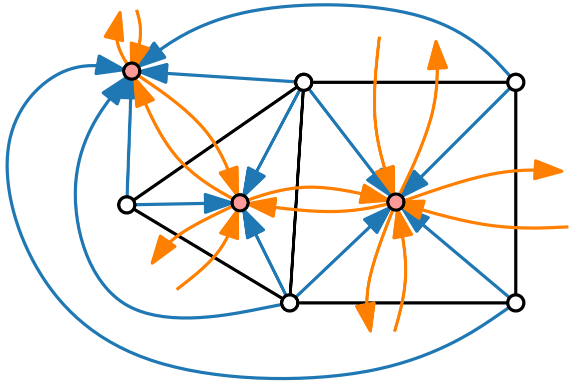
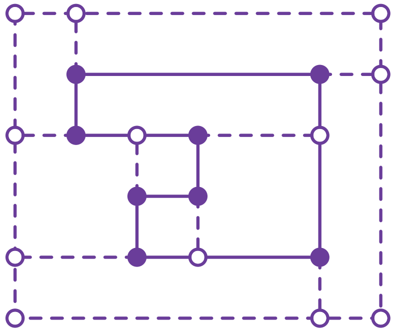
Universität Trier

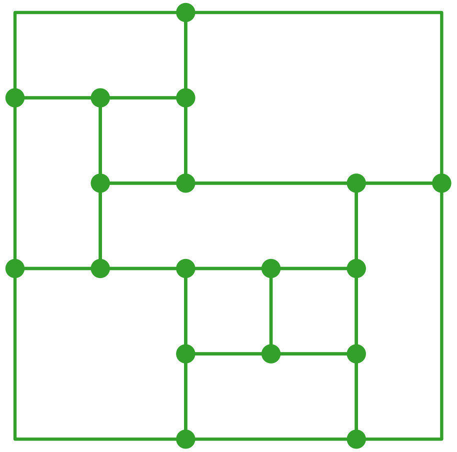
Visualization of Graphs





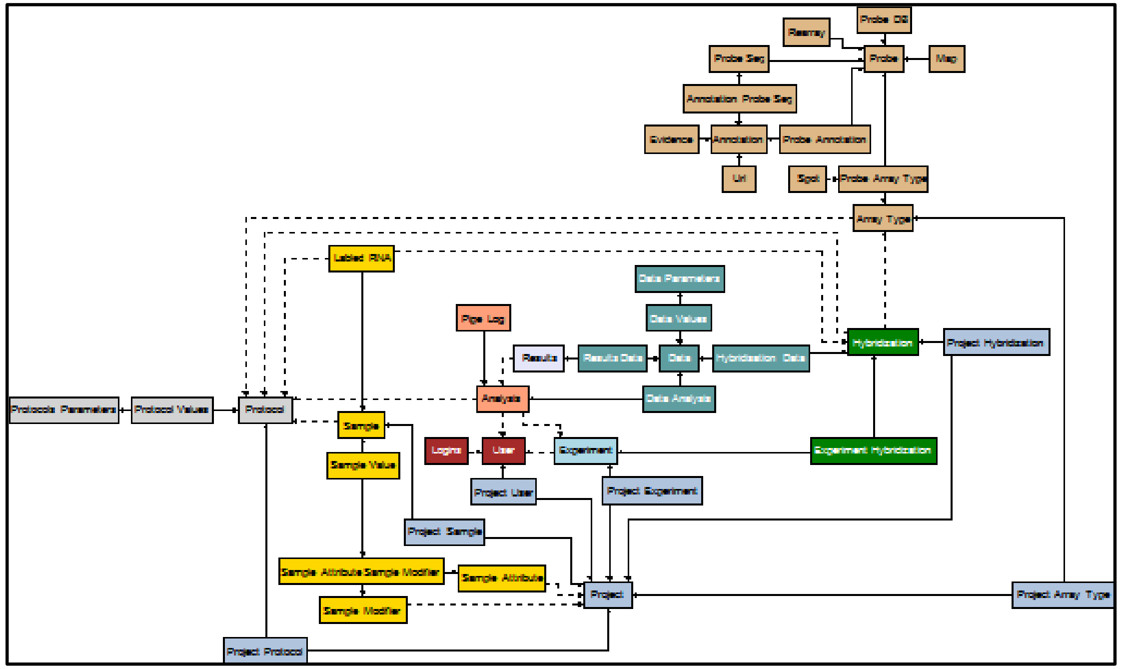
Lecture 6:
Orthogonal Layouts

Part I:
Topology – Shape – Metrics



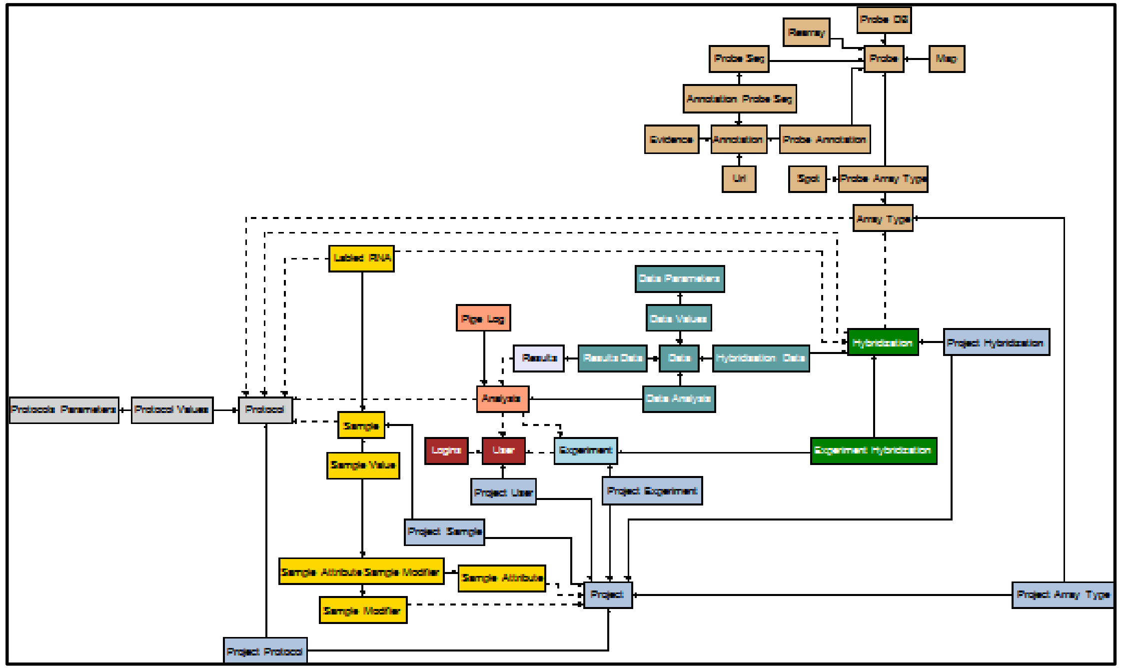
Philipp Kindermann

Orthogonal Layout – Applications

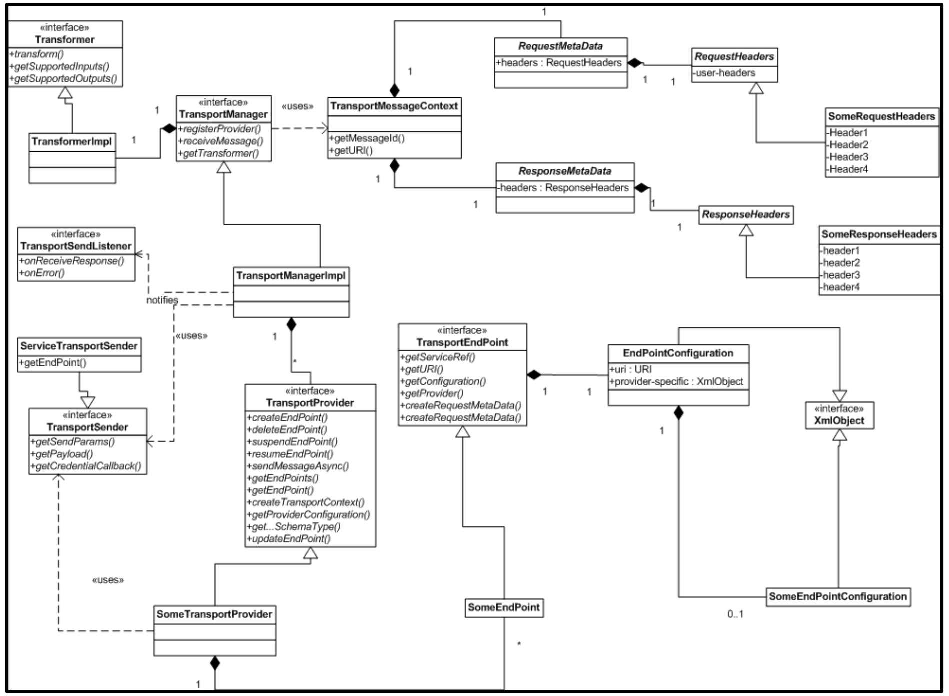


ER diagram in OGDF

Orthogonal Layout – Applications

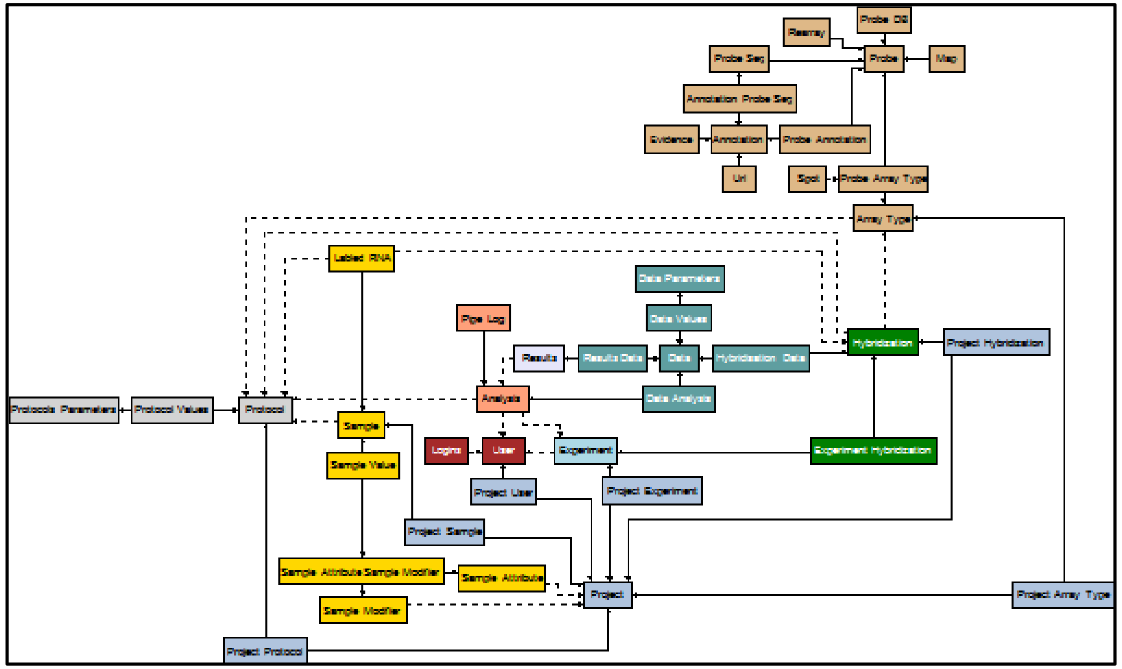


ER diagram in OGDF

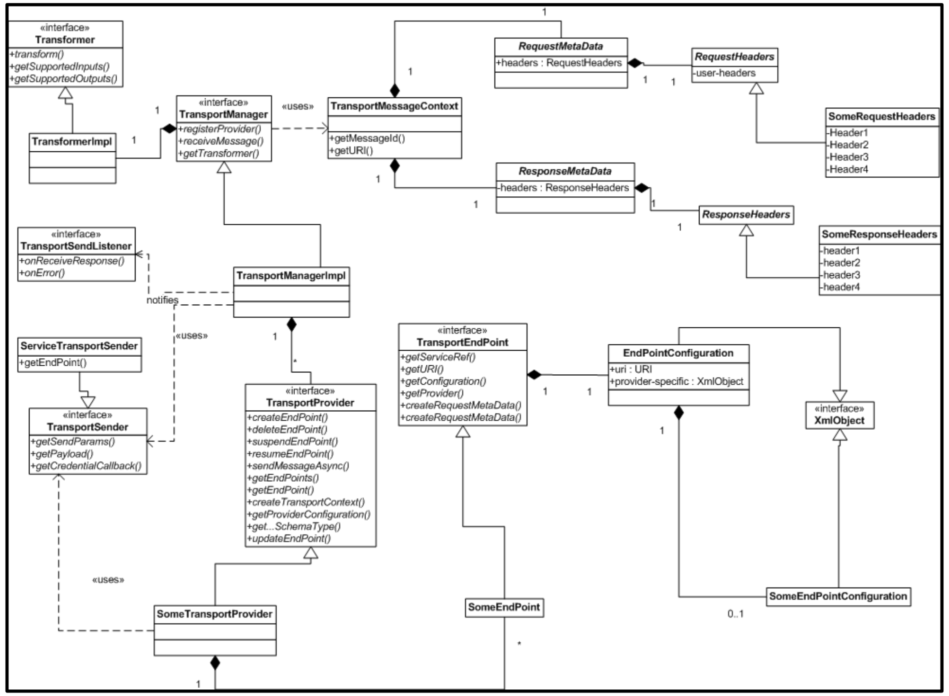


UML diagram by Oracle

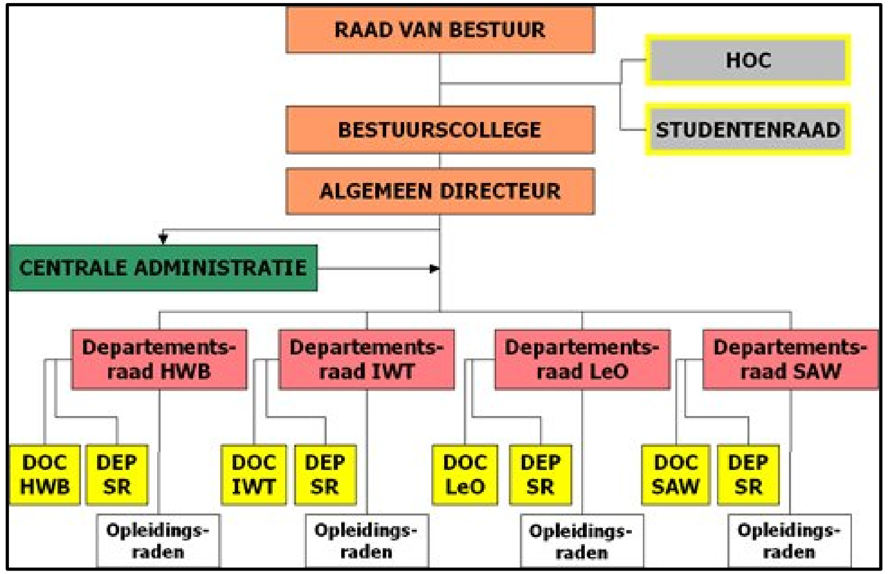
Orthogonal Layout – Applications



ER diagram in OGDF

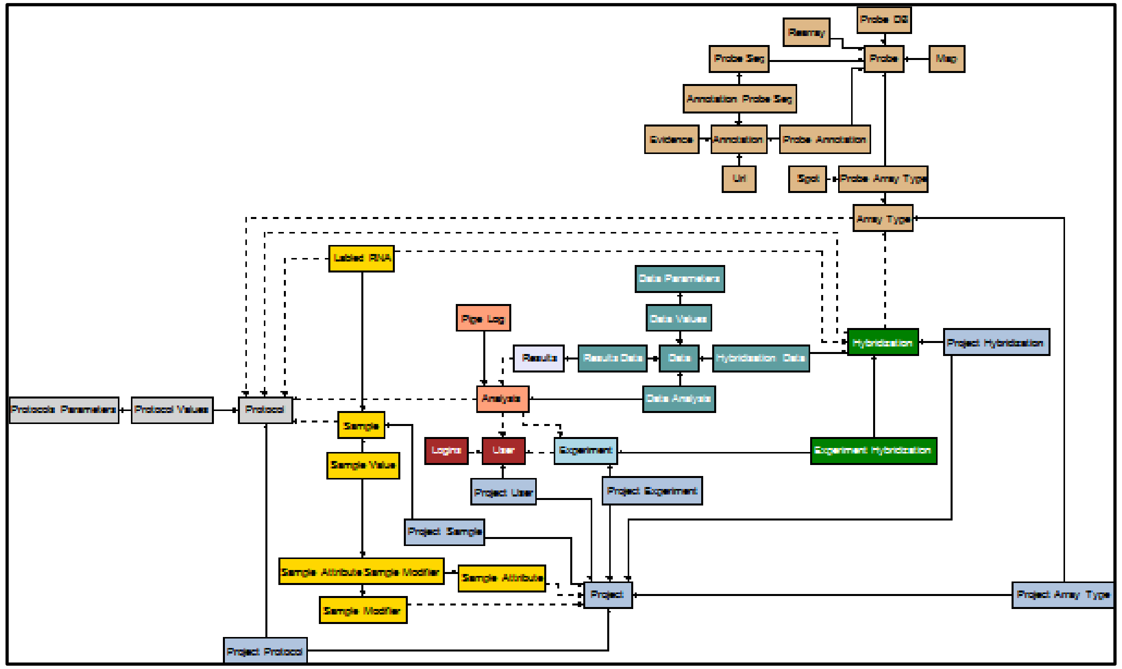


UML diagram by Oracle

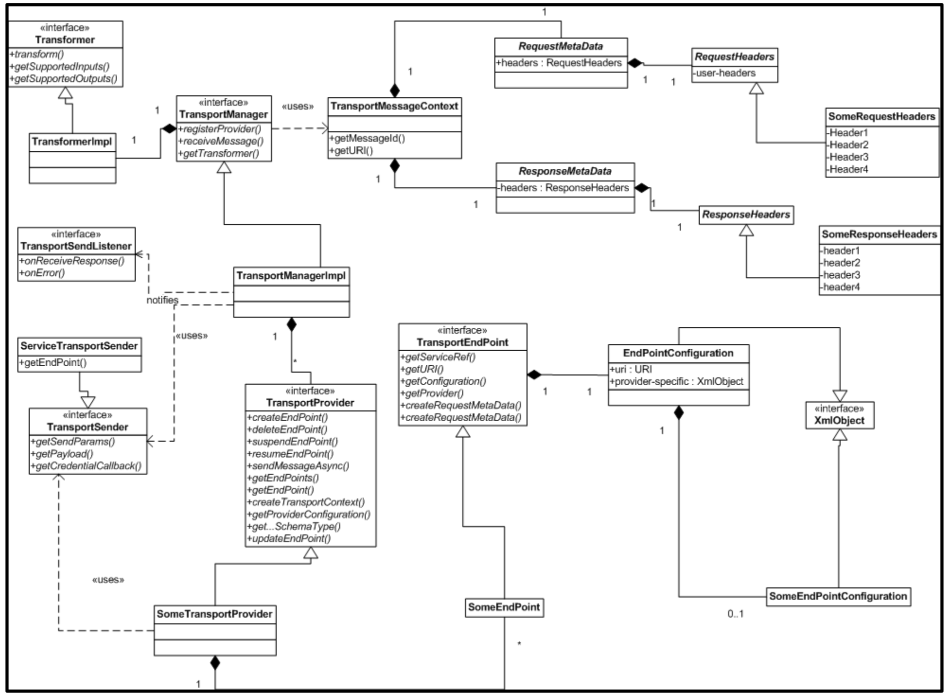


Organigram of HS Limburg

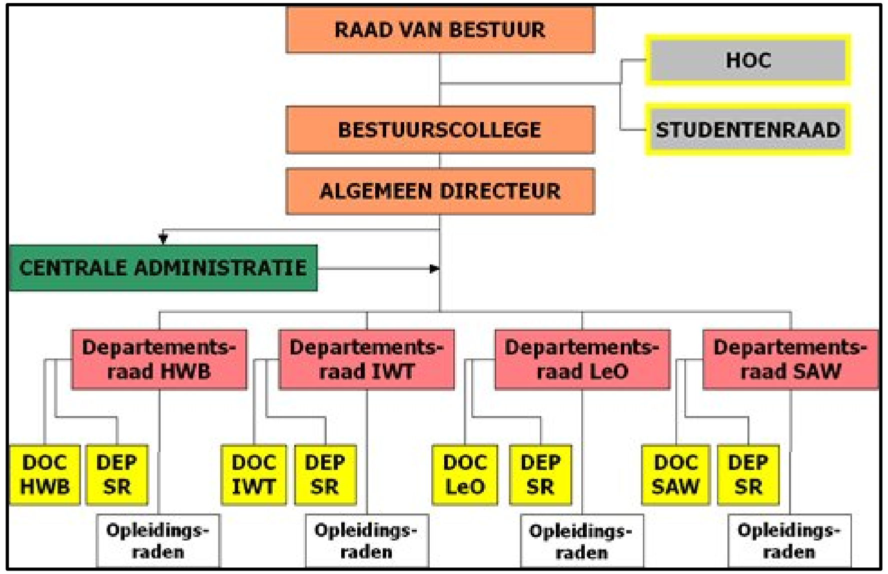
Orthogonal Layout – Applications



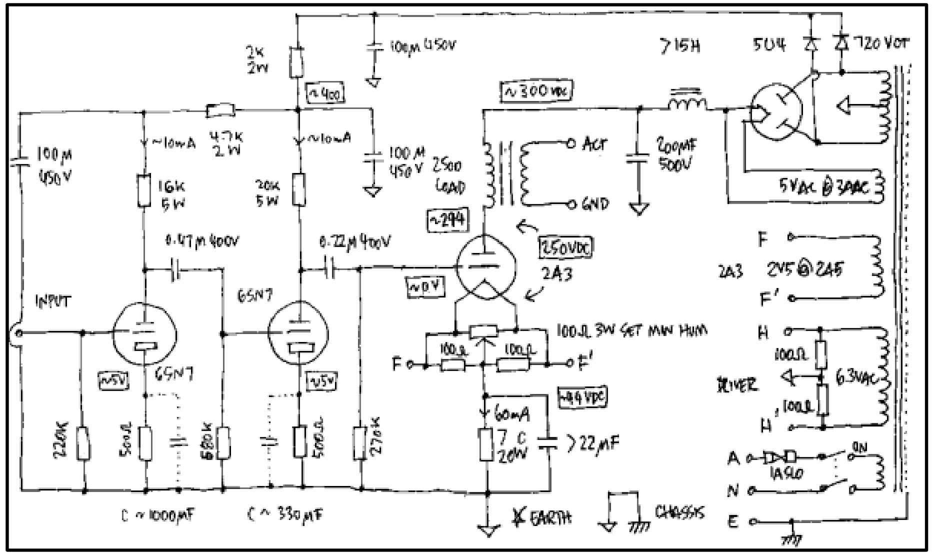
ER diagram in OGDF



UML diagram by Oracle



Organigram of HS Limburg

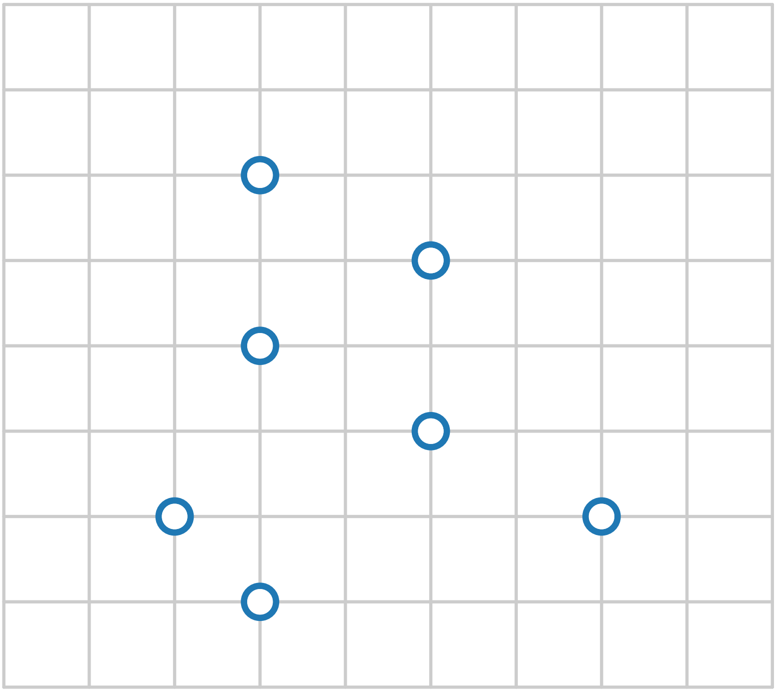


Circuit diagram by Jeff Atwood

Orthogonal Layout – Definition

Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if

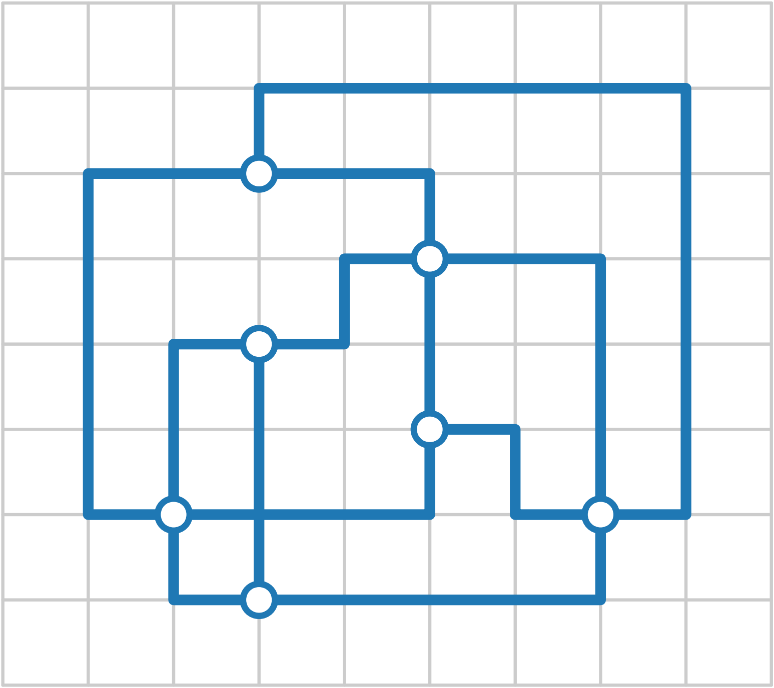
Orthogonal Layout – Definition



Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if

- vertices are drawn as points on a grid,

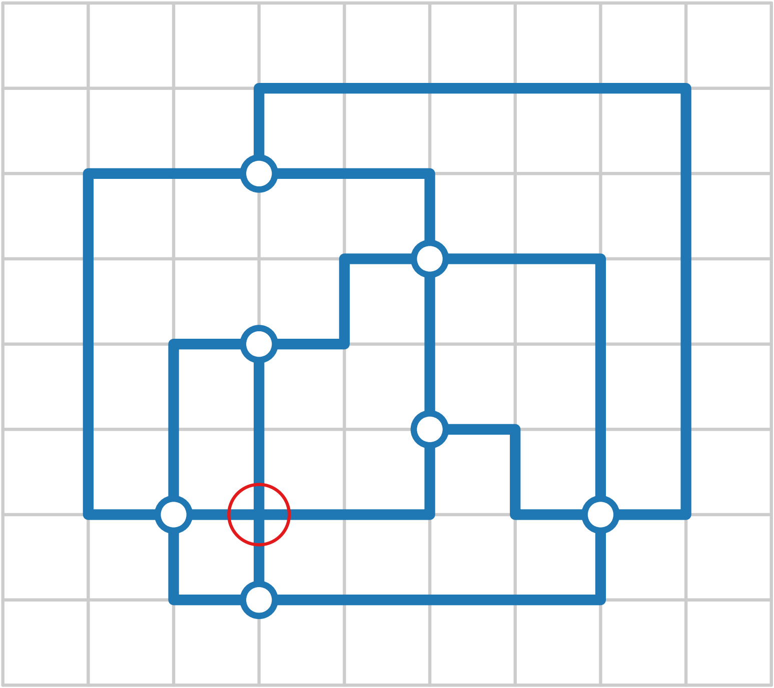
Orthogonal Layout – Definition



Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if

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

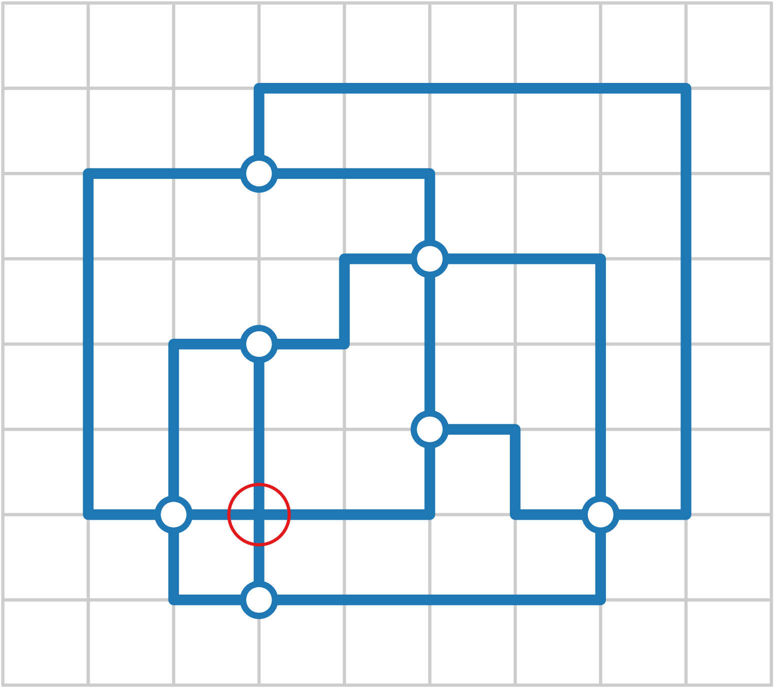
Orthogonal Layout – Definition



Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if

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

Orthogonal Layout – Definition

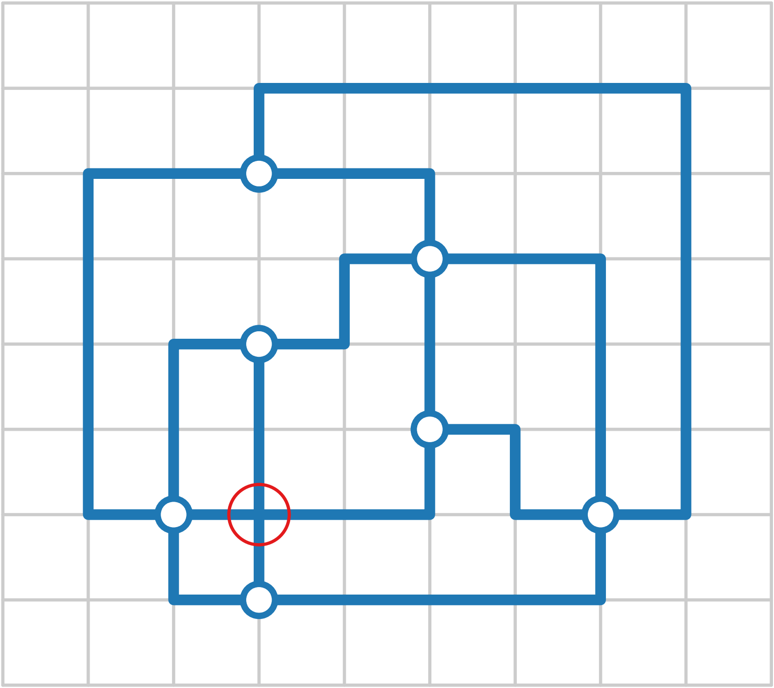


Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if

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

Observations.

Orthogonal Layout – Definition



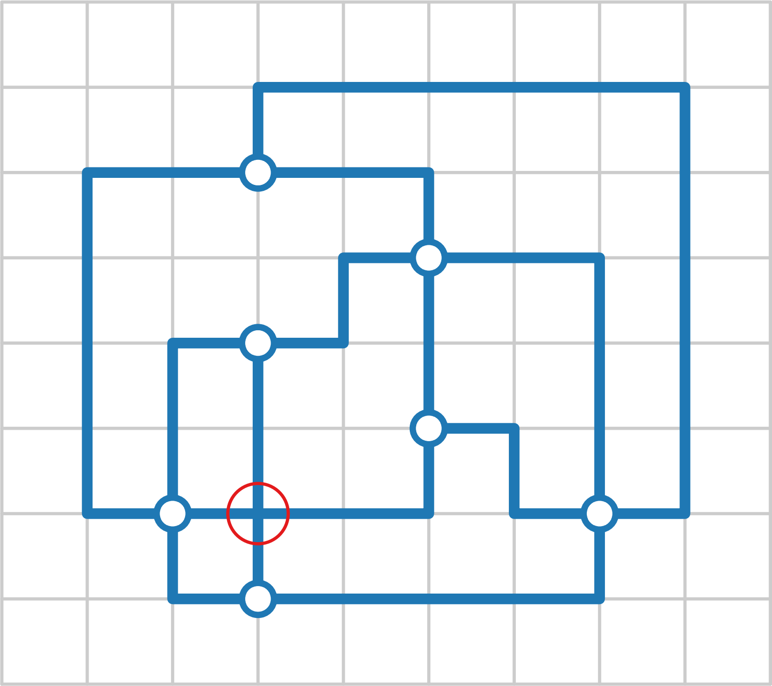
Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if

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

Observations.

- Edges lie on grid $\Rightarrow$ **bends** lie on grid points

Orthogonal Layout – Definition



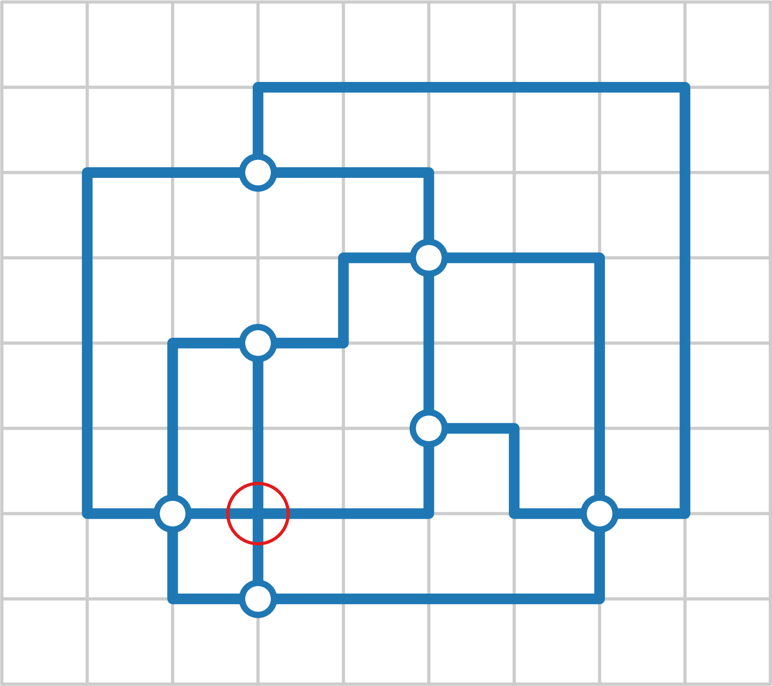
Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if

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

Observations.

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

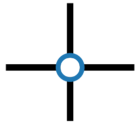
Orthogonal Layout – Definition



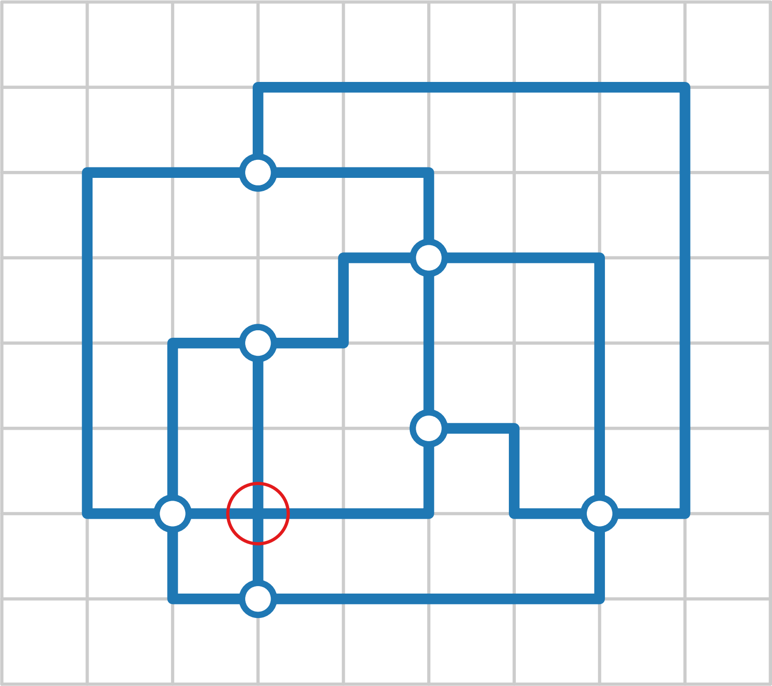
Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if

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

Observations.

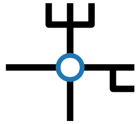
- Edges lie on grid ⇒ **bends** lie on grid points
- Max degree of each vertex is at most 4
- Otherwise 

Orthogonal Layout – Definition

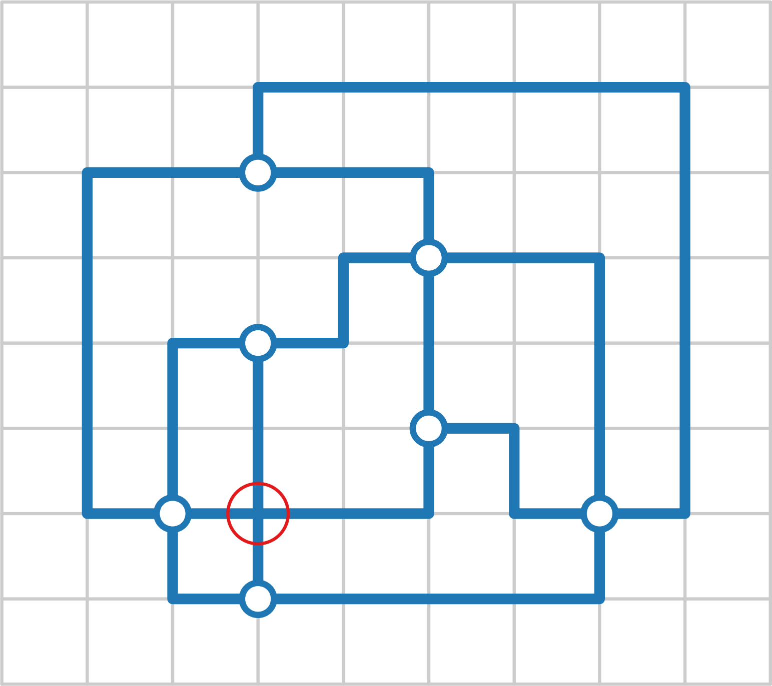


Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical segments, and
- pairs of edges are disjoint or cross orthogonally.

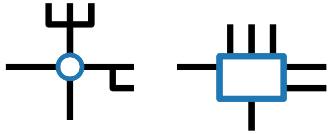
Observations.

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

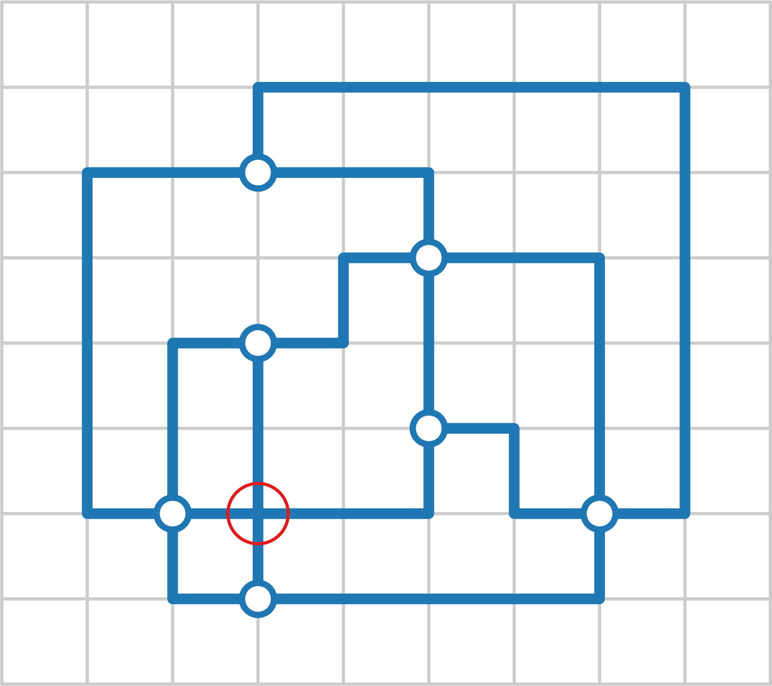
Orthogonal Layout – Definition



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

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

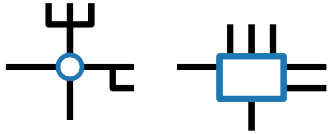
Orthogonal Layout – Definition



Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if

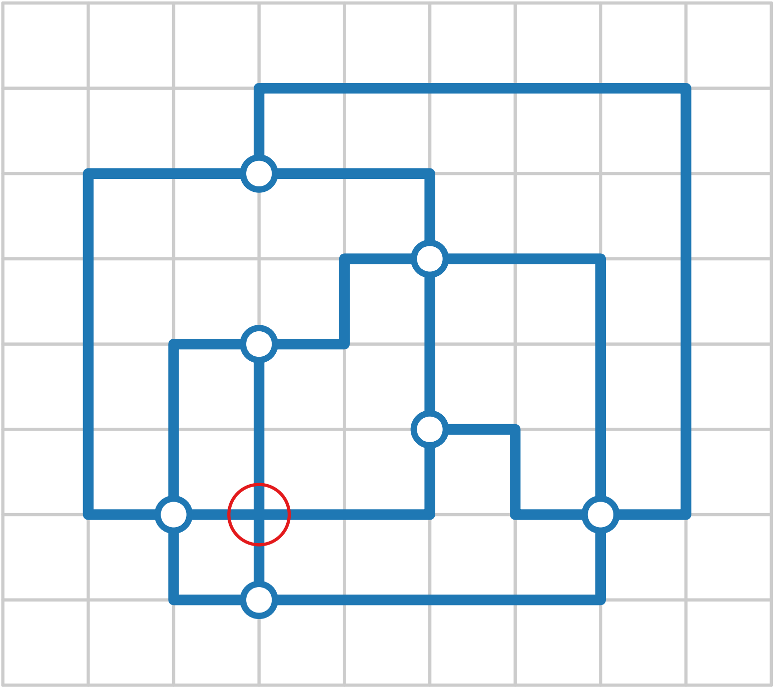
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical segments, and
- pairs of edges are disjoint or cross orthogonally.

Observations.

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

Planarization.

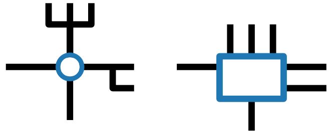
Orthogonal Layout – Definition



Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if

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

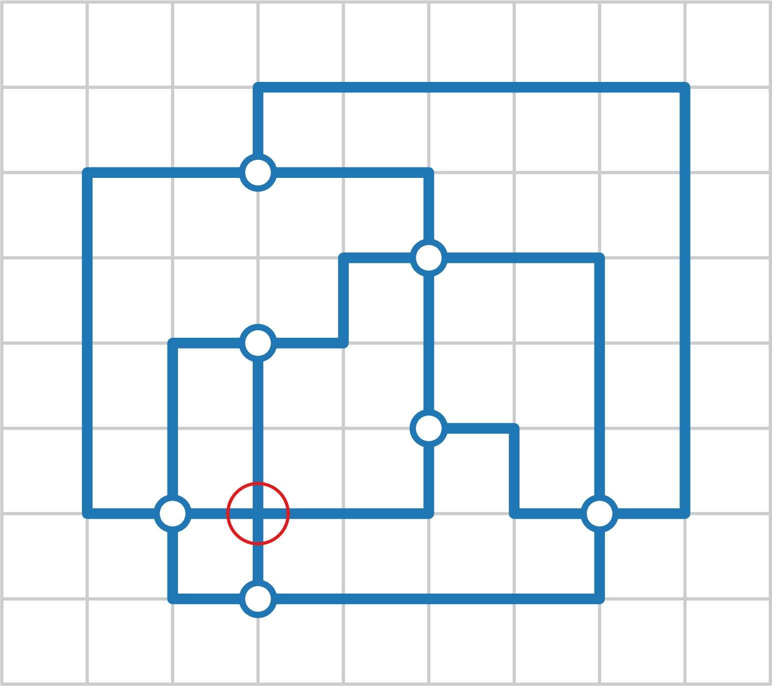
Observations.

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

Planarization.

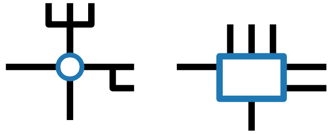
- Fix embedding

Orthogonal Layout – Definition



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

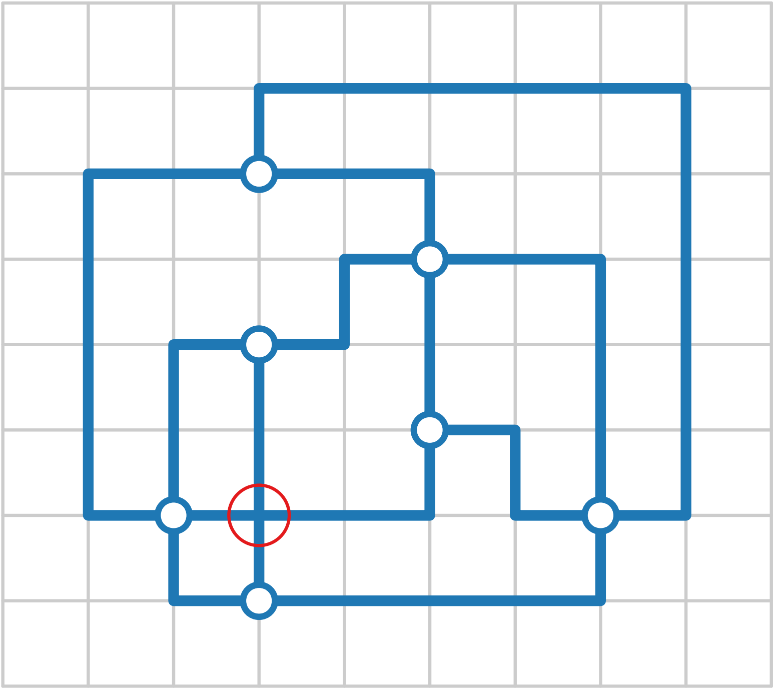
Observations.

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

Planarization.

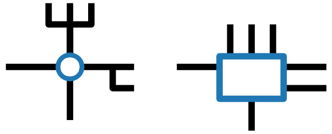
- Fix embedding
- Crossings become vertices

Orthogonal Layout – Definition

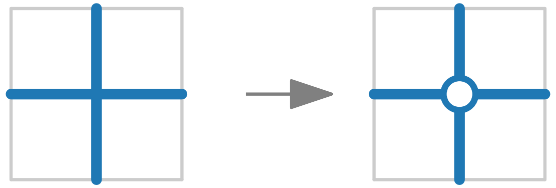


Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical segments, and
- pairs of edges are disjoint or cross orthogonally.

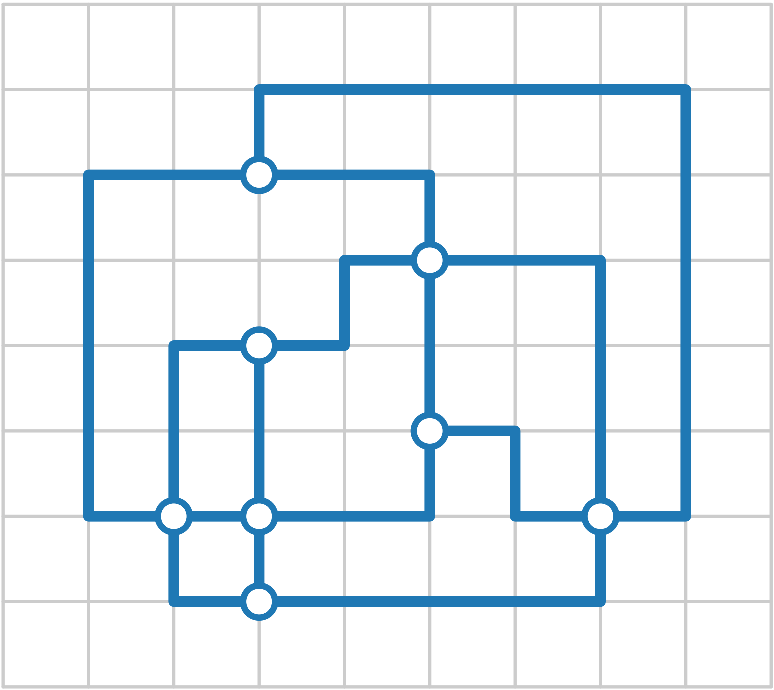
Observations.
- Edges lie on grid $\Rightarrow$ **bends** lie on grid points
- Max degree of each vertex is at most 4
- Otherwise



Planarization.
- Fix embedding
- Crossings become vertices



Orthogonal Layout – Definition

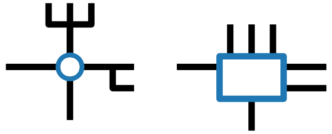


Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if

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

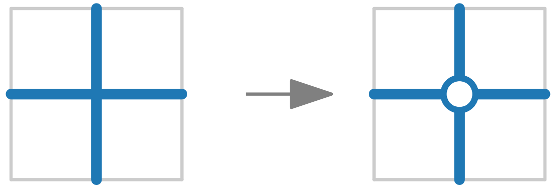
Observations.

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

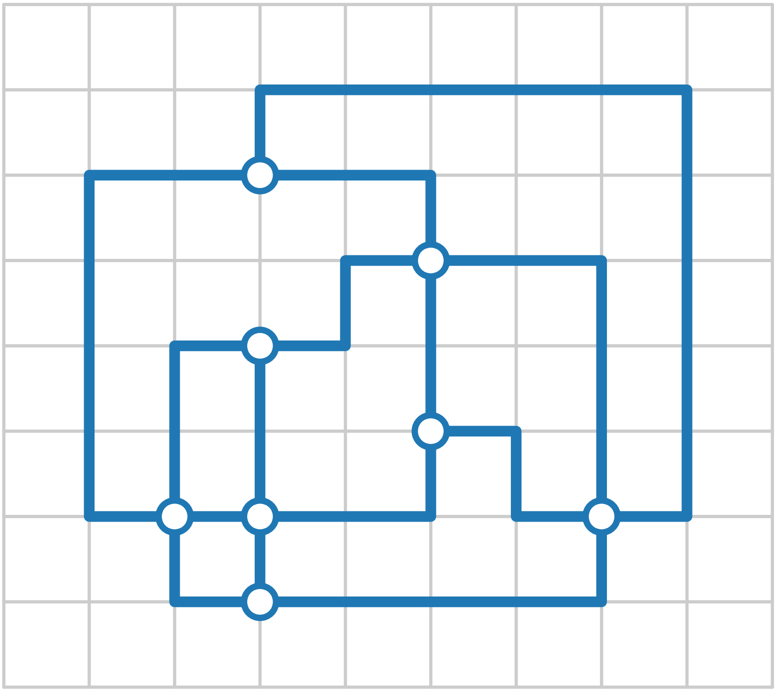


Planarization.

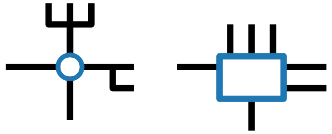
- Fix embedding
- Crossings become vertices



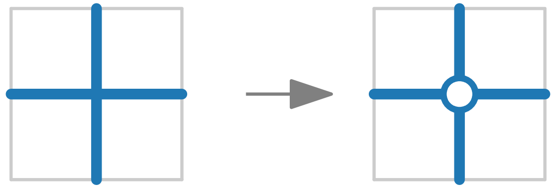
Orthogonal Layout – Definition



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

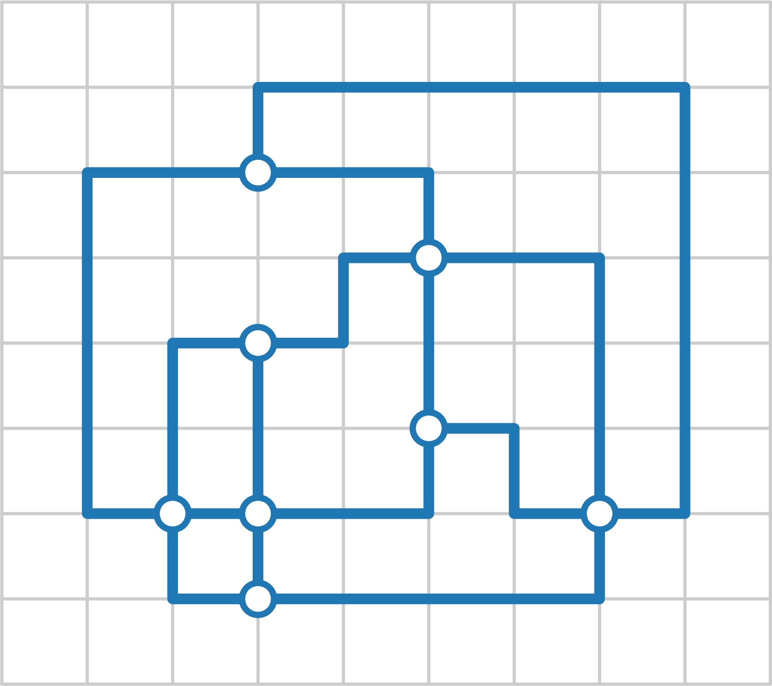
Observations.
- Edges lie on grid $\Rightarrow$ **bends** lie on grid points
- Max degree of each vertex is at most 4
- Otherwise 

Planarization.
- Fix embedding
- Crossings become vertices



Aesthetic criteria.

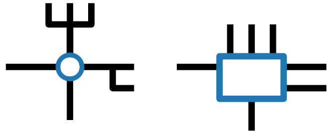
Orthogonal Layout – Definition



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

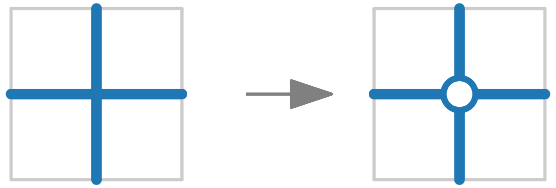
Observations.

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



Planarization.

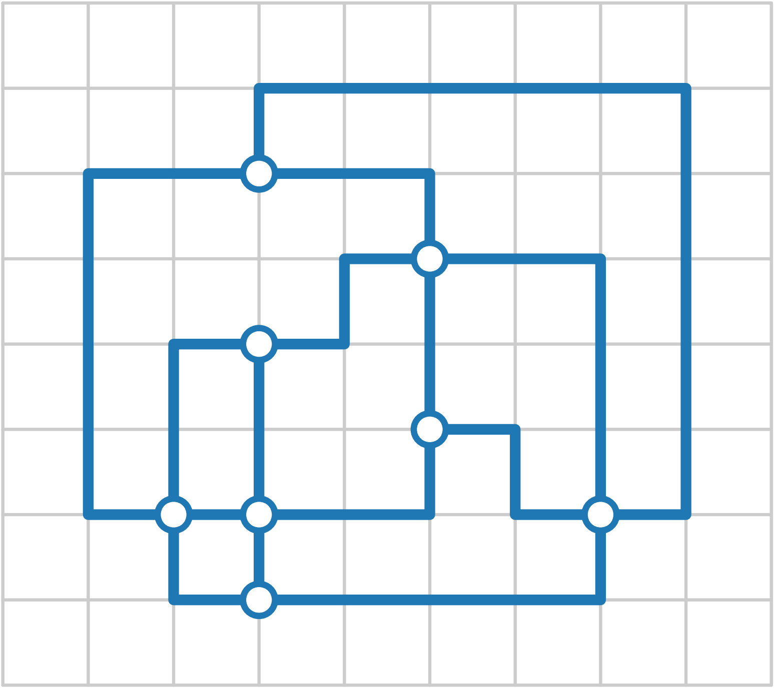
- Fix embedding
- Crossings become vertices



Aesthetic criteria.

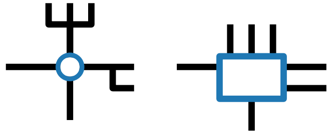
- Number of bends

Orthogonal Layout – Definition

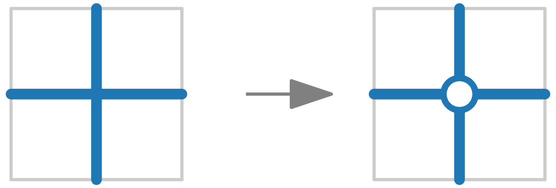


Definition.
A drawing Γ of a graph $G = (V, E)$ is called **orthogonal** if
- vertices are drawn as points on a grid,
- each edge is represented as a sequence of alternating horizontal and vertical segments, and
- pairs of edges are disjoint or cross orthogonally.

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

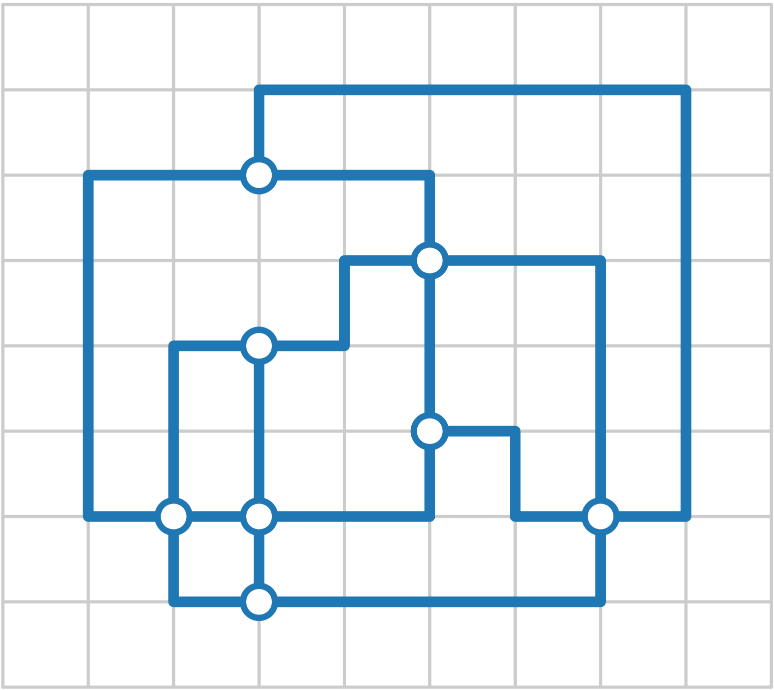


Planarization.
- Fix embedding
- Crossings become vertices



Aesthetic criteria.
- Number of bends
- Length of edges

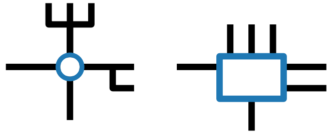
Orthogonal Layout – Definition



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

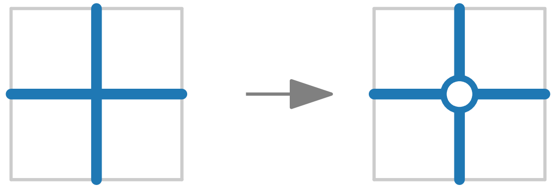
Observations.

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



Planarization.

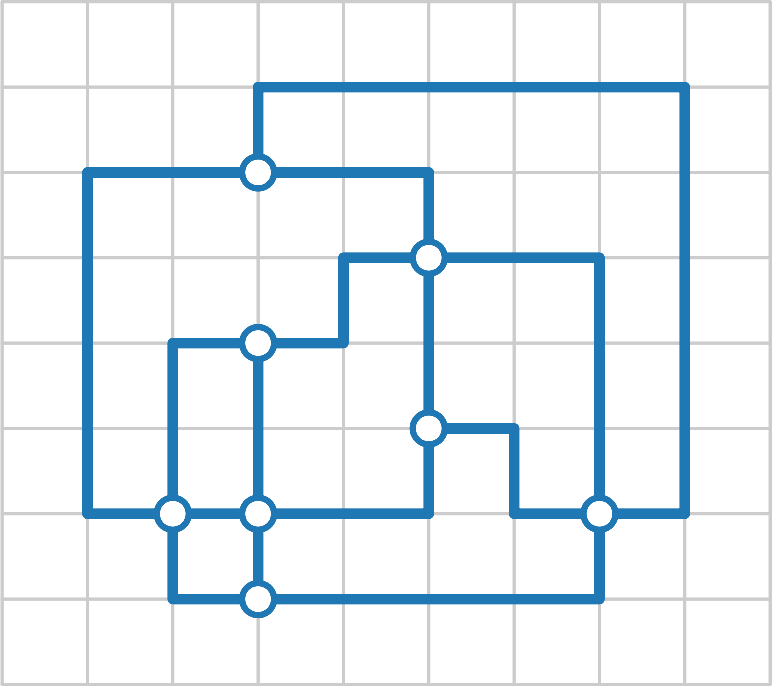
- Fix embedding
- Crossings become vertices



Aesthetic criteria.

- Number of bends
- Length of edges
- Width, height, area

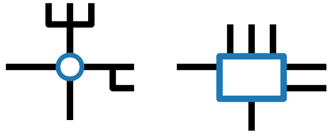
Orthogonal Layout – Definition



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

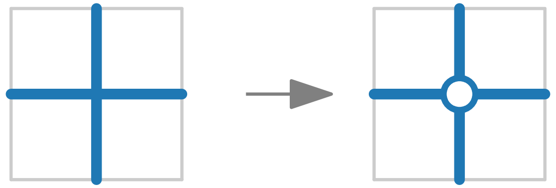
Observations.

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



Planarization.

- Fix embedding
- Crossings become vertices



Aesthetic criteria.

- 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]

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

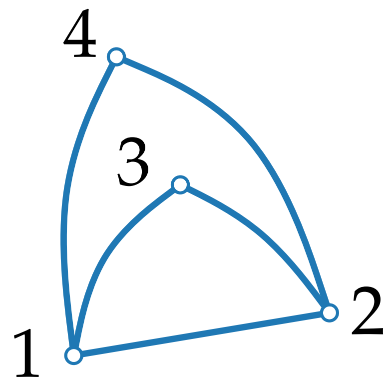
TOPOLOGY – SHAPE – METRICS

Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

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

combinatorial embedding/ planarization



TOPOLOGY — SHAPE — METRICS

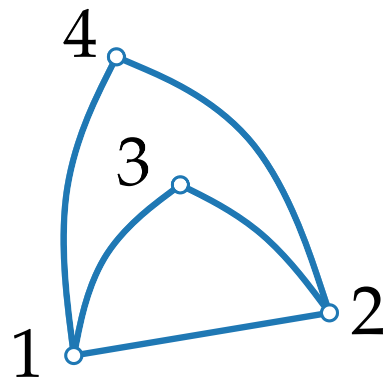
Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

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

combinatorial embedding/ planarization

reduce crossings



TOPOLOGY — SHAPE — METRICS

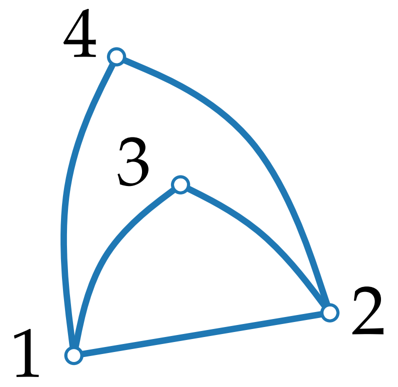
Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

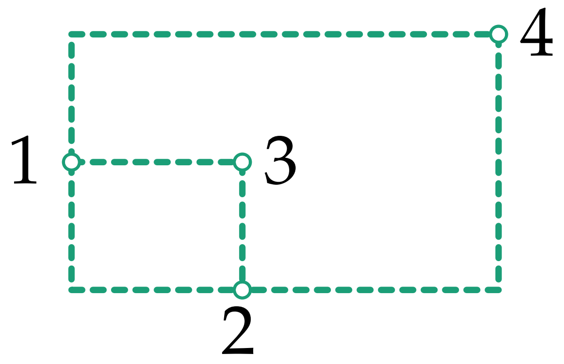
$V = \{v_1, v_2, v_3, v_4\}$
$E = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$

combinatorial embedding/ planarization

reduce crossings



orthogonal representation



TOPOLOGY — SHAPE — METRICS

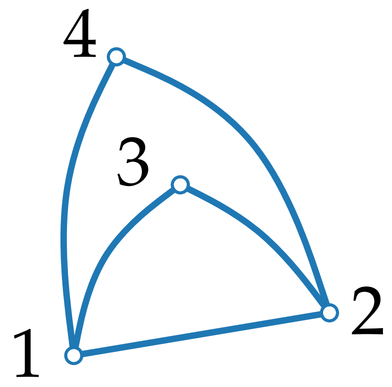
Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

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

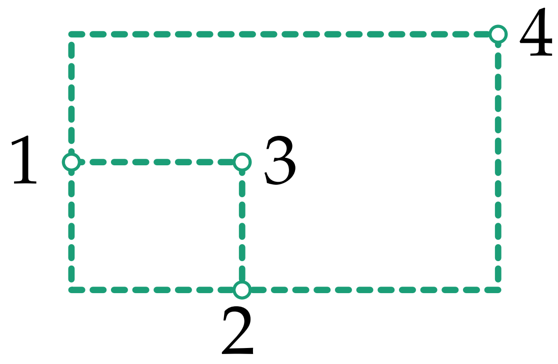
reduce crossings

combinatorial embedding/ planarization



bend minimization

orthogonal representation



TOPOLOGY — SHAPE — METRICS

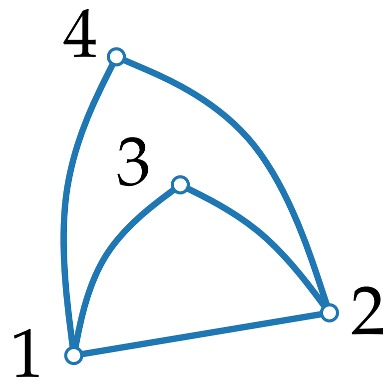
Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

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

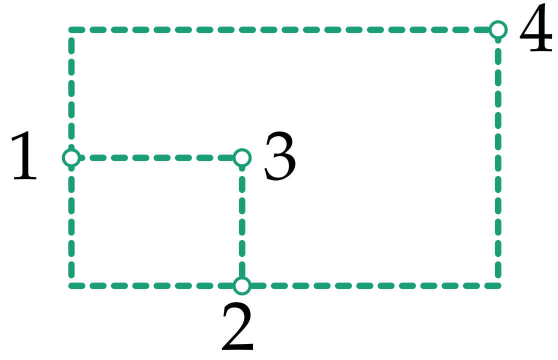
reduce crossings

combinatorial embedding/ planarization

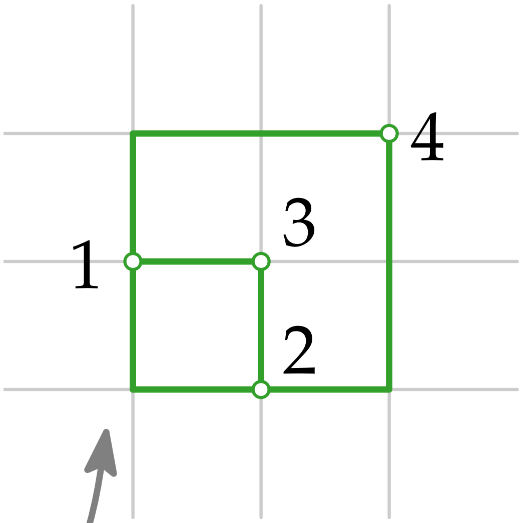


bend minimization

orthogonal representation



planar orthogonal drawing



Topology — Shape — Metrics

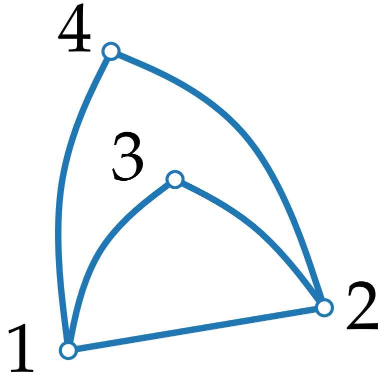
Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

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

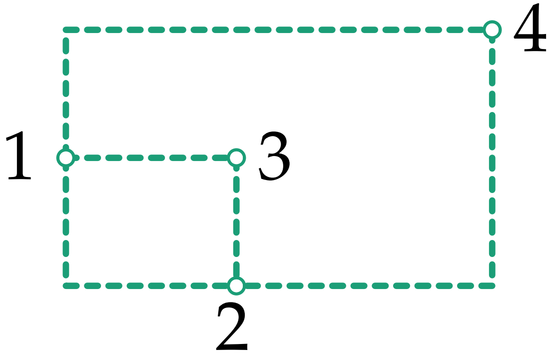
combinatorial embedding/ planarization

reduce crossings

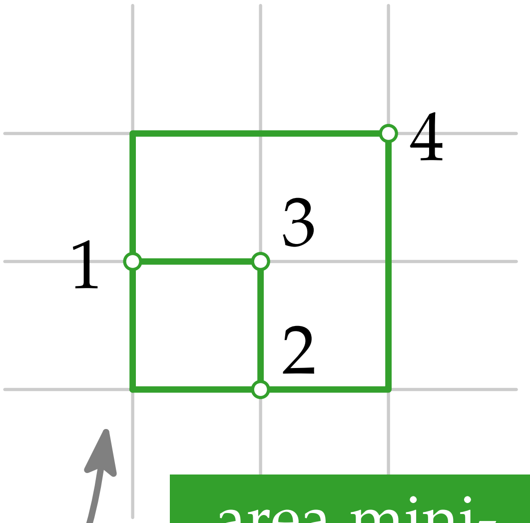


bend minimization

orthogonal representation



planar orthogonal drawing

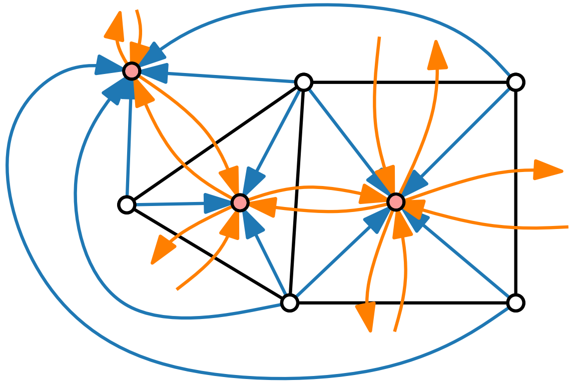


area minimization

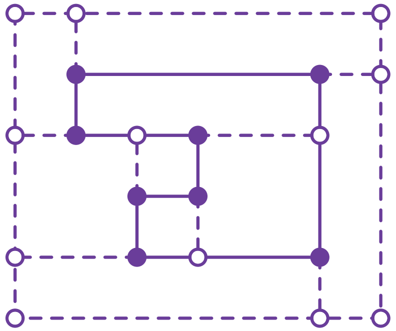
TOPOLOGY — SHAPE — METRICS

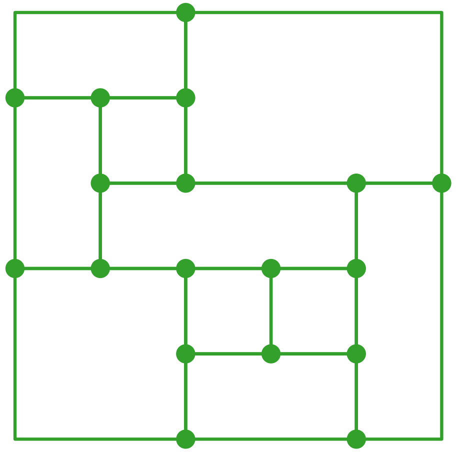
Universität Trier

Visualization of Graphs



Lecture 6:
Orthogonal Layouts





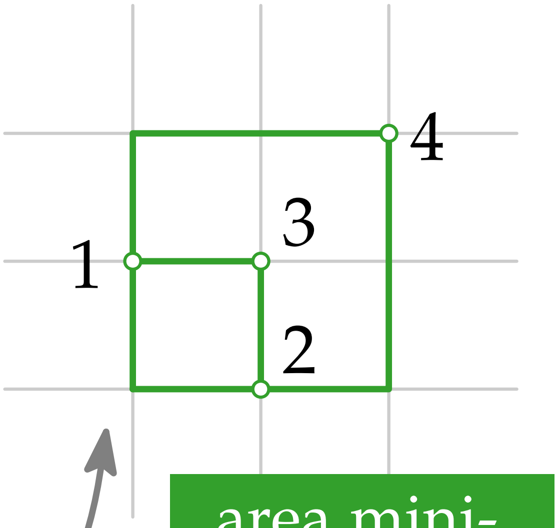
Part II:
Orthogonal Representation

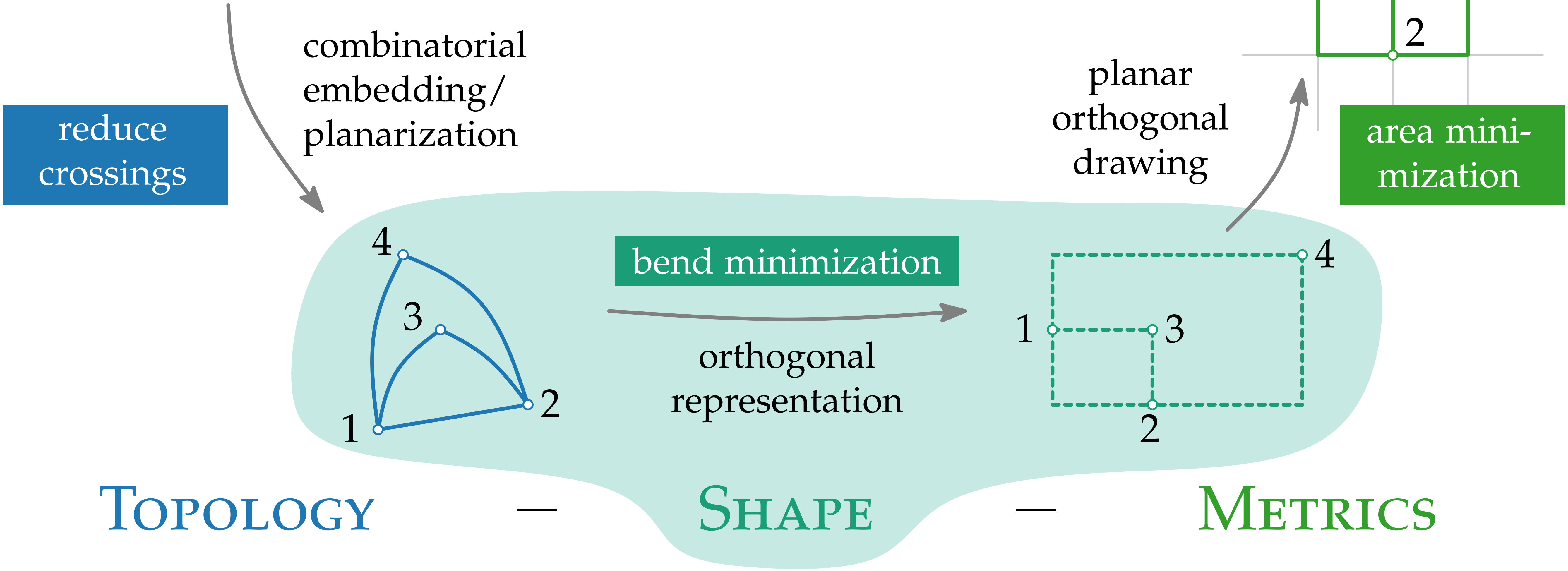
Philipp Kindermann

Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

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





Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

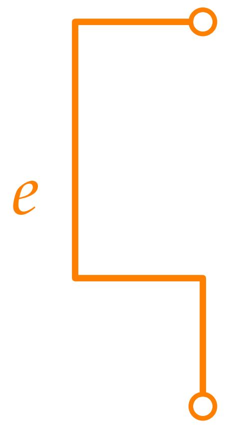
Definitions.
Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

Definitions.
Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

- Let e be an edge



Orthogonal Representation

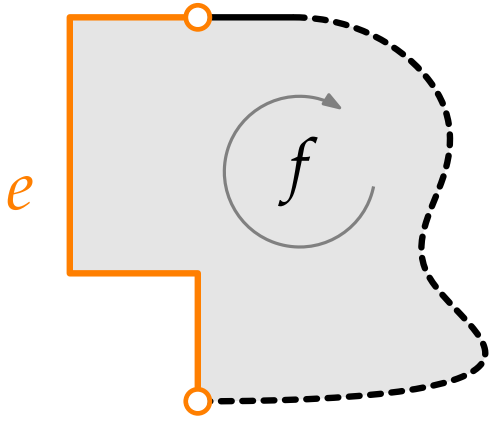
Idea.

Describe orthogonal drawing combinatorically.

Definitions.

Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

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

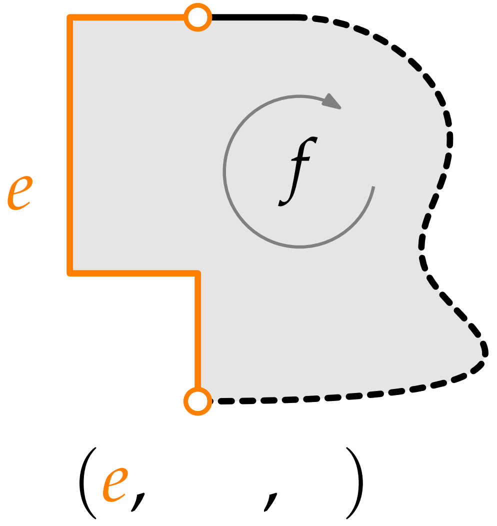


Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

Definitions.
Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

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

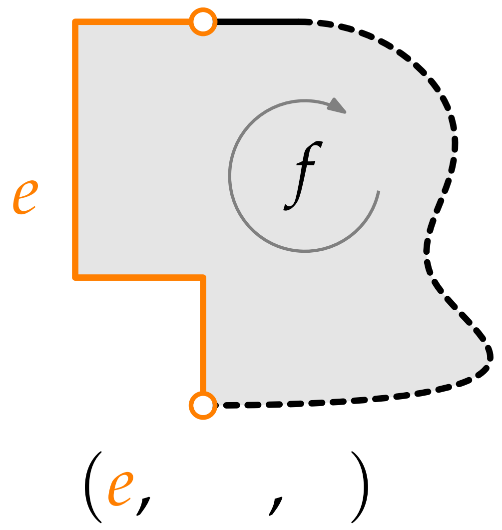


Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

Definitions.
Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

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

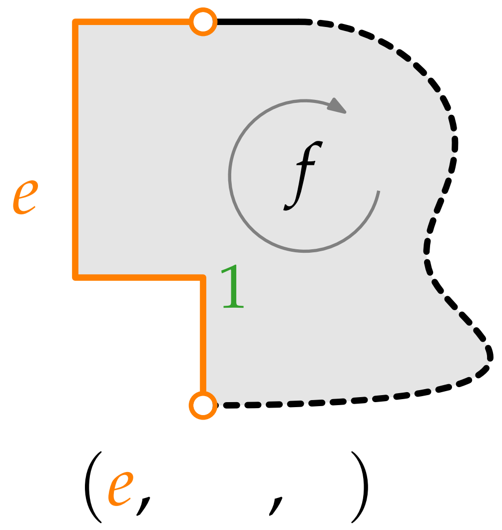


Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

Definitions.
Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

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

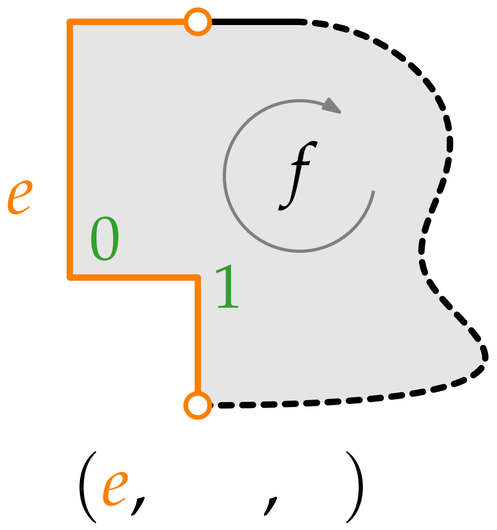


Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

Definitions.
Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

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

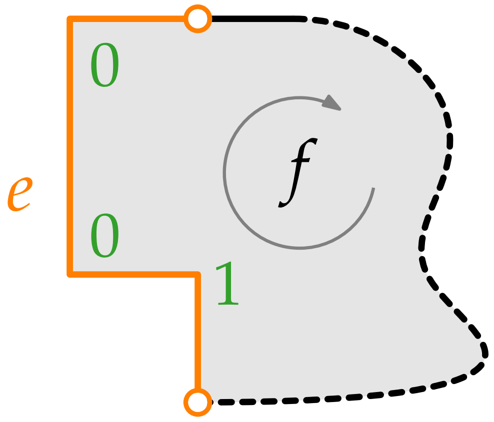


Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

Definitions.
Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

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



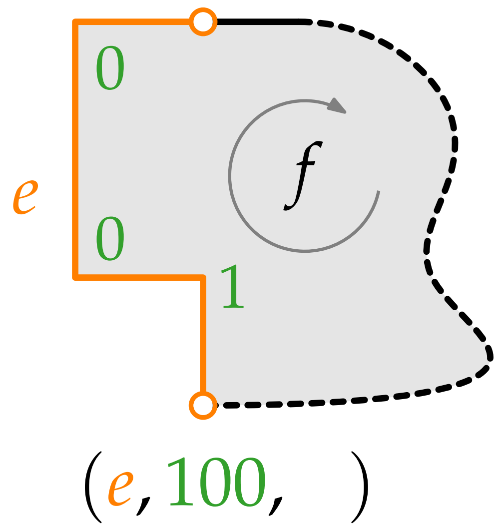
$(e, \quad , \quad)$

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

Definitions.
Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

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

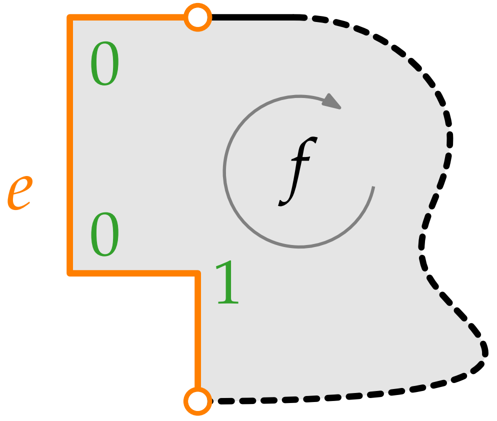


Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

Definitions.
Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

- Let e be an edge with the face f to the right.
  An **edge description** of e wrt f is a triple (e, δ, α) where
  - δ is a sequence of $\{0,1\}^*$ (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'



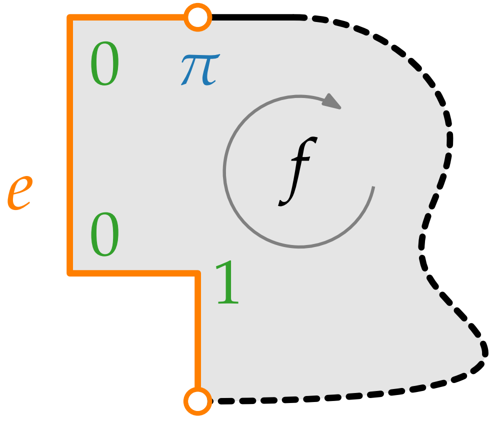
$(e, 100, \quad)$

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

Definitions.
Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

- Let e be an edge with the face f to the right.
  An **edge description** of e wrt f is a triple (e, δ, α) where
  - δ is a sequence of $\{0,1\}^*$ (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'



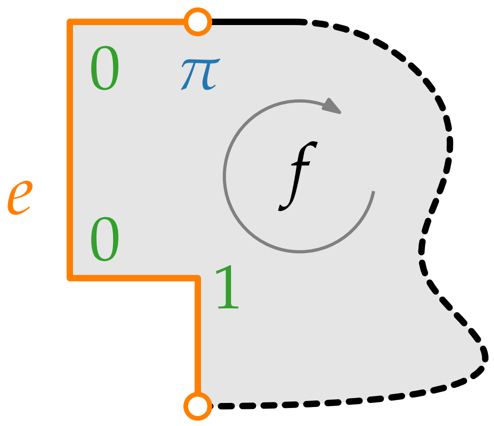
$(e, 100, \quad)$

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

Definitions.
Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

- Let e be an edge with the face f to the right.
  An **edge description** of e wrt f is a triple (e, δ, α) where
  - δ is a sequence of $\{0,1\}^*$ (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'



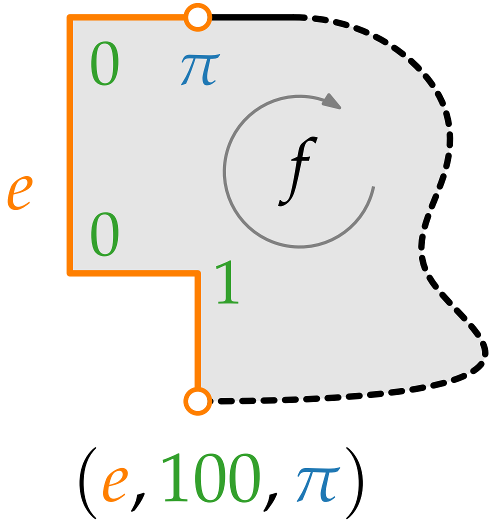
$(e, 100, \pi)$

Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

Definitions.
Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

- Let e be an edge with the face f to the right.
  An **edge description** of e wrt f is a triple (e, δ, α) where
  - δ is a sequence of $\{0,1\}^*$ (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 f is a clockwise ordered sequence of edge descriptions (e, δ, α).



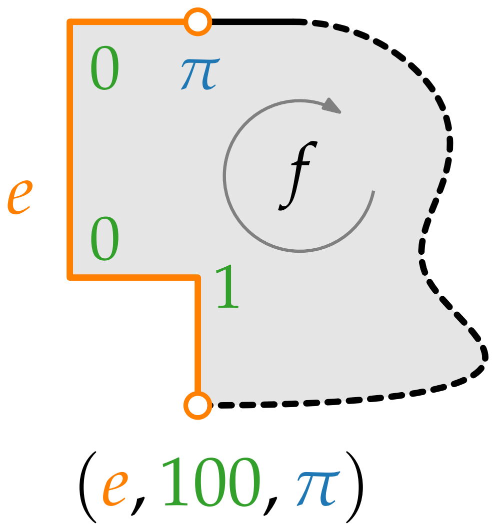
Orthogonal Representation

Idea.
Describe orthogonal drawing combinatorically.

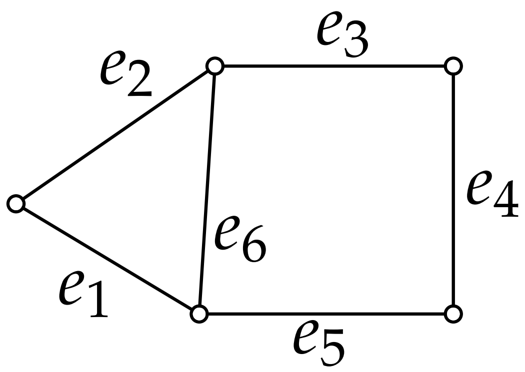
Definitions.
Let $G = (V, E)$ be a plane graph with faces F and outer face f_0.

