Planar L-Drawings of Directed Graphs

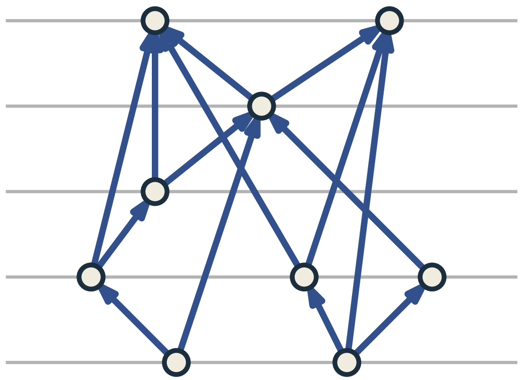
Steven Chaplick, Markus Chimani, Sabine Cornelsen, Giordano Da Lozzo, Martin Nöllenburg, Maurizio Patrignani, Ioannis G. Tollis, Alexander Wolff

Graph Drawing and Network Visualization 2017 · Boston

TU WIEN ac ALGORITHMS AND COMPLEXITY GROUP

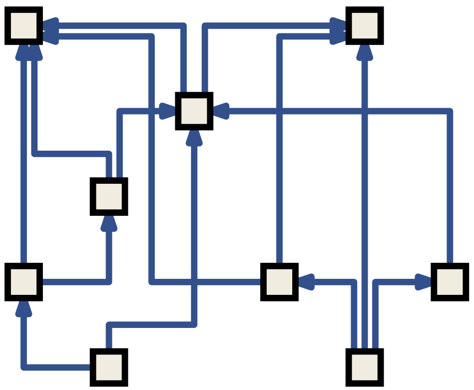
Drawing Directed Graphs

There is a variety of drawing styles for directed graphs, e.g.



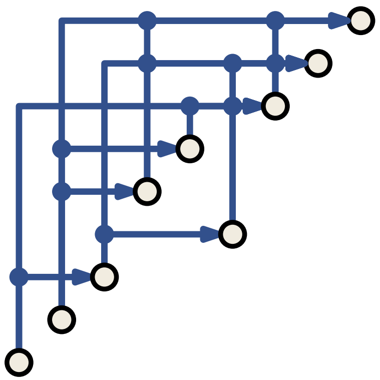
Layered layout

[Sugiyama, Tagawa, Toda 1981]



Kandinsky style layout

[Fößmeier, Kaufmann 1996]

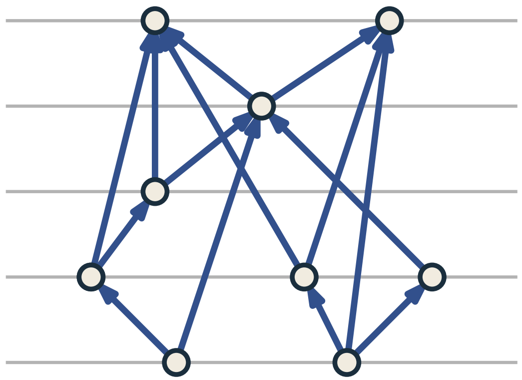


Overloaded orthogonal layout

[Kornaropoulos, Tollis 2011]

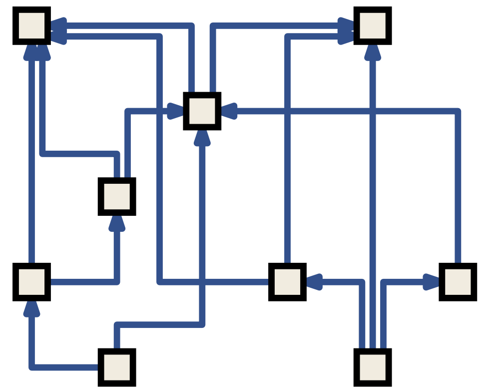
Drawing Directed Graphs

There is a variety of drawing styles for directed graphs, e.g.



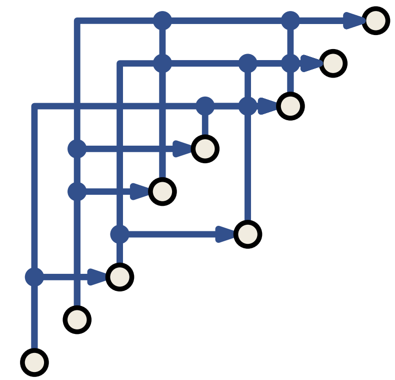
Layered layout

[Sugiyama, Tagawa, Toda 1981]



Kandinsky style layout

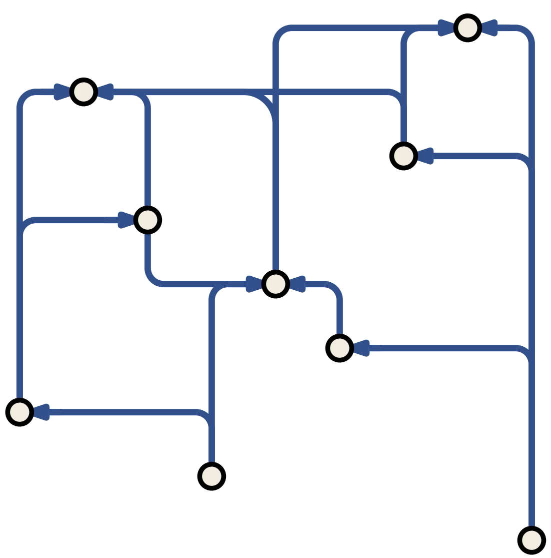
[Fößmeier, Kaufmann 1996]



Overloaded orthogonal layout

[Kornaropoulos, Tollis 2011]

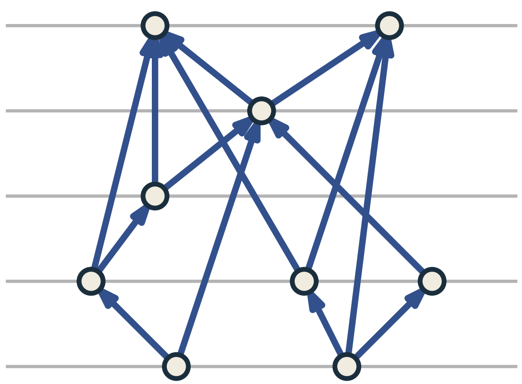
In 2016 Angelini et al. introduced **L-drawings**:



- exclusive x- and y-coordinates per vertex
- outgoing edges attach vertically
- incoming edges attach horizontally
- small arcs indicate L-bends
- crossings and “confluent” overlaps allowed
- exist for any graph
- ink minimization is NP-hard

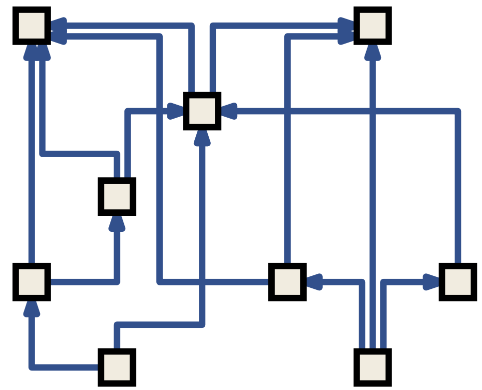
Drawing Directed Graphs

There is a variety of drawing styles for directed graphs, e.g.



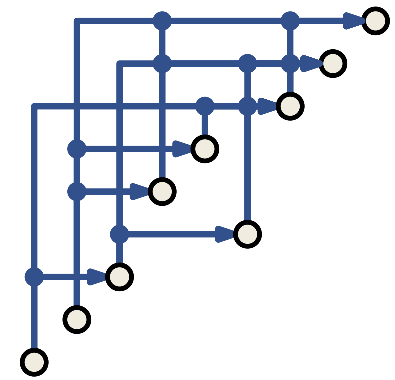
Layered layout

[Sugiyama, Tagawa, Toda 1981]



Kandinsky style layout

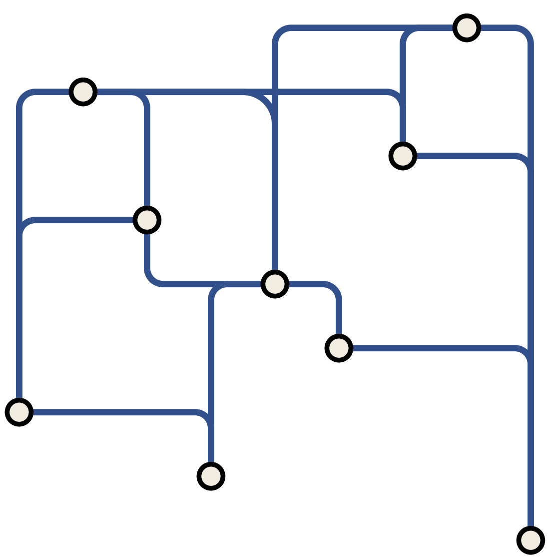
[Fößmeier, Kaufmann 1996]



