On surface cluster algebras: Snake graph calculus and dreaded torus

İlke Çanakçı[1]

[1]Department of Mathematics
University of Leicester

joint work with Ralf Schiffler

Geometry Seminar, University of Bath
March 25, 2014

Outline of Topics

Overview

- Cluster algebras were introduced by **Fomin and Zelevinsky** [FZ1] with the desire of creating an algebraic framework for the study of (dual) canonical bases in Lie theory.
- Cluster algebras are defined by **generators** and **relations**, and the set of generators is constructed recursively from some **initial data** $(\mathbf{x}, Q)$ called **seed**, where $\mathbf{x} = (x_1, \cdots, x_n)$ and Q is a quiver.
- Cluster algebras form a class of combinatorially defined commutative algebras, and the set of generators of a cluster algebra, **cluster variables**, is obtained by an iterative process called **seed mutation.**
- The cluster variables are **rational functions in several variables** $x_1, x_2, \cdots, x_n$ by construction.
- However, by a well-known result in [FZ1] they can be expressed as **Laurent polynomials** in $x_1, x_2, \cdots, x_n$ with integer coefficients.

Overview

- Cluster algebras were introduced by **Fomin and Zelevinsky** [FZ1] with the desire of creating an algebraic framework for the study of (dual) canonical bases in Lie theory.
- Cluster algebras are defined by **generators** and **relations**, and the set of generators is constructed recursively from some **initial data** $(\mathbf{x}, Q)$ called **seed**, where $\mathbf{x} = (x_1, \cdots, x_n)$ and Q is a quiver.
- Cluster algebras form a class of combinatorially defined commutative algebras, and the set of generators of a cluster algebra, **cluster variables**, is obtained by an iterative process called **seed mutation.**
- The cluster variables are **rational functions in several variables** $x_1, x_2, \cdots, x_n$ by construction.
- However, by a well-known result in [FZ1] they can be expressed as **Laurent polynomials** in $x_1, x_2, \cdots, x_n$ with integer coefficients.

Overview

- Cluster algebras were introduced by **Fomin and Zelevinsky** [FZ1] with the desire of creating an algebraic framework for the study of (dual) canonical bases in Lie theory.
- Cluster algebras are defined by **generators** and **relations**, and the set of generators is constructed recursively from some **initial data** $(\mathbf{x}, Q)$ called **seed**, where $\mathbf{x} = (x_1, \cdots, x_n)$ and Q is a quiver.
- Cluster algebras form a class of combinatorially defined commutative algebras, and the set of generators of a cluster algebra, **cluster variables**, is obtained by an iterative process called **seed mutation.**
- The cluster variables are **rational functions in several variables** $x_1, x_2, \cdots, x_n$ by construction.
- However, by a well-known result in [FZ1] they can be expressed as **Laurent polynomials** in $x_1, x_2, \cdots, x_n$ with integer coefficients.

Overview

- Cluster algebras were introduced by **Fomin and Zelevinsky** [FZ1] with the desire of creating an algebraic framework for the study of (dual) canonical bases in Lie theory.
- Cluster algebras are defined by **generators** and **relations**, and the set of generators is constructed recursively from some **initial data** $(\mathbf{x}, Q)$ called **seed**, where $\mathbf{x} = (x_1, \cdots, x_n)$ and Q is a quiver.
- Cluster algebras form a class of combinatorially defined commutative algebras, and the set of generators of a cluster algebra, **cluster variables**, is obtained by an iterative process called **seed mutation.**
- The cluster variables are **rational functions in several variables** $x_1, x_2, \cdots, x_n$ by construction.
- However, by a well-known result in [FZ1] they can be expressed as **Laurent polynomials** in $x_1, x_2, \cdots, x_n$ with integer coefficients.

Overview

- Cluster algebras were introduced by **Fomin and Zelevinsky** [FZ1] with the desire of creating an algebraic framework for the study of (dual) canonical bases in Lie theory.
- Cluster algebras are defined by **generators** and **relations**, and the set of generators is constructed recursively from some **initial data** $(\mathbf{x}, Q)$ called **seed**, where $\mathbf{x} = (x_1, \cdots, x_n)$ and Q is a quiver.
- Cluster algebras form a class of combinatorially defined commutative algebras, and the set of generators of a cluster algebra, **cluster variables**, is obtained by an iterative process called **seed mutation.**
- The cluster variables are **rational functions in several variables** $x_1, x_2, \cdots, x_n$ by construction.
- However, by a well-known result in [FZ1] they can be expressed as **Laurent polynomials** in $x_1, x_2, \cdots, x_n$ with integer coefficients.

Overview

- Cluster algebras were introduced by **Fomin and Zelevinsky** [FZ1] with the desire of creating an algebraic framework for the study of (dual) canonical bases in Lie theory.
- Cluster algebras are defined by **generators** and **relations**, and the set of generators is constructed recursively from some **initial data** $(\mathbf{x}, Q)$ called **seed**, where $\mathbf{x} = (x_1, \cdots, x_n)$ and Q is a quiver.
- Cluster algebras form a class of combinatorially defined commutative algebras, and the set of generators of a cluster algebra, **cluster variables**, is obtained by an iterative process called **seed mutation.**
- The cluster variables are **rational functions in several variables** $x_1, x_2, \cdots, x_n$ by construction.
- However, by a well-known result in [FZ1] they can be expressed as **Laurent polynomials** in $x_1, x_2, \cdots, x_n$ with integer coefficients.

Overview

- **Cluster algebras from surfaces**, introduced in [FST], have a geometric interpretation in surfaces.
- A **surface cluster algebra** $\mathcal{A}$ is associated to a surface S with boundary that has finitely many marked points.
- **Cluster variables** are **in bijection with** certain curves [FST], called **arcs**. Two crossing arcs satisfy the skein relations [MW]
- The authors in [MSW] associate a connected graph, called the **snake graph** to each arc in the surface to obtain a direct formula, the **expansion formula**, for cluster variables of surface cluster algebras.

$$x_\gamma = \frac{1}{\text{cross}\,(\gamma, T)} \sum_{P \vdash G_\gamma} x(P)y(P)$$

Overview

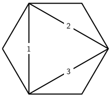
- **Cluster algebras from surfaces**, introduced in [FST], have a geometric interpretation in surfaces.
- A **surface cluster algebra** $\mathcal{A}$ is associated to a surface S with boundary that has finitely many marked points.

- **Cluster variables** are **in bijection with** certain curves [FST], called **arcs**. Two crossing arcs satisfy the skein relations [MSW]
- The authors in [MSW] associate a connected graph, called the **snake graph** to each arc in the surface to obtain a direct formula, the **expansion formula**, for cluster variables of surface cluster algebras.

$$x_\gamma = \frac{1}{\text{cross}\,(\gamma, T)} \sum_{P \vdash G_\gamma} x(P)y(P)$$

Overview

- **Cluster algebras from surfaces**, introduced in [FST], have a geometric interpretation in surfaces.
- A **surface cluster algebra** $\mathcal{A}$ is associated to a surface S with boundary that has finitely many marked points.

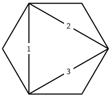


- **Cluster variables** are **in bijection with** certain curves [FST], called **arcs**. Two crossing arcs satisfy the **skein relations** [MSW].
- The authors in [MSW] associate a connected graph, called the **snake graph** to each arc in the surface to obtain a direct formula, the **expansion formula**, for cluster variables of surface cluster algebras.

$$x_\gamma = \frac{1}{\mathrm{cross}\,(\gamma, T)} \sum_{P \vdash G_\gamma} x(P)y(P)$$

Overview

- **Cluster algebras from surfaces**, introduced in [FST], have a geometric interpretation in surfaces.
- A **surface cluster algebra** $\mathcal{A}$ is associated to a surface S with boundary that has finitely many marked points.

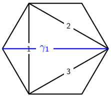


- **Cluster variables** are **in bijection with** certain curves [FST], called **arcs**. Two crossing arcs satisfy the **skein relations**, [MW].
- The authors in [MSW] associate a connected graph, called the **snake graph** to each arc in the surface to obtain a direct formula, the **expansion formula**, for cluster variables of surface cluster algebras.

$$x_\gamma = \frac{1}{\text{cross}(\gamma, T)} \sum_{P \vdash G_\gamma} x(P)y(P)$$

Overview

- **Cluster algebras from surfaces**, introduced in [FST], have a geometric interpretation in surfaces.
- A **surface cluster algebra** $\mathcal{A}$ is associated to a surface S with boundary that has finitely many marked points.

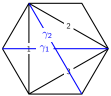


- **Cluster variables** are **in bijection with** certain curves [FST], called **arcs**. Two crossing arcs satisfy the **skein relations**, [MW].
- The authors in [MSW] associate a connected graph, called the **snake graph** to each arc in the surface to obtain a direct formula, the **expansion formula**, for cluster variables of surface cluster algebras.

$$x_\gamma = \frac{1}{\text{cross}(\gamma, T)} \sum_{P \vdash G_\gamma} x(P)y(P)$$

Overview

- **Cluster algebras from surfaces**, introduced in [FST], have a geometric interpretation in surfaces.
- A **surface cluster algebra** $\mathcal{A}$ is associated to a surface S with boundary that has finitely many marked points.

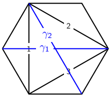


- **Cluster variables** are **in bijection with** certain curves [FST], called **arcs**. Two crossing arcs satisfy the **skein relations**, [MW].
- The authors in [MSW] associate a connected graph, called the **snake graph** to each arc in the surface to obtain a direct formula, the **expansion formula**, for cluster variables of surface cluster algebras.

$$x_\gamma = \frac{1}{\text{cross}(\gamma, T)} \sum_{P \vdash G_\gamma} x(P)y(P)$$

Overview

- **Cluster algebras from surfaces**, introduced in [FST], have a geometric interpretation in surfaces.
- A **surface cluster algebra** $\mathcal{A}$ is associated to a surface S with boundary that has finitely many marked points.

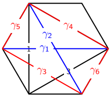


- **Cluster variables** are **in bijection with** certain curves [FST], called **arcs**. Two crossing arcs satisfy the **skein relations**, [MW].
- The authors in [MSW] associate a connected graph, called the **snake graph** to each arc in the surface to obtain a direct formula, the **expansion formula**, for cluster variables of surface cluster algebras.

$$x_\gamma = \frac{1}{\text{cross}(\gamma, T)} \sum_{P \vdash G_\gamma} x(P)y(P)$$

Overview

- **Cluster algebras from surfaces**, introduced in [FST], have a geometric interpretation in surfaces.
- A **surface cluster algebra** $\mathcal{A}$ is associated to a surface S with boundary that has finitely many marked points.

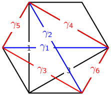


- **Cluster variables** are **in bijection with** certain curves [FST], called **arcs**. Two crossing arcs satisfy the **skein relations**, [MW].
- The authors in [MSW] associate a connected graph, called the **snake graph** to each arc in the surface to obtain a direct formula, the **expansion formula**, for cluster variables of surface cluster algebras.

$$x_\gamma = \frac{1}{\text{cross}\,(\gamma, T)} \sum_{P \vdash G_\gamma} x(P)y(P)$$

Overview

- **Cluster algebras from surfaces**, introduced in [FST], have a geometric interpretation in surfaces.
- A **surface cluster algebra** $\mathcal{A}$ is associated to a surface S with boundary that has finitely many marked points.



$$x_{\gamma_1} x_{\gamma_2} = * x_{\gamma_3} x_{\gamma_4} + * x_{\gamma_5} x_{\gamma_6}$$

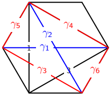
Skein relation ([MW])

- **Cluster variables** are **in bijection with** certain curves [FST], called **arcs**. Two crossing arcs satisfy the **skein relations**, [MW].
- The authors in [MSW] associate a connected graph, called the **snake graph** to each arc in the surface to obtain a direct formula, the **expansion formula**, for cluster variables of surface cluster algebras.

$$x_\gamma = \frac{1}{\text{cross}\,(\gamma, T)} \sum_{P \vdash G_\gamma} x(P) y(P)$$

Overview

- **Cluster algebras from surfaces**, introduced in [FST], have a geometric interpretation in surfaces.
- A **surface cluster algebra** $\mathcal{A}$ is associated to a surface S with boundary that has finitely many marked points.



$$x_{\gamma_1}x_{\gamma_2} = *x_{\gamma_3}x_{\gamma_4} + *x_{\gamma_5}x_{\gamma_6}$$

Skein relation ([MW])

- **Cluster variables** are **in bijection with** certain curves [FST], called **arcs**. Two crossing arcs satisfy the **skein relations**, [MW].
- The authors in [MSW] associate a connected graph, called the **snake graph** to each arc in the surface to obtain a direct formula, the **expansion formula**, for cluster variables of surface cluster algebras.

$$x_\gamma = \frac{1}{\text{cross}\,(\gamma, T)} \sum_{P \vdash G_\gamma} x(P)y(P)$$

Motivation

Let $\mathcal{A}(S, M)$ cluster algebra associated to a surface (S, M).

We have the following situation:

Question

"How much can we recover from snake graphs themselves?"

In particular,

- When do the two arcs corresponding to two snake graphs cross?
- What are the snake graphs corresponding to the skein relations?

Motivation

Let $\mathcal{A}(S, M)$ cluster algebra associated to a surface (S, M).

We have the following situation:

$$\text{cluster variable} \quad \xleftrightarrow{[\text{FST}]} \quad \text{arc}$$

Question

"How much can we recover from snake graphs themselves?"

In particular,

- When do the two arcs corresponding to two snake graphs cross?
- What are the snake graphs corresponding to the skein relations?

Motivation

Let $\mathcal{A}(S, M)$ cluster algebra associated to a surface (S, M).

We have the following situation:

$$\boxed{\text{cluster variable} \quad \overset{\text{[FST]}}{\longleftrightarrow} \quad \text{arc} \quad \overset{\text{[MSW]}}{\longrightarrow} \quad \textbf{snake graph}}$$

Question

"How much can we recover from snake graphs themselves?"

In particular,

- When do the two arcs corresponding to two snake graphs cross?
- What are the snake graphs corresponding to the skein relations?

Motivation

Let $\mathcal{A}(S, M)$ cluster algebra associated to a surface (S, M).

We have the following situation:

$$\boxed{\text{cluster variable} \quad \overset{[\mathrm{FST}]}{\longleftrightarrow} \quad \text{arc} \quad \overset{[\mathrm{MSW}]}{\longrightarrow} \quad \textbf{snake graph}}$$

Question

"How much can we recover from snake graphs themselves?"

In particular,

- When do the two arcs corresponding to two snake graphs cross?
- What are the snake graphs corresponding to the skein relations?

Motivation

Let $\mathcal{A}(S, M)$ cluster algebra associated to a surface (S, M).

We have the following situation:

$$\boxed{\text{cluster variable} \quad \xleftrightarrow{\text{[FST]}} \quad \text{arc} \quad \xrightarrow{\text{[MSW]}} \quad \textbf{snake graph}}$$

Question

“How much can we recover from snake graphs themselves?”

In particular,

- When do the two arcs corresponding to two snake graphs cross?
- What are the snake graphs corresponding to the skein relations?

Motivation

Let $\mathcal{A}(S, M)$ cluster algebra associated to a surface (S, M).

We have the following situation:

$$\boxed{\text{cluster variable} \quad \overset{\text{[FST]}}{\longleftrightarrow} \quad \text{arc} \quad \overset{\text{[MSW]}}{\longrightarrow} \quad \textbf{snake graph}}$$

Question

"How much can we recover from snake graphs themselves?"

In particular,

- When do the two arcs corresponding to two snake graphs cross?
- What are the snake graphs corresponding to the skein relations?

Motivation

Let $\mathcal{A}(S, M)$ cluster algebra associated to a surface (S, M).

We have the following situation:

cluster variable	$\overset{[FST]}{\longleftrightarrow}$	arc	$\overset{[MSW]}{\longrightarrow}$	**snake graph**

Question

"How much can we recover from snake graphs themselves?"

