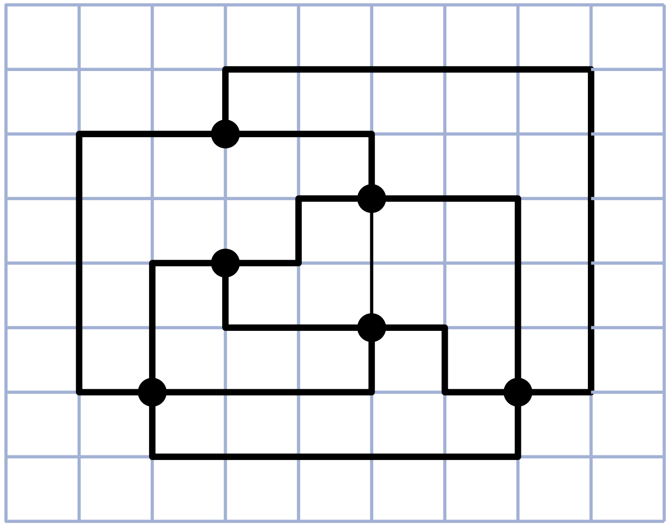
Karlsruhe Institute of Technology

Algorithms for graph visualization

Incremental algorithms. Orthogonal drawing.

WINTER SEMESTER 2013/2014

Tamara Mchedlidze – MARTIN NÖLLENBURG



Definition

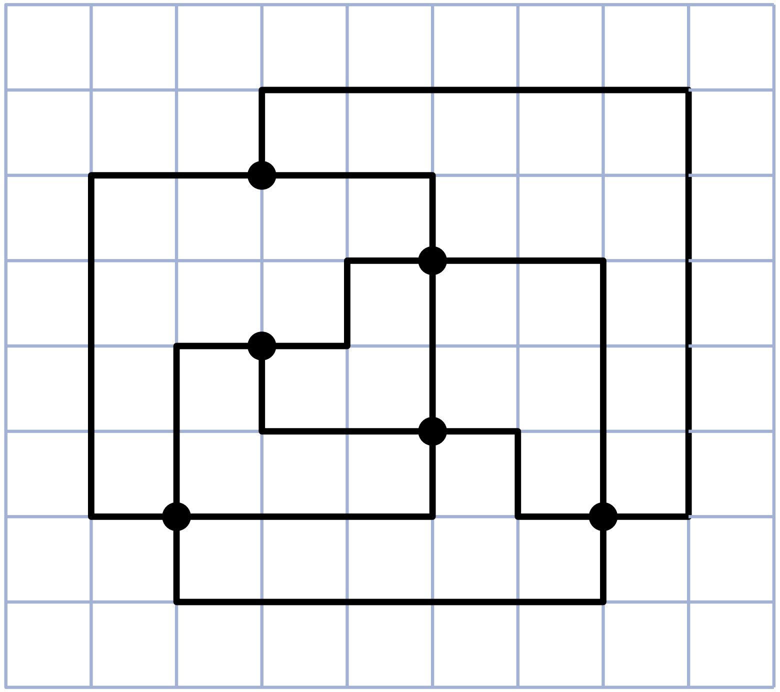
Definition: Orthogonal Drawing

A drawing Γ of a graph $G = (V, E)$ is called orthogonal if its veritices are drawn as points and each edge is represented as a sequence of alternating horizontal and vertical segments.

Definition

Definition: Orthogonal Drawing

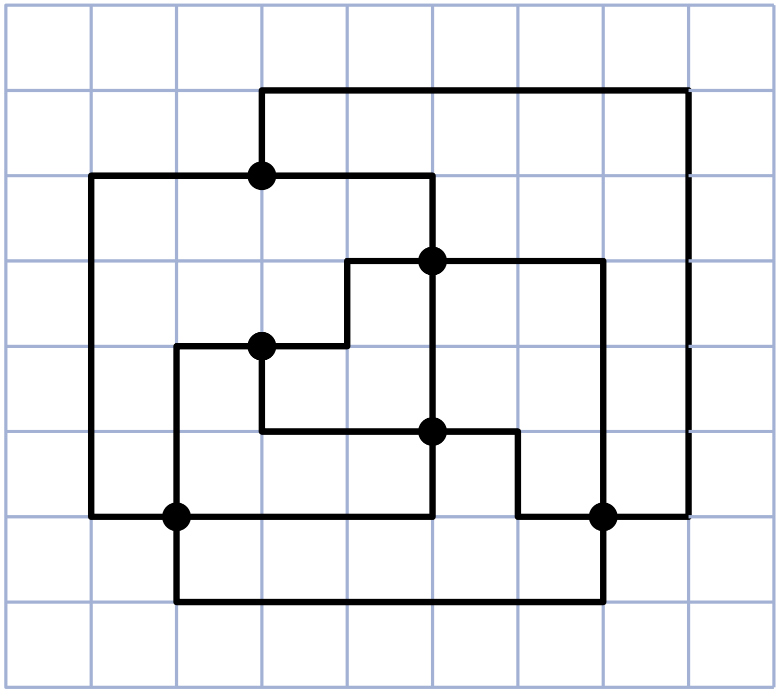
A drawing Γ of a graph $G = (V, E)$ is called orthogonal if its veritices are drawn as points and each edge is represented as a sequence of alternating horizontal and vertical segments.



Definition

Definition: Orthogonal Drawing

A drawing Γ of a graph $G = (V, E)$ is called orthogonal if its veritices are drawn as points and each edge is represented as a sequence of alternating horizontal and vertical segments.

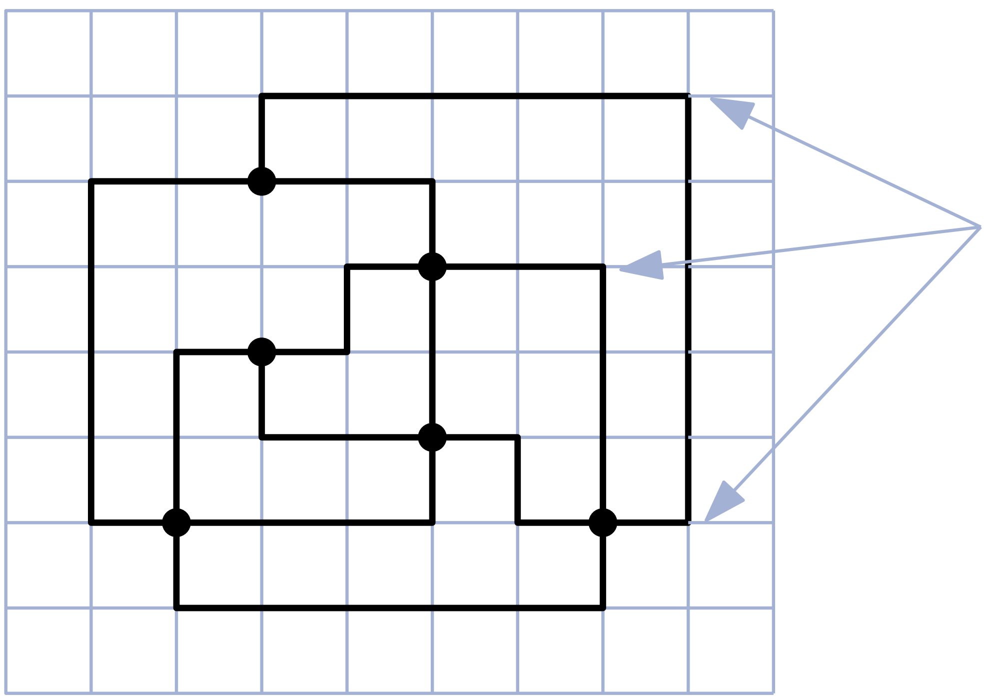


- degree of each vertex has to be at most 4

Definition

Definition: Orthogonal Drawing

A drawing Γ of a graph $G = (V, E)$ is called orthogonal if its veritices are drawn as points and each edge is represented as a sequence of alternating horizontal and vertical segments.



- degree of each vertex has to be at most 4
- bends on edges

st-ordering

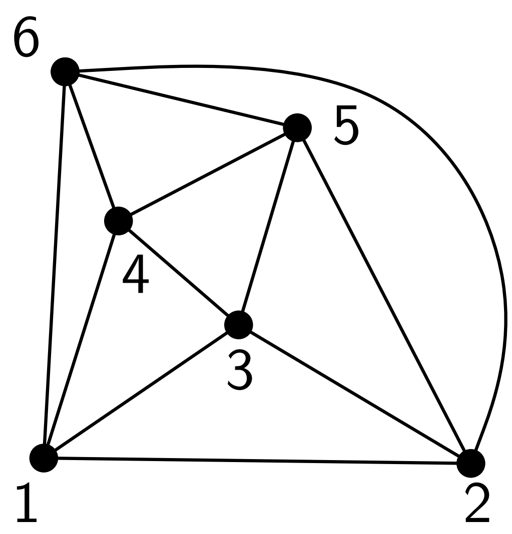
Definition: st-ordering

An st-ordering of a graph $G = (V, E)$ is an ordering of the vertices $\{v_1, v_2, \ldots, v_n\}$, such that for each j, $2 \leq j \leq n - 1$, vertex v_j has at least one neighbour v_i with $i < j$, and at least one neighbour v_k with $k > j$.

st-ordering

Definition: st-ordering

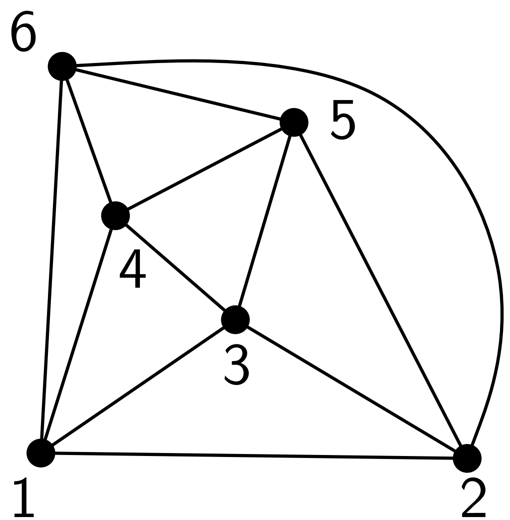
An st-ordering of a graph $G = (V, E)$ is an ordering of the vertices $\{v_1, v_2, \ldots, v_n\}$, such that for each j, $2 \leq j \leq n-1$, vertex v_j has at least one neighbour v_i with $i < j$, and at least one neighbour v_k with $k > j$.

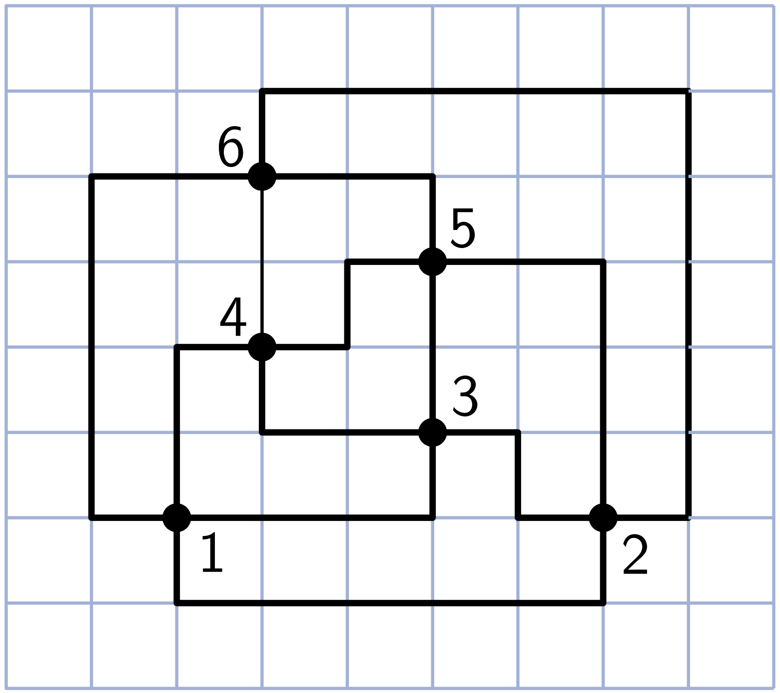


Theorem [Lempel, Even, Cederbaum, 66]

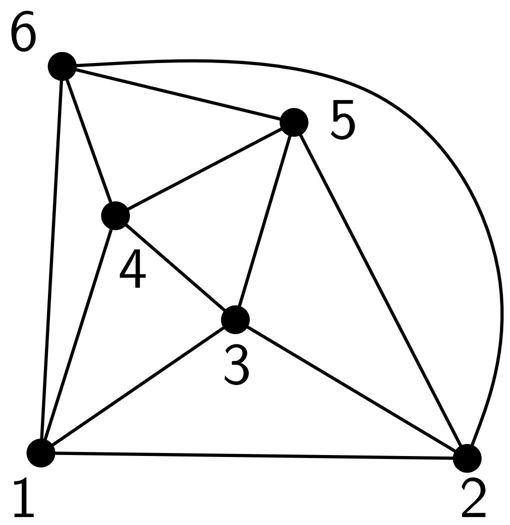
Let G be a biconnected graph G and let s, t be vetices of G. G has an st-ordering such that s appears as the first and t as the last vertex in this ordering.

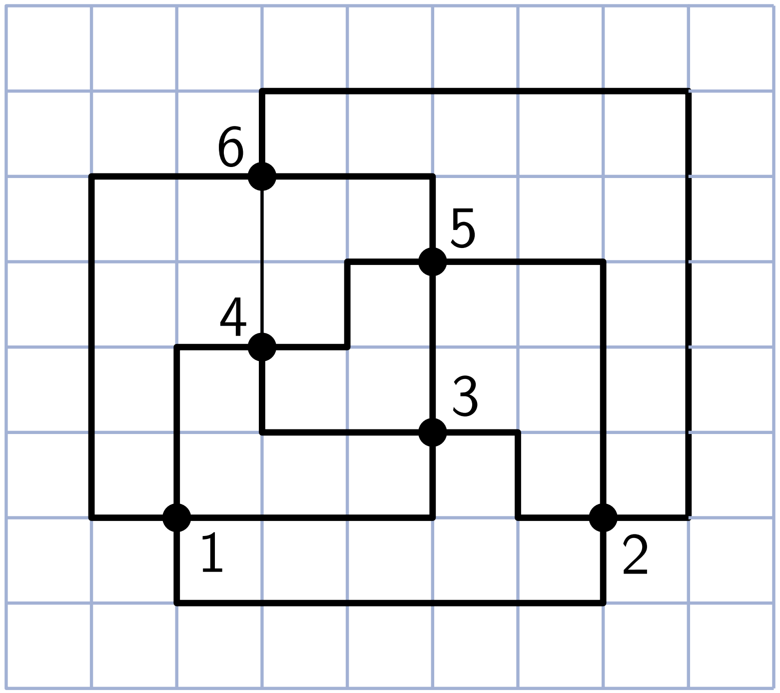
Biedl & Kant Orthogonal Drawing Algorithm