Overloaded orthogonal layout

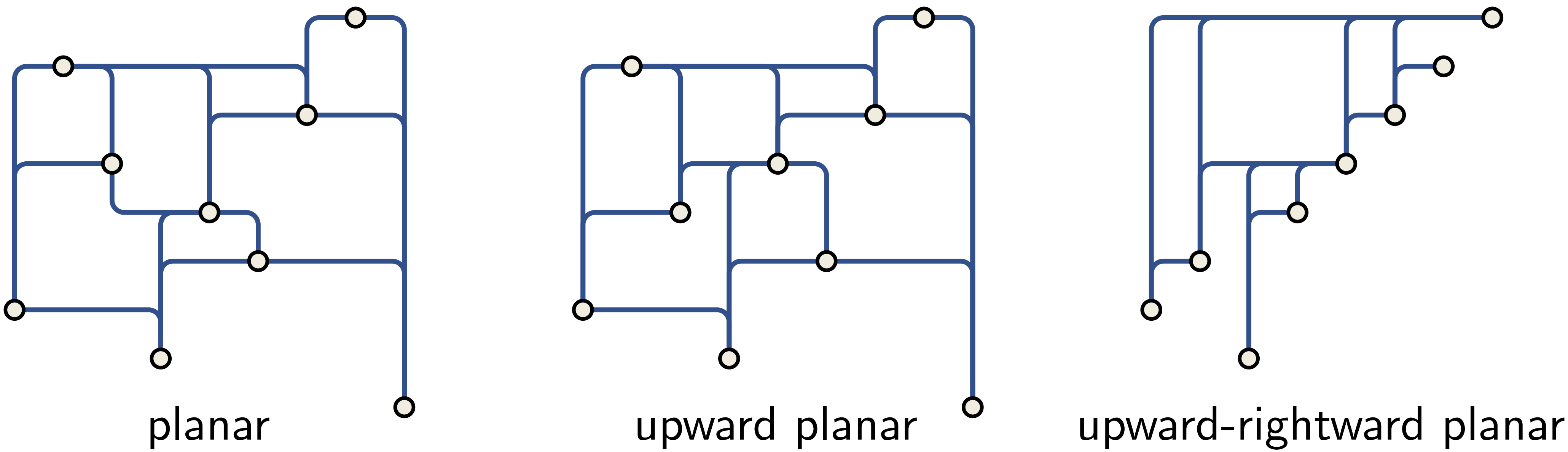
[Kornaropoulos, Tollis 2011]

In 2016 Angelini et al. introduced **L-drawings**:



- exclusive x- and y-coordinates per vertex
- outgoing edges attach vertically
- incoming edges attach horizontally
- small arcs indicate L-bends
- crossings and "confluent" overlaps allowed
- exist for any graph
- ink minimization is NP-hard

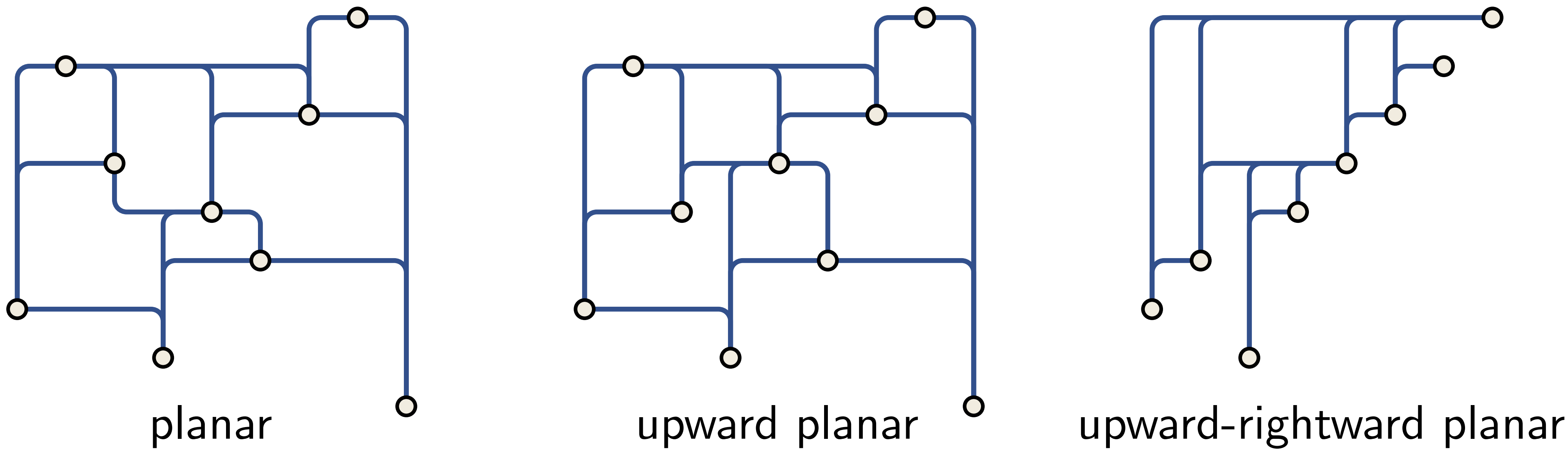
Planar L-Drawings



Definitions:

- **Planar L-drawing** if crossing-free
- **Upward planar L-drawing** if all edges y-increasing
- **Upward-rightward planar L-drawing** if all edges x- and y-increasing

Planar L-Drawings

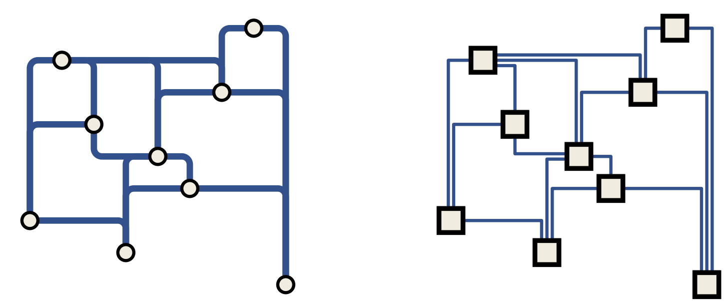


Definitions:

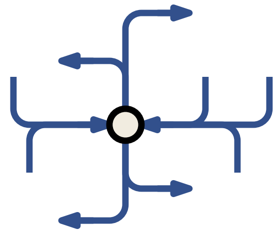
- **Planar L-drawing** if crossing-free
- **Upward planar L-drawing** if all edges y-increasing
- **Upward-rightward planar L-drawing** if all edges x- and y-increasing

Observation:

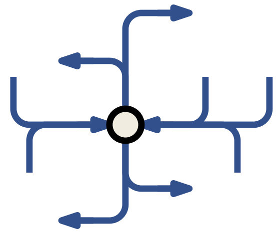
Planar L-drawings correspond to planar 1-bend Kandinsky drawings with extra constraints on cyclic edge orders of vertices.



Overview of Results

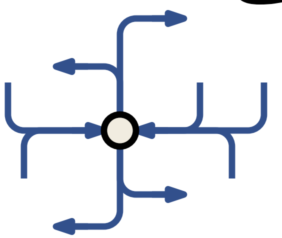
	 planar	 upward (-rightward) planar
directed planar graphs	NP-complete	
planar st-graphs		characterization constructive linear time algorithm
directed plane graphs + port assignment	linear time	

Overview of Results

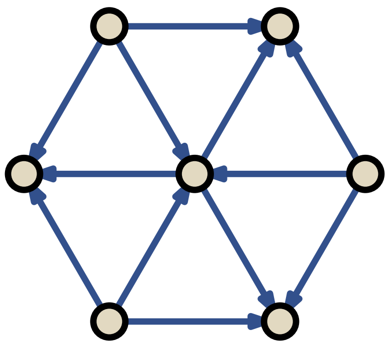
	 planar	 upward (-rightward) planar
directed planar graphs	NP-complete	
planar st-graphs		characterization constructive linear time algorithm
directed plane graphs + port assignment	linear time	

Planar L-Drawings of Directed Graphs

Any planar L-drawing implies 4-modal embedding.



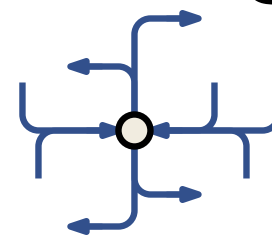
- There are planar directed graphs that do not admit planar L-drawings.



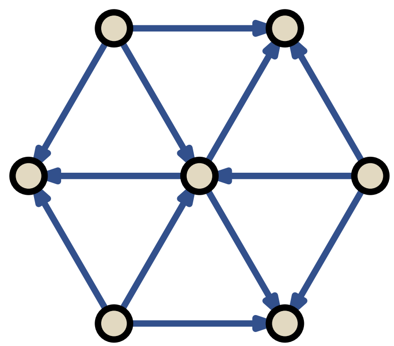
6-modal in any embedding

Planar L-Drawings of Directed Graphs

Any planar L-drawing implies 4-modal embedding.

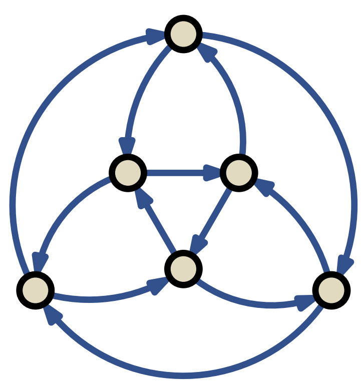


- There are planar directed graphs that do not admit planar L-drawings.



