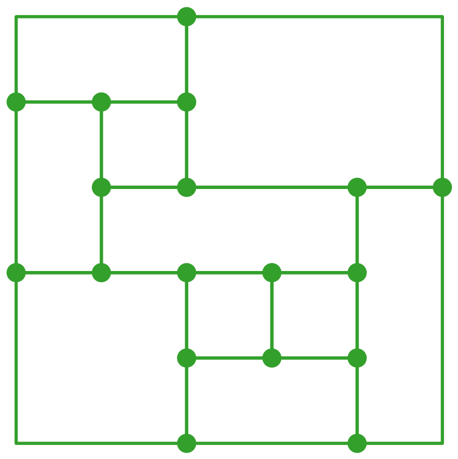
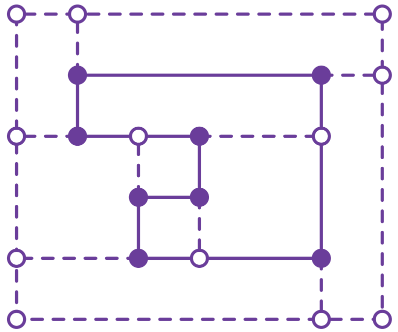
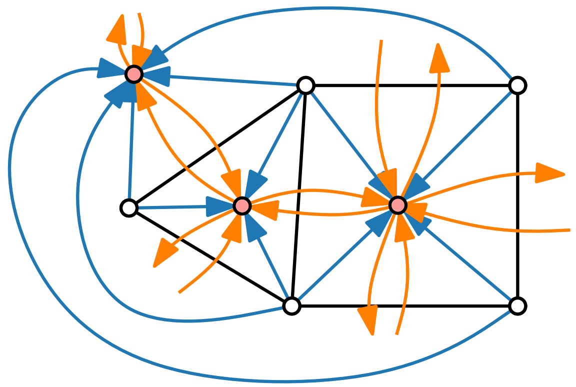
Visualization of Graphs



Lecture 6: Orthogonal Layouts

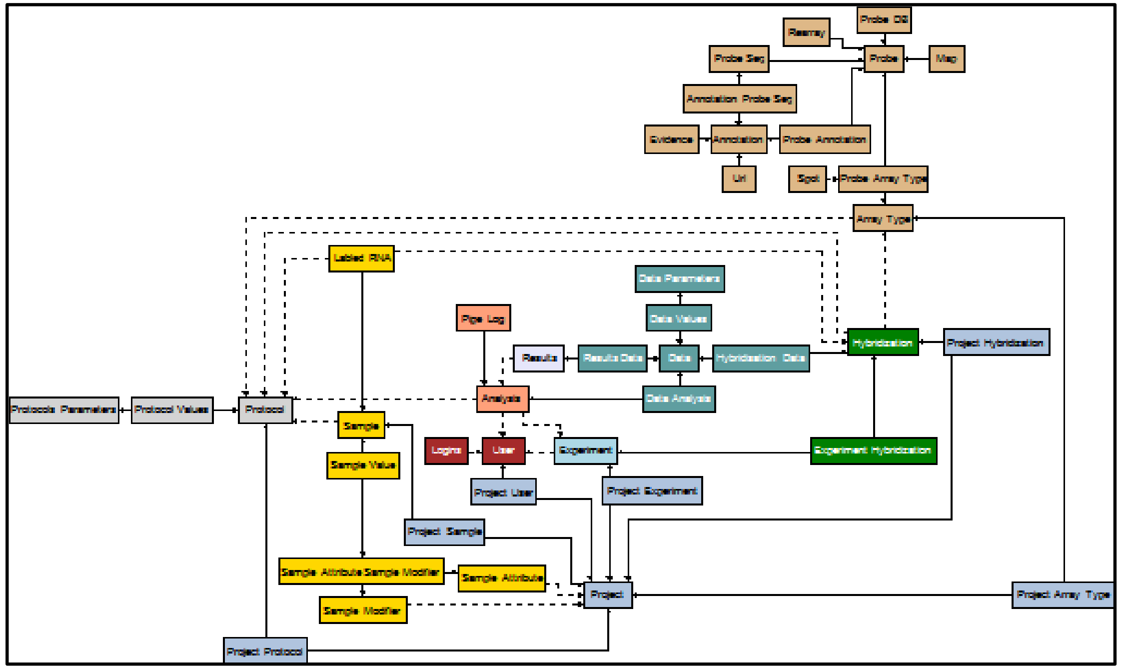




Johannes Zink

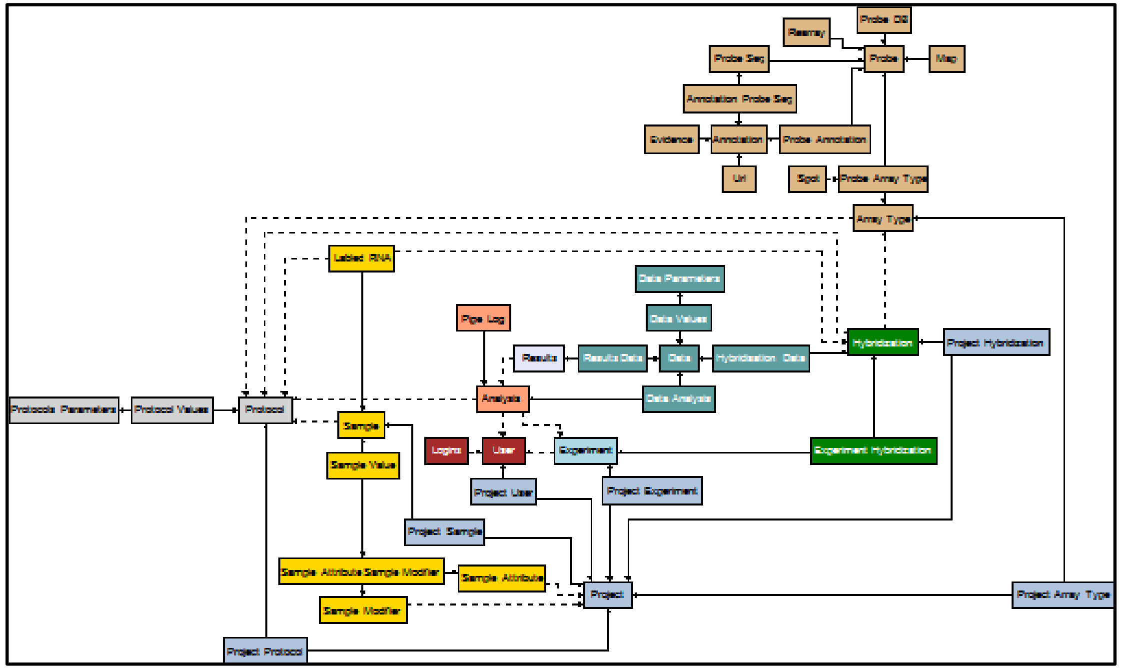
Summer semester 2024

Orthogonal Layout – Applications

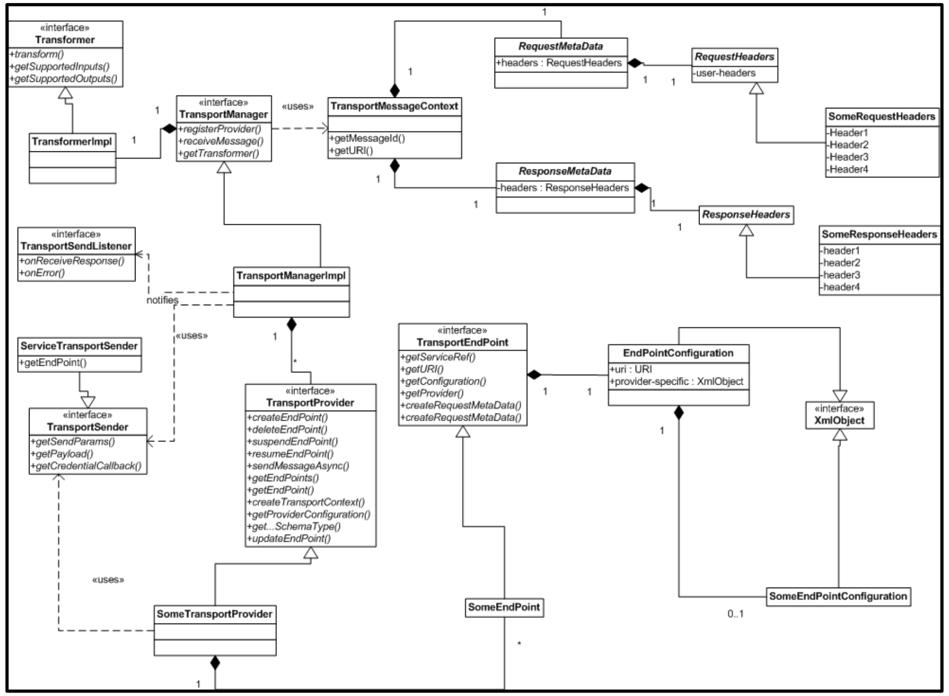


ER diagram in OGDF

Orthogonal Layout – Applications

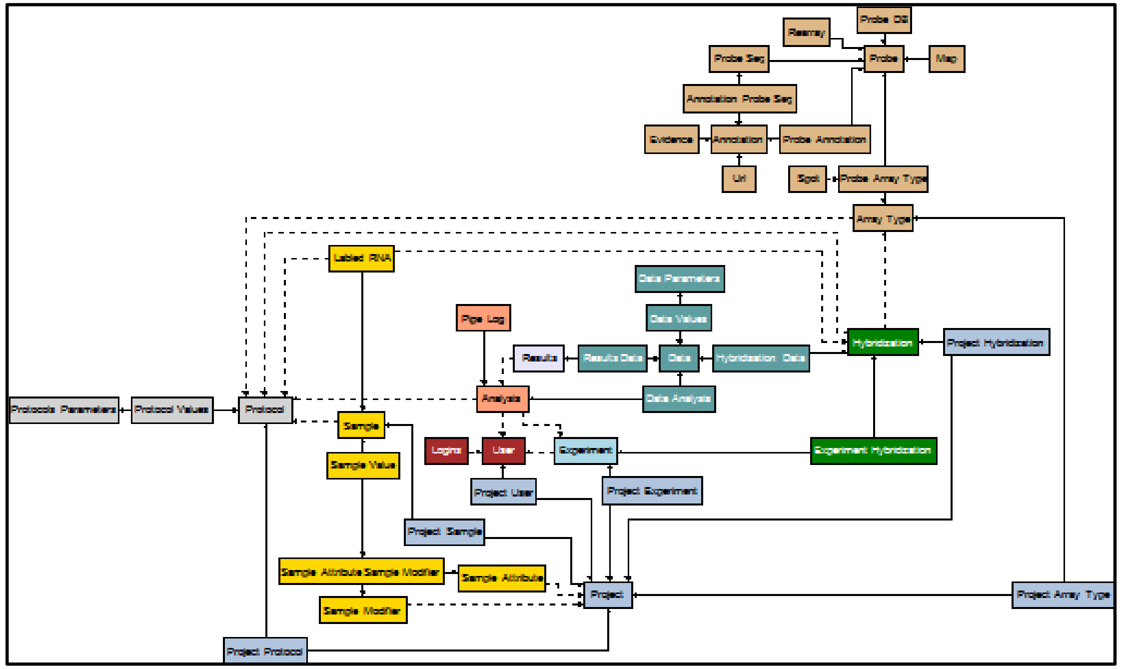


ER diagram in OGDF

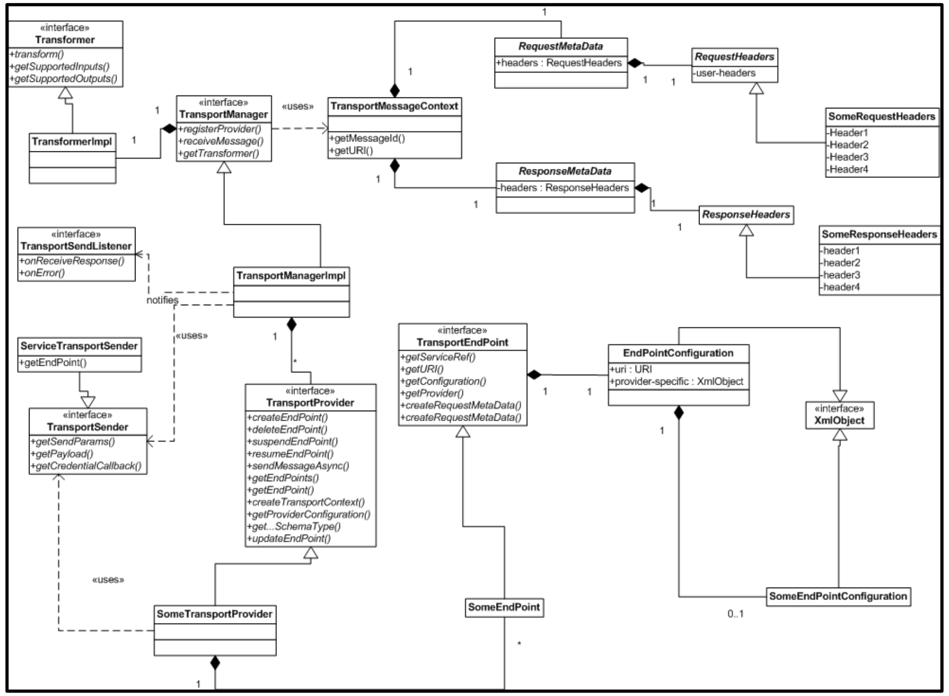


UML diagram by Oracle

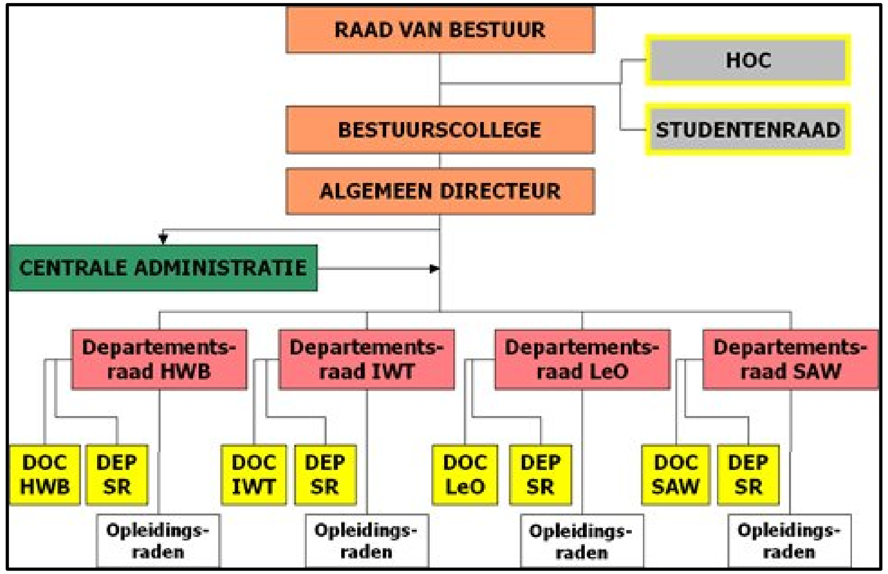
Orthogonal Layout – Applications



ER diagram in OGDF

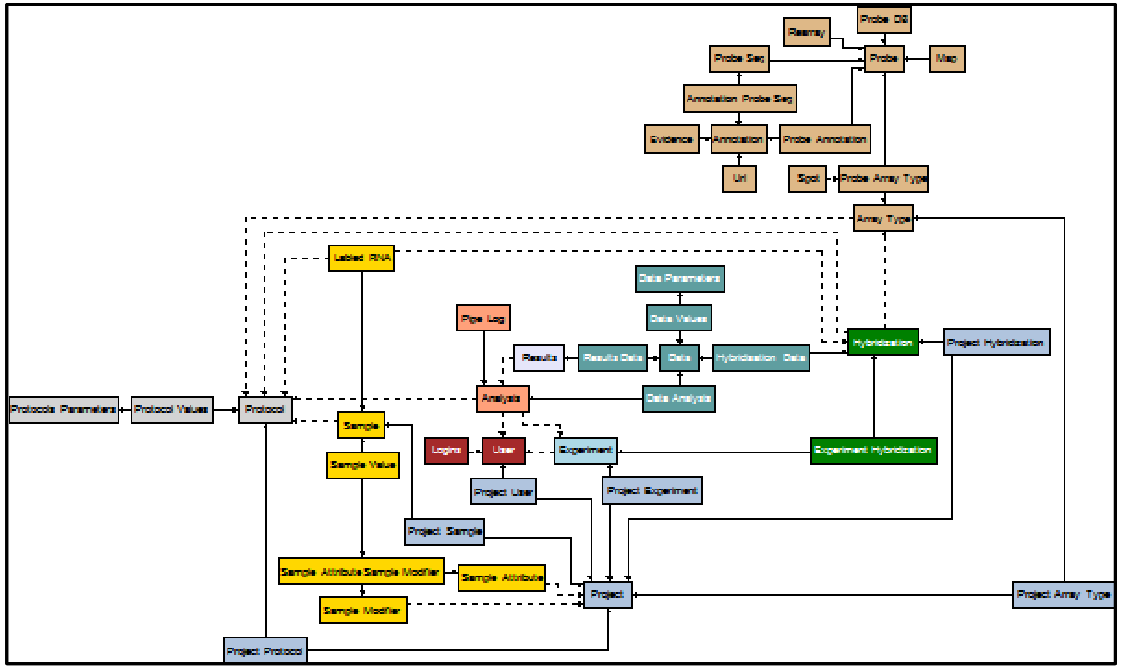


UML diagram by Oracle

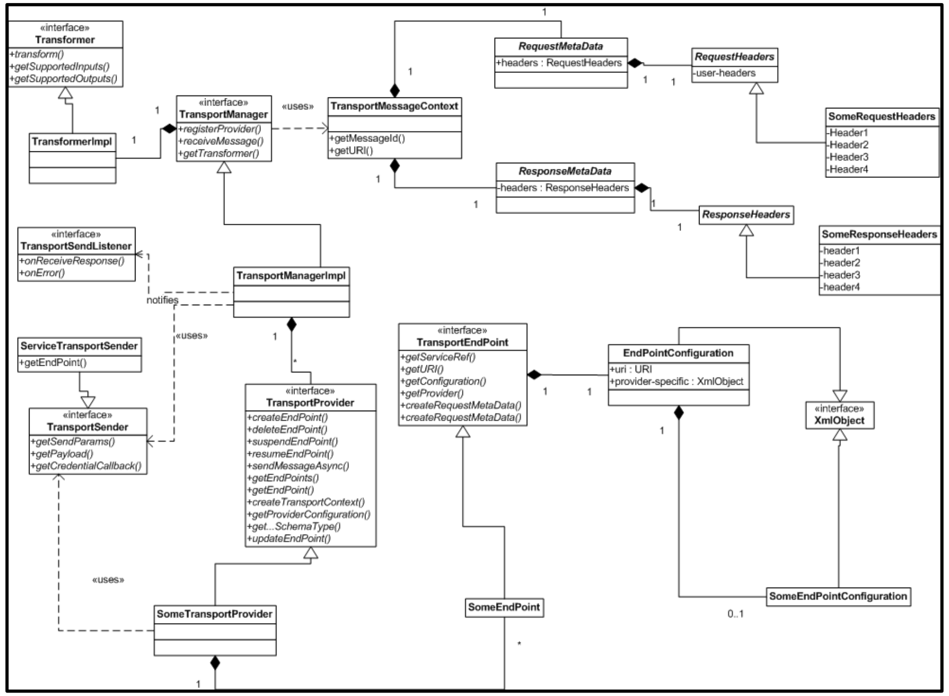


Organigram of HS Limburg

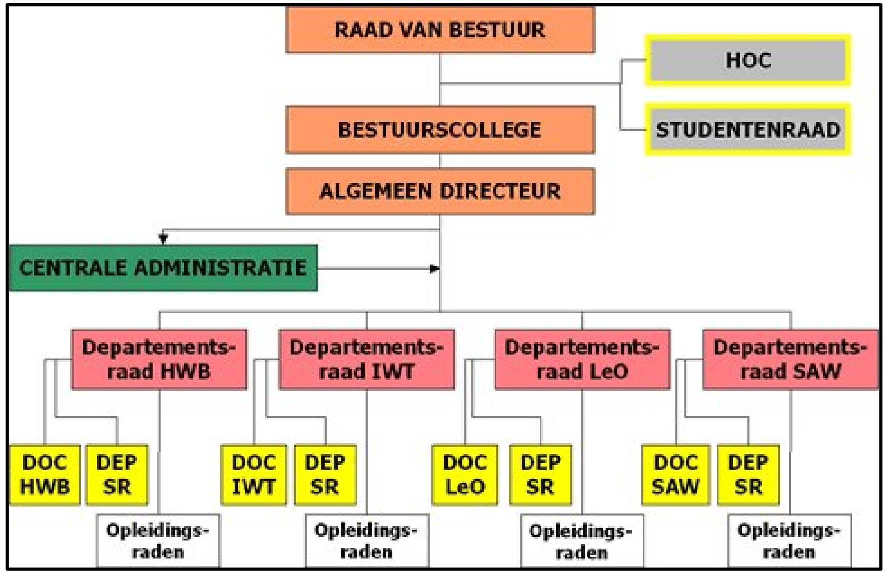
Orthogonal Layout – Applications



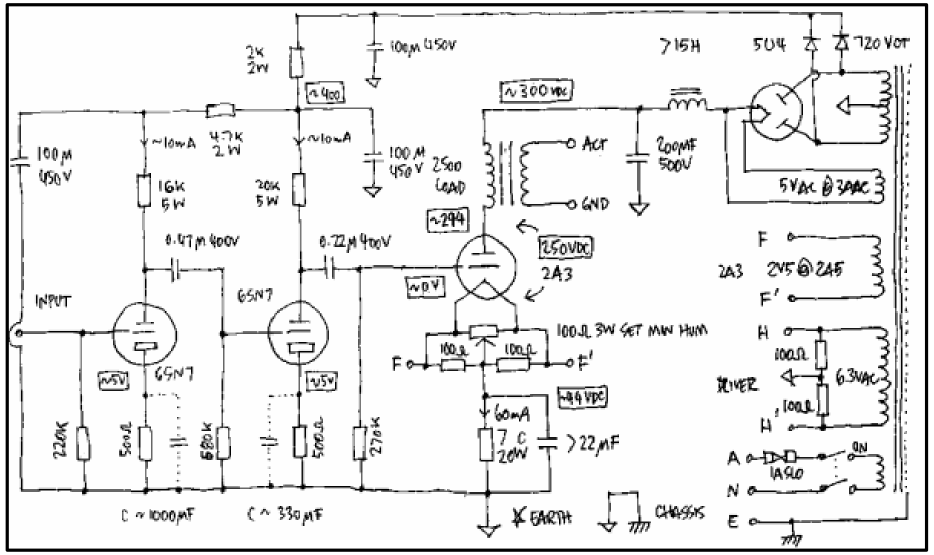
ER diagram in OGDF



UML diagram by Oracle



Organigram of HS Limburg

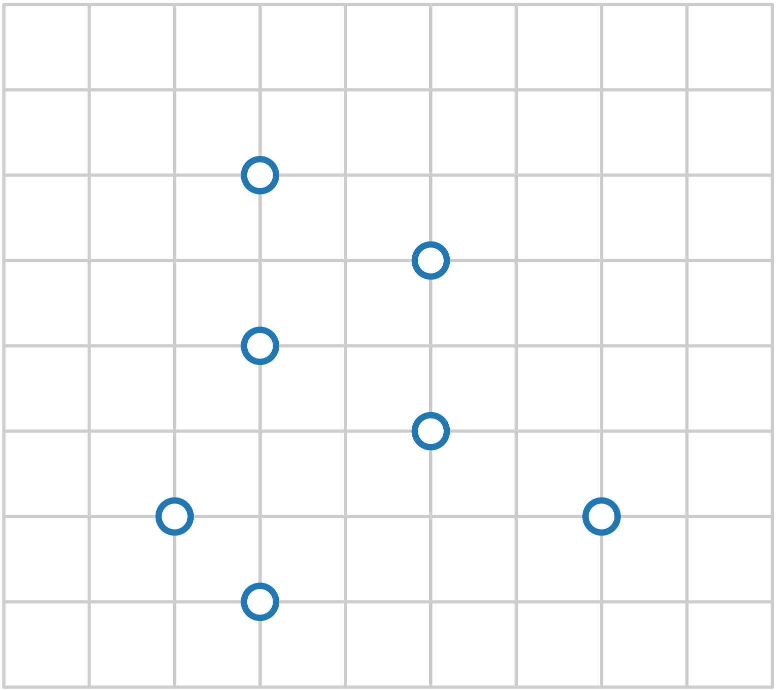


Circuit diagram by Jeff Atwood

Orthogonal Layout – Definition

Definition.
A drawing Γ of a graph G is called **orthogonal** if

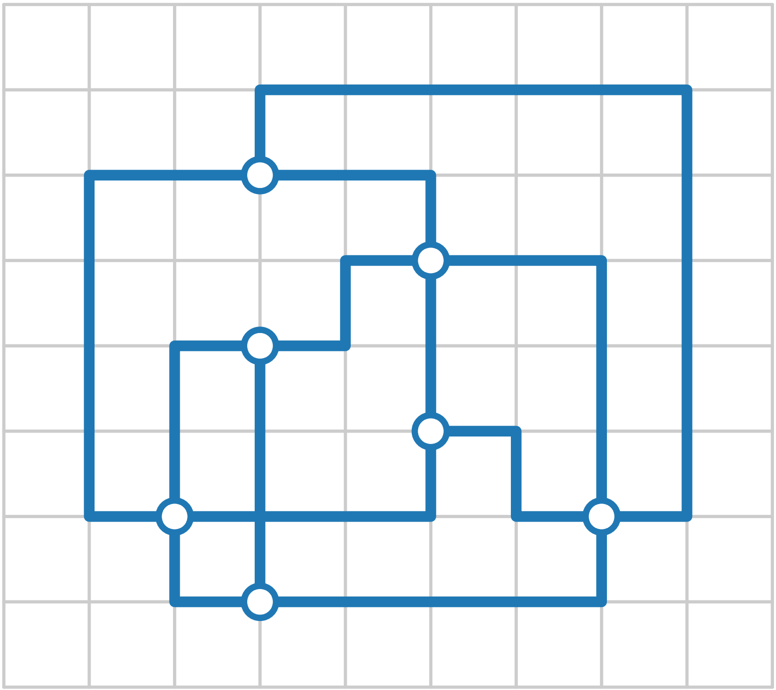
Orthogonal Layout – Definition



Definition.
A drawing Γ of a graph G is called **orthogonal** if

- vertices are drawn as points on a grid,

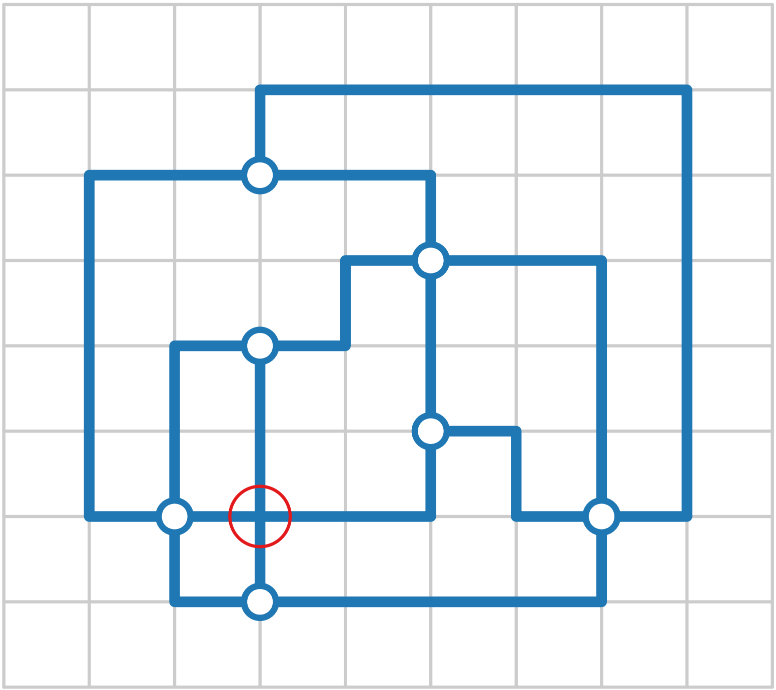
Orthogonal Layout – Definition



Definition.
A drawing Γ of a graph G is called **orthogonal** if

- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and

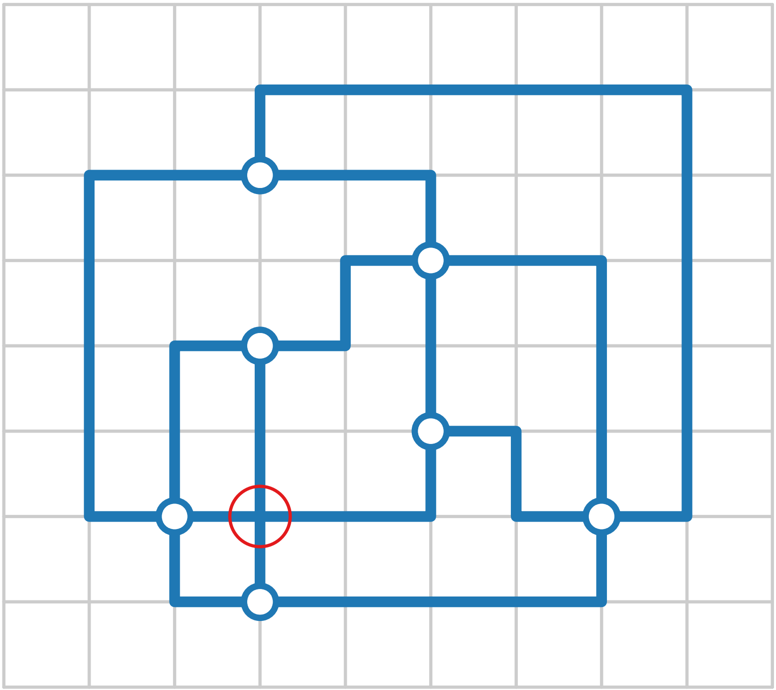
Orthogonal Layout – Definition



Definition.
A drawing Γ of a graph G is called **orthogonal** if

- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

Orthogonal Layout – Definition

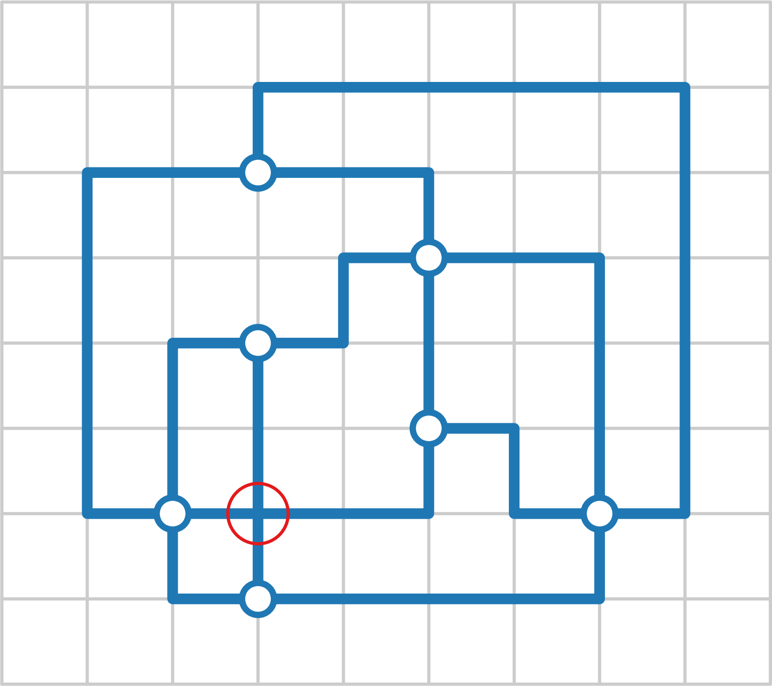


Definition.
A drawing Γ of a graph G is called **orthogonal** if

- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

Observations.

Orthogonal Layout – Definition



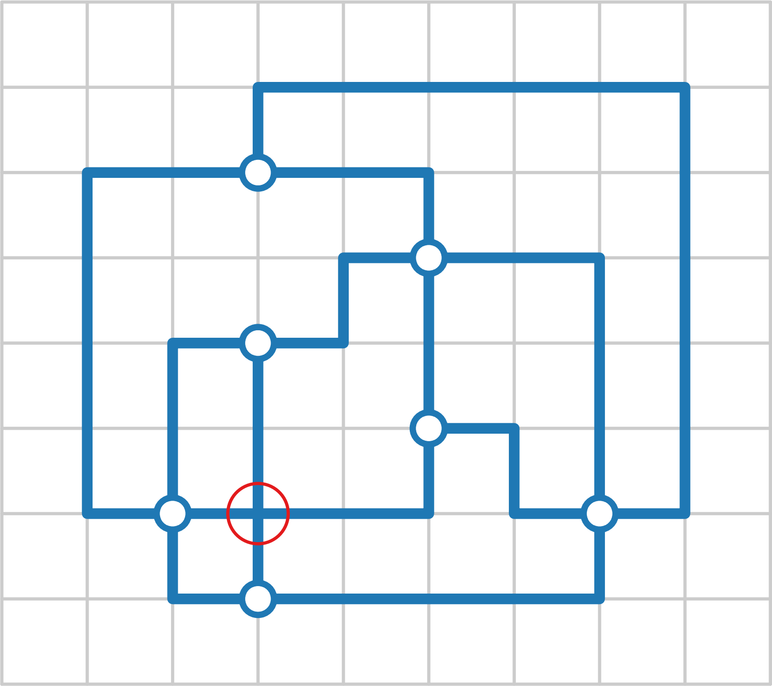
Definition.
A drawing Γ of a graph G is called **orthogonal** if

- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

Observations.

- Edges lie on a grid ⇒ **bends** lie on grid points

Orthogonal Layout – Definition



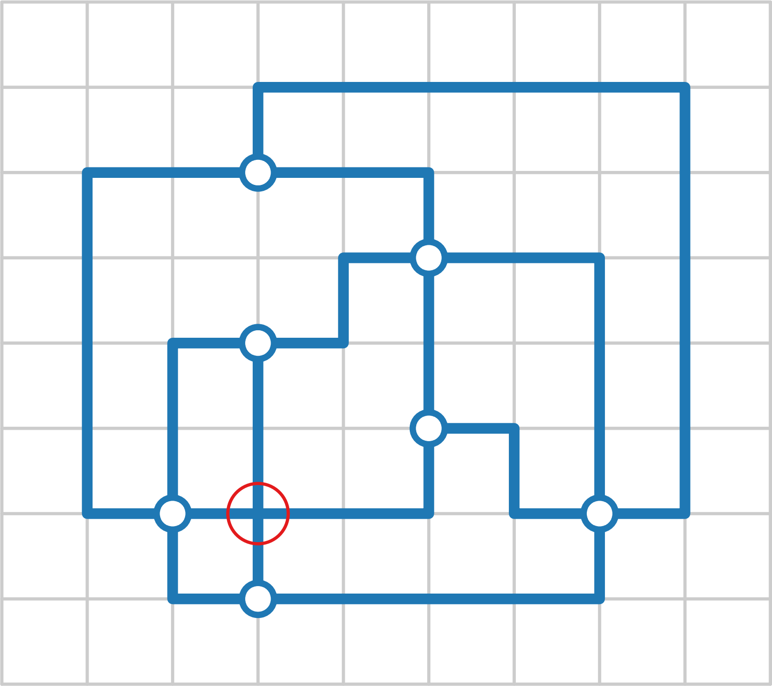
Definition.
A drawing Γ of a graph G is called **orthogonal** if

- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

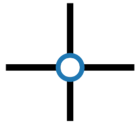
Observations.

- Edges lie on a grid ⇒ **bends** lie on grid points
- Max. degree of each vertex is at most 4

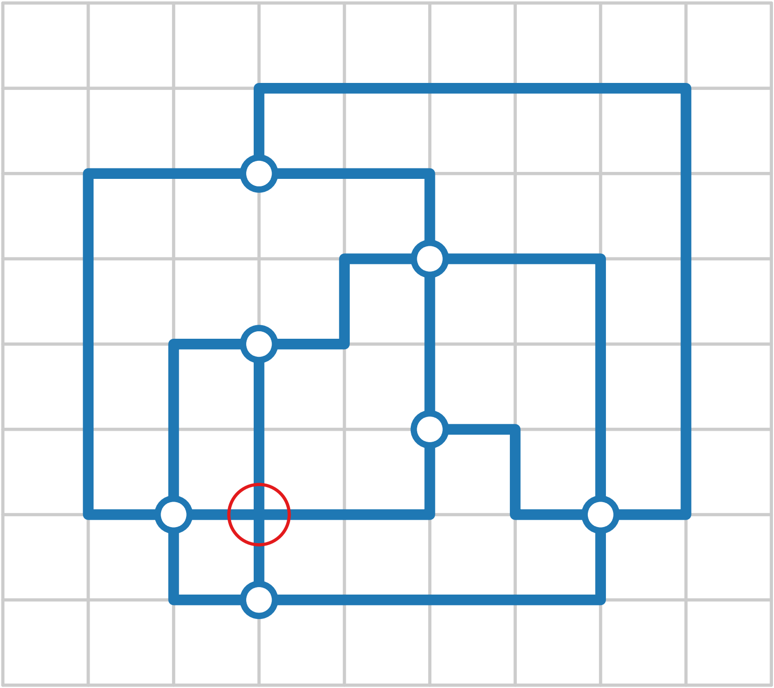
Orthogonal Layout – Definition



Definition.
A drawing Γ of a graph G is called **orthogonal** if
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

Observations.
- Edges lie on a grid $\Rightarrow$ **bends** lie on grid points
- Max. degree of each vertex is at most 4
- Otherwise 

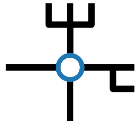
Orthogonal Layout – Definition



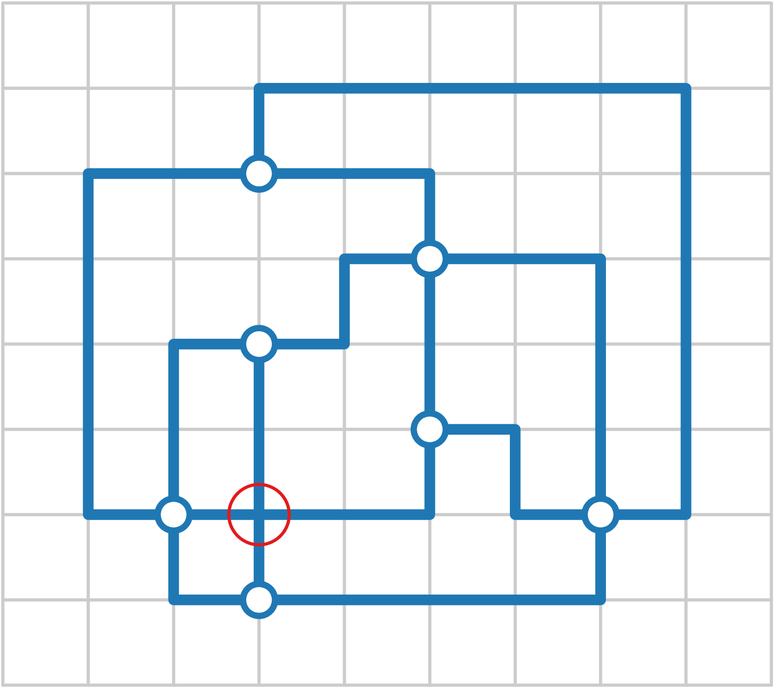
Definition.
A drawing Γ of a graph G is called **orthogonal** if

- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

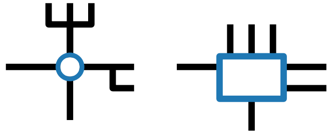
Observations.

- Edges lie on a grid $\Rightarrow$ **bends** lie on grid points
- Max. degree of each vertex is at most 4
- Otherwise 

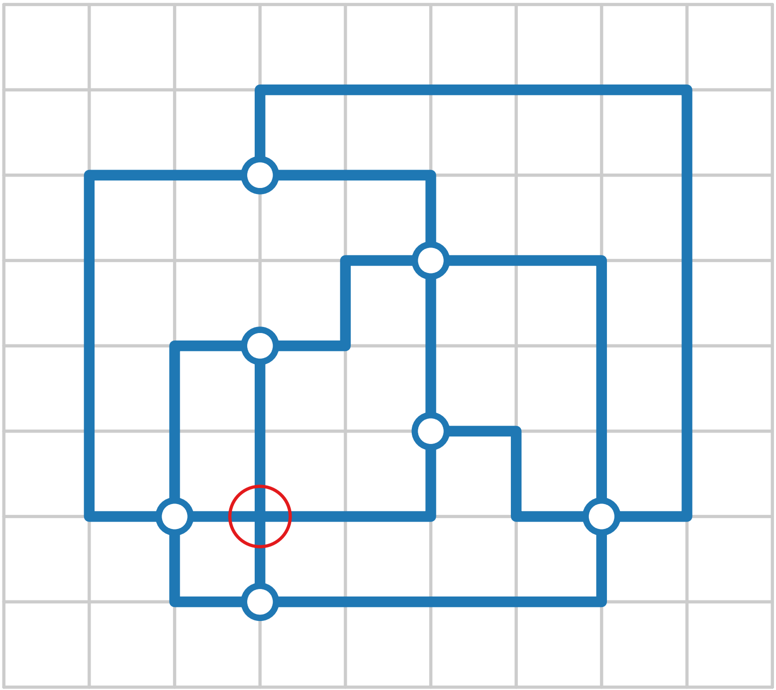
Orthogonal Layout – Definition



Definition.
A drawing Γ of a graph G is called **orthogonal** if
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

Observations.
- Edges lie on a grid ⇒ **bends** lie on grid points
- Max. degree of each vertex is at most 4
- Otherwise 

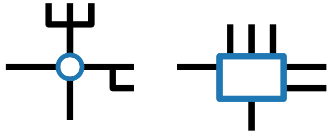
Orthogonal Layout – Definition



Definition.
A drawing Γ of a graph G is called **orthogonal** if
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

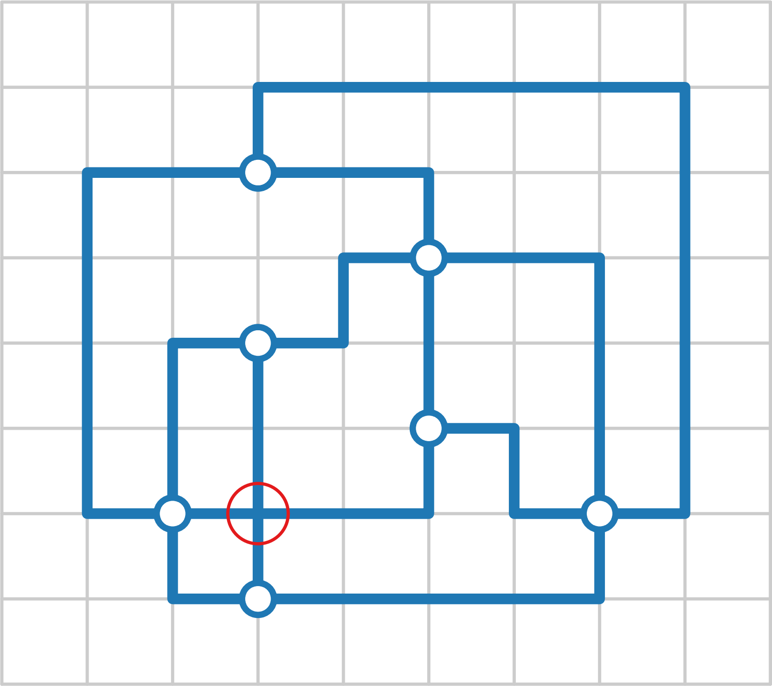
Observations.

- Edges lie on a grid ⇒ **bends** lie on grid points
- Max. degree of each vertex is at most 4
- Otherwise

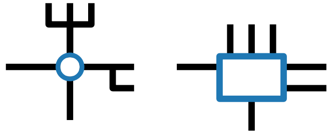


Planarization.

Orthogonal Layout – Definition

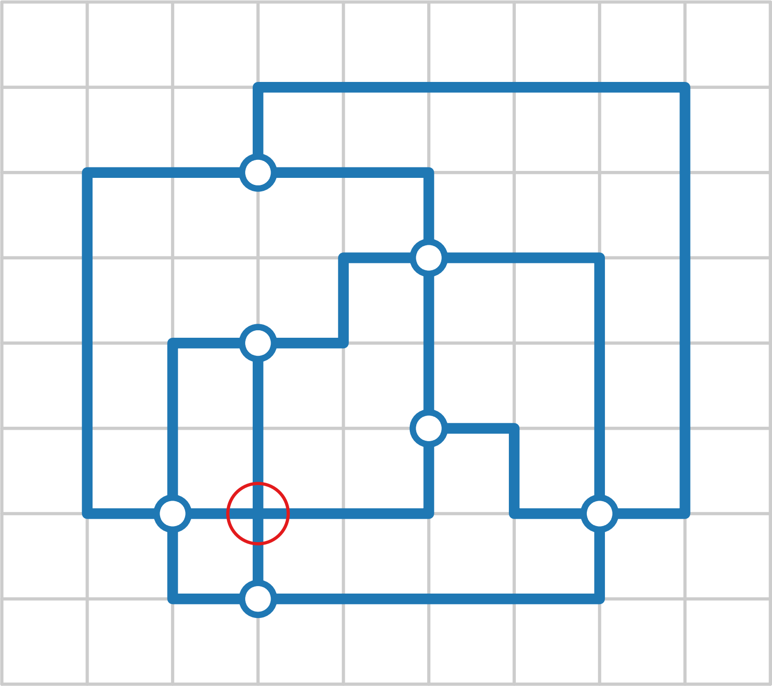


Definition.
A drawing Γ of a graph G is called **orthogonal** if
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

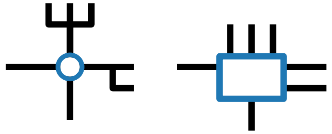
Observations.
- Edges lie on a grid ⇒ **bends** lie on grid points
- Max. degree of each vertex is at most 4
- Otherwise 

Planarization.
- Fix embedding

Orthogonal Layout – Definition

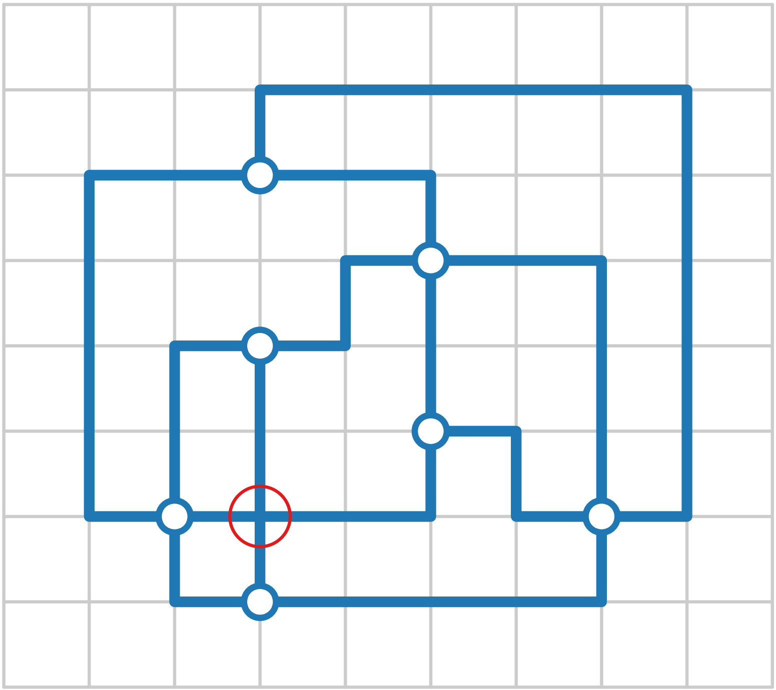


Definition.
A drawing Γ of a graph G is called **orthogonal** if
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

Observations.
- Edges lie on a grid ⇒ **bends** lie on grid points
- Max. degree of each vertex is at most 4
- Otherwise 

Planarization.
- Fix embedding
- Crossings become vertices

Orthogonal Layout – Definition

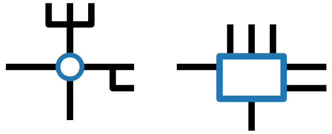


Definition.
A drawing Γ of a graph G is called **orthogonal** if

- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

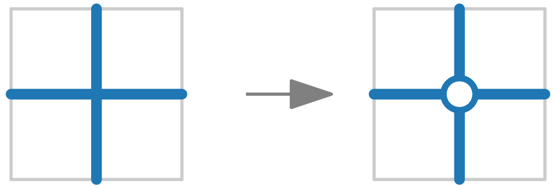
Observations.

- Edges lie on a grid ⇒ **bends** lie on grid points
- Max. degree of each vertex is at most 4
- Otherwise

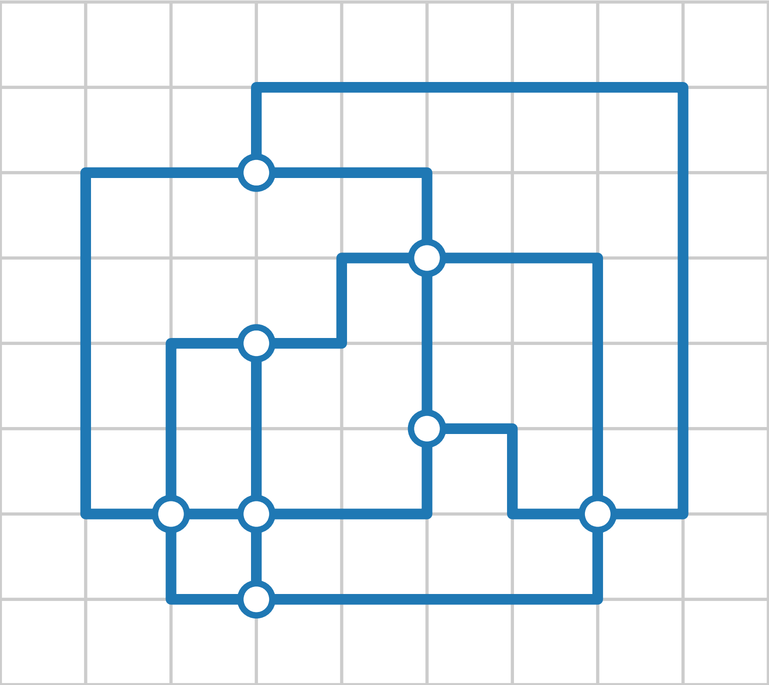


Planarization.

- Fix embedding
- Crossings become vertices



Orthogonal Layout – Definition

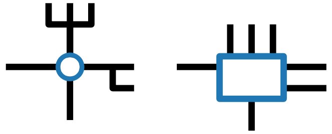


Definition.
A drawing Γ of a graph G is called **orthogonal** if

- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

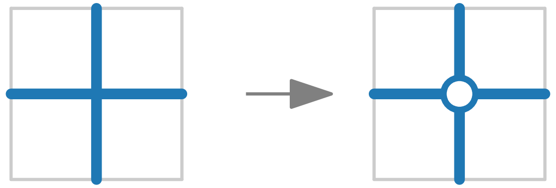
Observations.

- Edges lie on a grid $\Rightarrow$ **bends** lie on grid points
- Max. degree of each vertex is at most 4
- Otherwise

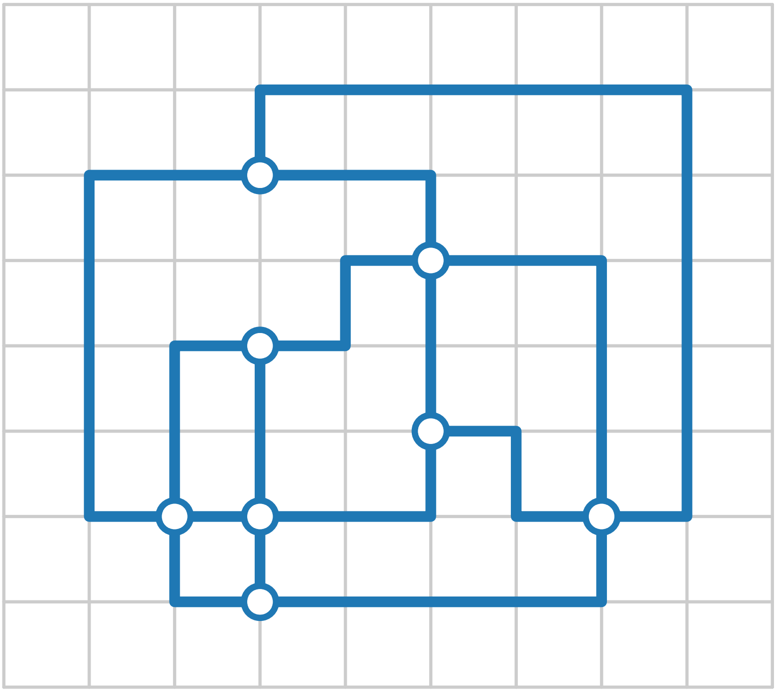


Planarization.

- Fix embedding
- Crossings become vertices

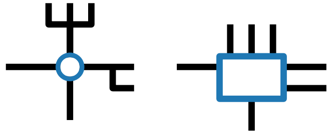


Orthogonal Layout – Definition

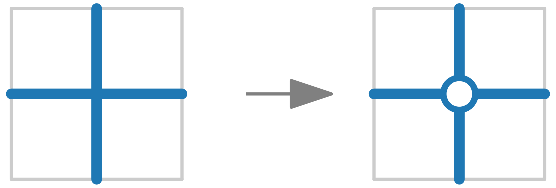


Definition.
A drawing Γ of a graph G is called **orthogonal** if
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

Observations.
- Edges lie on a grid ⇒ **bends** lie on grid points
- Max. degree of each vertex is at most 4
- Otherwise

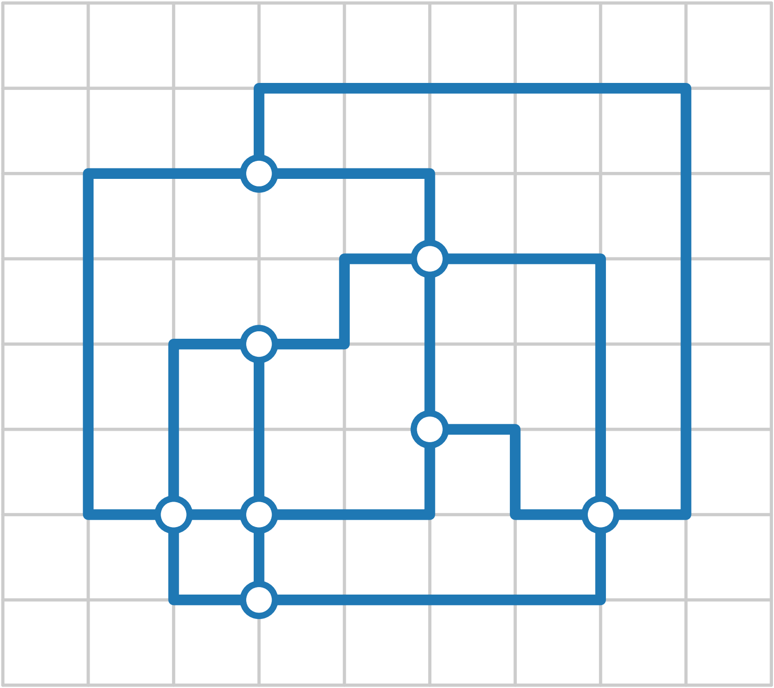


Planarization.
- Fix embedding
- Crossings become vertices



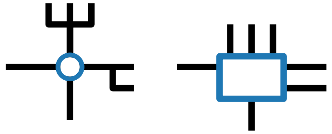
Aesthetic criteria to optimize.

Orthogonal Layout – Definition

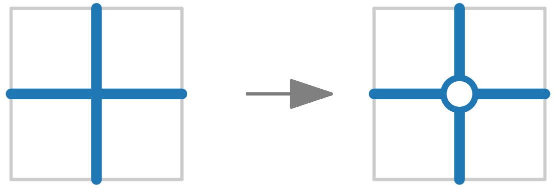


Definition.
A drawing Γ of a graph G is called **orthogonal** if
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

Observations.
- Edges lie on a grid ⇒ **bends** lie on grid points
- Max. degree of each vertex is at most 4
- Otherwise

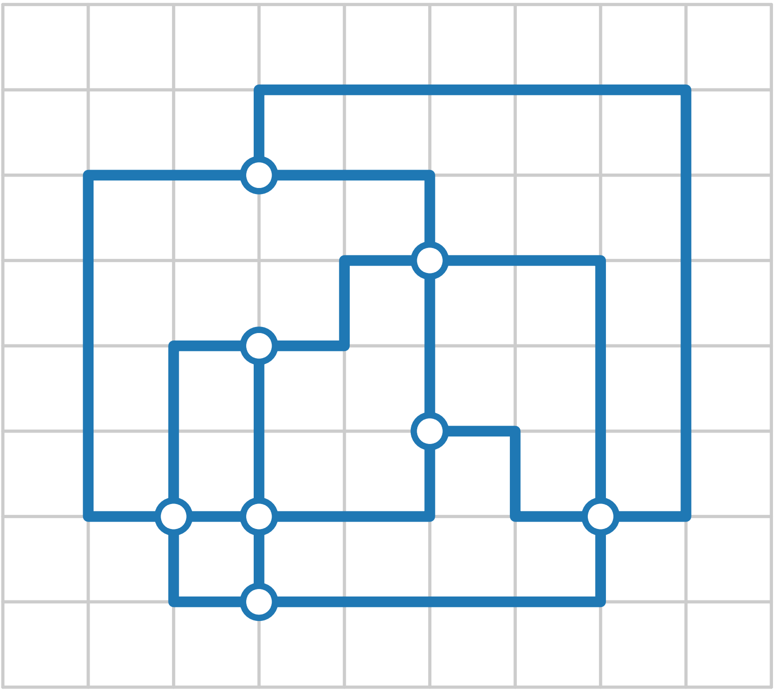


Planarization.
- Fix embedding
- Crossings become vertices



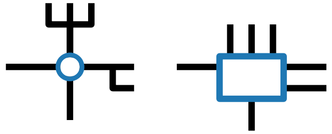
Aesthetic criteria to optimize.
- Number of bends

Orthogonal Layout – Definition

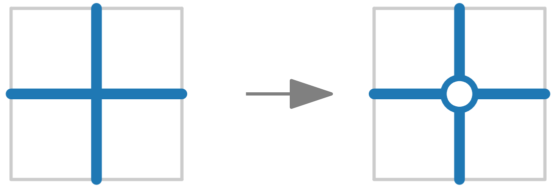


Definition.
A drawing Γ of a graph G is called **orthogonal** if
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

Observations.
- Edges lie on a grid ⇒ **bends** lie on grid points
- Max. degree of each vertex is at most 4
- Otherwise

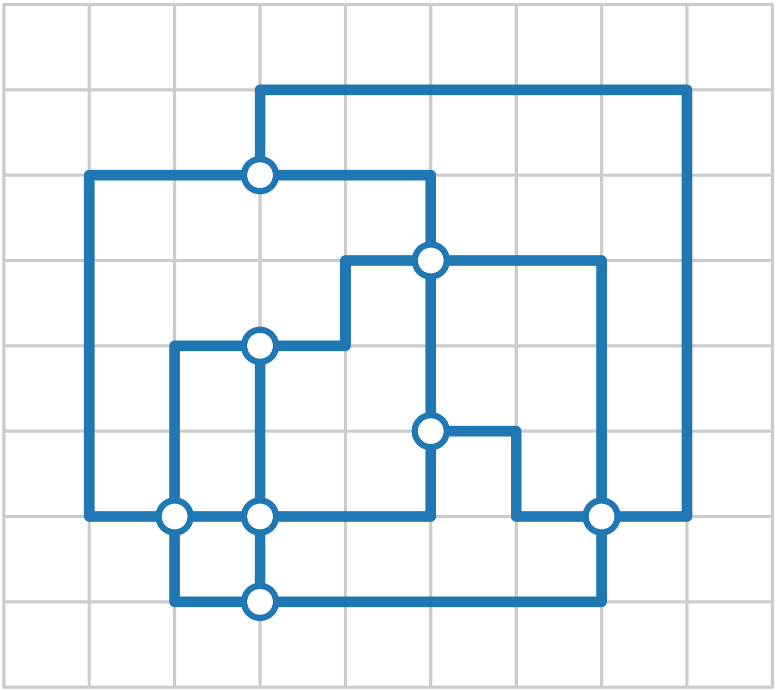


Planarization.
- Fix embedding
- Crossings become vertices



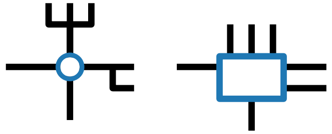
Aesthetic criteria to optimize.
- Number of bends
- Length of edges

Orthogonal Layout – Definition

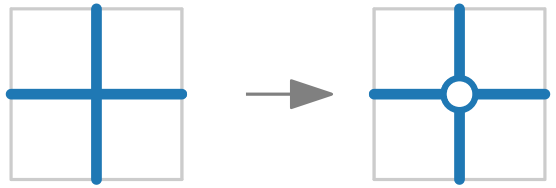


Definition.
A drawing Γ of a graph G is called **orthogonal** if
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

Observations.
- Edges lie on a grid ⇒ **bends** lie on grid points
- Max. degree of each vertex is at most 4
- Otherwise

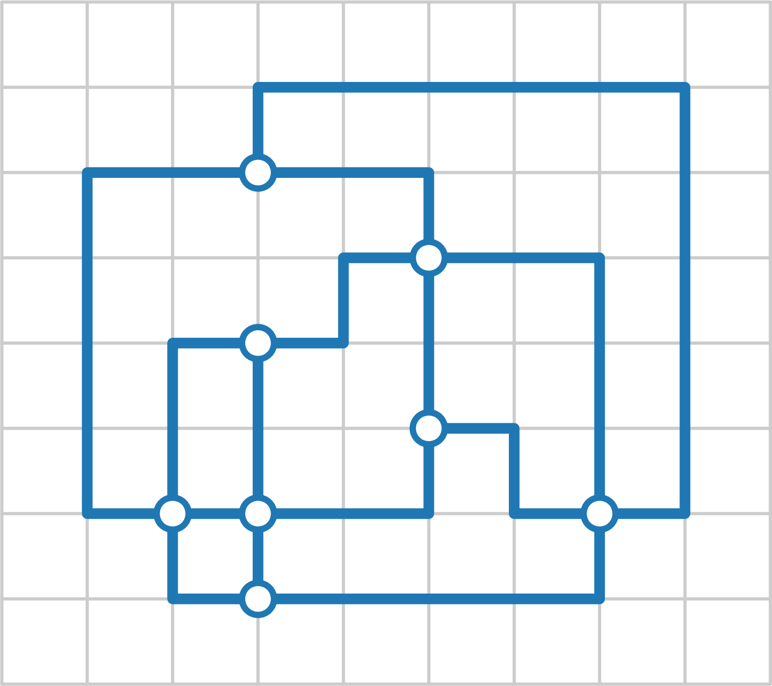


Planarization.
- Fix embedding
- Crossings become vertices



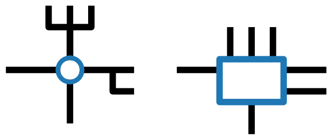
Aesthetic criteria to optimize.
- Number of bends
- Length of edges
- Width, height, area

Orthogonal Layout – Definition

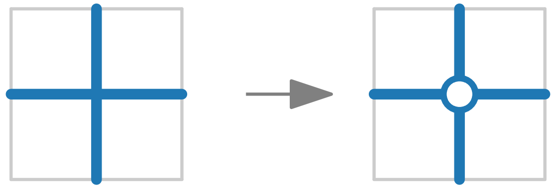


Definition.
A drawing Γ of a graph G is called **orthogonal** if
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical line segments of the grid, and
- pairs of edges are disjoint or cross orthogonally.

Observations.
- Edges lie on a grid ⇒ **bends** lie on grid points
- Max. degree of each vertex is at most 4
- Otherwise



Planarization.
- Fix embedding
- Crossings become vertices



Aesthetic criteria to optimize.
- Number of bends
- Length of edges
- Width, height, area
- Monotonicity of edges
- ...

Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

TOPOLOGY – SHAPE – METRICS

Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

$V(G) = \{v_1, v_2, v_3, v_4\}$
$E(G) = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$

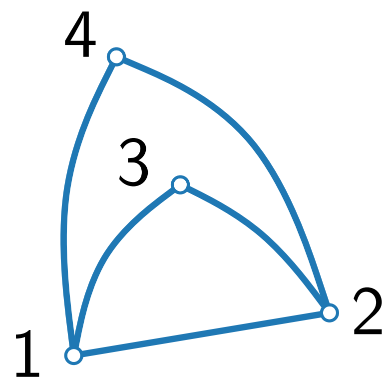
TOPOLOGY — SHAPE — METRICS

Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

$V(G) = \{v_1, v_2, v_3, v_4\}$
$E(G) = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$

combinatorial embedding/ planarization



TOPOLOGY — SHAPE — METRICS

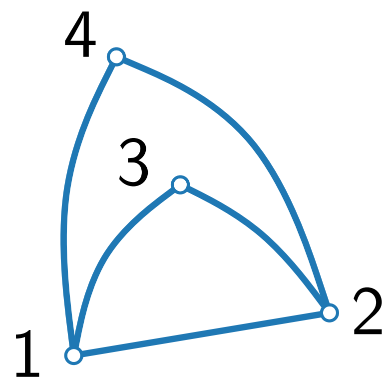
Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

$V(G) = \{v_1, v_2, v_3, v_4\}$
$E(G) = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$

combinatorial embedding/ planarization

reduce crossings



Topology — Shape — Metrics

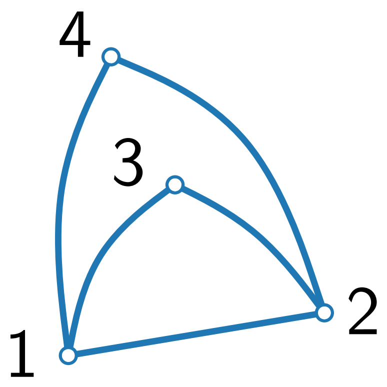
Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

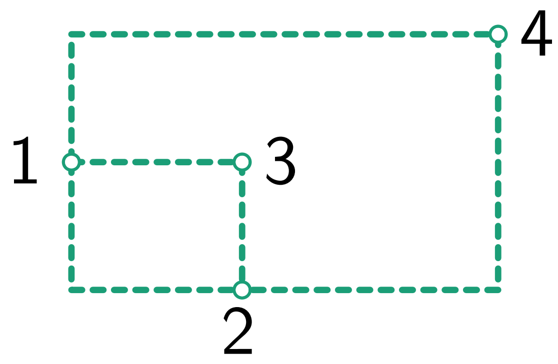
$V(G) = \{v_1, v_2, v_3, v_4\}$
$E(G) = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$

combinatorial embedding/ planarization

reduce crossings



orthogonal representation



Topology — Shape — Metrics

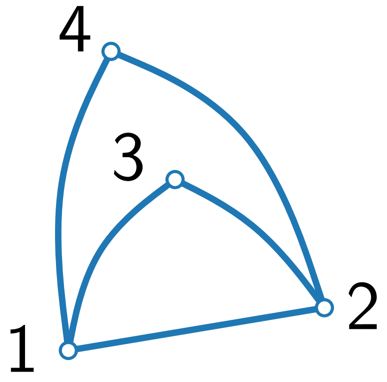
Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

$V(G) = \{v_1, v_2, v_3, v_4\}$
$E(G) = \{v_1 v_2, v_1 v_3, v_1 v_4, v_2 v_3, v_2 v_4\}$

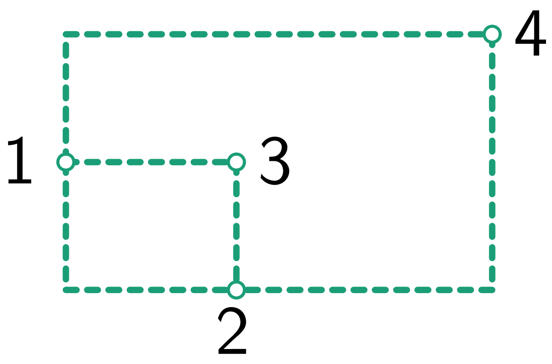
reduce crossings

combinatorial embedding/ planarization



bend minimization

orthogonal representation

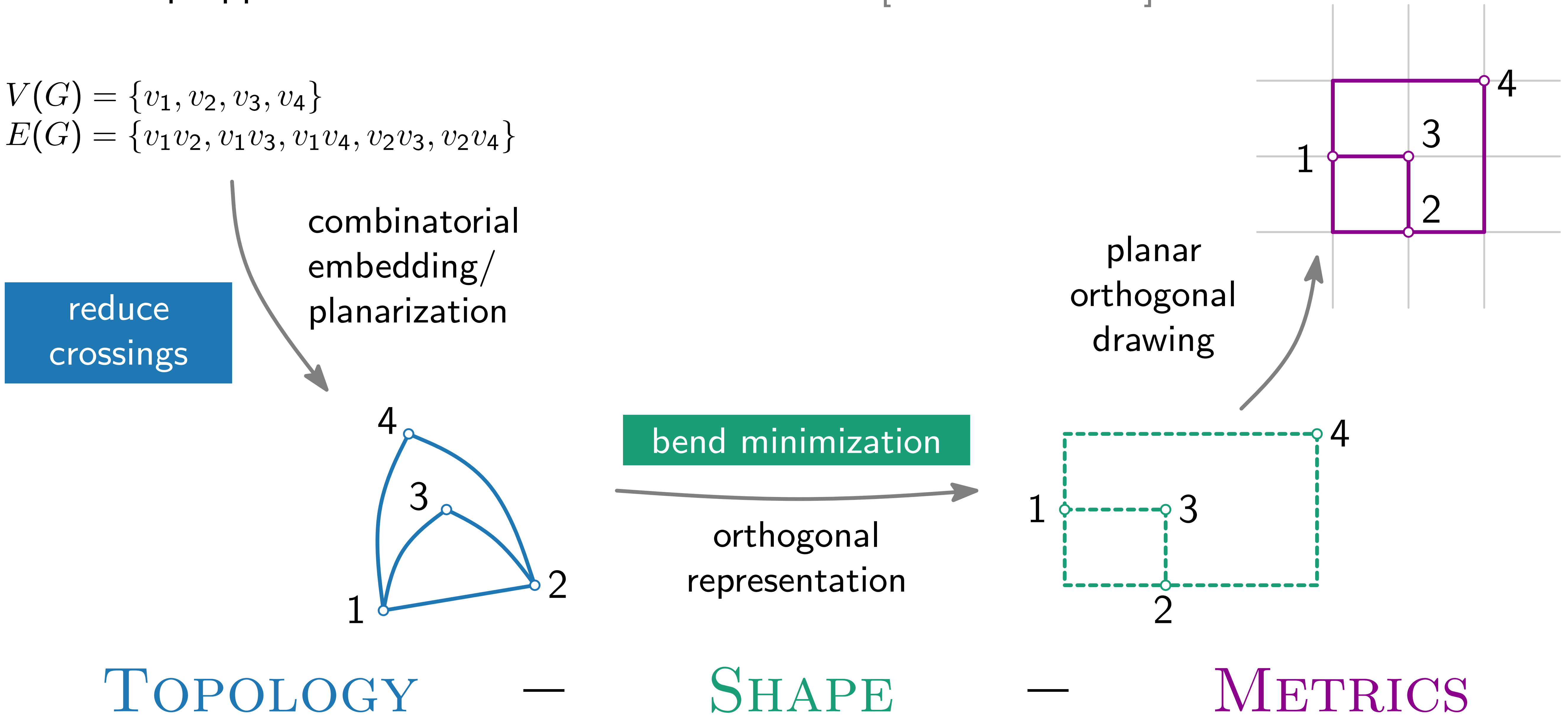


Topology — Shape — Metrics

Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

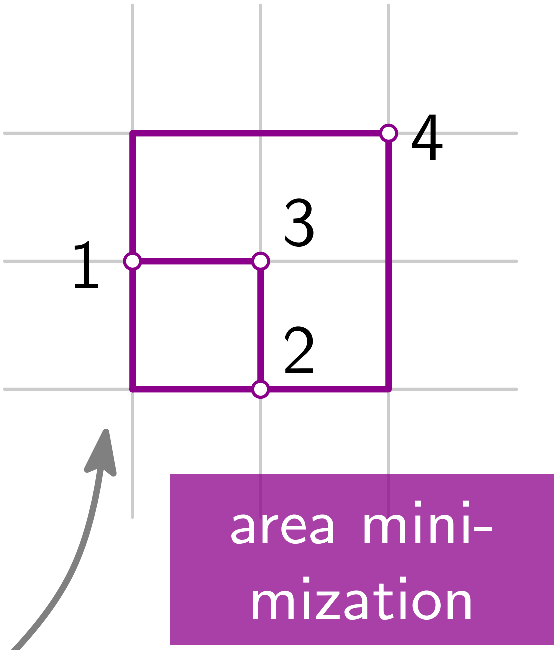
$$V(G) = \{v_1, v_2, v_3, v_4\}$$
$$E(G) = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$$

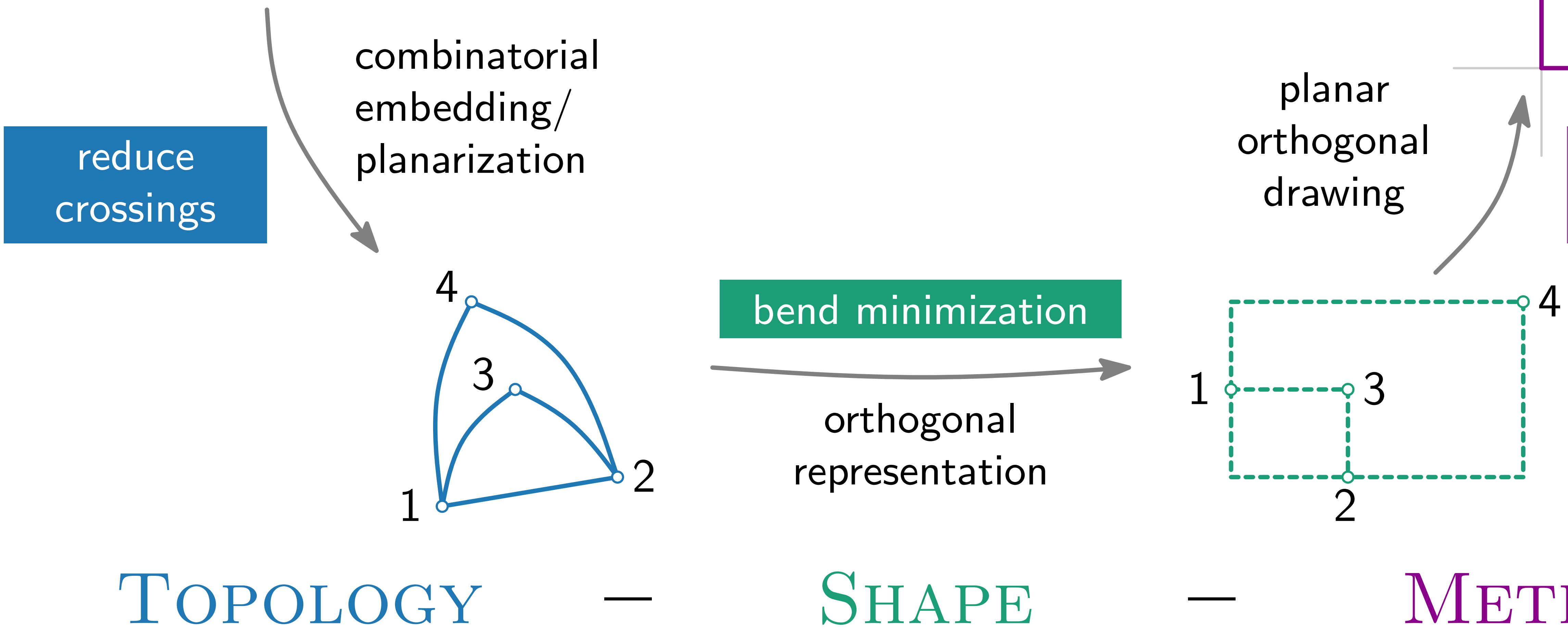


Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

$V(G) = \{v_1, v_2, v_3, v_4\}$
$E(G) = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$

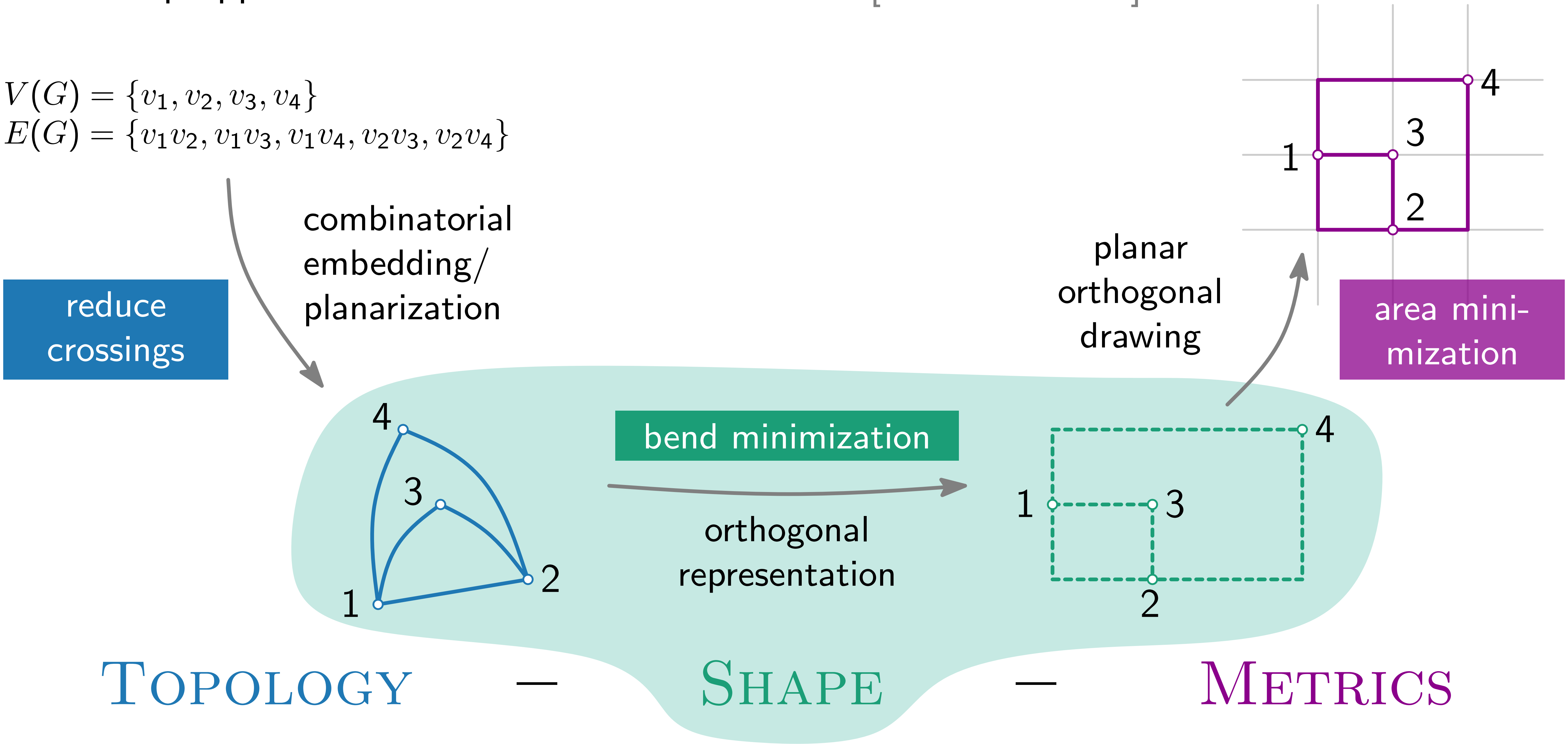




Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

$$V(G) = \{v_1, v_2, v_3, v_4\}$$
$$E(G) = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$$



Orthogonal Representation

Idea.

Describe orthogonal drawing combinatorially.

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorially.

Definitions.
Let G be a plane graph with set F of faces and outer face $f_0 \in F$.

Orthogonal Representation

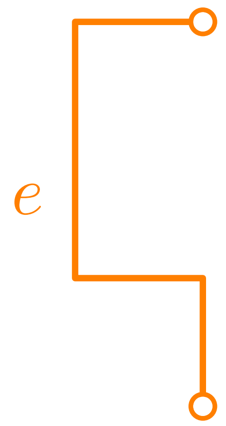
Idea.

Describe orthogonal drawing combinatorially.

Definitions.

Let G be a plane graph with set F of faces and outer face $f_0 \in F$.

- Let e be an edge

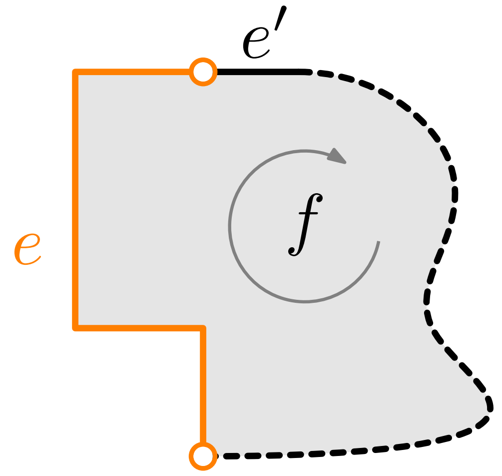


Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorially.

Definitions.
Let G be a plane graph with set F of faces and outer face $f_0 \in F$.

- Let e be an edge with the face f to the right.



Orthogonal Representation

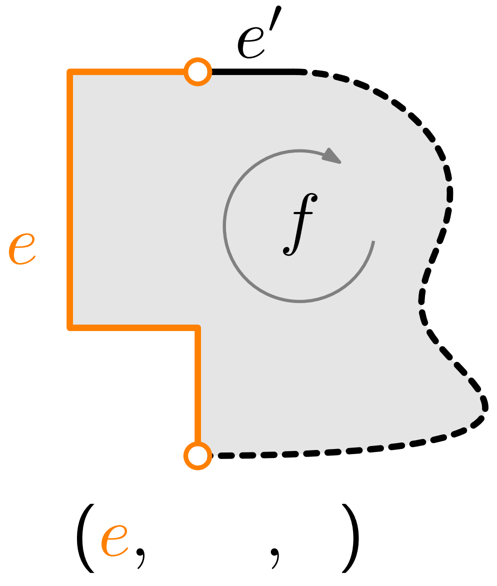
Idea.

Describe orthogonal drawing combinatorially.

Definitions.

Let G be a plane graph with set F of faces and outer face $f_0 \in F$.

- Let e be an edge with the face f to the right.
  An **edge description** of e w.r.t. f is a triple (e, δ, α) where

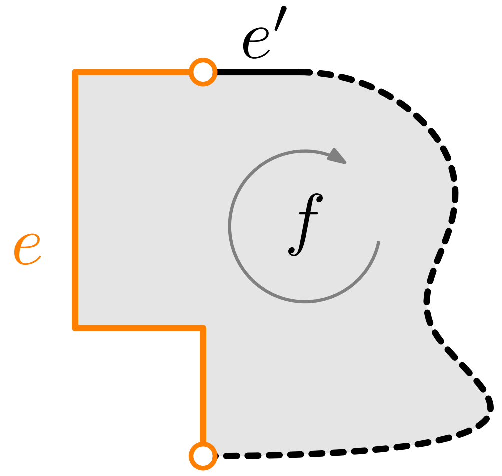


Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorially.

Definitions.
Let G be a plane graph with set F of faces and outer face $f_0 \in F$.

- Let e be an edge with the face f to the right.
  An **edge description** of e w.r.t. f is a triple (e, δ, α) where
  - $\delta \in \{0, 1\}^*$ (where 0 = right bend, 1 = left bend)



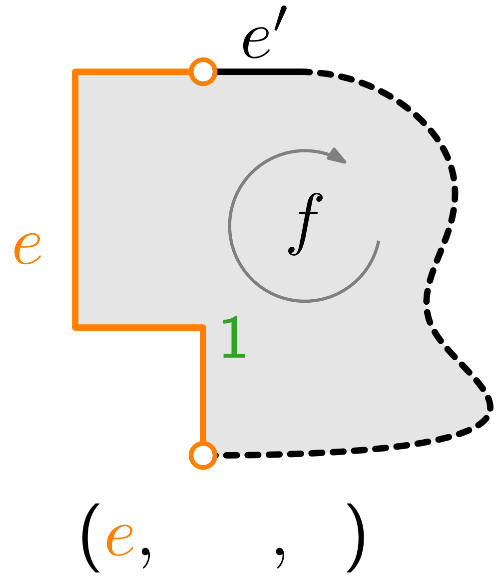
$(e, \quad , \quad)$

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorially.

Definitions.
Let G be a plane graph with set F of faces and outer face $f_0 \in F$.

- Let e be an edge with the face f to the right.
  An **edge description** of e w.r.t. f is a triple (e, δ, α) where
  - $\delta \in \{0, 1\}^*$ (where $0 =$ right bend, $1 =$ left bend)

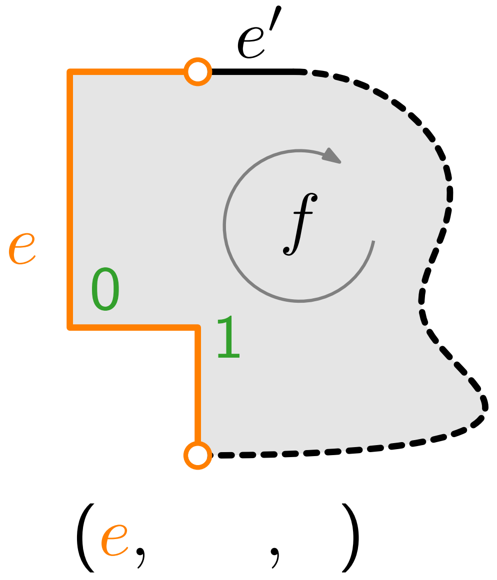


Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorially.

Definitions.
Let G be a plane graph with set F of faces and outer face $f_0 \in F$.

- Let e be an edge with the face f to the right.
  An **edge description** of e w.r.t. f is a triple (e, δ, α) where
  - $\delta \in \{0, 1\}^*$ (where $0 =$ right bend, $1 =$ left bend)

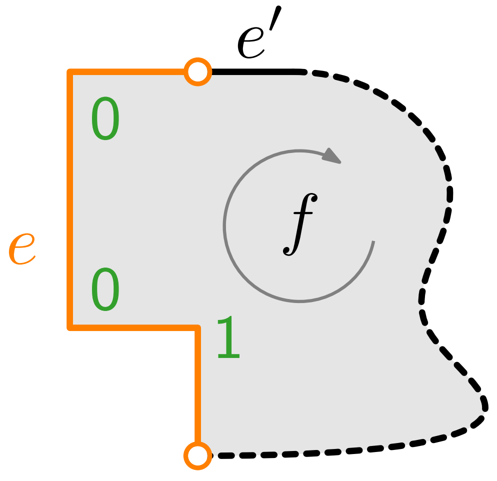


Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorially.

Definitions.
Let G be a plane graph with set F of faces and outer face $f_0 \in F$.

- Let e be an edge with the face f to the right.
  An **edge description** of e w.r.t. f is a triple (e, δ, α) where
  - $\delta \in \{0,1\}^*$ (where $0 =$ right bend, $1 =$ left bend)



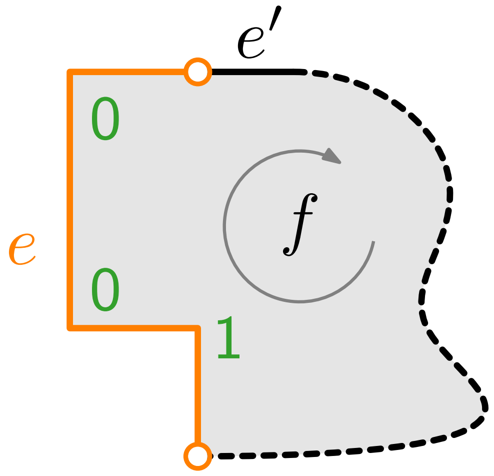
$(e, \quad , \quad)$

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorially.

Definitions.
Let G be a plane graph with set F of faces and outer face $f_0 \in F$.

- Let e be an edge with the face f to the right.
  An **edge description** of e w.r.t. f is a triple (e, δ, α) where
  - $\delta \in \{0,1\}^*$ (where 0 = right bend, 1 = left bend)



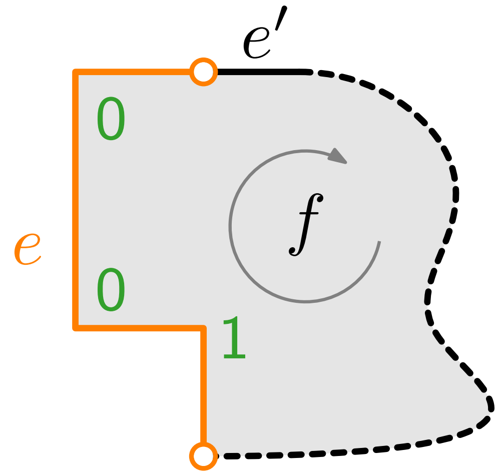
$(e, 100, \quad)$

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorially.

Definitions.
Let G be a plane graph with set F of faces and outer face $f_0 \in F$.

- Let e be an edge with the face f to the right.
  An **edge description** of e w.r.t. f is a triple (e, δ, α) where
  - $\delta \in \{0, 1\}^*$ (where 0 = right bend, 1 = left bend)
  - α is angle $\in \{\frac{\pi}{2}, \pi, \frac{3\pi}{2}, 2\pi\}$ between e and next edge e'



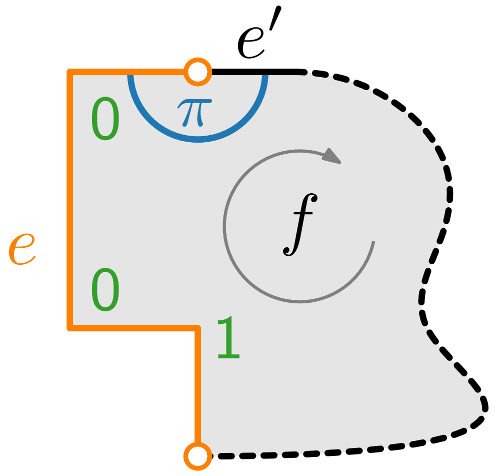
$(e, 100, \quad)$

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorially.

Definitions.
Let G be a plane graph with set F of faces and outer face $f_0 \in F$.

