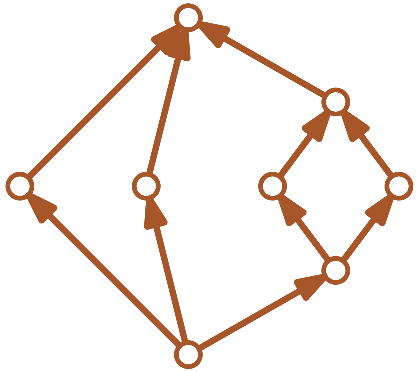
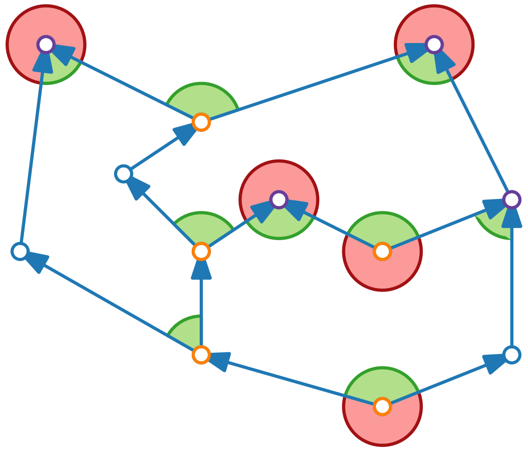
Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

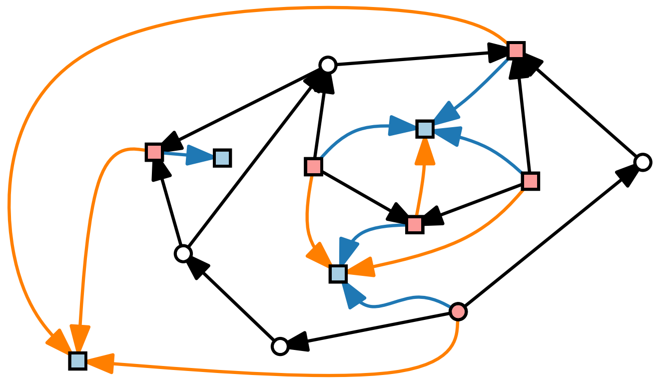
Visualization of Graphs



Lecture 5:
Upward Planar Drawings



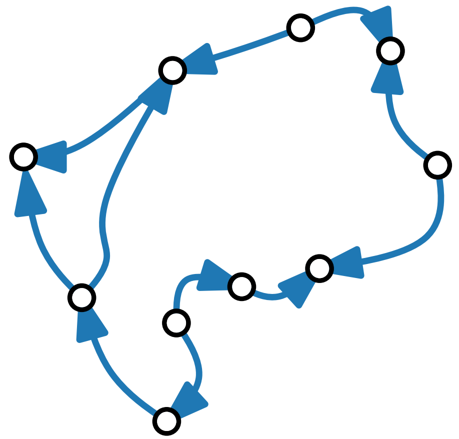
Part I:
Recognition



Johannes Zink

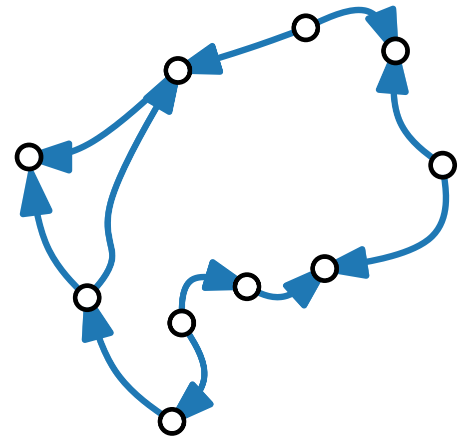
Summer semester 2024

Upward Planar Drawings – Motivation

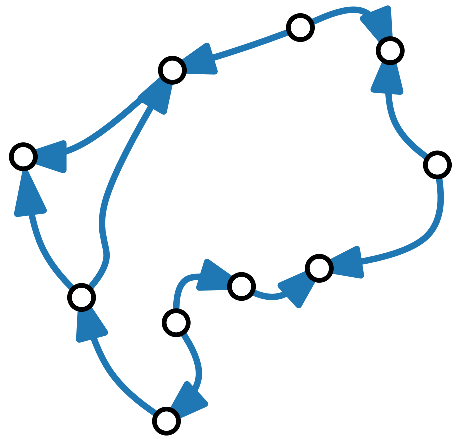


Upward Planar Drawings – Motivation

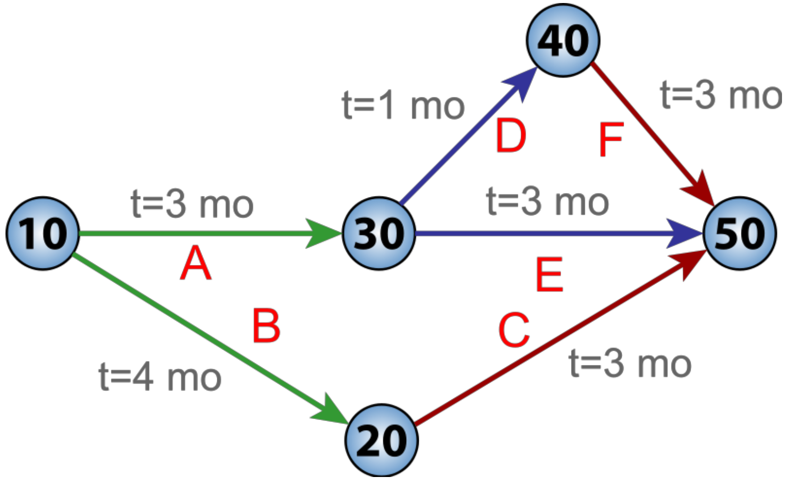
- What may the direction of edges in a directed graph represent?



Upward Planar Drawings – Motivation



- What may the direction of edges in a directed graph represent?
  - Time

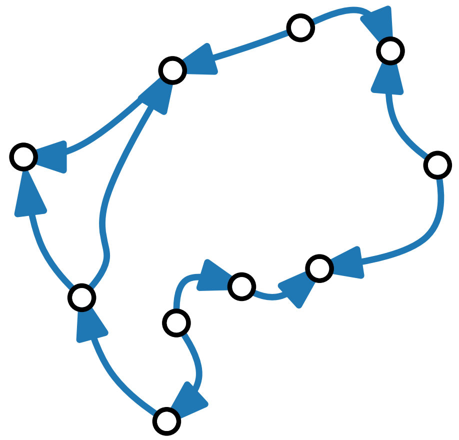


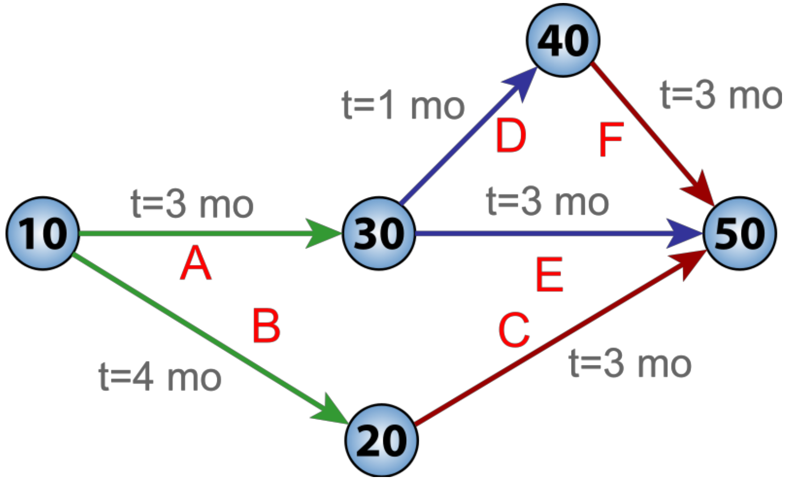
PERT diagram

Program Evaluation and Review Technique
(Project management)

Upward Planar Drawings – Motivation

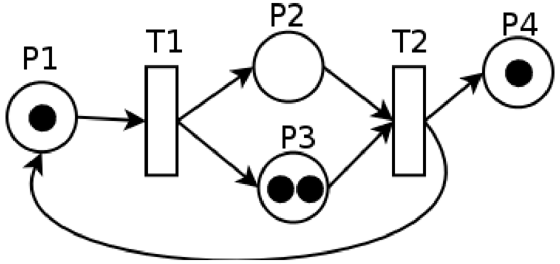
- What may the direction of edges in a directed graph represent?
  - Time
  - Flow





PERT diagram

Program Evaluation and Review Technique
(Project management)

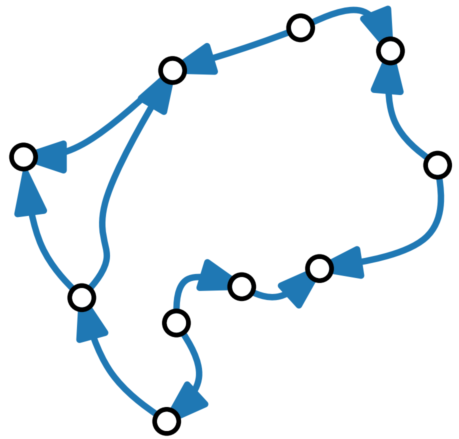


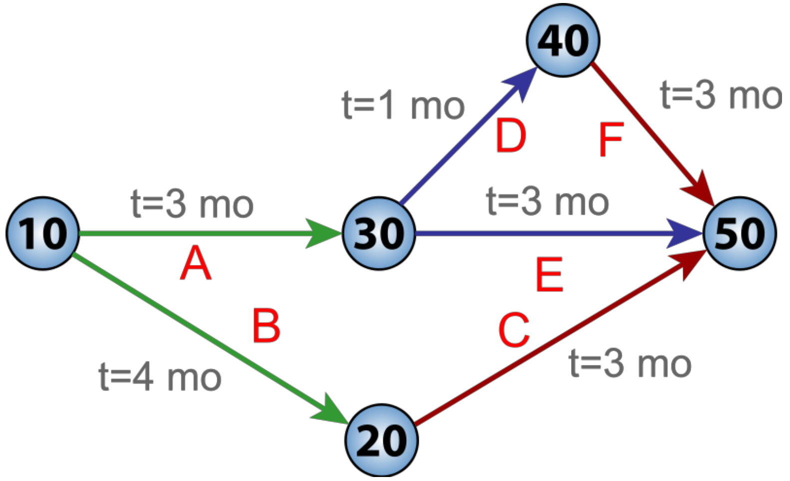
Petri net

Place/Transition net
(Modeling languages for distributed systems)

Upward Planar Drawings – Motivation

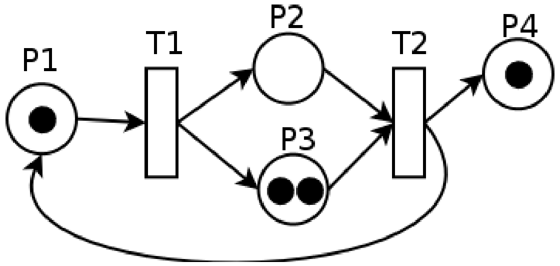
- What may the direction of edges in a directed graph represent?
  - Time
  - Flow
  - Hierarchy





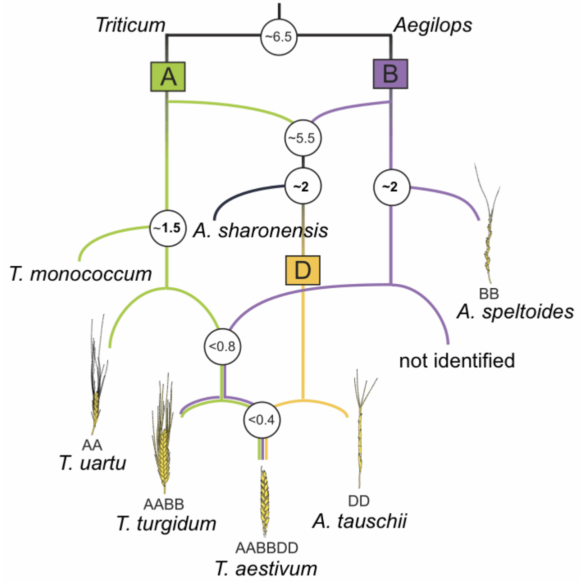
PERT diagram

Program Evaluation and Review Technique
(Project management)



Petri net

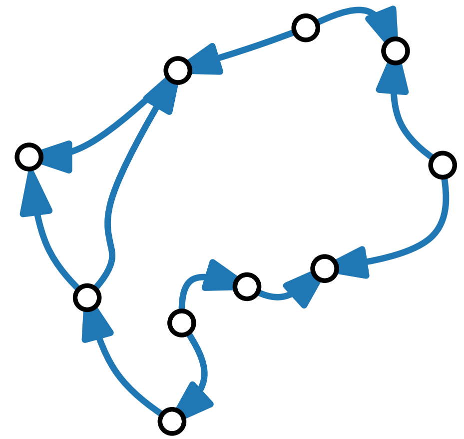
Place/Transition net
(Modeling languages for distributed systems)



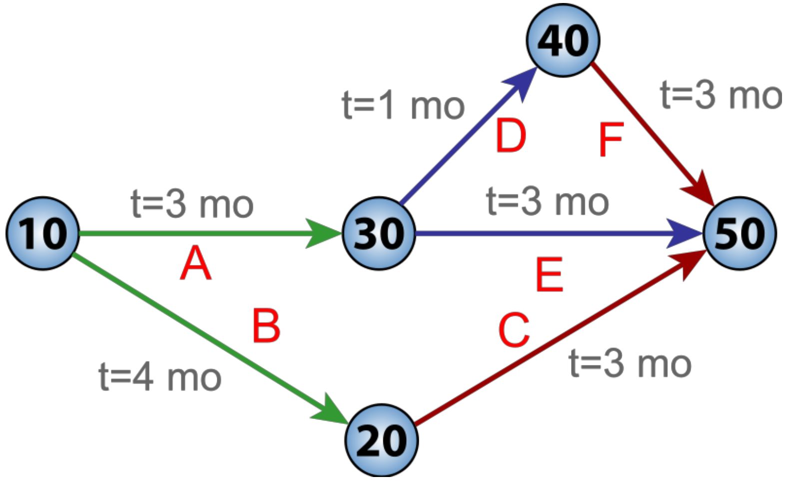
Phylogenetic network

Ancestral trees / networks
(Biology)

Upward Planar Drawings – Motivation

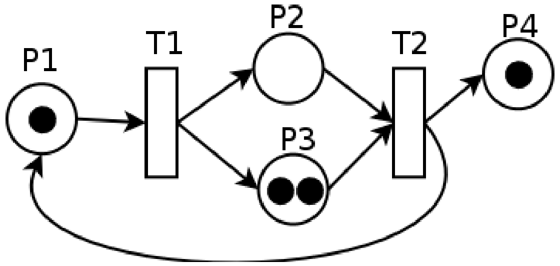


- What may the direction of edges in a directed graph represent?
  - Time
  - Flow
  - Hierarchy
  - ...



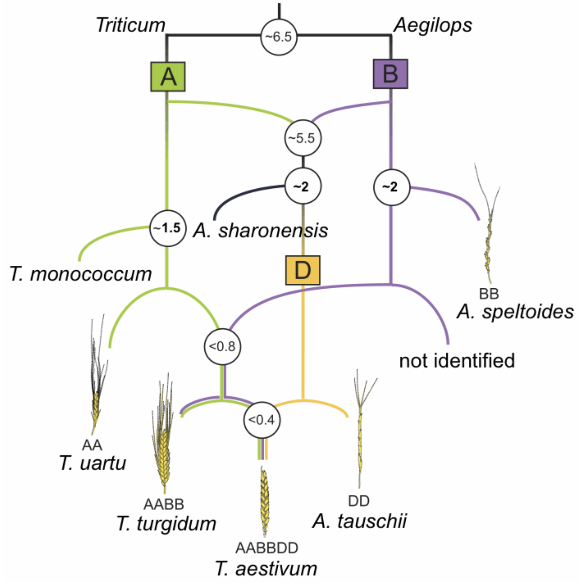
PERT diagram

Program Evaluation and Review Technique
(Project management)



Petri net

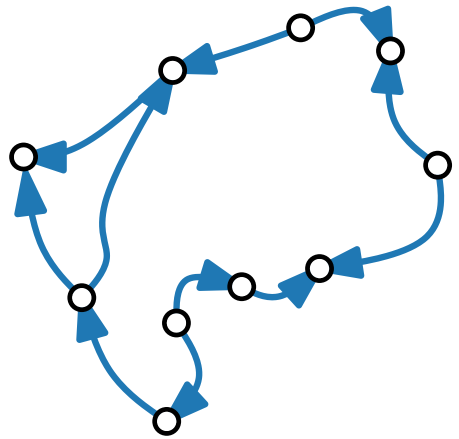
Place/Transition net
(Modeling languages for distributed systems)



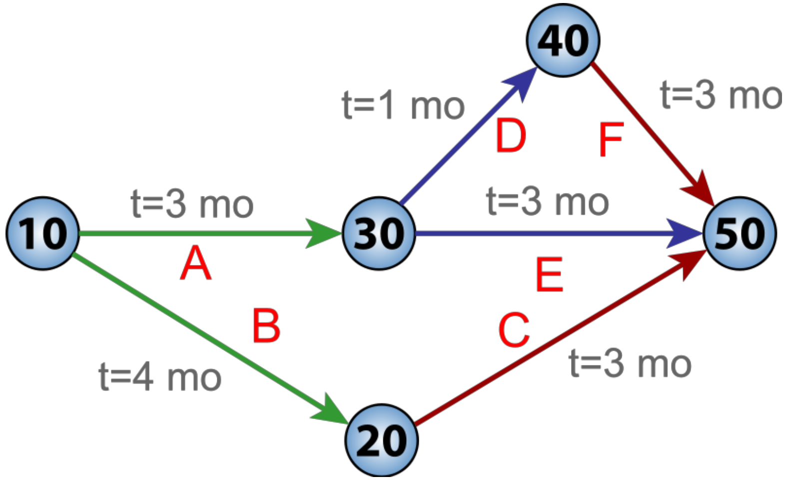
Phylogenetic network

Ancestral trees / networks
(Biology)

Upward Planar Drawings – Motivation

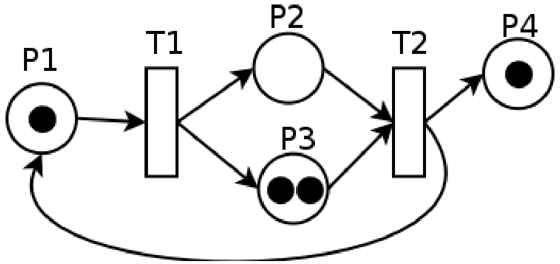


- What may the direction of edges in a directed graph represent?
  - Time
  - Flow
  - Hierarchy
  - . . .
- We aim for drawings where the general direction is preserved.



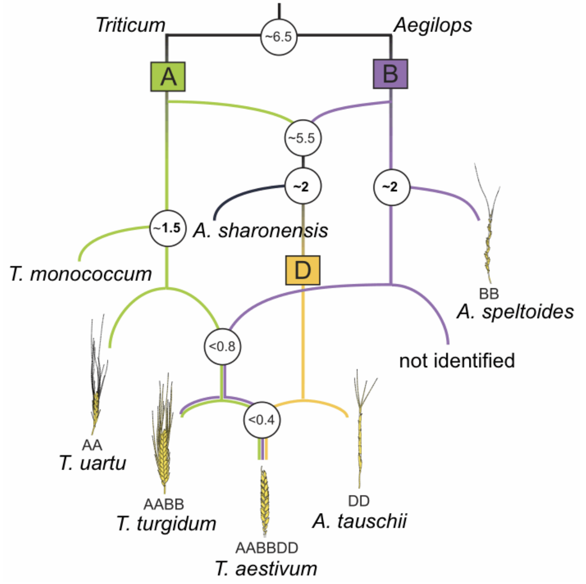
PERT diagram

Program Evaluation and Review Technique
(Project management)



Petri net

Place/Transition net
(Modeling languages for distributed systems)

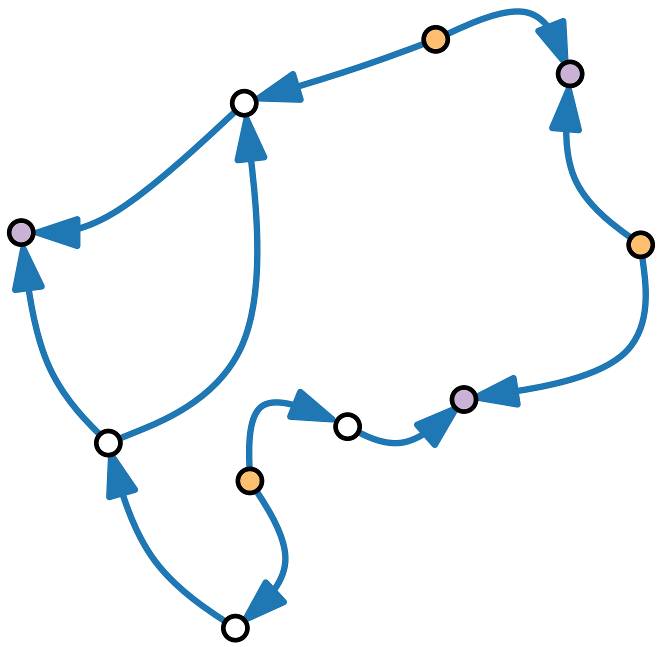


Phylogenetic network

Ancestral trees / networks
(Biology)

Upward Planar Drawings – Definition

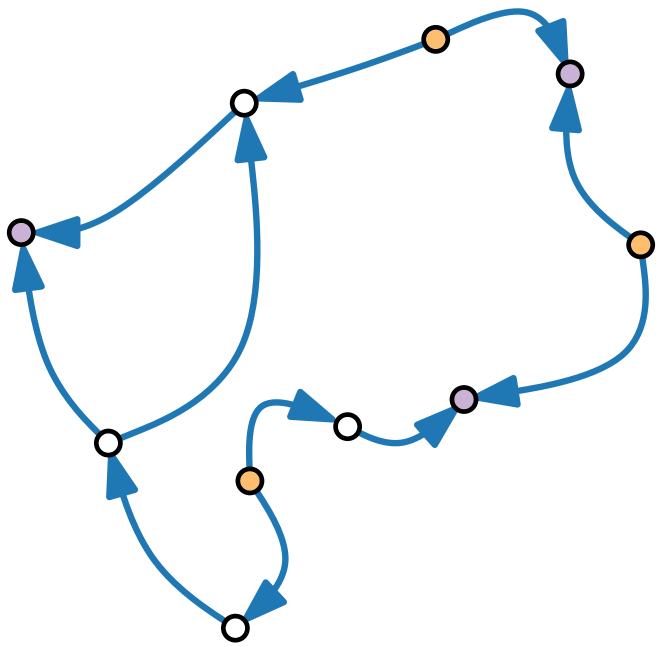
A directed graph (*digraph*) is **upward planar** when it admits a drawing



Upward Planar Drawings – Definition

A directed graph (*digraph*) is **upward planar** when it admits a drawing

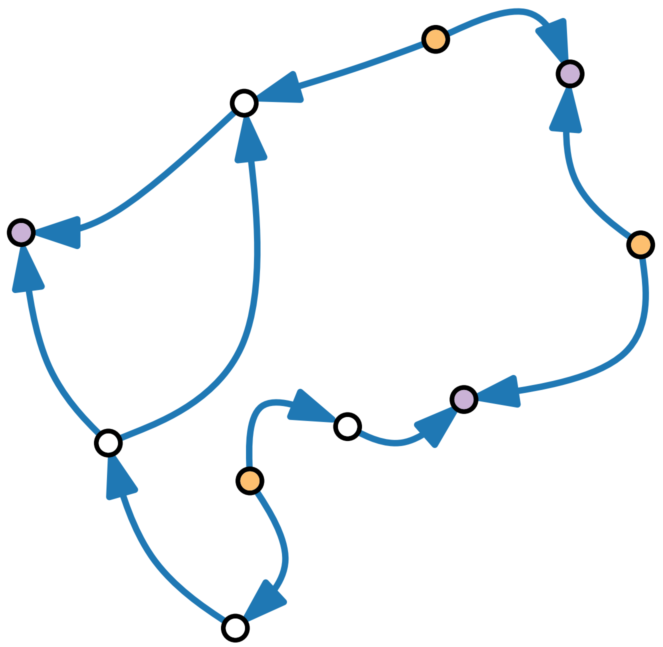
- that is planar



Upward Planar Drawings – Definition

A directed graph (*digraph*) is **upward planar** when it admits a drawing

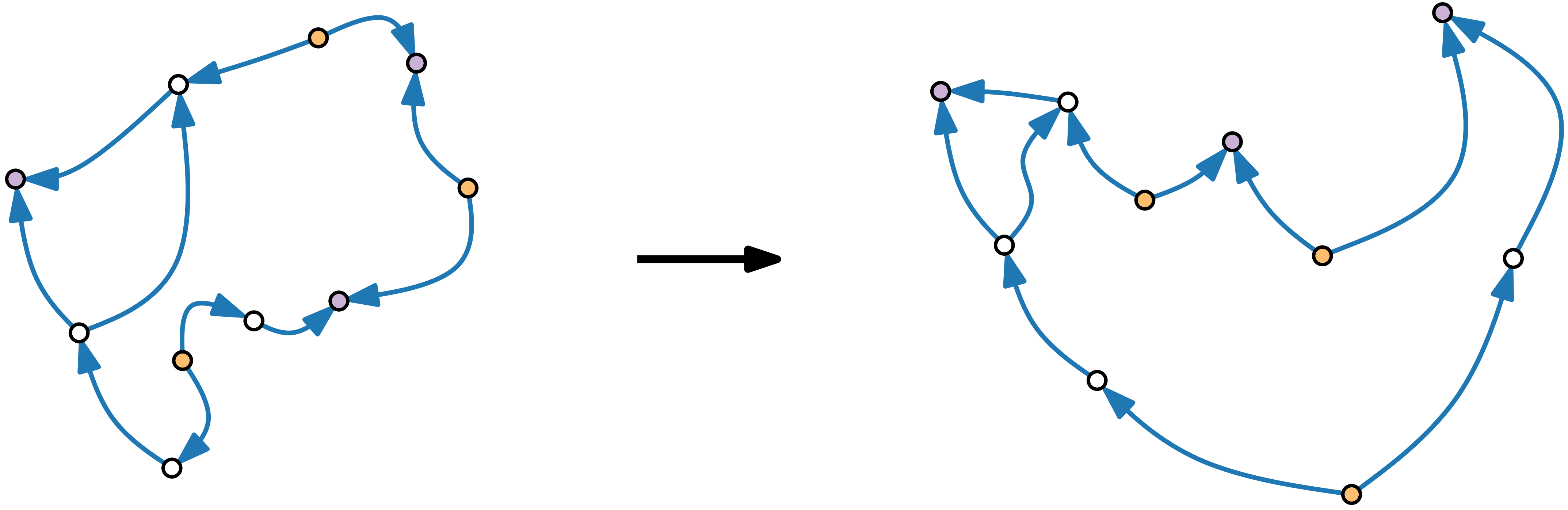
- that is planar and
- where each edge is drawn as an upward y-monotone curve.



Upward Planar Drawings – Definition

A directed graph (*digraph*) is **upward planar** when it admits a drawing

- that is planar and
- where each edge is drawn as an upward y-monotone curve.

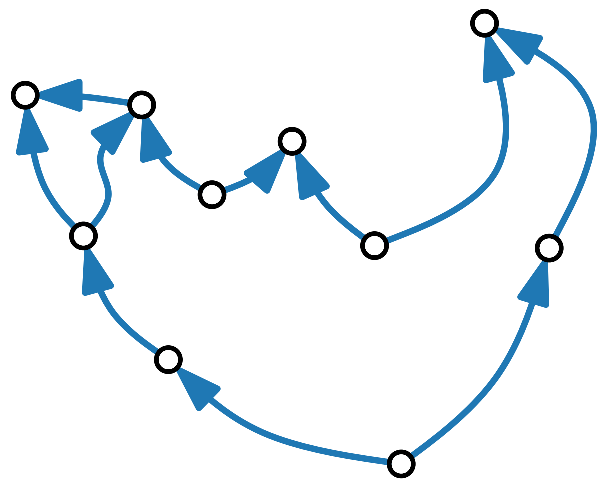


Upward Planarity – Necessary Conditions

- For an (embedded) digraph to be upward planar, it needs to . . .

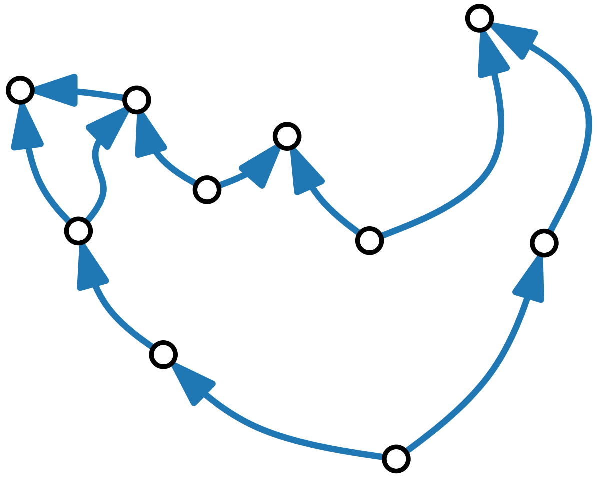
Upward Planarity – Necessary Conditions

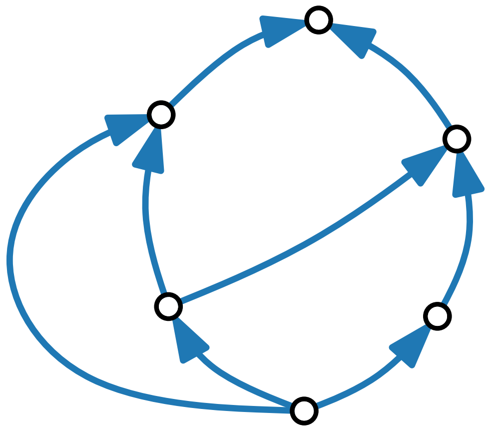
- For an (embedded) digraph to be upward planar, it needs to . . .
  - be planar



Upward Planarity – Necessary Conditions

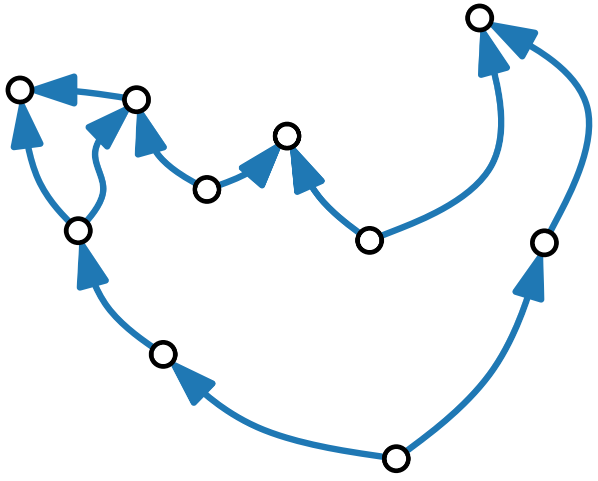
- For an (embedded) digraph to be upward planar, it needs to ...
  - be planar
  - be acyclic

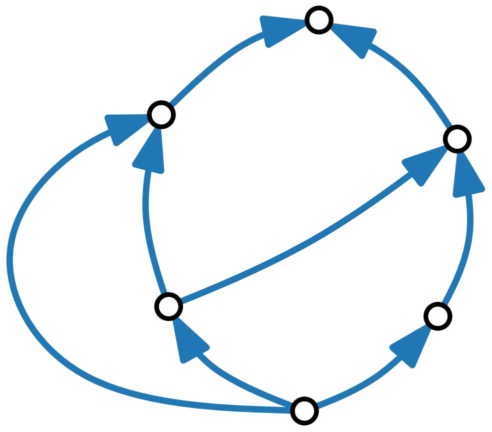


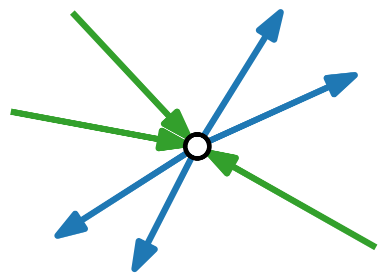


Upward Planarity – Necessary Conditions

- For an (embedded) digraph to be upward planar, it needs to ...
  - be planar
  - be acyclic

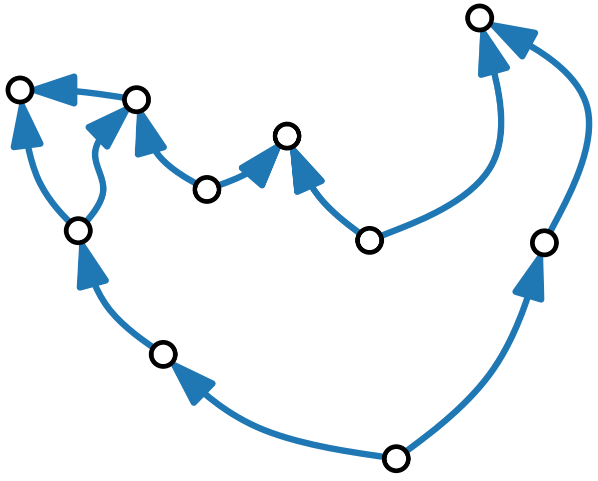


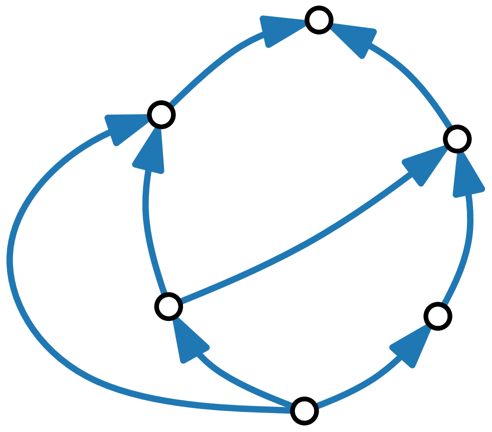


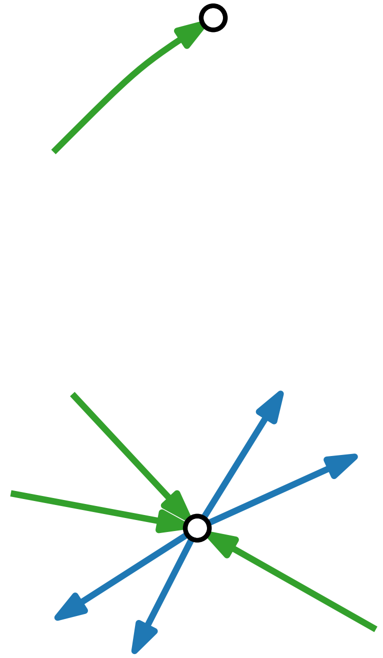


Upward Planarity – Necessary Conditions

- For an (embedded) digraph to be upward planar, it needs to . . .
  - be planar
  - be acyclic

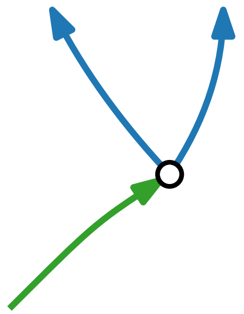


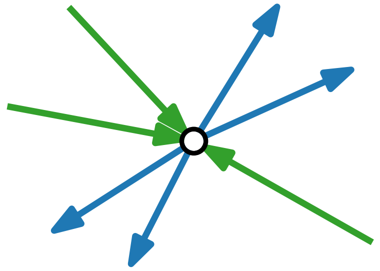


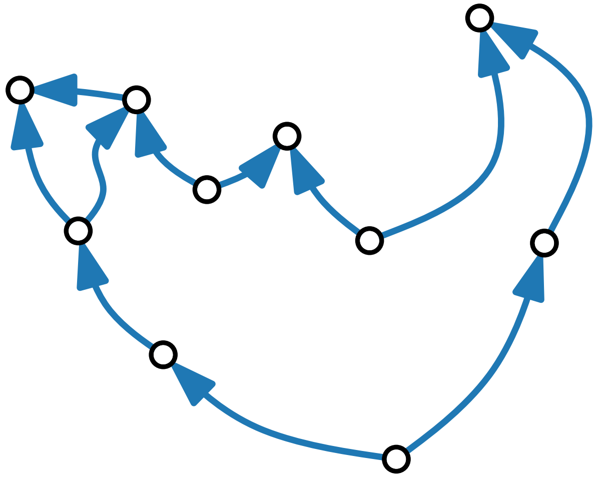


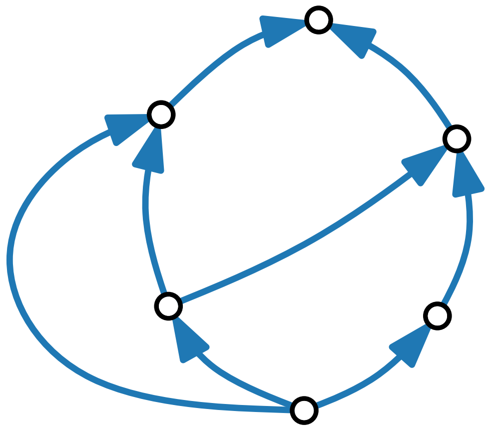
Upward Planarity – Necessary Conditions

- For an (embedded) digraph to be upward planar, it needs to ...
  - be planar
  - be acyclic



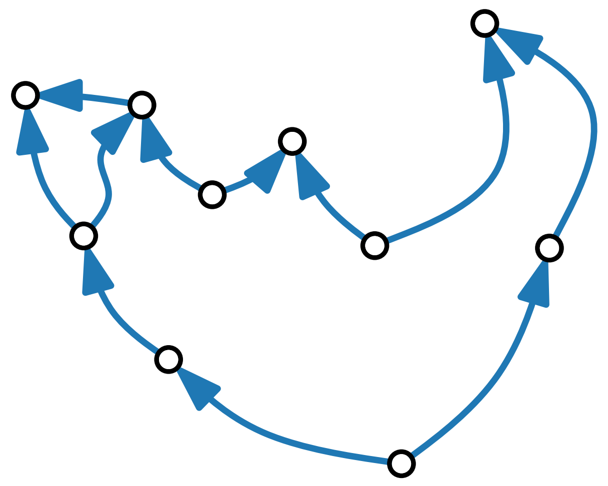


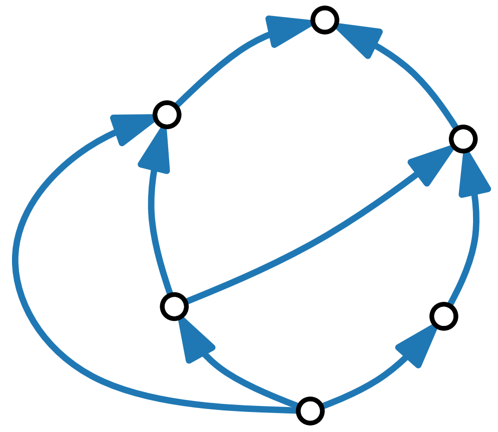


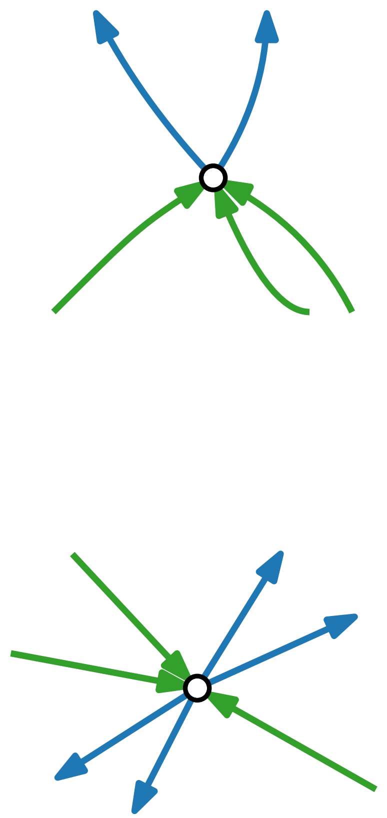


Upward Planarity – Necessary Conditions

- For an (embedded) digraph to be upward planar, it needs to ...
  - be planar
  - be acyclic

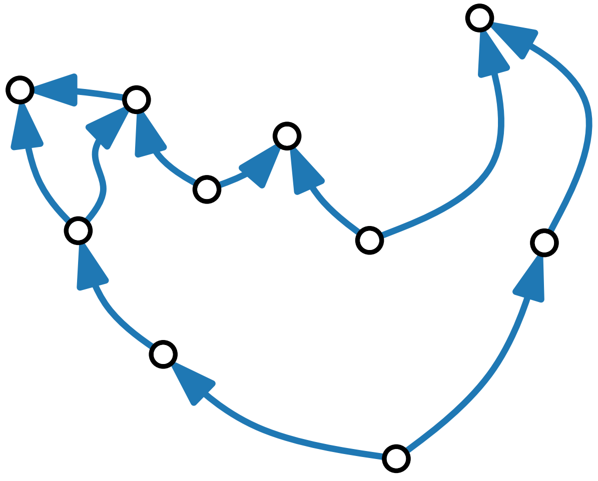


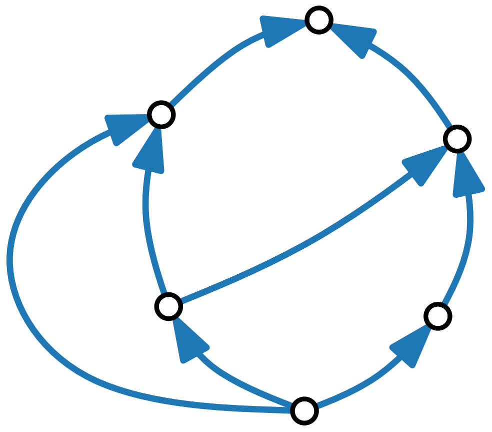


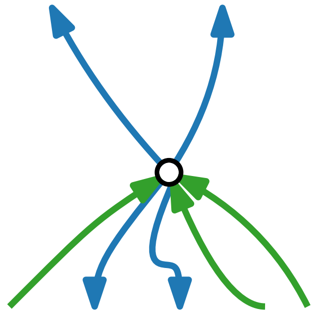


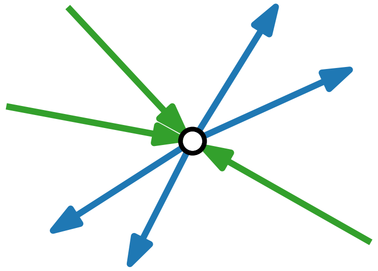
Upward Planarity – Necessary Conditions

- For an (embedded) digraph to be upward planar, it needs to . . .
  - be planar
  - be acyclic



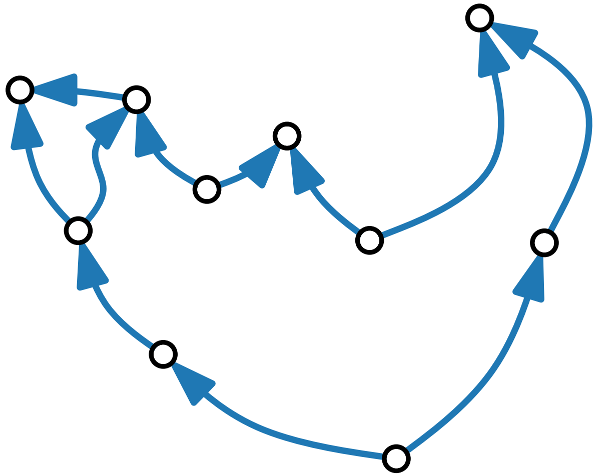


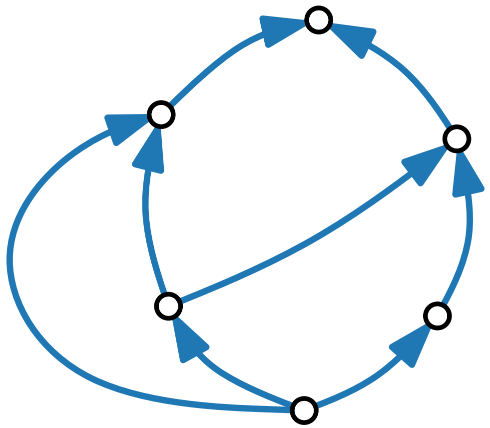


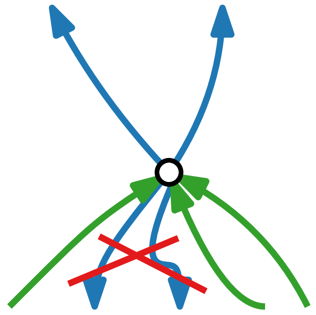


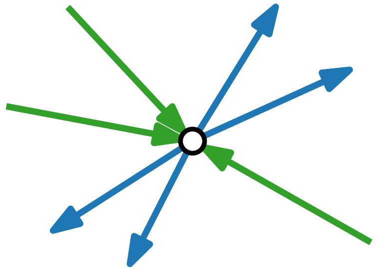
Upward Planarity – Necessary Conditions

- For an (embedded) digraph to be upward planar, it needs to ...
  - be planar
  - be acyclic



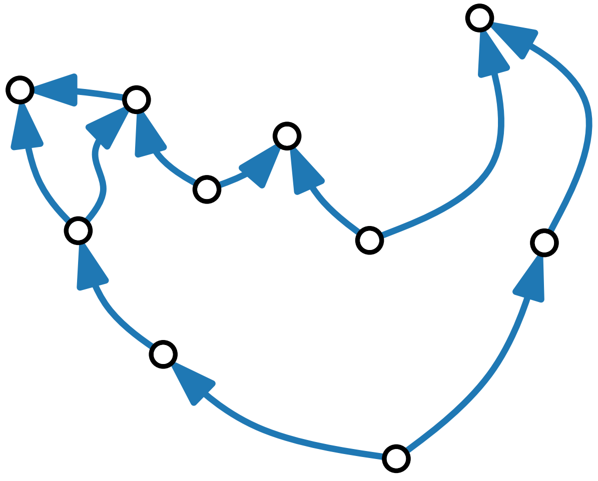


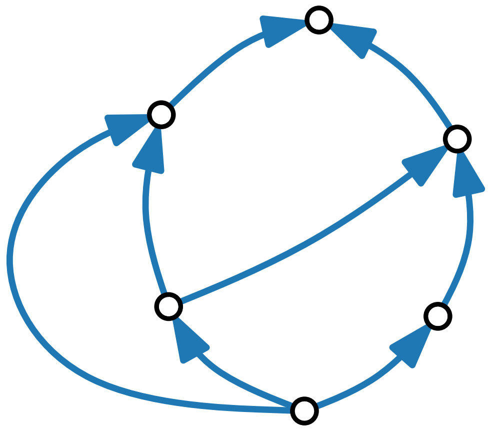


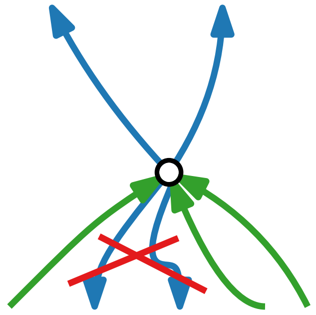


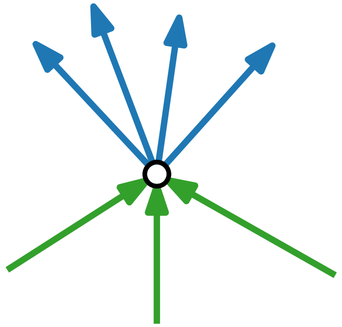
Upward Planarity – Necessary Conditions

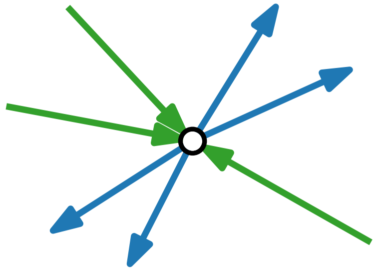
- For an (embedded) digraph to be upward planar, it needs to ...
  - be planar
  - be acyclic





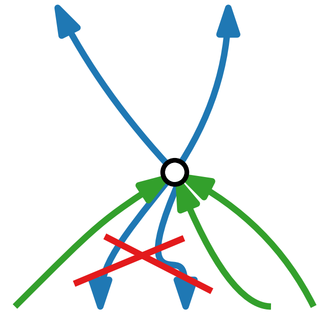


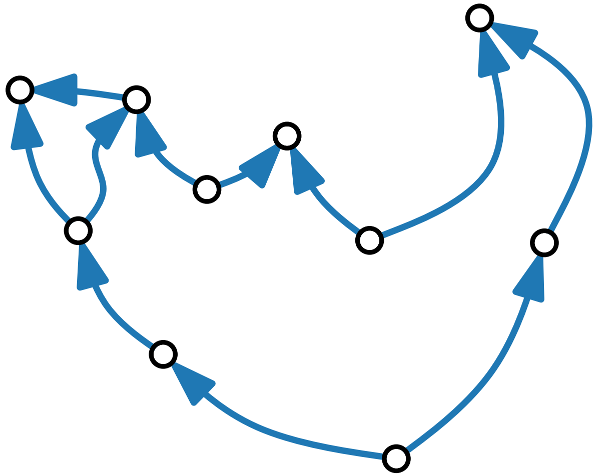


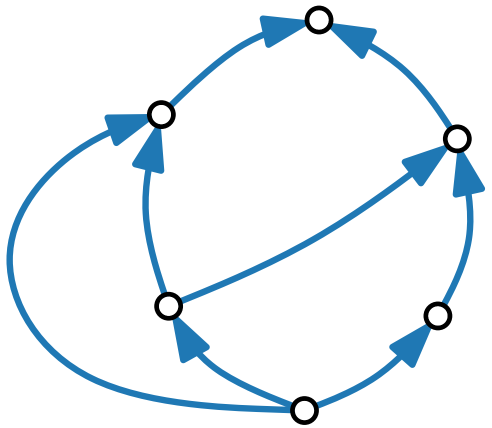


Upward Planarity – Necessary Conditions

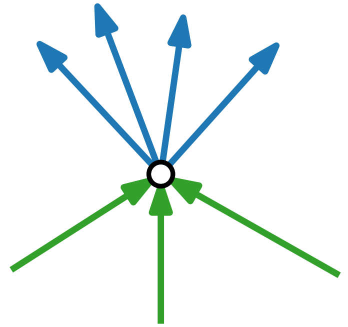
- For an (embedded) digraph to be upward planar, it needs to ...
  - be planar
  - be acyclic



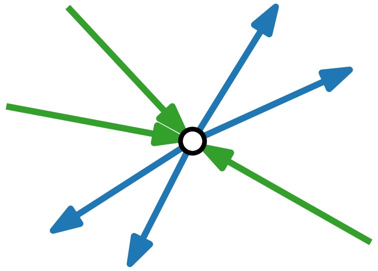




bimodal vertex

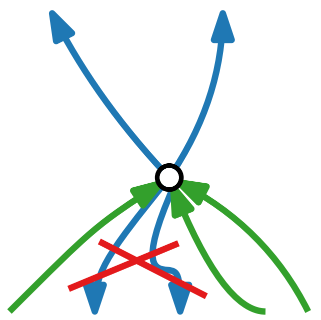


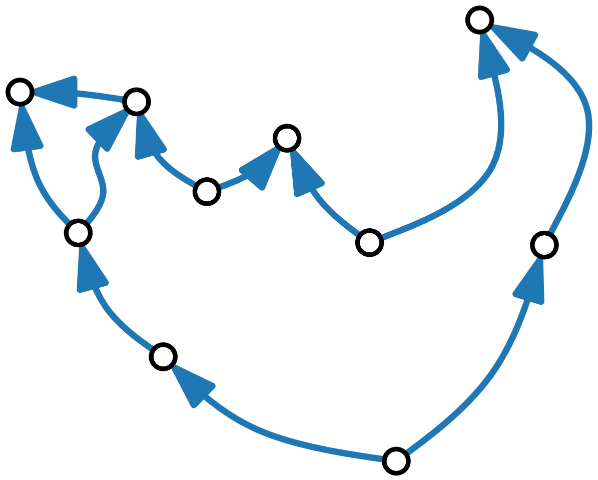
not bimodal

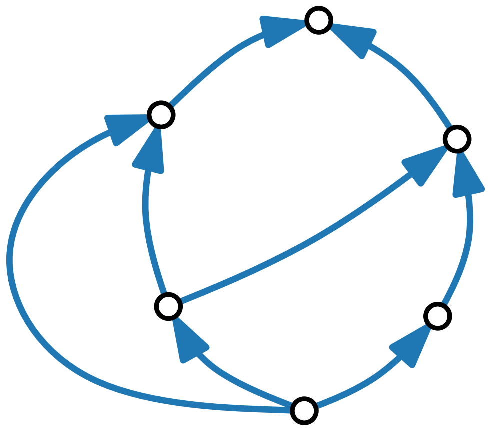


Upward Planarity – Necessary Conditions

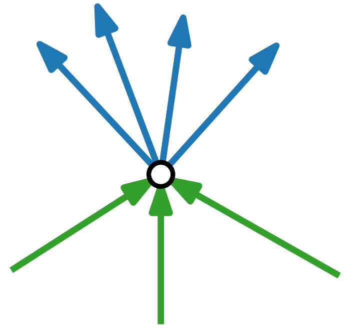
- For an (embedded) digraph to be upward planar, it needs to ...
  - be planar
  - be acyclic
  - have a bimodal embedding



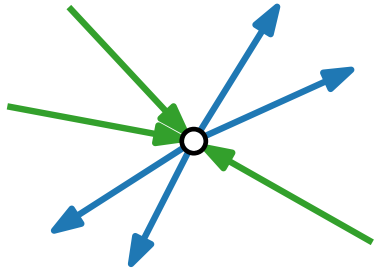




bimodal vertex

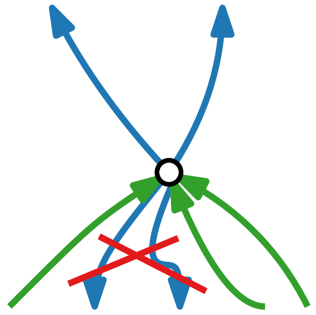


not bimodal

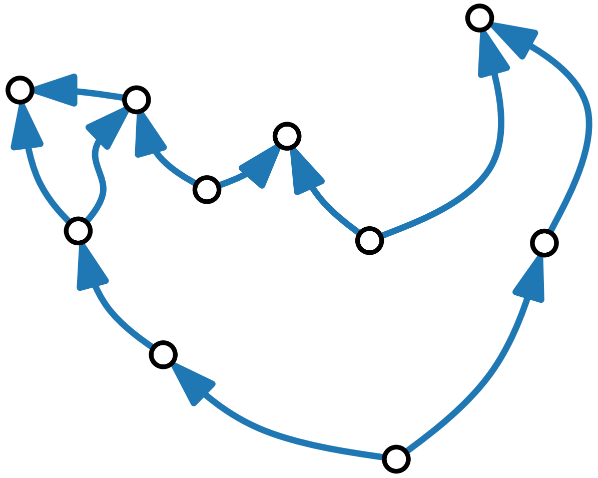


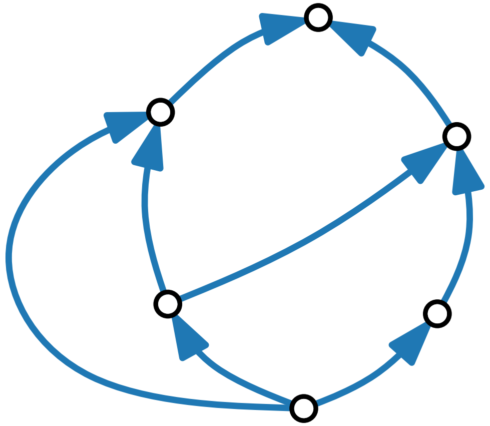
Upward Planarity – Necessary Conditions

- For an (embedded) digraph to be upward planar, it needs to . . .
  - be planar
  - be acyclic
  - have a bimodal embedding

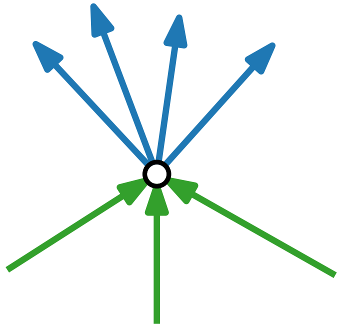


- . . . but these conditions are *not sufficient*.

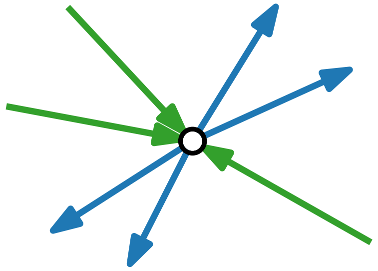




bimodal vertex

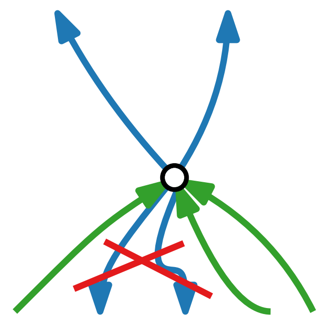


not bimodal

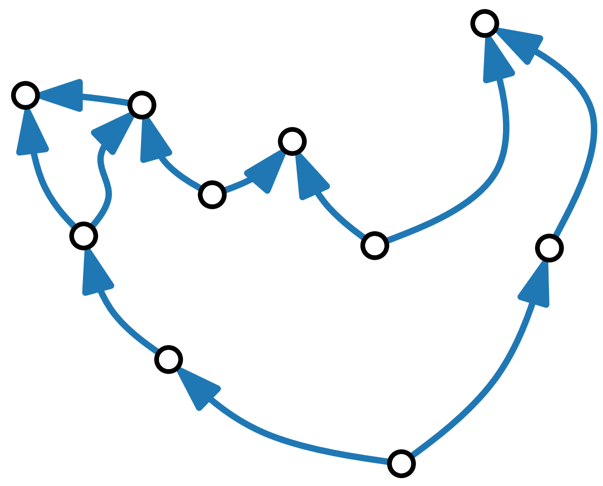


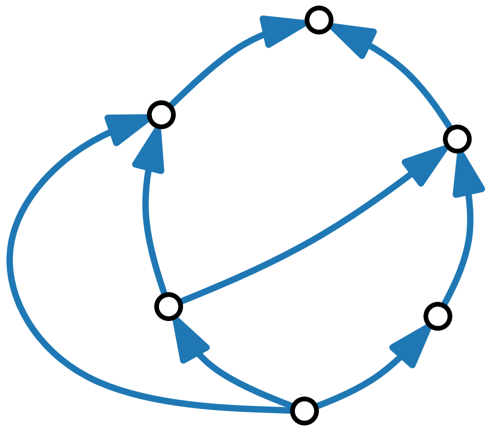
Upward Planarity – Necessary Conditions

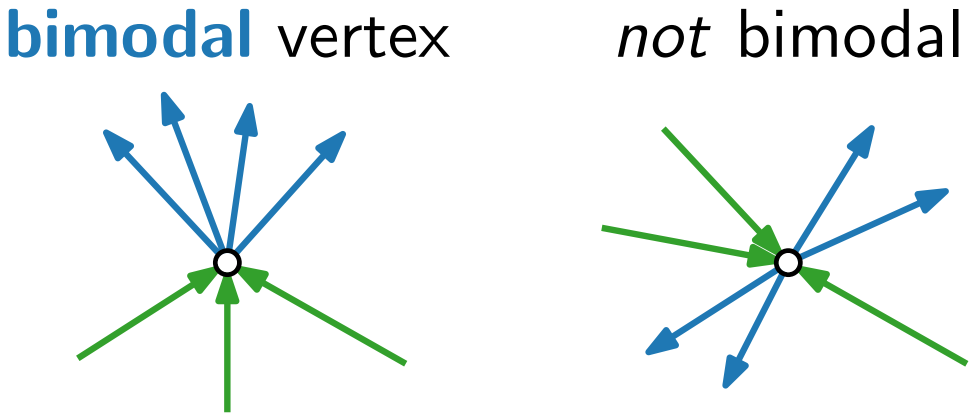
- For an (embedded) digraph to be upward planar, it needs to ...
  - be planar
  - be acyclic
  - have a bimodal embedding



- ... but these conditions are *not sufficient*. → **Exercise**







Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:

(1) G is upward planar.

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:

(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

no crossings

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]

For a digraph G, the following statements are equivalent:

(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

no crossings

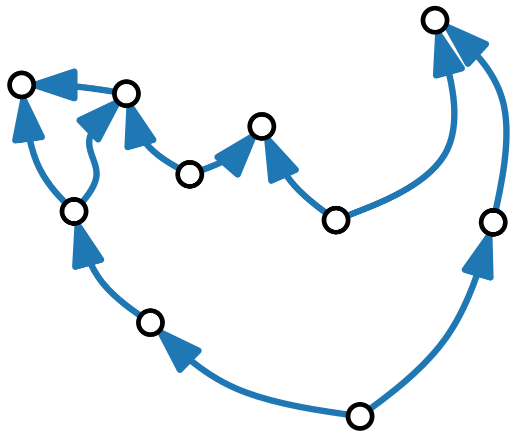
acyclic digraph with
a single source s and a single sink t

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

no crossings

acyclic digraph with
a single source s and a single sink t

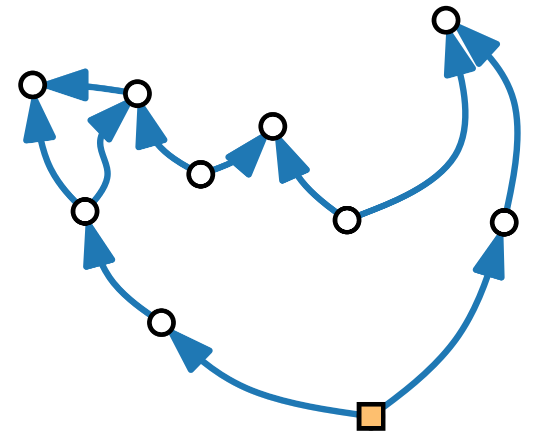


Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

no crossings

acyclic digraph with
a single source s and a single sink t

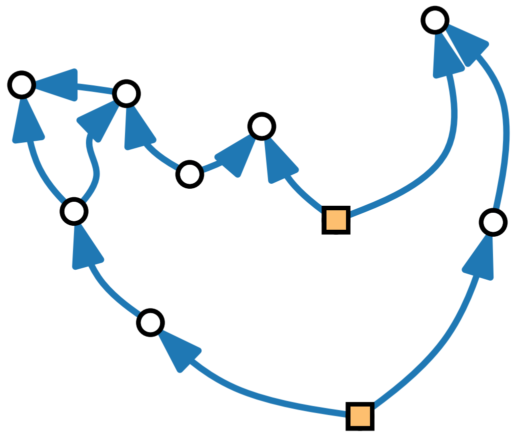


Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

no crossings

acyclic digraph with
a single source s and a single sink t

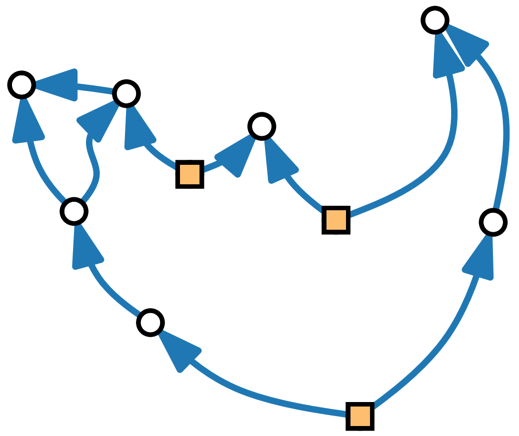


Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

no crossings

acyclic digraph with
a single source s and a single sink t

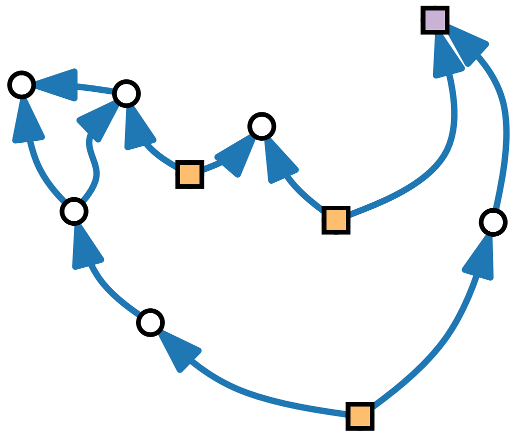


Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

no crossings

acyclic digraph with
a single source s and a single sink t

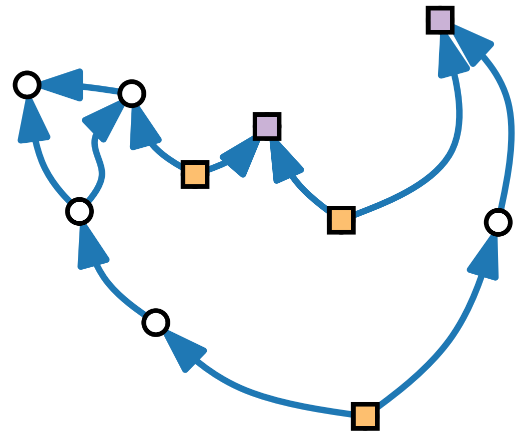


Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

no crossings

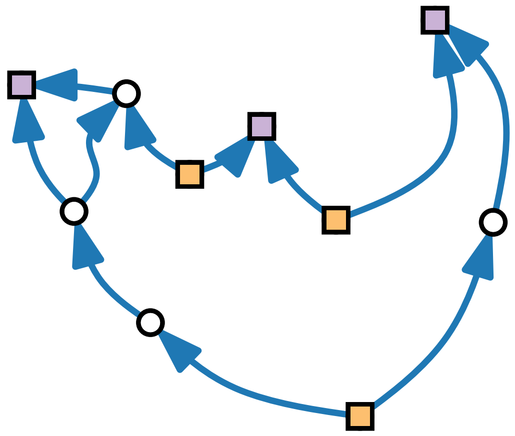
acyclic digraph with
a single source s and a single sink t



Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:

(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



no crossings

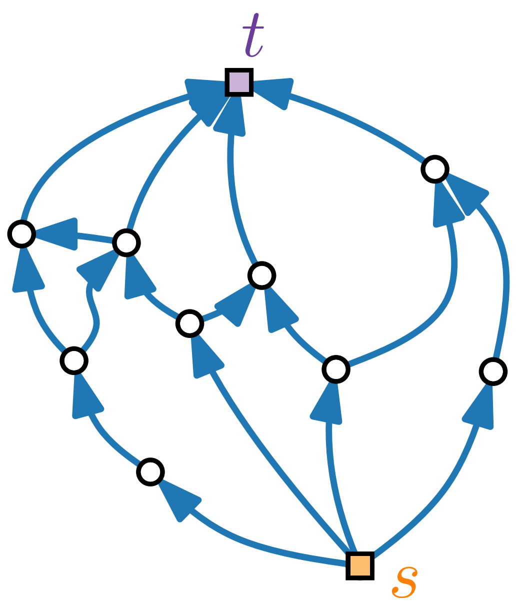
acyclic digraph with
a single source s and a single sink t

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

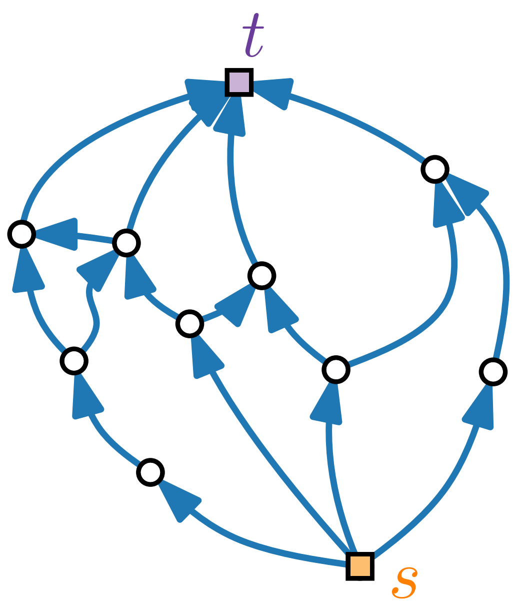
no crossings

acyclic digraph with
a single source s and a single sink t



Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Additionally:
Embedded such that s and t are on the outer face f_0.

no crossings

acyclic digraph with
a single source s and a single sink t

Upward Planarity – Characterization

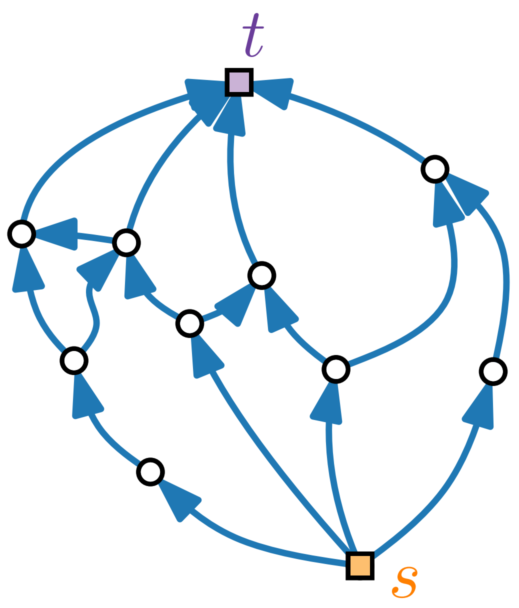
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

no crossings

acyclic digraph with
a single source s and a single sink t

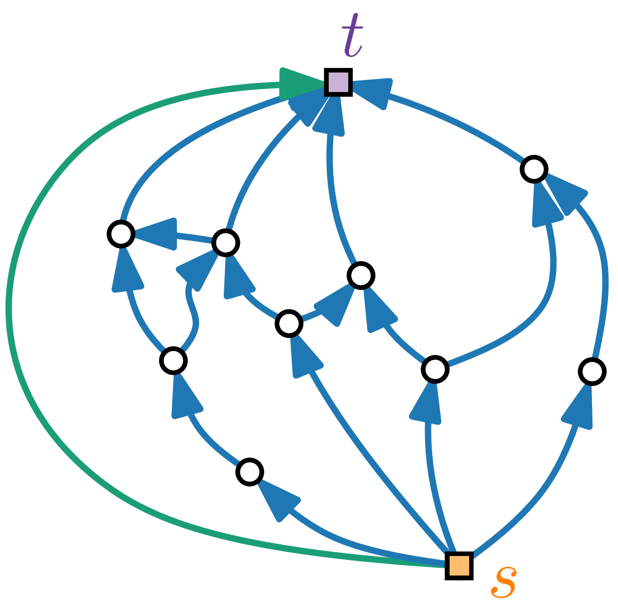
Additionally:
Embedded such that s and t are on the outer face f_0.

or:
Edge (s, t) exists.



Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



no crossings

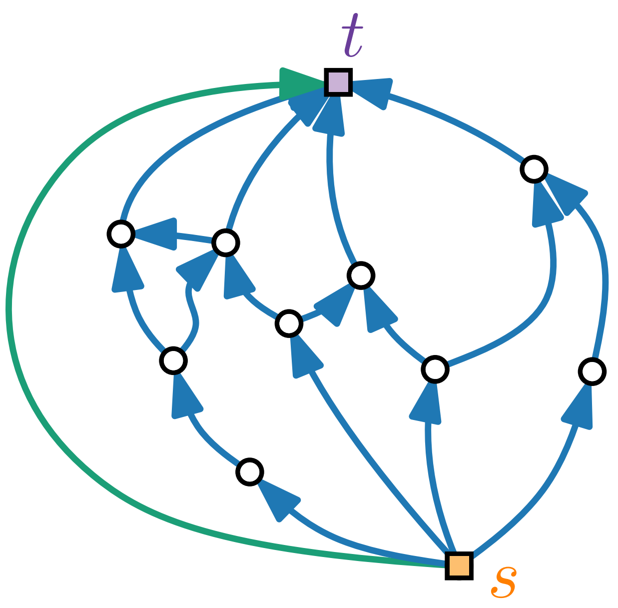
acyclic digraph with
a single source s and a single sink t

Additionally:
Embedded such that s and t are on the outer face f_0.

or:
Edge (s, t) exists.

Upward Planarity – Characterization

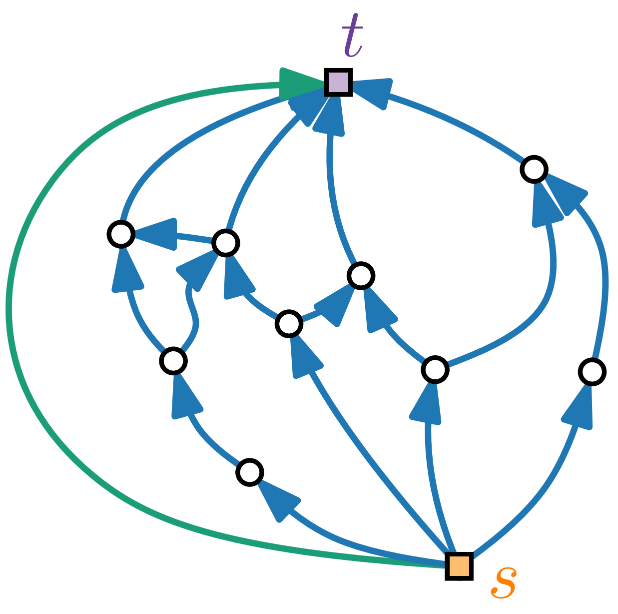
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

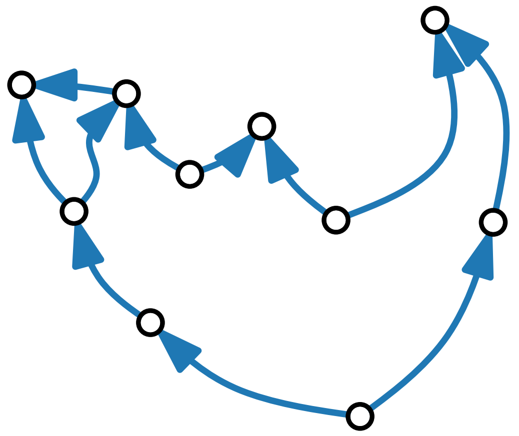


Proof.
(2) $\Rightarrow$ (1) By definition.

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:

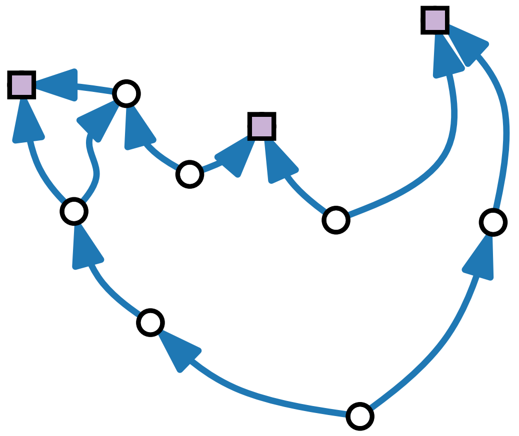
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.

Upward Planarity – Characterization

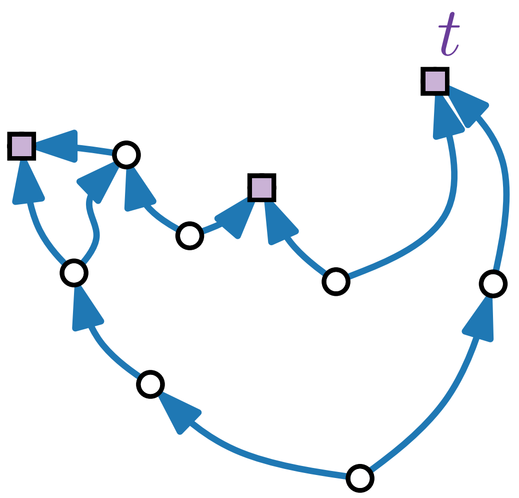
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.

Upward Planarity – Characterization

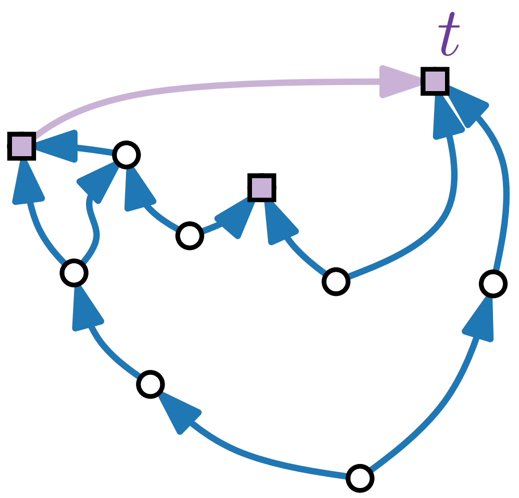
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.

Upward Planarity – Characterization

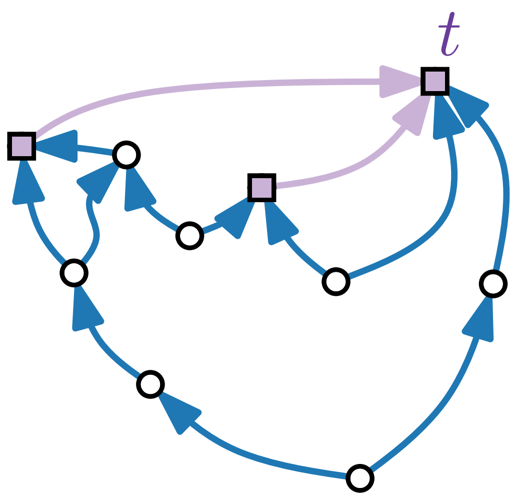
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.

Upward Planarity – Characterization

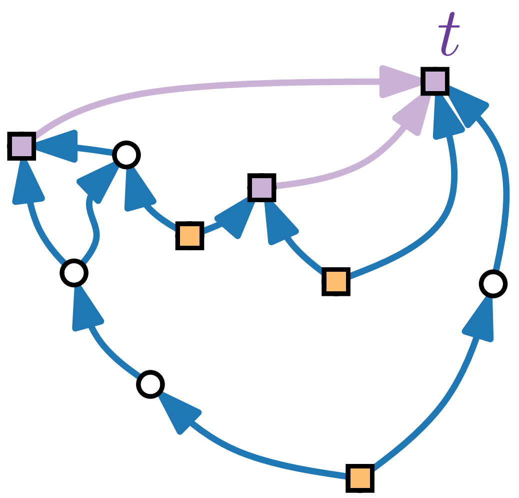
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.

Upward Planarity – Characterization

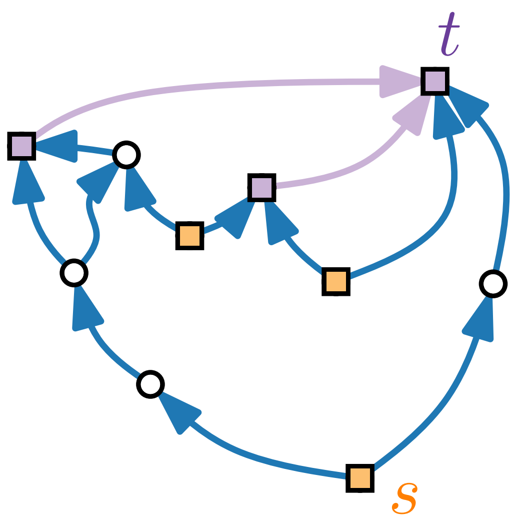
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.

Upward Planarity – Characterization

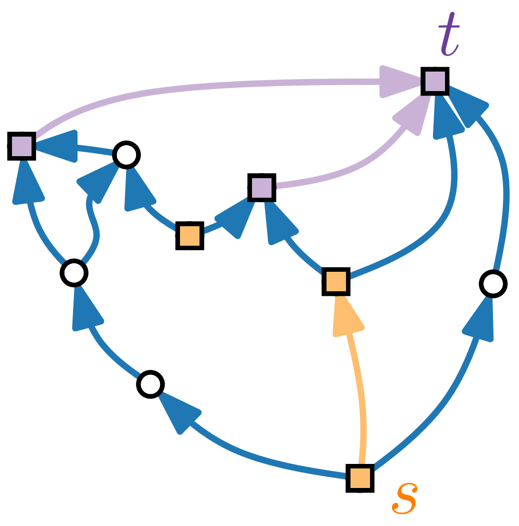
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.

Upward Planarity – Characterization

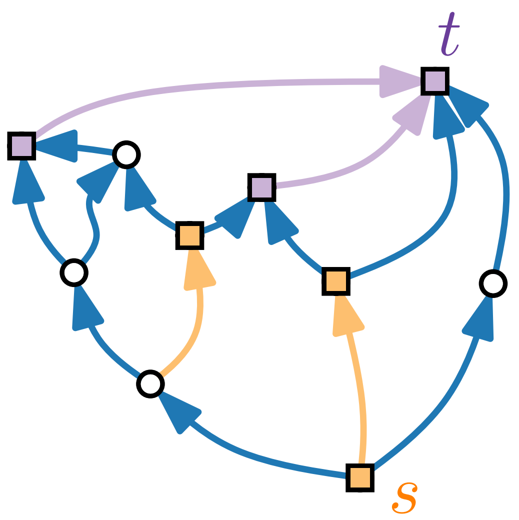
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.

Upward Planarity – Characterization

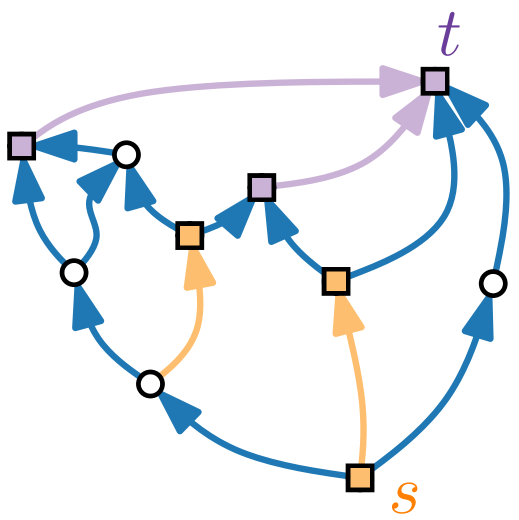
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.