6-modal in any embedding

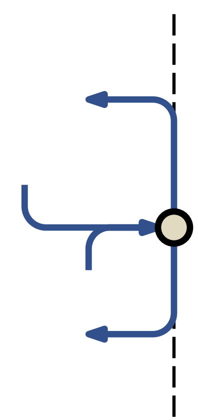
- There are graphs with 4-modal embedding but no planar L-drawing.



octahedron

every vertex is 4-modal . . .

. . . but rightmost vertex in L-drawing can be at most bimodal



NP-Completeness

Theorem: Deciding whether a directed graph admits a planar L-drawing is NP-complete.

Proof: (sketch)

- reduction from NP-complete **HV-rectilinear planarity testing**

[Didimo, Liotta, Patrignani 2014]

Given biconnected degree-4 planar graph G with edges labeled H and V, decide if G admits drawing with horizonal H-edges and vertical V-edges.

NP-Completeness

Theorem: Deciding whether a directed graph admits a planar L-drawing is NP-complete.

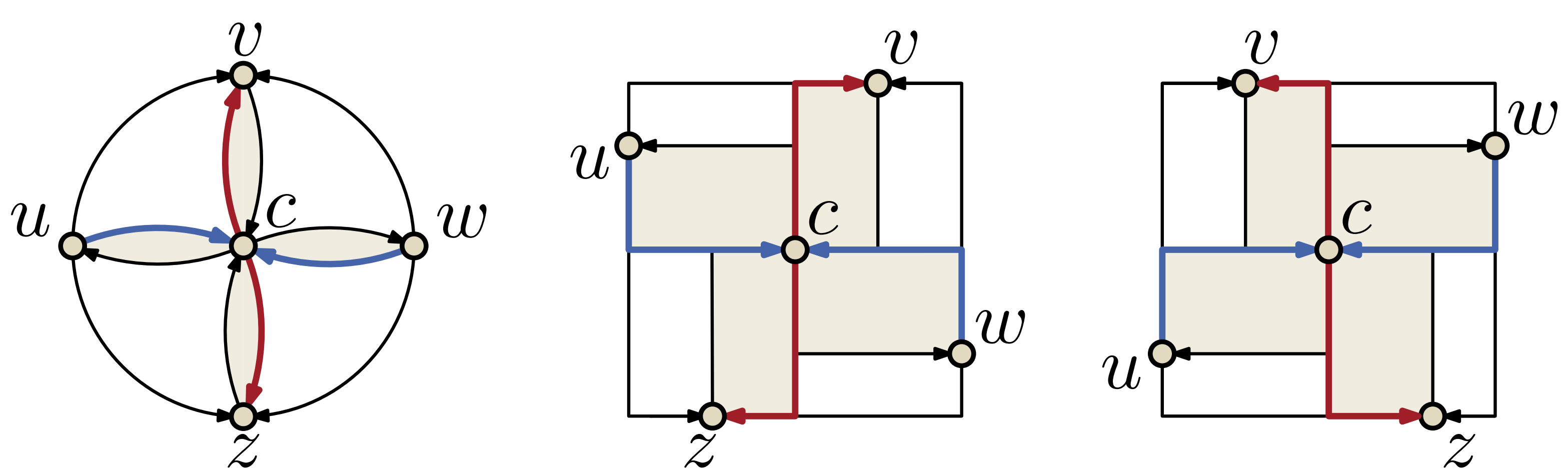
Proof: (sketch)

- reduction from NP-complete **HV-rectilinear planarity testing**

[Didimo, Liotta, Patrignani 2014]

> Given biconnected degree-4 planar graph G with edges labeled H and V, decide if G admits drawing with horizonal H-edges and vertical V-edges.

- core gadget: 4-wheel graph has basically two planar L-embeddings



NP-Completeness

Theorem: Deciding whether a directed graph admits a planar L-drawing is NP-complete.

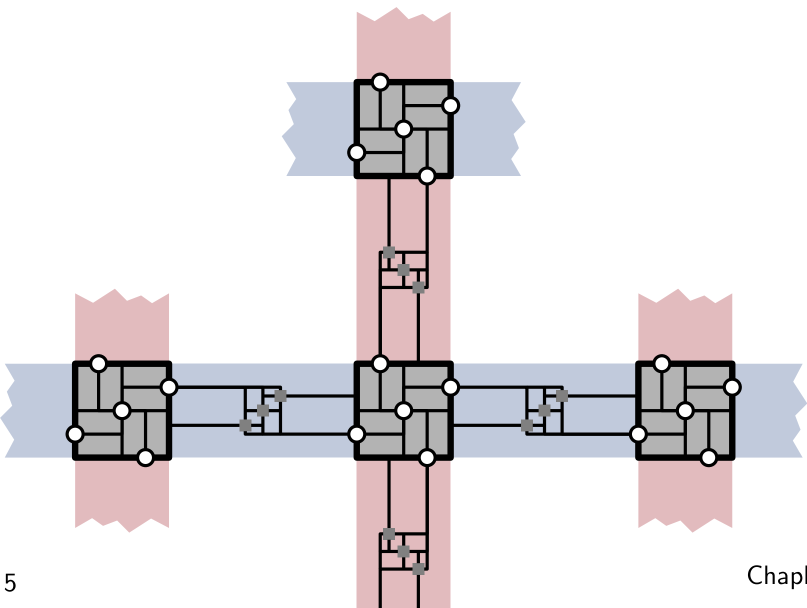
Proof: (sketch)

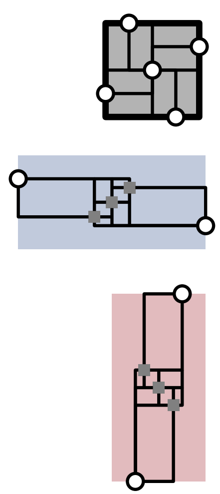
- reduction from NP-complete **HV-rectilinear planarity testing**

[Didimo, Liotta, Patrignani 2014]

> Given biconnected degree-4 planar graph G with edges labeled H and V, decide if G admits drawing with horizonal H-edges and vertical V-edges.

- core gadget: 4-wheel graph has basically two planar L-embeddings
- in HV-graph G replace vertices by 4-wheel and edges by H-/V-gadgets





4-wheel

H-gadget

V-gadget

NP-Completeness

Theorem: Deciding whether a directed graph admits a planar L-drawing is NP-complete.

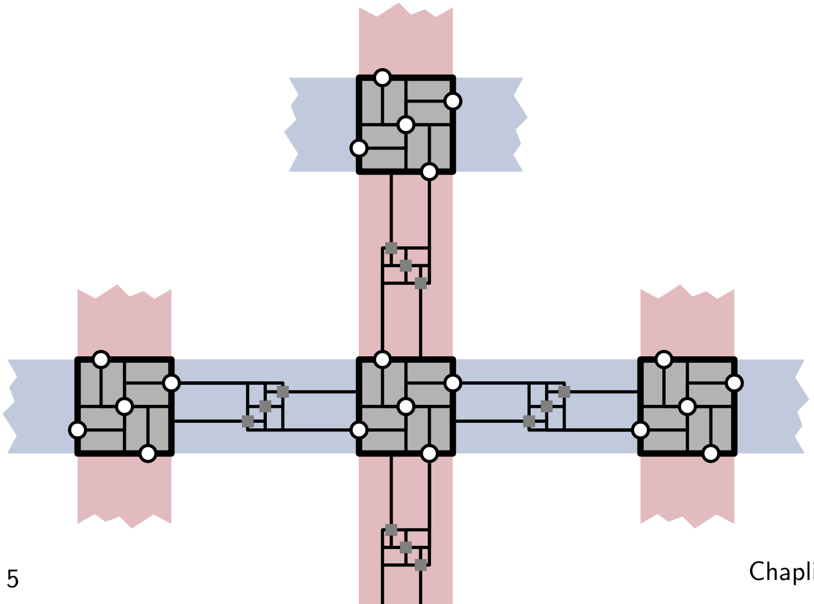
Proof: (sketch)

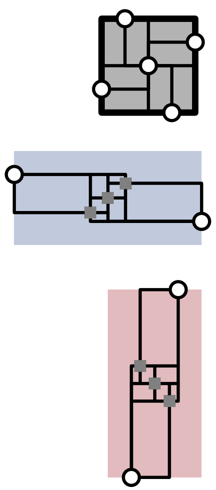
- reduction from NP-complete **HV-rectilinear planarity testing**

[Didimo, Liotta, Patrignani 2014]

> Given biconnected degree-4 planar graph G with edges labeled H and V, decide if G admits drawing with horizonal H-edges and vertical V-edges.

