Algorithms for graph visualization

Divide and Conquer - Series-Parallel Graphs
Series-parallel Graphs

Graph $G$ is **series-parallel**, if

- It contains a single edge $(s, t)$ ($s$-source, $t$-sink)
- It consists of two series-parallel graphs $G_1, G_2$ with sources $s_1, s_2$ and sinks $t_1, t_2$ which are combined using one of the following rules:

**Series composition:**
Identify $t_1$ and $s_2$, $s_1$ is the source of $G$, $t_2$ is the sink of $G$

**Parallel composition:**
Identify $s_1, s_2$ and set it to be source of $G$
Identify $t_1, t_2$ and set it to be sink of $G$
Series-parallel Graphs. Decomposition Tree.

Lemma

Series-parallel graphs are acyclic and planar.

In order to proof this statement we can use a decomposition tree of $G$, which is a binary tree $T$ with nodes of three types: S,P and Q-type.
Series-parallel Graphs. Decomposition Tree.

Lemma

Series-parallel graphs are acyclic and planar.

In order to prove this statement we can use a decomposition tree of $G$, which is a binary tree $T$ with nodes of three types: S, P and Q-type.

- If $G$ is a single edge, then the corresponding node is Q-node.
Series-parallel Graphs. Decomposition Tree.

Lemma

Series-parallel graphs are acyclic and planar.

In order to prove this statement we can use a decomposition tree of $G$, which is a binary tree $T$ with nodes of three types: S, P and Q-type.

- If $G$ is a single edge, then the corresponding node is Q-node.
- If $G$ is a parallel composition of $G_1$ (with tree $T_1$) and $G_2$ (with tree $T_2$), then the root of $T$ is P-node and $T_1$ is its left subtree, $T_2$ is its right subtree.
Series-parallel Graphs. Decomposition Tree.

Lemma

Series-parallel graphs are acyclic and planar.

In order to proof this statement we can use a decomposition tree of $G$, which is a binary tree $T$ with nodes of three types: S, P and Q-type.

- If $G$ is a single edge, then the corresponding node is Q-node
- If $G$ is a parallel composition of $G_1$ (with tree $T_1$) and $G_2$ (with tree $T_2$), then the root of $T$ is P-node and $T_1$ is its left subtree, $T_2$ is its right subtree
- If $G$ is a series composition of $G_1$ (with tree $T_1$) and $G_2$ (with tree $T_2$), then the root of $T$ is S-node and $T_1$ is its left subtree, $T_2$ is its right subtree
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.

**Flowcharts**

**PERT-Diagrams**

*Program Evaluation and Review Technique*

Flowcharts

PERT-Diagrams
(Program Evaluation and Review Technique)

Computational Complexity: Linear time algorithms for $NP$-hard problems (e.g. Maximum Matching, Maximum Independent Set, Hamiltonian Completion)
Straight-line Drawing of SP-Graphs

- Draw graph $G$ inside a right-angled isosceles bounding triangle $\Delta(G')$
Straight-line Drawing of SP-Graphs

- Draw graph $G$ inside a right-angled isosceles bounding triangle $\Delta(G')$
- Q-Nodes (Induction base):

\[ S \quad t \]
Straight-line Drawing of SP-Graphs

- Draw graph $G$ inside a right-angled isosceles bounding triangle $\Delta(G)$
- Q-Nodes (Induction base):
- S-Nodes (series composition)
Straight-line Drawing of SP-Graphs

- Draw graph $G$ inside a right-angled isosceles bounding triangle $\Delta(G)$
- Q-Nodes (Induction base):
- S-Nodes (series composition)
- P-Nodes (parallel composition)
Straight-line Drawing of SP-Graphs

- Draw graph $G$ inside a right-angled isosceles bounding triangle $\Delta(G)$
- Q-Nodes (Induction base):
- S-Nodes (series composition)
- P-Nodes (parallel composition)
Straight-line Drawing of SP-Graphs

- Draw graph $G$ inside a right-angled isosceles bounding triangle $\Delta(G)$
- Q-Nodes (Induction base):
- S-Nodes (series composition)
- P-Nodes (parallel composition)
Straight-line Drawing of SP-Graphs