- Let e be an edge with the face f to the right.
  An **edge description** of e w.r.t. f is a triple (e, δ, α) where
  - $\delta \in \{0,1\}^*$ (where 0 = right bend, 1 = left bend)
  - α is angle $\in \{\frac{\pi}{2}, \pi, \frac{3\pi}{2}, 2\pi\}$ between e and next edge e'



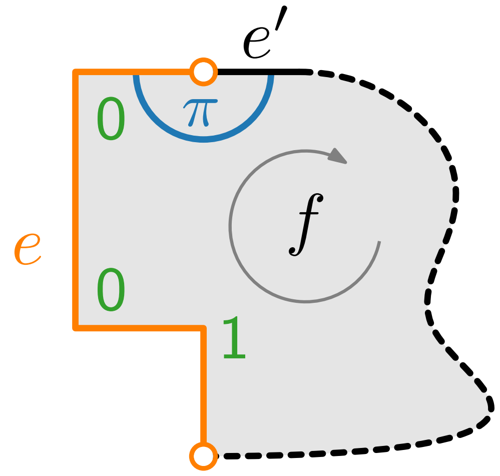
$(e, 100, \quad)$

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorially.

Definitions.
Let G be a plane graph with set F of faces and outer face $f_0 \in F$.

- Let e be an edge with the face f to the right.
  An **edge description** of e w.r.t. f is a triple (e, δ, α) where
  - $\delta \in \{0,1\}^*$ (where $0 =$ right bend, $1 =$ left bend)
  - α is angle $\in \{\frac{\pi}{2}, \pi, \frac{3\pi}{2}, 2\pi\}$ between e and next edge e'



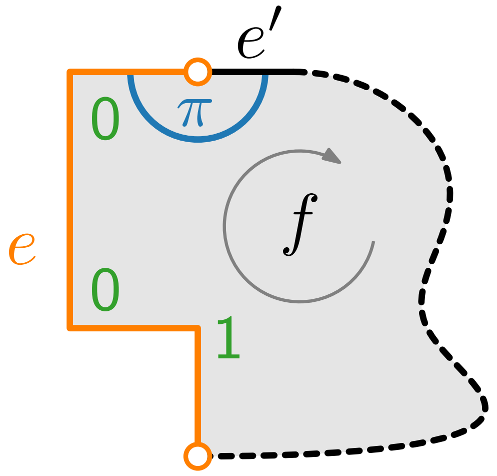
$(e, 100, \pi)$

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorially.

Definitions.
Let G be a plane graph with set F of faces and outer face $f_0 \in F$.

- Let e be an edge with the face f to the right.
  An **edge description** of e w.r.t. f is a triple (e, δ, α) where
  - $\delta \in \{0, 1\}^*$ (where $0 =$ right bend, $1 =$ left bend)
  - α is angle $\in \{\frac{\pi}{2}, \pi, \frac{3\pi}{2}, 2\pi\}$ between e and next edge e'



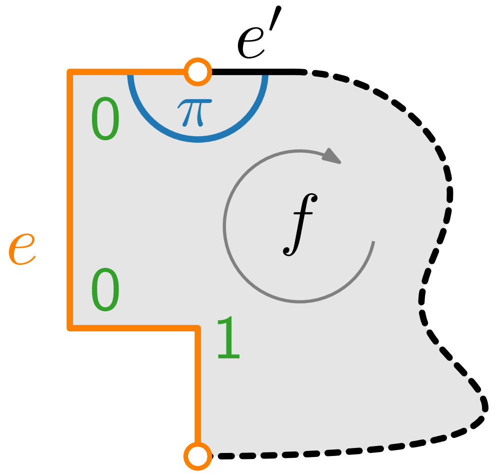
$(e, 100, \pi)$

- A **face representation** $H(f)$ of a face f is a clockwise ordered sequence $(e_1, \delta_1, \alpha_1), (e_2, \delta_2, \alpha_2), \dots, (e_{\deg(f)}, \delta_{\deg(f)}, \alpha_{\deg(f)})$ of edge descriptions w.r.t. f.

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorially.

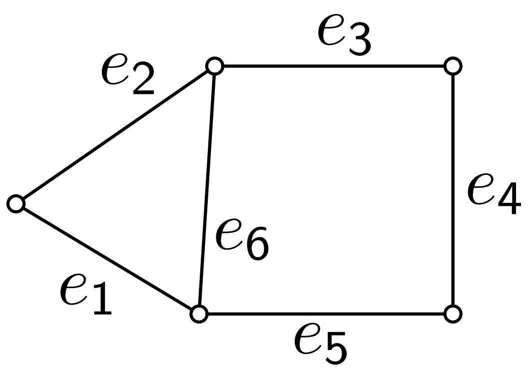
Definitions.
Let G be a plane graph with set F of faces and outer face $f_0 \in F$.



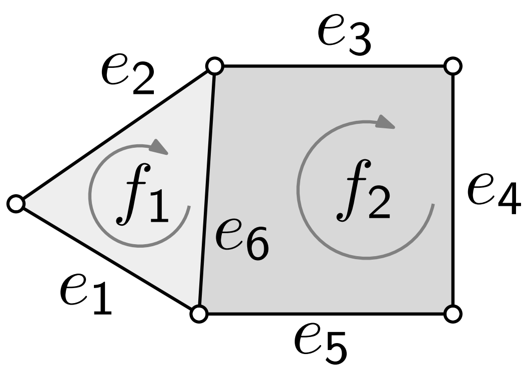
- Let e be an edge with the face f to the right.
  An **edge description** of e w.r.t. f is a triple (e, δ, α) where
  - $\delta \in \{0, 1\}^*$ (where 0 = right bend, 1 = left bend)
  - α is angle $\in \{\frac{\pi}{2}, \pi, \frac{3\pi}{2}, 2\pi\}$ between e and next edge e'
- A **face representation** $H(f)$ of a face f is a clockwise ordered sequence $(e_1, \delta_1, \alpha_1), (e_2, \delta_2, \alpha_2), \dots, (e_{\deg(f)}, \delta_{\deg(f)}, \alpha_{\deg(f)})$ of edge descriptions w.r.t. f.
- An **orthogonal representation** $H(G)$ of G is defined as

$$H(G) = \{H(f) \mid f \in F\}.$$

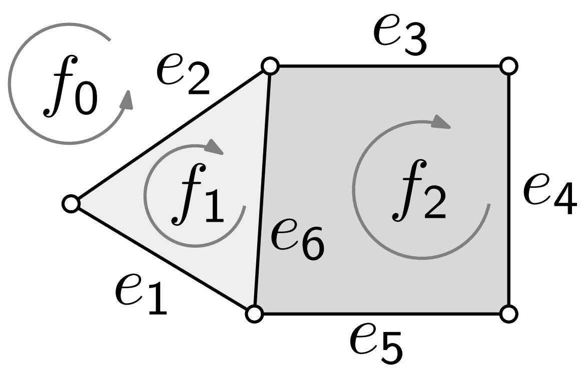
Orthogonal Representation – Example



Orthogonal Representation – Example

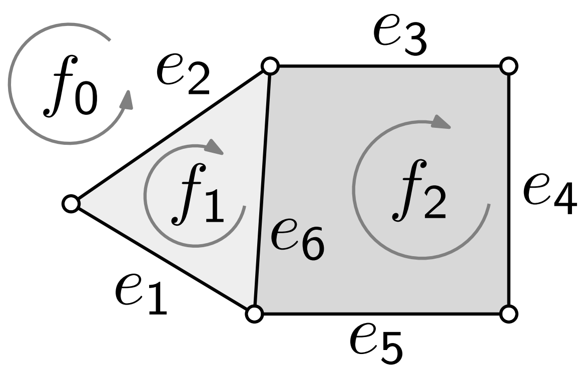


Orthogonal Representation – Example



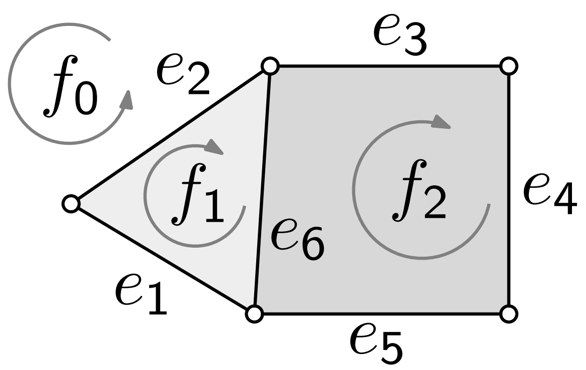
Orthogonal Representation – Example

$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$



Orthogonal Representation – Example

$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$



Combinatorial "drawing" of $H(G)$?

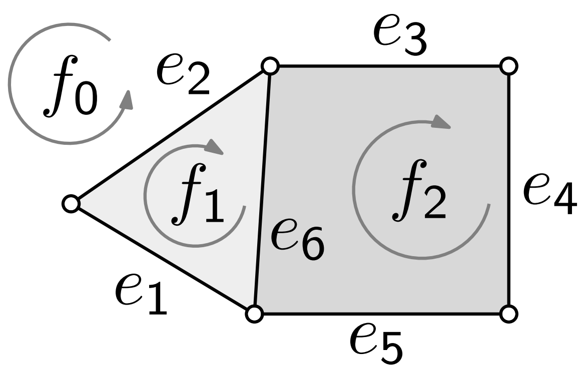
Orthogonal Representation – Example

$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$

$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$

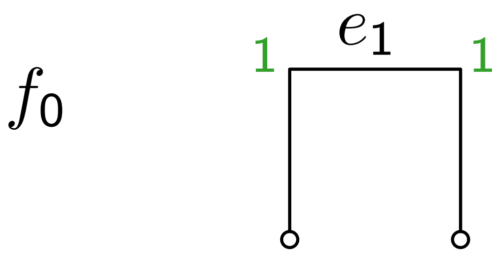
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

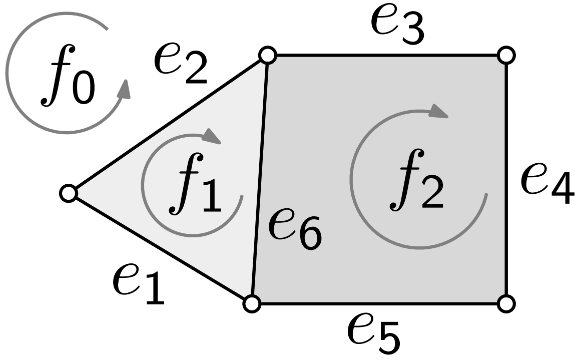
f_0



Orthogonal Representation – Example

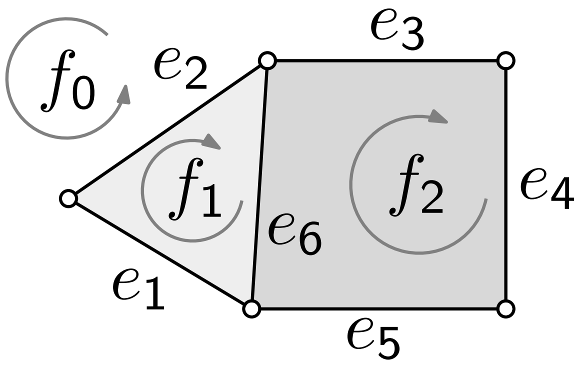
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

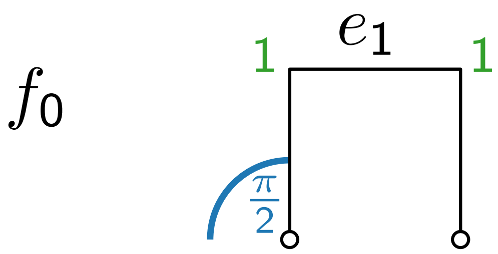




Orthogonal Representation – Example

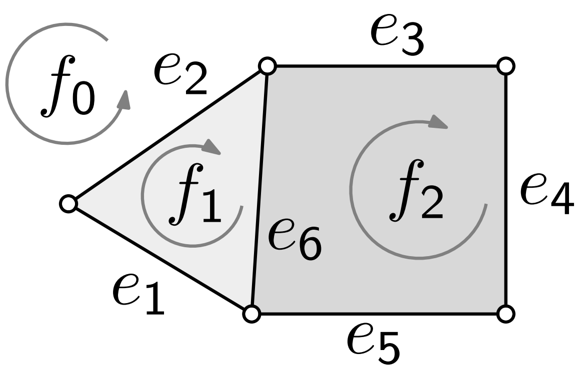
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

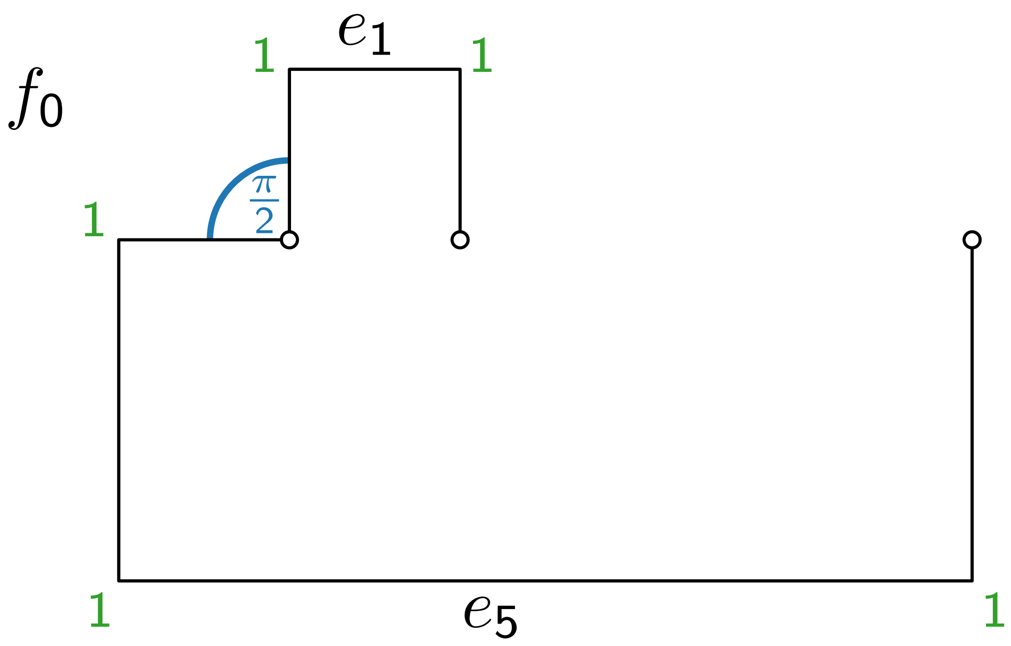




Orthogonal Representation – Example

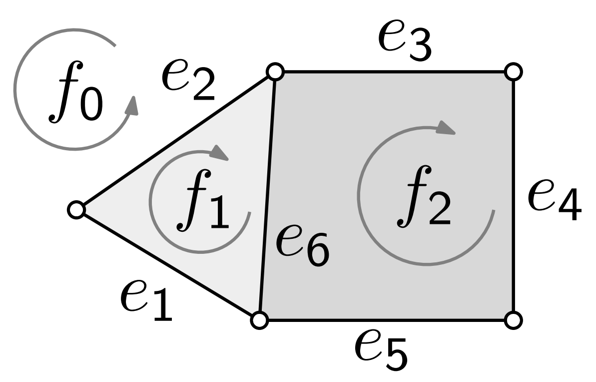
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

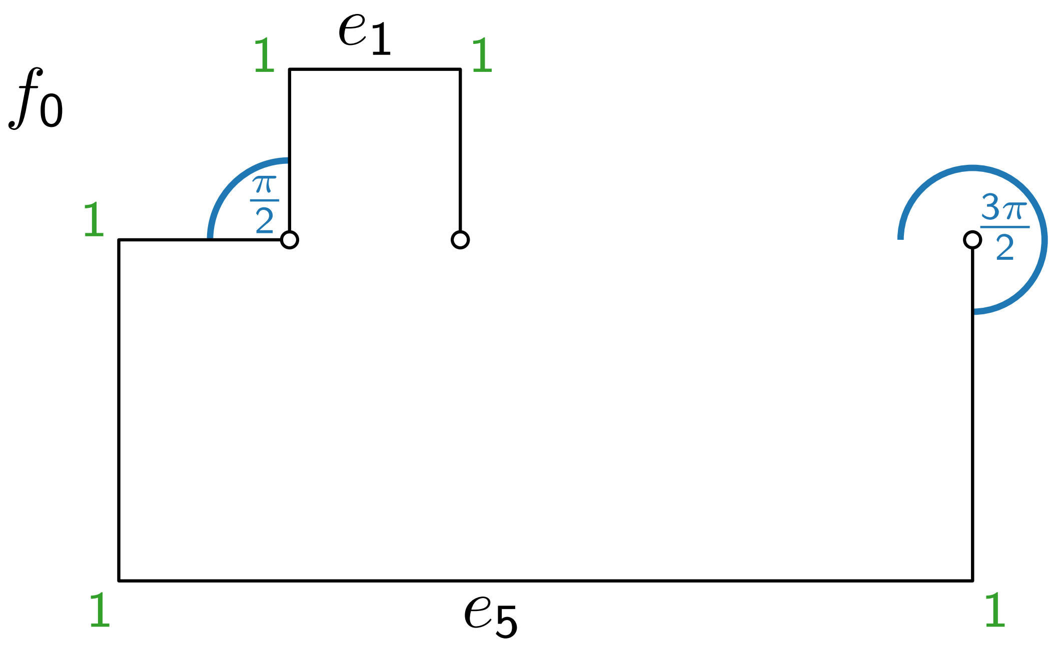




Orthogonal Representation – Example

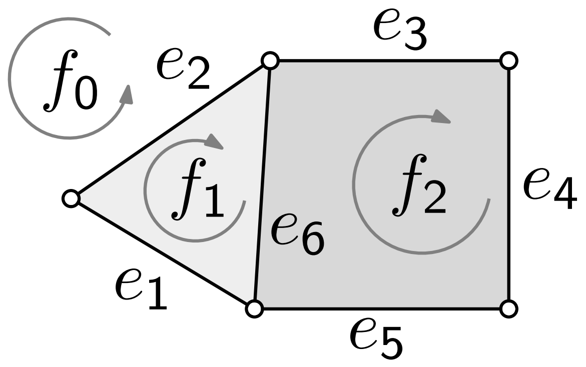
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

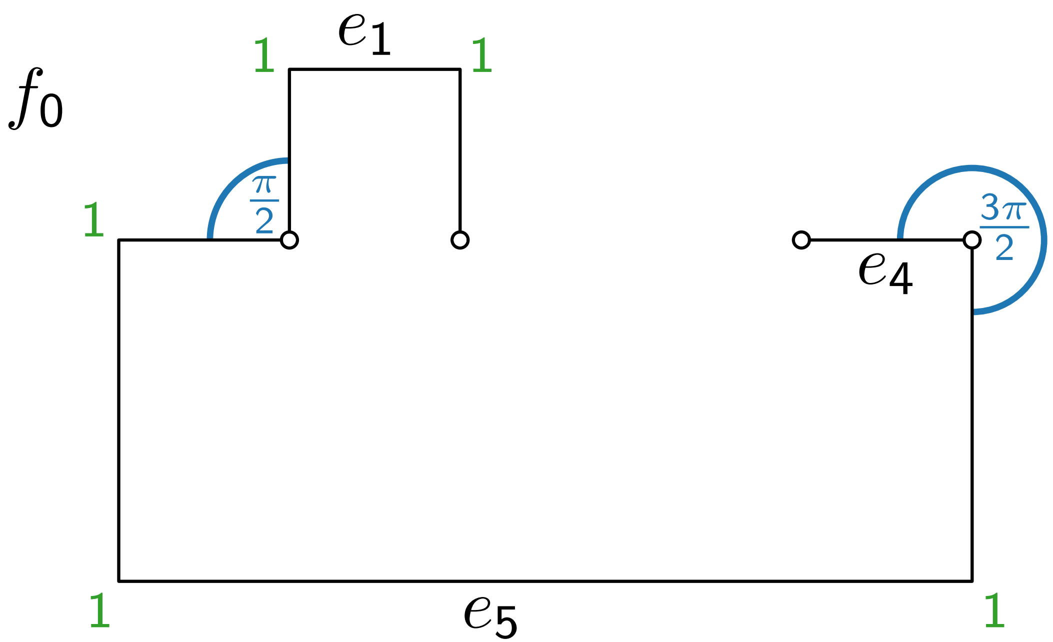




Orthogonal Representation – Example

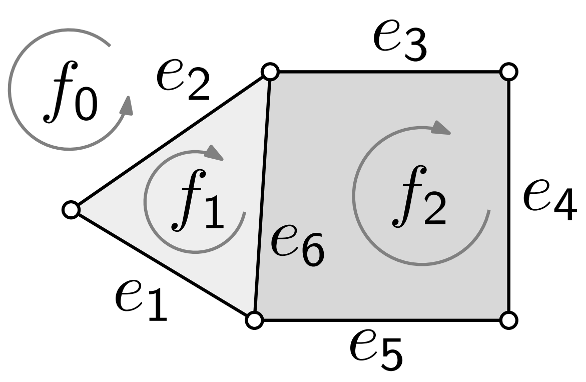
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

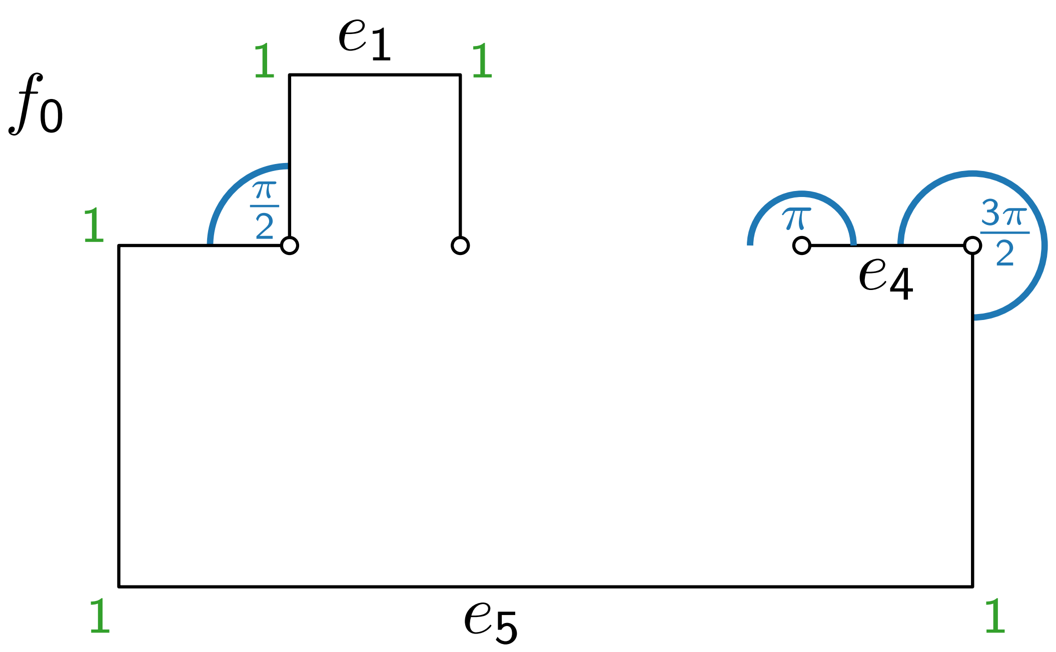




Orthogonal Representation – Example

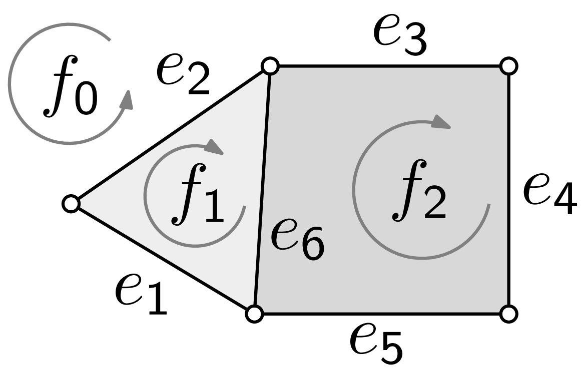
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

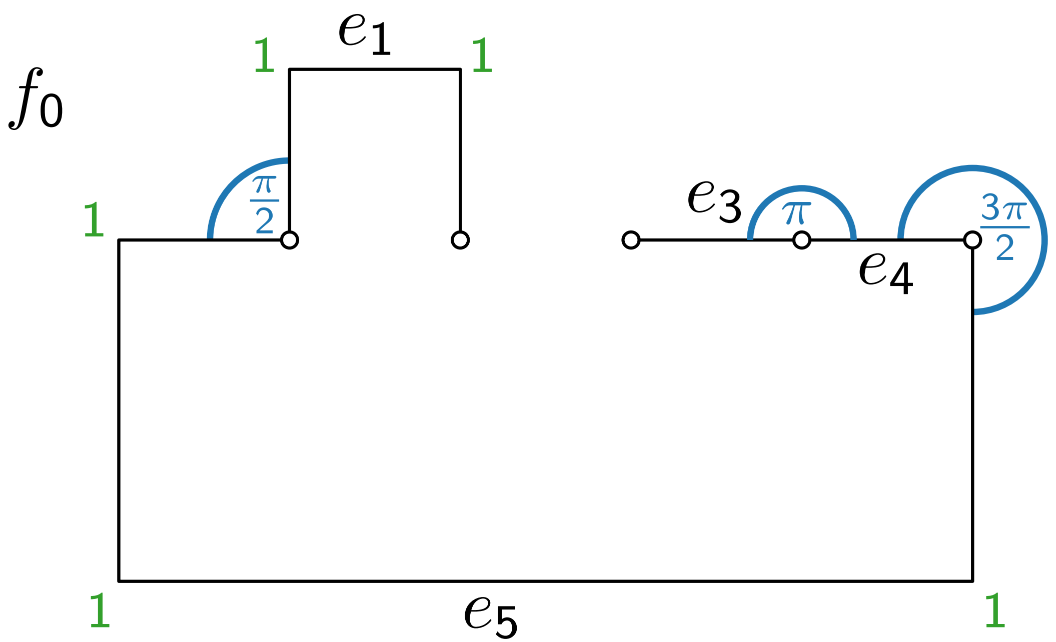




Orthogonal Representation – Example

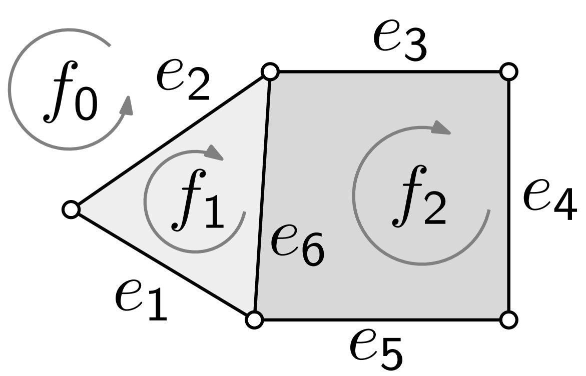
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

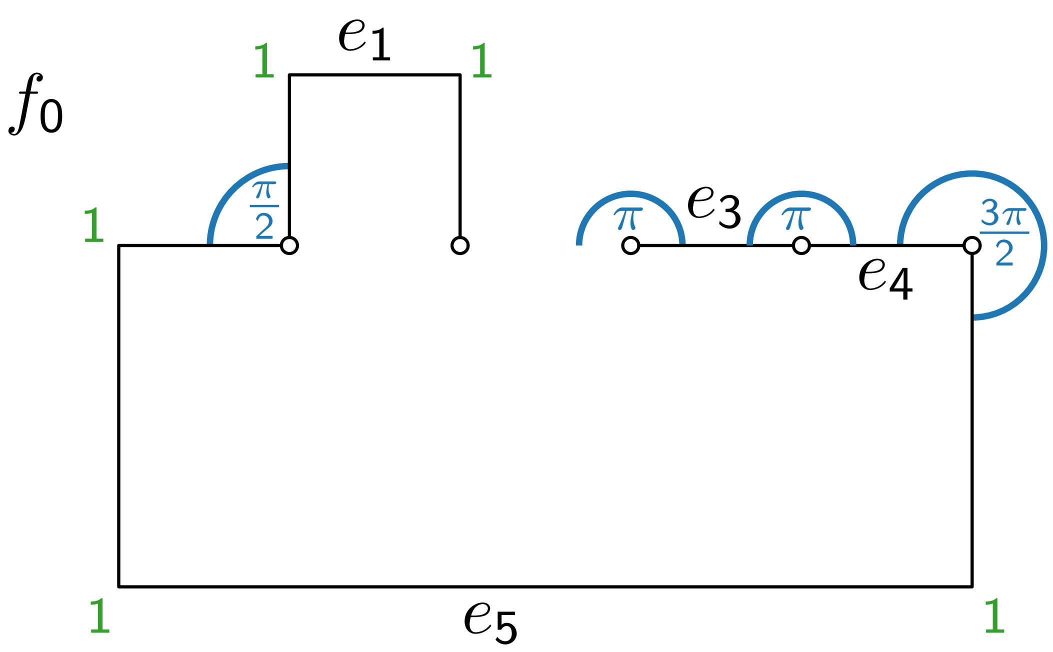




Orthogonal Representation – Example

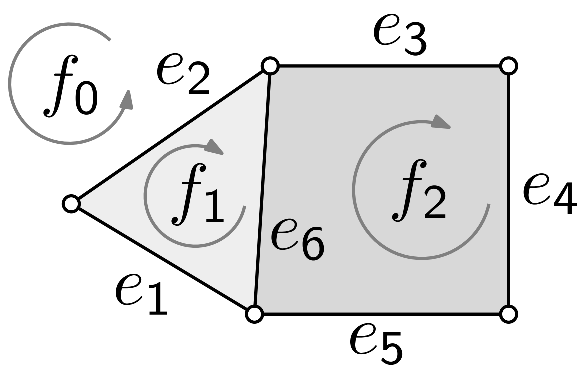
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

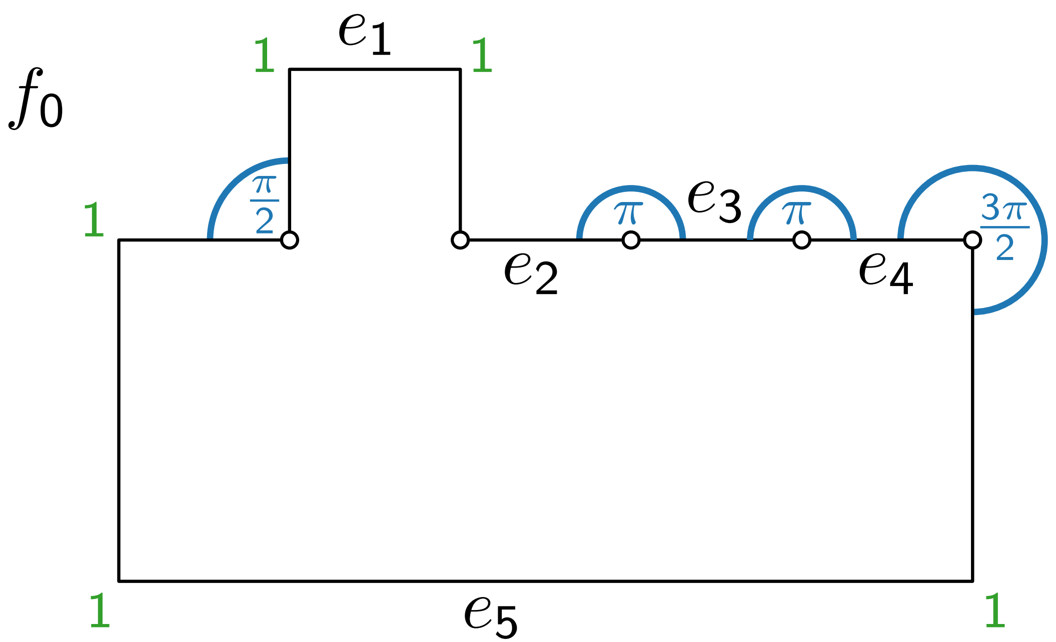




Orthogonal Representation – Example

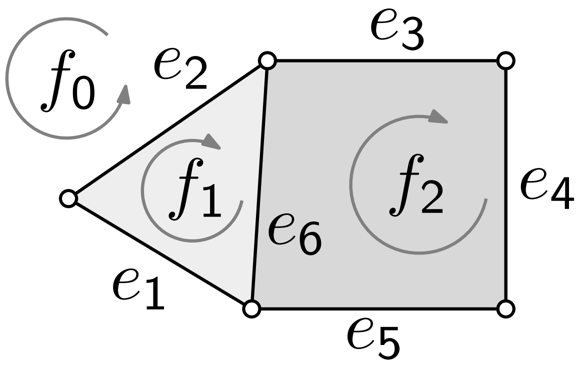
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

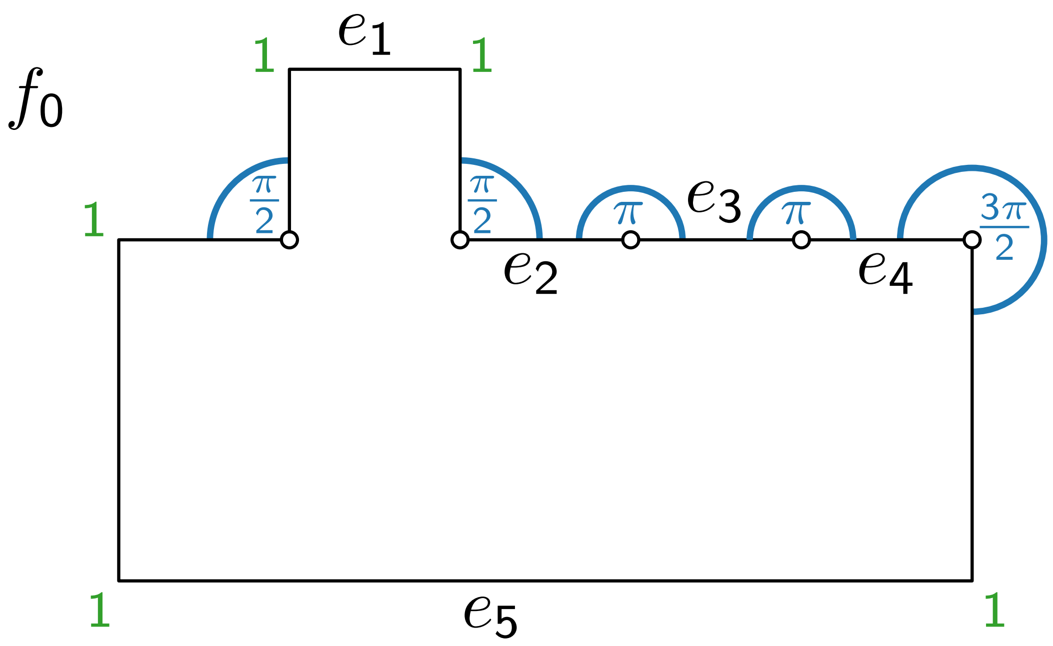




Orthogonal Representation – Example

$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$



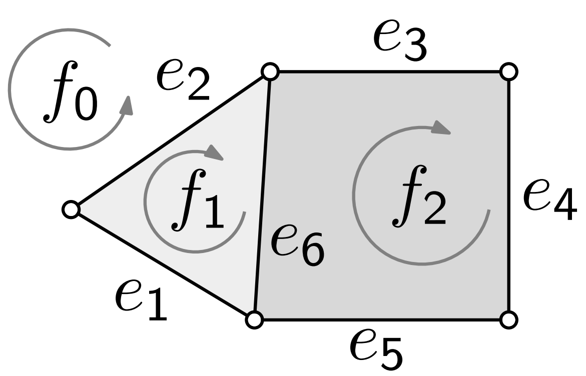


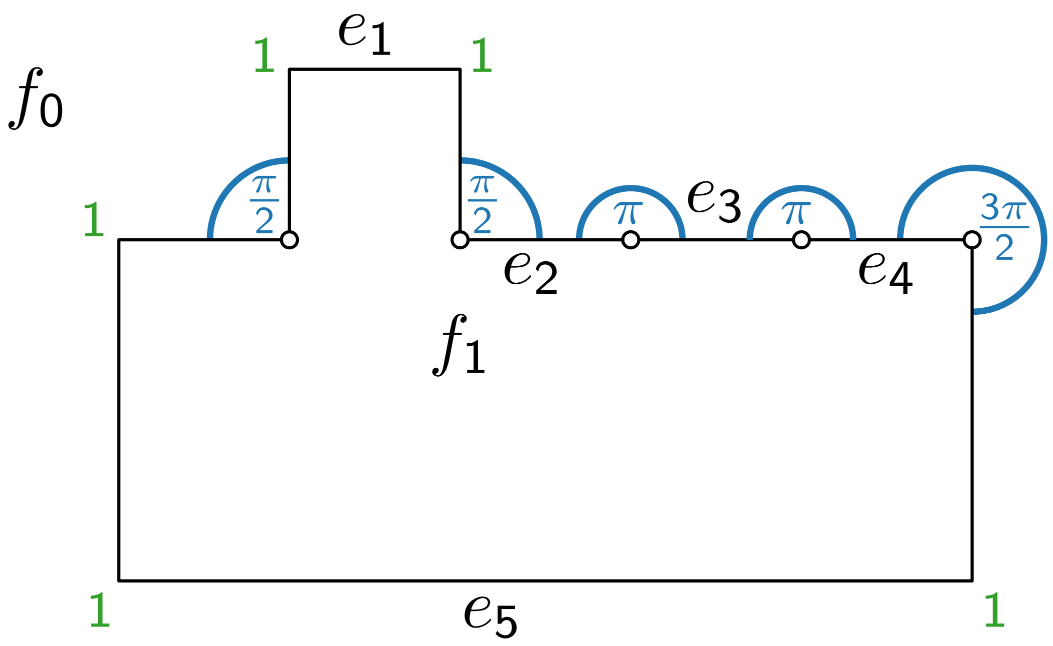
Orthogonal Representation – Example

$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$

$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$

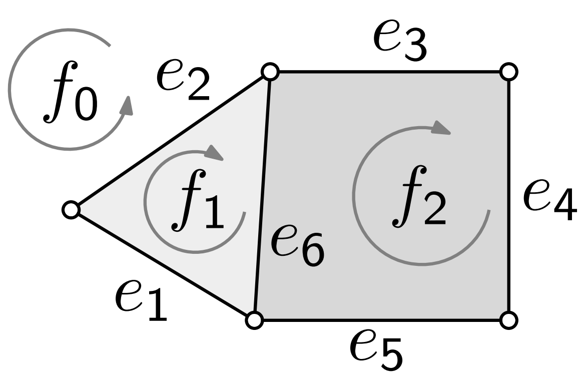
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

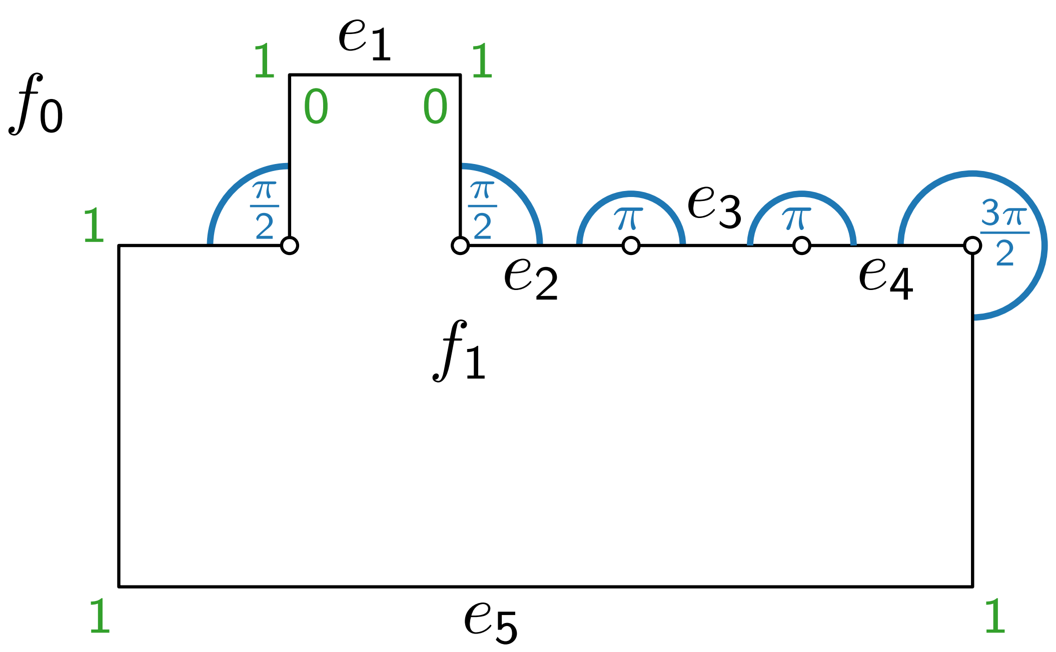




Orthogonal Representation – Example

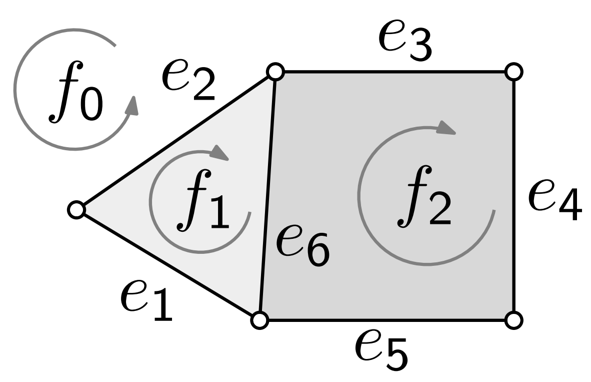
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

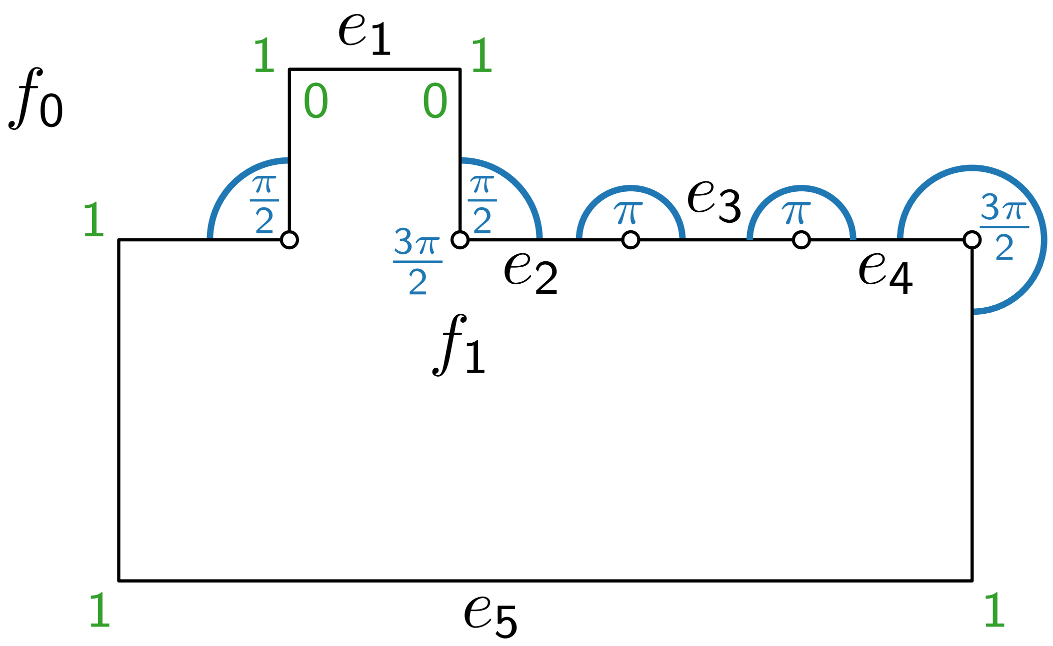




Orthogonal Representation – Example

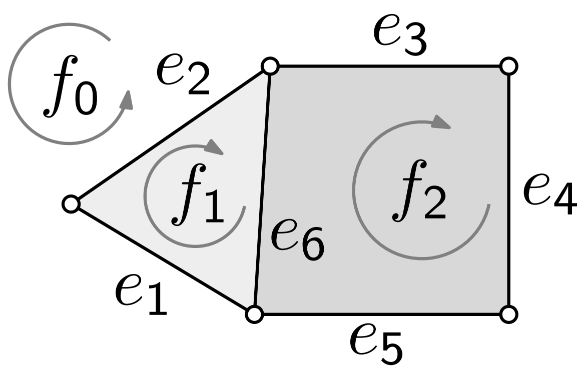
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

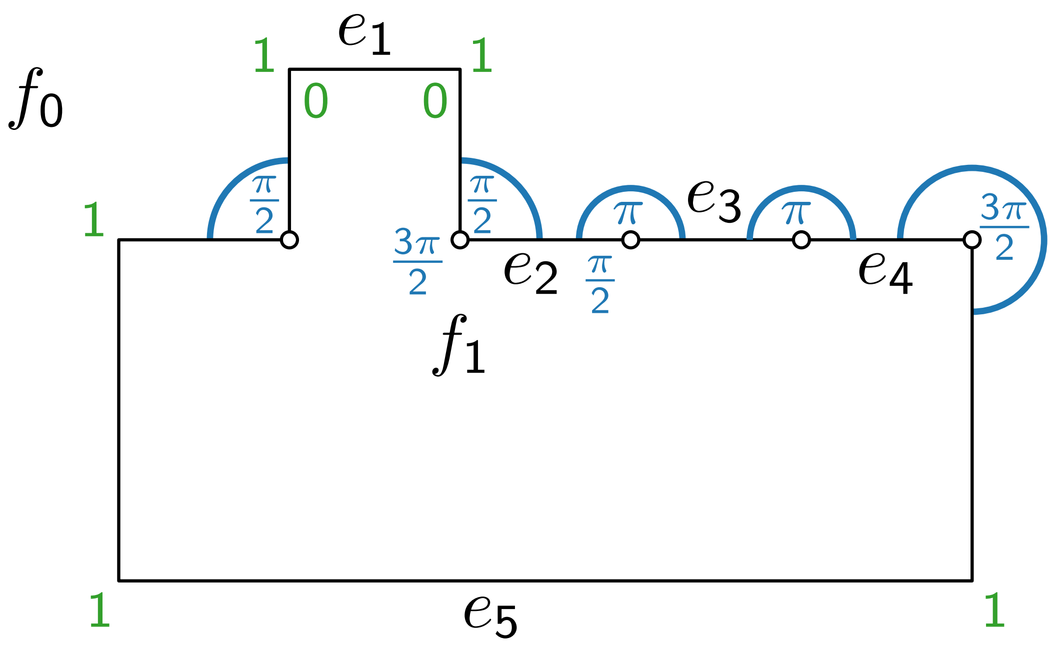




Orthogonal Representation – Example

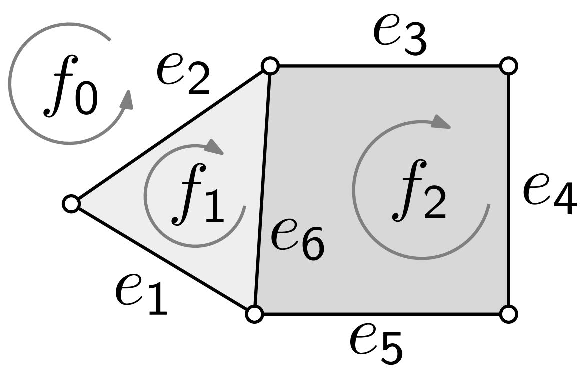
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

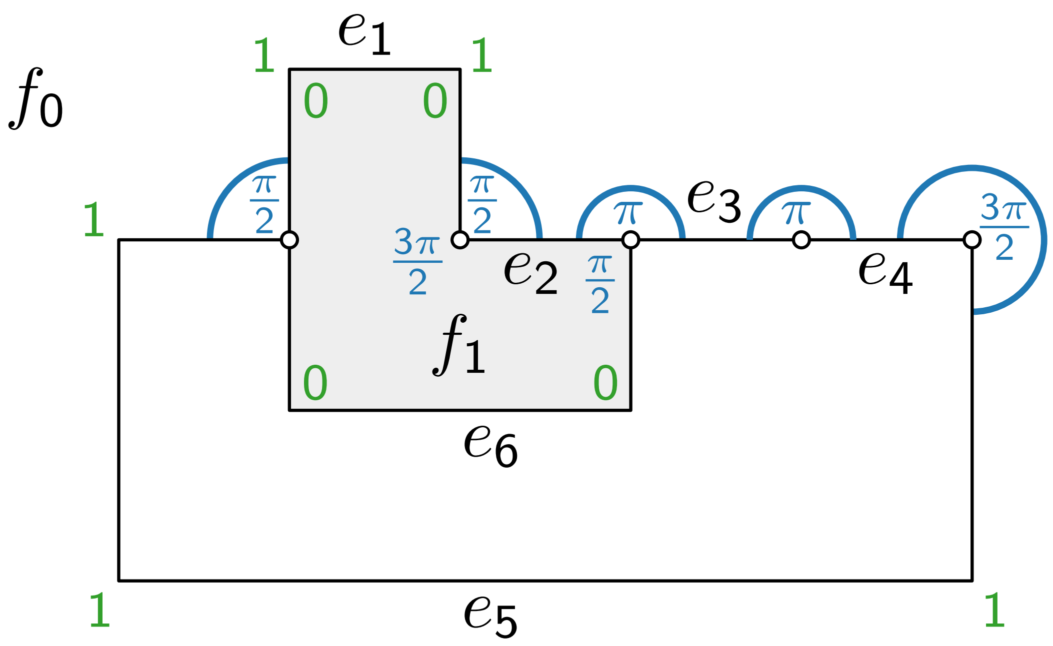




Orthogonal Representation – Example

$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$



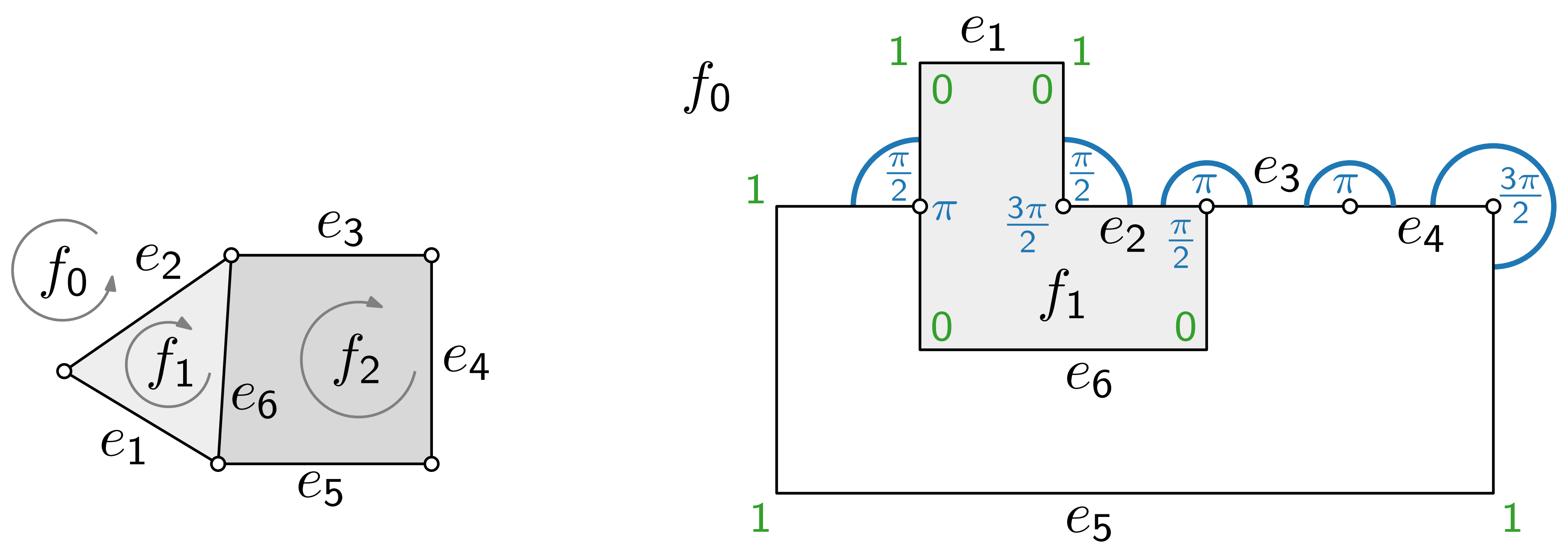


Orthogonal Representation – Example

$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$

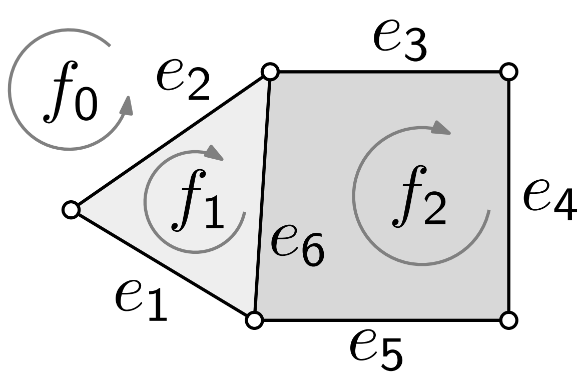
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$

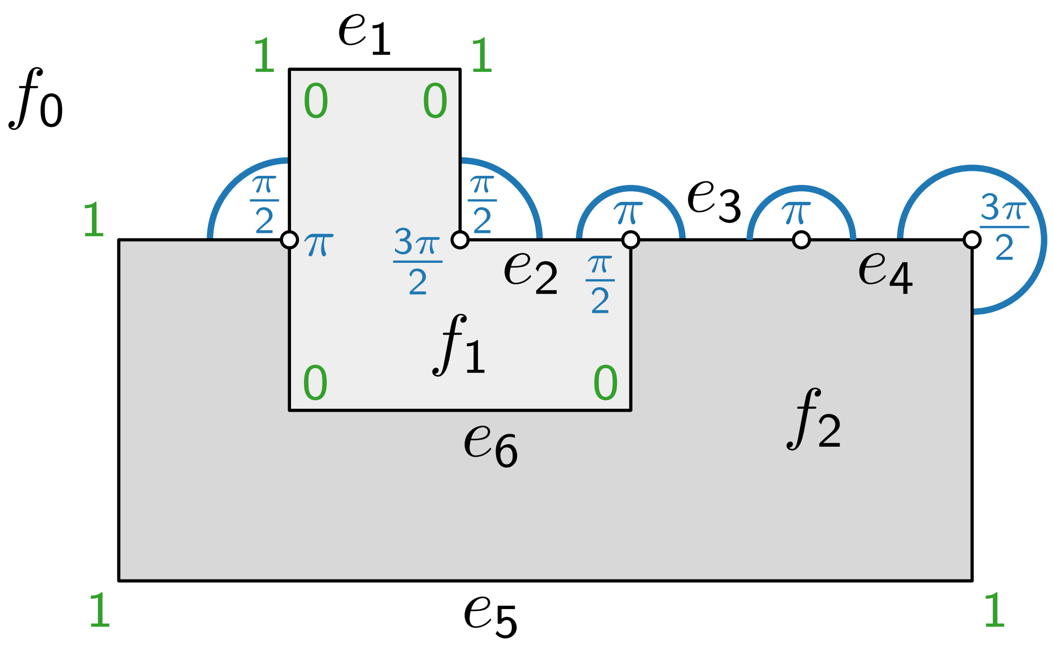
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$



Orthogonal Representation – Example

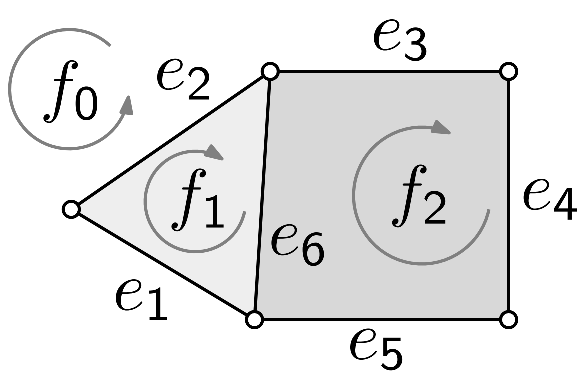
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

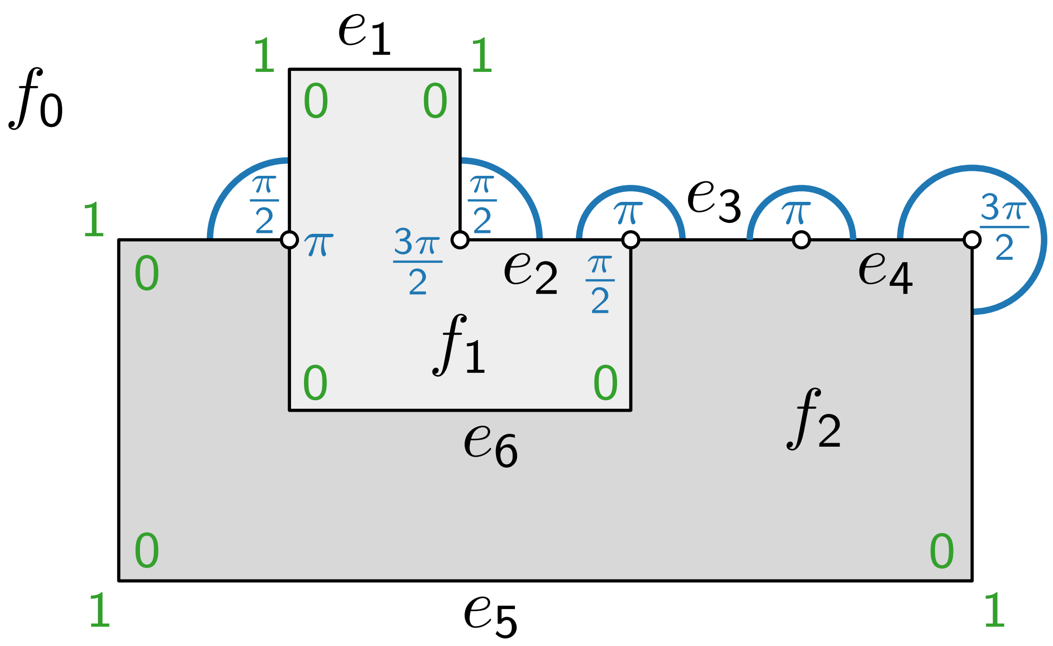




Orthogonal Representation – Example

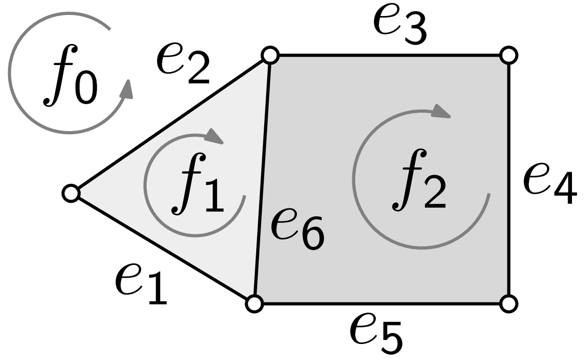
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

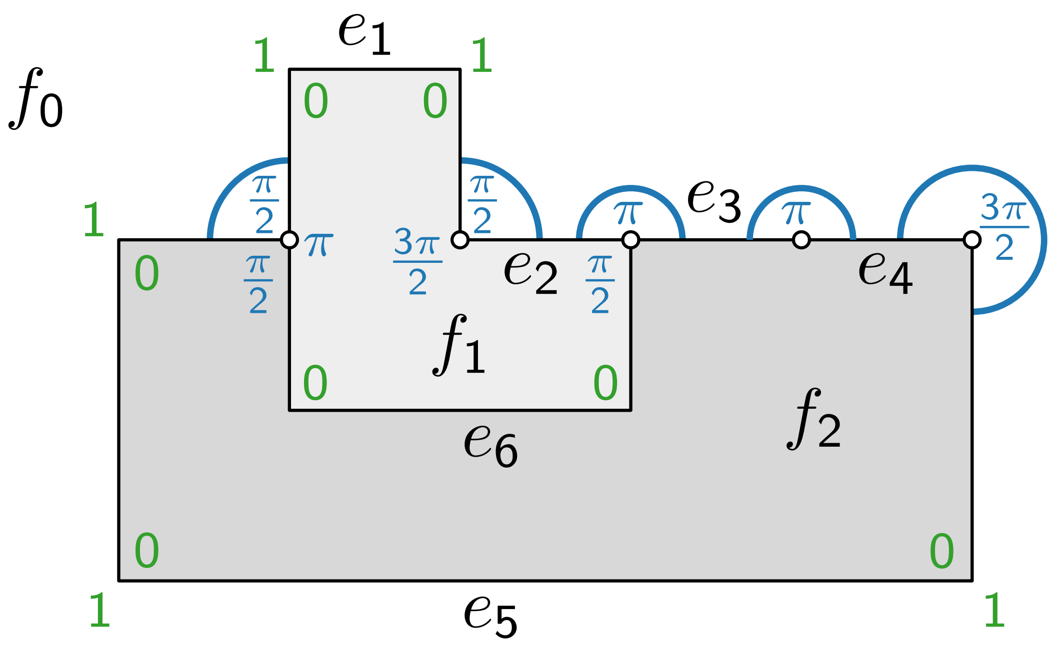




Orthogonal Representation – Example

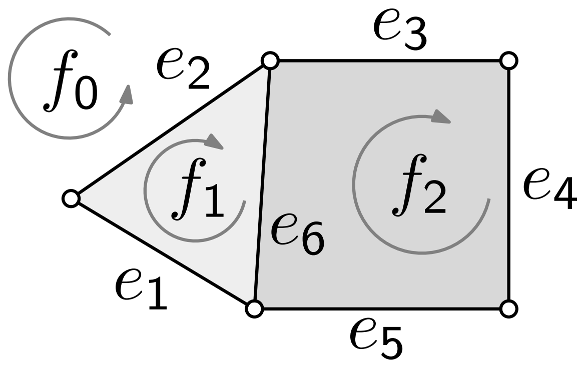
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

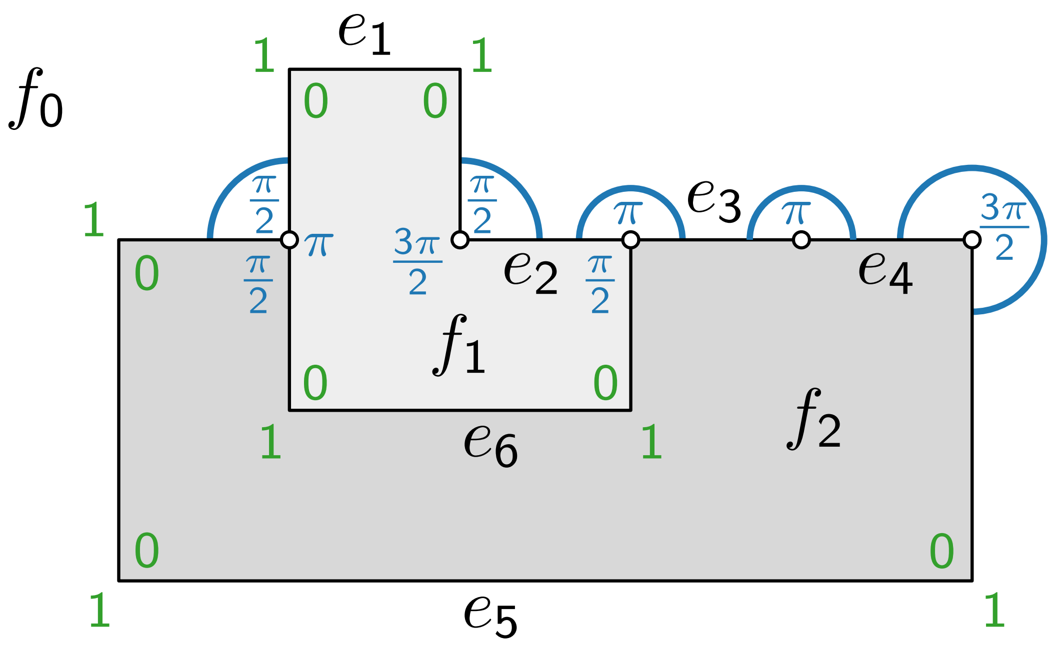




Orthogonal Representation – Example

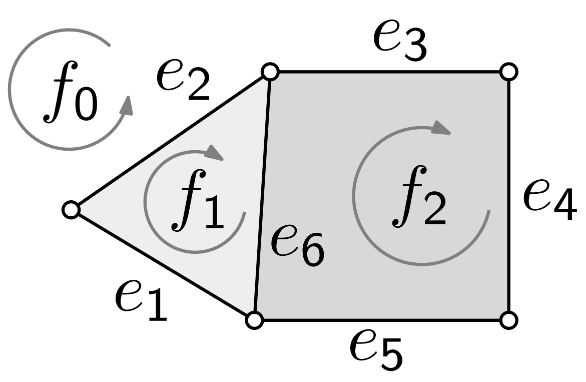
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

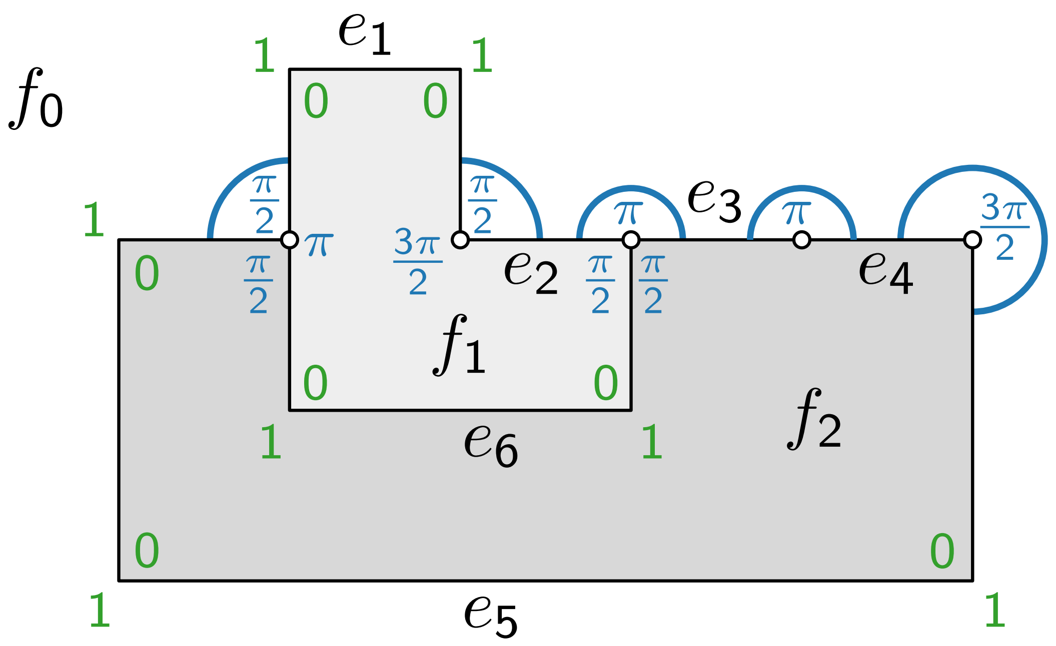




Orthogonal Representation – Example

$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$



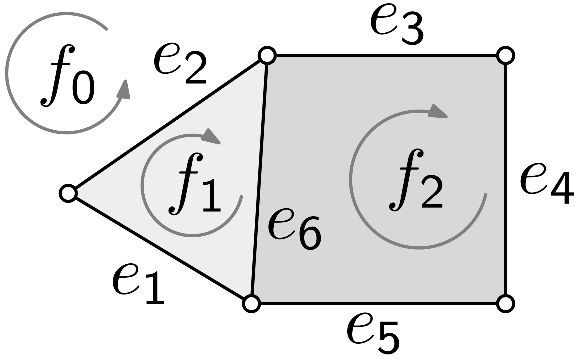


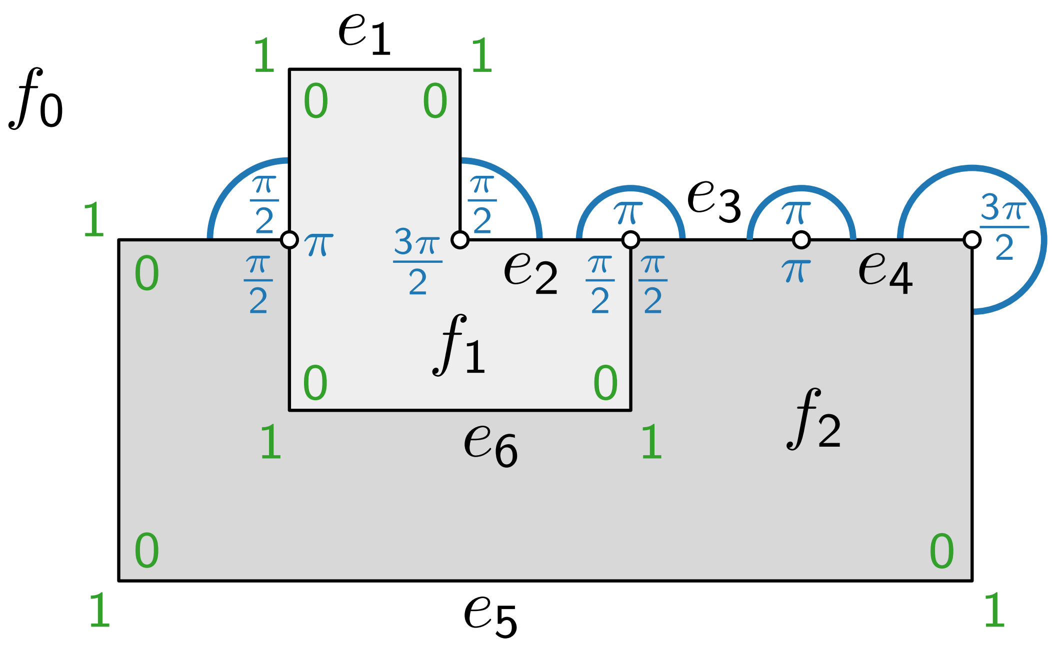
Orthogonal Representation – Example

$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$

$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$

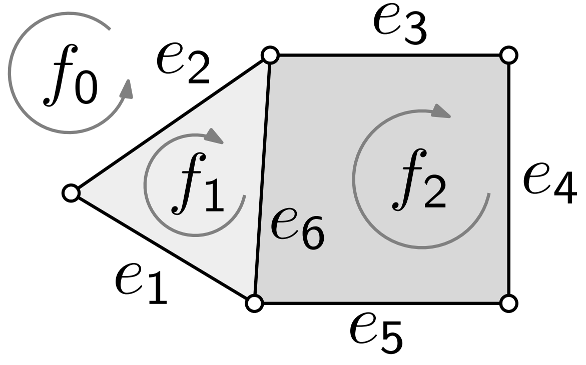
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

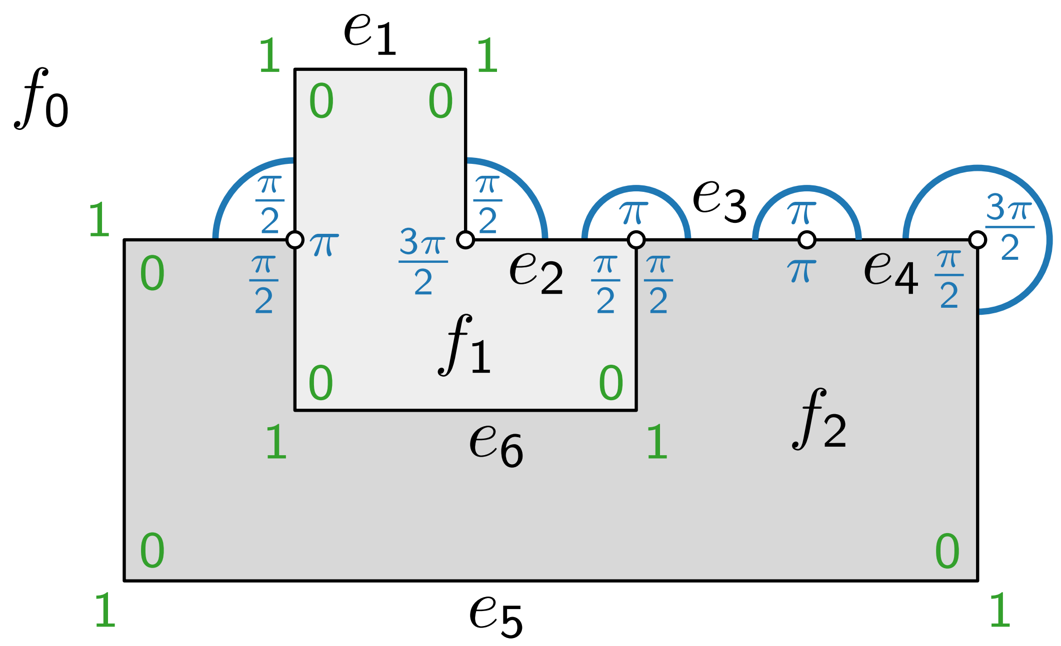




Orthogonal Representation – Example

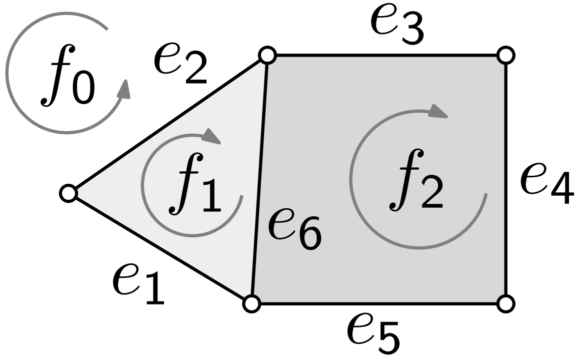
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

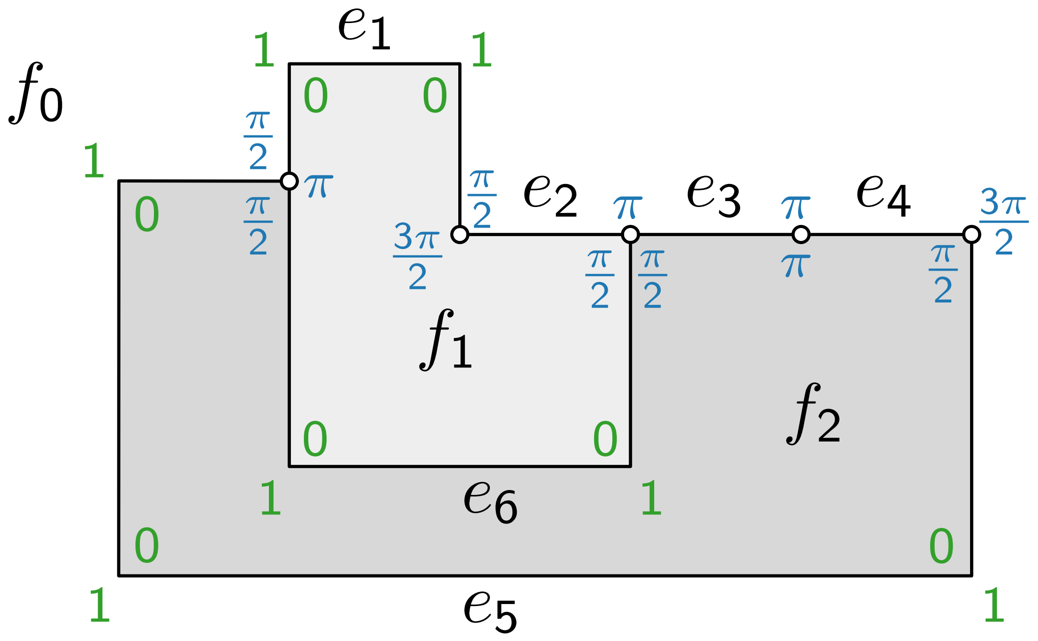




Orthogonal Representation – Example

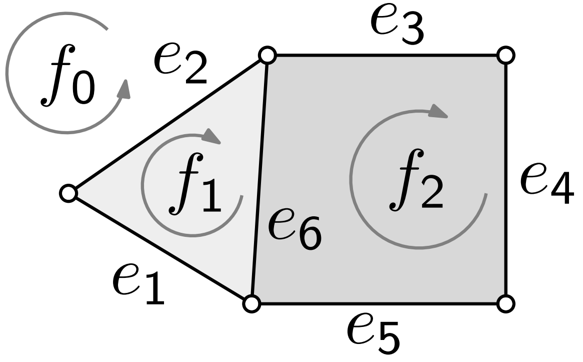
$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

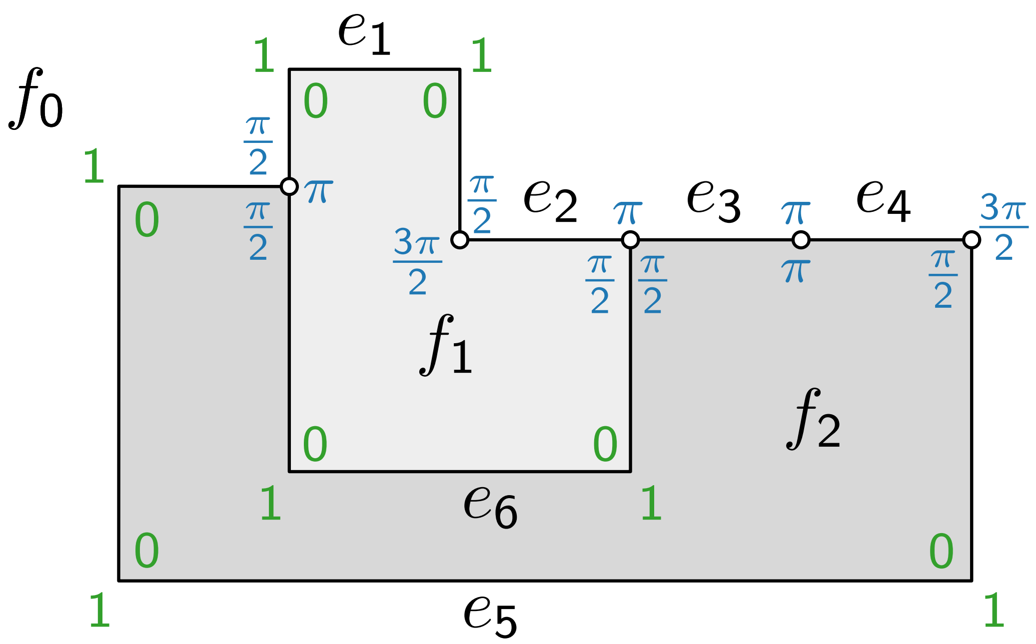




Orthogonal Representation – Example

$$H(f_0) = ((e_1, 11, \tfrac{\pi}{2}), (e_5, 111, \tfrac{3\pi}{2}), (e_4, \emptyset, \pi), (e_3, \emptyset, \pi), (e_2, \emptyset, \tfrac{\pi}{2}))$$
$$H(f_1) = ((e_1, 00, \tfrac{3\pi}{2}), (e_2, \emptyset, \tfrac{\pi}{2}), (e_6, 00, \pi))$$
$$H(f_2) = ((e_5, 000, \tfrac{\pi}{2}), (e_6, 11, \tfrac{\pi}{2}), (e_3, \emptyset, \pi), (e_4, \emptyset, \tfrac{\pi}{2}))$$

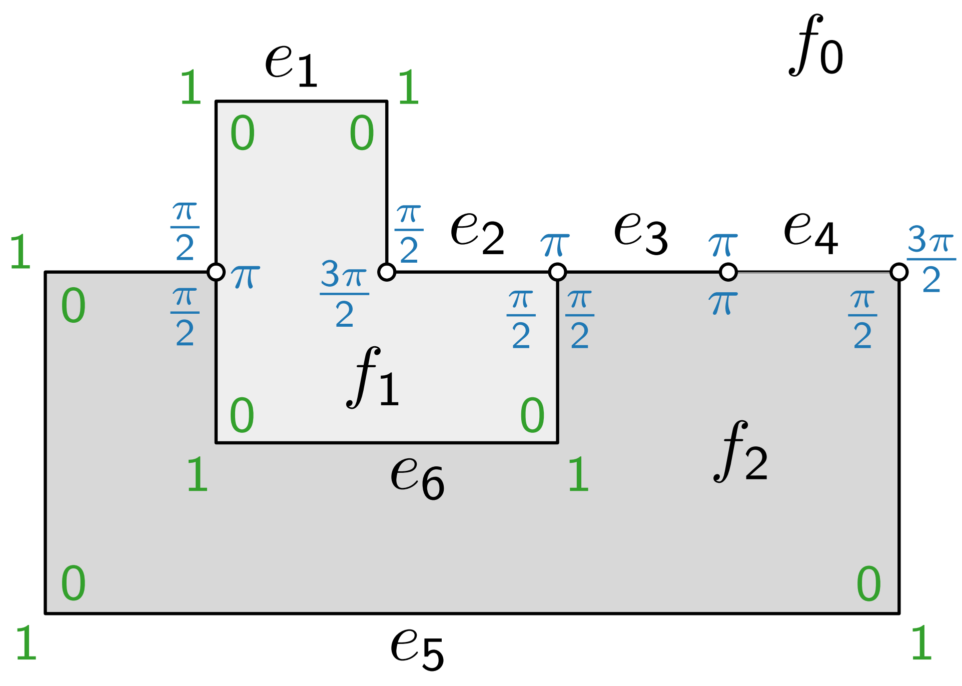




Coordinates are not fixed yet!

Correctness of an Orthogonal Representation

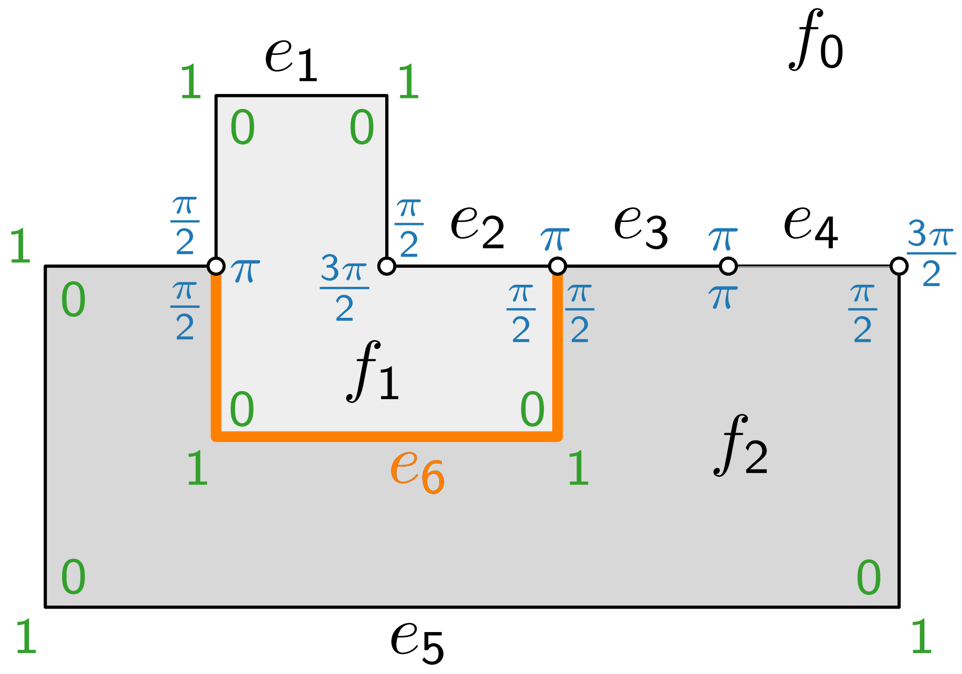
(H1) $H(G)$ corresponds to F, f_0.



Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

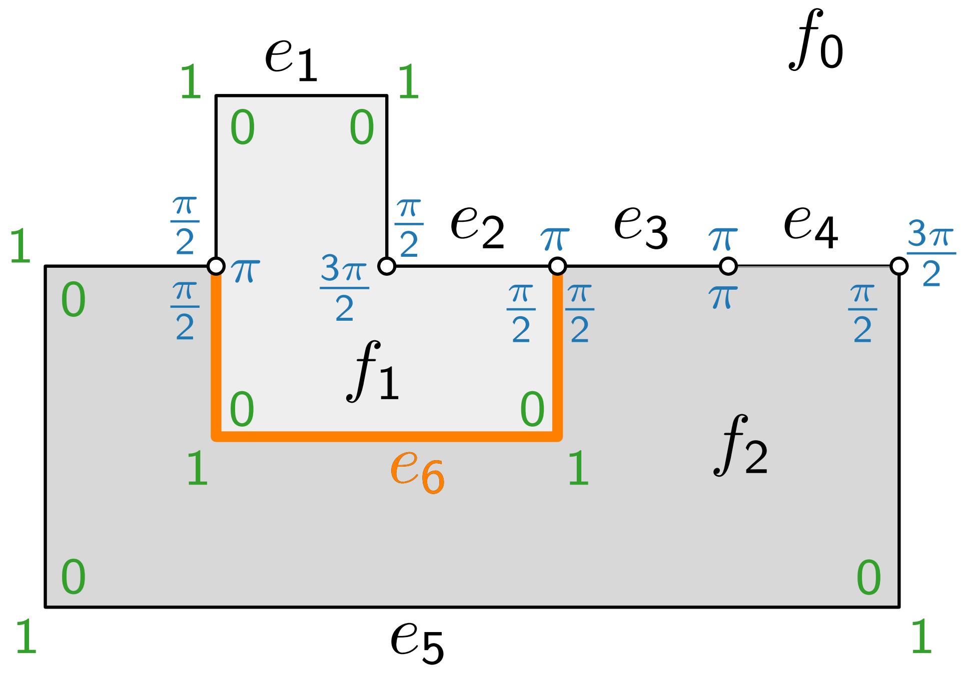
(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$



Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

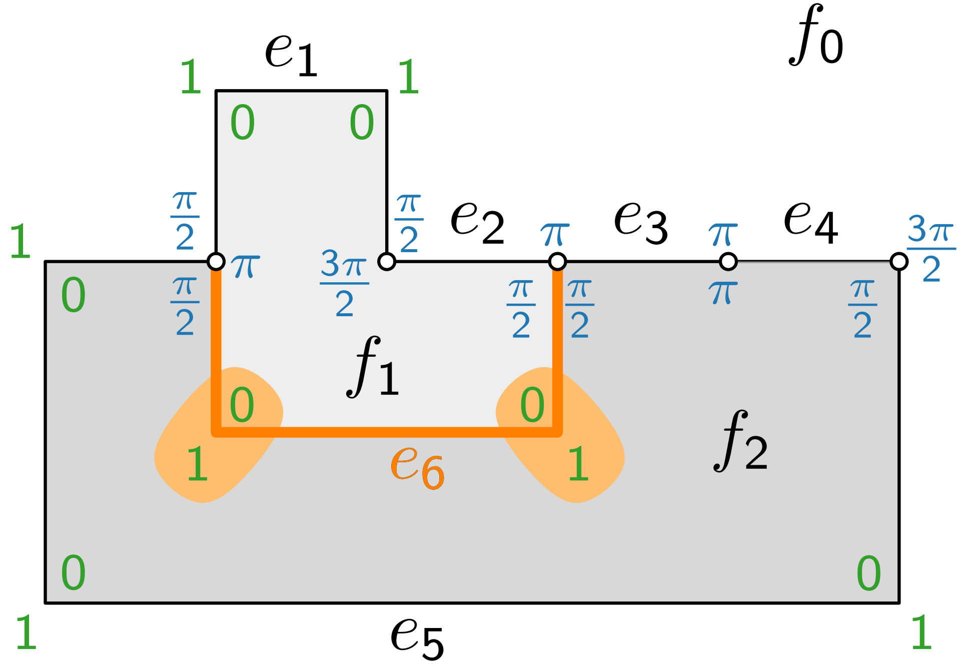
(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2



Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

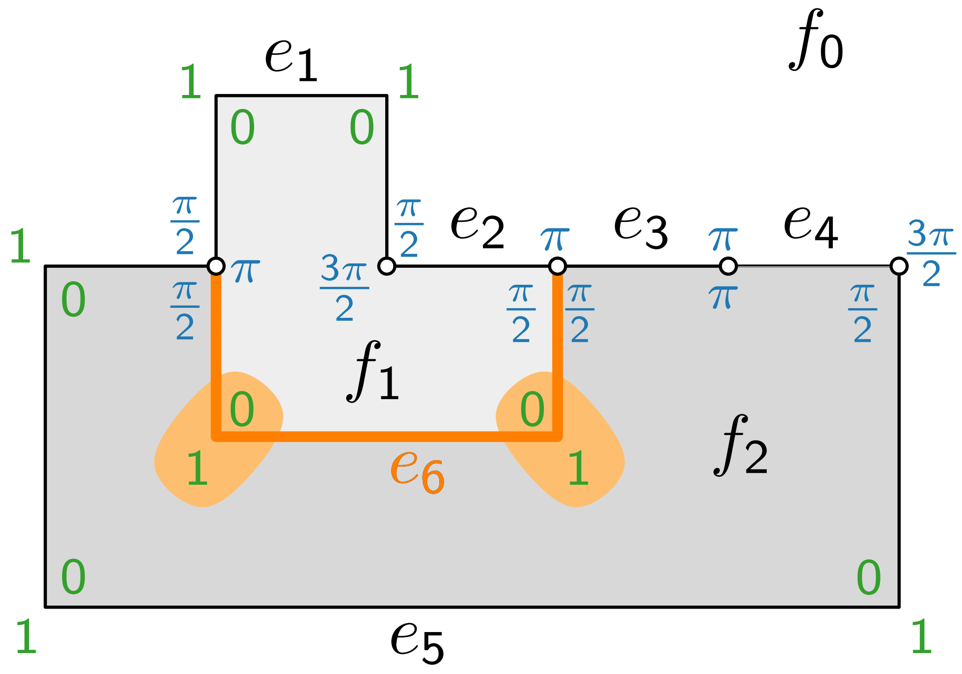


Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.