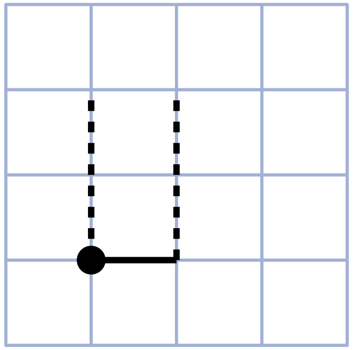


Biedl & Kant Orthogonal Drawing Algorithm

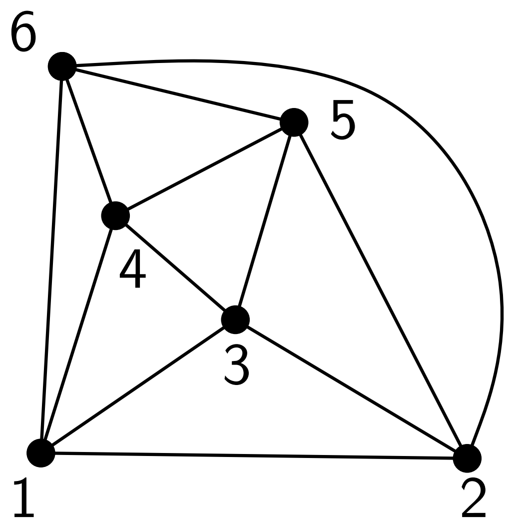


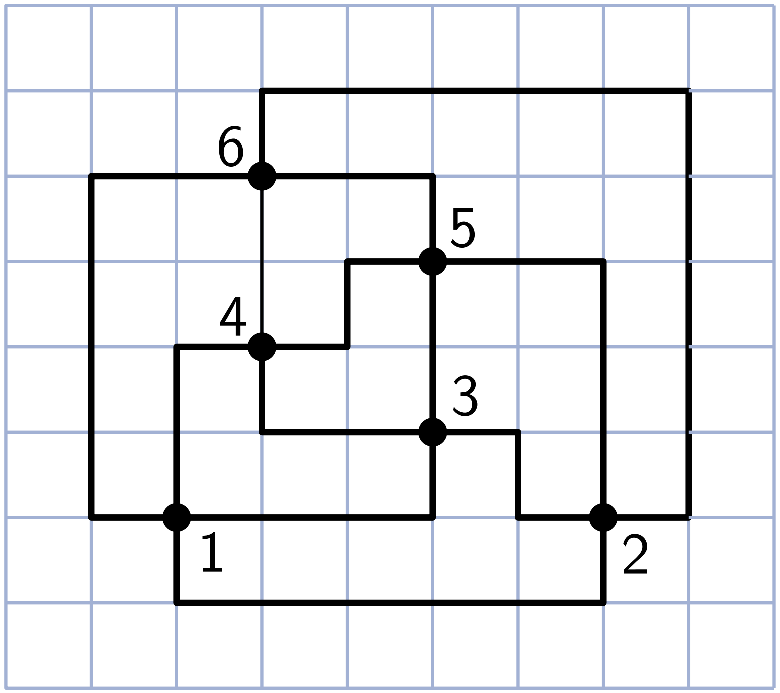


first vertex

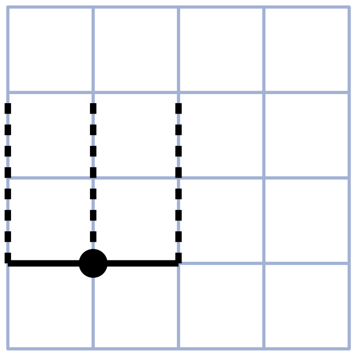


Biedl & Kant Orthogonal Drawing Algorithm

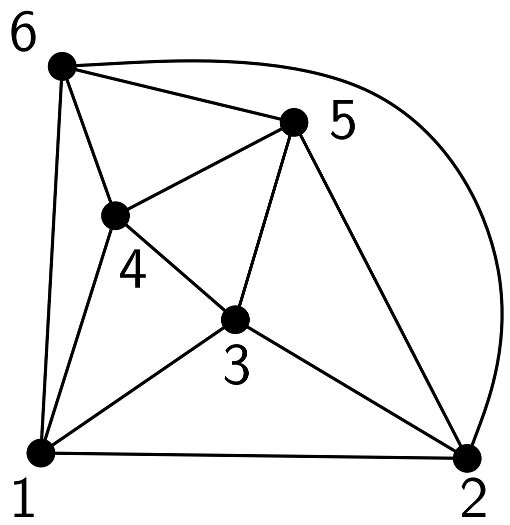


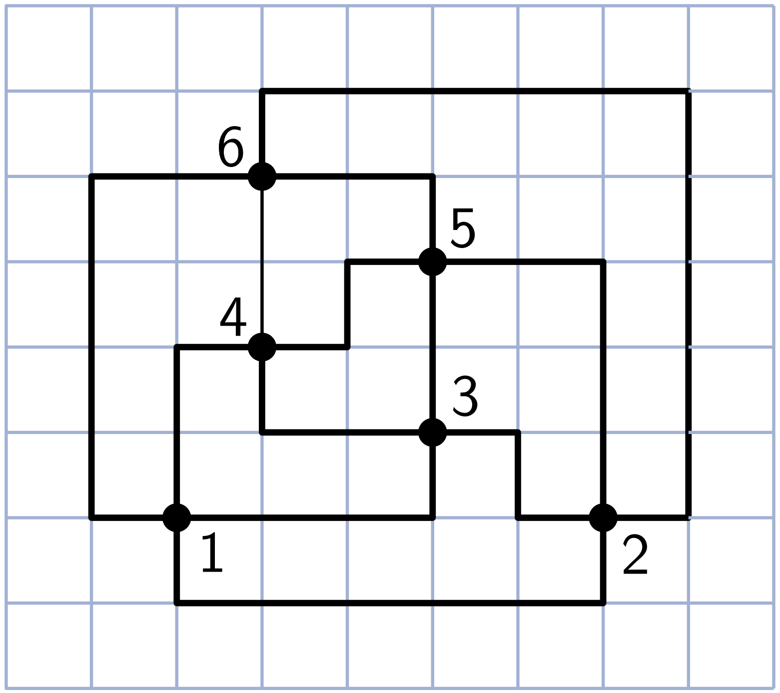


first vertex

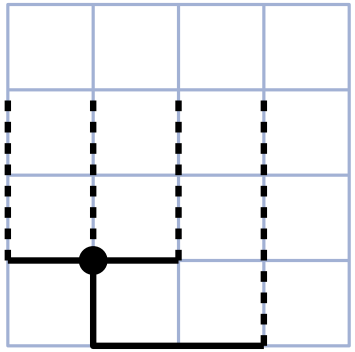


Biedl & Kant Orthogonal Drawing Algorithm

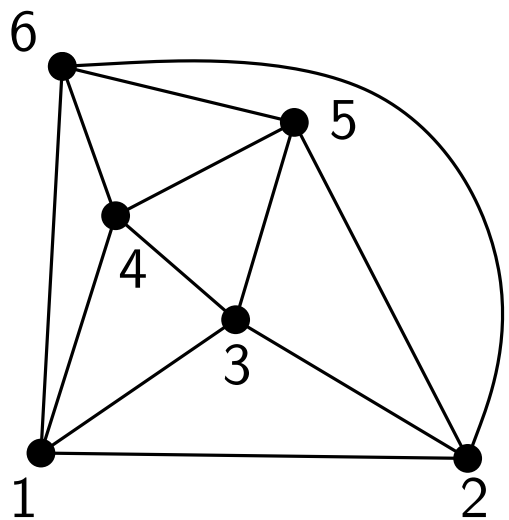


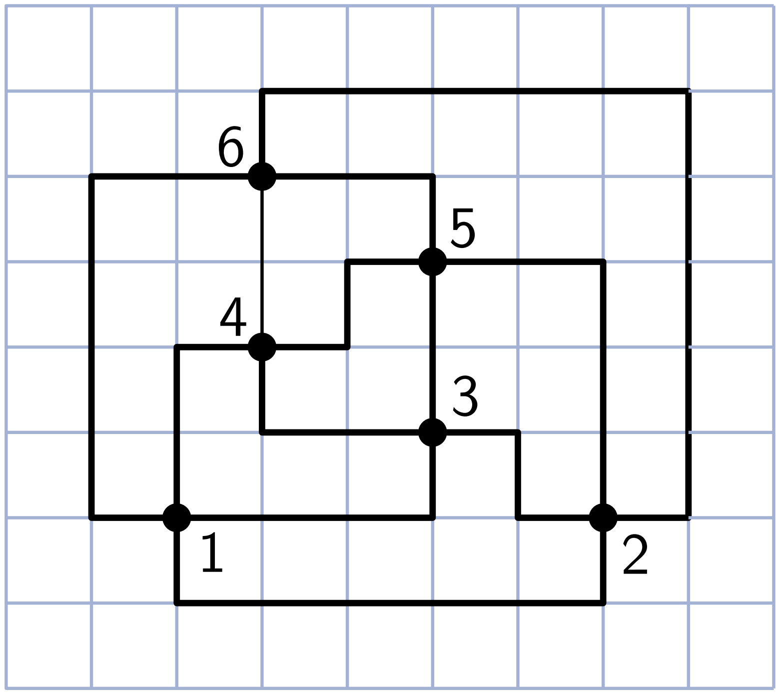


first vertex

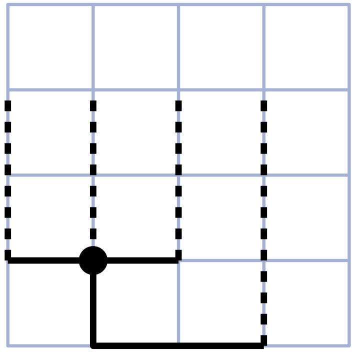


Biedl & Kant Orthogonal Drawing Algorithm

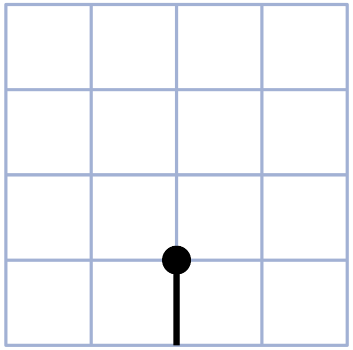




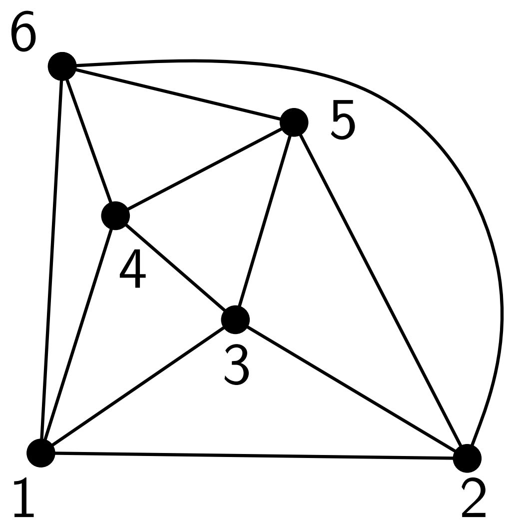
first vertex

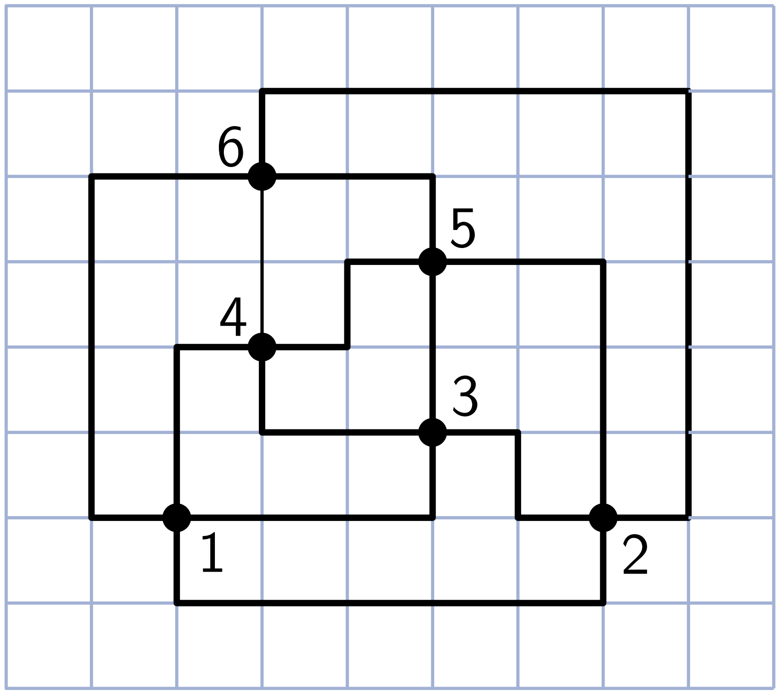


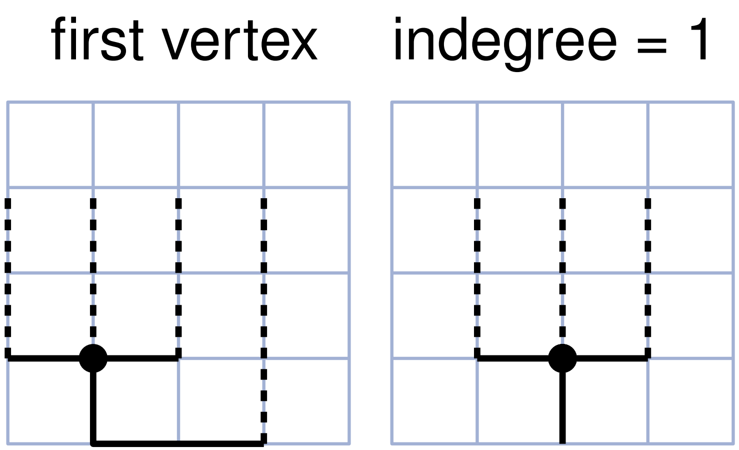
indegree = 1



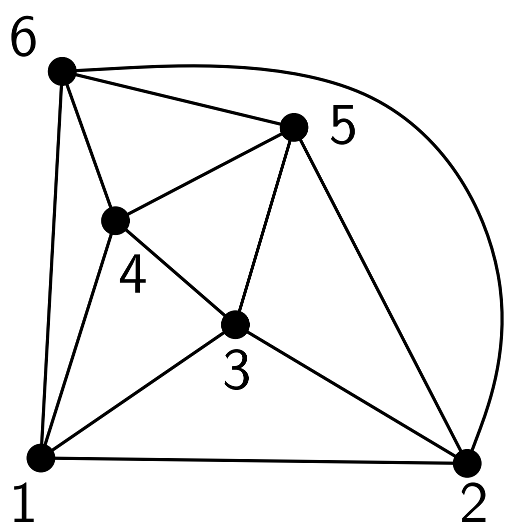
Biedl & Kant Orthogonal Drawing Algorithm