- Draw graph \( G \) inside a right-angled isosceles bounding triangle \( \Delta(G) \)
- Q-Nodes (Induction base):
- S-Nodes (series composition)
- P-Nodes (parallel composition)

change embedding!
Straight-line Drawing of SP-Graphs

- Draw graph $G$ inside a right-angled isosceles bounding triangle $\Delta(G)$
- Q-Nodes (Induction base):
- S-Nodes (series composition)
- P-Nodes (parallel composition)

change embedding!
Straight-line Drawing of SP-Graphs

- Draw graph $G$ inside a right-angled isosceles bounding triangle $\Delta(G)$
- Q-Nodes (Induction base):
- S-Nodes (series composition)
- P-Nodes (parallel composition)

change embedding!
Straight-line Drawing of SP-Graphs

- Draw graph $G$ inside a right-angled isosceles bounding triangle $\Delta(G)$
- Q-Nodes (Induction base):
- S-Nodes (series composition)
- P-Nodes (parallel composition)
Straight-line Drawing of SP-Graphs

What makes parallel composition possible without creating crossings?
Straight-line Drawing of SP-Graphs

- What makes parallel composition possible without creating crossings?
Straight-line Drawing of SP-Graphs

What makes parallel composition possible without creating crossings?
What makes parallel composition possible without creating crossings?
Straight-line Drawing of SP-Graphs

What makes parallel composition possible without creating crossings?
Straight-line Drawing of SP-Graphs

What makes parallel composition possible without creating crossings?
Straight-line Drawing of SP-Graphs

What makes parallel composition possible without creating crossings?

does not contain any vertex

π

\[
\frac{\pi}{4}
\]
Straight-line Drawing of SP-Graphs

What makes parallel composition possible without creating crossings?

This condition can be preserved during the induction step.
What makes parallel composition possible without creating crossings?

This condition can be preserved during the induction step.

The area of the drawing is?
Straight-line Drawing of SP-Graphs

What makes parallel composition possible without creating crossings?

This condition can be preserved during the induction step.

The area of the drawing is? $O(m^2)$, $m$ is the number of edges
Straight-line Drawing of SP-Graphs

- What makes parallel composition possible without creating crossings?

- This condition can be preserved during the induction step.

- The area of the drawing is? $O(m^2)$, $m$ is the number of edges

Theorem

A series-parallel graph $G$ (with variable embedding) admits an upward straight-line drawing with $O(n^2)$ area. The isomorphic components of $G$ have congruent drawings up to a translation.
Lower Bound for the Area

Theorem [Bertolazzi et al. 94]
There exists a $2^n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$. 
Lower Bound for the Area

**Theorem [Bertolazzi et al. 94]**

There exists a $2n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$.

Proof:

![Diagram of graphs $G_0$, $G_n$, and $G_{n+1}$ with vertices $s_0$, $s_n$, $s_{n+1}$, $t_0$, $t_n$, $t_{n+1}$, and the graph $G_n$ highlighted in blue.](image)
Lower Bound for the Area

Theorem [Bertolazzi et al. 94]
There exists a $2n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$.

Proof:

![Diagram](image)
Lower Bound for the Area

**Theorem [Bertolazzi et al. 94]**
There exists a $2n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$.

Proof:
Lower Bound for the Area

Theorem [Bertolazzi et al. 94]

There exists a $2n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$.

Proof:
Lower Bound for the Area

Theorem [Bertolazzi et al. 94]
There exists a $2n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$.

Proof:
Lower Bound for the Area

**Theorem [Bertolazzi et al. 94]**
There exists a $2n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$.

**Proof:**

![Diagram showing graphs $G_0$, $G_n$, and $G_{n+1}$ with vertices $t_0$, $s_0$, $t_n$, $s_n$, $t_{n+1}$, and $s_{n+1}$, illustrating the area lower bound.](image)
Lower Bound for the Area

**Theorem [Bertolazzi et al. 94]**

There exists a $2n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$.

**Proof:**
Lower Bound for the Area

Theorem [Bertolazzi et al. 94]
There exists a $2n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$.

Proof:

\[ G_0 \]
\[ G_{n+1} \]
\[ \Delta_1 \]
\[ \Delta_2 \]
\[ t_{n+1} \]
\[ s_{n-1} \]
Lower Bound for the Area

Theorem [Bertolazzi et al. 94]

There exists a $2n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$.

Proof:

- We have that: $\text{Area}(\Pi) > 2 \cdot \text{Area}(G_n)$
Lower Bound for the Area

**Theorem [Bertolazzi et al. 94]**

There exists a $2n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$.