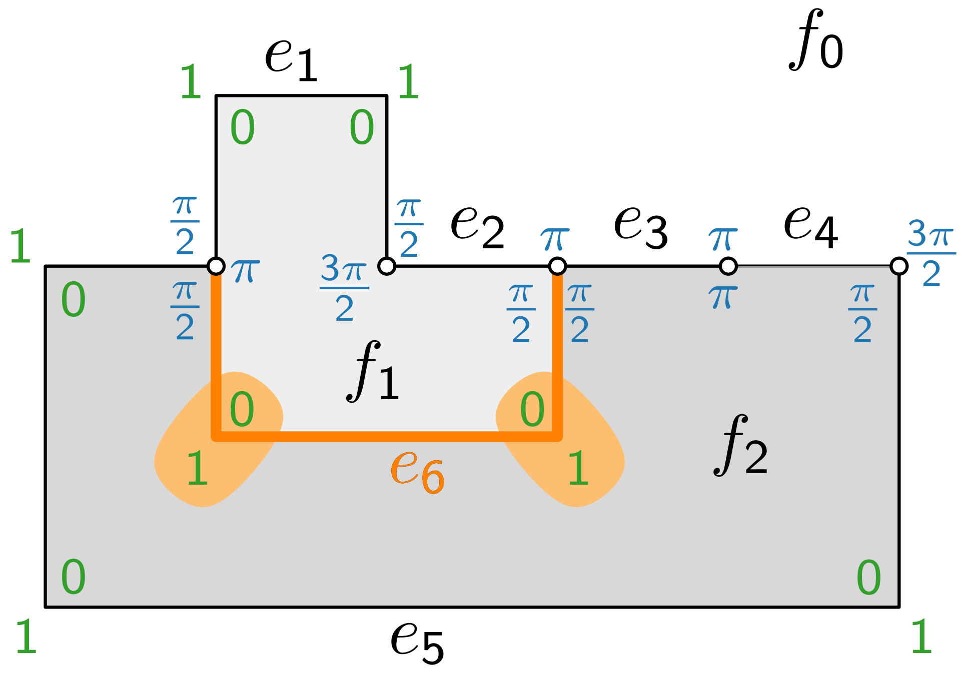


Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:



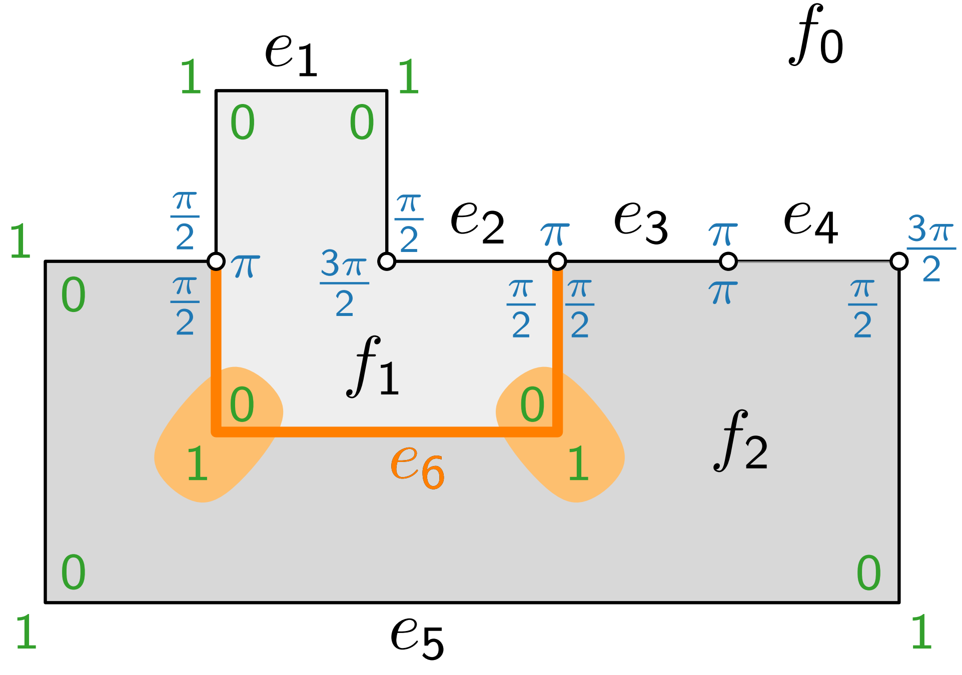
Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u,v), \delta_1, \alpha_1) \in H(f)$ and $((v,u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$



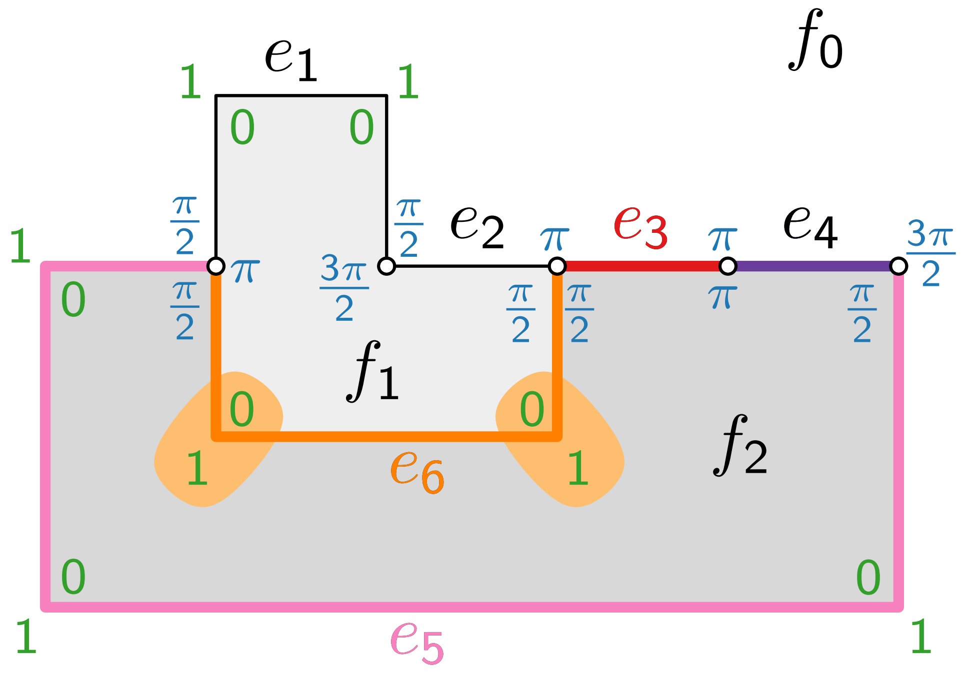
Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u,v), \delta_1, \alpha_1) \in H(f)$ and $((v,u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$



$$C((e_3, \emptyset, \pi)) = \quad - \quad - \quad + 2 =$$

$$C((e_4, \emptyset, \tfrac{\pi}{2})) = \quad - \quad - \quad + 2 =$$

$$C((e_5, 000, \tfrac{\pi}{2})) = \quad - \quad - \quad + 2 =$$

$$C((e_6, 11, \tfrac{\pi}{2})) = \quad - \quad - \quad + 2 =$$

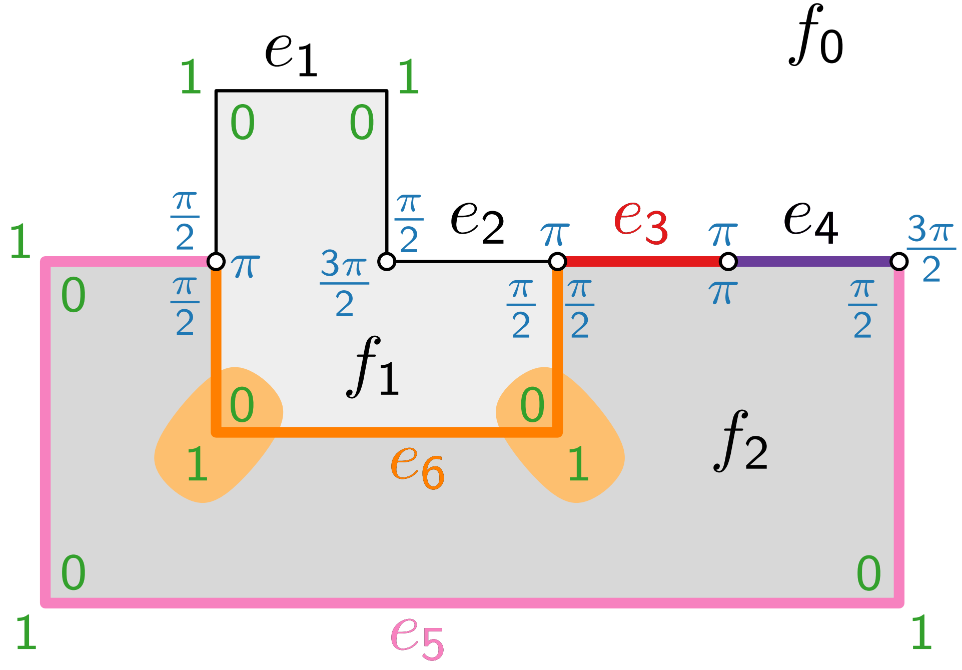
Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$



$C((e_3, \emptyset, \pi)) = 0 - \quad - \quad + 2 =$

$C((e_4, \emptyset, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

$C((e_5, 000, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

$C((e_6, 11, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

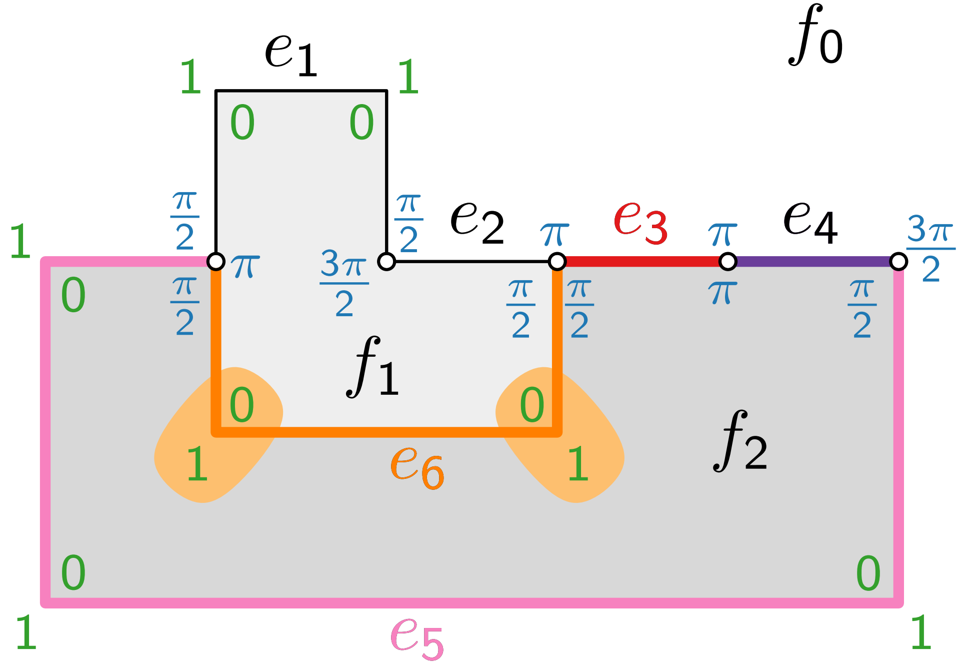
Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$



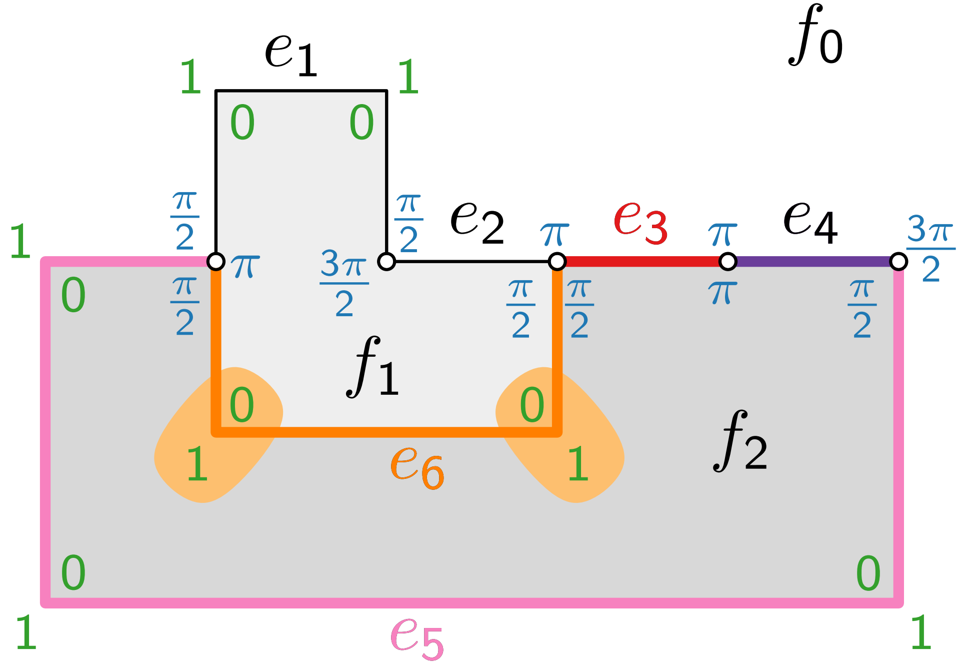
$C((e_3, \emptyset, \pi)) = 0 - 0 - \quad + 2 =$

$C((e_4, \emptyset, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

$C((e_5, 000, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

$C((e_6, 11, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

Correctness of an Orthogonal Representation



(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

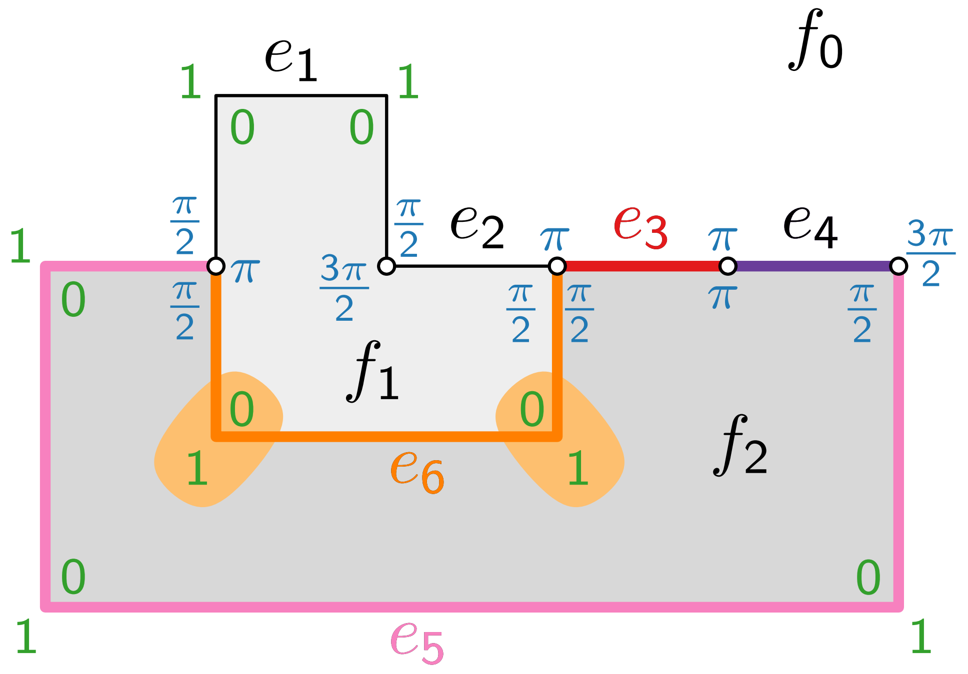
$C((e_3, \emptyset, \pi)) = 0 - 0 - 2 + 2 =$

$C((e_4, \emptyset, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

$C((e_5, 000, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

$C((e_6, 11, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

Correctness of an Orthogonal Representation



(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

$$C((e_3, \emptyset, \pi)) = 0 - 0 - 2 + 2 = 0$$

$$C((e_4, \emptyset, \tfrac{\pi}{2})) = \quad - \quad - \quad + 2 =$$

$$C((e_5, 000, \tfrac{\pi}{2})) = \quad - \quad - \quad + 2 =$$

$$C((e_6, 11, \tfrac{\pi}{2})) = \quad - \quad - \quad + 2 =$$

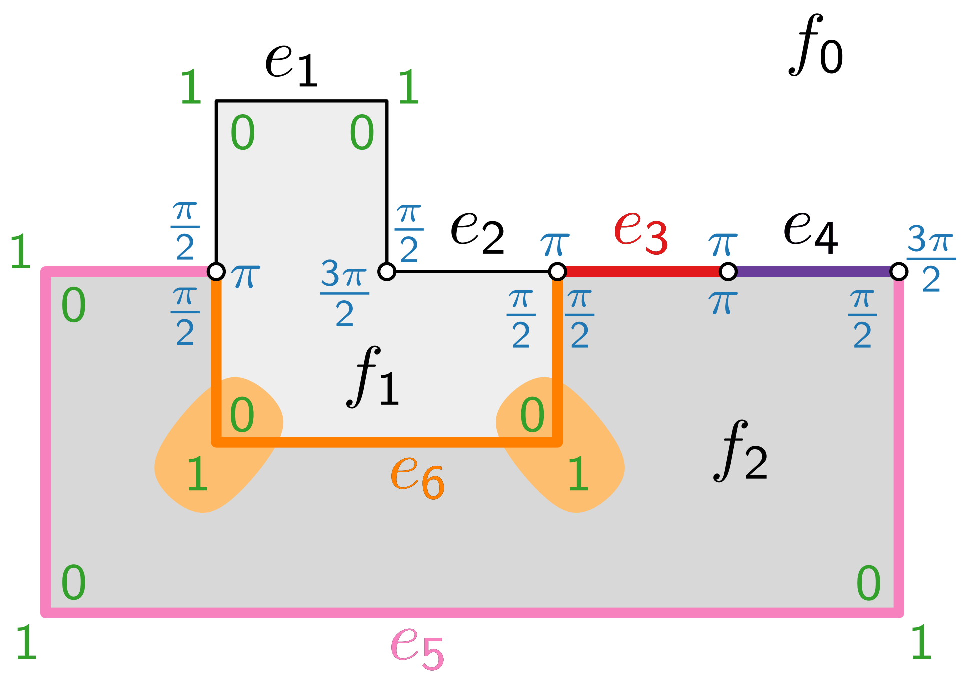
Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u,v), \delta_1, \alpha_1) \in H(f)$ and $((v,u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$



$C((e_3, \emptyset, \pi)) = 0 - 0 - 2 + 2 = 0$

$C((e_4, \emptyset, \frac{\pi}{2})) = 0 - 0 - 1 + 2 =$

$C((e_5, 000, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

$C((e_6, 11, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

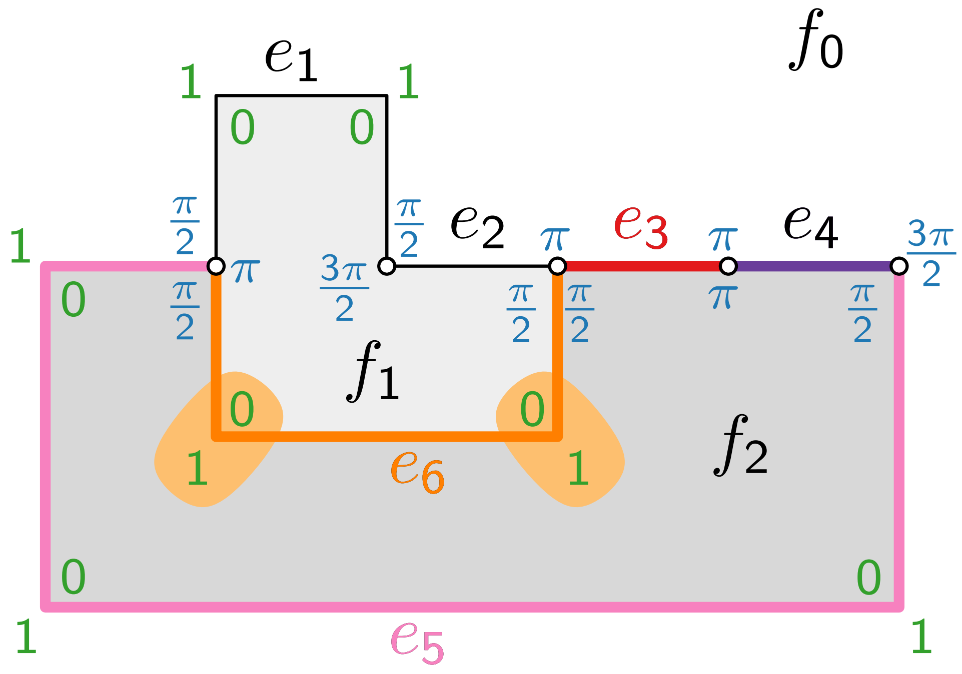
Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u,v), \delta_1, \alpha_1) \in H(f)$ and $((v,u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$



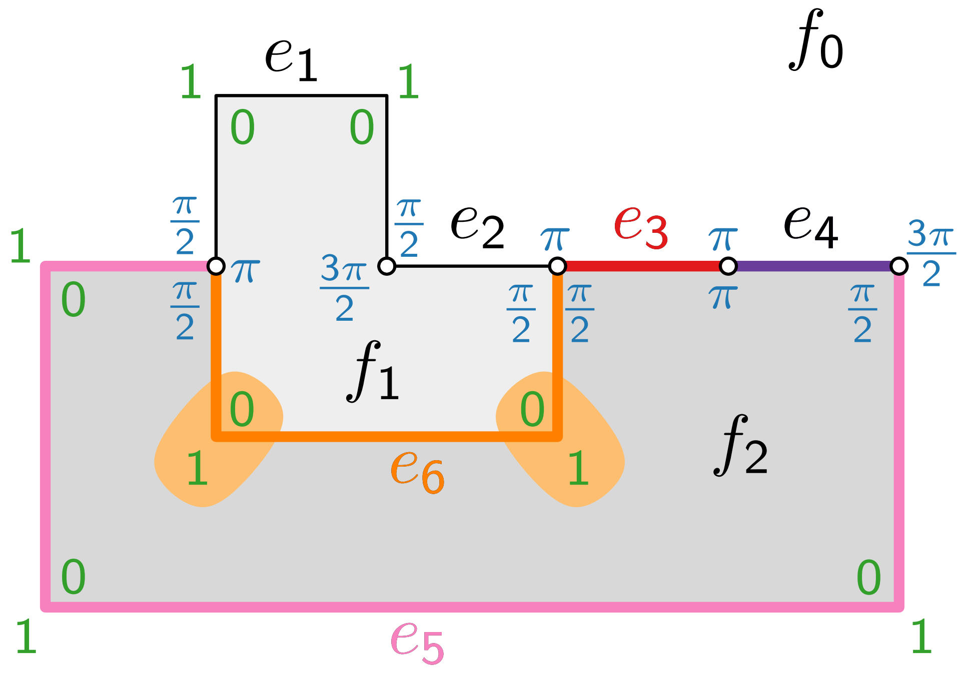
$C((e_3, \emptyset, \pi)) = 0 - 0 - 2 + 2 = 0$

$C((e_4, \emptyset, \frac{\pi}{2})) = 0 - 0 - 1 + 2 = 1$

$C((e_5, 000, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

$C((e_6, 11, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

Correctness of an Orthogonal Representation



(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

$$C((e_3, \emptyset, \pi)) = 0 - 0 - 2 + 2 = 0$$

$$C((e_4, \emptyset, \tfrac{\pi}{2})) = 0 - 0 - 1 + 2 = 1$$

$$C((e_5, 000, \tfrac{\pi}{2})) = 3 - 0 - \quad + 2 =$$

$$C((e_6, 11, \tfrac{\pi}{2})) = \quad - \quad - \quad + 2 =$$

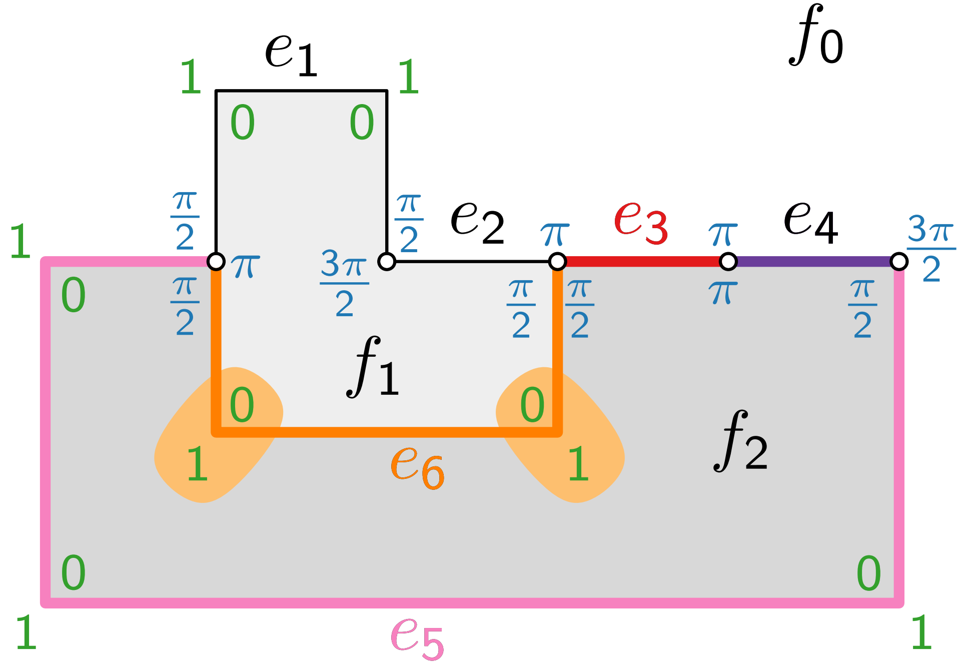
Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$



$$C((e_3, \emptyset, \pi)) = 0 - 0 - 2 + 2 = 0$$

$$C((e_4, \emptyset, \tfrac{\pi}{2})) = 0 - 0 - 1 + 2 = 1$$

$$C((e_5, 000, \tfrac{\pi}{2})) = 3 - 0 - 1 + 2 =$$

$$C((e_6, 11, \tfrac{\pi}{2})) = \quad - \quad - \quad + 2 =$$

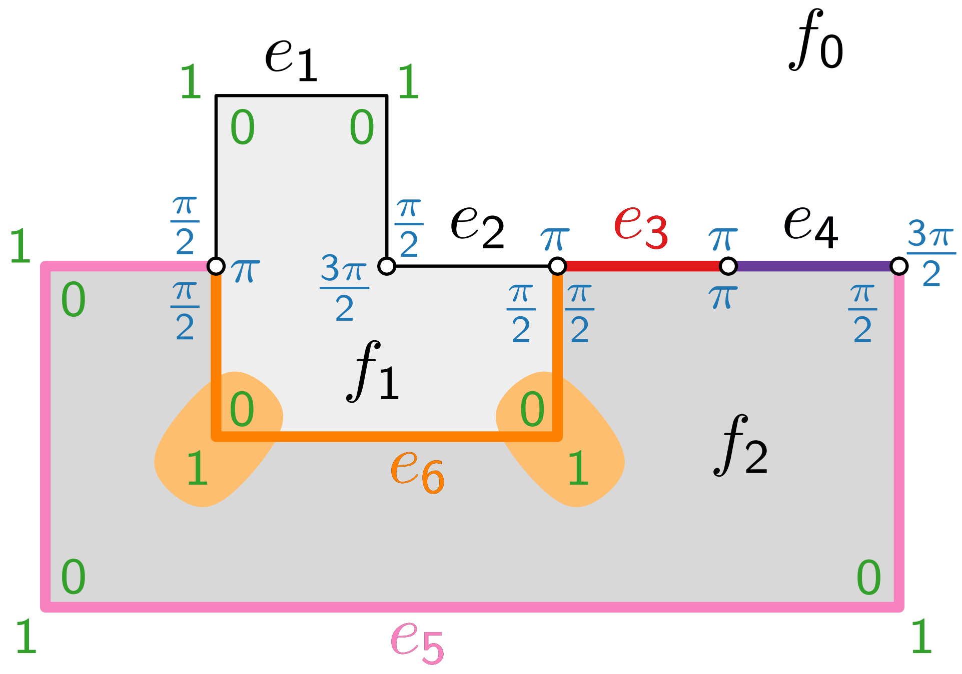
Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$



$C((e_3, \emptyset, \pi)) = 0 - 0 - 2 + 2 = 0$

$C((e_4, \emptyset, \frac{\pi}{2})) = 0 - 0 - 1 + 2 = 1$

$C((e_5, 000, \frac{\pi}{2})) = 3 - 0 - 1 + 2 = 4$

$C((e_6, 11, \frac{\pi}{2})) = \quad - \quad - \quad + 2 =$

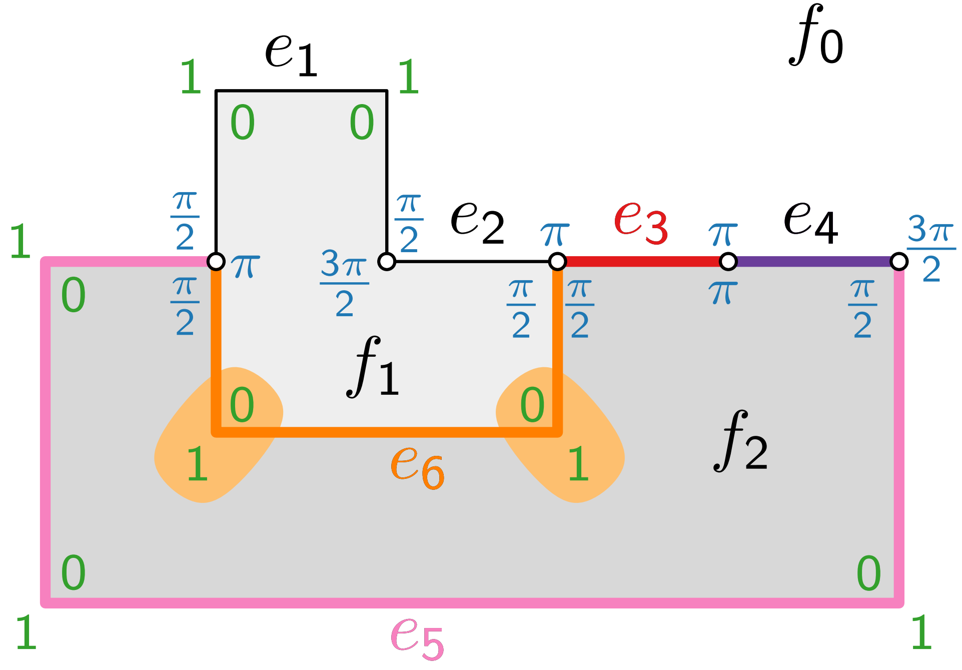
Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha / \frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$



$$C((e_3, \emptyset, \pi)) = 0 - 0 - 2 + 2 = 0$$

$$C((e_4, \emptyset, \tfrac{\pi}{2})) = 0 - 0 - 1 + 2 = 1$$

$$C((e_5, 000, \tfrac{\pi}{2})) = 3 - 0 - 1 + 2 = 4$$

$$C((e_6, 11, \tfrac{\pi}{2})) = 0 - 2 - 1 + 2 =$$

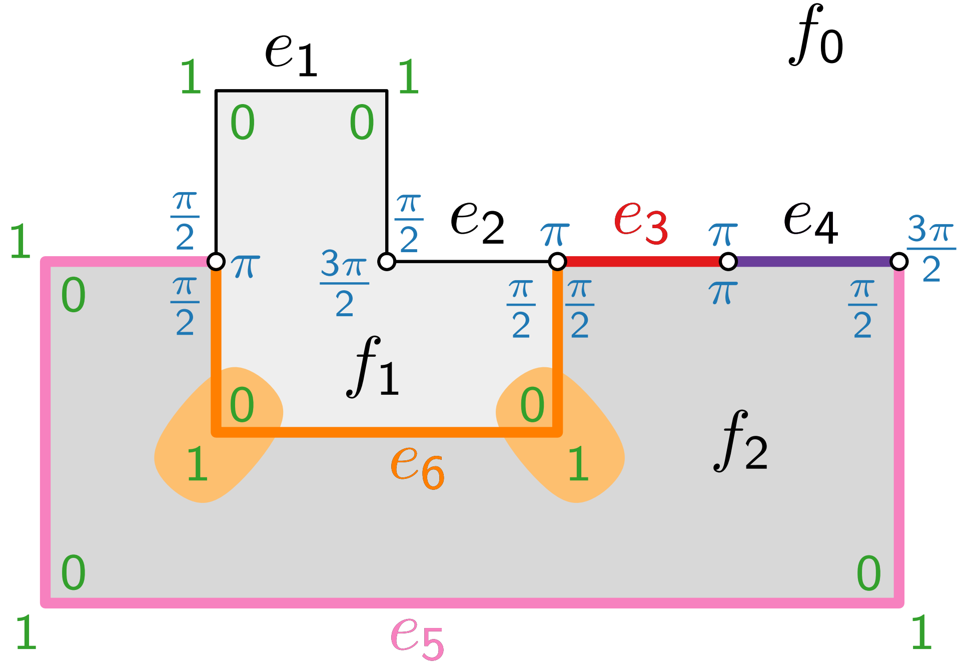
Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$



$$C((e_3, \emptyset, \pi)) = 0 - 0 - 2 + 2 = 0$$

$$C((e_4, \emptyset, \tfrac{\pi}{2})) = 0 - 0 - 1 + 2 = 1$$

$$C((e_5, 000, \tfrac{\pi}{2})) = 3 - 0 - 1 + 2 = 4$$

$$C((e_6, 11, \tfrac{\pi}{2})) = 0 - 2 - 1 + 2 = -1$$

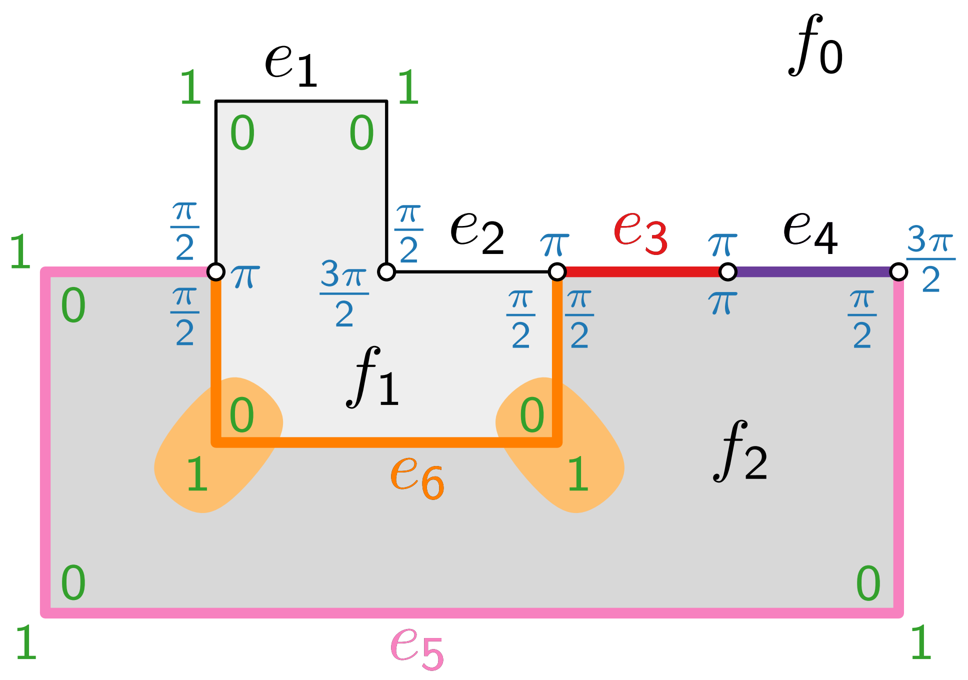
Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u,v), \delta_1, \alpha_1) \in H(f)$ and $((v,u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha / \frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$



$$C((e_3, \emptyset, \pi)) = 0 - 0 - 2 + 2 = 0$$

$$C((e_4, \emptyset, \tfrac{\pi}{2})) = 0 - 0 - 1 + 2 = 1$$

$$C((e_5, 000, \tfrac{\pi}{2})) = 3 - 0 - 1 + 2 = 4$$

$$C((e_6, 11, \tfrac{\pi}{2})) = 0 - 2 - 1 + 2 = -1$$

$$\sum_{r \in H(f_2)} C(r) =$$

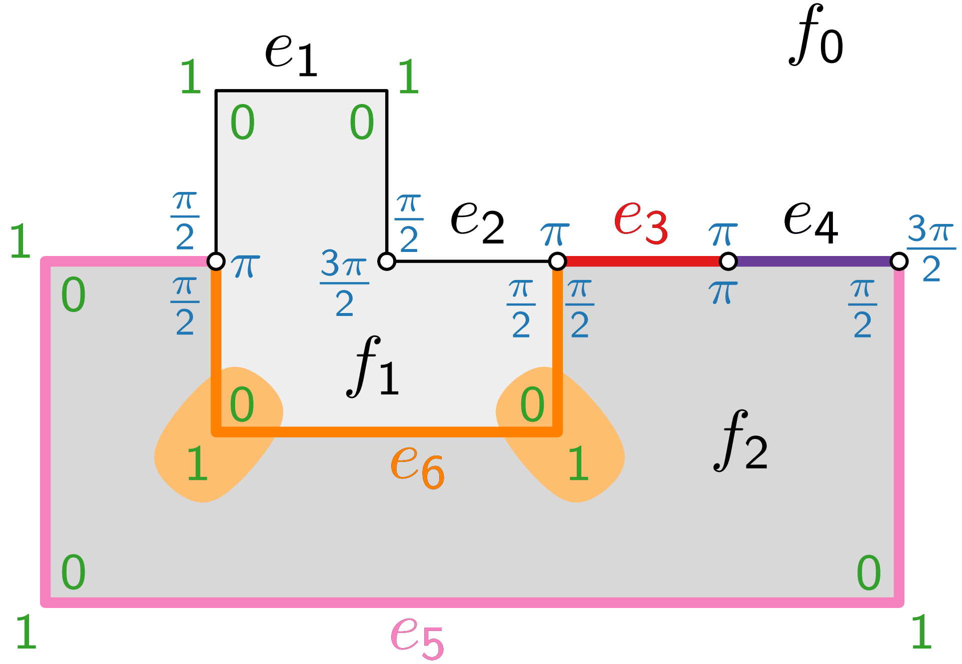
Correctness of an Orthogonal Representation

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u,v\}$ shared by faces f and g with $((u,v),\delta_1,\alpha_1) \in H(f)$ and $((v,u),\delta_2,\alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e,\delta,\alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$



$$C((e_3,\emptyset,\pi)) = 0 - 0 - 2 + 2 = 0$$

$$C((e_4,\emptyset,\tfrac{\pi}{2})) = 0 - 0 - 1 + 2 = 1$$

$$C((e_5,000,\tfrac{\pi}{2})) = 3 - 0 - 1 + 2 = 4$$

$$C((e_6,11,\tfrac{\pi}{2})) = 0 - 2 - 1 + 2 = -1$$

$$\sum_{r \in H(f_2)} C(r) = +4$$

Correctness of an Orthogonal Representation

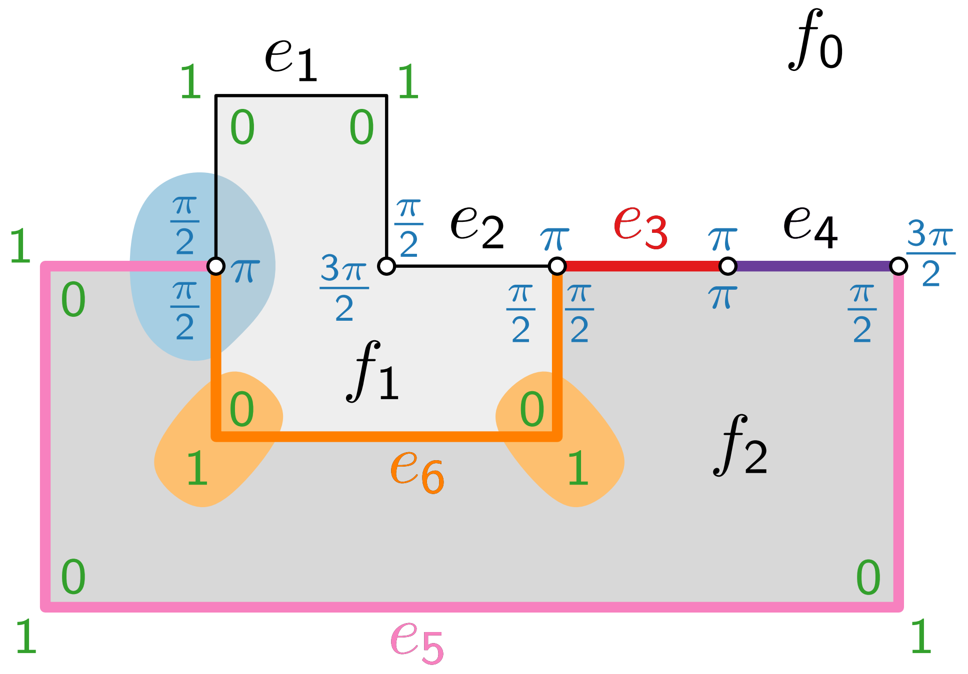
(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u, v), \delta_1, \alpha_1) \in H(f)$ and $((v, u), \delta_2, \alpha_2) \in H(g)$, the sequence δ_1 is like δ_2, but reversed and inverted.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ, and let $r = (e, \delta, \alpha)$.
Let $C(r) := |\delta|_0 - |\delta|_1 - \alpha/\frac{\pi}{2} + 2$.
For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.



$$C((e_3, \emptyset, \pi)) = 0 - 0 - 2 + 2 = 0$$

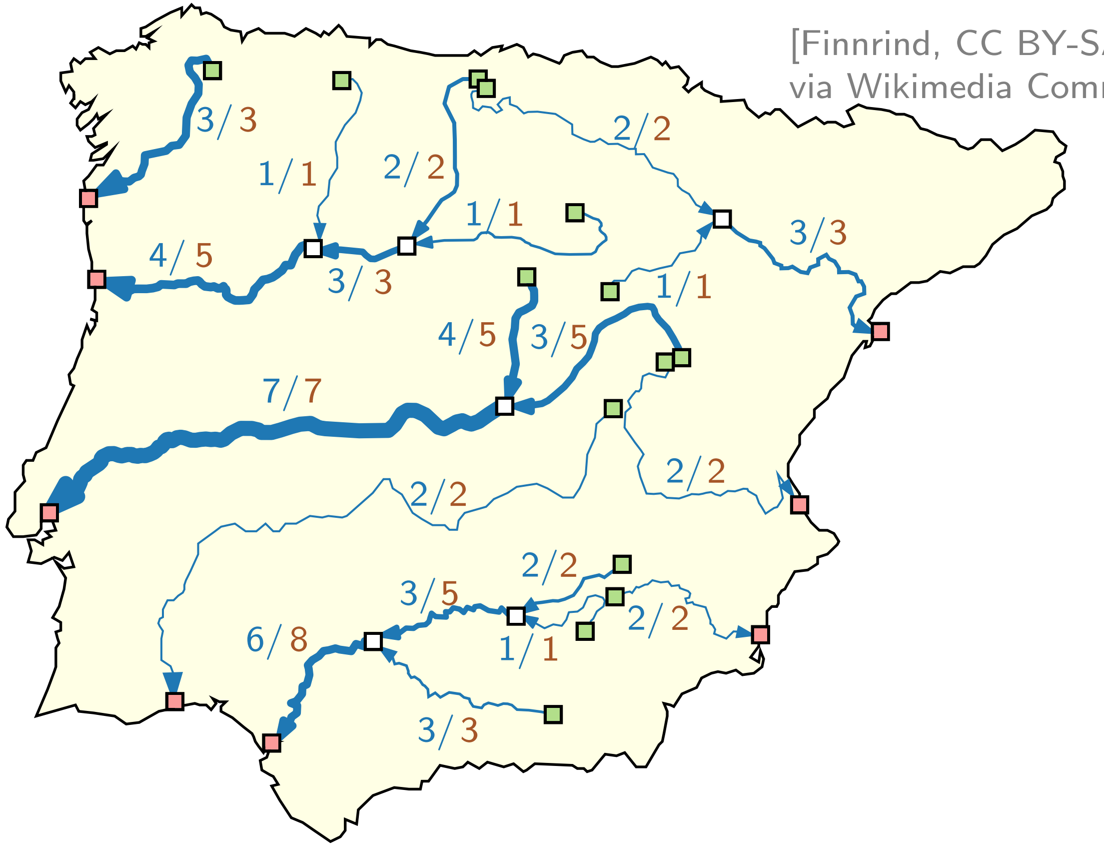
$$C((e_4, \emptyset, \tfrac{\pi}{2})) = 0 - 0 - 1 + 2 = 1$$

$$C((e_5, 000, \tfrac{\pi}{2})) = 3 - 0 - 1 + 2 = 4$$

$$C((e_6, 11, \tfrac{\pi}{2})) = 0 - 2 - 1 + 2 = -1$$

$$\sum_{r \in H(f_2)} C(r) = +4$$

Reminder: s–t Flow Networks



Flow network $(G; S, T; u)$ with

- directed graph G
- *sources* $S \subseteq V(G)$, *sinks* $T \subseteq V(G)$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

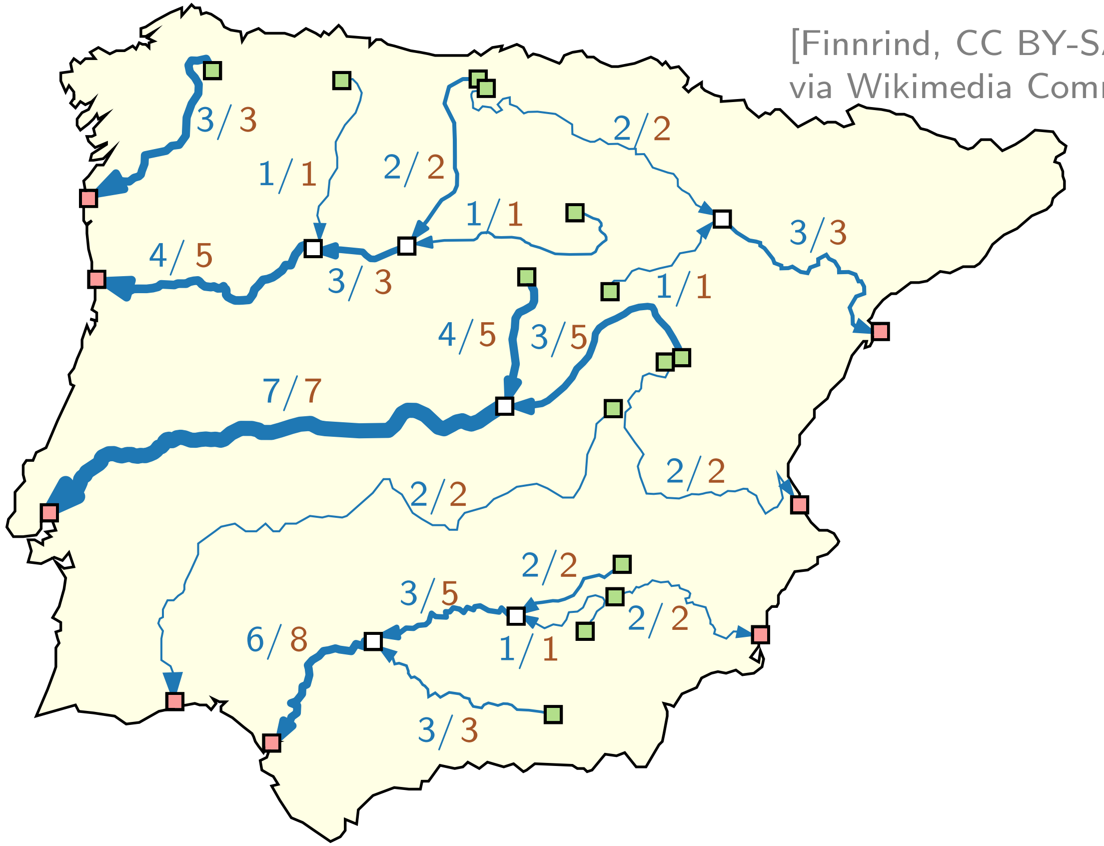
A function $X\colon E(G) \to \mathbb{R}_0^+$ is called S–T **flow** if:

$$0 \leq X(i,j) \leq u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus (S \cup T)$$

A **maximum** S–T flow is an S–T flow where $\sum_{(i,j)\in E(G), i \in S} X(i,j) - \sum_{(j,i)\in E(G), i\in S} X(j,i)$ is maximized.

Reminder: s–t Flow Networks



Flow network $(G; s, t; u)$ with

- directed graph G
- *source* $s \in V(G)$, *sink* $t \in V(G)$
- edge *capacity* $u \colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

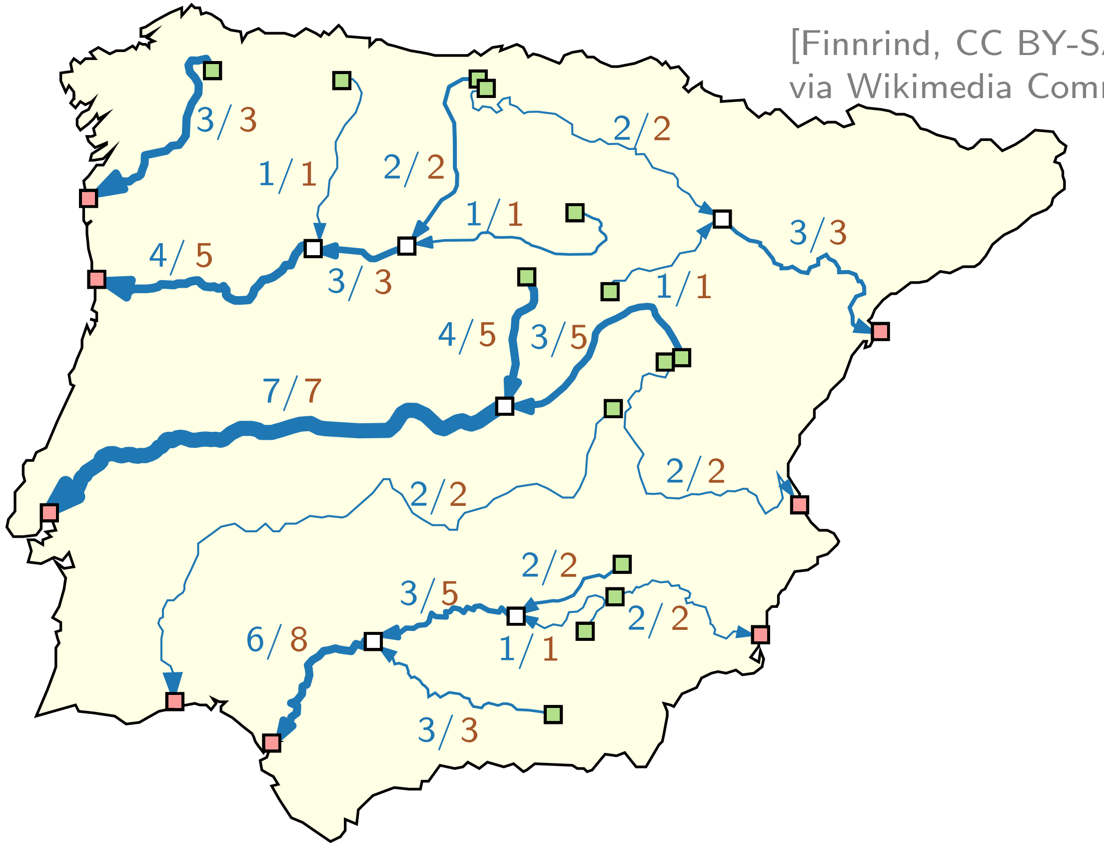
A function $X \colon E(G) \to \mathbb{R}_0^+$ is called S–T **flow** if:

$$0 \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus (S \cup T)$$

A **maximum** S–T flow is an S–T flow where $\sum_{(i,j)\in E(G), i \in S} X(i,j) - \sum_{(j,i)\in E(G), i\in S} X(j,i)$ is maximized.

Reminder: s–t Flow Networks



[Finnrind, CC BY-SA 3.0, via Wikimedia Commons]

Flow network $(G; s, t; u)$ with

- directed graph G
- *source* $s \in V(G)$, *sink* $t \in V(G)$
- edge *capacity* $u \colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

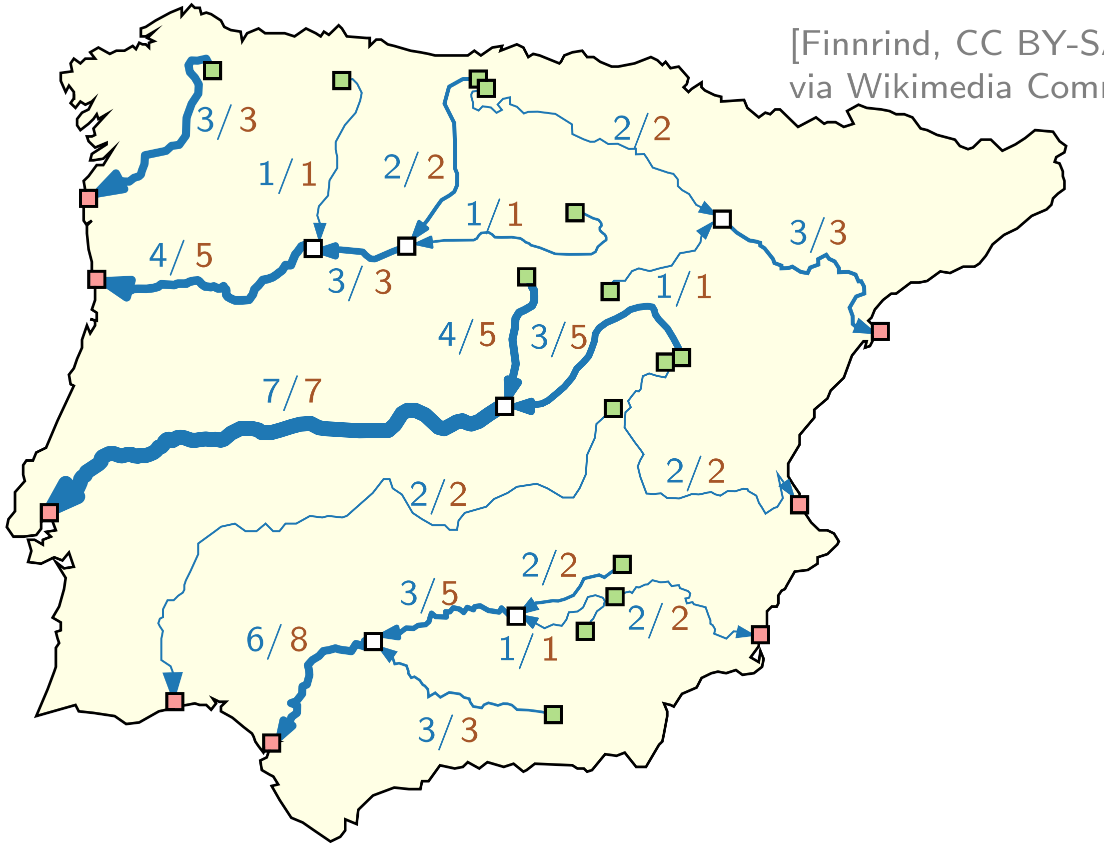
A function $X \colon E(G) \to \mathbb{R}_0^+$ is called s–t **flow** if:

$$0 \leq X(i,j) \leq u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j) \in E(G)} X(i,j) - \sum_{(j,i) \in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus \{s,t\}$$

A **maximum** S–T flow is an S–T flow where $\sum_{(i,j) \in E(G), i \in S} X(i,j) - \sum_{(j,i) \in E(G), i \in S} X(j,i)$ is maximized.

Reminder: s–t Flow Networks



Flow network $(G; s, t; u)$ with

- directed graph G
- *source* $s \in V(G)$, *sink* $t \in V(G)$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

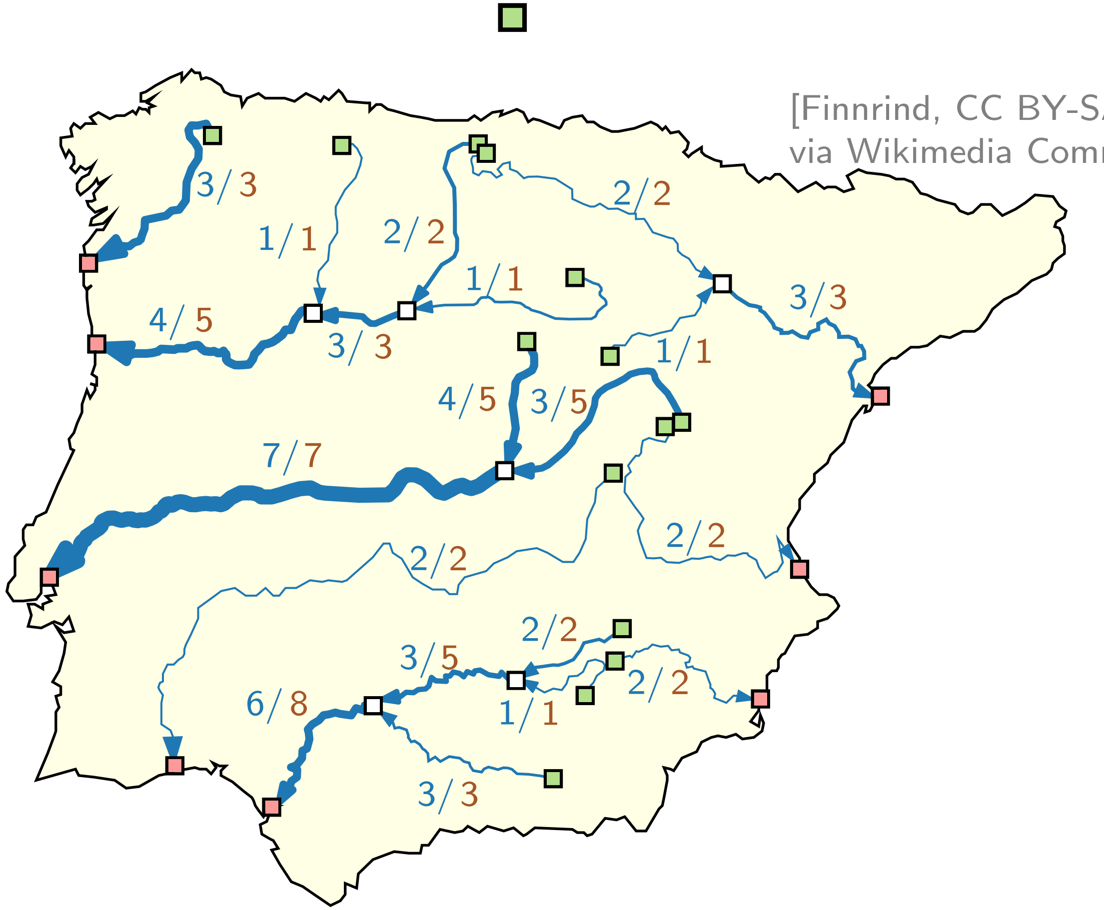
A function $X\colon E(G) \to \mathbb{R}_0^+$ is called s–t **flow** if:

$$0 \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus \{s,t\}$$

A **maximum** s–t flow is an s–t flow where $\sum_{(s,j)\in E(G)} X(s,j) - \sum_{(j,s)\in E(G)} X(j,s)$ is maximized.

Reminder: s–t Flow Networks



Flow network $(G; s, t; u)$ with

- directed graph G
- *source* $s \in V(G)$, *sink* $t \in V(G)$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

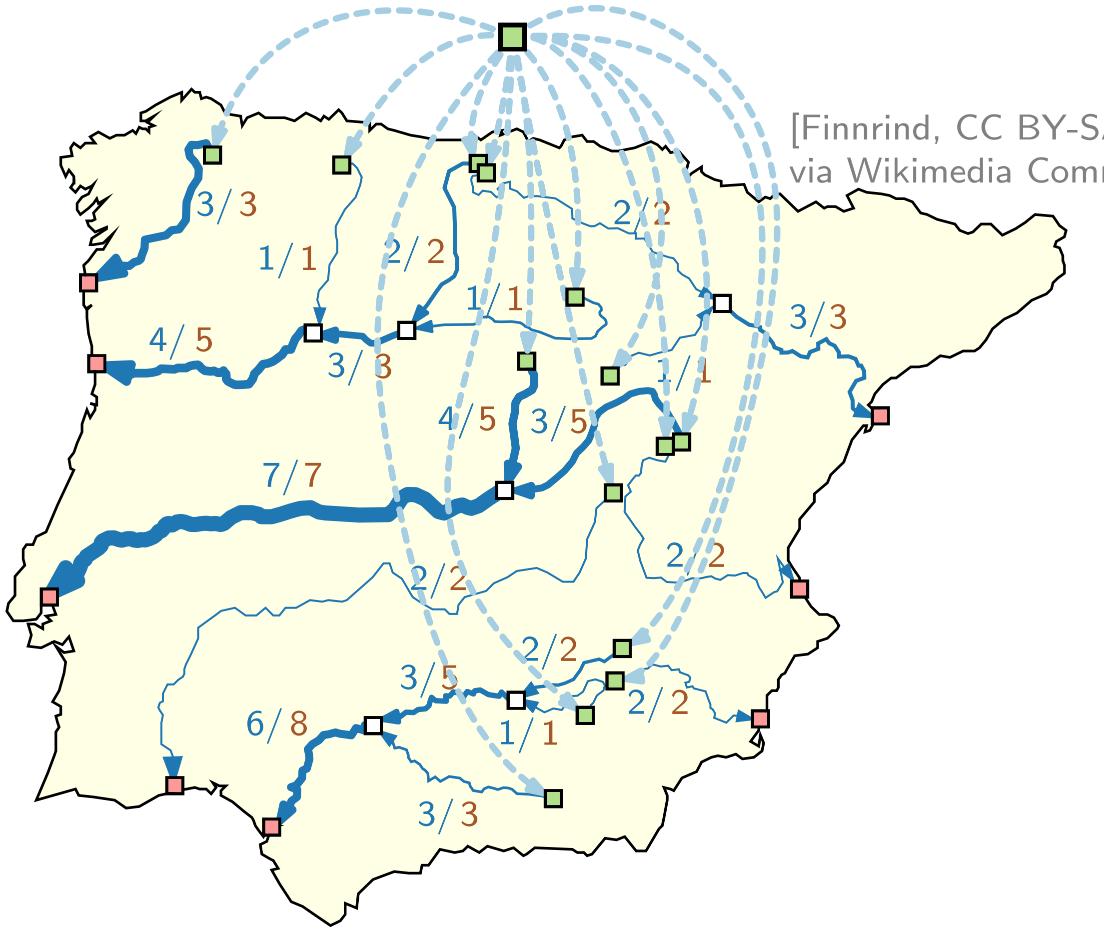
A function $X\colon E(G) \to \mathbb{R}_0^+$ is called s–t **flow** if:

$$0 \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus \{s,t\}$$

A **maximum** s–t flow is an s–t flow where $\sum_{(s,j)\in E(G)} X(s,j) - \sum_{(j,s)\in E(G)} X(j,s)$ is maximized.

Reminder: s–t Flow Networks



Flow network $(G; s, t; u)$ with

- directed graph G
- *source* $s \in V(G)$, *sink* $t \in V(G)$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

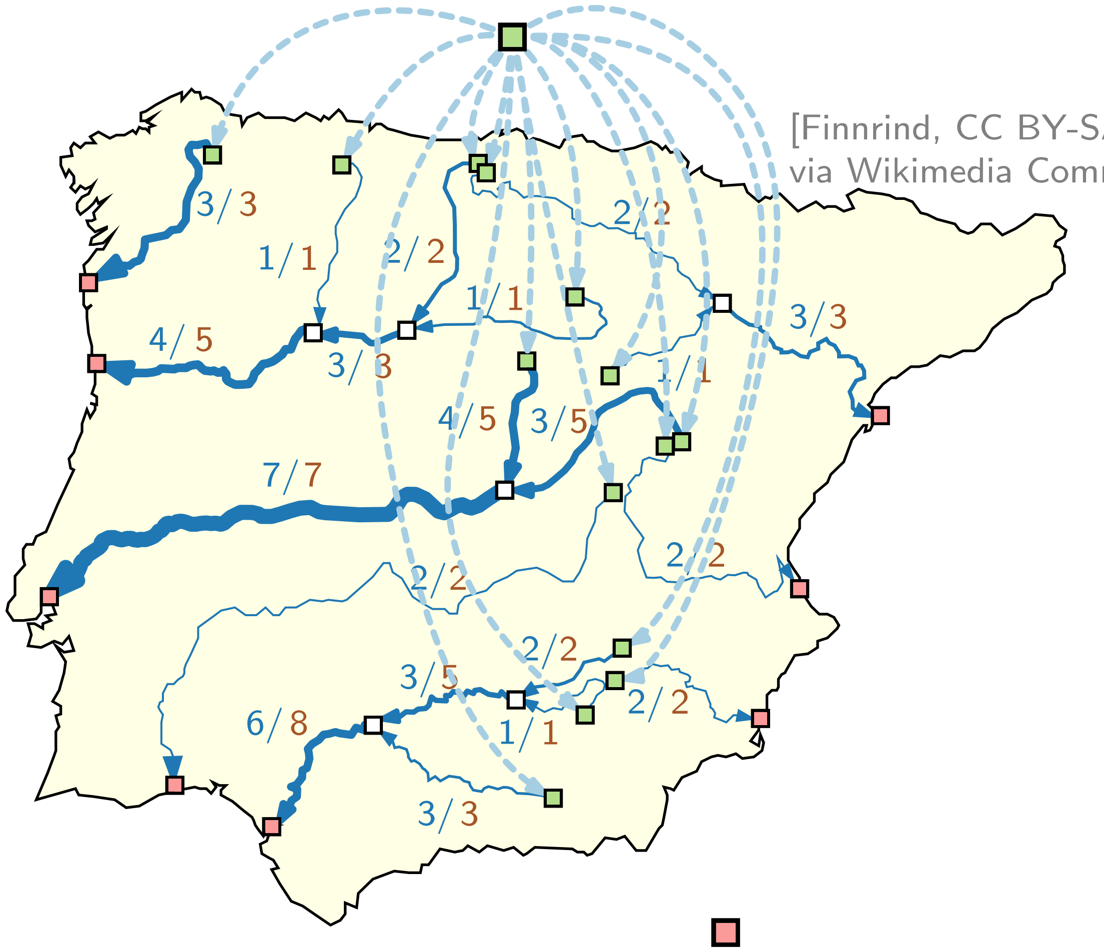
A function $X\colon E(G) \to \mathbb{R}_0^+$ is called s–t **flow** if:

$$0 \leq X(i,j) \leq u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus \{s,t\}$$

A **maximum** s–t flow is an s–t flow where $\sum_{(s,j)\in E(G)} X(s,j) - \sum_{(j,s)\in E(G)} X(j,s)$ is maximized.

Reminder: s–t Flow Networks



Flow network $(G; s, t; u)$ with

- directed graph G
- *source* $s \in V(G)$, *sink* $t \in V(G)$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

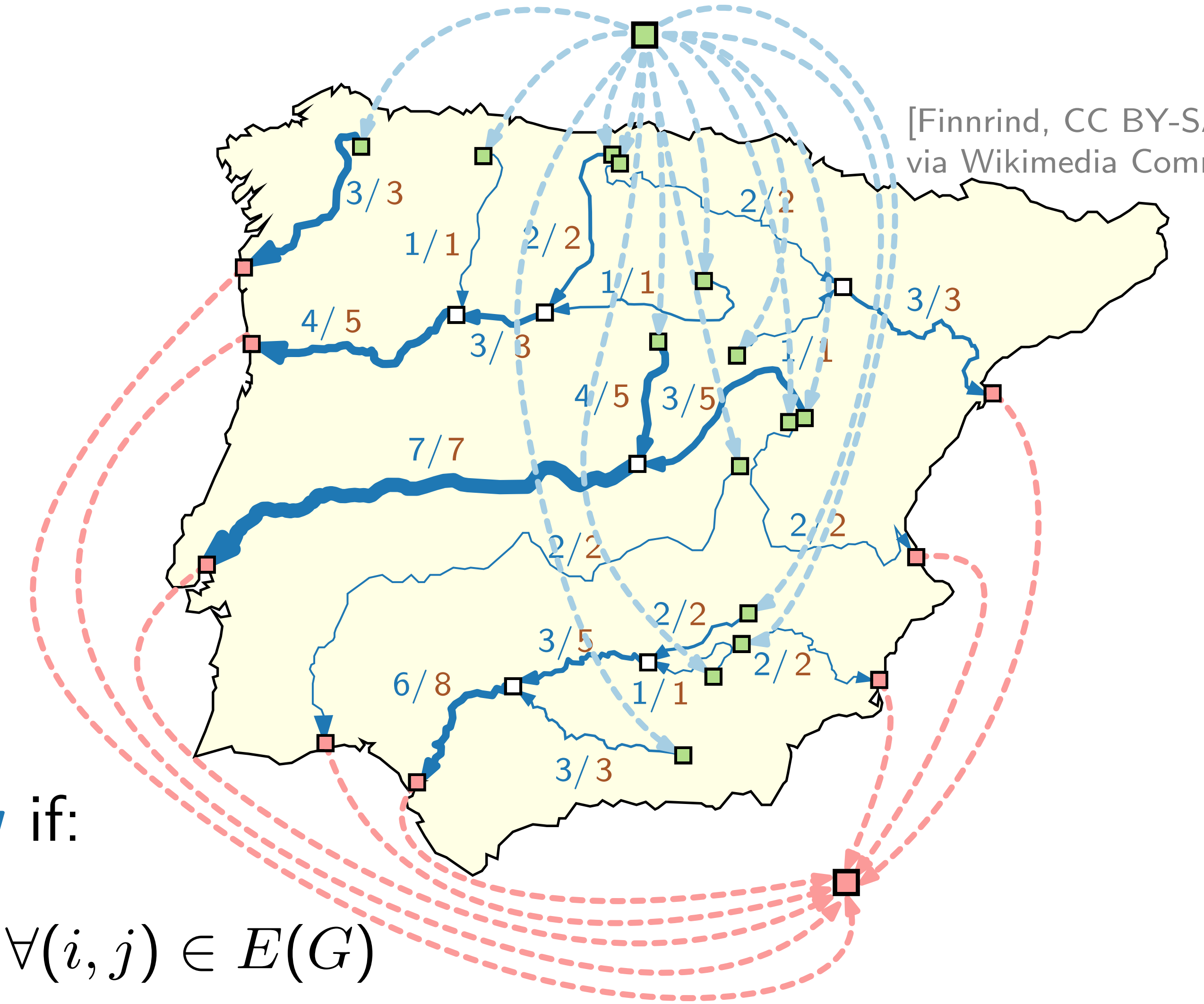
A function $X\colon E(G) \to \mathbb{R}_0^+$ is called s–t **flow** if:

$$0 \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus \{s,t\}$$

A **maximum** s–t flow is an s–t flow where $\sum_{(s,j)\in E(G)} X(s,j) - \sum_{(j,s)\in E(G)} X(j,s)$ is maximized.

Reminder: s–t Flow Networks



Flow network $(G; s, t; u)$ with

- directed graph G
- *source* $s \in V(G)$, *sink* $t \in V(G)$
- edge *capacity* $u \colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

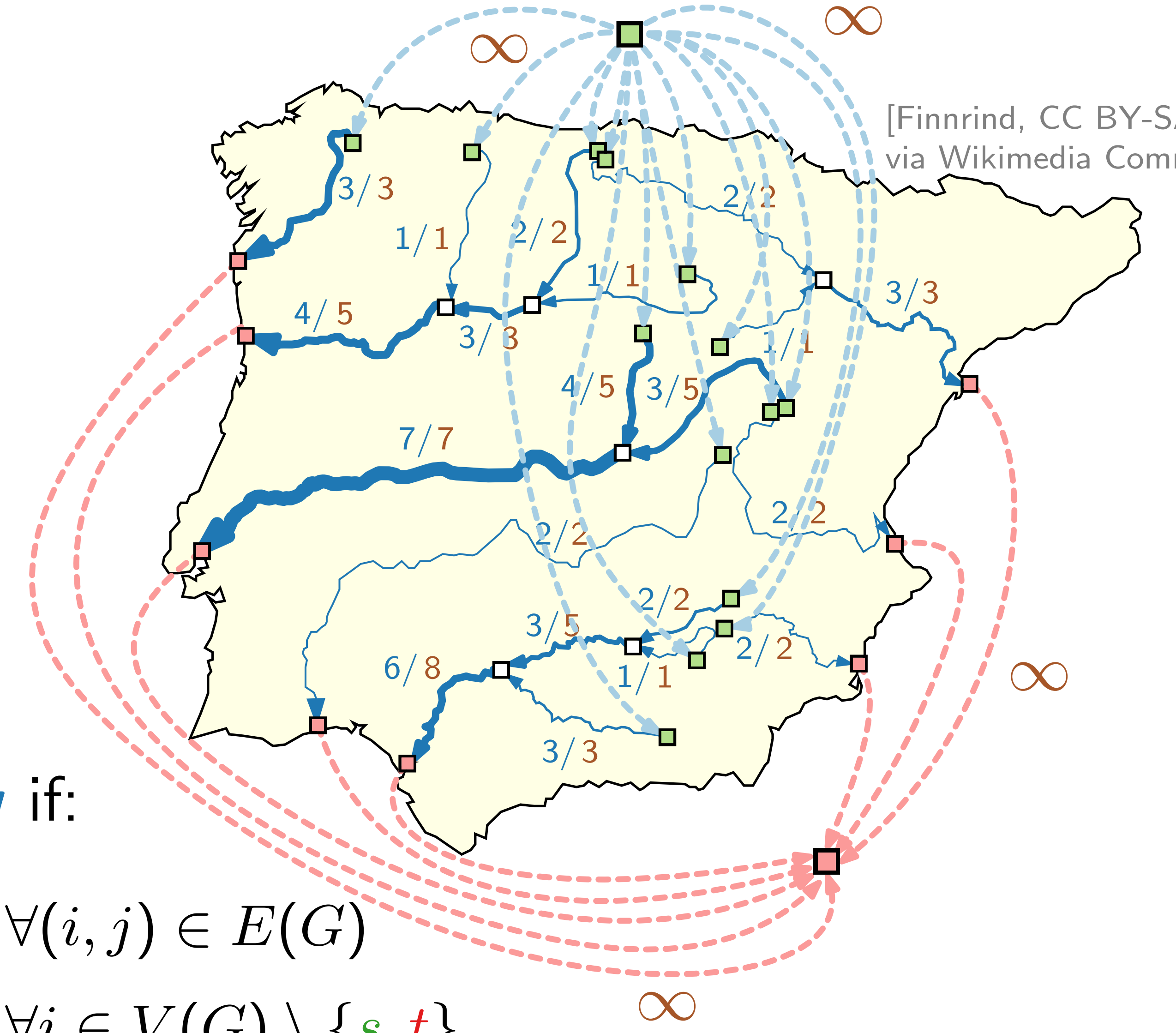
A function $X \colon E(G) \to \mathbb{R}_0^+$ is called s–t **flow** if:

$$0 \leq X(i,j) \leq u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus \{s,t\}$$

A **maximum** s–t flow is an s–t flow where $\sum_{(s,j)\in E(G)} X(s,j) - \sum_{(j,s)\in E(G)} X(j,s)$ is maximized.

Reminder: s–t Flow Networks



Flow network $(G; s, t; u)$ with

- directed graph G
- *source* $s \in V(G)$, *sink* $t \in V(G)$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

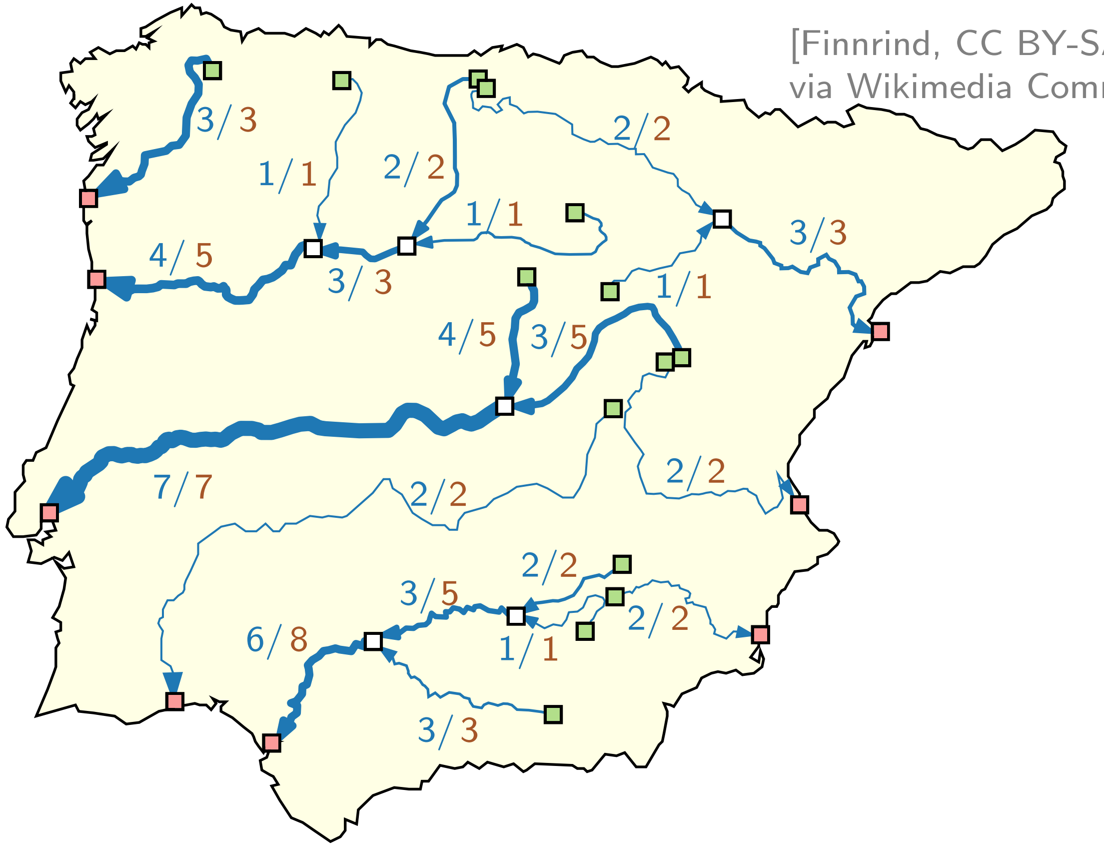
A function $X\colon E(G) \to \mathbb{R}_0^+$ is called s–t **flow** if:

$$0 \leq X(i,j) \leq u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus \{s,t\}$$

A **maximum** s–t flow is an s–t flow where $\sum_{(s,j)\in E(G)} X(s,j) - \sum_{(j,s)\in E(G)} X(j,s)$ is maximized.

General Flow Network



Flow network $(G; S, T; u)$ with

- directed graph G
- *sources* $S \subseteq V(G)$, *sinks* $T \subseteq V(G)$
- edge *capacity* $u \colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

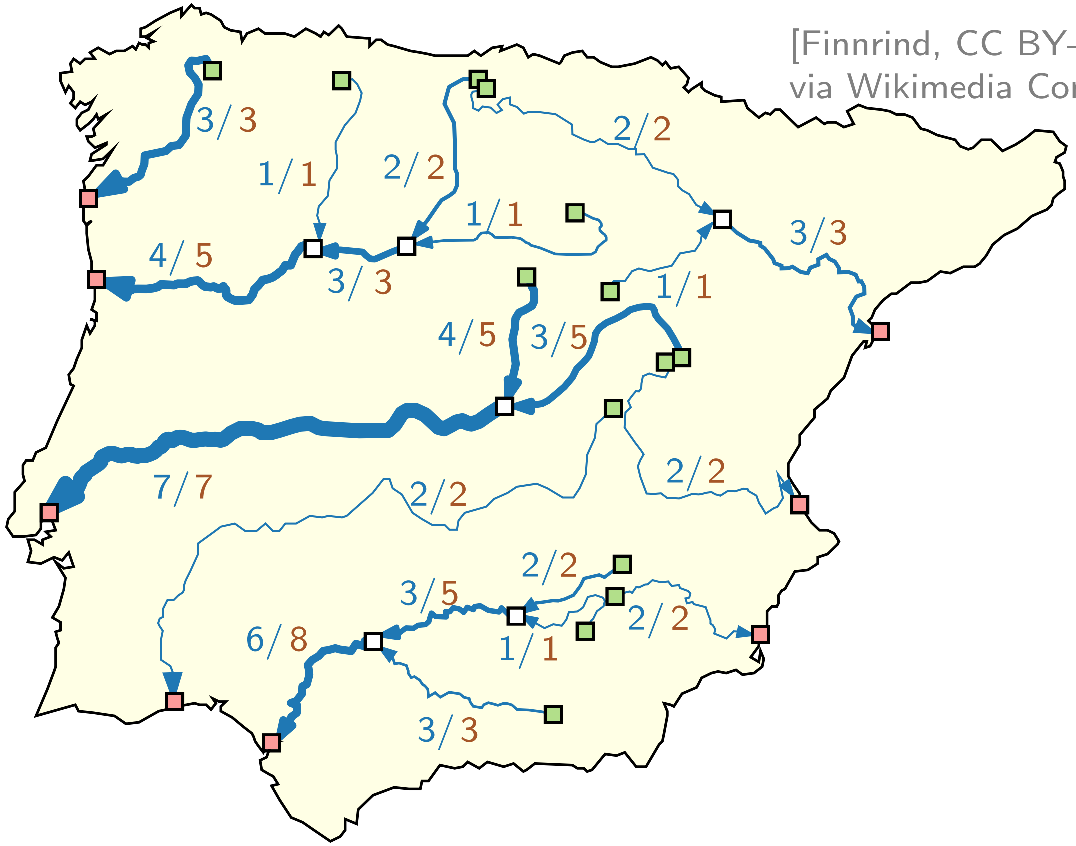
A function $X \colon E(G) \to \mathbb{R}_0^+$ is called S–T **flow** if:

$$0 \leq X(i,j) \leq u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus (S \cup T)$$

A **maximum** S–T flow is an S–T flow where $\sum_{(i,j)\in E(G), i \in S} X(i,j) - \sum_{(j,i)\in E(G), i\in S} X(j,i)$ is maximized.

General Flow Network



[Finnrind, CC BY-SA 3.0, via Wikimedia Commons]

Flow network $(G; S, T; \ell; u)$ with

- directed graph G
- *sources* $S \subseteq V(G)$, *sinks* $T \subseteq V(G)$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

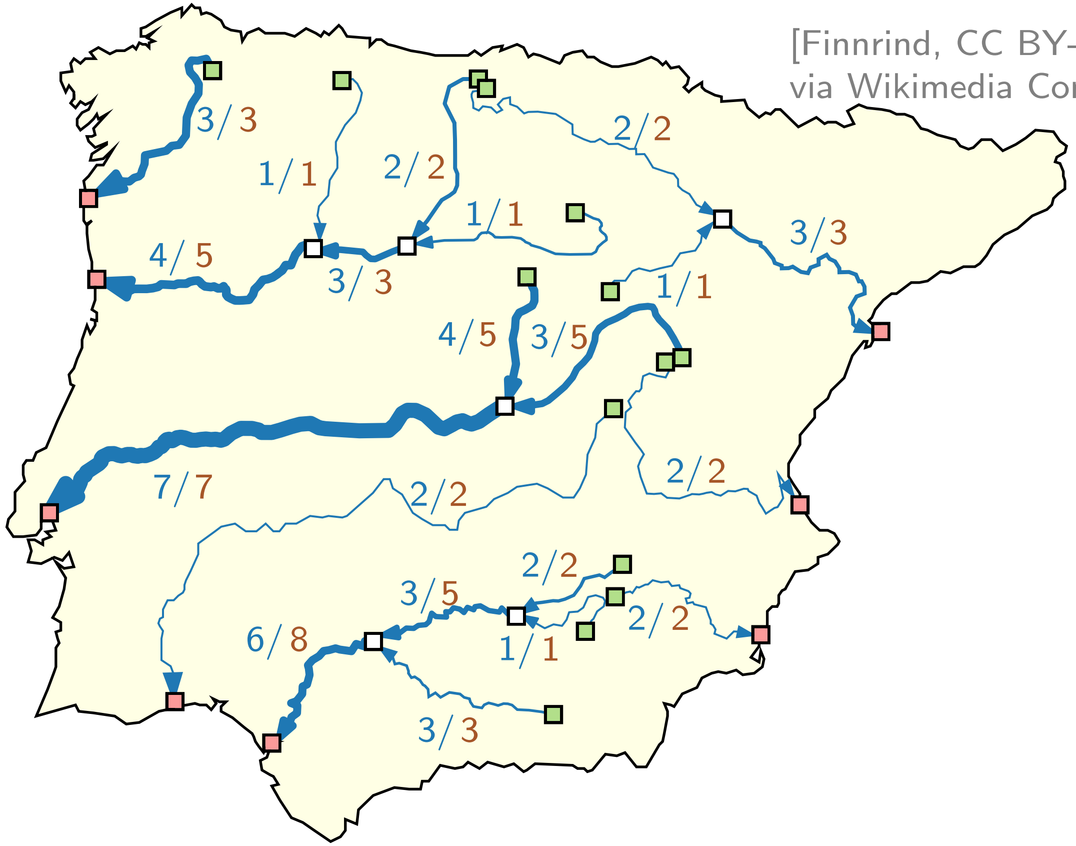
A function $X\colon E(G) \to \mathbb{R}_0^+$ is called S–T **flow** if:

$$0 \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus (S \cup T)$$

A **maximum** S–T flow is an S–T flow where $\sum_{(i,j)\in E(G), i\in S} X(i,j) - \sum_{(j,i)\in E(G), i \in S} X(j,i)$ is maximized.

General Flow Network



[Finnrind, CC BY-SA 3.0, via Wikimedia Commons]

Flow network $(G; S, T; \ell; u)$ with

- directed graph G
- *sources* $S \subseteq V(G)$, *sinks* $T \subseteq V(G)$
- edge *lower bound* $\ell\colon E(G) \to \mathbb{R}_0^+$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

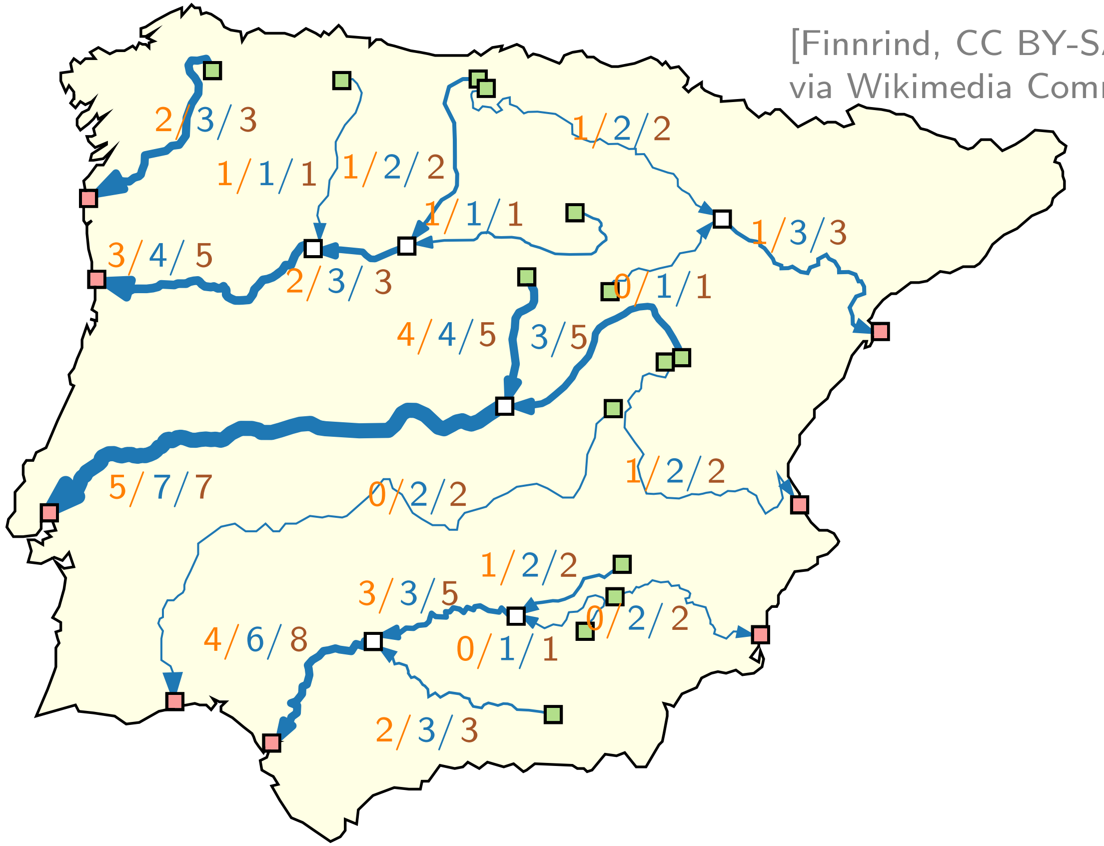
A function $X\colon E(G) \to \mathbb{R}_0^+$ is called S–T **flow** if:

$$0 \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus (S \cup T)$$

A **maximum** S–T flow is an S–T flow where $\sum_{(i,j)\in E(G), i\in S} X(i,j) - \sum_{(j,i)\in E(G), i \in S} X(j,i)$ is maximized.

General Flow Network



Flow network $(G; S, T; \ell; u)$ with

- directed graph G
- *sources* $S \subseteq V(G)$, *sinks* $T \subseteq V(G)$
- edge *lower bound* $\ell\colon E(G) \to \mathbb{R}_0^+$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

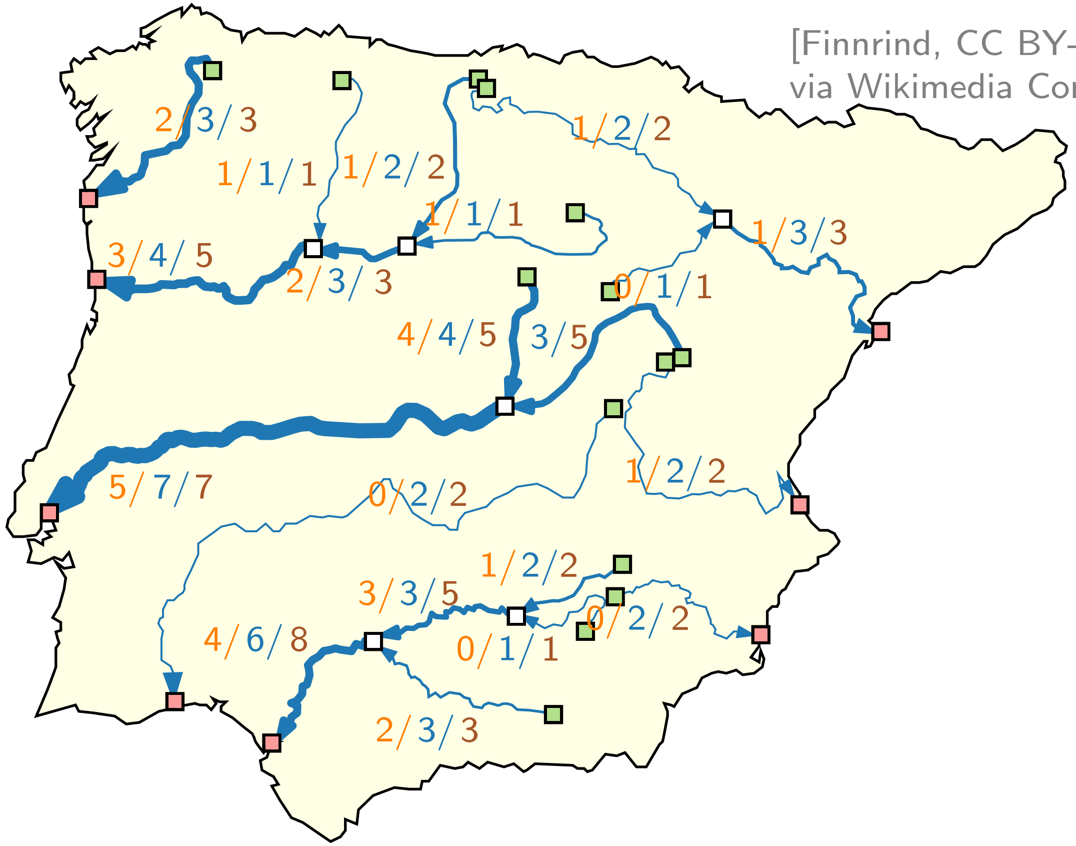
A function $X\colon E(G) \to \mathbb{R}_0^+$ is called S–T **flow** if:

$$0 \leq X(i,j) \leq u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus (S \cup T)$$

A **maximum** S–T flow is an S–T flow where $\sum_{(i,j)\in E(G), i\in S} X(i,j) - \sum_{(j,i)\in E(G), i\in S} X(j,i)$ is maximized.

