \text{Av}_{\underline{k}}(2-1-2, 23-\bar{2}-1)$ and $\text{TotalRep}_w(i) + \ell_w(i) \geq z(i)$ for all i where $k_i > 0$.

Other results

- Built full bijection by showing $\text{Slide}^\omega(\underline{k})$ satisfies same recurrence relation as $\text{Tour}(\underline{k})$

Other results

- Built full bijection by showing $\text{Slide}^\omega(\underline{k})$ satisfies same recurrence relation as $\text{Tour}(\underline{k})$
- Makes use of RSK-like insertion algorithm on trees

Other results

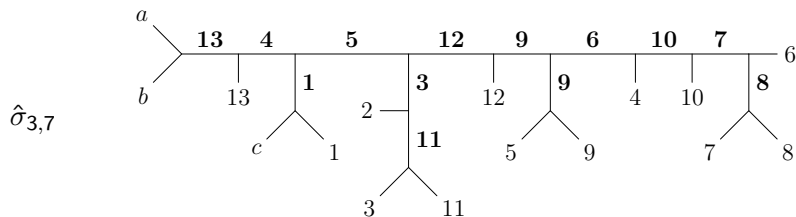
- Built full bijection by showing $\text{Slide}^\omega(\underline{k})$ satisfies same recurrence relation as $\text{Tour}(\underline{k})$
- Makes use of RSK-like insertion algorithm on trees
- Currently thinking about similar geometric spaces called *multicolored spaces*

Thank You

THANK YOU!

Insertion Algorithms

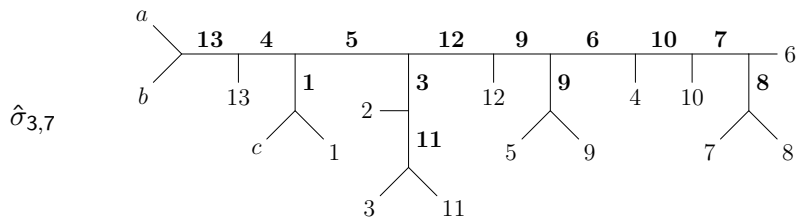
Definition (Construction of $\hat{\sigma}_{i,j}(T)$)



Insertion Algorithms

Definition (Construction of $\hat{\sigma}_{i,j}(T)$)

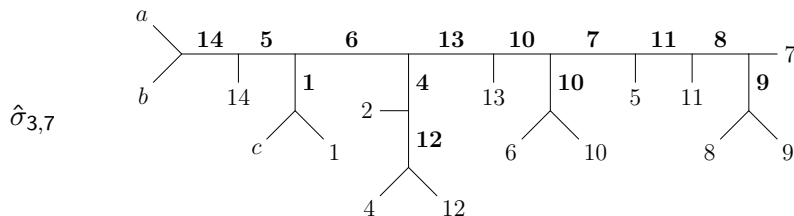
- 1 Increment all labels $\geq i$



Insertion Algorithms

Definition (Construction of $\hat{\sigma}_{i,j}(T)$)

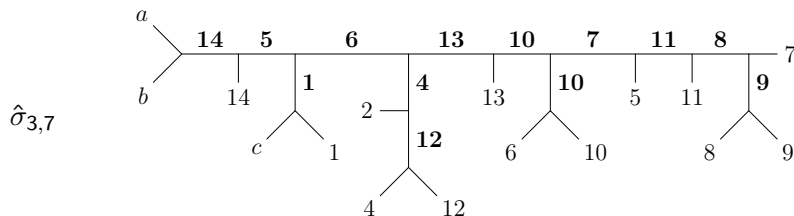
- 1 Increment all labels $\geq i$



Insertion Algorithms

Definition (Construction of $\hat{\sigma}_{i,j}(T)$)

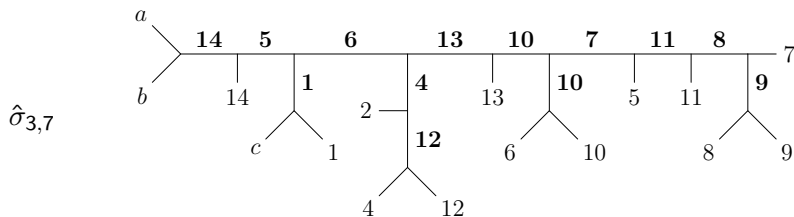
- ① Increment all labels $\geq i$
- ② On path from a to j , consider maximal length decreasing sequences of edges