**Proof:**

- We have that: $Area(\Pi) > 2 \cdot Area(G_n)$
Lower Bound for the Area

Theorem [Bertolazzi et al. 94]
There exists a $2n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$. 

Proof:
- We have that: $\text{Area}(\Pi) > 2 \cdot \text{Area}(G_n)$
Lower Bound for the Area

Theorem [Bertolazzi et al. 94]
There exists a $2n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$.

Proof:
- We have that: $\text{Area}(\Pi) > 2 \cdot \text{Area}(G_n)$
- $\text{Area}(G_{n+1}) \geq 2 \cdot \text{Area}(\Pi)$
**Lower Bound for the Area**

**Theorem [Bertolazzi et al. 94]**

There exists a $2^n$-vertex series-parallel graph $G_n$ such that any upward planar drawing of $G_n$ respecting embedding requires area $\Omega(4^n)$.

**Proof:**

- We have that: $\text{Area}(\Pi) > 2 \cdot \text{Area}(G_n)$
- $\text{Area}(G_{n+1}) \geq 2 \cdot \text{Area}(\Pi)$
- $\text{Area}(G_{n+1}) \geq 4 \cdot \text{Area}(G_n)$
Property of the Algorithm
Property of the Algorithm

Algorithm

nicer???
Property of the Algorithm

Algorithm

nicer???
Property of the Algorithm

Algorithm

nicer???
Property of the Algorithm

Algorithm

necer???
Property of the Algorithm

Graph $G = (\{a, b, c, d, e, f, g, h\},$
\{(a, h), (a, e), (b, g), (b, f), (c, g), (c, f), (d, e),
(d, h), (e, f), (h, g)\})$
Property of the Algorithm

- Graph $G = (\{a, b, c, d, e, f, g, h\},$
  $\{(a, h), (a, e), (b, g), (b, f), (c, g), (c, f), (d, e),
  (d, h), (e, f), (h, g)\})$

- Let $G''$ be $G$ where $b \rightarrow c \rightarrow b, a \rightarrow d \rightarrow a$. 
Property of the Algorithm

Graph $G = \{a, b, c, d, e, f, g, h\}$,
$\{(a, h), (a, e), (b, g), (b, f), (c, g), (c, f), (d, e), (d, h), (e, f), (h, g)\}$

Let $G'$ be $G$ where $b \rightarrow c \rightarrow b$, $a \rightarrow d \rightarrow a$.

$G$ and $G'$ are isomorphic.
Graph Automorphism

Definition: Automorphism of a digraph

An automorphism of a directed graph $G = (V, E)$ is a permutation of the vertex set which preserves adjacency of the vertices and either preserves or reverses all the directions of the edges:

- $(u, v) \in E \iff (\pi(u), \pi(v)) \in E$, or
- $(u, v) \in E \iff (\pi(v), \pi(u)) \in E$
**Graph Automorphism**

**Definition: Automorphism of a digraph**

An **automorphism** of a directed graph $G = (V, E)$ is a permutation of the vertex set which preserves adjacency of the vertices and either preserves or reverses all the directions of the edges:

1. $(u, v) \in E \iff (\pi(u), \pi(v)) \in E$, or
2. $(u, v) \in E \iff (\pi(v), \pi(u)) \in E$

- The set of all automorphisms (direction preserving and reversing) forms the **automorphism group** of $G$. 
Graph Automorphism

Definition: Automorphism of a digraph

An **automorphism** of a directed graph $G = (V, E)$ is a permutation of the vertex set which preserves adjacency of the vertices and either preserves or reverses all the directions of the edges:

- $(u, v) \in E \iff (\pi(u), \pi(v)) \in E$, or
- $(u, v) \in E \iff (\pi(v), \pi(u)) \in E$

- The set of all automorphisms (direction preserving and reversing) forms the **automorphism group** of $G$.
- Finding an automorphism group of a graph is **isomorphism complete**, that is equivalent to testing whether two graphs are isomorphic.
Graph Automorphism

Definition: Automorphism of a digraph

An **automorphism** of a directed graph \( G = (V, E) \) is a permutation of the vertex set which preserves adjacency of the vertices and either preserves or reverses all the directions of the edges:

\[
\begin{align*}
(u, v) \in E &\iff (\pi(u), \pi(v)) \in E, \text{ or} \\
(u, v) \in E &\iff (\pi(v), \pi(u)) \in E
\end{align*}
\]

- The set of all automorphisms (direction preserving and reversing) forms the **automorphism group** of \( G \).
- Finding an automorphism group of a graph is **isomorphism complete**, that is equivalent to testing whether two graphs are isomorphic.
- For planar graphs, graphs with bounded degree isomorphism problem has polynomial-time algorithms.
Different types of automorphism:
Different types of automorphism:

Automorphism $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 1$ is geometrically representable, while $1 \rightarrow 2 \rightarrow 3 \rightarrow 1$ is not.