Upward Planarity – Characterization

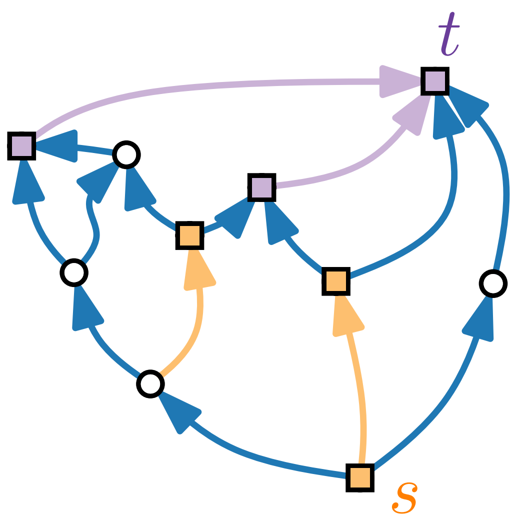
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2)

Upward Planarity – Characterization

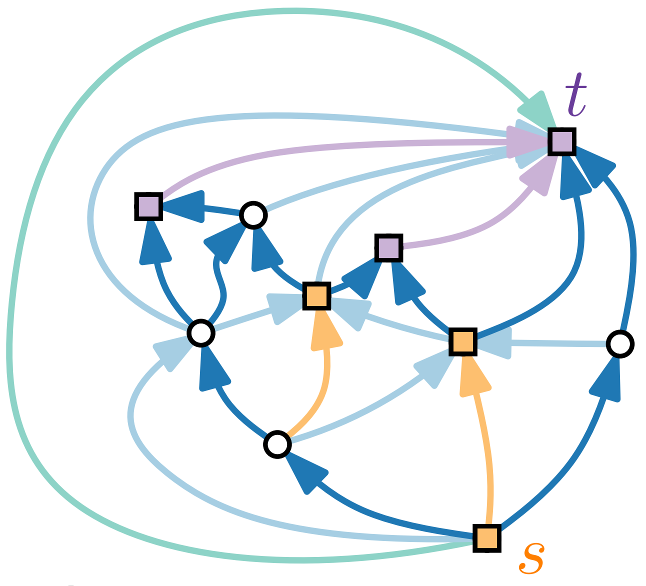
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Upward Planarity – Characterization

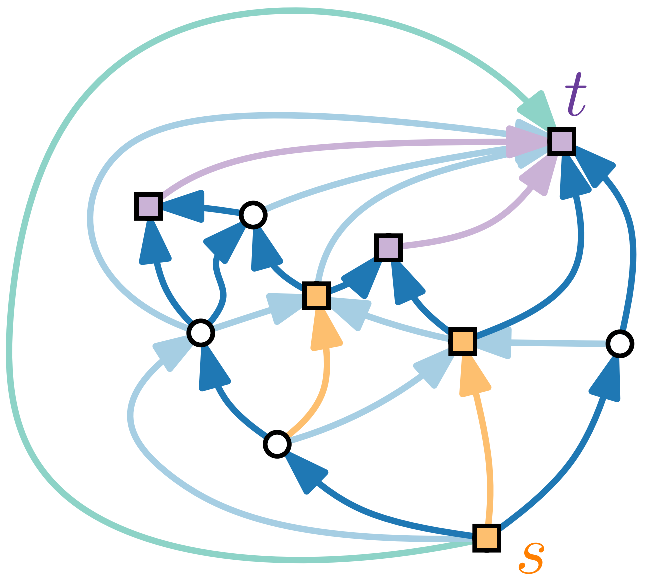
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

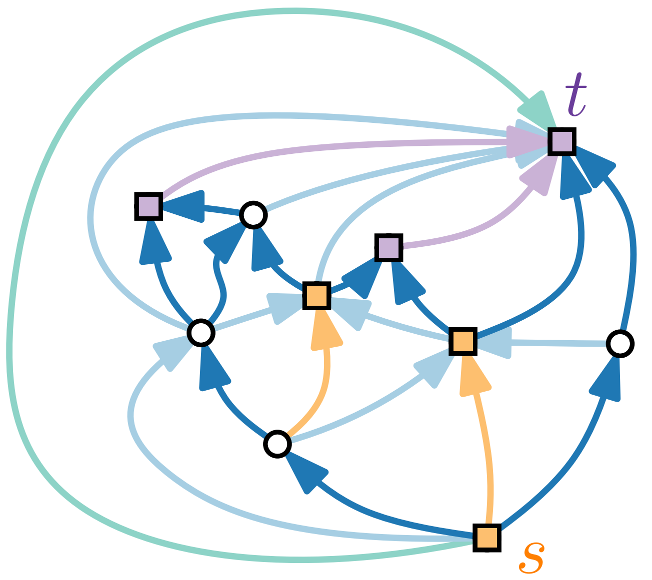


Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Claim.
Can be drawn in pre-specified triangle.

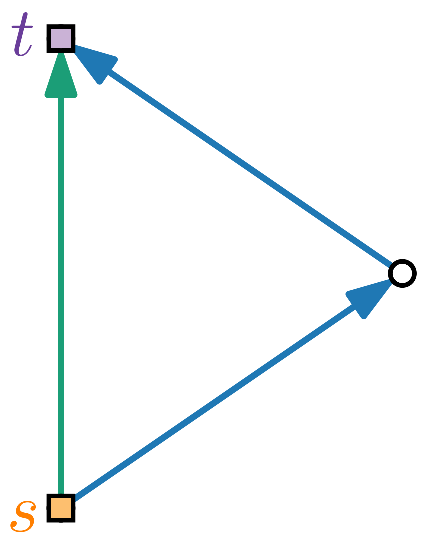
Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



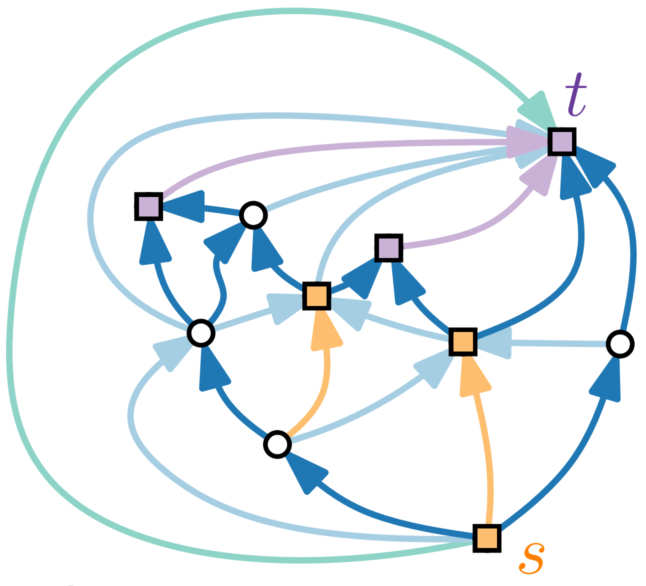
Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Claim.
Can be drawn in pre-specified triangle.



Upward Planarity – Characterization

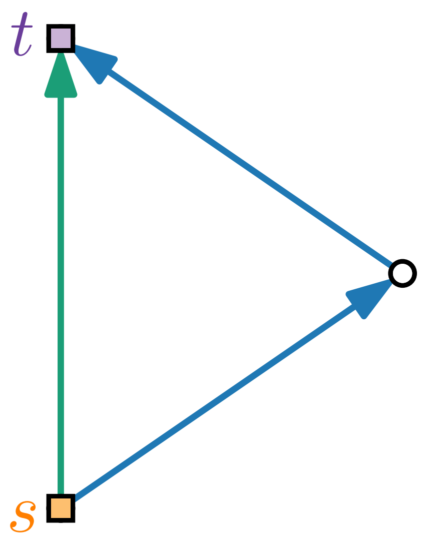
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

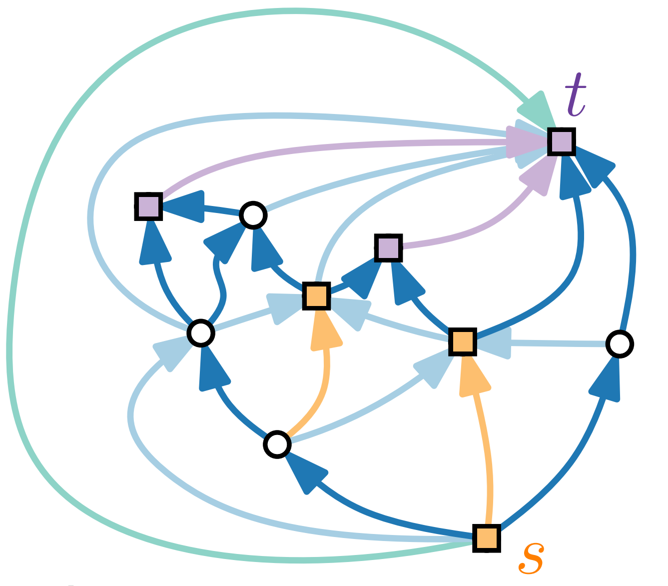
Claim.
Can be drawn in pre-specified triangle.

Induction on the number of vertices n.



Upward Planarity – Characterization

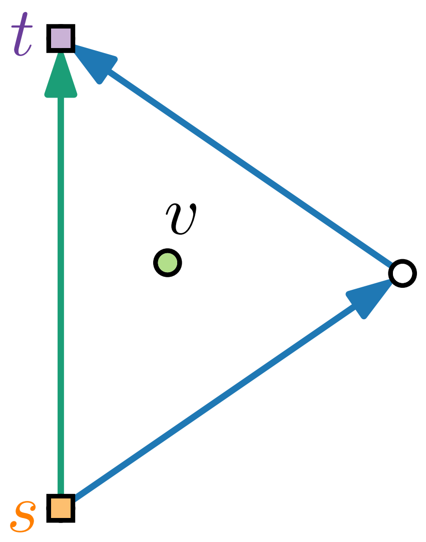
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

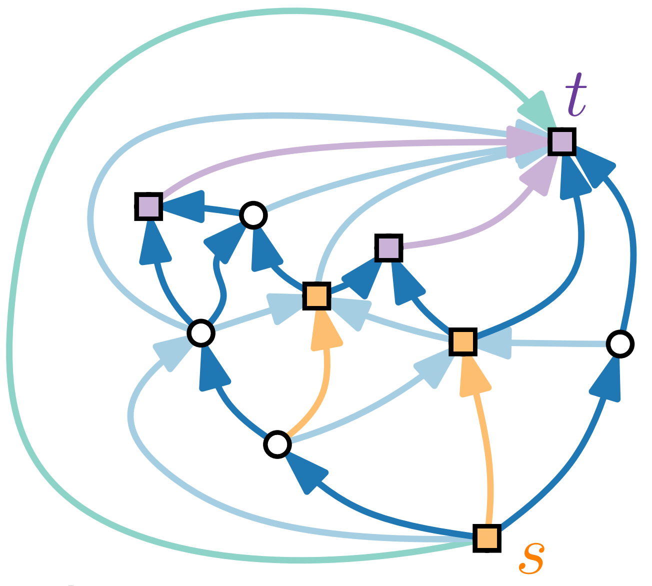
Claim.
Can be drawn in pre-specified triangle.

Induction on the number of vertices n.



Upward Planarity – Characterization

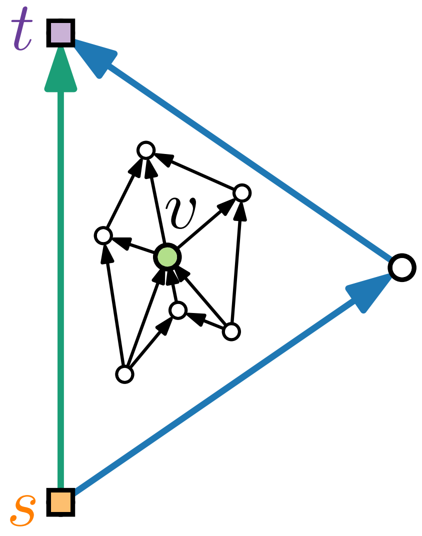
Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.



Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

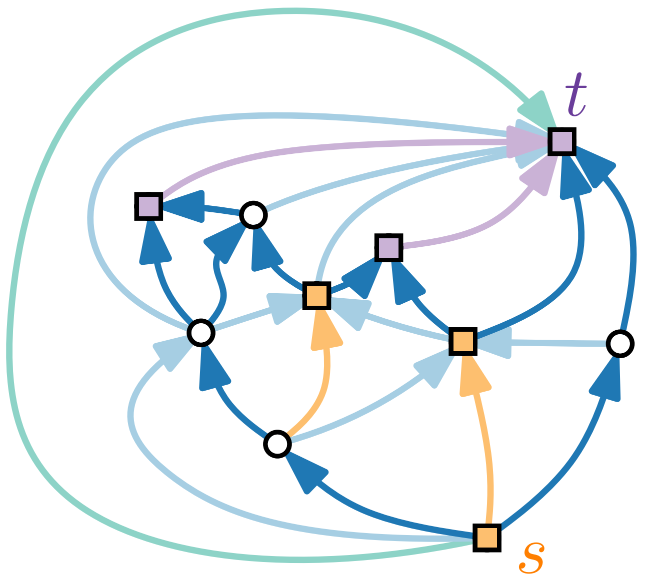
Claim.
Can be drawn in pre-specified triangle.

Induction on the number of vertices n.



Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

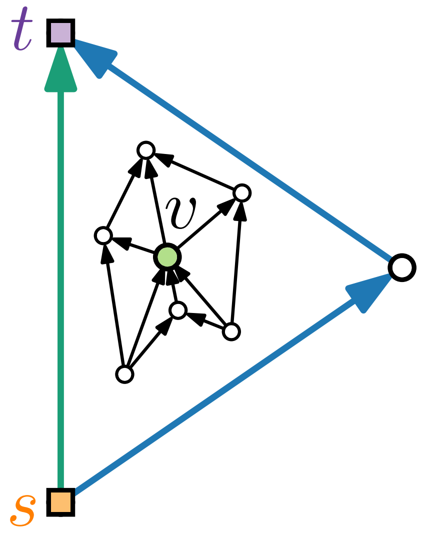


Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Claim.
Can be drawn in pre-specified triangle.

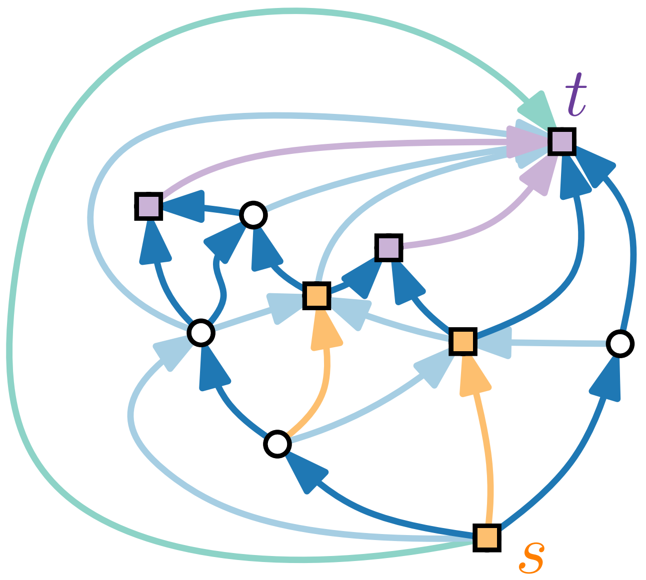
Induction on the number of vertices n.

Case 1: chord



Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

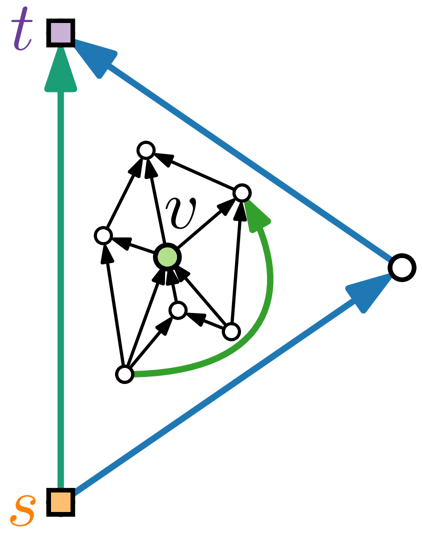


Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Claim.
Can be drawn in pre-specified triangle.

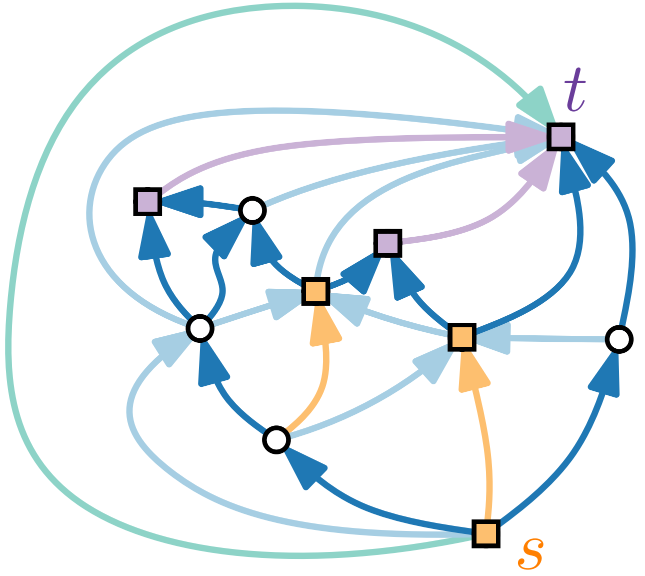
Induction on the number of vertices n.

Case 1: chord



Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

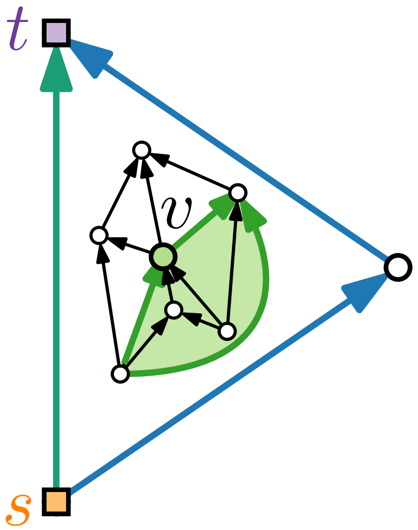


Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Claim.
Can be drawn in pre-specified triangle.

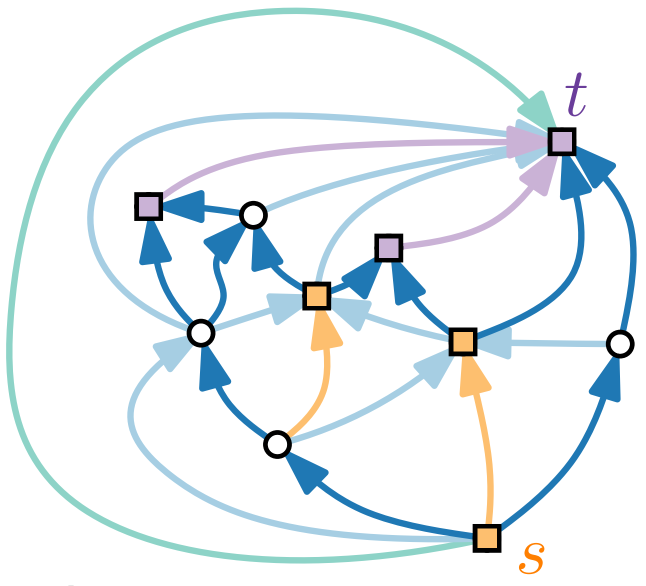
Induction on the number of vertices n.

Case 1: chord



Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

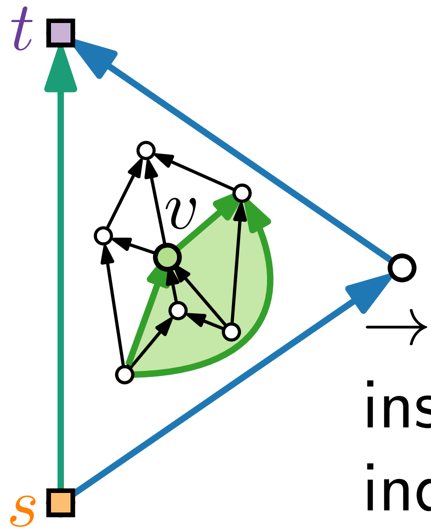


Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Claim.
Can be drawn in pre-specified triangle.

Induction on the number of vertices n.

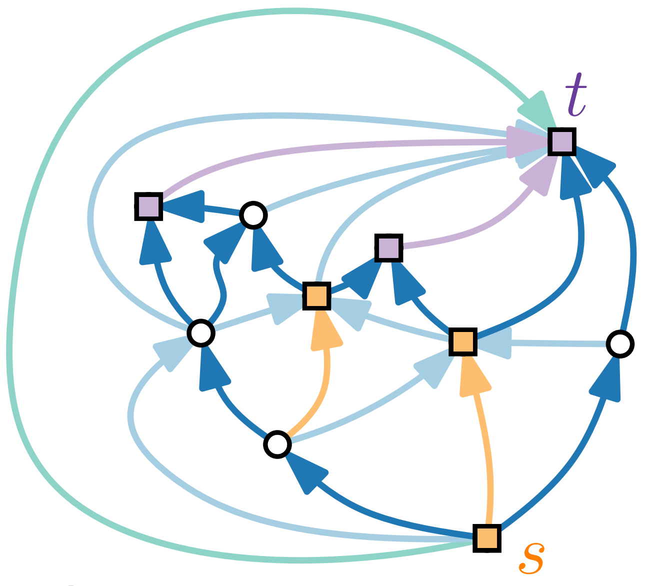
Case 1: chord



$\rightarrow$ two smaller instances; solve inductively

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

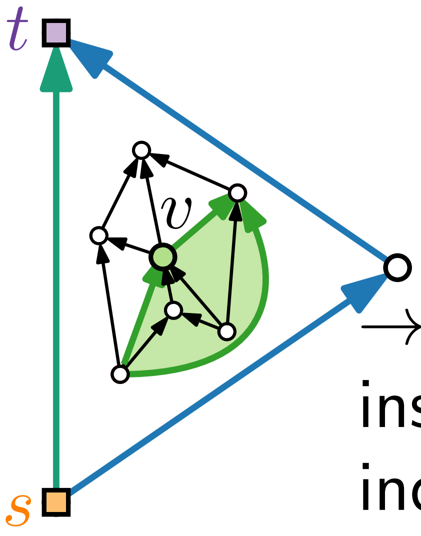


Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Claim.
Can be drawn in pre-specified triangle.

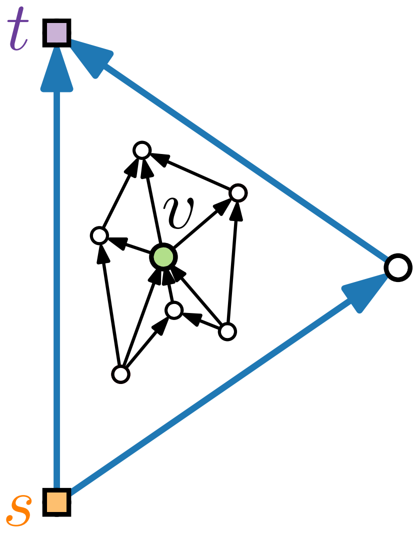
Induction on the number of vertices n.

Case 1: chord



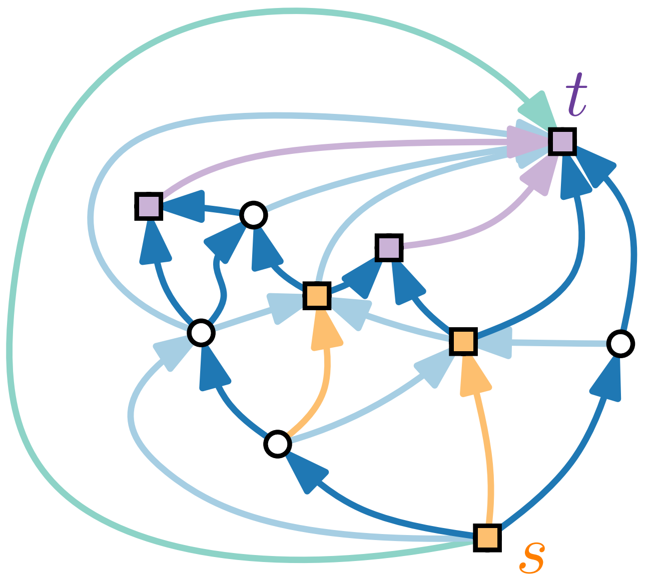
$\rightarrow$ two smaller instances; solve inductively

Case 2: no chord



Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

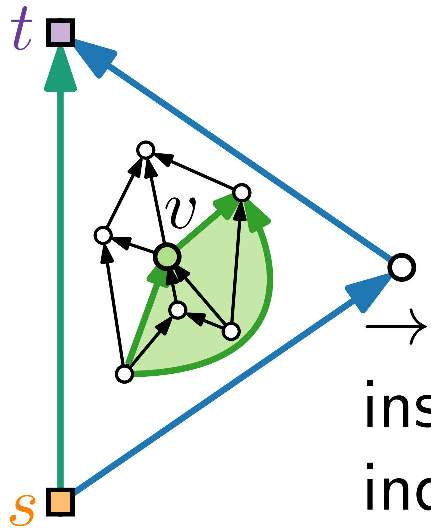


Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Claim.
Can be drawn in pre-specified triangle.

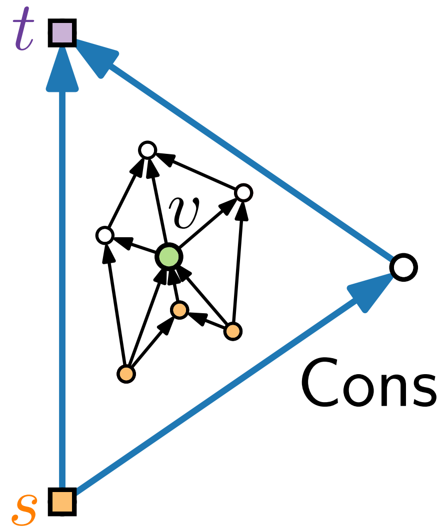
Induction on the number of vertices n.

Case 1: chord



$\rightarrow$ two smaller instances; solve inductively

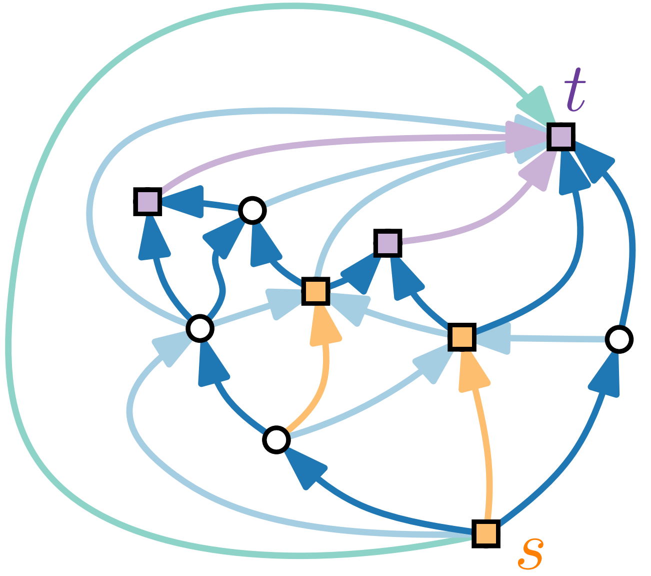
Case 2: no chord



Consider vertices below v.

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

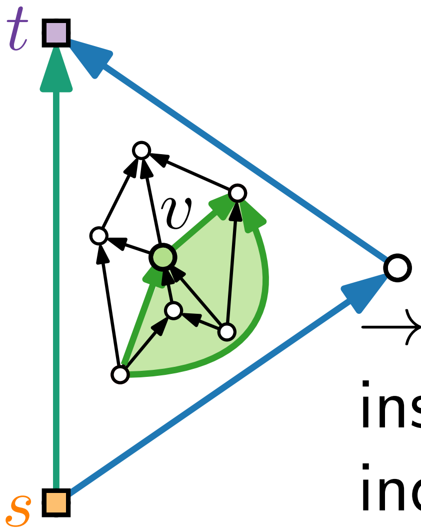


Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Claim.
Can be drawn in pre-specified triangle.

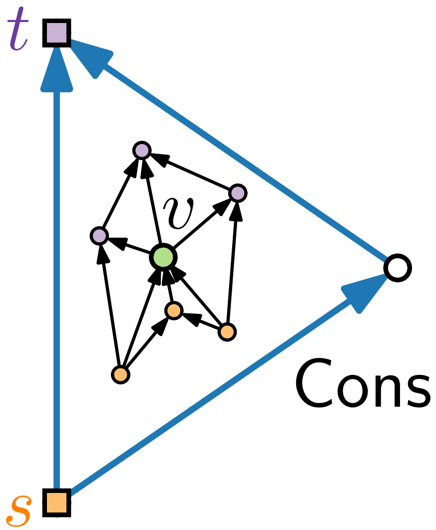
Induction on the number of vertices n.

Case 1: chord



$\rightarrow$ two smaller instances; solve inductively

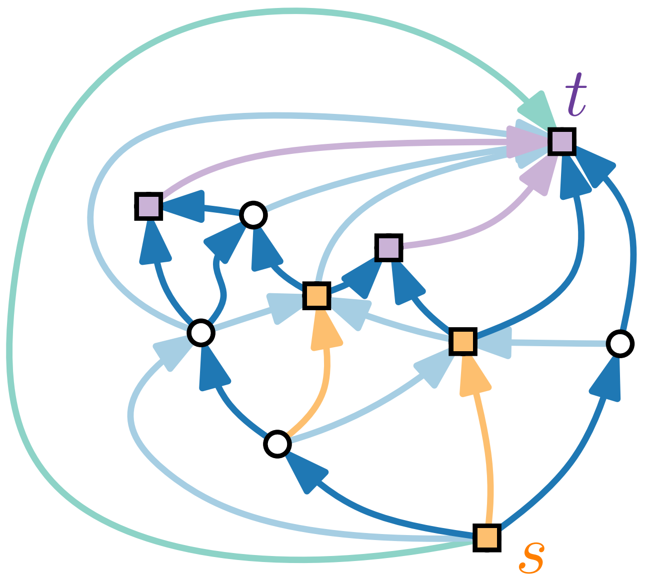
Case 2: no chord



Consider vertices below v.

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

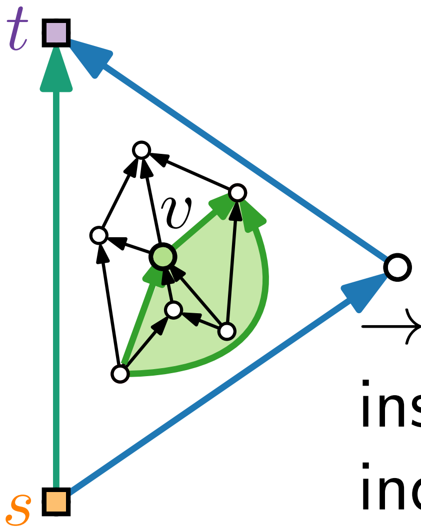


Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Claim.
Can be drawn in pre-specified triangle.

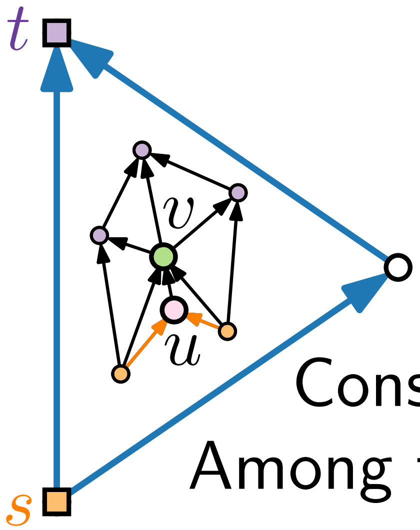
Induction on the number of vertices n.

Case 1: chord



$\rightarrow$ two smaller instances; solve inductively

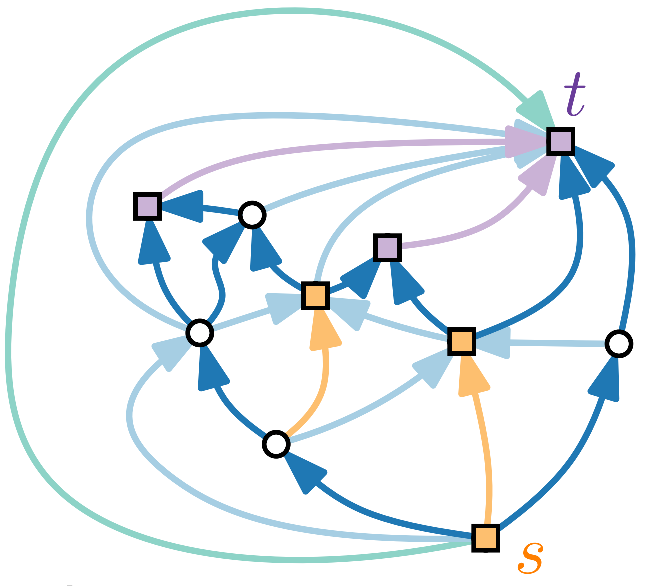
Case 2: no chord



Consider vertices below v.
Among these, take "highest."

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

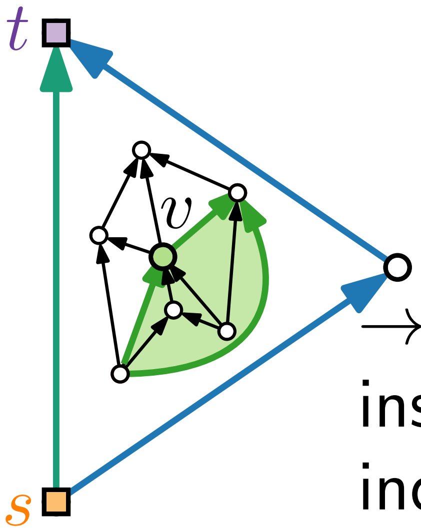


Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Claim.
Can be drawn in pre-specified triangle.

Induction on the number of vertices n.

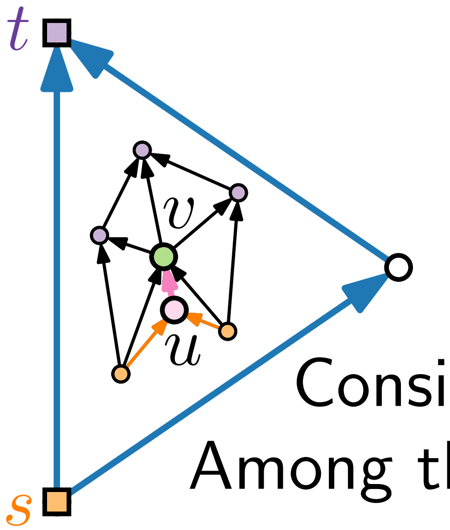
Case 1: chord



$\rightarrow$ two smaller instances; solve inductively

Idea: Contract uv!

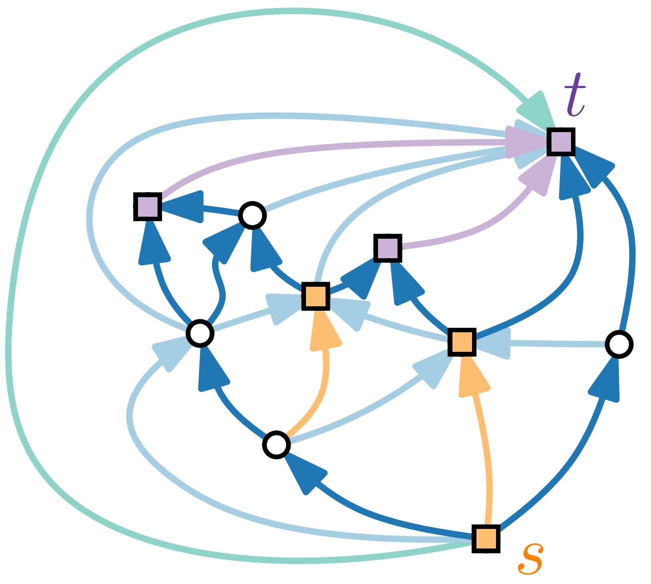
Case 2: no chord



Consider vertices below v. Among these, take "highest."

Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

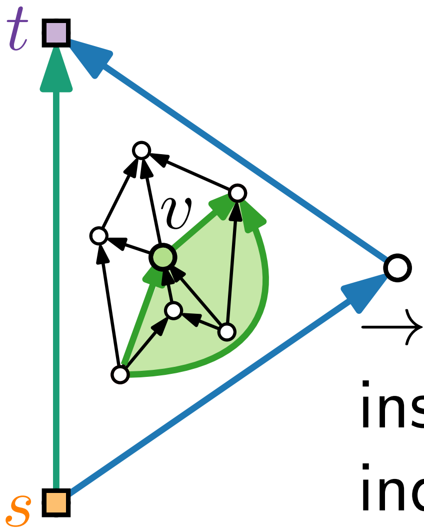


Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Claim.
Can be drawn in pre-specified triangle.

Induction on the number of vertices n.

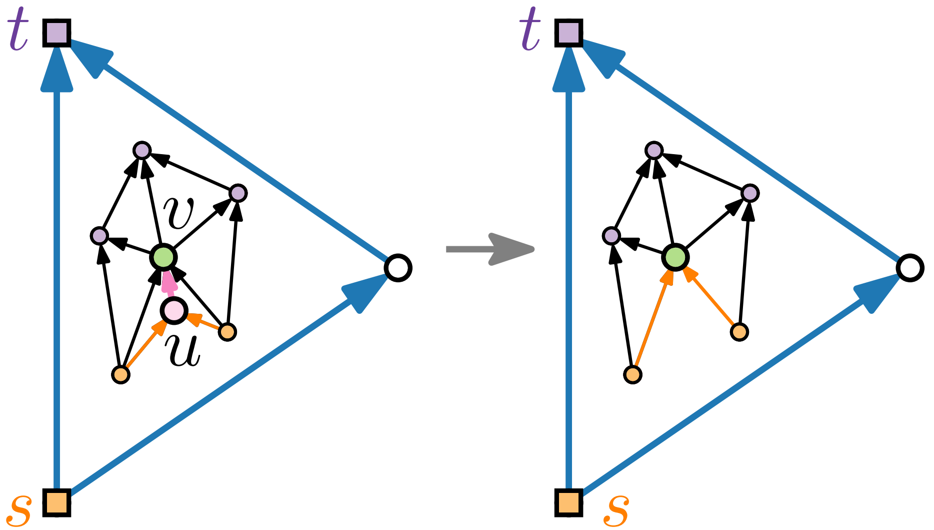
Case 1: chord



$\rightarrow$ two smaller instances; solve inductively

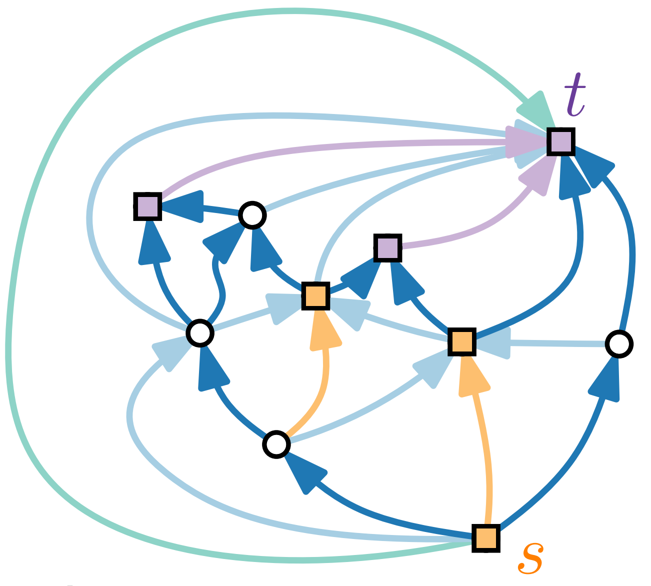
Case 2: no chord

Idea: Contract uv!



Upward Planarity – Characterization

Theorem 1. [Kelly 1987, Di Battista & Tamassia 1988]
For a digraph G, the following statements are equivalent:
(1) G is upward planar.
(2) G admits an upward planar straight-line drawing.
(3) G is a spanning subgraph of a planar st-digraph.

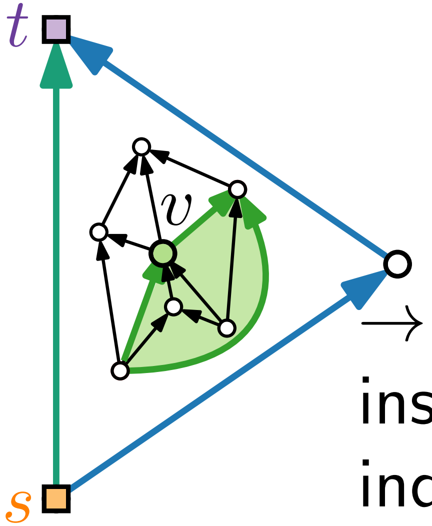


Proof.
(2) $\Rightarrow$ (1) By definition. (1) $\Rightarrow$ (3) For the proof idea, see the example above.
(3) $\Rightarrow$ (2) Triangulate & construct drawing:

Claim.
Can be drawn in pre-specified triangle.

Induction on the number of vertices n.

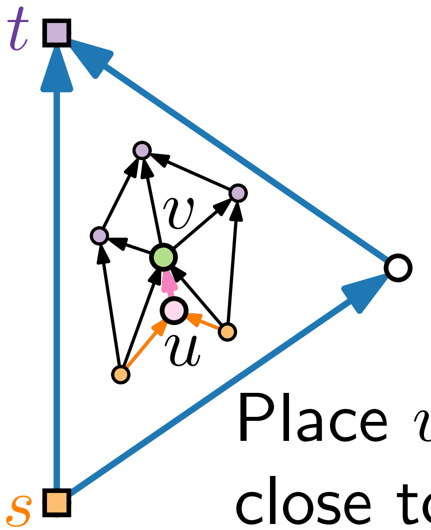
Case 1: chord

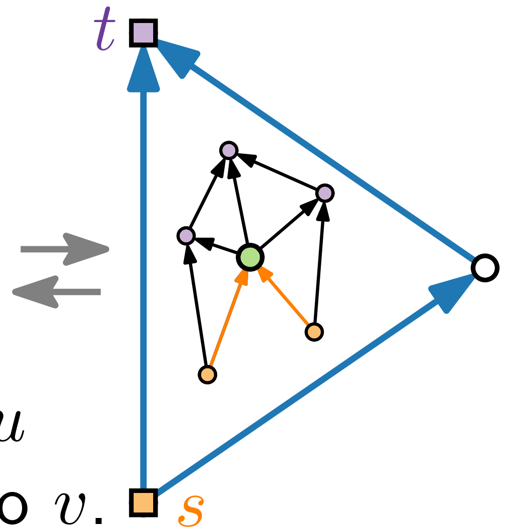


$\rightarrow$ two smaller instances; solve inductively

Case 2: no chord

Idea: Contract uv!





Place u close to v.

Upward Planarity – Complexity

Given a *planar acyclic* digraph G,
decide whether G is upward planar.

Upward Planarity – Complexity

Theorem. [Garg & Tamassia, 1995]
Given a *planar acyclic* digraph G,
it is NP-hard to decide whether G is upward planar.

Upward Planarity – Complexity

Theorem. [Garg & Tamassia, 1995]
Given a *planar acyclic* digraph G,
it is NP-hard to decide whether G is upward planar.

Theorem 2. [Bertolazzi, Di Battista, Mannino, Tamassia, 1994]
Given an *embedded* planar digraph G,
it can be tested in quadratic time whether G is upward planar.

Upward Planarity – Complexity

Theorem. [Garg & Tamassia, 1995]
Given a *planar acyclic* digraph G,
it is NP-hard to decide whether G is upward planar.

Theorem 2. [Bertolazzi, Di Battista, Mannino, Tamassia, 1994]
Given an *embedded* planar digraph G,
it can be tested in quadratic time whether G is upward planar.

Corollary.
Given a *triconnected* planar digraph G,
it can be tested in quadratic time whether G is upward planar.

Upward Planarity – Complexity

Theorem. [Garg & Tamassia, 1995]
Given a *planar acyclic* digraph G,
it is NP-hard to decide whether G is upward planar.

Theorem 2. [Bertolazzi, Di Battista, Mannino, Tamassia, 1994]
Given an *embedded* planar digraph G,
it can be tested in quadratic time whether G is upward planar.

Corollary.
Given a *triconnected* planar digraph G,
it can be tested in quadratic time whether G is upward planar.

Theorem. [Hutton & Lubiw, 1996]
Given an acyclic *single-source* digraph G,
it can be tested in linear time whether G is upward planar.

Upward Planarity – Complexity

Theorem. [Garg & Tamassia, 1995]
Given a *planar acyclic* digraph G,
it is NP-hard to decide whether G is upward planar.

Theorem 2. [Bertolazzi, Di Battista, Mannino, Tamassia, 1994]
Given an *embedded* planar digraph G,
it can be tested in quadratic time whether G is upward planar.

Corollary.
Given a *triconnected* planar digraph G,
it can be tested in quadratic time whether G is upward planar.

Theorem. [Hutton & Lubiw, 1996]
Given an acyclic *single-source* digraph G,
it can be tested in linear time whether G is upward planar.

The Problem

Fixed Embedding Upward Planarity Testing.
Let G be a plane digraph, let F be the set of faces of G, and let f_0 be the outer face of G.
Test whether G is upward planar (w.r.t. to F and f_0).

The Problem

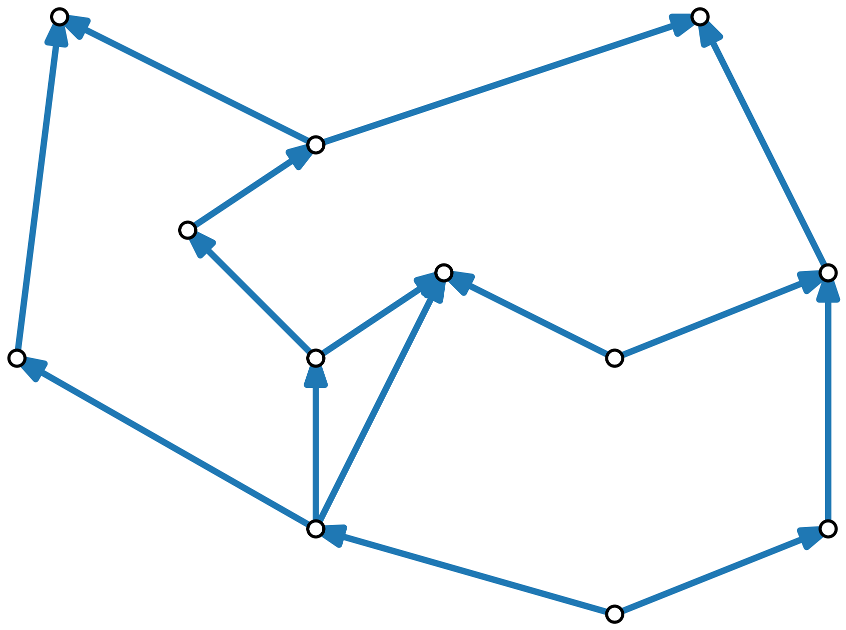
Fixed Embedding Upward Planarity Testing.
Let G be a plane digraph, let F be the set of faces of G, and let f_0 be the outer face of G.
Test whether G is upward planar (w.r.t. to F and f_0).

Plan.

- Find a property that any upward planar drawing of G satisfies.
- Formalize this property.
- Specify an algorithm to test this property.

Angles, Local Sources & Sinks

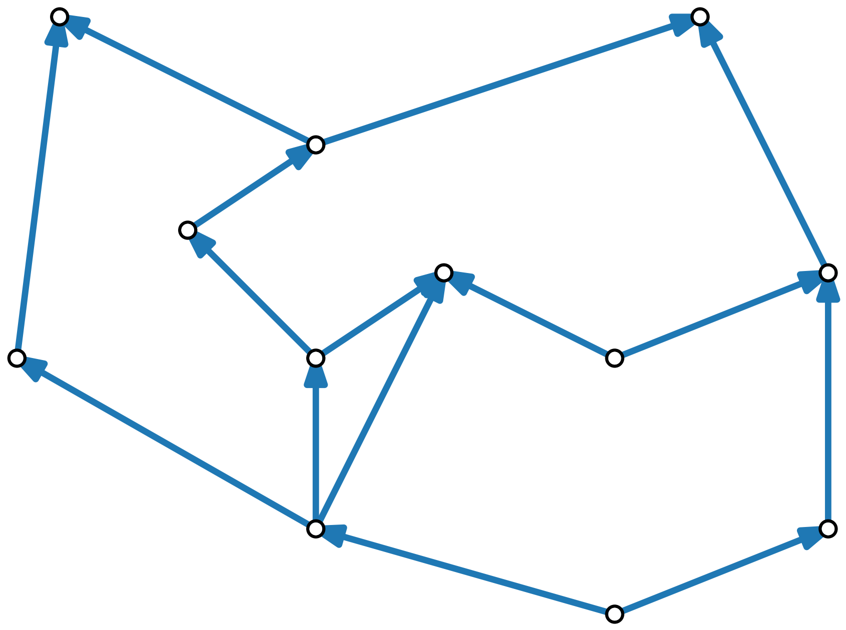
Definitions.



Angles, Local Sources & Sinks

Definitions.

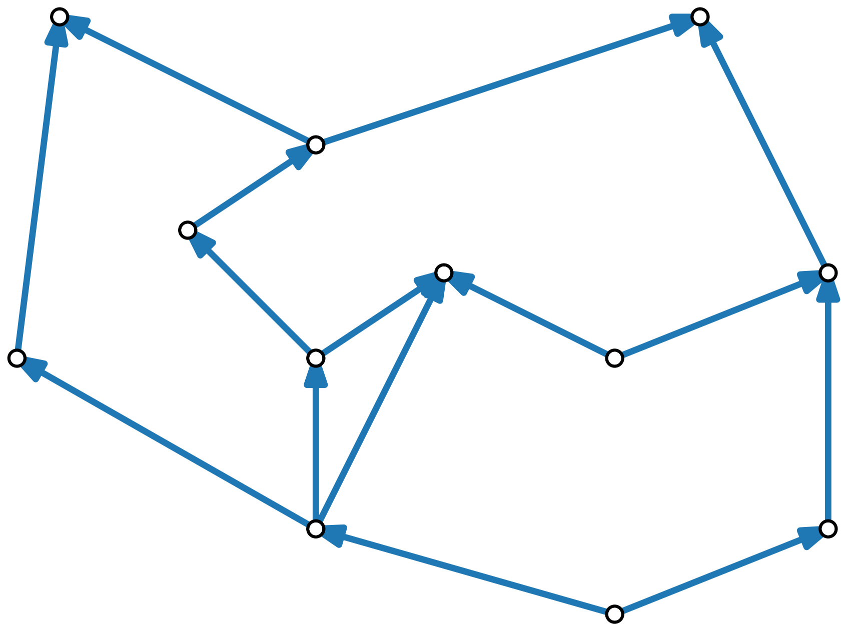
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f.



Angles, Local Sources & Sinks

Definitions.

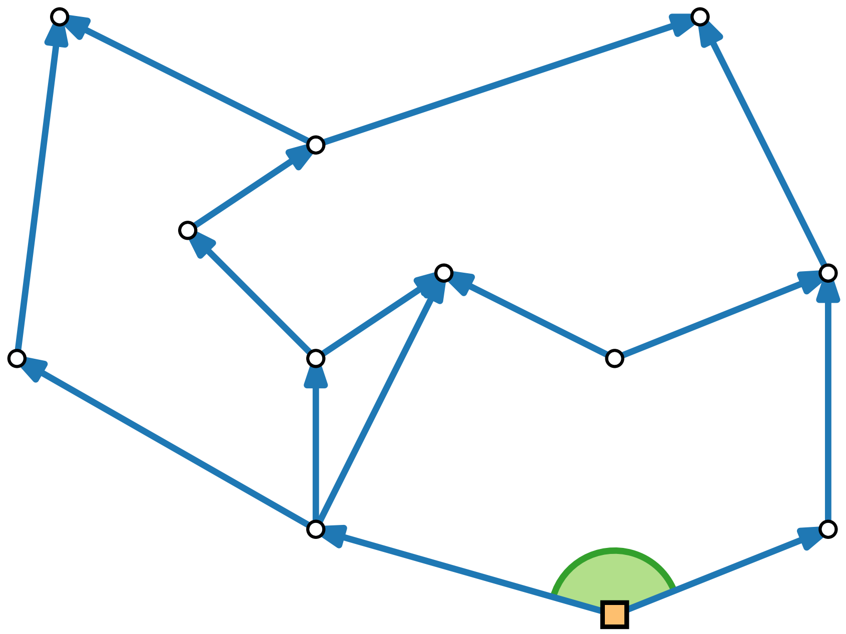
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f



Angles, Local Sources & Sinks

Definitions.

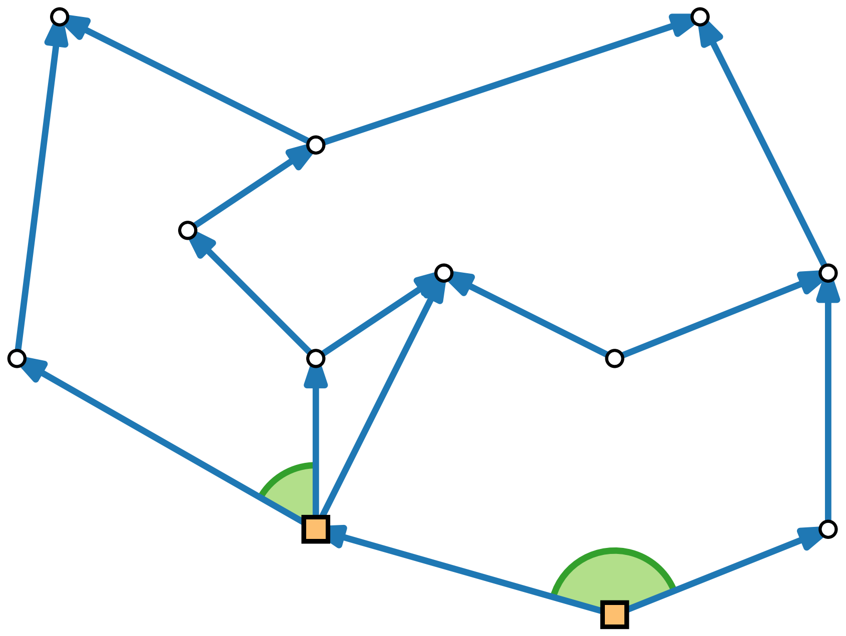
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f



Angles, Local Sources & Sinks

Definitions.

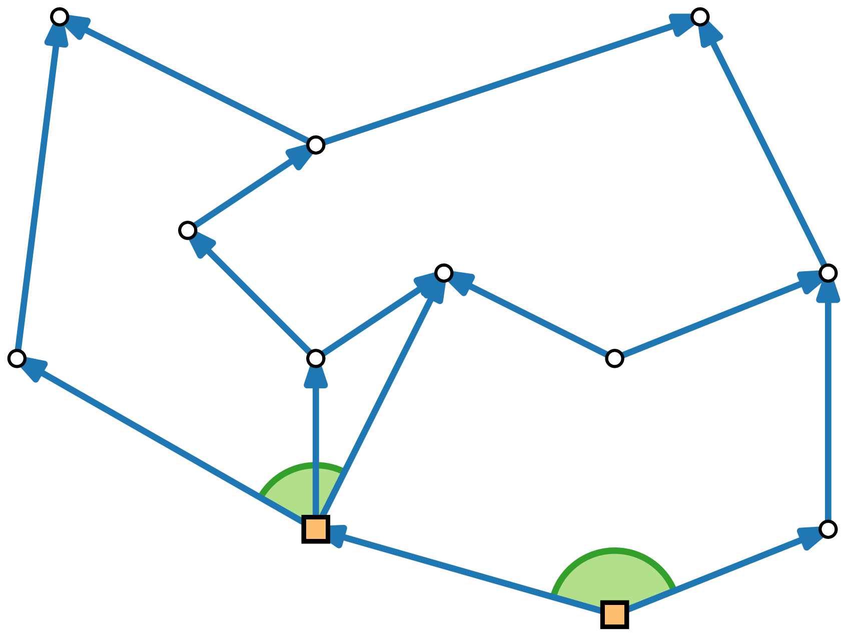
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. boundary of f



Angles, Local Sources & Sinks

Definitions.

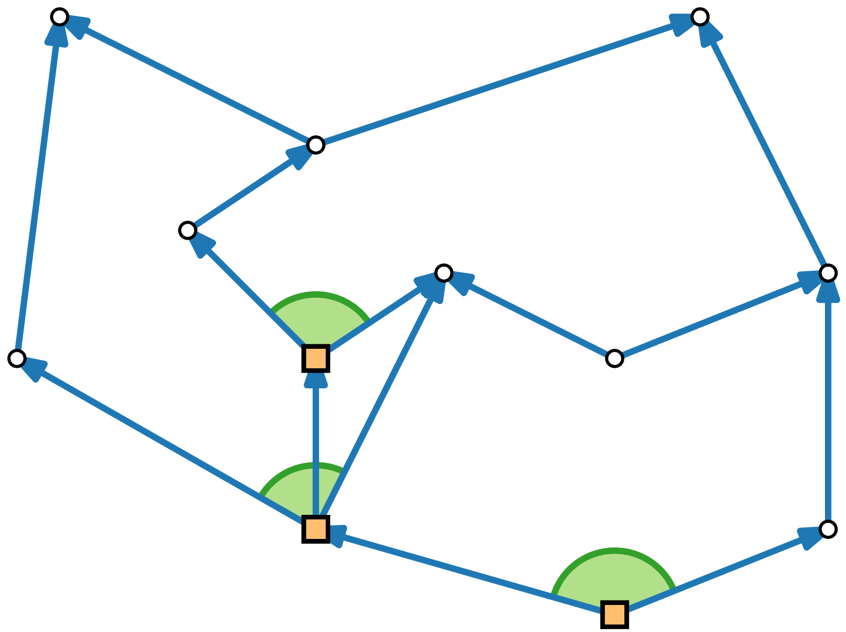
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. boundary of f



Angles, Local Sources & Sinks

Definitions.

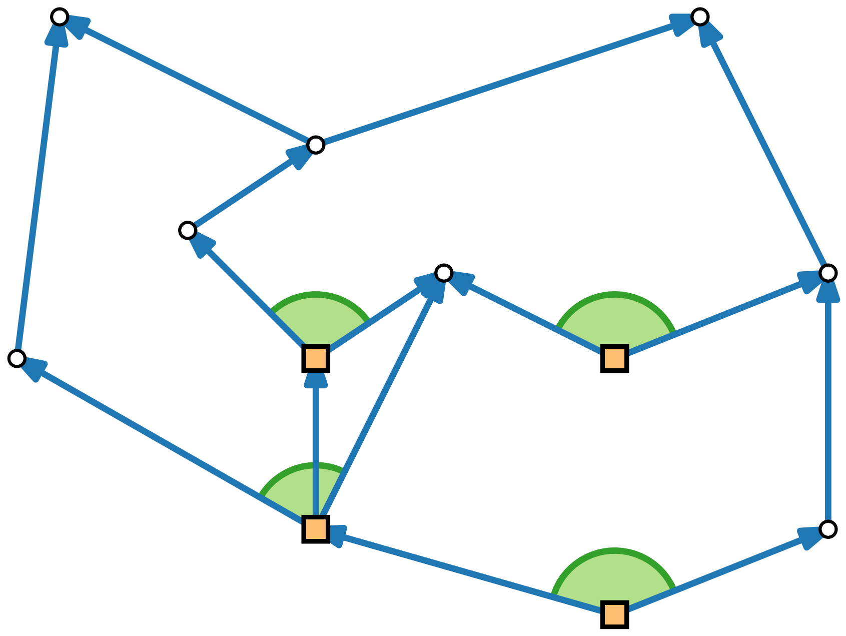
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. boundary of f



Angles, Local Sources & Sinks

Definitions.

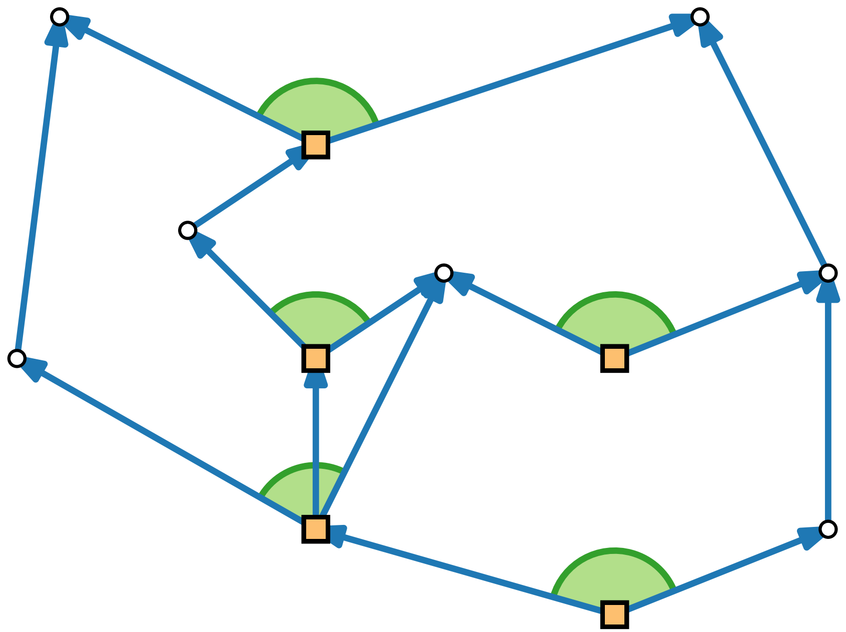
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f



Angles, Local Sources & Sinks

Definitions.

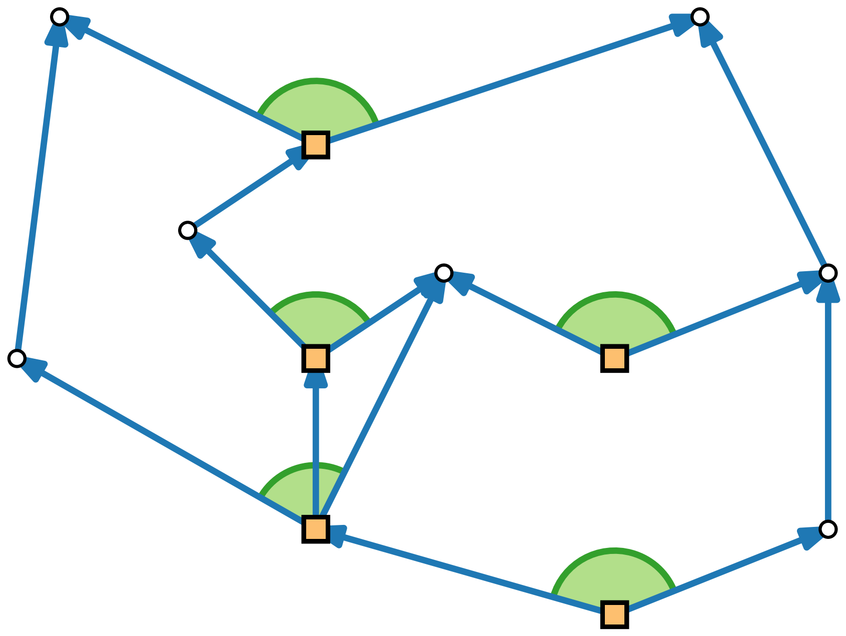
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. boundary of f



Angles, Local Sources & Sinks

Definitions.

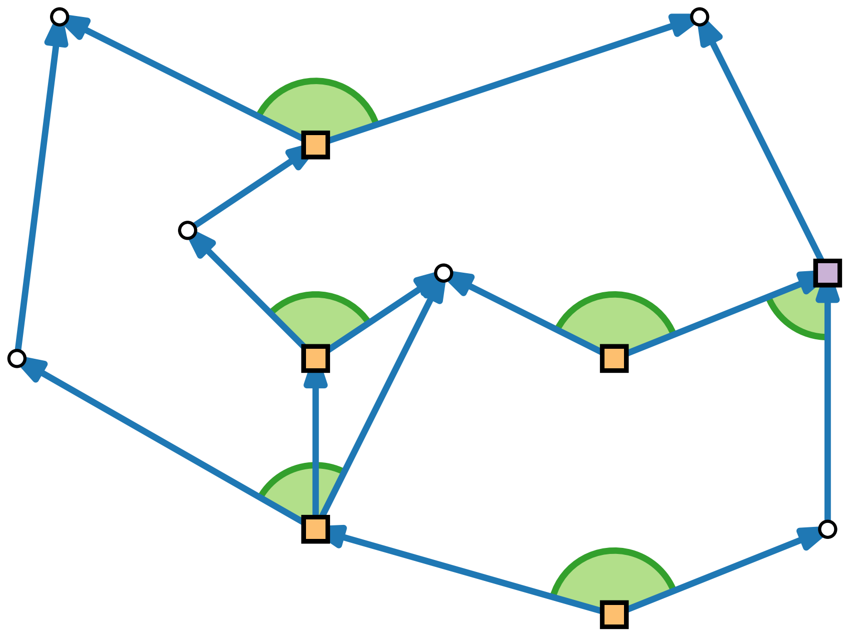
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.



Angles, Local Sources & Sinks

Definitions.

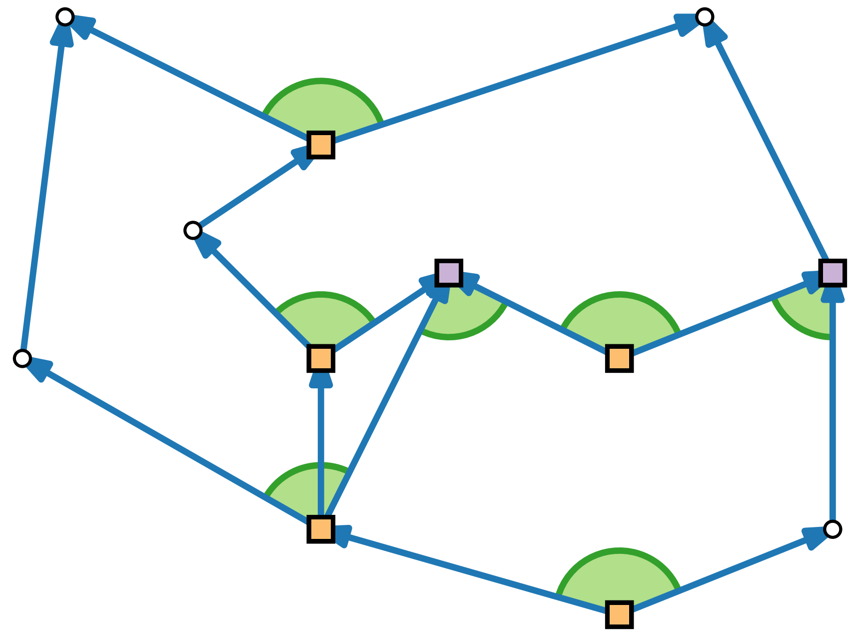
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.



Angles, Local Sources & Sinks

Definitions.

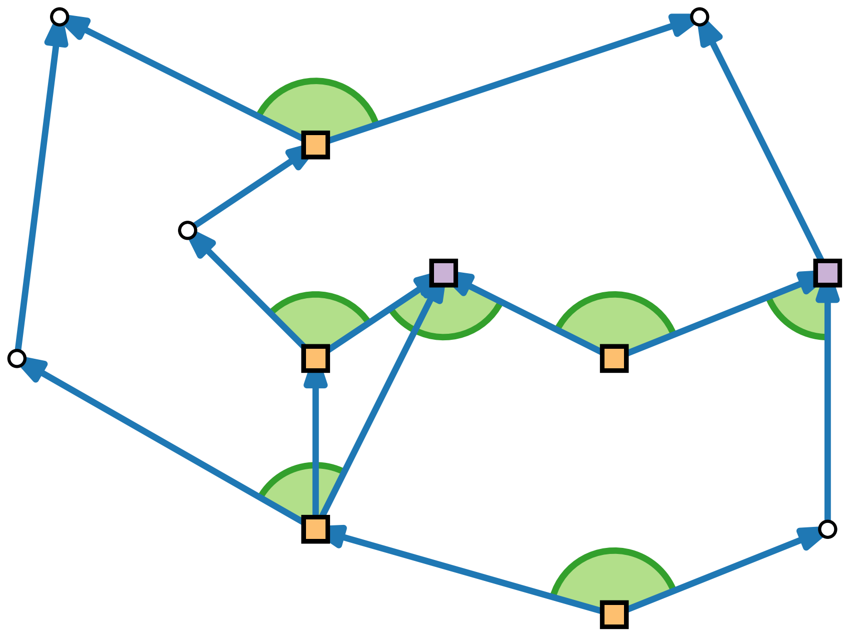
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.



Angles, Local Sources & Sinks

Definitions.

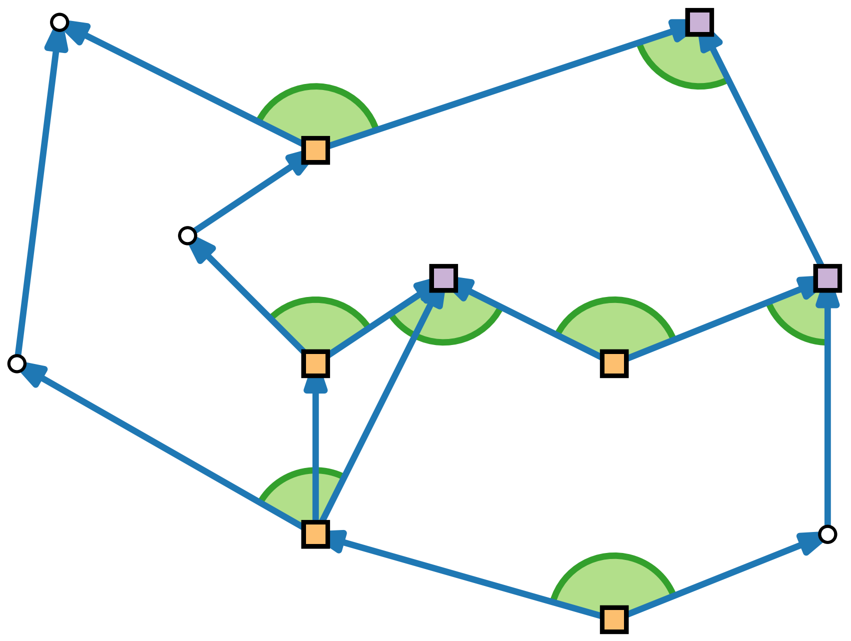
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.



Angles, Local Sources & Sinks

Definitions.

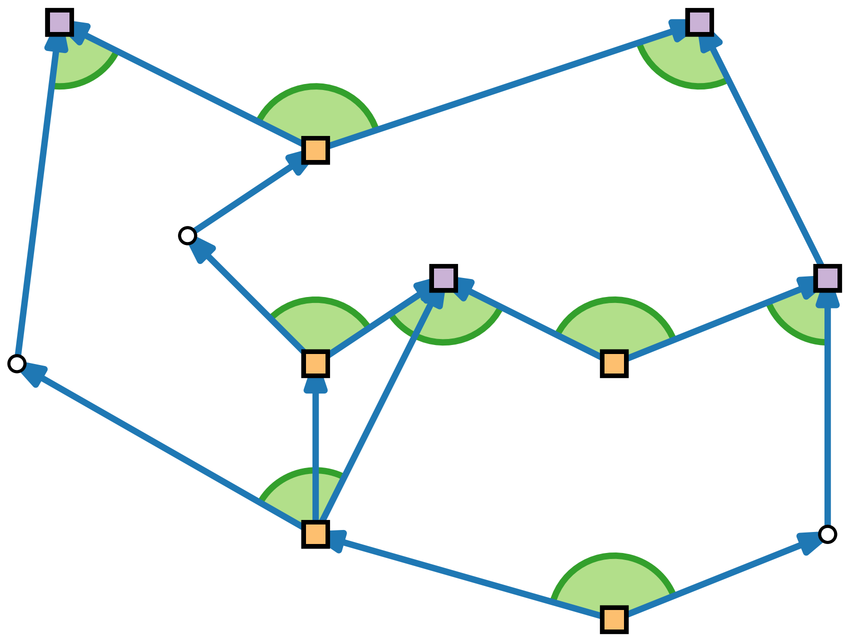
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.



Angles, Local Sources & Sinks

Definitions.

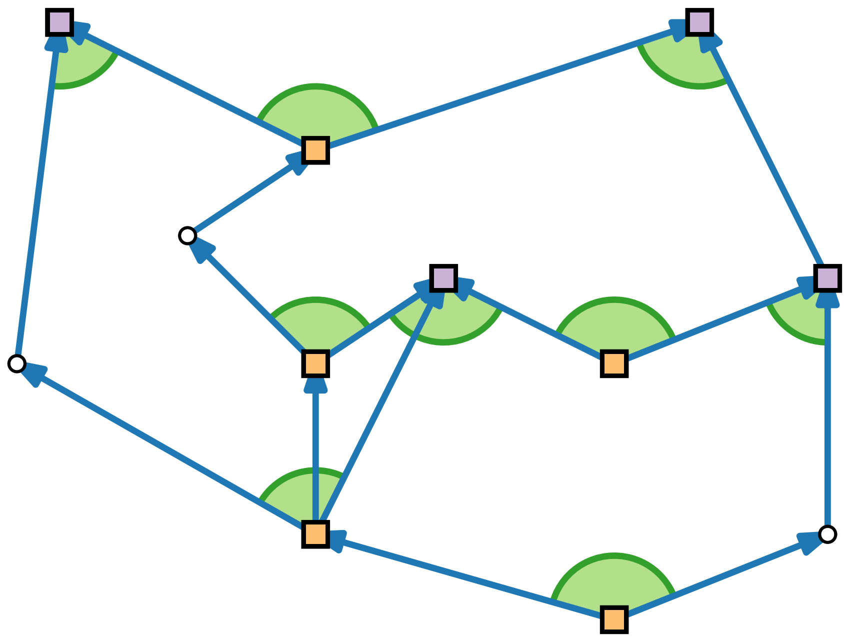
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.



Angles, Local Sources & Sinks

Definitions.

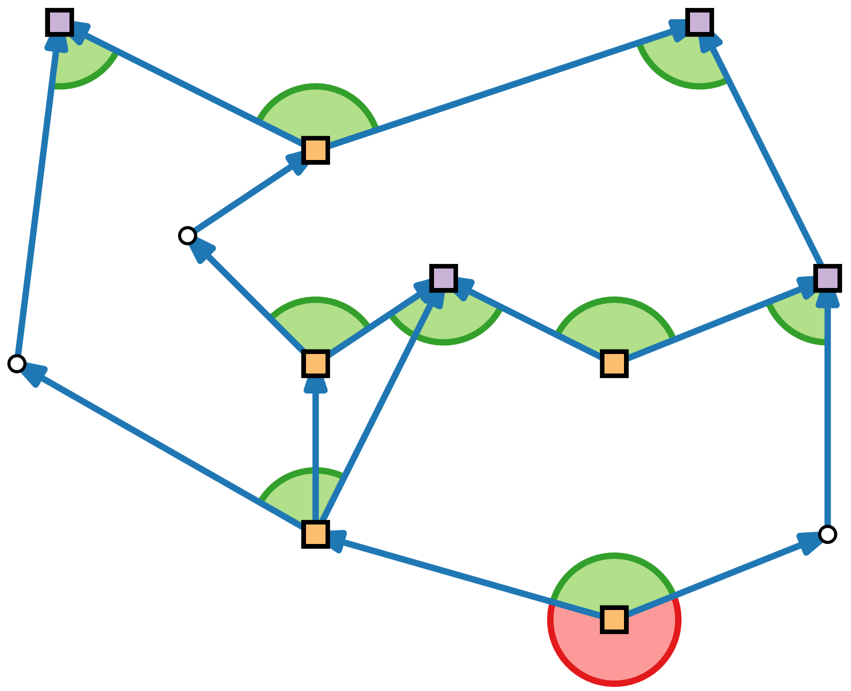
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.
- An angle α at a local source/sink is **large** if $\alpha > \pi$ and **small** otherwise.



Angles, Local Sources & Sinks

Definitions.

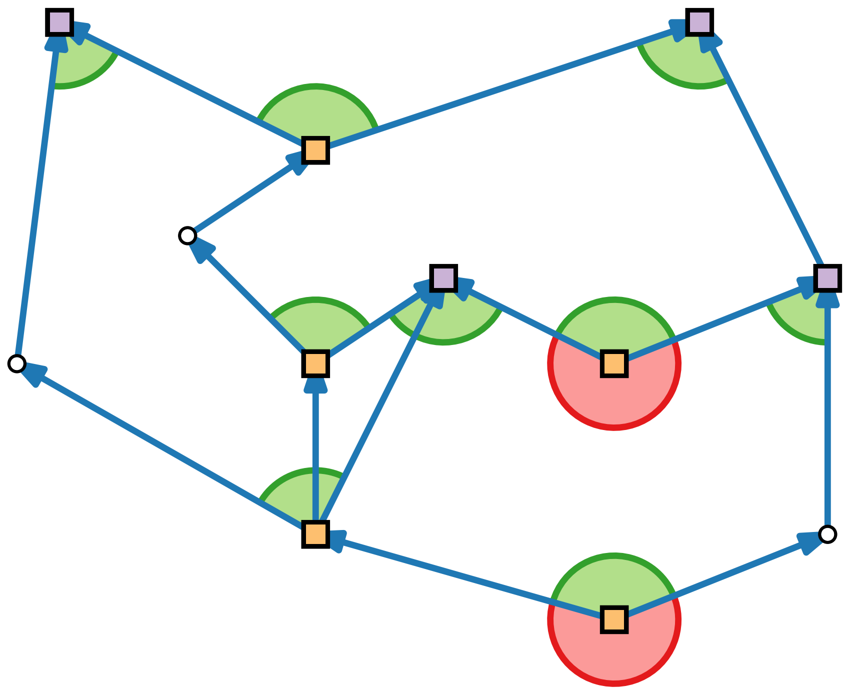
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.
- An angle α at a local source/sink is **large** if $\alpha > \pi$ and **small** otherwise.



Angles, Local Sources & Sinks

Definitions.

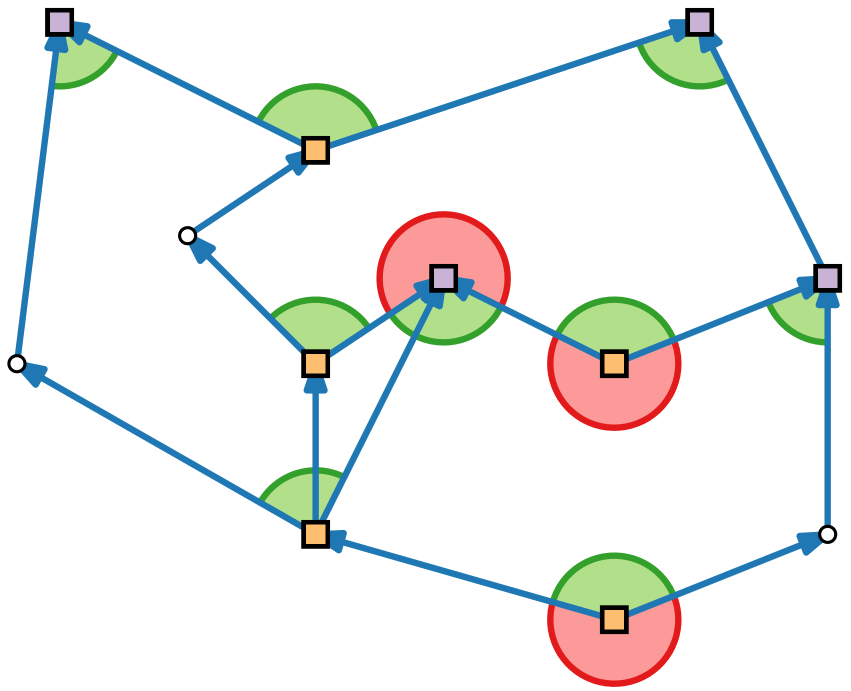
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.
- An angle α at a local source/sink is **large** if $\alpha > \pi$ and **small** otherwise.



Angles, Local Sources & Sinks

Definitions.

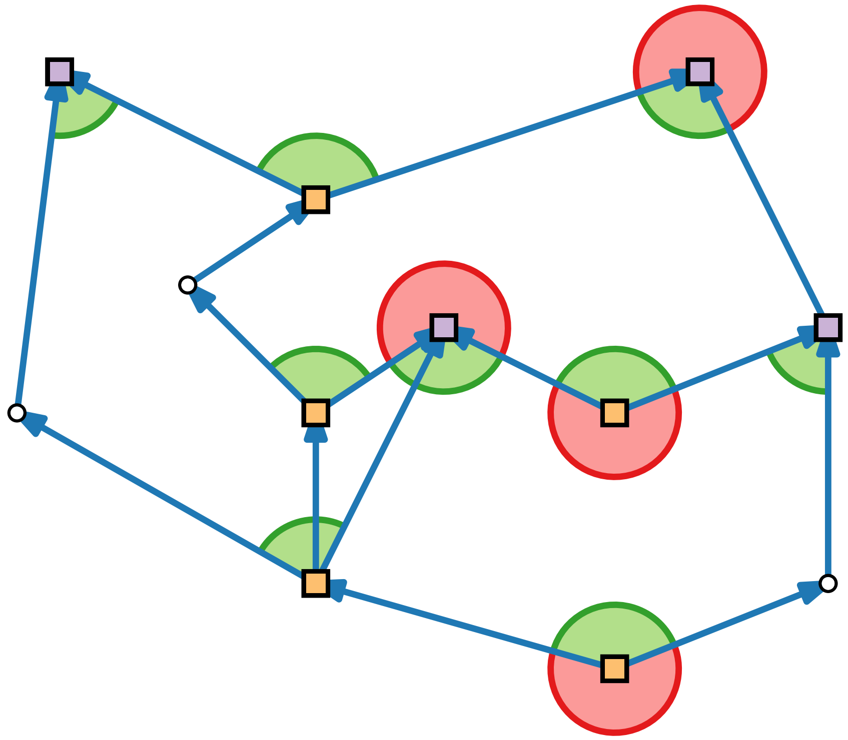
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.
- An angle α at a local source/sink is **large** if $\alpha > \pi$ and **small** otherwise.