In particular,

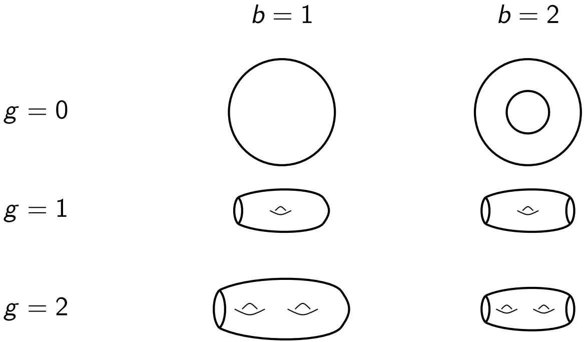
- When do the two arcs corresponding to two snake graphs cross?
- What are the snake graphs corresponding to the skein relations?

Surface Cluster Algebras

- Let S be a connected oriented 2-dimensional Riemann surface with nonempty boundary, and let M be a nonempty finite subset of the boundary of S, such that each boundary component of S contains at least one point of M. The elements of M are called *marked points*. The pair (S, M) is called a **bordered surface with marked points**.

Surface Cluster Algebras

- Let S be a connected oriented 2-dimensional Riemann surface with nonempty boundary, and let M be a nonempty finite subset of the boundary of S, such that each boundary component of S contains at least one point of M. The elements of M are called *marked points*. The pair (S, M) is called a **bordered surface with marked points**.



Surface Cluster Algebras

Definition

An **arc** γ in (S, M) is a curve in S, considered up to isotopy, such that:

- the endpoints of γ are in M;
- γ does not cross itself;
- except for the endpoints, γ is disjoint from the boundary of S; and
- γ does not cut out a monogon or a bigon.

Remark

Curves that connect two marked points and lie entirely on the boundary of S without passing through a third marked point are *boundary segments*. Note that **boundary segments are not arcs.**

Surface Cluster Algebras

Definition

An **arc** γ in (S, M) is a curve in S, considered up to isotopy, such that:

- the endpoints of γ are in M;
- γ does not cross itself;
- except for the endpoints, γ is disjoint from the boundary of S; and
- γ does not cut out a monogon or a bigon.

Remark

Curves that connect two marked points and lie entirely on the boundary of S without passing through a third marked point are *boundary segments*. Note that **boundary segments are not arcs.**

Surface Cluster Algebras

Definition

For any two arcs γ, γ' in S, let $e(\gamma, \gamma')$ be the **minimal number of crossings** of arcs α and α', where α and α' range over all arcs isotopic to γ and γ', respectively. We say that arcs γ and γ' are **compatible** if $e(\gamma, \gamma') = 0$.

Definition

A **triangulation** is a maximal collection of pairwise compatible arcs (together with all boundary segments).

Surface Cluster Algebras

Definition

For any two arcs γ, γ' in S, let $e(\gamma, \gamma')$ be the **minimal number of crossings** of arcs α and α', where α and α' range over all arcs isotopic to γ and γ', respectively. We say that arcs γ and γ' are **compatible** if $e(\gamma, \gamma') = 0$.

Definition

A **triangulation** is a maximal collection of pairwise compatible arcs (together with all boundary segments).

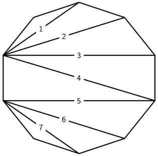
Surface Cluster Algebras

Definition

For any two arcs γ, γ' in S, let $e(\gamma, \gamma')$ be the **minimal number of crossings** of arcs α and α', where α and α' range over all arcs isotopic to γ and γ', respectively. We say that arcs γ and γ' are **compatible** if $e(\gamma, \gamma') = 0$.

Definition

A **triangulation** is a maximal collection of pairwise compatible arcs (together with all boundary segments).



Surface Cluster Algebras

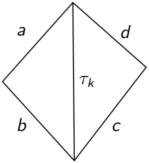
Definition

Triangulations are connected to each other by sequences of **flips**. Each flip replaces a single arc γ in a triangulation T by a (unique) arc $\gamma' \neq \gamma$ that, together with the remaining arcs in T, forms a new triangulation.

Surface Cluster Algebras

Definition

Triangulations are connected to each other by sequences of **flips**. Each flip replaces a single arc γ in a triangulation T by a (unique) arc $\gamma' \neq \gamma$ that, together with the remaining arcs in T, forms a new triangulation.



Surface Cluster Algebras

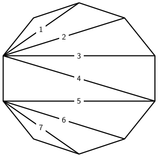
Definition

Triangulations are connected to each other by sequences of **flips**. Each flip replaces a single arc γ in a triangulation T by a (unique) arc $\gamma' \neq \gamma$ that, together with the remaining arcs in T, forms a new triangulation.

Surface Cluster Algebras

Definition

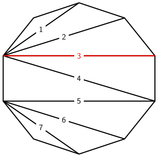
Triangulations are connected to each other by sequences of **flips**. Each flip replaces a single arc γ in a triangulation T by a (unique) arc $\gamma' \neq \gamma$ that, together with the remaining arcs in T, forms a new triangulation.



Surface Cluster Algebras

Definition

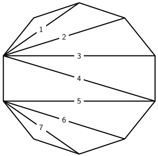
Triangulations are connected to each other by sequences of **flips**. Each flip replaces a single arc γ in a triangulation T by a (unique) arc $\gamma' \neq \gamma$ that, together with the remaining arcs in T, forms a new triangulation.

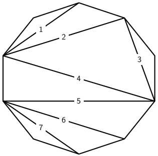


Surface Cluster Algebras

Definition

Triangulations are connected to each other by sequences of **flips**. Each flip replaces a single arc γ in a triangulation T by a (unique) arc $\gamma' \neq \gamma$ that, together with the remaining arcs in T, forms a new triangulation.

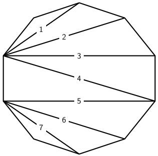


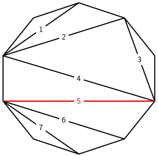


Surface Cluster Algebras

Definition

Triangulations are connected to each other by sequences of **flips**. Each flip replaces a single arc γ in a triangulation T by a (unique) arc $\gamma' \neq \gamma$ that, together with the remaining arcs in T, forms a new triangulation.

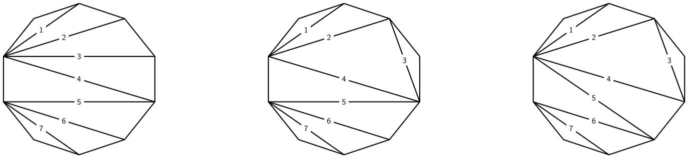




Surface Cluster Algebras

Definition

Triangulations are connected to each other by sequences of **flips**. Each flip replaces a single arc γ in a triangulation T by a (unique) arc $\gamma' \neq \gamma$ that, together with the remaining arcs in T, forms a new triangulation.



Surface Cluster Algebras

Theorem (FST,FT)

For cluster algebras from surfaces

- *there are bijections*

$$\begin{array}{ccc} \{\ \textit{arcs}\ \} & \longrightarrow & \{\ \textit{cluster variables}\ \} \\ \gamma & \mapsto & x_\gamma \end{array}$$

$$\begin{array}{ccc} \{\ \textit{triangulations}\ \} & \longrightarrow & \{\ \textit{clusters}\ \} \\ T = \{\tau_1, \cdots, \tau_n\} & \mapsto & \mathbf{x}_T = \{x_{\tau_1}, \cdots, x_{\tau_n}\} \end{array}$$

- *The triangulation $T \setminus \{\tau_k\} \cup \{\tau'_k\}$ obtained by flipping the arc τ_k corresponds to the mutation $\mu_k(\mathbf{x}_T) = \mathbf{x}_T \setminus \{x_{\tau_k}\} \cup \{x_{\tau'_k}\}$.*

Definition

The **surface cluster algebra** $\mathcal{A} = \mathcal{A}(S, M)$ associated to a surface (S, M) is a $\mathbb{Z}$-subalgebra of $\mathbb{Q}(x_1, \cdots, x_n)$ generated by all cluster variables x_γ.

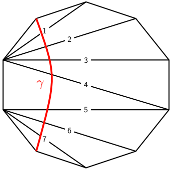
Snake graphs and perfect matchings

For each arc γ in a surface (S, M, T), we associate a weighted graph G_γ, called **snake graph**, from γ and T.

A perfect matching P of a graph G is a subset of the set of edges of G such that each vertex of G is incident to exactly one edge in P.

Snake graphs and perfect matchings

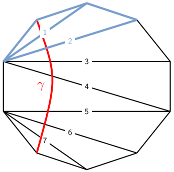
For each arc γ in a surface (S, M, T), we associate a weighted graph G_γ, called **snake graph**, from γ and T.



A **perfect matching** P of a graph G is a subset of the set of edges of G such that each vertex of G is incident to exactly one edge in P.

Snake graphs and perfect matchings

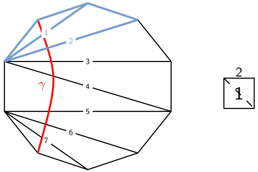
For each arc γ in a surface (S, M, T), we associate a weighted graph G_γ, called **snake graph**, from γ and T.



A **perfect matching** P of a graph G is a subset of the set of edges of G such that each vertex of G is incident to exactly one edge in P.

Snake graphs and perfect matchings

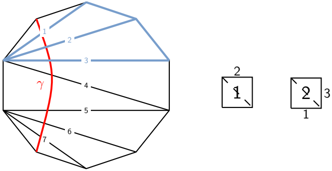
For each arc γ in a surface (S, M, T), we associate a weighted graph G_γ, called **snake graph**, from γ and T.



A **perfect matching** P of a graph G is a subset of the set of edges of G such that each vertex of G is incident to exactly one edge in P.

Snake graphs and perfect matchings

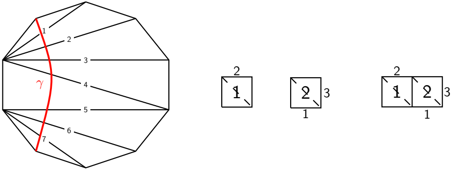
For each arc γ in a surface (S, M, T), we associate a weighted graph G_γ, called **snake graph**, from γ and T.



A **perfect matching** P of a graph G is a subset of the set of edges of G such that each vertex of G is incident to exactly one edge in P.

Snake graphs and perfect matchings

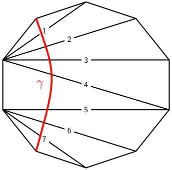
For each arc γ in a surface (S, M, T), we associate a weighted graph G_γ, called **snake graph**, from γ and T.

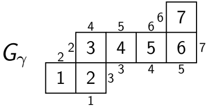


A **perfect matching** P of a graph G is a subset of the set of edges of G such that each vertex of G is incident to exactly one edge in P.

Snake graphs and perfect matchings

For each arc γ in a surface (S, M, T), we associate a weighted graph G_γ, called **snake graph**, from γ and T.

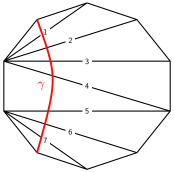


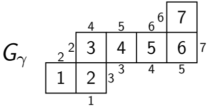


A **perfect matching** P of a graph G is a subset of the set of edges of G such that each vertex of G is incident to exactly one edge in P.

Snake graphs and perfect matchings

For each arc γ in a surface (S, M, T), we associate a weighted graph G_γ, called **snake graph**, from γ and T.

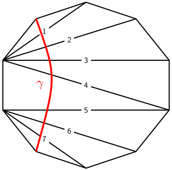




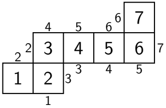
A **perfect matching** P of a graph G is a subset of the set of edges of G such that each vertex of G is incident to exactly one edge in P.

Snake graphs and perfect matchings

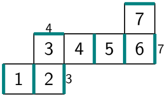
For each arc γ in a surface (S, M, T), we associate a weighted graph G_γ, called **snake graph**, from γ and T.



G_γ

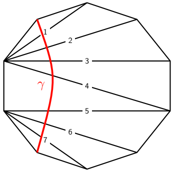


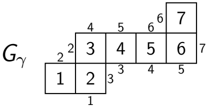
A **perfect matching** P of a graph G is a subset of the set of edges of G such that each vertex of G is incident to exactly one edge in P.



Snake graphs and perfect matchings

For each arc γ in a surface (S, M, T), we associate a weighted graph G_γ, called **snake graph**, from γ and T.





A **perfect matching** P of a graph G is a subset of the set of edges of G such that each vertex of G is incident to exactly one edge in P.

Expansion formula

The authors in [MSW] gives an explicit formula, called **expansion formula**, for cluster variables. The formula is given by

$$x_\gamma = \frac{1}{\operatorname{cross}(\gamma, T)} \sum_{P \vdash G_\gamma} x(P)y(P)$$

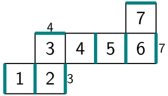
where the sum is over all perfect matchings P of G_γ.

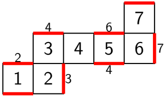
Expansion formula

The authors in [MSW] gives an explicit formula, called **expansion formula**, for cluster variables. The formula is given by

$$x_\gamma = \frac{1}{\operatorname{cross}(\gamma, T)} \sum_{P \vdash G_\gamma} x(P)y(P)$$

where the sum is over all perfect matchings P of G_γ.



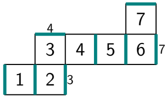


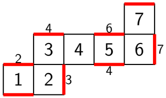
Expansion formula

The authors in [MSW] gives an explicit formula, called **expansion formula**, for cluster variables. The formula is given by

$$x_\gamma = \frac{1}{\operatorname{cross}(\gamma, T)} \sum_{P \vdash G_\gamma} x(P) y(P)$$

where the sum is over all perfect matchings P of G_γ.





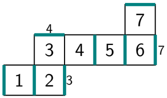
$$x(P) = x_3 x_4 x_7$$

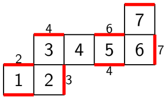
Expansion formula

The authors in [MSW] gives an explicit formula, called **expansion formula**, for cluster variables. The formula is given by

$$x_\gamma = \frac{1}{\mathrm{cross}\,(\gamma, T)} \sum_{P \vdash G_\gamma} x(P)y(P)$$

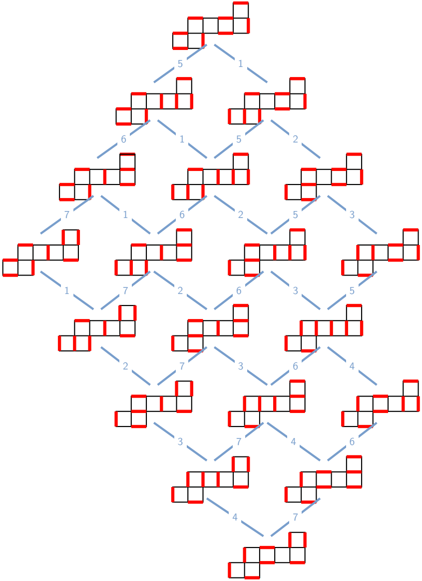
where the sum is over all perfect matchings P of G_γ.





$x(P) = x_3 x_4 x_7$

$x(P) = x_2 x_3 x_4^2 x_6 x_7$


5
1
6
1
5
2
7
1
6
2
5
3
1
7
2
6
3
5
2
7
3
6
4
3
7
4
6
4
7

Applying the formula, the cluster variable corresponding to the arc γ is given by

$$
\begin{aligned}
x_\gamma \quad = \quad & \frac{1}{x_1x_2x_3x_4x_5x_6x_7}\big(x_1x_2x_3x_5^2x_6 + y_4\ x_1x_2x_5x_6 + y_7\ x_1x_2x_3x_5^2 + \\
& y_3y_4\ x_1x_4x_5x_6 + y_4y_7\ x_1x_2x_5 + y_6y_7\ x_1x_2x_3x_5x_7 + \\
& y_2y_3y_4\ x_3x_4x_5x_6 + y_3y_4y_7\ x_1x_4x_5 + y_4y_6y_7\ x_1x_2x_7 + \\
& y_1y_2y_3y_4\ x_2x_3x_4x_5x_6 + y_2y_3y_4y_7\ x_3x_4x_5 + y_3y_4y_6y_7\ x_1x_4x_7 + \\
& y_4y_5y_6y_7\ x_1x_2x_4x_6x_7 + y_1y_2y_3y_4y_7\ x_2x_3x_4x_5 + y_2y_3y_4y_6y_7\ x_3x_4x_7 + \\
& y_3y_4y_5y_6y_7\ x_1x_4^2x_6x_7 + y_1y_2y_3y_4y_6y_7\ x_2x_3x_4x_7 + \\
& y_2y_3y_4y_5y_6y_7\ x_3x_4^2x_6x_7 + y_1y_2y_3y_4y_5y_6y_7\ x_2x_3x_4^2x_6x_7\big).
\end{aligned}
$$

Our results

- We introduce the notion of an **abstract snake graph**, which is not necessarily related to an arc in a surface.
- We define what it means for two abstract **snake graphs to cross.**
- Given two crossing snake graphs, we construct the **resolution** of the crossing as two pairs of snake graphs from the original pair of crossing snake graphs.
- We then prove that there is a **bijection** φ between the set of perfect matchings of the two crossing snake graphs and the set of perfect matchings of the resolution.
- We then apply our constructions to snake graphs arising from **unpunctured surfaces.**
- We then extend our results to **self-crossing snake graphs** associated to self-crossing arcs in a surface.