Automorphism $1 \rightarrow 2 \rightarrow 3 \rightarrow 1$ is geometrically representable, while $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 1$ is not.

Automorphism $1 \rightarrow 2 \rightarrow 3 \rightarrow 1$, $4 \rightarrow 5 \rightarrow 4$ is not geometrically representable.
Geometric Automorphism

- Different types of automorphism:

  ![Graph 1](image1.png)
  ![Graph 2](image2.png)
  ![Graph 3](image3.png)

  Automorphism 1 → 2 → 3 → 4 → 1 is geometrically representable, while 1 → 2 → 3 → 1 is not.
  Automorphism 1 → 2 → 3 → 1 is geometrically representable, while 1 → 2 → 3 → 4 → 1 is not.
  Automorphism 1 → 2 → 3 → 1, 4 → 5 → 4 is not geometrically representable.

- An automorphism group $P$ of a graph is geometric, if there exists a drawing of $G$ that displays each element of $P$ as a symmetry.

- For general graphs it is NP-hard to find a geometric automorphism of a graph.

- For planar graphs, planar geometric automorphisms can be found in polynomial time. For outerplanar graphs and trees in linear time.
Symmetries in SP-Graphs

\[ \pi_{\text{vert}} \]

\[ \pi(u) \]

\[ \pi(v) \]
Symmetries in SP-Graphs

\[ \pi_{\text{vert}} \]

\[ \pi_{\text{hor}} \]
Symmetries in SP-Graphs

\[ \pi_{\text{vert}} \]

\[ \pi_{\text{hor}} \]

\[ \pi_{\text{rot}} \]
Symmetries in SP-Graphs

\[ \{ \pi_{\text{vert}}, \pi_{\text{hor}}, \pi_{\text{rot}} \} \]
Symmetries in SP-Graphs

A geometric automorphism group $P$ of a graph $G$ is upward planar, if there exists an upward planar drawing of $G$ that displays each element of $P$ as a symmetry.
Symmetries in SP-Graphs

A geometric automorphism group $P$ of a graph $G$ is upward planar, if there exists an upward planar drawing of $G$ that displays each element of $P$ as a symmetry.

How does a geometric automorphism group for a series-parallel graph look like?
Symmetries in SP-Graphs

Theorem (Hong, Eades, Lee ’00)

An upward planar automorphism group of a series-parallel digraph is either

\[
\{ \text{id} \}, \quad \{ \text{id}, \pi \} \quad \text{with} \quad \pi \in \{ \pi_{\text{vert}}, \pi_{\text{hor}}, \pi_{\text{rot}} \}, \\
\{ \text{id}, \pi_{\text{vert}}, \pi_{\text{hor}}, \pi_{\text{rot}} \}.
\]
Vertical Automorphism
Vertical Automorphism
Vertical Automorphism
Vertical Automorphism
Vertical Automorphism

- $\text{code}(G)$ - two graphs at the same level have the same code iff they are isomorphic
- $\text{tuple}(G)$ - codes of the children
Vertical Automorphism

- **code**($G$) - two graphs at the same level have the same code iff they are isomorphic
- **tuple**($G'$) - codes of the children

$code(G) = 1$

$tuple(G) = <1, 1, 2>$. 

code = 1

code=1

code = 2
**Vertical Automorphism**

- \( \text{code}(G) \) - two graphs at the same level have the same code iff they are isomorphic
- \( \text{tuple}(G) \) - codes of the children

\[
\text{tuple}(G) = \langle 1, 1, 2 \rangle
\]

Why sorted?
Vertical Automorphism

- **code(G)** - two graphs at the same level have the same code iff they are isomorphic
- **tuple(G)** - codes of the children

Why sorted?

tuple(G) = <1, 1, 2>

code = 1

code = 2

code = 1

code = 2

compare :