General Flow Network



[Finnrind, CC BY-SA 3.0, via Wikimedia Commons]

Flow network $(G; S, T; \ell; u)$ with

- directed graph G
- *sources* $S \subseteq V(G)$, *sinks* $T \subseteq V(G)$
- edge *lower bound* $\ell \colon E(G) \to \mathbb{R}_0^+$
- edge *capacity* $u \colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

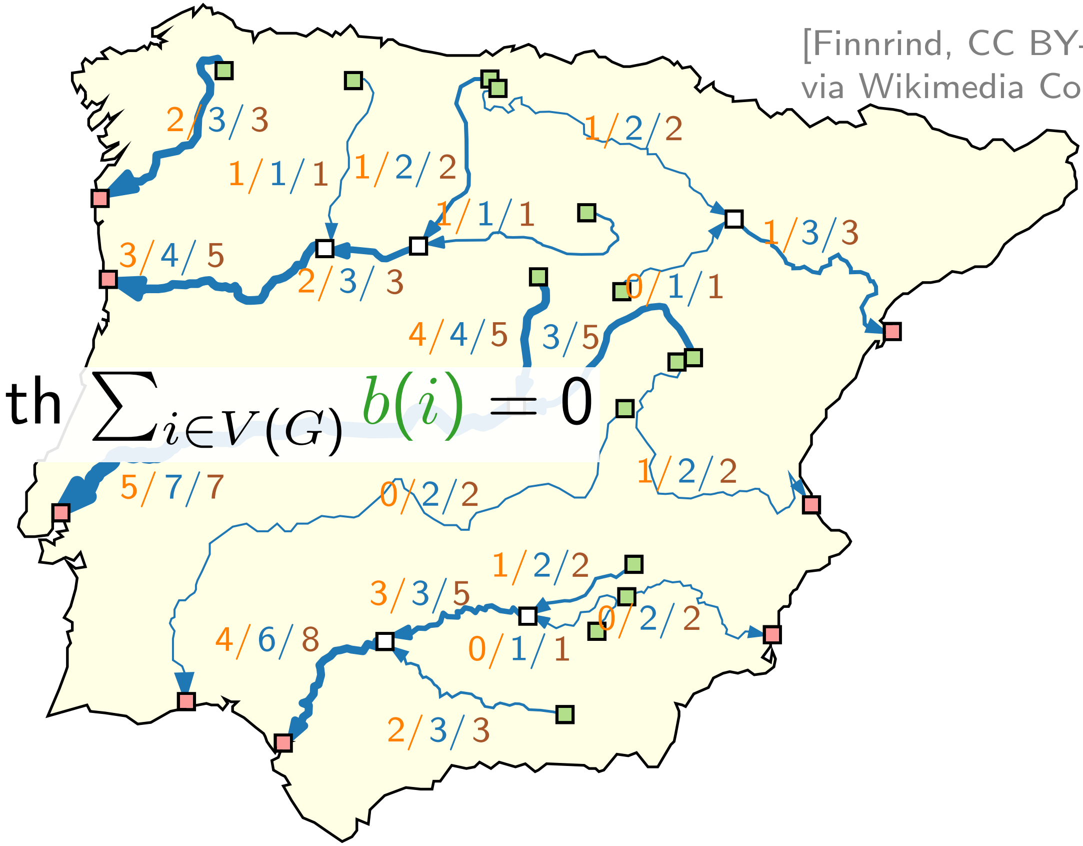
A function $X \colon E(G) \to \mathbb{R}_0^+$ is called S–T **flow** if:

$$\ell(i,j) \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus (S \cup T)$$

A **maximum** S–T flow is an S–T flow where $\sum_{(i,j)\in E(G), i \in S} X(i,j) - \sum_{(j,i)\in E(G), i\in S} X(j,i)$ is maximized.

General Flow Network



Flow network $(G; b; \ell; u)$ with

- directed graph G
- node *production/consumption* $b\colon V(G) \to \mathbb{R}$ with $\sum_{i \in V(G)} b(i) = 0$
- edge *lower bound* $\ell\colon E(G) \to \mathbb{R}_0^+$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

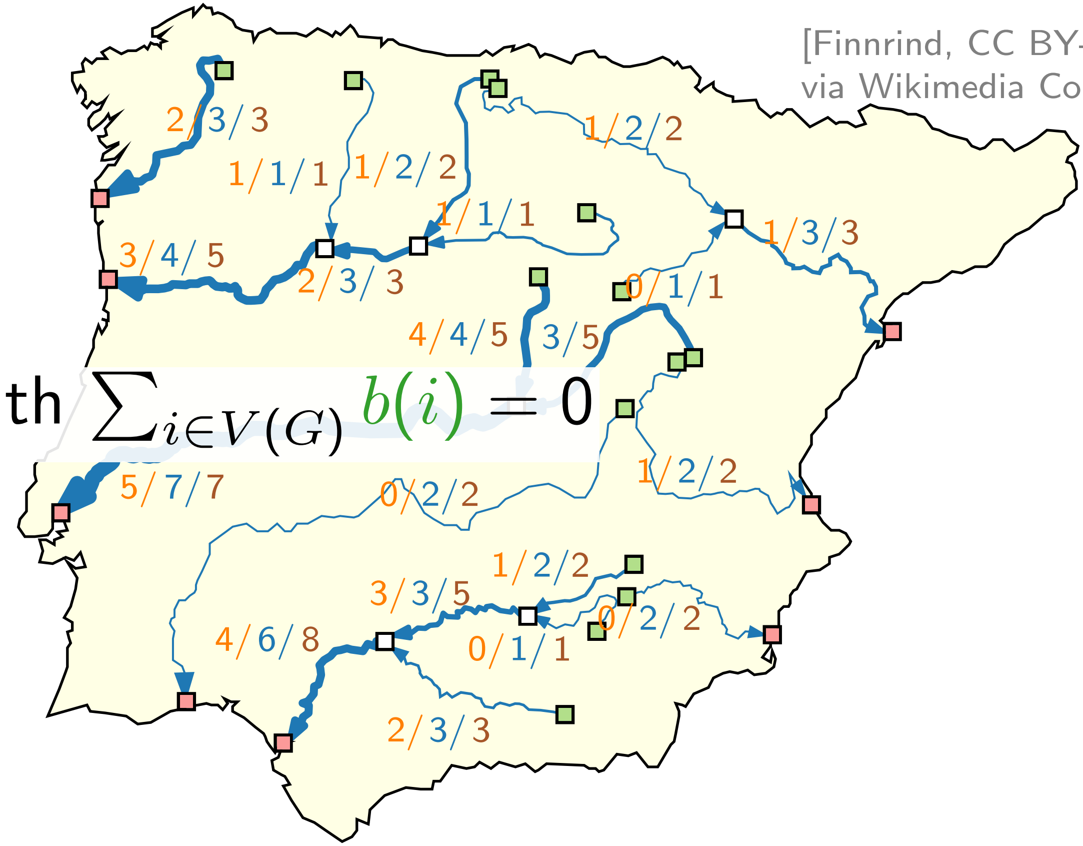
A function $X\colon E(G) \to \mathbb{R}_0^+$ is called S–T **flow** if:

$$\ell(i,j) \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = 0 \qquad \forall i \in V(G) \setminus (S \cup T)$$

A **maximum** S–T flow is an S–T flow where $\sum_{(i,j)\in E(G), i \in S} X(i,j) - \sum_{(j,i)\in E(G), i\in S} X(j,i)$ is maximized.

General Flow Network



Flow network $(G; b; \ell; u)$ with

- directed graph G
- node *production/consumption* $b\colon V(G) \to \mathbb{R}$ with $\sum_{i \in V(G)} b(i) = 0$
- edge *lower bound* $\ell\colon E(G) \to \mathbb{R}_0^+$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

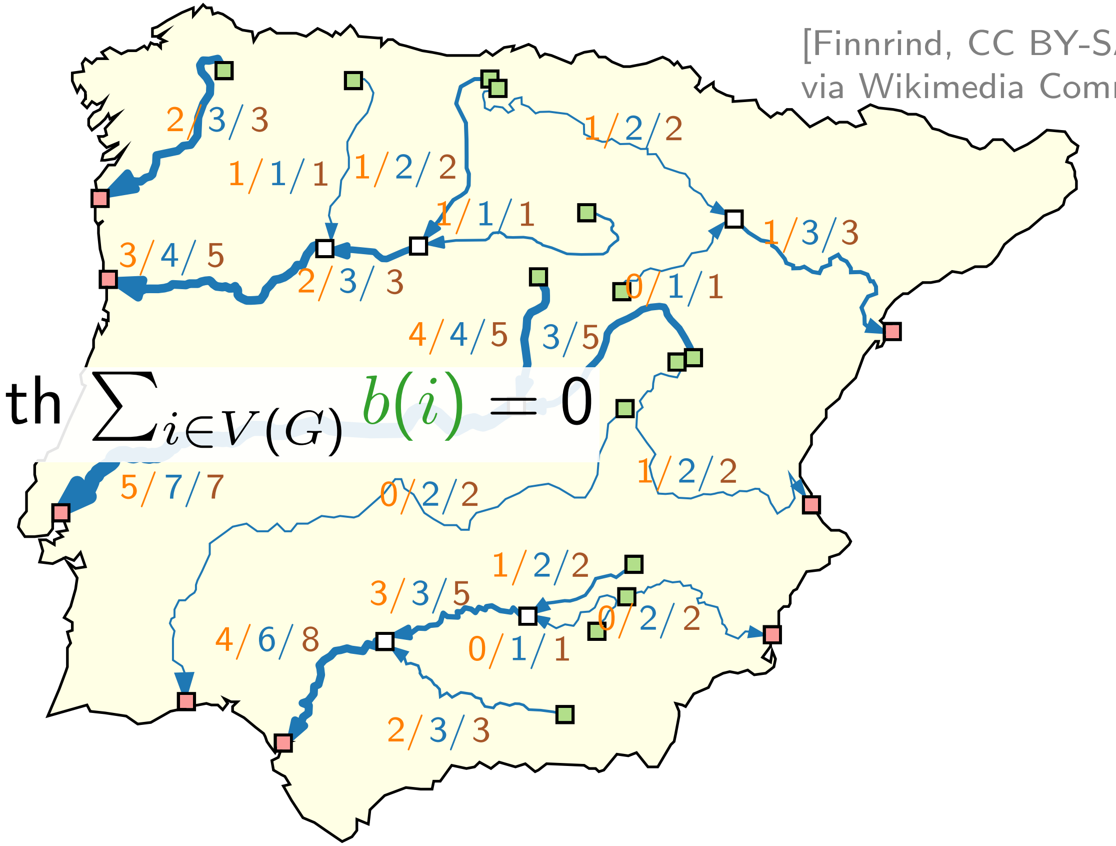
A function $X\colon E(G) \to \mathbb{R}_0^+$ is called **valid flow** if:

$$\ell(i,j) \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = b(i) \qquad \forall i \in V(G)$$

A **maximum** S–T flow is an S–T flow where $\sum_{(i,j)\in E(G), i\in S} X(i,j) - \sum_{(j,i)\in E(G), i \in S} X(j,i)$ is maximized.

General Flow Network



Flow network $(G; b; \ell; u)$ with

- directed graph G
- node *production/consumption* $b\colon V(G) \to \mathbb{R}$ with $\sum_{i \in V(G)} b(i) = 0$
- edge *lower bound* $\ell\colon E(G) \to \mathbb{R}_0^+$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

A function $X\colon E(G) \to \mathbb{R}_0^+$ is called **valid flow** if:

$$\ell(i,j) \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E(G)$$

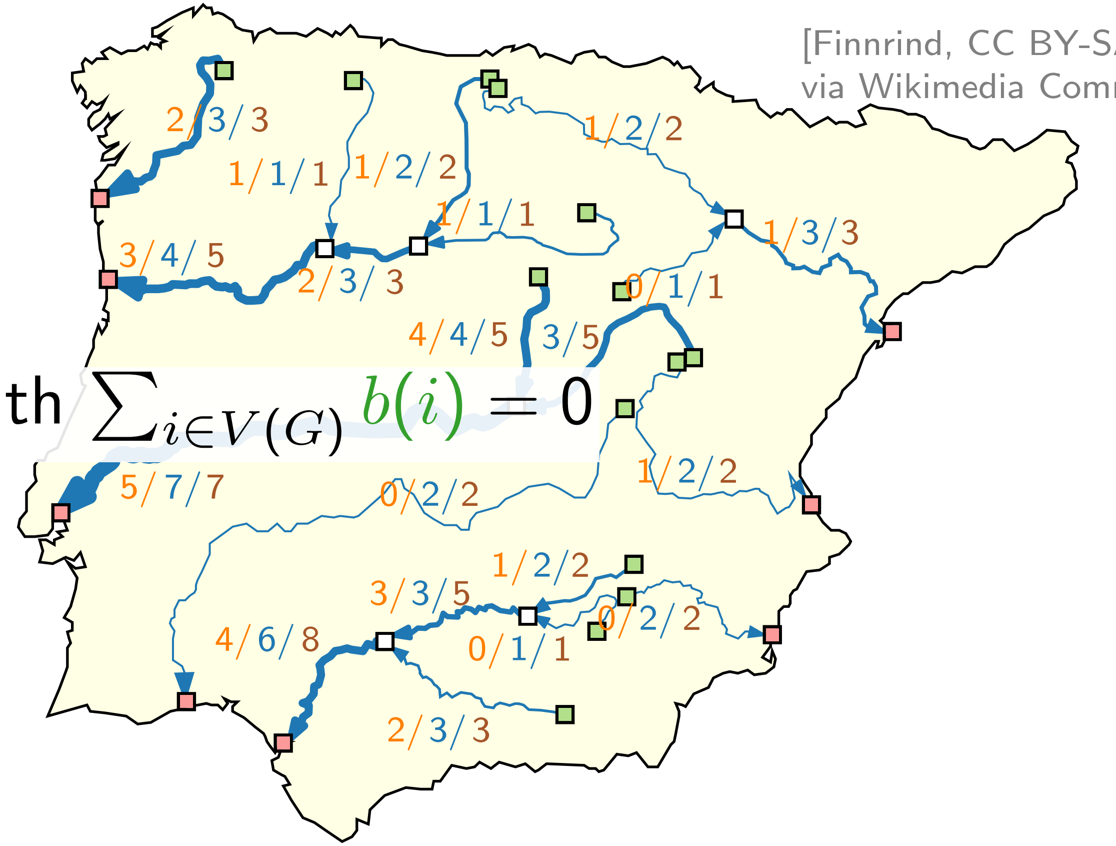
$$\sum_{(i,j)\in E(G)} X(i,j) - \sum_{(j,i)\in E(G)} X(j,i) = b(i) \qquad \forall i \in V(G)$$

~~A **maximum** S–T flow is an S–T flow where $\sum_{(i,j)\in E(G), i\in S} X(i,j) - \sum_{(j,i)\in E(G), i\in S} X(j,i)$ is maximized.~~

General Flow Network

Flow network $(G; b; \ell; u)$ with

- directed graph G
- node *production/consumption* $b\colon V(G) \to \mathbb{R}$ with $\sum_{i \in V(G)} b(i) = 0$
- edge *lower bound* $\ell\colon E(G) \to \mathbb{R}_0^+$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$



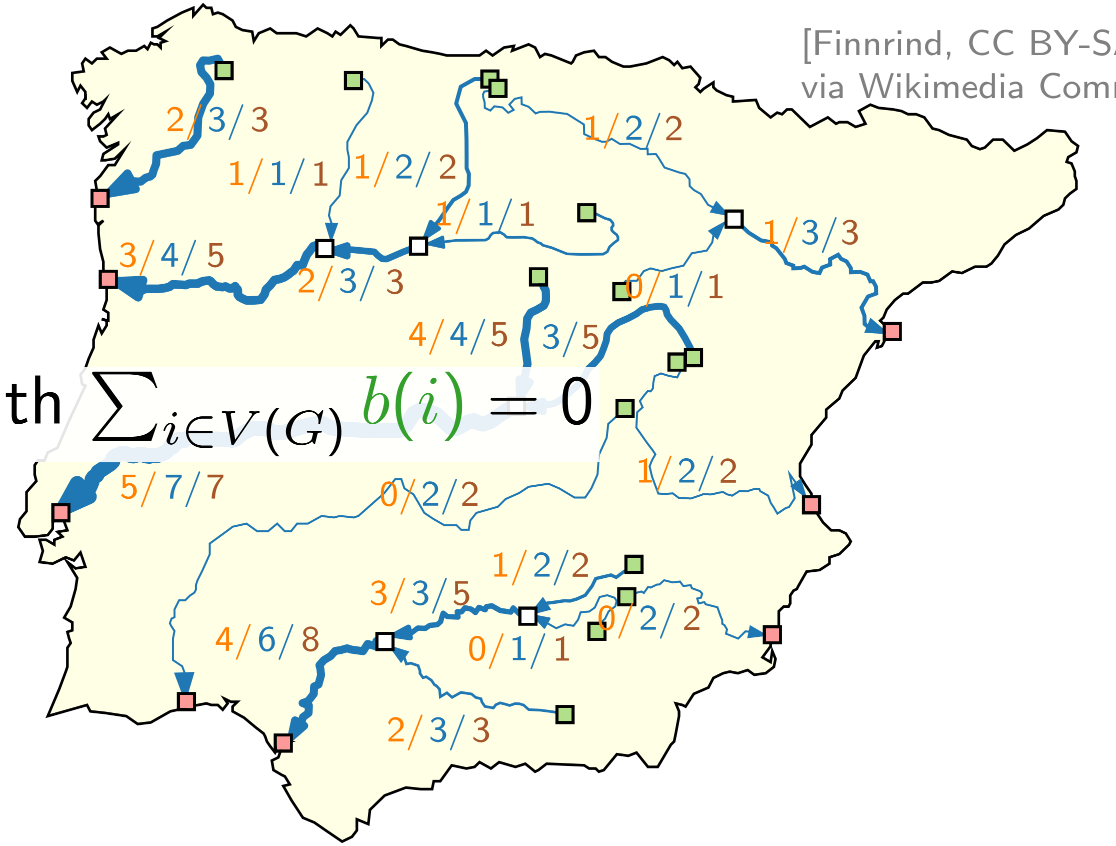
A function $X\colon E(G) \to \mathbb{R}_0^+$ is called **valid flow** if:

$$\ell(i,j) \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j) \in E(G)} X(i,j) - \sum_{(j,i) \in E(G)} X(j,i) = b(i) \qquad \forall i \in V(G)$$

- *Cost function*: cost$\colon E(G) \to \mathbb{R}_0^+$

General Flow Network



Flow network $(G; b; \ell; u)$ with

- directed graph G
- node *production/consumption* $b\colon V(G) \to \mathbb{R}$ with $\sum_{i \in V(G)} b(i) = 0$
- edge *lower bound* $\ell\colon E(G) \to \mathbb{R}_0^+$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

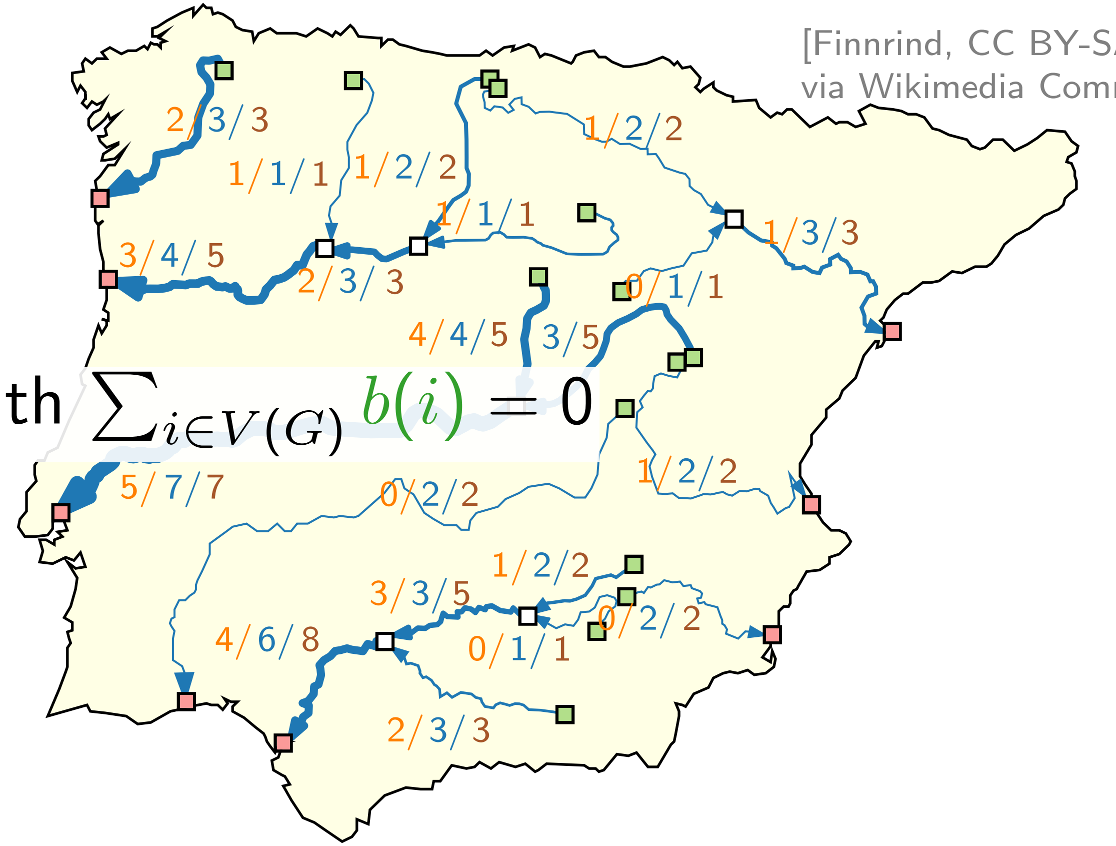
A function $X\colon E(G) \to \mathbb{R}_0^+$ is called **valid flow** if:

$$\ell(i,j) \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j) \in E(G)} X(i,j) - \sum_{(j,i) \in E(G)} X(j,i) = b(i) \quad \forall i \in V(G)$$

- *Cost function*: $\mathrm{cost}\colon E(G) \to \mathbb{R}_0^+$ and $\mathrm{cost}(X) := \sum_{(i,j) \in E(G)} \mathrm{cost}(i,j) \cdot X(i,j)$

General Flow Network



Flow network $(G; b; \ell; u)$ with

- directed graph G
- node *production/consumption* $b\colon V(G) \to \mathbb{R}$ with $\sum_{i \in V(G)} b(i) = 0$
- edge *lower bound* $\ell\colon E(G) \to \mathbb{R}_0^+$
- edge *capacity* $u\colon E(G) \to \mathbb{R}_0^+ \cup \{\infty\}$

A function $X\colon E(G) \to \mathbb{R}_0^+$ is called **valid flow** if:

$$\ell(i,j) \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E(G)$$

$$\sum_{(i,j) \in E(G)} X(i,j) - \sum_{(j,i) \in E(G)} X(j,i) = b(i) \qquad \forall i \in V(G)$$

- *Cost function*: $\text{cost}\colon E(G) \to \mathbb{R}_0^+$ and $\text{cost}(X) := \sum_{(i,j) \in E(G)} \text{cost}(i,j) \cdot X(i,j)$

X is a **minimum-cost flow** if X is a valid flow that minimizes $\text{cost}(X)$.

General Flow Network – Algorithms

n: #vertices
m: #edges

Polynomial Algorithms

#	Due to	Year	Running Time
1	Edmonds and Karp	1972	O((n + m') log U S(n, m, nC))
2	Rock	1980	O((n + m') log U S(n, m, nC))
3	Rock	1980	O(n log C M(n, m, U))
4	Bland and Jensen	1985	O(m log C M(n, m, U))
5	Goldberg and Tarjan	1987	$O(nm \log (n^2/m) \log (nC))$
6	Goldberg and Tarjan	1988	O(nm log n log (nC))
7	Ahuja, Goldberg, Orlin and Tarjan	1988	O(nm log log U log (nC))

Strongly Polynomial Algorithms

#	Due to	Year	Running Time
1	Tardos	1985	$O(m^4)$
2	Orlin	1984	$O((n + m')^2 \log n\, S(n, m))$
3	Fujishige	1986	$O((n + m')^2 \log n\, S(n, m))$
4	Galil and Tardos	1986	$O(n^2 \log n\, S(n, m))$
5	Goldberg and Tarjan	1987	$O(nm^2 \log n \log(n^2/m))$
6	Goldberg and Tarjan	1988	$O(nm^2 \log^2 n)$
7	Orlin (this paper)	1988	O((n + m') log n S(n, m))

S(n, m)	= O(m + n log n)	Fredman and Tarjan [1984]
S(n, m, C)	$= O(\text{Min}\ (m + n\sqrt{\log C}),$ $(m \log\log C))$	Ahuja, Mehlhorn, Orlin and Tarjan [1990]; Van Emde Boas, Kaas and Zijlstra[1977]
M(n, m)	$= O(\min\ (nm + n^{2+\varepsilon}, nm \log n)$ where ε is any fixed constant.	King, Rao, and Tarjan [1991]
M(n, m, U)	$= O(nm \log (\frac{n}{m}\sqrt{\log U} + 2))$	Ahuja, Orlin and Tarjan [1989]

[Orlin 1991]

General Flow Network – Algorithms

n: #vertices
m: #edges

Polynomial Algorithms

#	Due to	Year	Running Time
1	Edmonds and Karp	1972	O((n + m') log U S(n, m, nC))
2	Rock	1980	O((n + m') log U S(n, m, nC))
3	Rock	1980	O(n log C M(n, m, U))
4	Bland and Jensen	1985	O(m log C M(n, m, U))
5	Goldberg and Tarjan	1987	O(nm log (n^2/m) log (nC))
6	Goldberg and Tarjan	1988	O(nm log n log (nC))
7	Ahuja, Goldberg, Orlin and Tarjan	1988	O(nm log log U log (nC))

Strongly Polynomial Algorithms

#	Due to	Year	Running Time
1	Tardos	1985	O(m^4)
2	Orlin	1984	O($(n + m')^2$ log n S(n, m))
3	Fujishige	1986	O($(n + m')^2$ log n S(n, m))
4	Galil and Tardos	1986	O(n^2 log n S(n, m))
5	Goldberg and Tarjan	1987	O(nm^2 log n log(n^2/m))
6	Goldberg and Tarjan	1988	O(nm^2 $\log^2$ n)
7	Orlin (this paper)	1988	O((n + m') log n S(n, m))

S(n, m)	=	O(m + n log n)	Fredman and Tarjan [1984]
S(n, m, C)	=	O(Min (m + n$\sqrt{\log C}$), (m log log C))	Ahuja, Mehlhorn, Orlin and Tarjan [1990]; Van Emde Boas, Kaas and Zijlstra[1977]
M(n, m)	=	O(min (nm + $n^{2+\varepsilon}$, nm log n) where ε is any fixed constant.	King, Rao, and Tarjan [1991]
M(n, m, U)	=	O(nm log ($\frac{n}{m}\sqrt{\log U}$ + 2))	Ahuja, Orlin and Tarjan [1989]

[Orlin 1991]

Theorem. [Orlin 1991]
The minimum-cost flow problem can be solved in $O(n^2 \log^2 n + m^2 \log n)$ time.

General Flow Network – Algorithms

n: #vertices
m: #edges

Polynomial Algorithms

#	Due to	Year	Running Time
1	Edmonds and Karp	1972	O((n + m') log U S(n, m, nC))
2	Rock	1980	O((n + m') log U S(n, m, nC))
3	Rock	1980	O(n log C M(n, m, U))
4	Bland and Jensen	1985	O(m log C M(n, m, U))
5	Goldberg and Tarjan	1987	O(nm log (n^2/m) log (nC))
6	Goldberg and Tarjan	1988	O(nm log n log (nC))
7	Ahuja, Goldberg, Orlin and Tarjan	1988	O(nm log log U log (nC))

Strongly Polynomial Algorithms

#	Due to	Year	Running Time
1	Tardos	1985	O(m^4)
2	Orlin	1984	O($(n + m')^2$ log n S(n, m))
3	Fujishige	1986	O($(n + m')^2$ log n S(n, m))
4	Galil and Tardos	1986	O(n^2 log n S(n, m))
5	Goldberg and Tarjan	1987	O(nm^2 log n log(n^2/m))
6	Goldberg and Tarjan	1988	O(nm^2 $\log^2$ n)
7	Orlin (this paper)	1988	O((n + m') log n S(n, m))

S(n, m)	=	O(m + n log n)	Fredman and Tarjan [1984]
S(n, m, C)	=	O(Min (m + n$\sqrt{\log C}$), (m log log C))	Ahuja, Mehlhorn, Orlin and Tarjan [1990] Van Emde Boas, Kaas and Zijlstra[1977]
M(n, m)	=	O(min (nm + $n^{2+\varepsilon}$, nm log n) where ε is any fixed constant.	King, Rao, and Tarjan [1991]
M(n, m, U)	=	O(nm log ($\frac{n}{m}\sqrt{\log U}$ + 2))	Ahuja, Orlin and Tarjan [1989]

[Orlin 1991]

Theorem. [Orlin 1991]
The minimum-cost flow problem can be solved in $O(n^2 \log^2 n + m^2 \log n)$ time.

Theorem. [Cornelsen & Karrenbauer 2011]
The minimum-cost flow problem for planar graphs with bounded costs and face sizes can be solved in $O(n^{3/2})$ time.

General Flow Network – Algorithms

n: #vertices
m: #edges

Polynomial Algorithms

#	Due to	Year	Running Time
1	Edmonds and Karp	1972	O((n + m') log U S(n, m, nC))
2	Rock	1980	O((n + m') log U S(n, m, nC))
3	Rock	1980	O(n log C M(n, m, U))
4	Bland and Jensen	1985	O(m log C M(n, m, U))
5	Goldberg and Tarjan	1987	O(nm log (n^2/m) log (nC))
6	Goldberg and Tarjan	1988	O(nm log n log (nC))
7	Ahuja, Goldberg, Orlin and Tarjan	1988	O(nm log log U log (nC))

Strongly Polynomial Algorithms

#	Due to	Year	Running Time
1	Tardos	1985	O(m^4)
2	Orlin	1984	O($(n + m')^2$ log n S(n, m))
3	Fujishige	1986	O($(n + m')^2$ log n S(n, m))
4	Galil and Tardos	1986	O(n^2 log n S(n, m))
5	Goldberg and Tarjan	1987	O(nm^2 log n log(n^2/m))
6	Goldberg and Tarjan	1988	O(nm^2 $\log^2$ n)
7	Orlin (this paper)	1988	O((n + m') log n S(n, m))

S(n, m)	=	O(m + n log n)	Fredman and Tarjan [1984]
S(n, m, C)	=	O(Min (m + n$\sqrt{\log C}$), (m log log C))	Ahuja, Mehlhorn, Orlin and Tarjan [1990] Van Emde Boas, Kaas and Zijlstra[1977]
M(n, m)	=	O(min (nm + $n^{2+\varepsilon}$, nm log n) where ε is any fixed constant.	King, Rao, and Tarjan [1991]
M(n, m, U)	=	O(nm log ($\frac{n}{m}\sqrt{\log U}$ + 2))	Ahuja, Orlin and Tarjan [1989]

[Orlin 1991]

Theorem. [Orlin 1991]
The minimum-cost flow problem can be solved in $O(n^2 \log^2 n + m^2 \log n)$ time.

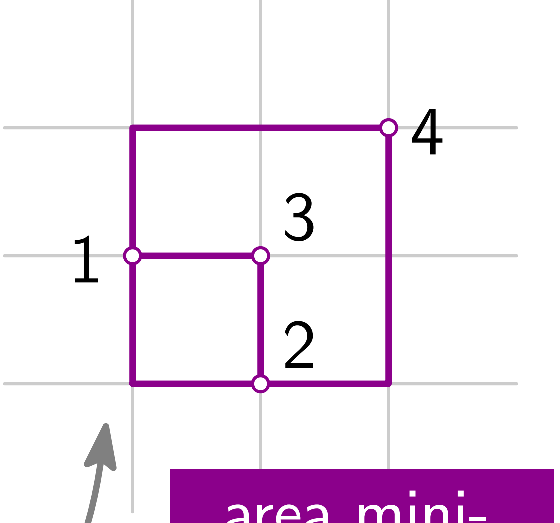
Theorem. [Cornelsen & Karrenbauer 2011]
The minimum-cost flow problem for planar graphs with bounded costs and face sizes can be solved in $O(n^{3/2})$ time.

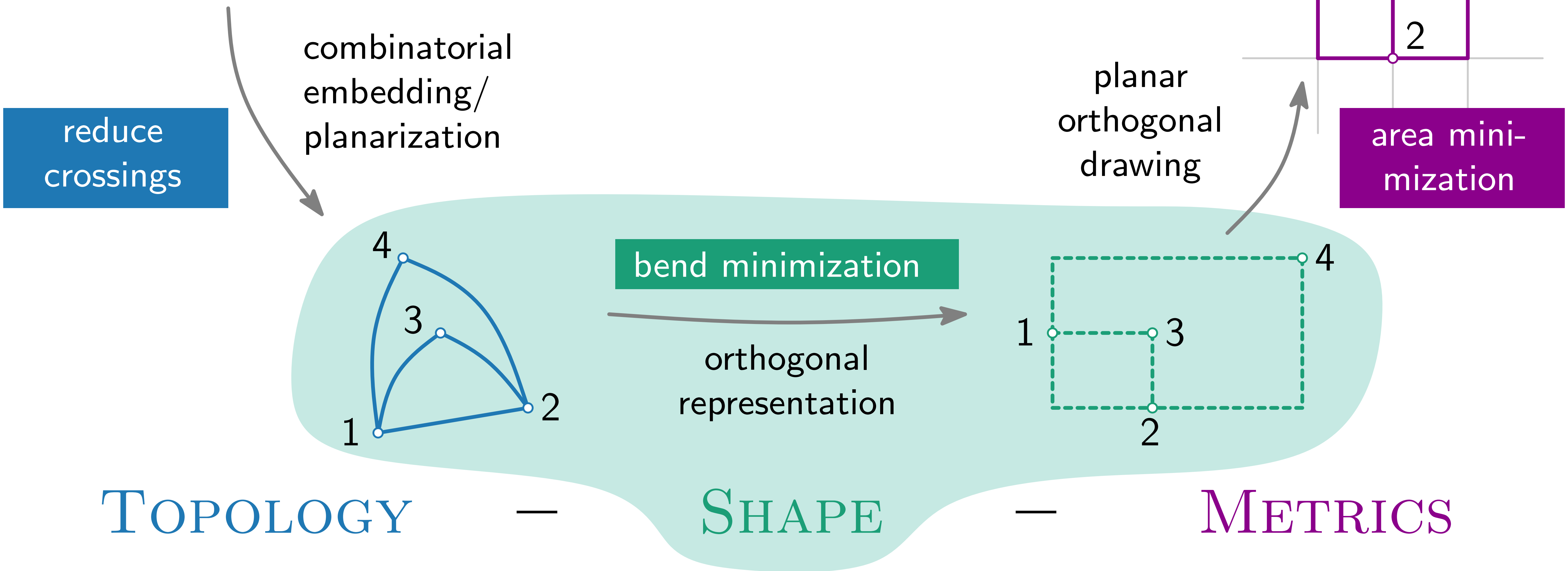
Theorem. [van den Brand, Chen, Kyng, Liu, Peng, Probst, Sachdeva, Sidford 2023]
The minimum-cost flow problem with integral vertex demands, edge capacities, and edge costs can be solved in $O(m^{1+o(1)} \log U \log C)$ time where U is the maximum capacity and C are the maximum costs.

Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

$V(G) = \{v_1, v_2, v_3, v_4\}$
$E(G) = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$





Bend Minimization with Given Embedding

Geometric orthogonal bend minimization.

Given:

Find:

Bend Minimization with Given Embedding

Geometric orthogonal bend minimization.

Given: ■ Plane graph G with maximum degree 4

Find:

Bend Minimization with Given Embedding

Geometric orthogonal bend minimization.

Given:
- Plane graph G with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find:

Bend Minimization with Given Embedding

Geometric orthogonal bend minimization.

Given:
- Plane graph G with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find: Orthogonal drawing with minimum number of bends that preserves the embedding.

Bend Minimization with Given Embedding

Geometric orthogonal bend minimization.

Given:
- Plane graph G with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find: Orthogonal drawing with minimum number of bends that preserves the embedding.

Compare with the following variant:

Combinatorial orthogonal bend minimization.

Given:

Find:

Bend Minimization with Given Embedding

Geometric orthogonal bend minimization.

Given:
- Plane graph G with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find: Orthogonal drawing with minimum number of bends that preserves the embedding.

Compare with the following variant:

Combinatorial orthogonal bend minimization.

Given:
- Plane graph G with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find:

Bend Minimization with Given Embedding

Geometric orthogonal bend minimization.

Given:
- Plane graph G with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find: Orthogonal drawing with minimum number of bends that preserves the embedding.

Compare with the following variant:

Combinatorial orthogonal bend minimization.

Given:
- Plane graph G with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find: **Orthogonal representation** $H(G)$ with minimum number of bends that preserves the embedding.

Bend Minimization with Given Embedding

How to solve the combinatorial orthogonal bend minimization problem?

Combinatorial orthogonal bend minimization.

Given:
- Plane graph G with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find: **Orthogonal representation** $H(G)$ with minimum number of bends that preserves the embedding.

Bend Minimization with Given Embedding

How to solve the combinatorial orthogonal bend minimization problem?

Idea.
Formulate as a network-flow problem:

Combinatorial orthogonal bend minimization.

Given:
- Plane graph G with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find: **Orthogonal representation** $H(G)$ with minimum number of bends that preserves the embedding.

Bend Minimization with Given Embedding

How to solve the combinatorial orthogonal bend minimization problem?

Idea.

Formulate as a network-flow problem:

- a unit of flow $= \measuredangle \frac{\pi}{2}$

Combinatorial orthogonal bend minimization.

Given:
- Plane graph G with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find: **Orthogonal representation** $H(G)$ with minimum number of bends that preserves the embedding.

Bend Minimization with Given Embedding

How to solve the combinatorial orthogonal bend minimization problem?

Idea.

Formulate as a network-flow problem:

- a unit of flow $= \measuredangle \frac{\pi}{2}$
- vertices $\xrightarrow{\measuredangle}$ faces ($\# \measuredangle \frac{\pi}{2}$ per face)

Combinatorial orthogonal bend minimization.

Given:
- Plane graph G with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find: **Orthogonal representation** $H(G)$ with minimum number of bends that preserves the embedding.

Bend Minimization with Given Embedding

How to solve the combinatorial orthogonal bend minimization problem?

Idea.
Formulate as a network-flow problem:

- a unit of flow $= \measuredangle \frac{\pi}{2}$
- vertices $\xrightarrow{\measuredangle}$ faces ($\# \measuredangle \frac{\pi}{2}$ per face)
- faces $\xrightarrow{\measuredangle}$ neighboring faces ($\#$ bends toward the neighbor)

Combinatorial orthogonal bend minimization.

Given:
- Plane graph G with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find: **Orthogonal representation** $H(G)$ with minimum number of bends that preserves the embedding.

Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Flow Network for Bend Minimization

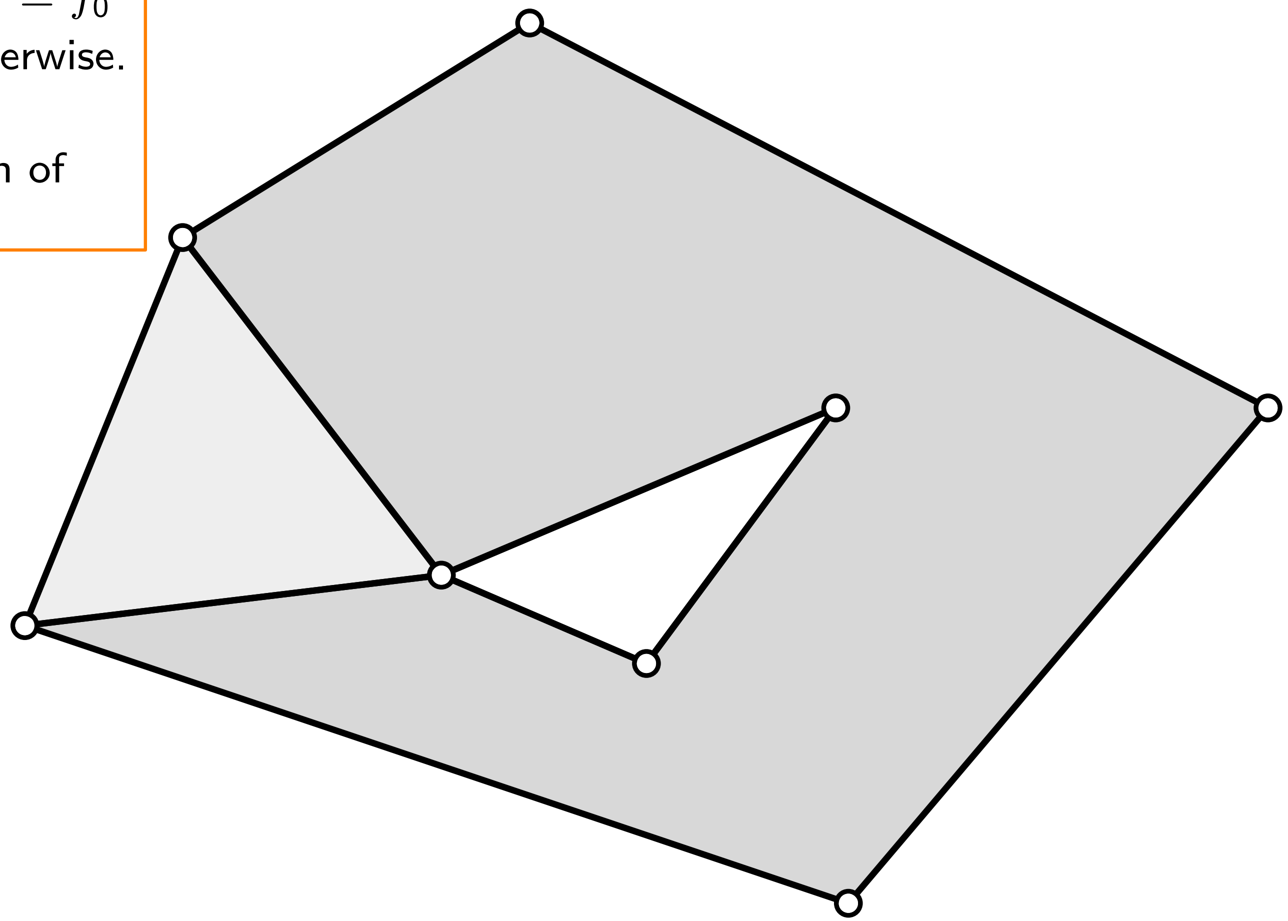
(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

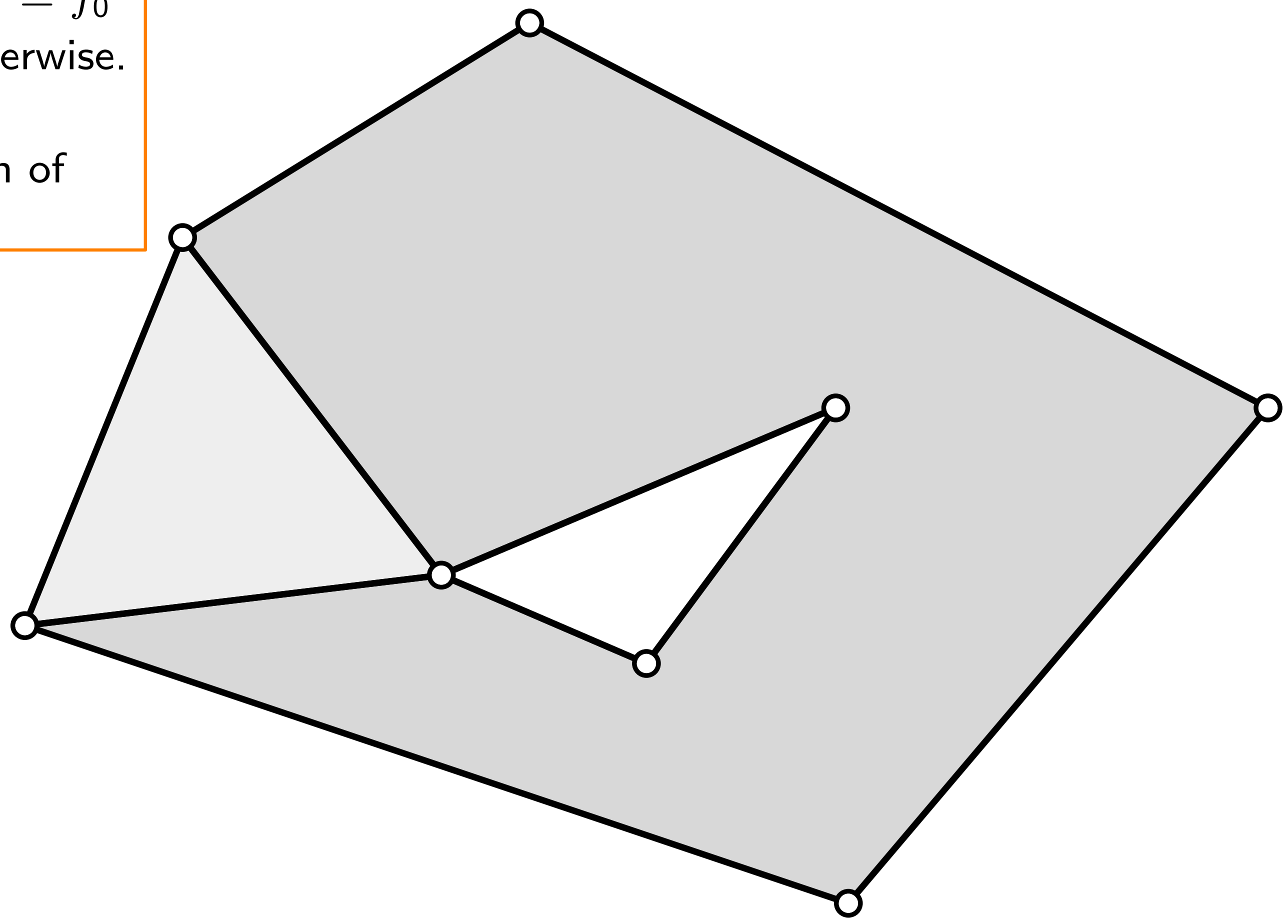
(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

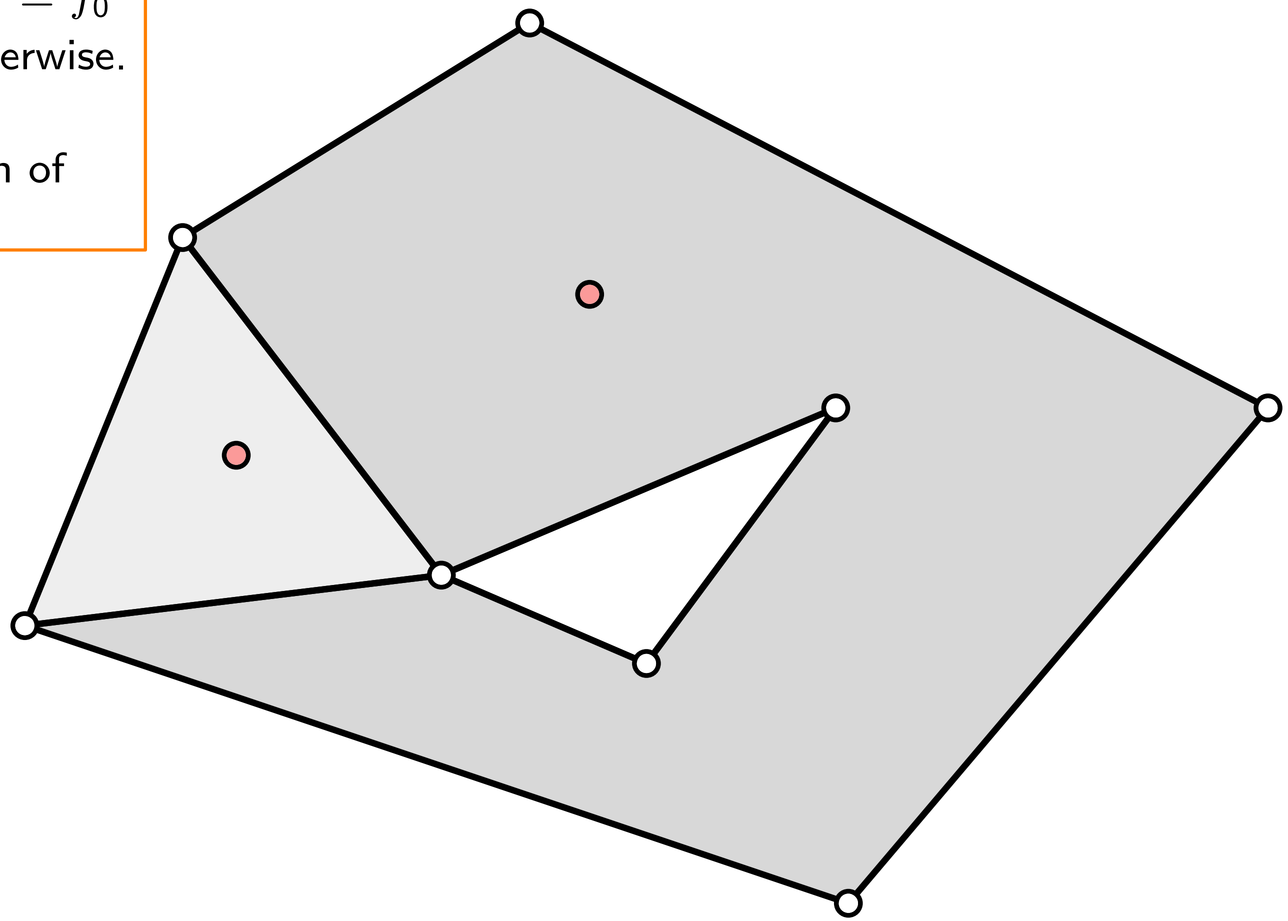
(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

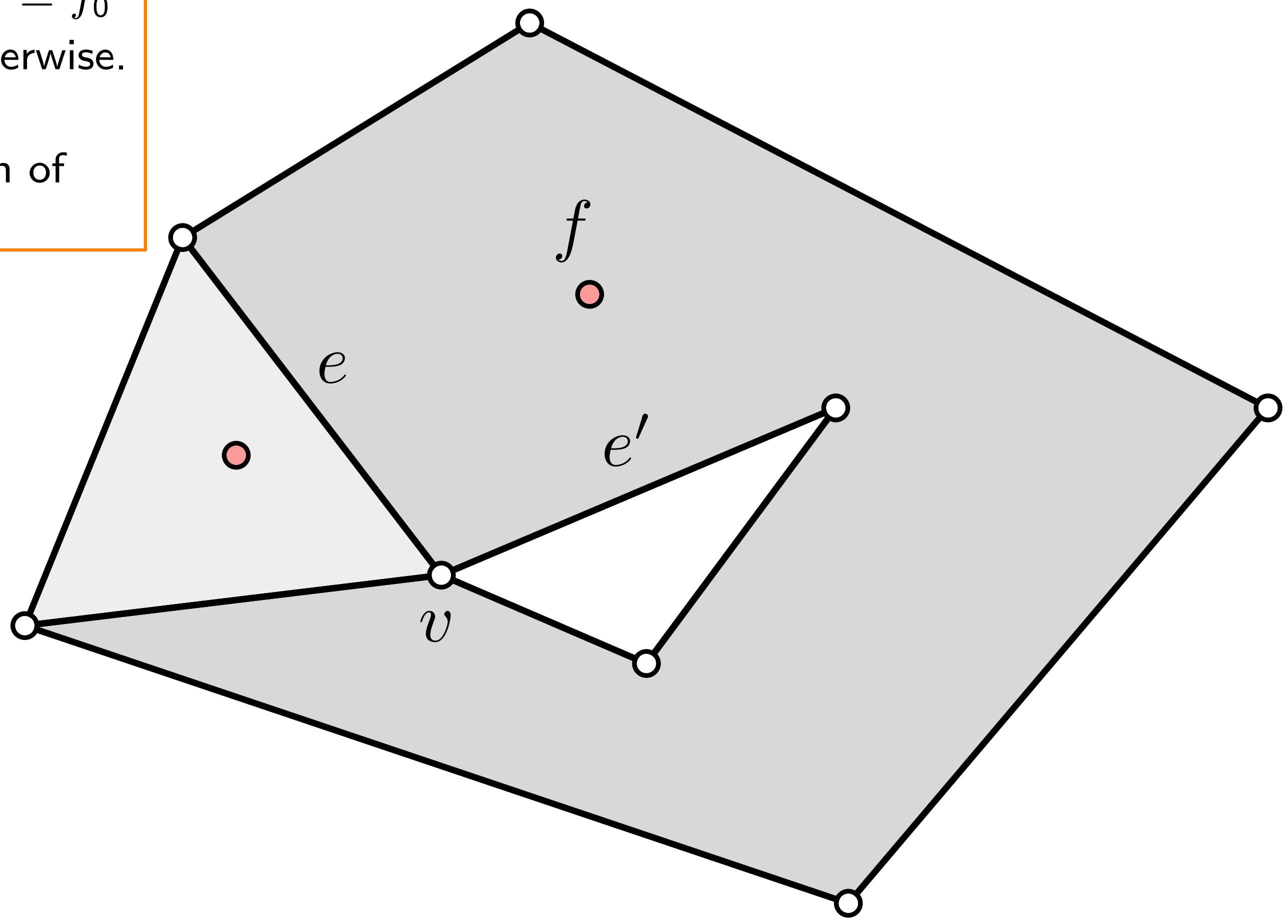
(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\}$



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

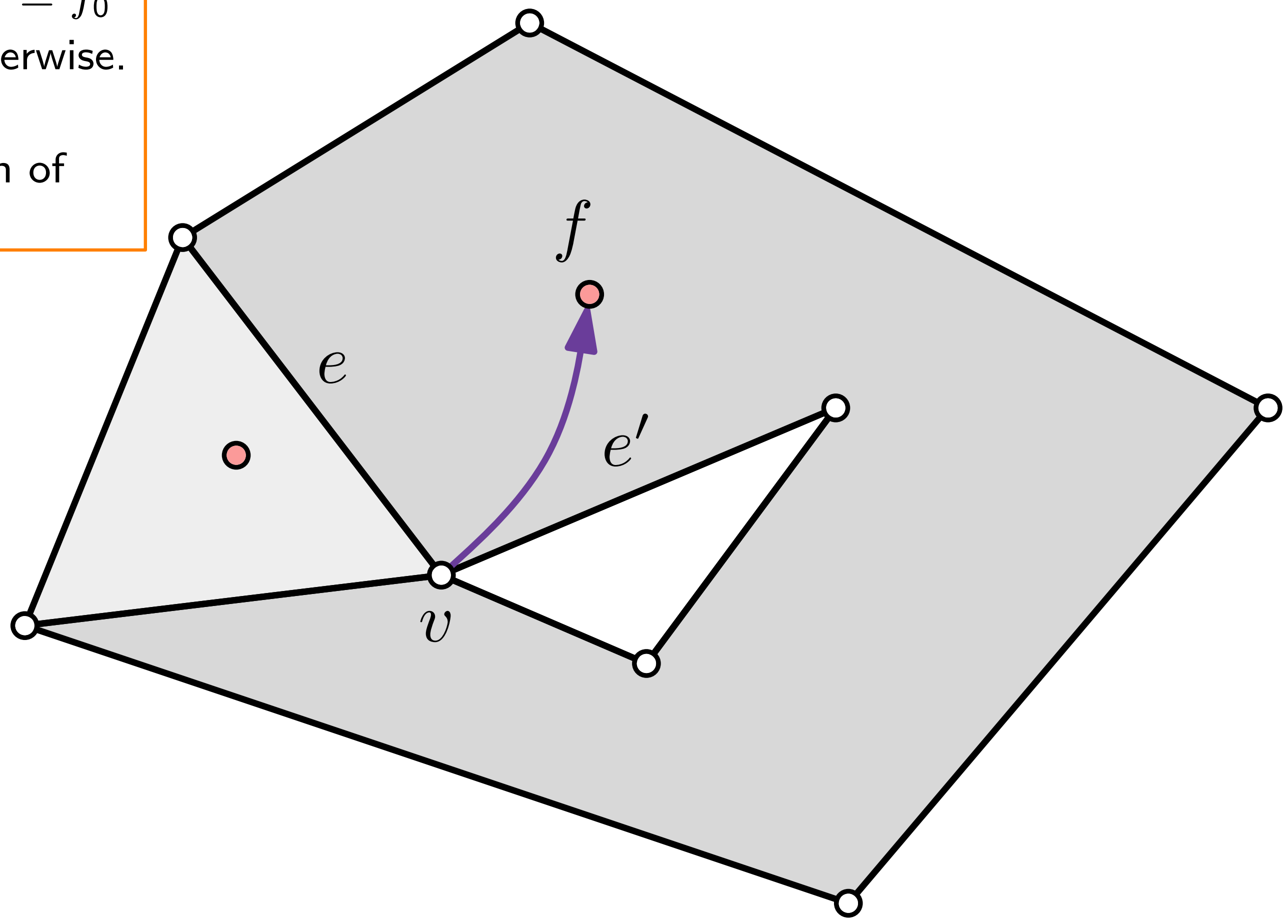
(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\}$



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

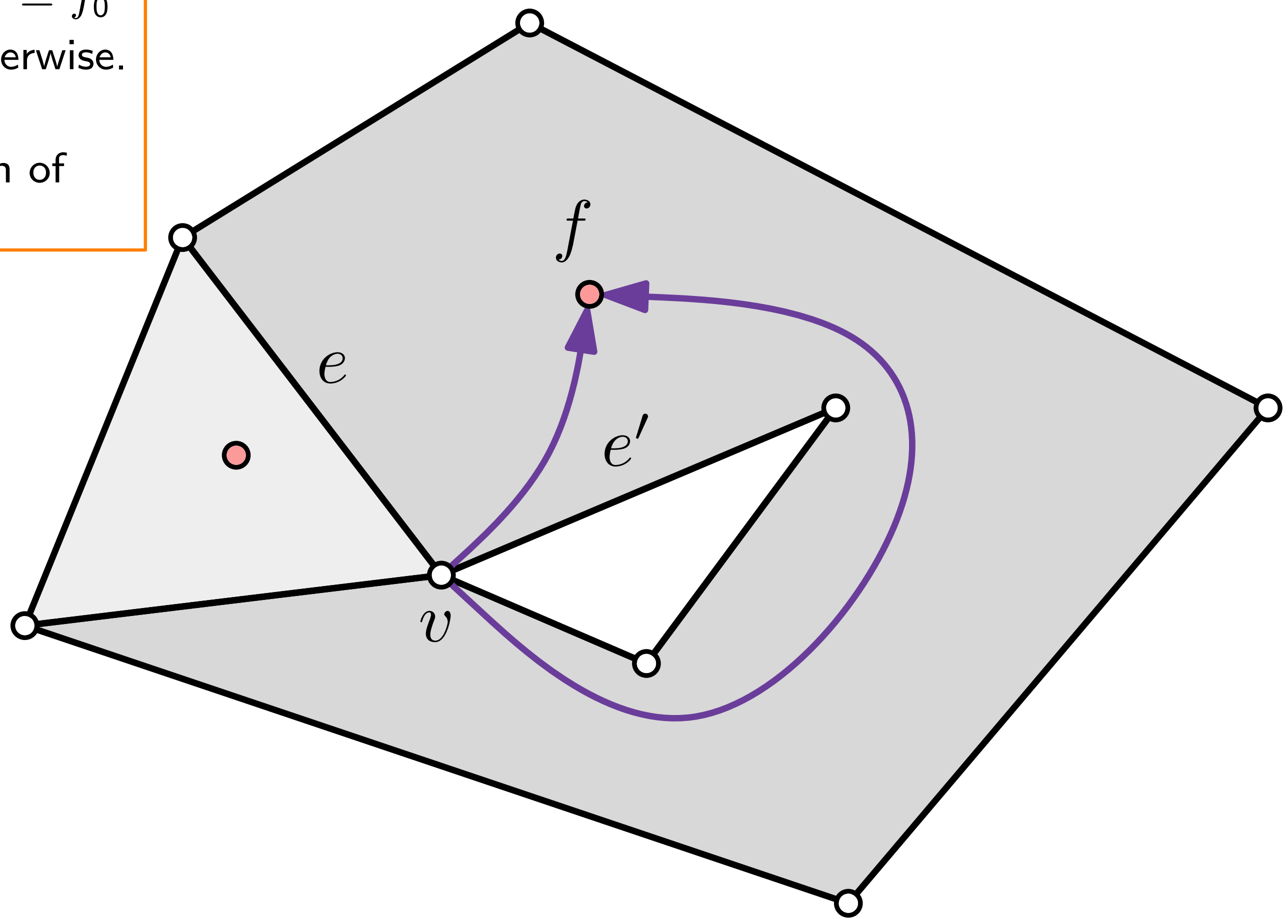
(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v$ between edges e, e' of $\partial f\}$



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

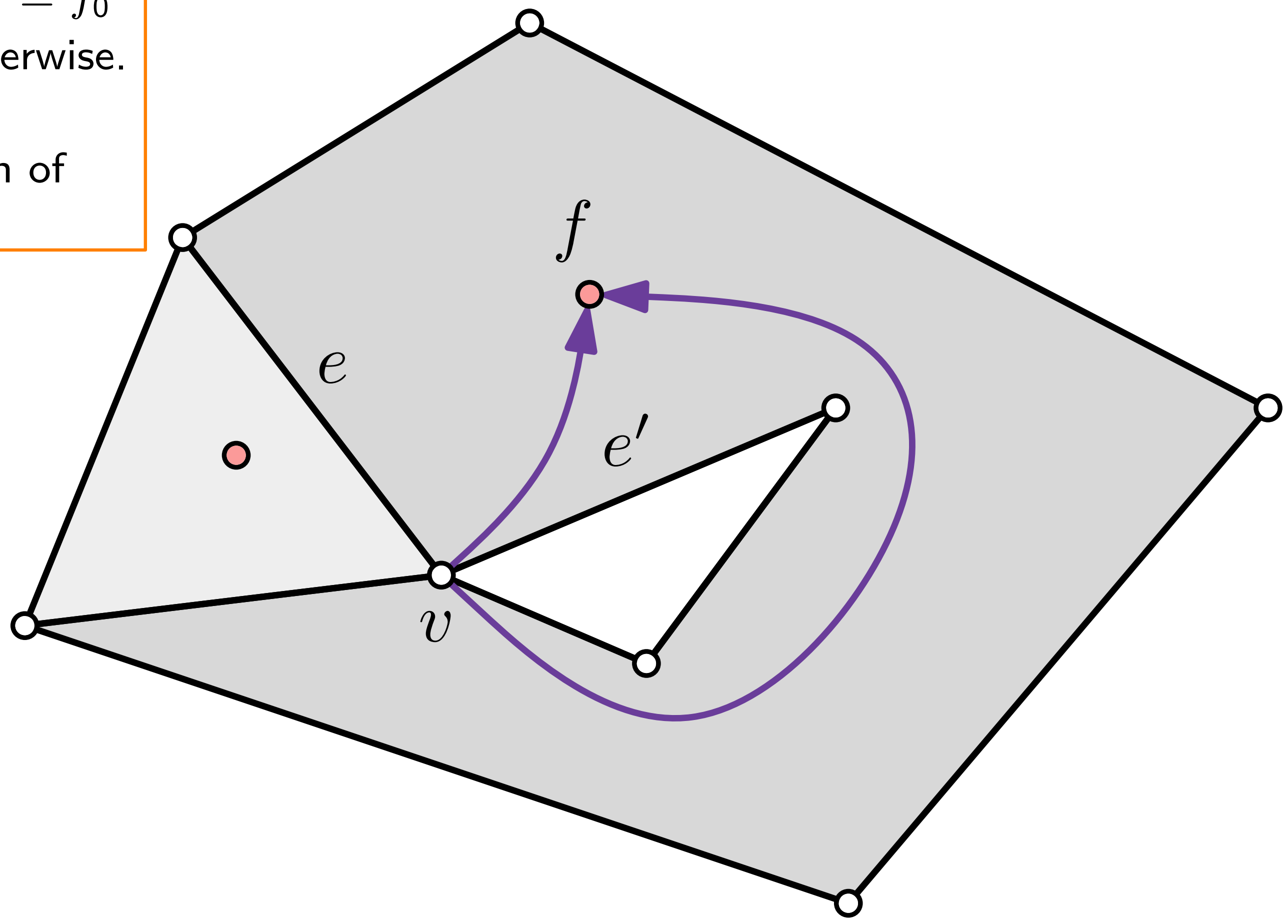
(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\}$



Directed multigraph!

Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

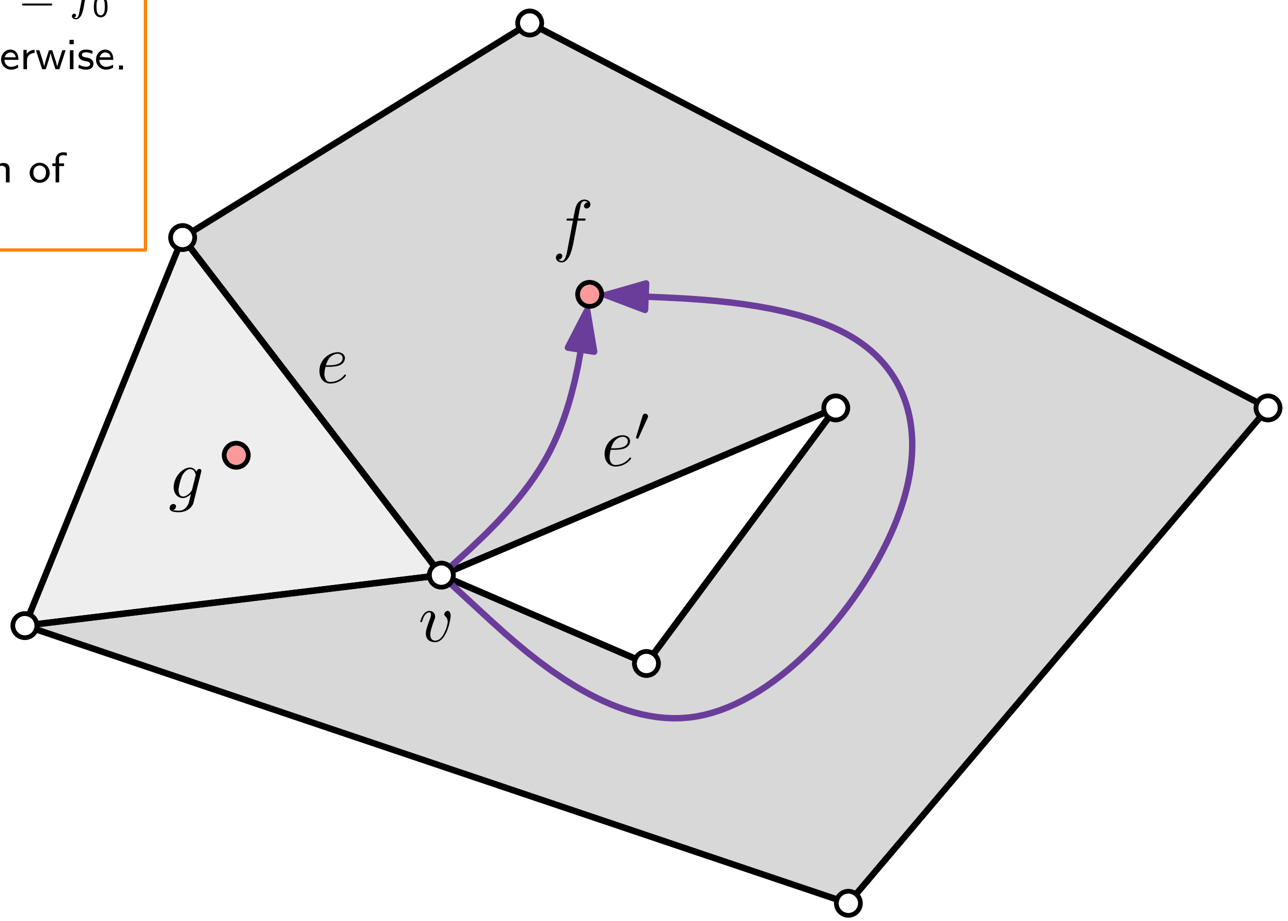
(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v$ between edges e, e' of $\partial f\} \cup \{(f, g)_e \in F \times F \mid f, g$ have common edge $e\}$



Directed multigraph!

Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

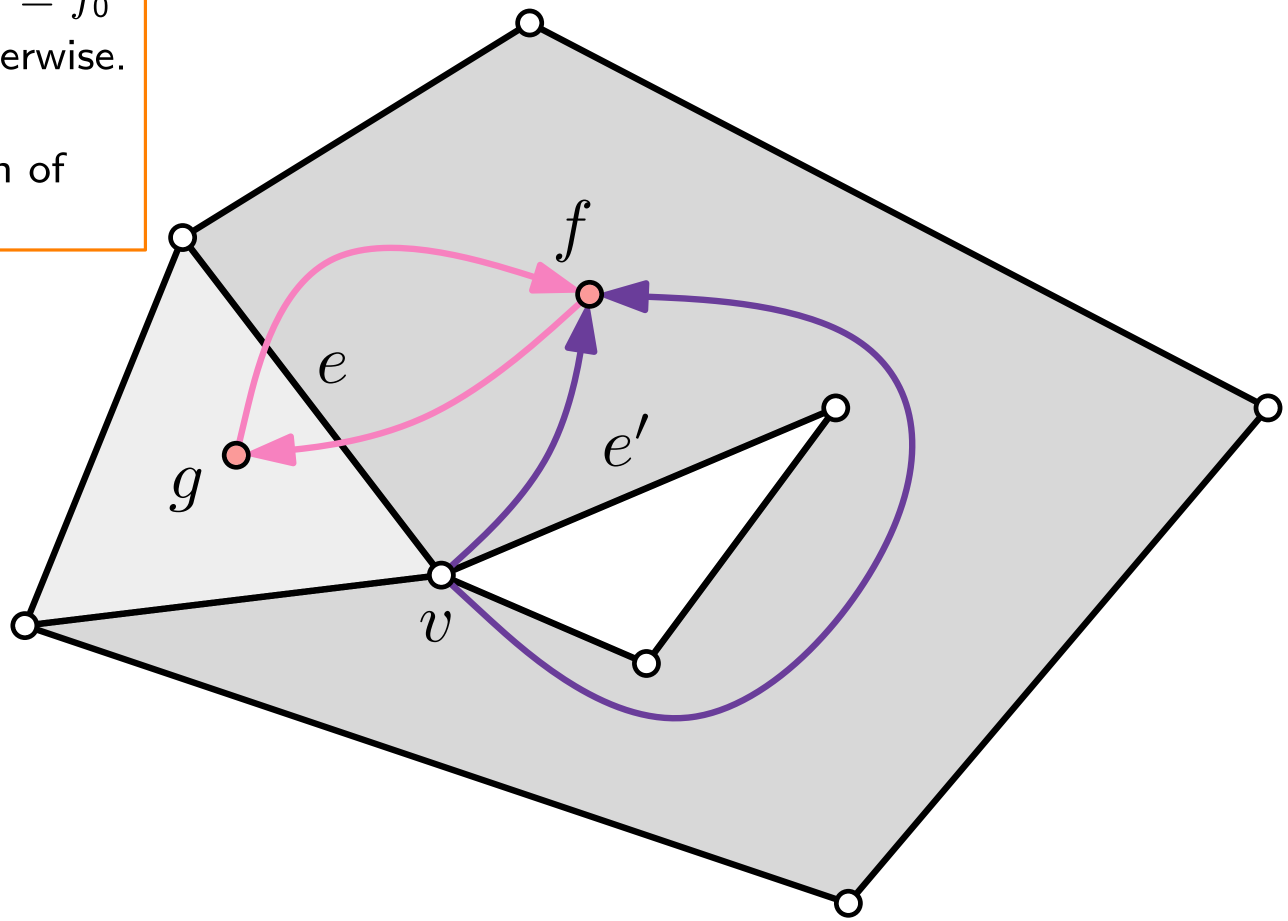
(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$



Directed multigraph!

Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

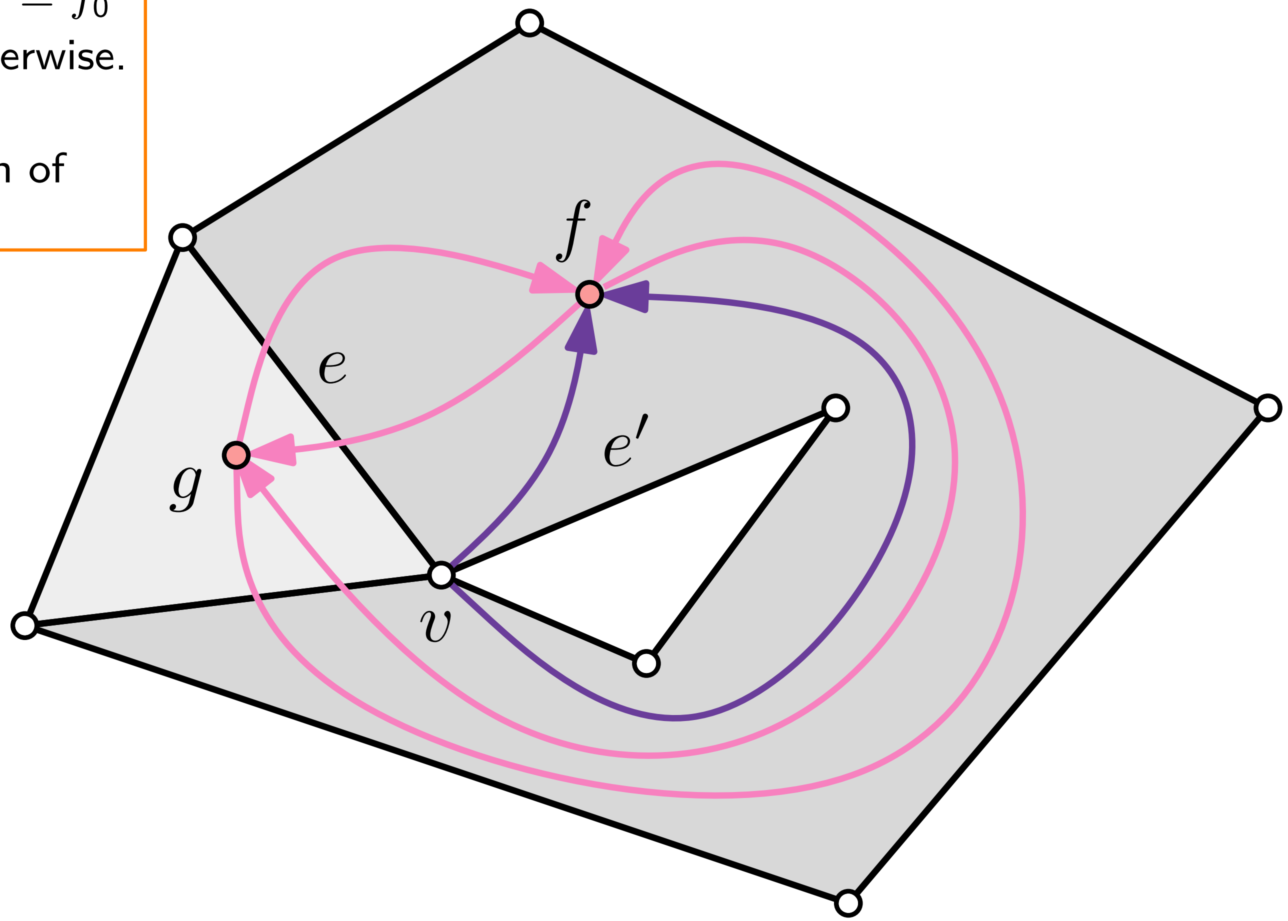
(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$



Directed multigraph!

Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$

Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

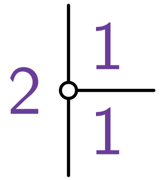
(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

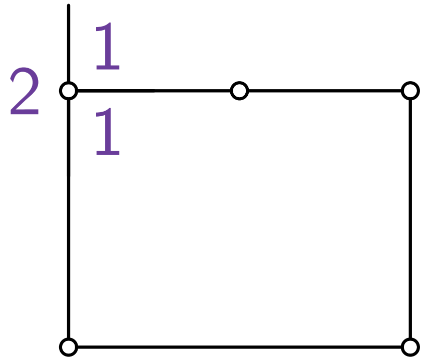
(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f)$



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

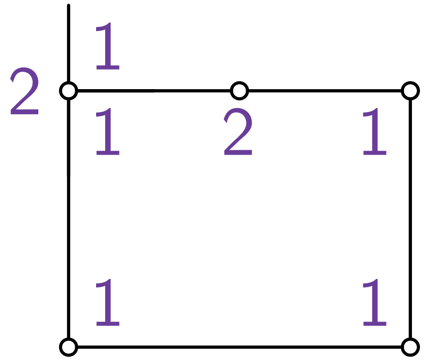
(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f)$



Flow Network for Bend Minimization

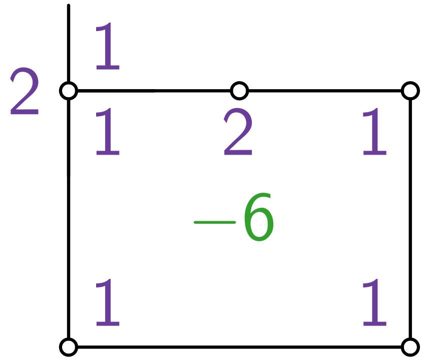
(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.



Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f)$

Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

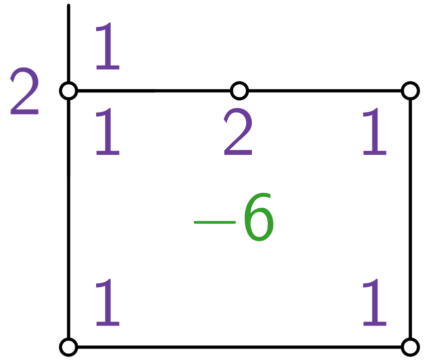
(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

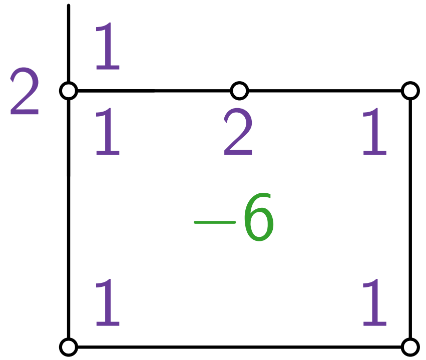
$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$

$$\Rightarrow \sum_{w \in V(G) \cup F} b(w) \stackrel{?}{=} 0$$



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

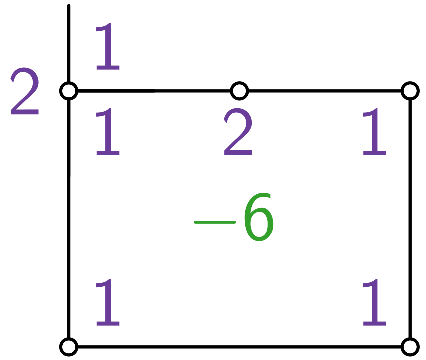
(H3) For each **face** f, it holds that:
$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$

$$\Rightarrow \sum_{w \in V(G) \cup F} b(w) = 0 \quad \text{(Euler)}$$



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

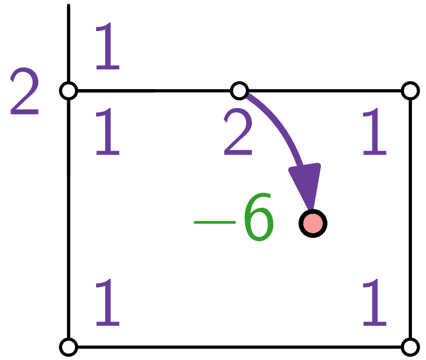
(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$

$\Rightarrow \sum_{w \in V(G) \cup F} b(w) = 0$ (Euler)

$\forall (v, f) \in E', v \in V(G), f \in F$



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

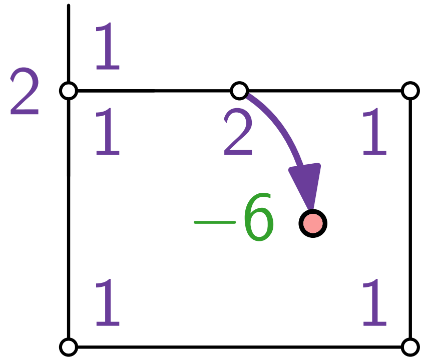
Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$

$$\Rightarrow \sum_{w \in V(G) \cup F} b(w) = 0 \quad \text{(Euler)}$$

$\forall (v, f) \in E', v \in V(G), f \in F \qquad \ell(v, f) := \quad \leq X(v, f) \leq \quad =: u(v, f)$

$\text{cost}(v, f) =$



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

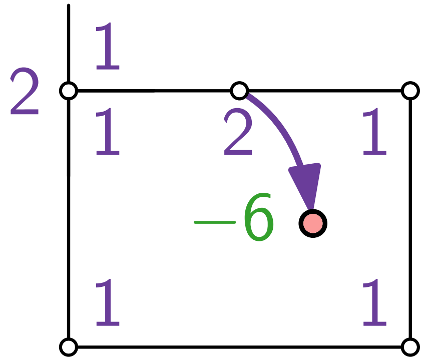
Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$

$\Rightarrow \sum_{w \in V(G) \cup F} b(w) = 0$ (Euler)

$\forall (v, f) \in E', v \in V(G), f \in F \qquad \ell(v, f) := 1 \le X(v, f) \le 4 =: u(v, f)$

$\text{cost}(v, f) =$



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

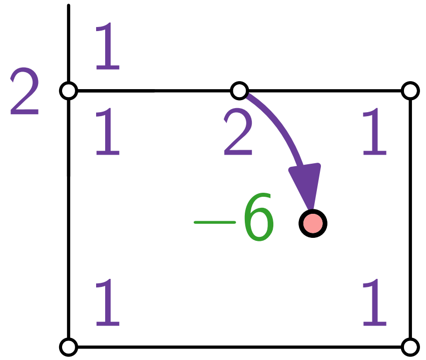
- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$

$$\Rightarrow \sum_{w \in V(G) \cup F} b(w) = 0 \quad \text{(Euler)}$$

$\forall (v, f) \in E', v \in V(G), f \in F$

$\ell(v, f) := 1 \leq X(v, f) \leq 4 =: u(v, f)$

$\text{cost}(v, f) = 0$



Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

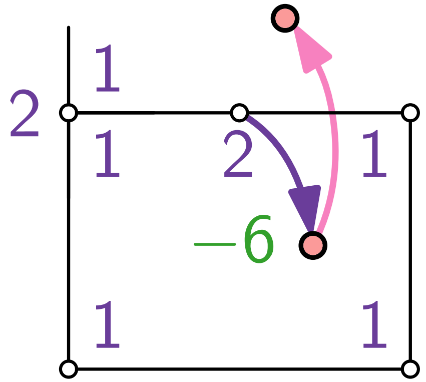
(H3) For each **face** f, it holds that:
$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$

$\Rightarrow \sum_{w \in V(G) \cup F} b(w) = 0$ (Euler)



$\forall (v, f) \in E', v \in V(G), f \in F \qquad \ell(v, f) := 1 \leq X(v, f) \leq 4 =: u(v, f)$

$\text{cost}(v, f) = 0$

$\forall (f, g) \in E', f, g \in F \qquad \ell(f, g) := \quad \leq X(f, g) \leq \quad =: u(f, g)$

$\text{cost}(f, g) =$

Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

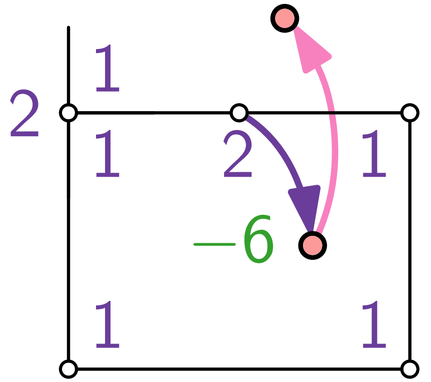
$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$

$\Rightarrow \sum_{w \in V(G) \cup F} b(w) = 0$ (Euler)



$\forall (v, f) \in E', v \in V(G), f \in F \qquad \ell(v, f) := 1 \leq X(v, f) \leq 4 =: u(v, f)$

$\text{cost}(v, f) = 0$

$\forall (f, g) \in E', f, g \in F \qquad \ell(f, g) := 0 \leq X(f, g) \leq \infty =: u(f, g)$

$\text{cost}(f, g) =$

Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

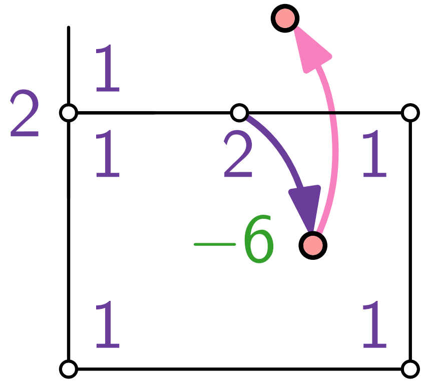
$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$

$\Rightarrow \sum_{w \in V(G) \cup F} b(w) = 0$ (Euler)



$\forall (v, f) \in E', v \in V(G), f \in F \qquad \ell(v, f) := 1 \leq X(v, f) \leq 4 =: u(v, f)$

$\text{cost}(v, f) = 0$

$\forall (f, g) \in E', f, g \in F \qquad \ell(f, g) := 0 \leq X(f, g) \leq \infty =: u(f, g)$

$\text{cost}(f, g) = 1$

Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

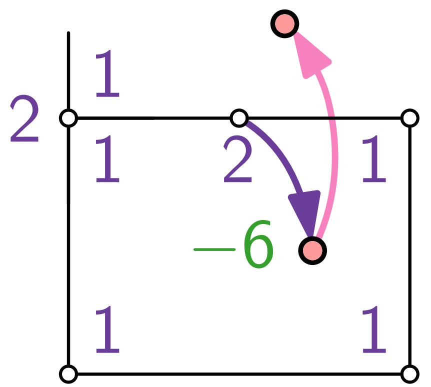
$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$

$\Rightarrow \sum_{w \in V(G) \cup F} b(w) = 0$ (Euler)

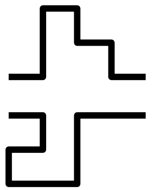


$\forall (v, f) \in E', v \in V(G), f \in F$ $\quad \ell(v, f) := 1 \leq X(v, f) \leq 4 =: u(v, f)$

$\text{cost}(v, f) = 0$

$\forall (f, g) \in E', f, g \in F$ $\quad \ell(f, g) := 0 \leq X(f, g) \leq \infty =: u(f, g)$

$\text{cost}(f, g) = 1$



We model only the *number* of bends. Why is it enough?

Flow Network for Bend Minimization

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, the sequence δ_1 is reversed and inverted copy of δ_2.

(H3) For each **face** f, it holds that:

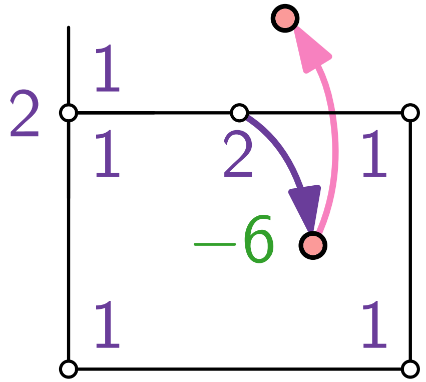
$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v, the sum of incident angles is 2π.