- core gadget: 4-wheel graph has basically two planar L-embeddings
- in HV-graph G replace vertices by 4-wheel and edges by H-/V-gadgets





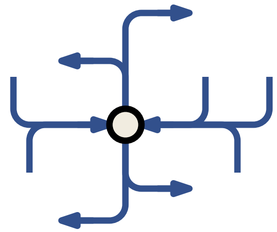
resulting graph G' has planar L-drawing

$\Leftrightarrow$

G has HV-drawing

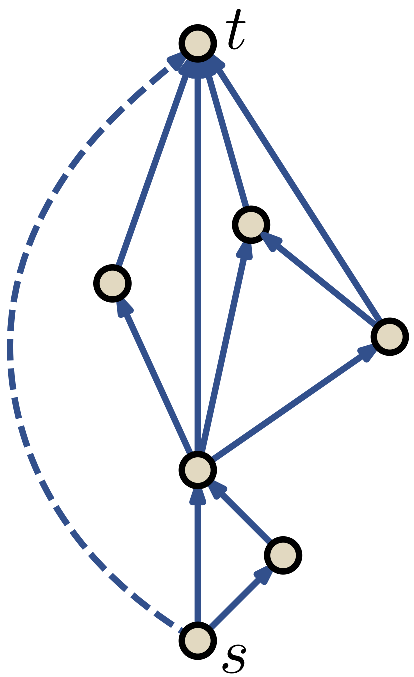
□

Overview of Results

	 planar	 upward (-rightward) planar
directed planar graphs	NP-complete	
planar st-graphs		characterization constructive linear time algorithm
directed plane graphs + port assignment	linear time	

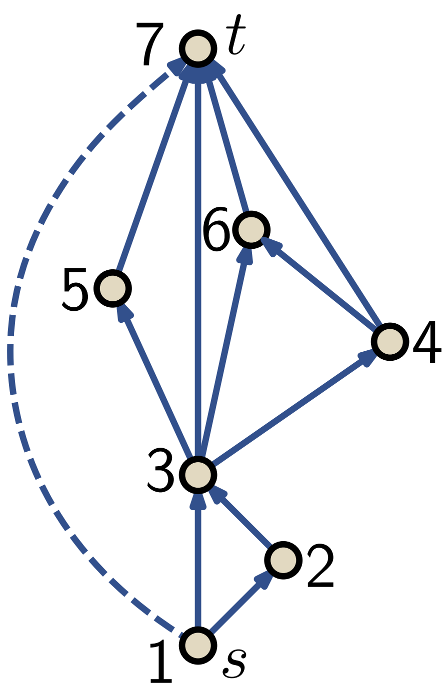
Planar st-Graphs and Bitonic st-Orderings

- A **planar st-graph** G is a directed acyclic graph with exactly one source s and one sink t, both embeddable on same face.



Planar st-Graphs and Bitonic st-Orderings

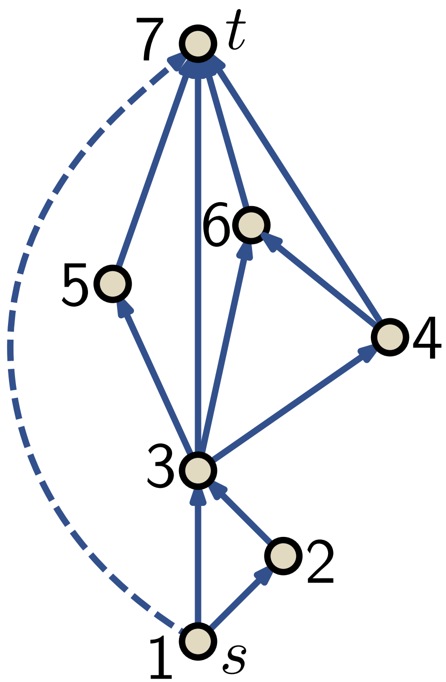
- A **planar st-graph** G is a directed acyclic graph with exactly one source s and one sink t, both embeddable on same face.



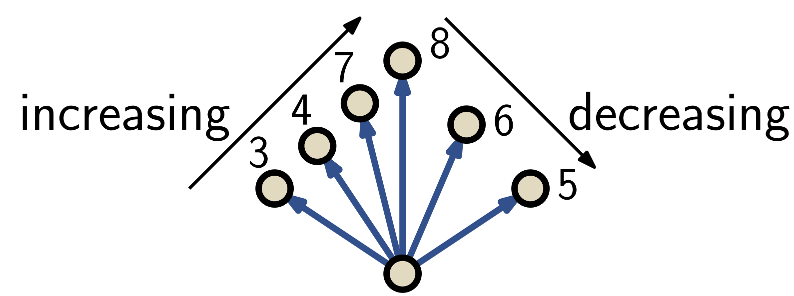
- planar st-graphs always admit straight-line upward planar drawings [Di Battista, Tamassia 1988]
- have **st-ordering** π respecting edge directions

Planar st-Graphs and Bitonic st-Orderings

- A **planar st-graph** G is a directed acyclic graph with exactly one source s and one sink t, both embeddable on same face.

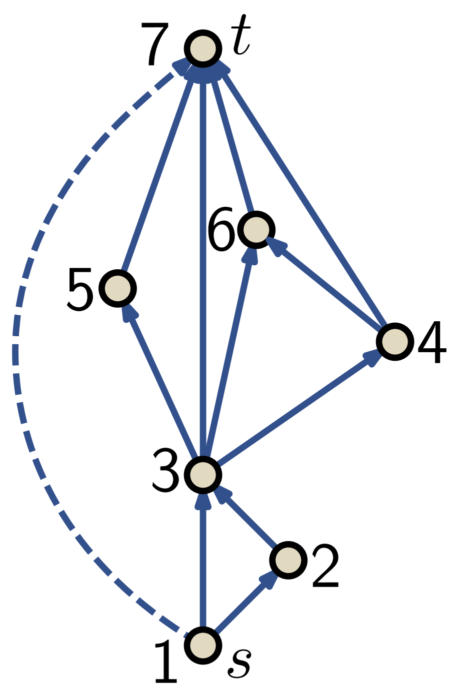


- planar st-graphs always admit straight-line upward planar drawings [Di Battista, Tamassia 1988]
- have **st-ordering** π respecting edge directions
- st-ordering π of plane st-graph (planar st-graph $+$ embedding) is **bitonic** if successors of each vertex form bitonic sequence [Gronemann 2014, 2016]

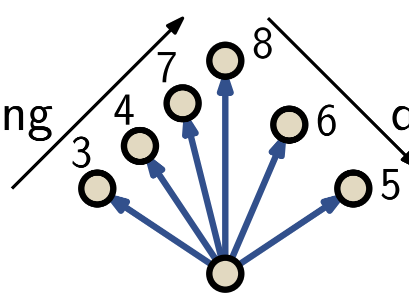


Planar st-Graphs and Bitonic st-Orderings

- A **planar st-graph** G is a directed acyclic graph with exactly one source s and one sink t, both embeddable on same face.



- planar st-graphs always admit straight-line upward planar drawings [Di Battista, Tamassia 1988]
- have **st-ordering** π respecting edge directions
- st-ordering π of plane st-graph (planar st-graph + embedding) is **bitonic** if successors of each vertex form bitonic sequence [Gronemann 2014, 2016]



- For a planar st-graph G define a **bitonic pair** $(\mathcal{E}, \pi)$ as an upward planar embedding $\mathcal{E}$ of G with a bitonic st-ordering π.

Characterization

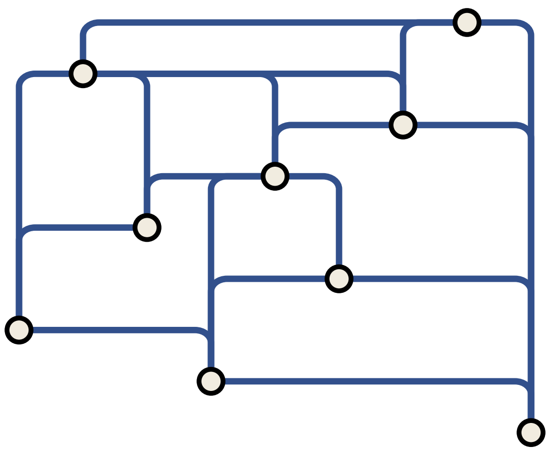
Theorem: A planar st-graph admits an upward-planar L-drawing if and only if it admits a bitonic pair.

Characterization

Theorem: A planar st-graph admits an upward-planar L-drawing if and only if it admits a bitonic pair.

Proof: (sketch)

$\Rightarrow$

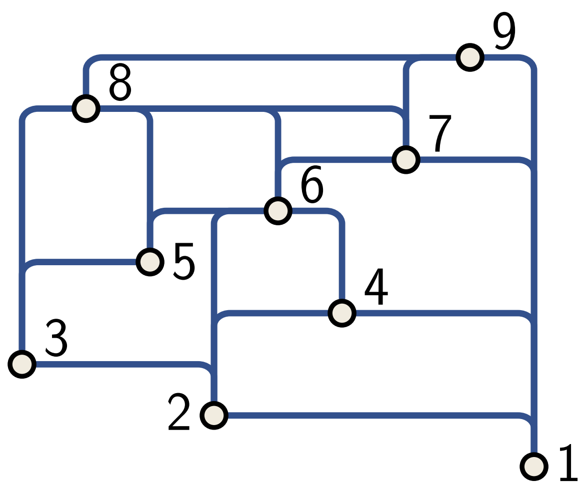


Characterization

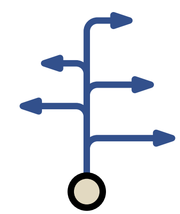
Theorem: A planar st-graph admits an upward-planar L-drawing if and only if it admits a bitonic pair.