<1 2 3 2 1 2 3 3>

<3 2 2 1 1 3 3 3>
**Vertical Automorphism**

- \( \text{code}(G) \) - two graphs at the same level have the same code iff they are isomorphic
- \( \text{tuple}(G) \) - codes of the children

\[ \text{code}(G) = 1 \]

\[ \text{tuple}(G) = \langle 1, 1, 2 \rangle \]

Why sorted?

\[ \text{tuple}(G) = \langle 1, 1, 2 \rangle \]

\[ \text{tuple}(G) = \langle 1, 1, 2 \rangle \]

Compare :)
Algorithm constructing a Canonical Labeling

- Set $\text{tuple}(G_i) = \langle 0 \rangle$ for all Q-nodes $G_i$ of $G$. 

Vertical Automorphism
Vertical Automorphism

Algorithm constructing a Canonical Labeling

- Set $\text{tuple}(G_i) = \langle 0 \rangle$ for all Q-nodes $G_i$ of $G$.
- For each $t = \max \text{depth}(G), \ldots, 0$
  - For each S- or P-node $G'$ at depth $t$ with children $G_1, \ldots, G_k$ set $\text{tuple}(G') = \langle \text{code}(G_1), \ldots, \text{code}(G_k) \rangle$. If $G'$ is a P-node, sort $\text{tuple}(G')$ in non-decreasing order.
Vertical Automorphism

Algorithm constructing a Canonical Labeling

- Set $\text{tuple}(G_i) = \langle 0 \rangle$ for all Q-nodes $G_i$ of $G$.
- For each $t = \max \text{depth}(G), \ldots, 0$
  - For each S- or P-node $G'$ at depth $t$ with children $G_1, \ldots, G_k$ set $\text{tuple}(G') = \langle \text{code}(G_1), \ldots, \text{code}(G_k) \rangle$. If $G'$ is a P-node, sort $\text{tuple}(G')$ in non-decreasing order.
  - Sort all the nodes at depth $t$ lexicographically according to tuples.
Vertical Automorphism

Algorithm constructing a Canonical Labeling

- Set \( \text{tuple}(G_i) = \langle 0 \rangle \) for all Q-nodes \( G_i \) of \( G \).
- For each \( t = \max \text{depth}(G), \ldots, 0 \)
  - For each S- or P-node \( G' \) at depth \( t \) with children \( G_1, \ldots, G_k \) set \( \text{tuple}(G') = \langle \text{code}(G_1), \ldots, \text{code}(G_k) \rangle \). If \( G' \) is a P-node, sort \( \text{tuple}(G') \) in non-decreasing order.
  - Sort all the nodes at depth \( t \) lexicographically according to tuples.
  - For each component \( G' \) at depth \( t \), compute \( \text{code}(G') \) as follows. Assign the integer 1 to those components represented by the first distinct tuple, assign 2 to the components with the second type of tuple, and etc.
Algorithm constructing a Canonical Labeling

- Set $\text{tuple}(G_i) = \langle 0 \rangle$ for all Q-nodes $G_i$ of $G$.
- For each $t = \max \text{depth}(G), \ldots, 0$
  - For each S- or P-node $G'$ at depth $t$ with children $G_1, \ldots, G_k$ set $\text{tuple}(G') = \langle \text{code}(G_1), \ldots, \text{code}(G_k) \rangle$. If $G'$ is a P-node, sort $\text{tuple}(G')$ in non-decreasing order.
- Sort all the nodes at depth $t$ lexicographically according to tuples.
- For each component $G'$ at depth $t$, compute $\text{code}(G')$ as follows. Assign the integer 1 to those components represented by the first distinct tuple, assign 2 to the components with the second type of tuple, and etc.

Lemma

Two nodes $u$ and $v$ at the same depth of the decomposition tree of $G$ represent isomorphic subgraphs of $G$ iff $\text{code}(u) = \text{code}(v)$. 

Vertical Automorphism
Vertical Automorphism
Vertical Automorphism
Vertical Automorphism
Vertical Automorphism
Vertical Automorphism
Vertical Automorphism
Vertical Automorphism
Vertical Automorphism
Vertical Automorphism

- Let \( G \) be composed out of \( G_1 \ldots G_n \) through series or parallel composition, \( \text{tuple}(G) \) contains the codes of \( G_1, \ldots, G_n \).
- How can we use \( \text{tuple}(G) \) do decide whether \( G \) has a vertical automorphism?

