Characterization of Unlabeled Level Planar Graphs

J. Joseph Fowler and Stephen G. Kobourov

Department of Computer Science, The University of Arizona

The 15th International Symposium on Graph Drawing (GD 2007)

Outline

- Background

Outline

- Background
  - Motivation
  - Definitions
  - Previous Work
  - New Results

Outline

- Background
- Unlabeled Level Planar Trees

Outline

- Background
- Unlabeled Level Planar Trees
  - Forbidden Trees

Outline

- Background
- Unlabeled Level Planar Trees
  - Forbidden Trees
  - Drawing Algorithms

Outline

- Background
- Unlabeled Level Planar Trees
- Unlabeled Level Planar Graphs

Outline

- Background
- Unlabeled Level Planar Trees
- Unlabeled Level Planar Graphs
  - Forbidden Graphs

Outline

- Background
- Unlabeled Level Planar Trees
- Unlabeled Level Planar Graphs
  - Forbidden Graphs
  - Extended Drawing Algorithms

Motivation – Level Planarity

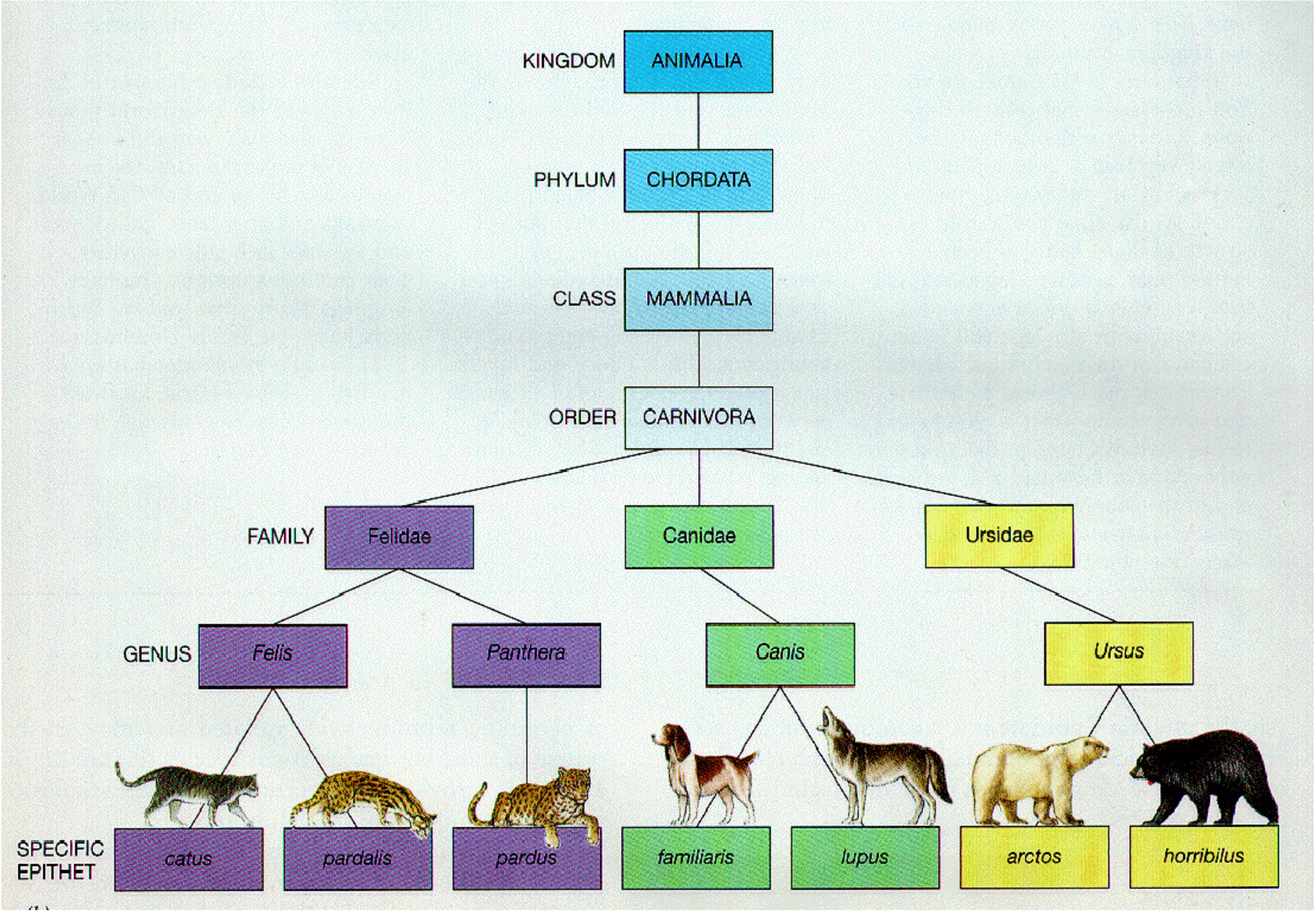
- Useful in visualizing hierarchical models and relationships

Motivation – Level Planarity

- Useful in visualizing hierarchical models and relationships
  - Many natural examples

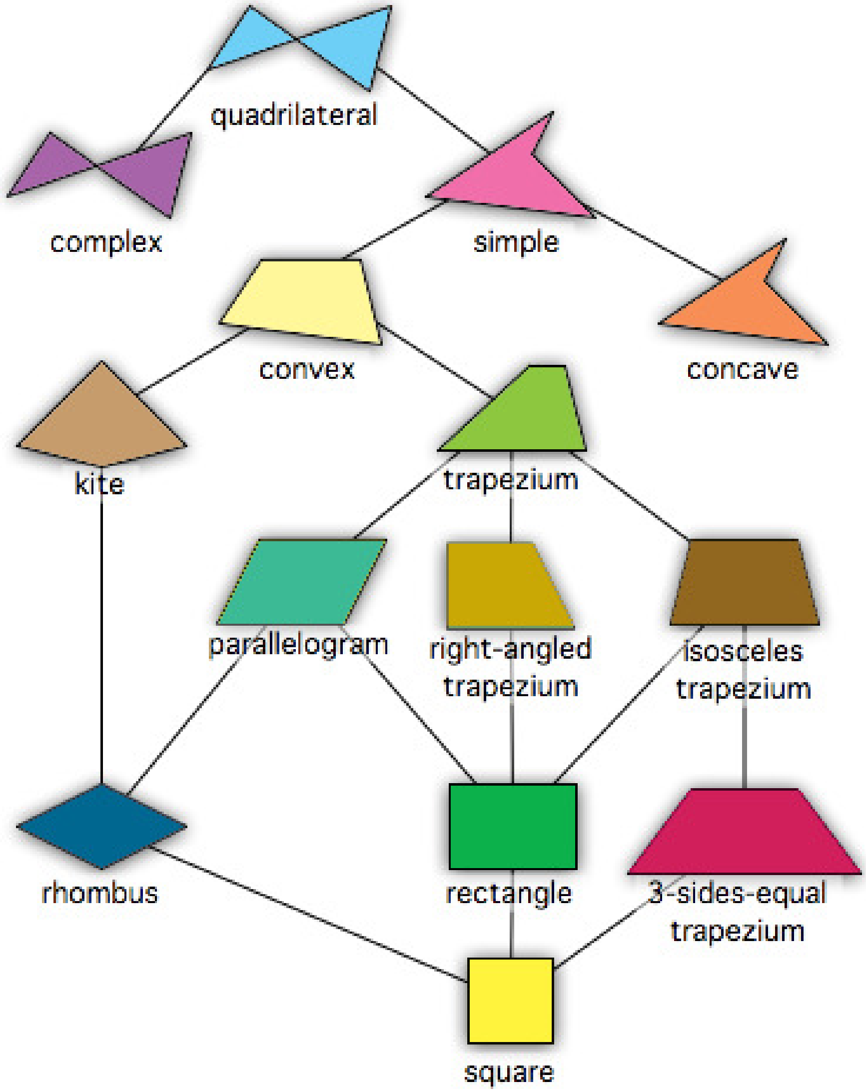
Motivation – Level Planarity

- Useful in visualizing hierarchical models and relationships
  - Many natural examples
    - Classic animal taxonomy