Angles, Local Sources & Sinks

Definitions.

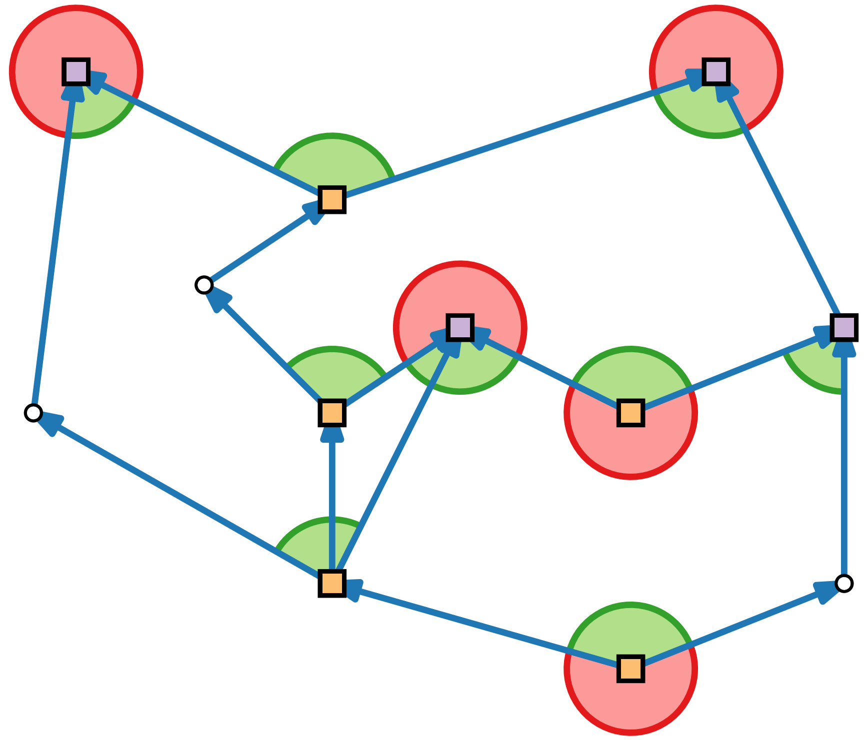
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.
- An angle α at a local source/sink is **large** if $\alpha > \pi$ and **small** otherwise.



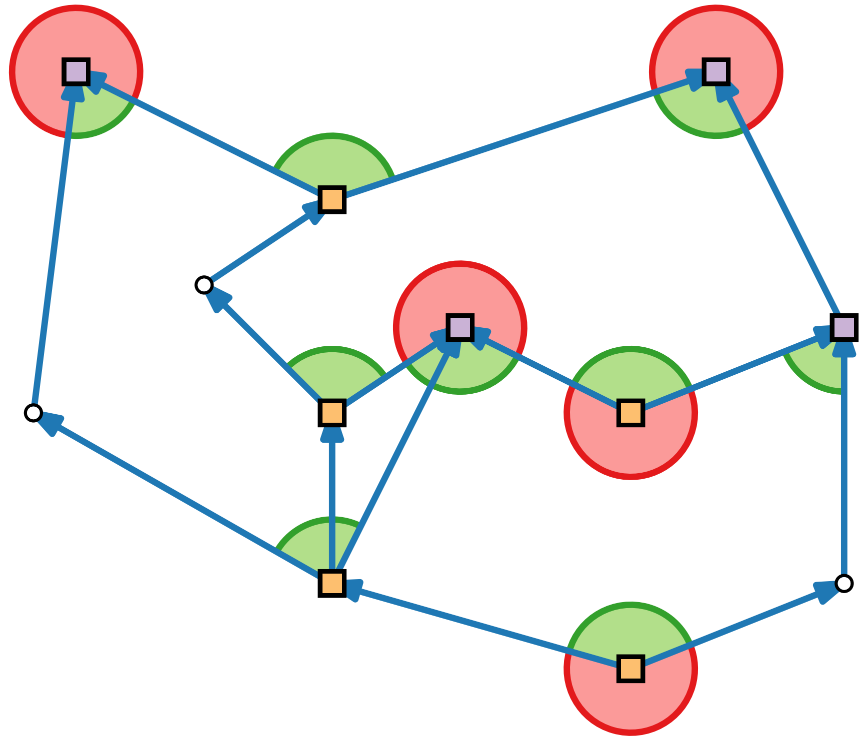
Angles, Local Sources & Sinks

Definitions.

- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.
- An angle α at a local source/sink is **large** if $\alpha > \pi$ and **small** otherwise.



Angles, Local Sources & Sinks



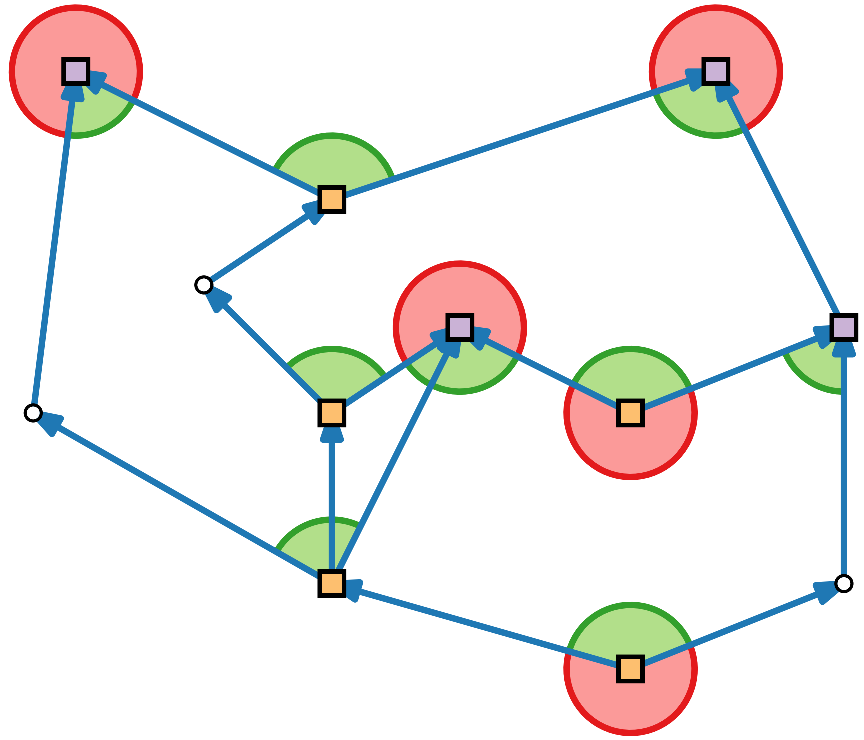
Definitions.

- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.
- An angle α at a local source/sink is **large** if $\alpha > \pi$ and **small** otherwise.
- $L(v) = \#$ large angles at v

Angles, Local Sources & Sinks

Definitions.

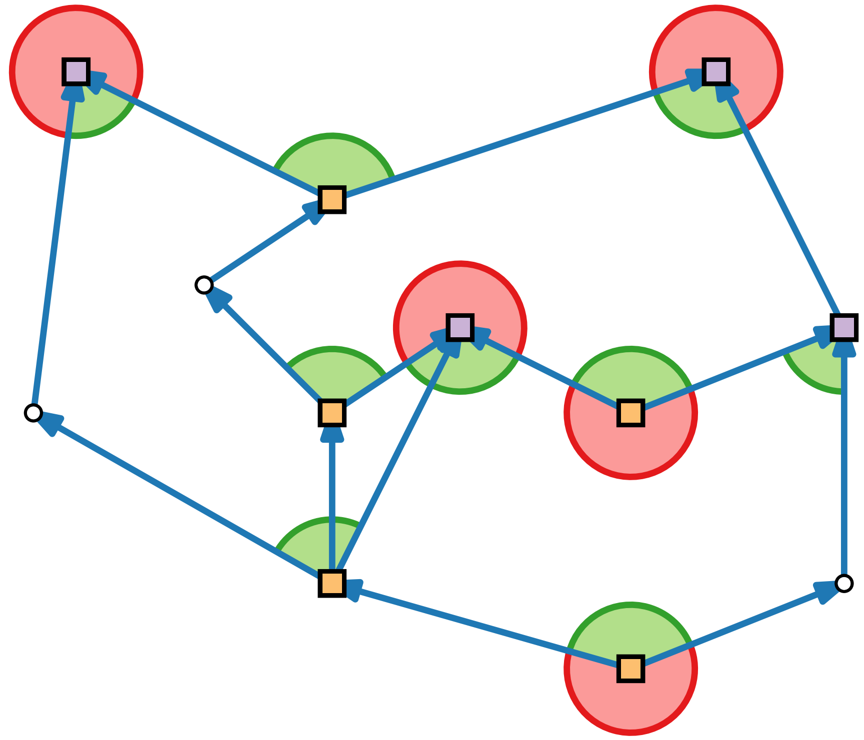
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.
- An angle α at a local source/sink is **large** if $\alpha > \pi$ and **small** otherwise.
- $L(v) = \#$ large angles at v
- $L(f) = \#$ large angles in f



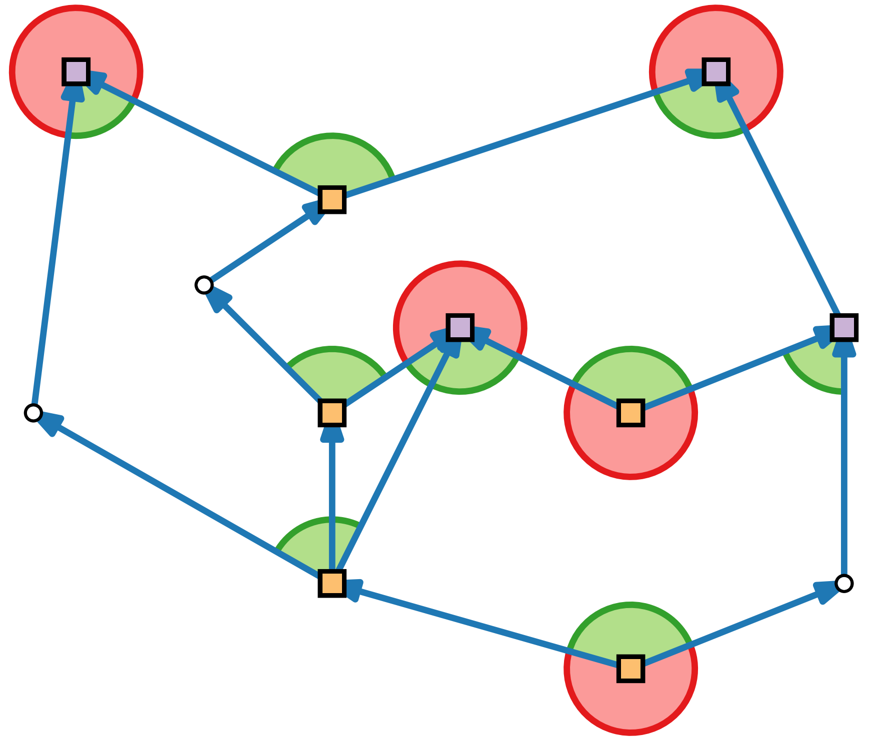
Angles, Local Sources & Sinks

Definitions.

- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.
- An angle α at a local source/sink is **large** if $\alpha > \pi$ and **small** otherwise.
- $L(v) = \#$ large angles at v
- $L(f) = \#$ large angles in f
- $S(v) = \#$ small angles at v
- $S(f) = \#$ small angles at f



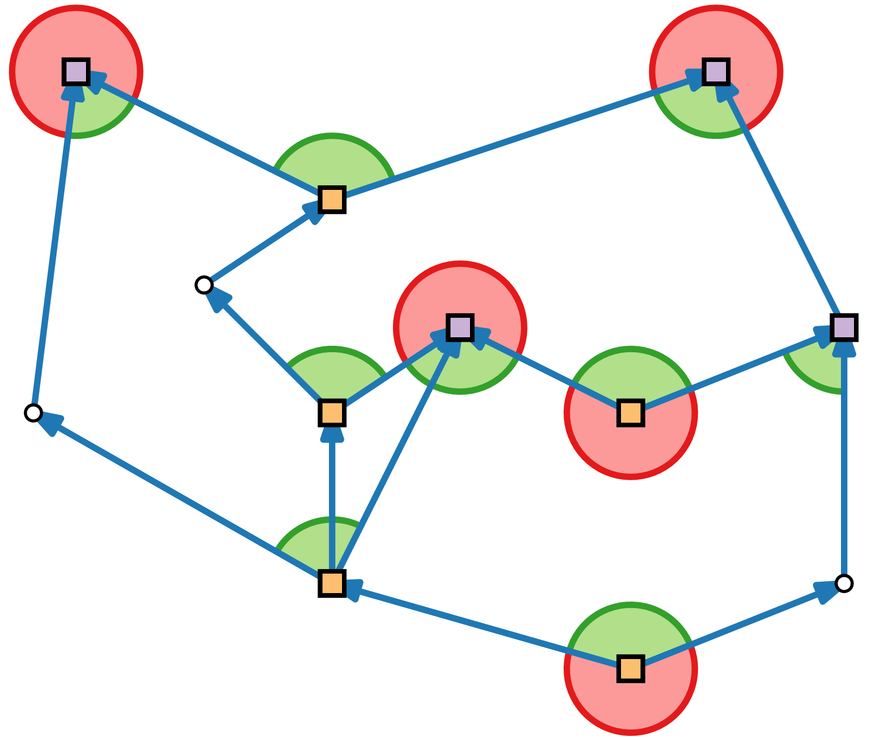
Angles, Local Sources & Sinks



Definitions.

- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.
- An angle α at a local source/sink is **large** if $\alpha > \pi$ and **small** otherwise.
- $L(v) = \#$ large angles at v
- $L(f) = \#$ large angles in f
- $S(v) = \#$ small angles at v
- $S(f) = \#$ small angles at f
- $A(f) = \#$ local sources w.r.t. to f

Angles, Local Sources & Sinks



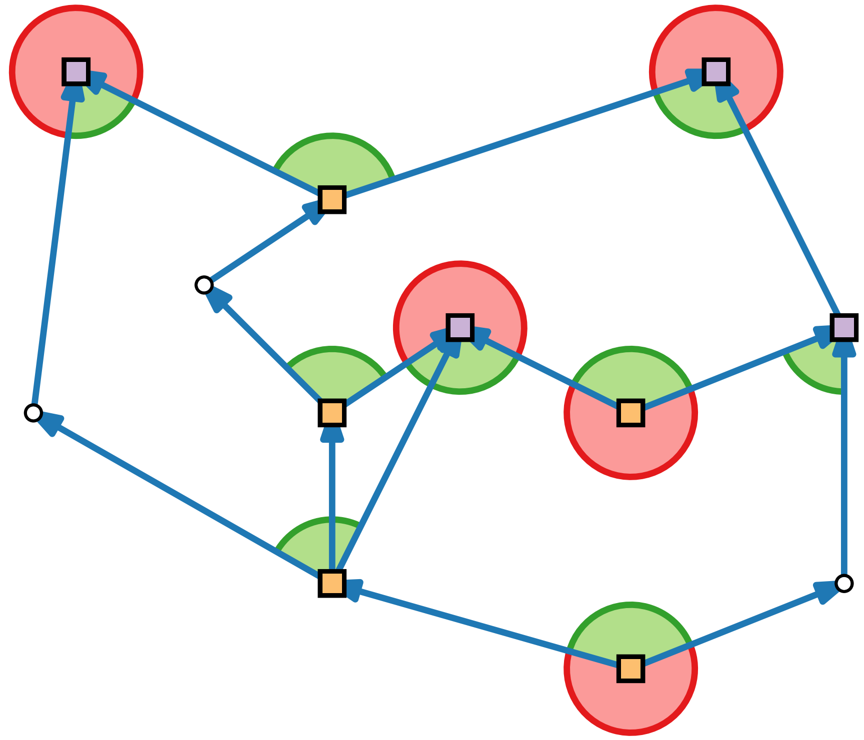
Definitions.

- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.
- An angle α at a local source/sink is **large** if $\alpha > \pi$ and **small** otherwise.
- $L(v) = \#$ large angles at v
- $L(f) = \#$ large angles in f
- $S(v) = \#$ small angles at v
- $S(f) = \#$ small angles at f
- $A(f) = \#$ local sources w.r.t. to f
  $= \#$ local sinks w.r.t. to f

Angles, Local Sources & Sinks

Definitions.

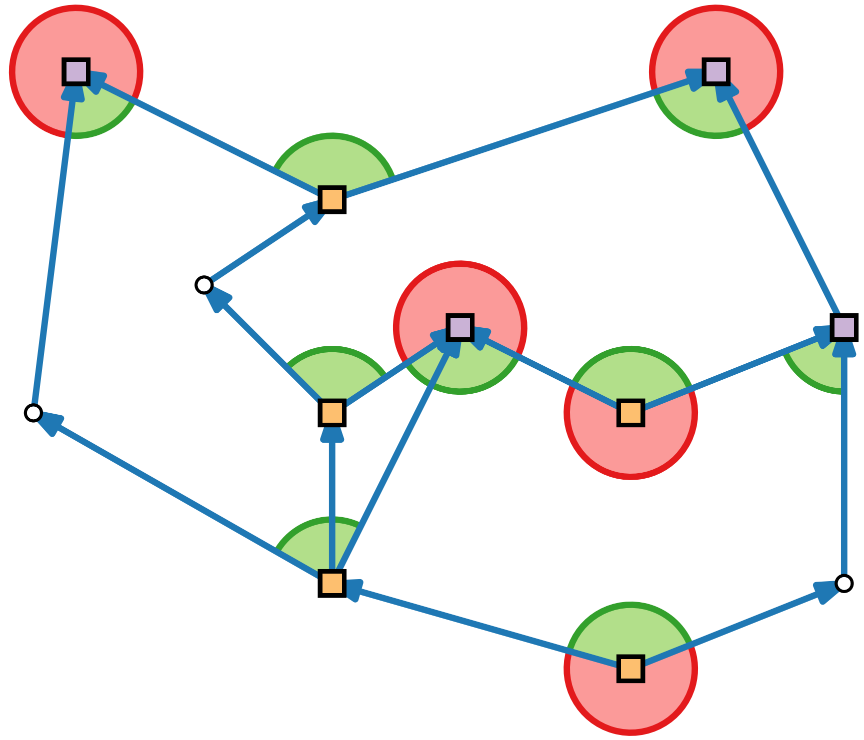
- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. ← boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.
- An angle α at a local source/sink is **large** if $\alpha > \pi$ and **small** otherwise.
- $L(v) = \#$ large angles at v
- $L(f) = \#$ large angles in f
- $S(v) = \#$ small angles at v
- $S(f) = \#$ small angles at f
- $A(f) = \#$ local sources w.r.t. to f
  $= \#$ local sinks w.r.t. to f



Angles, Local Sources & Sinks

Definitions.

- A vertex v is a **local source** w.r.t. to a face f if v has two outgoing edges on ∂f. boundary of f
- A vertex v is a **local sink** w.r.t. to a face f if v has two incoming edges on ∂f.
- An angle α at a local source/sink is **large** if $\alpha > \pi$ and **small** otherwise.
- $L(v)$ = # large angles at v
- $L(f)$ = # large angles in f
- $S(v)$ = # small angles at v
- $S(f)$ = # small angles at f
- $A(f)$ = # local sources w.r.t. to f
  = # local sinks w.r.t. to f



Lemma 1.
$L(f) + S(f) = 2A(f)$

Assignment Problem

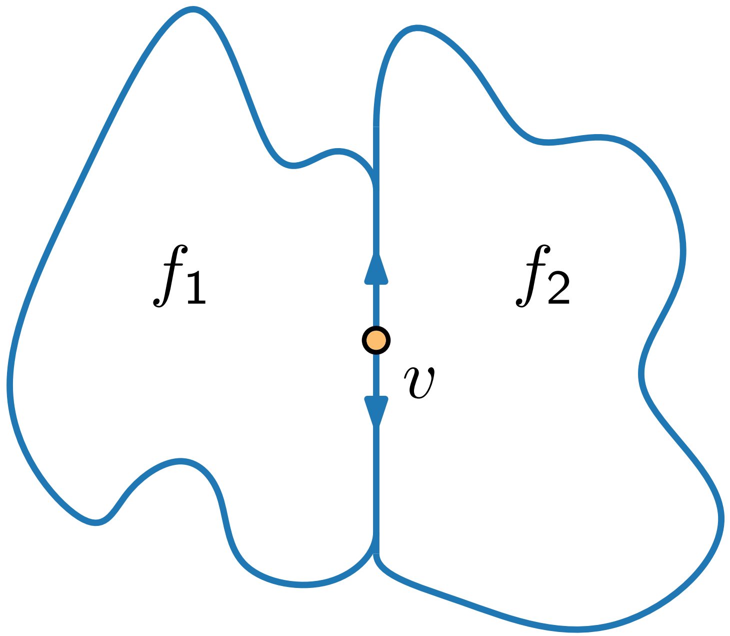
- Observe that the **global sources** and **global sinks** have precisely one large angle

Assignment Problem

- Observe that the **global sources** and **global sinks** have precisely one large angle
- All other vertices have only small angles.

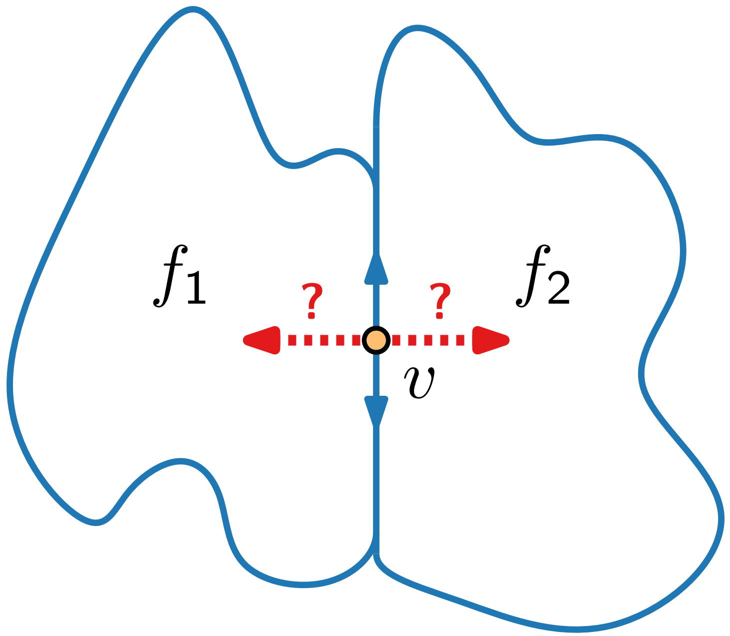
Assignment Problem

- Observe that the **global sources** and **global sinks** have precisely one large angle
- All other vertices have only small angles.
- Let v be a global source and let it be incident to faces f_1 and f_2.



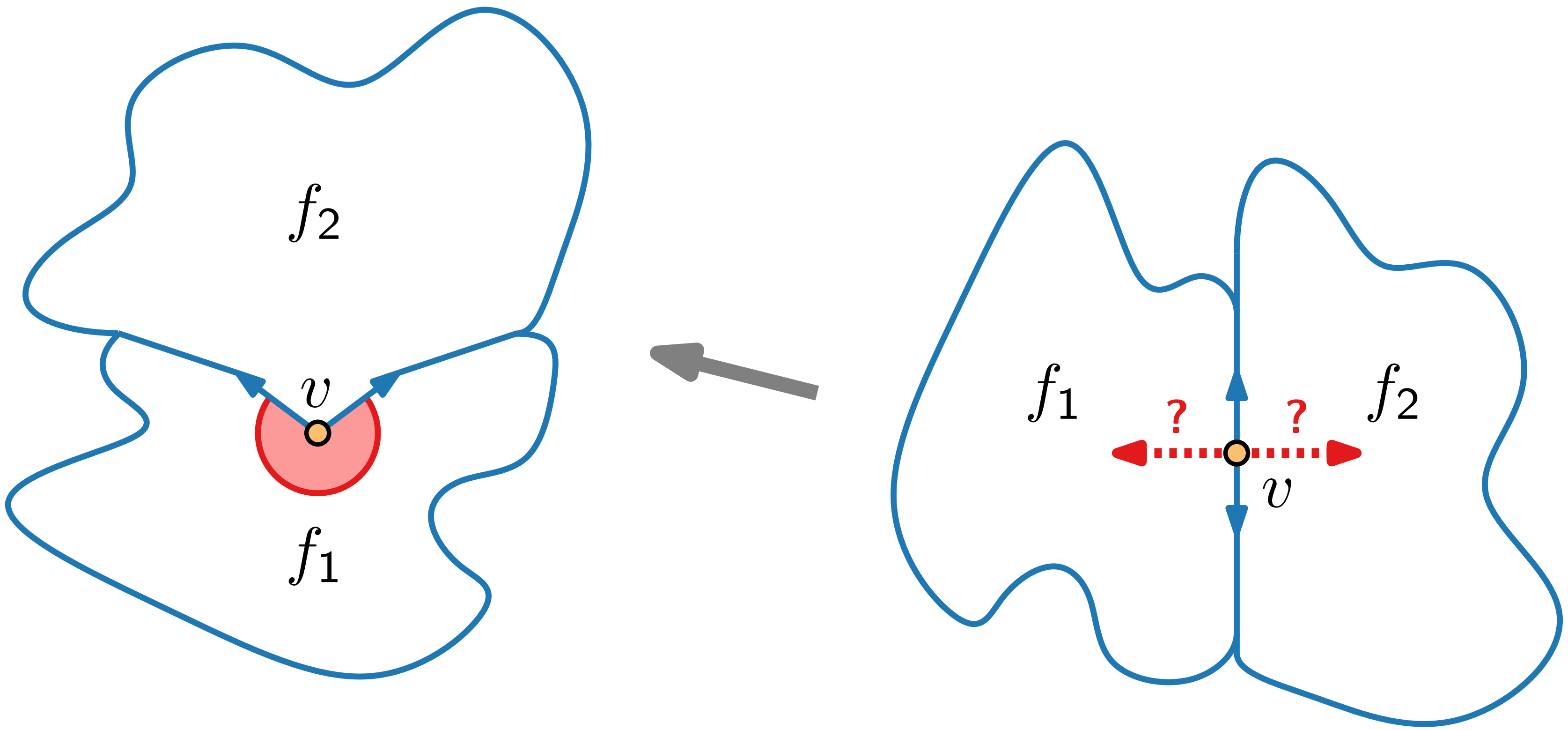
Assignment Problem

- Observe that the **global sources** and **global sinks** have precisely one large angle
- All other vertices have only small angles.
- Let v be a global source and let it be incident to faces f_1 and f_2.
- Does v have a large angle in f_1 or f_2?



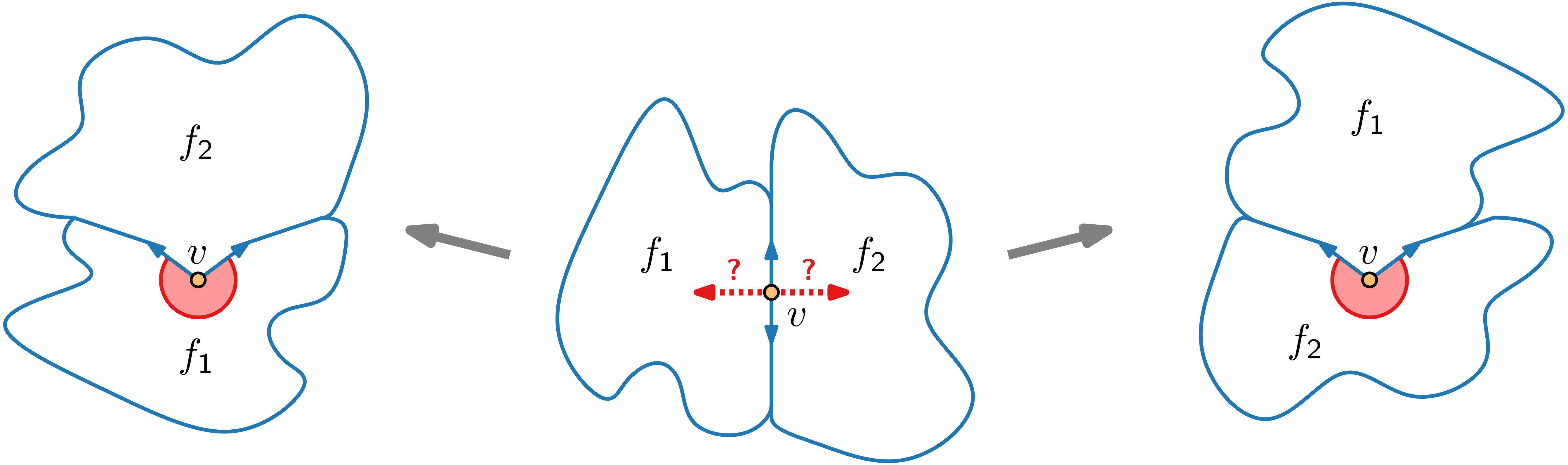
Assignment Problem

- Observe that the **global sources** and **global sinks** have precisely one large angle
- All other vertices have only small angles.
- Let v be a global source and let it be incident to faces f_1 and f_2.
- Does v have a large angle in f_1 or f_2?



Assignment Problem

- Observe that the **global sources** and **global sinks** have precisely one large angle
- All other vertices have only small angles.
- Let v be a global source and let it be incident to faces f_1 and f_2.
- Does v have a large angle in f_1 or f_2?



Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$

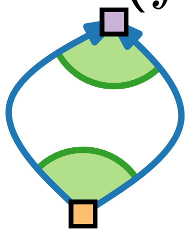
Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$

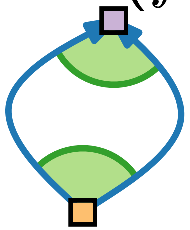


Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

Angle Relations

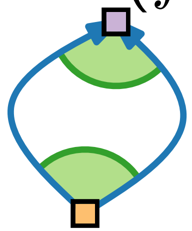
Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

- $L(f) \geq 1$

Proof by induction on $L(f)$.

- $L(f) = 0$



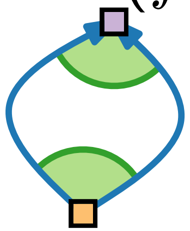
$\Rightarrow S(f) = 2$ ✓

Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

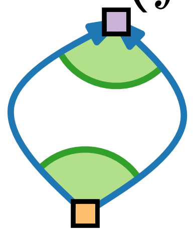
Split f with edge from a large angle at a "low" sink u to...

Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

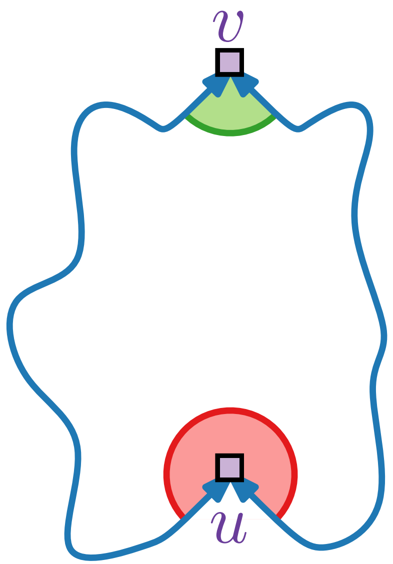
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a “low” sink u to...

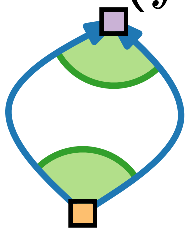
- sink v with small angle:



Angle Relations

Lemma 2.
$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

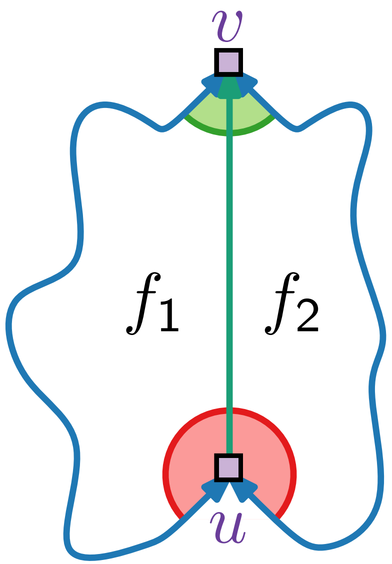
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

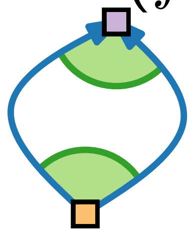
- sink v with small angle:



Angle Relations

Lemma 2.
$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

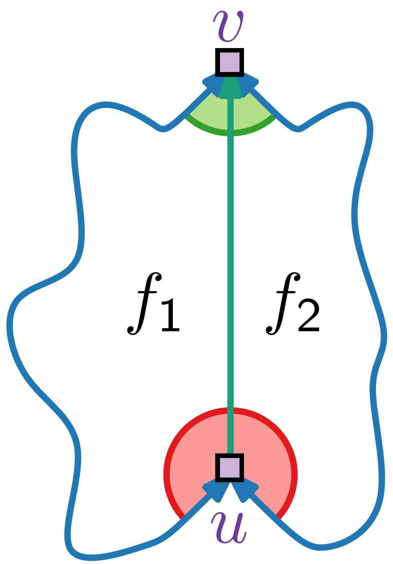
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- sink v with small angle:



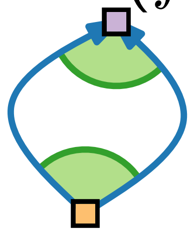
$L(f) - S(f)$

Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

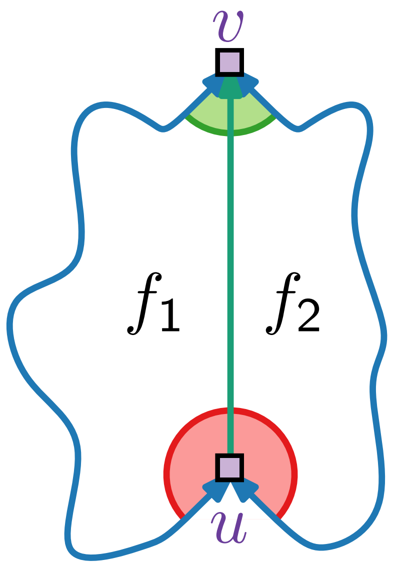
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- sink v with small angle:

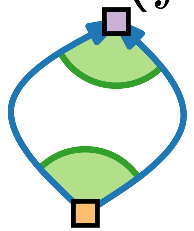


$$L(f) - S(f) = L(f_1) + L(f_2) + 1$$

Angle Relations

Lemma 2.
$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

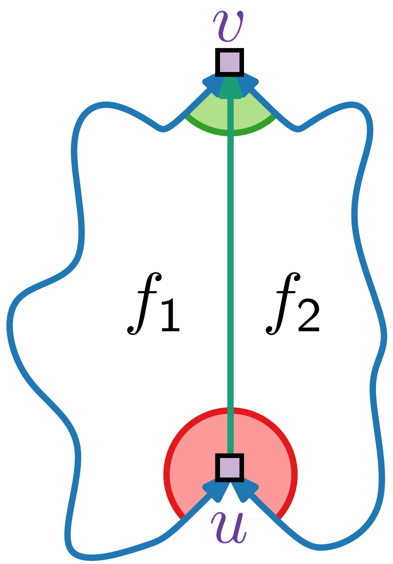
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- sink v with small angle:

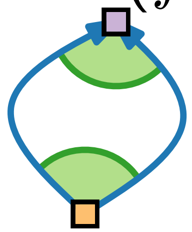


$$L(f) - S(f) = L(f_1) + L(f_2) + 1 \\ - (S(f_1) + S(f_2) - 1)$$

Angle Relations

Lemma 2. $L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$

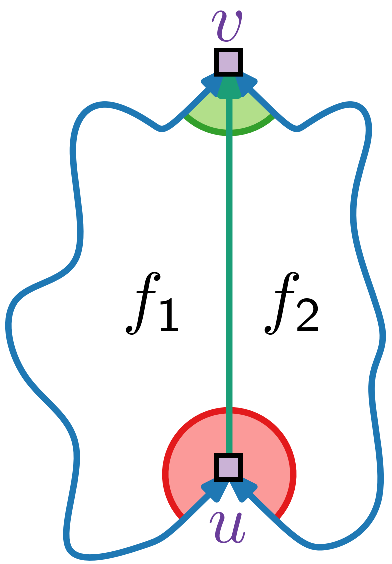
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- sink v with small angle:



$$\begin{aligned} & \qquad\qquad\quad\;\; -2 \qquad\quad -2 \\ L(f) - S(f) &= L(f_1) + L(f_2) + 1 \\ &\quad - (S(f_1) + S(f_2) - 1) \end{aligned}$$

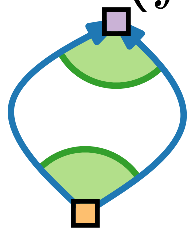
Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$

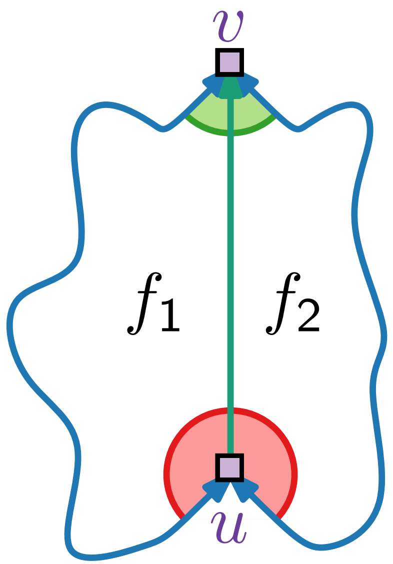


$\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- sink v with small angle:



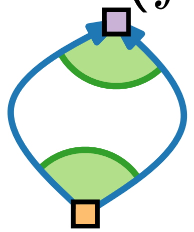
$$\begin{aligned} L(f) - S(f) &= \overbrace{L(f_1)}^{-2} + \overbrace{L(f_2)}^{-2} + 1 \\ &\quad - (S(f_1) + S(f_2) - 1) \\ &= -2 - 2 + 2 = -2 \end{aligned}$$

Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

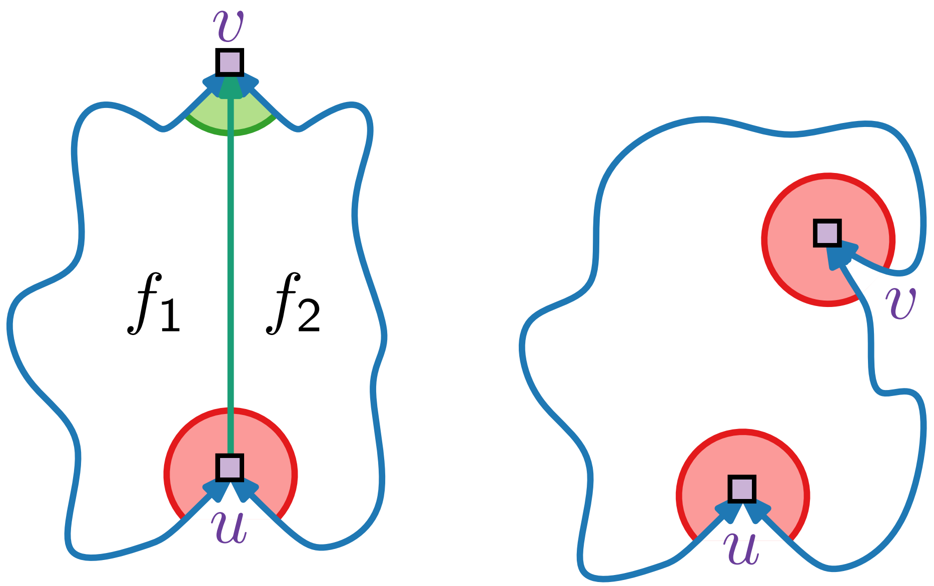
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a “low” sink u to...

- sink v with small/large angle:



$L(f) - S(f)$

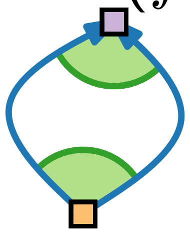
Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$

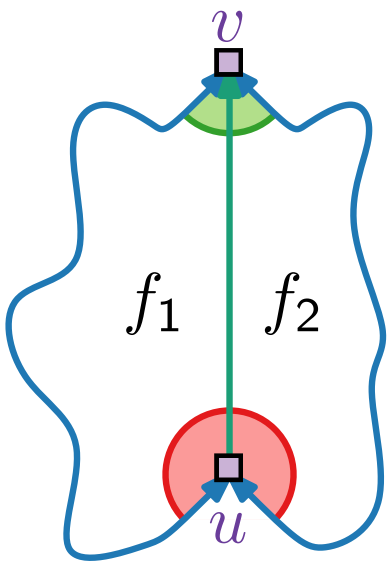


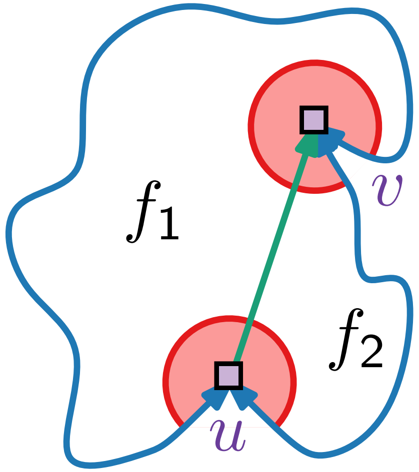
$\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- sink v with small/large angle:





$$L(f) - S(f) = L(f_1) + L(f_2) + 1 - (S(f_1) + S(f_2) - 1)$$

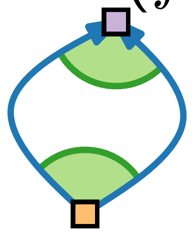
Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$

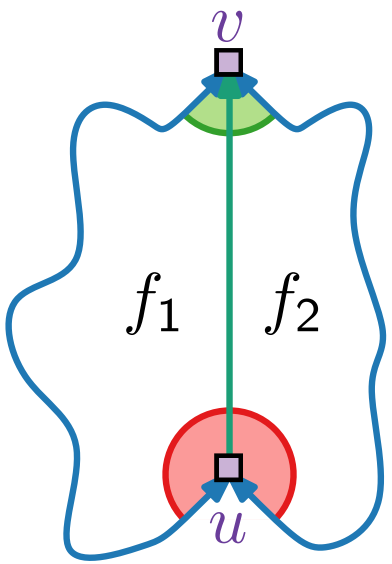


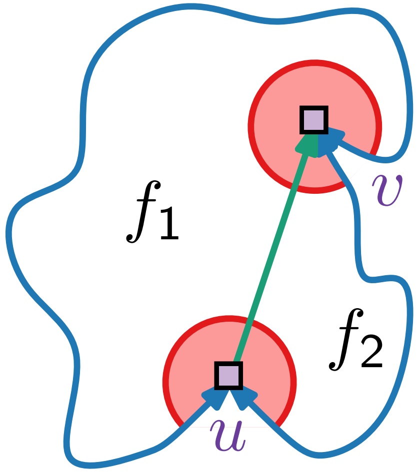
$\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- sink v with small/large angle:





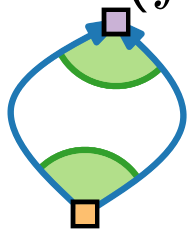
$$L(f) - S(f) = \overset{-2}{L(f_1)} + \overset{-2}{L(f_2)} + 1 - (S(f_1) + S(f_2) - 1)$$

Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

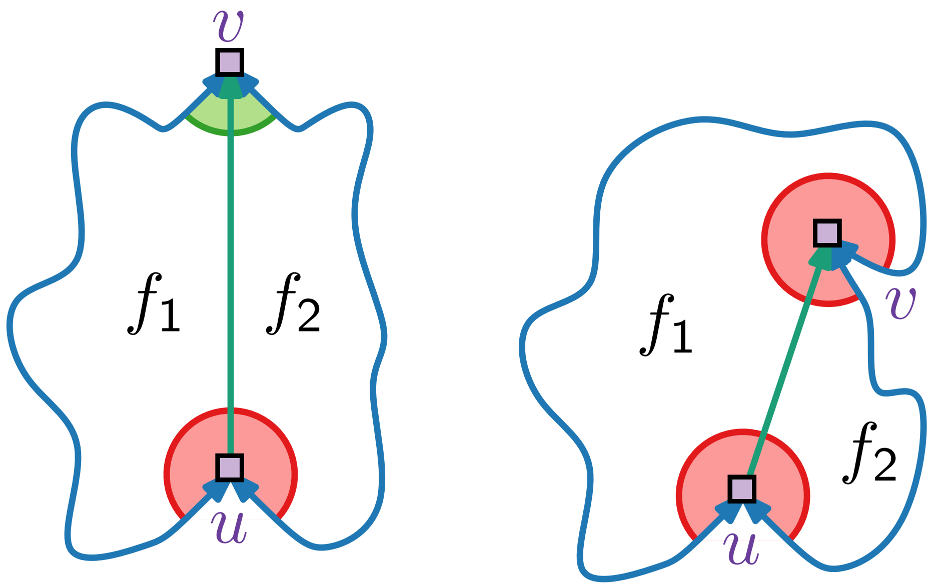
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- sink v with small/large angle:

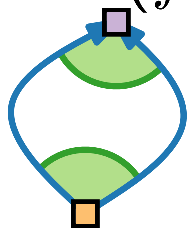


$$\begin{aligned} L(f) - S(f) &= \overset{-2}{L(f_1)} + \overset{-2}{L(f_2)} + 1 \\ &\quad - (S(f_1) + S(f_2) - 1) \\ &= -2 - 2 + 2 = -2 \end{aligned}$$

Angle Relations

Lemma 2.
$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

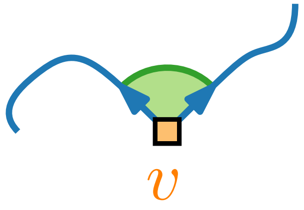
Proof by induction on $L(f)$.

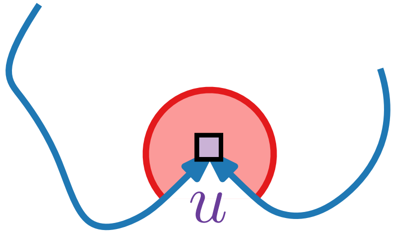
- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- source v with small angle:



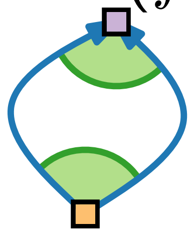


Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

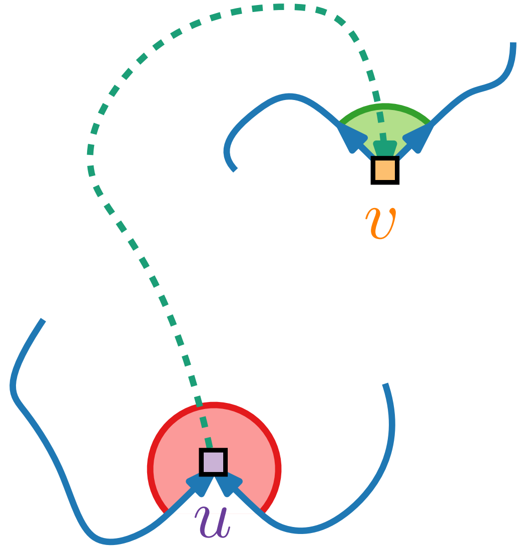
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a “low” sink u to...

- source v with small angle:

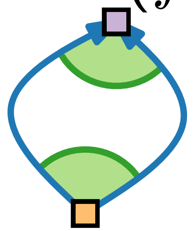


Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

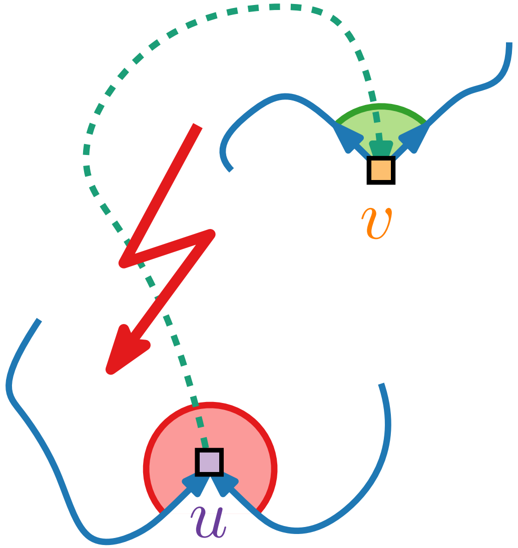
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- source v with ~~small~~ angle:

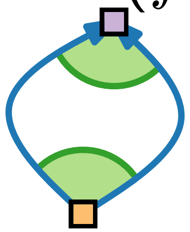


Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

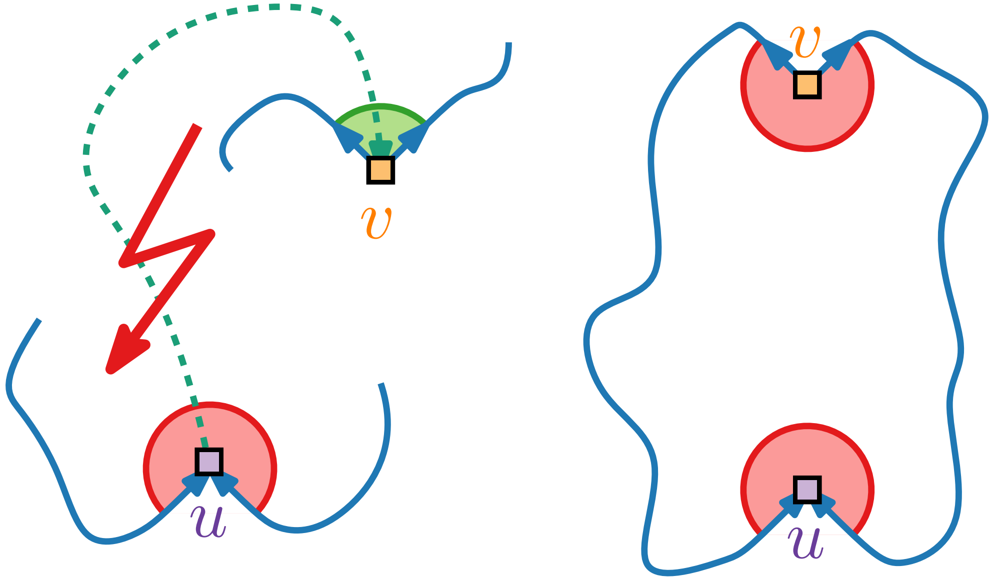
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- source v with ~~small~~/large angle:

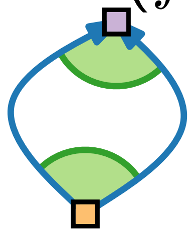


Angle Relations

Lemma 2.

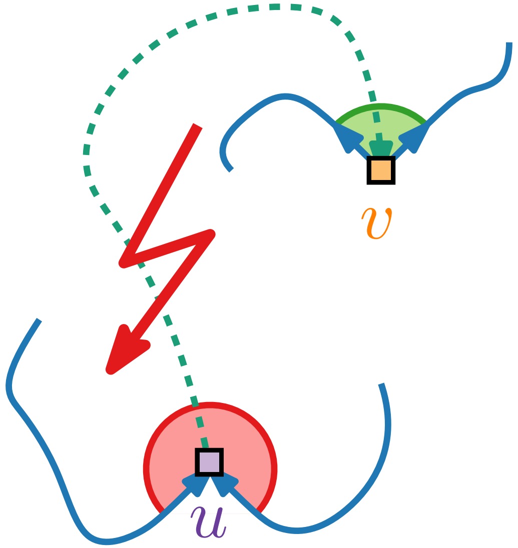
$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

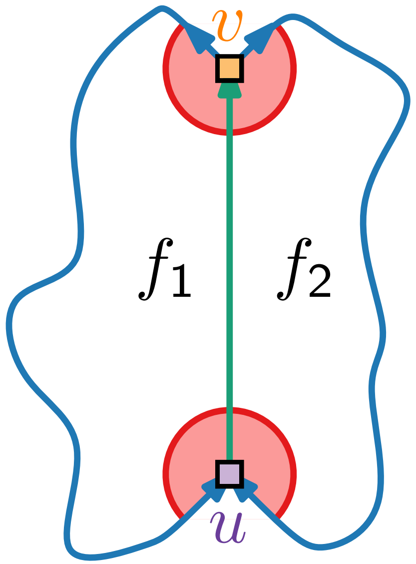
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓
- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- source v with ~~small~~/large angle:



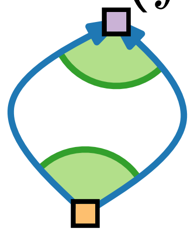


Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

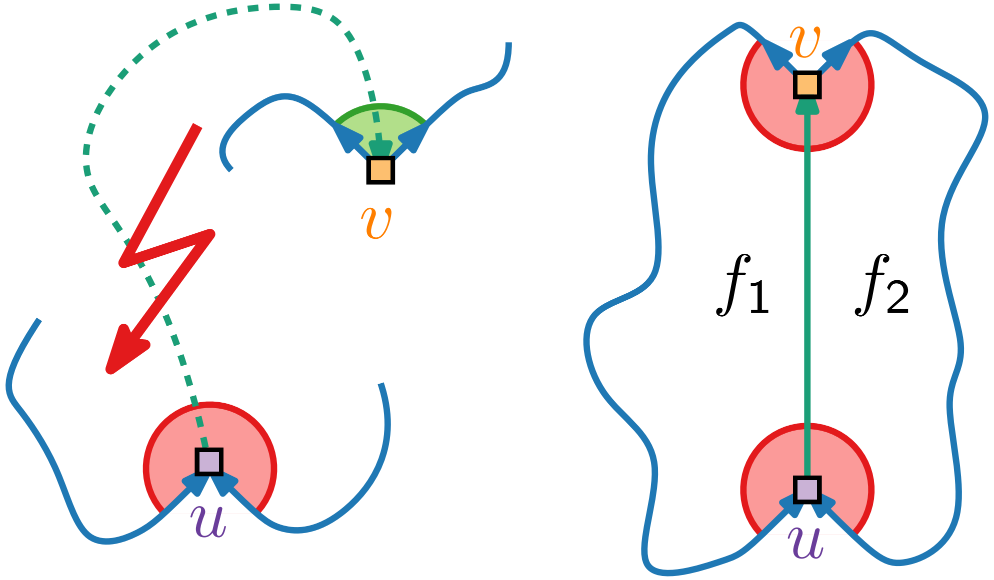
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- source v with ~~small~~/large angle:



$L(f) - S(f)$

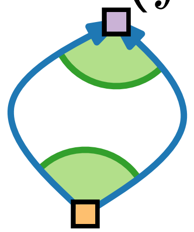
Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$

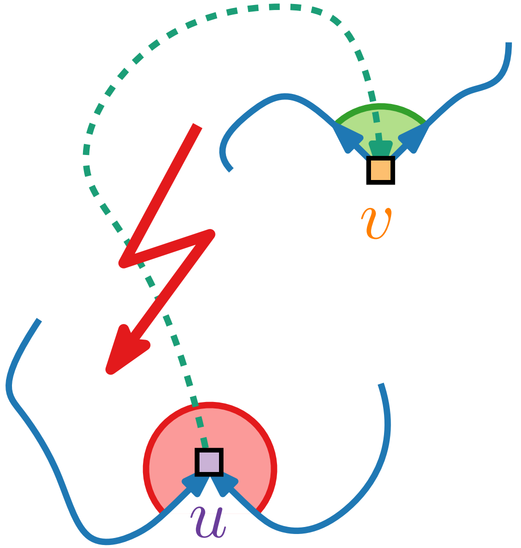


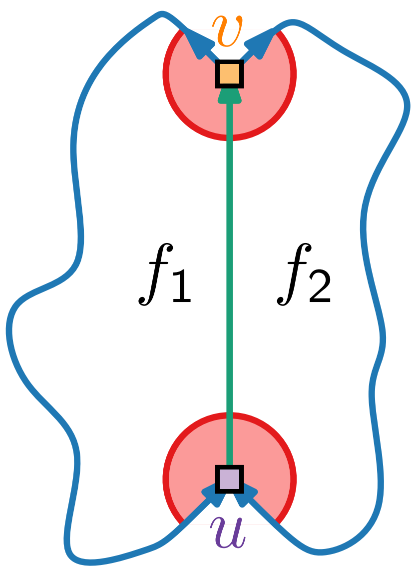
$\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a “low” sink u to...

- source v with ~~small~~/large angle:





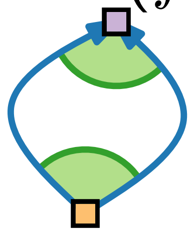
$$L(f) - S(f) = L(f_1) + L(f_2) + 2$$

Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

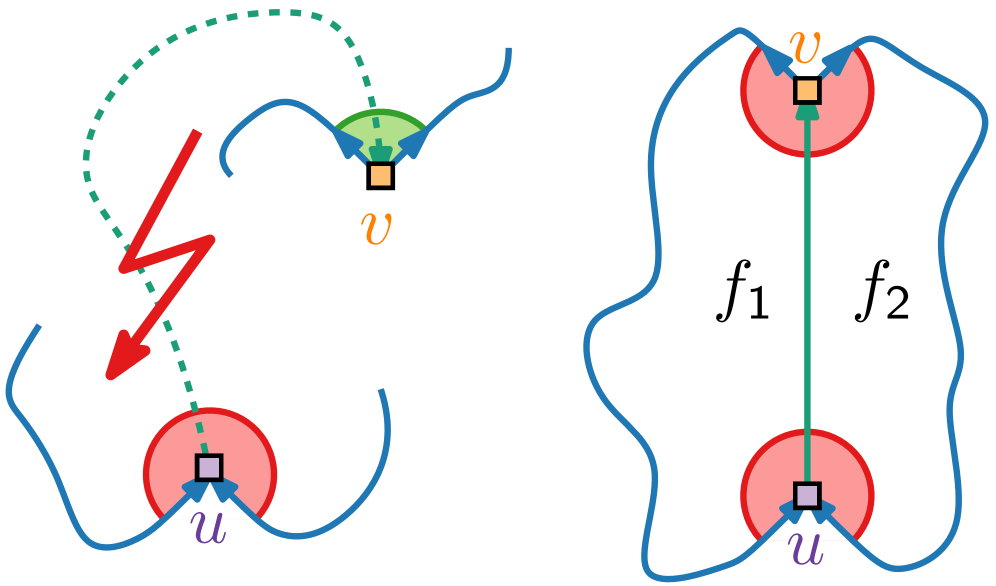
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- source v with ~~small~~/large angle:



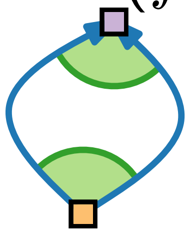
$$L(f) - S(f) = L(f_1) + L(f_2) + 2 - (S(f_1) + S(f_2))$$

Angle Relations

Lemma 2.
$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$

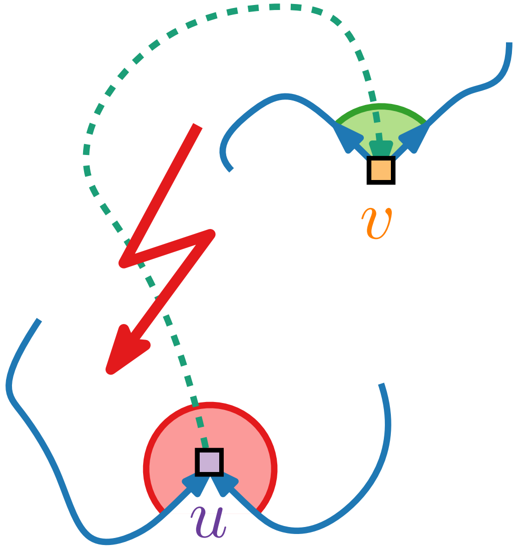


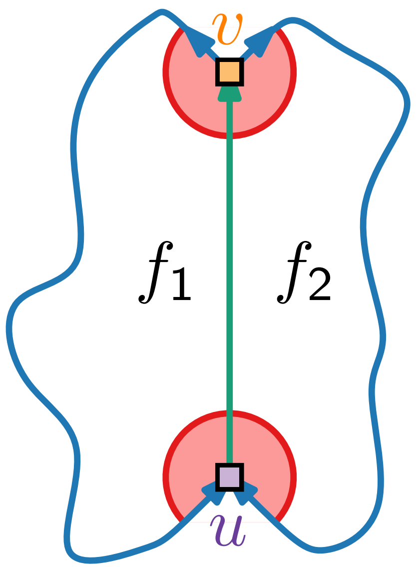
$\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- source v with ~~small~~/large angle:





$$L(f) - S(f) = \underbrace{L(f_1)}_{} + \underbrace{L(f_2)}_{} + 2 - (S(f_1) + S(f_2))$$

$$\overset{-2}{L(f_1) - S(f_1)} \qquad \overset{-2}{L(f_2) - S(f_2)}$$

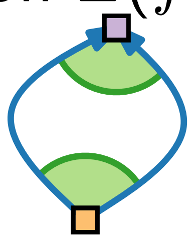
Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$

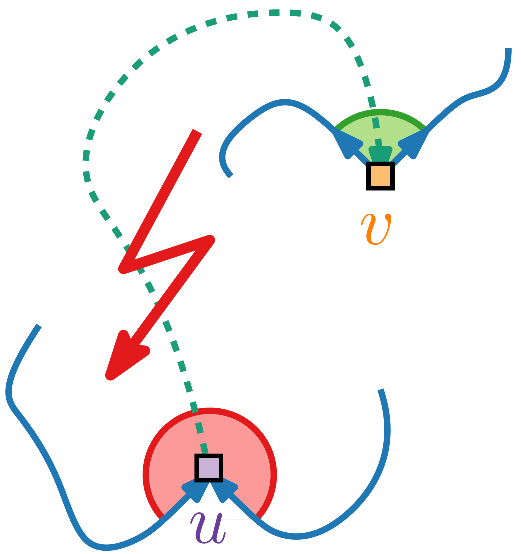


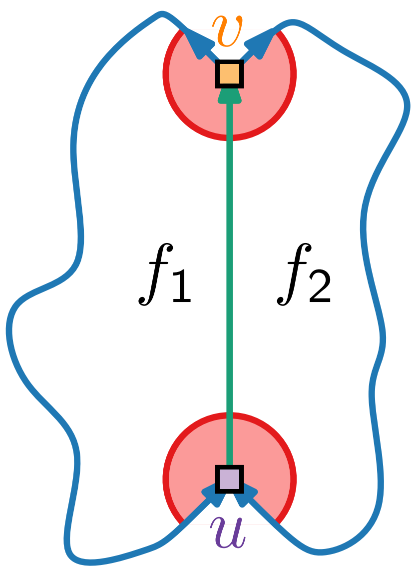
$\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a “low” sink u to...

- source v with ~~small~~/large angle:





$$\begin{aligned} L(f) - S(f) &= L(f_1) + L(f_2) + 2 \\ &\quad - (S(f_1) + S(f_2)) \\ &= -2 - 2 + 2 = -2 \end{aligned}$$

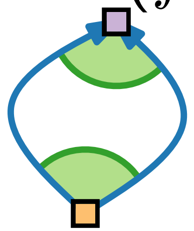
-2 -2

Angle Relations

Lemma 2.
$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$

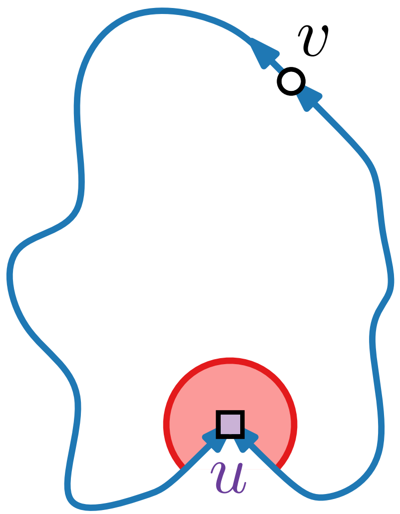


$\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- vertex v that is neither source nor sink:



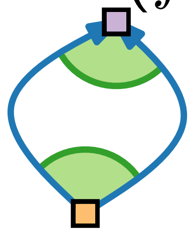
Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$

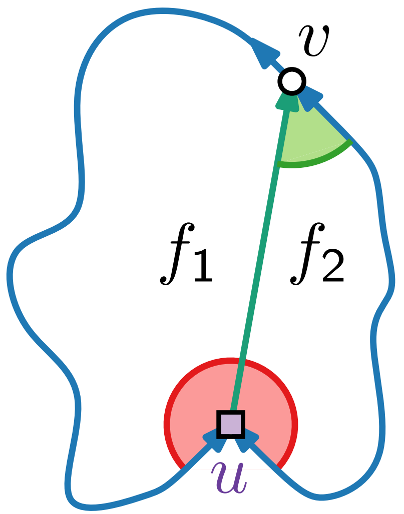


$\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- vertex v that is neither source nor sink:

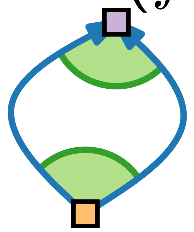


Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

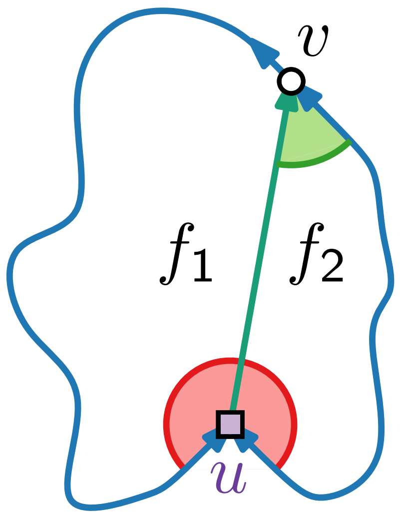
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- vertex v that is neither source nor sink:



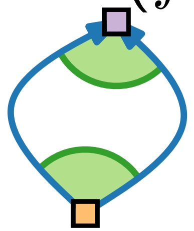
$L(f) - S(f)$

Angle Relations

Lemma 2.
$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$

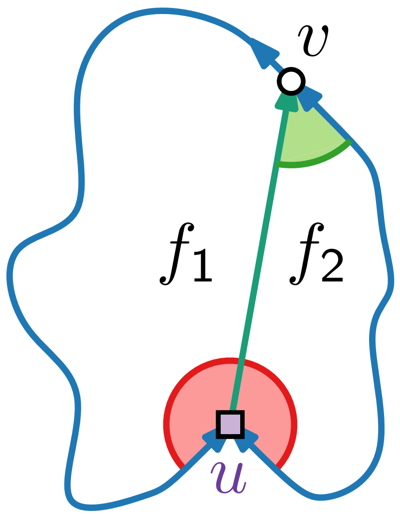


$\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- vertex v that is neither source nor sink:



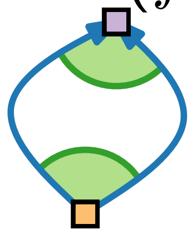
$$L(f) - S(f) = L(f_1) + L(f_2) + 1$$

Angle Relations

Lemma 2.

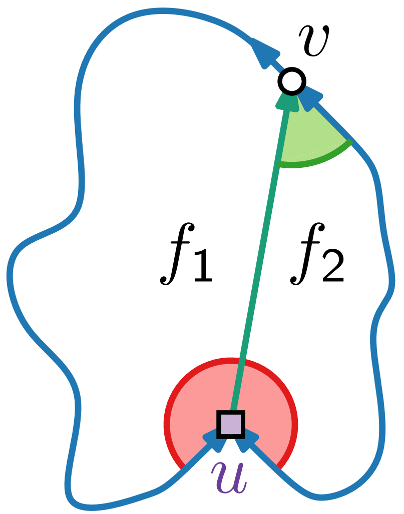
$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓
- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- vertex v that is neither source nor sink:



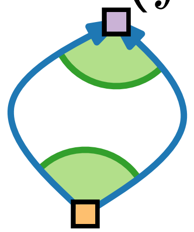
$$L(f) - S(f) = L(f_1) + L(f_2) + 1 \\ - (S(f_1) + S(f_2) - 1)$$

Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

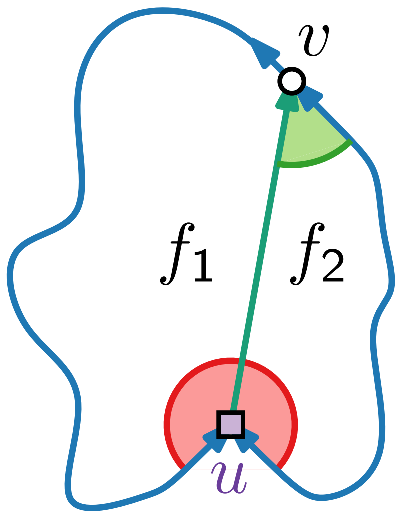
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- vertex v that is neither source nor sink:



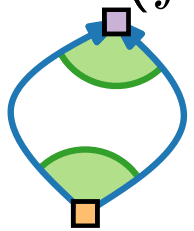
$$L(f) - S(f) = \overbrace{L(f_1)}^{-2} + \overbrace{L(f_2)}^{-2} + 1 - (S(f_1) + S(f_2) - 1)$$

Angle Relations

Lemma 2.

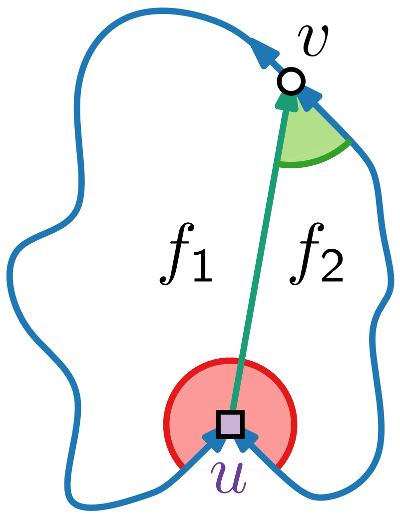
$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓
- $L(f) \geq 1$

Split f with edge from a large angle at a “low” sink u to...

- vertex v that is neither source nor sink:



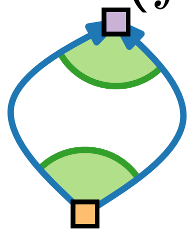
$$\begin{aligned} L(f) - S(f) &= \underbrace{L(f_1)}^{-2} + \underbrace{L(f_2)}^{-2} + 1 \\ &\quad - (S(f_1) + S(f_2) - 1) \\ &= -2 - 2 + 2 = -2 \end{aligned}$$

Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

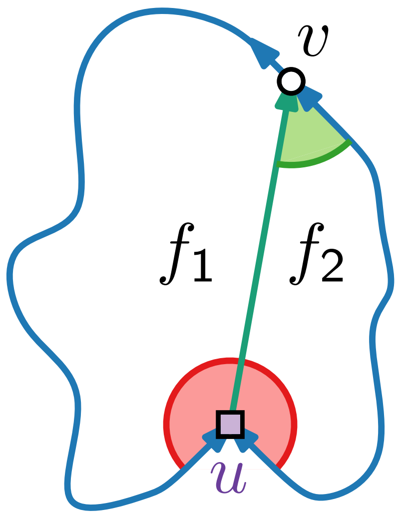
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- vertex v that is neither source nor sink:



-2 -2

$$\begin{aligned} L(f) - S(f) &= L(f_1) + L(f_2) + 1 \\ &\quad - (S(f_1) + S(f_2) - 1) \\ &= -2 - 2 + 2 = -2 \end{aligned}$$

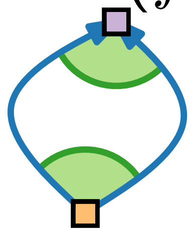
- Otherwise "high" source u exists. $\rightarrow$ symmetric

Angle Relations

Lemma 2.

$$L(f) - S(f) = \begin{cases} -2 & \text{if } f \neq f_0, \\ +2 & \text{if } f = f_0. \end{cases}$$

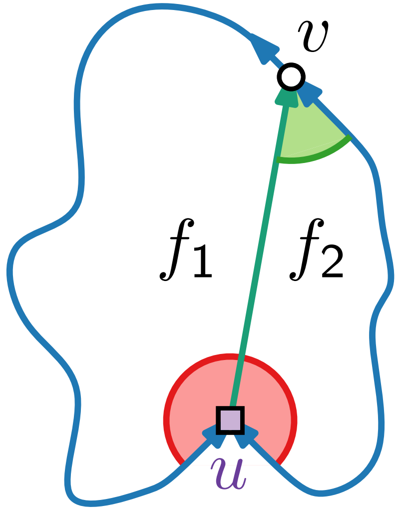
Proof by induction on $L(f)$.

- $L(f) = 0$  $\Rightarrow S(f) = 2$ ✓

- $L(f) \geq 1$

Split f with edge from a large angle at a "low" sink u to...

- vertex v that is neither source nor sink:



$$L(f) - S(f) = \underbrace{L(f_1)}_{-2} + \underbrace{L(f_2)}_{-2} + 1 - (S(f_1) + S(f_2) - 1)$$
$$= -2 - 2 + 2 = -2$$

- Otherwise "high" source u exists. $\rightarrow$ symmetric
- Similar argument for the outer face f_0.

Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

- for each vertex v: $L(v) = \begin{cases} 0 \\ 1 \end{cases}$

Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \end{cases}$

Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$

Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$
- for each face f: $L(f) =$

Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$
- for each face f: $L(f) = \begin{cases} A(f) - 1 & \text{if } f \neq f_0, \\ & \end{cases}$

Number of Large Angles

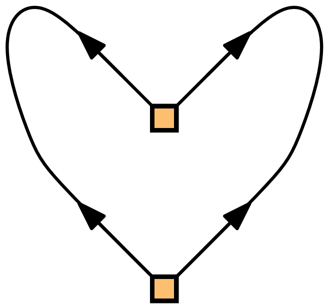
Lemma 3.
In every upward planar drawing of G, it holds that

- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$
- for each face f: $L(f) = \begin{cases} A(f) - 1 & \text{if } f \neq f_0, \\ A(f) + 1 & \text{if } f = f_0. \end{cases}$

Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

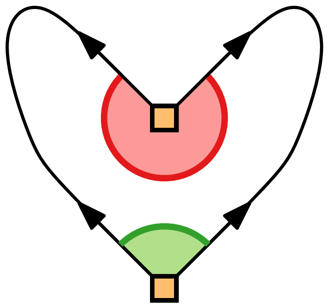
- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$
- for each face f: $L(f) = \begin{cases} A(f) - 1 & \text{if } f \neq f_0, \\ A(f) + 1 & \text{if } f = f_0. \end{cases}$



Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

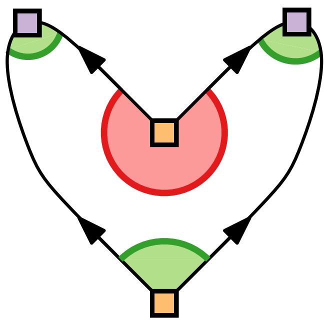
- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$
- for each face f: $L(f) = \begin{cases} A(f) - 1 & \text{if } f \neq f_0, \\ A(f) + 1 & \text{if } f = f_0. \end{cases}$



Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

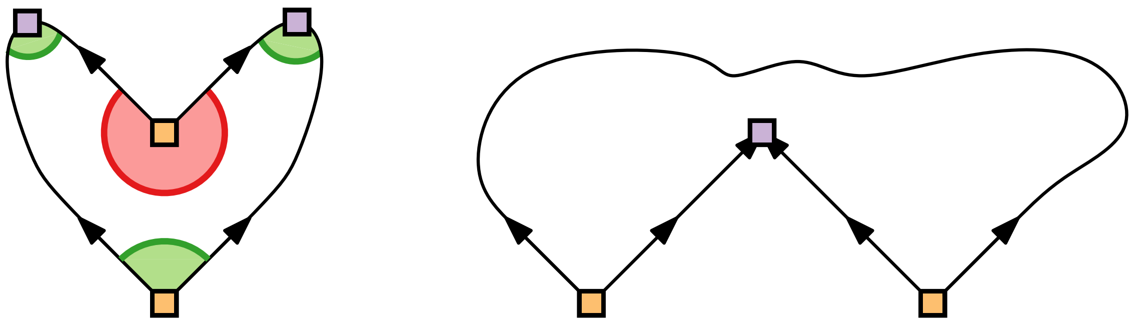
- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$
- for each face f: $L(f) = \begin{cases} A(f) - 1 & \text{if } f \neq f_0, \\ A(f) + 1 & \text{if } f = f_0. \end{cases}$



Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

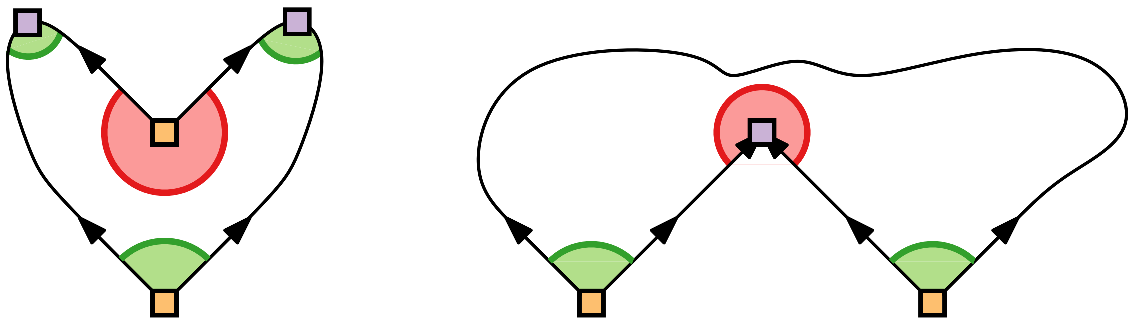
- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$
- for each face f: $L(f) = \begin{cases} A(f) - 1 & \text{if } f \neq f_0, \\ A(f) + 1 & \text{if } f = f_0. \end{cases}$



Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$
- for each face f: $L(f) = \begin{cases} A(f) - 1 & \text{if } f \neq f_0, \\ A(f) + 1 & \text{if } f = f_0. \end{cases}$

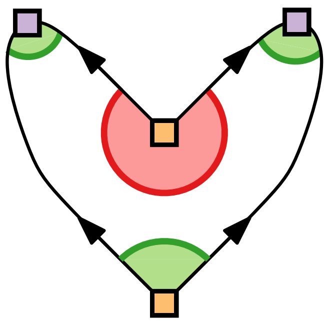


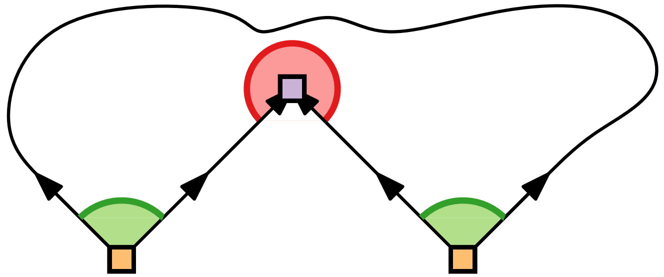
Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$
- for each face f: $L(f) = \begin{cases} A(f) - 1 & \text{if } f \neq f_0, \\ A(f) + 1 & \text{if } f = f_0. \end{cases}$

Proof.