\( G \) is an S-node
Let $G$ be composed out of $G_1 \ldots G_n$ through series or parallel composition, $\text{tuple}(G)$ contains the codes of $G_1, \ldots, G_n$.

How can we use $\text{tuple}(G)$ do decide whether $G$ has a vertical automorphism?

Lemma (Hong, Eades, Lee ’00)

If $G$ is an S-node, then $G$ has a vertical automorphism iff each of $G_1, \ldots, G_k$ has a vertical automorphism.
Vertical Automorphism

- Let $G$ be composed out of $G_1 \ldots G_n$ through series or parallel composition, $\text{tuple}(G)$ contains the codes of $G_1, \ldots, G_n$.
- How can we use $\text{tuple}(G)$ to decide whether $G$ has a vertical automorphism?

**Lemma (Hong, Eades, Lee '00)**

If $G$ is an S-node, then $G$ has a vertical automorphism iff each of $G_1, \ldots, G_k$ has a vertical automorphism.

**Proof:**

- Assume $G$ has a vertical automorphism $\alpha$
Vertical Automorphism

- Let $G$ be composed out of $G_1 \ldots G_n$ through series or parallel composition, $tuple(G)$ contains the codes of $G_1, \ldots, G_n$.
- How can we use $tuple(G)$ to decide whether $G$ has a vertical automorphism?

Lemma (Hong, Eades, Lee '00)

If $G$ is an S-node, then $G$ has a vertical automorphism iff each of $G_1, \ldots, G_k$ has a vertical automorphism.

Proof:
- Assume $G$ has a vertical automorphism $\alpha$
- Then $\alpha$ “fixes” all the components

$G$ is an S-node
Vertical Automorphism

- Let $G$ be composed out of $G_1 \ldots G_n$ through series or parallel composition, $\text{tuple}(G)$ contains the codes of $G_1, \ldots, G_n$.
- How can we use $\text{tuple}(G)$ to decide whether $G$ has a vertical automorphism?

**Lemma (Hong, Eades, Lee ’00)**

If $G$ is an S-node, then $G$ has a vertical automorphism iff each of $G_1, \ldots, G_k$ has a vertical automorphism.

**Proof:**

- Assume $G$ has a vertical automorphism $\alpha$
- Then $\alpha$ “fixes” all the components
- Therefore each of the series components has a vertical automorhism

$G$ is an S-node
**Vertical Automorphism**

- Let $G$ be composed out of $G_1 \ldots G_n$ through series or parallel composition, $\text{tuple}(G)$ contains the codes of $G_1, \ldots, G_n$.
- How can we use $\text{tuple}(G)$ do decide whether $G$ has a vertical automorphism?

**Lemma (Hong, Eades, Lee ’00)**

If $G$ is an S-node, then $G$ has a vertical automorphism iff each of $G_1, \ldots, G_k$ has a vertical automorphism.

**Proof:**

- Assume $G$ has a vertical automorphism $\alpha$
- Then $\alpha$ “fixes” all the components
- Therefore each of the series components has a vertical automorphism
- If each of $G_1, \ldots, G_n$ has a vertical isomorphism, arrange them as in Figure.
Lemma (Hong, Eades, Lee ’00)

If $G$ is a P-node, consider a partition of $C_j = \{G_i : 1 \leq i \leq k, \text{code}(G_i) = j\}$, $j = 1, \ldots, k$ into classes of isomorphic graphs.

- If $\forall j$, $|C_j|$ are even $\Rightarrow$ has a vertical automorphism.
- If there exists a unique $j$, such that $|C_j|$ is odd $\Rightarrow G$ has a vertical automorphism iff graphs of $C_j$ have a vertical automorphism.
- If there exists $|C_i|, |C_j|$ with $i \neq j$, both odd $\Rightarrow G$ does not have a vertical automorphism.

Proof:

- Arrange components as in Figure.

$G$ is P-node, $\text{tuple}(G) = <1 \ldots 1, 2 \ldots 2, \ldots>$

\[\text{even even}\]
Lemma (Hong, Eades, Lee ’00)

If $G$ is a P-node, consider a partition of $C_j = \{G_i : 1 \leq i \leq k, \text{code}(G_i) = j\}$, $j = 1, \ldots, k$ into classes of isomorphic graphs.