Our results

- We introduce the notion of an **abstract snake graph**, which is not necessarily related to an arc in a surface.
- We define what it means for two abstract **snake graphs to cross.**
- Given two crossing snake graphs, we construct the **resolution** of the crossing as two pairs of snake graphs from the original pair of crossing snake graphs.
- We then prove that there is a **bijection** φ between the set of perfect matchings of the two crossing snake graphs and the set of perfect matchings of the resolution.
- We then apply our constructions to snake graphs arising from **unpunctured surfaces.**
- We then extend our results to **self-crossing snake graphs** associated to self-crossing arcs in a surface.

Our results

- We introduce the notion of an **abstract snake graph**, which is not necessarily related to an arc in a surface.
- We define what it means for two abstract **snake graphs to cross.**
- Given two crossing snake graphs, we construct the **resolution** of the crossing as two pairs of snake graphs from the original pair of crossing snake graphs.
- We then prove that there is a **bijection** φ between the set of perfect matchings of the two crossing snake graphs and the set of perfect matchings of the resolution.
- We then apply our constructions to snake graphs arising from **unpunctured surfaces.**
- We then extend our results to **self-crossing snake graphs** associated to self-crossing arcs in a surface.

Our results

- We introduce the notion of an **abstract snake graph**, which is not necessarily related to an arc in a surface.
- We define what it means for two abstract **snake graphs to cross.**
- Given two crossing snake graphs, we construct the **resolution** of the crossing as two pairs of snake graphs from the original pair of crossing snake graphs.
- We then prove that there is a **bijection** φ between the set of perfect matchings of the two crossing snake graphs and the set of perfect matchings of the resolution.
- We then apply our constructions to snake graphs arising from **unpunctured surfaces.**
- We then extend our results to **self-crossing snake graphs** associated to self-crossing arcs in a surface.

Our results

- We introduce the notion of an **abstract snake graph**, which is not necessarily related to an arc in a surface.
- We define what it means for two abstract **snake graphs to cross.**
- Given two crossing snake graphs, we construct the **resolution** of the crossing as two pairs of snake graphs from the original pair of crossing snake graphs.
- We then prove that there is a **bijection** φ between the set of perfect matchings of the two crossing snake graphs and the set of perfect matchings of the resolution.
- We then apply our constructions to snake graphs arising from **unpunctured surfaces.**
- We then extend our results to **self-crossing snake graphs** associated to self-crossing arcs in a surface.

Our results

- We introduce the notion of an **abstract snake graph**, which is not necessarily related to an arc in a surface.
- We define what it means for two abstract **snake graphs to cross.**
- Given two crossing snake graphs, we construct the **resolution** of the crossing as two pairs of snake graphs from the original pair of crossing snake graphs.
- We then prove that there is a **bijection** φ between the set of perfect matchings of the two crossing snake graphs and the set of perfect matchings of the resolution.
- We then apply our constructions to snake graphs arising from **unpunctured surfaces.**
- We then extend our results to **self-crossing snake graphs** associated to self-crossing arcs in a surface.

Our results

- We introduce the notion of an **abstract snake graph**, which is not necessarily related to an arc in a surface.
- We define what it means for two abstract **snake graphs to cross.**
- Given two crossing snake graphs, we construct the **resolution** of the crossing as two pairs of snake graphs from the original pair of crossing snake graphs.
- We then prove that there is a **bijection** φ between the set of perfect matchings of the two crossing snake graphs and the set of perfect matchings of the resolution.
- We then apply our constructions to snake graphs arising from **unpunctured surfaces.**
- We then extend our results to **self-crossing snake graphs** associated to self-crossing arcs in a surface.

Abstract Snake Graphs

Definition

A **snake graph** $\mathcal{G}$ is a connected graph in $\mathbb{R}^2$ consisting of a finite sequence of tiles $G_1, G_2, \ldots, G_d$ with $d \geq 1$, such that for each $i = 1, \ldots, d-1$

(i) G_i and G_{i+1} share exactly one edge e_i and this edge is either the north edge of G_i and the south edge of G_{i+1} or the east edge of G_i and the west edge of G_{i+1}.

(ii) G_i and G_j have no edge in common whenever $|i-j| \geq 2$.

(ii) G_i and G_j are disjoint whenever $|i-j| \geq 3$.

Example

$\mathcal{G}$

Abstract Snake Graphs

Definition

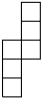
A **snake graph** $\mathcal{G}$ is a connected graph in $\mathbb{R}^2$ consisting of a finite sequence of tiles $G_1, G_2, \ldots, G_d$ with $d \geq 1$, such that for each $i = 1, \ldots, d-1$

(i) G_i and G_{i+1} share exactly one edge e_i and this edge is either the north edge of G_i and the south edge of G_{i+1} or the east edge of G_i and the west edge of G_{i+1}.

(ii) G_i and G_j have no edge in common whenever $|i-j| \geq 2$.

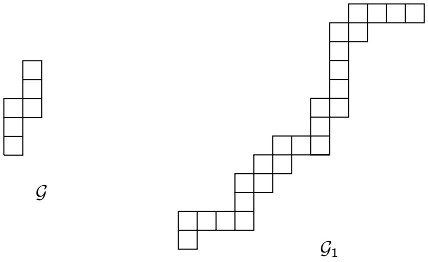
(ii) G_i and G_j are disjoint whenever $|i-j| \geq 3$.

Example



$\mathcal{G}$

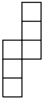
Example



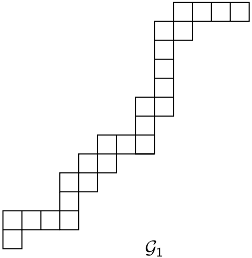
Notation

- $\mathcal{G} = (G_1, G_2, \ldots, G_d)$
- $\mathcal{G}[i, i+t] = (G_i, G_{i+1}, \ldots, G_{i+t})$
- We denote by e_i the interior edge between the tiles G_i and G_{i+1}.

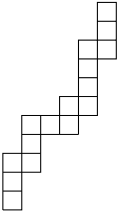
Example



$\mathcal{G}$



$\mathcal{G}_1$

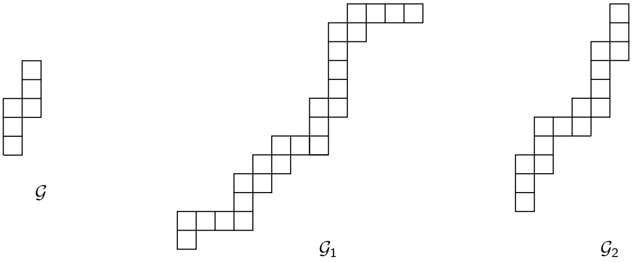


$\mathcal{G}_2$

Notation

- $\mathcal{G} = (G_1, G_2, \ldots, G_d)$
- $\mathcal{G}[i, i+t] = (G_i, G_{i+1}, \ldots, G_{i+t})$
- We denote by e_i the interior edge between the tiles G_i and G_{i+1}

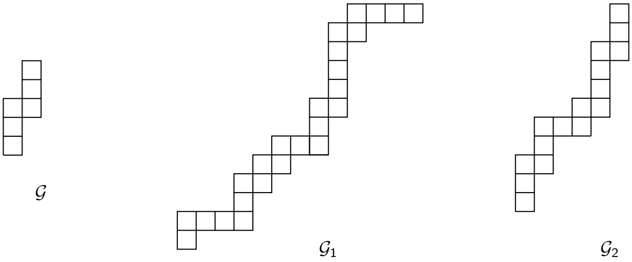
Example



Notation

- $\mathcal{G} = (G_1, G_2, \ldots, G_d)$
- $\mathcal{G}[i, i+t] = (G_i, G_{i+1}, \ldots, G_{i+t})$
- We denote by e_i the interior edge between the tiles G_i and G_{i+1}.

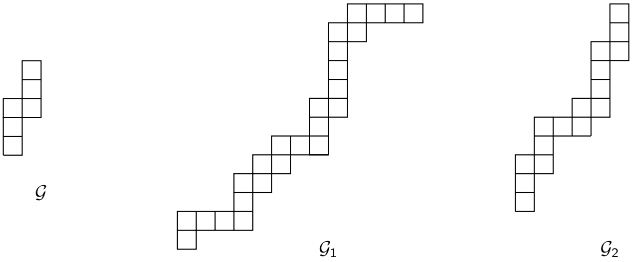
Example



Notation

- $\mathcal{G} = (G_1, G_2, \ldots, G_d)$
- $\mathcal{G}[i, i+t] = (G_i, G_{i+1}, \ldots, G_{i+t})$
- We denote by e_i the interior edge between the tiles G_i and G_{i+1}.

Example



Notation

- $\mathcal{G} = (G_1, G_2, \ldots, G_d)$
- $\mathcal{G}[i, i+t] = (G_i, G_{i+1}, \ldots, G_{i+t})$
- We denote by e_i the interior edge between the tiles G_i and G_{i+1}.

Local Overlaps

Definition

We say two snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ have a **local overlap** $\mathcal{G}$ if $\mathcal{G}$ is a maximal subgraph contained in both $\mathcal{G}_1$ and $\mathcal{G}_2$.

Notation: $\mathcal{G} \cong \mathcal{G}_1[s, \cdots, t] \cong \mathcal{G}_2[s', \cdots, t']$.

Example

Therefore $\mathcal{G}$ is a local overlap of $\mathcal{G}_1$ and $\mathcal{G}_2$.

- Note that two snake graphs may have several overlaps.

Local Overlaps

Definition

We say two snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ have a **local overlap** $\mathcal{G}$ if $\mathcal{G}$ is a maximal subgraph contained in both $\mathcal{G}_1$ and $\mathcal{G}_2$.

Notation: $\mathcal{G} \cong \mathcal{G}_1[s, \cdots, t] \cong \mathcal{G}_2[s', \cdots, t']$.

Example

Therefore $\mathcal{G}$ is a local overlap of $\mathcal{G}_1$ and $\mathcal{G}_2$.

- Note that two snake graphs may have several overlaps.

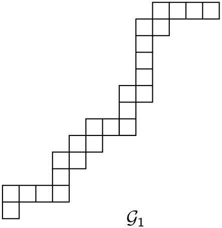
Local Overlaps

Definition

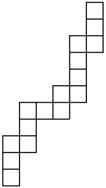
We say two snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ have a **local overlap** $\mathcal{G}$ if $\mathcal{G}$ is a maximal subgraph contained in both $\mathcal{G}_1$ and $\mathcal{G}_2$.

Notation: $\mathcal{G} \cong \mathcal{G}_1[s, \cdots, t] \cong \mathcal{G}_2[s', \cdots, t']$.

Example



$\mathcal{G}_1$



$\mathcal{G}_2$

Therefore $\mathcal{G}$ is a local overlap of $\mathcal{G}_1$ and $\mathcal{G}_2$.

- Note that two snake graphs may have several overlaps.

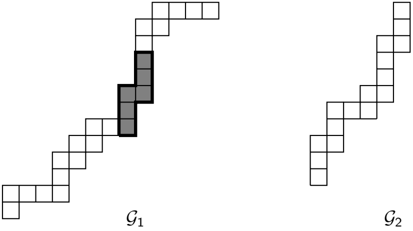
Local Overlaps

Definition

We say two snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ have a **local overlap** $\mathcal{G}$ if $\mathcal{G}$ is a maximal subgraph contained in both $\mathcal{G}_1$ and $\mathcal{G}_2$.

Notation: $\mathcal{G} \cong \mathcal{G}_1[s, \cdots, t] \cong \mathcal{G}_2[s', \cdots, t']$.

Example



Therefore $\mathcal{G}$ is a local overlap of $\mathcal{G}_1$ and $\mathcal{G}_2$.

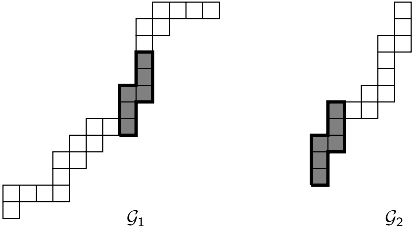
Local Overlaps

Definition

We say two snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ have a **local overlap** $\mathcal{G}$ if $\mathcal{G}$ is a maximal subgraph contained in both $\mathcal{G}_1$ and $\mathcal{G}_2$.

Notation: $\mathcal{G} \cong \mathcal{G}_1[s, \cdots, t] \cong \mathcal{G}_2[s', \cdots, t']$.

Example



$\mathcal{G}_1$ $\mathcal{G}_2$

Therefore $\mathcal{G}$ is a local overlap of $\mathcal{G}_1$ and $\mathcal{G}_2$.

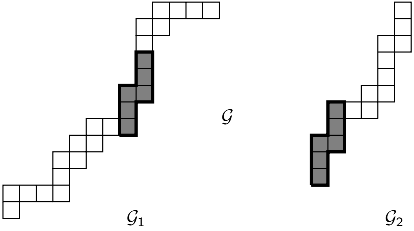
Local Overlaps

Definition

We say two snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ have a **local overlap** $\mathcal{G}$ if $\mathcal{G}$ is a maximal subgraph contained in both $\mathcal{G}_1$ and $\mathcal{G}_2$.

Notation: $\mathcal{G} \cong \mathcal{G}_1[s, \cdots, t] \cong \mathcal{G}_2[s', \cdots, t']$.

Example



Therefore $\mathcal{G}$ is a local overlap of $\mathcal{G}_1$ and $\mathcal{G}_2$.

- Note that two snake graphs may have several overlaps.

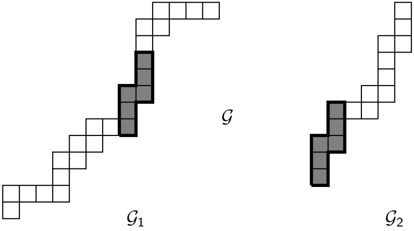
Local Overlaps

Definition

We say two snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ have a **local overlap** $\mathcal{G}$ if $\mathcal{G}$ is a maximal subgraph contained in both $\mathcal{G}_1$ and $\mathcal{G}_2$.

Notation: $\mathcal{G} \cong \mathcal{G}_1[s, \cdots, t] \cong \mathcal{G}_2[s', \cdots, t']$.

Example



Therefore $\mathcal{G}$ is a local overlap of $\mathcal{G}_1$ and $\mathcal{G}_2$.

- Note that two snake graphs may have several overlaps.

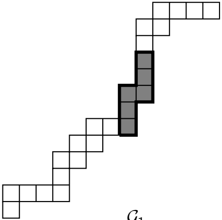
Local Overlaps

Definition

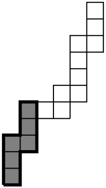
We say two snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ have a **local overlap** $\mathcal{G}$ if $\mathcal{G}$ is a maximal subgraph contained in both $\mathcal{G}_1$ and $\mathcal{G}_2$.

Notation: $\mathcal{G} \cong \mathcal{G}_1[s, \cdots, t] \cong \mathcal{G}_2[s', \cdots, t']$.

Example



$\mathcal{G}_1$



$\mathcal{G}_2$

Therefore $\mathcal{G}$ is a local overlap of $\mathcal{G}_1$ and $\mathcal{G}_2$.

- Note that two snake graphs may have several overlaps.

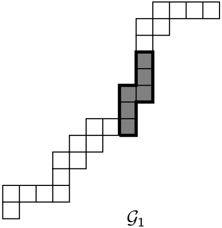
Local Overlaps

Definition

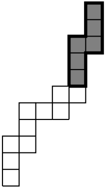
We say two snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ have a **local overlap** $\mathcal{G}$ if $\mathcal{G}$ is a maximal subgraph contained in both $\mathcal{G}_1$ and $\mathcal{G}_2$.