- Let e be an edge with the face f to the right.
  An **edge description** of e wrt f is a triple (e, δ, α) where
  - δ is a sequence of $\{0,1\}^*$ (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 f is a clockwise ordered sequence of edge descriptions (e, δ, α).
- An **orthogonal representation** $H(G)$ of G is defined as

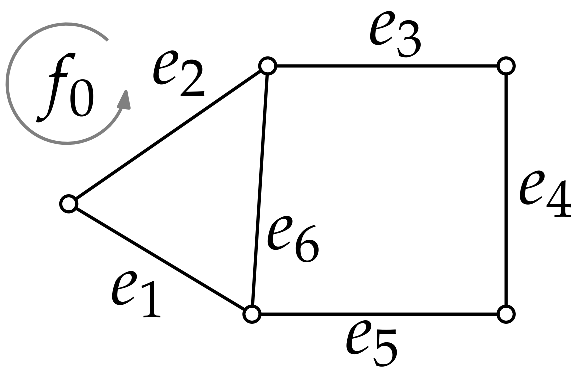
$$H(G) = \{H(f) \mid f \in F\}.$$



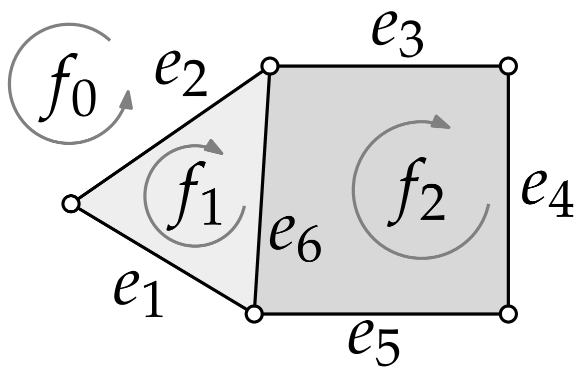
Orthogonal Representation – Example



Orthogonal Representation – Example



Orthogonal Representation – Example

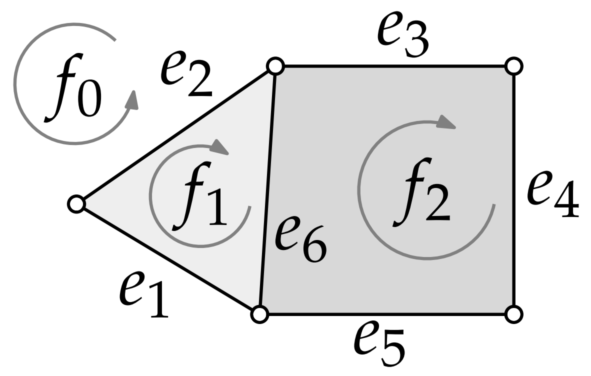


Orthogonal Representation – Example

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

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

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

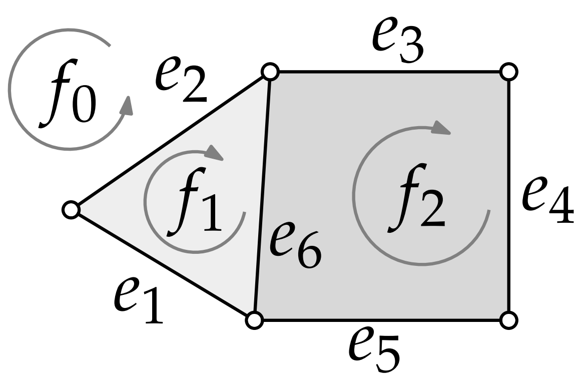


Orthogonal Representation – Example

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

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

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



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

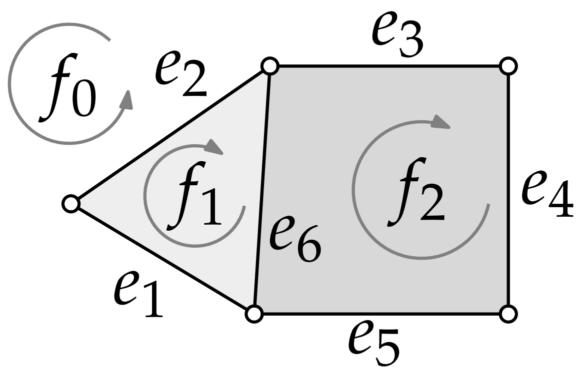
Orthogonal Representation – Example

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

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

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

f_0

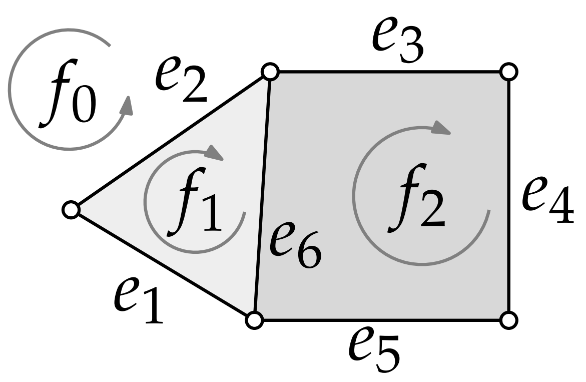


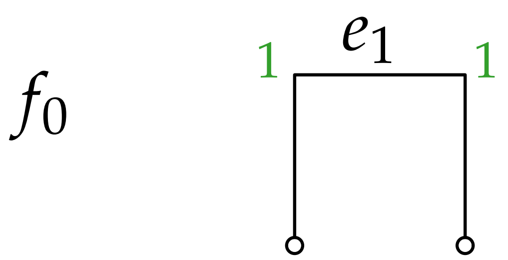
Orthogonal Representation – Example

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

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

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



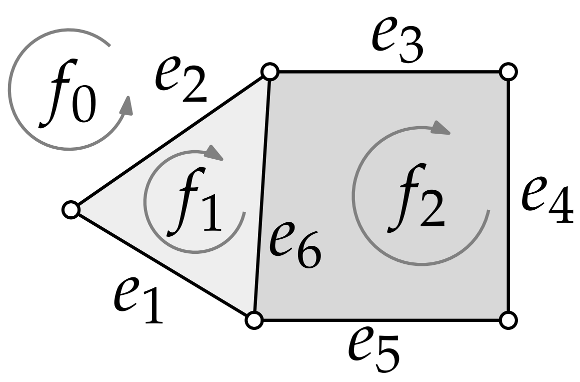


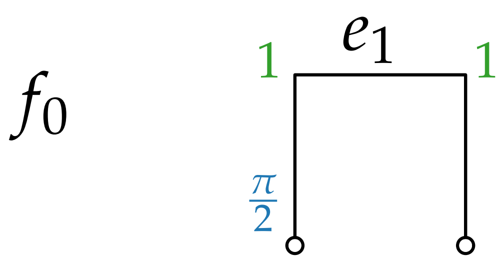
Orthogonal Representation – Example

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

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

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



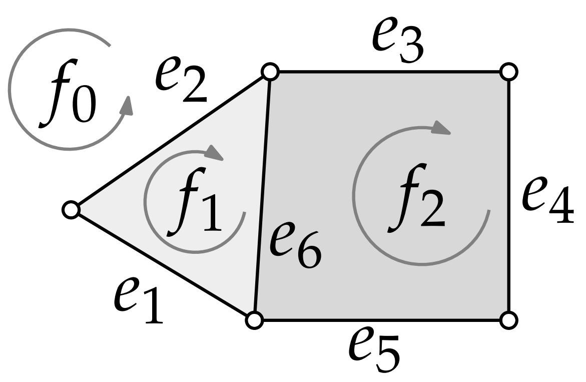


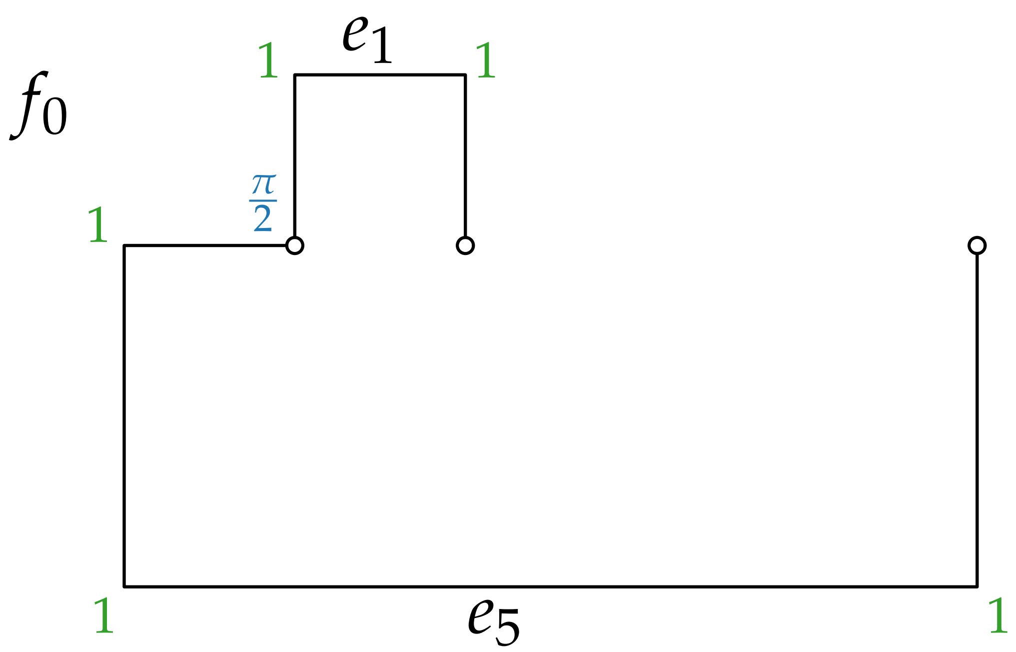
Orthogonal Representation – Example

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

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

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



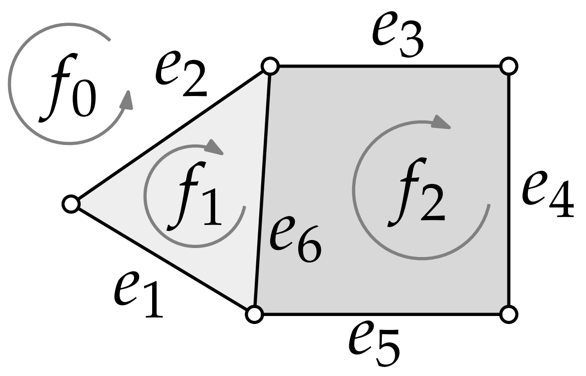


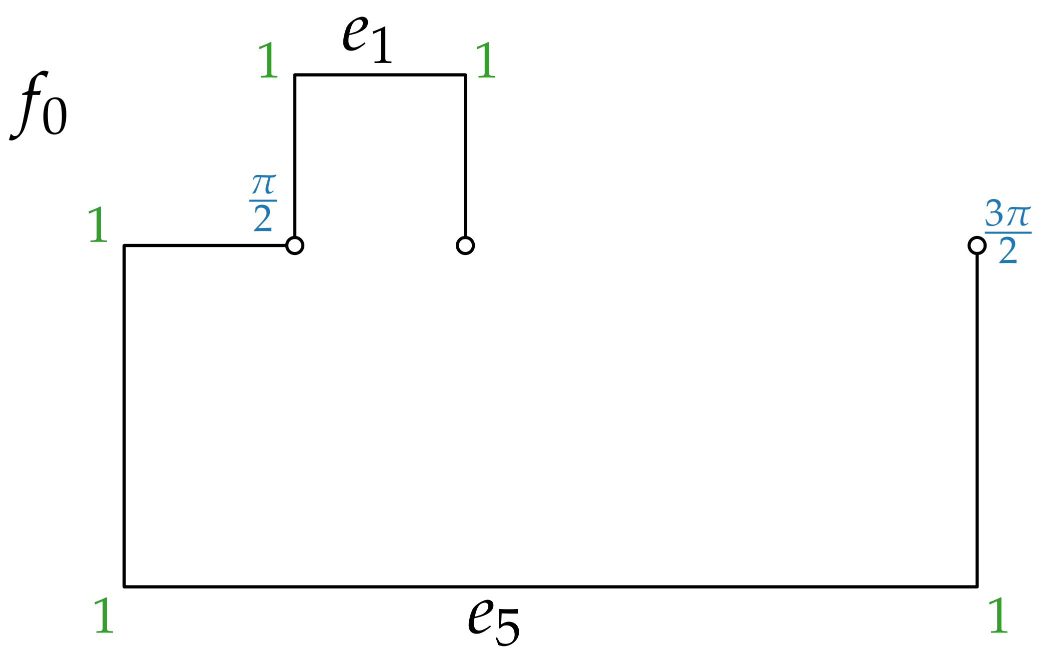
Orthogonal Representation – Example

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

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

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



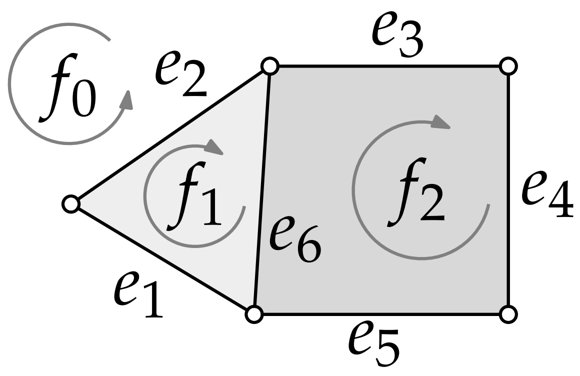


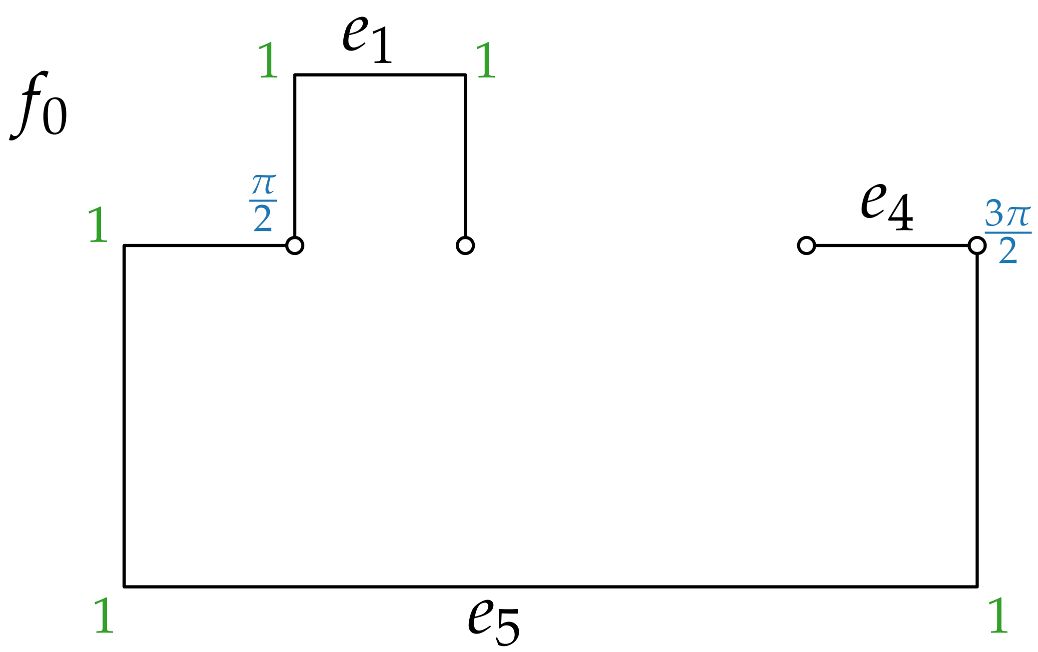
Orthogonal Representation – Example

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

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

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



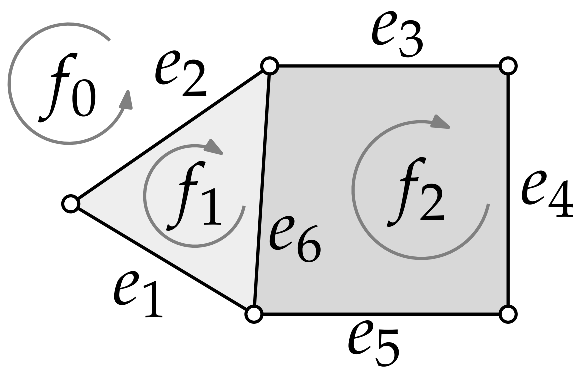


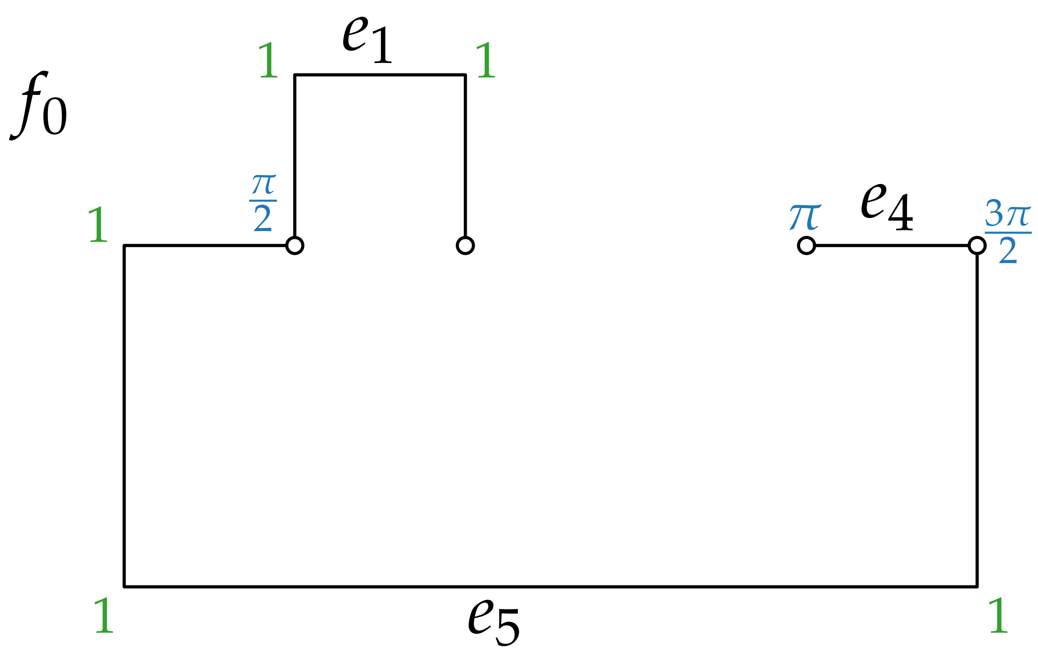
Orthogonal Representation – Example

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

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

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



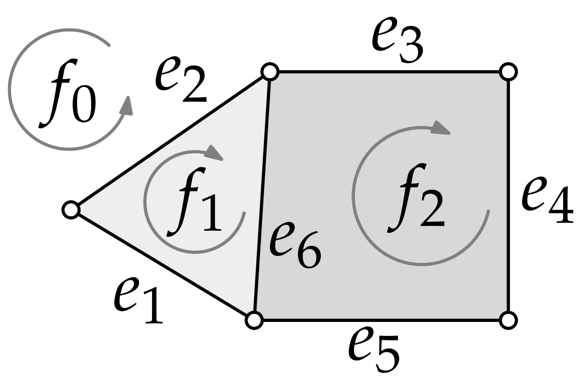


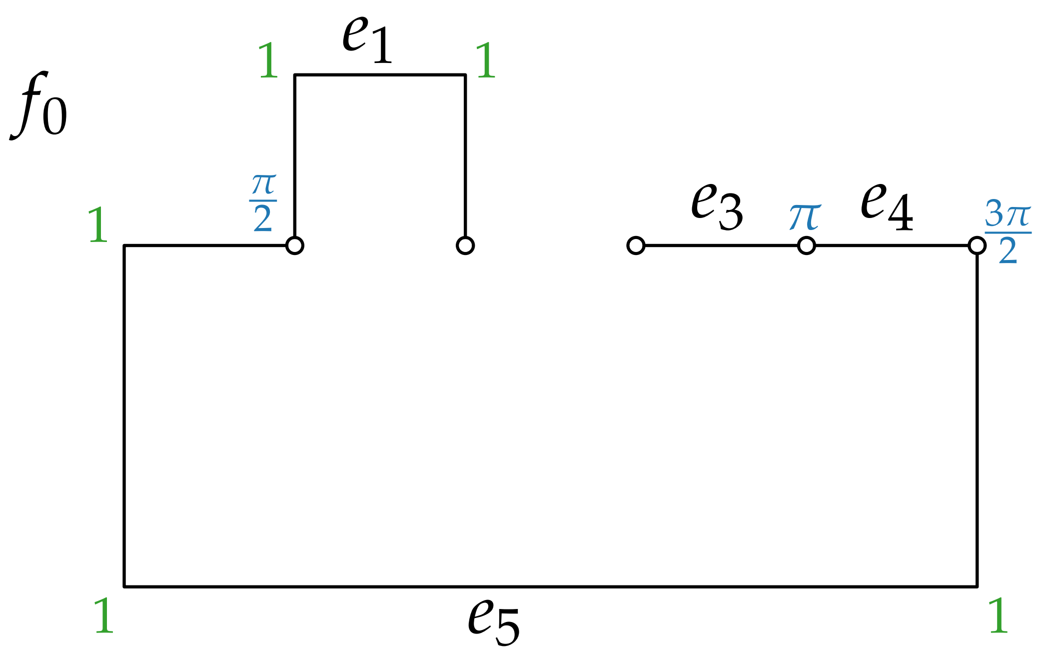
Orthogonal Representation – Example

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

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

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



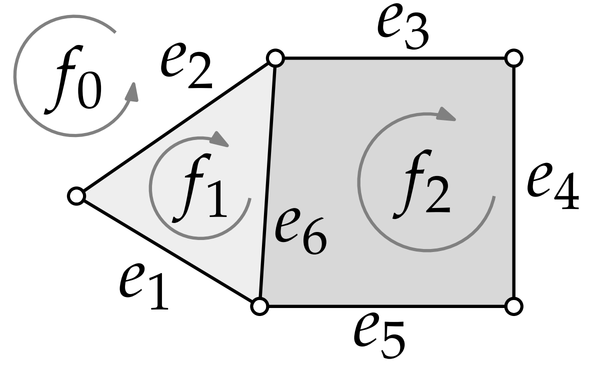


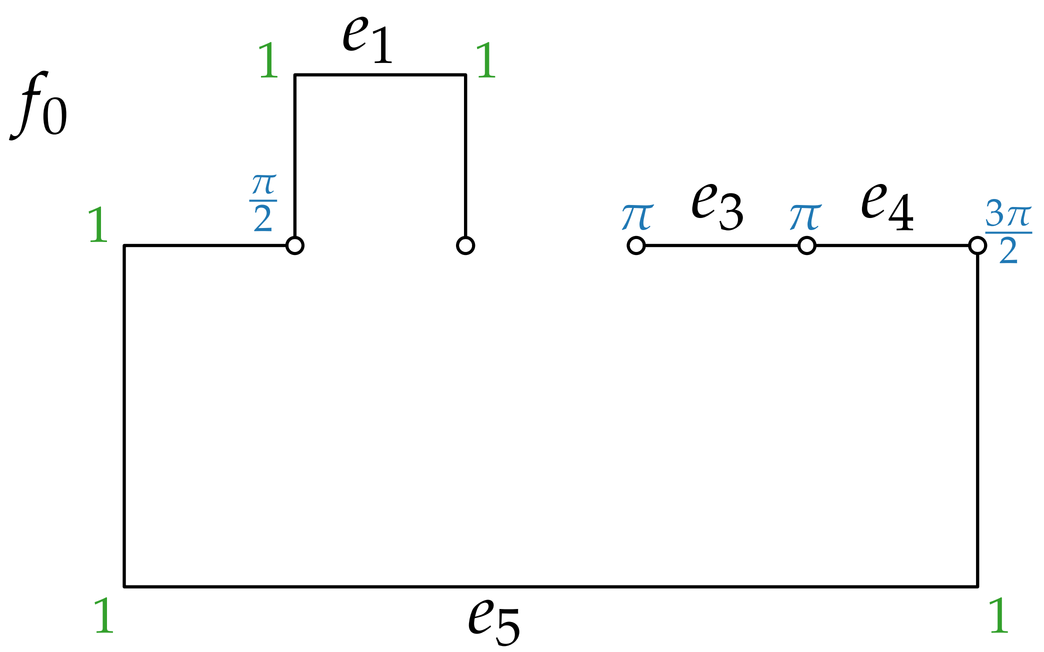
Orthogonal Representation – Example

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

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

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



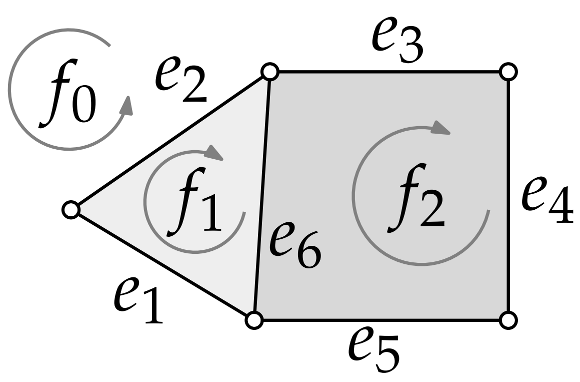


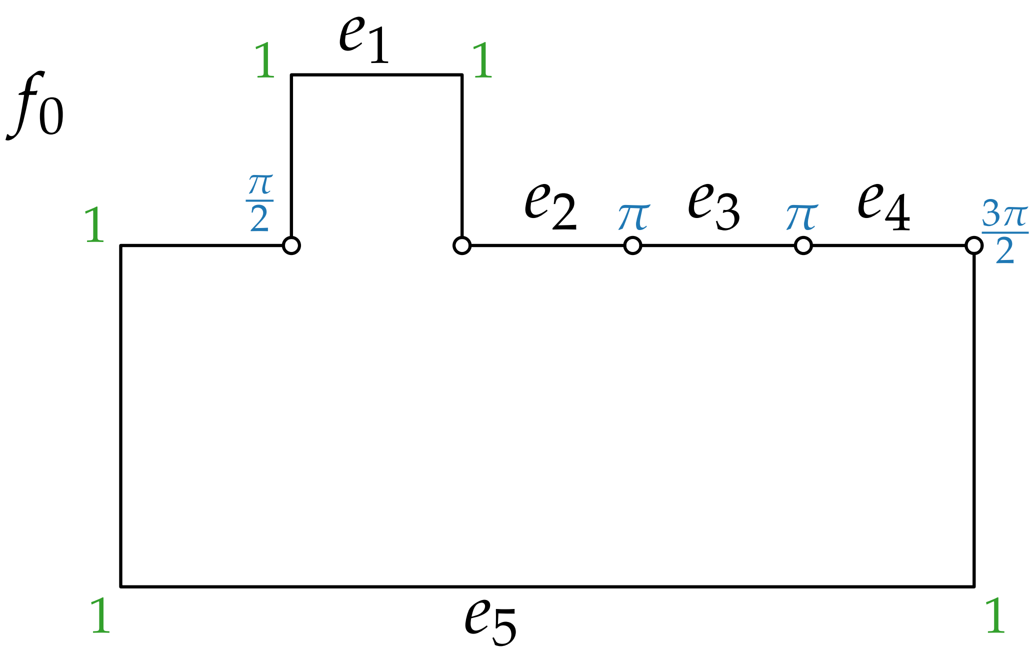
Orthogonal Representation – Example

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

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

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



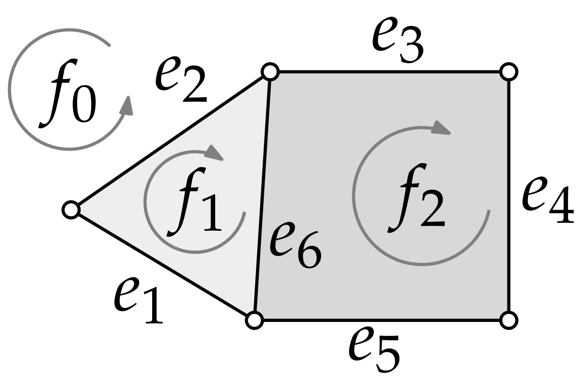


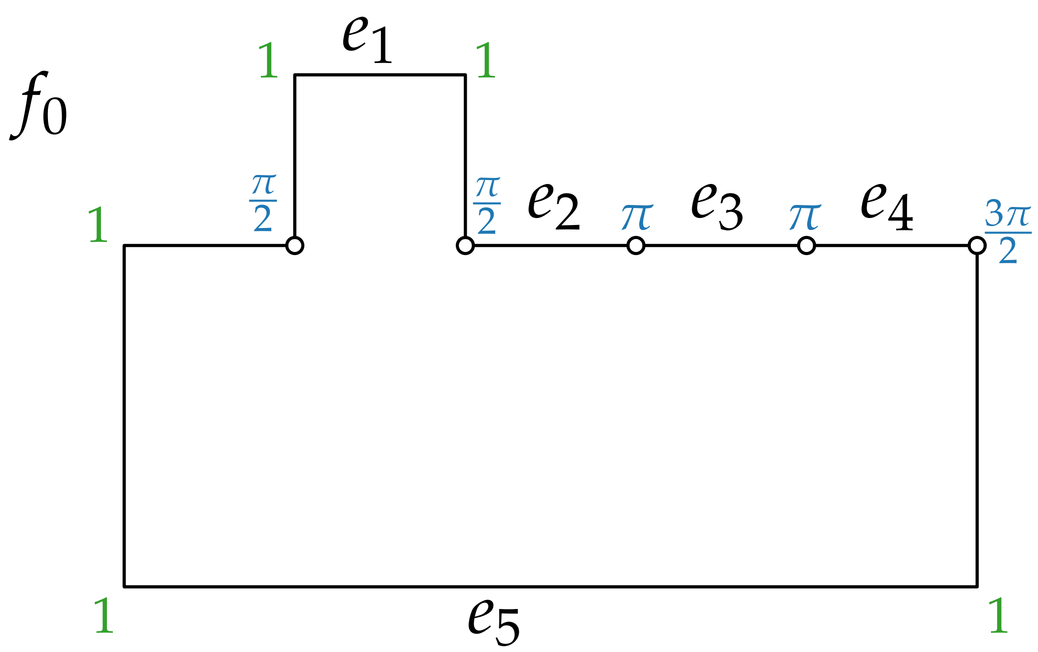
Orthogonal Representation – Example

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

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

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



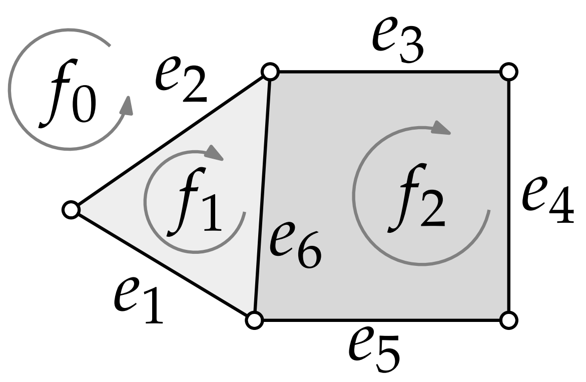


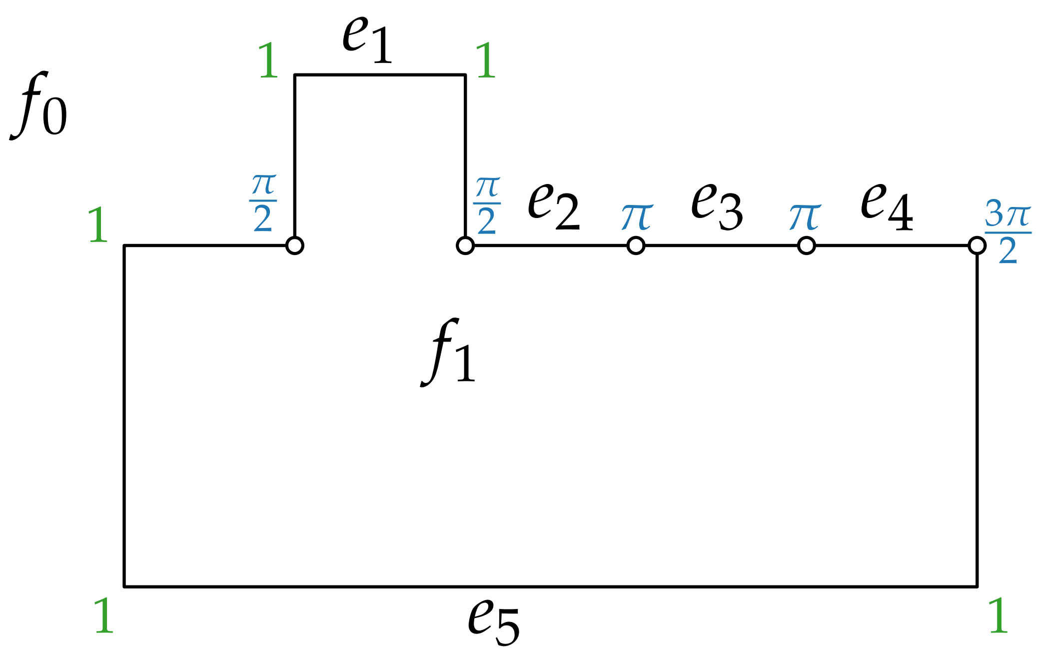
Orthogonal Representation – Example

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

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

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



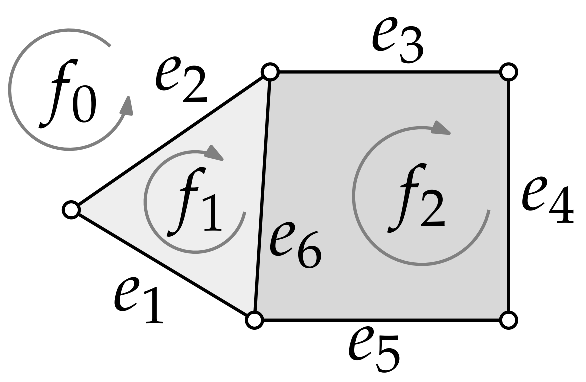


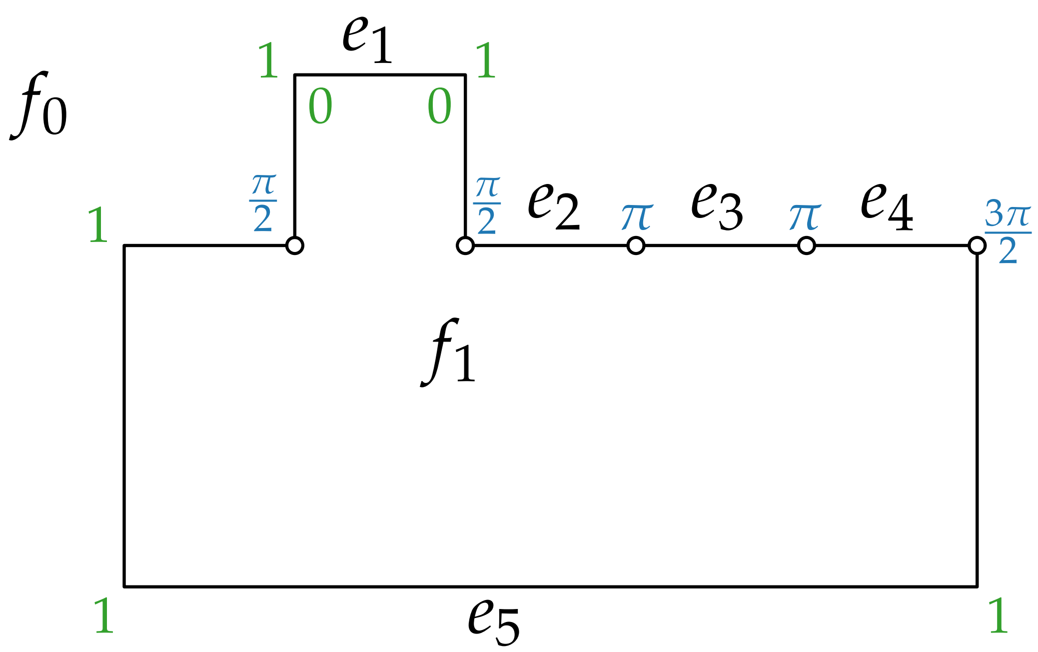
Orthogonal Representation – Example

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

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

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



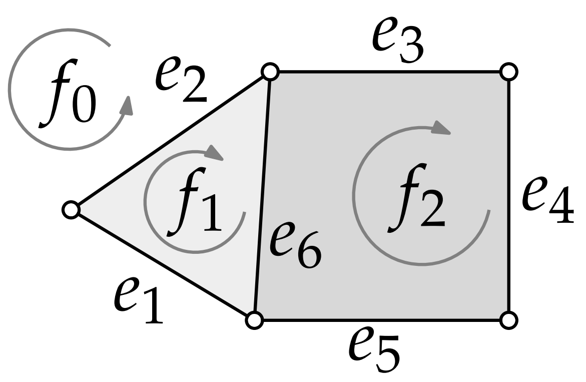


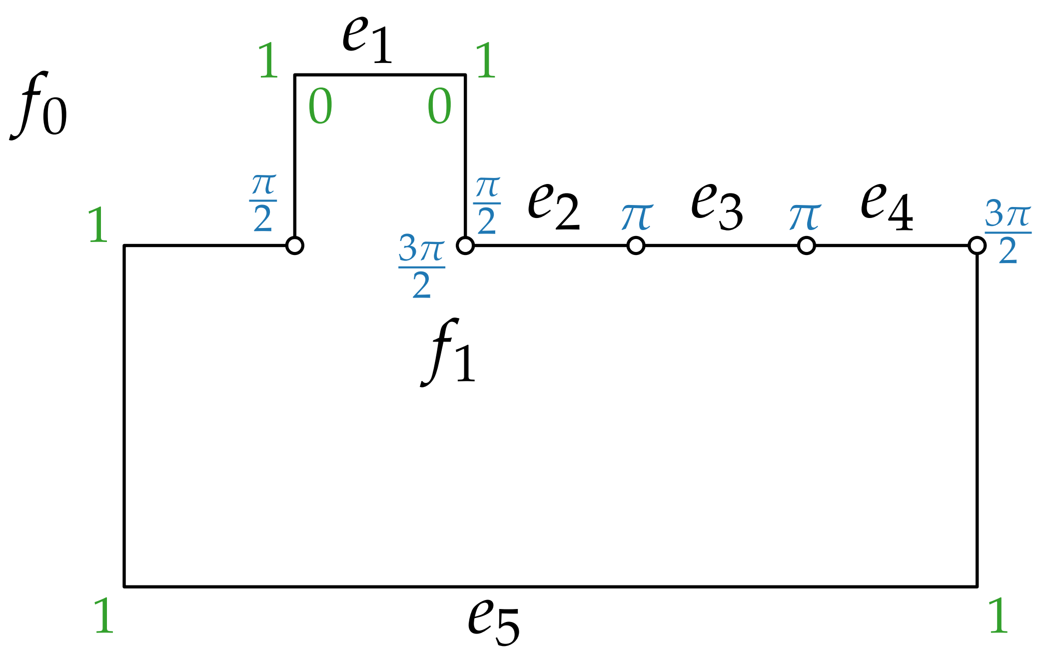
Orthogonal Representation – Example

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

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

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



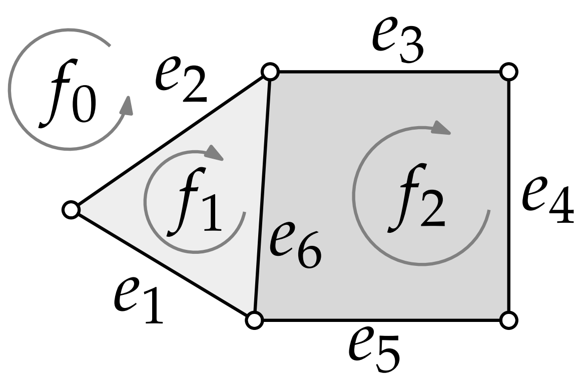


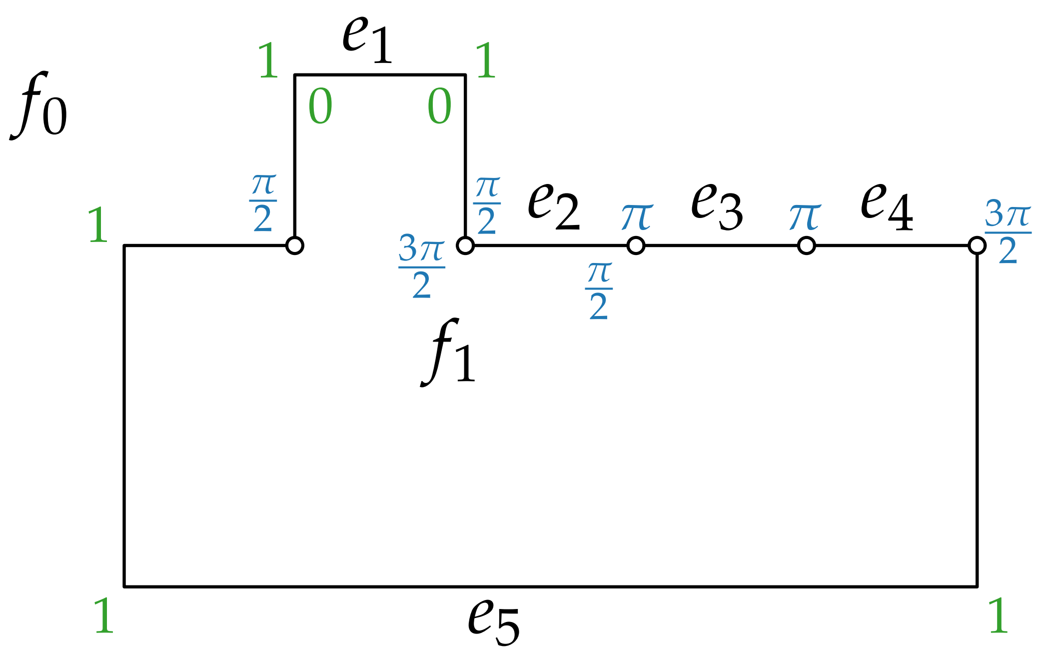
Orthogonal Representation – Example

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

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

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



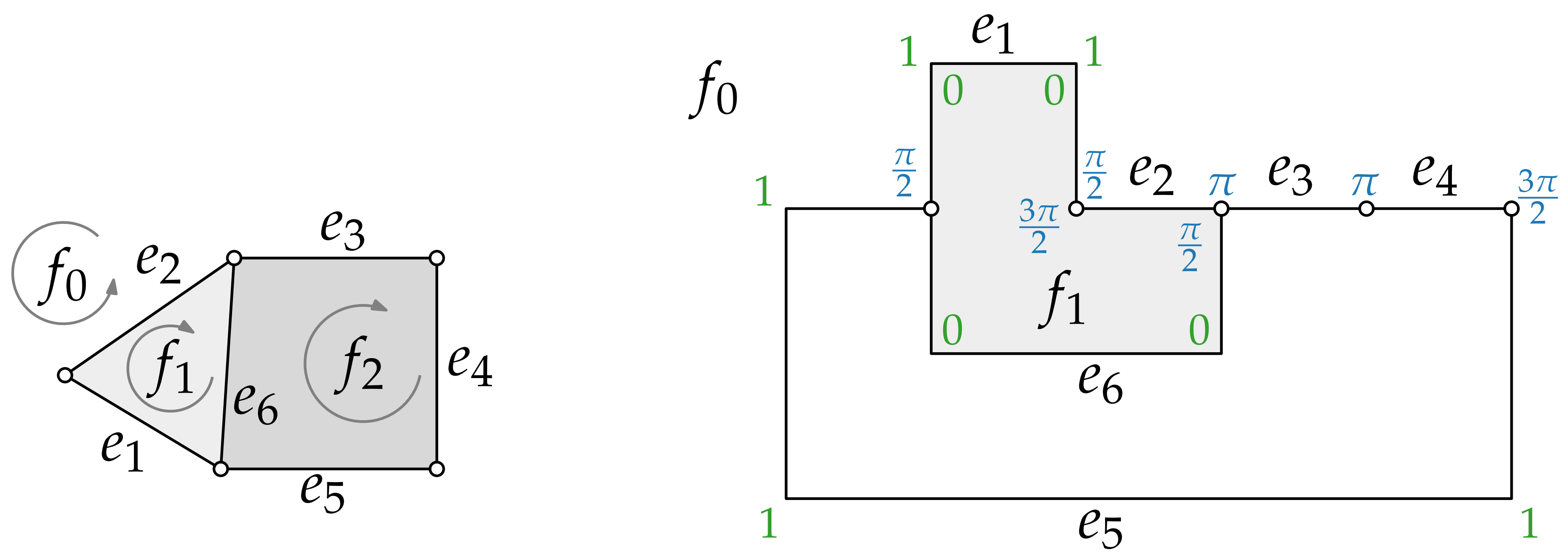


Orthogonal Representation – Example

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

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

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

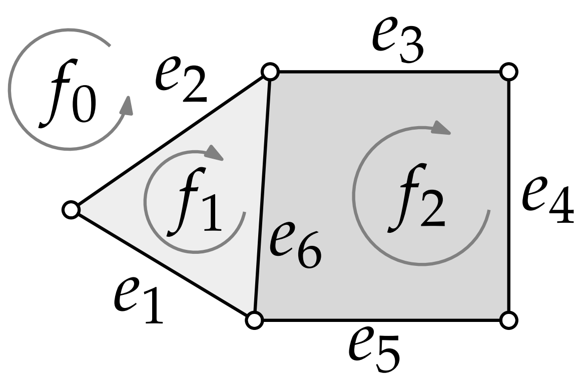


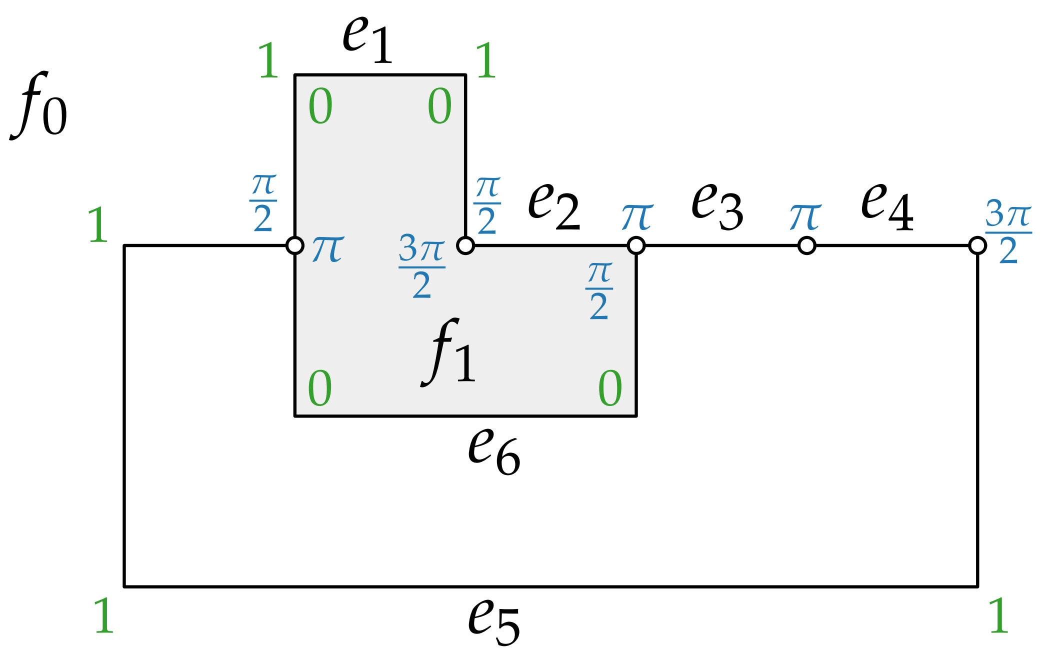
Orthogonal Representation – Example

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

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

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



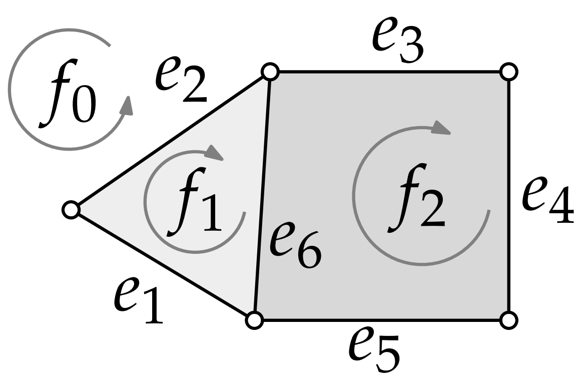


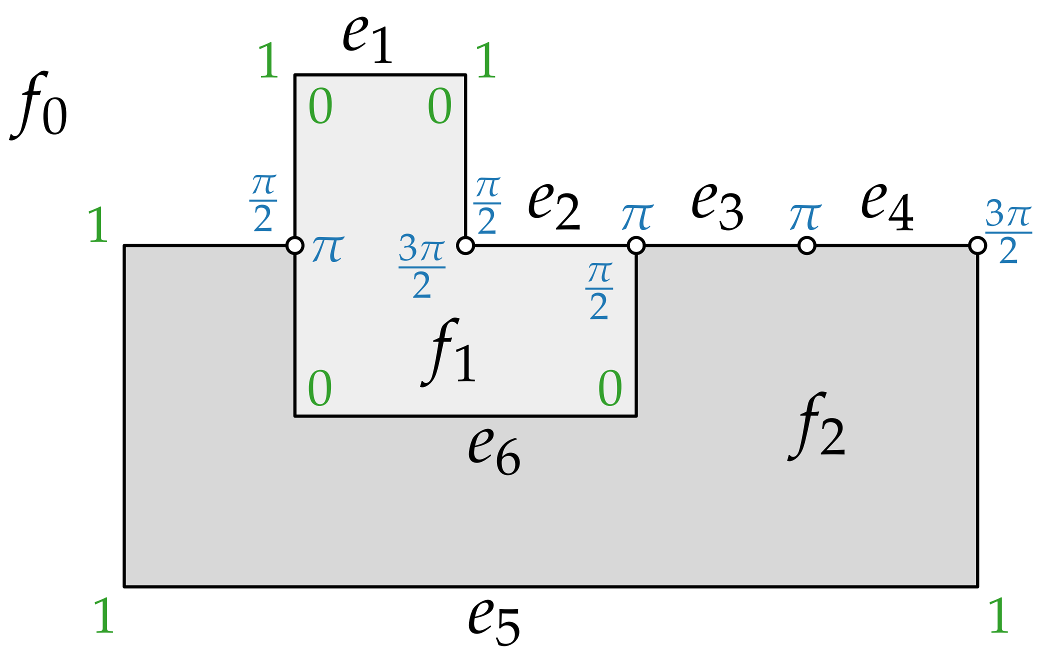
Orthogonal Representation – Example

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

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

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



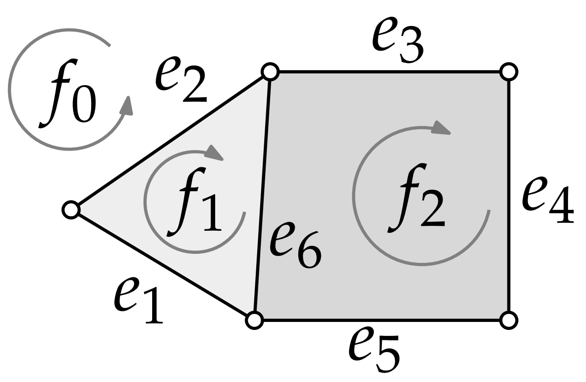


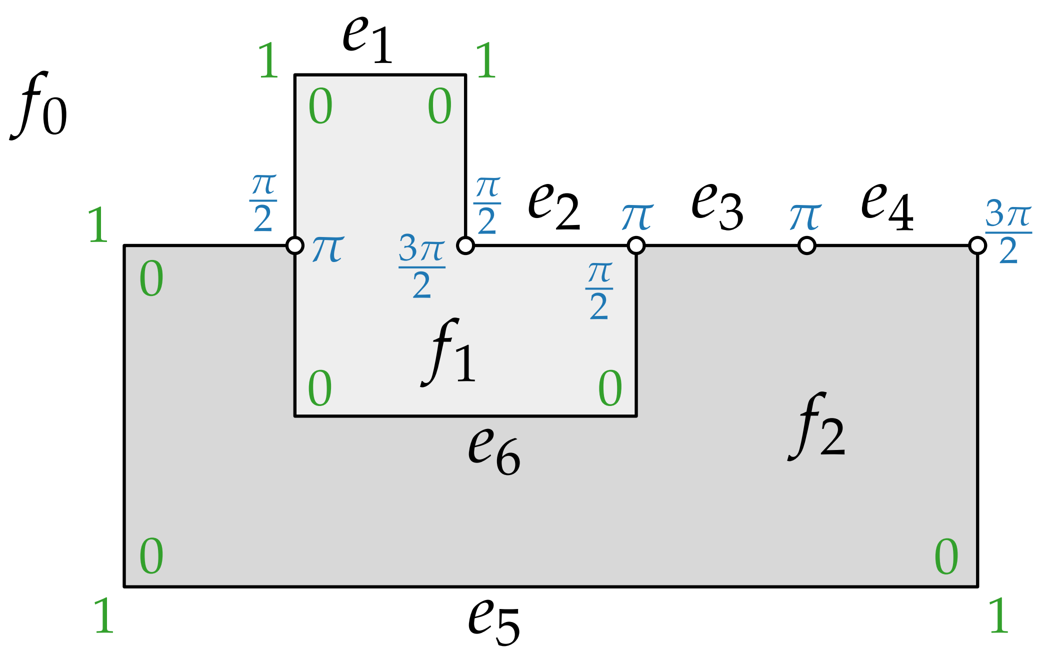
Orthogonal Representation – Example

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

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

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



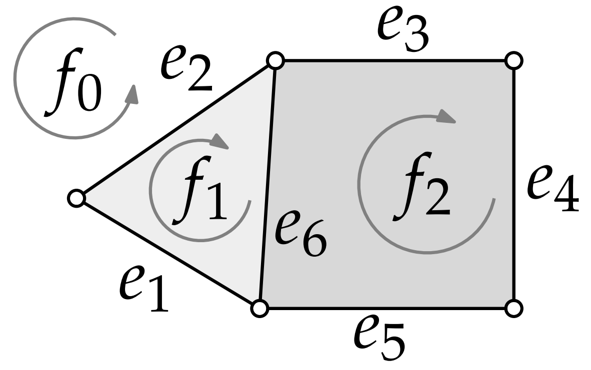


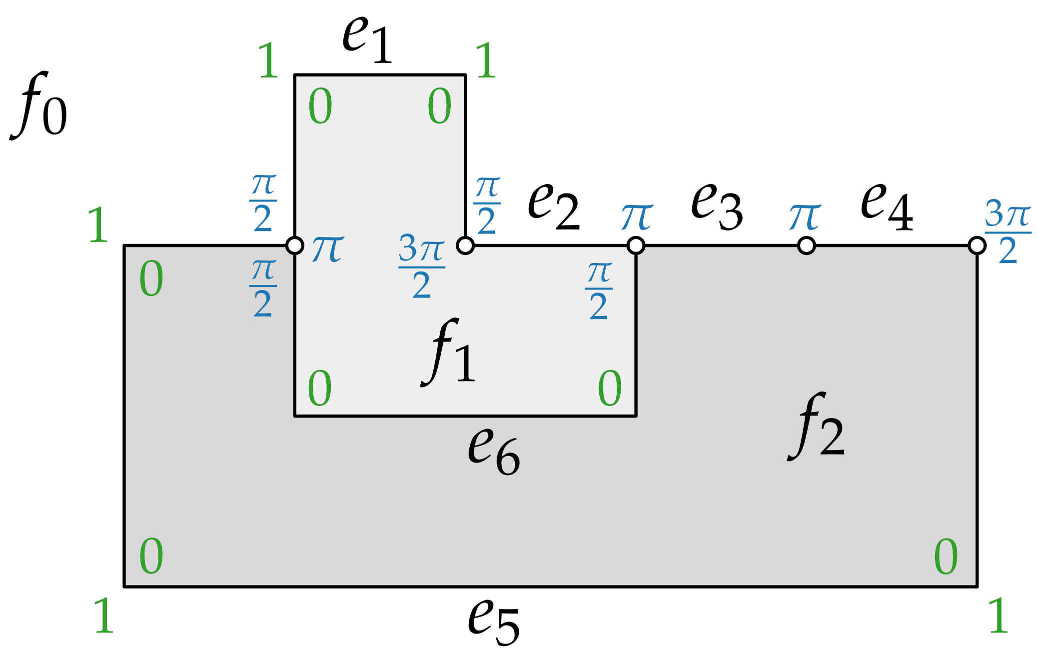
Orthogonal Representation – Example

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

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

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



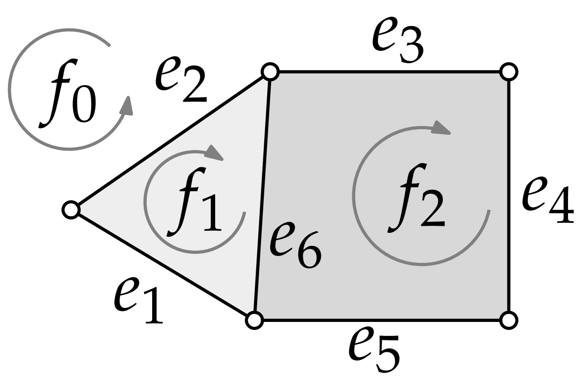


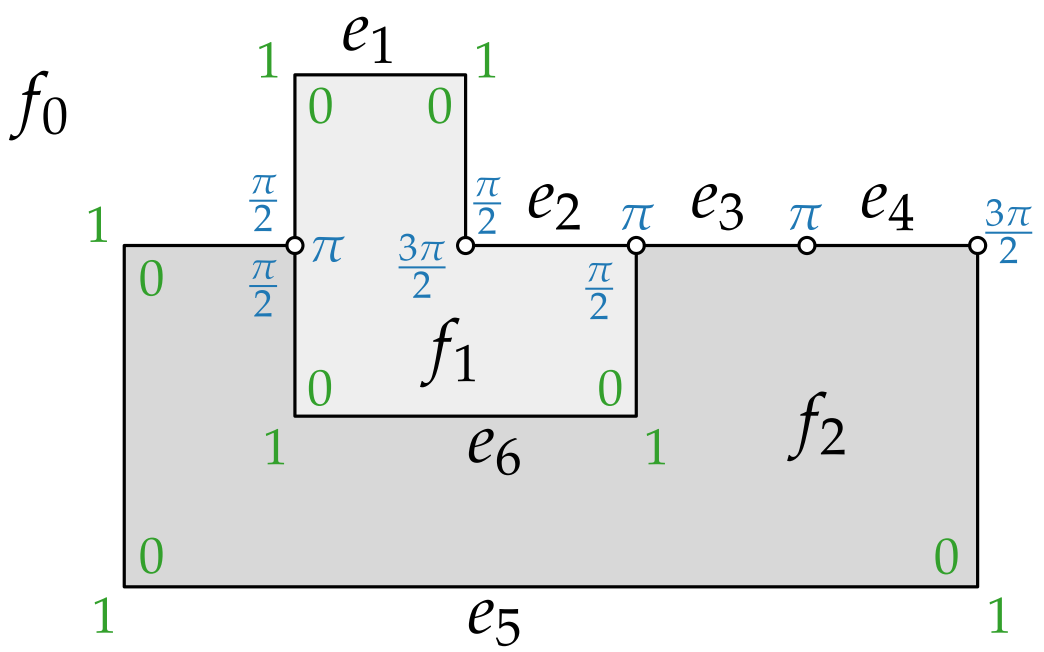
Orthogonal Representation – Example

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

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

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



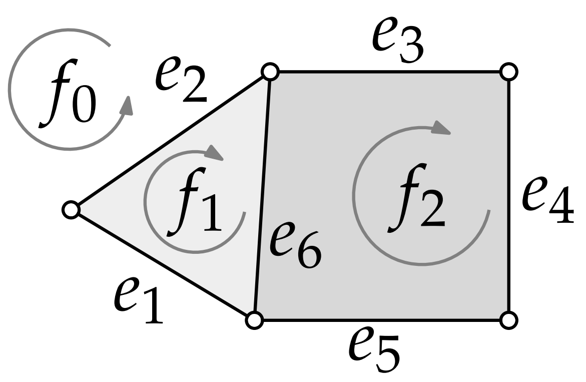


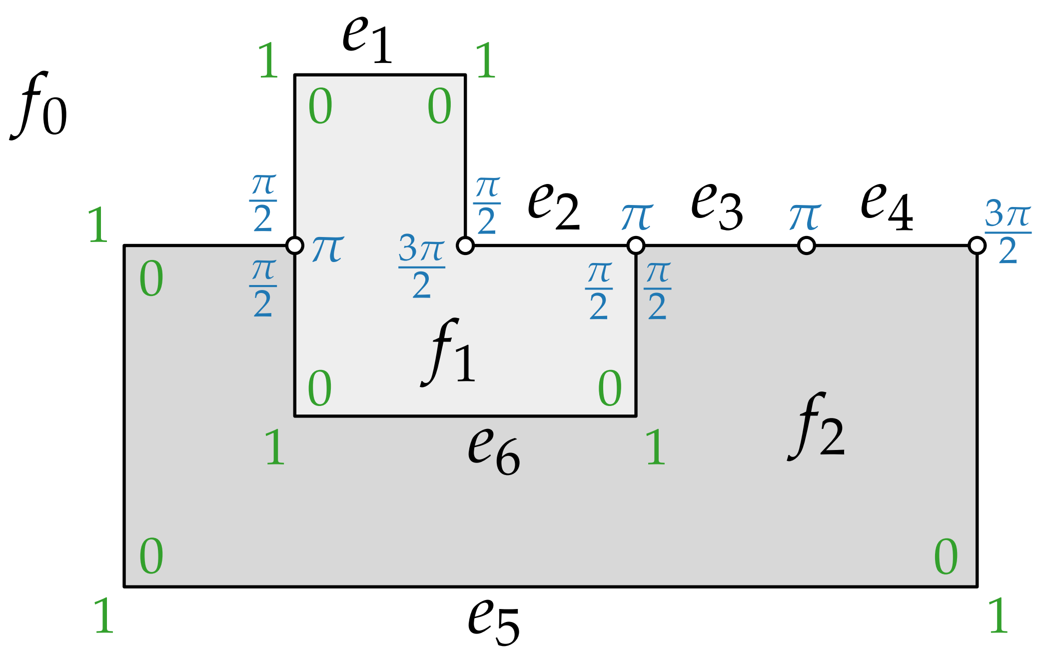
Orthogonal Representation – Example

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

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

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



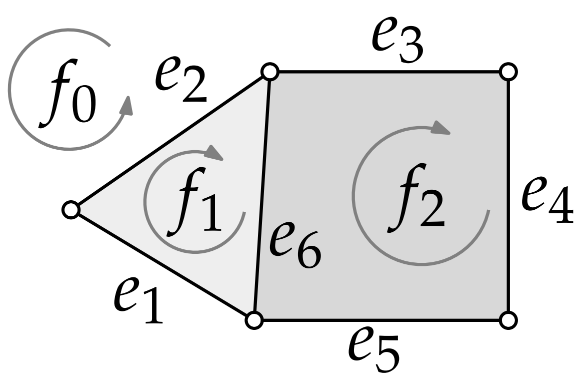


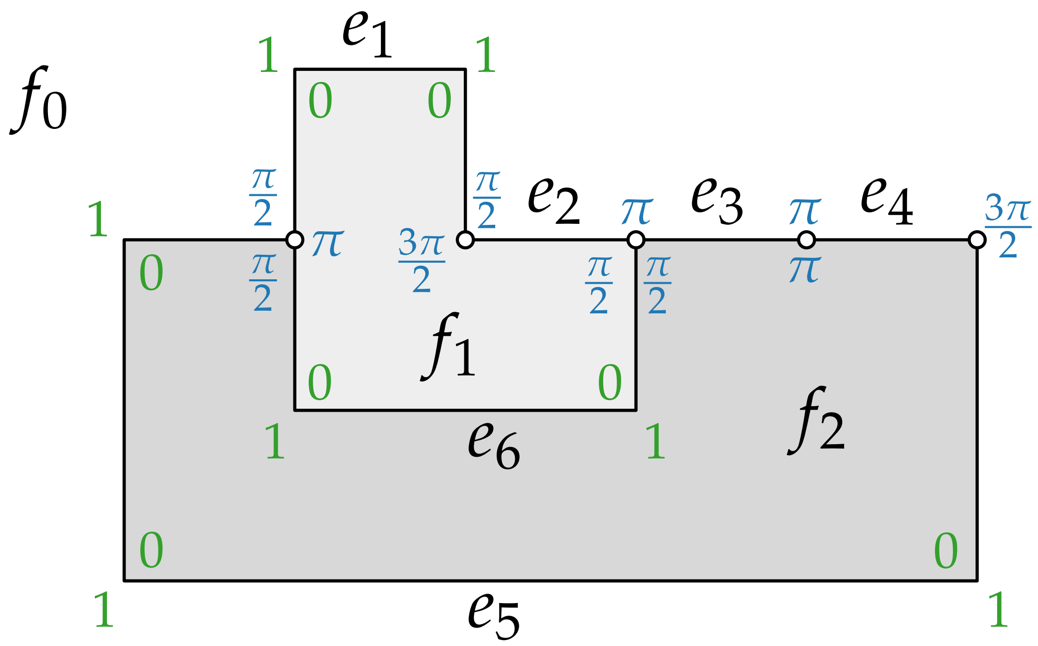
Orthogonal Representation – Example

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

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

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



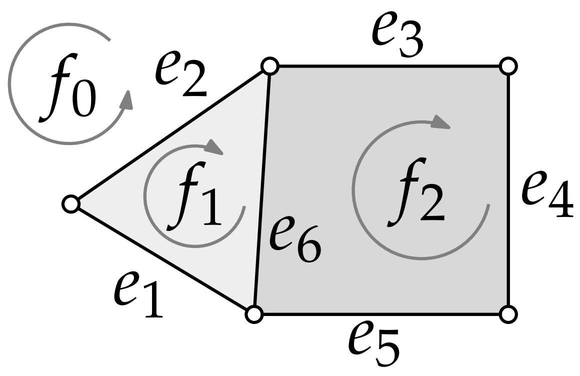


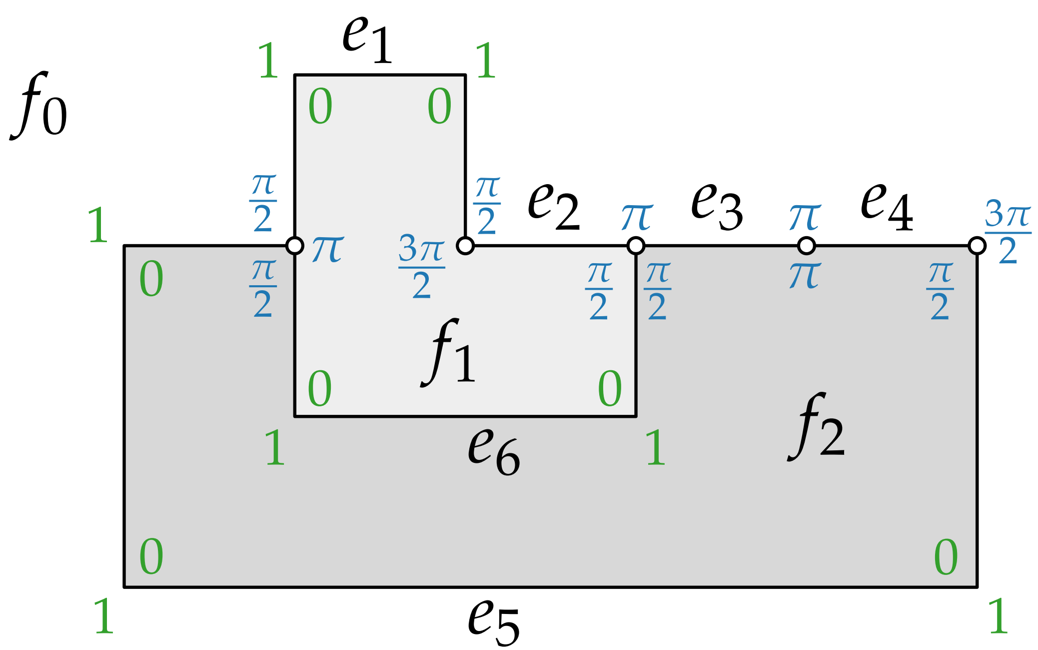
Orthogonal Representation – Example

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

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

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



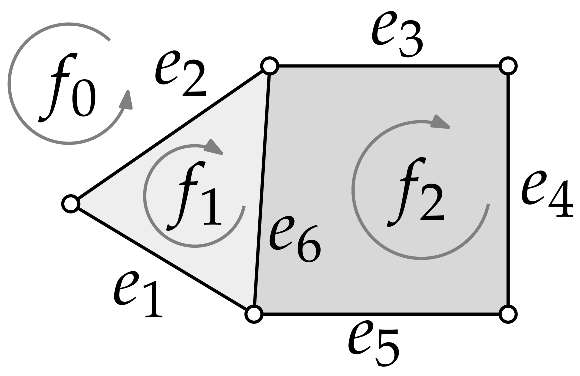


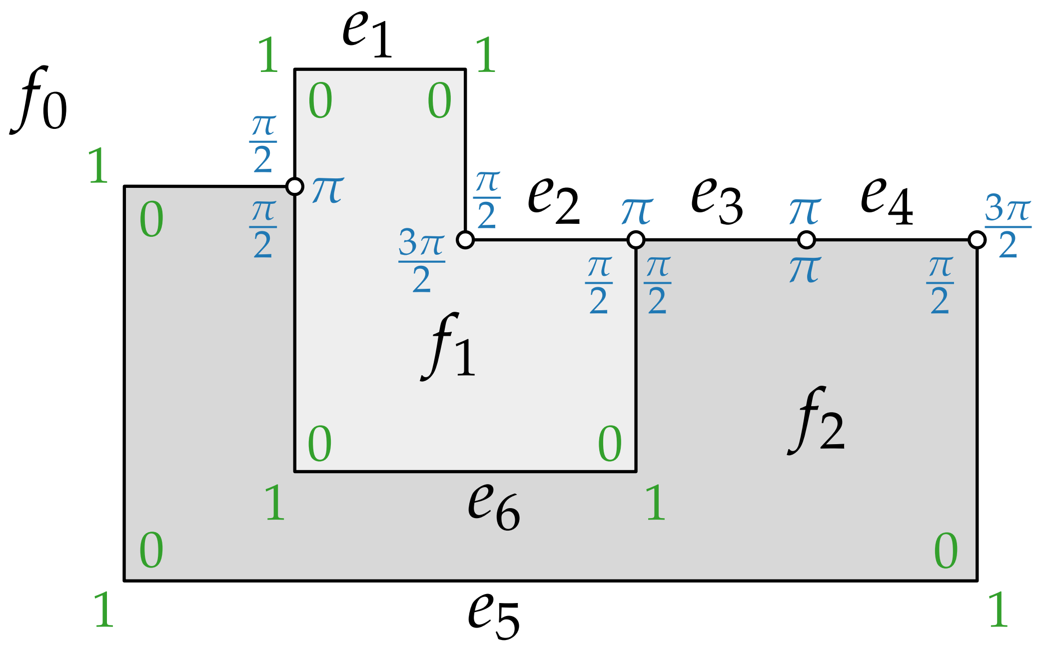
Orthogonal Representation – Example

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

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

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



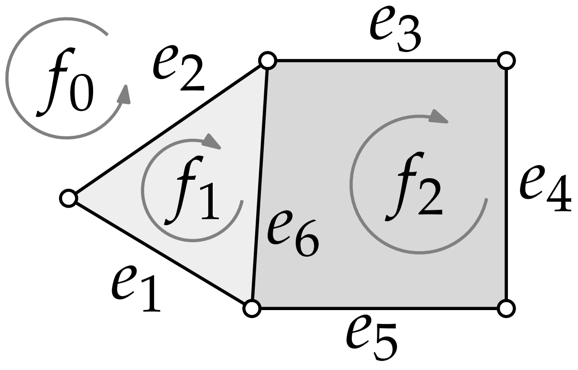


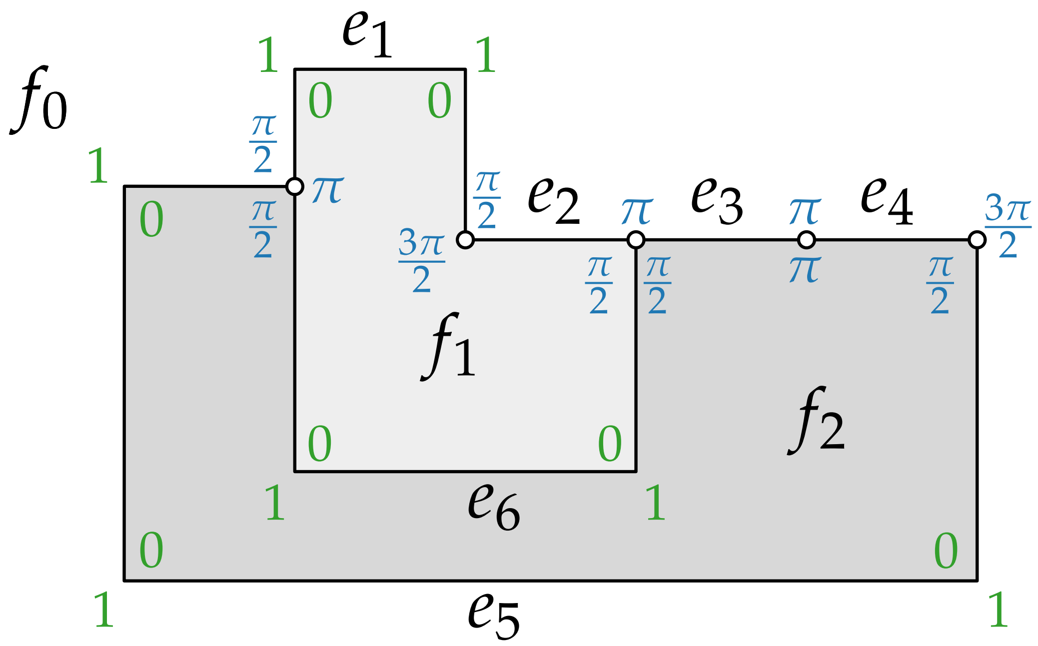
Orthogonal Representation – Example

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

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

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

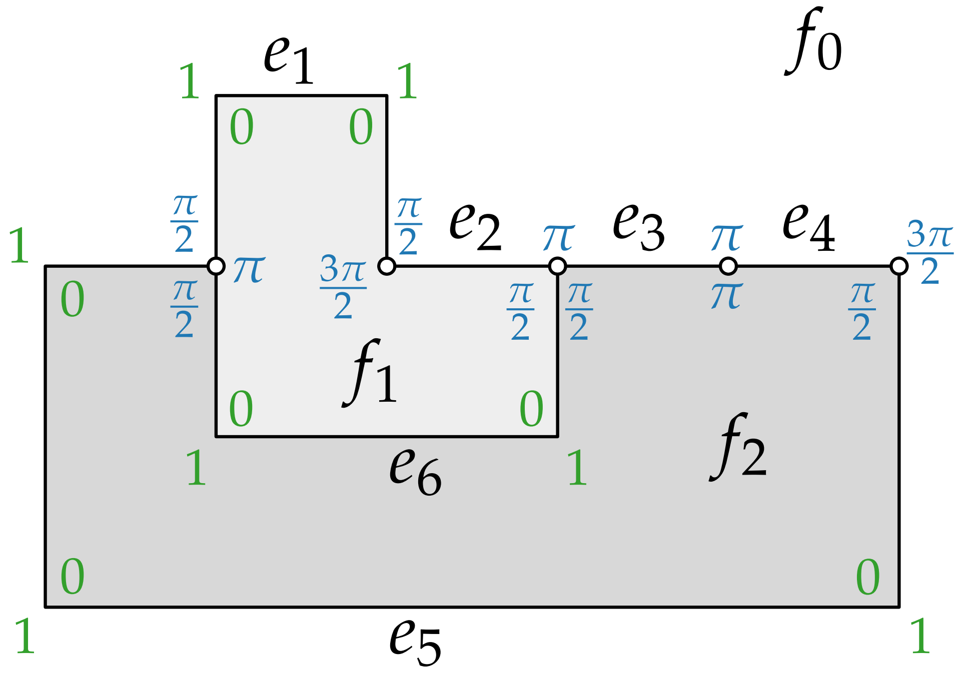




Concrete coordinates are not fixed yet!

Correctness of an Orthogonal Representation

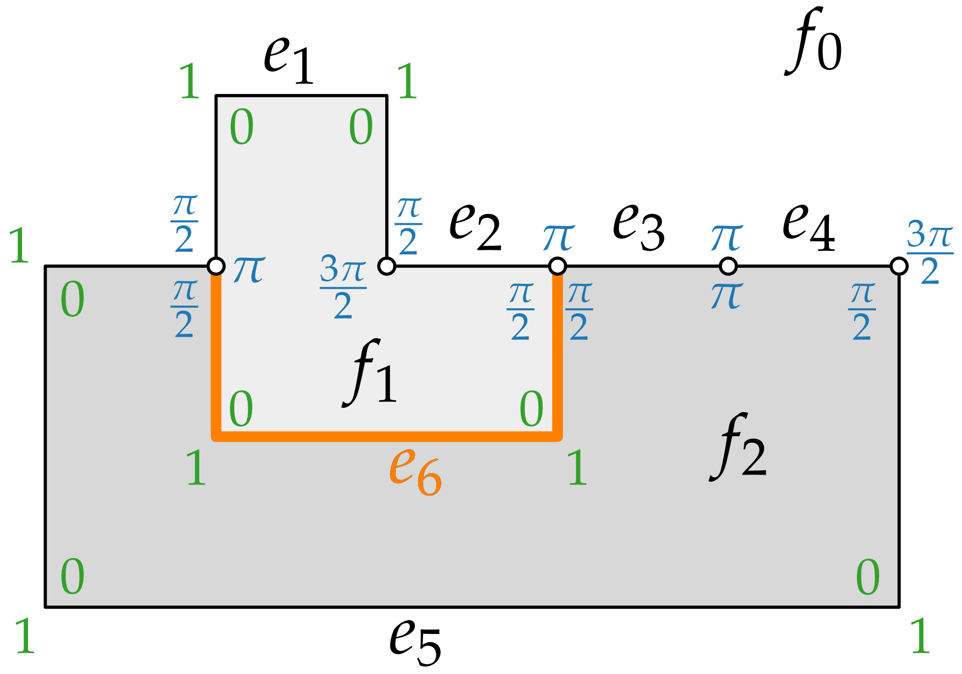
(H1) $H(G)$ corresponds to F, f_0.



Correctness of an Orthogonal Representation

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

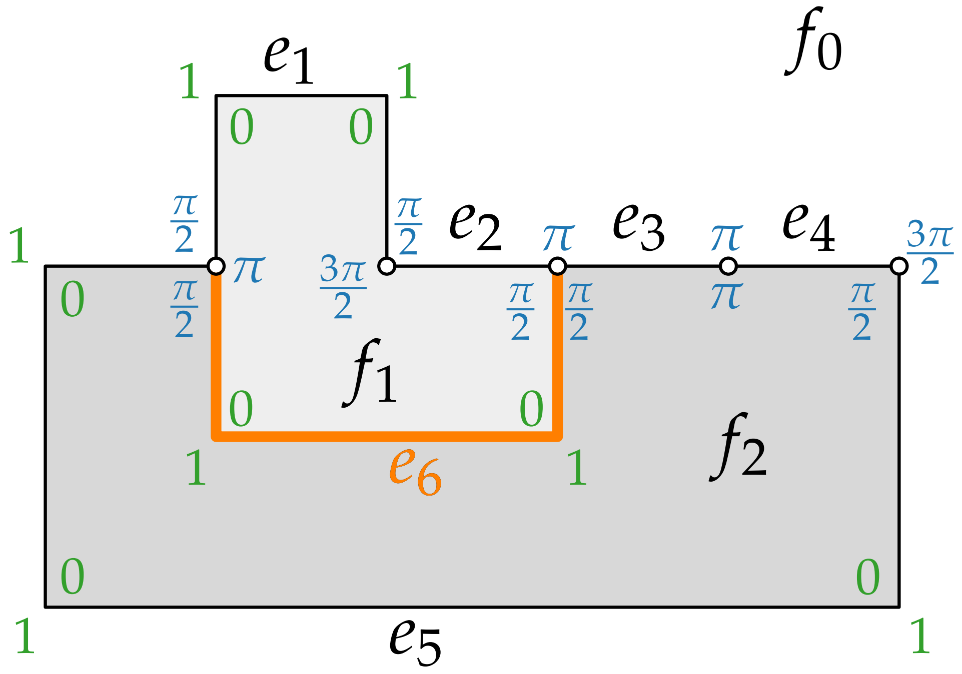
(H2) For each **edge** $\{u, v\}$ shared by faces f and g



Correctness of an Orthogonal Representation

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

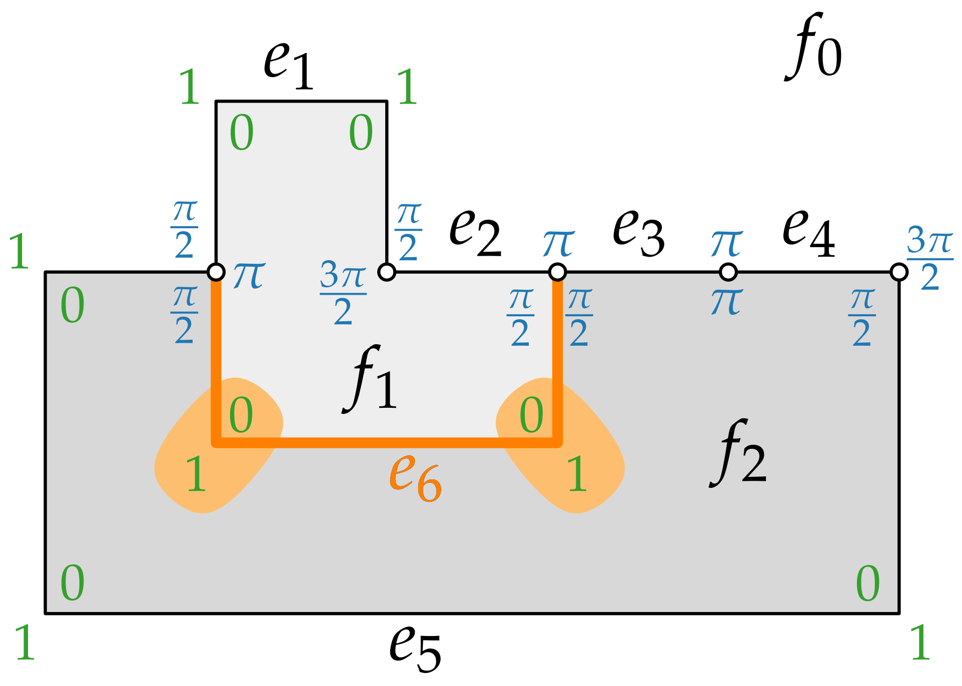
(H2) For each **edge** $\{u, v\}$ shared by faces f and g with $((u,v), \delta_1, \alpha_1) \in H(f)$ and $((v,u), \delta_2, \alpha_2) \in H(g)$



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 $((v,u), \delta_2, \alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

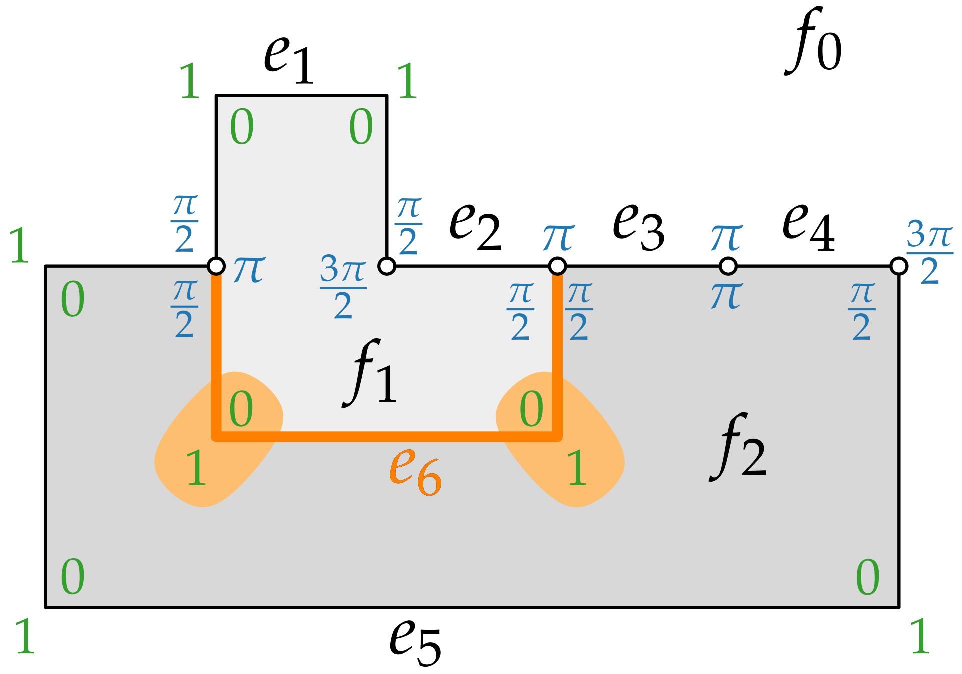


Correctness of an Orthogonal Representation

(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 $((v,u),\delta_2,\alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

(H3) Let $|\delta|_0$ (resp. $|\delta|_1$) be the number of zeros (resp. ones) in δ and $r = (e,\delta,\alpha)$.

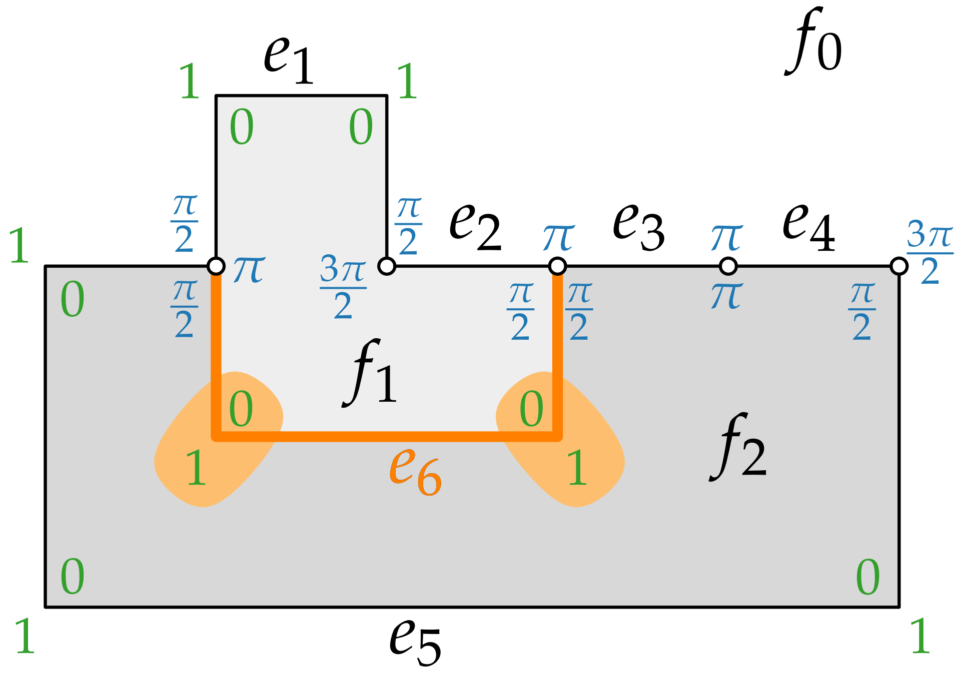


Correctness of an Orthogonal Representation

(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 $((v, u), \delta_2, \alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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



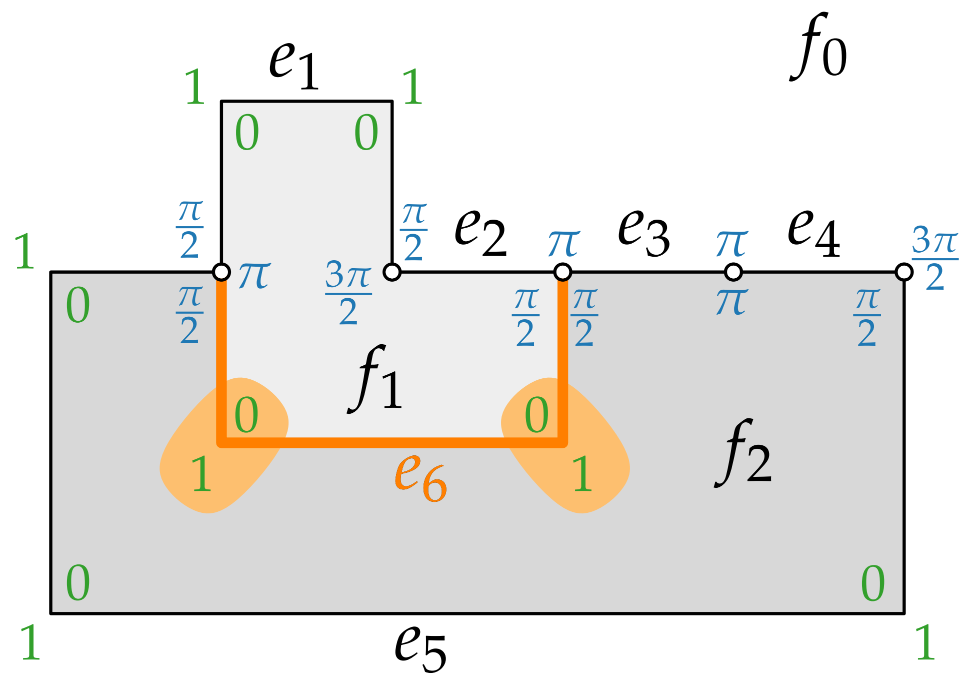
Correctness of an Orthogonal Representation

(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 $((v, u), \delta_2, \alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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

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



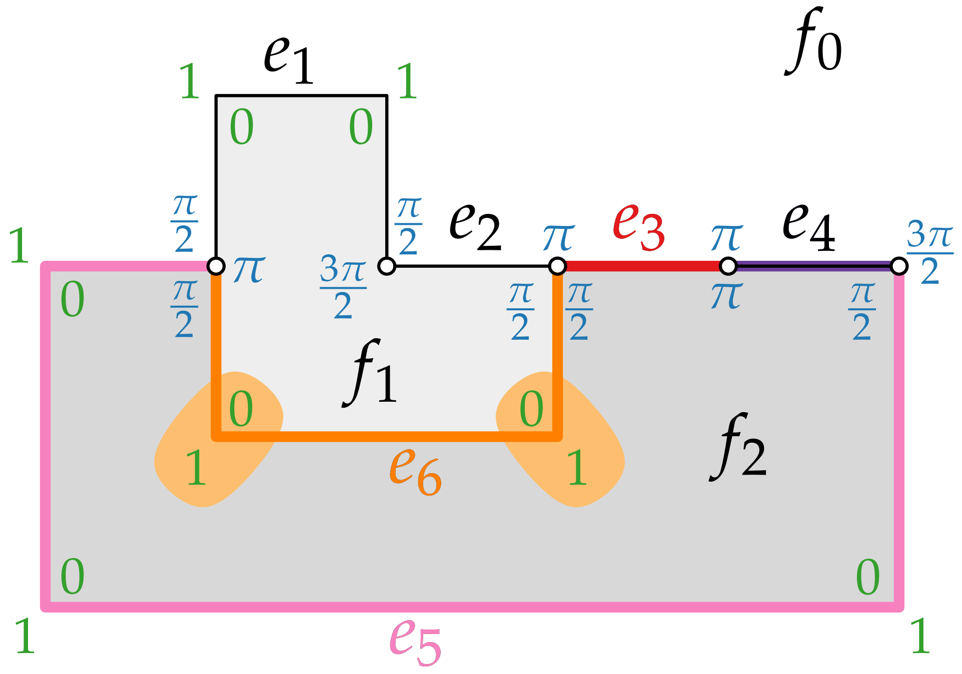
Correctness of an Orthogonal Representation

(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 $((v,u), \delta_2, \alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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

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



$C(e_3) = \quad - \quad + 2 - \quad =$

$C(e_4) = \quad - \quad + 2 - \quad =$

$C(e_5) = \quad - \quad + 2 - \quad =$

$C(e_6) = \quad - \quad + 2 - \quad =$

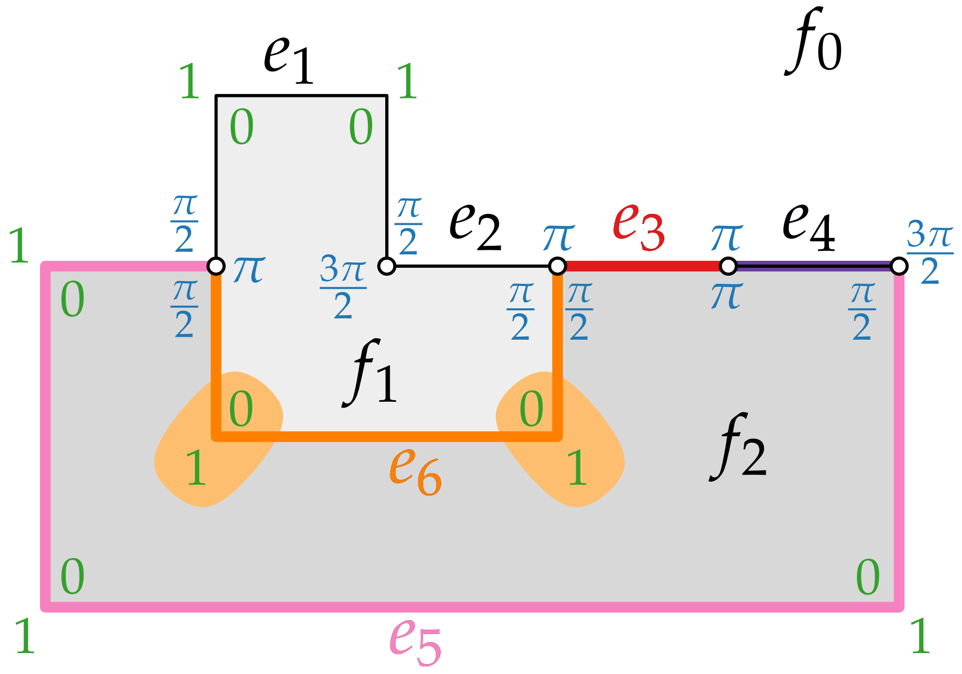
Correctness of an Orthogonal Representation

(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 $((v,u),\delta_2,\alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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

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



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

$C(e_4) = \quad - \quad + 2 - \quad =$

$C(e_5) = \quad - \quad + 2 - \quad =$

$C(e_6) = \quad - \quad + 2 - \quad =$

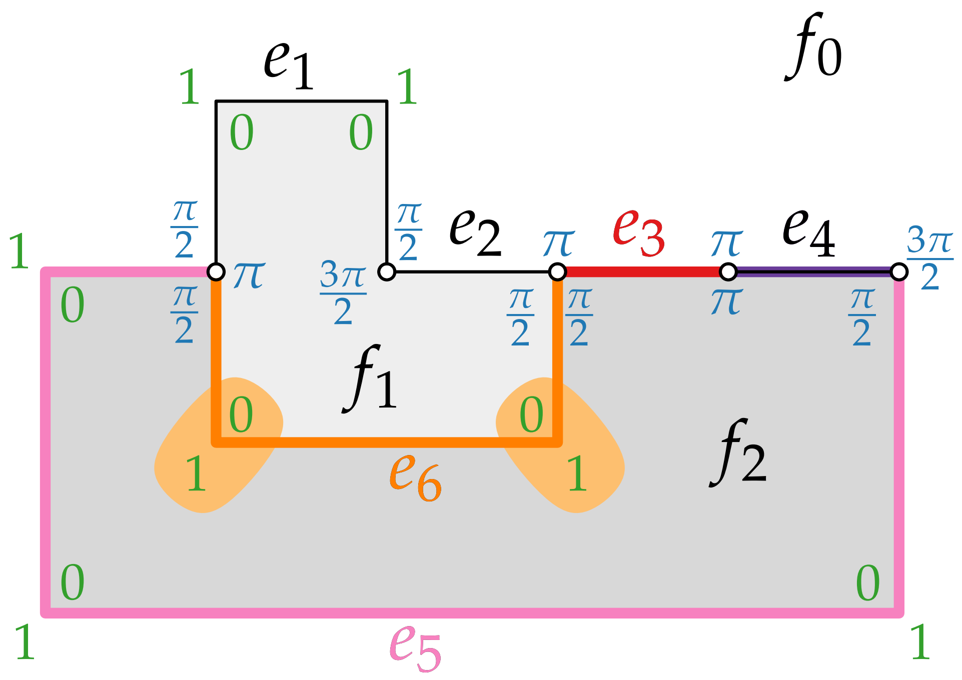
Correctness of an Orthogonal Representation

(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 $((v,u), \delta_2, \alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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

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



$C(e_3) = 0 - 0 + 2 - \qquad =$

$C(e_4) = \qquad - \qquad + 2 - \qquad =$

$C(e_5) = \qquad - \qquad + 2 - \qquad =$

$C(e_6) = \qquad - \qquad + 2 - \qquad =$

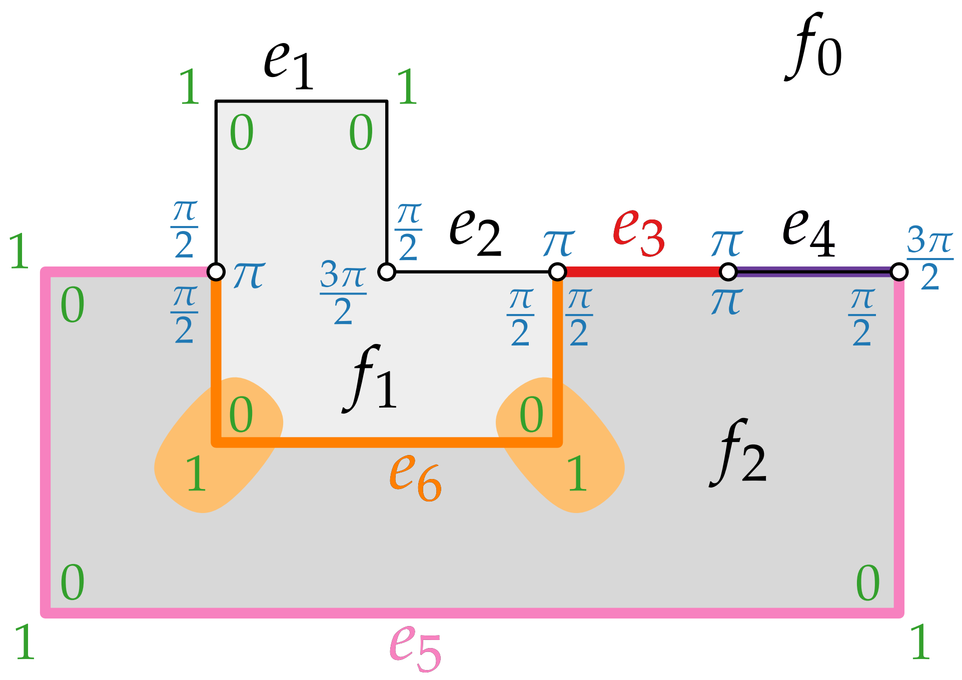
Correctness of an Orthogonal Representation

(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 $((v, u), \delta_2, \alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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

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



$C(e_3) = 0 - 0 + 2 - \pi \cdot \frac{2}{\pi} =$

$C(e_4) = \quad - \quad + 2 - \quad =$

$C(e_5) = \quad - \quad + 2 - \quad =$

$C(e_6) = \quad - \quad + 2 - \quad =$

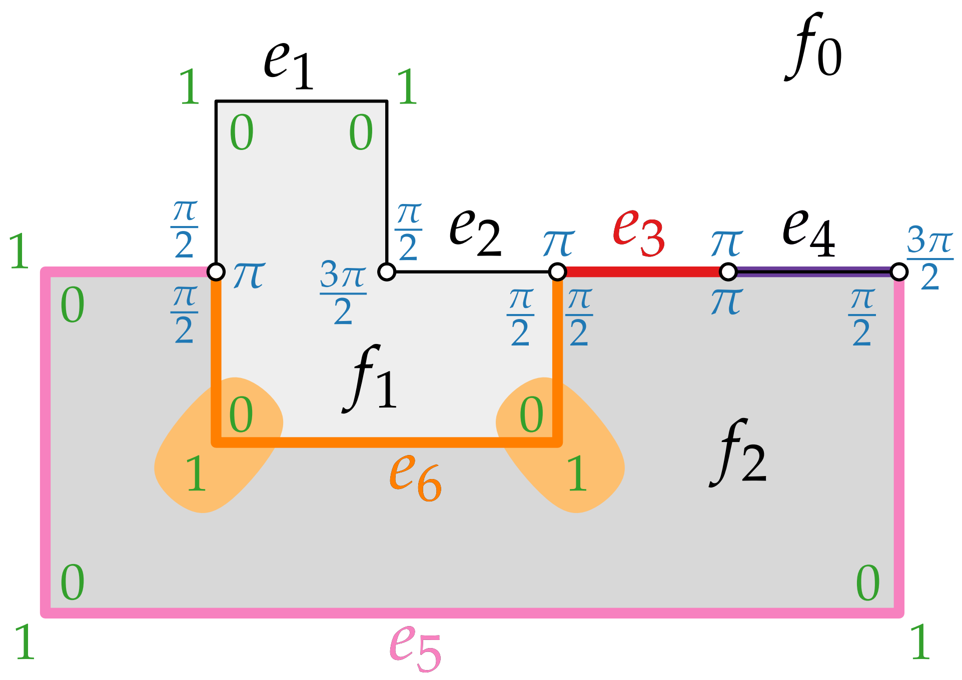
Correctness of an Orthogonal Representation

(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 $((v,u), \delta_2, \alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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

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



$C(e_3) = 0 - 0 + 2 - \pi \cdot \frac{2}{\pi} = 0$

$C(e_4) = \quad - \quad + 2 - \quad =$

$C(e_5) = \quad - \quad + 2 - \quad =$

$C(e_6) = \quad - \quad + 2 - \quad =$

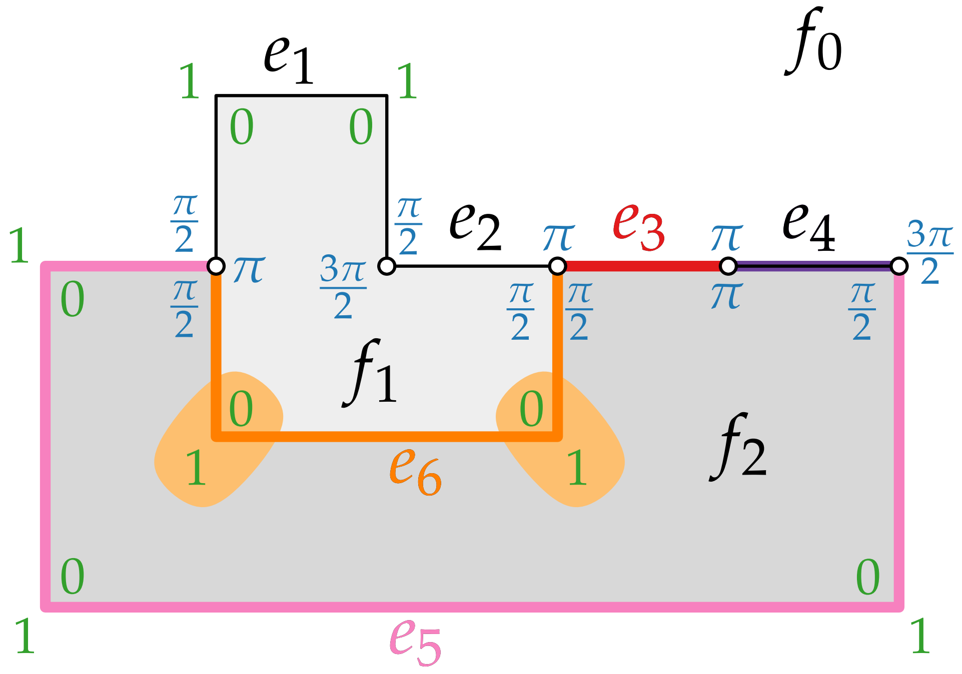
Correctness of an Orthogonal Representation

(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 $((v,u),\delta_2,\alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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

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



$C(e_3) = 0 - 0 + 2 - \pi \cdot \frac{2}{\pi} = 0$

$C(e_4) = 0 - 0 + 2 - \frac{\pi}{2} \cdot \frac{2}{\pi} =$

$C(e_5) = \quad - \quad + 2 - \quad =$

$C(e_6) = \quad - \quad + 2 - \quad =$

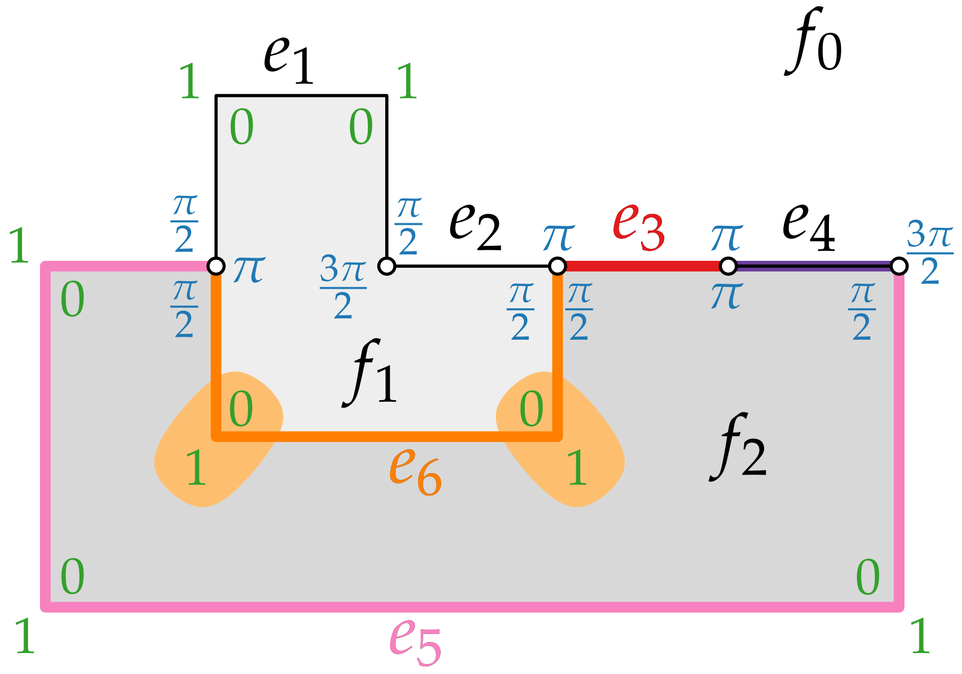
Correctness of an Orthogonal Representation

(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 $((v,u),\delta_2,\alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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

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



$C(e_3) = 0 - 0 + 2 - \pi \cdot \frac{2}{\pi} = 0$

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

$C(e_5) = \quad - \quad + 2 - \quad =$

$C(e_6) = \quad - \quad + 2 - \quad =$

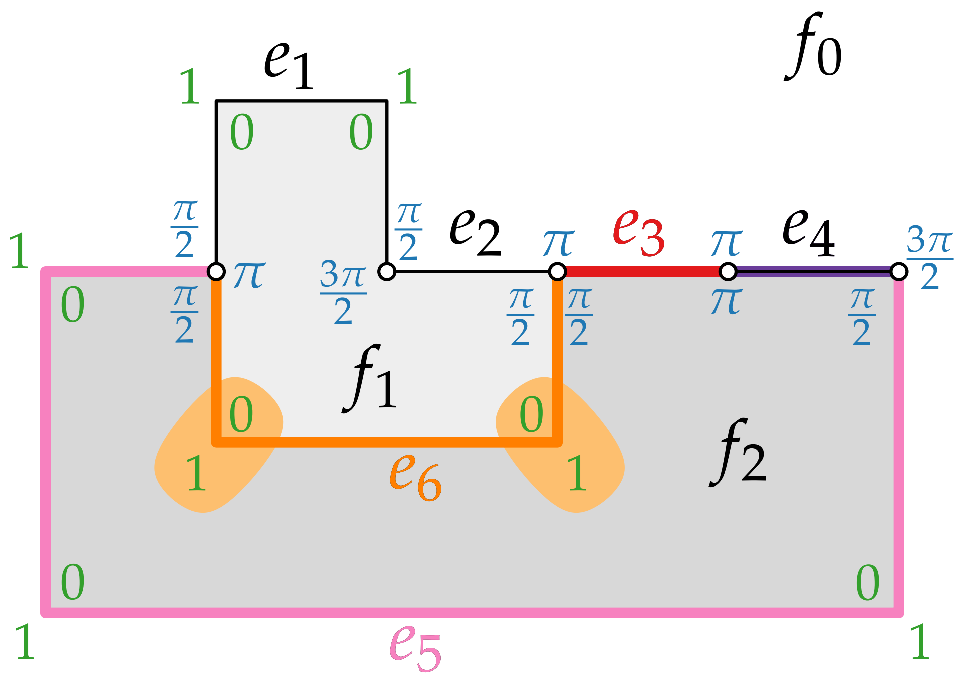
Correctness of an Orthogonal Representation

(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 $((v,u),\delta_2,\alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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

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



$C(e_3) = 0 - 0 + 2 - \pi \cdot \frac{2}{\pi} = 0$

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

$C(e_5) = 3 - 0 + 2 - \qquad =$

$C(e_6) = \qquad - \qquad + 2 - \qquad =$

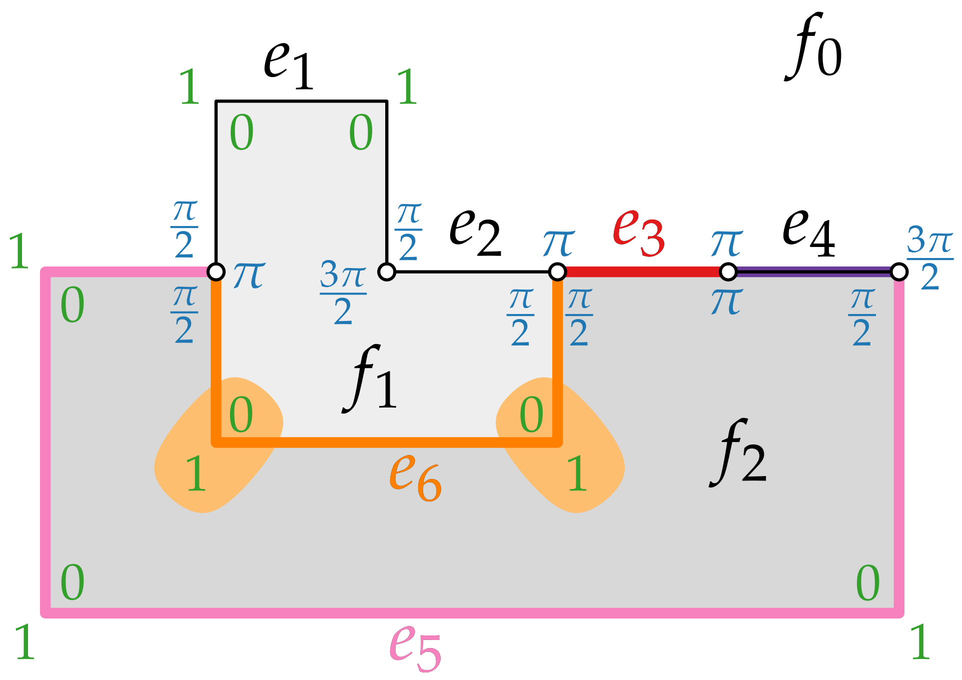
Correctness of an Orthogonal Representation

(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 $((v,u), \delta_2, \alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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

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



$C(e_3) = 0 - 0 + 2 - \pi \cdot \frac{2}{\pi} = 0$

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

$C(e_5) = 3 - 0 + 2 - \frac{\pi}{2} \cdot \frac{2}{\pi} =$

$C(e_6) = \quad - \quad + 2 - \quad =$

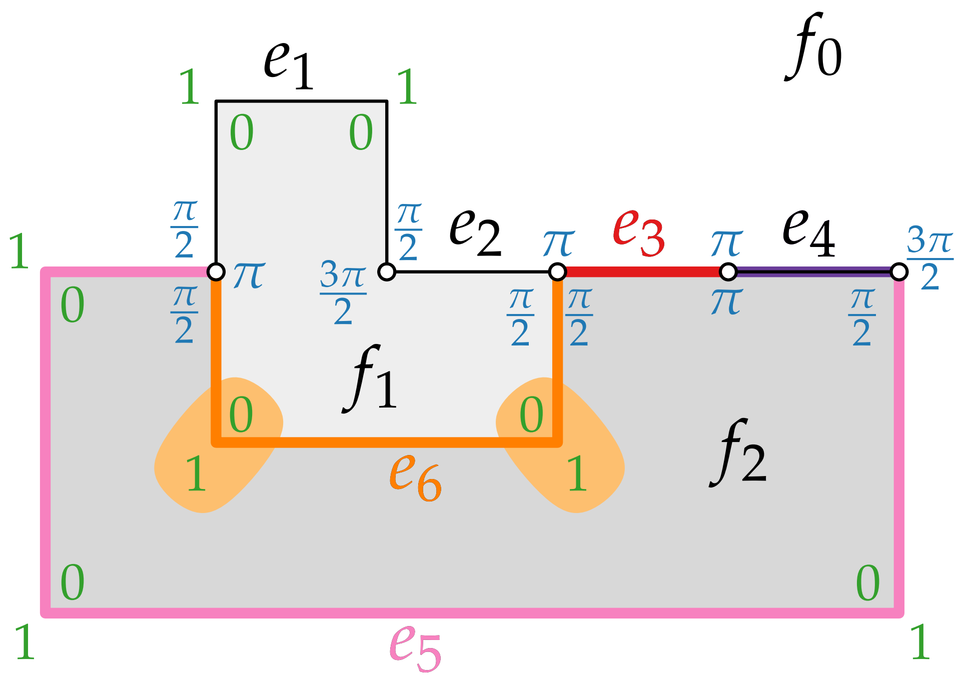
Correctness of an Orthogonal Representation

(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 $((v,u), \delta_2, \alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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

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



$C(e_3) = 0 - 0 + 2 - \pi \cdot \frac{2}{\pi} = 0$

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

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

$C(e_6) = \quad - \quad + 2 - \quad =$

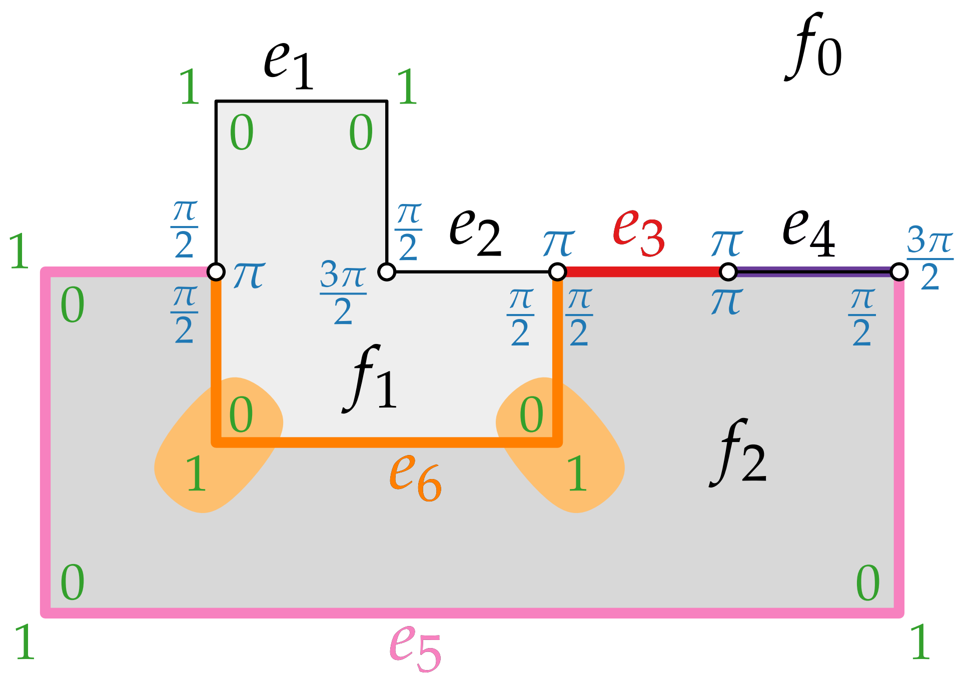
Correctness of an Orthogonal Representation

(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 $((v,u),\delta_2,\alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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

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



$$C(e_3) = 0 - 0 + 2 - \pi \cdot \frac{2}{\pi} = 0$$

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

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

$$C(e_6) = 0 - 2 + 2 - \frac{\pi}{2} \cdot \frac{2}{\pi} =$$

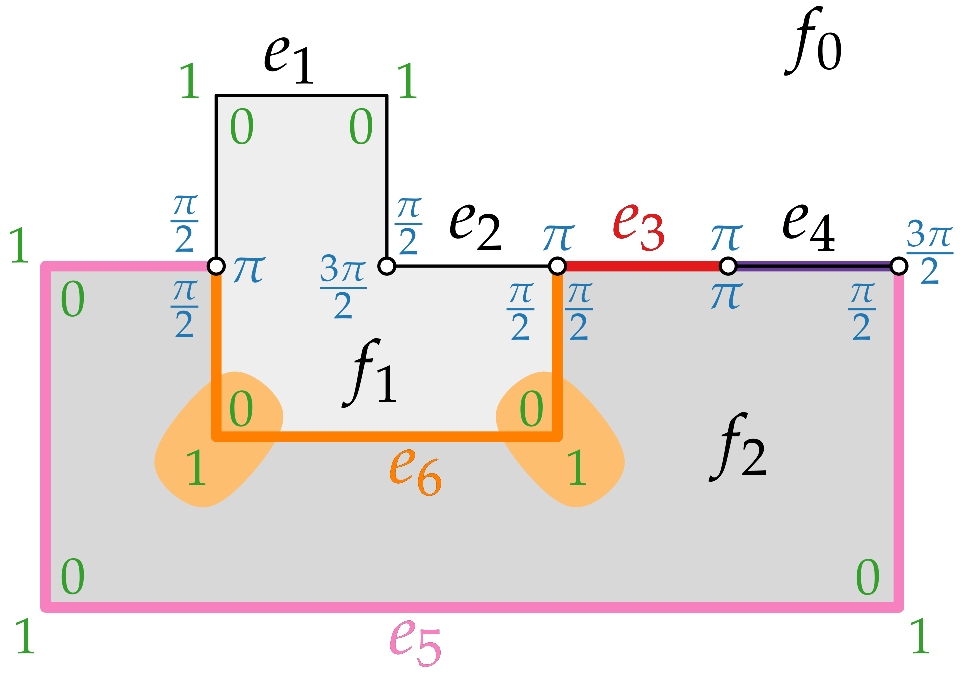
Correctness of an Orthogonal Representation

(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 $((v,u),\delta_2,\alpha_2) \in H(g)$ sequence δ_1 is reversed and inverted δ_2.