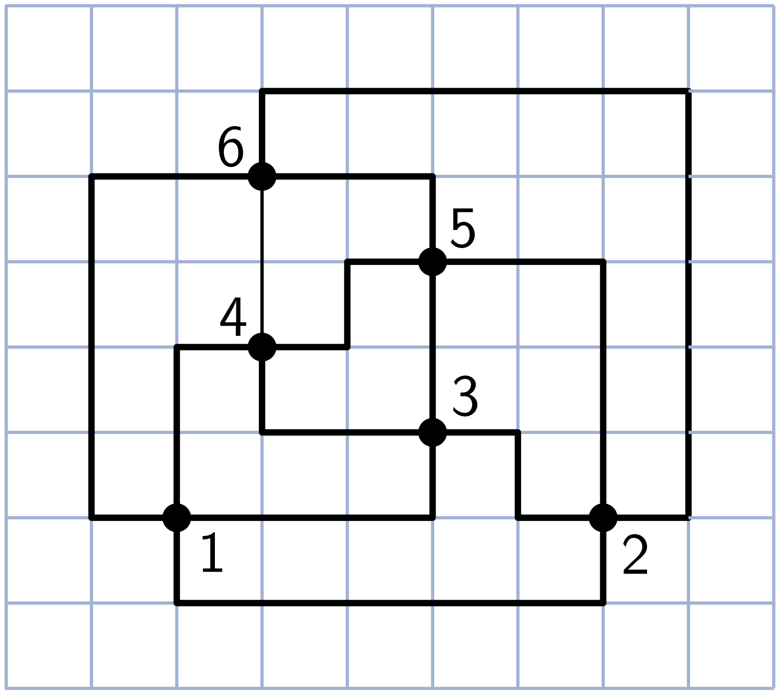




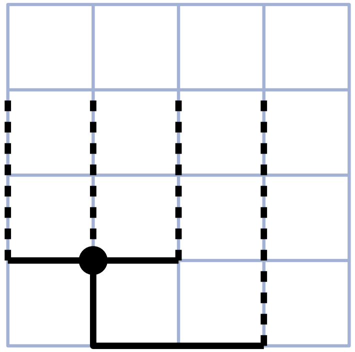


Biedl & Kant Orthogonal Drawing Algorithm

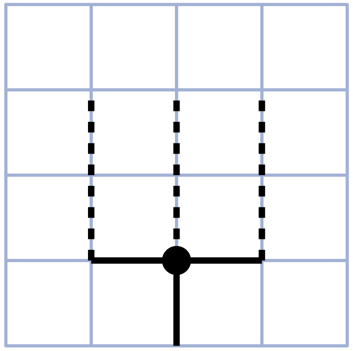




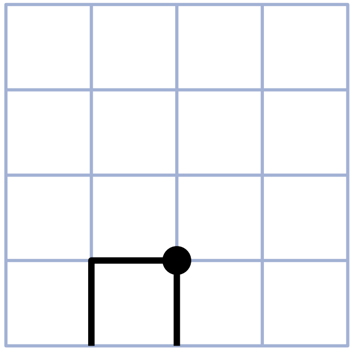
first vertex



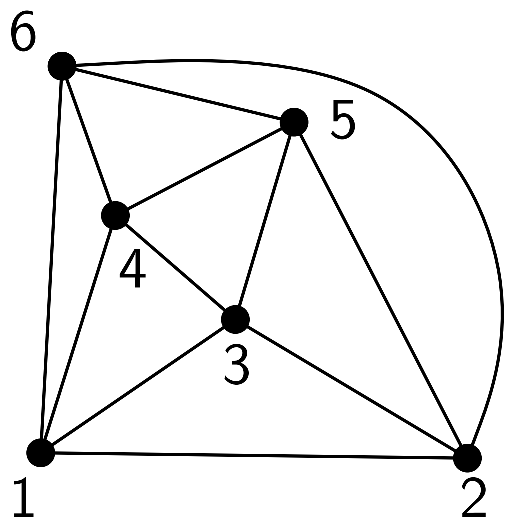
indegree = 1

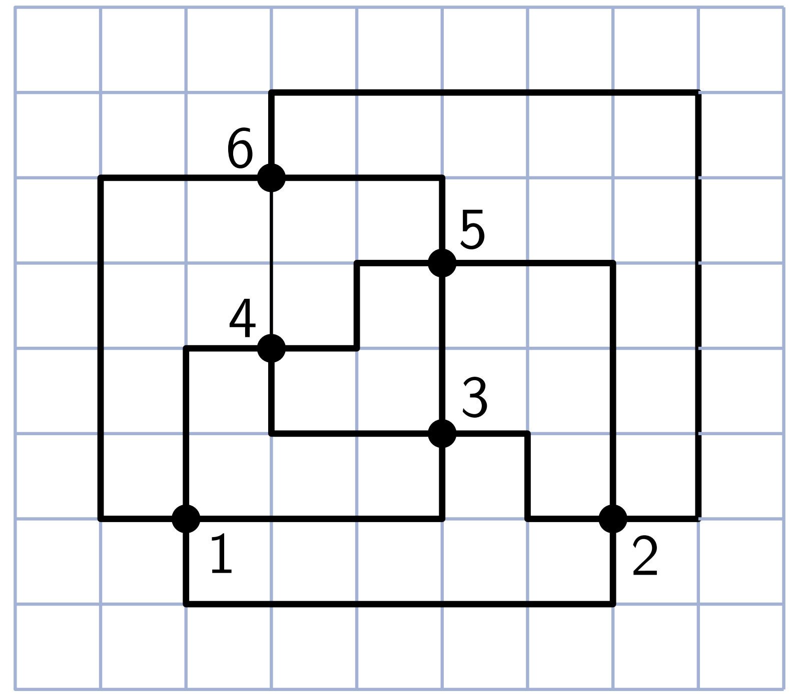


indegree = 2

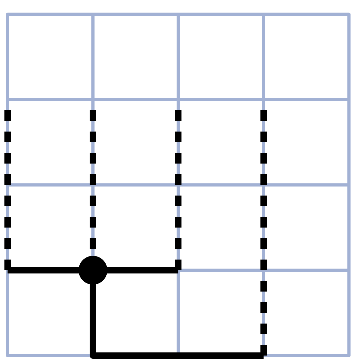


Biedl & Kant Orthogonal Drawing Algorithm

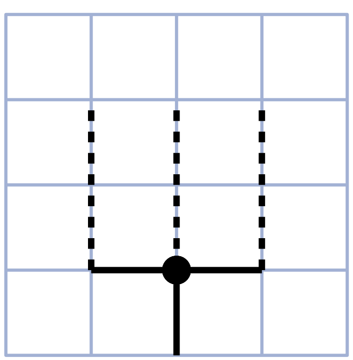




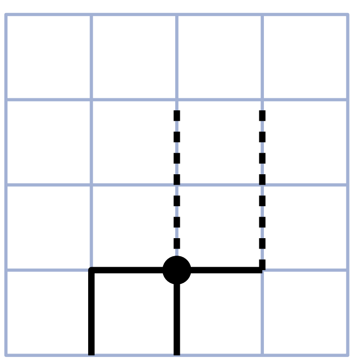
first vertex



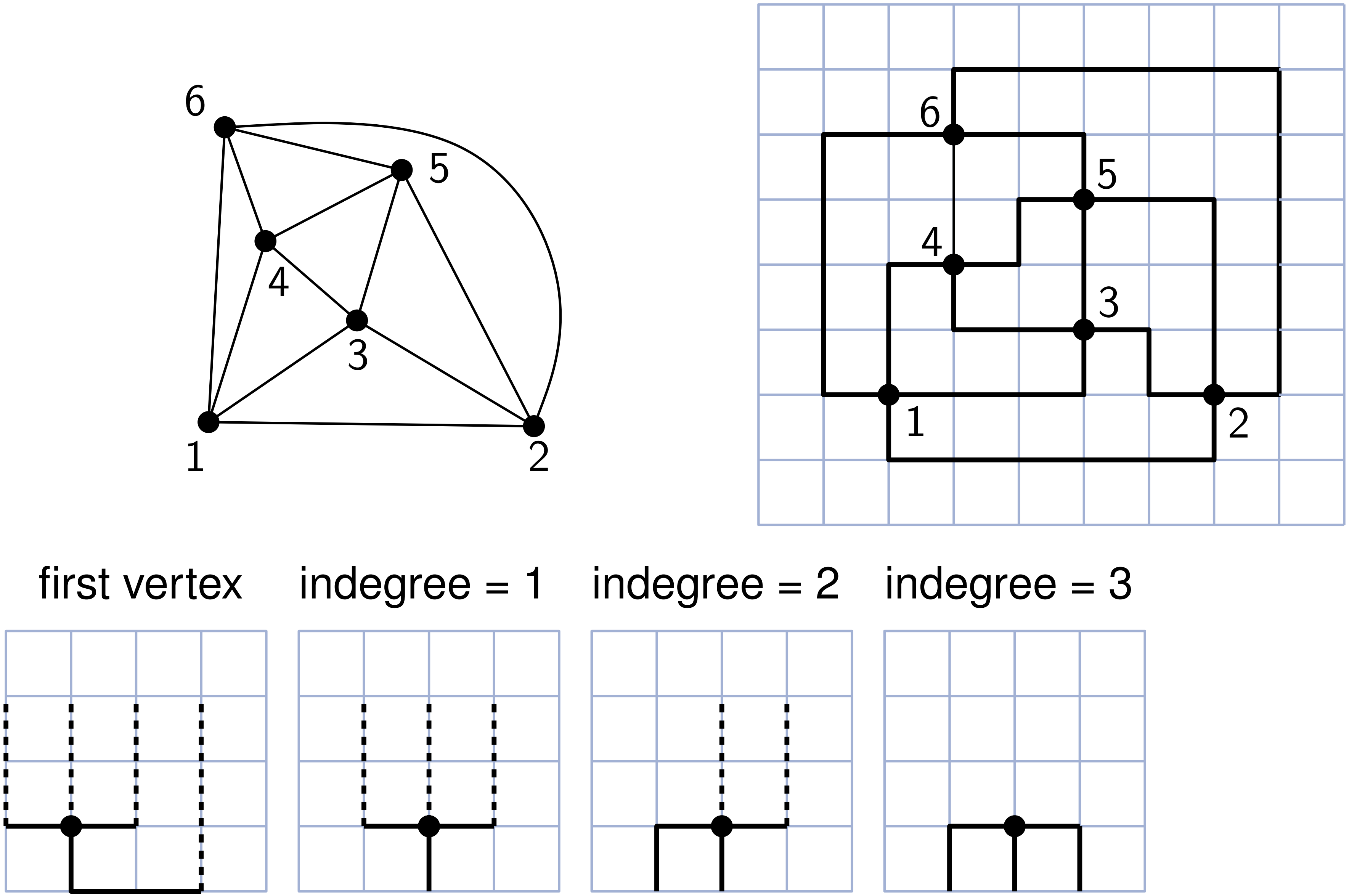
indegree = 1



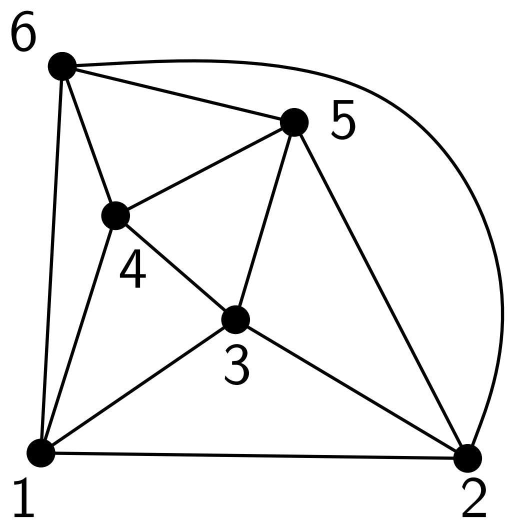
indegree = 2

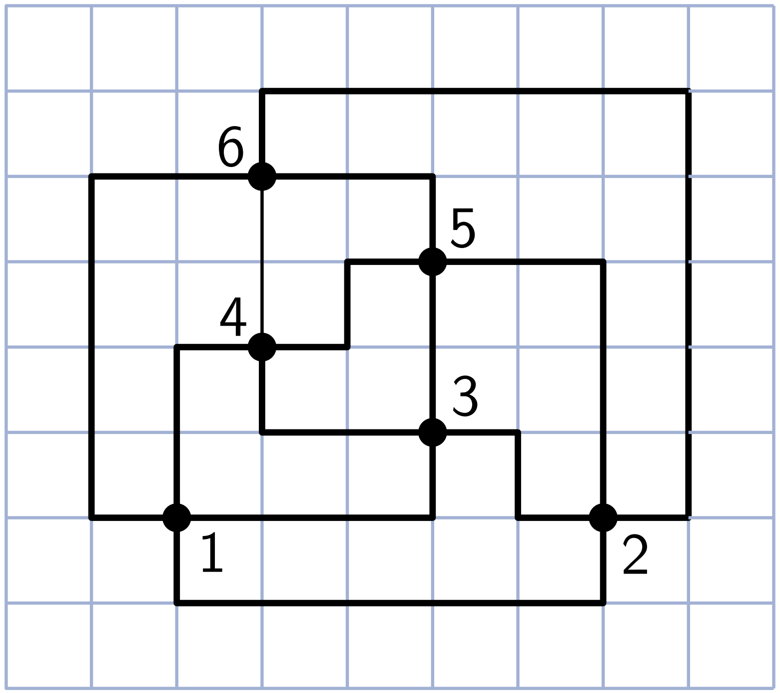


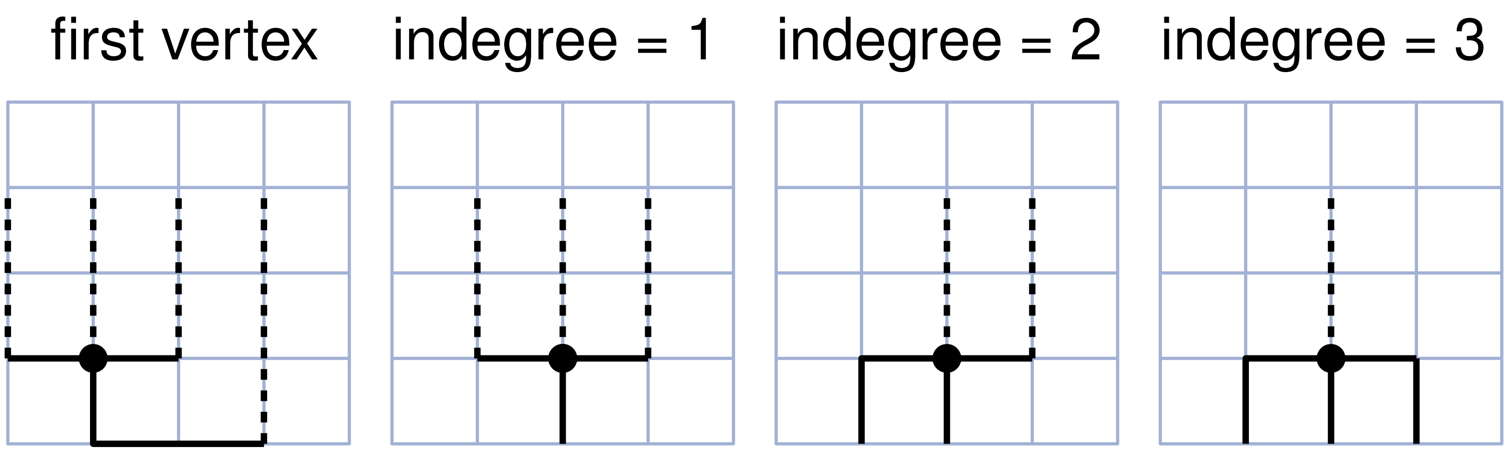
Biedl & Kant Orthogonal Drawing Algorithm



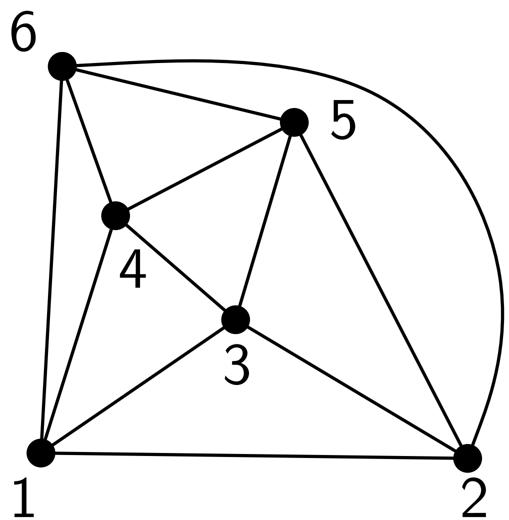
Biedl & Kant Orthogonal Drawing Algorithm

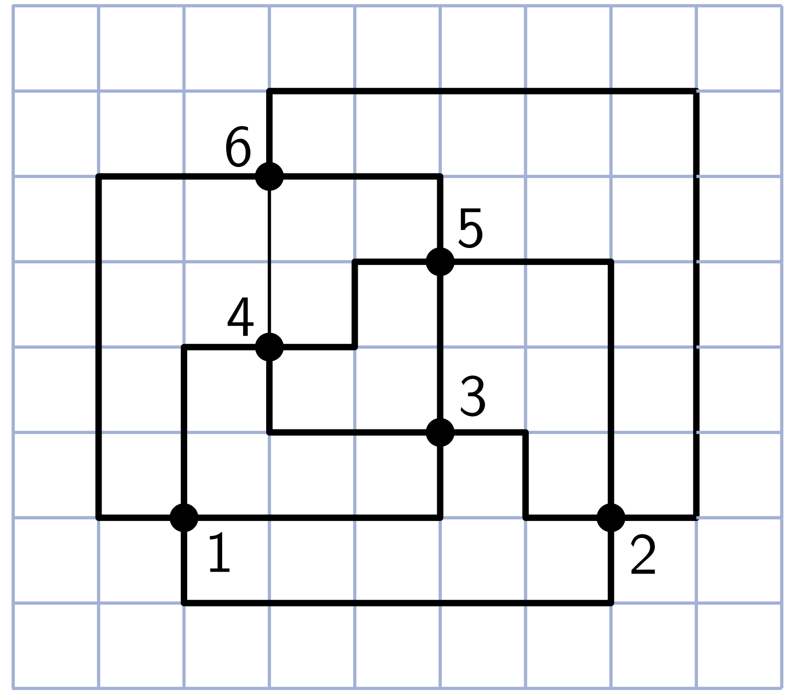




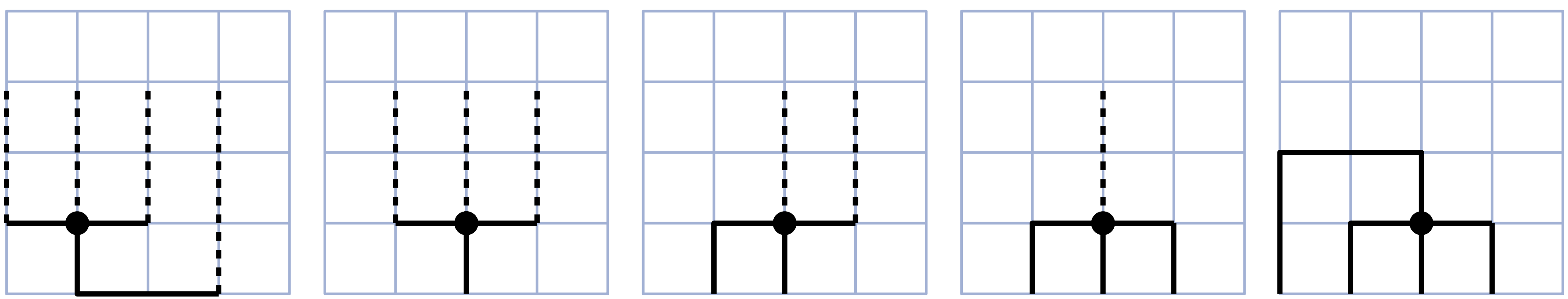


Biedl & Kant Orthogonal Drawing Algorithm





first vertex | indegree = 1 | indegree = 2 | indegree = 3 | indegree = 4



Biedl & Kant Orthogonal Drawing Algorithm

Lemma (Area of Biedl & Kant drawing)

The width is $m - n + 1$ and the height at most $n + 1$.

Biedl & Kant Orthogonal Drawing Algorithm

Lemma (Area of Biedl & Kant drawing)

The width is $m - n + 1$ and the height at most $n + 1$.

Proof

- Width: At each step we increase the number of columns by $outdeg(v_i) - 1$, if $i > 1$ and $outdeg(v_1)$ for v_1.

Karlsruhe Institute of Technology

Lemma (Area of Biedl & Kant drawing)

The width is $m - n + 1$ and the height at most $n + 1$.

Proof

- Width: At each step we increase the number of columns by $outdeg(v_i) - 1$, if $i > 1$ and $outdeg(v_1)$ for v_1.
- Height: Every vertex except for v_2 is placed at a new row. Vertex v_n uses one more row if $indeg(v_n) = 4$.

Biedl & Kant Orthogonal Drawing Algorithm

Lemma (Area of Biedl & Kant drawing)

The width is $m - n + 1$ and the height at most $n + 1$.

Proof

- **Width:** At each step we increase the number of columns by $outdeg(v_i) - 1$, if $i > 1$ and $outdeg(v_1)$ for v_1.
- **Height:** Every vertex except for v_2 is placed at a new row. Vertex v_n uses one more row if $indeg(v_n) = 4$.

Lemma (Number of bends in Biedl & Kant drawing)

There are at most $2m - 2n + 4$ bends.

Biedl & Kant Orthogonal Drawing Algorithm

Karlsruhe Institute of Technology

Lemma (Area of Biedl & Kant drawing)

The width is $m - n + 1$ and the height at most $n + 1$.

Proof

- Width: At each step we increase the number of columns by $outdeg(v_i) - 1$, if $i > 1$ and $outdeg(v_1)$ for v_1.
- Height: Every vertex except for v_2 is placed at a new row. Vertex v_n uses one more row if $indeg(v_n) = 4$.

Lemma (Number of bends in Biedl & Kant drawing)

There are at most $2m - 2n + 4$ bends.

Proof

- Each vertex v_i, $i \neq 1, n$, introduces $indeg(v_i) - 1$ and $outdeg(v_i) - 1$ new bends.

Biedl & Kant Orthogonal Drawing Algorithm

Lemma (Number of bends per edge in Biedl & Kant drawing)

All edges but one bent at most twice. The exceptional edge bent at most three times.

Biedl & Kant Orthogonal Drawing Algorithm

Karlsruhe Institute of Technology

Lemma (Number of bends per edge in Biedl & Kant drawing)

All edges but one bent at most twice. The exceptional edge bent at most three times.

Proof

- Let (v_i, v_j), $i < j$, $i, j \neq 1, n$. Then $outdeg(v_i)$, $indeg(v_j) \leq 3$. I.e (v_i, v_j) gets at most one bend after placement of v_i and at most one before placement of v_j. Edges outgoing from v_1 can me made 2-bend by using the column below v_1 for the edge (v_1, v_2).

Biedl & Kant Orthogonal Drawing Algorithm

Lemma (Number of bends per edge in Biedl & Kant drawing)

All edges but one bent at most twice. The exceptional edge bent at most three times.

Proof

- Let (v_i, v_j), $i < j$, $i, j \neq 1, n$. Then $outdeg(v_i)$, $indeg(v_j) \leq 3$. I.e (v_i, v_j) gets at most one bend after placement of v_i and at most one before placement of v_j. Edges outgoing from v_1 can me made 2-bend by using the column below v_1 for the edge (v_1, v_2).

Lemma (planarity)

For planar graphs the algorithm produces a planar drawing.

Biedl & Kant Orthogonal Drawing Algorithm

Karlsruhe Institute of Technology

Lemma (Number of bends per edge in Biedl & Kant drawing)

All edges but one bent at most twice. The exceptional edge bent at most three times.

Proof

- Let (v_i, v_j), $i < j$, $i, j \neq 1, n$. Then $outdeg(v_i)$, $indeg(v_j) \leq 3$. I.e (v_i, v_j) gets at most one bend after placement of v_i and at most one before placement of v_j. Edges outgoing from v_1 can me made 2-bend by using the column below v_1 for the edge (v_1, v_2).

Lemma (planarity)

For planar graphs the algorithm produces a planar drawing.

Proof

- Consider a planar embedding of G. Let $v_1, \ldots, v_n$ be an st-ordering of G. Let G_i be the graph induced by $v_1, \ldots, v_i$. We will prove later that if G is planar, vertex v_{i+1} lies on the outer face of G_i.

Biedl & Kant Orthogonal Drawing Algorithm

Lemma (planarity)

For planar graphs the algorithm produces a planar drawing.

Proof (Continuation)

- Let E_i be the edges outgoing from the vertices of G_i in the order they appear in the embedded G.

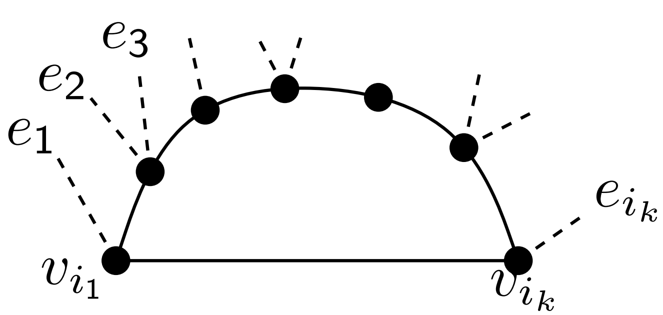
Biedl & Kant Orthogonal Drawing Algorithm

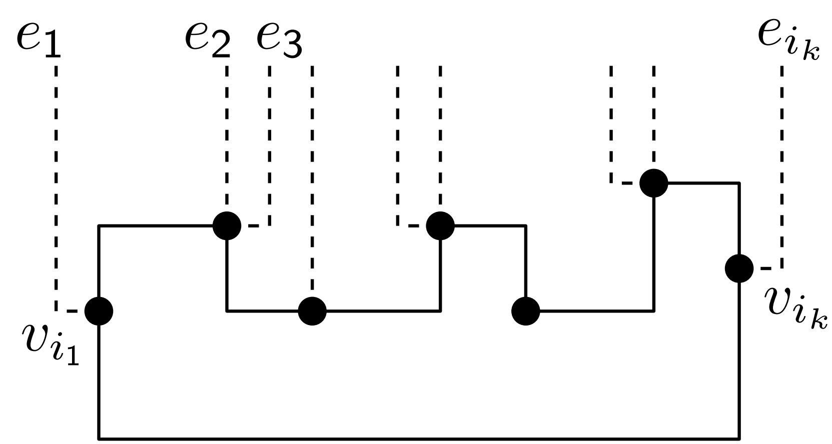
Lemma (planarity)

For planar graphs the algorithm produces a planar drawing.