Define flow network $N(G) = ((V(G) \cup F, E'); b; \ell; u; \text{cost})$:

- $E' = \{(v, f)_{ee'} \in V(G) \times F \mid v \text{ between edges } e, e' \text{ of } \partial f\} \cup \{(f, g)_e \in F \times F \mid f, g \text{ have common edge } e\}$
- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$

$\Rightarrow \sum_{w \in V(G) \cup F} b(w) = 0$ (Euler)

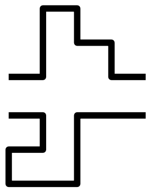


$\forall (v, f) \in E', v \in V(G), f \in F \qquad \ell(v, f) := 1 \leq X(v, f) \leq 4 =: u(v, f)$

$\text{cost}(v, f) = 0$

$\forall (f, g) \in E', f, g \in F \qquad \ell(f, g) := 0 \leq X(f, g) \leq \infty =: u(f, g)$

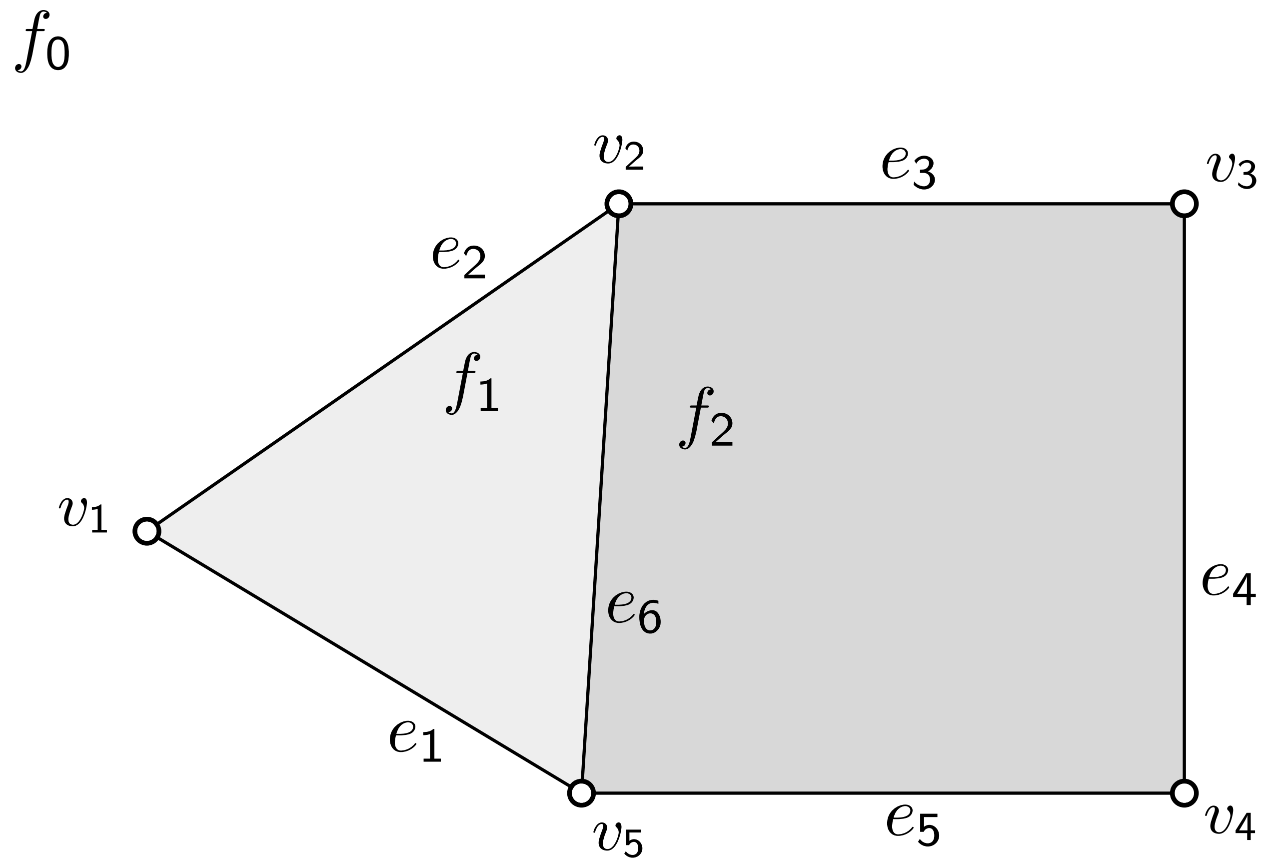
$\text{cost}(f, g) = 1$



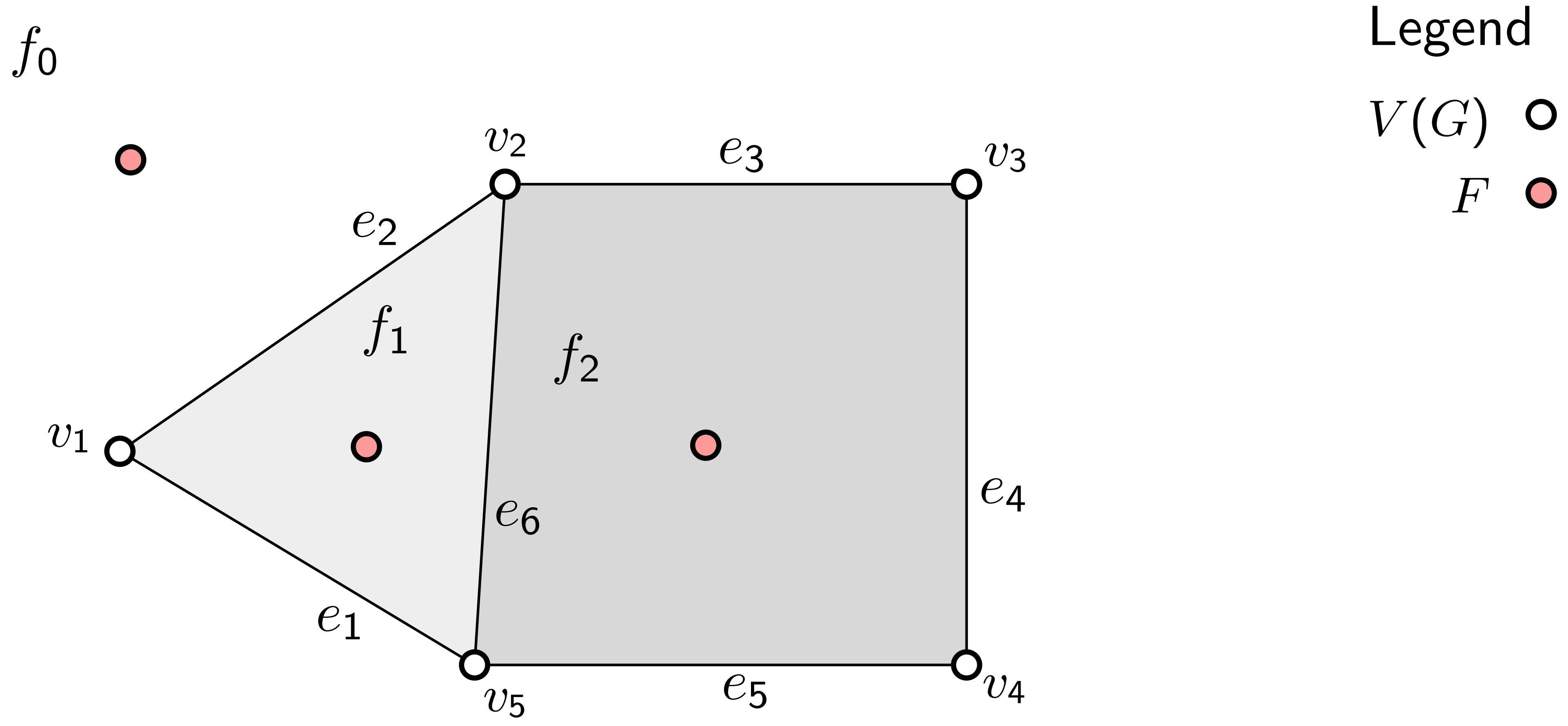
We model only the *number* of bends. Why is it enough?

→ *Exercise!*

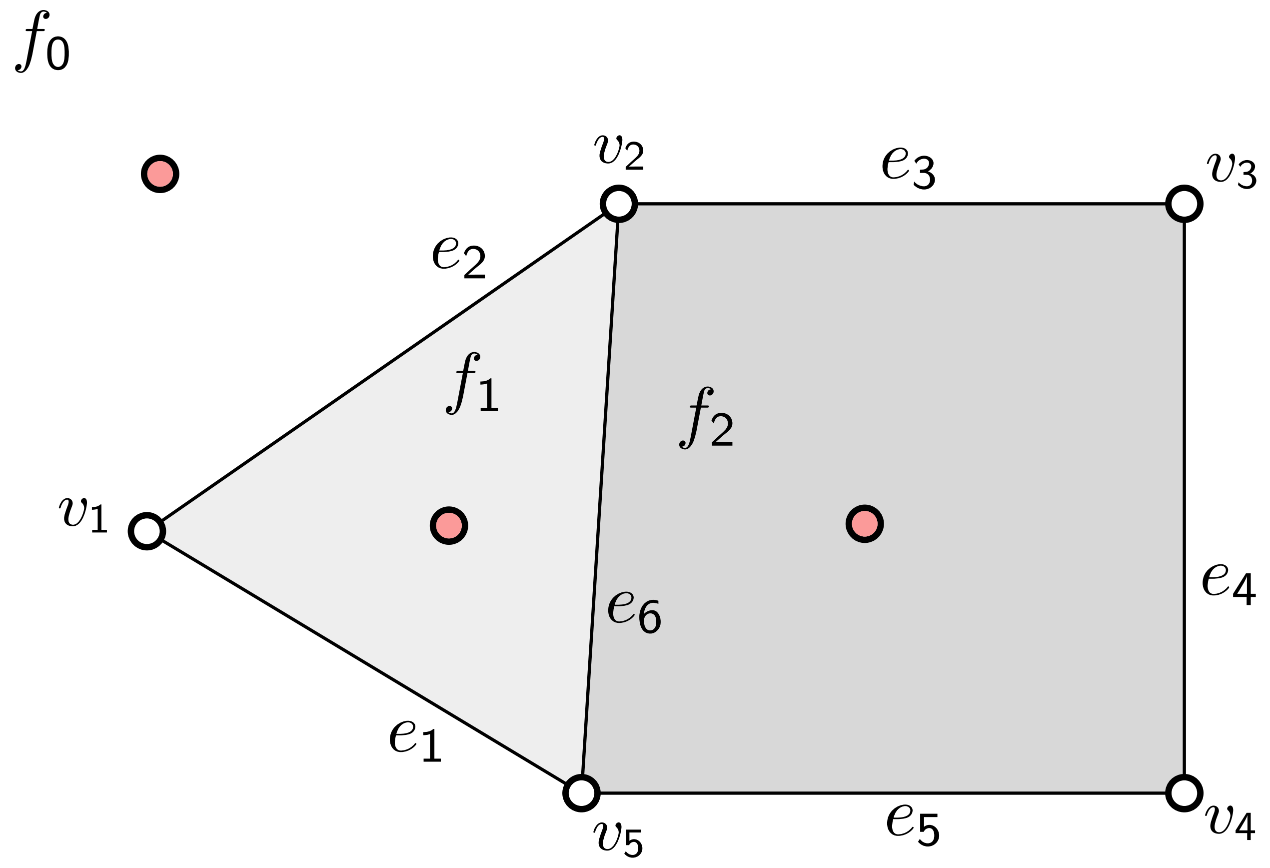
Flow Network Example

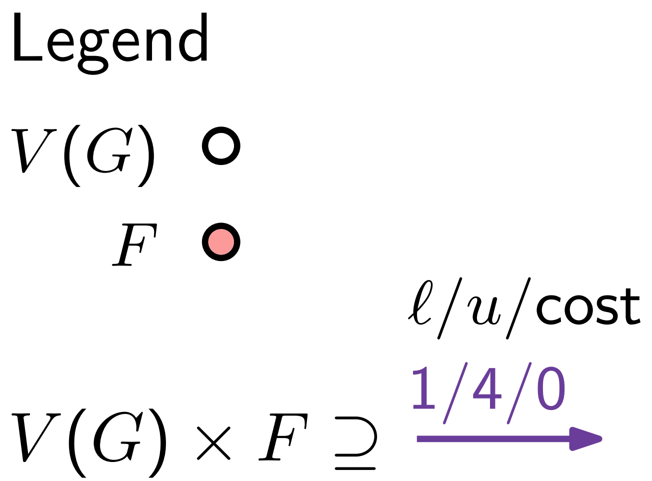


Flow Network Example

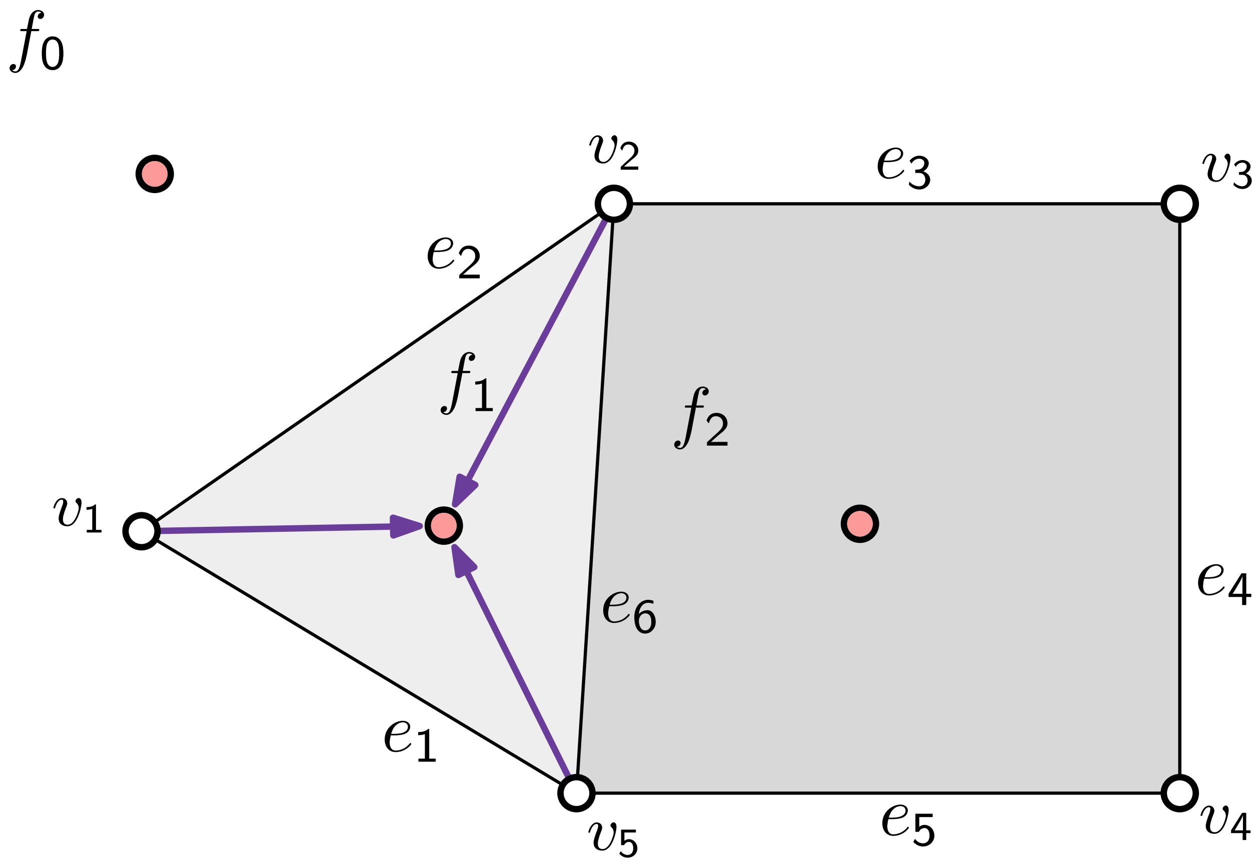


Flow Network Example





Flow Network Example



Legend

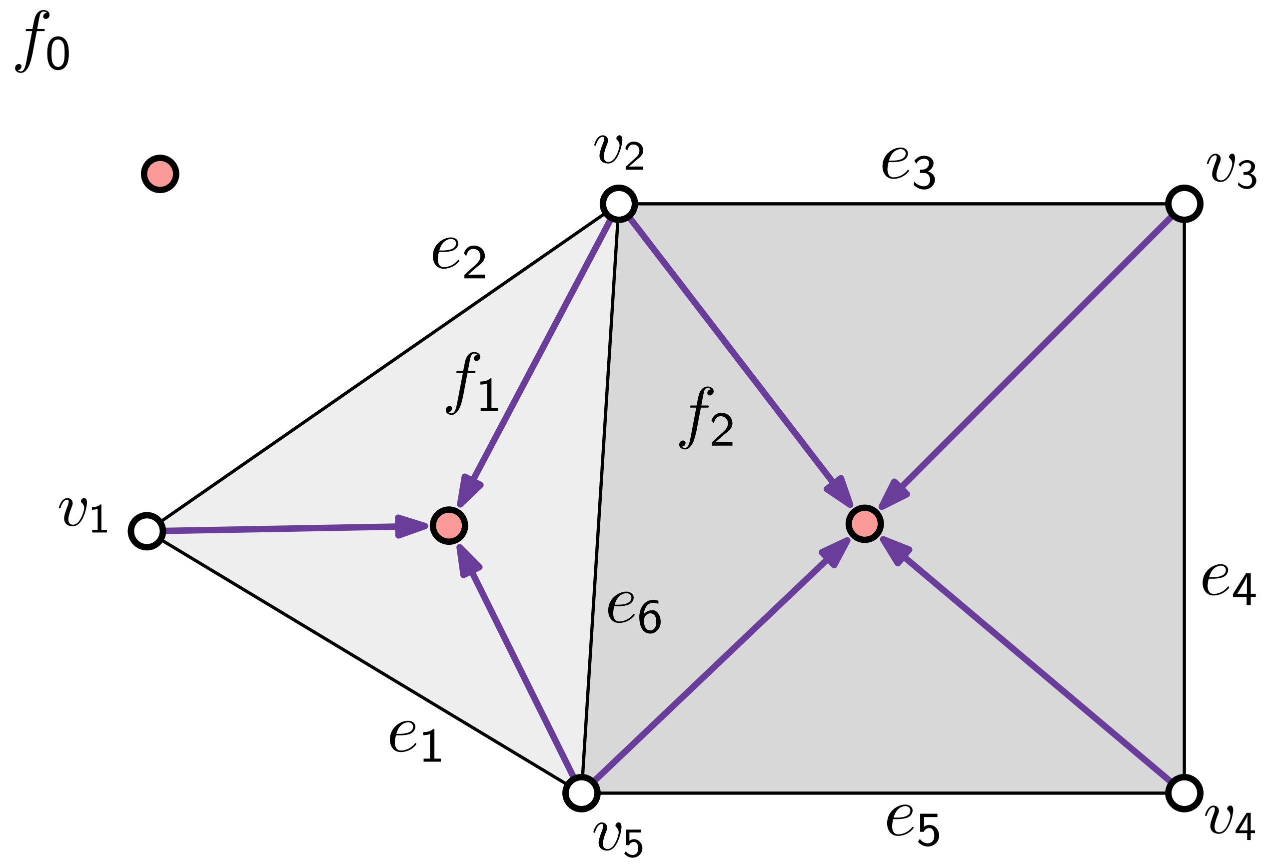
$V(G)$

F

$\ell/u/$cost

$V(G) \times F \supseteq$ 1/4/0

Flow Network Example



Legend

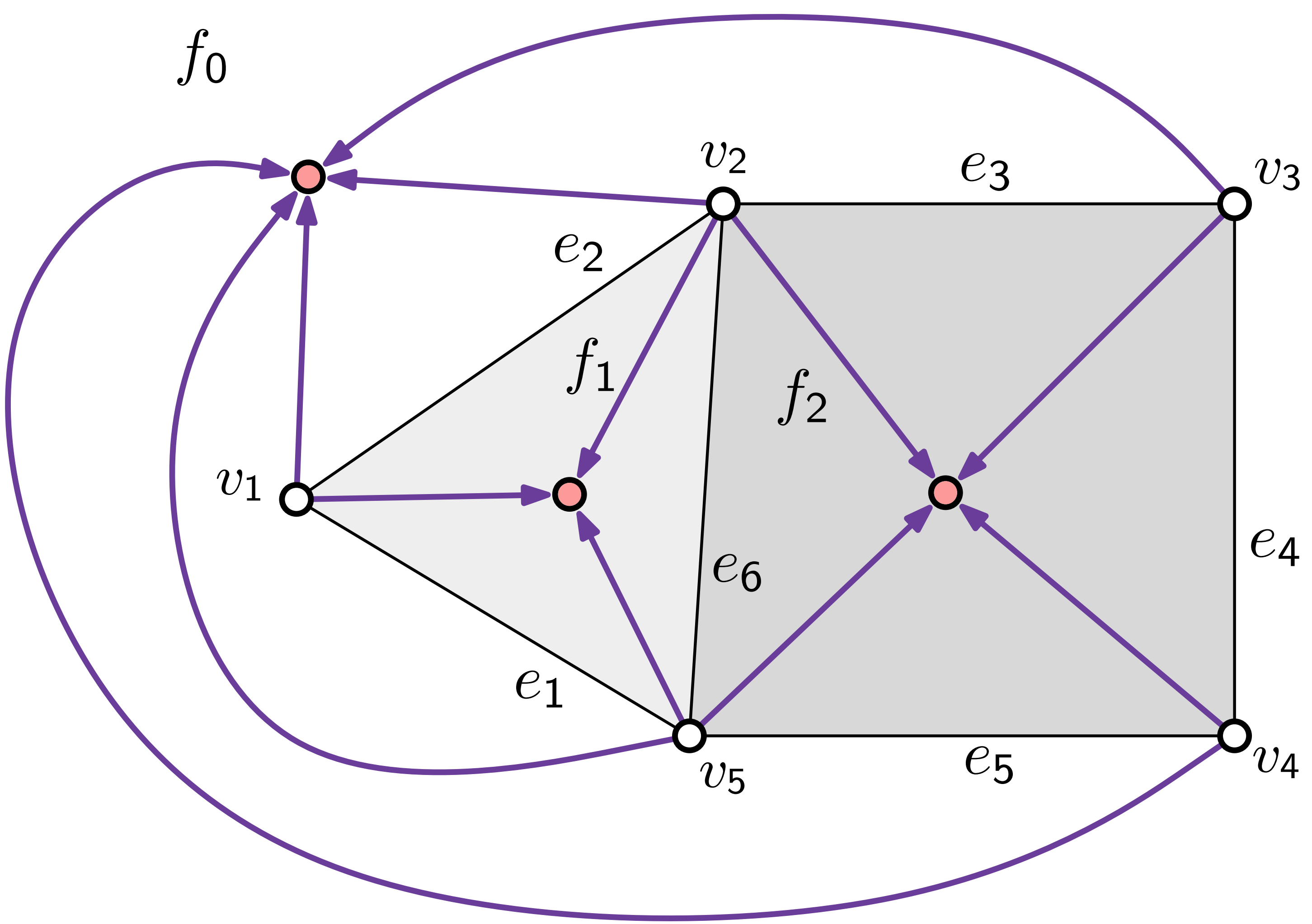
$V(G)$ ○

F ●

$\ell/u/$cost

$V(G) \times F \supseteq$ 1/4/0 →

Flow Network Example



Legend

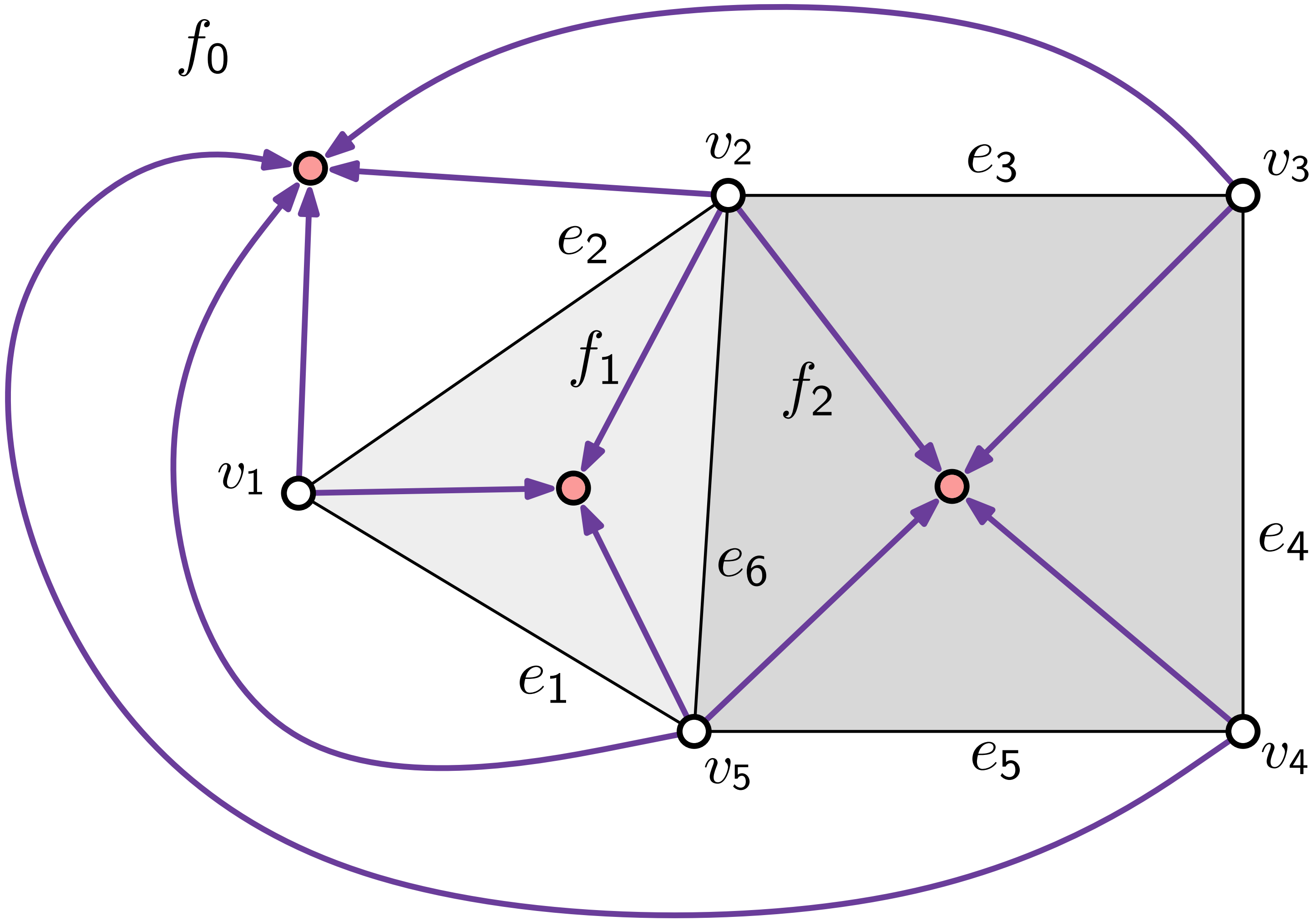
$V(G)$ ○

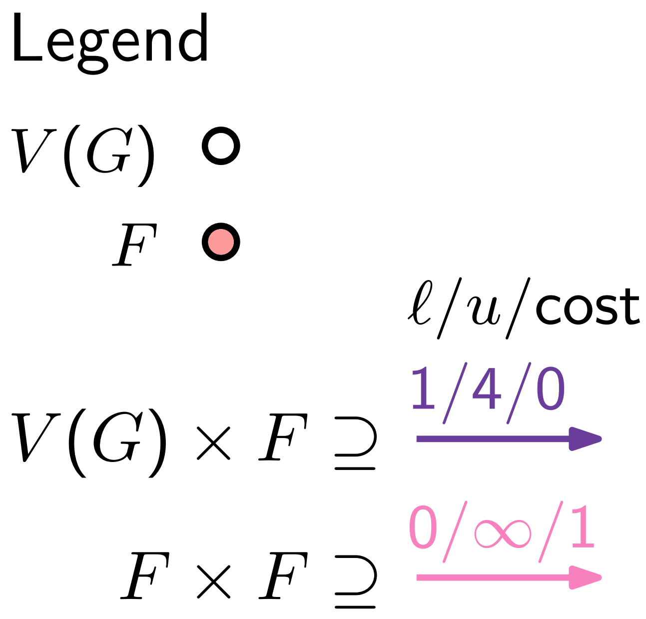
F ●

$\ell/u/$cost

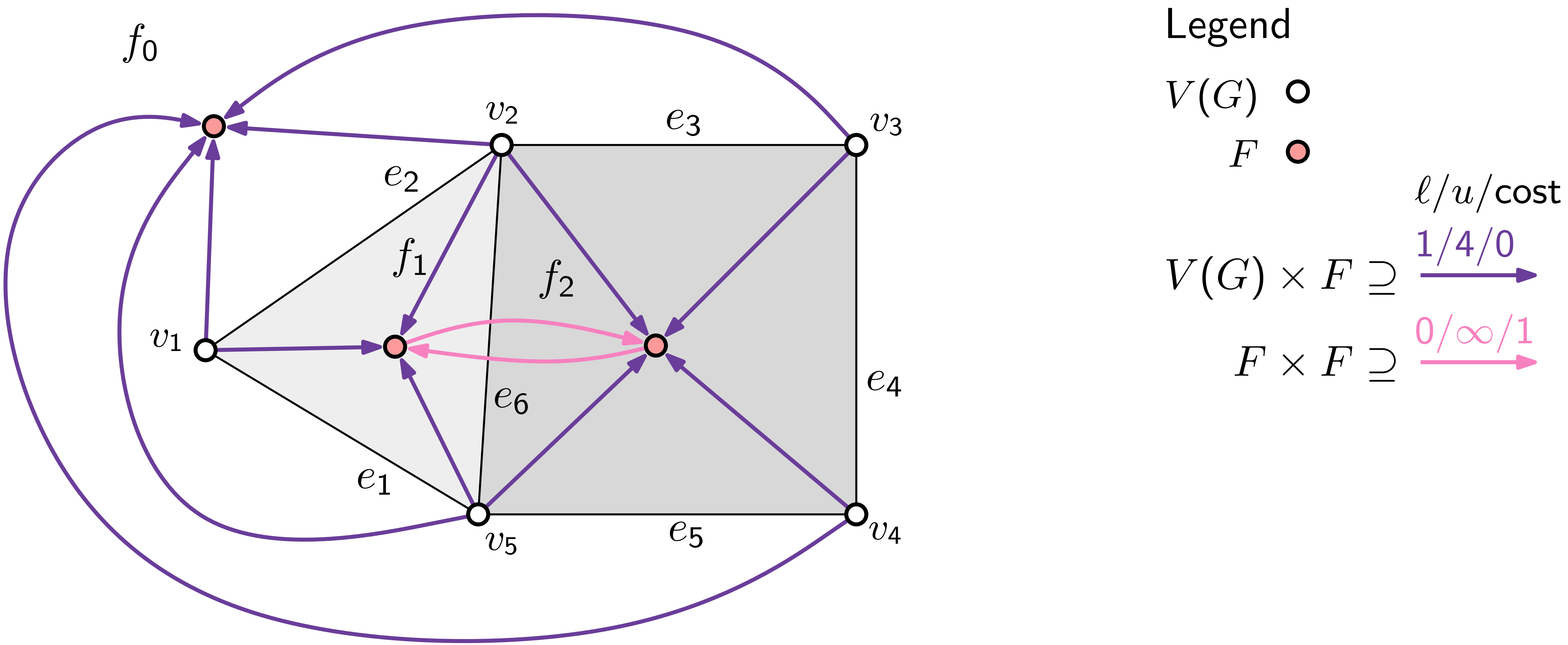
$V(G) \times F \supseteq$ $\xrightarrow{1/4/0}$

Flow Network Example

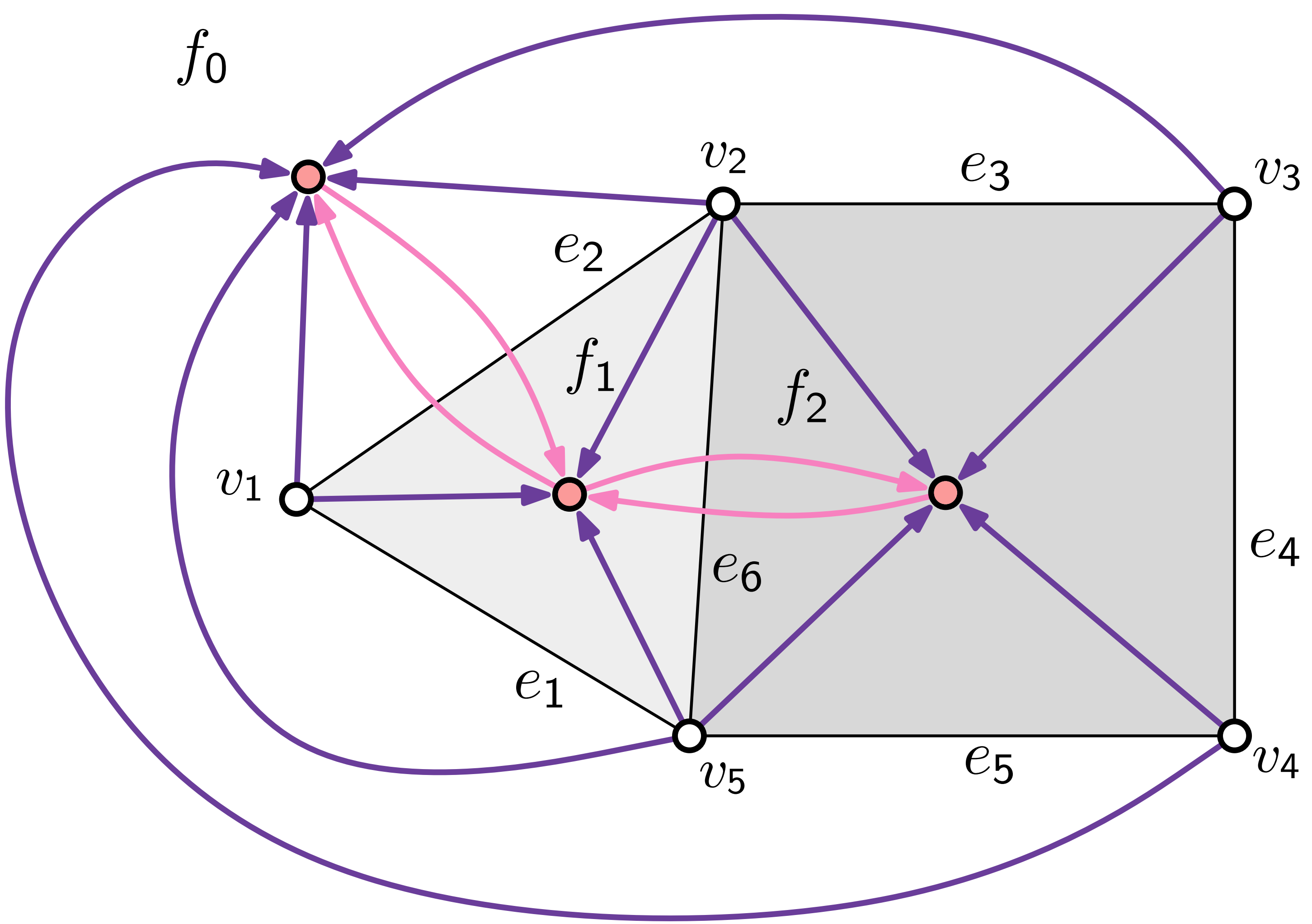


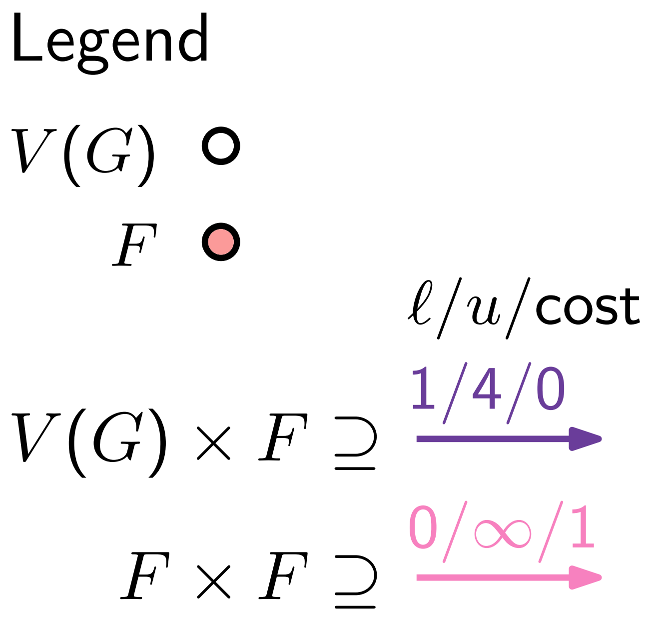


Flow Network Example

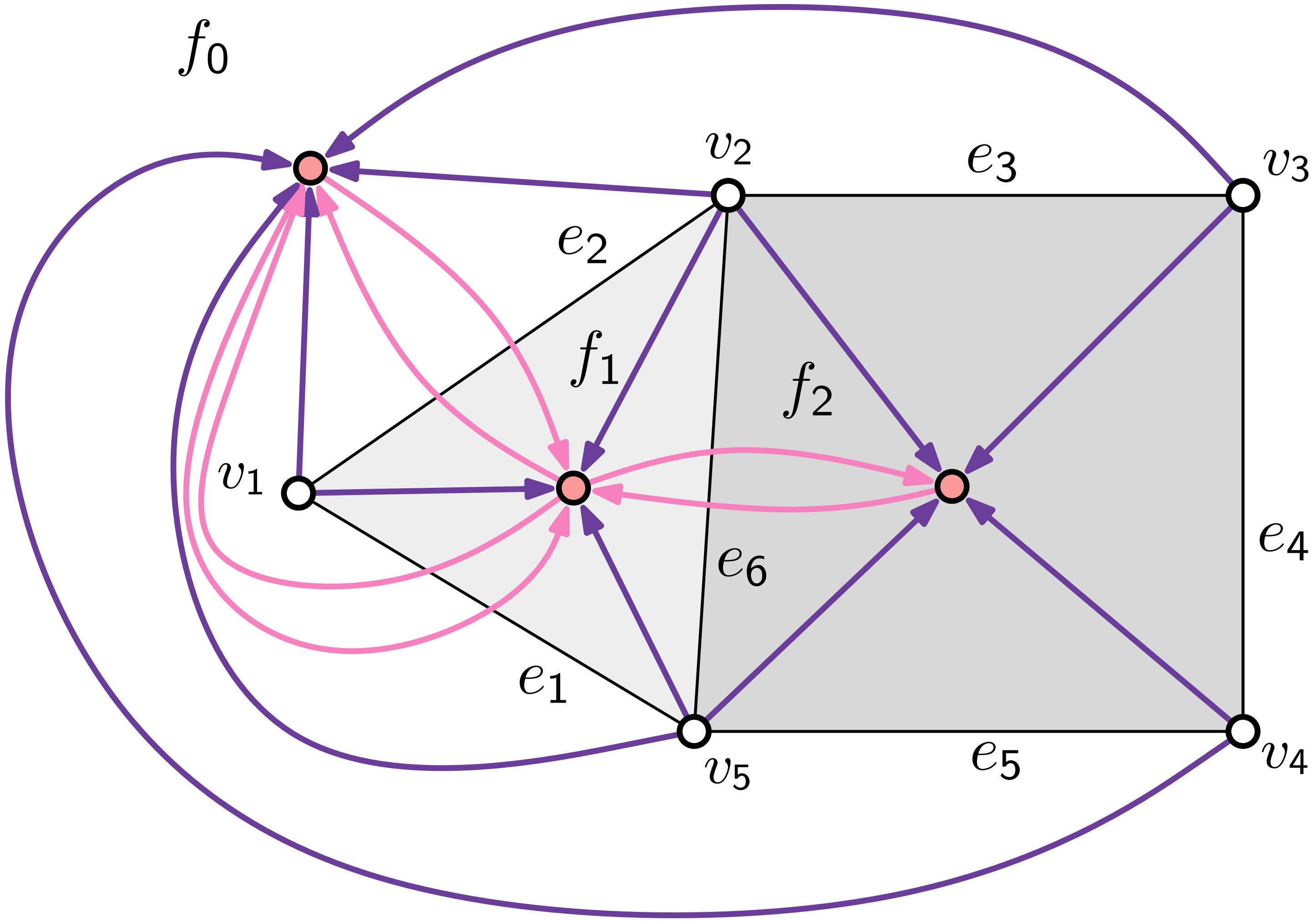


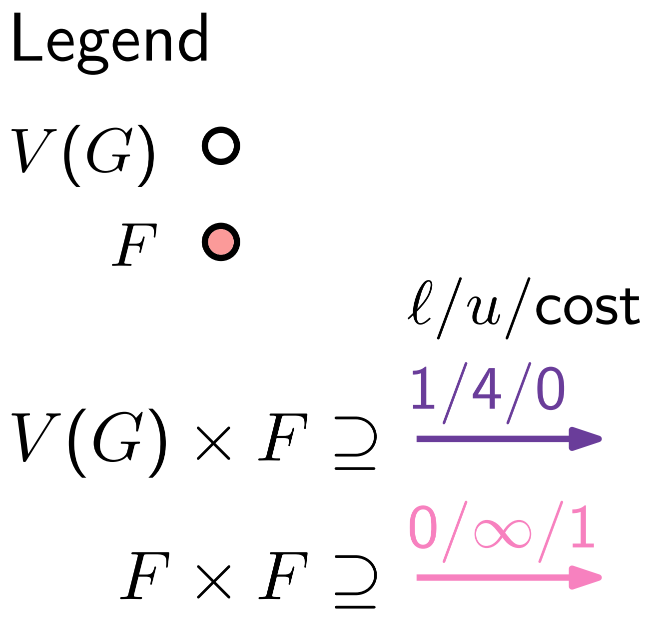
Flow Network Example



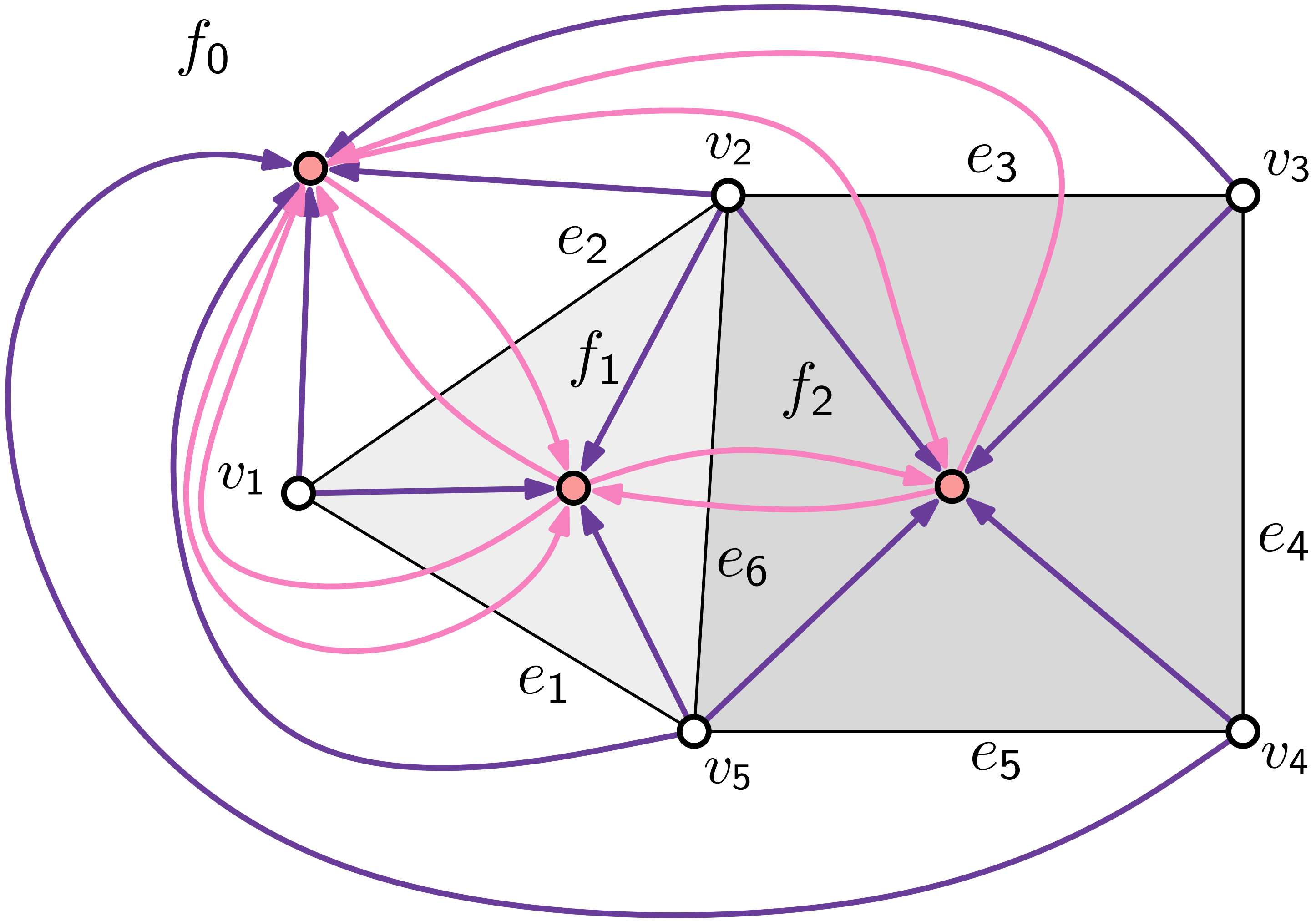


Flow Network Example





Flow Network Example



Legend

$V(G)$ ○

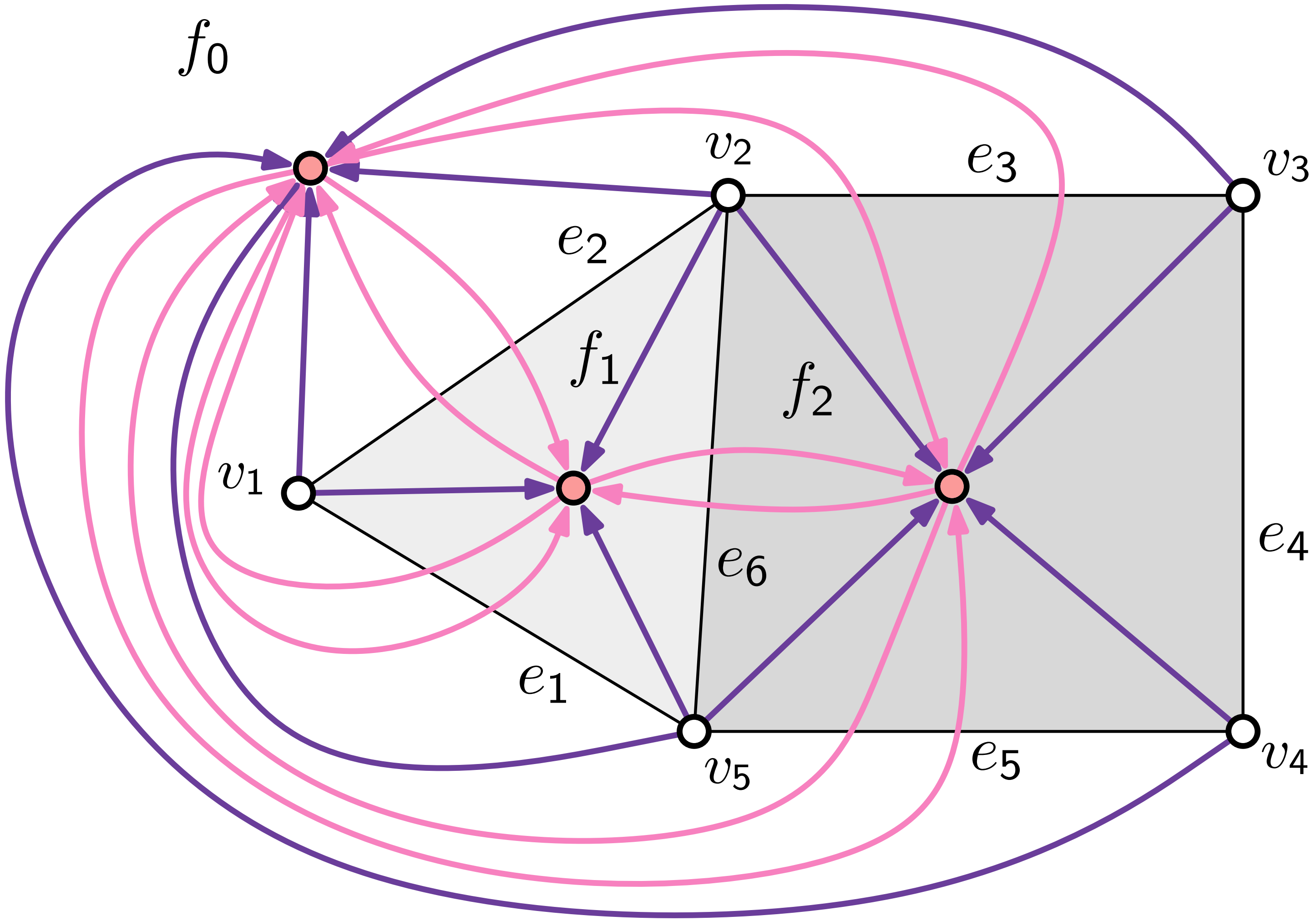
F ●

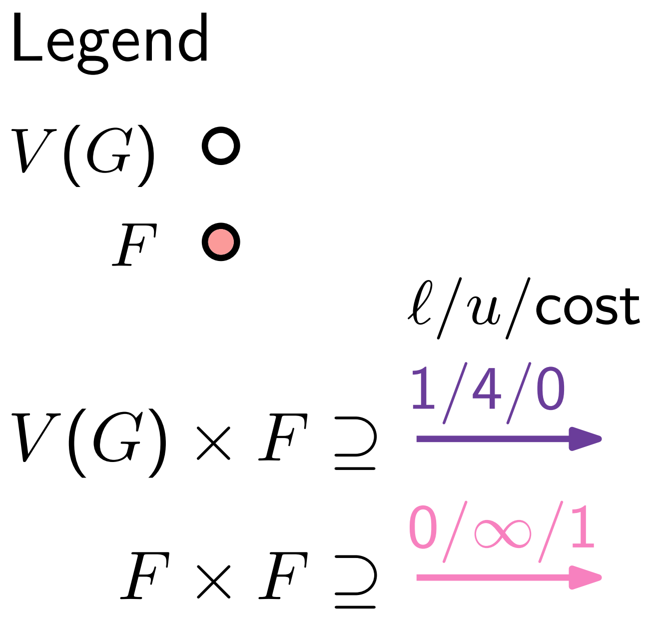
$\ell/u/$cost

$V(G) \times F \supseteq$ 1/4/0 →

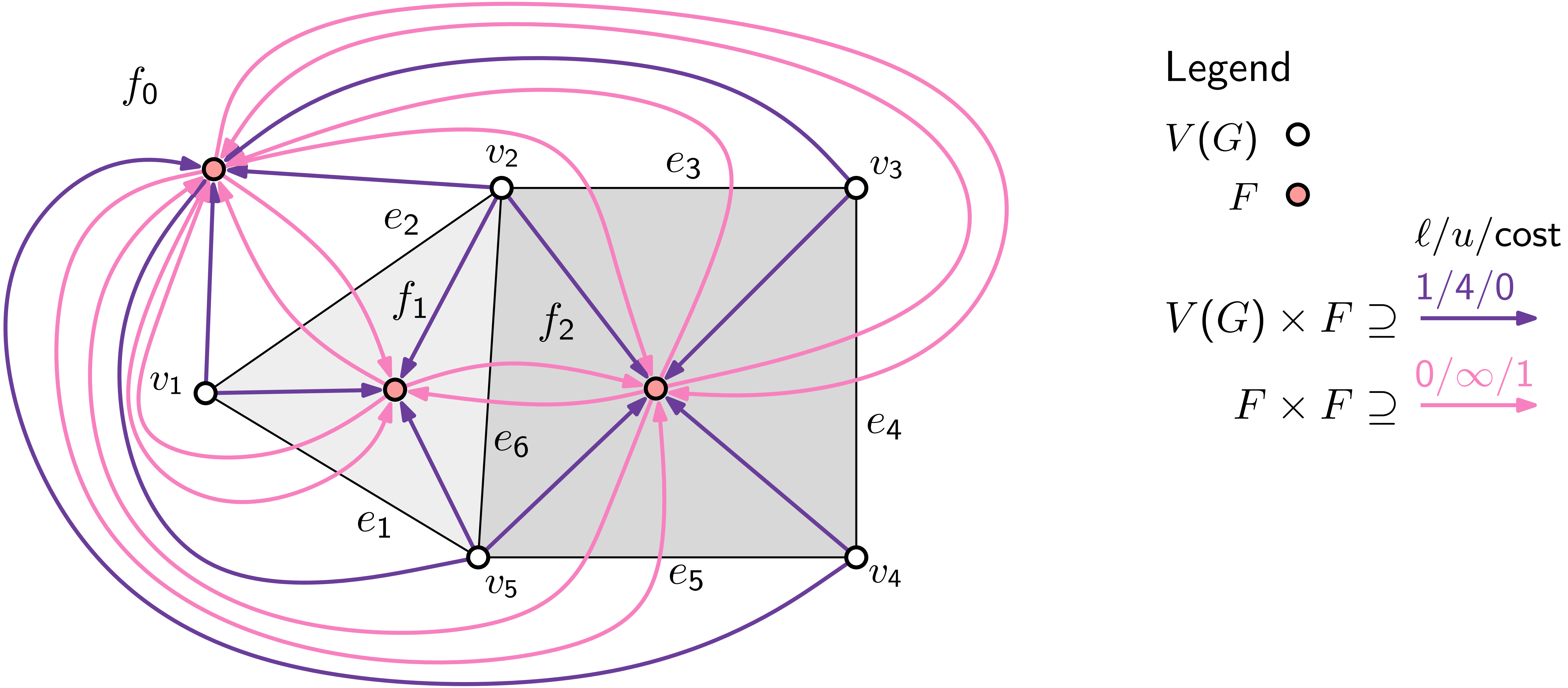
$F \times F \supseteq$ 0/∞/1 →

Flow Network Example

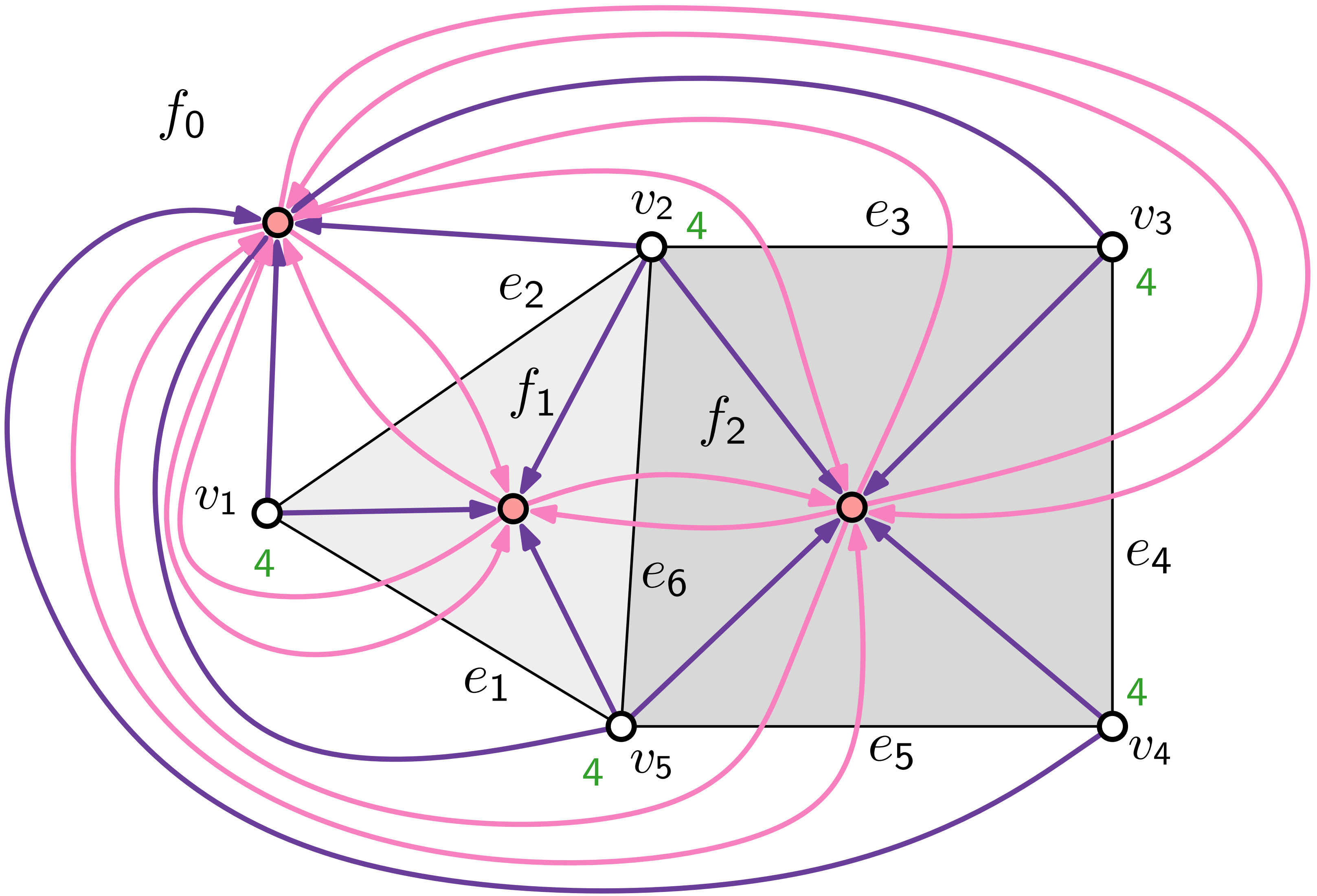




Flow Network Example



Flow Network Example



Legend

$V(G)$ ○

F ●

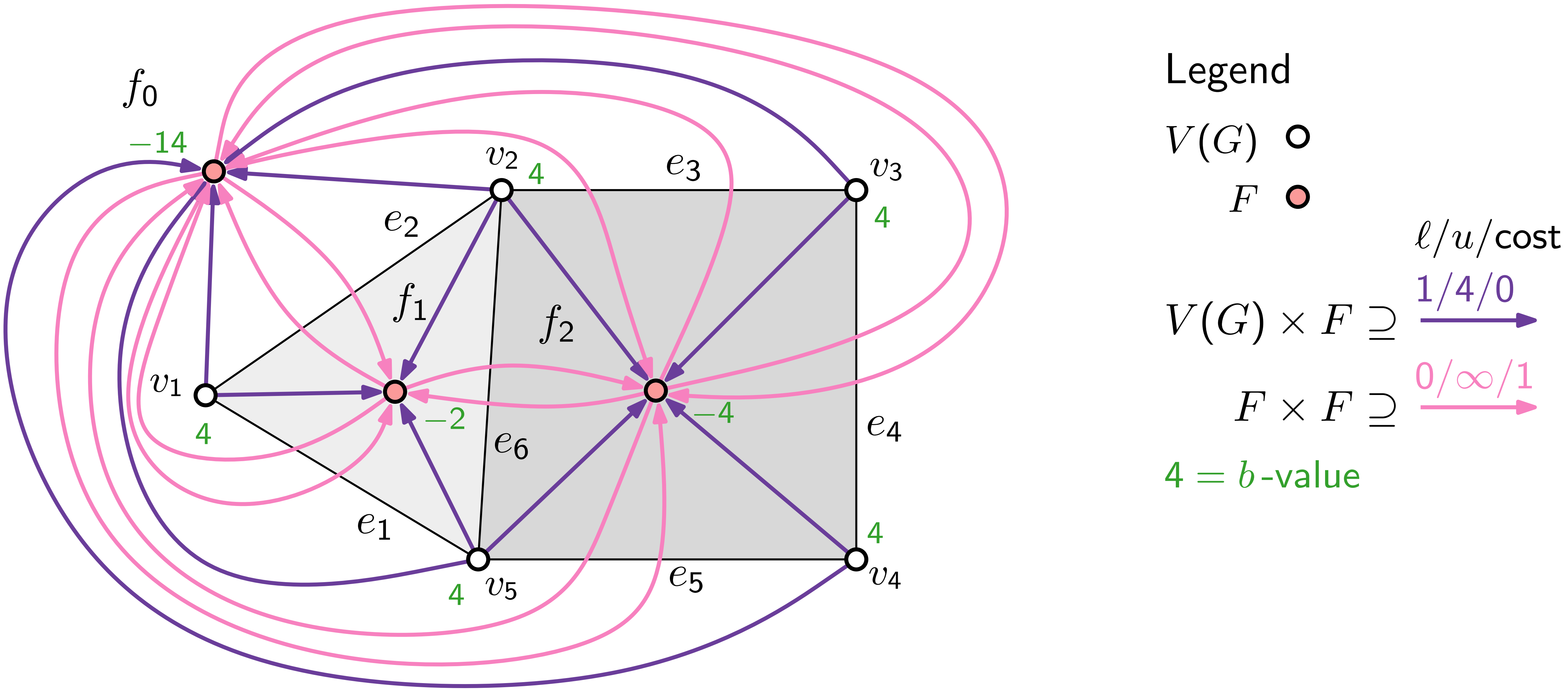
$\ell/u/$cost

$V(G) \times F \supseteq$ 1/4/0 →

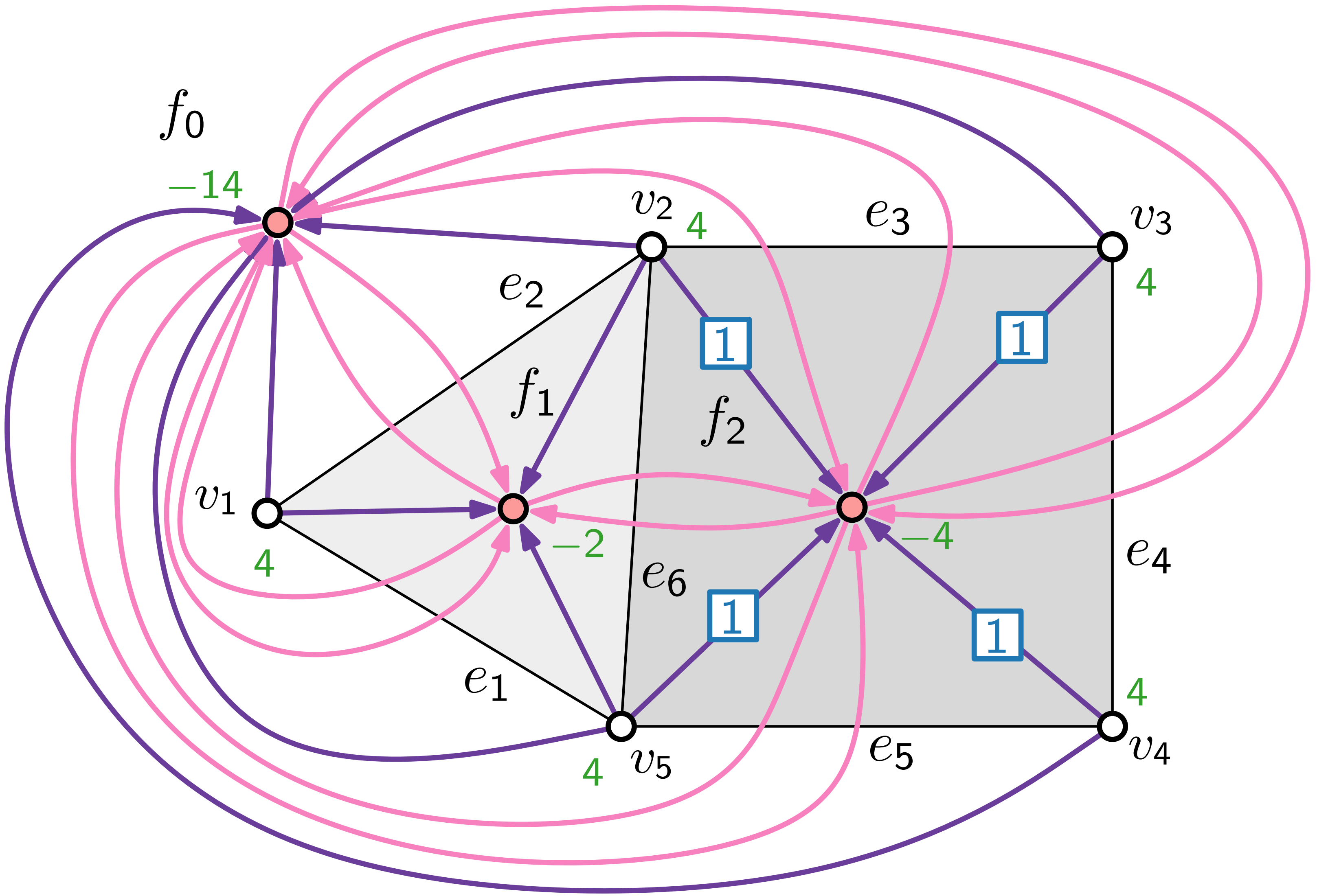
$F \times F \supseteq$ $0/\infty/1$ →

$4 = b$-value

Flow Network Example



Flow Network Example



Legend

$V(G)$ ○

F ●

$\ell/u/$cost

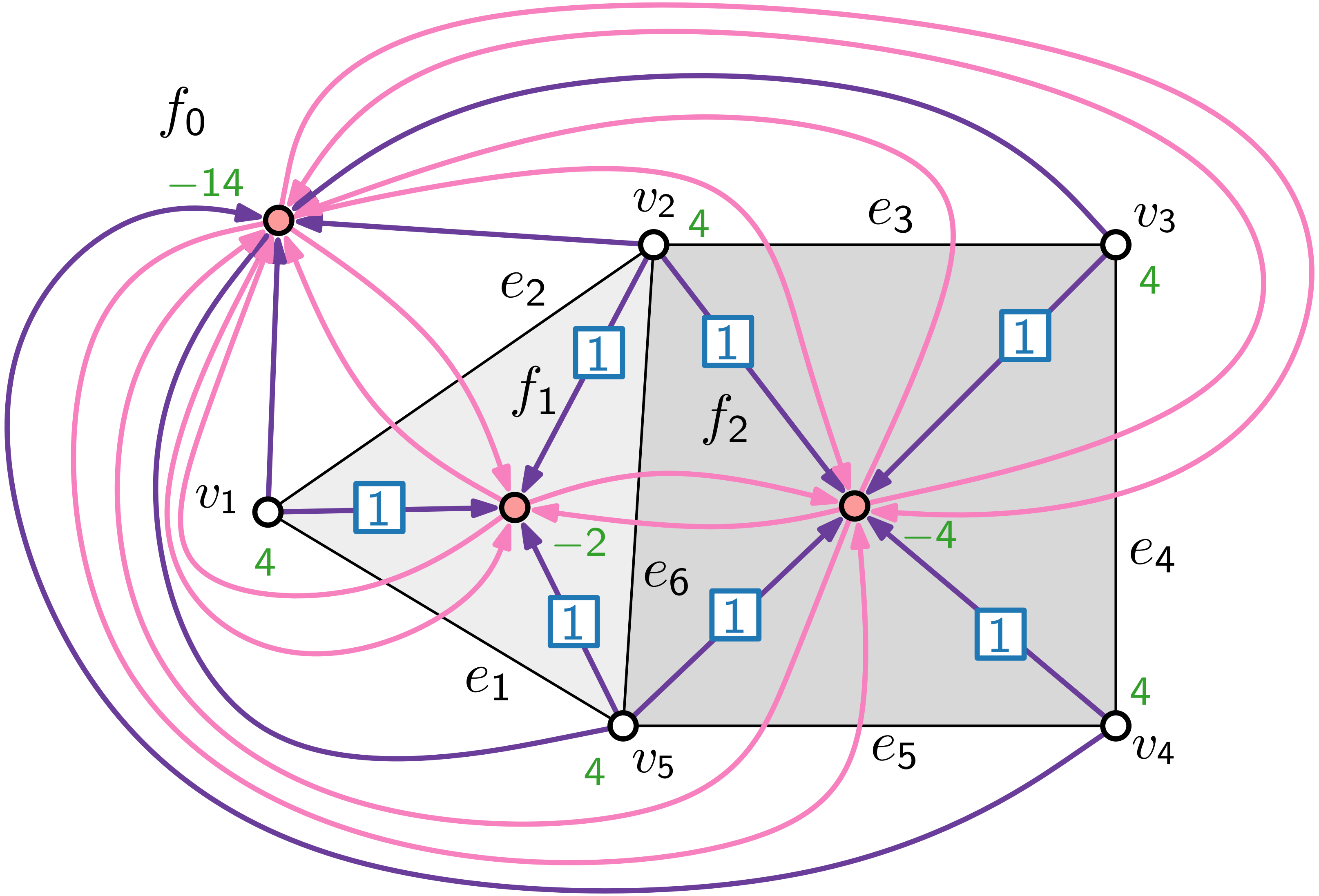
$V(G) \times F \supseteq$ 1/4/0 →

$F \times F \supseteq$ 0/∞/1 →

4 = b-value

3 flow

Flow Network Example



Legend

$V(G)$ ○

F ●

$\ell/u/\text{cost}$

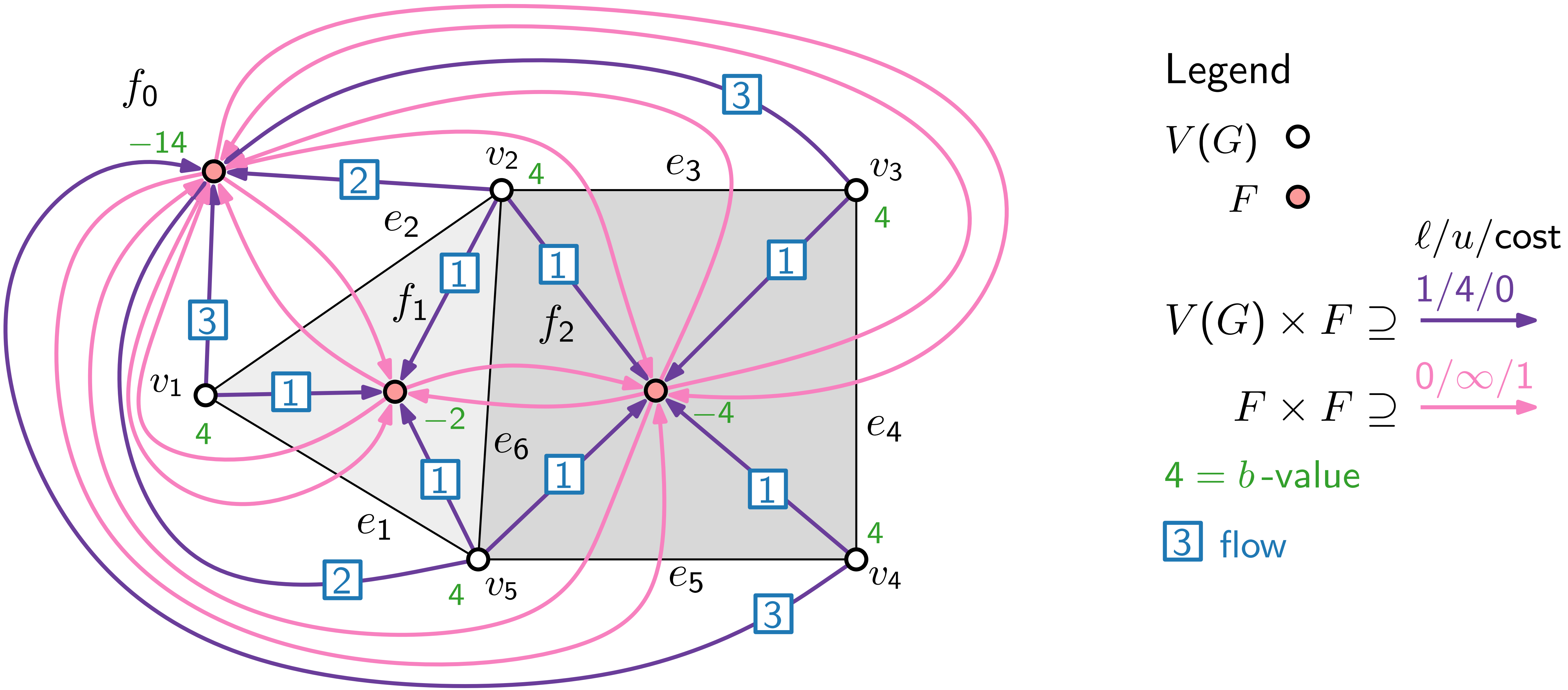
$V(G) \times F \supseteq$ 1/4/0 →

$F \times F \supseteq$ $0/\infty/1$ →

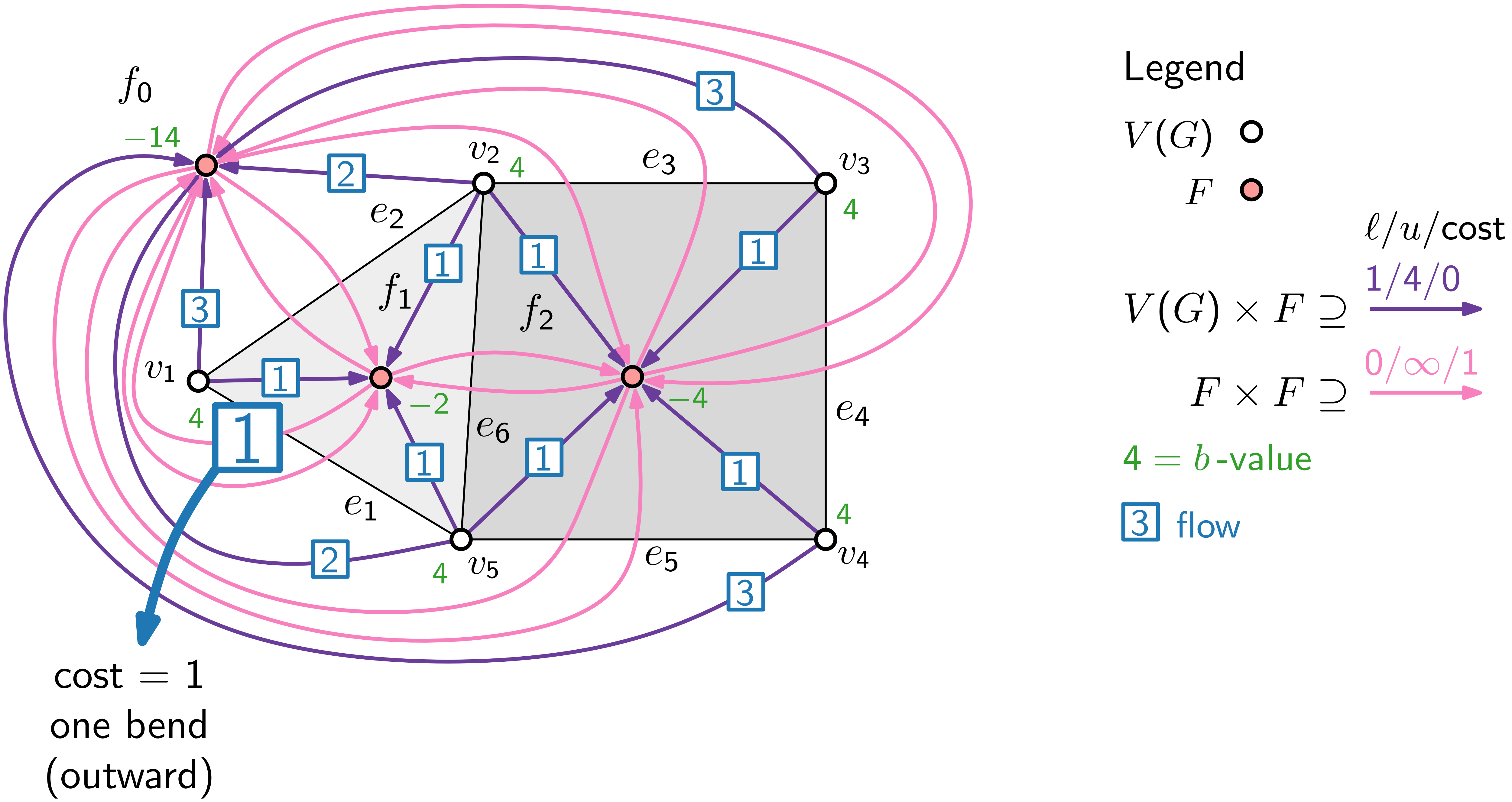
$4 = b$-value

3 flow

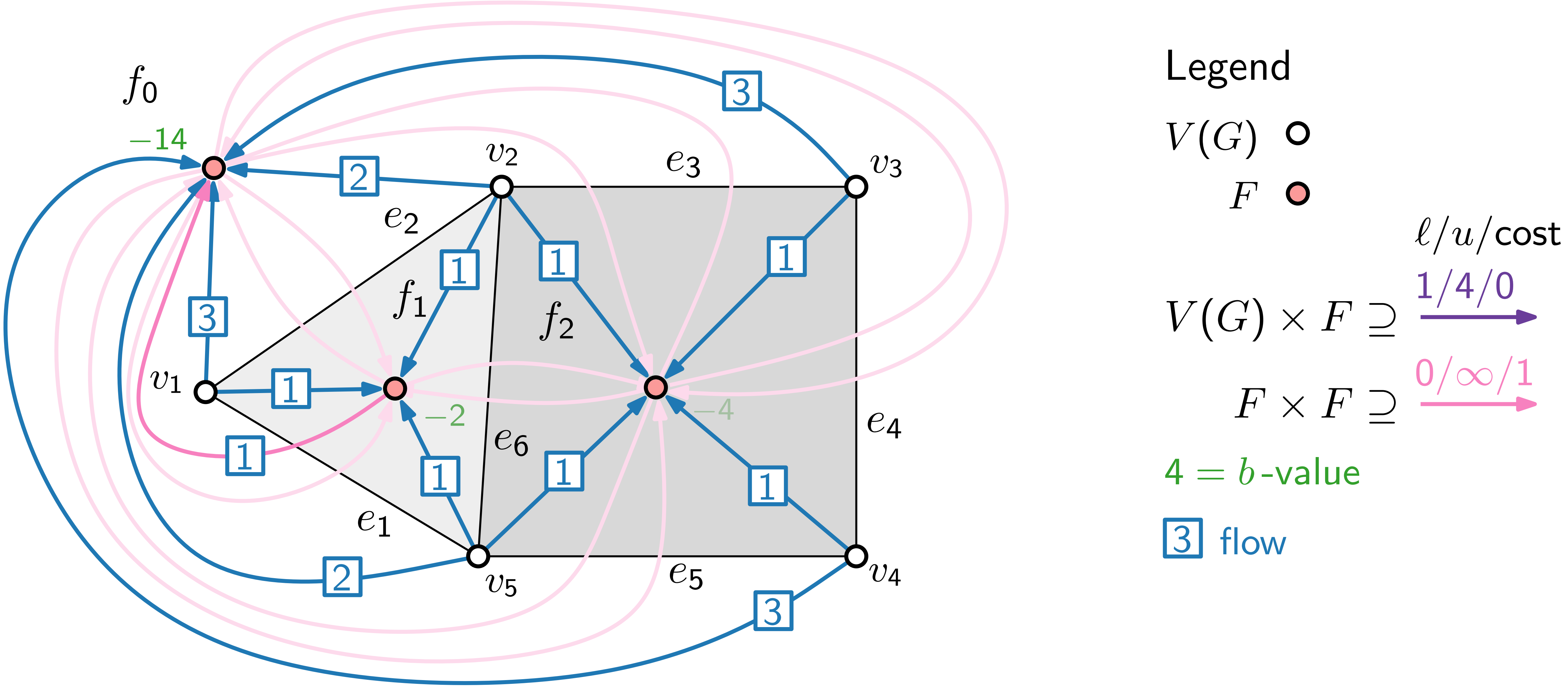
Flow Network Example



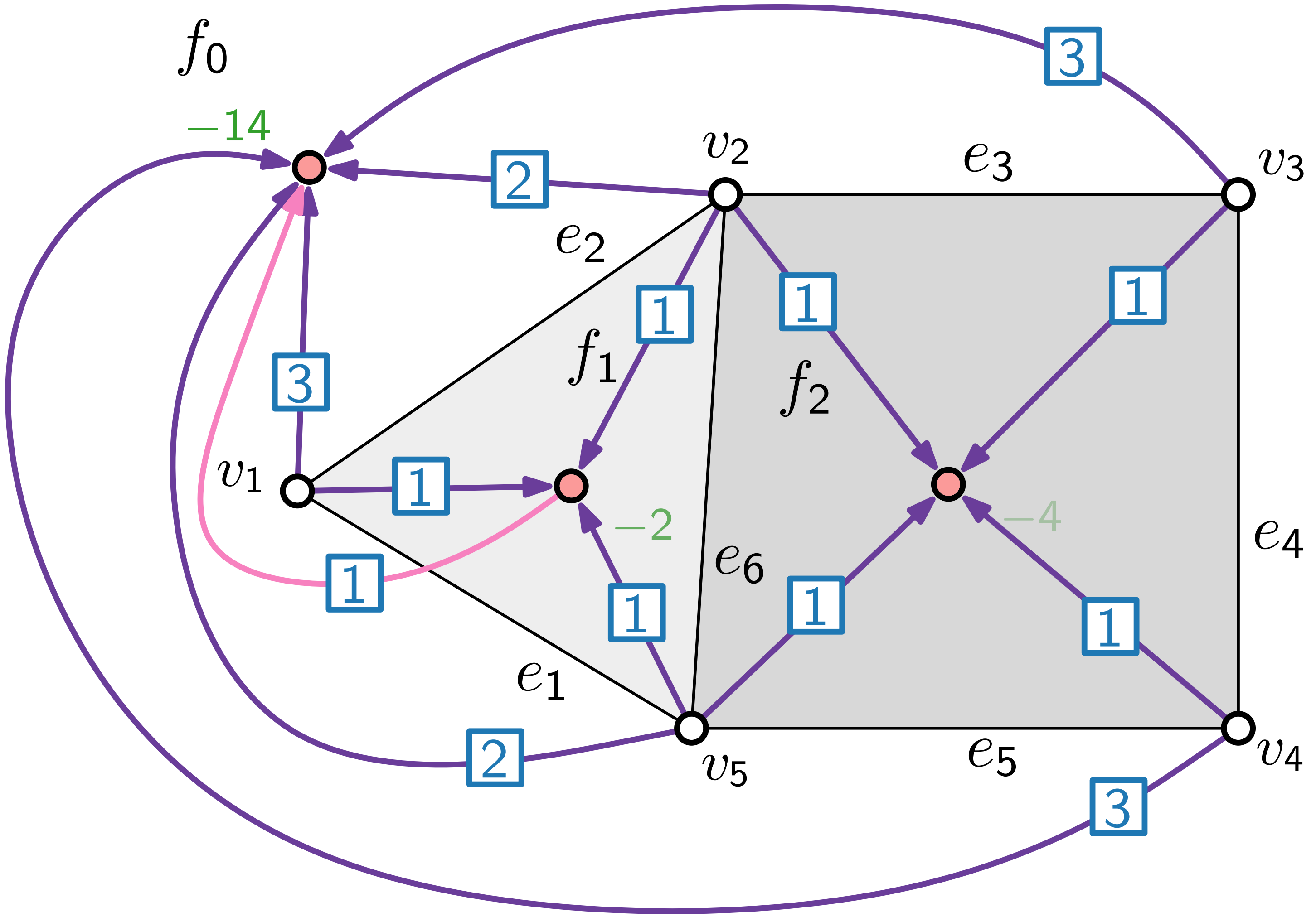
Flow Network Example



Flow Network Example



Flow Network Example



Legend

$V(G)$ ○

F ●

$\ell/u/\text{cost}$

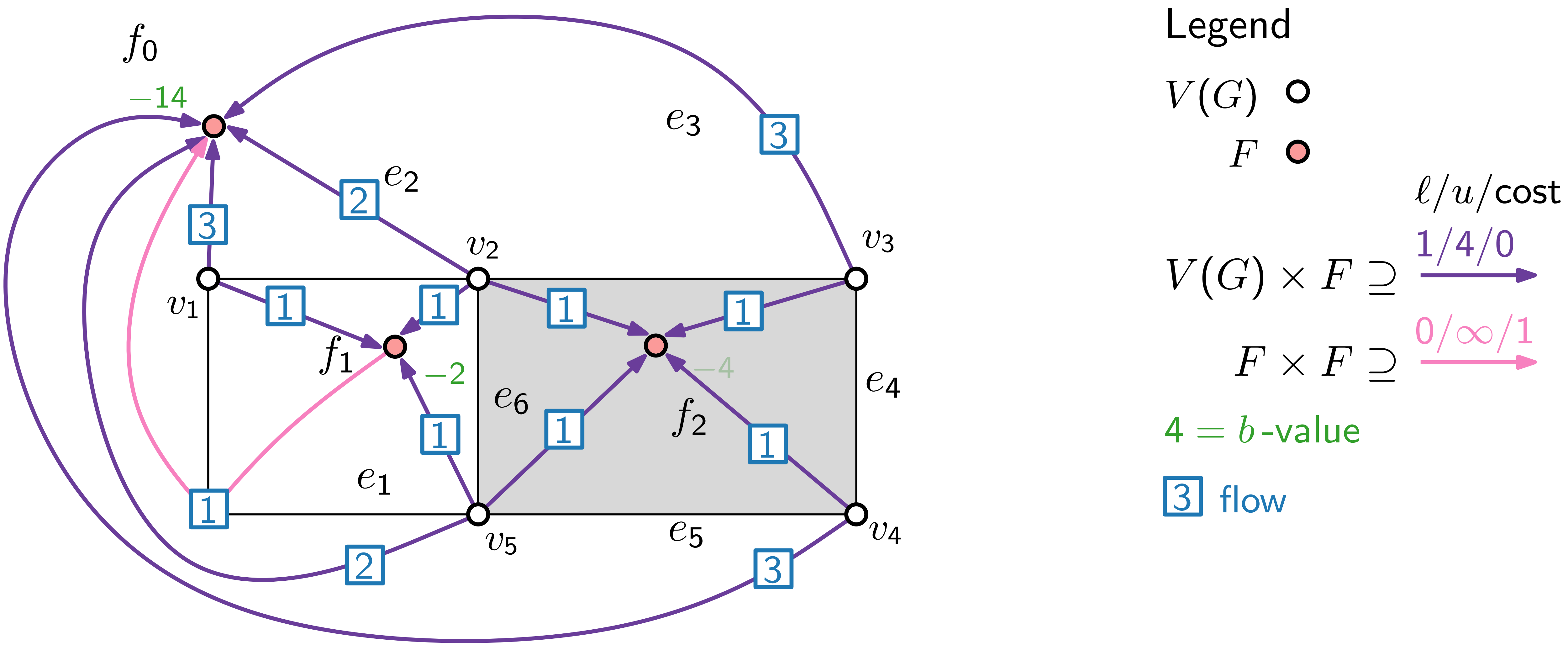
$V(G) \times F \supseteq$ 1/4/0 →

$F \times F \supseteq$ $0/\infty/1$ →

$4 = b$-value

3 flow

Flow Network Example



Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$

The flow network $N(G)$ has a valid flow X with cost k.

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

Proof.

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

Proof.
"$\Leftarrow$": Given a valid flow X in $N(G)$ of cost k,

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

Proof.
"$\Leftarrow$": Given a valid flow X in $N(G)$ of cost k,
construct an orthogonal representation $H(G)$ with k bends.

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

Proof.
"$\Leftarrow$": Given a valid flow X in $N(G)$ of cost k,
construct an orthogonal representation $H(G)$ with k bends.

- Transform from flow to orthogonal description.

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, sequence δ_1 is reversed and inverted δ_2.

(H3) For each **face** f it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v the sum of incident angles is 2π.

Proof.

"$\Leftarrow$": Given a valid flow X in $N(G)$ of cost k, construct an orthogonal representation $H(G)$ with k bends.

- Transform from flow to orthogonal description.
- Show properties (H1)–(H4).

Bend Minimization – Result

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, sequence δ_1 is reversed and inverted δ_2.

(H3) For each **face** f it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v the sum of incident angles is 2π.

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

Proof.

"$\Leftarrow$": Given a valid flow X in $N(G)$ of cost k, construct an orthogonal representation $H(G)$ with k bends.

- Transform from flow to orthogonal description.
- Show properties (H1)–(H4).

(H1) H(G) matches F, f_0 ✓

Bend Minimization – Result

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, sequence δ_1 is reversed and inverted δ_2.

(H3) For each **face** f it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v the sum of incident angles is 2π.

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

Proof.
"$\Leftarrow$": Given a valid flow X in $N(G)$ of cost k, construct an orthogonal representation $H(G)$ with k bends.

- Transform from flow to orthogonal description.
- Show properties (H1)–(H4).

(H1) H(G) matches F, f_0 ✓

(H4) Total angle at each vertex $= 2\pi$ ✓

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, sequence δ_1 is reversed and inverted δ_2.

(H3) For each **face** f it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v the sum of incident angles is 2π.

Proof.

"$\Leftarrow$": Given a valid flow X in $N(G)$ of cost k, construct an orthogonal representation $H(G)$ with k bends.

- Transform from flow to orthogonal description.
- Show properties (H1)–(H4).

(H1) H(G) matches F, f_0 ✓

(H2) Bend order inverted and reversed on opposite sides ✓

(H4) Total angle at each vertex $= 2\pi$ ✓

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, sequence δ_1 is reversed and inverted δ_2.

(H3) For each **face** f it holds that:

$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

(H4) For each **vertex** v the sum of incident angles is 2π.

Proof.

"$\Leftarrow$": Given a valid flow X in $N(G)$ of cost k, construct an orthogonal representation $H(G)$ with k bends.

- Transform from flow to orthogonal description.
- Show properties (H1)–(H4).

(H1) H(G) matches F, f_0 ✓

(H2) Bend order inverted and reversed on opposite sides ✓

(H3) Angle sum of $f = \pm 4$ ✓ $\rightarrow$ *Exercise.*

(H4) Total angle at each vertex $= 2\pi$ ✓

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

Proof.
"$\Rightarrow$": Given an orthogonal representation $H(G)$ with k bends, construct a valid flow X in $N(G)$ of cost k.

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

Proof.
"$\Rightarrow$": Given an orthogonal representation $H(G)$ with k bends, construct a valid flow X in $N(G)$ of cost k.

- Define flow $X \colon E' \to \mathbb{R}_0^+$.
- Show that X is a valid flow and has cost k.

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$
- $\ell(v, f) := 1 \leq X(v, f) \leq 4 =: u(v, f)$
  $\text{cost}(v, f) = 0$
  $\ell(f, g) := 0 \leq X(f, g) \leq \infty =: u(f, g)$
  $\text{cost}(f, g) = 1$

Proof.

"$\Rightarrow$": Given an orthogonal representation $H(G)$ with k bends, construct a valid flow X in $N(G)$ of cost k.

- Define flow $X \colon E' \to \mathbb{R}_0^+$.
- Show that X is a valid flow and has cost k.

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$
- $\ell(v, f) := 1 \leq X(v, f) \leq 4 =: u(v, f)$
  $\text{cost}(v, f) = 0$
  $\ell(f, g) := 0 \leq X(f, g) \leq \infty =: u(f, g)$
  $\text{cost}(f, g) = 1$

Proof.

"$\Rightarrow$": Given an orthogonal representation $H(G)$ with k bends, construct a valid flow X in $N(G)$ of cost k.

- Define flow $X\colon E' \to \mathbb{R}_0^+$.
- Show that X is a valid flow and has cost k.

(N1) $X(vf) = 1/2/3/4$ ✓

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$
- $\ell(v,f) := 1 \leq X(v,f) \leq 4 =: u(v,f)$
  $\text{cost}(v,f) = 0$
  $\ell(f,g) := 0 \leq X(f,g) \leq \infty =: u(f,g)$
  $\text{cost}(f,g) = 1$

Proof.

"$\Rightarrow$": Given an orthogonal representation $H(G)$ with k bends, construct a valid flow X in $N(G)$ of cost k.

- Define flow $X \colon E' \to \mathbb{R}_0^+$.
- Show that X is a valid flow and has cost k.

(N1) $X(vf) = 1/2/3/4$ ✓

(N2) $X((fg)_e) = |\delta|_0$, where (e, δ, x) describes edge e in $H(f)$ ✓

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$
- $\ell(v, f) := 1 \leq X(v, f) \leq 4 =: u(v, f)$
  $\text{cost}(v, f) = 0$
  $\ell(f, g) := 0 \leq X(f, g) \leq \infty =: u(f, g)$
  $\text{cost}(f, g) = 1$

Proof.

"$\Rightarrow$": Given an orthogonal representation $H(G)$ with k bends, construct a valid flow X in $N(G)$ of cost k.

- Define flow $X \colon E' \to \mathbb{R}_0^+$.
- Show that X is a valid flow and has cost k.

(N1) $X(vf) = 1/2/3/4$ ✓

(N2) $X((fg)_e) = |\delta|_0$, where (e, δ, x) describes edge e in $H(f)$ ✓

(N3) capacities, deficit/demand coverage ✓

Bend Minimization – Result

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends. $\Leftrightarrow$
The flow network $N(G)$ has a valid flow X with cost k.

- $b(v) = 4 \quad \forall v \in V(G)$
- $b(f) = -2\deg_G(f) + \begin{cases} -4 & \text{if } f = f_0, \\ +4 & \text{otherwise} \end{cases}$
- $\ell(v, f) := 1 \leq X(v, f) \leq 4 =: u(v, f)$
  $\text{cost}(v, f) = 0$
  $\ell(f, g) := 0 \leq X(f, g) \leq \infty =: u(f, g)$
  $\text{cost}(f, g) = 1$

Proof.

"$\Rightarrow$": Given an orthogonal representation $H(G)$ with k bends, construct a valid flow X in $N(G)$ of cost k.

- Define flow $X\colon E' \to \mathbb{R}_0^+$.
- Show that X is a valid flow and has cost k.

(N1) $X(vf) = 1/2/3/4$ ✓

(N2) $X((fg)_e) = |\delta|_0$, where (e, δ, x) describes edge e in $H(f)$ ✓

(N3) capacities, deficit/demand coverage ✓

(N4) cost $= k$ ✓

Bend Minimization – Remarks

- The theorem implies that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for min-cost flow.

Bend Minimization – Remarks

- The theorem implies that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for min-cost flow.