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

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



$$C(e_3) = 0 - 0 + 2 - \pi \cdot \frac{2}{\pi} = 0$$

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

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

$$C(e_6) = 0 - 2 + 2 - \frac{\pi}{2} \cdot \frac{2}{\pi} = -1$$

Correctness of an Orthogonal Representation

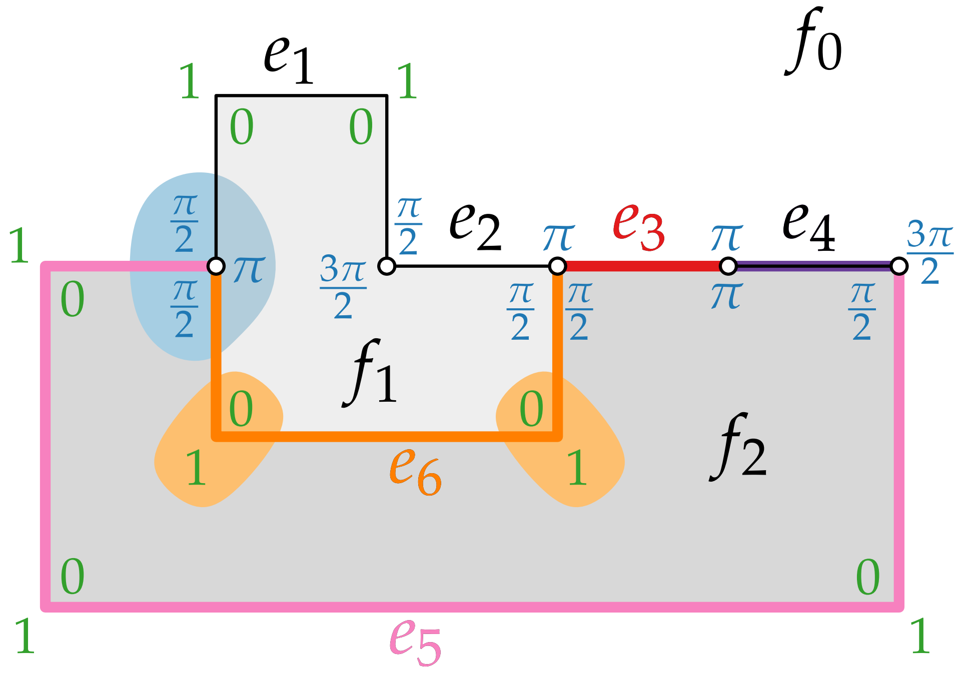
(H1) $H(G)$ corresponds to F, f_0.

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

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



$$C(e_3) = 0 - 0 + 2 - \pi \cdot \frac{2}{\pi} = 0$$

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

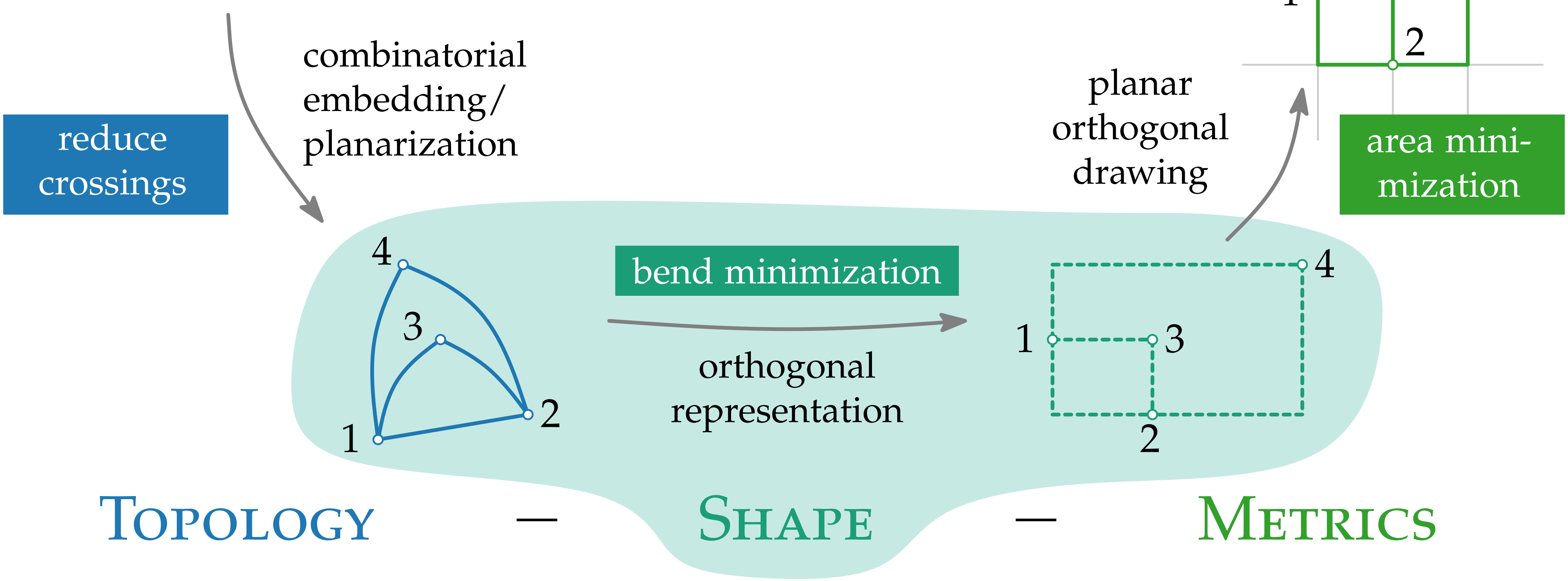
$$C(e_5) = 3 - 0 + 2 - \frac{\pi}{2} \cdot \frac{2}{\pi} = 4$$

$$C(e_6) = 0 - 2 + 2 - \frac{\pi}{2} \cdot \frac{2}{\pi} = -1$$

Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

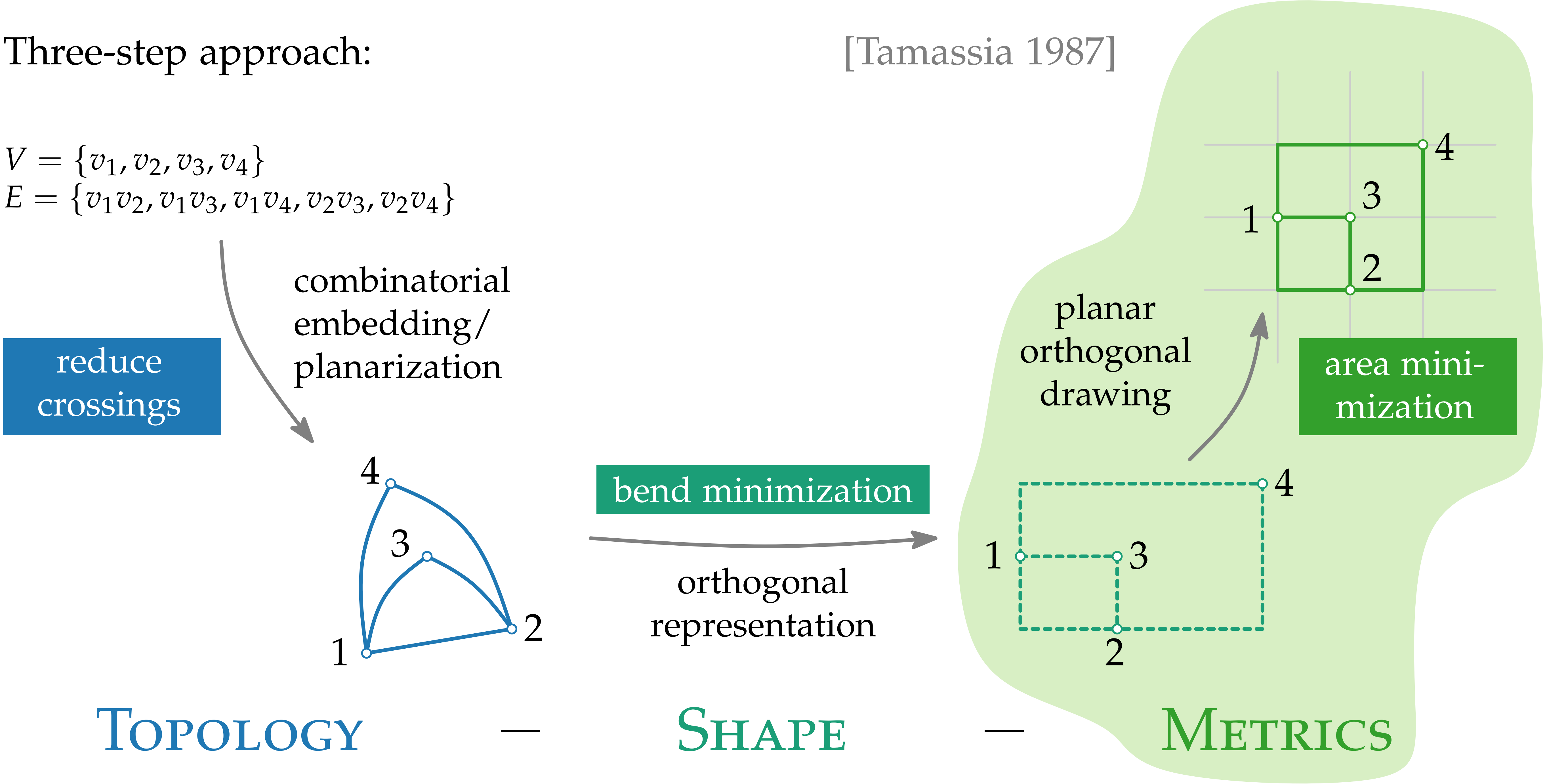
$V = \{v_1, v_2, v_3, v_4\}$
$E = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$



Topology – Shape – Metrics

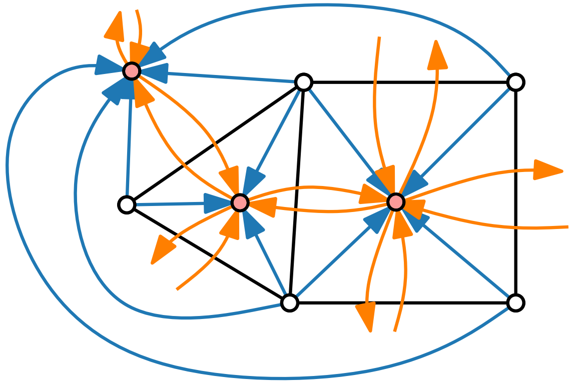
Three-step approach: [Tamassia 1987]

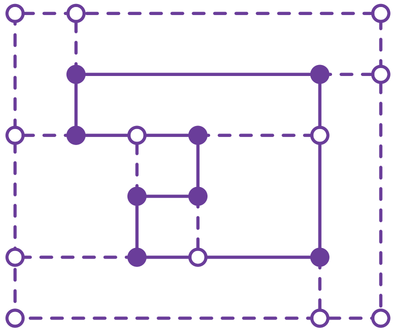
$V = \{v_1, v_2, v_3, v_4\}$
$E = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$



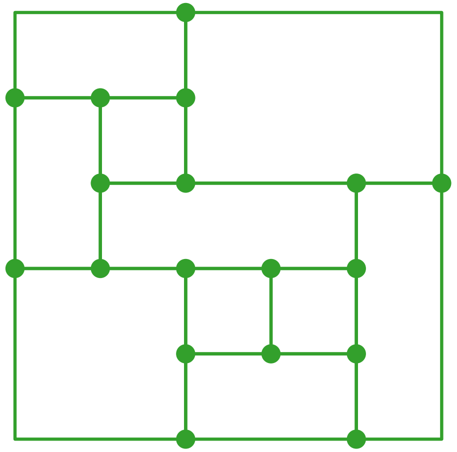
Universität Trier

Visualization of Graphs





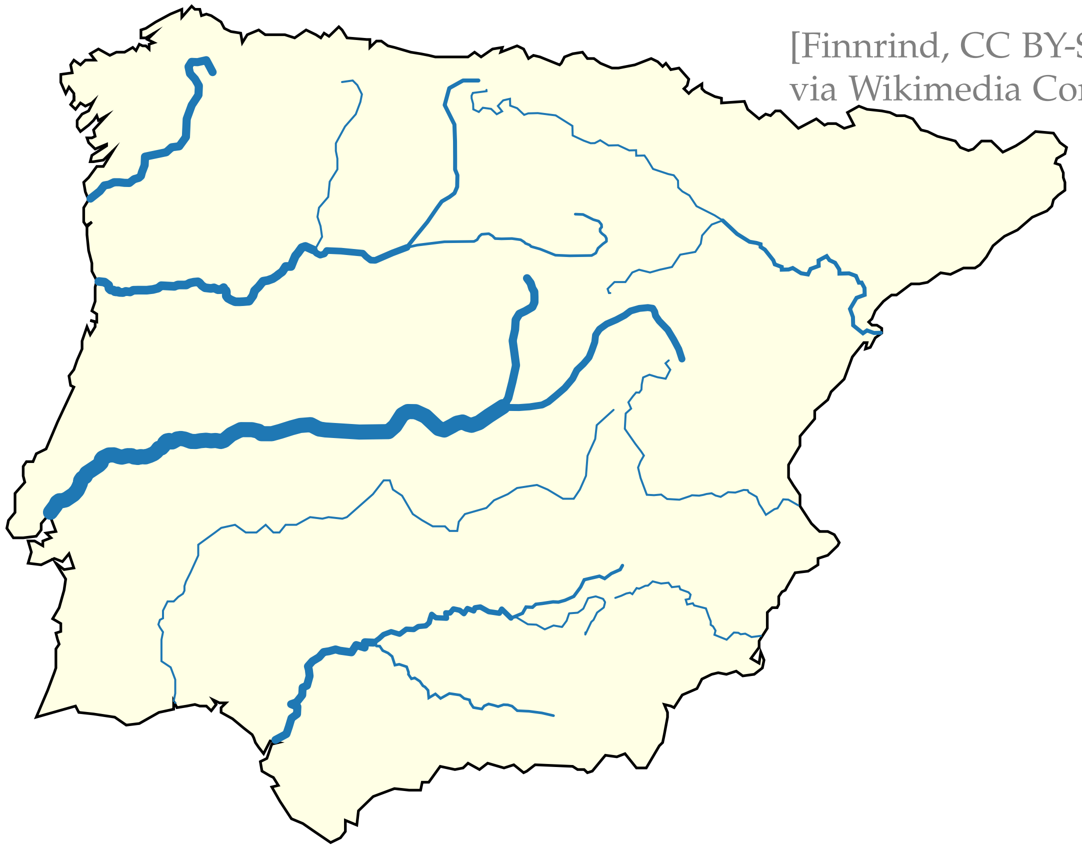
Lecture 6:
Orthogonal Layouts



Part III:
Flow Networks

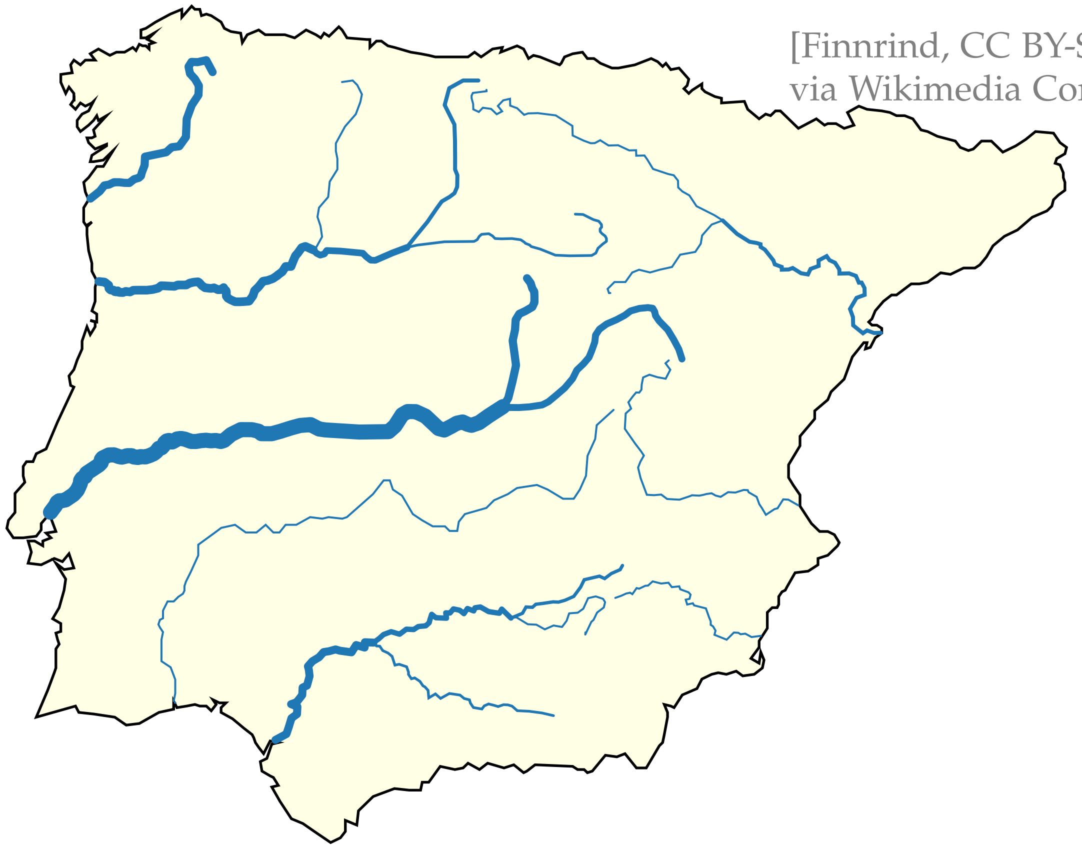
Philipp Kindermann

Flow Networks



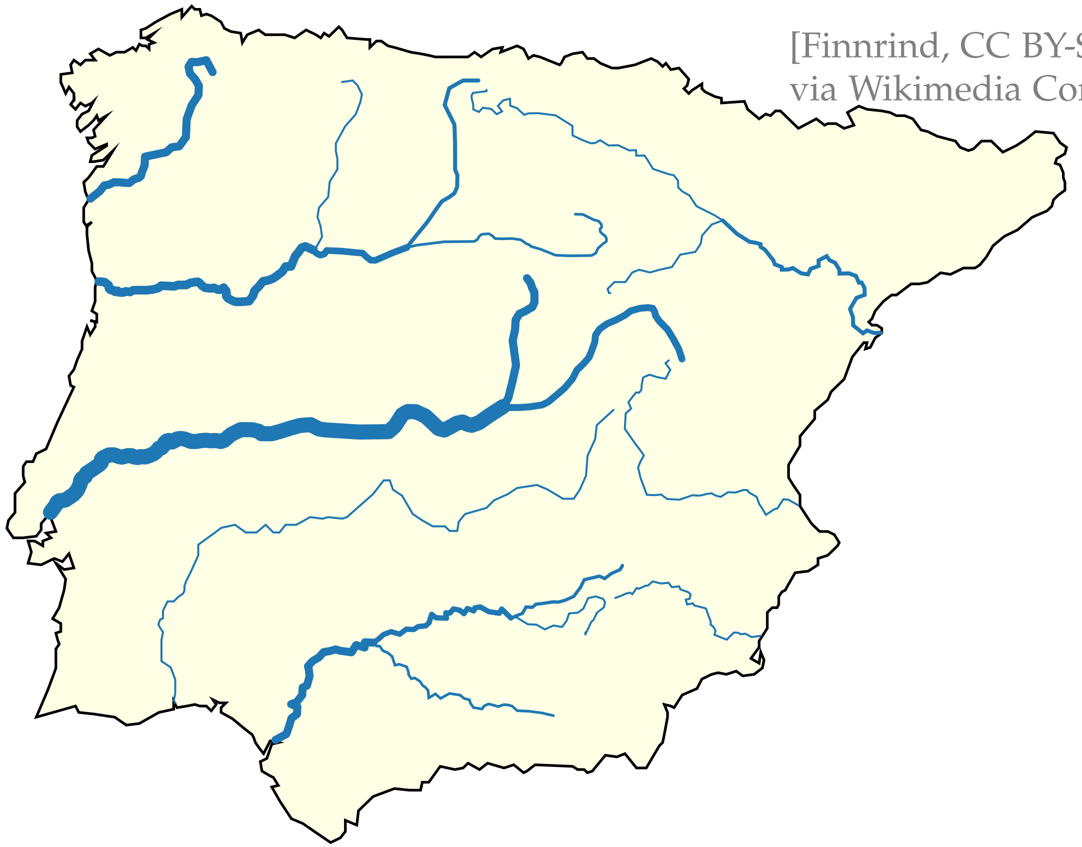
[Finnrind, CC BY-SA 3.0,
via Wikimedia Commons]

Flow Networks



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

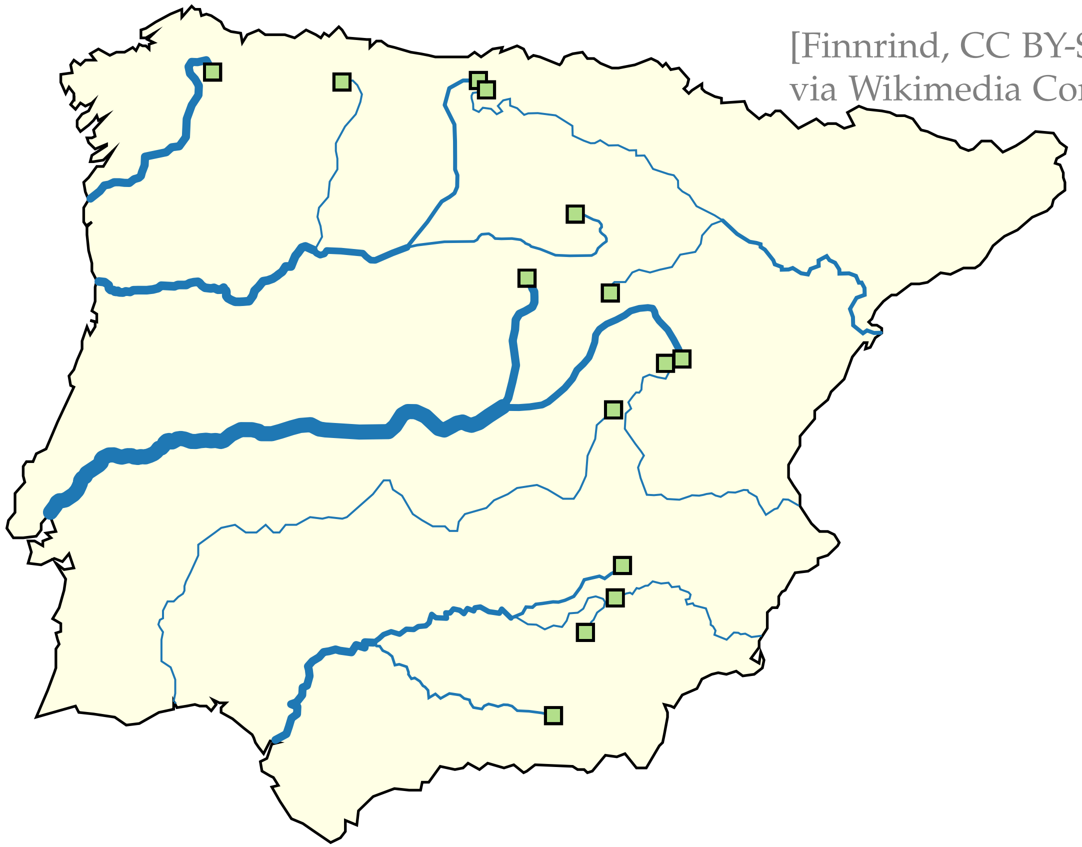
Flow Networks



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

- directed graph $G = (V, E)$

Flow Networks

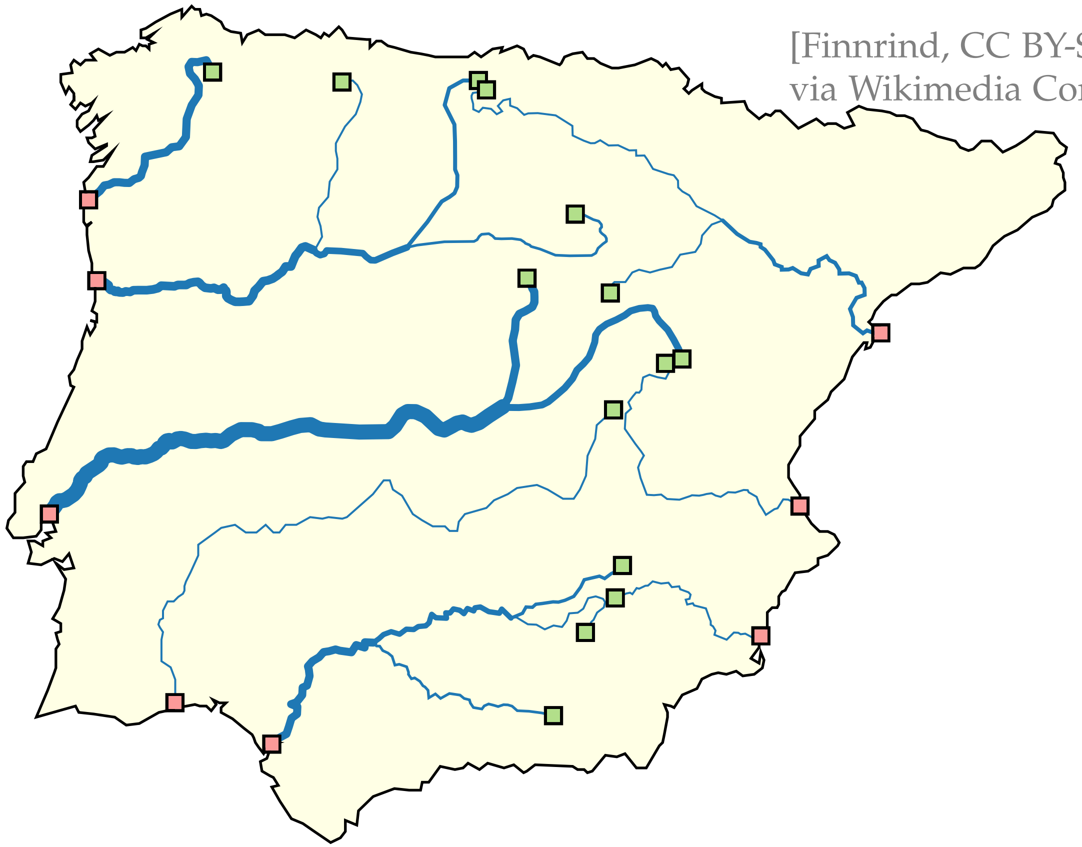


[Finnrind, CC BY-SA 3.0, via Wikimedia Commons]

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

- directed graph $G = (V, E)$
- *sources* $S \subseteq V$

Flow Networks

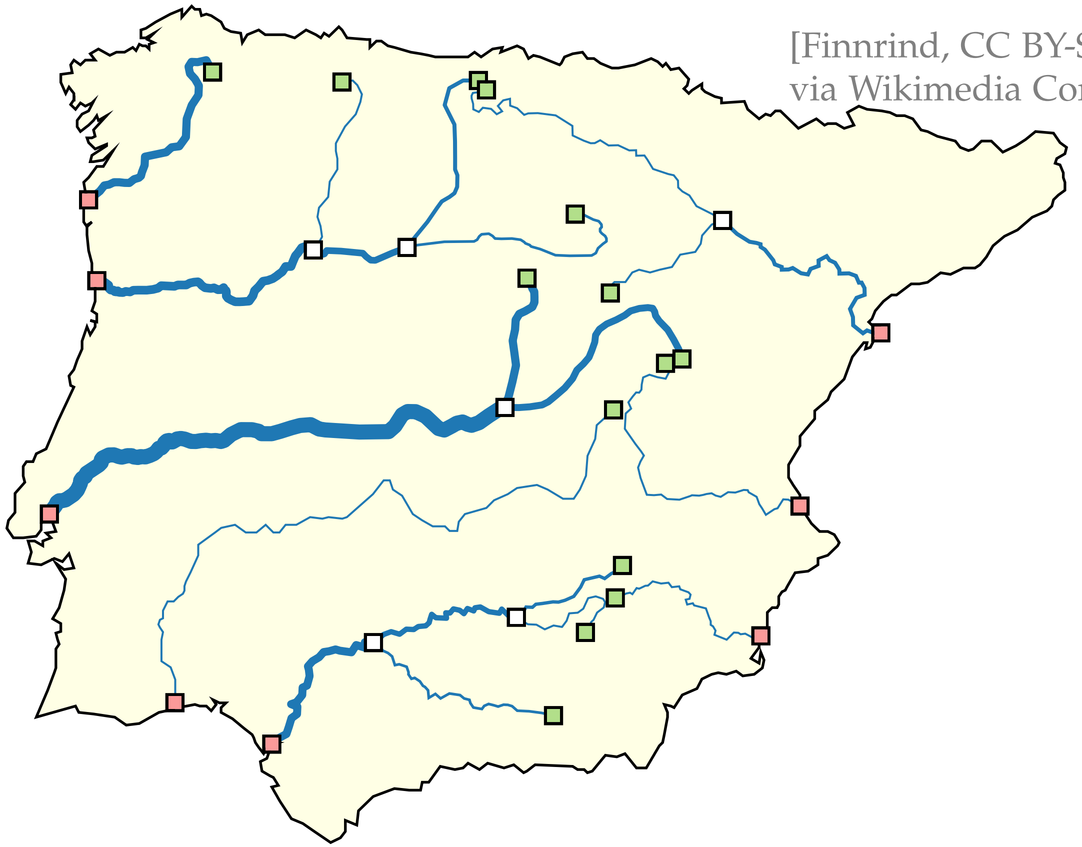


[Finnrind, CC BY-SA 3.0, via Wikimedia Commons]

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

- directed graph $G = (V, E)$
- *sources* $S \subseteq V$, *sinks* $T \subseteq V$

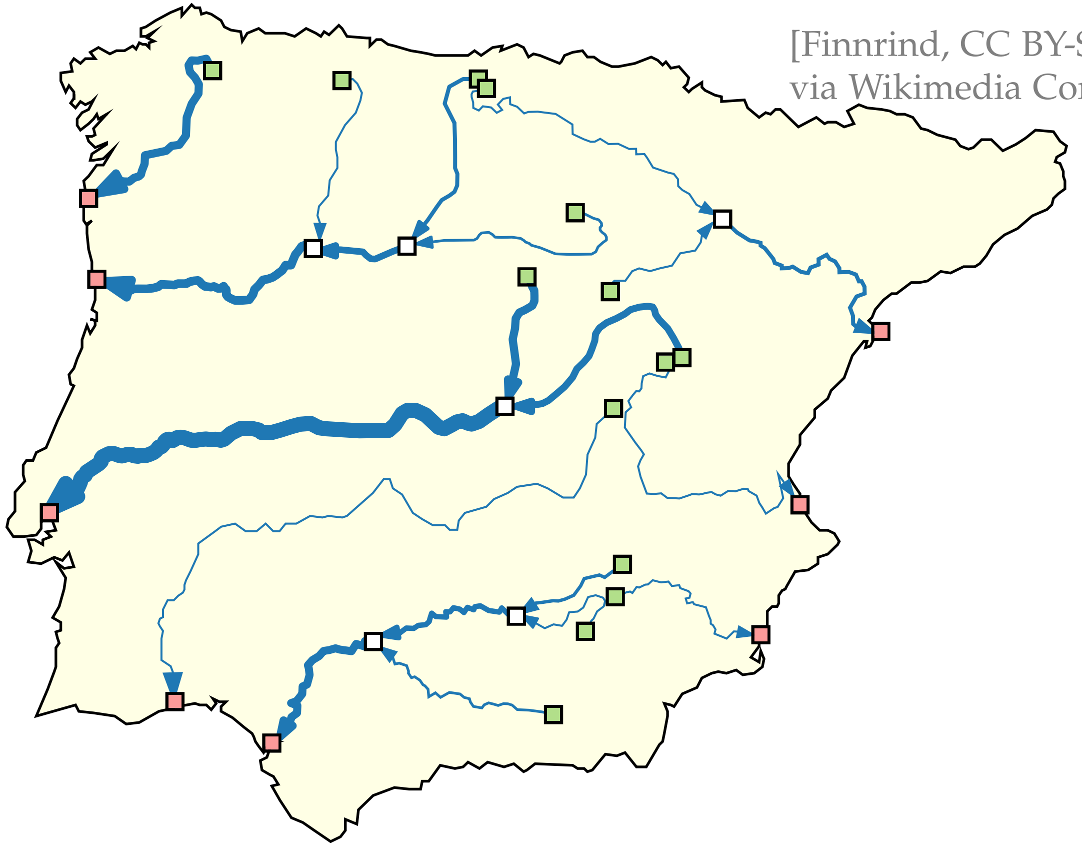
Flow Networks



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

- directed graph $G = (V, E)$
- *sources* $S \subseteq V$, *sinks* $T \subseteq V$

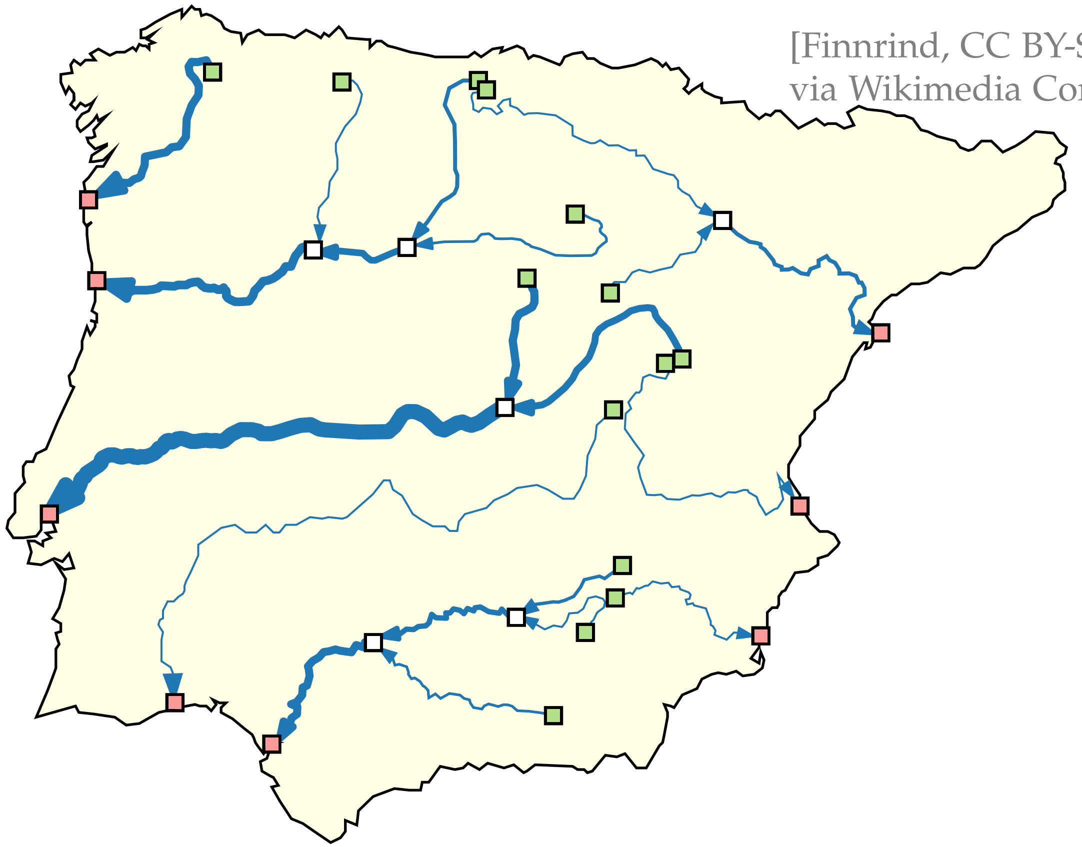
Flow Networks



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

- directed graph $G = (V, E)$
- *sources* $S \subseteq V$, *sinks* $T \subseteq V$

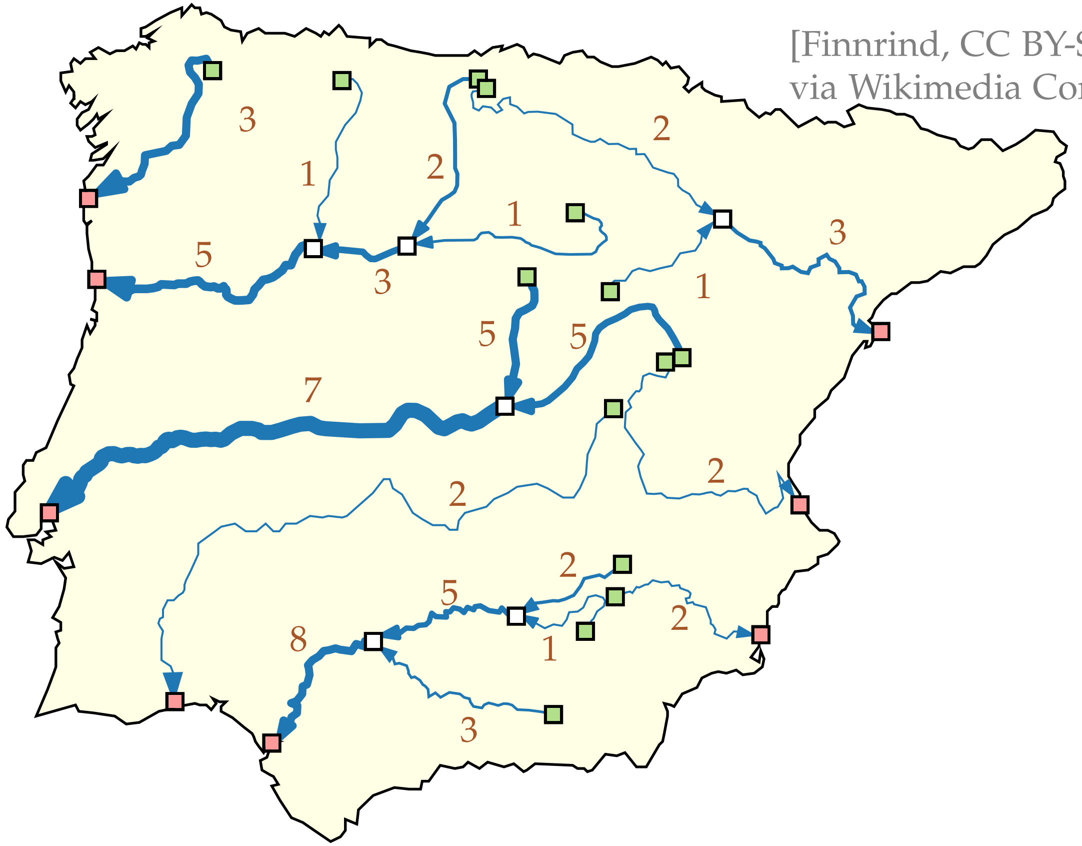
Flow Networks



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

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

Flow Networks

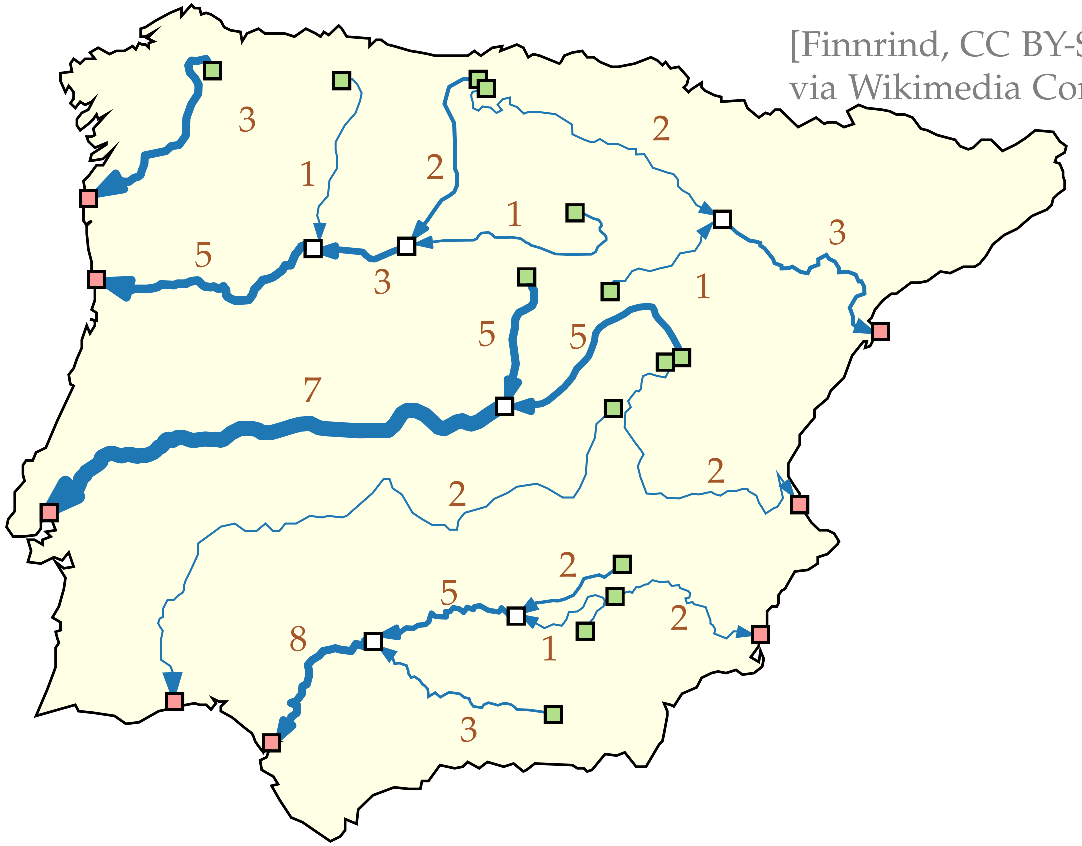


[Finnrind, CC BY-SA 3.0, via Wikimedia Commons]

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

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

Flow Networks

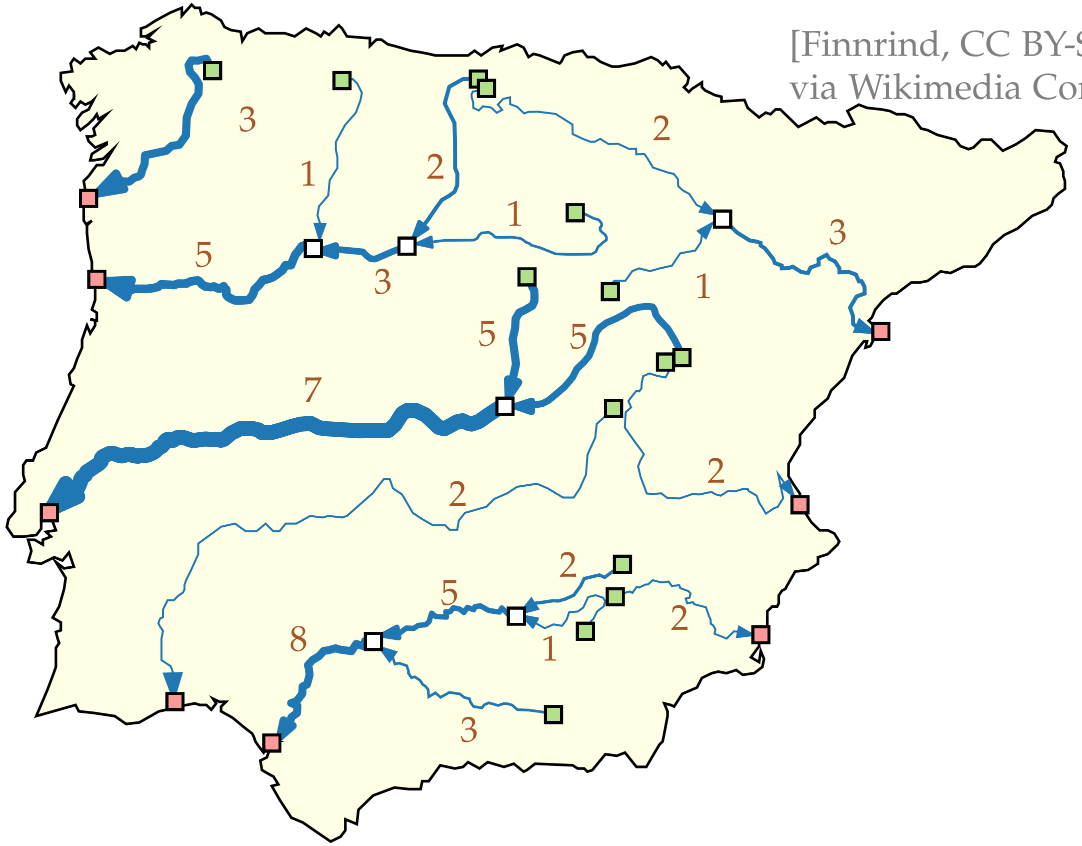


Flow network $(G = (V, E); S, T; u)$ with

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

A function $X \colon E \to \mathbb{R}_0^+$ is called **S-T-flow**, if:

Flow Networks



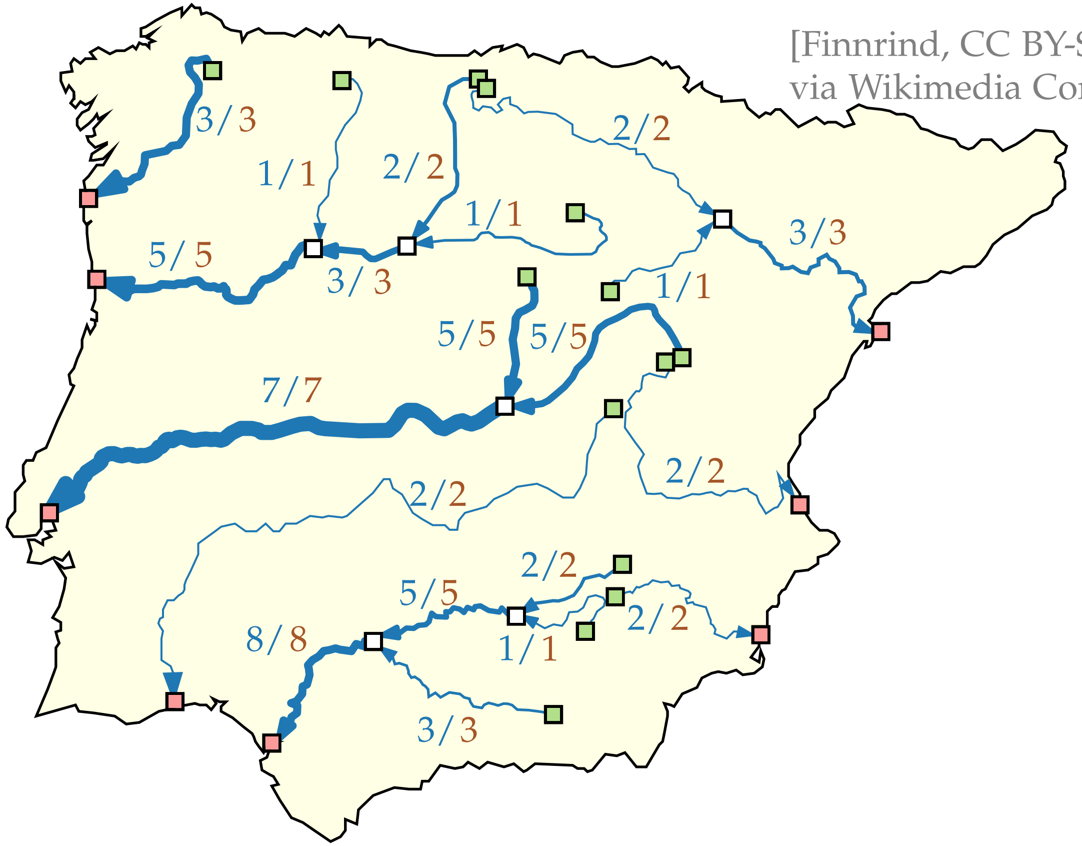
Flow network $(G = (V, E); S, T; u)$ with

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

A function $X\colon E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

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

Flow Networks



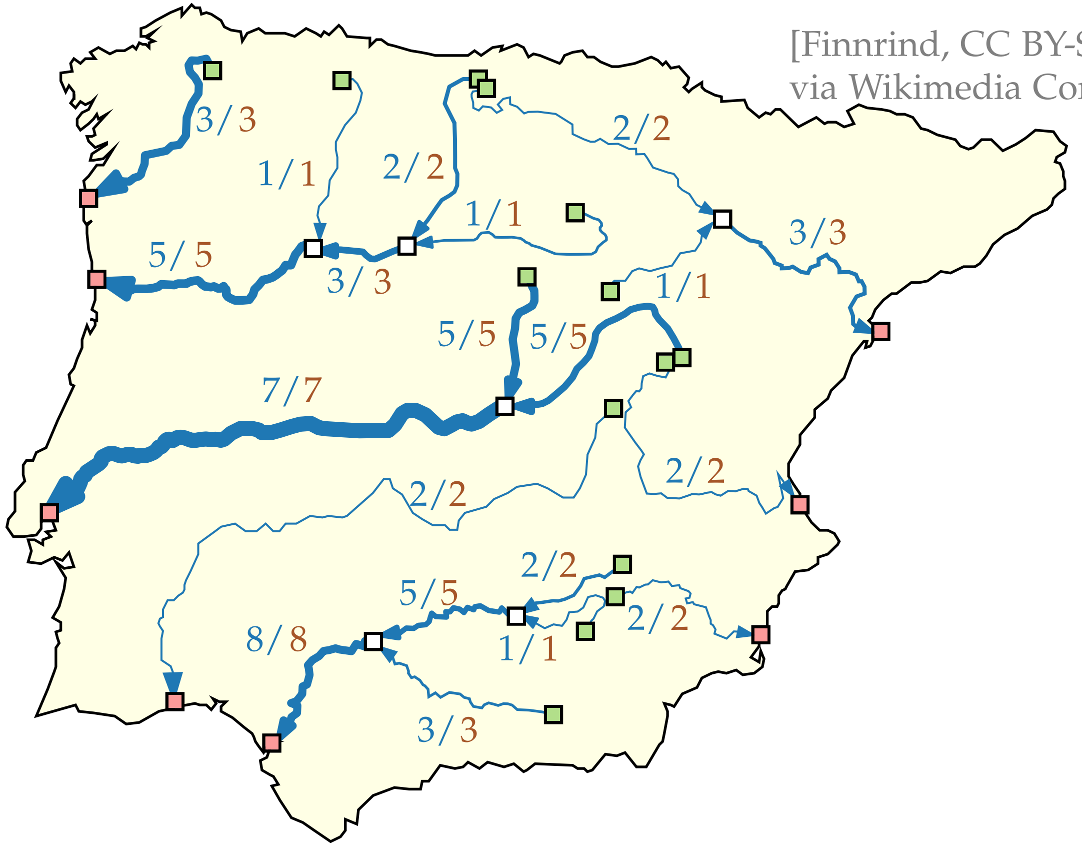
Flow network $(G = (V, E); S, T; u)$ with

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

A function $X \colon E \to \mathbb{R}_0^+$ is called **S-T-flow**, if:

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

Flow Networks



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

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

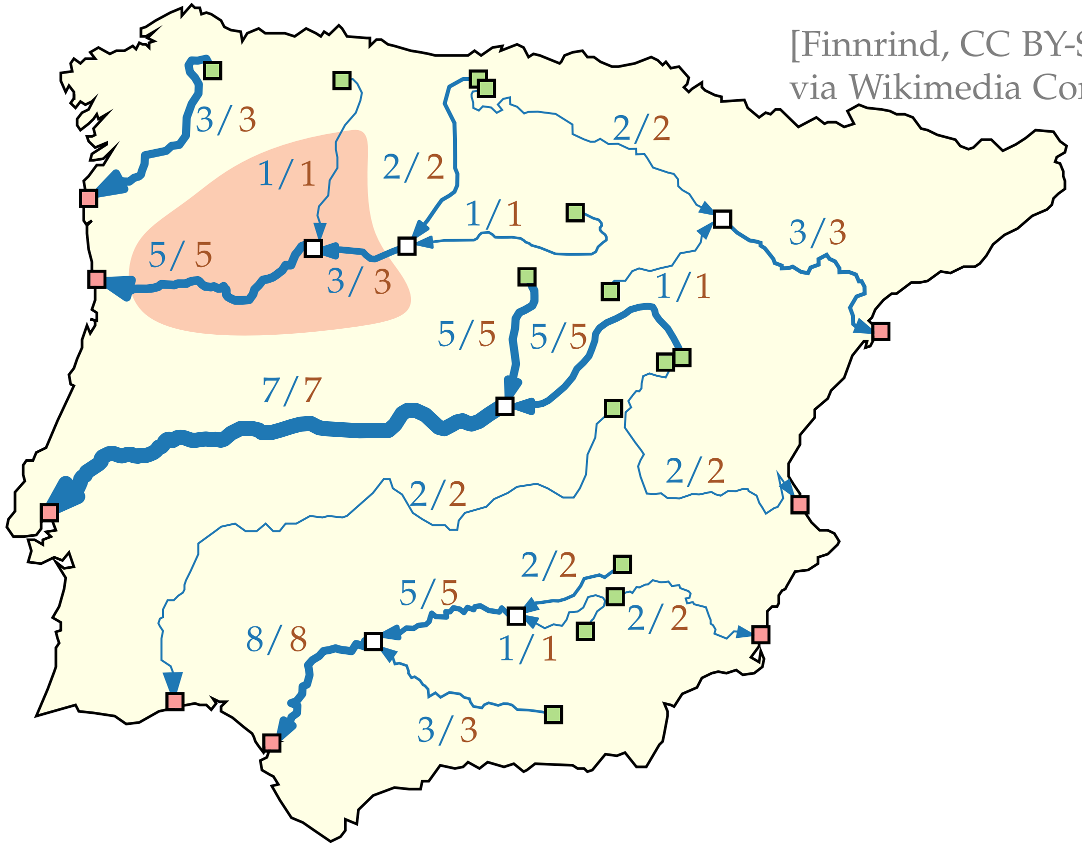
- directed graph $G = (V, E)$
- *sources* $S \subseteq V$, *sinks* $T \subseteq V$
- edge *capacity* $u \colon E \to \mathbb{R}_0^+$

A function $X \colon E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

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

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

Flow Networks



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

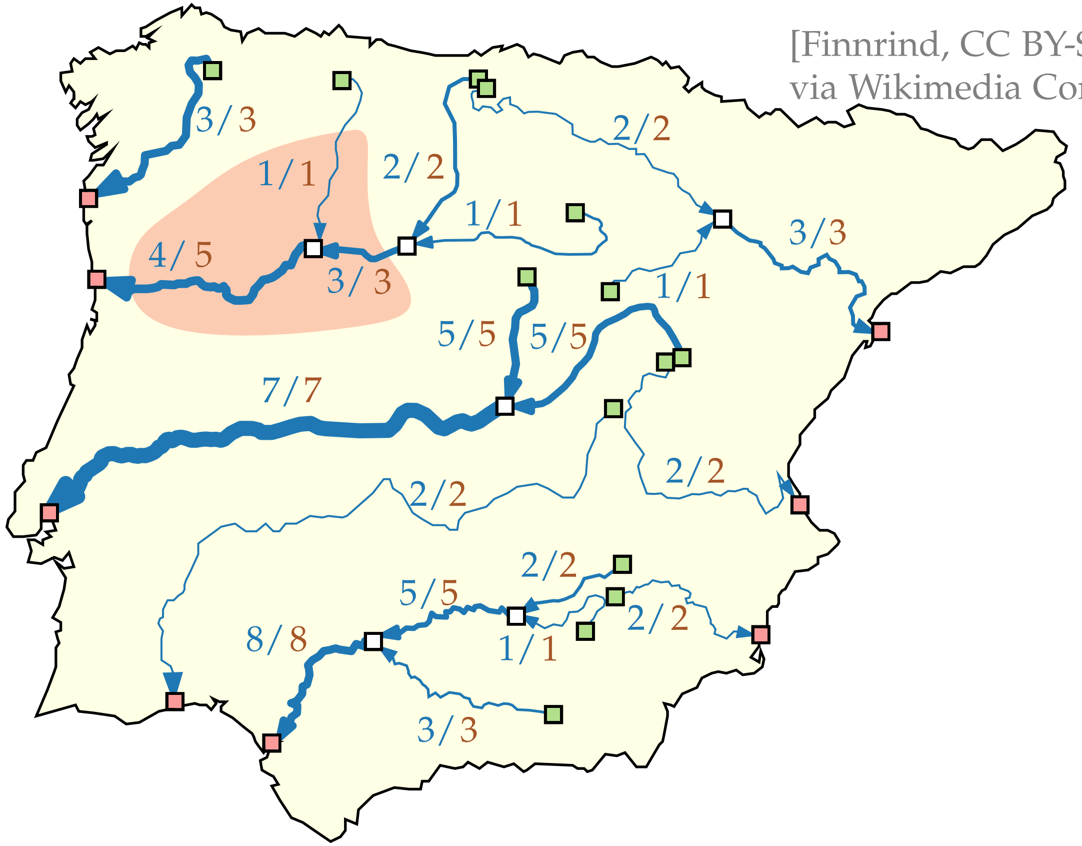
- directed graph $G = (V, E)$
- *sources* $S \subseteq V$, *sinks* $T \subseteq V$
- edge *capacity* $u \colon E \to \mathbb{R}_0^+$

A function $X \colon E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

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

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

Flow Networks



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

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

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

A function $X \colon E \to \mathbb{R}_0^+$ is called **S-T-flow**, if:

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

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

Flow Networks

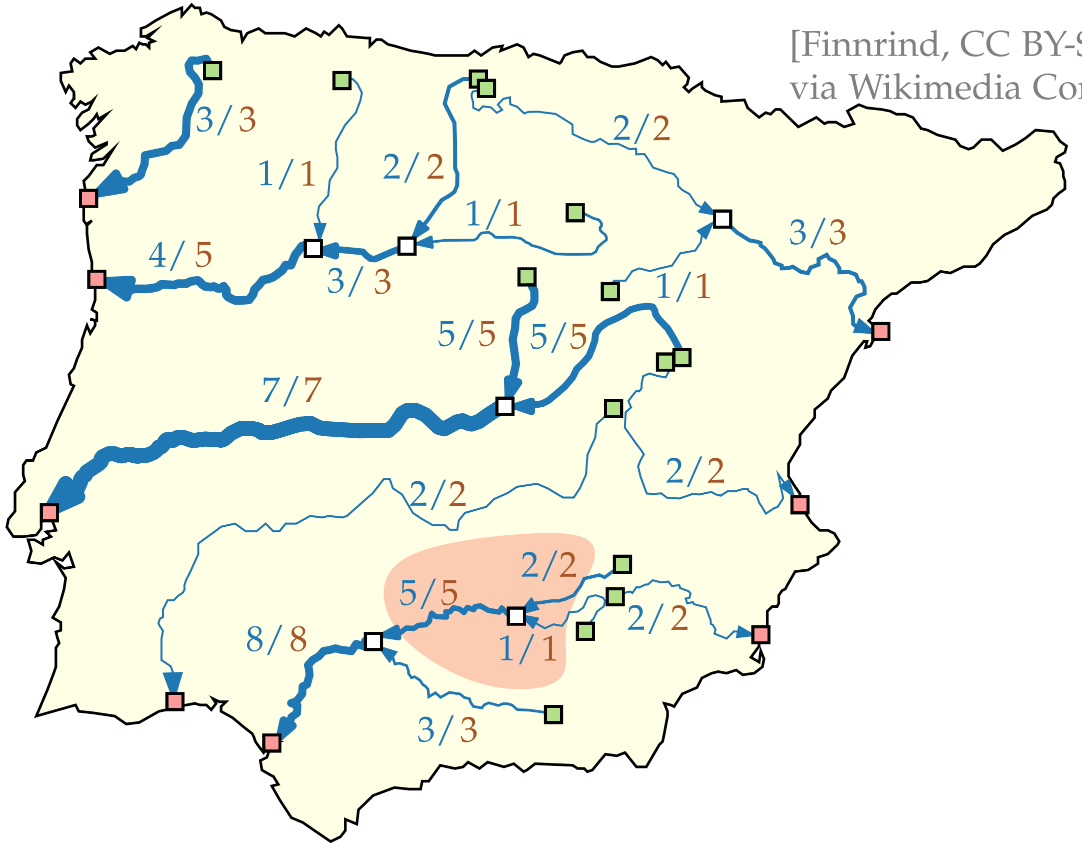
Flow network $(G = (V, E); S, T; u)$ with

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

A function $X\colon E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

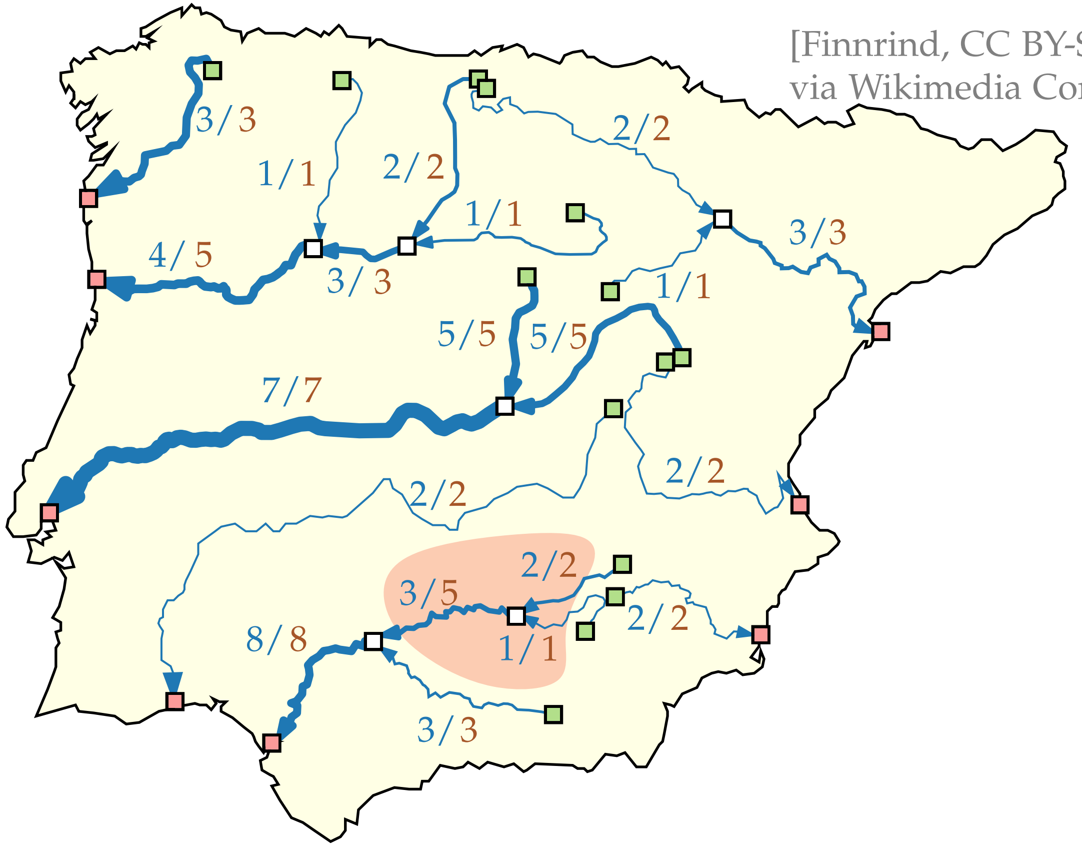
$$0 \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E$$

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



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

Flow Networks



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

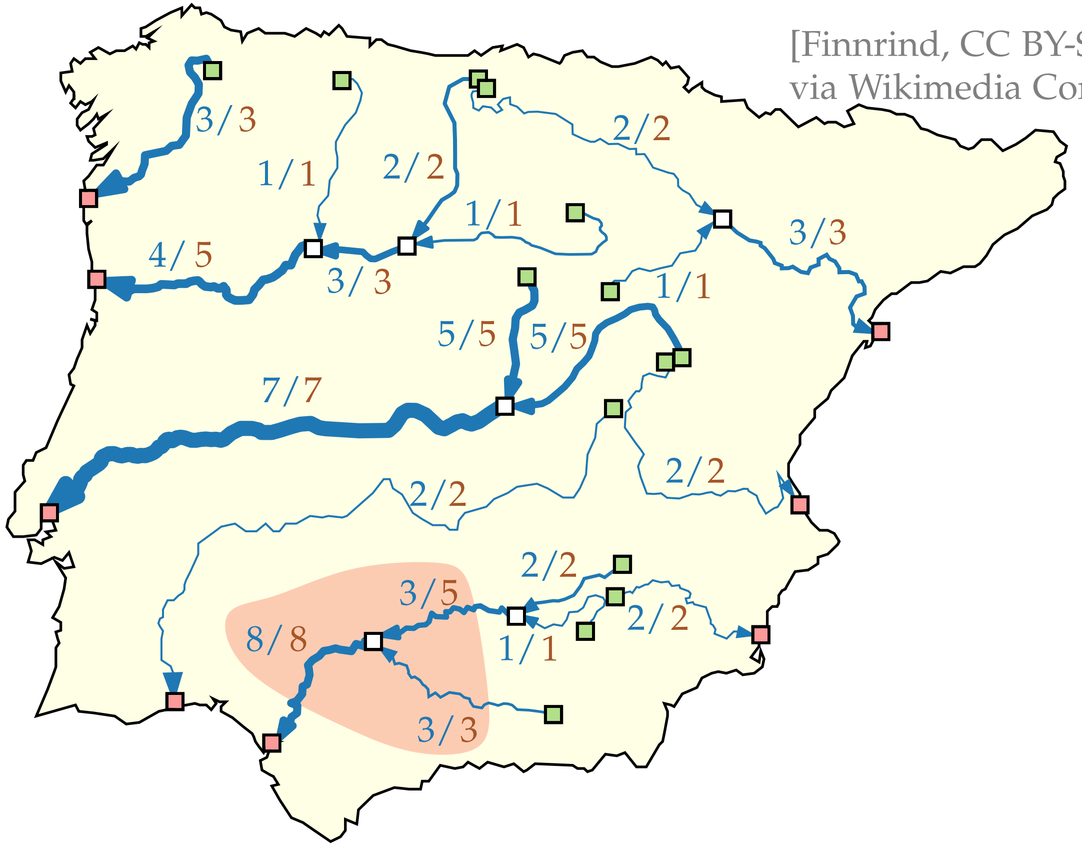
- directed graph $G = (V, E)$
- *sources* $S \subseteq V$, *sinks* $T \subseteq V$
- edge *capacity* $u \colon E \to \mathbb{R}_0^+$

A function $X \colon E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

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

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

Flow Networks



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

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

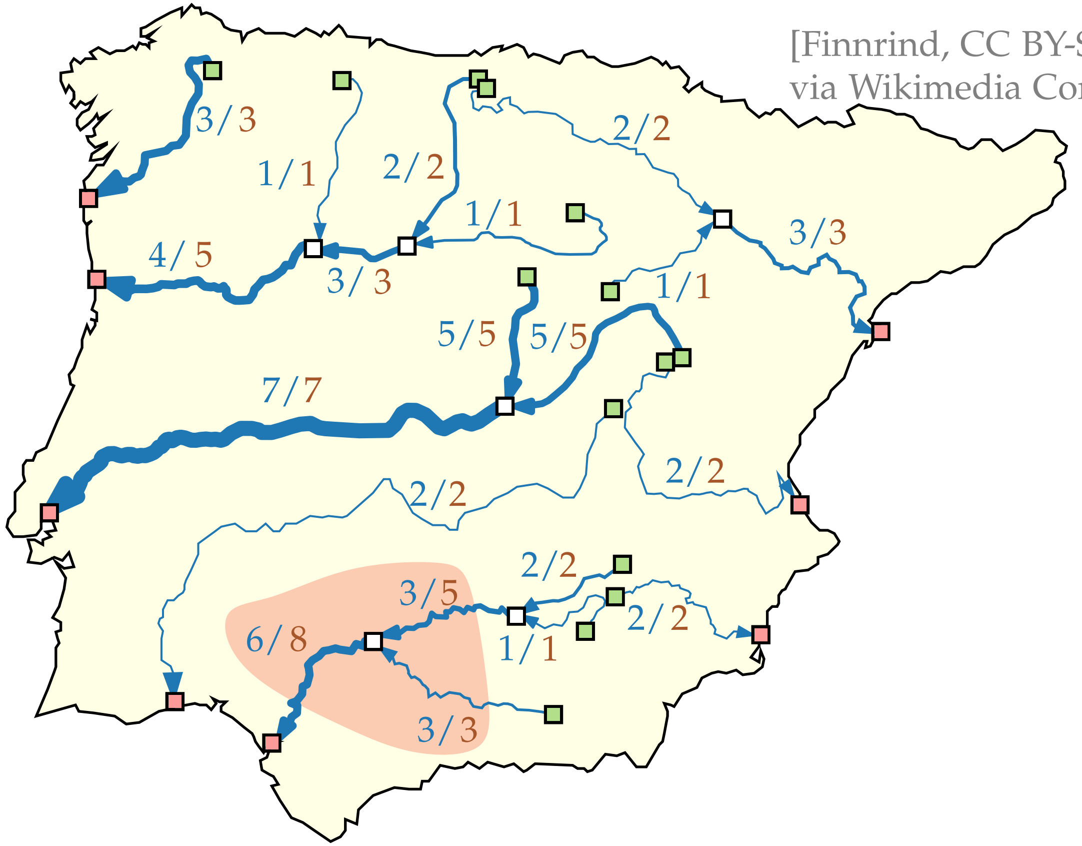
- directed graph $G = (V, E)$
- *sources* $S \subseteq V$, *sinks* $T \subseteq V$
- edge *capacity* $u \colon E \to \mathbb{R}_0^+$

A function $X \colon E \to \mathbb{R}_0^+$ is called ***S*-*T*-flow**, if:

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

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

Flow Networks



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

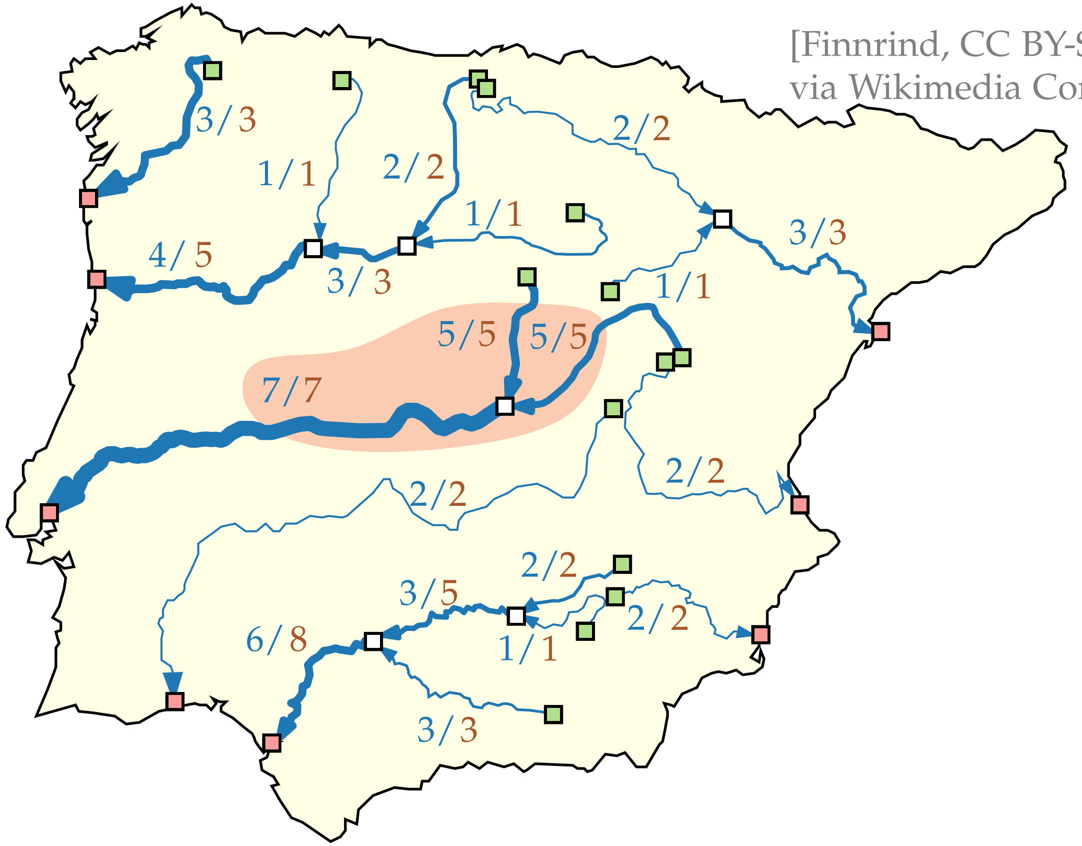
Flow network $(G = (V, E); S, T; u)$ with

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

A function $X \colon E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

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

Flow Networks



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

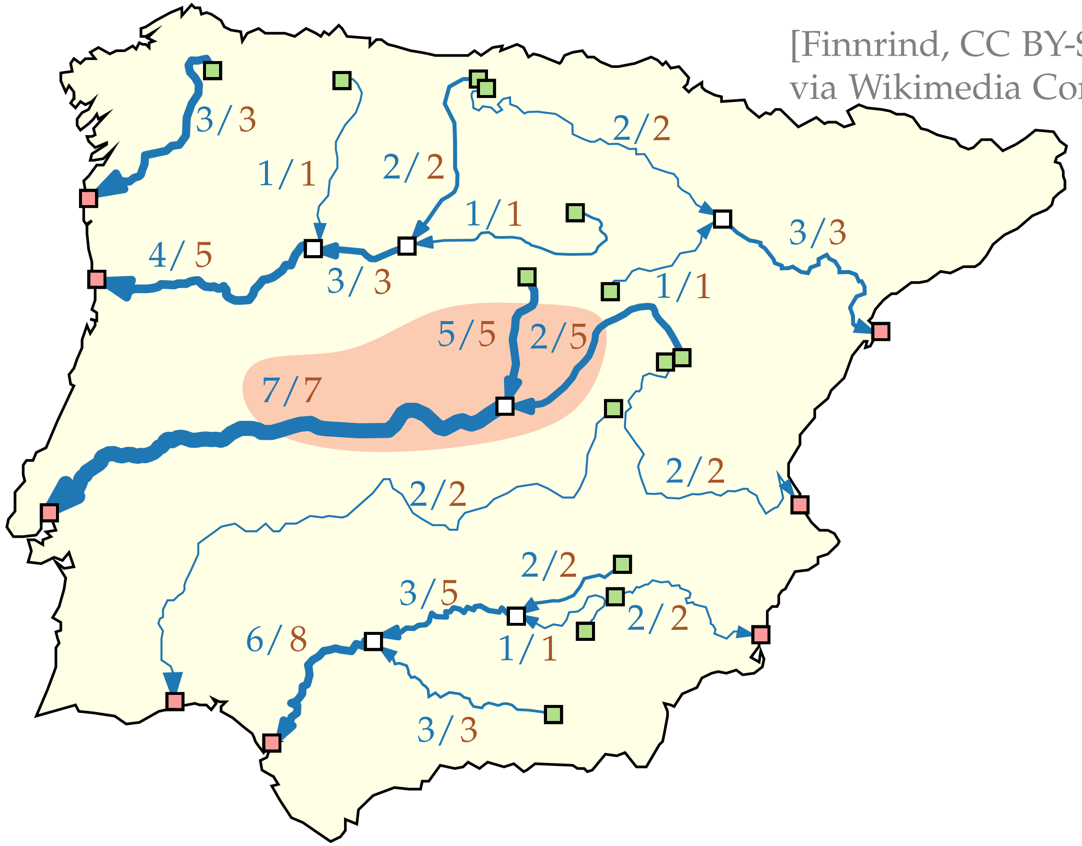
- directed graph $G = (V, E)$
- *sources* $S \subseteq V$, *sinks* $T \subseteq V$
- edge *capacity* $u \colon E \to \mathbb{R}_0^+$

A function $X \colon E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

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

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

Flow Networks



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

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

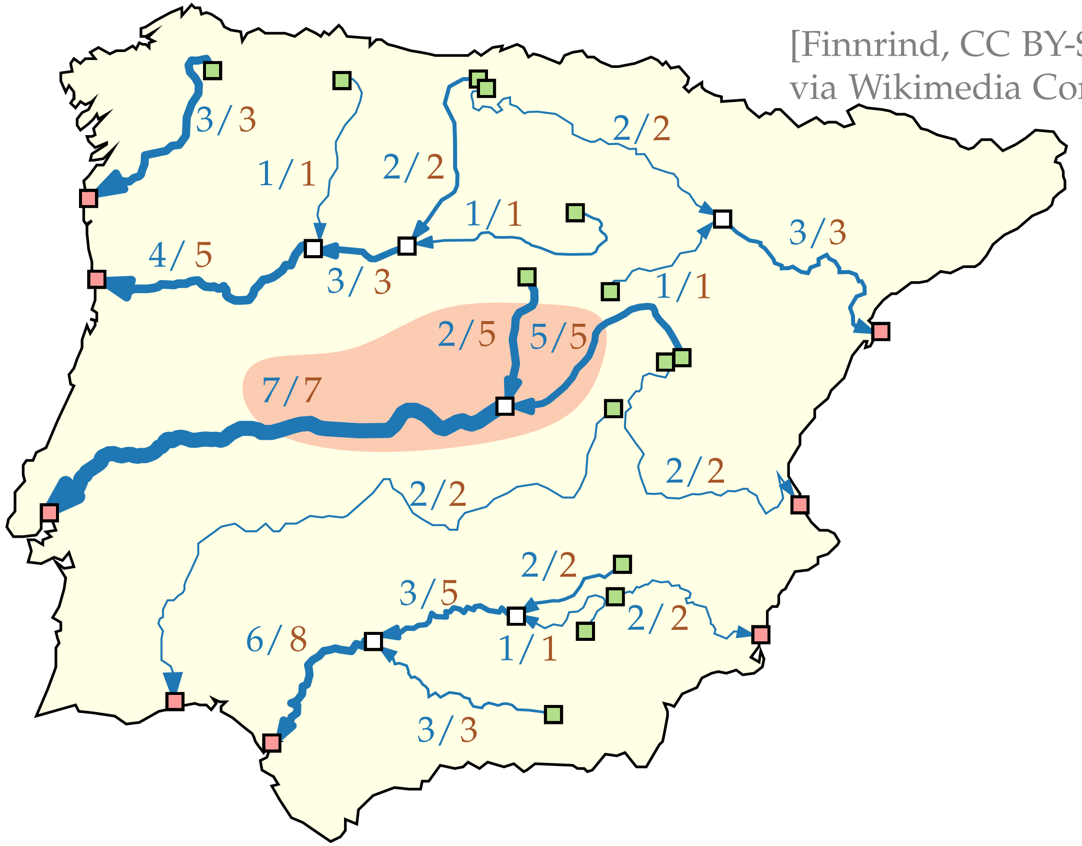
- directed graph $G = (V, E)$
- *sources* $S \subseteq V$, *sinks* $T \subseteq V$
- edge *capacity* $u \colon E \to \mathbb{R}_0^+$

A function $X \colon E \to \mathbb{R}_0^+$ is called ***S*-*T*-flow**, if:

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

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

Flow Networks



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

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

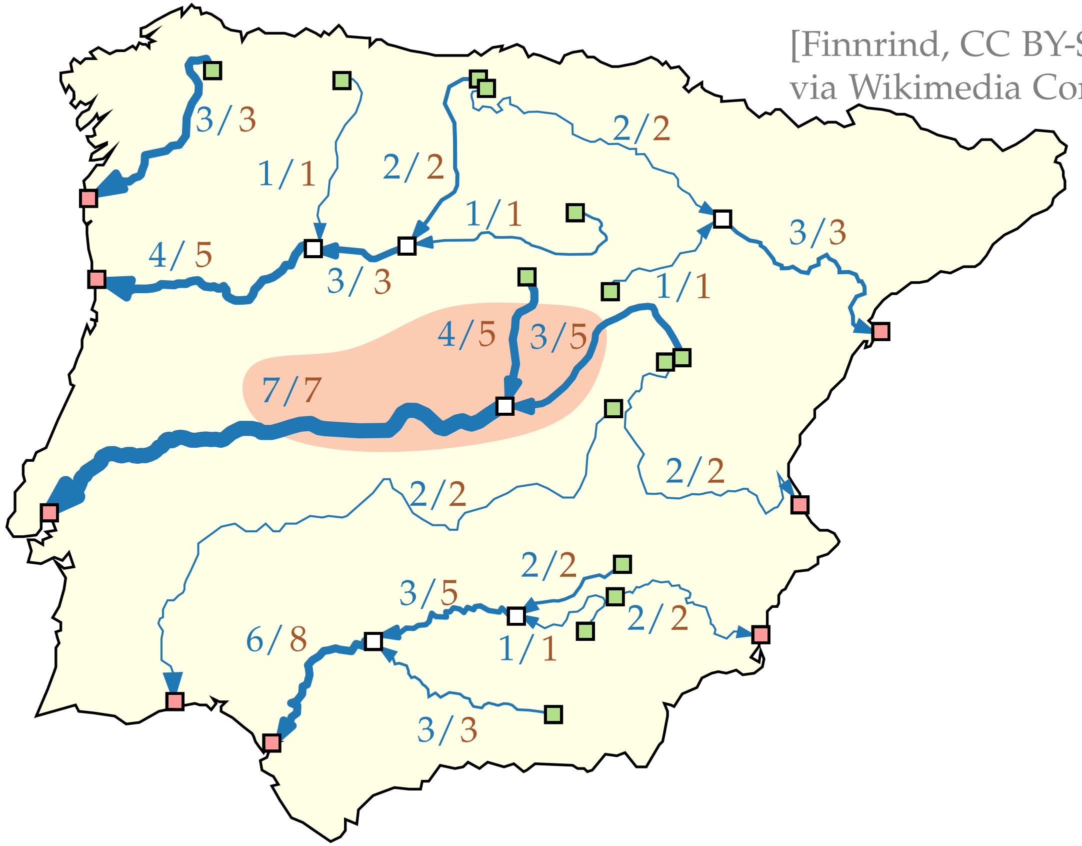
- directed graph $G = (V, E)$
- *sources* $S \subseteq V$, *sinks* $T \subseteq V$
- edge *capacity* $u \colon E \to \mathbb{R}_0^+$

A function $X \colon E \to \mathbb{R}_0^+$ is called ***S*-*T*-flow**, if:

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

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

Flow Networks



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

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

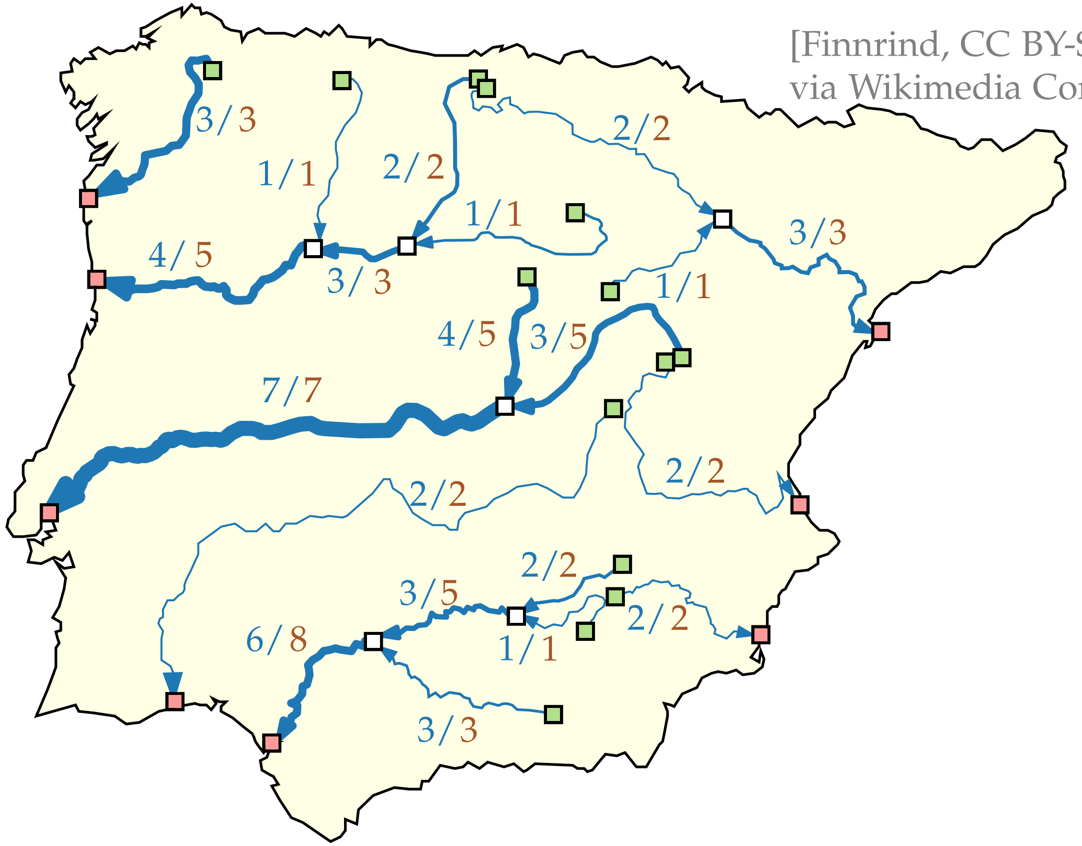
- directed graph $G = (V, E)$
- *sources* $S \subseteq V$, *sinks* $T \subseteq V$
- edge *capacity* $u \colon E \to \mathbb{R}_0^+$

A function $X \colon E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

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

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

Flow Networks



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

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

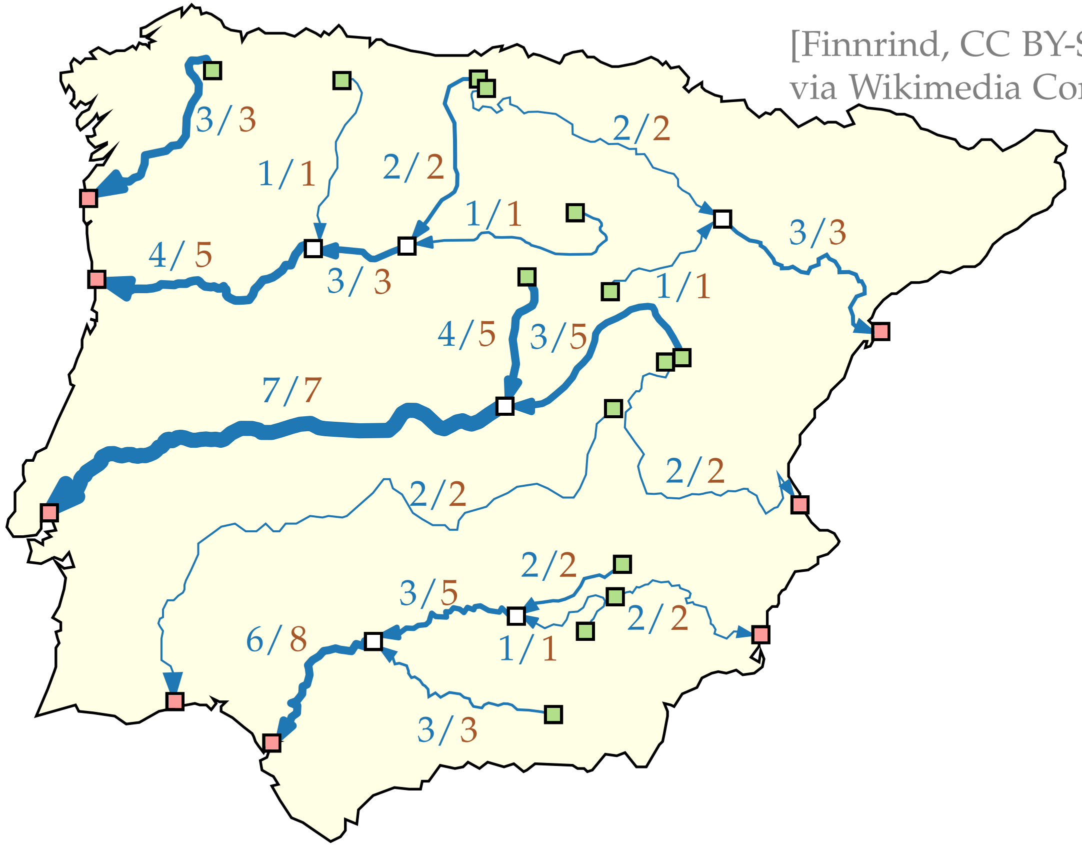
A function $X \colon E \to \mathbb{R}_0^+$ is called ***S-T*-flow**, if:

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

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

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

s-t-Flow Networks



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

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

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

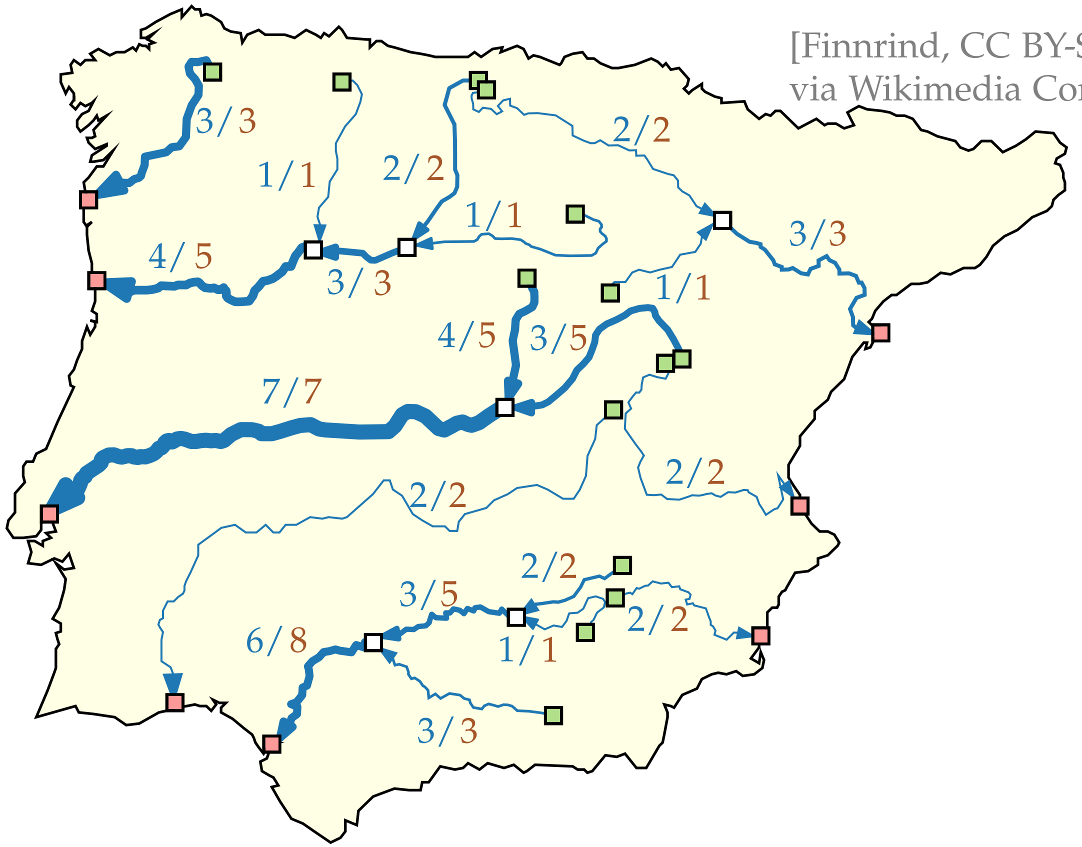
A function $X \colon E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

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

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

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

s-t-Flow Networks



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

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

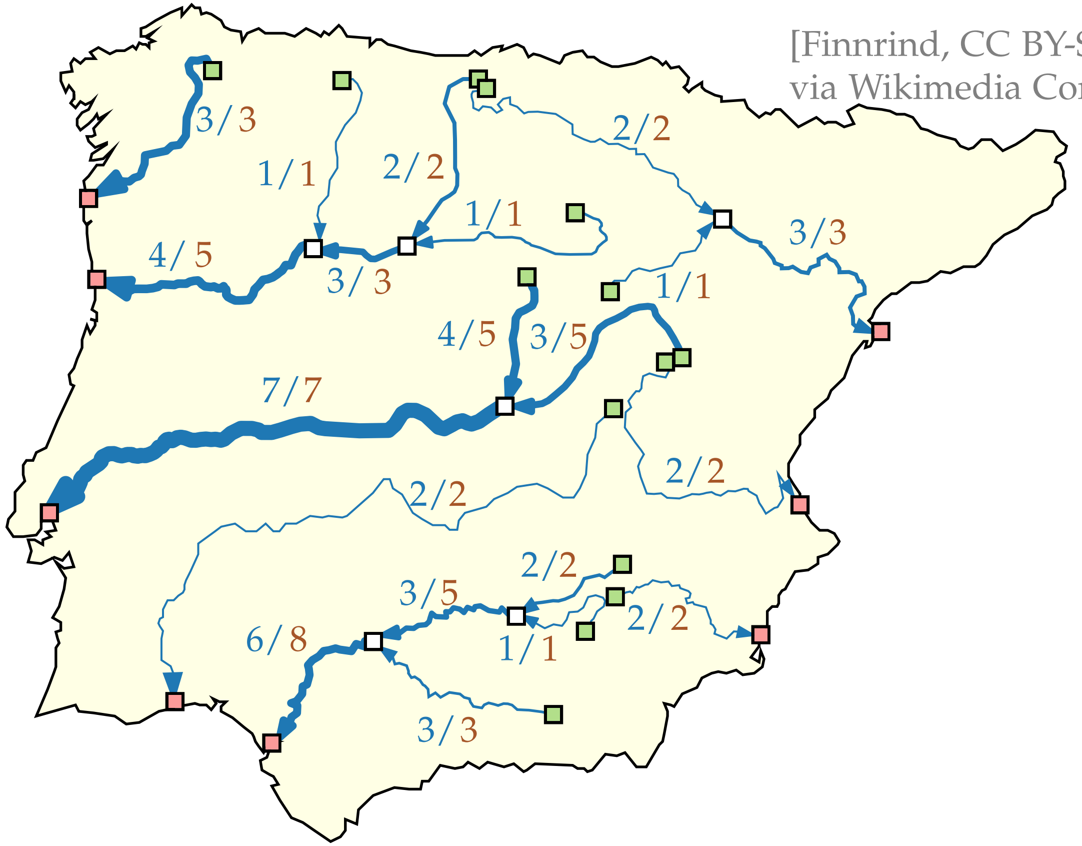
A function $X\colon E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

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

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

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

s-t-Flow Networks



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

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

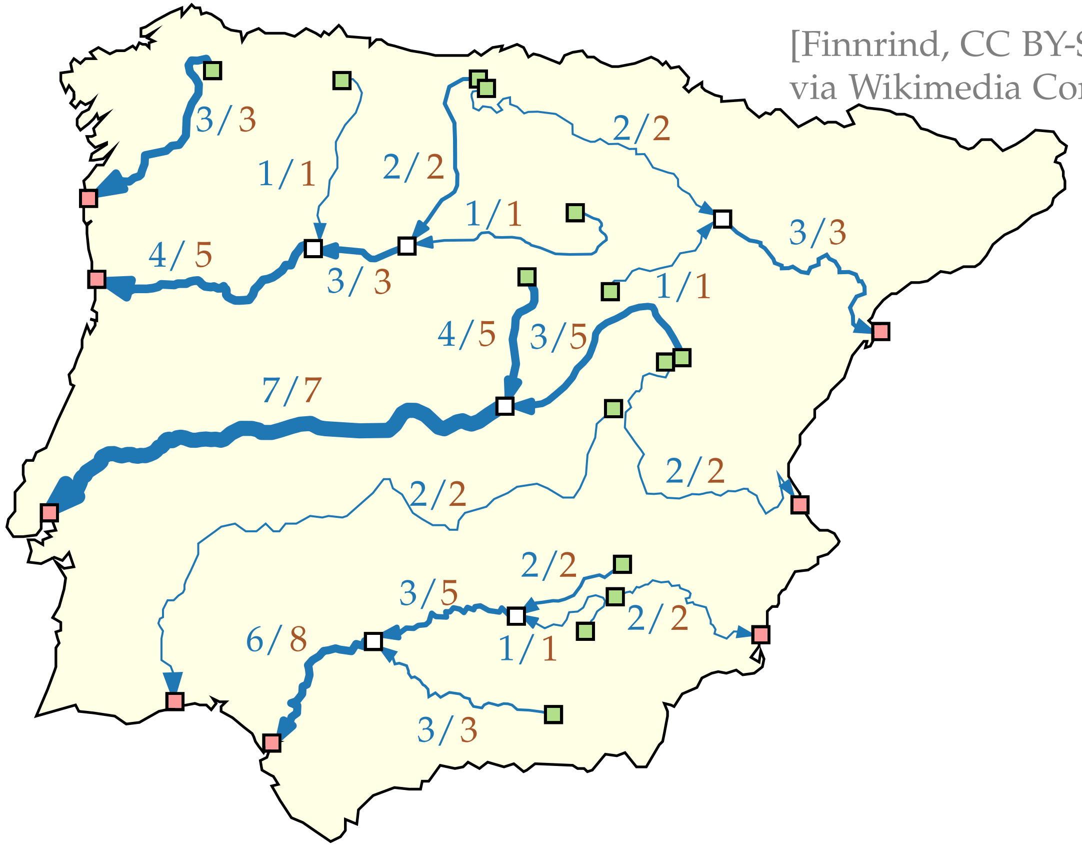
A function $X \colon E \to \mathbb{R}_0^+$ is called ***s*-*t*-flow**, if:

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

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

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

s-t-Flow Networks



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

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

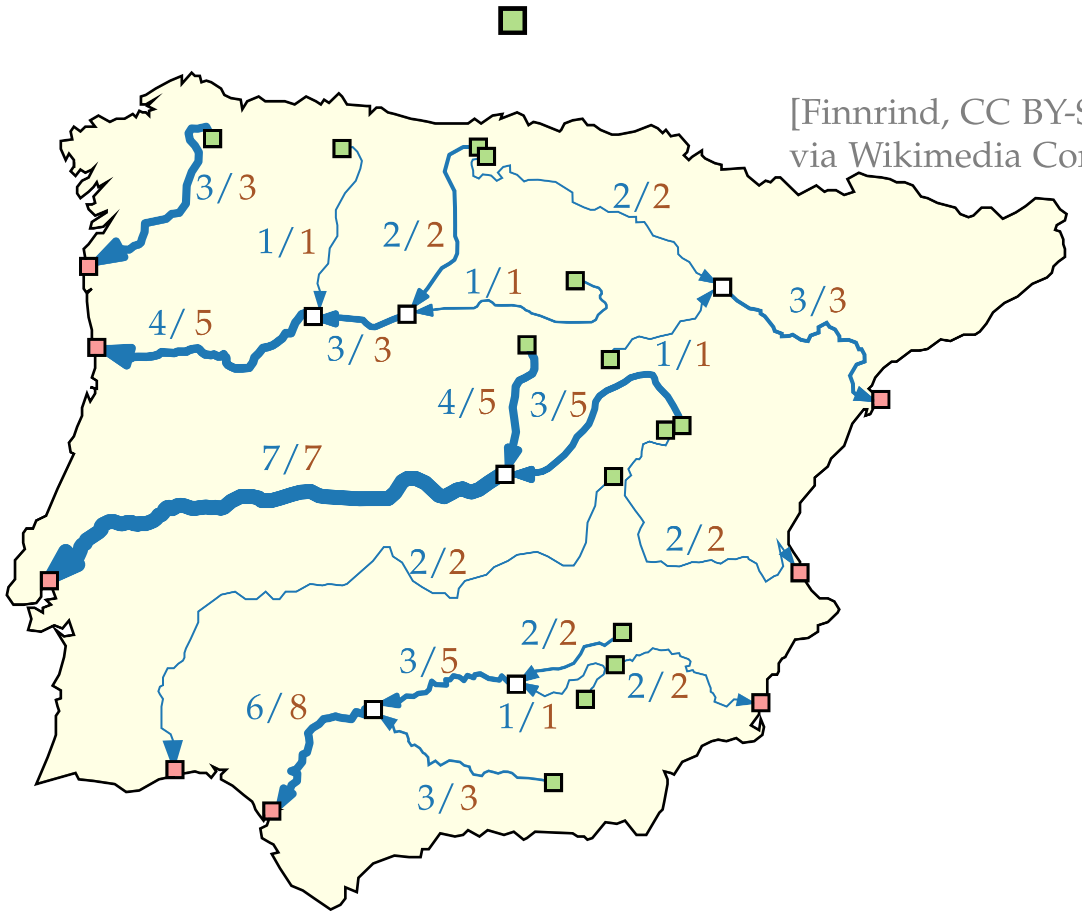
A function $X \colon E \to \mathbb{R}_0^+$ is called **s-t-flow**, if:

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

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

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

s-t-Flow Networks



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

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

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

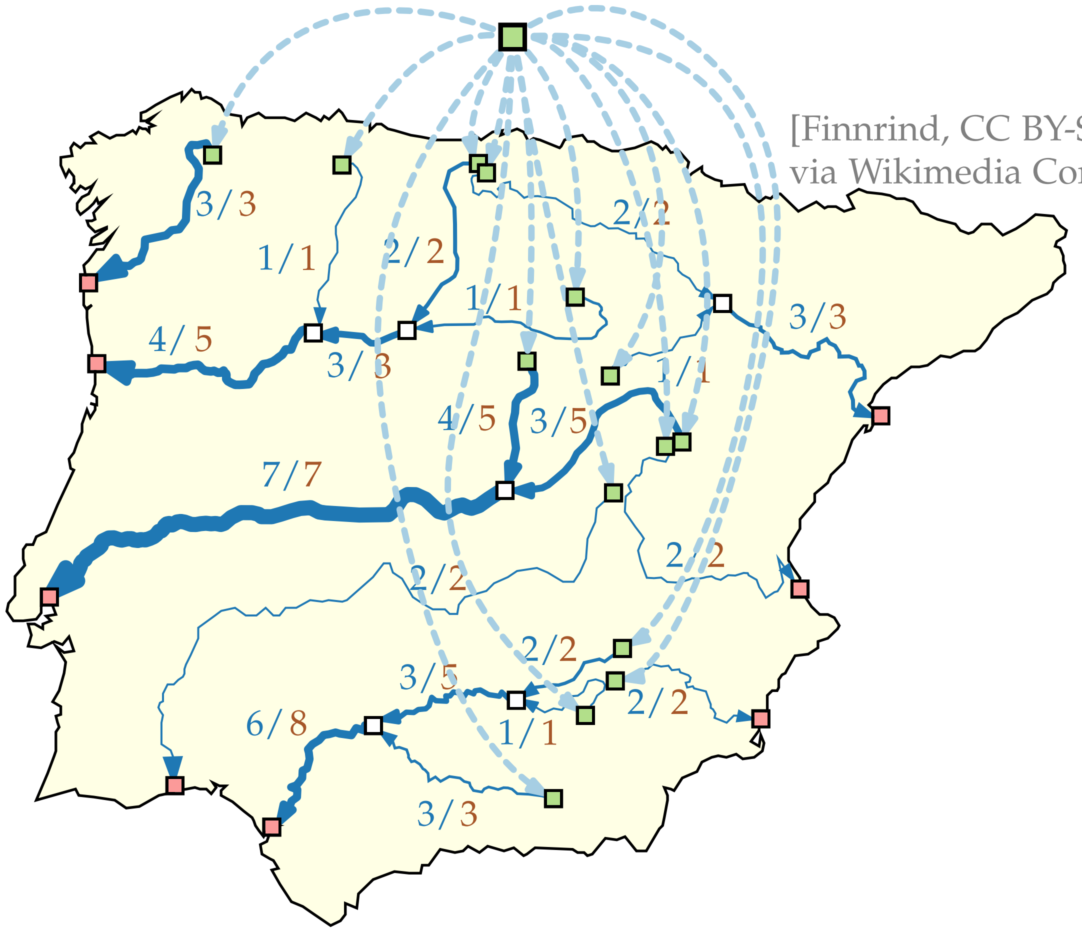
A function $X: E \to \mathbb{R}_0^+$ is called **s-t-flow**, if:

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

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

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

s-t-Flow Networks



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

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

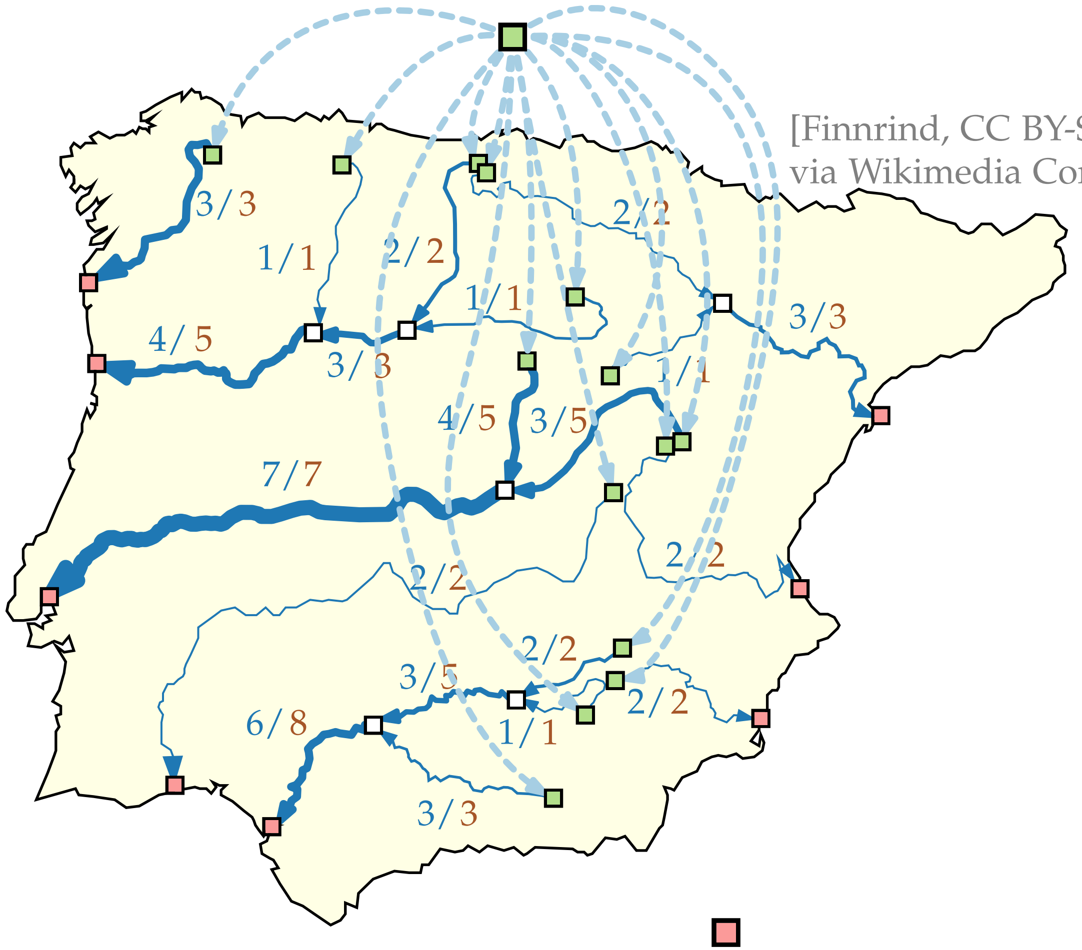
A function $X \colon E \to \mathbb{R}_0^+$ is called ***s-t*-flow**, if:

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

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

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

s-t-Flow Networks



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

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

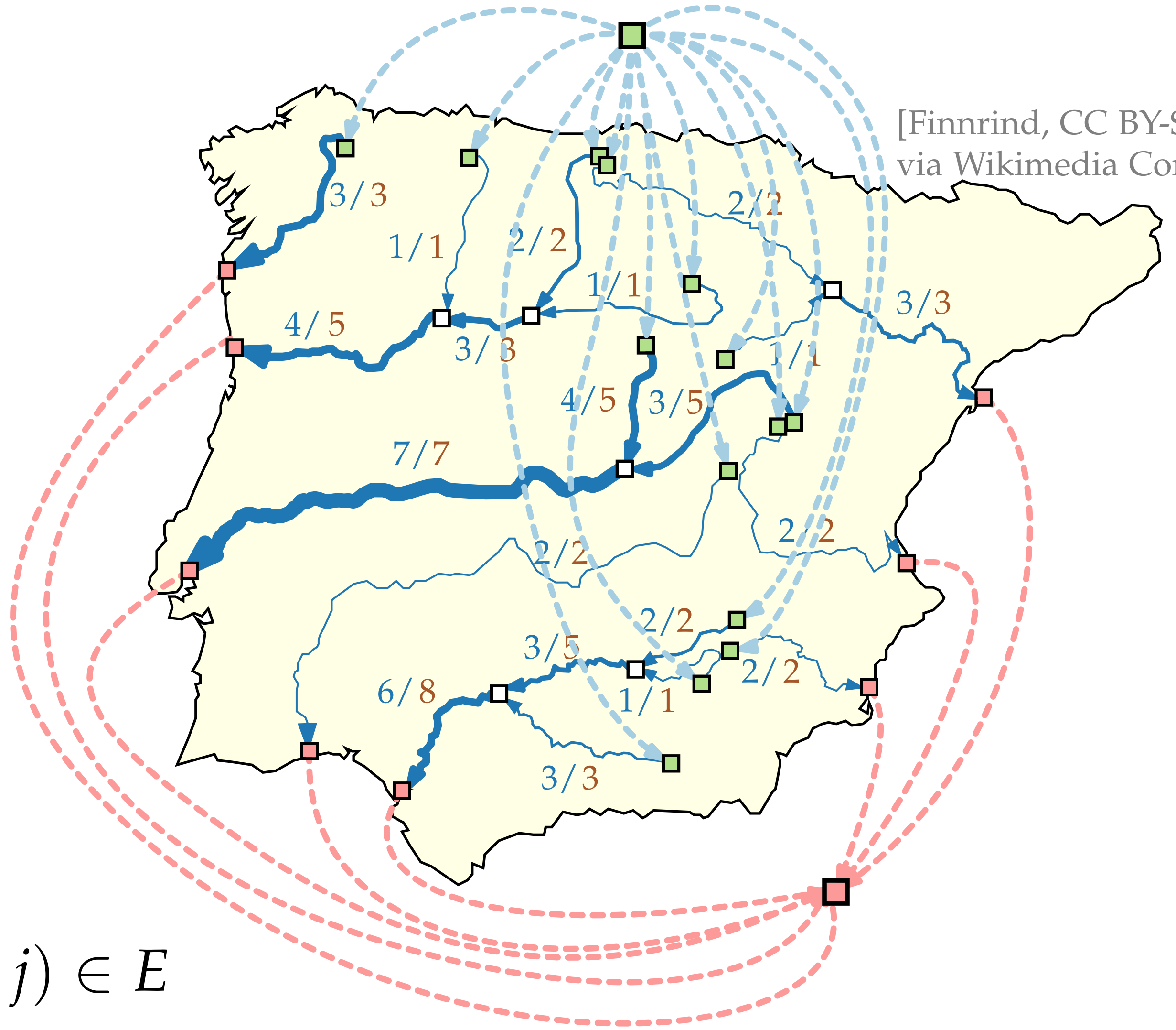
A function $X \colon E \to \mathbb{R}_0^+$ is called ***s-t*-flow**, if:

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

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

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

s-t-Flow Networks



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

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

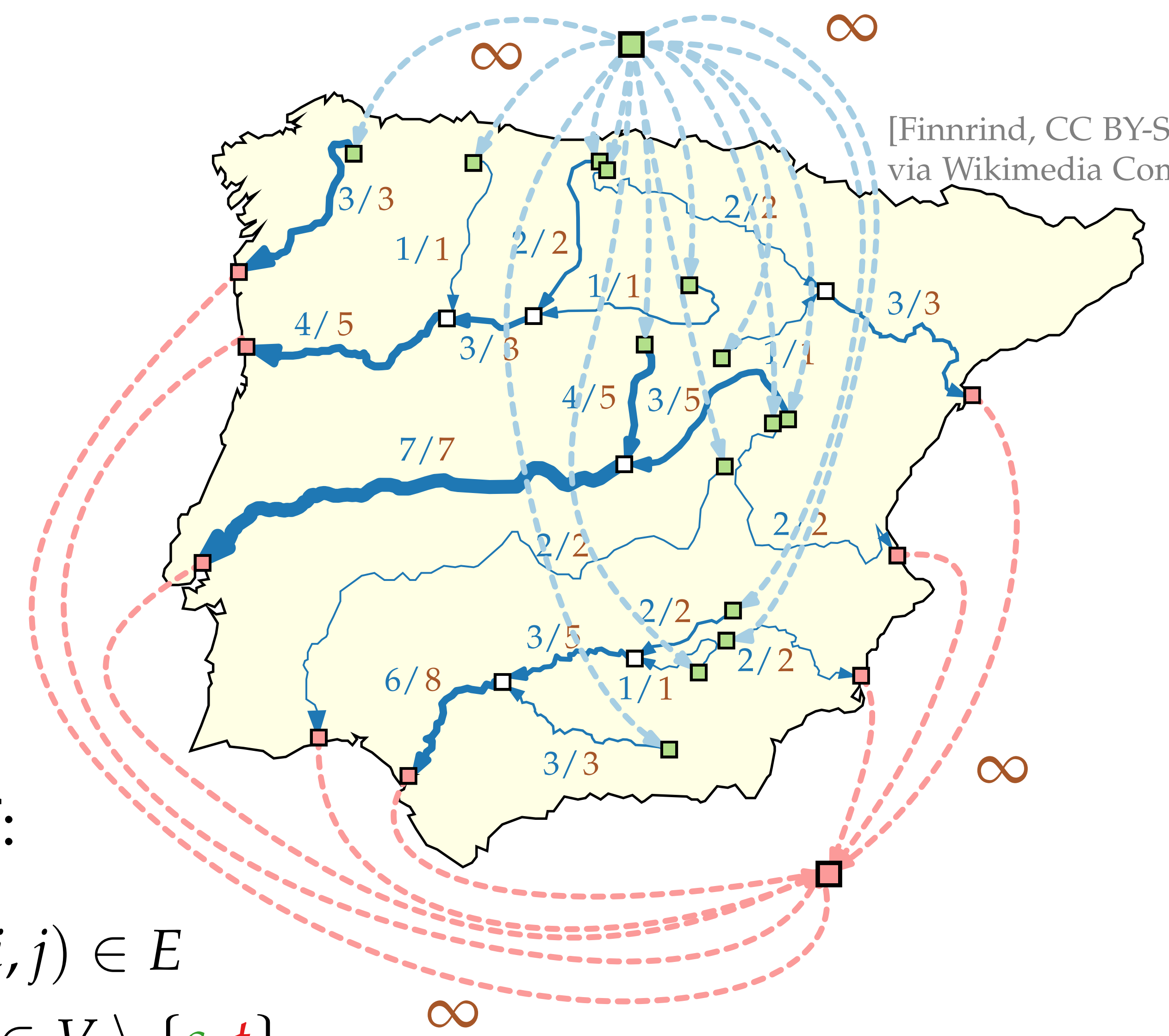
A function $X \colon E \to \mathbb{R}_0^+$ is called ***s-t*-flow**, if:

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

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

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

s-*t*-Flow Networks



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

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

A function $X \colon E \to \mathbb{R}_0^+$ is called ***s*-*t*-flow**, if:

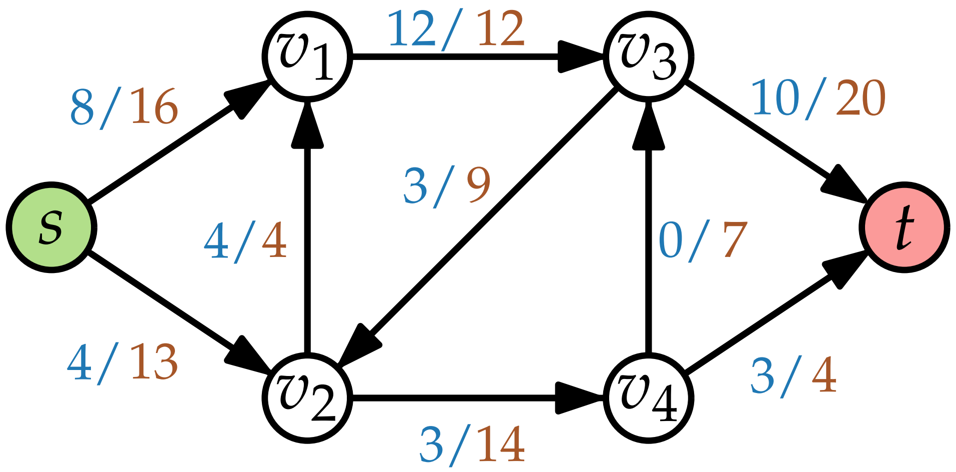
$$0 \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E$$

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

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

Residual Network

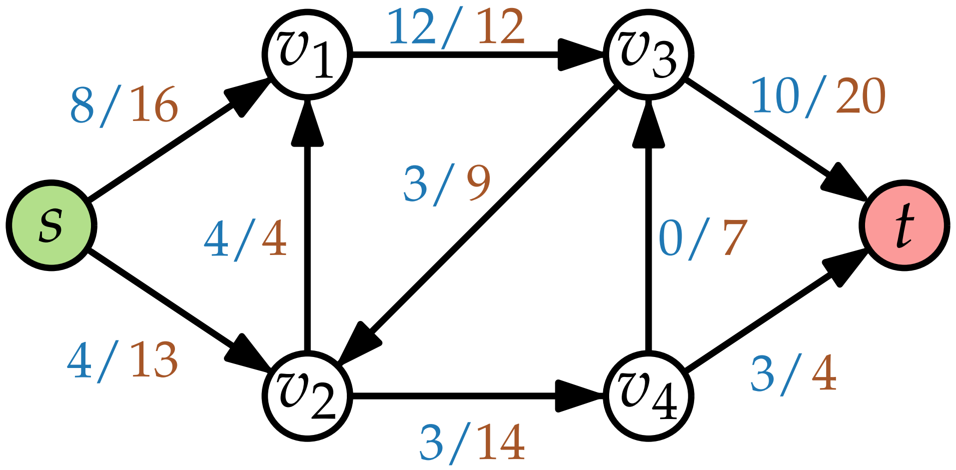
Flow network $(G = (V, E); s, t; u)$



Residual Network

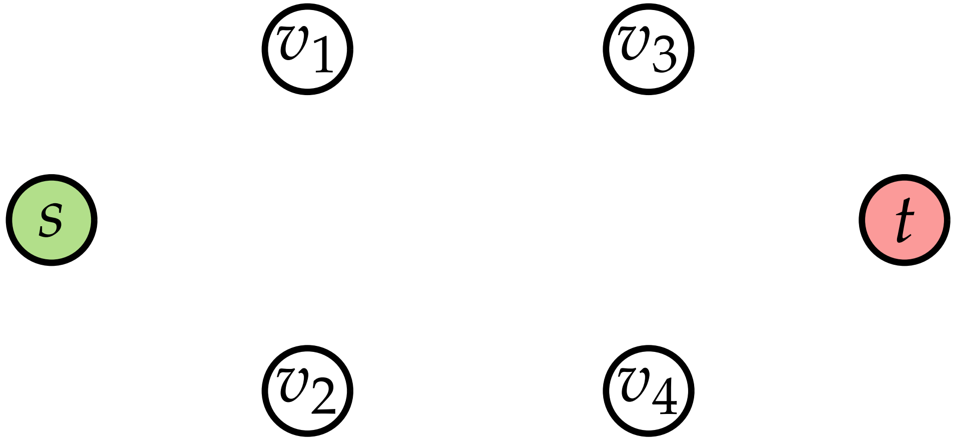
Residual network $G_X = (V, E')$:

Flow network $(G = (V, E); s, t; u)$

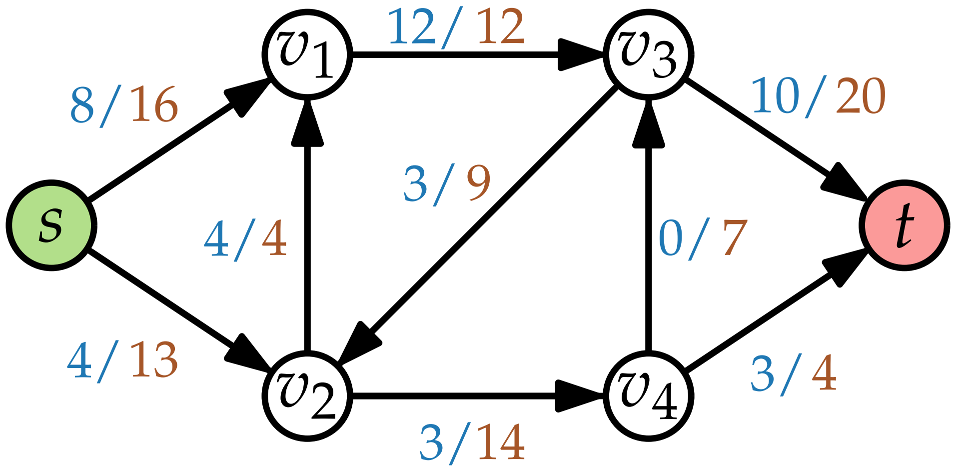


Residual Network

Residual network $G_X = (V, E')$:



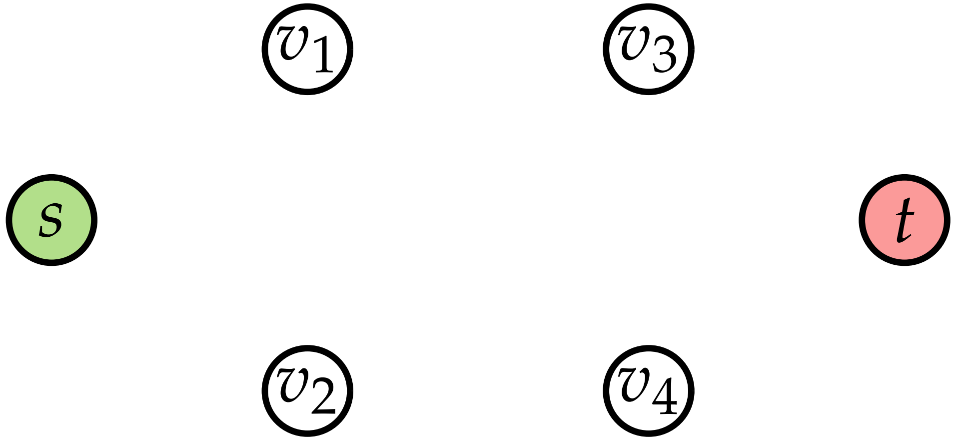
Flow network $(G = (V, E); s, t; u)$



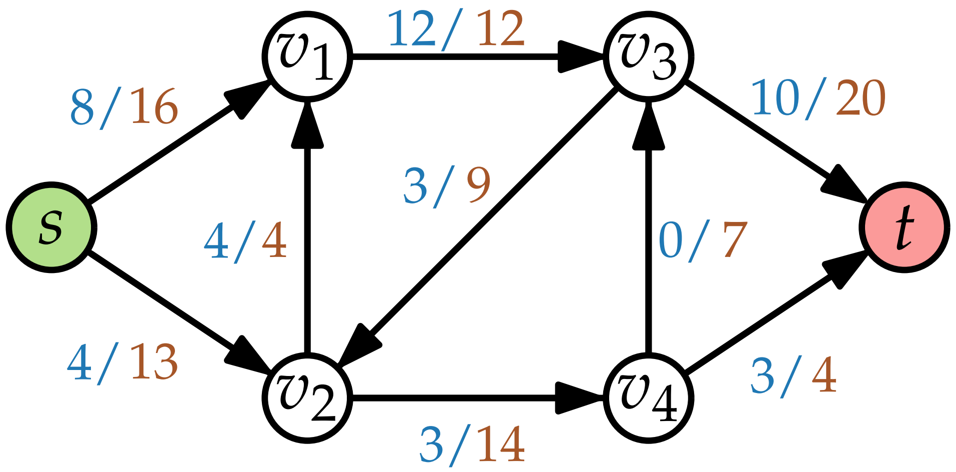
Residual Network

Residual network $G_X = (V, E')$:

- $X(v, v') < u(v, v') \Rightarrow (v, v') \in E'$



Flow network $(G = (V, E); s, t; u)$

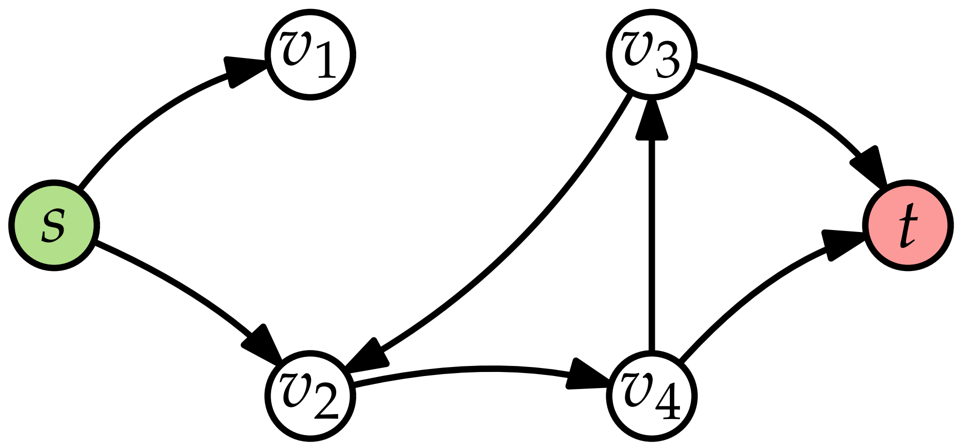


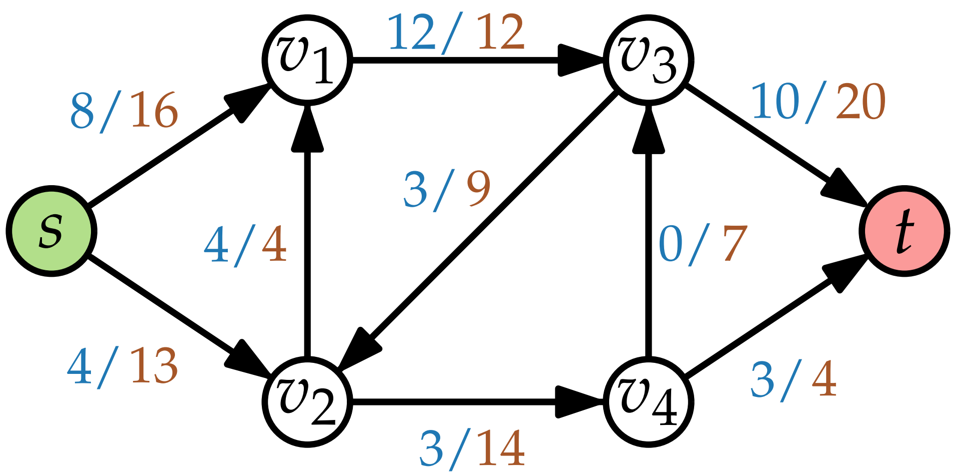
Residual Network

Residual network $G_X = (V, E')$:

- $X(v, v') < u(v, v') \Rightarrow (v, v') \in E'$

Flow network $(G = (V, E); s, t; u)$

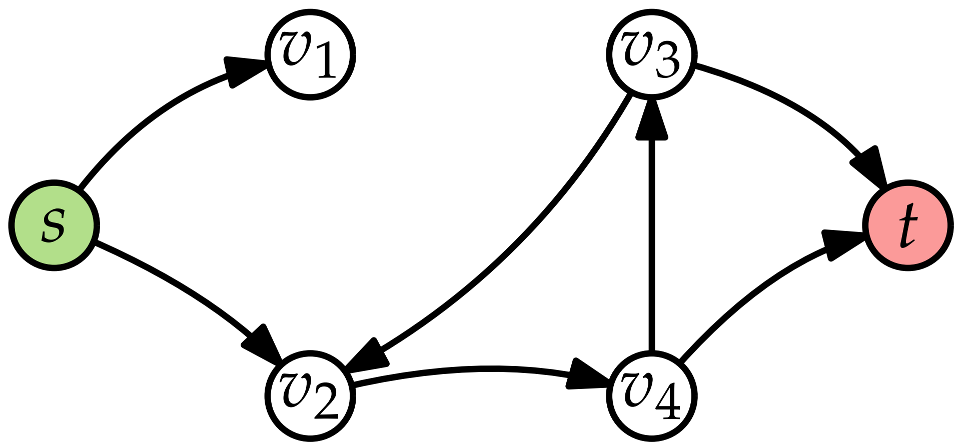




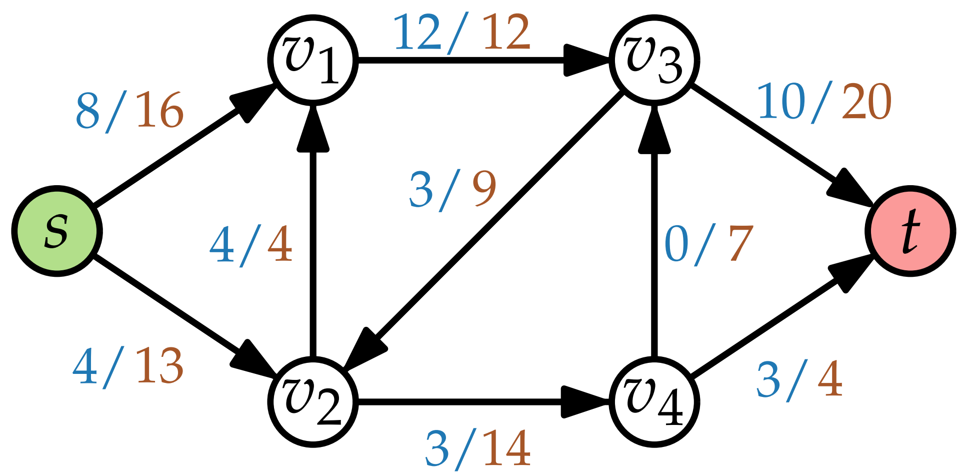
Residual Network

Residual network $G_X = (V, E')$:

- $X(v, v') < u(v, v') \Rightarrow (v, v') \in E'$
- $X(v, v') > 0 \Rightarrow (v', v) \in E'$



Flow network $(G = (V, E); s, t; u)$



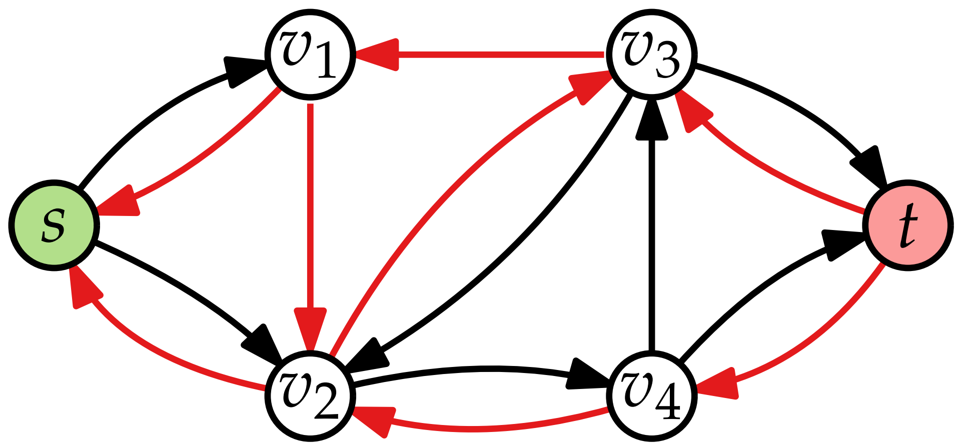
Residual Network

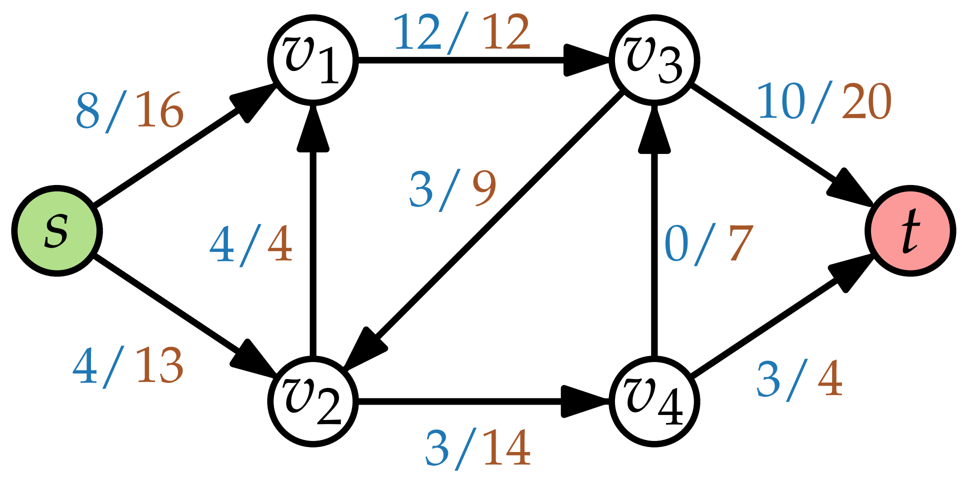
Residual network $G_X = (V, E')$:

- $X(v, v') < u(v, v') \Rightarrow (v, v') \in E'$

- $X(v, v') > 0 \Rightarrow (v', v) \in E'$

Flow network $(G = (V, E); s, t; u)$

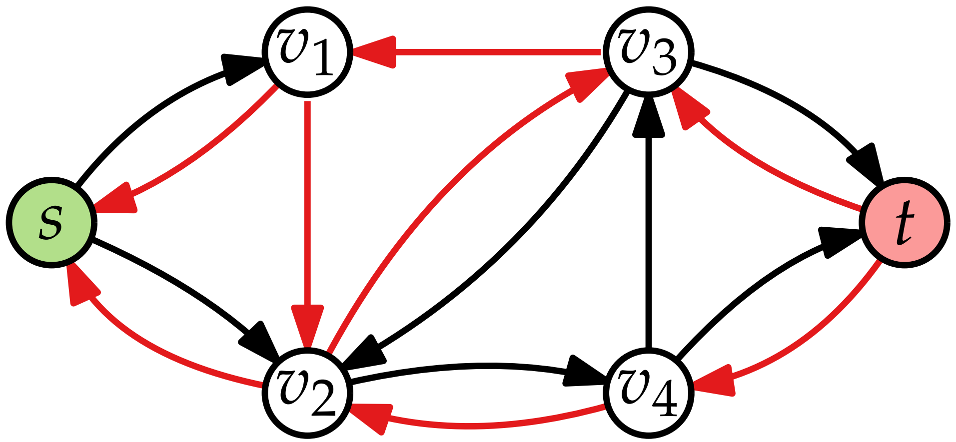




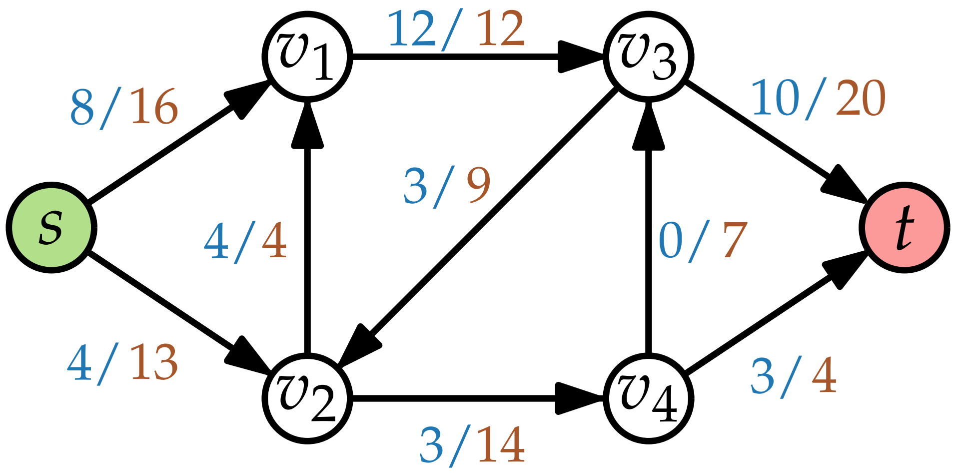
Residual Network

Residual network $G_X = (V, E')$:

- $X(v,v') < u(v,v') \Rightarrow (v,v') \in E'$
  $c(v,v') = u(v,v') - (v,v')$
- $X(v,v') > 0 \Rightarrow (v',v) \in E'$



Flow network $(G = (V, E); s, t; u)$

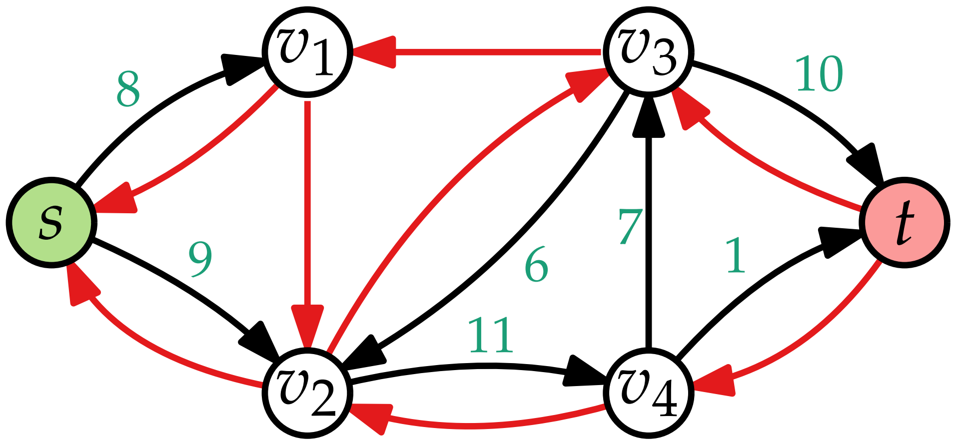


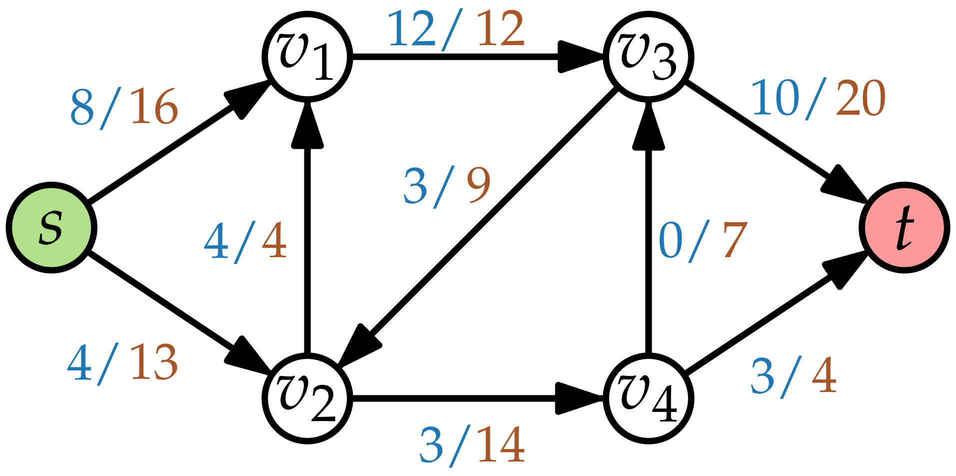
Residual Network

Residual network $G_X = (V, E')$:

- $X(v,v') < u(v,v') \Rightarrow (v,v') \in E'$
  $c(v,v') = u(v,v') - (v,v')$
- $X(v,v') > 0 \Rightarrow (v',v) \in E'$

Flow network $(G = (V,E); s, t; u)$

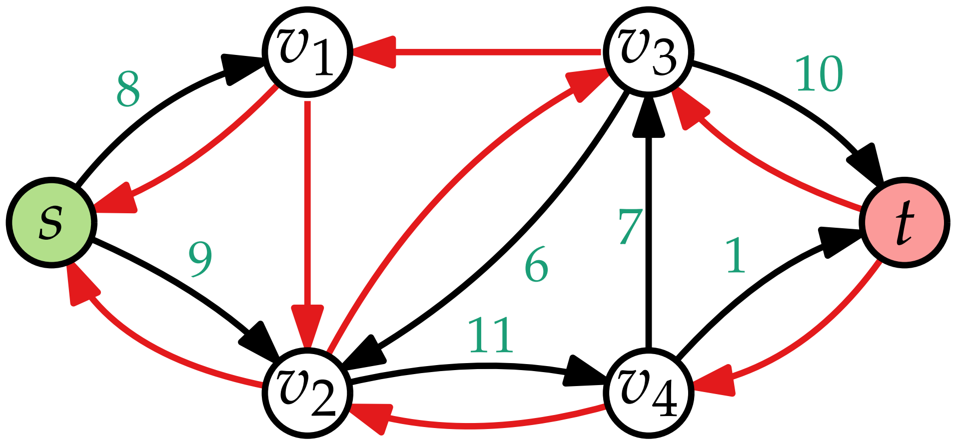




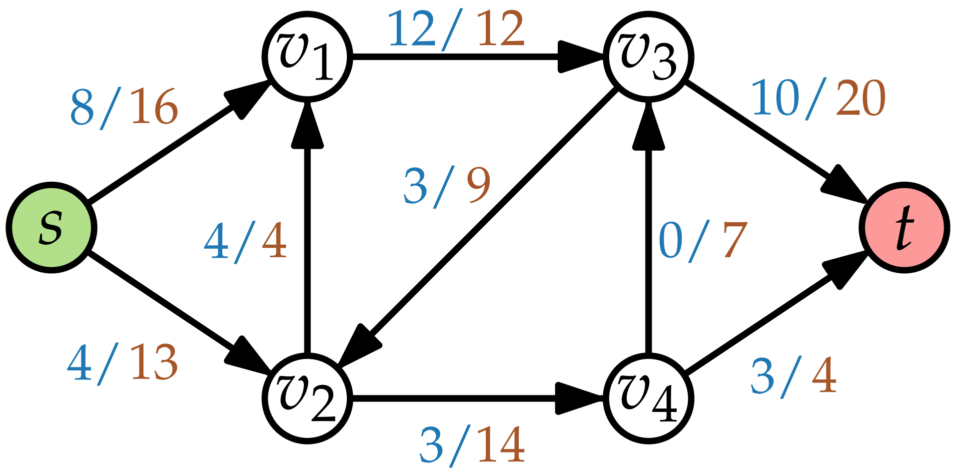
Residual Network

Residual network $G_X = (V, E')$:

- $X(v,v') < u(v,v') \Rightarrow (v,v') \in E'$
  $$c(v,v') = u(v,v') - (v,v')$$
- $X(v,v') > 0 \Rightarrow (v',v) \in E'$
  $$c(v,v') = u(v,v')$$



Flow network $(G = (V, E); s, t; u)$



Residual Network

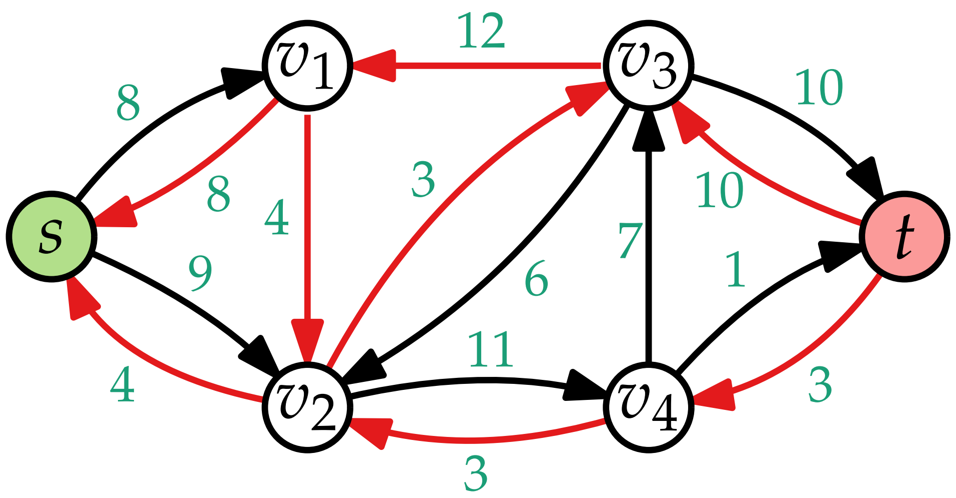
Residual network $G_X = (V, E')$:

- $X(v,v') < u(v,v') \Rightarrow (v,v') \in E'$

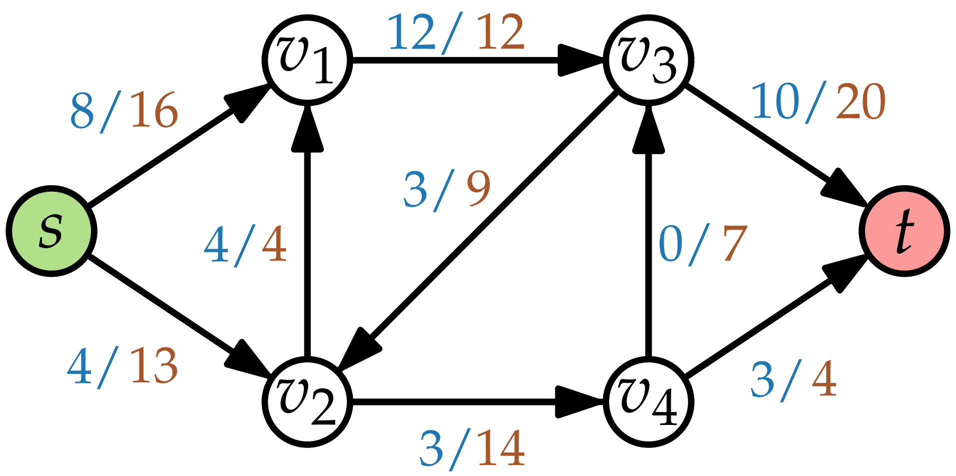
  $c(v,v') = u(v,v') - (v,v')$

- $X(v,v') > 0 \Rightarrow (v',v) \in E'$

  $c(v,v') = u(v,v')$



Flow network $(G = (V,E); s, t; u)$



Residual Network

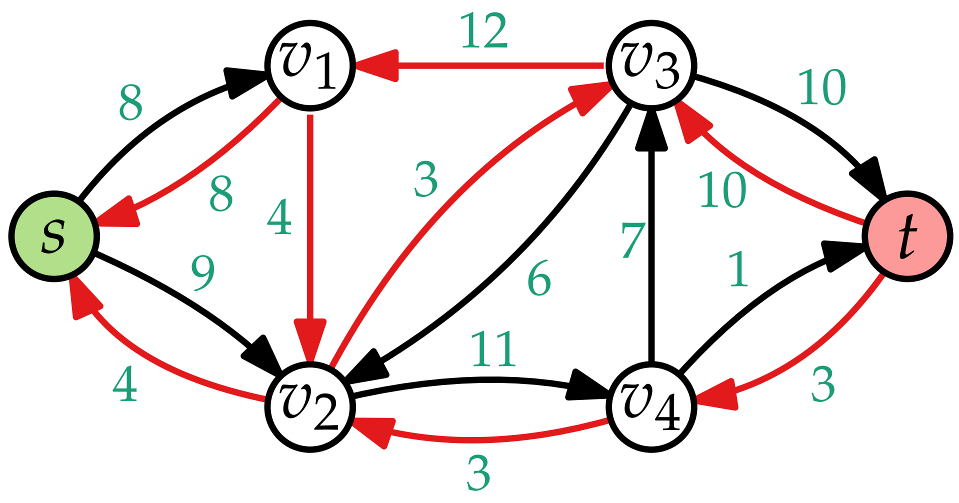
Residual network $G_X = (V, E')$:

- $X(v,v') < u(v,v') \Rightarrow (v,v') \in E'$

  $c(v,v') = u(v,v') - (v,v')$

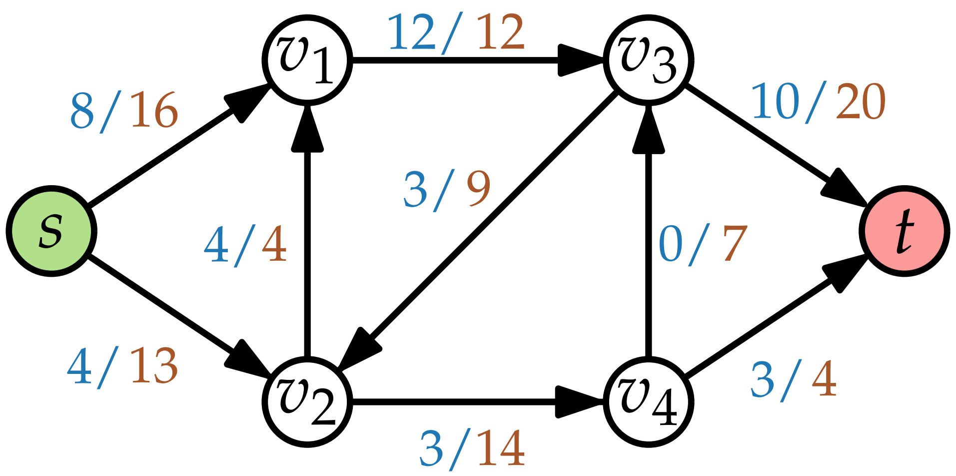
- $X(v,v') > 0 \Rightarrow (v',v) \in E'$

  $c(v,v') = u(v,v')$



Flow-increasing path W

Flow network $(G = (V, E); s, t; u)$

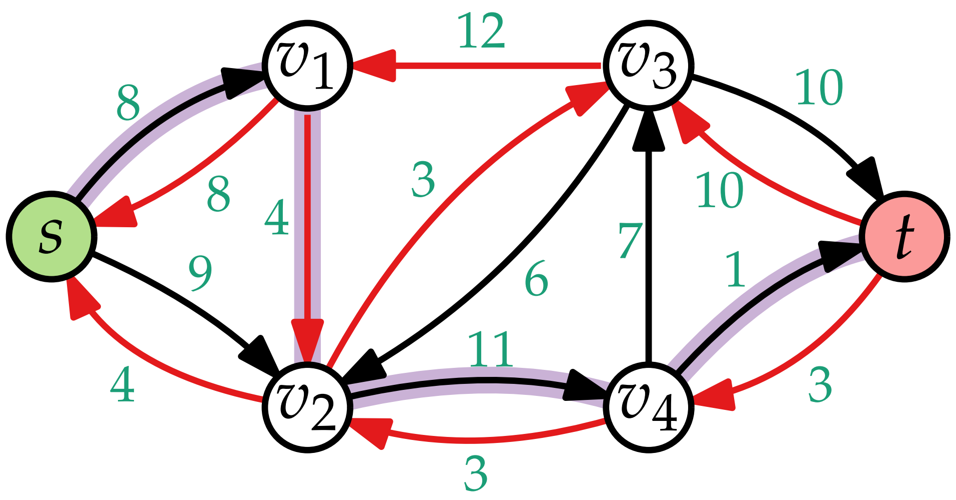


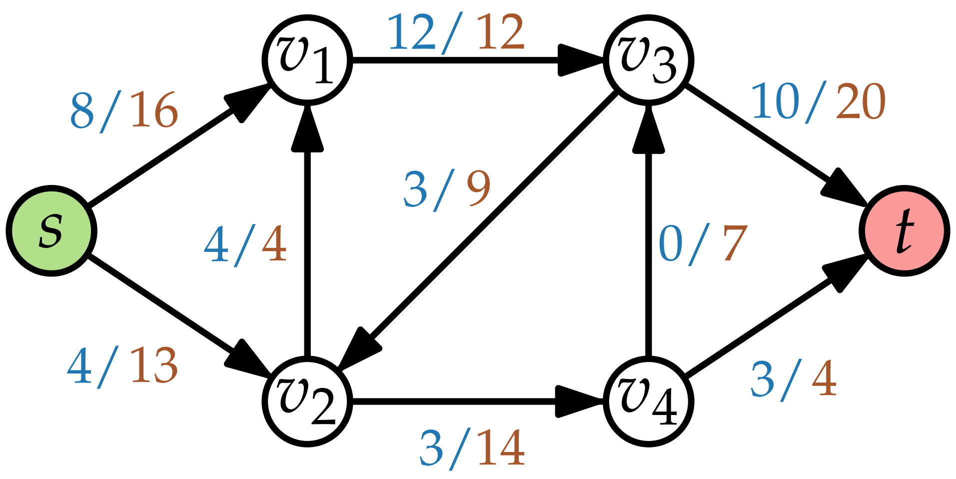
Residual Network

Residual network $G_X = (V, E')$:

- $X(v,v') < u(v,v') \Rightarrow (v,v') \in E'$
  $c(v,v') = u(v,v') - (v,v')$
- $X(v,v') > 0 \Rightarrow (v',v) \in E'$
  $c(v,v') = u(v,v')$

Flow network $(G = (V,E); s, t; u)$





Flow-increasing path W

Residual Network

Residual network $G_X = (V, E')$:

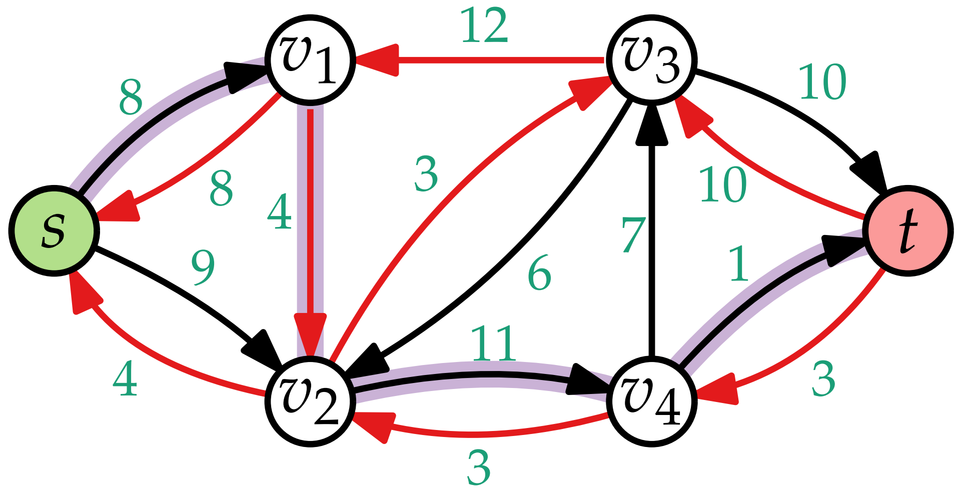
- $X(v,v') < u(v,v') \Rightarrow (v,v') \in E'$

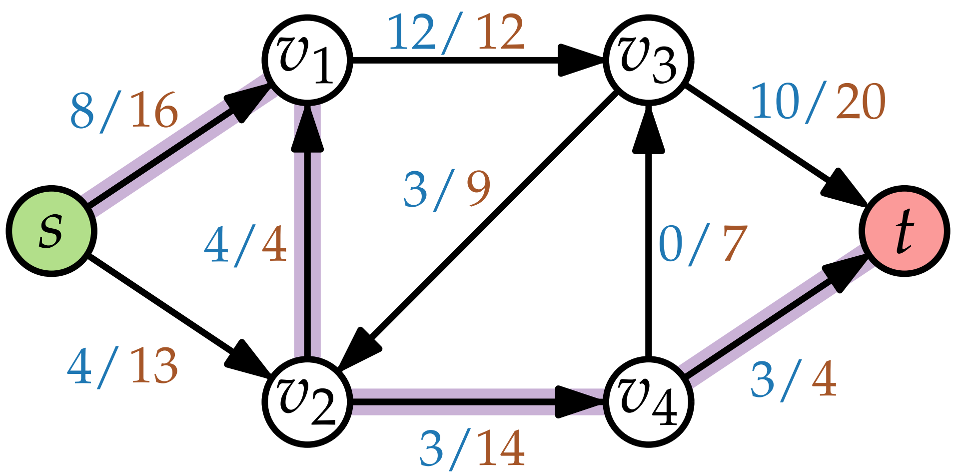
  $c(v,v') = u(v,v') - (v,v')$

- $X(v,v') > 0 \Rightarrow (v',v) \in E'$

  $c(v,v') = u(v,v')$

Flow network $(G = (V,E); s,t; u)$





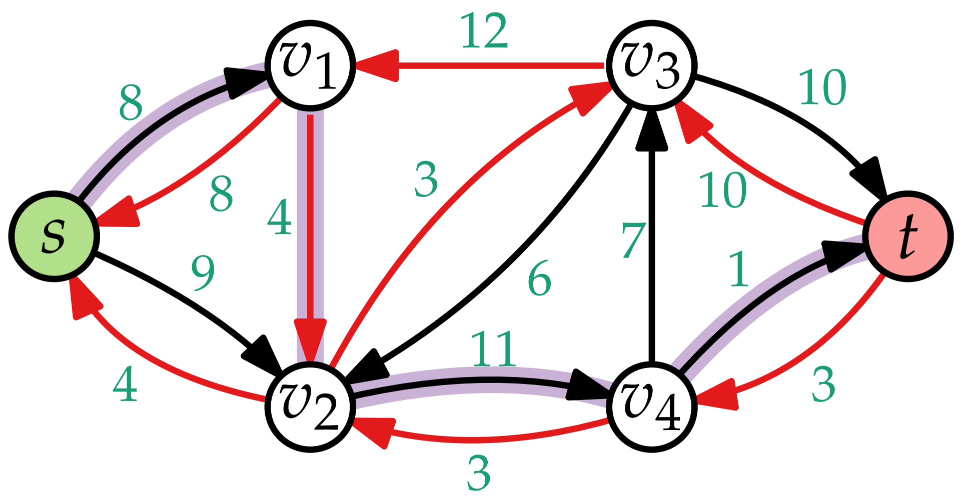
Flow-increasing path W

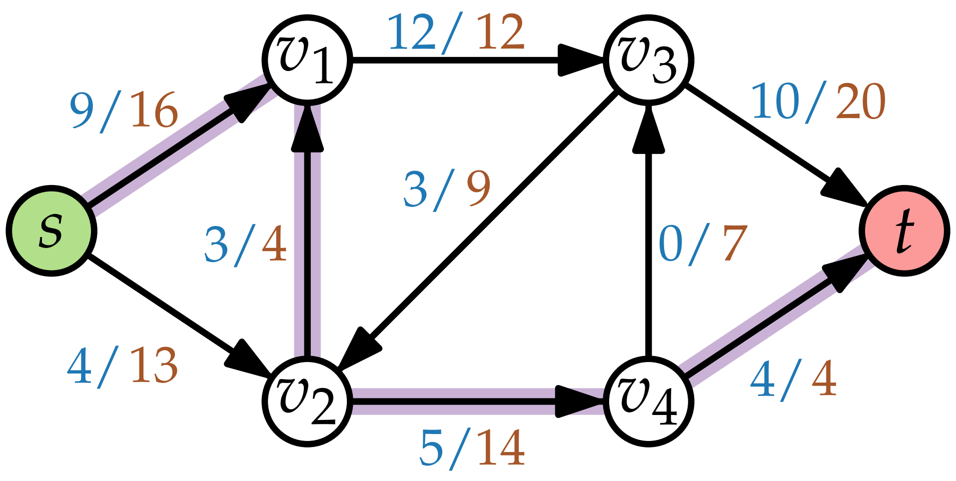
Residual Network

Residual network $G_X = (V, E')$:

- $X(v,v') < u(v,v') \Rightarrow (v,v') \in E'$
  $c(v,v') = u(v,v') - (v,v')$
- $X(v,v') > 0 \Rightarrow (v',v) \in E'$
  $c(v,v') = u(v,v')$

Flow network $(G = (V,E); s, t; u)$





Flow-increasing path W

FordFulkerson

FordFulkerson($G = (V, E); s, t; u$)

FordFulkerson

FordFulkerson($G = (V, E); s, t; u$)

foreach $(v, v') \in E$ **do**

  $X(v, v') = 0$

return X

FordFulkerson

FordFulkerson($G = (V, E); s, t; u$)

foreach $(v, v') \in E$ **do**

  $X(v, v') = 0$

return X

} Initialization with Zero-flow

FordFulkerson

FordFulkerson($G = (V, E); s, t; u$)

foreach $(v, v') \in E$ **do**
  $X(v, v') = 0$

} Initialization with Zero-flow

return X

} Max Flow

FordFulkerson

FordFulkerson($G = (V, E); s, t; u$)

foreach $(v, v') \in E$ **do**
  $X(v, v') = 0$

while G_X contains s-t-path W **do**

return X

} Initialization with Zero-flow

} Max Flow

FordFulkerson

FordFulkerson($G = (V, E); s, t; u$)

foreach $(v, v') \in E$ **do**
    $X(v, v') = 0$

} Initialization with Zero-flow

while G_X contains s-t-path W **do**
    $\Delta_W = \min_{(v,v') \in W} c(v, v')$

return X

} Max Flow

FordFulkerson

FordFulkerson($G = (V, E); s, t; u$)

foreach $(v, v') \in E$ **do**
- $X(v, v') = 0$

while G_X contains s-t-path W **do**
- $\Delta_W = \min_{(v,v') \in W} c(v, v')$

return X

} Initialization with Zero-flow

} Capacity of W

} Max Flow

FordFulkerson

FordFulkerson($G = (V, E); s, t; u$)

foreach $(v, v') \in E$ **do**
  $X(v, v') = 0$

while G_X contains s-t-path W **do**
  $\Delta_W = \min_{(v,v') \in W} c(v, v')$
  foreach $(v, v') \in W$ **do**

return X

} Initialization with Zero-flow

} Capacity of W

} Max Flow

FordFulkerson

FordFulkerson($G = (V, E); s, t; u$)

foreach $(v, v') \in E$ **do**
- $X(v, v') = 0$

while G_X contains s-t-path W **do**
- $\Delta_W = \min_{(v,v') \in W} c(v, v')$
- **foreach** $(v, v') \in W$ **do**
  - **if** $(v, v') \in E$ **then**
    - $X(v, v') = X(v, v') + \Delta_W$

return X

} Initialization with Zero-flow

} Capacity of W

} Max Flow

FordFulkerson

```
FordFulkerson(G = (V, E); s, t; u)
  foreach (v, v') ∈ E do
  └ X(v, v') = 0                                   } Initialization with Zero-flow
  while G_X contains s-t-path W do
  │ Δ_W = min_{(v,v')∈W} c(v, v')                  } Capacity of W
  │ foreach (v, v') ∈ W do
  │ │ if (v, v') ∈ E then
  │ │ │ X(v, v') = X(v, v') + Δ_W
  │ │ else
  │ │ └ X(v, v') = X(v, v') − Δ_W
  │ └
  └
  return X                                         } Max Flow
```

FordFulkerson

FordFulkerson($G = (V, E); s, t; u$)

foreach $(v, v') \in E$ **do**
- $X(v, v') = 0$

while G_X contains s-t-path W **do**
- $\Delta_W = \min_{(v,v') \in W} c(v, v')$
- **foreach** $(v, v') \in W$ **do**
  - **if** $(v, v') \in E$ **then**
    - $X(v, v') = X(v, v') + \Delta_W$
  - **else**
    - $X(v, v') = X(v, v') - \Delta_W$

return X

} Initialization with Zero-flow

} Capacity of W

} Increasing flow along W

} Max Flow

FordFulkerson

FordFulkerson($G = (V, E); s, t; u$)

foreach $(v, v') \in E$ **do**
- $X(v, v') = 0$

while G_X contains s-t-path W **do**
- $\Delta_W = \min_{(v,v') \in W} c(v, v')$
- **foreach** $(v, v') \in W$ **do**
  - **if** $(v, v') \in E$ **then**
    - $X(v, v') = X(v, v') + \Delta_W$
  - **else**
    - $X(v, v') = X(v, v') - \Delta_W$

return X

} Initialization with Zero-flow

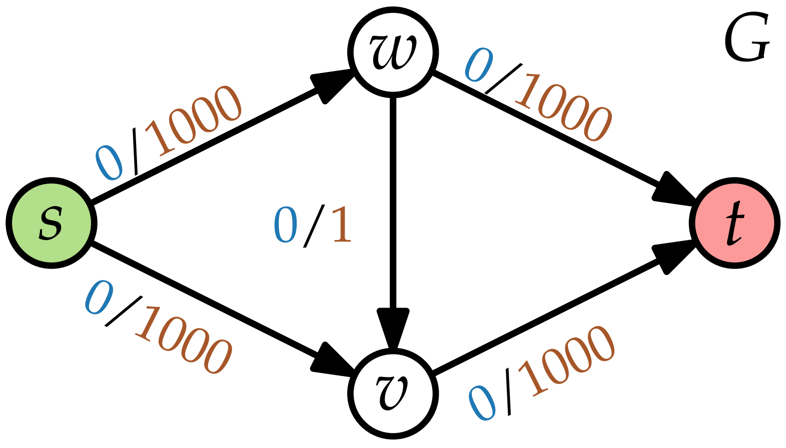
} Capacity of W

} Increasing flow along W

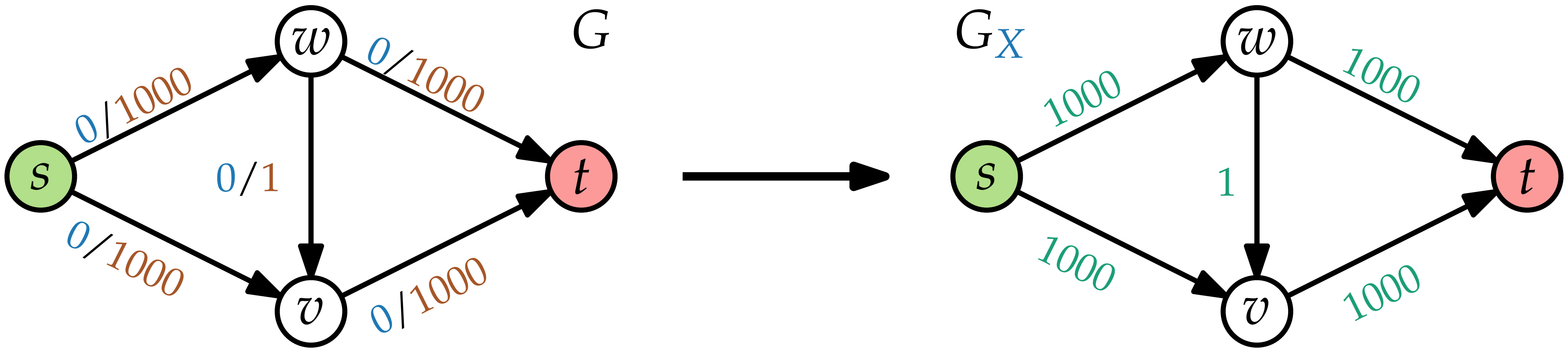
} Max Flow

FordFulkerson finds a maximum s-t-flow in $O(|X^*| \cdot n)$ time.

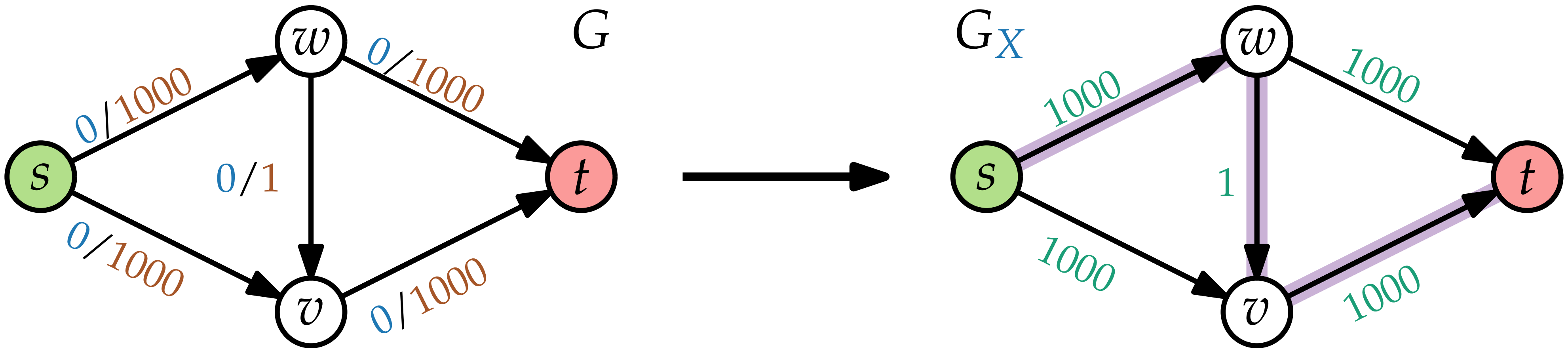
FordFulkerson – Example



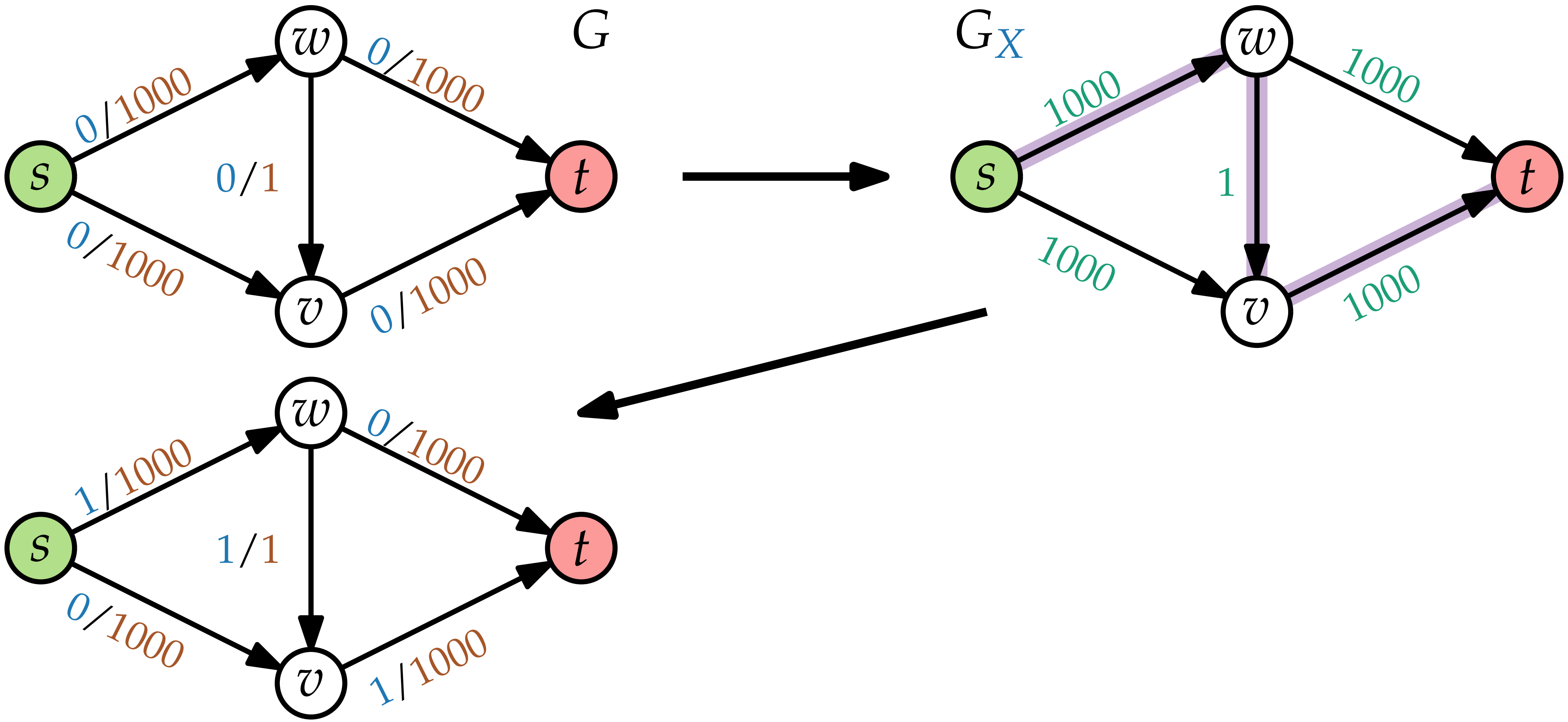
FordFulkerson – Example



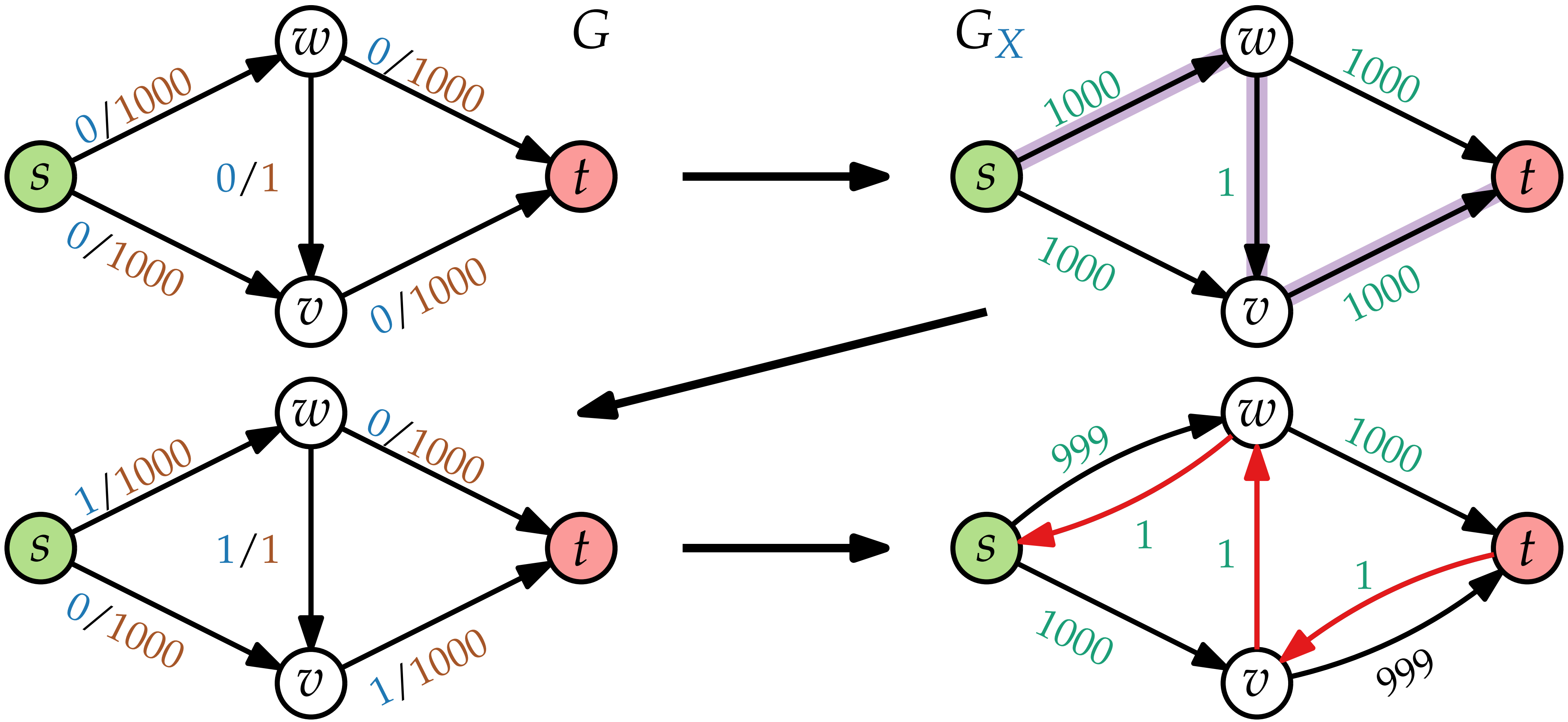
FordFulkerson – Example



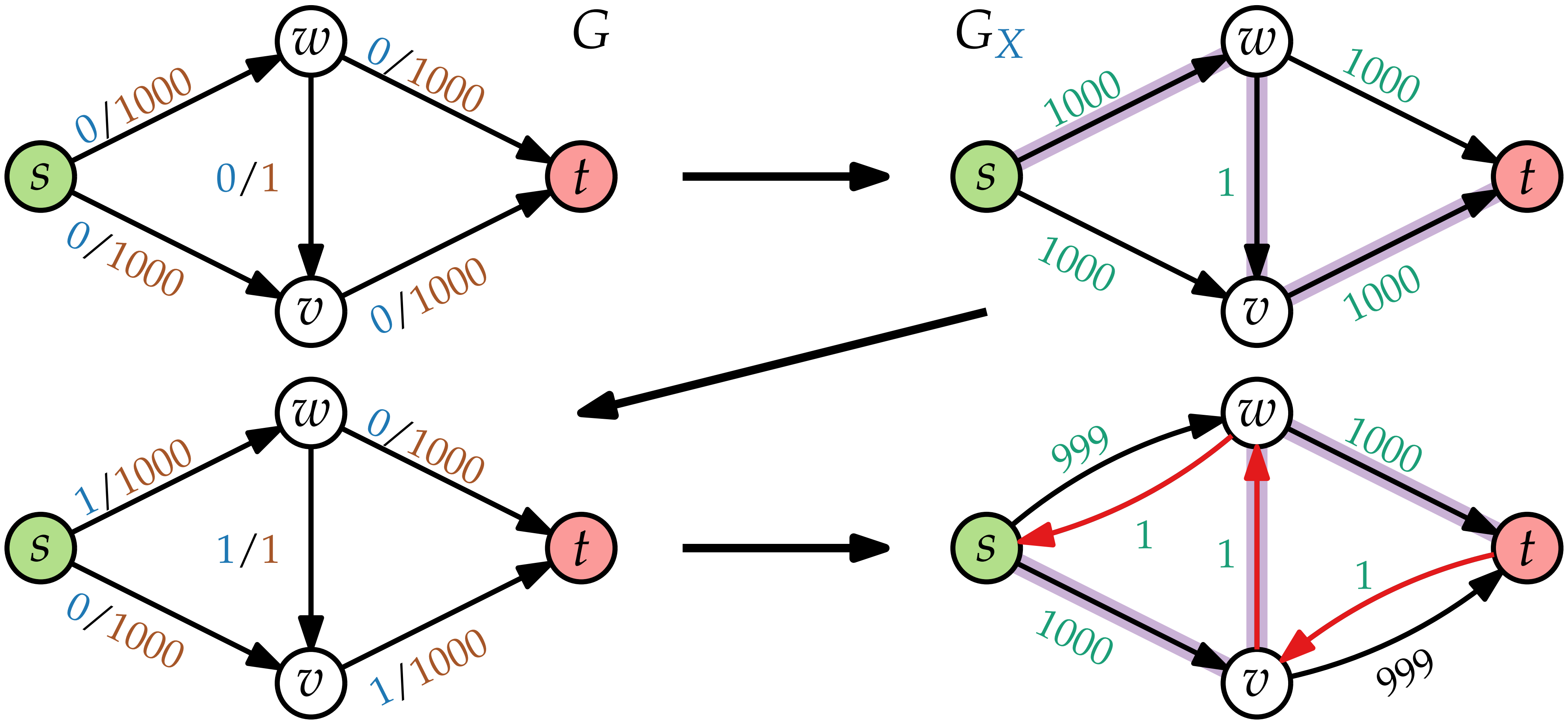
FordFulkerson – Example



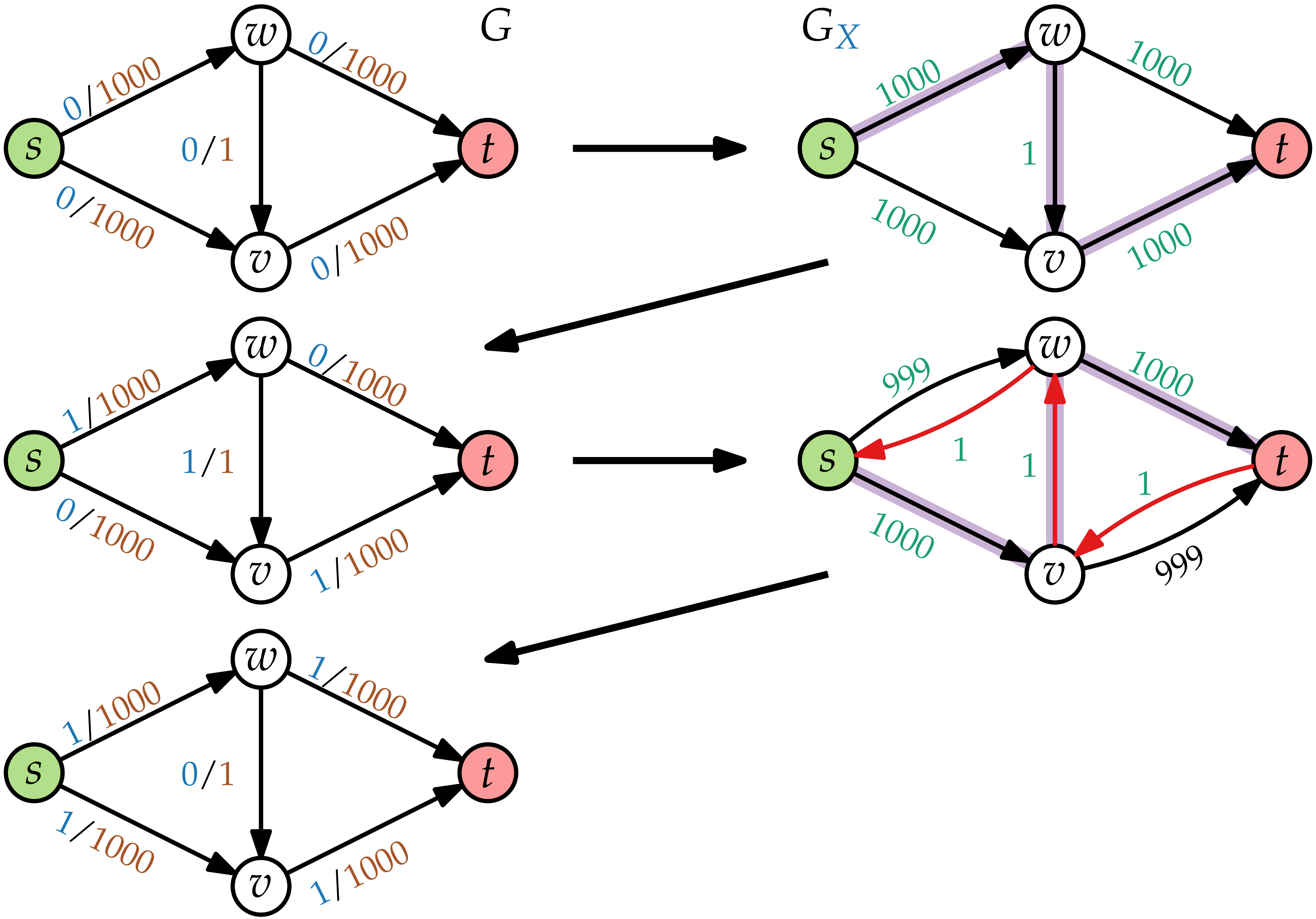
FordFulkerson – Example



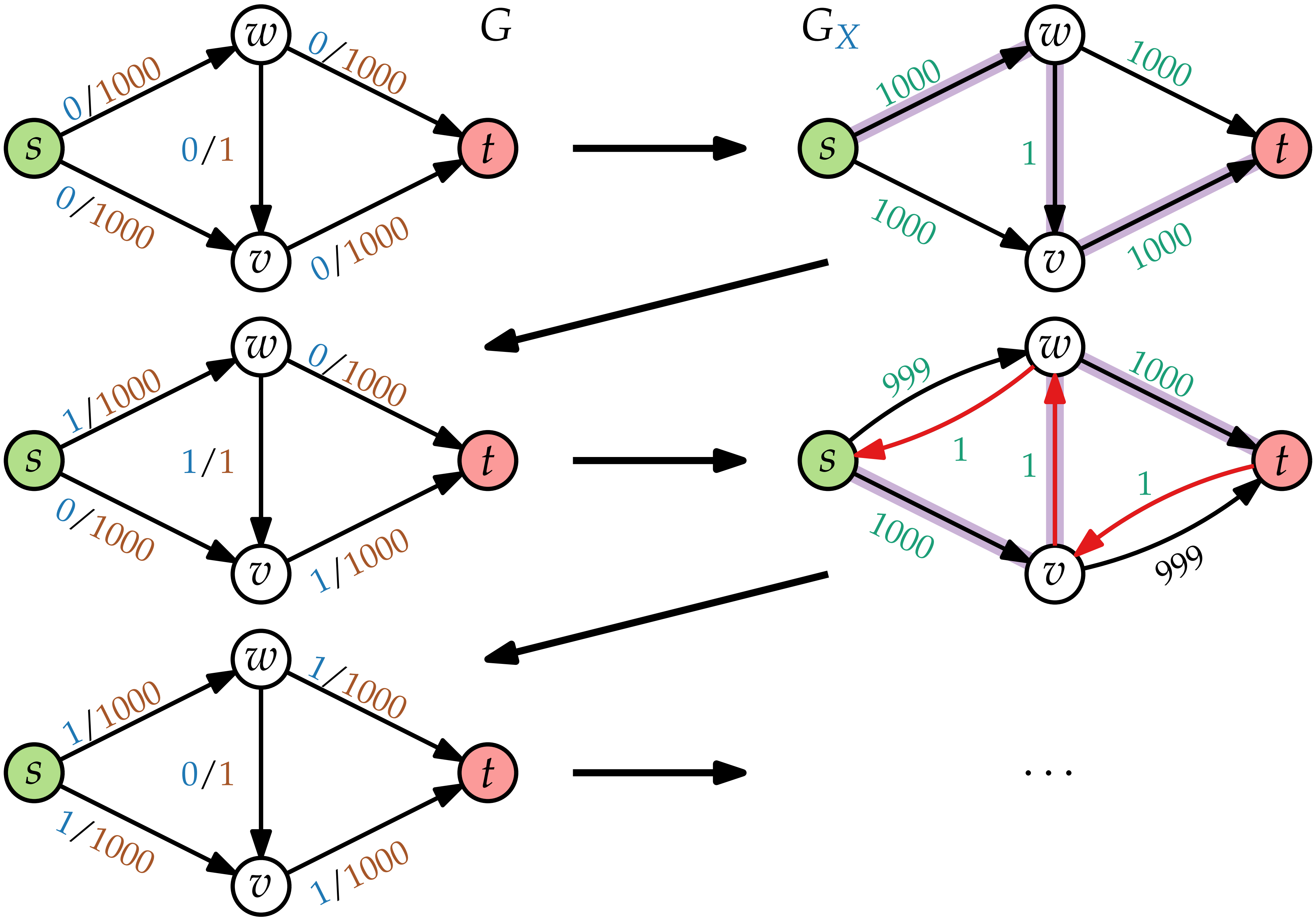
FordFulkerson – Example



FordFulkerson – Example



FordFulkerson – Example



EdmondsKarp

FordFulkerson($G = (V, E); s, t; u$)

  foreach $(v, v') \in E$ **do**

    $X(v, v') = 0$

  while G_X contains s-t-path W **do**

    $W =$ s-t-path in G_X

    $\Delta_W = \min_{(v,v') \in c(v,v')}$

    foreach $(v, v') \in W$ **do**

      if $(v, v') \in E$ **then**

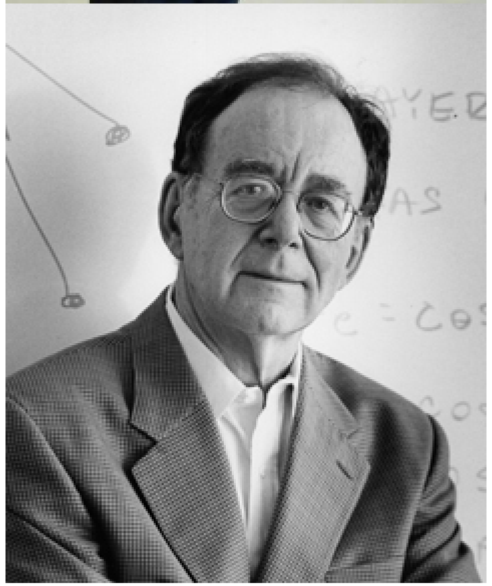
        $X(v, v') = X(v, v') + \Delta_W$

      else

        $X(v, v') = X(v, v') - \Delta_W$

  return X

Jack R. Edmonds
*1934



Richard M. Karp
*1935 Boston, MA

EdmondsKarp

EdmondsKarp

~~FordFulkerson~~$(G = (V, E); s, t; u)$

foreach $(v, v') \in E$ **do**

  $X(v, v') = 0$

while G_X contains s-t-path W **do**

  $W =$ s-t-path in G_X

  $\Delta_W = \min_{(v,v') \in c(v,v')}$

  foreach $(v, v') \in W$ **do**

    if $(v, v') \in E$ **then**

      $X(v, v') = X(v, v') + \Delta_W$

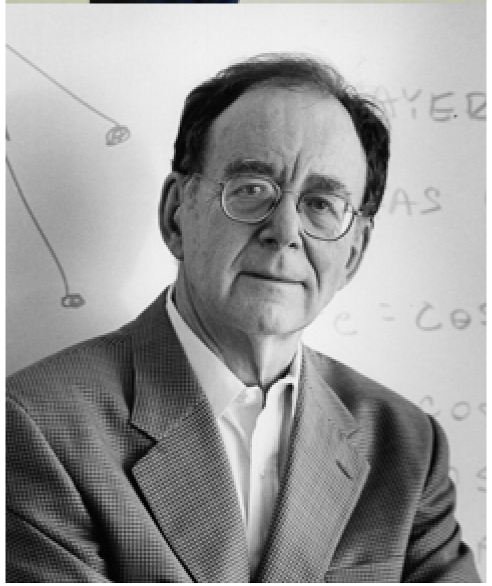
    else

      $X(v, v') = X(v, v') - \Delta_W$

return X

Jack R. Edmonds *1934

Richard M. Karp *1935 Boston, MA



EdmondsKarp

EdmondsKarp

~~FordFulkerson~~$(G = (V, E); s, t; u)$

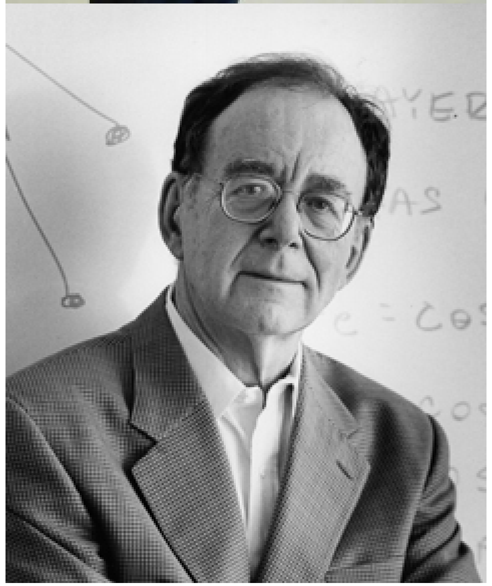
foreach $(v, v') \in E$ **do**
  $X(v, v') = 0$

while G_X contains s-t-path W **do**
  $W =$ shortest s-t-path in G_X
  $\Delta_W = \min_{(v,v') \in c(v,v')}$
  foreach $(v, v') \in W$ **do**
    if $(v, v') \in E$ **then**
      $X(v, v') = X(v, v') + \Delta_W$
    else
      $X(v, v') = X(v, v') - \Delta_W$

return X

Jack R. Edmonds
*1934

Richard M. Karp
*1935 Boston, MA



EdmondsKarp

EdmondsKarp
~~FordFulkerson~~$(G = (V, E); s, t; u)$

foreach $(v, v') \in E$ **do**
  $X(v, v') = 0$

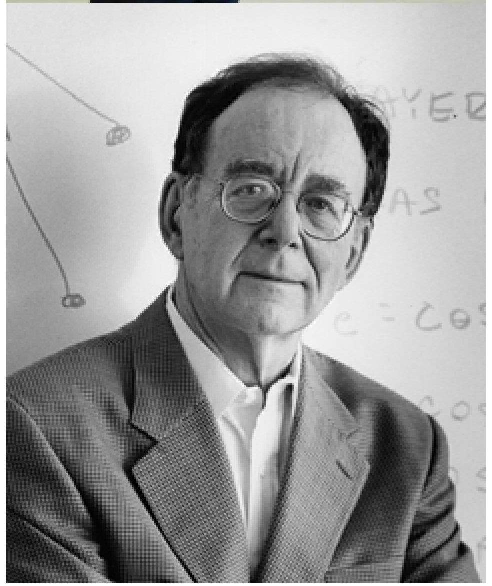
while G_X contains s-t-path W **do**
  $W =$ shortest s-t-path in G_X
  $\Delta_W = \min_{(v,v') \in c(v,v')}$
  foreach $(v, v') \in W$ **do**
    if $(v, v') \in E$ **then**
      $X(v, v') = X(v, v') + \Delta_W$
    else
      $X(v, v') = X(v, v') - \Delta_W$

return X

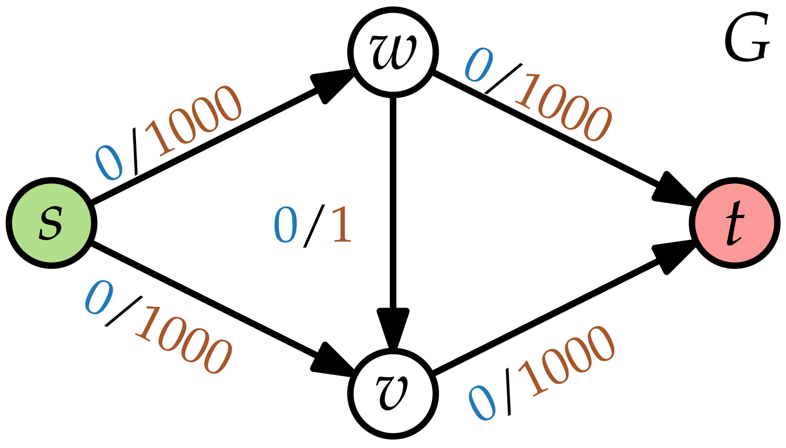
EdmondsKarp finds a maximum s-t-flow in $O(nm^2)$ time.