- If $\forall j$, $\vert C_j \vert$ are even $\Rightarrow$ has a vertical automorphism.
- If there exists a unique $j$, such that $\vert C_j \vert$ is odd $\Rightarrow G$ has a vertical automorphism iff graphs of $C_j$ have a vertical automorphism.
- If there exists $\vert C_i \vert, \vert C_j \vert$ with $i \neq j$, both odd $\Rightarrow G$ does not have a vertical automorphism.

Proof:

$$\text{tuple}(G) = \langle 1 \ldots 1, 2 \ldots 2, 3 \ldots 3, \ldots \rangle$$
**Lemma (Hong, Eades, Lee ’00)**

If $G$ is a P-node, consider a partition of $C_j = \{G_i : 1 \leq i \leq k, \text{code}(G_i) = j\}$, $j = 1, \ldots, k$ into classes of isomorphic graphs.

- If $\forall j$, $|C_j|$ are even $\Rightarrow$ has a vertical automorphism.
- If there exists a unique $j$, such that $|C_j|$ is odd $\Rightarrow G$ has a vertical automorphism iff graphs of $C_j$ have a vertical automorphism.
- If there exists $|C_i|$, $|C_j|$ with $i \neq j$, both odd $\Rightarrow G$ does not have a vertical automorphism.

**Proof:**

- Any vertical automorphism “fixes” a member of $C_j$, therefore it has a vertical automorphism.

\[
tuple(G) = < 1 \ldots 1, 2 \ldots 2, 3 \ldots 3, \ldots >
\]

\[
\begin{array}{c}
\text{odd} \\
\text{even} \\
\text{even}
\end{array}
\]
Lemma (Hong, Eades, Lee ’00)

If $G$ is a P-node, consider a partition of $C_j = \{G_i : 1 \leq i \leq k, code(G_i) = j\}$, $j = 1, \ldots, k$ into classes of isomorphic graphs.

- If $\forall j$, $|C_j|$ are even $\Rightarrow$ has a vertical automorphism.
- If there exists a unique $j$, such that $|C_j|$ is odd $\Rightarrow G$ has a vertical automorphism iff graphs of $C_j$ have a vertical automorphism.
- If there exists $|C_i|, |C_j|$ with $i \neq j$, both odd $\Rightarrow G$ does not have a vertical automorphism.

Proof:

- Any vertical automorphism “fixes” a member of $C_j$, therefore it has a vertical automorphism.
- Conversely, arrange as in figure.

\[
tuple(G) = \langle 1 \ldots 1, 2 \ldots 2, 3 \ldots 3, \ldots \rangle
\]

\[
tuple(G) = \langle \underbrace{1 \ldots 1}_\text{odd}, \underbrace{2 \ldots 2}_\text{even}, \underbrace{3 \ldots 3}_\text{even}, \ldots \rangle
\]
**Vertical Automorphism**

**Lemma (Hong, Eades, Lee ’00)**

If \( G \) is a P-node, consider a partition of \( C_j = \{G_i : 1 \leq i \leq k, \text{code}(G_i) = j\} \), \( j = 1, \ldots, k \) into classes of isomorphic graphs.

- If \( \forall j \), \( |C_j| \) are even \( \Rightarrow \) has a vertical automorphism.
- If there exists a unique \( j \), such that \( |C_j| \) is odd \( \Rightarrow \) \( G \) has a vertical automorphism iff graphs of \( C_j \) have a vertical automorphism.
- If there exists \( |C_i|, |C_j| \) with \( i \neq j \), both odd \( \Rightarrow \) \( G \) does not have a vertical automorphism.

\[
tuple(G) = <1 \ldots 1, 2 \ldots 2, 3 \ldots 3, \ldots>
\]

\( \text{odd} \quad \text{odd} \quad \text{even} \quad \cdots \)
**Vertical Automorphism**

**Lemma (Hong, Eades, Lee ’00)**

If $G$ is a P-node, consider a partition of $C_j = \{G_i : 1 \leq i \leq k, \text{code}(G_i) = j\}$, $j = 1, \ldots, k$ into classes of isomorphic graphs.

- If $\forall j$, $|C_j|$ are even $\Rightarrow$ has a vertical automorphism.
- If there exists a unique $j$, such that $|C_j|$ is odd $\Rightarrow G$ has a vertical automorphism iff graphs of $C_j$ have a vertical automorphism.
- If there exists $|C_i|, |C_j|$ with $i \neq j$, both odd $\Rightarrow G$ does not have a vertical automorphism.