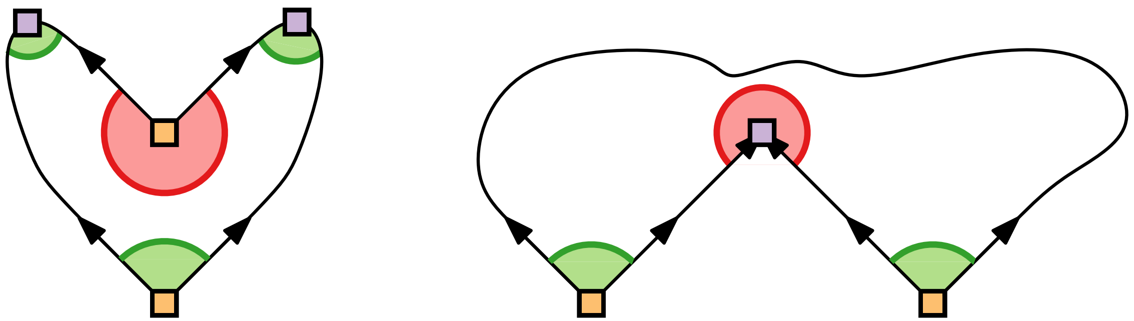


Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$
- for each face f: $L(f) = \begin{cases} A(f) - 1 & \text{if } f \neq f_0, \\ A(f) + 1 & \text{if } f = f_0. \end{cases}$

Proof. Lemma 1: $L(f) + S(f) = 2A(f)$

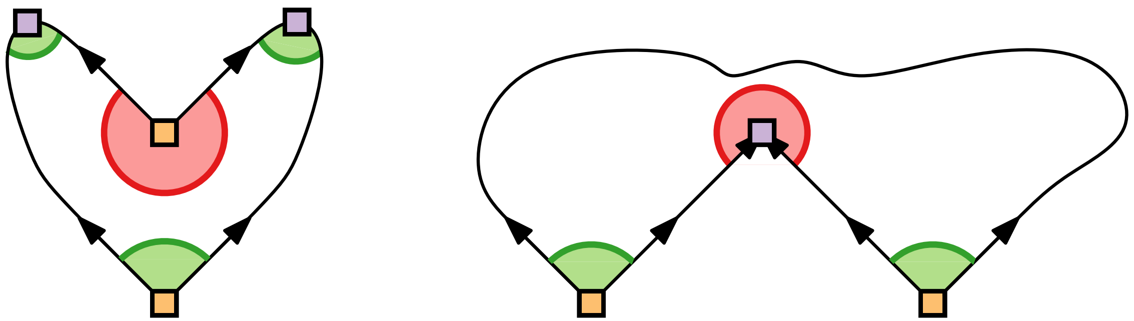


Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$
- for each face f: $L(f) = \begin{cases} A(f) - 1 & \text{if } f \neq f_0, \\ A(f) + 1 & \text{if } f = f_0. \end{cases}$

Proof. Lemma 1: $L(f) + S(f) = 2A(f)$
Lemma 2: $L(f) - S(f) = \pm 2.$

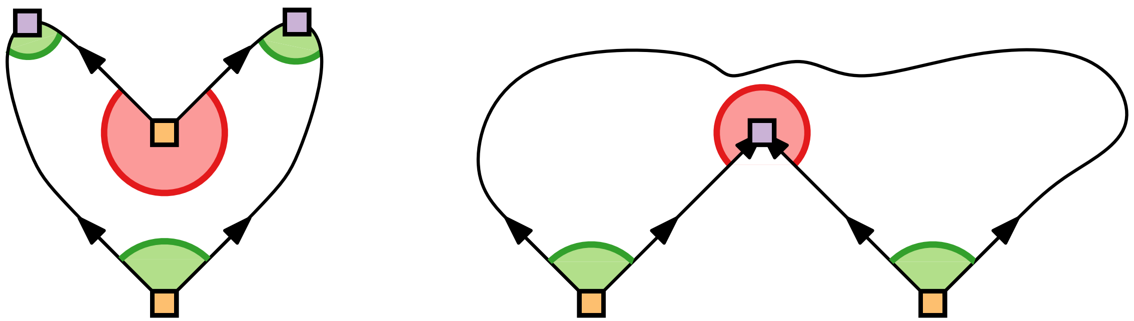


Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$
- for each face f: $L(f) = \begin{cases} A(f) - 1 & \text{if } f \neq f_0, \\ A(f) + 1 & \text{if } f = f_0. \end{cases}$

Proof. Lemma 1: $L(f) + S(f) = 2A(f)$
Lemma 2: $L(f) - S(f) = \pm 2.$



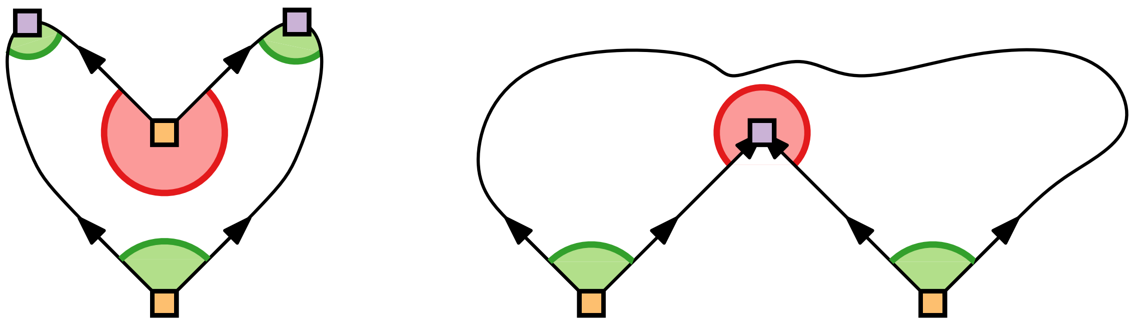
Number of Large Angles

Lemma 3.
In every upward planar drawing of G, it holds that

- for each vertex v: $L(v) = \begin{cases} 0 & \text{if } v \text{ is an inner vertex,} \\ 1 & \text{if } v \text{ is a gobal source / sink;} \end{cases}$
- for each face f: $L(f) = \begin{cases} A(f) - 1 & \text{if } f \neq f_0, \\ A(f) + 1 & \text{if } f = f_0. \end{cases}$

Proof. Lemma 1: $L(f) + S(f) = 2A(f)$
Lemma 2: $L(f) - S(f) = \pm 2.$

$\Rightarrow 2L(f) = 2A(f) \pm 2.$



Assignment of Large Angles to Faces

Let S be the set of (global) sources, and let T be the set of (global) sinks.

Assignment of Large Angles to Faces

Let S be the set of (global) sources, and let T be the set of (global) sinks.

Definition.
A **consistent assignment** $\Phi\colon S \cup T \to F$ is a mapping with

Assignment of Large Angles to Faces

Let S be the set of (global) sources, and let T be the set of (global) sinks.

Definition.
A **consistent assignment** $\Phi\colon S \cup T \to F$ is a mapping with

$$\Phi\colon v \mapsto \text{ incident face, where } v \text{ forms a large angle}$$

such that

Assignment of Large Angles to Faces

Let S be the set of (global) sources, and let T be the set of (global) sinks.

Definition.
A **consistent assignment** $\Phi\colon S \cup T \to F$ is a mapping with

$$\Phi\colon v \mapsto \text{ incident face, where } v \text{ forms a large angle}$$

such that

$$|\Phi^{-1}(f)| =$$

Assignment of Large Angles to Faces

Let S be the set of (global) sources, and let T be the set of (global) sinks.

Definition.
A **consistent assignment** $\Phi\colon S \cup T \to F$ is a mapping with

$$\Phi\colon v \mapsto \text{ incident face, where } v \text{ forms a large angle}$$

such that

$$|\Phi^{-1}(f)| = L(f) =$$

Assignment of Large Angles to Faces

Let S be the set of (global) sources, and let T be the set of (global) sinks.

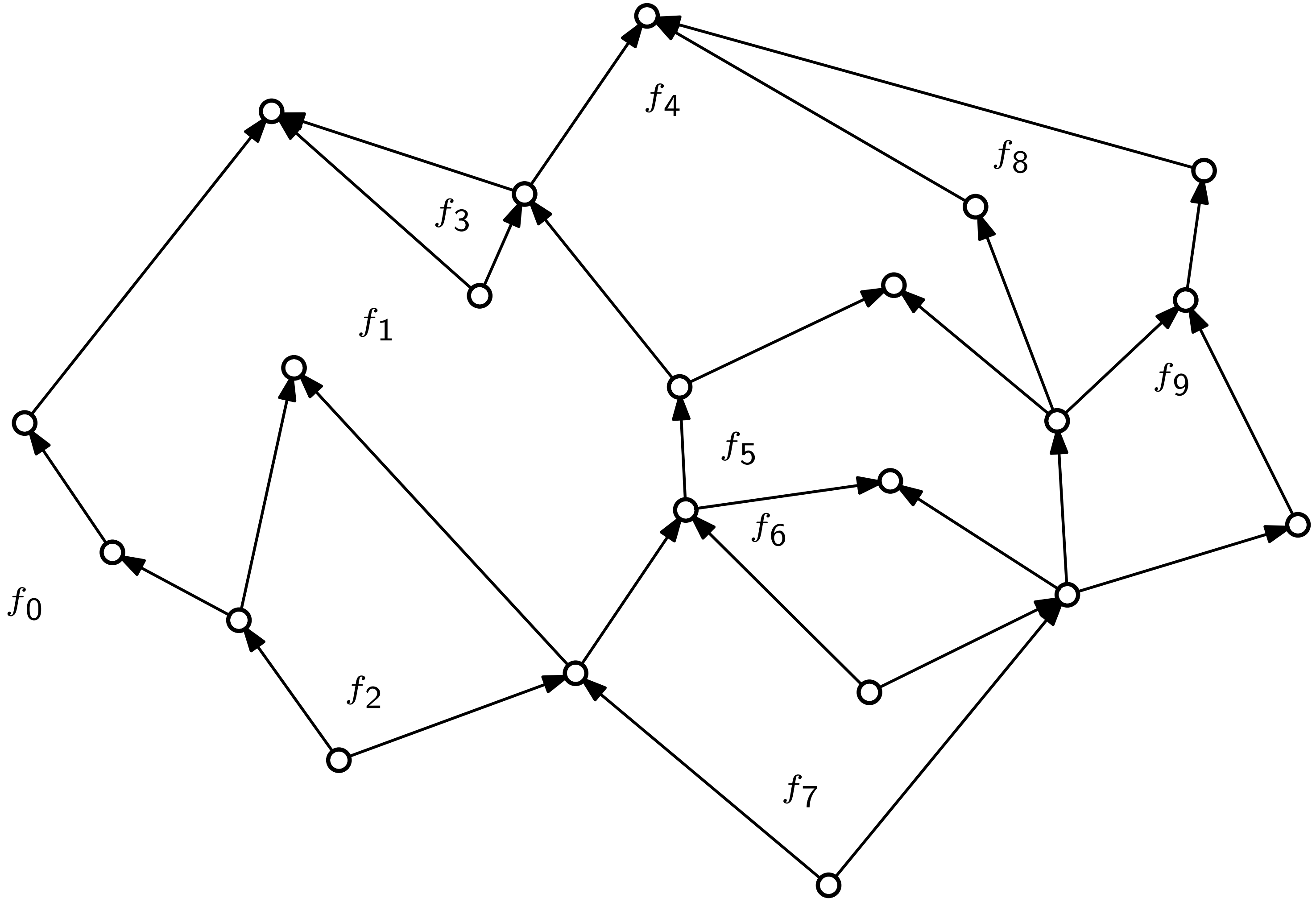
Definition.
A **consistent assignment** $\Phi \colon S \cup T \to F$ is a mapping with

$$\Phi \colon v \mapsto \text{ incident face, where } v \text{ forms a large angle}$$

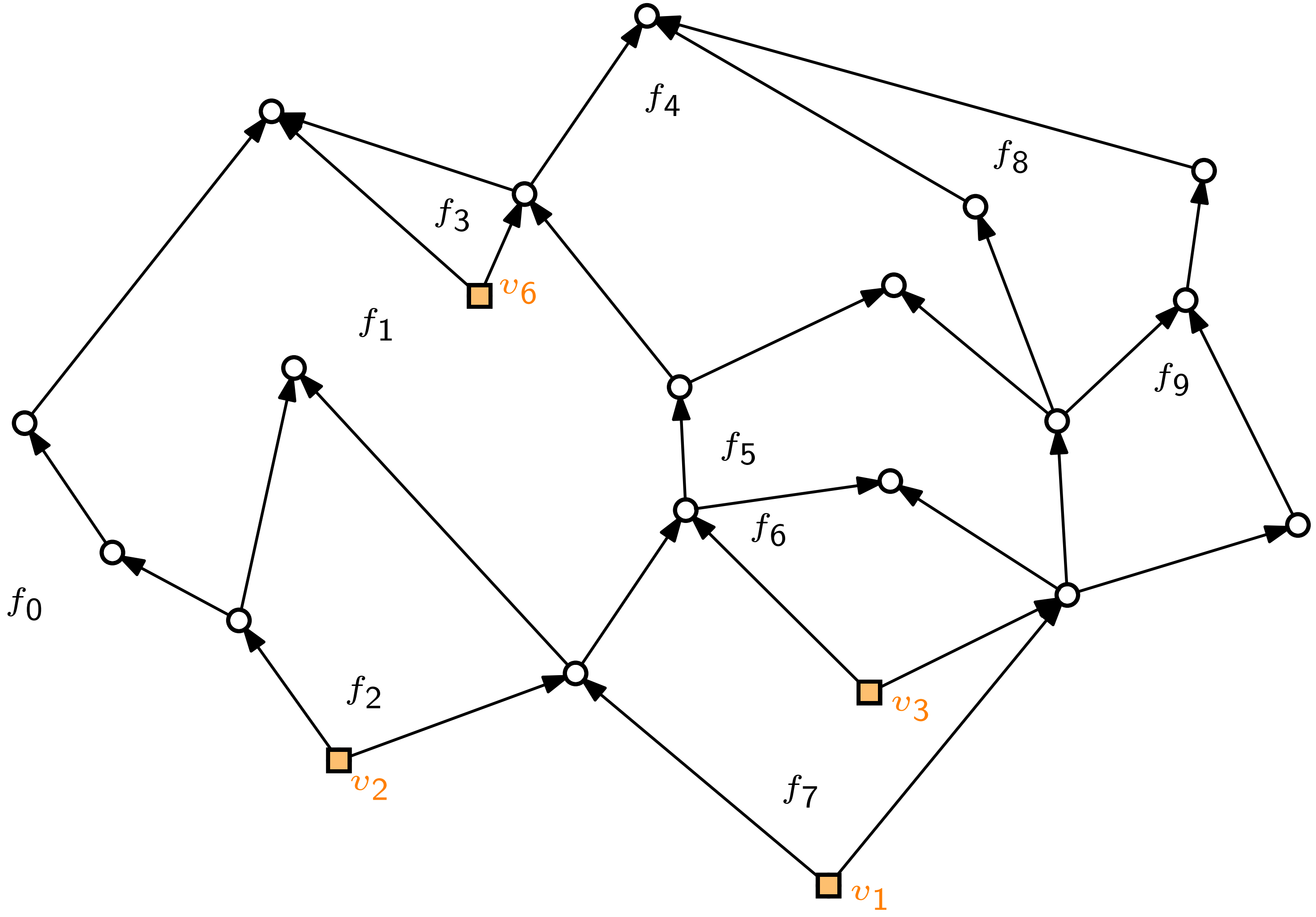
such that

$$|\Phi^{-1}(f)| = L(f) = \begin{cases} A(f) - 1 & \text{if } f \neq f_0, \\ A(f) + 1 & \text{if } f = f_0. \end{cases}$$

Example of Angle-to-Face Assignment

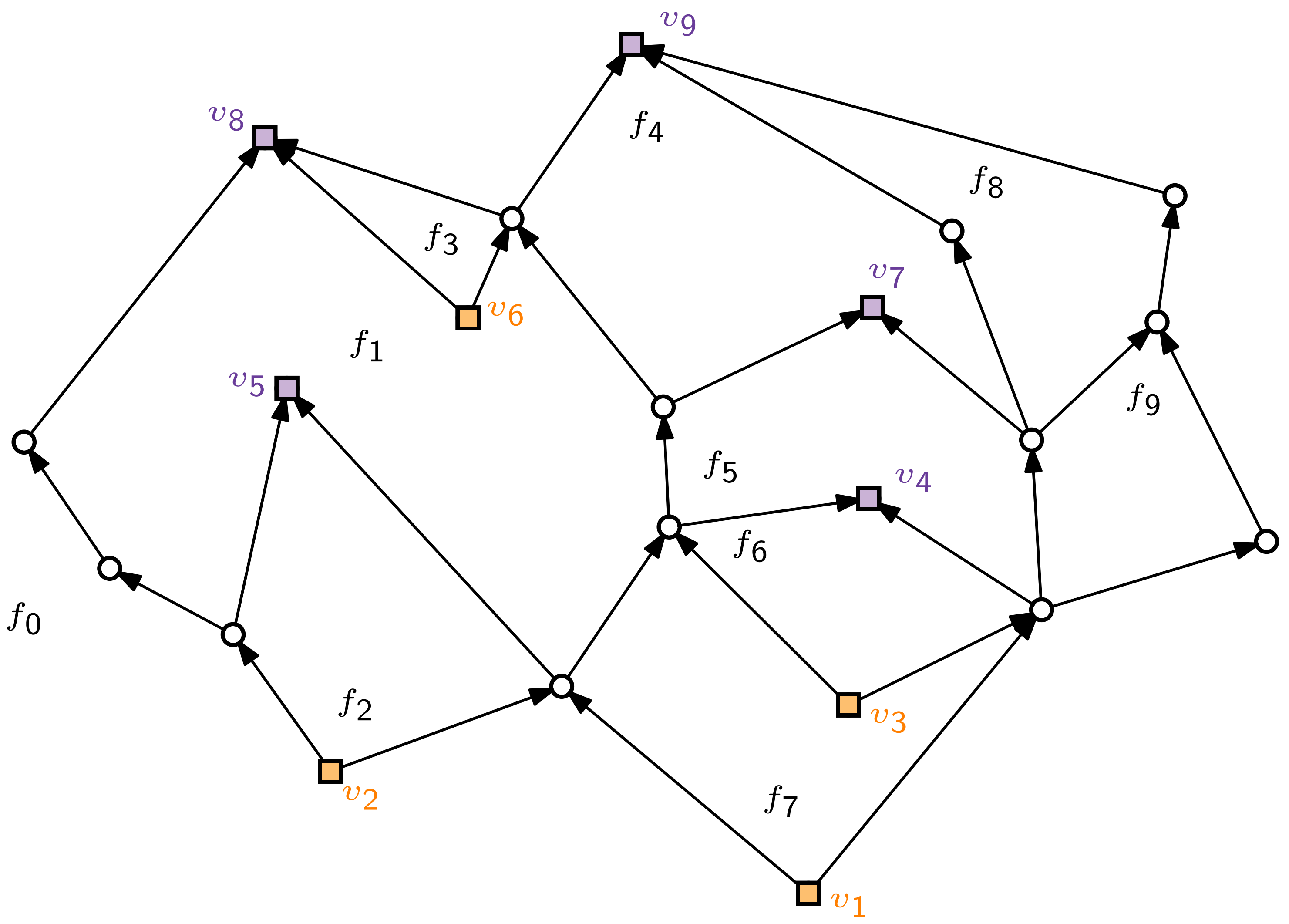


Example of Angle-to-Face Assignment



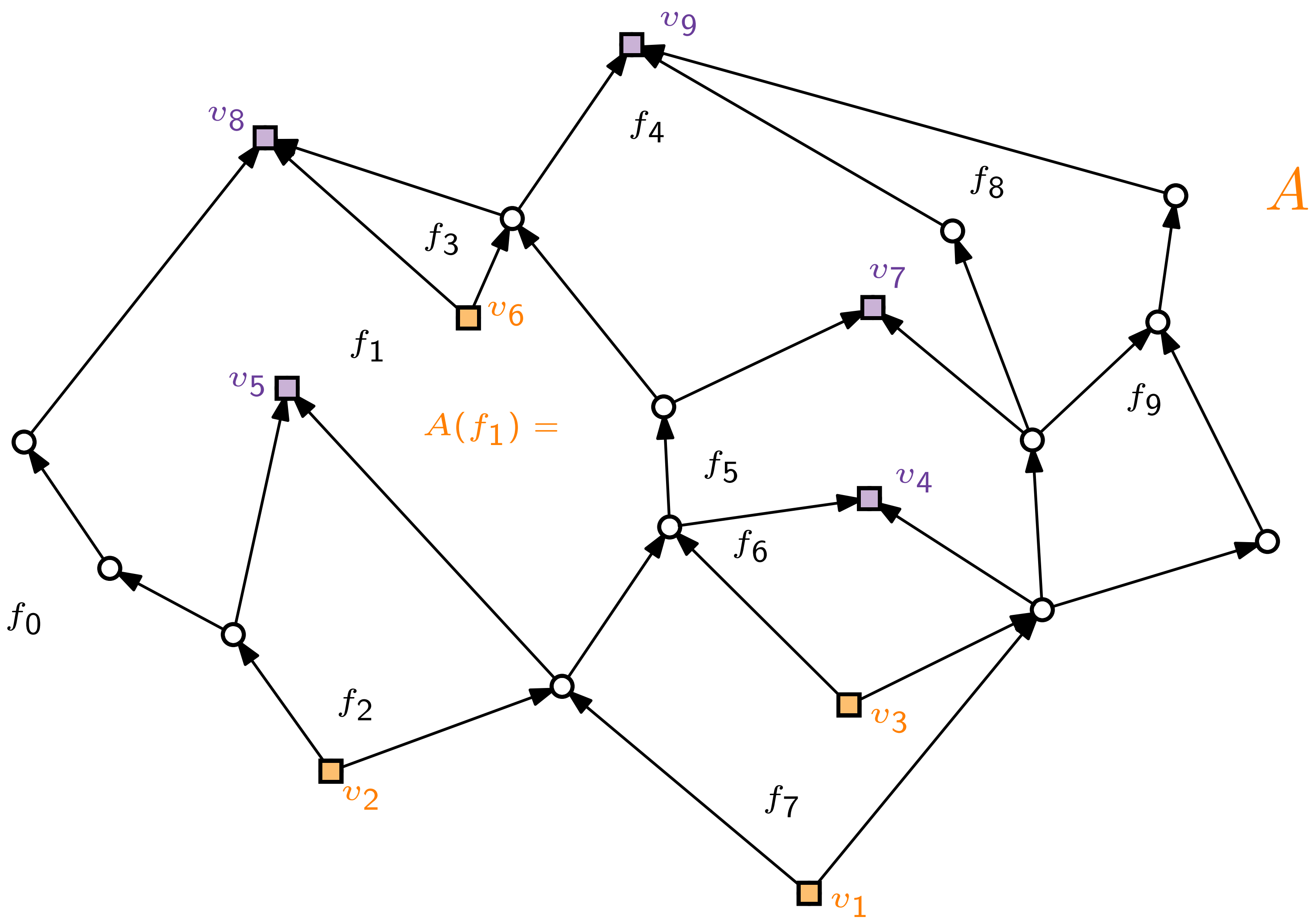
global sources

Example of Angle-to-Face Assignment



global sources & sinks

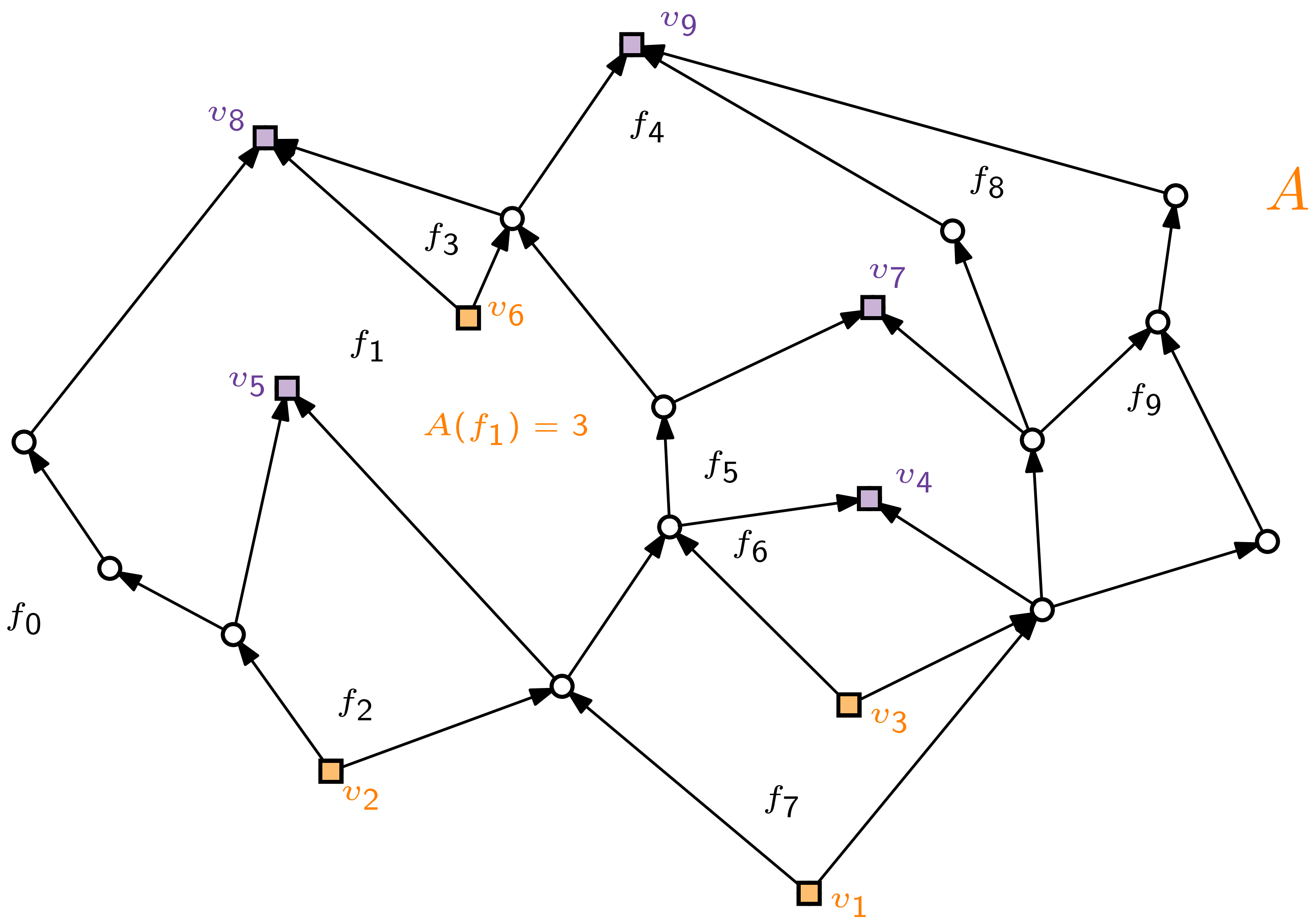
Example of Angle-to-Face Assignment



global sources & sinks

$A(f) =$ # local sources/sinks of f

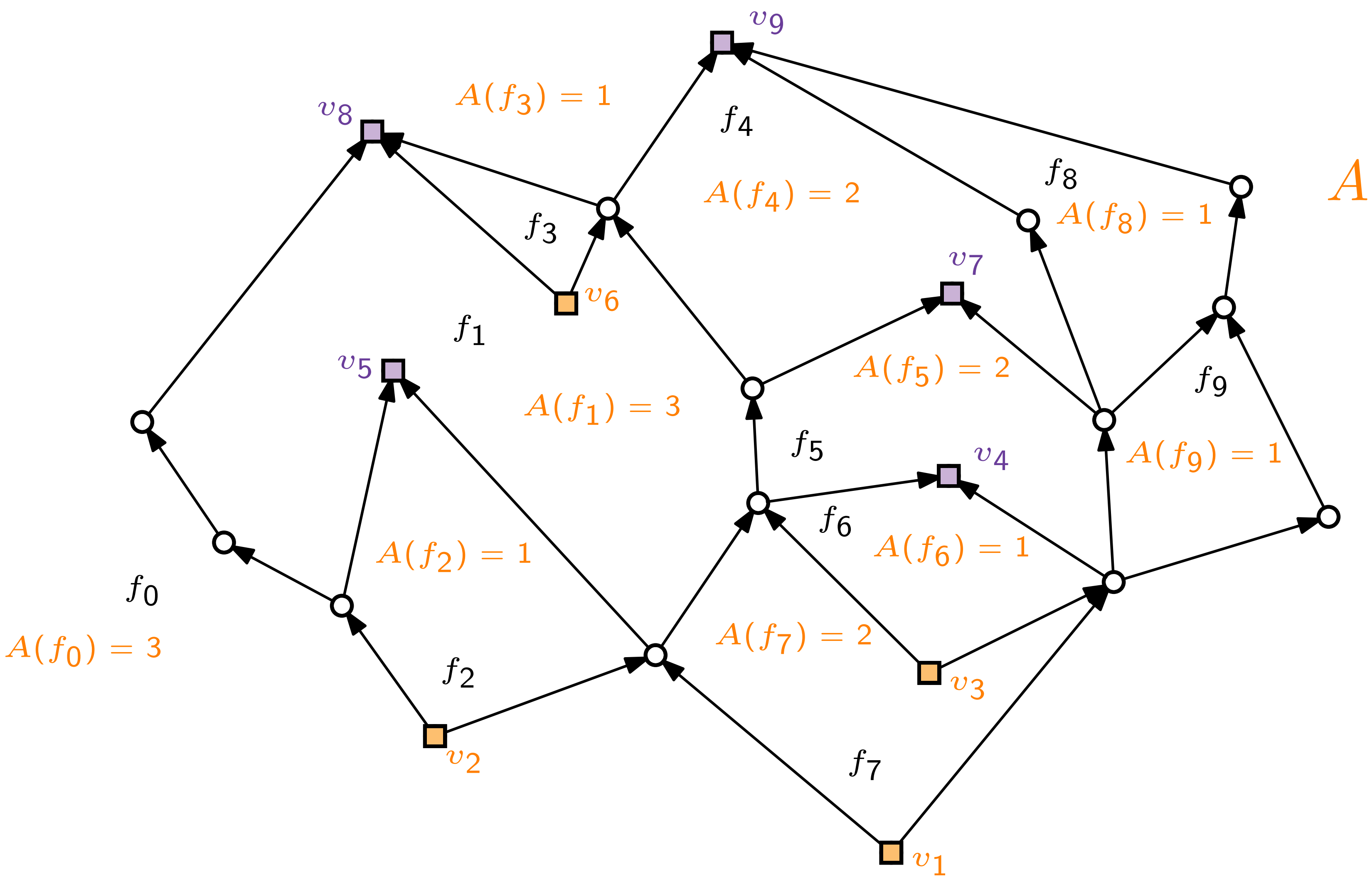
Example of Angle-to-Face Assignment



global sources & sinks

$A(f) = \#$ local sources/sinks of f

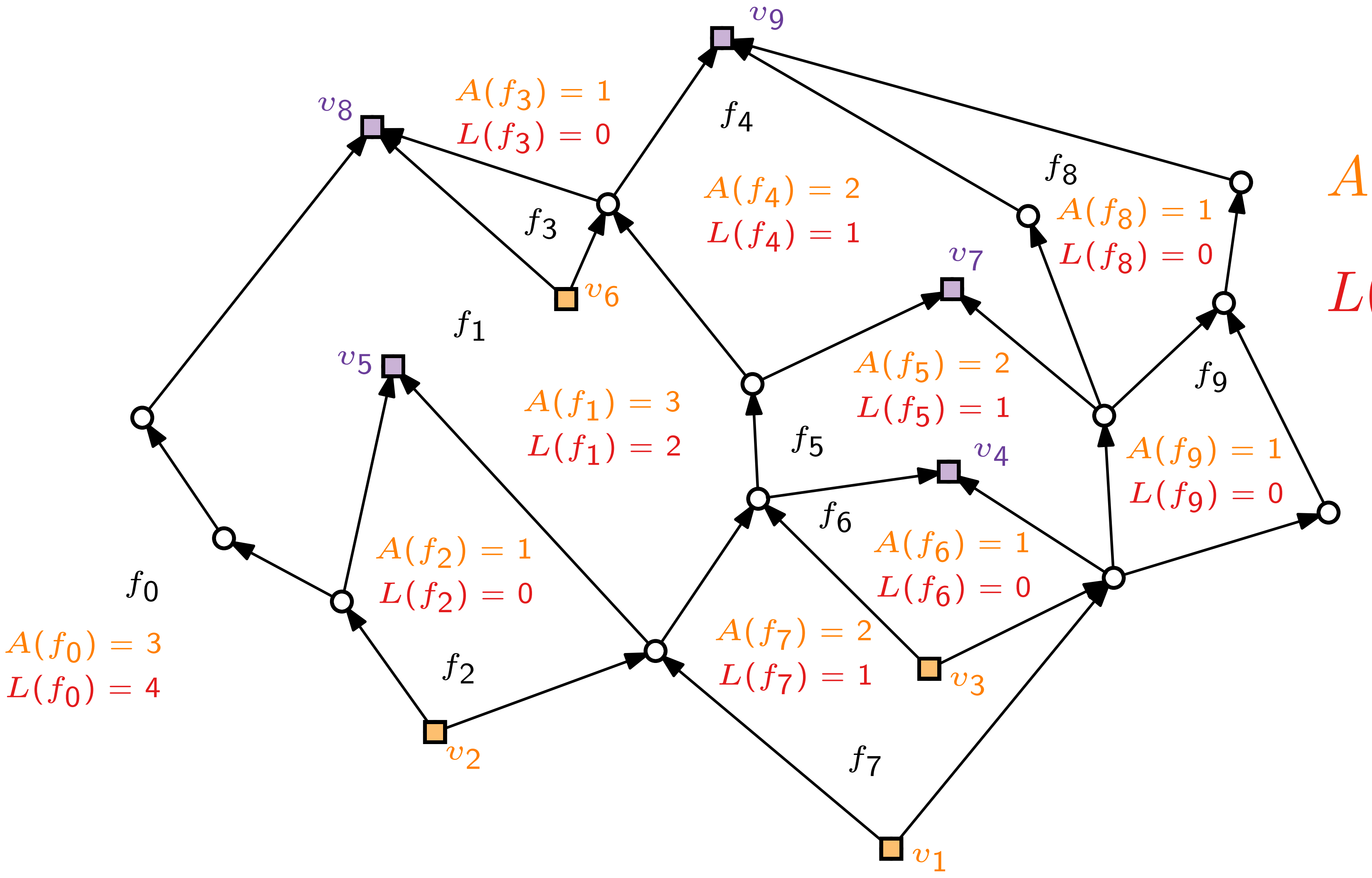
Example of Angle-to-Face Assignment



global sources & sinks

$A(f) =$ # local sources/sinks of f

Example of Angle-to-Face Assignment

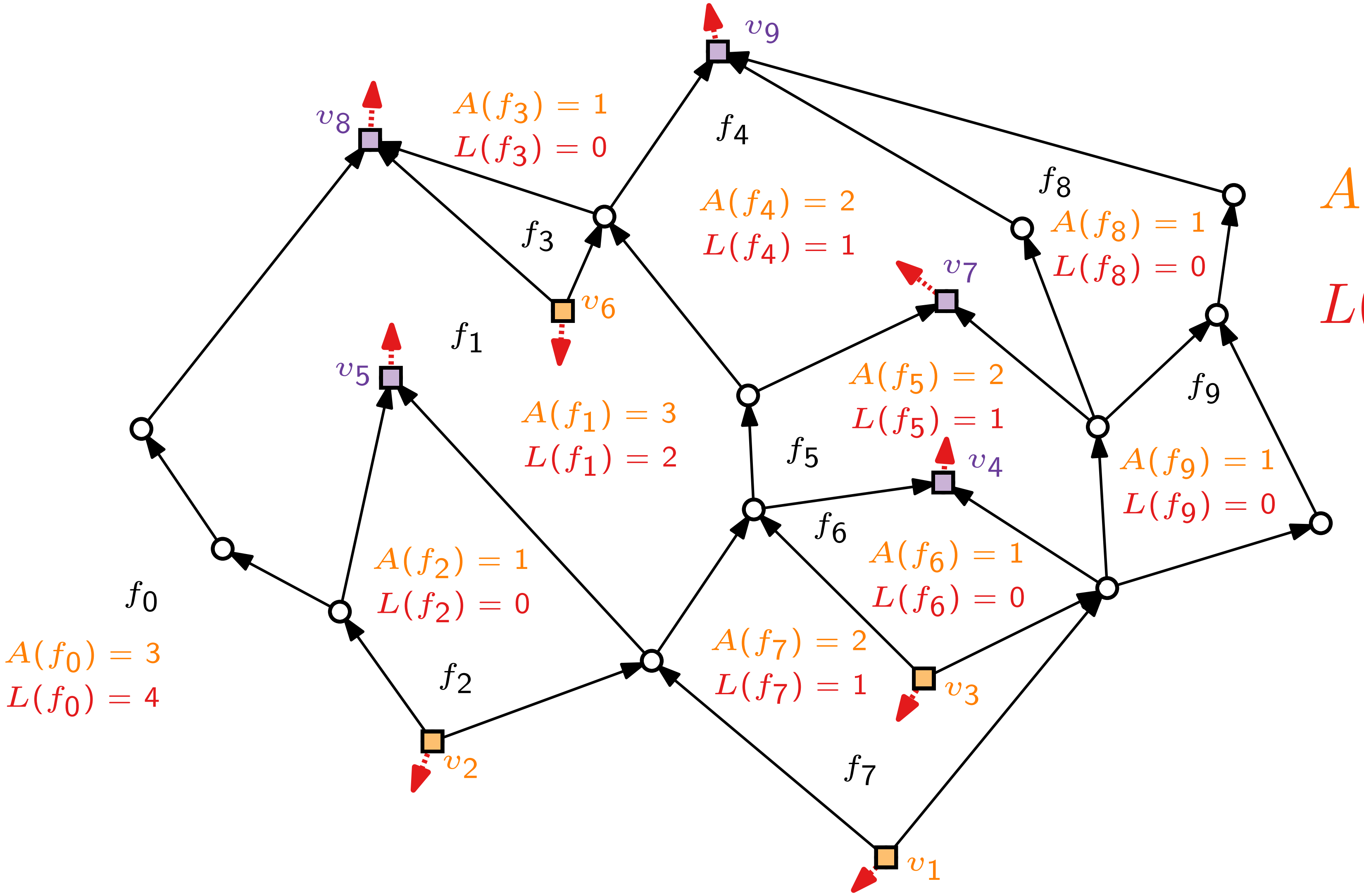


global sources & sinks

$A(f) =$ # local sources/sinks of f

$L(f) =$ # large angles of f

Example of Angle-to-Face Assignment



global sources & sinks

$A(f) =$ # local sources/sinks of f

$L(f) =$ # large angles of f

assignment
$\Phi \colon S \cup T \to F$

Result Characterization

Theorem 3.
Let G be an acyclic plane digraph with embedding given by F and f_0.

Result Characterization

Theorem 3.
Let G be an acyclic plane digraph with embedding given by F and f_0.

Then G is upward planar (respecting F and f_0)
$\Leftrightarrow$ G is bimodal and there exists a consistent assignment Φ.

Result Characterization

Theorem 3.
Let G be an acyclic plane digraph with embedding given by F and f_0.

Then G is upward planar (respecting F and f_0)
$\Leftrightarrow$ G is bimodal and there exists a consistent assignment Φ.

Proof.
$\Rightarrow$: As constructed before.

Result Characterization

Theorem 3.
Let G be an acyclic plane digraph with embedding given by F and f_0.

Then G is upward planar (respecting F and f_0)
$\Leftrightarrow$ G is bimodal and there exists a consistent assignment Φ.

Proof.
$\Rightarrow$: As constructed before.

$\Leftarrow$: Idea:

- Construct planar st-digraph that is a supergraph of G.

Result Characterization

Theorem 3.
Let G be an acyclic plane digraph with embedding given by F and f_0.

Then G is upward planar (respecting F and f_0)
$\Leftrightarrow$ G is bimodal and there exists a consistent assignment Φ.

Proof.
$\Rightarrow$: As constructed before.

$\Leftarrow$: Idea:

- Construct planar st-digraph that is a supergraph of G.
- Apply equivalence from Theorem 1.

G is upward planar. $\Leftrightarrow$ G is a spanning subgraph of a planar st-digraph.

Refinement Algorithm: $\Phi, F, f_0 \to$ st-digraph

Refinement Algorithm: $\Phi, F, f_0 \rightarrow$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

Refinement Algorithm: $\Phi, F, f_0 \rightarrow$ st-digraph

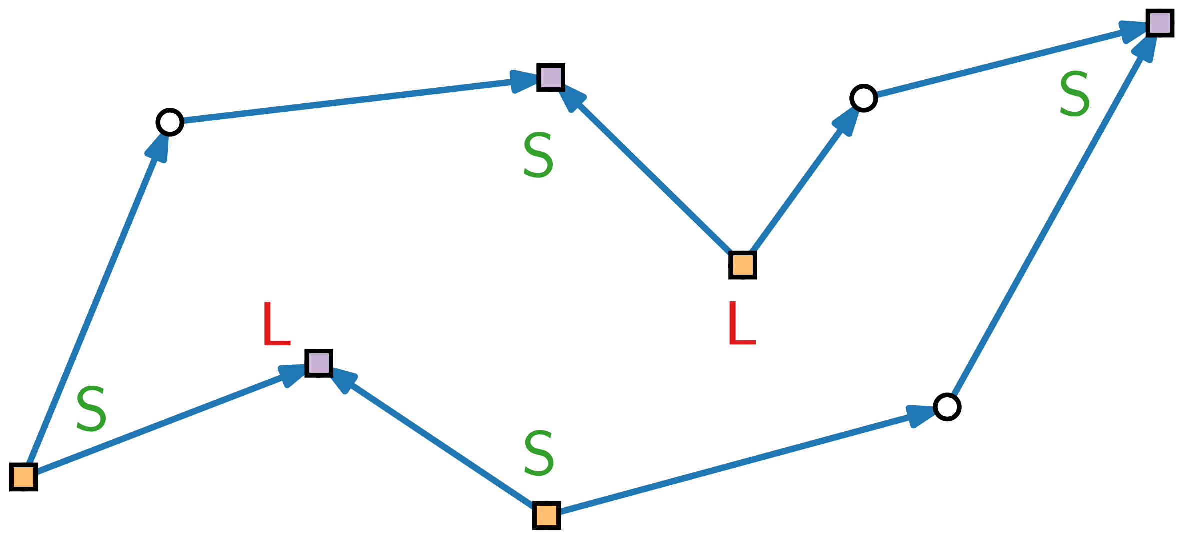
Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

- Goal: Add edges to break large angles (sources and sinks).

Refinement Algorithm: $\Phi, F, f_0 \rightarrow$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

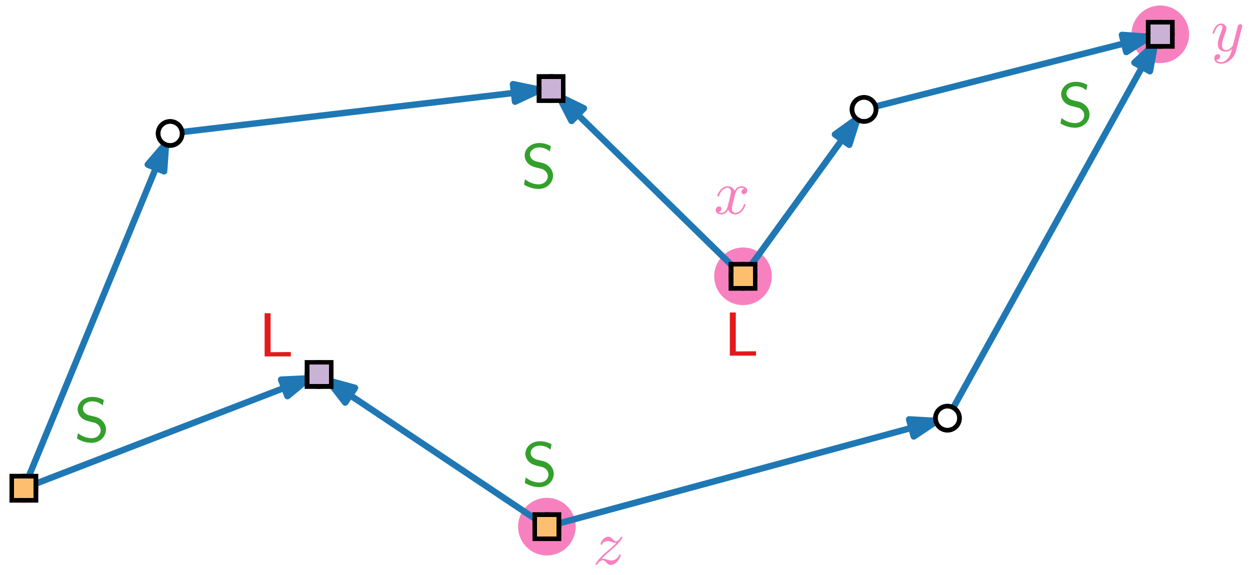
- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:



Refinement Algorithm: $\Phi, F, f_0 \to$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

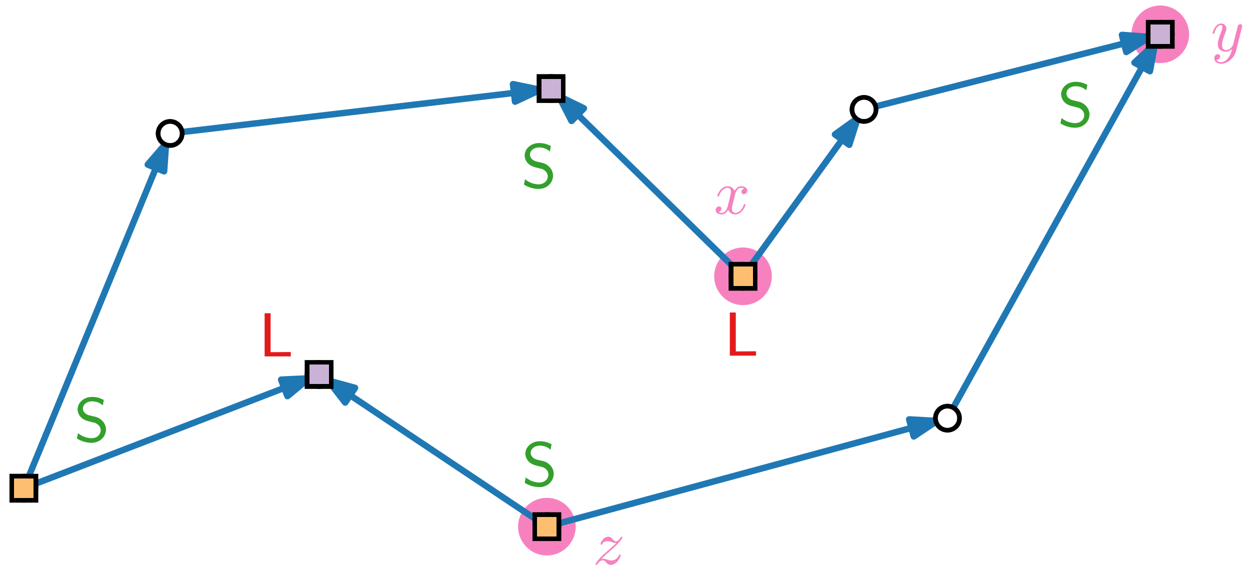
- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:



Refinement Algorithm: $\Phi, F, f_0 \to$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

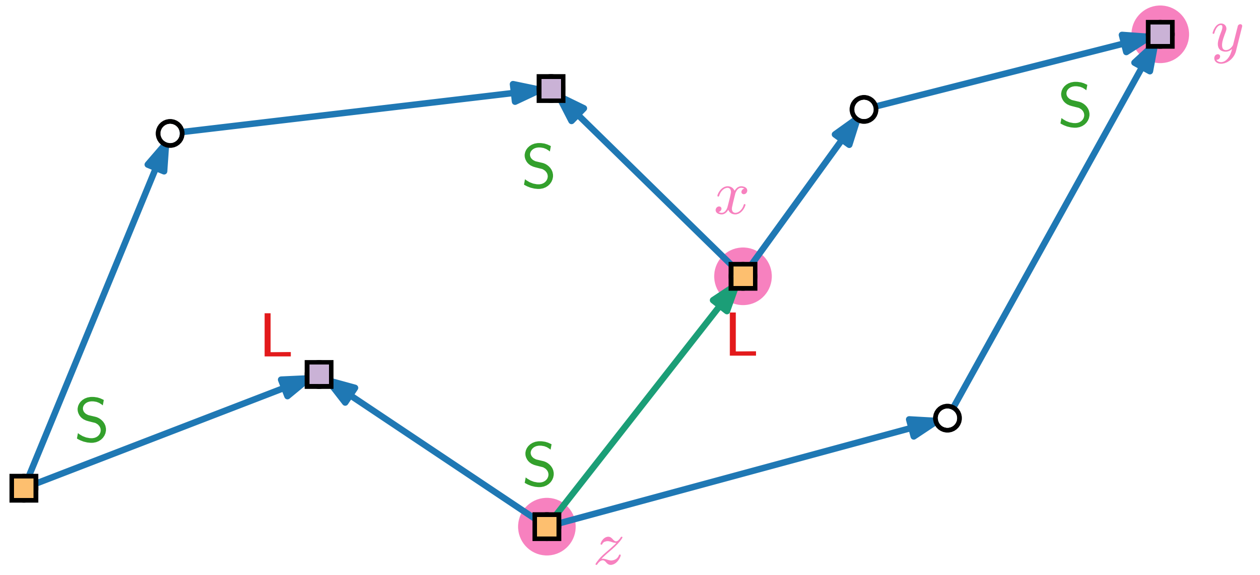
- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:
- x source $\Rightarrow$ insert edge (z, x)



Refinement Algorithm: $\Phi, F, f_0 \rightarrow$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

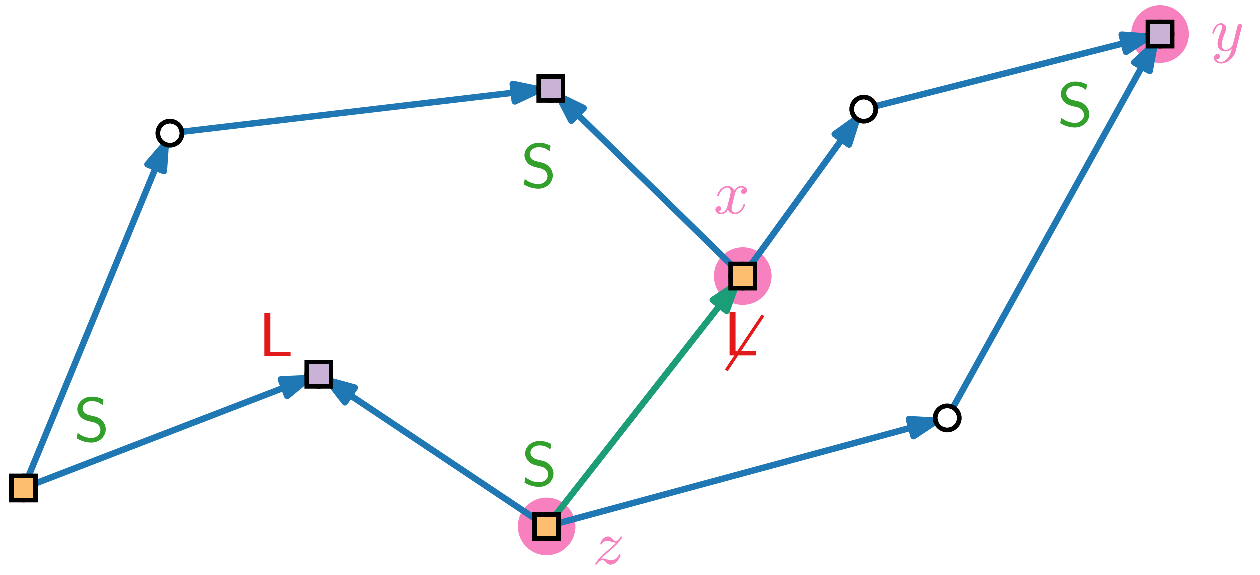
- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:
- x source $\Rightarrow$ insert edge (z, x)



Refinement Algorithm: $\Phi, F, f_0 \to$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

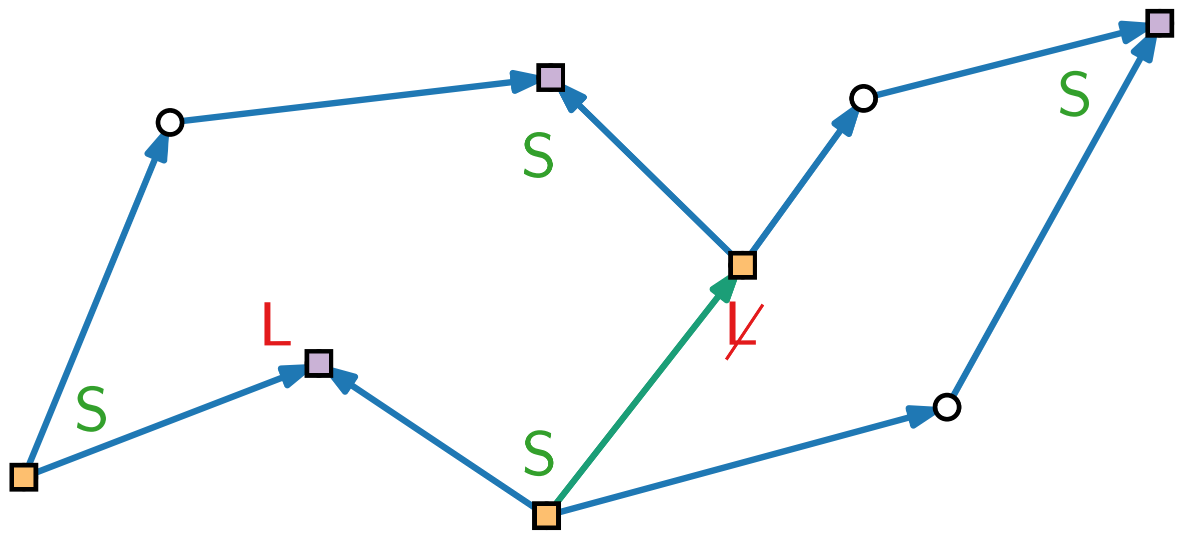
- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:
- x source $\Rightarrow$ insert edge (z, x)



Refinement Algorithm: $\Phi, F, f_0 \to$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

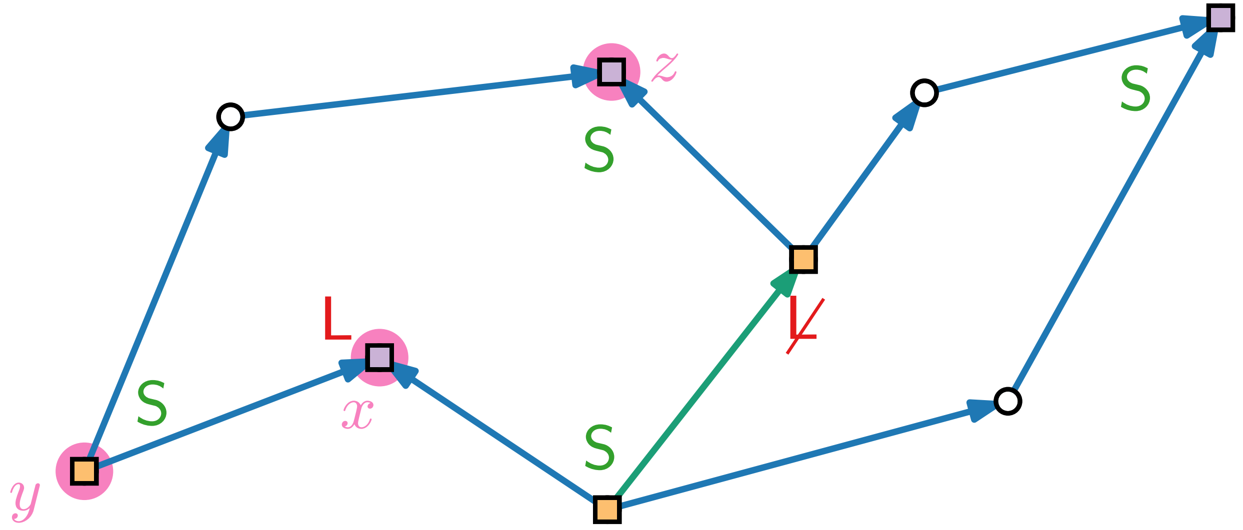
- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:
- x source $\Rightarrow$ insert edge (z, x)



Refinement Algorithm: $\Phi, F, f_0 \to$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

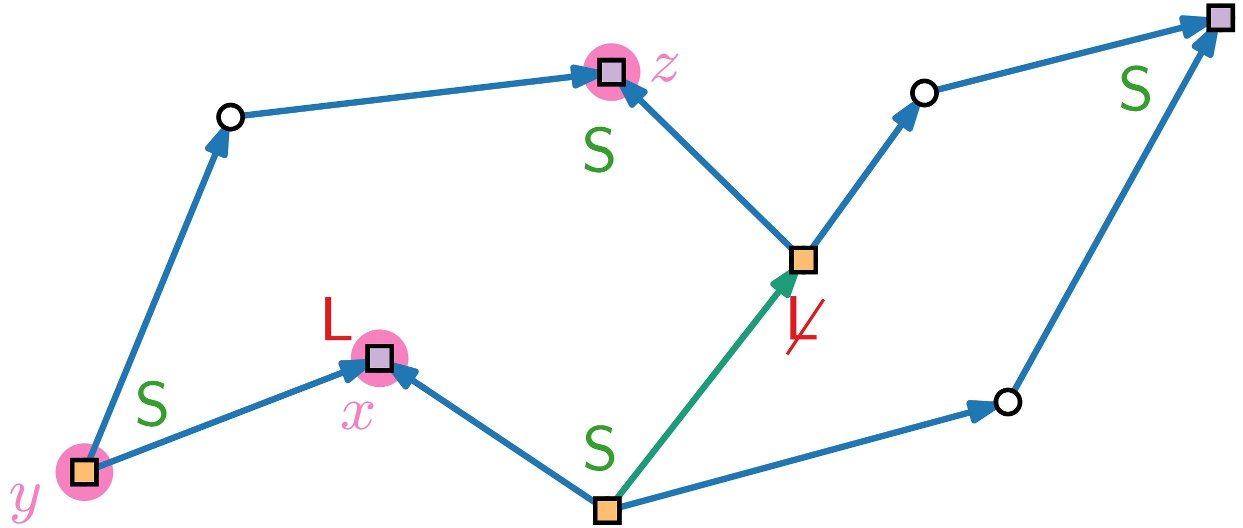
- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:
- x source $\Rightarrow$ insert edge (z, x)



Refinement Algorithm: $\Phi, F, f_0 \to$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

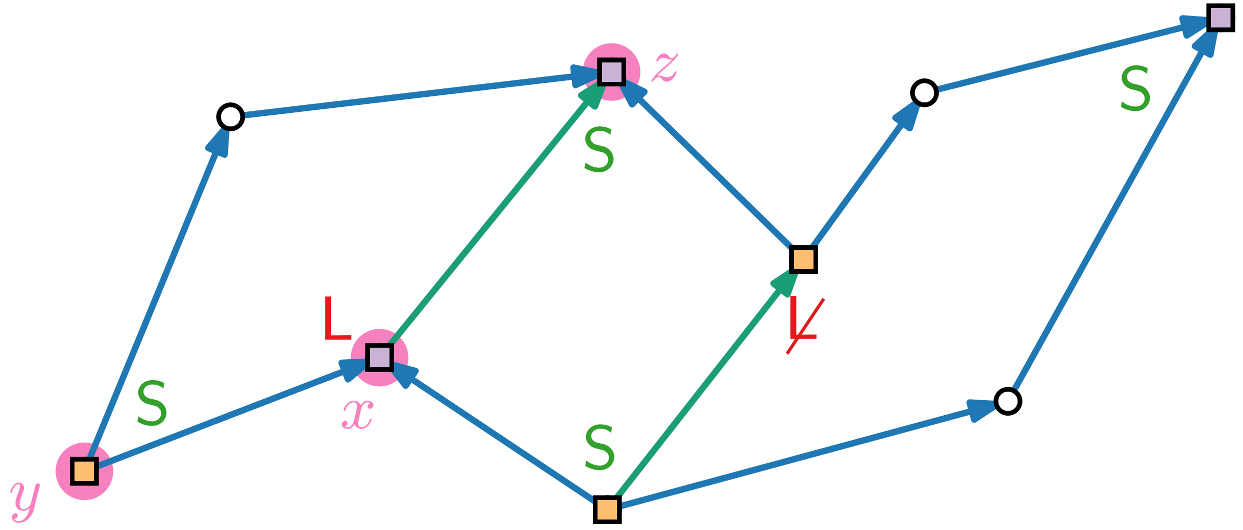
- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:
- x source $\Rightarrow$ insert edge (z, x)
- x sink $\Rightarrow$ insert edge (x, z).



Refinement Algorithm: $\Phi, F, f_0 \to$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

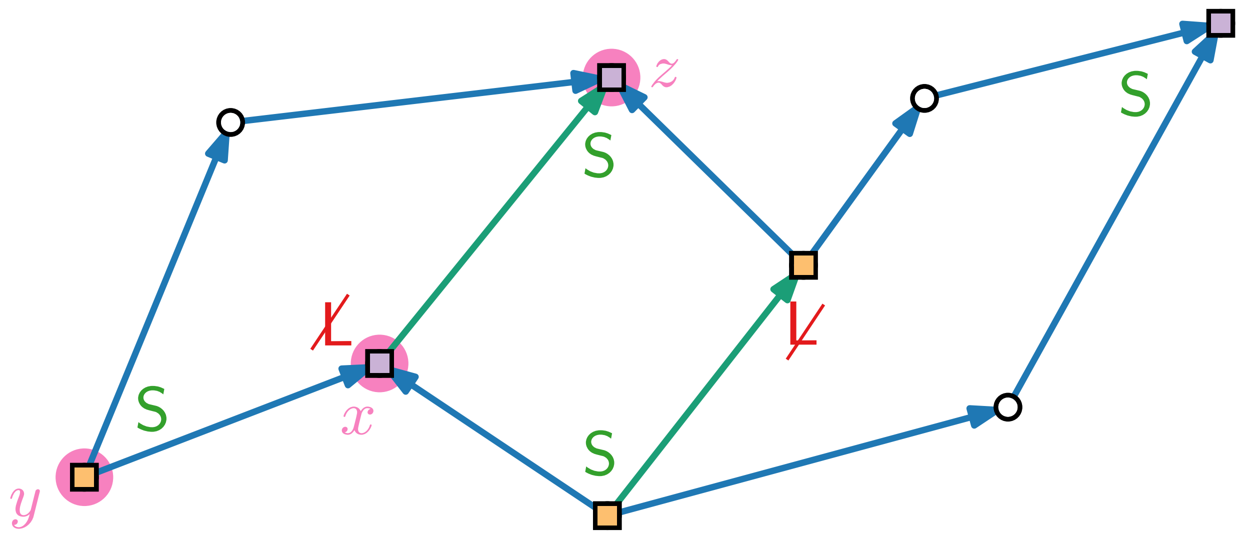
- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:
- x source $\Rightarrow$ insert edge (z, x)
- x sink $\Rightarrow$ insert edge (x, z).



Refinement Algorithm: $\Phi, F, f_0 \to$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

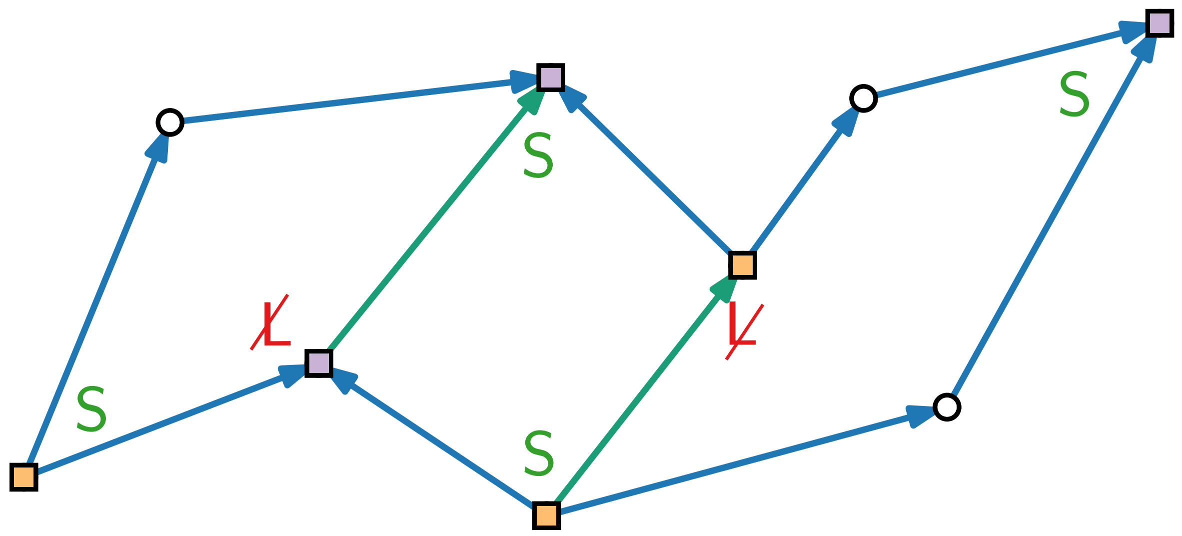
- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:
- x source $\Rightarrow$ insert edge (z, x)
- x sink $\Rightarrow$ insert edge (x, z).



Refinement Algorithm: $\Phi, F, f_0 \to$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

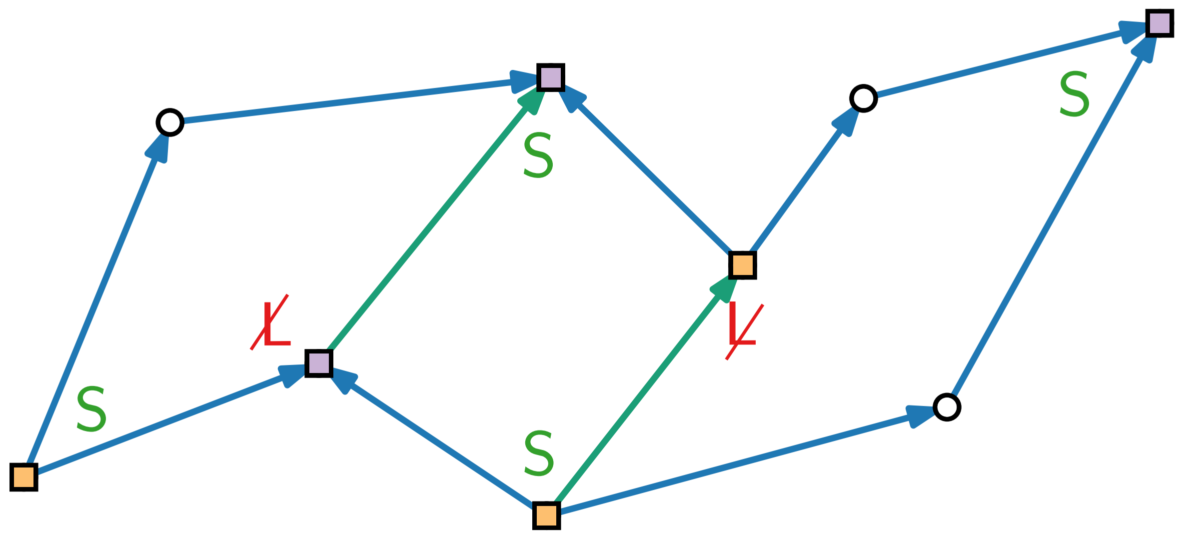
- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:
- x source $\Rightarrow$ insert edge (z, x)
- x sink $\Rightarrow$ insert edge (x, z).



Refinement Algorithm: $\Phi, F, f_0 \to$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:
- x source $\Rightarrow$ insert edge (z, x)
- x sink $\Rightarrow$ insert edge (x, z).
- Refine outer face f_0 similarly.

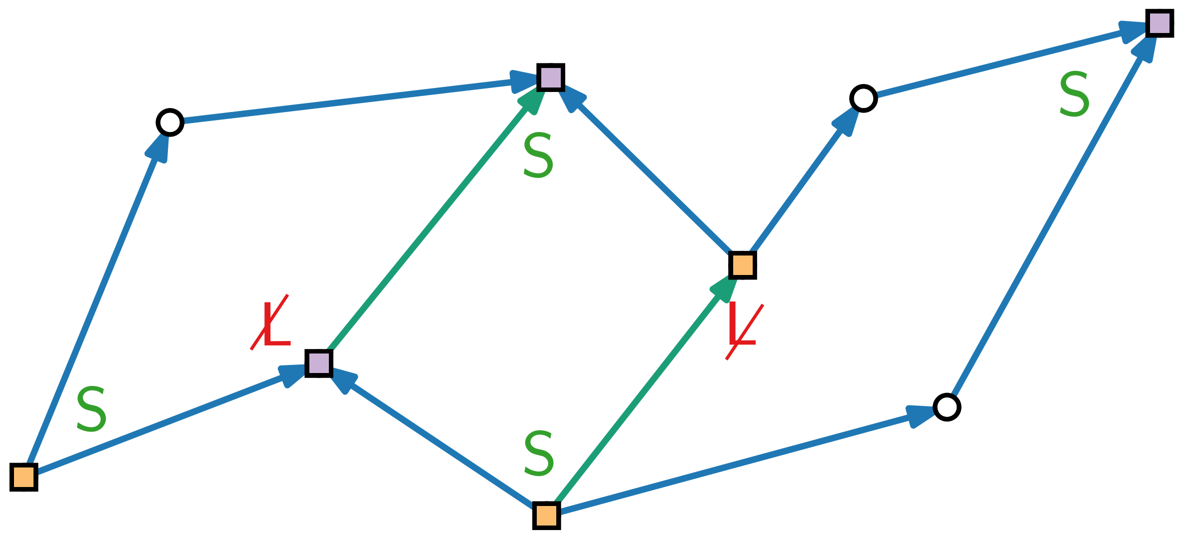


Refinement Algorithm: $\Phi, F, f_0 \rightarrow$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:
- x source $\Rightarrow$ insert edge (z, x)
- x sink $\Rightarrow$ insert edge (x, z).
- Refine outer face f_0 similarly.

  $\rightarrow$ **Exercise**

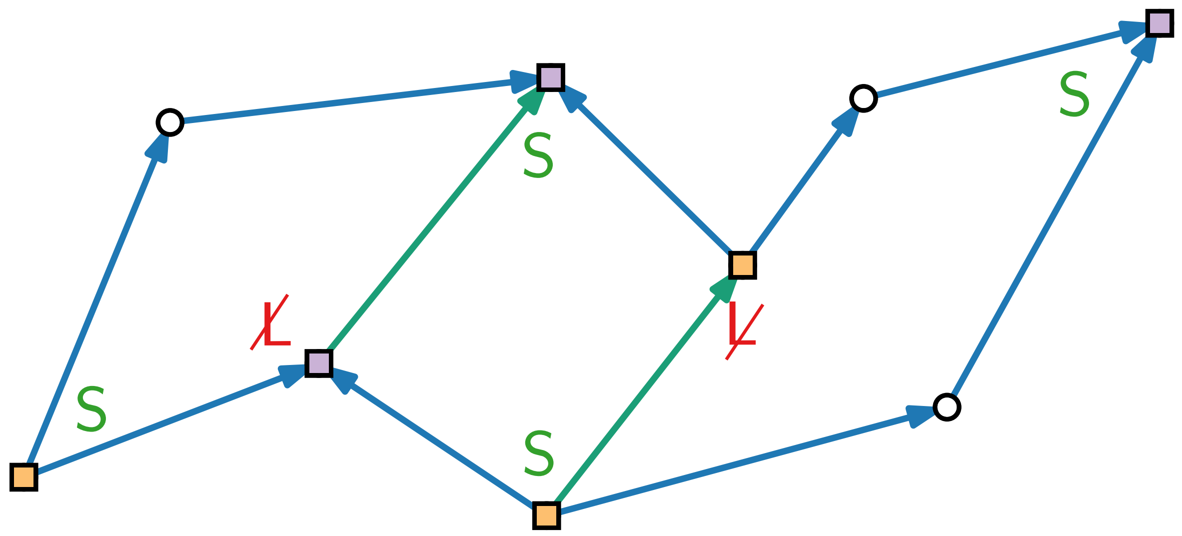


Refinement Algorithm: $\Phi, F, f_0 \to$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:
- x source $\Rightarrow$ insert edge (z, x)
- x sink $\Rightarrow$ insert edge (x, z).
- Refine outer face f_0 similarly.

  $\to$ **Exercise**



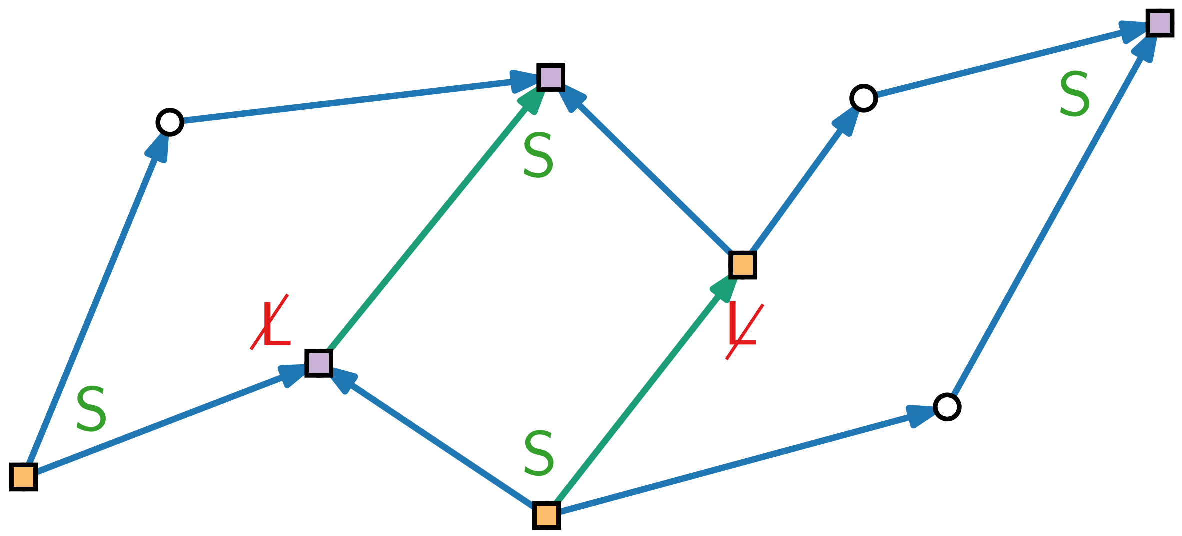
- Refine all faces. $\Rightarrow$ G is contained in a planar st-digraph.

Refinement Algorithm: $\Phi, F, f_0 \to$ st-digraph

Let f be a face.
Consider the clockwise angle sequence σ_f of L / S on local sources and sinks of f.

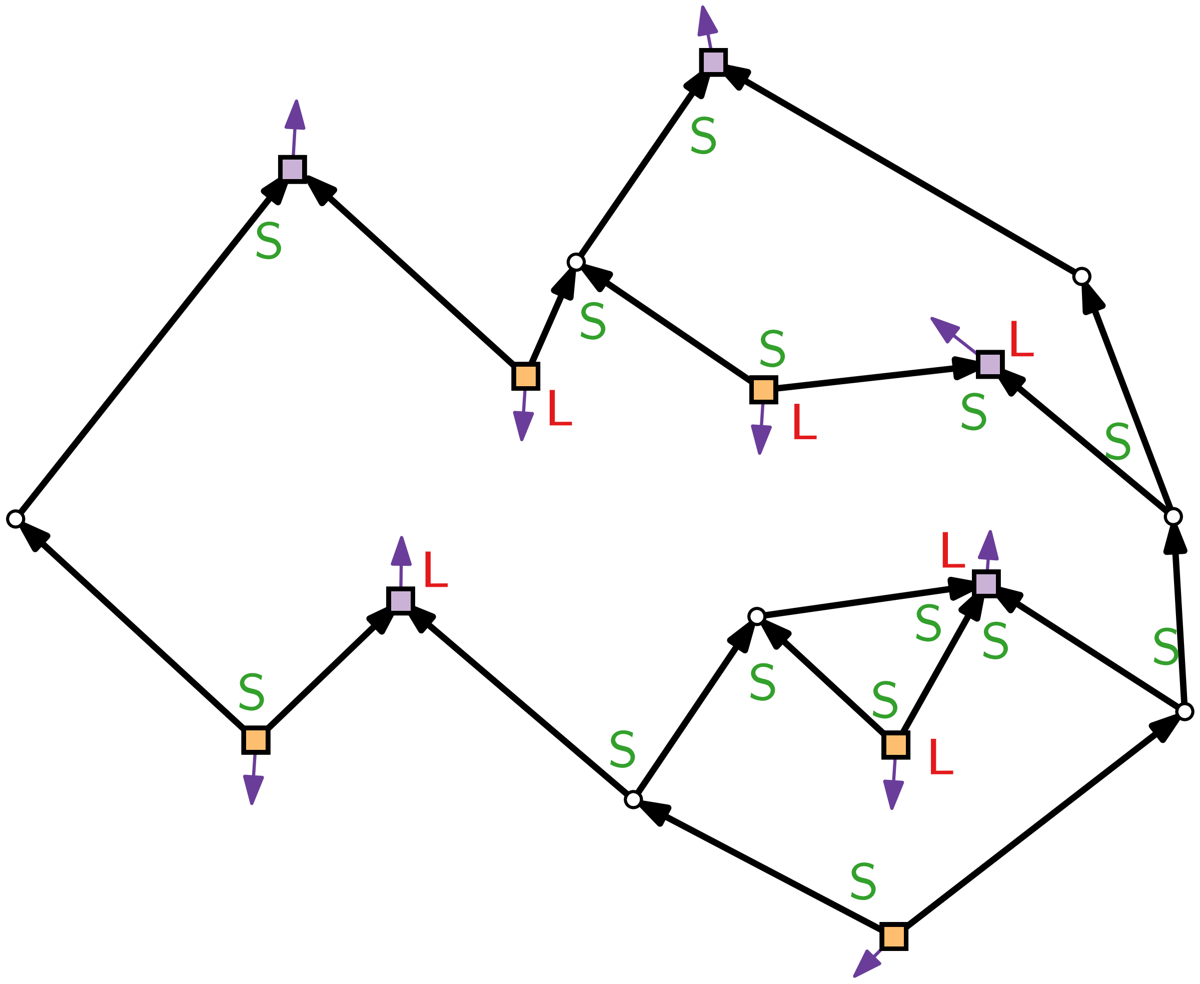
- Goal: Add edges to break large angles (sources and sinks).
- For $f \neq f_0$ with $|\sigma_f| \geq 2$ containing $\langle$L, S, S$\rangle$ at vertices x, y, z:
- x source $\Rightarrow$ insert edge (z, x)
- x sink $\Rightarrow$ insert edge (x, z).
- Refine outer face f_0 similarly.

  $\to$ **Exercise**

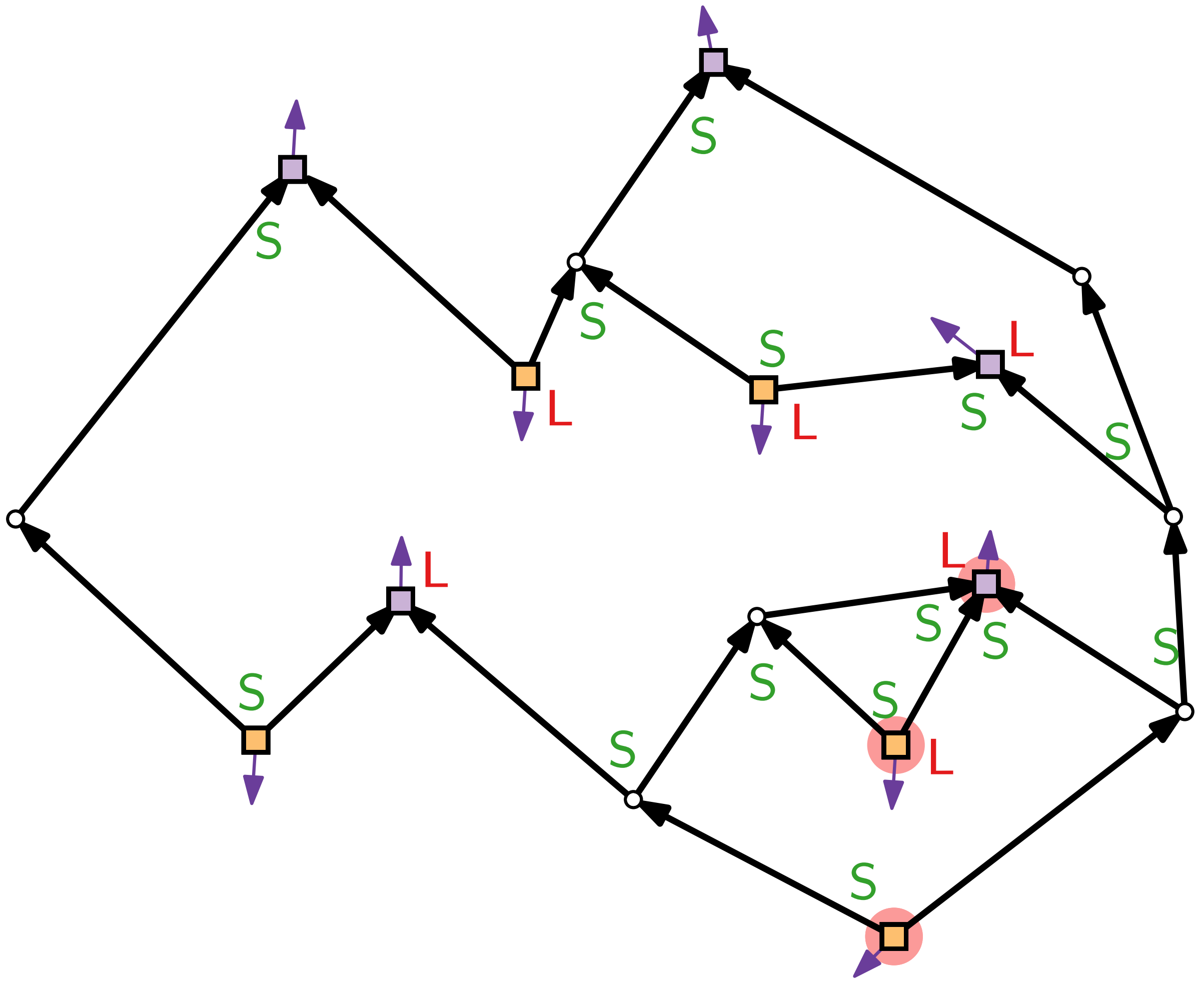


- Refine all faces. $\Rightarrow$ G is contained in a planar st-digraph.
- Planarity, acyclicity, bimodality are invariants under construction.

