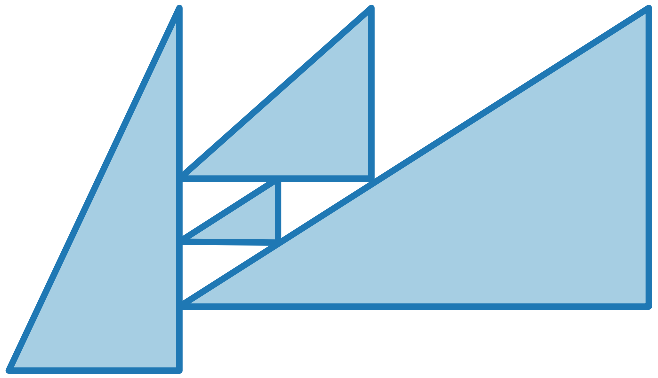
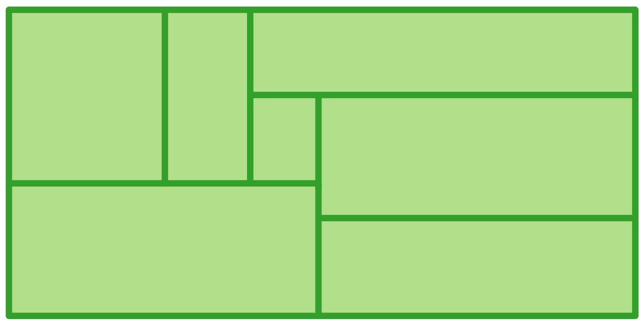
Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Visualization of Graphs

Lecture 7:
Contact Representations of Planar Graphs:
Triangle Contacts and Rectangular Duals

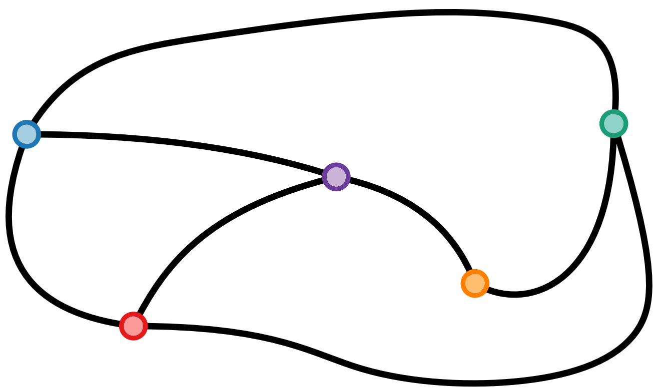


Johannes Zink



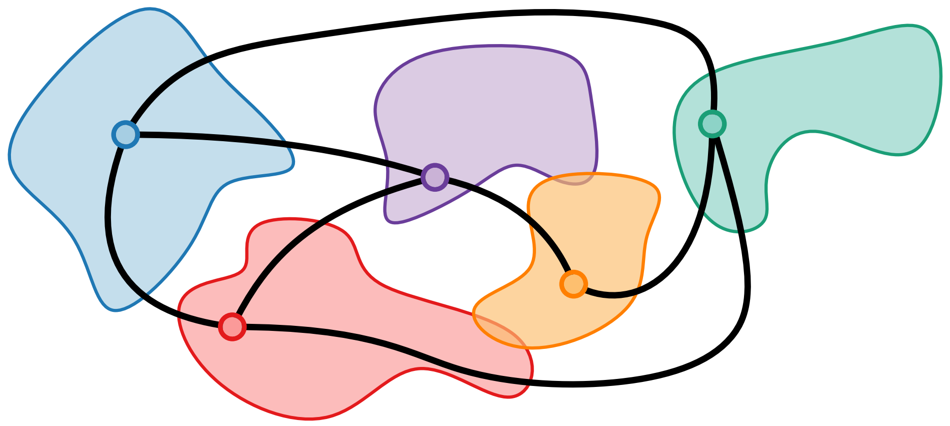
Intersection Representation of Graphs

In an **intersection representation** of a graph,
– each vertex is represented by a set



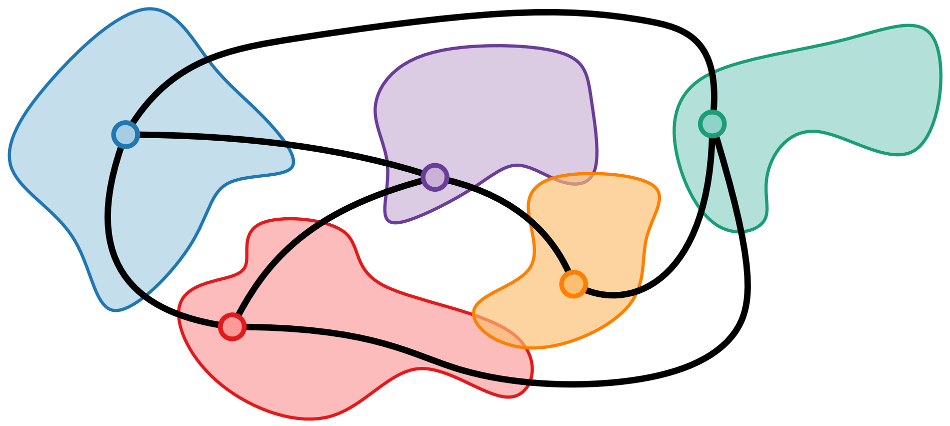
Intersection Representation of Graphs

In an **intersection representation** of a graph,
– each vertex is represented by a set
– such that



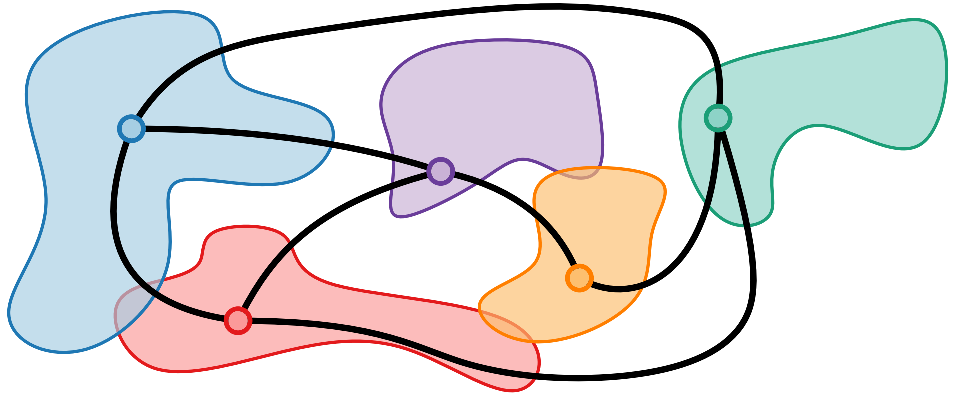
Intersection Representation of Graphs

In an **intersection representation** of a graph,
– each vertex is represented by a set
– such that two sets intersect ⇔ the corresponding vertices are adjacent.



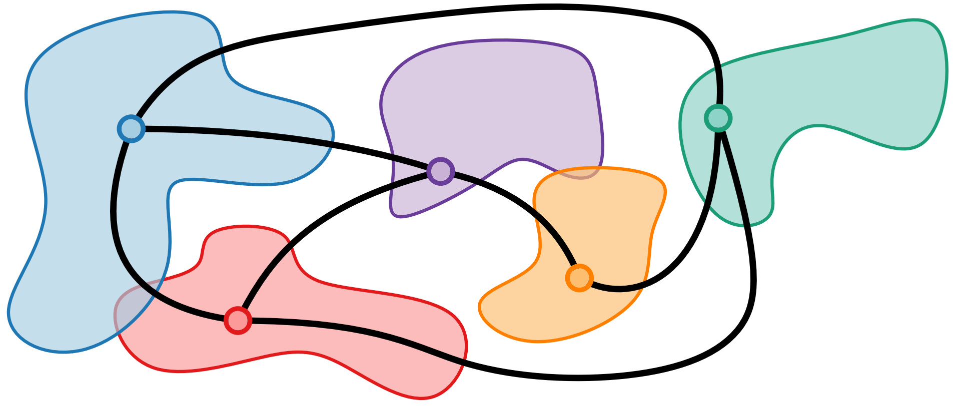
Intersection Representation of Graphs

In an **intersection representation** of a graph,
- each vertex is represented by a set
- such that two sets intersect ⇔ the corresponding vertices are adjacent.



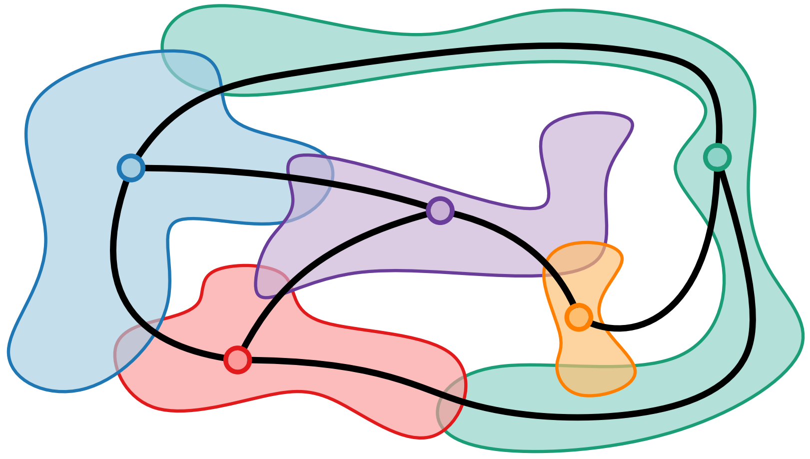
Intersection Representation of Graphs

In an **intersection representation** of a graph,
– each vertex is represented by a set
– such that two sets intersect ⇔ the corresponding vertices are adjacent.



Intersection Representation of Graphs

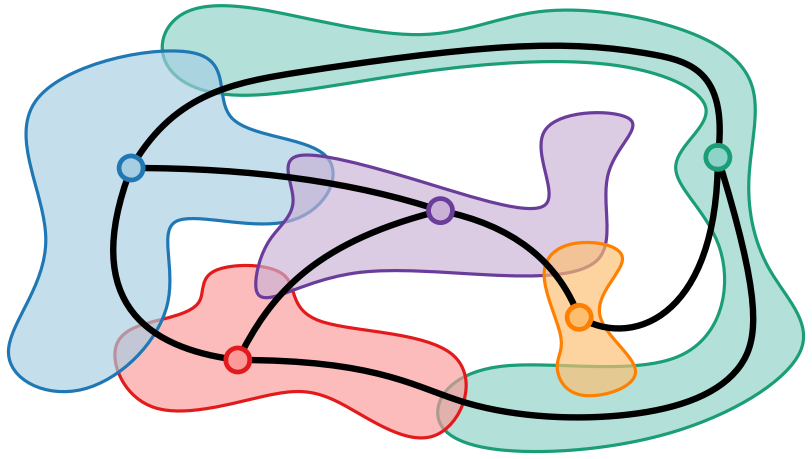
In an **intersection representation** of a graph,
– each vertex is represented by a set
– such that two sets intersect ⇔ the corresponding vertices are adjacent.



Intersection Representation of Graphs

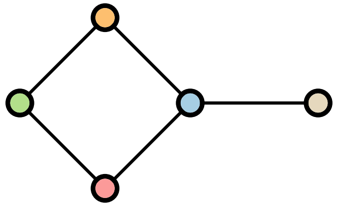
In an **intersection representation** of a graph,
– each vertex is represented by a set
– such that two sets intersect ⇔ the corresponding vertices are adjacent.

For a collection $\mathcal{S}$ of sets, the **intersection graph** $G(\mathcal{S})$ of $\mathcal{S}$ has vertex set $\mathcal{S}$ and edge set $\{\{S, S'\} \colon S, S' \in \mathcal{S}, S \neq S', \text{ and } S \cap S' \neq \emptyset\}$.



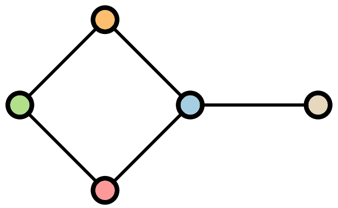
Contact Representation of Graphs

Let G be a graph.



Contact Representation of Graphs

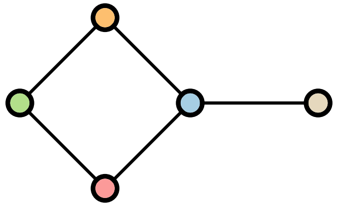
Let G be a graph.



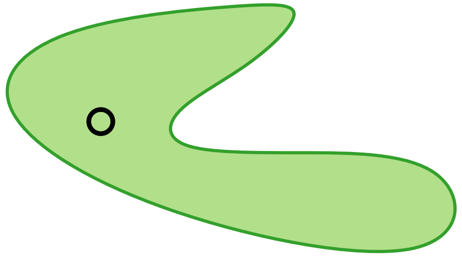
Represent each vertex v by a geometric object $S(v)$

Contact Representation of Graphs

Let G be a graph.

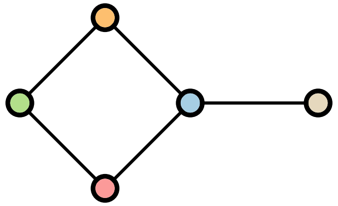


Represent each vertex v by a geometric object $S(v)$

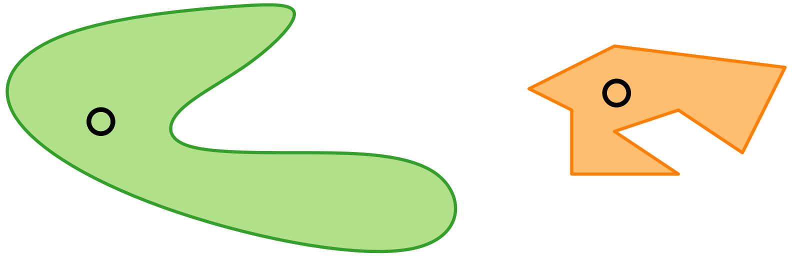


Contact Representation of Graphs

Let G be a graph.

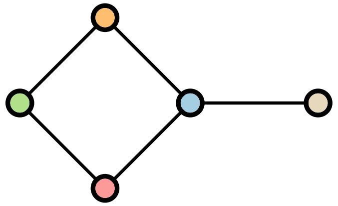


Represent each vertex v by a geometric object $S(v)$

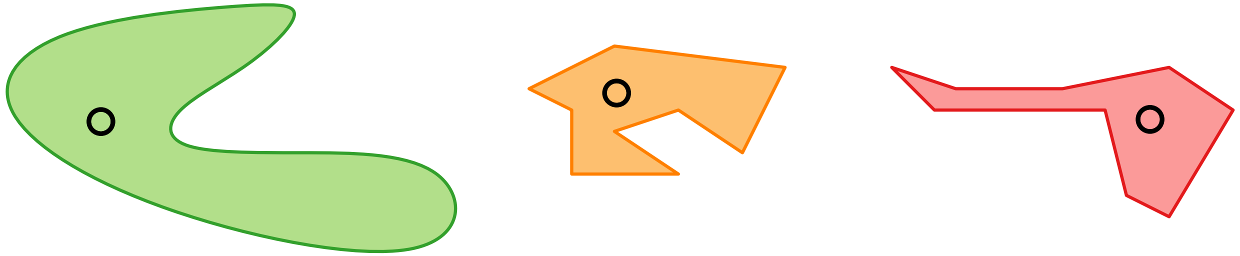


Contact Representation of Graphs

Let G be a graph.

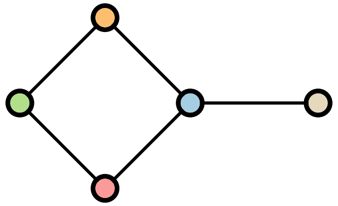


Represent each vertex v by a geometric object $S(v)$

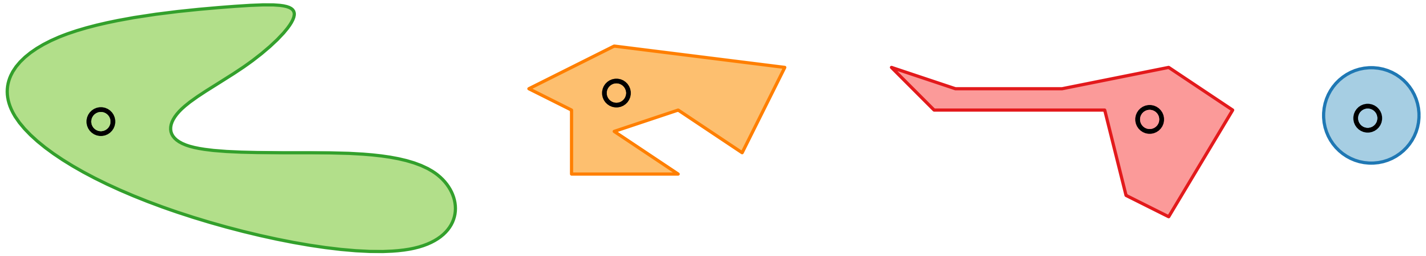


Contact Representation of Graphs

Let G be a graph.

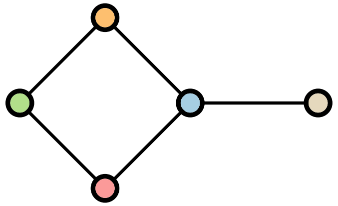


Represent each vertex v by a geometric object $S(v)$

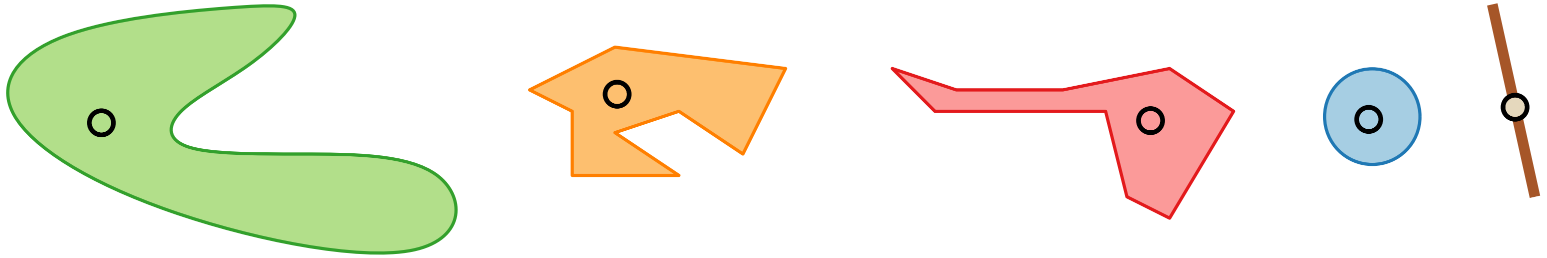


Contact Representation of Graphs

Let G be a graph.

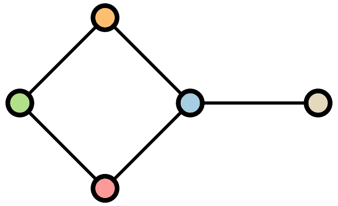


Represent each vertex v by a geometric object $S(v)$

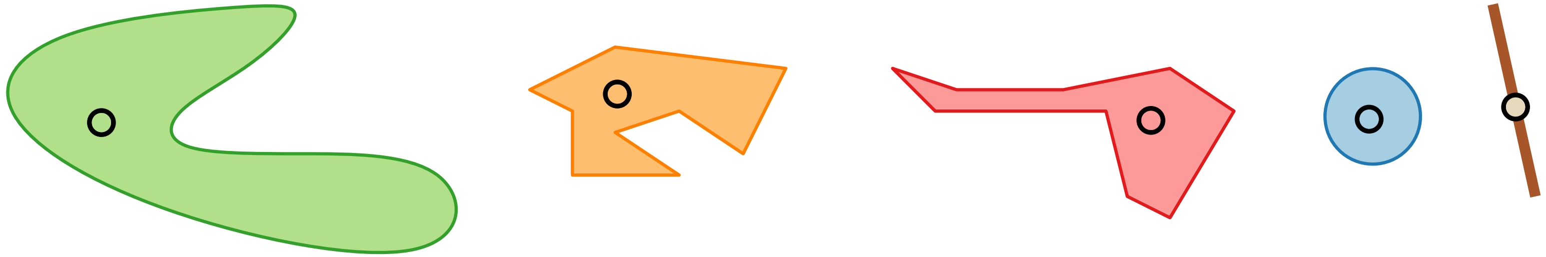


Contact Representation of Graphs

Let G be a graph.



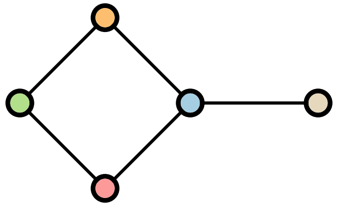
Represent each vertex v by a geometric object $S(v)$



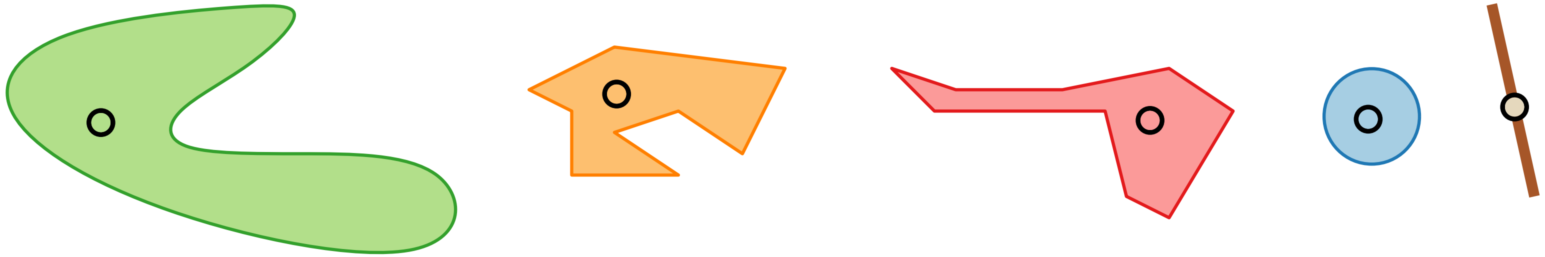
In a **contact representation** of G, $S(u)$ and $S(v)$ *touch* iff $uv \in E$

Contact Representation of Graphs

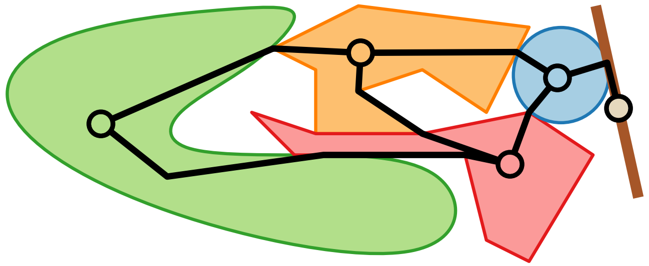
Let G be a graph.



Represent each vertex v by a geometric object $S(v)$

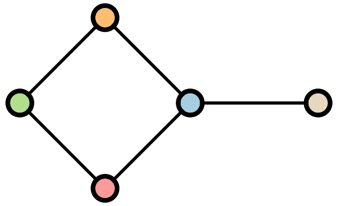


In a **contact representation** of G, $S(u)$ and $S(v)$ *touch* iff $uv \in E$



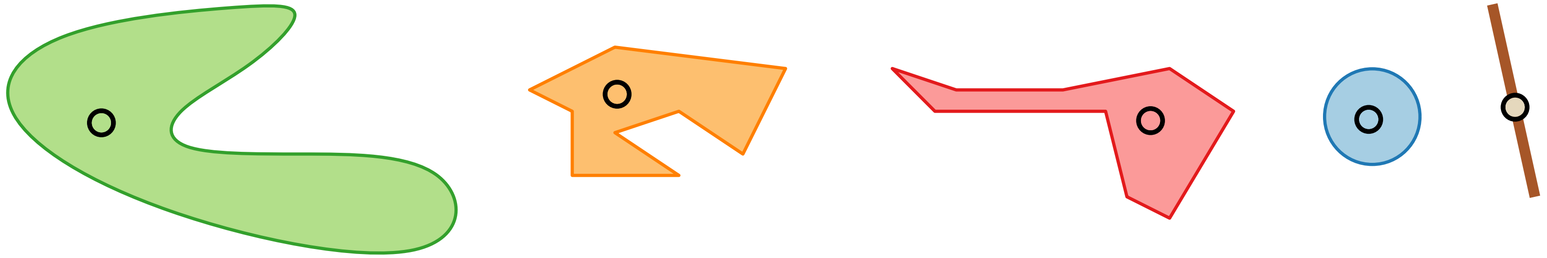
Contact Representation of Graphs

Let G be a graph.

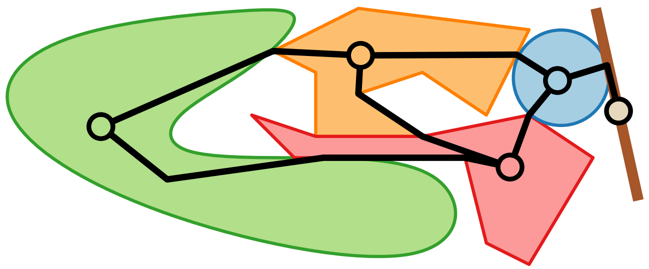


Let $\mathcal{S}$ be a family of geometric objects (e.g., disks).

Represent each vertex v by a geometric object $S(v)$

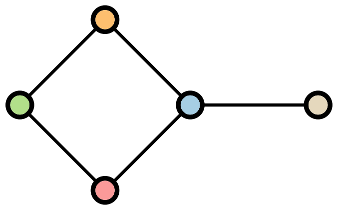


In a **contact representation** of G, $S(u)$ and $S(v)$ *touch* iff $uv \in E$



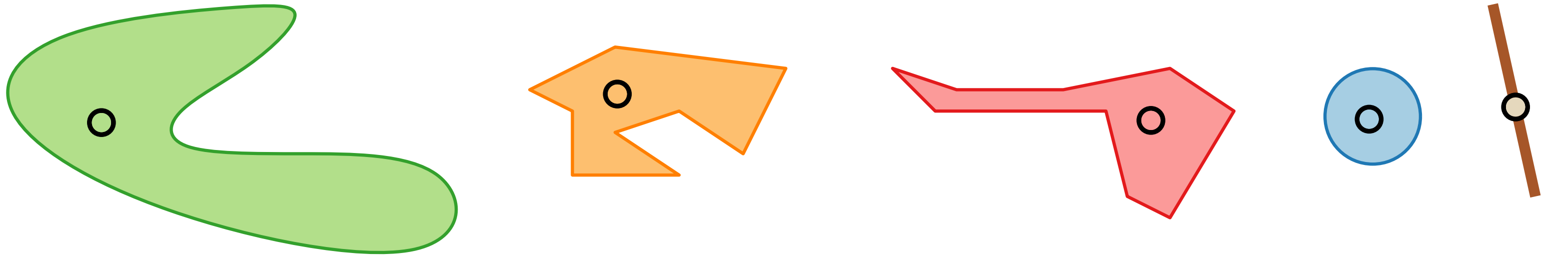
Contact Representation of Graphs

Let G be a graph.

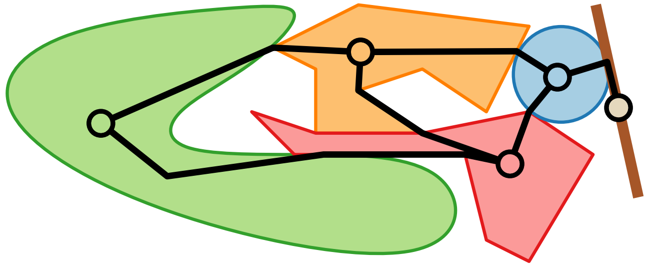


Let $\mathcal{S}$ be a family of geometric objects (e.g., disks).

Represent each vertex v by a geometric object $S(v) \in \mathcal{S}$

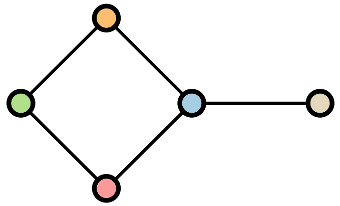


In a **contact representation** of G, $S(u)$ and $S(v)$ *touch* iff $uv \in E$



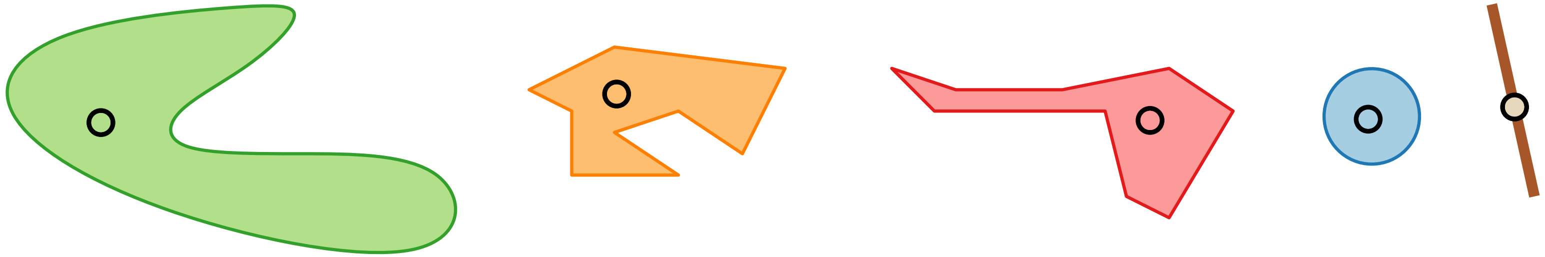
Contact Representation of Graphs

Let G be a graph.

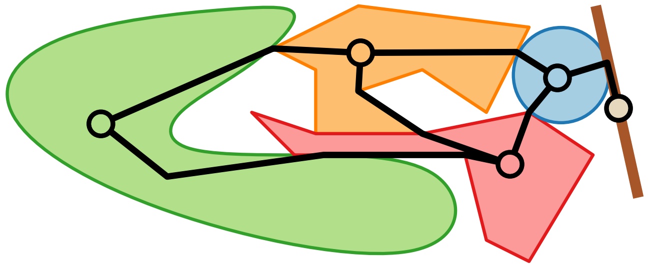


Let $\mathcal{S}$ be a family of geometric objects (e.g., disks).

Represent each vertex v by a geometric object $S(v) \in \mathcal{S}$

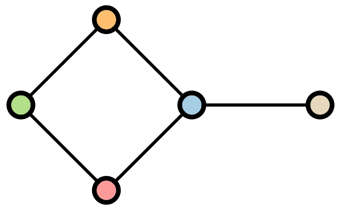


In an **$\mathcal{S}$-contact representation** of G, $S(u)$ and $S(v)$ *touch* iff $uv \in E$



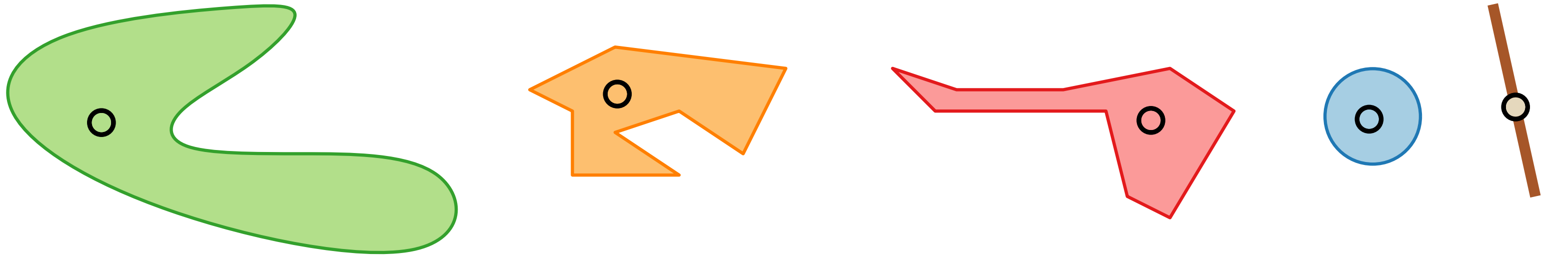
Contact Representation of Graphs

Let G be a graph.

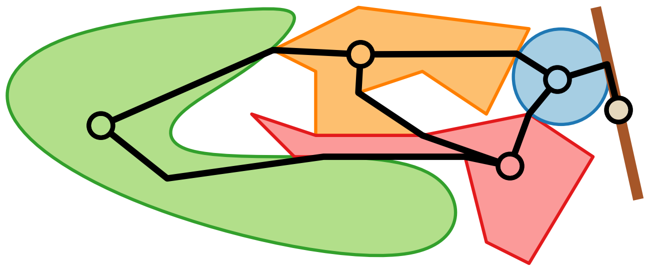


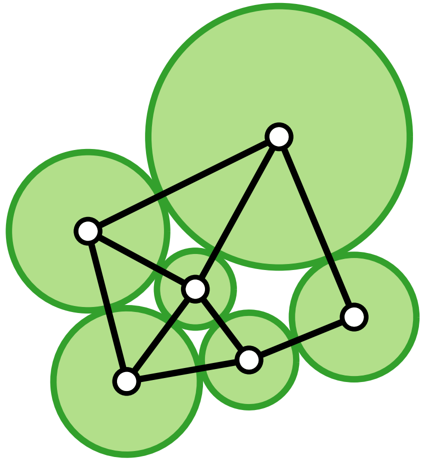
Let $\mathcal{S}$ be a family of geometric objects (e.g., disks).

Represent each vertex v by a geometric object $S(v) \in \mathcal{S}$



In an **$\mathcal{S}$-contact representation** of G, $S(u)$ and $S(v)$ *touch* iff $uv \in E$

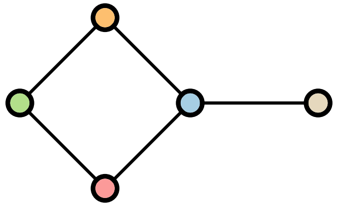




disks

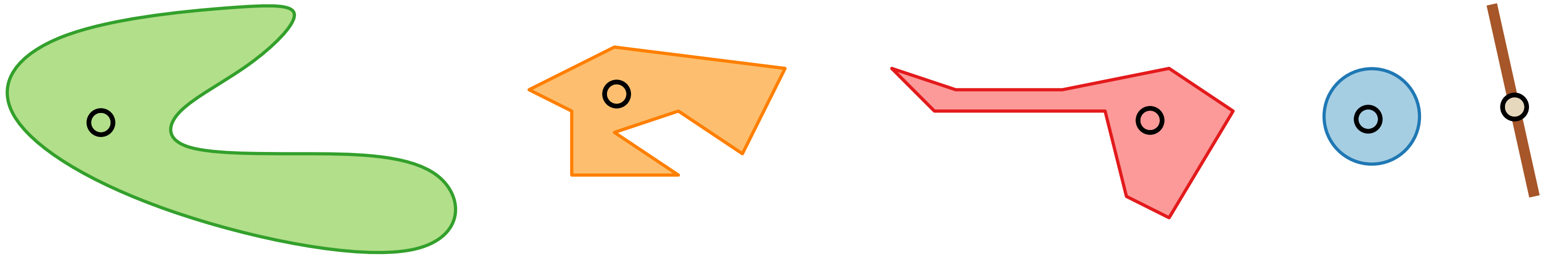
Contact Representation of Graphs

Let G be a graph.

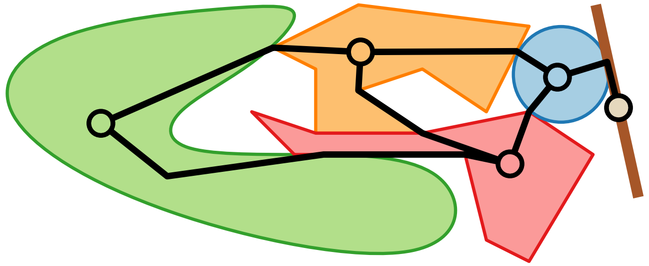


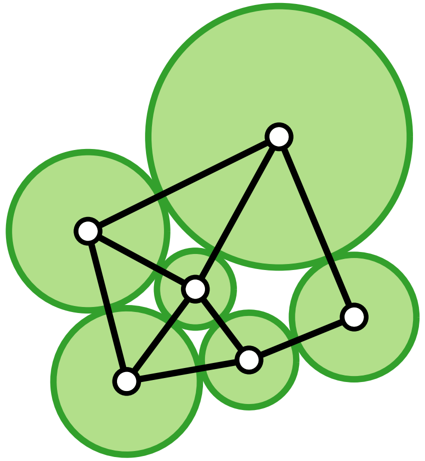
Let $\mathcal{S}$ be a family of geometric objects (e.g., disks).

Represent each vertex v by a geometric object $S(v) \in \mathcal{S}$

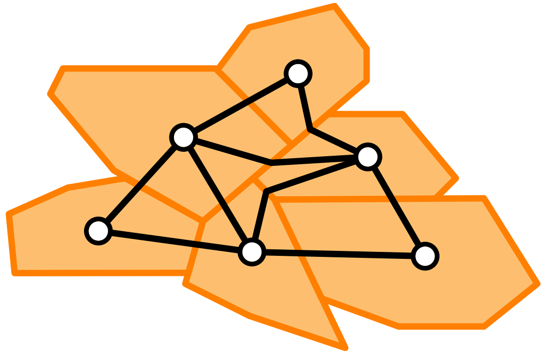


In an $\mathcal{S}$-**contact representation** of G, $S(u)$ and $S(v)$ *touch* iff $uv \in E$



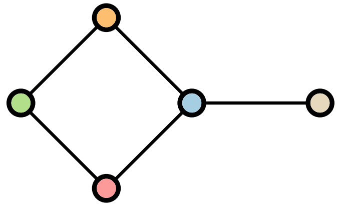


disks

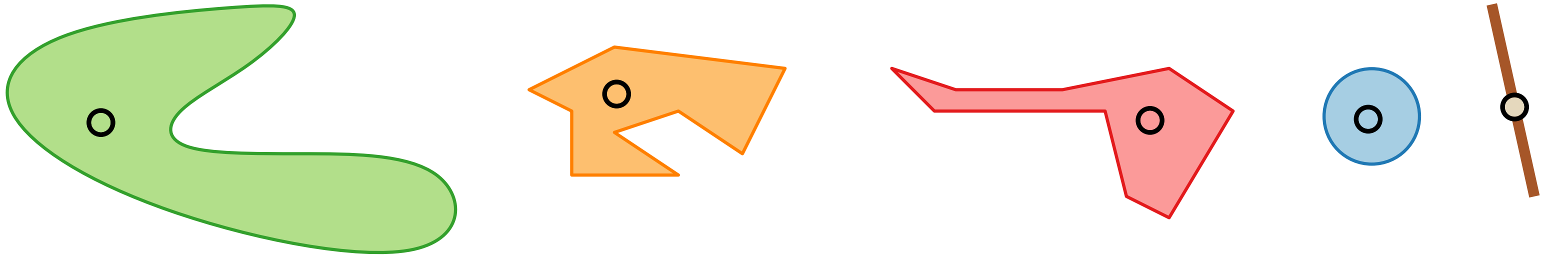


polygons

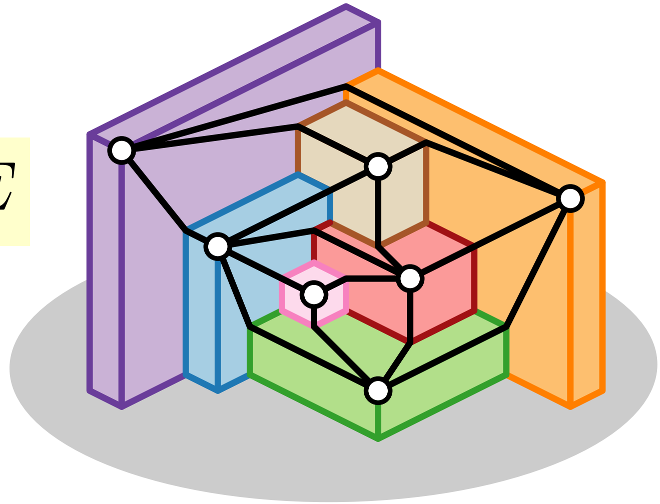
Contact Representation of Graphs

Let G be a graph.  Let $\mathcal{S}$ be a family of geometric objects (e.g., disks).

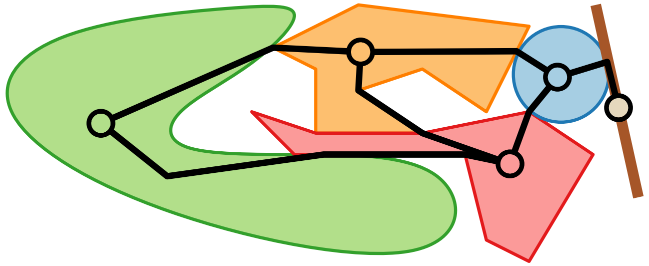
Represent each vertex v by a geometric object $S(v) \in \mathcal{S}$

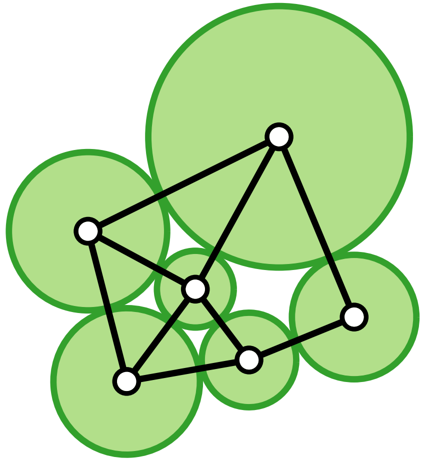


rectangular cuboids

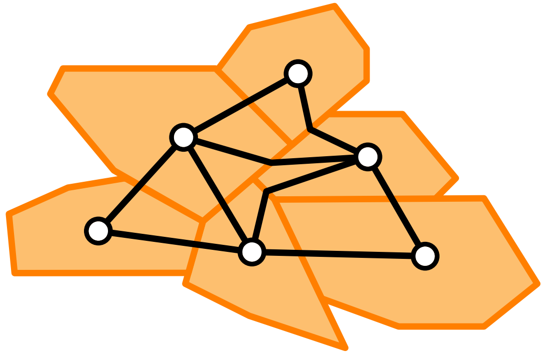


In an $\mathcal{S}$-**contact representation** of G, $S(u)$ and $S(v)$ *touch* iff $uv \in E$





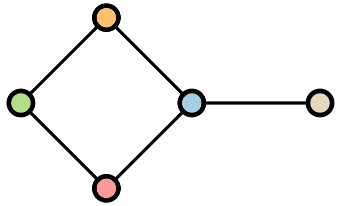
disks



polygons

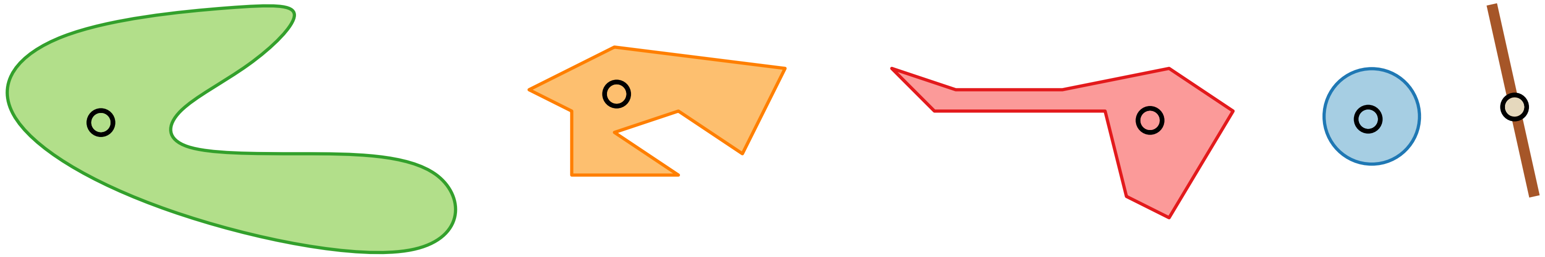
Contact Representation of Graphs

Let G be a graph.

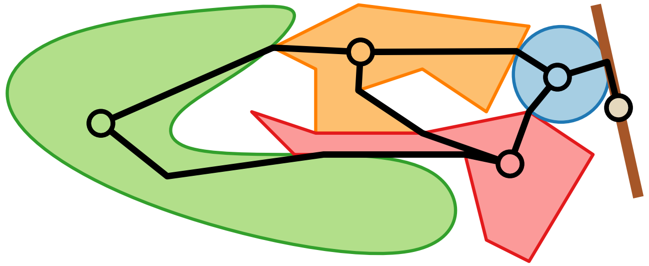


Let $\mathcal{S}$ be a family of geometric objects (e.g., disks).

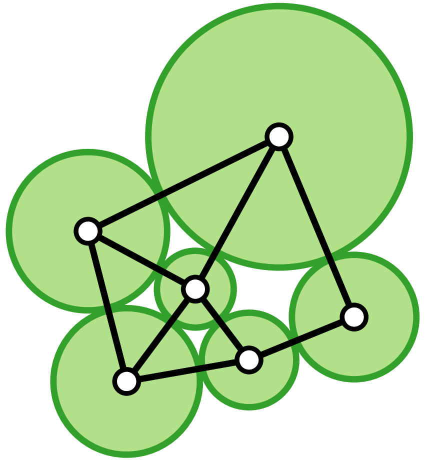
Represent each vertex v by a geometric object $S(v) \in \mathcal{S}$



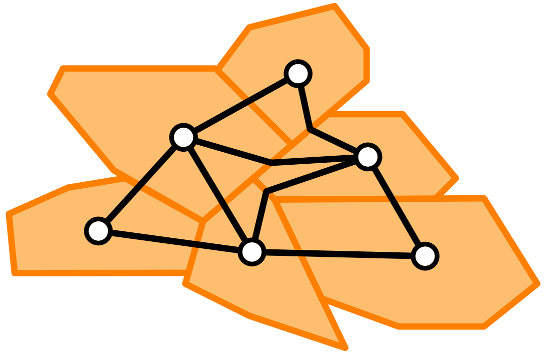
In an **$\mathcal{S}$-contact representation** of G, $S(u)$ and $S(v)$ *touch* iff $uv \in E$



G is planar

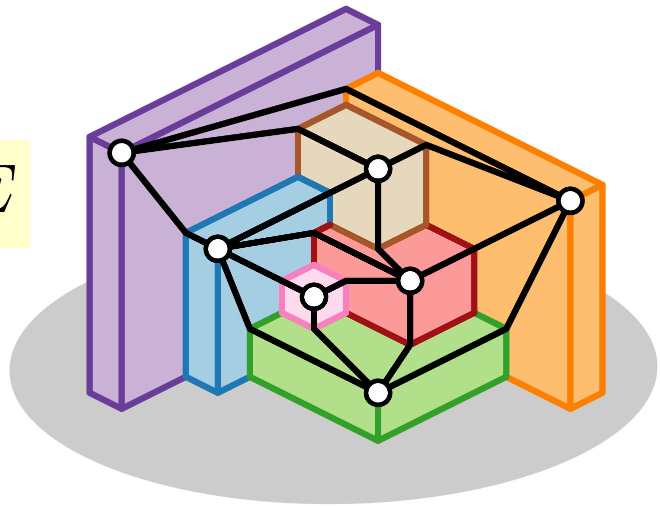


disks



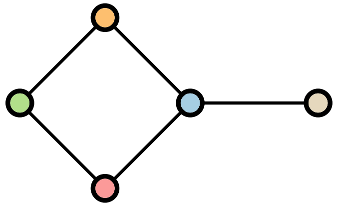
polygons

rectangular cuboids



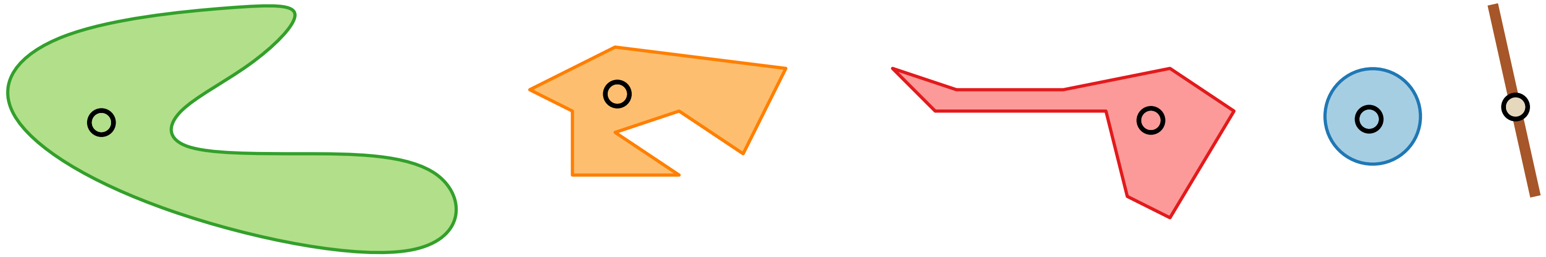
Contact Representation of Graphs

Let G be a graph.

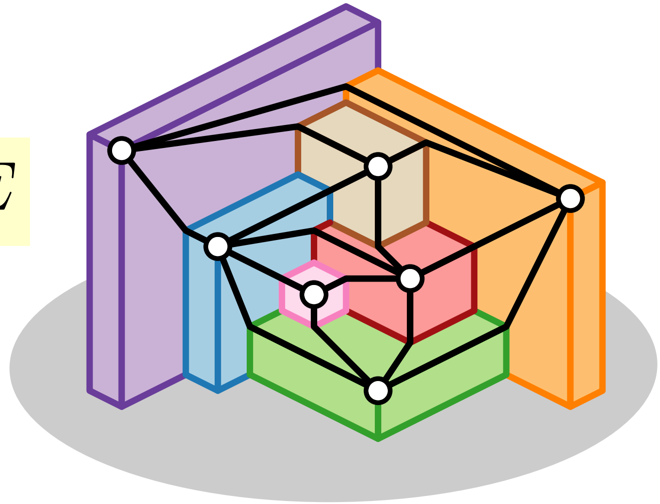


Let $\mathcal{S}$ be a family of geometric objects (e.g., disks).

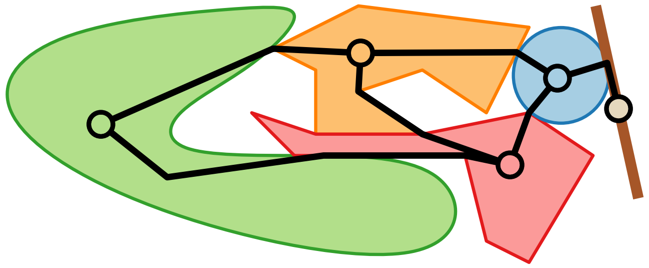
Represent each vertex v by a geometric object $S(v) \in \mathcal{S}$

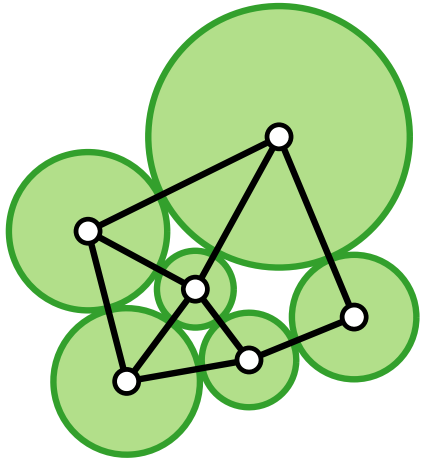


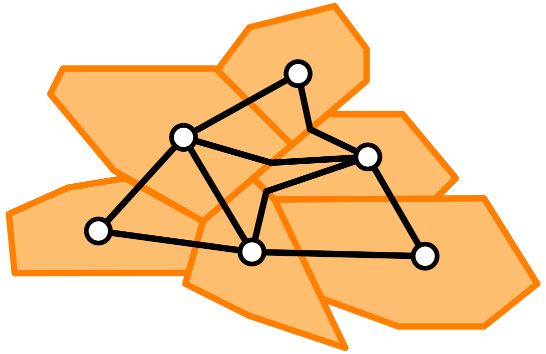
rectangular cuboids



In an **$\mathcal{S}$-contact representation** of G, $S(u)$ and $S(v)$ *touch* iff $uv \in E$





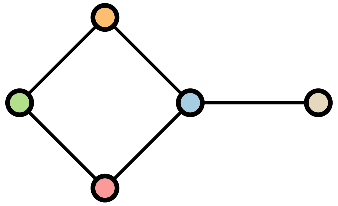


G is planar → disks [Koebe 1936]

polygons

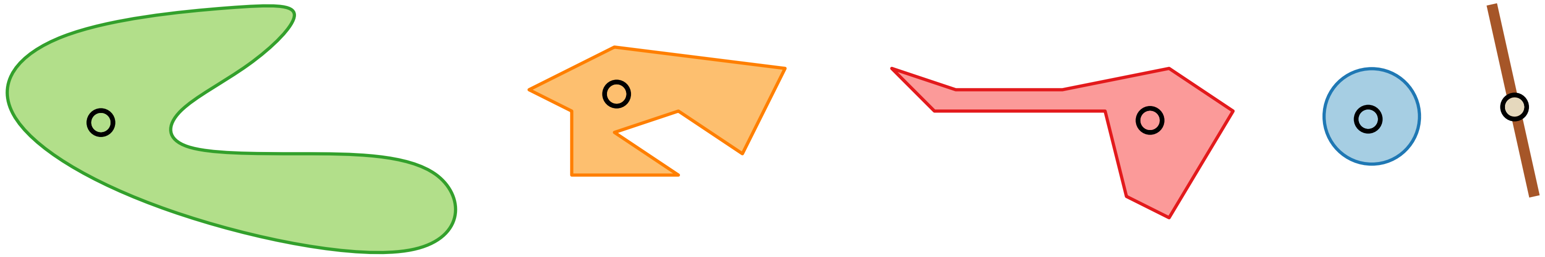
Contact Representation of Graphs

Let G be a graph.

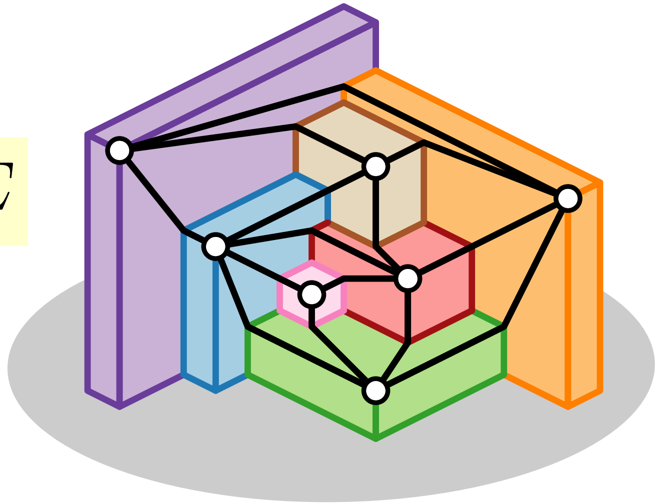


Let $\mathcal{S}$ be a family of geometric objects (e.g., disks).

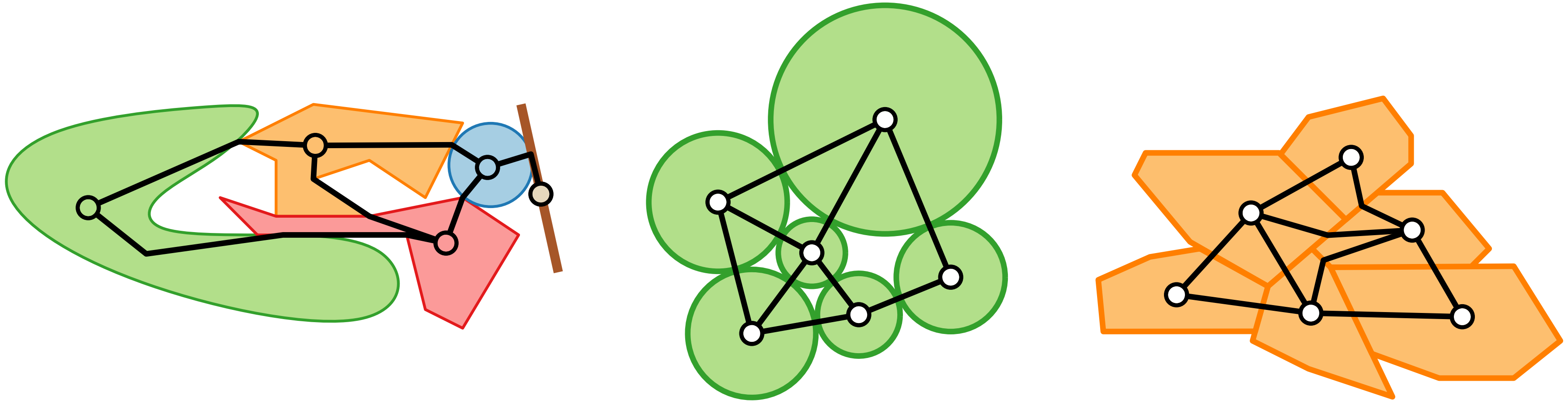
Represent each vertex v by a geometric object $S(v) \in \mathcal{S}$



rectangular cuboids



In an **$\mathcal{S}$-contact representation** of G, $S(u)$ and $S(v)$ *touch* iff $uv \in E$

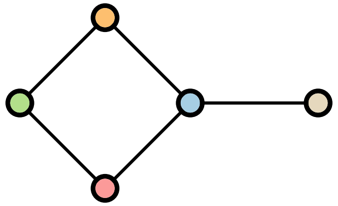


G is planar → disks → polygons

[Koebe 1936]

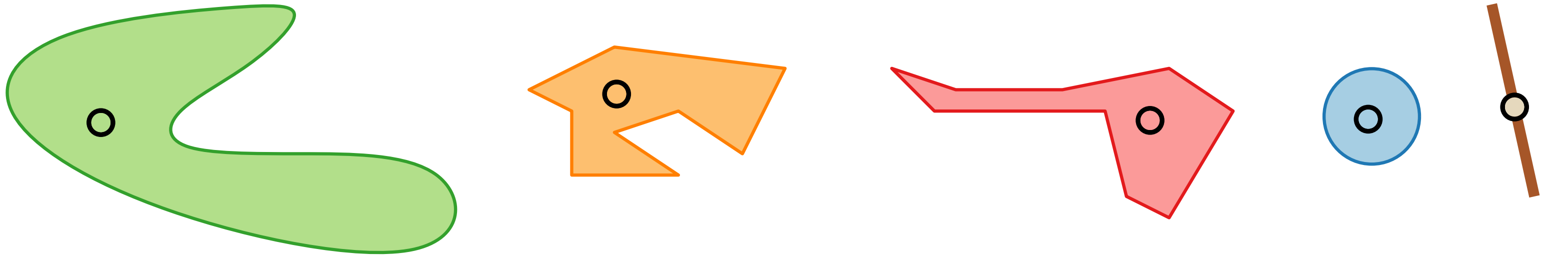
Contact Representation of Graphs

Let G be a graph.

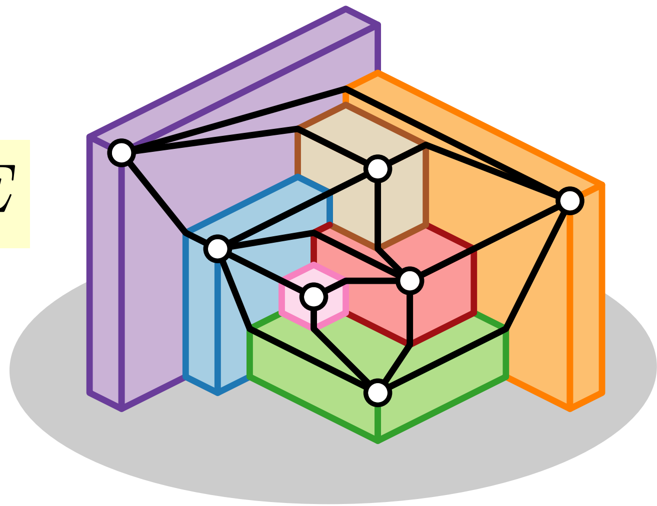


Let $\mathcal{S}$ be a family of geometric objects (e.g., disks).

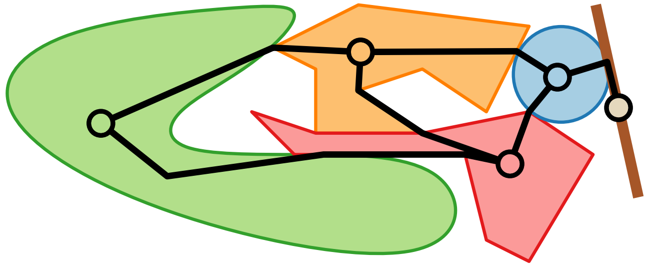
Represent each vertex v by a geometric object $S(v) \in \mathcal{S}$

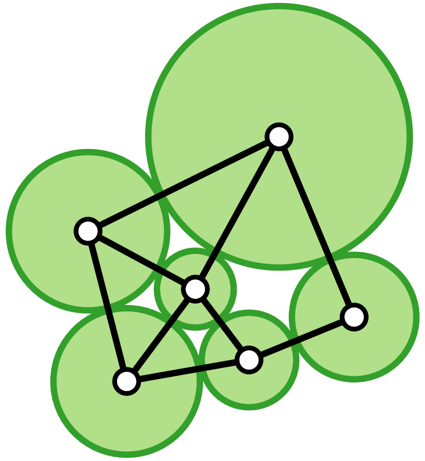


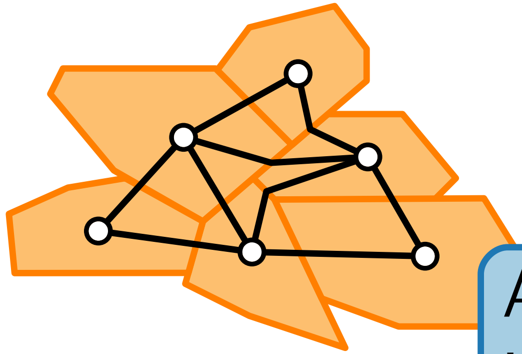
rectangular cuboids



In an $\mathcal{S}$-**contact representation** of G, $S(u)$ and $S(v)$ *touch* iff $uv \in E$







G is planar → [Koebe 1936] disks → polygons

A contact representation is an intersection representation with interior-disjoint sets.

Contact Representation of Planar Graphs

Is the intersection graph of a contact representation always planar?

Contact Representation of Planar Graphs

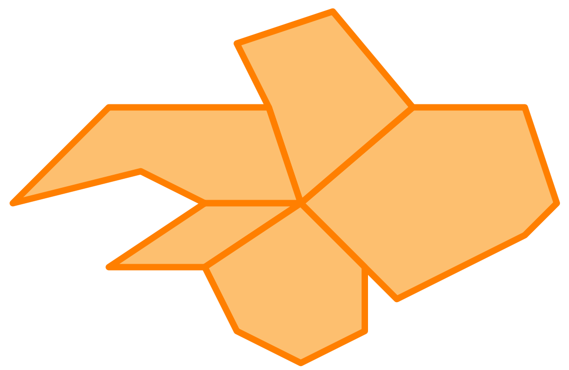
Is the intersection graph of a contact representation always planar?

- No, not even for connected object types in the plane.

Contact Representation of Planar Graphs

Is the intersection graph of a contact representation always planar?

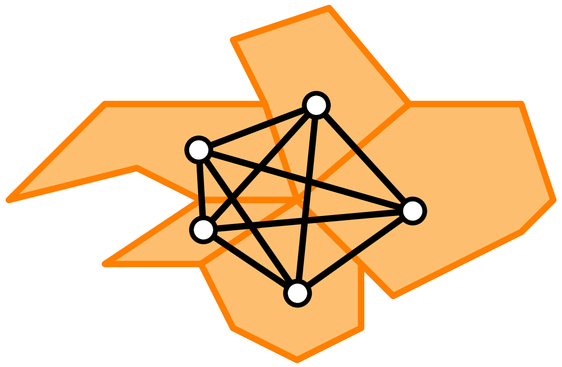
- No, not even for connected object types in the plane.



Contact Representation of Planar Graphs

Is the intersection graph of a contact representation always planar?

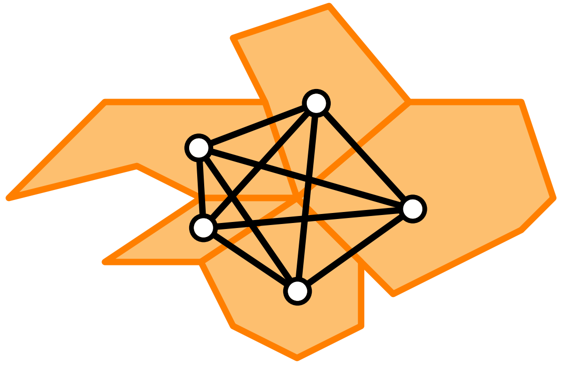
- No, not even for connected object types in the plane.



Contact Representation of Planar Graphs

Is the intersection graph of a contact representation always planar?

- No, not even for connected object types in the plane.

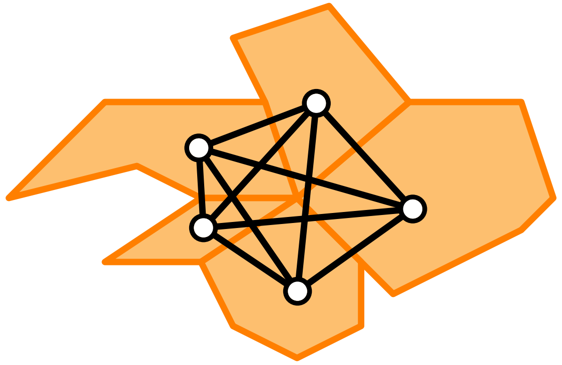


Some object types are used to represent special classes of planar graphs:

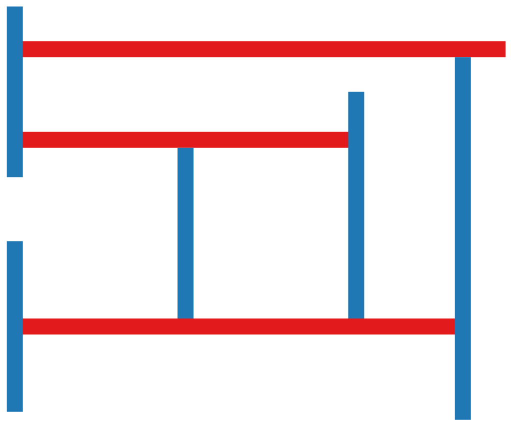
Contact Representation of Planar Graphs

Is the intersection graph of a contact representation always planar?

- No, not even for connected object types in the plane.



Some object types are used to represent special classes of planar graphs:

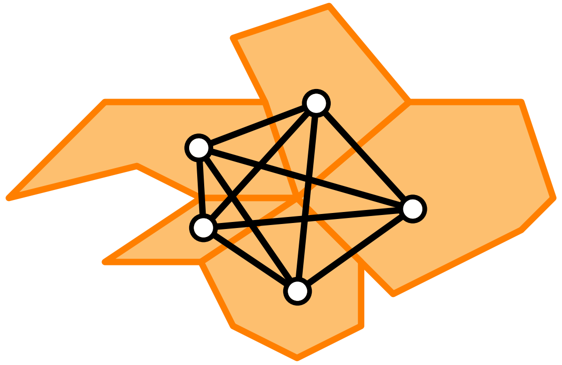


bipartite planar graphs

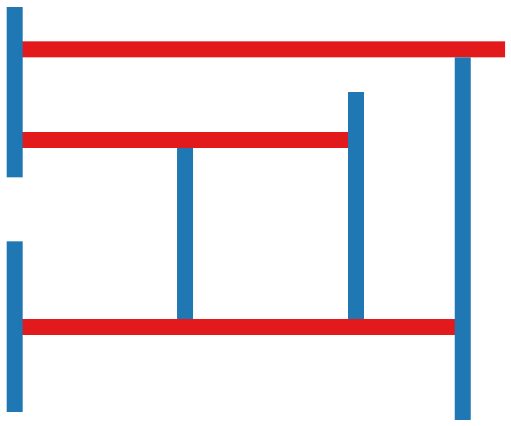
Contact Representation of Planar Graphs

Is the intersection graph of a contact representation always planar?

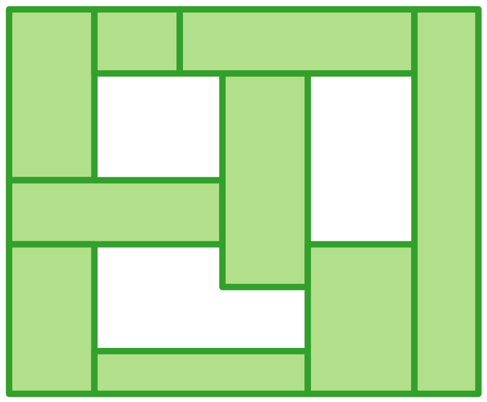
- No, not even for connected object types in the plane.



Some object types are used to represent special classes of planar graphs:



bipartite planar graphs

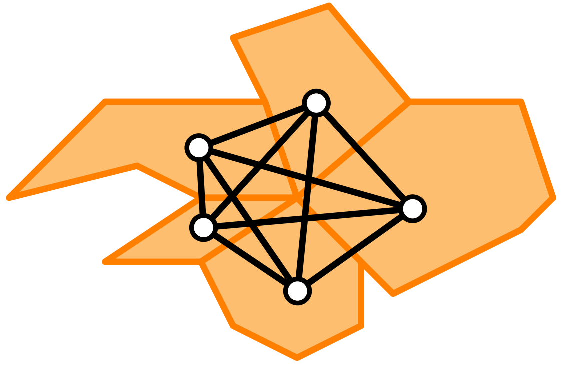


max. triangle-free planar graphs

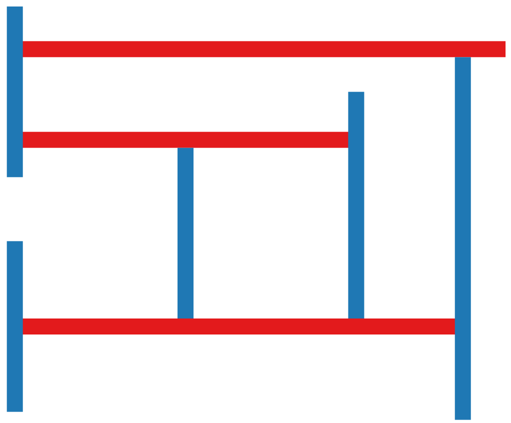
Contact Representation of Planar Graphs

Is the intersection graph of a contact representation always planar?

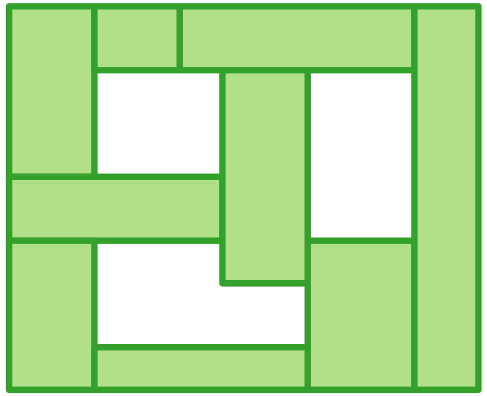
- No, not even for connected object types in the plane.



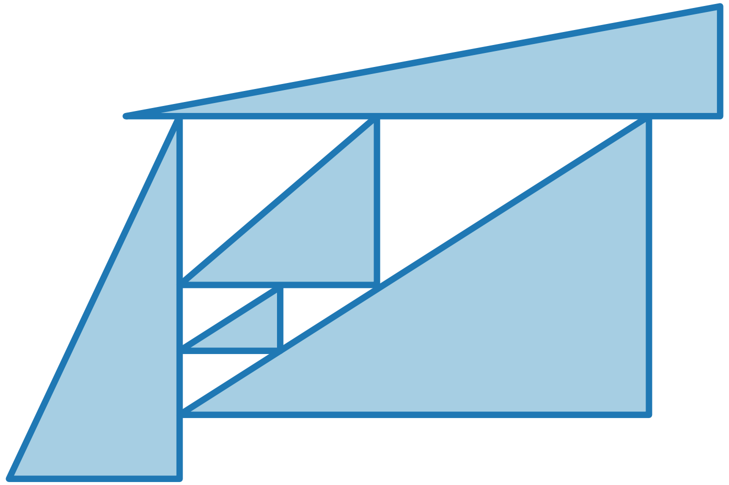
Some object types are used to represent special classes of planar graphs:



bipartite planar graphs



max. triangle-free planar graphs



planar triangulations

General Approach

How to compute a contact representation of a given graph G?

General Approach

How to compute a contact representation of a given graph G?

- Consider only inner triangulations
  (or maximal bipartite graphs, etc.)

General Approach

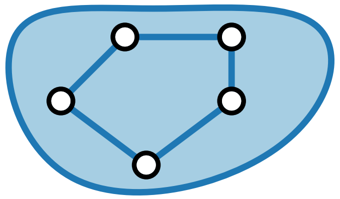
How to compute a contact representation of a given graph G?

- Consider only inner triangulations (or maximal bipartite graphs, etc.)
  - Triangulate by adding vertices, not by adding edges

General Approach

How to compute a contact representation of a given graph G?

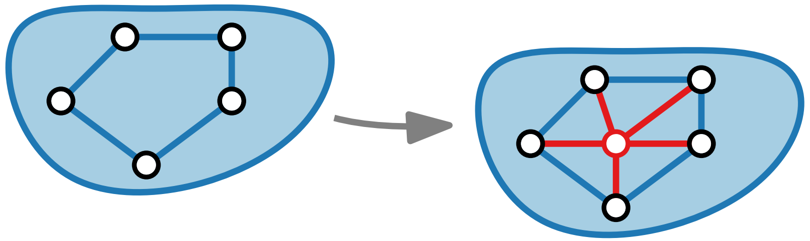
- Consider only inner triangulations (or maximal bipartite graphs, etc.)
  - Triangulate by adding vertices, not by adding edges



General Approach

How to compute a contact representation of a given graph G?

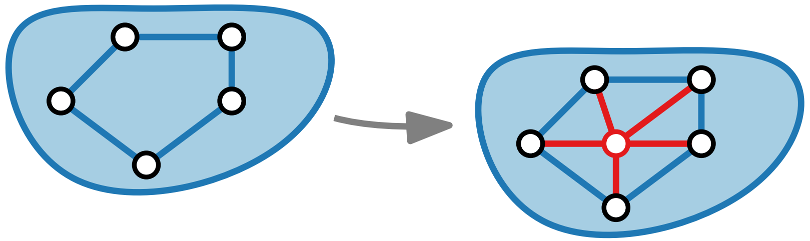
- Consider only inner triangulations (or maximal bipartite graphs, etc.)
  - Triangulate by adding vertices, not by adding edges



General Approach

How to compute a contact representation of a given graph G?

- Consider only inner triangulations
  (or maximal bipartite graphs, etc.)
  - Triangulate by adding vertices,
    not by adding edges

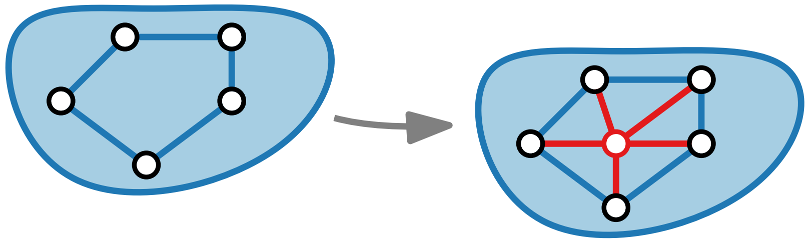


- Describe contact representation combinatorially.

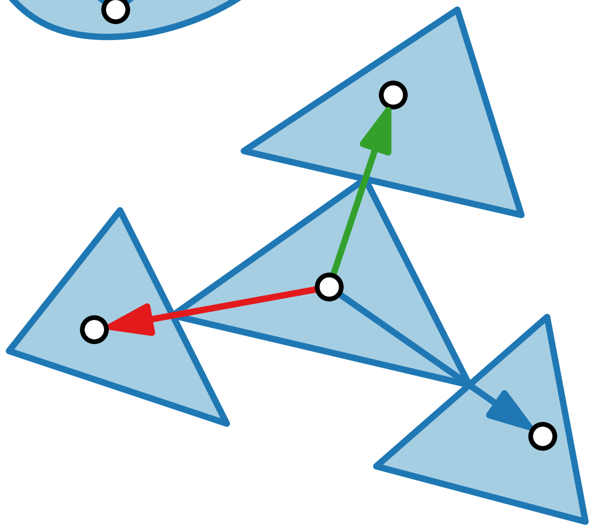
General Approach

How to compute a contact representation of a given graph G?

- Consider only inner triangulations (or maximal bipartite graphs, etc.)
  - Triangulate by adding vertices, not by adding edges



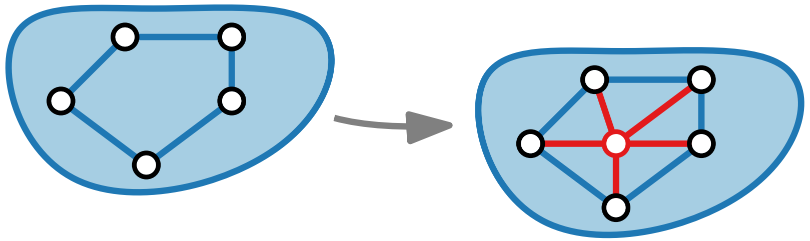
- Describe contact representation combinatorially.



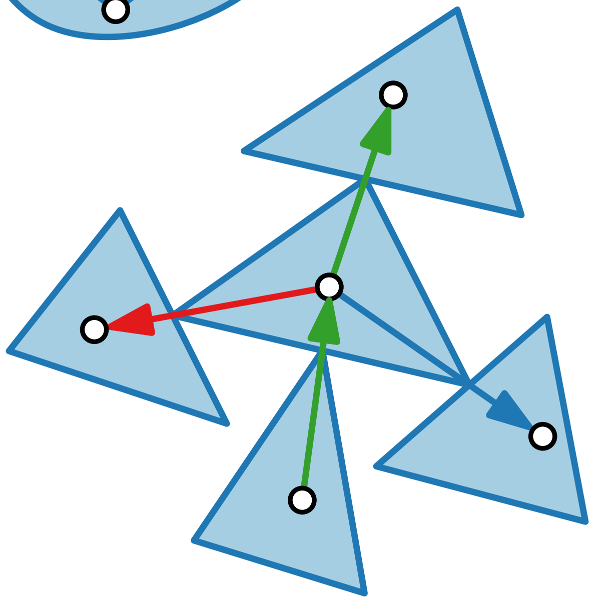
General Approach

How to compute a contact representation of a given graph G?

- Consider only inner triangulations (or maximal bipartite graphs, etc.)
  - Triangulate by adding vertices, not by adding edges



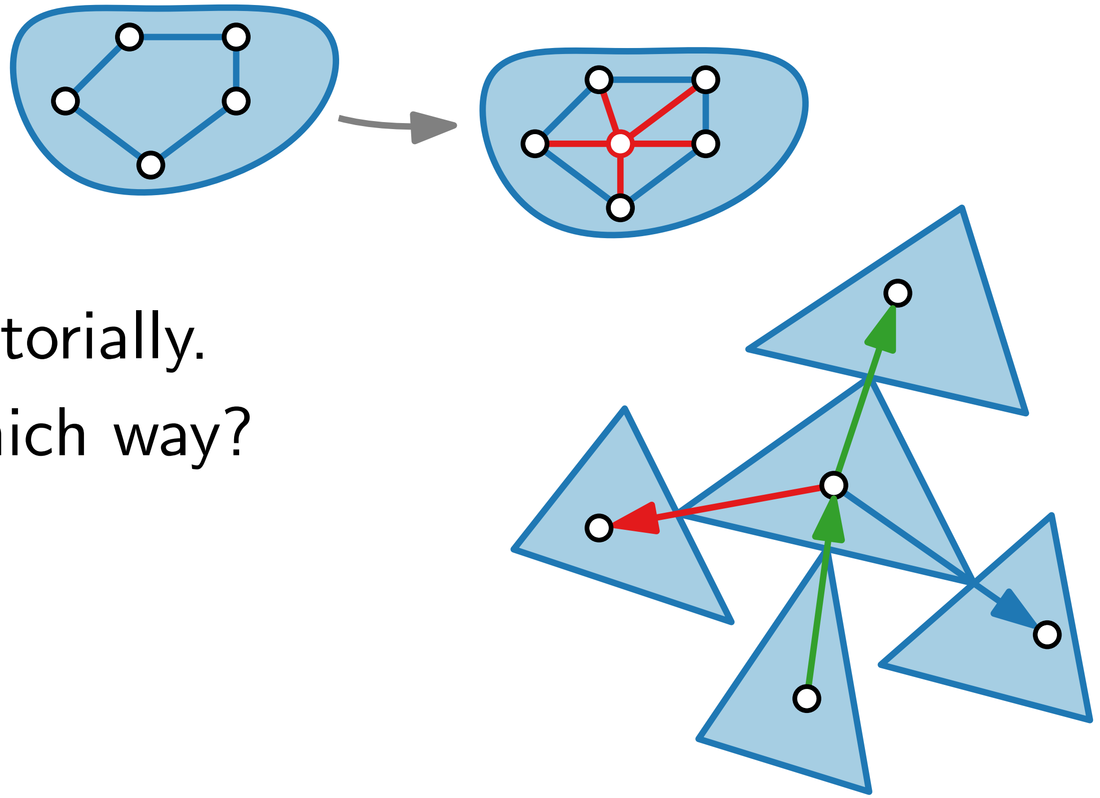
- Describe contact representation combinatorially.
  - Which objects touch each other in which way?



General Approach

How to compute a contact representation of a given graph G?

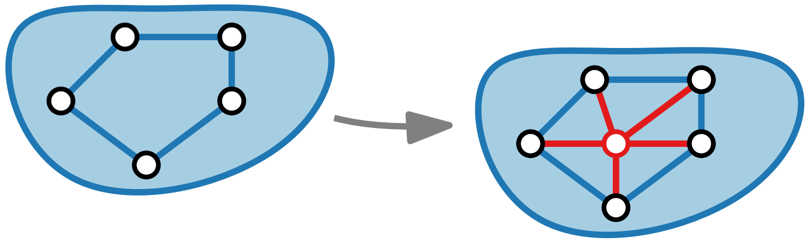
- Consider only inner triangulations (or maximal bipartite graphs, etc.)
  - Triangulate by adding vertices, not by adding edges
- Describe contact representation combinatorially.
  - Which objects touch each other in which way?
- Compute combinatorial description.



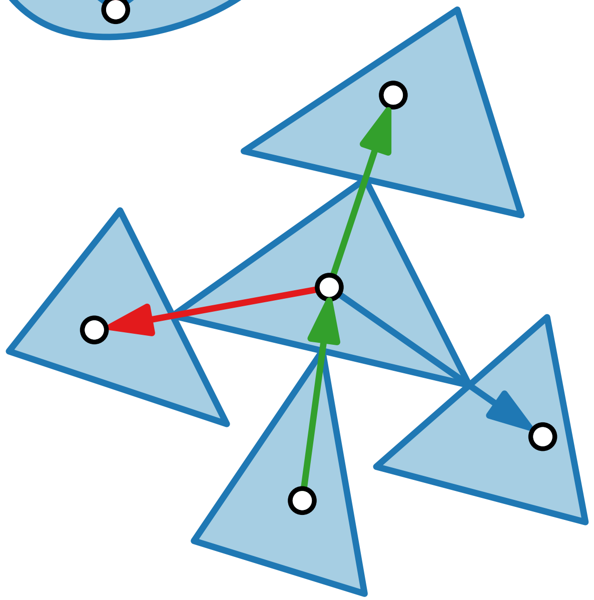
General Approach

How to compute a contact representation of a given graph G?

- Consider only inner triangulations (or maximal bipartite graphs, etc.)
  - Triangulate by adding vertices, not by adding edges

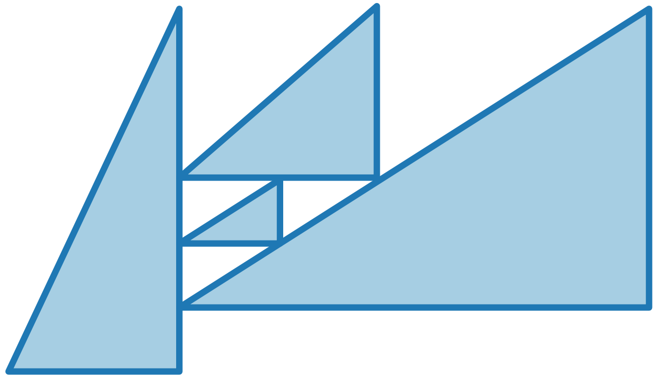


- Describe contact representation combinatorially.
  - Which objects touch each other in which way?
- Compute combinatorial description.
- Show that combinatorial description can be used to construct drawing.



This Lecture

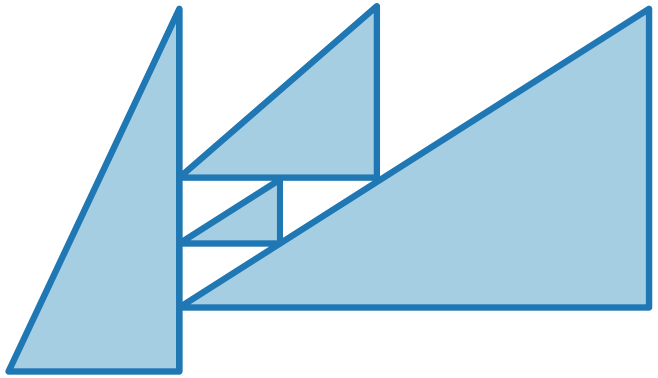
Representation with right-triangles and corner contact:



This Lecture

Representation with right-triangles and corner contact:

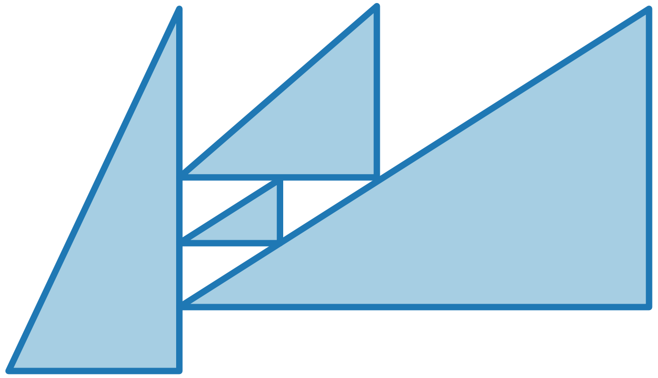
- Use Schnyder realizer to describe contacts between triangles.



This Lecture

Representation with right-triangles and corner contact:

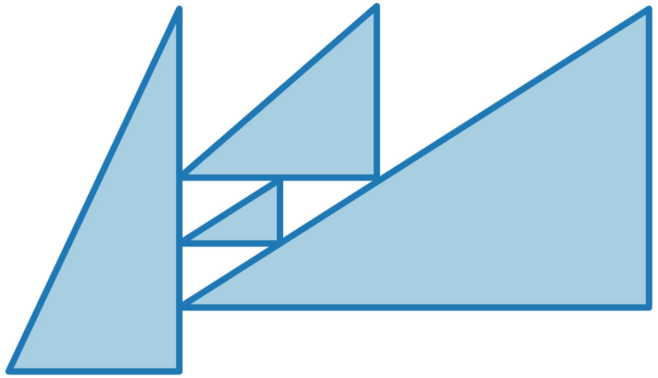
- Use Schnyder realizer to describe contacts between triangles.
- Use canonical order to compute drawing.



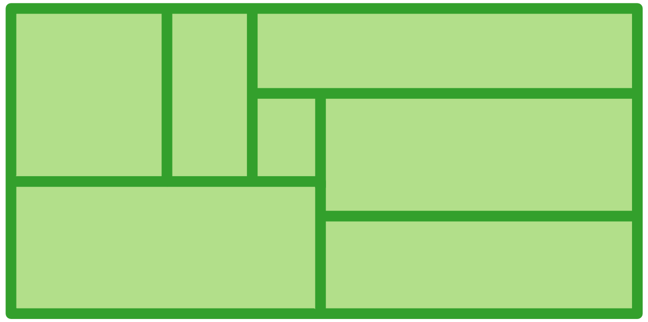
This Lecture

Representation with right-triangles and corner contact:

- Use Schnyder realizer to describe contacts between triangles.
- Use canonical order to compute drawing.



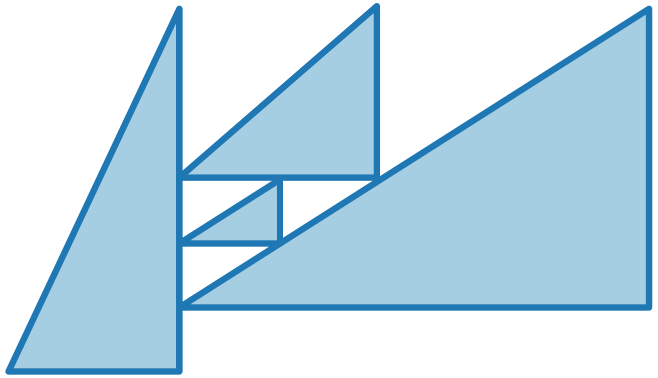
Representation with dissection of a rectangle, called **rectangular dual**:



This Lecture

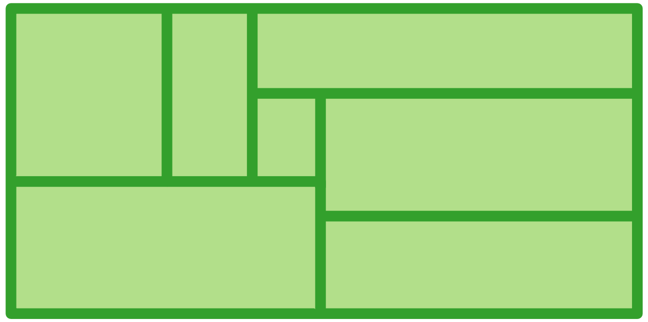
Representation with right-triangles and corner contact:

- Use Schnyder realizer to describe contacts between triangles.
- Use canonical order to compute drawing.



Representation with dissection of a rectangle, called **rectangular dual**:

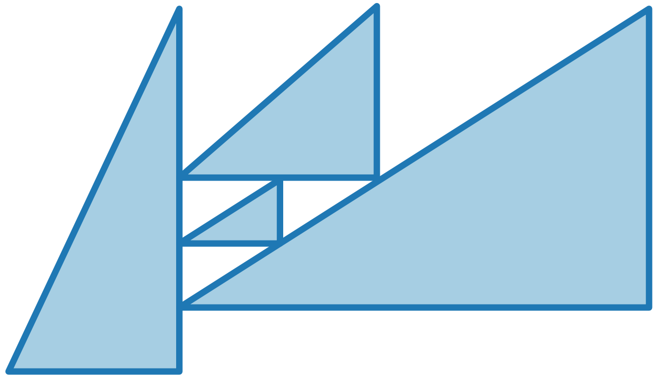
- Find a description similar to a Schnyder realizer for rectangles.



This Lecture

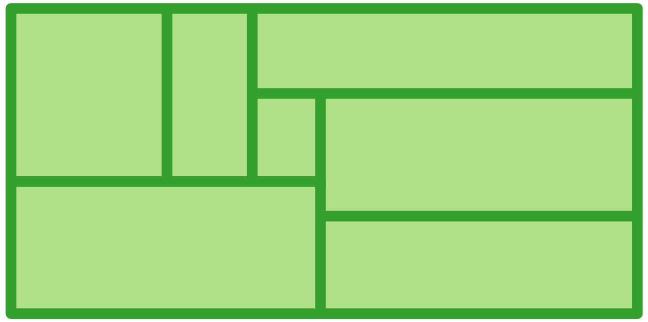
Representation with right-triangles and corner contact:

- Use Schnyder realizer to describe contacts between triangles.
- Use canonical order to compute drawing.



Representation with dissection of a rectangle, called **rectangular dual**:

- Find a description similar to a Schnyder realizer for rectangles.
- Construct drawing via st-digraphs, duals, and topological sorting.



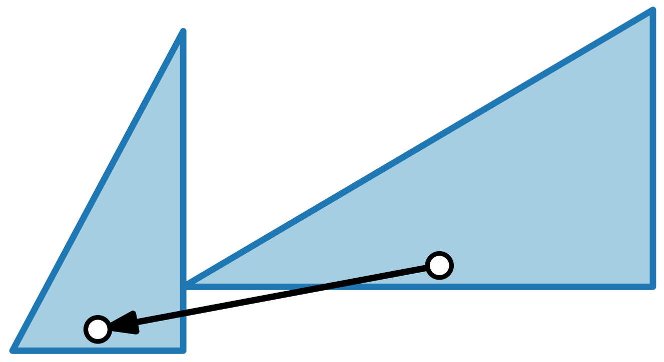
Triangle Corner Contact Representation

Main Idea.
Use canonical order and Schnyder realizer to find coordinates for triangles.

Triangle Corner Contact Representation

Main Idea.

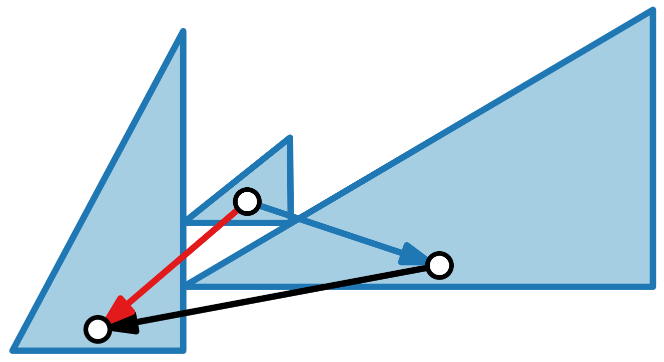
Use canonical order and Schnyder realizer to find coordinates for triangles.



Triangle Corner Contact Representation

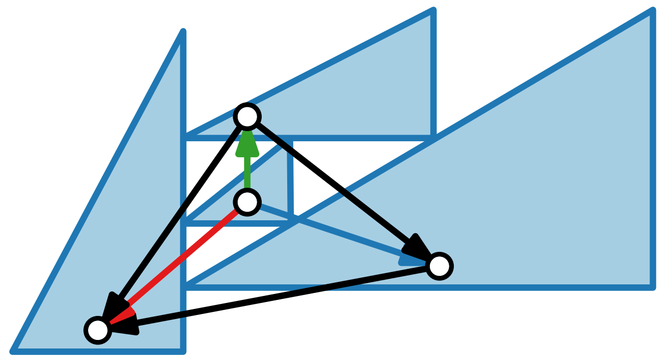
Main Idea.

Use canonical order and Schnyder realizer to find coordinates for triangles.

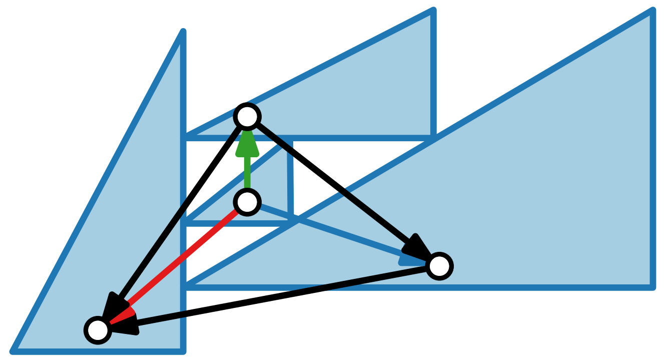


Triangle Corner Contact Representation

Main Idea.
Use canonical order and Schnyder realizer to find coordinates for triangles.



Triangle Corner Contact Representation



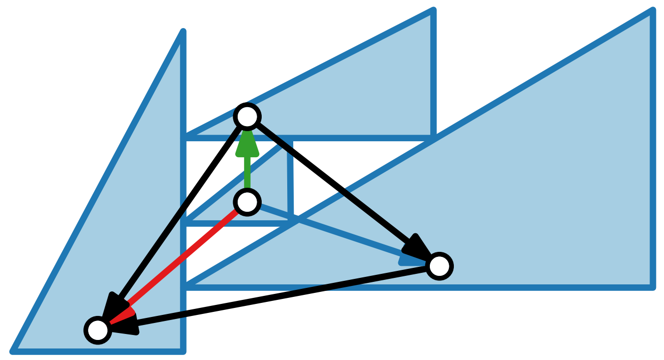
Main Idea.

Use canonical order and Schnyder realizer to find coordinates for triangles.

Detailed Idea.

- Place base of triangle at height equal to position in canonical order.

Triangle Corner Contact Representation



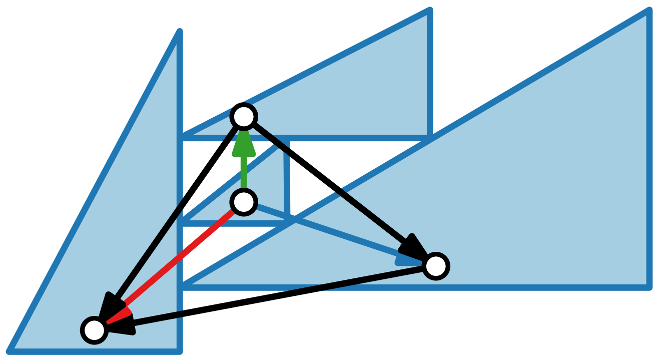
Main Idea.

Use canonical order and Schnyder realizer to find coordinates for triangles.

Detailed Idea.

- Place base of triangle at height equal to position in canonical order.
- Triangle tip is precisely at base of triangle corresponding to cover neighbor.

Triangle Corner Contact Representation



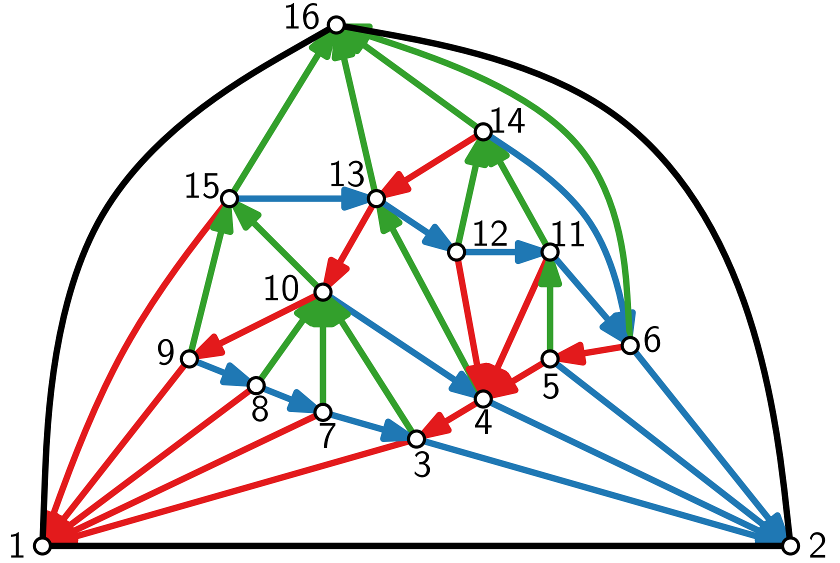
Main Idea.

Use canonical order and Schnyder realizer to find coordinates for triangles.

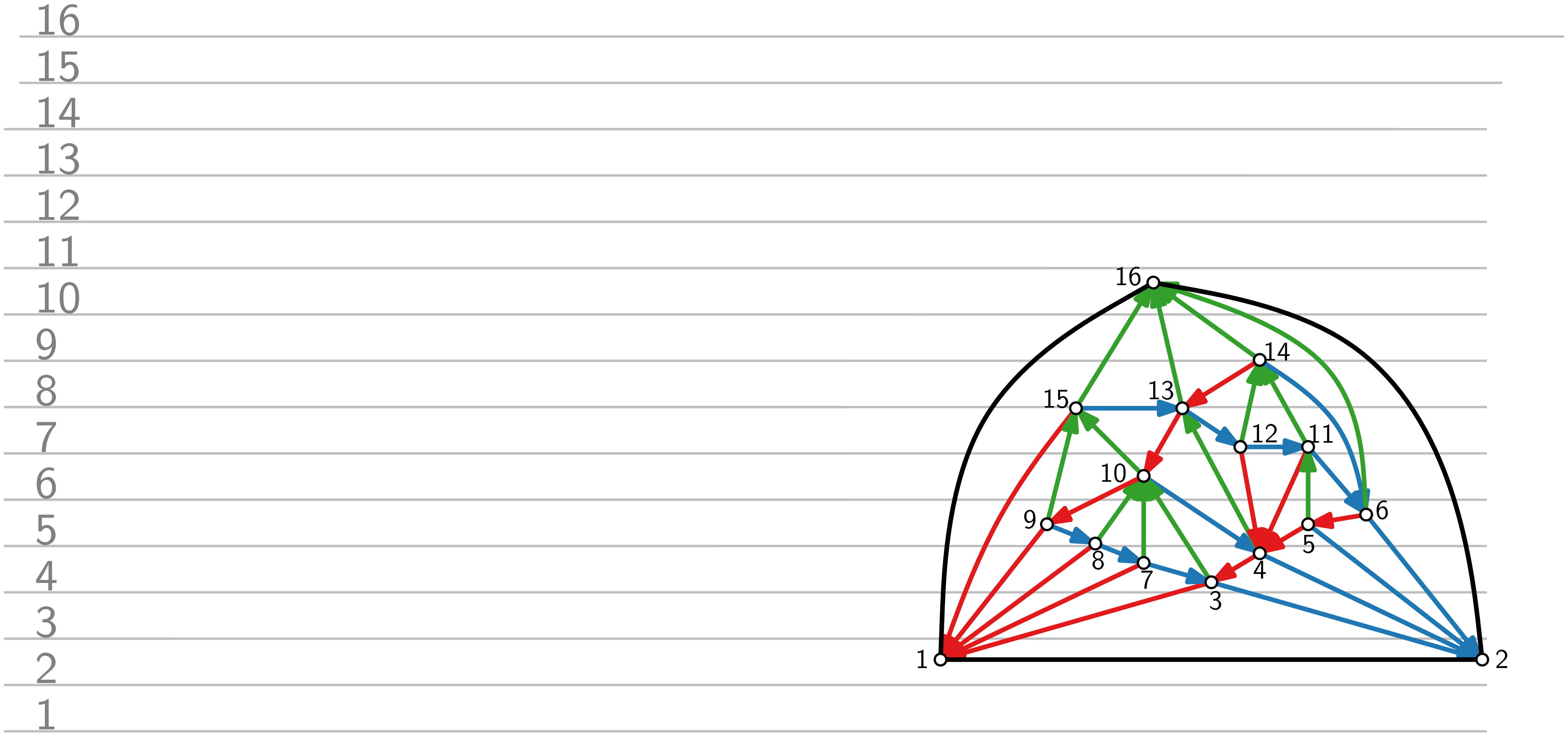
Detailed Idea.

- Place base of triangle at height equal to position in canonical order.
- Triangle tip is precisely at base of triangle corresponding to cover neighbor.
- Outgoing edges in Schnyder forest indicate corner contacts.