Theorem. [Garg & Tamassia 1996]
The min-cost flow problem for planar graphs with bounded costs and vertex degrees can be solved in $O(n^{7/4}\sqrt{\log n})$ time.

Bend Minimization – Remarks

- The theorem implies that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for min-cost flow.

Theorem. [Garg & Tamassia 1996]
The min-cost flow problem for planar graphs with bounded costs and vertex degrees can be solved in $O(n^{7/4}\sqrt{\log n})$ time.

Theorem. [van den Brand, Chen, Kyng, Liu, Peng, Probst, Sachdeva, Sidford 2023]
The minimum-cost flow problem with integral vertex demands, edge capacities & costs can be solved in $O(m^{1+o(1)} \log U \log C)$ time where U is max. capacity and C are max. costs.

Bend Minimization – Remarks

- The theorem implies that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for min-cost flow.

Theorem. [Garg & Tamassia 1996]
The min-cost flow problem for planar graphs with bounded costs and vertex degrees can be solved in $O(n^{7/4}\sqrt{\log n})$ time.

Theorem. [van den Brand, Chen, Kyng, Liu, Peng, Probst, Sachdeva, Sidford 2023]
The minimum-cost flow problem with integral vertex demands, edge capacities & costs can be solved in $O(m^{1+o(1)} \log U \log C)$ time where U is max. capacity and C are max. costs.

$m \in O(n)$ for planar graphs

Bend Minimization – Remarks

- The theorem implies that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for min-cost flow.

Theorem. [Garg & Tamassia 1996]
The min-cost flow problem for planar graphs with bounded costs and vertex degrees can be solved in $O(n^{7/4}\sqrt{\log n})$ time.

Theorem. [van den Brand, Chen, Kyng, Liu, Peng, Probst, Sachdeva, Sidford 2023]
The minimum-cost flow problem with integral vertex demands, edge capacities & costs can be solved in $O(m^{1+o(1)} \log U \log C)$ time where U is max. capacity and C are max. costs.

$m \in O(n)$ for planar graphs $\quad$ $C \in \{0, 1\}$

Bend Minimization – Remarks

- The theorem implies that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for min-cost flow.

Theorem. [Garg & Tamassia 1996]
The min-cost flow problem for planar graphs with bounded costs and vertex degrees can be solved in $O(n^{7/4}\sqrt{\log n})$ time.

Theorem. [van den Brand, Chen, Kyng, Liu, Peng, Probst, Sachdeva, Sidford 2023]
The minimum-cost flow problem with integral vertex demands, edge capacities & costs can be solved in $O(m^{1+o(1)} \log U \log C)$ time where U is max. capacity and C are max. costs.

$U \in O(n)$ because $2n + 4$ bends in total are always sufficient [Storer 1984]

$m \in O(n)$ for planar graphs

$C \in \{0, 1\}$

Bend Minimization – Remarks

- The theorem implies that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for min-cost flow.

Theorem. [Garg & Tamassia 1996]
The min-cost flow problem for planar graphs with bounded costs and vertex degrees can be solved in $O(n^{7/4}\sqrt{\log n})$ time.

Theorem. [van den Brand, Chen, Kyng, Liu, Peng, Probst, Sachdeva, Sidford 2023]
The minimum-cost flow problem with integral vertex demands, edge capacities & costs can be solved in $O(m^{1+o(1)} \log U \log C)$ time where U is max. capacity and C are max. costs.

$U \in O(n)$ because $2n + 4$ bends in total are always sufficient [Storer 1984]

$m \in O(n)$ for planar graphs

$C \in \{0, 1\}$

Further, $\log n = n^{\log_n \log n} = n^{\log \log n / \log n} \in n^{o(1)}$ since $\lim_{n \to \infty} \frac{\log \log n}{\log n} = 0$

Bend Minimization – Remarks

- The theorem implies that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for min-cost flow.

Theorem. [Garg & Tamassia 1996]
The min-cost flow problem for planar graphs with bounded costs and vertex degrees can be solved in $O(n^{7/4}\sqrt{\log n})$ time.

Theorem. [van den Brand, Chen, Kyng, Liu, Peng, Probst, Sachdeva, Sidford 2023]
The minimum-cost flow problem with integral vertex demands, edge capacities & costs can be solved in $O(m^{1+o(1)} \log U \log C)$ time where U is max. capacity and C are max. costs.

$U \in O(n)$ because $2n+4$ bends in total are always sufficient [Storer 1984]

$m \in O(n)$ for planar graphs

$C \in \{0,1\}$

Further, $\log n = n^{\log_n \log n} = n^{\log\log n / \log n} \in n^{o(1)}$ since $\lim_{n\to\infty} \frac{\log\log n}{\log n} = 0$

Corollary.
The combinatorial orthogonal bend minimization problem can be solved in $O(n^{1+o(1)})$ time.

Bend Minimization – Remarks

- The theorem implies that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for min-cost flow.

Theorem. [Garg & Tamassia 1996]
The min-cost flow problem for planar graphs with bounded costs and vertex degrees can be solved in $O(n^{7/4}\sqrt{\log n})$ time.

Theorem. [van den Brand, Chen, Kyng, Liu, Peng, Probst, Sachdeva, Sidford 2023]
The minimum-cost flow problem with integral vertex demands, edge capacities & costs can be solved in $O(m^{1+o(1)} \log U \log C)$ time where U is max. capacity and C are max. costs.

$U \in O(n)$ because $2n+4$ bends in total are always sufficient [Storer 1984]

$m \in O(n)$ for planar graphs $C \in \{0,1\}$ Further, $\log n = n^{\log_n \log n} = n^{\log\log n / \log n} \in n^{o(1)}$ since $\lim_{n\to\infty} \frac{\log\log n}{\log n} = 0$

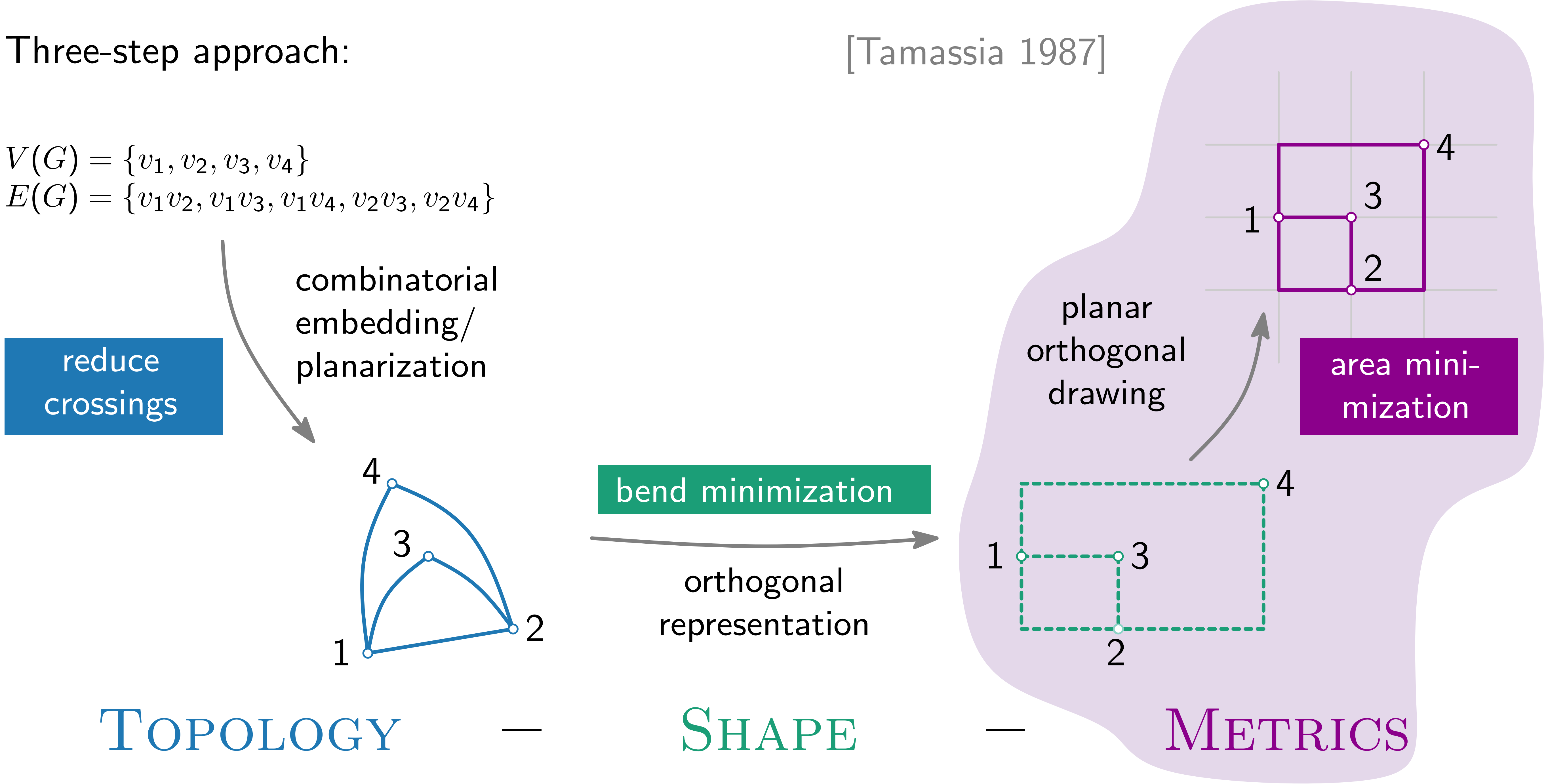
Corollary.
The combinatorial orthogonal bend minimization problem can be solved in $O(n^{1+o(1)})$ time.

Theorem. [Garg & Tamassia 2001]
Bend minimization without given combinatorial embedding is NP-hard.

Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

$V(G) = \{v_1, v_2, v_3, v_4\}$
$E(G) = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$



Compaction

Compaction problem.

Given:

Find:

Compaction

Compaction problem.

Given: ■ Plane graph G with maximum degree 4

Find:

Compaction

Compaction problem.

Given:
- Plane graph G with maximum degree 4
- Orthogonal representation $H(G)$

Find:

Compaction

Compaction problem.

Given:
- Plane graph G with maximum degree 4
- Orthogonal representation $H(G)$

Find: Compact orthogonal layout of G that realizes $H(G)$

Compaction

Compaction problem.

Given:
- Plane graph G with maximum degree 4
- Orthogonal representation $H(G)$

Find: Compact orthogonal layout of G that realizes $H(G)$

Special case.

All faces are rectangles.

Compaction

Compaction problem.

Given:
- Plane graph G with maximum degree 4
- Orthogonal representation $H(G)$

Find: Compact orthogonal layout of G that realizes $H(G)$

Special case.

All faces are rectangles.

This guarantees:

Compaction

Compaction problem.

Given:
- Plane graph G with maximum degree 4
- Orthogonal representation $H(G)$

Find: Compact orthogonal layout of G that realizes $H(G)$

Special case.

All faces are rectangles.

This guarantees:
- minimum total edge length

Compaction

Compaction problem.

Given:
- Plane graph G with maximum degree 4
- Orthogonal representation $H(G)$

Find: Compact orthogonal layout of G that realizes $H(G)$

Special case.

All faces are rectangles.

This guarantees:
- minimum total edge length
- minimum area

Compaction

Compaction problem.

Given:
- Plane graph G with maximum degree 4
- Orthogonal representation $H(G)$

Find: Compact orthogonal layout of G that realizes $H(G)$

Special case.

All faces are rectangles.

This guarantees:
- minimum total edge length
- minimum area

Properties.

Compaction

Compaction problem.

Given:
- Plane graph G with maximum degree 4
- Orthogonal representation $H(G)$

Find: Compact orthogonal layout of G that realizes $H(G)$

Special case.

All faces are rectangles.

This guarantees:
- minimum total edge length
- minimum area

Properties.

- bends only on the outer face

Compaction

Compaction problem.

Given:
- Plane graph G with maximum degree 4
- Orthogonal representation $H(G)$

Find: Compact orthogonal layout of G that realizes $H(G)$

Special case.

All faces are rectangles.

This guarantees:
- minimum total edge length
- minimum area

Properties.

- bends only on the outer face
- opposite sides of a face have the same length

Compaction

Compaction problem.

Given:
- Plane graph G with maximum degree 4
- Orthogonal representation $H(G)$

Find: Compact orthogonal layout of G that realizes $H(G)$

Special case.

All faces are rectangles.

This guarantees:
- minimum total edge length
- minimum area

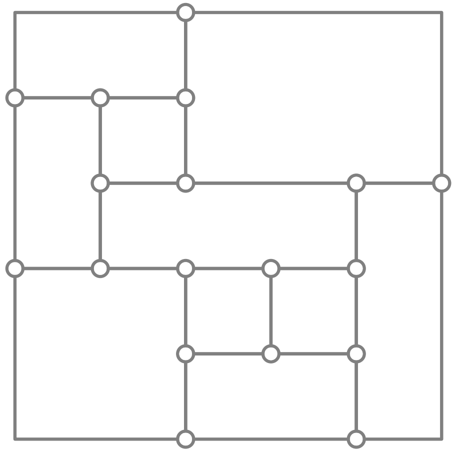
Properties.

- bends only on the outer face
- opposite sides of a face have the same length

Idea.

- Formulate flow network for horizontal/vertical compaction

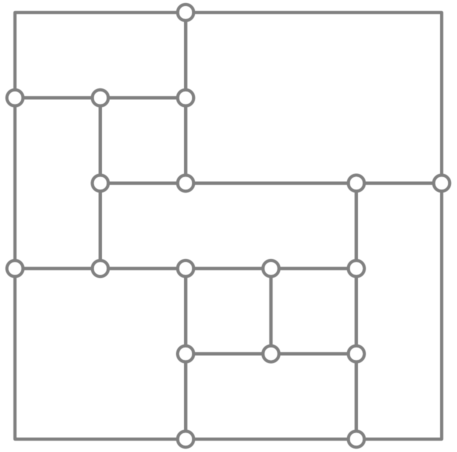
Flow Network for Edge-Length Assignment



Flow Network for Edge-Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

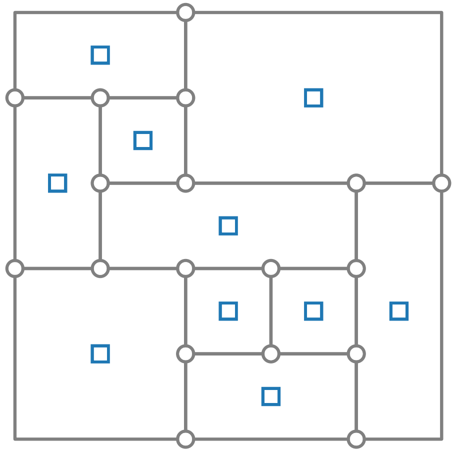


Flow Network for Edge-Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

- $W_{\text{hor}} = F \setminus \{f_0\}$ □

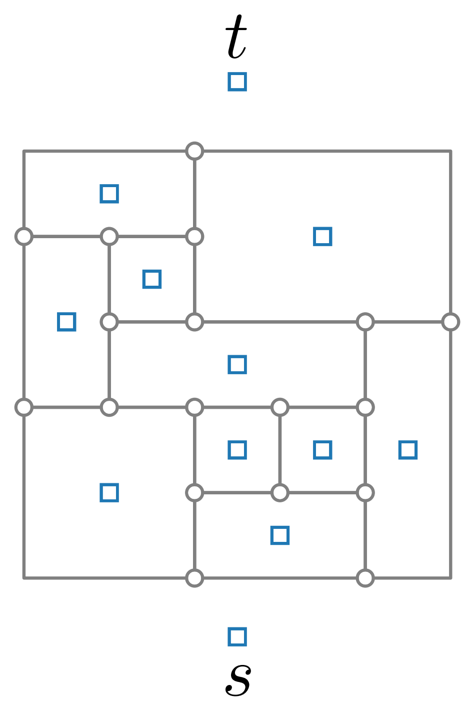


Flow Network for Edge-Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

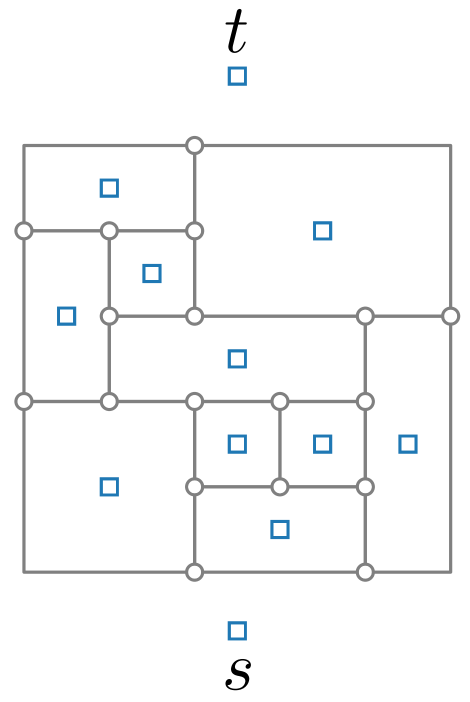
- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$ □



Flow Network for Edge-Length Assignment

Definition.
Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$

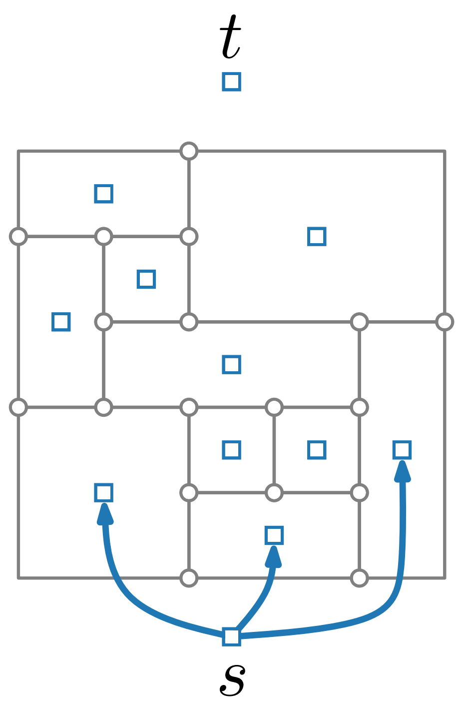


Flow Network for Edge-Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$

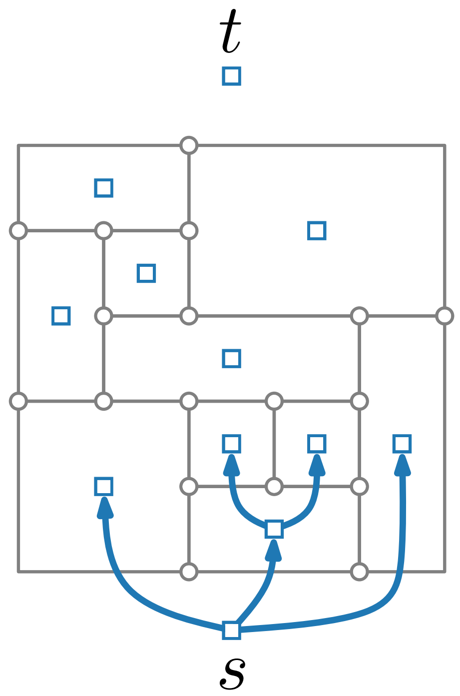


Flow Network for Edge-Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

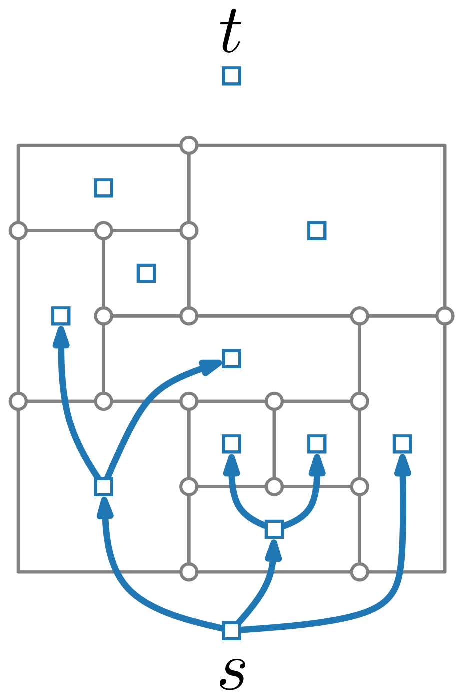
- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$



Flow Network for Edge-Length Assignment

Definition.
Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$ □
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$

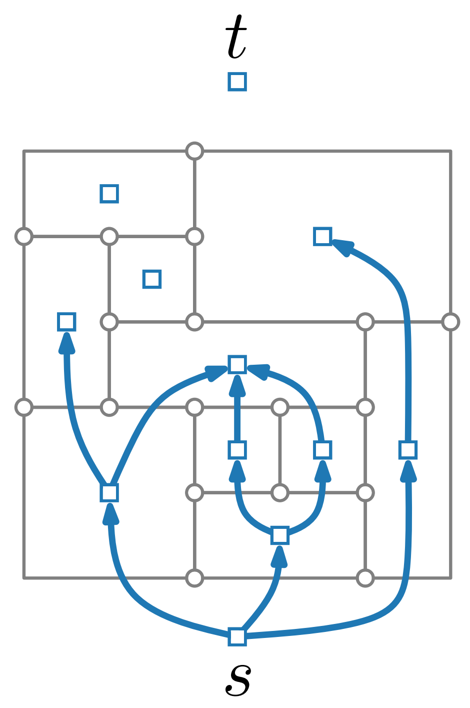


Flow Network for Edge-Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$

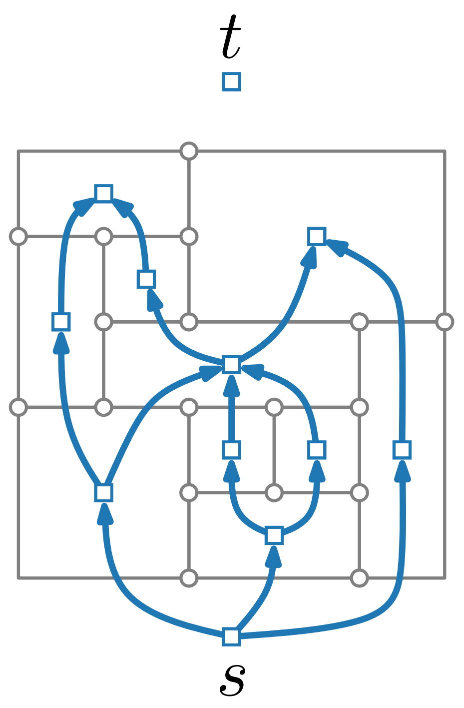


Flow Network for Edge-Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$ ◻
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$

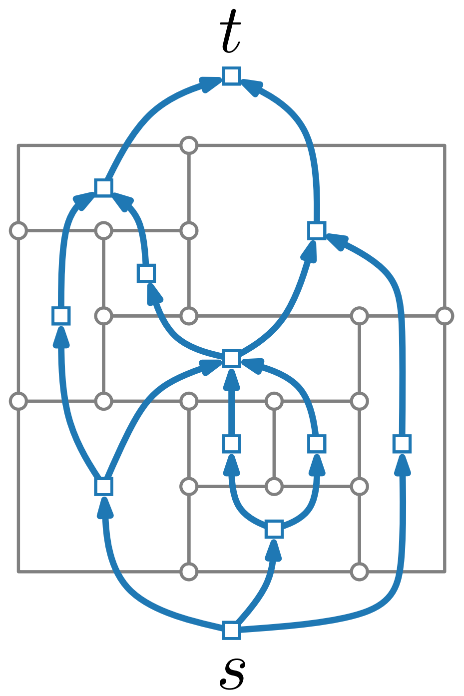


Flow Network for Edge-Length Assignment

Definition.

Flow Network $N_{\mathrm{hor}} = ((W_{\mathrm{hor}}, E_{\mathrm{hor}}); b; \ell; u; \mathrm{cost})$

- $W_{\mathrm{hor}} = F \setminus \{f_0\} \cup \{s, t\}$ □
- $E_{\mathrm{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$

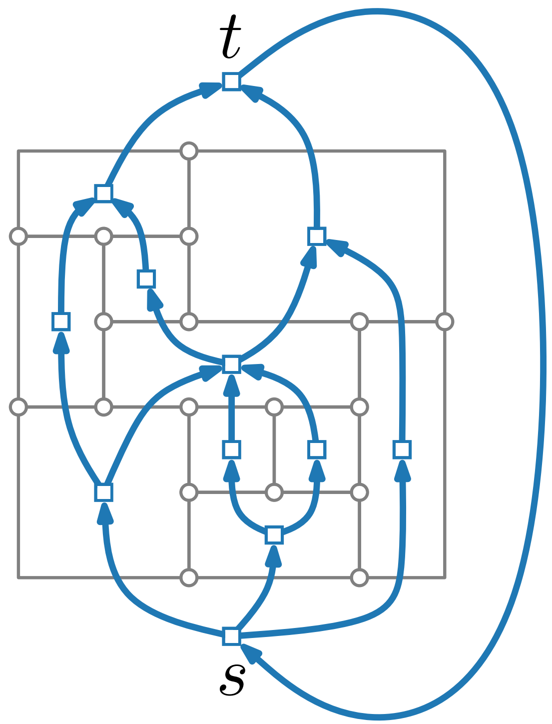


Flow Network for Edge-Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$ ▫
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\} \cup \{(t, s)\}$

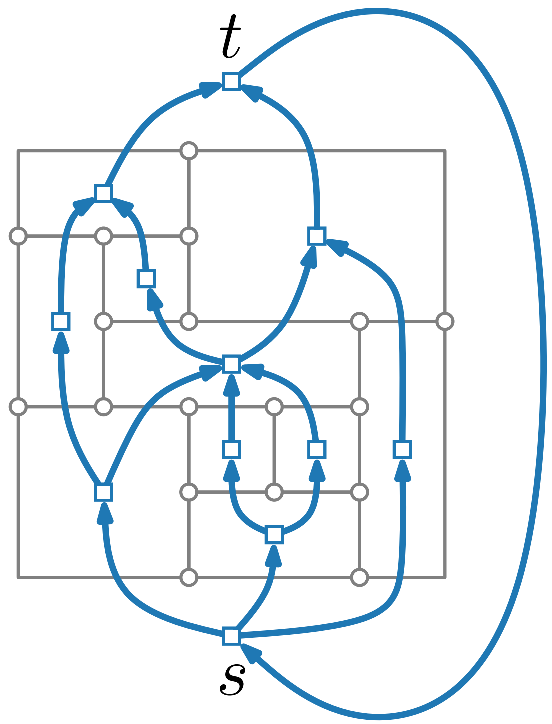


Flow Network for Edge-Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$ □
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\} \cup \{(t, s)\}$
- $\ell(a) = 1 \quad \forall a \in E_{\text{hor}}$

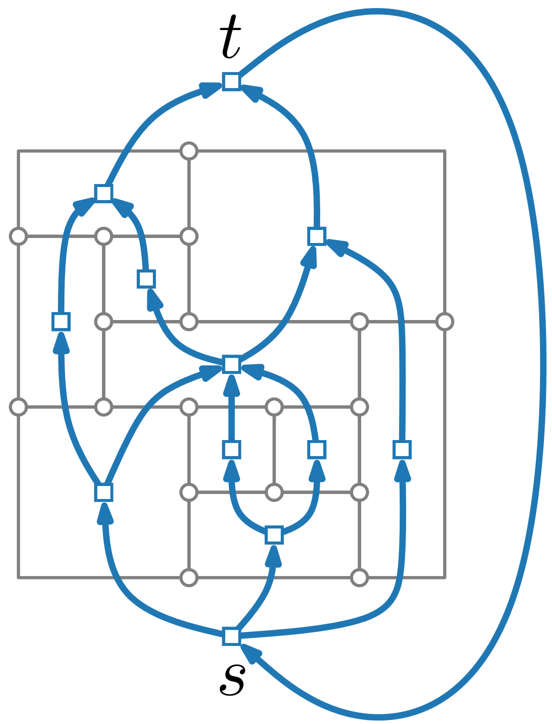


Flow Network for Edge-Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\} \cup \{(t, s)\}$
- $\ell(a) = 1 \quad \forall a \in E_{\text{hor}}$
- $u(a) = \infty \quad \forall a \in E_{\text{hor}}$

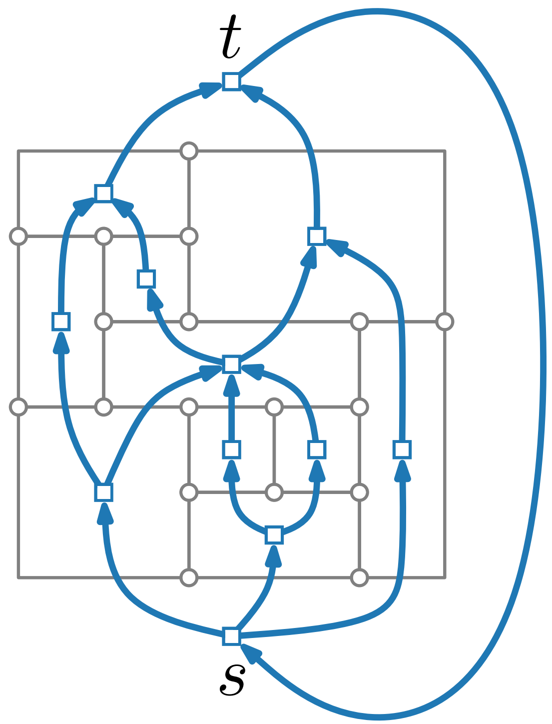


Flow Network for Edge-Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$ ◻
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\} \cup \{(t, s)\}$
- $\ell(a) = 1 \quad \forall a \in E_{\text{hor}}$
- $u(a) = \infty \quad \forall a \in E_{\text{hor}}$
- $\text{cost}(a) = 1 \quad \forall a \in E_{\text{hor}}$

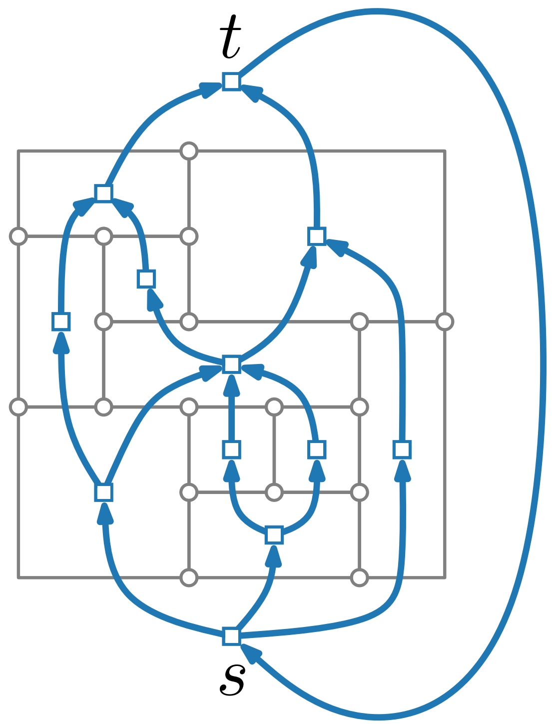


Flow Network for Edge-Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

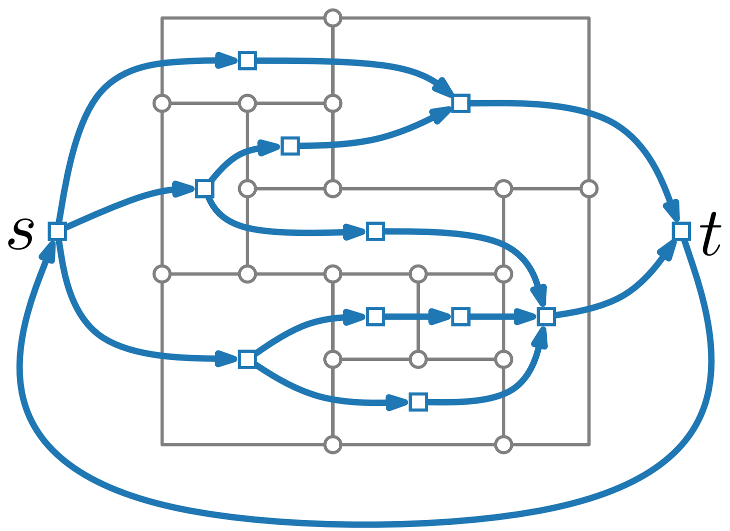
- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\} \cup \{(t, s)\}$
- $\ell(a) = 1 \quad \forall a \in E_{\text{hor}}$
- $u(a) = \infty \quad \forall a \in E_{\text{hor}}$
- $\text{cost}(a) = 1 \quad \forall a \in E_{\text{hor}}$
- $b(f) = 0 \quad \forall f \in W_{\text{hor}}$



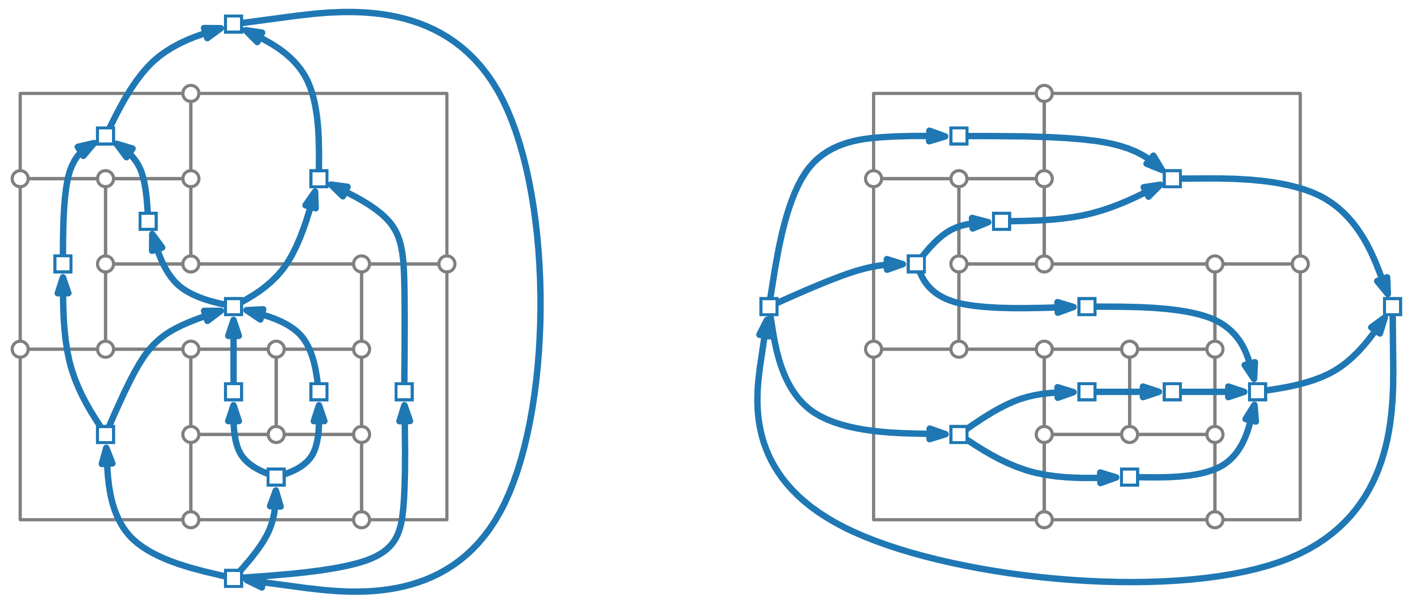
Flow Network for Edge-Length Assignment

Definition.
Flow Network $N_{\text{ver}} = ((W_{\text{ver}}, E_{\text{ver}}); b; \ell; u; \text{cost})$

- $W_{\text{ver}} = F \setminus \{f_0\} \cup \{s, t\}$ □
- $E_{\text{ver}} = \{(f, g) \mid f, g$ share a *vertical* segment and f lies to the *left* of $g\} \cup \{(t, s)\}$
- $\ell(a) = 1 \quad \forall a \in E_{\text{ver}}$
- $u(a) = \infty \quad \forall a \in E_{\text{ver}}$
- $\text{cost}(a) = 1 \quad \forall a \in E_{\text{ver}}$
- $b(f) = 0 \quad \forall f \in W_{\text{ver}}$

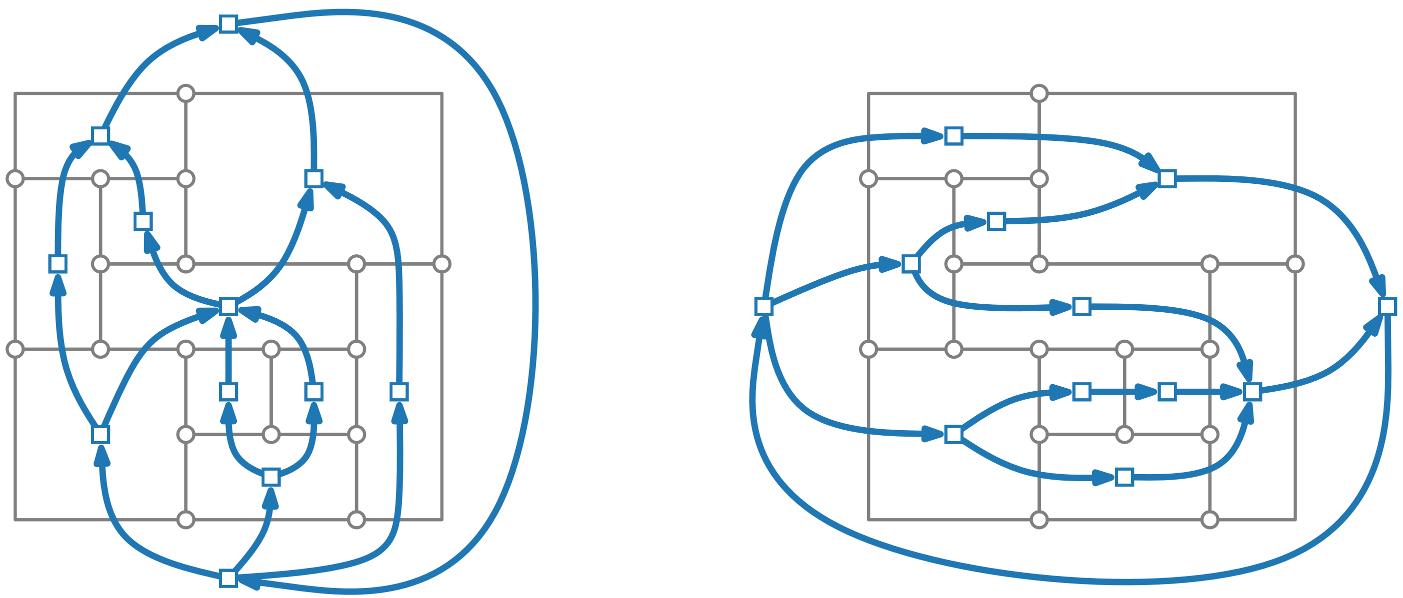


Compaction – Result



Theorem.
A valid flow for N_{hor} and N_{ver} exists $\Leftrightarrow$ corresponding edge lengths induce an orthogonal drawing.

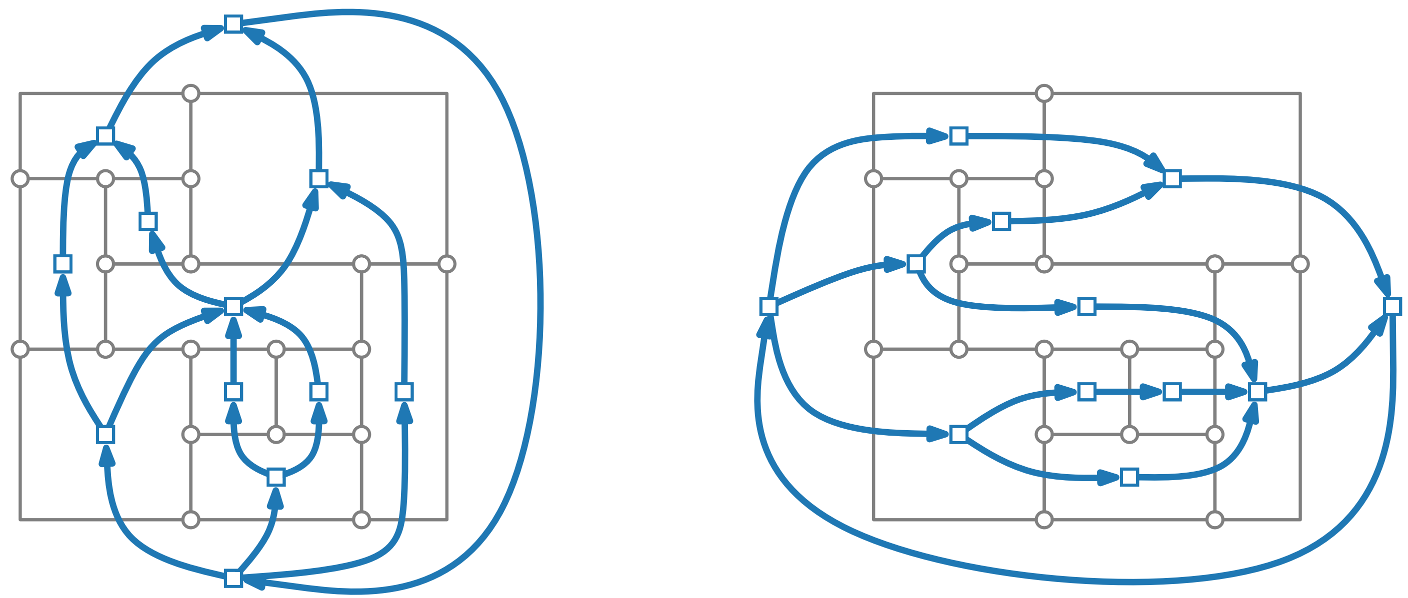
Compaction – Result



Theorem.
A valid flow for N_{hor} and N_{ver} exists $\Leftrightarrow$
corresponding edge lengths induce an orthogonal drawing.

What values of the drawing do the following quantities represent?

Compaction – Result

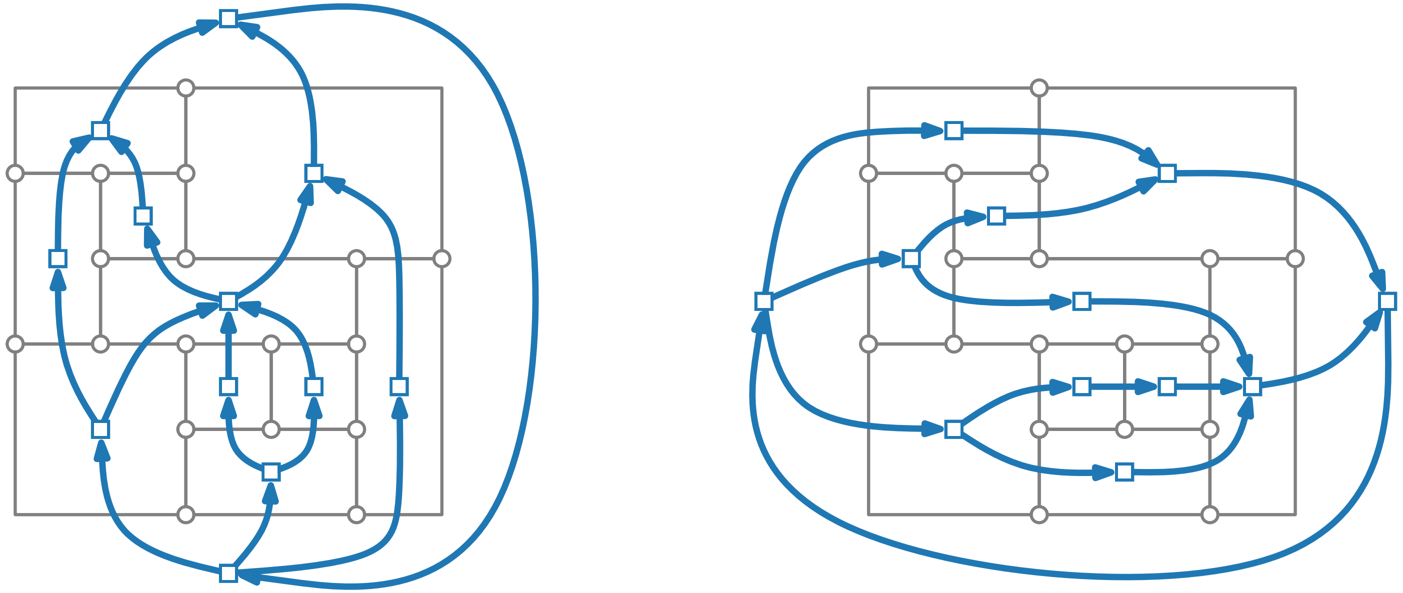


Theorem.
A valid flow for N_{hor} and N_{ver} exists $\Leftrightarrow$
corresponding edge lengths induce an orthogonal drawing.

What values of the drawing do the following quantities represent?

- $|X_{\text{hor}}(t, s)|$ and $|X_{\text{ver}}(t, s)|$?

Compaction – Result

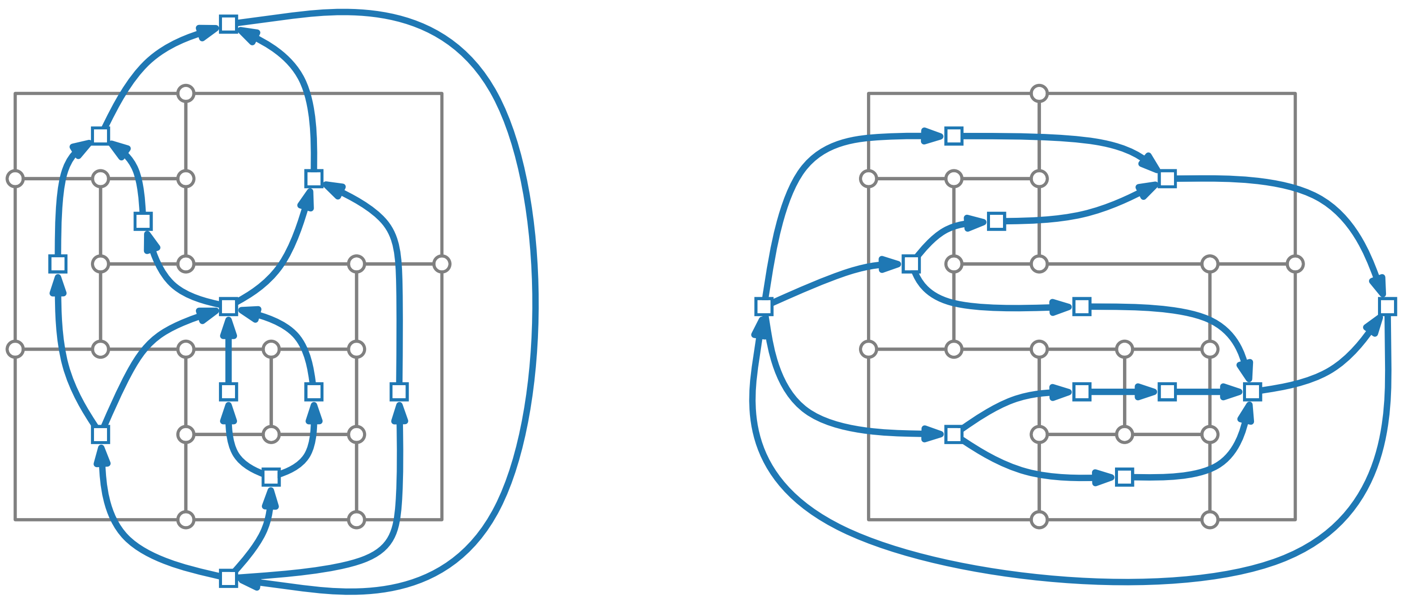


Theorem.
A valid flow for N_{hor} and N_{ver} exists $\Leftrightarrow$
corresponding edge lengths induce an orthogonal drawing.

What values of the drawing do the following quantities represent?

- $|X_{\text{hor}}(t,s)|$ and $|X_{\text{ver}}(t,s)|$? width and height of the drawing

Compaction – Result

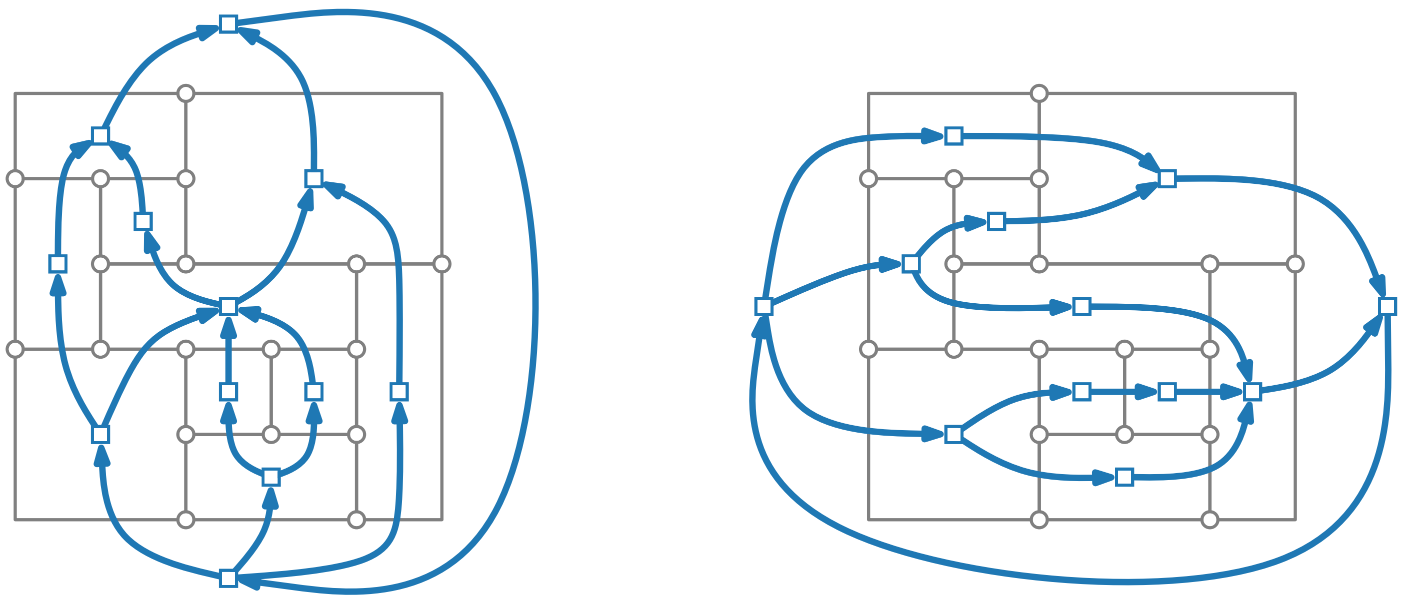


Theorem.
A valid flow for N_{hor} and N_{ver} exists $\Leftrightarrow$
corresponding edge lengths induce an orthogonal drawing.

What values of the drawing do the following quantities represent?

- $|X_{\text{hor}}(t,s)|$ and $|X_{\text{ver}}(t,s)|$? width and height of the drawing
- $\sum_{e \in E_{\text{hor}}} X_{\text{hor}}(e) + \sum_{e \in E_{\text{ver}}} X_{\text{ver}}(e)$

Compaction – Result

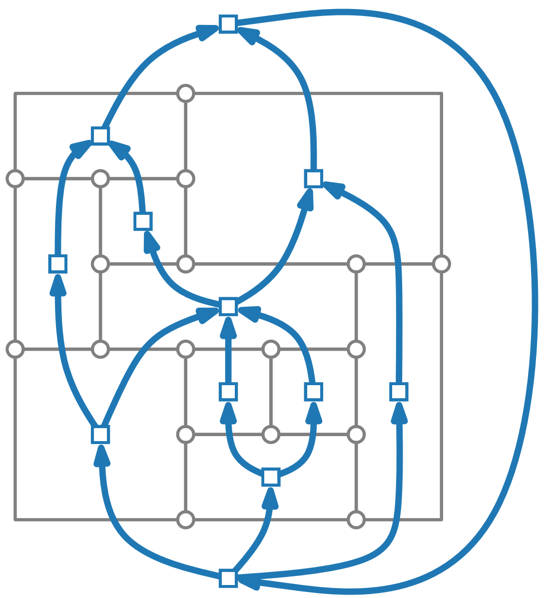


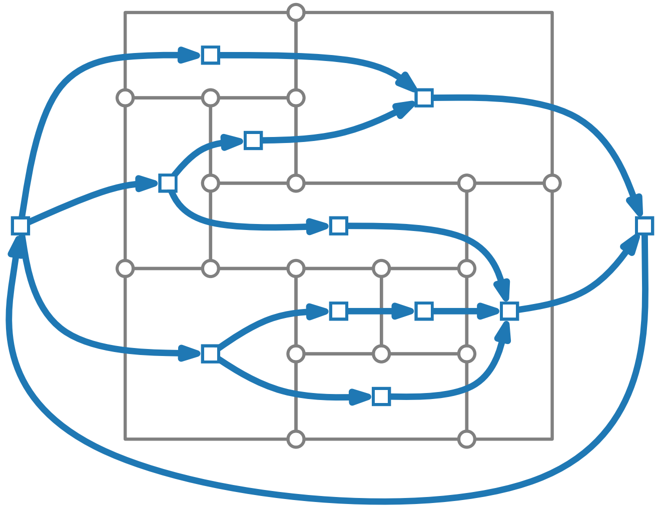
Theorem.
A valid flow for N_{hor} and N_{ver} exists $\Leftrightarrow$
corresponding edge lengths induce an orthogonal drawing.

What values of the drawing do the following quantities represent?

- $|X_{\text{hor}}(t,s)|$ and $|X_{\text{ver}}(t,s)|$? width and height of the drawing
- $\sum_{e \in E_{\text{hor}}} X_{\text{hor}}(e) + \sum_{e \in E_{\text{ver}}} X_{\text{ver}}(e)$ total edge length

Compaction – Result





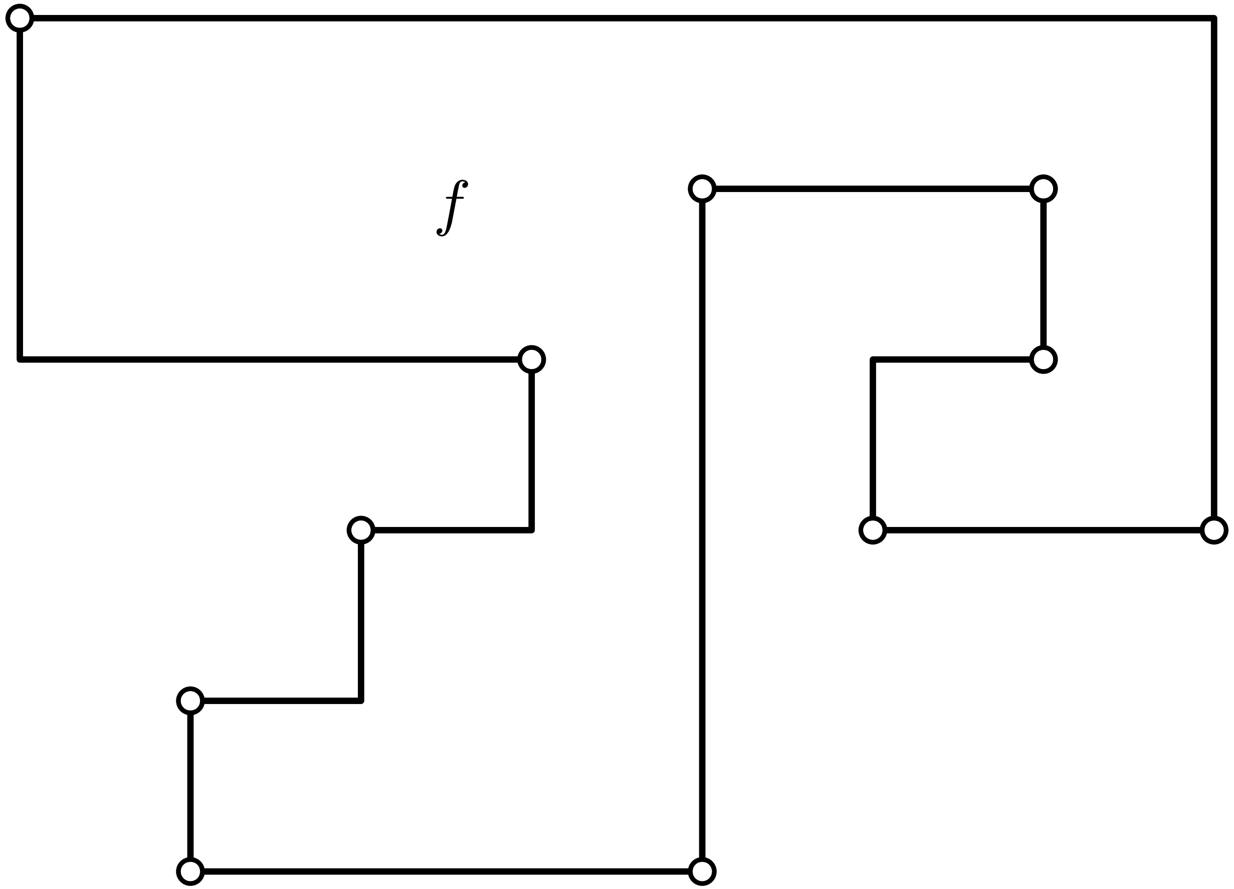
What if not all faces are rectangular?

Theorem.
A valid flow for N_{hor} and N_{ver} exists $\Leftrightarrow$ corresponding edge lengths induce an orthogonal drawing.

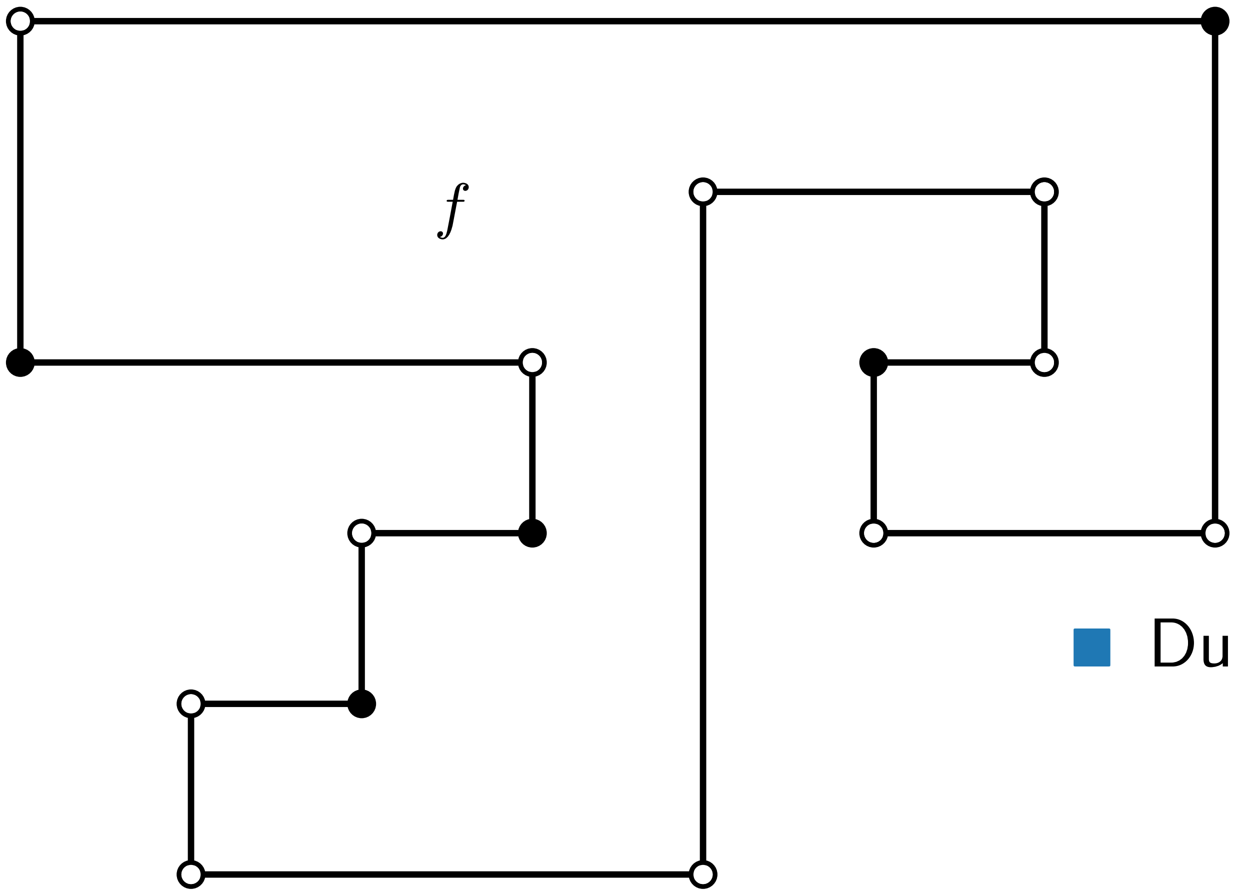
What values of the drawing do the following quantities represent?

- $|X_{\text{hor}}(t,s)|$ and $|X_{\text{ver}}(t,s)|$? width and height of the drawing
- $\sum_{e \in E_{\text{hor}}} X_{\text{hor}}(e) + \sum_{e \in E_{\text{ver}}} X_{\text{ver}}(e)$ total edge length

Refinement of G and $H(G)$ – Inner Face

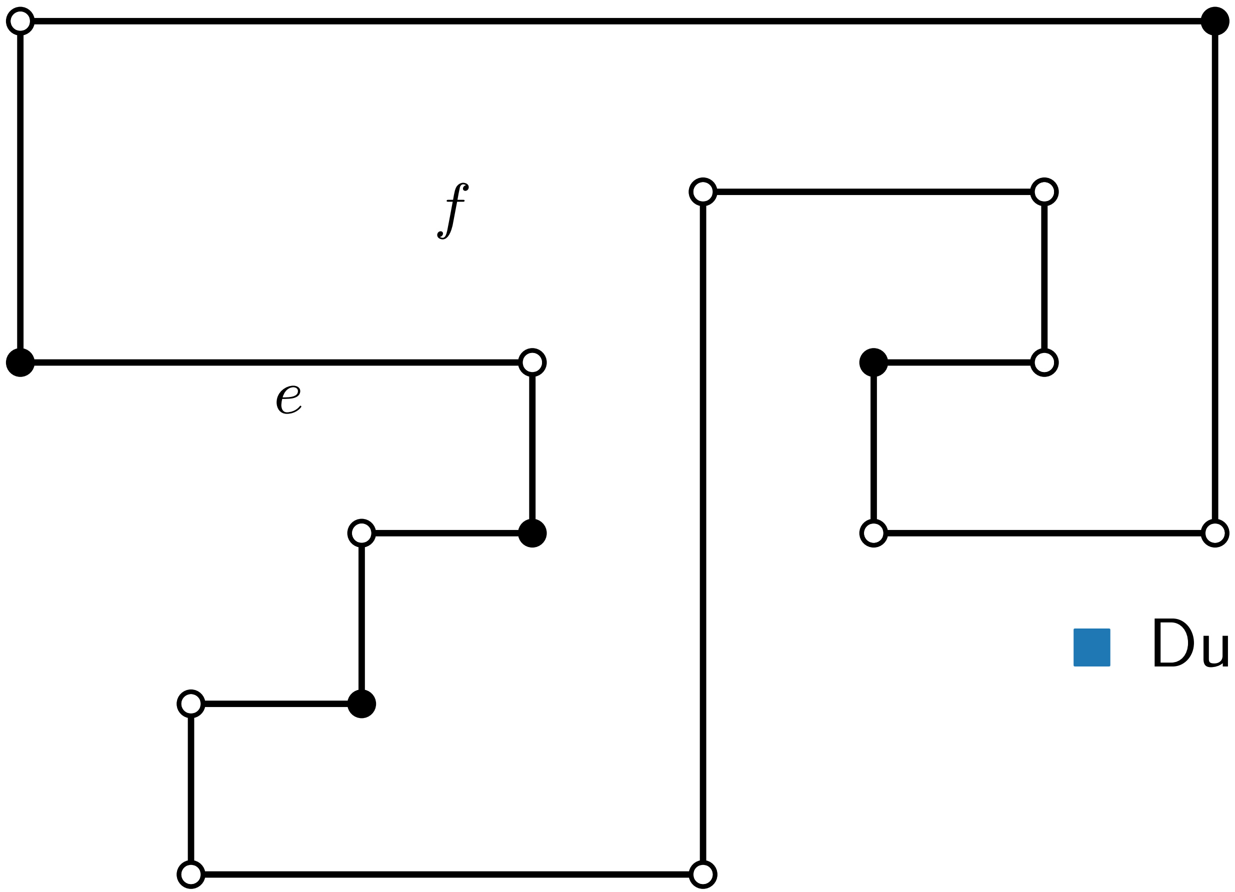


Refinement of G and $H(G)$ – Inner Face



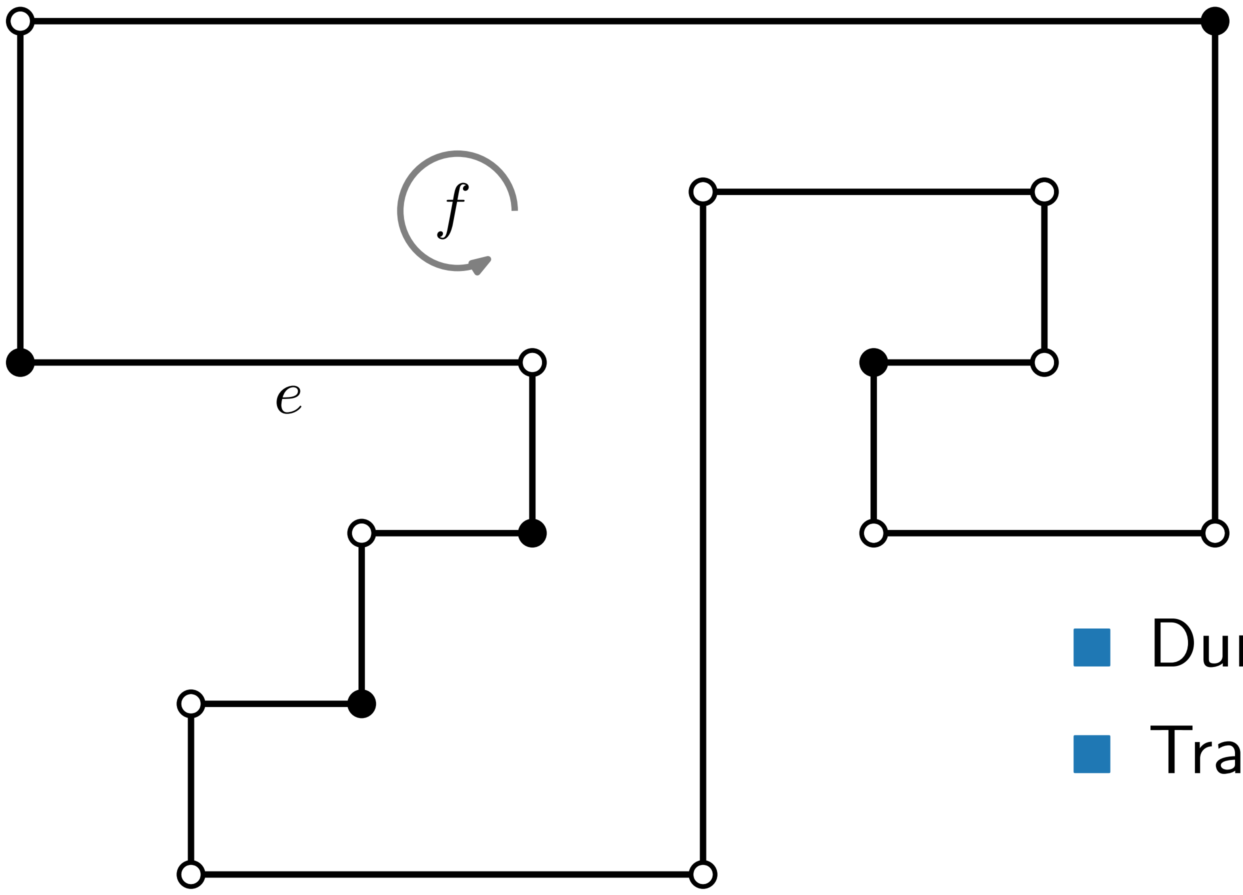
- Dummy vertices for bends

Refinement of G and $H(G)$ – Inner Face



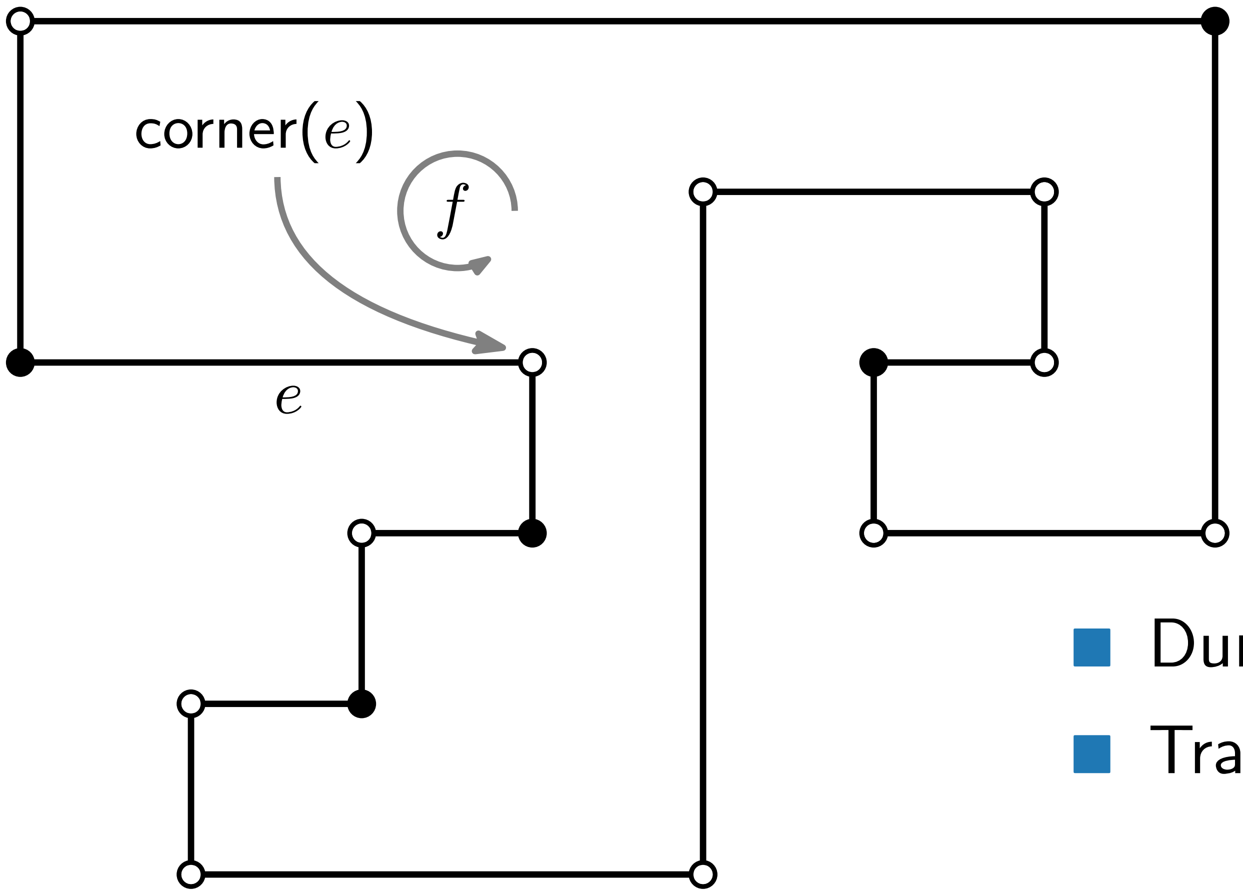
- Dummy vertices for bends

Refinement of G and $H(G)$ – Inner Face



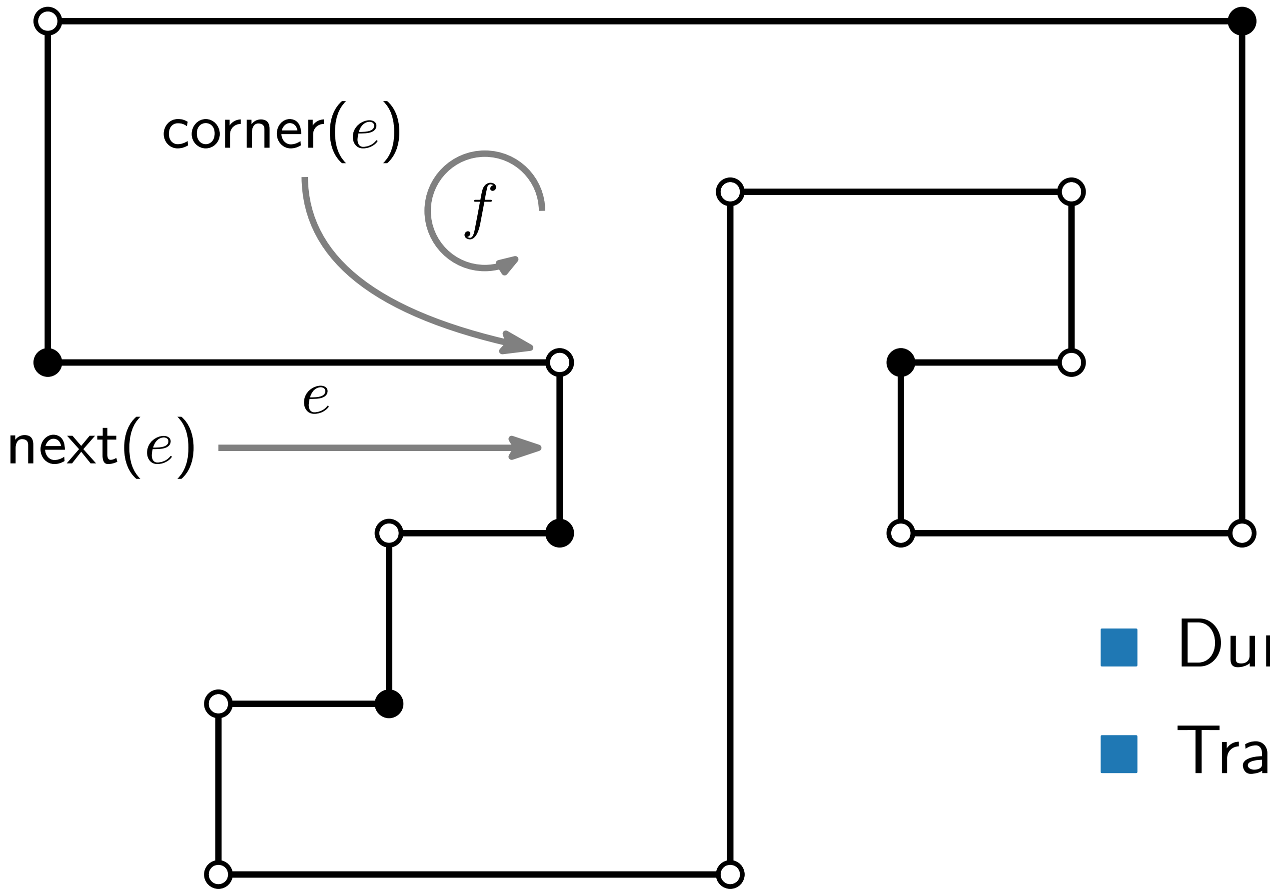
- Dummy vertices for bends
- Traverse counter-clockwise

Refinement of G and $H(G)$ – Inner Face



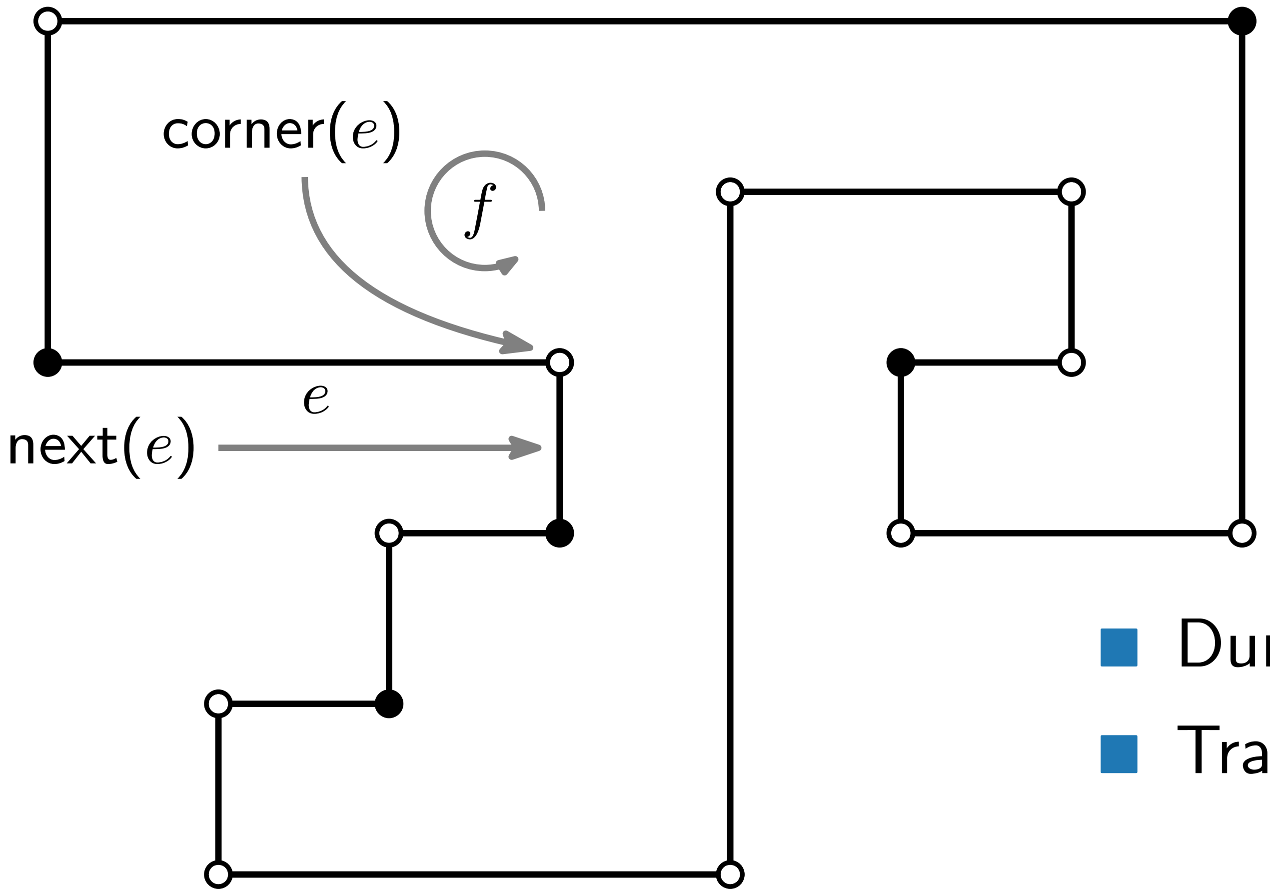
- Dummy vertices for bends
- Traverse counter-clockwise

Refinement of G and $H(G)$ – Inner Face



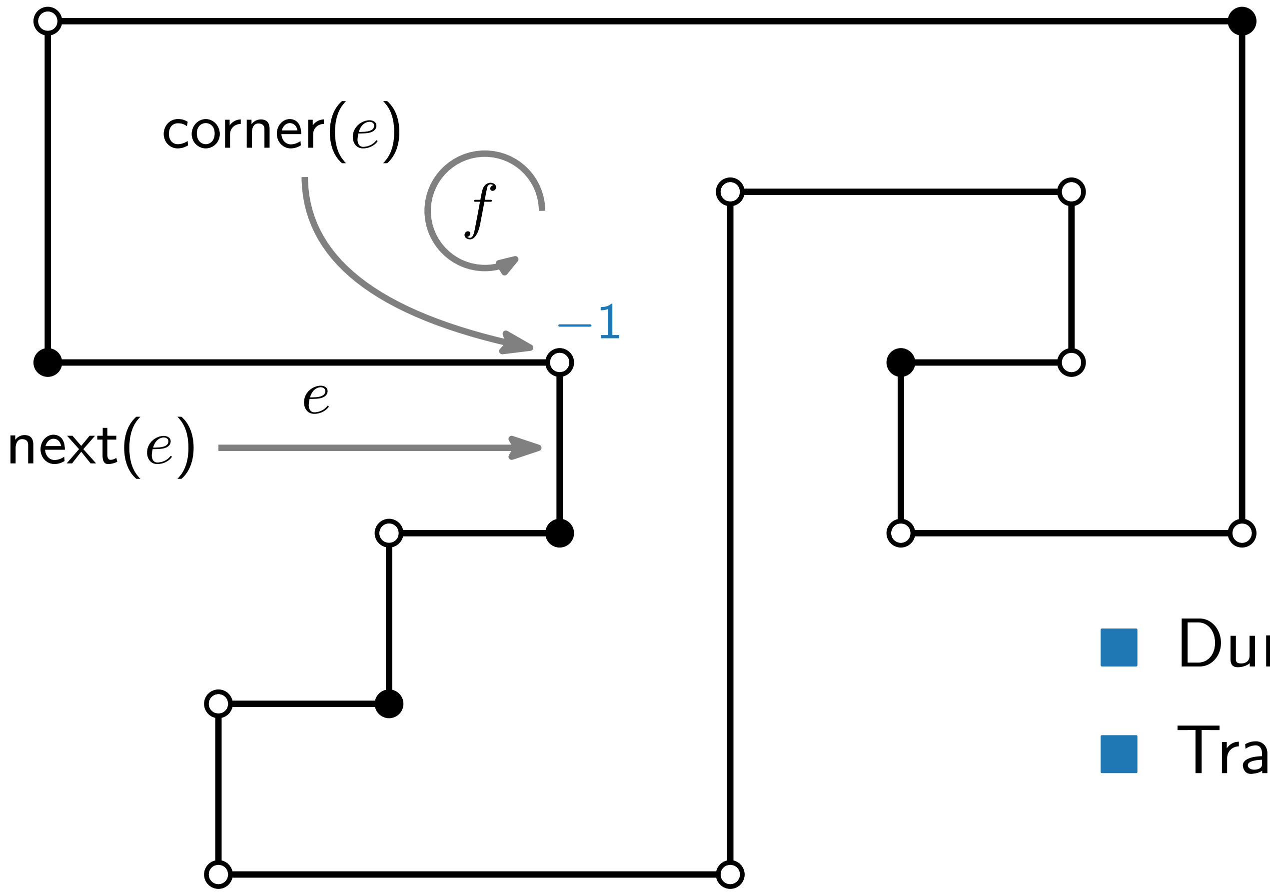
- Dummy vertices for bends
- Traverse counter-clockwise

Refinement of G and $H(G)$ – Inner Face



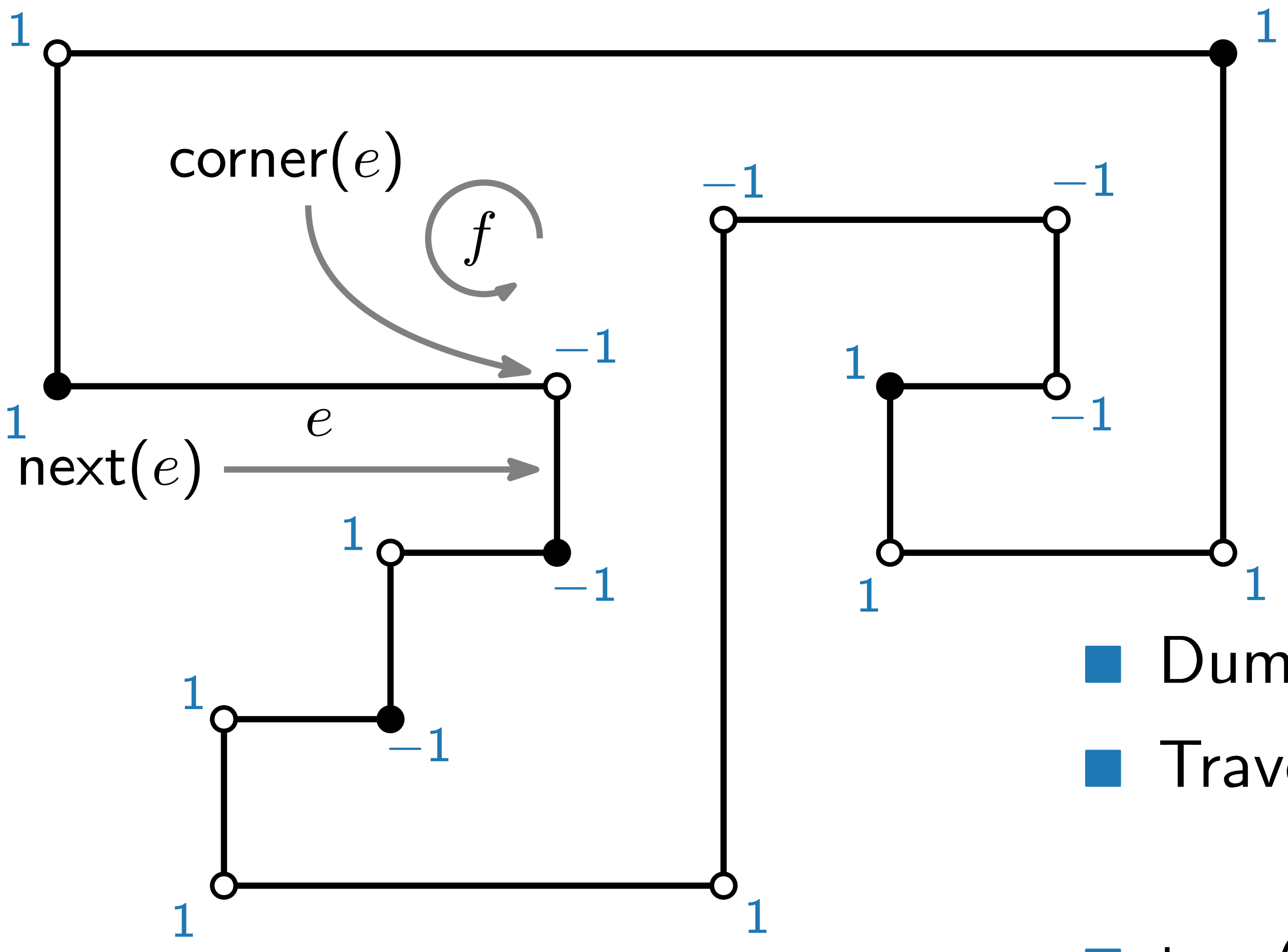
- Dummy vertices for bends
- Traverse counter-clockwise
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of G and $H(G)$ – Inner Face



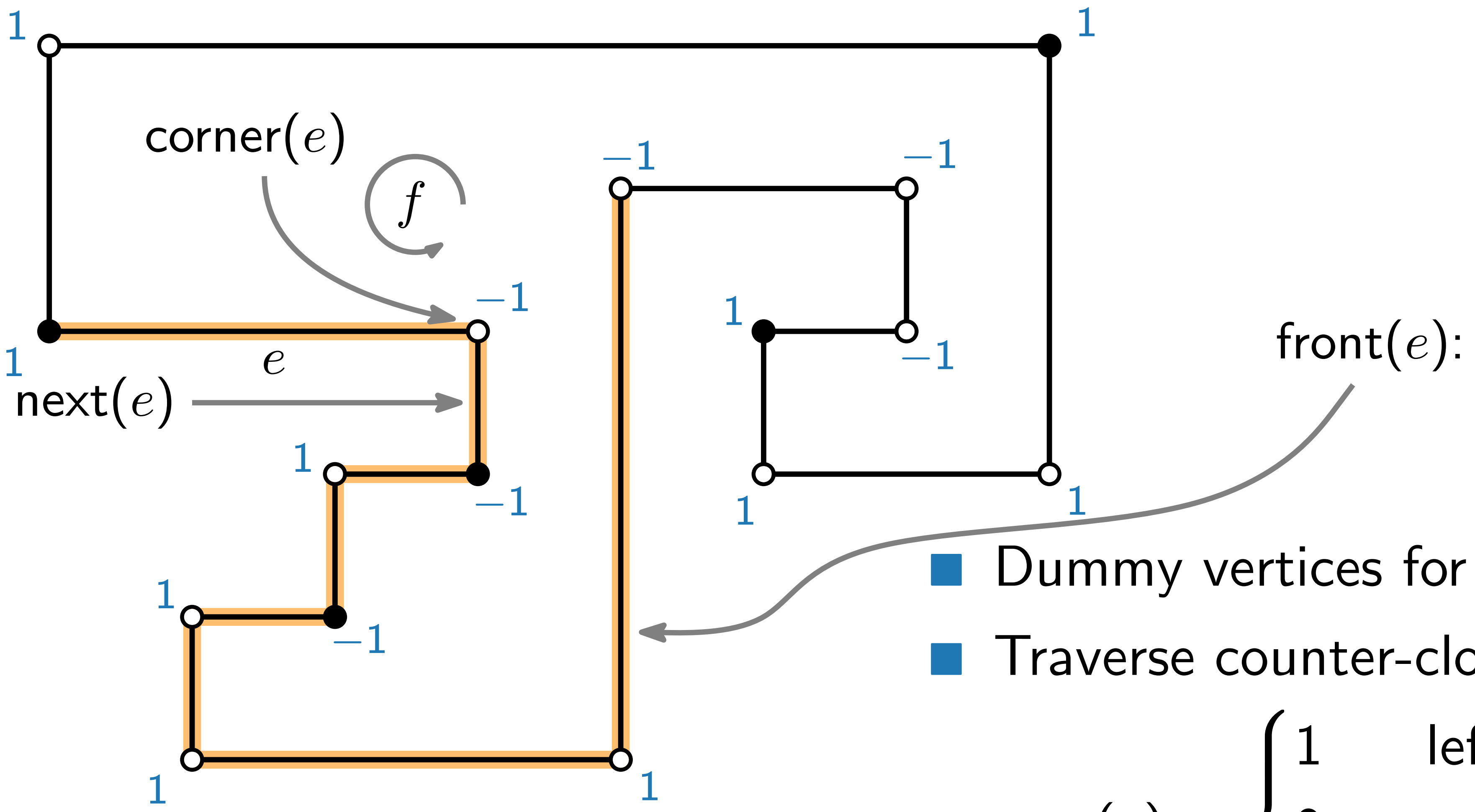
- Dummy vertices for bends
- Traverse counter-clockwise
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of G and $H(G)$ – Inner Face



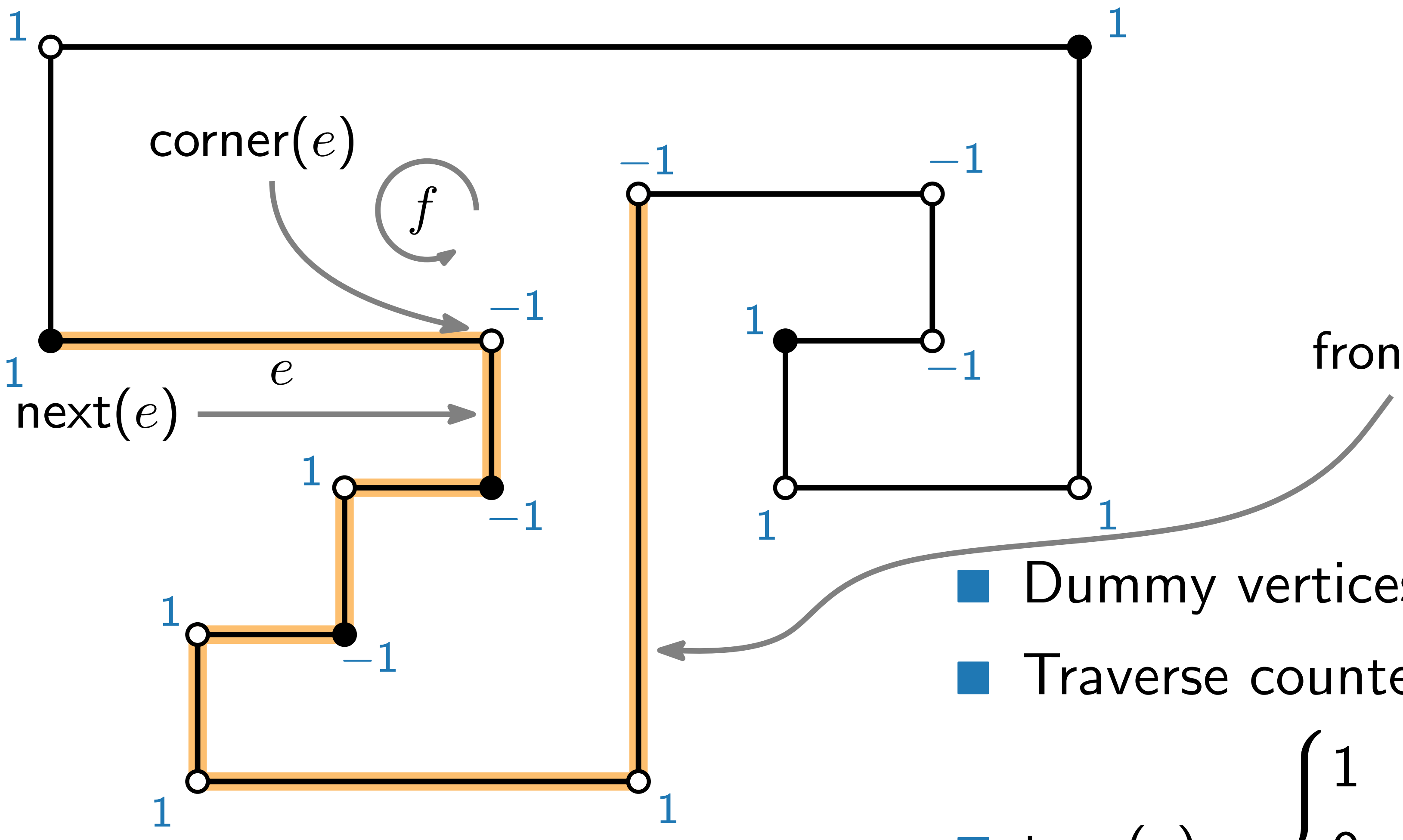
- Dummy vertices for bends
- Traverse counter-clockwise
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of G and $H(G)$ – Inner Face



- Dummy vertices for bends
- Traverse counter-clockwise
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

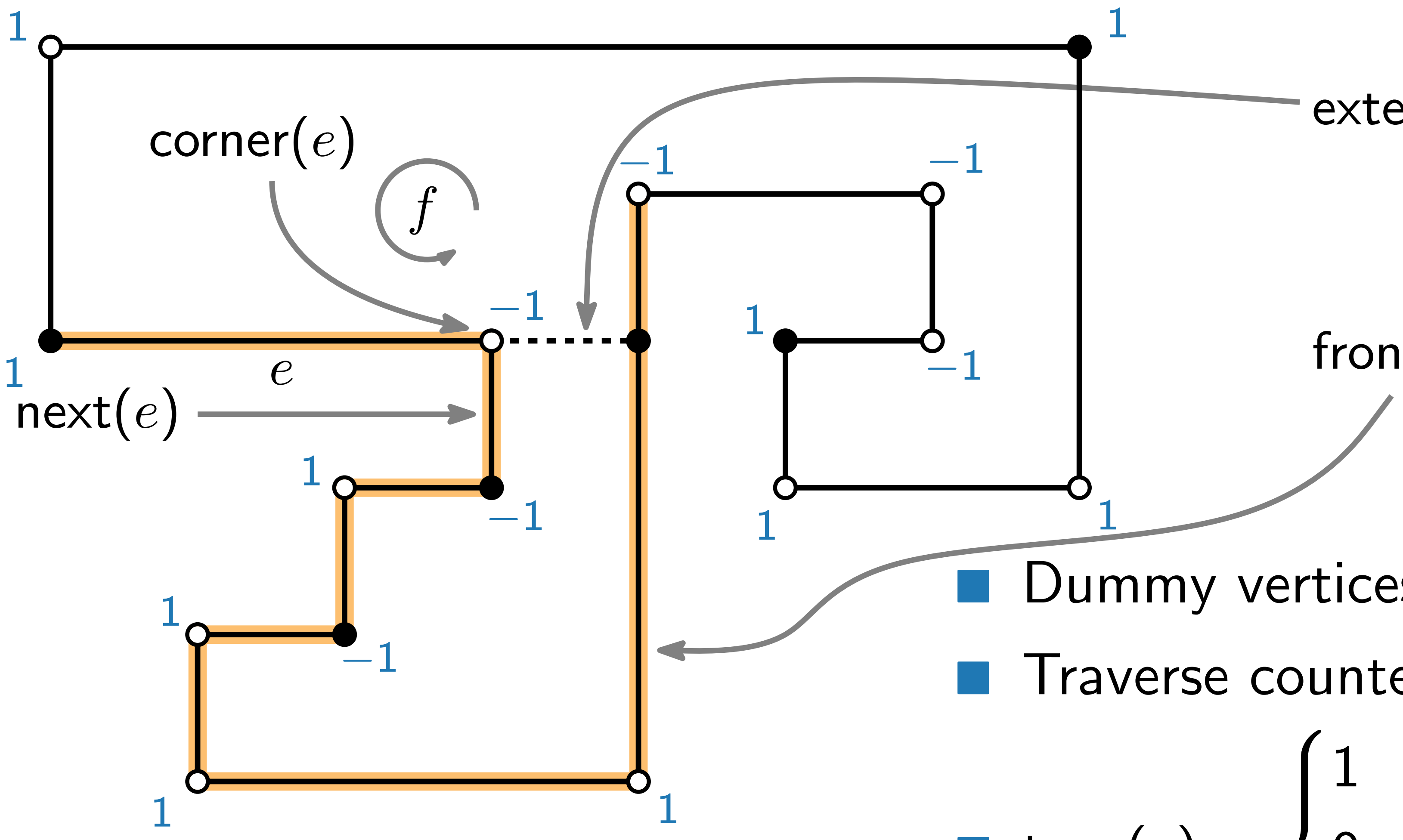
Refinement of G and $H(G)$ – Inner Face



front(e): First edge e after e such that the angle sum between e and e' is positive.