Proof: (sketch)

⇒



- y-coordinates induce st-ordering π
- π is bitonic due to upward L-properties

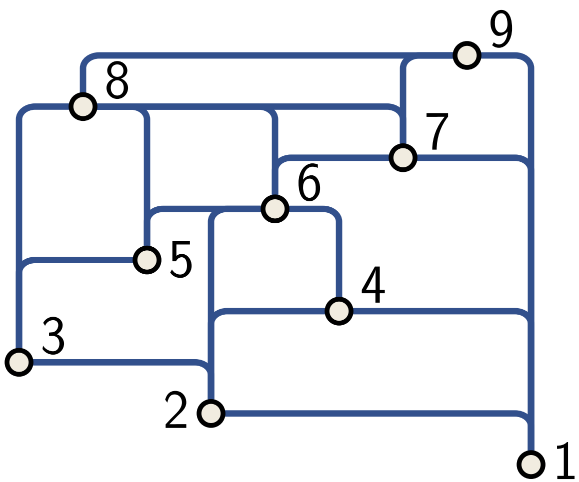


Characterization

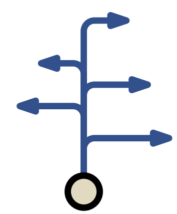
Theorem: A planar st-graph admits an upward-planar L-drawing if and only if it admits a bitonic pair.

Proof: (sketch)

$\Rightarrow$

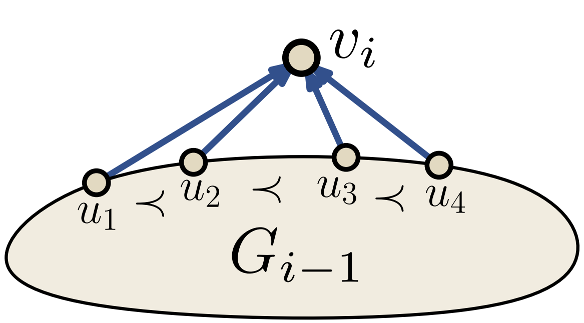


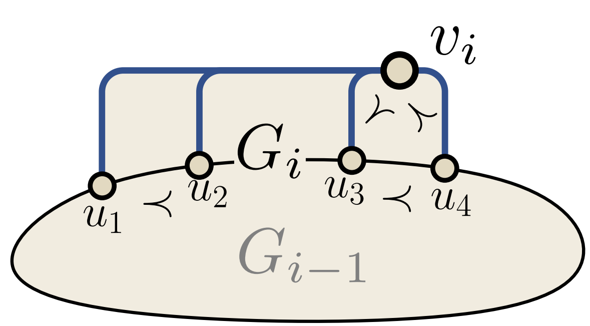
- y-coordinates induce st-ordering π
- π is bitonic due to upward L-properties



$\Leftarrow$

- use π for y-coordinates
- incrementally construct partial order $\prec$ as basis for x-coordinates





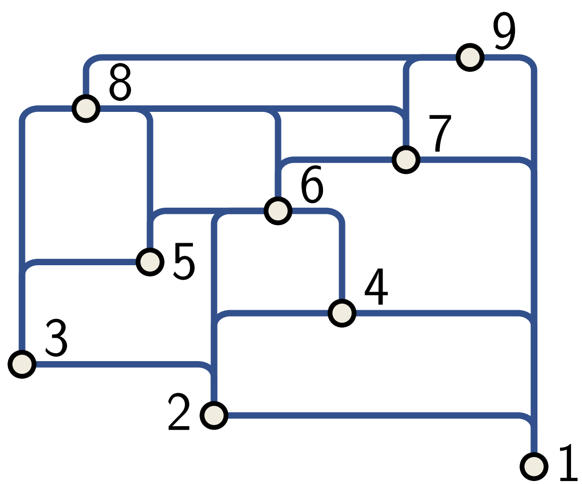
- invariant: outer face of G_i simple cycle ordered by $\prec$
- insert each v_i btw. last two predecessors

Characterization

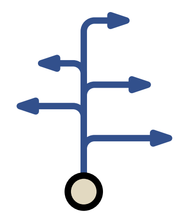
Theorem: A planar st-graph admits an upward-planar L-drawing if and only if it admits a bitonic pair.

Proof: (sketch)

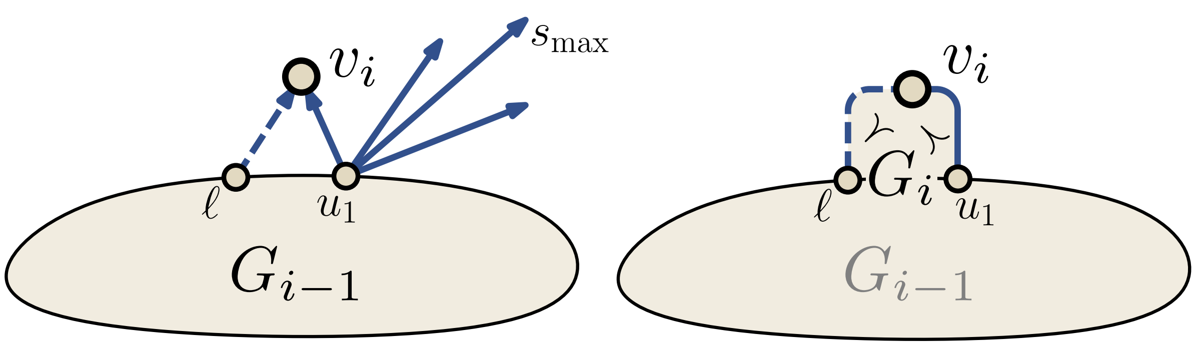
$\Rightarrow$



- y-coordinates induce st-ordering π
- π is bitonic due to upward L-properties



$\Leftarrow$
- use π for y-coordinates
- incrementally construct partial order $\prec$ as basis for x-coordinates

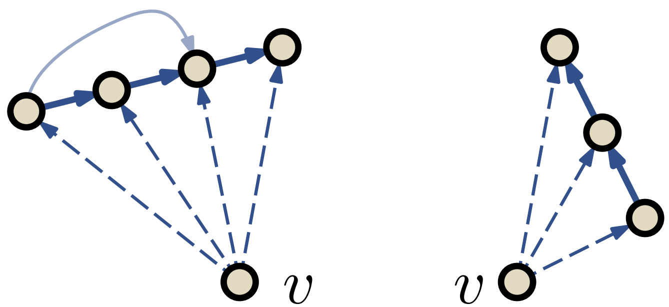


- invariant: outer face of G_i simple cycle ordered by $\prec$
- insert each v_i btw. last two predecessors

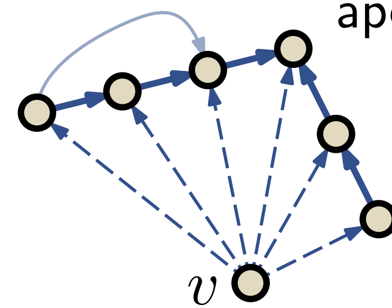
special case: just one predecessor $\rightarrow$ augment graph similar to [Gronemann 2016]

Finding Bitonic Pairs

- **Assumption:** st-graph G is biconnected and has edge (s, t)
- We say G is **v-monotonic** or **(strictly) v-bitonic** if for every vertex v
  - subgraph induced by successors of v ($-$ transitive edges) is a path p
  - p is **monotonic** or **(strictly) bitonic**



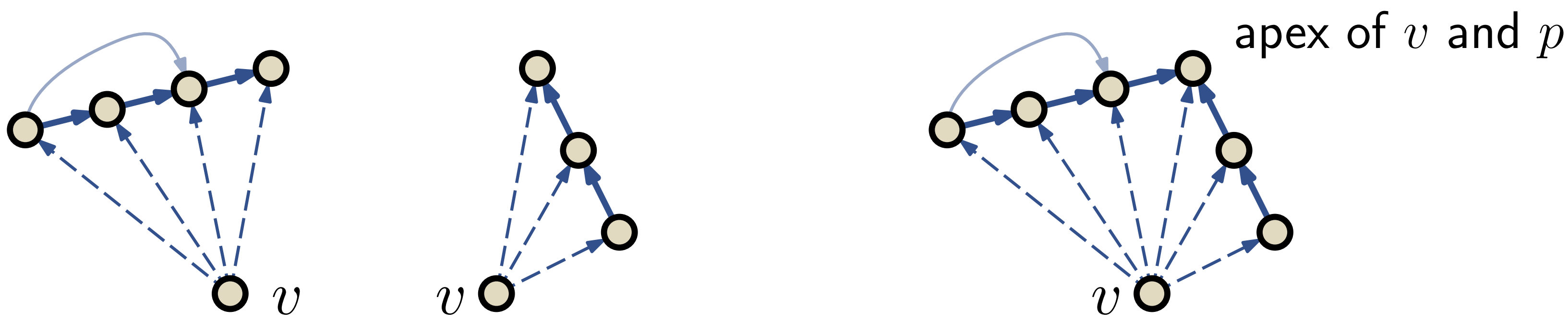
monotonic $\subset$ bitonic



strictly bitonic $\subset$ bitonic

Finding Bitonic Pairs

- **Assumption:** st-graph G is biconnected and has edge (s, t)
- We say G is v-**monotonic** or **(strictly)** v-**bitonic** if for every vertex v
  - subgraph induced by successors of v ($-$ transitive edges) is a path p
  - p is **monotonic** or **(strictly) bitonic**