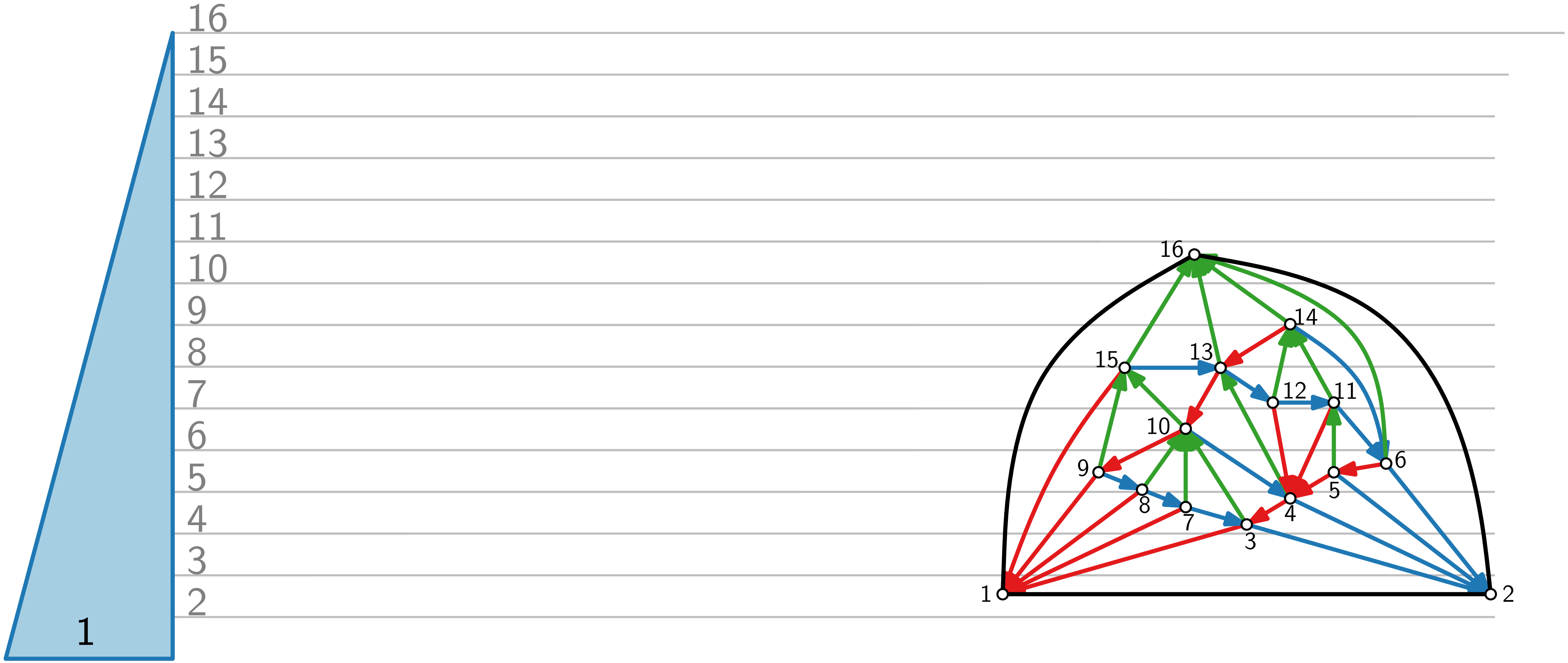
Triangle Contact Representation Example



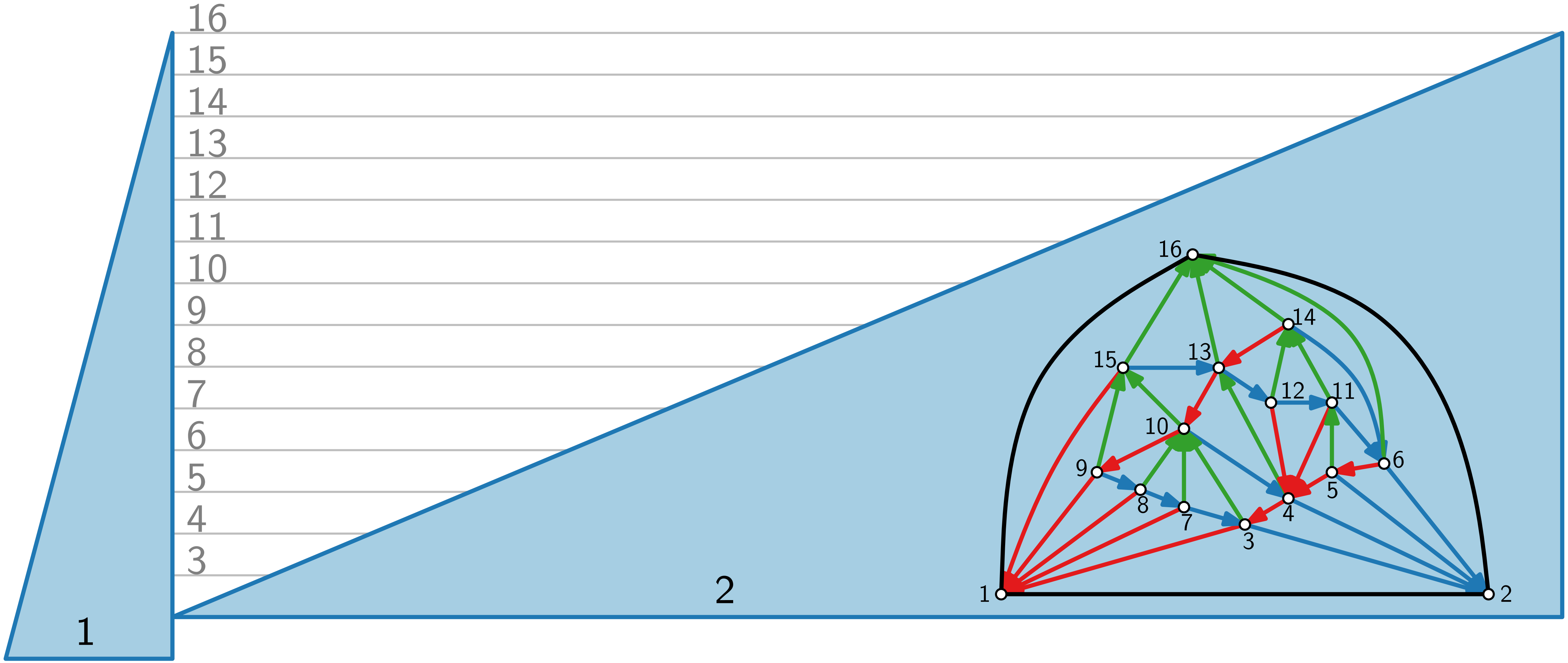
Triangle Contact Representation Example



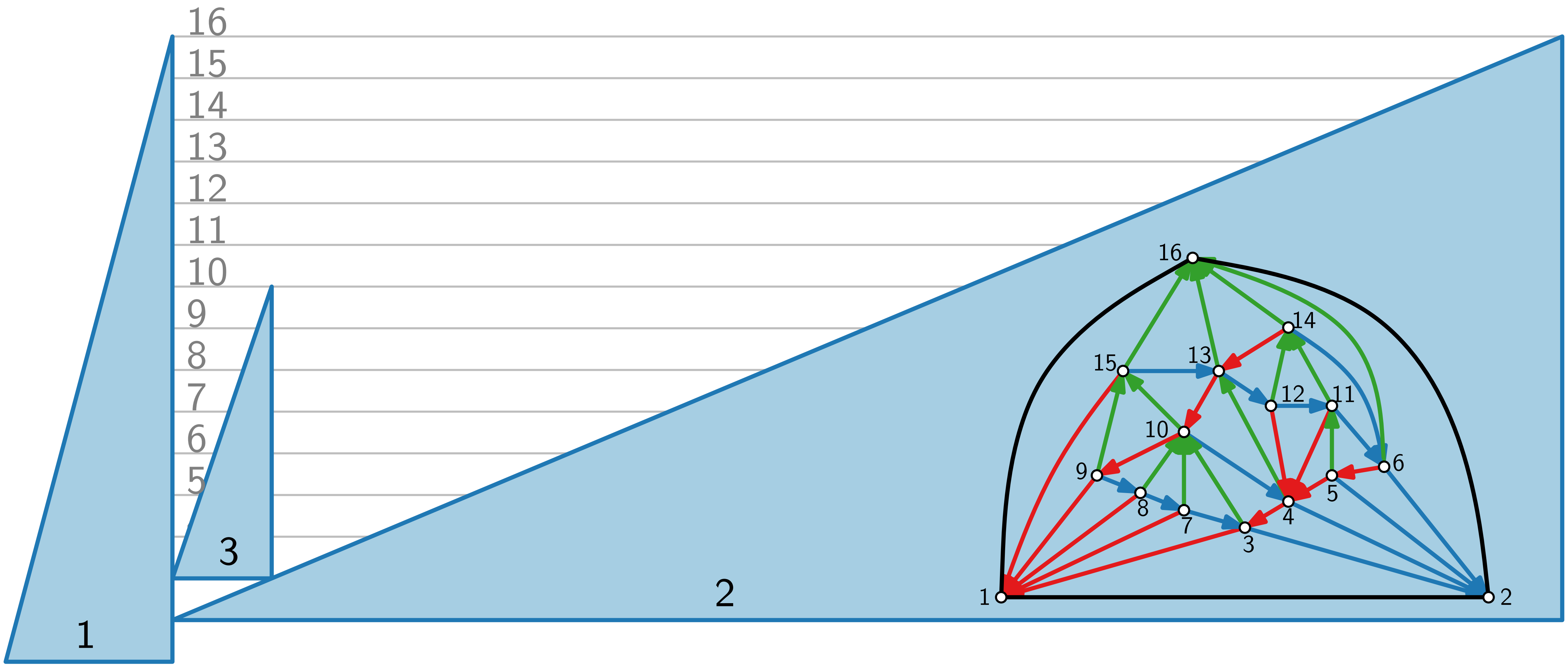
Triangle Contact Representation Example



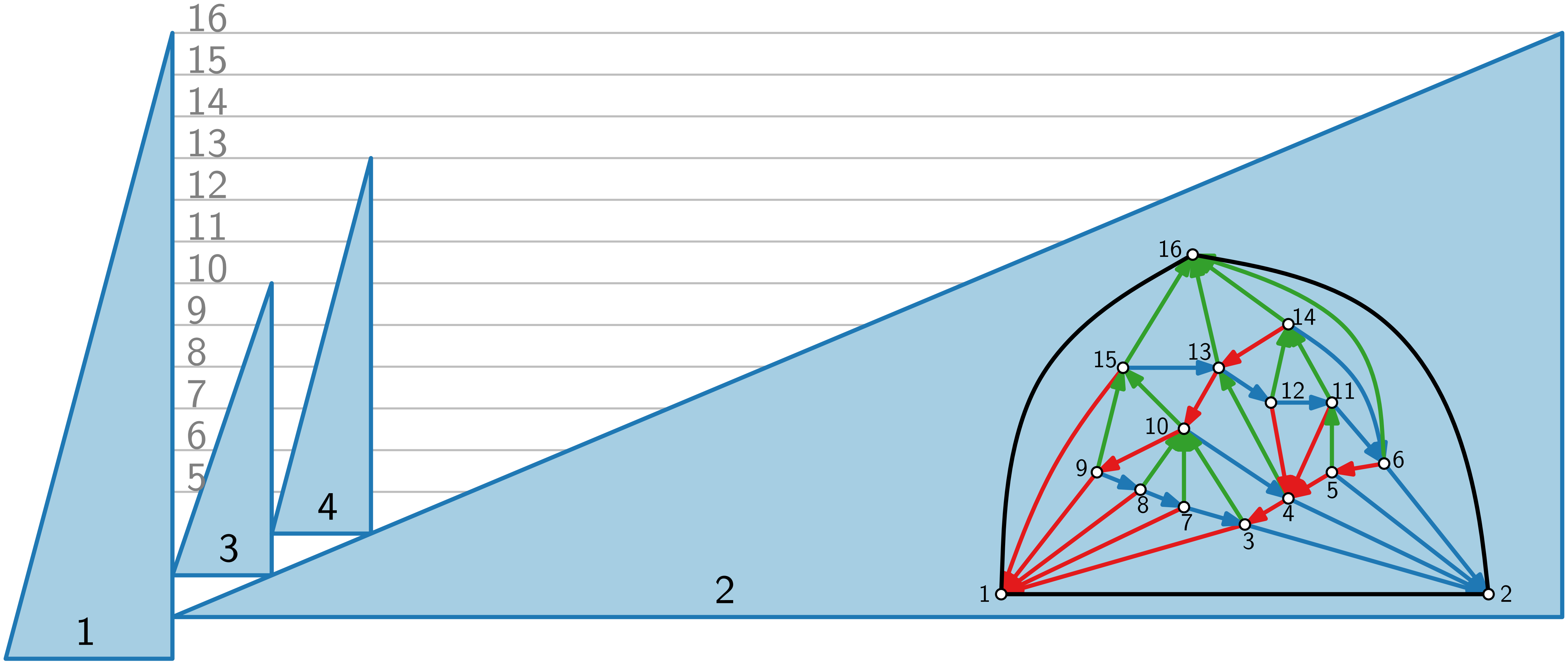
Triangle Contact Representation Example



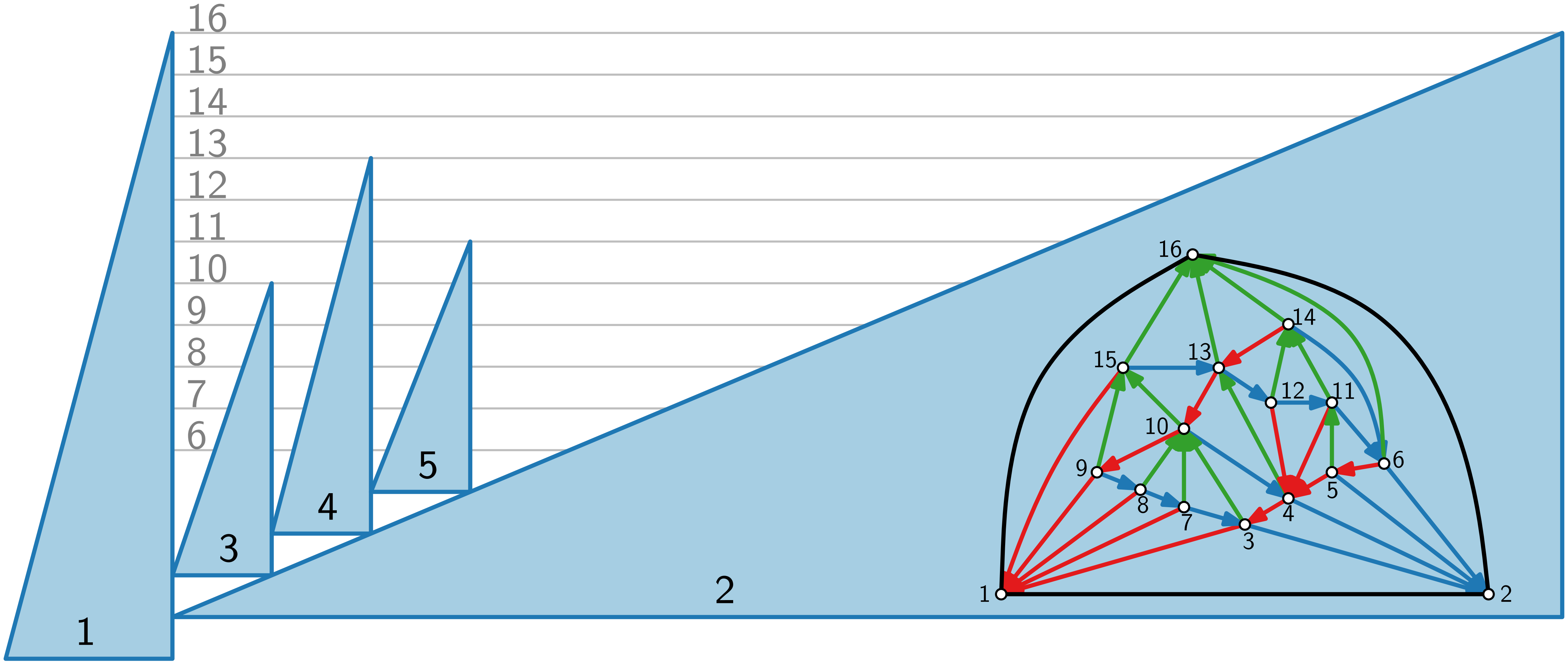
Triangle Contact Representation Example



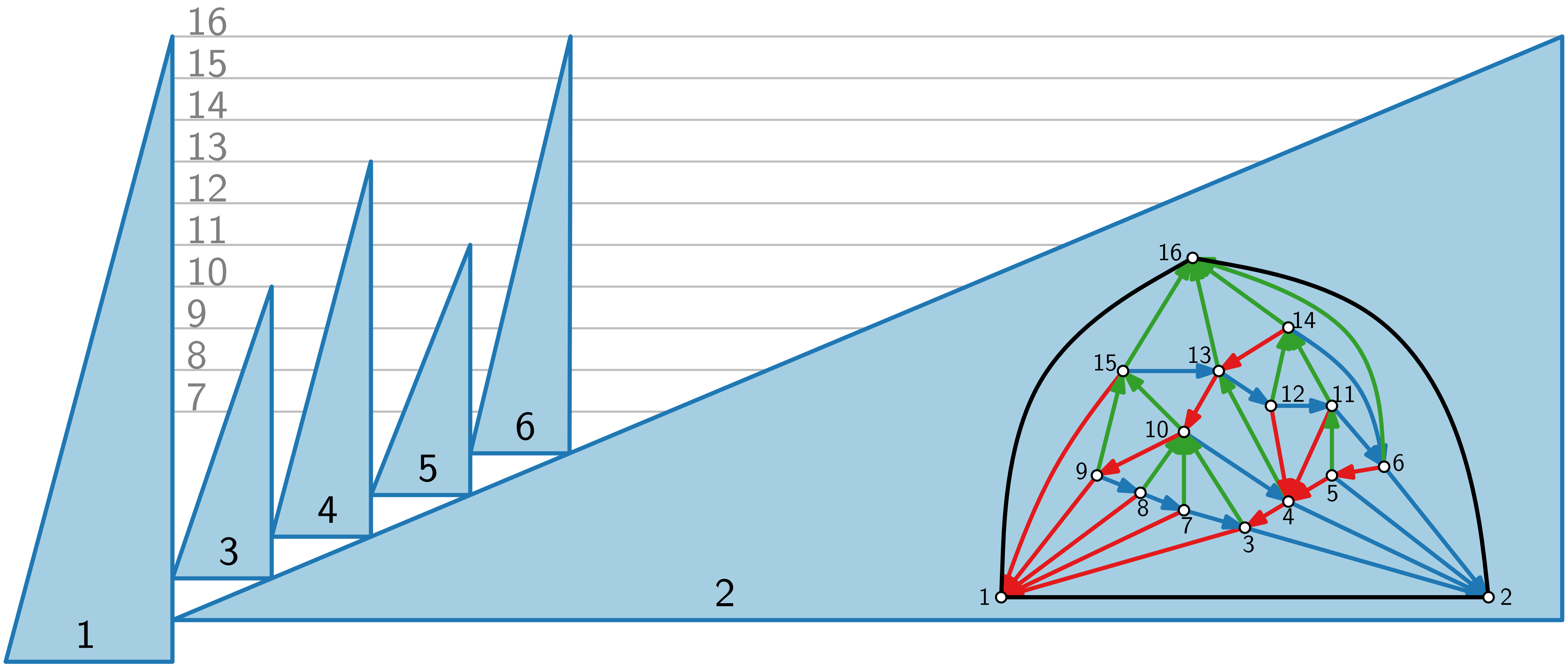
Triangle Contact Representation Example



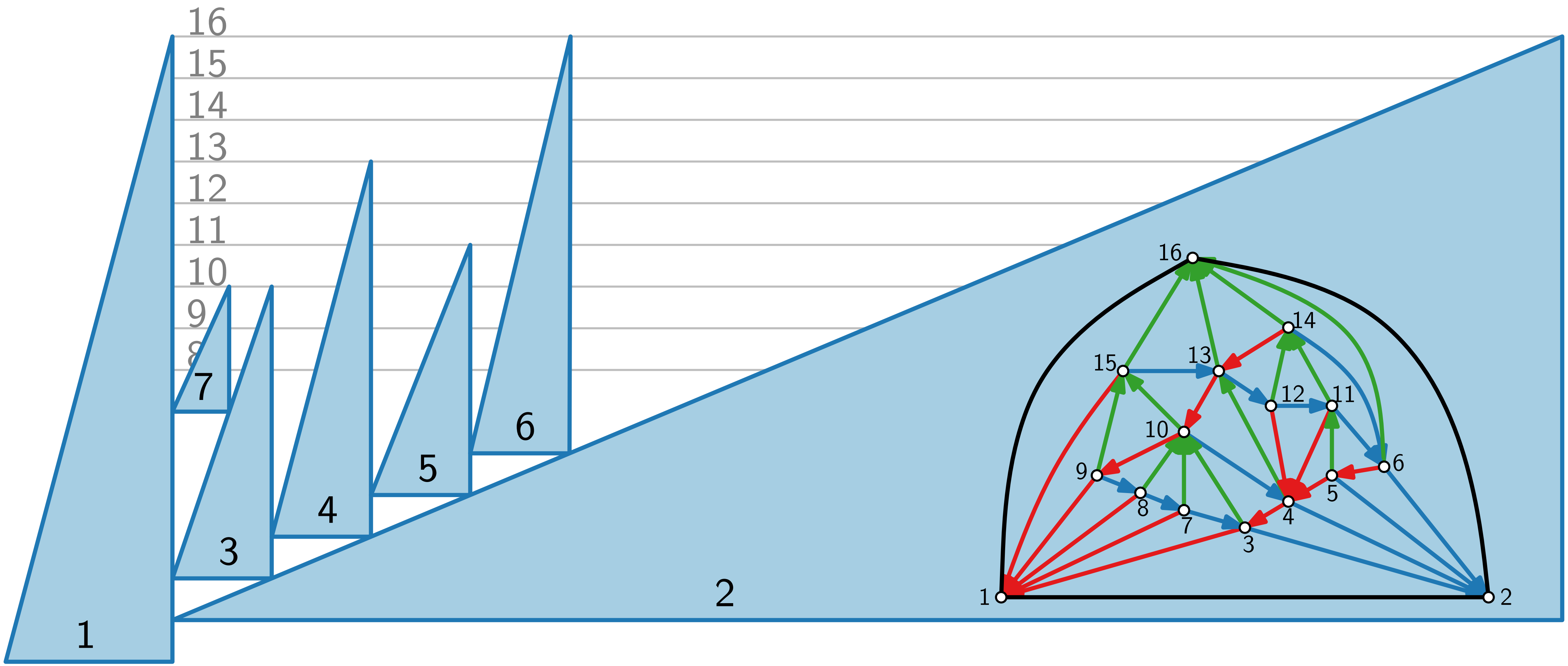
Triangle Contact Representation Example



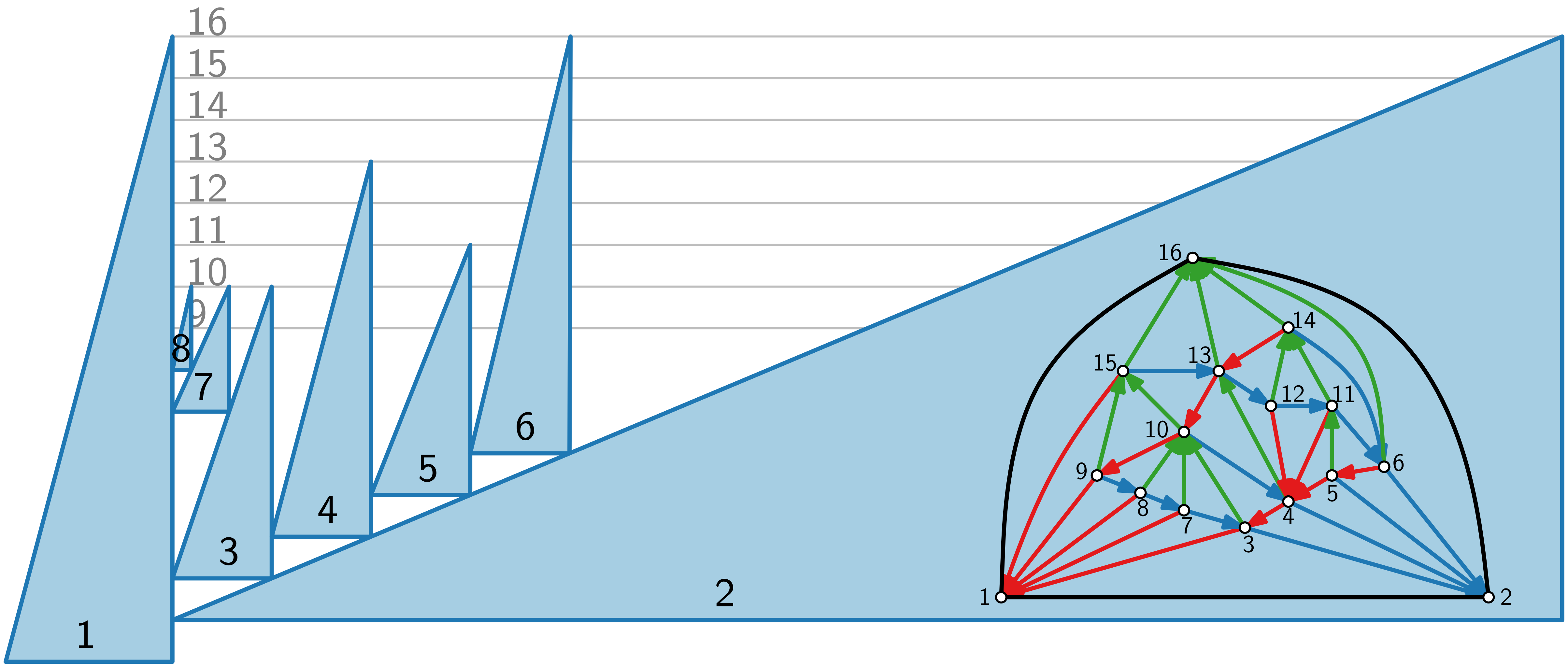
Triangle Contact Representation Example



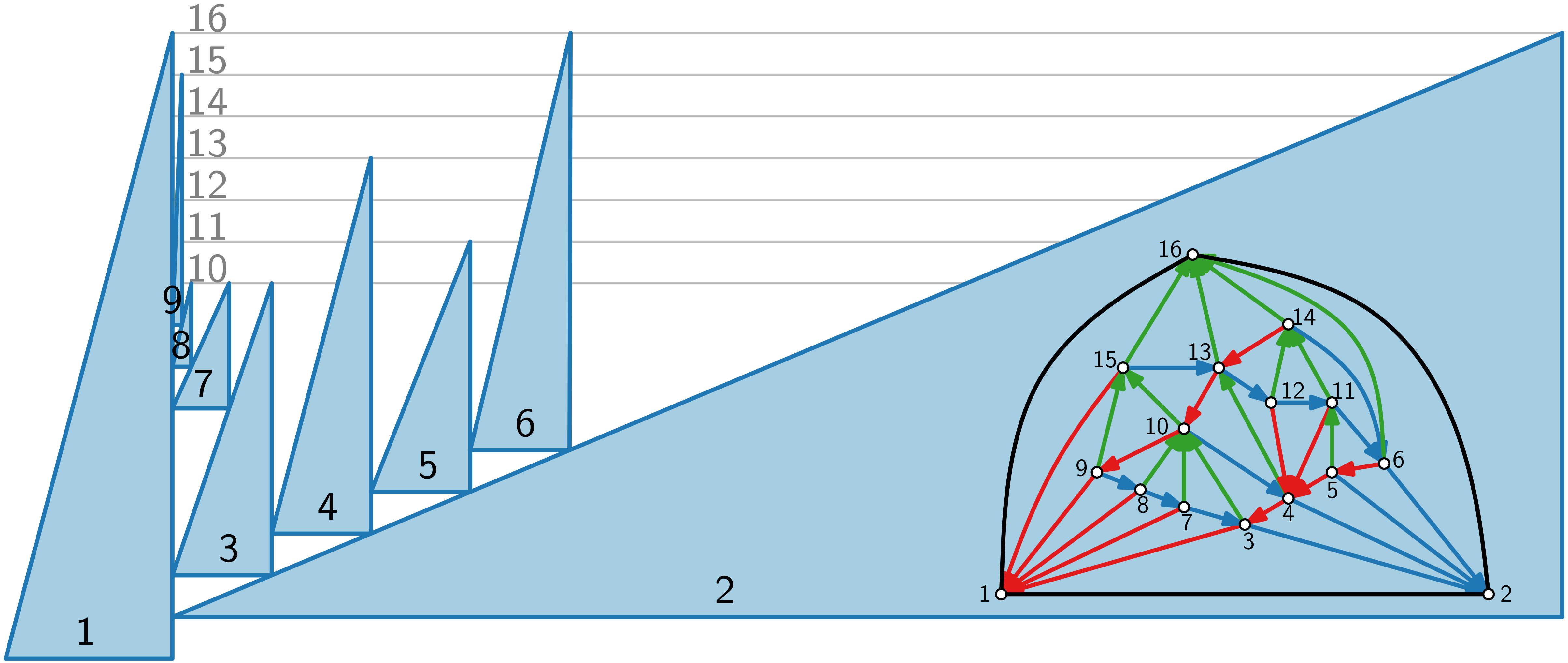
Triangle Contact Representation Example



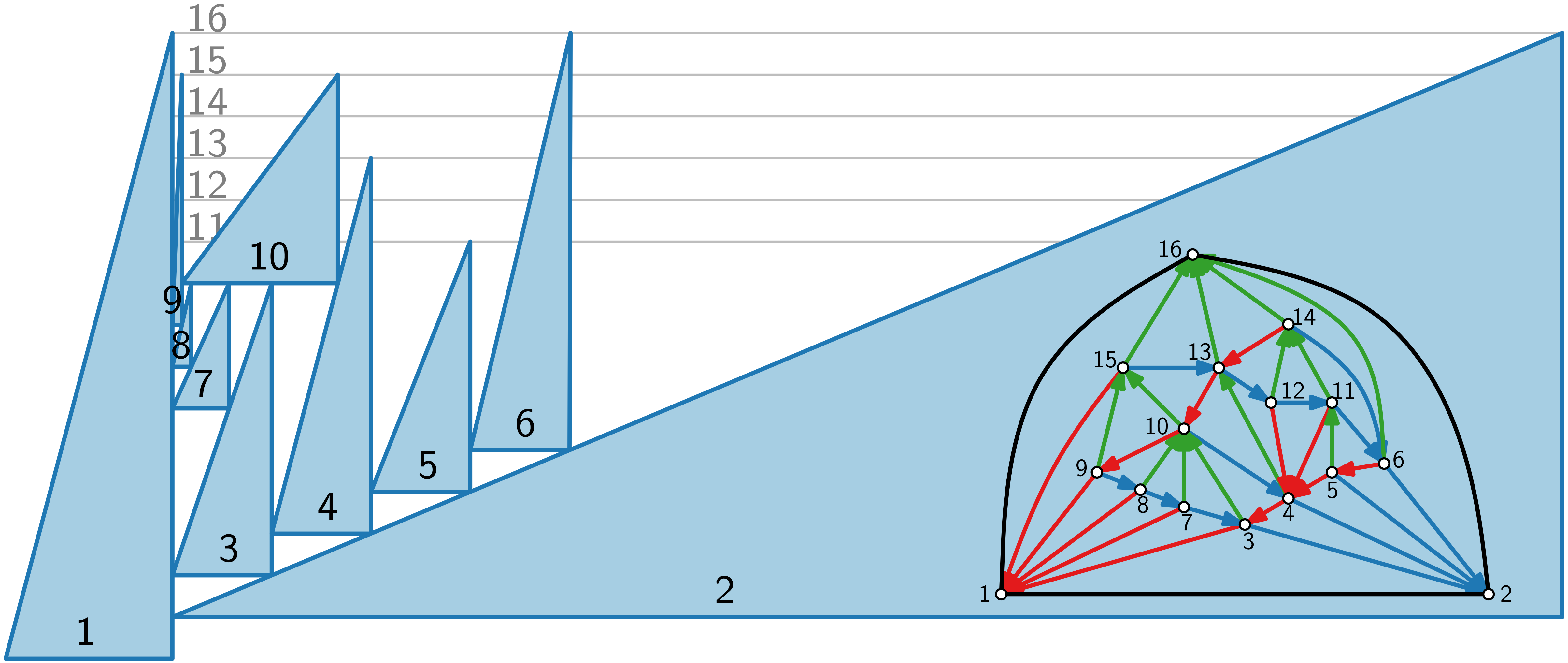
Triangle Contact Representation Example



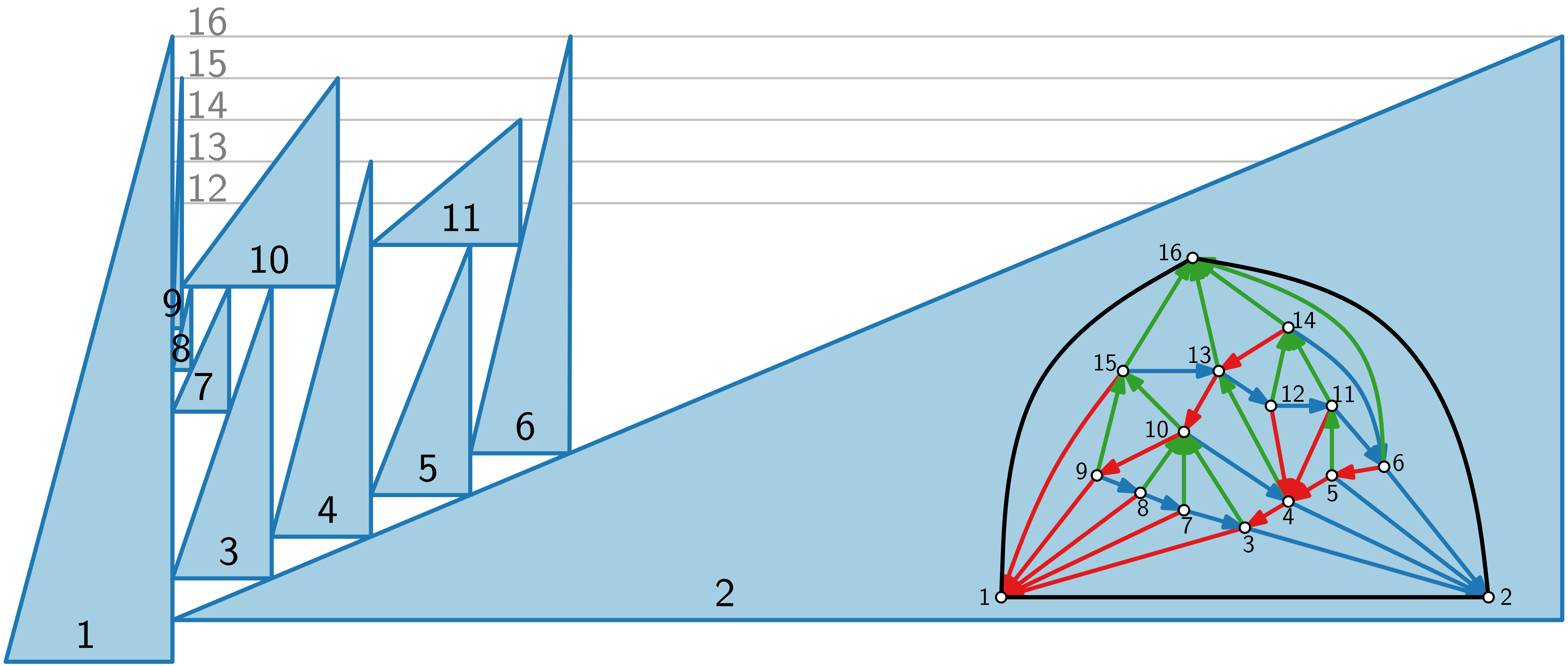
Triangle Contact Representation Example



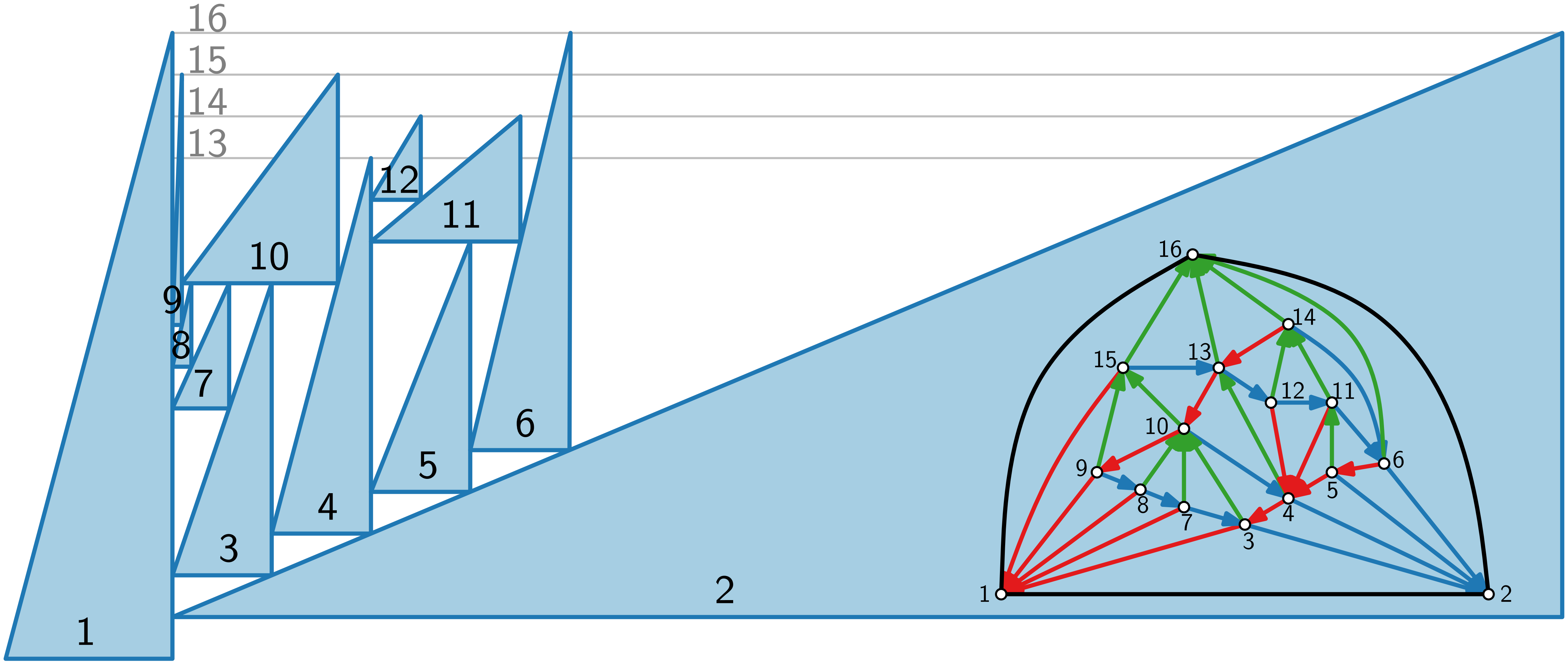
Triangle Contact Representation Example



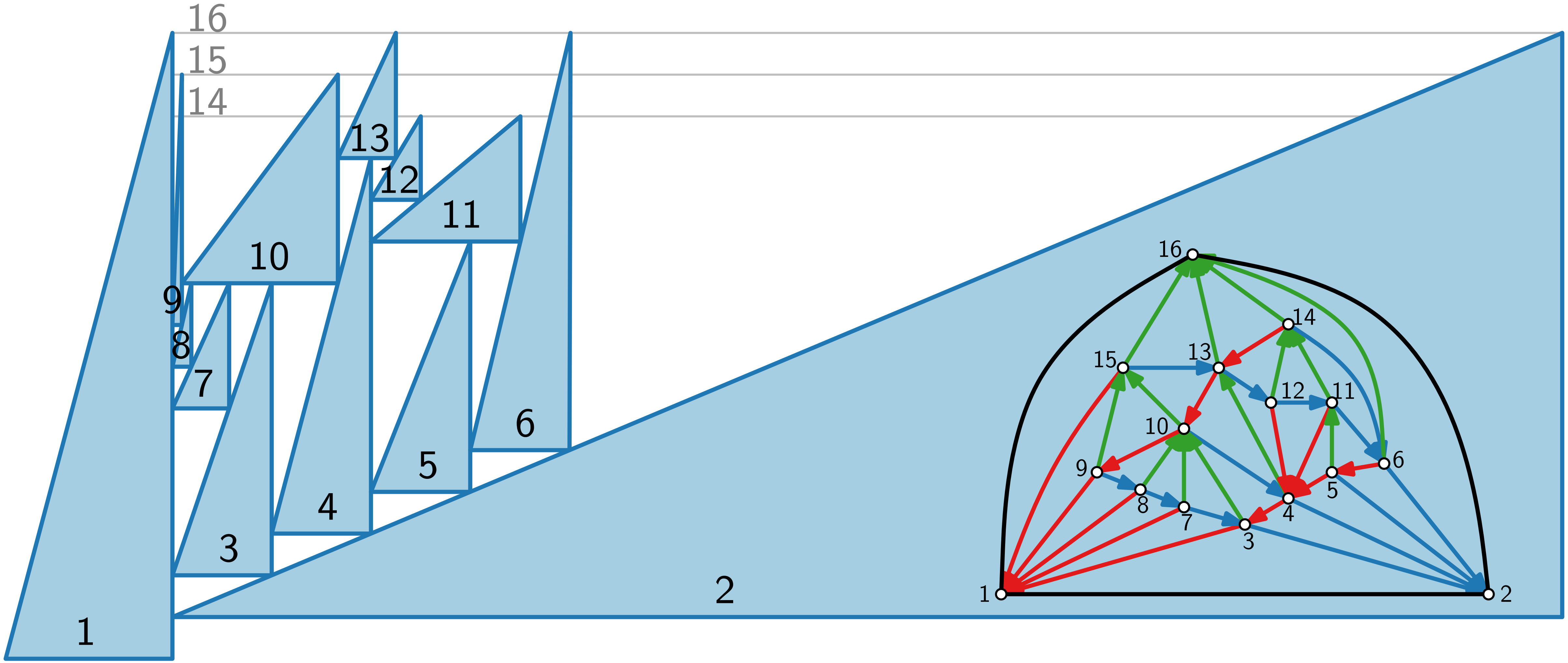
Triangle Contact Representation Example



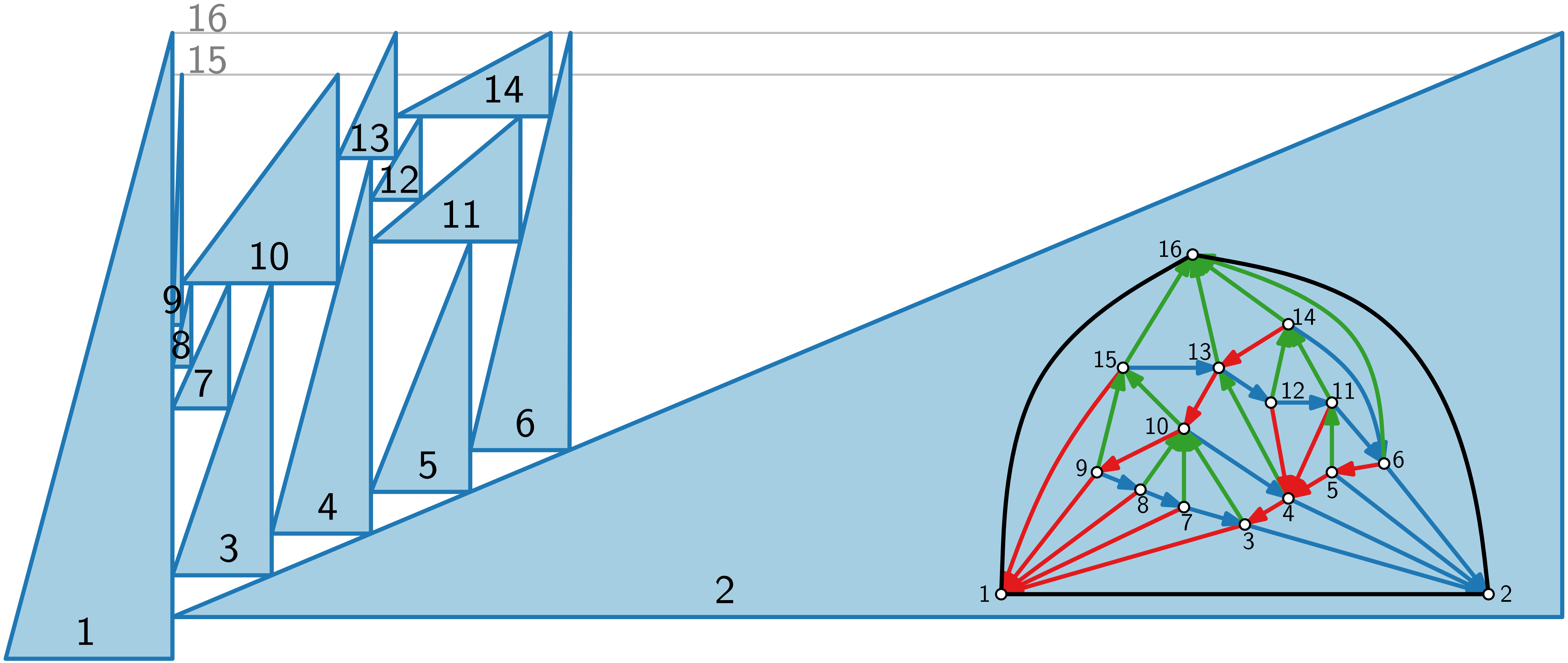
Triangle Contact Representation Example



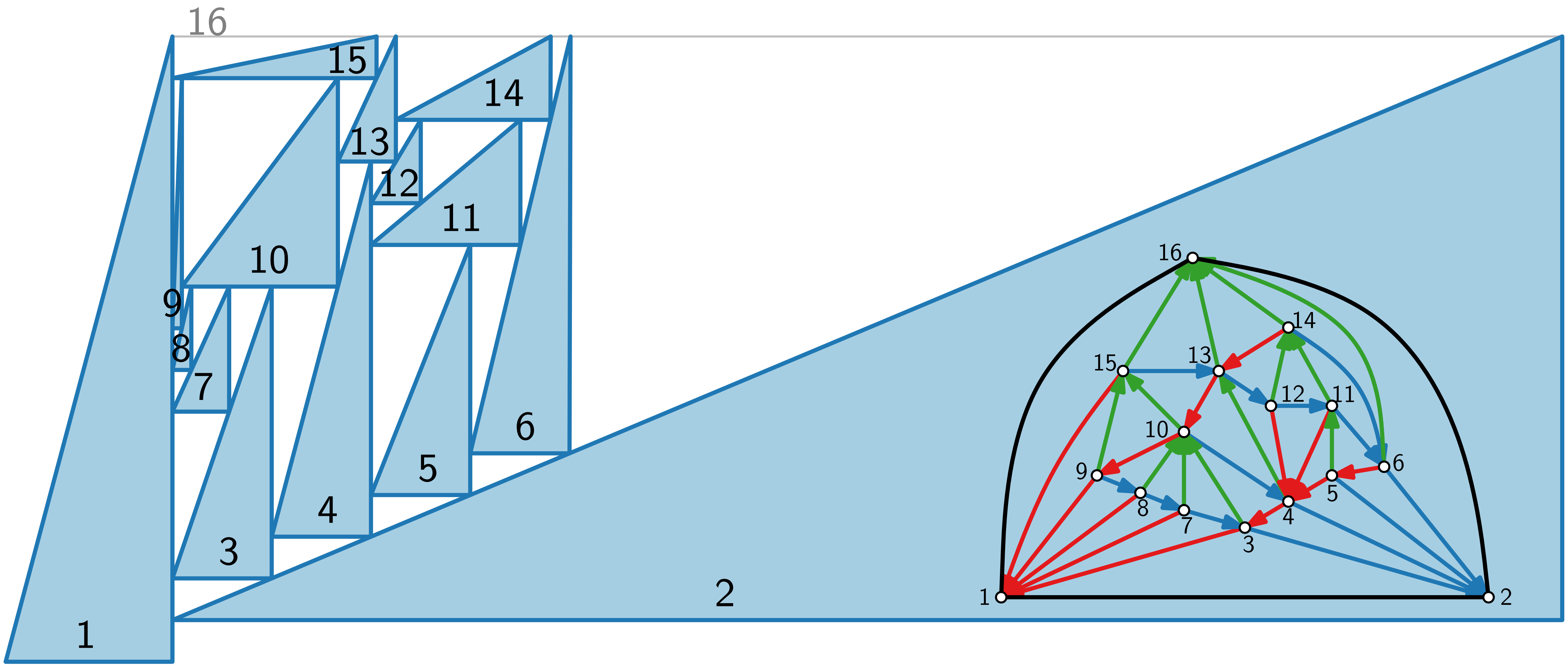
Triangle Contact Representation Example



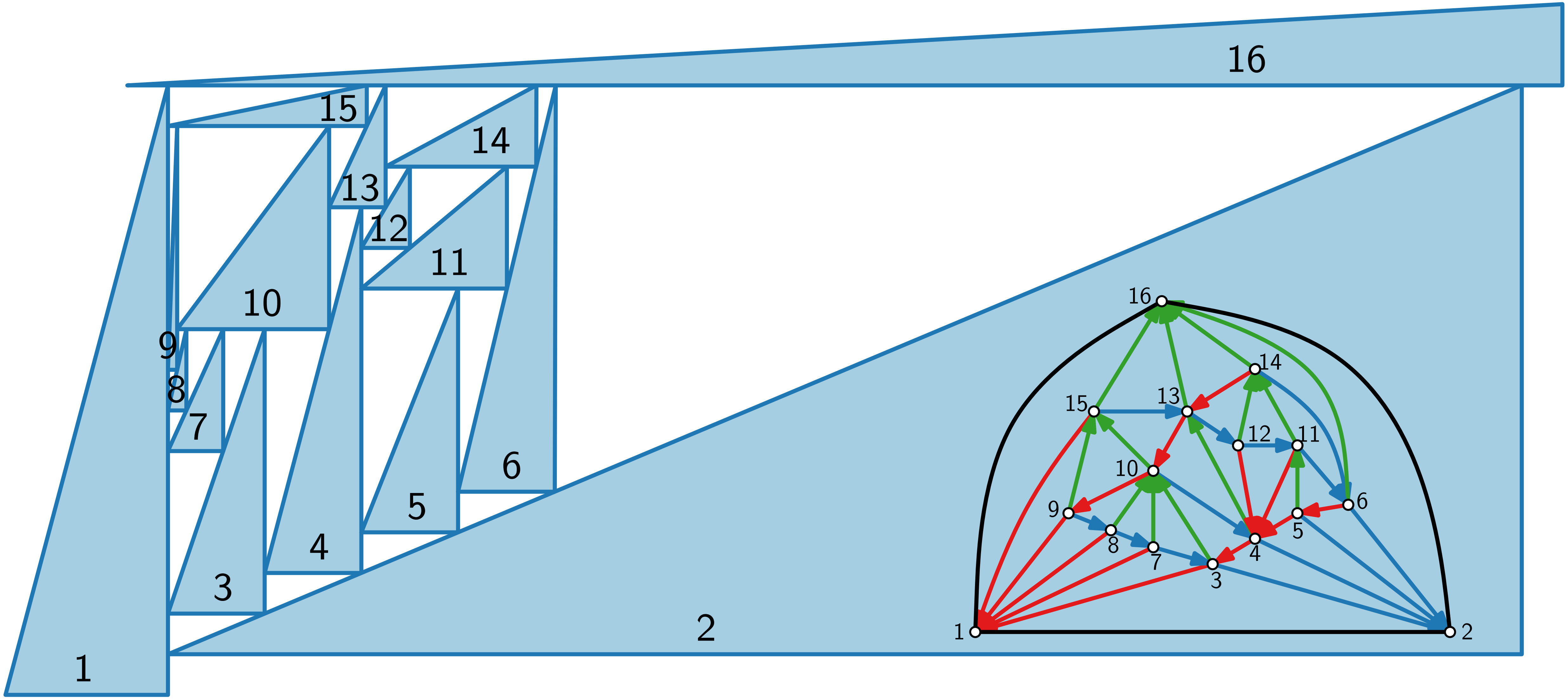
Triangle Contact Representation Example



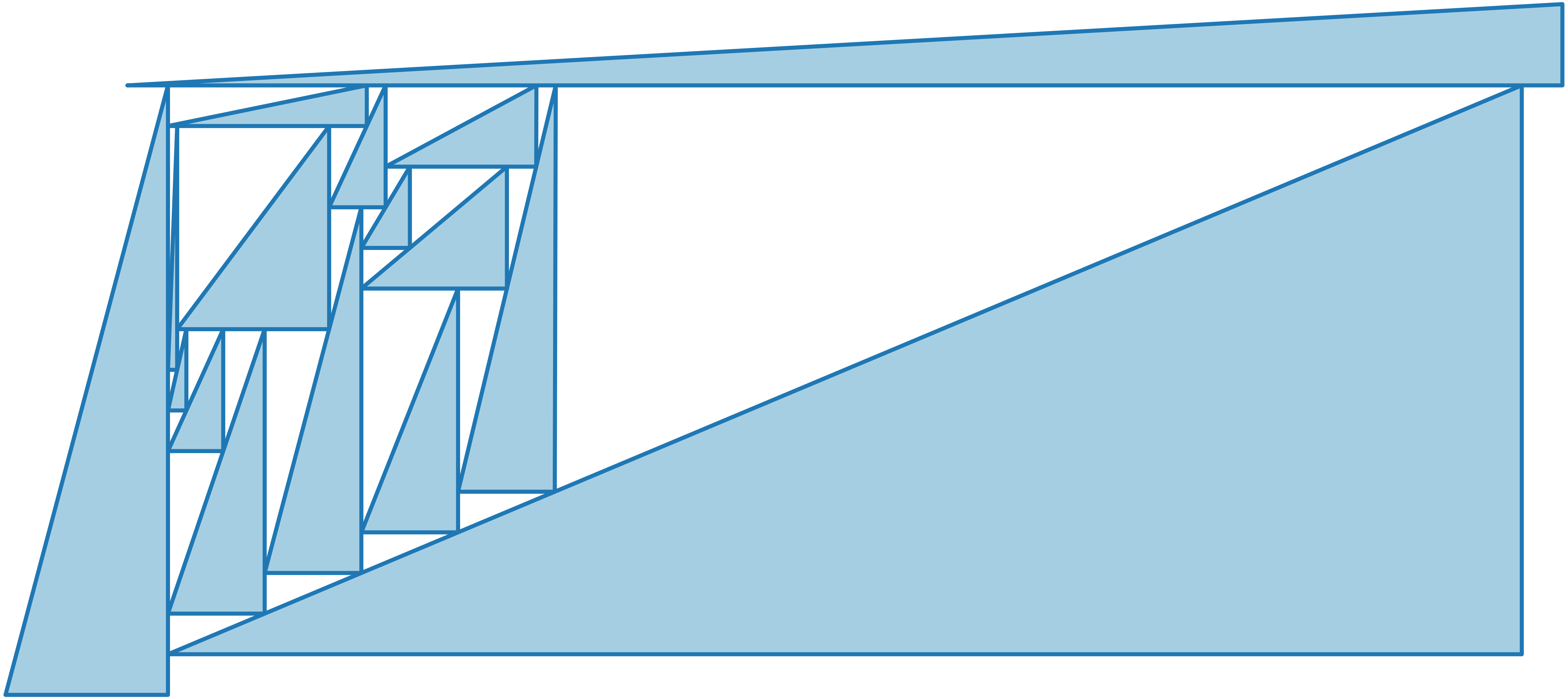
Triangle Contact Representation Example



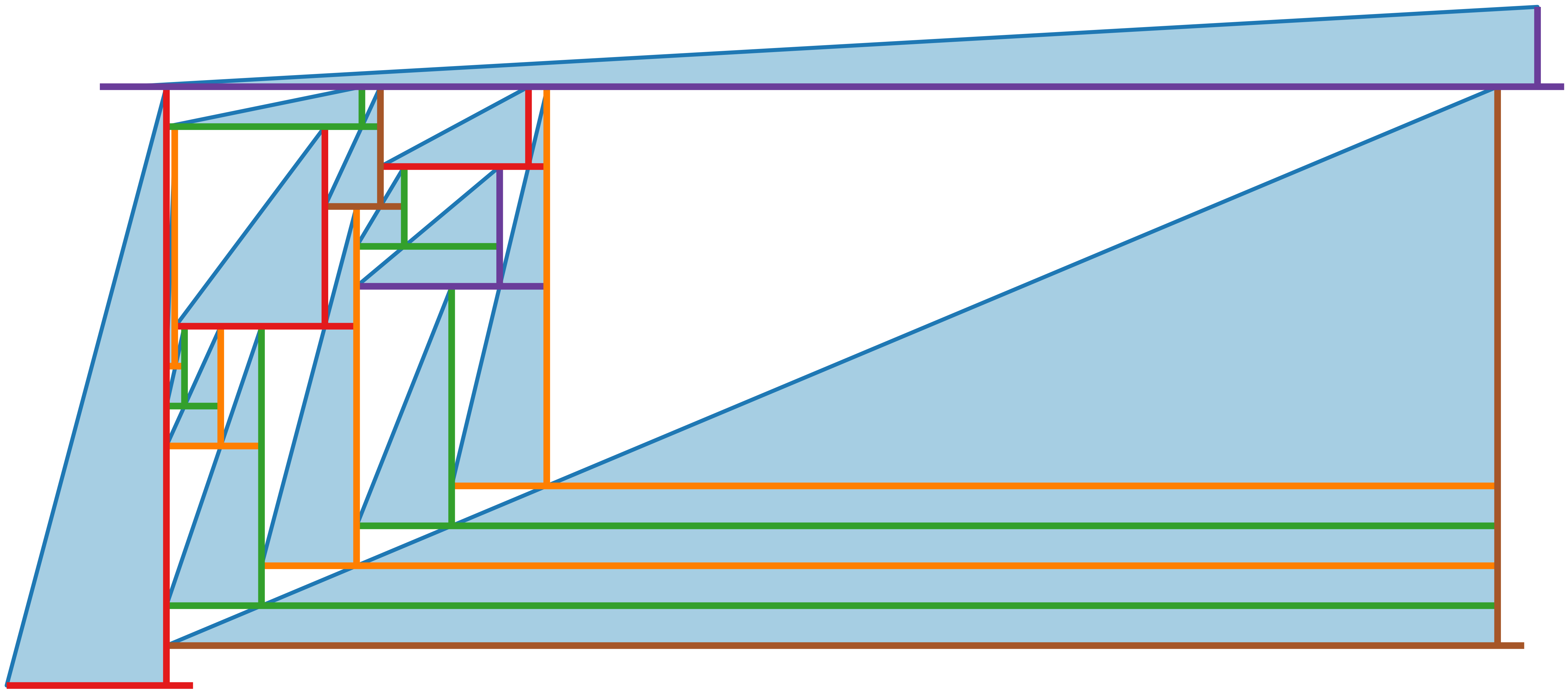
Triangle Contact Representation Example



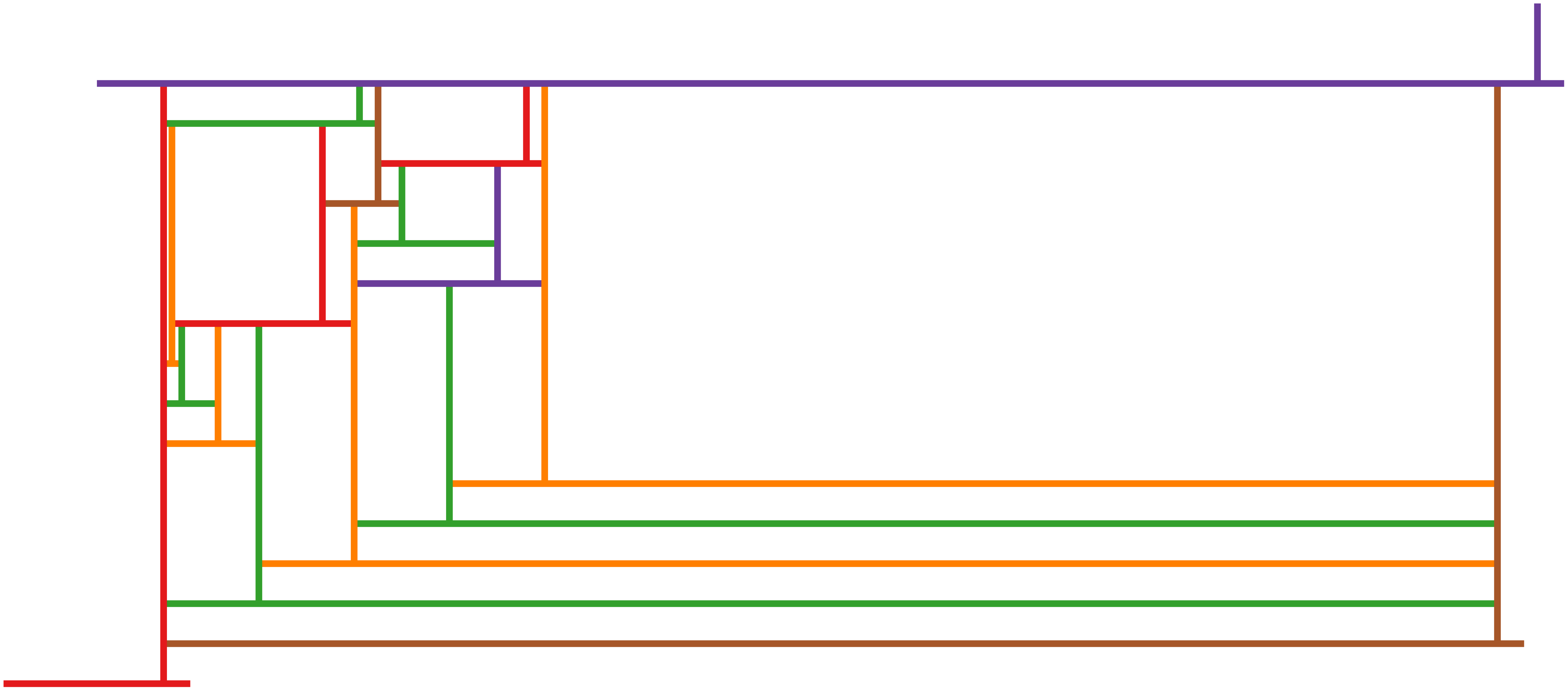
T-shape Contact Representation



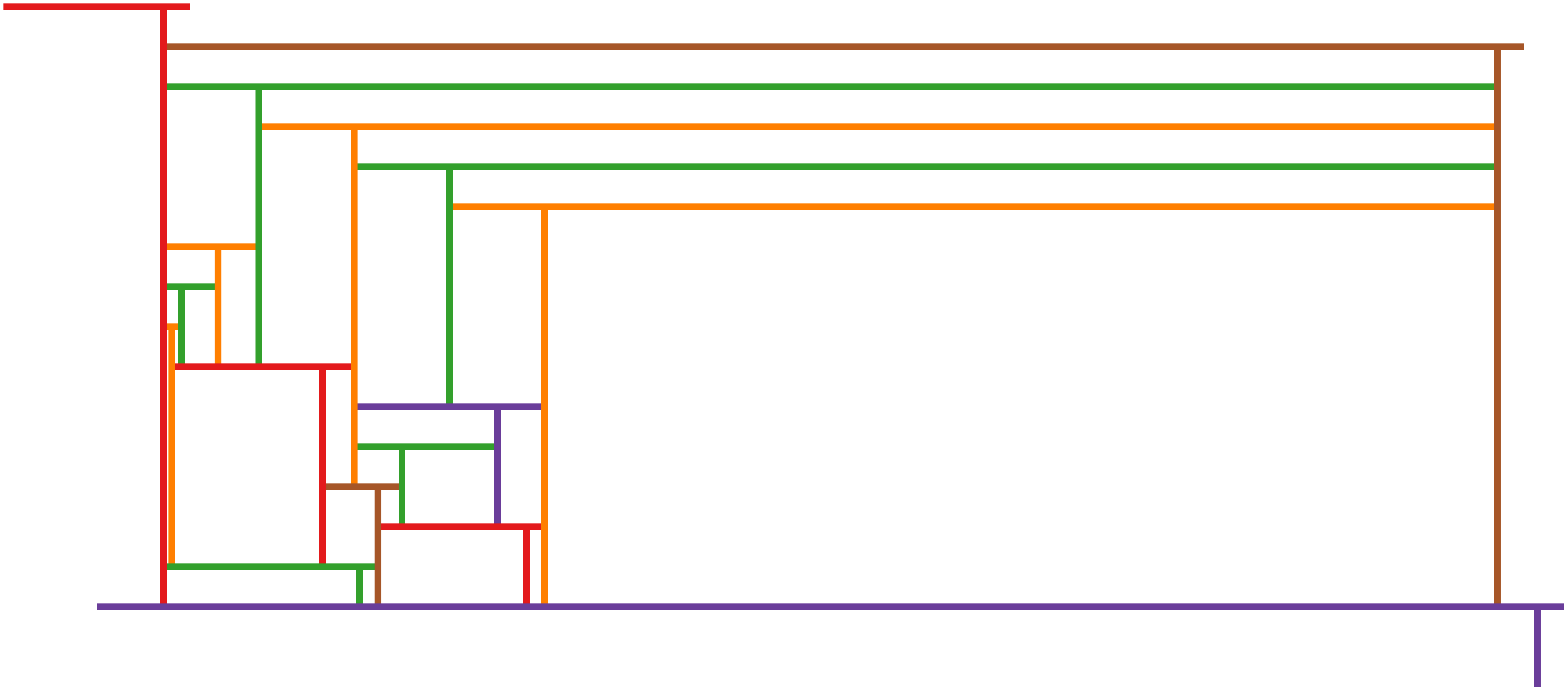
T-shape Contact Representation



T-shape Contact Representation

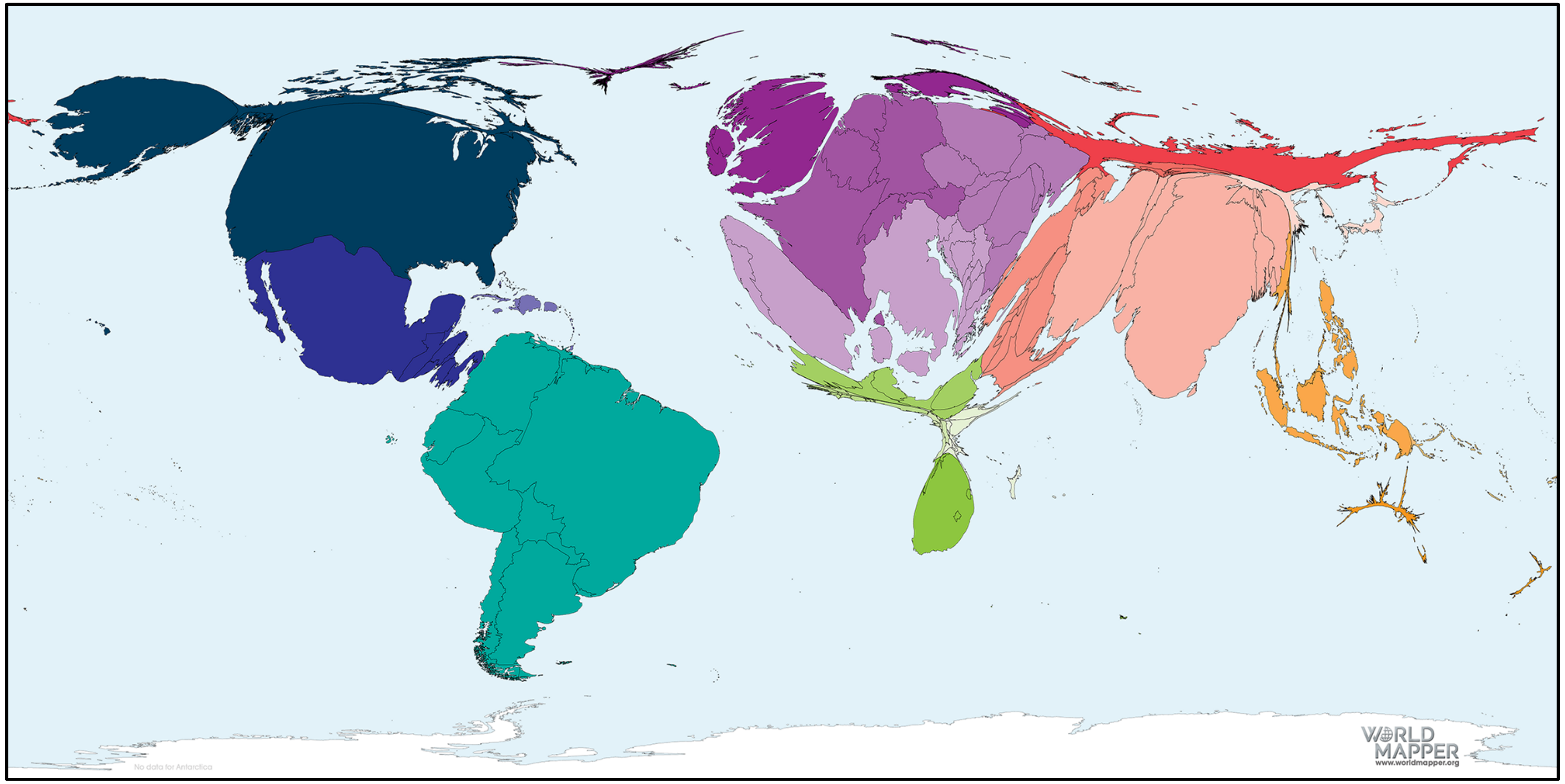


T-shape Contact Representation



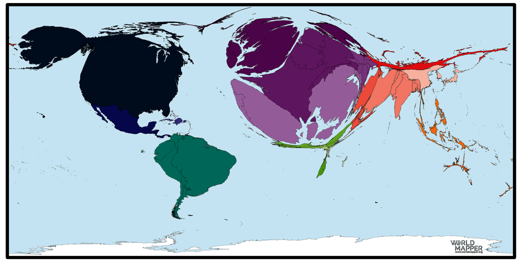
Cartograms

Cartograms

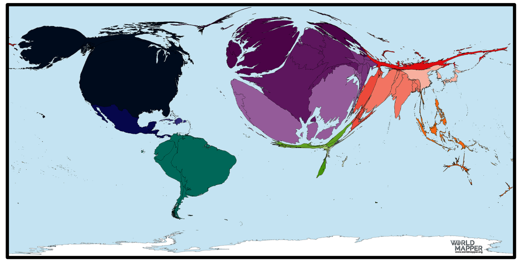


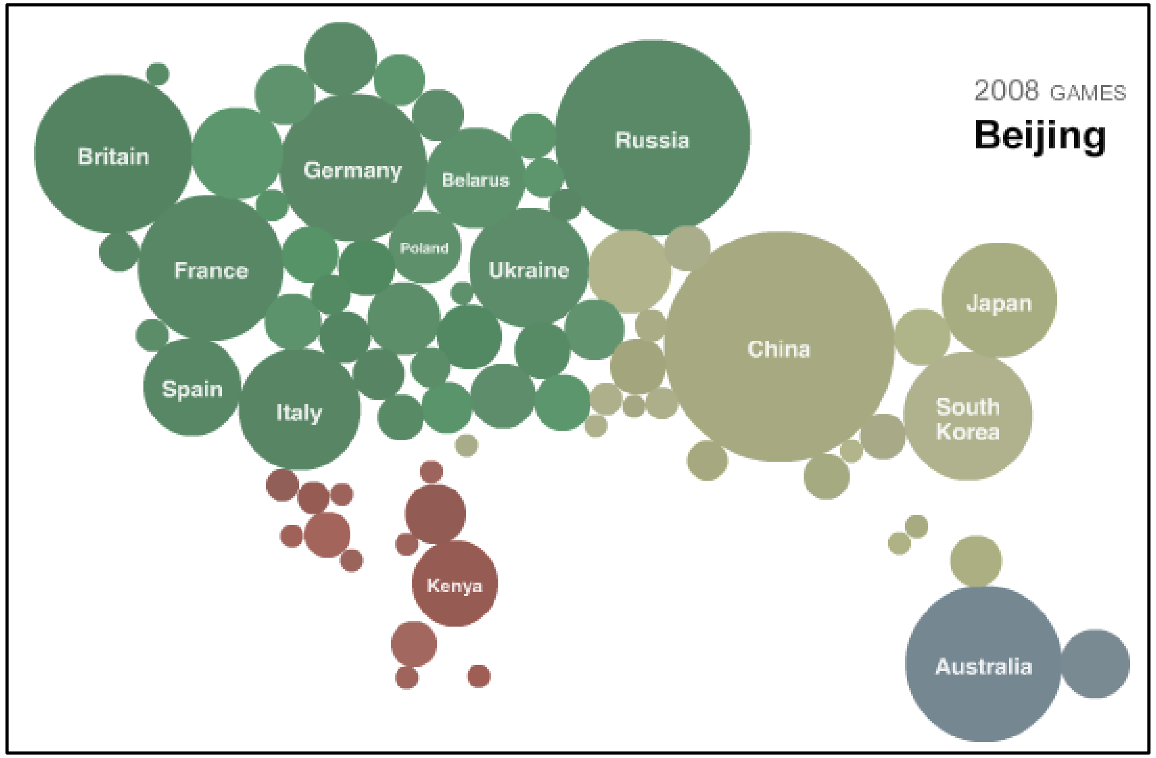
COVID19 reported deaths (January 1, 2021)

Cartograms



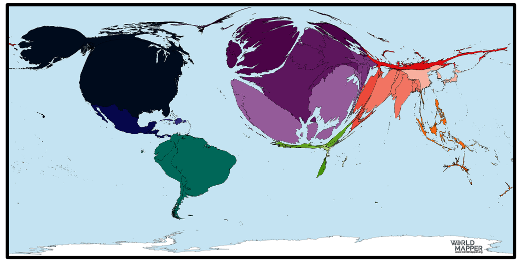
Cartograms



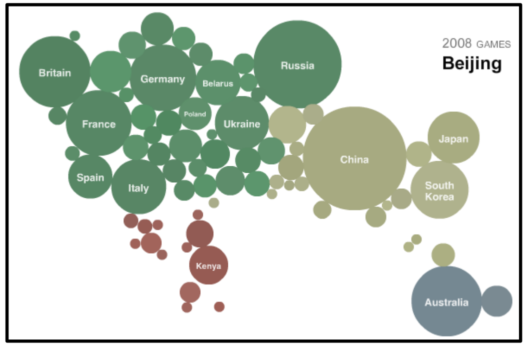


2008 GAMES
Beijing
Britain
Germany
Belarus
Russia
France
Poland
Ukraine
China
Japan
Spain
Italy
South Korea
Kenya
Australia