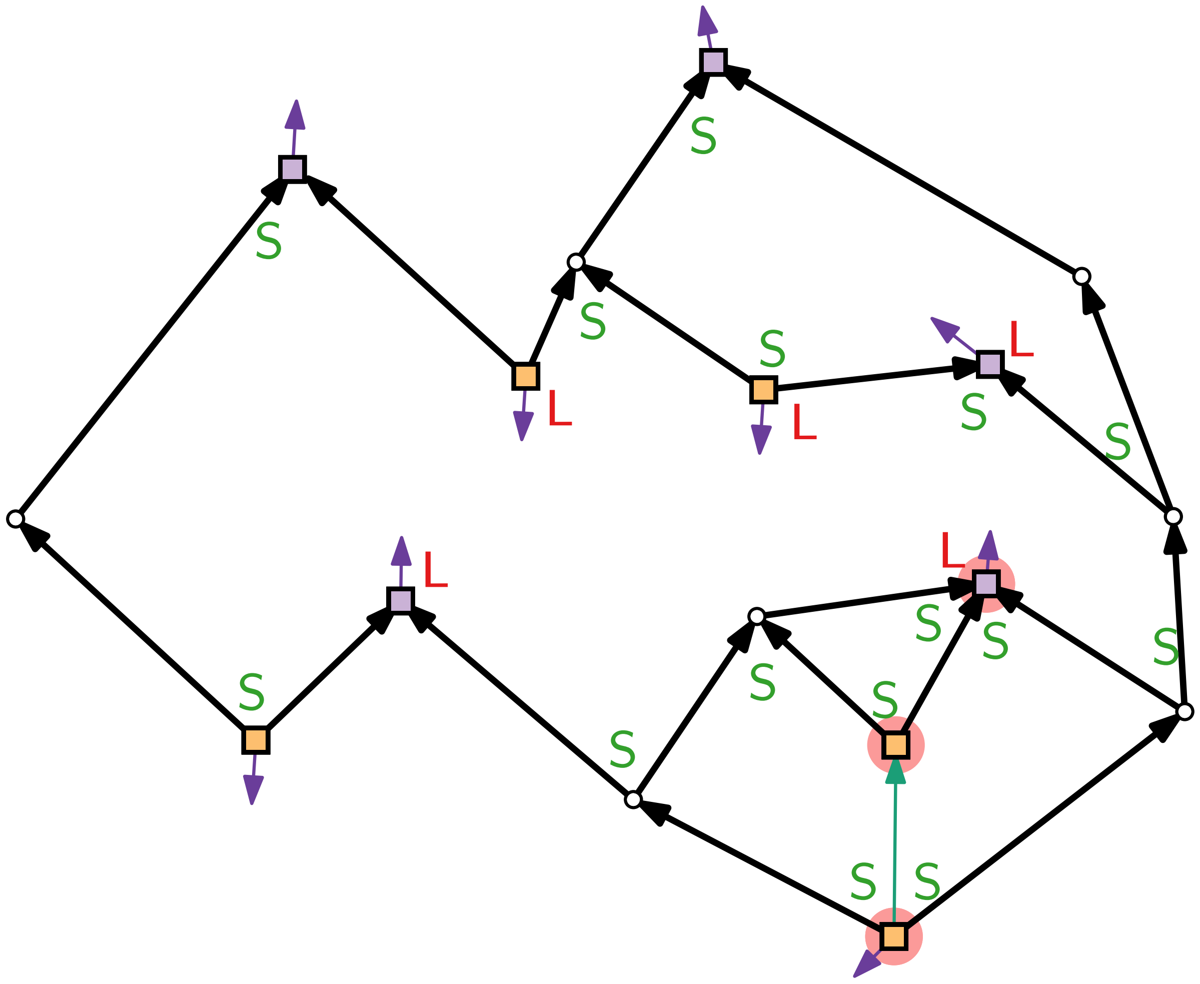
Refinement Example



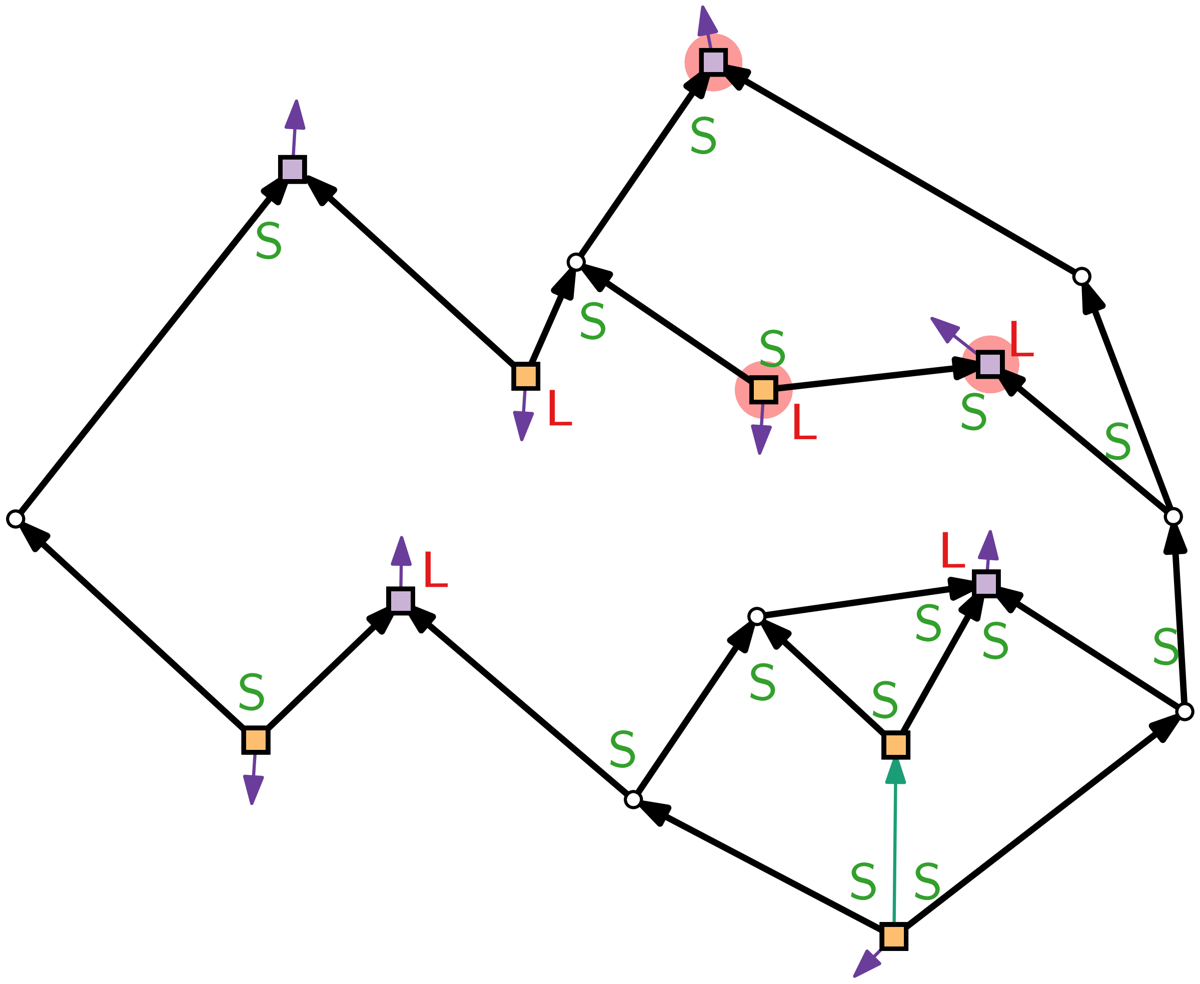
Refinement Example



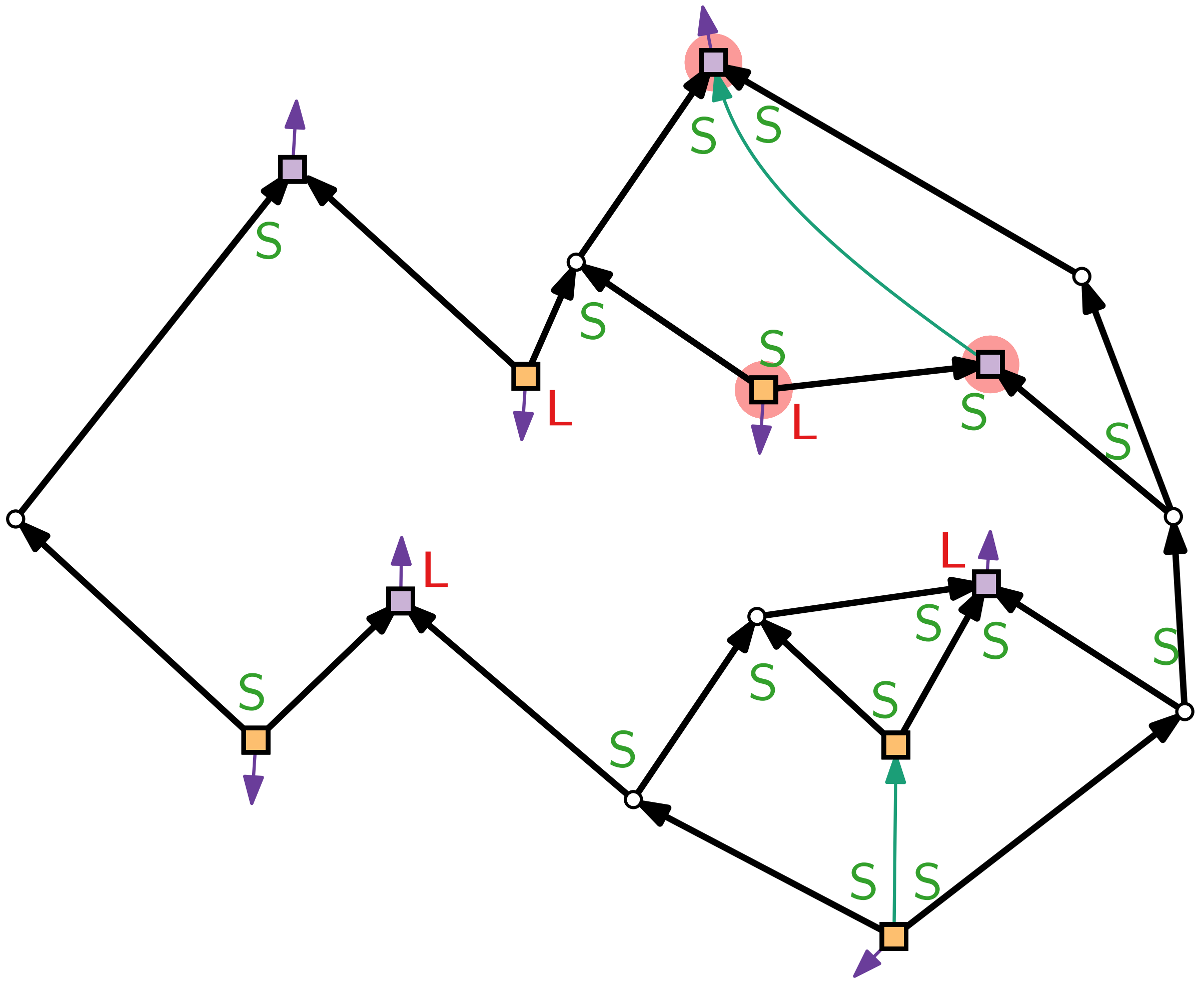
Refinement Example



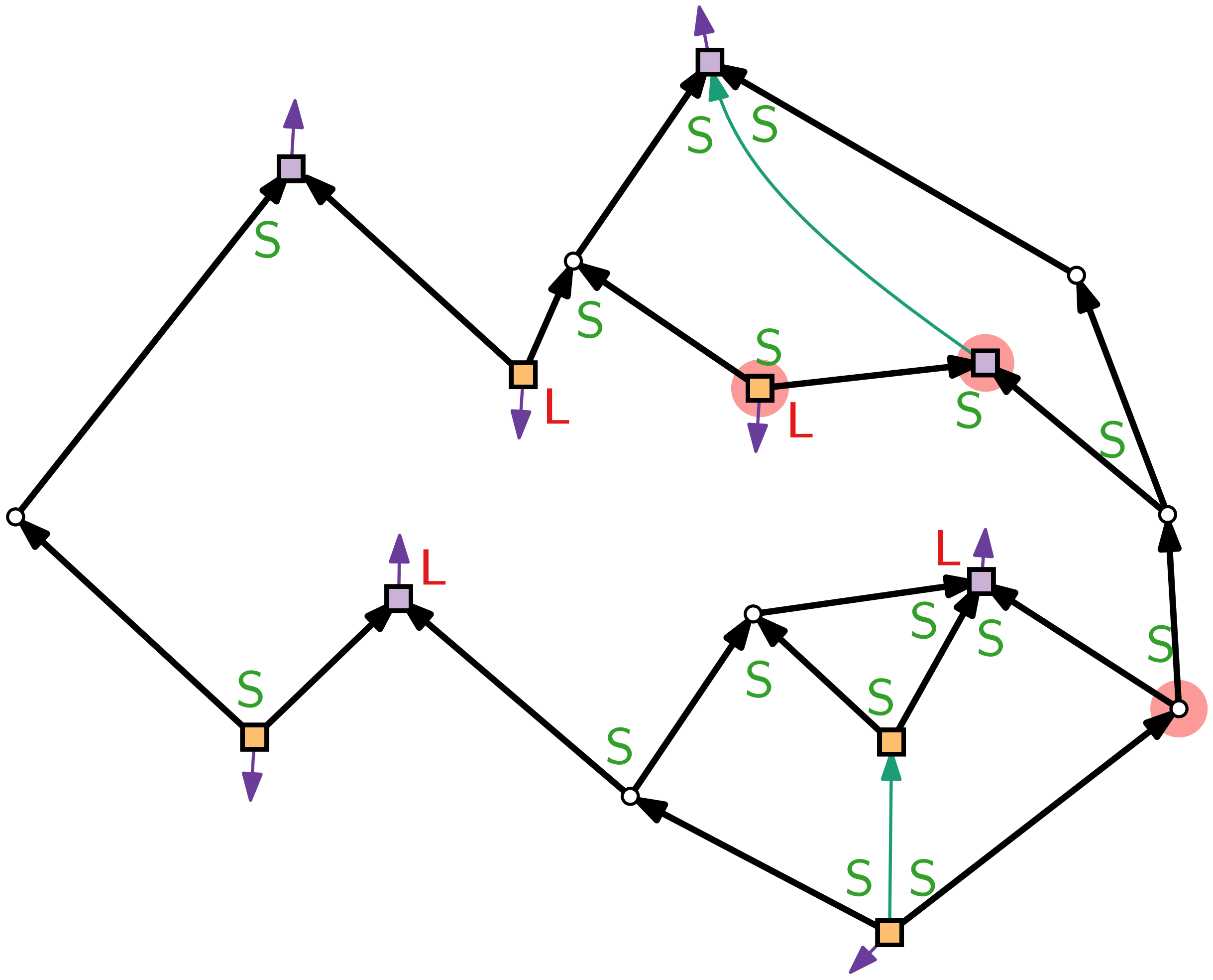
Refinement Example



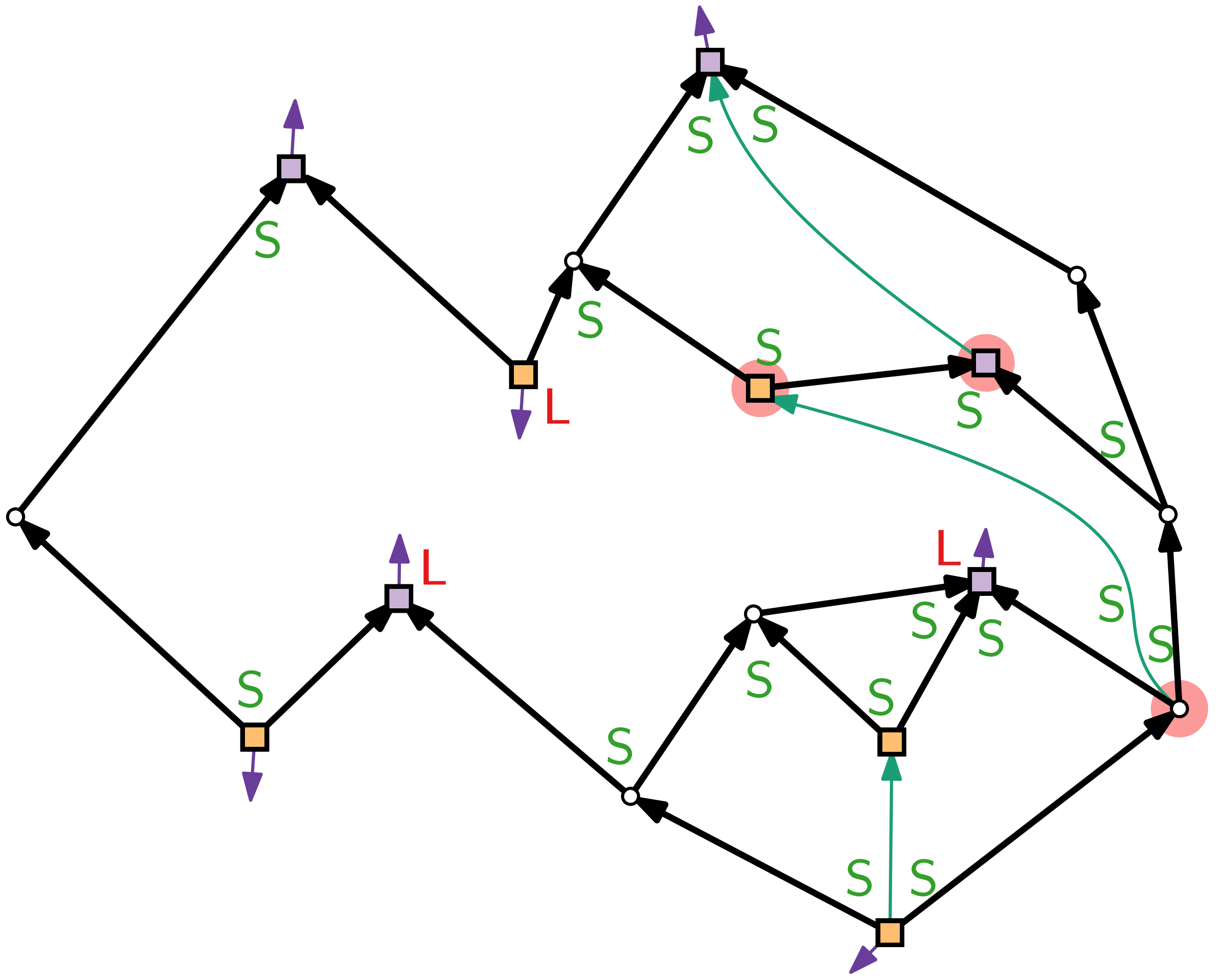
Refinement Example



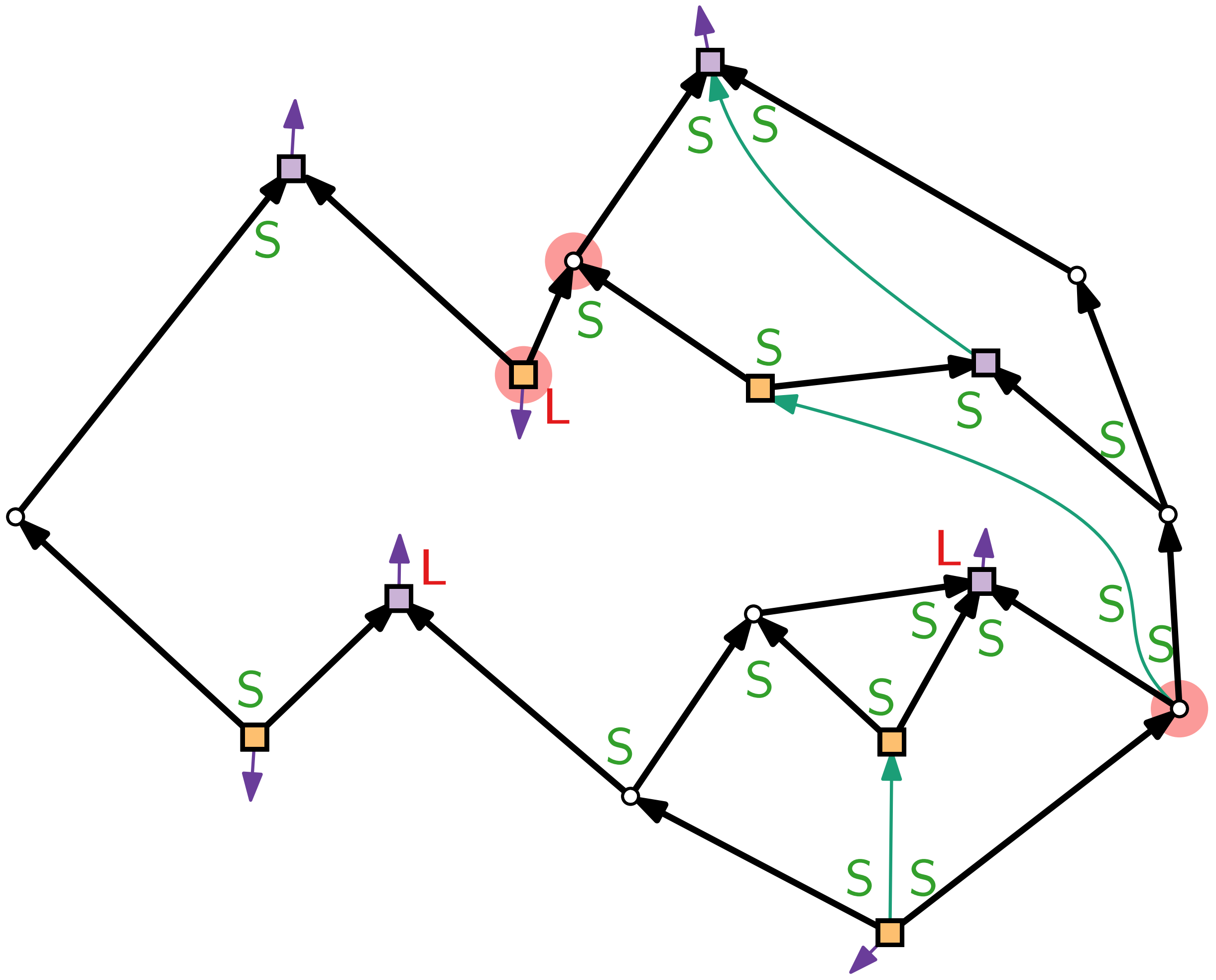
Refinement Example



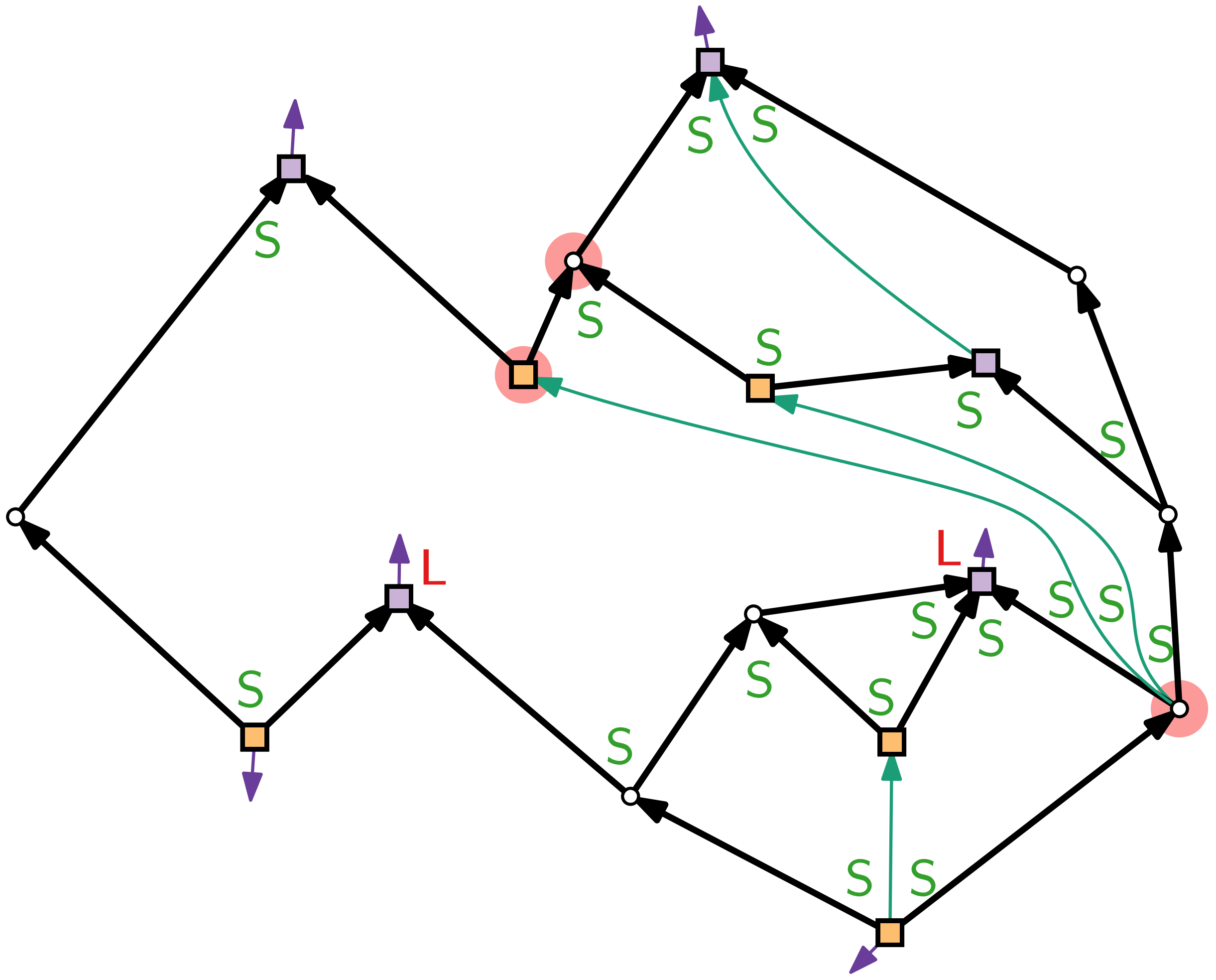
Refinement Example



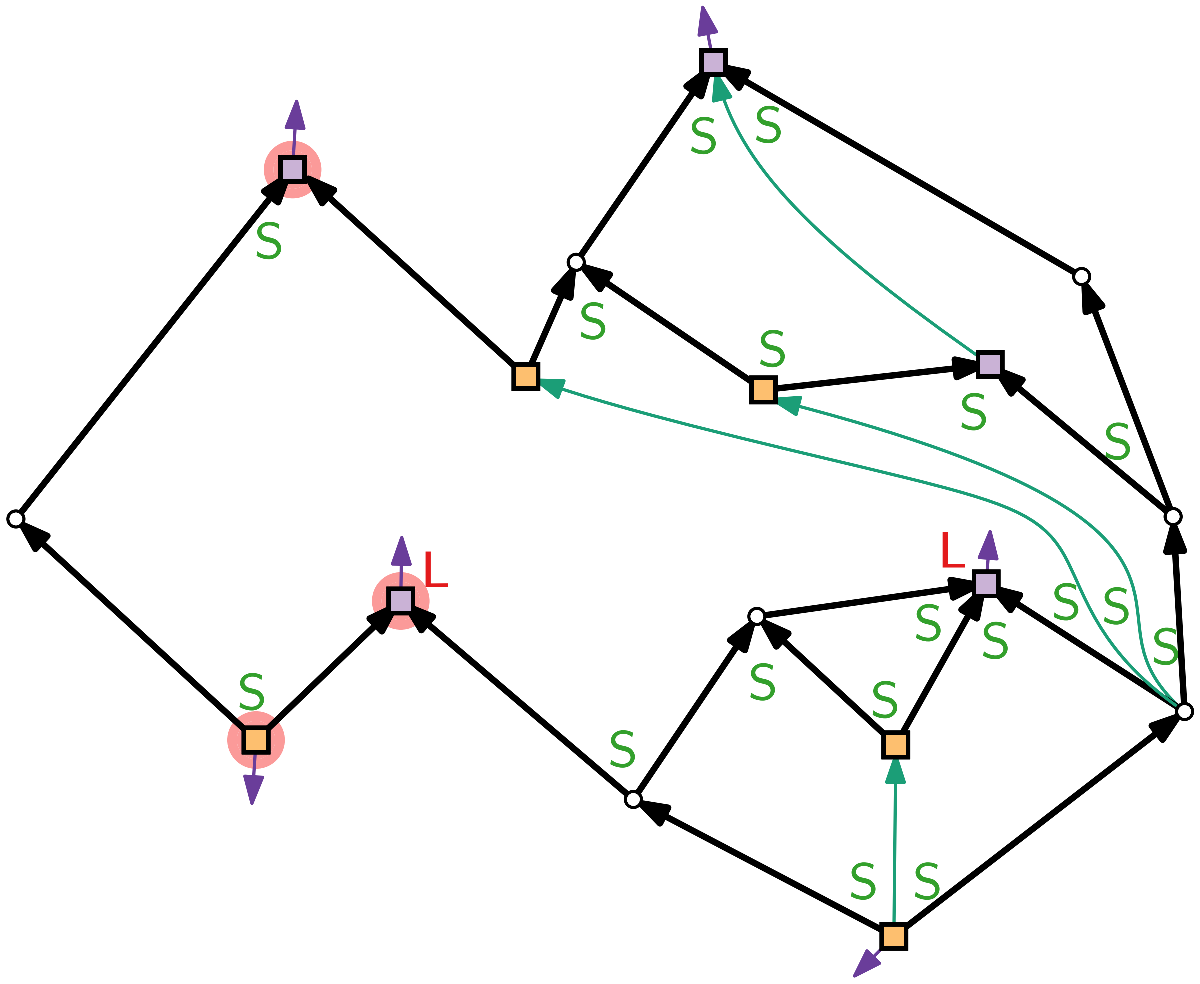
Refinement Example



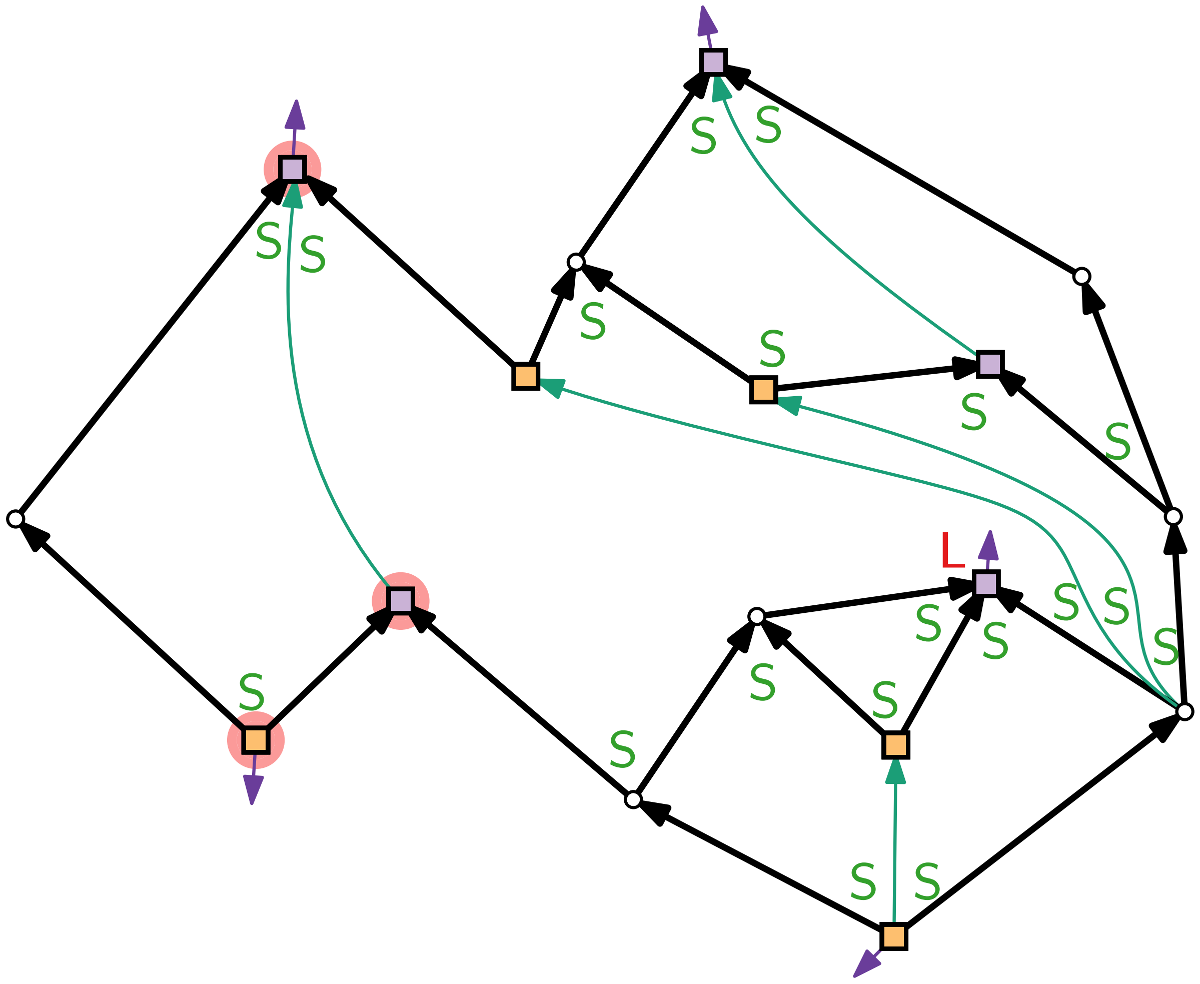
Refinement Example



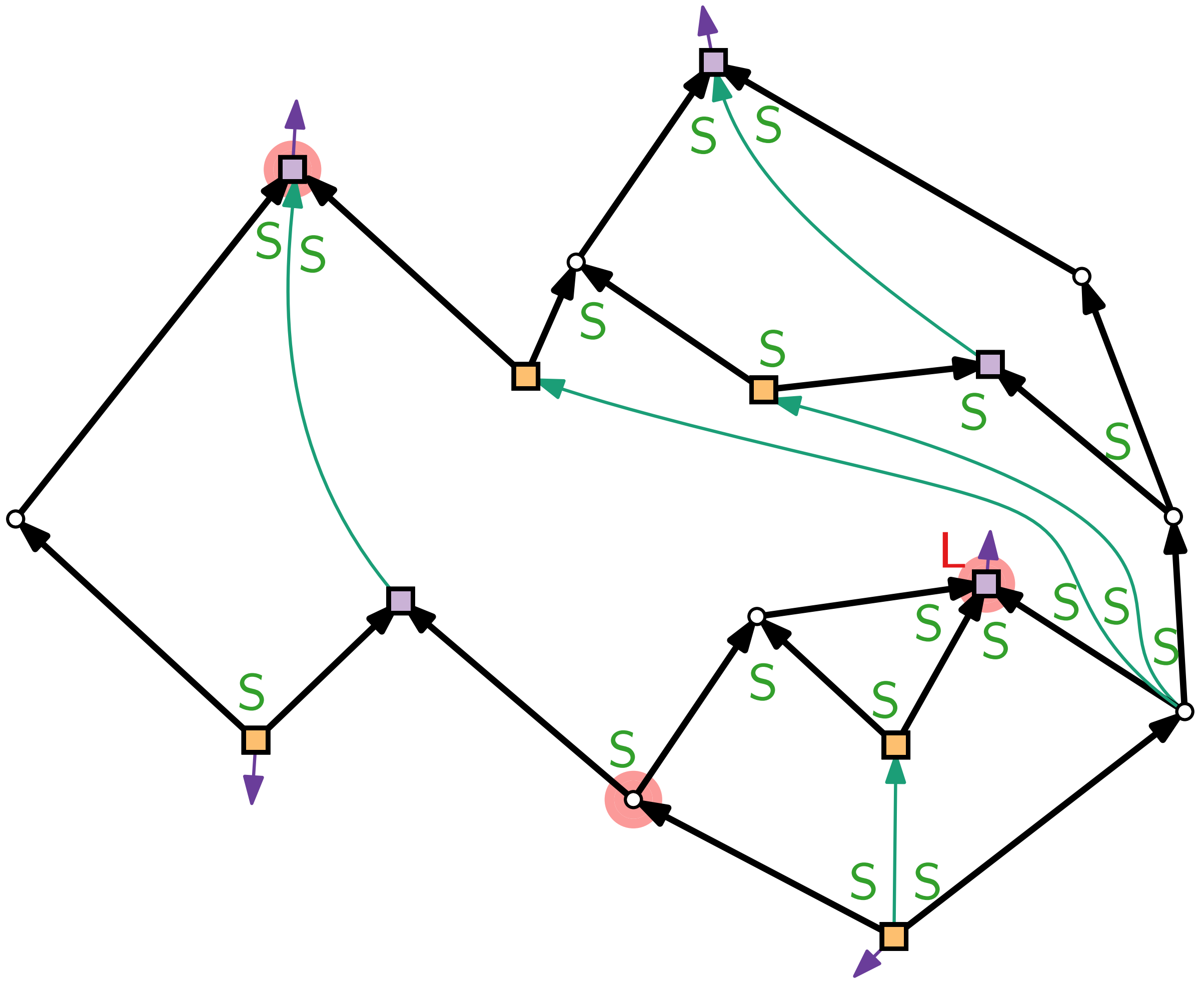
Refinement Example



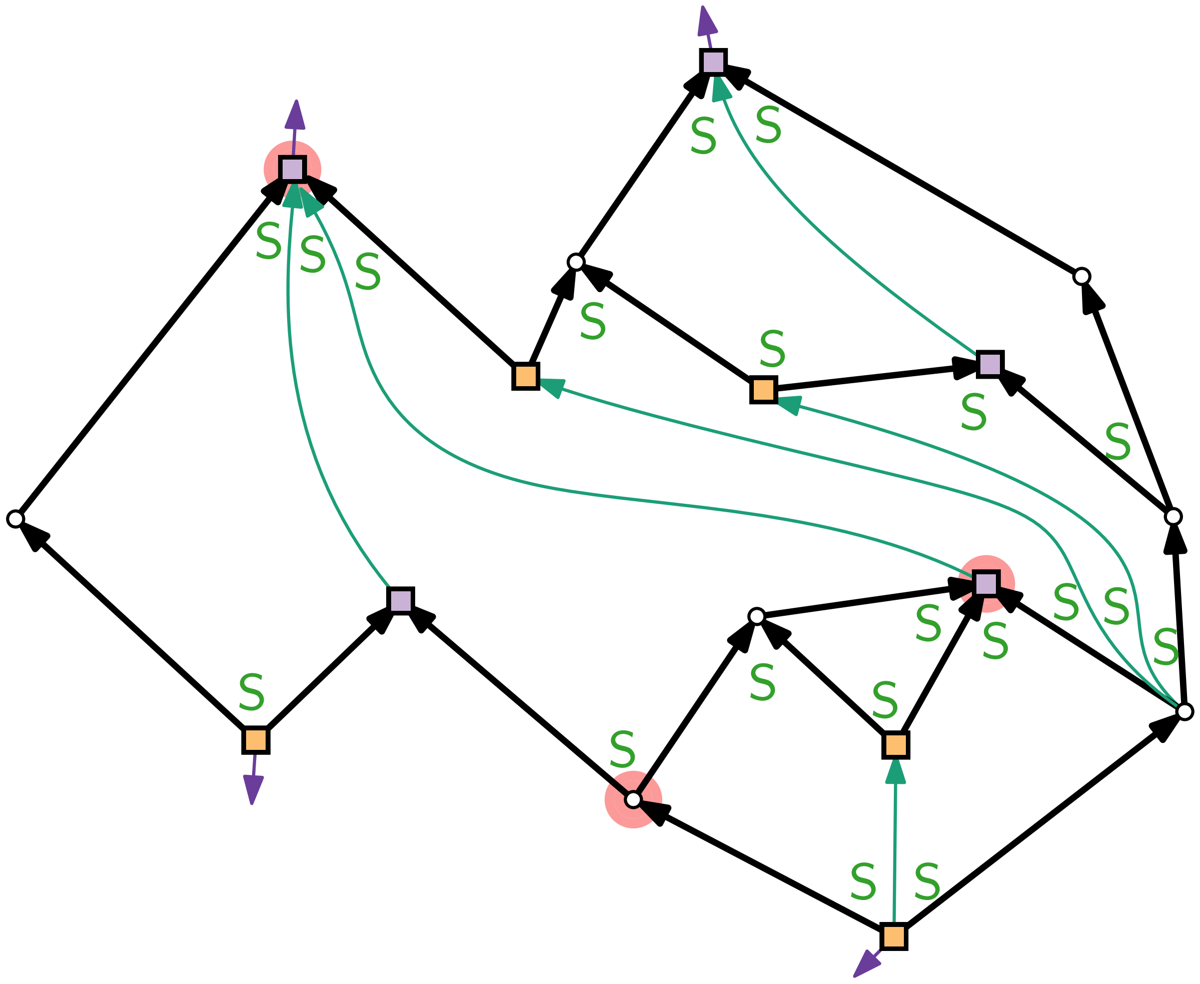
Refinement Example



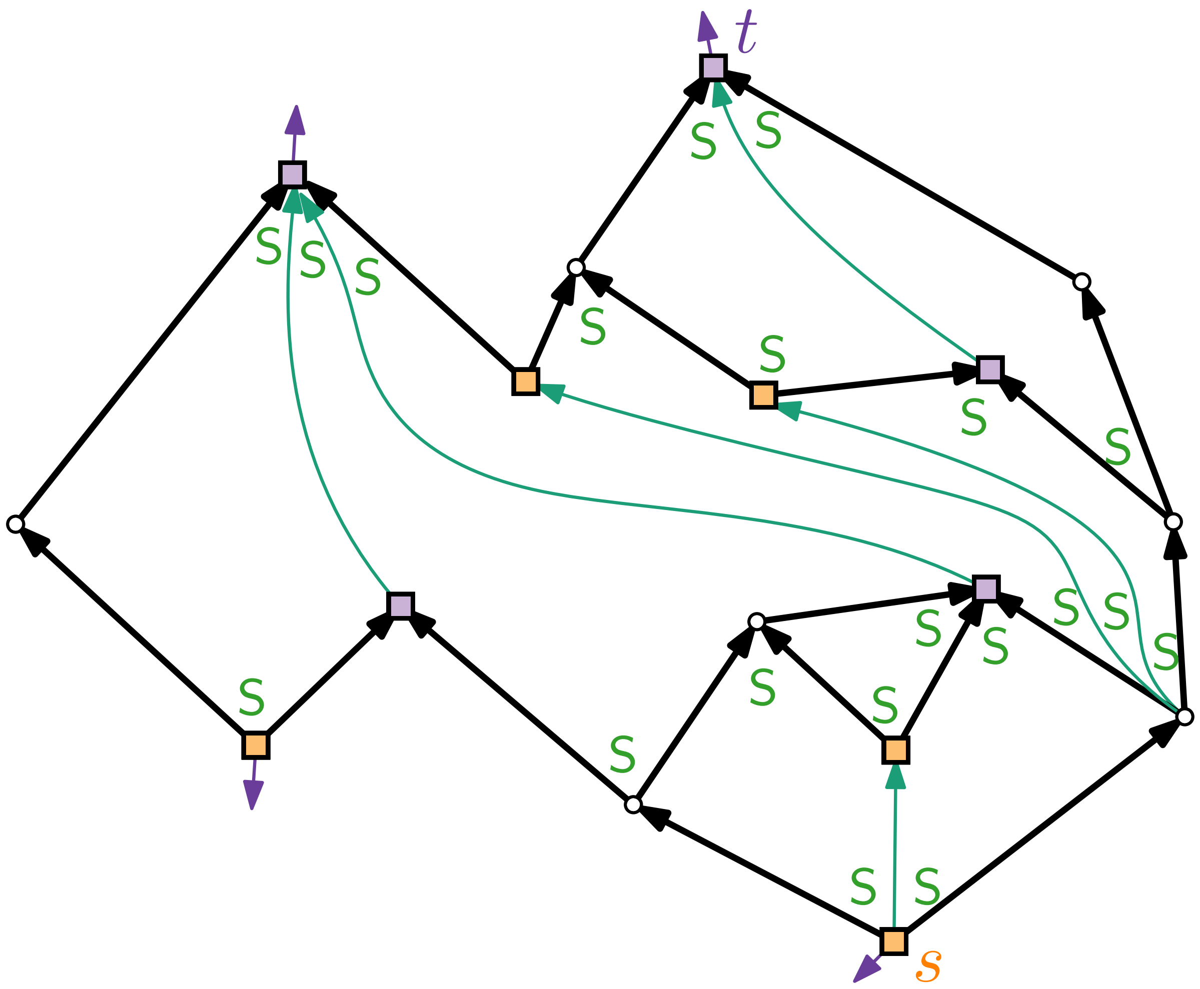
Refinement Example



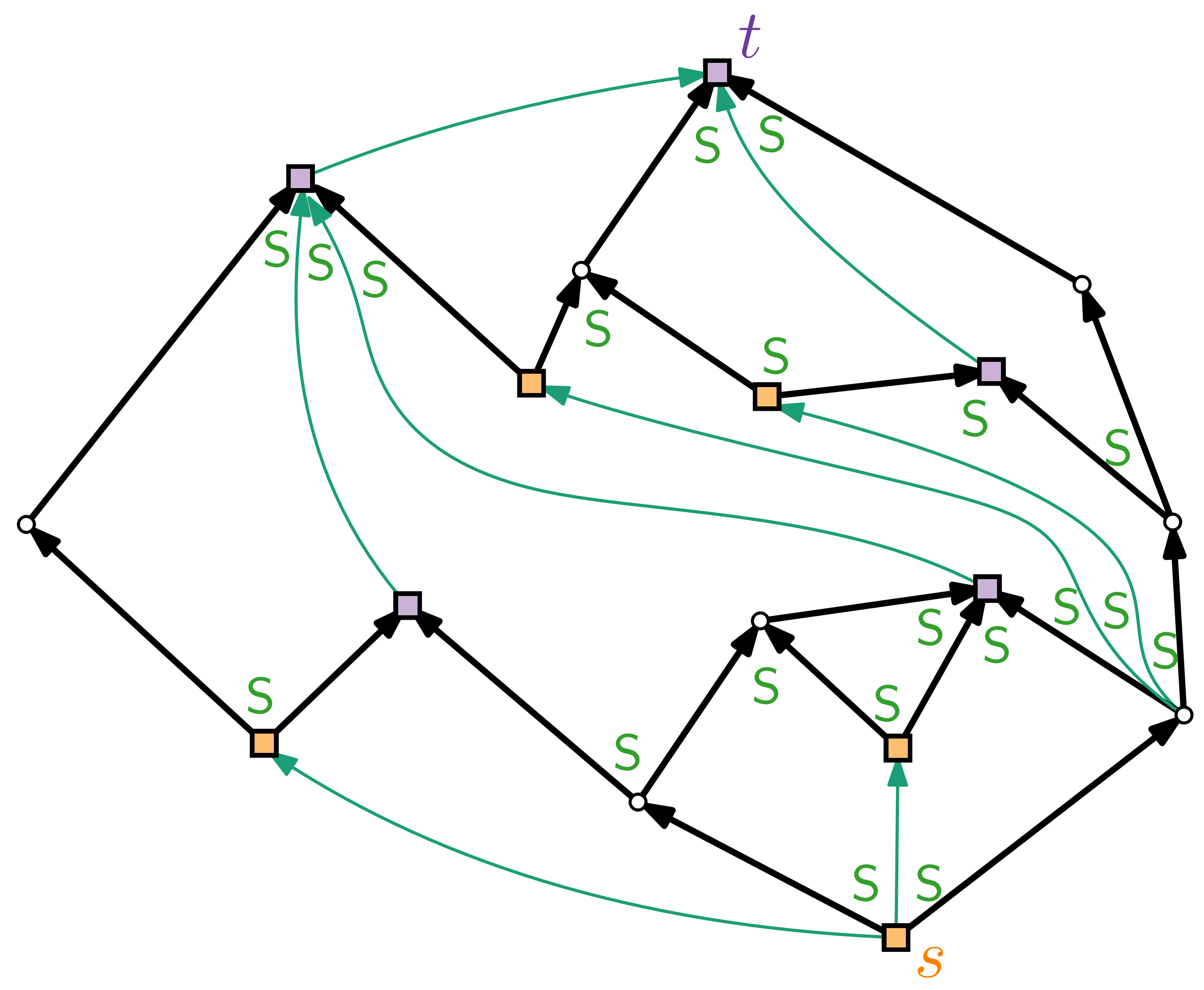
Refinement Example



Refinement Example



Refinement Example



Result Upward Planarity Test

Theorem 2. [Bertolazzi, Di Battista, Mannino, Tamassia '94]
Given an *embedded* planar digraph G,
we can test in quadratic time whether G is upward planar.

Result Upward Planarity Test

Theorem 2. [Bertolazzi, Di Battista, Mannino, Tamassia '94]
Given an *embedded* planar digraph G,
we can test in quadratic time whether G is upward planar.

Proof.

- Test for bimodality.

Result Upward Planarity Test

Theorem 2. [Bertolazzi, Di Battista, Mannino, Tamassia '94]
Given an *embedded* planar digraph G,
we can test in quadratic time whether G is upward planar.

Proof.

- Test for bimodality.
- Test for a consistent assignment Φ (via flow network).

Result Upward Planarity Test

Theorem 2. [Bertolazzi, Di Battista, Mannino, Tamassia '94]
Given an *embedded* planar digraph G,
we can test in quadratic time whether G is upward planar.

Proof.

- Test for bimodality.
- Test for a consistent assignment Φ (via flow network).
- If G bimodal and Φ exists, refine G to plane st-digraph H.

Result Upward Planarity Test

Theorem 2. [Bertolazzi, Di Battista, Mannino, Tamassia '94]
Given an *embedded* planar digraph G,
we can test in quadratic time whether G is upward planar.

Proof.

- Test for bimodality.
- Test for a consistent assignment Φ (via flow network).
- If G bimodal and Φ exists, refine G to plane st-digraph H.
- Draw H upward planar.

Result Upward Planarity Test

Theorem 2. [Bertolazzi, Di Battista, Mannino, Tamassia '94]
Given an *embedded* planar digraph G,
we can test in quadratic time whether G is upward planar.

Proof.

- Test for bimodality.
- Test for a consistent assignment Φ (via flow network).
- If G bimodal and Φ exists, refine G to plane st-digraph H.
- Draw H upward planar.
- Deleted edges added in refinement step.

Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

Flow network.
$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$

- $W =$
- $E' =$
- $\ell(e) =$
- $u(e) =$
- $b(w) =$

Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

nodes of flow network   edges of flow network   supplies/demands of nodes   lower/upper bounds on edge capcities

Flow network.
$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$

- $W =$
- $E' =$
- $\ell(e) =$
- $u(e) =$

- $b(w) =$

Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

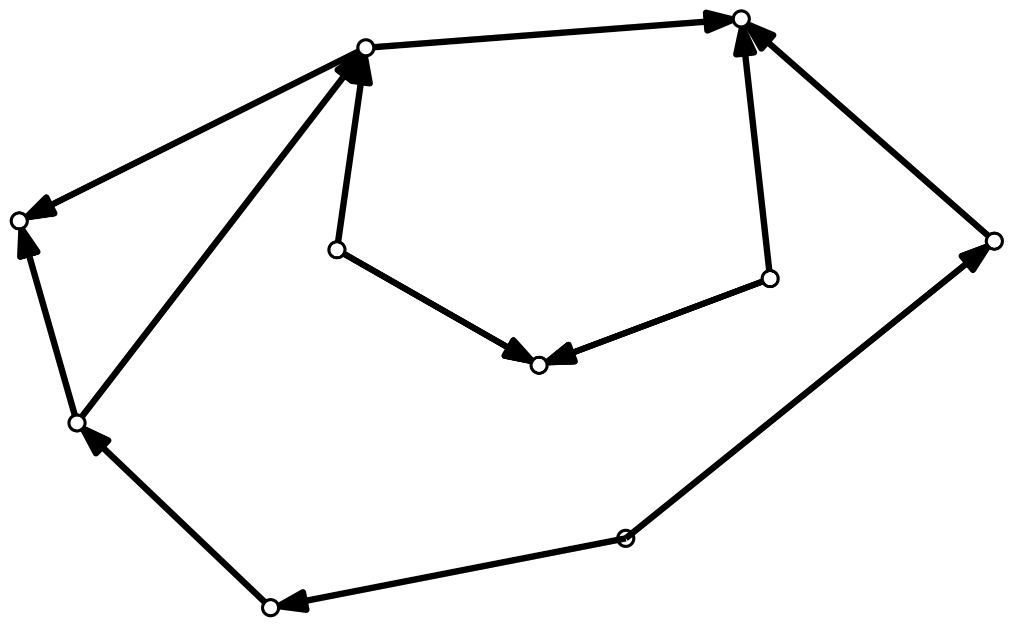
nodes of flow network edges of flow network
supplies/demands of nodes lower/upper bounds on edge capcities

Flow network.

$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$

- $W =$
- $E' =$
- $\ell(e) =$
- $u(e) =$
- $b(w) =$

Example.



Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

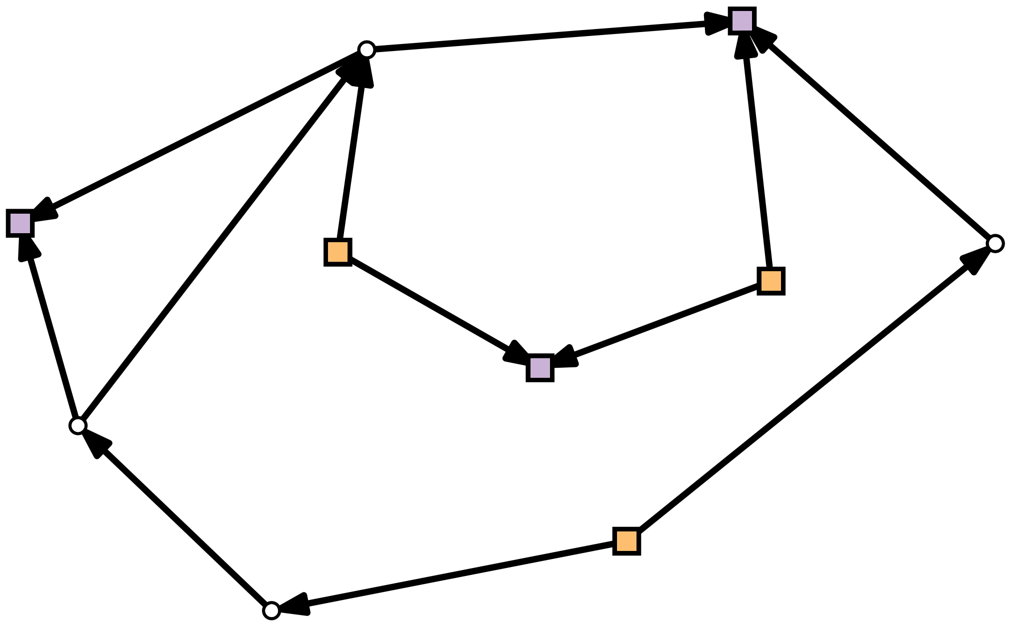
nodes of flow network   edges of flow network   supplies/demands of nodes   lower/upper bounds on edge capcities

Flow network.

$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$

- $W = \{v \in V(G) \mid v$ source or sink$\} \cup$
- $E' =$
- $\ell(e) =$
- $u(e) =$
- $b(w) =$

Example.



Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

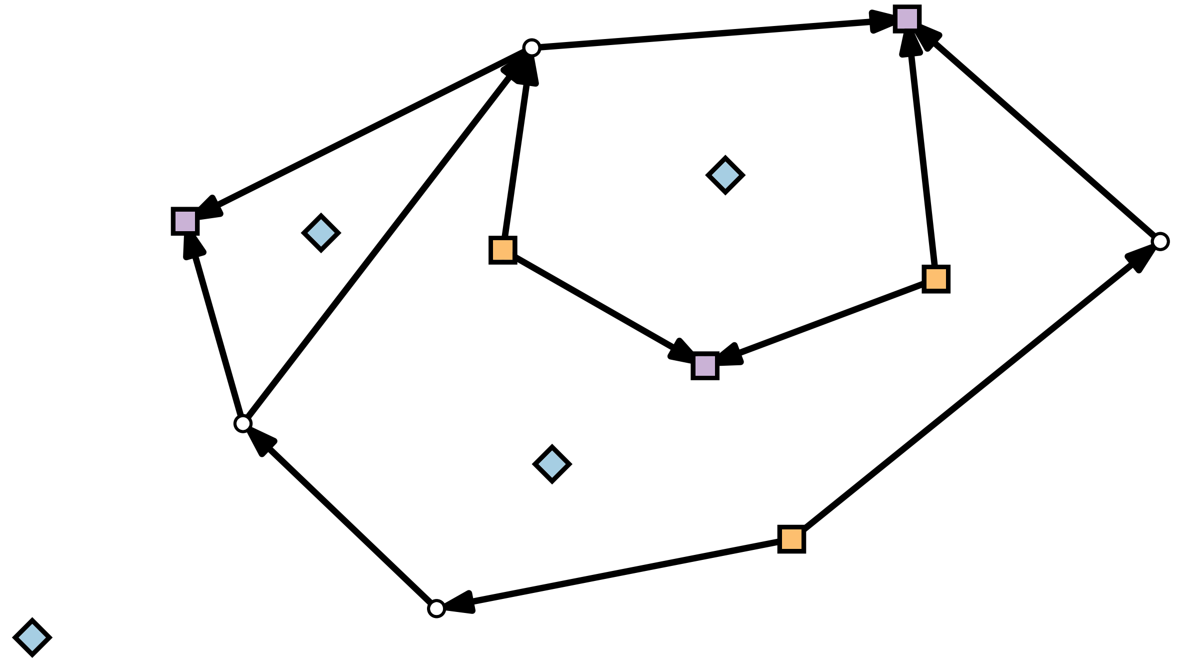
nodes of flow network edges of flow network lower/upper bounds on edge capcities
supplies/demands of nodes

Flow network.

$$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$$

- $W = \{v \in V(G) \mid v$ source or sink$\} \cup F(G)$
- $E' =$
- $\ell(e) =$
- $u(e) =$
- $b(w) =$

Example.



Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

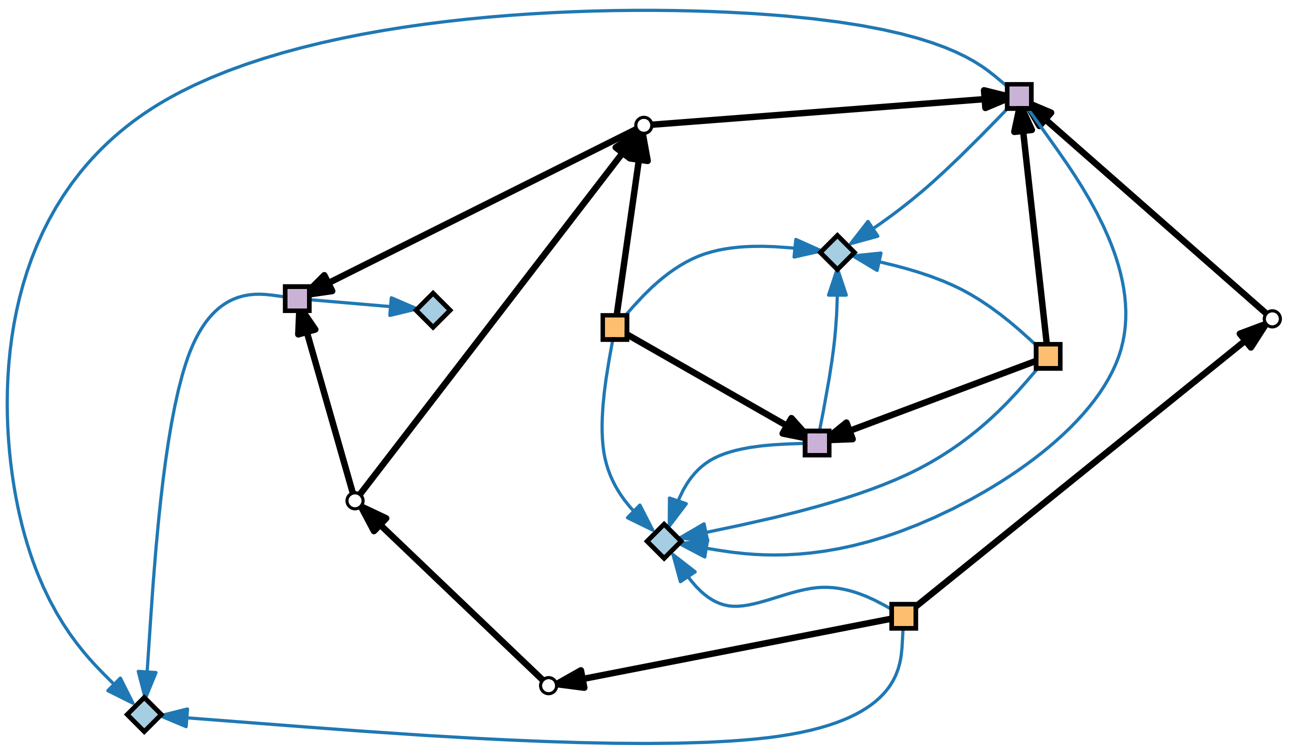
nodes of flow network
edges of flow network
supplies/demands of nodes
lower/upper bounds on edge capcities

Flow network.

$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$

- $W = \{v \in V(G) \mid v$ source or sink$\} \cup F(G)$
- $E' = \{(v, f) \mid v$ incident to $f\}$
- $\ell(e) =$
- $u(e) =$
- $b(w) =$

Example.



Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

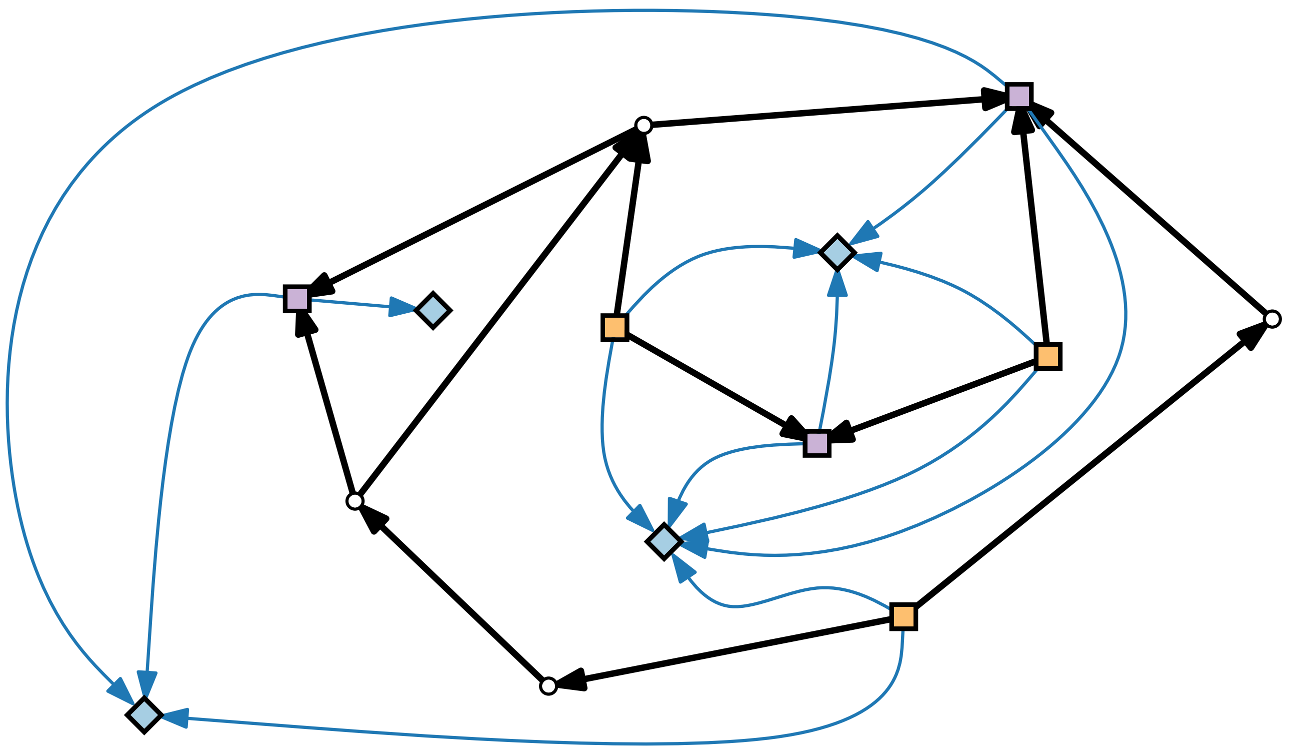
nodes of flow network — edges of flow network — supplies/demands of nodes — lower/upper bounds on edge capcities

Flow network.

$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$

- $W = \{v \in V(G) \mid v$ source or sink$\} \cup F(G)$
- $E' = \{(v, f) \mid v$ incident to $f\}$
- $\ell(e) = 0 \; \forall e \in E'$
- $u(e) = 1 \; \forall e \in E'$
- $b(w) =$

Example.



Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

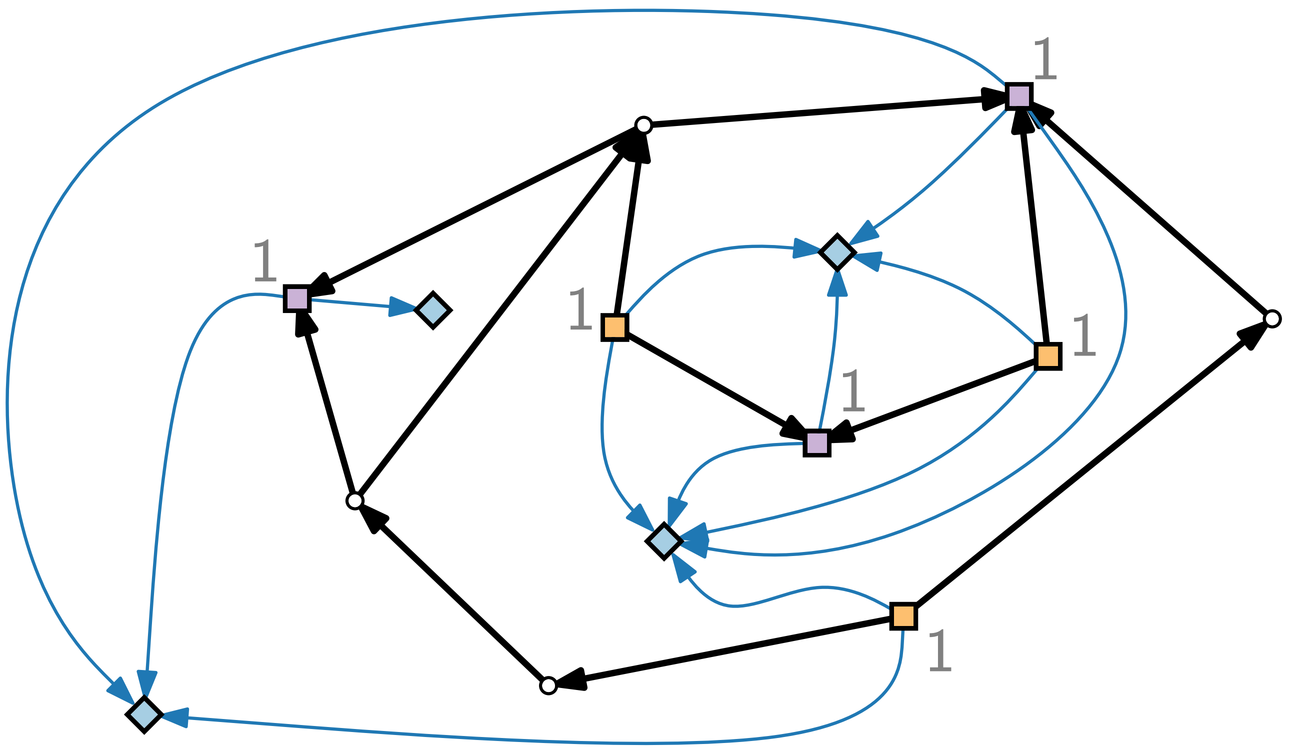
nodes of flow network edges of flow network supplies/demands of nodes lower/upper bounds on edge capcities

Flow network.

$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$

- $W = \{v \in V(G) \mid v$ source or sink$\} \cup F(G)$
- $E' = \{(v, f) \mid v$ incident to $f\}$
- $\ell(e) = 0 \; \forall e \in E'$
- $u(e) = 1 \; \forall e \in E'$
- $b(w) = \begin{cases} 1 & \forall w \in W \cap V(G) \\ & \\ & \end{cases}$

Example.



Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

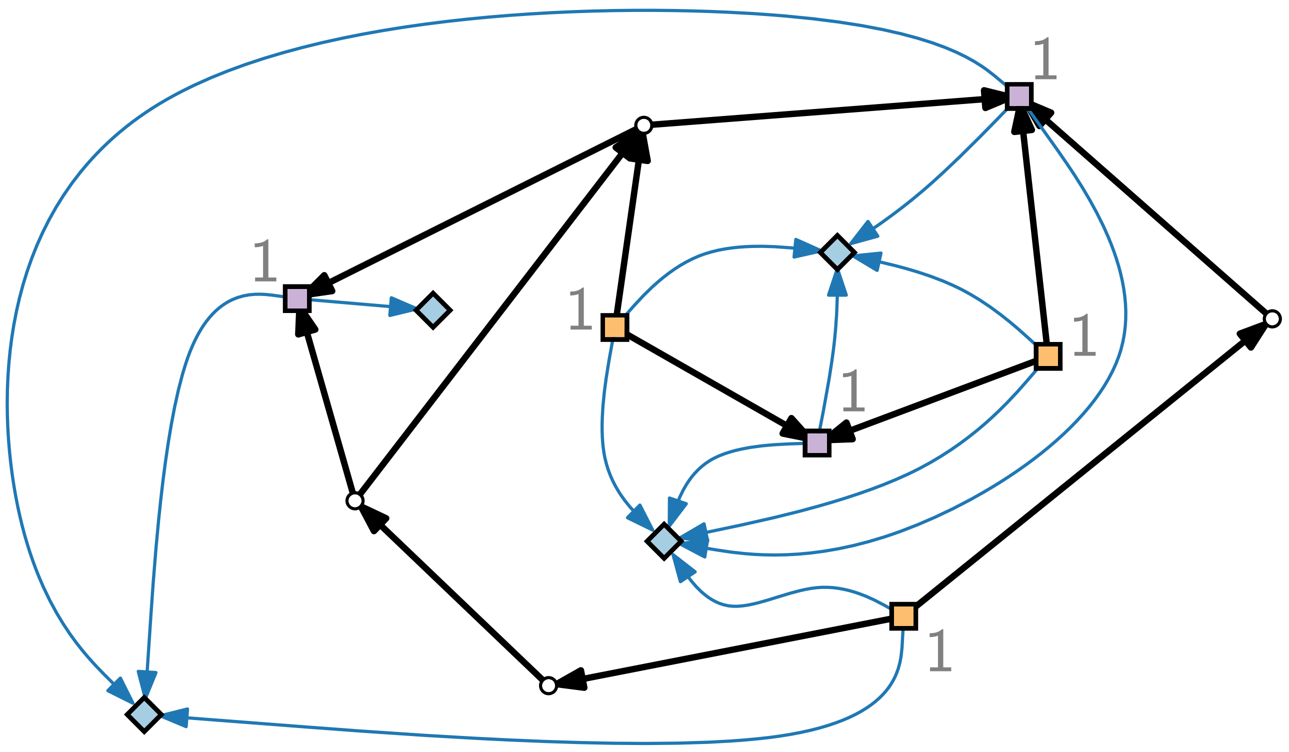
nodes of flow network   edges of flow network   supplies/demands of nodes   lower/upper bounds on edge capcities

Flow network.

$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$

- $W = \{v \in V(G) \mid v$ source or sink$\} \cup F(G)$
- $E' = \{(v, f) \mid v$ incident to $f\}$
- $\ell(e) = 0 \; \forall e \in E'$
- $u(e) = 1 \; \forall e \in E'$
- $b(w) = \begin{cases} 1 & \forall w \in W \cap V(G) \\ -(A(w) - 1) & \forall w \in F(G) \setminus \{f_0\} \end{cases}$

Example.



Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

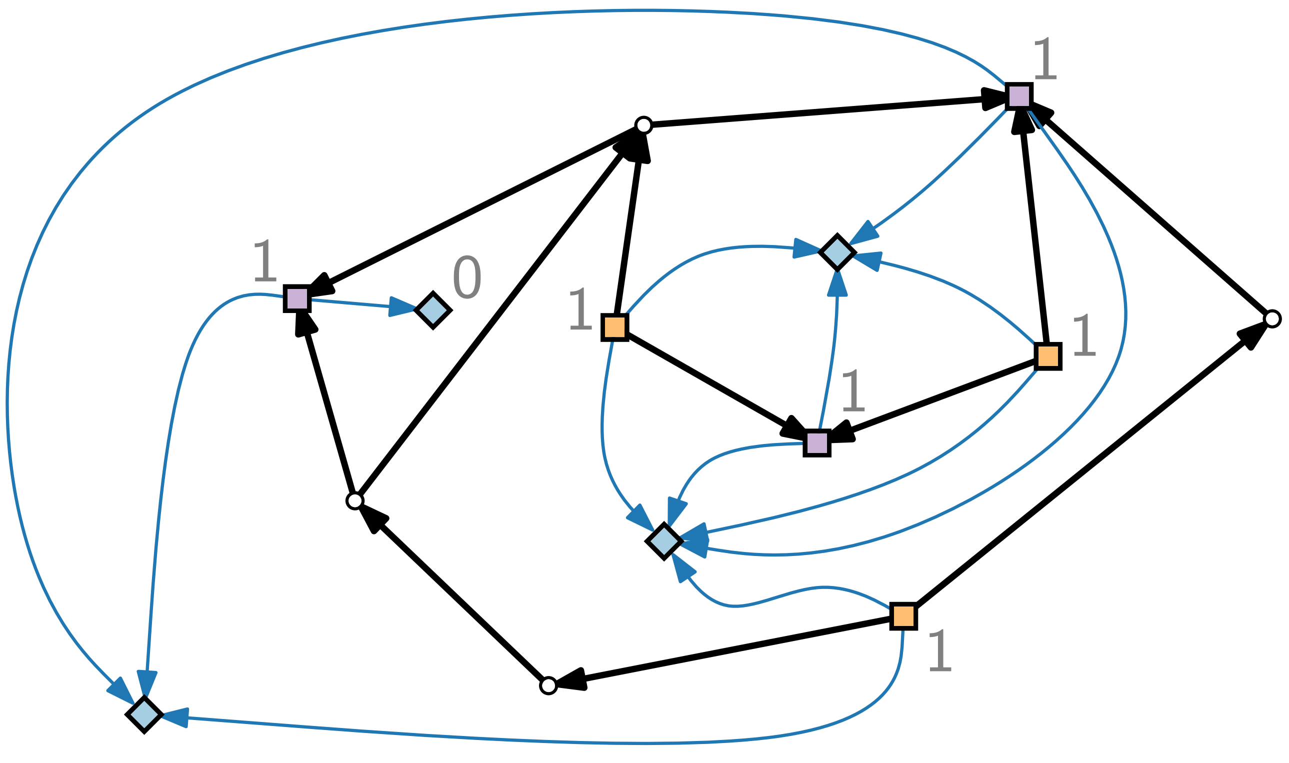
nodes of flow network edges of flow network supplies/demands of nodes lower/upper bounds on edge capcities

Flow network.

$$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$$

- $W = \{v \in V(G) \mid v \text{ source or sink}\} \cup F(G)$
- $E' = \{(v, f) \mid v \text{ incident to } f\}$
- $\ell(e) = 0 \ \forall e \in E'$
- $u(e) = 1 \ \forall e \in E'$
- $b(w) = \begin{cases} 1 & \forall w \in W \cap V(G) \\ -(A(w) - 1) & \forall w \in F(G) \setminus \{f_0\} \end{cases}$

Example.



Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

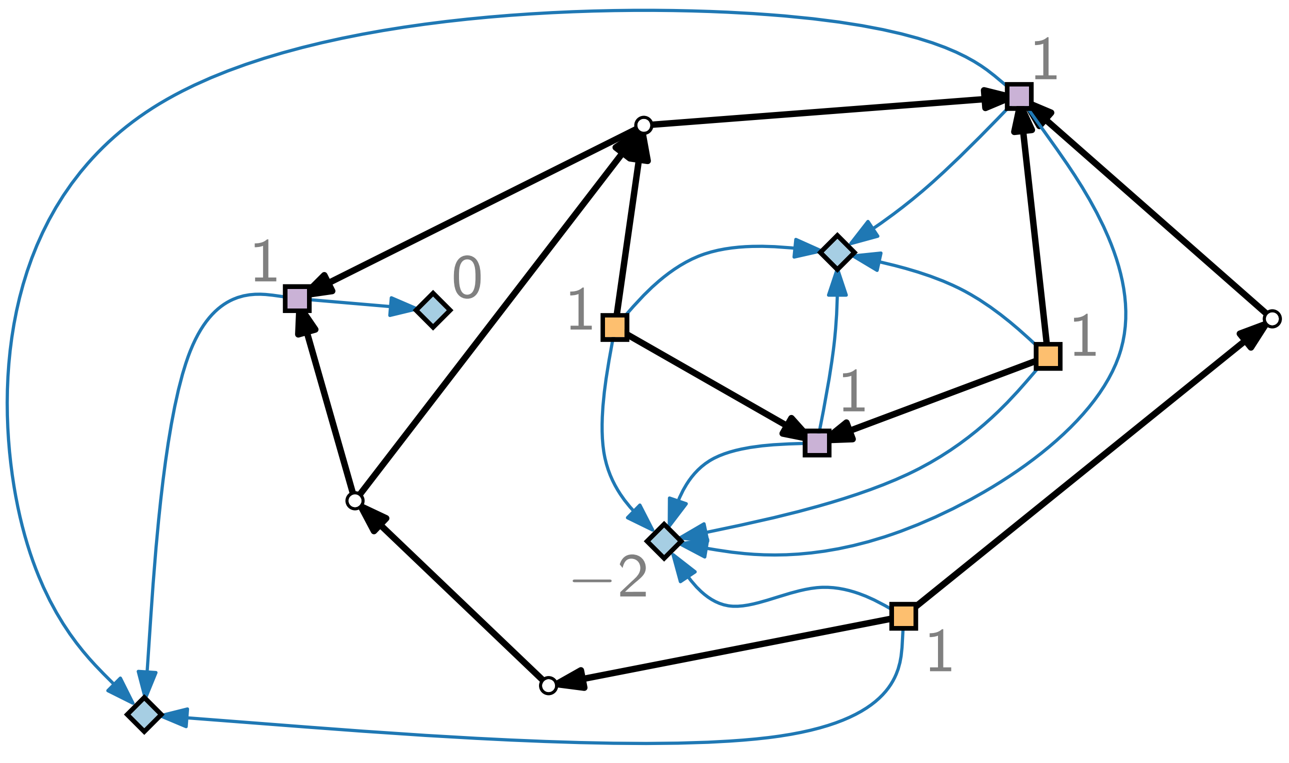
nodes of flow network   edges of flow network
supplies/demands of nodes   lower/upper bounds on edge capcities

Flow network.

$$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$$

- $W = \{v \in V(G) \mid v$ source or sink$\} \cup F(G)$
- $E' = \{(v, f) \mid v$ incident to $f\}$
- $\ell(e) = 0 \ \forall e \in E'$
- $u(e) = 1 \ \forall e \in E'$
- $b(w) = \begin{cases} 1 & \forall w \in W \cap V(G) \\ -(A(w) - 1) & \forall w \in F(G) \setminus \{f_0\} \\ \end{cases}$

Example.



Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

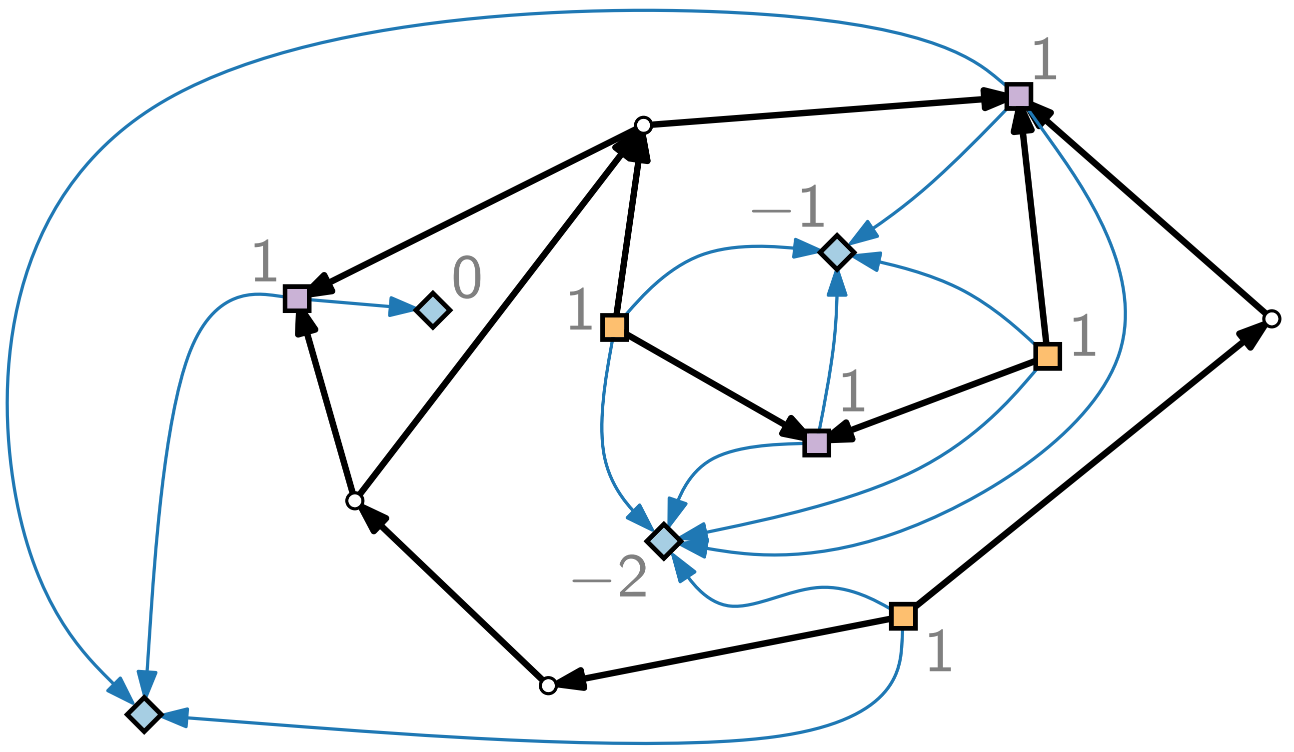
nodes of flow network   edges of flow network
supplies/demands of nodes   lower/upper bounds on edge capcities

Flow network.

$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$

- $W = \{v \in V(G) \mid v$ source or sink$\} \cup F(G)$
- $E' = \{(v, f) \mid v$ incident to $f\}$
- $\ell(e) = 0 \ \forall e \in E'$
- $u(e) = 1 \ \forall e \in E'$
- $b(w) = \begin{cases} 1 & \forall w \in W \cap V(G) \\ -(A(w) - 1) & \forall w \in F(G) \setminus \{f_0\} \end{cases}$

Example.



Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

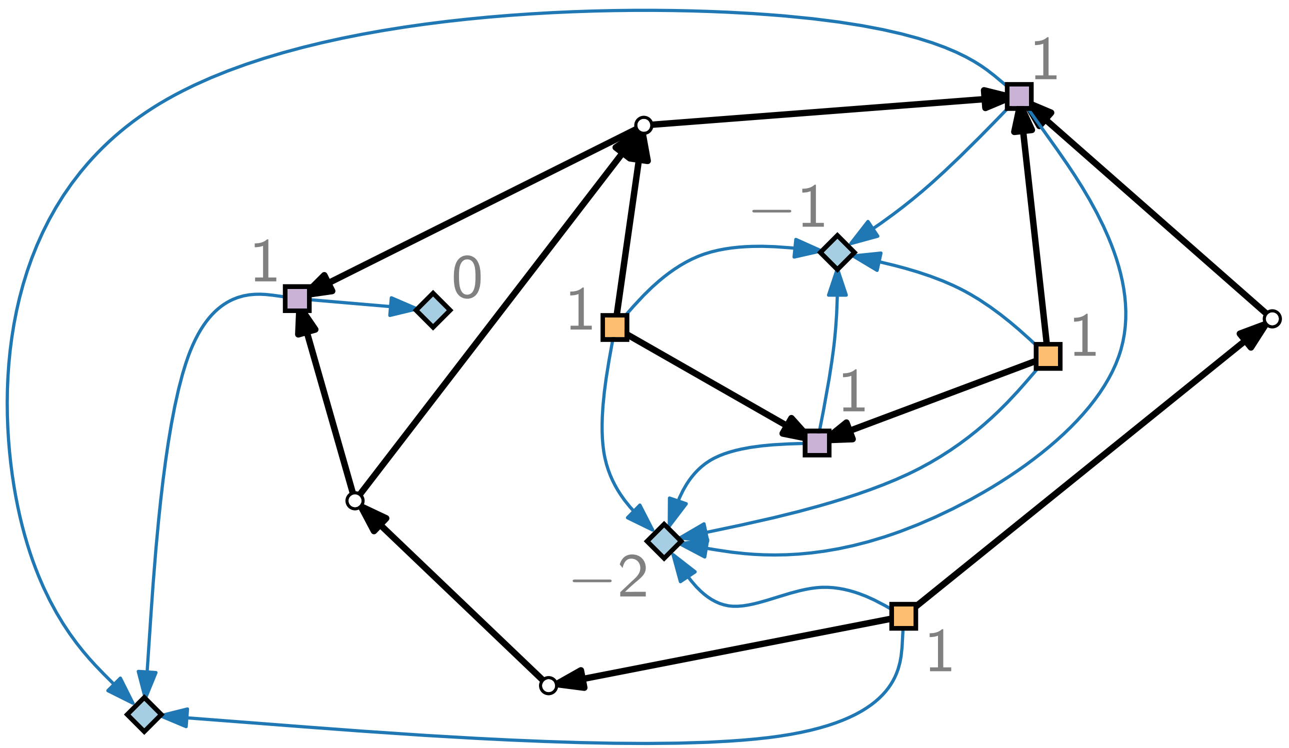
nodes of flow network edges of flow network supplies/demands of nodes lower/upper bounds on edge capcities

Flow network.

$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$

- $W = \{v \in V(G) \mid v$ source or sink$\} \cup F(G)$
- $E' = \{(v, f) \mid v$ incident to $f\}$
- $\ell(e) = 0 \; \forall e \in E'$
- $u(e) = 1 \; \forall e \in E'$
- $b(w) = \begin{cases} 1 & \forall w \in W \cap V(G) \\ -(A(w) - 1) & \forall w \in F(G) \setminus \{f_0\} \\ -(A(w) + 1) & w = f_0 \end{cases}$

Example.



Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

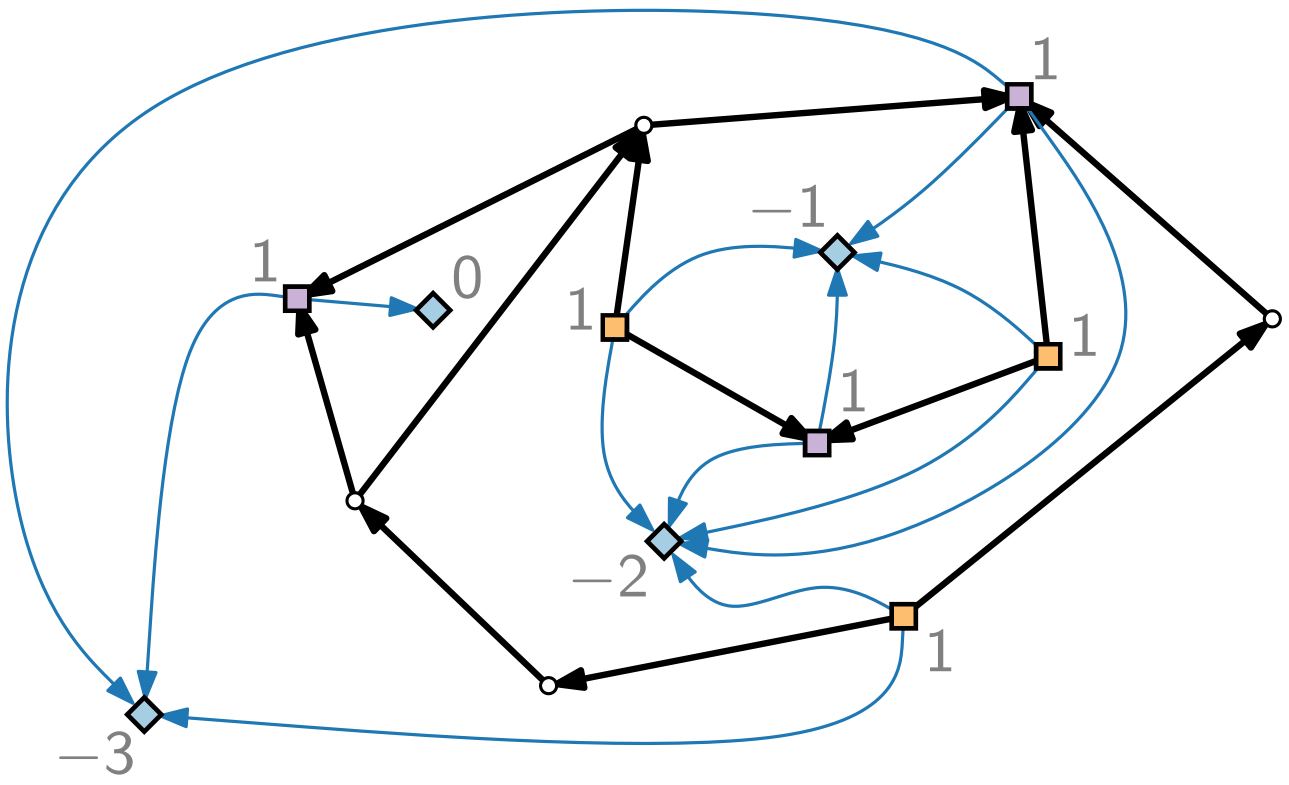
nodes of flow network edges of flow network supplies/demands of nodes lower/upper bounds on edge capcities

Flow network.

$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$

- $W = \{v \in V(G) \mid v$ source or sink$\} \cup F(G)$
- $E' = \{(v, f) \mid v$ incident to $f\}$
- $\ell(e) = 0 \;\forall e \in E'$
- $u(e) = 1 \;\forall e \in E'$
- $b(w) = \begin{cases} 1 & \forall w \in W \cap V(G) \\ -(A(w) - 1) & \forall w \in F(G) \setminus \{f_0\} \\ -(A(w) + 1) & w = f_0 \end{cases}$

Example.



Finding a Consistent Assignment

Idea. Flow $(v, f) = 1$
from global source / sink v to the incident face f its large angle gets assigned to.

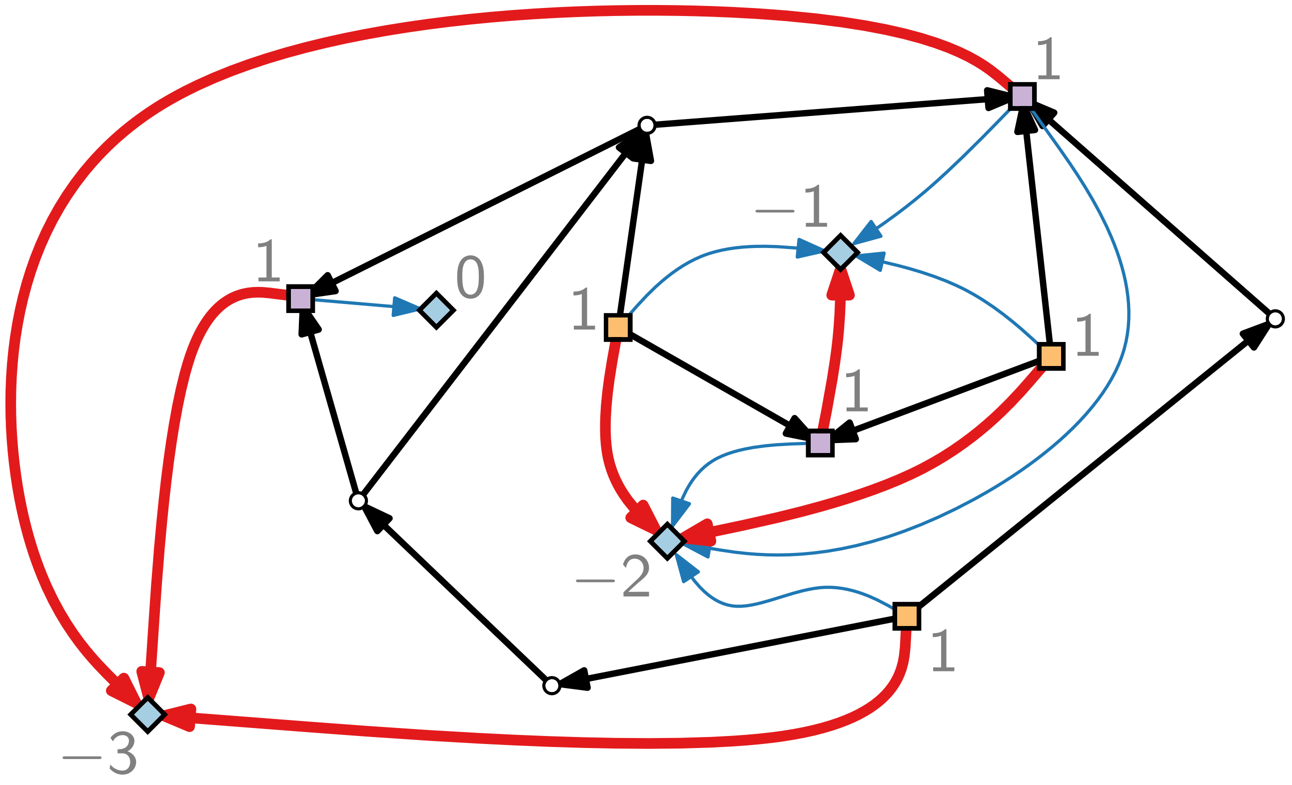
nodes of flow network
edges of flow network
supplies/demands of nodes
lower/upper bounds on edge capcities

Flow network.

$N_{F,f_0}(G) = ((W, E'); b; \ell; u)$

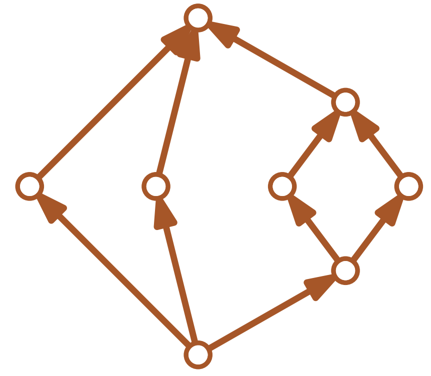
- $W = \{v \in V(G) \mid v$ source or sink$\} \cup F(G)$
- $E' = \{(v, f) \mid v$ incident to $f\}$
- $\ell(e) = 0 \; \forall e \in E'$
- $u(e) = 1 \; \forall e \in E'$
- $b(w) = \begin{cases} 1 & \forall w \in W \cap V(G) \\ -(A(w) - 1) & \forall w \in F(G) \setminus \{f_0\} \\ -(A(w) + 1) & w = f_0 \end{cases}$

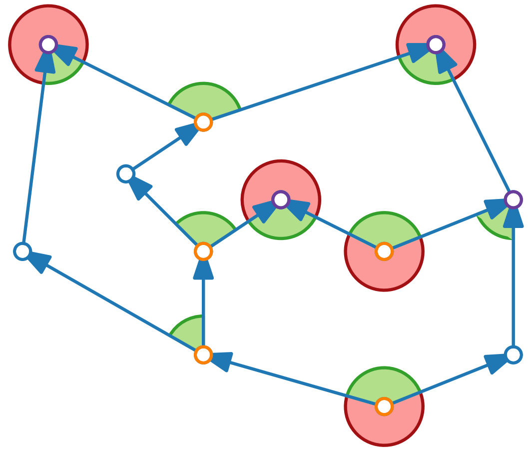
Example.



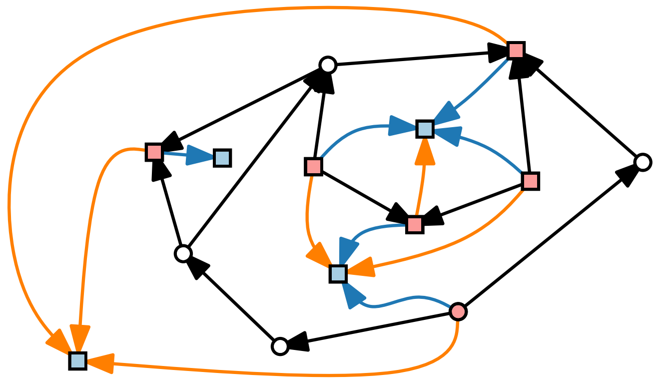
Visualization of Graphs

Lecture 5:
Upward Planar Drawings





Part II:
Series-Parallel Graphs



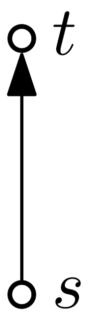
Series-Parallel Graphs

A graph G is **series-parallel** if

Series-Parallel Graphs

A graph G is **series-parallel** if

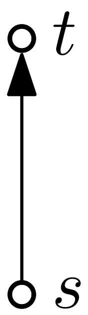
- it contains a single (directed) edge (s, t), or

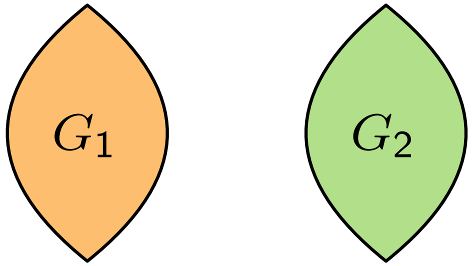


Series-Parallel Graphs

A graph G is **series-parallel** if

- it contains a single (directed) edge (s, t), or
- it consists of two series-parallel graphs G_1, G_2

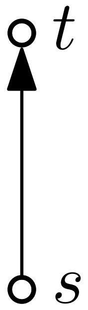


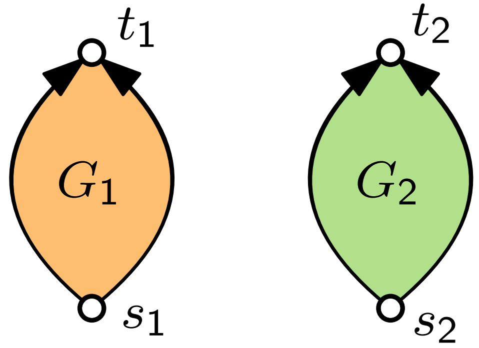


Series-Parallel Graphs

A graph G is **series-parallel** if

- it contains a single (directed) edge (s, t), or
- it consists of two series-parallel graphs G_1, G_2 with sources s_1, s_2 and sinks t_1, t_2

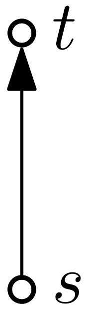


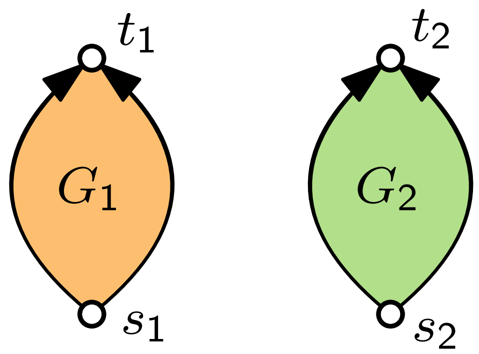


Series-Parallel Graphs

A graph G is **series-parallel** if

- it contains a single (directed) edge (s, t), or
- it consists of two series-parallel graphs G_1, G_2 with sources s_1, s_2 and sinks t_1, t_2 that are combined using one of the following rules:

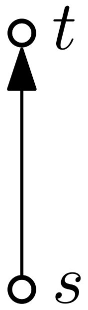




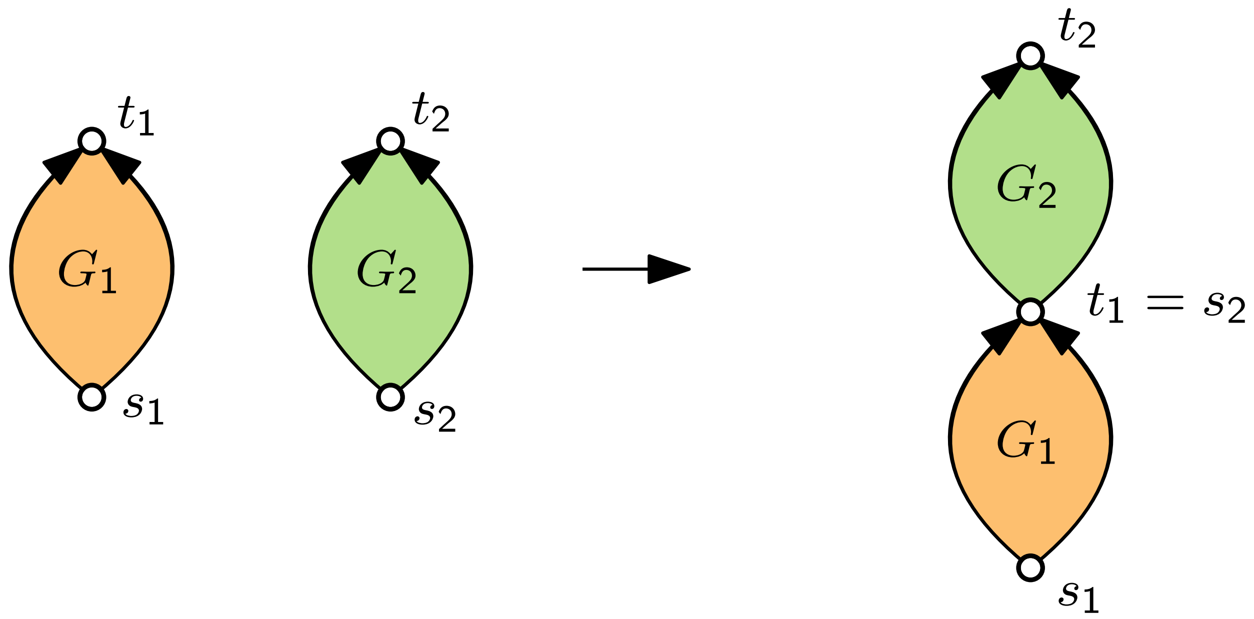
Series-Parallel Graphs

A graph G is **series-parallel** if

- it contains a single (directed) edge (s, t), or
- it consists of two series-parallel graphs G_1, G_2 with sources s_1, s_2 and sinks t_1, t_2 that are combined using one of the following rules:



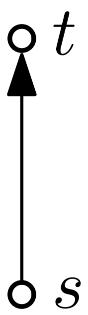
Series composition

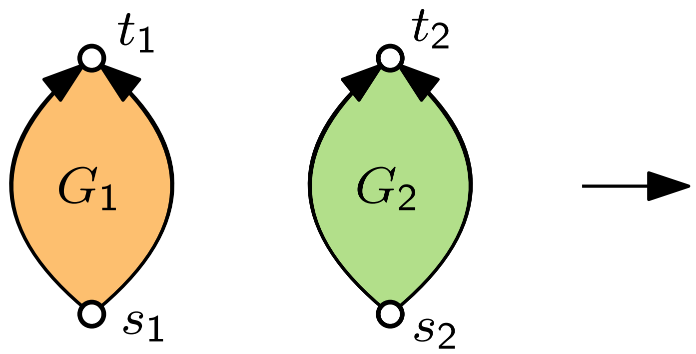


Series-Parallel Graphs

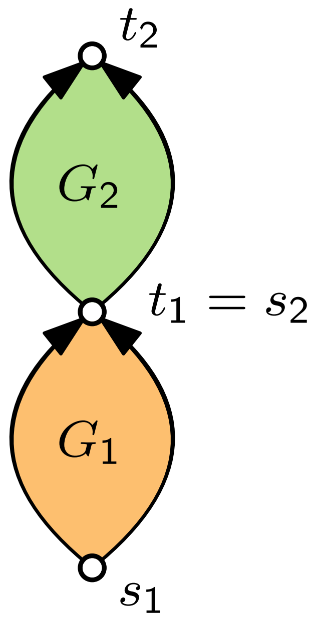
A graph G is **series-parallel** if

- it contains a single (directed) edge (s, t), or
- it consists of two series-parallel graphs G_1, G_2 with sources s_1, s_2 and sinks t_1, t_2 that are combined using one of the following rules:

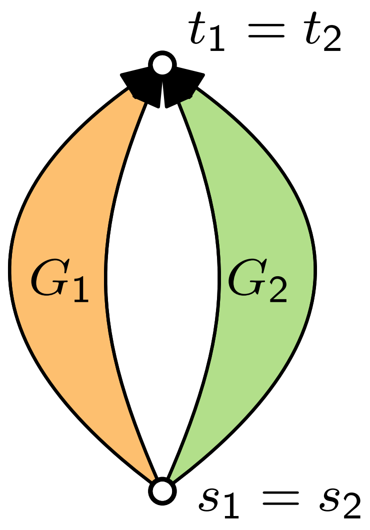




Series composition



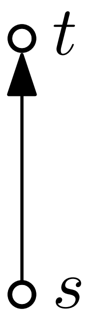
Parallel composition



Series-Parallel Graphs

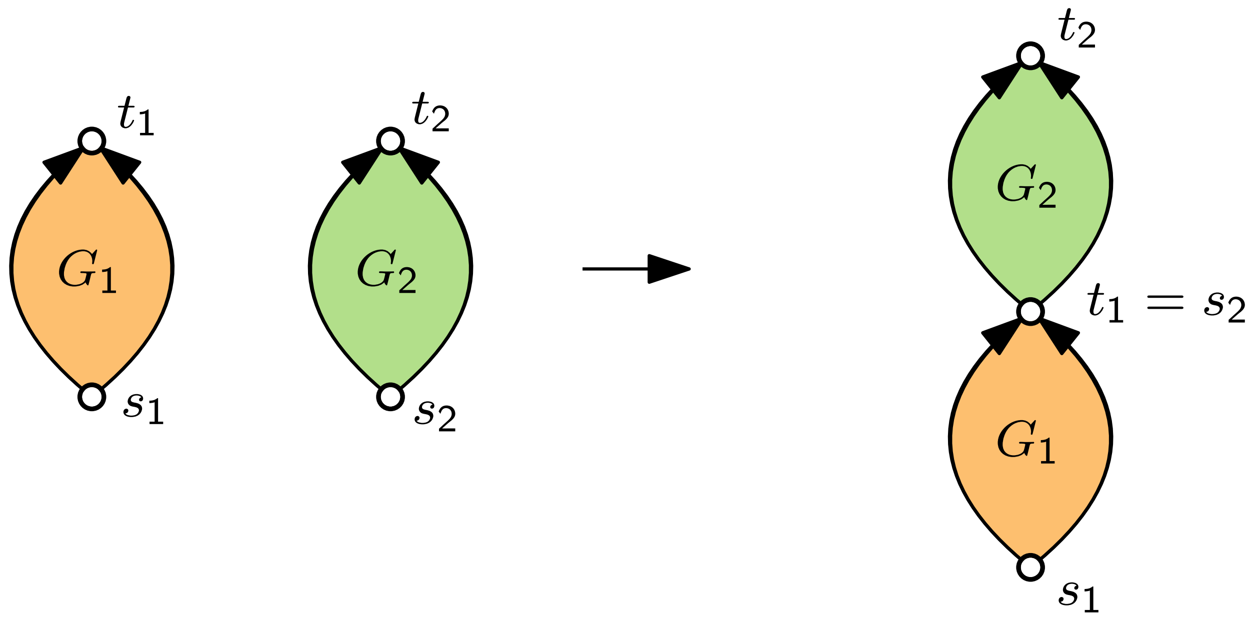
A graph G is **series-parallel** if

- it contains a single (directed) edge (s, t), or
- it consists of two series-parallel graphs G_1, G_2 with sources s_1, s_2 and sinks t_1, t_2 that are combined using one of the following rules:

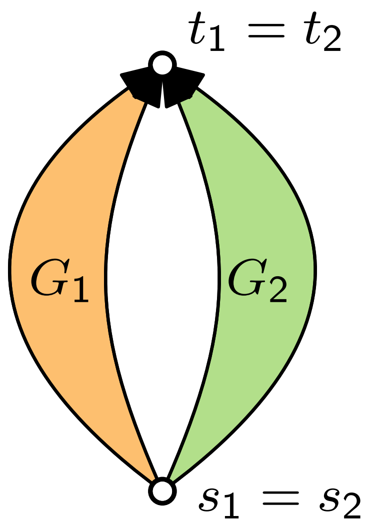


Convince yourself that series-parallel graphs are (upward) planar!

Series composition



Parallel composition



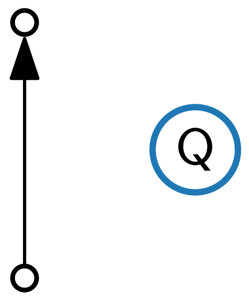
Series-Parallel Graphs – Decomposition Tree

A **decomposition tree** of G is a binary tree T with nodes of three types: **S**, **P** and **Q**.

Series-Parallel Graphs – Decomposition Tree

A **decomposition tree** of G is a binary tree T with nodes of three types: **S**, **P** and **Q**.

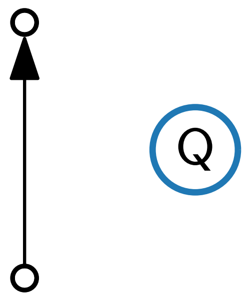
- A **Q**-node represents a single edge.

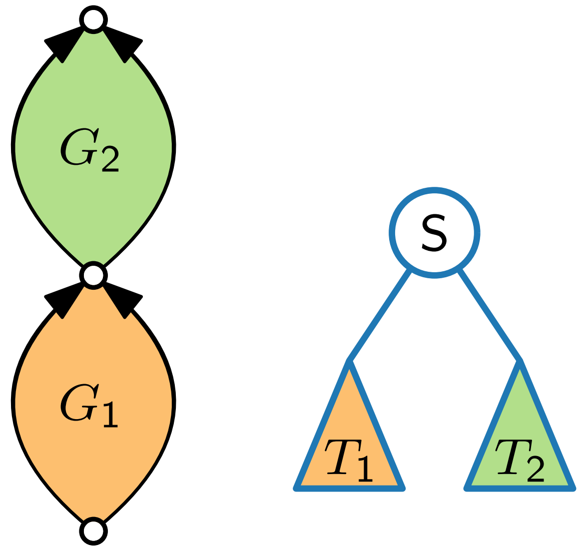


Series-Parallel Graphs – Decomposition Tree

A **decomposition tree** of G is a binary tree T with nodes of three types: **S**, **P** and **Q**.

- A **Q**-node represents a single edge.
- An **S**-node represents a series composition;
  its children T_1 and T_2 represent G_1 and G_2.

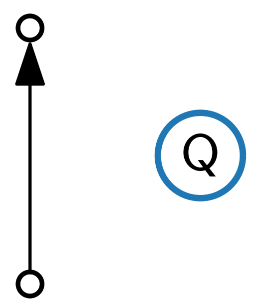


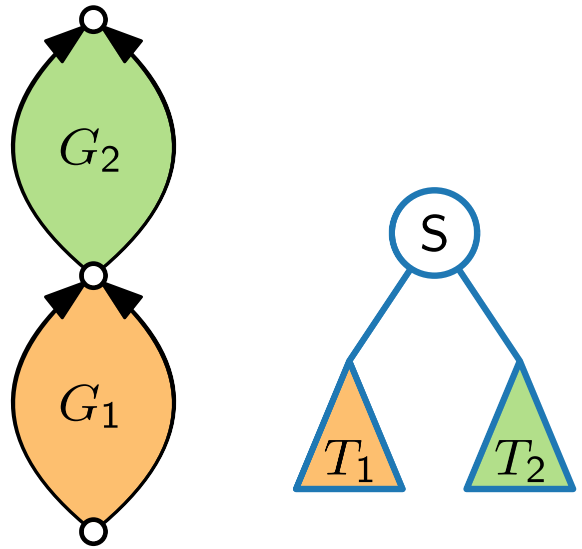


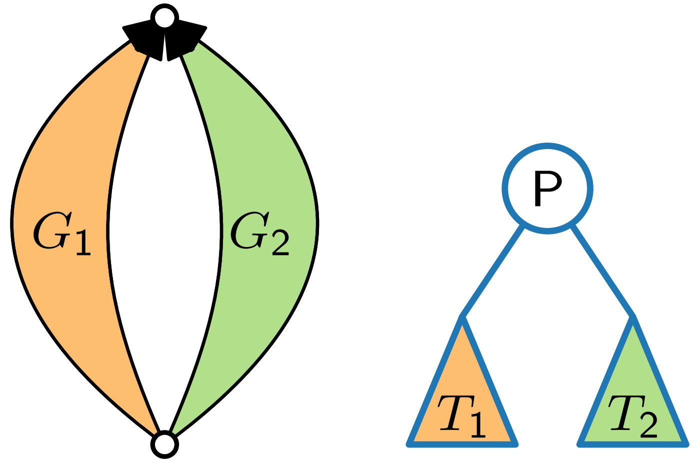
Series-Parallel Graphs – Decomposition Tree

A **decomposition tree** of G is a binary tree T with nodes of three types: **S**, **P** and **Q**.

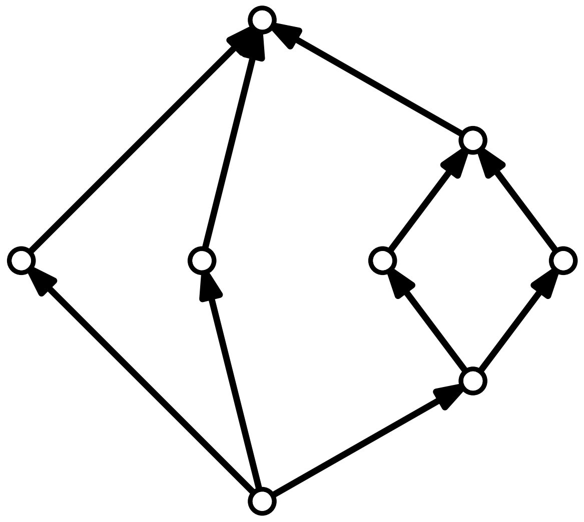
- A **Q**-node represents a single edge.
- An **S**-node represents a series composition; its children T_1 and T_2 represent G_1 and G_2.
- A **P**-node represents a parallel composition; its children T_1 and T_2 represent G_1 and G_2



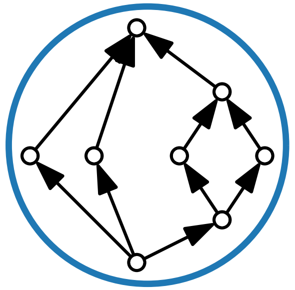




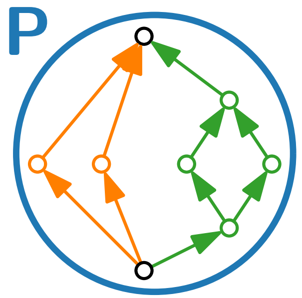
Series-Parallel Graphs – Decomposition Example



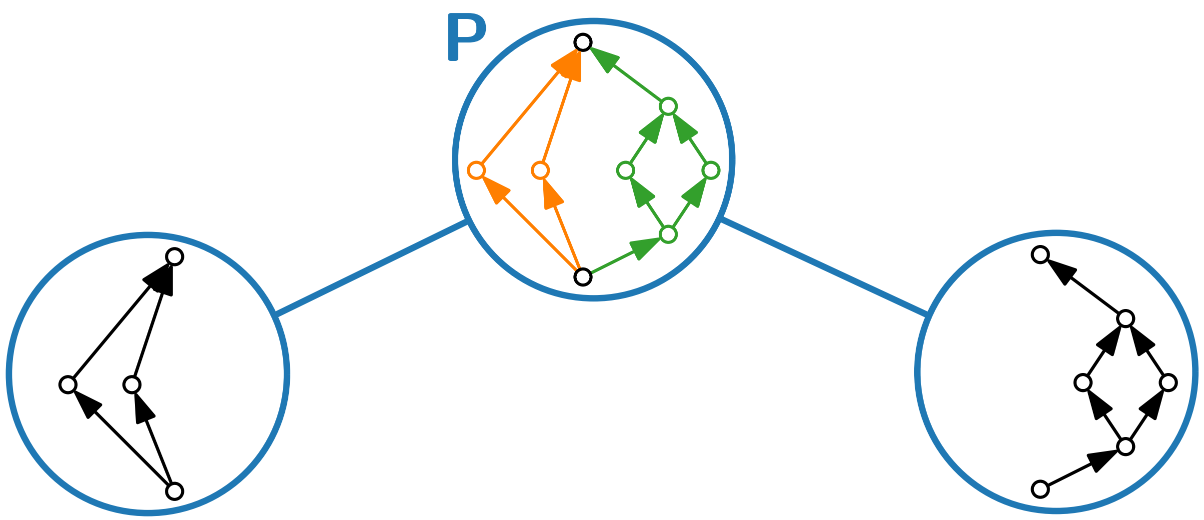
Series-Parallel Graphs – Decomposition Example



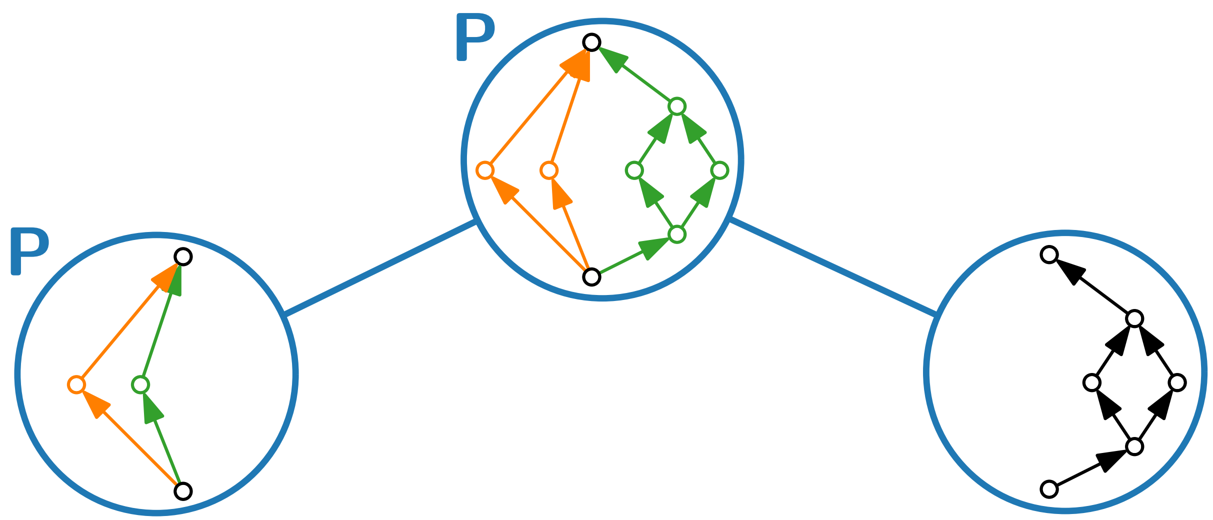
Series-Parallel Graphs – Decomposition Example



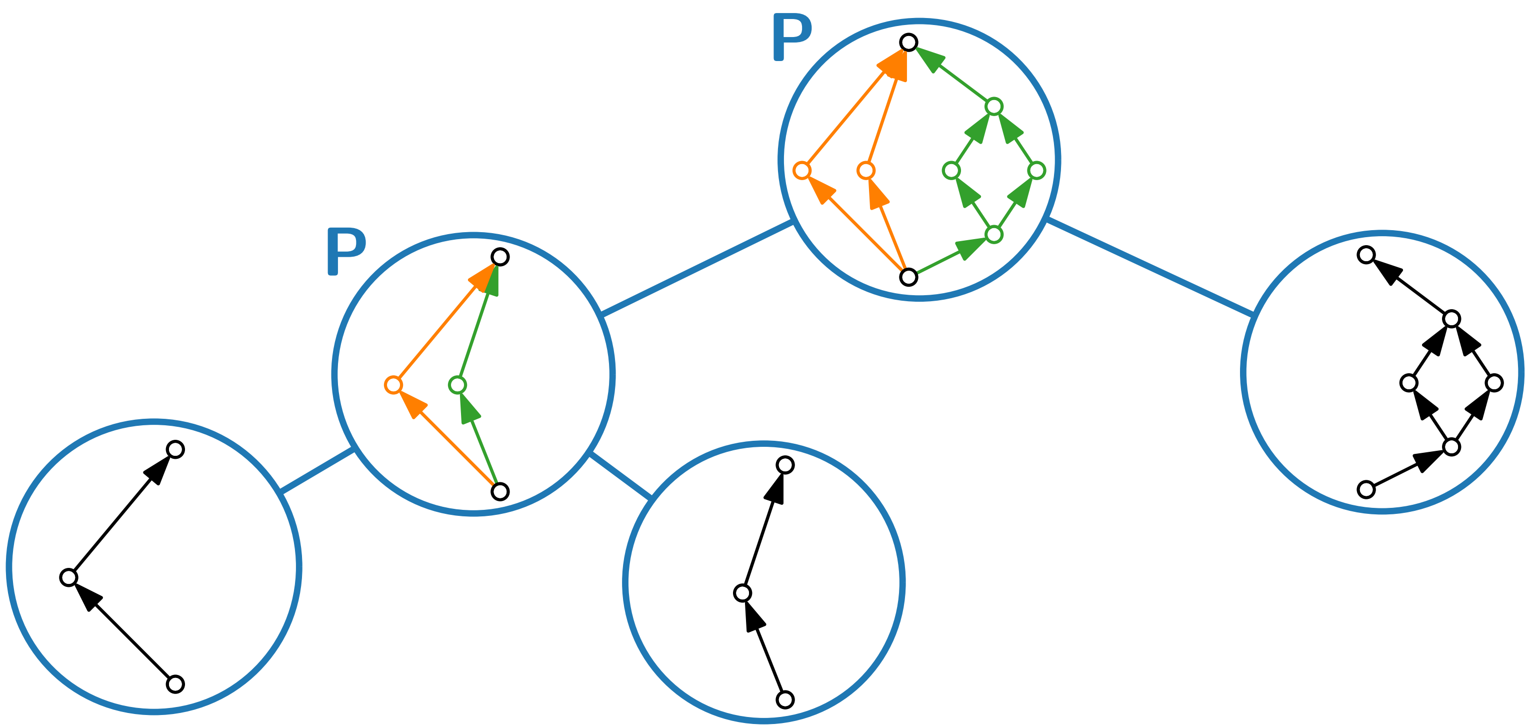
Series-Parallel Graphs – Decomposition Example



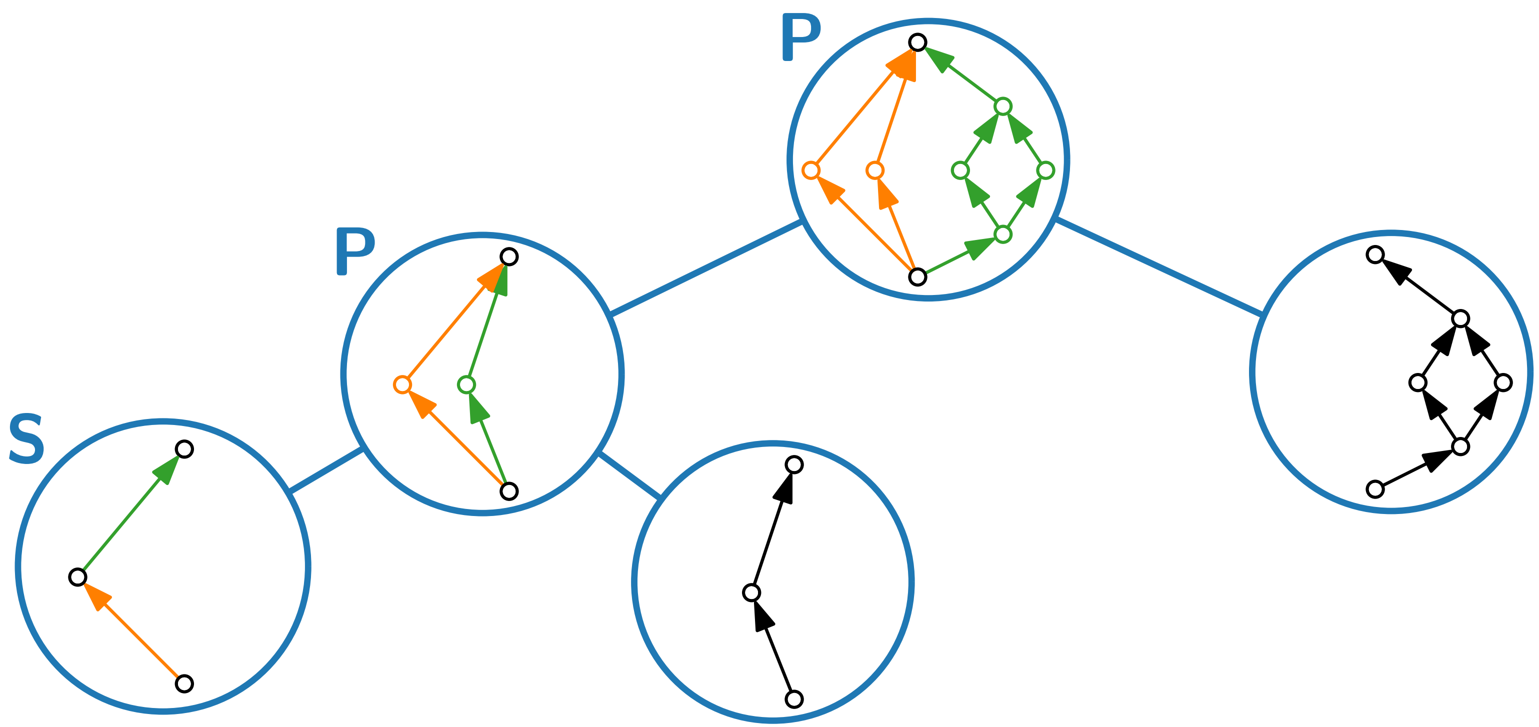
Series-Parallel Graphs – Decomposition Example



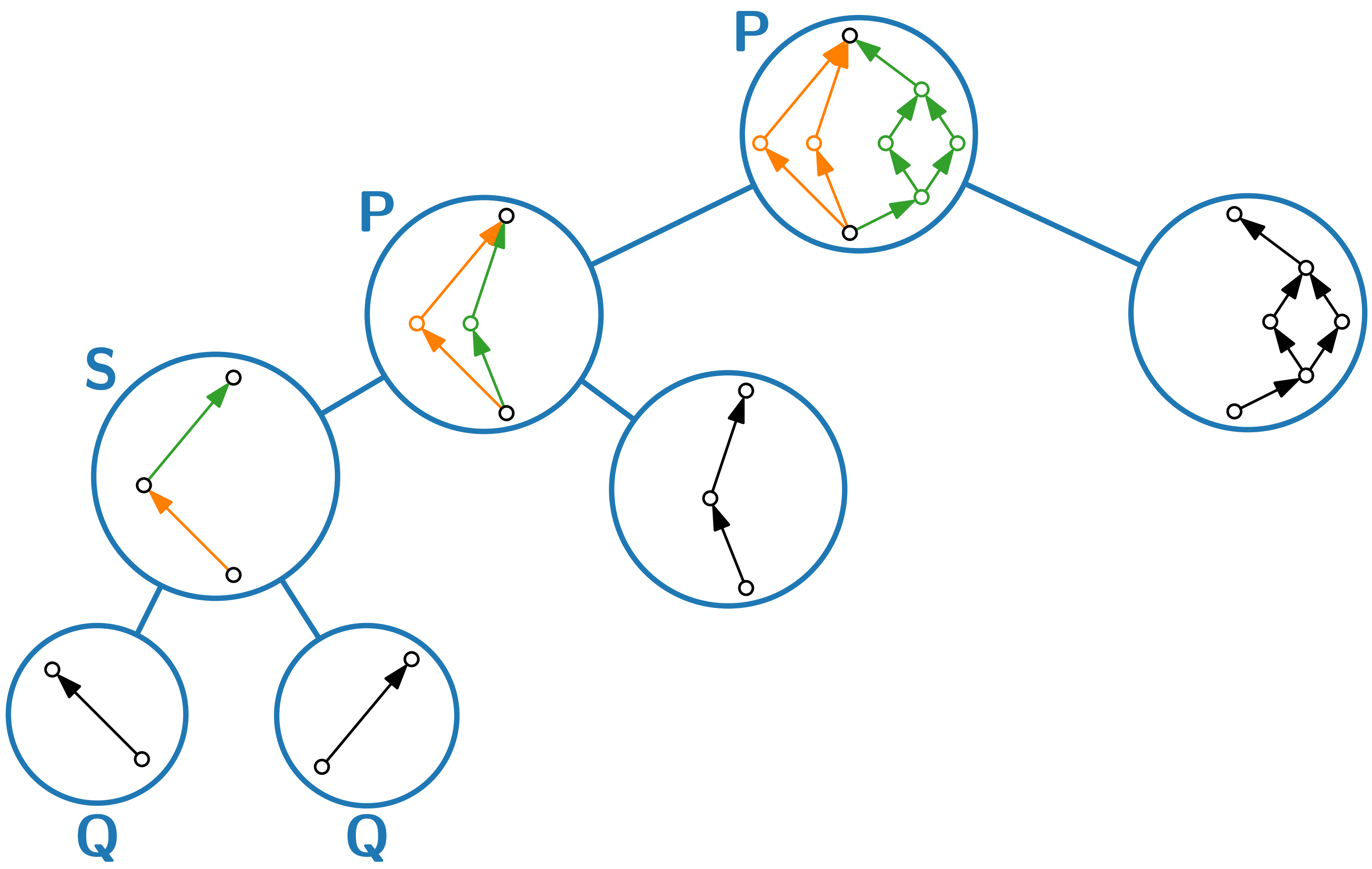
Series-Parallel Graphs – Decomposition Example



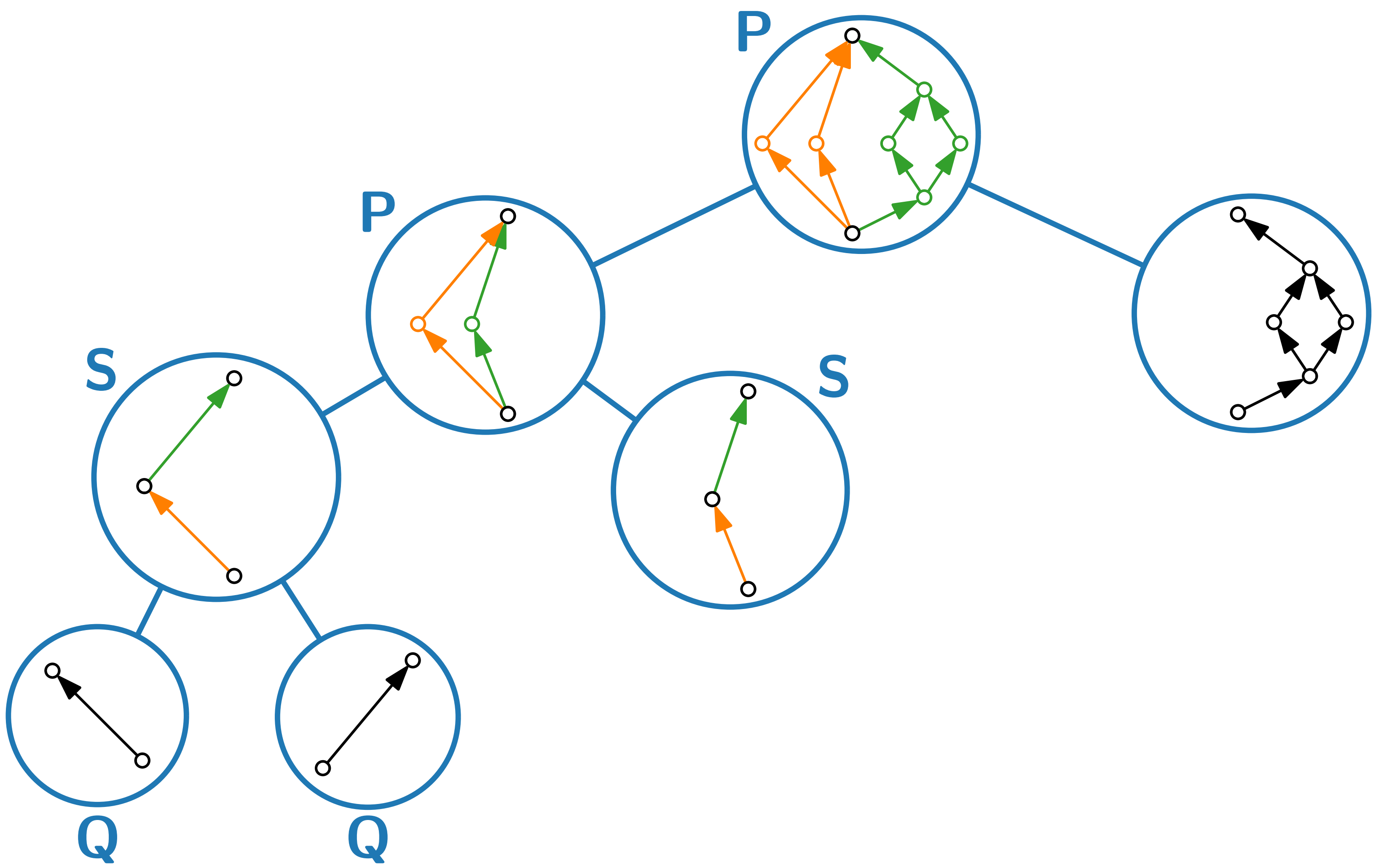
Series-Parallel Graphs – Decomposition Example



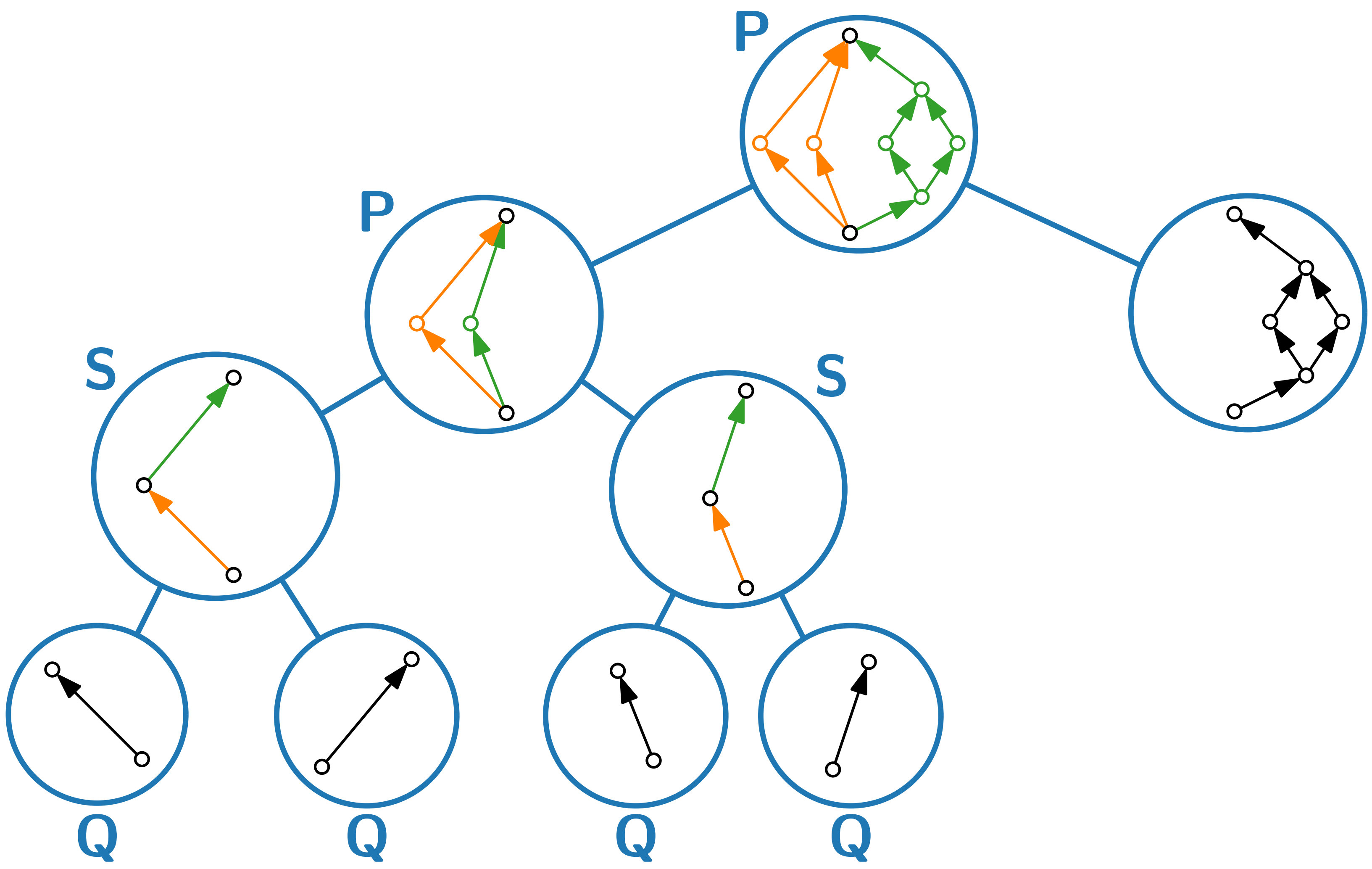
Series-Parallel Graphs – Decomposition Example



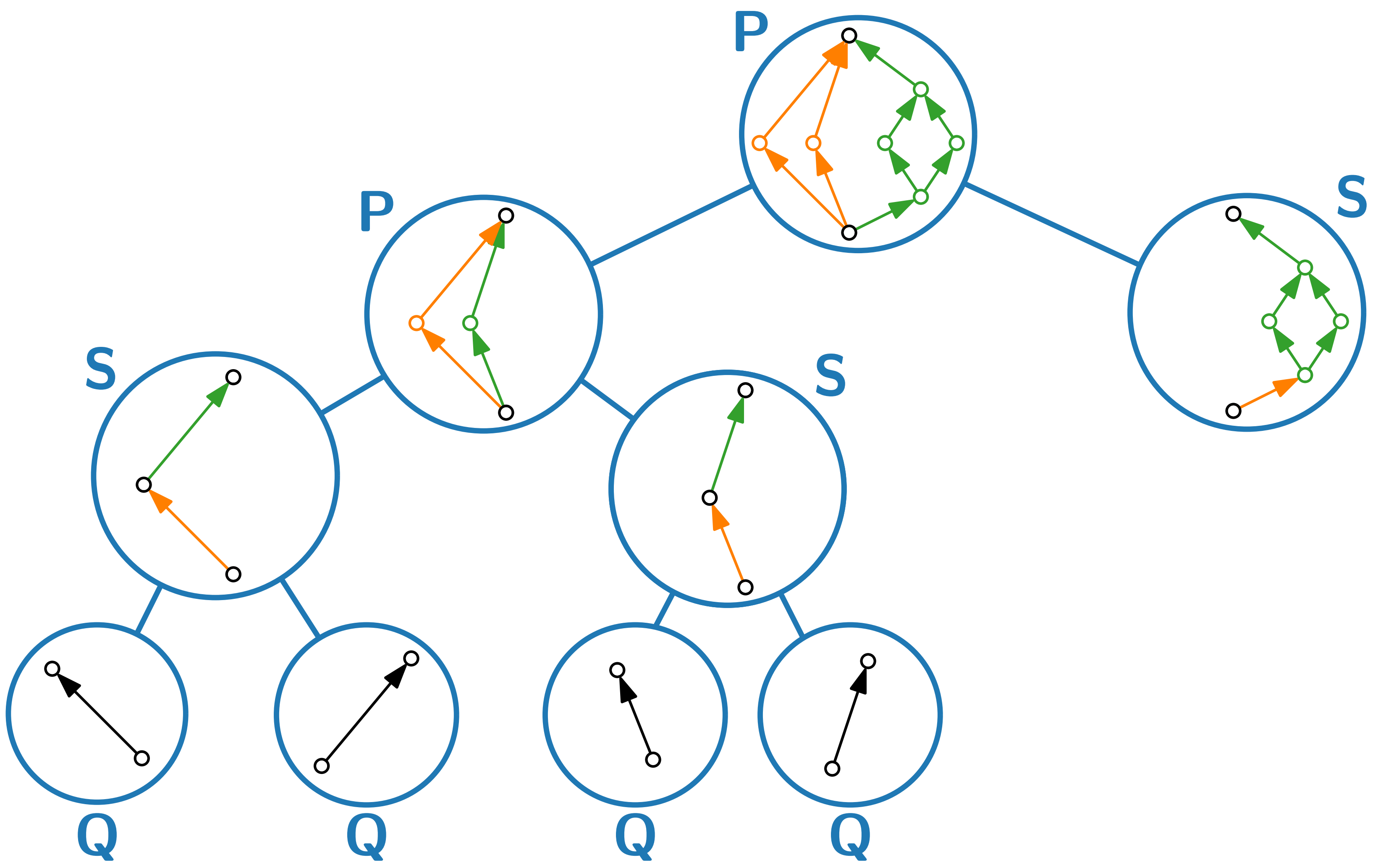
Series-Parallel Graphs – Decomposition Example



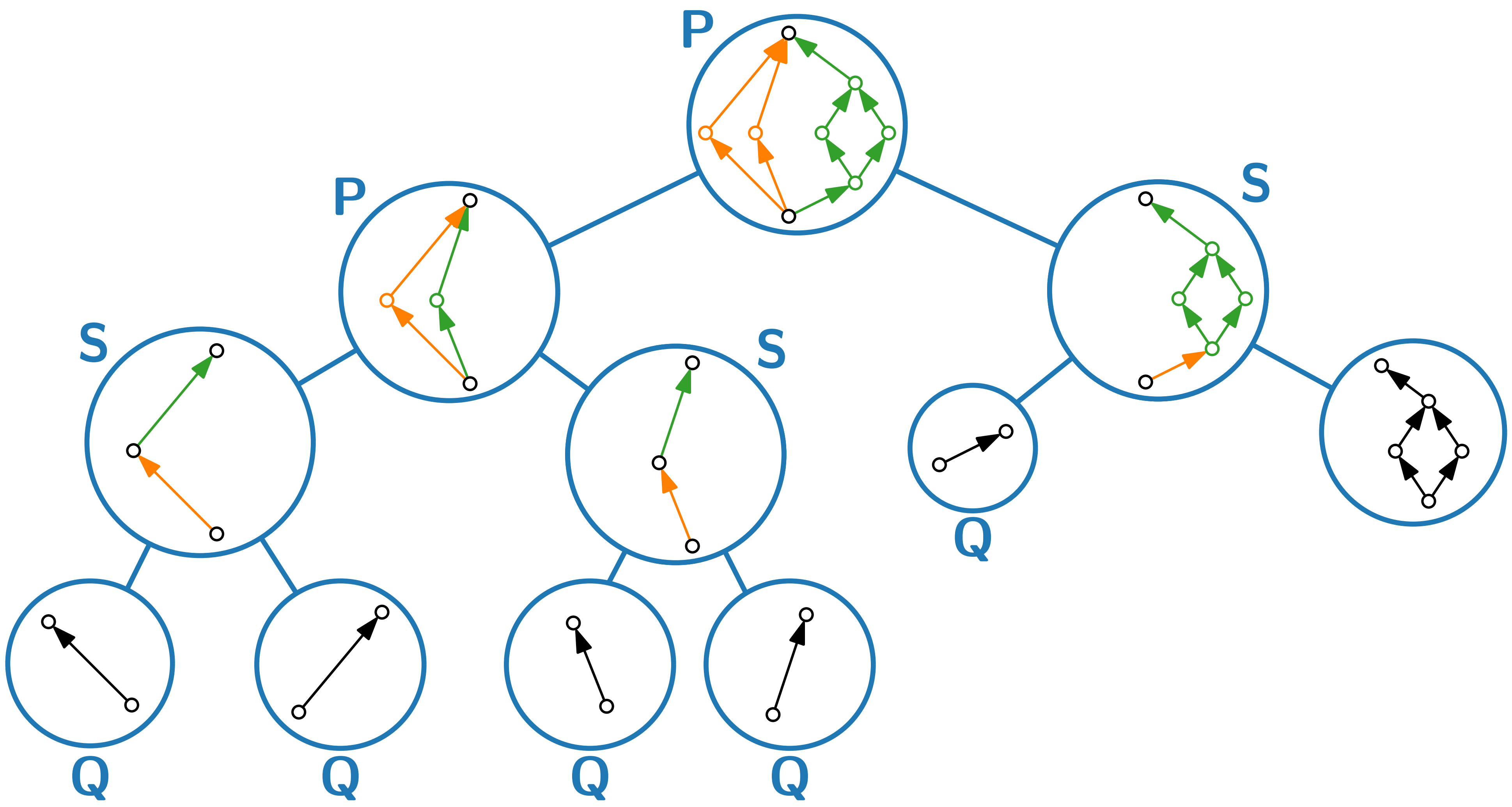
Series-Parallel Graphs – Decomposition Example



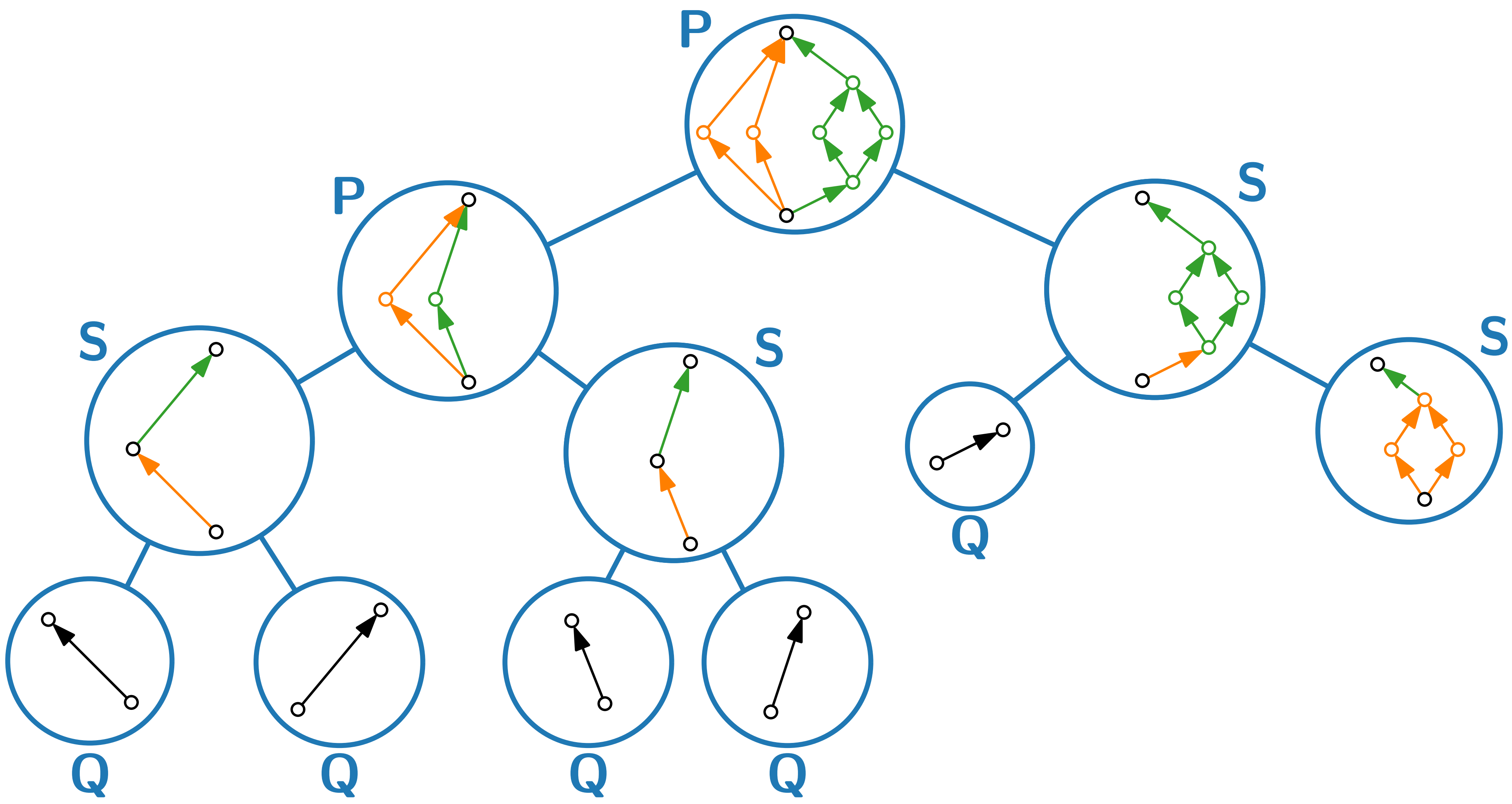
Series-Parallel Graphs – Decomposition Example



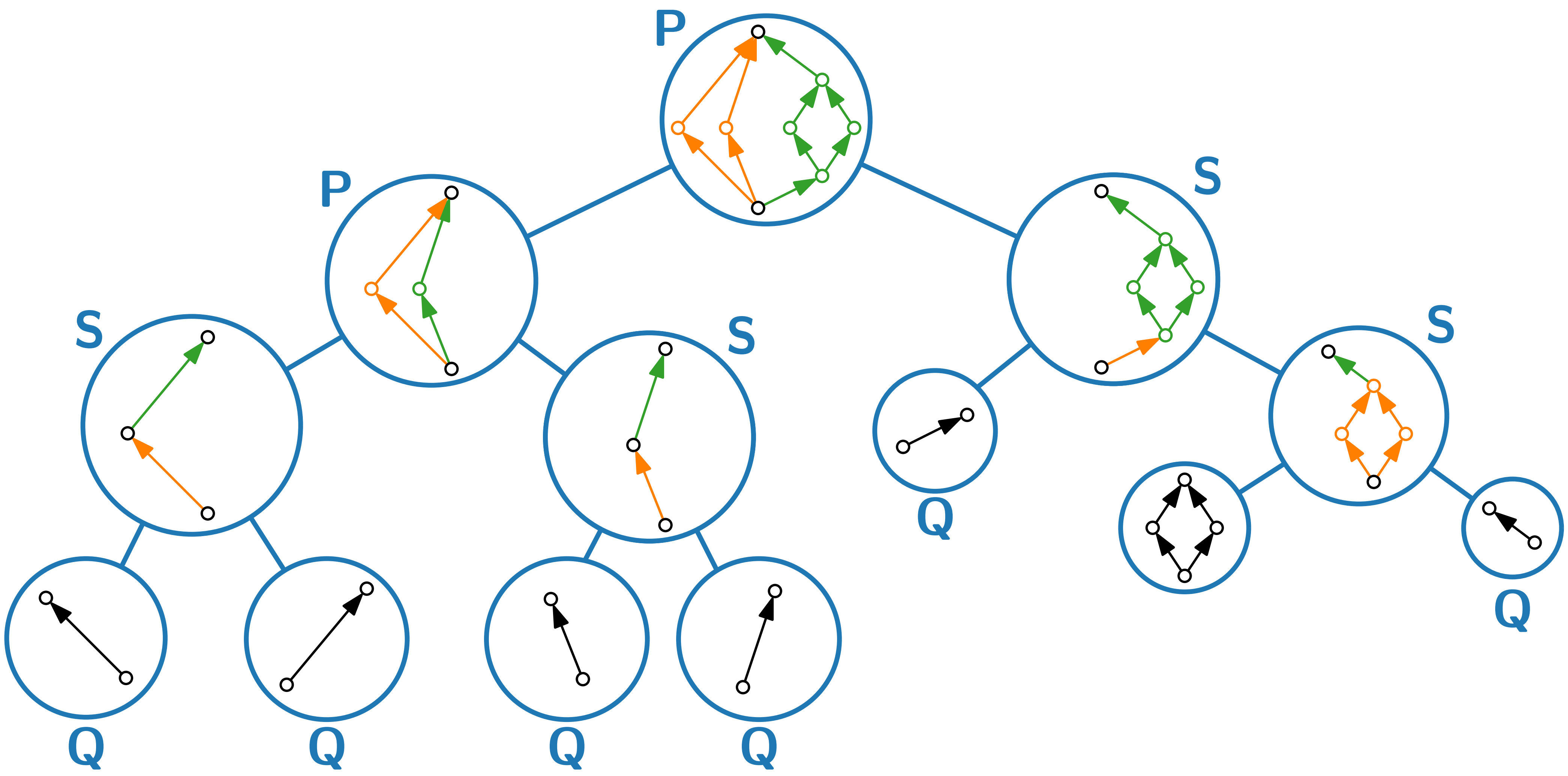
Series-Parallel Graphs – Decomposition Example



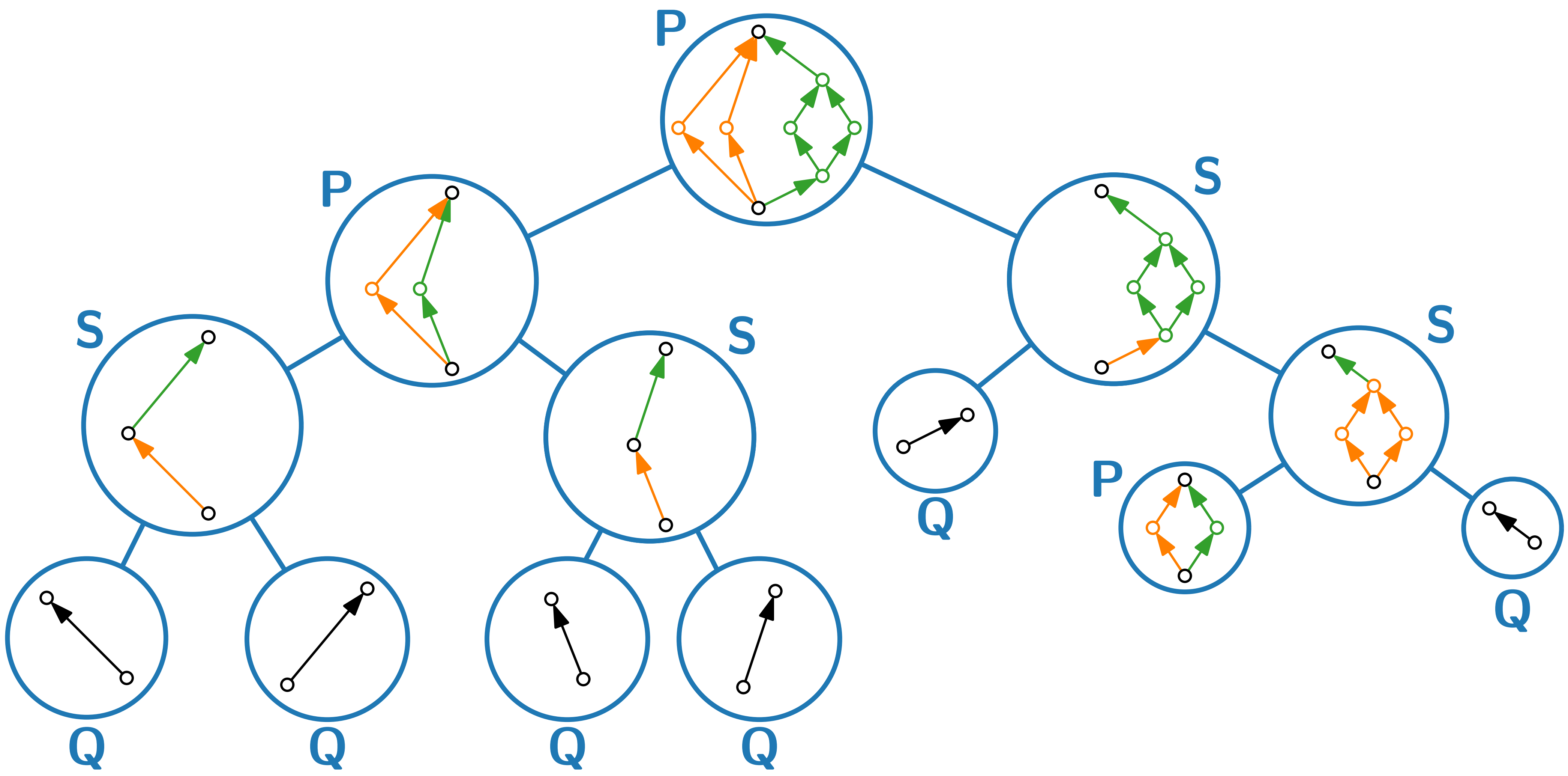
Series-Parallel Graphs – Decomposition Example



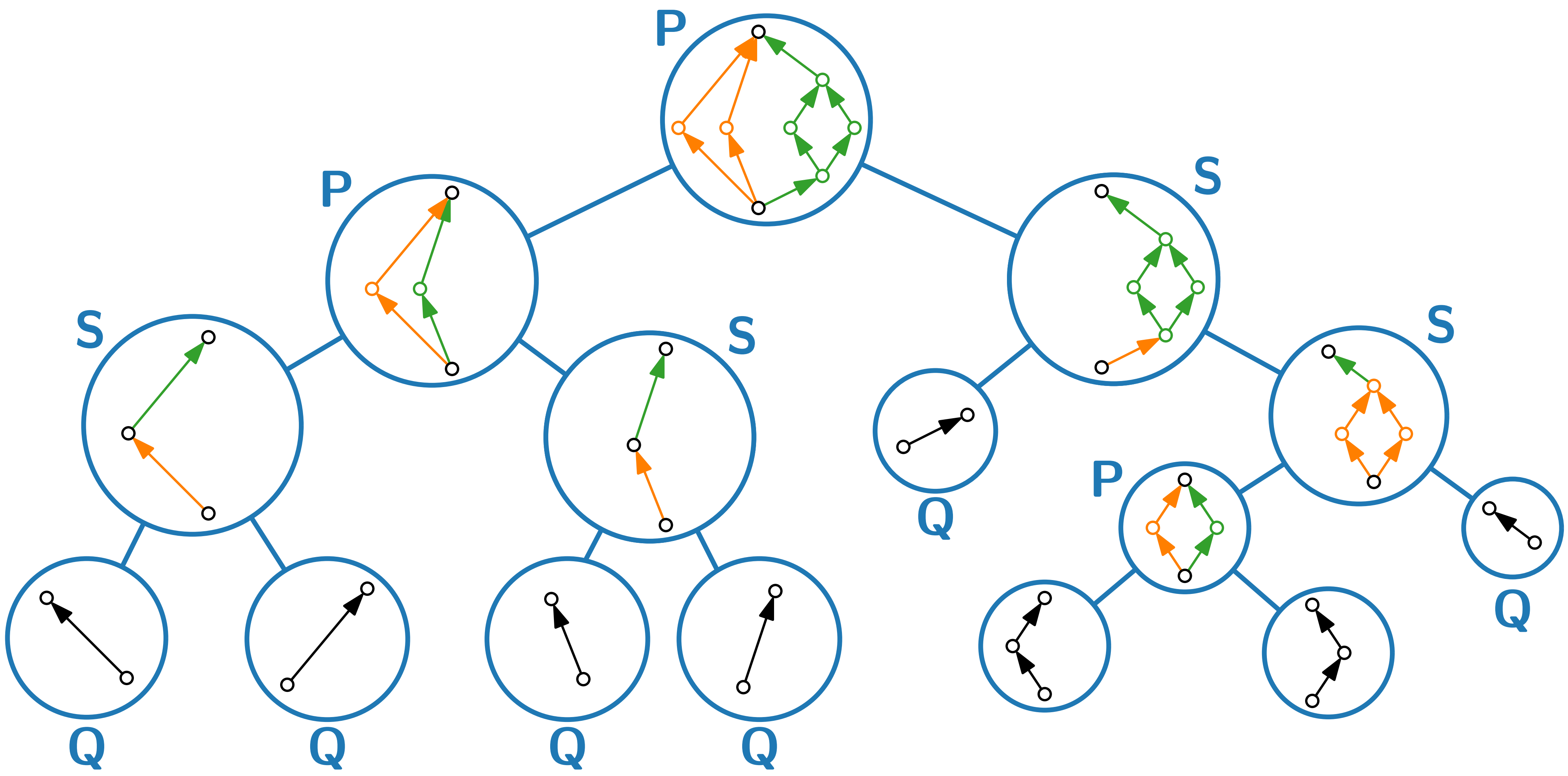
Series-Parallel Graphs – Decomposition Example



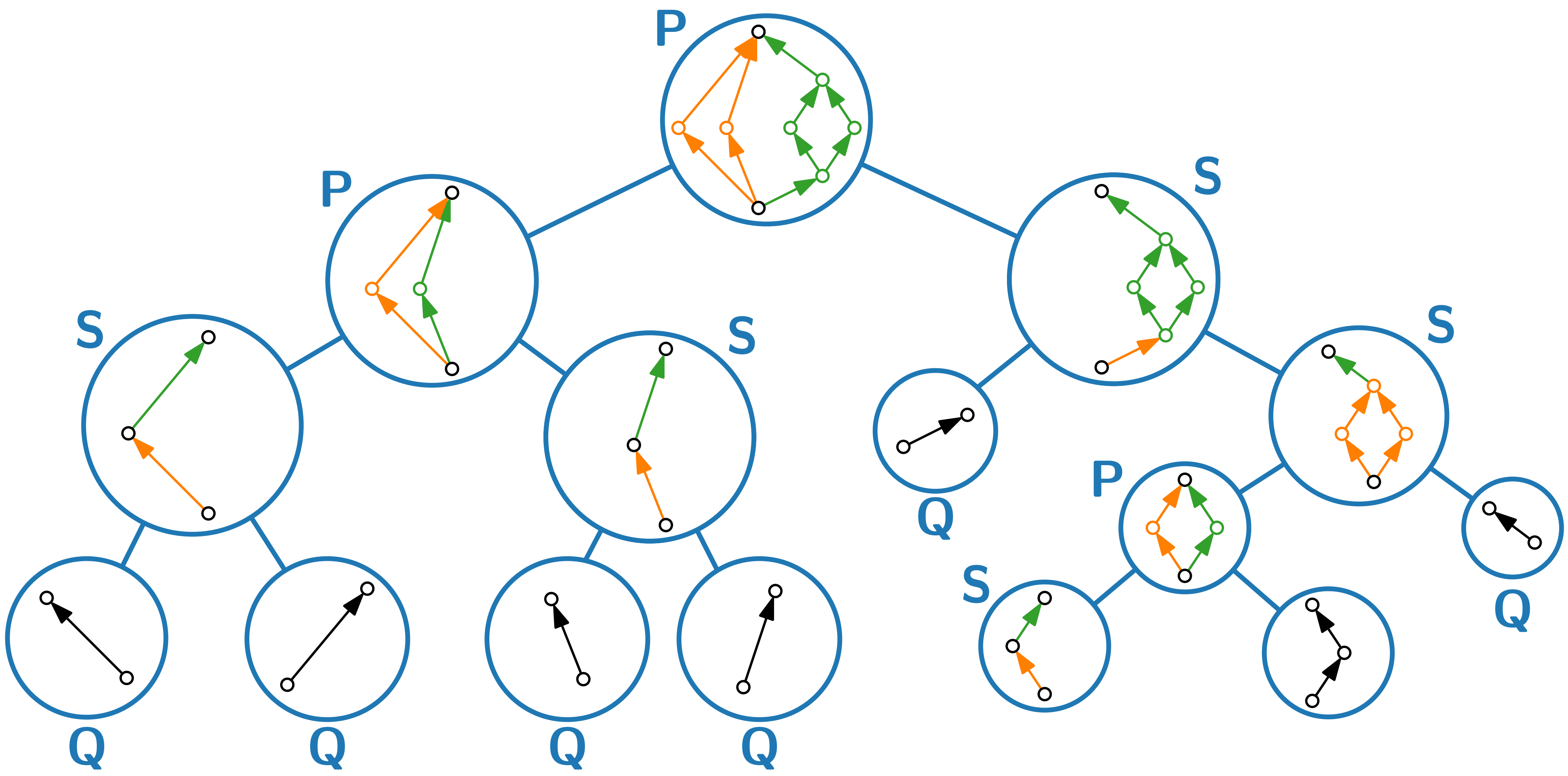
Series-Parallel Graphs – Decomposition Example