Proof (Continuation)

- Let E_i be the edges outgoing from the vertices of G_i in the order they appear in the embedded G.
- By induction we can show that E_i appear in the same order in the orthogonal drawing of G_i.





Biedl & Kant Orthogonal Drawing Algorithm

Theorem (Biedl & Kant 98)

A biconnected graph G with vertex-degree at most 4 admits an orthogonal drawing on a $(m - n + 1) \times n + 1$ grid, such that each edge, except maybe for one, have at most 2 bends per edge, while the exceptional edge has at most 3 bends. The total number if bends is $2m - 2n + 4$. If G is planar, the the orthogonal drawing is also planar.

Biedl & Kant Orthogonal Drawing Algorithm

Karlsruhe Institute of Technology

Theorem (Biedl & Kant 98)

A biconnected graph G with vertex-degree at most 4 admits an orthogonal drawing on a $(m-n+1) \times n+1$ grid, such that each edge, except maybe for one, have at most 2 bends per edge, while the exceptional edge has at most 3 bends. The total number if bends is $2m-2n+4$. If G is planar, the the orthogonal drawing is also planar.

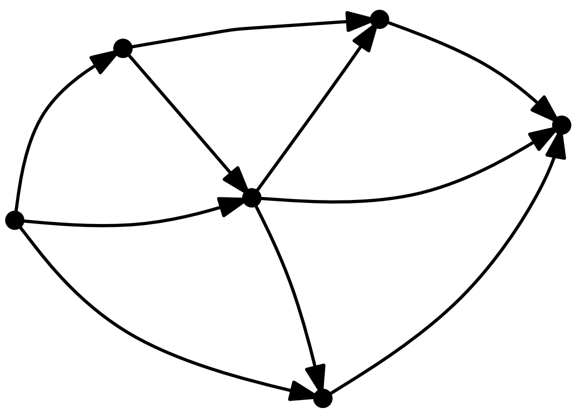
What have we used for the consruction?

- st-ordering $v_1, \ldots, v_n$ of G.
- The following fact: if G is planar, vertex v_{i+1} belongs to the outer face of G_i, where G_i is graph induced by $v_1, \ldots, v_i$.

st-graph, topological ordering

Definition: st-graph

Let G be a directed graph. A vertex s (resp. t) is called **source** (resp. **sink**) of G if it cas only outgoing (resp. incomming edges). A directed acyclic graph with one source and one sink is called st**-graph**.



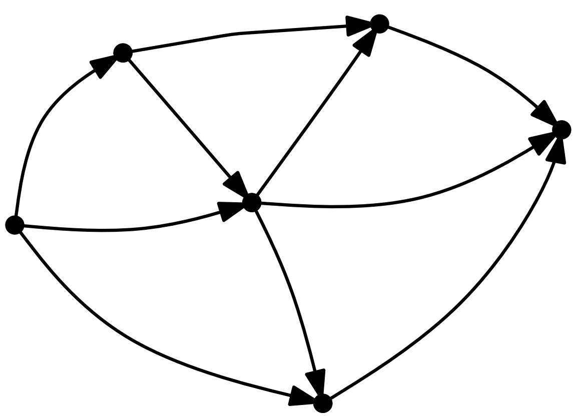
st-graph, topological ordering

Definition: st-graph

Let G be a directed graph. A vertex s (resp. t) is called **source** (resp. **sink**) of G if it cas only outgoing (resp. incomming edges). A directed acyclic graph with one source and one sink is called **st-graph**.

Definition: topological ordering

A **topological numbering** of a directed graph G is an assignment of numbers to the vertices of G, such that for every edge (u, v), number$(v) >$ number(u). **Topological ordering** is a topological numbering where each vertex has a distinct number between 1 and n.



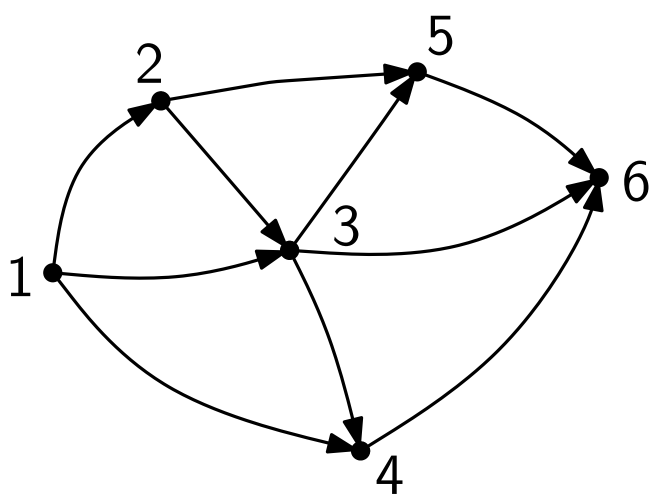
st-graph, topological ordering

Definition: st-graph

Let G be a directed graph. A vertex s (resp. t) is called **source** (resp. **sink**) of G if it cas only outgoing (resp. incomming edges). A directed acyclic graph with one source and one sink is called ***st*-graph**.

Definition: topological ordering

A **topological numbering** of a directed graph G is an assignment of numbers to the vertices of G, such that for every edge (u, v), number$(v) >$ number(u). **Topological ordering** is a topological numbering where each vertex has a distinct number between 1 and n.



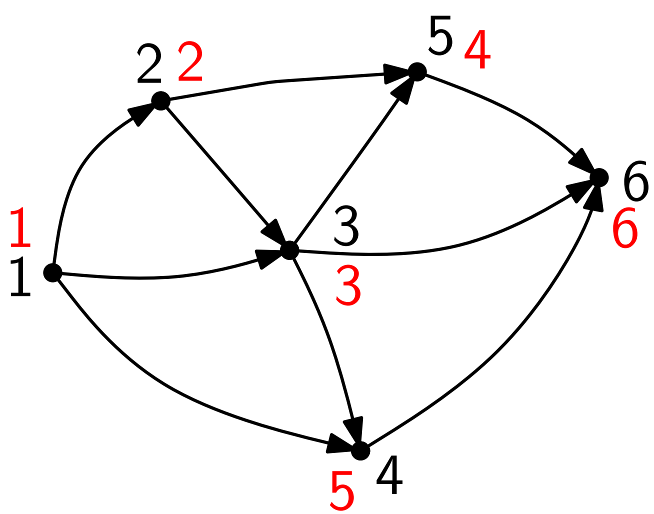
st-graph, topological ordering

Definition: st-graph

Let G be a directed graph. A vertex s (resp. t) is called **source** (resp. **sink**) of G if it cas only outgoing (resp. incomming edges). A directed acyclic graph with one source and one sink is called ***st*-graph**.

Definition: topological ordering

A **topological numbering** of a directed graph G is an assignment of numbers to the vertices of G, such that for every edge (u, v), number(v) $>$ number(u). **Topological ordering** is a topological numbering where each vertex has a distinct number between 1 and n.



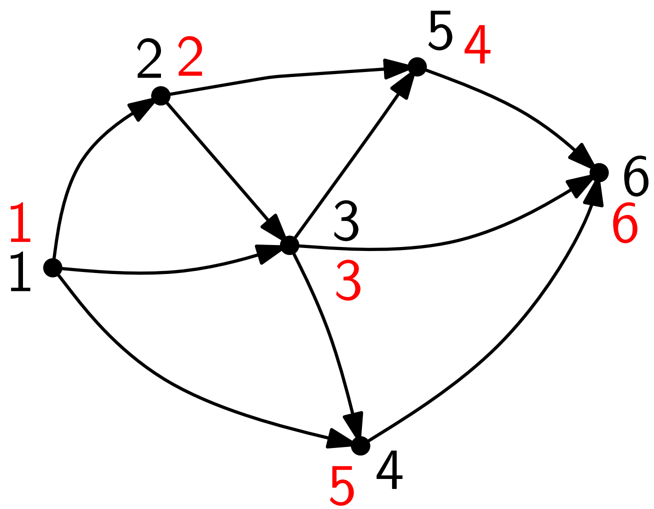
st-graph, topological ordering

Definition: st-graph

Let G be a directed graph. A vertex s (resp. t) is called **source** (resp. **sink**) of G if it cas only outgoing (resp. incomming edges). A directed acyclic graph with one source and one sink is called **st-graph**.

Definition: topological ordering

A **topological numbering** of a directed graph G is an assignment of numbers to the vertices of G, such that for every edge (u, v), number$(v) >$ number(u). **Topological ordering** is a topological numbering where each vertex has a distinct number between 1 and n.



How to construct a topological ordering?

st-ordering

How to construct an st-ordering?

- Recall topological ordering for st-graphs.

st-ordering

How to construct an st-ordering?

- Recall topological ordering for st-graphs.
- Take an undirected G and orient its edges so that you get an st-graph G'.

st-ordering

How to construct an st-ordering?

- Recall topological ordering for st-graphs.
- Take an undirected G and orient its edges so that you get an st-graph G'.
- A topological ordering of G' is an st-ordering of G.

st-ordering

How to construct an st-ordering?

- Recall topological ordering for st-graphs.
- Take an undirected G and orient its edges so that you get an st-graph G'.
- A topological ordering of G' is an st-ordering of G.
- How to orient edges of G to obtain an st-graph G'?

st-ordering

How to construct an st-ordering?

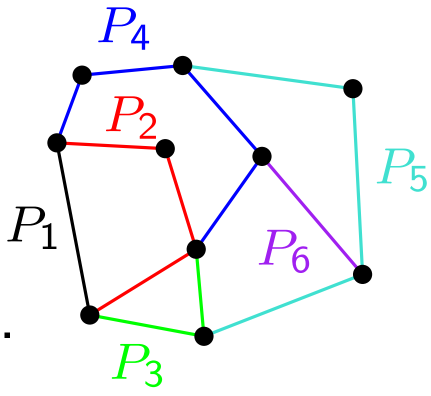
- Recall topological ordering for st-graphs.
- Take an undirected G and orient its edges so that you get an st-graph G'.
- A topological ordering of G' is an st-ordering of G.
- How to orient edges of G to obtain an st-graph G'?

Definition: Ear decomposition

An ear decomposition $D = (P_0, \ldots, P_r)$ of an undirected graph $G = (V, E)$ is a partition of E into an ordered collection of edge disjoint paths $P_0, \ldots, P_r$, such that:

- P_0 is an edge
- $P_0 \cup P_1$ is a simple cycle
- both end-vertices of P_i belong to $P_0 \cup \cdots \cup P_{i-1}$
- no internal vertex of P_i belong to $P_0 \cup \cdots \cup P_{i-1}$

An ear decomposition of open if $P_0, \ldots, P_r$ are simple paths.



st-ordering

Lemma (Ear decomposition)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. G has an open ear decomposition $(P_0, \dots, P_r)$, where $P_0 = (s, t)$.

st-ordering

Lemma (Ear decomposition)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. G has an open ear decomposition $(P_0, \dots, P_r)$, where $P_0 = (s, t)$.

Proof

- Let $P_0 = (s, t)$ and P_1 be path between s and t, it exists since G is biconnected.

st-ordering

Lemma (Ear decomposition)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. G has an open ear decomposition $(P_0, \ldots, P_r)$, where $P_0 = (s, t)$.

Proof

- Let $P_0 = (s, t)$ and P_1 be path between s and t, it exists since G is biconnected.
- Induction hypothesis: $P_0, \ldots, P_i$ are ears.

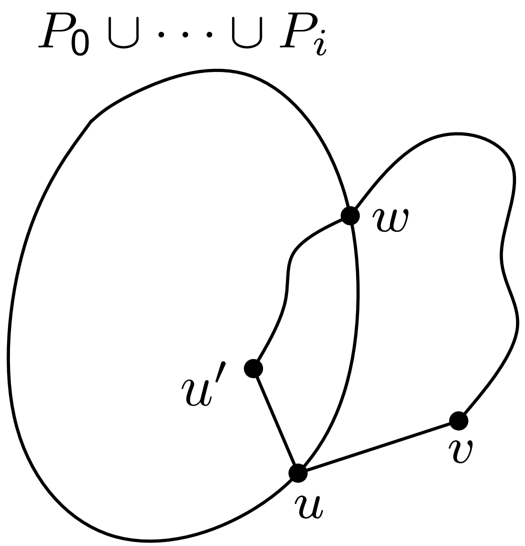
st-ordering

Lemma (Ear decomposition)

Let $G = (V, E)$ be a biconnected graph G and let $(s,t) \in E$. G has an open ear decomposition $(P_0, \dots, P_r)$, where $P_0 = (s,t)$.

Proof

- Let $P_0 = (s,t)$ and P_1 be path between s and t, it exists since G is biconnected.
- Induction hypothesis: $P_0, \dots, P_i$ are ears.
- Let (u,v) be an edge in G such that $u \in P_0 \cup \dots \cup P_i$ and $v \notin P_0 \cup \dots \cup P_i$. Let $(u,u') \in P_0 \cup \dots \cup P_i$. Let P be a path between v and u', disjoint from path $u' - u - v$. P exists since G is biconnected.



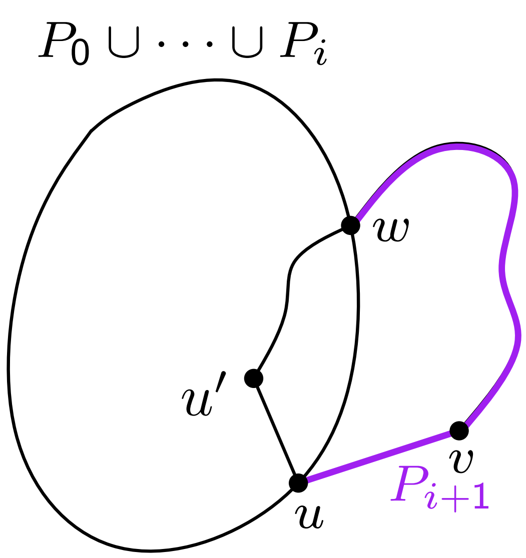
st-ordering

Lemma (Ear decomposition)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. G has an open ear decomposition $(P_0, \dots, P_r)$, where $P_0 = (s, t)$.

Proof

- Let $P_0 = (s, t)$ and P_1 be path between s and t, it exists since G is biconnected.
- Induction hypothesis: $P_0, \dots, P_i$ are ears.
- Let (u, v) be an edge in G such that $u \in P_0 \cup \dots \cup P_i$ and $v \notin P_0 \cup \dots \cup P_i$. Let $(u, u') \in P_0 \cup \dots \cup P_i$. Let P be a path between v and u', disjoint from path $u' - u - v$. P exists since G is biconnected.
- Let w be the first vertex of P that is contained in $P_0 \cup \dots \cup P_i$. Set $P_{i+1} = (u, v) \cup P(v - \dots - w)$.



st-ordering

Lemma (st-orientation)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. There is an orientation G' of G which represents an st-graph. G' is called st-orientation of G.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik I

st-ordering

Karlsruhe Institute of Technology

Lemma (st-orientation)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. There is an orientation G' of G which represents an st-graph. G' is called st-orientation of G.

Proof

- Let $D = (P_0, \ldots, P_r)$ be an ear decomposition of $G = (V, E)$. Notice that $G = P_0 \cup \cdots \cup P_r$.

st-ordering

Lemma (st-orientation)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. There is an orientation G' of G which represents an st-graph. G' is called st-orientation of G.

Proof

- Let $D = (P_0, \ldots, P_r)$ be an ear decomposition of $G = (V, E)$. Notice that $G = P_0 \cup \cdots \cup P_r$.
- Let $G_i = P_0 \cup \cdots \cup P_i$. We prove that G_i has an st-orientation by induction on i.

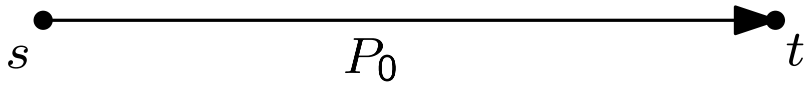
st-ordering

Lemma (st-orientation)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. There is an orientation G' of G which represents an st-graph. G' is called st-orientation of G.

Proof

- Let $D = (P_0, \ldots, P_r)$ be an ear decomposition of $G = (V, E)$. Notice that $G = P_0 \cup \cdots \cup P_r$.
- Let $G_i = P_0 \cup \cdots \cup P_i$. We prove that G_i has an st-orientation by induction on i.



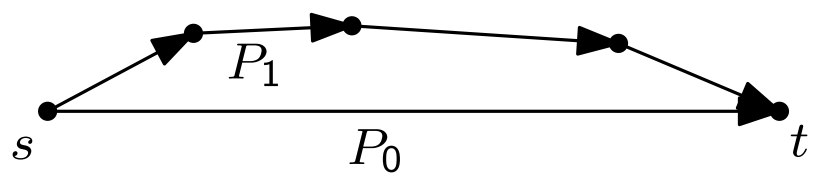
st-ordering

Lemma (st-orientation)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. There is an orientation G' of G which represents an st-graph. G' is called st-orientation of G.

Proof

- Let $D = (P_0, \ldots, P_r)$ be an ear decomposition of $G = (V, E)$. Notice that $G = P_0 \cup \cdots \cup P_r$.
- Let $G_i = P_0 \cup \cdots \cup P_i$. We prove that G_i has an st-orientation by induction on i.



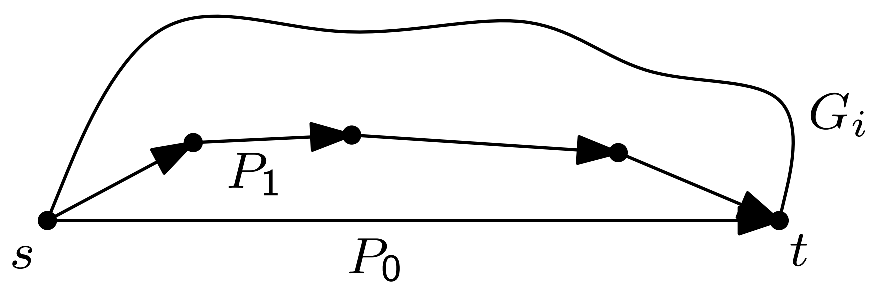
st-ordering

Lemma (st-orientation)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. There is an orientation G' of G which represents an st-graph. G' is called st-orientation of G.