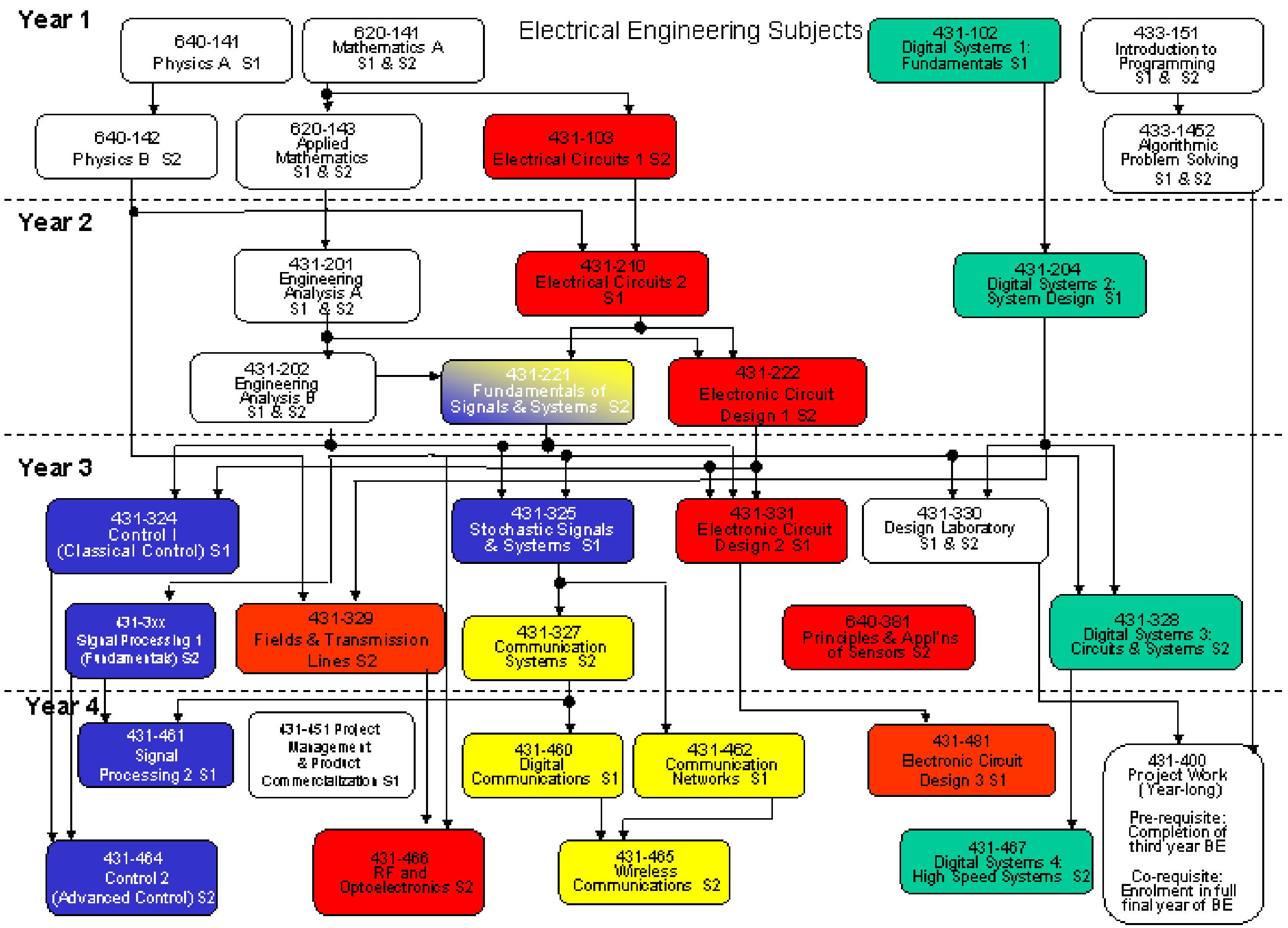
Motivation – Level Planarity

- Useful in visualizing hierarchical models and relationships
  - Many natural examples
    - Polygon hierarchy



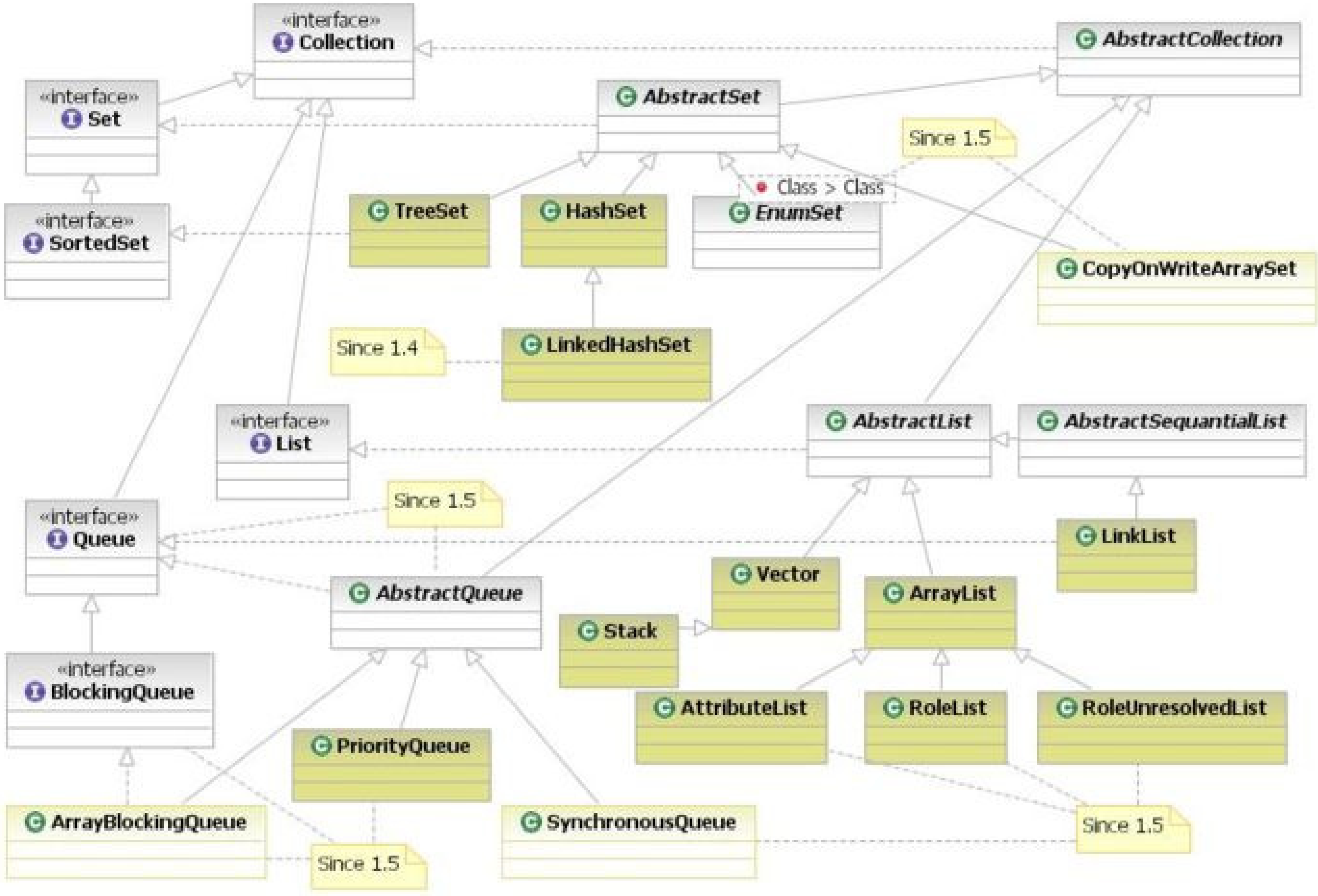
Motivation – Level Planarity

- Useful in visualizing hierarchical models and relationships
  - Many natural examples
    - Flow chart of University Melbourne EE classes



Motivation – Level Planarity

- Useful in visualizing hierarchical models and relationships
  - Many natural examples
    - UML class diagram of the Java Collection



Motivation – Level Planarity

- Useful in visualizing hierarchical models and relationships
  - Many natural examples
  - Any directed acyclic graph (DAG) can be visualized as a *hierarchy*

Motivation – Level Planarity

- Useful in visualizing hierarchical models and relationships
  - Many natural examples
  - Any directed acyclic graph (DAG) can be visualized as a *hierarchy*
- Applications within graph drawing

Motivation – Level Planarity

- Useful in visualizing hierarchical models and relationships
  - Many natural examples
  - Any directed acyclic graph (DAG) can be visualized as a *hierarchy*
- Applications within graph drawing
  - Traditional – Sugiyama's algorithm

Motivation – Level Planarity

- Useful in visualizing hierarchical models and relationships
  - Many natural examples
  - Any directed acyclic graph (DAG) can be visualized as a *hierarchy*
- Applications within graph drawing
  - Traditional – Sugiyama's algorithm
    - Draws DAG's in a top-down manner

Motivation – Level Planarity

- Useful in visualizing hierarchical models and relationships
  - Many natural examples
  - Any directed acyclic graph (DAG) can be visualized as a *hierarchy*
- Applications within graph drawing
  - Traditional – Sugiyama's algorithm
    - $\Rightarrow$ Desire to use *minimum* the number of levels

Motivation – Level Planarity

- Useful in visualizing hierarchical models and relationships
  - Many natural examples
  - Any directed acyclic graph (DAG) can be visualized as a *hierarchy*
- Applications within graph drawing
  - Traditional – Sugiyama's algorithm
    - $\Rightarrow$ Desire to use *minimum* the number of levels
  - Nontraditional – simultaneous embedding

Motivation – Level Planarity

- Useful in visualizing hierarchical models and relationships
  - Many natural examples
  - Any directed acyclic graph (DAG) can be visualized as a *hierarchy*
- Applications within graph drawing
  - Traditional – Sugiyama's algorithm
    - $\Rightarrow$ Desire to use *minimum* the number of levels
  - Nontraditional – simultaneous embedding
    - Number of levels equal number of vertices

Motivation – Level Planarity

- Useful in visualizing hierarchical models and relationships
  - Many natural examples
  - Any directed acyclic graph (DAG) can be visualized as a *hierarchy*
- Applications within graph drawing
  - Traditional – Sugiyama's algorithm
    - $\Rightarrow$ Desire to use *minimum* the number of levels
  - Nontraditional – simultaneous embedding
    - $\Rightarrow$ Uses *maximum* the number of levels

Definitions – Planar Graphs

- Graph $G(V, E)$ is *planar* if and only if

Definitions – Planar Graphs

- Graph $G(V, E)$ is *planar* if and only if
  - G can be drawn in the plane without crossings

Definitions – Planar Graphs

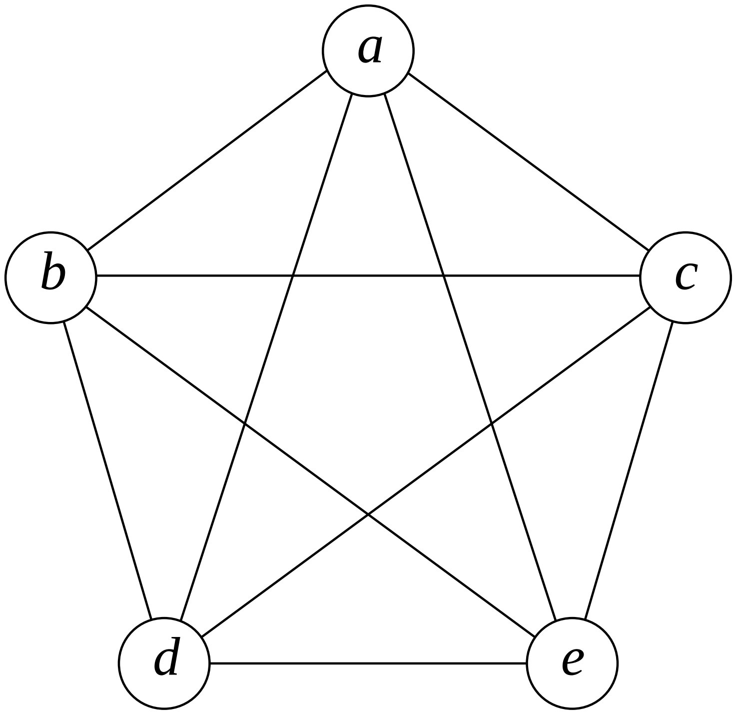
- Graph $G(V, E)$ is *planar* if and only if
  - G can be drawn in the plane without crossings
    - Edges can have bends or only be straight-line edges