Notation: $\mathcal{G} \cong \mathcal{G}_1[s, \cdots, t] \cong \mathcal{G}_2[s', \cdots, t']$.

Example



$\mathcal{G}_1$



$\mathcal{G}_2$

Therefore $\mathcal{G}$ is a local overlap of $\mathcal{G}_1$ and $\mathcal{G}_2$.

- Note that two snake graphs may have several overlaps.

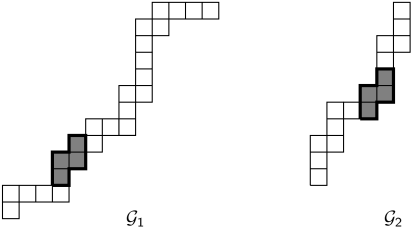
Local Overlaps

Definition

We say two snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ have a **local overlap** $\mathcal{G}$ if $\mathcal{G}$ is a maximal subgraph contained in both $\mathcal{G}_1$ and $\mathcal{G}_2$.

Notation: $\mathcal{G} \cong \mathcal{G}_1[s, \cdots, t] \cong \mathcal{G}_2[s', \cdots, t']$.

Example



Therefore $\mathcal{G}$ is a local overlap of $\mathcal{G}_1$ and $\mathcal{G}_2$.

- Note that two snake graphs may have several overlaps.

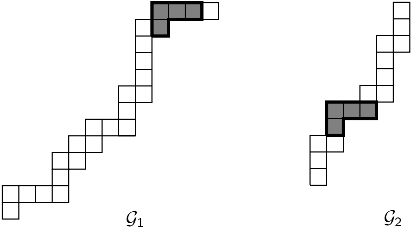
Local Overlaps

Definition

We say two snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ have a **local overlap** $\mathcal{G}$ if $\mathcal{G}$ is a maximal subgraph contained in both $\mathcal{G}_1$ and $\mathcal{G}_2$.

Notation: $\mathcal{G} \cong \mathcal{G}_1[s, \cdots, t] \cong \mathcal{G}_2[s', \cdots, t']$.

Example



Therefore $\mathcal{G}$ is a local overlap of $\mathcal{G}_1$ and $\mathcal{G}_2$.

- Note that two snake graphs may have several overlaps.

Sign Function

Definition

A **sign function** f on a snake graph $\mathcal{G}$ is a map f from the set of edges of $\mathcal{G}$ to $\{+, -\}$ such that on every tile in $\mathcal{G}$ the north and the west edge have the same sign, the south and the east edge have the same sign and the sign on the north edge is opposite to the sign on the south edge.

Example

A sign function on $\mathcal{G}_1$ and $\mathcal{G}_2$

Sign Function

Definition

A **sign function** f on a snake graph $\mathcal{G}$ is a map f from the set of edges of $\mathcal{G}$ to $\{+,-\}$ such that on every tile in $\mathcal{G}$ the north and the west edge have the same sign, the south and the east edge have the same sign and the sign on the north edge is opposite to the sign on the south edge.

Example

A sign function on $\mathcal{G}_1$ and $\mathcal{G}_2$

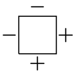
Sign Function

Definition

A **sign function** f on a snake graph $\mathcal{G}$ is a map f from the set of edges of $\mathcal{G}$ to $\{+,-\}$ such that on every tile in $\mathcal{G}$ the north and the west edge have the same sign, the south and the east edge have the same sign and the sign on the north edge is opposite to the sign on the south edge.

Example

A sign function on $\mathcal{G}_1$ and $\mathcal{G}_2$



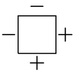
Sign Function

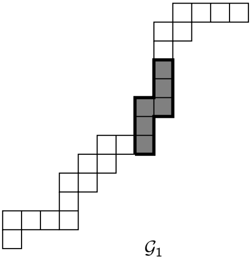
Definition

A **sign function** f on a snake graph $\mathcal{G}$ is a map f from the set of edges of $\mathcal{G}$ to $\{+,-\}$ such that on every tile in $\mathcal{G}$ the north and the west edge have the same sign, the south and the east edge have the same sign and the sign on the north edge is opposite to the sign on the south edge.

Example

A sign function on $\mathcal{G}_1$ and $\mathcal{G}_2$





$\mathcal{G}_1$

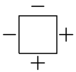
Sign Function

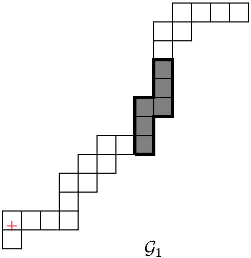
Definition

A **sign function** f on a snake graph $\mathcal{G}$ is a map f from the set of edges of $\mathcal{G}$ to $\{+, -\}$ such that on every tile in $\mathcal{G}$ the north and the west edge have the same sign, the south and the east edge have the same sign and the sign on the north edge is opposite to the sign on the south edge.

Example

A sign function on $\mathcal{G}_1$ and $\mathcal{G}_2$





$\mathcal{G}_1$

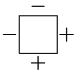
Sign Function

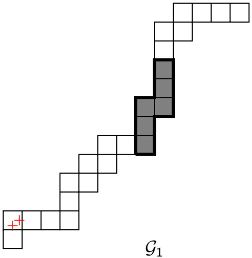
Definition

A **sign function** f on a snake graph $\mathcal{G}$ is a map f from the set of edges of $\mathcal{G}$ to $\{+,-\}$ such that on every tile in $\mathcal{G}$ the north and the west edge have the same sign, the south and the east edge have the same sign and the sign on the north edge is opposite to the sign on the south edge.

Example

A sign function on $\mathcal{G}_1$ and $\mathcal{G}_2$





$\mathcal{G}_1$

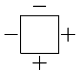
Sign Function

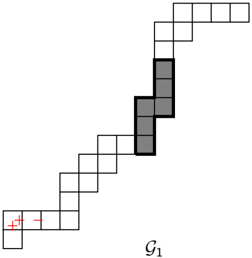
Definition

A **sign function** f on a snake graph $\mathcal{G}$ is a map f from the set of edges of $\mathcal{G}$ to $\{+,-\}$ such that on every tile in $\mathcal{G}$ the north and the west edge have the same sign, the south and the east edge have the same sign and the sign on the north edge is opposite to the sign on the south edge.

Example

A sign function on $\mathcal{G}_1$ and $\mathcal{G}_2$





$\mathcal{G}_1$

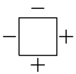
Sign Function

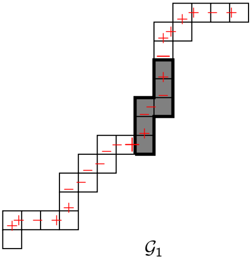
Definition

A **sign function** f on a snake graph $\mathcal{G}$ is a map f from the set of edges of $\mathcal{G}$ to $\{+,-\}$ such that on every tile in $\mathcal{G}$ the north and the west edge have the same sign, the south and the east edge have the same sign and the sign on the north edge is opposite to the sign on the south edge.

Example

A sign function on $\mathcal{G}_1$ and $\mathcal{G}_2$





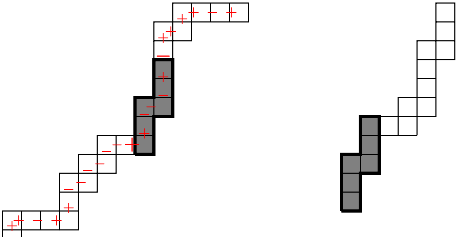
Sign Function

Definition

A **sign function** f on a snake graph $\mathcal{G}$ is a map f from the set of edges of $\mathcal{G}$ to $\{+, -\}$ such that on every tile in $\mathcal{G}$ the north and the west edge have the same sign, the south and the east edge have the same sign and the sign on the north edge is opposite to the sign on the south edge.

Example

A sign function on $\mathcal{G}_1$ and $\mathcal{G}_2$



$\mathcal{G}_1$ $\mathcal{G}_2$

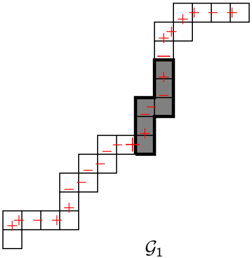
Sign Function

Definition

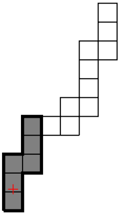
A **sign function** f on a snake graph $\mathcal{G}$ is a map f from the set of edges of $\mathcal{G}$ to $\{+,-\}$ such that on every tile in $\mathcal{G}$ the north and the west edge have the same sign, the south and the east edge have the same sign and the sign on the north edge is opposite to the sign on the south edge.

Example

A sign function on $\mathcal{G}_1$ and $\mathcal{G}_2$



$\mathcal{G}_1$



$\mathcal{G}_2$

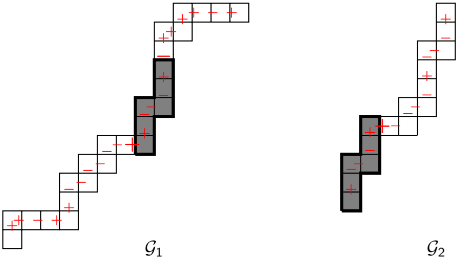
Sign Function

Definition

A **sign function** f on a snake graph $\mathcal{G}$ is a map f from the set of edges of $\mathcal{G}$ to $\{+,-\}$ such that on every tile in $\mathcal{G}$ the north and the west edge have the same sign, the south and the east edge have the same sign and the sign on the north edge is opposite to the sign on the south edge.

Example

A sign function on $\mathcal{G}_1$ and $\mathcal{G}_2$



Crossing

Definition

We say that $\mathcal{G}_1$ and $\mathcal{G}_2$ **cross in a local overlap** $\mathcal{G}$ if one of the following conditions hold.

- $f_1(e_{s-1}) = -f_1(e_t)$ if $s > 1,\ t < d$
- $f_1(e_{s-1}) = f_2(e'_{t'})$ if $s > 1,\ t < d,\ s' = 1,\ t' < d'$

Example

$\mathcal{G}_1$ and $\mathcal{G}_2$ cross at the overlap $\mathcal{G}$.

Crossing

Definition

We say that $\mathcal{G}_1$ and $\mathcal{G}_2$ **cross in a local overlap** $\mathcal{G}$ if one of the following conditions hold.

- $f_1(e_{s-1}) = -f_1(e_t)$ if $s > 1,\ t < d$
- $f_1(e_{s-1}) = f_2(e'_{t'})$ if $s > 1,\ t < d,\ s' = 1,\ t' < d'$

Example

$\mathcal{G}_1$ and $\mathcal{G}_2$ cross at the overlap $\mathcal{G}$.

Crossing

Definition

We say that $\mathcal{G}_1$ and $\mathcal{G}_2$ **cross in a local overlap** $\mathcal{G}$ if one of the following conditions hold.

- $f_1(e_{s-1}) = -f_1(e_t)$ if $s > 1, \ t < d$
- $f_1(e_{s-1}) = f_2(e'_{t'})$ if $s > 1, \ t < d, \ s' = 1, \ t' < d'$

Example

$\mathcal{G}_1$ and $\mathcal{G}_2$ cross at the overlap $\mathcal{G}$.

$\mathcal{G}_1$ $\mathcal{G}_2$

Crossing

Definition

We say that $\mathcal{G}_1$ and $\mathcal{G}_2$ **cross in a local overlap** $\mathcal{G}$ if one of the following conditions hold.

- $f_1(e_{s-1}) = -f_1(e_t)$ if $s > 1,\ t < d$
- $f_1(e_{s-1}) = f_2(e'_{t'})$ if $s > 1,\ t < d,\ s' = 1,\ t' < d'$

Example

$\mathcal{G}_1$ and $\mathcal{G}_2$ cross at the overlap $\mathcal{G}$.

Crossing

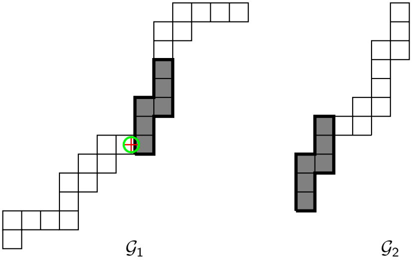
Definition

We say that $\mathcal{G}_1$ and $\mathcal{G}_2$ **cross in a local overlap** $\mathcal{G}$ if one of the following conditions hold.

- $f_1(e_{s-1}) = -f_1(e_t)$ if $s > 1,\ t < d$
- $f_1(e_{s-1}) = f_2(e'_{t'})$ if $s > 1,\ t < d,\ s' = 1,\ t' < d'$

Example

$\mathcal{G}_1$ and $\mathcal{G}_2$ cross at the overlap $\mathcal{G}$.



Crossing

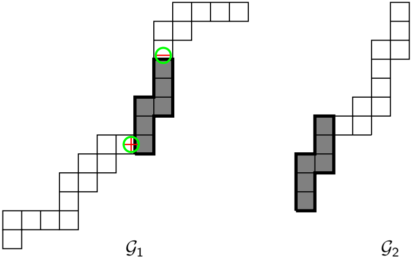
Definition

We say that $\mathcal{G}_1$ and $\mathcal{G}_2$ **cross in a local overlap** $\mathcal{G}$ if one of the following conditions hold.

- $f_1(e_{s-1}) = -f_1(e_t)$ if $s > 1,\ t < d$
- $f_1(e_{s-1}) = f_2(e'_{t'})$ if $s > 1,\ t < d,\ s' = 1,\ t' < d'$

Example

$\mathcal{G}_1$ and $\mathcal{G}_2$ cross at the overlap $\mathcal{G}$.



Crossing

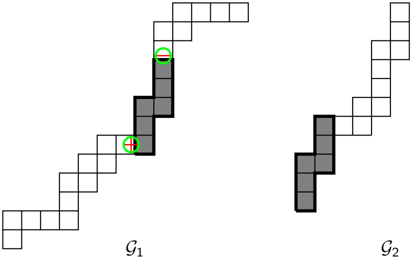
Definition

We say that $\mathcal{G}_1$ and $\mathcal{G}_2$ **cross in a local overlap** $\mathcal{G}$ if one of the following conditions hold.

- $f_1(e_{s-1}) = -f_1(e_t)$ if $s > 1, \ t < d$
- $f_1(e_{s-1}) = f_2(e'_{t'})$ if $s > 1, \ t < d, \ s' = 1, \ t' < d'$

Example

$\mathcal{G}_1$ and $\mathcal{G}_2$ cross at the overlap $\mathcal{G}$.



Crossing

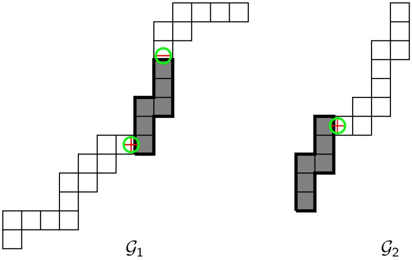
Definition

We say that $\mathcal{G}_1$ and $\mathcal{G}_2$ **cross in a local overlap** $\mathcal{G}$ if one of the following conditions hold.

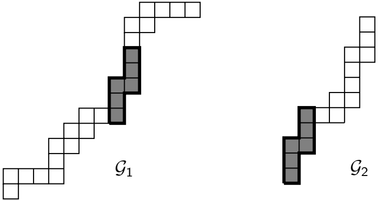
- $f_1(e_{s-1}) = -f_1(e_t)$ if $s > 1,\ t < d$
- $f_1(e_{s-1}) = f_2(e'_{t'})$ if $s > 1,\ t < d,\ s' = 1,\ t' < d'$

Example

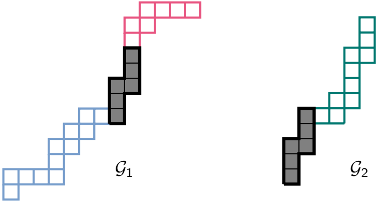
$\mathcal{G}_1$ and $\mathcal{G}_2$ cross at the overlap $\mathcal{G}$.