Proof

- Let $D = (P_0, \ldots, P_r)$ be an ear decomposition of $G = (V, E)$. Notice that $G = P_0 \cup \cdots \cup P_r$.
- Let $G_i = P_0 \cup \cdots \cup P_i$. We prove that G_i has an st-orientation by induction on i.



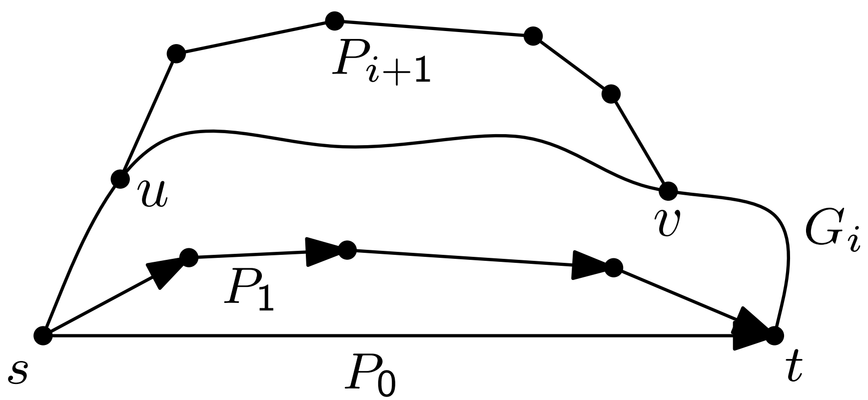
st-ordering

Lemma (st-orientation)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. There is an orientation G' of G which represents an st-graph. G' is called st-orientation of G.

Proof

- Let $D = (P_0, \ldots, P_r)$ be an ear decomposition of $G = (V, E)$. Notice that $G = P_0 \cup \cdots \cup P_r$.
- Let $G_i = P_0 \cup \cdots \cup P_i$. We prove that G_i has an st-orientation by induction on i.



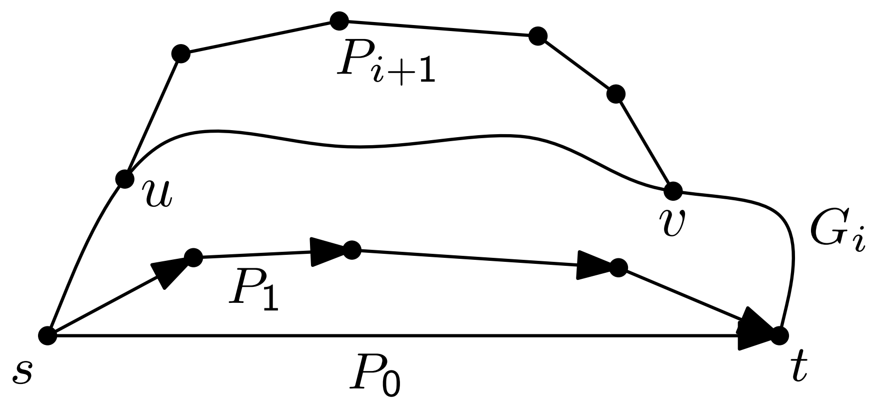
st-ordering

Lemma (st-orientation)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. There is an orientation G' of G which represents an st-graph. G' is called st-orientation of G.

Proof

- Let $D = (P_0, \ldots, P_r)$ be an ear decomposition of $G = (V, E)$. Notice that $G = P_0 \cup \cdots \cup P_r$.
- Let $G_i = P_0 \cup \cdots \cup P_i$. We prove that G_i has an st-orientation by induction on i.



- Distinguish two cases based on whether u and v are connected by a directed path or not.

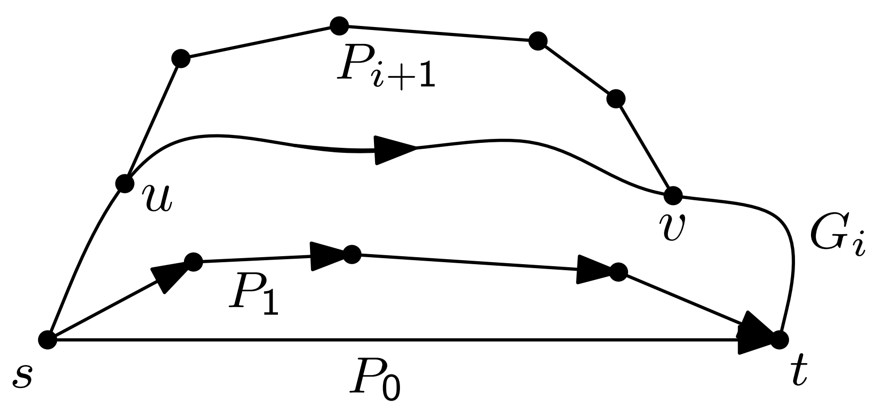
st-ordering

Lemma (st-orientation)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. There is an orientation G' of G which represents an st-graph. G' is called st-orientation of G.

Proof

- Let $D = (P_0, \ldots, P_r)$ be an ear decomposition of $G = (V, E)$. Notice that $G = P_0 \cup \cdots \cup P_r$.
- Let $G_i = P_0 \cup \cdots \cup P_i$. We prove that G_i has an st-orientation by induction on i.



- Distinguish two cases based on whether u and v are connected by a directed path or not.

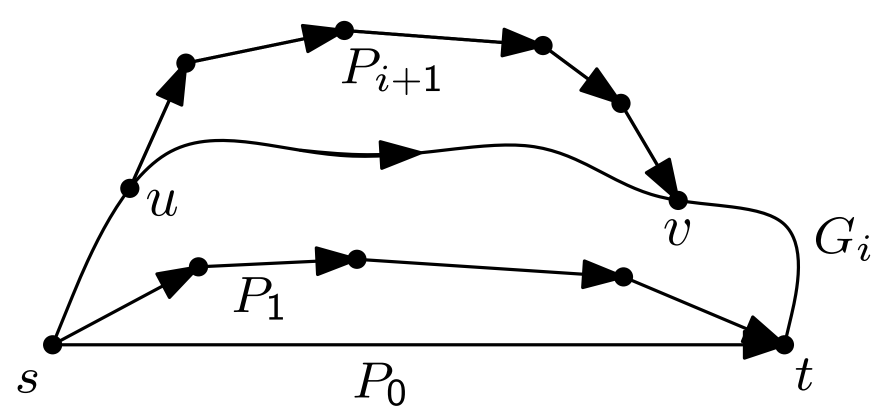
st-ordering

Lemma (st-orientation)

Let $G = (V, E)$ be a biconnected graph G and let $(s, t) \in E$. There is an orientation G' of G which represents an st-graph. G' is called st-orientation of G.

Proof

- Let $D = (P_0, \ldots, P_r)$ be an ear decomposition of $G = (V, E)$. Notice that $G = P_0 \cup \cdots \cup P_r$.
- Let $G_i = P_0 \cup \cdots \cup P_i$. We prove that G_i has an st-orientation by induction on i.



- Distinguish two cases based on whether u and v are connected by a directed path or not.

st-ordering

- Recall that if G is biconnected graph and G' is an st-orientation of G, then a topological ordering of G' is an st-ordering of G.

st-ordering

- Recall that if G is biconnected graph and G' is an st-orientation of G, then a topological ordering of G' is an st-ordering of G.

But how to obtain an st-ordering directly from ear decomposition?

st-ordering

- Recall that if G is biconnected graph and G' is an st-orientation of G, then a topological ordering of G' is an st-ordering of G.

But how to obtain an st-ordering directly from ear decomposition?

- We construct it incrementally, considering $G_i = P_0 \cup \cdots \cup P_i$, $i = 0, \ldots, r$.

st-ordering

- Recall that if G is biconnected graph and G' is an st-orientation of G, then a topological ordering of G' is an st-ordering of G.

But how to obtain an st-ordering directly from ear decomposition?

- We construct it incrementally, considering $G_i = P_0 \cup \cdots \cup P_i$, $i = 0, \ldots, r$.
- For G_1, let $P_1 = \{u_1^1, \ldots, u_{r_1}^1\}$, here $u_1^1 = s$ and $u_{i_1}^1 = t$. The sequence $L = \{u_1^1, \ldots, u_{r_1}^1\}$ is an st-ordering of G_1.

st-ordering

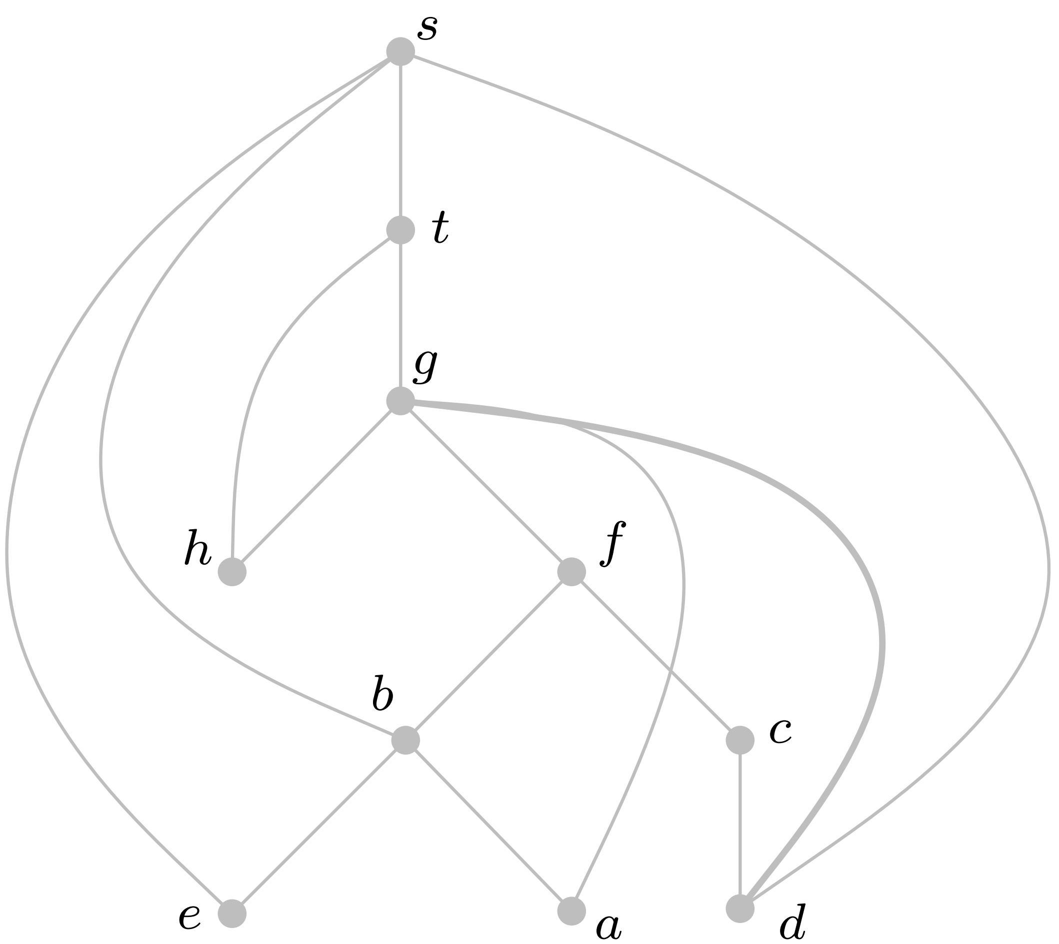
- Recall that if G is biconnected graph and G' is an st-orientation of G, then a topological ordering of G' is an st-ordering of G.

But how to obtain an st-ordering directly from ear decomposition?

- We construct it incrementally, considering $G_i = P_0 \cup \cdots \cup P_i$, $i = 0, \ldots, r$.
- For G_1, let $P_1 = \{u_1^1, \ldots, u_{r_1}^1\}$, here $u_1^1 = s$ and $u_{i_1}^1 = t$. The sequence $L = \{u_1^1, \ldots, u_{r_1}^1\}$ is an st-ordering of G_1.
- Assume that L contains an st-ordering of G_i and let ear $P_{i+1} = \{u_1^{i+1}, \ldots, u_{r_{i+1}}^{i+1}\}$. We insert vertices $u_1^{i+1}, \ldots, u_{r_{i+1}}^{i+1}$ to L after vertex u_1^{i+1}. Let G'_{i+1} be an st-orientation of G_i as constructed in the previous proof. L is a topological ordering of G'_{i+1} and therefore an st-ordering of G_i.

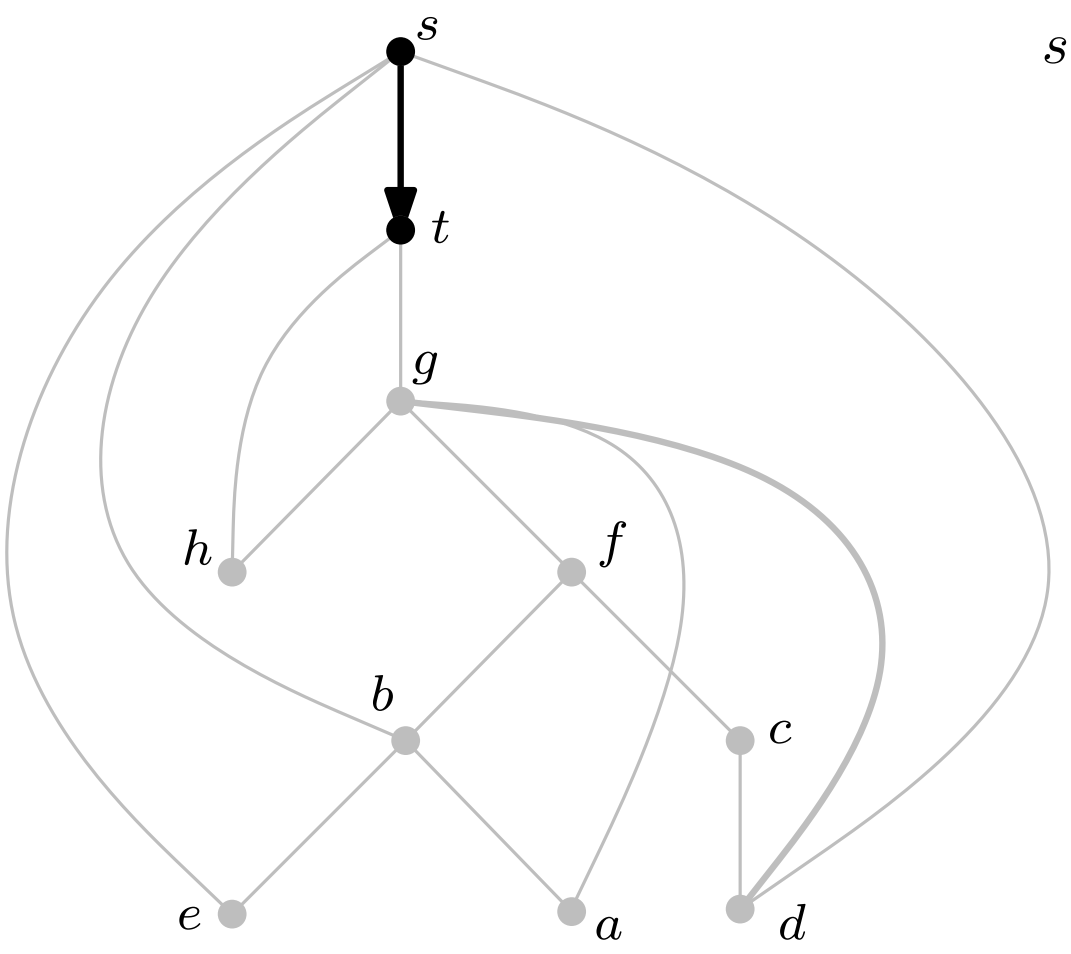
st-ordering

Algorithm: st-ordering (example)



st-ordering

Algorithm: st-ordering (example)