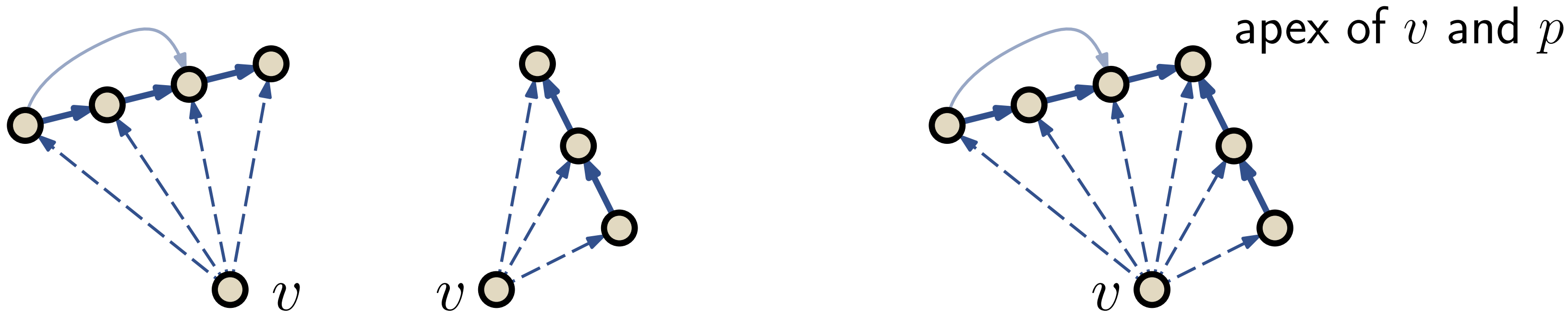


monotonic $\subset$ bitonic **strictly bitonic $\subset$ bitonic**

- **Goal:** augment G into G^* by adding edges s.t. G^* is v-bitonic

Finding Bitonic Pairs

- **Assumption:** st-graph G is biconnected and has edge (s, t)
- We say G is **v-monotonic** or **(strictly) v-bitonic** if for every vertex v
  - subgraph induced by successors of v ($-$ transitive edges) is a path p
  - p is **monotonic** or **(strictly) bitonic**



monotonic $\subset$ bitonic **strictly bitonic $\subset$ bitonic**

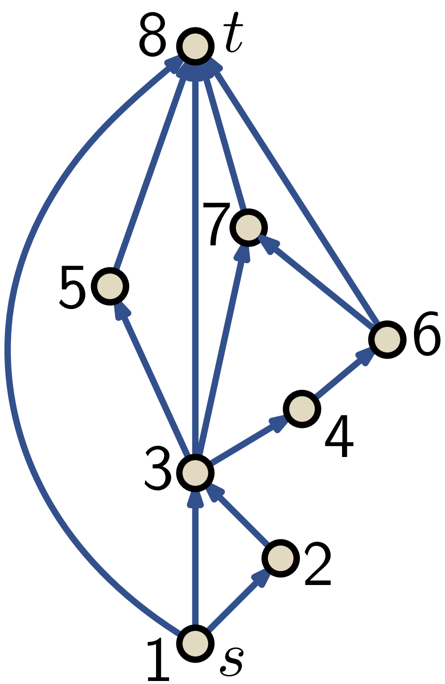
- **Goal:** augment G into G^* by adding edges s.t. G^* is v-bitonic

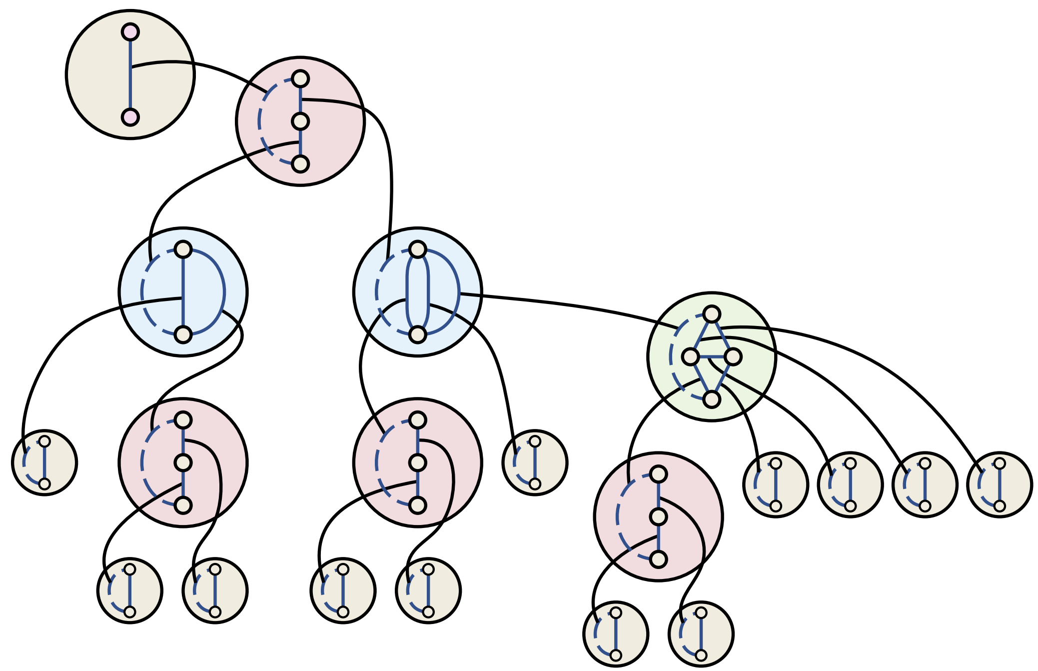
Theorem: Plane st-graph G admits bitonic st-ordering iff G^* is v-bitonic.
Any st-ordering of G^* is bitonic st-ordering of G. ~ [Gronemann 2016]

$\rightarrow$ The task of finding a bitonic pair of G reduces to finding an augmentation G^* of G that is v-bitonic.

Finding a v-Bitonic Augmentation

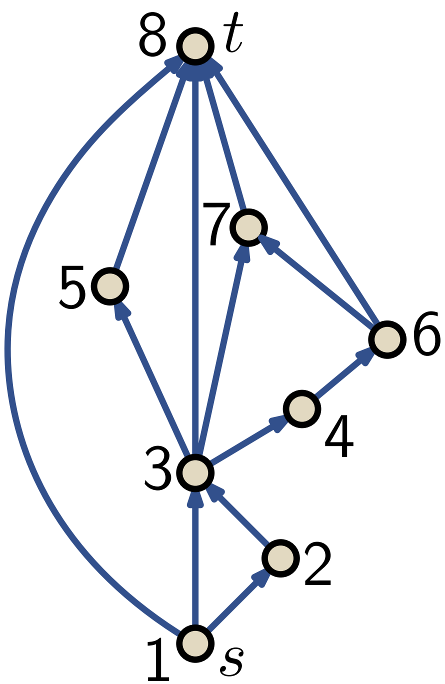
Visiting the SPQR-tree of biconnected planar st-graph G rooted at edge (s,t) in bottom-up fashion find augmentation G^* and embedding (if one exists).

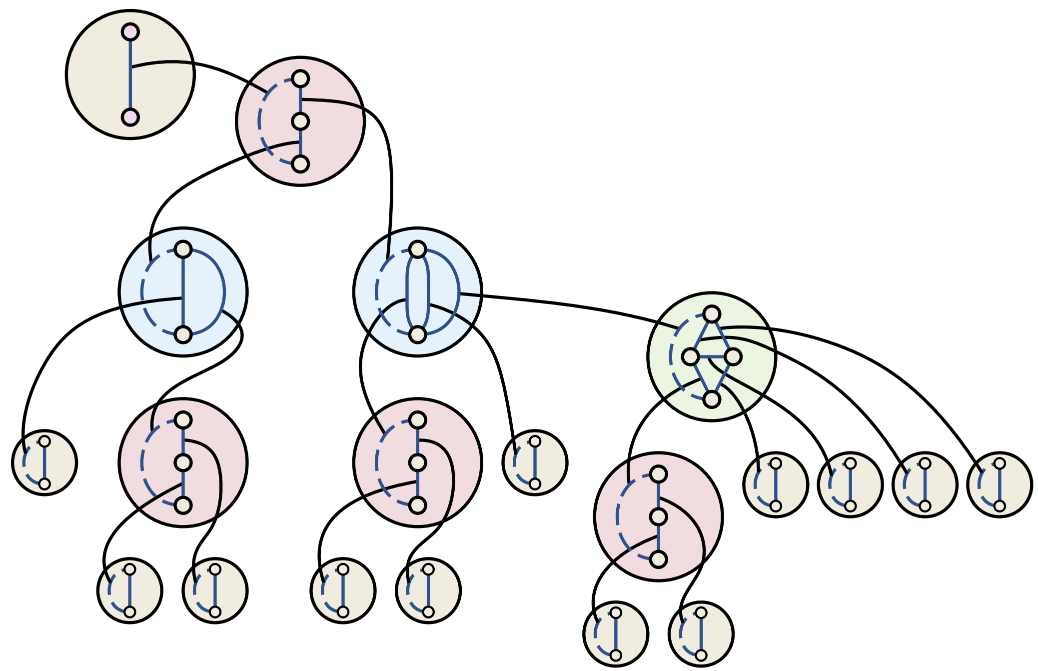




Finding a v-Bitonic Augmentation

Visiting the SPQR-tree of biconnected planar st-graph G rooted at edge (s,t) in bottom-up fashion find augmentation G^* and embedding (if one exists).