Cartograms

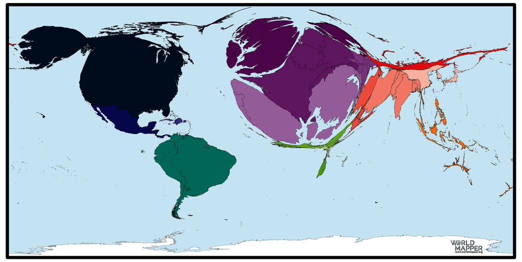


© worldmapper.org

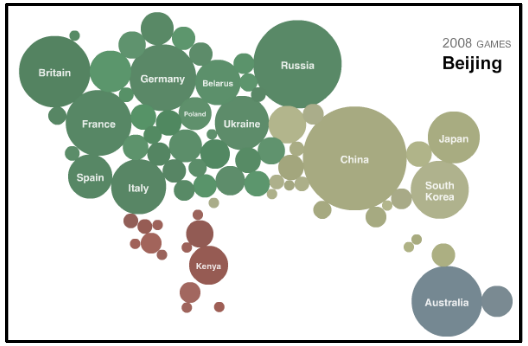


© New York Times

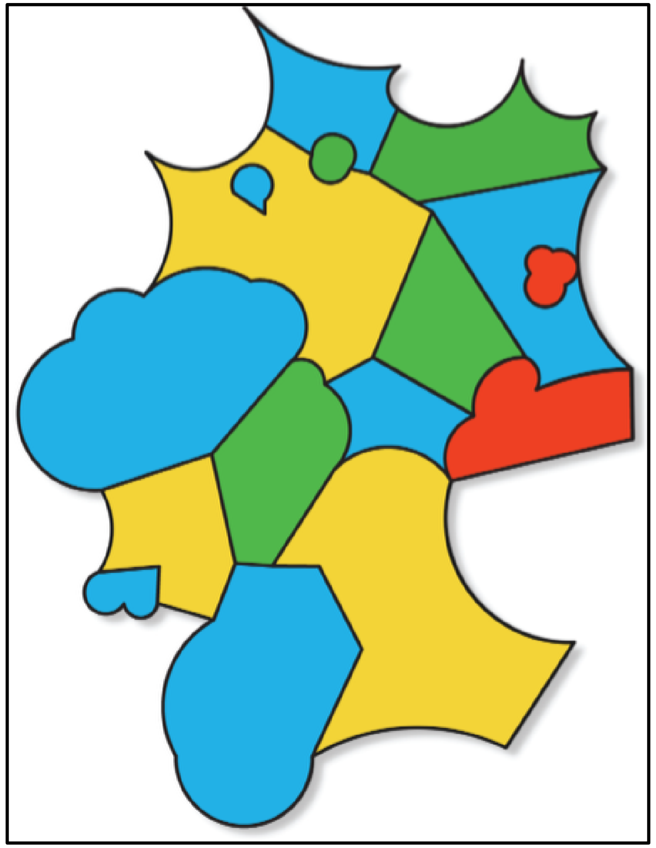
Cartograms



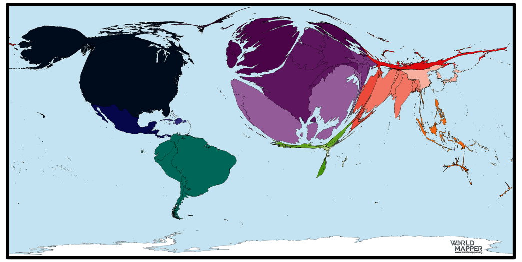
© worldmapper.org



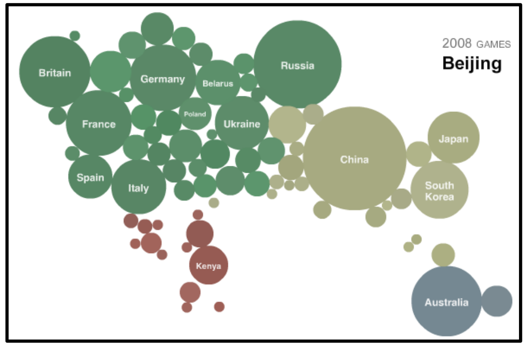
© New York Times



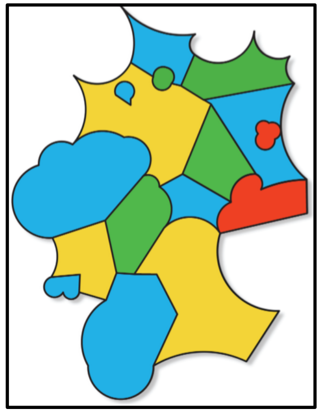
Cartograms



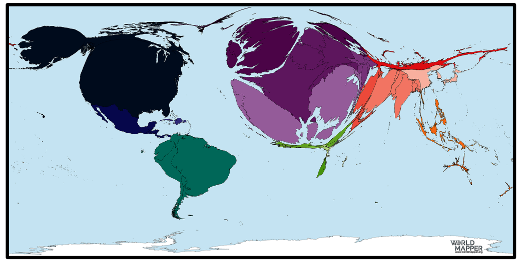
© worldmapper.org



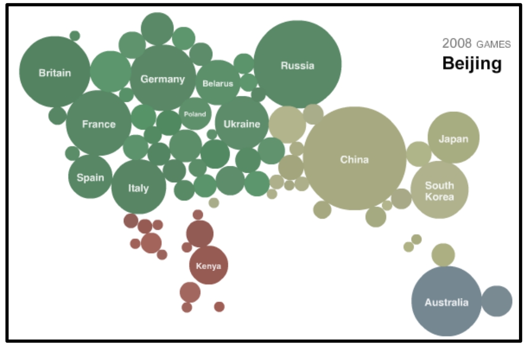
© New York Times



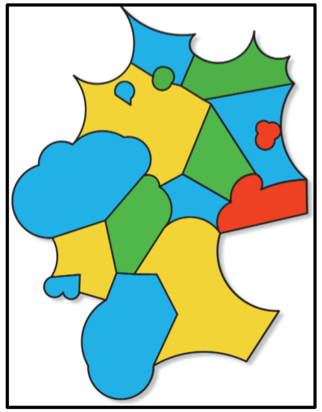
Cartograms

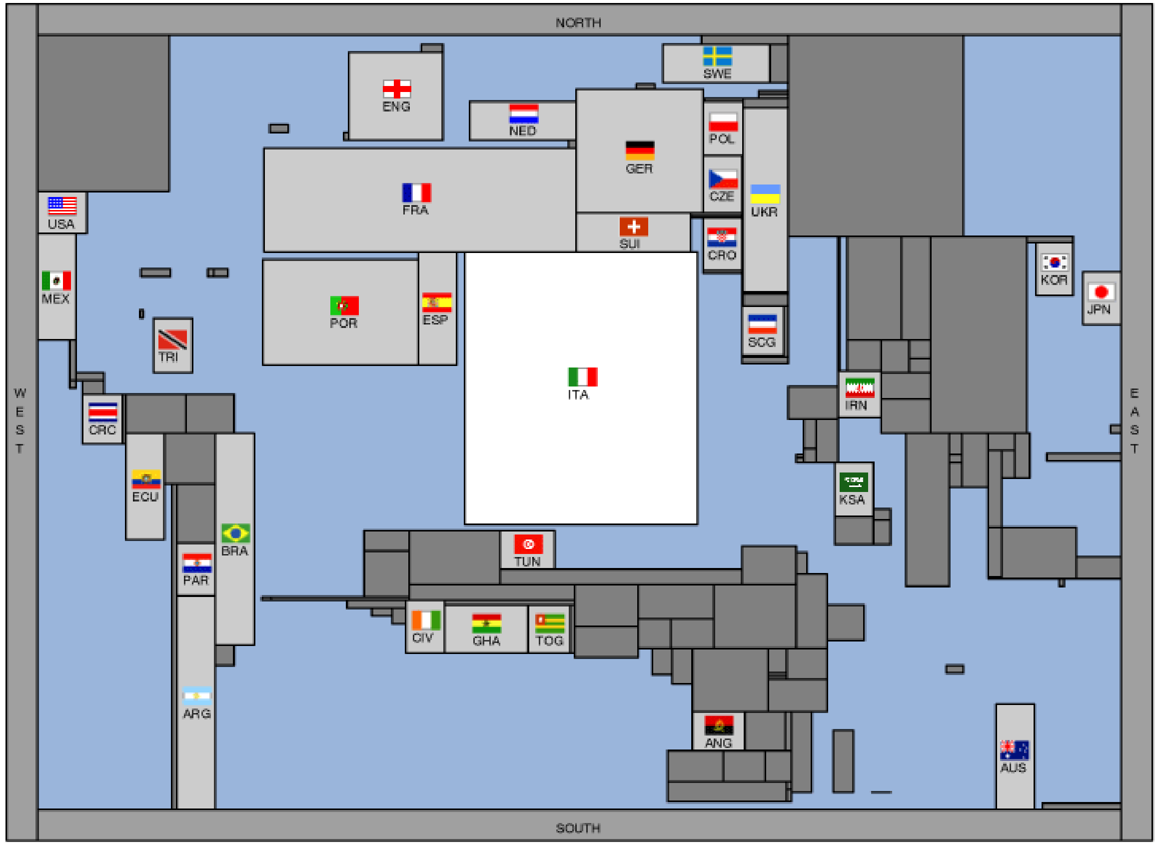


© worldmapper.org

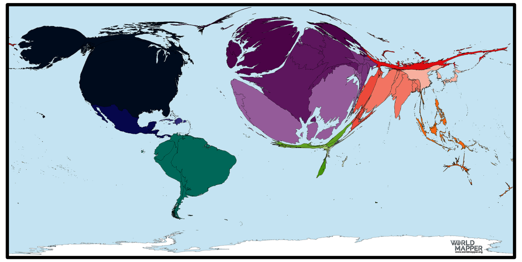


© New York Times

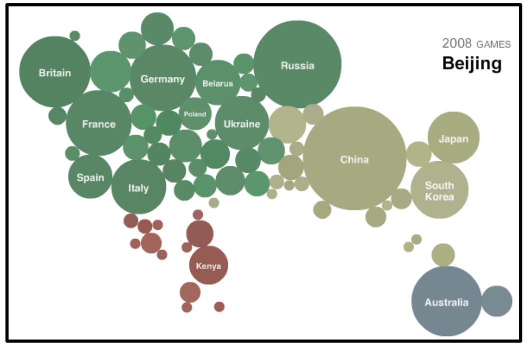




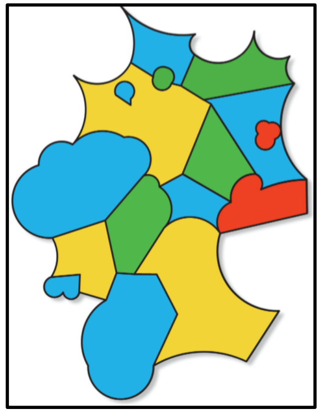
Cartograms

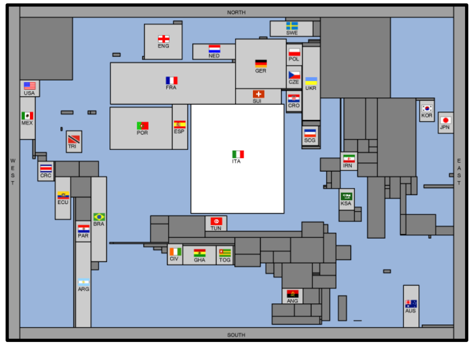


© worldmapper.org



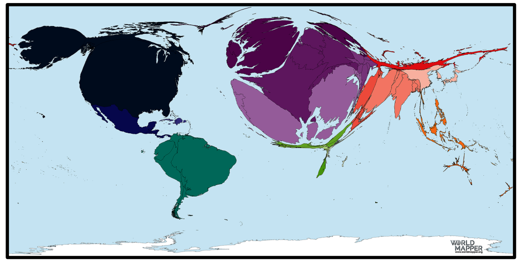
© New York Times



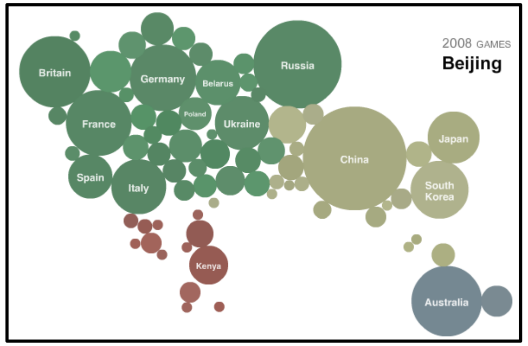


© Bettina Speckmann

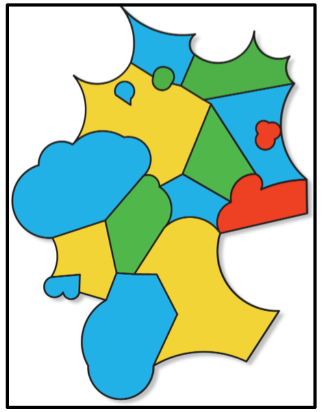
Cartograms

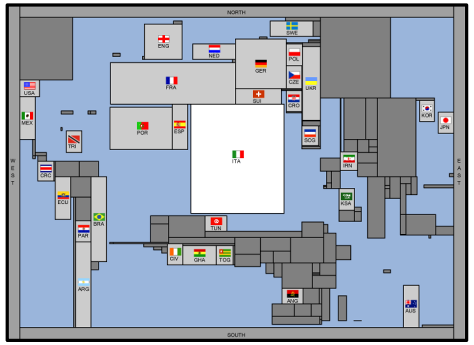


© worldmapper.org

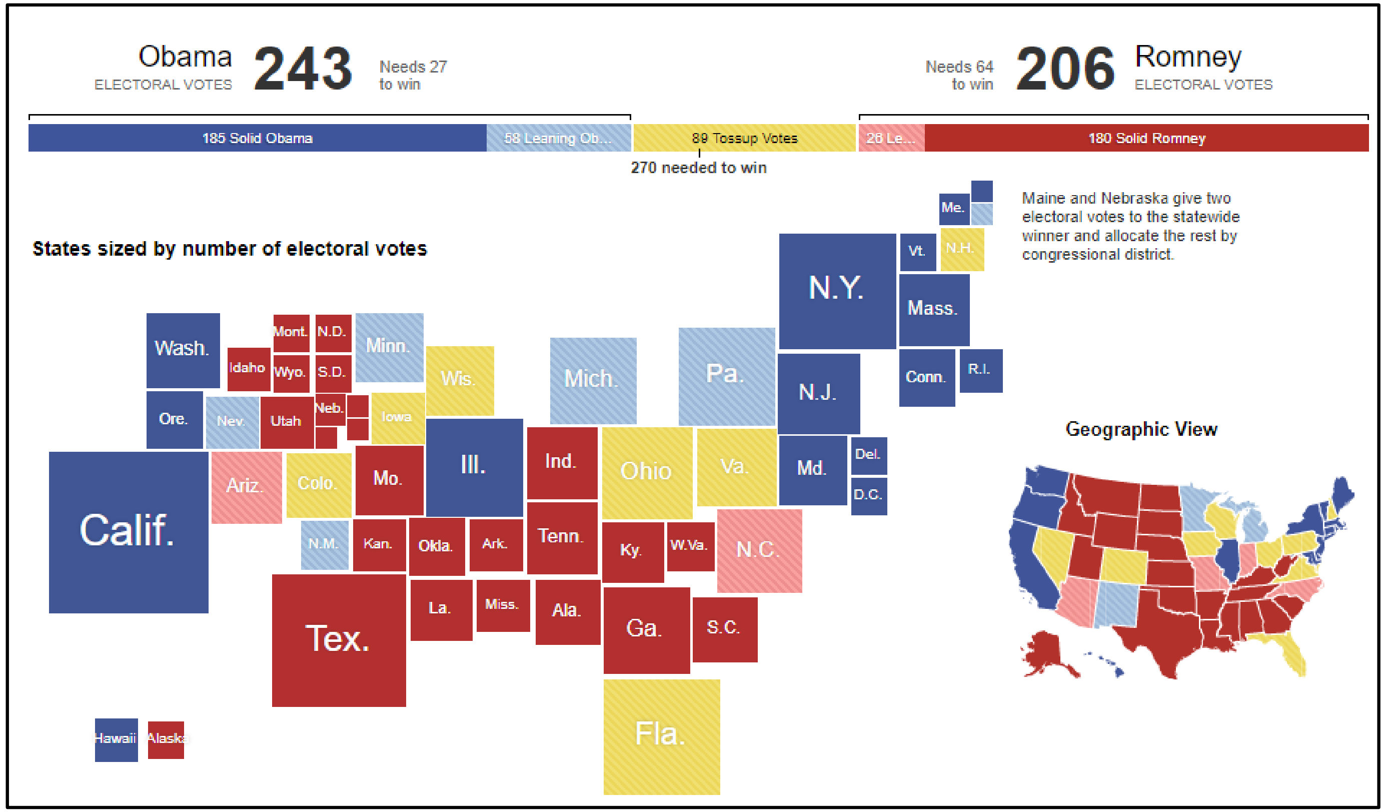


© New York Times

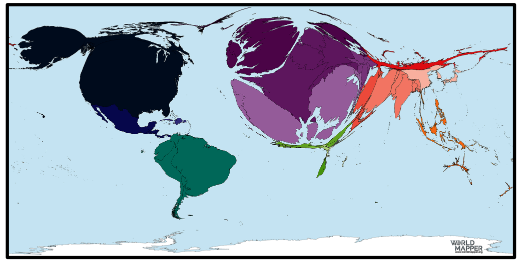




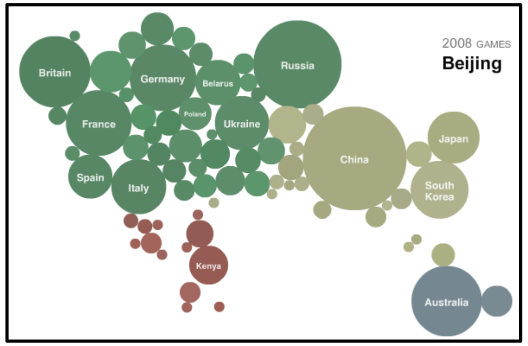
© Bettina Speckmann



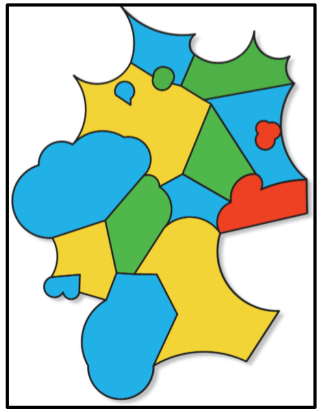
Cartograms

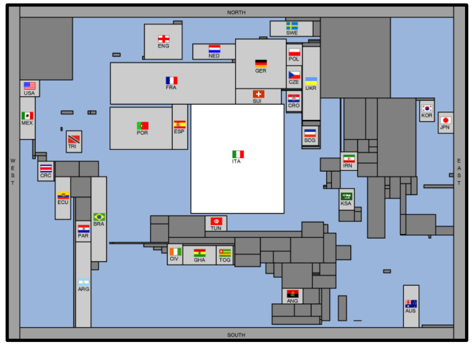


© worldmapper.org

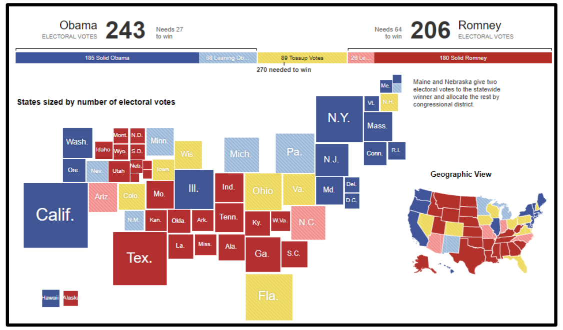


© New York Times



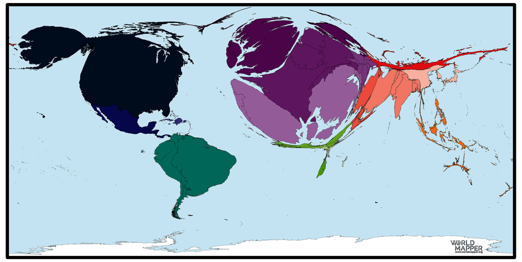


© Bettina Speckmann

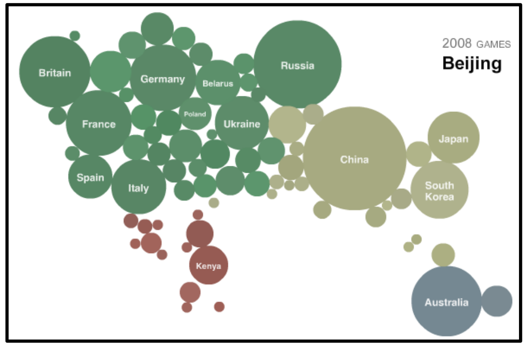


© New York Times

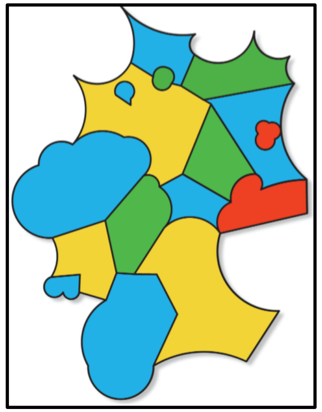
Cartograms

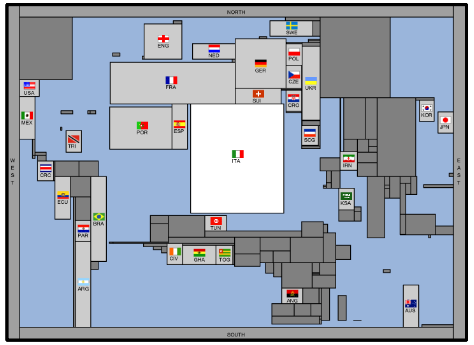


© worldmapper.org

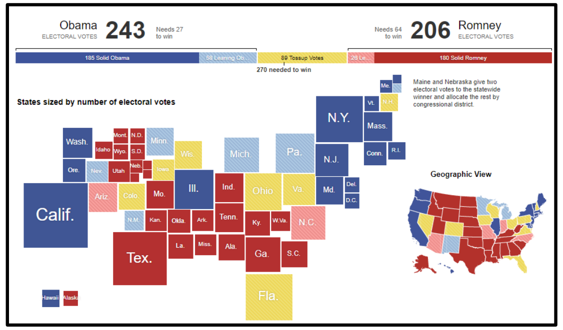


© New York Times

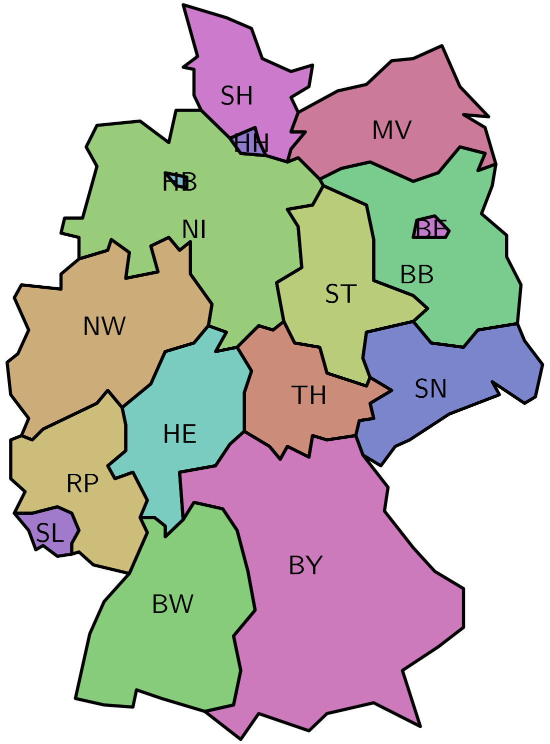




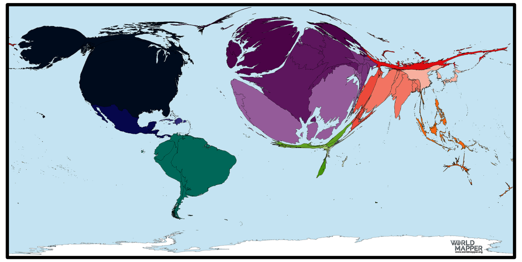
© Bettina Speckmann



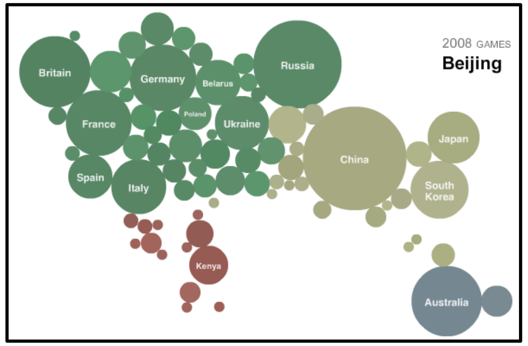
© New York Times



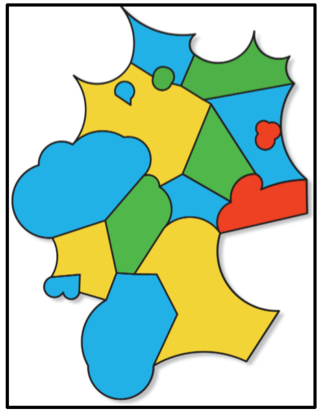
Cartograms

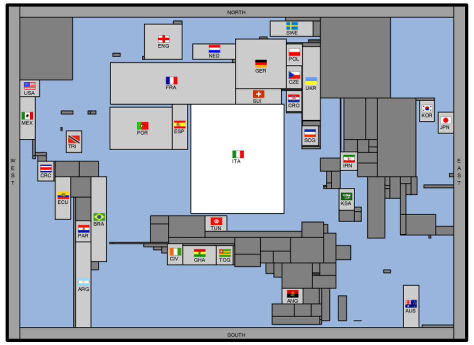


© worldmapper.org

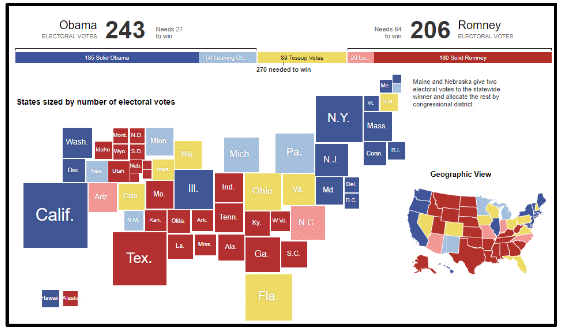


© New York Times

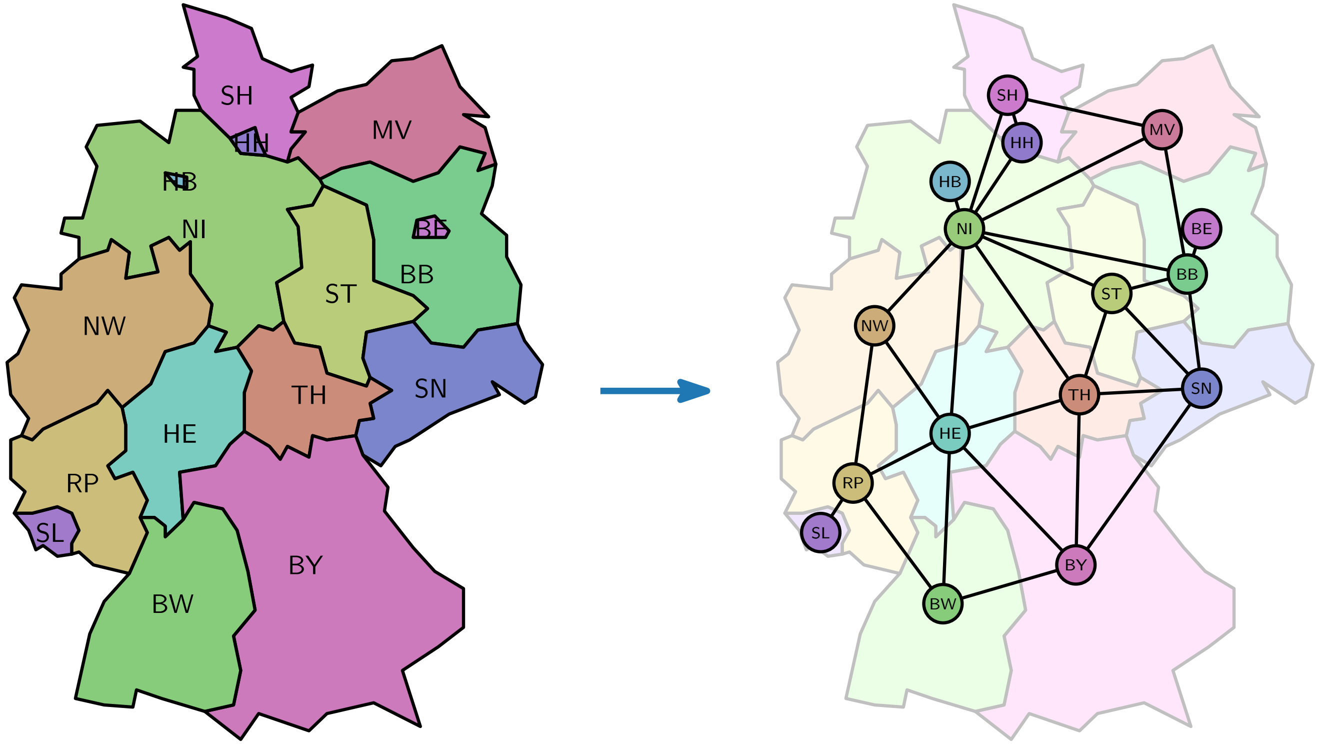




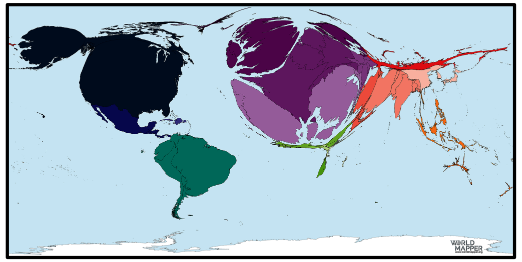
© Bettina Speckmann



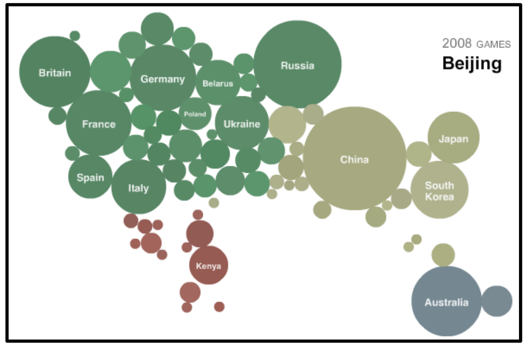
© New York Times



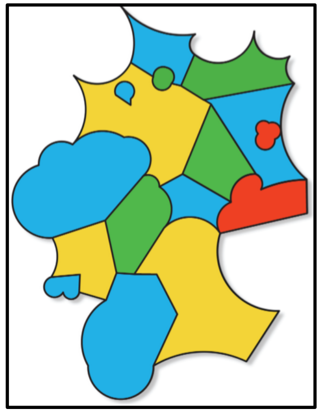
Cartograms

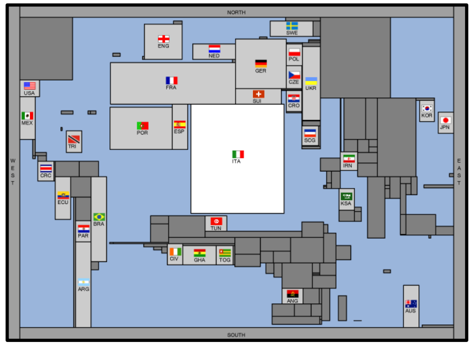


© worldmapper.org

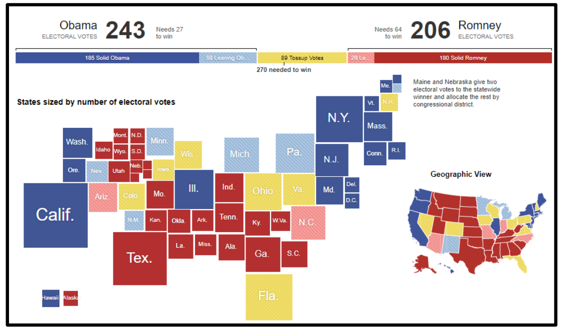


© New York Times

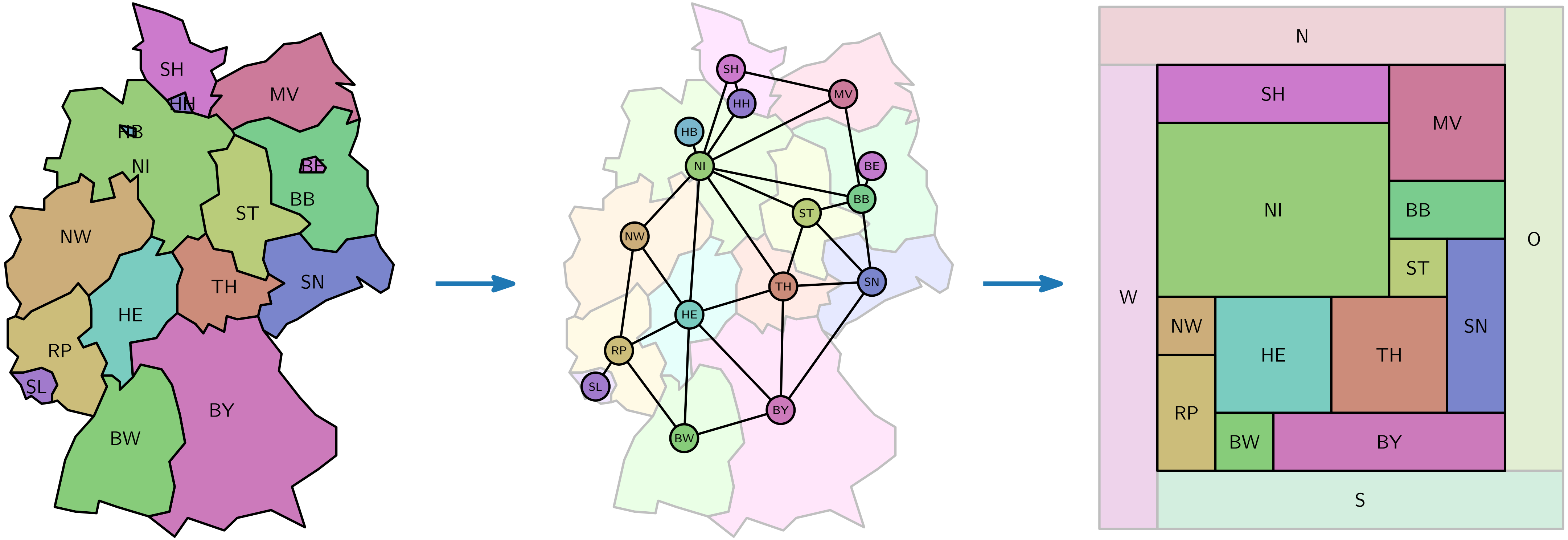




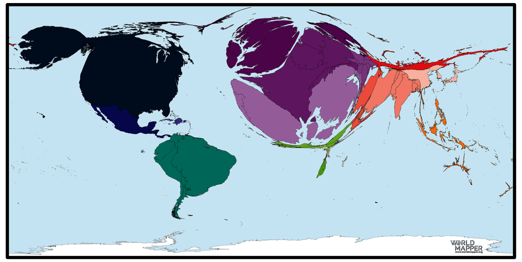
© Bettina Speckmann



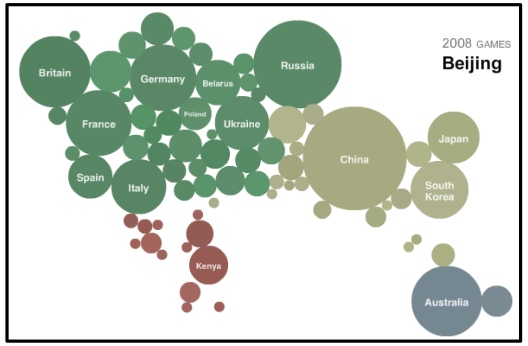
© New York Times



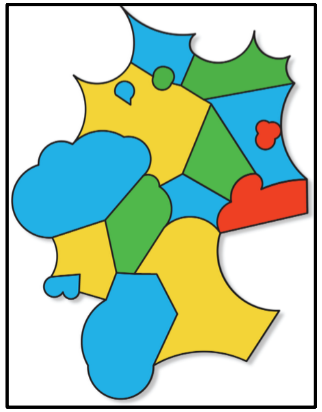
Cartograms

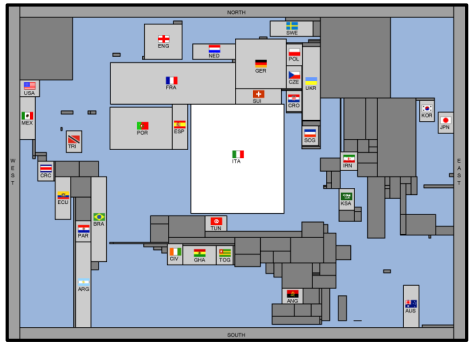


© worldmapper.org

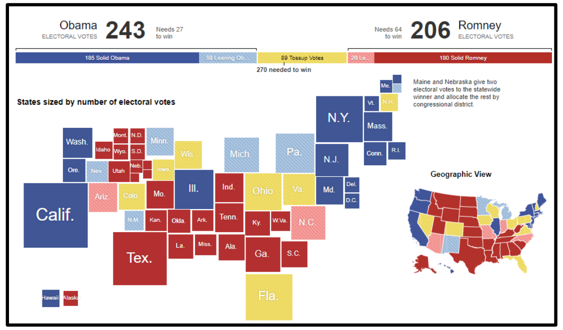


© New York Times

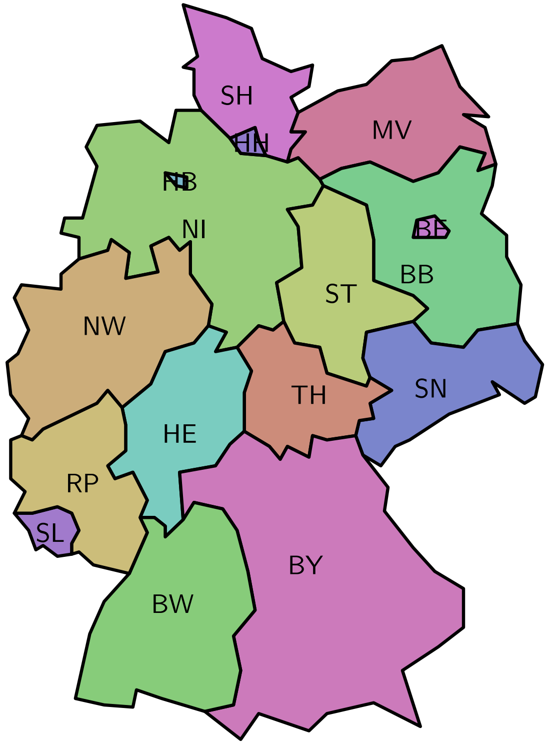


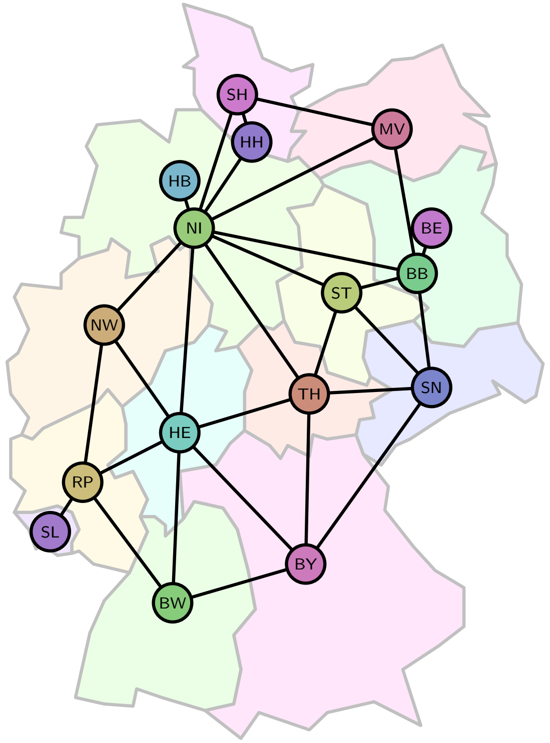


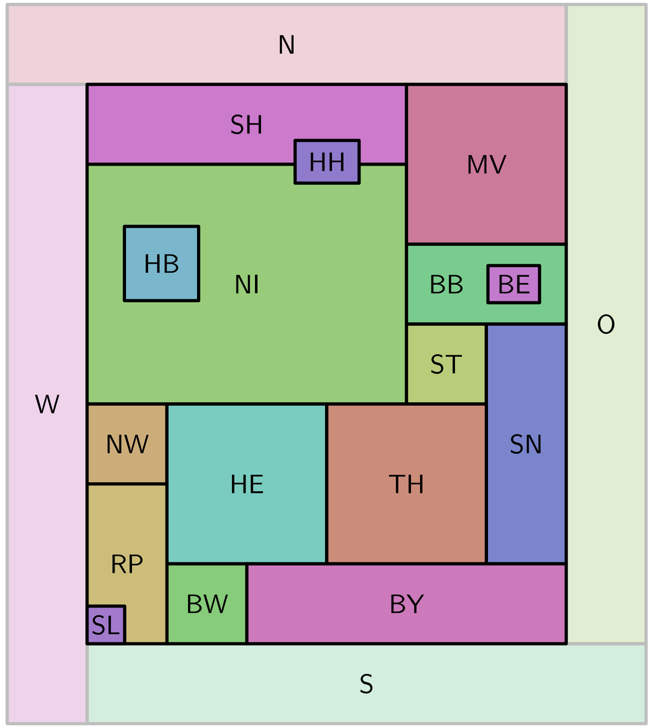
© Bettina Speckmann



© New York Times

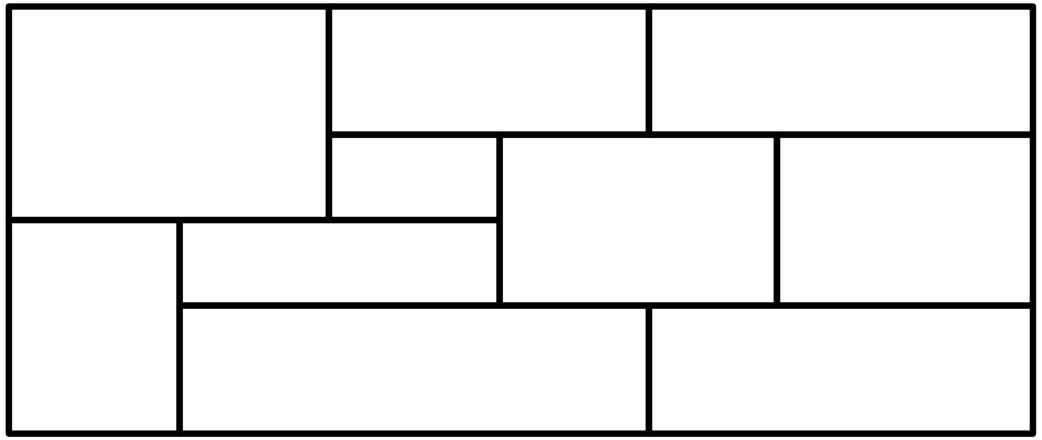




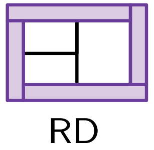


Rectangular Dual

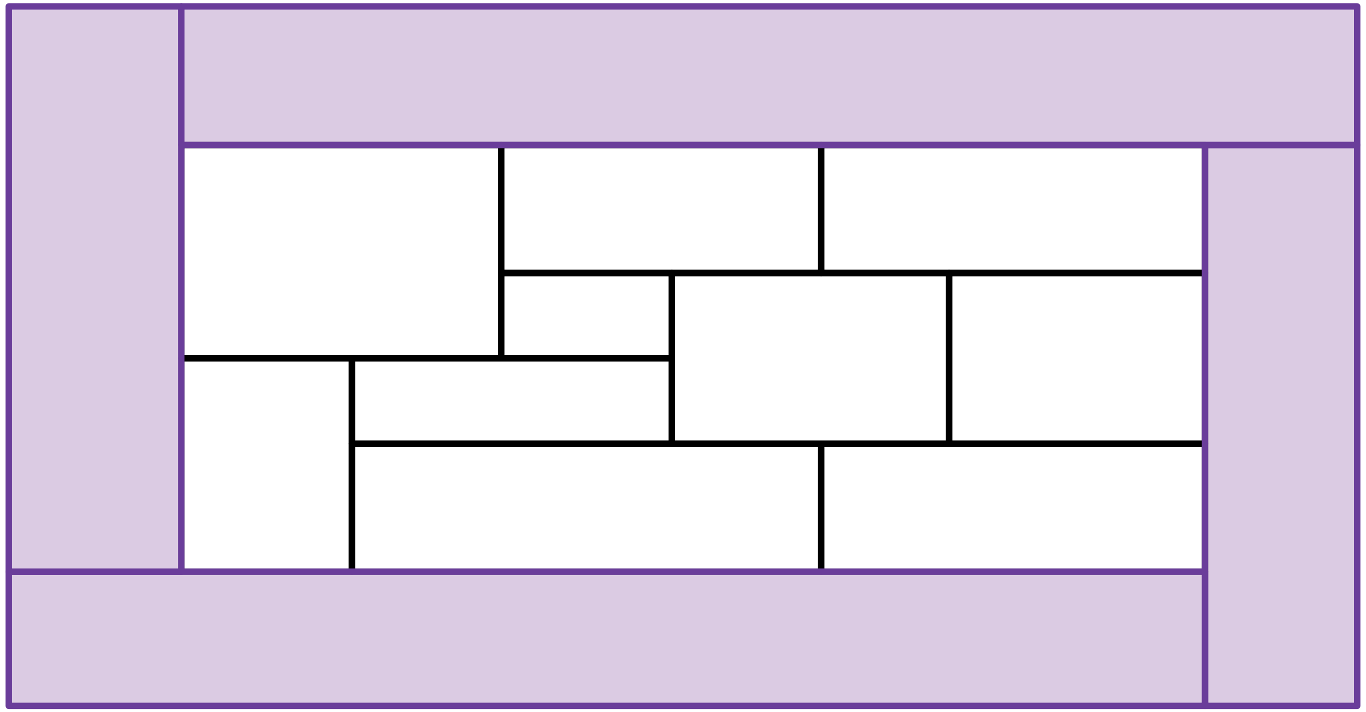
Rectangular Dual



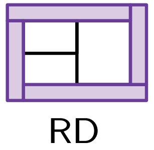
Rectangular Dual



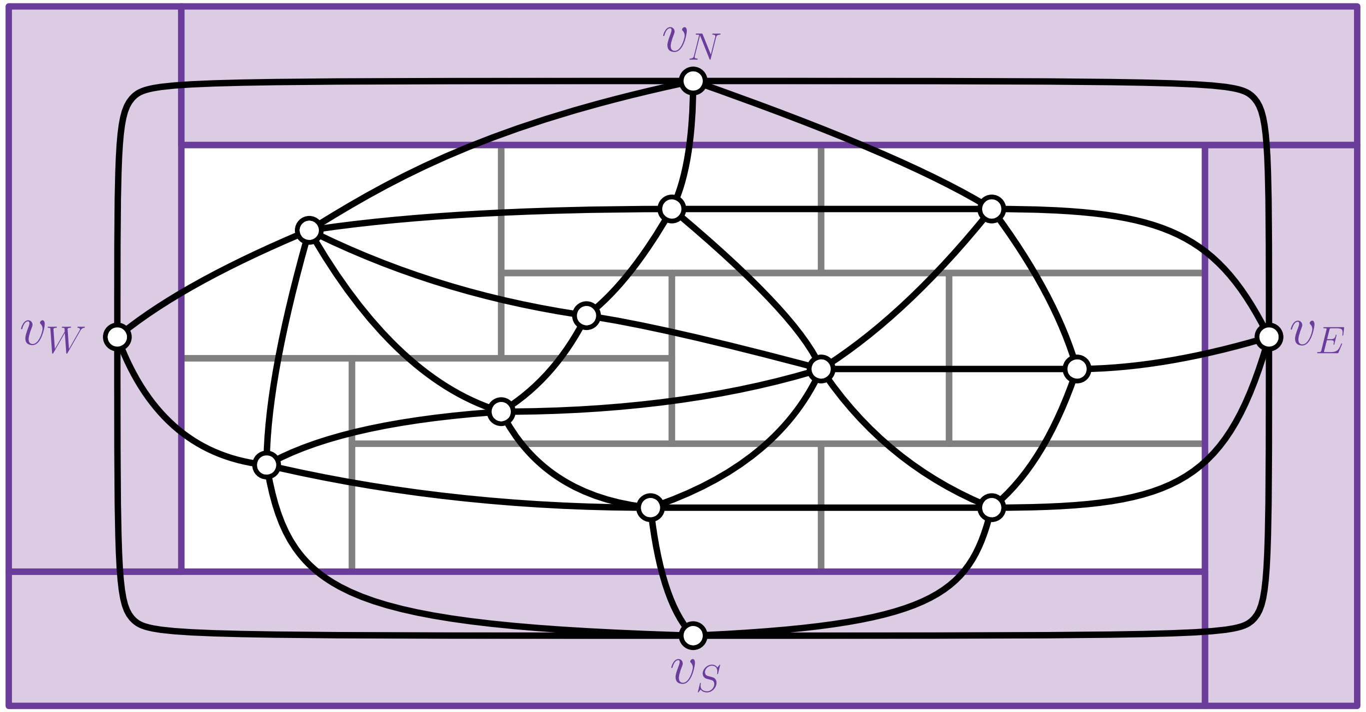
Rectangular Dual $\mathcal{R}$



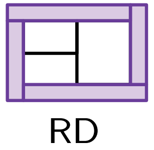
Rectangular Dual



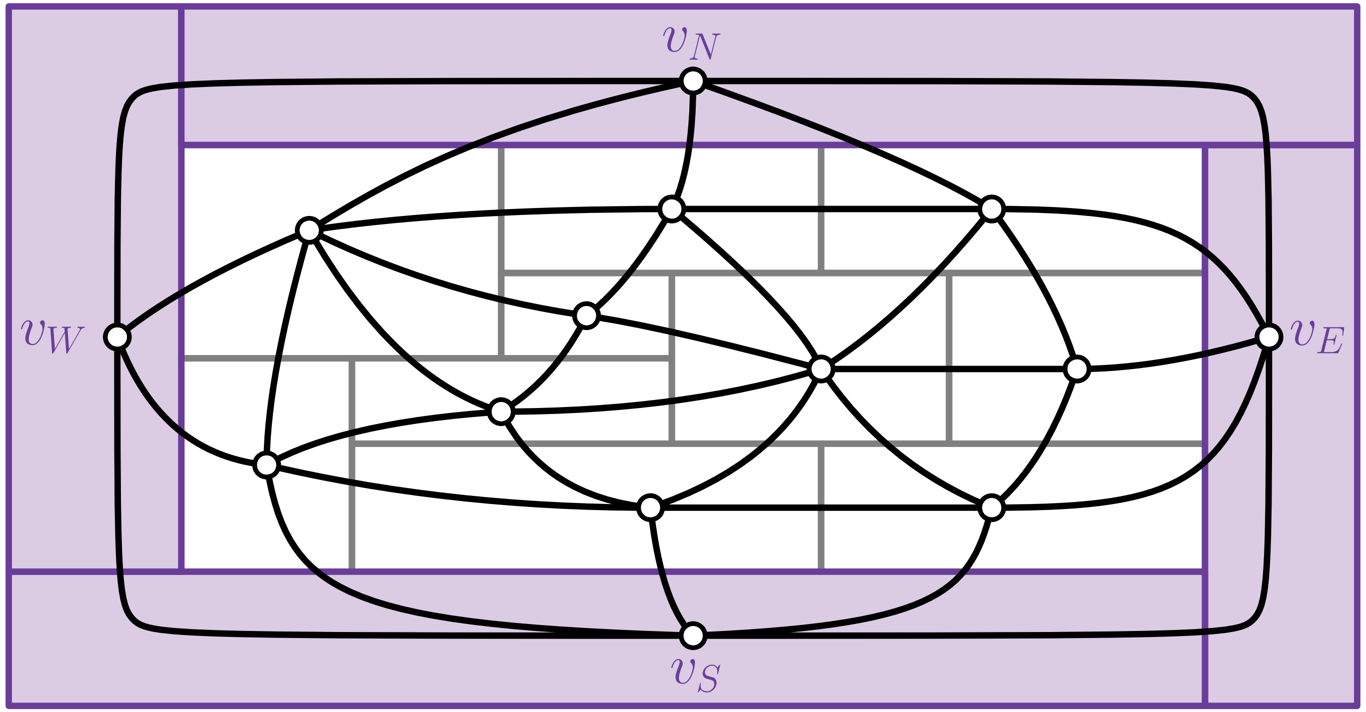
Rectangular Dual $\mathcal{R}$



Rectangular Dual

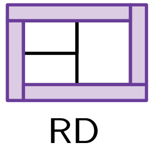


Rectangular Dual $\mathcal{R}$

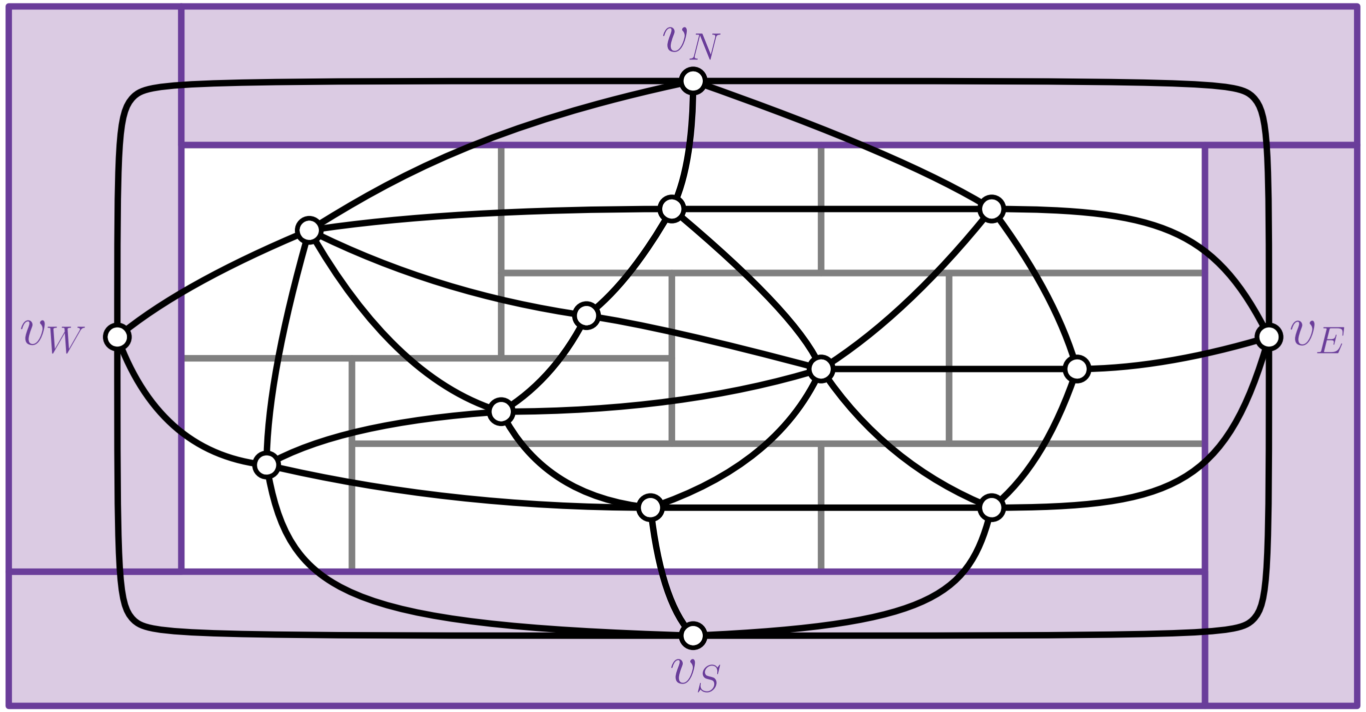


A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

Rectangular Dual

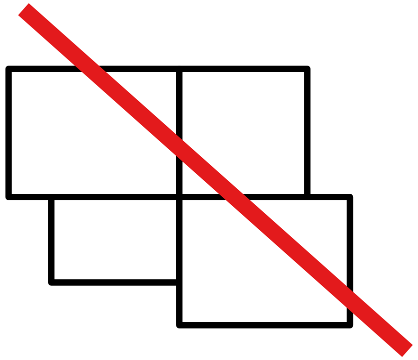


Rectangular Dual $\mathcal{R}$

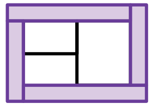


A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

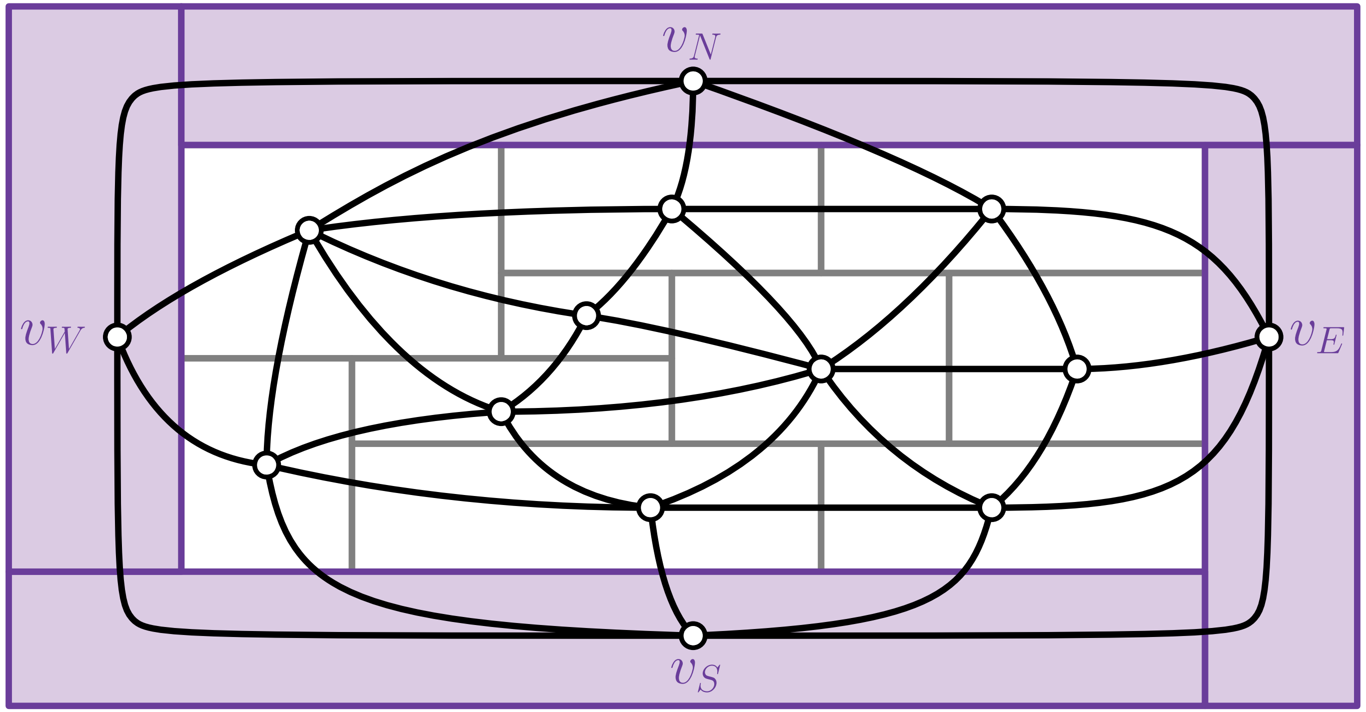
- no four rectangles share a point,



Rectangular Dual

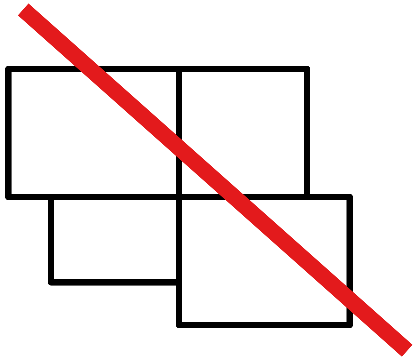


Rectangular Dual $\mathcal{R}$

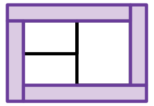


A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

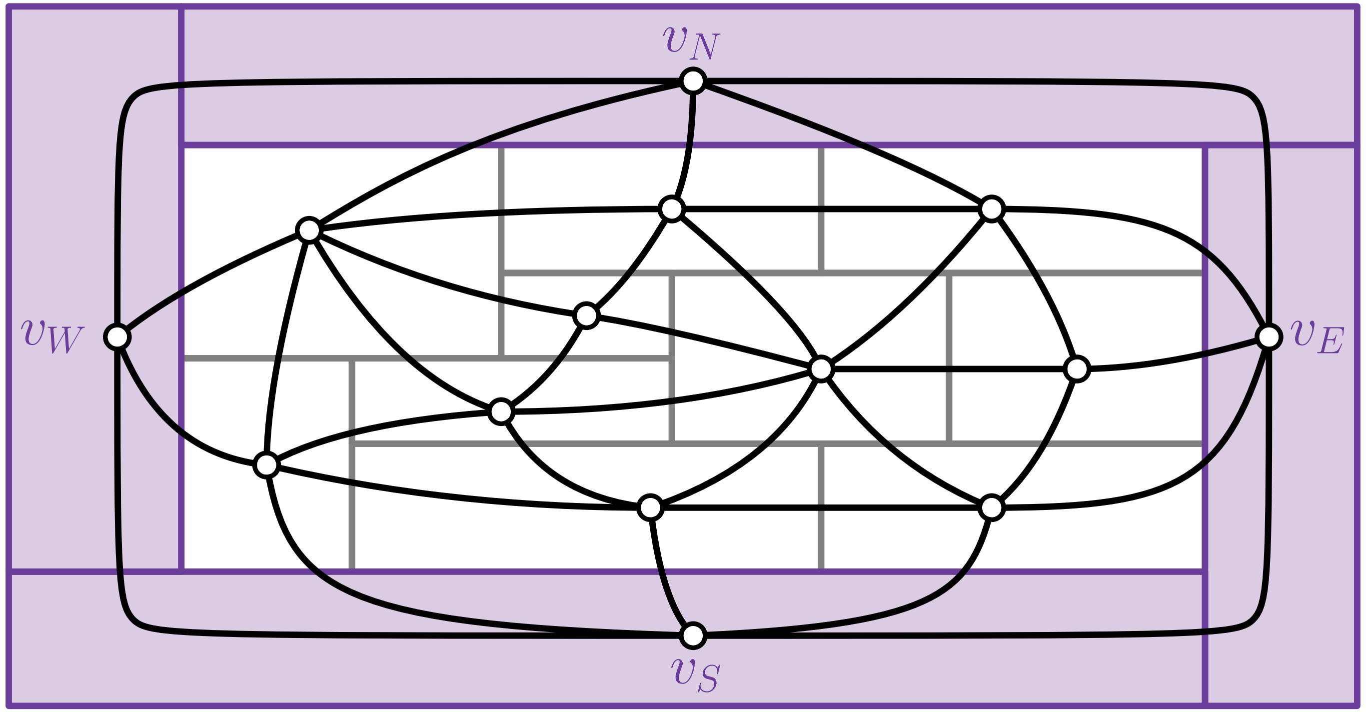
- no four rectangles share a point, and
- the union of all rectangles is a rectangle



Rectangular Dual

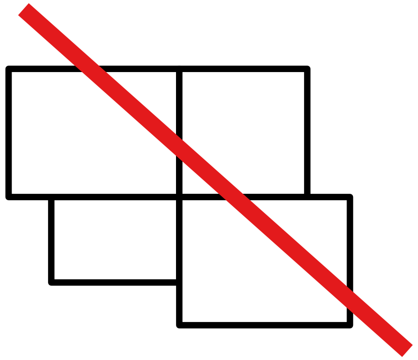


Rectangular Dual $\mathcal{R}$



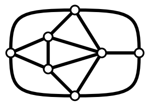
A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

- no four rectangles share a point, and
- the union of all rectangles is a rectangle

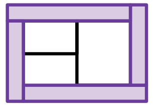


Theorem. [Koźmiński, Kinnen '85]
A graph G has a rectangular dual $\mathcal{R}$ if and only if G is a PTP graph.

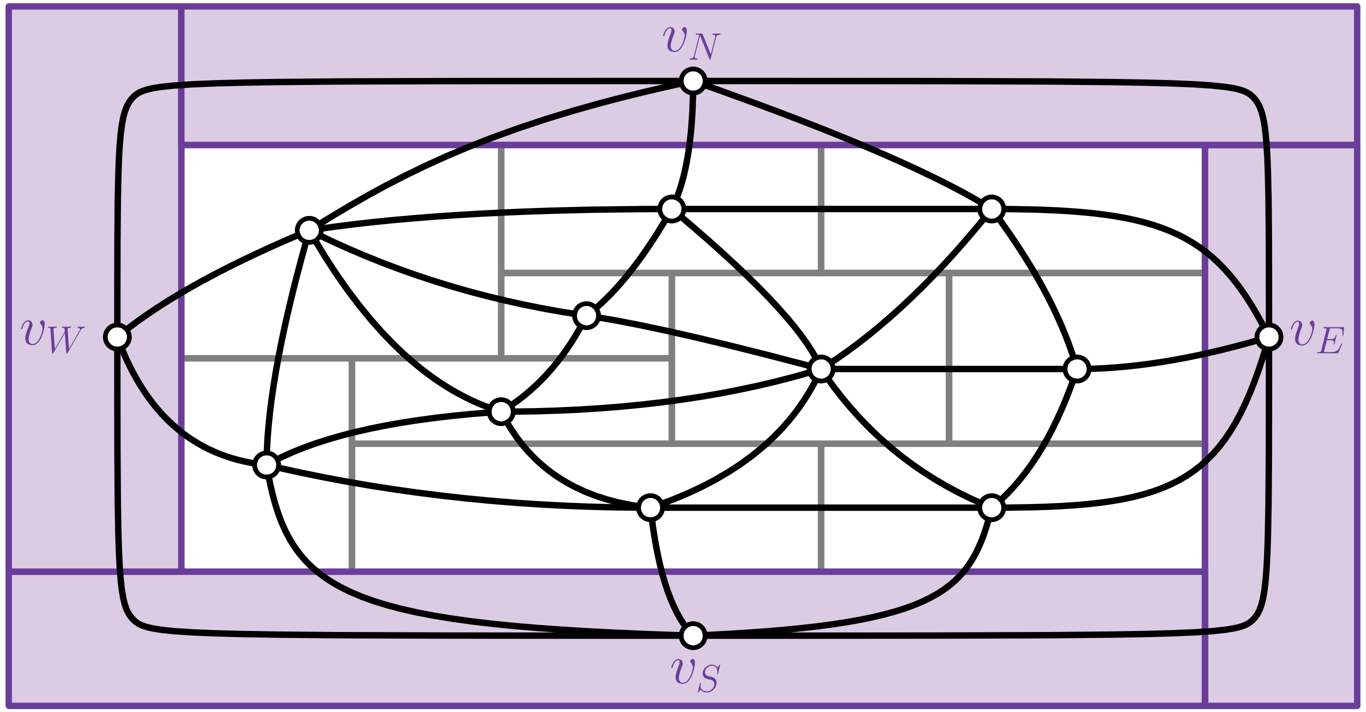
Rectangular Dual



Properly Triangulated Planar Graph G

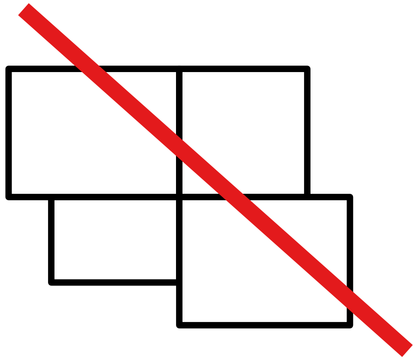


Rectangular Dual $\mathcal{R}$



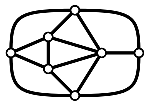
A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

- no four rectangles share a point, and
- the union of all rectangles is a rectangle

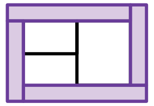


Theorem. [Koźmiński, Kinnen '85]
A graph G has a rectangular dual $\mathcal{R}$ if and only if G is a PTP graph.

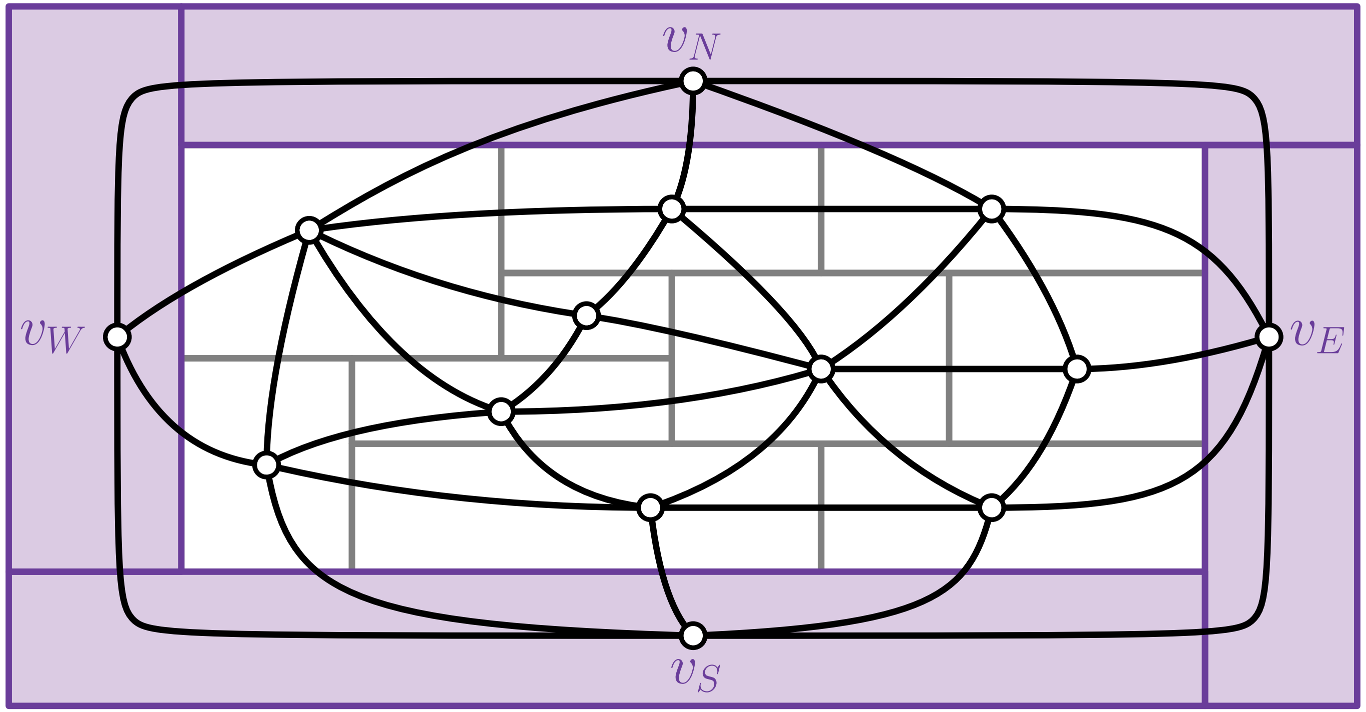
Rectangular Dual



Properly Triangulated Planar Graph G

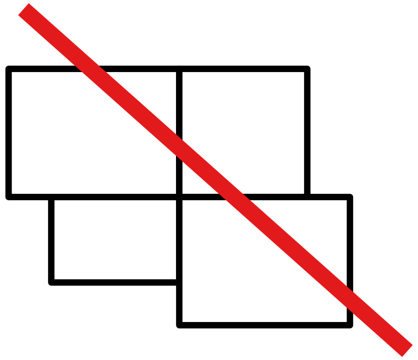


Rectangular Dual $\mathcal{R}$



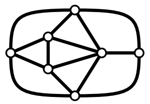
A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

- no four rectangles share a point, and
- the union of all rectangles is a rectangle

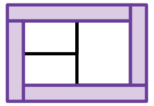


Theorem. [Koźmiński, Kinnen '85]
A graph G has a rectangular dual $\mathcal{R}$ if and only if G is a PTP graph.

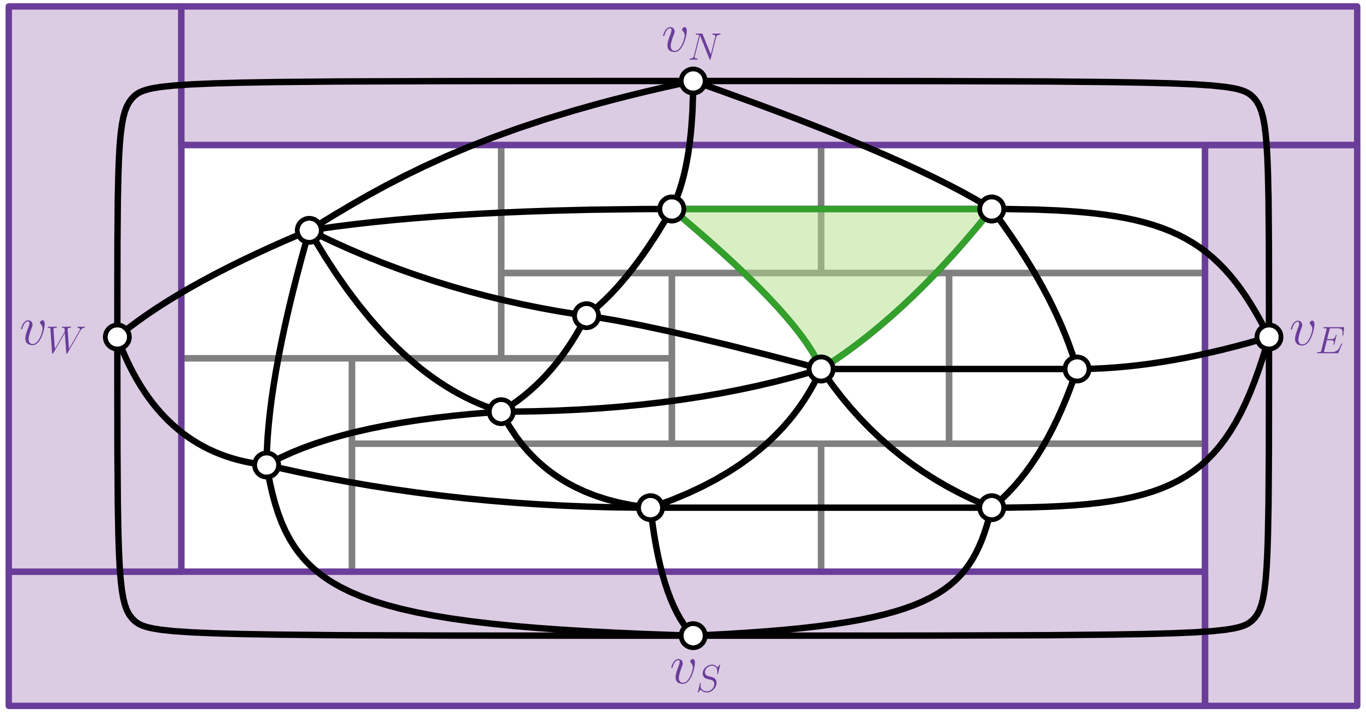
Rectangular Dual



Properly Triangulated Planar Graph G

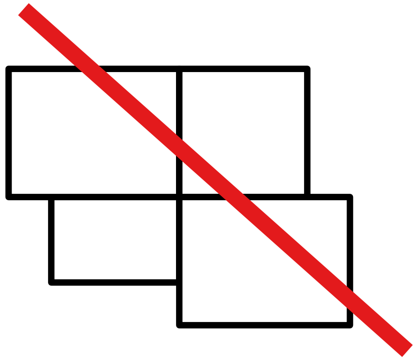


Rectangular Dual $\mathcal{R}$



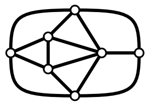
A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

- no four rectangles share a point, and
- the union of all rectangles is a rectangle

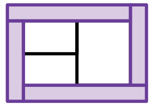


Theorem. [Koźmiński, Kinnen '85]
A graph G has a rectangular dual $\mathcal{R}$ if and only if G is a PTP graph.

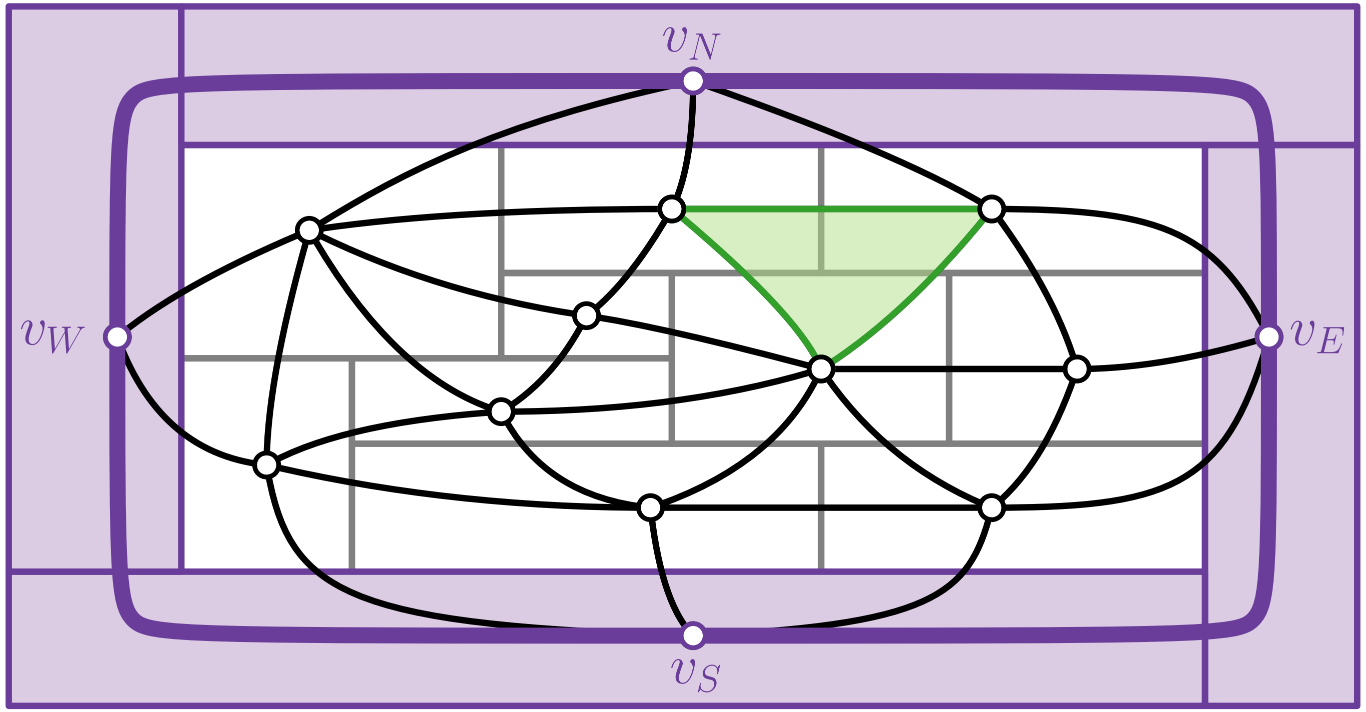
Rectangular Dual



Properly Triangulated Planar Graph G

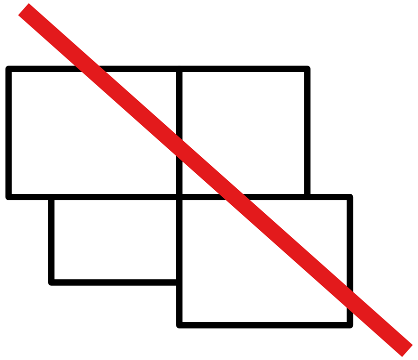


Rectangular Dual $\mathcal{R}$



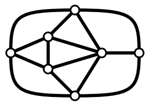
A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

- no four rectangles share a point, and
- the union of all rectangles is a rectangle

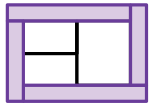


Theorem. [Koźmiński, Kinnen ’85]
A graph G has a rectangular dual $\mathcal{R}$ if and only if G is a PTP graph.

Rectangular Dual

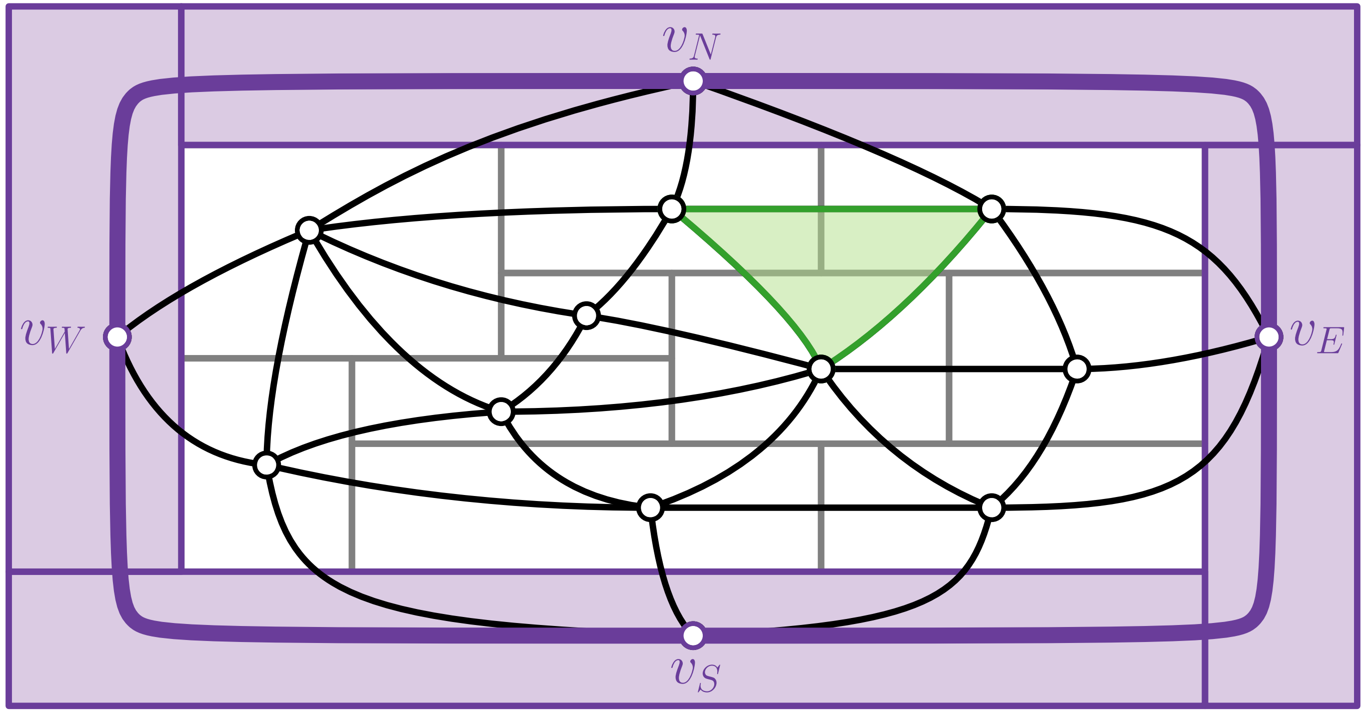


Properly Triangulated Planar Graph G



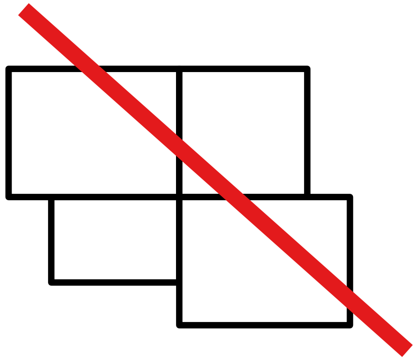
Rectangular Dual $\mathcal{R}$

Exactly 4 vertices on outer face



A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

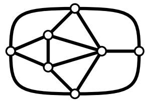
- no four rectangles share a point, and
- the union of all rectangles is a rectangle



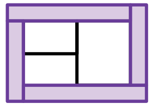
Theorem. [Koźmiński, Kinnen '85]
A graph G has a rectangular dual $\mathcal{R}$ if and only if G is a PTP graph.

Rectangular Dual

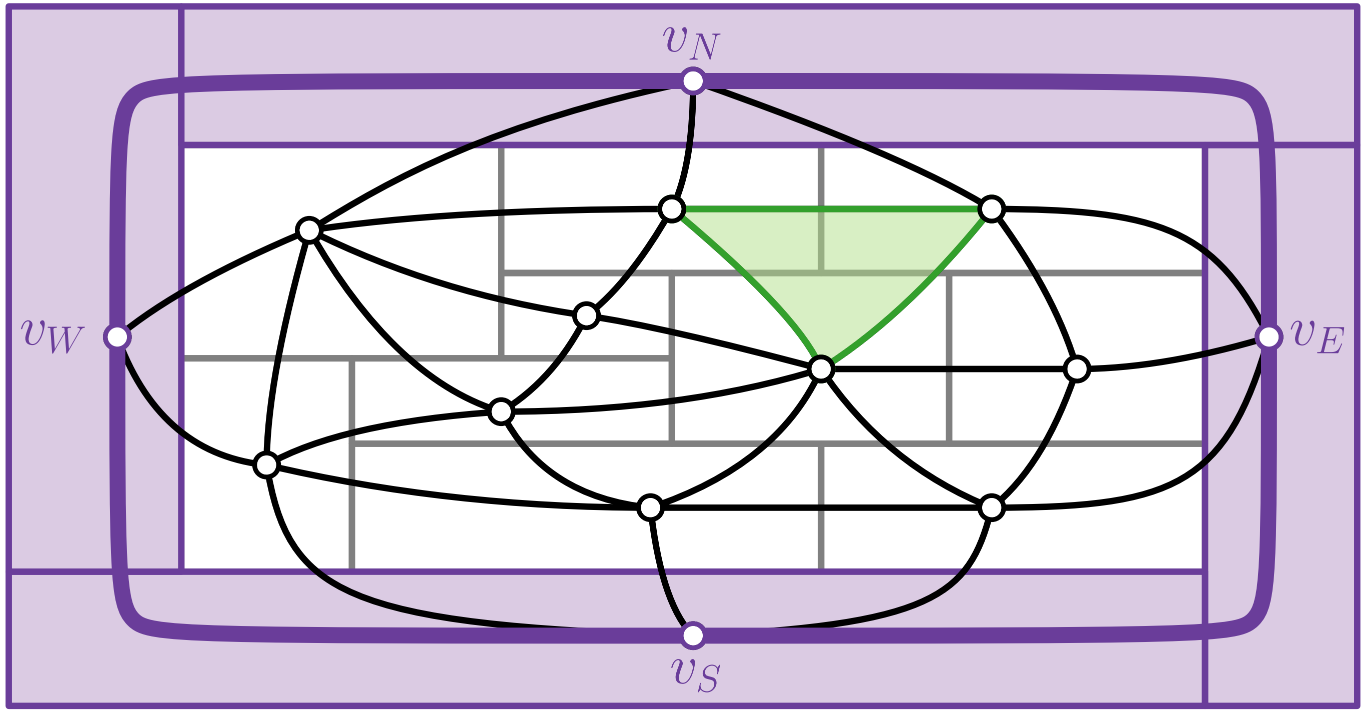
Exactly 4 vertices on outer face



Properly Triangulated Planar Graph G



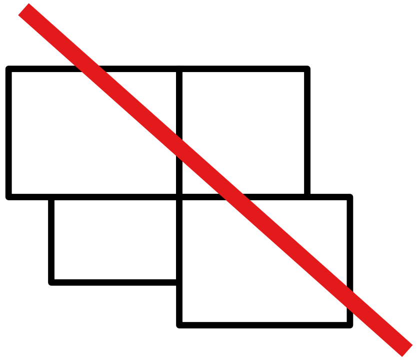
Rectangular Dual $\mathcal{R}$



no separating triangle

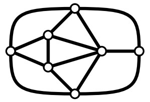
A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

- no four rectangles share a point, and
- the union of all rectangles is a rectangle

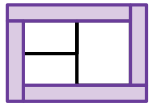


Theorem. [Koźmiński, Kinnen '85]
A graph G has a rectangular dual $\mathcal{R}$ if and only if G is a PTP graph.

Rectangular Dual

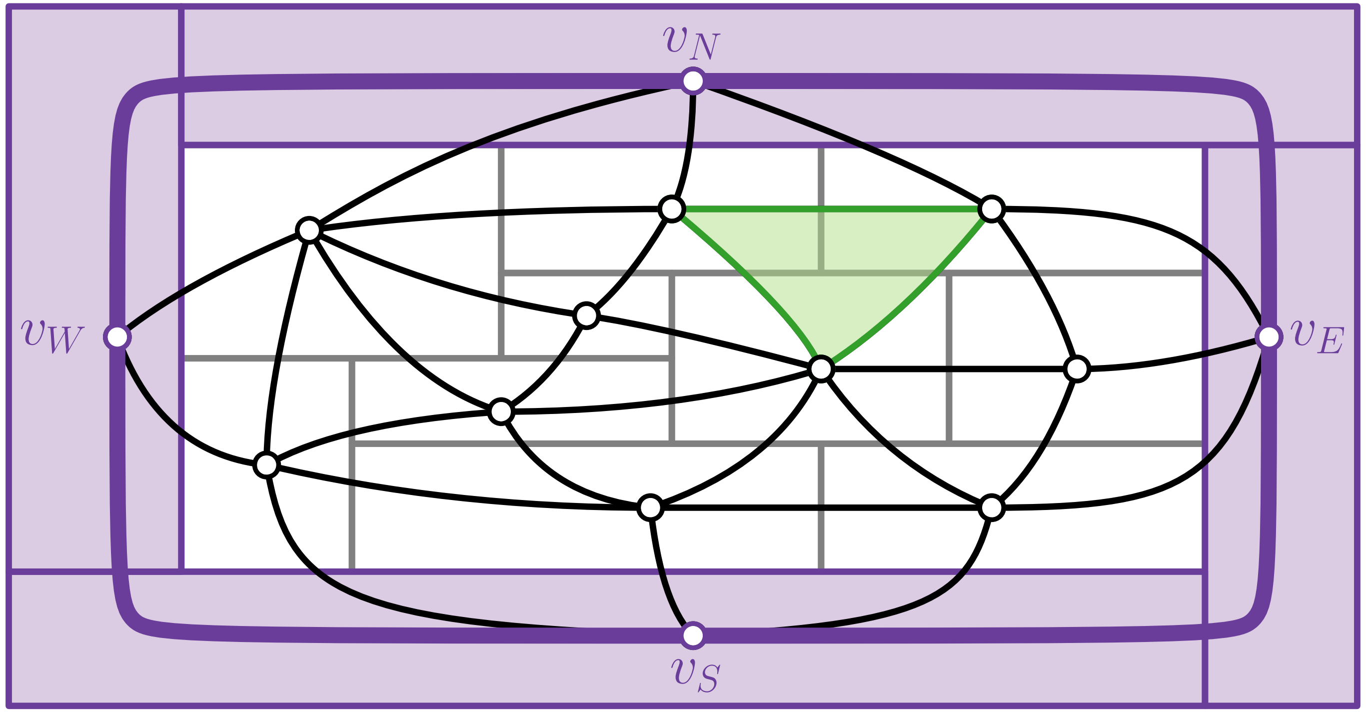


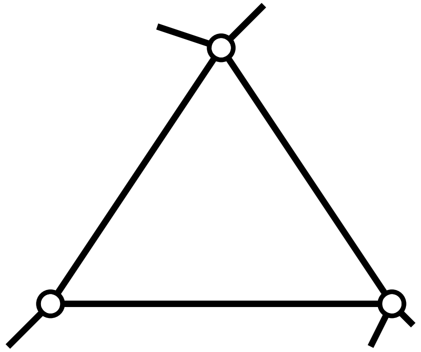
Properly Triangulated Planar Graph G



Rectangular Dual $\mathcal{R}$

Exactly 4 vertices on outer face

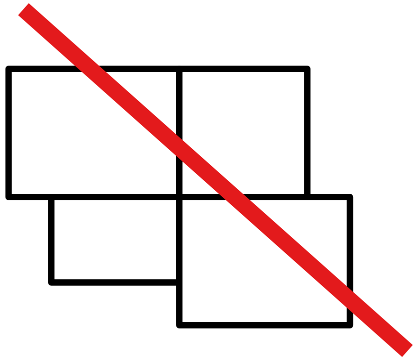




no separating triangle

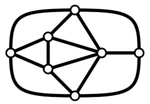
A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

- no four rectangles share a point, and
- the union of all rectangles is a rectangle



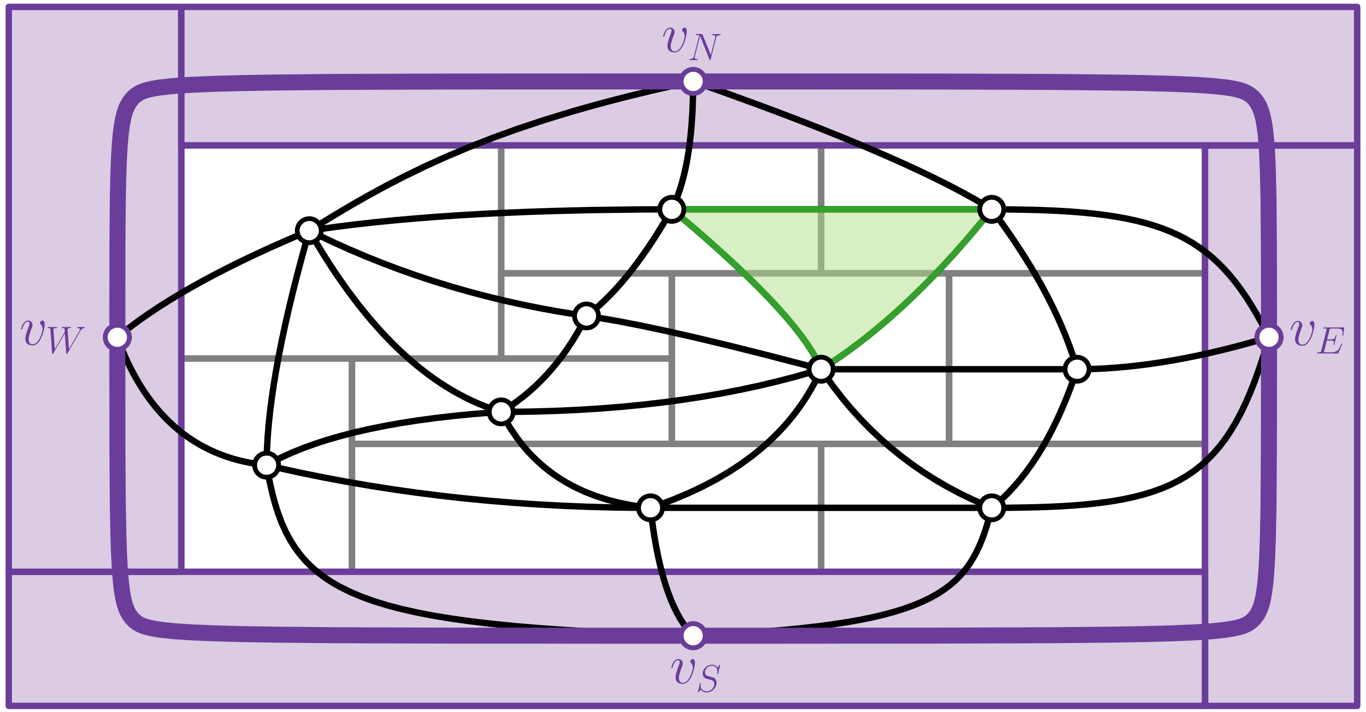
Theorem. [Koźmiński, Kinnen '85]
A graph G has a rectangular dual $\mathcal{R}$ if and only if G is a PTP graph.

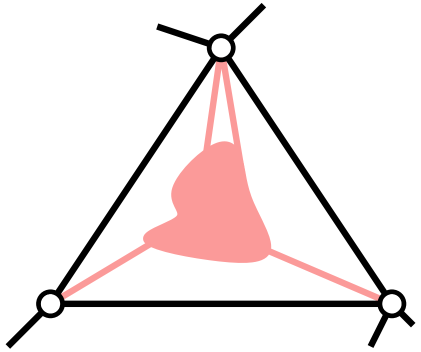
Rectangular Dual


PTP

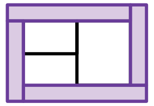
Properly Triangulated Planar Graph G

Exactly 4 vertices on outer face





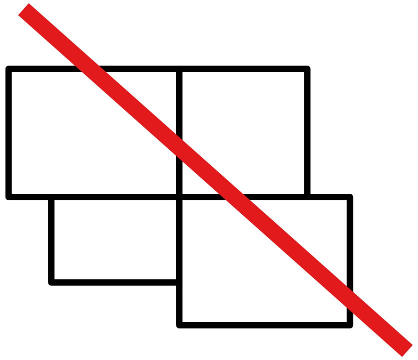
no separating triangle


RD

Rectangular Dual $\mathcal{R}$

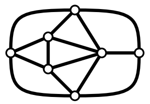
A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

- no four rectangles share a point, and
- the union of all rectangles is a rectangle

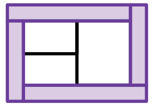


Theorem. [Koźmiński, Kinnen '85]
A graph G has a rectangular dual $\mathcal{R}$ if and only if G is a PTP graph.

Rectangular Dual

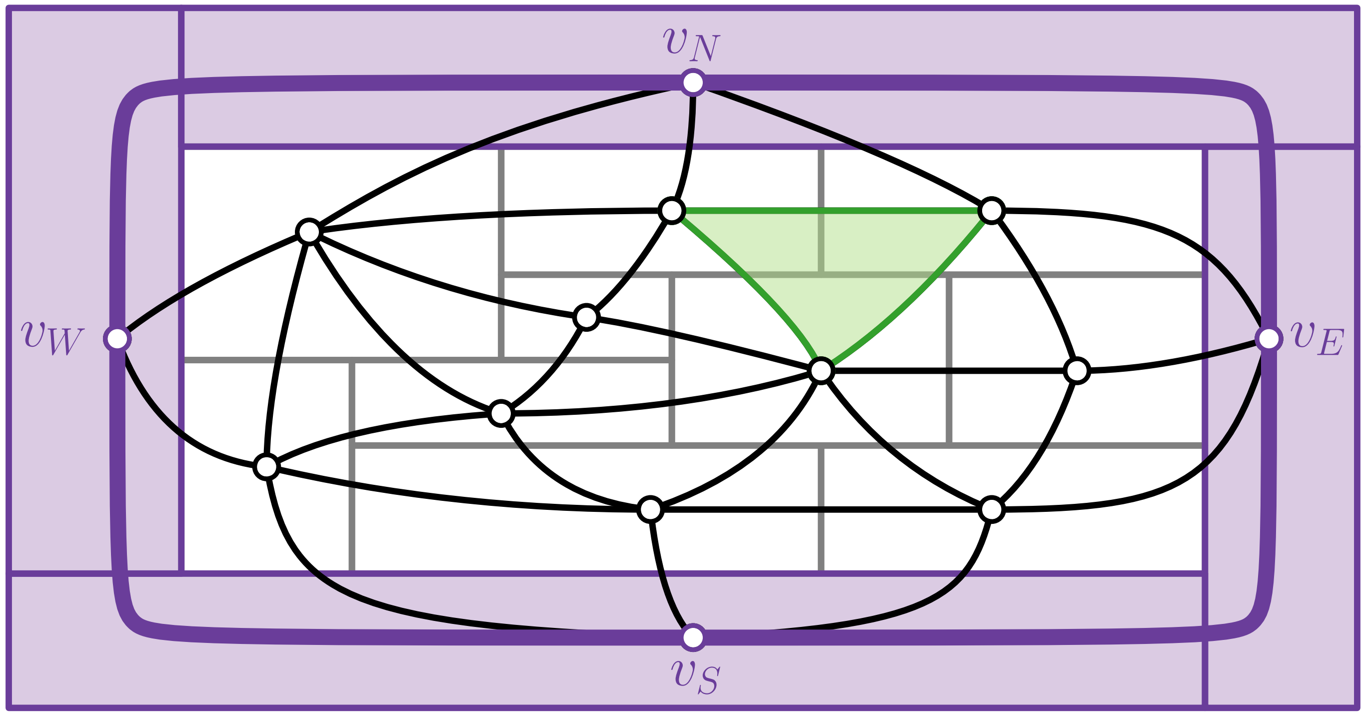

PTP

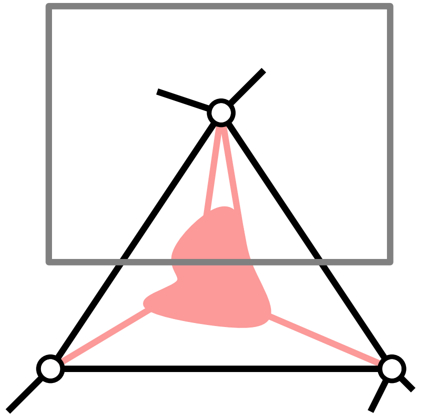
Properly Triangulated Planar Graph G


RD

Rectangular Dual $\mathcal{R}$

Exactly 4 vertices on outer face

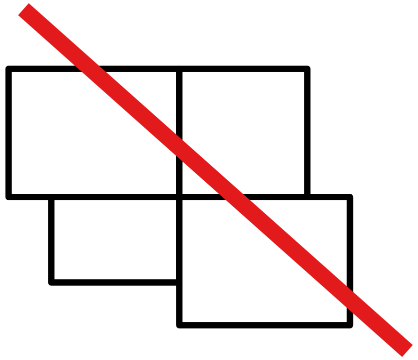




no separating triangle

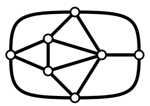
A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

- no four rectangles share a point, and
- the union of all rectangles is a rectangle

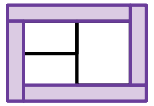


Theorem. [Koźmiński, Kinnen '85]
A graph G has a rectangular dual $\mathcal{R}$ if and only if G is a PTP graph.

Rectangular Dual

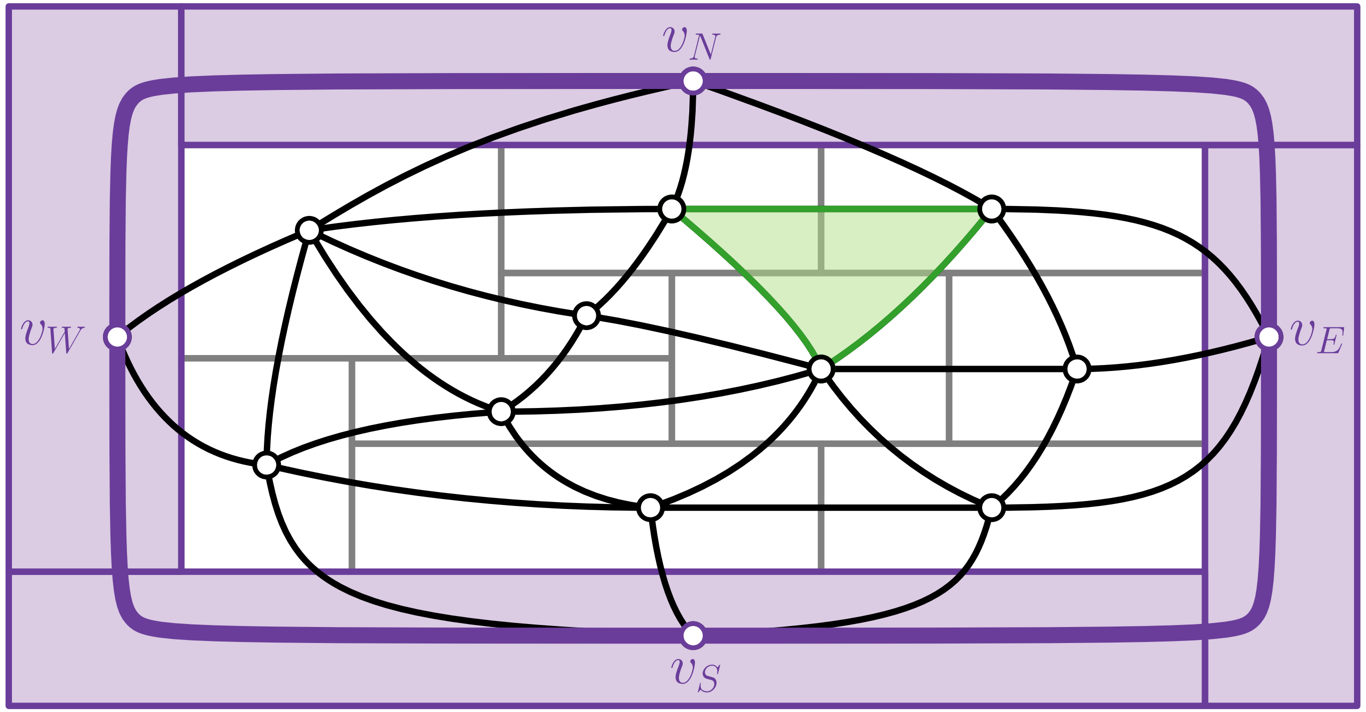


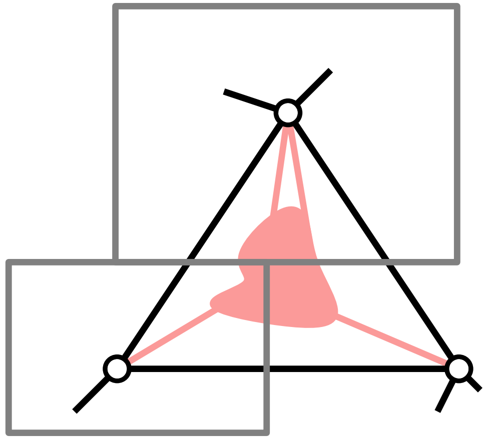
Properly Triangulated Planar Graph G



Rectangular Dual $\mathcal{R}$

Exactly 4 vertices on outer face

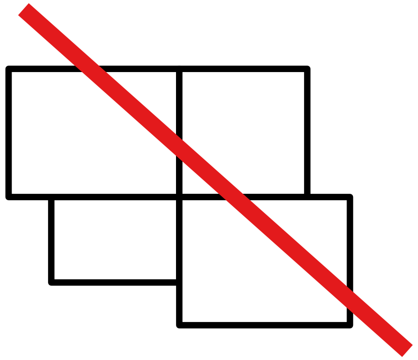




no separating triangle

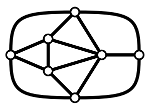
A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

- no four rectangles share a point, and
- the union of all rectangles is a rectangle

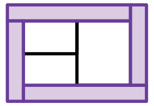


Theorem. [Koźmiński, Kinnen '85]
A graph G has a rectangular dual $\mathcal{R}$ if and only if G is a PTP graph.

Rectangular Dual

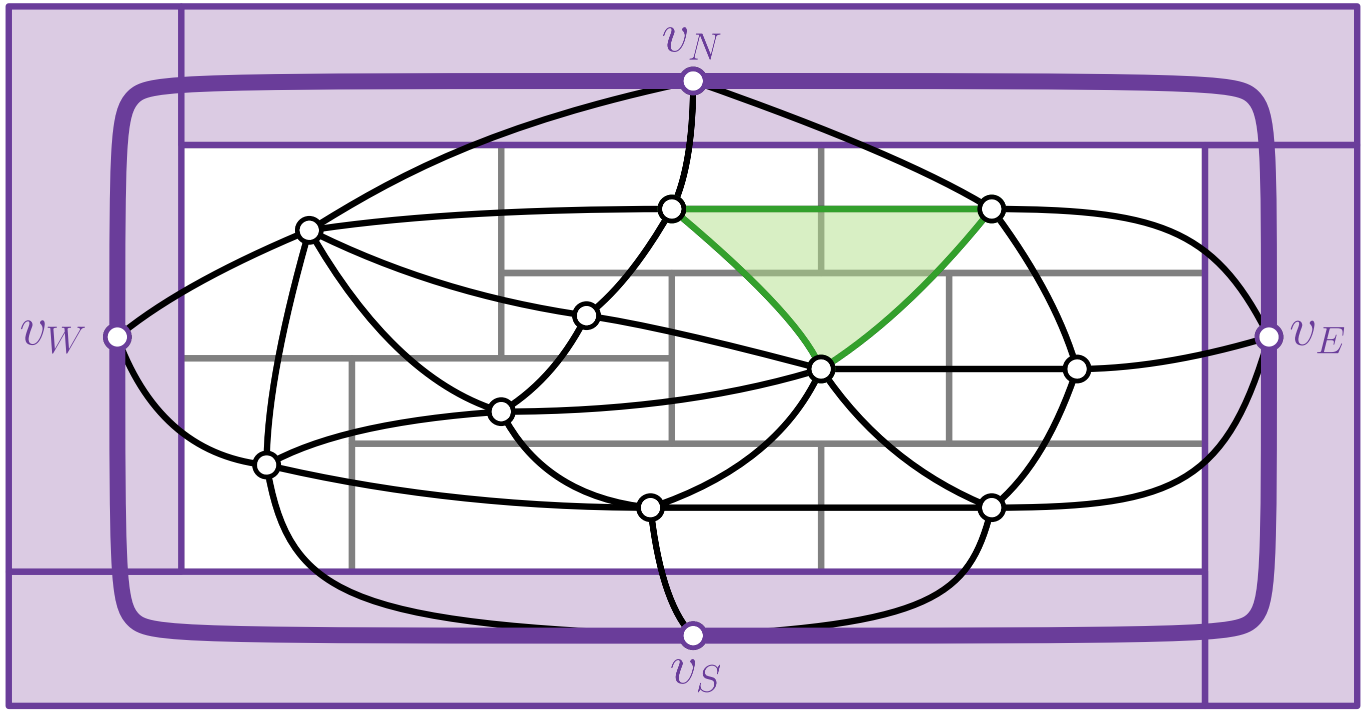

PTP

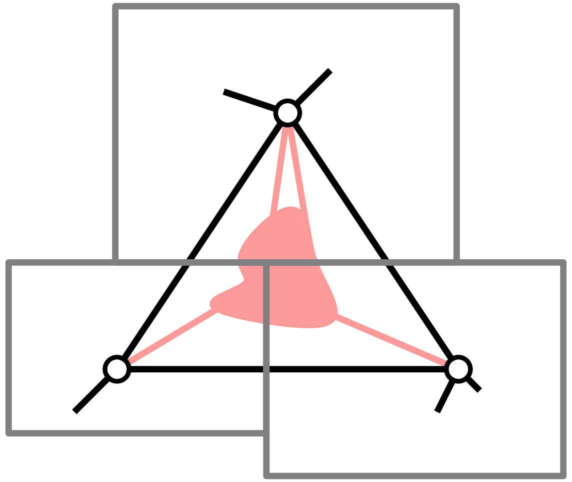
Properly Triangulated Planar Graph G


RD

Rectangular Dual $\mathcal{R}$

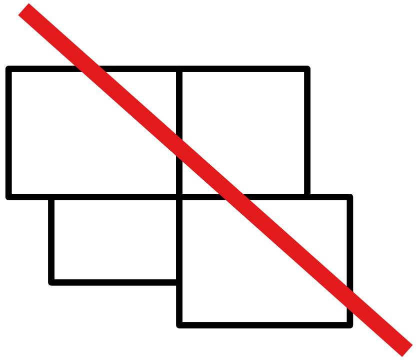
Exactly 4 vertices on outer face




no separating triangle

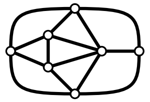
A **rectangular dual** of a graph G is a contact representation with axis-aligned rectangles s.t.

- no four rectangles share a point, and
- the union of all rectangles is a rectangle

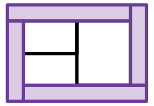


Theorem. [Koźmiński, Kinnen '85]
A graph G has a rectangular dual $\mathcal{R}$ if and only if G is a PTP graph.