Definitions – Planar Graphs

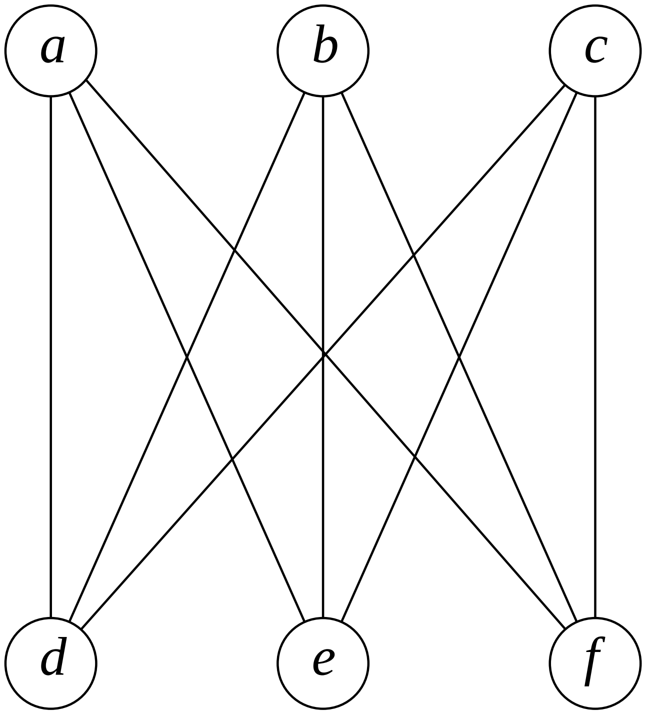
- Graph $G(V, E)$ is *planar* if and only if
  - Equivalently by Kuratowski's Theorem

Definitions – Planar Graphs

- Graph $G(V, E)$ is *planar* if and only if
  - Equivalently by Kuratowski's Theorem
    - G contains no homeomorphic copy of K_5 or $K_{3,3}$



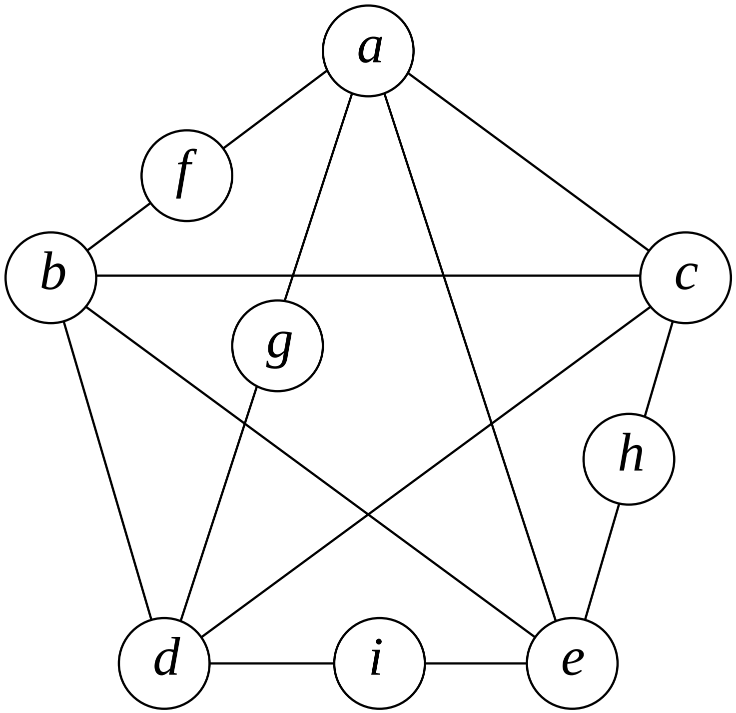
K_5



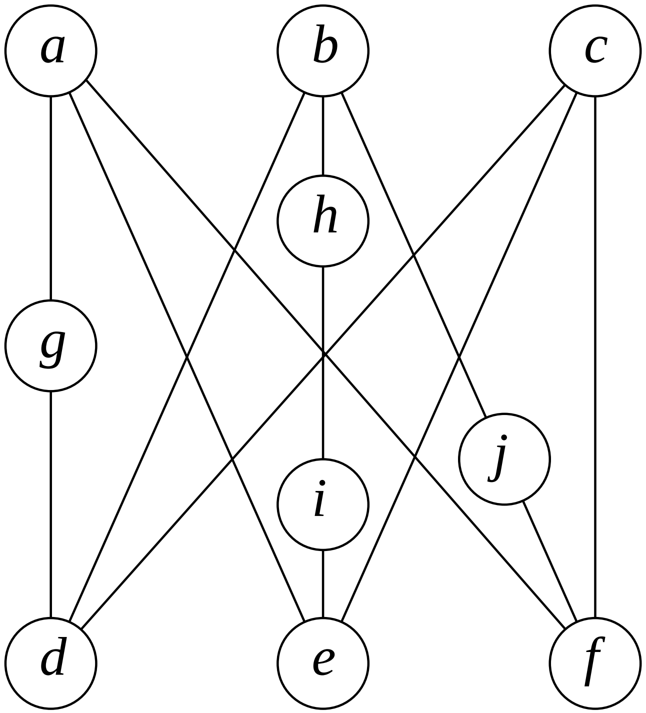
$K_{3,3}$

Definitions – Planar Graphs

- Graph $G(V, E)$ is *planar* if and only if
  - Equivalently by Kuratowski's Theorem
    - G contains no homeomorphic copy of K_5 or $K_{3,3}$