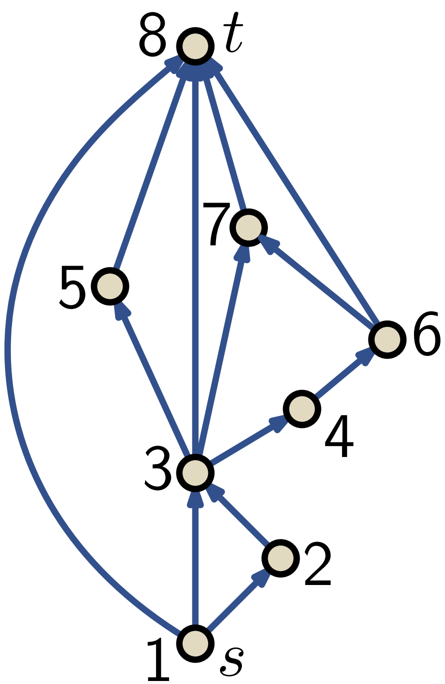


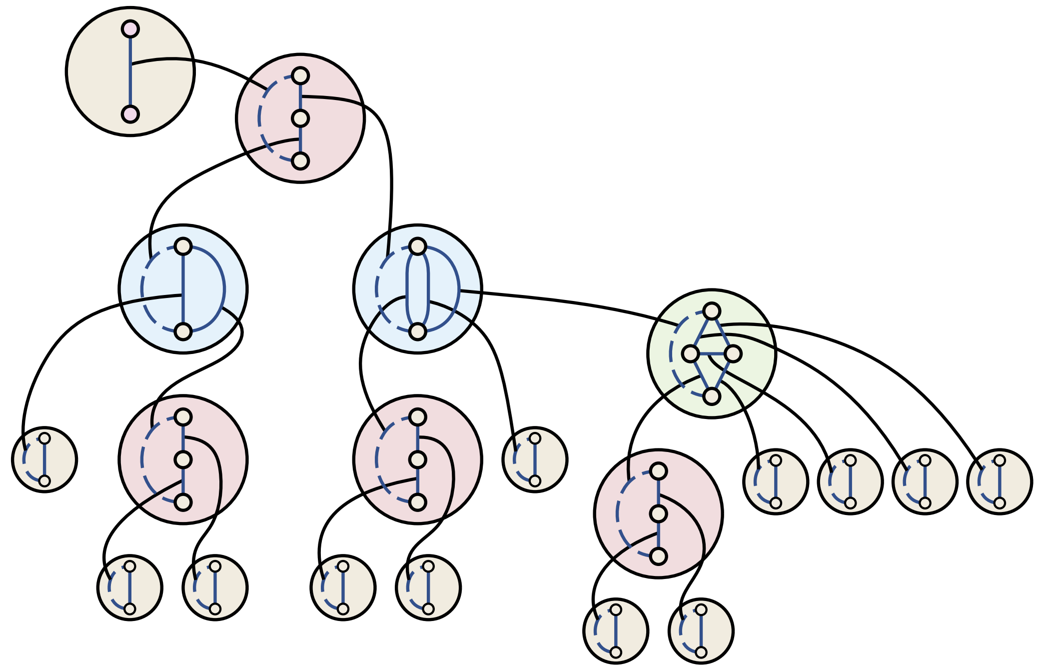
An SPQR-node μ with source s_μ is of

- **type M** if the augmented pertinent graph is s_μ-monotonic
- **type B** if the augmented pertinent graph is strictly s_μ-bitonic

Finding a v-Bitonic Augmentation

Visiting the SPQR-tree of biconnected planar st-graph G rooted at edge (s,t) in bottom-up fashion find augmentation G^* and embedding (if one exists).



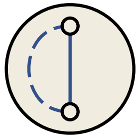


An SPQR-node μ with source s_μ is of

- **type M** if the augmented pertinent graph is s_μ-monotonic
- **type B** if the augmented pertinent graph is strictly s_μ-bitonic

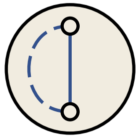
When processing an SPQR-node μ our primary goal is to make it type M and otherwise type B. If both fails, no v-bitonic augmentation of G exists.

Processing SPQR-Nodes

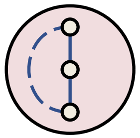


Q-node: trivially type M

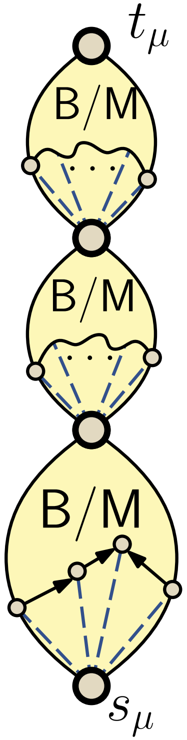
Processing SPQR-Nodes



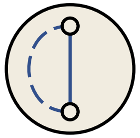
Q-node: trivially type M



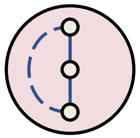
S-node: replace each virtual edge by augmented pertinent graph of child node with arbitrarily flipped embedding.
Node type is inherited from bottom child.



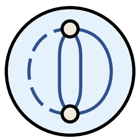
Processing SPQR-Nodes



Q-node: trivially type M

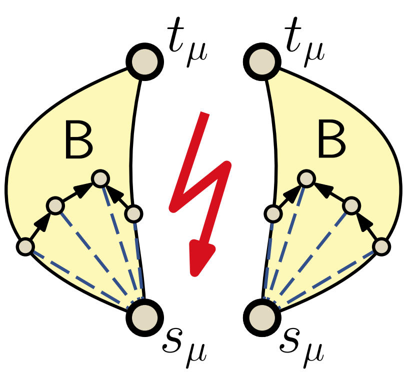


S-node: replace each virtual edge by augmented pertinent graph of child node with arbitrarily flipped embedding.
Node type is inherited from bottom child.



P-node:

- if two or more children are of type B
  $\rightarrow$ successors of s_μ have two apices, so regardless of embedding no s_μ-bitonic augmentation exists



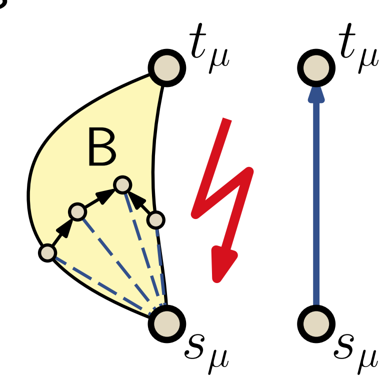
Processing SPQR-Nodes

Q-node: trivially type M

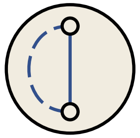
S-node: replace each virtual edge by augmented pertinent graph of child node with arbitrarily flipped embedding.
Node type is inherited from bottom child.

P-node:

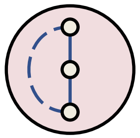
- if two or more children are of type B
  $\rightarrow$ successors of s_μ have two apices, so regardless of embedding no s_μ-bitonic augmentation exists
- if one child is of type B and one child is Q-node for (s_μ, t_μ)
  $\rightarrow$ apex of type-B node $\neq t_\mu$, but t_μ must be apex of s_μ;
  again s_μ has two apices and no s_μ-bitonic augmentation exists