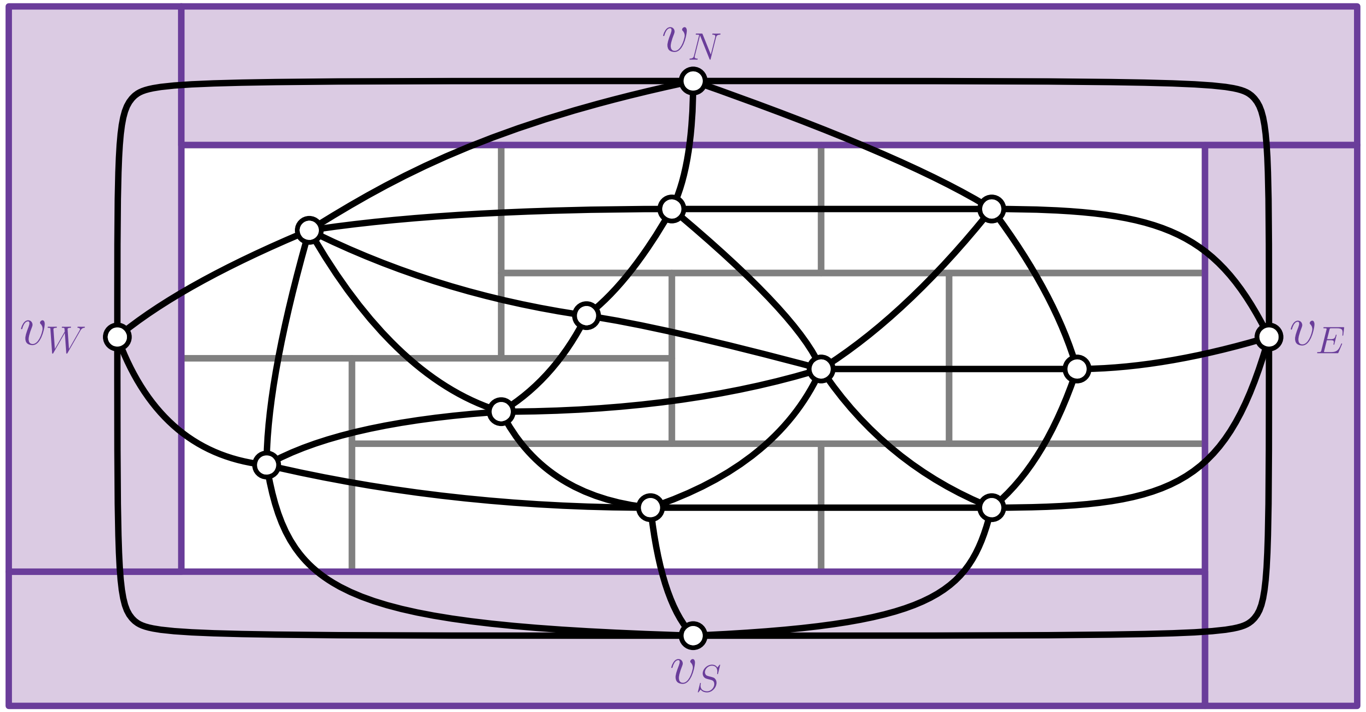
Regular Edge Labeling



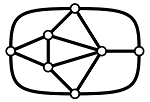
Properly Triangulated Planar Graph G



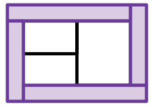
Rectangular Dual $\mathcal{R}$



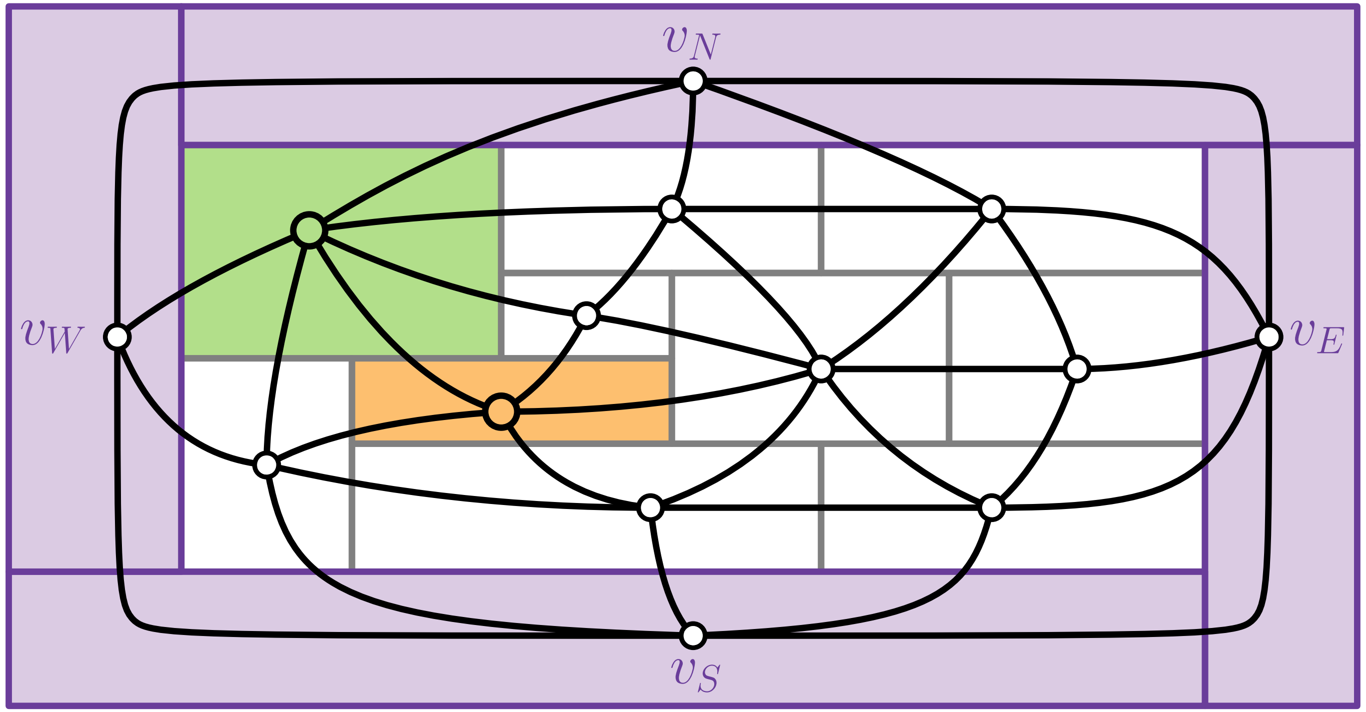
Regular Edge Labeling



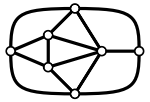
Properly Triangulated Planar Graph G



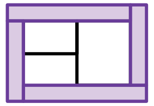
Rectangular Dual $\mathcal{R}$



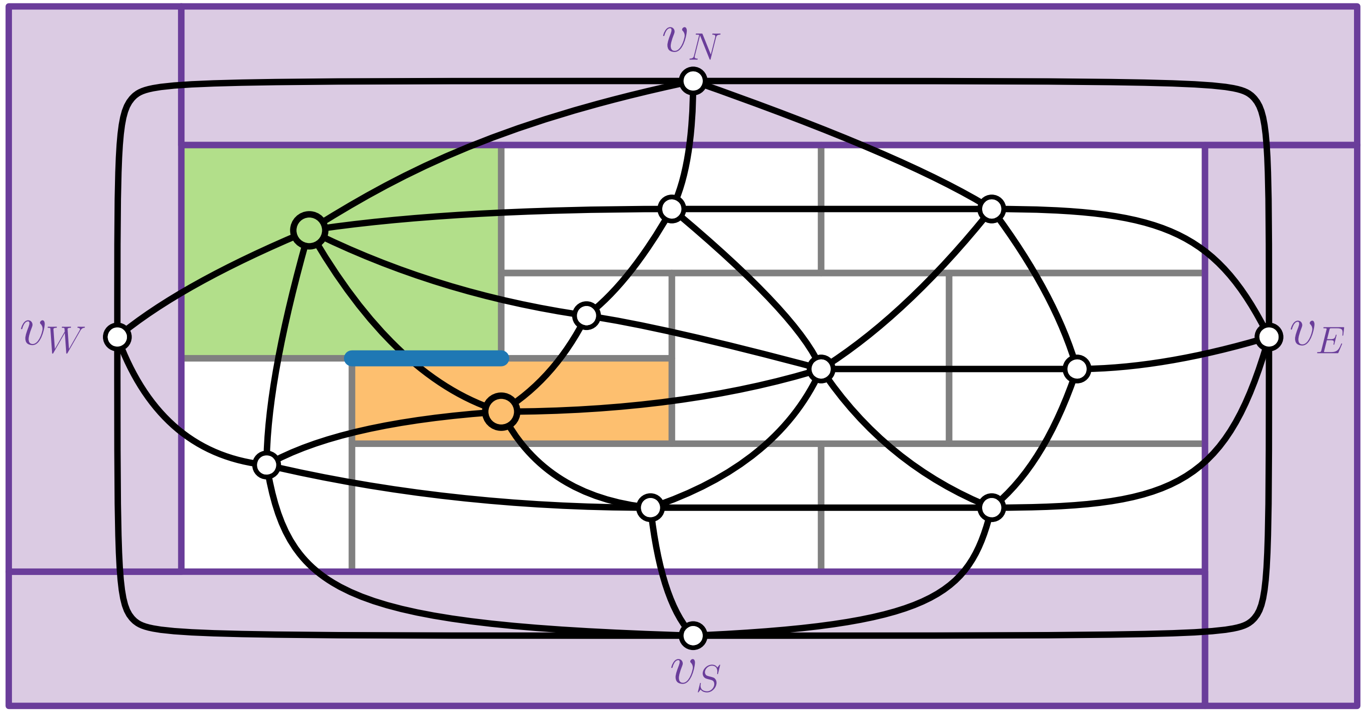
Regular Edge Labeling



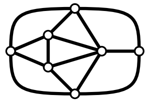
Properly Triangulated Planar Graph G



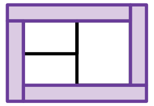
Rectangular Dual $\mathcal{R}$



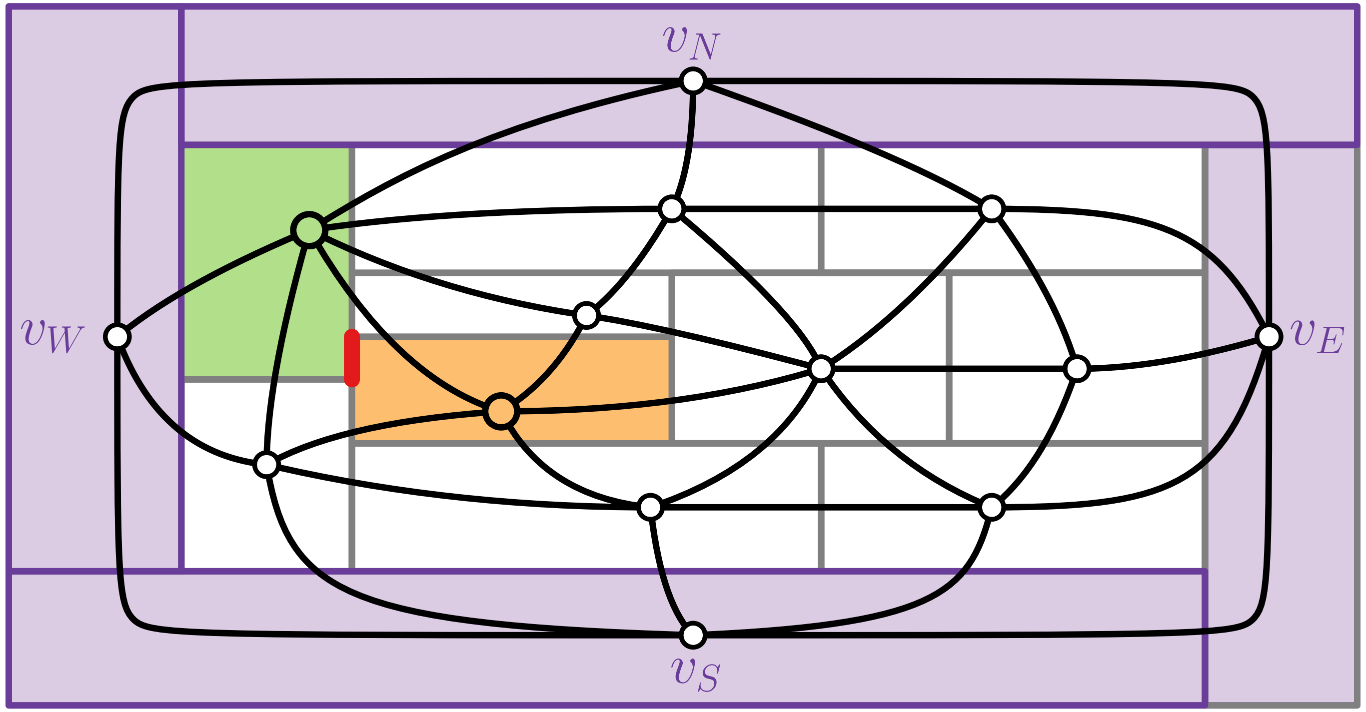
Regular Edge Labeling



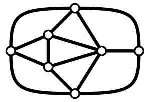
Properly Triangulated Planar Graph G



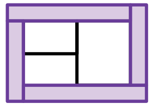
Rectangular Dual $\mathcal{R}$



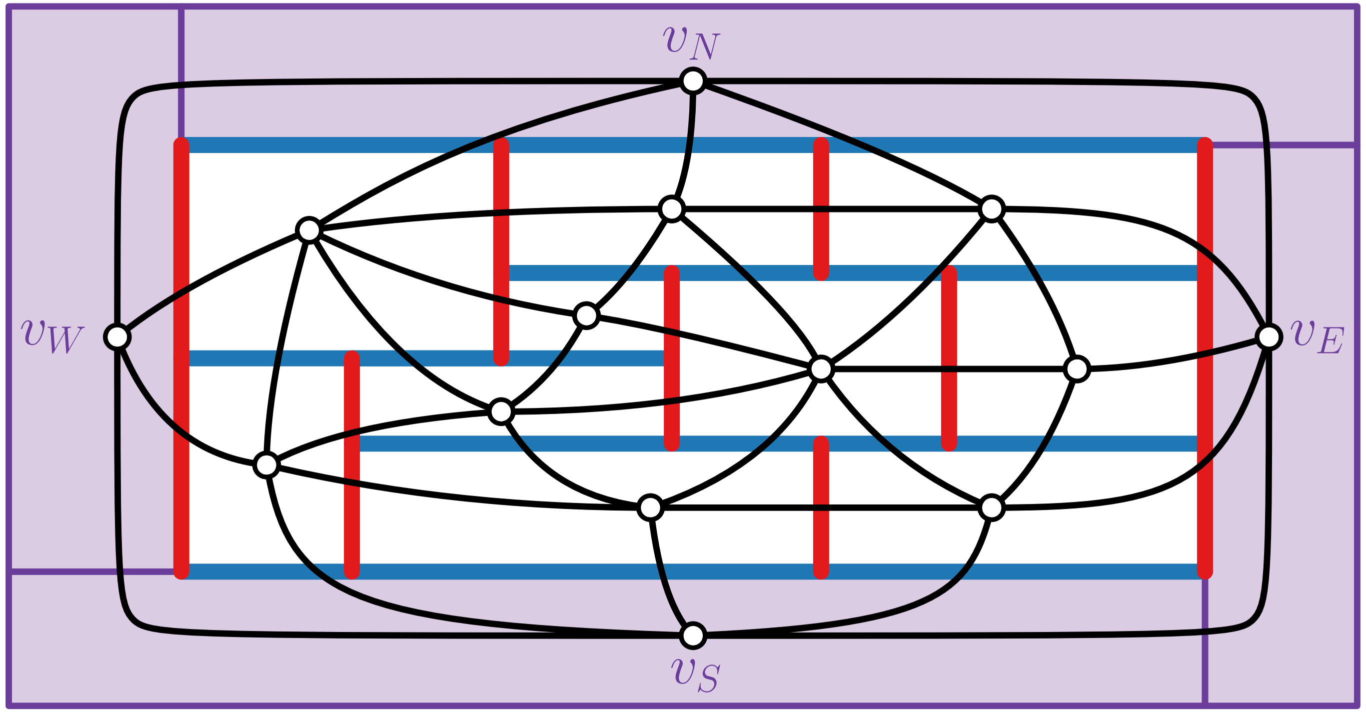
Regular Edge Labeling



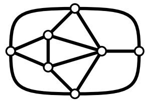
Properly Triangulated Planar Graph G



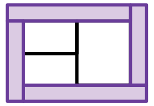
Rectangular Dual $\mathcal{R}$



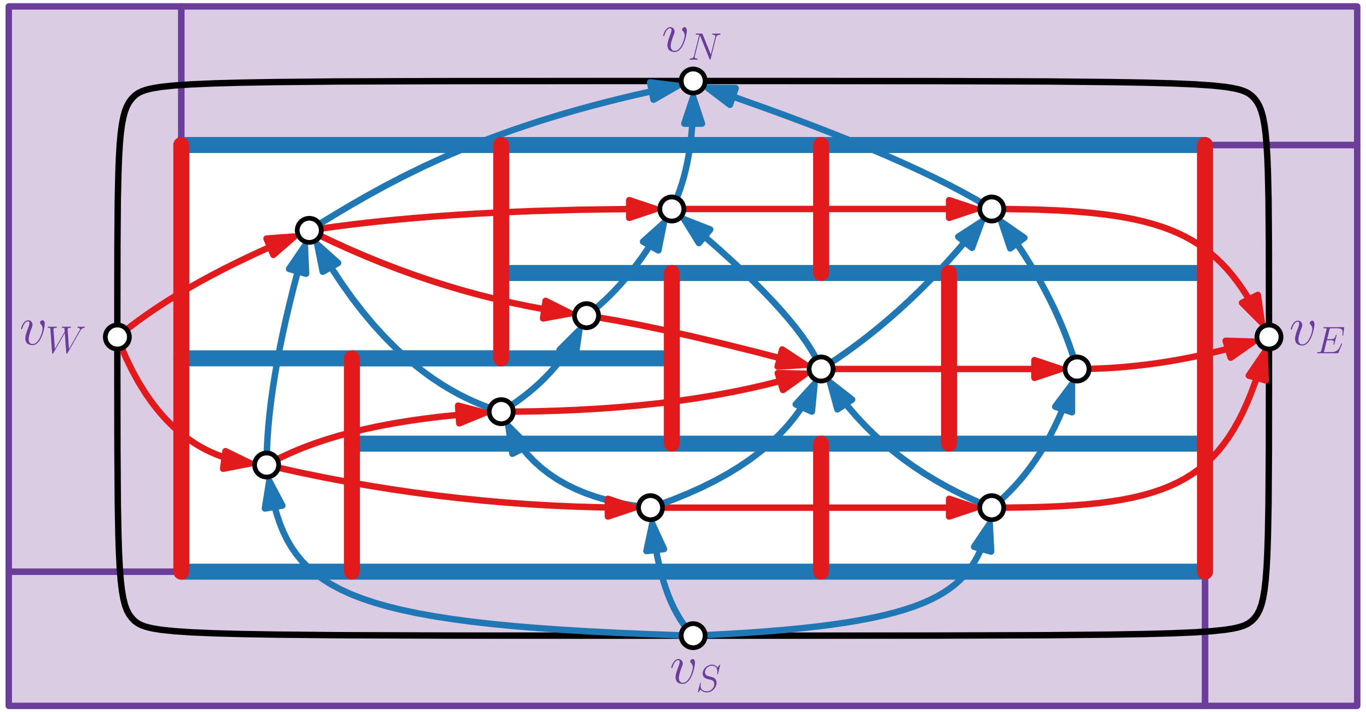
Regular Edge Labeling



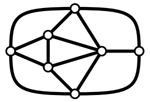
Properly Triangulated Planar Graph G



Rectangular Dual $\mathcal{R}$

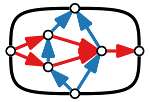


Regular Edge Labeling



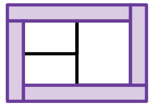
PTP

Properly Triangulated Planar Graph G



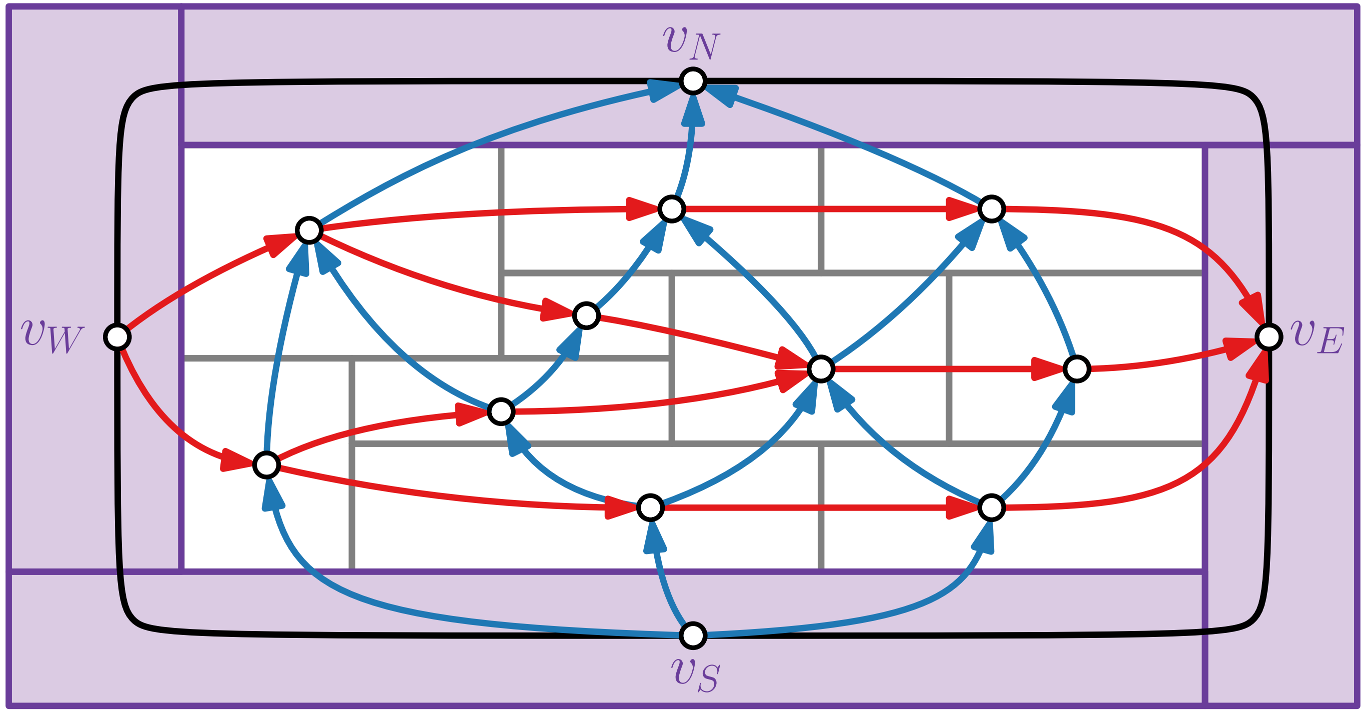
REL

Regular Edge Labeling

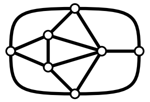


RD

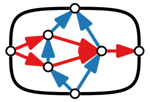
Rectangular Dual $\mathcal{R}$



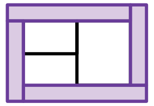
Regular Edge Labeling



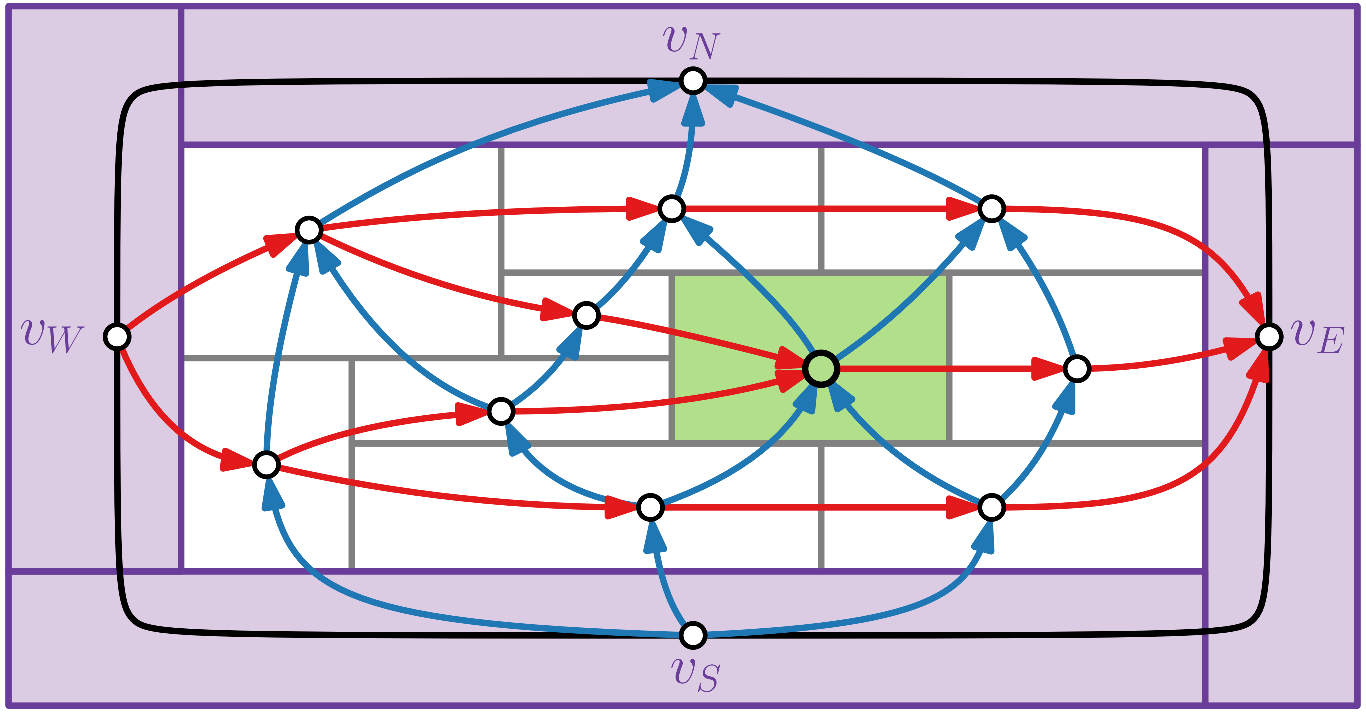
Properly Triangulated Planar Graph G



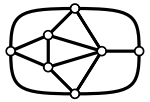
Regular Edge Labeling



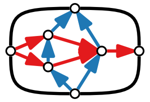
Rectangular Dual $\mathcal{R}$



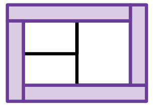
Regular Edge Labeling


PTP

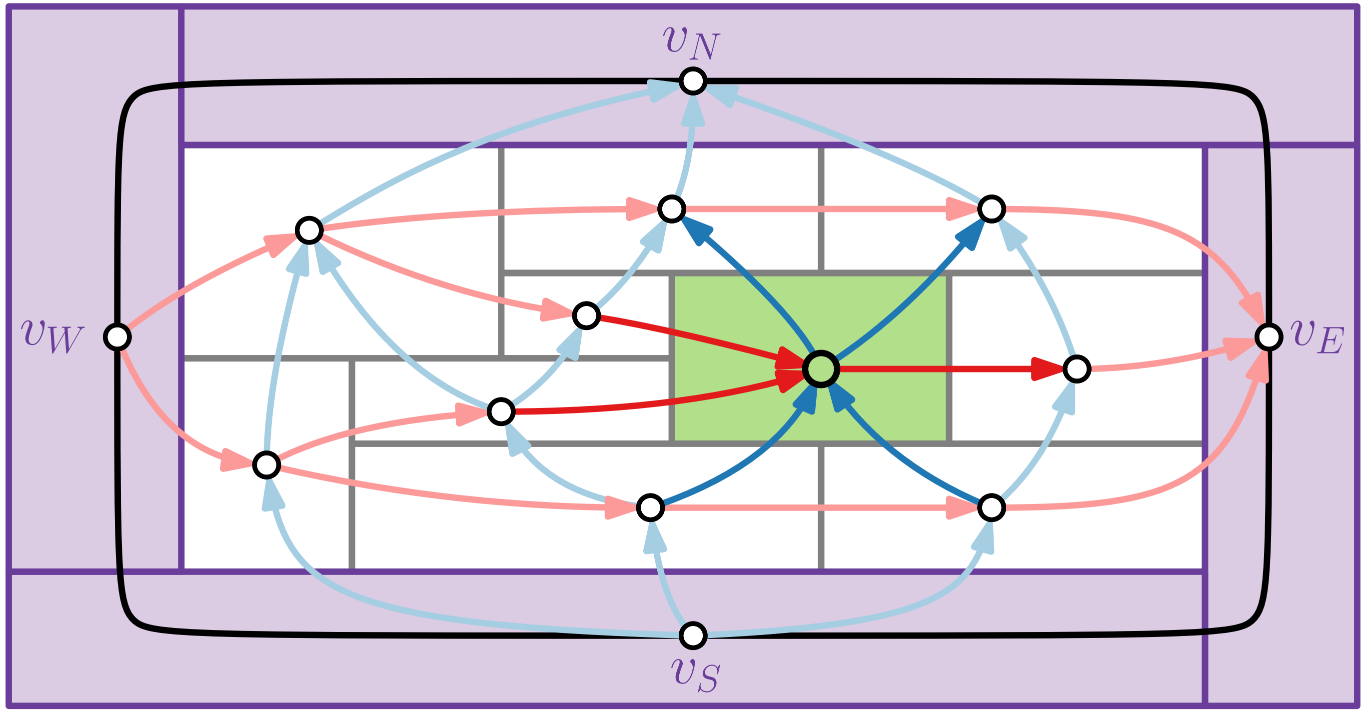
Properly Triangulated Planar Graph G


REL

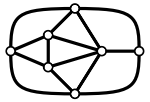
Regular Edge Labeling


RD

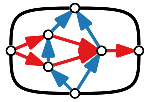
Rectangular Dual $\mathcal{R}$



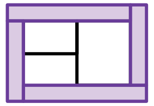
Regular Edge Labeling


PTP

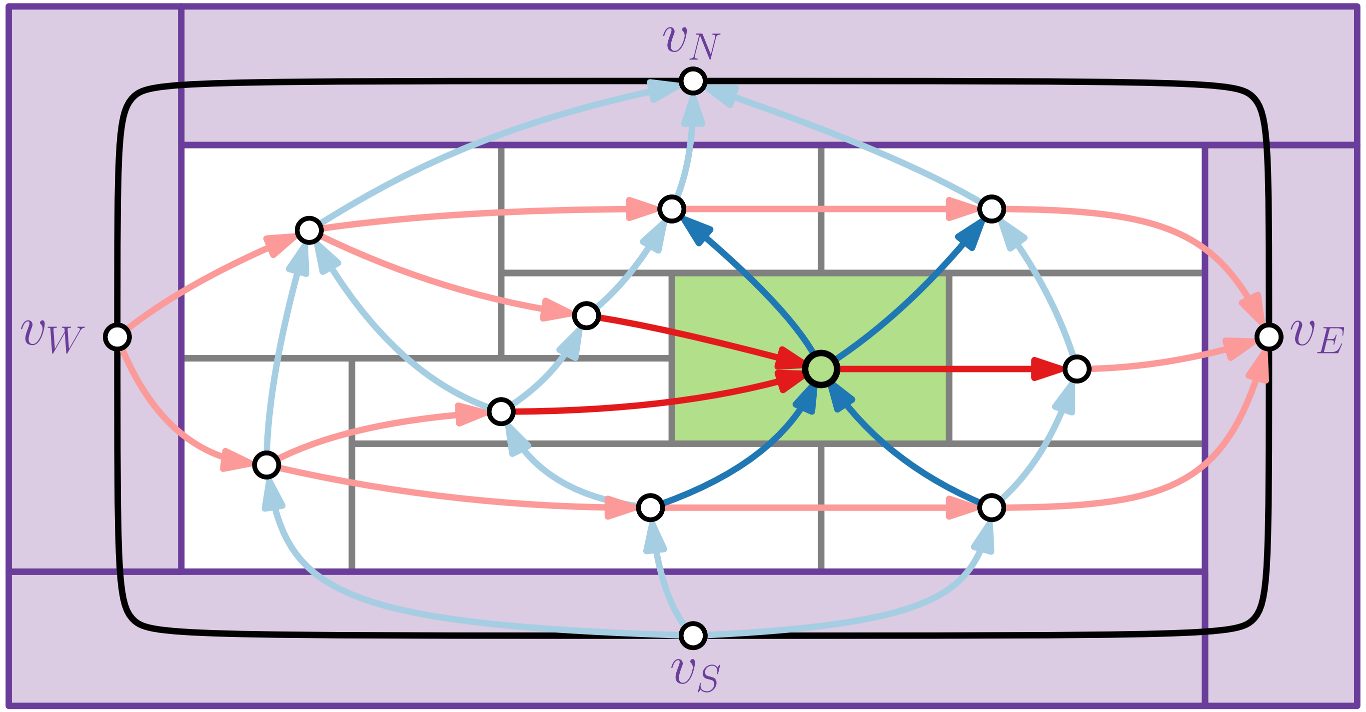
Properly Triangulated Planar Graph G

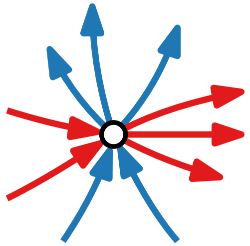

REL

Regular Edge Labeling


RD

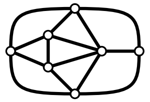
Rectangular Dual $\mathcal{R}$



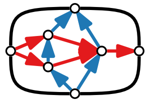


for every inner vertex

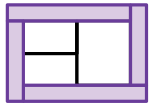
Regular Edge Labeling



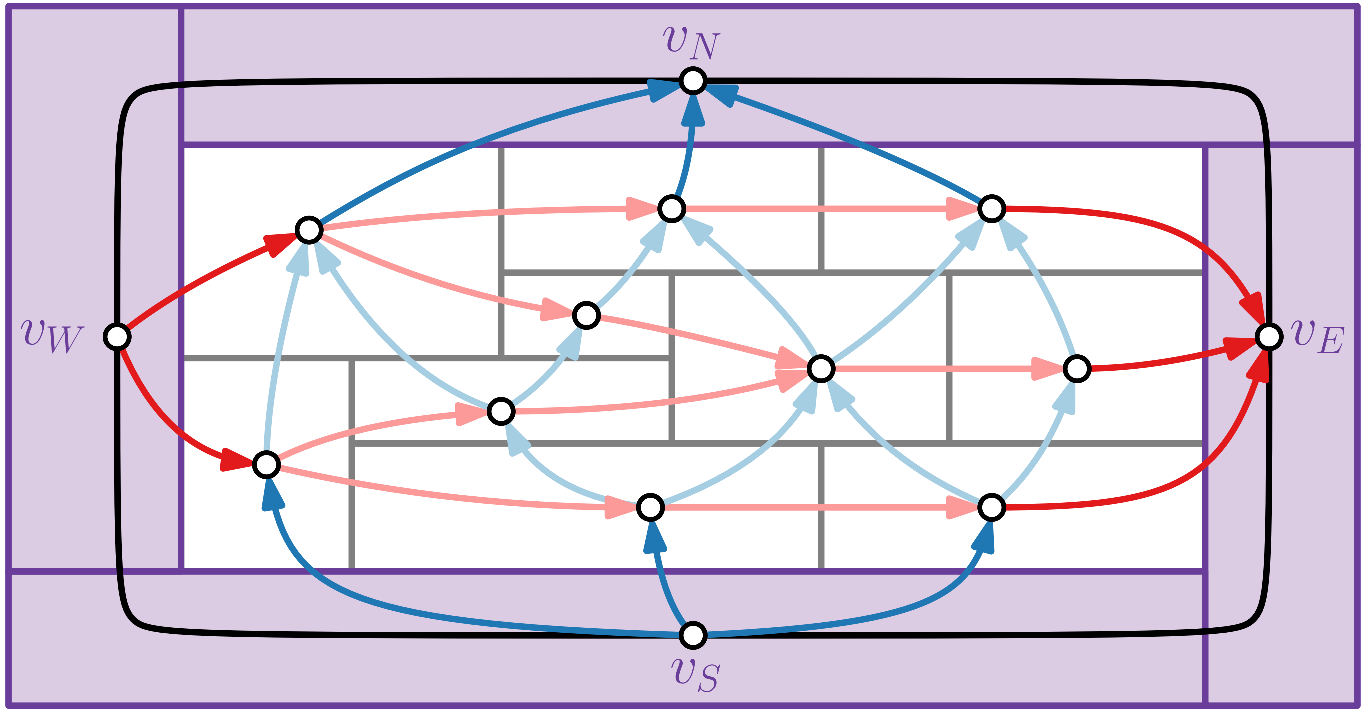
Properly Triangulated Planar Graph G

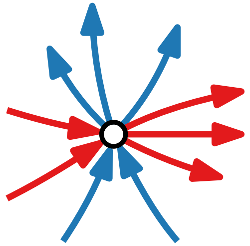


Regular Edge Labeling



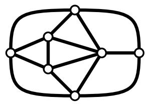
Rectangular Dual $\mathcal{R}$



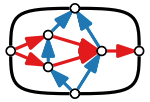


for every inner vertex

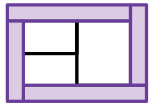
Regular Edge Labeling



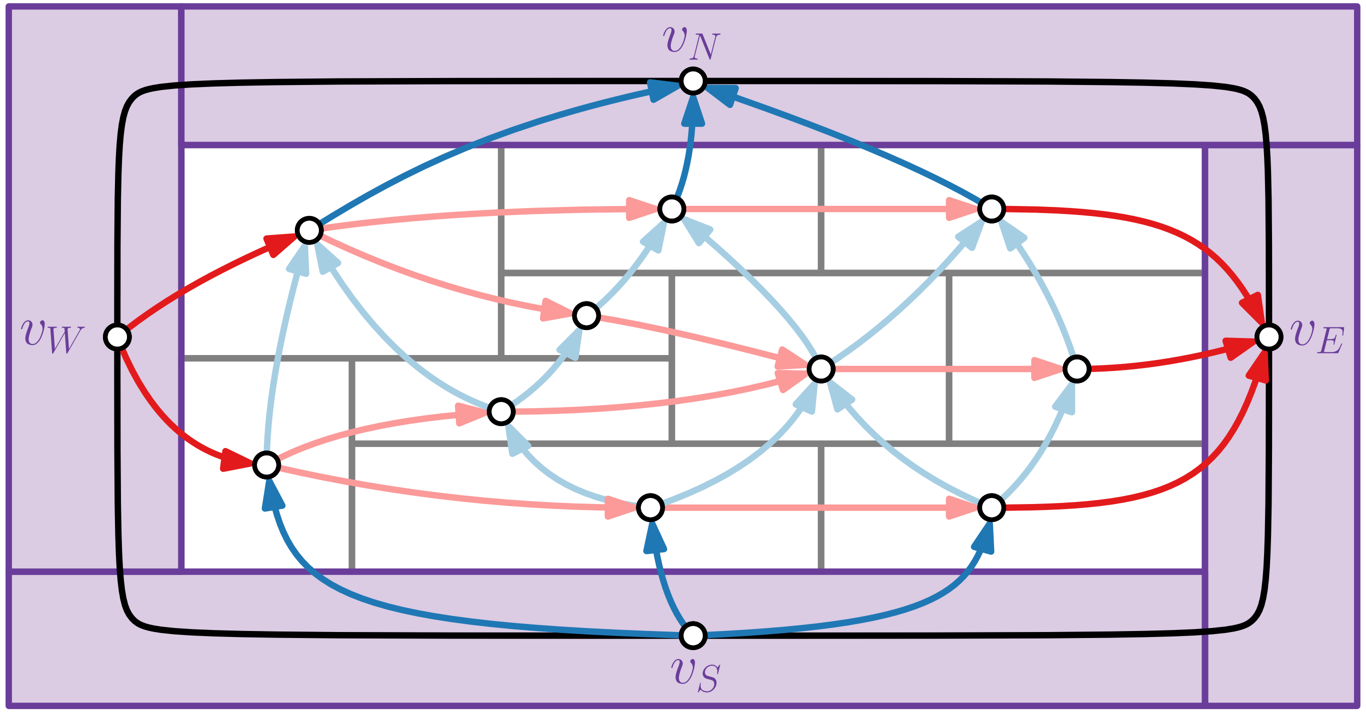
Properly Triangulated Planar Graph G

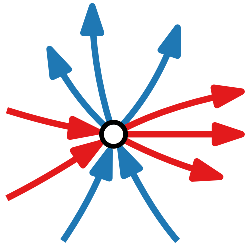


Regular Edge Labeling

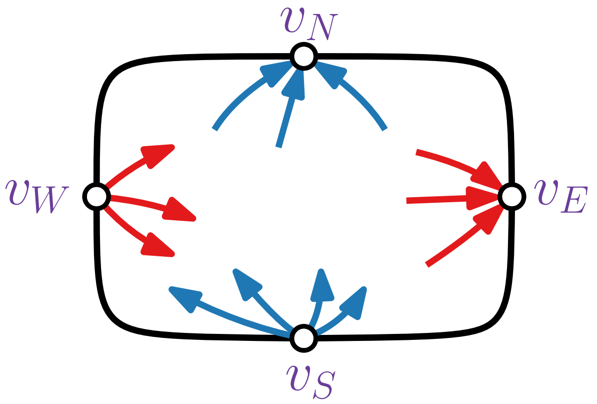


Rectangular Dual $\mathcal{R}$



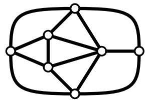


for every inner vertex

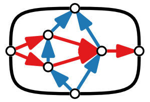


for four outer vertices

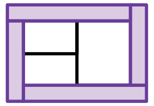
Regular Edge Labeling



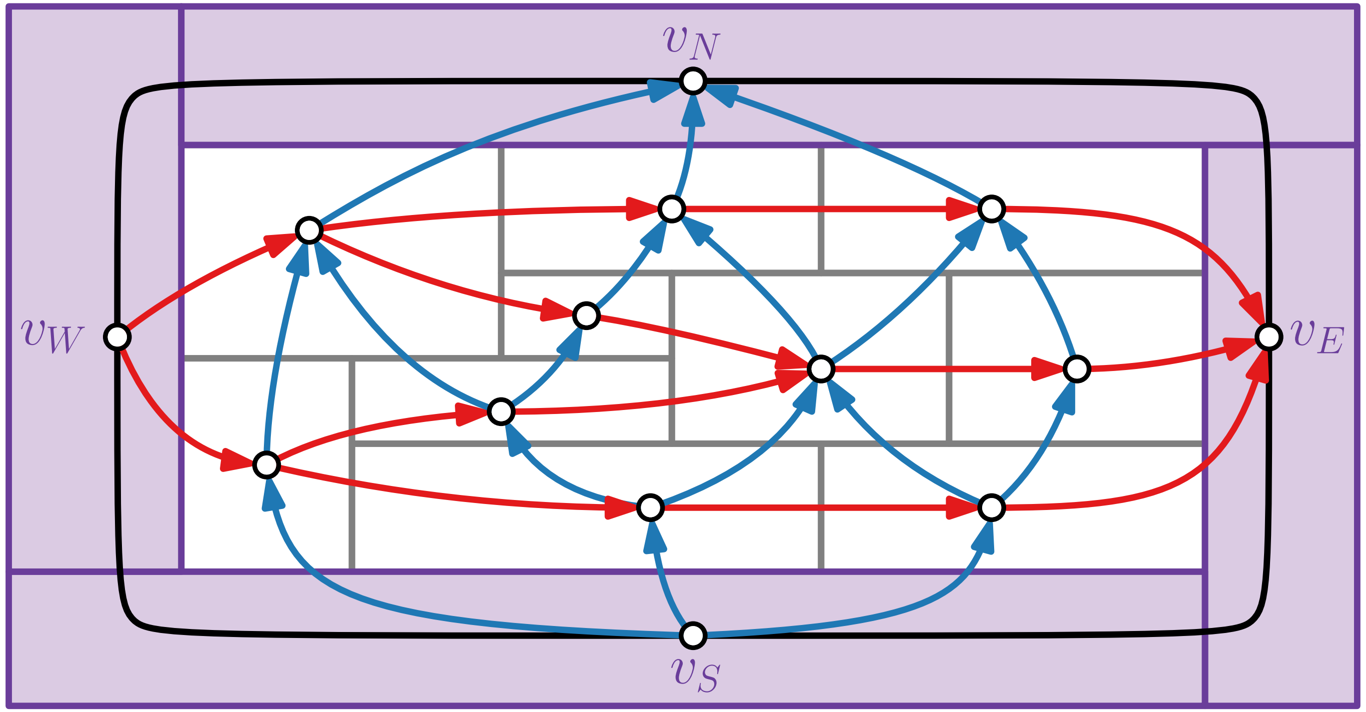
Properly Triangulated Planar Graph G

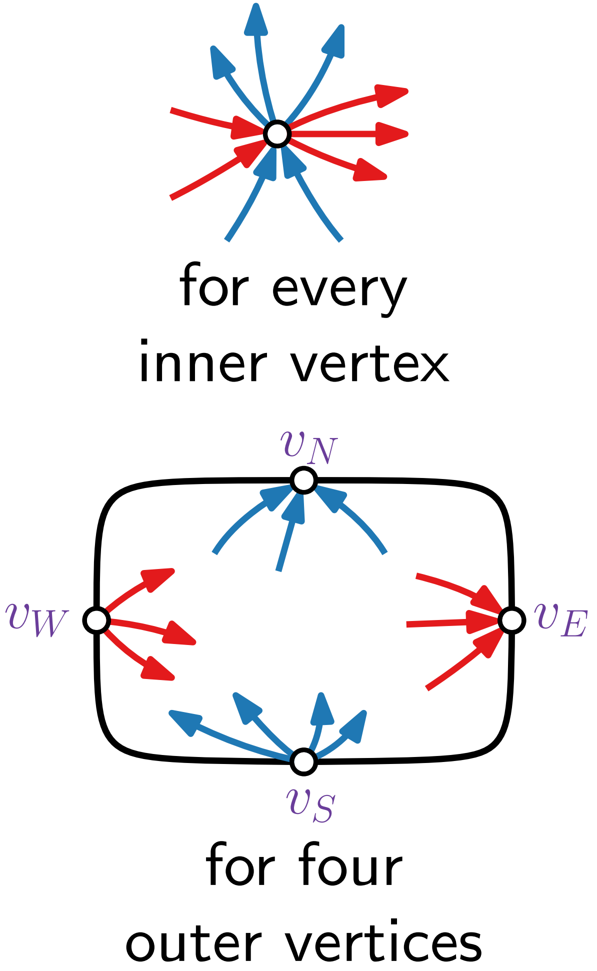


Regular Edge Labeling

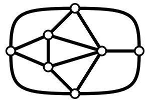


Rectangular Dual $\mathcal{R}$

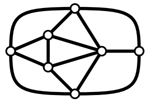


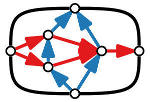


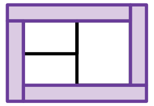
[Kant, He '94]: In linear time

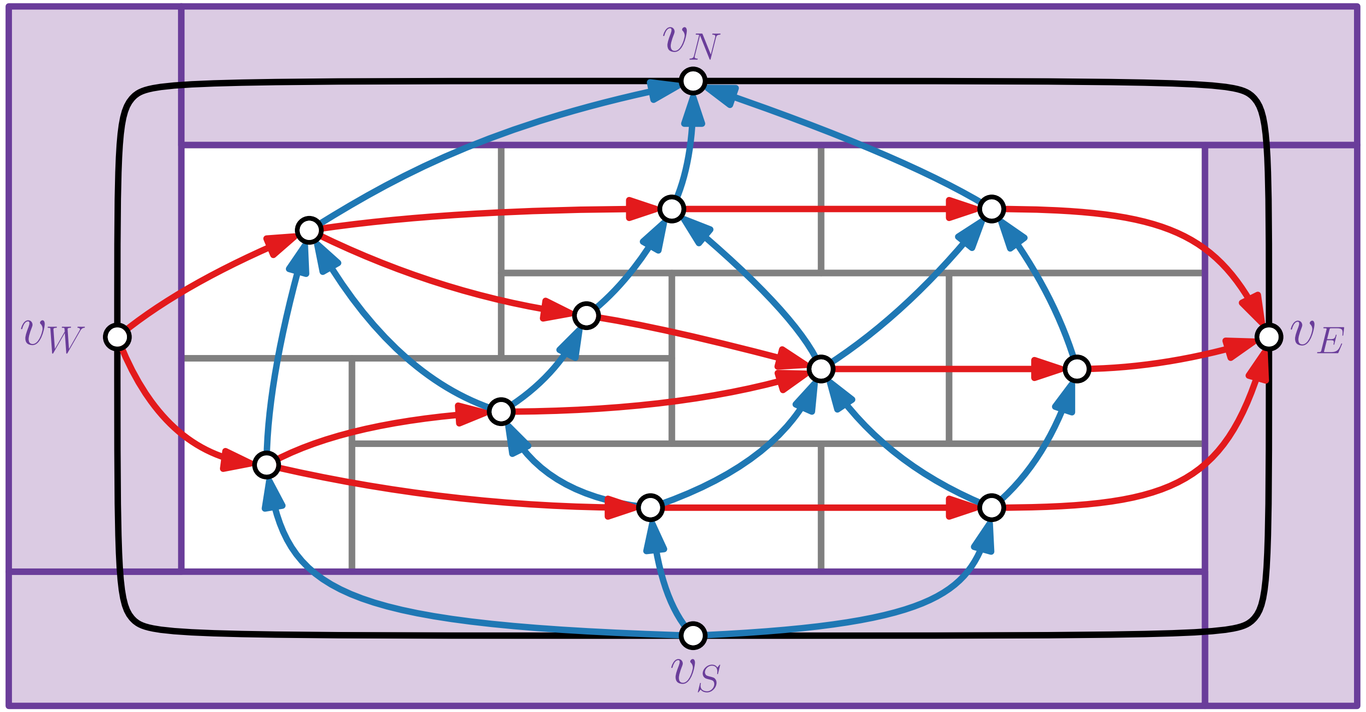


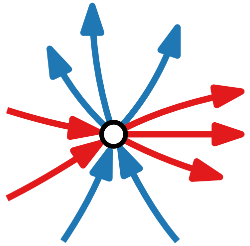
Regular Edge Labeling


PTP
Properly Triangulated Planar Graph G

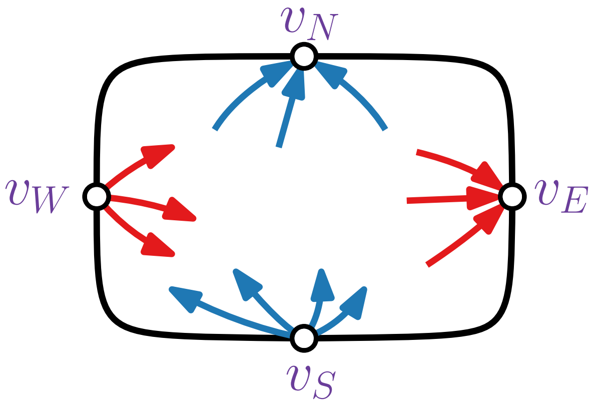

REL
Regular Edge Labeling


RD
Rectangular Dual $\mathcal{R}$



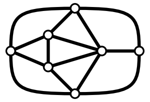


for every inner vertex

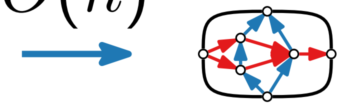


for four outer vertices

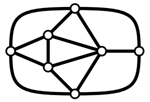
[Kant, He '94]: In linear time


PTP

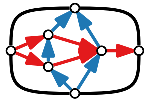
$O(n)$


REL

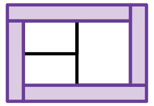
Regular Edge Labeling



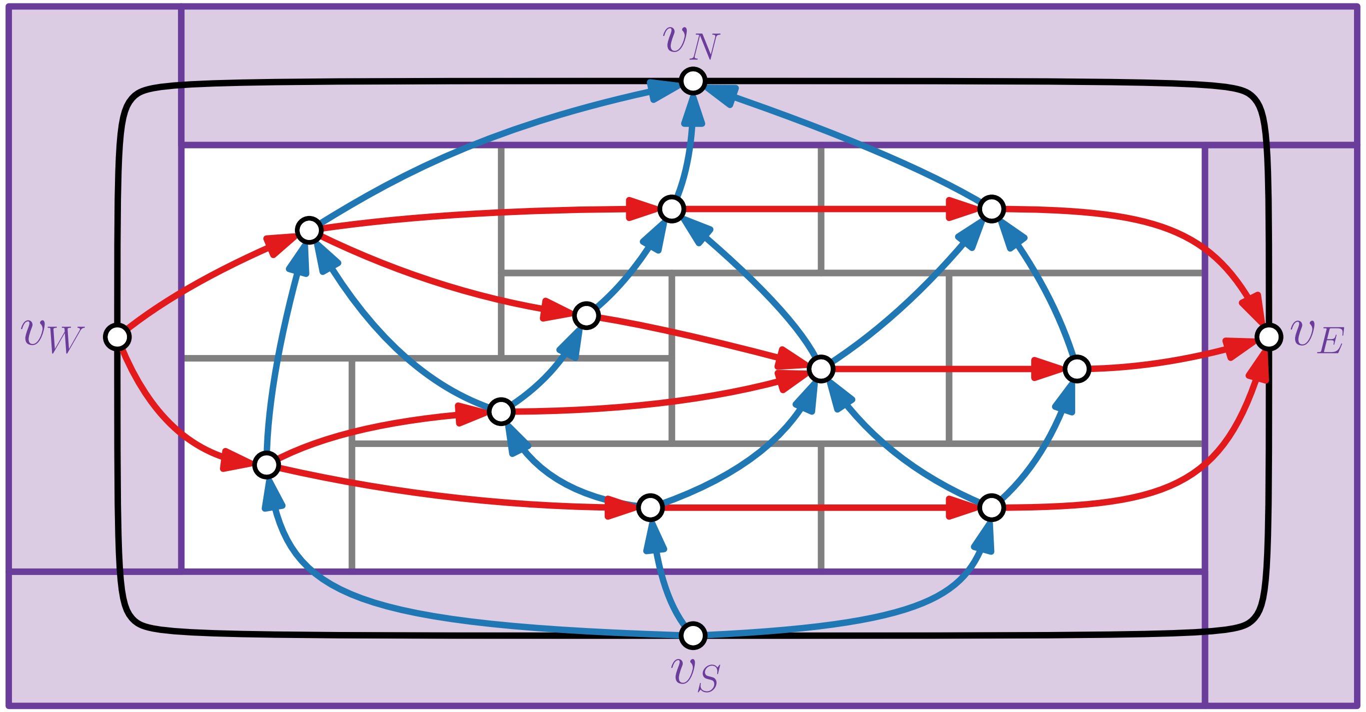
Properly Triangulated Planar Graph G

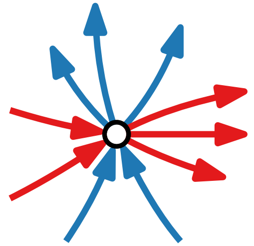


Regular Edge Labeling

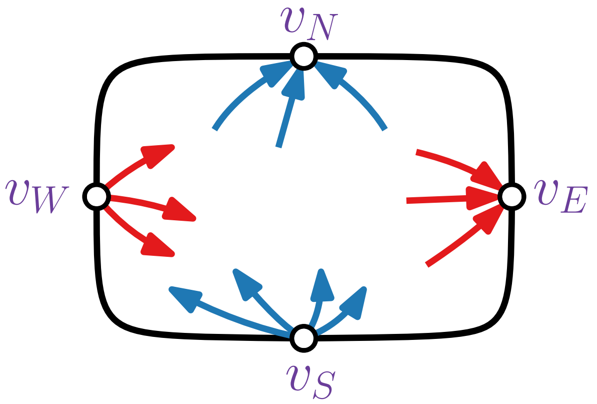


Rectangular Dual $\mathcal{R}$



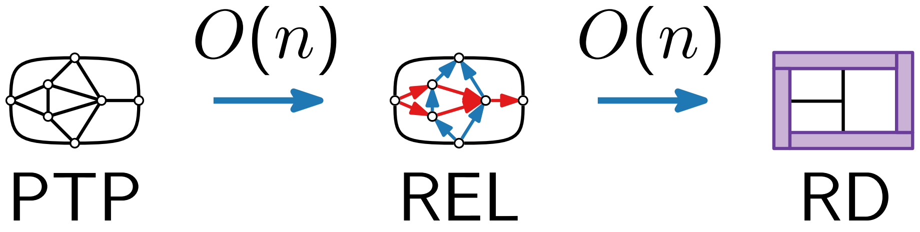


for every inner vertex



for four outer vertices

[Kant, He '94]: In linear time



Refined Canonical Order

Theorem.
Let G be a PTP graph.

Refined Canonical Order

Theorem.
Let G be a PTP graph. There exists a labeling
$v_1 = v_S, v_2 = v_W, v_3, \ldots, v_n = v_N$ of the vertices of G

Refined Canonical Order

Theorem.
Let G be a PTP graph. There exists a labeling $v_1 = v_S, v_2 = v_W, v_3, \ldots, v_n = v_N$ of the vertices of G such that for every $4 \leq k \leq n$:

Refined Canonical Order

Theorem.
Let G be a PTP graph. There exists a labeling $v_1 = v_S, v_2 = v_W, v_3, \ldots, v_n = v_N$ of the vertices of G such that for every $4 \leq k \leq n$:

- The subgraph G_{k-1} induced by $v_1, \ldots, v_{k-1}$ is biconnected

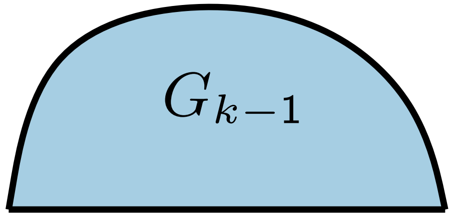
.

Refined Canonical Order

Theorem.
Let G be a PTP graph. There exists a labeling $v_1 = v_S, v_2 = v_W, v_3, \ldots, v_n = v_N$ of the vertices of G such that for every $4 \leq k \leq n$:

- The subgraph G_{k-1} induced by $v_1, \ldots, v_{k-1}$ is biconnected

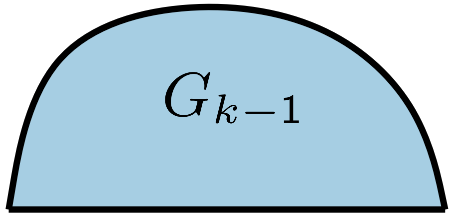
.



Refined Canonical Order

Theorem.
Let G be a PTP graph. There exists a labeling $v_1 = v_S, v_2 = v_W, v_3, \ldots, v_n = v_N$ of the vertices of G such that for every $4 \leq k \leq n$:

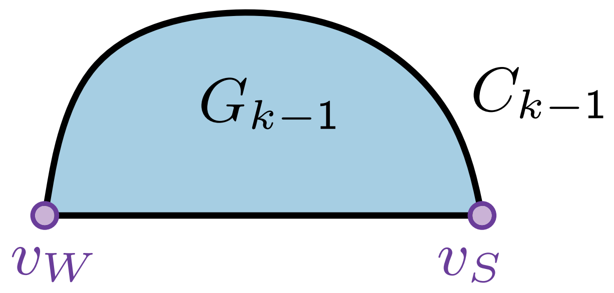
- The subgraph G_{k-1} induced by $v_1, \ldots, v_{k-1}$ is biconnected and the boundary C_{k-1} of G_{k-1} contains the edge (v_S, v_W).



Refined Canonical Order

Theorem.
Let G be a PTP graph. There exists a labeling $v_1 = v_S, v_2 = v_W, v_3, \dots, v_n = v_N$ of the vertices of G such that for every $4 \leq k \leq n$:

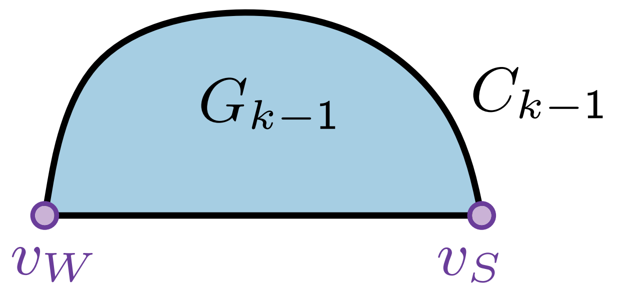
- The subgraph G_{k-1} induced by $v_1, \dots, v_{k-1}$ is biconnected and the boundary C_{k-1} of G_{k-1} contains the edge (v_S, v_W).



Refined Canonical Order

Theorem.
Let G be a PTP graph. There exists a labeling $v_1 = v_S, v_2 = v_W, v_3, \ldots, v_n = v_N$ of the vertices of G such that for every $4 \leq k \leq n$:

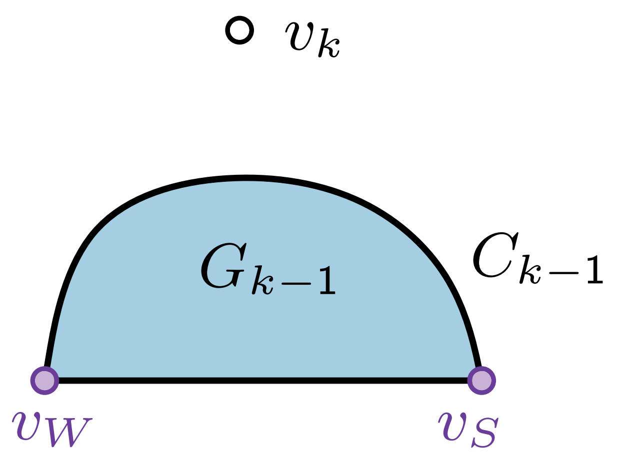
- The subgraph G_{k-1} induced by $v_1, \ldots, v_{k-1}$ is biconnected and the boundary C_{k-1} of G_{k-1} contains the edge (v_S, v_W).
- v_k is in exterior face of G_{k-1}



Refined Canonical Order

Theorem.
Let G be a PTP graph. There exists a labeling $v_1 = v_S, v_2 = v_W, v_3, \dots, v_n = v_N$ of the vertices of G such that for every $4 \leq k \leq n$:

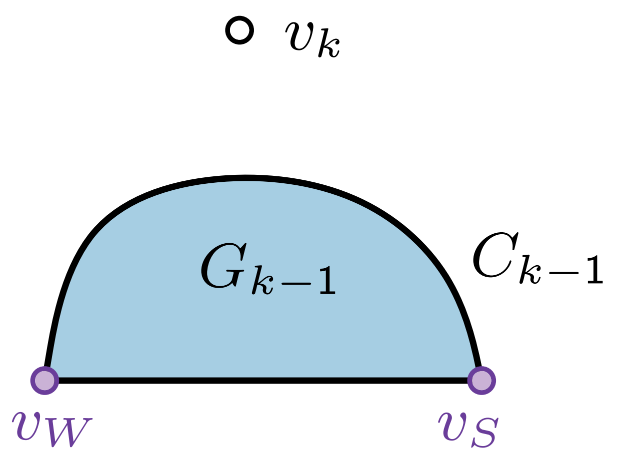
- The subgraph G_{k-1} induced by $v_1, \dots, v_{k-1}$ is biconnected and the boundary C_{k-1} of G_{k-1} contains the edge (v_S, v_W).
- v_k is in exterior face of G_{k-1}



Refined Canonical Order

Theorem.
Let G be a PTP graph. There exists a labeling $v_1 = v_S, v_2 = v_W, v_3, \ldots, v_n = v_N$ of the vertices of G such that for every $4 \leq k \leq n$:

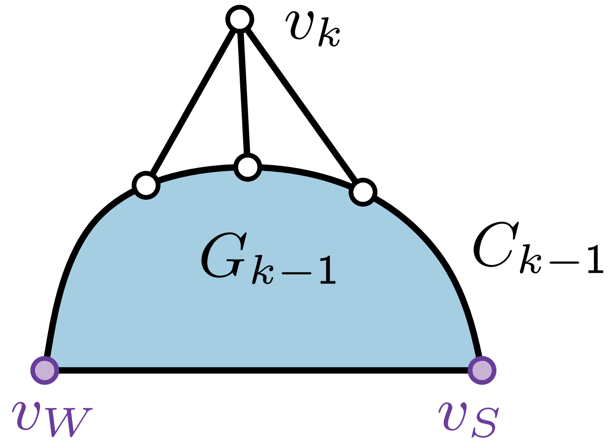
- The subgraph G_{k-1} induced by $v_1, \ldots, v_{k-1}$ is biconnected and the boundary C_{k-1} of G_{k-1} contains the edge (v_S, v_W).
- v_k is in exterior face of G_{k-1}, and its neighbors in G_{k-1} form an (at least 2-element) subinterval of the path $C_{k-1} \setminus (v_S, v_W)$.



Refined Canonical Order

Theorem.
Let G be a PTP graph. There exists a labeling $v_1 = v_S, v_2 = v_W, v_3, \dots, v_n = v_N$ of the vertices of G such that for every $4 \le k \le n$:

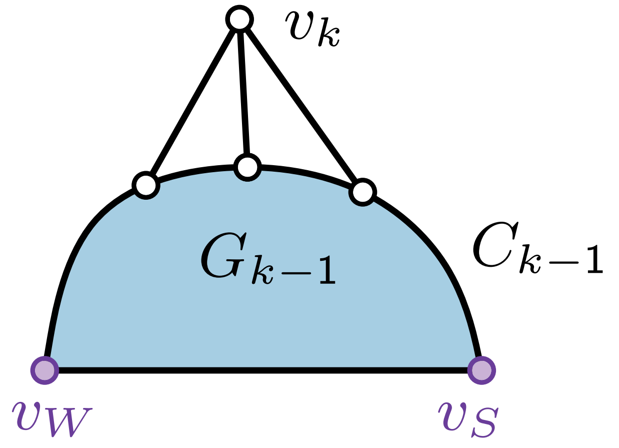
- The subgraph G_{k-1} induced by $v_1, \dots, v_{k-1}$ is biconnected and the boundary C_{k-1} of G_{k-1} contains the edge (v_S, v_W).
- v_k is in exterior face of G_{k-1}, and its neighbors in G_{k-1} form an (at least 2-element) subinterval of the path $C_{k-1} \setminus (v_S, v_W)$.



Refined Canonical Order

Theorem.
Let G be a PTP graph. There exists a labeling $v_1 = v_S, v_2 = v_W, v_3, \ldots, v_n = v_N$ of the vertices of G such that for every $4 \leq k \leq n$:

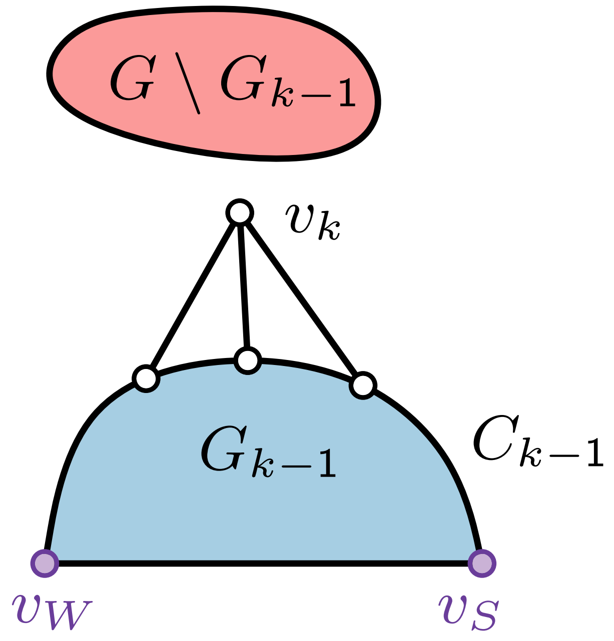
- The subgraph G_{k-1} induced by $v_1, \ldots, v_{k-1}$ is biconnected and the boundary C_{k-1} of G_{k-1} contains the edge (v_S, v_W).
- v_k is in exterior face of G_{k-1}, and its neighbors in G_{k-1} form an (at least 2-element) subinterval of the path $C_{k-1} \setminus (v_S, v_W)$.
- If $k \leq n-2$, then v_k has at least 2 neighbors in $G \setminus G_{k-1}$.



Refined Canonical Order

Theorem.
Let G be a PTP graph. There exists a labeling $v_1 = v_S, v_2 = v_W, v_3, \ldots, v_n = v_N$ of the vertices of G such that for every $4 \leq k \leq n$:

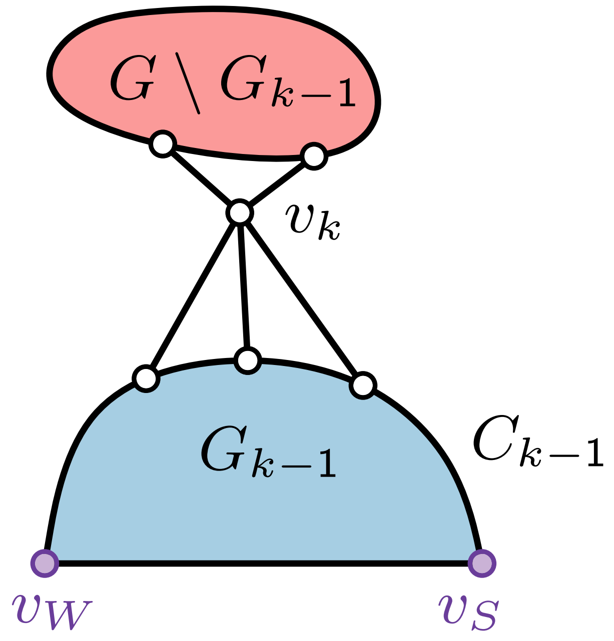
- The subgraph G_{k-1} induced by $v_1, \ldots, v_{k-1}$ is biconnected and the boundary C_{k-1} of G_{k-1} contains the edge (v_S, v_W).
- v_k is in exterior face of G_{k-1}, and its neighbors in G_{k-1} form an (at least 2-element) subinterval of the path $C_{k-1} \setminus (v_S, v_W)$.
- If $k \leq n-2$, then v_k has at least 2 neighbors in $G \setminus G_{k-1}$.



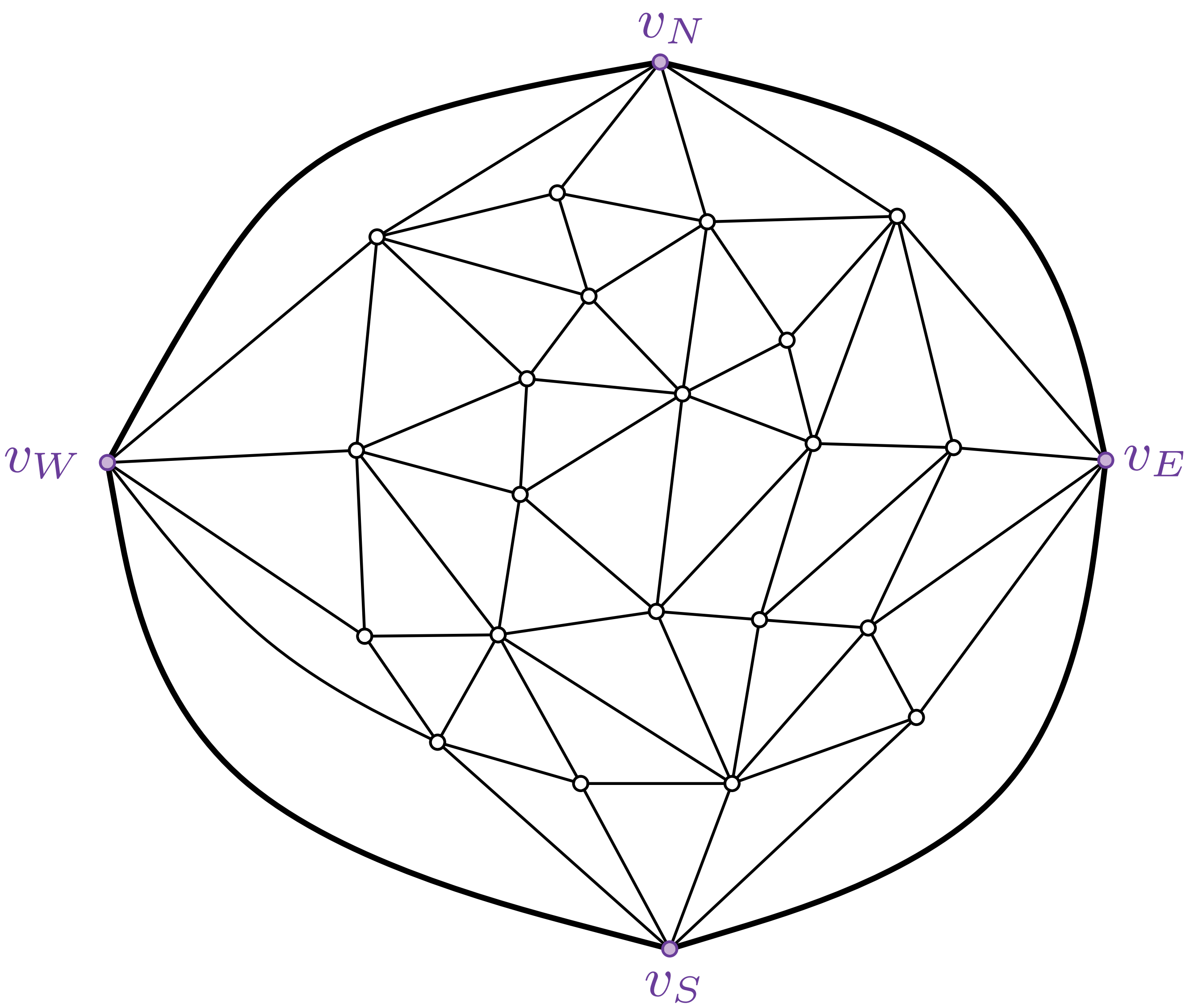
Refined Canonical Order

Theorem.
Let G be a PTP graph. There exists a labeling $v_1 = v_S, v_2 = v_W, v_3, \ldots, v_n = v_N$ of the vertices of G such that for every $4 \leq k \leq n$:

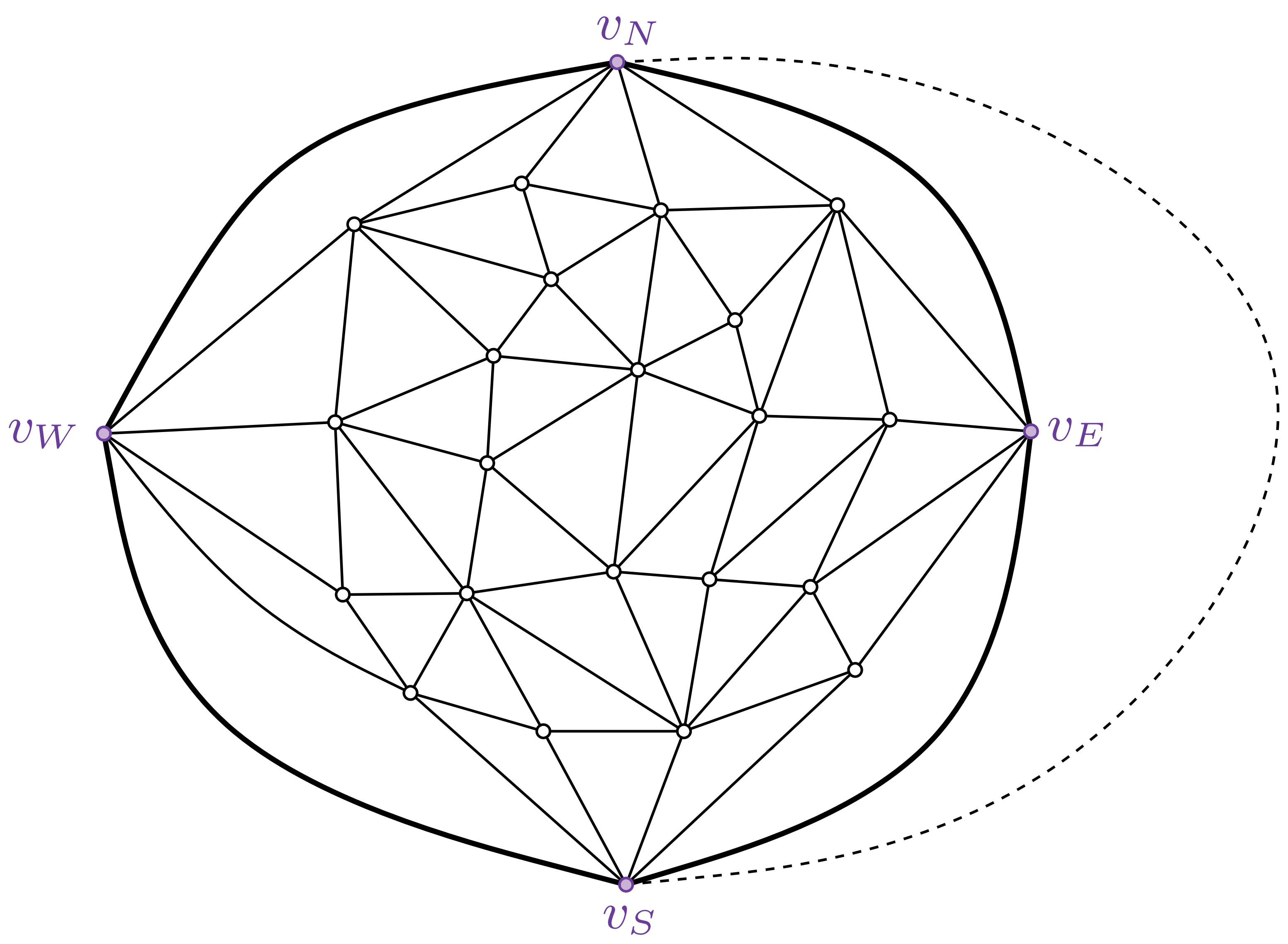
- The subgraph G_{k-1} induced by $v_1, \ldots, v_{k-1}$ is biconnected and the boundary C_{k-1} of G_{k-1} contains the edge (v_S, v_W).
- v_k is in exterior face of G_{k-1}, and its neighbors in G_{k-1} form an (at least 2-element) subinterval of the path $C_{k-1} \setminus (v_S, v_W)$.
- If $k \leq n-2$, then v_k has at least 2 neighbors in $G \setminus G_{k-1}$.