- Dummy vertices for bends
- Traverse counter-clockwise
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

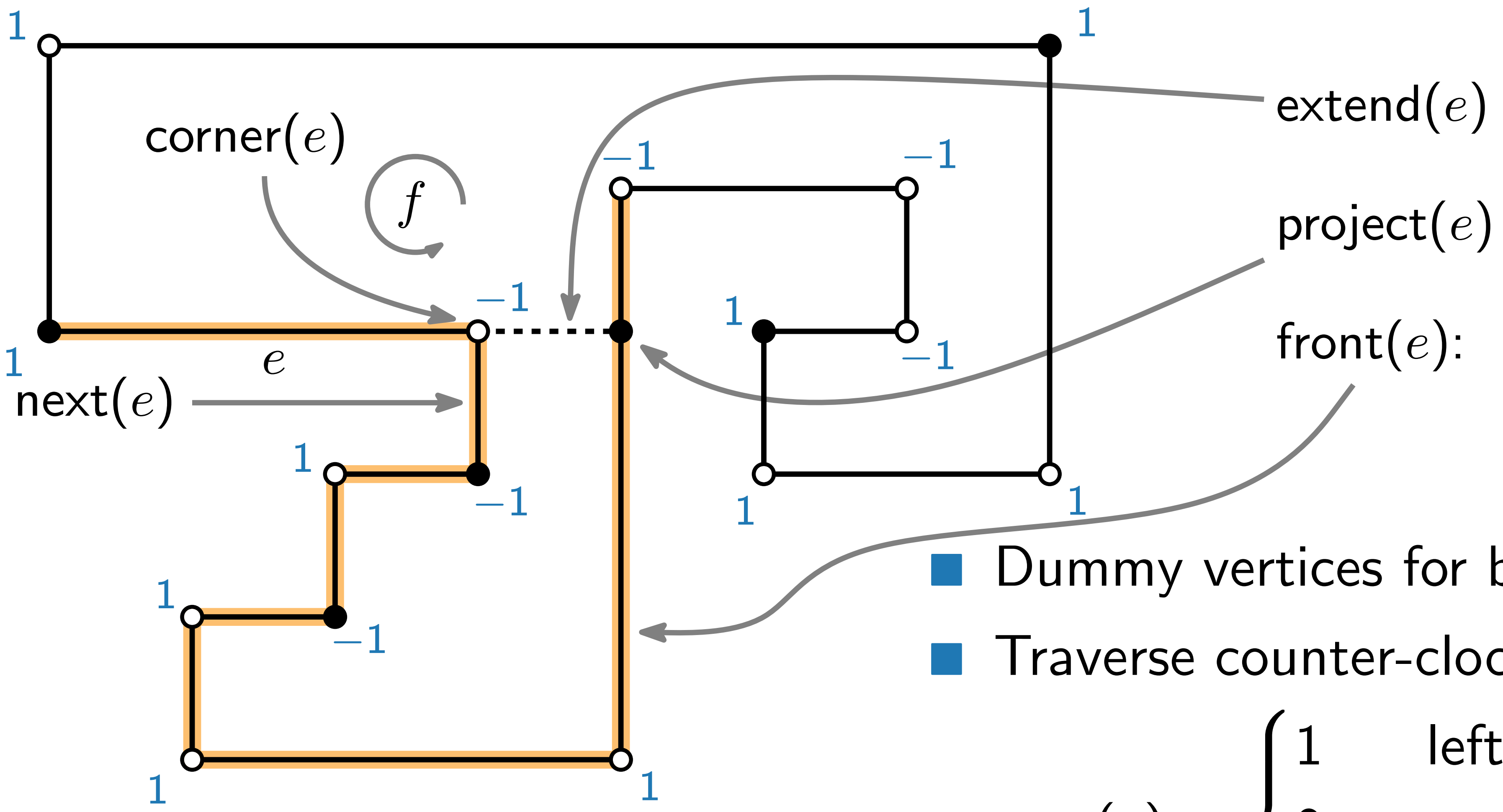
Refinement of G and $H(G)$ – Inner Face



front(e): First edge e after e such that the angle sum between e and e' is positive.

- Dummy vertices for bends
- Traverse counter-clockwise
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

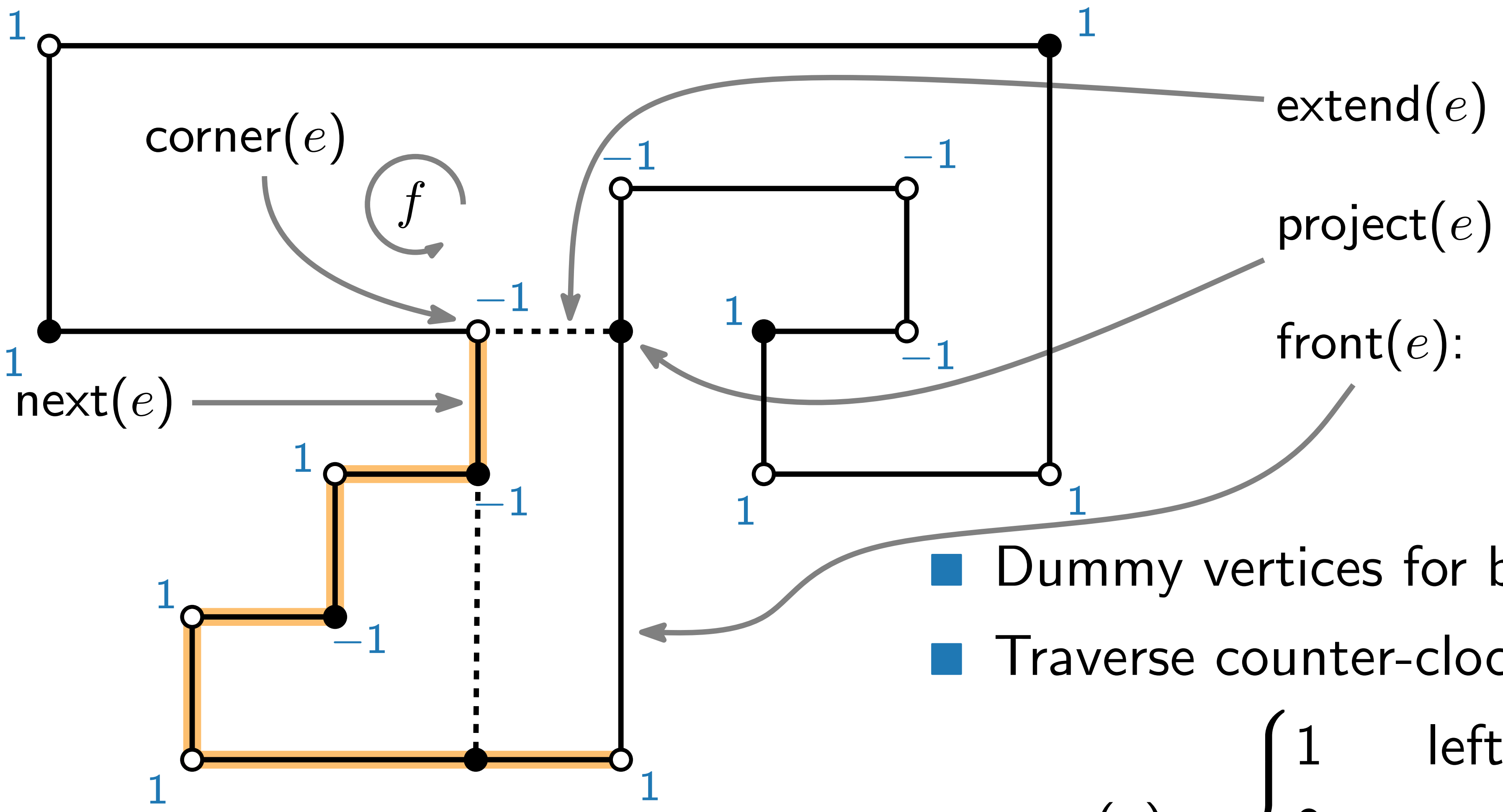
Refinement of G and $H(G)$ – Inner Face



front(e): First edge e after e such that the angle sum between e and e' is positive.

- Dummy vertices for bends
- Traverse counter-clockwise
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

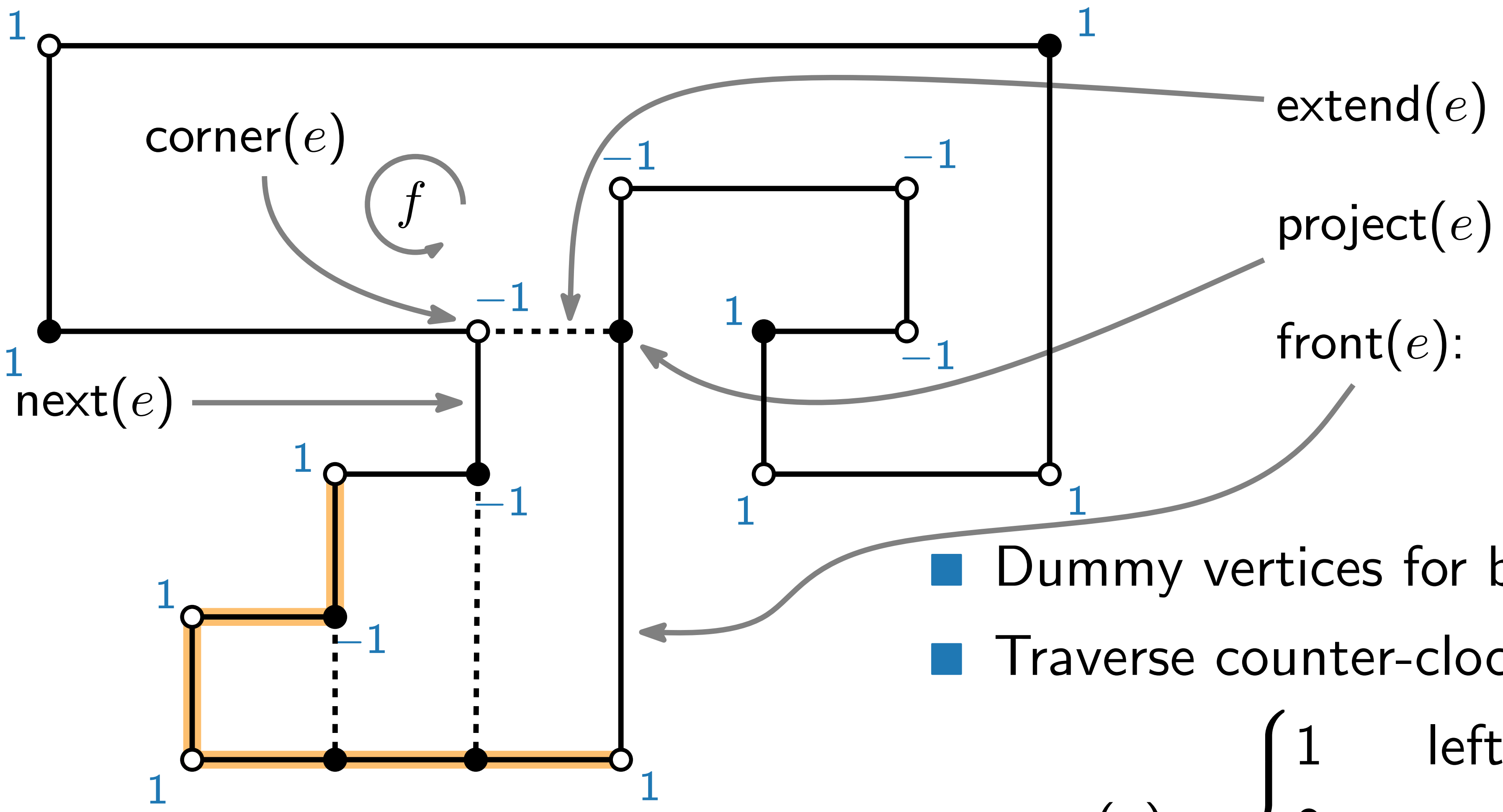
Refinement of G and $H(G)$ – Inner Face



First edge e after e such that the angle sum between e and e' is positive.

- Dummy vertices for bends
- Traverse counter-clockwise
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

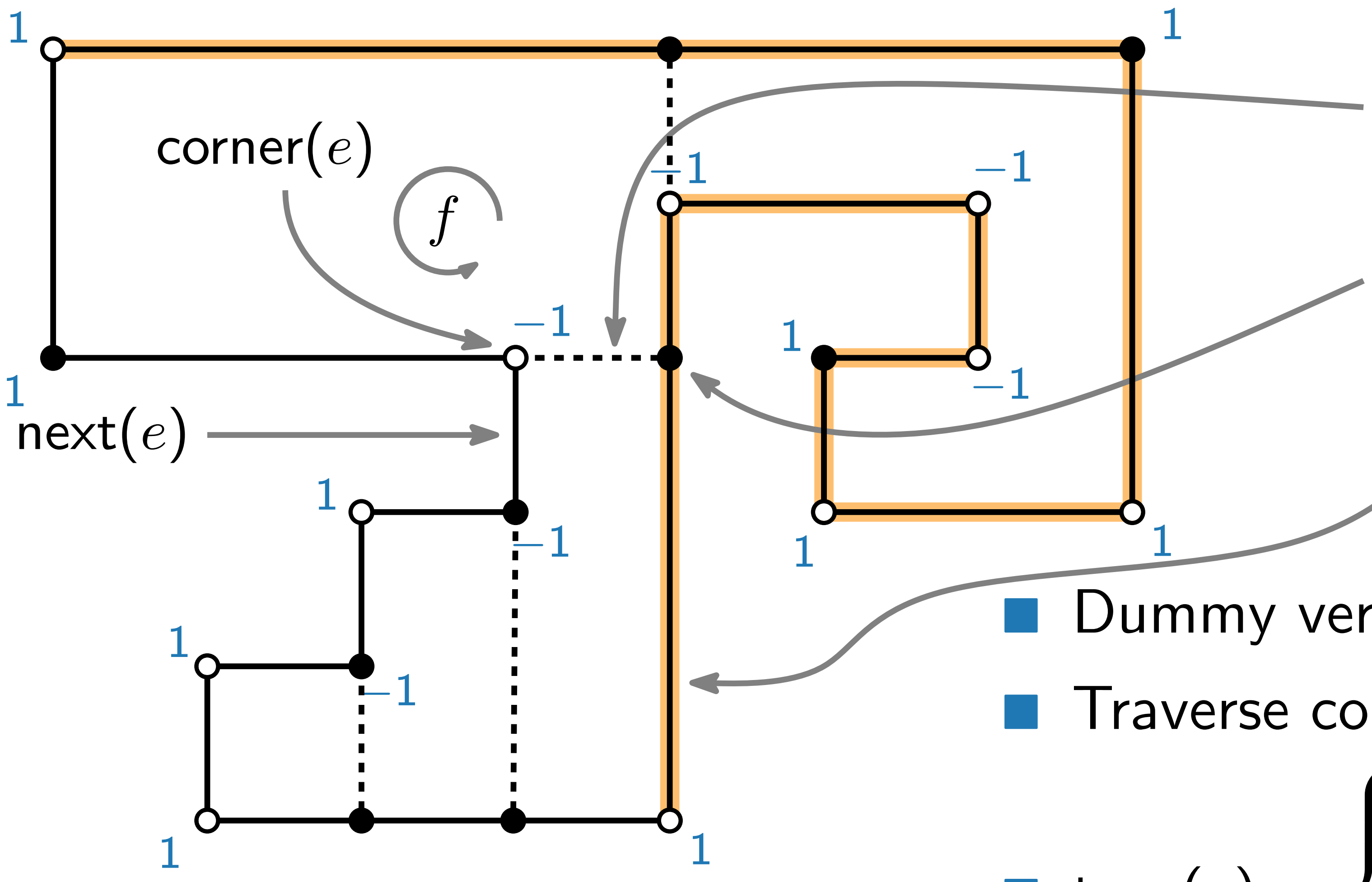
Refinement of G and $H(G)$ – Inner Face



front(e): First edge e after e such that the angle sum between e and e' is positive.

- Dummy vertices for bends
- Traverse counter-clockwise
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

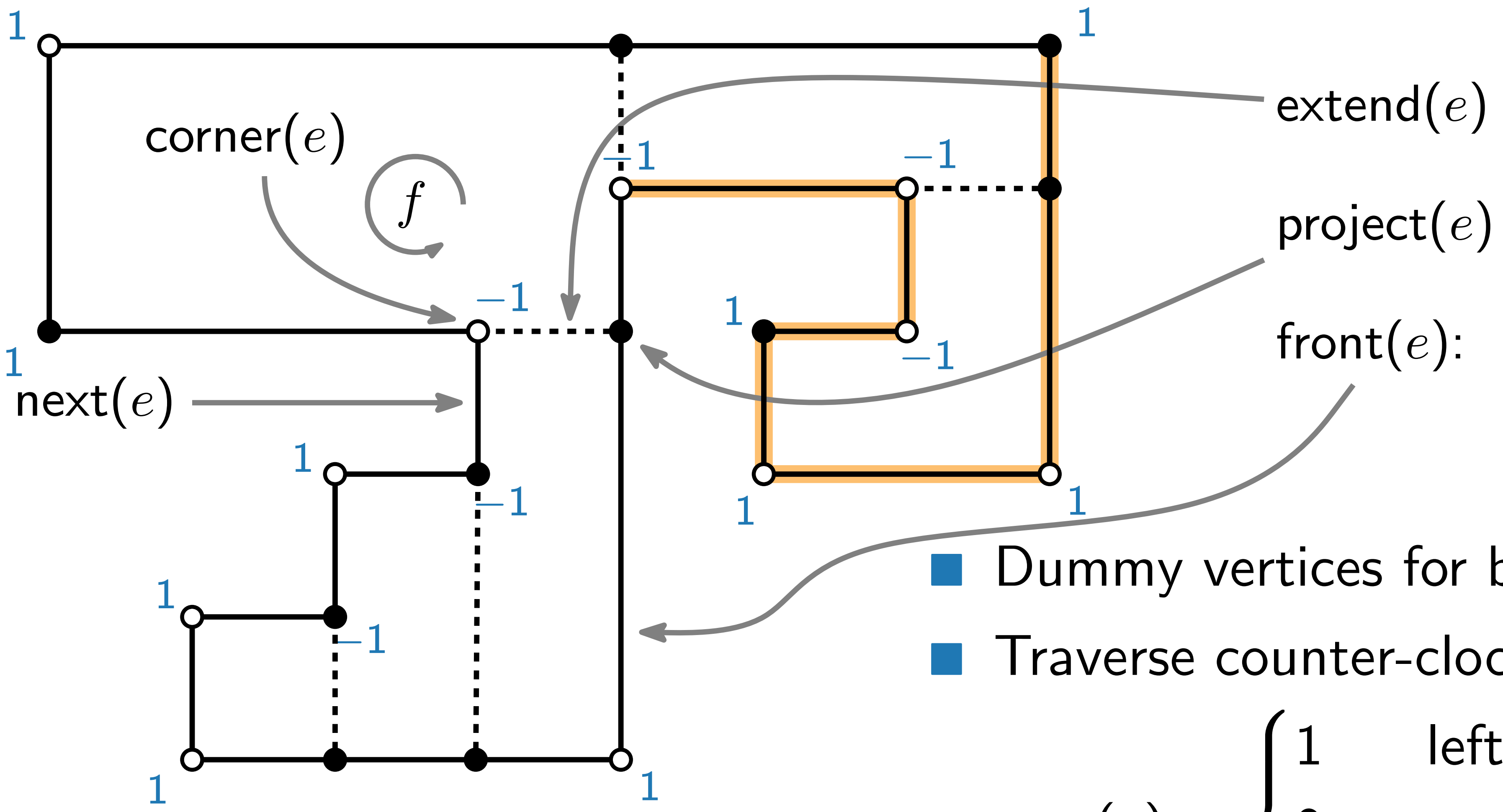
Refinement of G and $H(G)$ – Inner Face



front(e): First edge e after e such that the angle sum between e and e' is positive.

- Dummy vertices for bends
- Traverse counter-clockwise
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

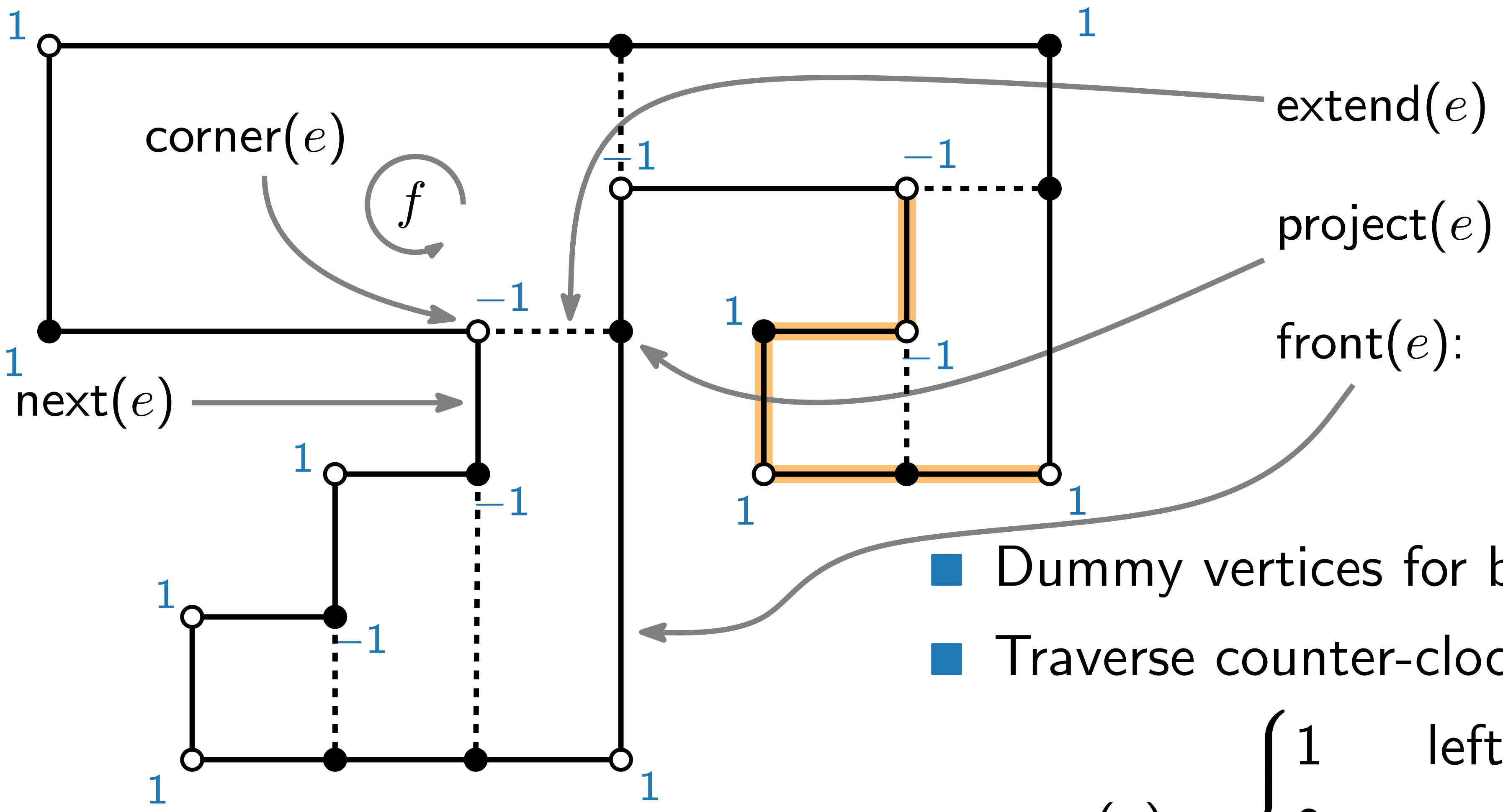
Refinement of G and $H(G)$ – Inner Face



front(e): First edge e after e such that the angle sum between e and e' is positive.

- Dummy vertices for bends
- Traverse counter-clockwise
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

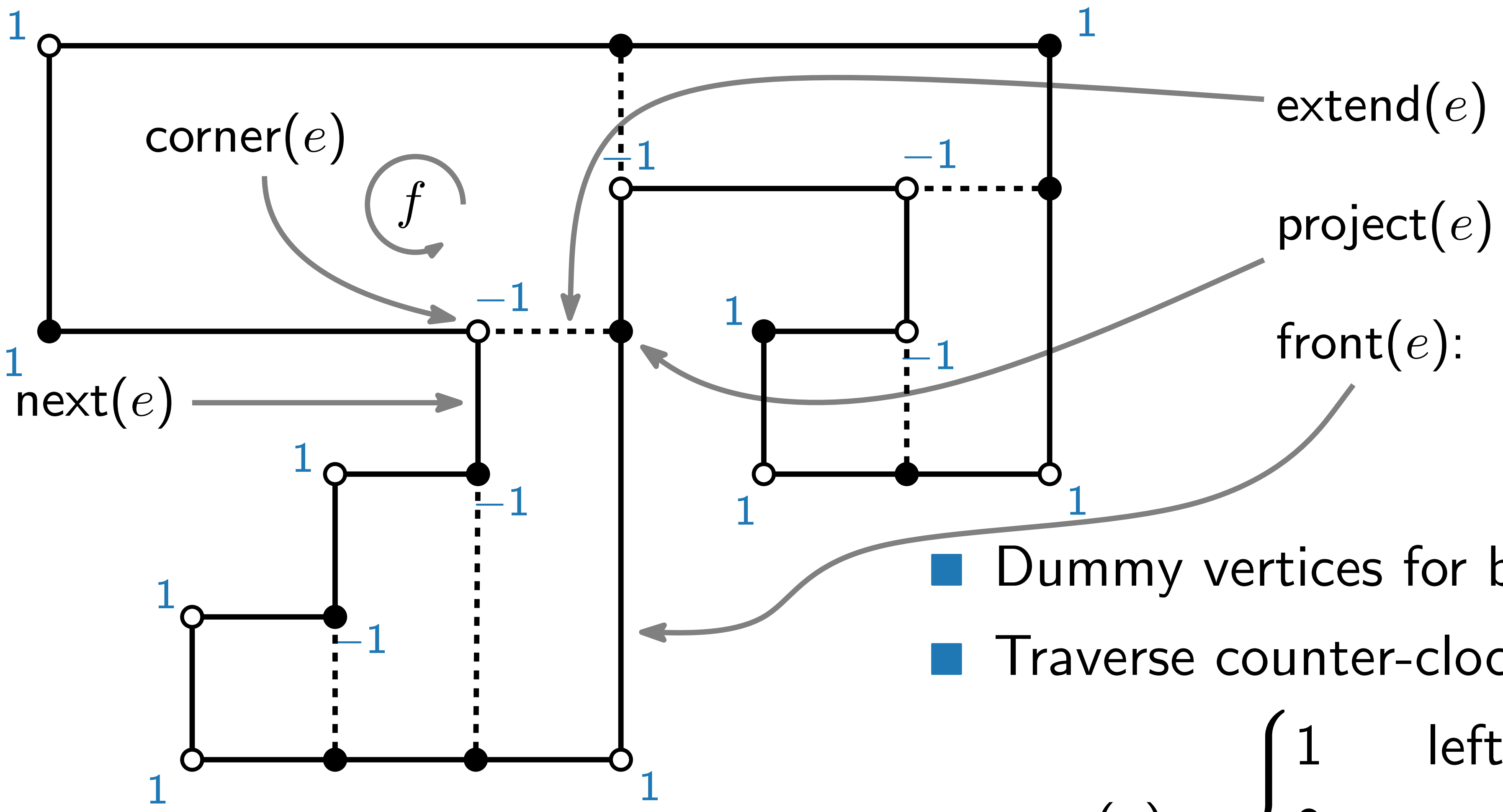
Refinement of G and $H(G)$ – Inner Face



front(e): First edge e after e such that the angle sum between e and e' is positive.

- Dummy vertices for bends
- Traverse counter-clockwise
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

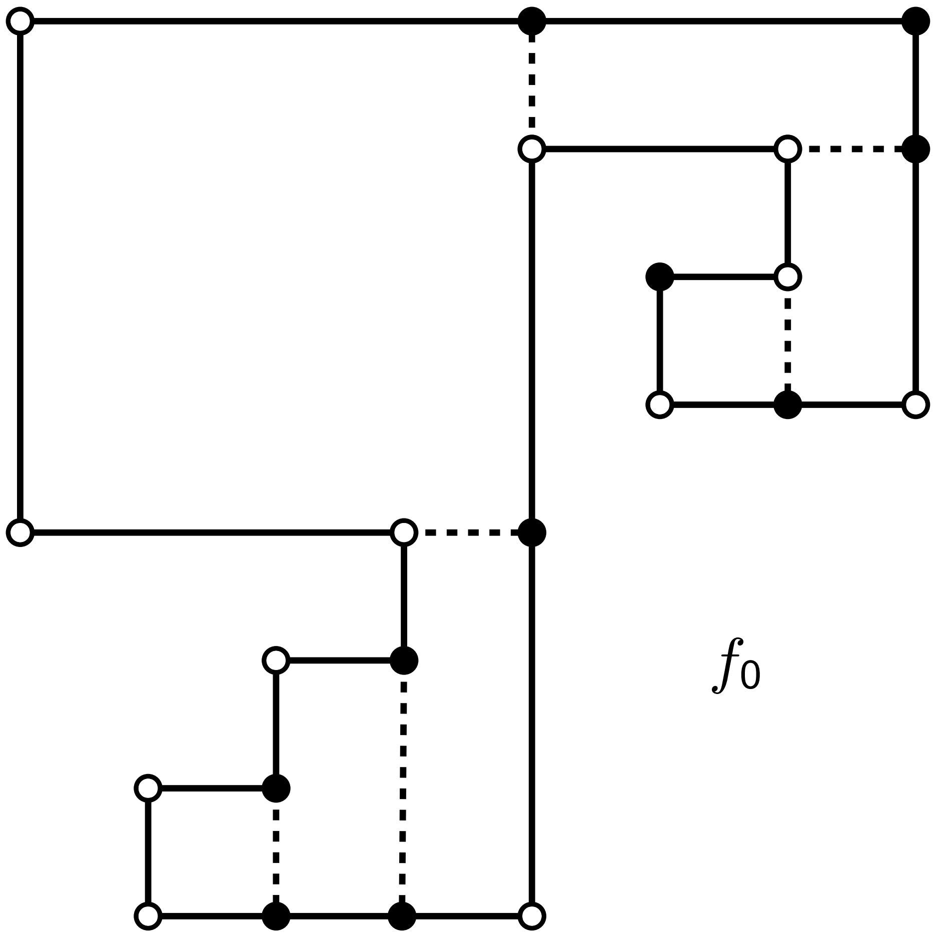
Refinement of G and $H(G)$ – Inner Face



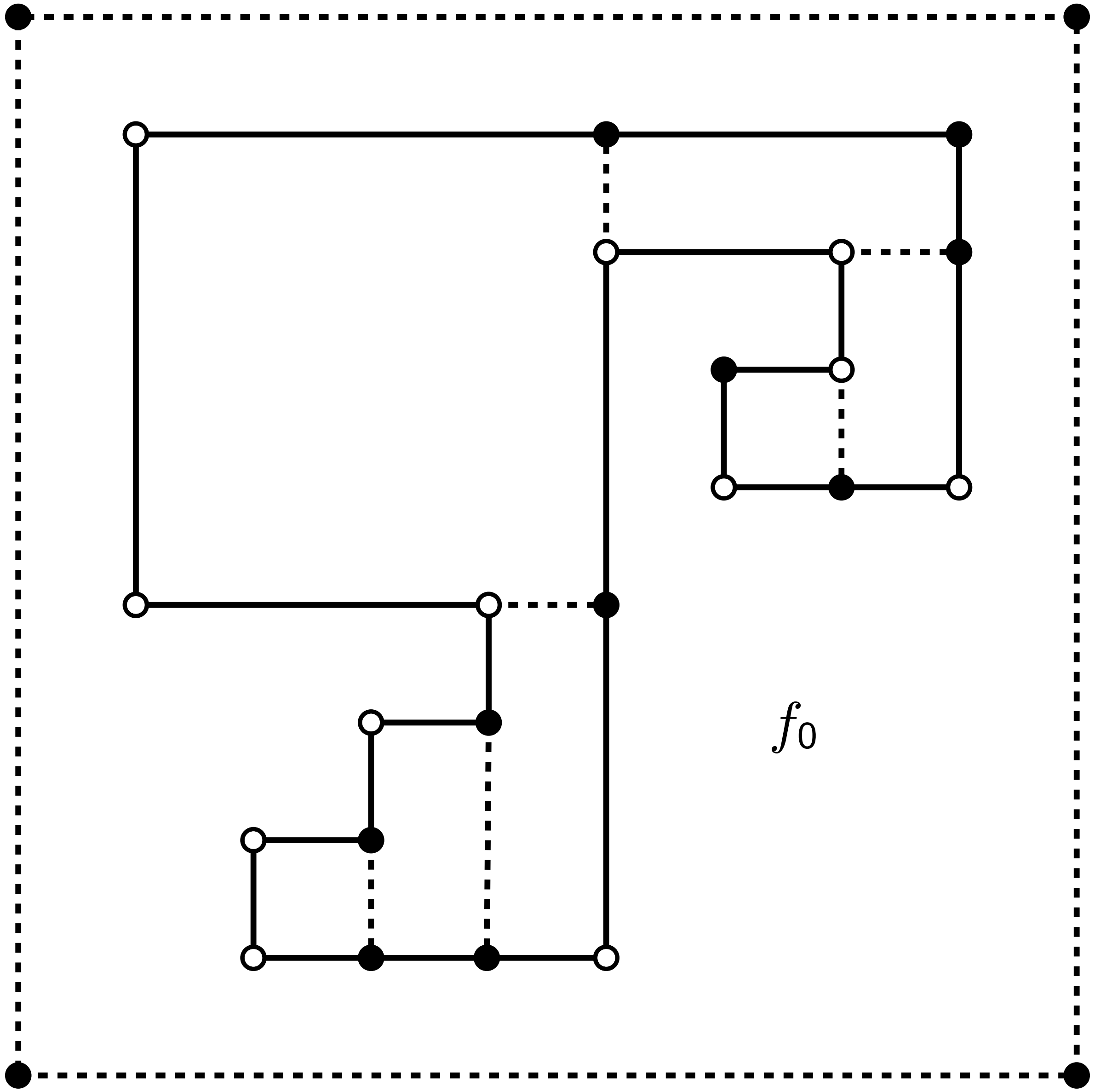
front(e): First edge e after e such that the angle sum between e and e' is positive.

- Dummy vertices for bends
- Traverse counter-clockwise
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of G and $H(G)$ – Outer Face

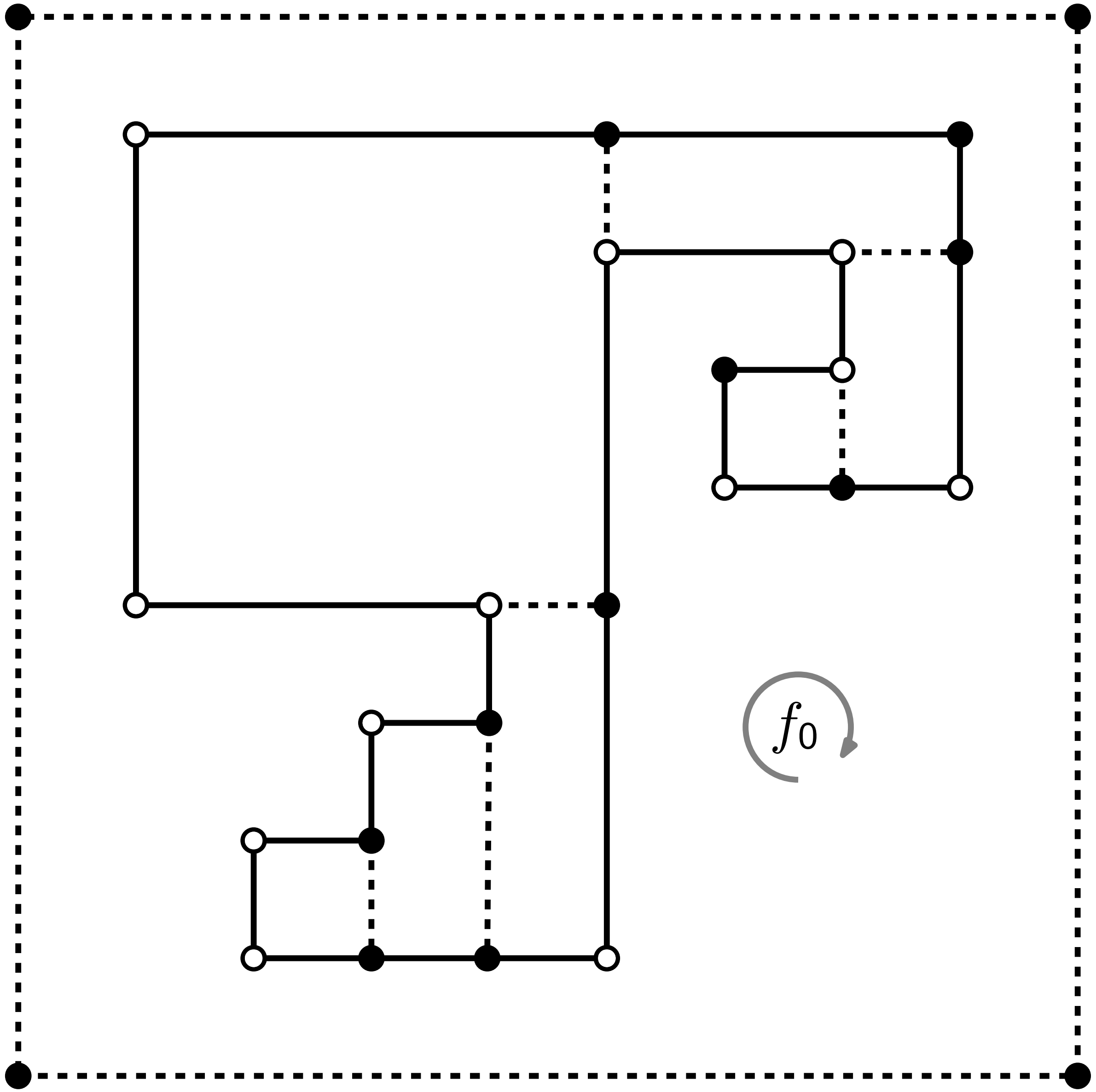


Refinement of G and $H(G)$ – Outer Face



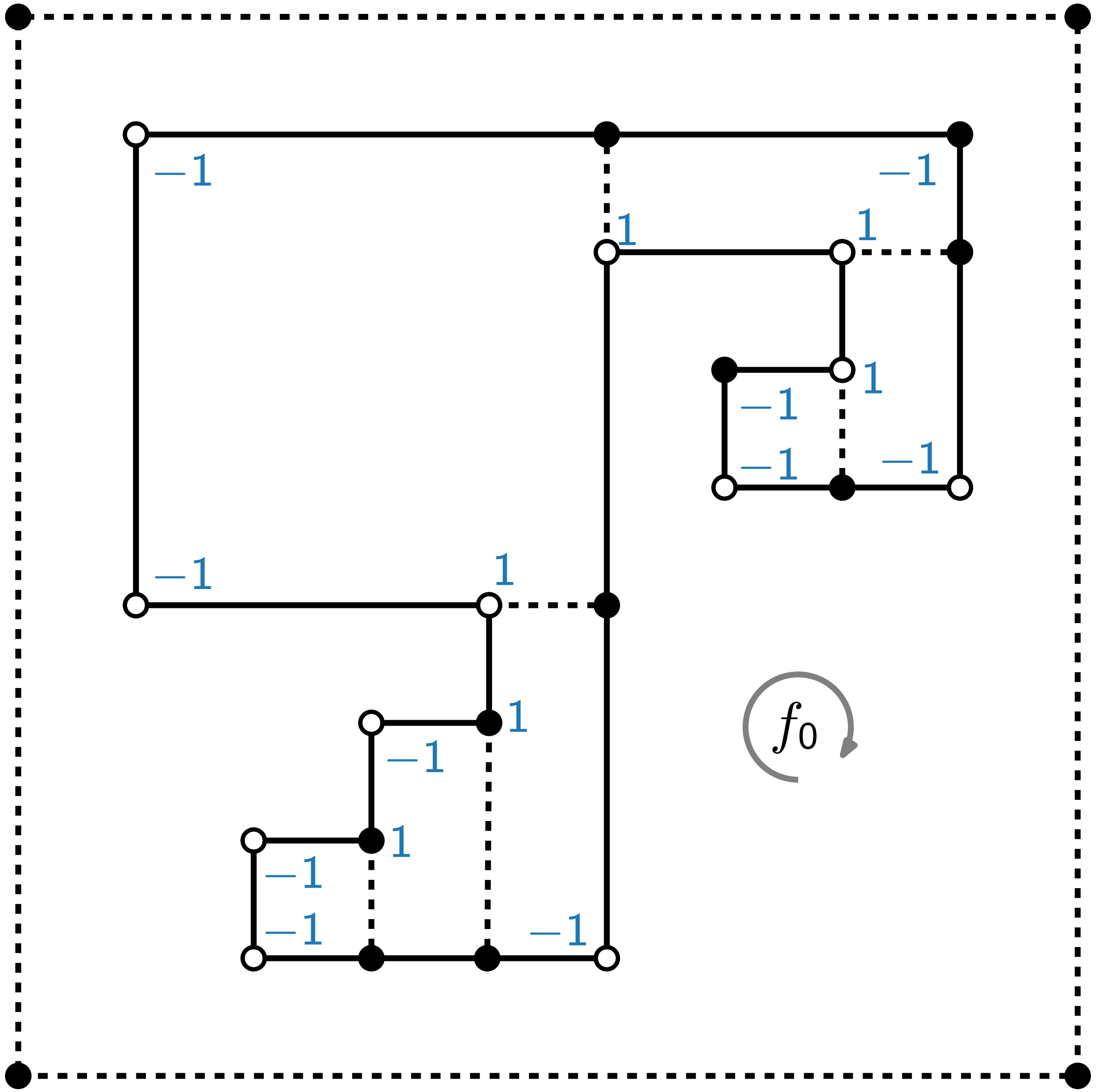
- Add an outer rectangle

Refinement of G and $H(G)$ – Outer Face



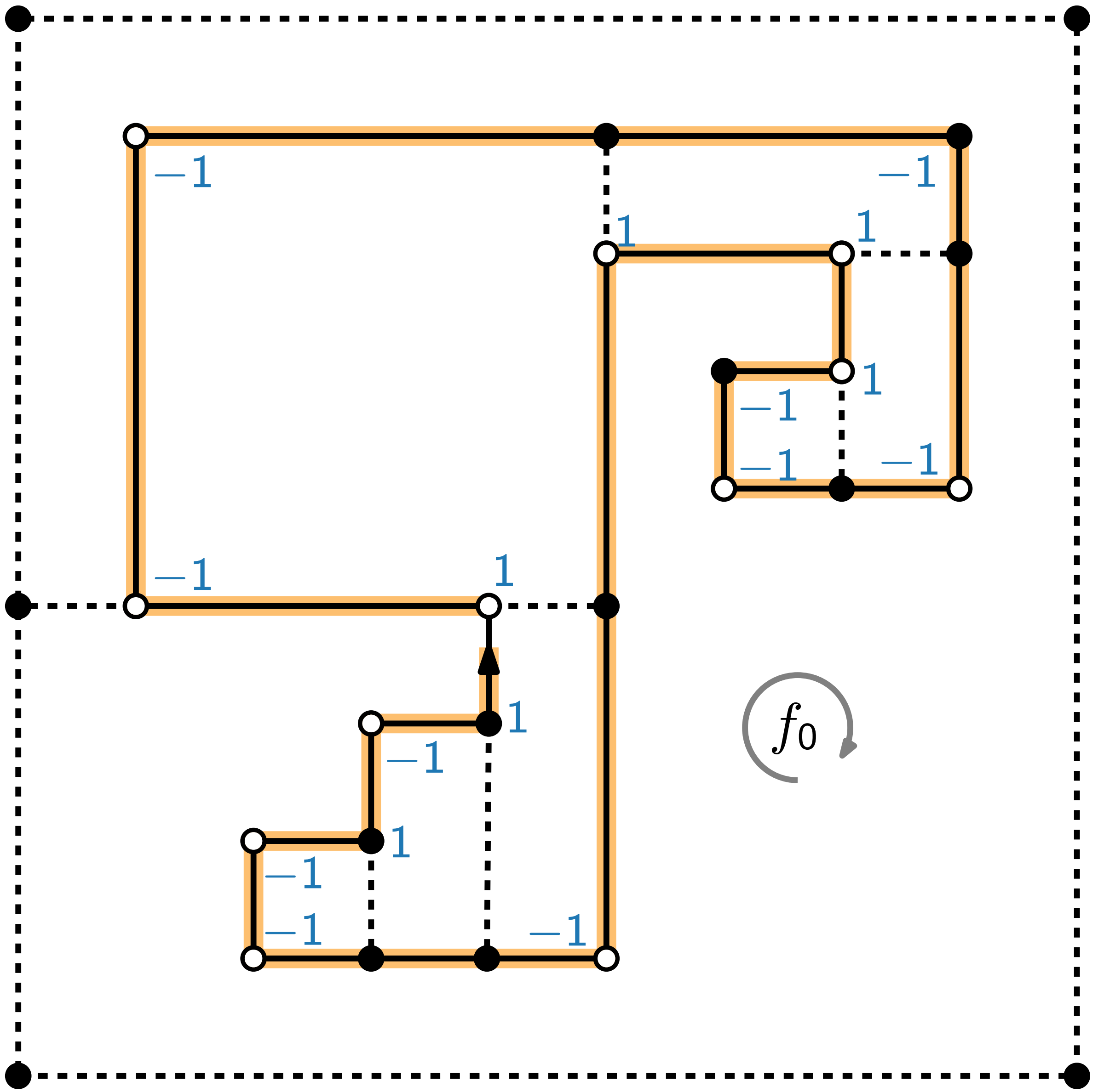
- Add an outer rectangle
- Traverse clockwise

Refinement of G and $H(G)$ – Outer Face



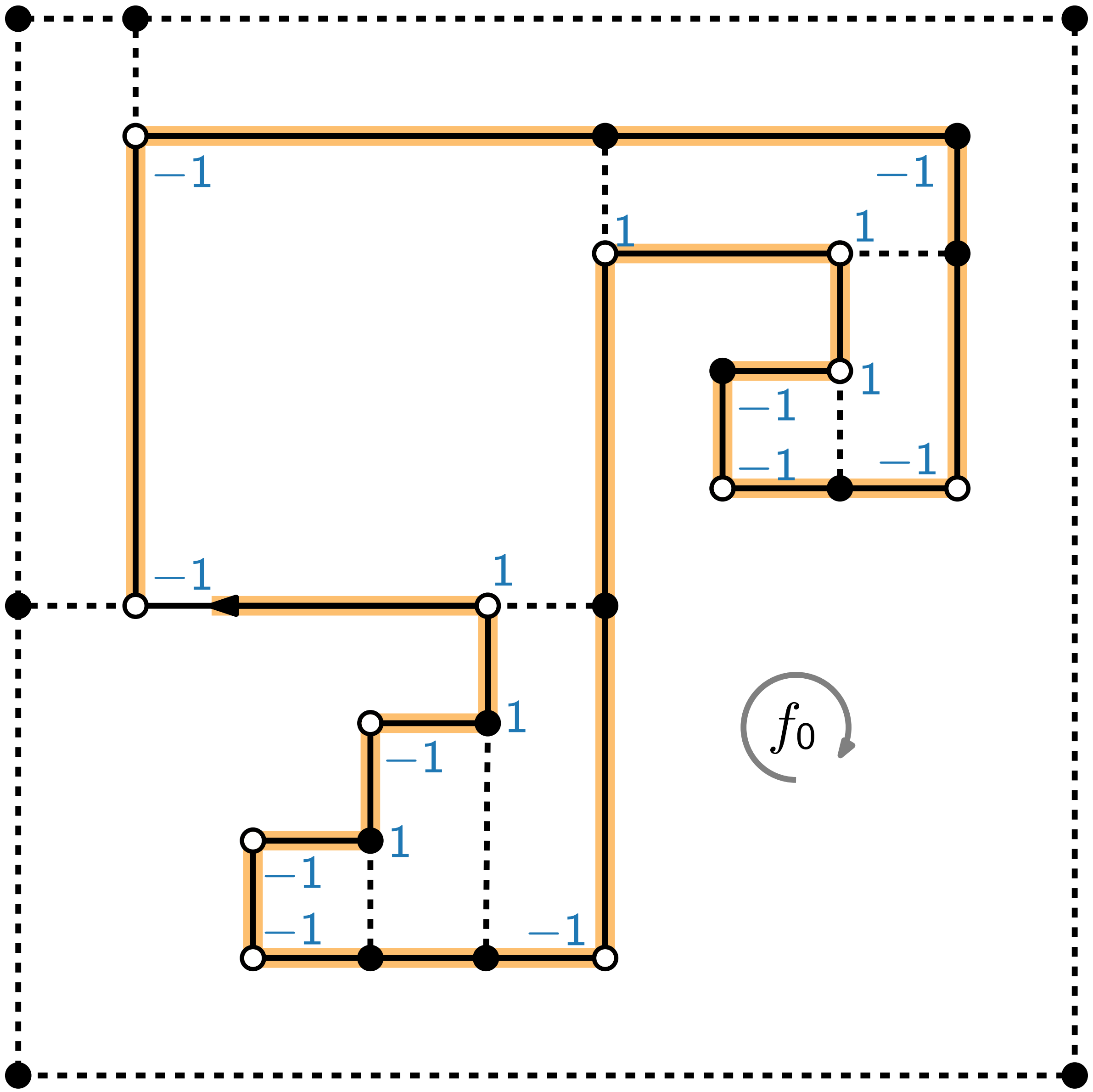
- Add an outer rectangle
- Traverse clockwise

Refinement of G and $H(G)$ – Outer Face



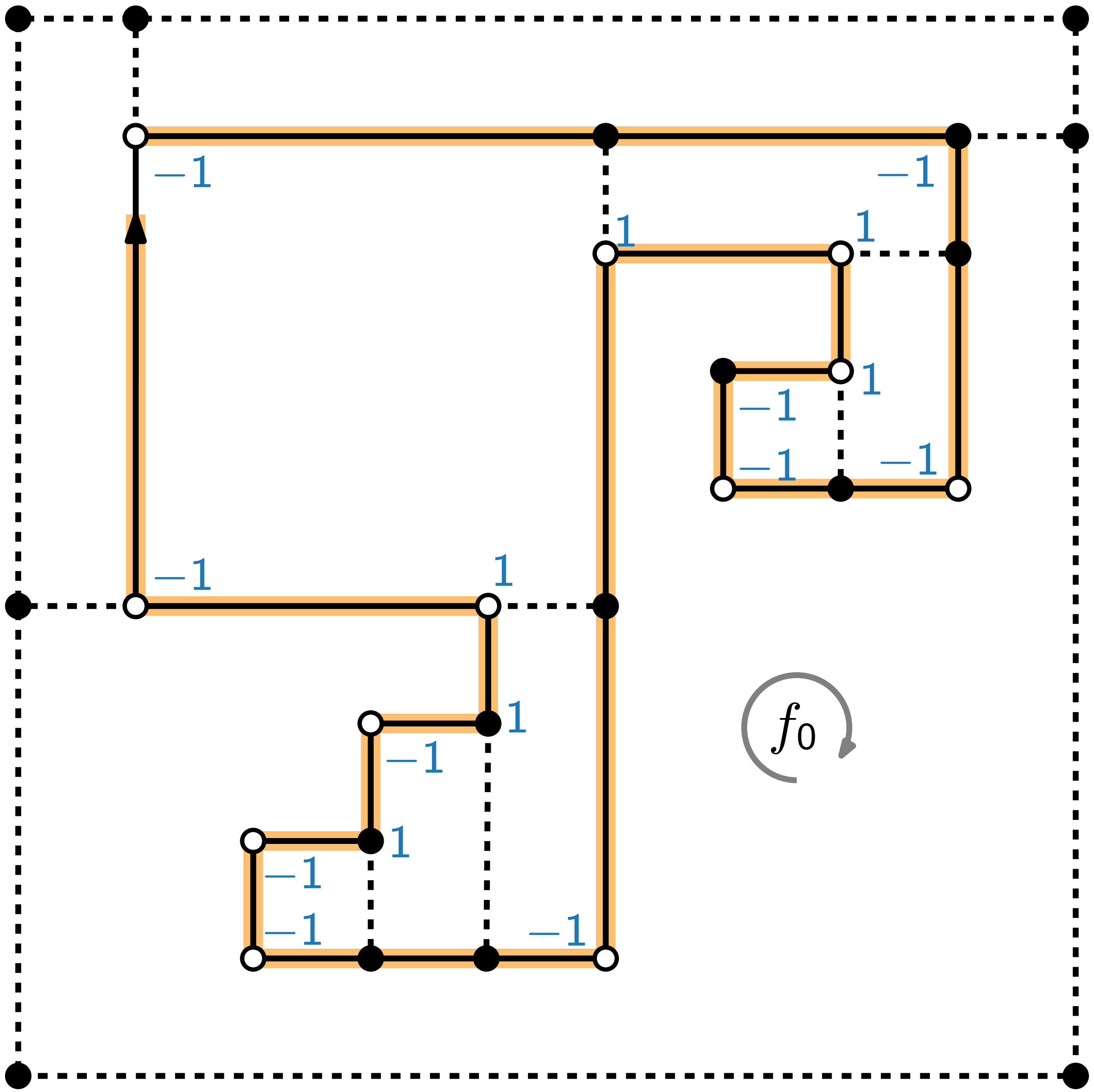
- Add an outer rectangle
- Traverse clockwise

Refinement of G and $H(G)$ – Outer Face



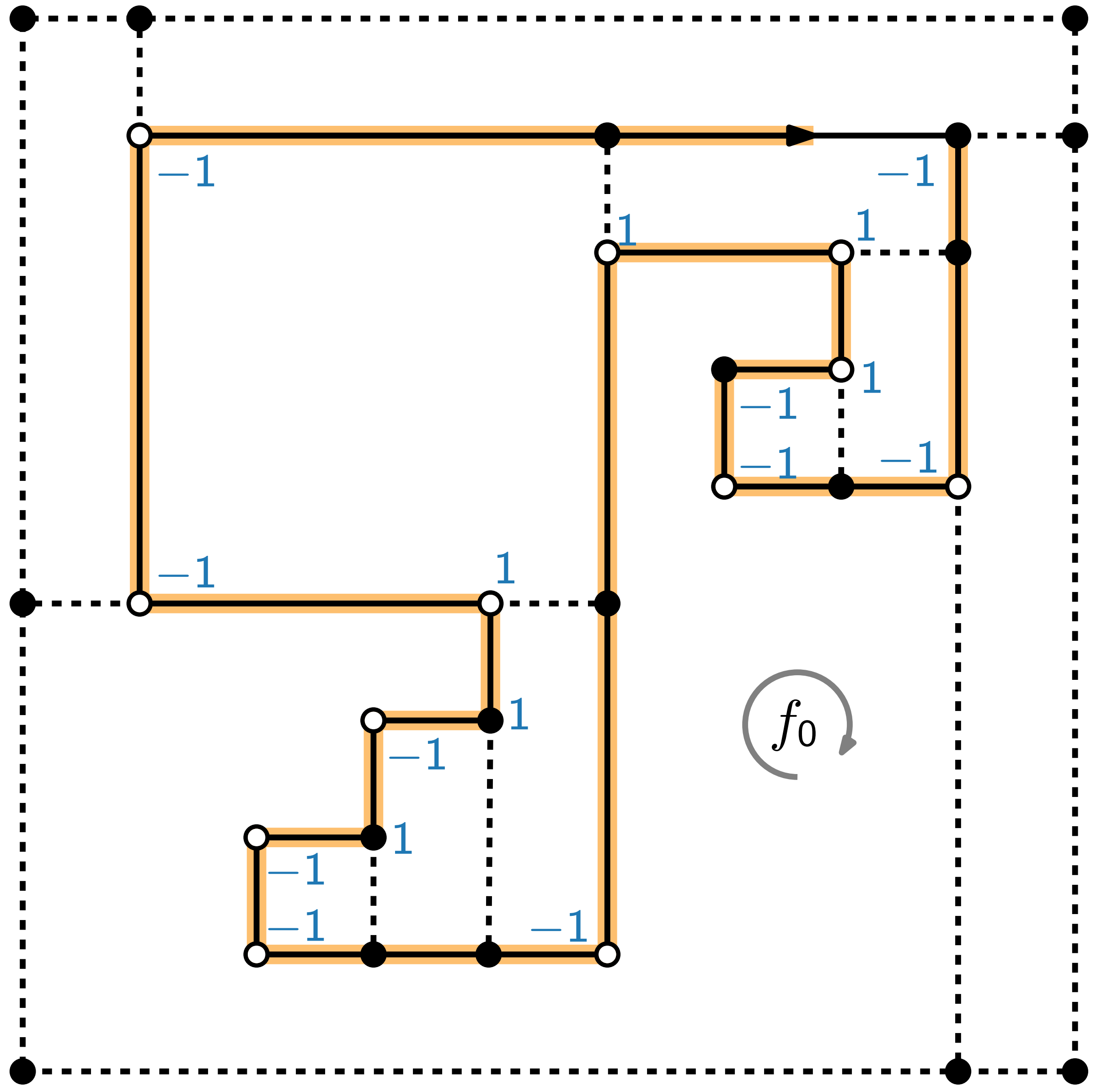
- Add an outer rectangle
- Traverse clockwise

Refinement of G and $H(G)$ – Outer Face



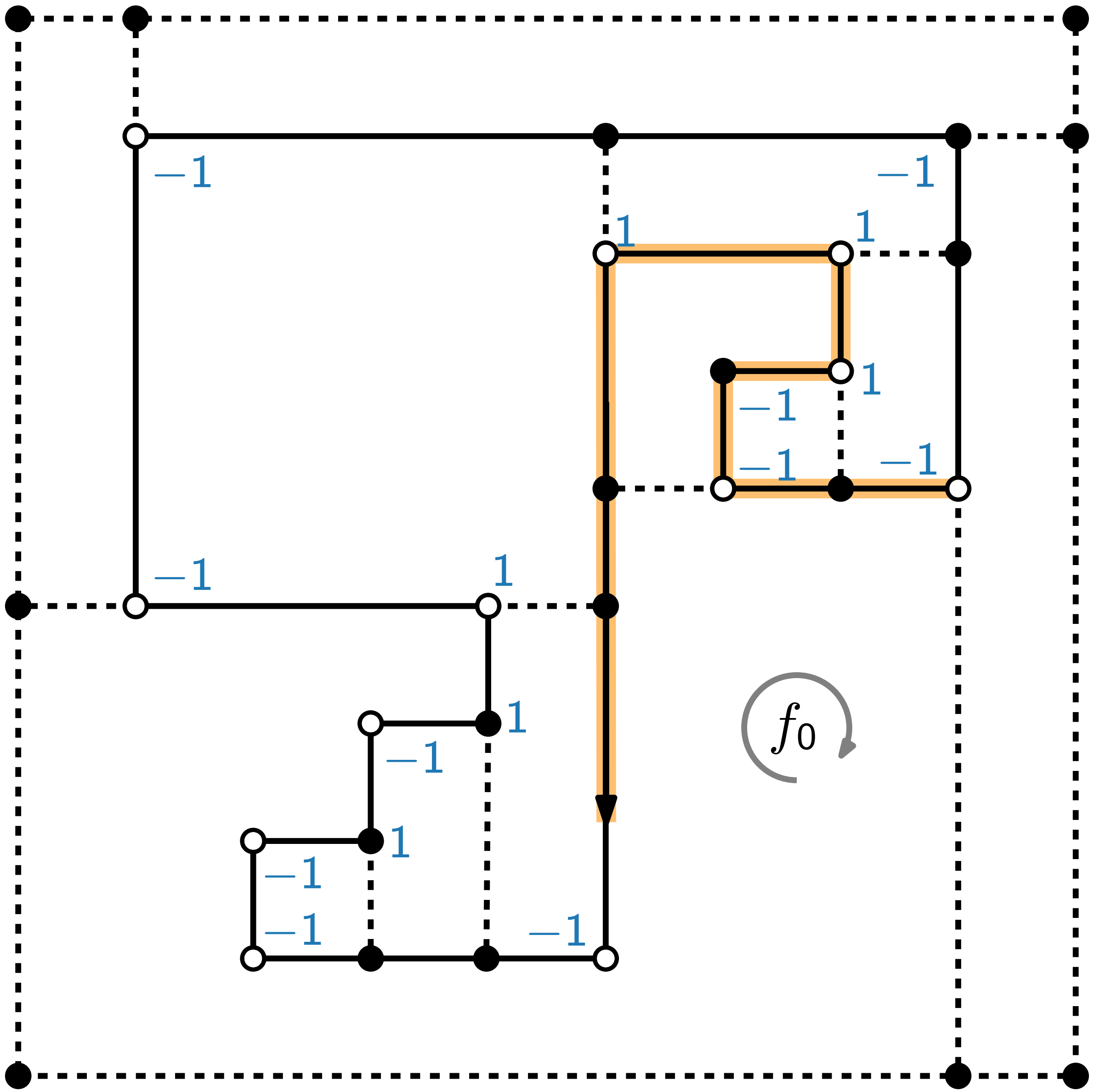
- Add an outer rectangle
- Traverse clockwise

Refinement of G and $H(G)$ – Outer Face



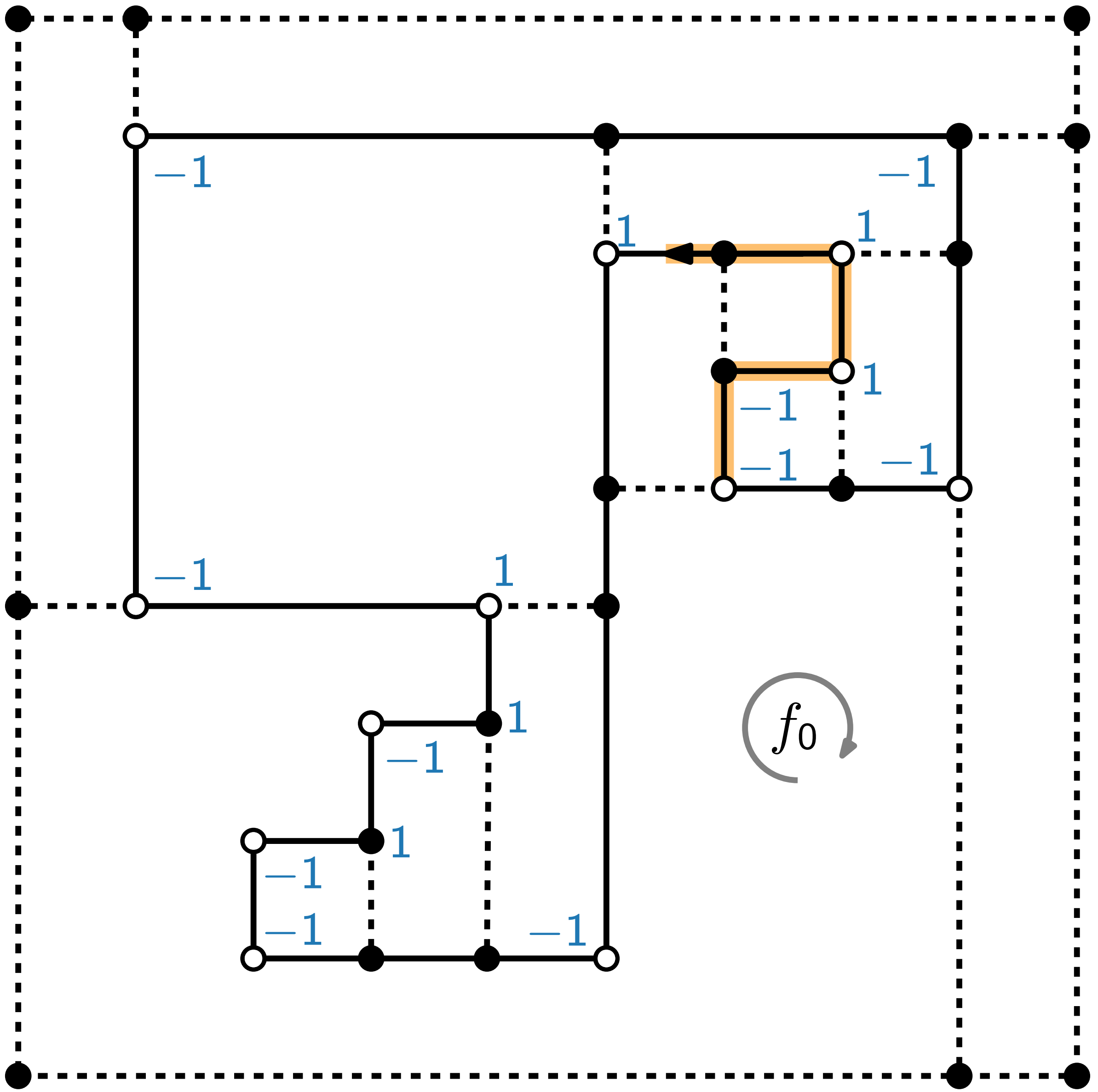
- Add an outer rectangle
- Traverse clockwise

Refinement of G and $H(G)$ – Outer Face



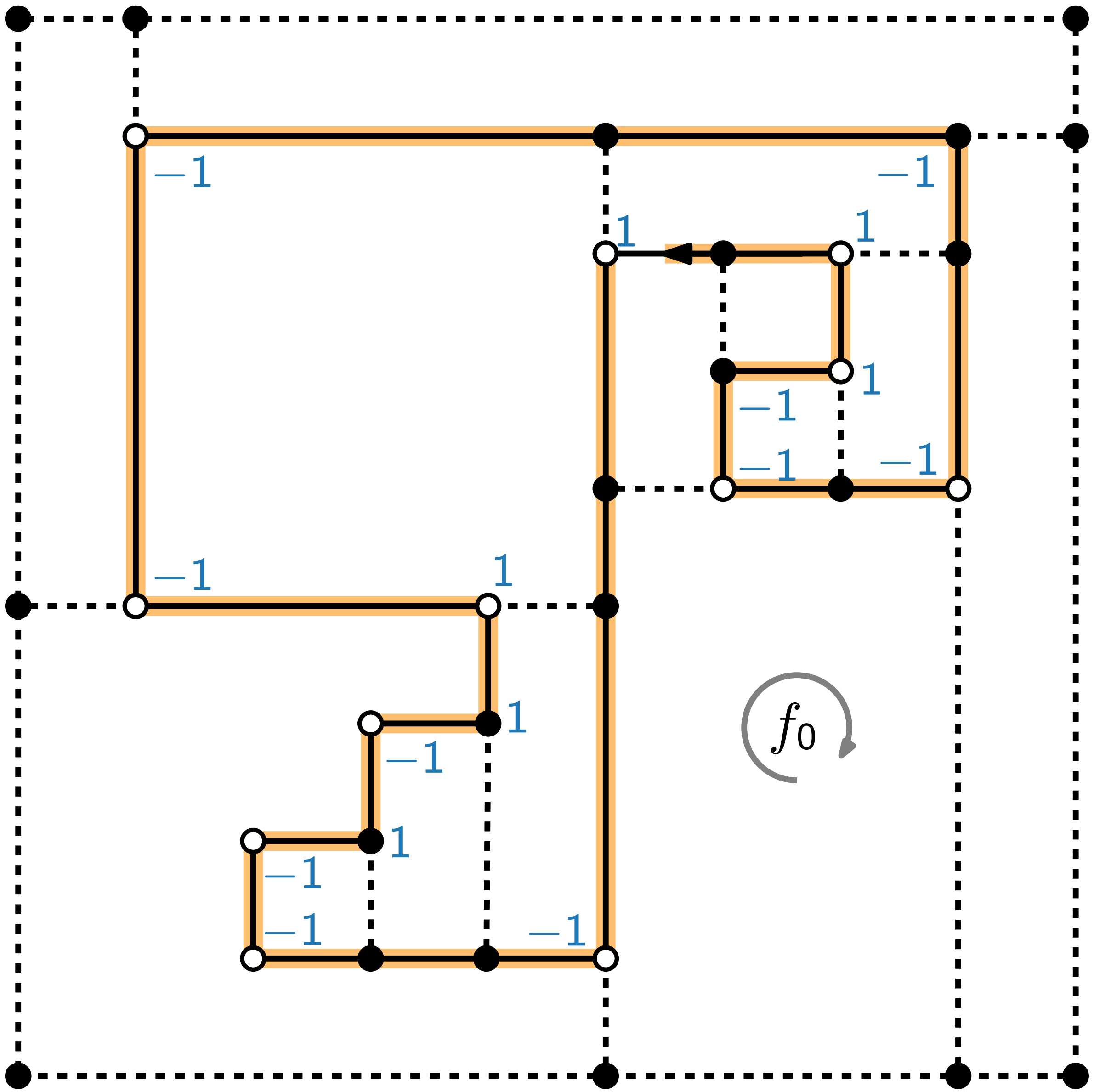
- Add an outer rectangle
- Traverse clockwise

Refinement of G and $H(G)$ – Outer Face



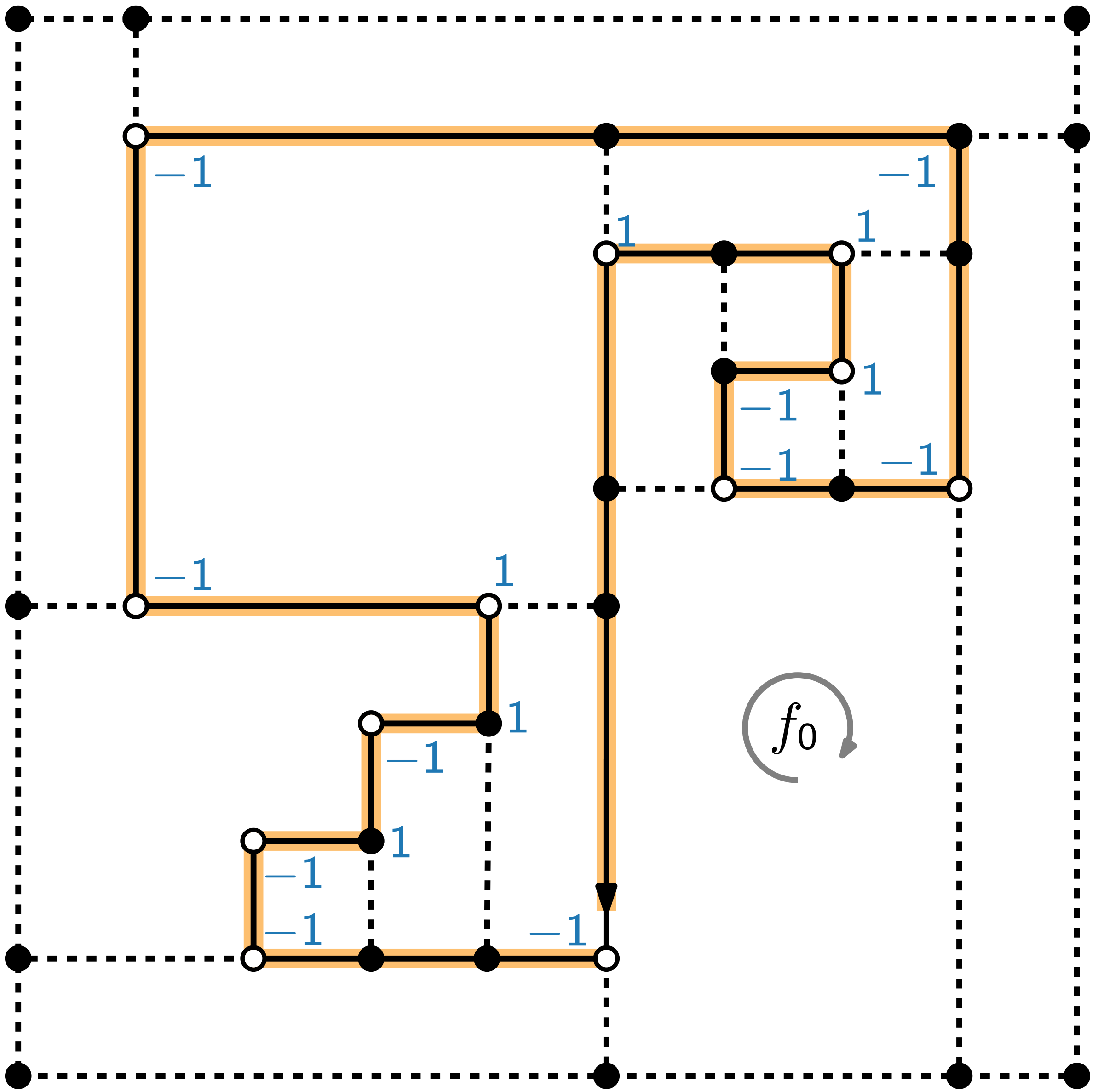
- Add an outer rectangle
- Traverse clockwise

Refinement of G and $H(G)$ – Outer Face



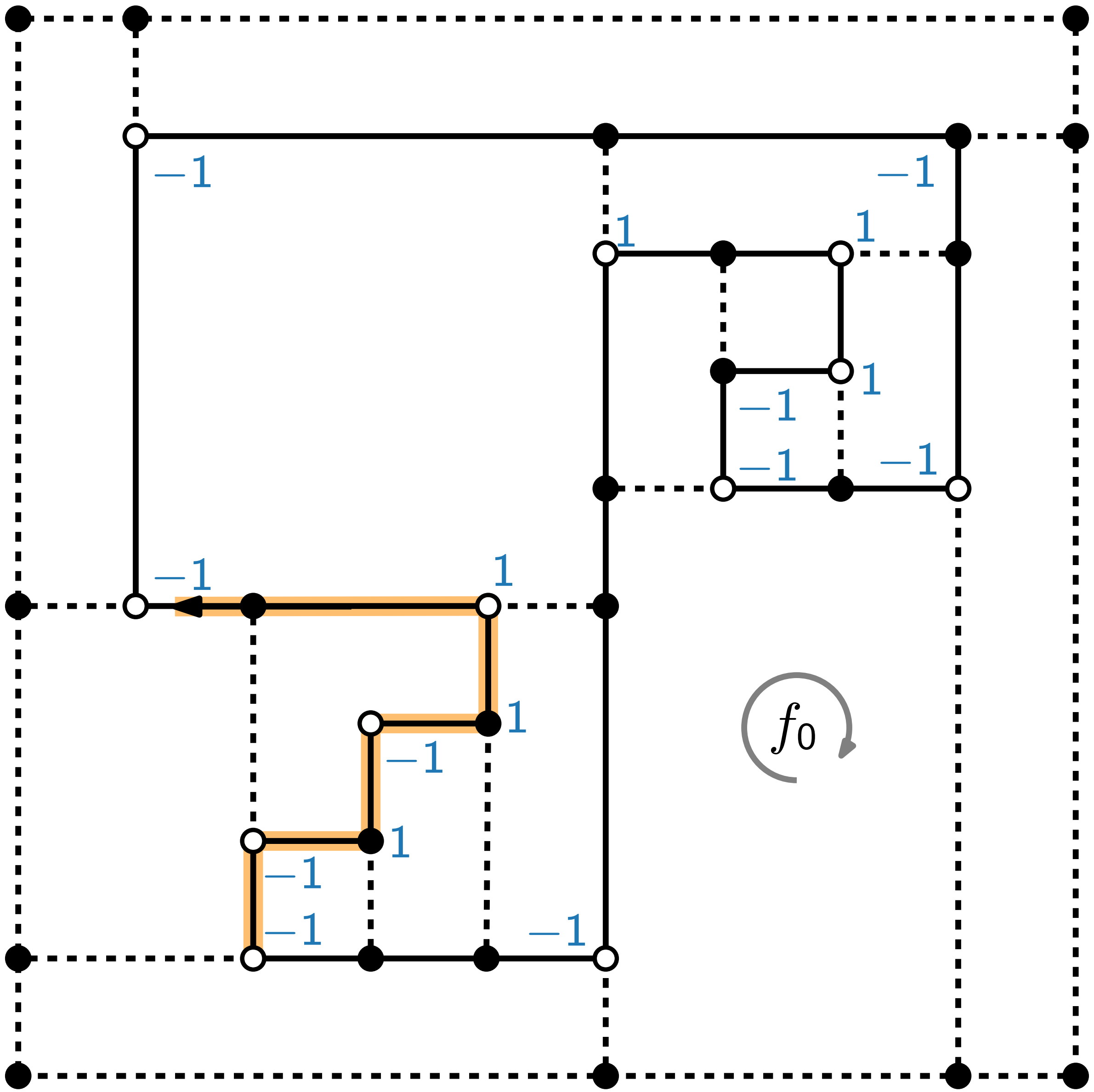
- Add an outer rectangle
- Traverse clockwise

Refinement of G and $H(G)$ – Outer Face



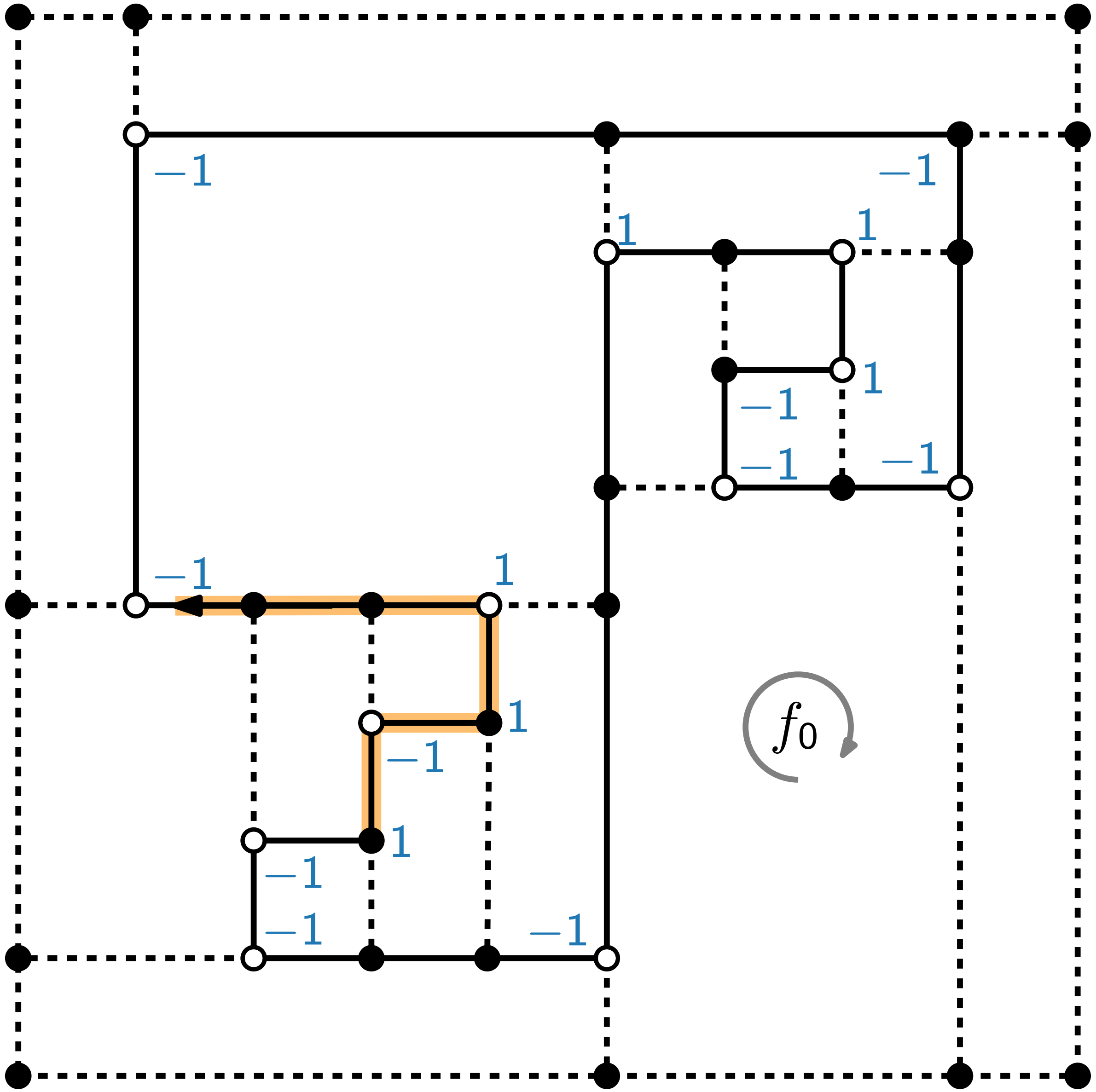
- Add an outer rectangle
- Traverse clockwise

Refinement of G and $H(G)$ – Outer Face



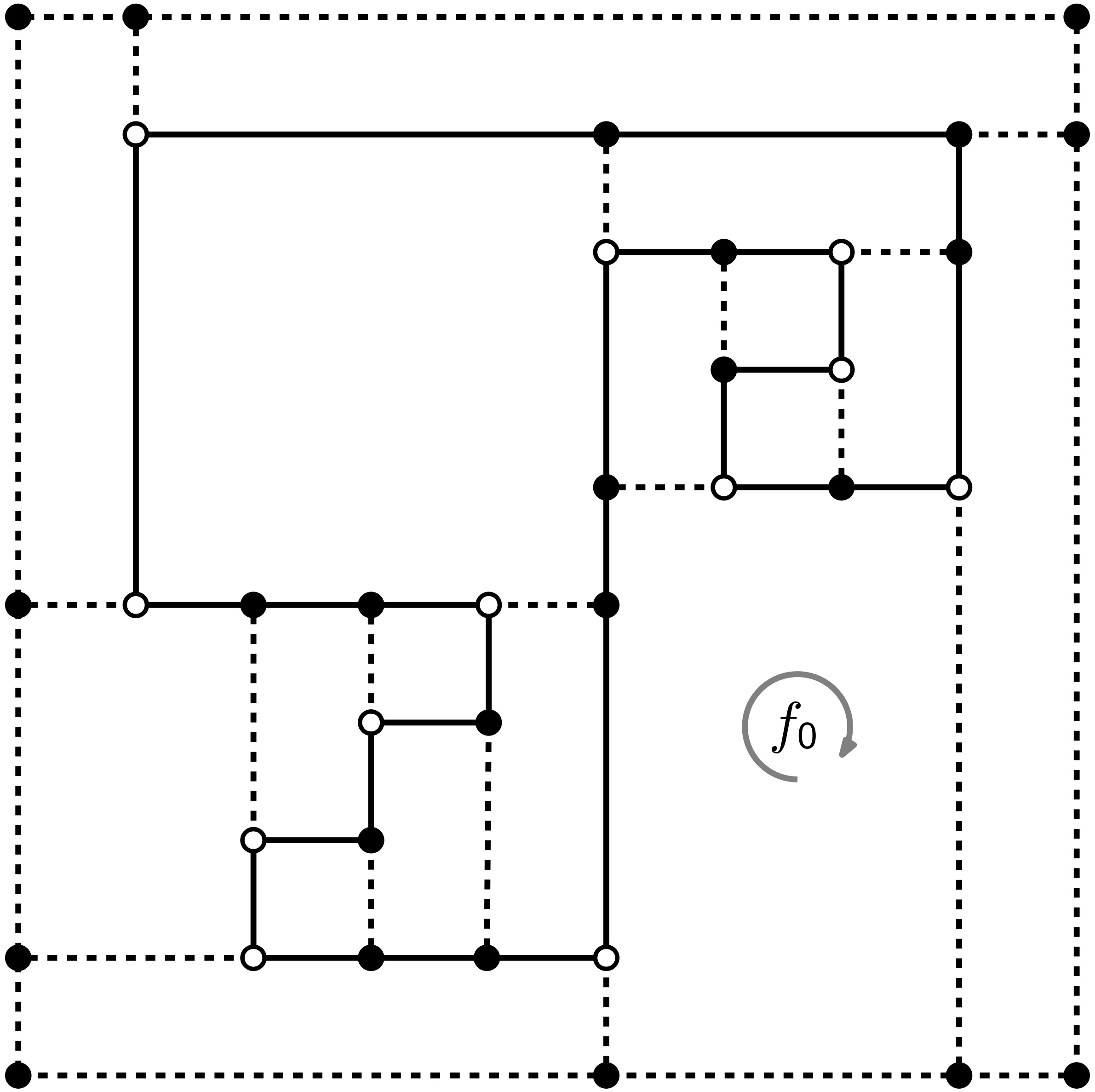
- Add an outer rectangle
- Traverse clockwise

Refinement of G and $H(G)$ – Outer Face



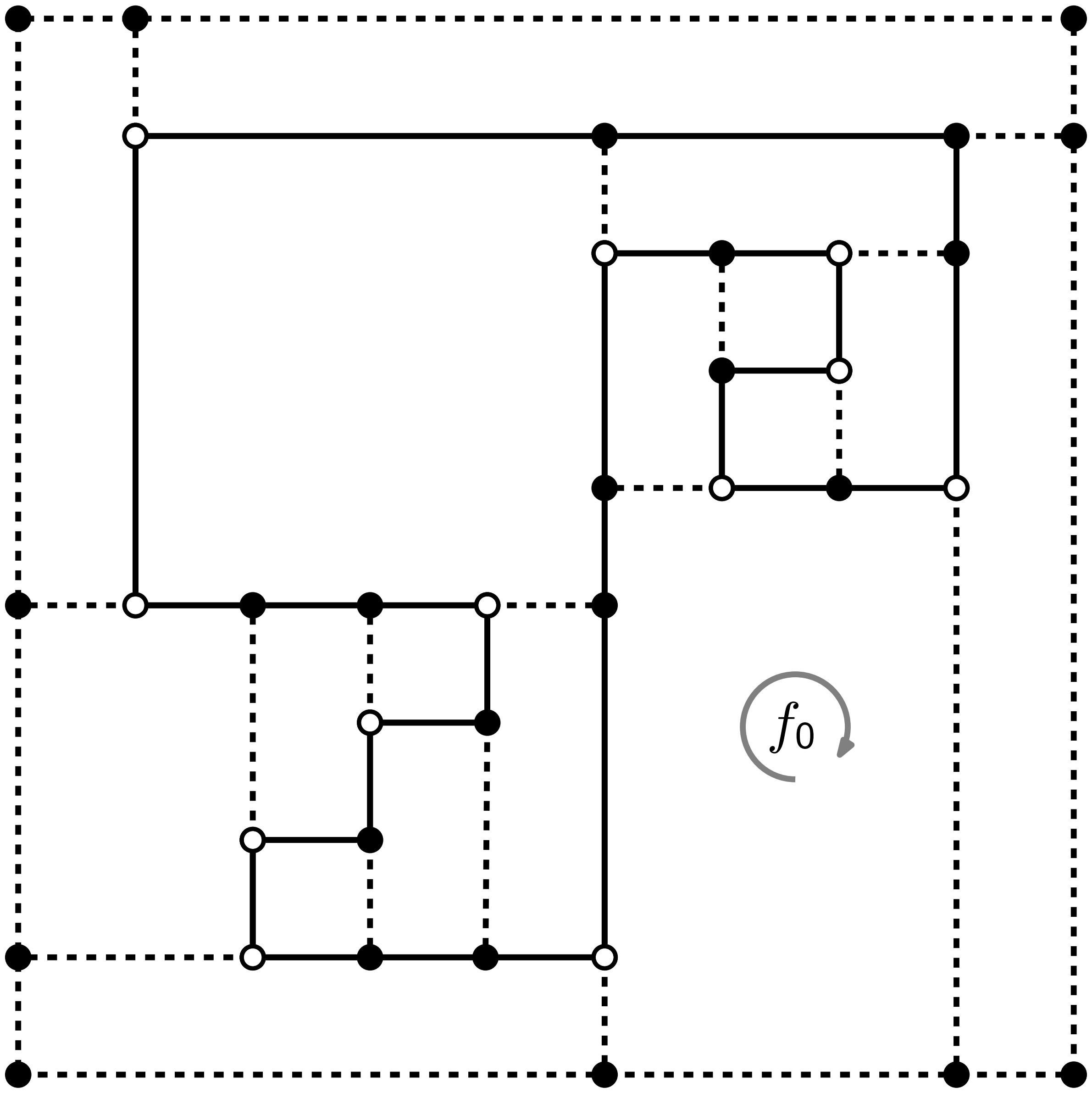
- Add an outer rectangle
- Traverse clockwise

Refinement of G and $H(G)$ – Outer Face



- Add an outer rectangle
- Traverse clockwise

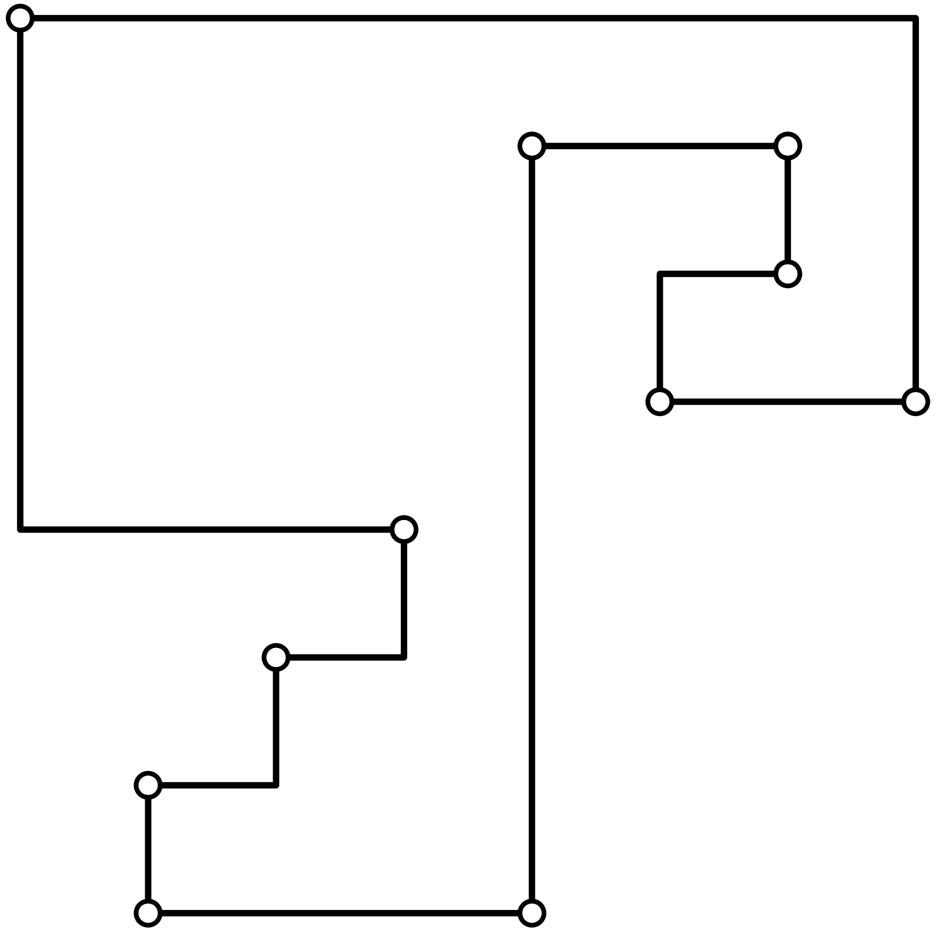
Refinement of G and $H(G)$ – Outer Face



- Add an outer rectangle
- Traverse clockwise

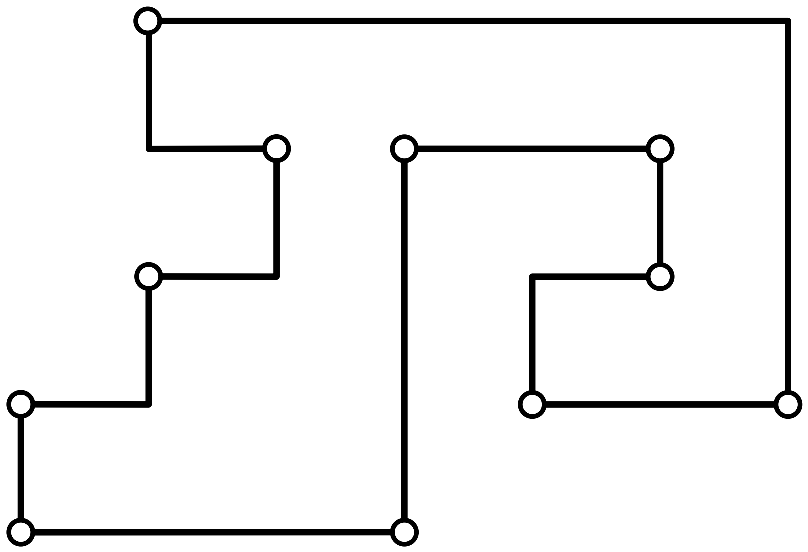
Area minimized?