Subdivided K_5

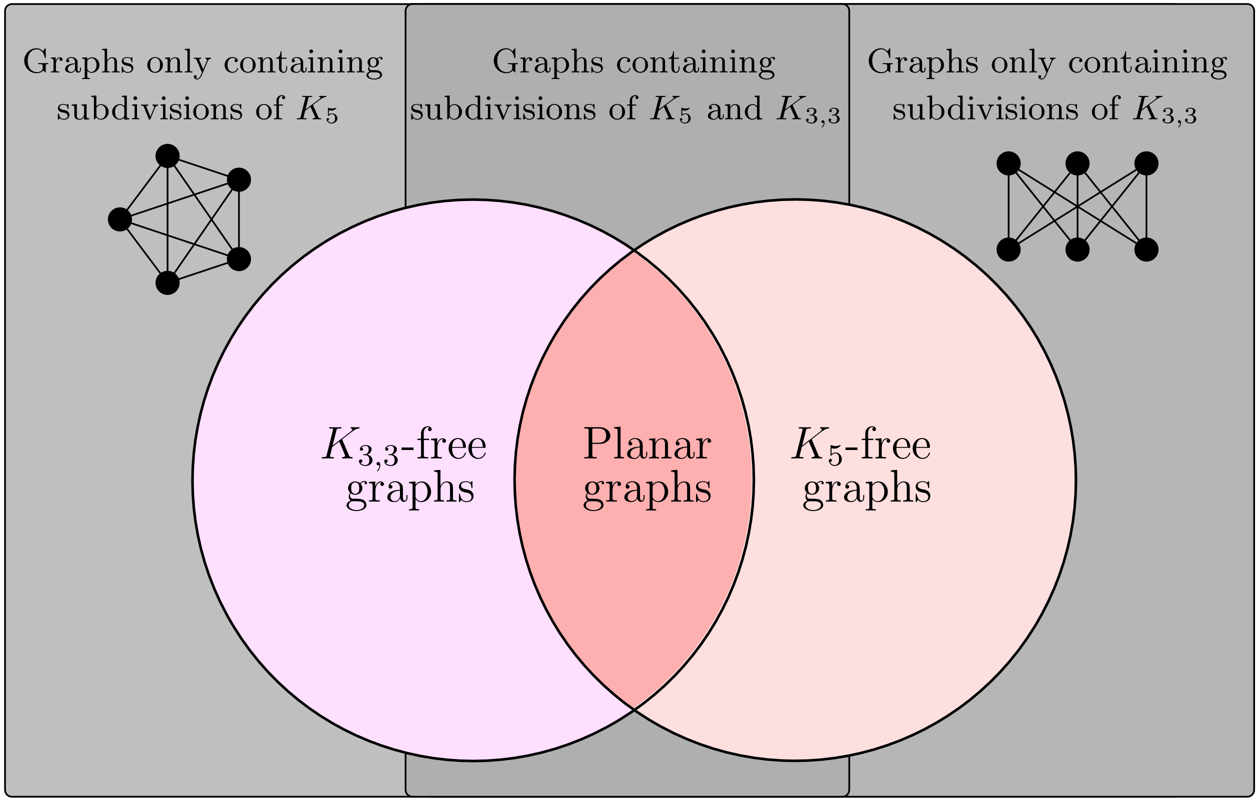


Subdivided $K_{3,3}$

    - I.e., no subgraph in G is a subdivision of K_5 or $K_{3,3}$

Definitions – Planar Graphs

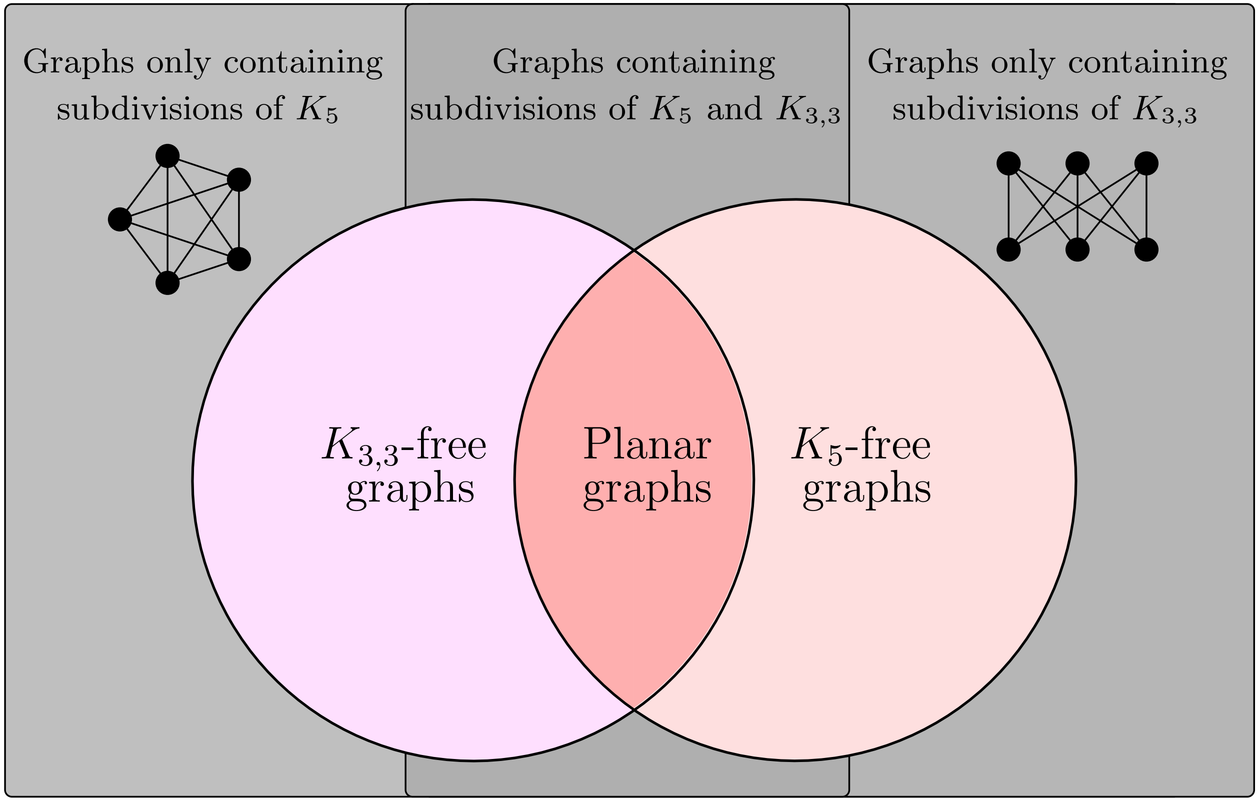
- Graph $G(V, E)$ is *planar* if and only if
  - Equivalently by Kuratowski's Theorem
    - G contains no homeomorphic copy of K_5 or $K_{3,3}$



    - I.e., no subgraph in G is a subdivision of K_5 or $K_{3,3}$

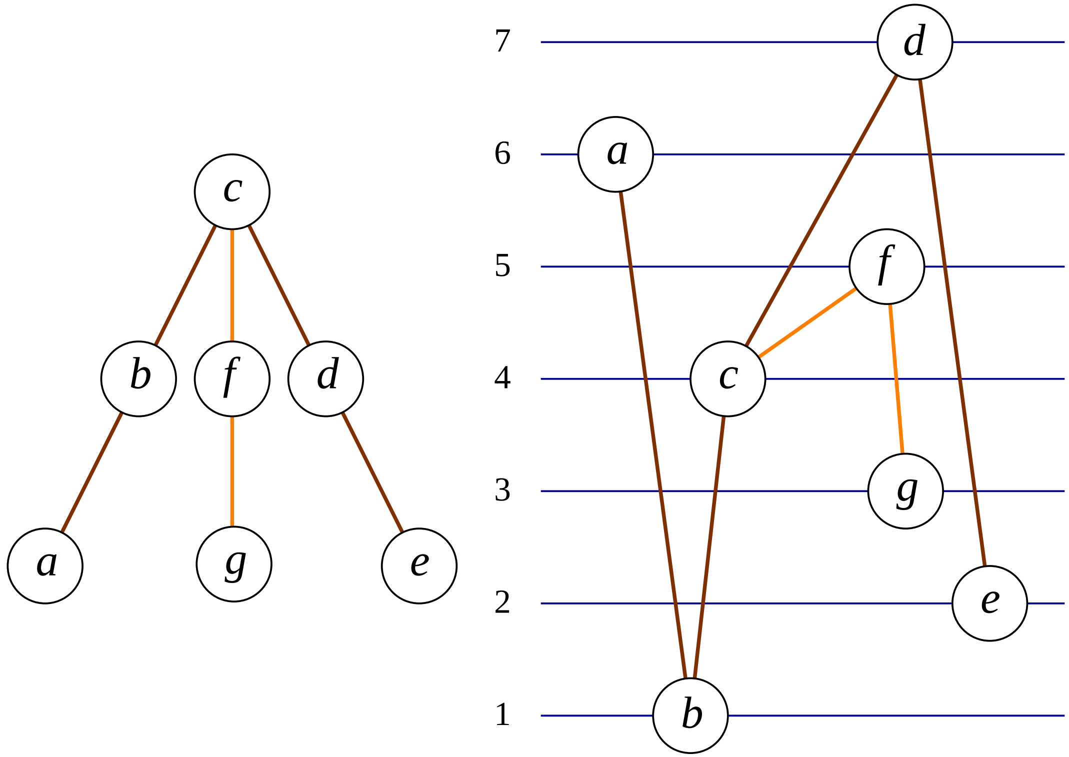
Definitions – Planar Graphs

- Graph $G(V, E)$ is *planar* if and only if
  - Equivalently by Kuratowski's Theorem
    - G contains no homeomorphic copy of K_5 or $K_{3,3}$



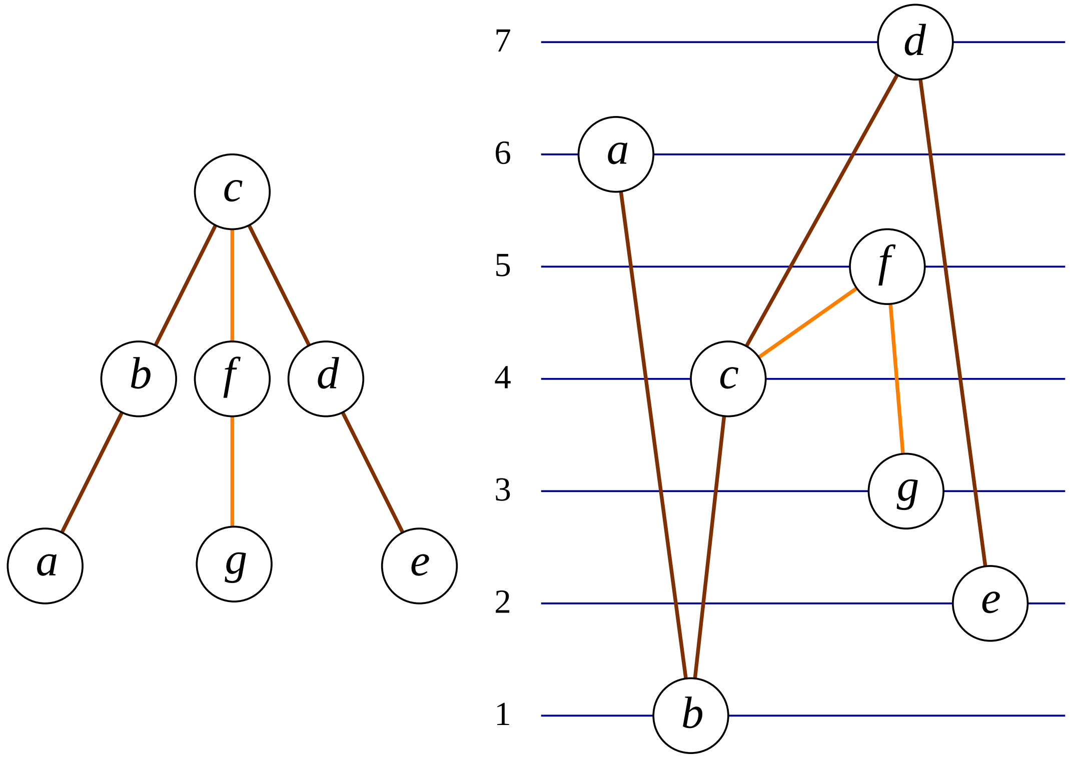
- Similar forbidden subdivision characterization for ULP graphs