Jack R. Edmonds *1934

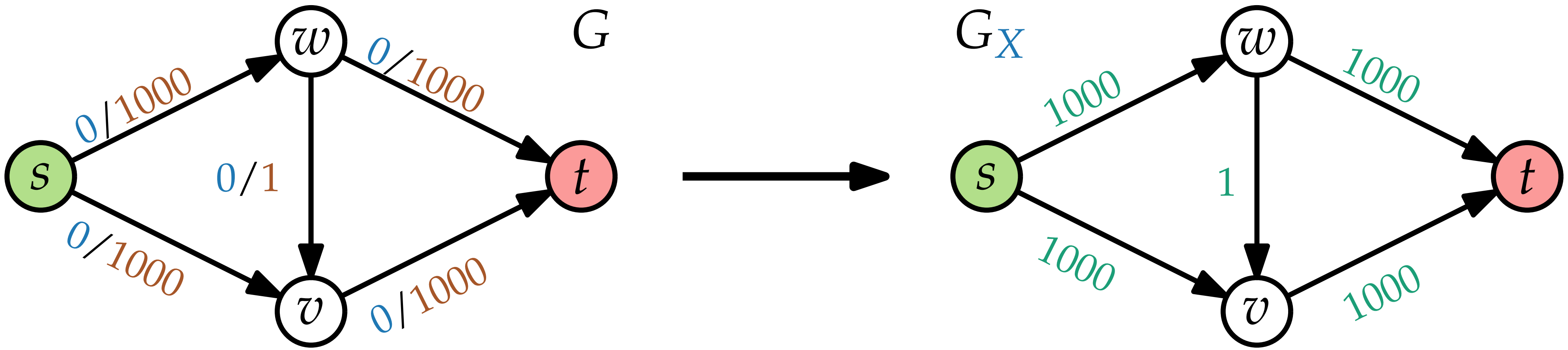
Richard M. Karp *1935 Boston, MA



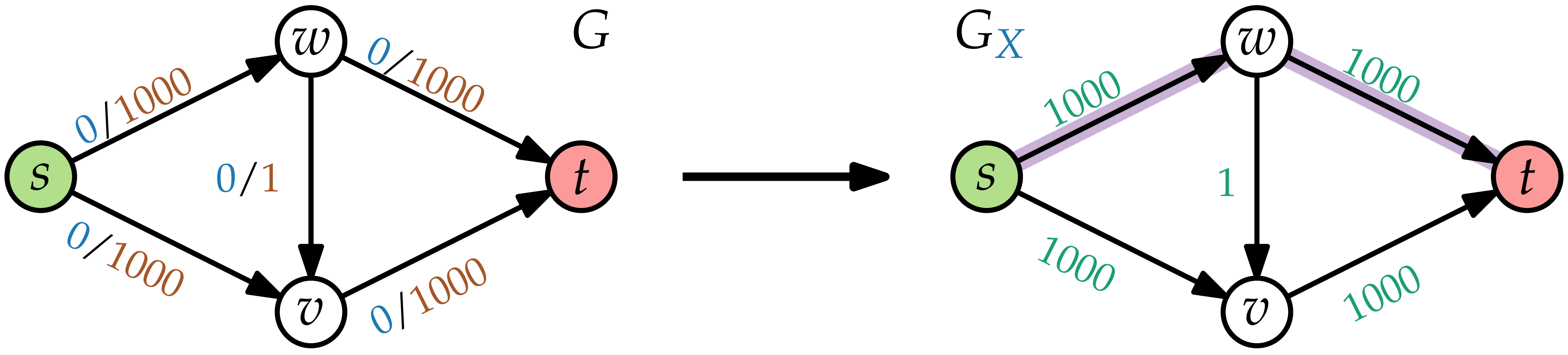
EdmondsKarp – Example



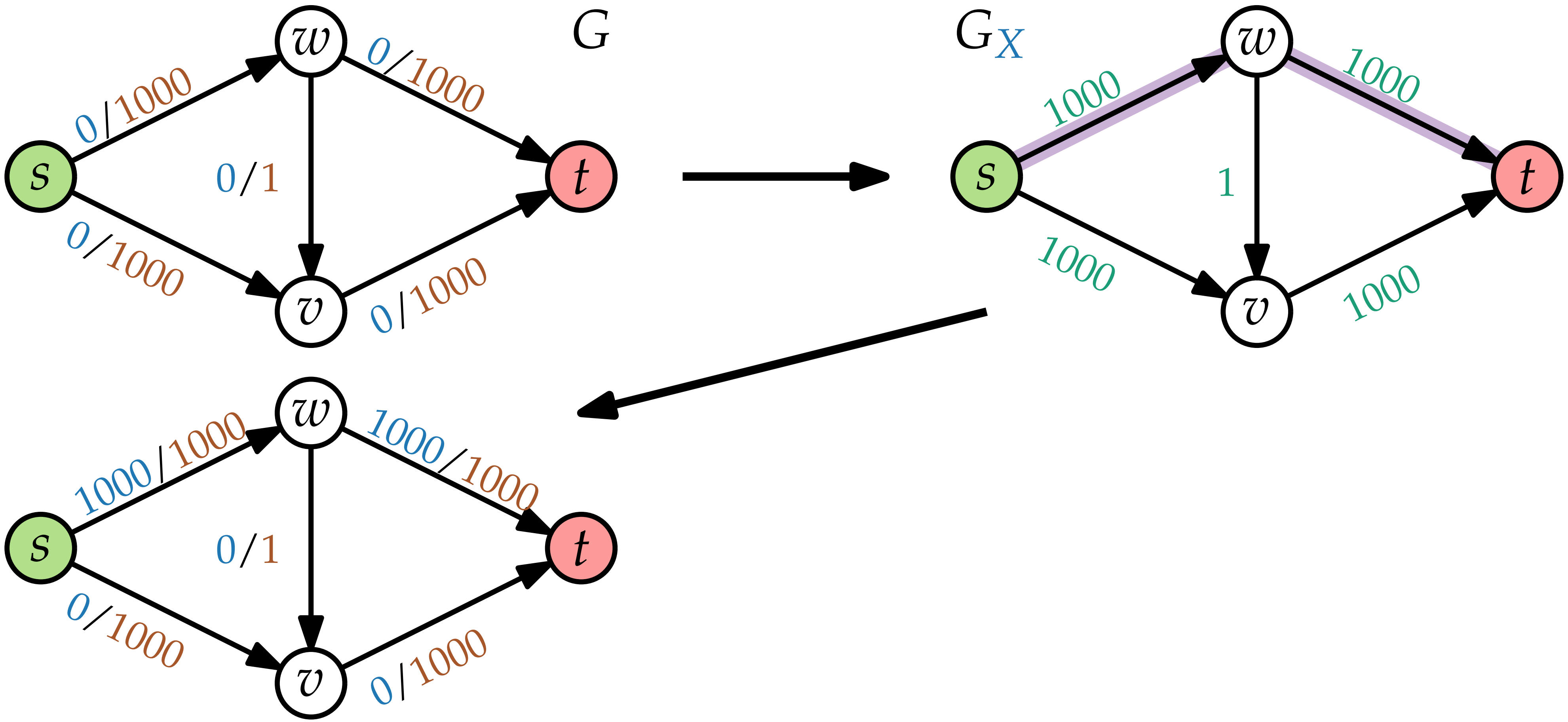
EdmondsKarp – Example



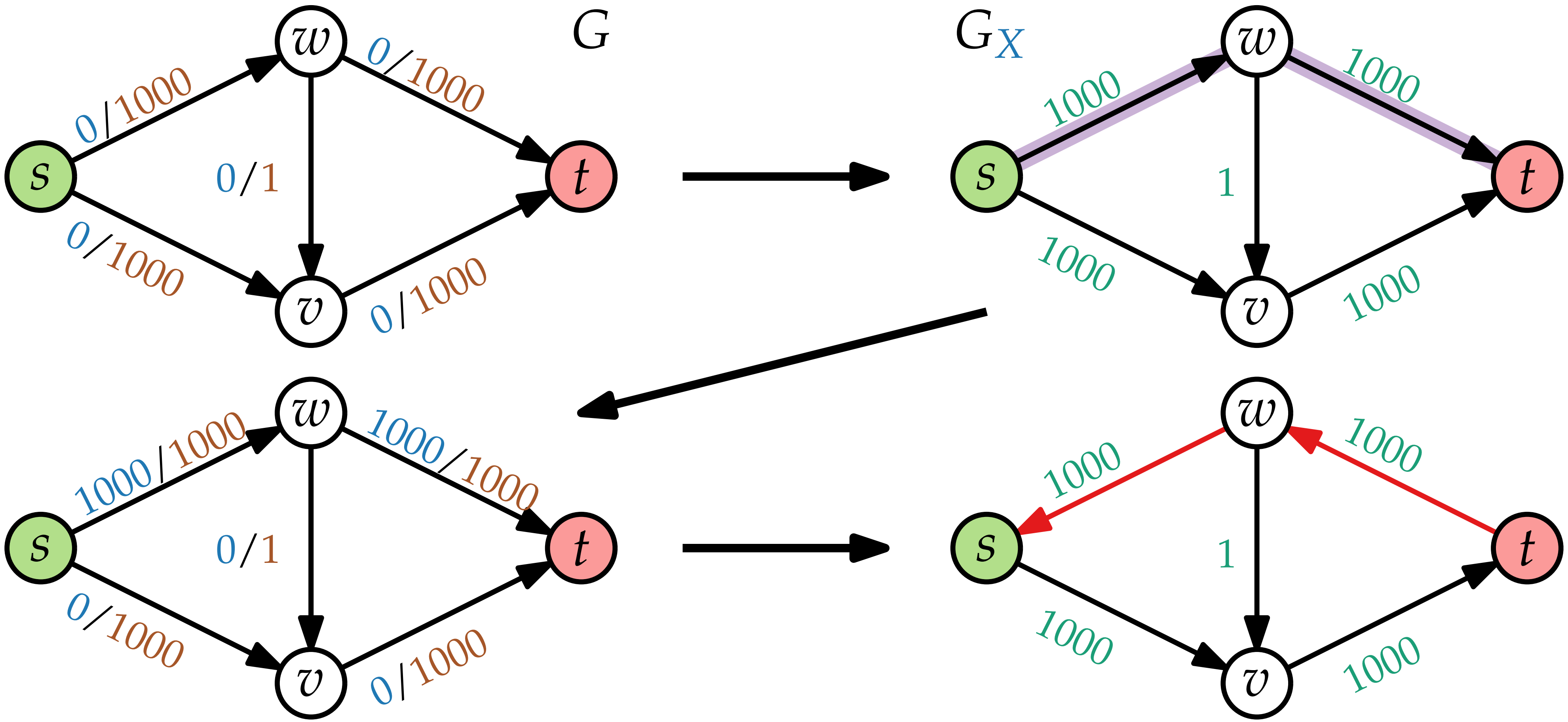
EdmondsKarp – Example



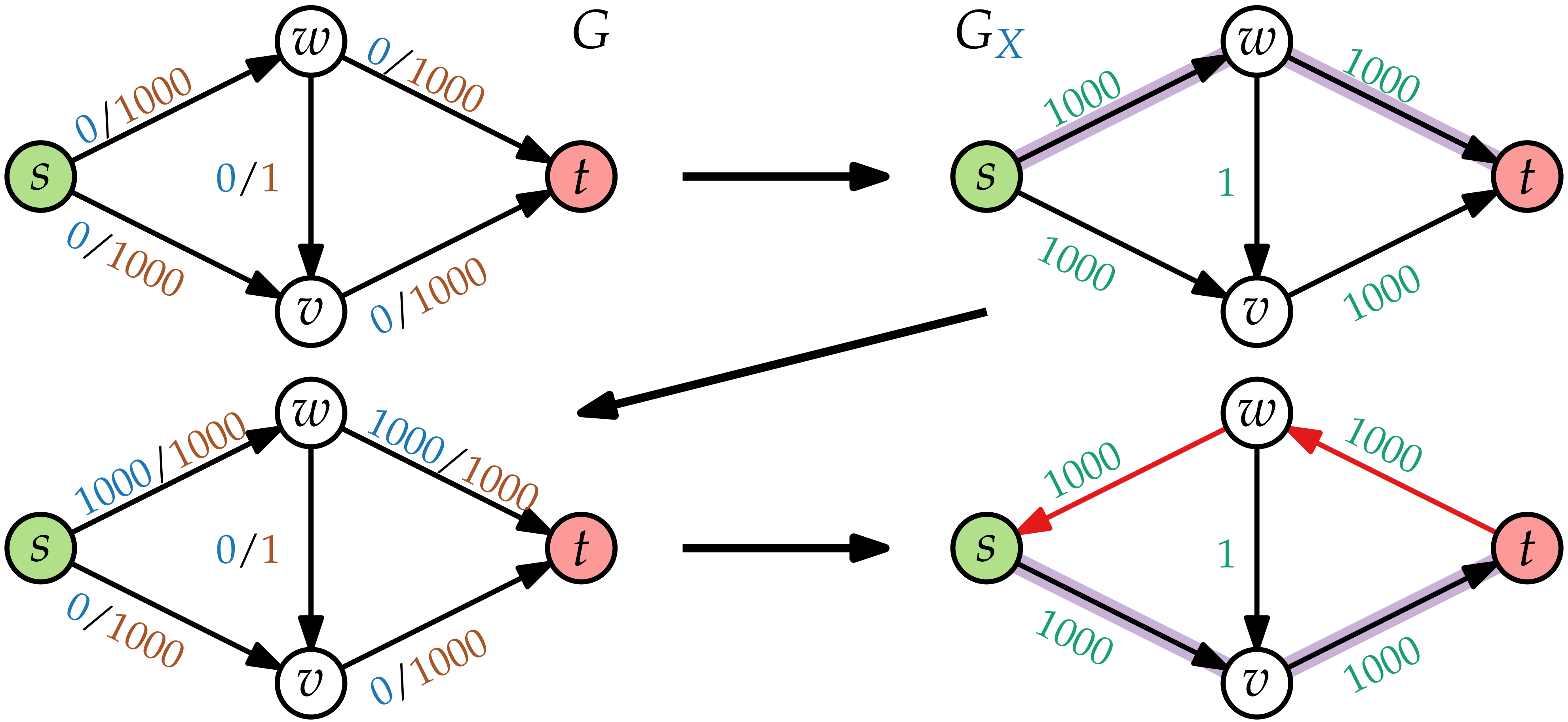
EdmondsKarp – Example



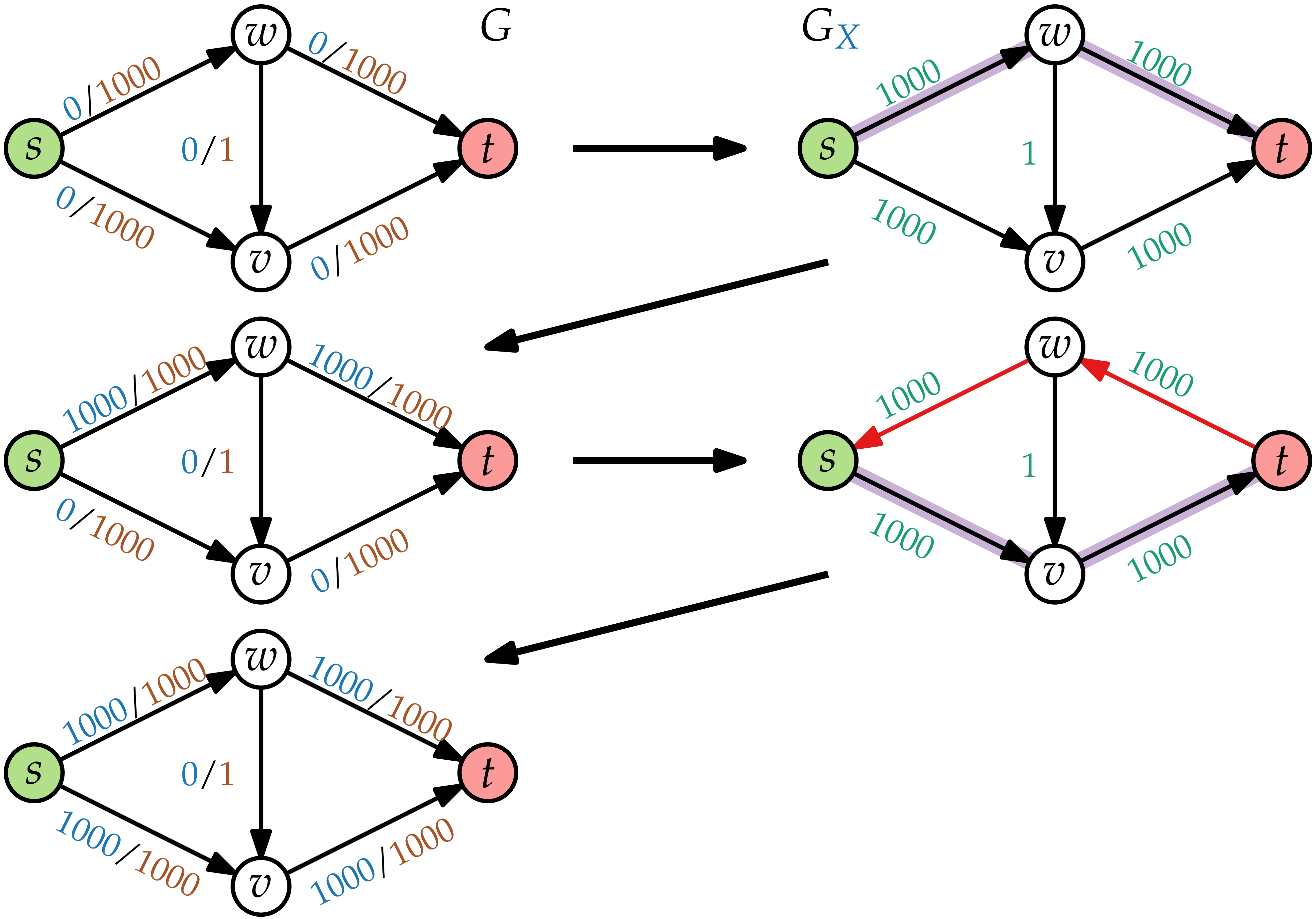
EdmondsKarp – Example



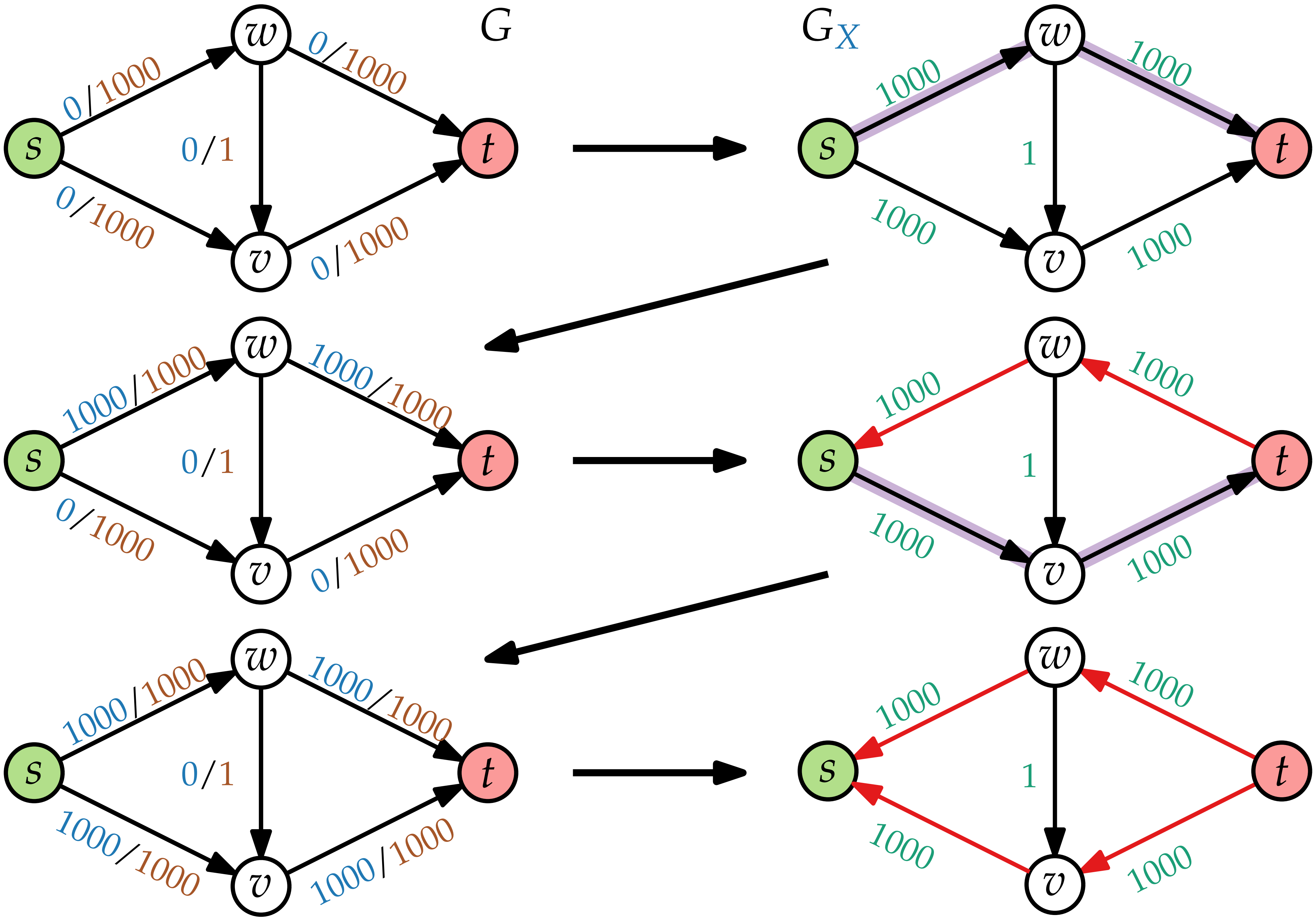
EdmondsKarp – Example



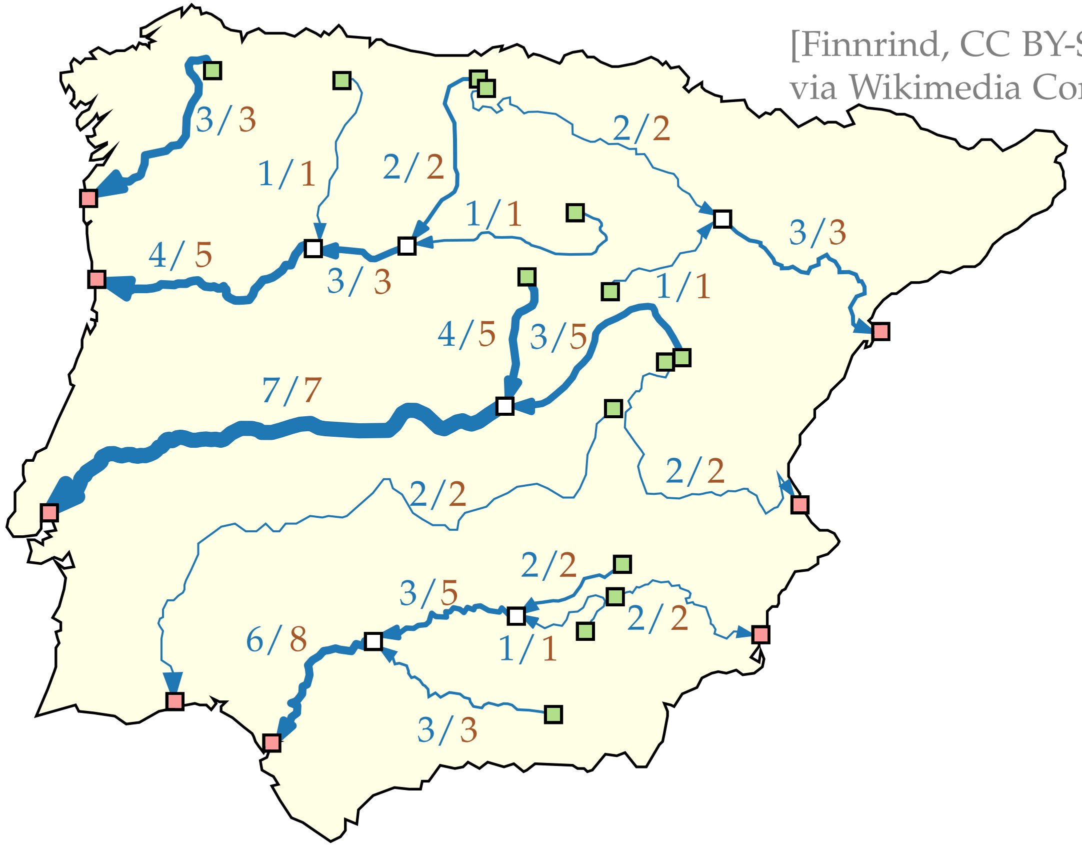
EdmondsKarp – Example



EdmondsKarp – Example



General Flow Network



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

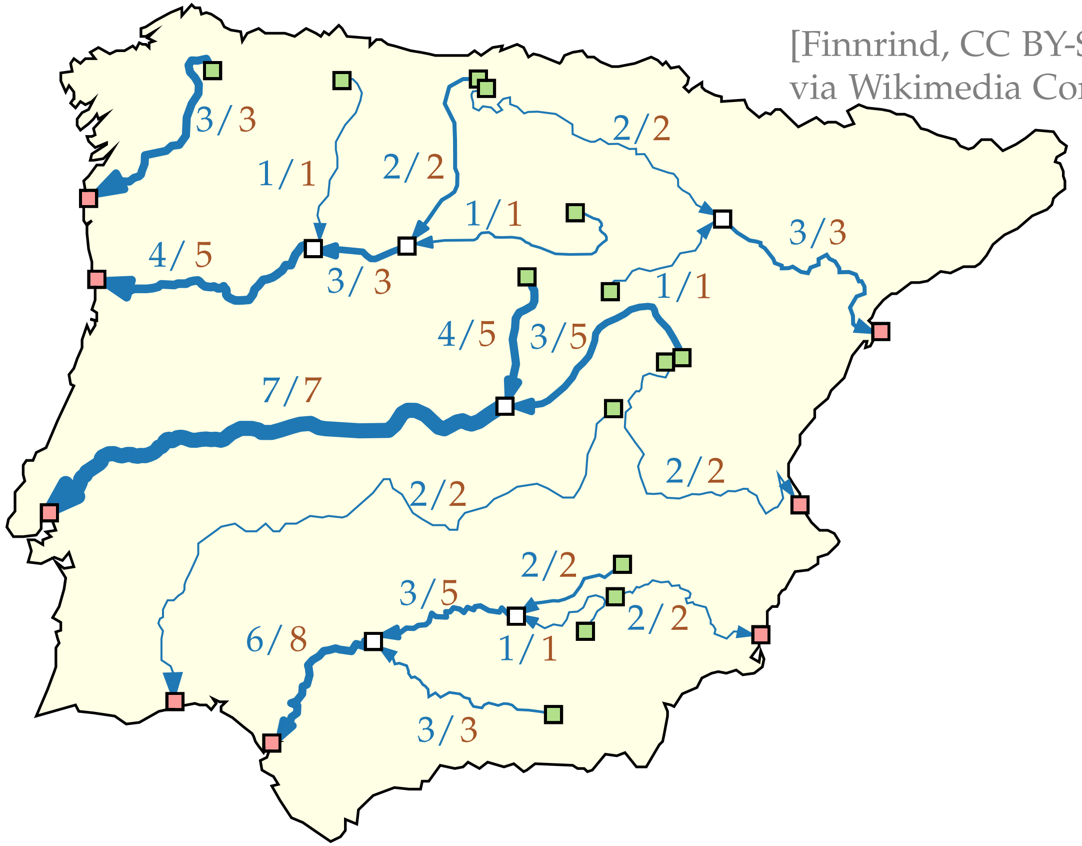
- directed graph $G = (V, E)$
- *sources* $S \subseteq V$, *sinks* $T \subseteq V$
- edge *capacity* $u \colon E \to \mathbb{R}_0^+$

A function $X \colon E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

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

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

General Flow Network



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

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

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

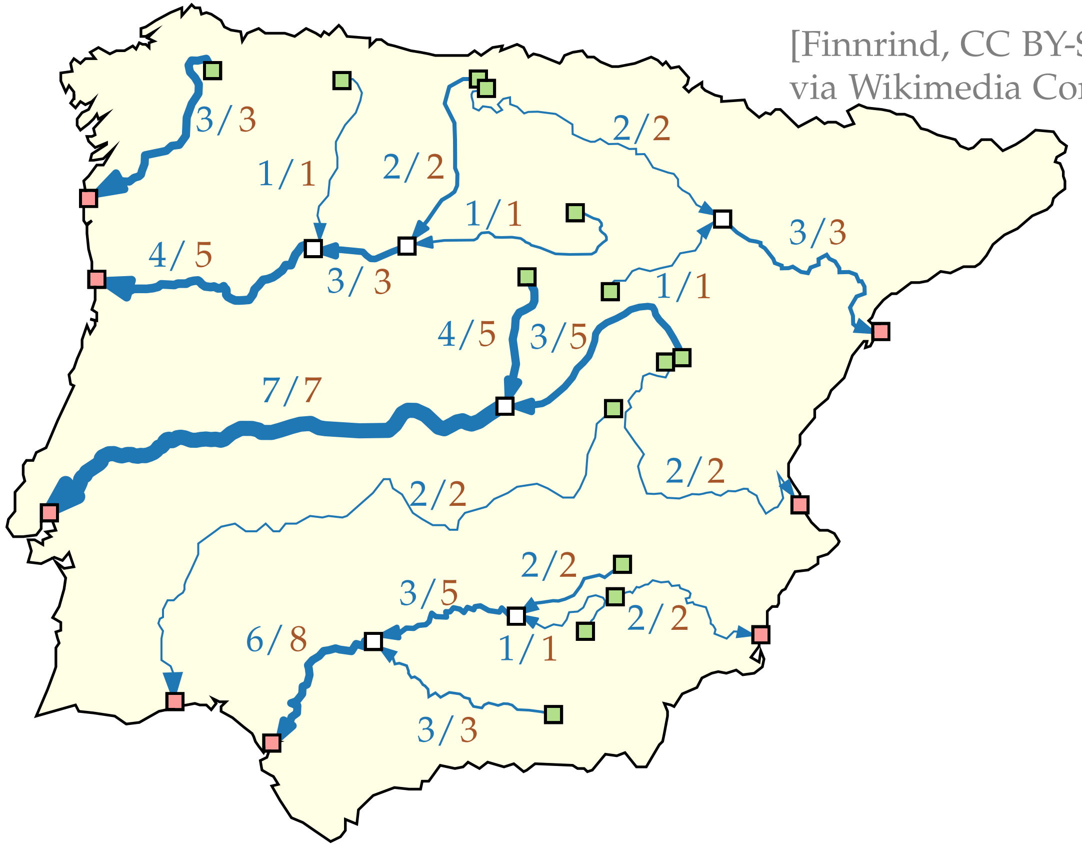
A function $X \colon E \to \mathbb{R}_0^+$ is called **S-T-flow**, if:

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

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

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

General Flow Network



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

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

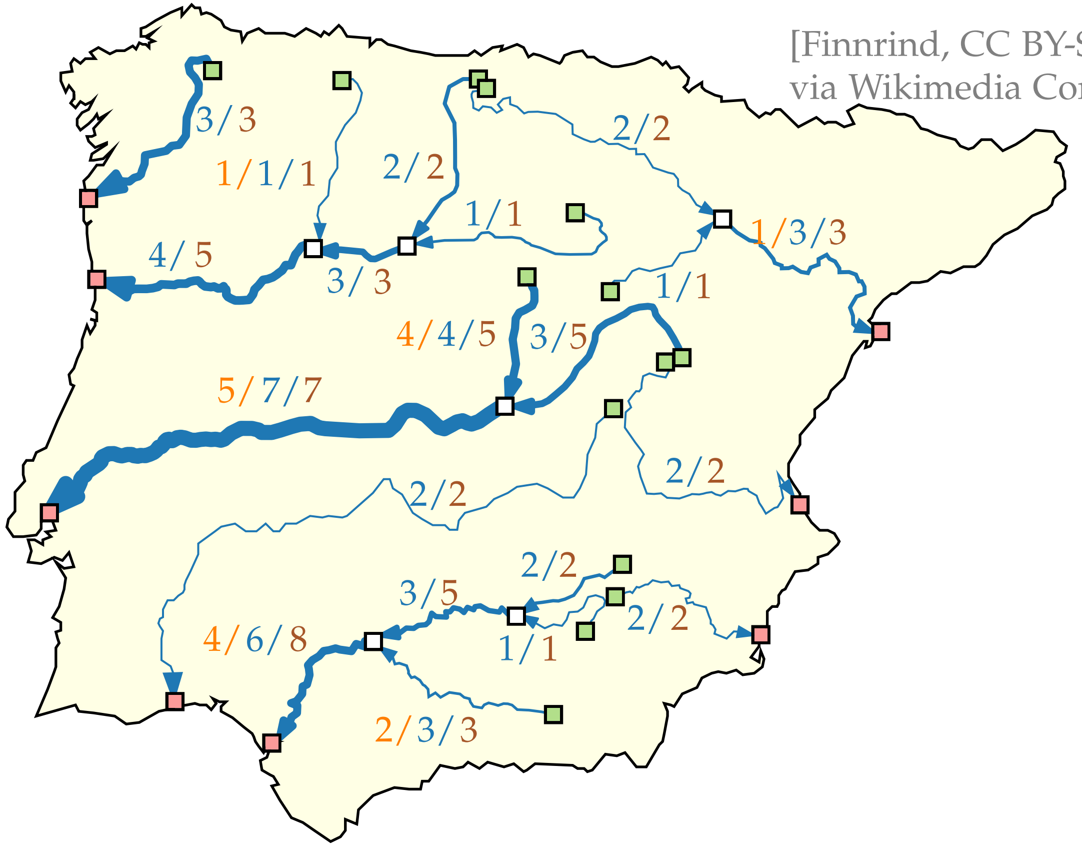
A function $X: E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

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

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

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

General Flow Network



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

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

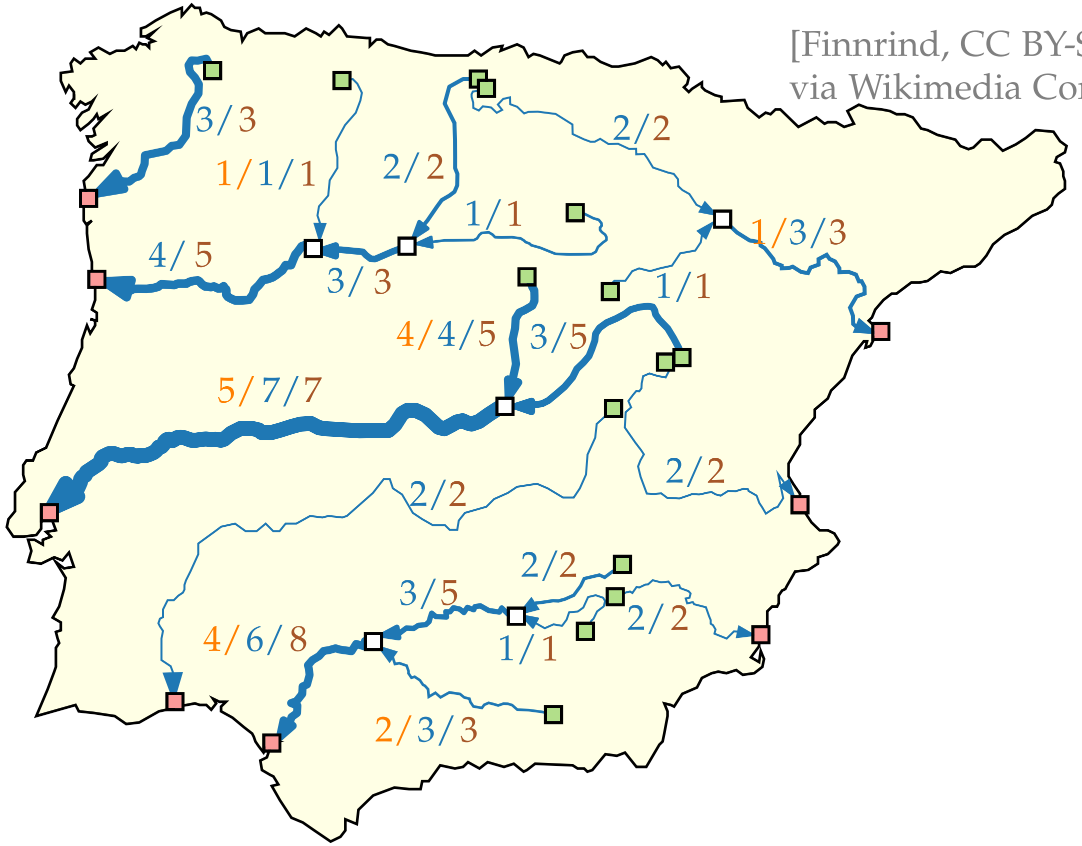
A function $X \colon E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

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

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

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

General Flow Network



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

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

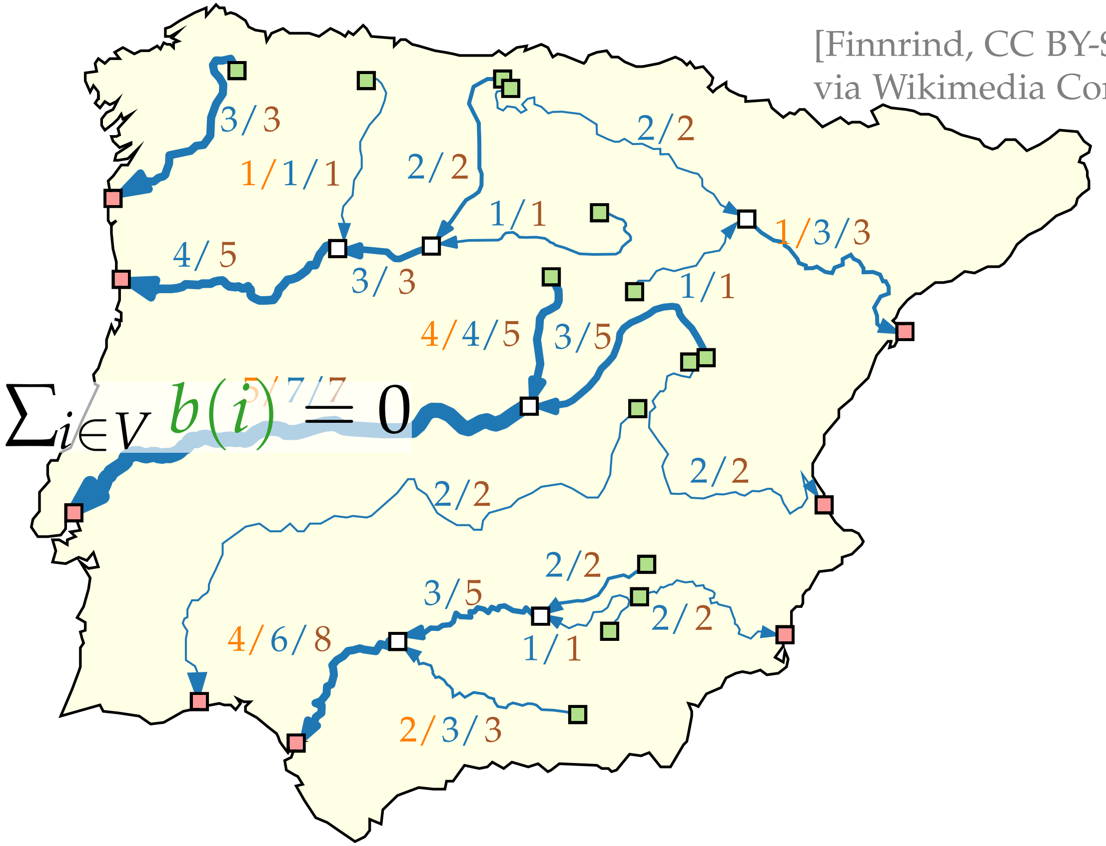
A function $X : E \to \mathbb{R}_0^+$ is called ***S*-*T*-flow**, if:

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

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

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

General Flow Network



Flow network $(G = (V, E); b; \ell; u)$ with

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

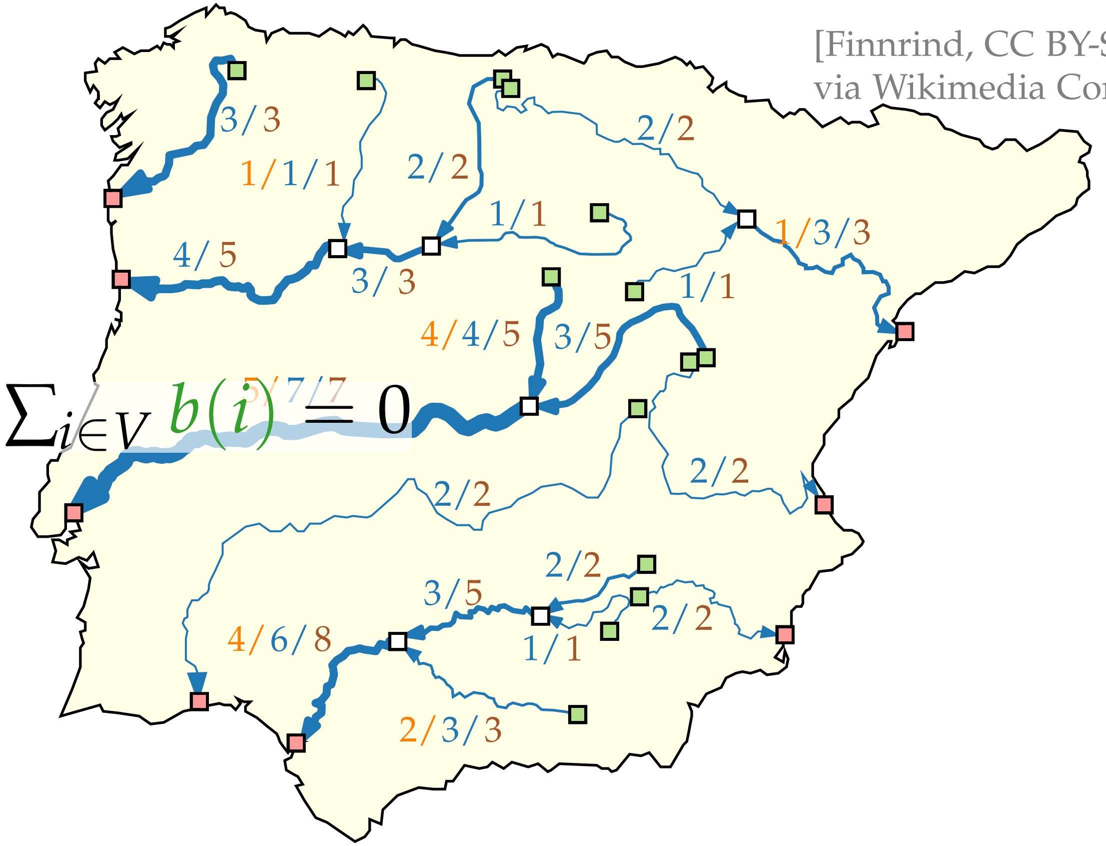
A function $X\colon E \to \mathbb{R}_0^+$ is called S-T-**flow**, if:

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

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

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

General Flow Network



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

Flow network $(G = (V, E); b; \ell; u)$ with

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

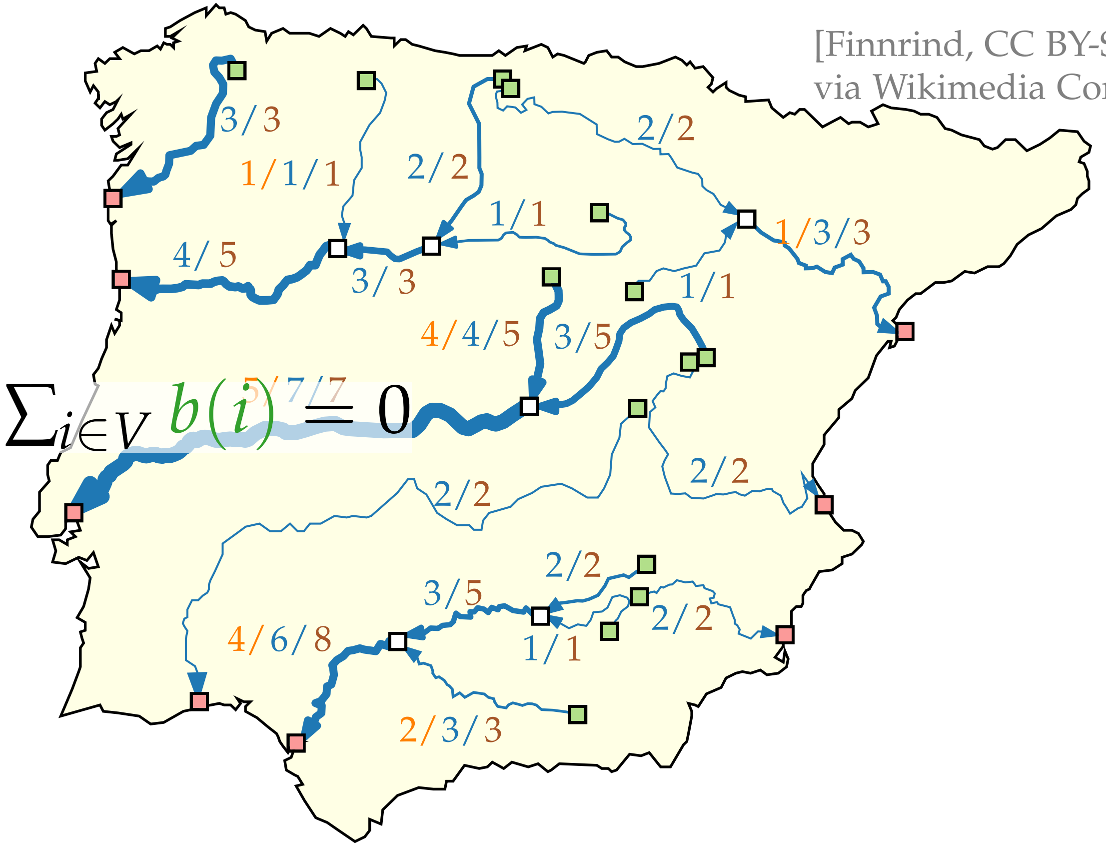
A function $X \colon E \to \mathbb{R}_0^+$ is called **valid flow**, if:

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

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

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

General Flow Network



Flow network $(G = (V, E); b; \ell; u)$ with

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

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

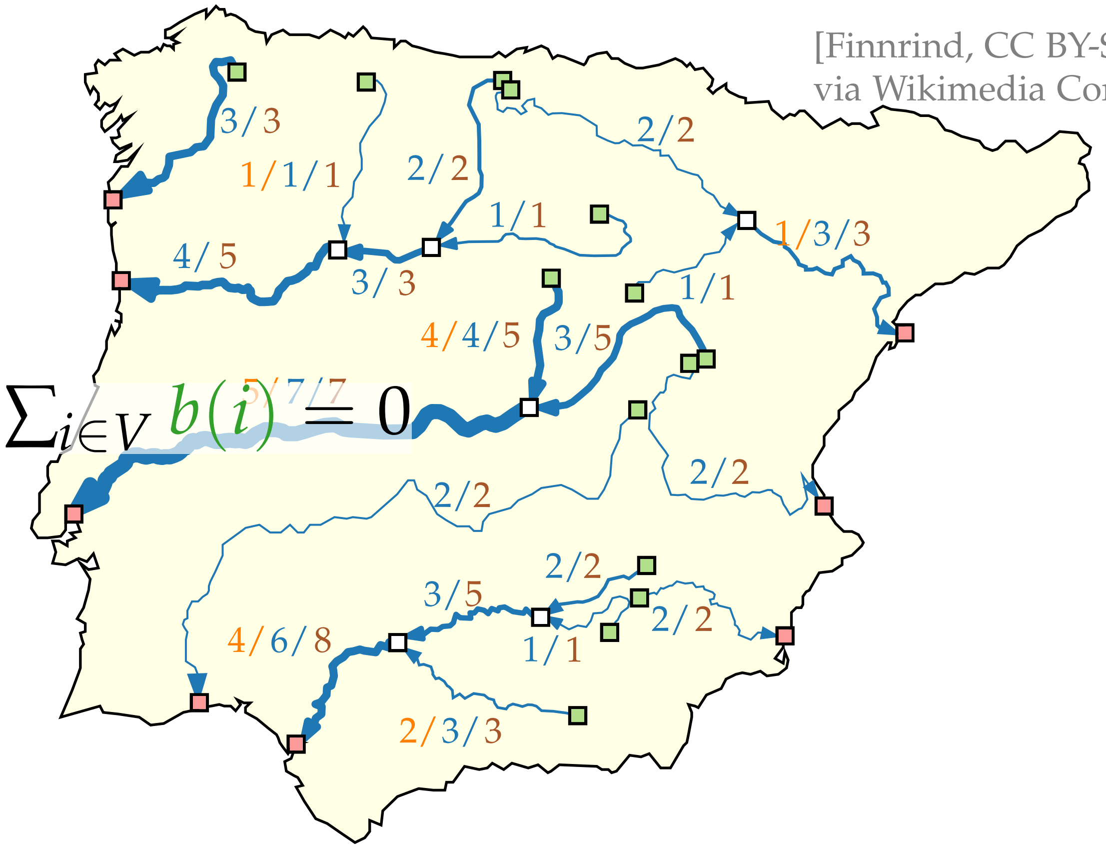
$$\ell(i,j) \leq X(i,j) \leq u(i,j) \qquad \forall (i,j) \in E$$

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

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

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

General Flow Network



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

Flow network $(G = (V, E); b; \ell; u)$ with

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

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

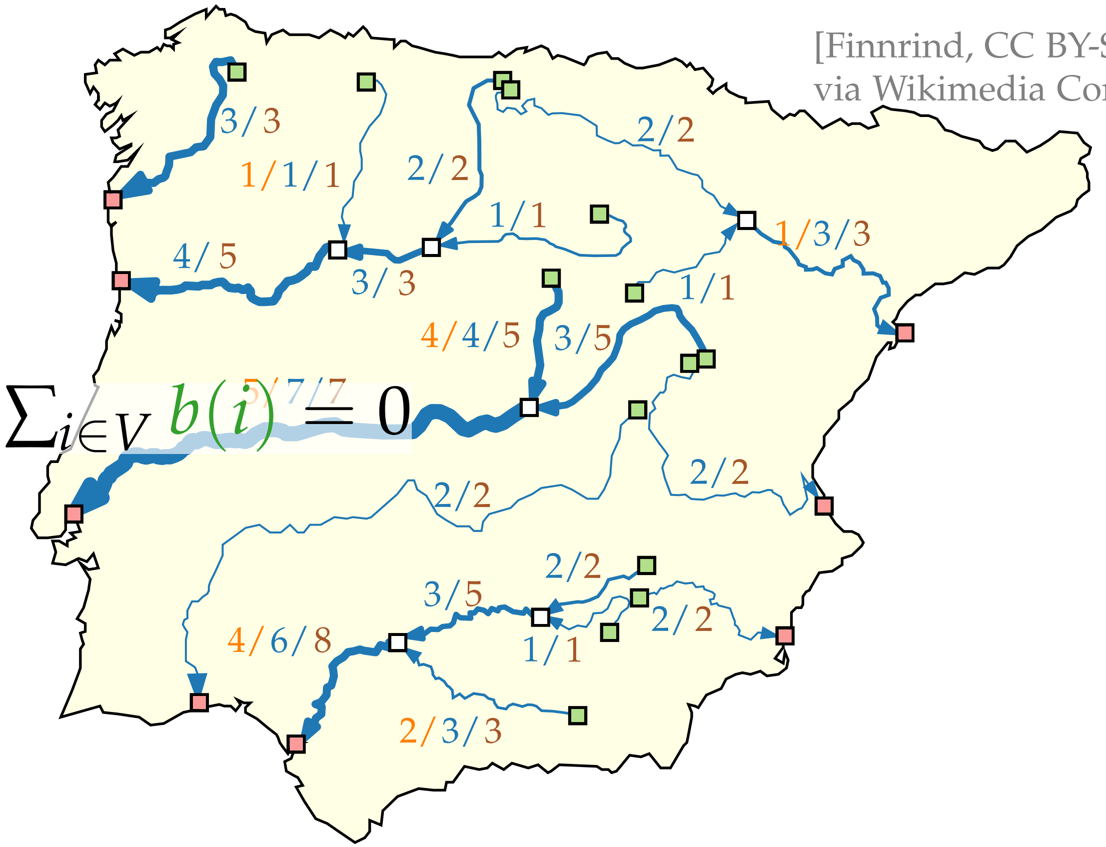
$$\ell(i,j) \le X(i,j) \le u(i,j) \qquad \forall (i,j) \in E$$

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

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

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

General Flow Network



Flow network $(G = (V, E); b; \ell; u)$ with

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

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

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

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

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

A **minimum cost flow** is a valid flow where $\text{cost}(X)$ is minimized.

General Flow Network – Algorithms

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

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]
$S(n, m, C)$	=	$O(\text{Min}\,(m + n\sqrt{\log C}),$	Ahuja, Mehlhorn, Orlin and Tarjan [1990]
		$(m \log \log C))$	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

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]

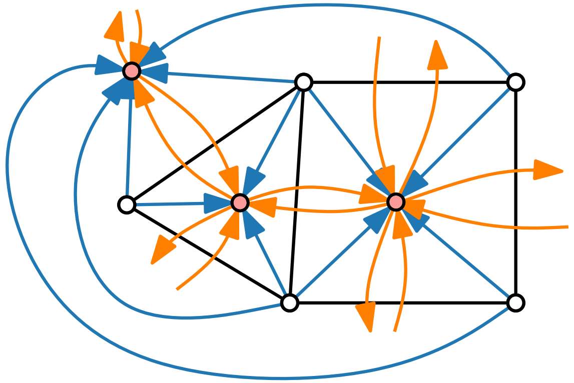
[Orlin 1991]

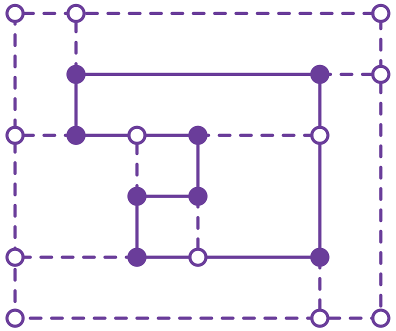
Theorem. [Orlin 1991]
The minimum cost flow problem can be solved in $O(n^2 \log^2 n + m^2 \log n)$ time.

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

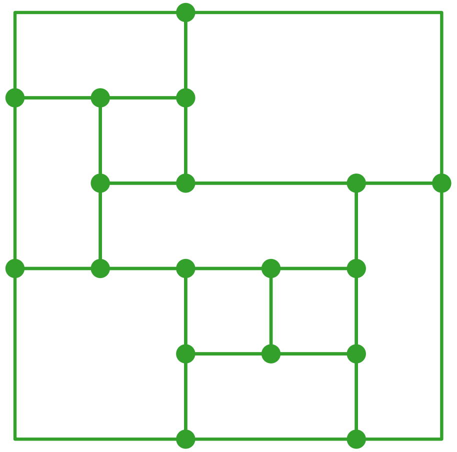
Universität Trier

Visualization of Graphs





Lecture 6:
Orthogonal Layouts



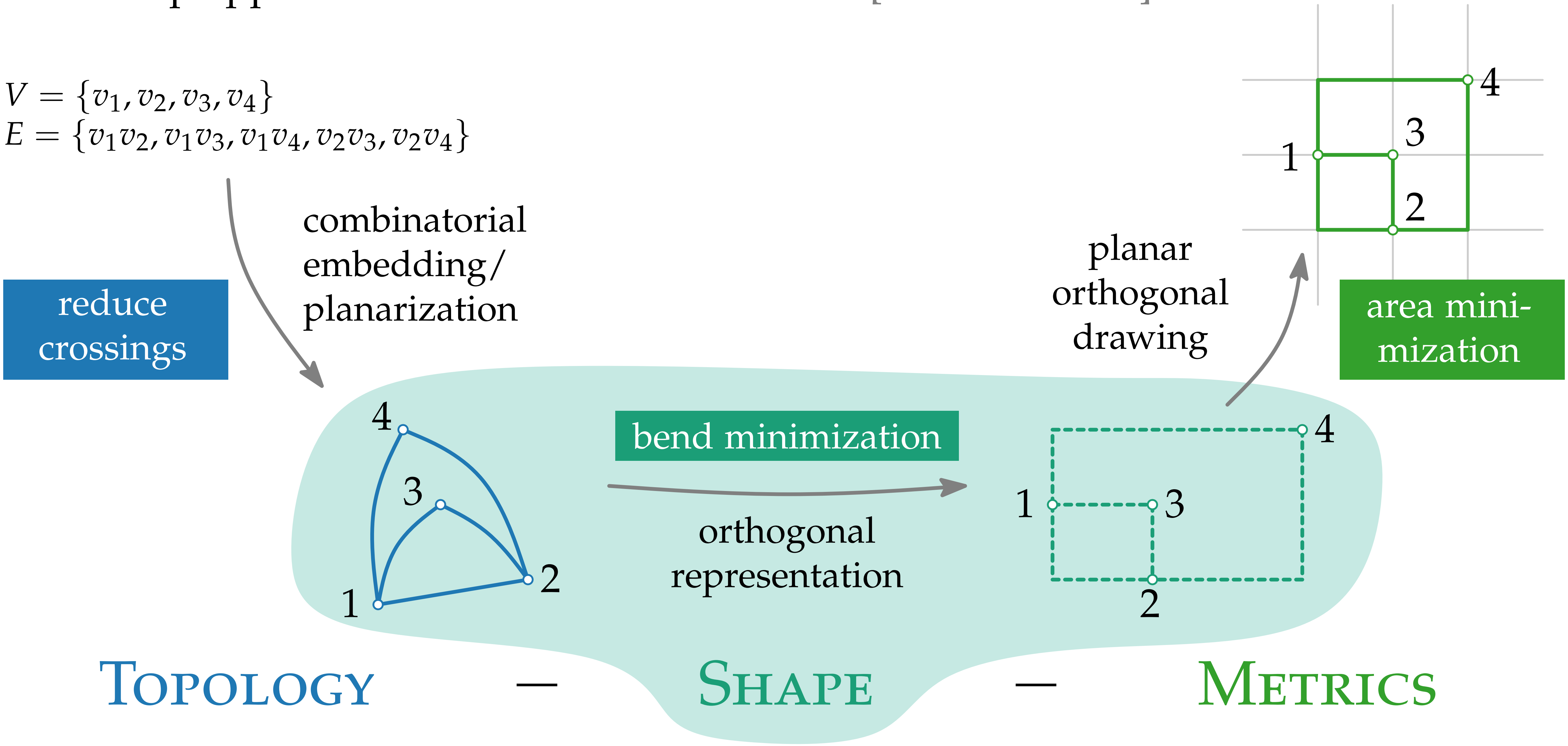
Part IV:
Bend Minimization

Philipp Kindermann

Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

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



Bend Minimization with Given Embedding

Geometric bend minimization.

Given:

Find:

Bend Minimization with Given Embedding

Geometric bend minimization.

Given:
- Plane graph $G = (V, E)$ with maximum degree 4

Find:

Bend Minimization with Given Embedding

Geometric bend minimization.

Given:
- Plane graph $G = (V, E)$ with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find:

Bend Minimization with Given Embedding

Geometric bend minimization.

Given:
- Plane graph $G = (V, E)$ 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 bend minimization.

Given:
- Plane graph $G = (V, E)$ 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 variation.

Combinatorial bend minimization.

Given:

Find:

Bend Minimization with Given Embedding

Geometric bend minimization.

Given:
- Plane graph $G = (V, E)$ 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 variation.

Combinatorial bend minimization.

Given:
- Plane graph $G = (V, E)$ with maximum degree 4
- Combinatorial embedding F and outer face f_0

Find:

Bend Minimization with Given Embedding

Geometric bend minimization.

Given:
- Plane graph $G = (V, E)$ 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 variation.

Combinatorial bend minimization.

Given:
- Plane graph $G = (V, E)$ 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.

Combinatorial Bend Minimization

Combinatorial bend minimization.

Given:
- Plane graph $G = (V, E)$ 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

Combinatorial Bend Minimization

Combinatorial bend minimization.

Given:
- Plane graph $G = (V, E)$ 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

Idea.

Formulate as a network flow problem:

Combinatorial Bend Minimization

Combinatorial bend minimization.

Given:
- Plane graph $G = (V, E)$ 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

Idea.

Formulate as a network flow problem:

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

Combinatorial Bend Minimization

Combinatorial bend minimization.

Given:
- Plane graph $G = (V, E)$ 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

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 Bend Minimization

Combinatorial bend minimization.

Given:
- Plane graph $G = (V, E)$ 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

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 $\xrightarrow{\measuredangle}$ neighbouring faces (# bends toward the neighbour)

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, 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.

Flow Network for Bend Minimization

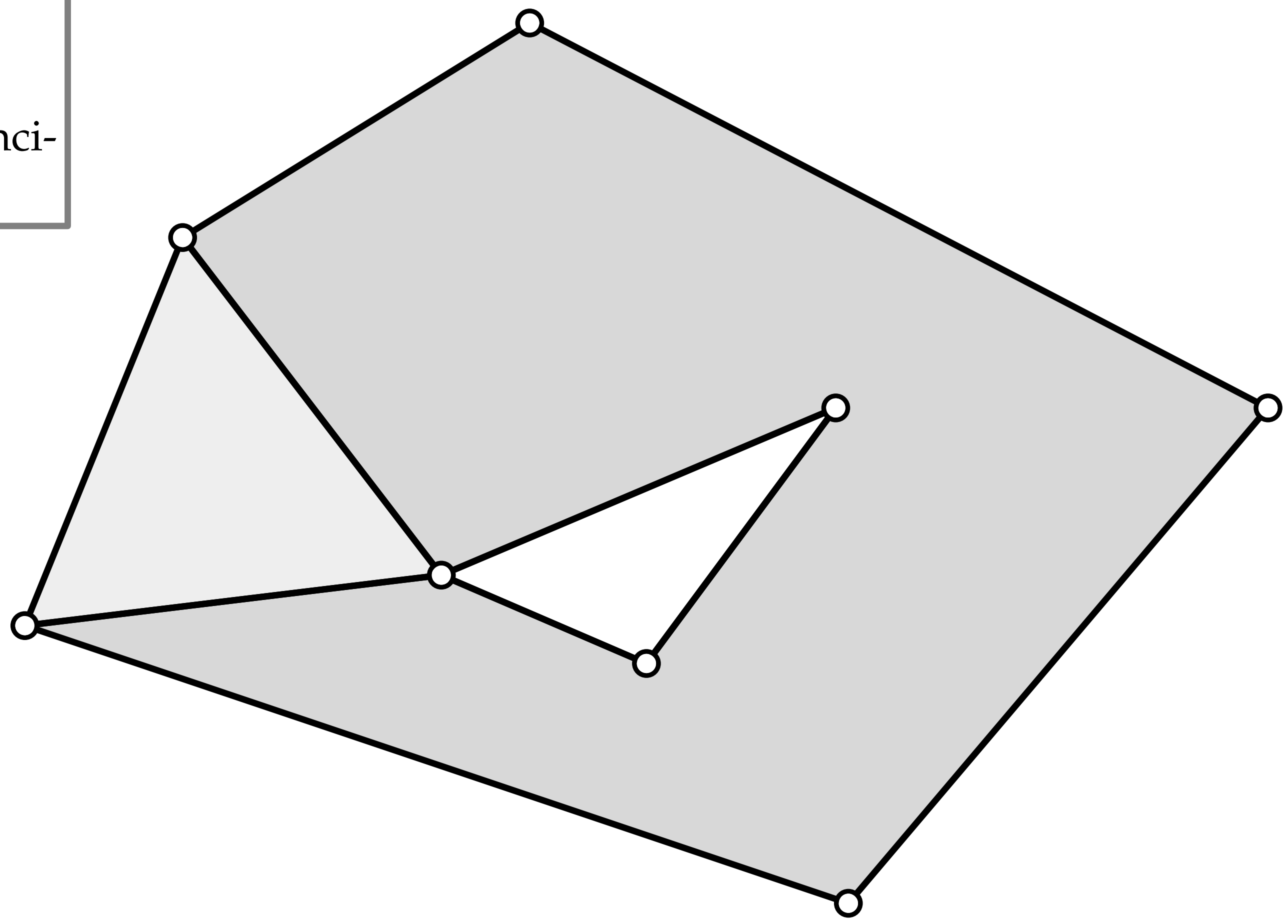
(H1) $H(G)$ corresponds to F, f_0.

(H2) For each **edge** $\{u, v\}$ shared by faces f and g, 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.



Flow Network for Bend Minimization

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

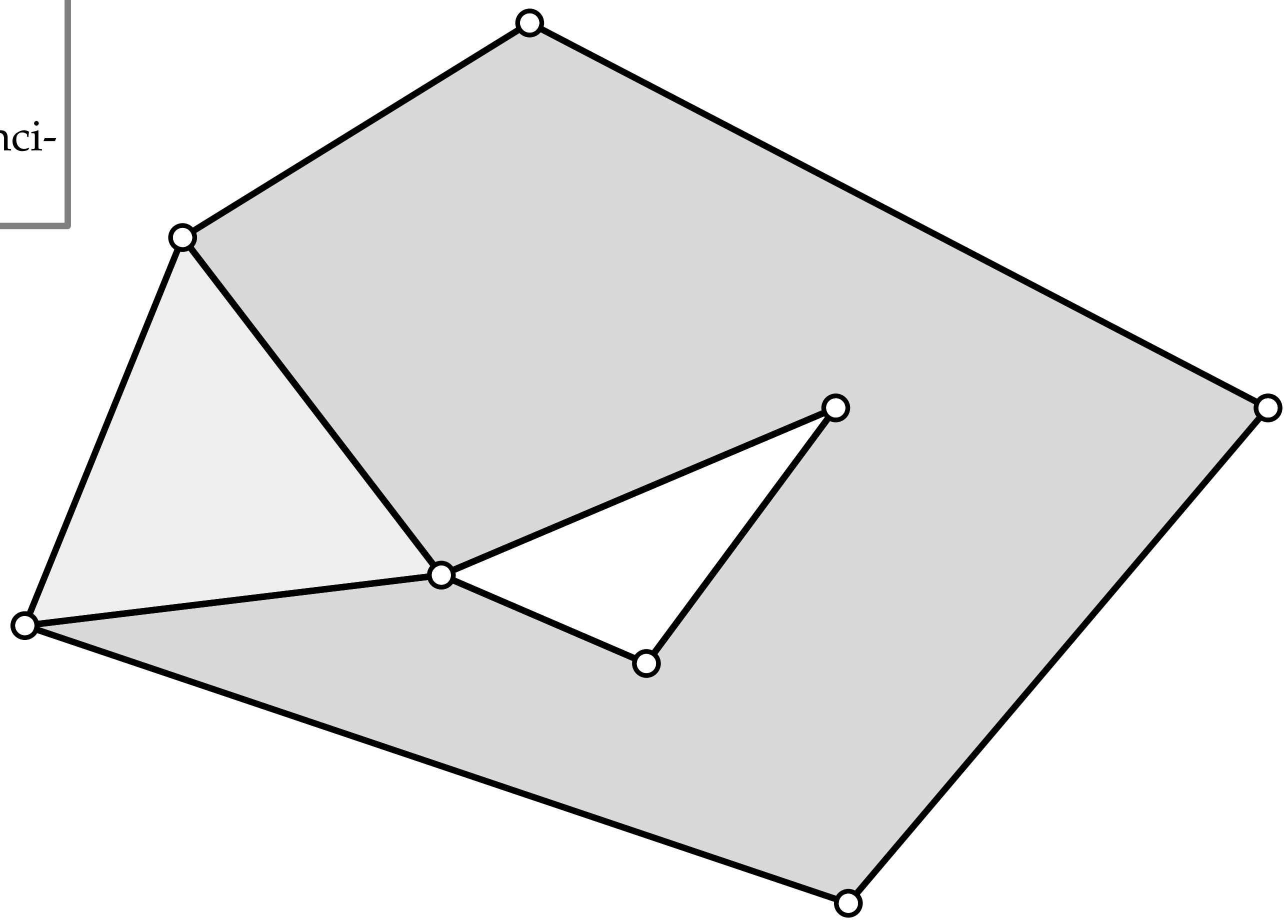
(H2) For each **edge** $\{u, v\}$ shared by faces f and g, sequence δ_1 is reversed and inverted δ_2.

(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.

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



Flow Network for Bend Minimization

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

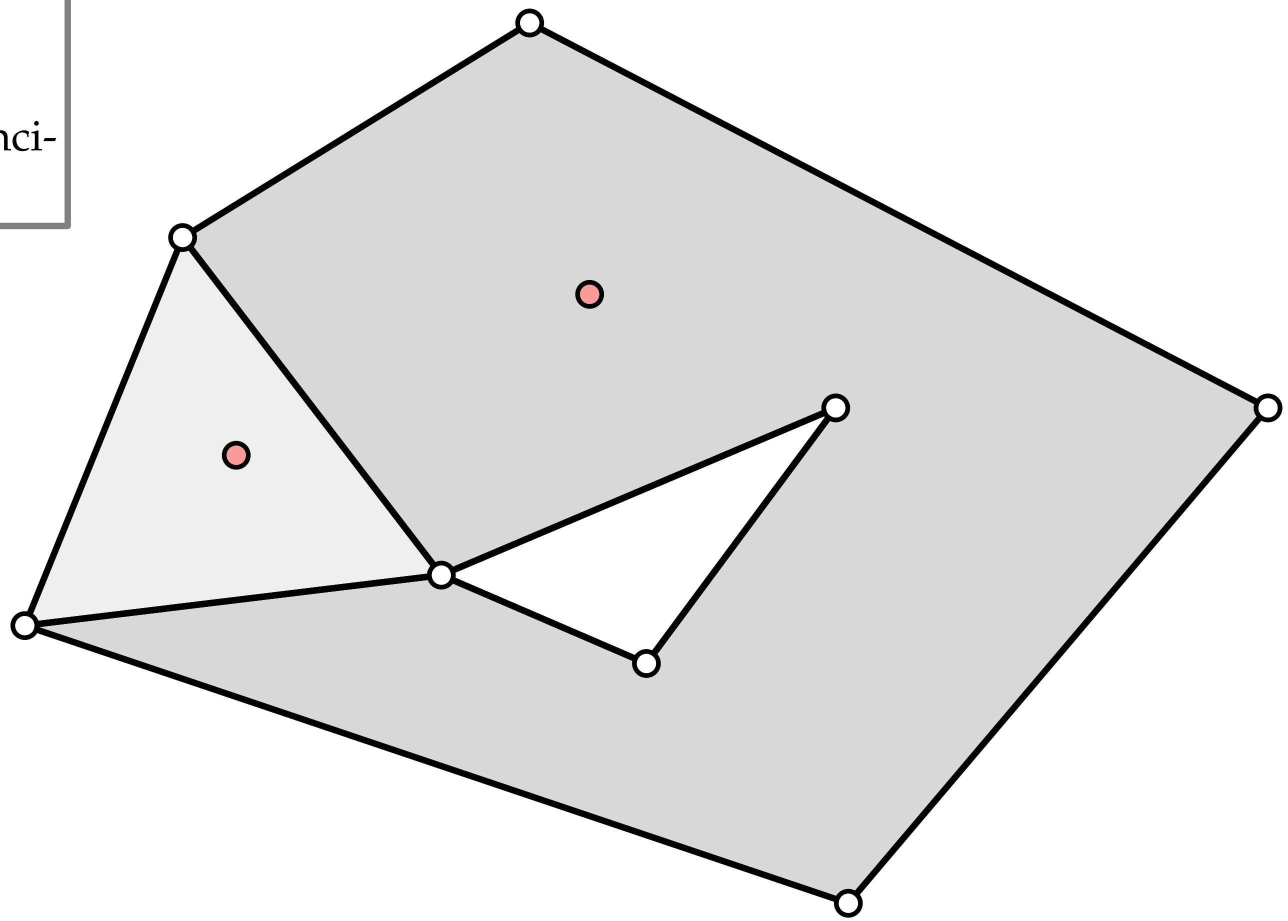
(H2) For each **edge** $\{u, v\}$ shared by faces f and g, sequence δ_1 is reversed and inverted δ_2.

(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.

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



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, sequence δ_1 is reversed and inverted δ_2.

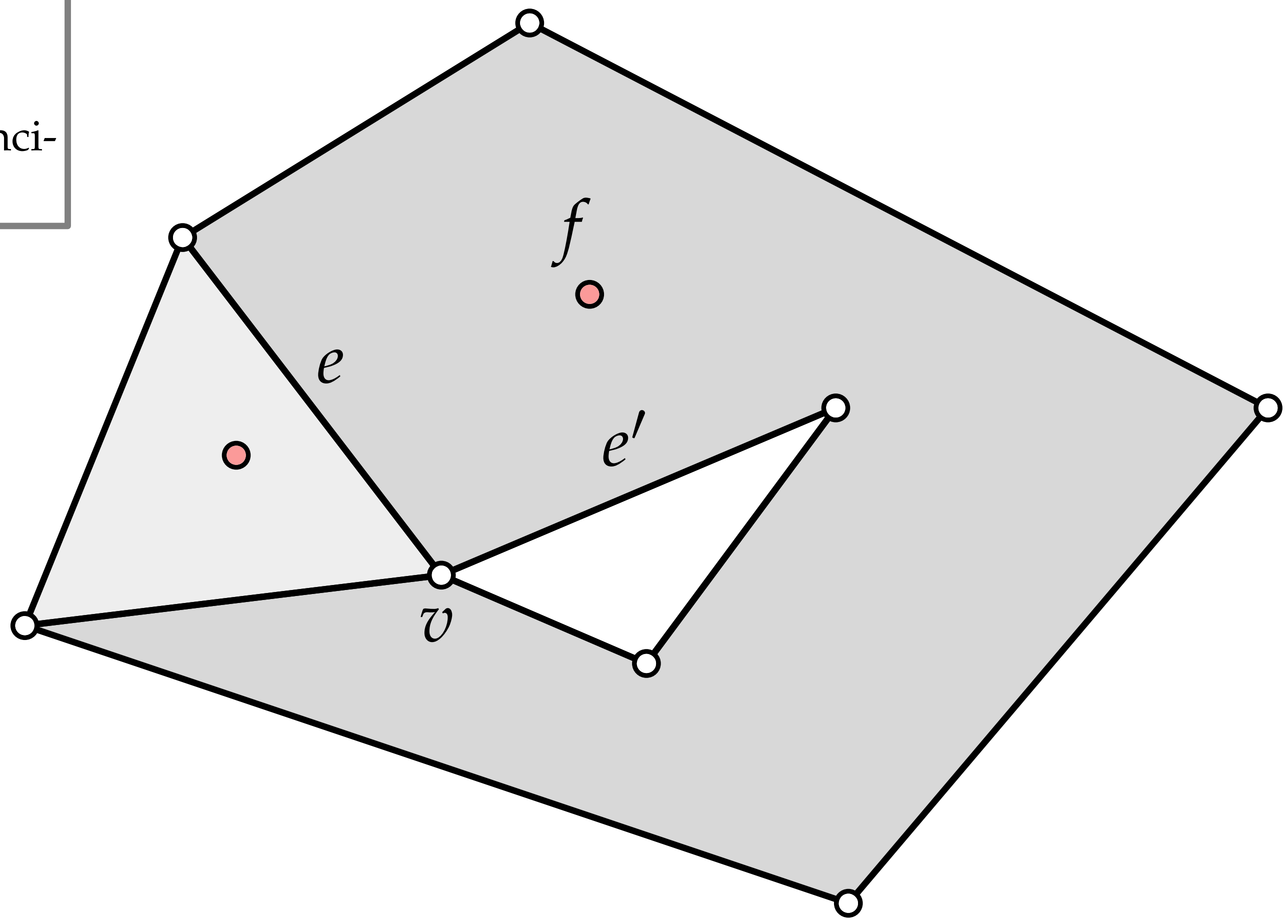
(H3) For each **face** f it holds that:

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

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

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

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



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, sequence δ_1 is reversed and inverted δ_2.

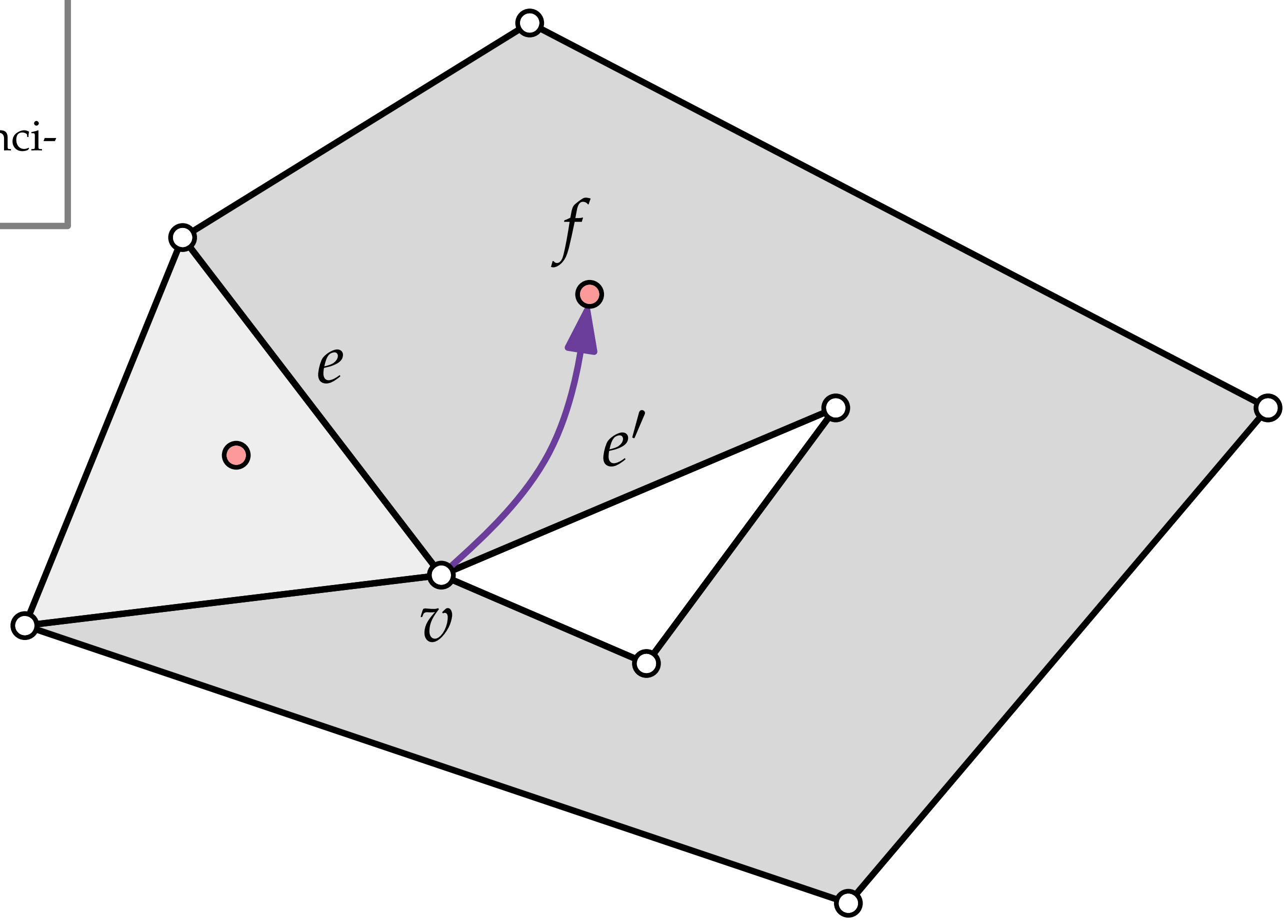
(H3) For each **face** f it holds that:

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

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

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

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



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, sequence δ_1 is reversed and inverted δ_2.

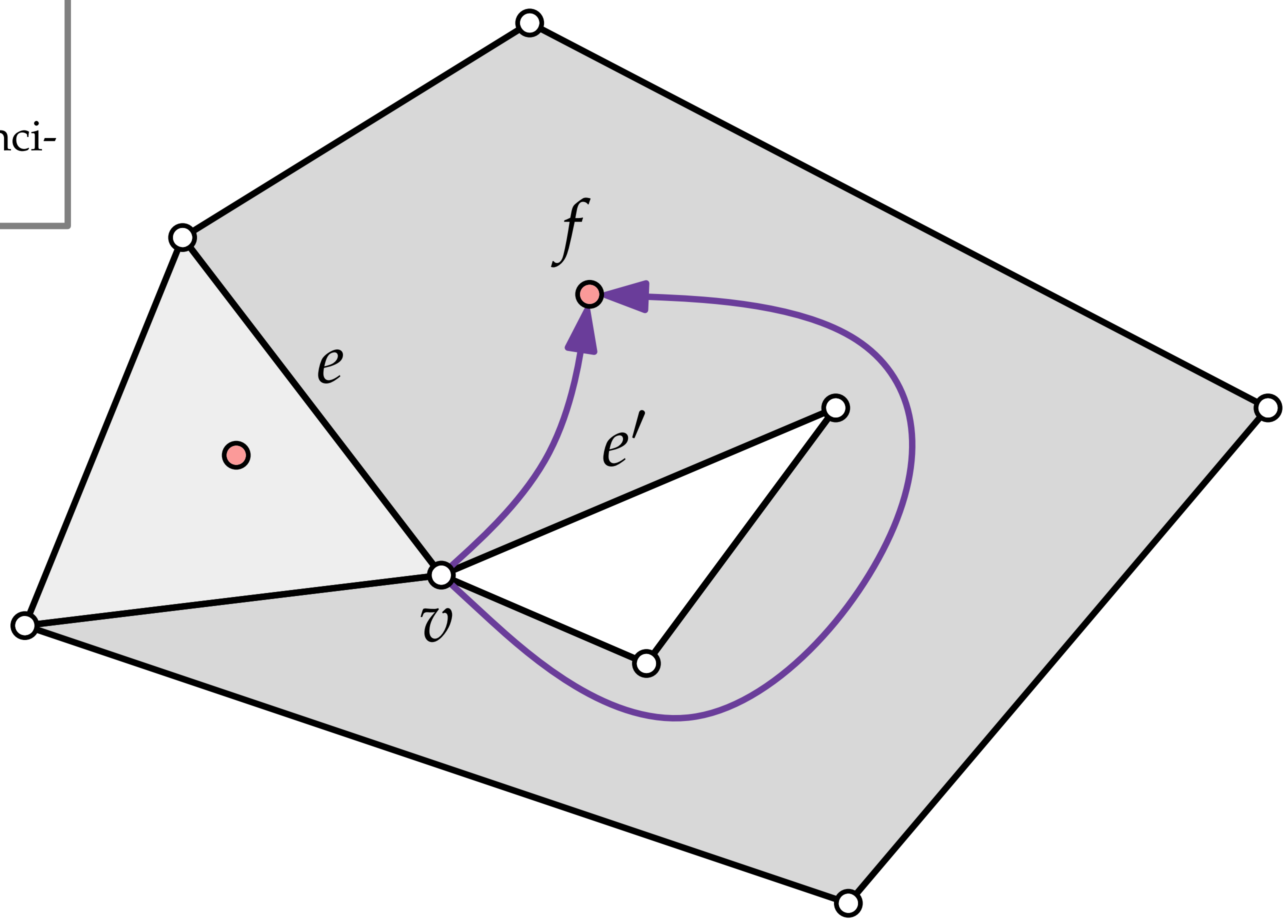
(H3) For each **face** f it holds that:

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

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

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

- $E = \{(v, f)_{ee'} \in V \times F \mid v$ 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, sequence δ_1 is reversed and inverted δ_2.