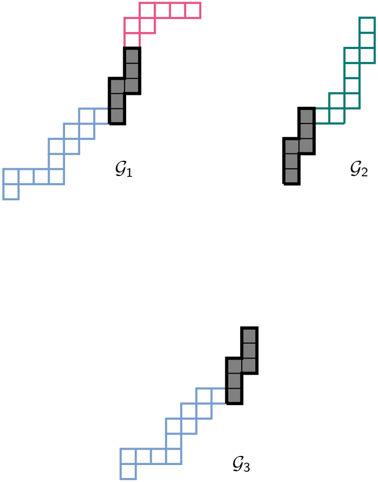
Example: Resolution $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$



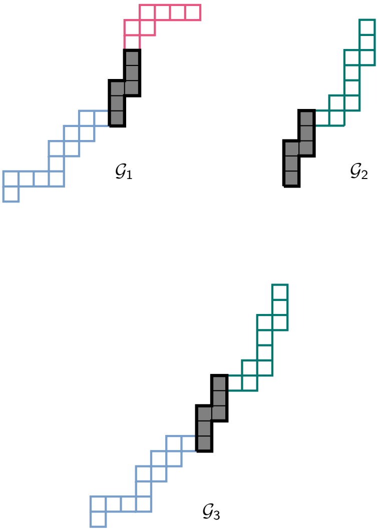
Example: Resolution $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$



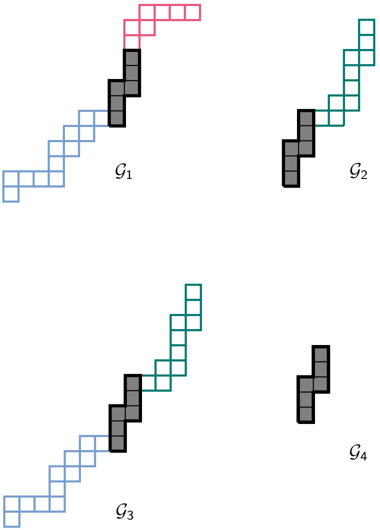
Example: Resolution $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$



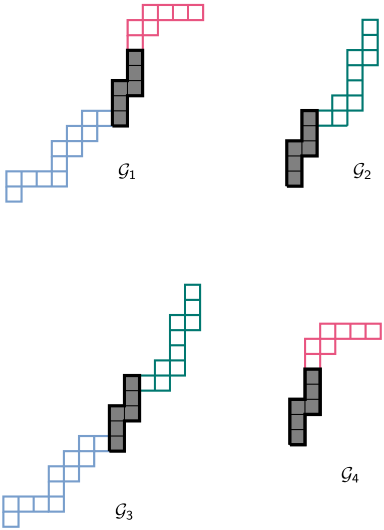
Example: Resolution $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$



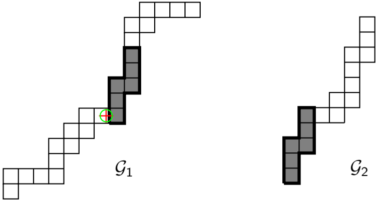
Example: Resolution $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$



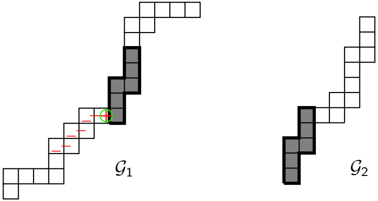
Example: Resolution $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$



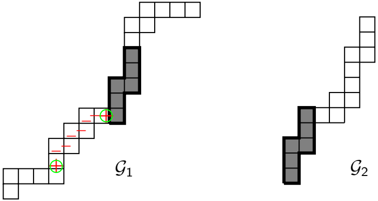
Example: Resolution $\text{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$



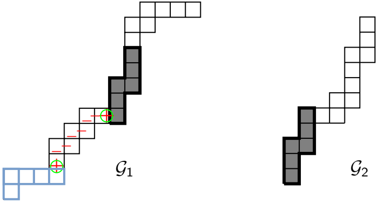
Example: Resolution $\text{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$



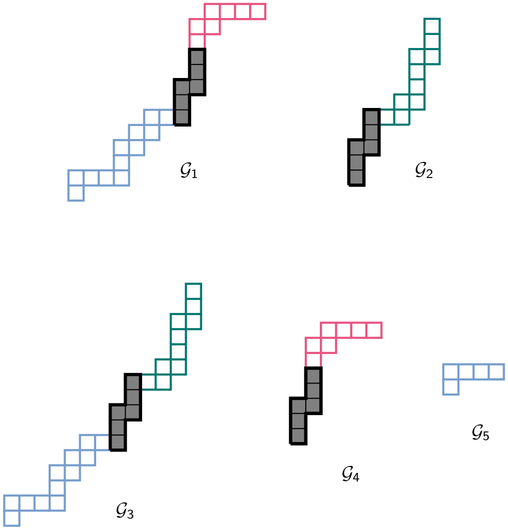
Example: Resolution $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$



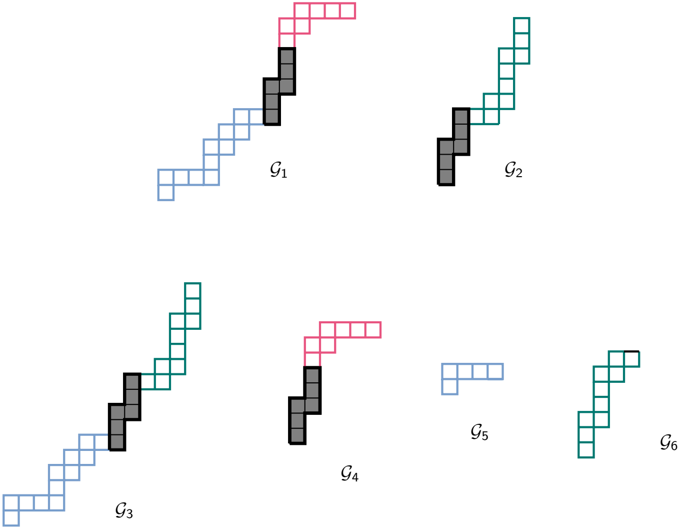
Example: Resolution $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$



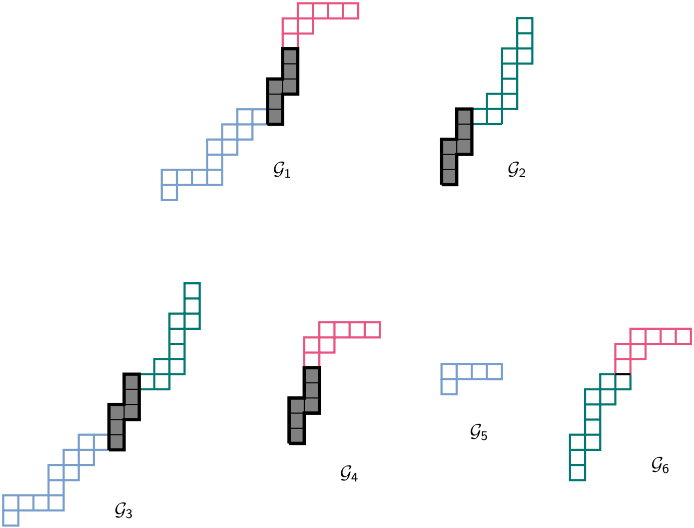
Example: Resolution (Continued)



Example: Resolution (Continued)



Example: Resolution (Continued)



Resolution: Definition

Assumption: We will assume that $s > 1$, $t < d$, $s' = 1$ and $t' < d'$. For all other cases, see [CS].

We define four connected snakegraphs as follows.

- $\mathcal{G}_3 = \mathcal{G}_1[1, t] \cup \mathcal{G}_2[t' + 1, d']$,
- $\mathcal{G}_4 = \mathcal{G}_2[1, t'] \cup \mathcal{G}_1[t + 1, d]$,
- $\mathcal{G}_5 = \mathcal{G}_1[1, k]$ where $k < s - 1$ is the largest integer such that the sign on the interior edge between tiles k and $k + 1$ is the same as the sign on the interior edge of tiles $s - 1$ and s,
- $\mathcal{G}_6 = \overline{\mathcal{G}}_2[d', t' + 1] \cup \mathcal{G}_1[t + 1, d]$ where the two subgraphs are glued along the south G_{t+1} and the north of $G'_{t'+1}$ if G_{t+1} is north of G_t in $\mathcal{G}_1$.

Definition

The **resolution of the crossing** of $\mathcal{G}_1$ and $\mathcal{G}_2$ in $\mathcal{G}$ is defined to be $(\mathcal{G}_3 \sqcup \mathcal{G}_4, \mathcal{G}_5 \sqcup \mathcal{G}_6)$ and is denoted by $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$.

Resolution: Definition

Assumption: We will assume that $s > 1$, $t < d$, $s' = 1$ and $t' < d'$. For all other cases, see [CS].

We define four connected snakegraphs as follows.

- $\mathcal{G}_3 = \mathcal{G}_1[1,t] \cup \mathcal{G}_2[t'+1, d']$,
- $\mathcal{G}_4 = \mathcal{G}_2[1,t'] \cup \mathcal{G}_1[t+1, d]$,
- $\mathcal{G}_5 = \mathcal{G}_1[1,k]$ where $k < s-1$ is the largest integer such that the sign on the interior edge between tiles k and $k+1$ is the same as the sign on the interior edge of tiles $s-1$ and s,
- $\mathcal{G}_6 = \overline{\mathcal{G}}_2[d', t'+1] \cup \mathcal{G}_1[t+1, d]$ where the two subgraphs are glued along the south G_{t+1} and the north of $G'_{t'+1}$ if G_{t+1} is north of G_t in $\mathcal{G}_1$.

Definition

The **resolution of the crossing** of $\mathcal{G}_1$ and $\mathcal{G}_2$ in $\mathcal{G}$ is defined to be $(\mathcal{G}_3 \sqcup \mathcal{G}_4, \mathcal{G}_5 \sqcup \mathcal{G}_6)$ and is denoted by $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$.

Resolution: Definition

Assumption: We will assume that $s > 1$, $t < d$, $s' = 1$ and $t' < d'$. For all other cases, see [CS].

We define four connected snakegraphs as follows.

- $\mathcal{G}_3 = \mathcal{G}_1[1, t] \cup \mathcal{G}_2[t' + 1, d']$,
- $\mathcal{G}_4 = \mathcal{G}_2[1, t'] \cup \mathcal{G}_1[t + 1, d]$,
- $\mathcal{G}_5 = \mathcal{G}_1[1, k]$ where $k < s - 1$ is the largest integer such that the sign on the interior edge between tiles k and $k + 1$ is the same as the sign on the interior edge of tiles $s - 1$ and s,
- $\mathcal{G}_6 = \overline{\mathcal{G}}_2[d', t' + 1] \cup \mathcal{G}_1[t + 1, d]$ where the two subgraphs are glued along the south G_{t+1} and the north of $G'_{t'+1}$ if G_{t+1} is north of G_t in $\mathcal{G}_1$.

Definition

The **resolution of the crossing** of $\mathcal{G}_1$ and $\mathcal{G}_2$ in $\mathcal{G}$ is defined to be $(\mathcal{G}_3 \sqcup \mathcal{G}_4, \mathcal{G}_5 \sqcup \mathcal{G}_6)$ and is denoted by $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$.

Bijection of Perfect Matchings

- Let Match (G) denote the set of all perfect matchings of the graph G and
  $\text{Match}\,(\text{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)) = \text{Match}\,(\mathcal{G}_3 \sqcup \mathcal{G}_4) \cup \text{Match}\,(\mathcal{G}_5 \sqcup \mathcal{G}_6)$.

Theorem (CS)

Let $\mathcal{G}_1, \mathcal{G}_2$ be two snake graphs. Then there is a bijection

$$\text{Match}\,(\mathcal{G}_1 \sqcup \mathcal{G}_2) \longrightarrow \text{Match}\,(\text{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2))$$

- Note that we construct the bijection map and its inverse map explicitly.

Bijection of Perfect Matchings

- Let $\mathrm{Match}\,(G)$ denote the set of all perfect matchings of the graph G and $\mathrm{Match}\,(\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)) = \mathrm{Match}\,(\mathcal{G}_3 \sqcup \mathcal{G}_4) \cup \mathrm{Match}\,(\mathcal{G}_5 \sqcup \mathcal{G}_6)$.

Theorem (CS)

Let $\mathcal{G}_1, \mathcal{G}_2$ be two snake graphs. Then there is a bijection

$$\mathrm{Match}\,(\mathcal{G}_1 \sqcup \mathcal{G}_2) \longrightarrow \mathrm{Match}\,(\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2))$$

- Note that we construct the bijection map and its inverse map explicitly.

Bijection of Perfect Matchings

- Let $\mathrm{Match}\,(G)$ denote the set of all perfect matchings of the graph G and
$\mathrm{Match}\,(\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1,\mathcal{G}_2)) = \mathrm{Match}\,(\mathcal{G}_3 \sqcup \mathcal{G}_4) \cup \mathrm{Match}\,(\mathcal{G}_5 \sqcup \mathcal{G}_6)$.

Theorem (CS)

Let $\mathcal{G}_1, \mathcal{G}_2$ be two snake graphs. Then there is a bijection

$$\mathrm{Match}\,(\mathcal{G}_1 \sqcup \mathcal{G}_2) \longrightarrow \mathrm{Match}\,(\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1,\mathcal{G}_2))$$

- Note that we construct the bijection map and its inverse map explicitly.

Bijection of Perfect Matchings

- Let Match (G) denote the set of all perfect matchings of the graph G and
Match $(\text{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)) = \text{Match}\,(\mathcal{G}_3 \sqcup \mathcal{G}_4) \cup \text{Match}\,(\mathcal{G}_5 \sqcup \mathcal{G}_6)$.

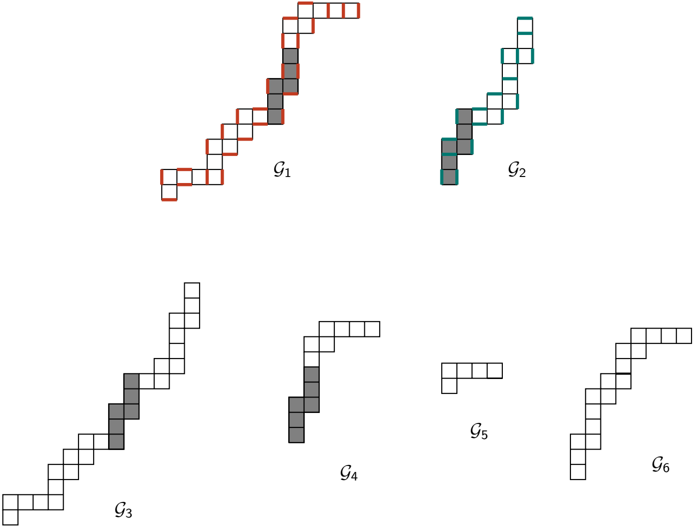
Theorem (CS)

Let $\mathcal{G}_1, \mathcal{G}_2$ be two snake graphs. Then there is a bijection

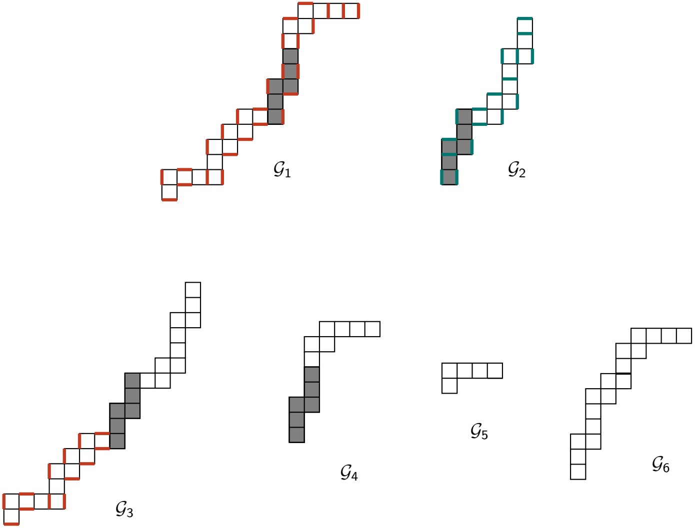
$$\text{Match}\,(\mathcal{G}_1 \sqcup \mathcal{G}_2) \longrightarrow \text{Match}\,(\text{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2))$$

- Note that we construct the bijection map and its inverse map explicitly.