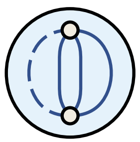
Processing SPQR-Nodes



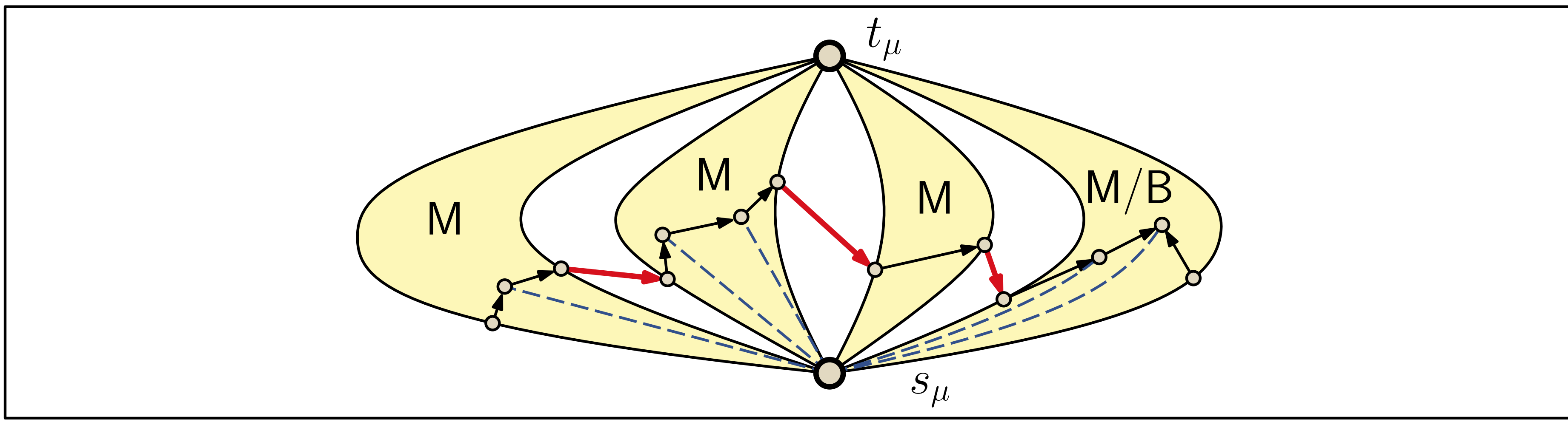
Q-node: trivially type M



S-node: replace each virtual edge by augmented pertinent graph of child node with arbitrarily flipped embedding.
Node type is inherited from bottom child.

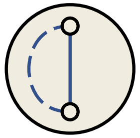


P-node:

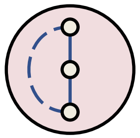


- else embed child of type B or Q-node for (s_μ, t_μ) rightmost (if any) and connect successors of s_μ in order of embedding; node type is M $\Leftrightarrow$ rightmost child is of type M

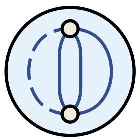
Processing SPQR-Nodes



Q-node: trivially type M

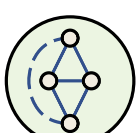


S-node: replace each virtual edge by augmented pertinent graph of child node with arbitrarily flipped embedding.
Node type is inherited from bottom child.



P-node:

- if two or more children are of type B
  $\rightarrow$ successors of s_μ have two apices, so regardless of embedding no s_μ-bitonic augmentation exists
- if one child is of type B and one child is Q-node for (s_μ, t_μ)
  $\rightarrow$ apex of type-B node $\neq t_\mu$, but t_μ must be apex of s_μ; again s_μ has two apices and no s_μ-bitonic augmentation exists
- else embed child of type B or Q-node for (s_μ, t_μ) rightmost (if any) and connect successors of s_μ in order of embedding; node type is M $\Leftrightarrow$ rightmost child is of type M



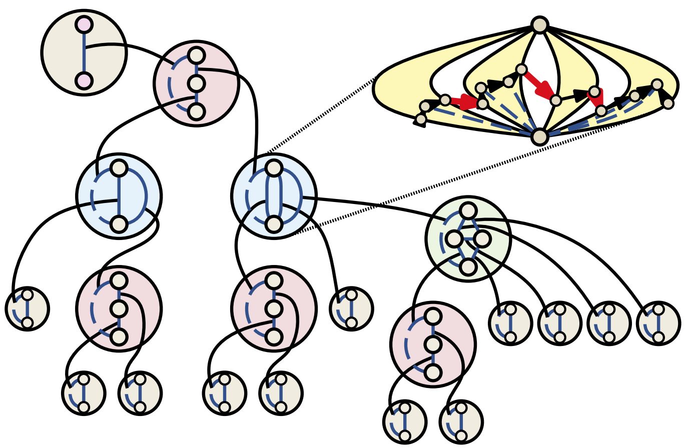
R-node: more complicated, see paper

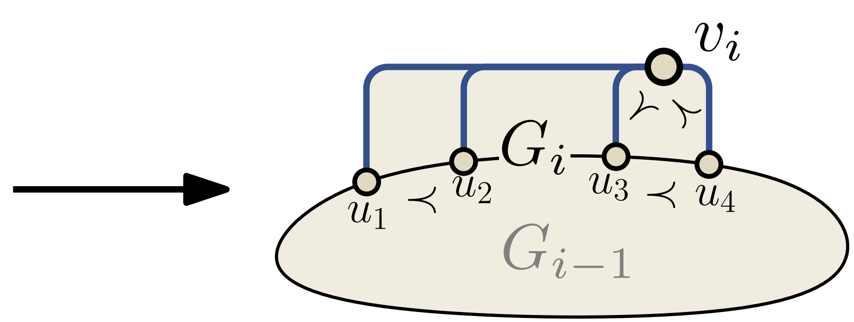
Upward Planar L-Drawings

Theorem: It can be tested in linear time whether a planar st-graph G admits an upward-planar L-drawing. If it does, it can also be constructed in linear time.

Proof: (sketch)

- process SPQR-tree to find v-bitonic augmentation G^* and embedding $\mathcal{E}^*$ in root node (if any)
- any st-ordering π of G^* yields bitonic pair $(\mathcal{E}, \pi)$ of G
- G has bitonic pair $\Leftrightarrow$ G admits upward-planar L-drawing
- all steps can be implemented in linear time





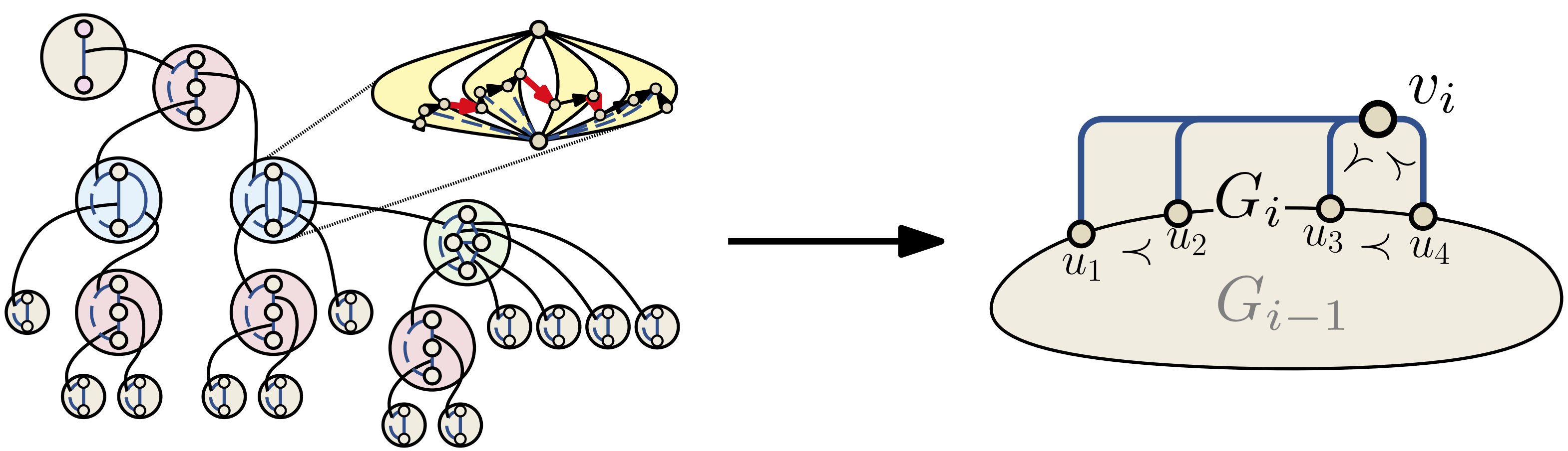
□

Upward Planar L-Drawings

Theorem: It can be tested in linear time whether a planar st-graph G admits an upward-planar L-drawing.
If it does, it can also be constructed in linear time.

Proof: (sketch)

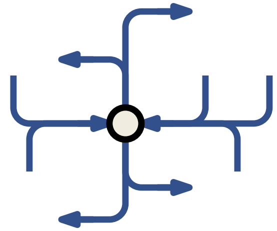
- process SPQR-tree to find v-bitonic augmentation G^* and embedding $\mathcal{E}^*$ in root node (if any)
- any st-ordering π of G^* yields bitonic pair $(\mathcal{E}, \pi)$ of G
- G has bitonic pair $\Leftrightarrow$ G admits upward-planar L-drawing
- all steps can be implemented in linear time



□

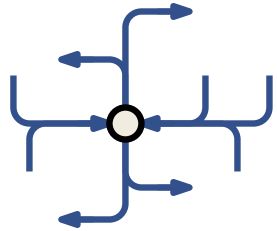
Remark: Same approach can be used to decide existence and construct upward-rightward-planar L-drawings.

Summary



	planar	upward (-rightward) planar
directed planar graphs	NP-complete	
planar st-graphs		characterization constructive linear time algorithm
directed plane graphs + port assignment	linear time $\rightarrow$ see paper	
directed plane graphs upward planar graphs bimodal graphs		

Summary

	 planar	 upward (-rightward) planar
directed planar graphs	NP-complete	
planar st-graphs		characterization constructive linear time algorithm
directed plane graphs + port assignment	linear time $\rightarrow$ see paper	
directed plane graphs upward planar graphs bimodal graphs	**?**	