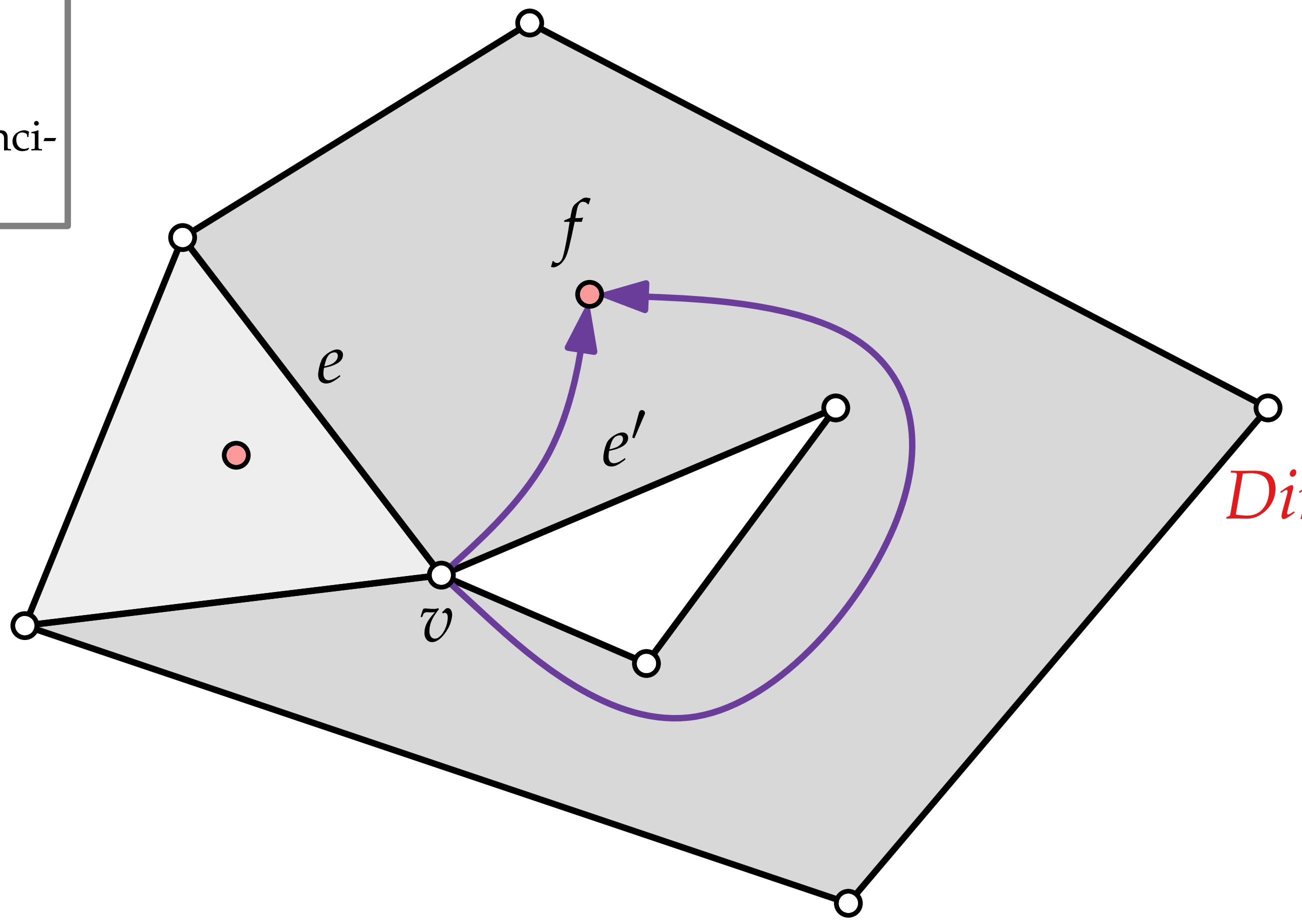
(H3) For each **face** f it holds that:

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

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

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

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



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, sequence δ_1 is reversed and inverted δ_2.

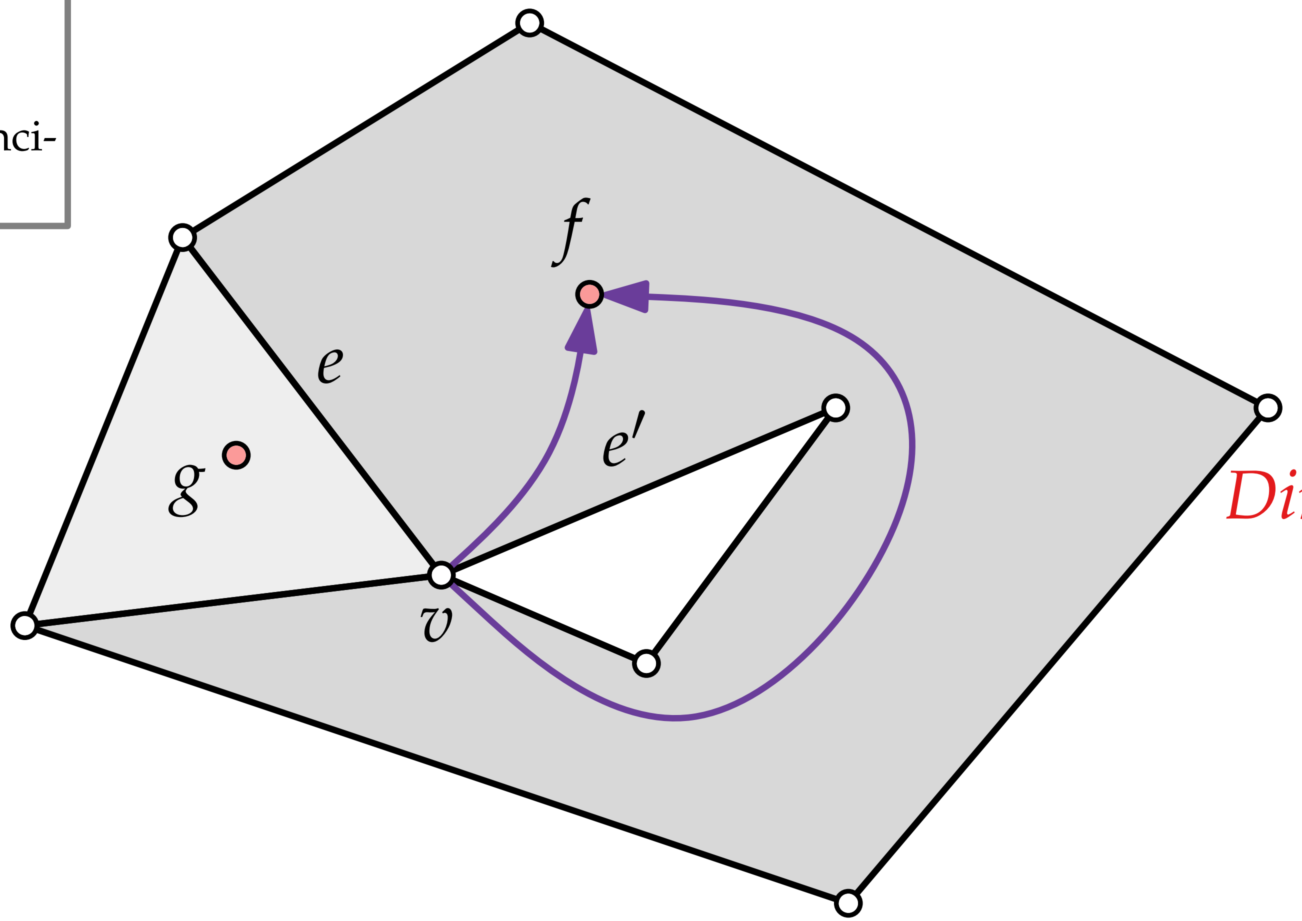
(H3) For each **face** f it holds that:

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

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

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

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



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, sequence δ_1 is reversed and inverted δ_2.

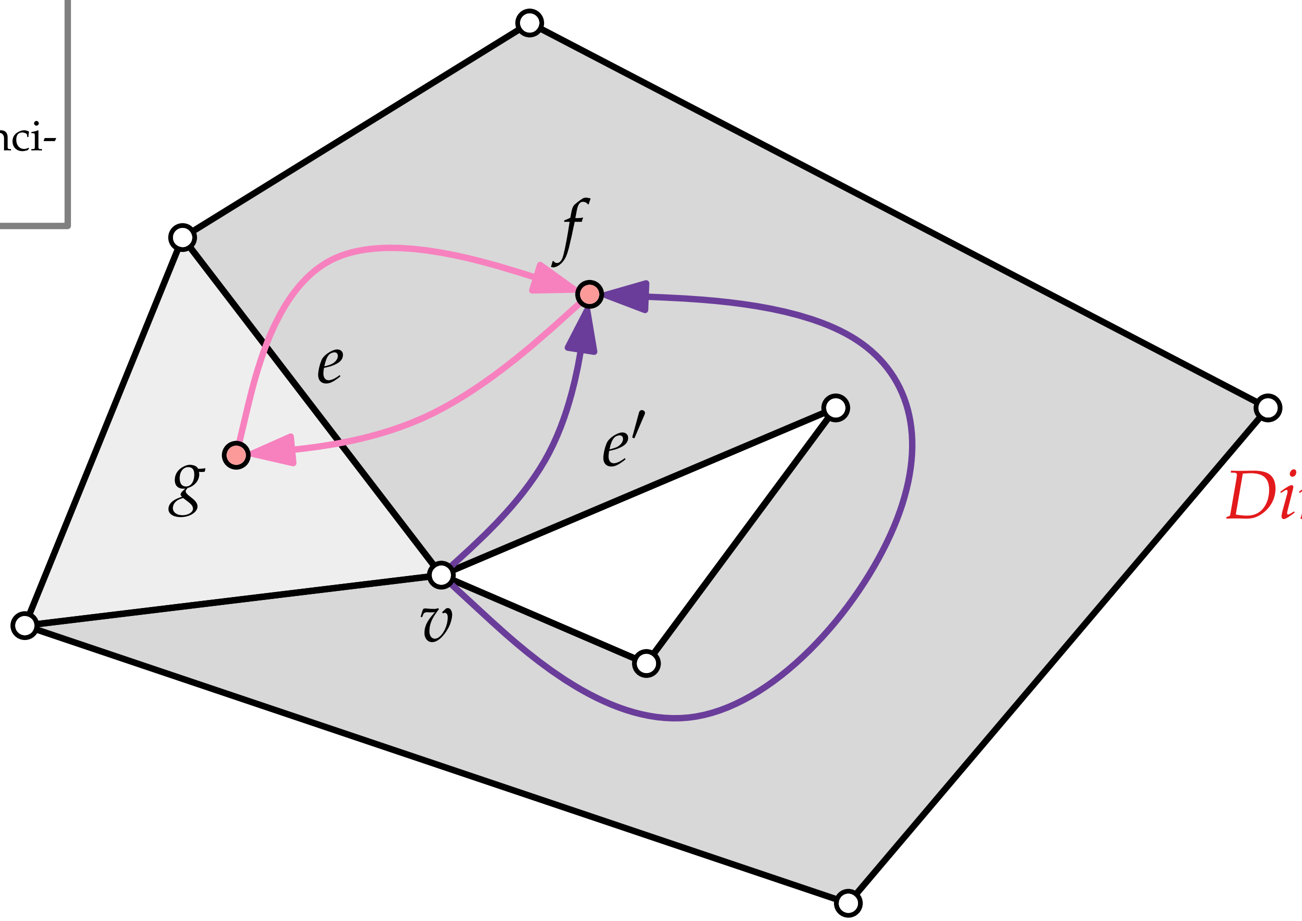
(H3) For each **face** f it holds that:

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

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

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

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



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, sequence δ_1 is reversed and inverted δ_2.

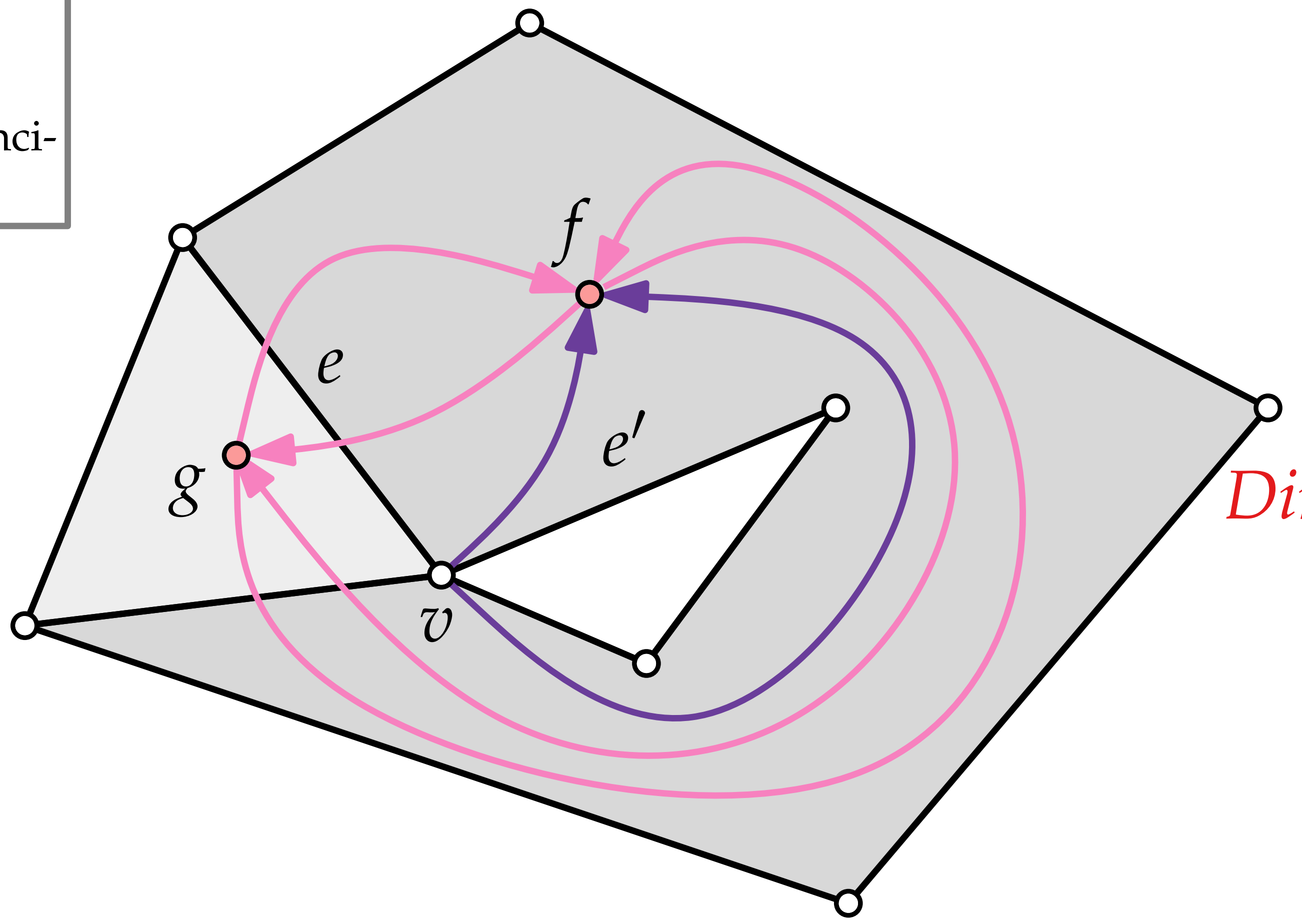
(H3) For each **face** f it holds that:

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

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

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

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



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, 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.

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

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

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, sequence δ_1 is reversed and inverted δ_2.

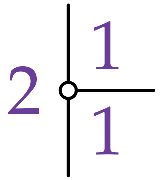
(H3) For each **face** f it holds that:

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

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

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

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



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, sequence δ_1 is reversed and inverted δ_2.

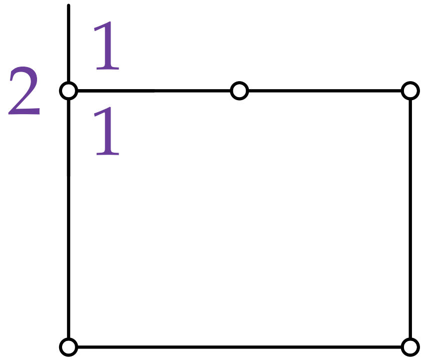
(H3) For each **face** f it holds that:

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

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

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

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



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, sequence δ_1 is reversed and inverted δ_2.

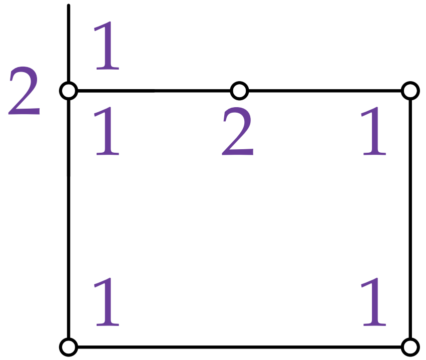
(H3) For each **face** f it holds that:

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

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

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

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



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, sequence δ_1 is reversed and inverted δ_2.

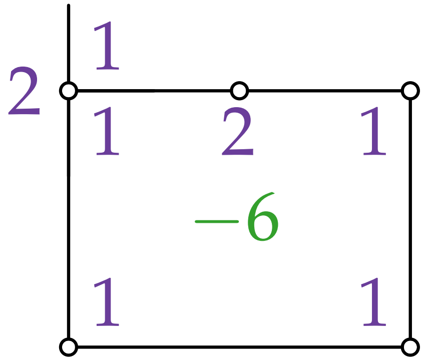
(H3) For each **face** f it holds that:

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

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

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

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



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, sequence δ_1 is reversed and inverted δ_2.

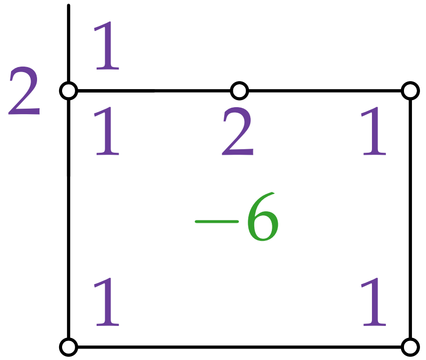
(H3) For each **face** f it holds that:

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

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

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

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



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, sequence δ_1 is reversed and inverted δ_2.

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

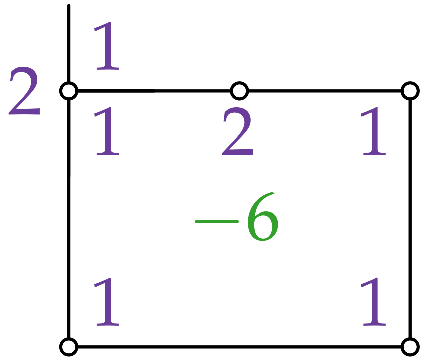
$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

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

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

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

$\Rightarrow \sum_w b(w) \overset{?}{=} 0$



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, sequence δ_1 is reversed and inverted δ_2.

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

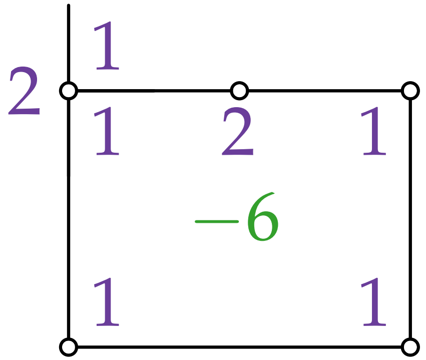
$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

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

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

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

$\Rightarrow \sum_w b(w) = 0$ (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, sequence δ_1 is reversed and inverted δ_2.

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

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

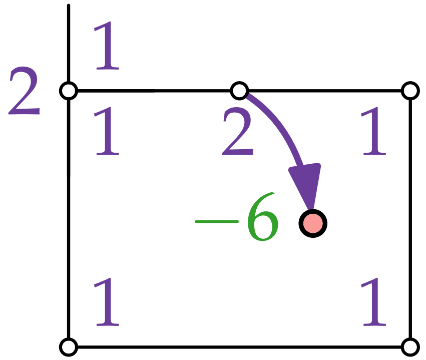
(H4) For each **vertex** v the sum of incident angles is 2π.

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

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

$\Rightarrow \sum_w b(w) = 0$ (Euler)

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



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, sequence δ_1 is reversed and inverted δ_2.

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

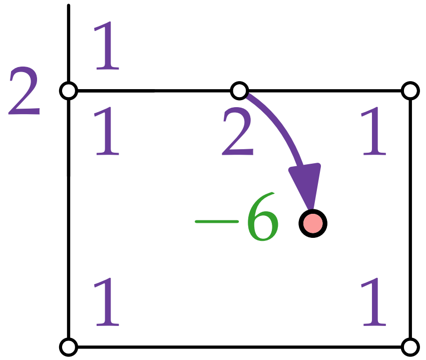
$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

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

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

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

$\Rightarrow \sum_w b(w) = 0$ (Euler)



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

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

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, 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.

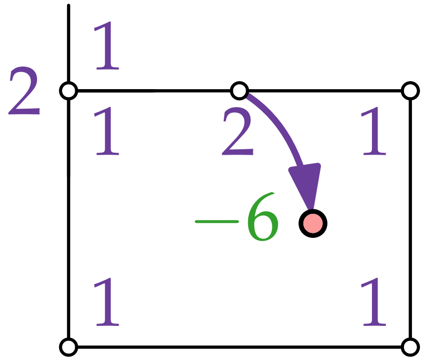
Define flow network $N(G) = ((V \cup F, E); b; \ell; u; \text{cost})$:

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

$\Rightarrow \sum_w b(w) = 0$ (Euler)

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

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



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, 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.

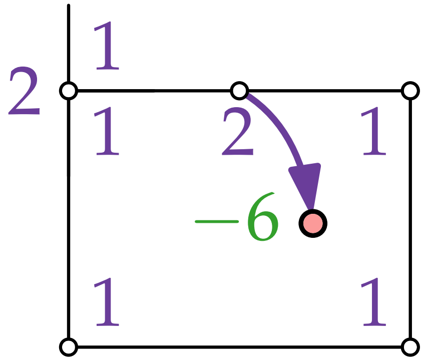
Define flow network $N(G) = ((V \cup F, E); b; \ell; u; \text{cost})$:

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

$\Rightarrow \sum_w b(w) = 0$ (Euler)

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

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



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, 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.

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

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

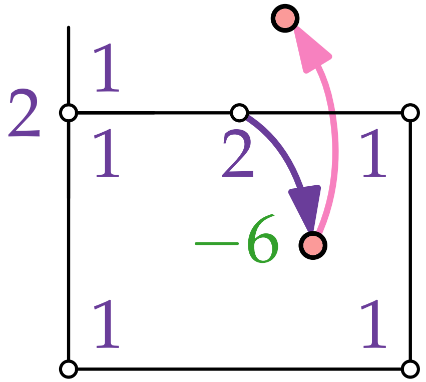
$\Rightarrow \sum_w b(w) = 0$ (Euler)

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

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

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

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



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, sequence δ_1 is reversed and inverted δ_2.

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

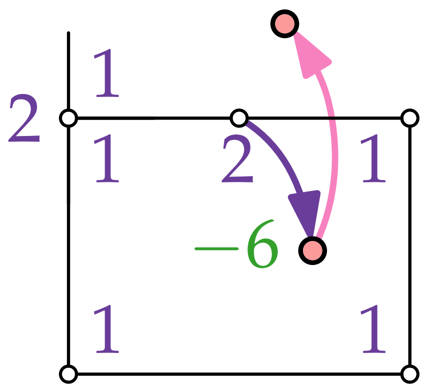
$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

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

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

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

$\Rightarrow \sum_w b(w) = 0$ (Euler)



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

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

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

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

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, sequence δ_1 is reversed and inverted δ_2.

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

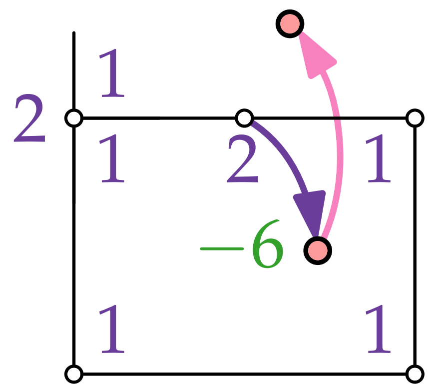
$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

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

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

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

$\Rightarrow \sum_w b(w) = 0$ (Euler)

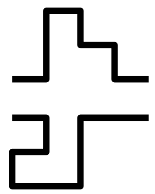


$\forall (v, f) \in E, v \in V, f \in F \qquad \ell(v, f) := 1 \leq X(v, f) \leq 4 =: u(v, f)$

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

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

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



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

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, sequence δ_1 is reversed and inverted δ_2.

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

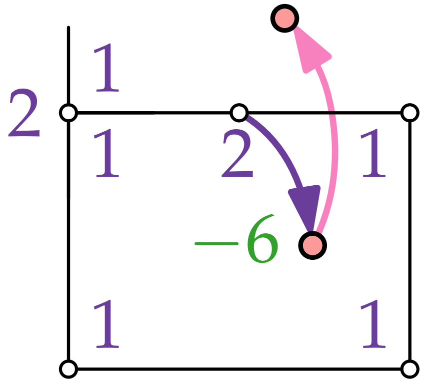
$$\sum_{r \in H(f)} C(r) = \begin{cases} -4 & \text{if } f = f_0 \\ +4 & \text{otherwise.} \end{cases}$$

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

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

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

$\Rightarrow \sum_w b(w) = 0$ (Euler)

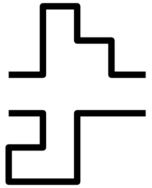


$\forall (v, f) \in E, v \in V, f \in F \quad \ell(v, f) := 1 \leq X(v, f) \leq 4 =: u(v, f)$

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

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

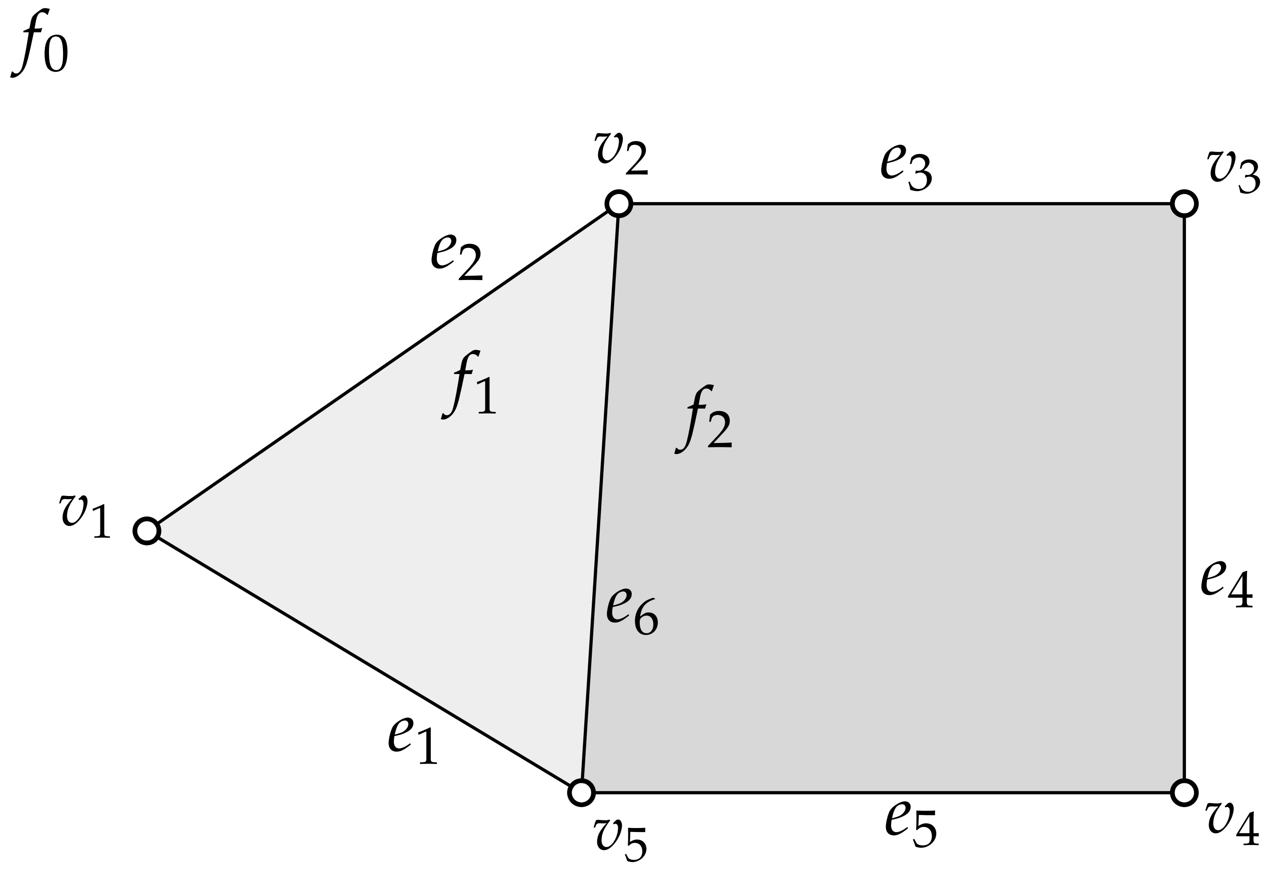
$\text{cost}(f, g) = 1$



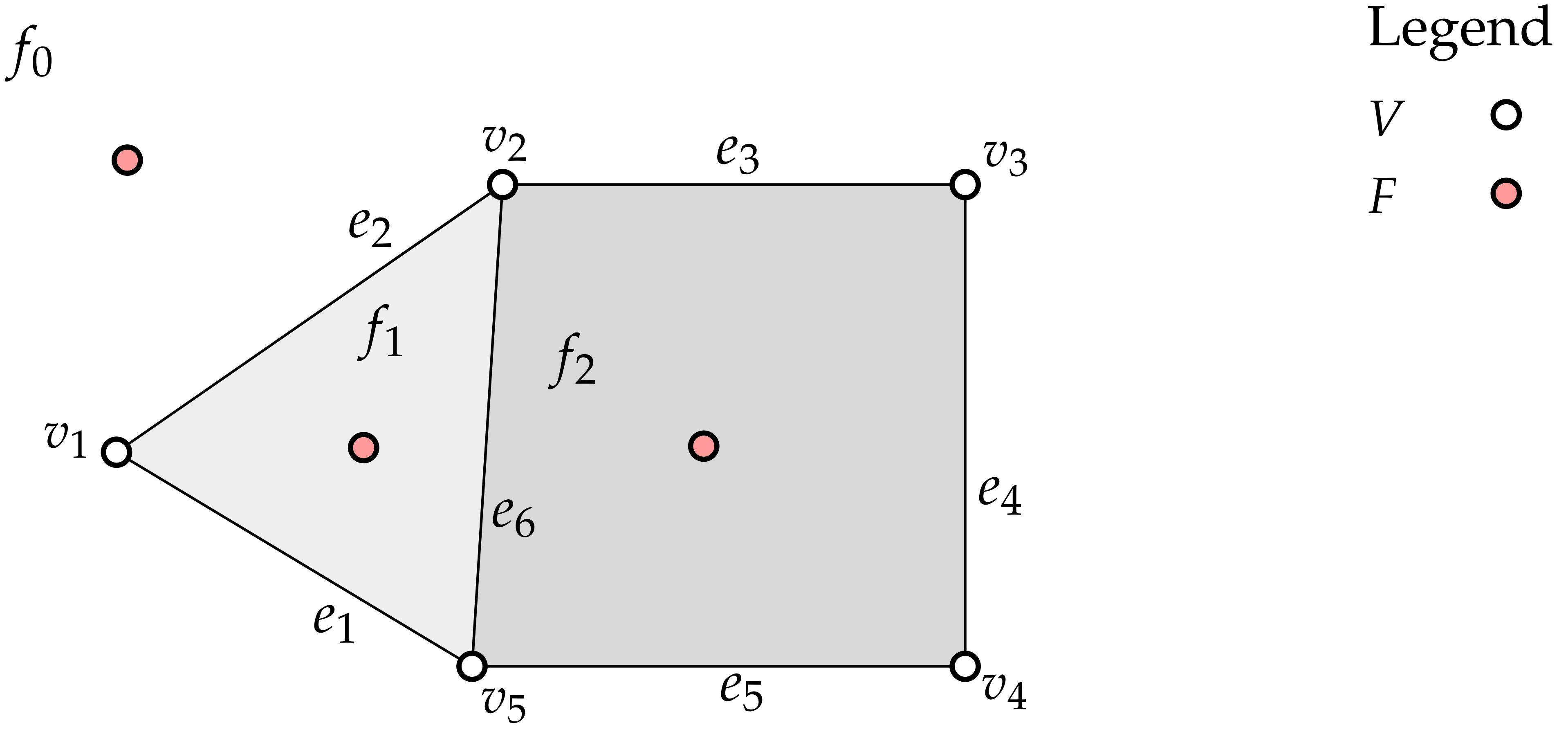
We model only the *number* of bends. Why is it enough?