**Proof:**

- Any vertical automorphism has to “fix” two distinct components.

$$\text{tuple}(G) = <1 \ldots 1, 2 \ldots 2, 3 \ldots 3, \ldots>$$

odd  odd  even
Vertical Automorphism

Lemma (Hong, Eades, Lee ’00)

If \( G \) is a P-node, consider a partition of \( C_j = \{G_i : 1 \leq i \leq k, \text{code}(G_i) = j\} \), \( j = 1, \ldots, k \) into classes of isomorphic graphs.

- If \( \forall j \), \( |C_j| \) are even \( \Rightarrow \) has a vertical automorphism.
- If there exists a unique \( j \), such that \( |C_j| \) is odd \( \Rightarrow G \) has a vertical automorphism
  iff graphs of \( C_j \) have a vertical automorphism.
- If there exists \( |C_i|, |C_j| \) with \( i \neq j \), both odd \( \Rightarrow G \) does not have a vertical automorphism.

\[ \text{tuple}(G) = <1 \ldots 1, 2 \ldots 2, 3 \ldots 3, \ldots > \]

Proof:

- Any vertical automorphism has to “fix” two distinct components.
- In both components we can find a path on which some vertices are aligned on the axis. Contradicts planarity.
Vertical Automorphism

**Theorem (Hong, Eades, Lee ’00)**

Given a decomposition tree of a series-parallel graph and its canonical labeling. Let $G$ be a component which consists from $G_1, \ldots, G_k$ through series or parallel composition.

- If $G$ is an S-node, then $G$ has a vertical automorphism iff each of $G_1, \ldots, G_k$ has a vertical automorphism.
Vertical Automorphism

Theorem (Hong, Eades, Lee ’00)

Given a decomposition tree of a series-parallel graph and its canonical labeling. Let $G$ be a component which consists from $G_1, \ldots, G_k$ through series or parallel composition.

- If $G$ is an S-node, then $G$ has a vertical automorphism iff each of $G_1, \ldots, G_k$ has a vertical automorphism.

- If $G$ is a P-node, consider a partition of $C_j = \{G_i : 1 \leq i \leq k, \text{code}(G_i) = j\}$, $j = 1, \ldots, k$ into classes of isomorphic graphs.
  - If $\forall j$, $|C_j|$ are even $\Rightarrow$ has a vertical automorphism.
Vertical Automorphism

Theorem (Hong, Eades, Lee ’00)

Given a decomposition tree of a series-parallel graph and its canonical labeling. Let $G$ be a component which consists from $G_1, \ldots, G_k$ through series or parallel composition.

- If $G$ is an S-node, then $G$ has a vertical automorphism iff each of $G_1, \ldots, G_k$ has a vertical automorphism.

- If $G$ is a P-node, consider a partition of $C_j = \{G_i : 1 \leq i \leq k, \text{code}(G_i) = j\}$, $j = 1, \ldots, k$ into classes of isomorphic graphs.
  - If $\forall j$, $|C_j|$ are even $\Rightarrow$ has a vertical automorphism.
  - If there exists a unique $j$, such that $|C_j|$ is odd $\Rightarrow G$ has a vertical automorphism if graphs of $C_j$ have a vertical automorphism.
Vertical Automorphism

Theorem (Hong, Eades, Lee ’00)

Given a decomposition tree of a series-parallel graph and its canonical labeling. Let $G$ be a component which consists from $G_1, \ldots, G_k$ through series or parallel composition.

- If $G$ is an S-node, then $G$ has a vertical automorphism iff each of $G_1, \ldots, G_k$ has a vertical automorphism.

- If $G$ is a P-node, consider a partition of $C_j = \{G_i : 1 \leq i \leq k, \text{code}(G_i) = j\}$, $j = 1, \ldots, k$ into classes of isomorphic graphs.
  - If $\forall j$, $|C_j|$ are even $\Rightarrow$ has a vertical automorphism.
  - If there exists a unique $j$, such that $|C_j|$ is odd $\Rightarrow$ $G$ has a vertical automorphism iff graphs of $C_j$ have a vertical automorphism.
  - If there exists $|C_i|, |C_j|$ with $i \neq j$, both odd $\Rightarrow G$ does not have a vertical automorphism.