Refinement of G and $H(G)$ – Outer Face



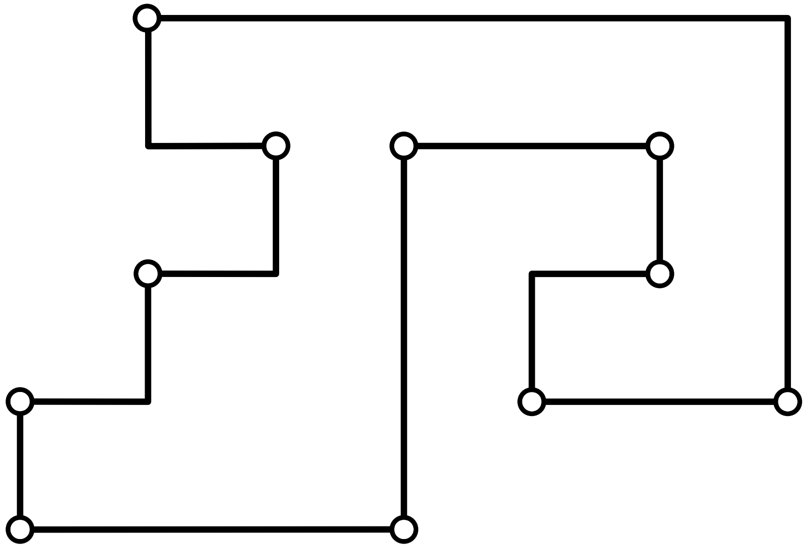
Area minimized?

Refinement of G and $H(G)$ – Outer Face



Area minimized? No!

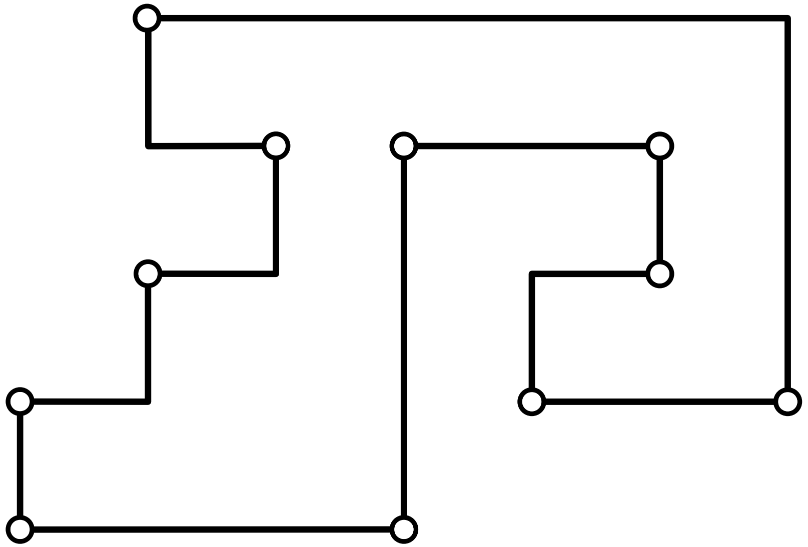
Refinement of G and $H(G)$ – Outer Face



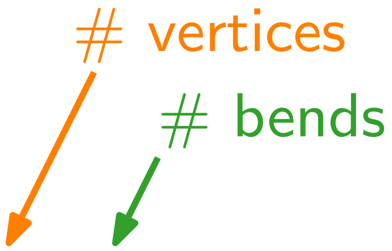
Area minimized? No!

But we get bound $O((n+b)^2)$ on the area.

Refinement of G and $H(G)$ – Outer Face

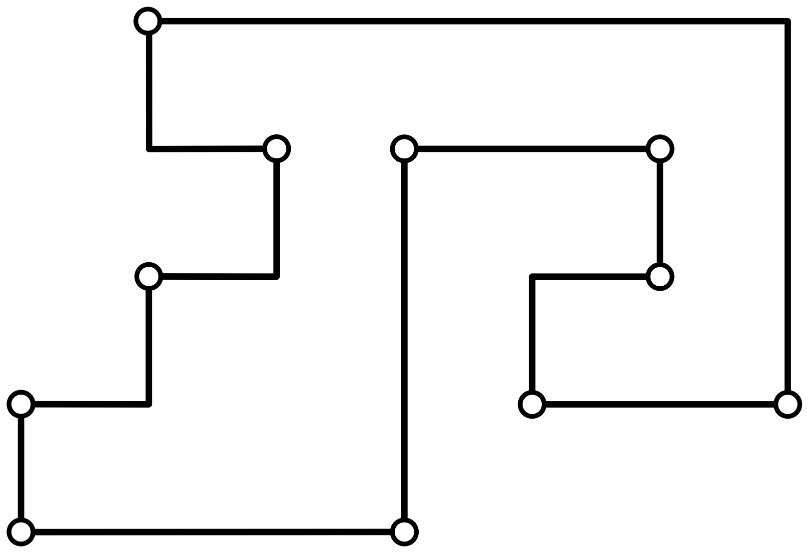


Area minimized? No!



But we get bound $O((n + b)^2)$ on the area.

Refinement of G and $H(G)$ – Outer Face



Area minimized? No!

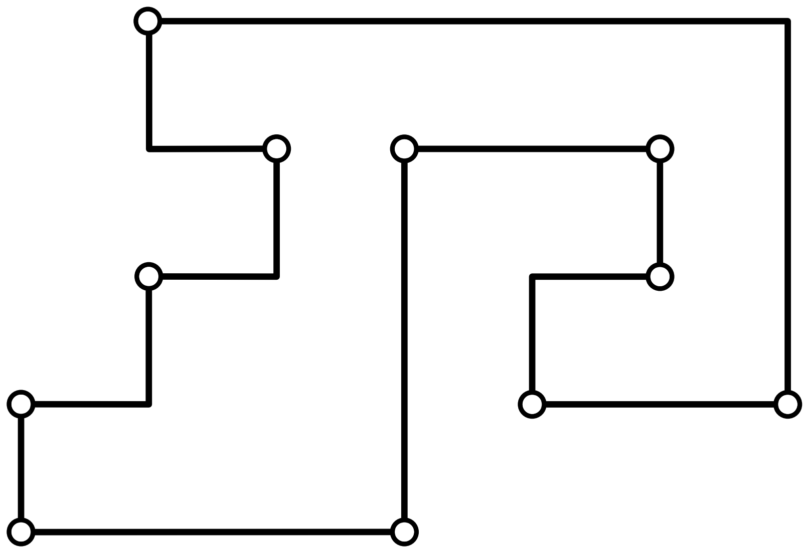
vertices

bends

But we get bound $O((n+b)^2)$ on the area.

Theorem. [Patrignani 2001]
Compaction for a given orthogonal representation is NP-hard in general.

Refinement of G and $H(G)$ – Outer Face



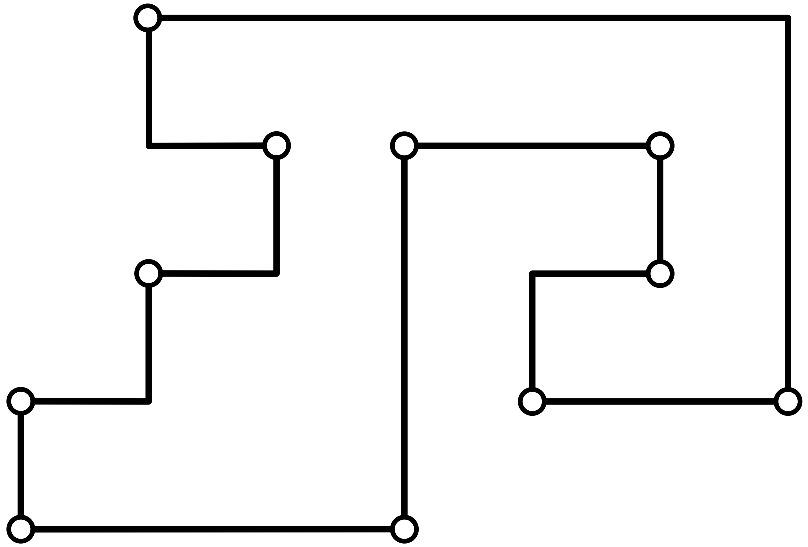
Area minimized? No!

But we get bound $O((n + b)^2)$ on the area.

Theorem. [Patrignani 2001]
Compaction for a given orthogonal representation is NP-hard in general.

Theorem. [EFKSSW 2022]
Compaction is NP-hard even for orthogonal representations of *cycles*.

Refinement of G and $H(G)$ – Outer Face



Area minimized? No!

vertices

bends

But we get bound $O((n + b)^2)$ on the area.

Theorem. [Patrignani 2001]
Compaction for a given orthogonal representation is NP-hard in general.

Theorem. [EFKSSW 2022]
Compaction is NP-hard even for orthogonal representations of *cycles*.

Compaction is NP-hard

Polynomial-time reduction from the NP-complete satisfiability problem (SAT).

Compaction is NP-hard

Polynomial-time reduction from the NP-complete satisfiability problem (SAT).

In an instance of the SAT problem we have:

Compaction is NP-hard

Polynomial-time reduction from the NP-complete satisfiability problem (SAT).

In an instance of the SAT problem we have:

- set of n Boolean variables $X = \{x_1, x_2, \ldots, x_n\}$

Compaction is NP-hard

Polynomial-time reduction from the NP-complete satisfiability problem (SAT).

In an instance of the SAT problem we have:

- set of n Boolean variables $X = \{x_1, x_2, \ldots, x_n\}$
- m clauses $C_1, C_2, \ldots, C_m$

Compaction is NP-hard

Polynomial-time reduction from the NP-complete satisfiability problem (SAT).

In an instance of the SAT problem we have:

- set of n Boolean variables $X = \{x_1, x_2, \ldots, x_n\}$
- m clauses $C_1, C_2, \ldots, C_m$, where
  each clause is a disjunction of literals from X,

Compaction is NP-hard

Polynomial-time reduction from the NP-complete satisfiability problem (SAT).

In an instance of the SAT problem we have:

- set of n Boolean variables $X = \{x_1, x_2, \ldots, x_n\}$
- m clauses $C_1, C_2, \ldots, C_m$, where each clause is a disjunction of literals from X, e.g., $C_1 = x_1 \vee \neg x_2 \vee x_3$

a literal is a variable x or a negated variable $\neg x$

Compaction is NP-hard

Polynomial-time reduction from the NP-complete satisfiability problem (SAT).

In an instance of the SAT problem we have:

- set of n Boolean variables $X = \{x_1, x_2, \dots, x_n\}$
- m clauses $C_1, C_2, \dots, C_m$, where each clause is a disjunction of literals from X, e.g., $C_1 = x_1 \vee \neg x_2 \vee x_3$

  a literal is a variable x or a negated variable $\neg x$
- Boolean formula $\Phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$

Compaction is NP-hard

Polynomial-time reduction from the NP-complete satisfiability problem (SAT).

In an instance of the SAT problem we have:

- set of n Boolean variables $X = \{x_1, x_2, \ldots, x_n\}$
- m clauses $C_1, C_2, \ldots, C_m$, where each clause is a disjunction of literals from X, e.g., $C_1 = x_1 \vee \neg x_2 \vee x_3$

  a literal is a variable x or a negated variable $\neg x$
- Boolean formula $\Phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$

Question: Is there an assignment of truth values to the variables in X such that Φ is true?

Compaction is NP-hard

Polynomial-time reduction from the NP-complete satisfiability problem (SAT).

In an instance of the SAT problem we have:

- set of n Boolean variables $X = \{x_1, x_2, \ldots, x_n\}$
- m clauses $C_1, C_2, \ldots, C_m$, where each clause is a disjunction of literals from X, e.g., $C_1 = x_1 \lor \neg x_2 \lor x_3$

  a literal is a variable x or a negated variable $\neg x$
- Boolean formula $\Phi = C_1 \land C_2 \land \cdots \land C_m$

Question: Is there an assignment of truth values to the variables in X such that Φ is true?

Idea of the reduction:

Compaction is NP-hard

Polynomial-time reduction from the NP-complete satisfiability problem (SAT).

In an instance of the SAT problem we have:

- set of n Boolean variables $X = \{x_1, x_2, \ldots, x_n\}$
- m clauses $C_1, C_2, \ldots, C_m$, where each clause is a disjunction of literals from X, e.g., $C_1 = x_1 \vee \neg x_2 \vee x_3$

  a literal is a variable x or a negated variable $\neg x$
- Boolean formula $\Phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$

Question: Is there an assignment of truth values to the variables in X such that Φ is true?

Idea of the reduction:

- Given SAT instance $\Phi \Rightarrow$ construct a plane graph G and a orthogonal description $H(G)$

Compaction is NP-hard

Polynomial-time reduction from the NP-complete satisfiability problem (SAT).

In an instance of the SAT problem we have:

- set of n Boolean variables $X = \{x_1, x_2, \ldots, x_n\}$
- m clauses $C_1, C_2, \ldots, C_m$, where each clause is a disjunction of literals from X, e.g., $C_1 = x_1 \vee \neg x_2 \vee x_3$

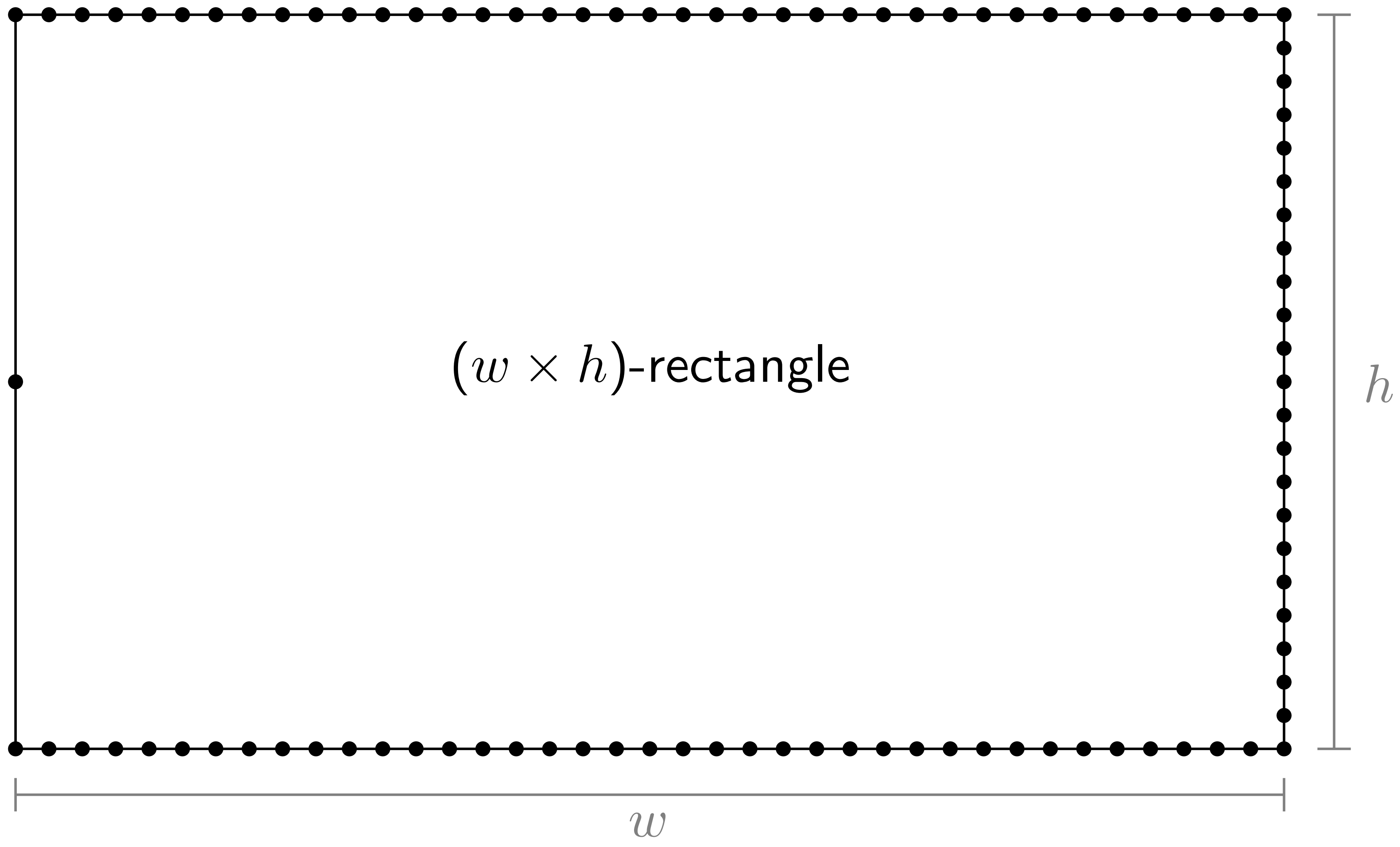
  a literal is a variable x or a negated variable $\neg x$
- Boolean formula $\Phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$

Question: Is there an assignment of truth values to the variables in X such that Φ is true?

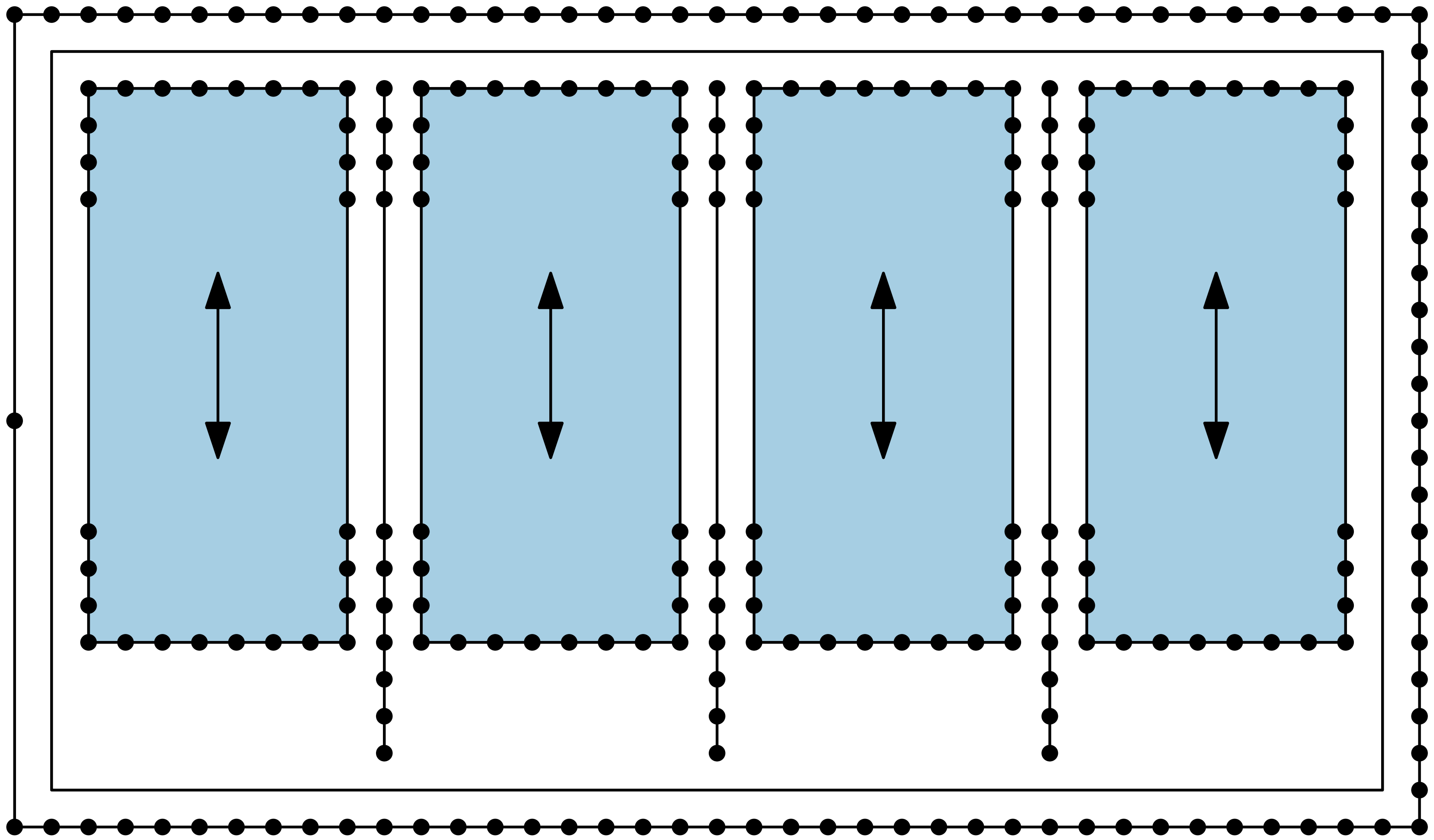
Idea of the reduction:

- Given SAT instance $\Phi \Rightarrow$ construct a plane graph G and a orthogonal description $H(G)$
- Φ is satisfiable $\Leftrightarrow$ G can be drawn w.r.t. $H(G)$ in area K for some specific number K

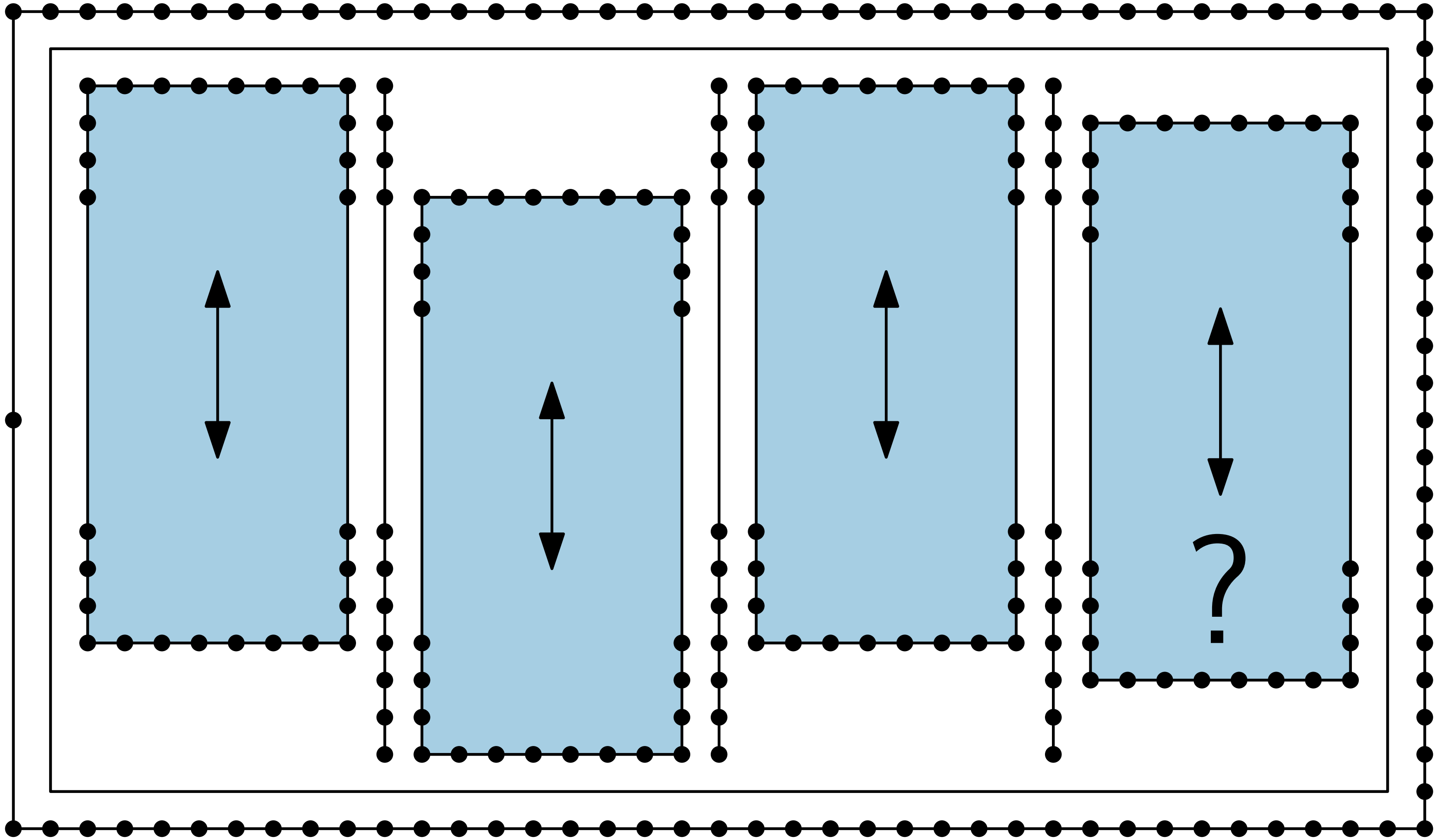
Boundary, Belt, and “Piston” Gadget



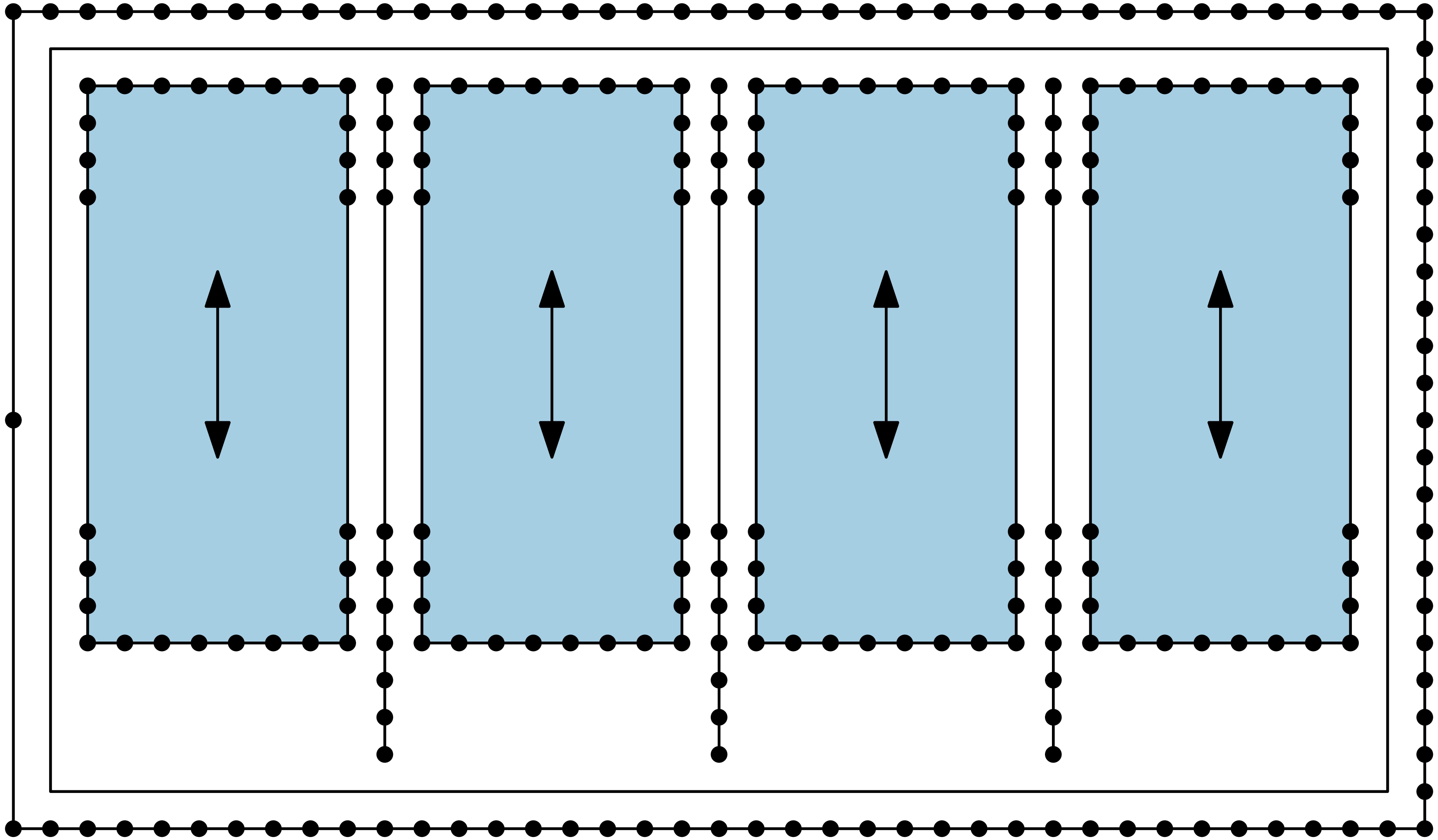
Boundary, Belt, and "Piston" Gadget



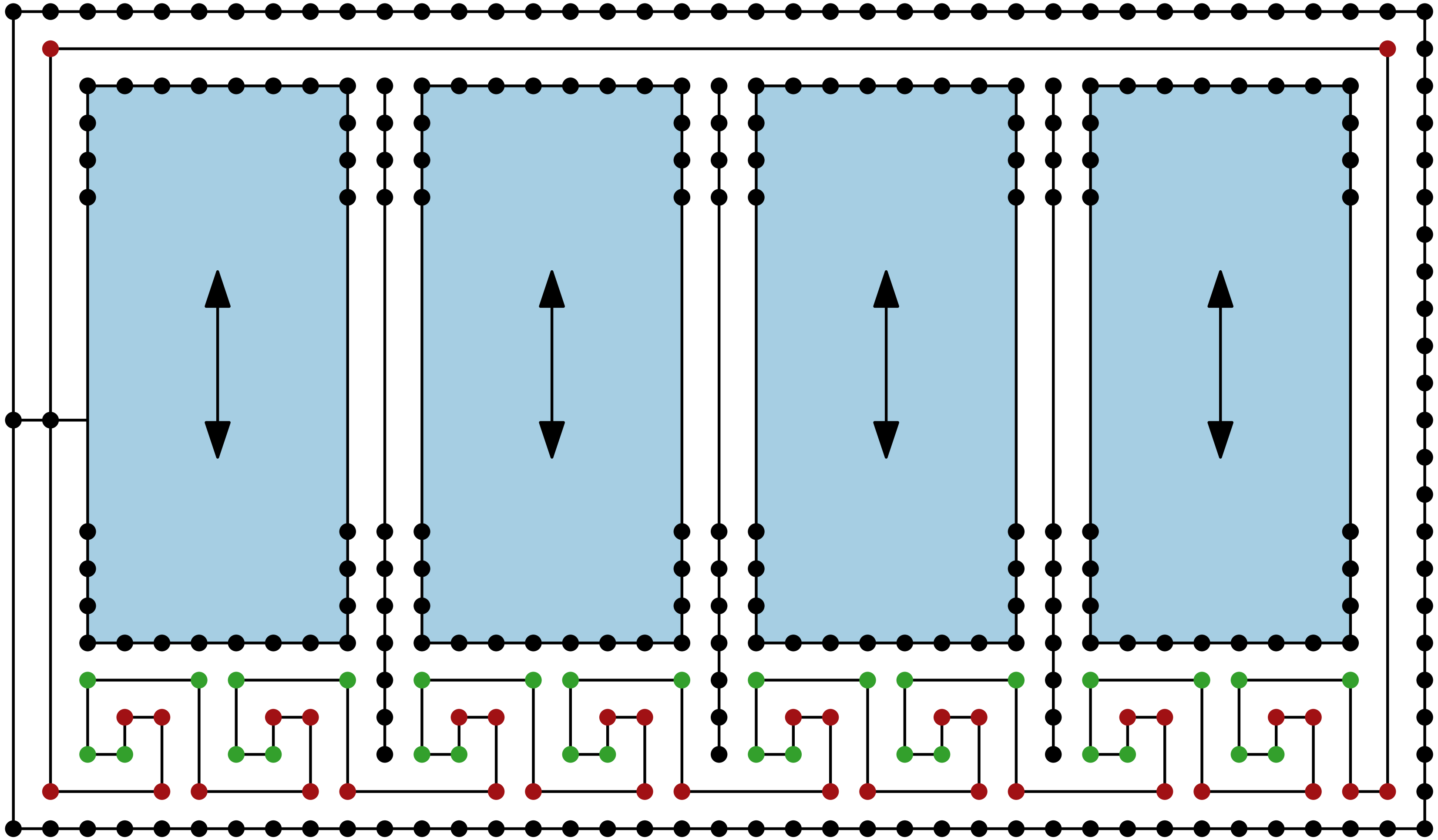
Boundary, Belt, and “Piston” Gadget



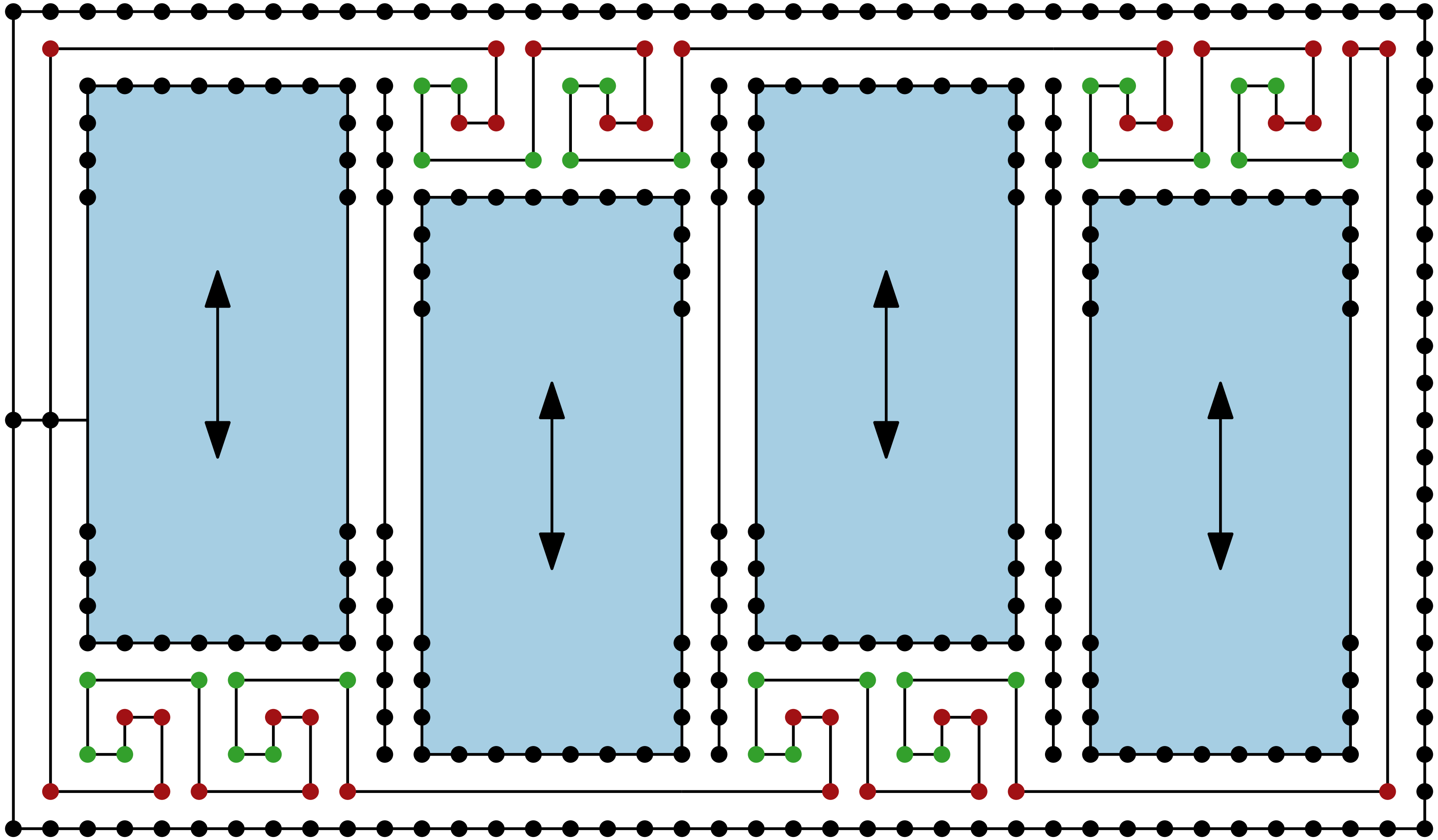
Boundary, Belt, and “Piston” Gadget



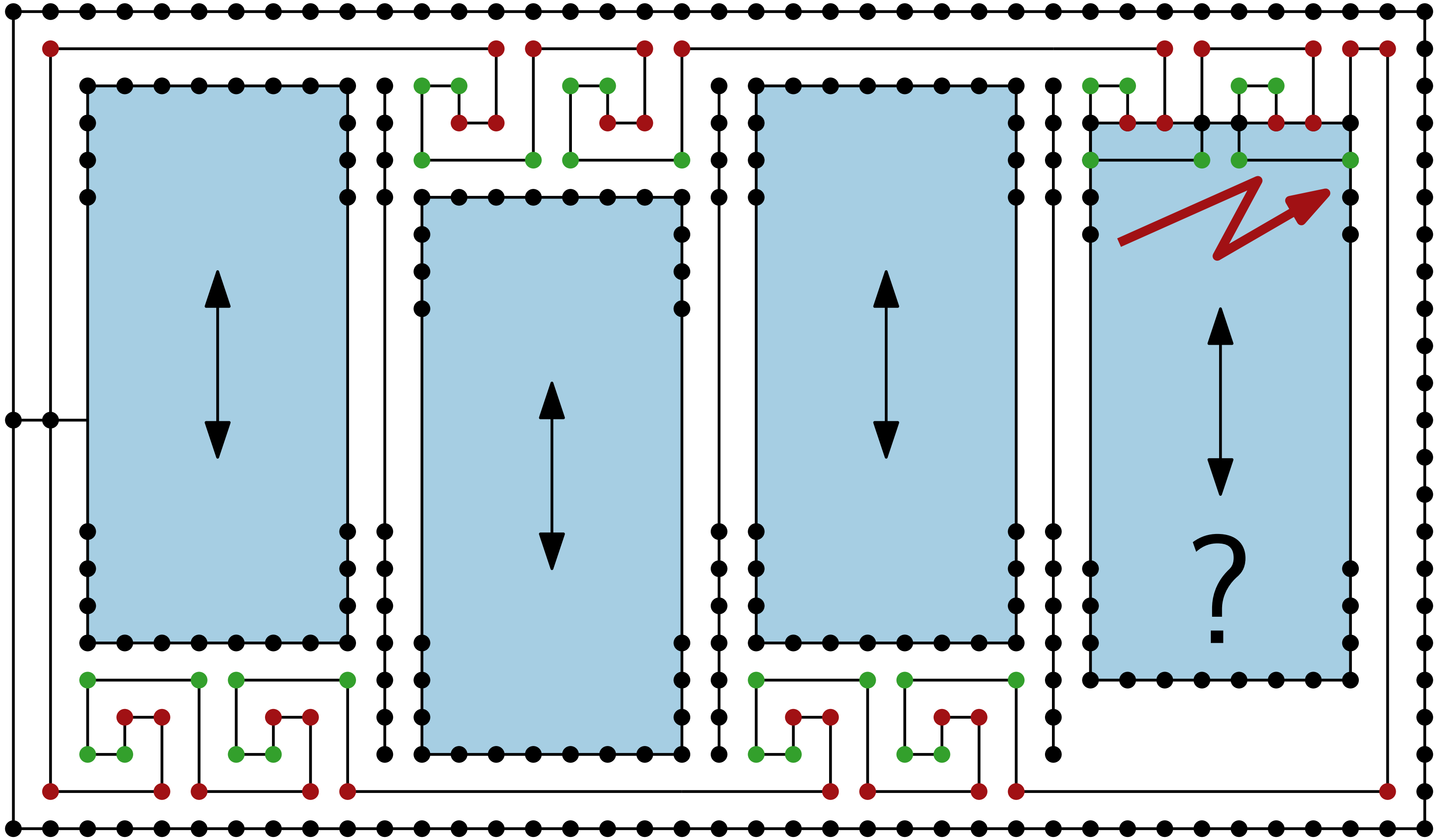
Boundary, Belt, and “Piston” Gadget



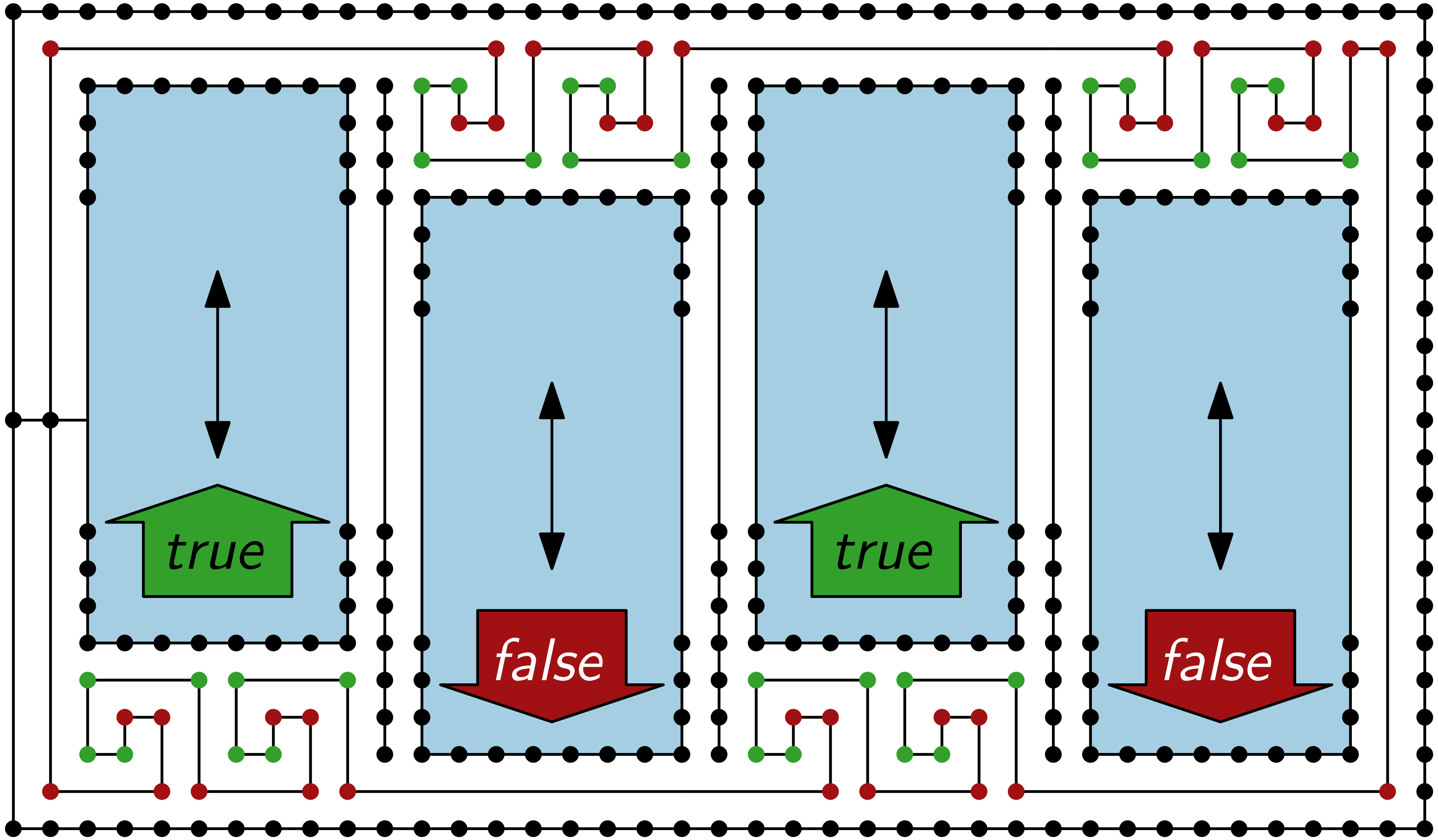
Boundary, Belt, and “Piston” Gadget



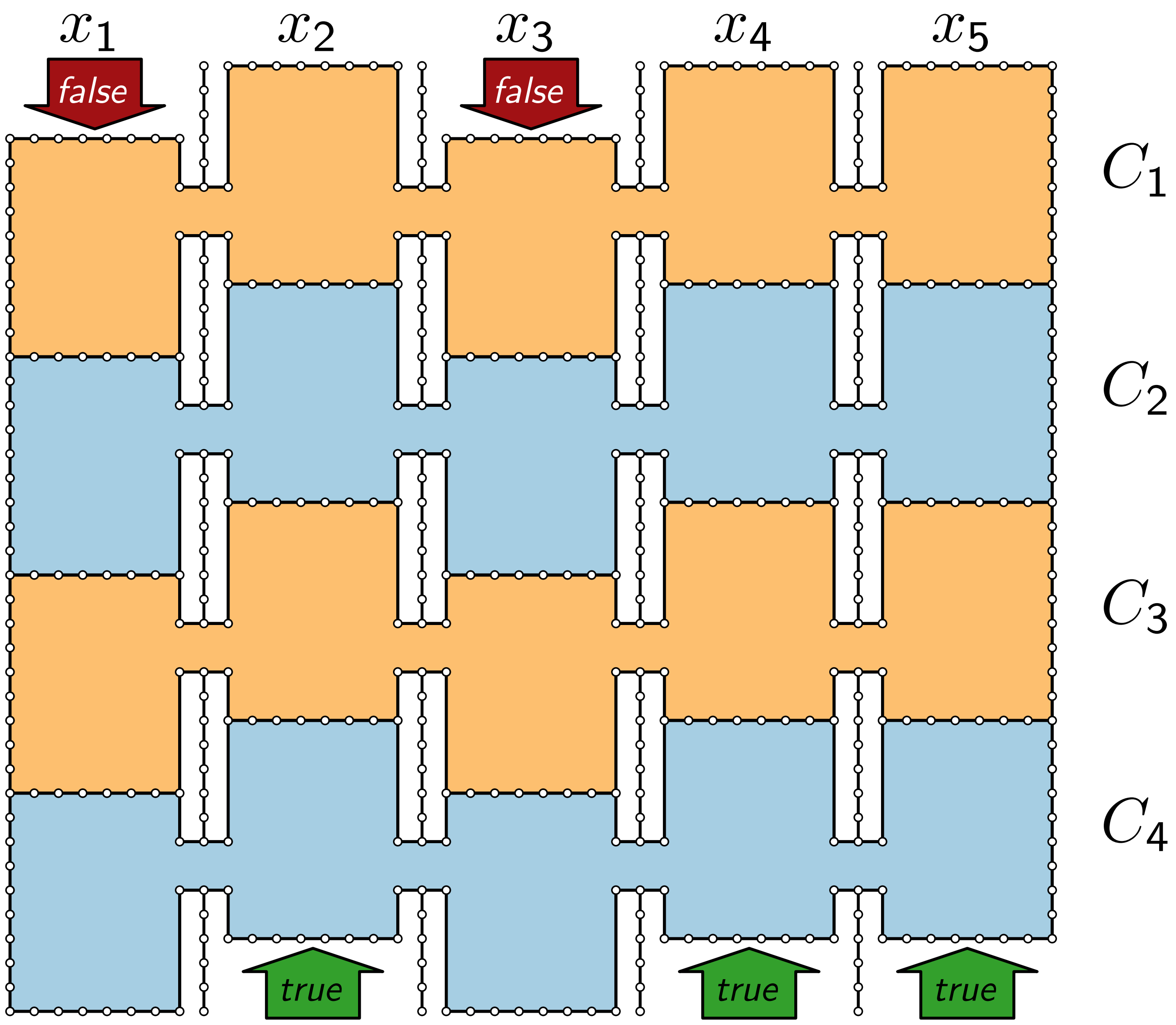
Boundary, Belt, and "Piston" Gadget



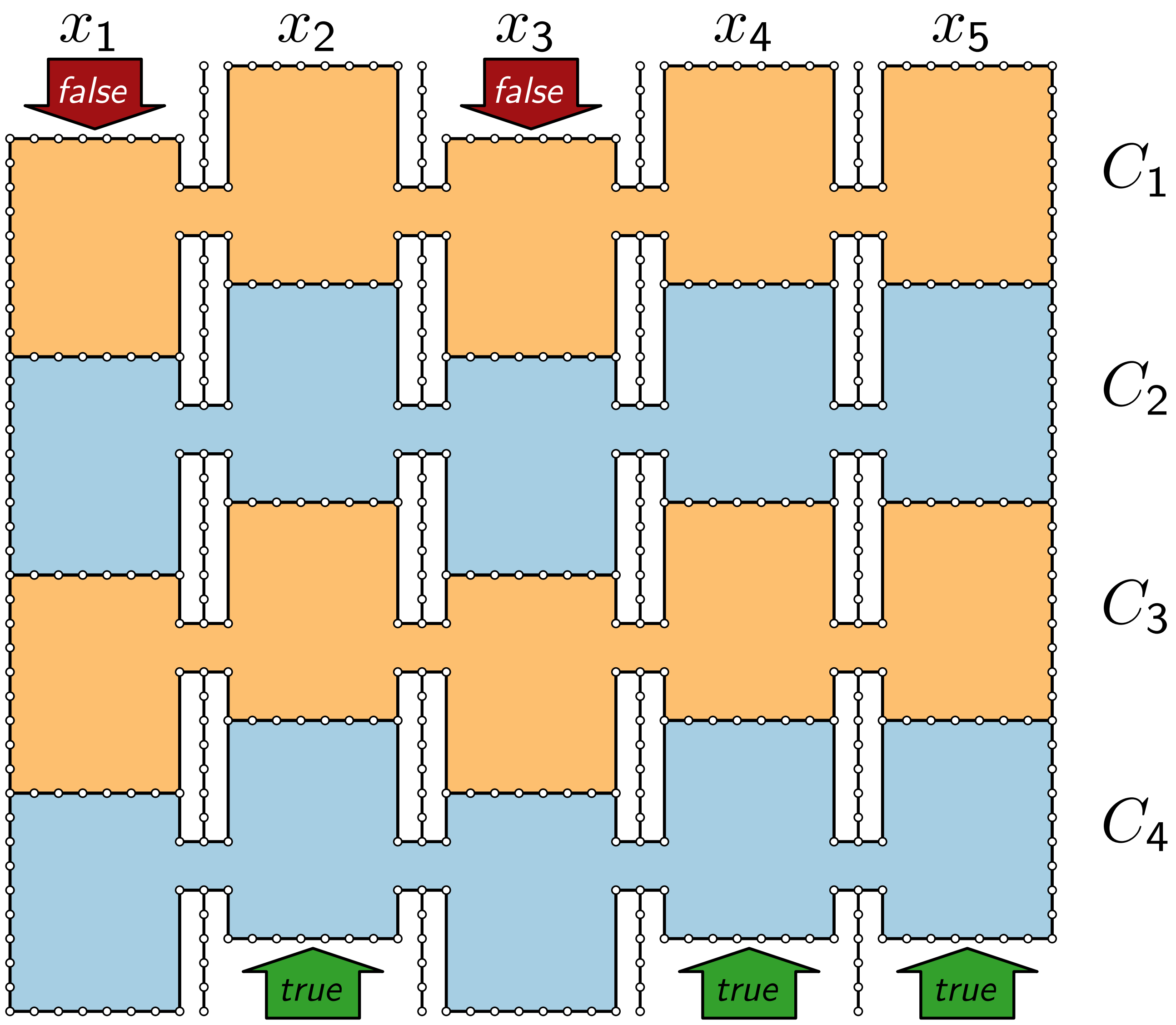
Boundary, Belt, and "Piston" Gadget



Clause Gadgets



Clause Gadgets



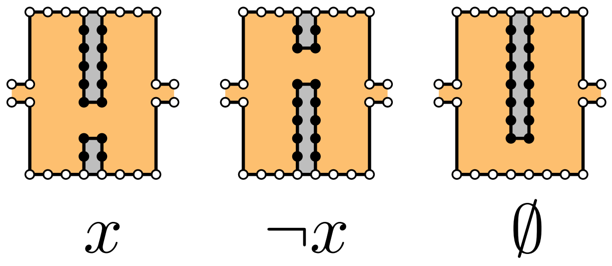
Example:

$C_1 = x_2 \vee \neg x_4$

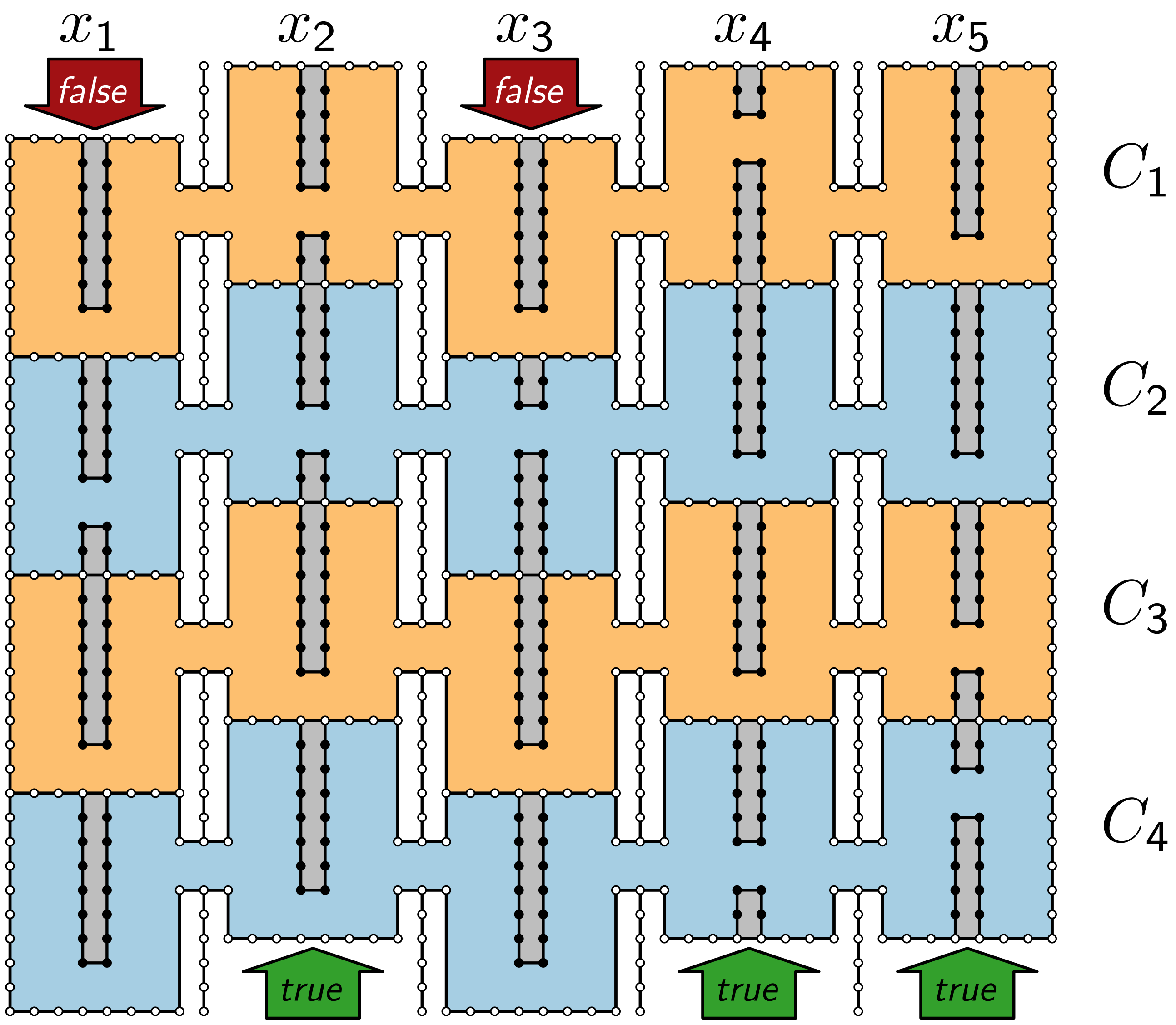
$C_2 = x_1 \vee x_2 \vee \neg x_3$

$C_3 = x_5$

$C_4 = x_4 \vee \neg x_5$



Clause Gadgets



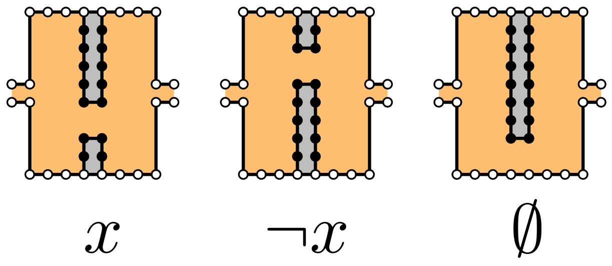
Example:

$C_1 = x_2 \vee \neg x_4$

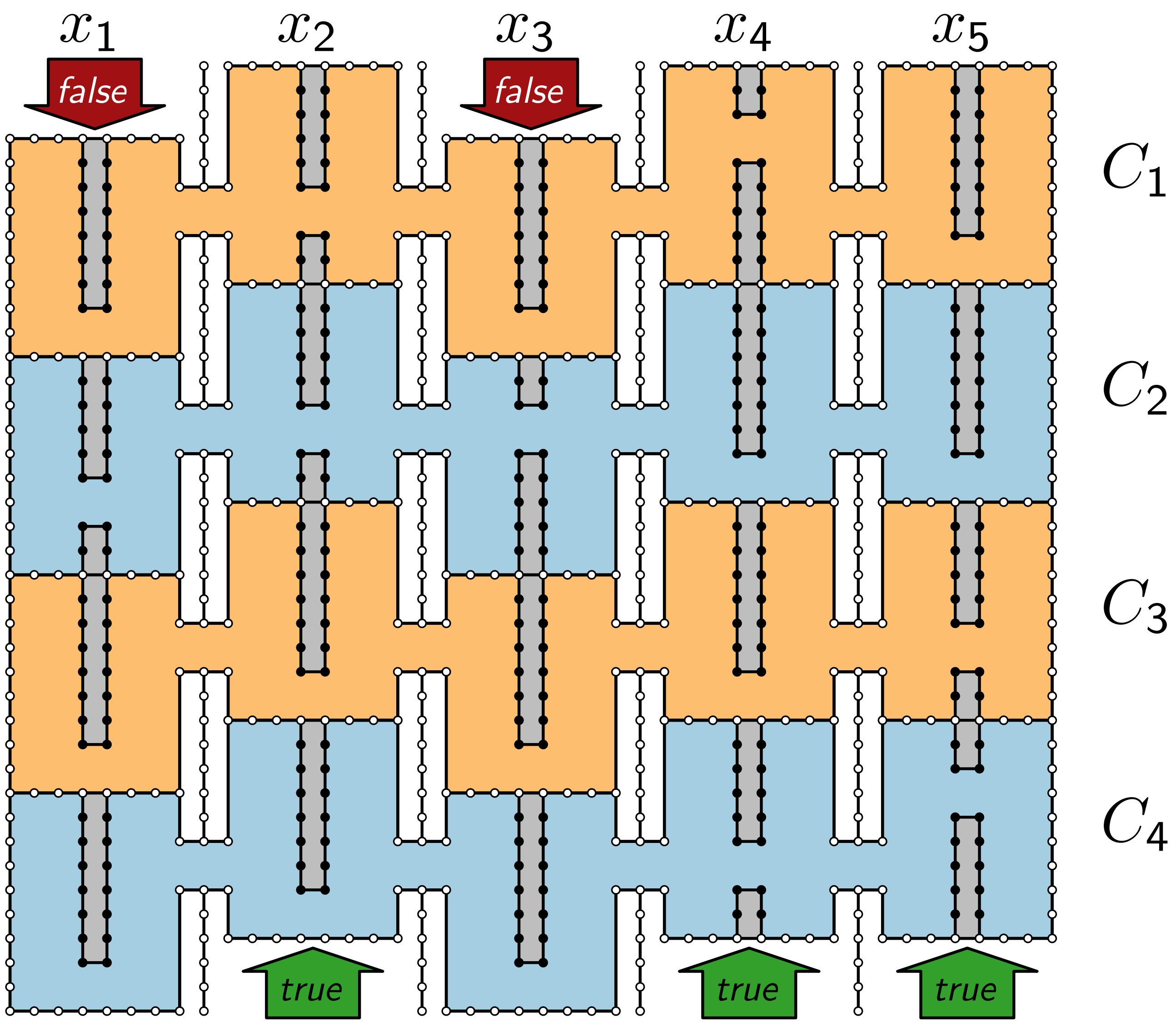
$C_2 = x_1 \vee x_2 \vee \neg x_3$

$C_3 = x_5$

$C_4 = x_4 \vee \neg x_5$



Clause Gadgets



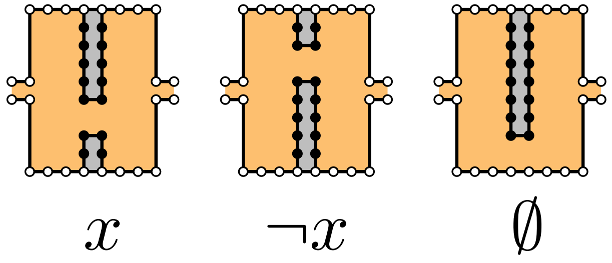
Example:

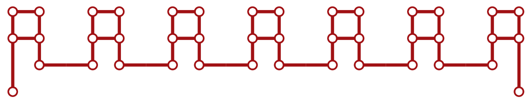
$C_1 = x_2 \vee \neg x_4$

$C_2 = x_1 \vee x_2 \vee \neg x_3$

$C_3 = x_5$

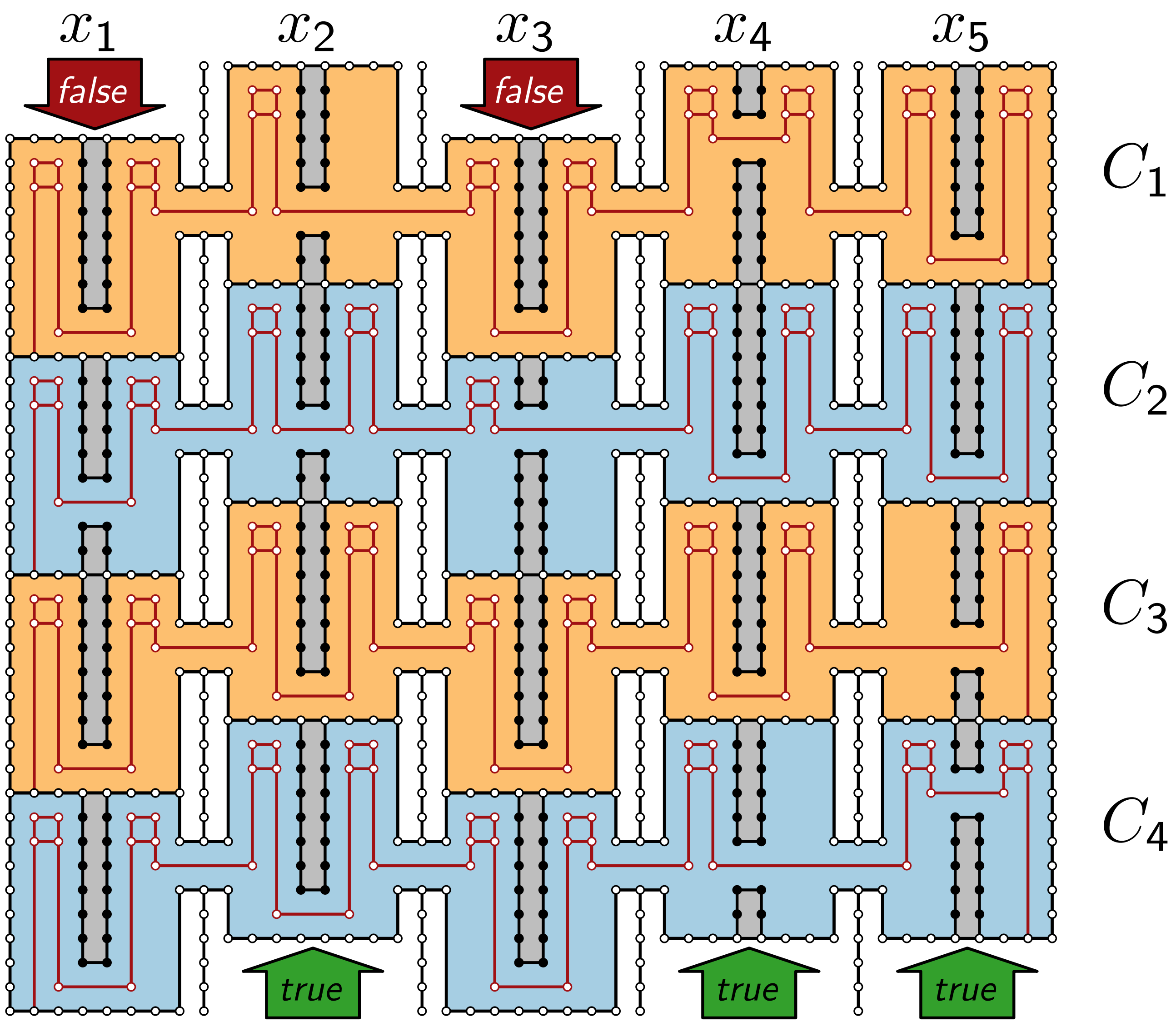
$C_4 = x_4 \vee \neg x_5$





insert $(2n-1)$-chain through each clause

Clause Gadgets



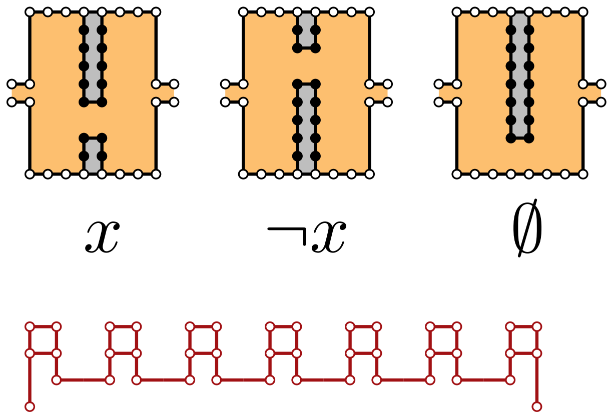
Example:

$C_1 = x_2 \vee \neg x_4$

$C_2 = x_1 \vee x_2 \vee \neg x_3$

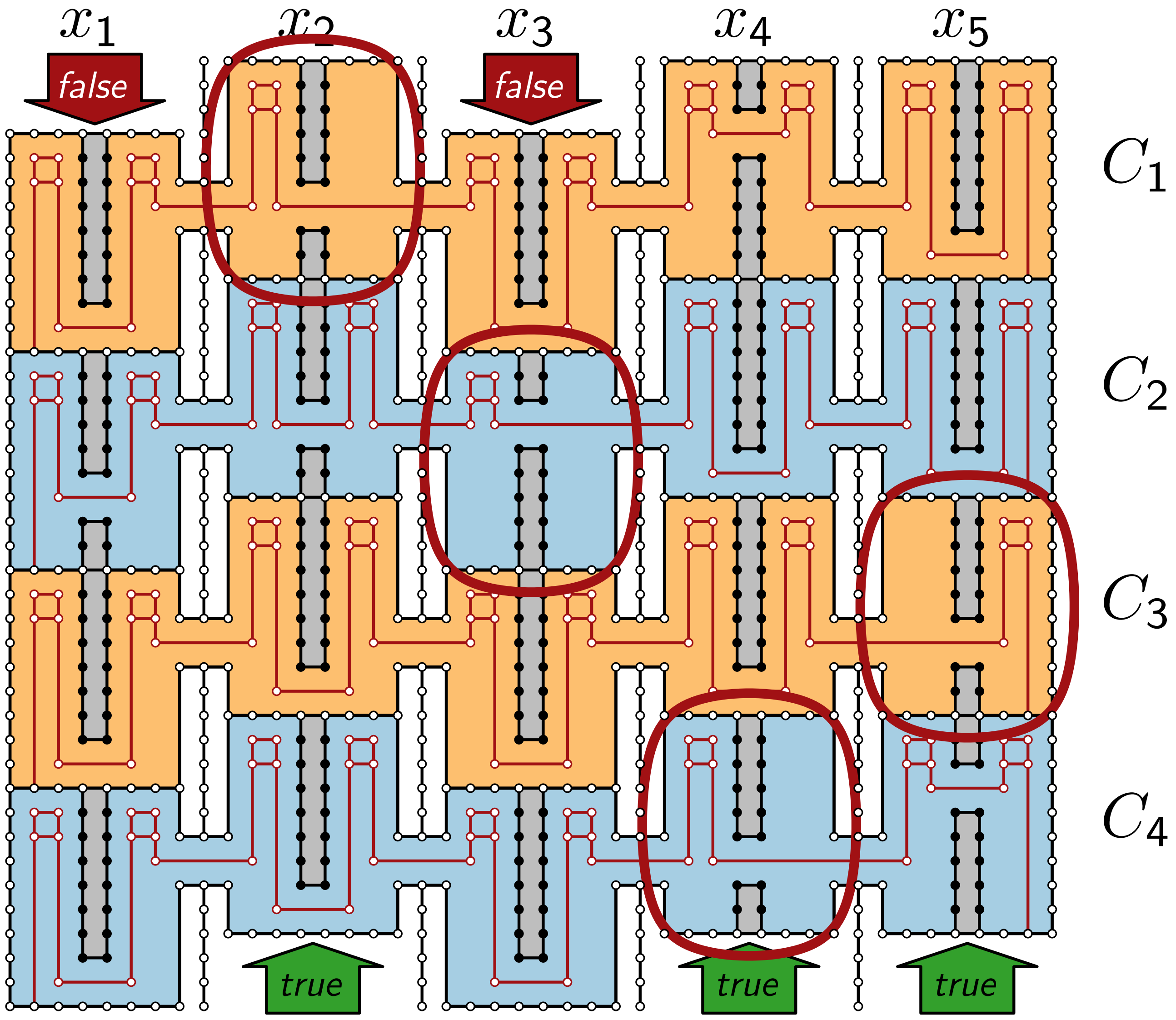
$C_3 = x_5$

$C_4 = x_4 \vee \neg x_5$



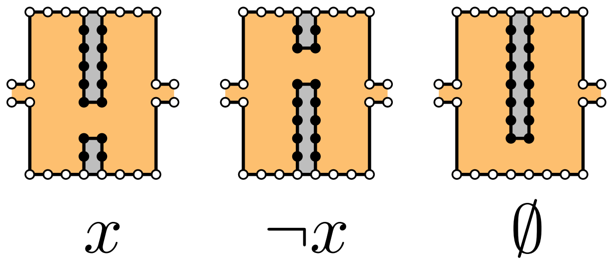
insert $(2n-1)$-chain through each clause

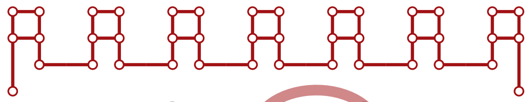
Clause Gadgets



Example:

$C_1 = x_2 \vee \neg x_4$
$C_2 = x_1 \vee x_2 \vee \neg x_3$
$C_3 = x_5$
$C_4 = x_4 \vee \neg x_5$

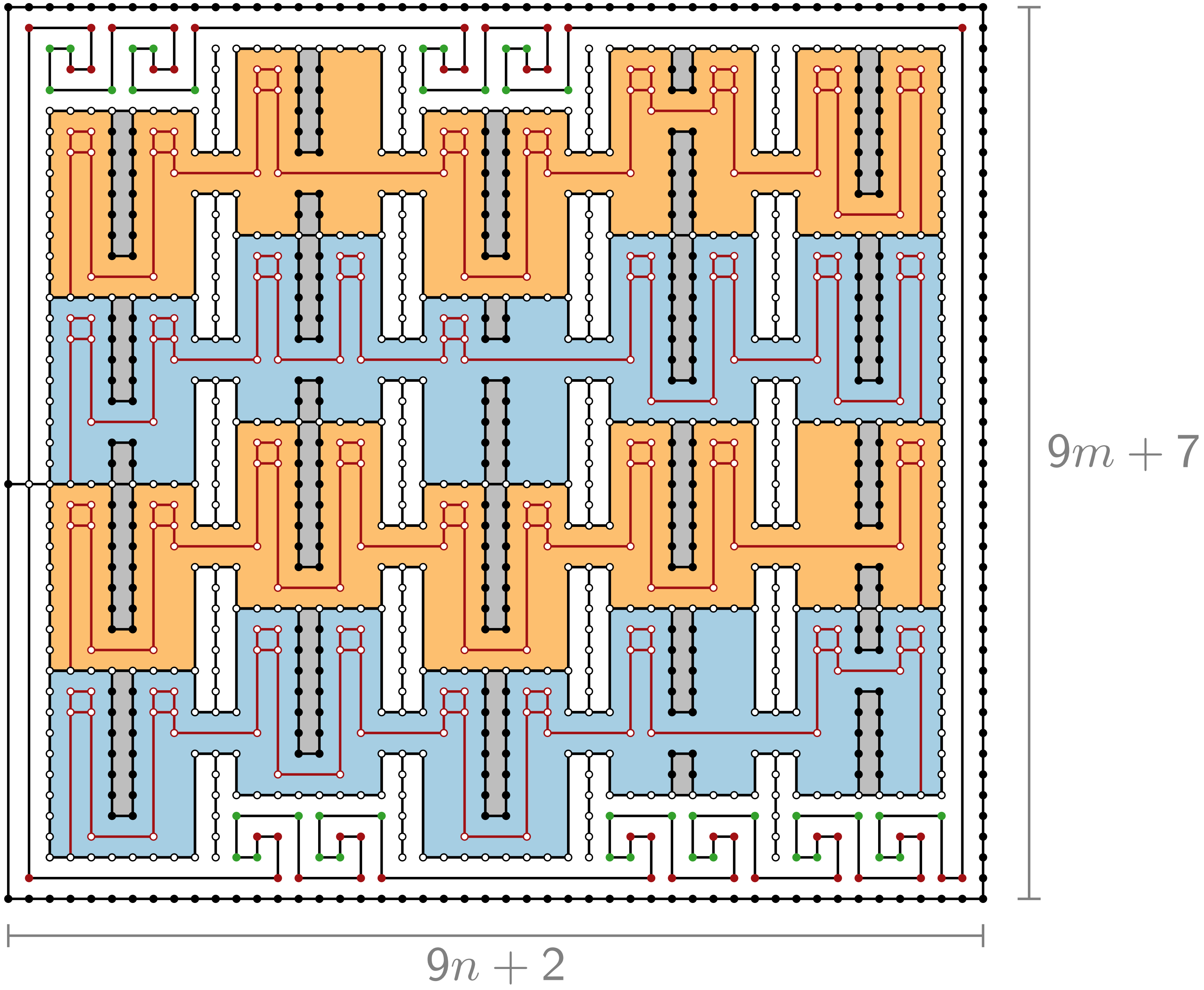




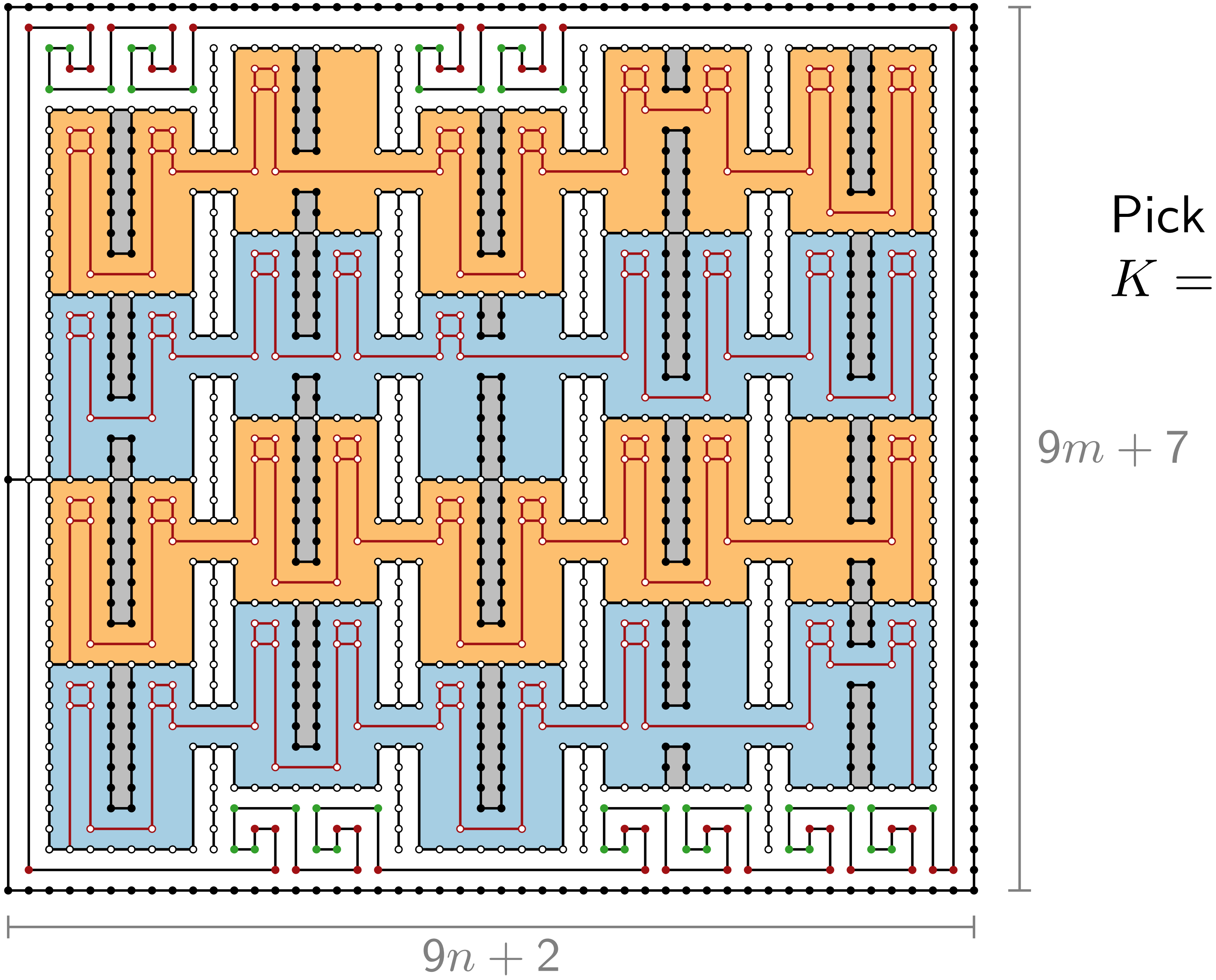
insert $(2n - 1)$-chain through each clause

→ for every clause, there needs to be ≥ 1 "gap of a literal" to be on the same height as the "tunnel" to the next literal

Complete Reduction



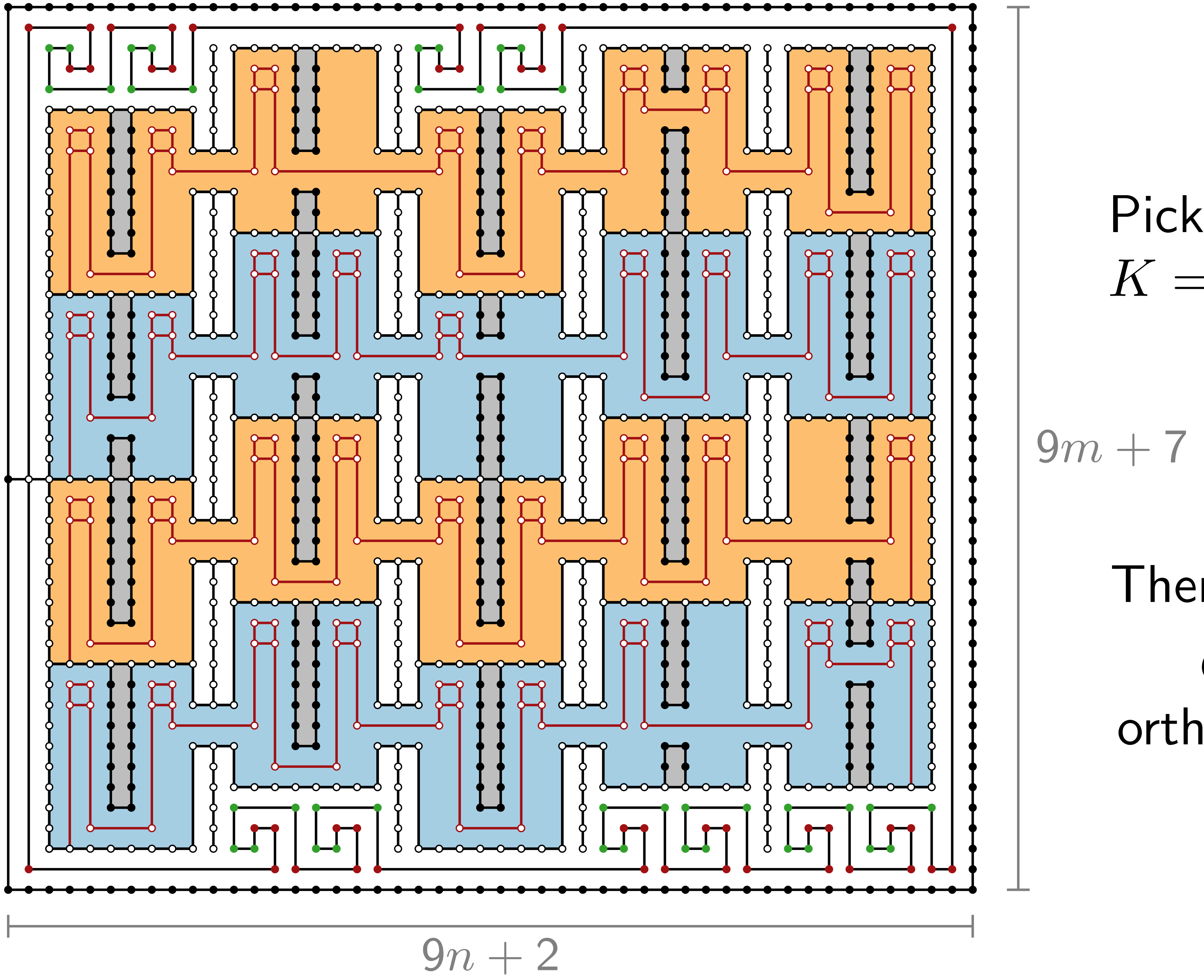
Complete Reduction



Pick

$K = (9n+2) \times (9m+7)$

Complete Reduction



Pick
$K = (9n+2) \times (9m+7)$

Then:

G under $H(G)$ has an orthogonal drawing in area K

$\Leftrightarrow$

Φ satisfiable

□

Literature

- [GD Ch. 5] for detailed explanation
- [Tamassia 1987] "On embedding a graph in the grid with the minmum number of bends"
  Original paper on flow for bend minimization.
- [van den Brand, Chen, Kyng, Liu, Peng, Probst, Sachdeva, Sidford 2023]
  "A Deterministic Almost-Linear Time Algorithm for Minimum-Cost Flow"
  State-of-the-art algorithm for solving the minimum-cost flow problem
  (published recently in the proceedings of the FOCS 2023 conference).
- [Patrignani 2001] "On the complexity of orthogonal compaction"
  NP-hardness proof for orthogonal representation of planar max-degree-4 graphs.
- [Evans, Fleszar, Kindermann, Saeedi, Shin, Wolff 2022]
  "Minimum rectilinear polygons for given angle sequences"
  NP-hardness proof for compaction of cycles.