→ Exercise

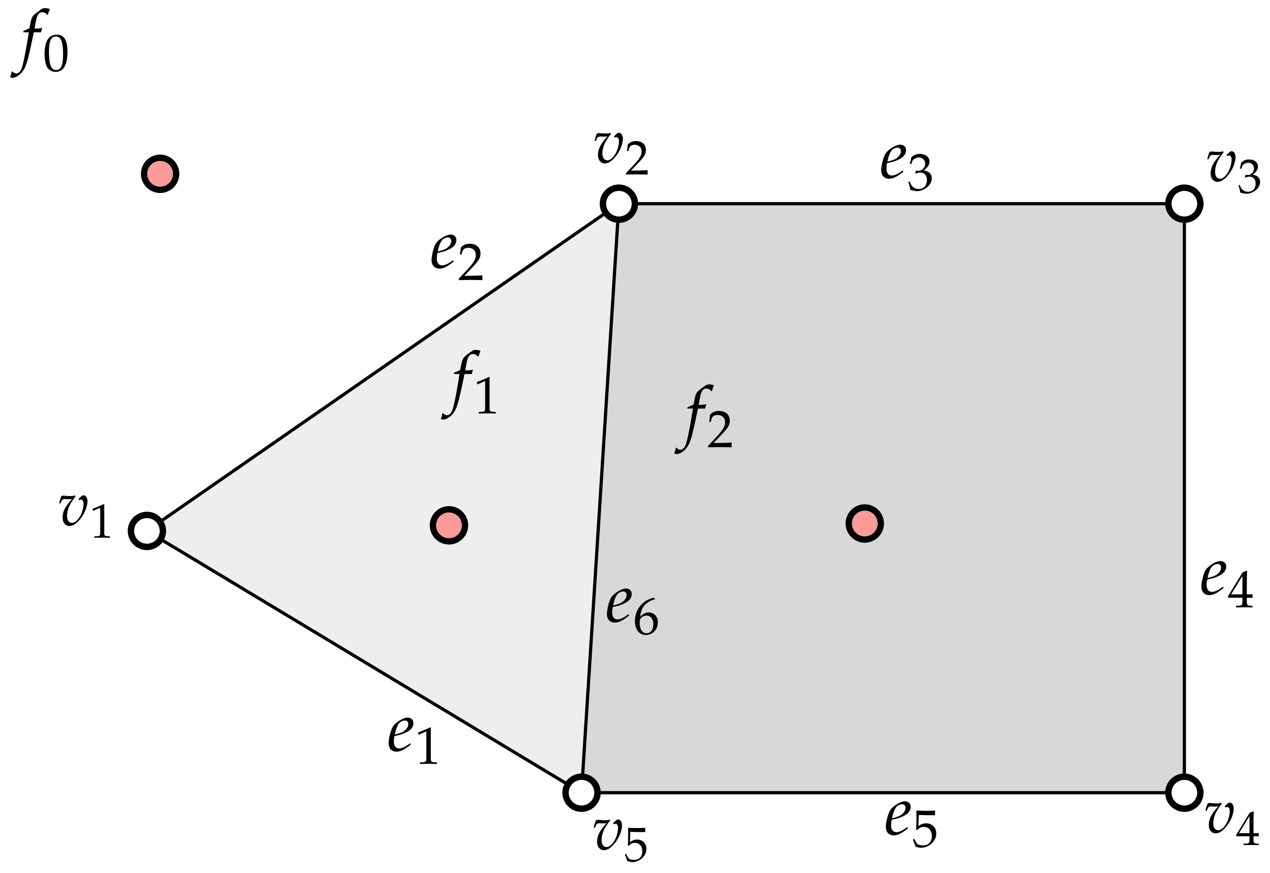
Flow Network Example

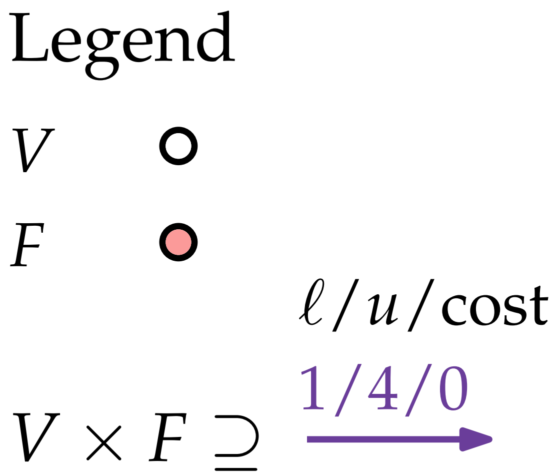


Flow Network Example

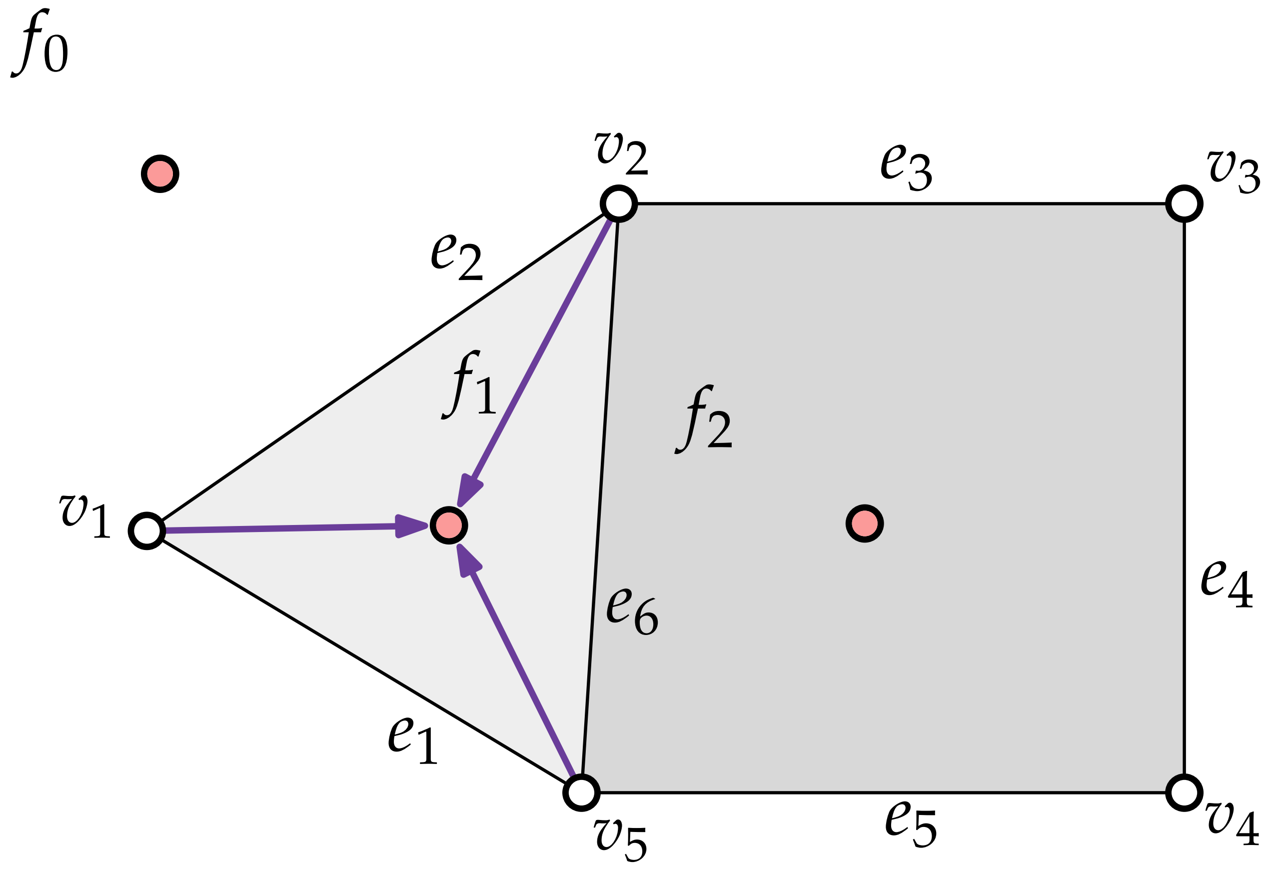


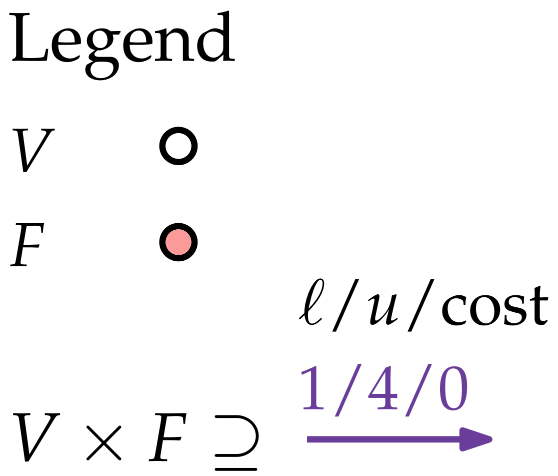
Flow Network Example



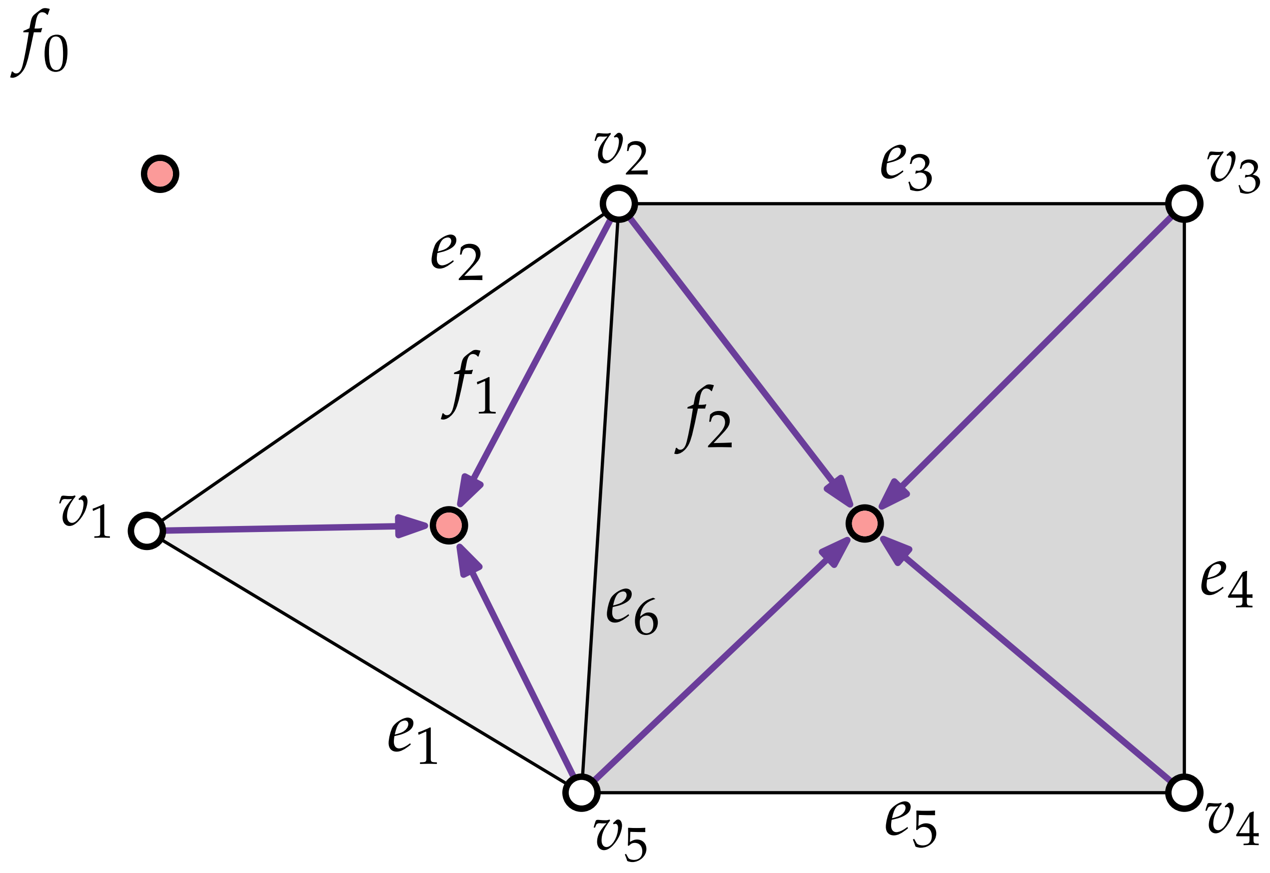


Flow Network Example





Flow Network Example



Legend

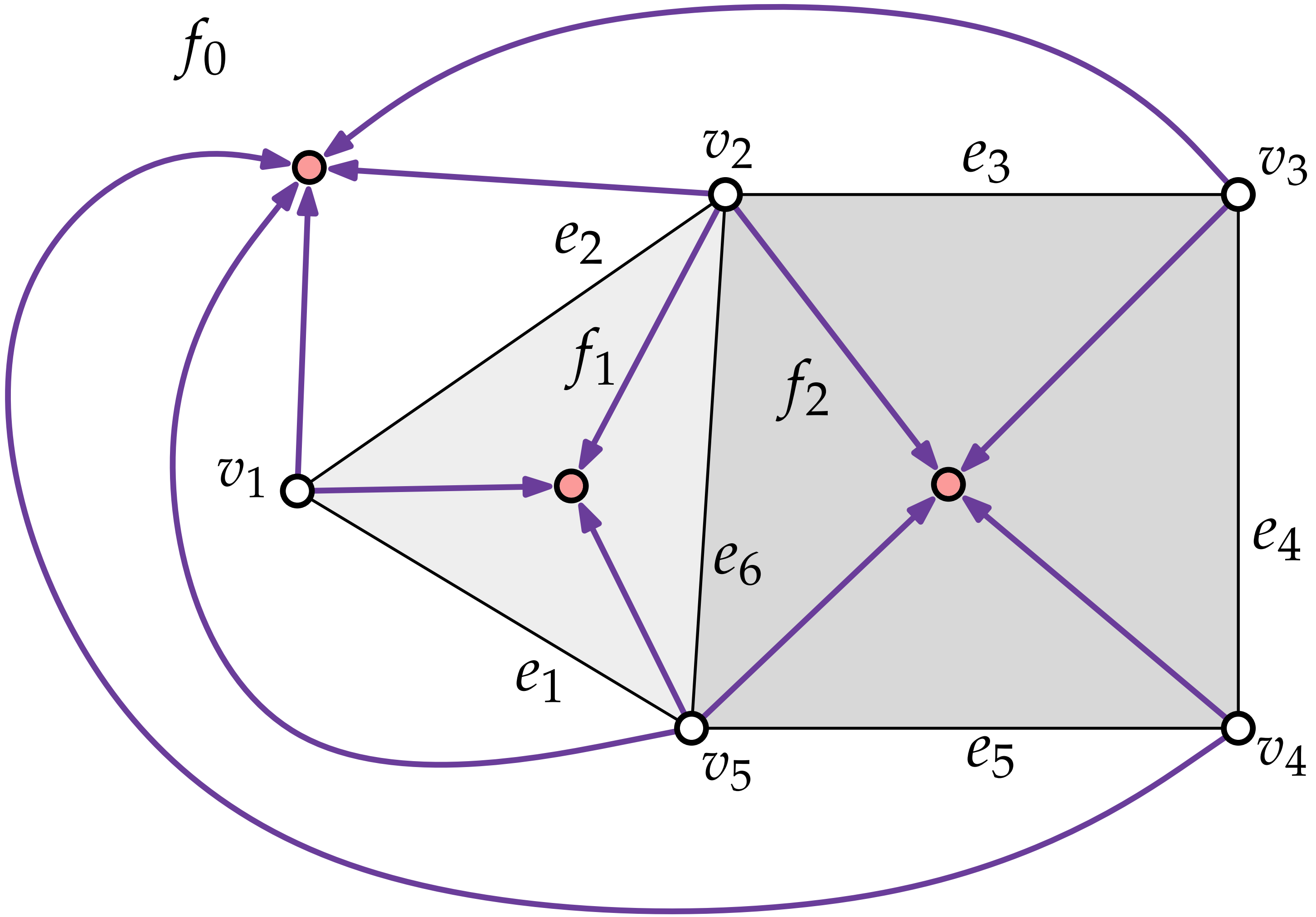
V ○

F ●

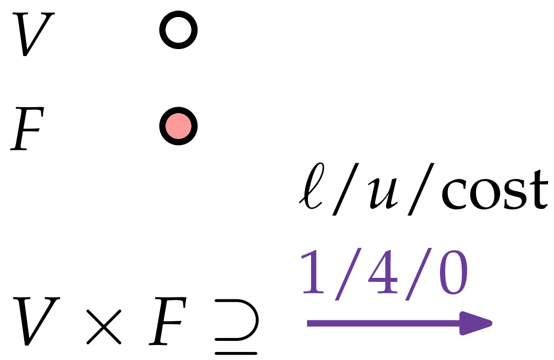
$\ell/u/\text{cost}$

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

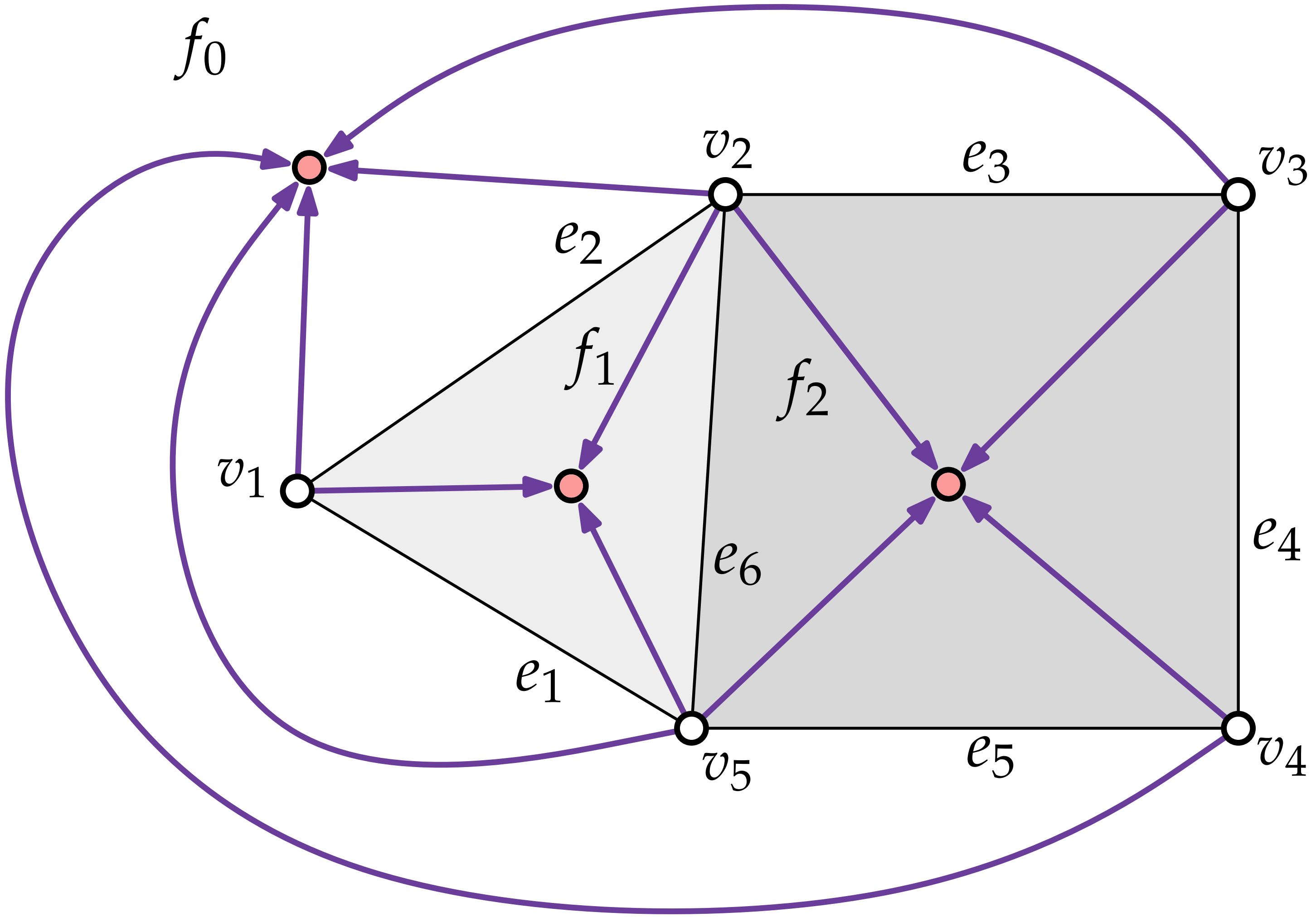
Flow Network Example

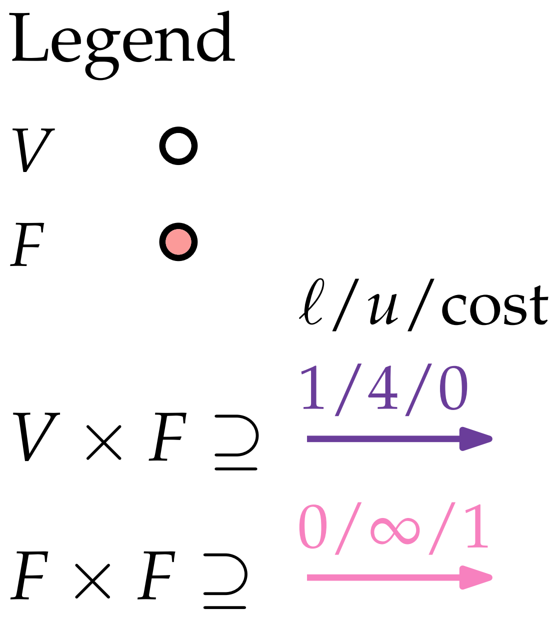


Legend

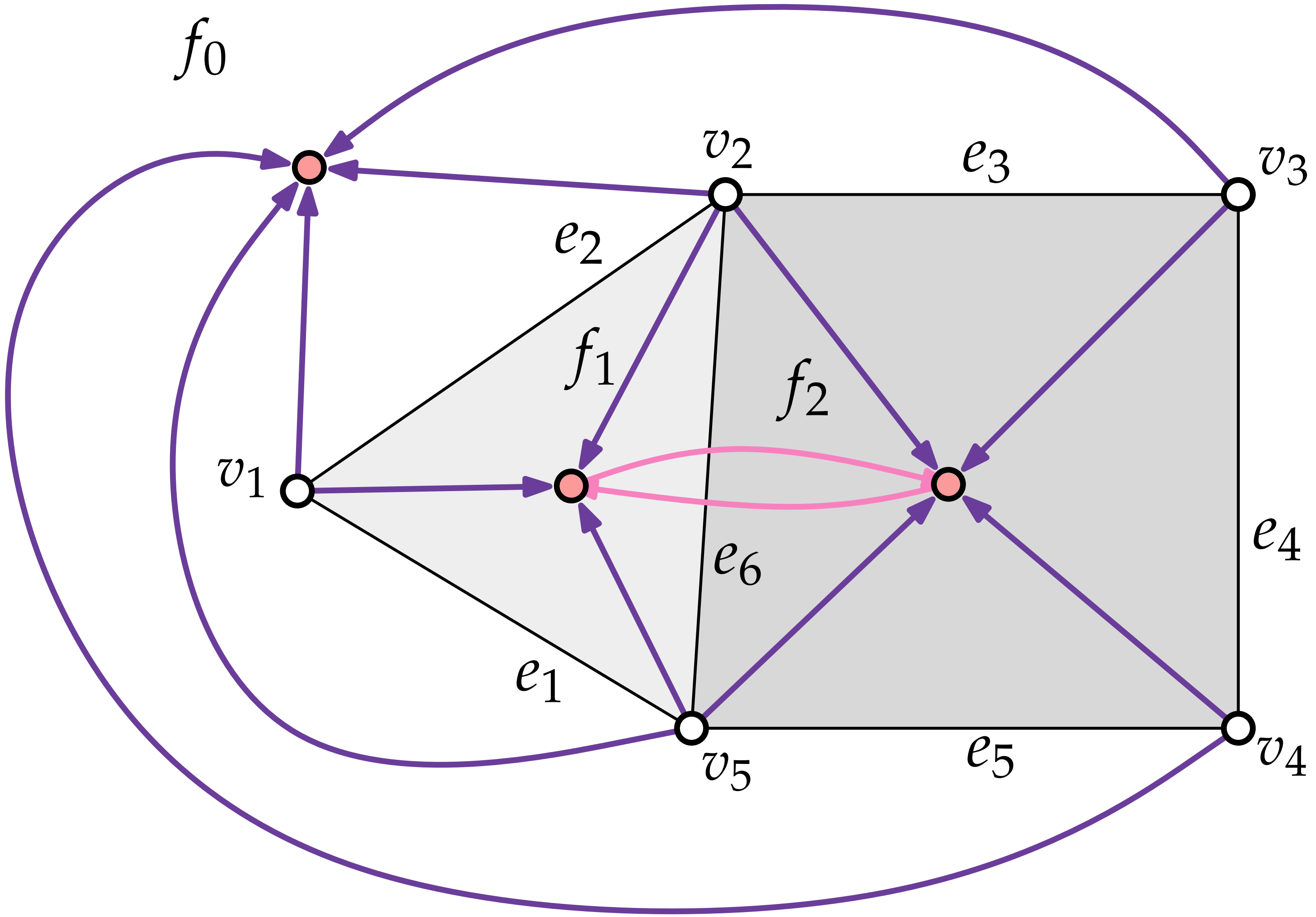


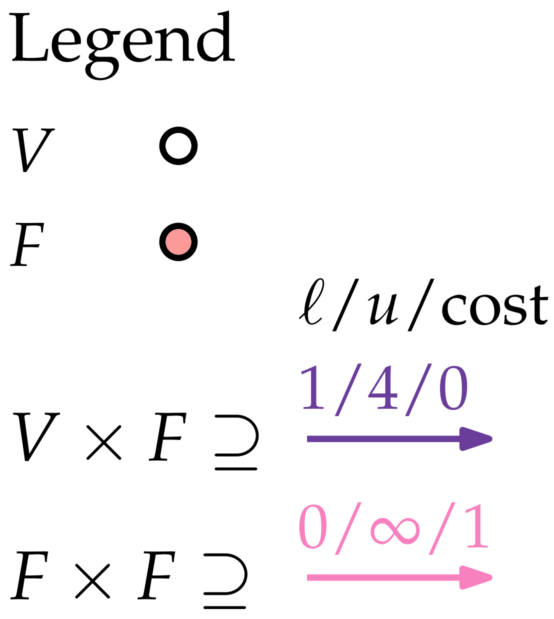
Flow Network Example



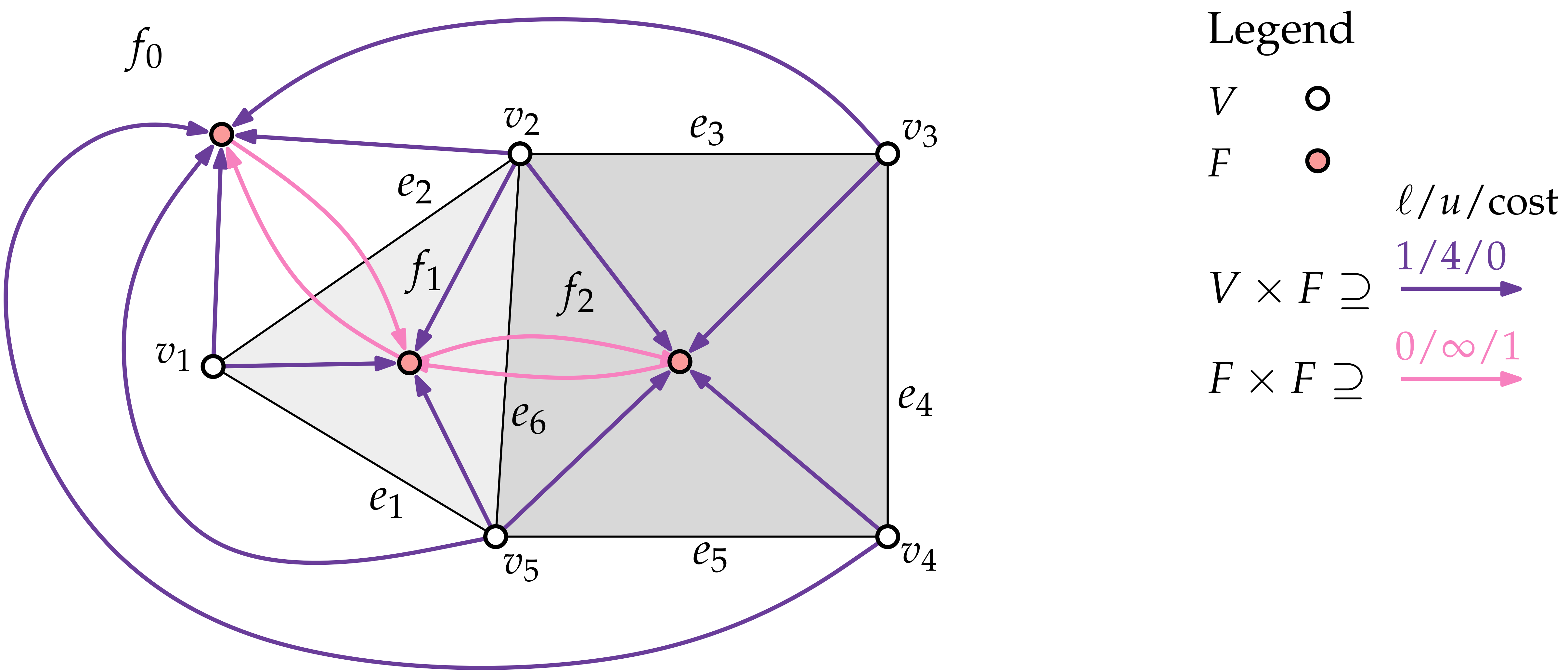


Flow Network Example

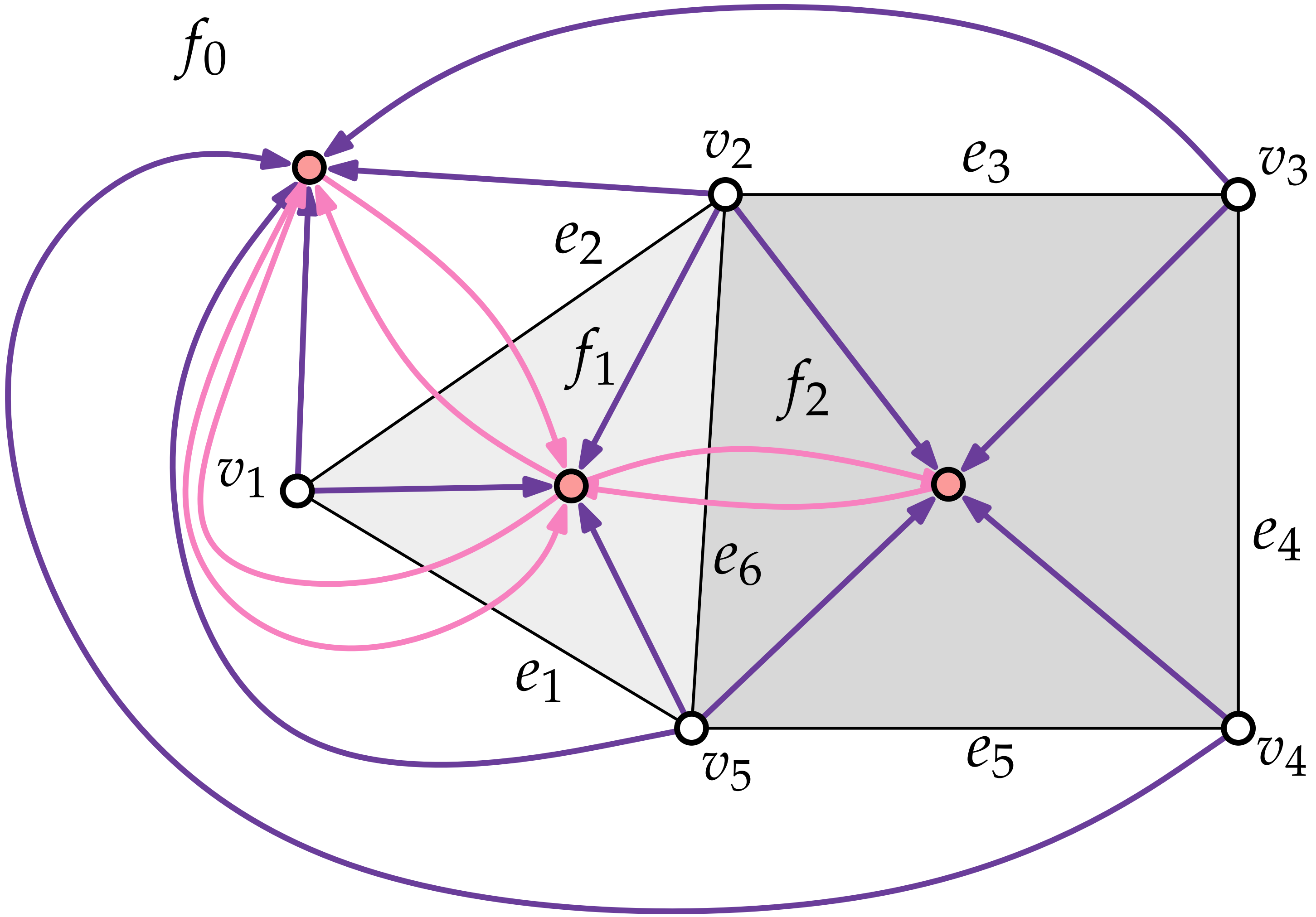




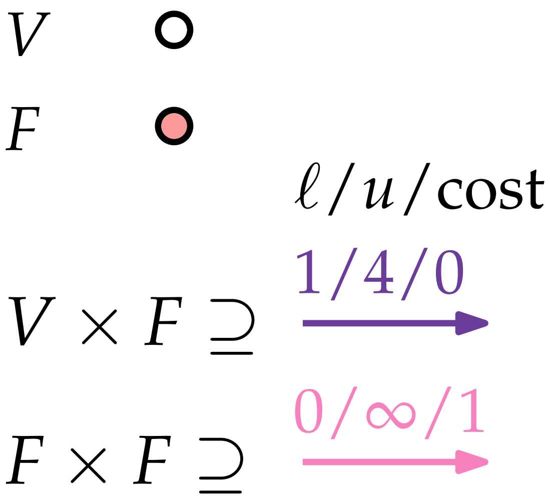
Flow Network Example



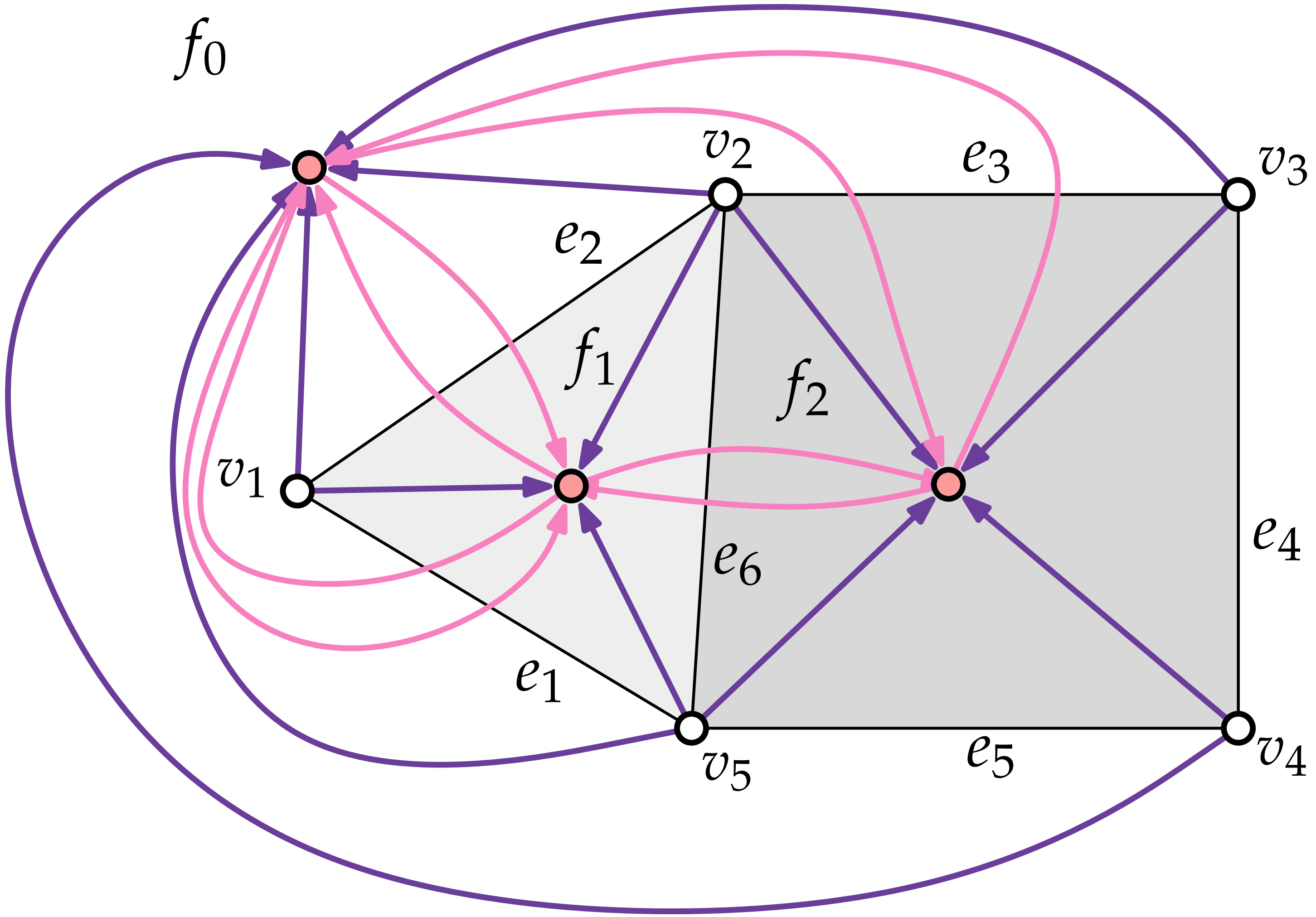
Flow Network Example

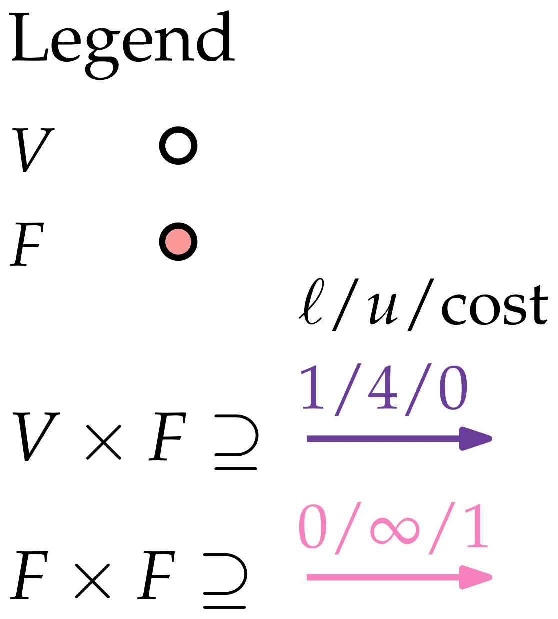


Legend

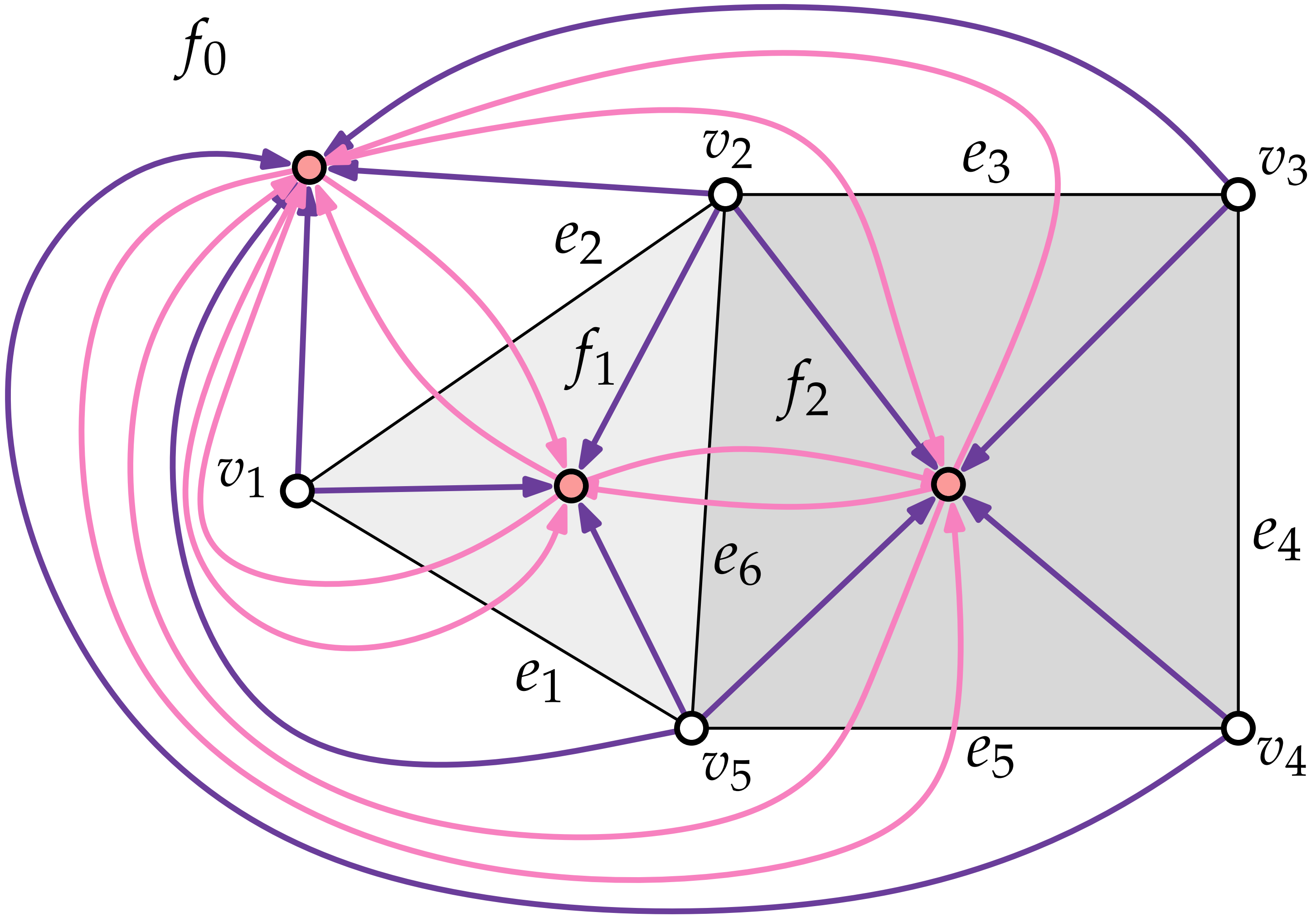


Flow Network Example





Flow Network Example



Legend

V ○

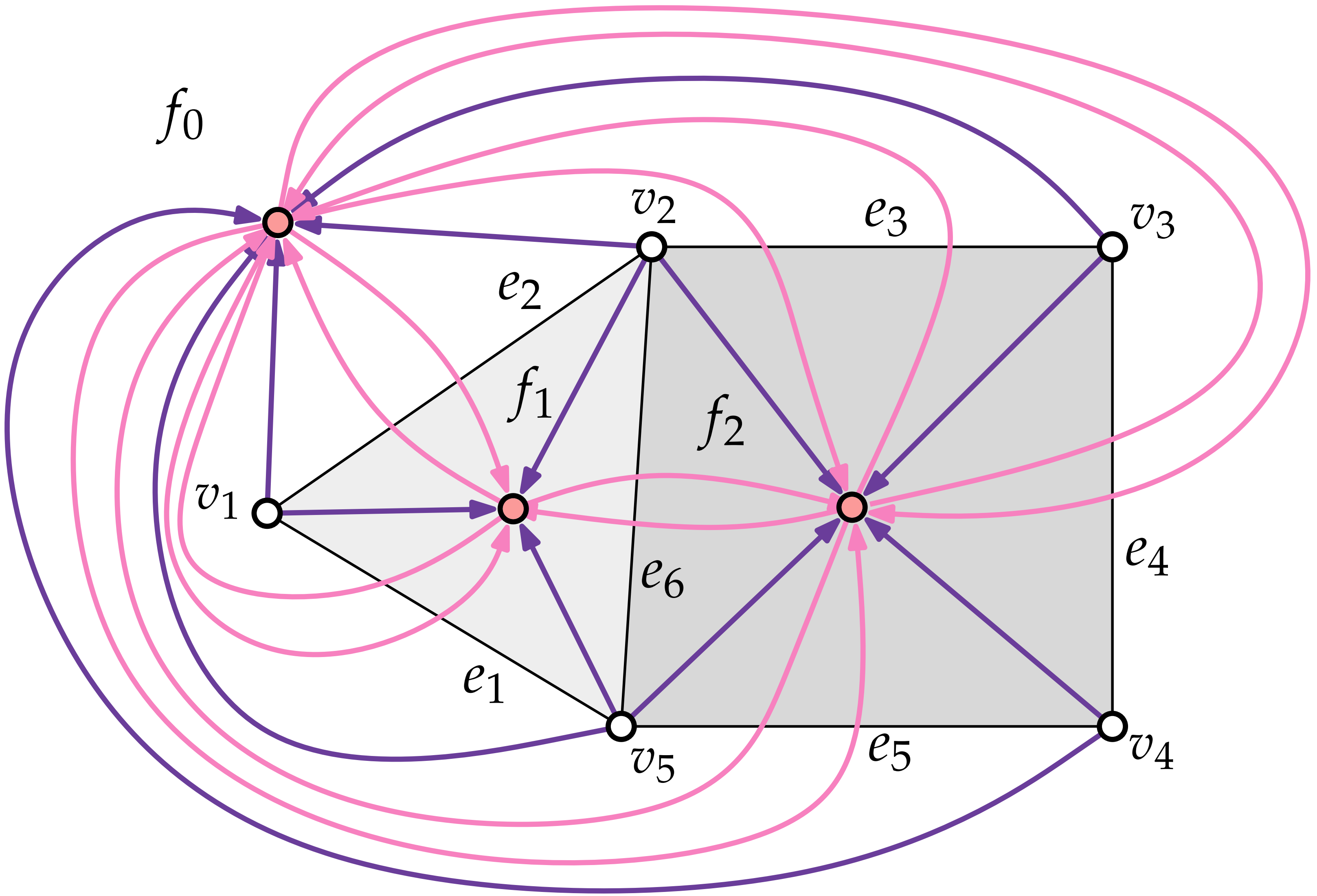
F ●

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

$V \times F \supseteq$ 1/4/0 →

$F \times F \supseteq$ $0/\infty/1$ →

Flow Network Example



Legend

V

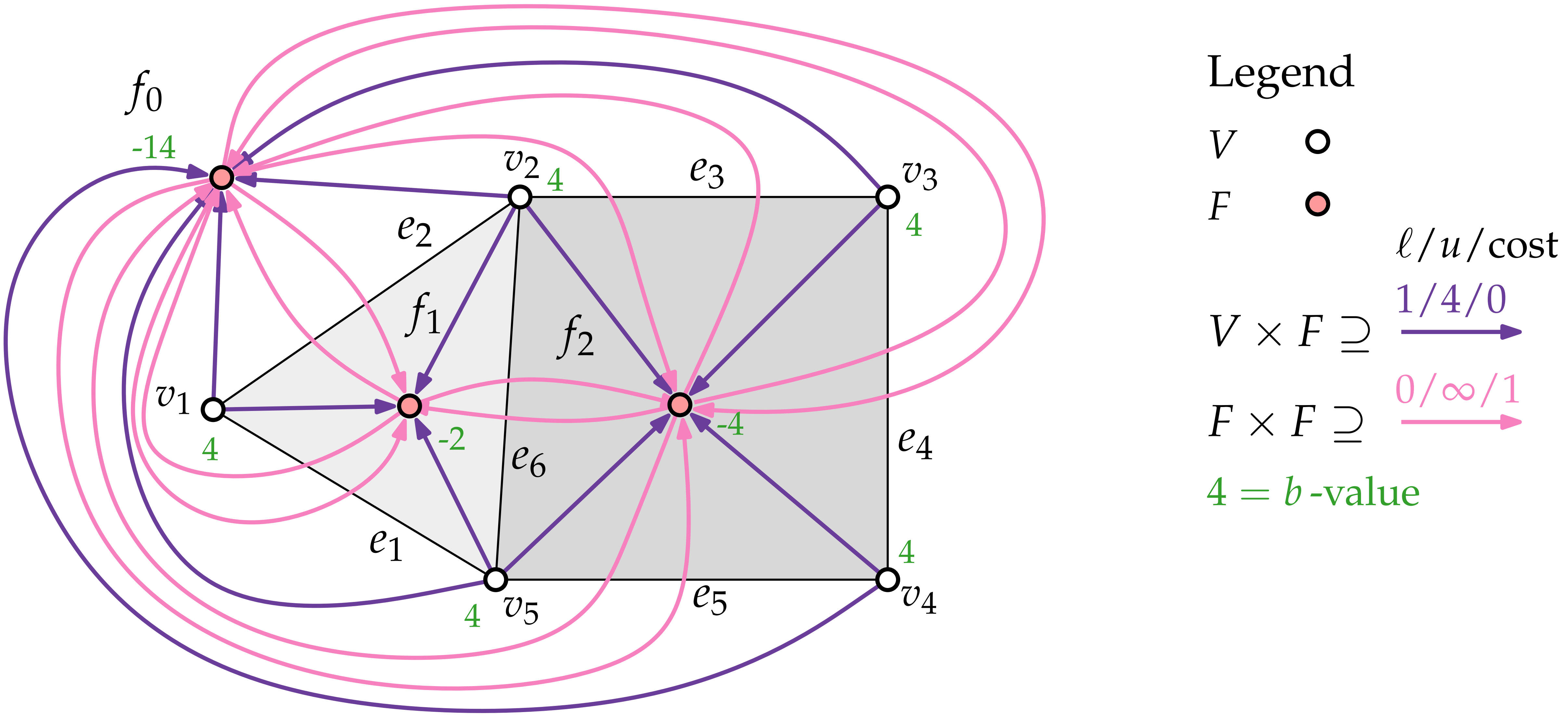
F

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

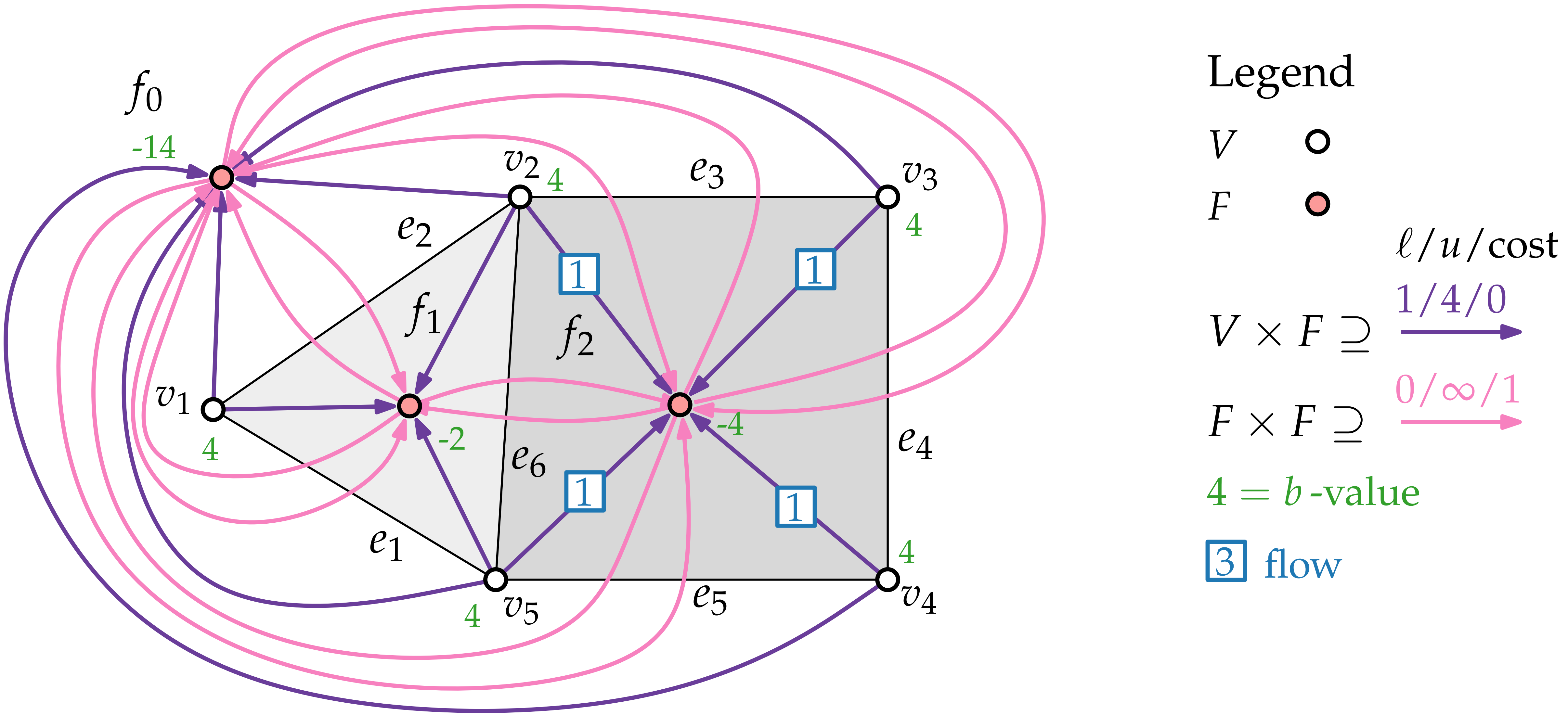
$V \times F \supseteq$ 1/4/0

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

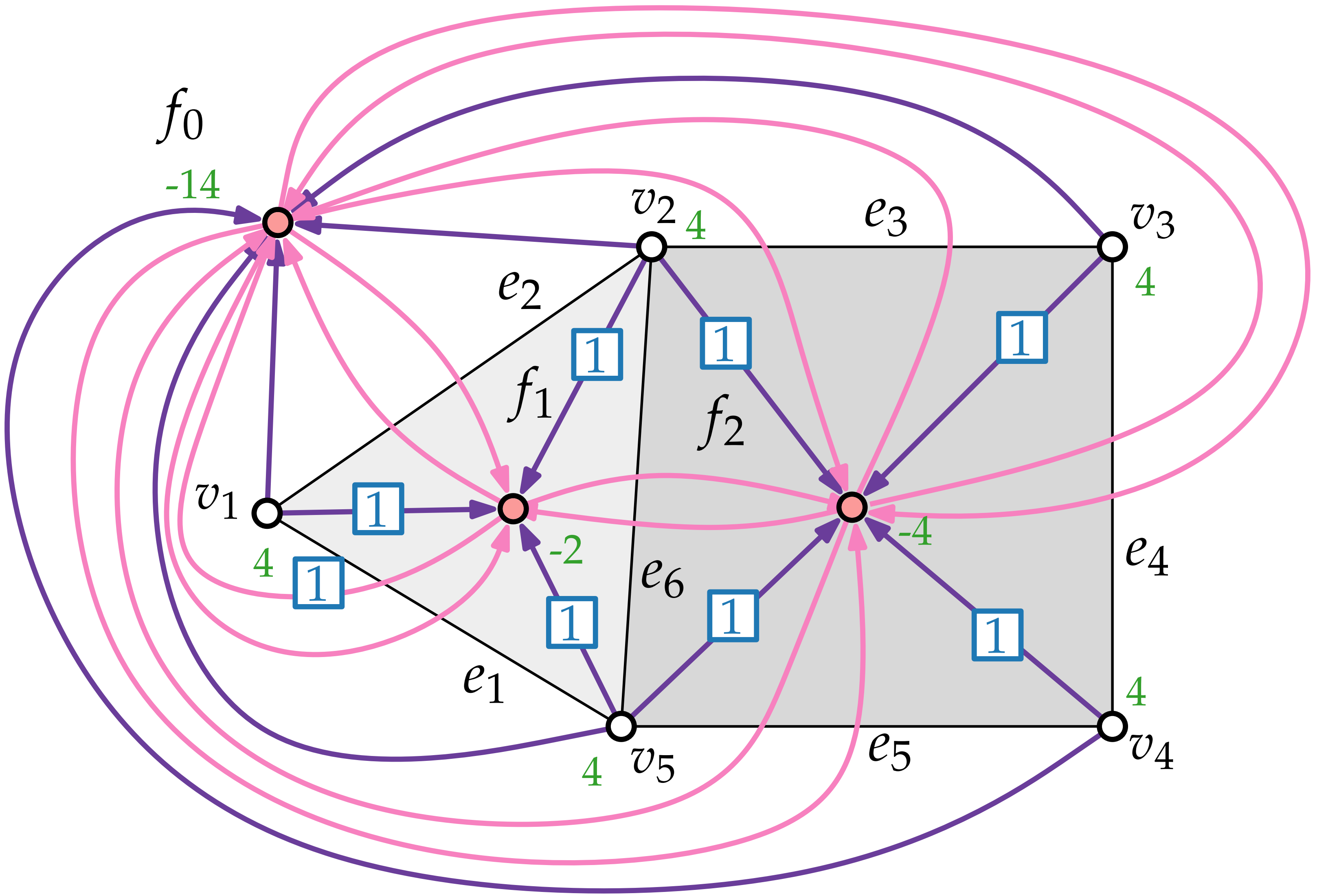
Flow Network Example

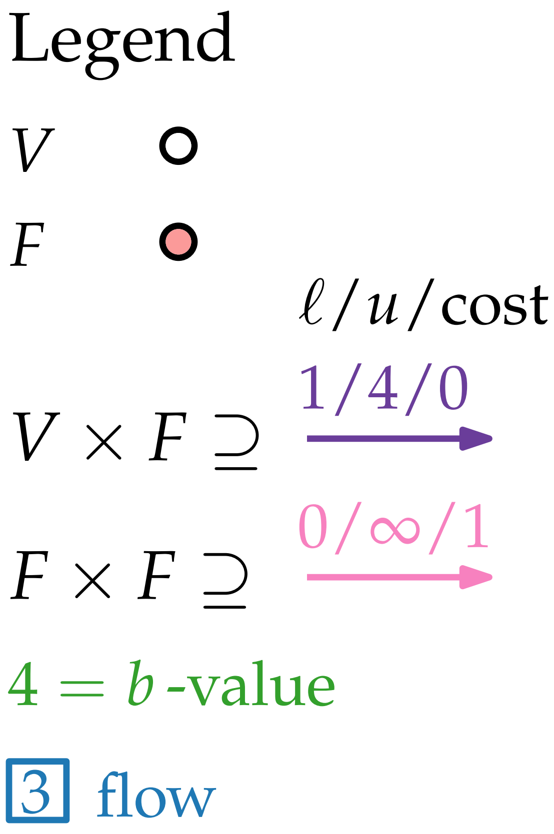


Flow Network Example

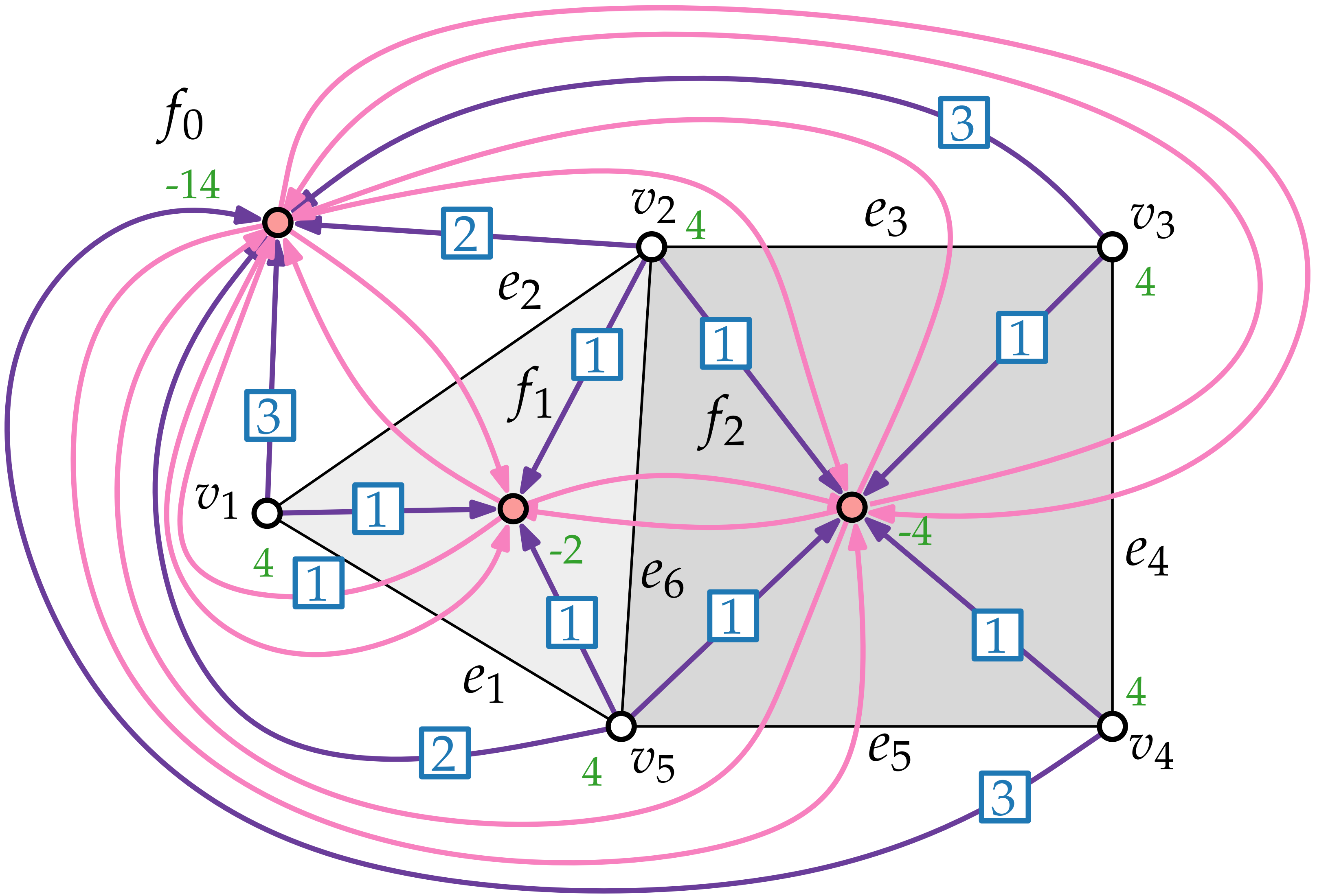


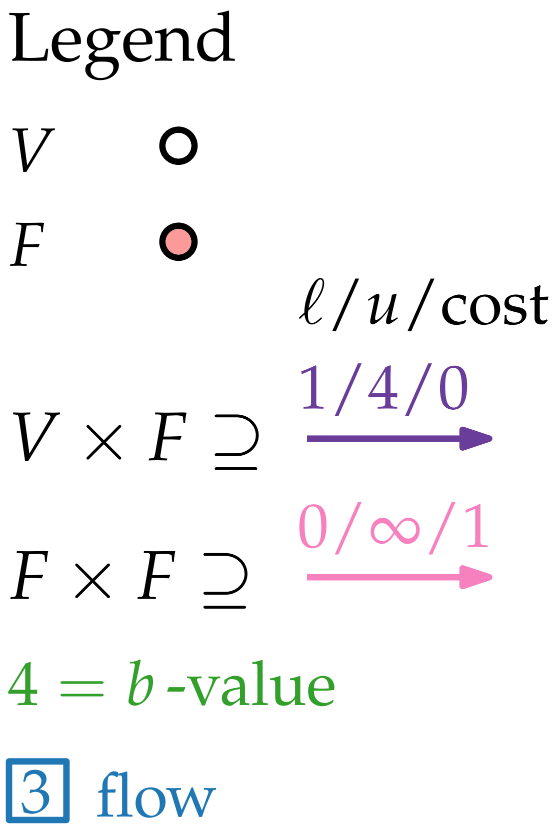
Flow Network Example



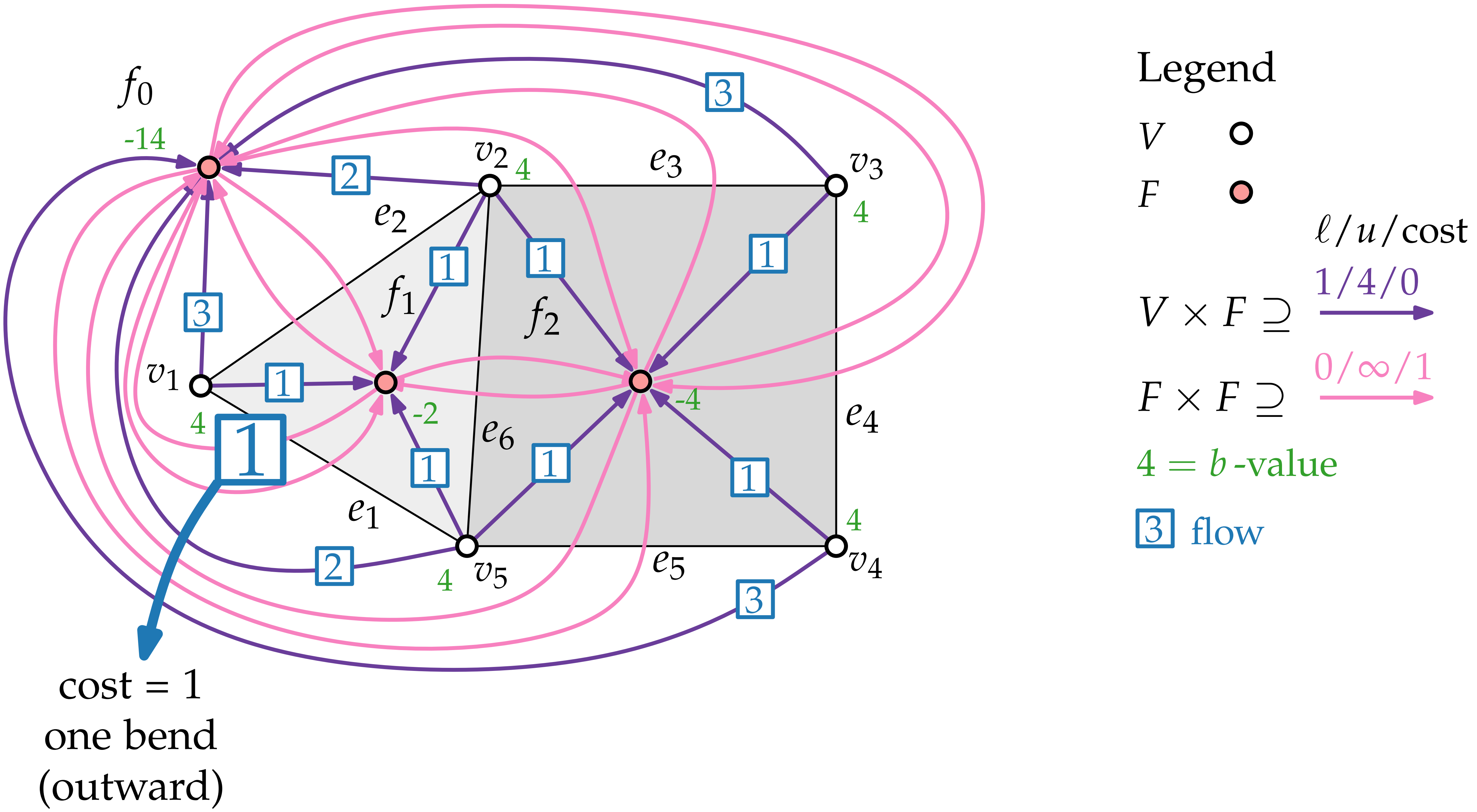


Flow Network Example

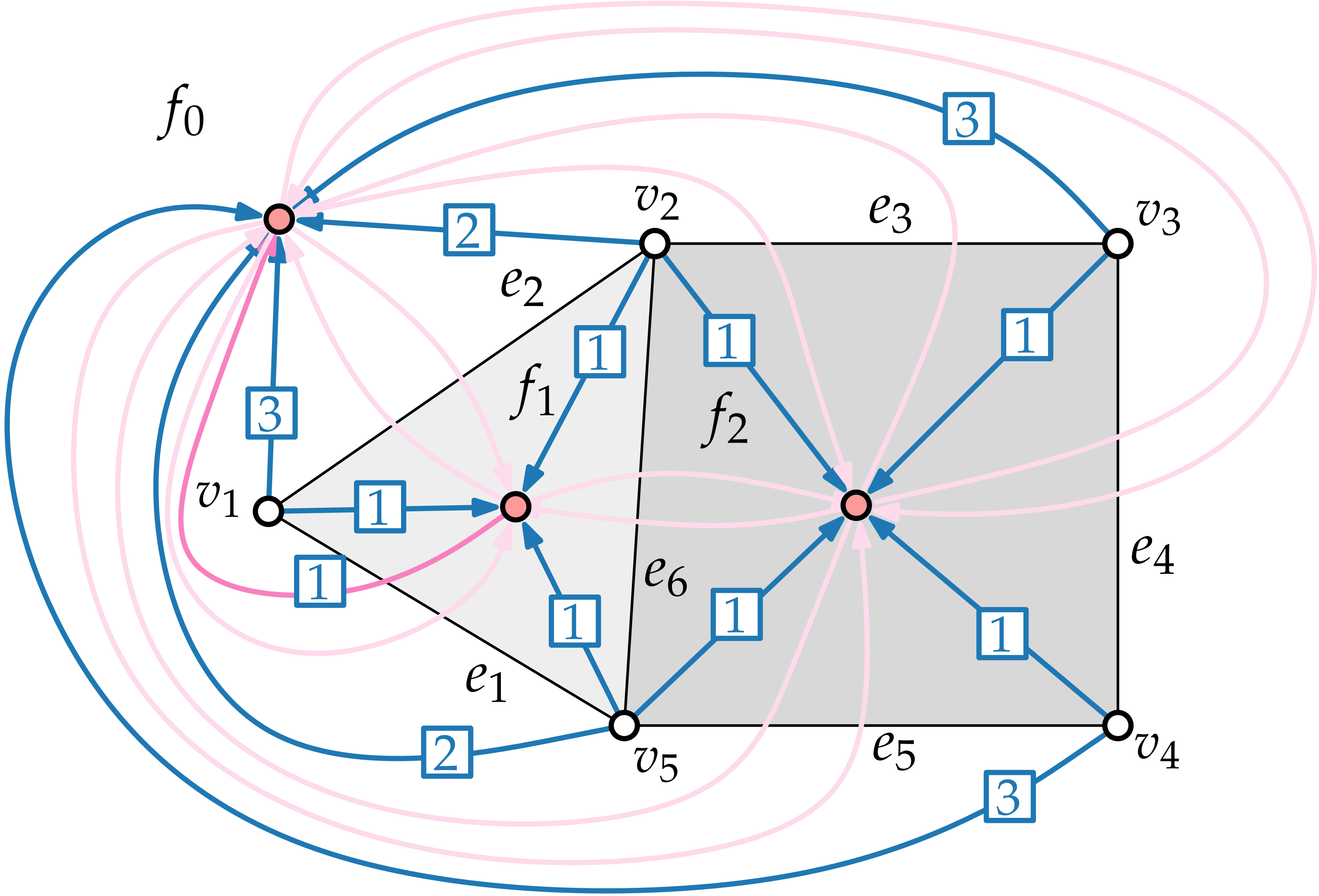


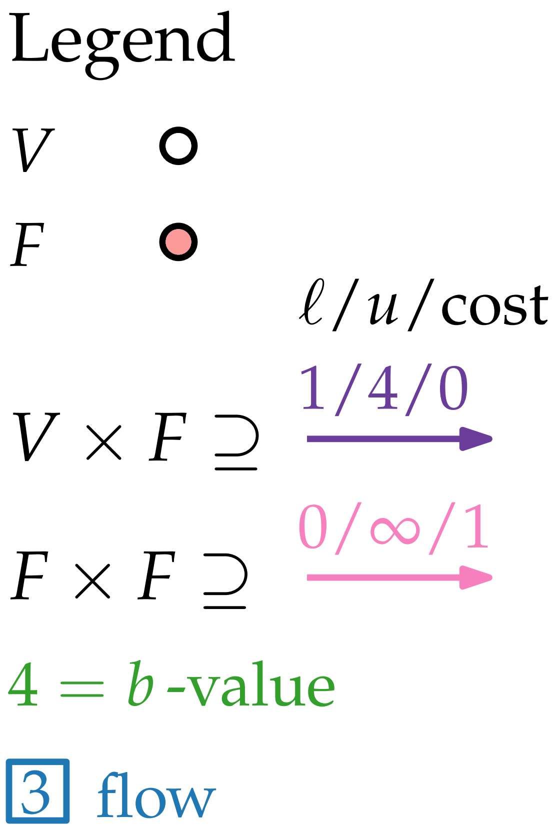


Flow Network Example

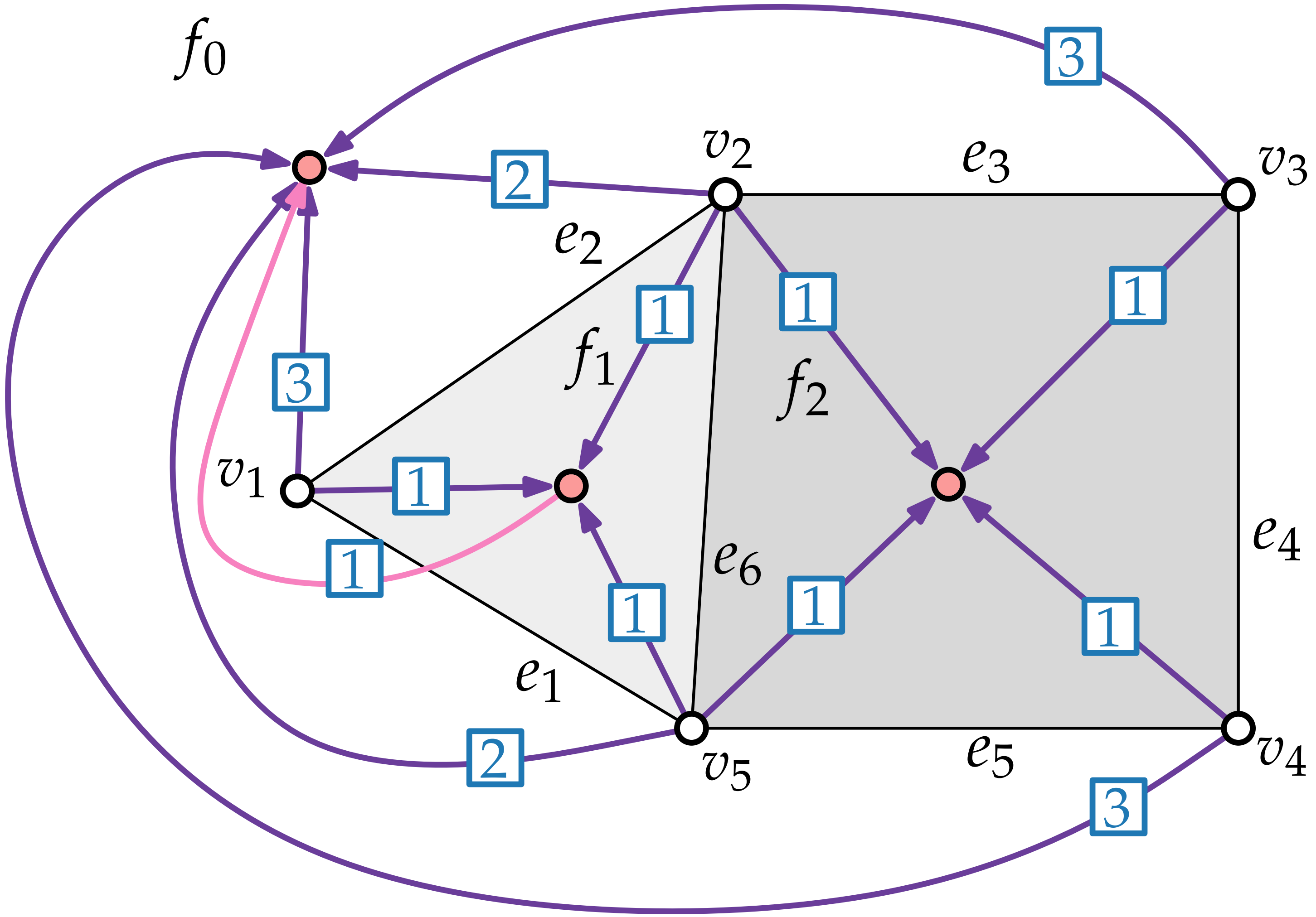


Flow Network Example





Flow Network Example



Legend

V

F

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

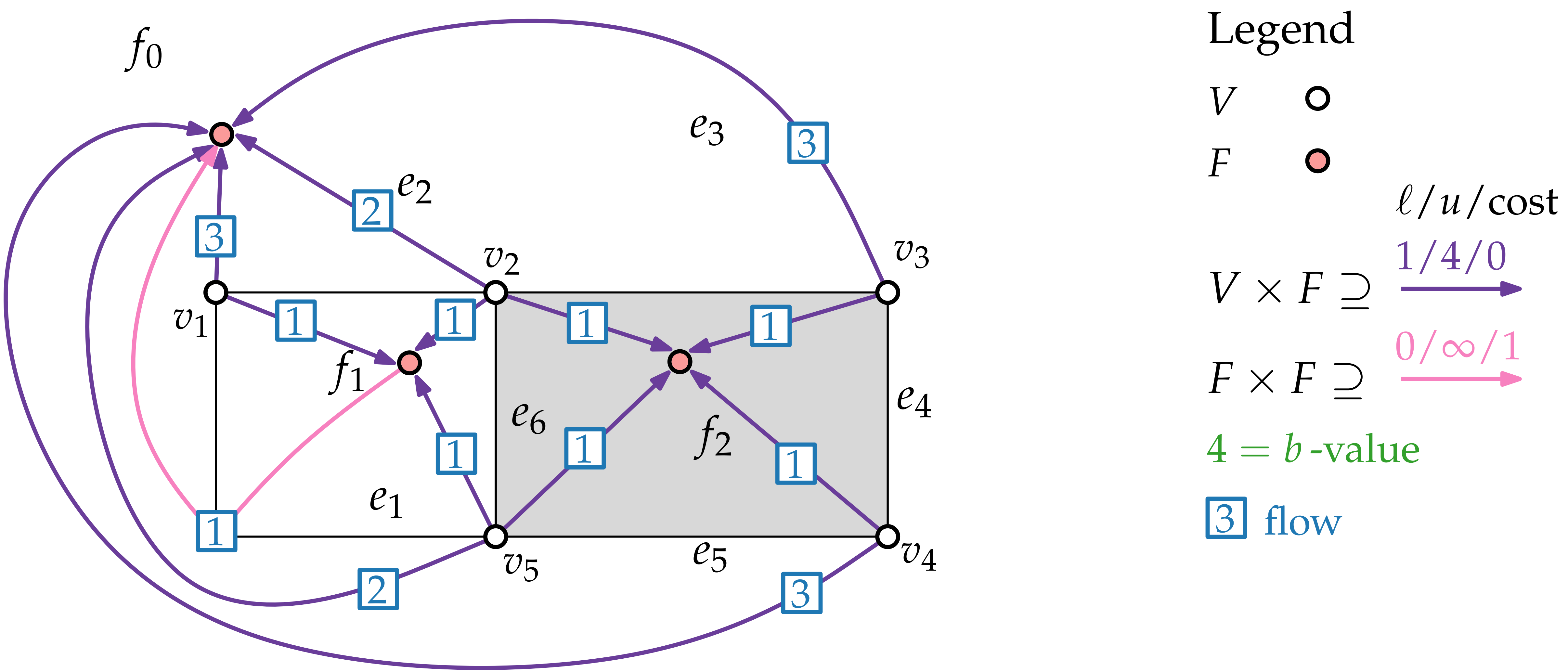
$V \times F \supseteq$ 1/4/0

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

$4 = b$-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 iff 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 iff 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 iff the flow network $N(G)$ has a valid flow X with cost k.

Proof.

$\Leftarrow$ Given valid flow X in $N(G)$ with cost k.
Construct

Bend Minimization – Result

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

Proof.

$\Leftarrow$ Given valid flow X in $N(G)$ with cost k.
Construct 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 iff the flow network $N(G)$ has a valid flow X with cost k.

Proof.

$\Leftarrow$ Given valid flow X in $N(G)$ with cost k.
Construct 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 iff 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 valid flow X in $N(G)$ with cost k.
Construct orthogonal representation $H(G)$ with k bends.

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

(H1)

(H2)

(H3)

(H4)

Bend Minimization – Result

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

(H3)

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

Proof.

$\Leftarrow$ Given valid flow X in $N(G)$ with cost k.
Construct 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)

(H3)

(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 iff 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 valid flow X in $N(G)$ with cost k.
Construct 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)

(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 iff 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 valid flow X in $N(G)$ with cost k.
Construct 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 $f = \pm 4$ ✓ 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 iff 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 valid flow X in $N(G)$ 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 iff 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 valid flow X in $N(G)$ with 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 iff the flow network $N(G)$ has a valid flow X with cost k.

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

Proof.

$\Rightarrow$ Given an orthogonal representation $H(G)$ with k bends. Construct valid flow X in $N(G)$ with 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 iff the flow network $N(G)$ has a valid flow X with cost k.

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

Proof.

$\Rightarrow$ Given an orthogonal representation $H(G)$ with k bends. Construct valid flow X in $N(G)$ with 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 iff the flow network $N(G)$ has a valid flow X with cost k.

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

Proof.

$\Rightarrow$ Given an orthogonal representation $H(G)$ with k bends.
Construct valid flow X in $N(G)$ with 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) $X(fg) = |\delta_{fg}|_0$, (e, δ_{fg}, x) describes $e \overset{*}{=} fg$ from f ✓

Bend Minimization – Result

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

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

Proof.

$\Rightarrow$ Given an orthogonal representation $H(G)$ with k bends.
Construct valid flow X in $N(G)$ with 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) $X(fg) = |\delta_{fg}|_0$, (e, δ_{fg}, x) describes $e \stackrel{*}{=} fg$ from f ✓

(N3) capacities, deficit/demand coverage ✓

Bend Minimization – Result

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

Theorem. [Tamassia '87]
A plane graph (G, F, f_0) has a valid orthogonal representation $H(G)$ with k bends iff 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 valid flow X in $N(G)$ with 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) $X(fg) = |\delta_{fg}|_0$, (e, δ_{fg}, x) describes $e \stackrel{*}{=} fg$ from f ✓

(N3) capacities, deficit/demand coverage ✓

(N4) cost = k ✓

Bend Minimization – Remarks

- From Theorem follows that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for the Min-Cost-Flow problem.

Bend Minimization – Remarks

- From Theorem follows that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for the Min-Cost-Flow problem.

Theorem. [Garg & Tamassia 1996]
The minimum cost flow problem can be solved in $O(|X^*|^{3/4} m \sqrt{\log n})$ time.

Bend Minimization – Remarks

- From Theorem follows that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for the Min-Cost-Flow problem.

Theorem. [Garg & Tamassia 1996]
The minimum 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

- From Theorem follows that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for the Min-Cost-Flow problem.

Theorem. [Garg & Tamassia 1996]
The minimum 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. [Cornelsen & Karrenbauer 2011]
The minimum cost flow problem for planar graphs with bounded costs and faze sizes can be solved in $O(n^{3/2})$ time.

Bend Minimization – Remarks

- From Theorem follows that the combinatorial orthogonal bend minimization problem for plane graphs can be solved using an algorithm for the Min-Cost-Flow problem.

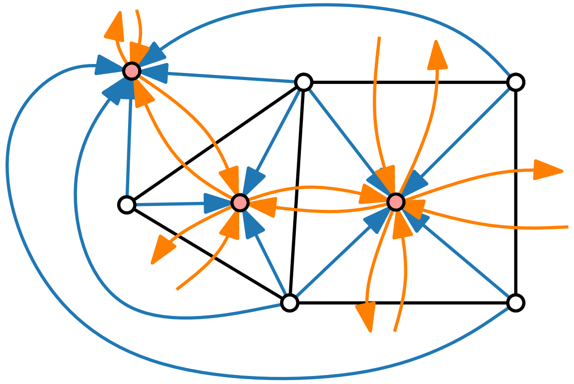
Theorem. [Garg & Tamassia 1996]
The minimum cost flow problem for planar graphs with bounded costs and vertex degrees can be solved in $O(n^{7/4}\sqrt{\log n})$ time.

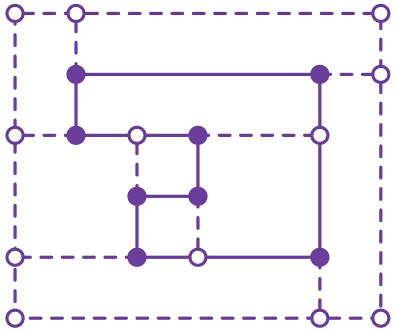
Theorem. [Cornelsen & Karrenbauer 2011]
The minimum cost flow problem for planar graphs with bounded costs and faze sizes can be solved in $O(n^{3/2})$ time.

Theorem. [Garg & Tamassia 2001]
Bend Minimization without a given combinatorial embedding is an NP-hard problem.

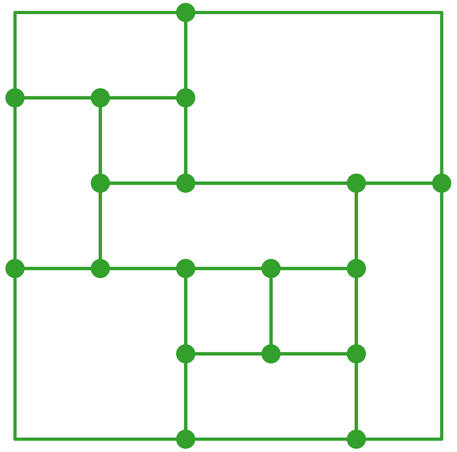
Universität Trier

Visualization of Graphs





Lecture 6:
Orthogonal Layouts



Part V:
Area Minimization

Philipp Kindermann

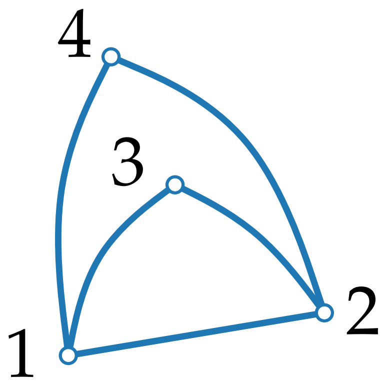
Topology – Shape – Metrics

Three-step approach: [Tamassia 1987]

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

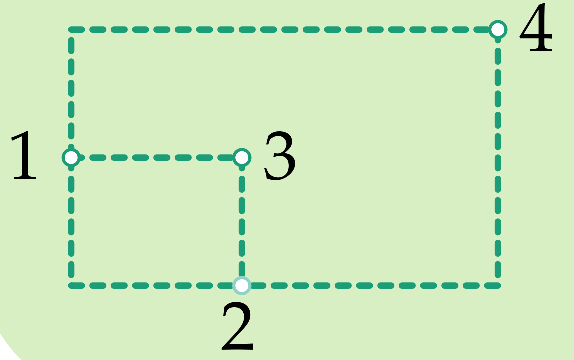
combinatorial embedding/ planarization

reduce crossings



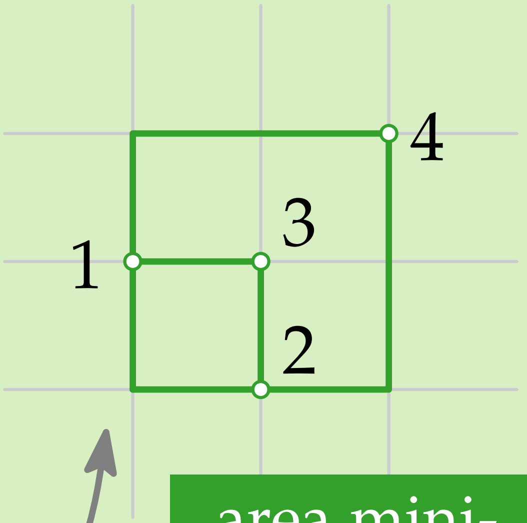
bend minimization

orthogonal representation



planar orthogonal drawing

area minimization



TOPOLOGY — SHAPE — METRICS

Compaction

Compaction problem.

Given:

Find:

Compaction

Compaction problem.

Given: ■ Plane graph $G = (V, E)$ with maximum degree 4

Find:

Compaction

Compaction problem.

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

Find:

Compaction

Compaction problem.

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

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

Compaction

Compaction problem.

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

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

Special case.

All faces are rectangles.

$\rightarrow$ Guarantees possible

Compaction

Compaction problem.

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

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

Special case.

All faces are rectangles.

$\rightarrow$ Guarantees possible
- minimum total edge length

Compaction

Compaction problem.

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

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

Special case.

All faces are rectangles.

$\rightarrow$ Guarantees possible
- minimum total edge length
- minimum area

Compaction

Compaction problem.

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

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

Special case.

All faces are rectangles.

$\rightarrow$ Guarantees possible
- minimum total edge length
- minimum area

Properties.

Compaction

Compaction problem.

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

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

Special case.

All faces are rectangles.

→ Guarantees possible
- minimum total edge length
- minimum area

Properties.

- bends only on the outer face

Compaction

Compaction problem.

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

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

Special case.

All faces are rectangles.

$\rightarrow$ Guarantees possible
- 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 $G = (V, E)$ with maximum degree 4
- Orthogonal representation $H(G)$

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

Special case.

All faces are rectangles.

$\rightarrow$ Guarantees possible
- minimum total edge length
- minimum area

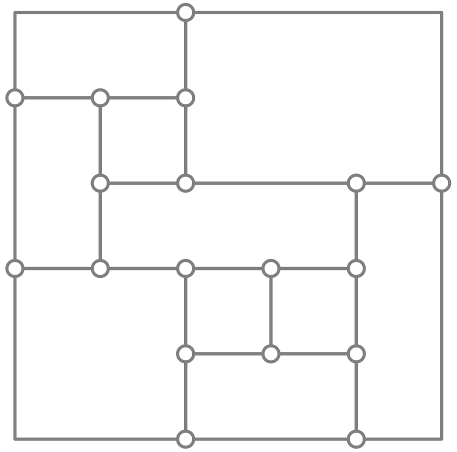
Properties.

- bends only on the outer face
- opposite sides of a face have the same length

Idea.

- Formulate flow network for horizontal/vertical compaction

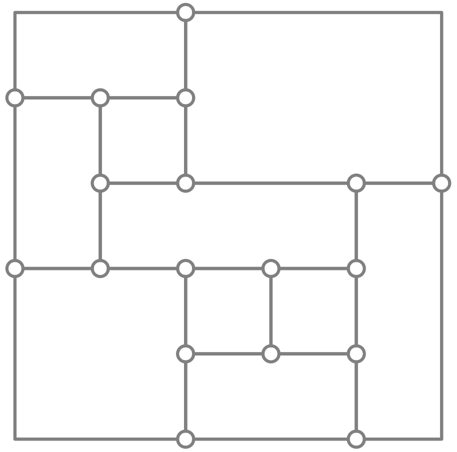
Flow Network for Edge Length Assignment



Flow Network for Edge Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

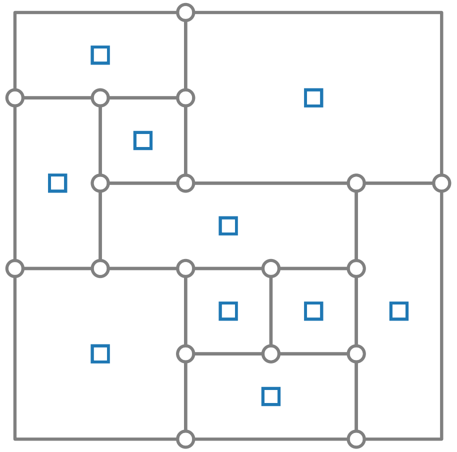


Flow Network for Edge Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

- $W_{\text{hor}} = F \setminus \{f_0\}$ □

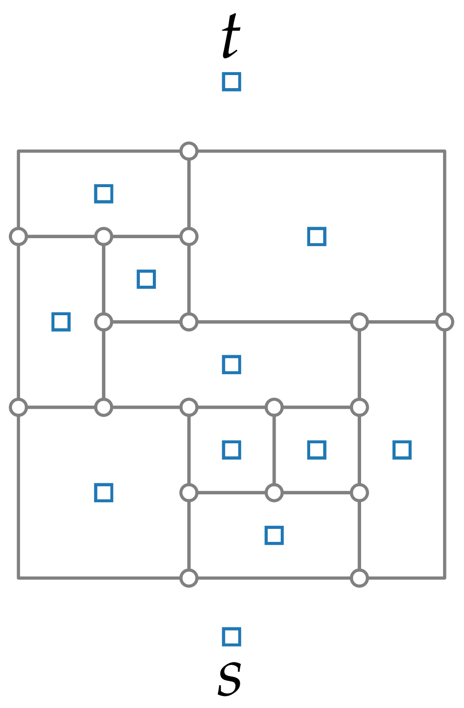


Flow Network for Edge Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

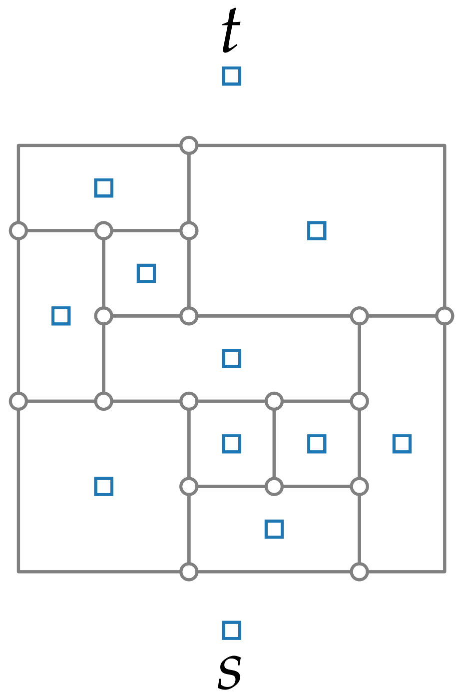
- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$ □



Flow Network for Edge Length Assignment

Definition.
Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

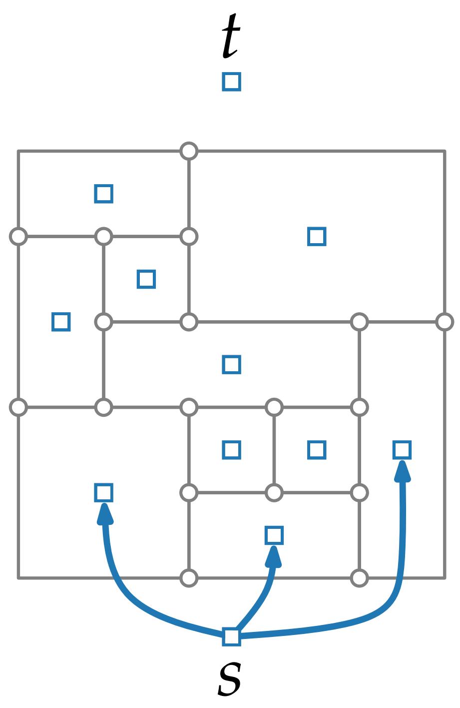
- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$



Flow Network for Edge Length Assignment

Definition.
Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

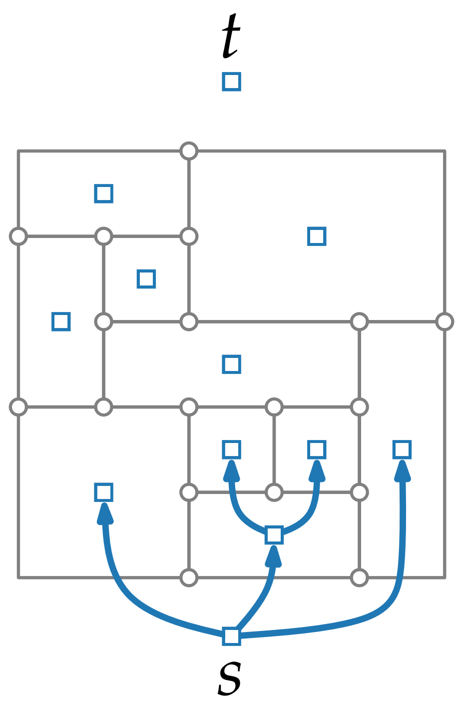
- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$ □
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$



Flow Network for Edge Length Assignment

Definition.
Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

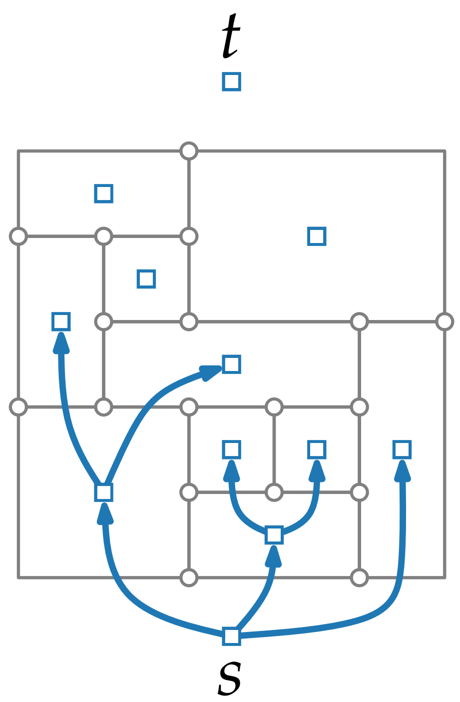
- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$



Flow Network for Edge Length Assignment

Definition.
Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

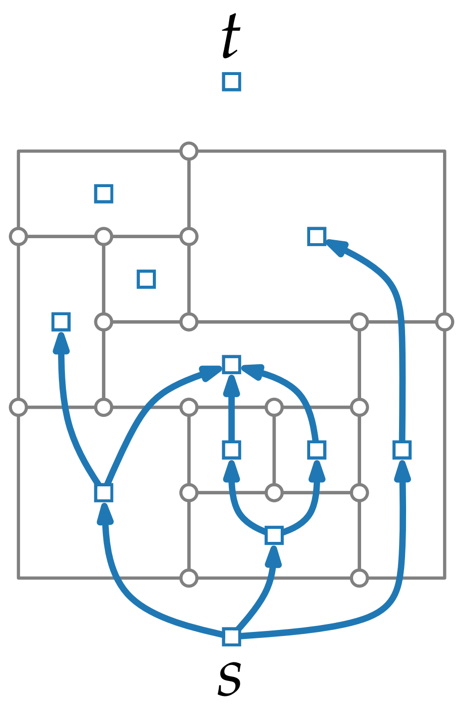
- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$



Flow Network for Edge Length Assignment

Definition.
Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

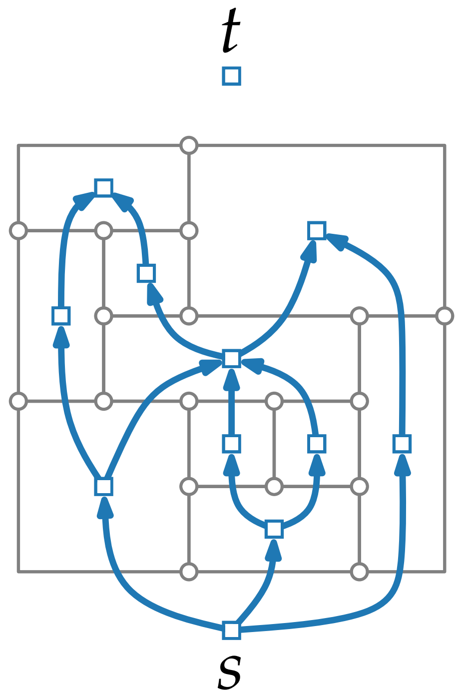
- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$



Flow Network for Edge Length Assignment

Definition.
Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

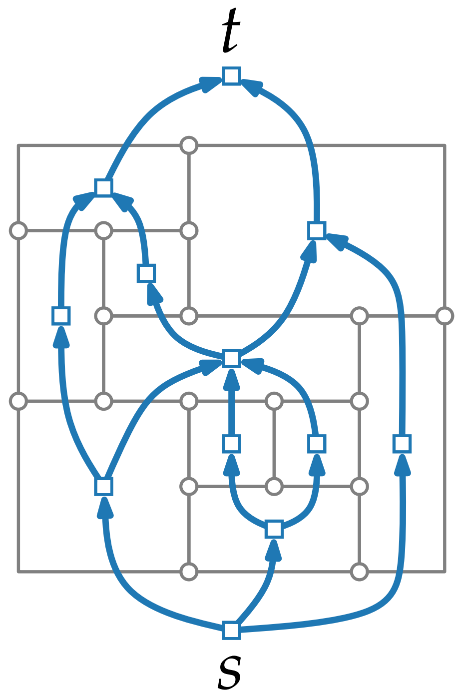
- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$



Flow Network for Edge Length Assignment

Definition.
Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\}$

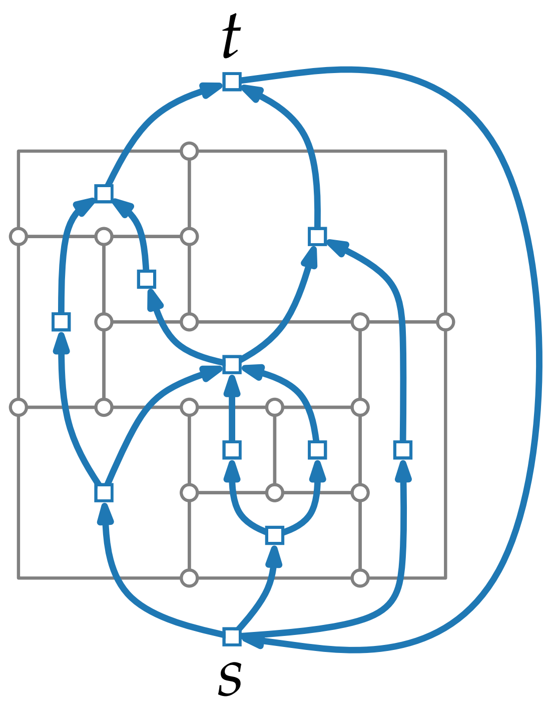


Flow Network for Edge Length Assignment

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* $g\} \cup \{(t, s)\}$

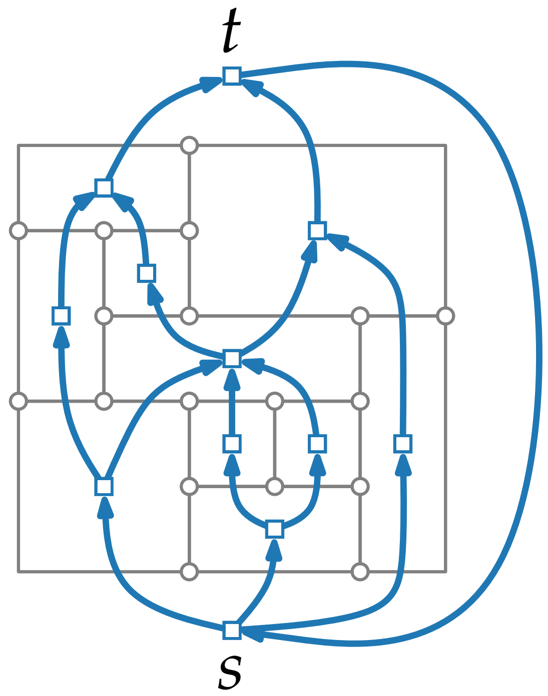


Flow Network for Edge Length Assignment

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* $g\} \cup \{(t, s)\}$
- $\ell(a) = 1 \quad \forall a \in E_{\mathrm{hor}}$

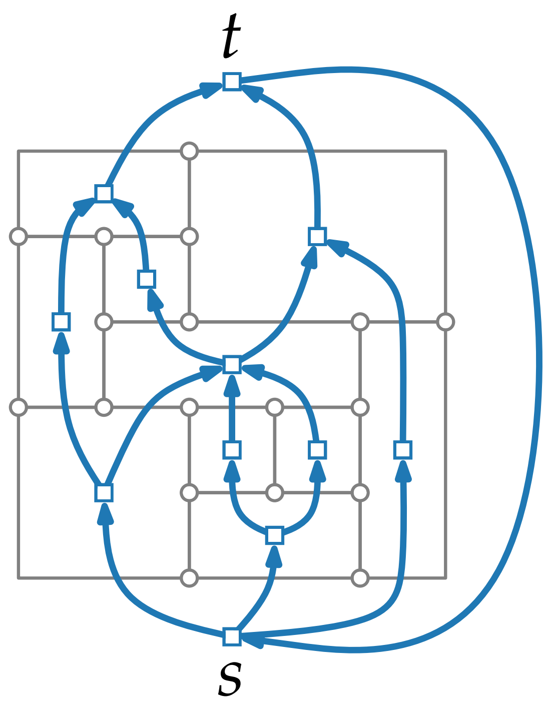


Flow Network for Edge Length Assignment

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* $g\} \cup \{(t, s)\}$
- $\ell(a) = 1 \quad \forall a \in E_{\text{hor}}$
- $u(a) = \infty \quad \forall a \in E_{\text{hor}}$

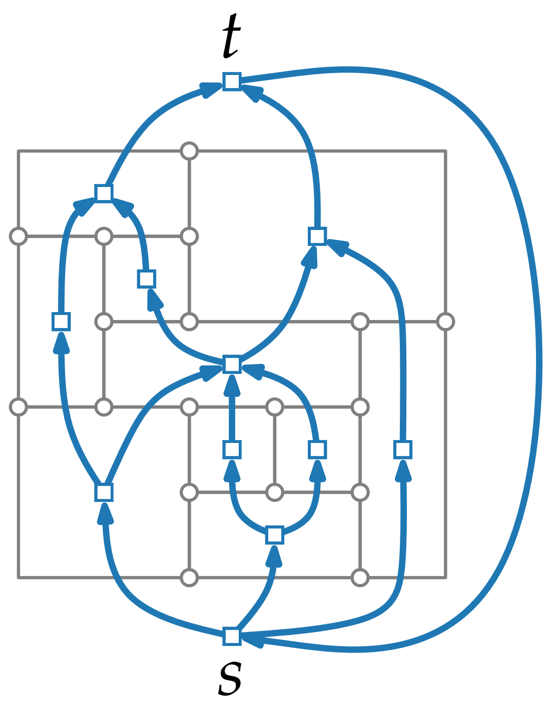


Flow Network for Edge Length Assignment

Definition.

Flow Network $N_{\text{hor}} = ((W_{\text{hor}}, E_{\text{hor}}); b; \ell; u; \text{cost})$

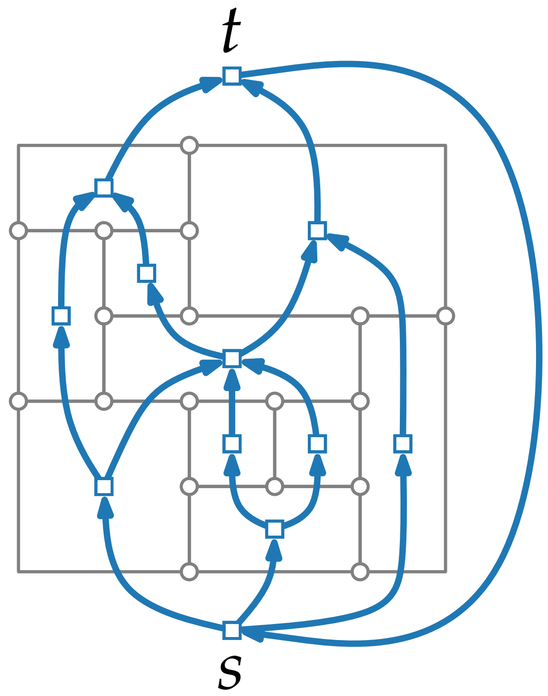
- $W_{\text{hor}} = F \setminus \{f_0\} \cup \{s, t\}$ □
- $E_{\text{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\} \cup \{(t, s)\}$
- $\ell(a) = 1 \quad \forall a \in E_{\text{hor}}$
- $u(a) = \infty \quad \forall a \in E_{\text{hor}}$
- $\text{cost}(a) = 1 \quad \forall a \in E_{\text{hor}}$



Flow Network for Edge Length Assignment

Definition.
Flow Network $N_{\mathrm{hor}} = ((W_{\mathrm{hor}}, E_{\mathrm{hor}}); b; \ell; u; \mathrm{cost})$

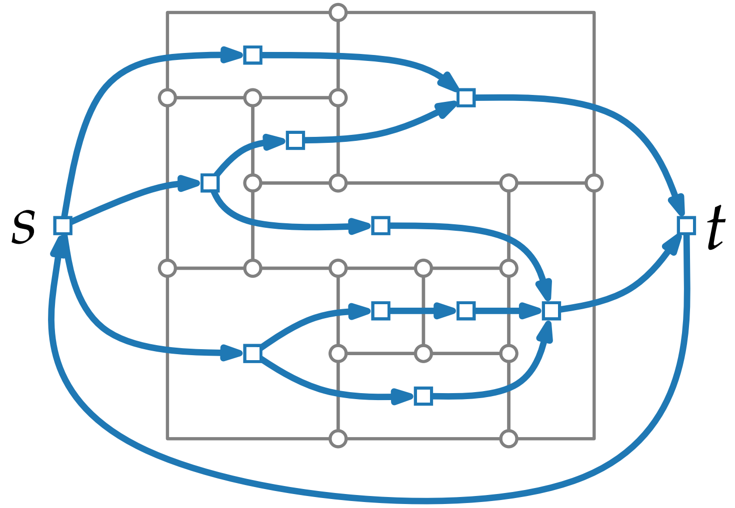
- $W_{\mathrm{hor}} = F \setminus \{f_0\} \cup \{s, t\}$ □
- $E_{\mathrm{hor}} = \{(f, g) \mid f, g$ share a *horizontal* segment and f lies *below* $g\} \cup \{(t, s)\}$
- $\ell(a) = 1 \quad \forall a \in E_{\mathrm{hor}}$
- $u(a) = \infty \quad \forall a \in E_{\mathrm{hor}}$
- $\mathrm{cost}(a) = 1 \quad \forall a \in E_{\mathrm{hor}}$
- $b(f) = 0 \quad \forall f \in W_{\mathrm{hor}}$



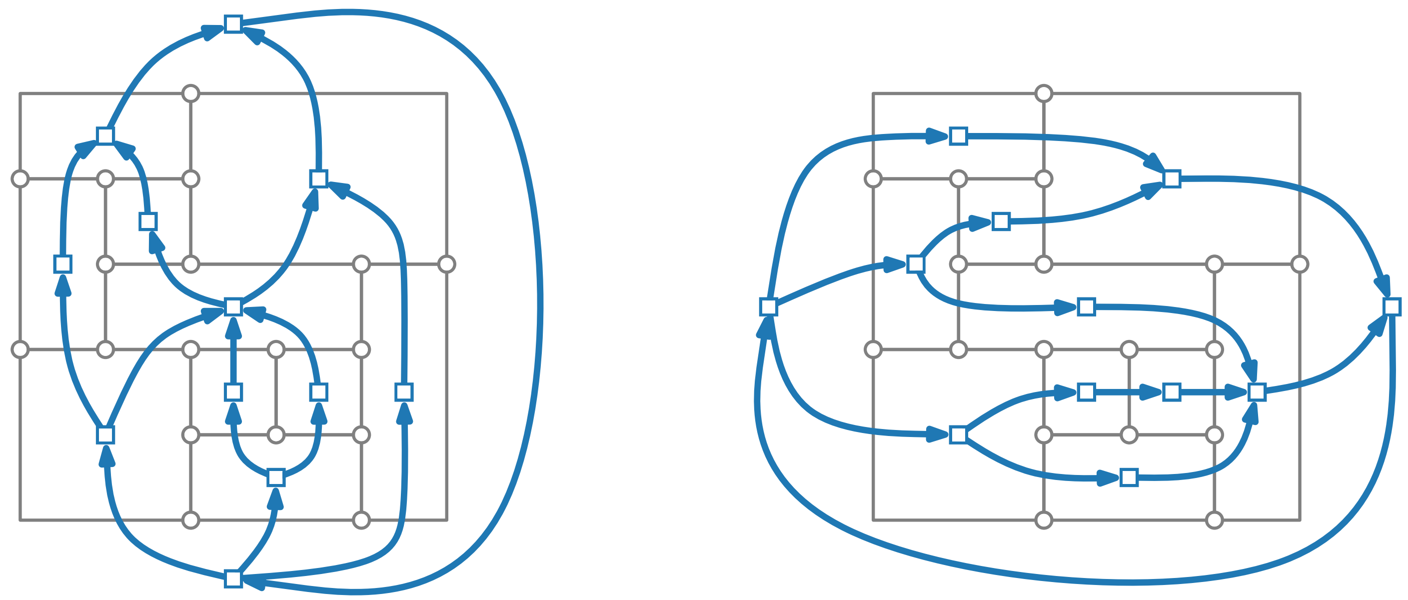
Flow Network for Edge Length Assignment

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 $g\} \cup \{(t, s)\}$
- $\ell(a) = 1 \quad \forall a \in E_{\text{ver}}$
- $u(a) = \infty \quad \forall a \in E_{\text{ver}}$
- $\text{cost}(a) = 1 \quad \forall a \in E_{\text{ver}}$
- $b(f) = 0 \quad \forall f \in W_{\text{ver}}$

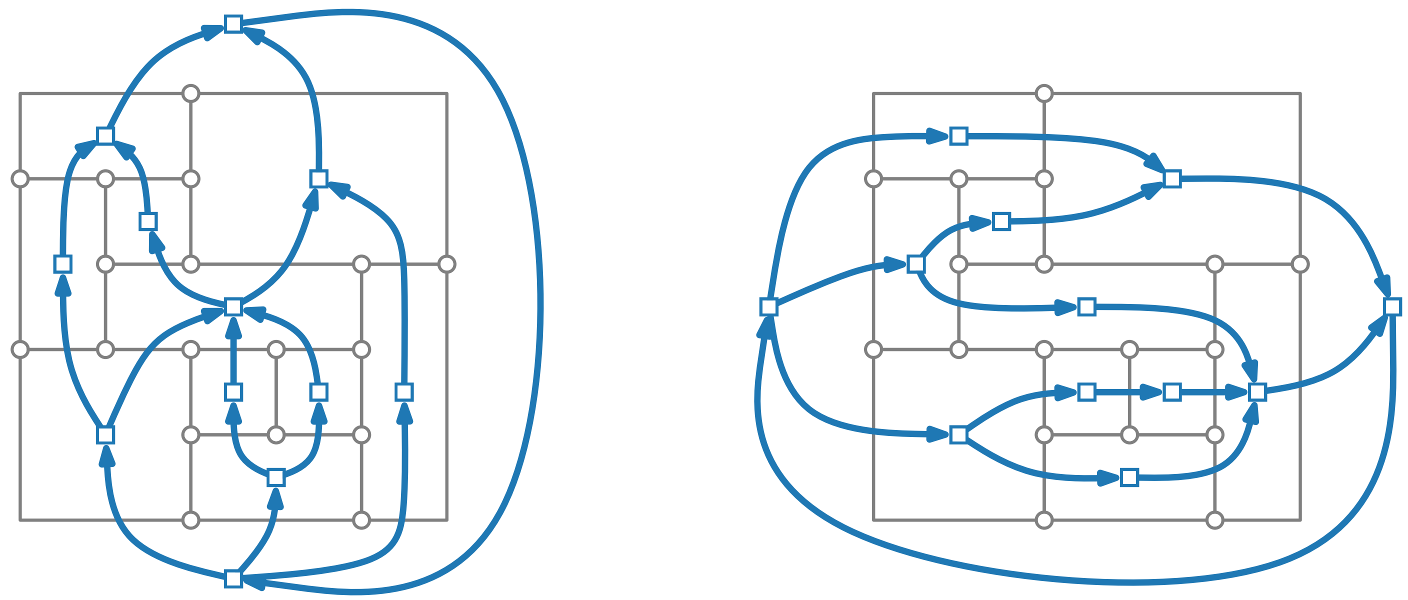


Compaction – Result



Theorem.
Valid min-cost-flows for N_{hor} and N_{ver} exists iff corresponding edge lenghts induce orthogonal drawing.

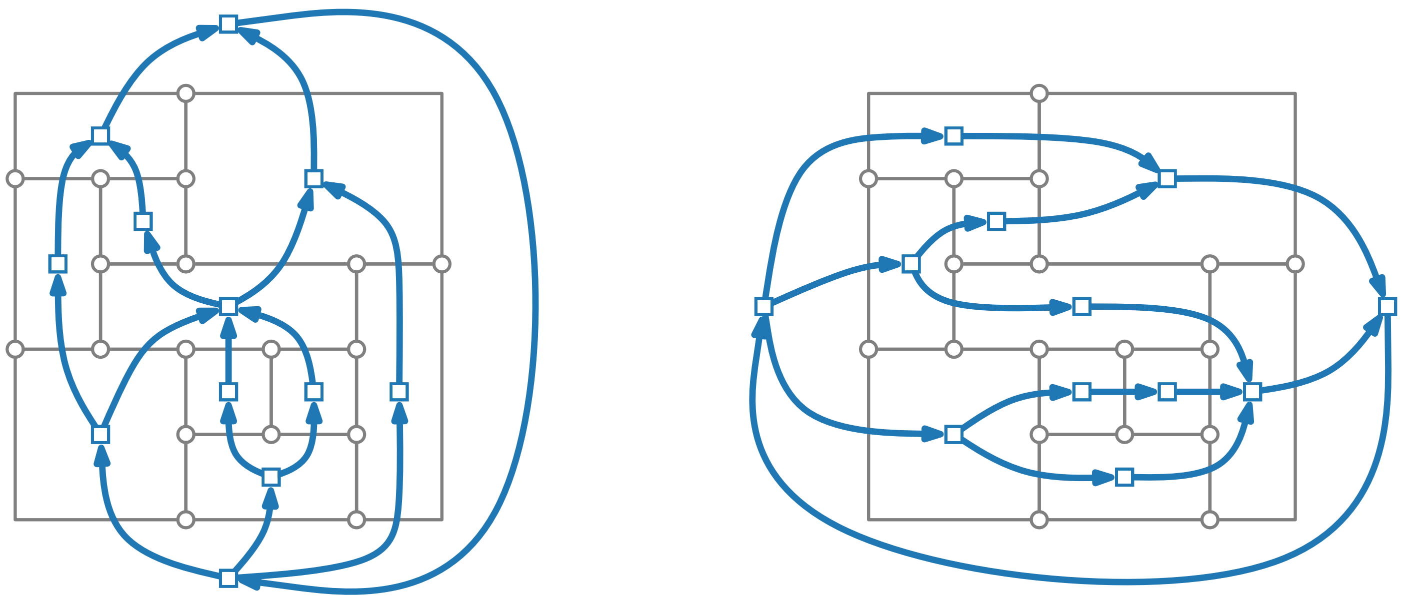
Compaction – Result



Theorem.
Valid min-cost-flows for N_{hor} and N_{ver} exists iff corresponding edge lenghts induce orthogonal drawing.

What values of the drawing represent the following?

Compaction – Result

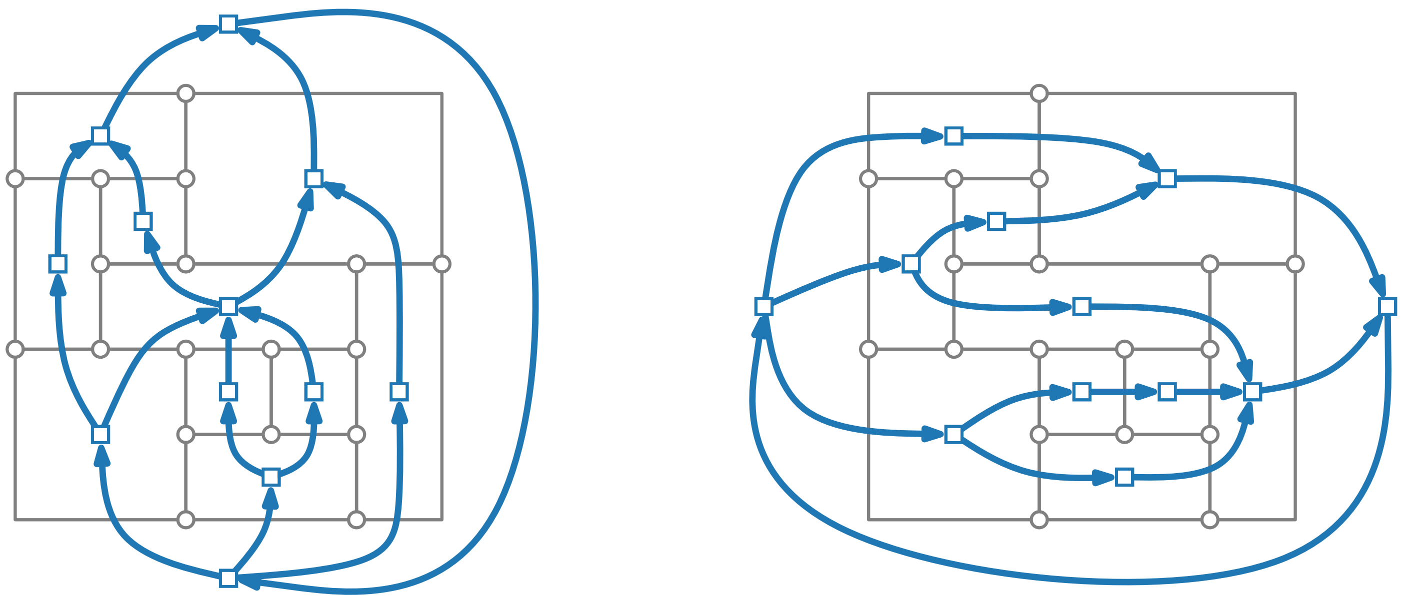


Theorem.
Valid min-cost-flows for N_{hor} and N_{ver} exists iff corresponding edge lenghts induce orthogonal drawing.

What values of the drawing represent the following?

- $|X_{\text{hor}}(t,s)|$ and $|X_{\text{ver}}(t,s)|$?

Compaction – Result

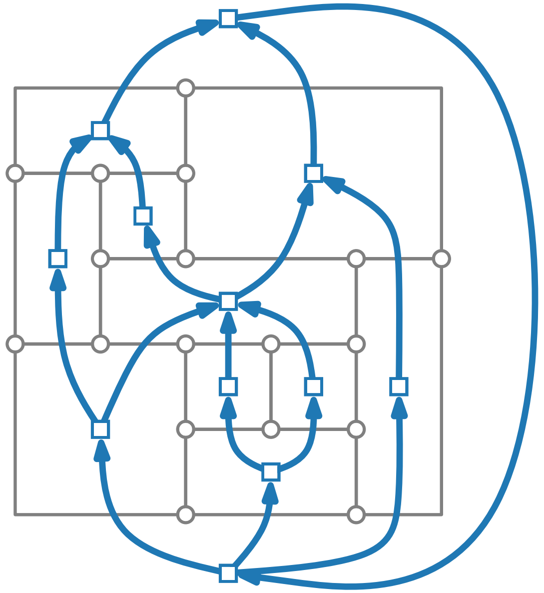


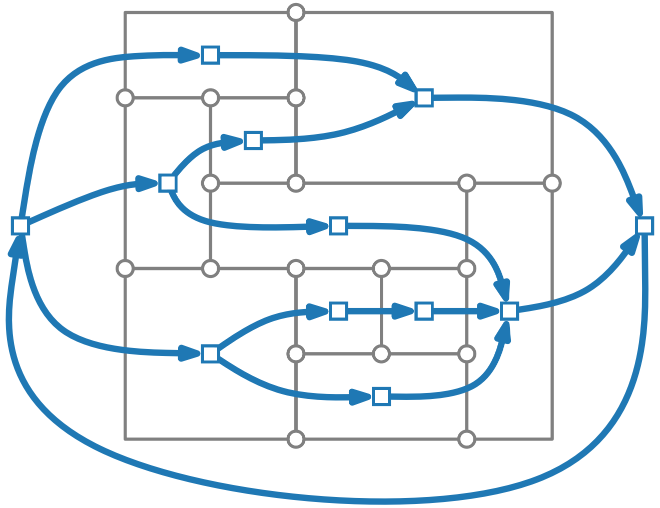
Theorem.
Valid min-cost-flows for N_{hor} and N_{ver} exists iff corresponding edge lenghts induce orthogonal drawing.

What values of the drawing represent the following?

- $|X_{\text{hor}}(t,s)|$ and $|X_{\text{ver}}(t,s)|$? width and height of drawing

Compaction – Result



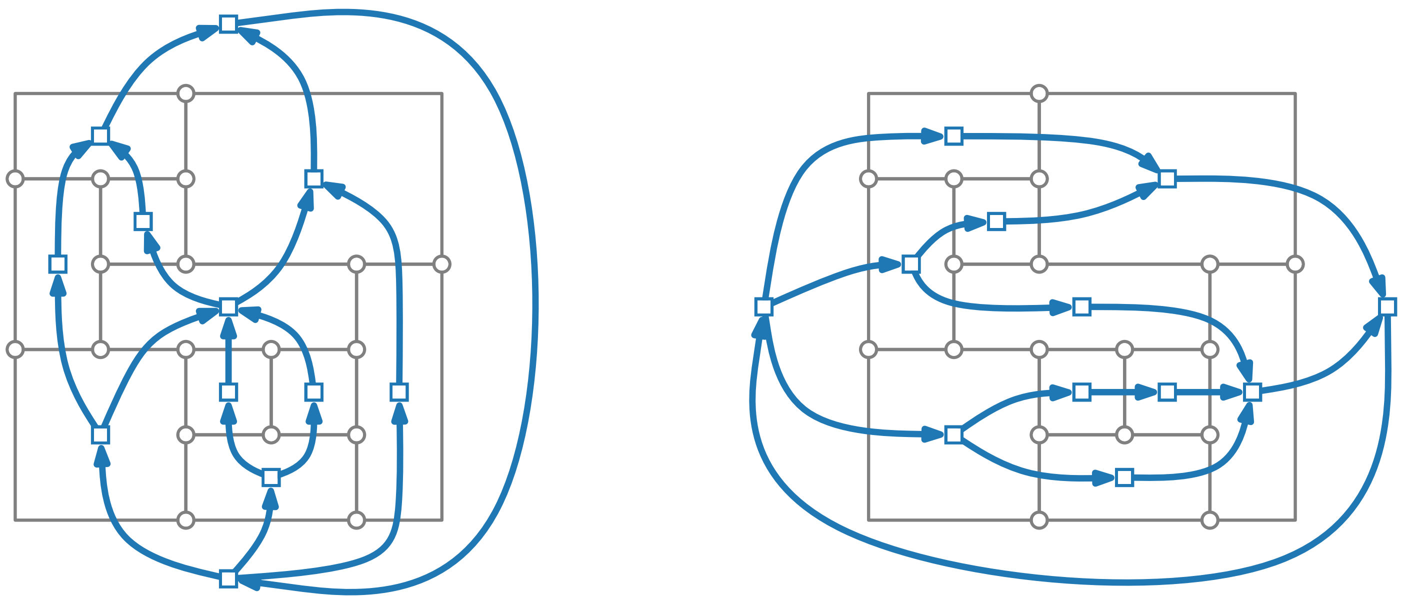


Theorem.
Valid min-cost-flows for N_{hor} and N_{ver} exists iff corresponding edge lenghts induce orthogonal drawing.

What values of the drawing represent the following?

- $|X_{\text{hor}}(t,s)|$ and $|X_{\text{ver}}(t,s)|$? width and height of drawing
- $\sum_{e \in E_{\text{hor}}} X_{\text{hor}}(e) + \sum_{e \in E_{\text{ver}}} X_{\text{ver}}(e)$

Compaction – Result

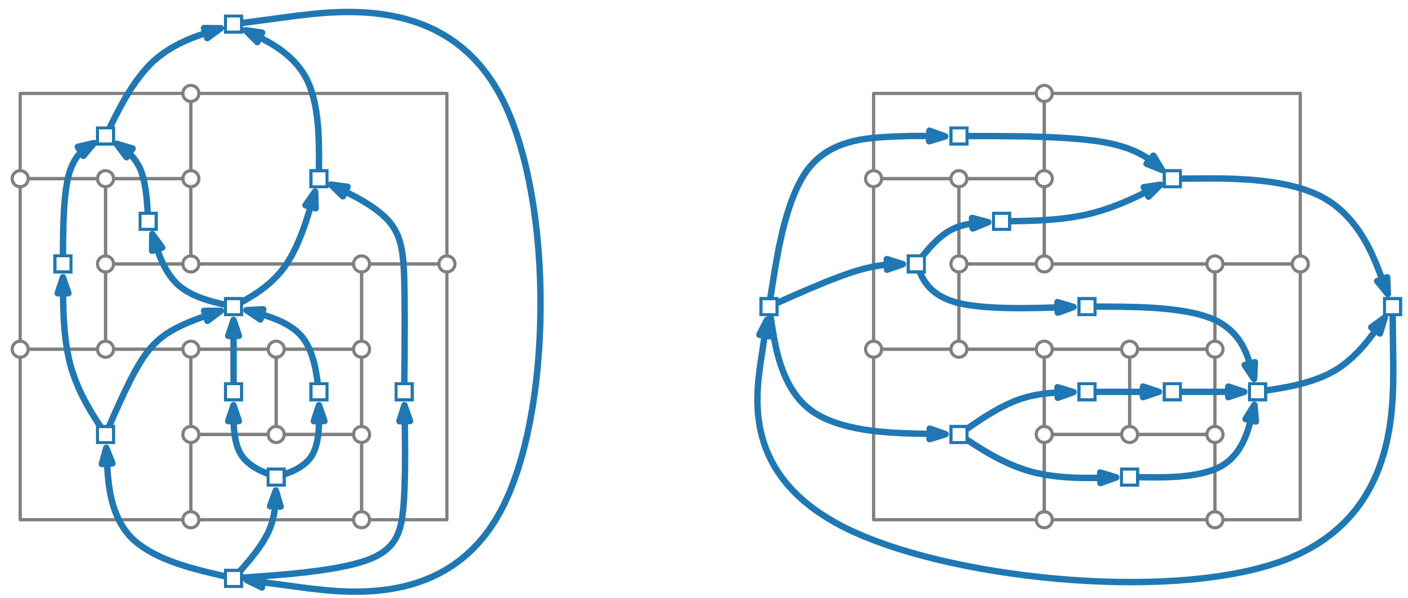


Theorem.
Valid min-cost-flows for N_{hor} and N_{ver} exists iff corresponding edge lenghts induce orthogonal drawing.

What values of the drawing represent the following?

- $|X_{\text{hor}}(t,s)|$ and $|X_{\text{ver}}(t,s)|$? width and height of 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



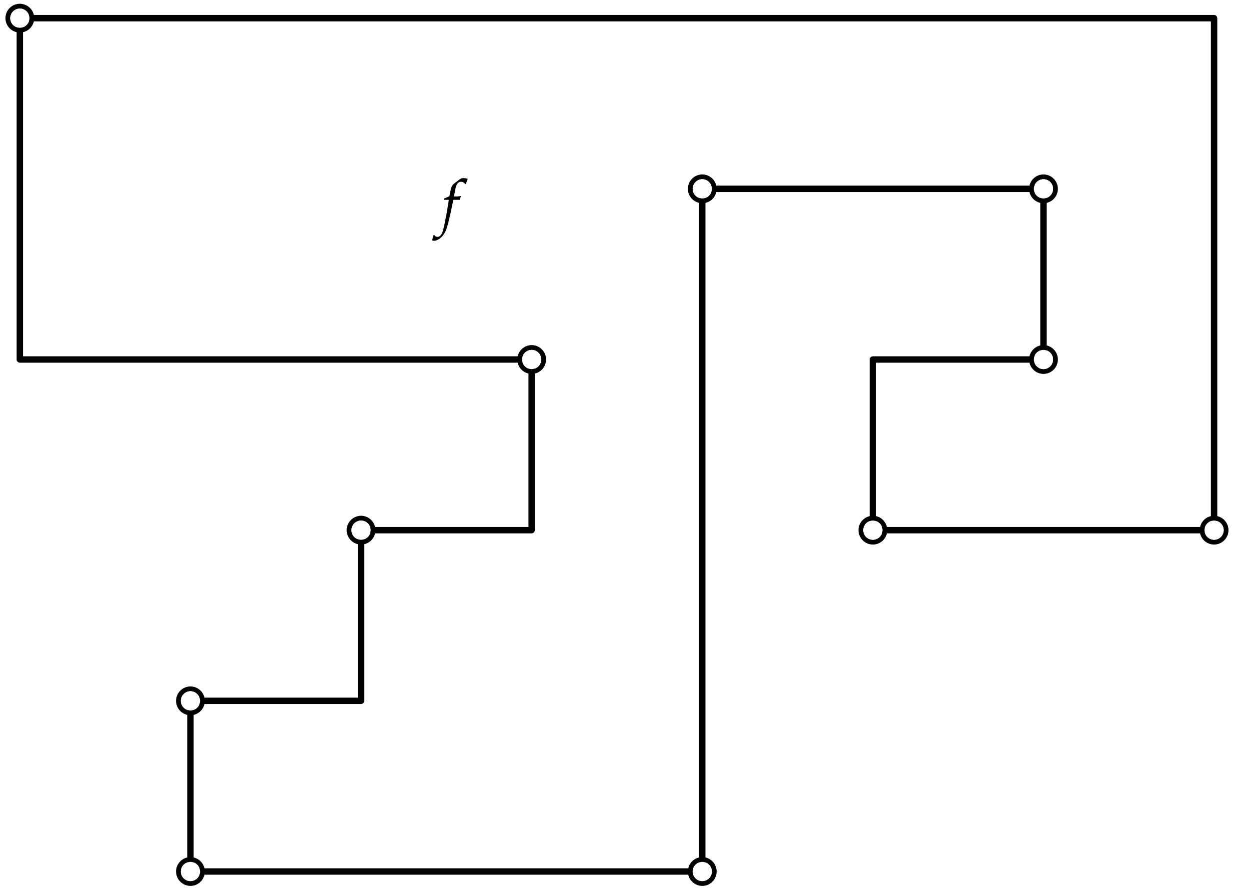
What if not all faces rectangular?

Theorem.
Valid min-cost-flows for N_{hor} and N_{ver} exists iff corresponding edge lenghts induce orthogonal drawing.

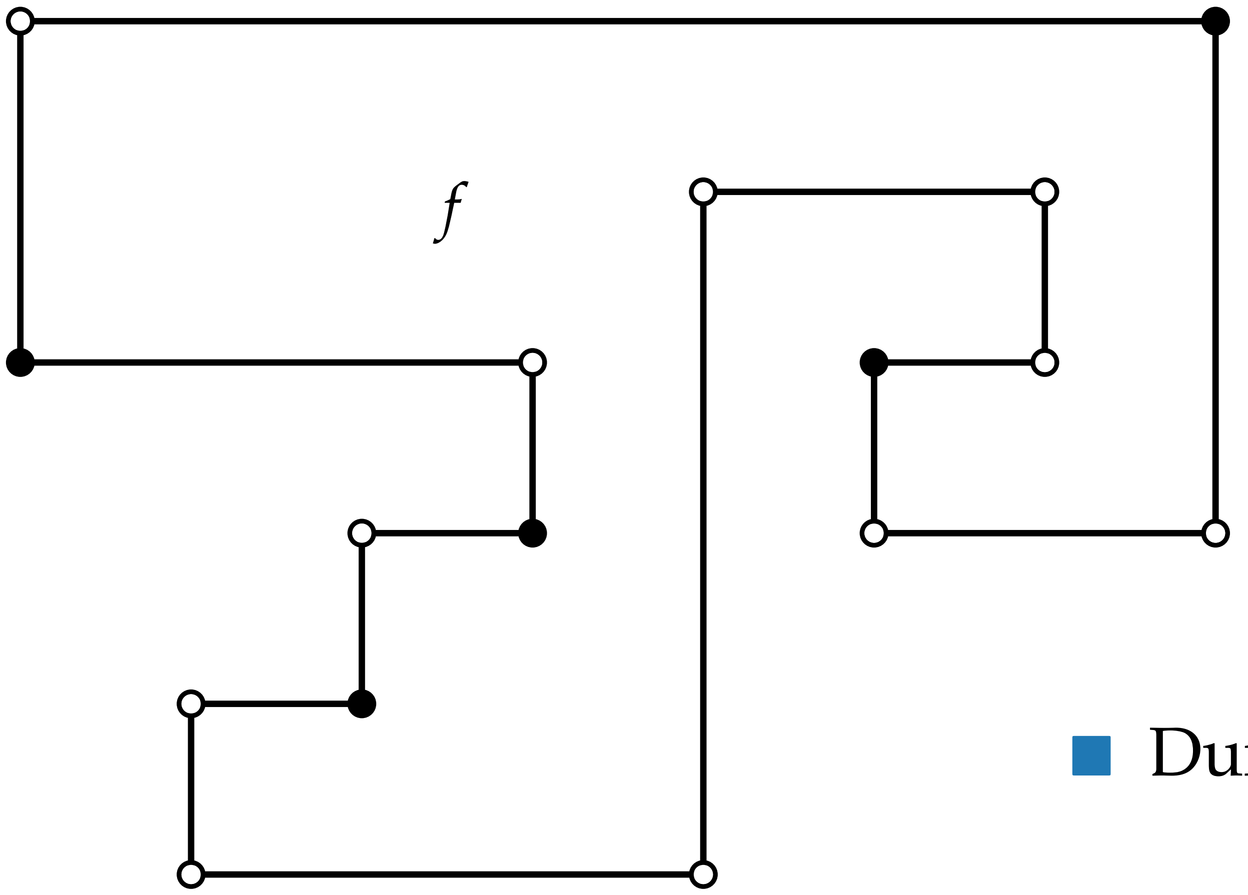
What values of the drawing represent the following?

- $|X_{\text{hor}}(t,s)|$ and $|X_{\text{ver}}(t,s)|$? — width and height of 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, H) – Inner Face

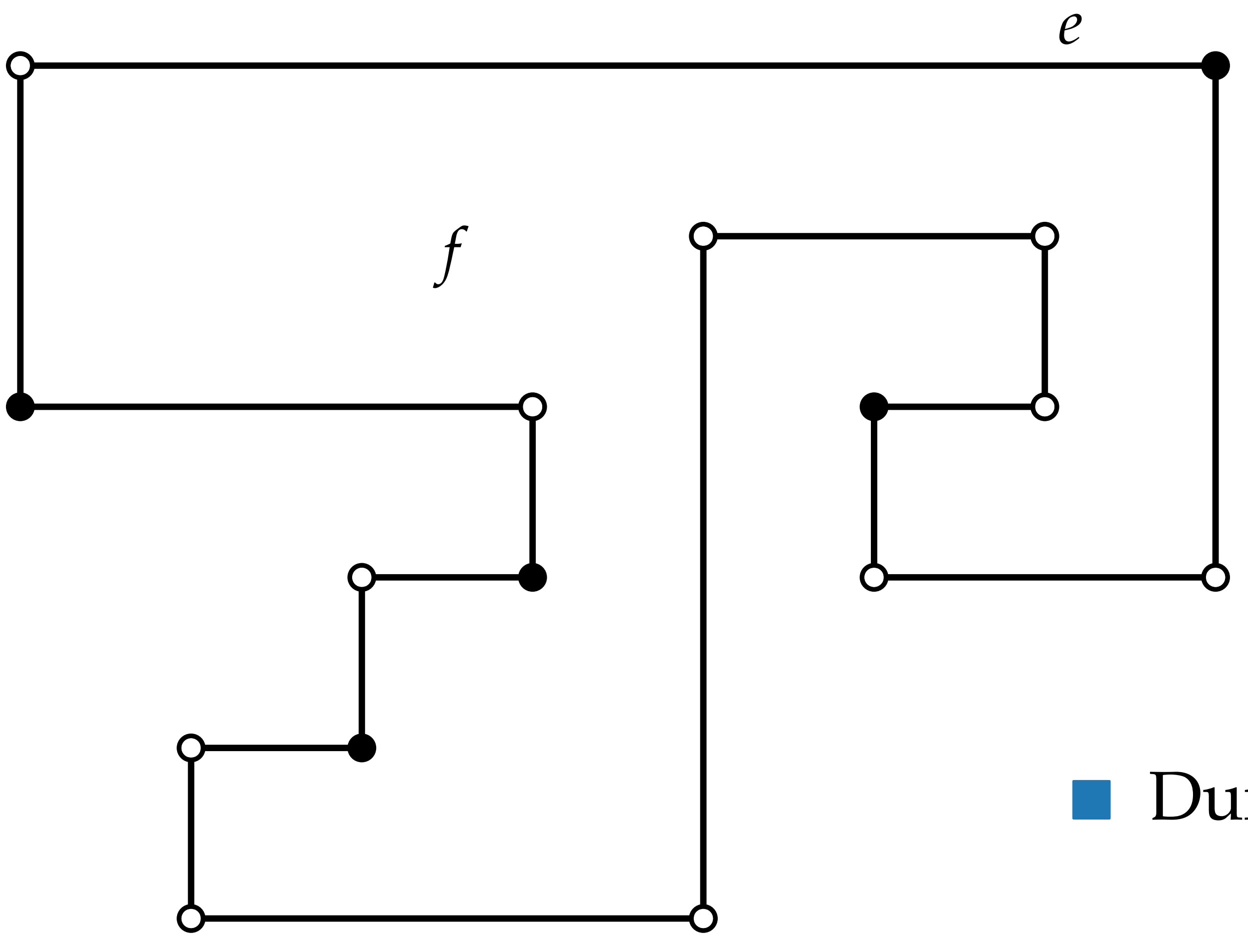


Refinement of (G, H) – Inner Face



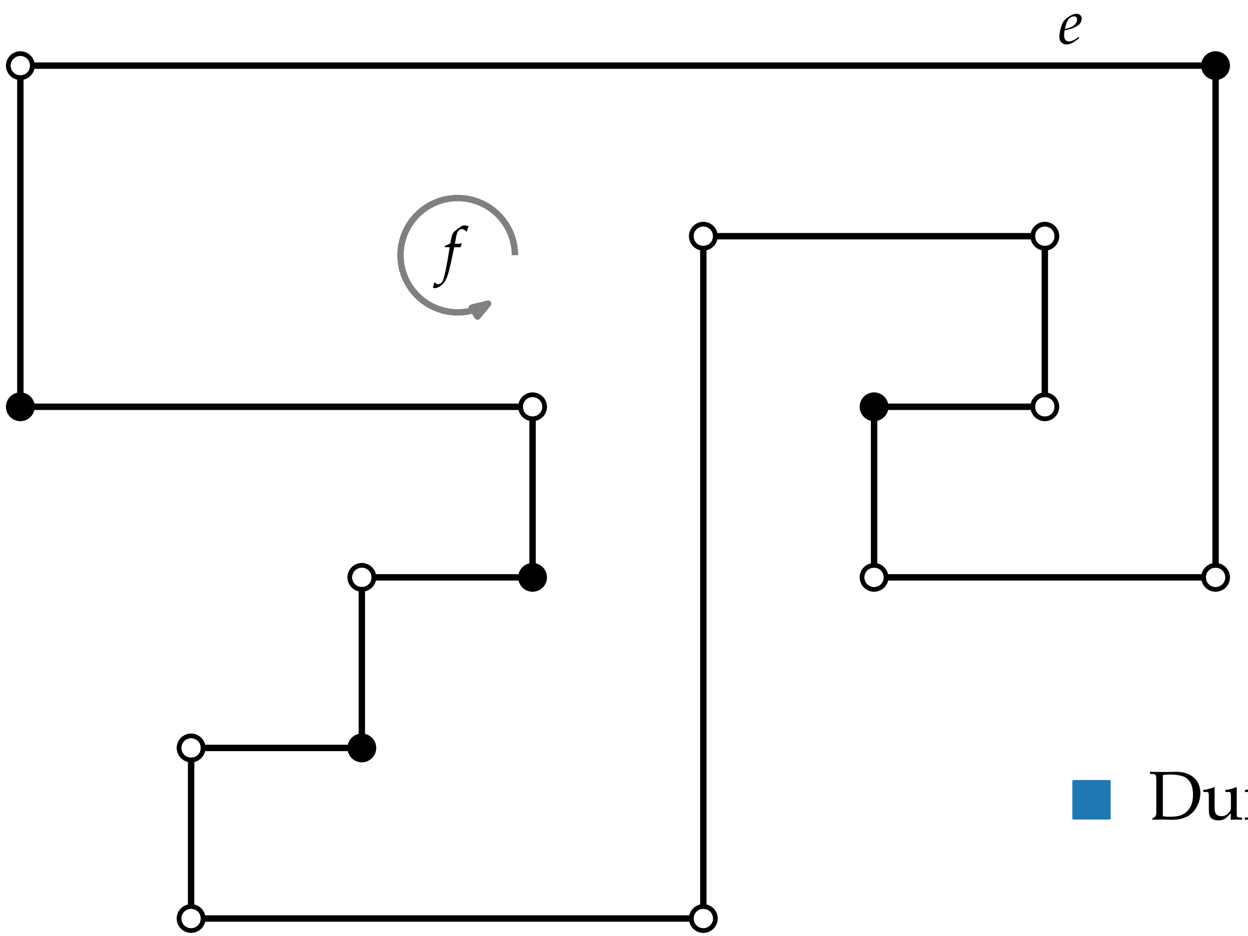
- Dummy vertices for bends

Refinement of (G, H) – Inner Face



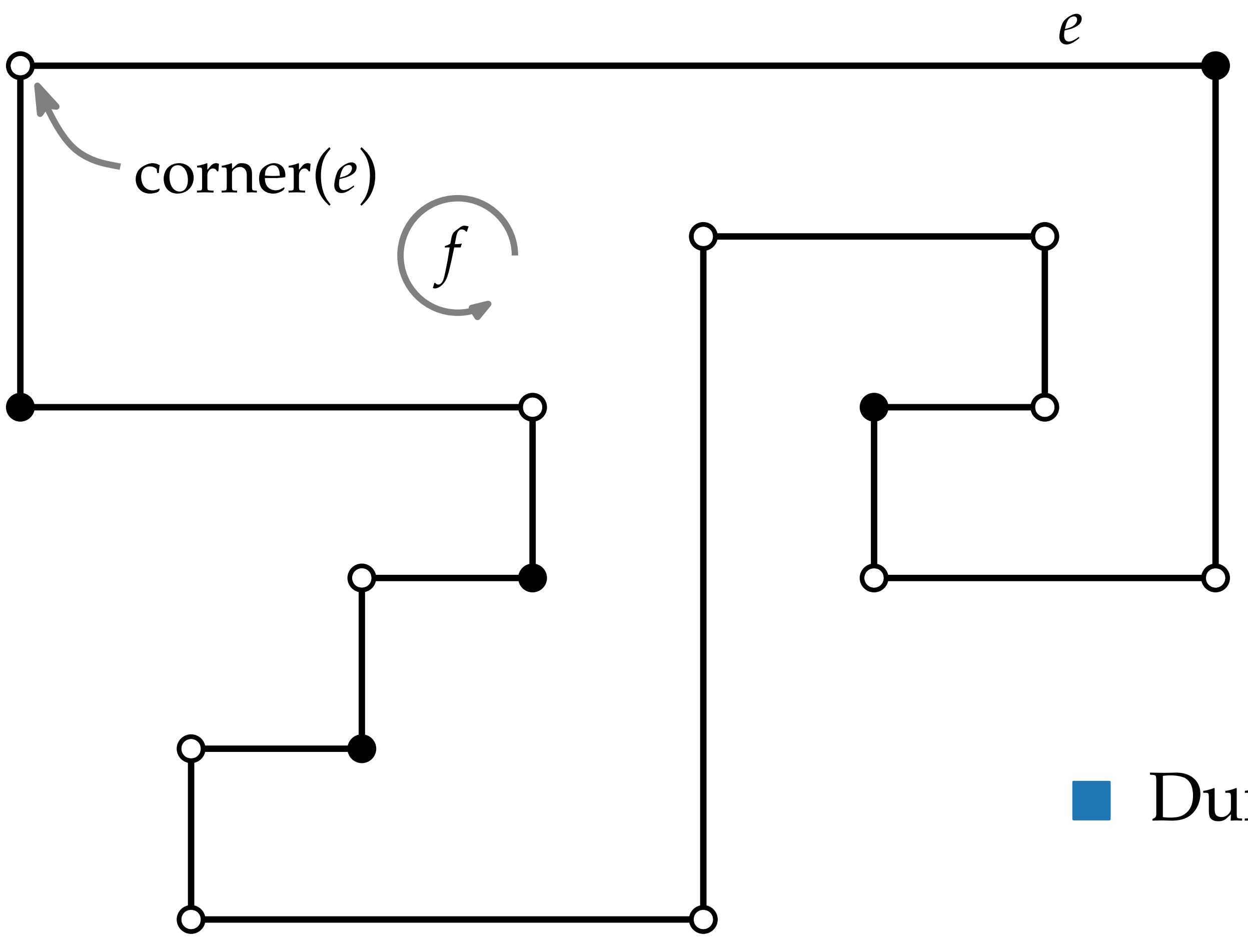
- Dummy vertices for bends

Refinement of (G, H) – Inner Face



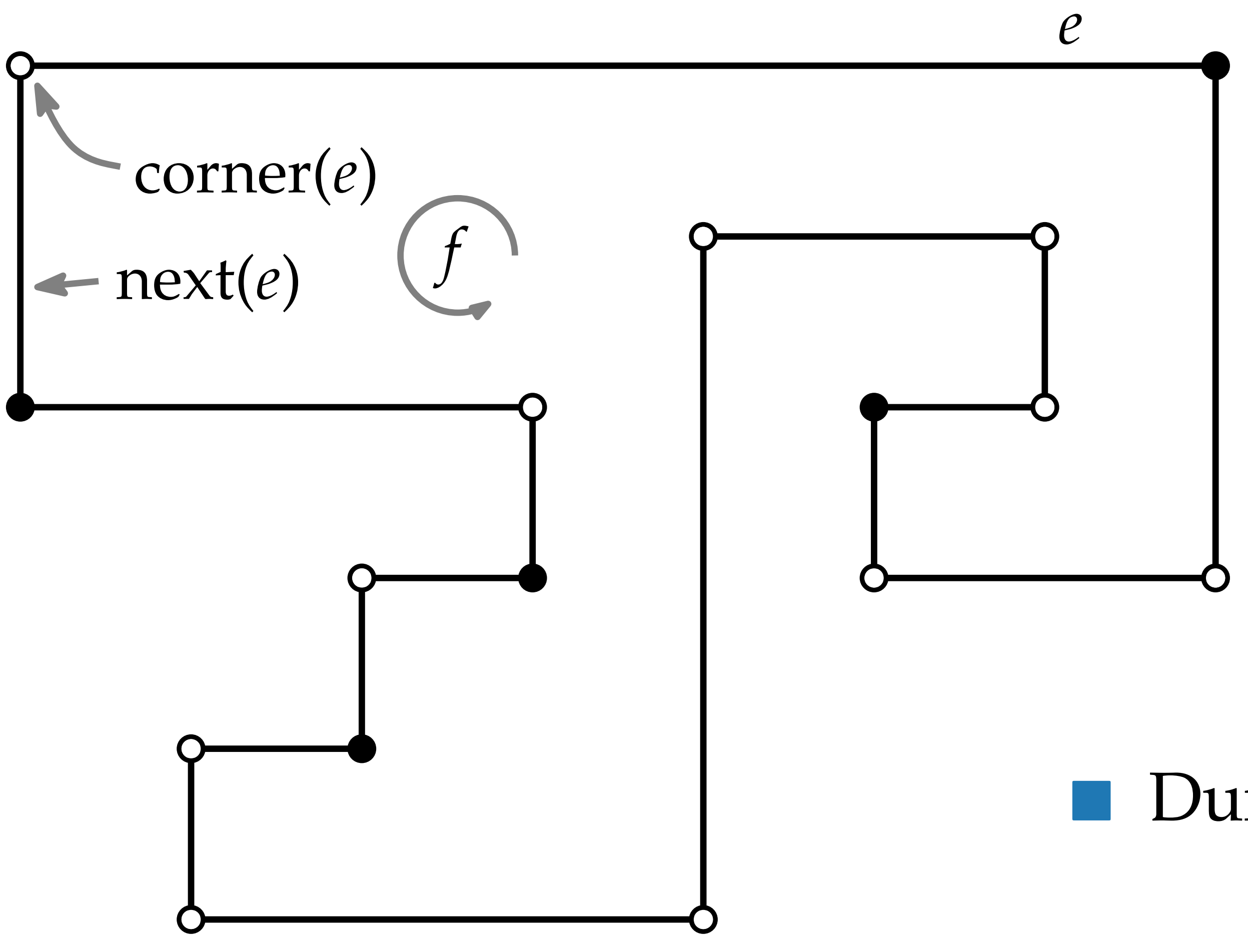
- Dummy vertices for bends

Refinement of (G, H) – Inner Face



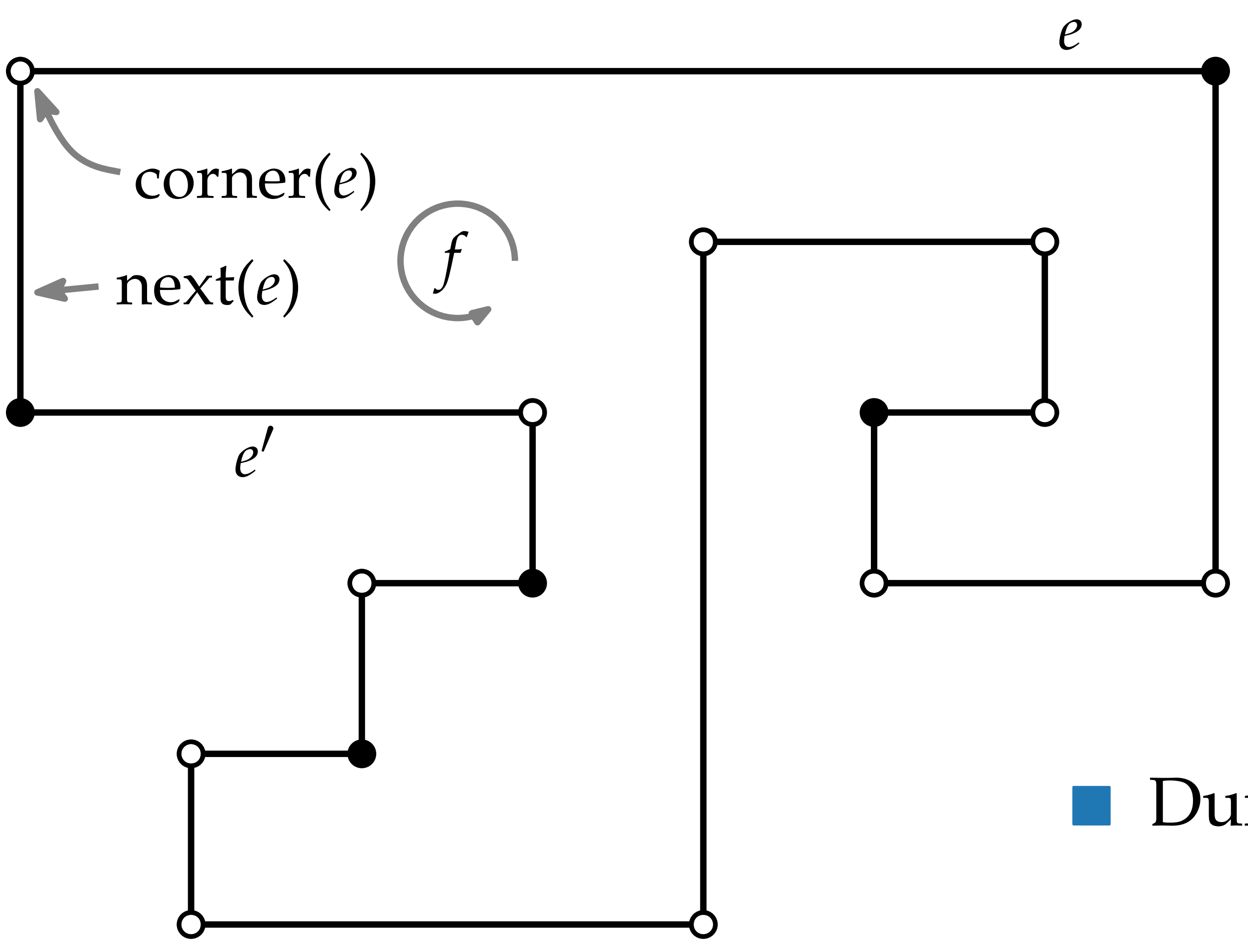
- Dummy vertices for bends

Refinement of (G, H) – Inner Face



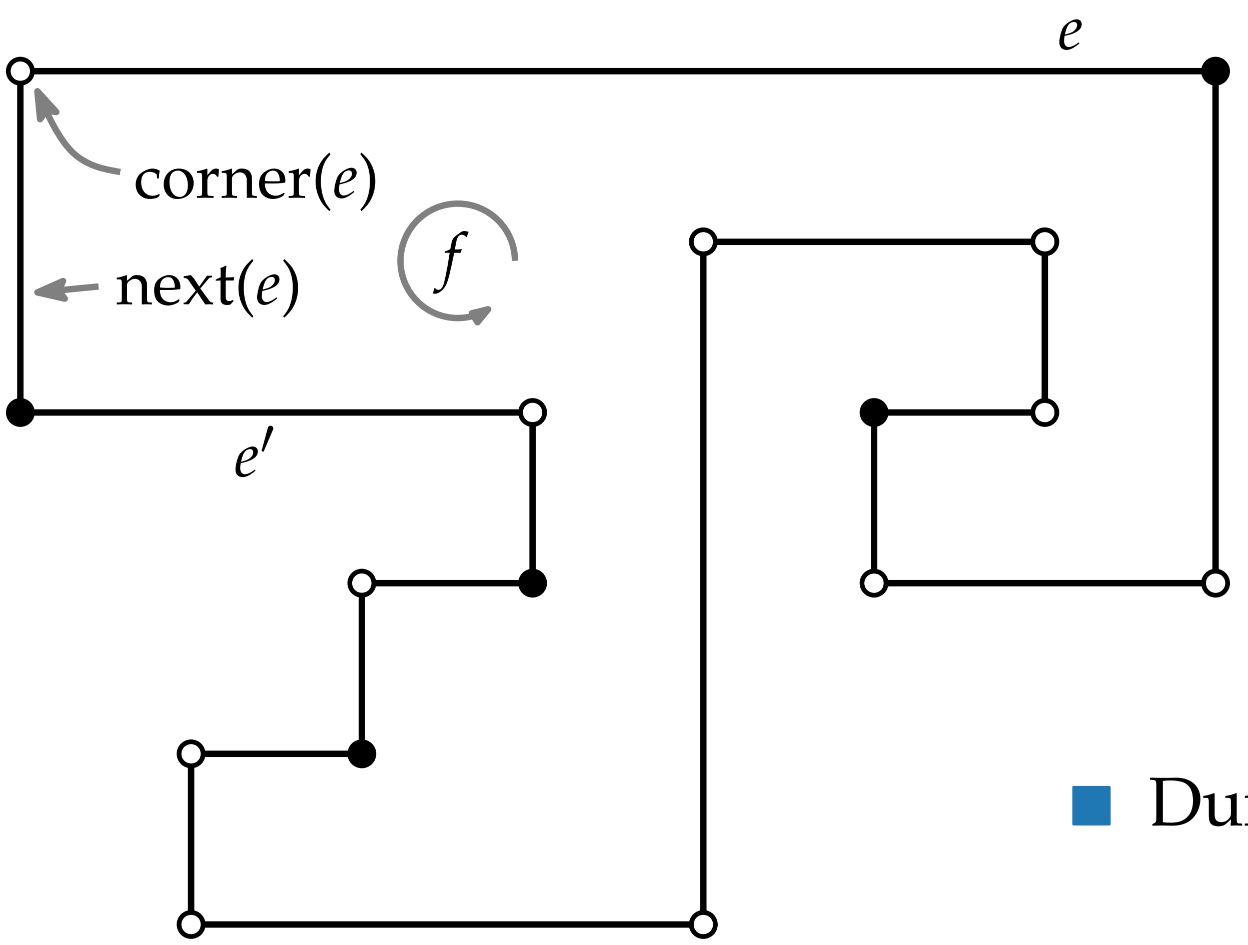
- Dummy vertices for bends

Refinement of (G, H) – Inner Face



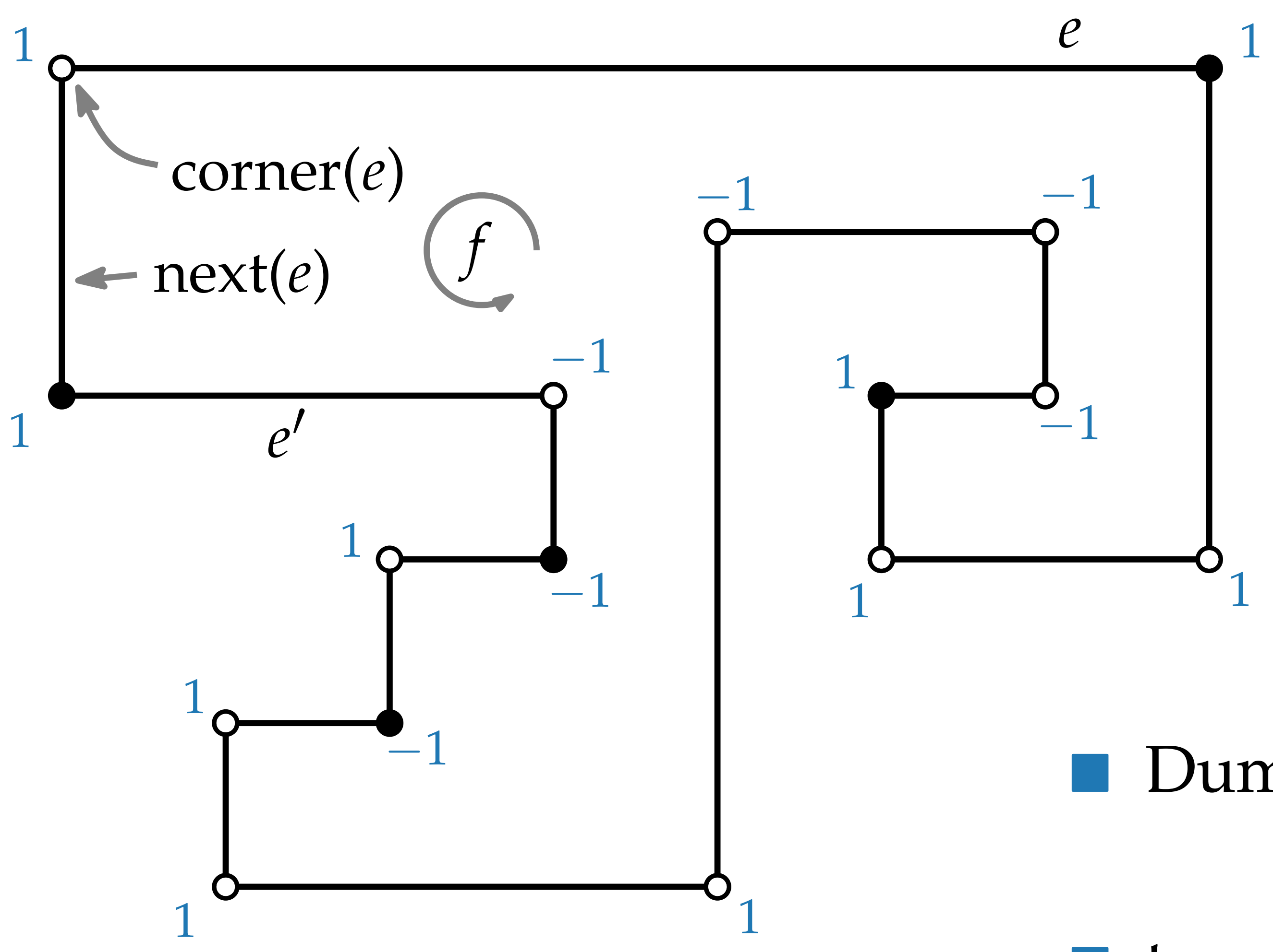
- Dummy vertices for bends

Refinement of (G, H) – Inner Face



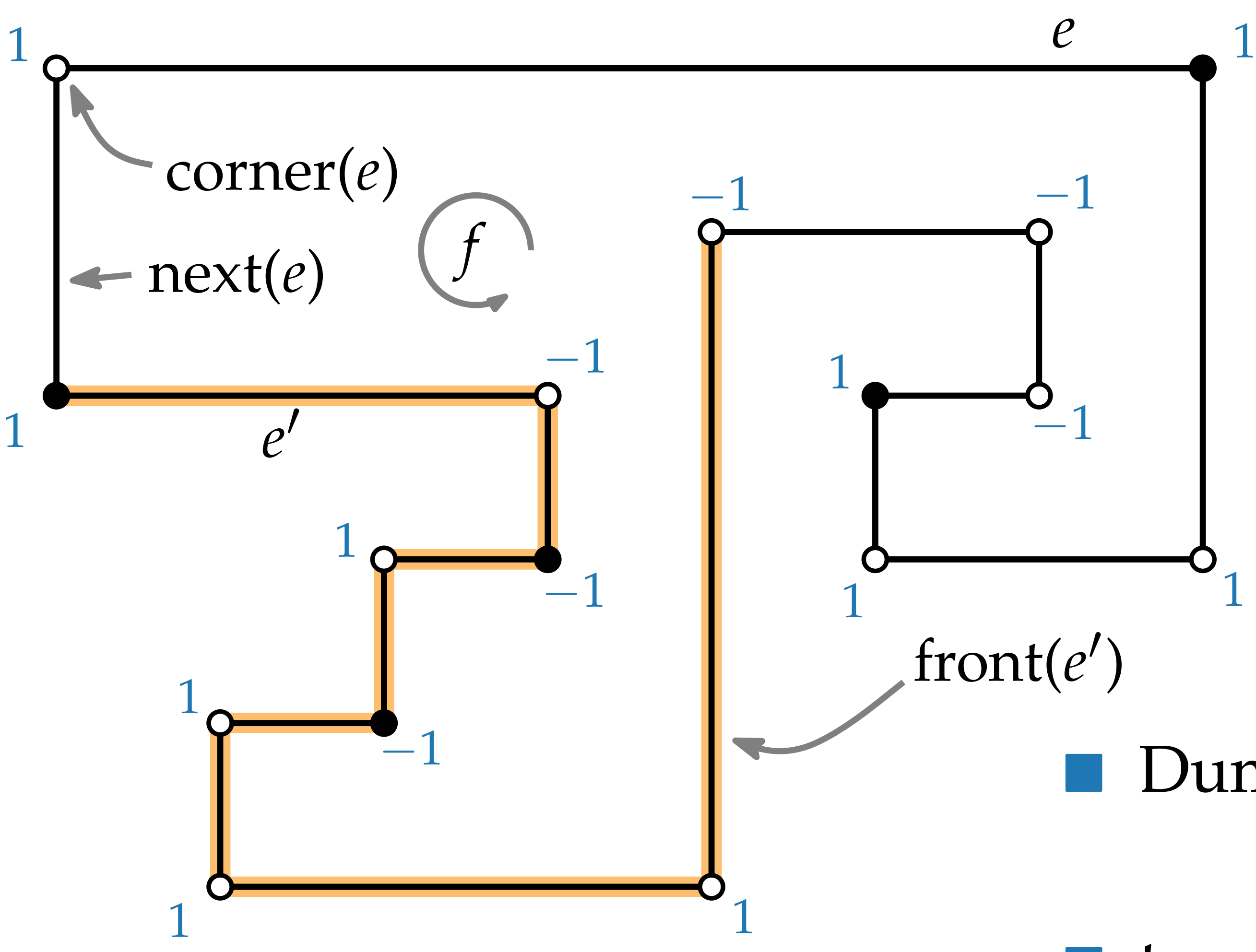
- Dummy vertices for bends
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of (G, H) – Inner Face