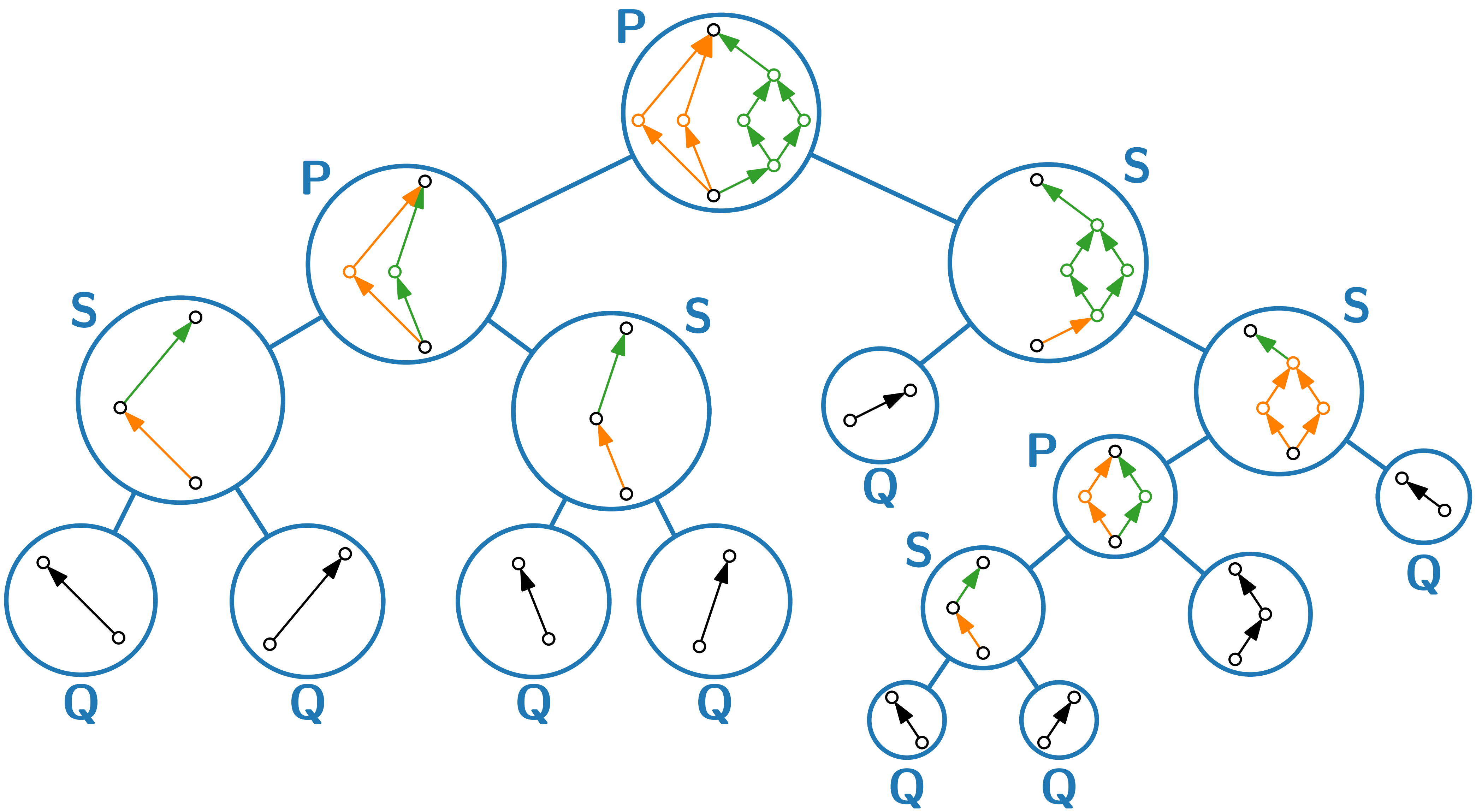
Series-Parallel Graphs – Decomposition Example



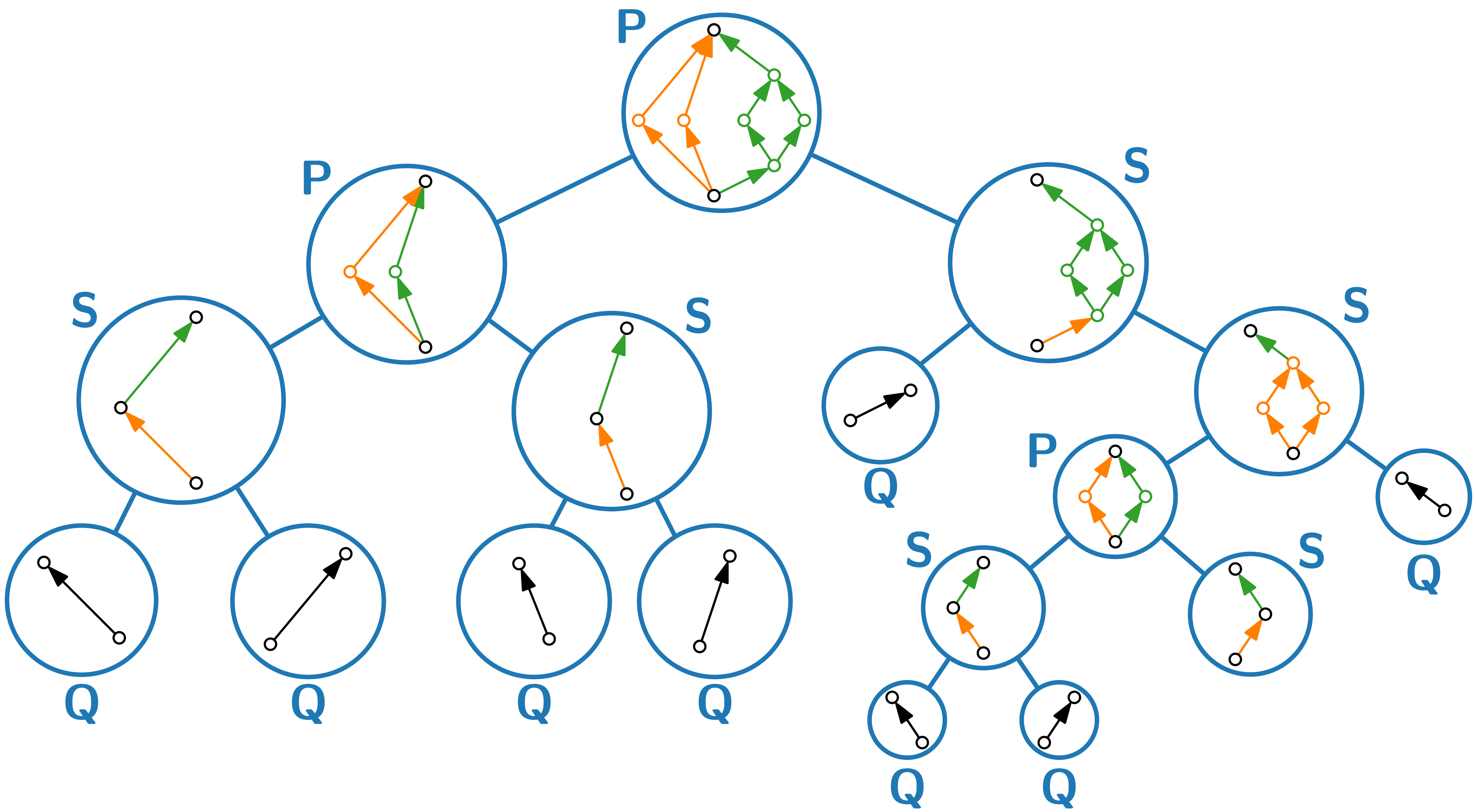
Series-Parallel Graphs – Decomposition Example



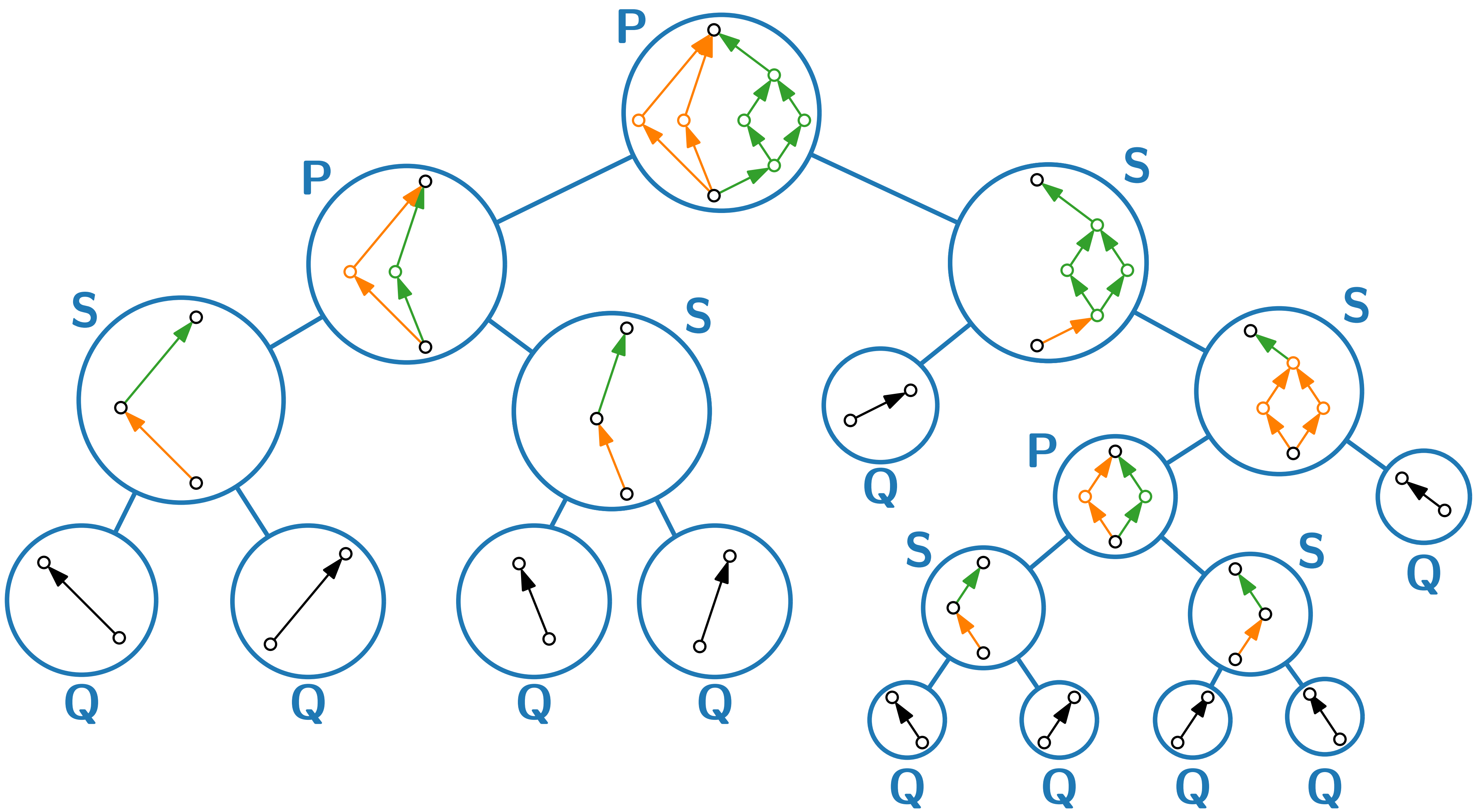
Series-Parallel Graphs – Decomposition Example



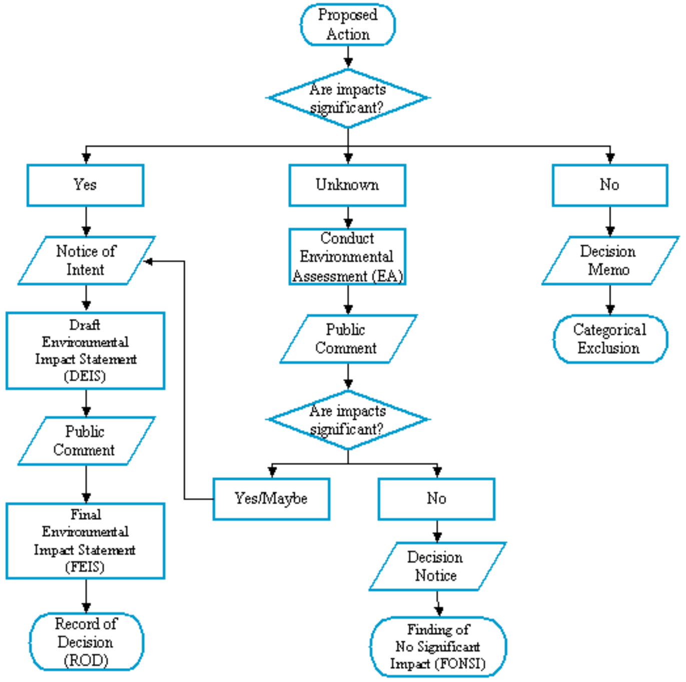
Series-Parallel Graphs – Decomposition Example



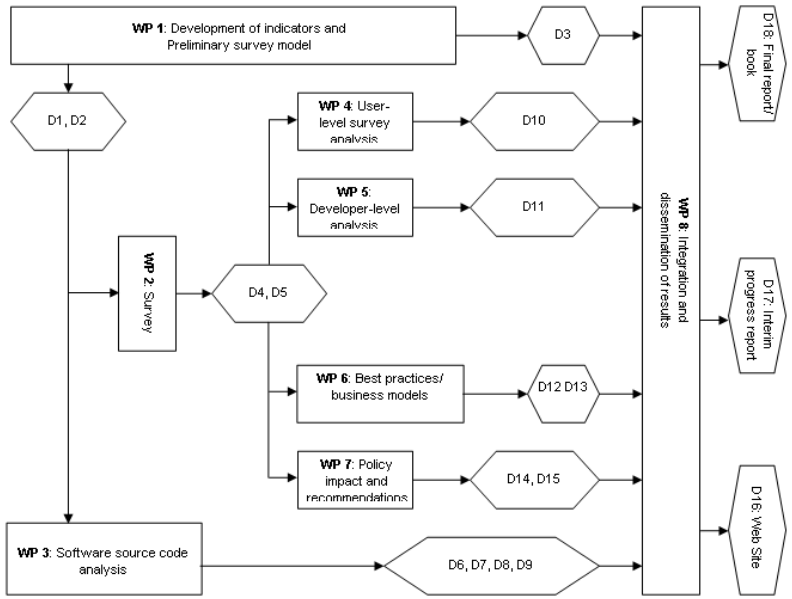
Series-Parallel Graphs – Decomposition Example



Series-Parallel Graphs – Applications



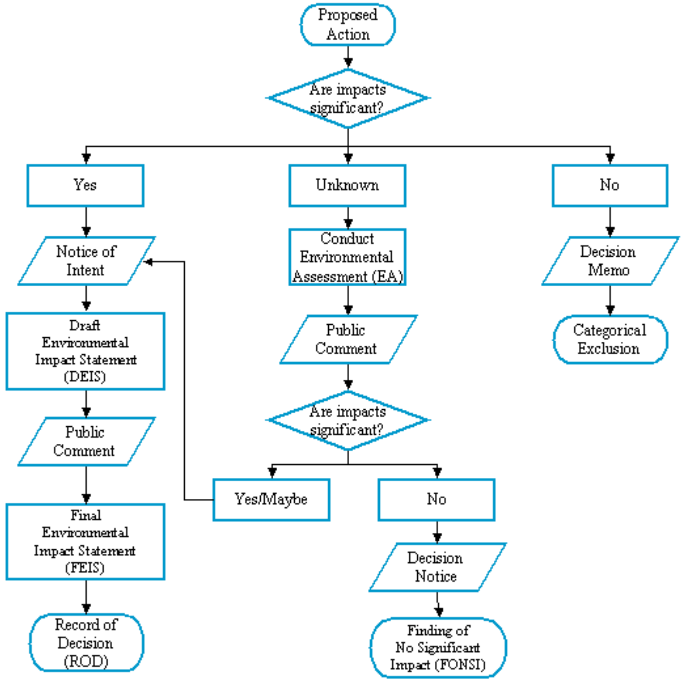
Flowcharts



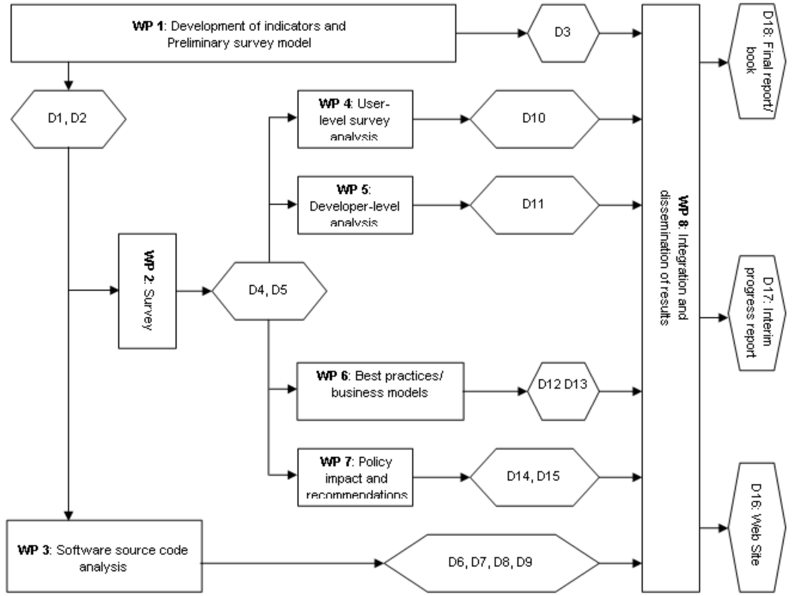
PERT-Diagrams

(Program Evaluation and Review Technique)

Series-Parallel Graphs – Applications



Flowcharts



PERT-Diagrams

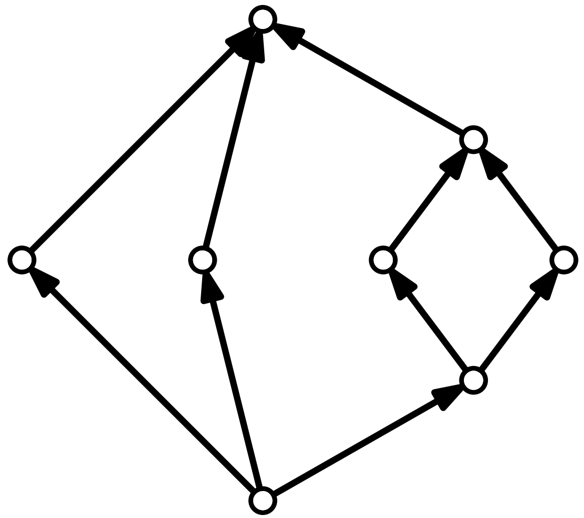
(Program Evaluation and Review Technique)

Computational complexity:

Series-parallel graphs often admit linear-time algorithms for NP-hard problems, e.g., minimum maximal matching, maximum independent set, Hamiltonian completion.

Series-Parallel Graphs – Drawing Style

Drawing conventions

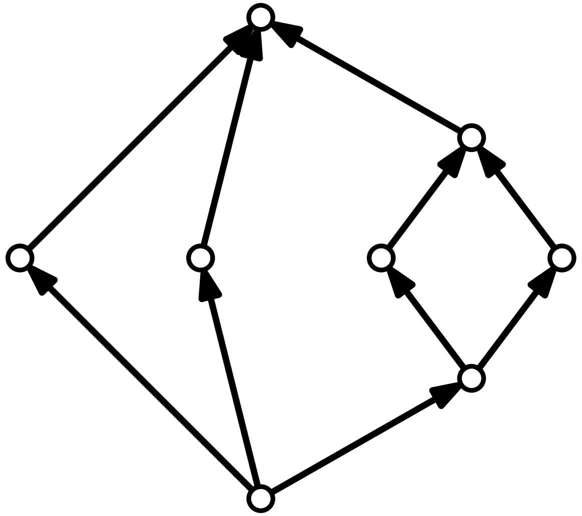


Drawing aesthetics to optimize

Series-Parallel Graphs – Drawing Style

Drawing conventions

- Planarity

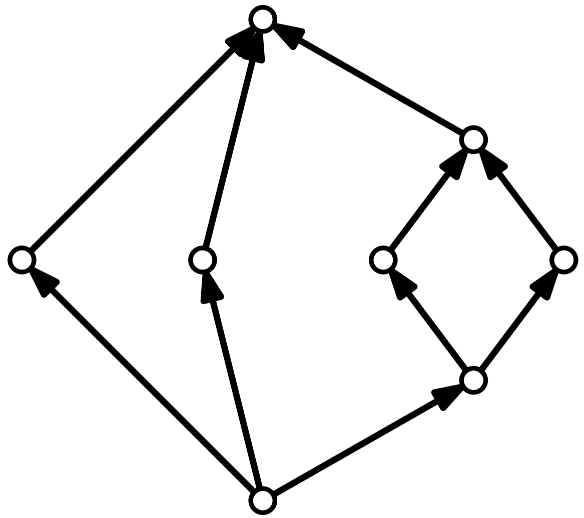


Drawing aesthetics to optimize

Series-Parallel Graphs – Drawing Style

Drawing conventions

- Planarity
- Straight-line edges



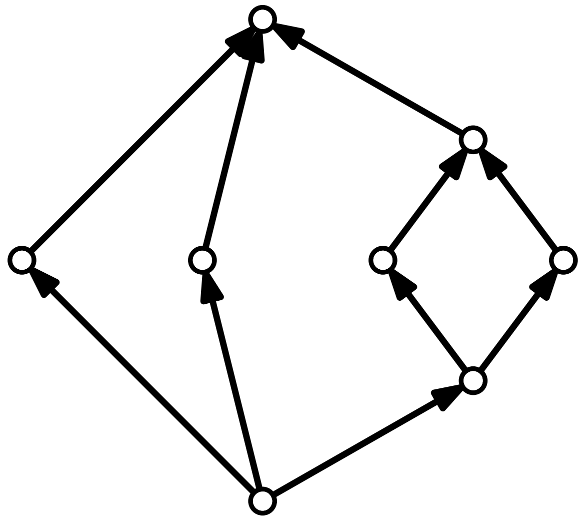
Drawing aesthetics to optimize

Series-Parallel Graphs – Drawing Style

Drawing conventions

- Planarity
- Straight-line edges
- Upward

Drawing aesthetics to optimize



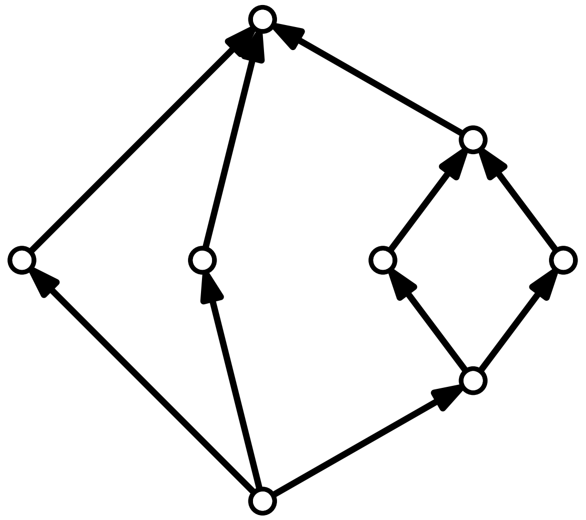
Series-Parallel Graphs – Drawing Style

Drawing conventions

- Planarity
- Straight-line edges
- Upward

Drawing aesthetics to optimize

- Area



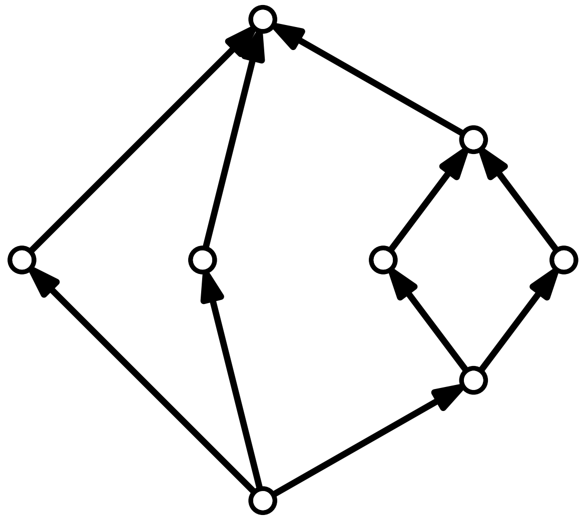
Series-Parallel Graphs – Drawing Style

Drawing conventions

- Planarity
- Straight-line edges
- Upward

Drawing aesthetics to optimize

- Area
- Symmetry



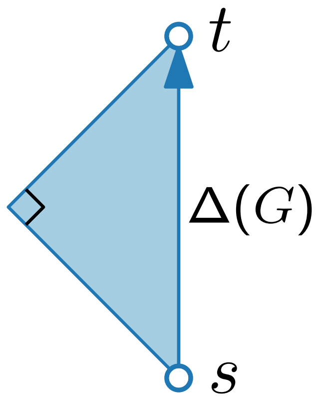
Series-Parallel Graphs – Straight-Line Drawings

Divide & conquer algorithm using the decomposition tree

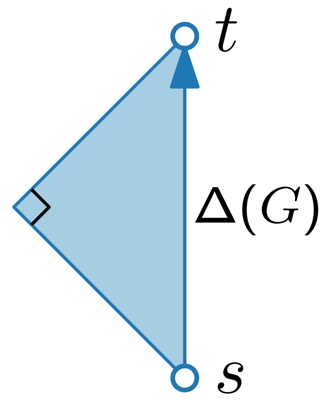
Series-Parallel Graphs – Straight-Line Drawings

Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top



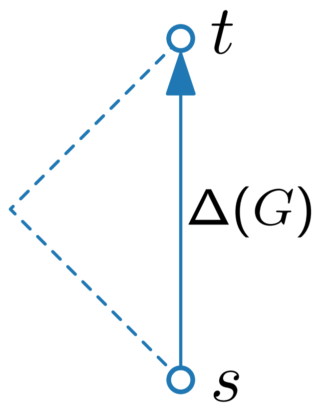
Series-Parallel Graphs – Straight-Line Drawings



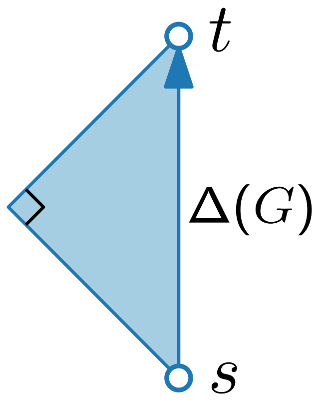
Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top

Base case: Q-nodes



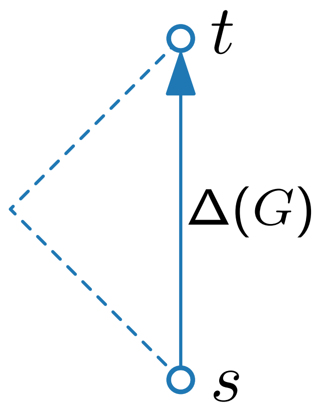
Series-Parallel Graphs – Straight-Line Drawings



Divide & conquer algorithm using the decomposition tree

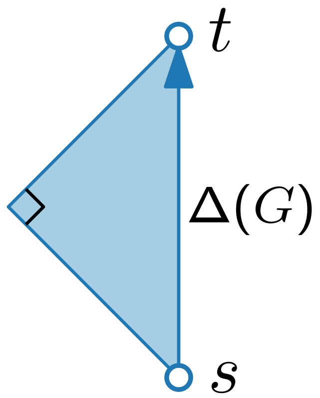
- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top

Base case: Q-nodes



Divide: Draw G_1 and G_2 first

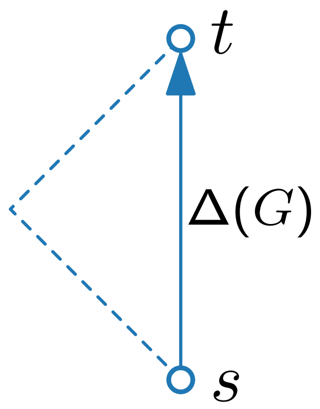
Series-Parallel Graphs – Straight-Line Drawings



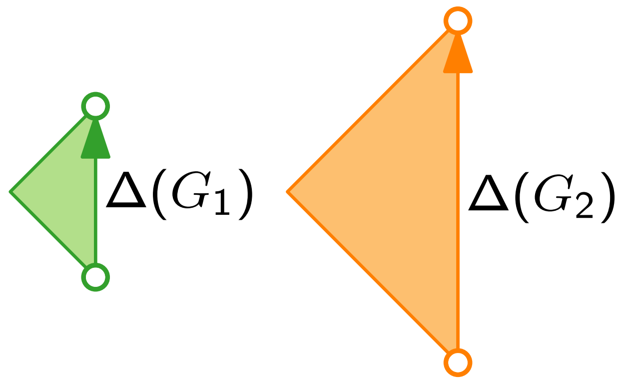
Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top

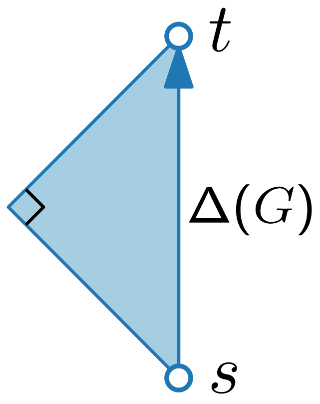
Base case: Q-nodes



Divide: Draw G_1 and G_2 first



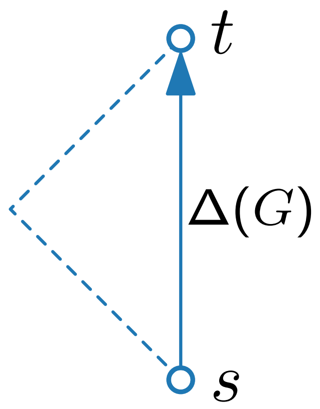
Series-Parallel Graphs – Straight-Line Drawings



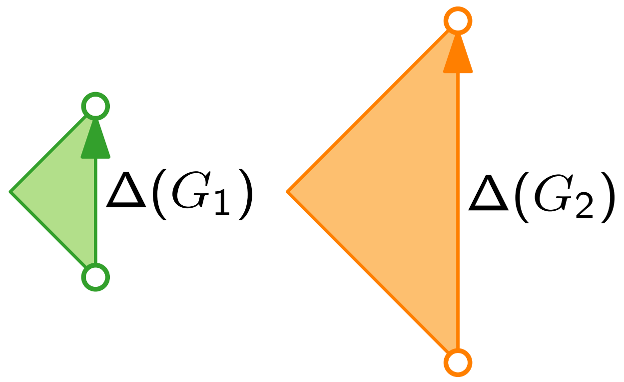
Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top

Base case: Q-nodes



Divide: Draw G_1 and G_2 first

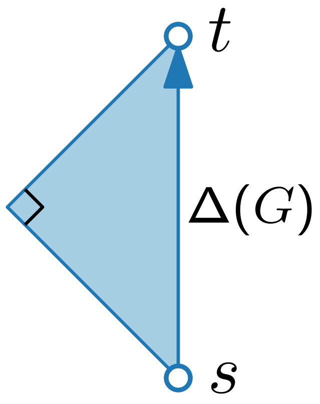


Conquer:

Series-Parallel Graphs – Straight-Line Drawings

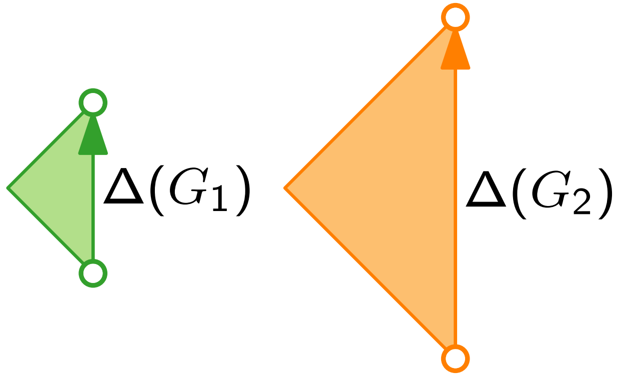
Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top



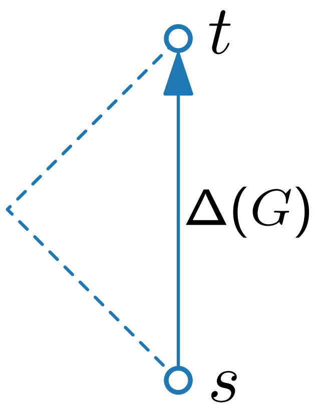
Base case: Q-nodes

Divide: Draw G_1 and G_2 first



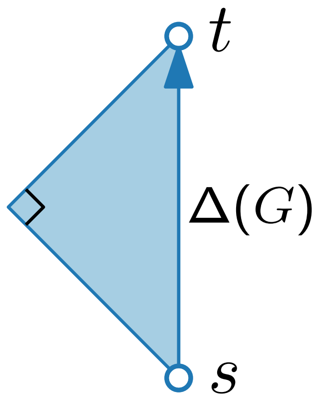
Conquer:

- S-nodes: series compositions



Series-Parallel Graphs – Straight-Line Drawings

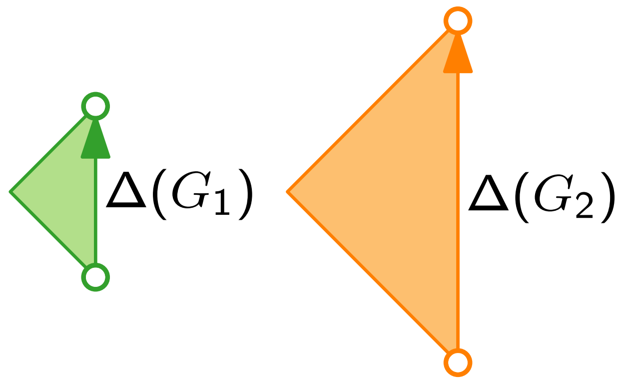
Divide & conquer algorithm using the decomposition tree



- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top

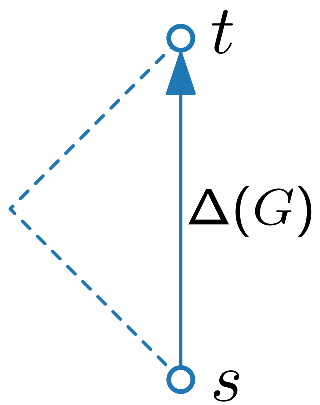
Base case: Q-nodes

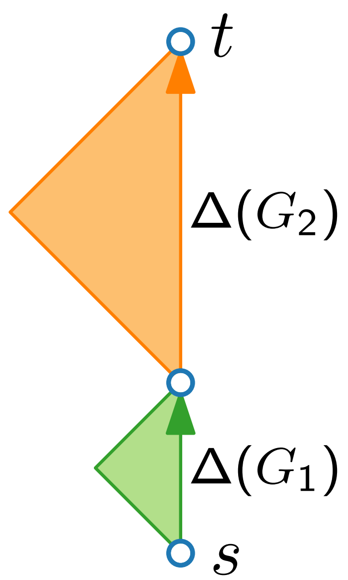
Divide: Draw G_1 and G_2 first



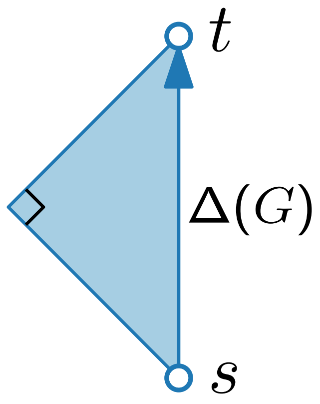
Conquer:

- S-nodes: series compositions





Series-Parallel Graphs – Straight-Line Drawings

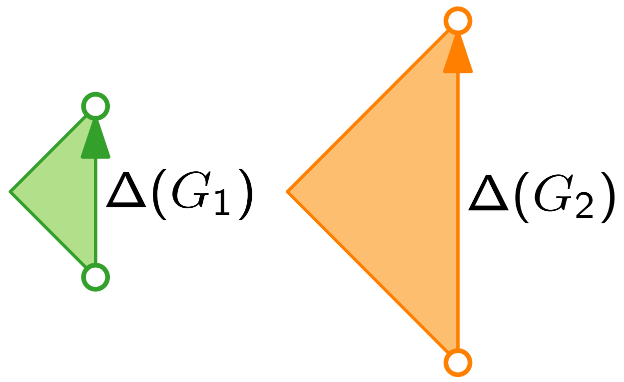


Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top

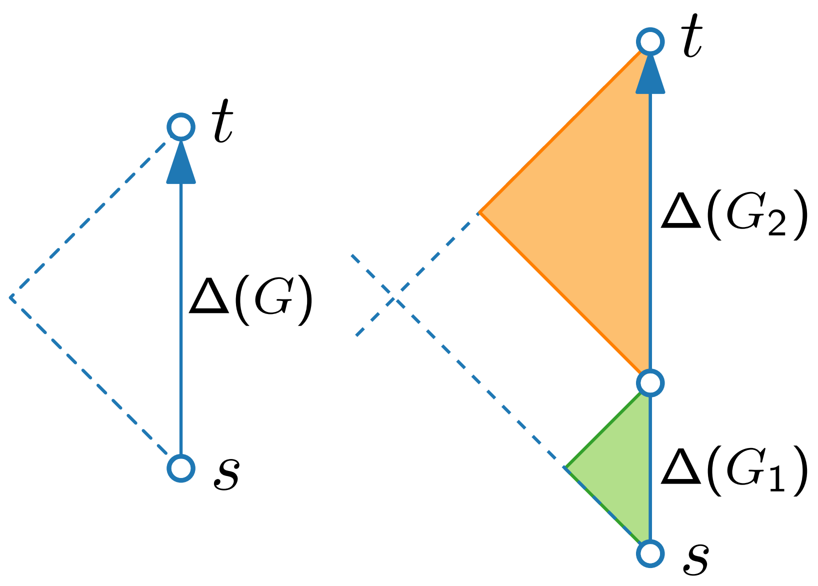
Base case: Q-nodes

Divide: Draw G_1 and G_2 first



Conquer:

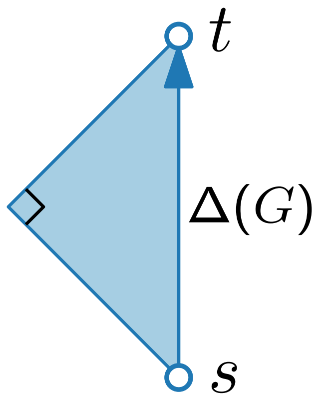
- S-nodes: series compositions



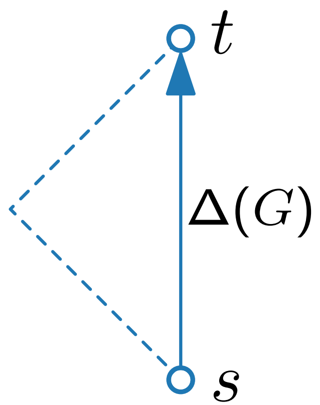
Series-Parallel Graphs – Straight-Line Drawings

Divide & conquer algorithm using the decomposition tree

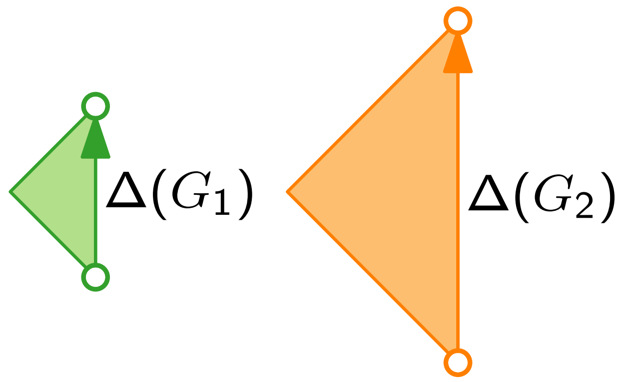
- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top



Base case: Q-nodes

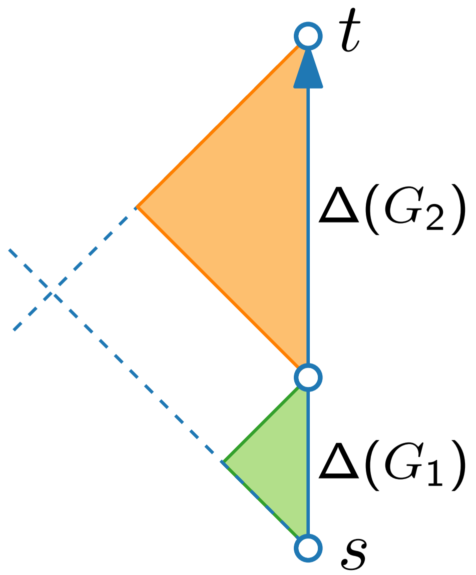


Divide: Draw G_1 and G_2 first



Conquer:

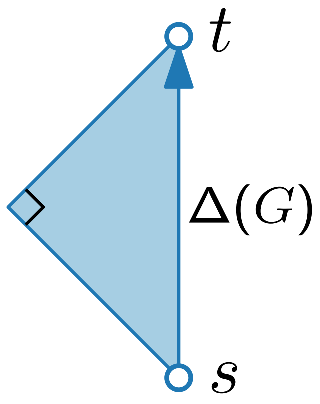
- S-nodes: series compositions
- P-nodes: parallel compositions



Series-Parallel Graphs – Straight-Line Drawings

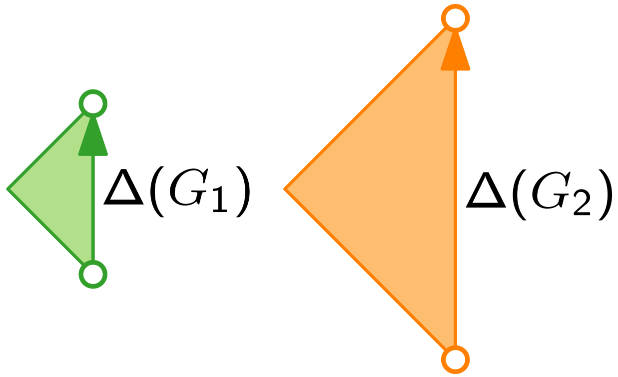
Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top



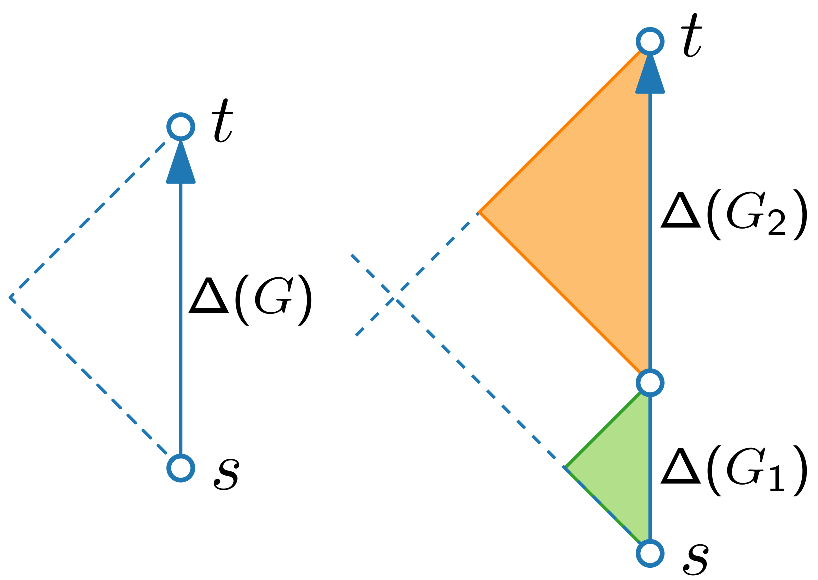
Base case: Q-nodes

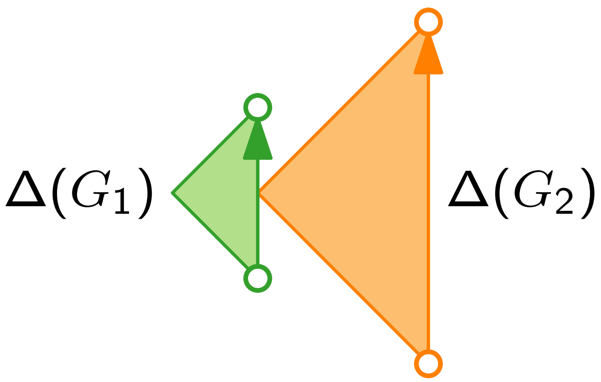
Divide: Draw G_1 and G_2 first



Conquer:

- S-nodes: series compositions
- P-nodes: parallel compositions

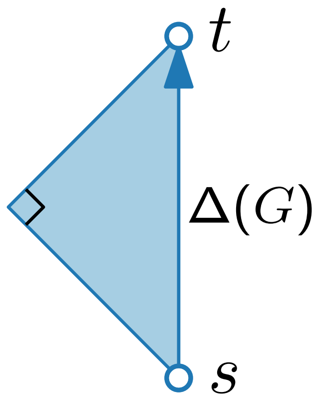




Series-Parallel Graphs – Straight-Line Drawings

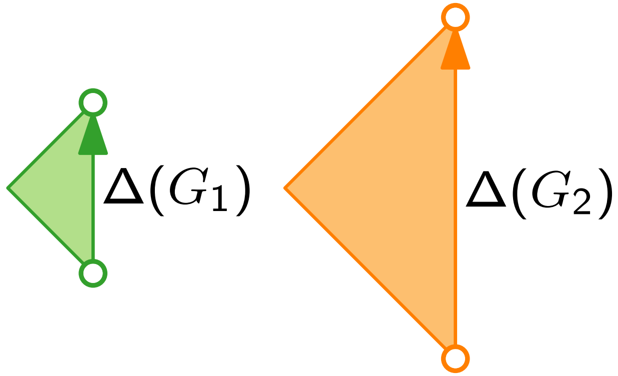
Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top



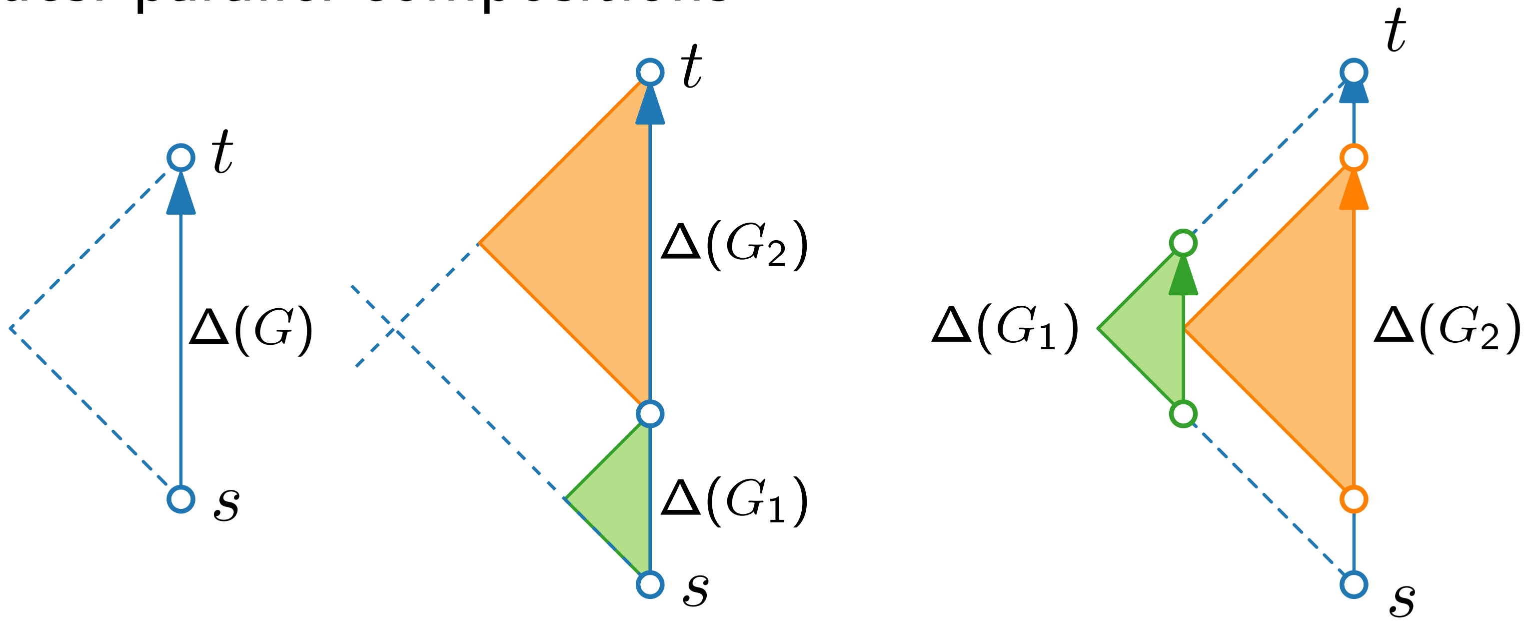
Base case: Q-nodes

Divide: Draw G_1 and G_2 first



Conquer:

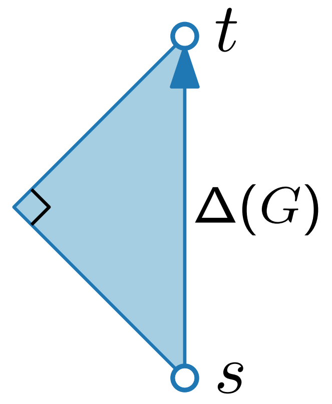
- S-nodes: series compositions
- P-nodes: parallel compositions



Series-Parallel Graphs – Straight-Line Drawings

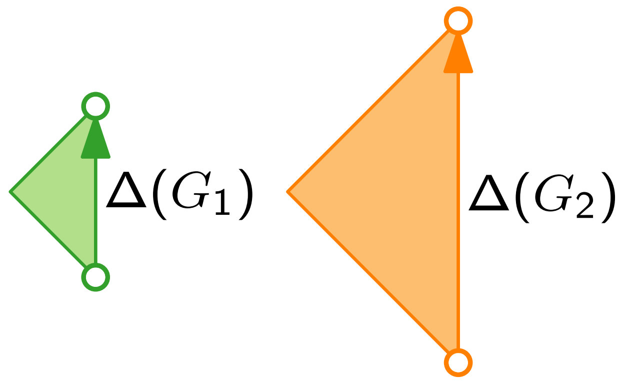
Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top



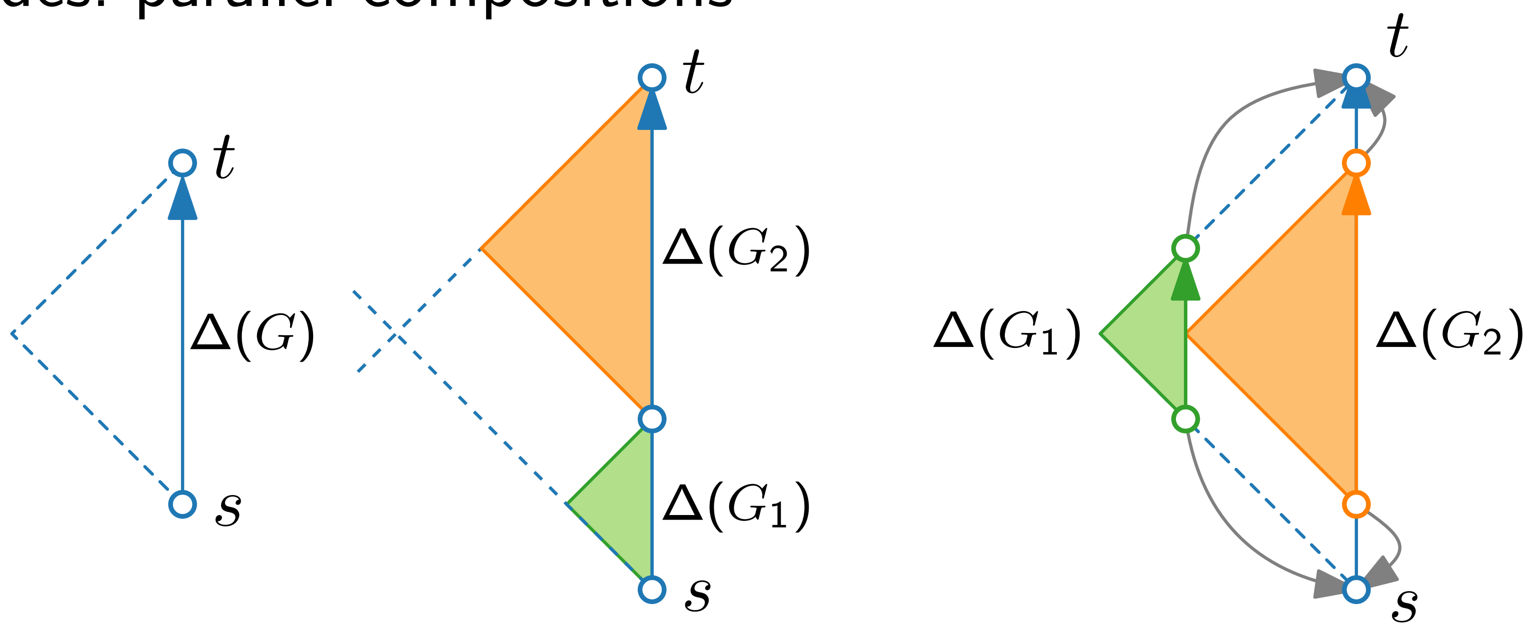
Base case: Q-nodes

Divide: Draw G_1 and G_2 first

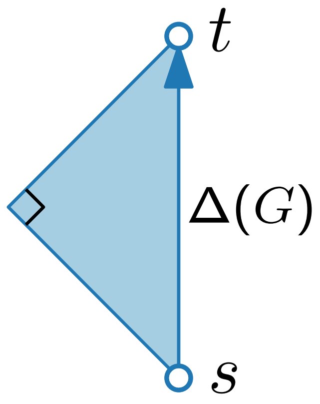


Conquer:

- S-nodes: series compositions
- P-nodes: parallel compositions



Series-Parallel Graphs – Straight-Line Drawings

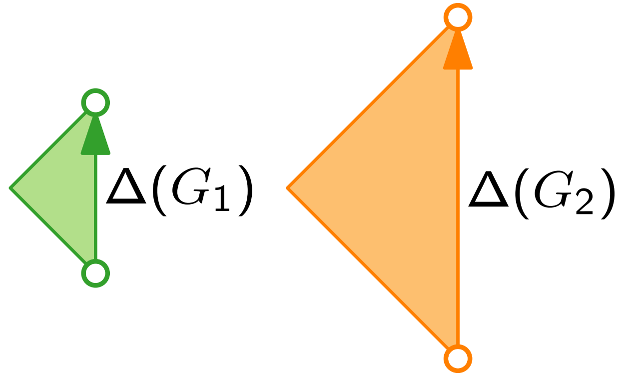


Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top

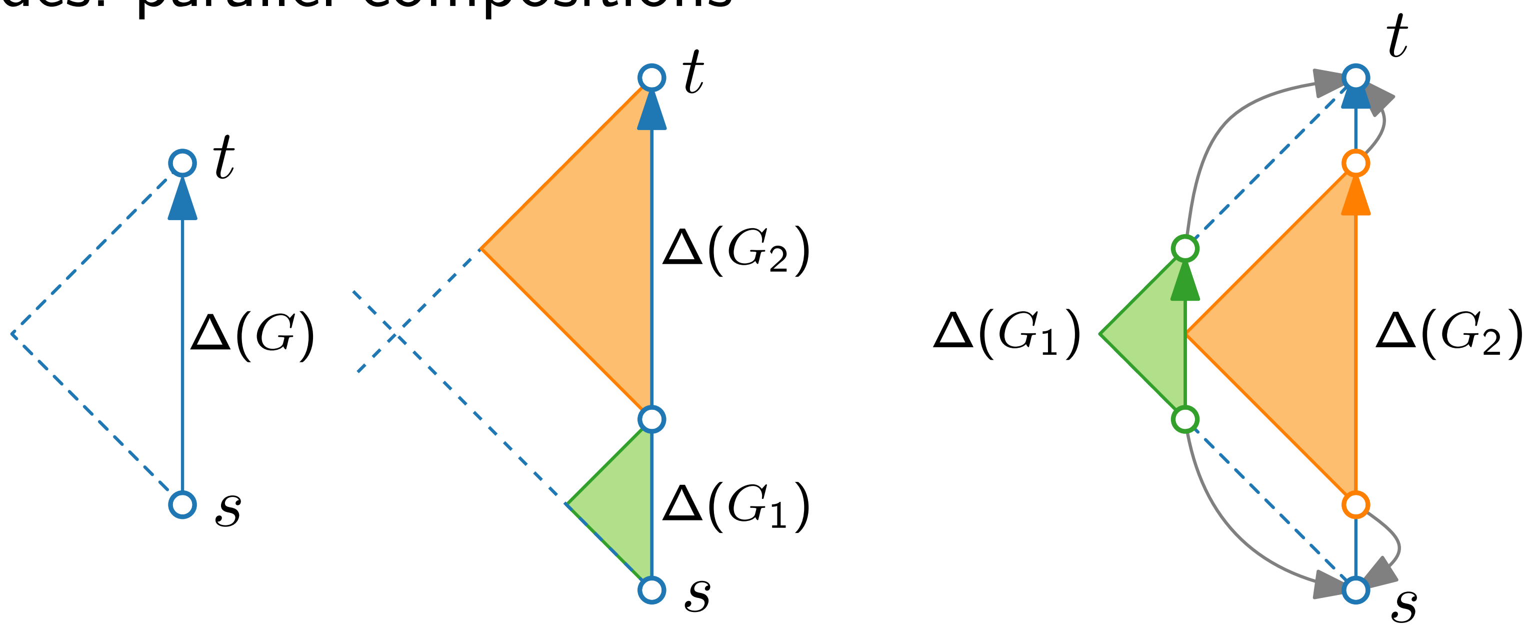
Base case: Q-nodes

Divide: Draw G_1 and G_2 first



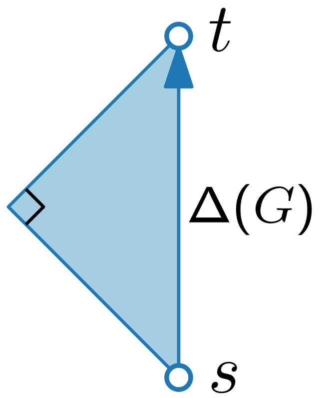
Conquer:

- S-nodes: series compositions
- P-nodes: parallel compositions



Do you see any problem?

Series-Parallel Graphs – Straight-Line Drawings

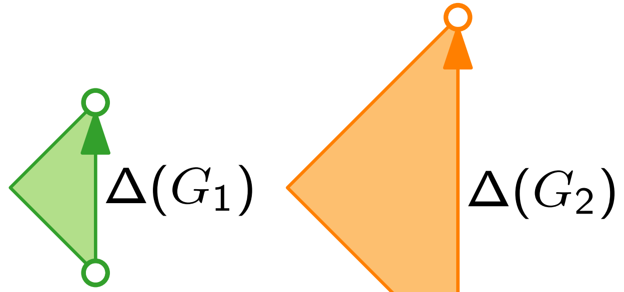


Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top

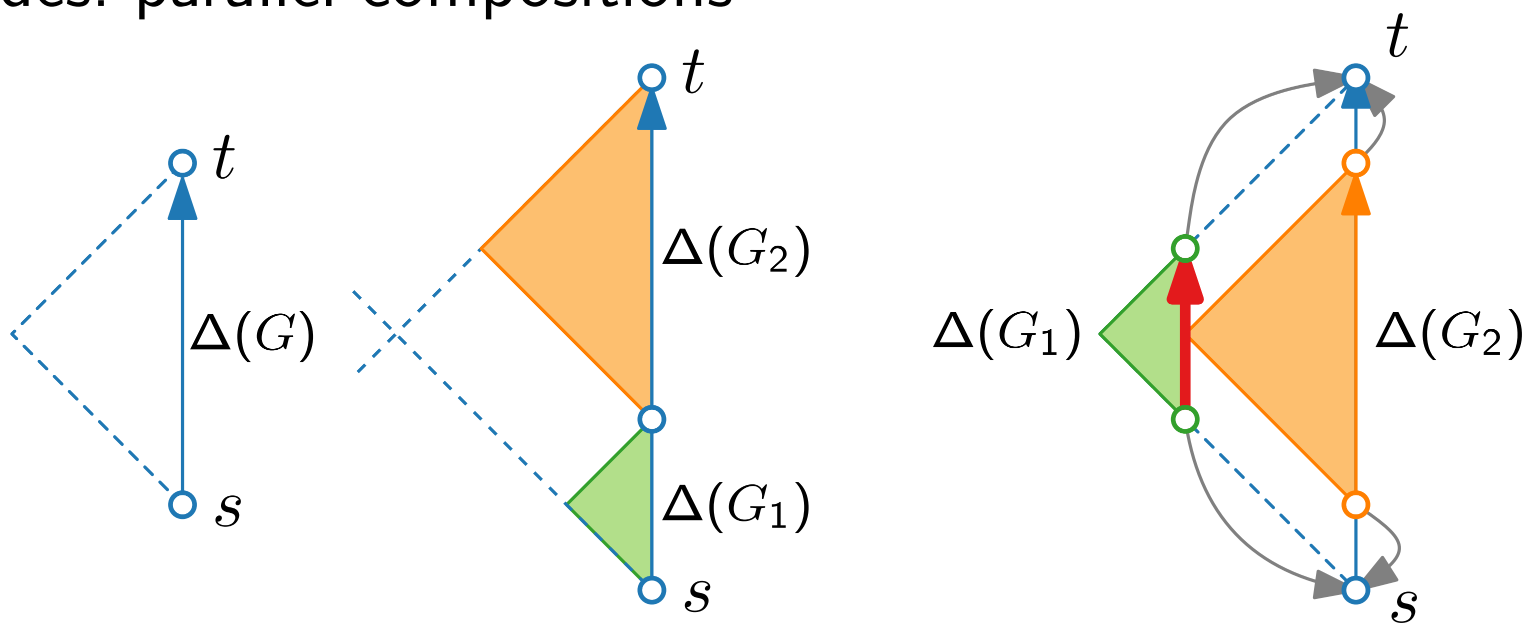
Base case: Q-nodes

Divide: Draw G_1 and G_2 first



Conquer:

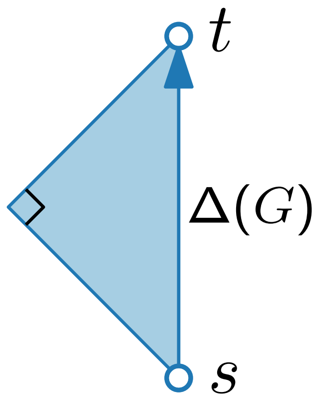
- S-nodes: series compositions
- P-nodes: parallel compositions



Do you see any problem?

single edge

Series-Parallel Graphs – Straight-Line Drawings

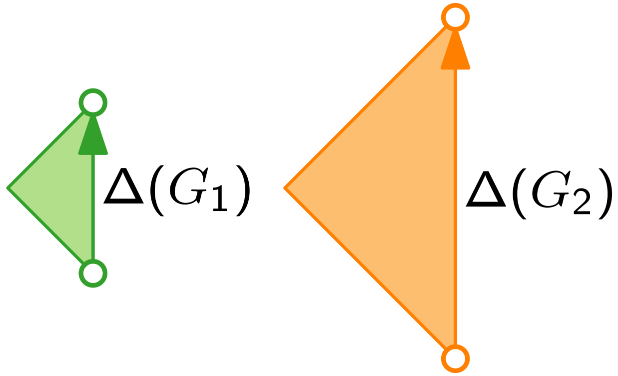


Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top

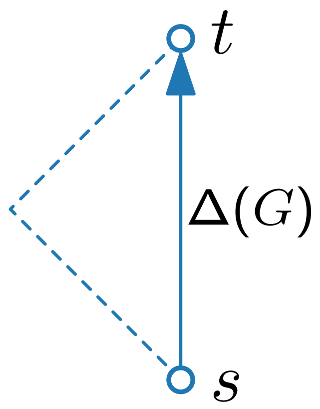
Base case: Q-nodes

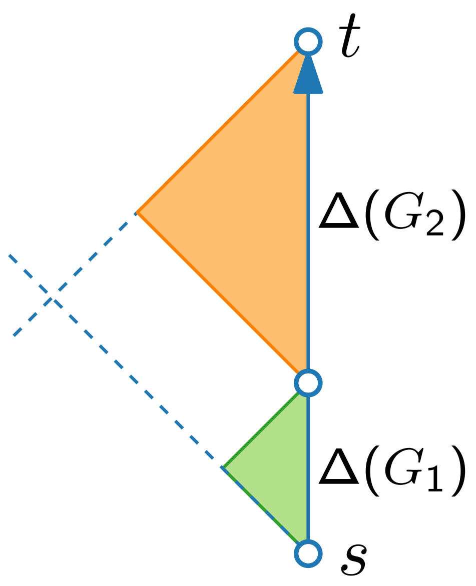
Divide: Draw G_1 and G_2 first

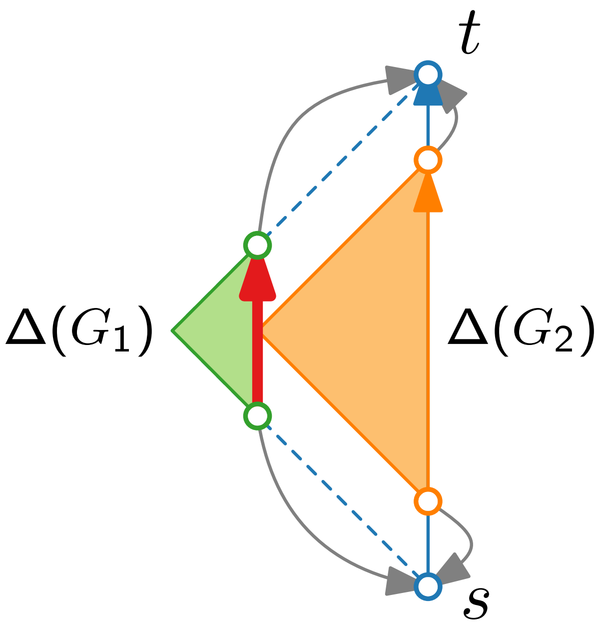


Conquer:

- S-nodes: series compositions
- P-nodes: parallel compositions





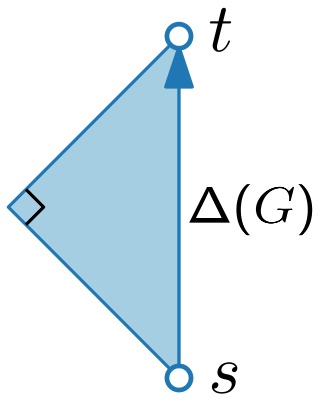


Do you see any problem?

single edge

change embedding!

Series-Parallel Graphs – Straight-Line Drawings

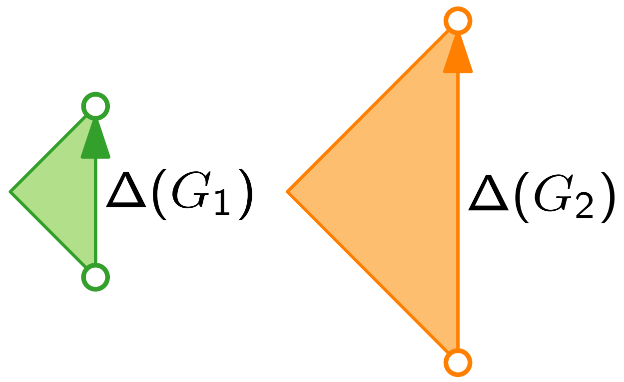


Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top

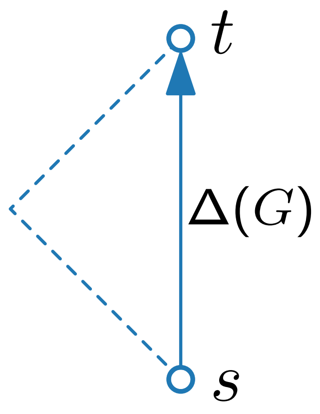
Base case: Q-nodes

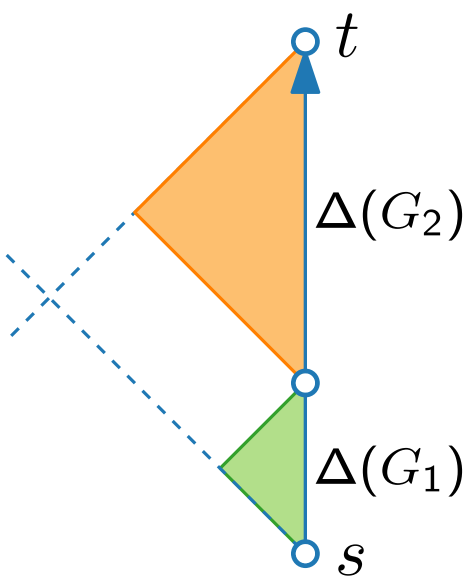
Divide: Draw G_1 and G_2 first

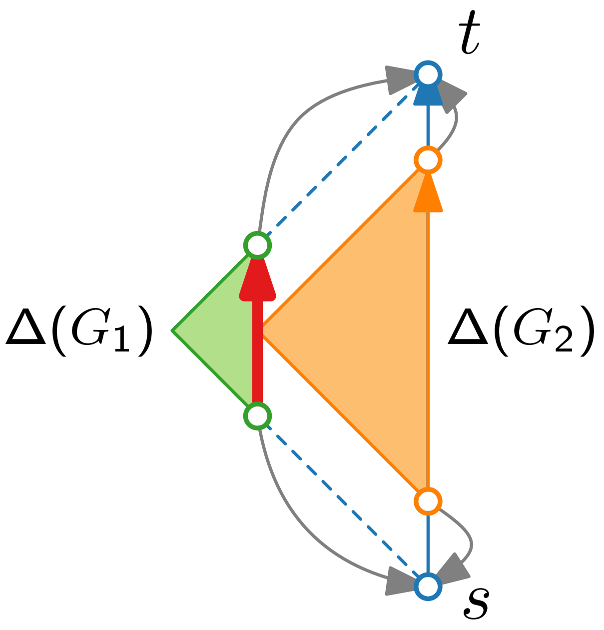


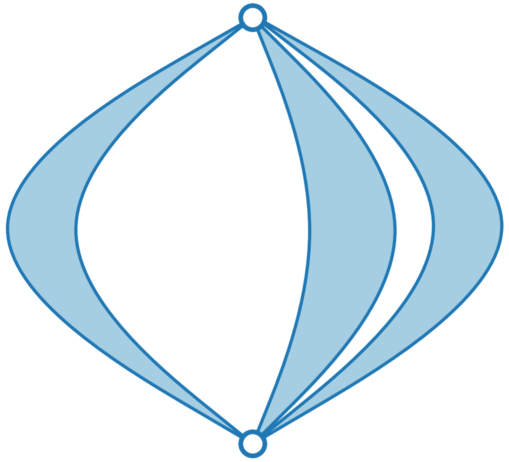
Conquer:

- S-nodes: series compositions
- P-nodes: parallel compositions





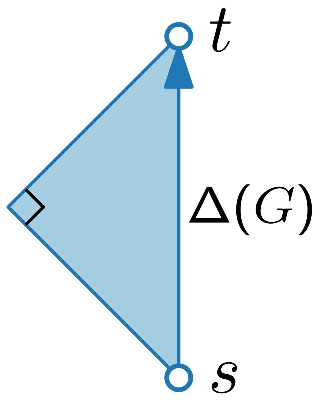




Series-Parallel Graphs – Straight-Line Drawings

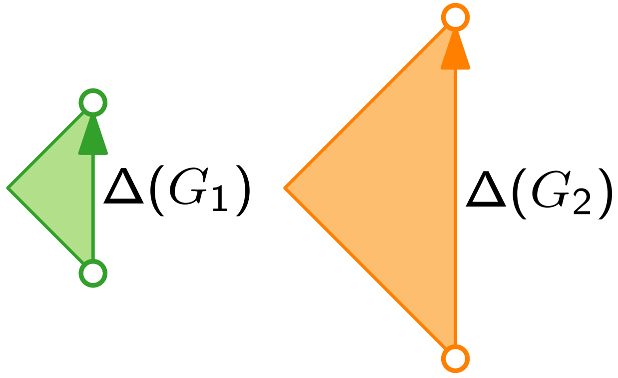
Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top



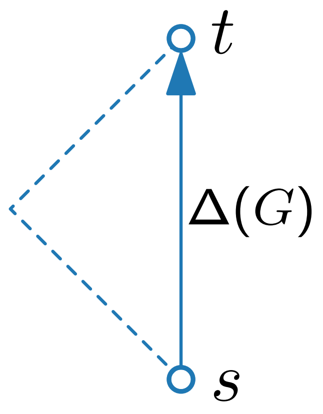
Base case: Q-nodes

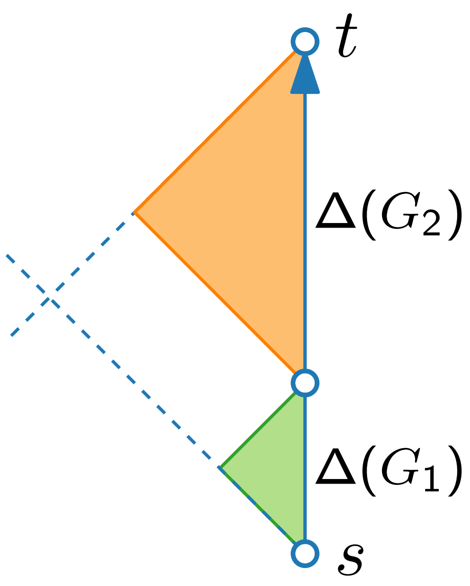
Divide: Draw G_1 and G_2 first

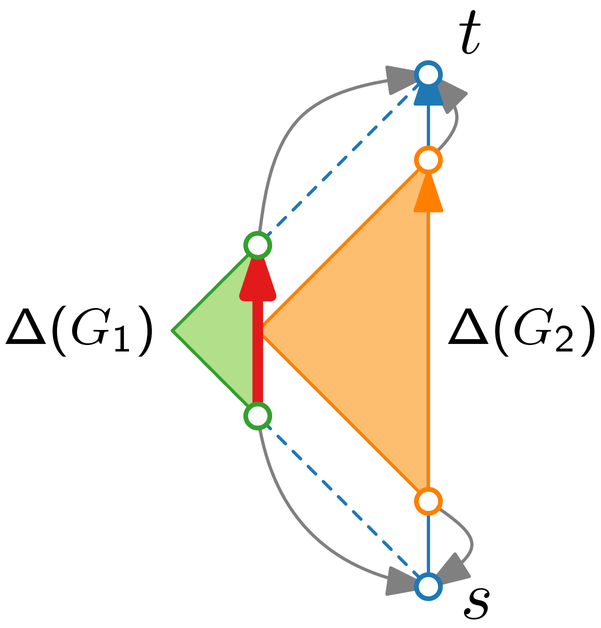


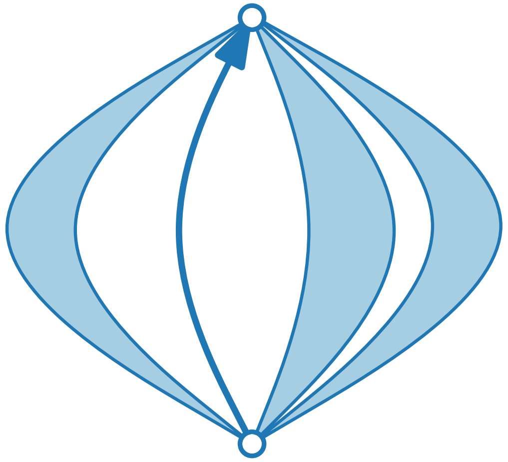
Conquer:

- S-nodes: series compositions
- P-nodes: parallel compositions

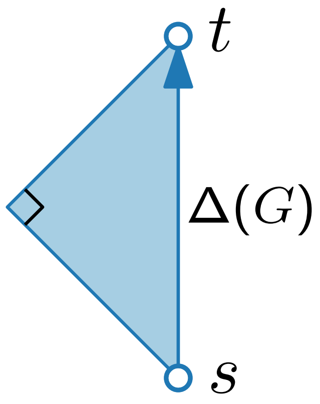








Series-Parallel Graphs – Straight-Line Drawings

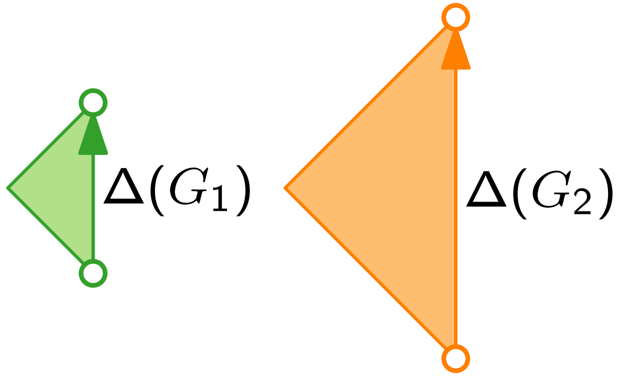


Divide & conquer algorithm using the decomposition tree

- Invariant: draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with s at the bottom and t at the top

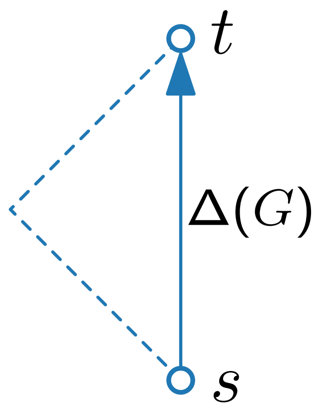
Base case: Q-nodes

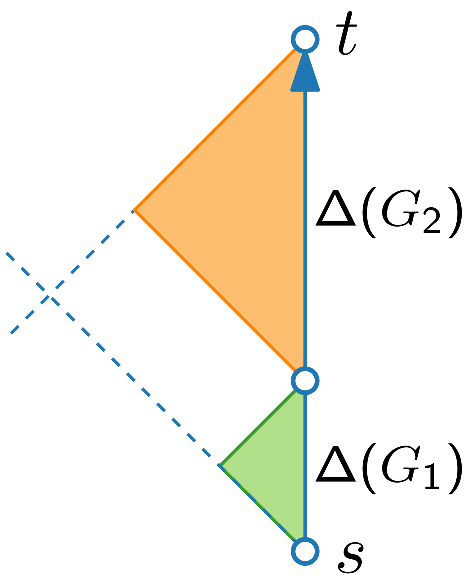
Divide: Draw G_1 and G_2 first

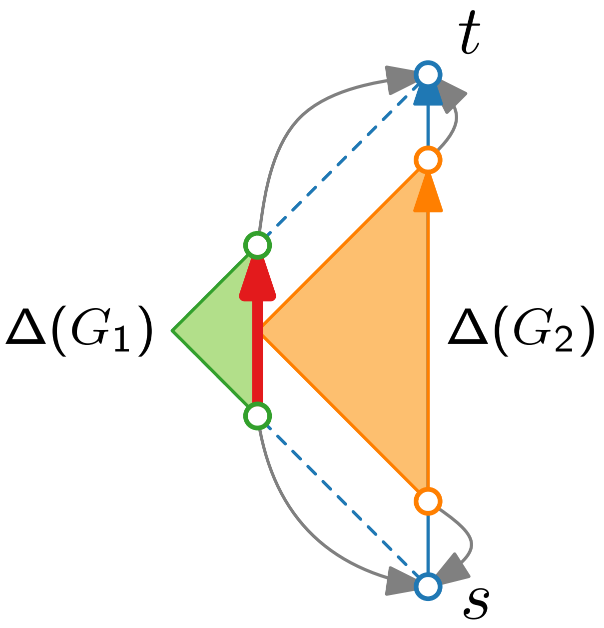


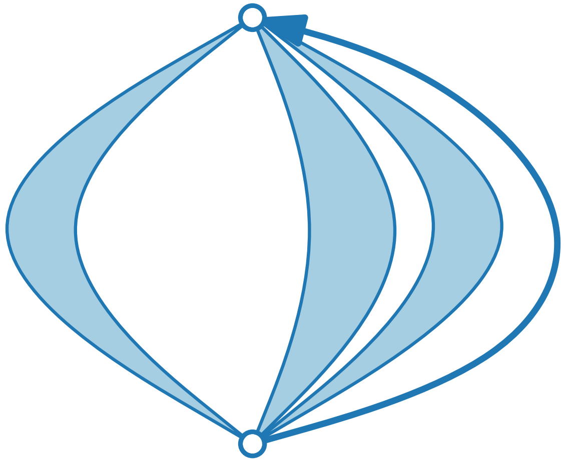
Conquer:

- S-nodes: series compositions
- P-nodes: parallel compositions







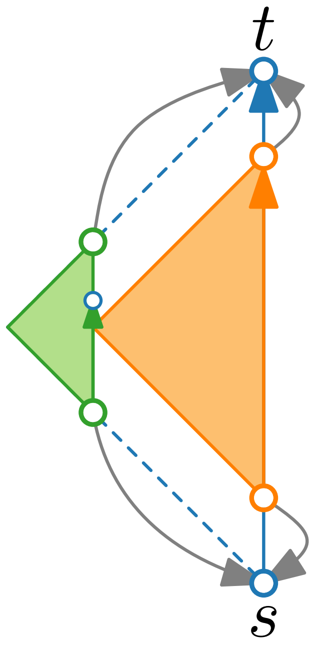


Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?

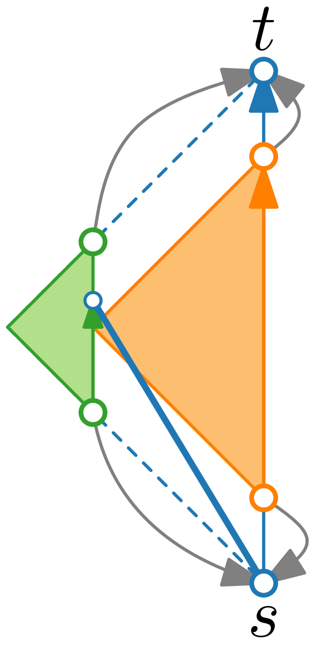
Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?



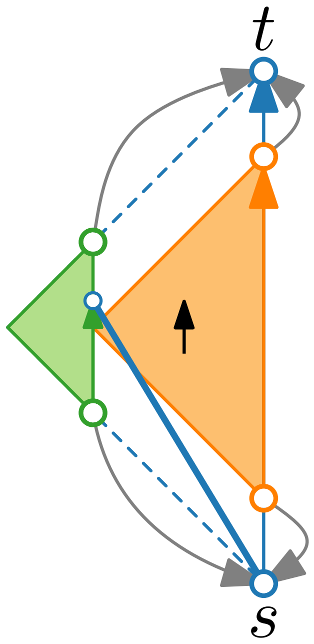
Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?