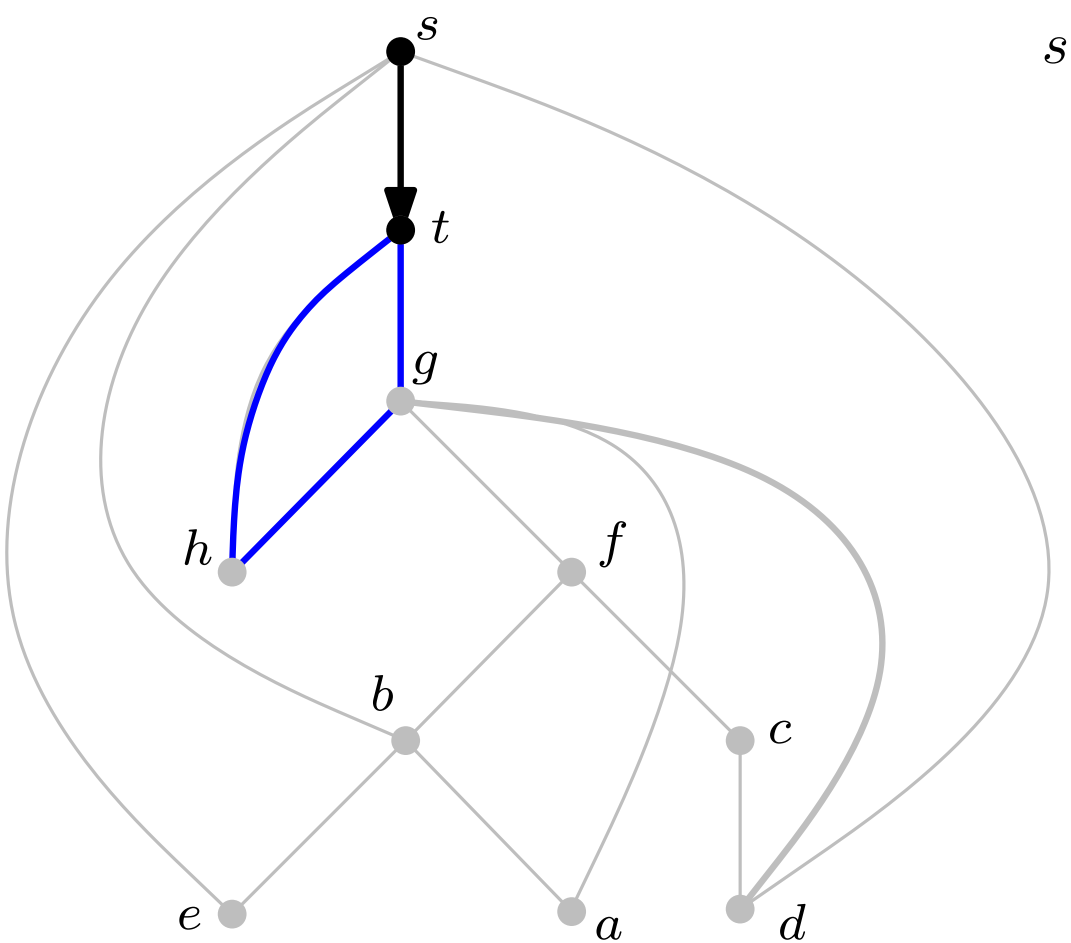


s, t

st-ordering

KIT Karlsruhe Institute of Technology

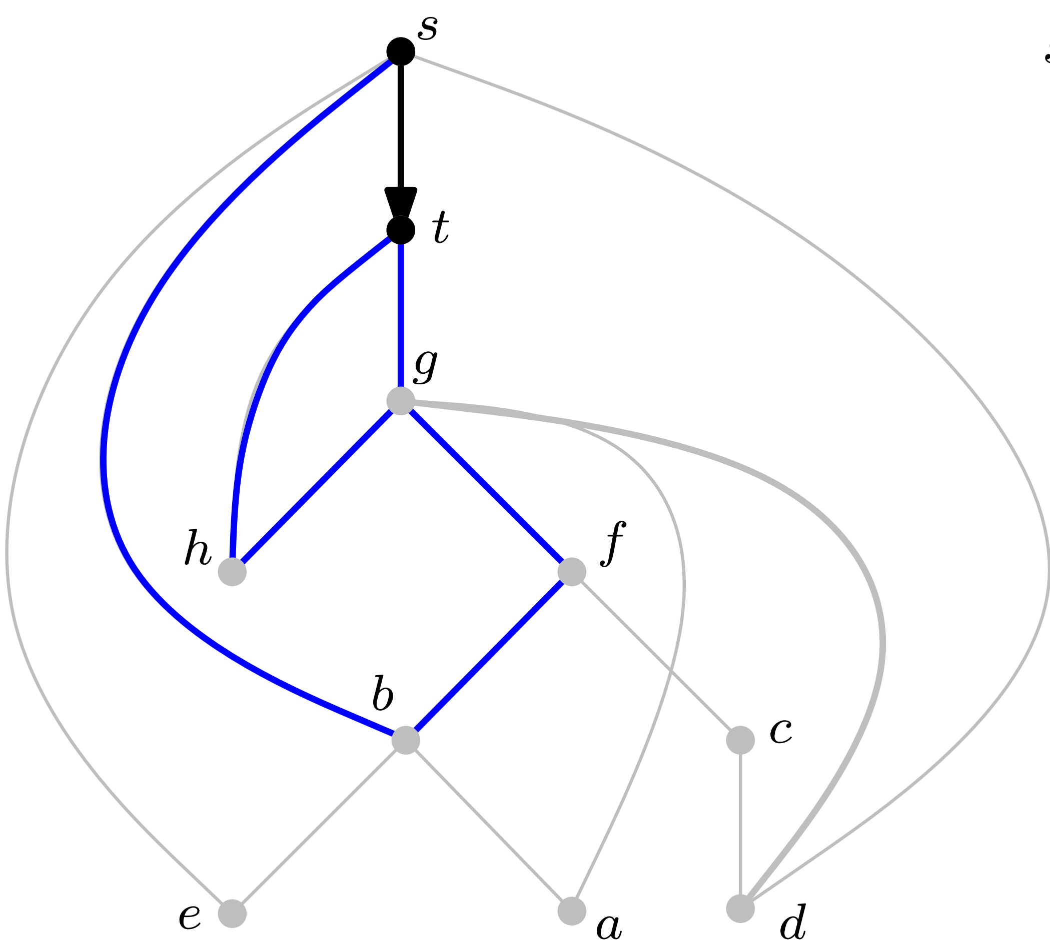
Algorithm: st-ordering (example)



s, t

st-ordering

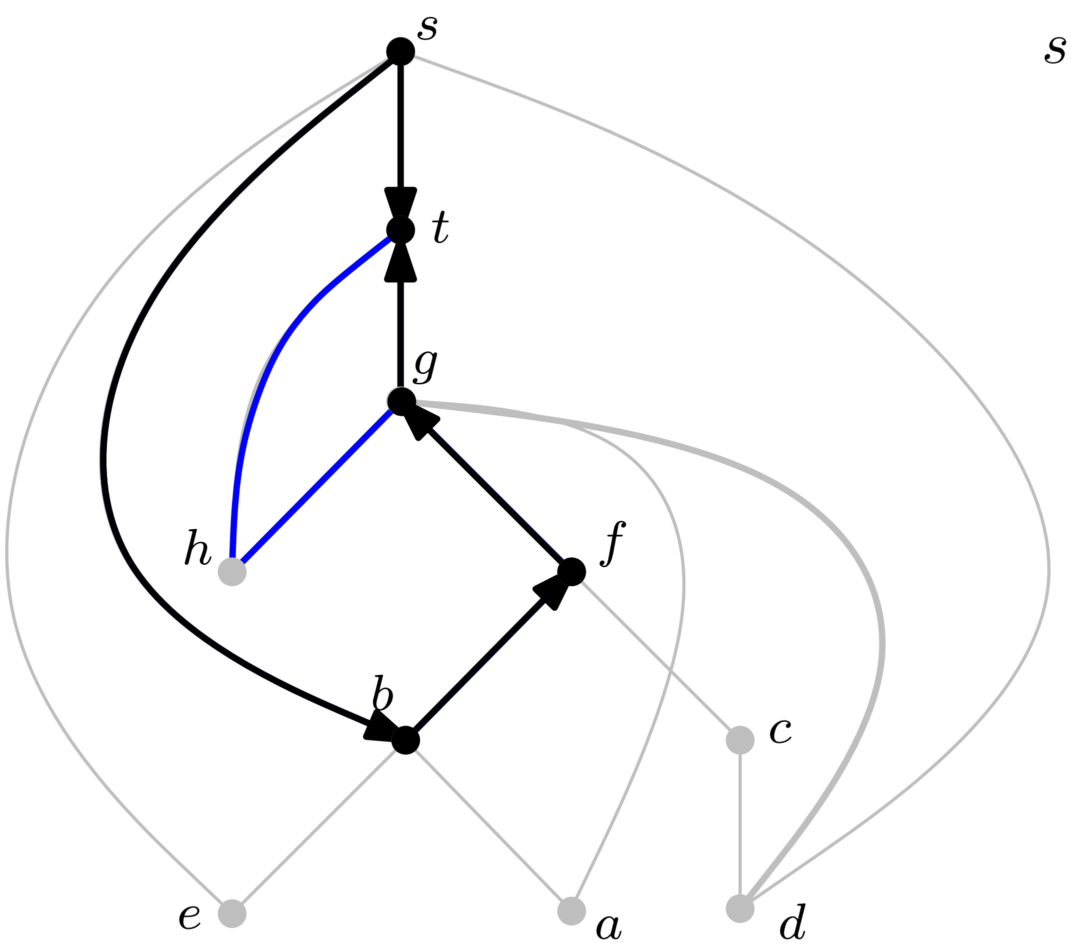
Algorithm: st-ordering (example)



s, t

st-ordering

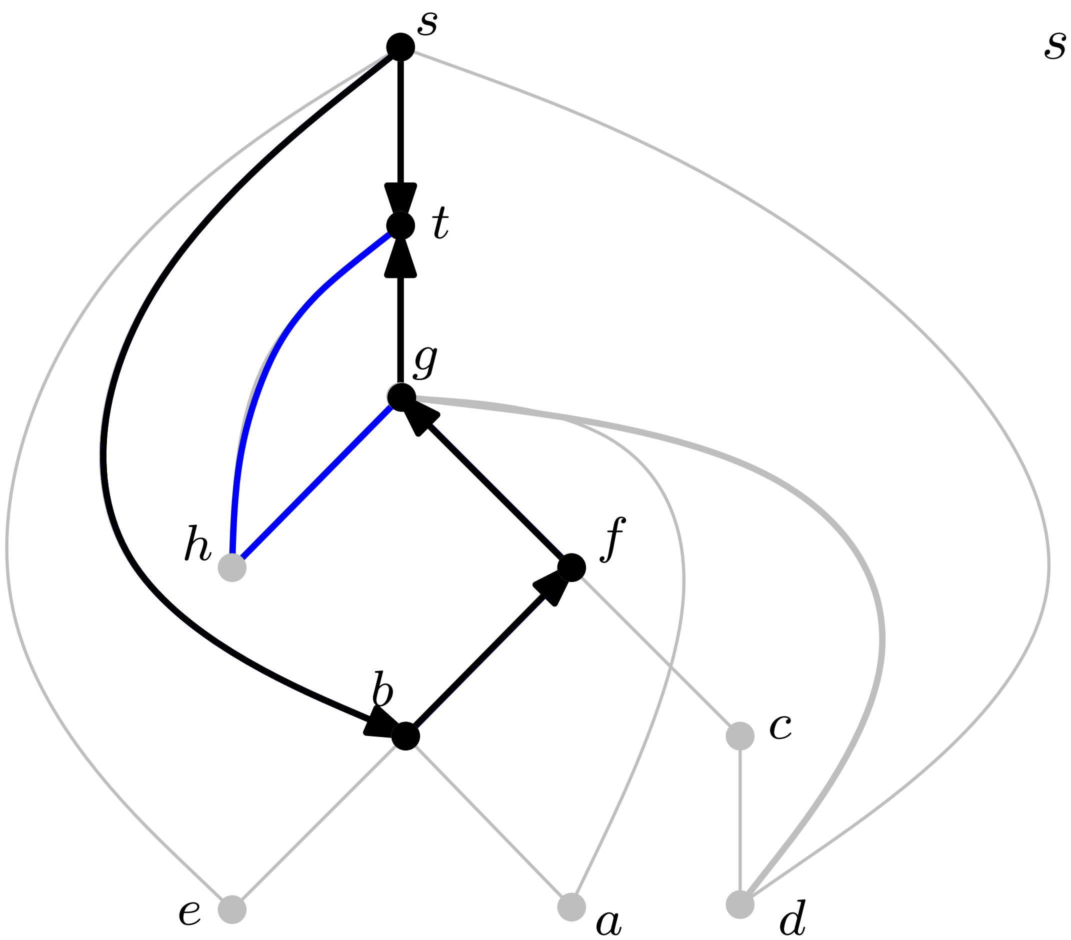
Algorithm: st-ordering (example)



s, t

st-ordering

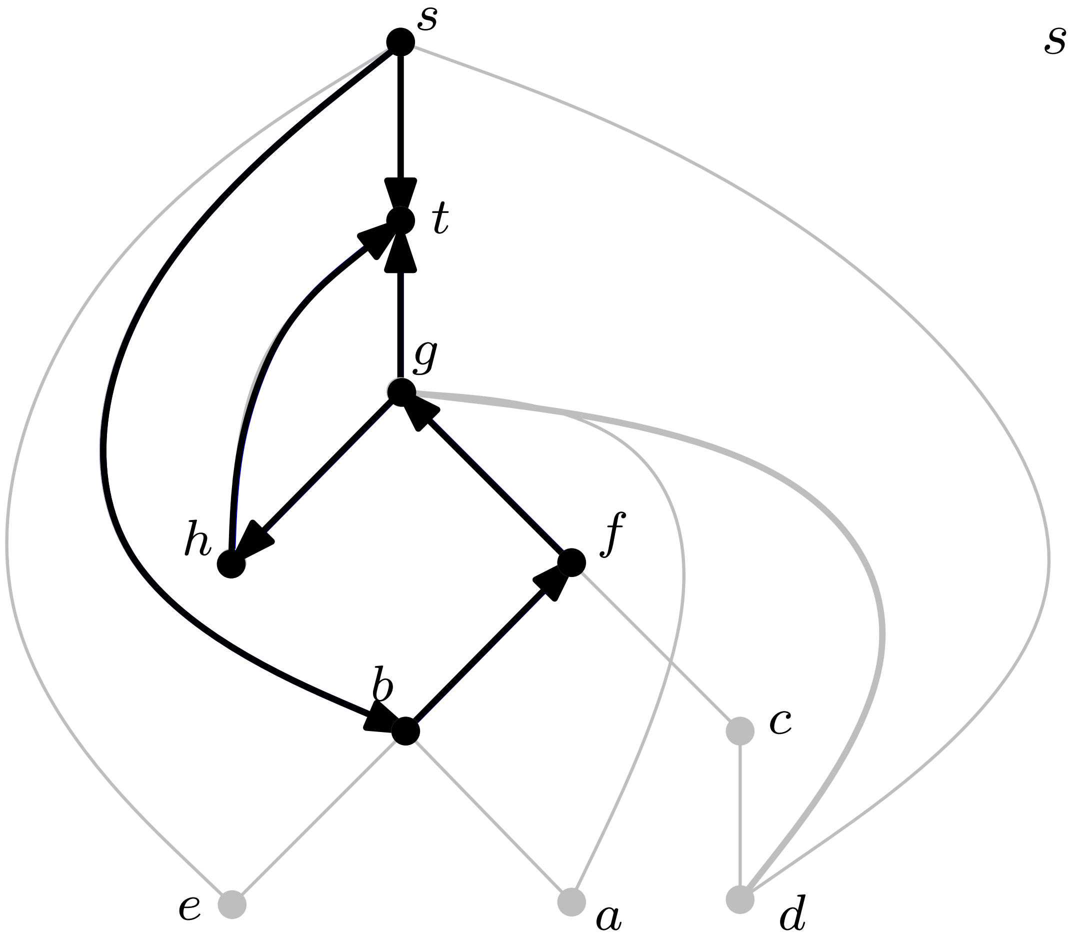
Algorithm: st-ordering (example)



$s, \underline{b}, \underline{f}, \underline{g}, t$

st-ordering

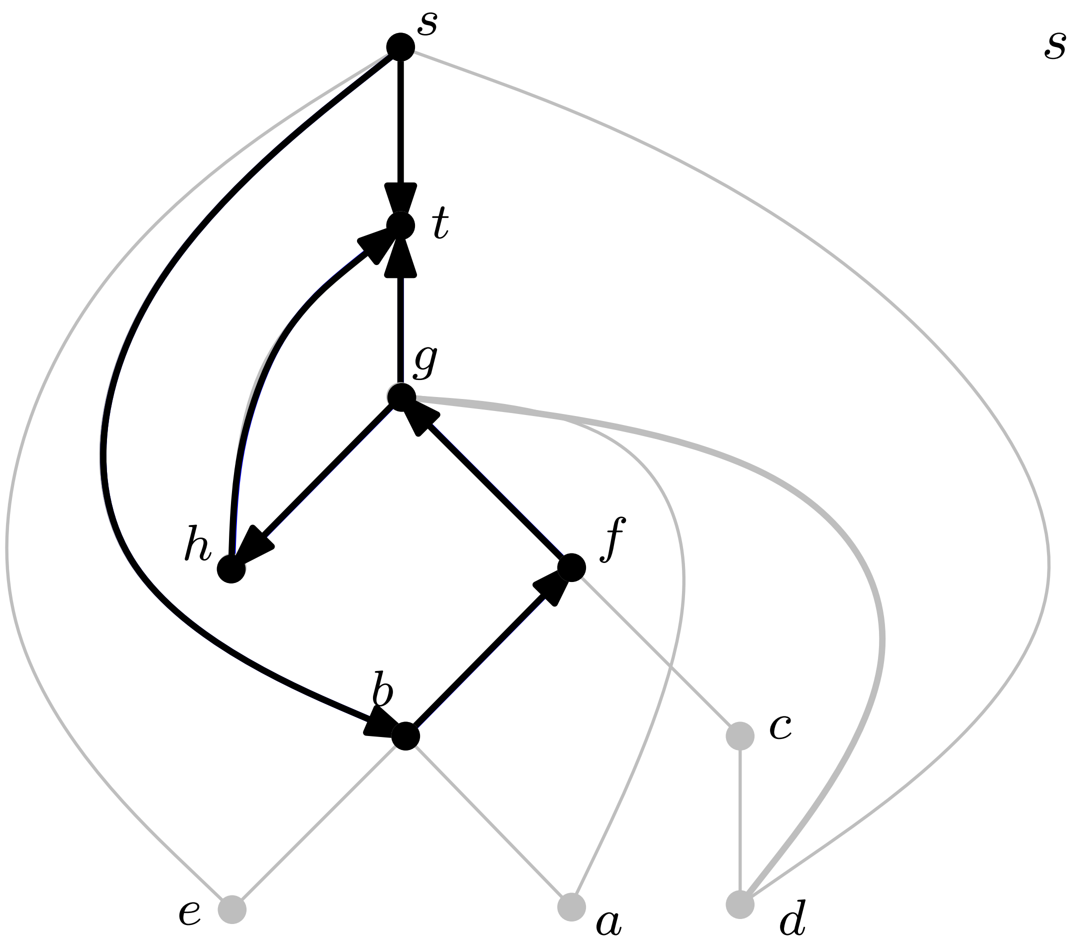
Algorithm: st-ordering (example)



$s, \underline{b}, \underline{f}, \underline{g}, t$

st-ordering

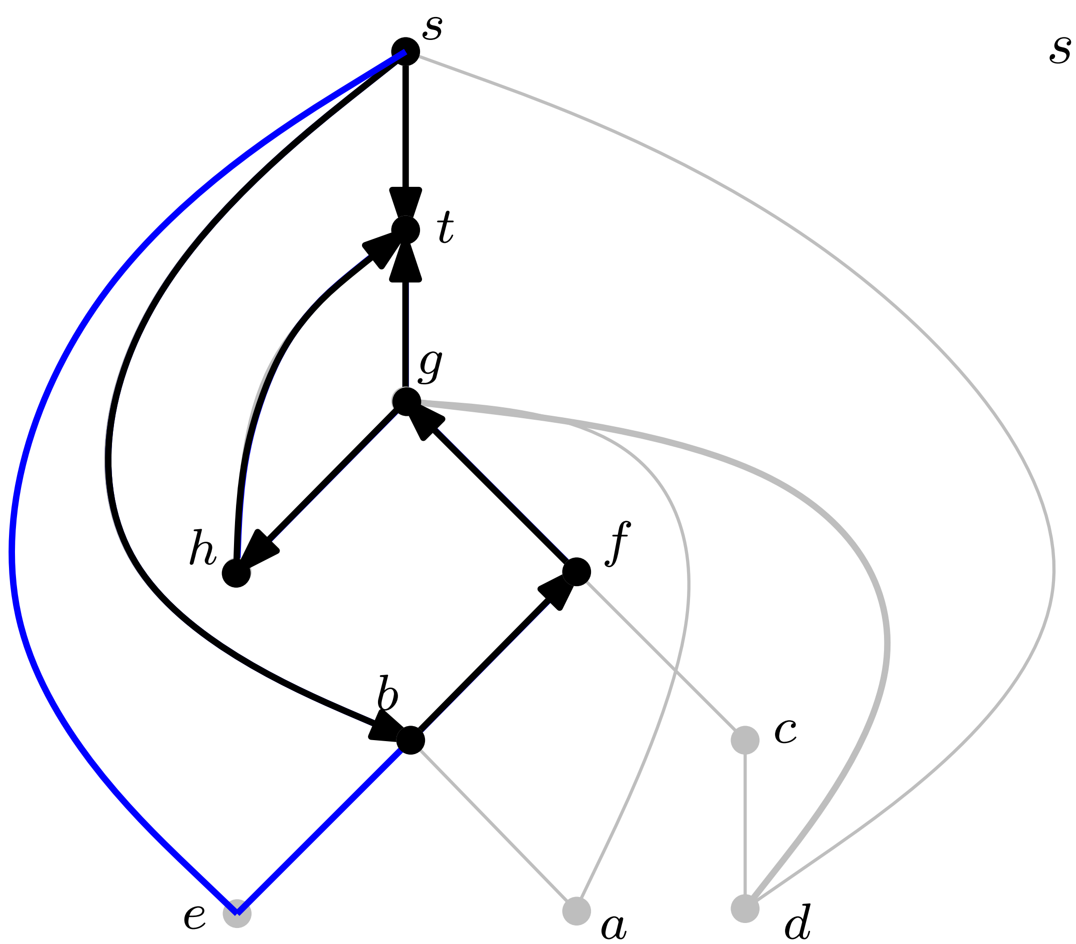
Algorithm: st-ordering (example)