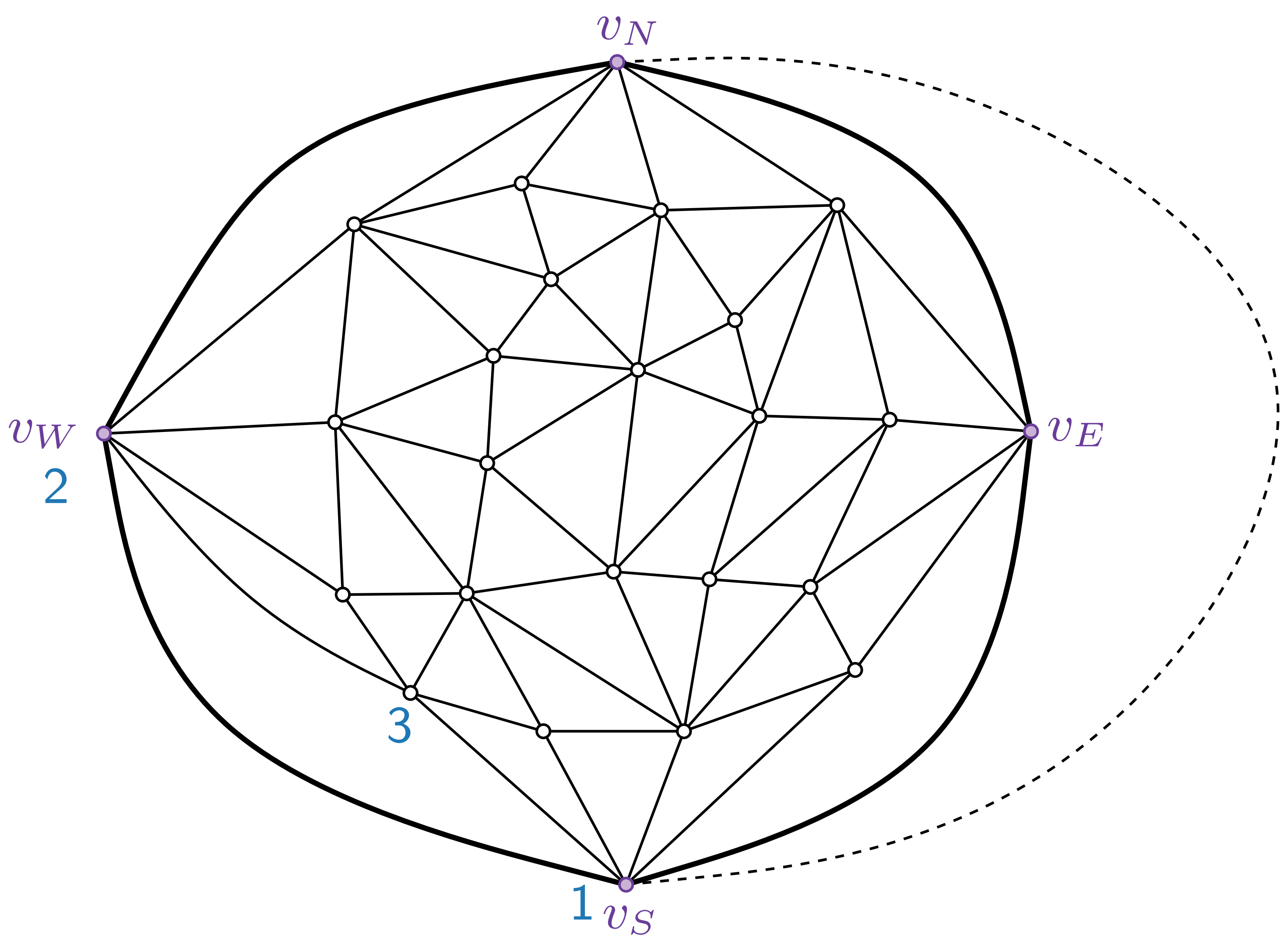
Refined Canonical Order Example



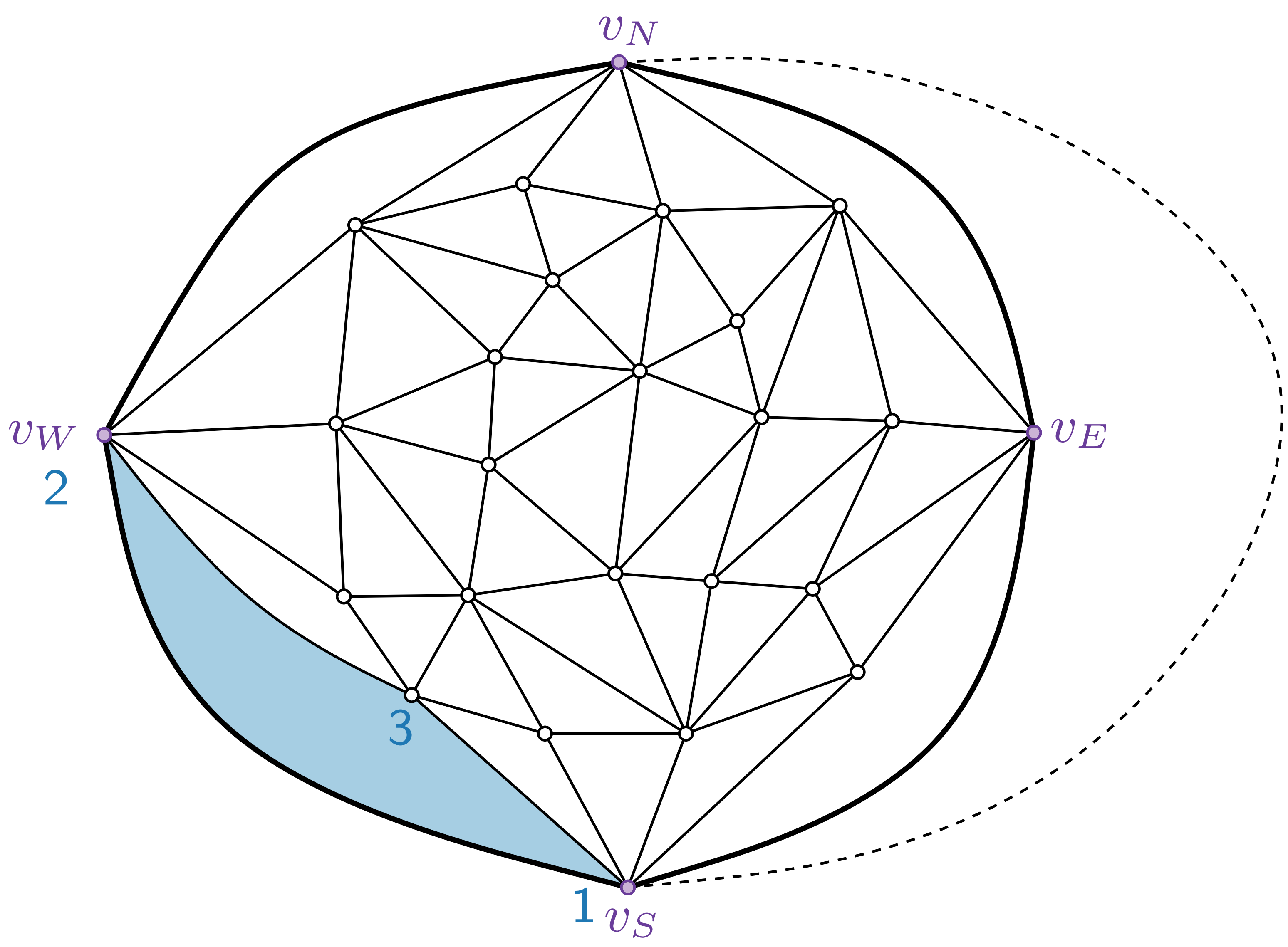
Refined Canonical Order Example



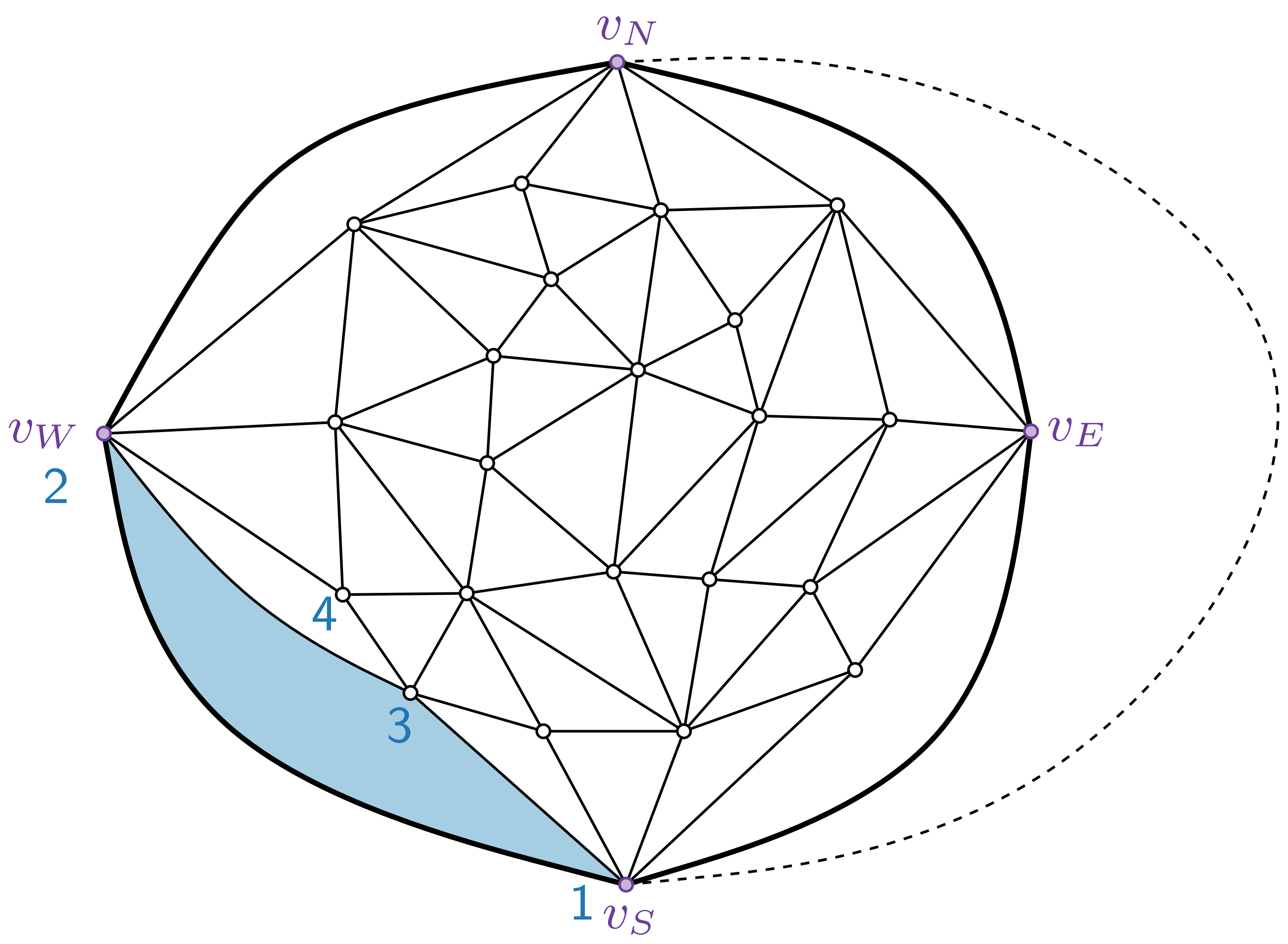
Refined Canonical Order Example



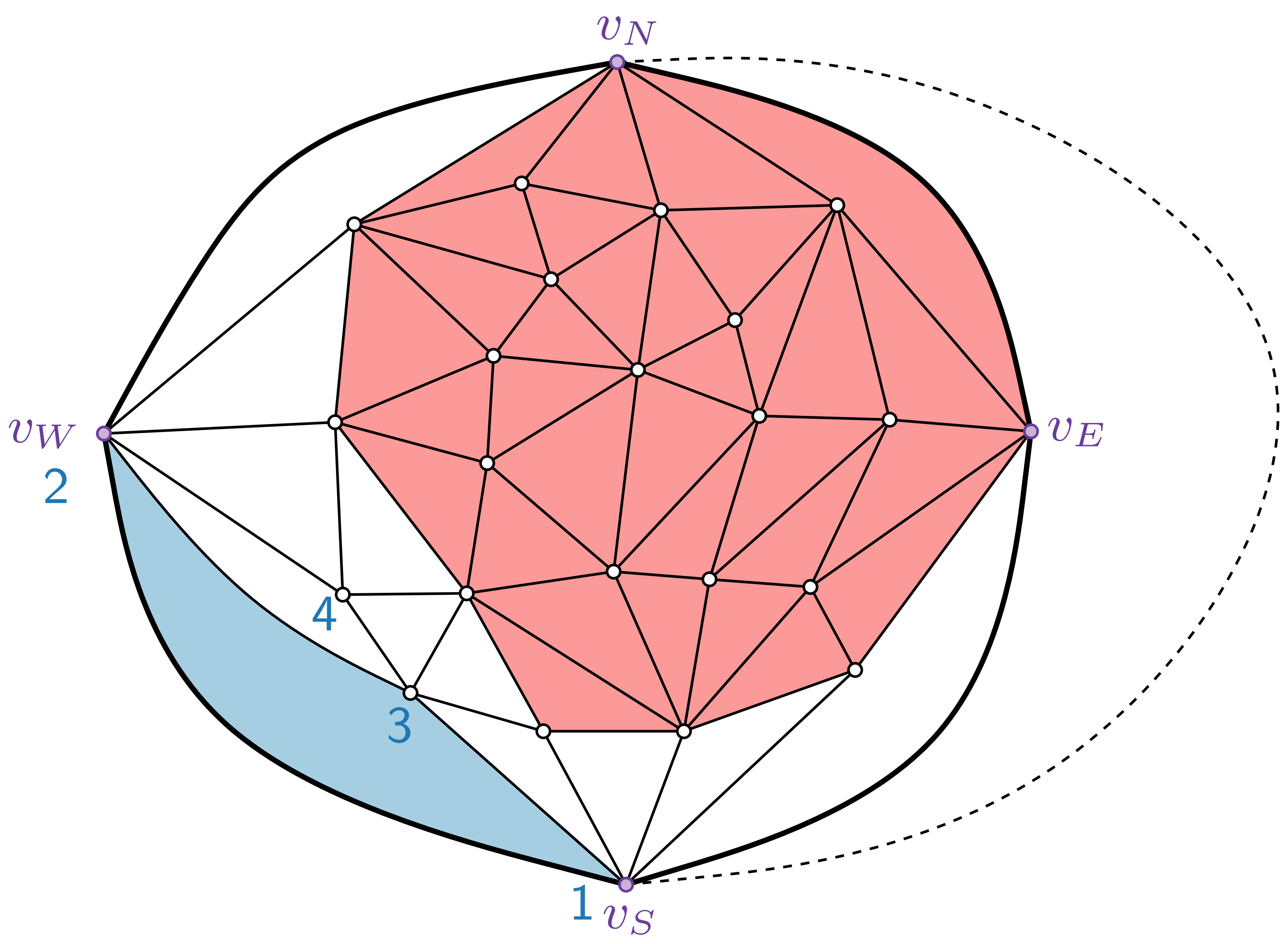
Refined Canonical Order Example



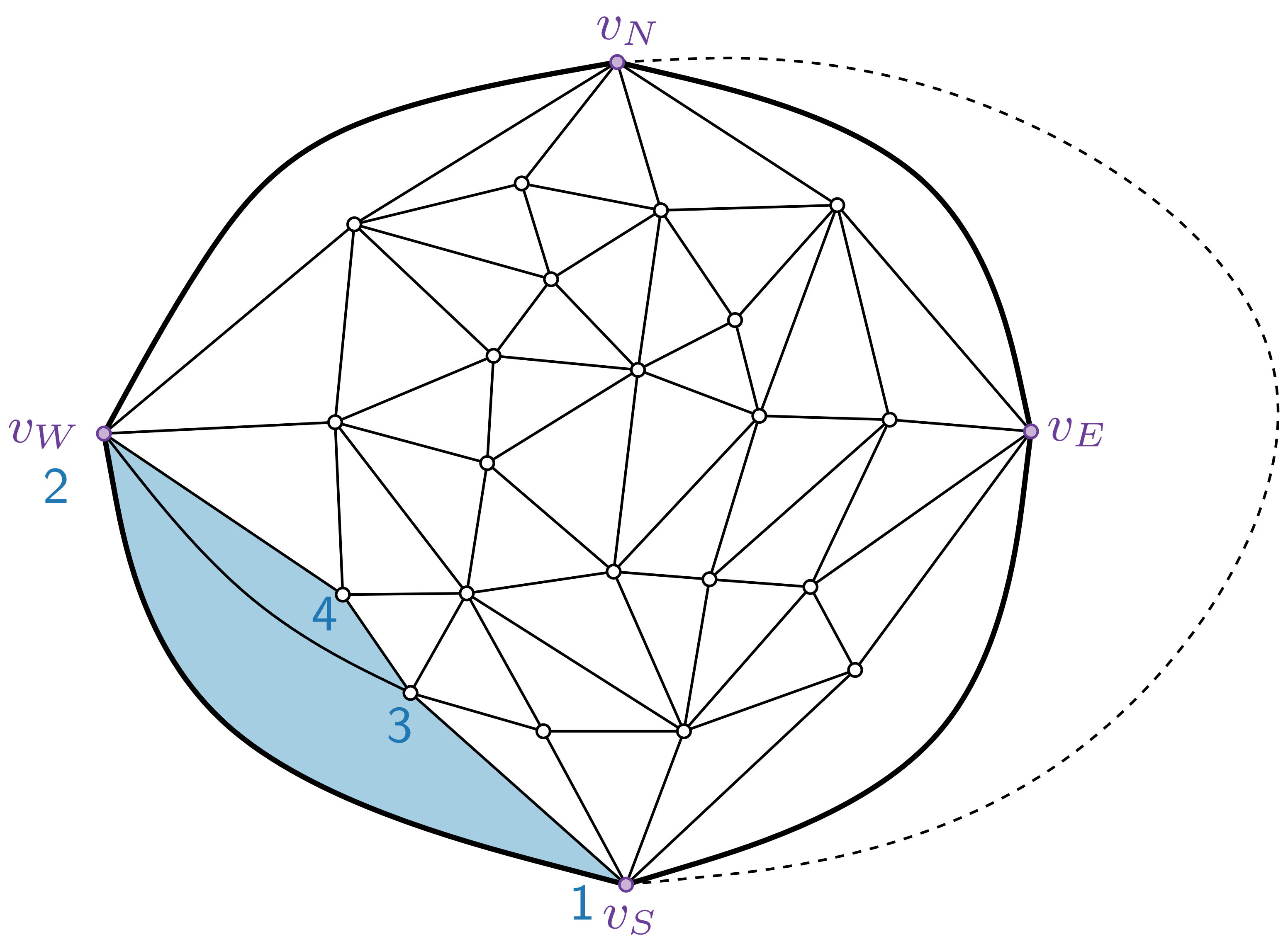
Refined Canonical Order Example



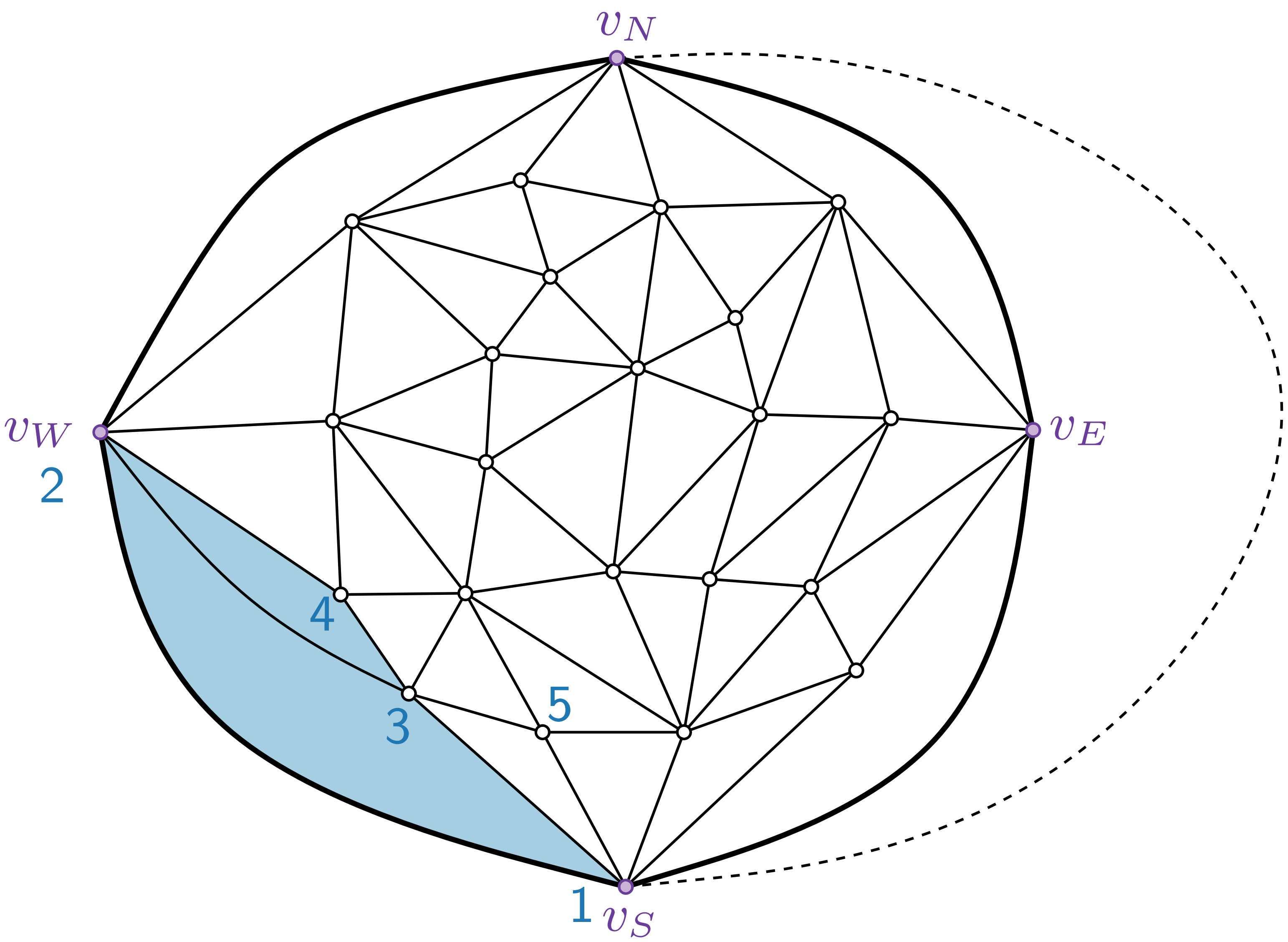
Refined Canonical Order Example



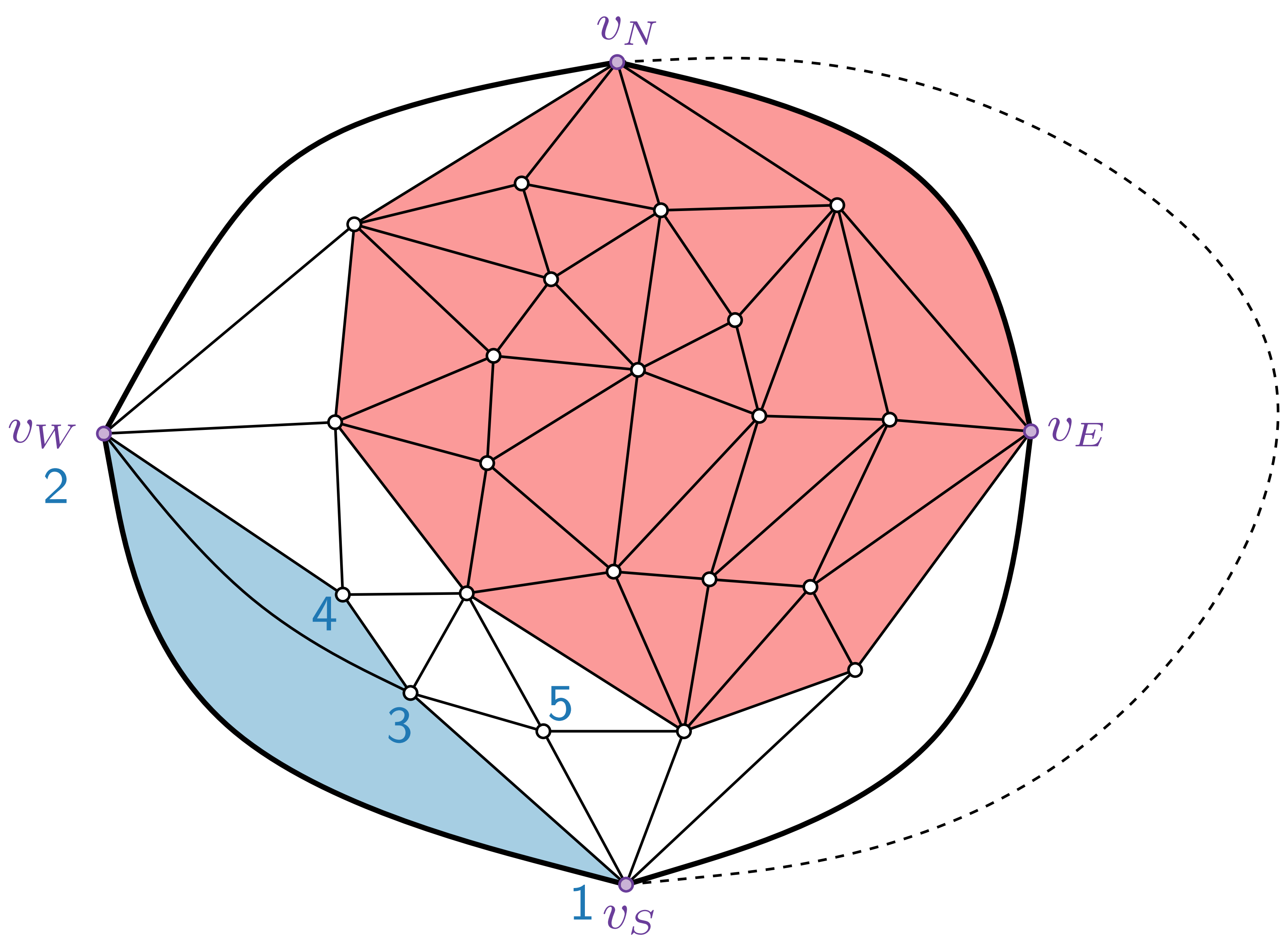
Refined Canonical Order Example



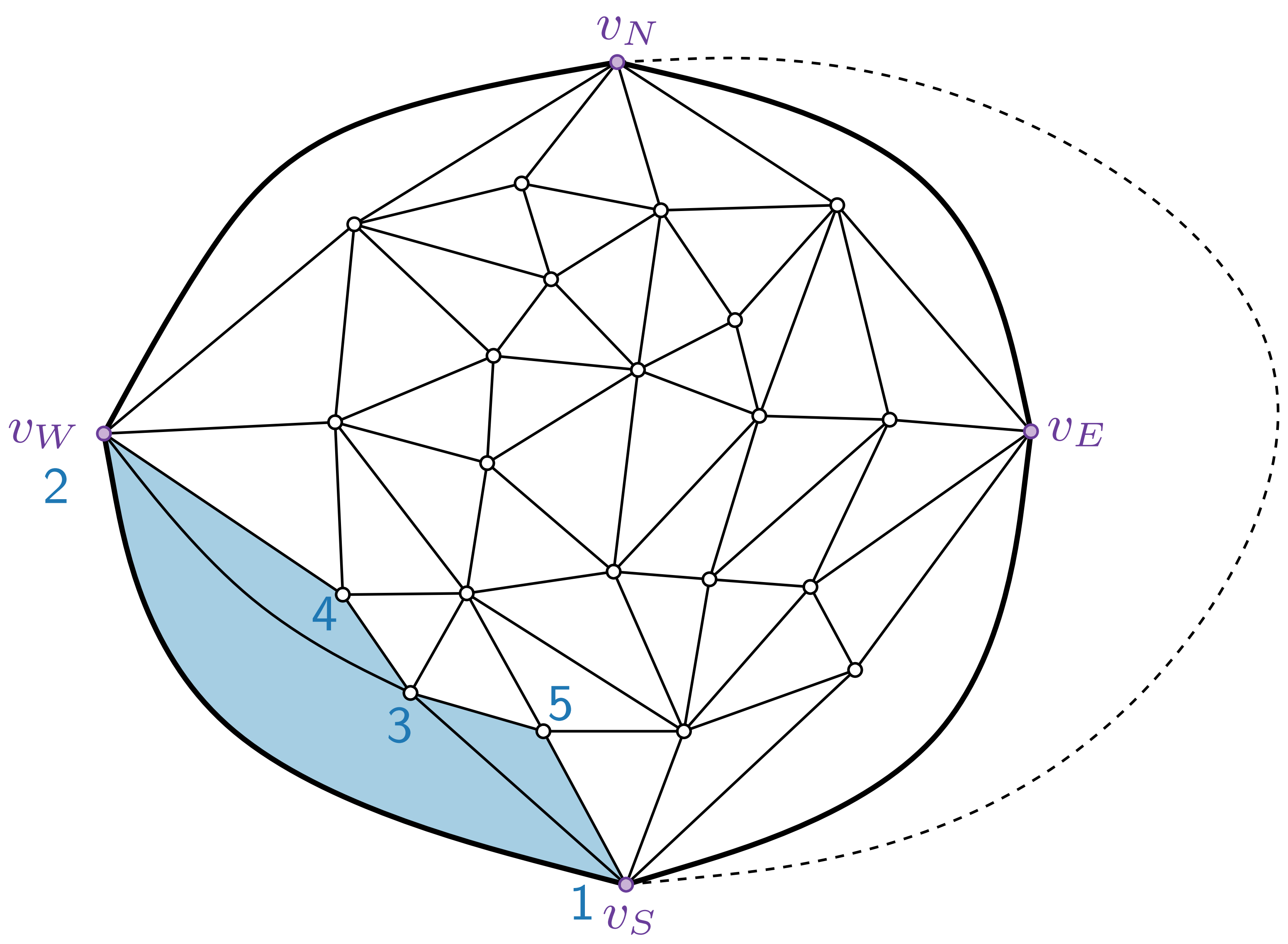
Refined Canonical Order Example



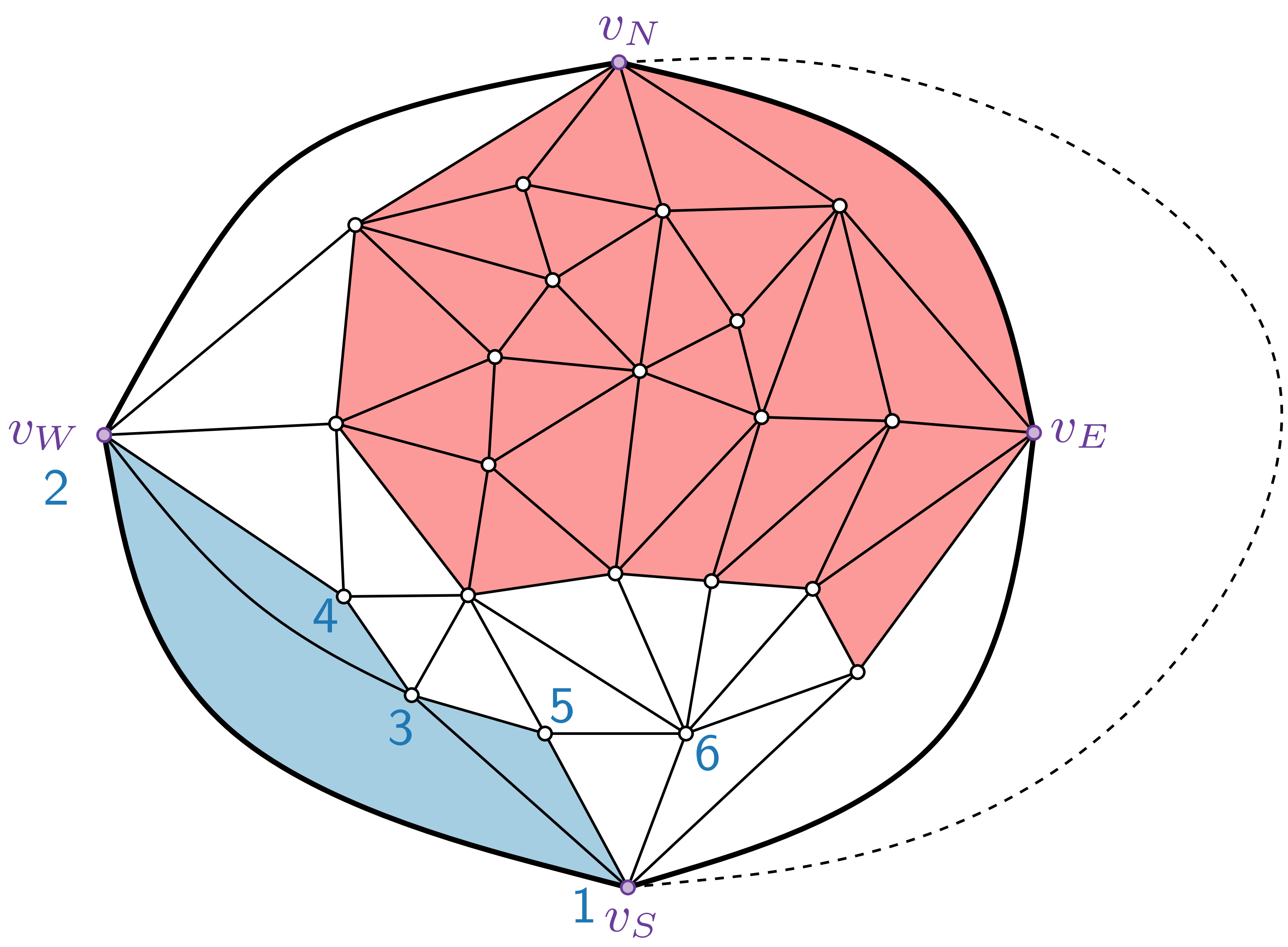
Refined Canonical Order Example



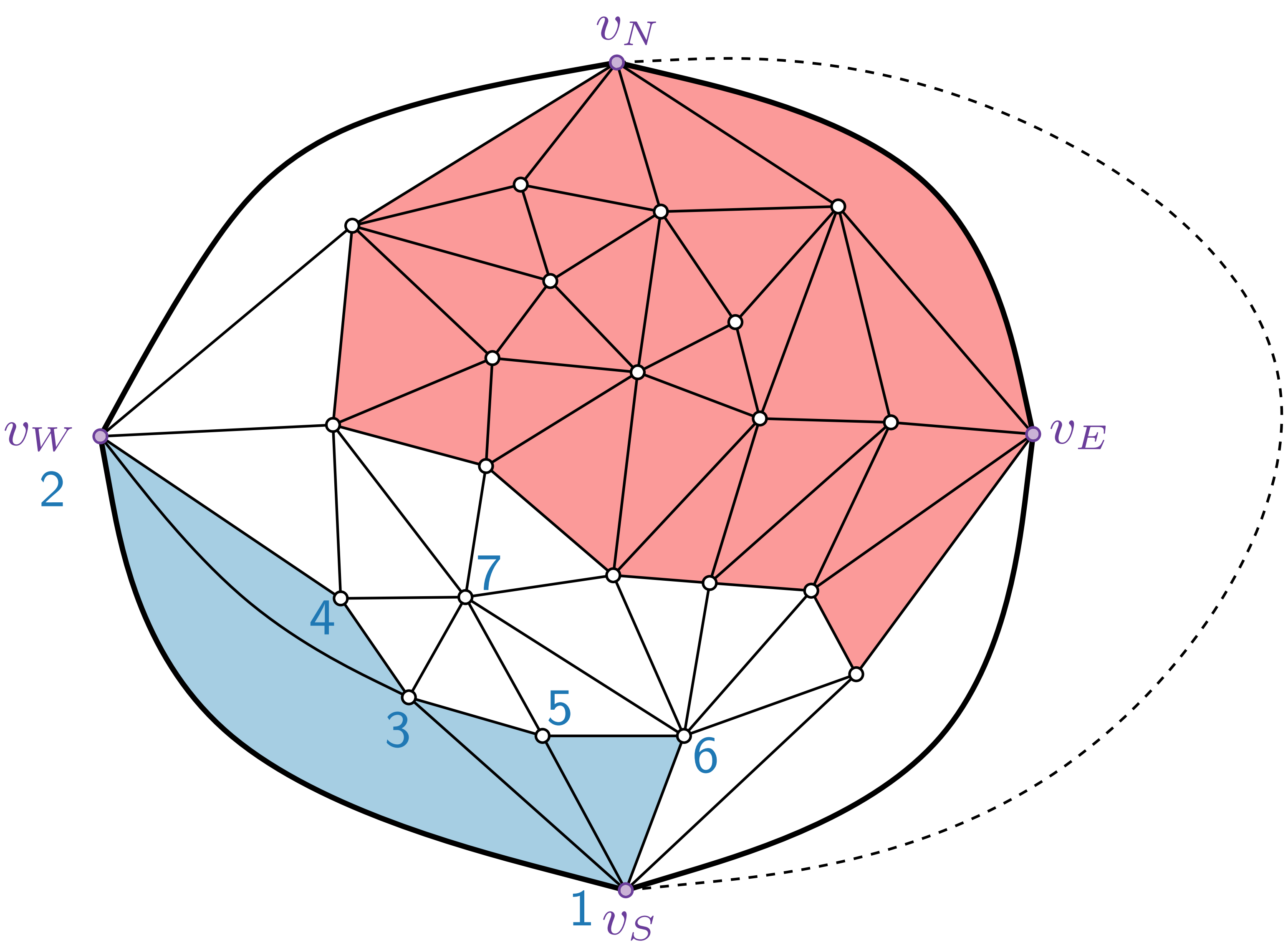
Refined Canonical Order Example



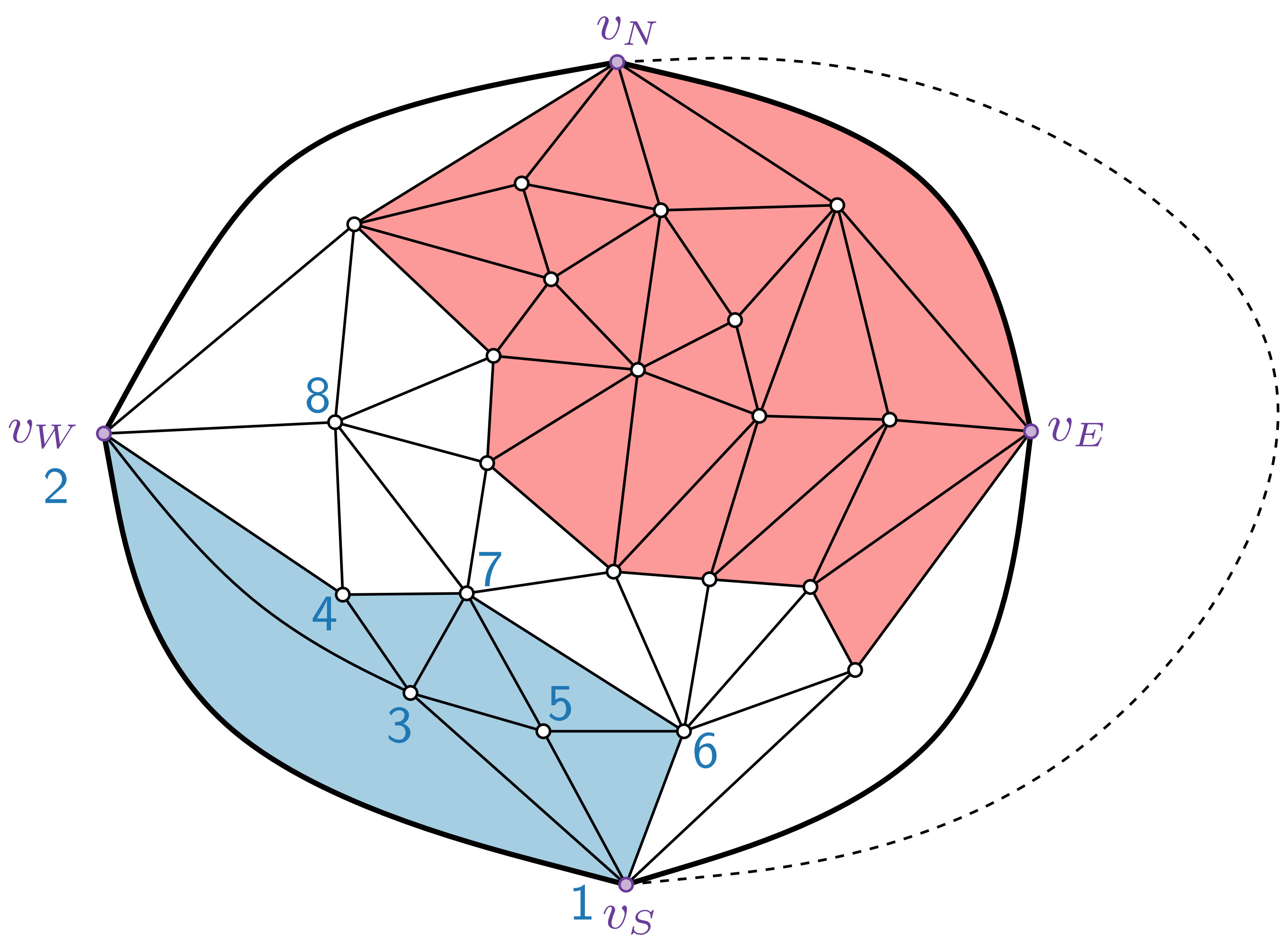
Refined Canonical Order Example



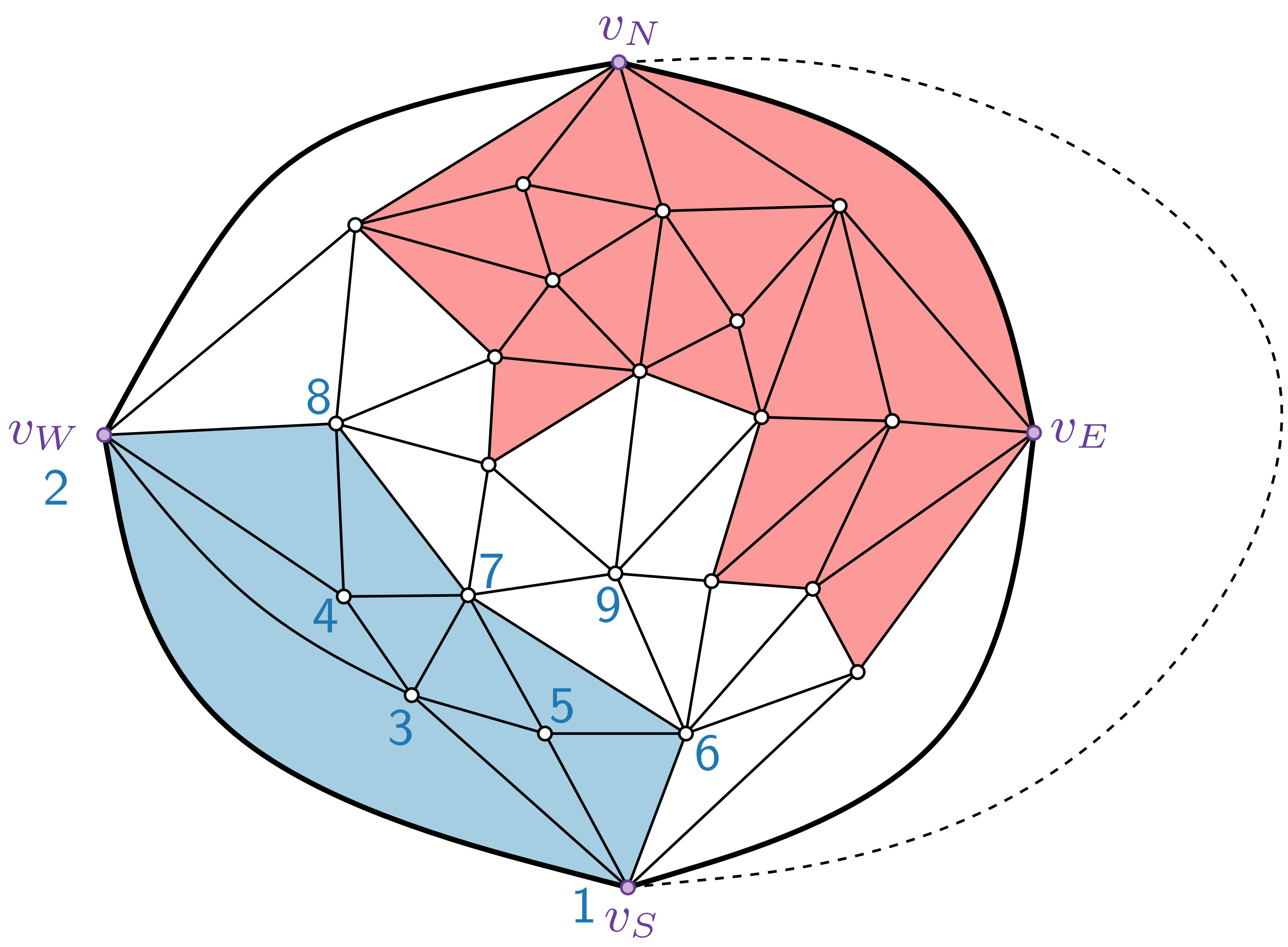
Refined Canonical Order Example



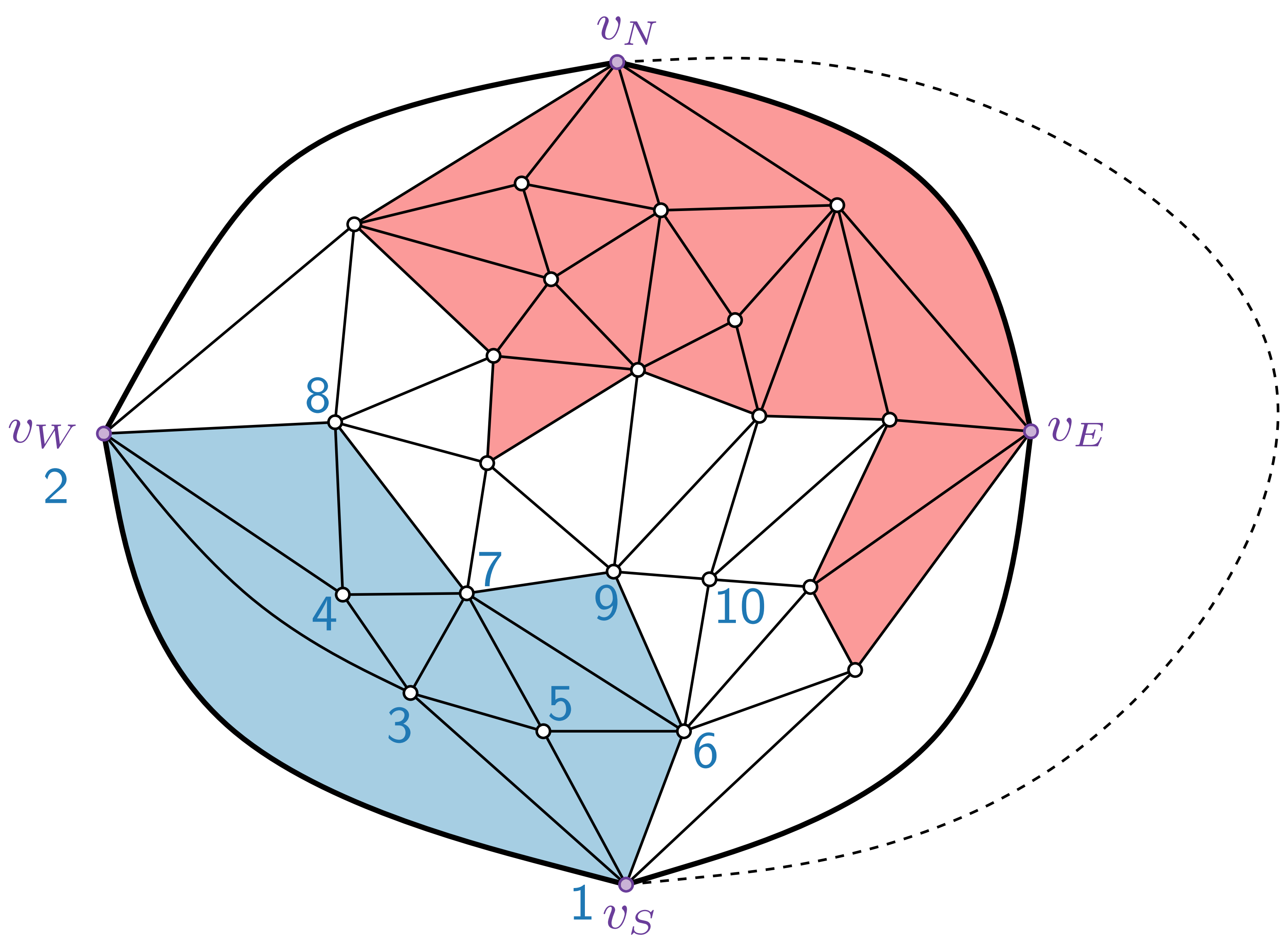
Refined Canonical Order Example



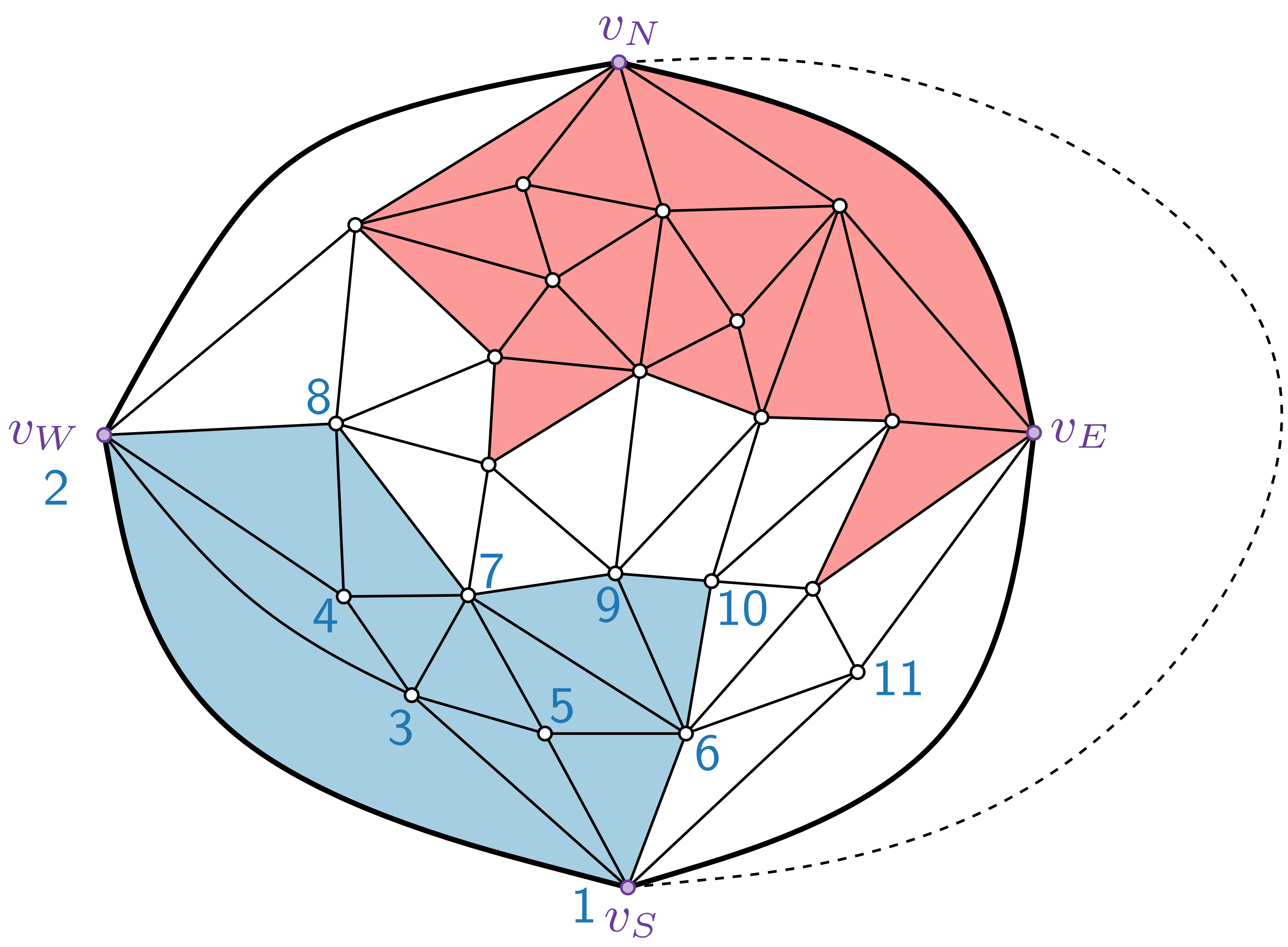
Refined Canonical Order Example



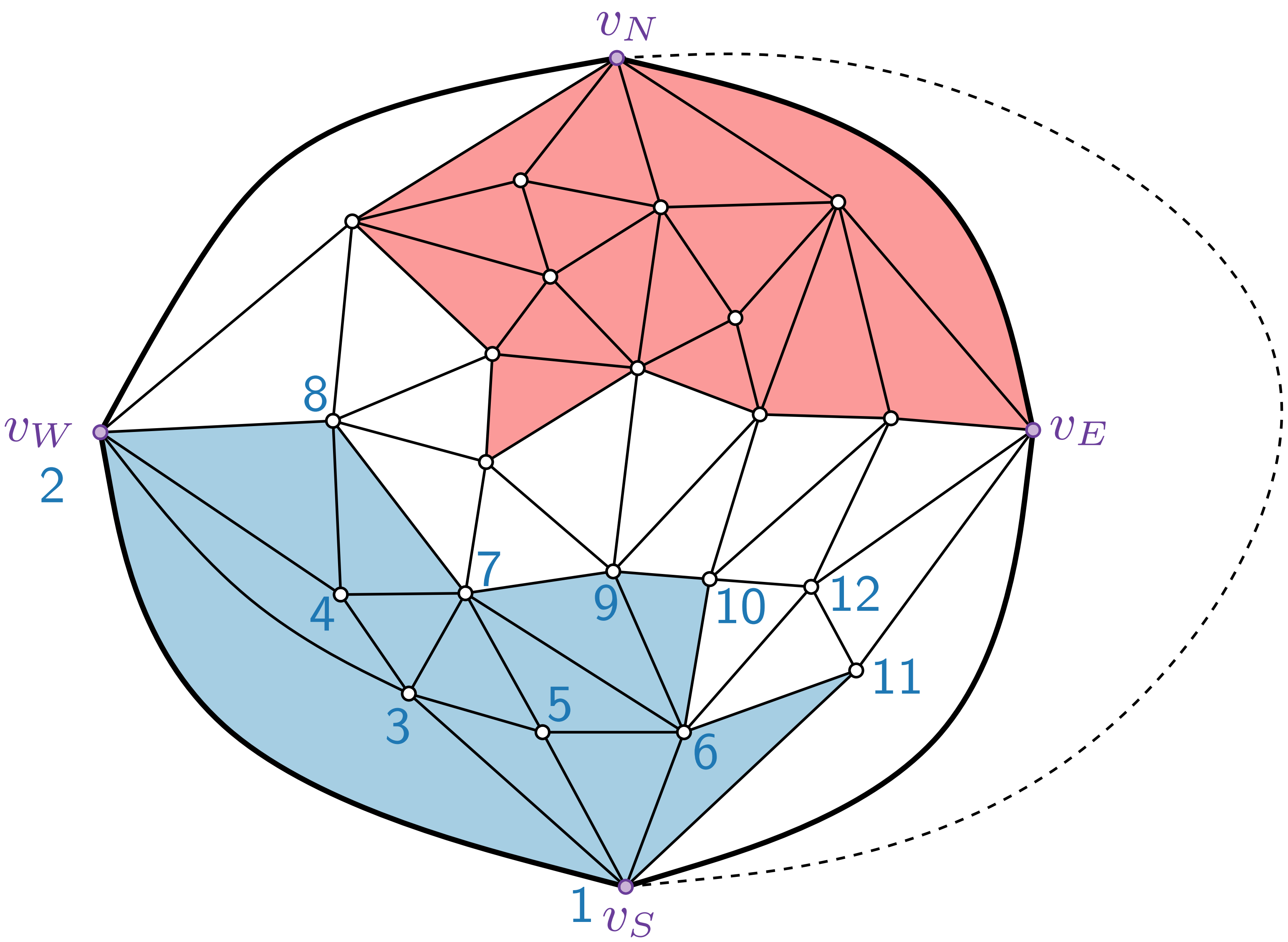
Refined Canonical Order Example



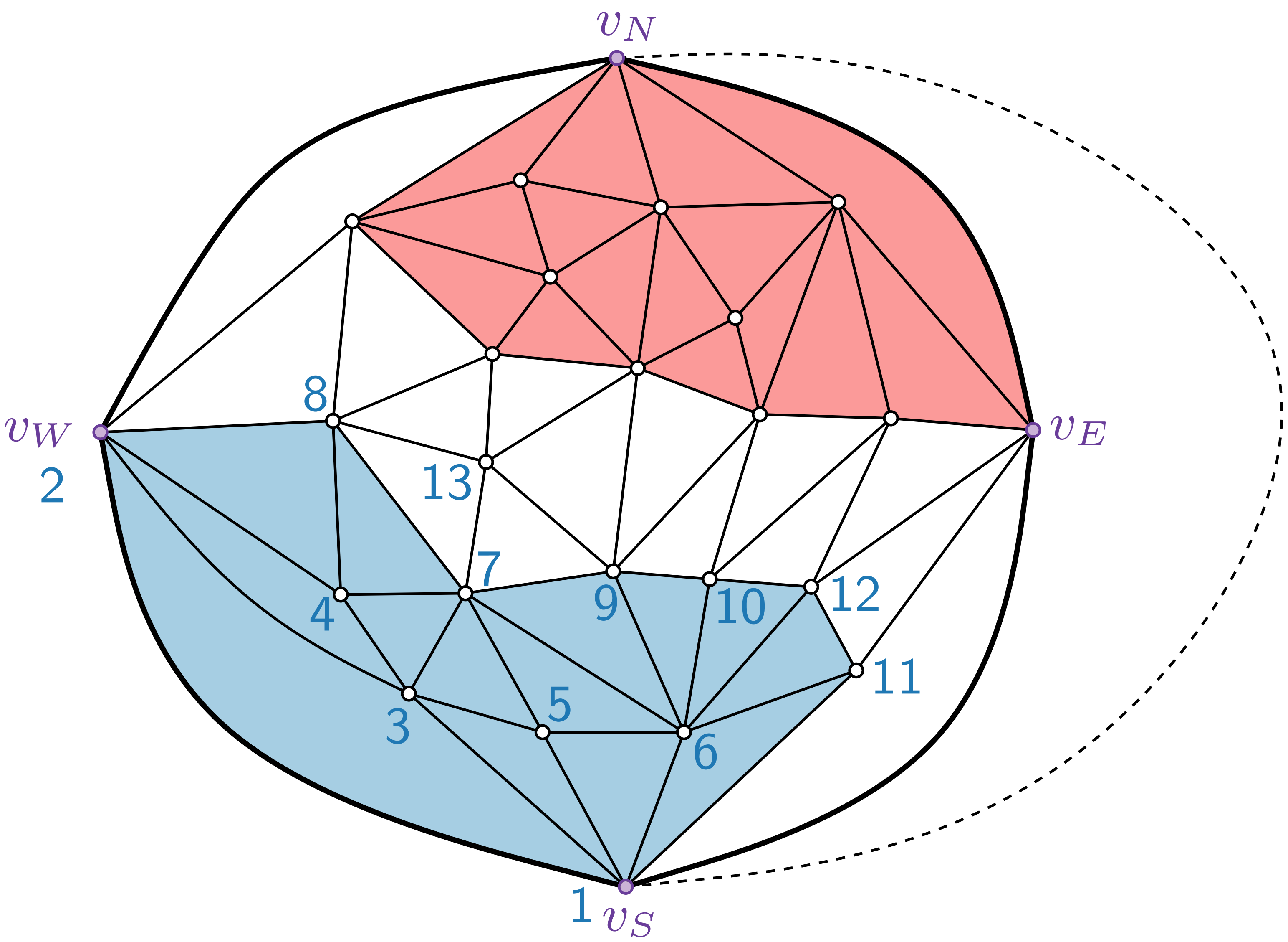
Refined Canonical Order Example



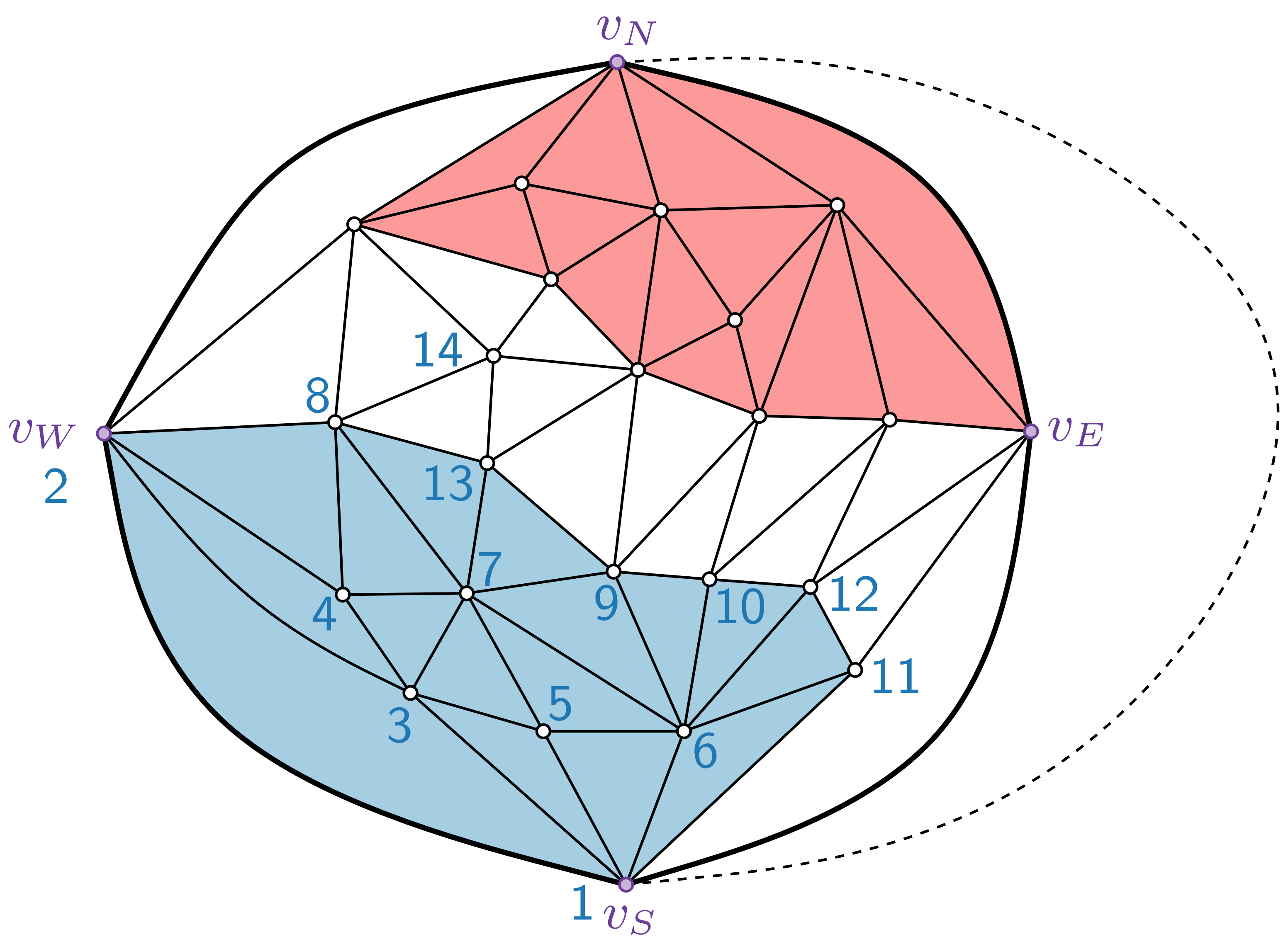
Refined Canonical Order Example



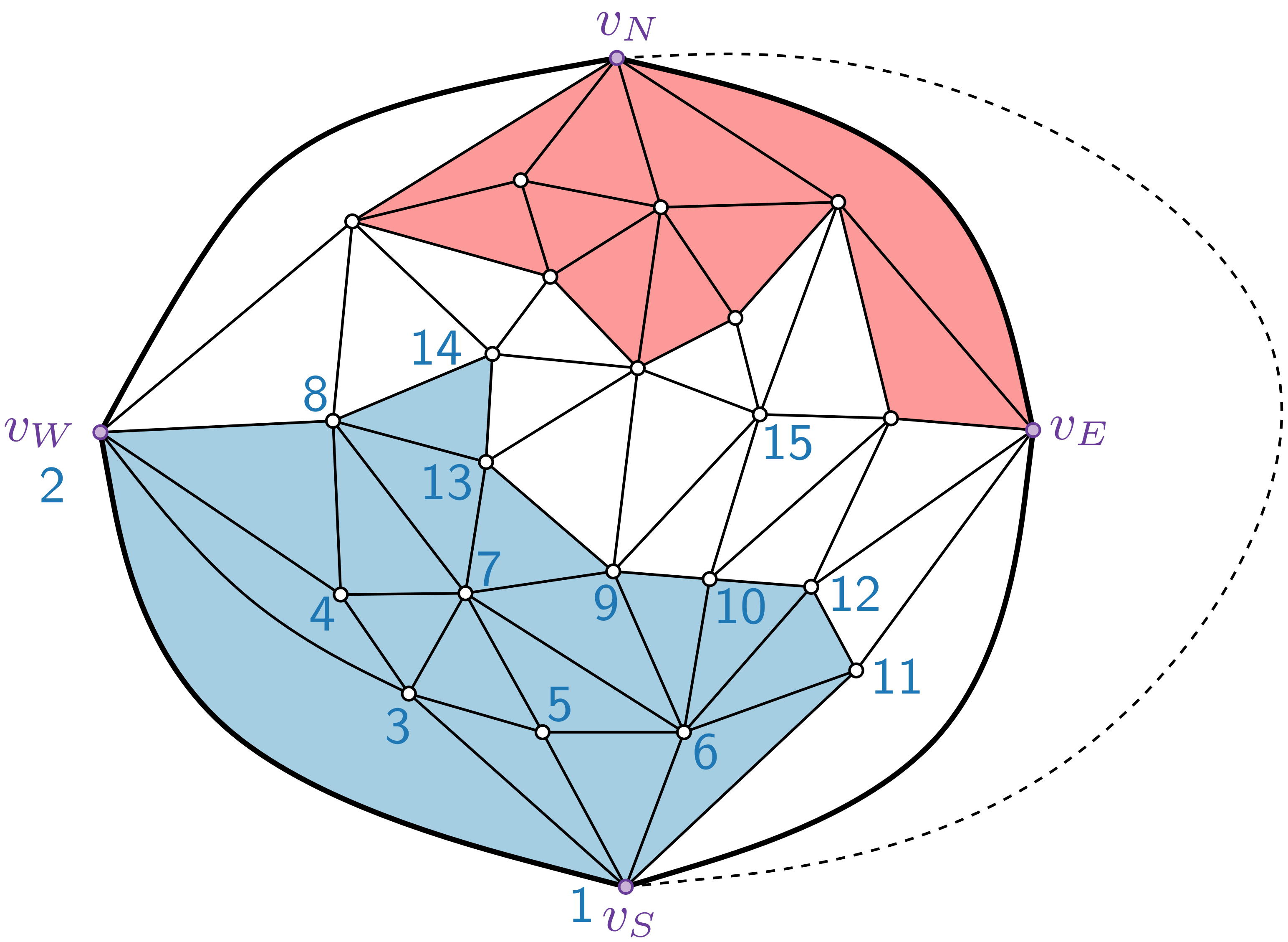
Refined Canonical Order Example



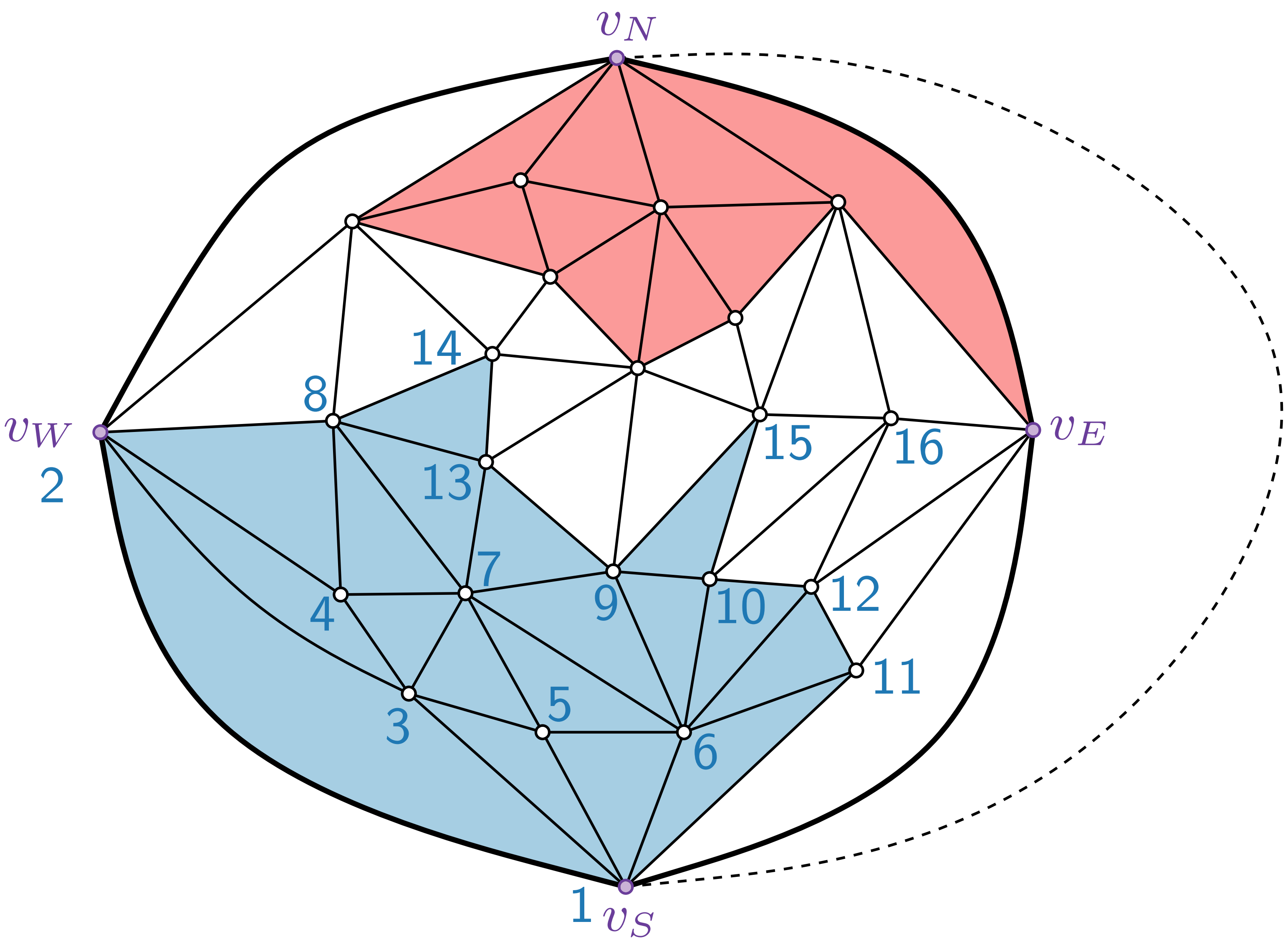
Refined Canonical Order Example



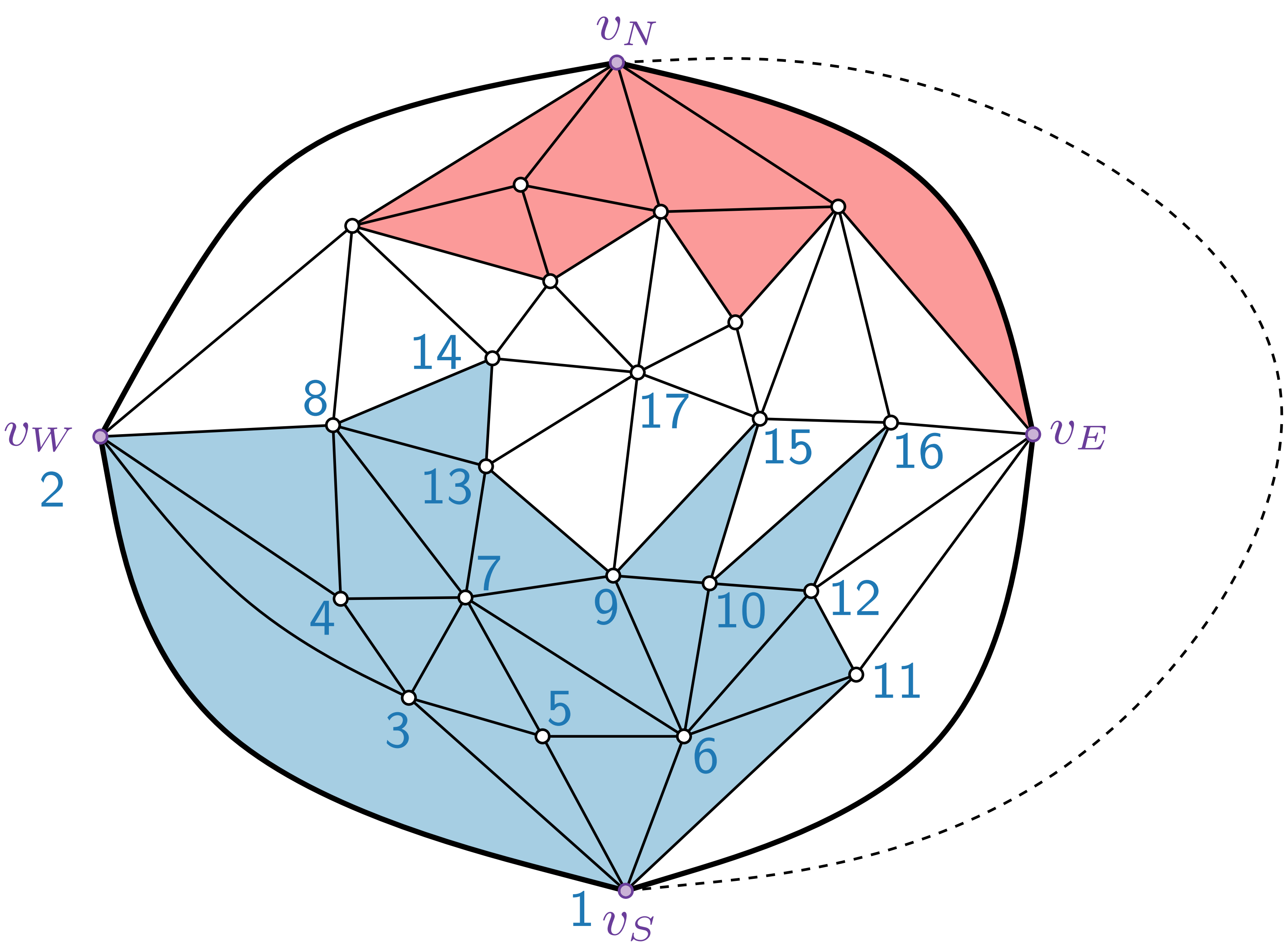
Refined Canonical Order Example



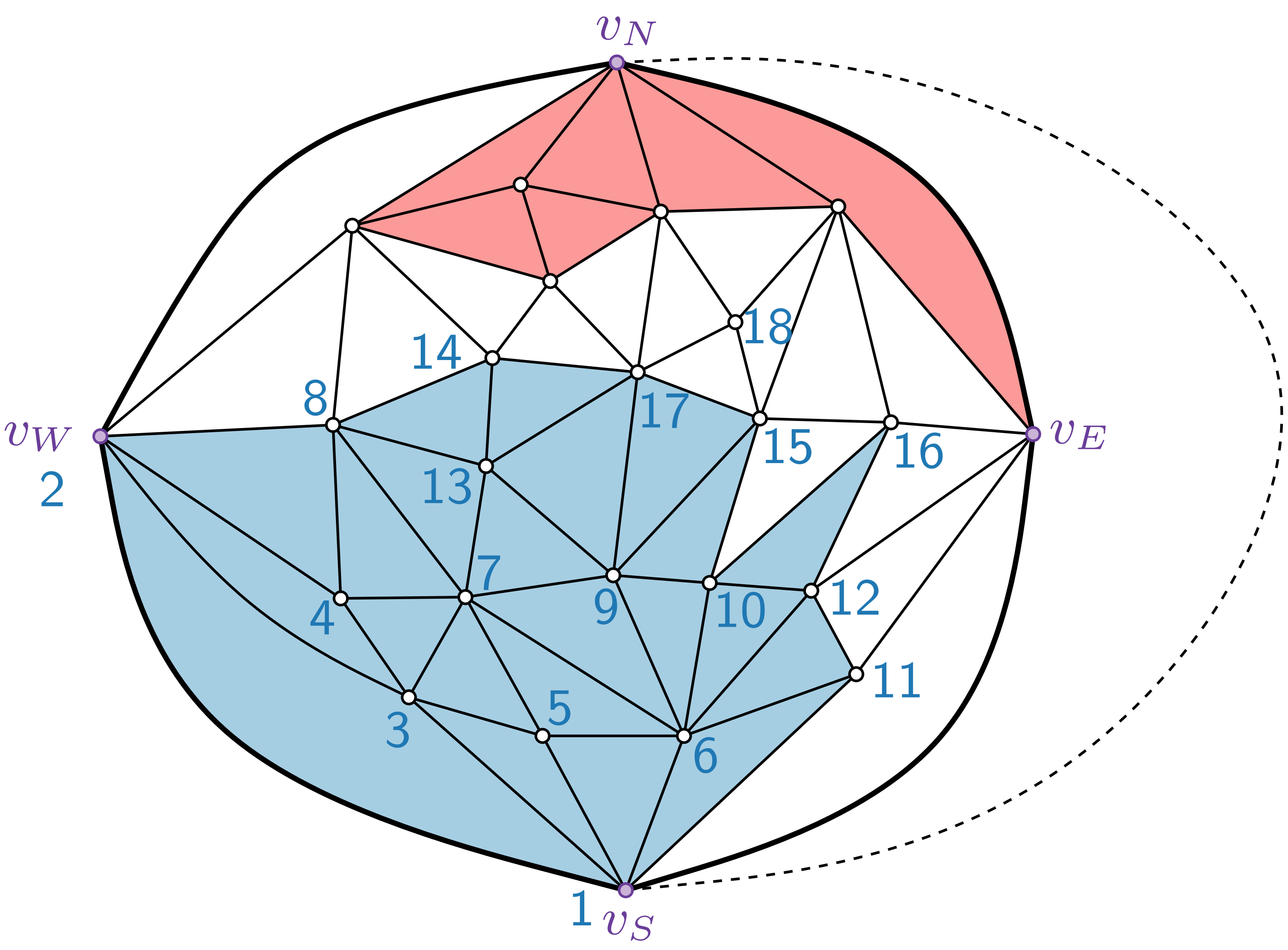
Refined Canonical Order Example



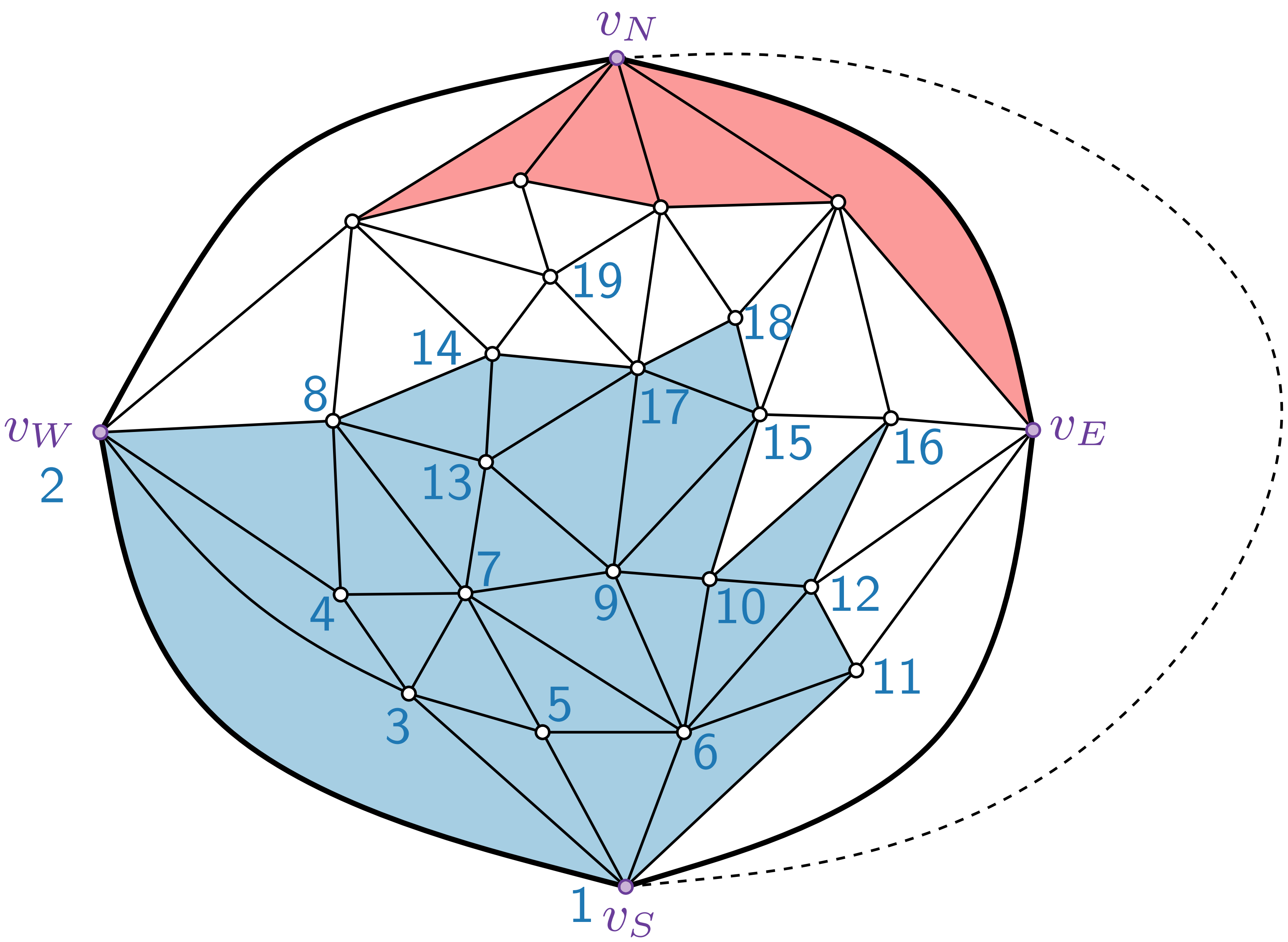
Refined Canonical Order Example



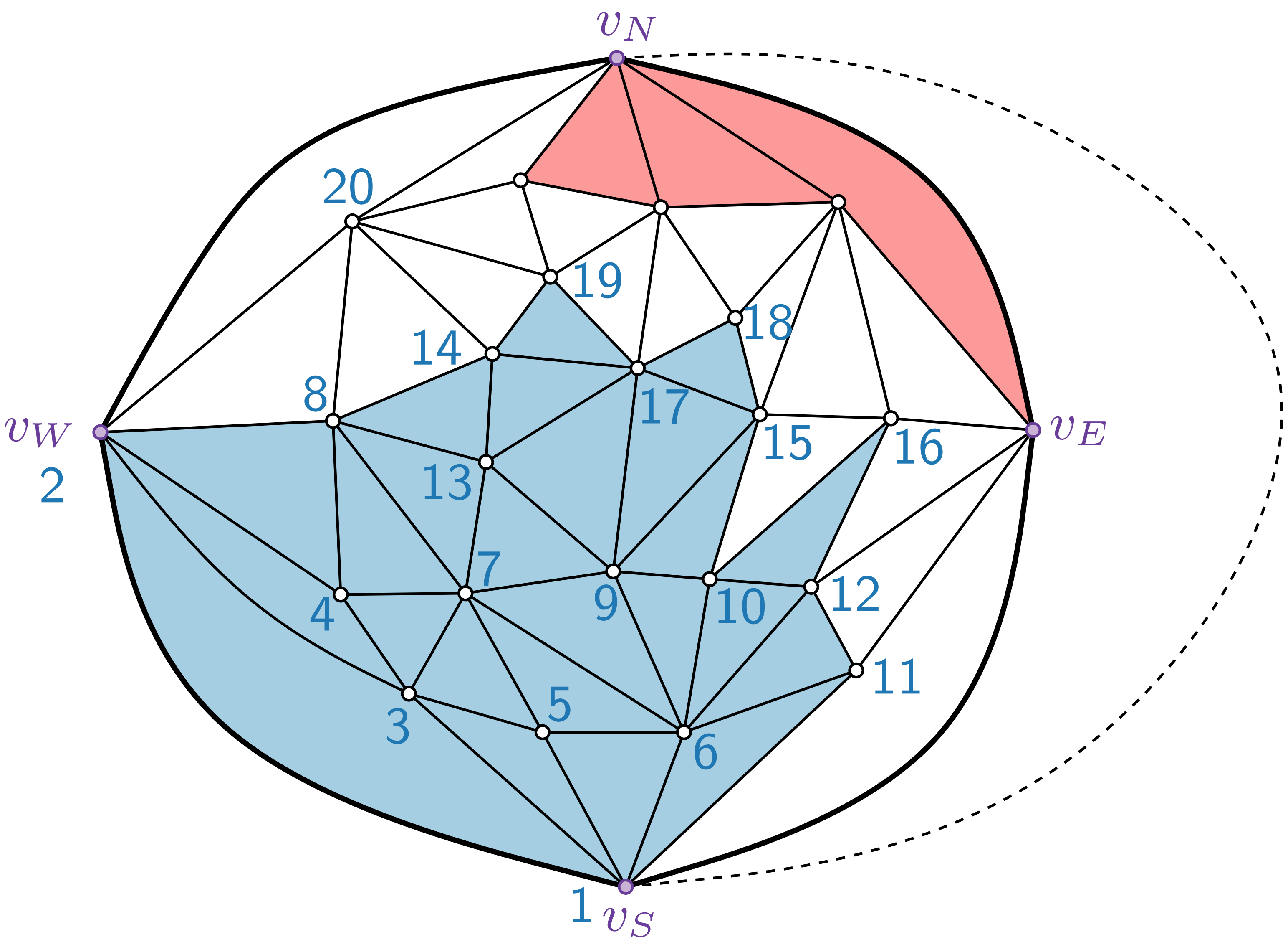
Refined Canonical Order Example



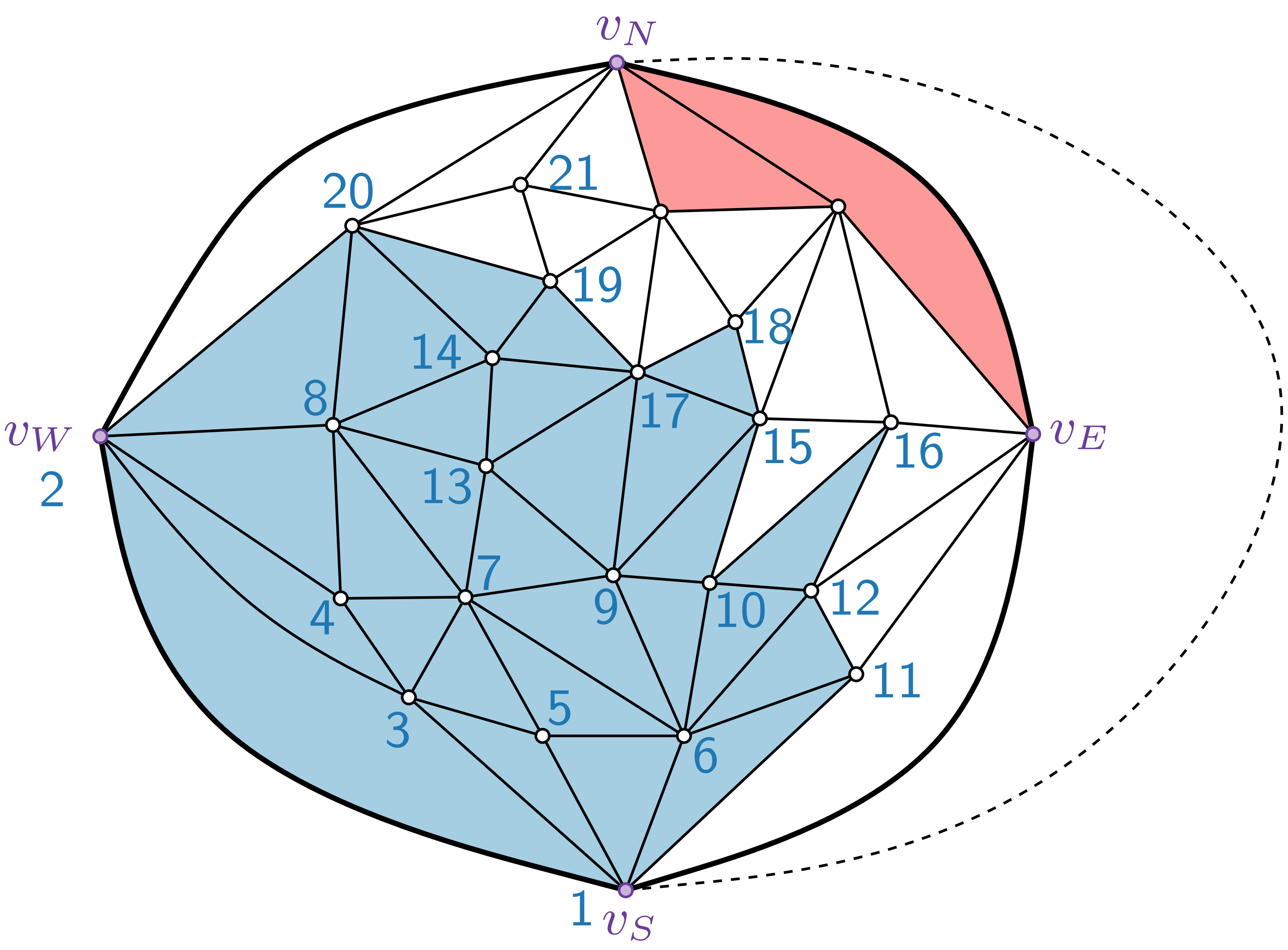
Refined Canonical Order Example



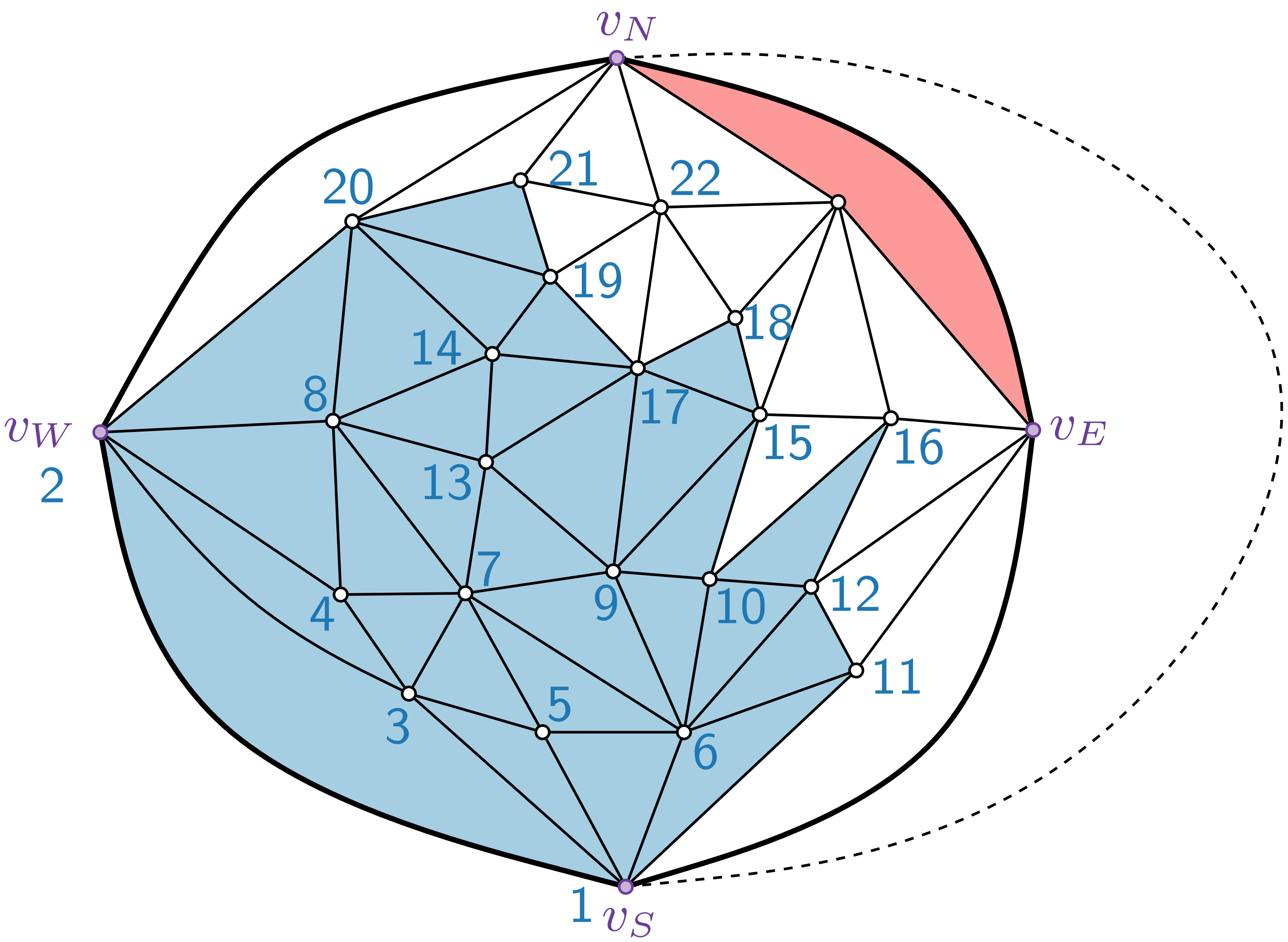
Refined Canonical Order Example



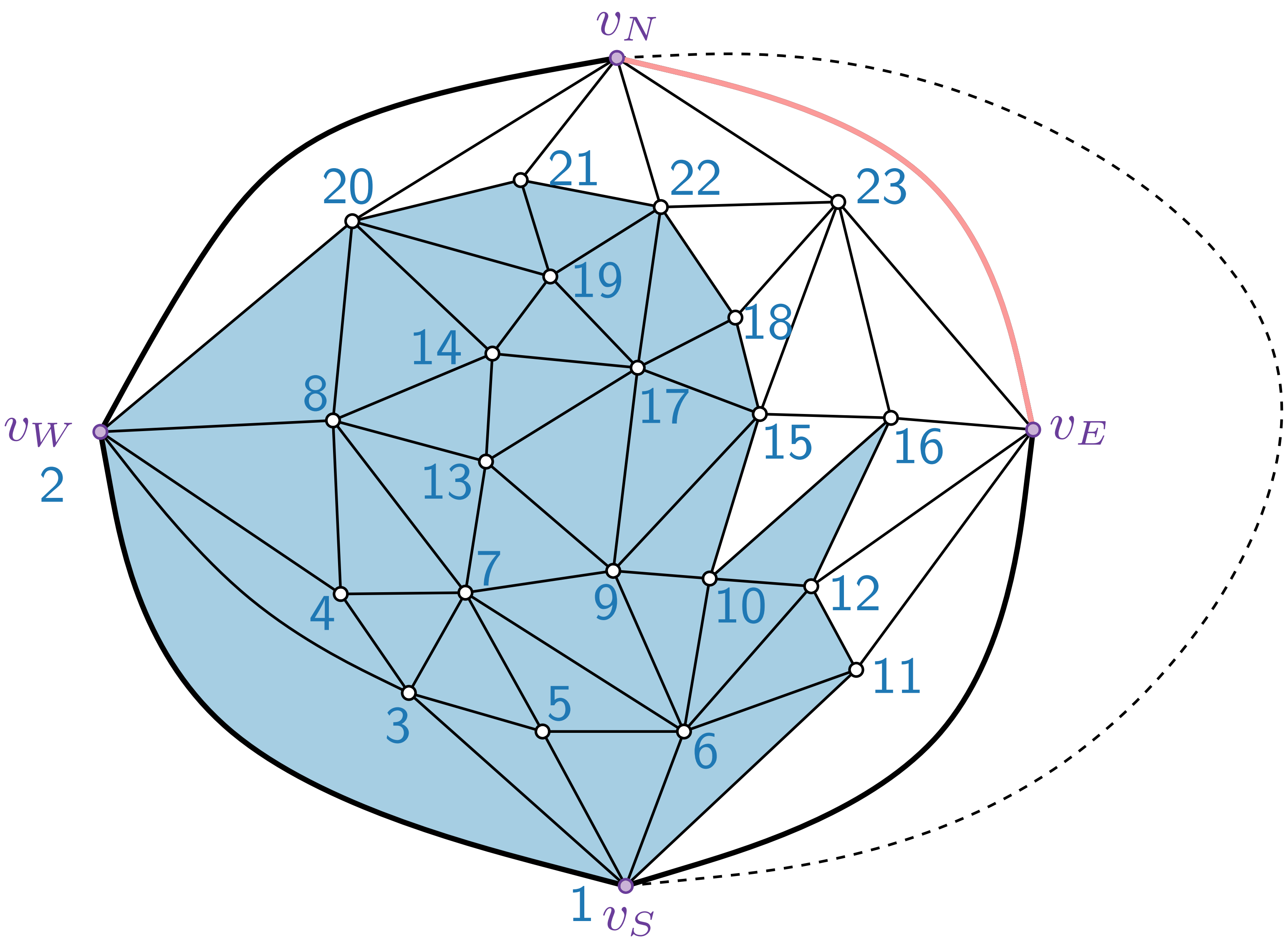
Refined Canonical Order Example



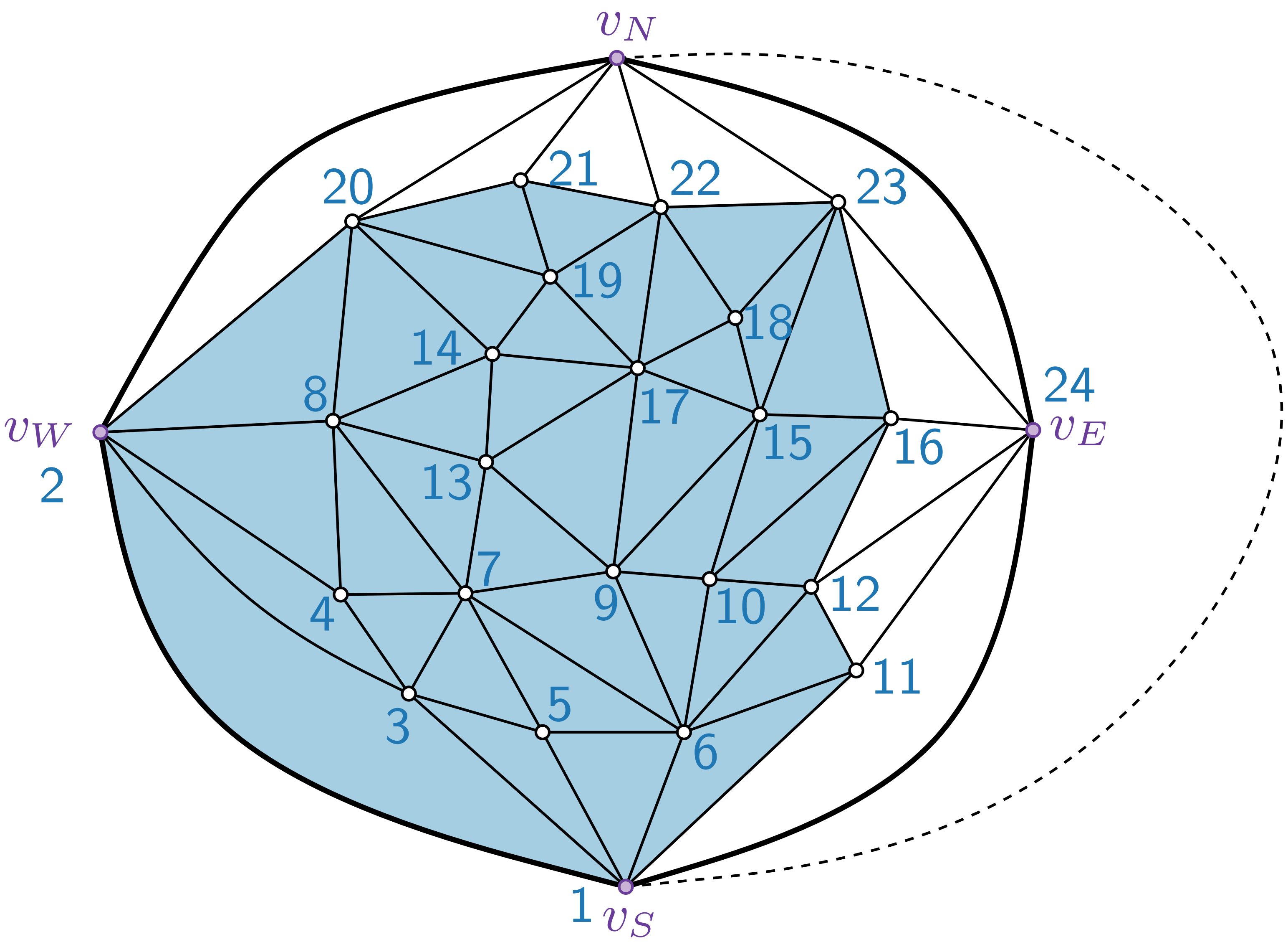
Refined Canonical Order Example



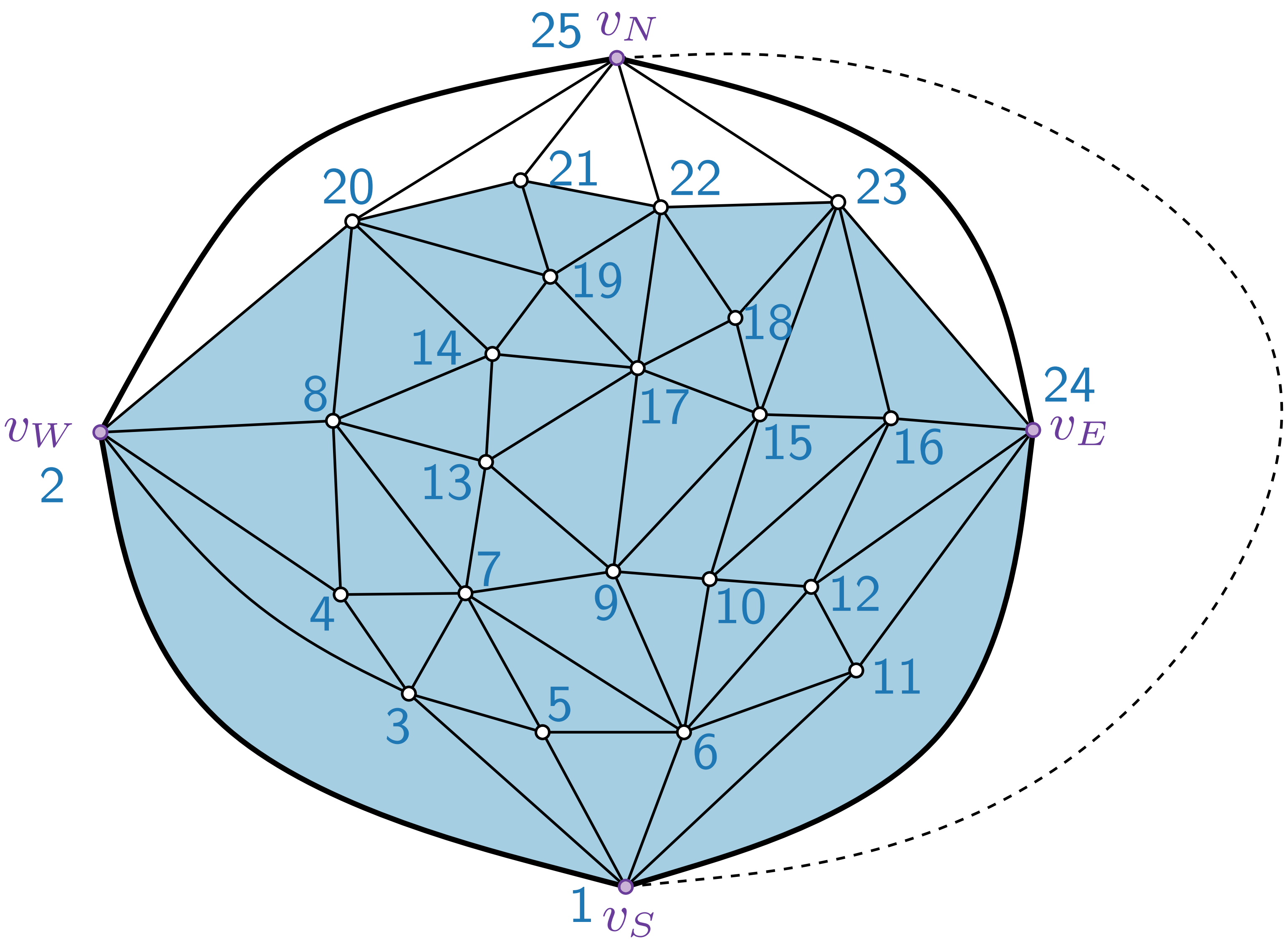
Refined Canonical Order Example



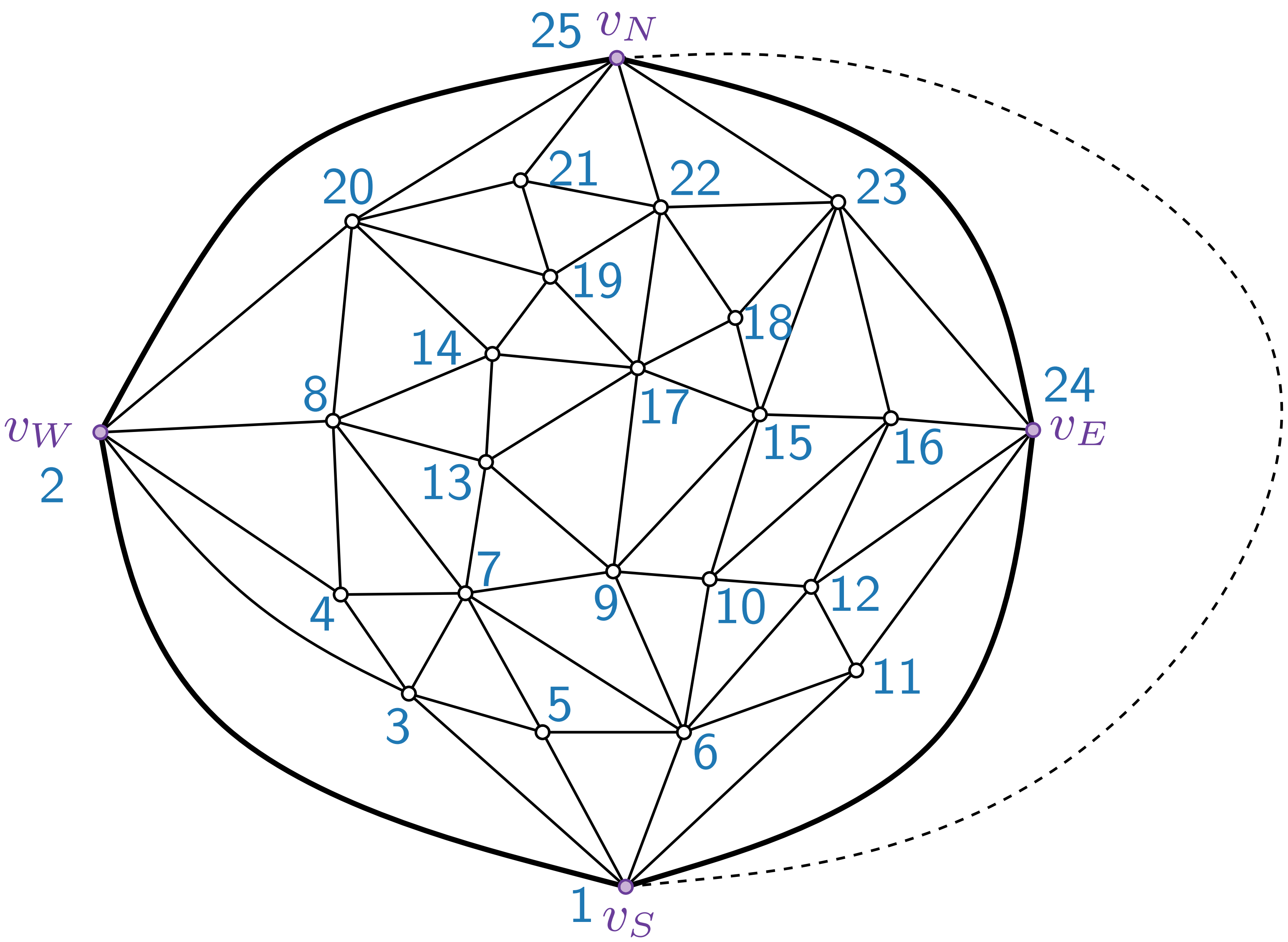
Refined Canonical Order Example



Refined Canonical Order Example

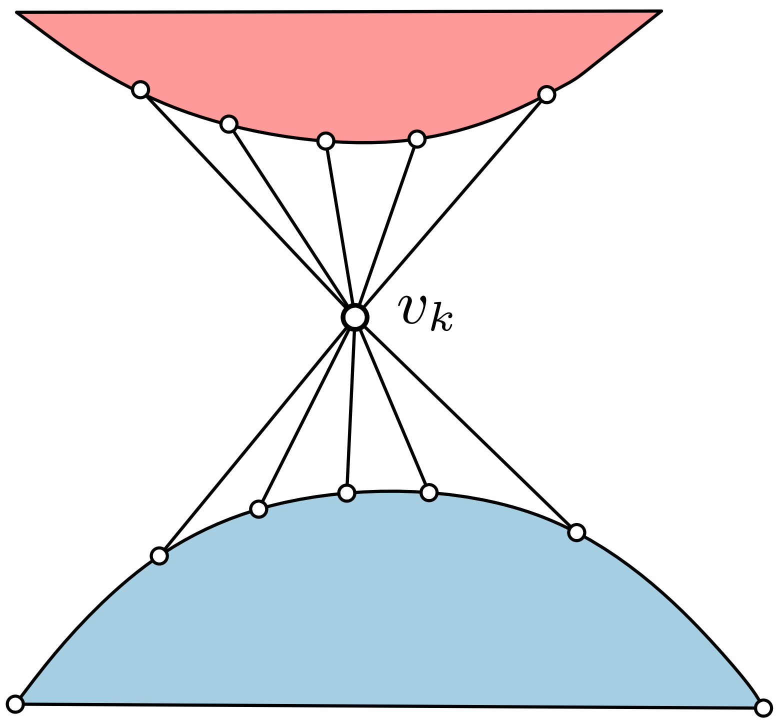


Refined Canonical Order Example



Refined Canonical Order → REL

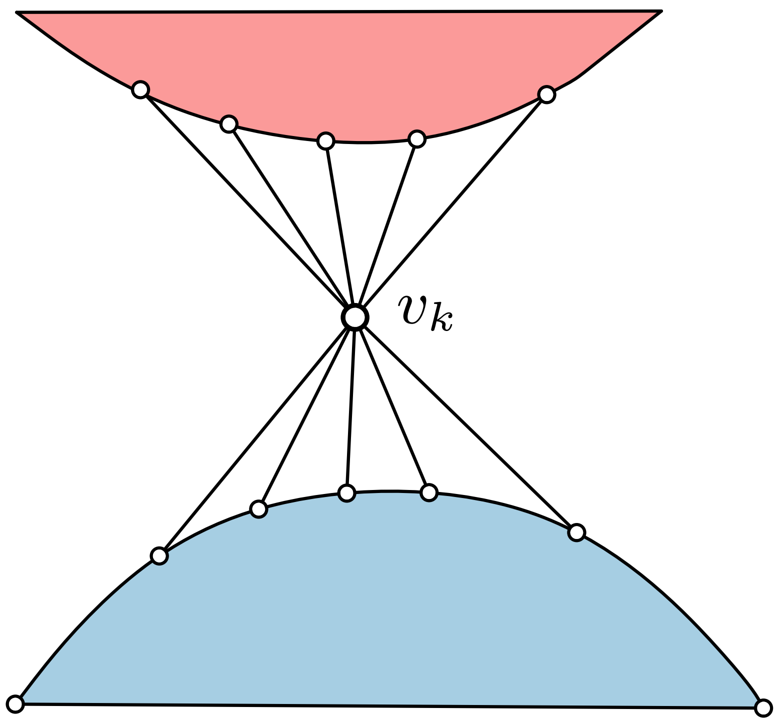
We construct a REL as follows:



Refined Canonical Order $\rightarrow$ REL

We construct a REL as follows:

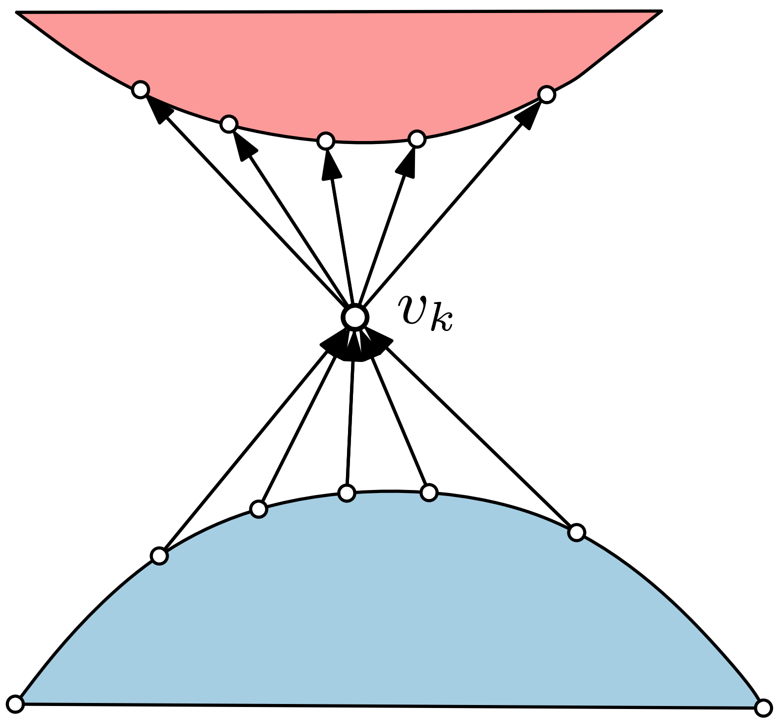
- For $i < j$, orient (v_i, v_j) from v_i to v_j;



Refined Canonical Order → REL

We construct a REL as follows:

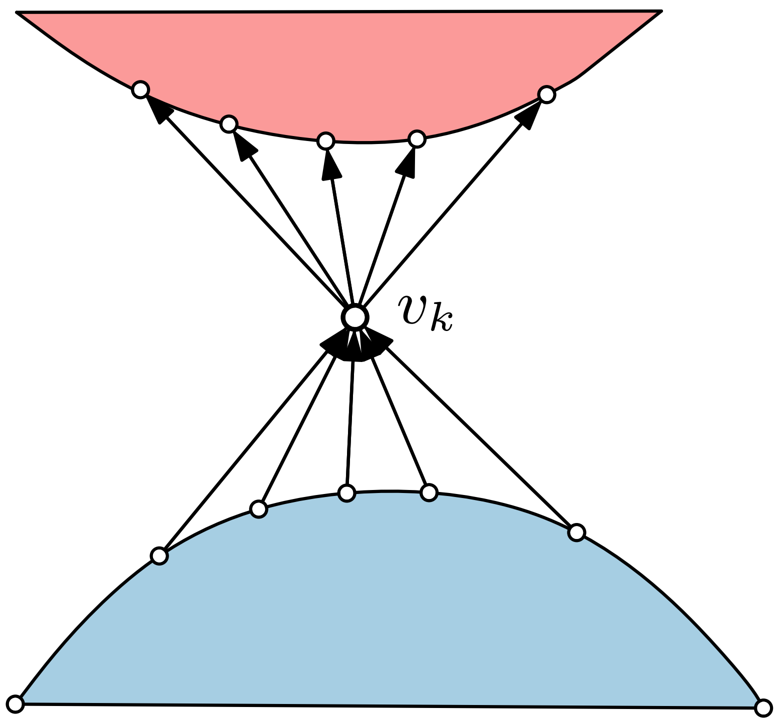
- For $i < j$, orient (v_i, v_j) from v_i to v_j;



Refined Canonical Order → REL

We construct a REL as follows:

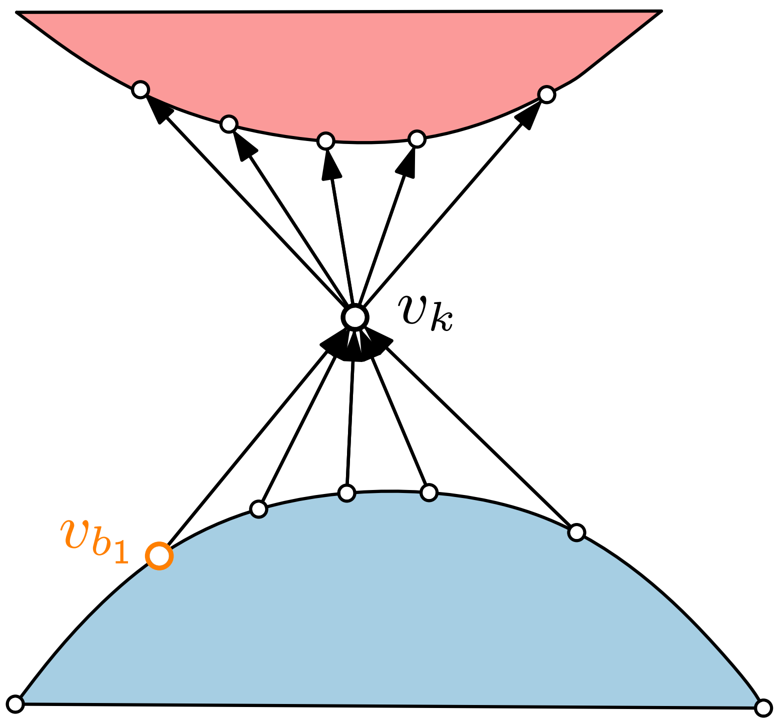
- For $i < j$, orient (v_i, v_j) from v_i to v_j;
- v_k has incoming edges from $v_{b_1}, \ldots, v_{b_l}$, we say that v_{b_1} is **left point** of v_k and v_{b_l} is **right point** of v_k.



Refined Canonical Order → REL

We construct a REL as follows:

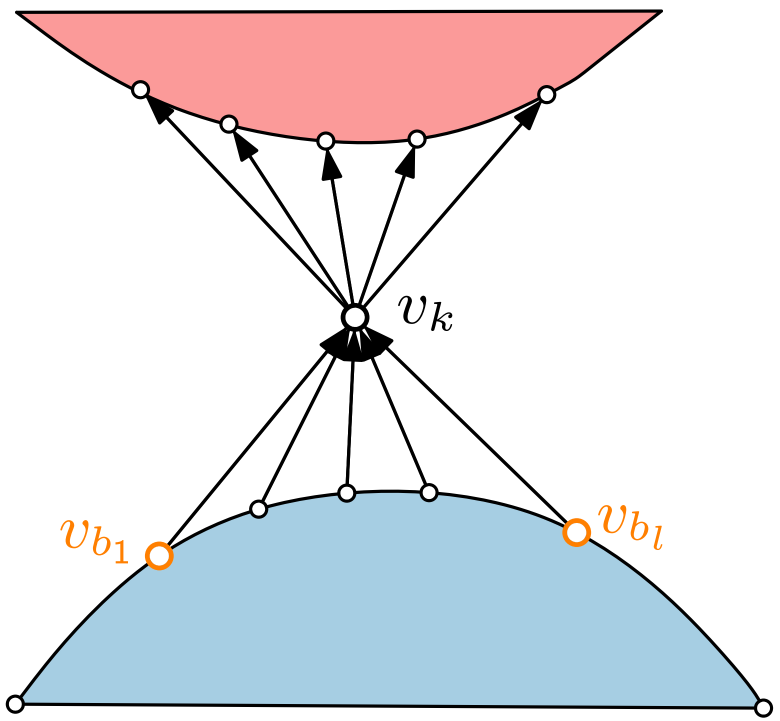
- For $i < j$, orient (v_i, v_j) from v_i to v_j;
- v_k has incoming edges from $v_{b_1}, \ldots, v_{b_l}$, we say that v_{b_1} is **left point** of v_k and v_{b_l} is **right point** of v_k.



Refined Canonical Order $\rightarrow$ REL

We construct a REL as follows:

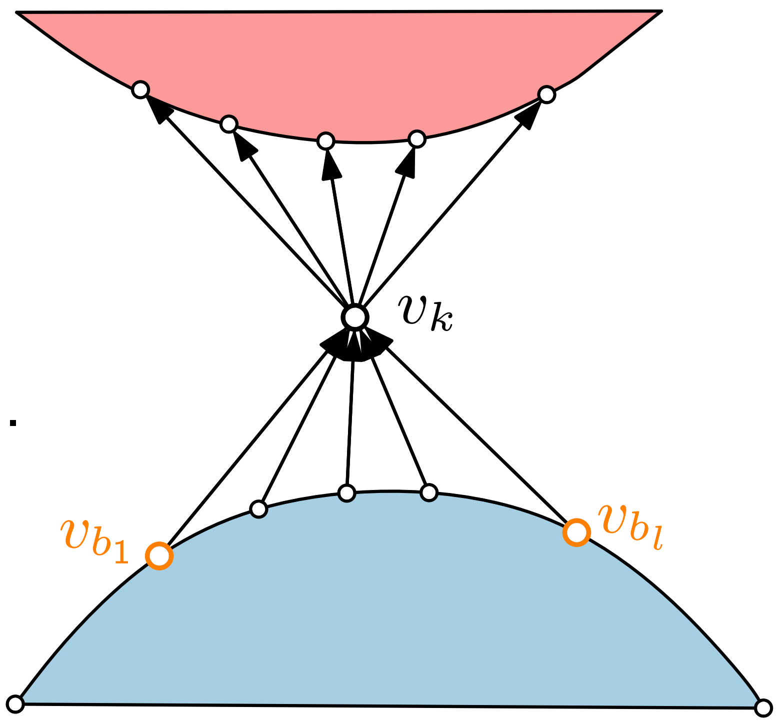
- For $i < j$, orient (v_i, v_j) from v_i to v_j;
- v_k has incoming edges from $v_{b_1}, \ldots, v_{b_l}$, we say that v_{b_1} is **left point** of v_k and v_{b_l} is **right point** of v_k.



Refined Canonical Order $\rightarrow$ REL

We construct a REL as follows:

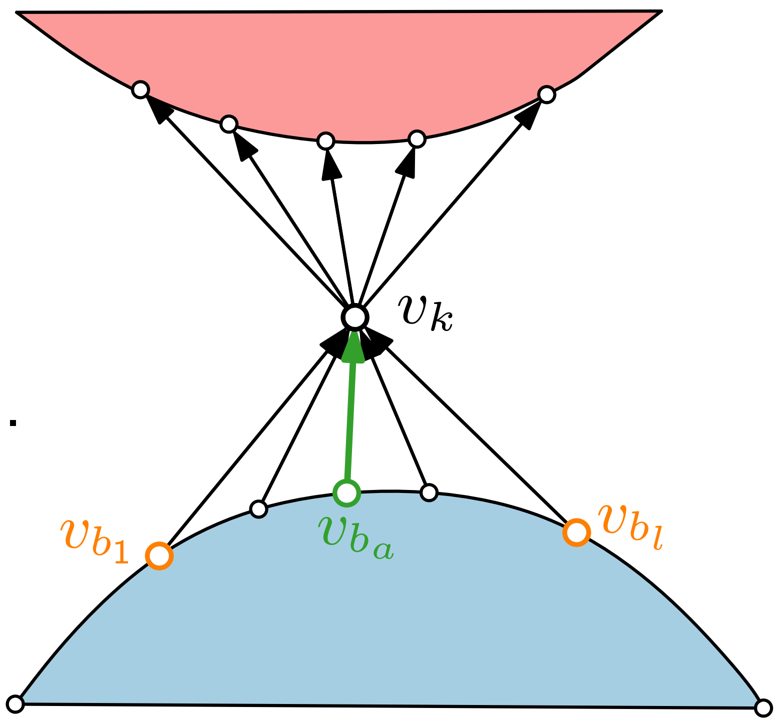
- For $i < j$, orient (v_i, v_j) from v_i to v_j;
- v_k has incoming edges from $v_{b_1}, \ldots, v_{b_l}$, we say that v_{b_1} is **left point** of v_k and v_{b_l} is **right point** of v_k.
- **Base edge** of v_k is (v_{b_a}, v_k), where $b_a \in \{b_1, \ldots, b_l\}$ is minimal.



Refined Canonical Order → REL

We construct a REL as follows:

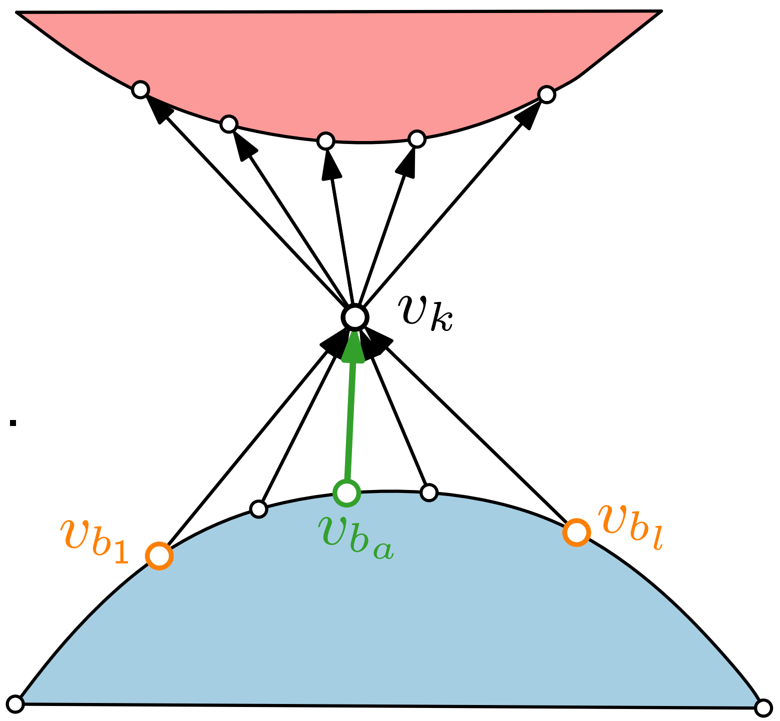
- For $i < j$, orient (v_i, v_j) from v_i to v_j;
- v_k has incoming edges from $v_{b_1}, \ldots, v_{b_l}$, we say that v_{b_1} is **left point** of v_k and v_{b_l} is **right point** of v_k.
- **Base edge** of v_k is (v_{b_a}, v_k), where $b_a \in \{b_1, \ldots, b_l\}$ is minimal.



Refined Canonical Order → REL

We construct a REL as follows:

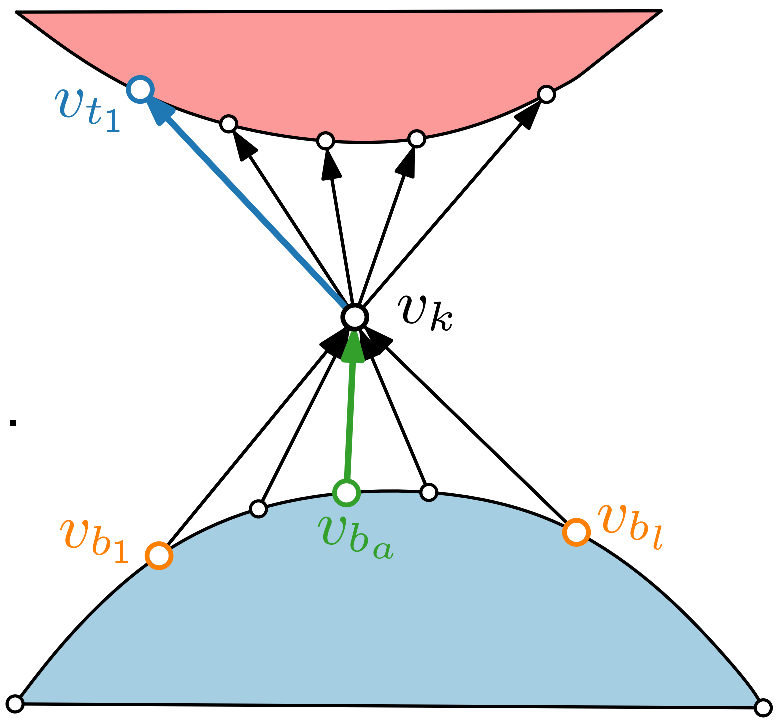
- For $i < j$, orient (v_i, v_j) from v_i to v_j;
- v_k has incoming edges from $v_{b_1}, \ldots, v_{b_l}$, we say that v_{b_1} is **left point** of v_k and v_{b_l} is **right point** of v_k.
- **Base edge** of v_k is (v_{b_a}, v_k), where $b_a \in \{b_1, \ldots, b_l\}$ is minimal.
- If $v_{t_1}, \ldots, v_{t_o}$ are higher numbered neighbors of v_k, we call (v_k, v_{t_1}) **left edge** and (v_k, v_{t_o}) **right edge**.



Refined Canonical Order → REL

We construct a REL as follows:

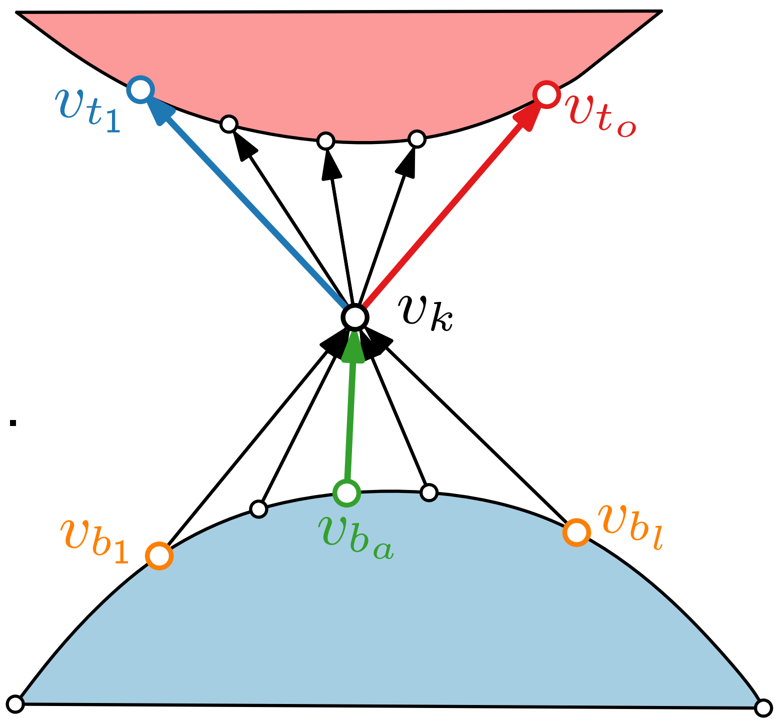
- For $i < j$, orient (v_i, v_j) from v_i to v_j;
- v_k has incoming edges from $v_{b_1}, \dots, v_{b_l}$, we say that v_{b_1} is **left point** of v_k and v_{b_l} is **right point** of v_k.
- **Base edge** of v_k is (v_{b_a}, v_k), where $b_a \in \{b_1, \dots, b_l\}$ is minimal.
- If $v_{t_1}, \dots, v_{t_o}$ are higher numbered neighbors of v_k, we call (v_k, v_{t_1}) **left edge** and (v_k, v_{t_o}) **right edge**.



Refined Canonical Order → REL

We construct a REL as follows:

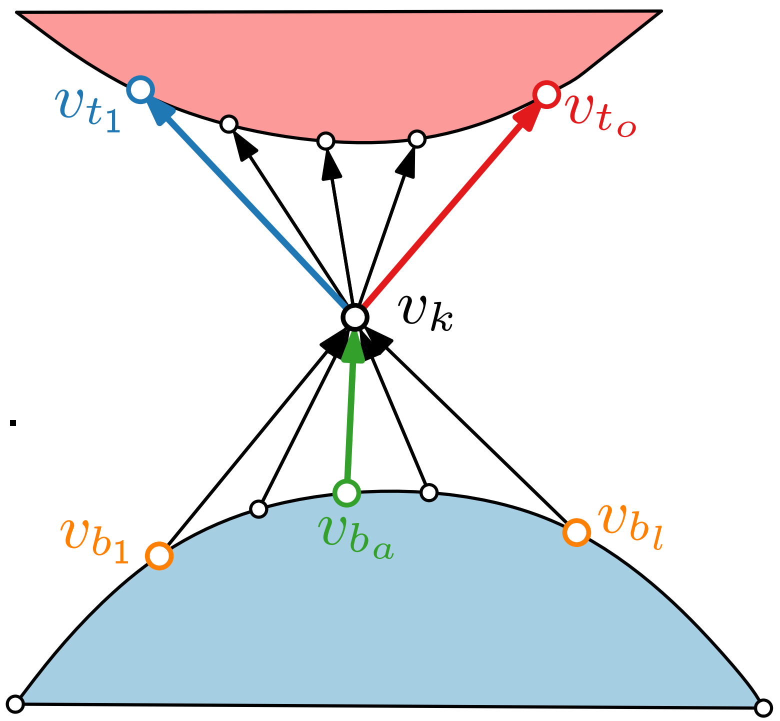
- For $i < j$, orient (v_i, v_j) from v_i to v_j;
- v_k has incoming edges from $v_{b_1}, \ldots, v_{b_l}$, we say that v_{b_1} is **left point** of v_k and v_{b_l} is **right point** of v_k.
- **Base edge** of v_k is (v_{b_a}, v_k), where $b_a \in \{b_1, \ldots, b_l\}$ is minimal.
- If $v_{t_1}, \ldots, v_{t_o}$ are higher numbered neighbors of v_k, we call (v_k, v_{t_1}) **left edge** and (v_k, v_{t_o}) **right edge**.



Refined Canonical Order $\rightarrow$ REL

We construct a REL as follows:

- For $i < j$, orient (v_i, v_j) from v_i to v_j;
- v_k has incoming edges from $v_{b_1}, \ldots, v_{b_l}$, we say that v_{b_1} is **left point** of v_k and v_{b_l} is **right point** of v_k.
- **Base edge** of v_k is (v_{b_a}, v_k), where $b_a \in \{b_1, \ldots, b_l\}$ is minimal.
- If $v_{t_1}, \ldots, v_{t_o}$ are higher numbered neighbors of v_k, we call (v_k, v_{t_1}) **left edge** and (v_k, v_{t_o}) **right edge**.

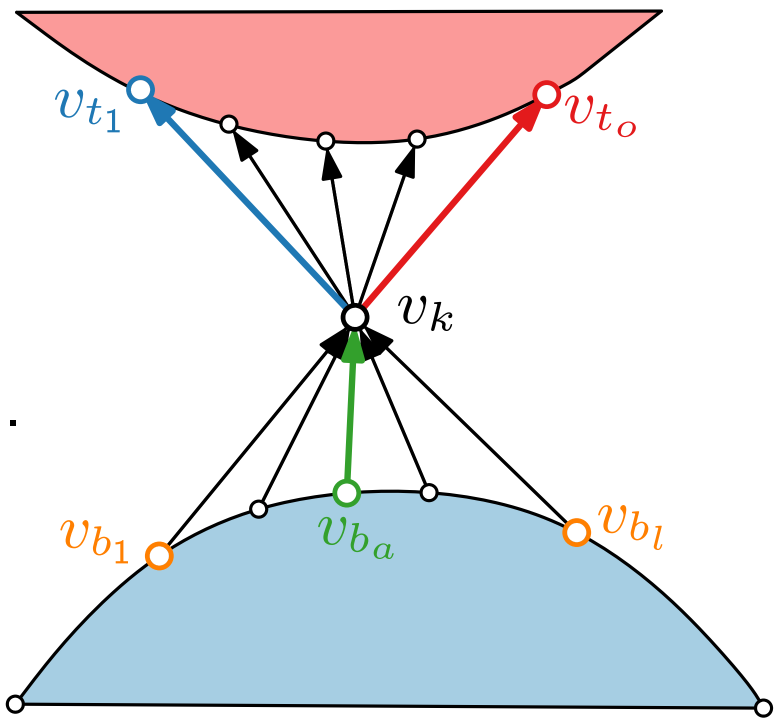


Lemma 1.
A left edge or right edge cannot be a base edge.

Refined Canonical Order $\to$ REL

We construct a REL as follows:

- For $i < j$, orient (v_i, v_j) from v_i to v_j;
- v_k has incoming edges from $v_{b_1}, \ldots, v_{b_l}$, we say that v_{b_1} is **left point** of v_k and v_{b_l} is **right point** of v_k.
- **Base edge** of v_k is (v_{b_a}, v_k), where $b_a \in \{b_1, \ldots, b_l\}$ is minimal.
- If $v_{t_1}, \ldots, v_{t_o}$ are higher numbered neighbors of v_k, we call (v_k, v_{t_1}) **left edge** and (v_k, v_{t_o}) **right edge**.



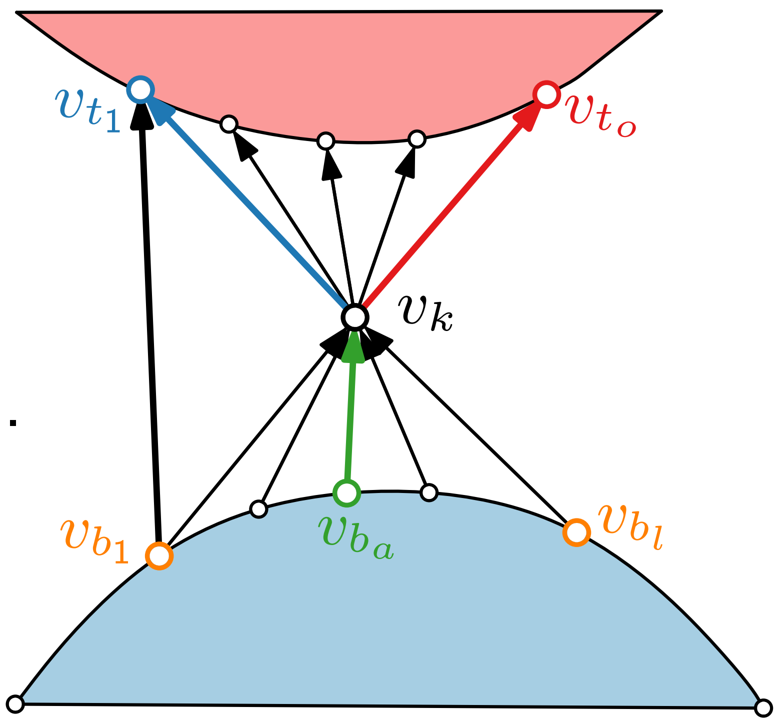
Lemma 1.
A left edge or right edge cannot be a base edge.

Proof. Suppose left edge (v_k, v_{t_1}) is base edge of v_{t_1}.

Refined Canonical Order → REL

We construct a REL as follows:

- For $i < j$, orient (v_i, v_j) from v_i to v_j;
- v_k has incoming edges from $v_{b_1}, \ldots, v_{b_l}$, we say that v_{b_1} is **left point** of v_k and v_{b_l} is **right point** of v_k.
- **Base edge** of v_k is (v_{b_a}, v_k), where $b_a \in \{b_1, \ldots, b_l\}$ is minimal.
- If $v_{t_1}, \ldots, v_{t_o}$ are higher numbered neighbors of v_k, we call (v_k, v_{t_1}) **left edge** and (v_k, v_{t_o}) **right edge**.



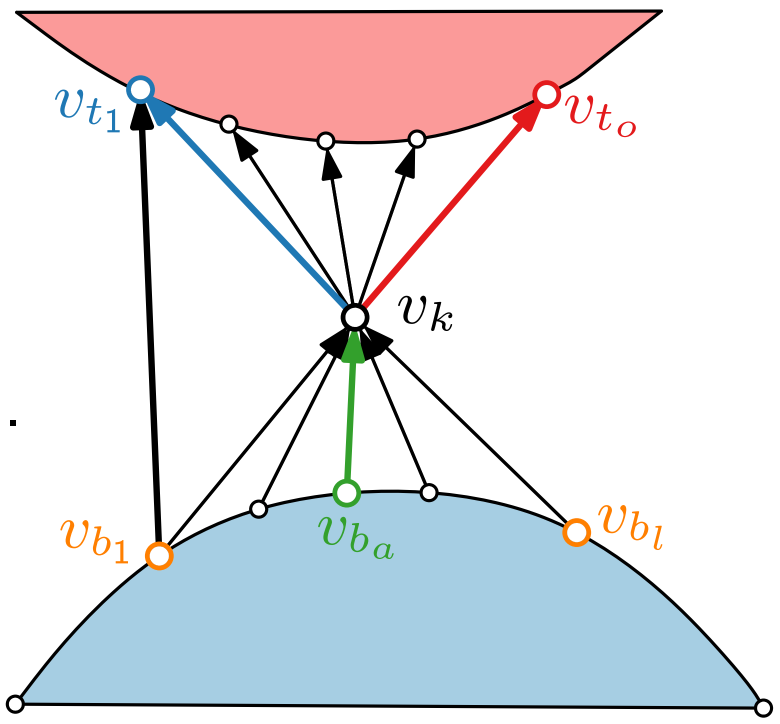
Lemma 1.
A left edge or right edge cannot be a base edge.

Proof. Suppose left edge (v_k, v_{t_1}) is base edge of v_{t_1}.
Since G triangulated, $(v_{b_1}, v_{t_1}) \in E(G)$.

Refined Canonical Order → REL

We construct a REL as follows:

- For $i < j$, orient (v_i, v_j) from v_i to v_j;
- v_k has incoming edges from $v_{b_1}, \ldots, v_{b_l}$, we say that v_{b_1} is **left point** of v_k and v_{b_l} is **right point** of v_k.
- **Base edge** of v_k is (v_{b_a}, v_k), where $b_a \in \{b_1, \ldots, b_l\}$ is minimal.
- If $v_{t_1}, \ldots, v_{t_o}$ are higher numbered neighbors of v_k, we call (v_k, v_{t_1}) **left edge** and (v_k, v_{t_o}) **right edge**.

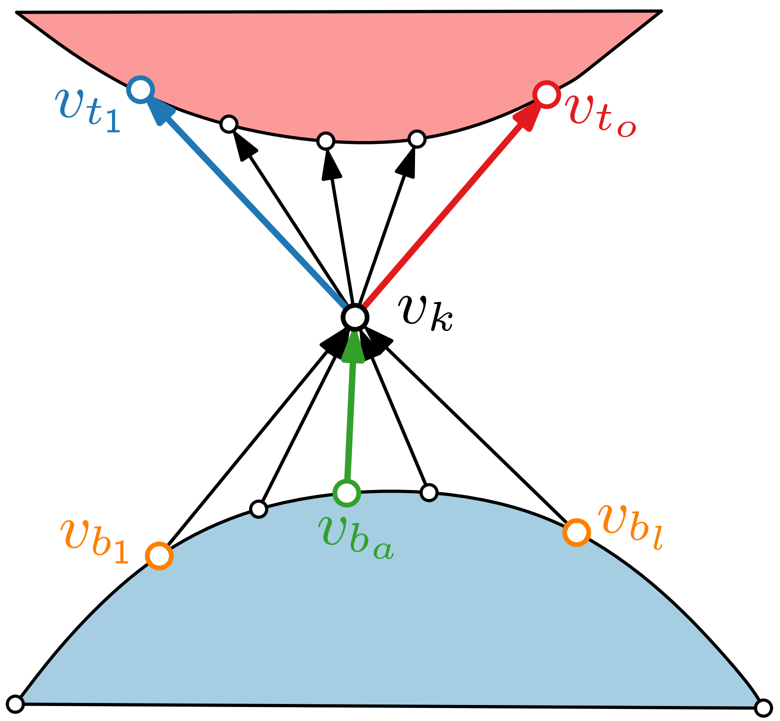


Lemma 1.
A left edge or right edge cannot be a base edge.

Proof. Suppose left edge (v_k, v_{t_1}) is base edge of v_{t_1}.
Since G triangulated, $(v_{b_1}, v_{t_1}) \in E(G)$.
Contradiction since $k > b_1$.

Refined Canonical Order → REL

Lemma 2.
An edge is either a left edge, a right edge or a base edge.

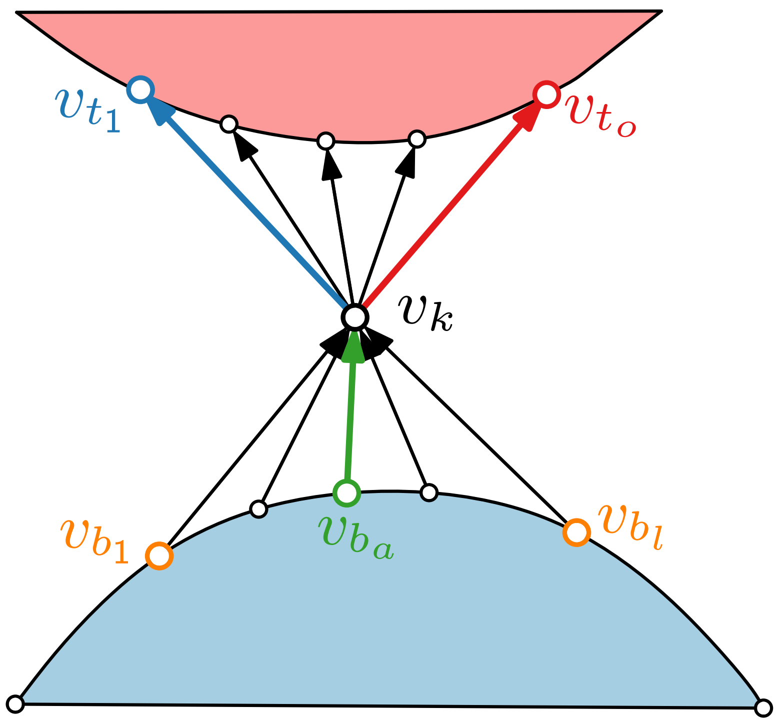


Refined Canonical Order → REL

Lemma 2.
An edge is either a left edge, a right edge or a base edge.

Proof.

- Exclusive "or" follows from Lemma 1.

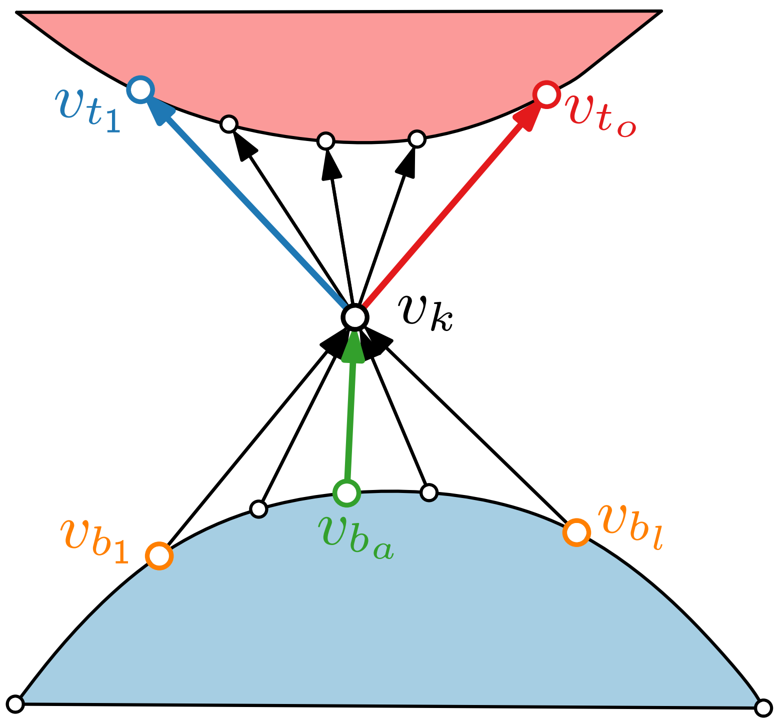


Refined Canonical Order → REL

Lemma 2.
An edge is either a left edge, a right edge or a base edge.

Proof.

- Exclusive "or" follows from Lemma 1.
- Let (v_{b_a}, v_k) be base edge of v_k.

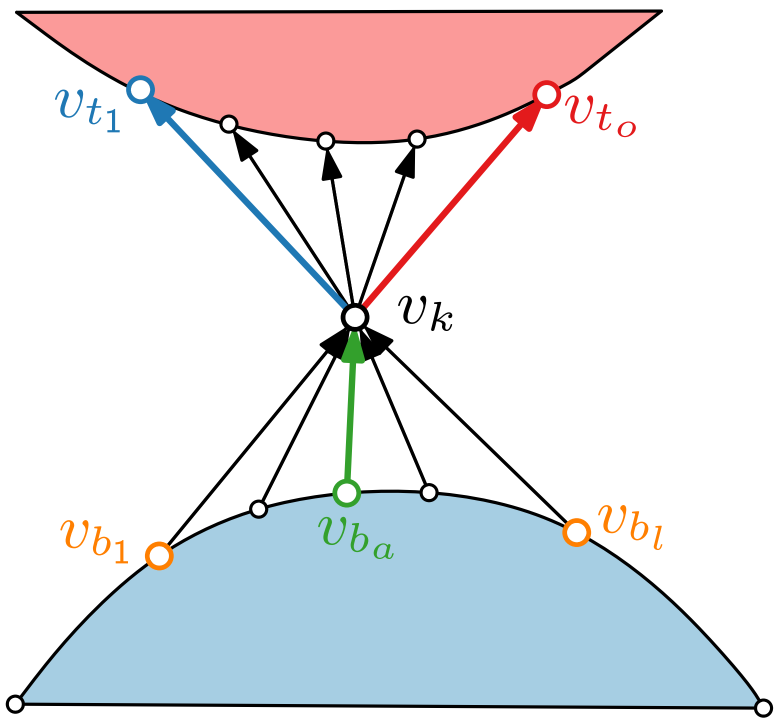


Refined Canonical Order → REL

Lemma 2.
An edge is either a left edge, a right edge or a base edge.

Proof.

- Exclusive "or" follows from Lemma 1.
- Let (v_{b_a}, v_k) be base edge of v_k.
- v_{b_a} is right point of $v_{b_{a-1}}$.

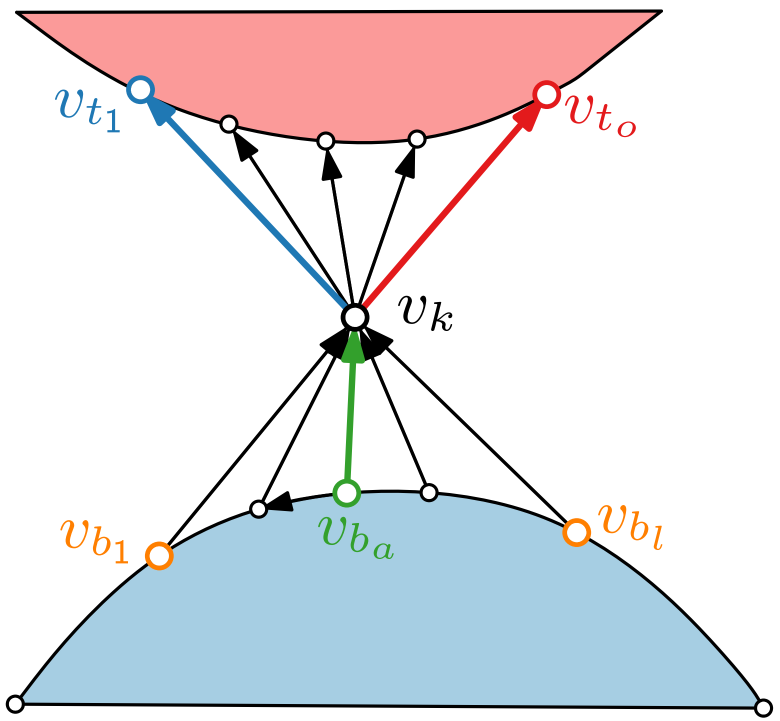


Refined Canonical Order $\rightarrow$ REL

Lemma 2.
An edge is either a left edge, a right edge or a base edge.

Proof.

- Exclusive "or" follows from Lemma 1.
- Let (v_{b_a}, v_k) be base edge of v_k.
- v_{b_a} is right point of $v_{b_{a-1}}$.

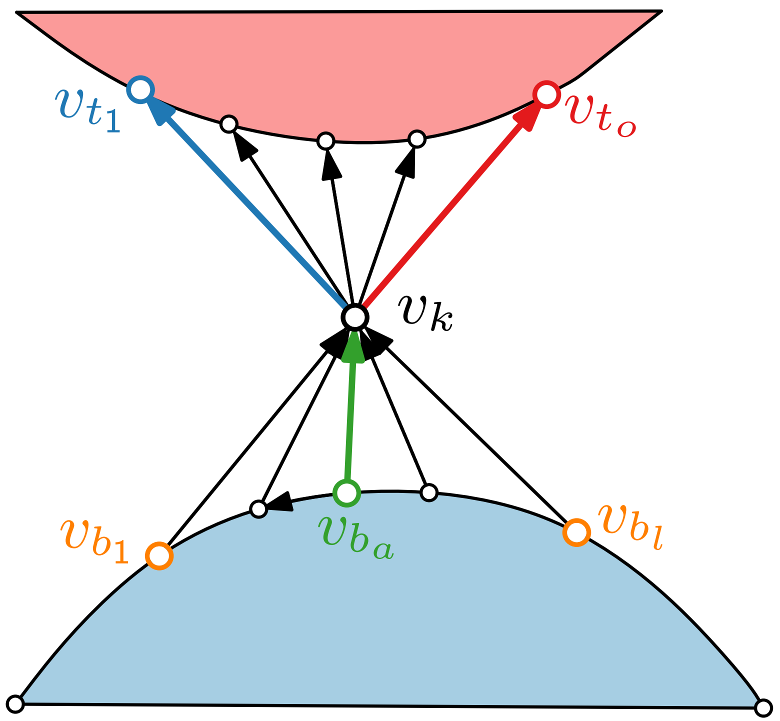


Refined Canonical Order → REL

Lemma 2.
An edge is either a left edge, a right edge or a base edge.

Proof.

- Exclusive "or" follows from Lemma 1.
- Let (v_{b_a}, v_k) be base edge of v_k.
- v_{b_a} is right point of $v_{b_{a-1}}$.
  - v_{b_i} has at least two higher-numbered neighbors.