Definitions – Level Planar Graphs



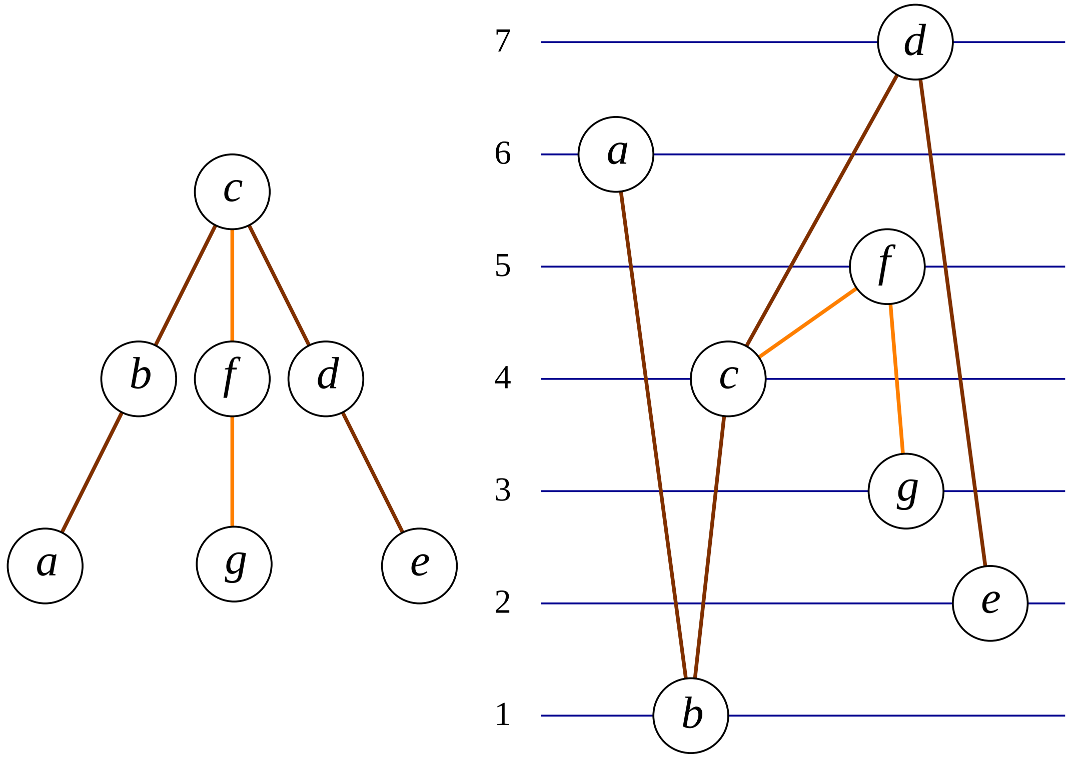
- An n-*level graph* $G(V, E, \phi)$

Definitions – Level Planar Graphs



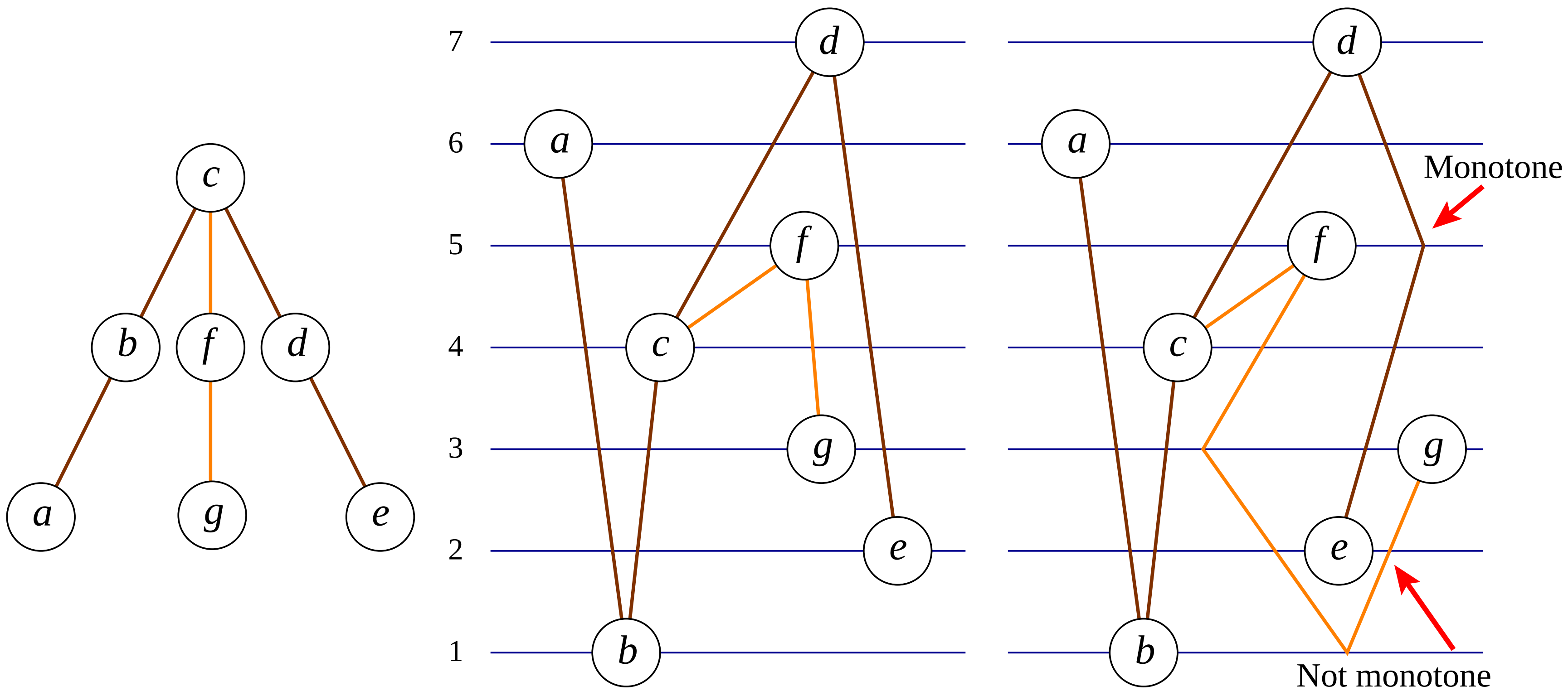
- An n-*level graph* $G(V, E, \phi)$
  - Has n vertices with a bijective *leveling* $\phi : V \rightarrow [1..n]$

Definitions – Level Planar Graphs



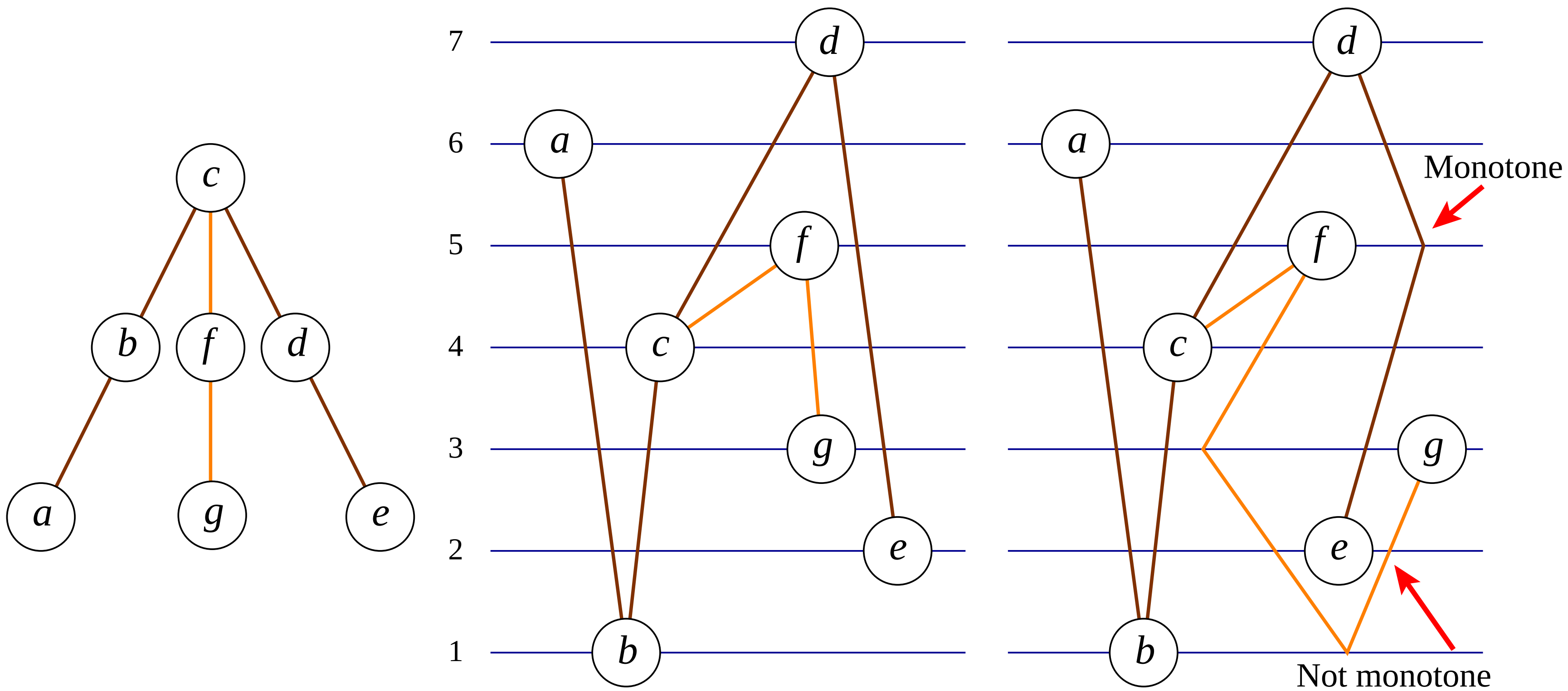
- An n-*level graph* $G(V, E, \phi)$
  - Has n vertices with a bijective *leveling* $\phi : V \rightarrow [1..n]$
    - Assigns exactly one vertex per level

Definitions – Level Planar Graphs



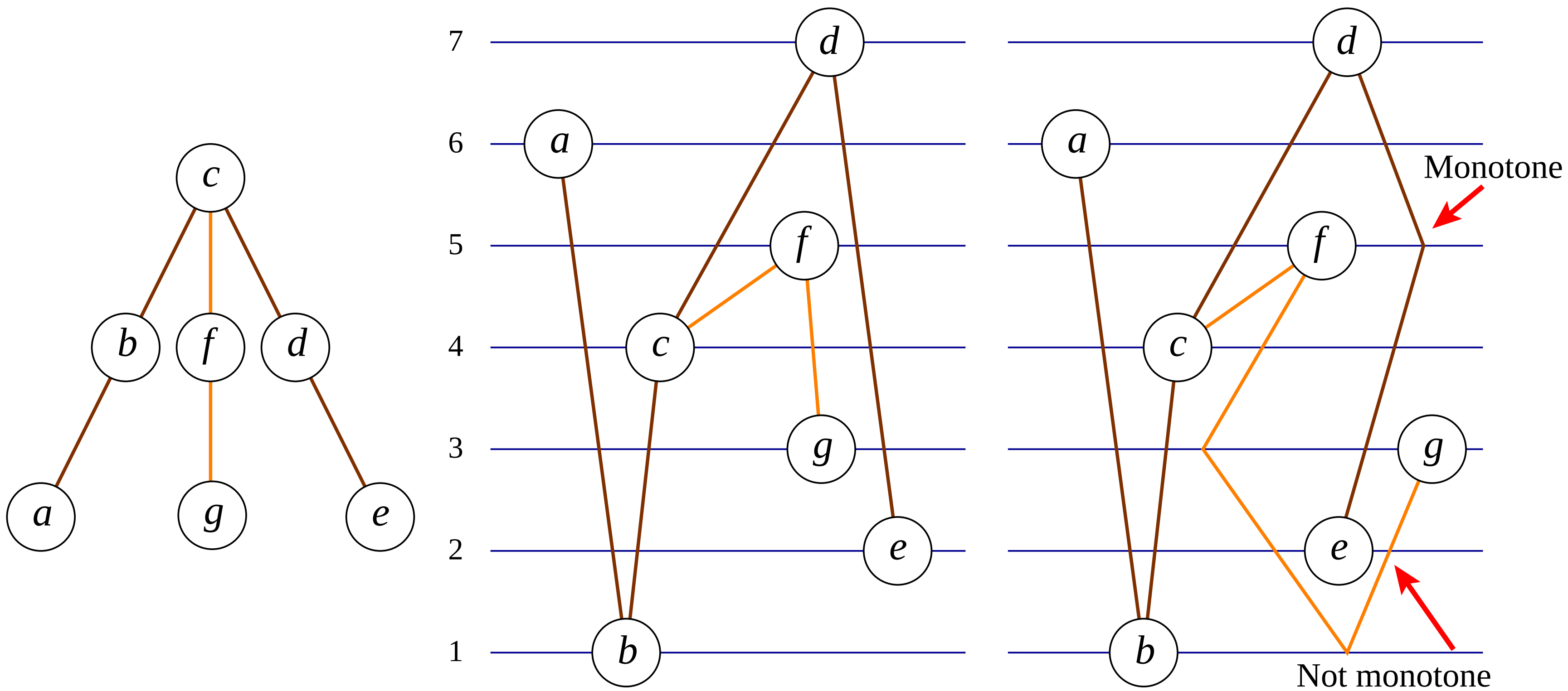
- An n-*level graph* $G(V, E, \phi)$
  - Has n vertices with a bijective *leveling* $\phi : V \rightarrow [1..n]$
  - Edges are y-monotone