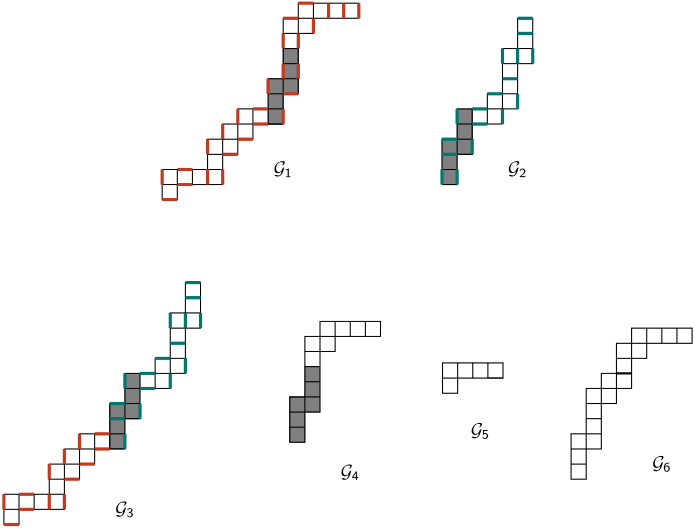
'Idea' of proof



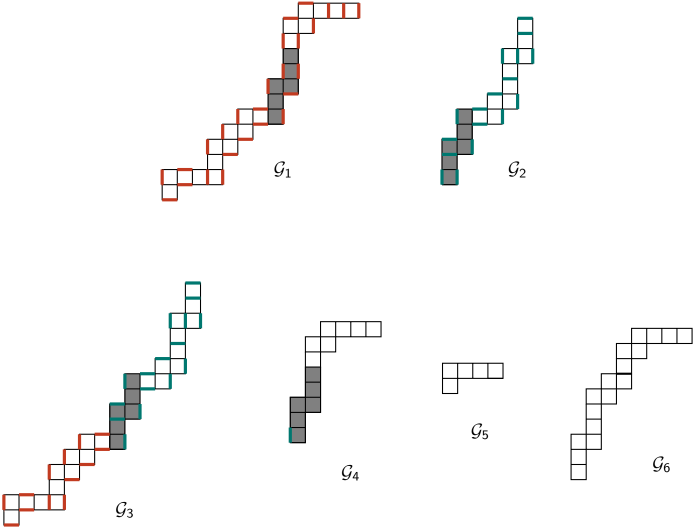
'Idea' of proof



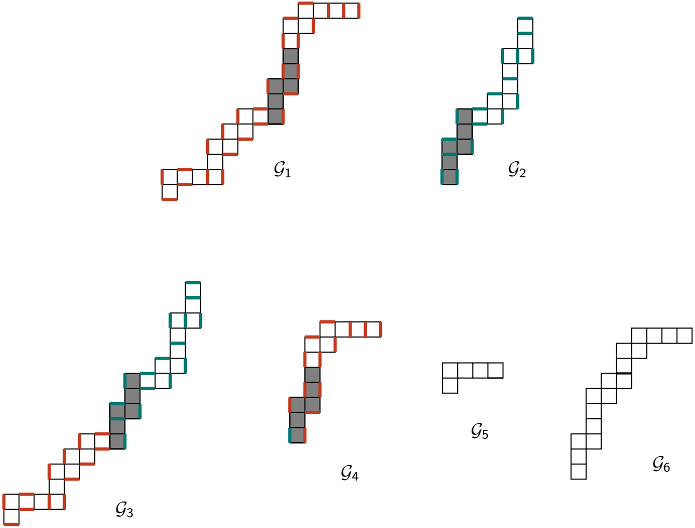
'Idea' of proof



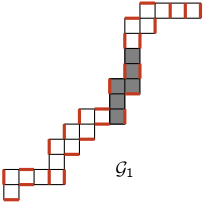
'Idea' of proof



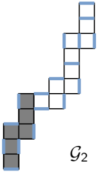
'Idea' of proof



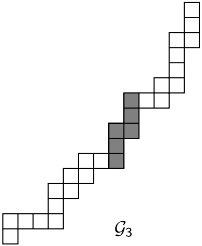
'Idea' of proof



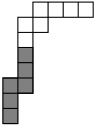
$\mathcal{G}_1$



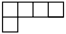
$\mathcal{G}_2$



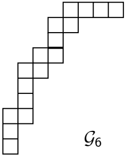
$\mathcal{G}_3$



$\mathcal{G}_4$

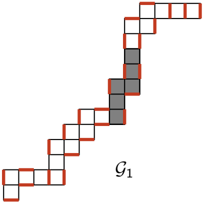


$\mathcal{G}_5$

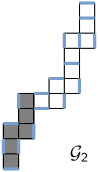


$\mathcal{G}_6$

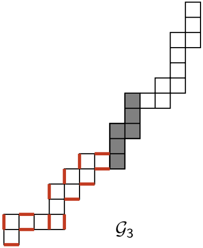
‘Idea’ of proof



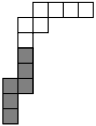
$\mathcal{G}_1$



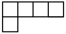
$\mathcal{G}_2$



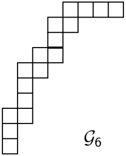
$\mathcal{G}_3$



$\mathcal{G}_4$

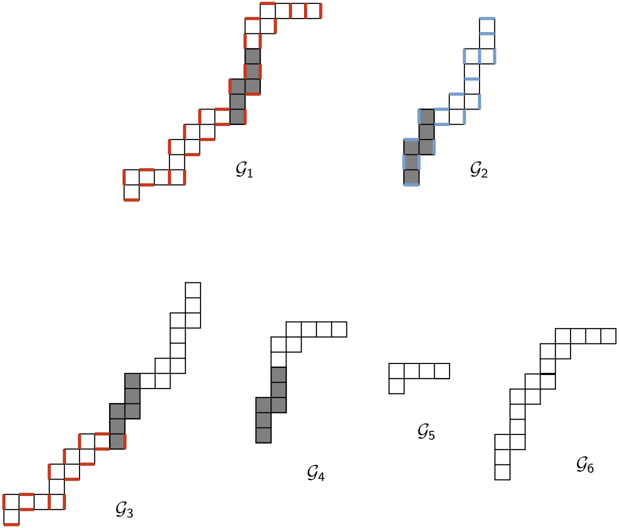


$\mathcal{G}_5$

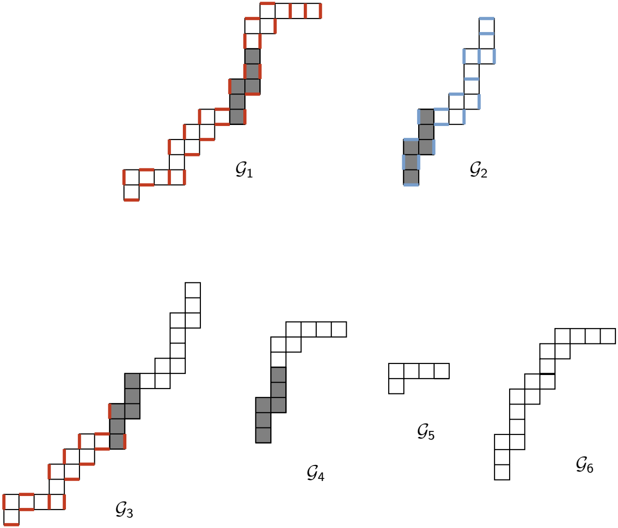


$\mathcal{G}_6$

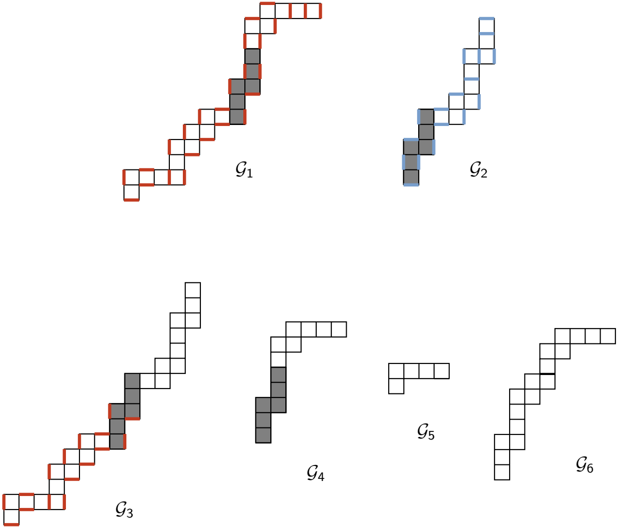
'Idea' of proof



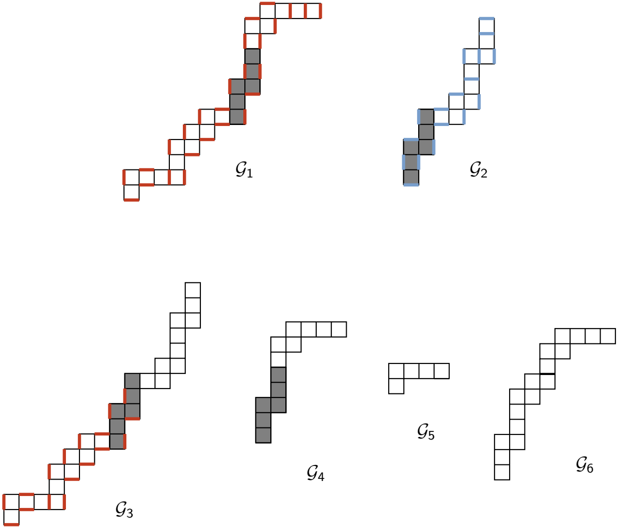
‘Idea’ of proof



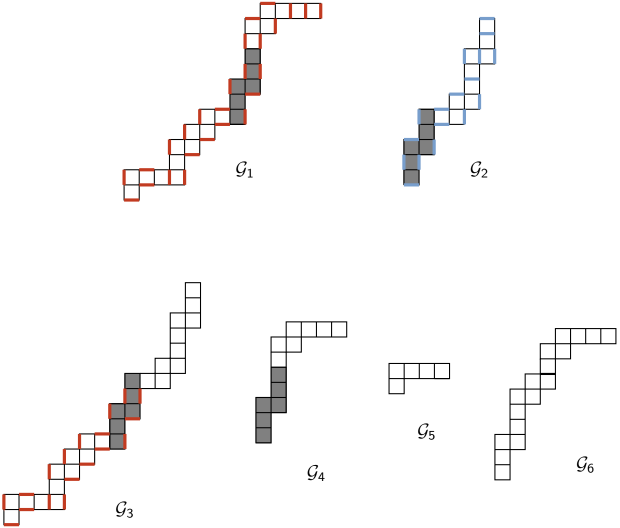
‘Idea’ of proof



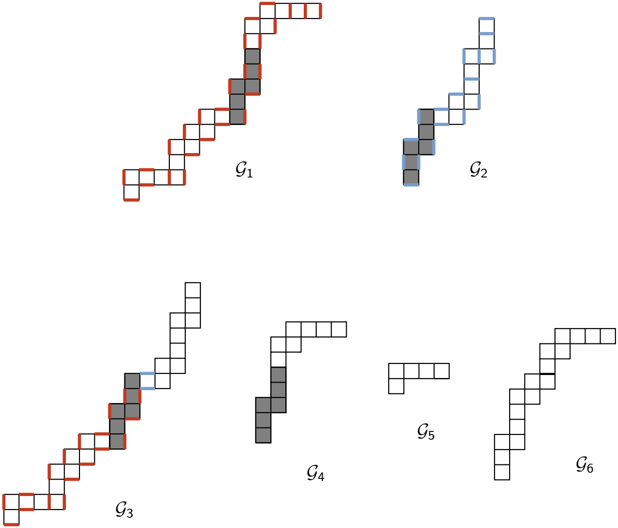
‘Idea’ of proof



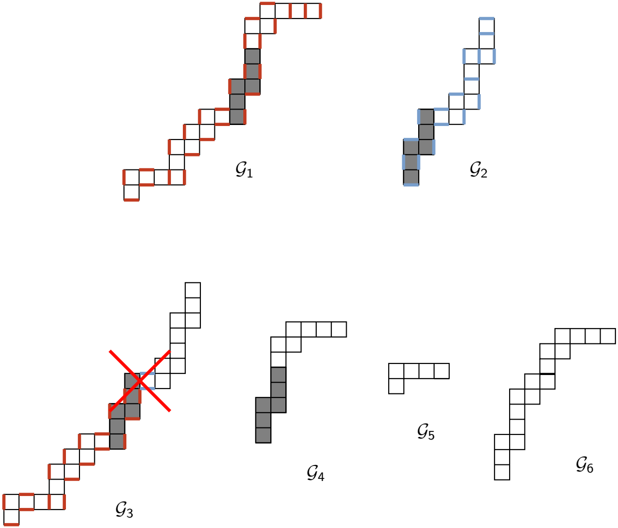
'Idea' of proof



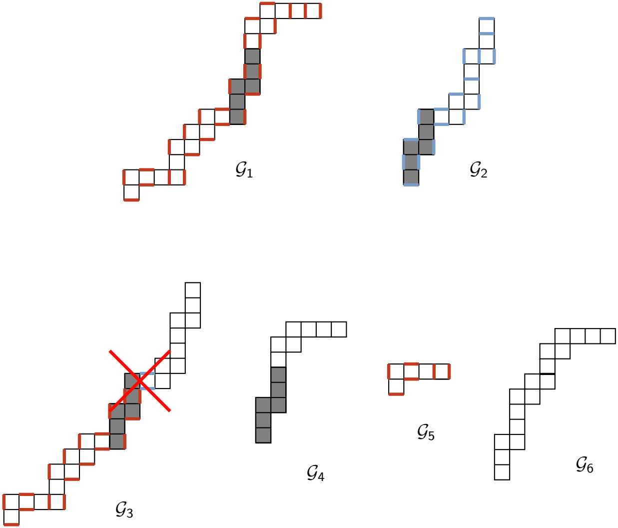
'Idea' of proof



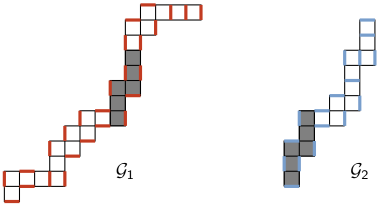
‘Idea’ of proof

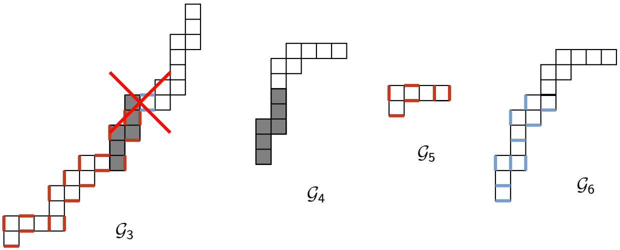


'Idea' of proof

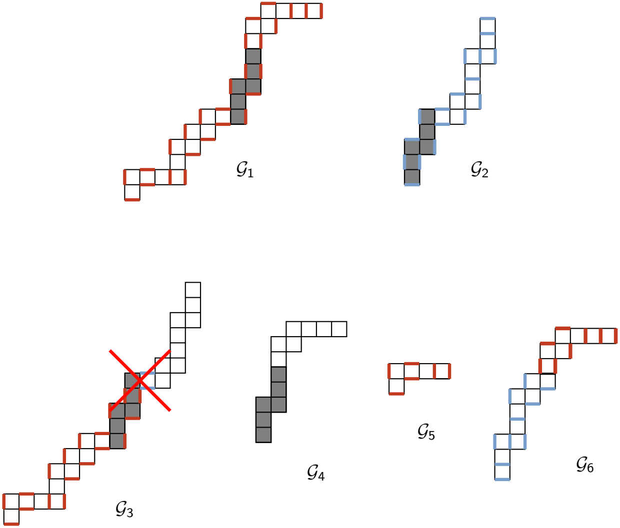


'Idea' of proof

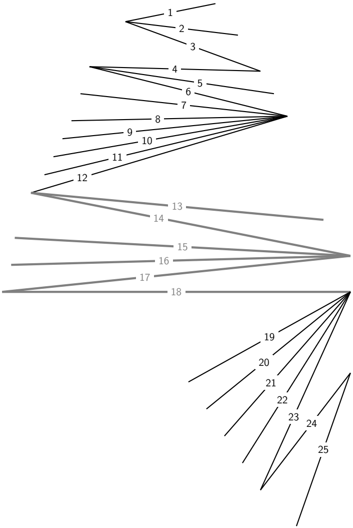




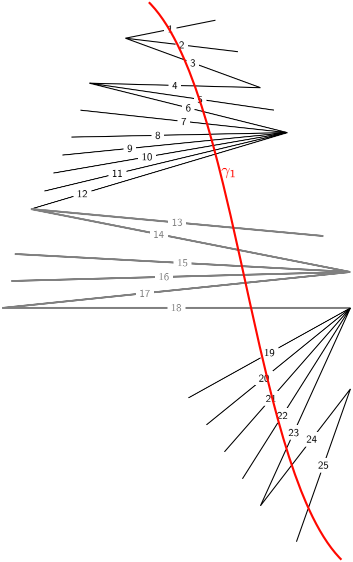
'Idea' of proof



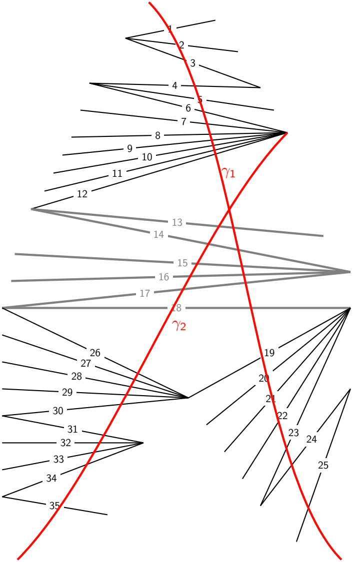
Surface Example



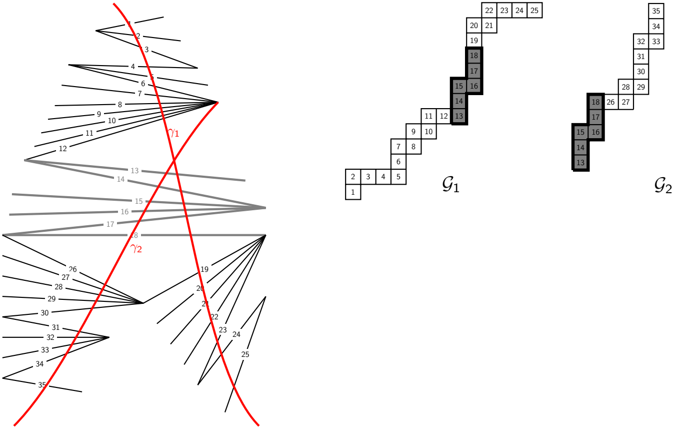
Surface Example



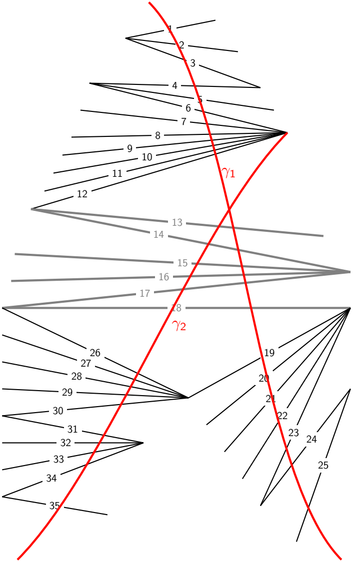
Surface Example



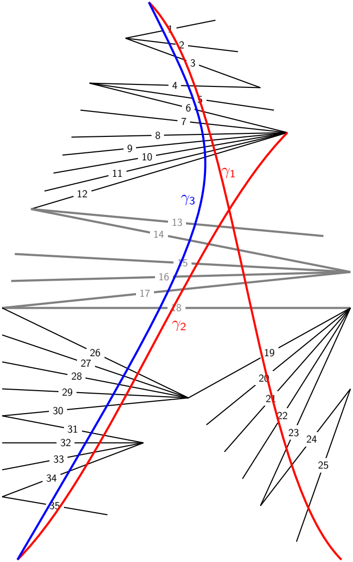
Surface Example



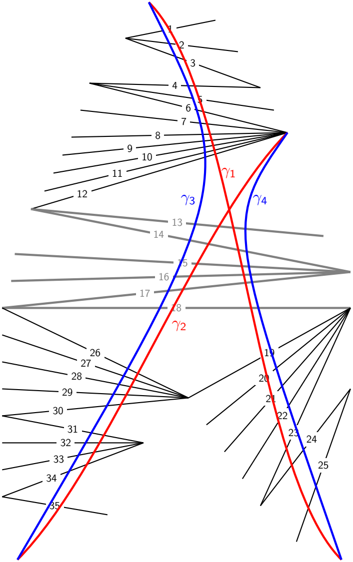
Surface Example



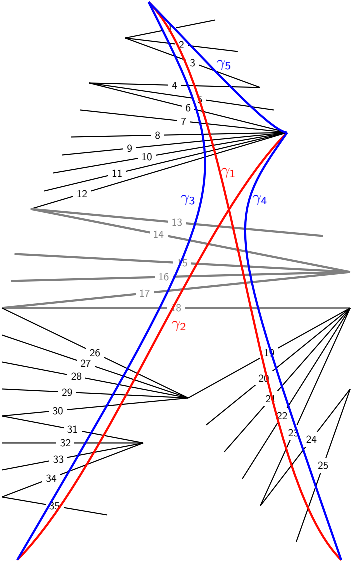
Surface Example



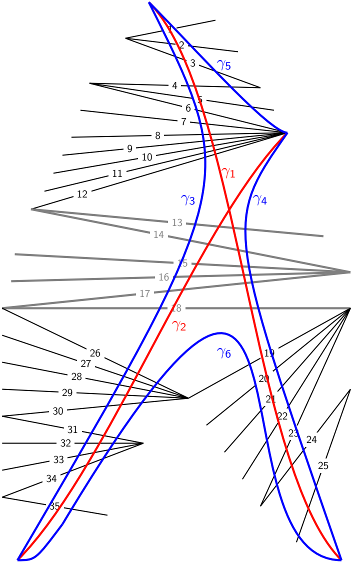
Surface Example



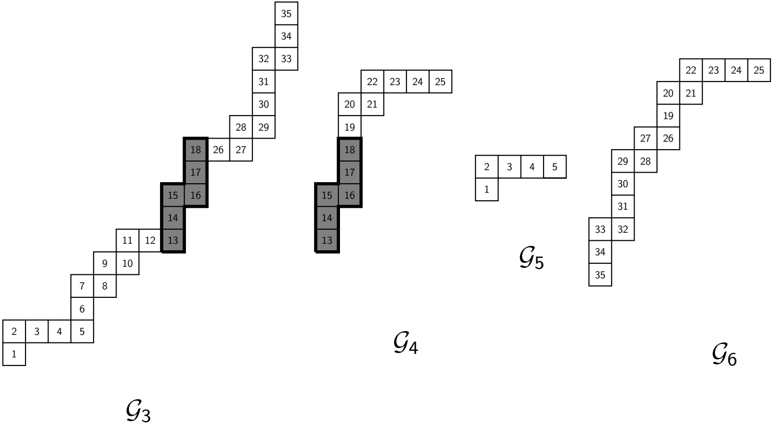
Surface Example



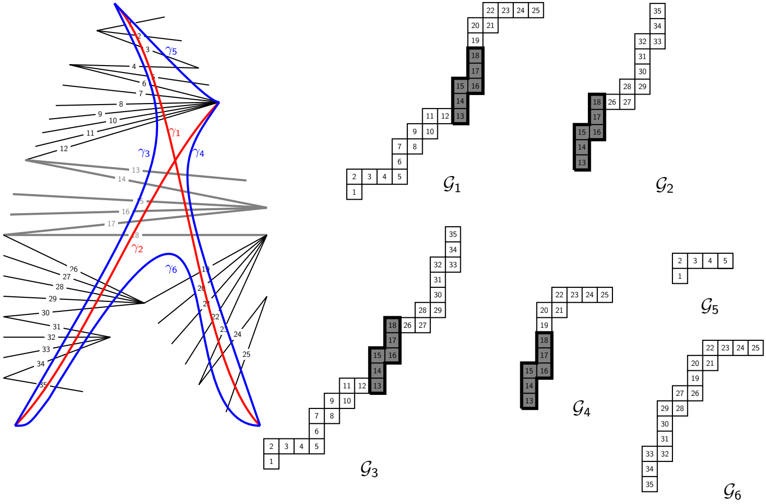
Surface Example



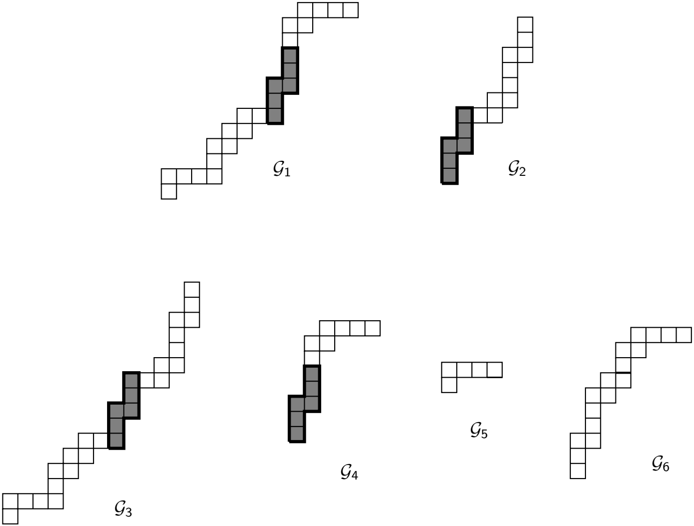
Surface Example



Surface Example



Surface Example



Relation to Cluster Algebras

Let γ_1 and γ_2 be two arcs and $\mathcal{G}_1$ and $\mathcal{G}_2$ their corresponding snake graphs.

Theorem (CS)

γ_1 and γ_2 **cross** *if and only if $\mathcal{G}_1$ and $\mathcal{G}_2$* **cross** *as snake graphs.*

Theorem (CS)

If γ_1 and γ_2 cross, then the snake graphs of the four arcs obtained by **smoothing the crossing** *are given by the* **resolution** $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$ *of the crossing of the snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ at the overlap $\mathcal{G}$.*

Remark

We do not assume that γ_1 and γ_2 cross only once. If the arcs cross multiple times the theorem can be used to resolve any of the crossings.

Relation to Cluster Algebras

Let γ_1 and γ_2 be two arcs and $\mathcal{G}_1$ and $\mathcal{G}_2$ their corresponding snake graphs.

Theorem (CS)

γ_1 and γ_2 **cross** *if and only if $\mathcal{G}_1$ and $\mathcal{G}_2$* **cross** *as snake graphs.*

Theorem (CS)

If γ_1 and γ_2 cross, then the snake graphs of the four arcs obtained by **smoothing the crossing** *are given by the* **resolution** $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$ *of the crossing of the snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ at the overlap $\mathcal{G}$.*

Remark

We do not assume that γ_1 and γ_2 cross only once. If the arcs cross multiple times the theorem can be used to resolve any of the crossings.

Relation to Cluster Algebras

Let γ_1 and γ_2 be two arcs and $\mathcal{G}_1$ and $\mathcal{G}_2$ their corresponding snake graphs.

Theorem (CS)

γ_1 and γ_2 **cross** *if and only if $\mathcal{G}_1$ and $\mathcal{G}_2$* **cross** *as snake graphs.*

Theorem (CS)

If γ_1 and γ_2 cross, then the snake graphs of the four arcs obtained by **smoothing the crossing** *are given by the* **resolution** $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$ *of the crossing of the snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ at the overlap $\mathcal{G}$.*