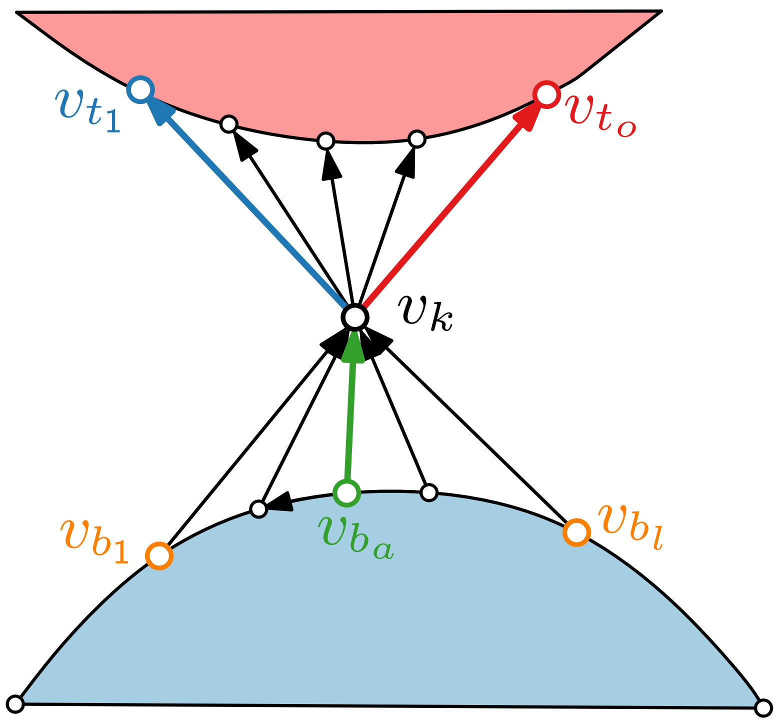


Refined Canonical Order → REL

Lemma 2.
An edge is either a left edge, a right edge or a base edge.

Proof.

- Exclusive "or" follows from Lemma 1.
- Let (v_{b_a}, v_k) be base edge of v_k.
- v_{b_a} is right point of $v_{b_{a-1}}$.
  - v_{b_i} has at least two higher-numbered neighbors.
  - One of them is v_k; the other one is $v_{b_{i-1}}$ or $v_{b_{i+1}}$.

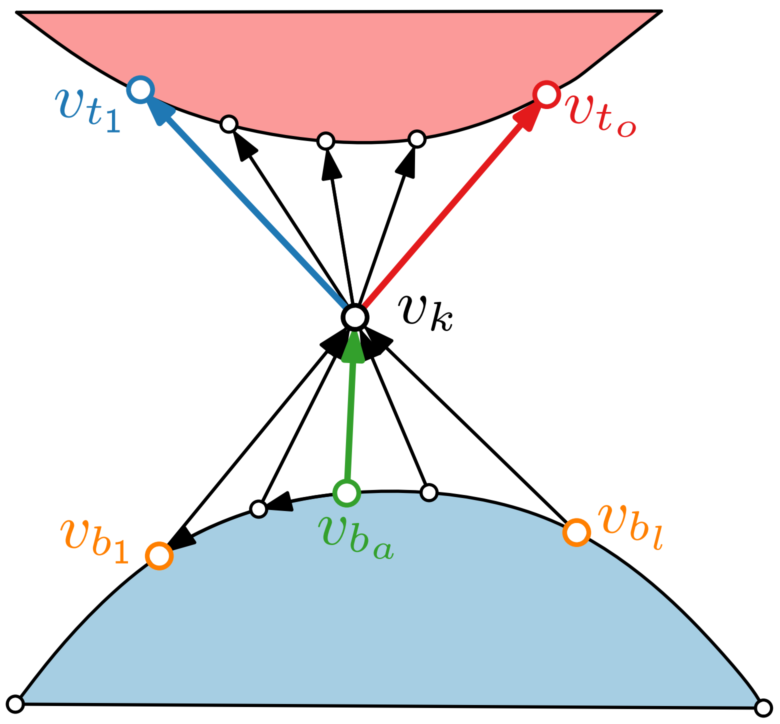


Refined Canonical Order $\rightarrow$ REL

Lemma 2.
An edge is either a left edge, a right edge or a base edge.

Proof.

- Exclusive "or" follows from Lemma 1.
- Let (v_{b_a}, v_k) be base edge of v_k.
- v_{b_a} is right point of $v_{b_{a-1}}$.
  - v_{b_i} has at least two higher-numbered neighbors.
  - One of them is v_k; the other one is $v_{b_{i-1}}$ or $v_{b_{i+1}}$.
  - For $1 \leq i < a-1$, it is $v_{b_{i-1}}$. Thus, v_{b_i} is right point of $v_{b_{i-1}}$.

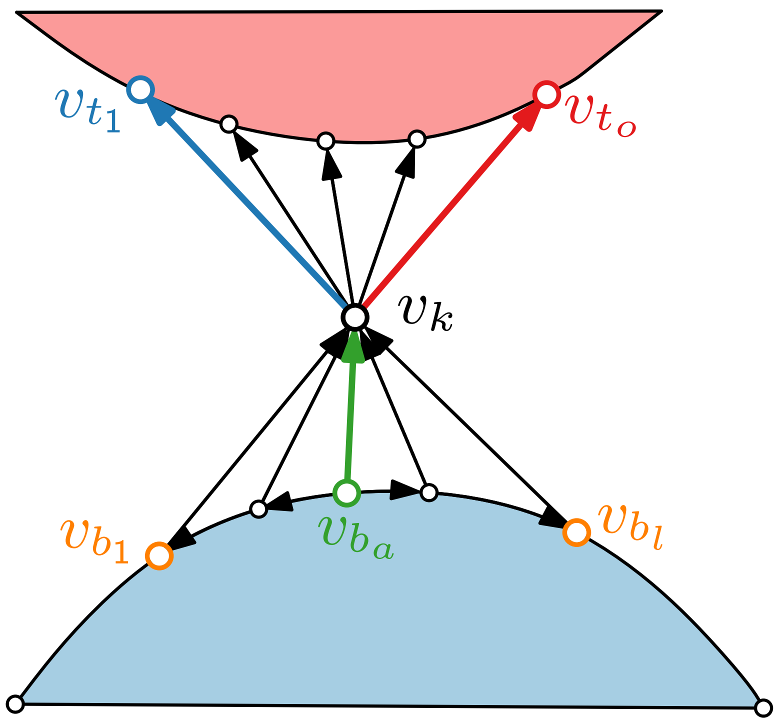


Refined Canonical Order → REL

Lemma 2.
An edge is either a left edge, a right edge or a base edge.

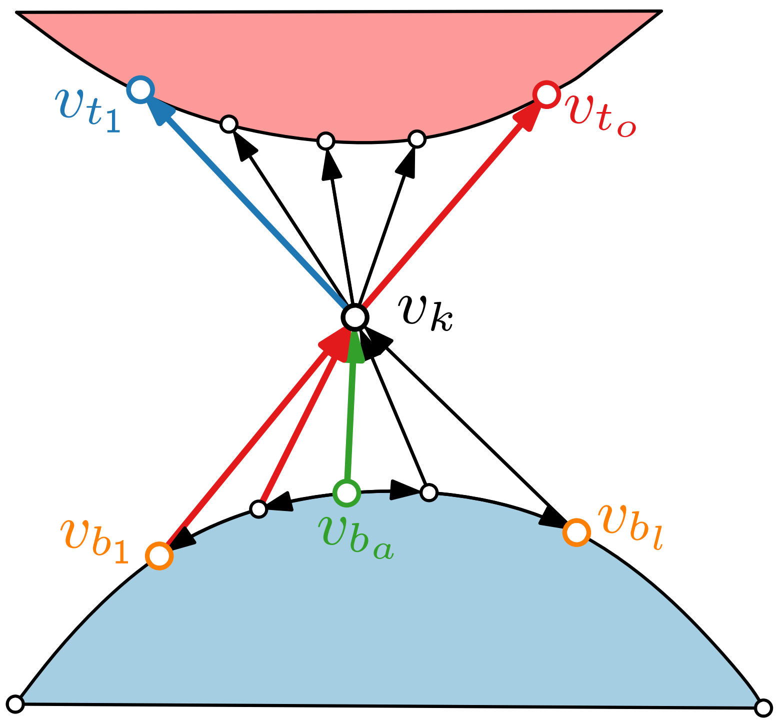
Proof.

- Exclusive "or" follows from Lemma 1.
- Let (v_{b_a}, v_k) be base edge of v_k.
- v_{b_a} is right point of $v_{b_{a-1}}$.
  - v_{b_i} has at least two higher-numbered neighbors.
  - One of them is v_k; the other one is $v_{b_{i-1}}$ or $v_{b_{i+1}}$.
  - For $1 \leq i < a-1$, it is $v_{b_{i-1}}$. Thus, v_{b_i} is right point of $v_{b_{i-1}}$.
- Analogously, v_{b_i} is left point of $v_{b_{i+1}}$ for $i \geq a$.



Refined Canonical Order → REL

Lemma 2.
An edge is either a left edge, a right edge or a base edge.



Proof.

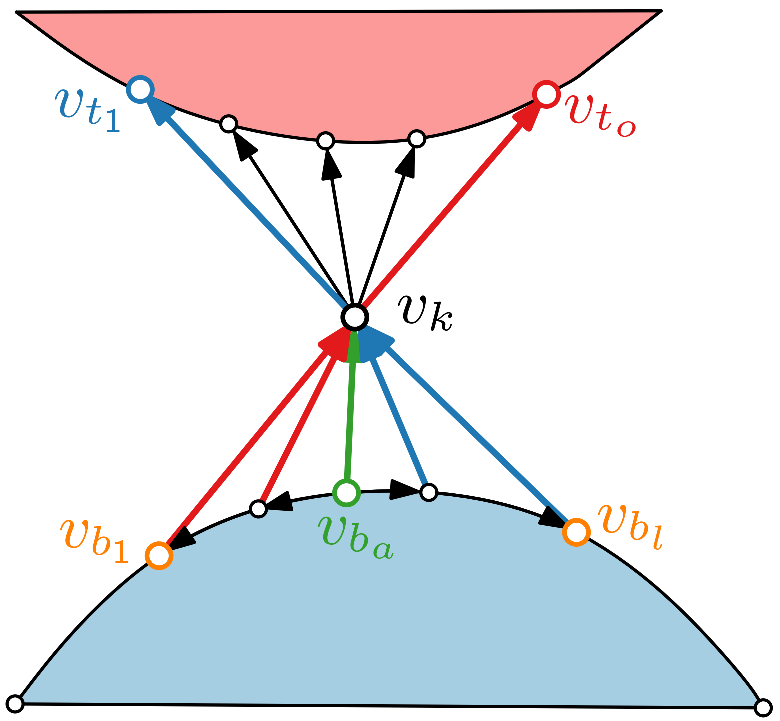
- Exclusive "or" follows from Lemma 1.
- Let (v_{b_a}, v_k) be base edge of v_k.
- v_{b_a} is right point of $v_{b_{a-1}}$.
  - v_{b_i} has at least two higher-numbered neighbors.
  - One of them is v_k; the other one is $v_{b_{i-1}}$ or $v_{b_{i+1}}$.
  - For $1 \leq i < a-1$, it is $v_{b_{i-1}}$. Thus, v_{b_i} is right point of $v_{b_{i-1}}$.
- Analogously, v_{b_i} is left point of $v_{b_{i+1}}$ for $i \geq a$.
- Edges (v_{b_i}, v_k), $1 \leq i < a-1$, are right edges.

Refined Canonical Order → REL

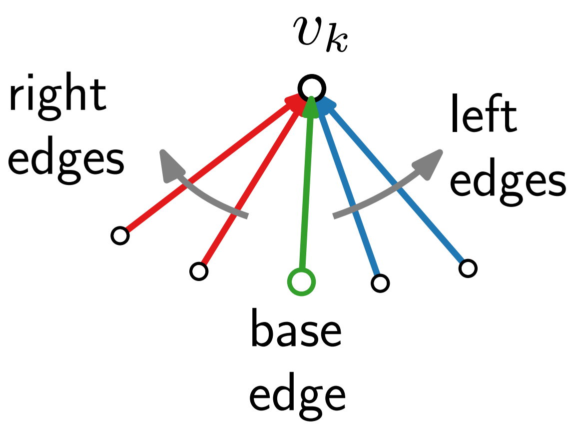
Lemma 2.
An edge is either a left edge, a right edge or a base edge.

Proof.

- Exclusive "or" follows from Lemma 1.
- Let (v_{b_a}, v_k) be base edge of v_k.
- v_{b_a} is right point of $v_{b_{a-1}}$.
  - v_{b_i} has at least two higher-numbered neighbors.
  - One of them is v_k; the other one is $v_{b_{i-1}}$ or $v_{b_{i+1}}$.
  - For $1 \leq i < a-1$, it is $v_{b_{i-1}}$. Thus, v_{b_i} is right point of $v_{b_{i-1}}$.
- Analogously, v_{b_i} is left point of $v_{b_{i+1}}$ for $i \geq a$.
- Edges (v_{b_i}, v_k), $1 \leq i < a-1$, are right edges.
- Similarly, (v_{b_i}, v_k), for $a+1 \leq i \leq l$, are left edges.



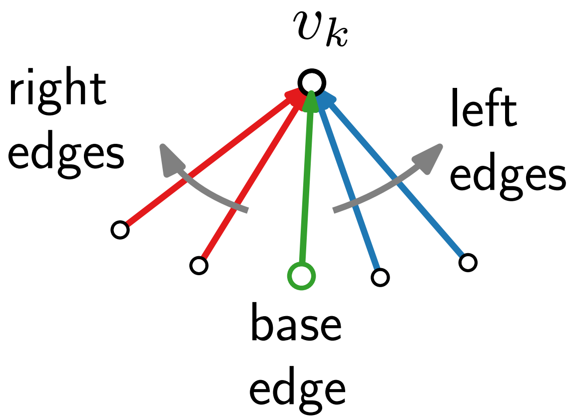
Refined Canonical Order → REL



Refined Canonical Order → REL

Coloring.

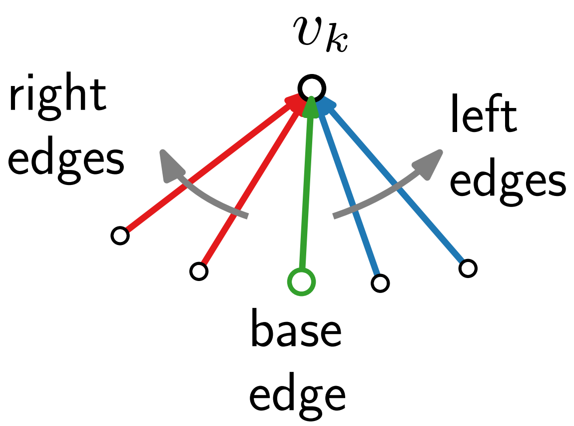
- Color right (left) edges in red (blue).



Refined Canonical Order → REL

Coloring.

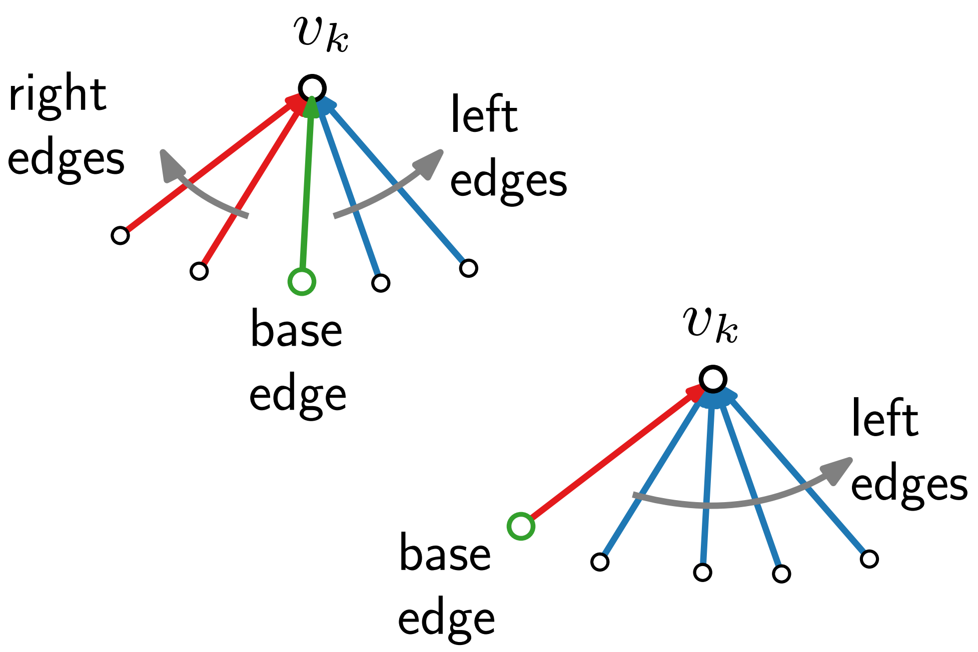
- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.



Refined Canonical Order → REL

Coloring.

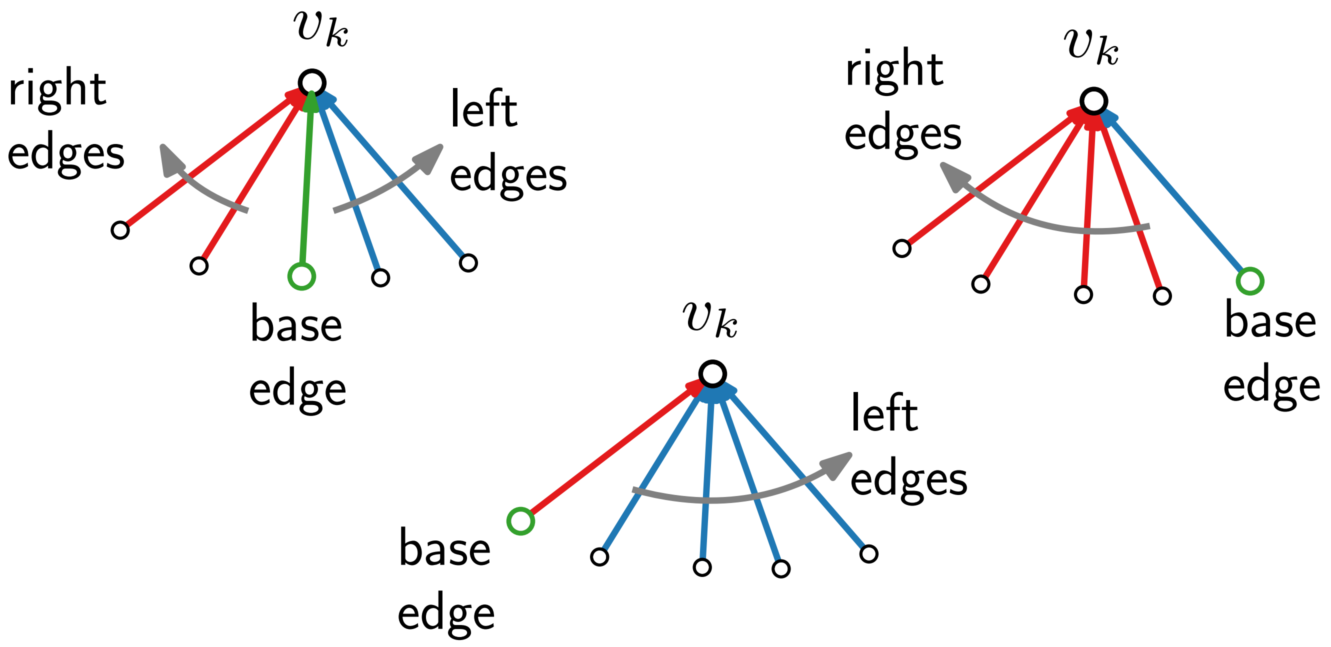
- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.



Refined Canonical Order → REL

Coloring.

- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.

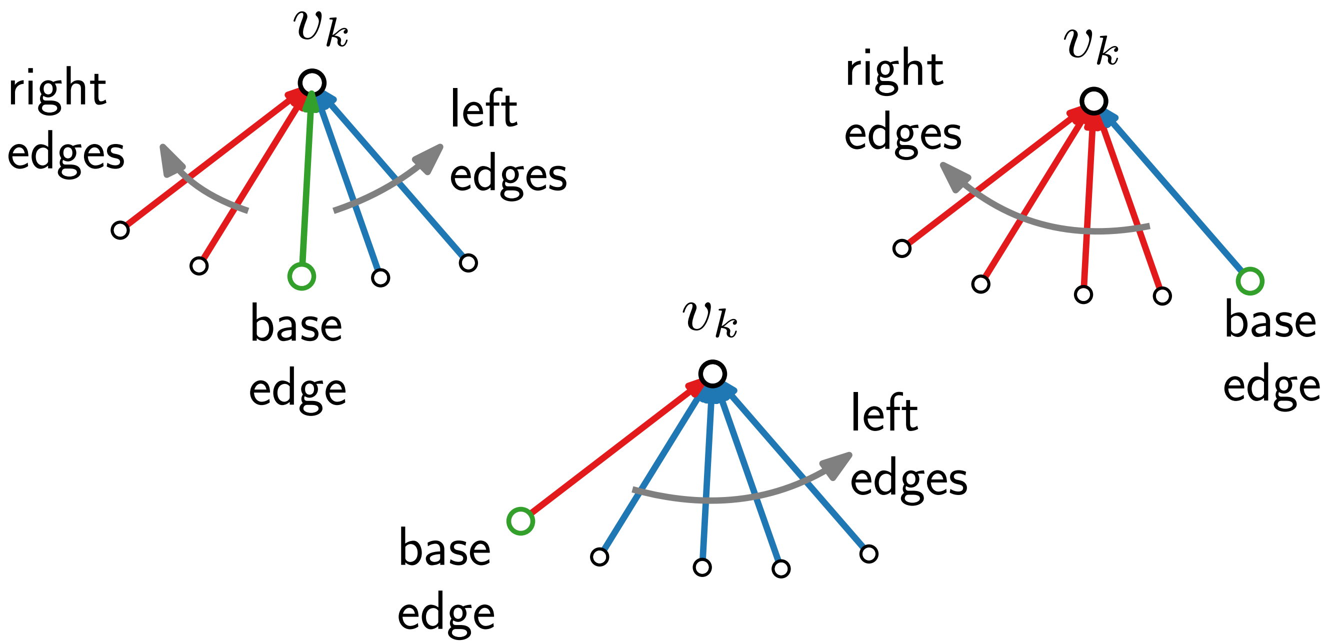


Refined Canonical Order → REL

Coloring.

- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.

Let T_r be the red edges and T_b the blue edges.

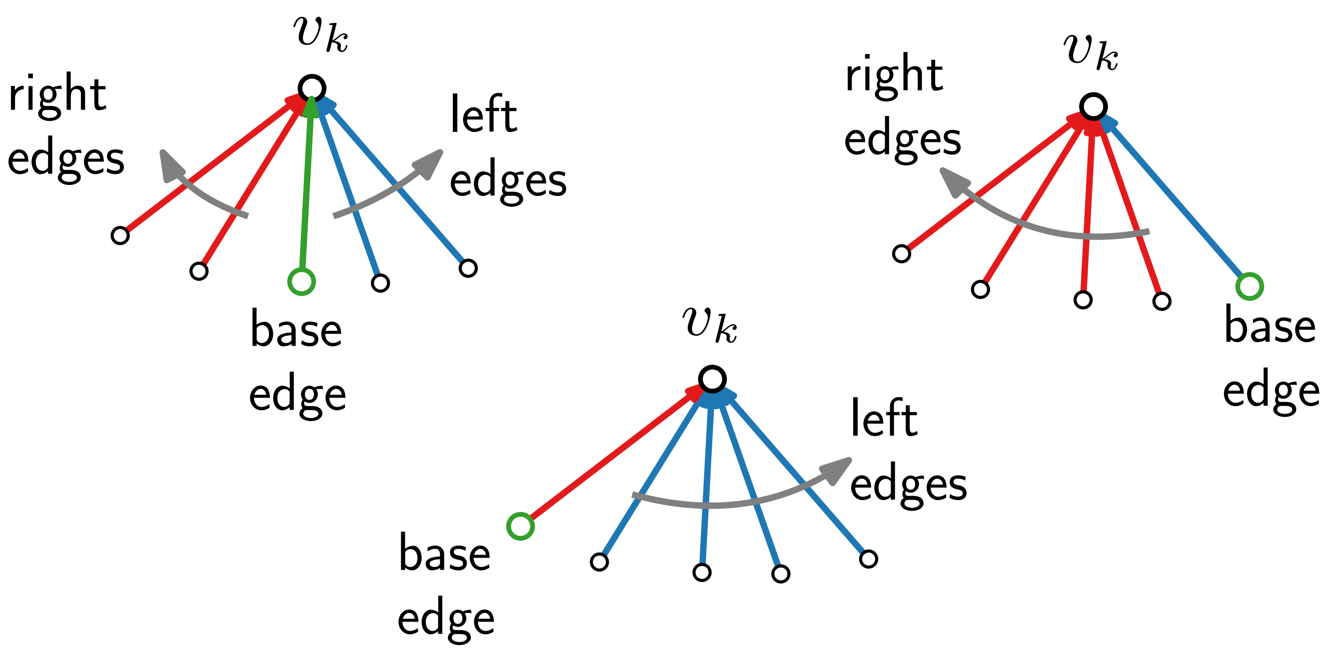


Refined Canonical Order → REL

Coloring.

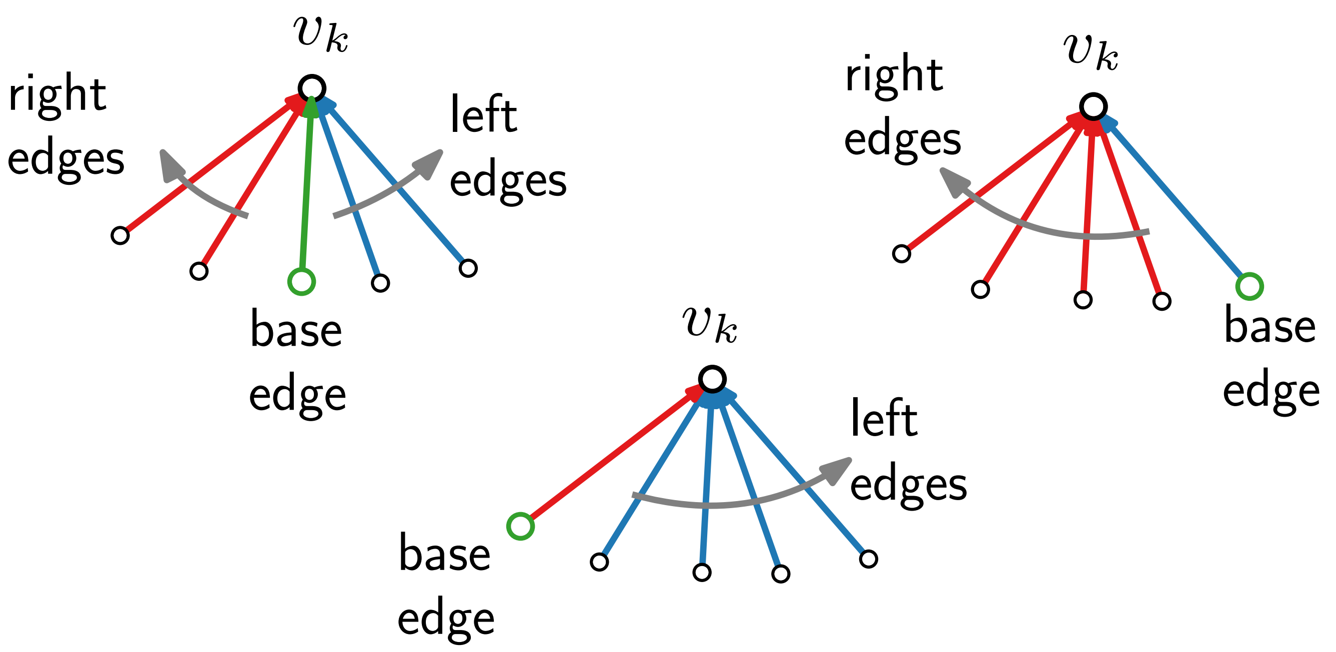
- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.

Let T_r be the red edges and T_b the blue edges.



Lemma 3.
$\{T_r, T_b\}$ is a regular edge labeling.

Refined Canonical Order → REL



Coloring.

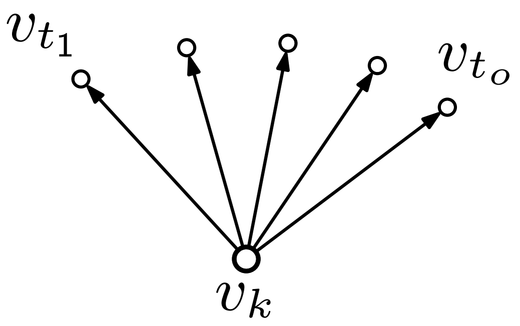
- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.

Let T_r be the red edges and T_b the blue edges.

Lemma 3.
$\{T_r, T_b\}$ is a regular edge labeling.

Proof.

$t_o \geq 2$

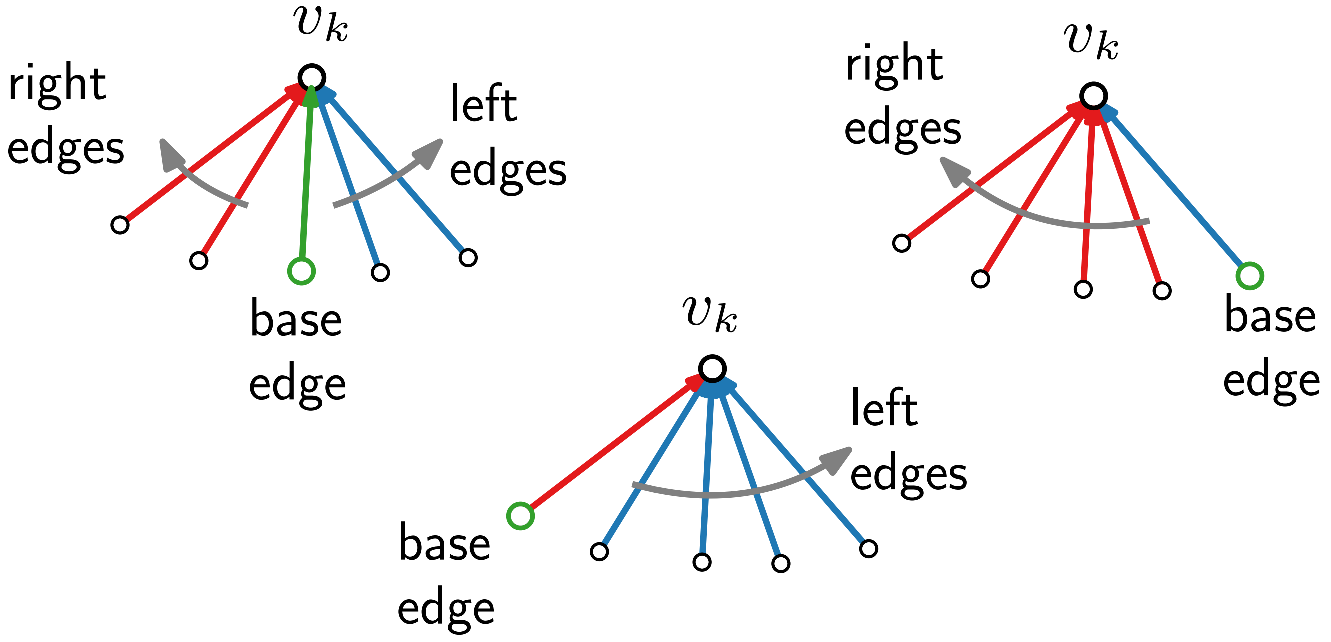


Refined Canonical Order → REL

Coloring.

- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.

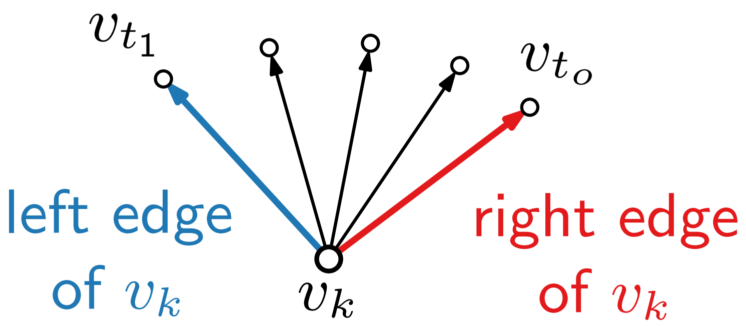
Let T_r be the red edges and T_b the blue edges.



Lemma 3.
$\{T_r, T_b\}$ is a regular edge labeling.

Proof.

$t_o \geq 2$

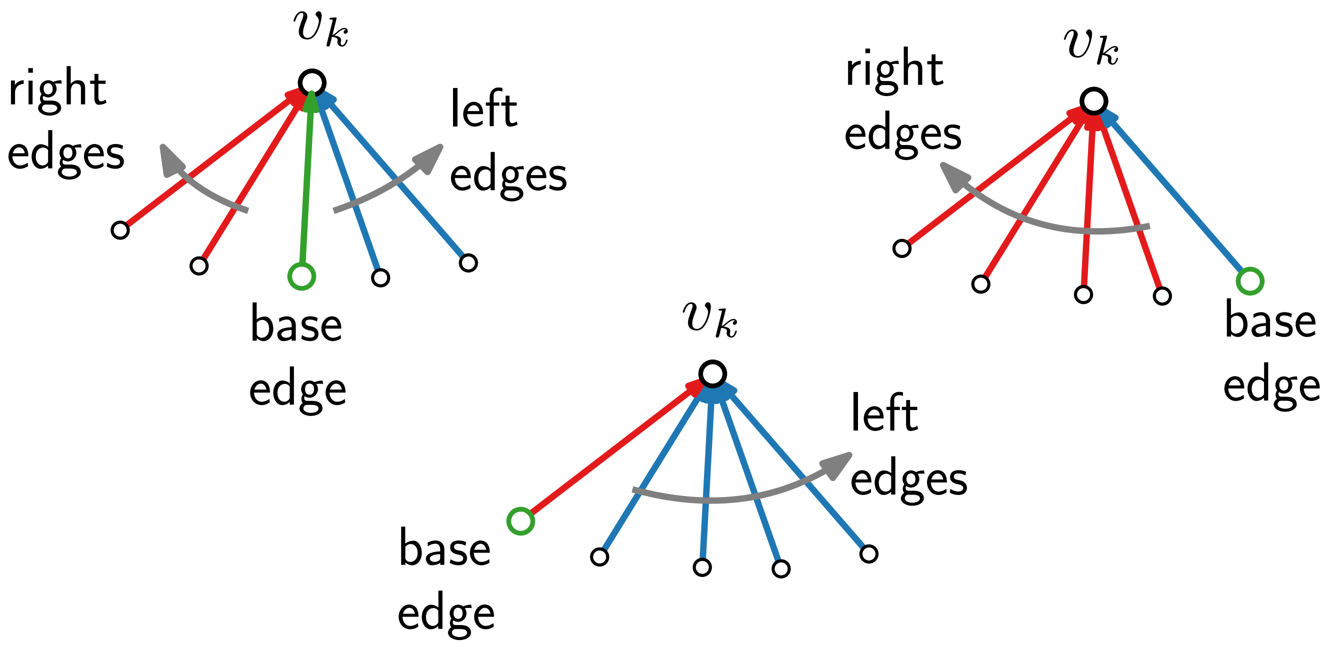


Refined Canonical Order → REL

Coloring.

- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.

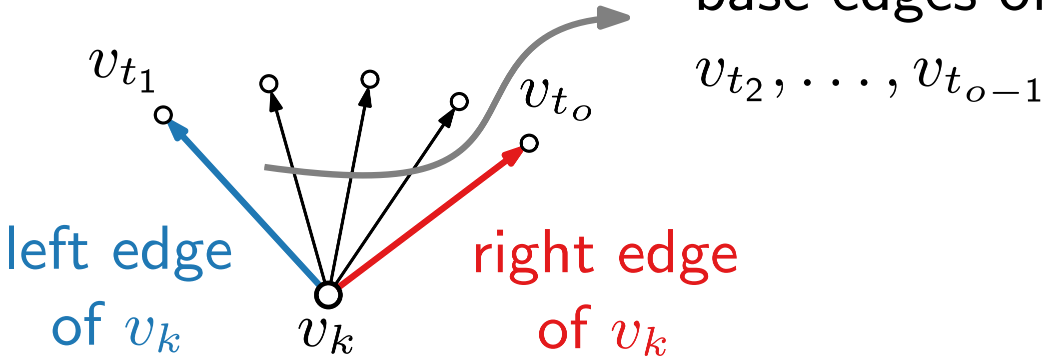
Let T_r be the red edges and T_b the blue edges.



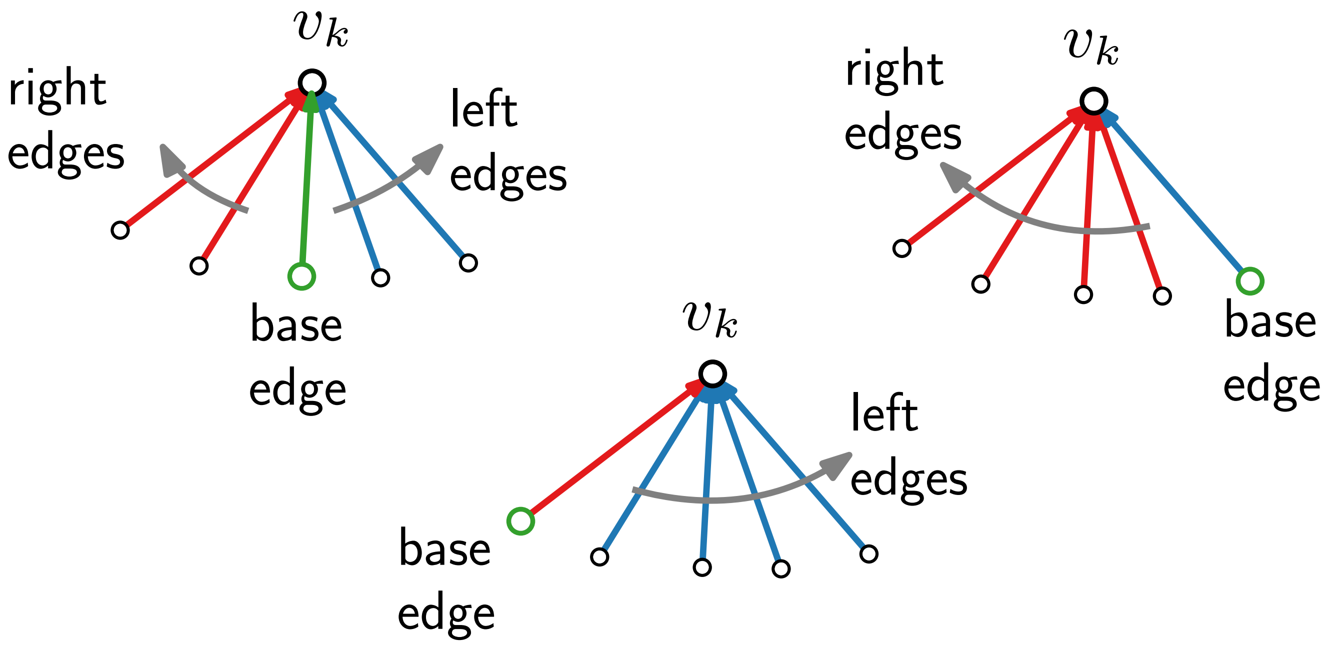
Lemma 3.
$\{T_r, T_b\}$ is a regular edge labeling.

Proof.

$t_o \geq 2$



Refined Canonical Order → REL



Coloring.

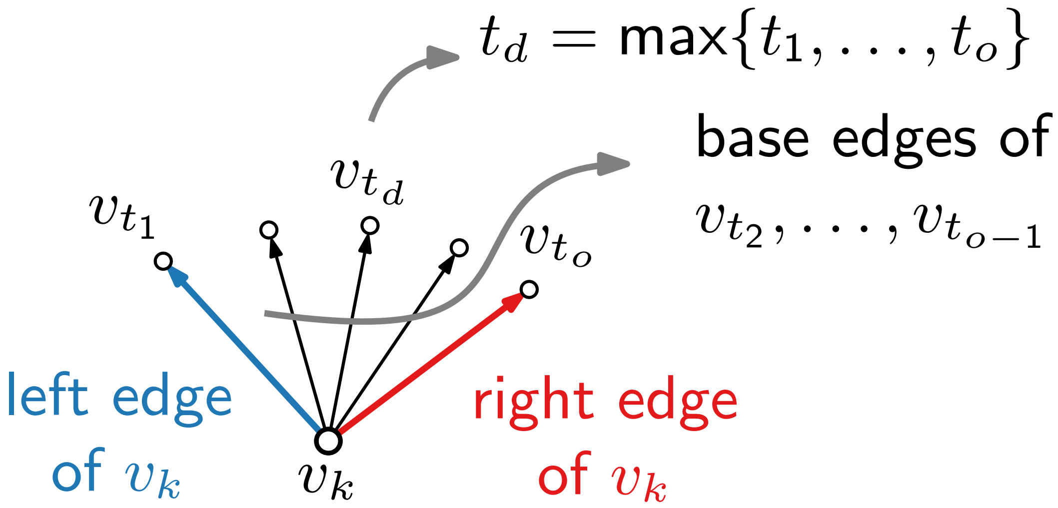
- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.

Let T_r be the red edges and T_b the blue edges.

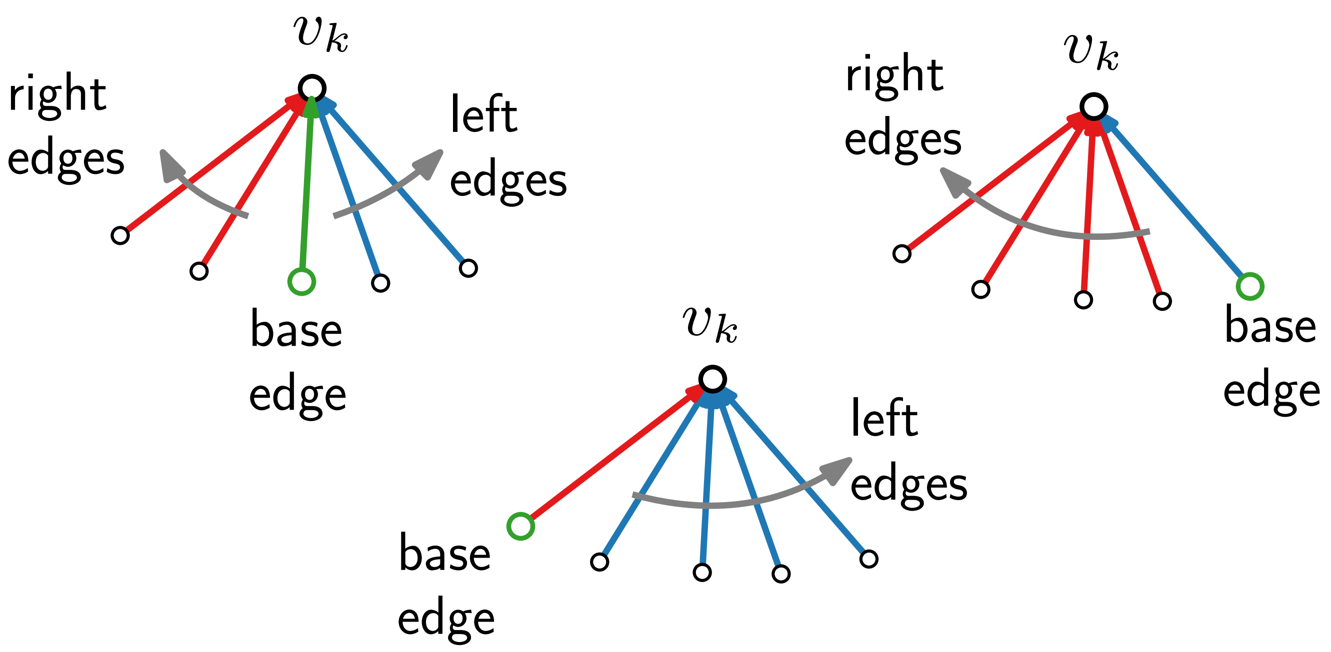
Lemma 3.
$\{T_r, T_b\}$ is a regular edge labeling.

Proof.

$t_o \geq 2$



Refined Canonical Order $\rightarrow$ REL



Coloring.

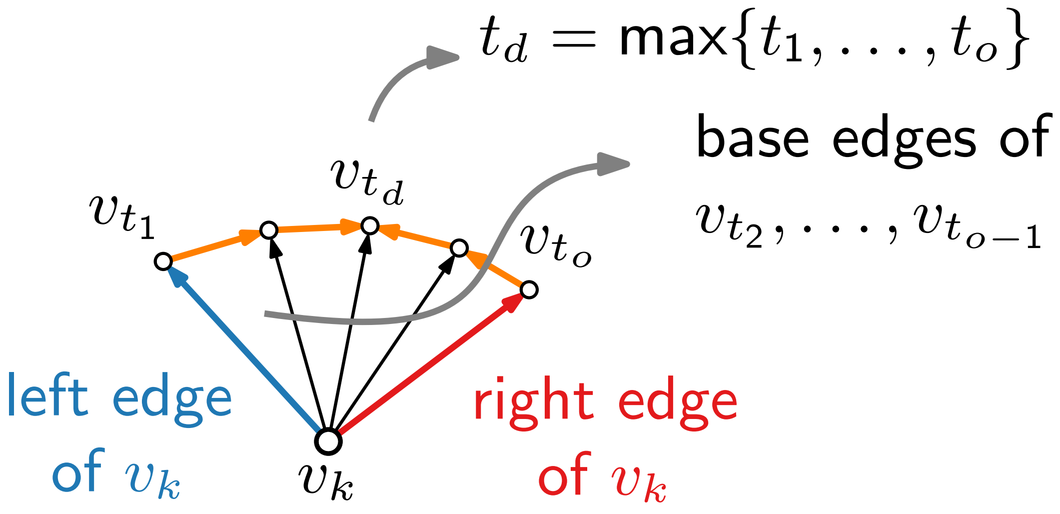
- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.

Let T_r be the red edges and T_b the blue edges.

Lemma 3.
$\{T_r, T_b\}$ is a regular edge labeling.

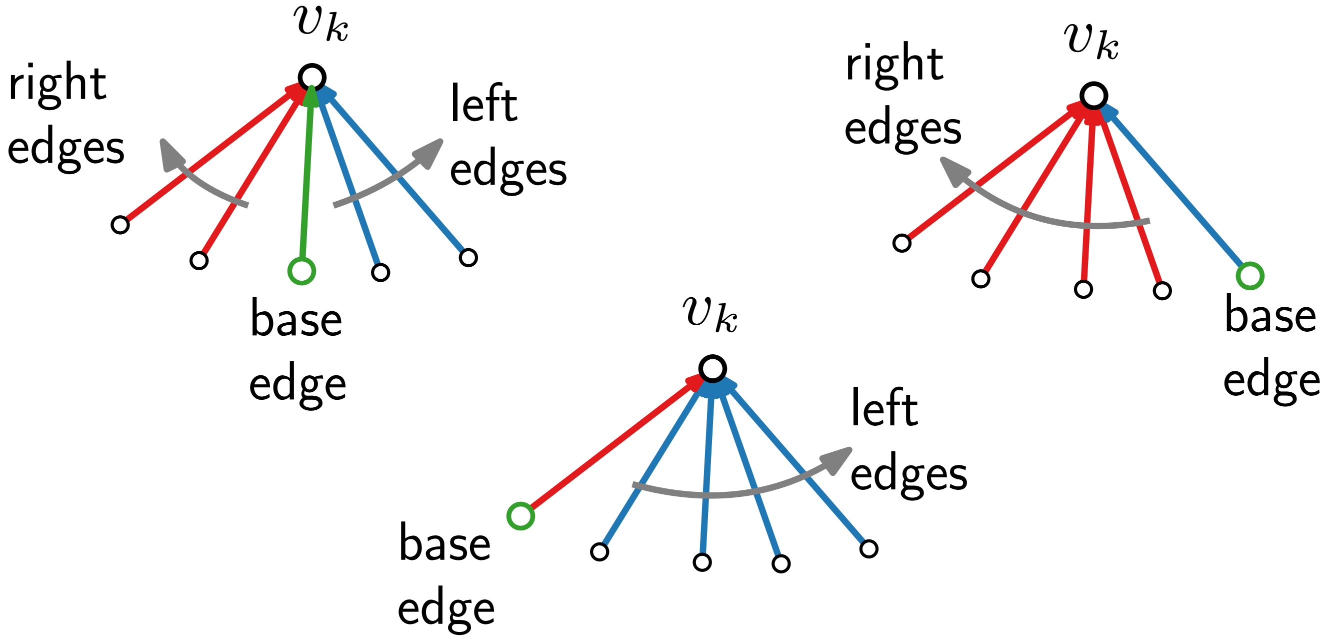
Proof.

$t_o \geq 2$



- $t_1 < t_2 < \ldots < t_d$ and $t_d > t_{d+1} > \ldots > t_o$

Refined Canonical Order → REL



Coloring.

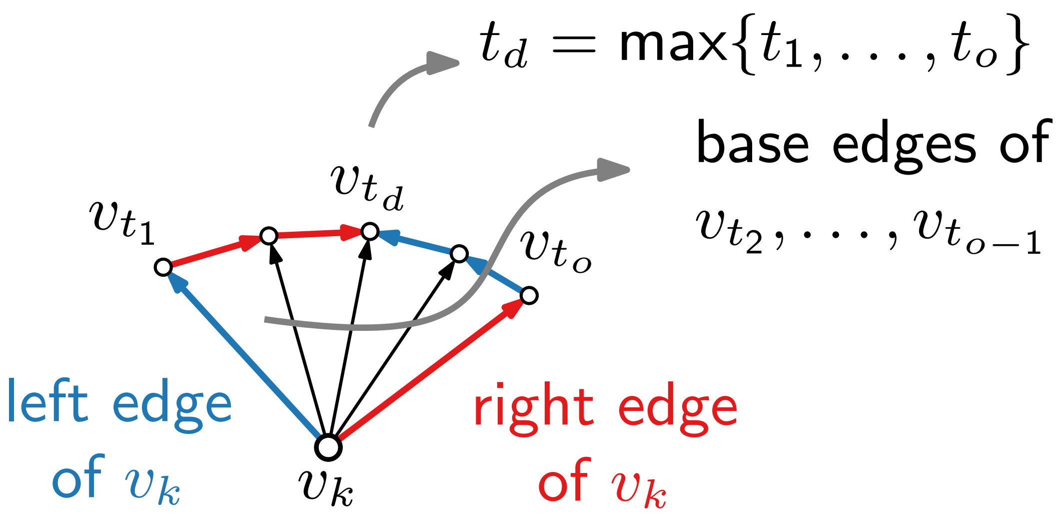
- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.

Let T_r be the red edges and T_b the blue edges.

Lemma 3.
$\{T_r, T_b\}$ is a regular edge labeling.

Proof.

$t_o \geq 2$



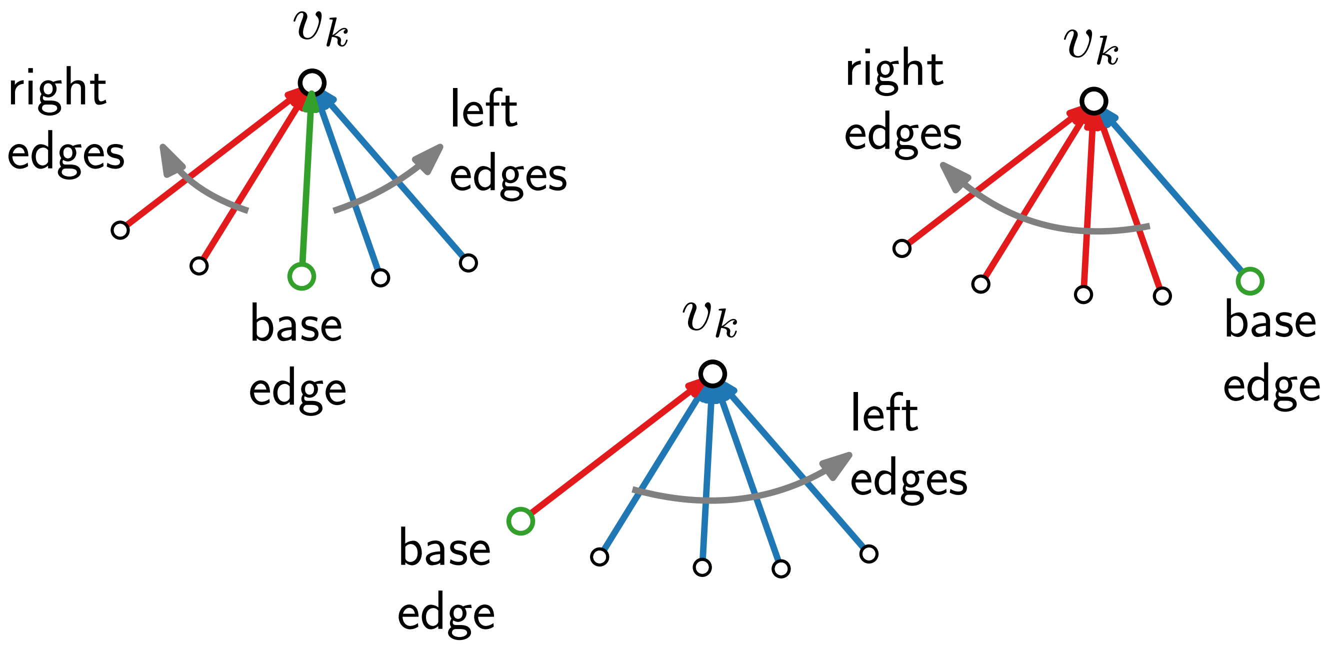
- $t_1 < t_2 < \ldots < t_d$ and $t_d > t_{d+1} > \ldots > t_o$

Refined Canonical Order → REL

Coloring.

- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.

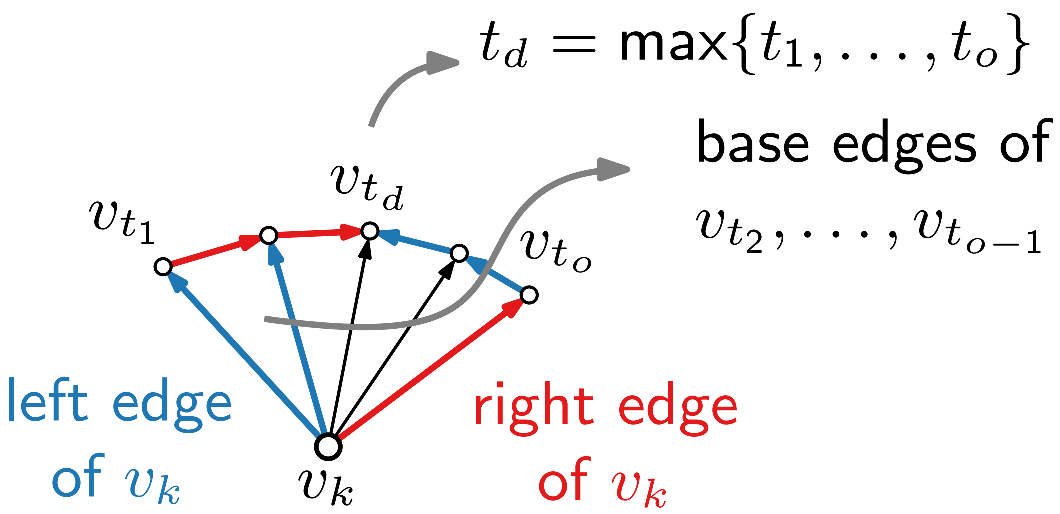
Let T_r be the red edges and T_b the blue edges.



Lemma 3.
$\{T_r, T_b\}$ is a regular edge labeling.

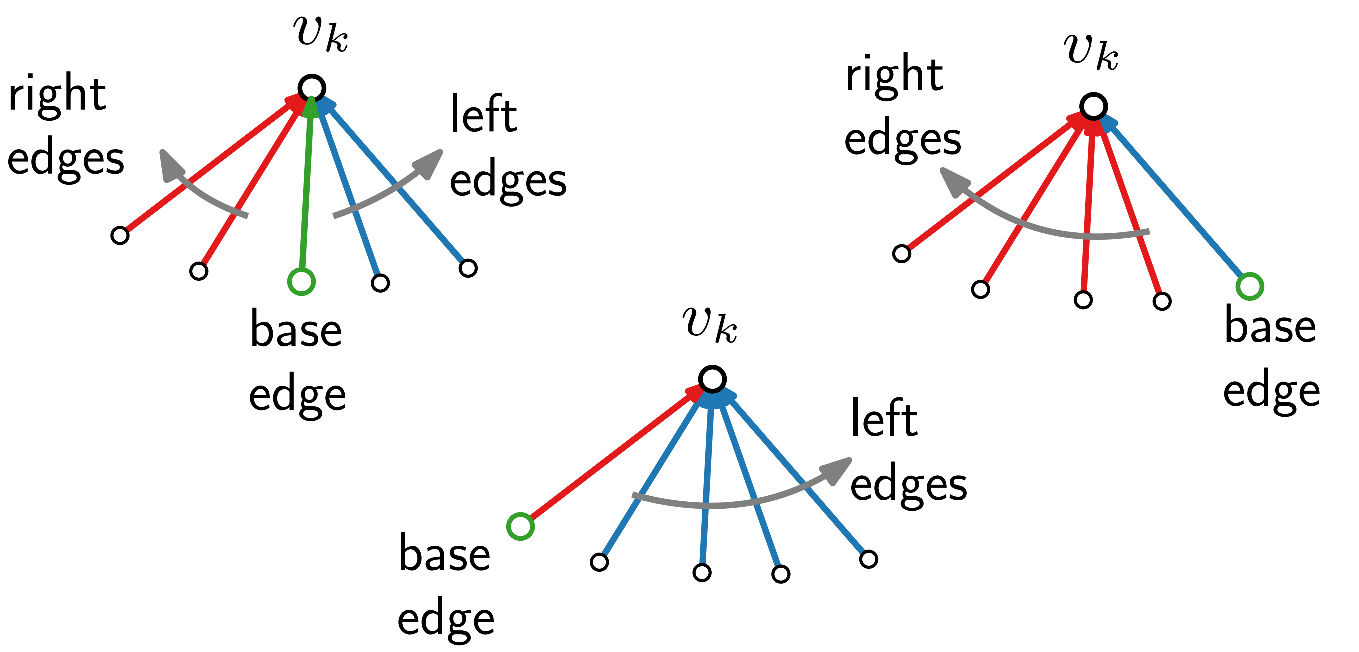
Proof.

$t_o \geq 2$



- $t_1 < t_2 < \ldots < t_d$ and $t_d > t_{d+1} > \ldots > t_o$
- (v_k, v_{t_i}), $2 \leq i \leq d - 1$ are blue

Refined Canonical Order → REL



Coloring.

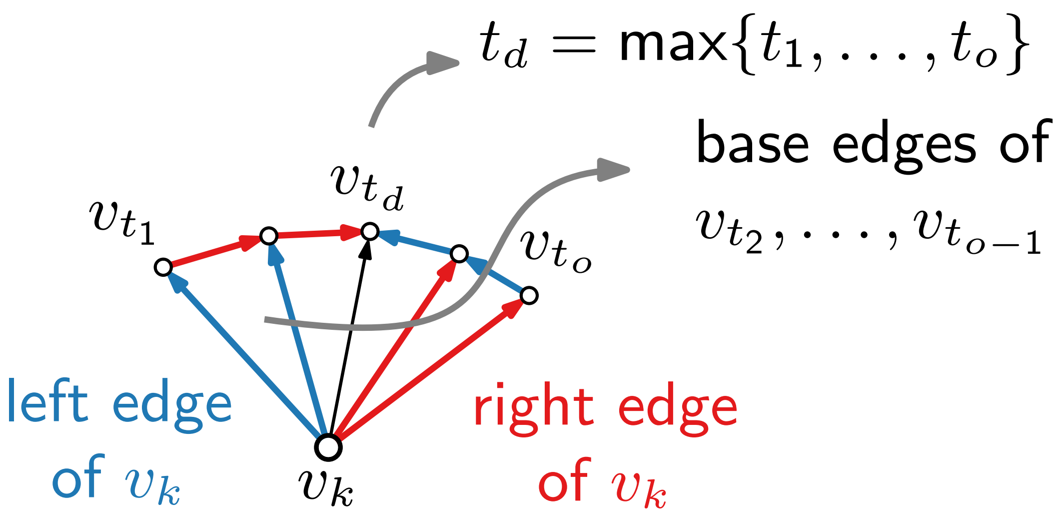
- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.

Let T_r be the red edges and T_b the blue edges.

Lemma 3.
$\{T_r, T_b\}$ is a regular edge labeling.

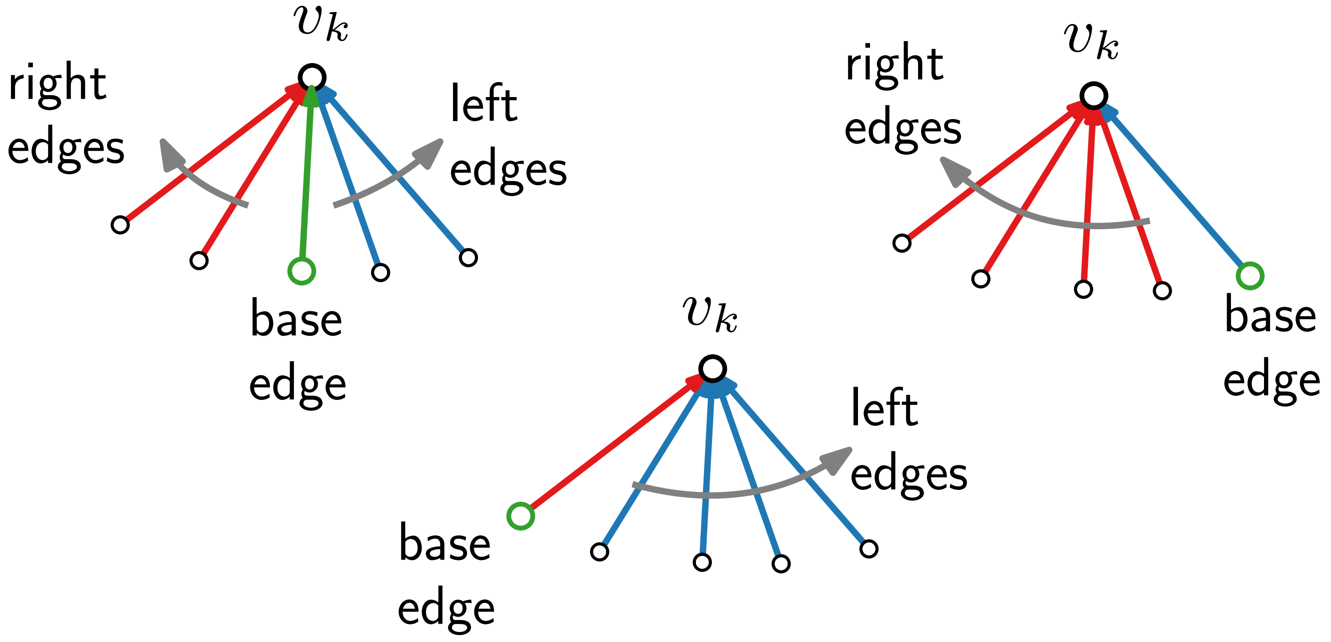
Proof.

$t_o \geq 2$



- $t_1 < t_2 < \ldots < t_d$ and $t_d > t_{d+1} > \ldots > t_o$
- (v_k, v_{t_i}), $2 \leq i \leq d-1$ are blue
- (v_k, v_{t_i}), $d+1 \leq i \leq o-1$ are red

Refined Canonical Order → REL



Coloring.

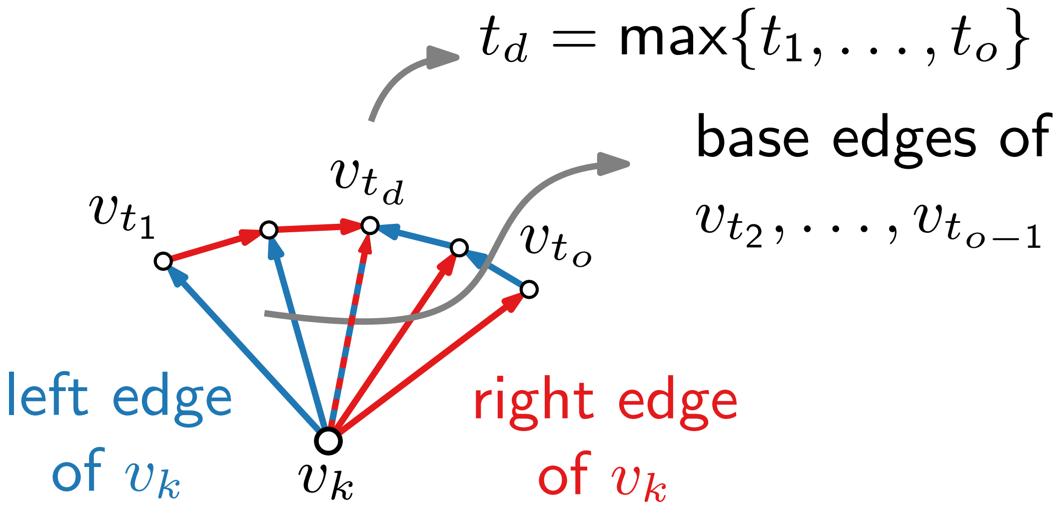
- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.

Let T_r be the red edges and T_b the blue edges.

> **Lemma 3.**
> $\{T_r, T_b\}$ is a regular edge labeling.

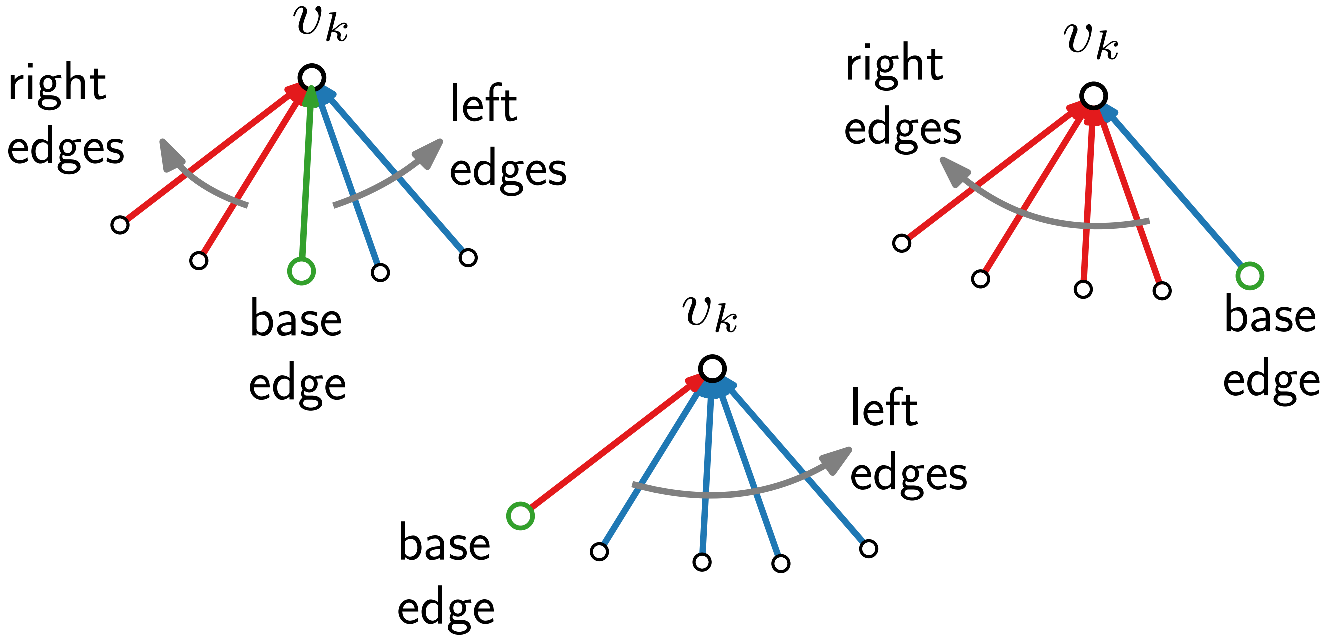
Proof.

$t_o \geq 2$



- $t_1 < t_2 < \ldots < t_d$ and $t_d > t_{d+1} > \ldots > t_o$
- (v_k, v_{t_i}), $2 \leq i \leq d-1$ are blue
- (v_k, v_{t_i}), $d+1 \leq i \leq o-1$ are red
- (v_k, v_{t_d}) is either red or blue

Refined Canonical Order → REL



Coloring.

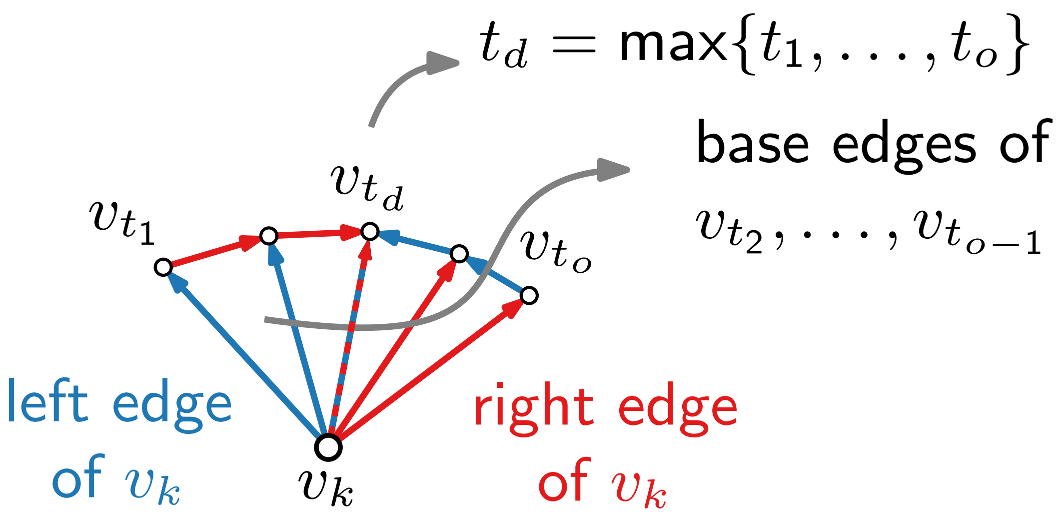
- Color right (left) edges in red (blue).
- Color a base edge (v_{b_i}, v_k) red if $i = 1$ and blue if $i = l$ and otherwise arbitrarily.

Let T_r be the red edges and T_b the blue edges.

Lemma 3.
$\{T_r, T_b\}$ is a regular edge labeling.

Proof.

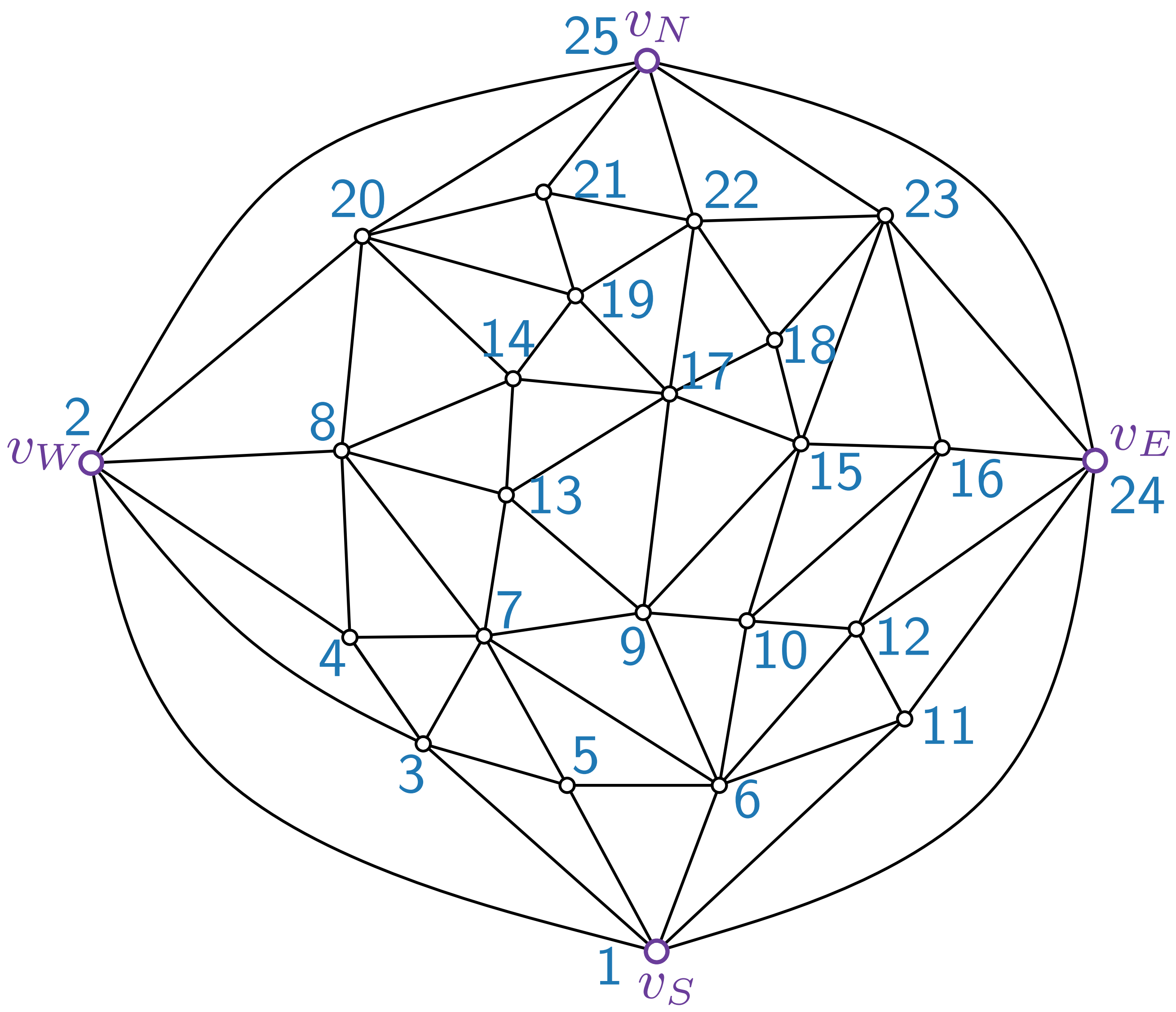
$t_o \geq 2$



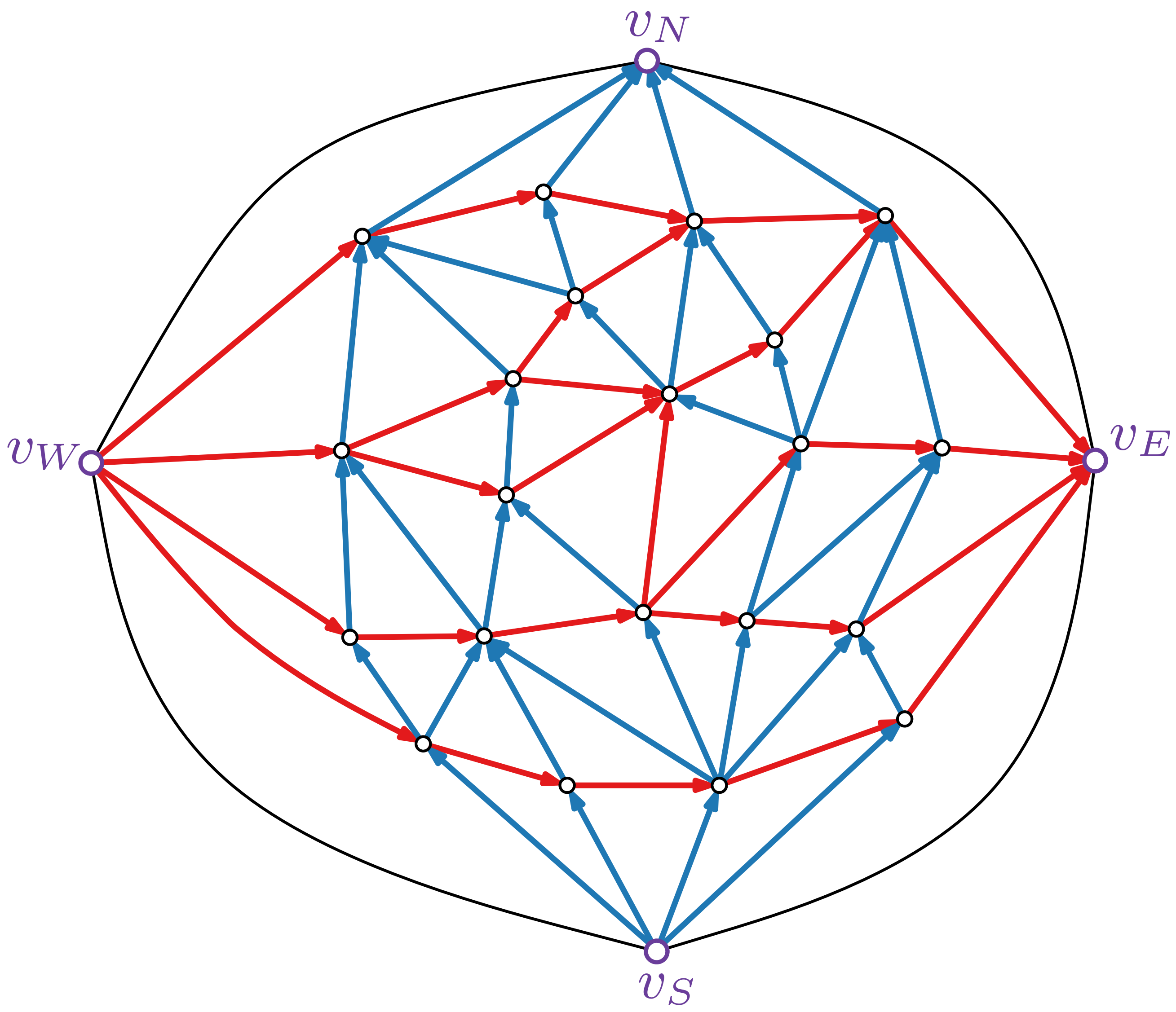
- $t_1 < t_2 < \dots < t_d$ and $t_d > t_{d+1} > \dots > t_o$
- (v_k, v_{t_i}), $2 \leq i \leq d-1$ are blue
- (v_k, v_{t_i}), $d+1 \leq i \leq o-1$ are red
- (v_k, v_{t_d}) is either red or blue

⇒ Circular order of outgoing edges at v_k correct.