Definitions – Level Planar Graphs



- An n-*level graph* $G(V, E, \phi)$
  - Has n vertices with a bijective *leveling* $\phi : V \rightarrow [1..n]$
  - Edges are y-monotone
- G is *level planar if*

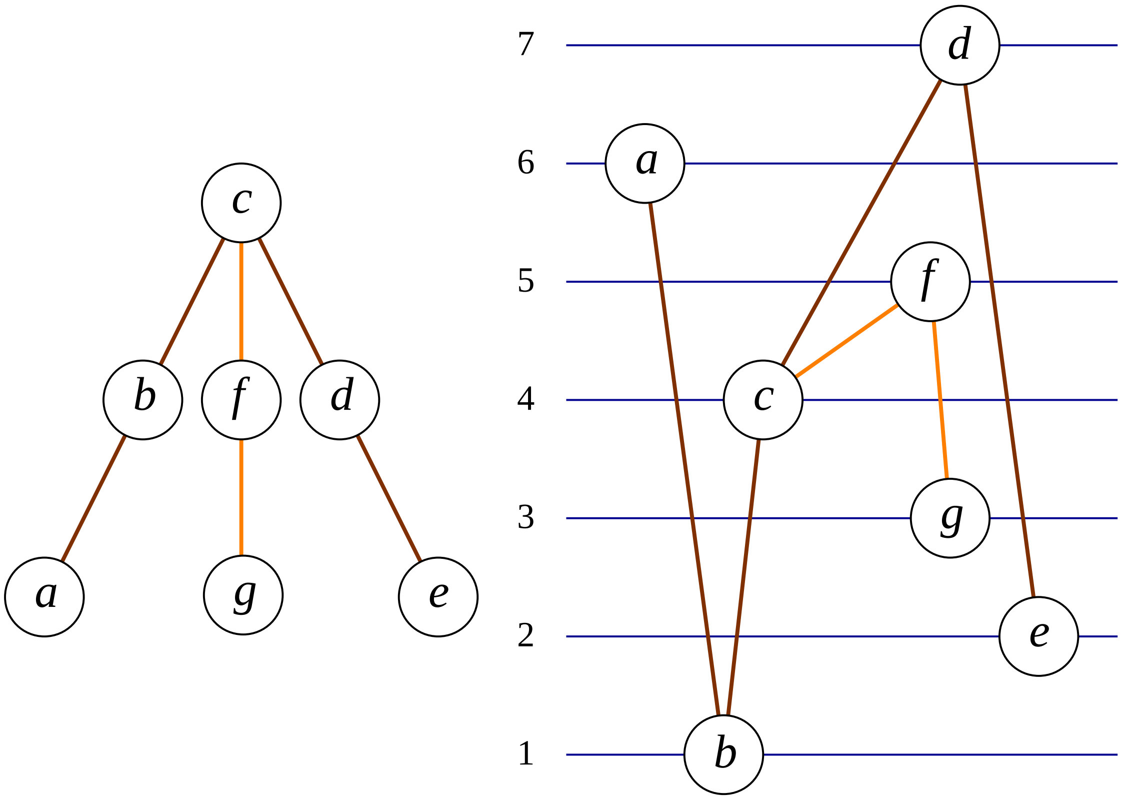
Definitions – Level Planar Graphs



- An n-*level graph* $G(V, E, \phi)$
  - Has n vertices with a bijective *leveling* $\phi : V \rightarrow [1..n]$
  - Edges are y-monotone
- G is *level planar if*
  - G can be drawn without crossings AND each vertex remains on its level

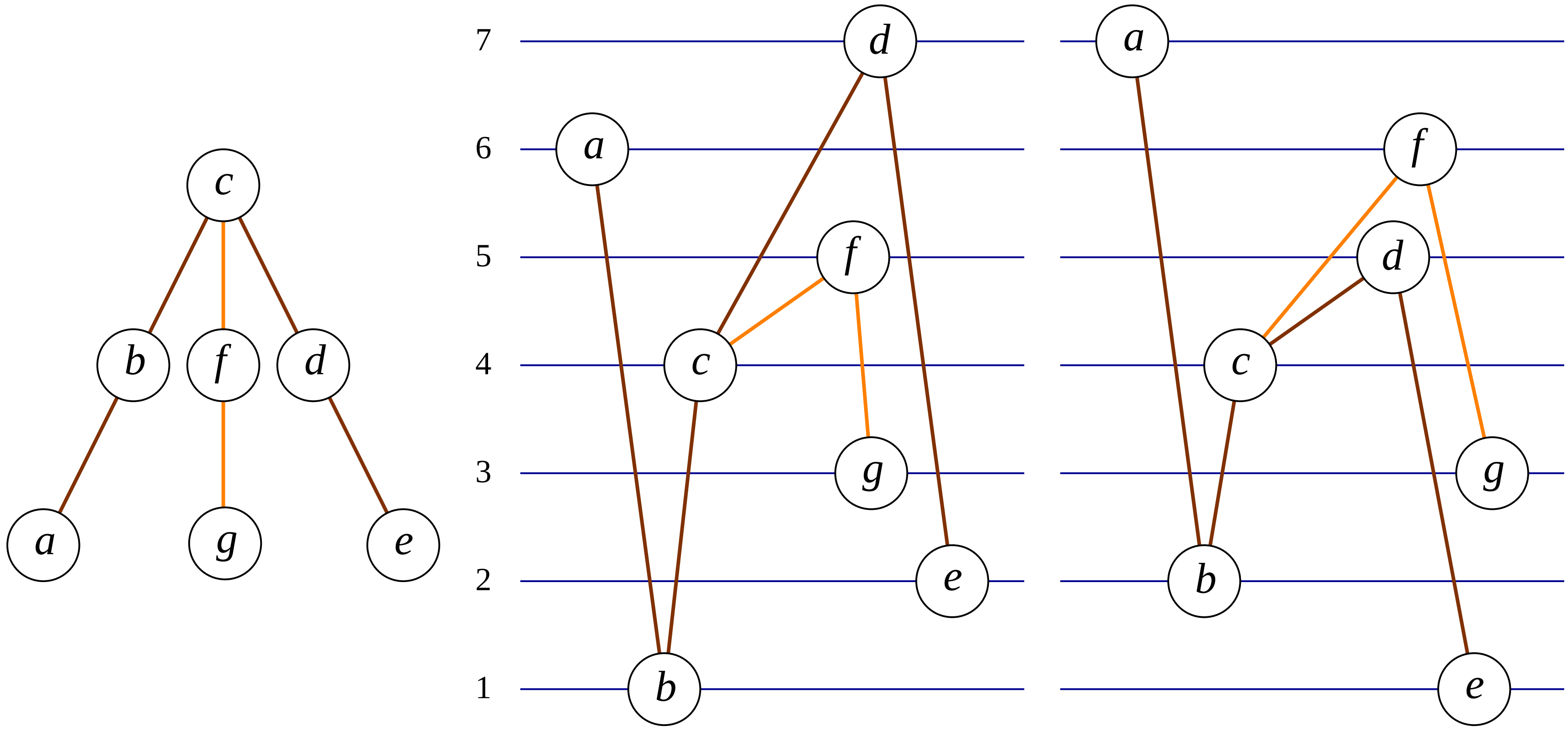
Definitions – Unlabeled Level Planarity

- Only *some* planar graphs are level planar over *every* bijective leveling



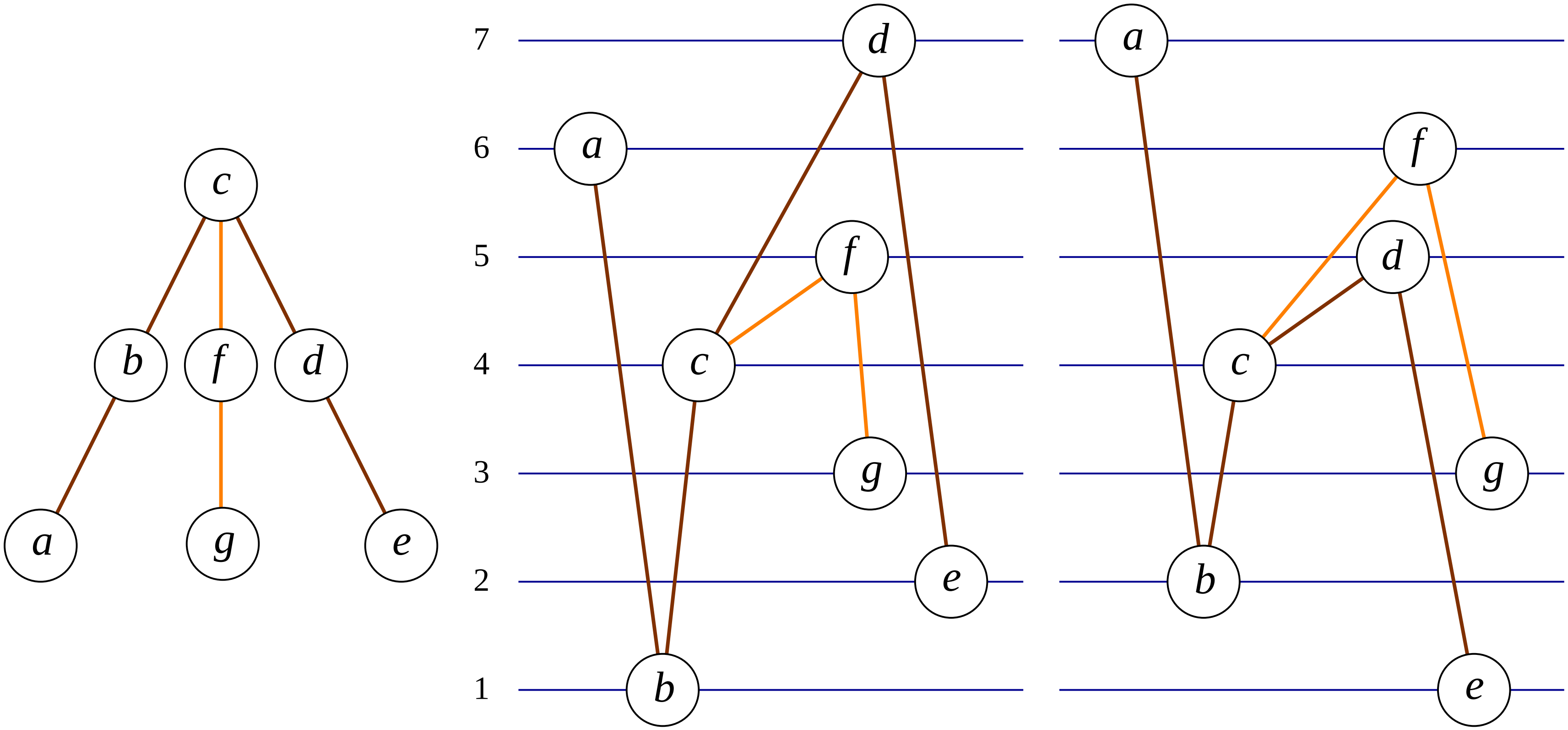
Definitions – Unlabeled Level Planarity

- Only *some* planar graphs are level planar over *every* bijective leveling



Definitions – Unlabeled Level Planarity

- Only *some* planar graphs are level planar over *every* bijective leveling
  - Such graphs are called Unlabeled Level Planar (ULP)