$s, b, f, g, \underline{h}, t$

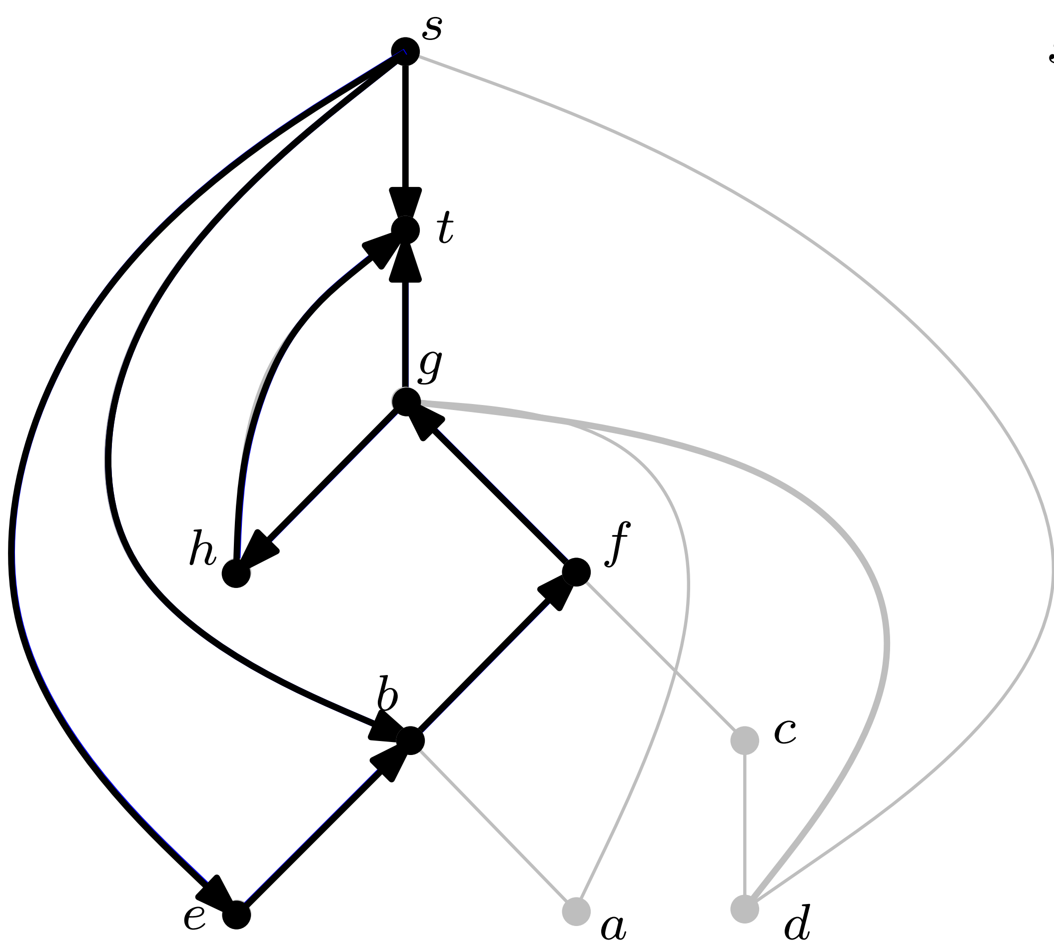
st-ordering

Algorithm: st-ordering (example)



$s, b, f, g, \underline{h}, t$

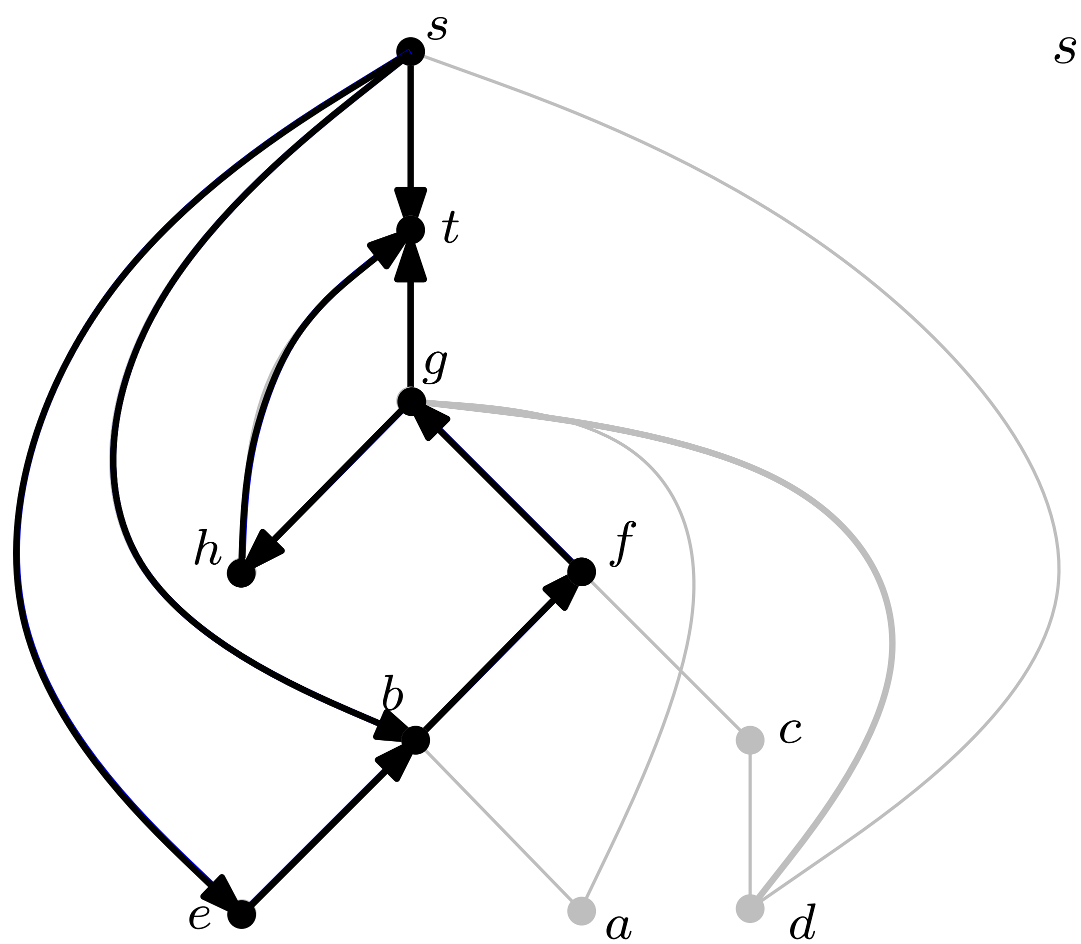
Algorithm: st-ordering (example)



$s, b, f, g, \underline{h}, t$

st-ordering

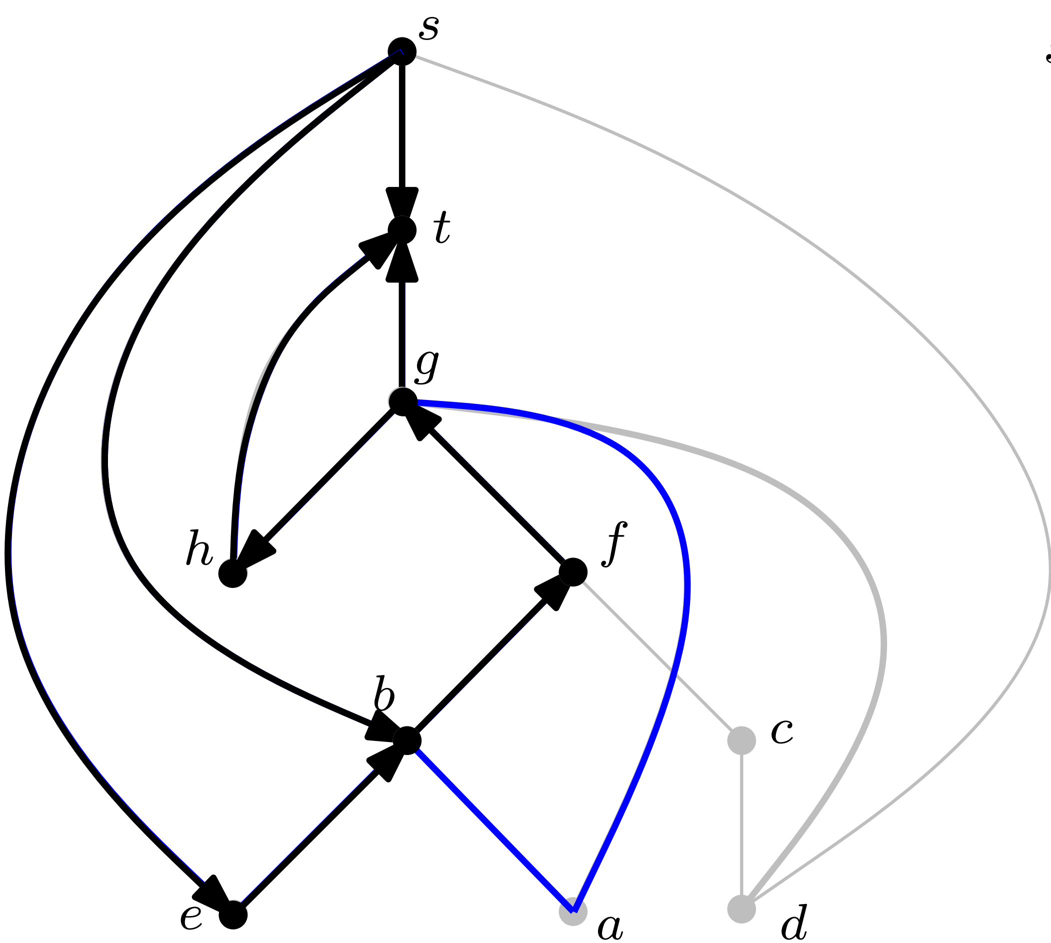
Algorithm: st-ordering (example)



$s, \underline{e}, b, f, g, h, t$

st-ordering

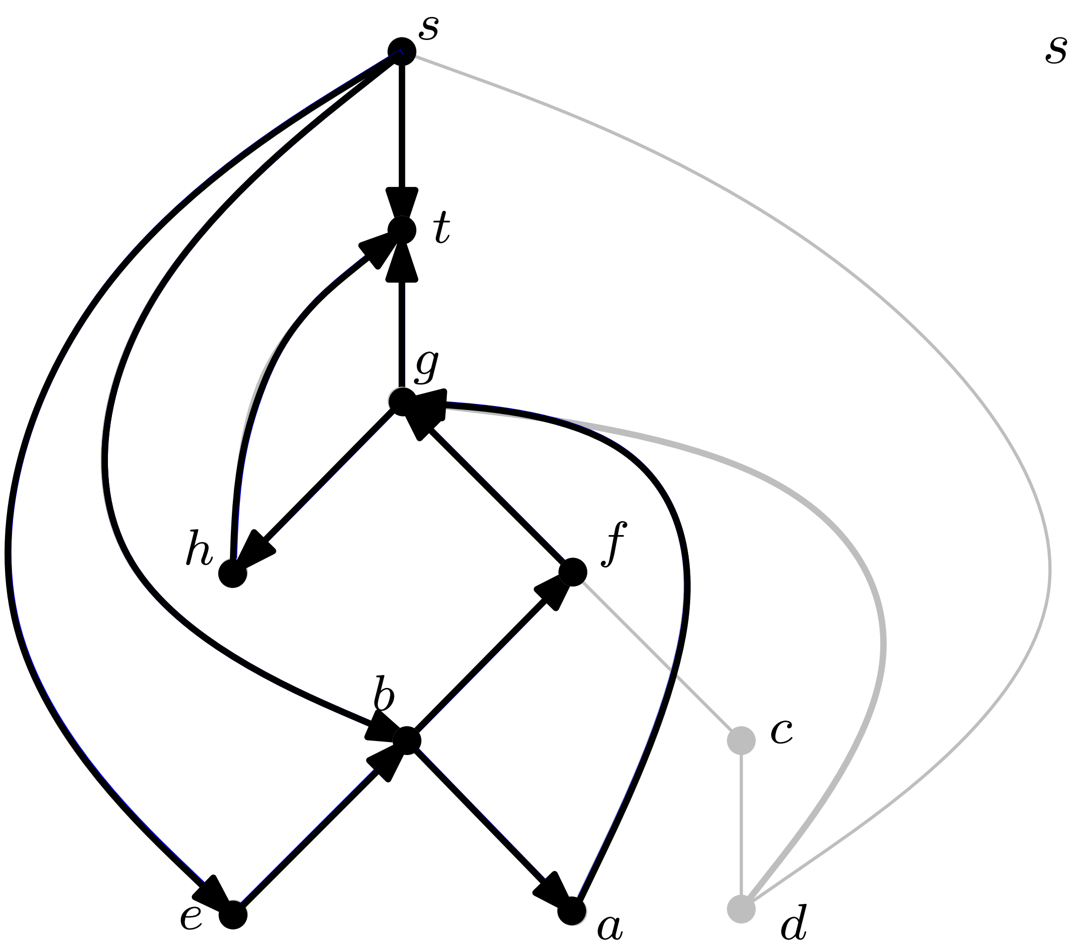
Algorithm: st-ordering (example)



$s, \underline{e}, b, f, g, h, t$

st-ordering

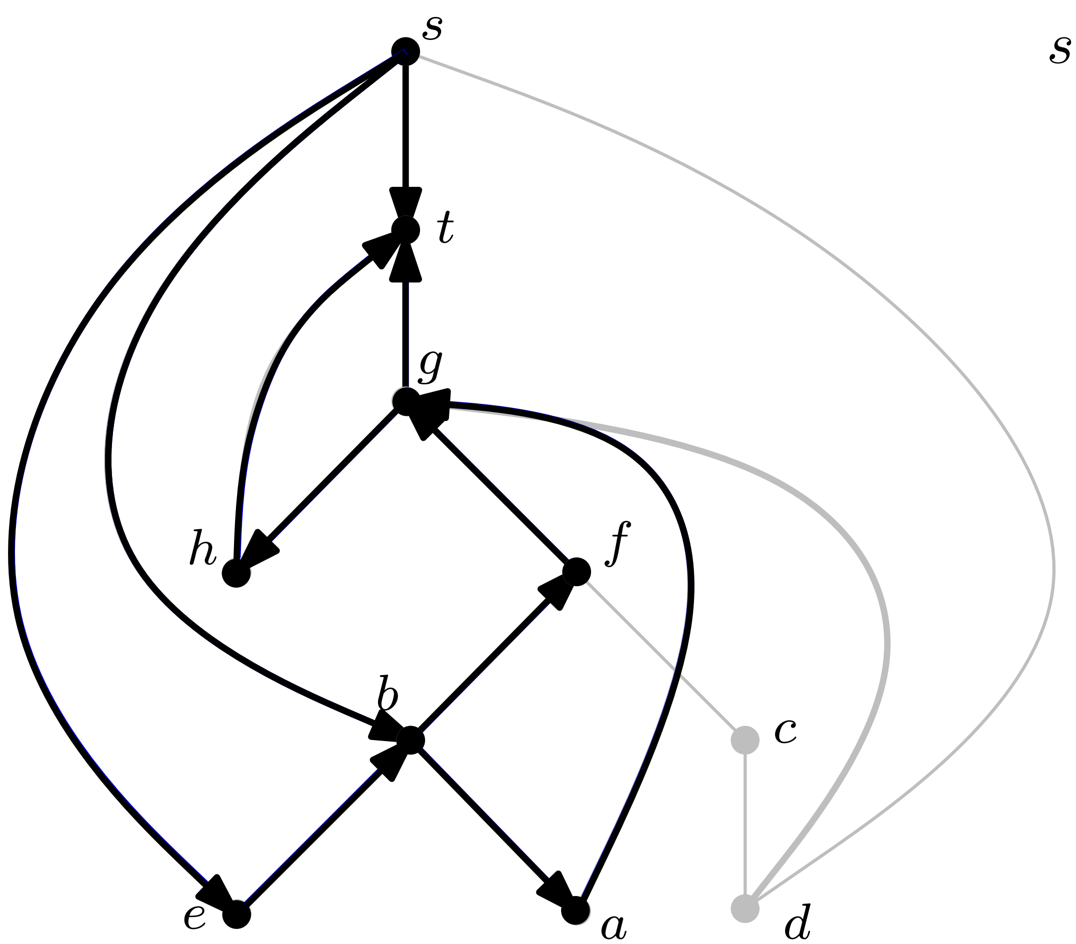
Algorithm: st-ordering (example)