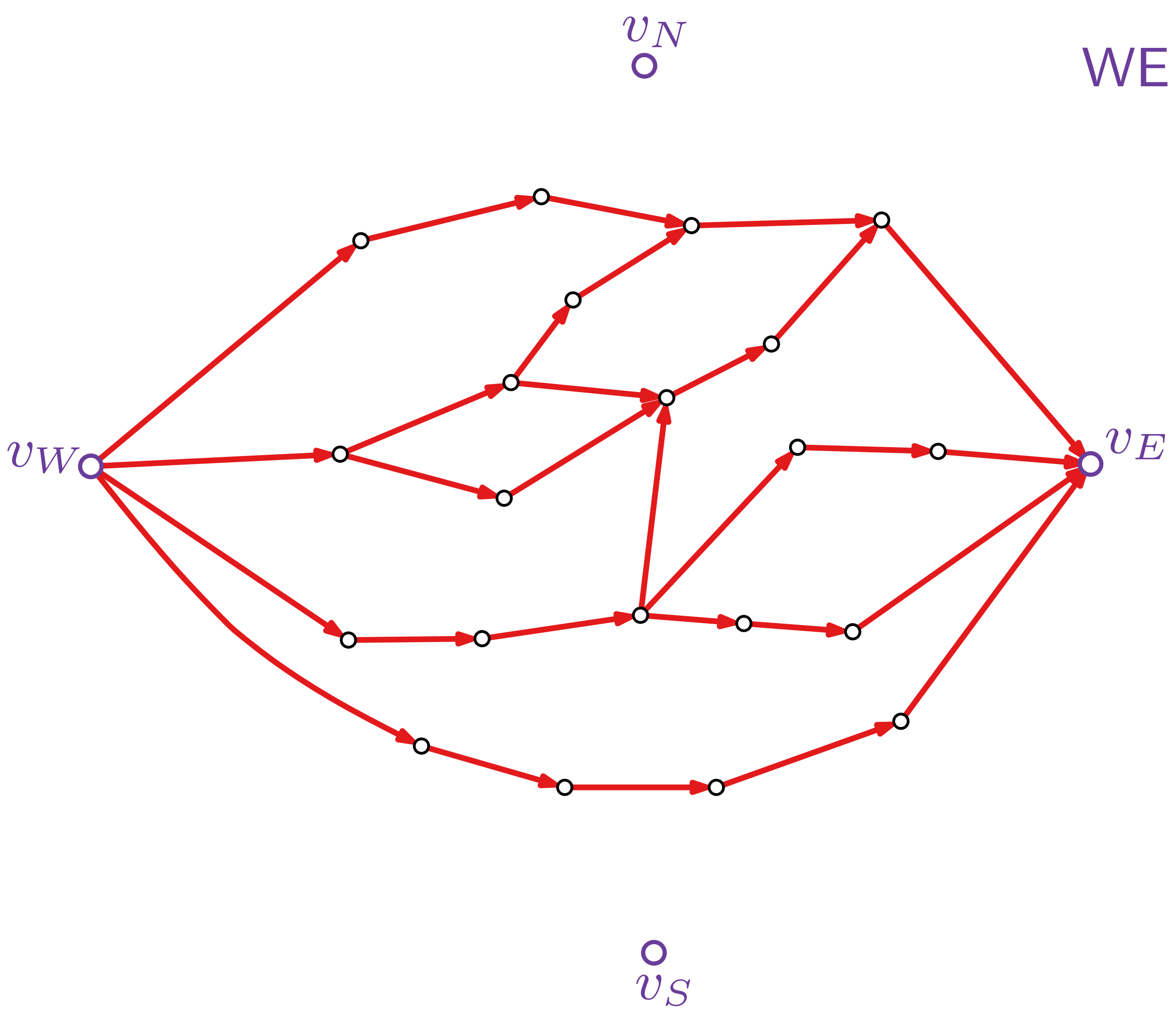
From REL to st-Digraphs to Coordinates



From REL to st-Digraphs to Coordinates

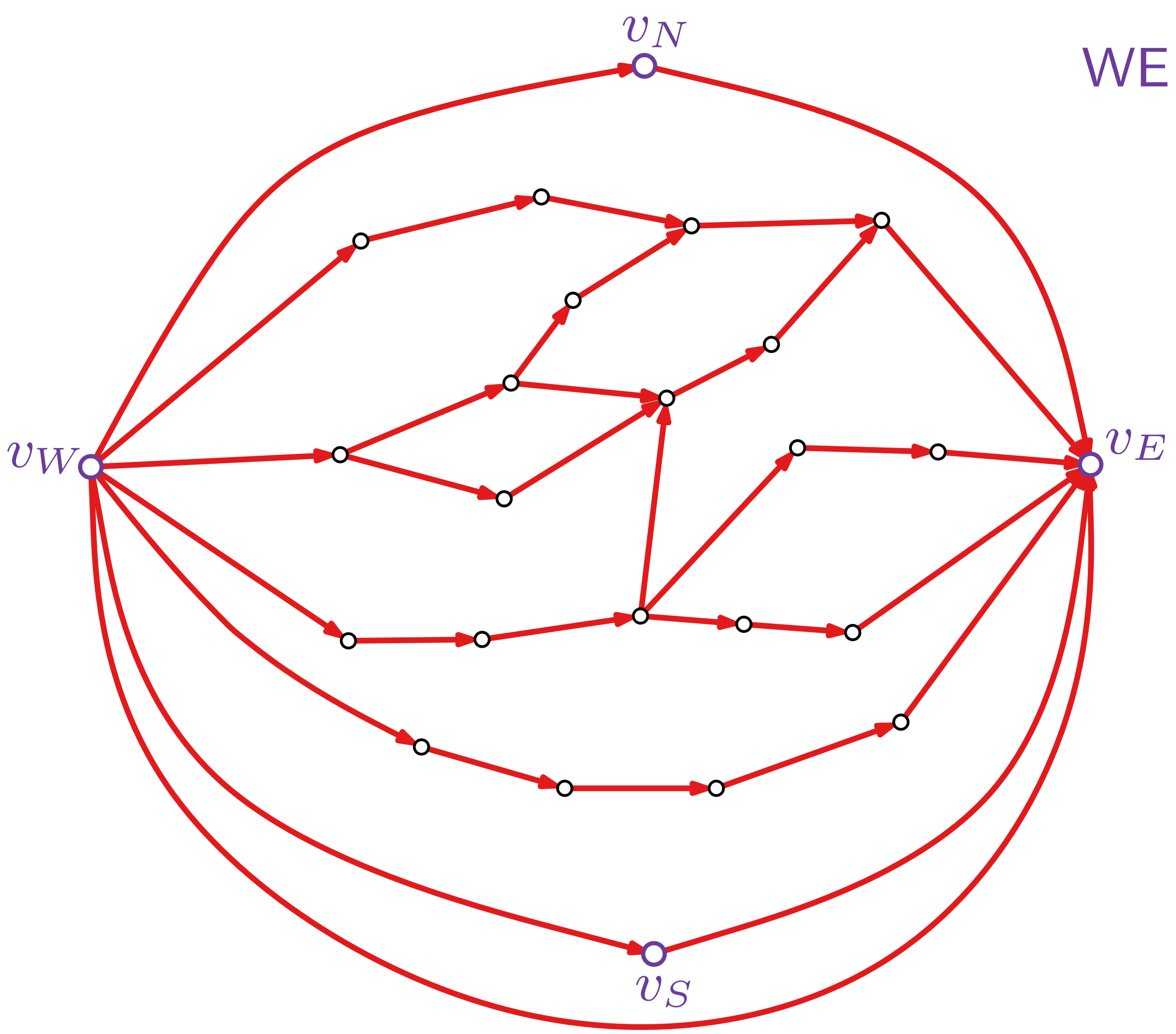


From REL to st-Digraphs to Coordinates



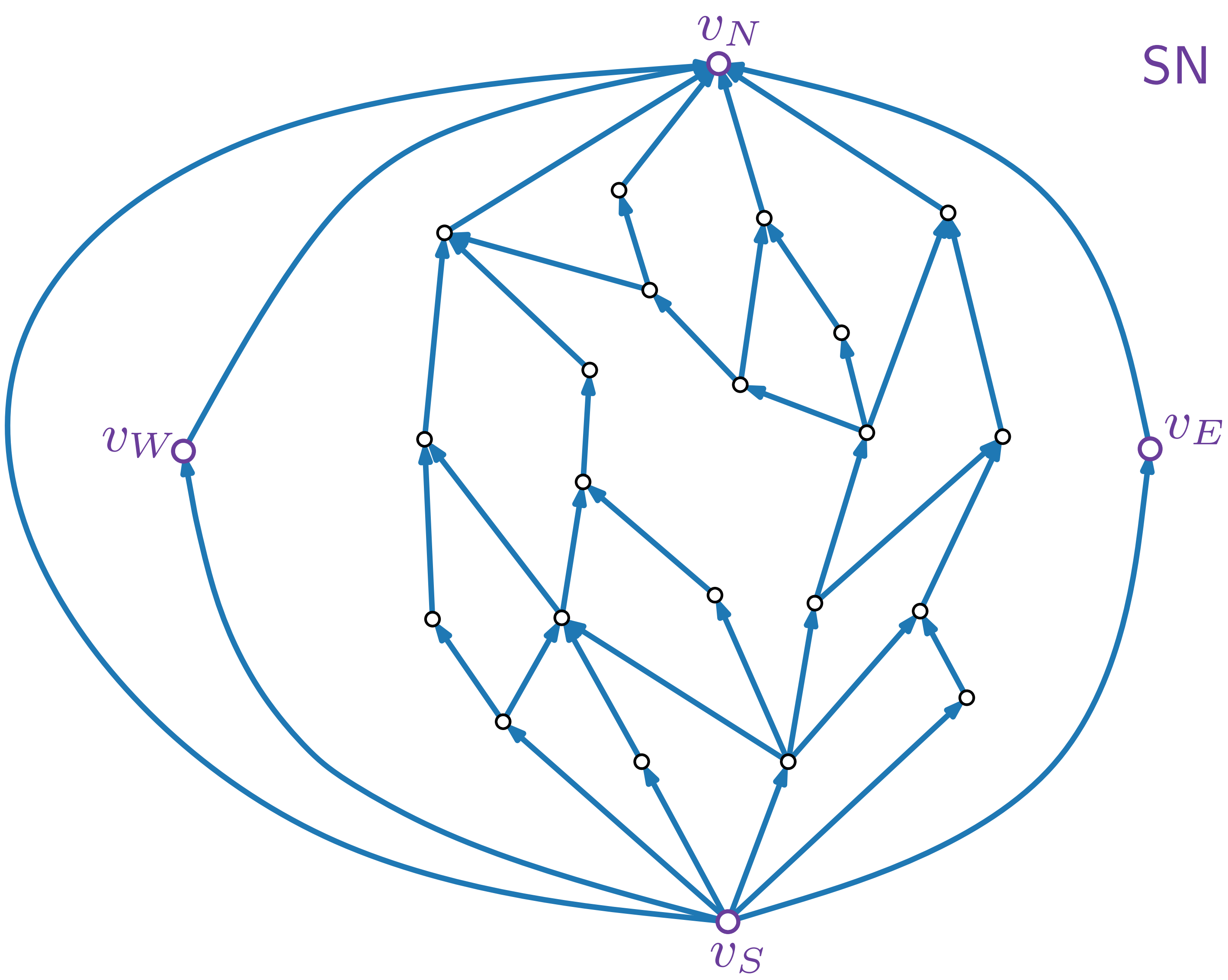
WE network G_{hor}

From REL to st-Digraphs to Coordinates



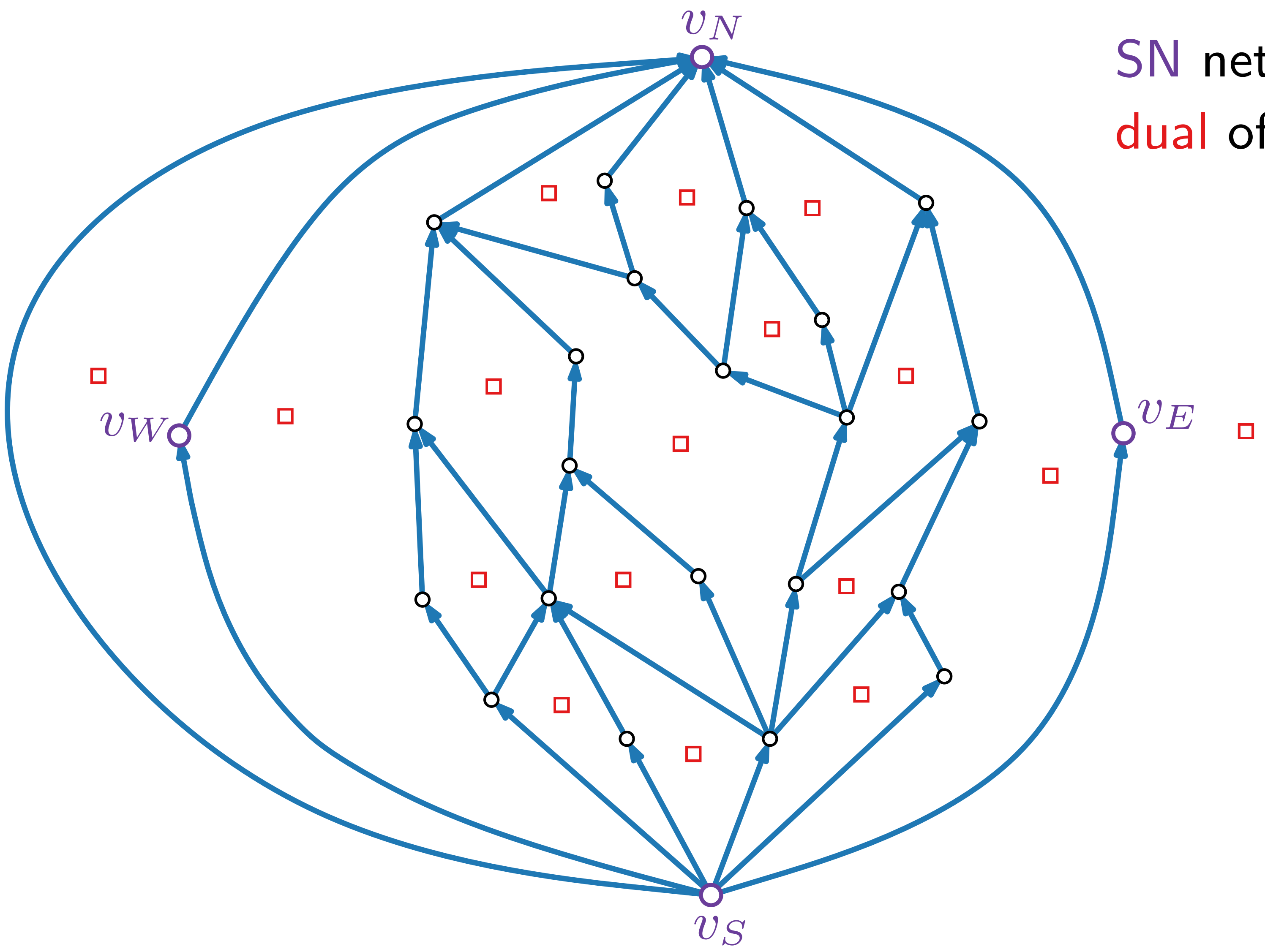
WE network G_{hor}

From REL to st-Digraphs to Coordinates



SN network G_{ver}

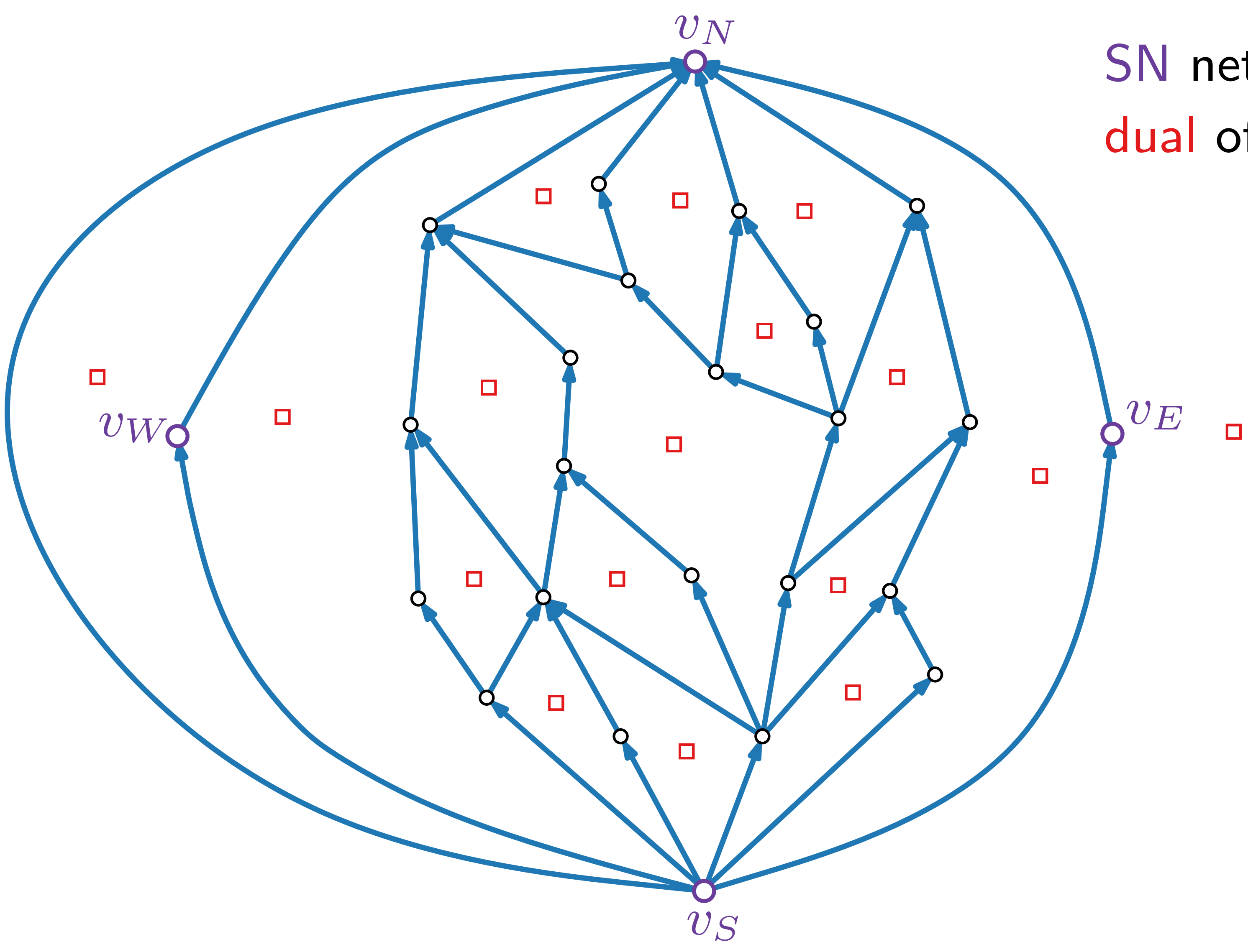
From REL to st-Digraphs to Coordinates



SN network G_{ver}

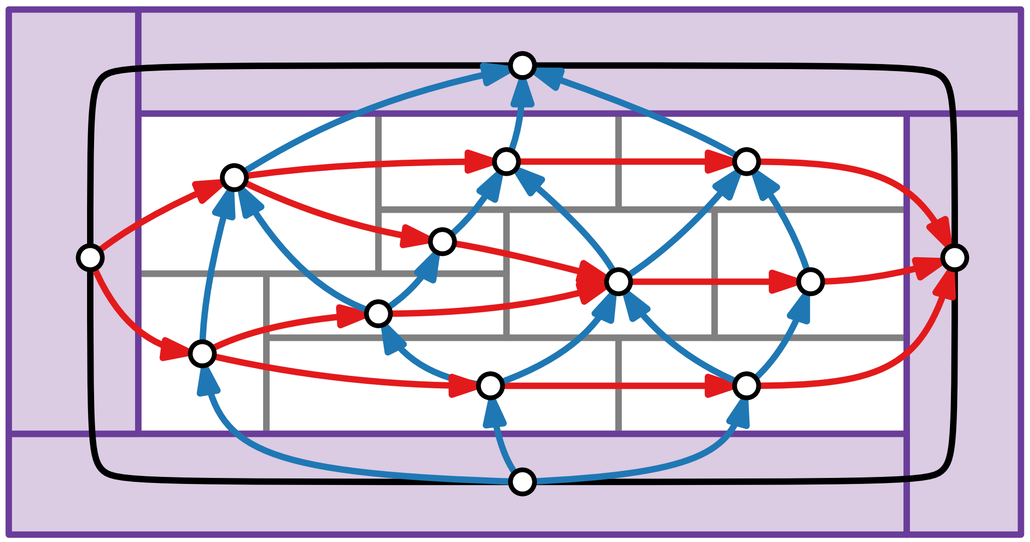
dual of G_{ver}

From REL to st-Digraphs to Coordinates

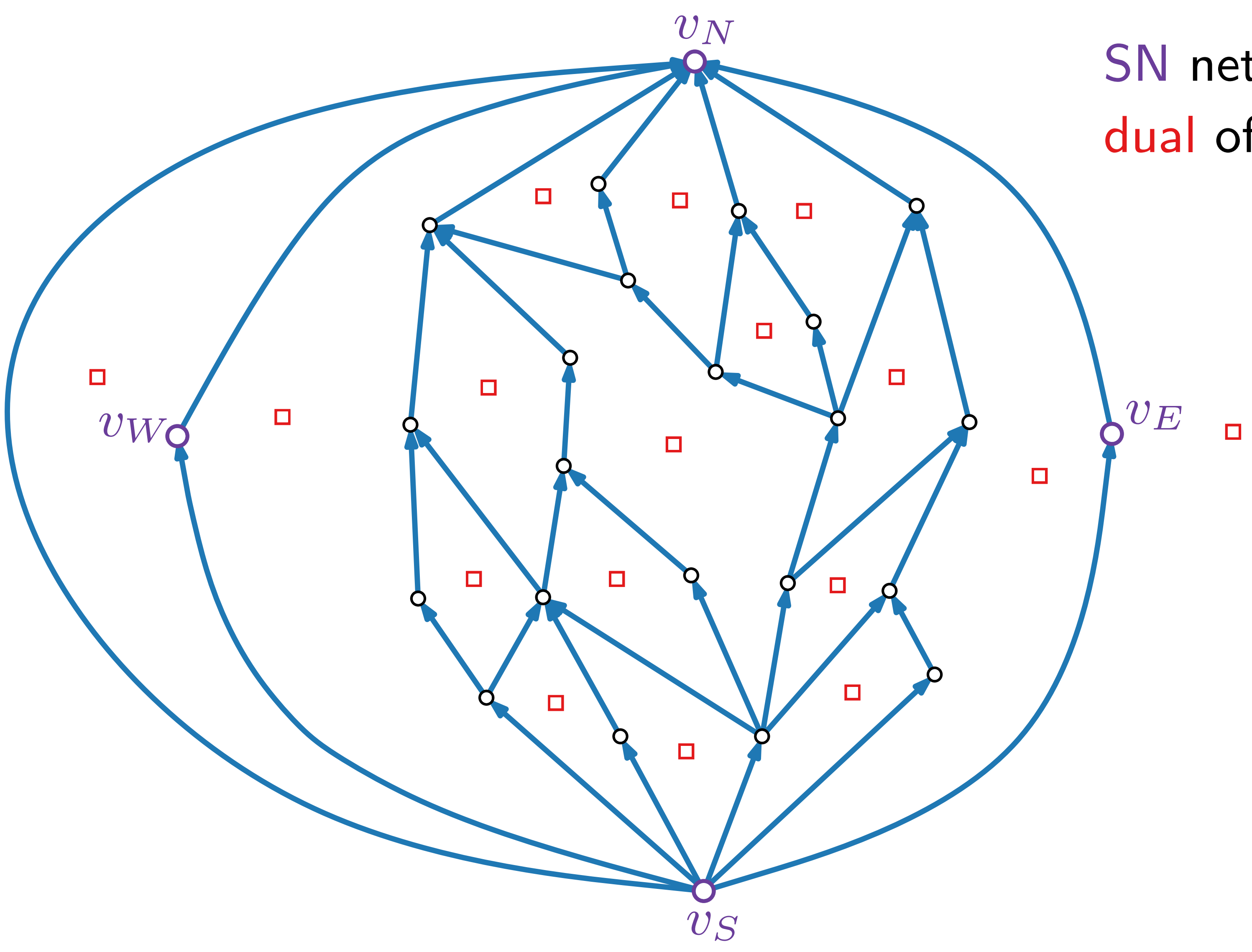


SN network G_{ver}

dual of G_{ver}

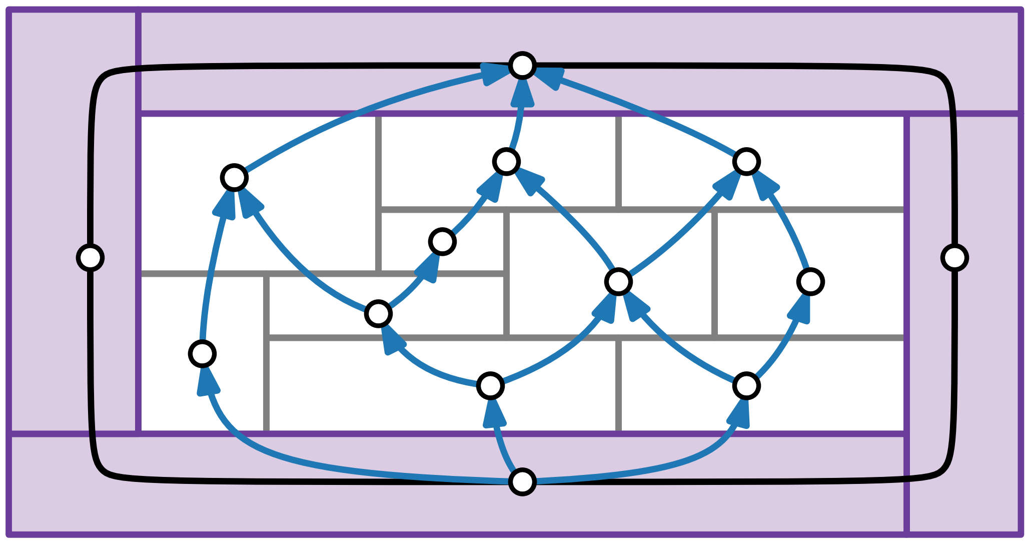


From REL to st-Digraphs to Coordinates

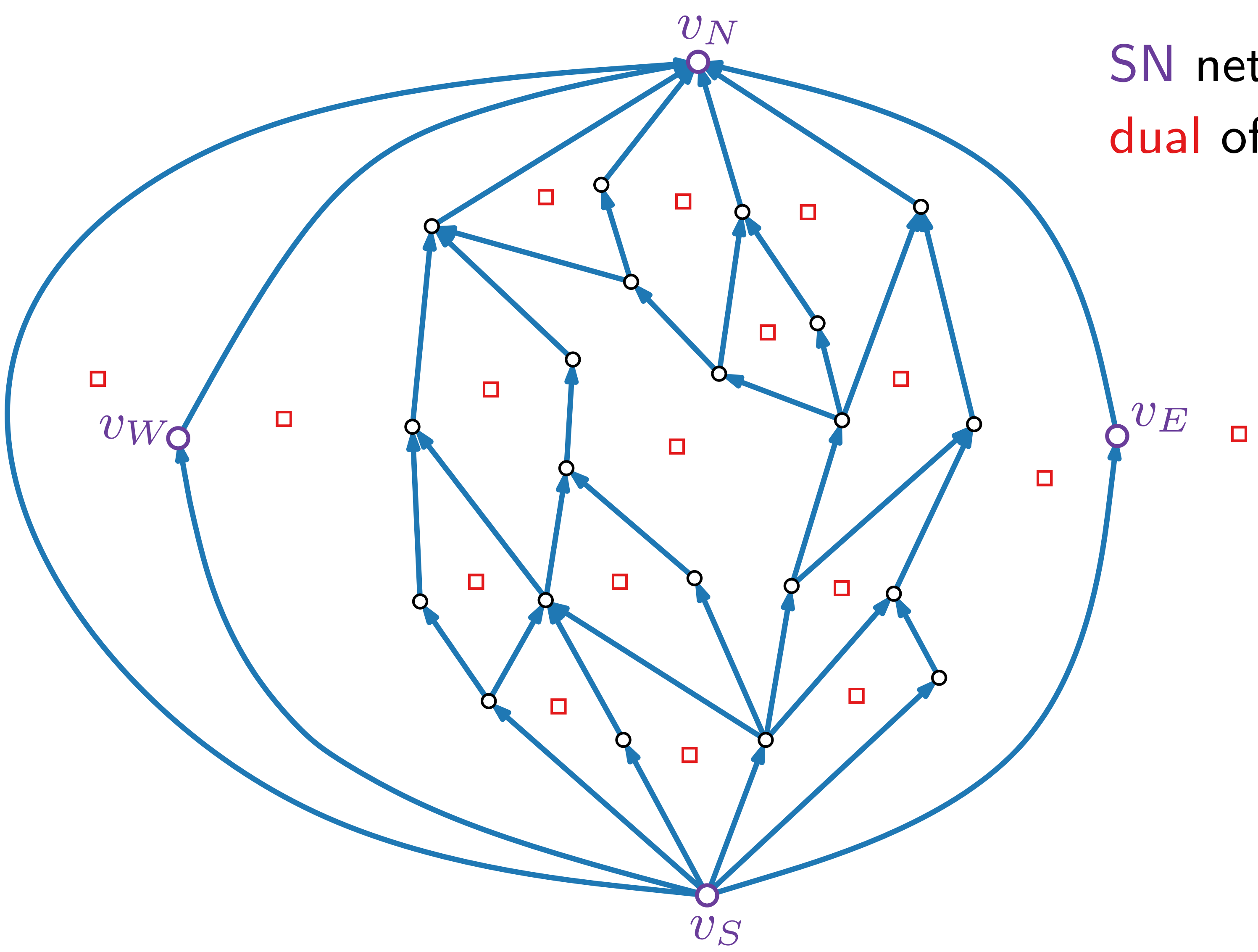


SN network G_{ver}

dual of G_{ver}

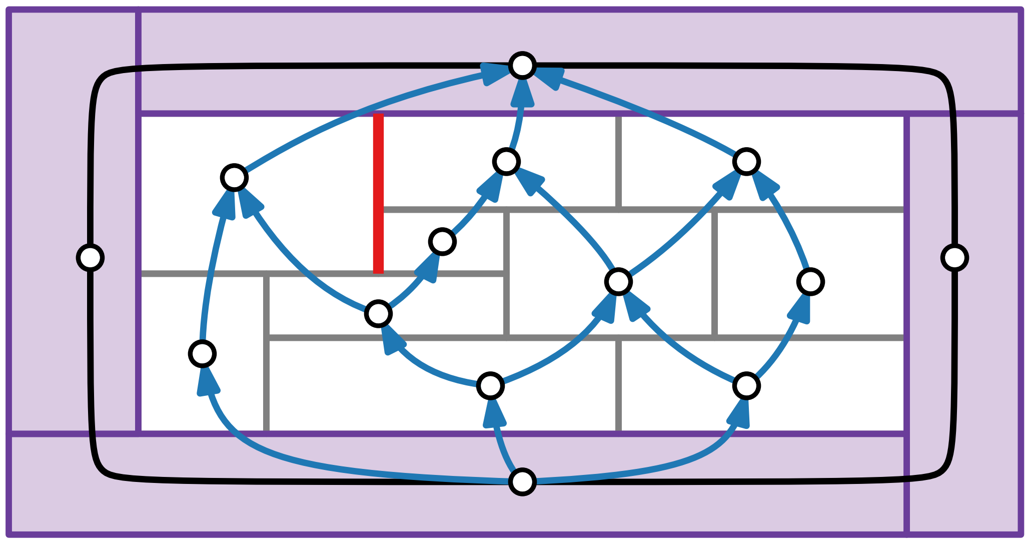


From REL to st-Digraphs to Coordinates

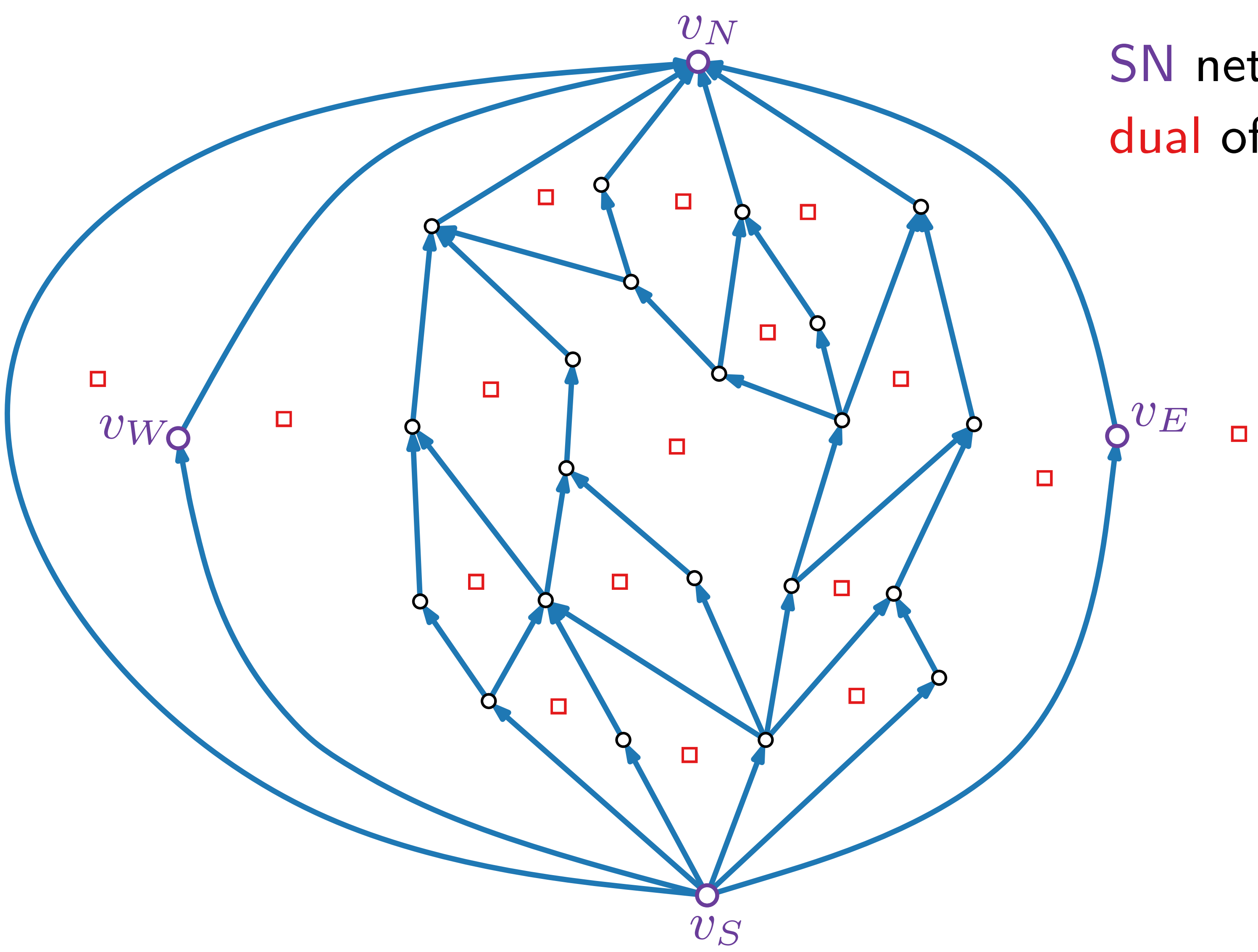


SN network G_{ver}

dual of G_{ver}

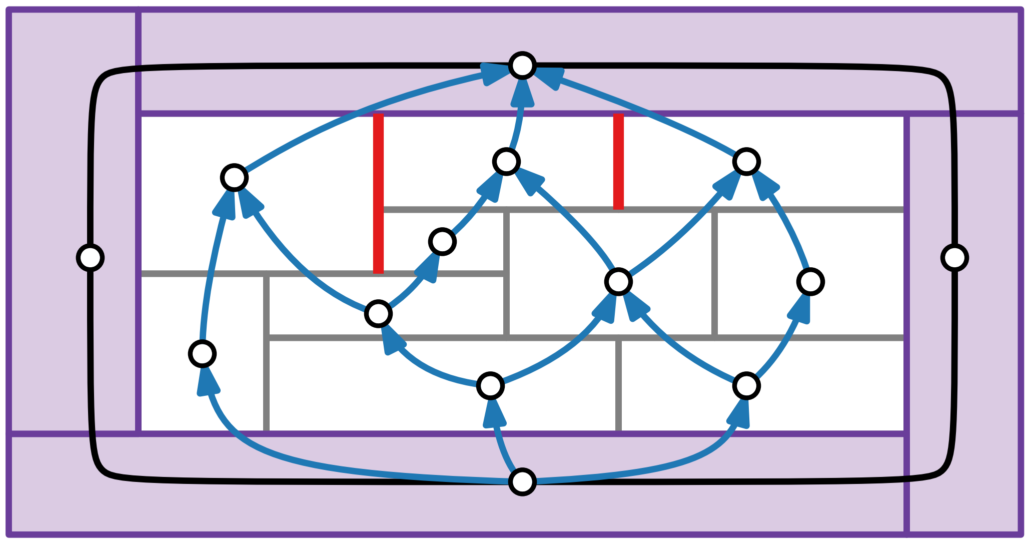


From REL to st-Digraphs to Coordinates

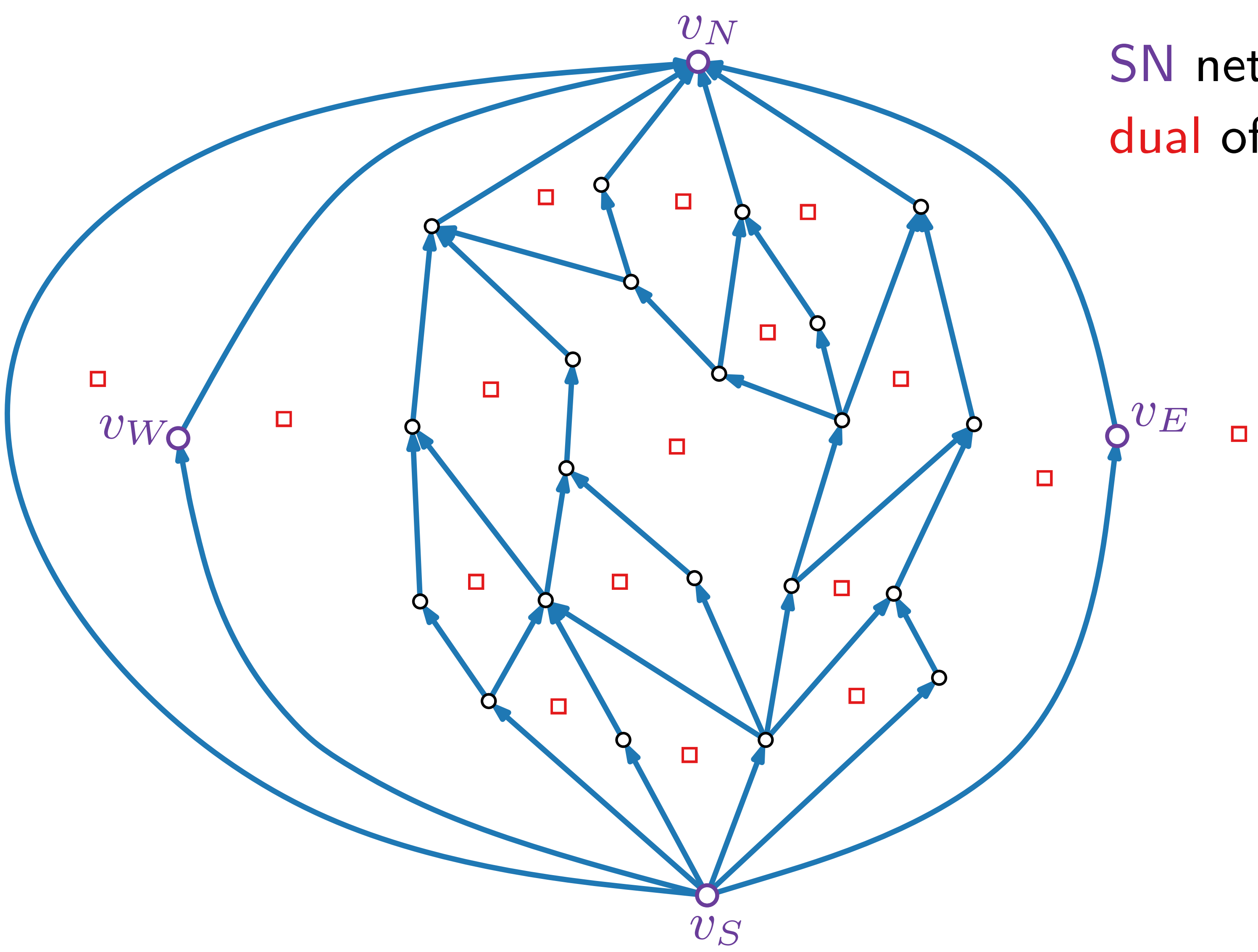


SN network G_{ver}

dual of G_{ver}

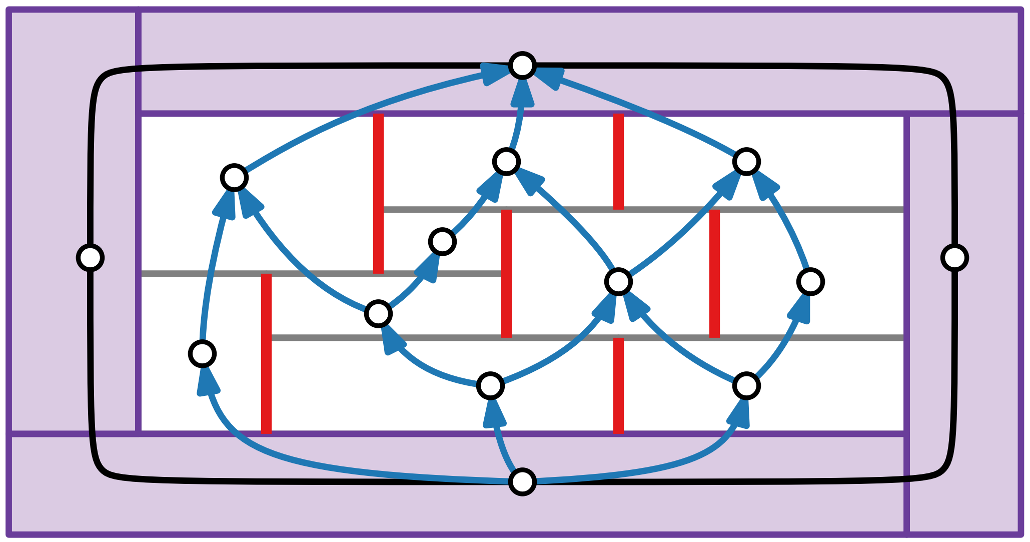


From REL to st-Digraphs to Coordinates

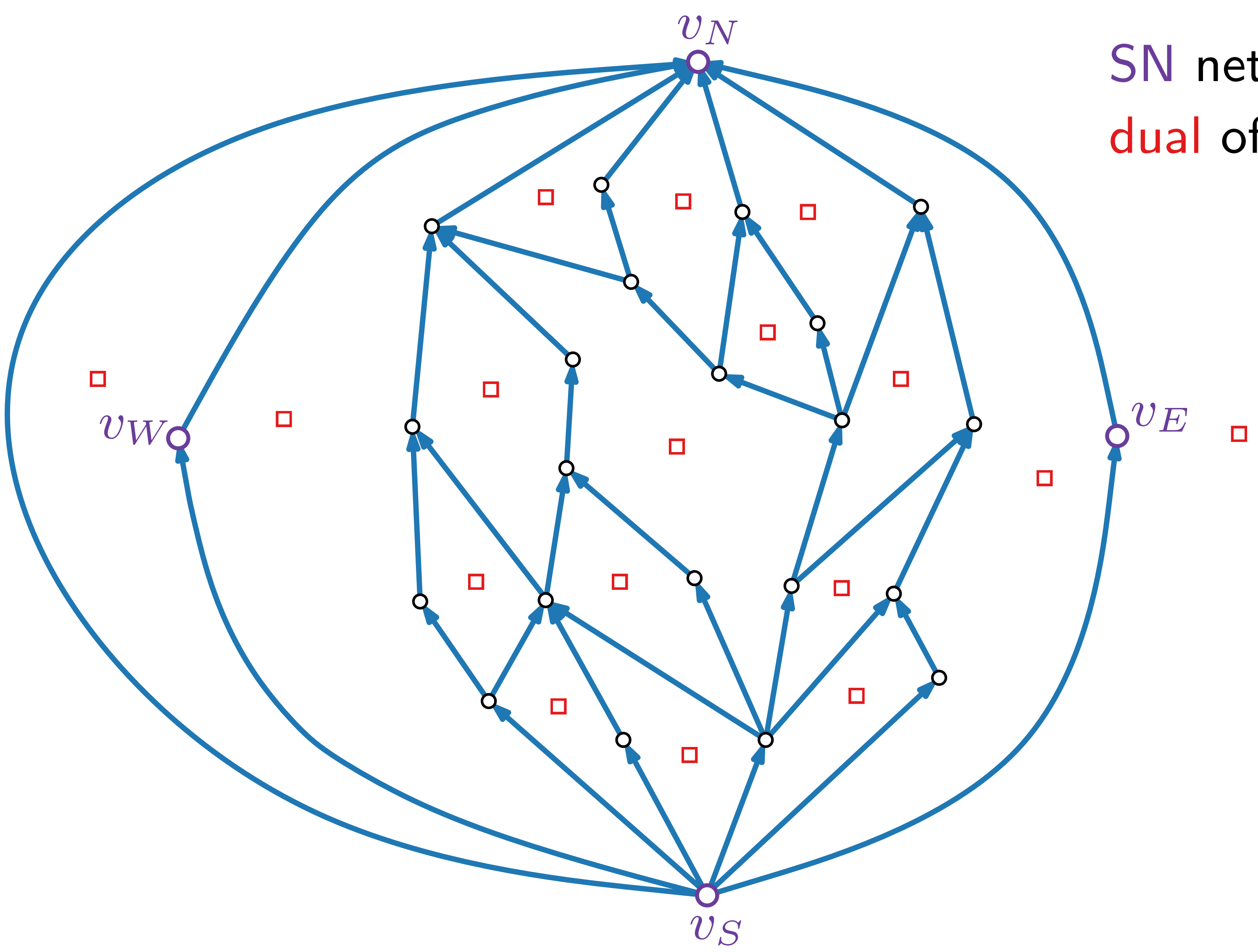


SN network G_{ver}

dual of G_{ver}

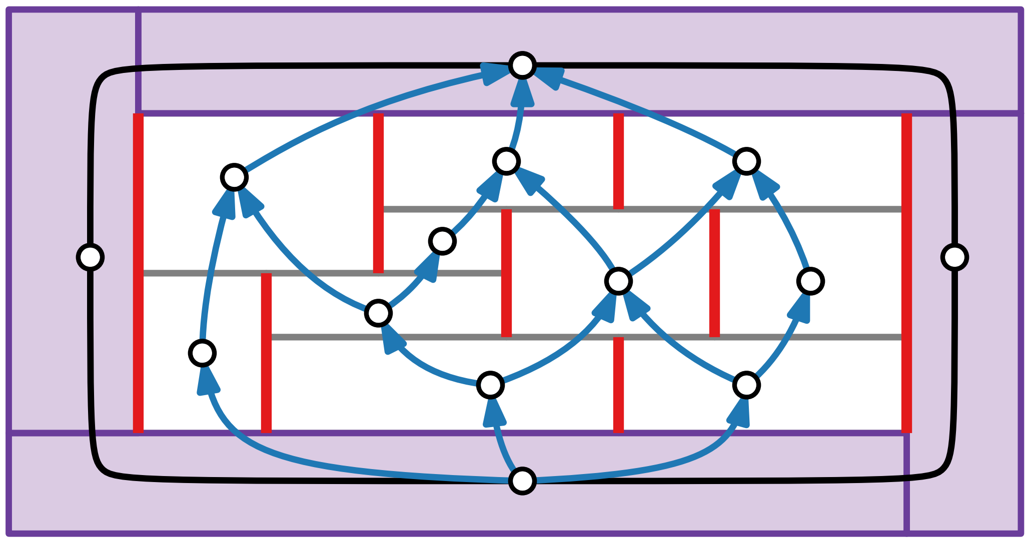


From REL to st-Digraphs to Coordinates

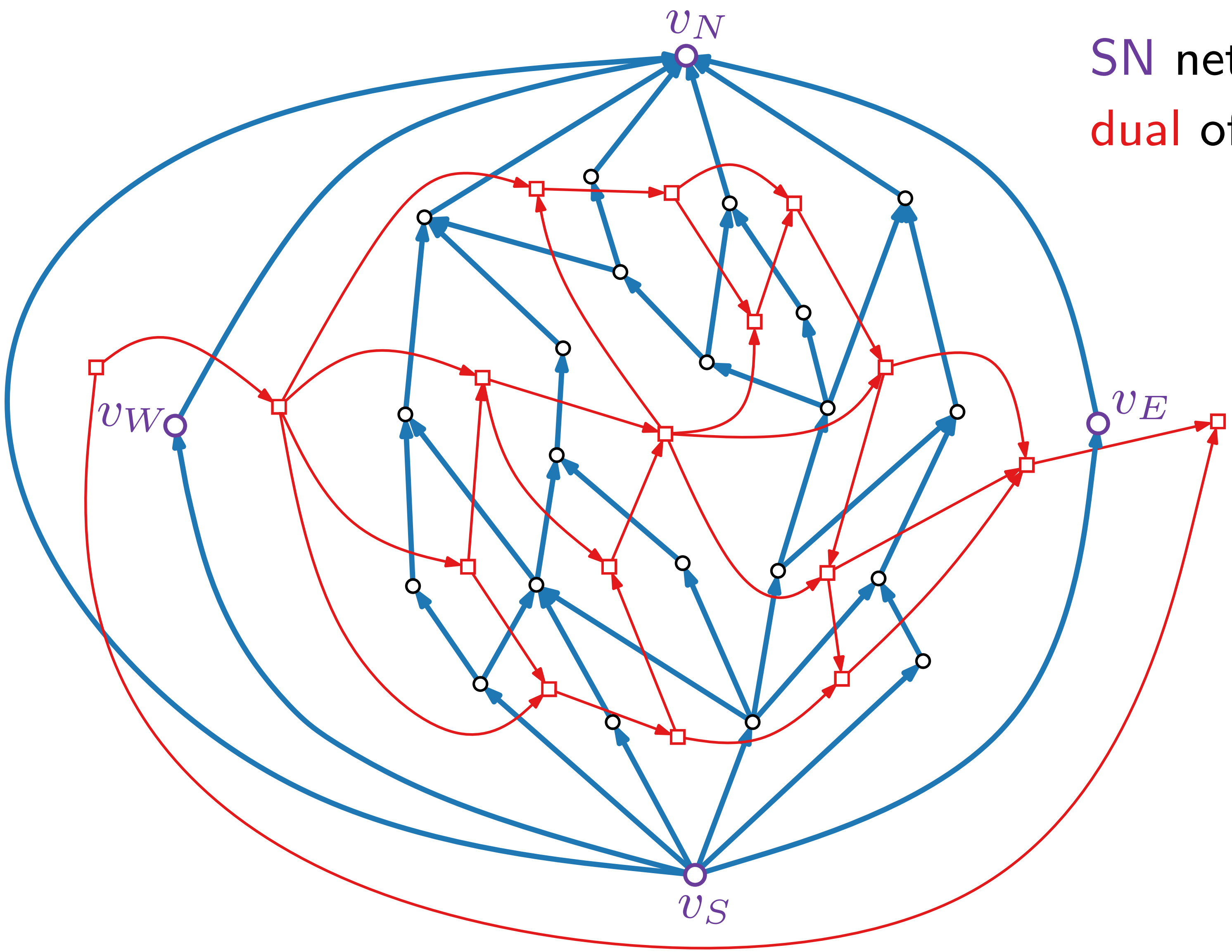


SN network G_{ver}

dual of G_{ver}



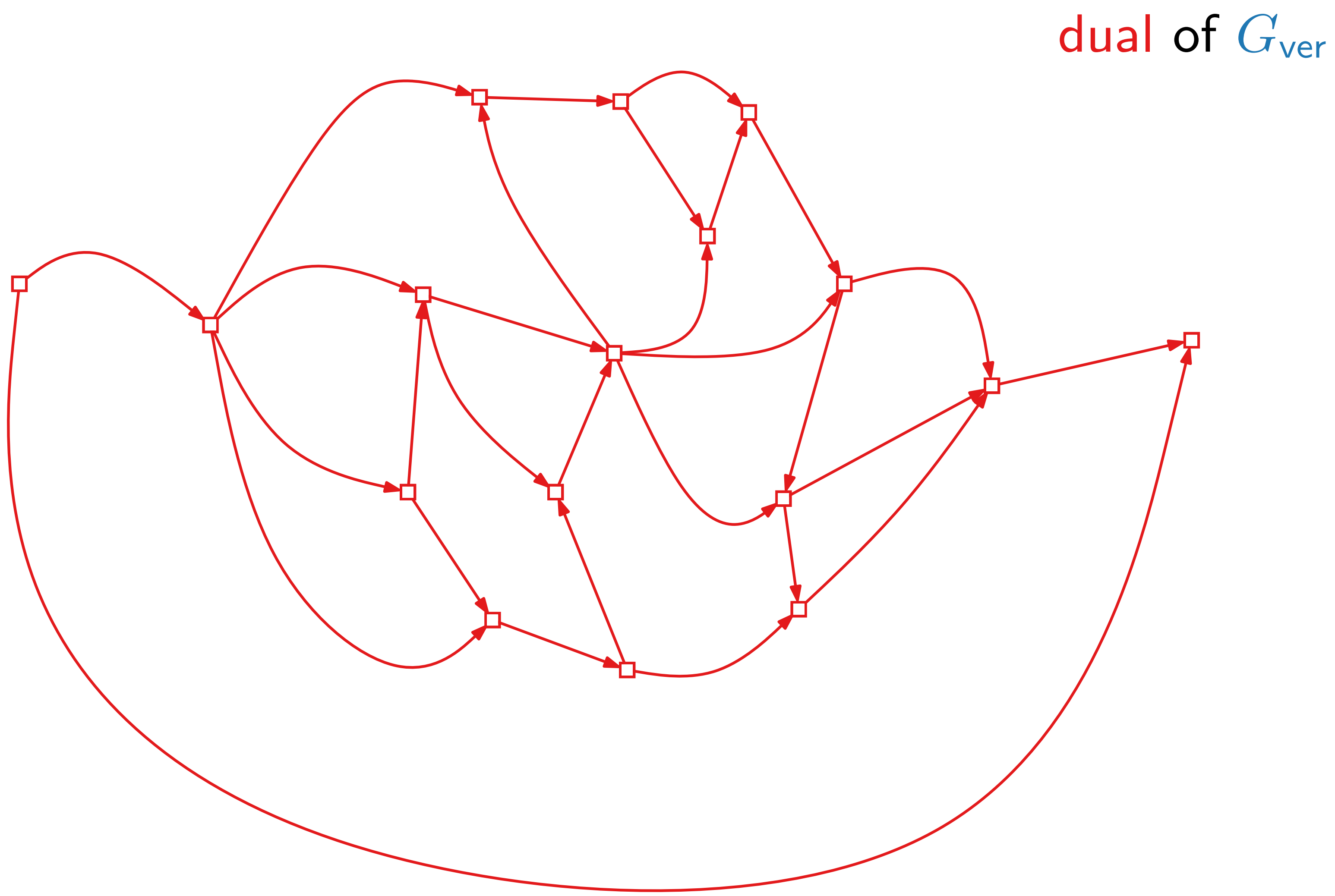
From REL to st-Digraphs to Coordinates



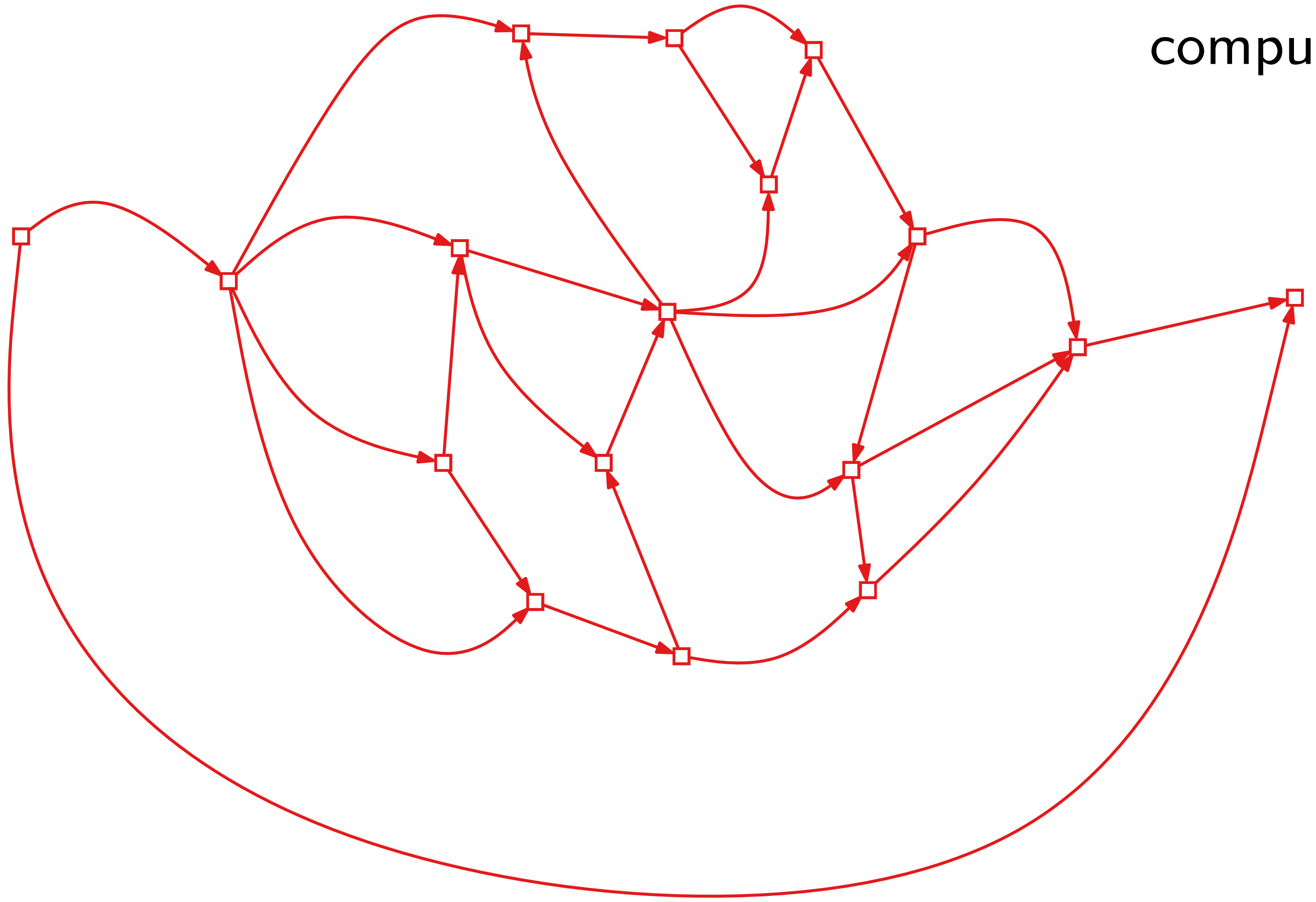
SN network G_{ver}

dual of G_{ver}

From REL to st-Digraphs to Coordinates



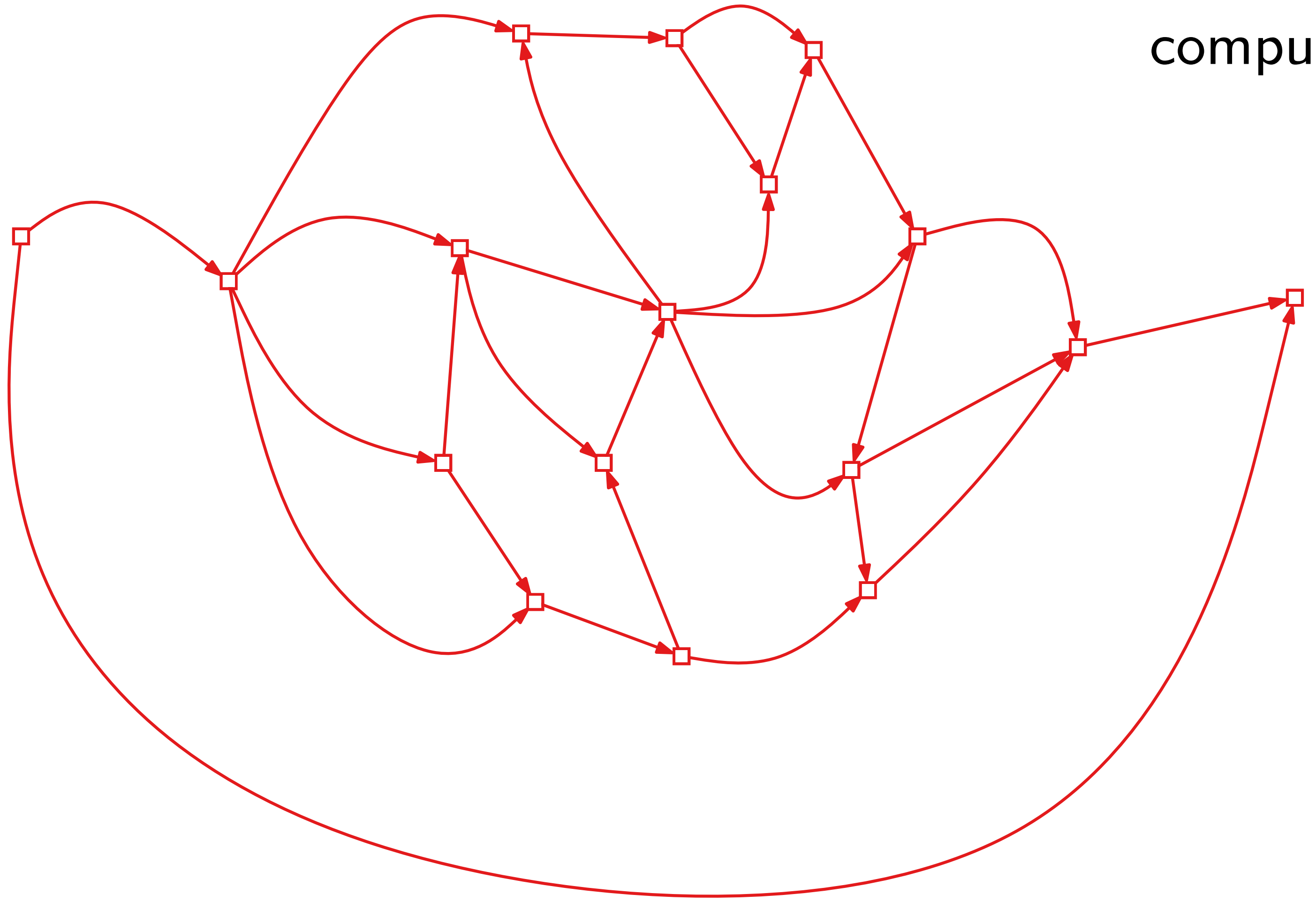
From REL to st-Digraphs to Coordinates



dual of G_{ver}

compute topological order

From REL to st-Digraphs to Coordinates

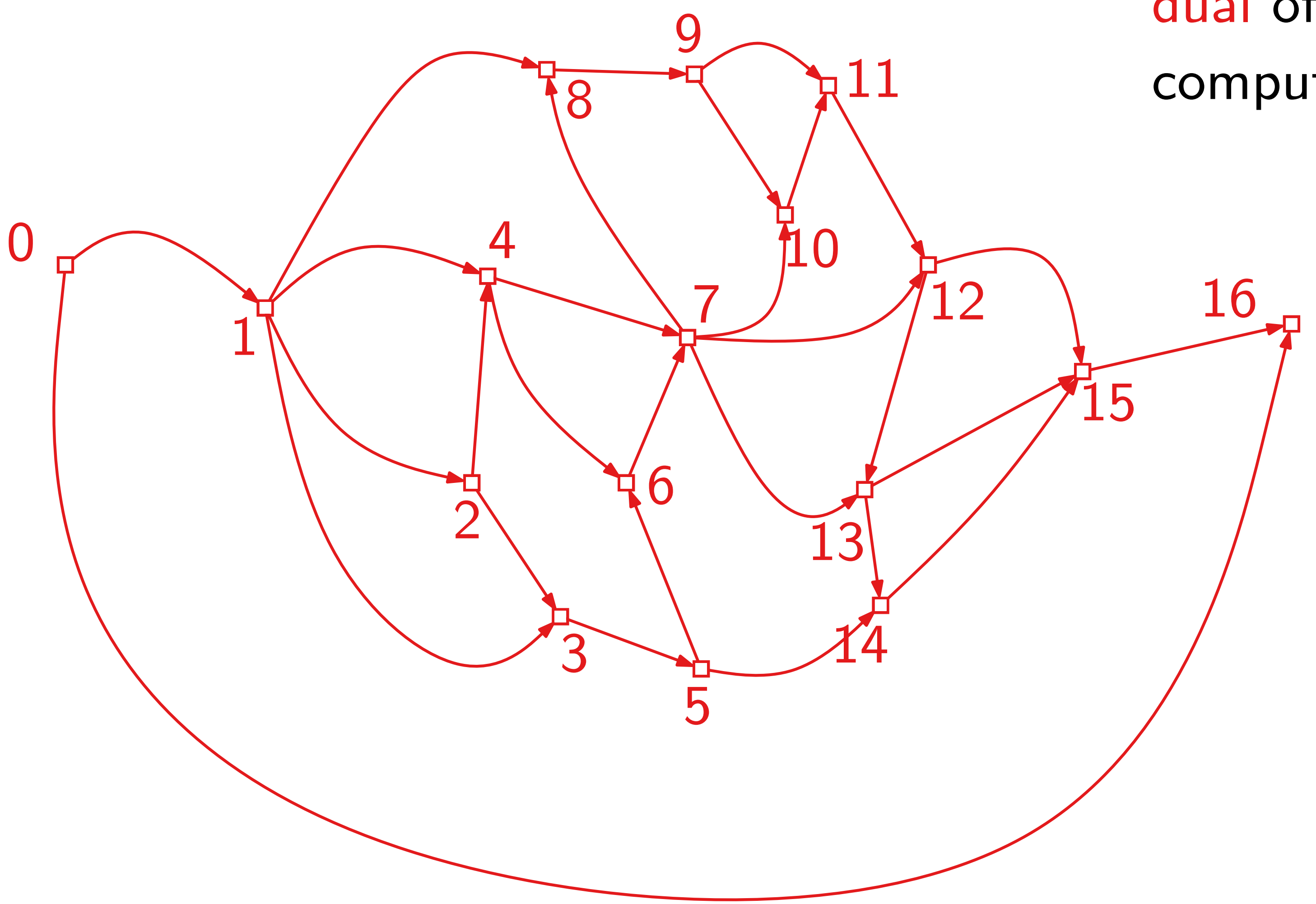


dual of G_{ver}

compute topological order

i.e., linear order of all vertices such that all directed edges point in the same direction

From REL to st-Digraphs to Coordinates

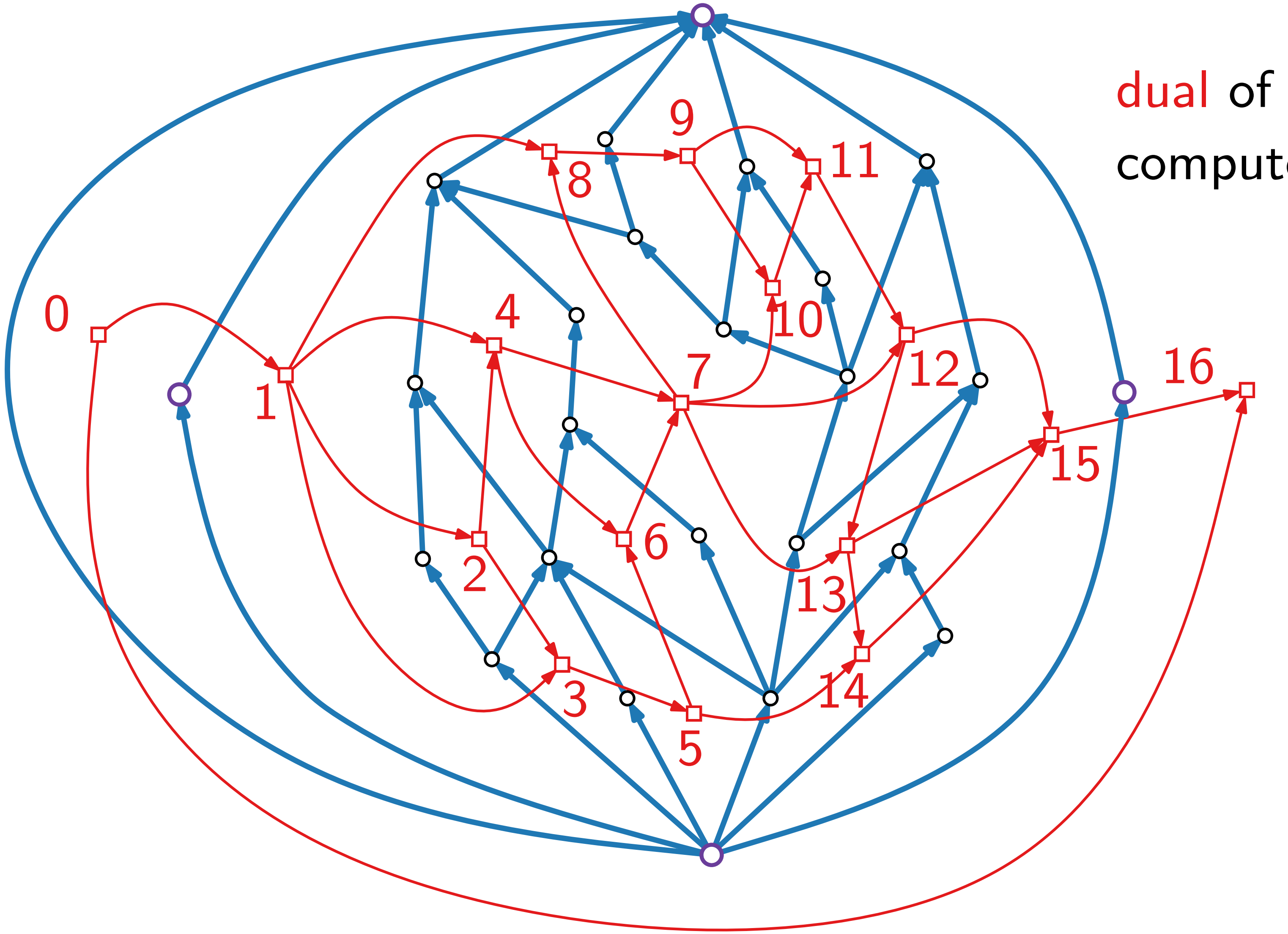


dual of G_{ver}

compute topological order

i.e., linear order of all vertices such that all directed edges point in the same direction

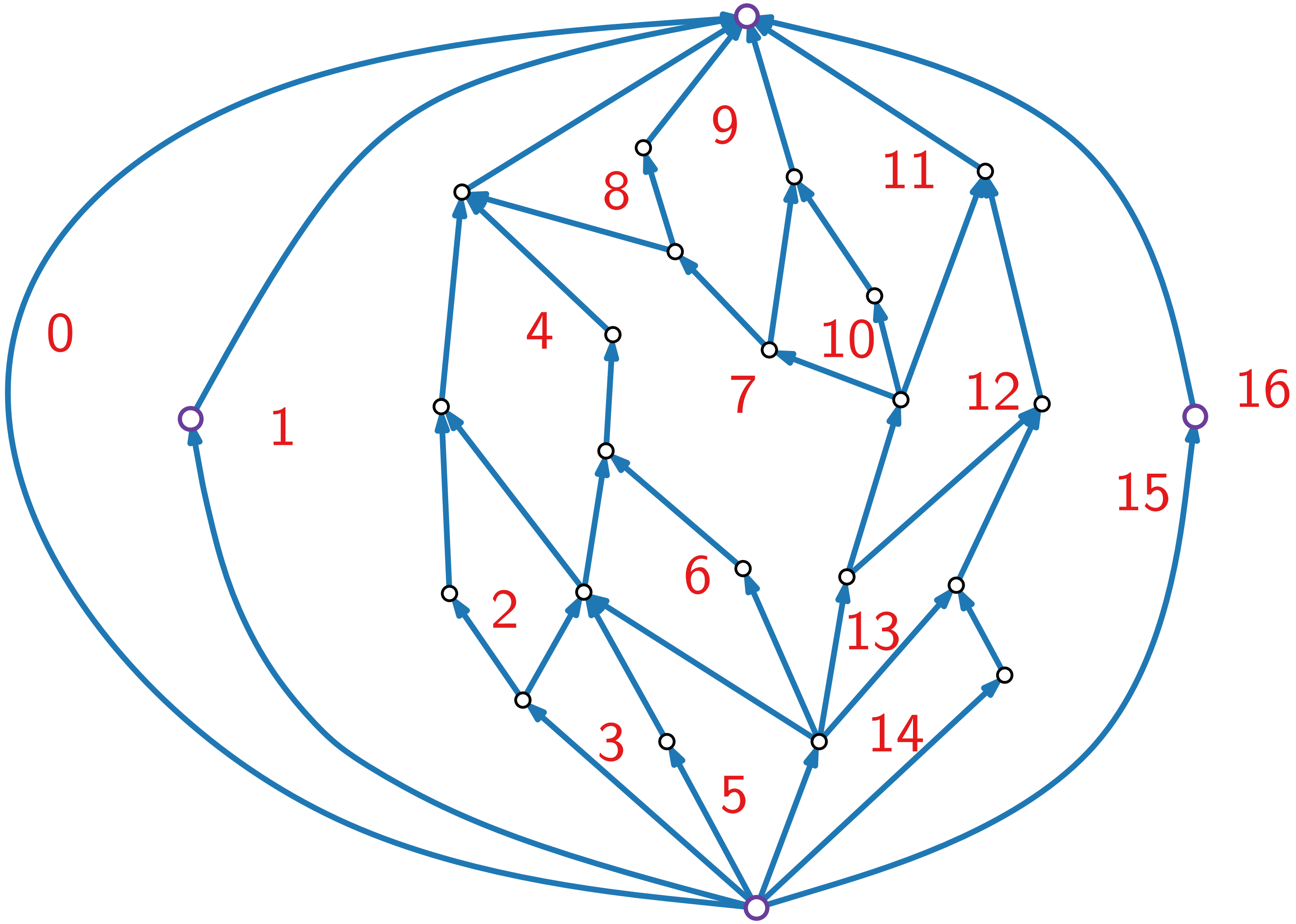
From REL to st-Digraphs to Coordinates



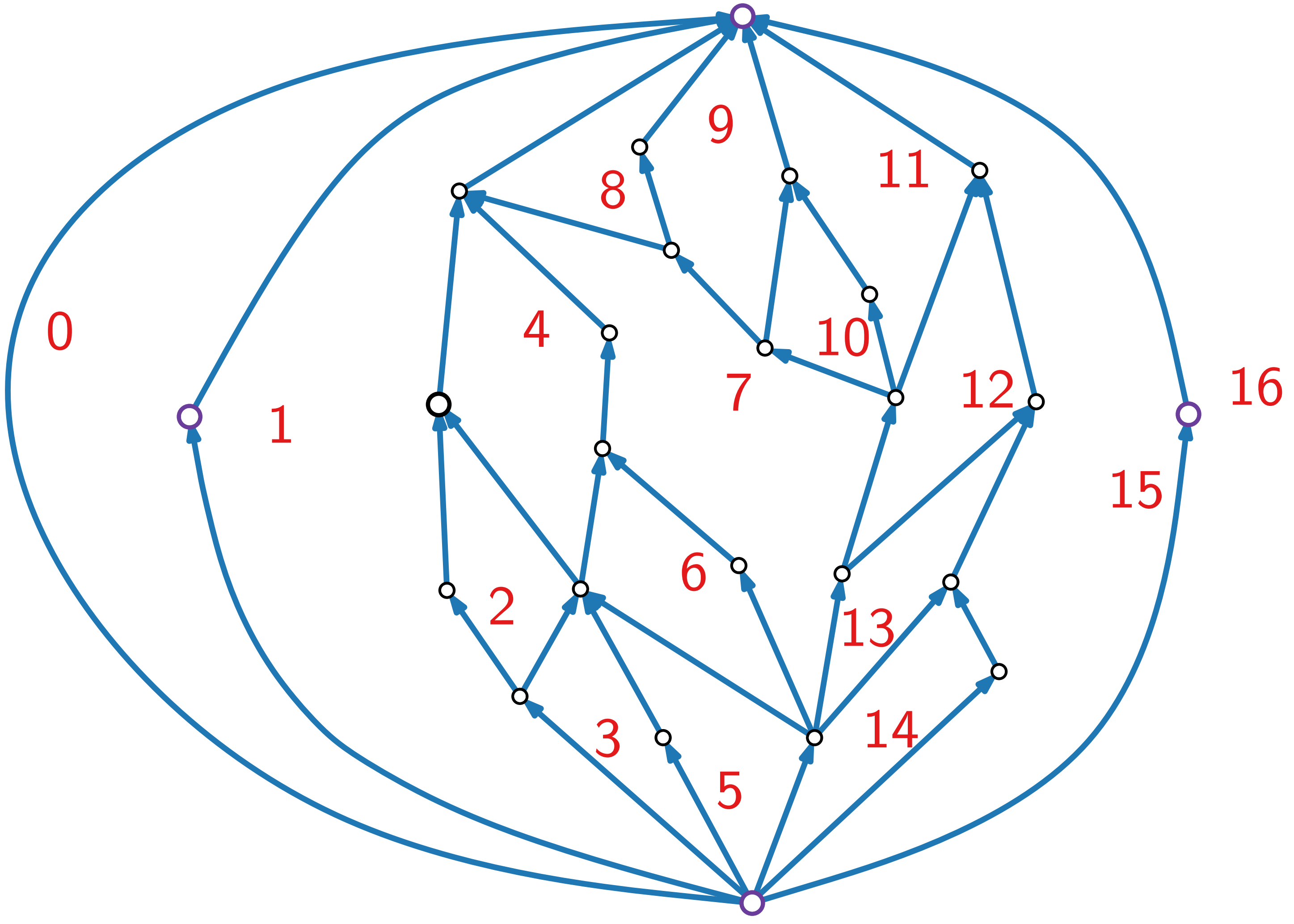
dual of G_{ver}

compute topological order

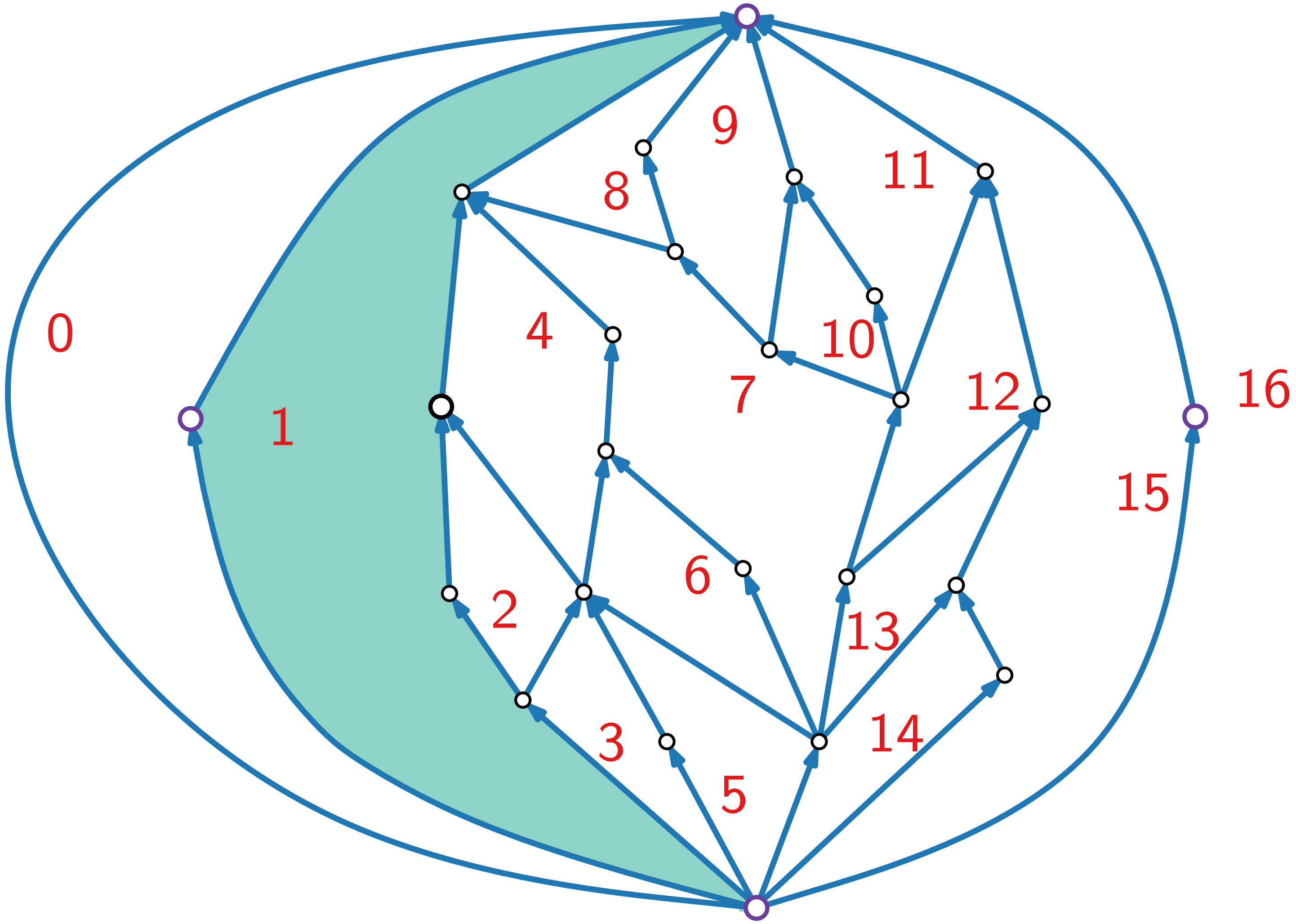
From REL to st-Digraphs to Coordinates



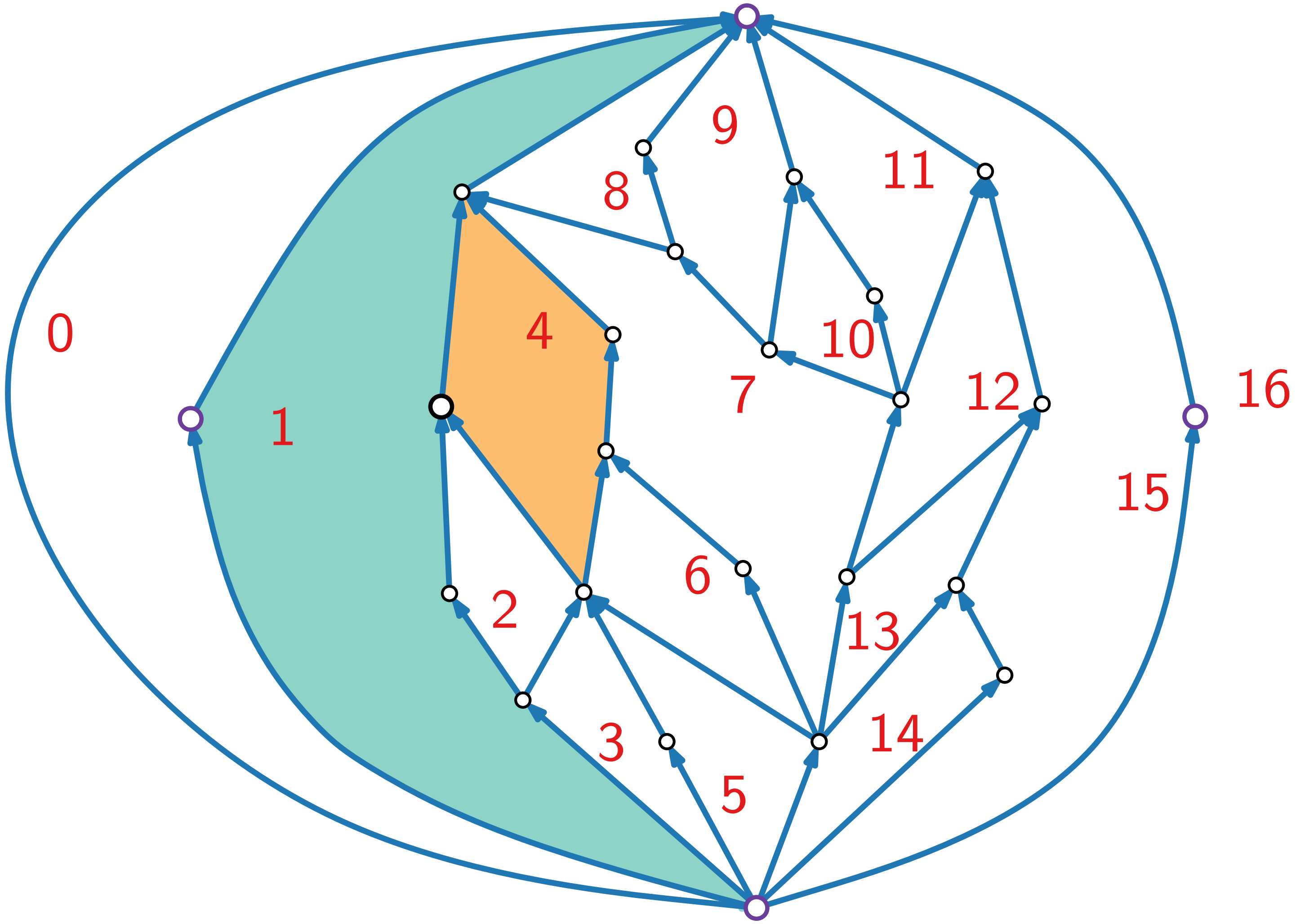
From REL to st-Digraphs to Coordinates



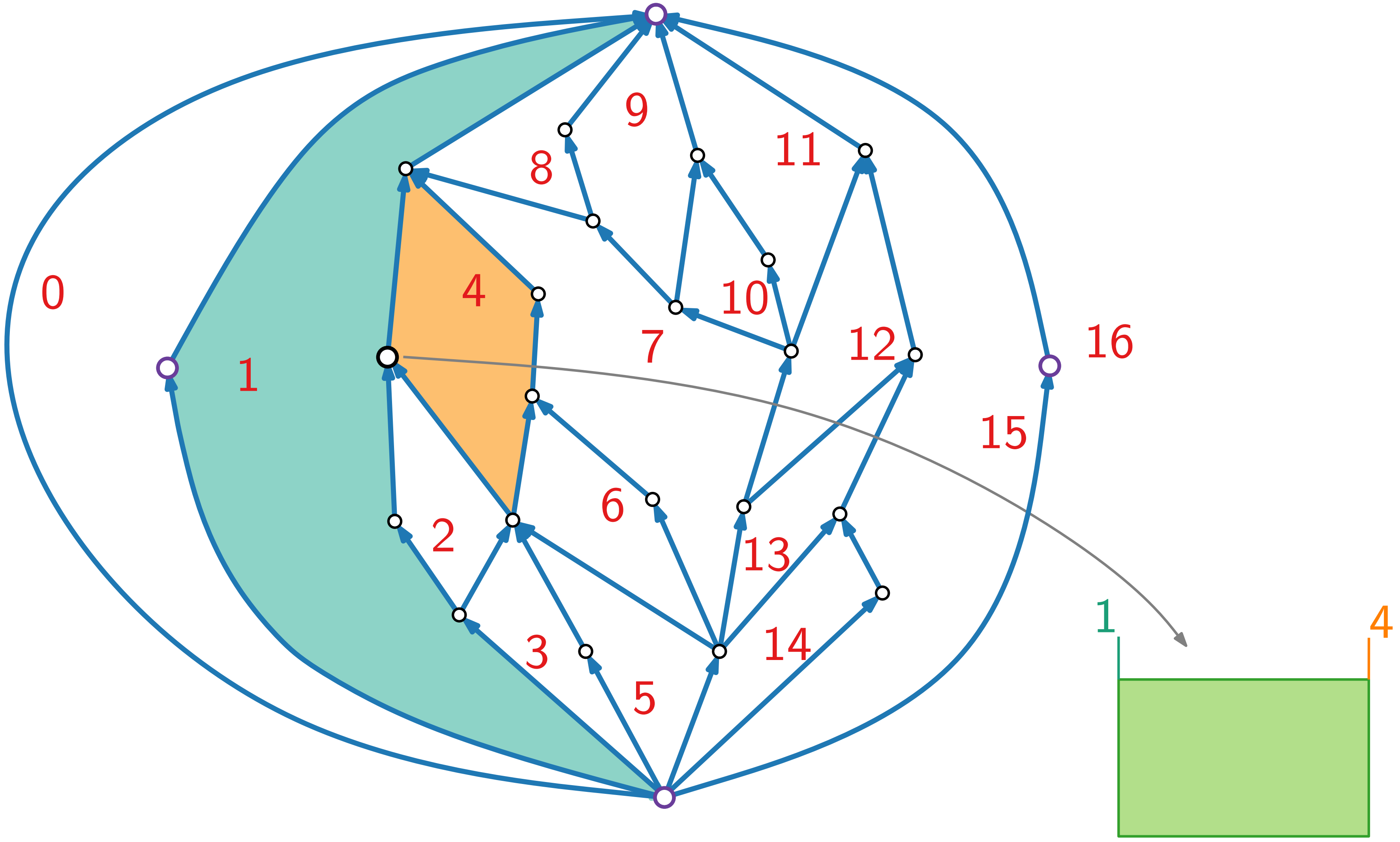
From REL to st-Digraphs to Coordinates



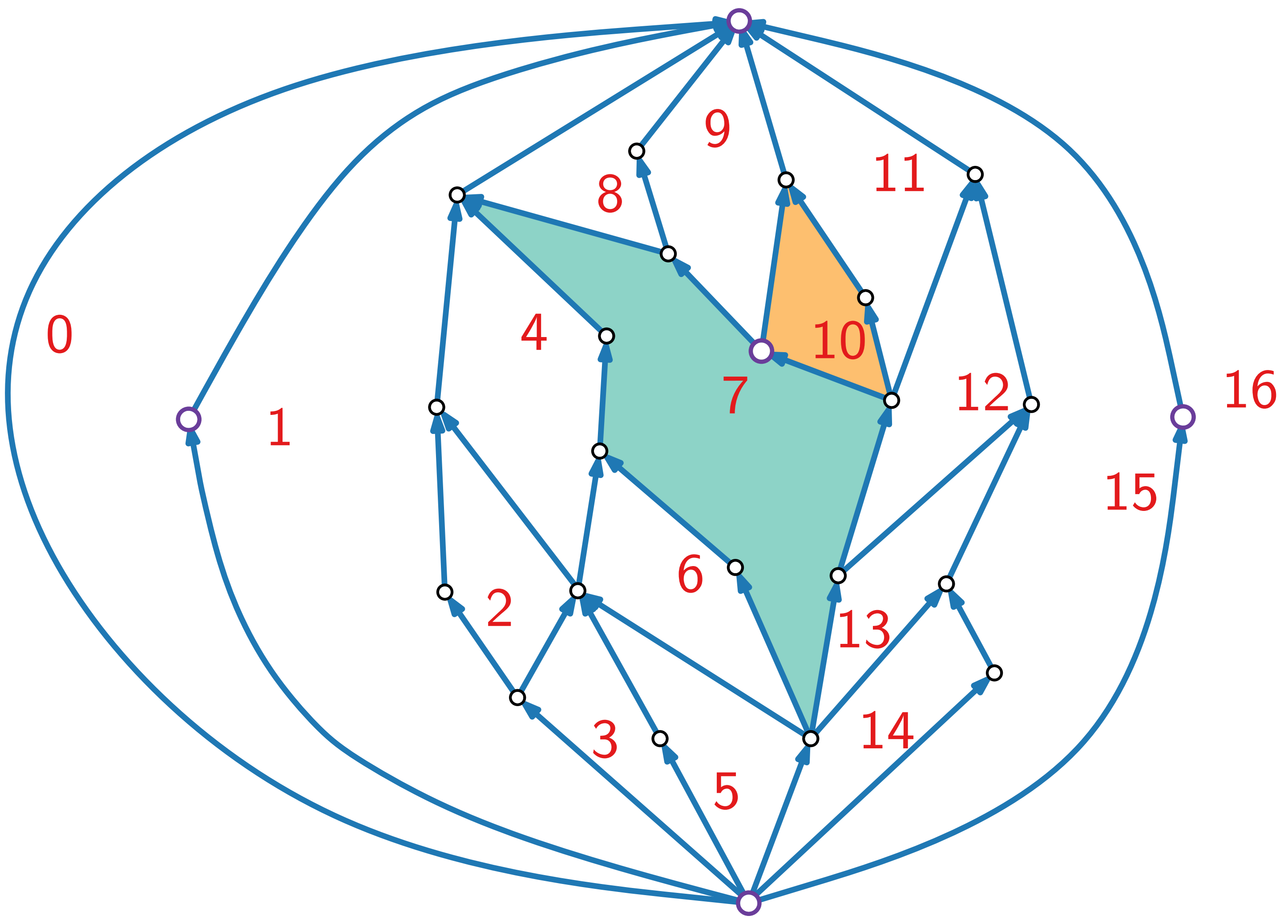
From REL to st-Digraphs to Coordinates



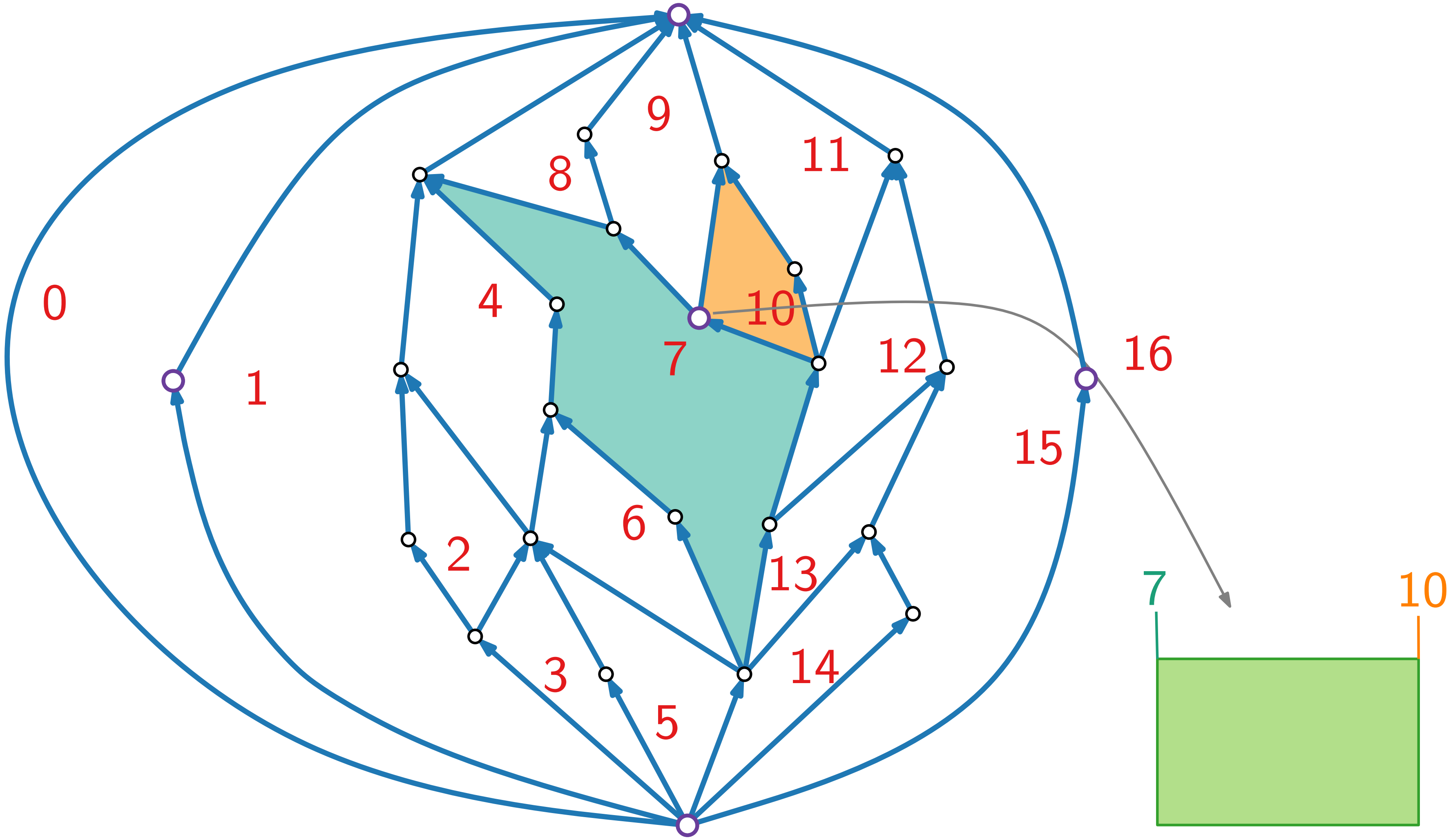
From REL to st-Digraphs to Coordinates



From REL to st-Digraphs to Coordinates



From REL to st-Digraphs to Coordinates



Rectangular Dual Algorithm

For a PTP graph $G = (V, E)$:

Rectangular Dual Algorithm

For a PTP graph $G = (V, E)$:

- Find a REL $\{T_r, T_b\}$ of G;

Rectangular Dual Algorithm

For a PTP graph $G = (V, E)$:

- Find a REL $\{T_r, T_b\}$ of G;
- Construct a SN network G_{ver} of G (consists of T_b plus outer edges)

Rectangular Dual Algorithm

For a PTP graph $G = (V, E)$:

- Find a REL $\{T_r, T_b\}$ of G;
- Construct a SN network G_{ver} of G (consists of T_b plus outer edges)
- Construct the dual $G^\star_{\text{ver}}$ of G_{ver} and compute a topological ordering f_{ver} of $G^\star_{\text{ver}}$

Rectangular Dual Algorithm

For a PTP graph $G = (V, E)$:

- Find a REL $\{T_r, T_b\}$ of G;
- Construct a SN network G_{ver} of G (consists of T_b plus outer edges)
- Construct the dual $G^\star_{\text{ver}}$ of G_{ver} and compute a topological ordering f_{ver} of $G^\star_{\text{ver}}$
- For each vertex $v \in V$, let g and h be the face on the left and face on the right of v.

Rectangular Dual Algorithm

For a PTP graph $G = (V, E)$:

- Find a REL $\{T_r, T_b\}$ of G;
- Construct a SN network G_{ver} of G (consists of T_b plus outer edges)
- Construct the dual $G^\star_{\text{ver}}$ of G_{ver} and compute a topological ordering f_{ver} of $G^\star_{\text{ver}}$
- For each vertex $v \in V$, let g and h be the face on the left and face on the right of v. Set $x_1(v) = f_{\text{ver}}(g)$ and $x_2(v) = f_{\text{ver}}(h)$.

Rectangular Dual Algorithm

For a PTP graph $G = (V, E)$:

- Find a REL $\{T_r, T_b\}$ of G;
- Construct a SN network G_{ver} of G (consists of T_b plus outer edges)
- Construct the dual $G^\star_{\text{ver}}$ of G_{ver} and compute a topological ordering f_{ver} of $G^\star_{\text{ver}}$
- For each vertex $v \in V$, let g and h be the face on the left and face on the right of v. Set $x_1(v) = f_{\text{ver}}(g)$ and $x_2(v) = f_{\text{ver}}(h)$.
- Define $x_1(v_N) = 0, x_1(v_S) = 1$ and $x_2(v_N) = \max f_{\text{ver}} - 1, x_2(v_S) = \max f_{\text{ver}}$

Rectangular Dual Algorithm

For a PTP graph $G = (V, E)$:

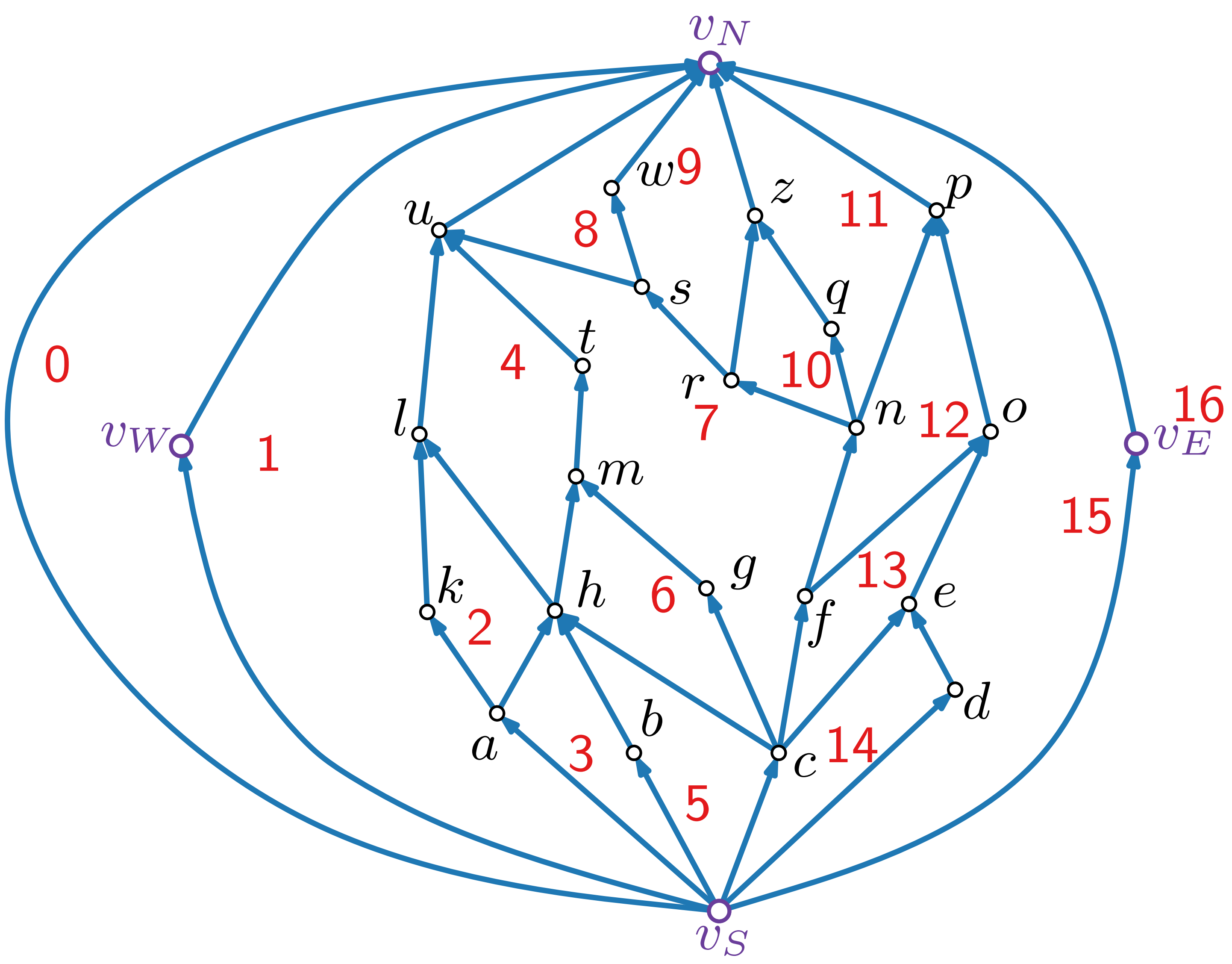
- Find a REL $\{T_r, T_b\}$ of G;
- Construct a SN network G_{ver} of G (consists of T_b plus outer edges)
- Construct the dual $G^\star_{\text{ver}}$ of G_{ver} and compute a topological ordering f_{ver} of $G^\star_{\text{ver}}$
- For each vertex $v \in V$, let g and h be the face on the left and face on the right of v. Set $x_1(v) = f_{\text{ver}}(g)$ and $x_2(v) = f_{\text{ver}}(h)$.
- Define $x_1(v_N) = 0, x_1(v_S) = 1$ and $x_2(v_N) = \max f_{\text{ver}} - 1, x_2(v_S) = \max f_{\text{ver}}$
- Analogously compute y_1 and y_2 with G_{hor}.

Rectangular Dual Algorithm

For a PTP graph $G = (V, E)$:

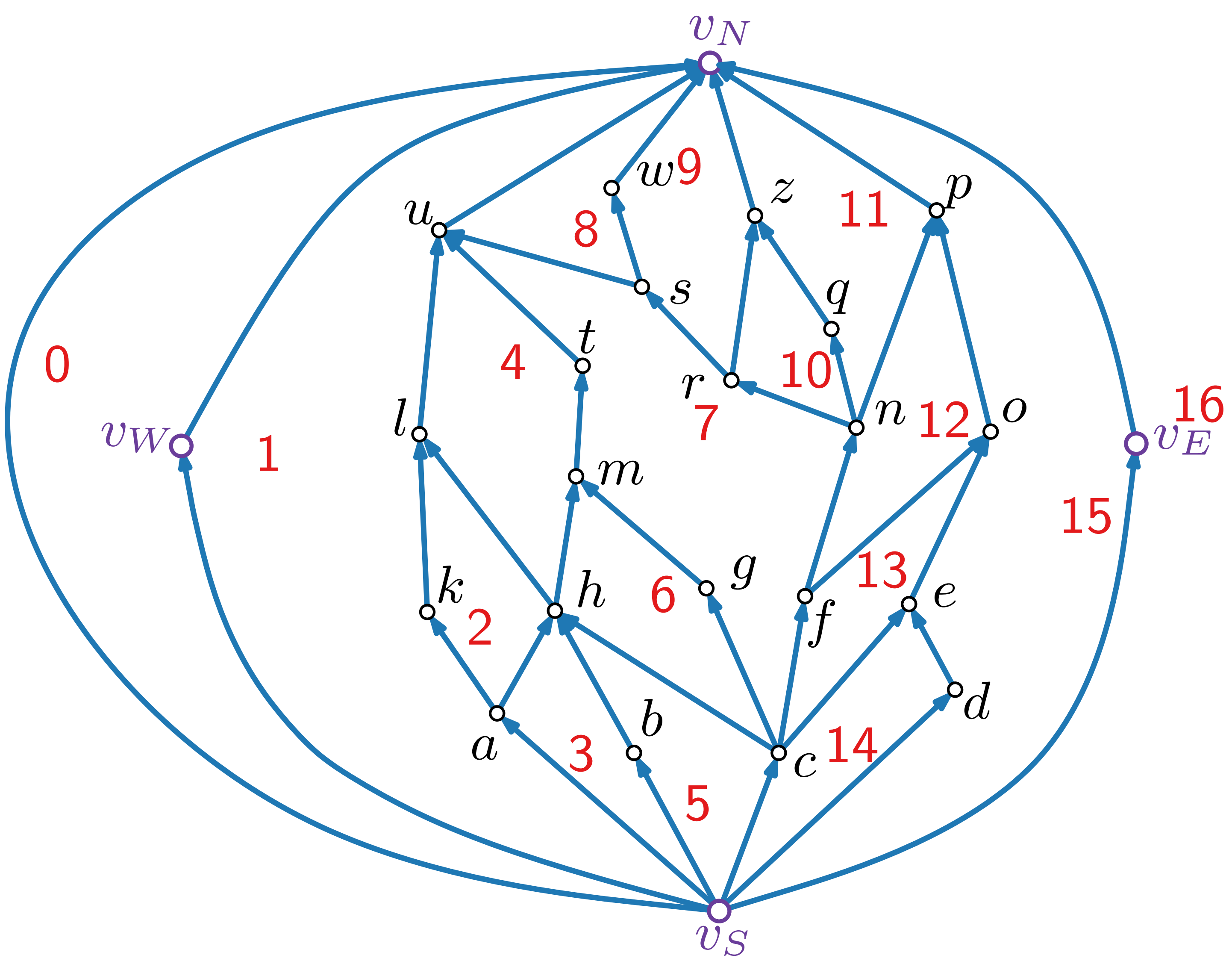
- Find a REL $\{T_r, T_b\}$ of G;
- Construct a SN network G_{ver} of G (consists of T_b plus outer edges)
- Construct the dual $G^\star_{\text{ver}}$ of G_{ver} and compute a topological ordering f_{ver} of $G^\star_{\text{ver}}$
- For each vertex $v \in V$, let g and h be the face on the left and face on the right of v. Set $x_1(v) = f_{\text{ver}}(g)$ and $x_2(v) = f_{\text{ver}}(h)$.
- Define $x_1(v_N) = 0, x_1(v_S) = 1$ and $x_2(v_N) = \max f_{\text{ver}} - 1, x_2(v_S) = \max f_{\text{ver}}$
- Analogously compute y_1 and y_2 with G_{hor}.
- For each $v \in V$, let $R(v) = [x_1(v), x_2(v)] \times [y_1(v), y_2(v)]$.

Reading off Coordinates to Get Rectangular Dual

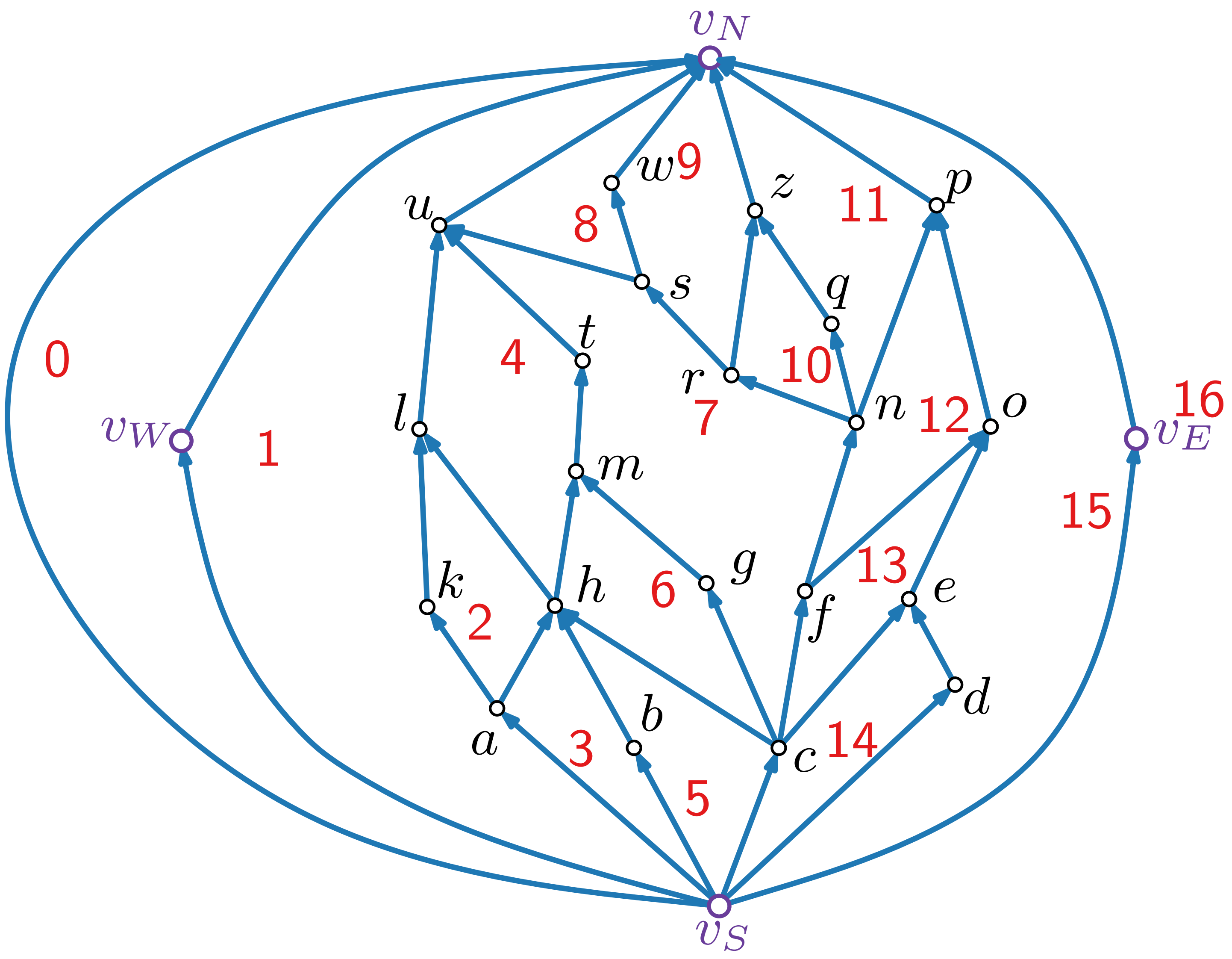


Reading off Coordinates to Get Rectangular Dual

$x_1(v_N) = 0,\ x_2(v_N) = 15$

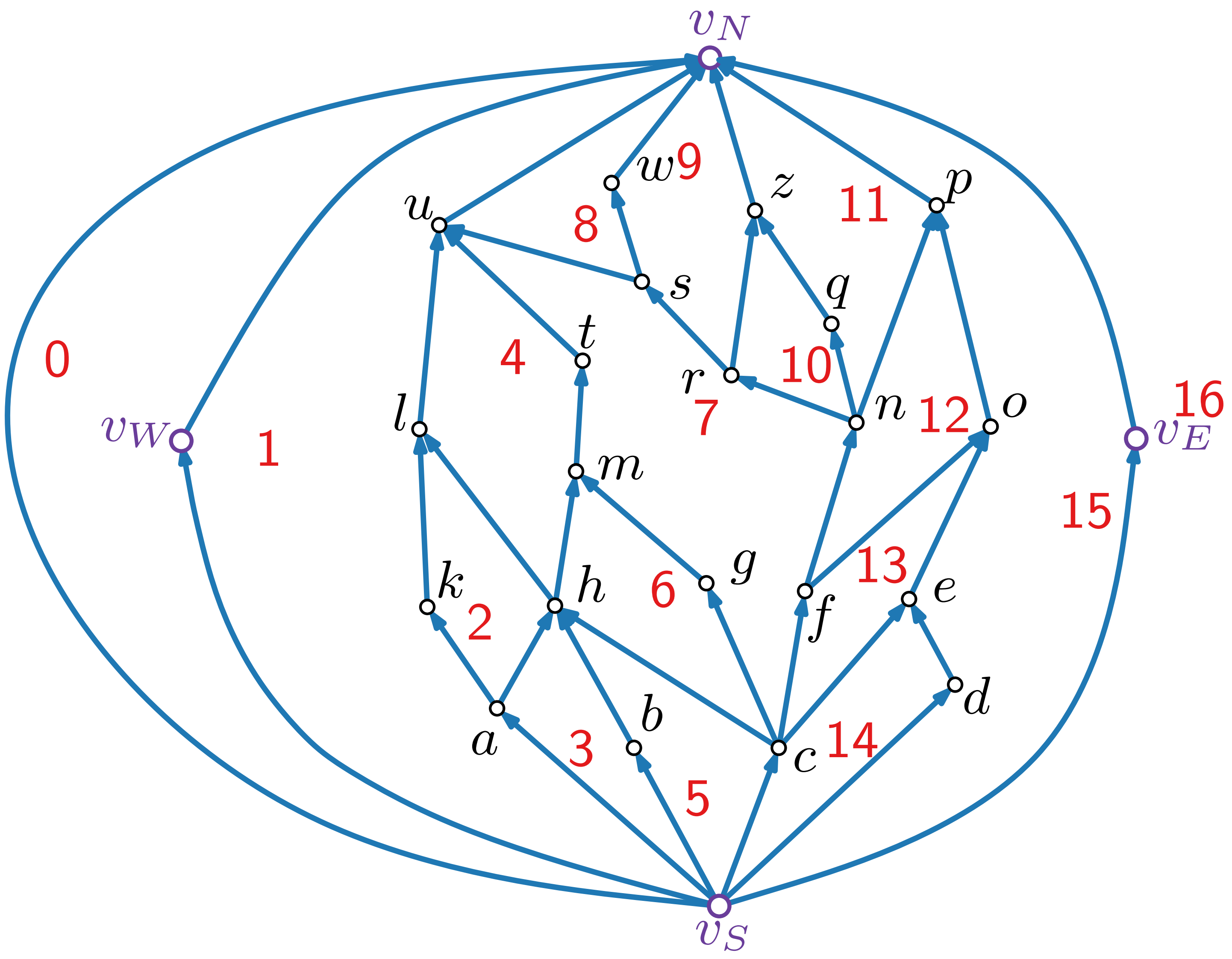


Reading off Coordinates to Get Rectangular Dual



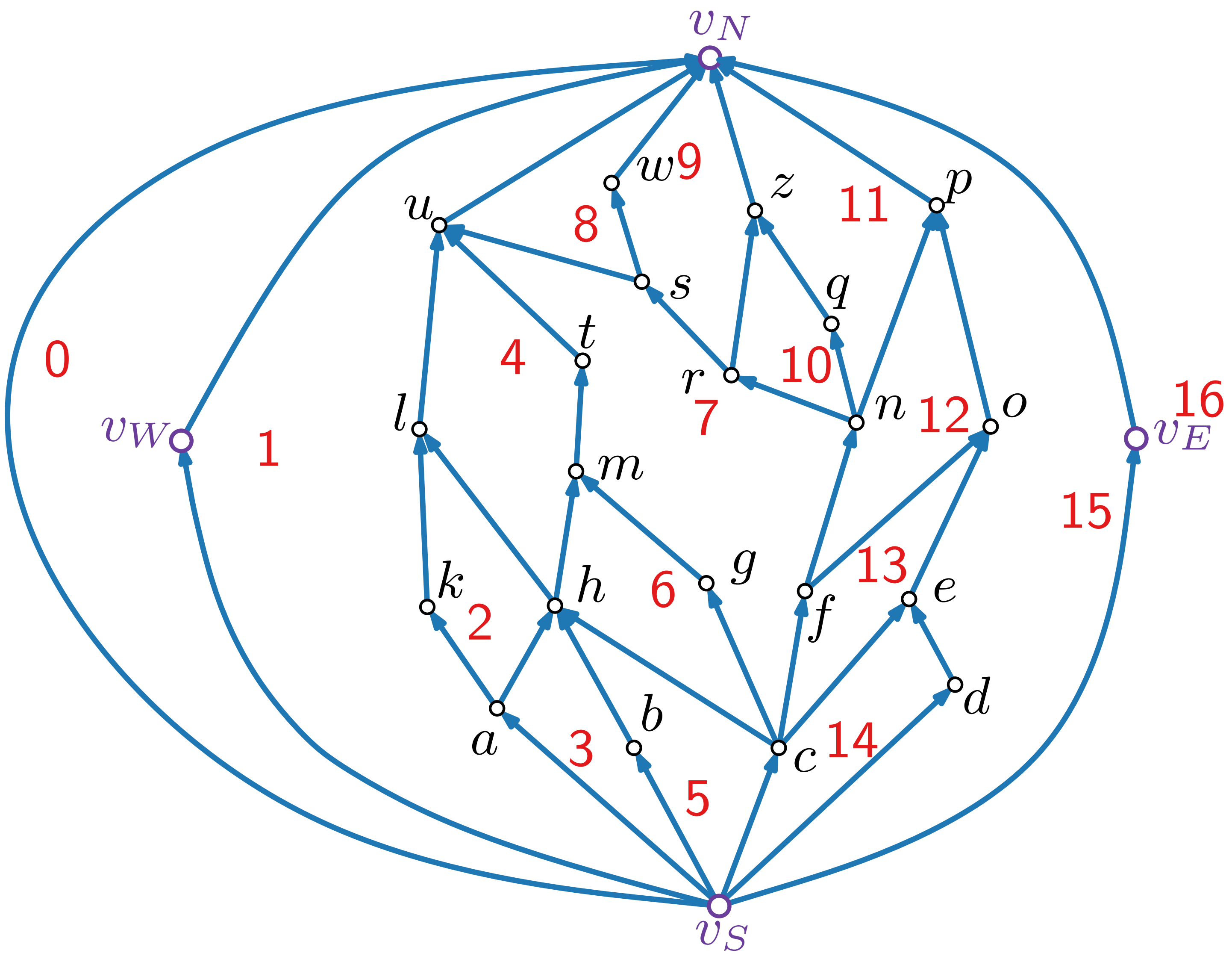
$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$

Reading off Coordinates to Get Rectangular Dual



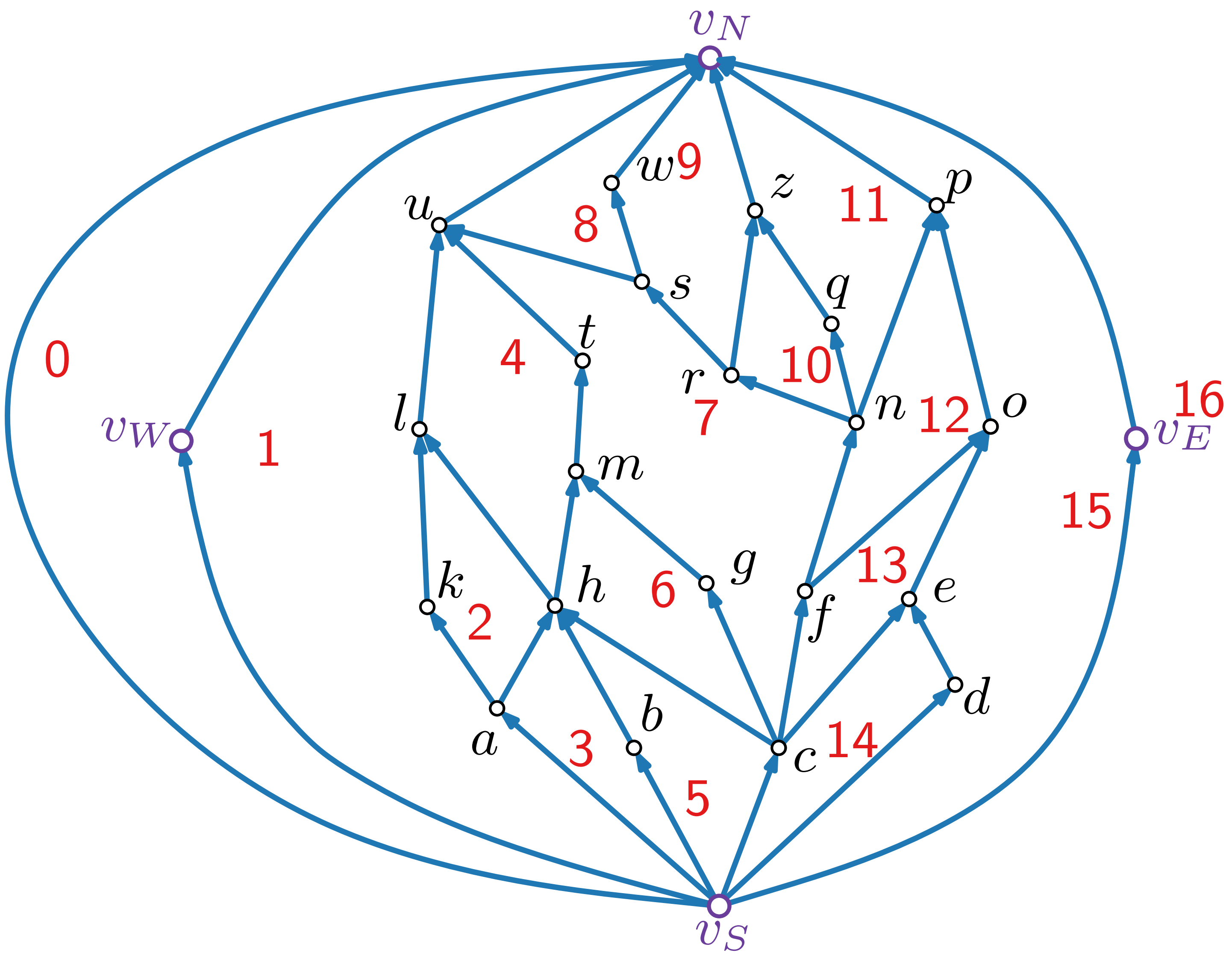
$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$

Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$

Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$

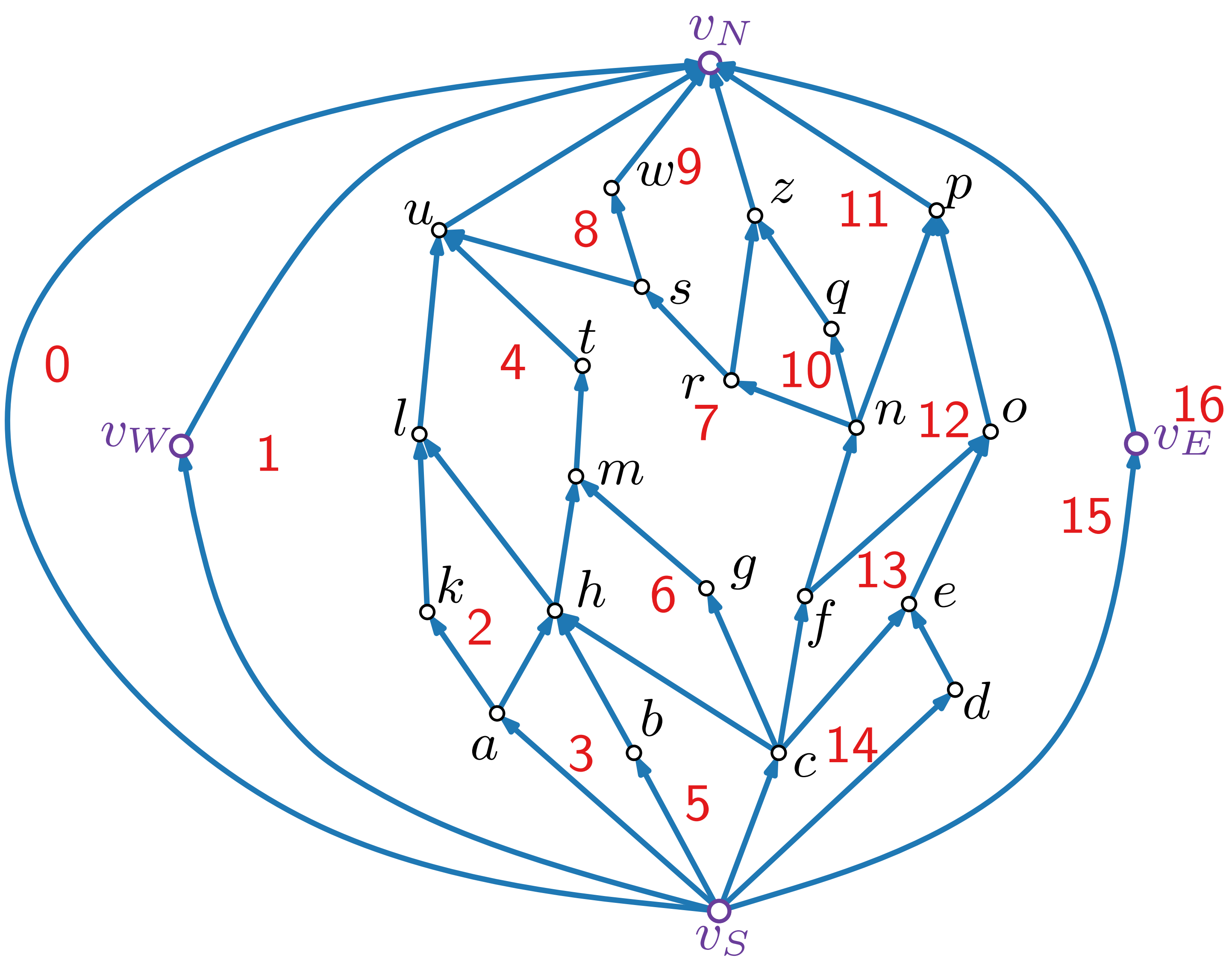
$x_1(v_S) = 1,\ x_2(v_S) = 16$

$x_1(v_W) = 0, x_2(v_W) = 1$

$x_1(v_E) = 15,\ x_2(v_E) = 16$

$x_1(a) = 1,\ x_2(a) = 3$

Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$

$x_1(v_S) = 1,\ x_2(v_S) = 16$

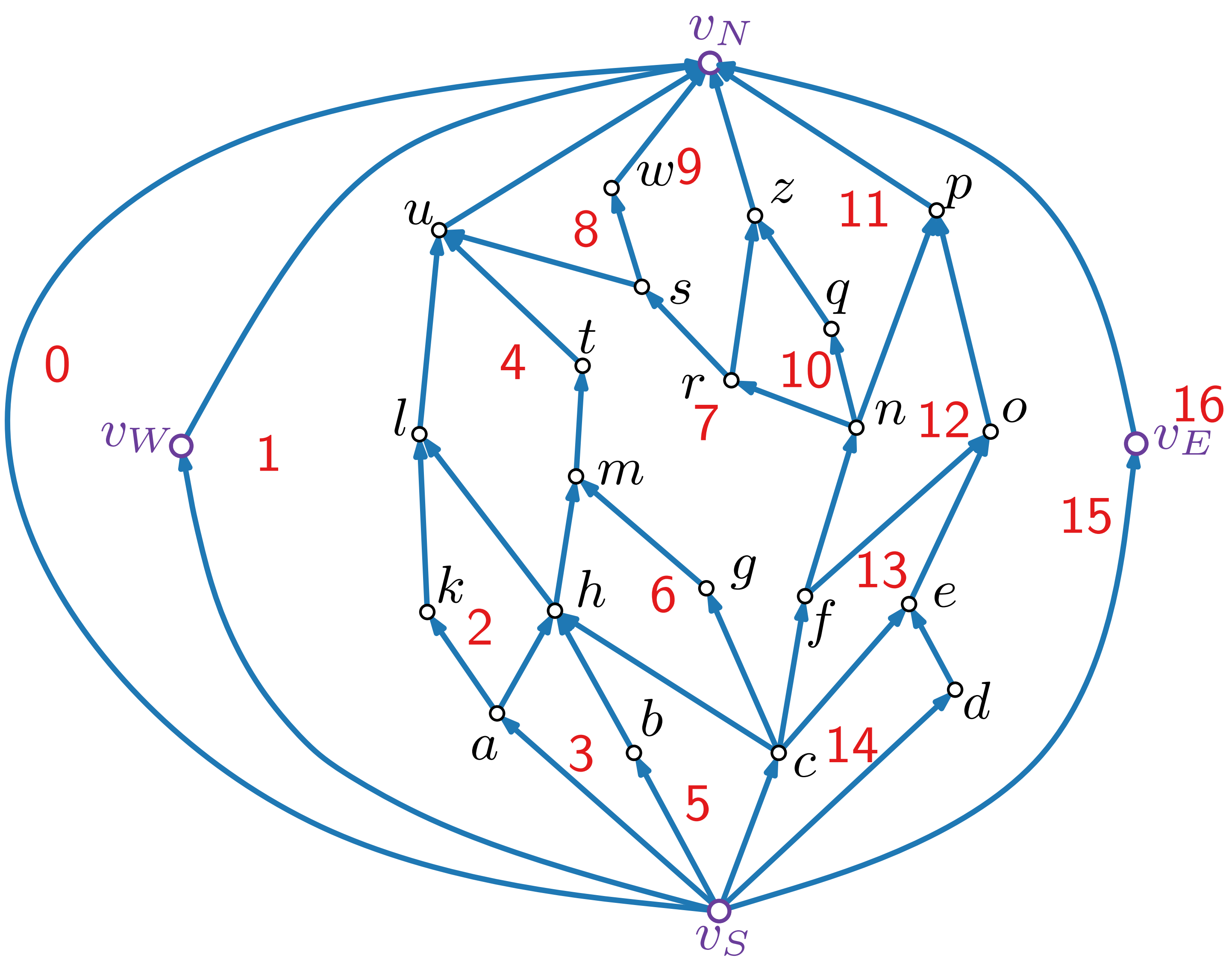
$x_1(v_W) = 0, x_2(v_W) = 1$

$x_1(v_E) = 15,\ x_2(v_E) = 16$

$x_1(a) = 1,\ x_2(a) = 3$

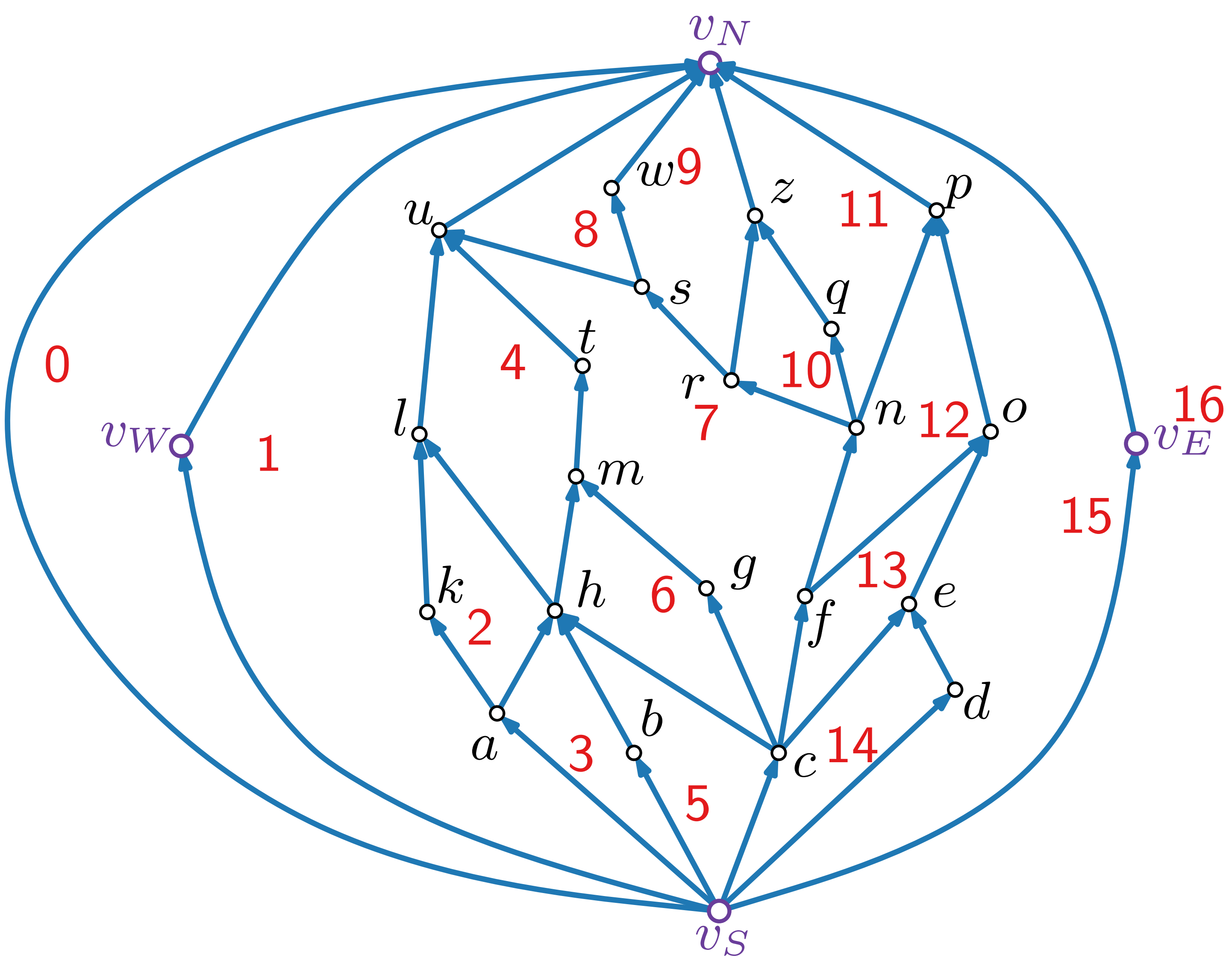
$x_1(b) = 3,\ x_2(b) = 5$

Reading off Coordinates to Get Rectangular Dual



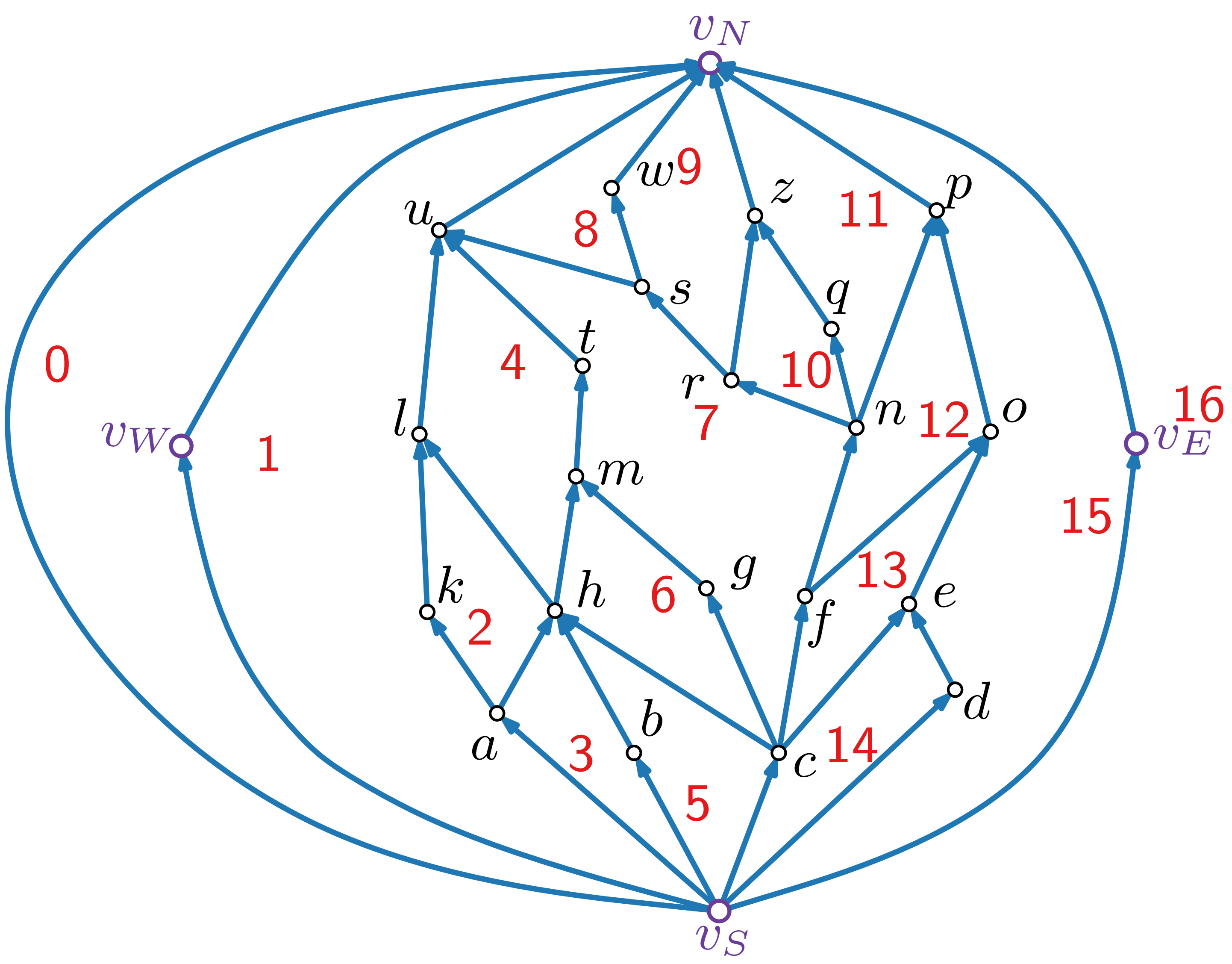
$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$

Reading off Coordinates to Get Rectangular Dual



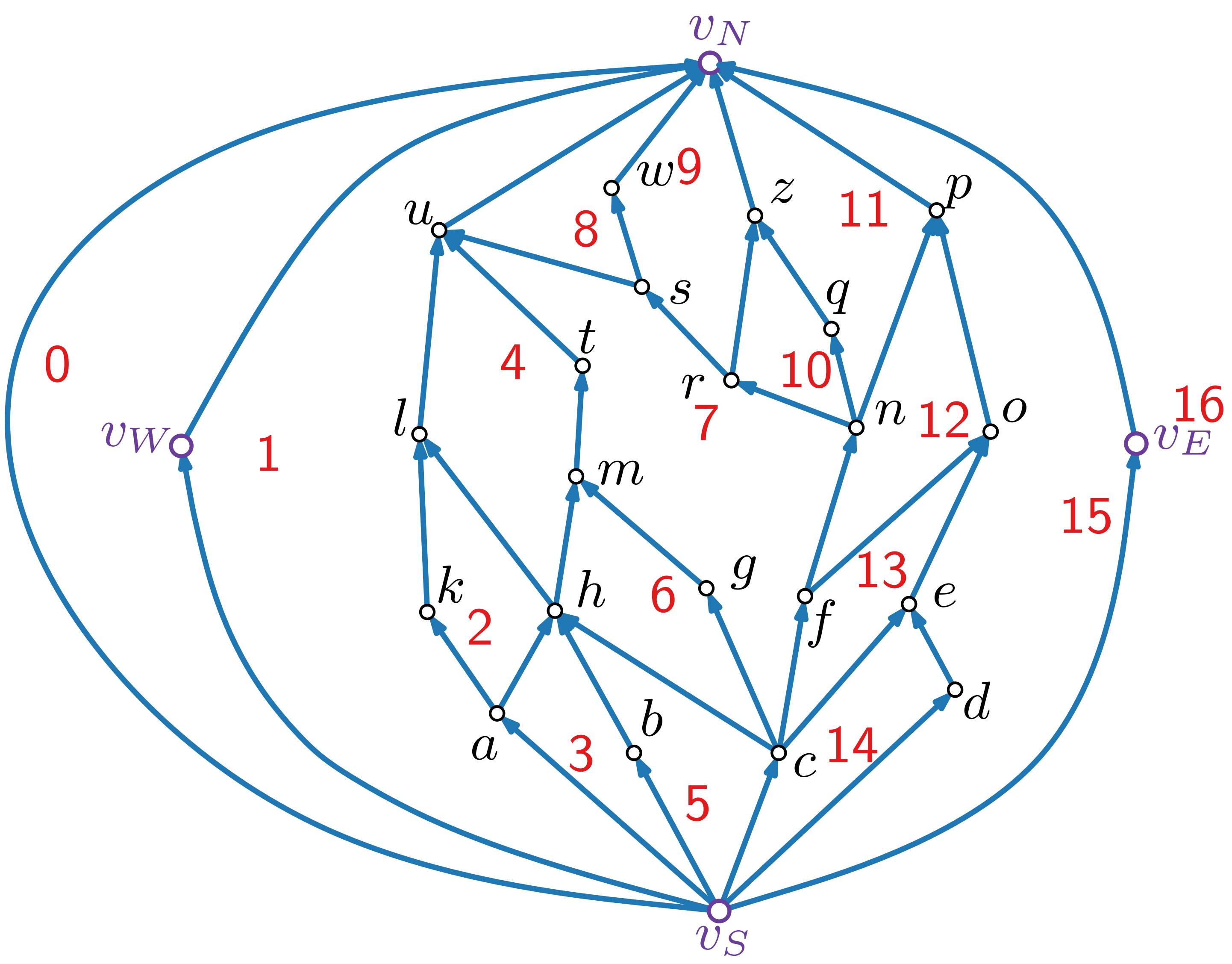
$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$

Reading off Coordinates to Get Rectangular Dual



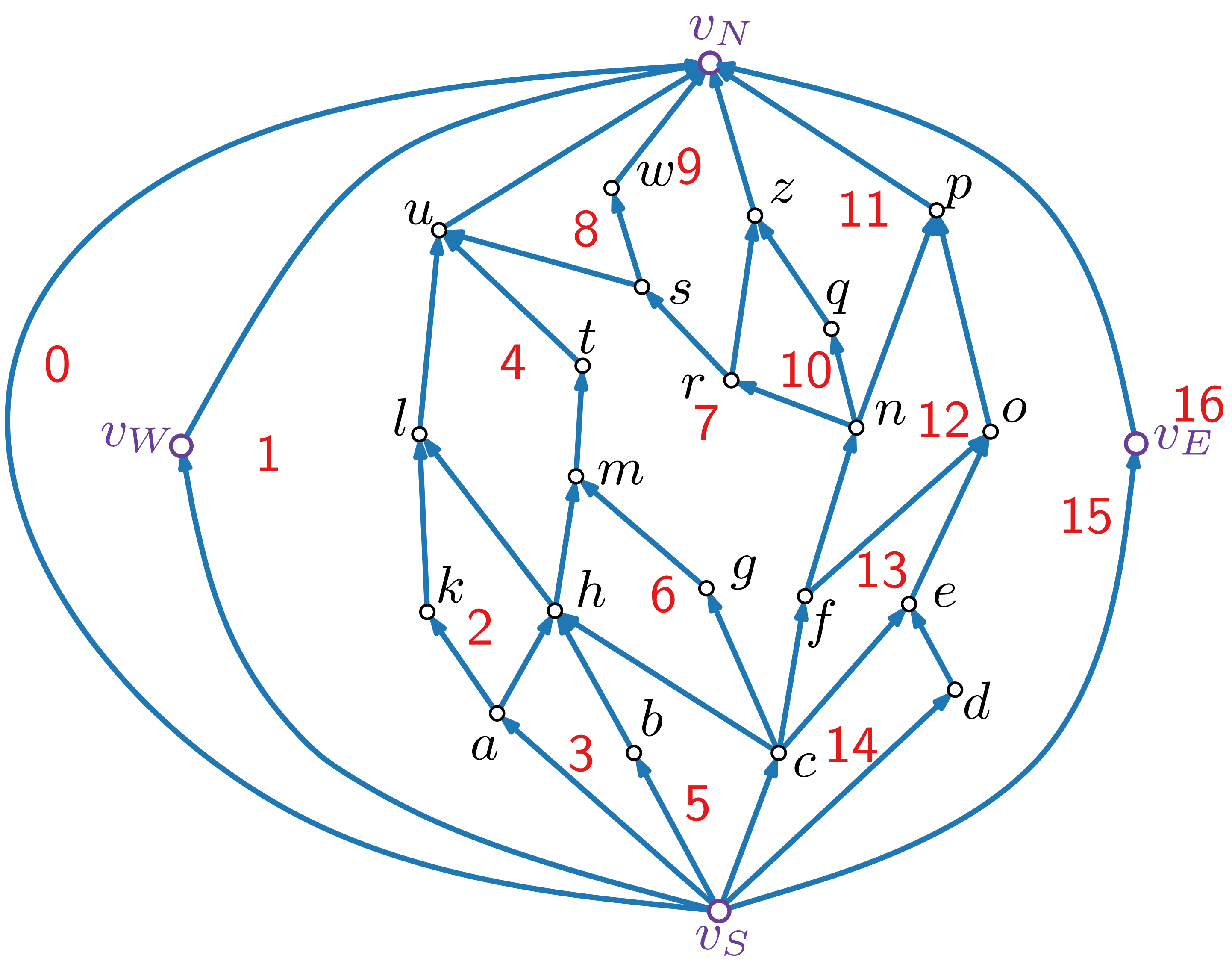
$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$

Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
$\ldots$

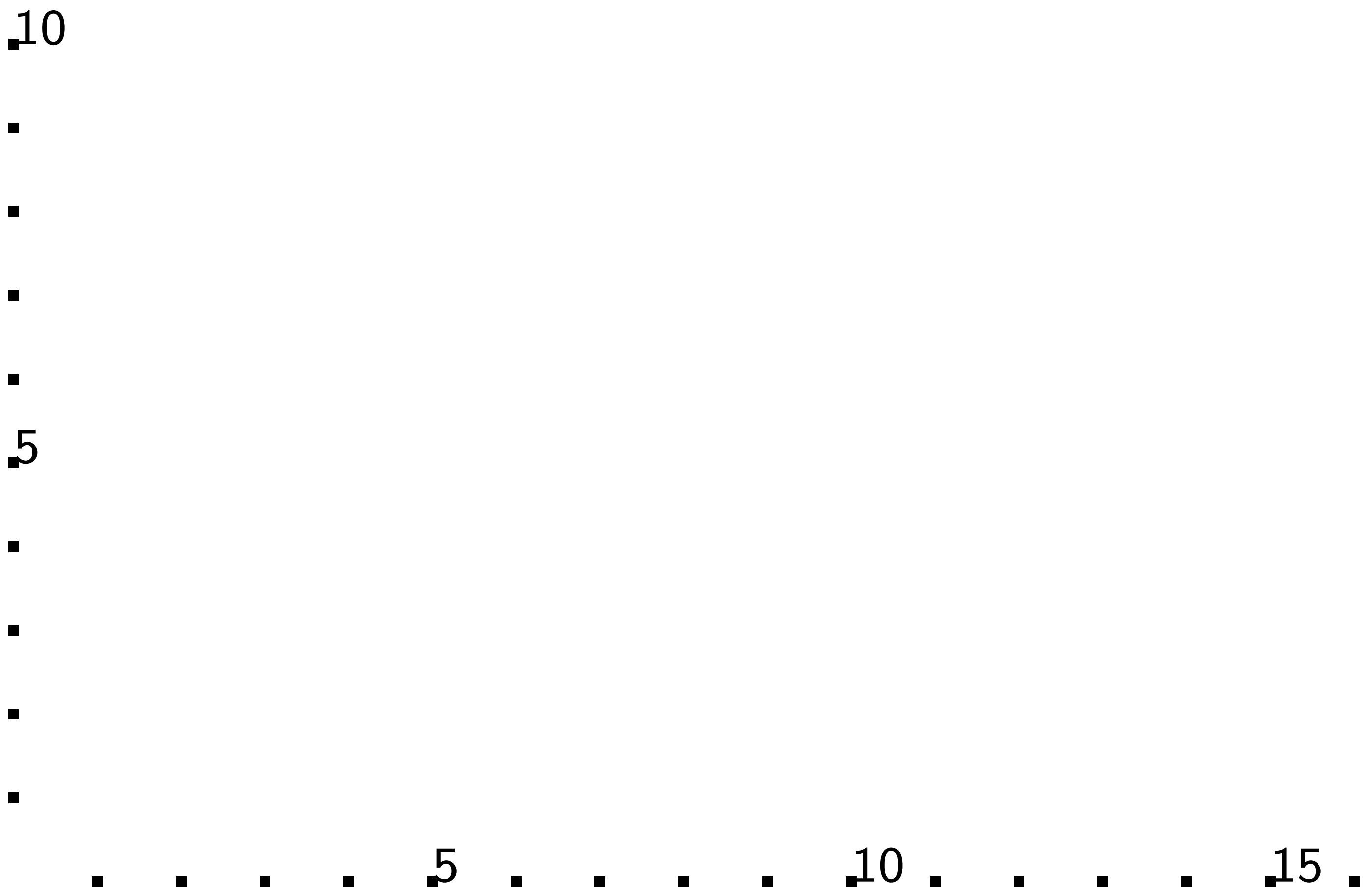
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
...

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
...

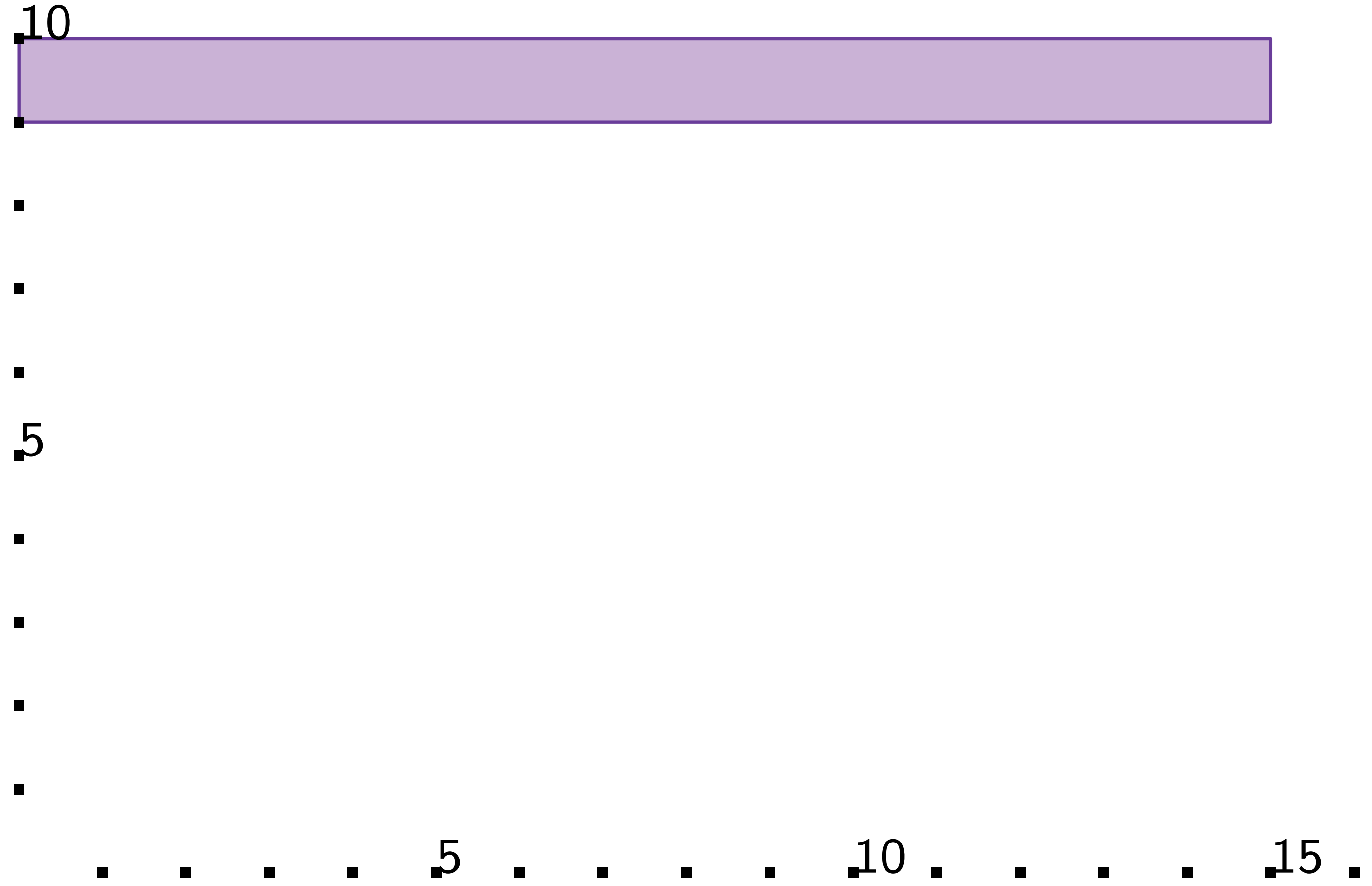
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
...

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
...

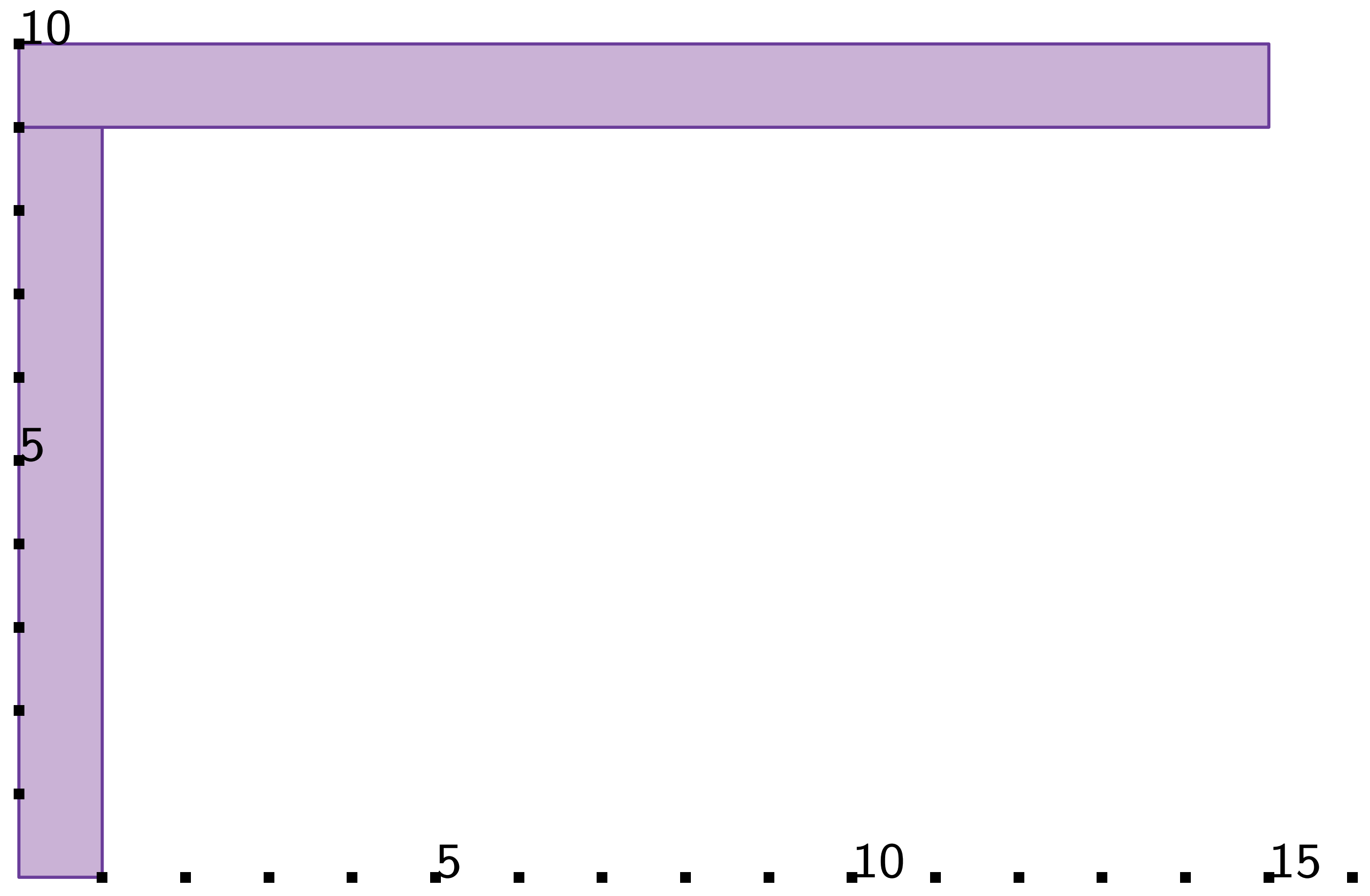
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
...

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
...

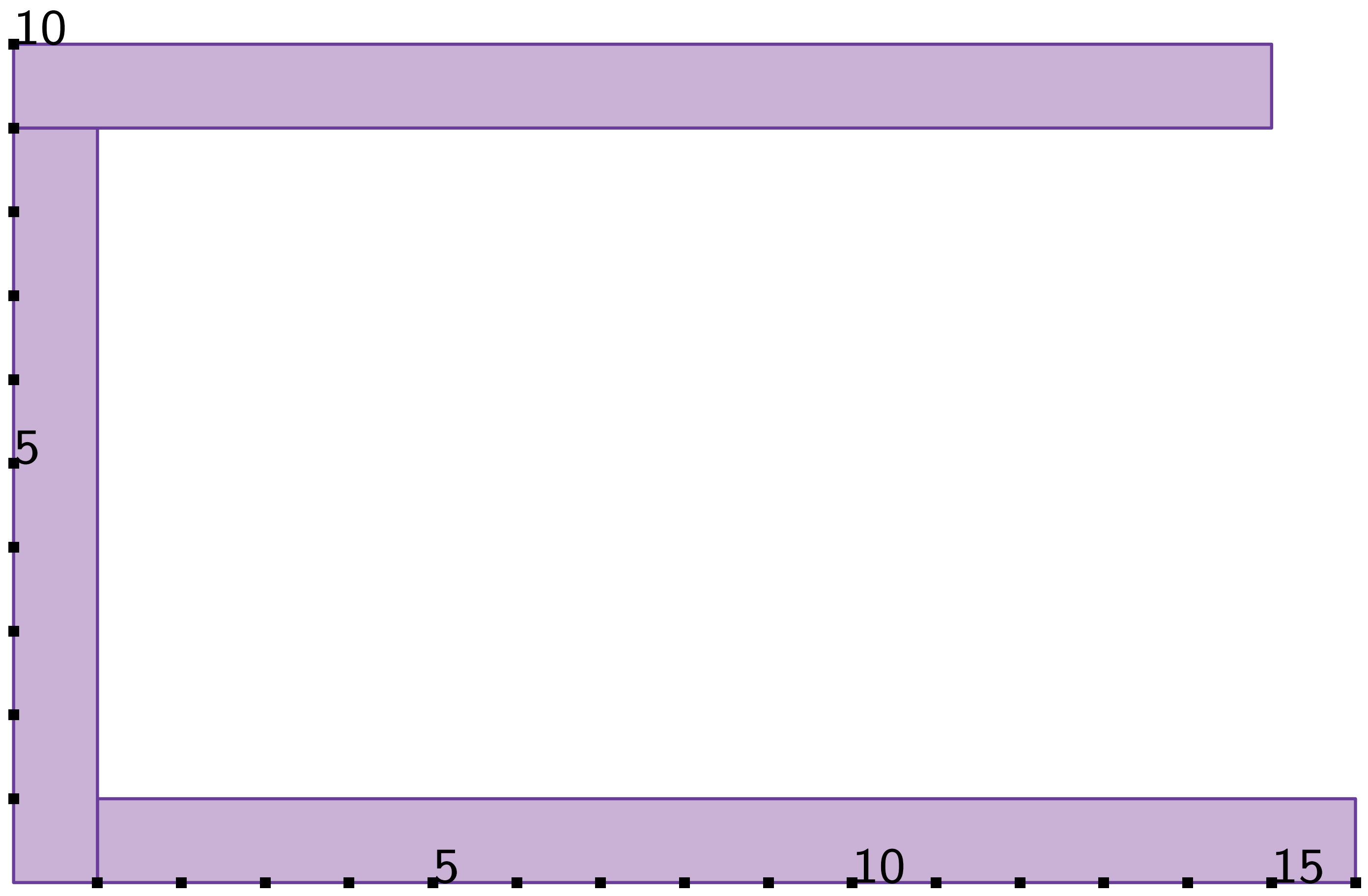
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
...

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
...

Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
...

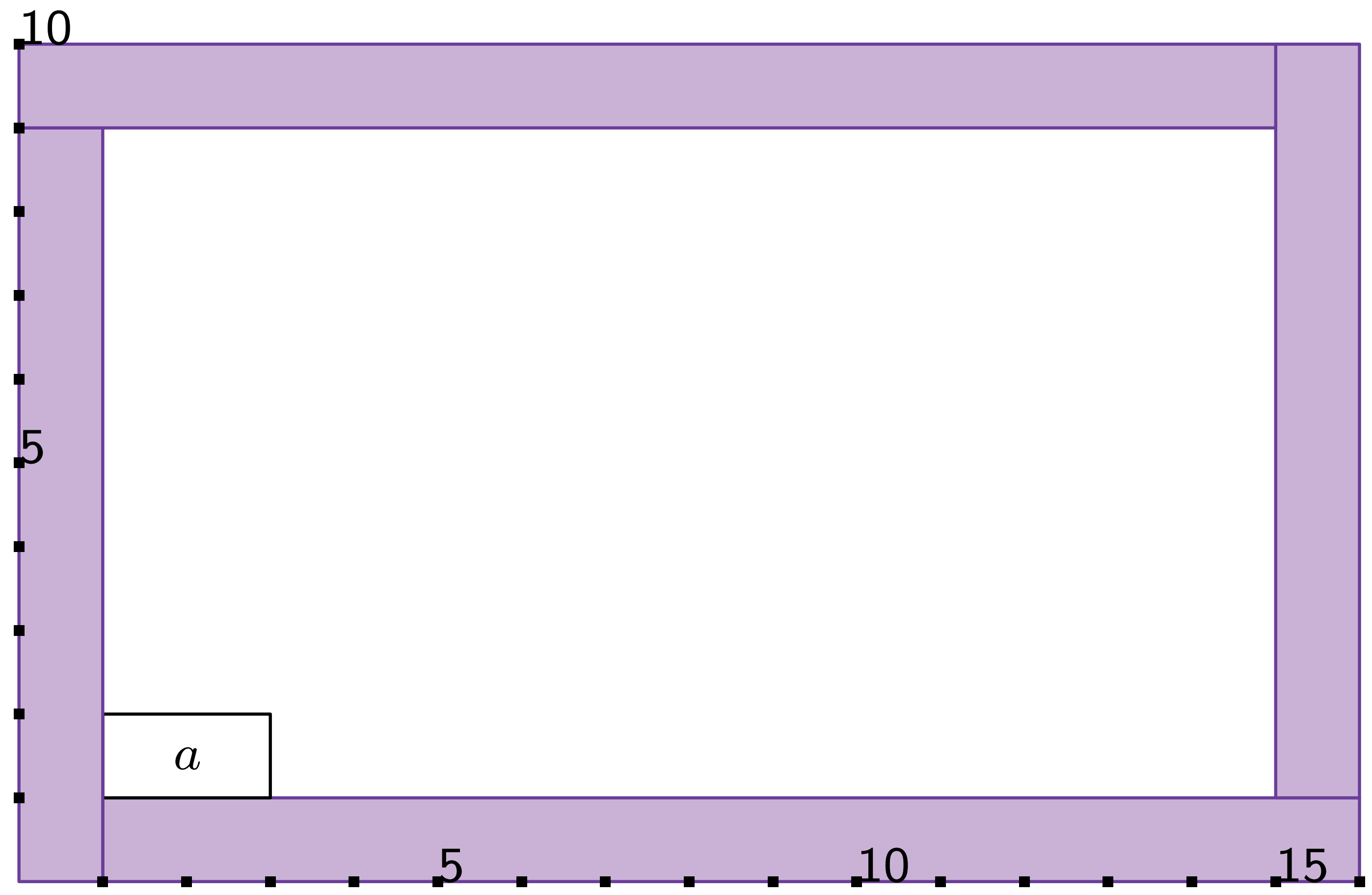
$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
...

Reading off Coordinates to Get Rectangular Dual

$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
...

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
...

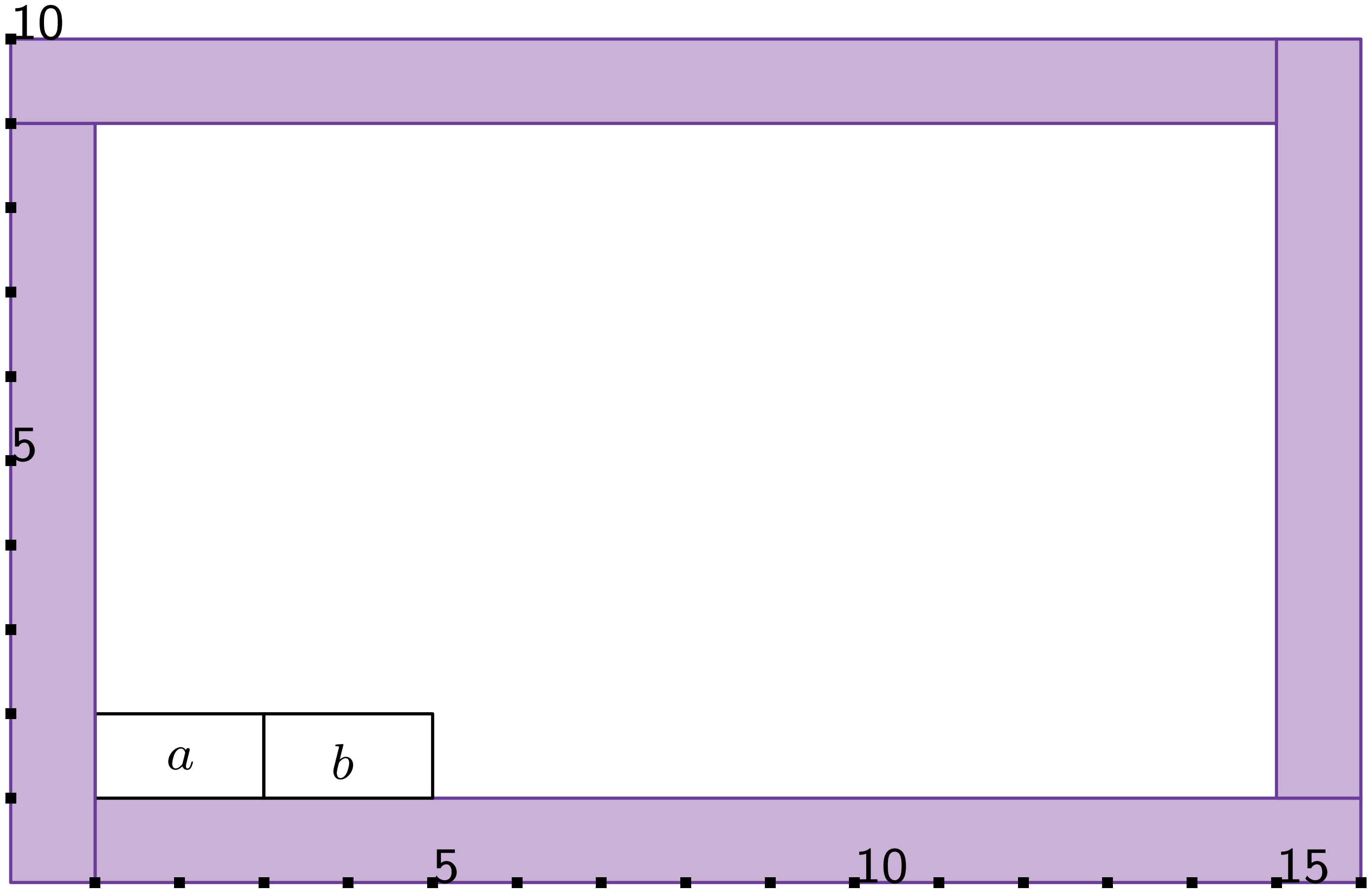
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
...

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
...

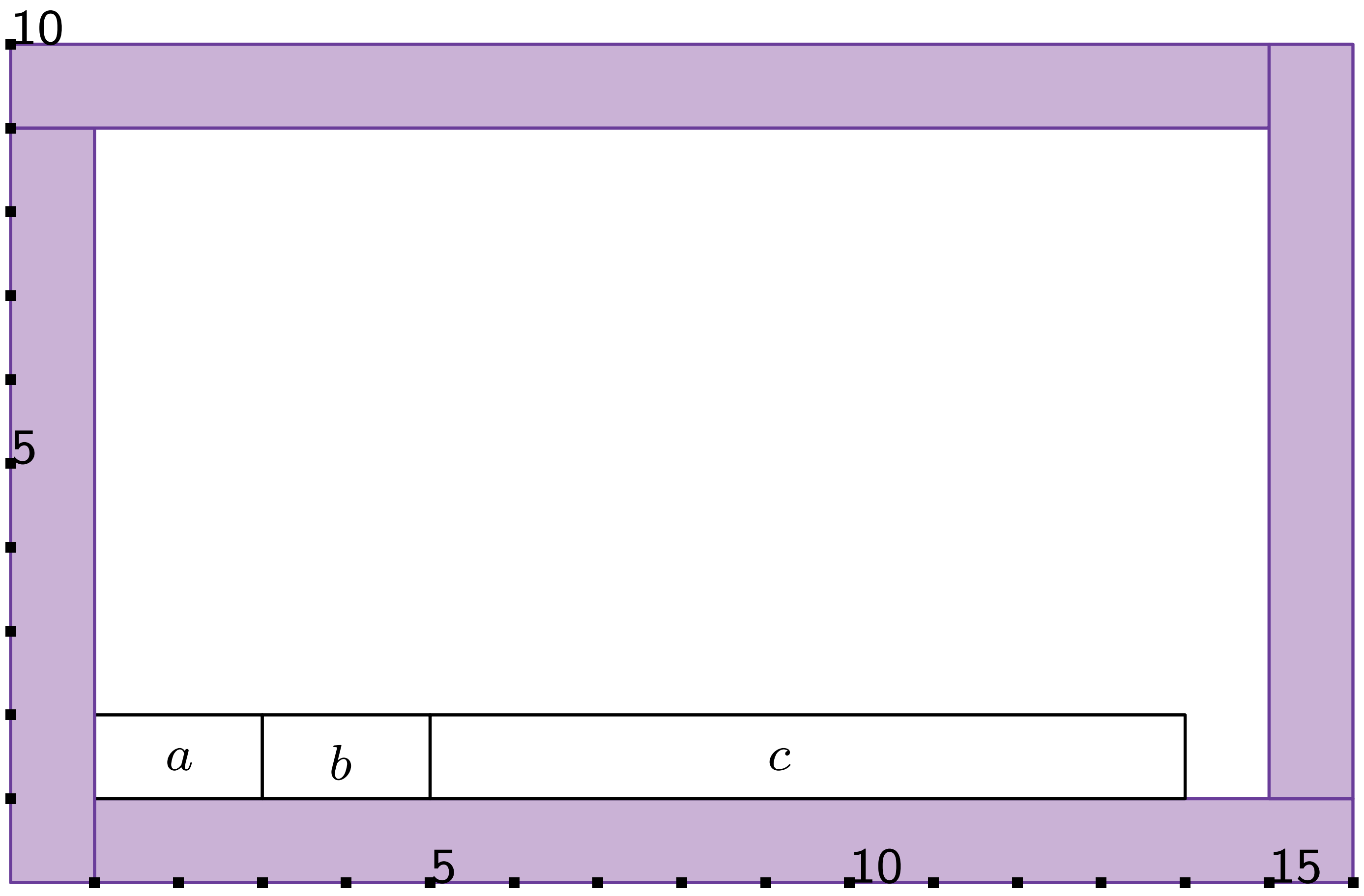
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
...

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
...

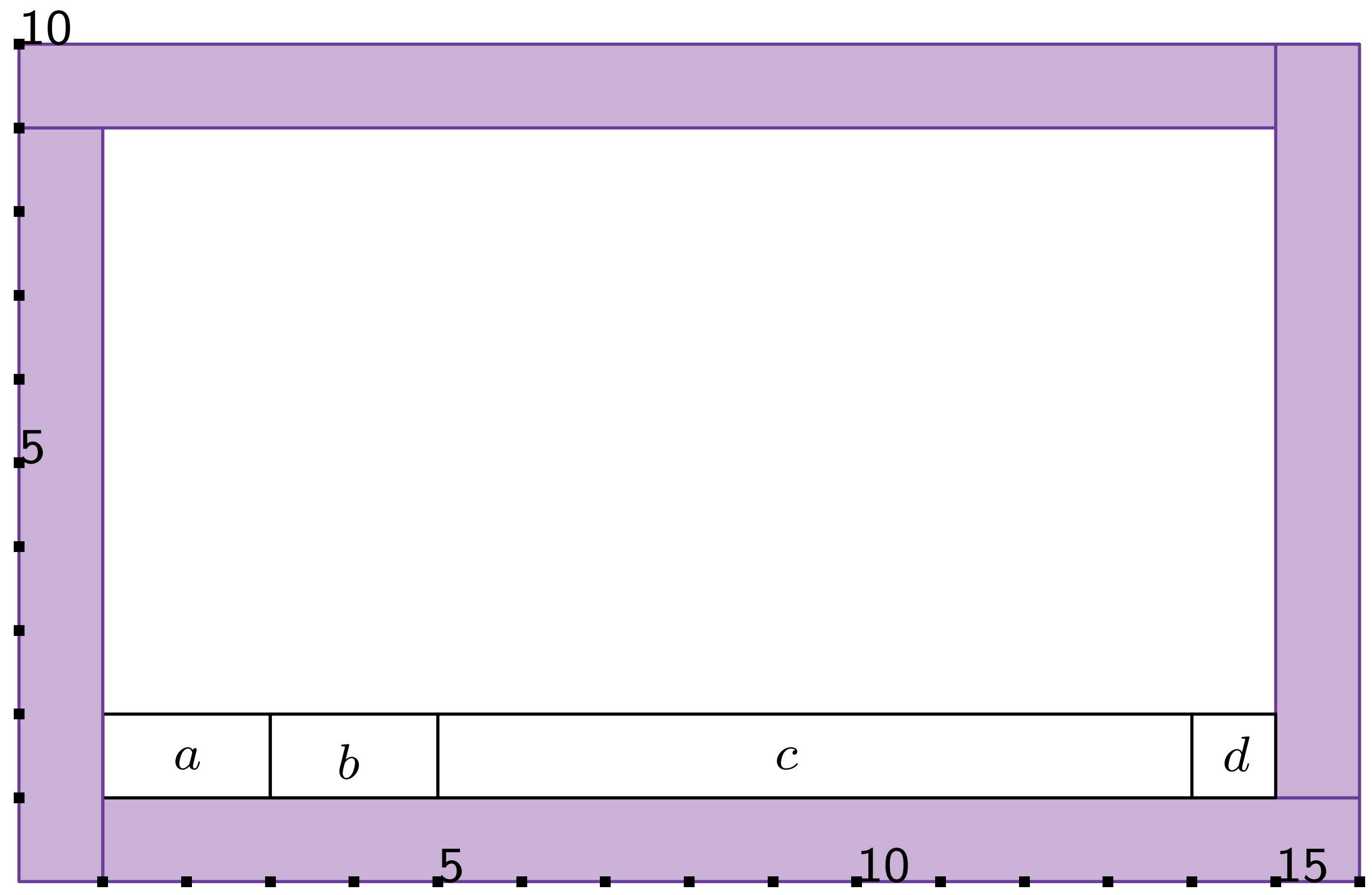
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
$\ldots$

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
$\ldots$

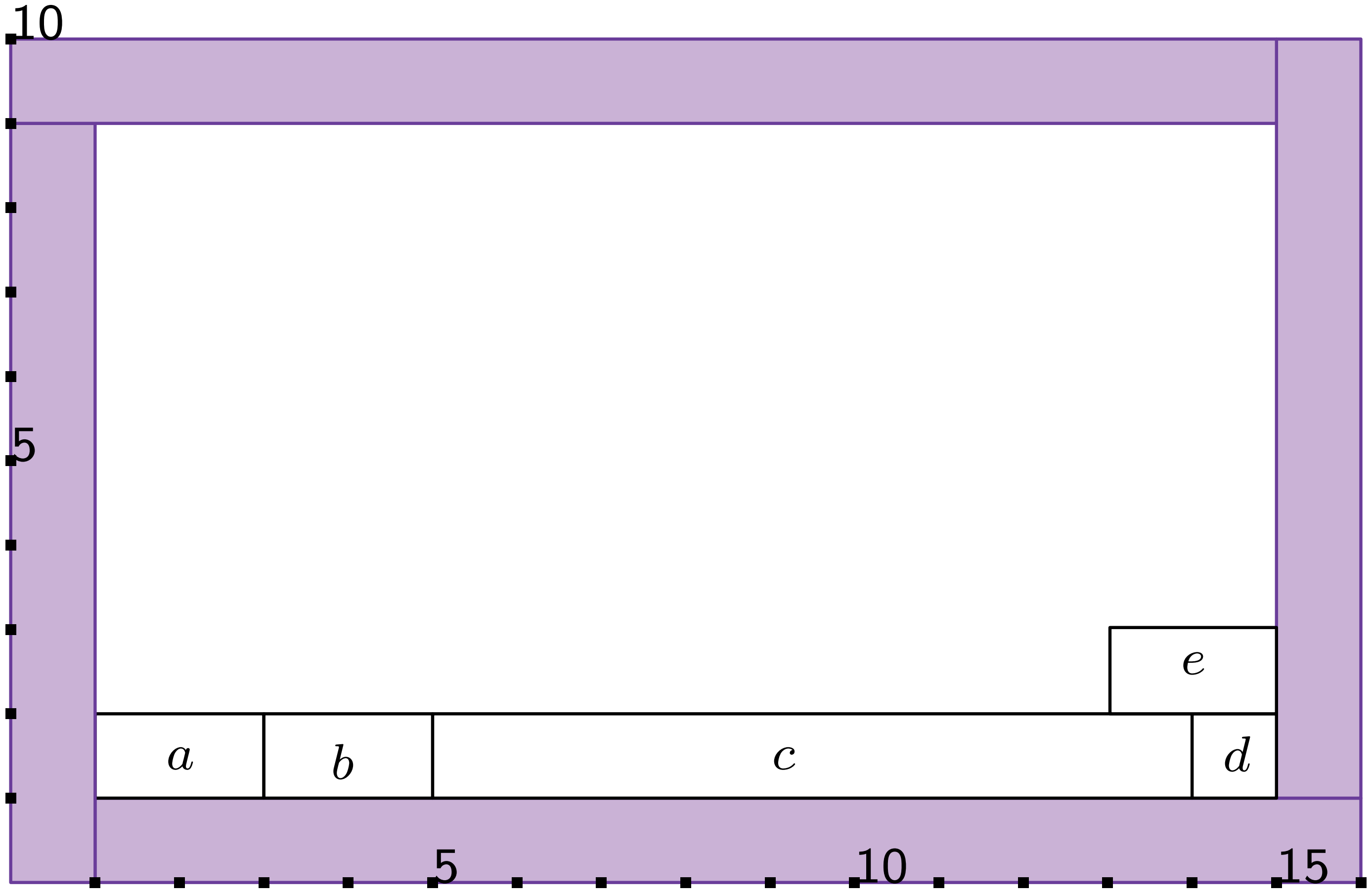
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
...

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
...

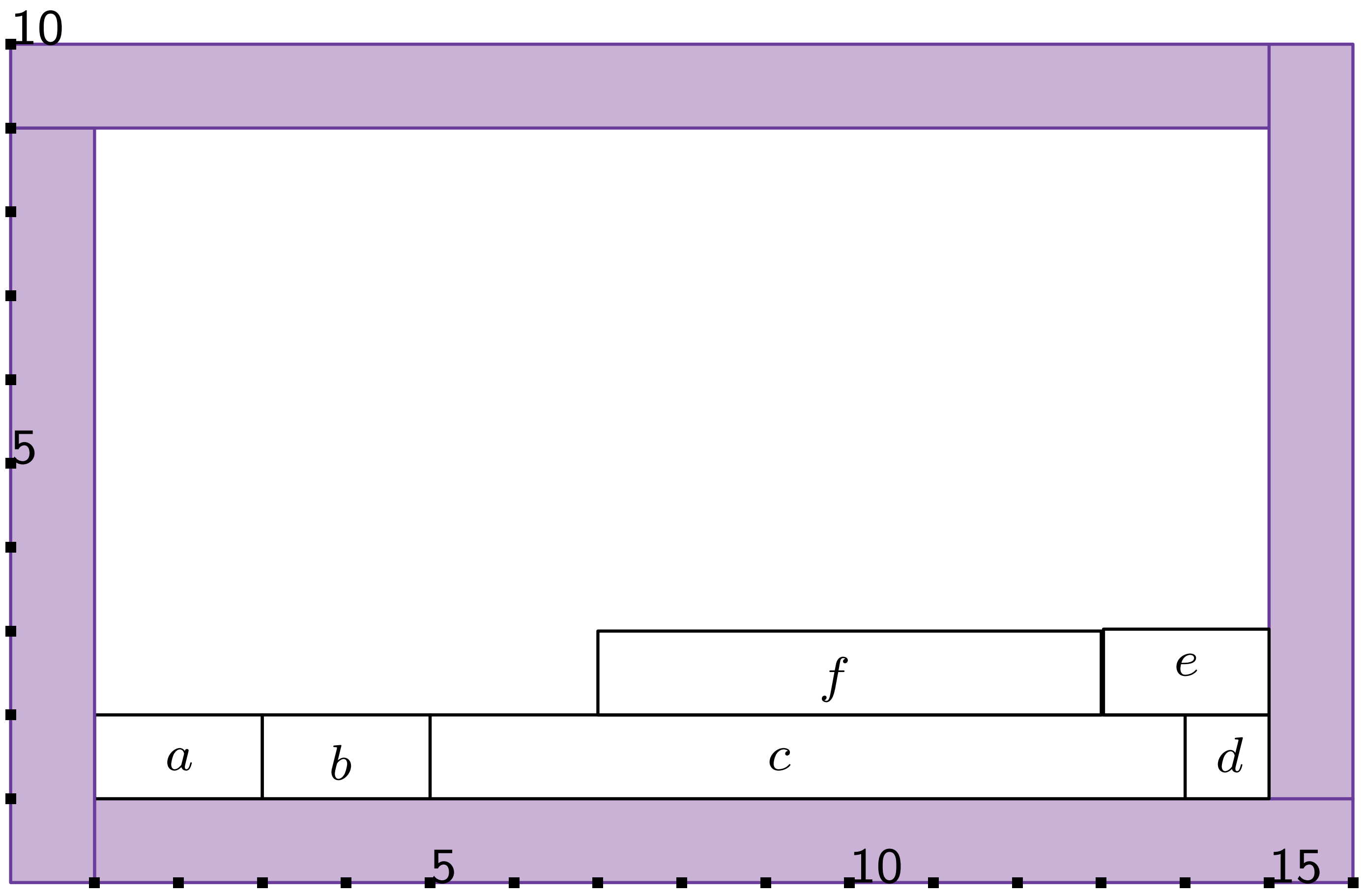
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
$\ldots$

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
$\ldots$

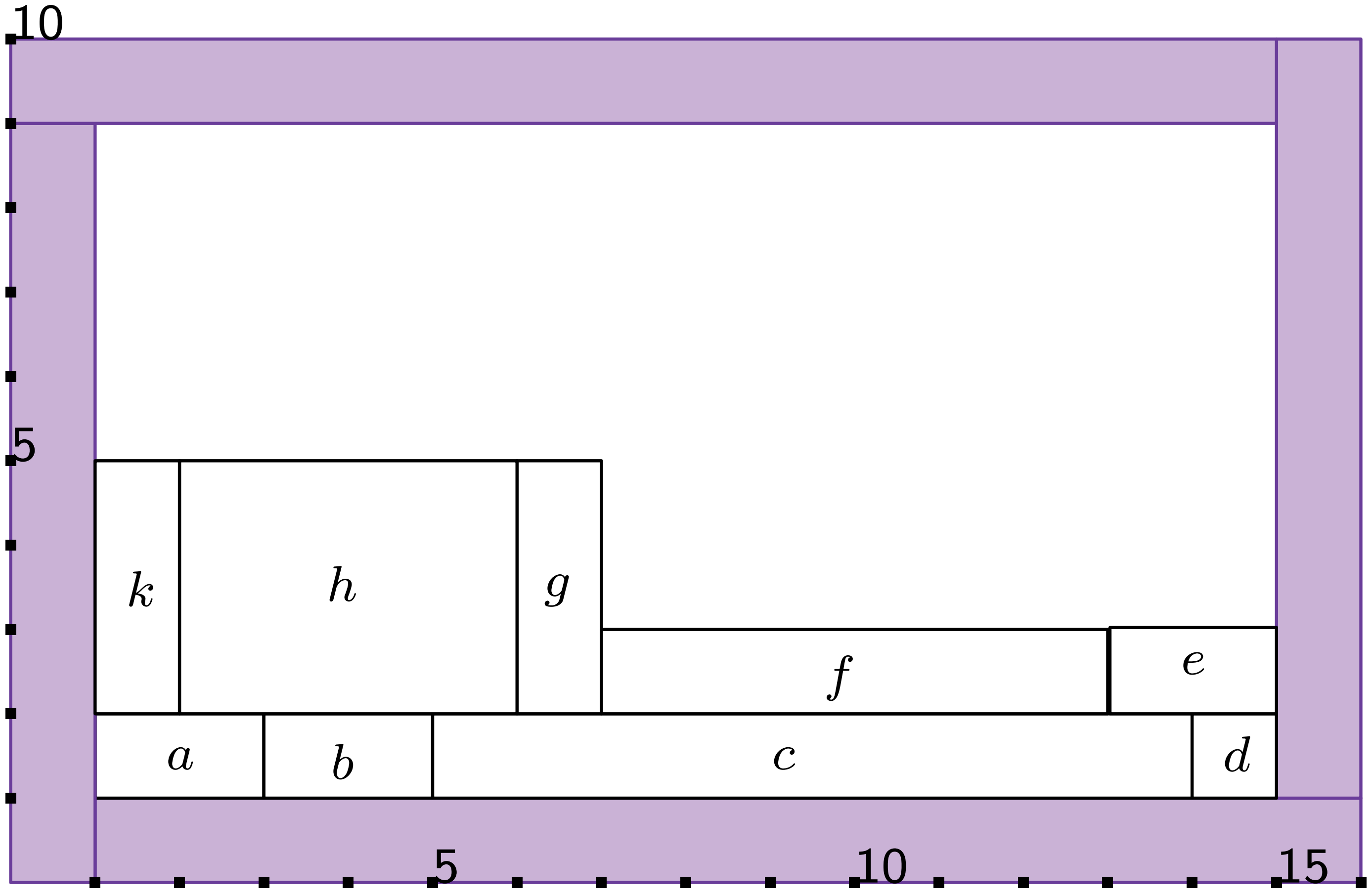
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
...

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
...

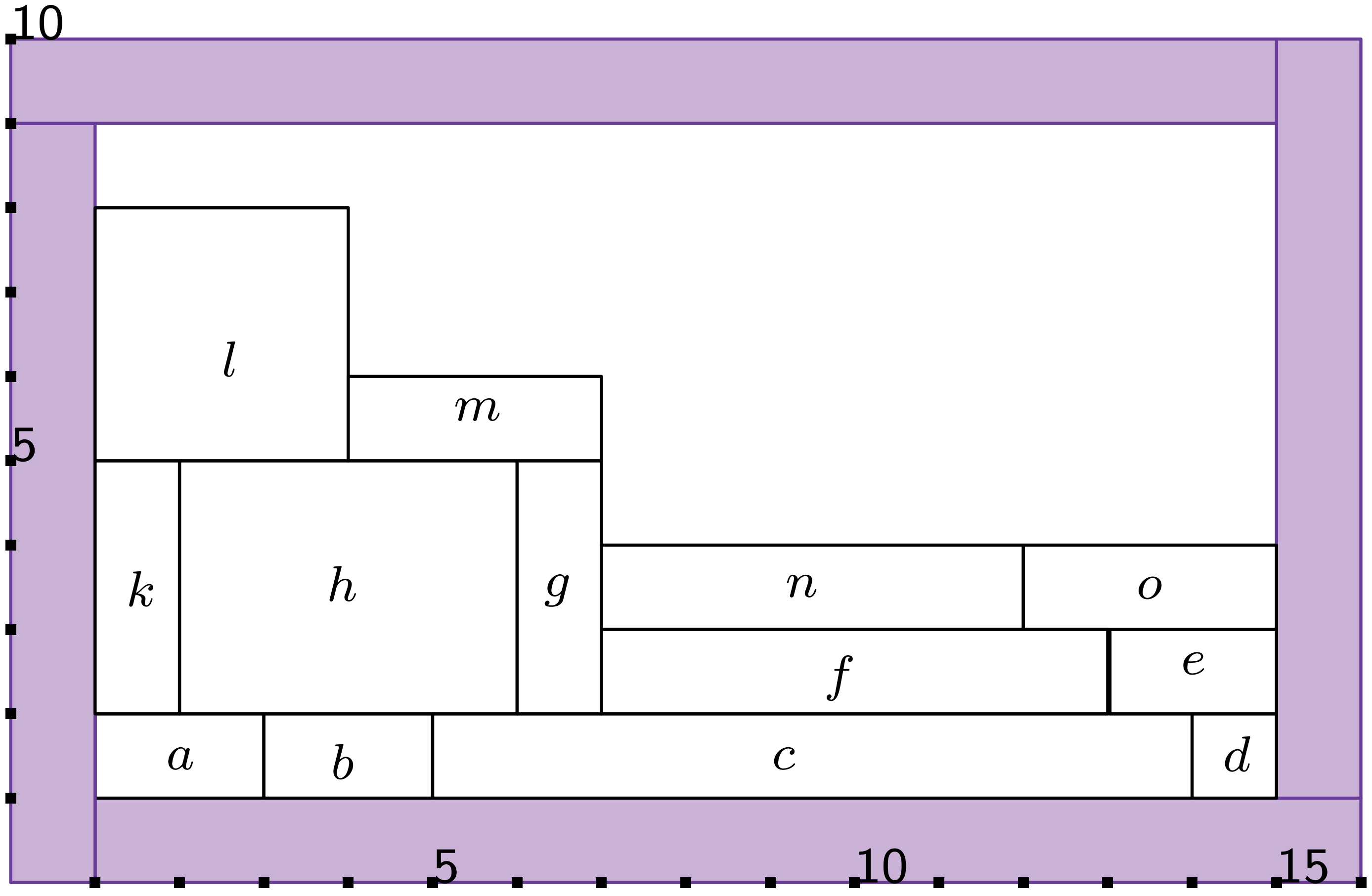
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
$\dots$

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
$\dots$

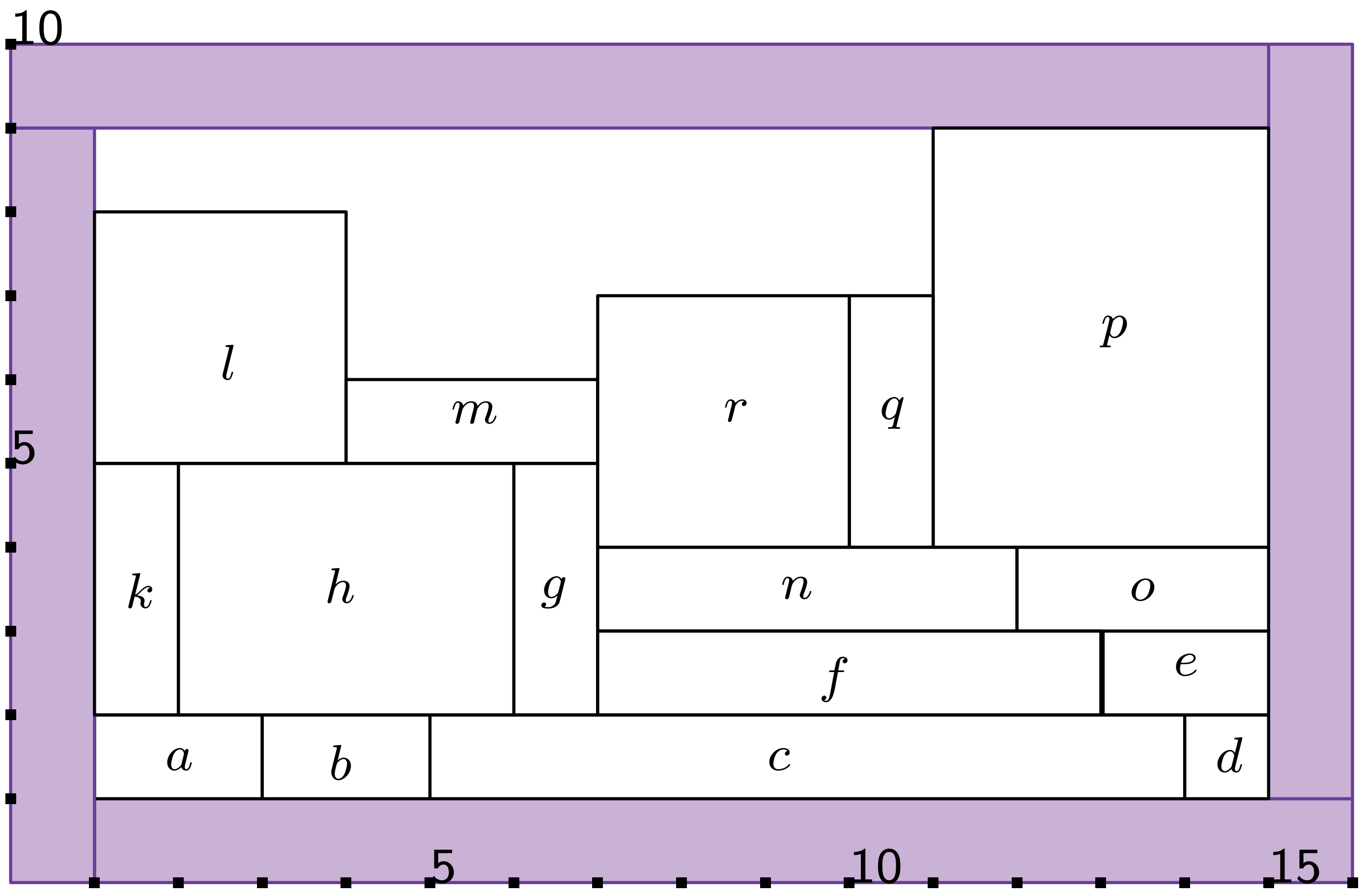
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
...

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
...

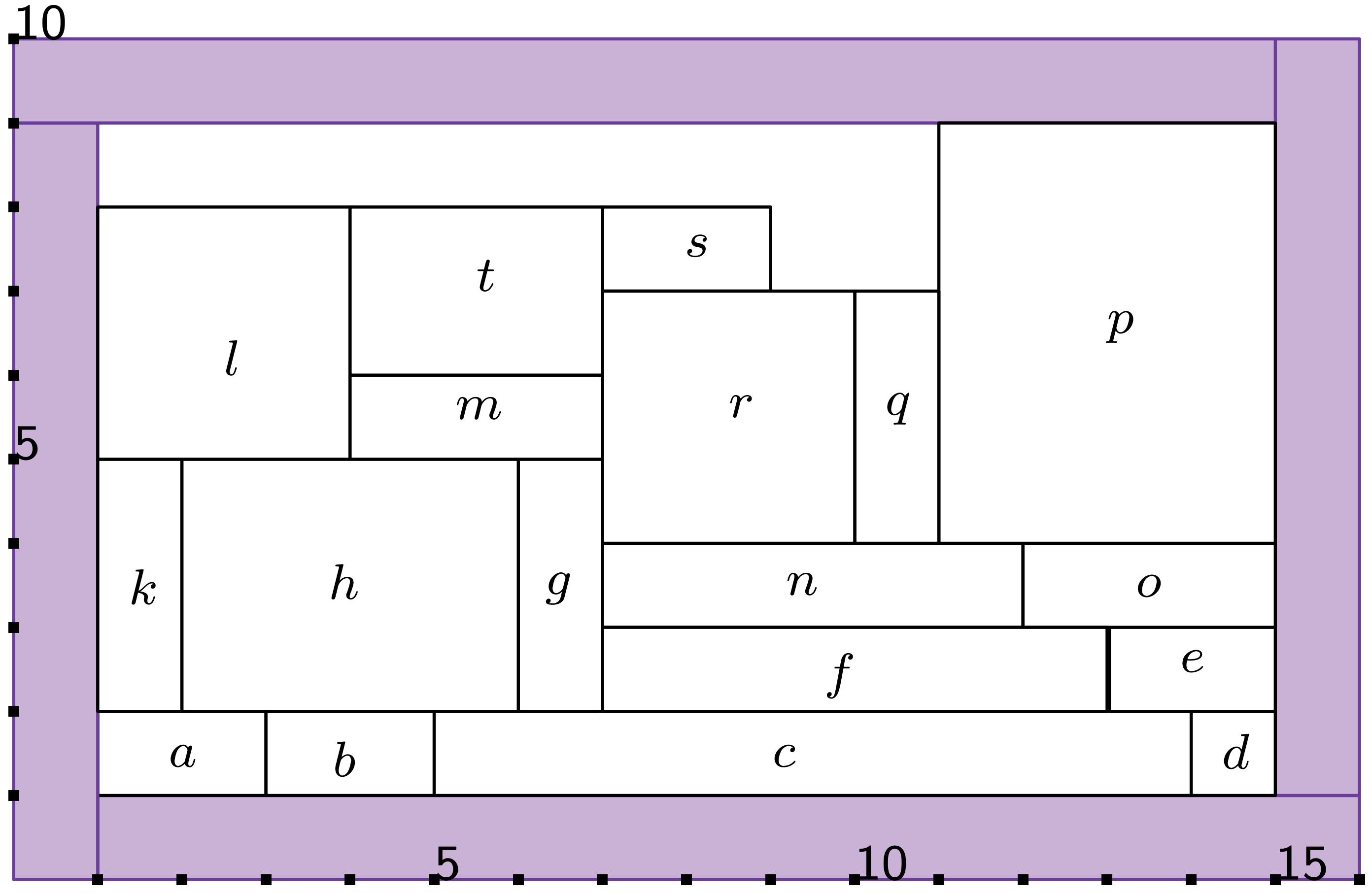
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
$\dots$

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
$\dots$

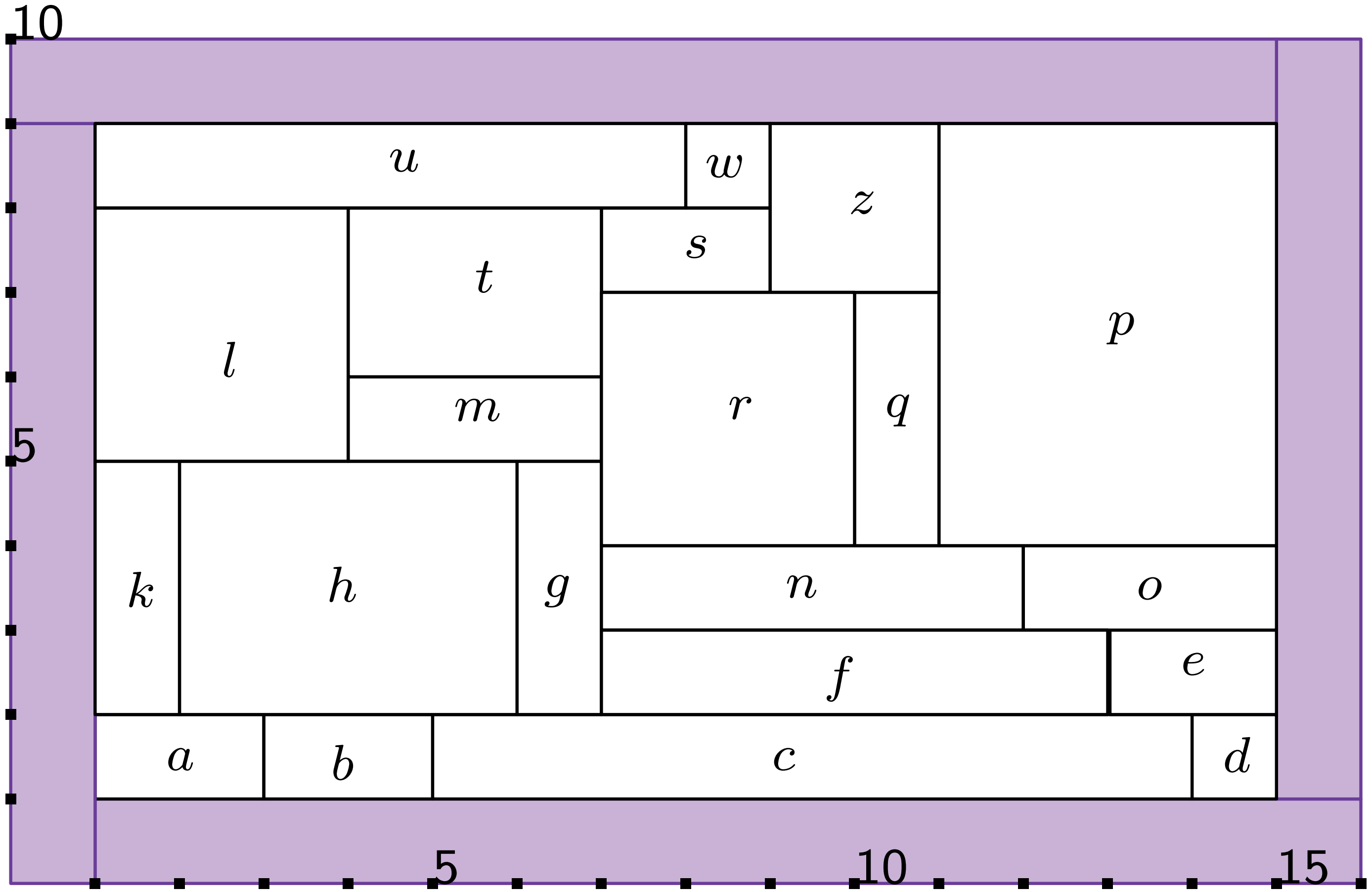
Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
...

$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
...

Reading off Coordinates to Get Rectangular Dual



$x_1(v_N) = 0,\ x_2(v_N) = 15$
$x_1(v_S) = 1,\ x_2(v_S) = 16$
$x_1(v_W) = 0, x_2(v_W) = 1$
$x_1(v_E) = 15,\ x_2(v_E) = 16$
$x_1(a) = 1,\ x_2(a) = 3$
$x_1(b) = 3,\ x_2(b) = 5$
$x_1(c) = 5,\ x_2(c) = 14$
$x_1(d) = 14,\ x_2(d) = 15$
$x_1(e) = 13,\ x_2(e) = 15$
$\ldots$

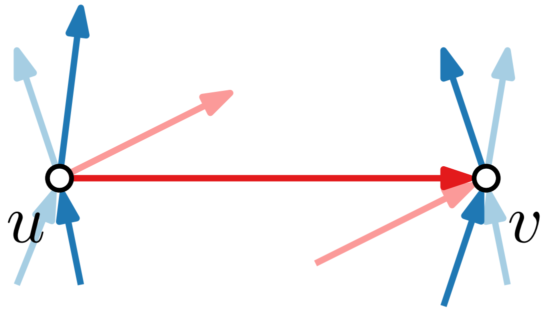
$y_1(v_W) = 0, y_2(v_W) = 9$
$y_1(v_E) = 1,\ y_2(v_E) = 10$
$y_1(v_N) = 9,\ y_2(v_N) = 10$
$y_1(v_S) = 0,\ y_2(v_S) = 1$
$y_1(a) = 1,\ y_2(a) = 2$
$y_1(b) = 1,\ y_2(b) = 2$
$\ldots$

Correctness of Algorithm (Sketch)

- If edge (u, v) exists, then $x_2(u) = x_1(v)$

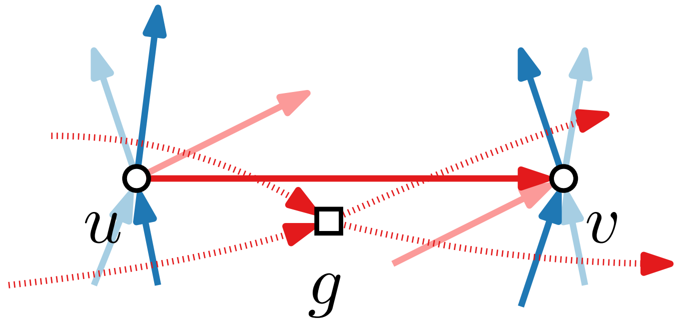
Correctness of Algorithm (Sketch)

- If edge (u, v) exists, then $x_2(u) = x_1(v)$



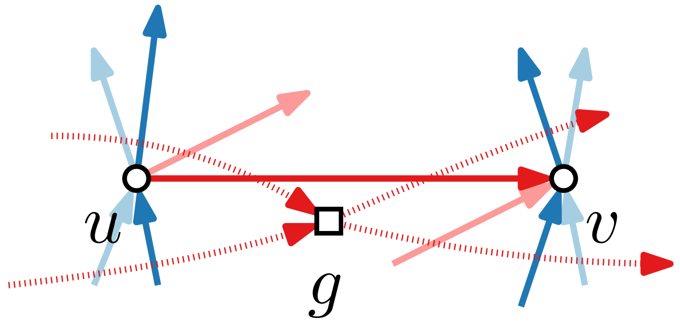
Correctness of Algorithm (Sketch)

- If edge (u, v) exists, then $x_2(u) = x_1(v)$



Correctness of Algorithm (Sketch)

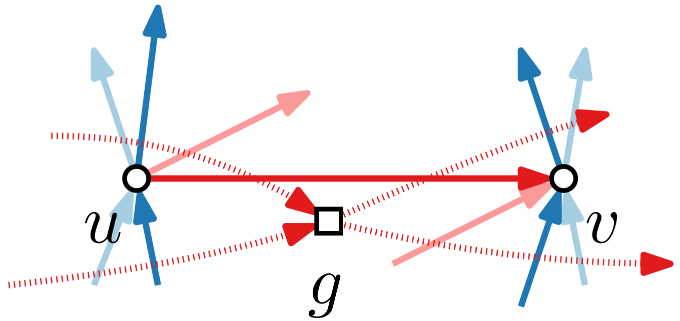
- If edge (u, v) exists, then $x_2(u) = x_1(v)$



$$x_2(u) = f_{\text{ver}}(g) = x_1(v)$$

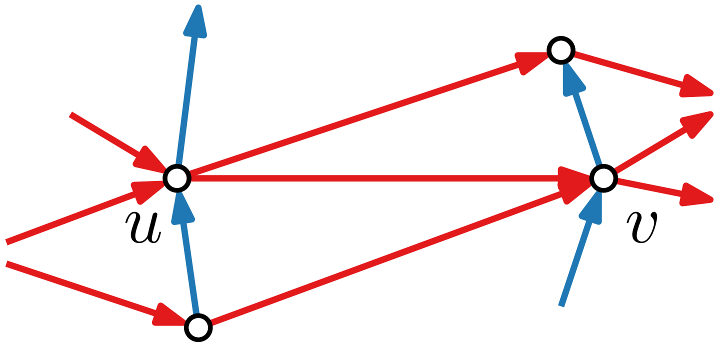
Correctness of Algorithm (Sketch)

- If edge (u, v) exists, then $x_2(u) = x_1(v)$



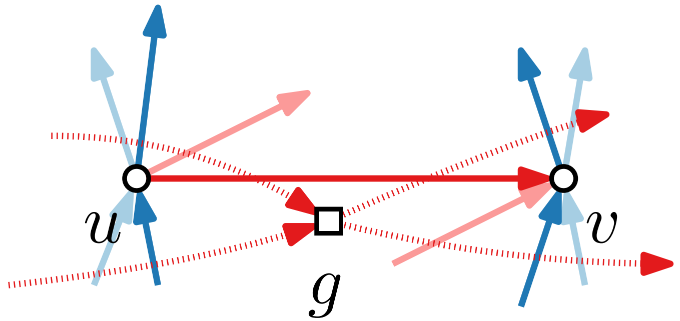
$$x_2(u) = f_{\text{ver}}(g) = x_1(v)$$

- and the vertical segments of their rectangles overlap



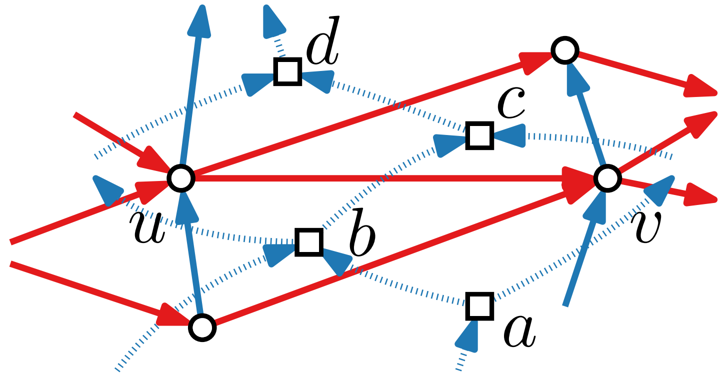
Correctness of Algorithm (Sketch)

- If edge (u, v) exists, then $x_2(u) = x_1(v)$



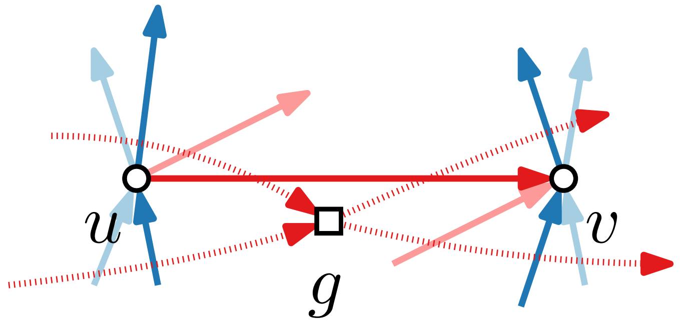
$$x_2(u) = f_{\text{ver}}(g) = x_1(v)$$

- and the vertical segments of their rectangles overlap



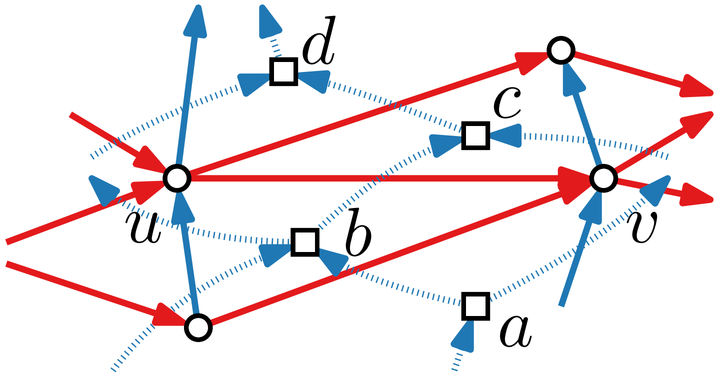
Correctness of Algorithm (Sketch)

- If edge (u, v) exists, then $x_2(u) = x_1(v)$



$$x_2(u) = f_{\text{ver}}(g) = x_1(v)$$

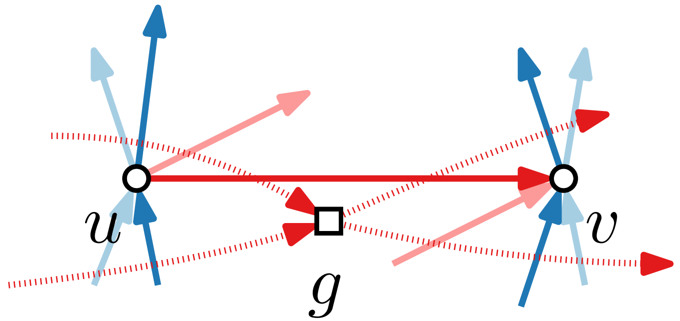
- and the vertical segments of their rectangles overlap



$$y_1(v) = f_{\text{hor}}(a)$$

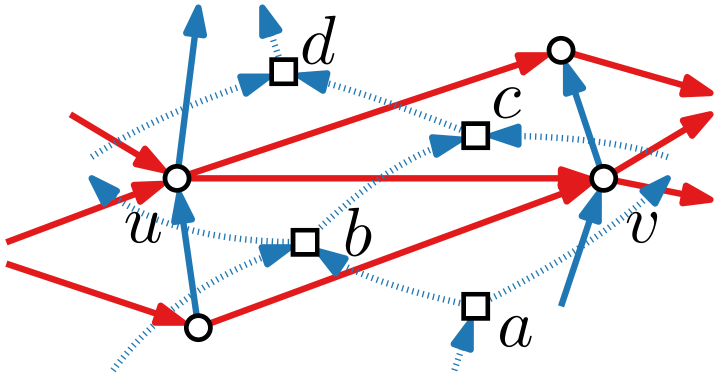
Correctness of Algorithm (Sketch)

- If edge (u, v) exists, then $x_2(u) = x_1(v)$



$$x_2(u) = f_{\text{ver}}(g) = x_1(v)$$

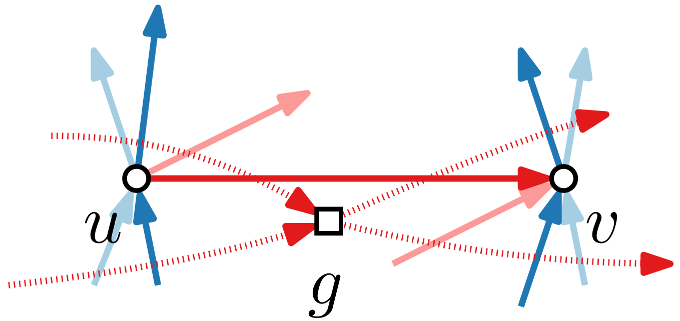
- and the vertical segments of their rectangles overlap



$$y_1(v) = f_{\text{hor}}(a) \leq y_1(u) = f_{\text{hor}}(b)$$

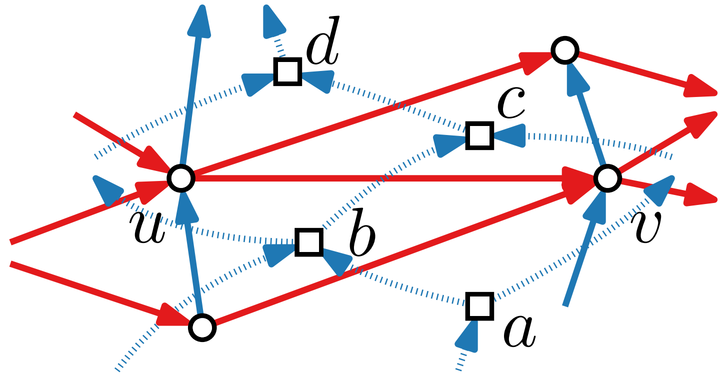
Correctness of Algorithm (Sketch)

- If edge (u, v) exists, then $x_2(u) = x_1(v)$



$$x_2(u) = f_{\text{ver}}(g) = x_1(v)$$

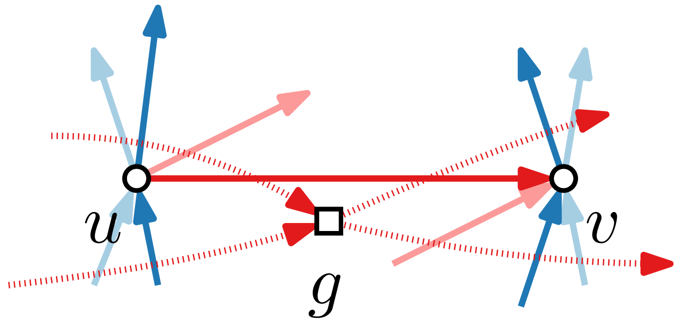
- and the vertical segments of their rectangles overlap



$$y_1(v) = f_{\text{hor}}(a) \leq y_1(u) = f_{\text{hor}}(b)$$
$$< y_2(v) = f_{\text{hor}}(c)$$

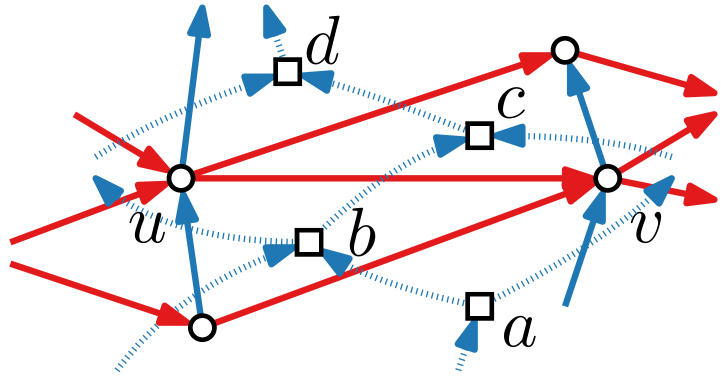
Correctness of Algorithm (Sketch)

- If edge (u, v) exists, then $x_2(u) = x_1(v)$



$$x_2(u) = f_{\text{ver}}(g) = x_1(v)$$

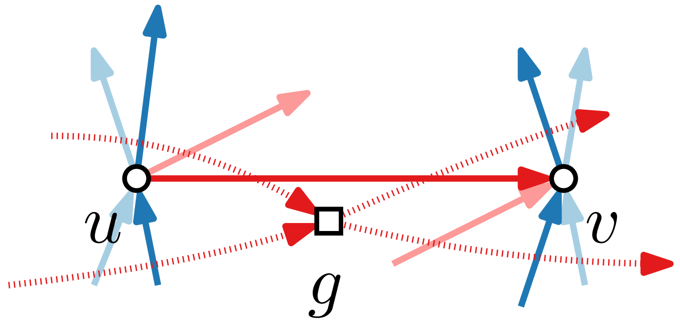
- and the vertical segments of their rectangles overlap



$$y_1(v) = f_{\text{hor}}(a) \leq y_1(u) = f_{\text{hor}}(b)$$
$$< y_2(v) = f_{\text{hor}}(c) \leq y_2(u) = f_{\text{hor}}(d)$$

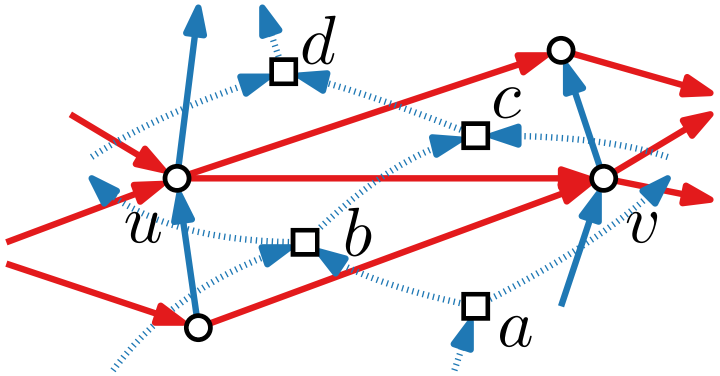
Correctness of Algorithm (Sketch)

- If edge (u, v) exists, then $x_2(u) = x_1(v)$



$$x_2(u) = f_{\text{ver}}(g) = x_1(v)$$

- and the vertical segments of their rectangles overlap

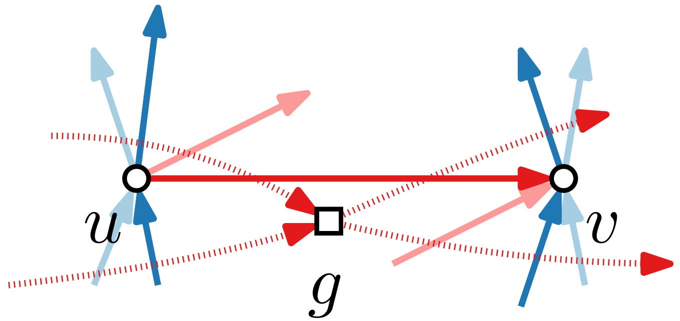


$$y_1(v) = f_{\text{hor}}(a) \leq y_1(u) = f_{\text{hor}}(b)$$
$$< y_2(v) = f_{\text{hor}}(c) \leq y_2(u) = f_{\text{hor}}(d)$$

- If path from u to v in red at least two edges long, then $x_2(u) < x_1(v)$.

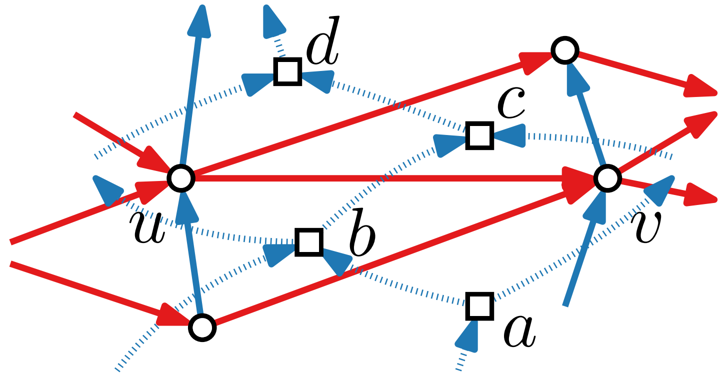
Correctness of Algorithm (Sketch)

- If edge (u, v) exists, then $x_2(u) = x_1(v)$



$$x_2(u) = f_{\text{ver}}(g) = x_1(v)$$

- and the vertical segments of their rectangles overlap

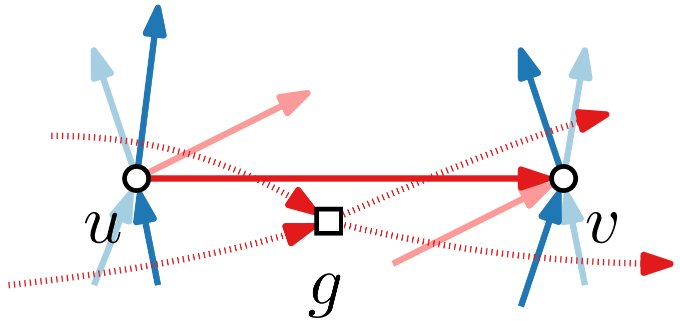


$$y_1(v) = f_{\text{hor}}(a) \leq y_1(u) = f_{\text{hor}}(b)$$
$$< y_2(v) = f_{\text{hor}}(c) \leq y_2(u) = f_{\text{hor}}(d)$$

- If path from u to v in red at least two edges long, then $x_2(u) < x_1(v)$.
- No two boxes overlap.

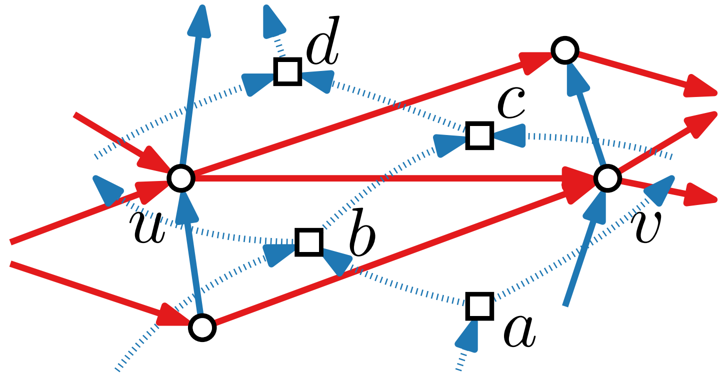
Correctness of Algorithm (Sketch)

- If edge (u, v) exists, then $x_2(u) = x_1(v)$



$$x_2(u) = f_{\text{ver}}(g) = x_1(v)$$

- and the vertical segments of their rectangles overlap



$$y_1(v) = f_{\text{hor}}(a) \leq y_1(u) = f_{\text{hor}}(b)$$
$$< y_2(v) = f_{\text{hor}}(c) \leq y_2(u) = f_{\text{hor}}(d)$$

- If path from u to v in red at least two edges long, then $x_2(u) < x_1(v)$.
- No two boxes overlap.
- For details, see He's paper [He '93].

Rectangular Dual Result

Theorem.
Every PTP graph G has a rectangular dual, which can be computed in linear time.

Rectangular Dual Result

Theorem.
Every PTP graph G has a rectangular dual, which can be computed in linear time.

Proof.

- Compute a planar embedding of G.

Rectangular Dual Result

Theorem.
Every PTP graph G has a rectangular dual, which can be computed in linear time.

Proof.

- Compute a planar embedding of G.
- Compute a refined canonical ordering of G.

Rectangular Dual Result

Theorem.
Every PTP graph G has a rectangular dual, which can be computed in linear time.

Proof.

- Compute a planar embedding of G.
- Compute a refined canonical ordering of G.
- Traverse the graph and color the edges. $\rightarrow$ REL

Rectangular Dual Result

Theorem.
Every PTP graph G has a rectangular dual, which can be computed in linear time.

Proof.

- Compute a planar embedding of G.
- Compute a refined canonical ordering of G.
- Traverse the graph and color the edges. $\rightarrow$ REL
- Construct G_{ver} and G_{hor}.

Rectangular Dual Result

Theorem.
Every PTP graph G has a rectangular dual, which can be computed in linear time.

Proof.

- Compute a planar embedding of G.
- Compute a refined canonical ordering of G.
- Traverse the graph and color the edges. $\rightarrow$ REL
- Construct G_{ver} and G_{hor}.
- Construct their duals $G^{\star}_{\mathsf{ver}}$ and $G^{\star}_{\mathsf{hor}}$.

Rectangular Dual Result

Theorem.
Every PTP graph G has a rectangular dual, which can be computed in linear time.

Proof.

- Compute a planar embedding of G.
- Compute a refined canonical ordering of G.
- Traverse the graph and color the edges. $\rightarrow$ REL
- Construct G_{ver} and G_{hor}.
- Construct their duals $G^\star_{\text{ver}}$ and $G^\star_{\text{hor}}$.
- Compute a topological ordering for vertices of $G^\star_{\text{ver}}$ and $G^\star_{\text{hor}}$.

Rectangular Dual Result

Theorem.
Every PTP graph G has a rectangular dual, which can be computed in linear time.

Proof.

- Compute a planar embedding of G.
- Compute a refined canonical ordering of G.
- Traverse the graph and color the edges. $\rightarrow$ REL
- Construct G_{ver} and G_{hor}.
- Construct their duals $G^\star_{\text{ver}}$ and $G^\star_{\text{hor}}$.
- Compute a topological ordering for vertices of $G^\star_{\text{ver}}$ and $G^\star_{\text{hor}}$.
- Assing coordinates to the rectangles representing vertices.

Discussion

- A layout is **area-universal** if any assignment of areas to rectangles can be realized by a combinatorially equivalent rectangular layout.

Discussion

- A layout is **area-universal** if any assignment of areas to rectangles can be realized by a combinatorially equivalent rectangular layout.
- A rectangular layout is **area-universal** if and only if it is **one-sided**.
  [Eppstein et al. SIAM J. Comp. 2012]

Discussion

- A layout is **area-universal** if any assignment of areas to rectangles can be realized by a combinatorially equivalent rectangular layout.
- A rectangular layout is **area-universal** if and only if it is **one-sided**. [Eppstein et al. SIAM J. Comp. 2012]

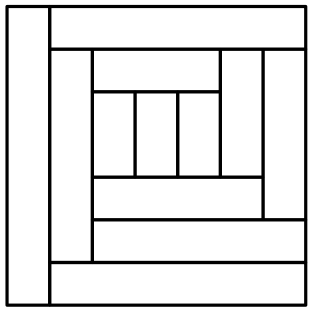
i.e., every segment belongs to exactly one rectangle

Discussion

- A layout is **area-universal** if any assignment of areas to rectangles can be realized by a combinatorially equivalent rectangular layout.
- A rectangular layout is **area-universal** if and only if it is **one-sided**. [Eppstein et al. SIAM J. Comp. 2012]

i.e., every segment belongs to exactly one rectangle

one-sided

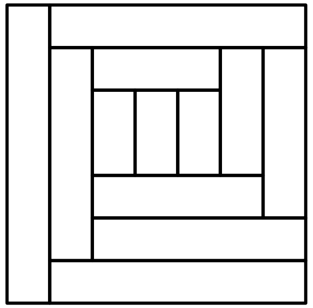


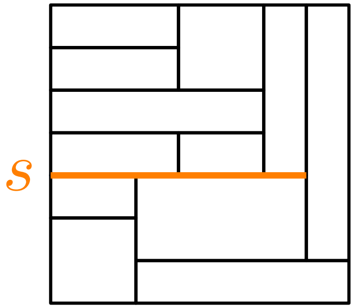
Discussion

- A layout is **area-universal** if any assignment of areas to rectangles can be realized by a combinatorially equivalent rectangular layout.
- A rectangular layout is **area-universal** if and only if it is **one-sided**.
  [Eppstein et al. SIAM J. Comp. 2012]

i.e., every segment belongs to exactly one rectangle

one-sided





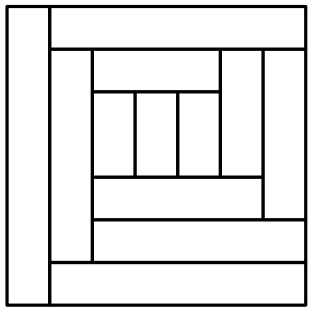
not one-sided

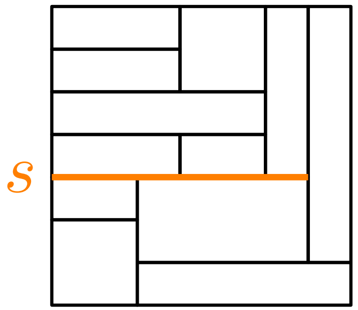
Discussion

- A layout is **area-universal** if any assignment of areas to rectangles can be realized by a combinatorially equivalent rectangular layout.
- A rectangular layout is **area-universal** if and only if it is **one-sided**.
  [Eppstein et al. SIAM J. Comp. 2012]

i.e., every segment belongs to exactly one rectangle

one-sided





not one-sided

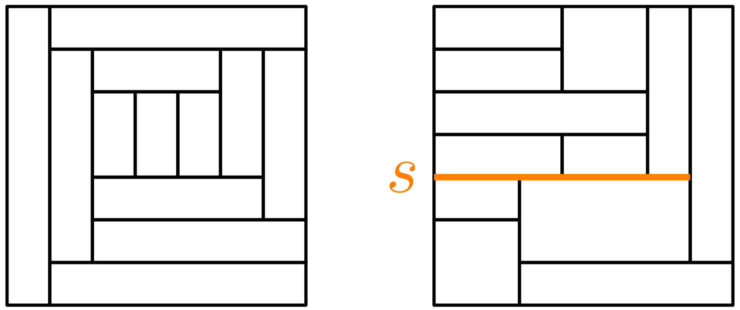
- Area-universal **rectlinear** representation: possible for all planar graphs.

Discussion

- A layout is **area-universal** if any assignment of areas to rectangles can be realized by a combinatorially equivalent rectangular layout.
- A rectangular layout is **area-universal** if and only if it is **one-sided**.
  [Eppstein et al. SIAM J. Comp. 2012]

i.e., every segment belongs to exactly one rectangle

one-sided



not one-sided

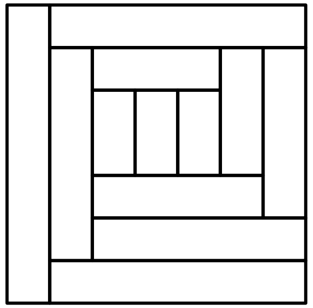
- Area-universal **rectlinear** representation: possible for all planar graphs.
- [Alam et al. 2013]: 8 sides (matches the lower bound)

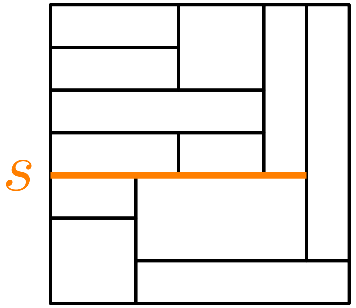
Discussion

- A layout is **area-universal** if any assignment of areas to rectangles can be realized by a combinatorially equivalent rectangular layout.
- A rectangular layout is **area-universal** if and only if it is **one-sided**.
  [Eppstein et al. SIAM J. Comp. 2012]

i.e., every segment belongs to exactly one rectangle

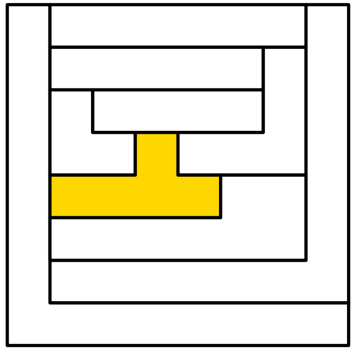
one-sided





not one-sided

- Area-universal **rectlinear** representation: possible for all planar graphs.
- [Alam et al. 2013]: 8 sides (matches the lower bound)



Literature

Construction of triangle contact representations based on

- [de Fraysseix, Ossona de Mendez, Rosenstiehl ’94] On Triangle Contact Graphs

Construction of rectangular dual based on

- [He ’93] On Finding the Rectangular Duals of Planar Triangulated Graphs
- [Kant, He ’94] Two algorithms for finding rectangular duals of planar graphs

and originally from

- [Koźmiński, Kinnen ’85] Rectangular Duals of Planar Graphs