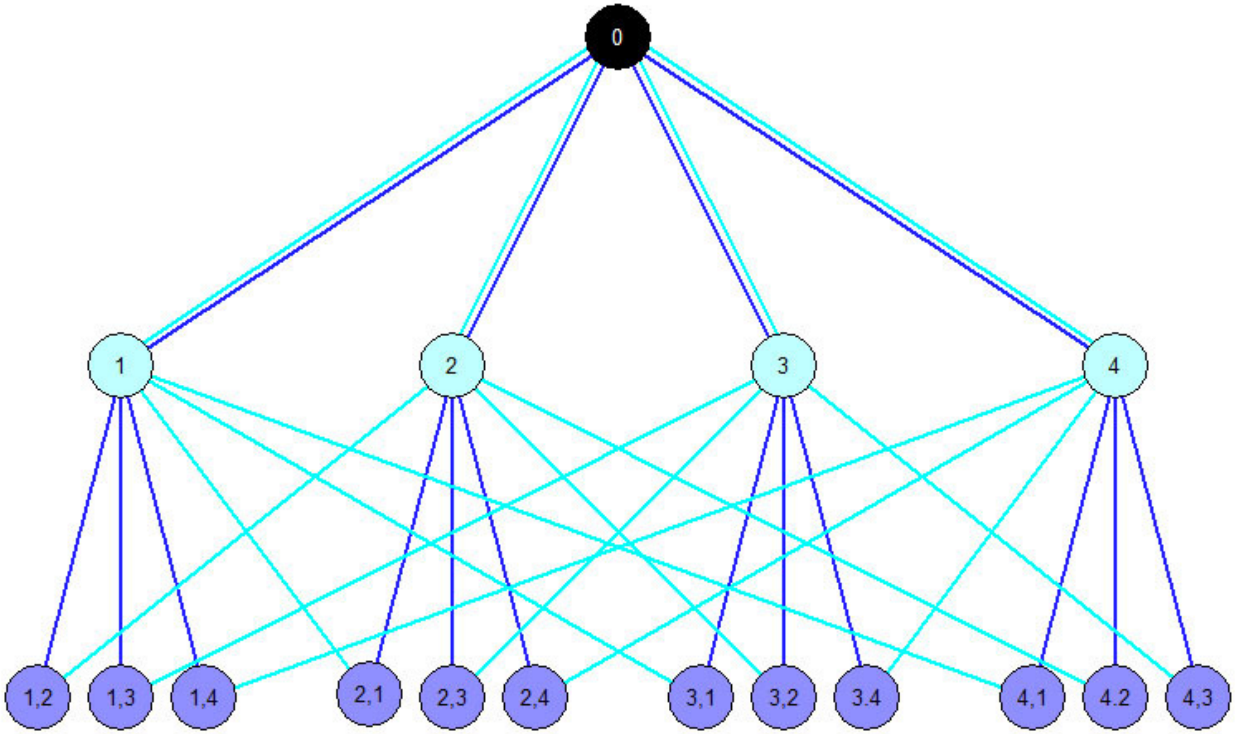


Definitions – Simultaneous Embedding

- Embedding multiple planar graphs on the same vertex set V

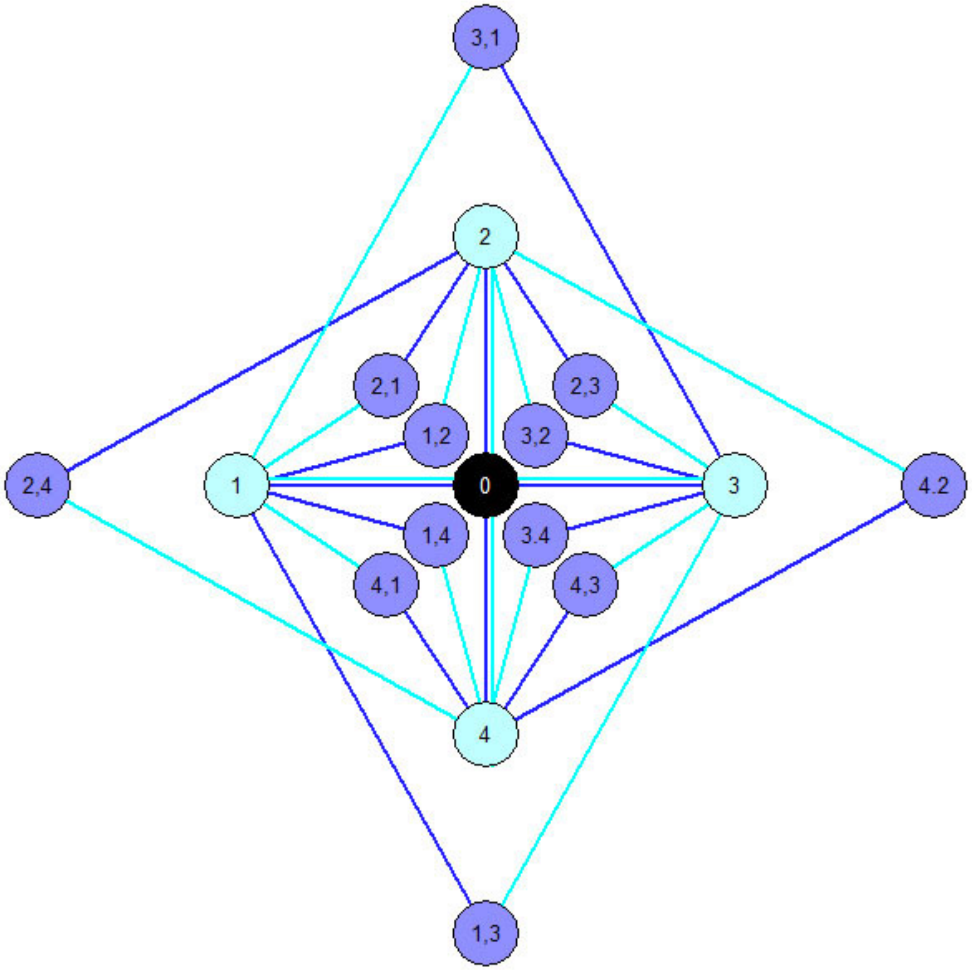
Definitions – Simultaneous Embedding

- Embedding multiple planar graphs on the same vertex set V
  - Generalizes the notion of planarity



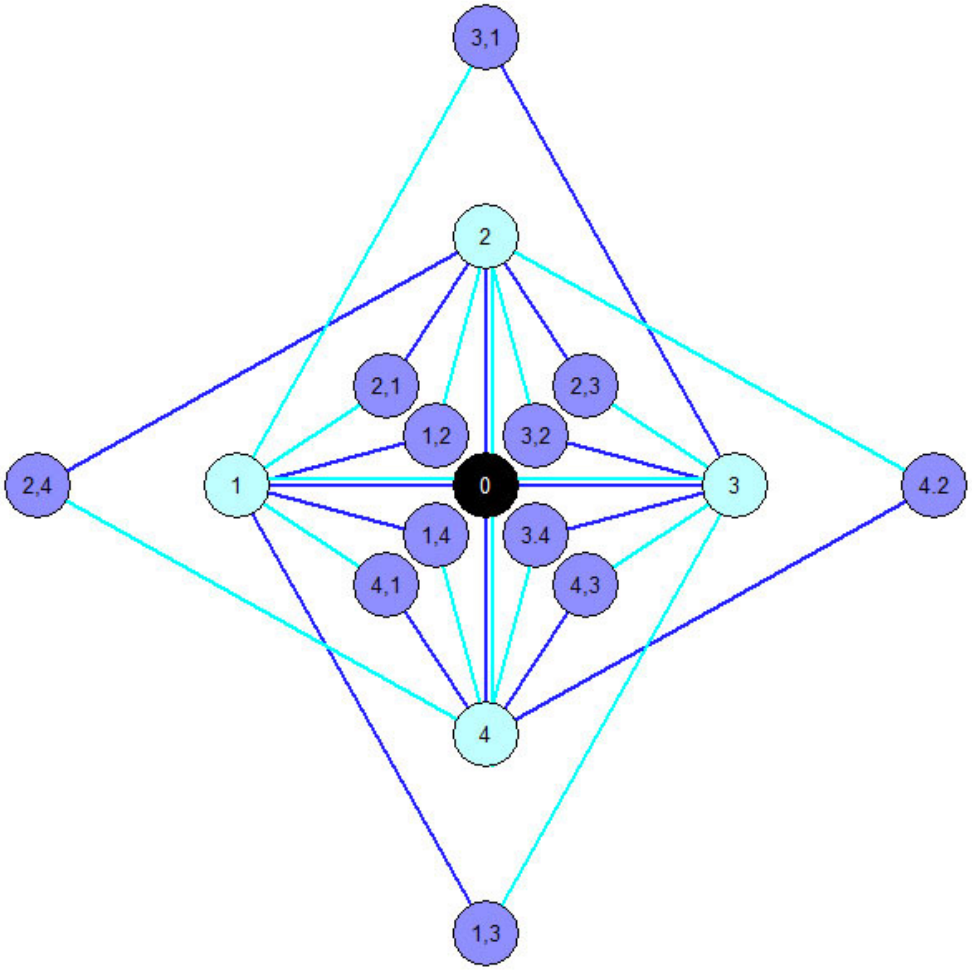
Definitions – Simultaneous Embedding

- Embedding multiple planar graphs on the same vertex set V
  - Related to geometric thickness