Remark

We do not assume that γ_1 and γ_2 cross only once. If the arcs cross multiple times the theorem can be used to resolve any of the crossings.

Relation to Cluster Algebras

Let γ_1 and γ_2 be two arcs and $\mathcal{G}_1$ and $\mathcal{G}_2$ their corresponding snake graphs.

Theorem (CS)

γ_1 and γ_2 **cross** *if and only if $\mathcal{G}_1$ and $\mathcal{G}_2$* **cross** *as snake graphs.*

Theorem (CS)

If γ_1 and γ_2 cross, then the snake graphs of the four arcs obtained by **smoothing the crossing** *are given by the* **resolution** $\mathrm{Res}_{\mathcal{G}}(\mathcal{G}_1, \mathcal{G}_2)$ *of the crossing of the snake graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ at the overlap $\mathcal{G}$.*

Remark

We do not assume that γ_1 and γ_2 cross only once. If the arcs cross multiple times the theorem can be used to resolve any of the crossings.

Skein Relations

As a corollary we obtain a new proof of the skein relations [MW].

Corollary (CS)

Let γ_1 and γ_2 be two arcs which cross and let (γ_3, γ_4) and (γ_5, γ_6) be the two pairs of arcs obtained by smoothing the crossing. Then

$$x_{\gamma_1} x_{\gamma_2} = x_{\gamma_3} x_{\gamma_4} + y(\tilde{\mathcal{G}}) x_{\gamma_5} x_{\gamma_6}$$

where $\tilde{\mathcal{G}} = (\mathcal{G}_3 \cup \mathcal{G}_4) \setminus (\mathcal{G}_5 \cup \mathcal{G}_6)$ *and* $y(\tilde{\mathcal{G}}) = \prod_{G_i \text{ a tile in } \tilde{\mathcal{G}}} y_i.$

Remark

- Note that Musiker and Williams in [MW] use **hyperbolic geometry** to prove the skein relations.
- Our proof is purely **combinatorial**. The key ingredient to our proof is Theorem 17 where we show the bijection between the perfect matchings.

Skein Relations

As a corollary we obtain a new proof of the skein relations [MW].

Corollary (CS)

Let γ_1 and γ_2 be two arcs which cross and let (γ_3, γ_4) and (γ_5, γ_6) be the two pairs of arcs obtained by smoothing the crossing. Then

$$x_{\gamma_1} x_{\gamma_2} = x_{\gamma_3} x_{\gamma_4} + y(\tilde{\mathcal{G}}) x_{\gamma_5} x_{\gamma_6}$$

where $\tilde{\mathcal{G}} = (\mathcal{G}_3 \cup \mathcal{G}_4) \setminus (\mathcal{G}_5 \cup \mathcal{G}_6)$ *and* $y(\tilde{\mathcal{G}}) = \prod_{G_i \text{ a tile in } \tilde{\mathcal{G}}} y_i$.

Remark

- Note that Musiker and Williams in [MW] use **hyperbolic geometry** to prove the skein relations.
- Our proof is **purely combinatorial**. The key ingredient to our proof is Theorem 17 where we show the bijection between the perfect matchings.

Skein Relations

As a corollary we obtain a new proof of the skein relations [MW].

Corollary (CS)

Let γ_1 and γ_2 be two arcs which cross and let (γ_3, γ_4) and (γ_5, γ_6) be the two pairs of arcs obtained by smoothing the crossing. Then

$$x_{\gamma_1} x_{\gamma_2} = x_{\gamma_3} x_{\gamma_4} + y(\tilde{\mathcal{G}}) x_{\gamma_5} x_{\gamma_6}$$

where $\tilde{\mathcal{G}} = (\mathcal{G}_3 \cup \mathcal{G}_4) \setminus (\mathcal{G}_5 \cup \mathcal{G}_6)$ *and* $y(\tilde{\mathcal{G}}) = \prod_{G_i \text{ a tile in } \tilde{\mathcal{G}}} y_i.$

Remark

- Note that Musiker and Williams in [MW] use **hyperbolic geometry** to prove the skein relations.
- Our proof is **purely combinatorial**. The key ingredient to our proof is Theorem 17 where we show the bijection between the perfect matchings.

Skein Relations

As a corollary we obtain a new proof of the skein relations [MW].

Corollary (CS)

Let γ_1 and γ_2 be two arcs which cross and let (γ_3, γ_4) and (γ_5, γ_6) be the two pairs of arcs obtained by smoothing the crossing. Then

$$x_{\gamma_1} x_{\gamma_2} = x_{\gamma_3} x_{\gamma_4} + y(\tilde{\mathcal{G}}) x_{\gamma_5} x_{\gamma_6}$$

where $\tilde{\mathcal{G}} = (\mathcal{G}_3 \cup \mathcal{G}_4) \backslash (\mathcal{G}_5 \cup \mathcal{G}_6)$ *and* $y(\tilde{\mathcal{G}}) = \prod_{G_i \text{ a tile in } \tilde{\mathcal{G}}} y_i.$

Remark

- Note that Musiker and Williams in [MW] use **hyperbolic geometry** to prove the skein relations.
- Our proof is **purely combinatorial**. The key ingredient to our proof is Theorem 17 where we show the bijection between the perfect matchings.

Skein Relations

As a corollary we obtain a new proof of the skein relations [MW].