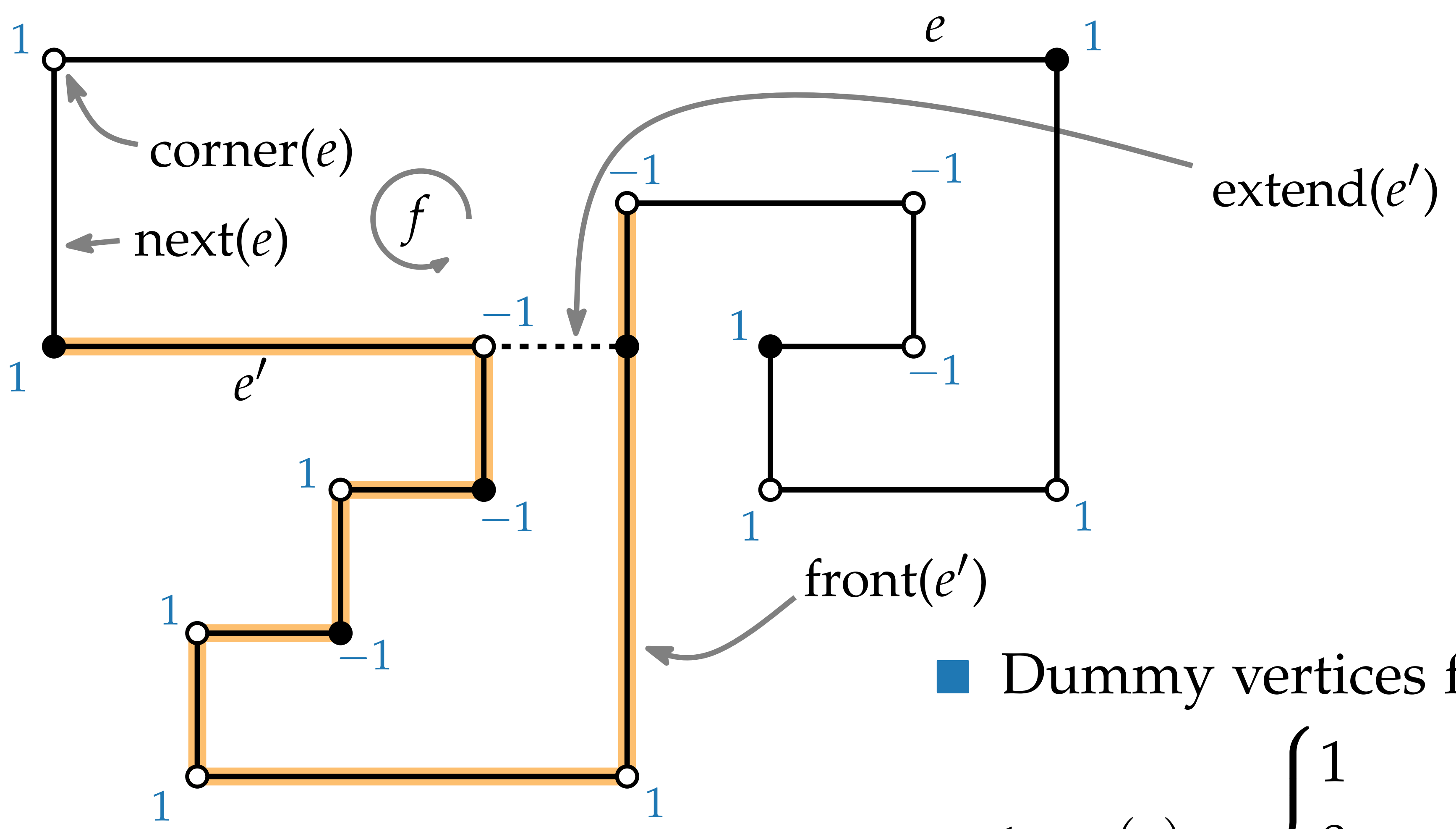
- Dummy vertices for bends
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of (G, H) – Inner Face



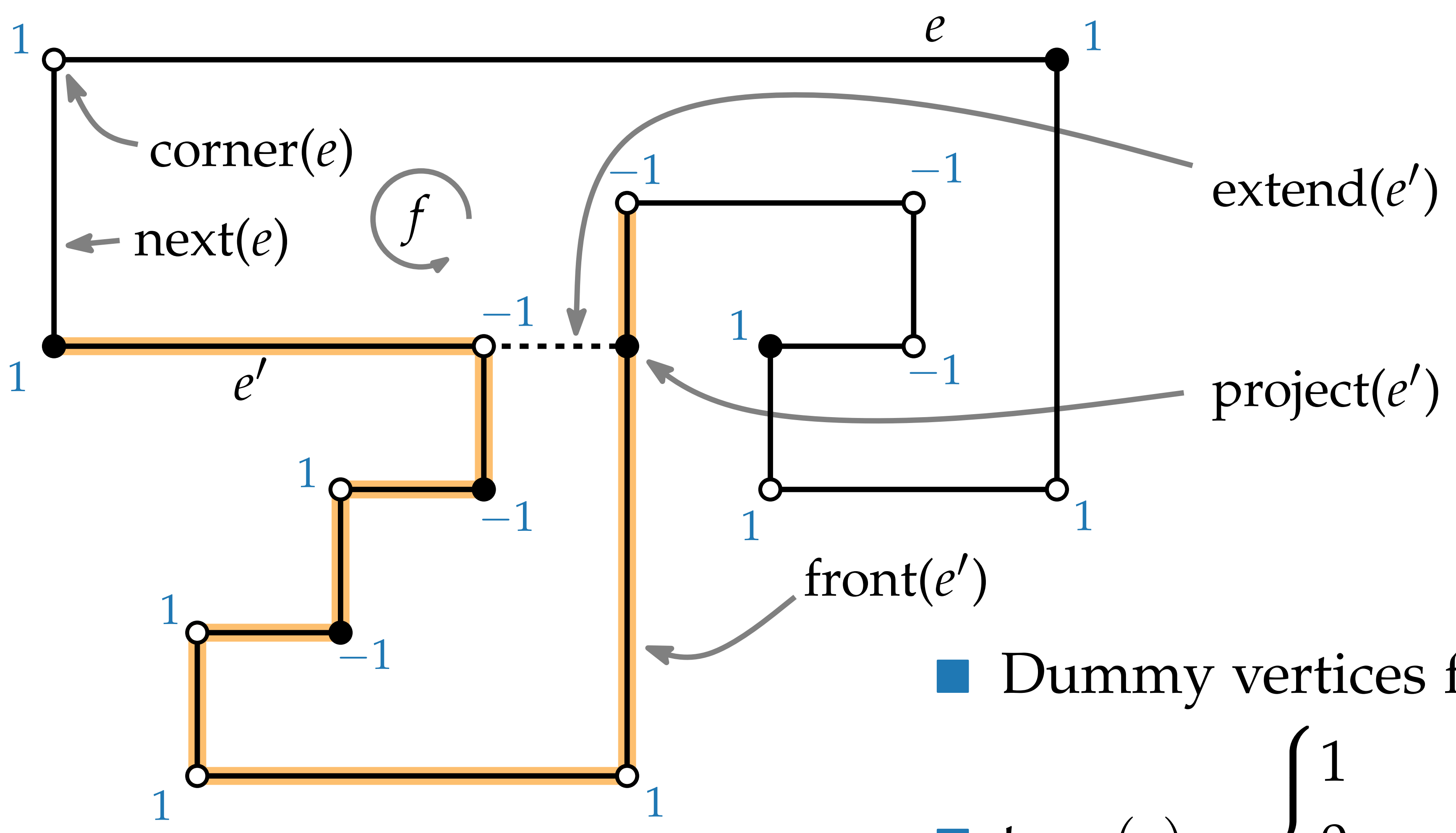
- Dummy vertices for bends
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of (G, H) – Inner Face



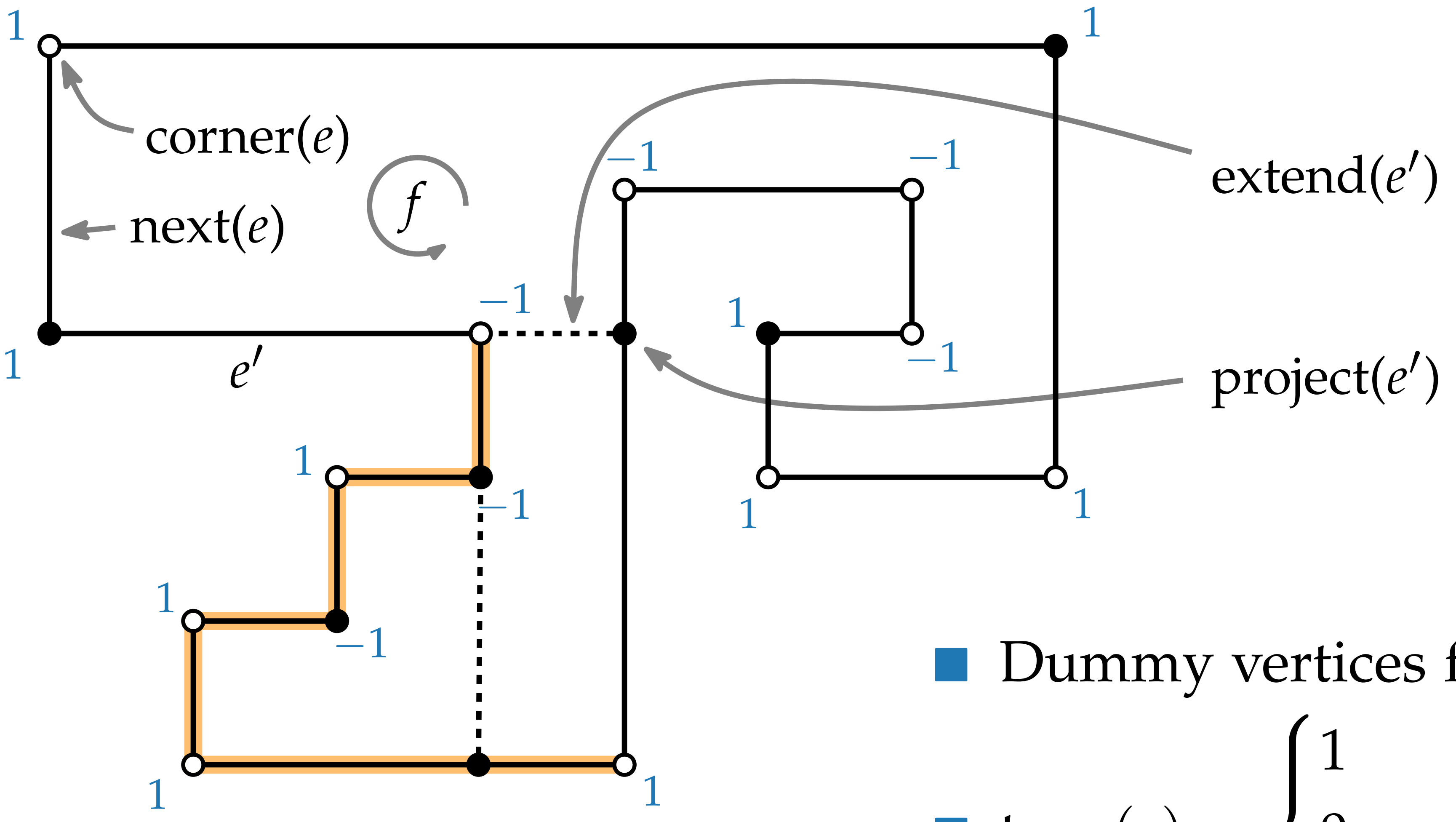
- Dummy vertices for bends
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of (G, H) – Inner Face



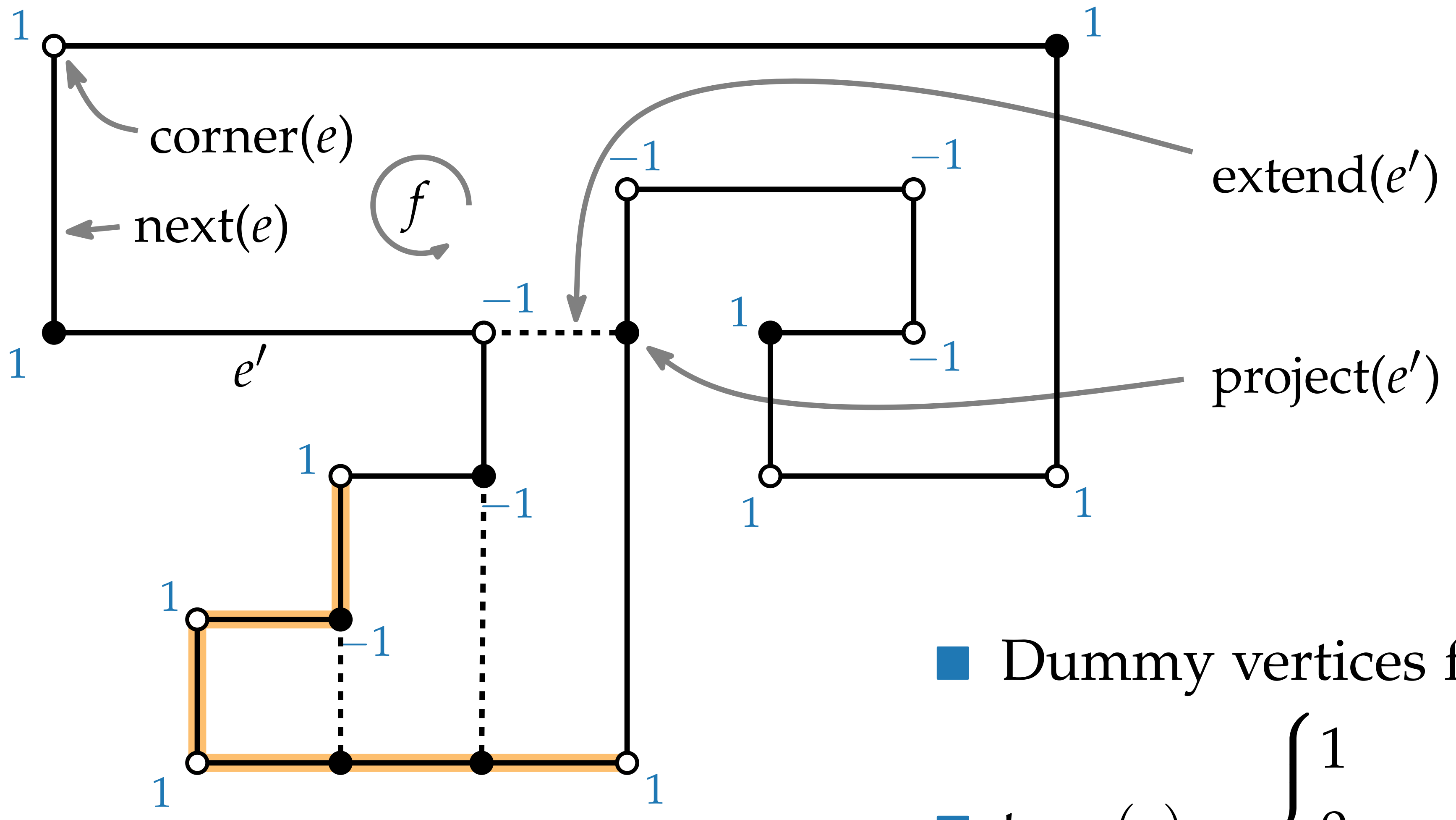
- Dummy vertices for bends
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of (G, H) – Inner Face



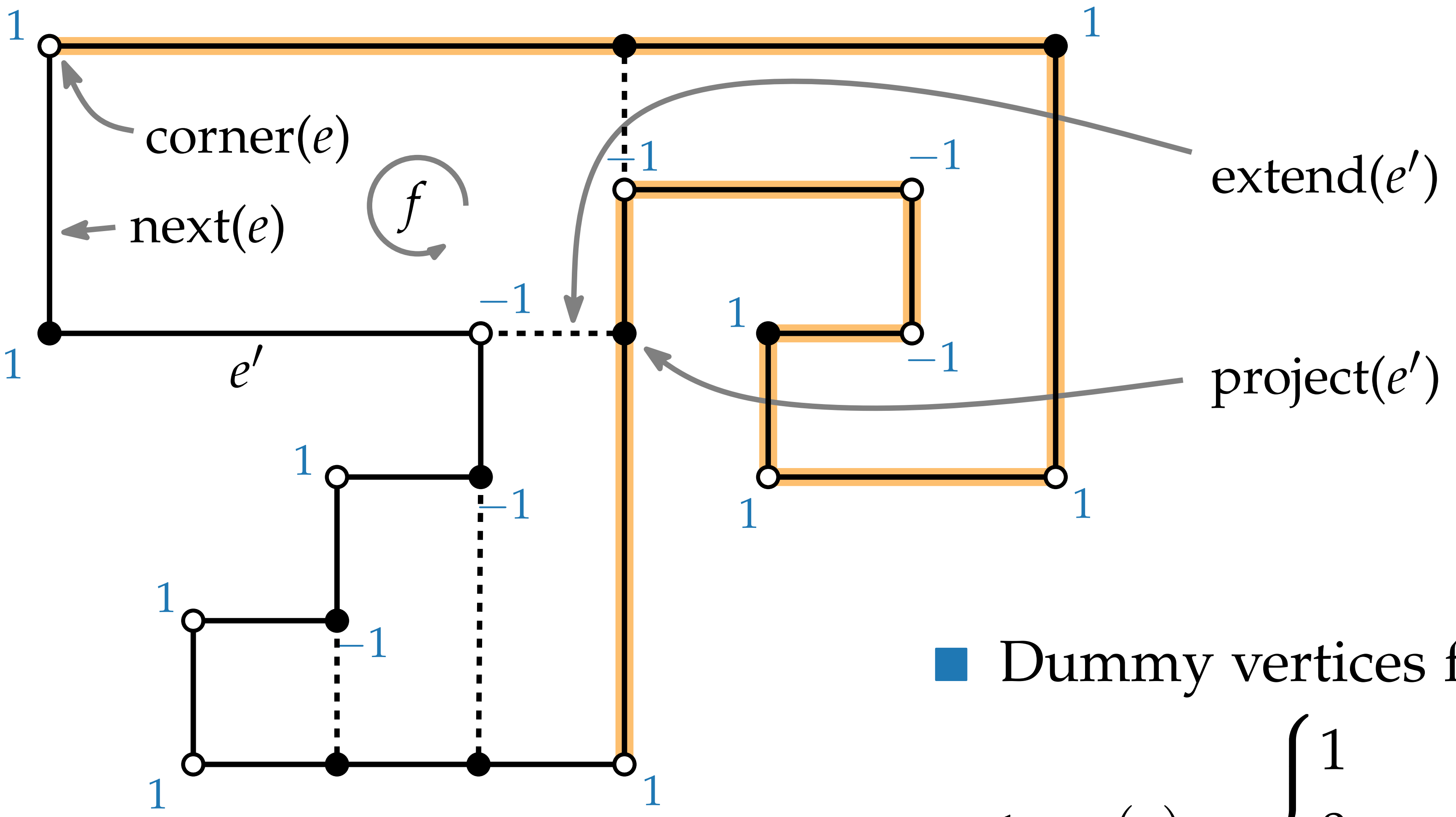
- Dummy vertices for bends
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of (G, H) – Inner Face



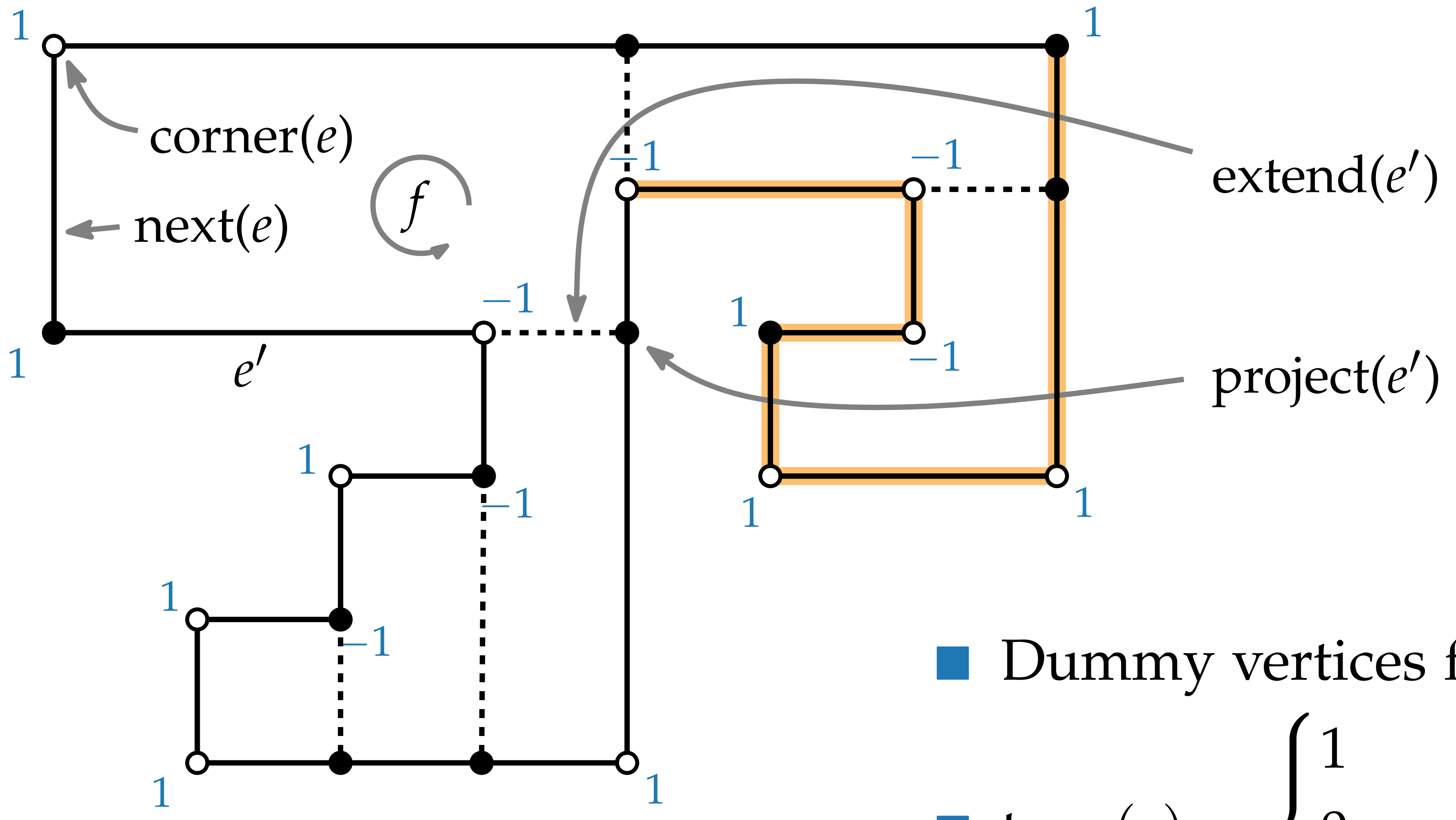
- Dummy vertices for bends
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of (G, H) – Inner Face



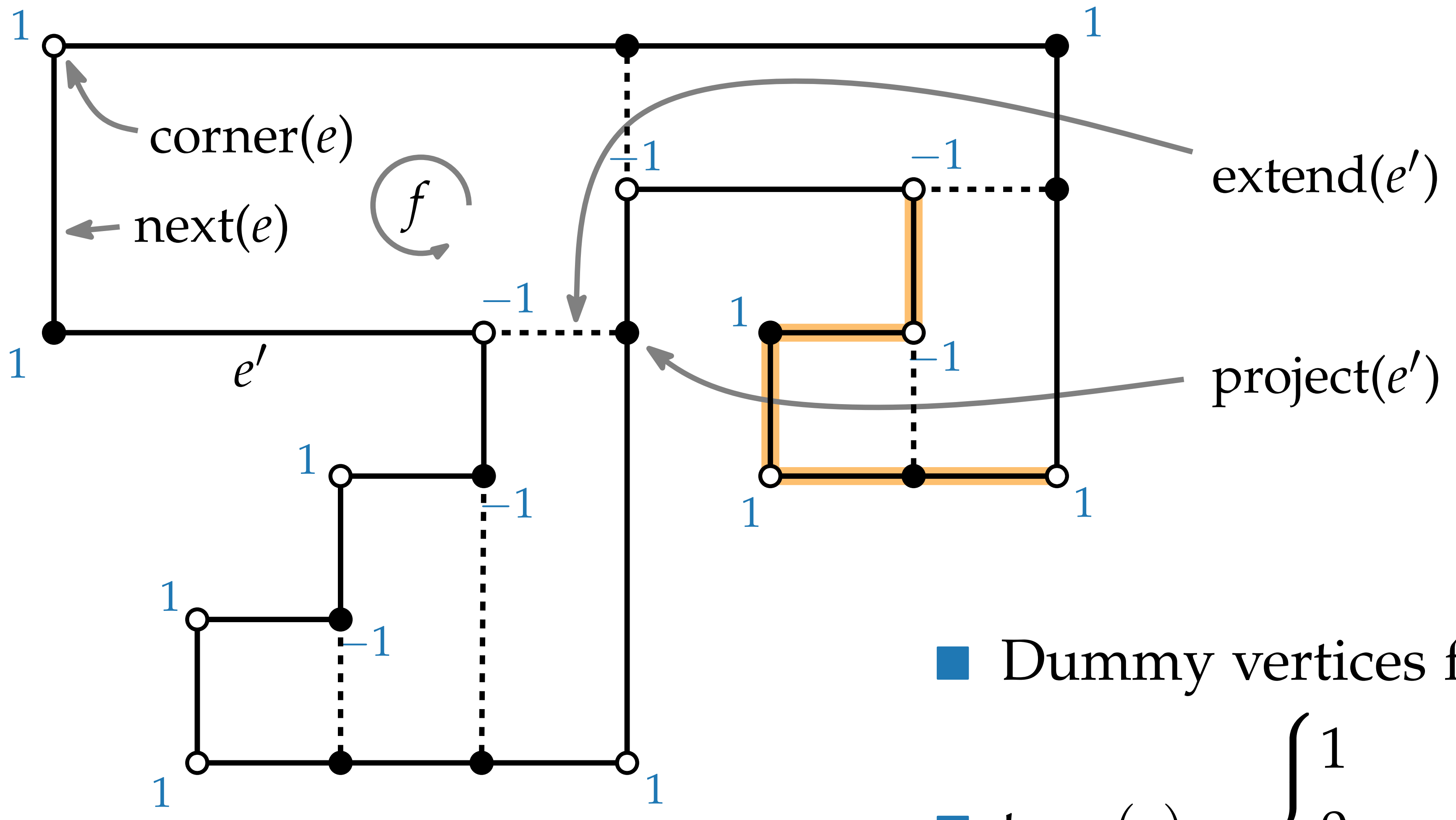
- Dummy vertices for bends
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of (G, H) – Inner Face



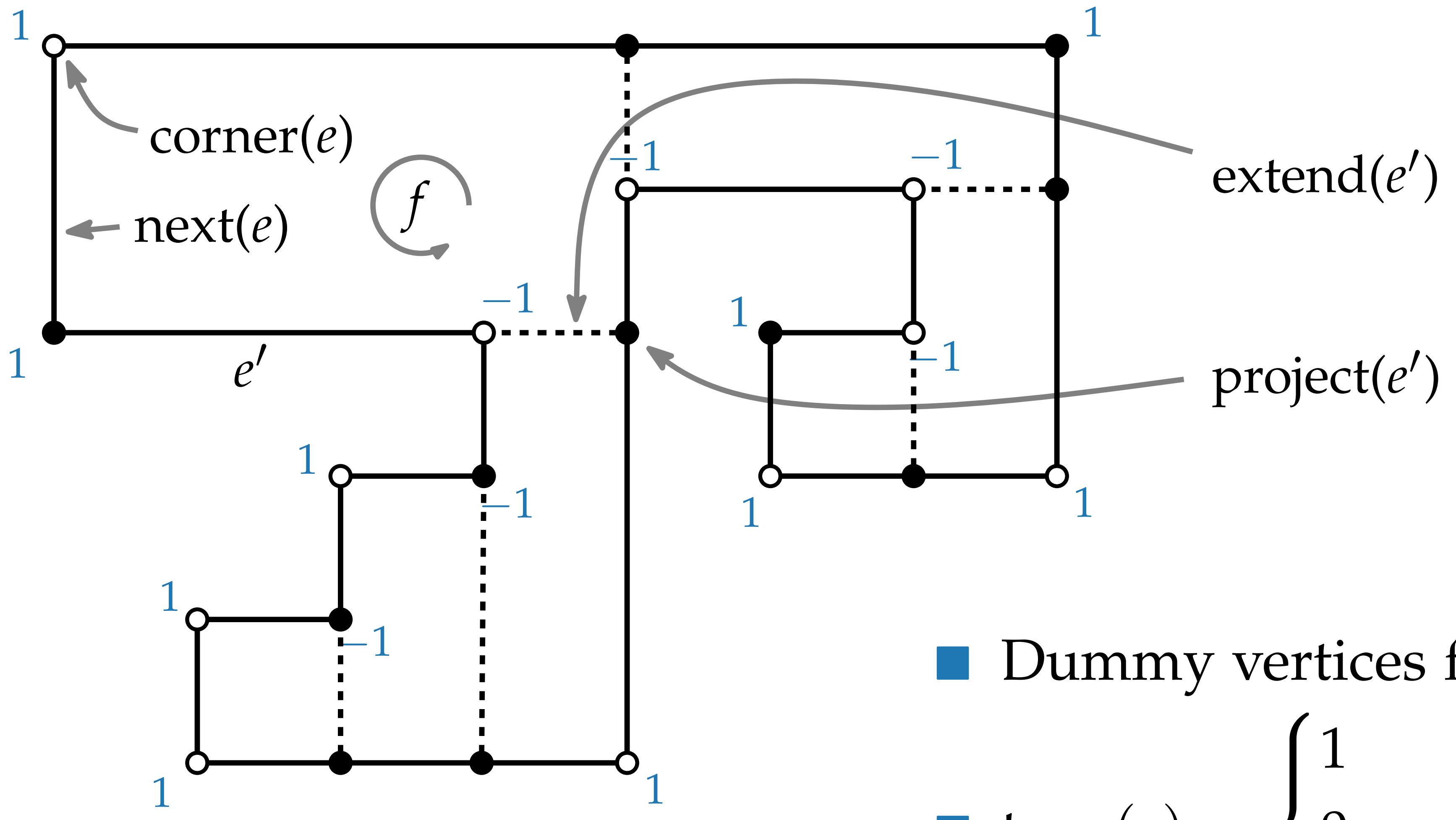
- Dummy vertices for bends
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of (G, H) – Inner Face



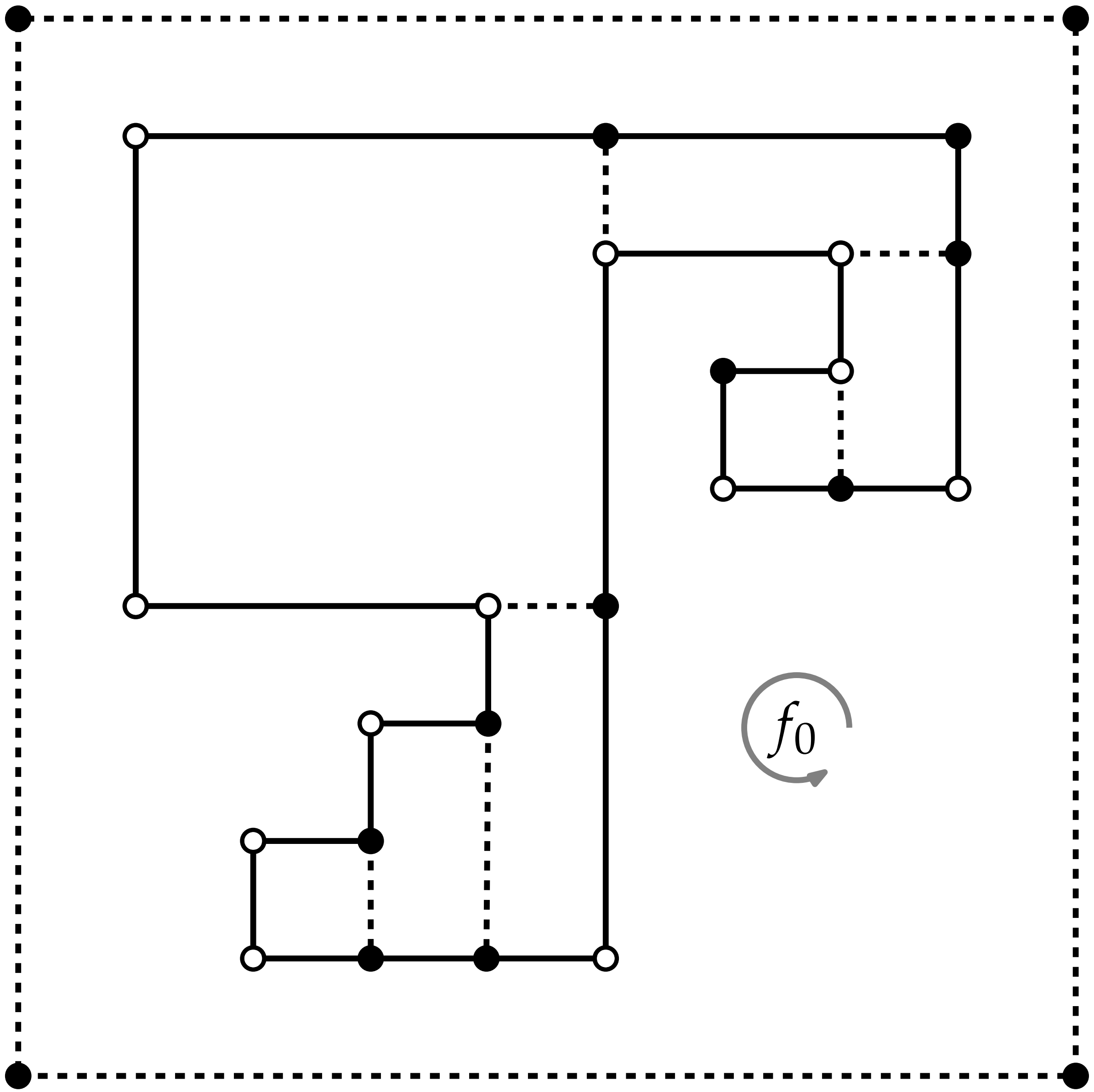
- Dummy vertices for bends
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

Refinement of (G, H) – Inner Face

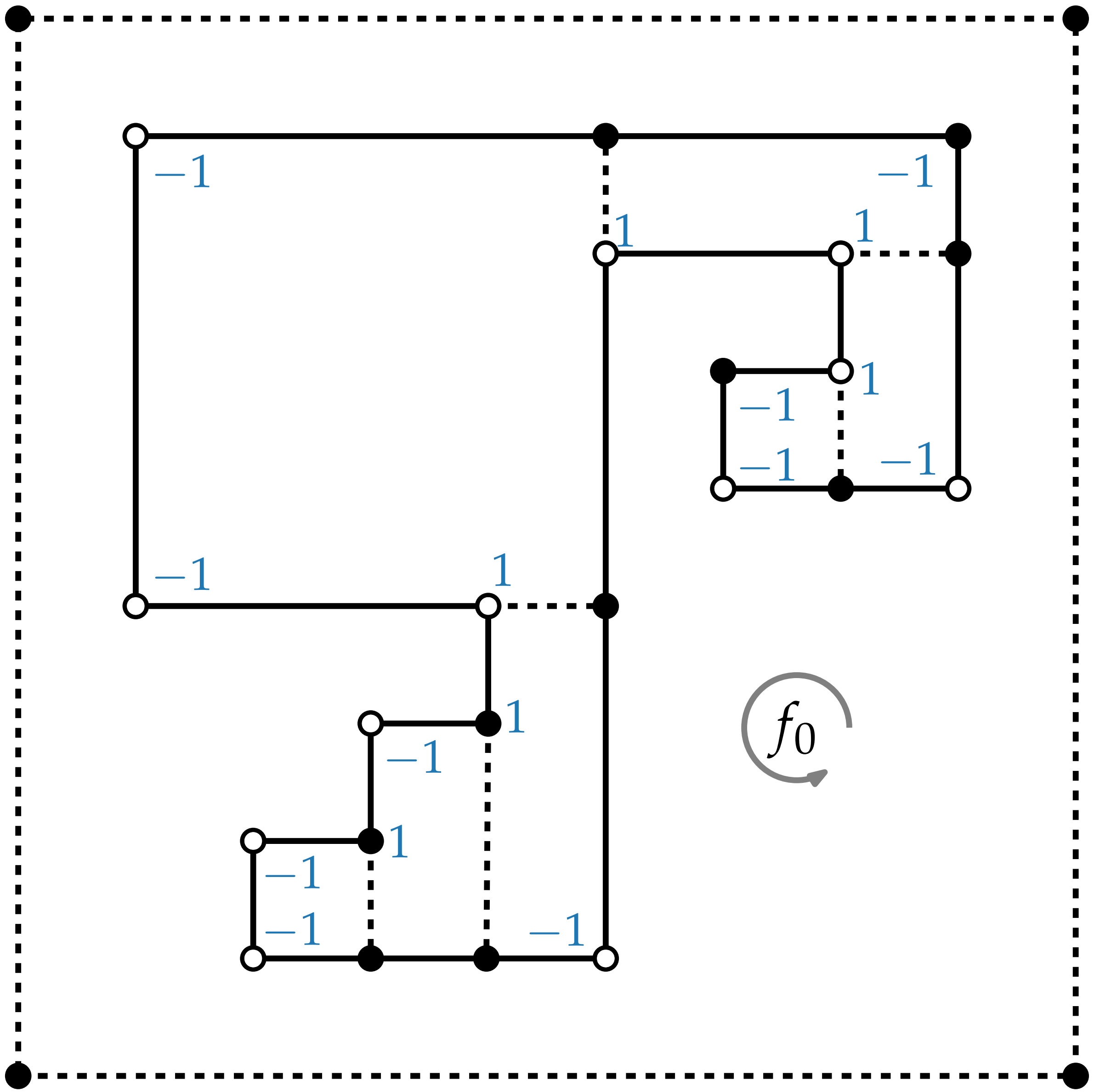


- Dummy vertices for bends
- $\text{turn}(e) = \begin{cases} 1 & \text{left turn} \\ 0 & \text{no turn} \\ -1 & \text{right turn} \end{cases}$

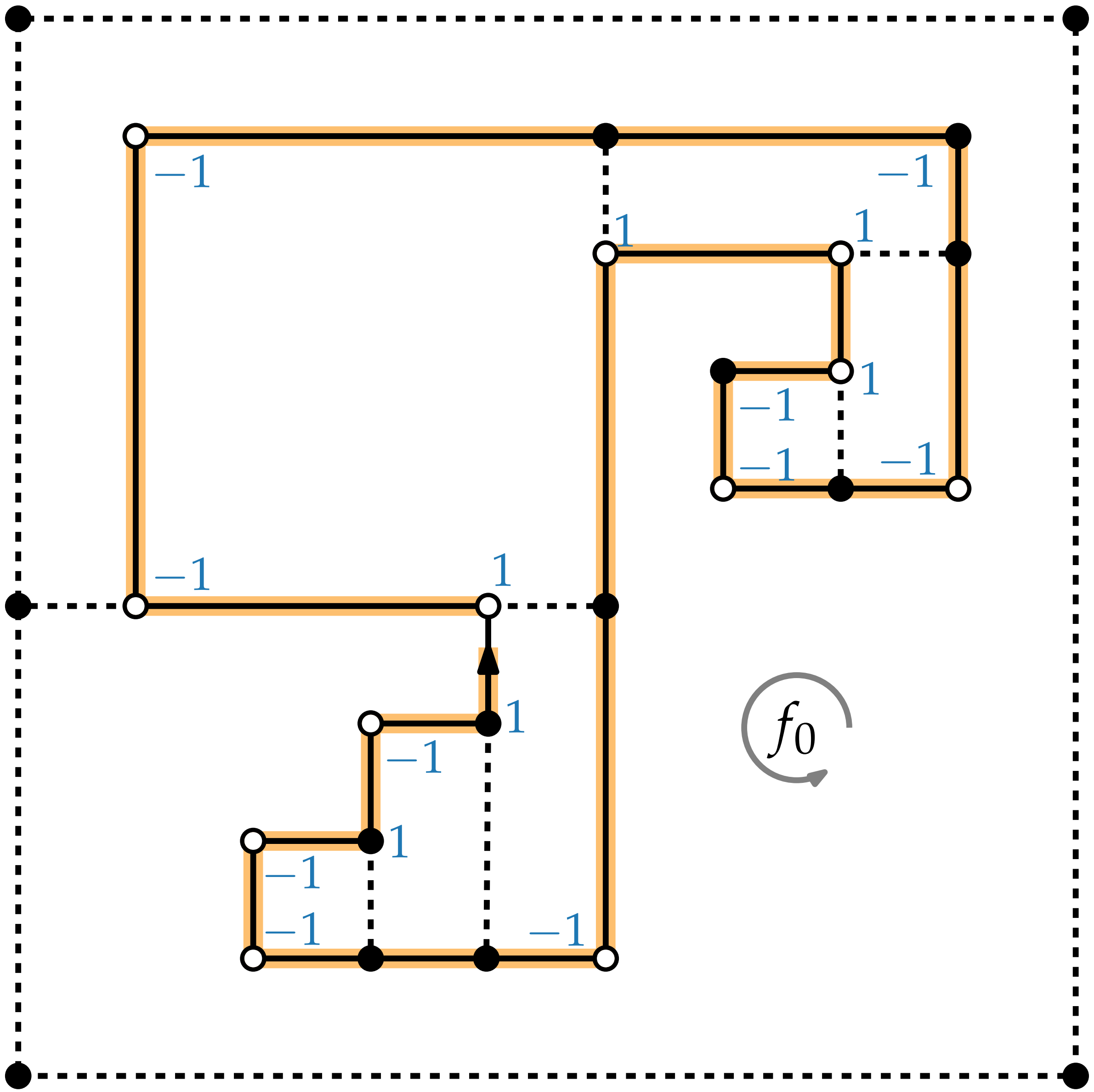
Refinement of (G, H) – Outer Face



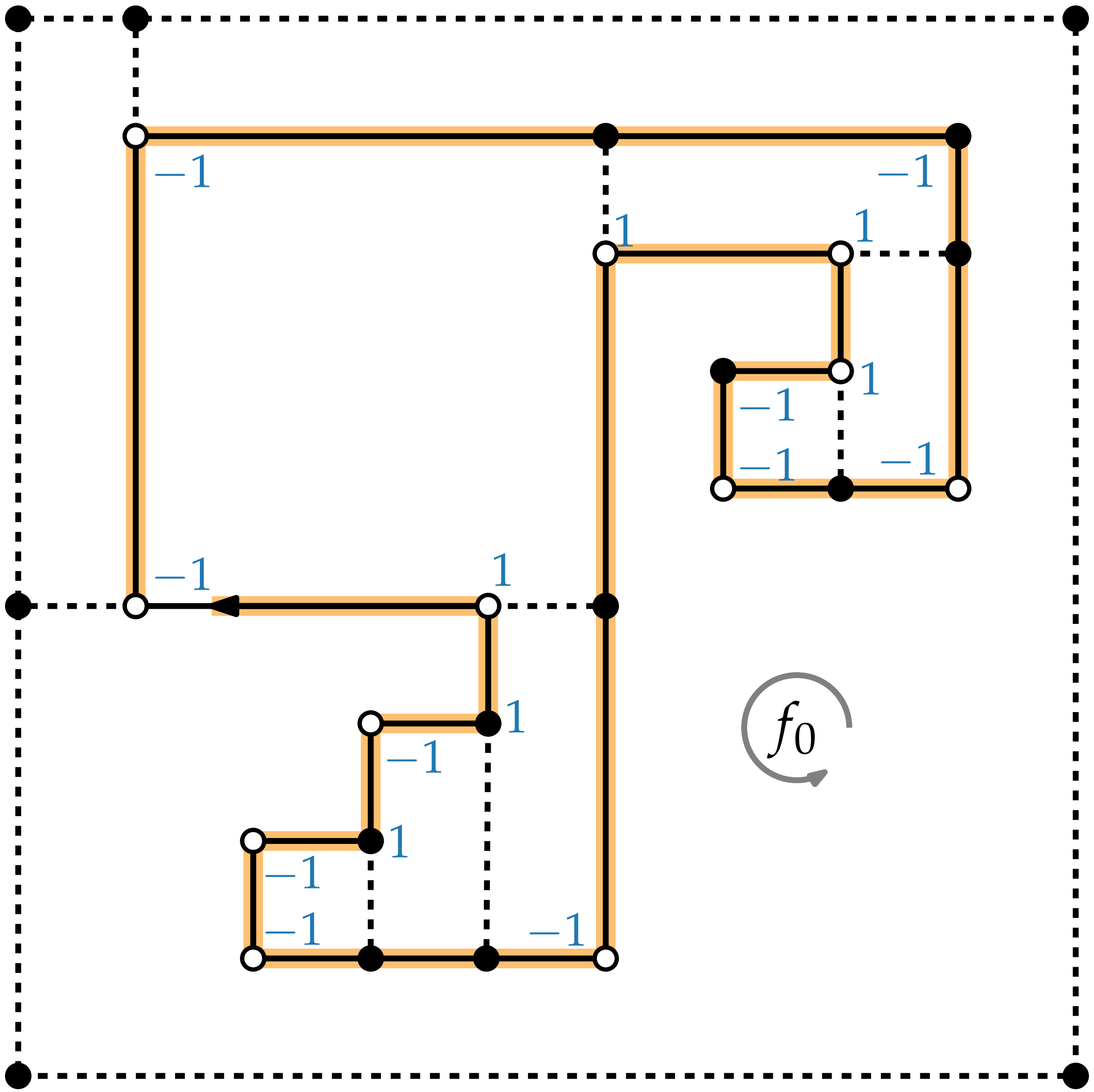
Refinement of (G, H) – Outer Face



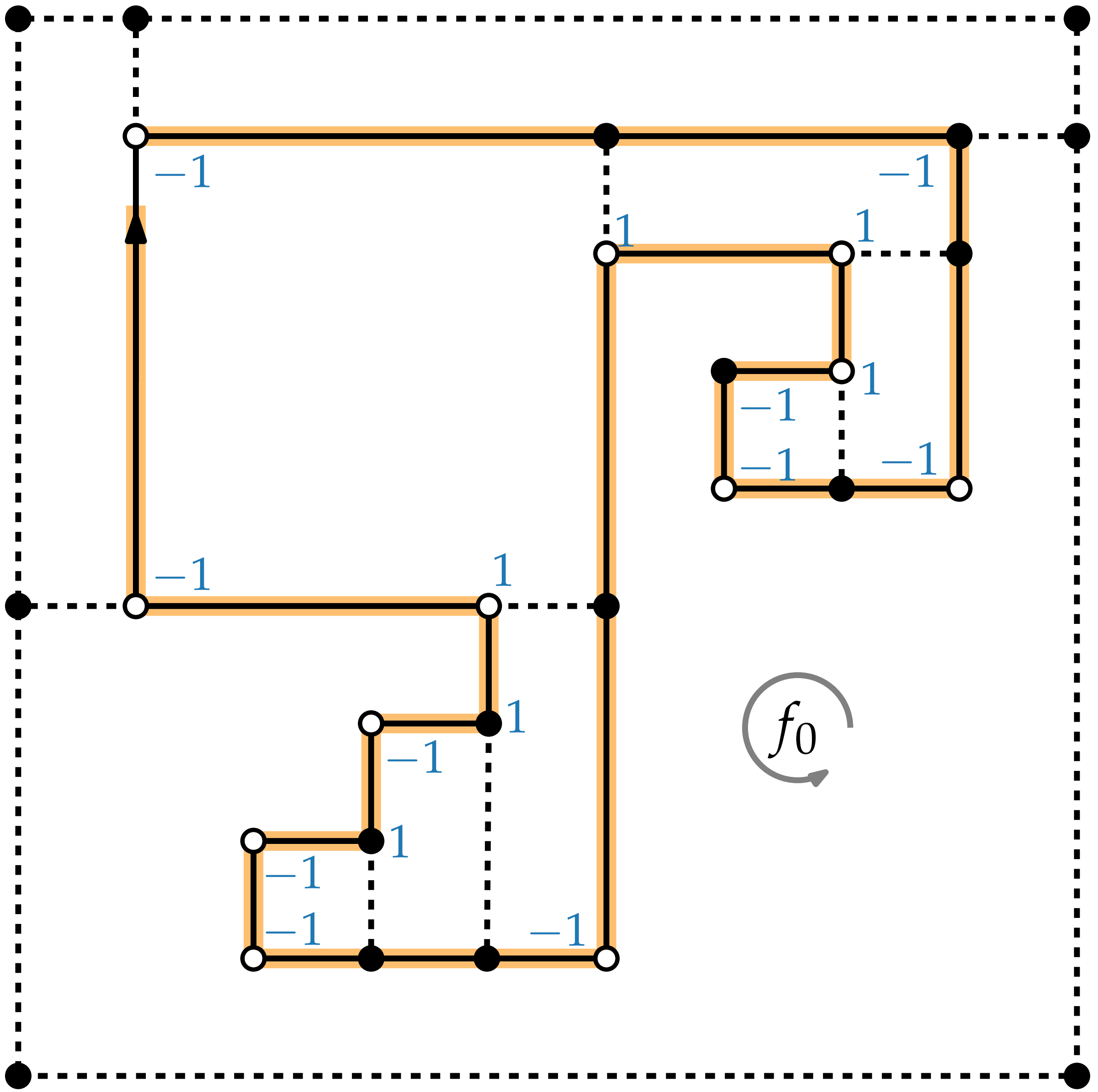
Refinement of (G, H) – Outer Face



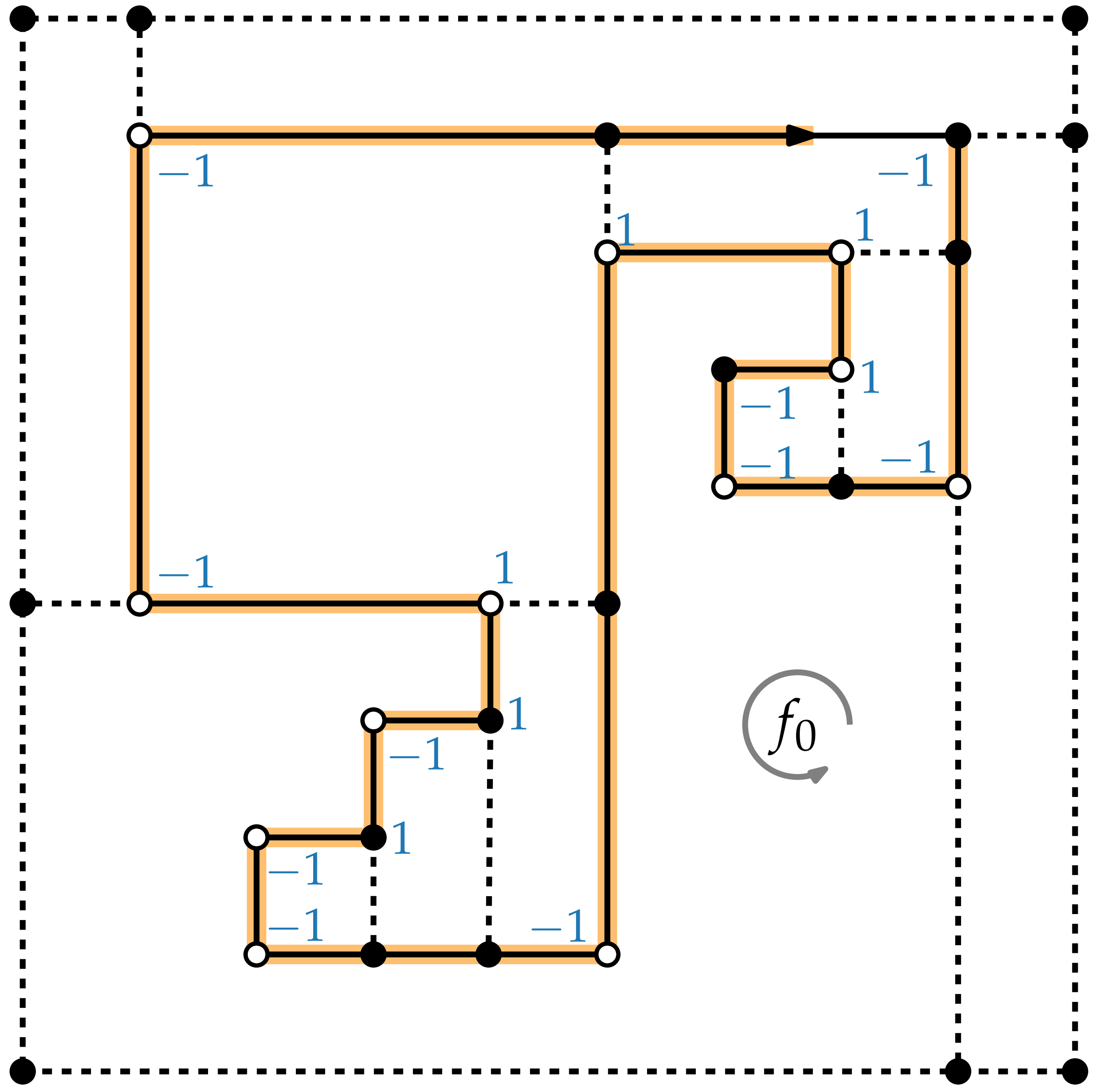
Refinement of (G, H) – Outer Face



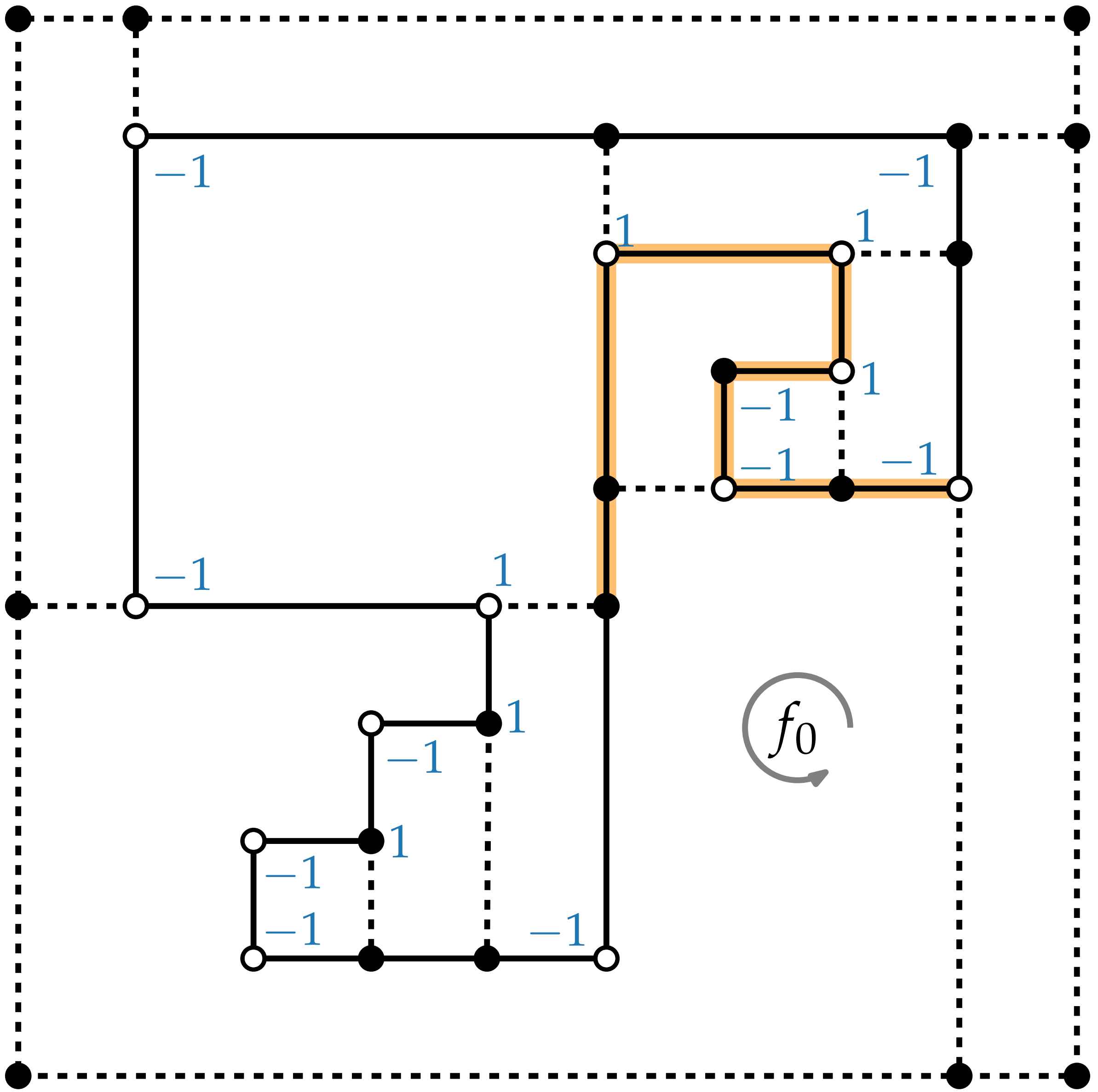
Refinement of (G, H) – Outer Face



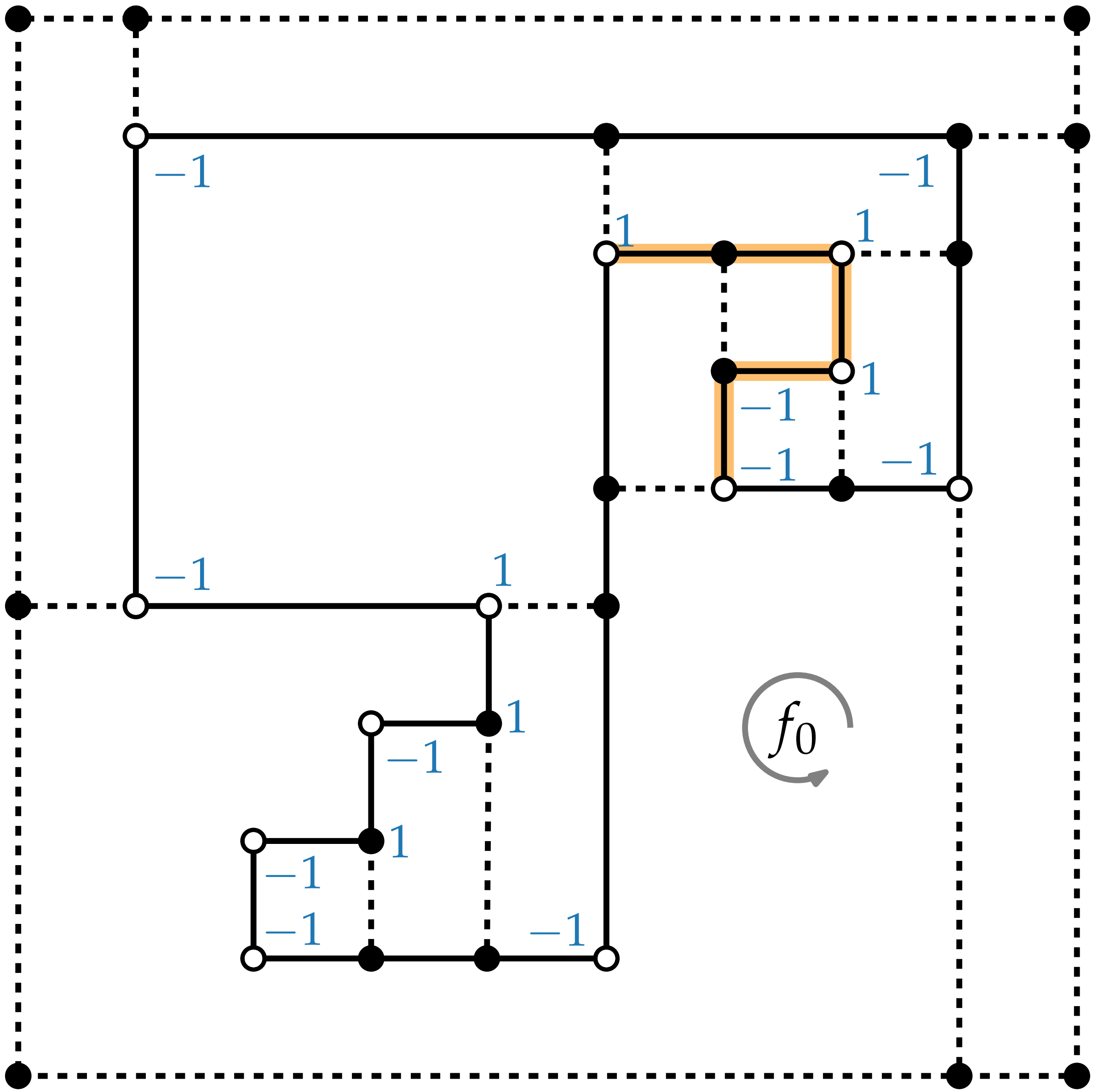
Refinement of (G, H) – Outer Face



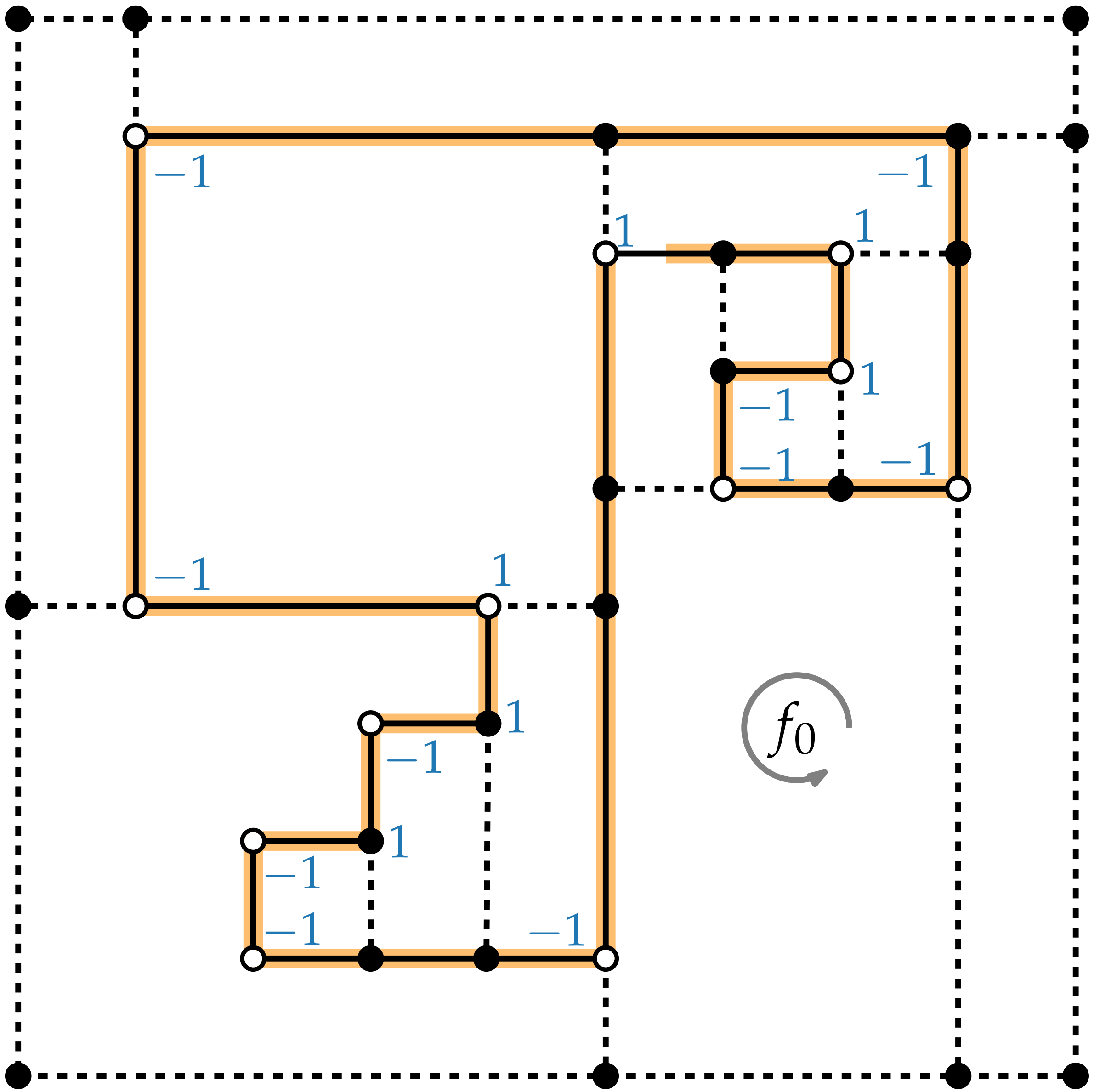
Refinement of (G, H) – Outer Face



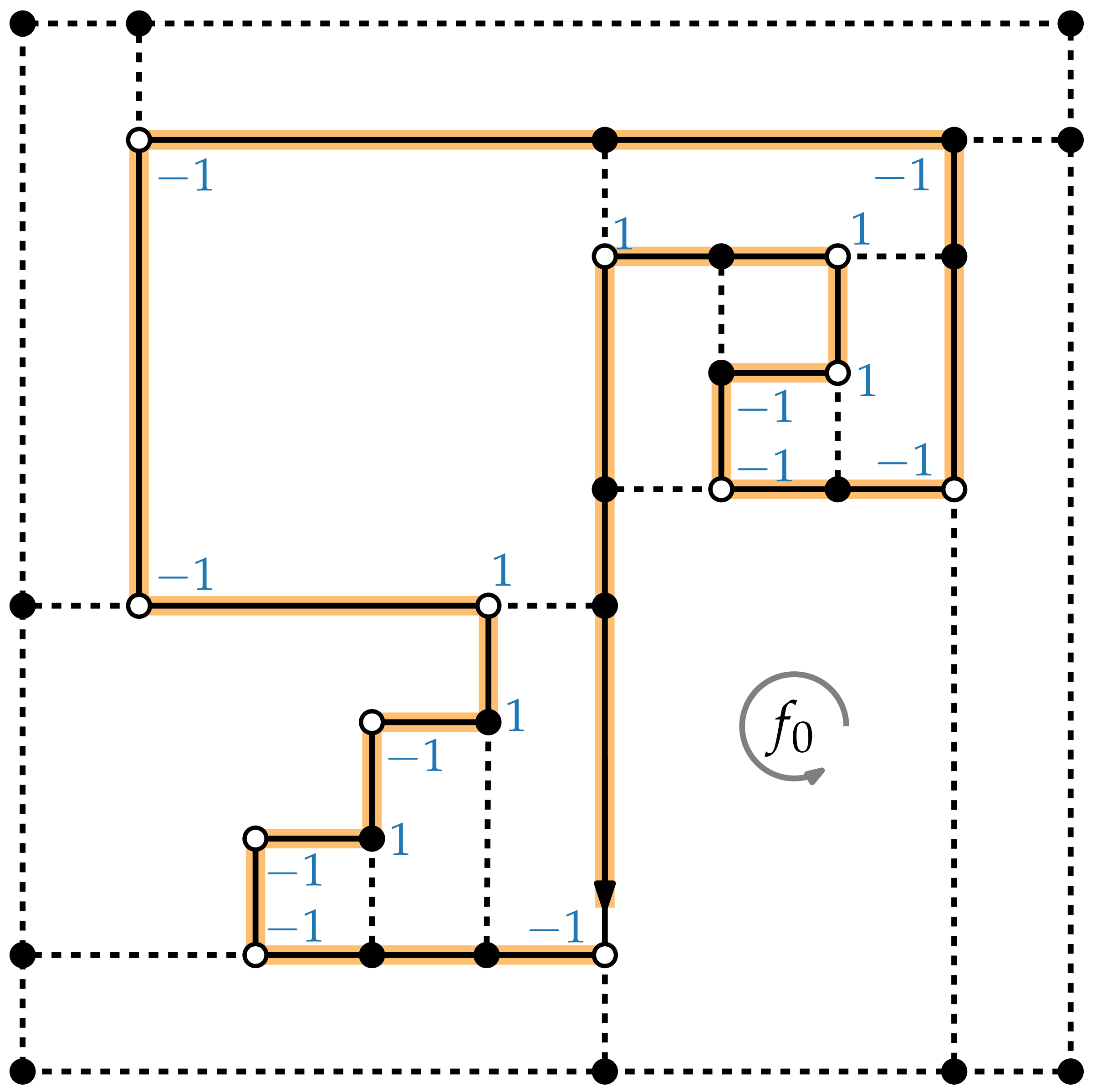
Refinement of (G, H) – Outer Face



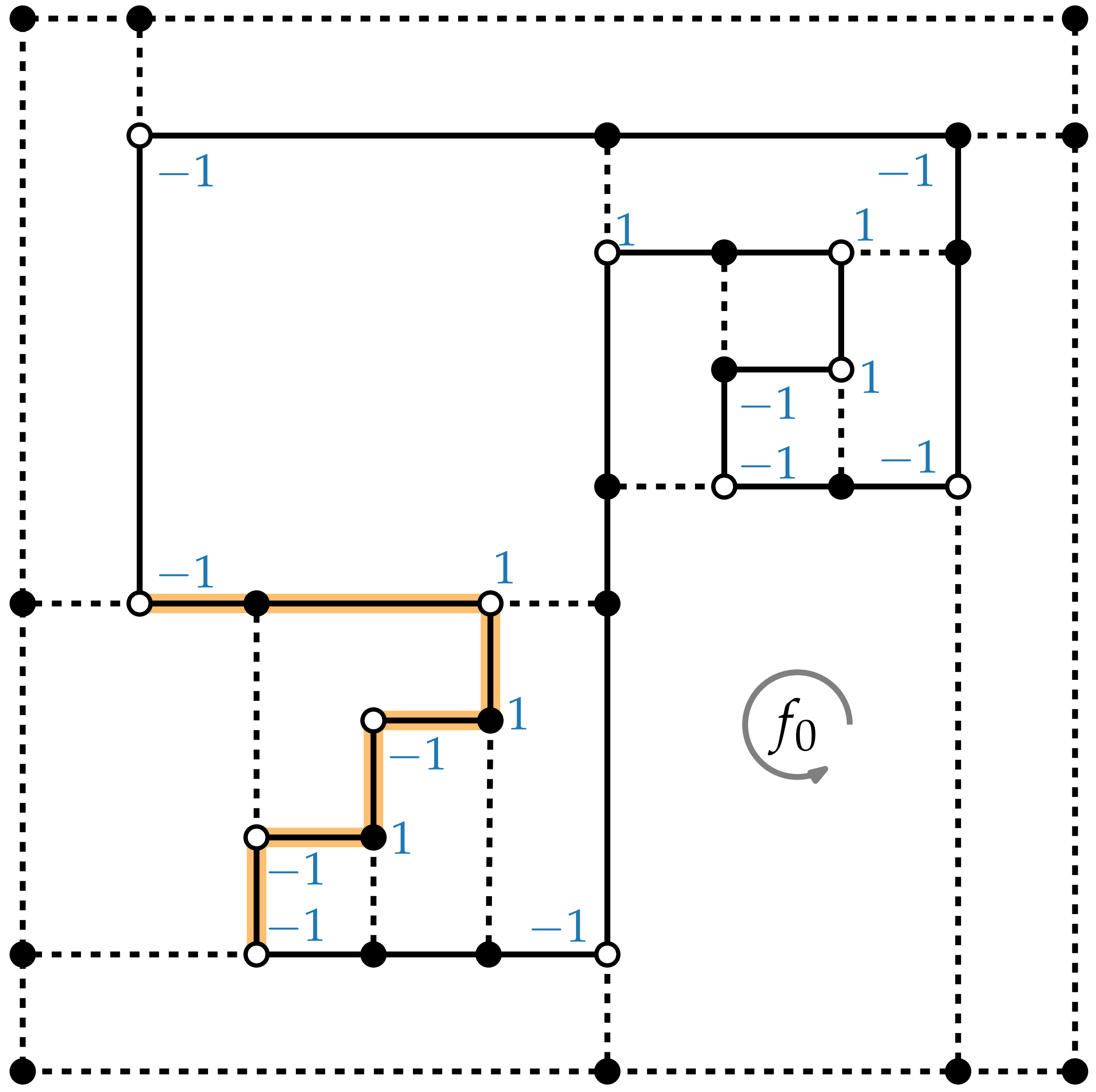
Refinement of (G, H) – Outer Face



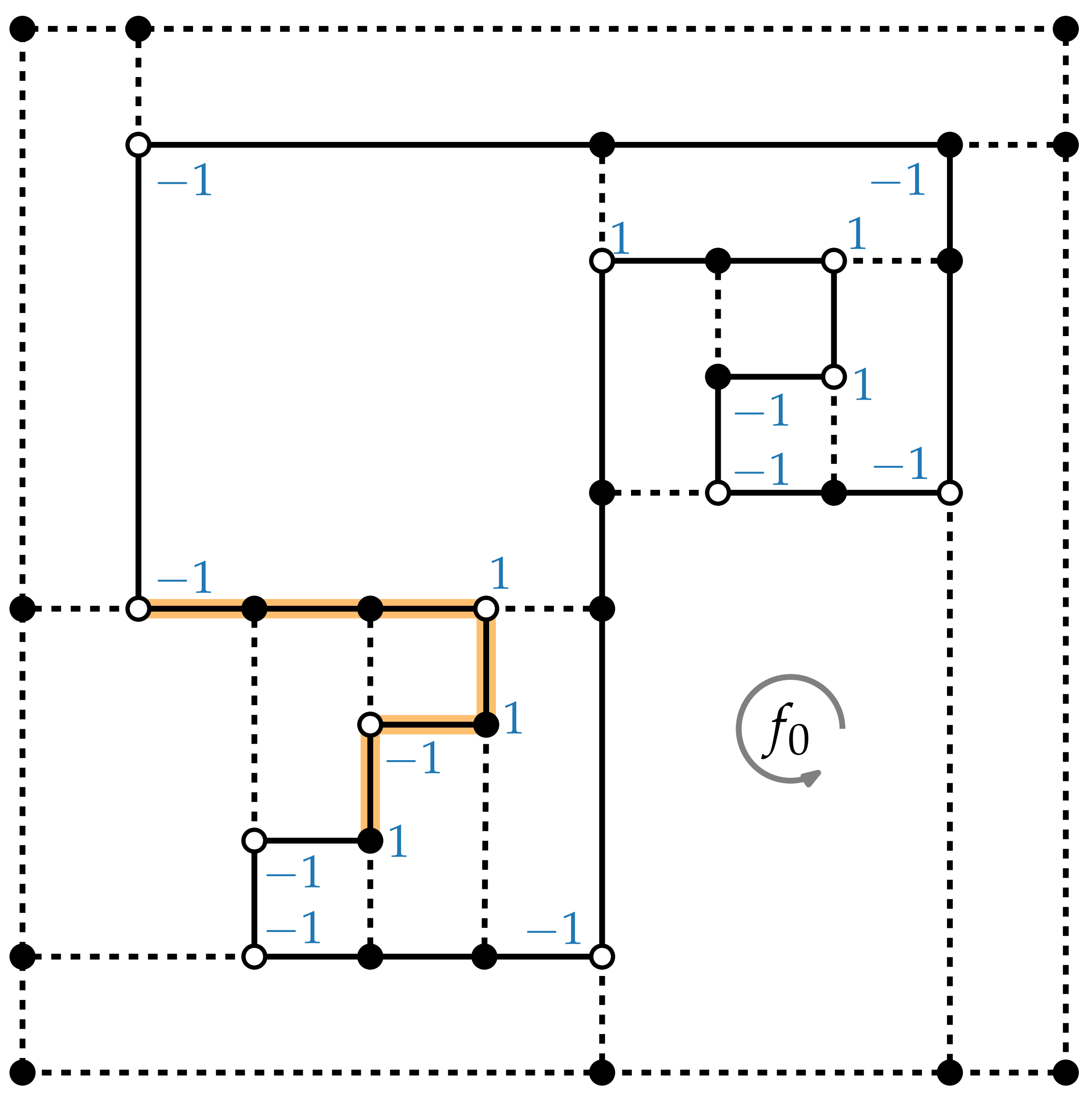
Refinement of (G, H) – Outer Face



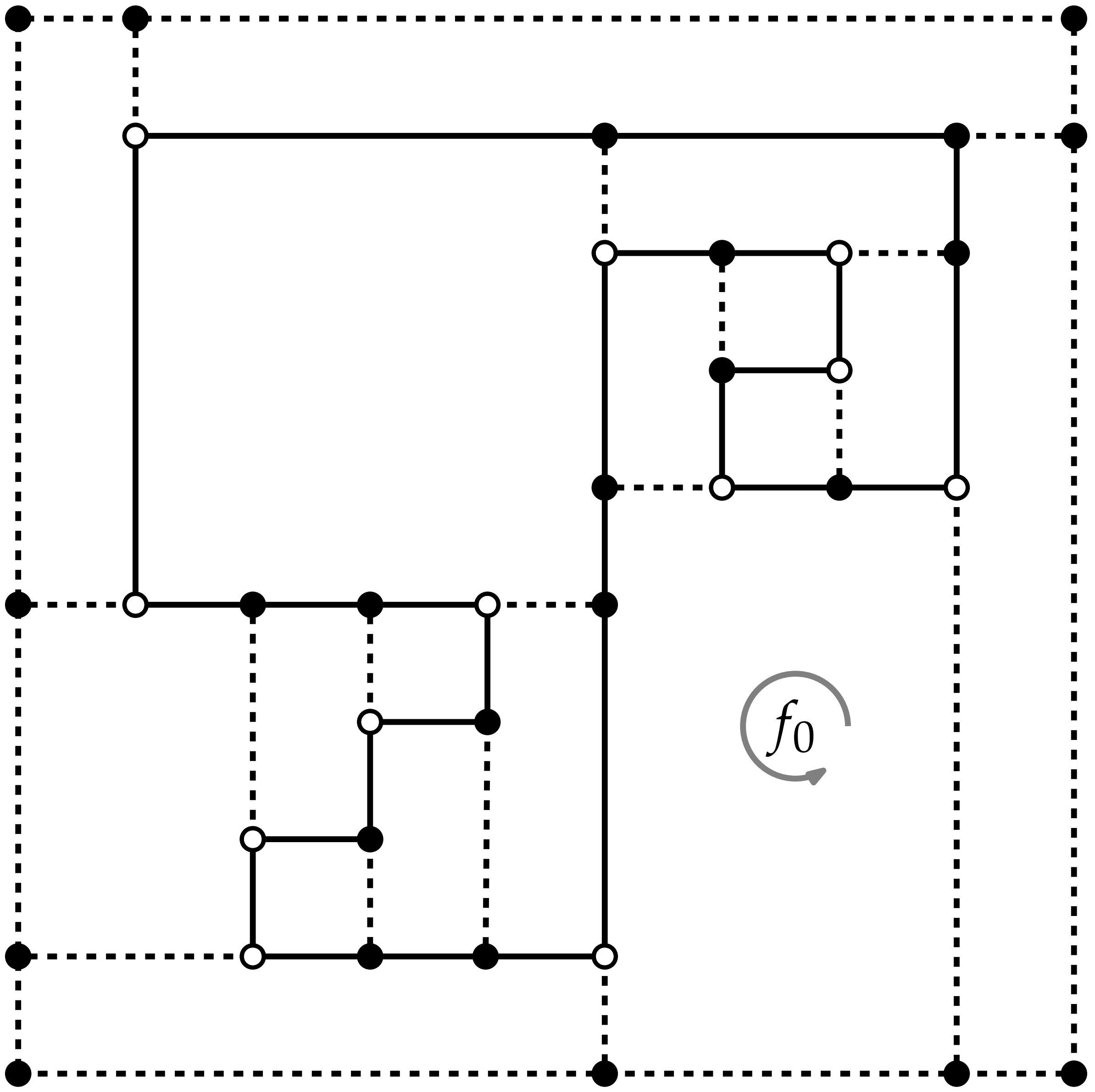
Refinement of (G, H) – Outer Face



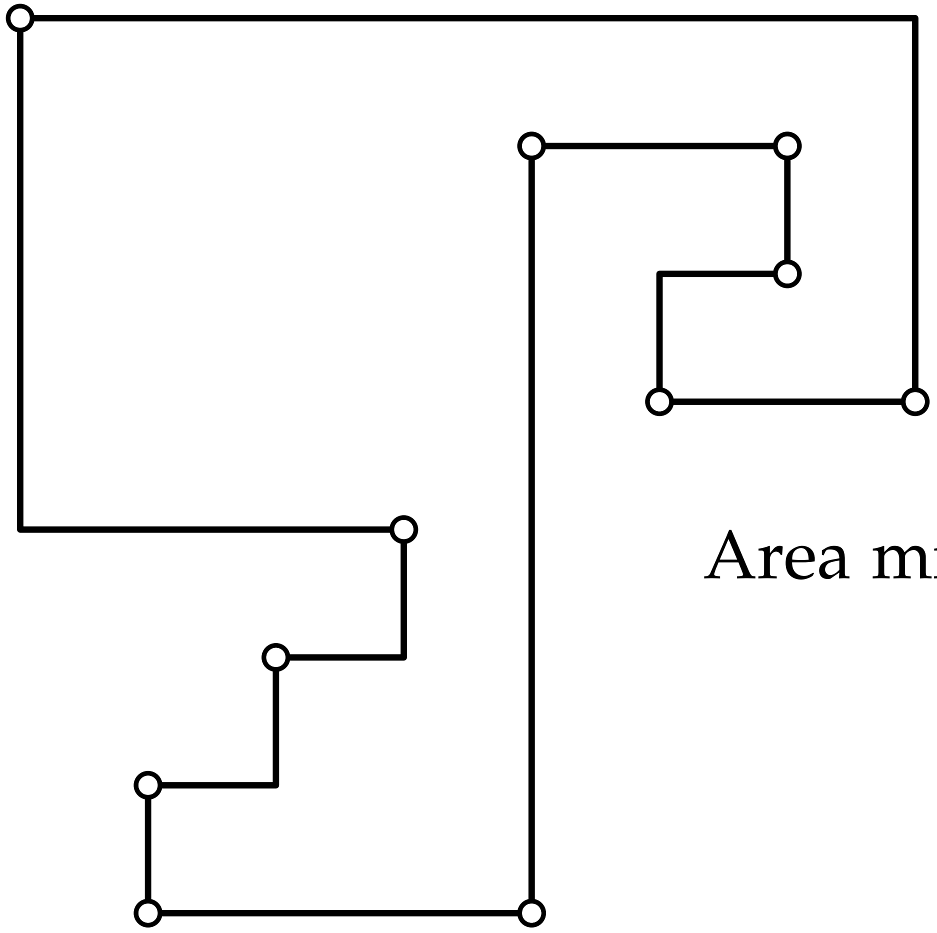
Refinement of (G, H) – Outer Face



Refinement of (G, H) – Outer Face

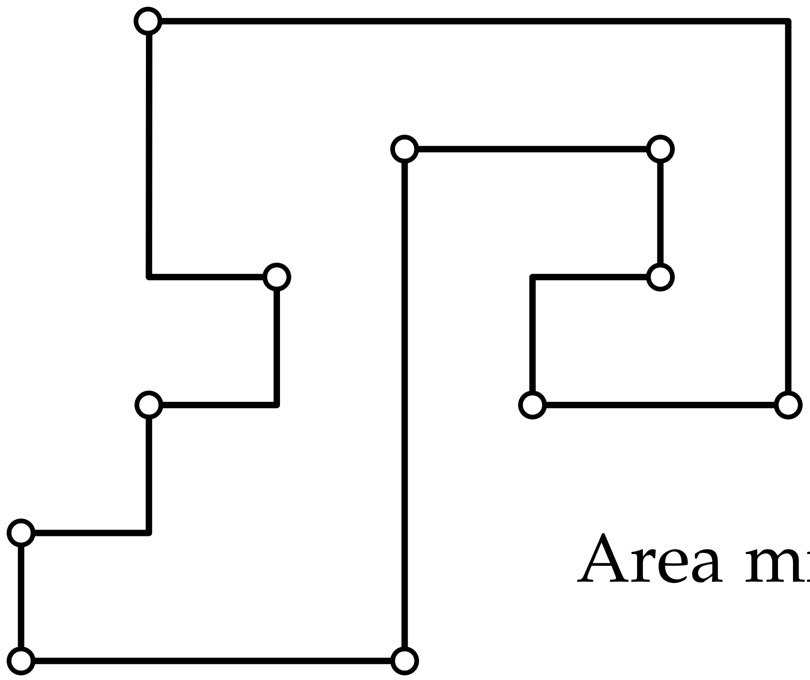


Refinement of (G, H) – Outer Face



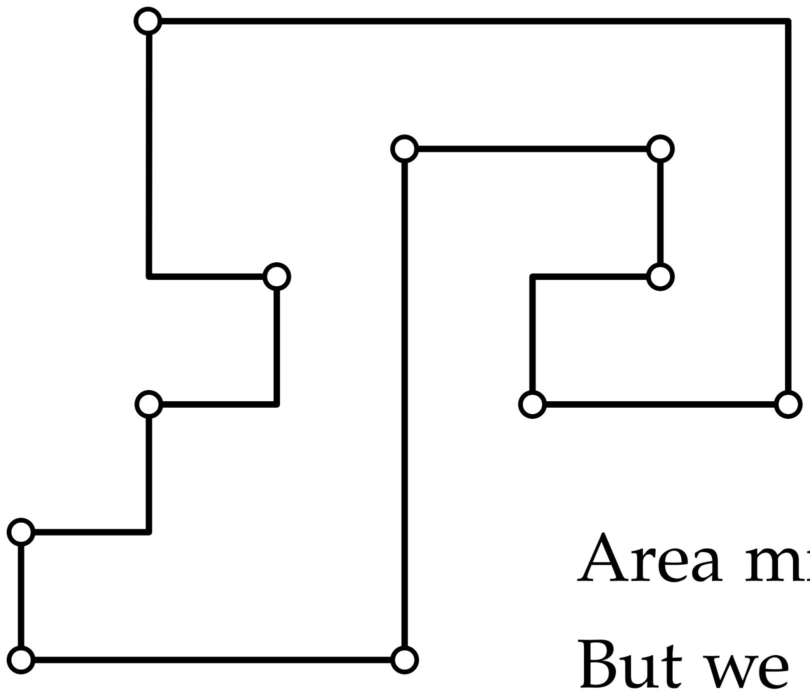
Area minimized?

Refinement of (G, H) – Outer Face



Area minimized? No!

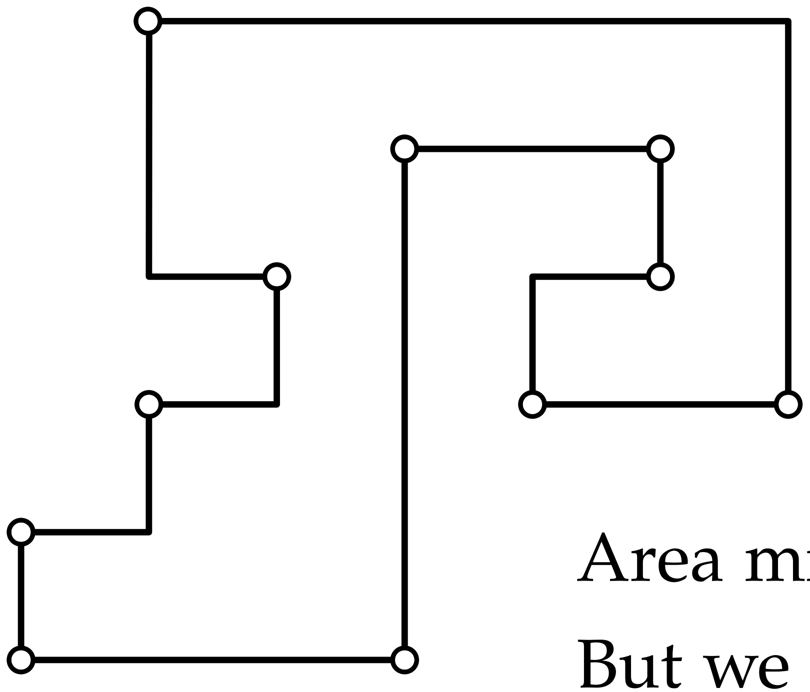
Refinement of (G, H) – Outer Face



Area minimized? No!

But we get bound $O((n+b)^2)$ on the area.

Refinement of (G, H) – Outer Face



Area minimized? No!

But we get bound $O((n+b)^2)$ on the area.

Theorem. [Patrignani 2001]
Compaction for given orthogonal representation is in general NP-hard.