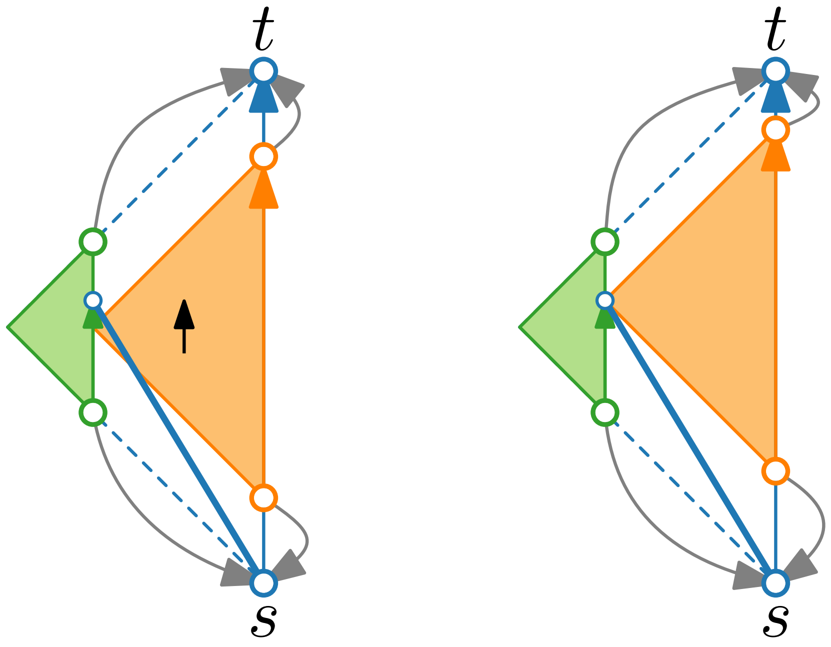
Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?



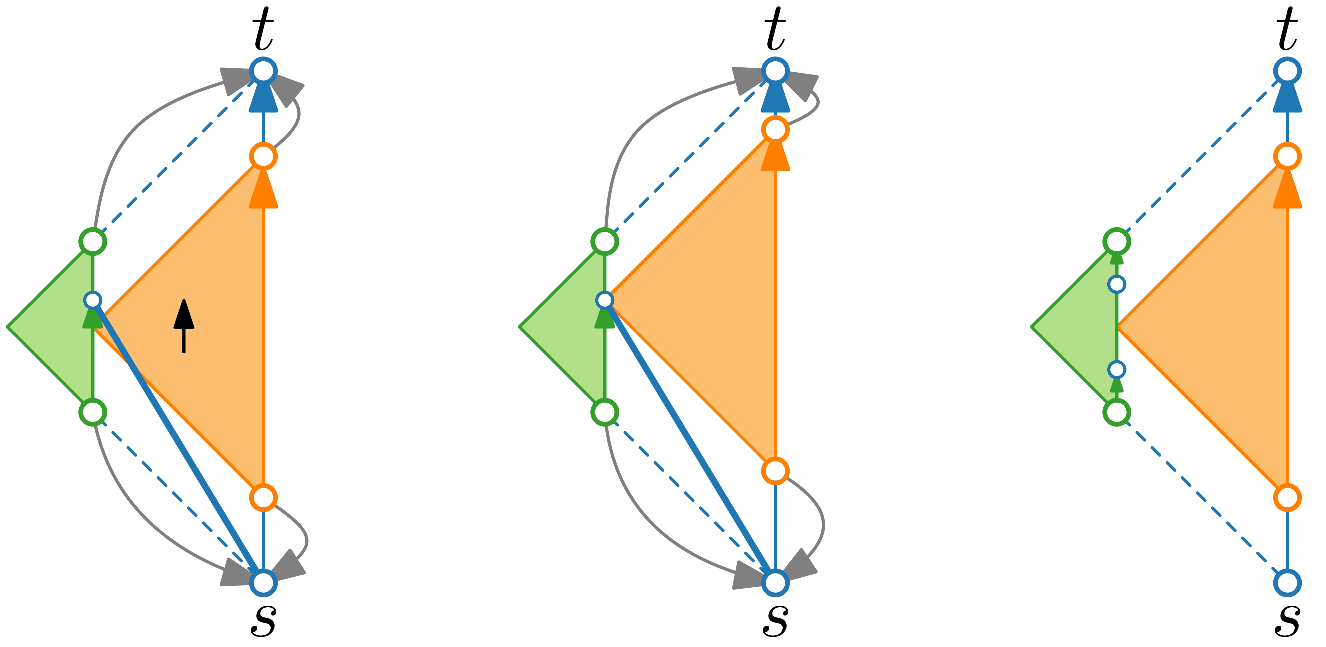
Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?



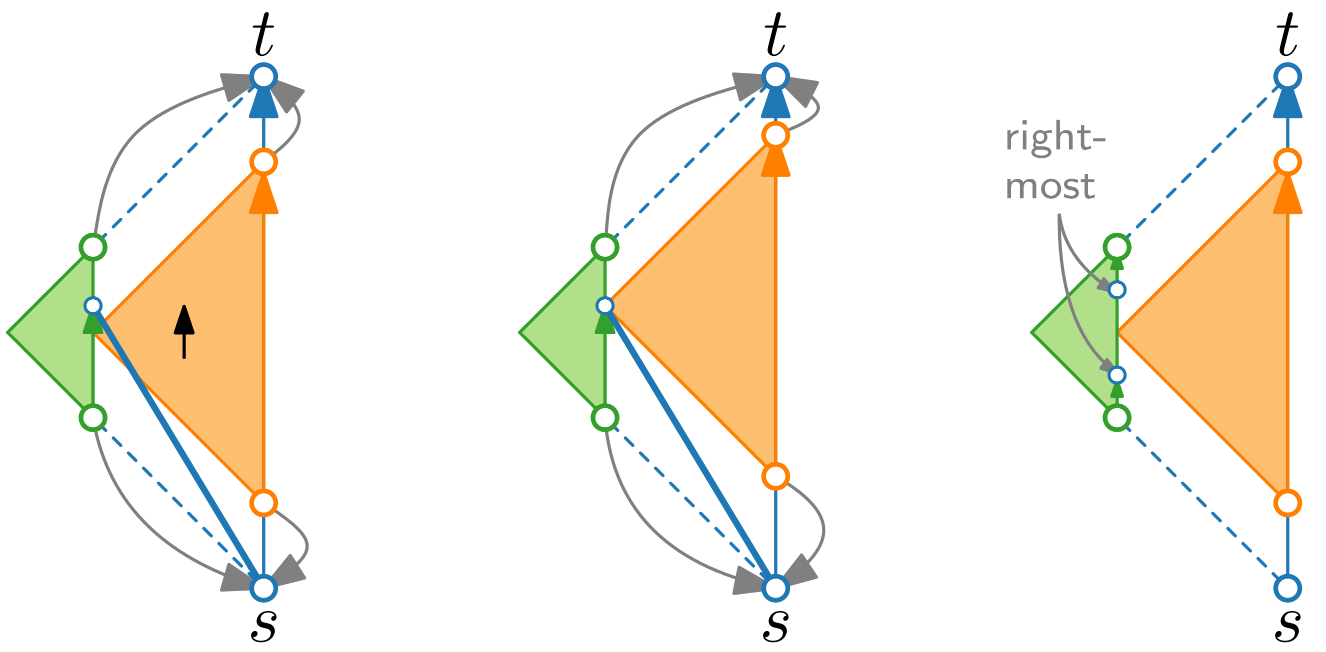
Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?



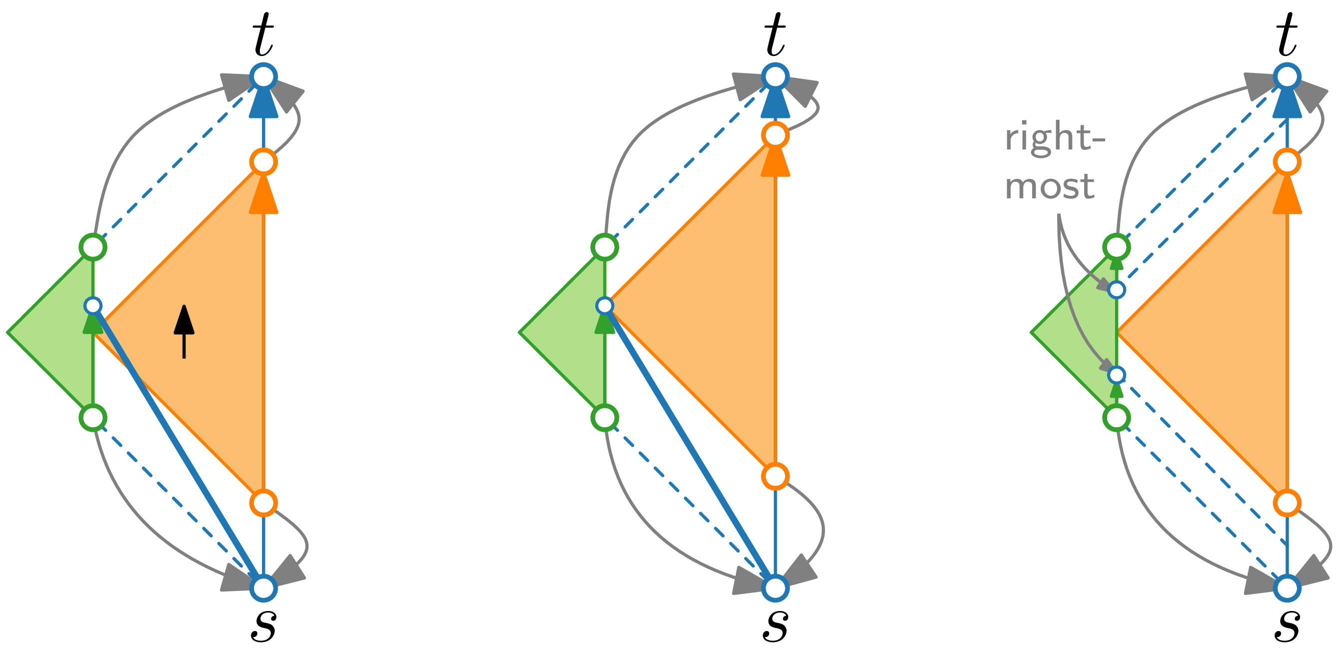
Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?



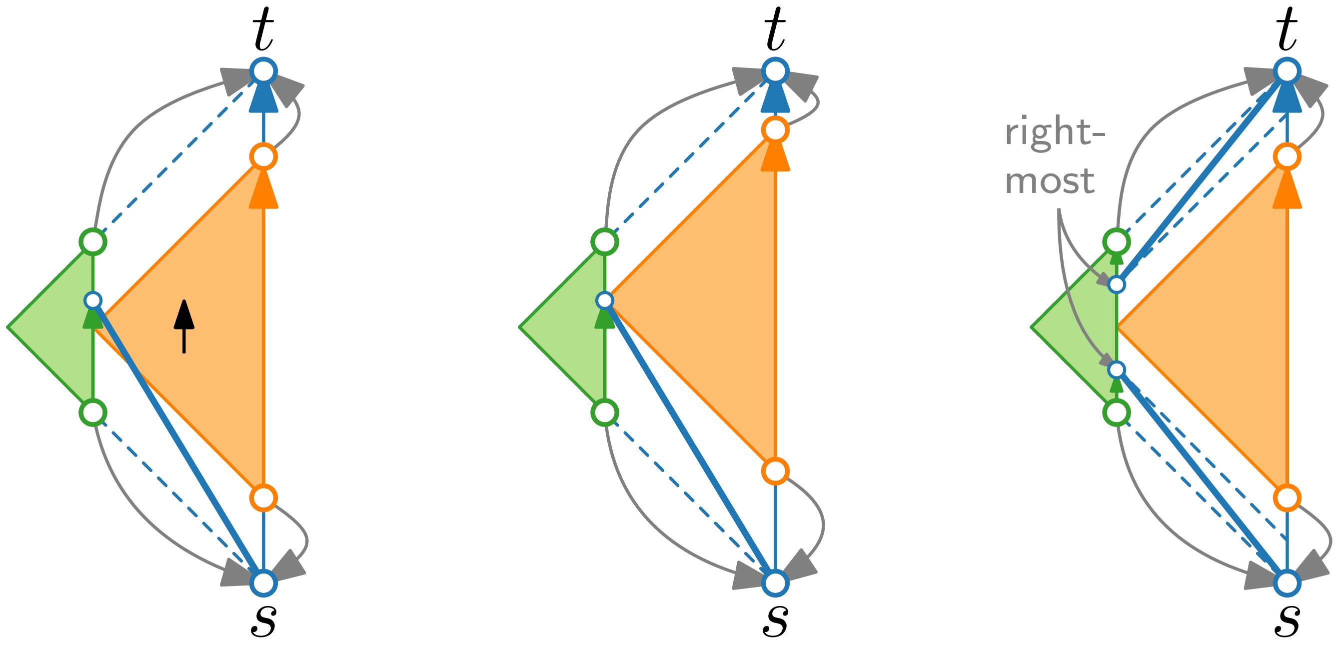
Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?



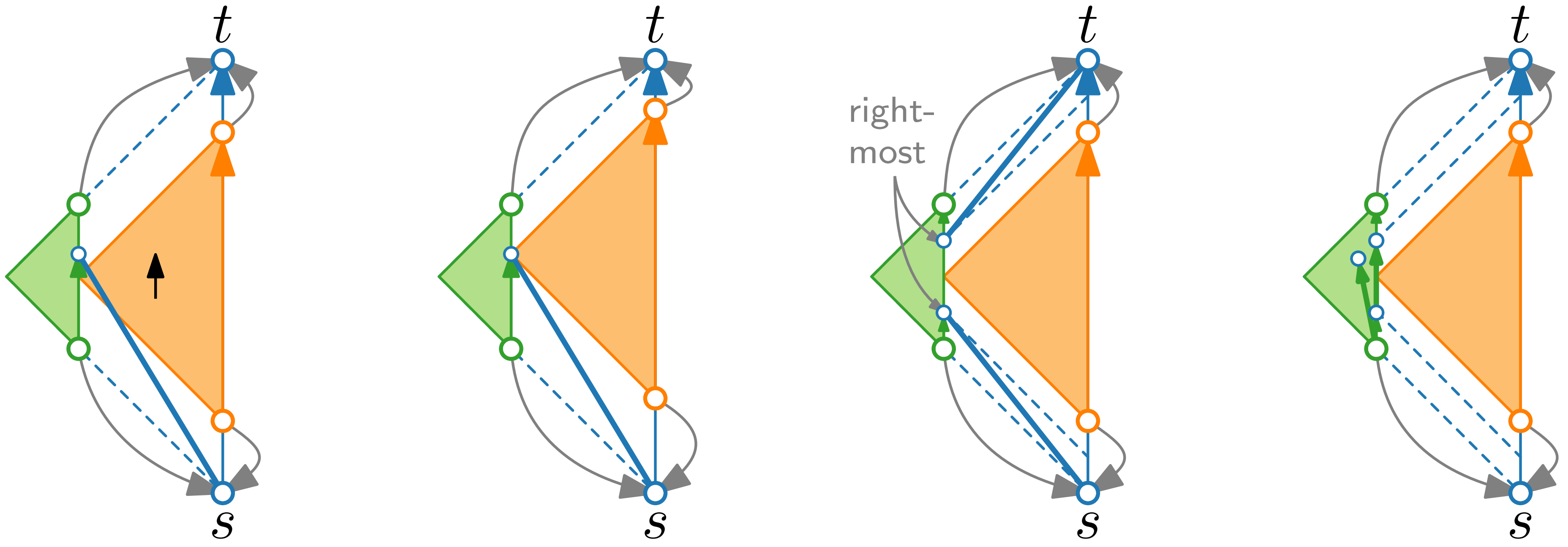
Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?



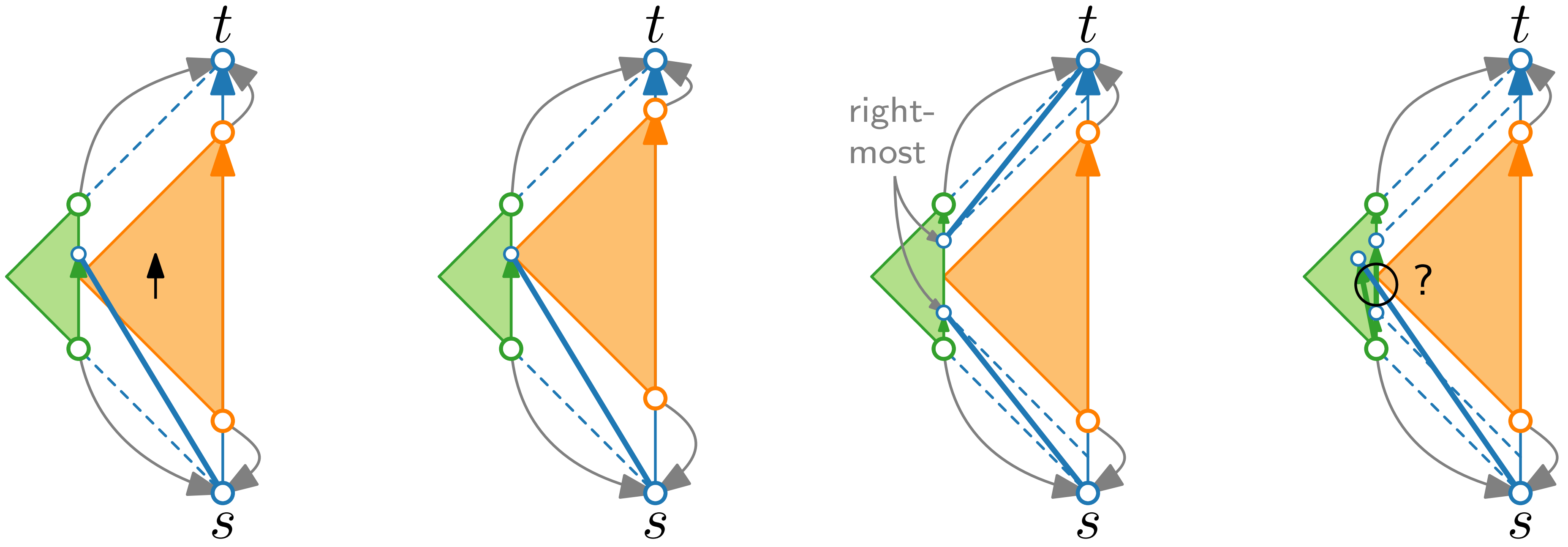
Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?



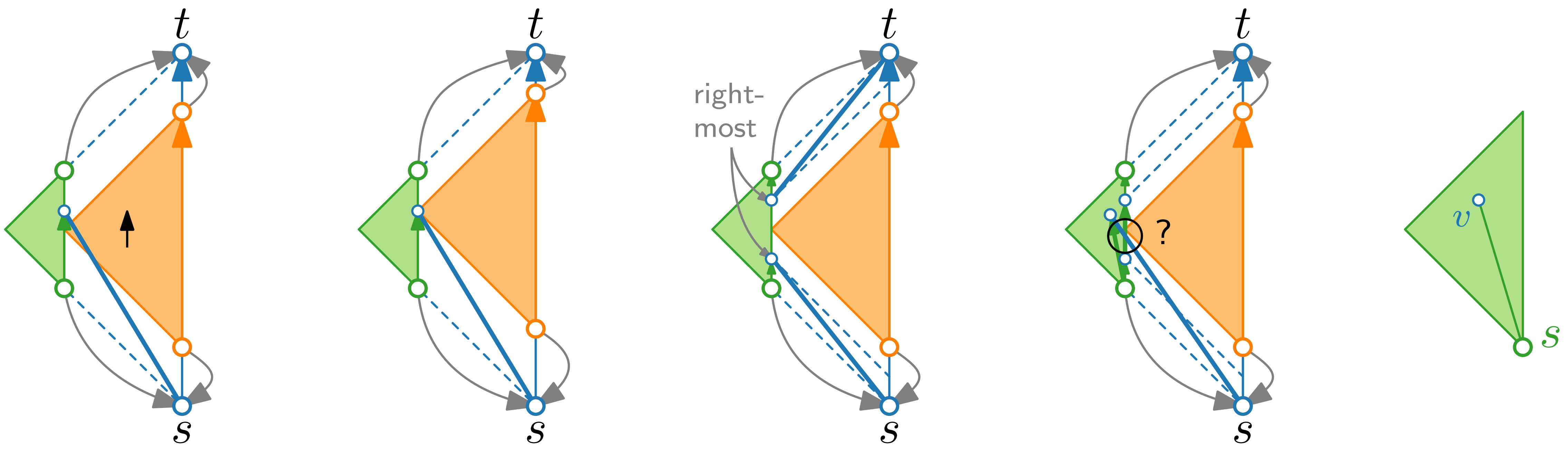
Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?



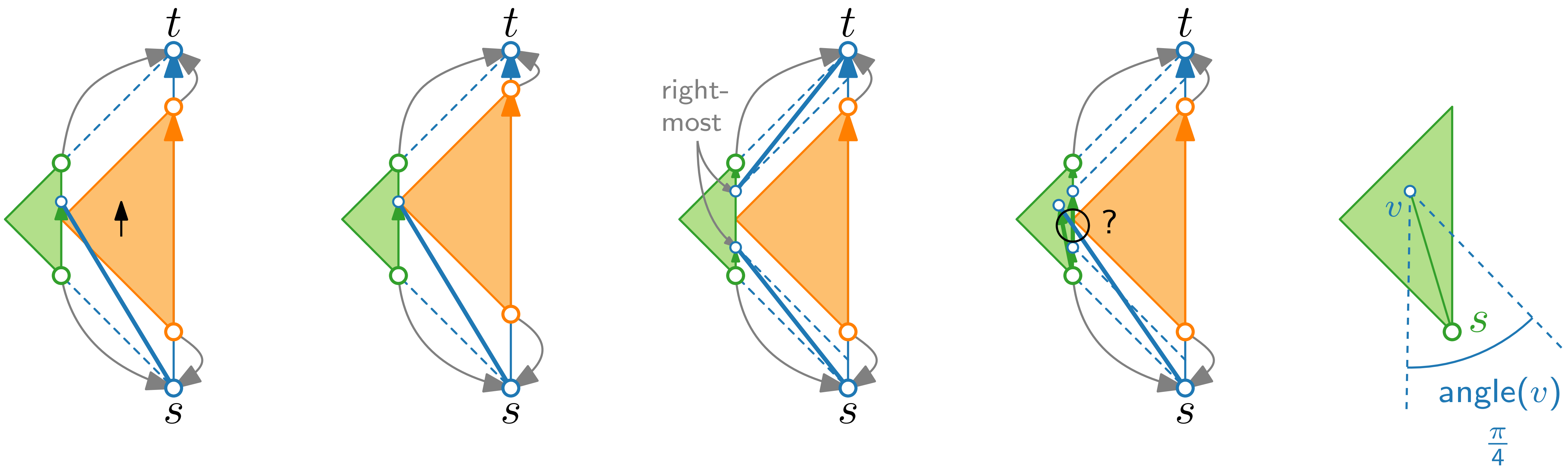
Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?



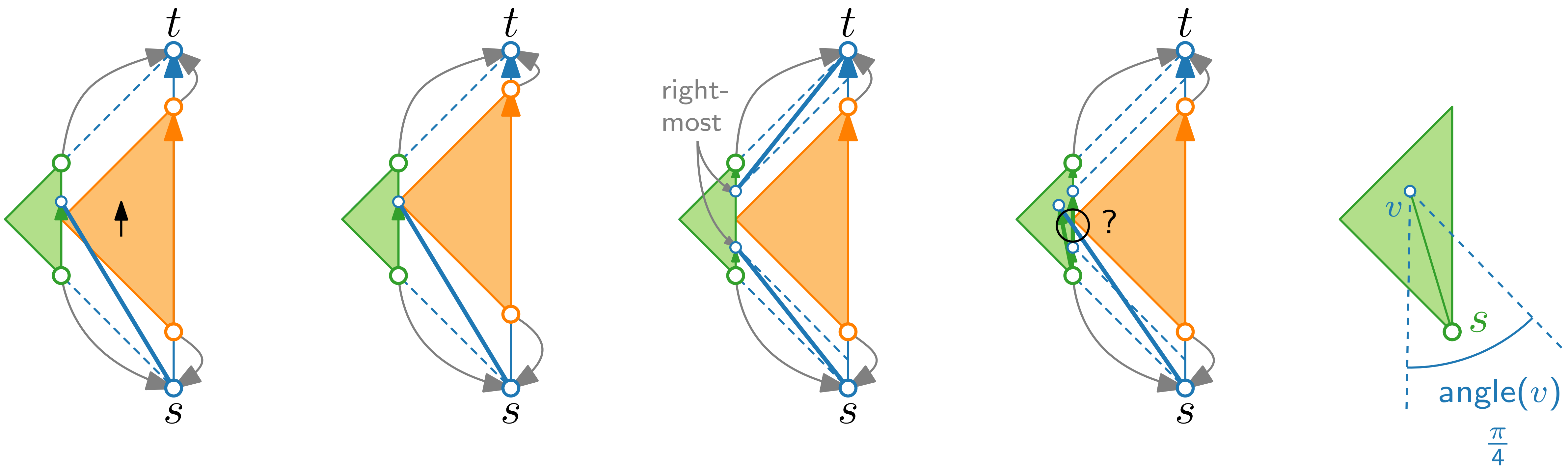
Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?



Series-Parallel Graphs – Straight-Line Drawings

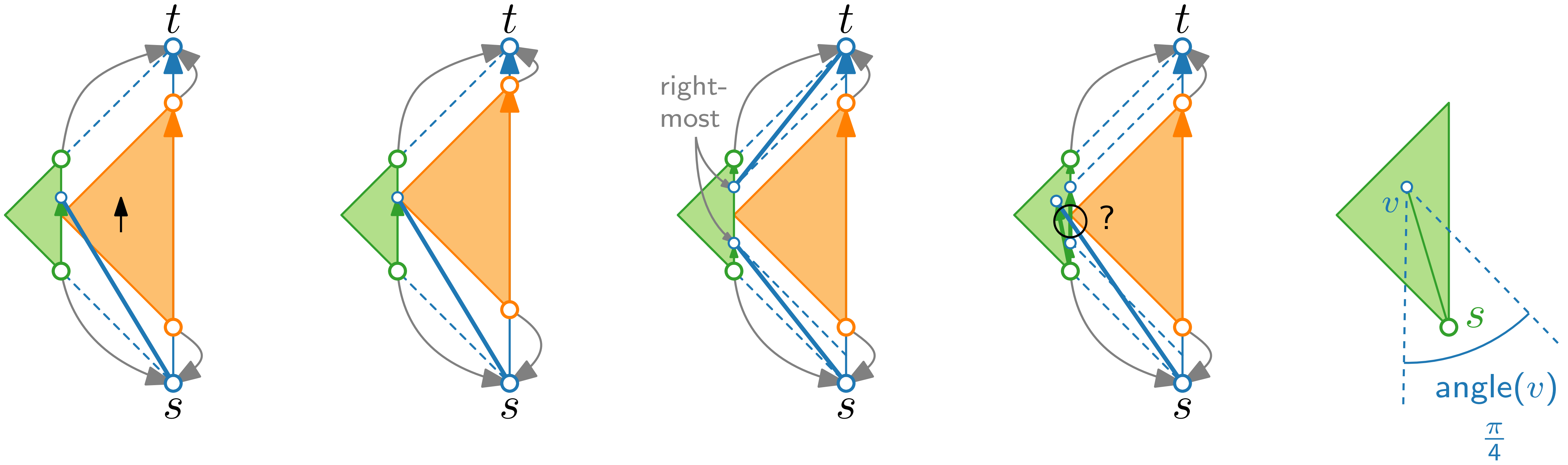
- What makes parallel composition possible without creating crossings?



Assume the following holds:
the only vertex in angle(v) is s

Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?

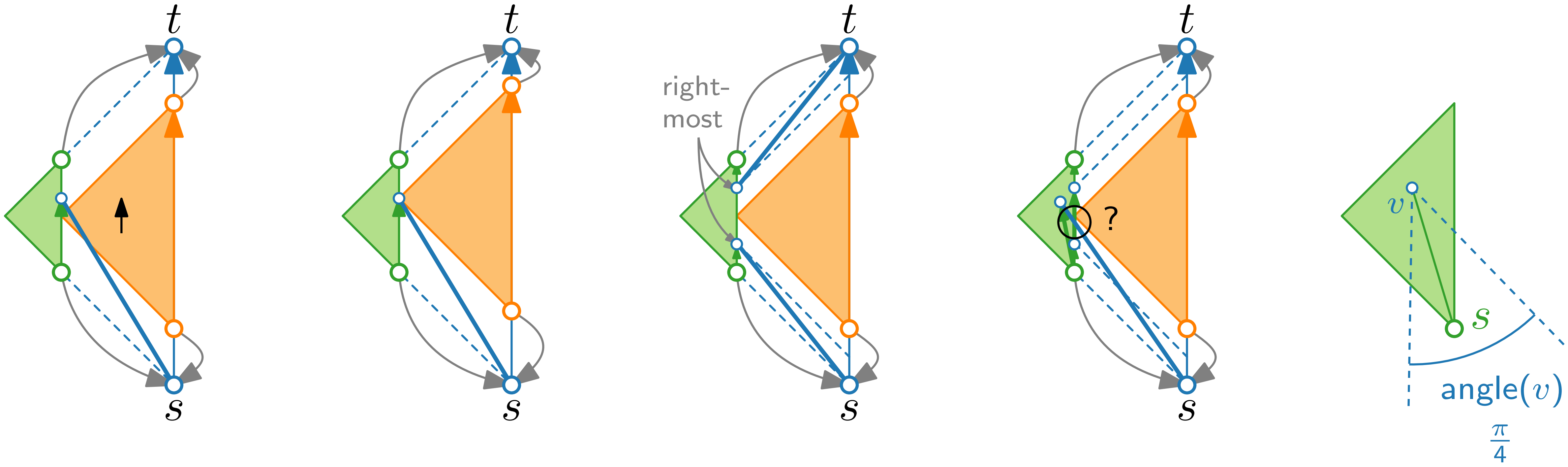


- This condition **is** preserved during the induction step.

Assume the following holds:
the only vertex in angle(v) is s

Series-Parallel Graphs – Straight-Line Drawings

- What makes parallel composition possible without creating crossings?



- This condition **is** preserved during the induction step.

Assume the following holds:
the only vertex in angle(v) is s

Lemma.
The drawing produced by the algorithm is planar.

Series-Parallel Graphs – Result

Theorem.
Let G be a series-parallel graph. Then G (with **variable embedding**) admits a drawing Γ that

Series-Parallel Graphs – Result

Theorem.
Let G be a series-parallel graph. Then G (with **variable embedding**) admits a drawing Γ that

- is upward planar,

Series-Parallel Graphs – Result

Theorem.
Let G be a series-parallel graph. Then G (with **variable embedding**) admits a drawing Γ that

- is upward planar,
- is straight-line, and

Series-Parallel Graphs – Result

Theorem.
Let G be a series-parallel graph. Then G (with **variable embedding**) admits a drawing Γ that

- is upward planar,
- is straight-line, and
- uses quadratic area.

Series-Parallel Graphs – Result

Theorem.
Let G be a series-parallel graph. Then G (with **variable embedding**) admits a drawing Γ that

- is upward planar,
- is straight-line, and
- uses quadratic area.
- Isomorphic components of G have congruent drawings up to translation.

Series-Parallel Graphs – Result

Theorem.
Let G be a series-parallel graph. Then G (with **variable embedding**) admits a drawing Γ that

- is upward planar,
- is straight-line, and
- uses quadratic area.
- Isomorphic components of G have congruent drawings up to translation.

Γ can be computed in linear time.

Series-Parallel Graphs – Fixed Embedding

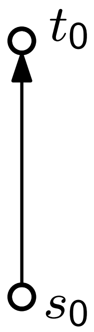
Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

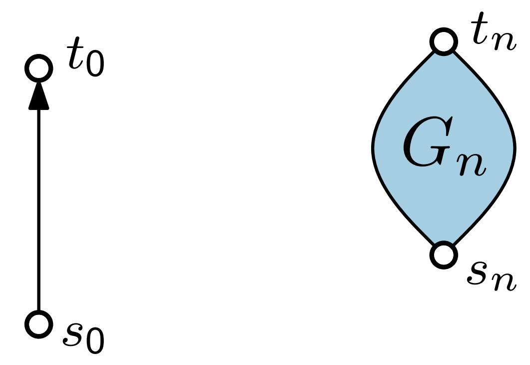


G_1

Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

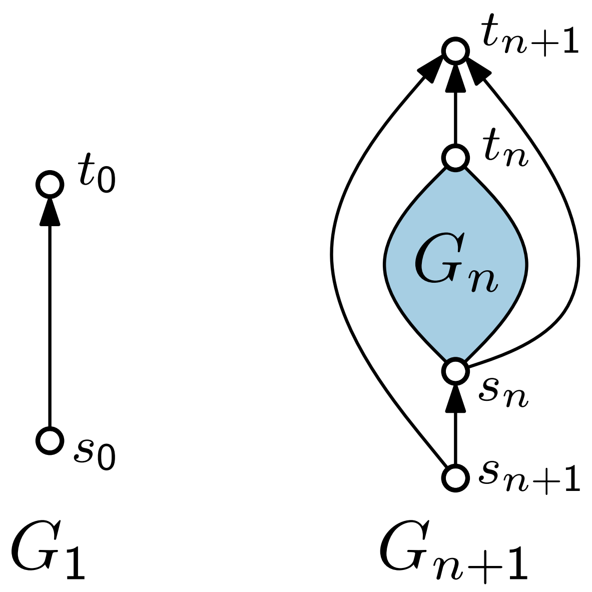


G_1 G_{n+1}

Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

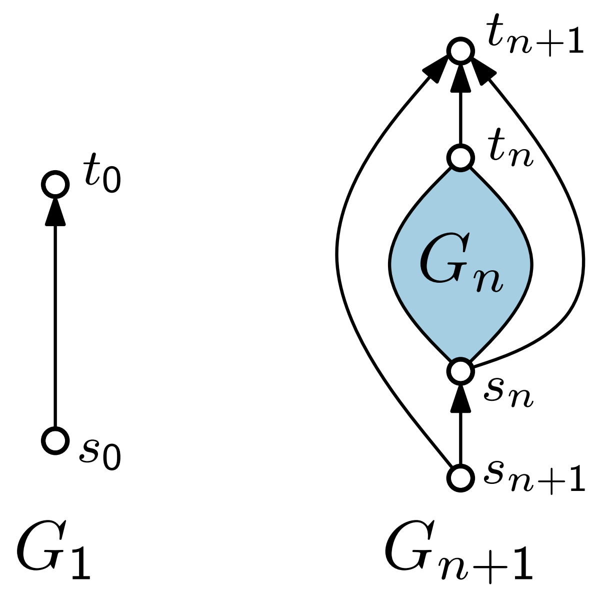
For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

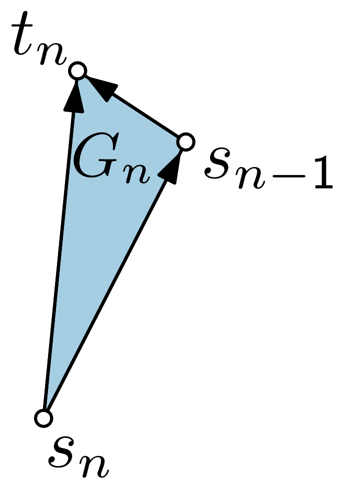


Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

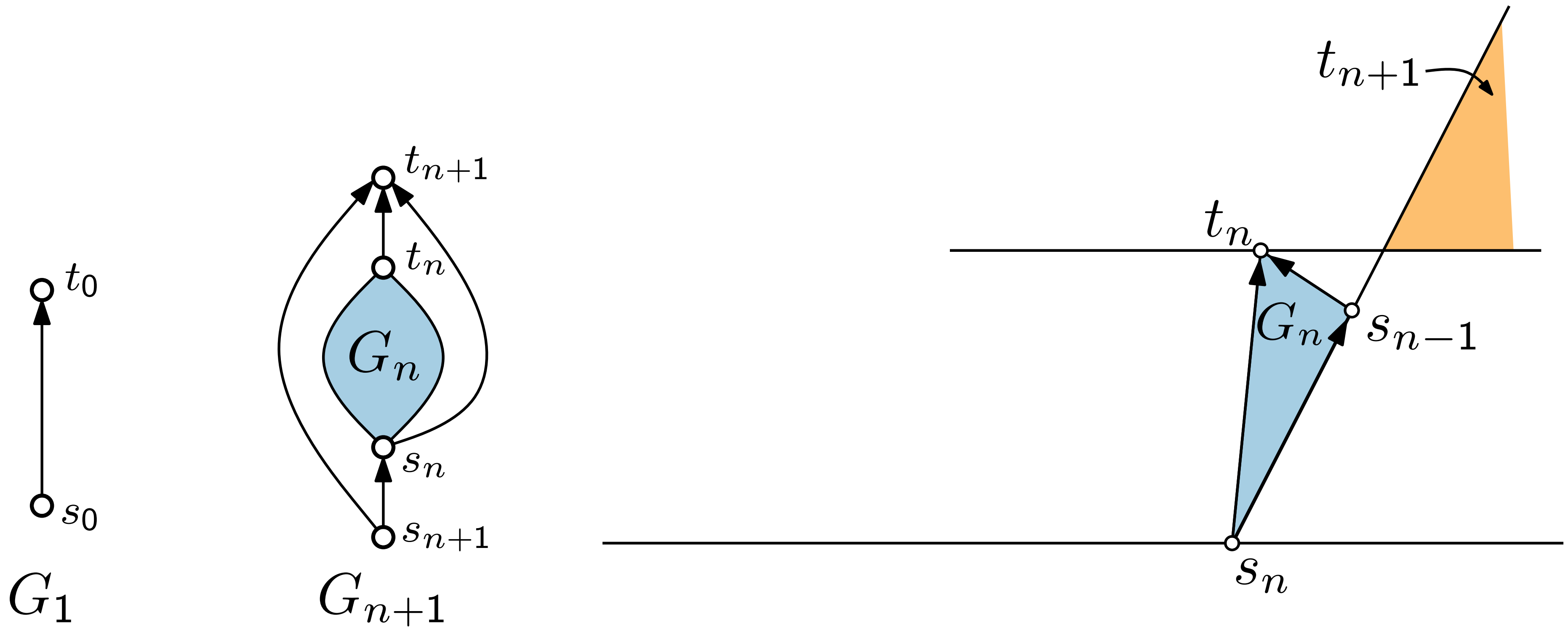




Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

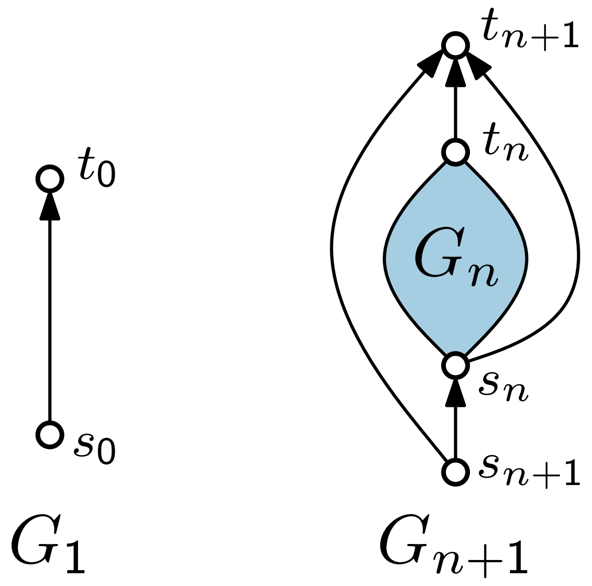
For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

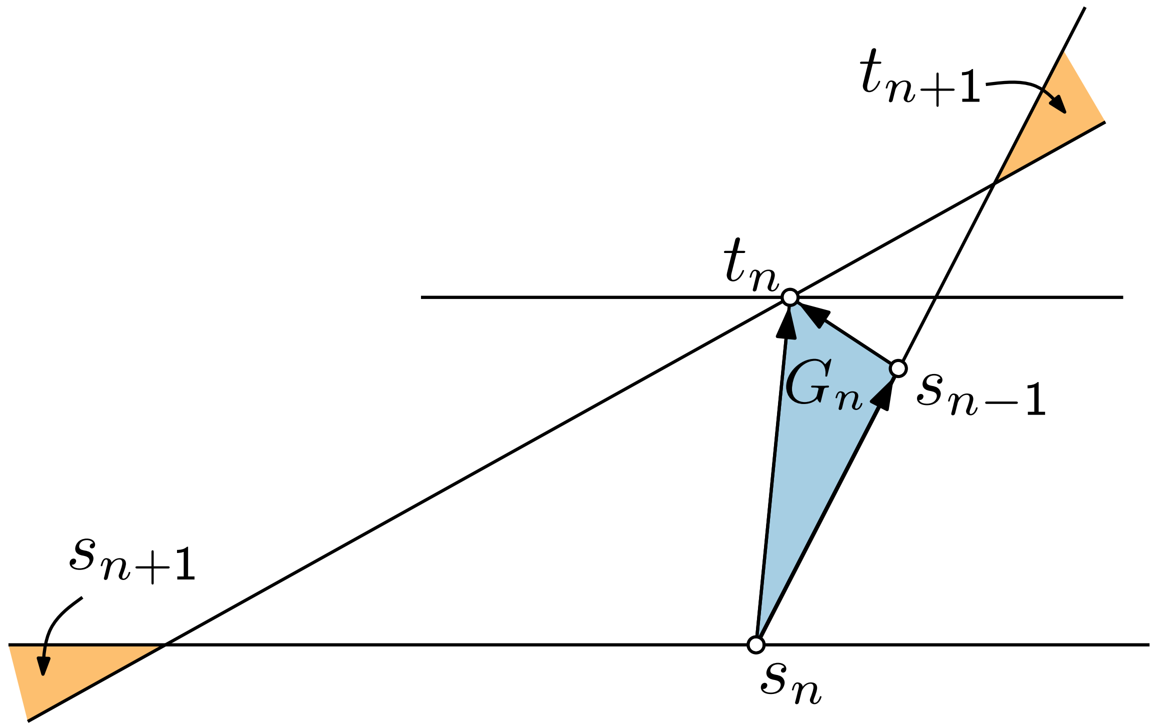


Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

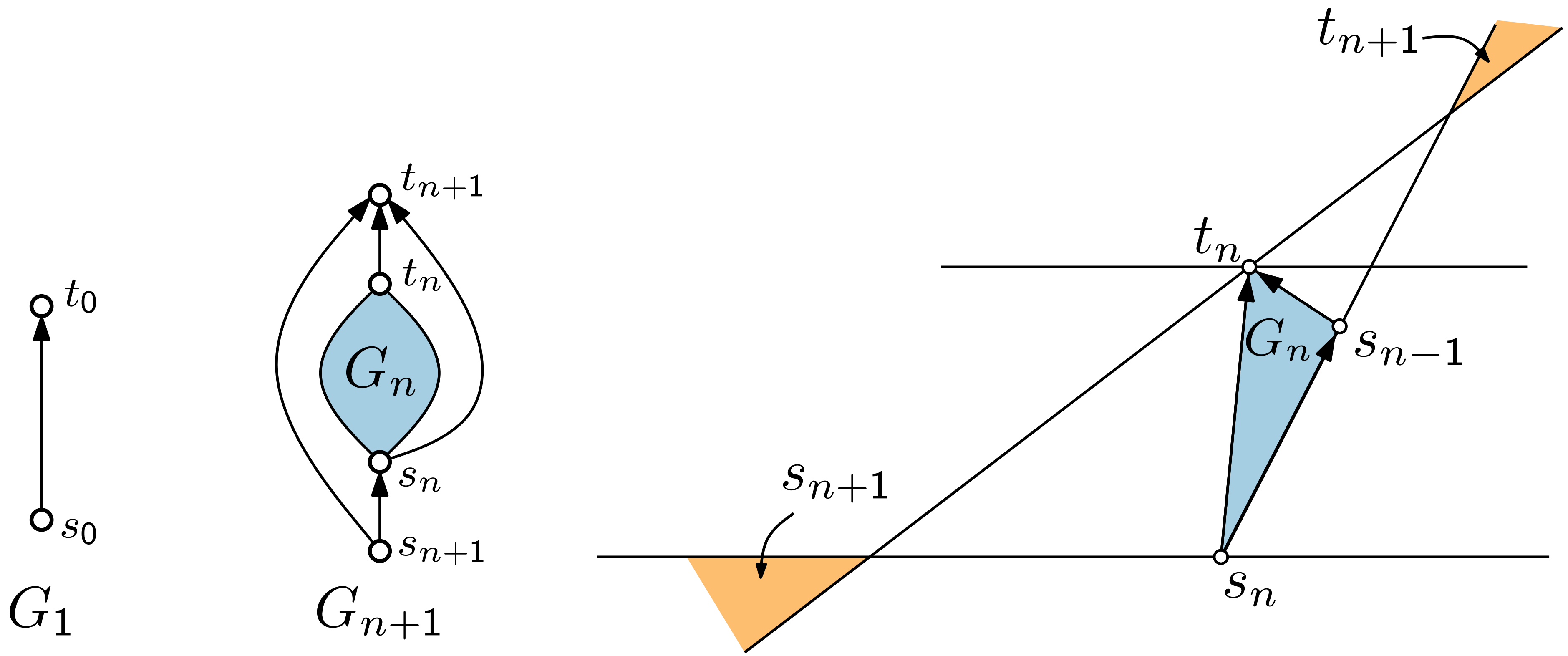




Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

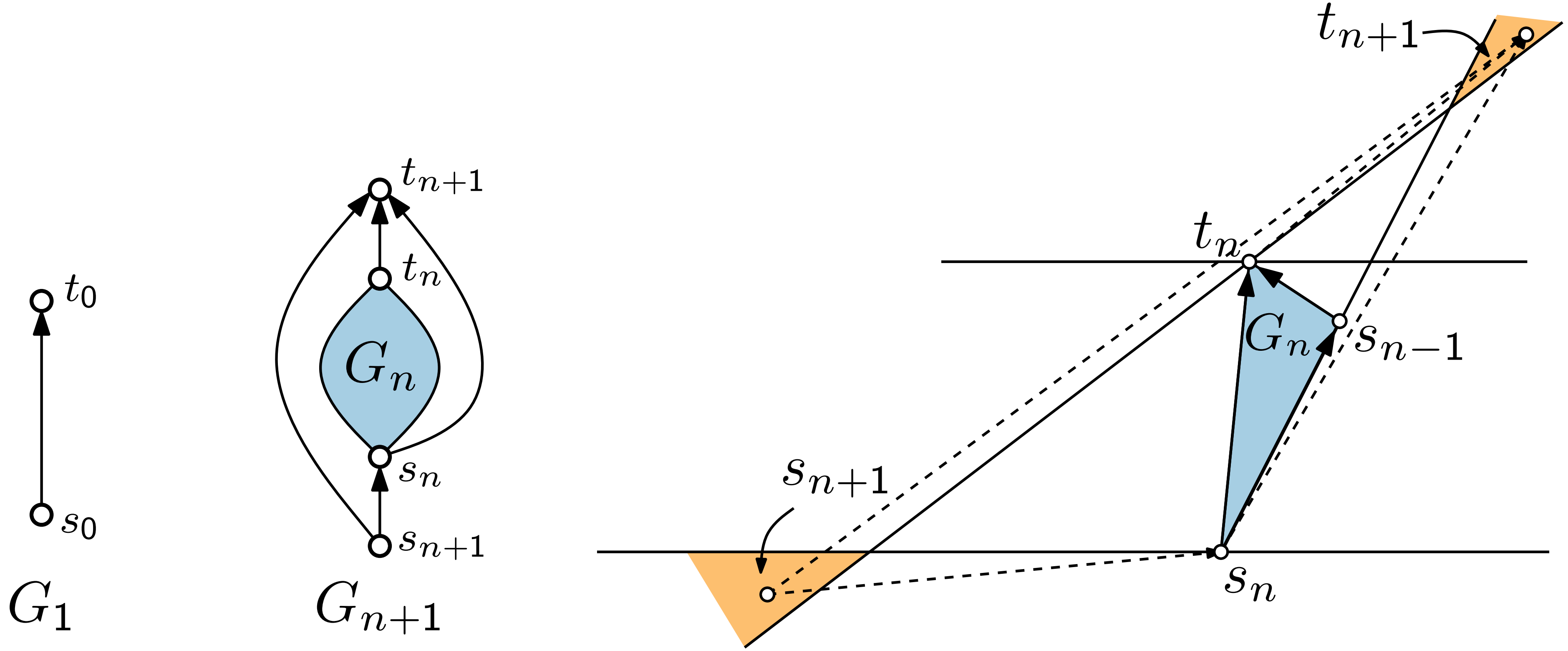
For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.



Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

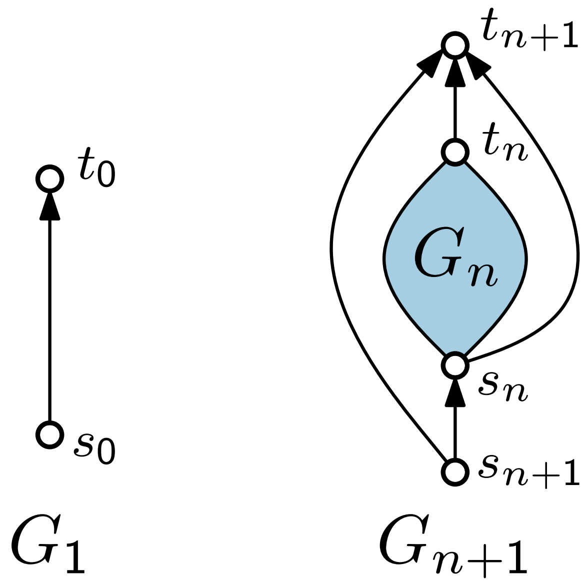
For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

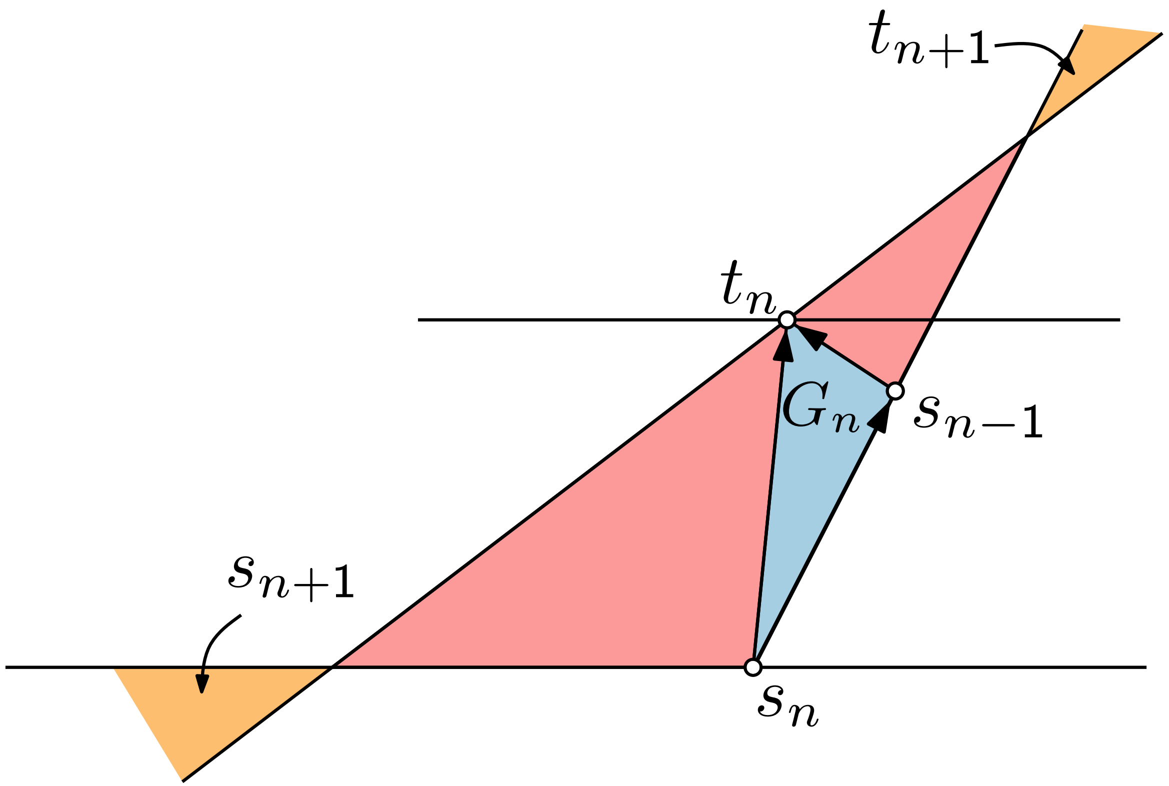


Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

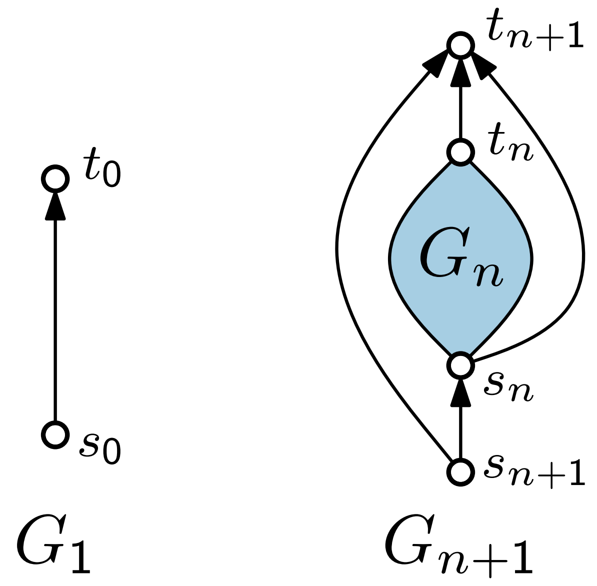


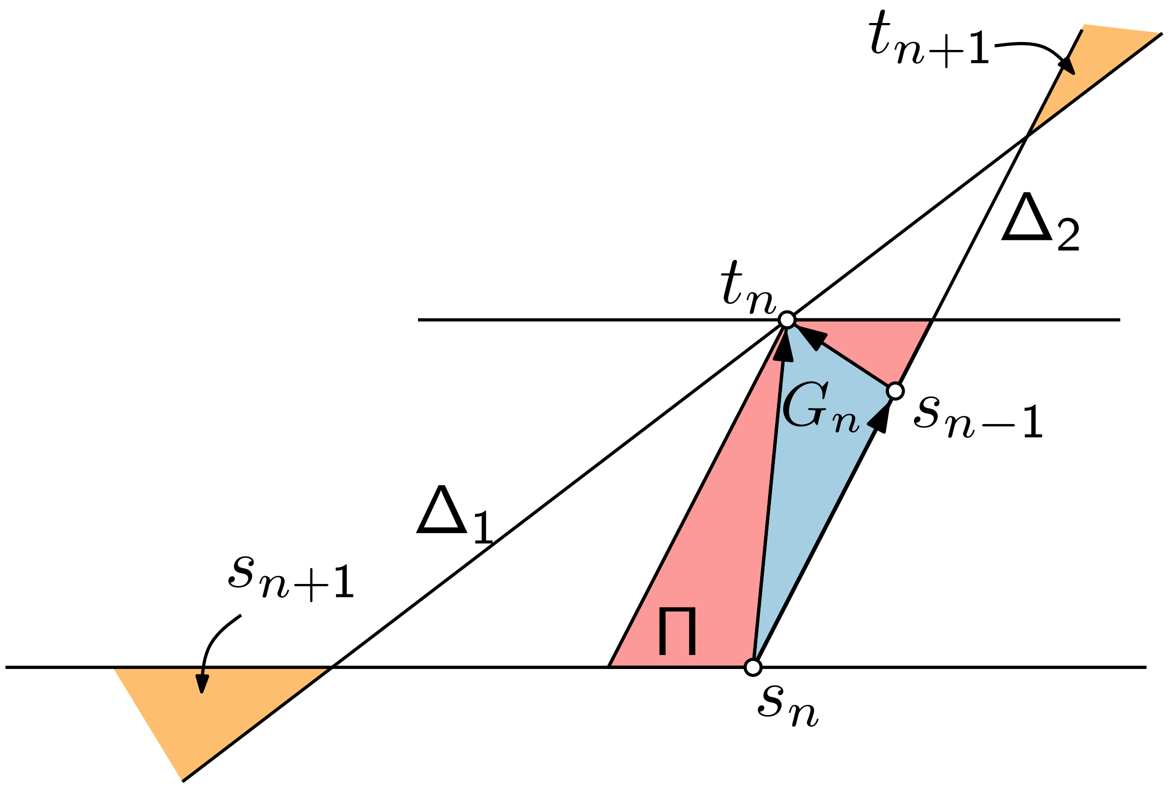


Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.



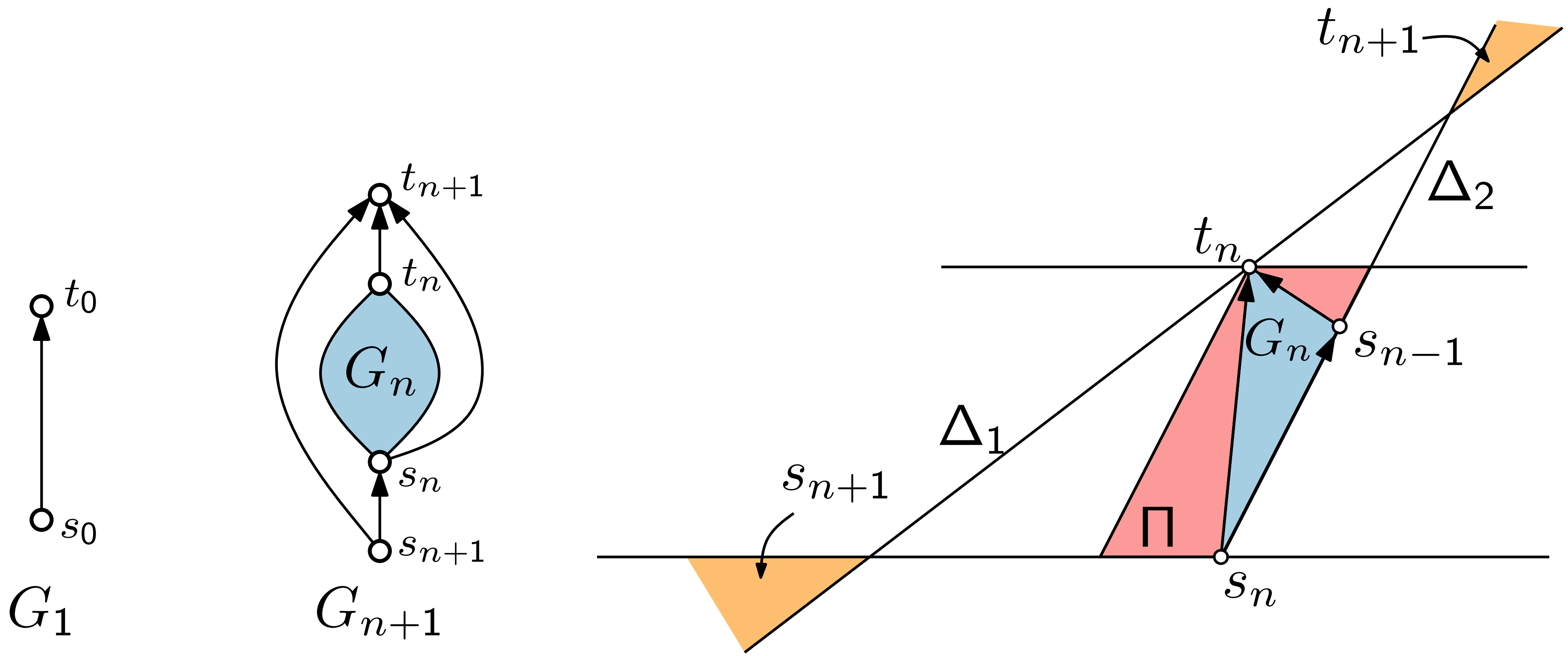


Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

- $2 \cdot \text{Area}(G_n) < \text{Area}(\Pi)$

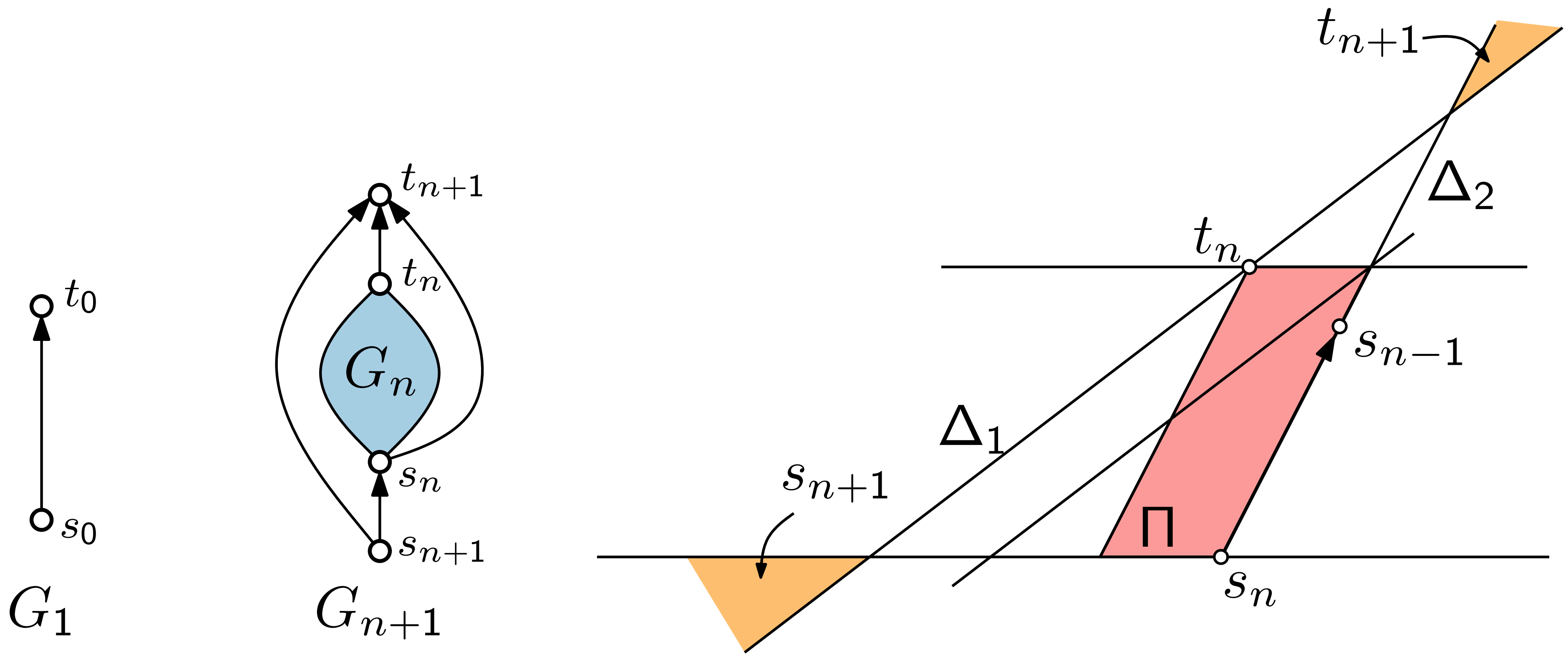


Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

- $2 \cdot \text{Area}(G_n) < \text{Area}(\Pi)$

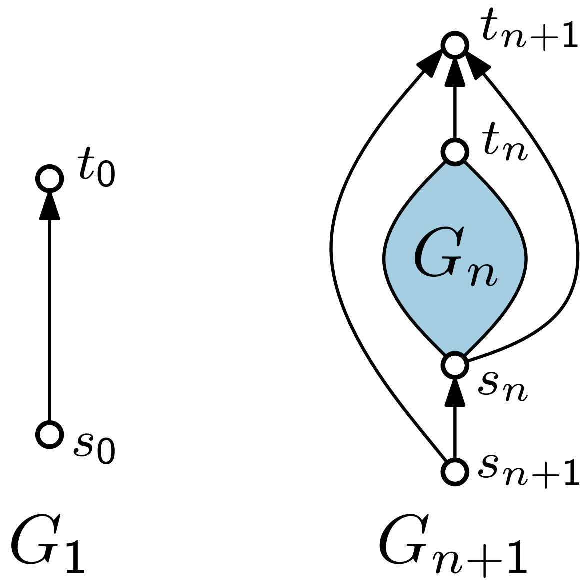


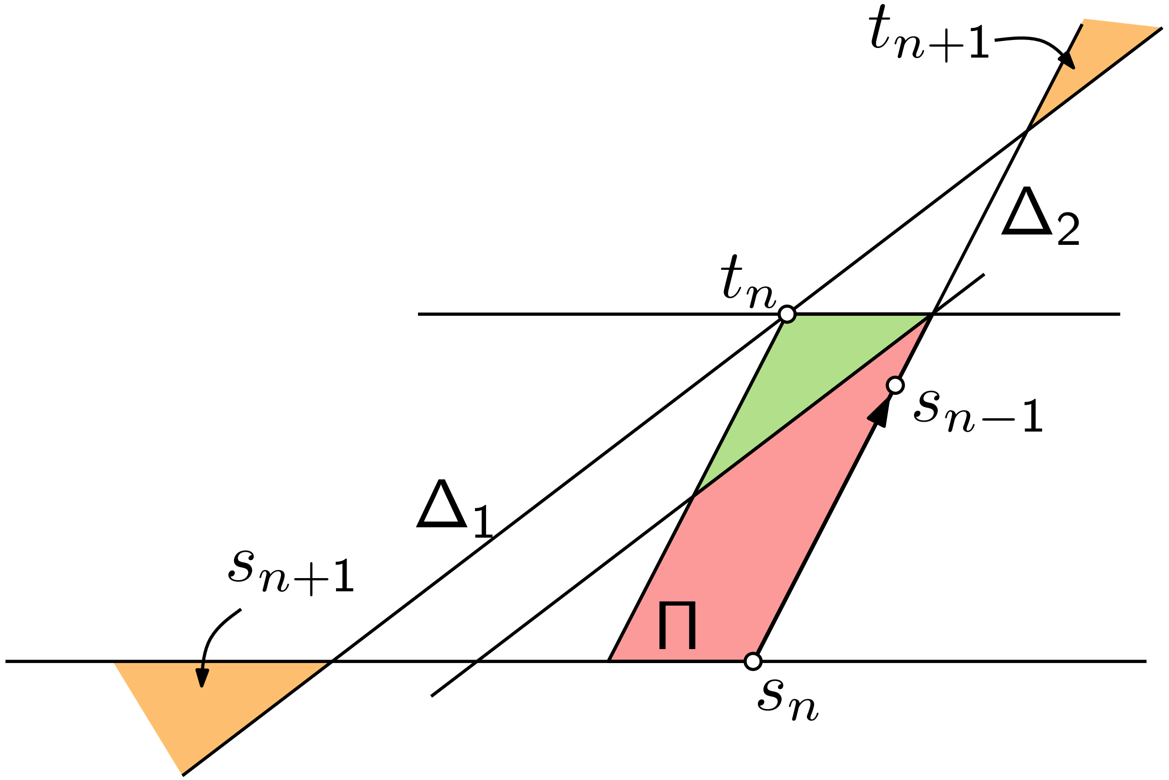
Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

- $2 \cdot \text{Area}(G_n) < \text{Area}(\Pi)$



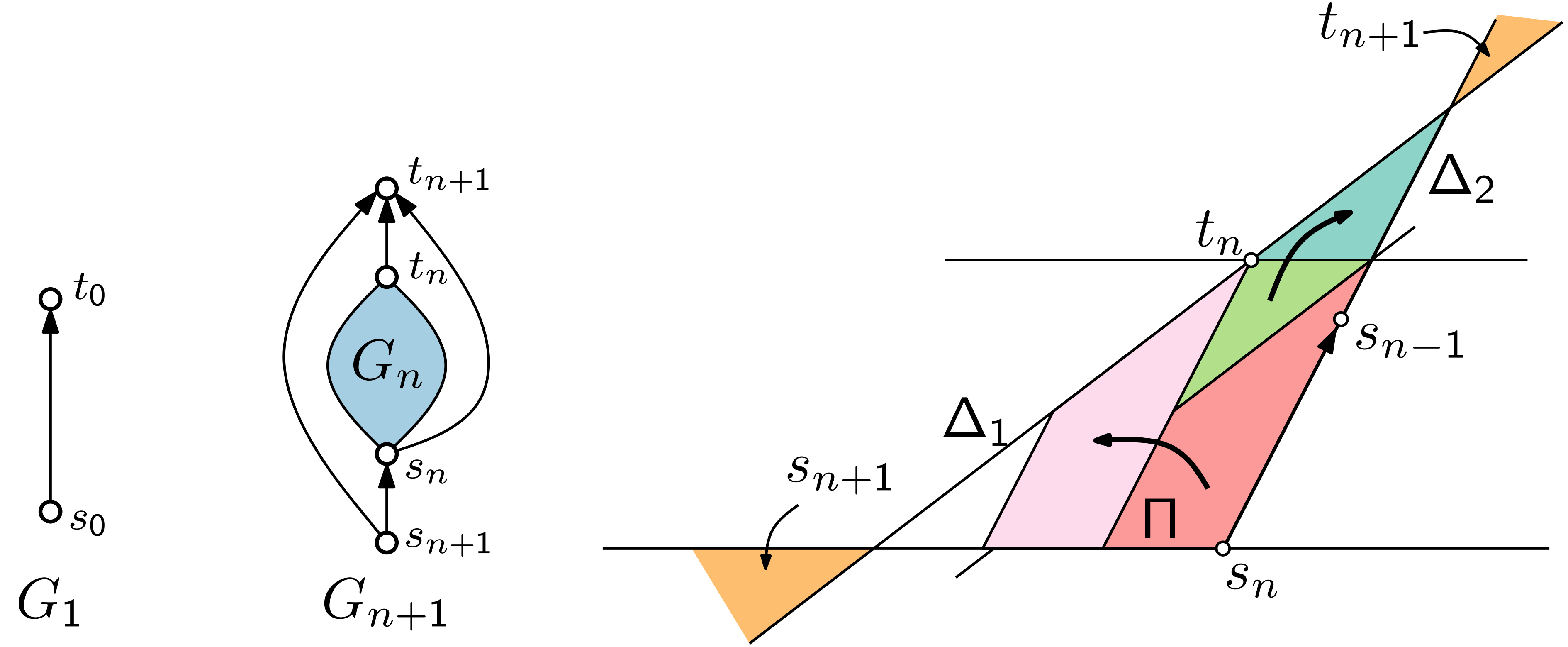


Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

- $2 \cdot \text{Area}(G_n) < \text{Area}(\Pi)$

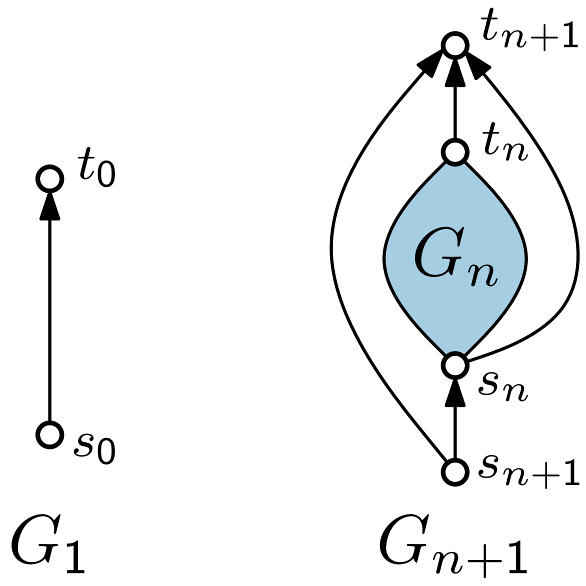


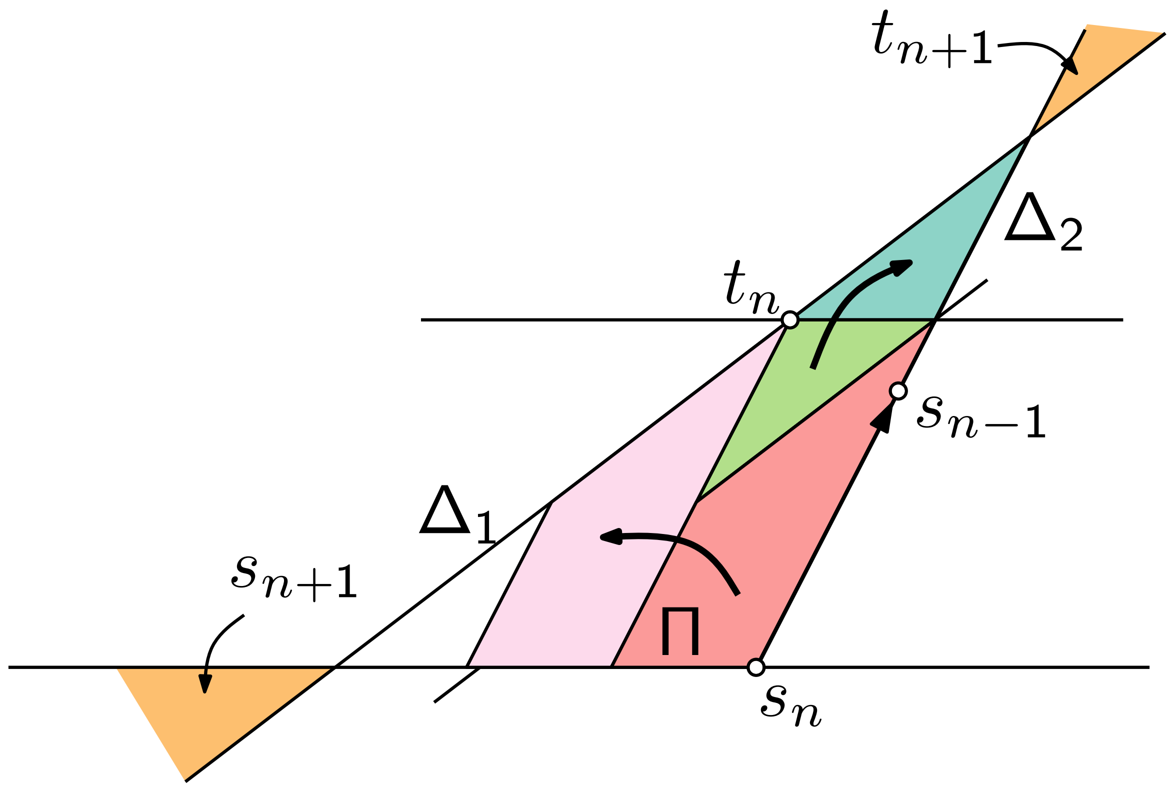
Series-Parallel Graphs – Fixed Embedding

Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

- $2 \cdot \text{Area}(G_n) < \text{Area}(\Pi)$
- $2 \cdot \text{Area}(\Pi) \quad \leq \text{Area}(G_{n+1})$





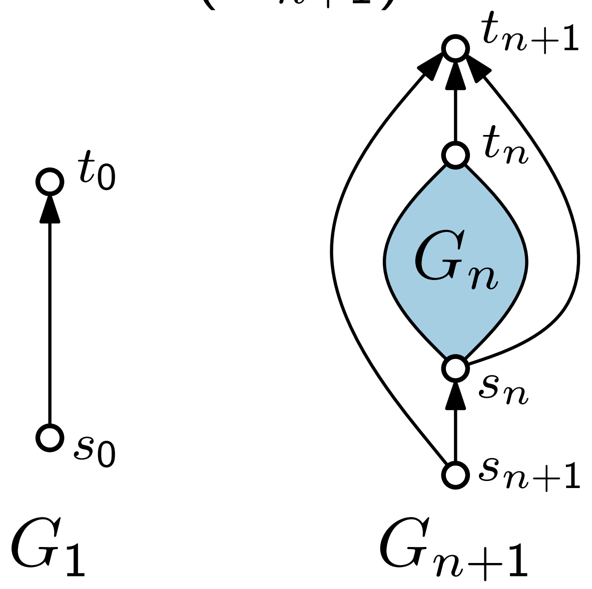
Series-Parallel Graphs – Fixed Embedding

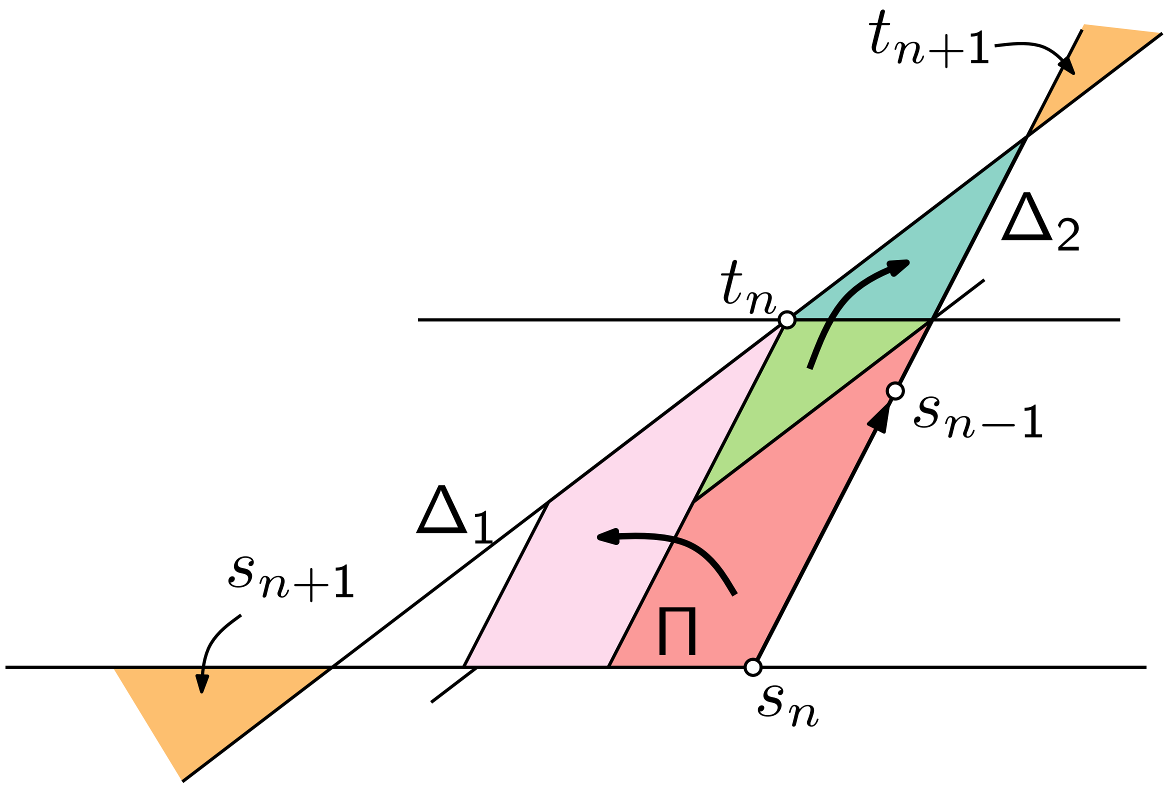
Theorem. [Bertolazzi, Di Battista, Mannino, Tamassia '94]

For any $n \geq 1$, there exists a $2n$-vertex series-parallel graph G_n in an embedding such that any upward planar straight-line drawing of G_n that respects the given embedding requires $\Omega(4^n)$ area.

- $2 \cdot \text{Area}(G_n) < \text{Area}(\Pi)$
- $2 \cdot \text{Area}(\Pi) \leq \text{Area}(G_{n+1})$

$\Rightarrow 4 \cdot \text{Area}(G_n) < \text{Area}(G_{n+1})$





Discussion

- There exist fixed-parameter (FPT) algorithms to test upward planarity of general digraphs with the parameter being the number of triconnected components.

  [Healy, Lynch 2005, Didimo et al. 2009]

Discussion

- There exist fixed-parameter (FPT) algorithms to test upward planarity of general digraphs with the parameter being the number of triconnected components.
  [Healy, Lynch 2005, Didimo et al. 2009]

- Finding a consistent assignment (Theorem 2) can be sped up to $\mathcal{O}(n + r^{1.5})$, where $r = \#$ sources.
  [Abbasi, Healy, Rextin 2010]

Discussion

- There exist fixed-parameter (FPT) algorithms to test upward planarity of general digraphs with the parameter being the number of triconnected components.
[Healy, Lynch 2005, Didimo et al. 2009]

- Finding a consistent assignment (Theorem 2) can be sped up to $\mathcal{O}(n + r^{1.5})$, where $r = \#$ sources.
[Abbasi, Healy, Rextin 2010]

- Many related concepts have been studied:
upward drawings of mixed graphs, upward drawings with layers for the vertices, upward planarity on cylinder/torus, ...

Literature

- See [GD Ch. 6] for detailed explanation on upward planarity.
- See [GD Ch. 3] for divide and conquer methods of series-parallel graphs

Orginal papers referenced:

- [Kelly '87] Fundamentals of Planar Ordered Sets
- [Di Battista &Tamassia '88] Algorithms for Plane Representations of Acyclic Digraphs
- [Garg &Tamassia '95]
  On the Computational Complexity of Upward and Rectilinear Planarity Testing
- [Hutton & Lubiw '96] Upward Planar Drawing of Single-Source Acyclic Digraphs
- [Bertolazzi, Di Battista, Mannino, Tamassia '94]
  Upward Drawings of Triconnected Digraphs
- [Healy & Lynch '05] Building Blocks of Upward Planar Digraphs
- [Didimo, Giordano, Liotta '09] Upward Spirality and Upward Planarity Testing
- [Abbasi, Healy, Rextin '10]
  Improving the running time of embedded upward planarity testing