Corollary (CS)

Let γ_1 and γ_2 be two arcs which cross and let (γ_3, γ_4) and (γ_5, γ_6) be the two pairs of arcs obtained by smoothing the crossing. Then

$$x_{\gamma_1} x_{\gamma_2} = x_{\gamma_3} x_{\gamma_4} + y(\tilde{\mathcal{G}}) x_{\gamma_5} x_{\gamma_6}$$

where $\tilde{\mathcal{G}} = (\mathcal{G}_3 \cup \mathcal{G}_4) \setminus (\mathcal{G}_5 \cup \mathcal{G}_6)$ *and* $y(\tilde{\mathcal{G}}) = \prod_{G_i \text{ a tile in } \tilde{\mathcal{G}}} y_i.$

Remark

- Note that Musiker and Williams in [MW] use **hyperbolic geometry** to prove the skein relations.
- Our proof is **purely combinatorial**. The key ingredient to our proof is Theorem 17 where we show the bijection between the perfect matchings.

Self-crossing snake graphs and band graphs

- Self-crossing arcs and closed loops appear naturally in the process of smoothing crossings. Consider the following example.

Example

In this example we resolve two crossings of the following arcs.

Example (Band graph)

Self-crossing snake graphs and band graphs

- Self-crossing arcs and closed loops appear naturally in the process of smoothing crossings. Consider the following example.

Example

In this example we resolve two crossings of the following arcs.

Example (Band graph)

Self-crossing snake graphs and band graphs

- Self-crossing arcs and closed loops appear naturally in the process of smoothing crossings. Consider the following example.

Example

In this example we resolve two crossings of the following arcs.

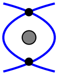
Example (Band graph)

Self-crossing snake graphs and band graphs

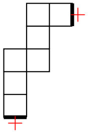
- Self-crossing arcs and closed loops appear naturally in the process of smoothing crossings. Consider the following example.

Example

In this example we resolve two crossings of the following arcs.



Example (Band graph)

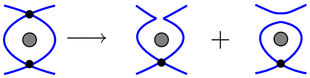


Self-crossing snake graphs and band graphs

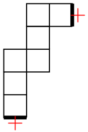
- Self-crossing arcs and closed loops appear naturally in the process of smoothing crossings. Consider the following example.

Example

In this example we resolve two crossings of the following arcs.



Example (Band graph)

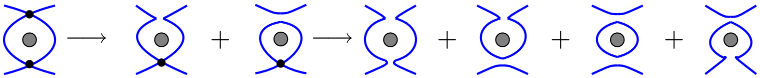


Self-crossing snake graphs and band graphs

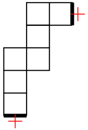
- Self-crossing arcs and closed loops appear naturally in the process of smoothing crossings. Consider the following example.

Example

In this example we resolve two crossings of the following arcs.



Example (Band graph)



Self-crossing snake graphs

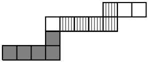
- Similar to the definition of a local overlap for two snake graphs, we define the notion of **self-overlap** for abstract snake graphs. Here we have two subcases.
  - Self-overlap in the same direction
    - without intersection

Self-crossing snake graphs

- Similar to the definition of a local overlap for two snake graphs, we define the notion of **self-overlap** for abstract snake graphs. Here we have two subcases.
  - Self-overlap in the same direction
    - without intersection
    - with intersection
  - Self-overlap in the opposite direction
- We then define what it means for a snake graph to self-cross in a self-overlap.
- We give the resolution of a self-crossing snake graph which consists of two snake graphs and a band graph.
- Finally, we show a bijection between perfect matchings of a self-crossing snake graph with perfect matchings of its resolution.

Self-crossing snake graphs

- Similar to the definition of a local overlap for two snake graphs, we define the notion of **self-overlap** for abstract snake graphs. Here we have two subcases
  - Self-overlap in the same direction
    - without intersection

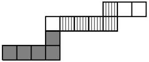


    - with intersection

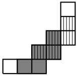
  - Self-overlap in the opposite direction
- We then define what it means for a snake graph to self-cross in a self-overlap.
- We give the resolution of a self-crossing snake graph which consists of two snake graphs and a band graph.
- Finally, we show a bijection between perfect matchings of a self-crossing snake graph with perfect matchings of its resolution.

Self-crossing snake graphs

- Similar to the definition of a local overlap for two snake graphs, we define the notion of **self-overlap** for abstract snake graphs. Here we have two subcases.
  - Self-overlap in the same direction
    - without intersection



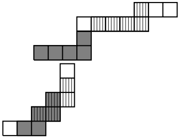
    - with intersection



  - Self-overlap in the opposite direction

Self-crossing snake graphs

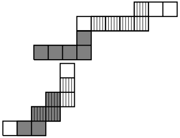
- Similar to the definition of a local overlap for two snake graphs, we define the notion of **self-overlap** for abstract snake graphs. Here we have two subcases.
  - Self-overlap in the same direction
    - without intersection
    - with intersection



  - Self-overlap in the opposite direction
- We then define what it means for a snake graph to **self-cross** in a self-overlap.
- We give the **resolution** of a self-crossing snake graph which consists of **two snake graphs** and a **band graph**.
- Finally, we show a **bijection** between perfect matchings of a self-crossing snake graph with perfect matchings of its resolution.

Self-crossing snake graphs

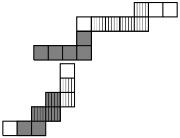
- Similar to the definition of a local overlap for two snake graphs, we define the notion of **self-overlap** for abstract snake graphs. Here we have two subcases.
  - Self-overlap in the same direction
    - without intersection
    - with intersection



  - Self-overlap in the opposite direction
- We then define what it means for a snake graph to **self-cross** in a self-overlap.
- We give the **resolution** of a self-crossing snake graph which consists of **two snake graphs** and a **band graph**.
- Finally, we show a **bijection** between perfect matchings of a self-crossing snake graph with perfect matchings of its resolution.

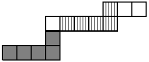
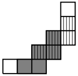
Self-crossing snake graphs

- Similar to the definition of a local overlap for two snake graphs, we define the notion of **self-overlap** for abstract snake graphs. Here we have two subcases.
  - Self-overlap in the same direction
    - without intersection
    - with intersection



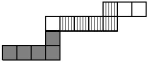
  - Self-overlap in the opposite direction
- We then define what it means for a snake graph to **self-cross** in a self-overlap.
- We give the **resolution** of a self-crossing snake graph which consists of **two snake graphs** and a **band graph**.
- Finally, we show a **bijection** between perfect matchings of a self-crossing snake graph with perfect matchings of its resolution.

Self-crossing snake graphs

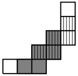
- Similar to the definition of a local overlap for two snake graphs, we define the notion of **self-overlap** for abstract snake graphs. Here we have two subcases.
  - Self-overlap in the same direction
    - without intersection 
    - with intersection 
  - Self-overlap in the opposite direction
- We then define what it means for a snake graph to **self-cross** in a self-overlap.
- We give the **resolution** of a self-crossing snake graph which consists of **two snake graphs** and a **band graph**.
- Finally, we show a **bijection** between perfect matchings of a self-crossing snake graph with perfect matchings of its resolution.

Self-crossing snake graphs

- Similar to the definition of a local overlap for two snake graphs, we define the notion of **self-overlap** for abstract snake graphs. Here we have two subcases.
  - Self-overlap in the same direction
    - without intersection

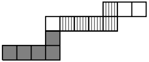
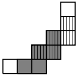


    - with intersection

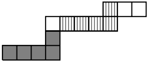
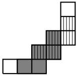


  - Self-overlap in the opposite direction
- We then define what it means for a snake graph to **self-cross** in a self-overlap.
- We give the **resolution** of a self-crossing snake graph which consists of **two snake graphs** and a **band graph**.
- Finally, we show a **bijection** between perfect matchings of a self-crossing snake graph with perfect matchings of its resolution.

Self-crossing snake graphs

- Similar to the definition of a local overlap for two snake graphs, we define the notion of **self-overlap** for abstract snake graphs. Here we have two subcases.
  - Self-overlap in the same direction
    - without intersection 
    - with intersection 
  - Self-overlap in the opposite direction
- We then define what it means for a snake graph to **self-cross** in a self-overlap.
- We give the **resolution** of a self-crossing snake graph which consists of **two snake graphs** and a **band graph**.
- Finally, we show a **bijection** between perfect matchings of a self-crossing snake graph with perfect matchings of its resolution.

Self-crossing snake graphs

- Similar to the definition of a local overlap for two snake graphs, we define the notion of **self-overlap** for abstract snake graphs. Here we have two subcases.
  - Self-overlap in the same direction
    - without intersection 
    - with intersection 
  - Self-overlap in the opposite direction
- We then define what it means for a snake graph to **self-cross** in a self-overlap.
- We give the **resolution** of a self-crossing snake graph which consists of **two snake graphs** and a **band graph**.
- Finally, we show a **bijection** between perfect matchings of a self-crossing snake graph with perfect matchings of its resolution.

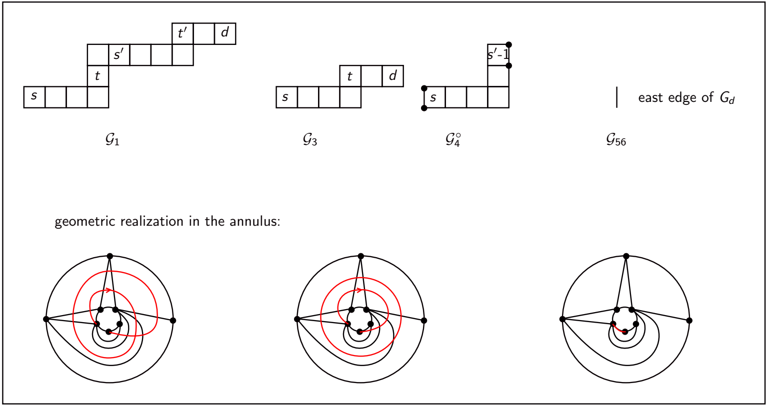
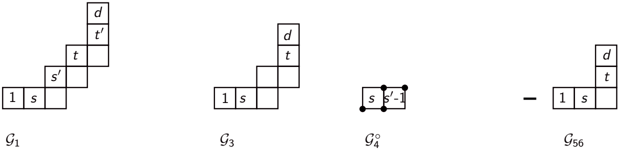


Figure: Example of resolution of selfcrossing when $s' < t$ and $s = 1$ together with geometric realization on the annulus. Here the snake graph G_{56} is a single edge and the corresponding arc in the surface is a boundary segment.



geometric realization in the punctured disk:

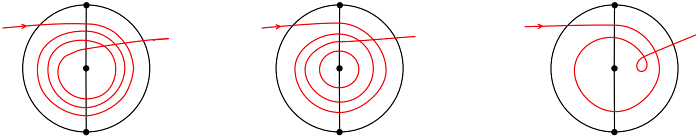


Figure: Example of resolution of selfcrossing when $s' < t$ together with geometric realization on the punctured disk

Dreaded torus

Definition (Upper cluster algebra)

$$\mathcal{U} = \bigcap_{\mathbf{x} \text{ seed}} \mathbb{Z}[\mathbf{x}].$$

Theorem (C, Kyungyong Lee, S)

Let $\mathcal{A}$ be the cluster algebra associated to the dreaded torus and $\mathcal{U}$ be its upper cluster algebra. Then $\mathcal{A} = \mathcal{U}$.

Sketch of proof. By [MM], it suffices to show that three particular Laurent polynomials given by the band graphs of three loops X, Y, Z belong to the cluster algebra.

Dreaded torus

Definition (Upper cluster algebra)

$$\mathcal{U} = \bigcap_{\mathbf{x} \text{ seed}} \mathbb{Z}[\mathbf{x}].$$

Theorem (C, Kyungyong Lee, S)

Let $\mathcal{A}$ be the cluster algebra associated to the dreaded torus and $\mathcal{U}$ be its upper cluster algebra. Then $\mathcal{A} = \mathcal{U}$.

Sketch of proof. By [MM], it suffices to show that three particular Laurent polynomials given by the band graphs of three loops X, Y, Z belong to the cluster algebra.

$X =$

Dreaded torus

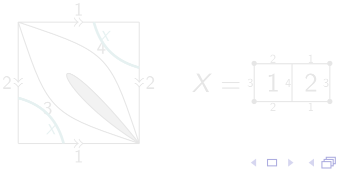
Definition (Upper cluster algebra)

$$\mathcal{U} = \bigcap_{\mathbf{x} \text{ seed}} \mathbb{Z}[\mathbf{x}].$$

Theorem (C, Kyungyong Lee, S)

Let $\mathcal{A}$ be the cluster algebra associated to the dreaded torus and $\mathcal{U}$ be its upper cluster algebra. Then $\mathcal{A} = \mathcal{U}$.

Sketch of proof. By [MM], it suffices to show that three particular Laurent polynomials given by the band graphs of three loops X, Y, Z belong to the cluster algebra.



Dreaded torus

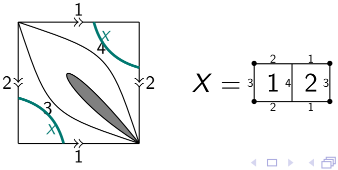
Definition (Upper cluster algebra)

$$\mathcal{U} = \bigcap_{\mathbf{x}\ \text{seed}} \mathbb{Z}[\mathbf{x}].$$

Theorem (C, Kyungyong Lee, S)

Let $\mathcal{A}$ be the cluster algebra associated to the dreaded torus and $\mathcal{U}$ be its upper cluster algebra. Then $\mathcal{A} = \mathcal{U}$.

Sketch of proof. By [MM], it suffices to show that three particular Laurent polynomials given by the band graphs of three loops X, Y, Z belong to the cluster algebra.



Dreaded torus

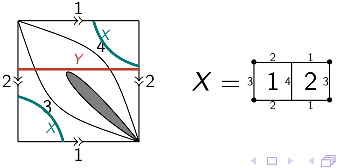
Definition (Upper cluster algebra)

$$\mathcal{U} = \bigcap_{\mathbf{x} \text{ seed}} \mathbb{Z}[\mathbf{x}].$$

Theorem (C, Kyungyong Lee, S)

Let $\mathcal{A}$ be the cluster algebra associated to the dreaded torus and $\mathcal{U}$ be its upper cluster algebra. Then $\mathcal{A} = \mathcal{U}$.

Sketch of proof. By [MM], it suffices to show that three particular Laurent polynomials given by the band graphs of three loops X, Y, Z belong to the cluster algebra.



Dreaded torus

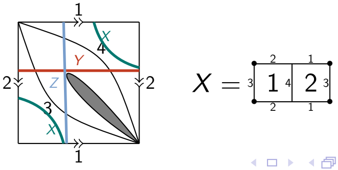
Definition (Upper cluster algebra)

$$\mathcal{U} = \bigcap_{\mathbf{x} \text{ seed}} \mathbb{Z}[\mathbf{x}].$$

Theorem (C, Kyungyong Lee, S)

Let $\mathcal{A}$ be the cluster algebra associated to the dreaded torus and $\mathcal{U}$ be its upper cluster algebra. Then $\mathcal{A} = \mathcal{U}$.

Sketch of proof. By [MM], it suffices to show that three particular Laurent polynomials given by the band graphs of three loops X, Y, Z belong to the cluster algebra.



Bibliography

I. Canakci, R. Schiffler *Snake graph calculus and cluster algebras from surfaces*, to appear in *Journal of Algebra*, **382**, 240-281.

S. Fomin, M. Shapiro and D. Thurston, Cluster algebras and triangulated surfaces. Part I: Cluster complexes, *Acta Math.* **201** (2008), 83-146.

S. Fomin and A. Zelevinsky, Cluster algebras I: Foundations, *J. Amer. Math. Soc.* **15** (2002), 497–529.

J. Matherne and G. Muller, Computing upper cluster algebras, preprint, arXiv:1307.0579.

G. Musiker, R. Schiffler and L. Williams, Positivity for cluster algebras from surfaces, *Adv. Math.* **227**, (2011), 2241–2308.

G. Musiker, R. Schiffler and L. Williams, Bases for cluster algebras from surfaces, to appear in *Compos. Math.*

G. Musiker and L. Williams, Matrix formulae and skein relations for cluster algebras from surfaces, preprint, arXiv:1108.3382.