Insertion Algorithms

Definition (Construction of $\hat{\sigma}_{i,j}(T)$)

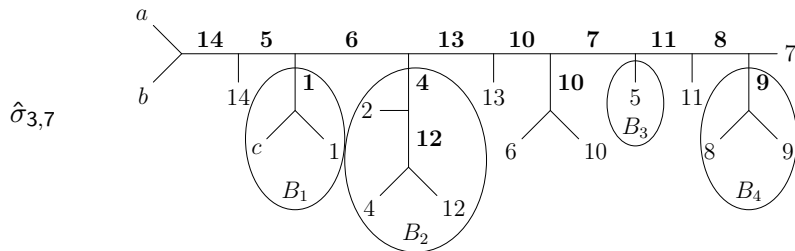
- ① Increment all labels $\geq i$
- ② On path from a to j , consider maximal length decreasing sequences of edges
- ③ Let $B_1, B_2, \dots, B_l$ be branches of T between these subsequences



Insertion Algorithms

Definition (Construction of $\hat{\sigma}_{i,j}(T)$)

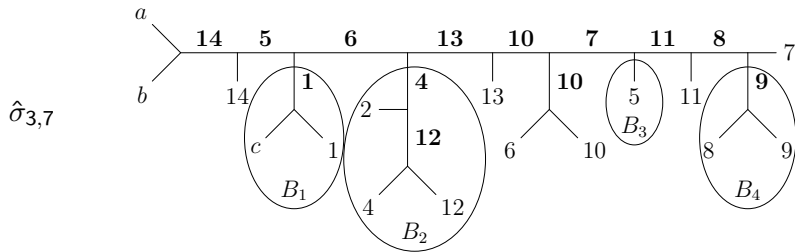
- 1 Increment all labels $\geq i$
- 2 On path from a to j , consider maximal length decreasing sequences of edges
- 3 Let $B_1, B_2, \dots, B_l$ be branches of T between these subsequences



Insertion Algorithms

Definition (Construction of $\hat{\sigma}_{i,j}(T)$)

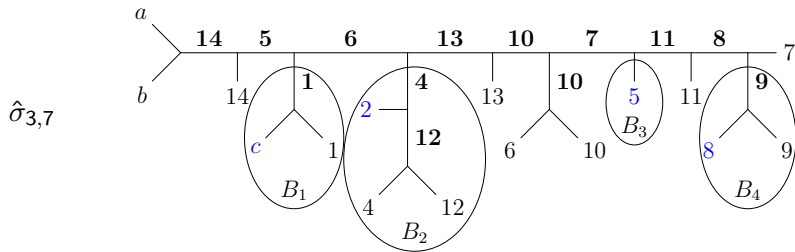
- ➊ Increment all labels $\geq i$
- ➋ On path from a to j , consider maximal length decreasing sequences of edges
- ➌ Let $B_1, B_2, \dots, B_l$ be branches of T between these subsequences
- ➍ Define m_r as the minimal leaf of B_r



Insertion Algorithms

Definition (Construction of $\hat{\sigma}_{i,j}(T)$)

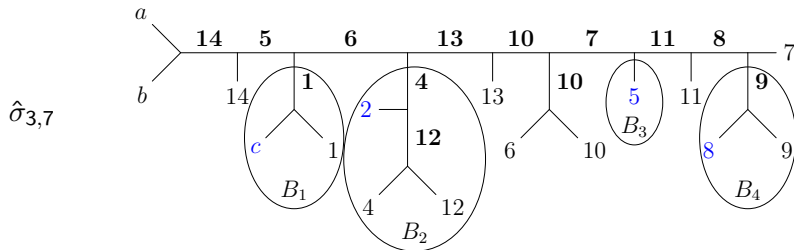
- ➊ Increment all labels $\geq i$
- ➋ On path from a to j , consider maximal length decreasing sequences of edges
- ➌ Let $B_1, B_2, \dots, B_l$ be branches of T between these subsequences
- ➍ Define m_r as the minimal leaf of B_r