$s, \underline{e}, b, f, g, h, t$

st-ordering

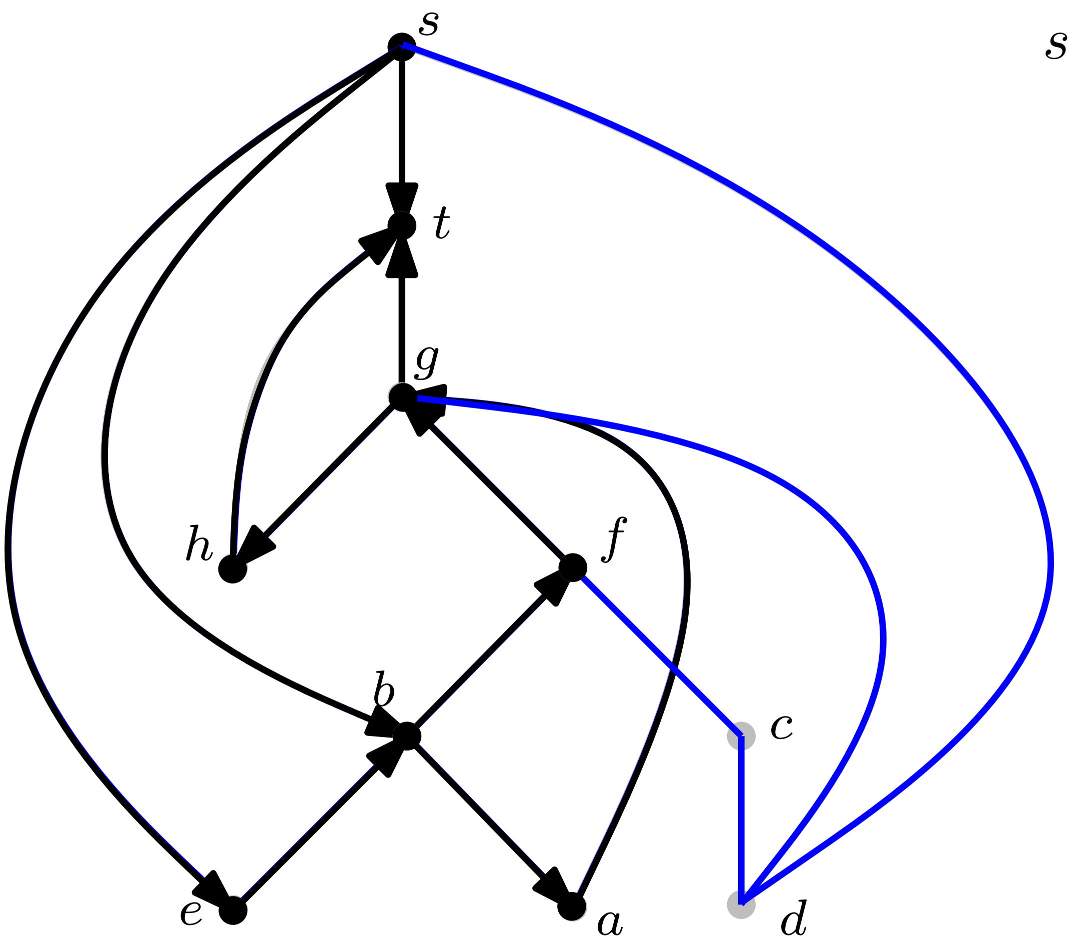
Algorithm: st-ordering (example)



$s, e, b, \underline{a}, f, g, h, t$

st-ordering

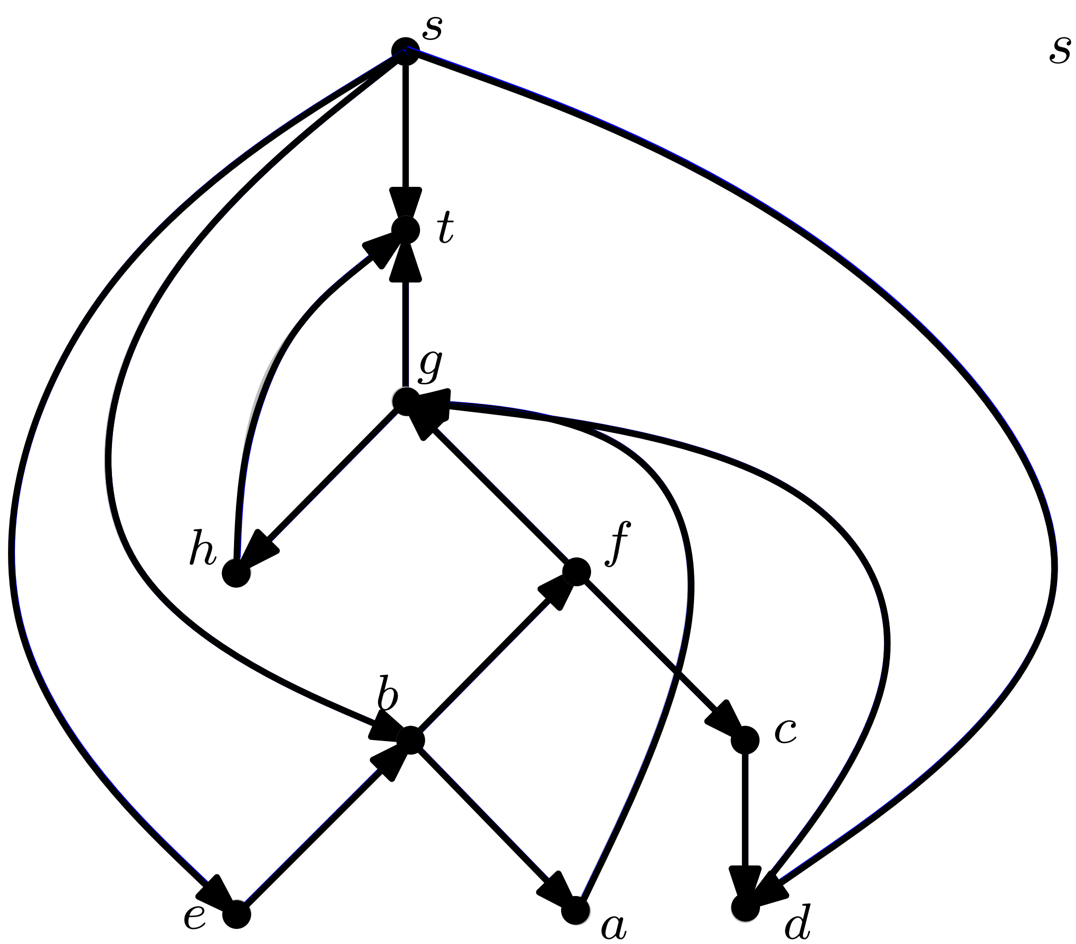
Algorithm: st-ordering (example)



$s, e, b, \underline{a}, f, g, h, t$

st-ordering

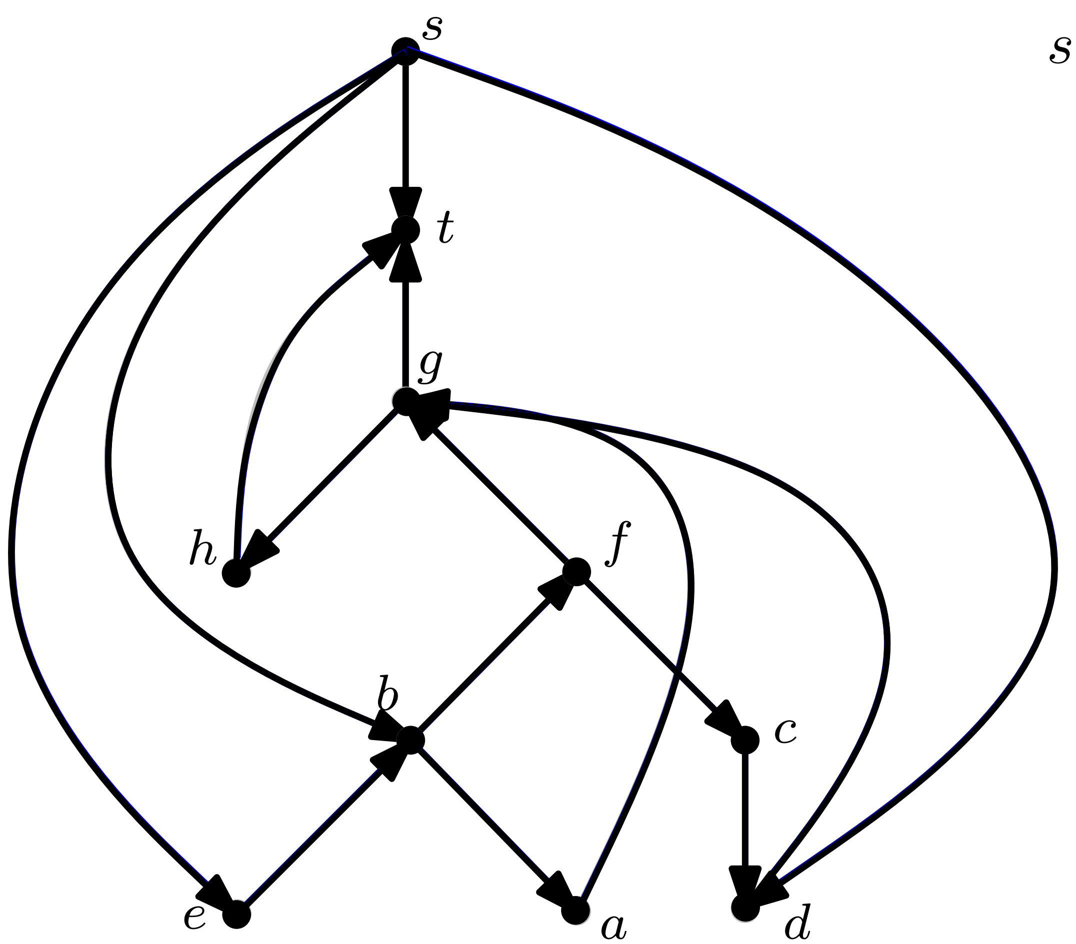
Algorithm: st-ordering (example)



$$s, e, b, \underline{a}, f, g, h, t$$

st-ordering

Algorithm: st-ordering (example)

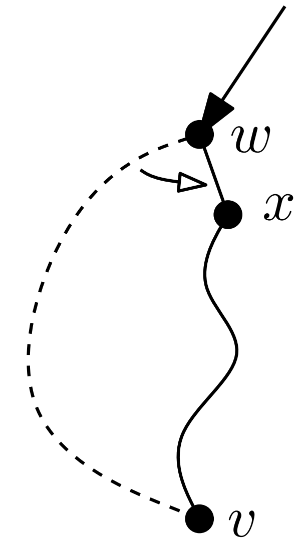


$s, e, b, a, f, \underline{c}, \underline{d}, g, h, t$

st-ordering

Algorithm st-ordering

Data: Undirected biconnected graph $G = (V, E)$, edge $\{s, t\} \in E$
Result: List L of nodes representing an st-ordering of G)



dfs(vertex v) begin
  $i \leftarrow i + 1$; $DFS[v] \leftarrow i$;
  while *there exists non-enumerated* $e = \{v, w\}$ **do**
    $DFS[e] \leftarrow DFS[v]$;
    if w *not enumerated* **then**
      $CHILDEDGE[v] \leftarrow e$; $PARENT[w] \leftarrow v$;
      $dfs(w)$;
    else
      $\{w, x\} \leftarrow CHILDEDGE[w]$; $D[\{w, x\}] \leftarrow D[\{w, x\}] \cup \{e\}$;
      if $x \in L$ **then** $process\_ears(w \rightarrow x)$;

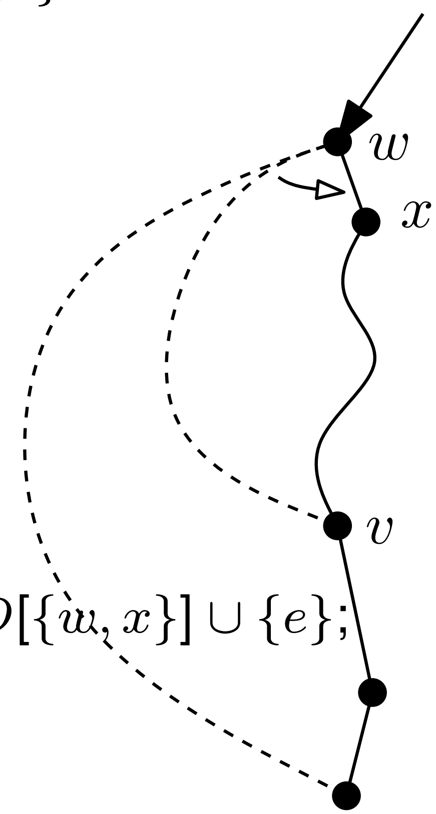
begin
  initialize L as $\{s,\ t\}$;
  $DFS[s] \leftarrow 1$; $i \leftarrow 1$; $DFS[\{s, t\}] \leftarrow 1$; $CHILDEDGE[s] \leftarrow \{s, t\}$;
  $dfs(t)$;

st-ordering

Algorithm st-ordering

Data: Undirected biconnected graph $G = (V, E)$, edge $\{s, t\} \in E$
Result: List L of nodes representing an st-ordering of G)

dfs(vertex v) begin
- $i \leftarrow i + 1$; $DFS[v] \leftarrow i$;
- **while** *there exists non-enumerated* $e = \{v, w\}$ **do**
  - $DFS[e] \leftarrow DFS[v]$;
  - **if** w *not enumerated* **then**
    - $CHILDEDGE[v] \leftarrow e$; $PARENT[w] \leftarrow v$;
    - $dfs(w)$;
  - **else**
    - $\{w, x\} \leftarrow CHILDEDGE[w]$; $D[\{w, x\}] \leftarrow D[\{w, x\}] \cup \{e\}$;
    - **if** $x \in L$ **then** $process\_ears(w \rightarrow x)$;



begin
- initialize L as $\{s,\ t\}$;
- $DFS[s] \leftarrow 1$; $i \leftarrow 1$; $DFS[\{s, t\}] \leftarrow 1$; $CHILDEDGE[s] \leftarrow \{s, t\}$;
- $dfs(t)$;

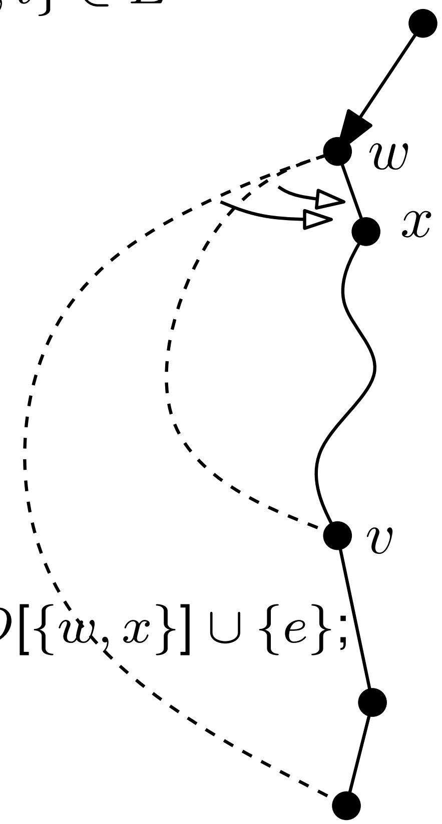
st-ordering

KIT Karlsruhe Institute of Technology

Algorithm st-ordering

Data: Undirected biconnected graph $G = (V, E)$, edge $\{s, t\} \in E$
Result: List L of nodes representing an st-ordering of G)

dfs(vertex v) begin
- $i \leftarrow i + 1$; $DFS[v] \leftarrow i$;
- **while** *there exists non-enumerated* $e = \{v, w\}$ **do**
  - $DFS[e] \leftarrow DFS[v]$;
  - **if** w *not enumerated* **then**
    - $CHILDEDGE[v] \leftarrow e$; $PARENT[w] \leftarrow v$;
    - $dfs(w)$;
  - **else**
    - $\{w, x\} \leftarrow CHILDEDGE[w]$; $D[\{w, x\}] \leftarrow D[\{w, x\}] \cup \{e\}$;
    - **if** $x \in L$ **then** $process\_ears(w \rightarrow x)$;



begin
- initialize L as $\{s, t\}$;
- $DFS[s] \leftarrow 1$; $i \leftarrow 1$; $DFS[\{s, t\}] \leftarrow 1$; $CHILDEDGE[s] \leftarrow \{s, t\}$;
- $dfs(t)$;

st-ordering

Function *process_ears*

process_ears(tree edge $w \to x$) begin

  foreach $v \hookrightarrow w \in D[w \to x]$ **do**

    $u \leftarrow v$;

    while $u \notin L$ **do** $u \leftarrow PARENT[u]$;

    $P \leftarrow (u \xrightarrow{*} v \hookrightarrow w)$;

    if $w \to x$ *is oriented from* w *to* x *(resp.from* x *to* w*)* **then**

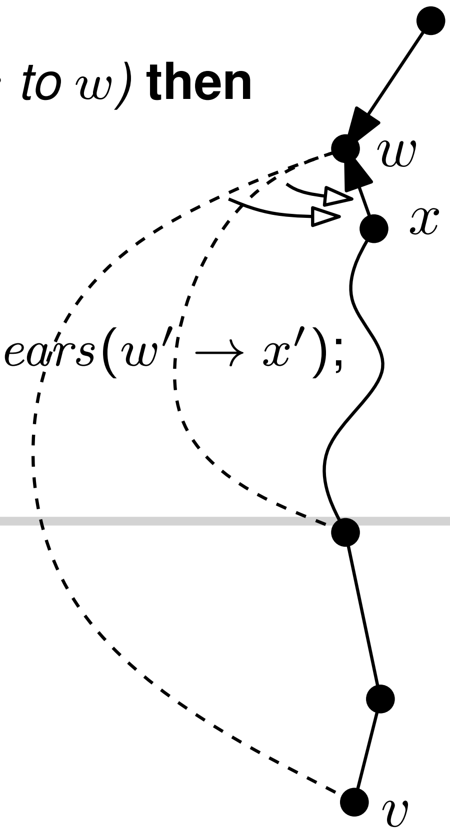
      orient P from w to u (resp. from u to w);

      paste the inner nodes of P to L

      before (resp. after) u ;

    foreach *tree edge* $w' \to x'$ *of* P **do** $process\_ears(w' \to x')$;

  $D[\{w, x\}] \leftarrow \emptyset$;



st-ordering

Theorem (Correctness and time complexity)

The described algorithm produces an st-ordering of a given biconnected graph $G = (V, E)$ in $O(E)$ time.

Proof

- Correctness can be proven by induction on ears. Notice that several new ears are added when function process_ears is called. Notice that after adition of an ear and its orientation, we have a biconnected st-graph and its topological ordering.

st-ordering

Theorem (Correctness and time complexity)

The described algorithm produces an st-ordering of a given biconnected graph $G = (V, E)$ in $O(E)$ time.

Proof

- Correctness can be proven by induction on ears. Notice that several new ears are added when function process_ears is called. Notice that after adition of an ear and its orientation, we have a biconnected st-graph and its topological ordering.

Lemma (Necessary for planarity of orthogonal drawing of planar graphs)

Let G be a plane graph and edge (s, t) on the boudary of G. Let $v_1, \ldots, v_n$ be an st-ordering of G. If G_i is the graph induced by the vertices $v_1, \ldots, v_i$ then vertex v_{i+1} lies on the outer face of G_i. (Exersize sheet 3)