Insertion Algorithms

Definition (Construction of $\hat{\sigma}_{i,j}(T)$)

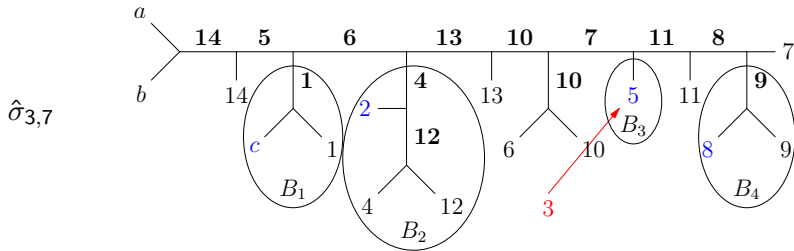
- ➊ Increment all labels $\geq i$
- ➋ On path from a to j , consider maximal length decreasing sequences of edges
- ➌ Let $B_1, B_2, \dots, B_l$ be branches of T between these subsequences
- ➍ Define m_r as the minimal leaf of B_r
- ➎ Here, $m_1 < \dots < m_{d-1} < i < m_d < \dots < j < m_l$: replace m_d by i , m_{d+1} by m_d , $\dots$, and replace leaf j by an edge $\mathbf{j}$ with leaves j and m_{l-1} .



Insertion Algorithms

Definition (Construction of $\hat{\sigma}_{i,j}(T)$)

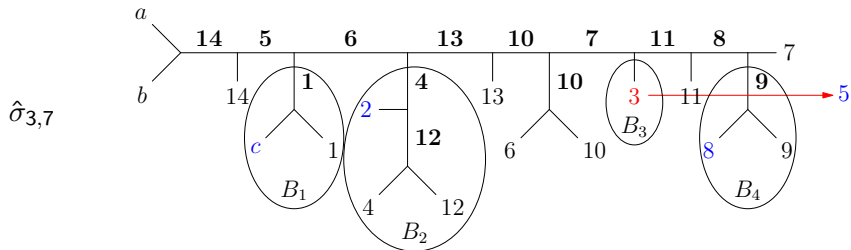
- ➊ Increment all labels $\geq i$
- ➋ On path from a to j , consider maximal length decreasing sequences of edges
- ➌ Let $B_1, B_2, \dots, B_l$ be branches of T between these subsequences
- ➍ Define m_r as the minimal leaf of B_r
- ➎ Here, $m_1 < \dots < m_{d-1} < i < m_d < \dots < j < m_l$: replace m_d by i , m_{d+1} by m_d , $\dots$, and replace leaf j by an edge $\mathbf{j}$ with leaves j and m_{l-1} .



Insertion Algorithms

Definition (Construction of $\hat{\sigma}_{i,j}(T)$)

- 1 Increment all labels $\geq i$
- 2 On path from a to j , consider maximal length decreasing sequences of edges
- 3 Let $B_1, B_2, \dots, B_l$ be branches of T between these subsequences
- 4 Define m_r as the minimal leaf of B_r
- 5 Here, $m_1 < \dots < m_{d-1} < i < m_d < \dots < j < m_l$: replace m_d by i , m_{d+1} by m_d , $\dots$, and replace leaf j by an edge $\mathbf{j}$ with leaves j and m_{l-1} .



Insertion Algorithms

Definition (Construction of $\hat{\sigma}_{i,j}(T)$)

- ① Increment all labels $\geq i$
- ② On path from a to j , consider maximal length decreasing sequences of edges
- ③ Let $B_1, B_2, \dots, B_l$ be branches of T between these subsequences
- ④ Define m_r as the minimal leaf of B_r
- ⑤ Here, $m_1 < \dots < m_{d-1} < i < m_d < \dots < j < m_l$: replace m_d by i , m_{d+1} by m_d , $\dots$, and replace leaf j by an edge $\mathbf{j}$ with leaves j and m_{l-1} .