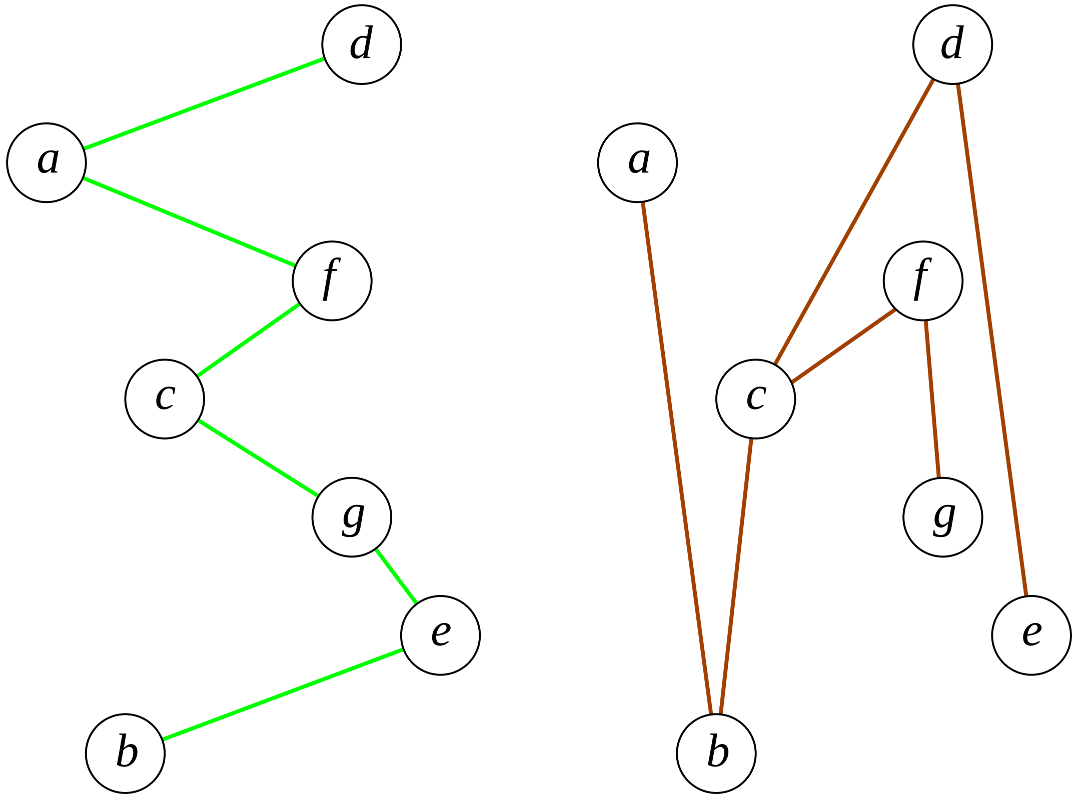
Definitions – Simultaneous Embedding

- Embedding multiple planar graphs on the same vertex set V
  - Desire straight-line edges and each layer is planar



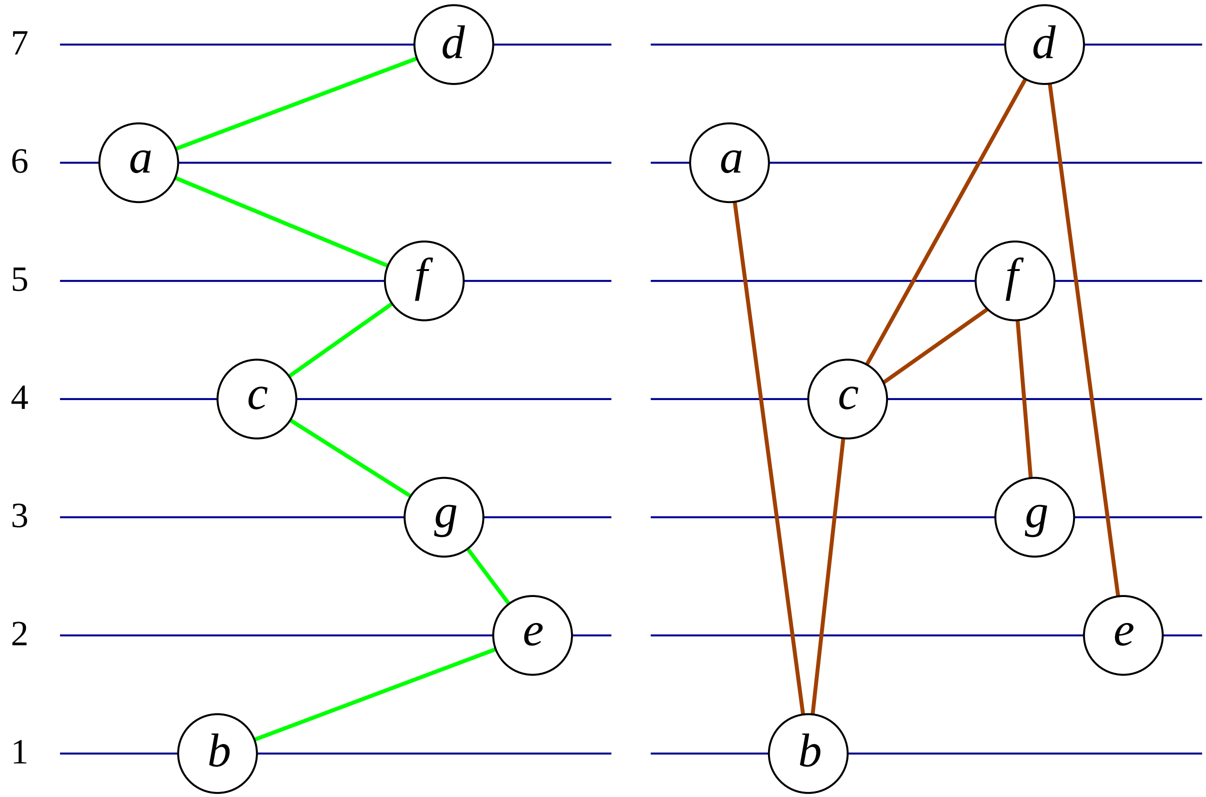
Definitions – Simultaneous Embedding

- Embedding multiple planar graphs on the same vertex set V
  - Desire straight-line edges and each layer is planar
- Simultaneously embed monotone path with any ULP graph
  - Mapping between vertices is given by labeling



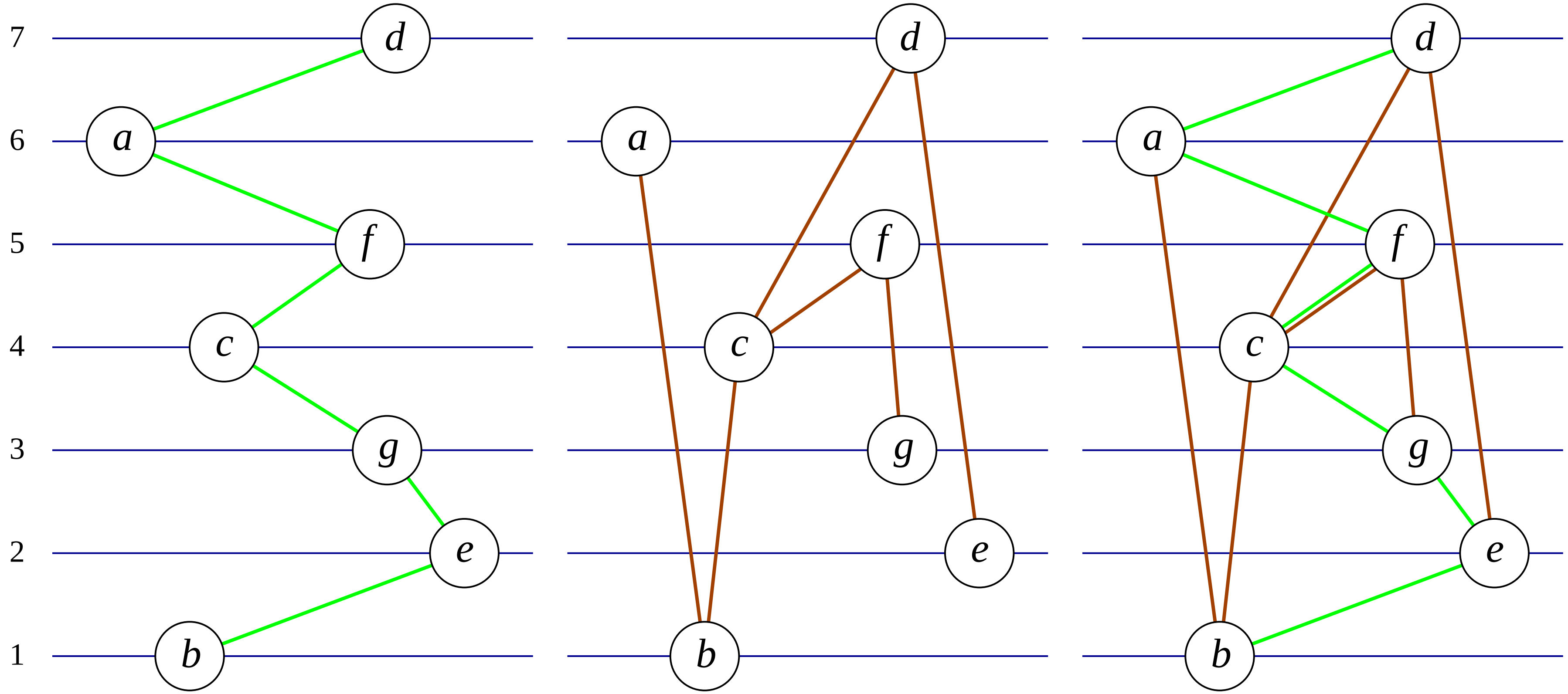
Definitions – Simultaneous Embedding

- Embedding multiple planar graphs on the same vertex set V
  - Desire straight-line edges and each layer is planar
- Simultaneously embed monotone path with any ULP graph
  - Ordering of the vertices in path gives a leveling



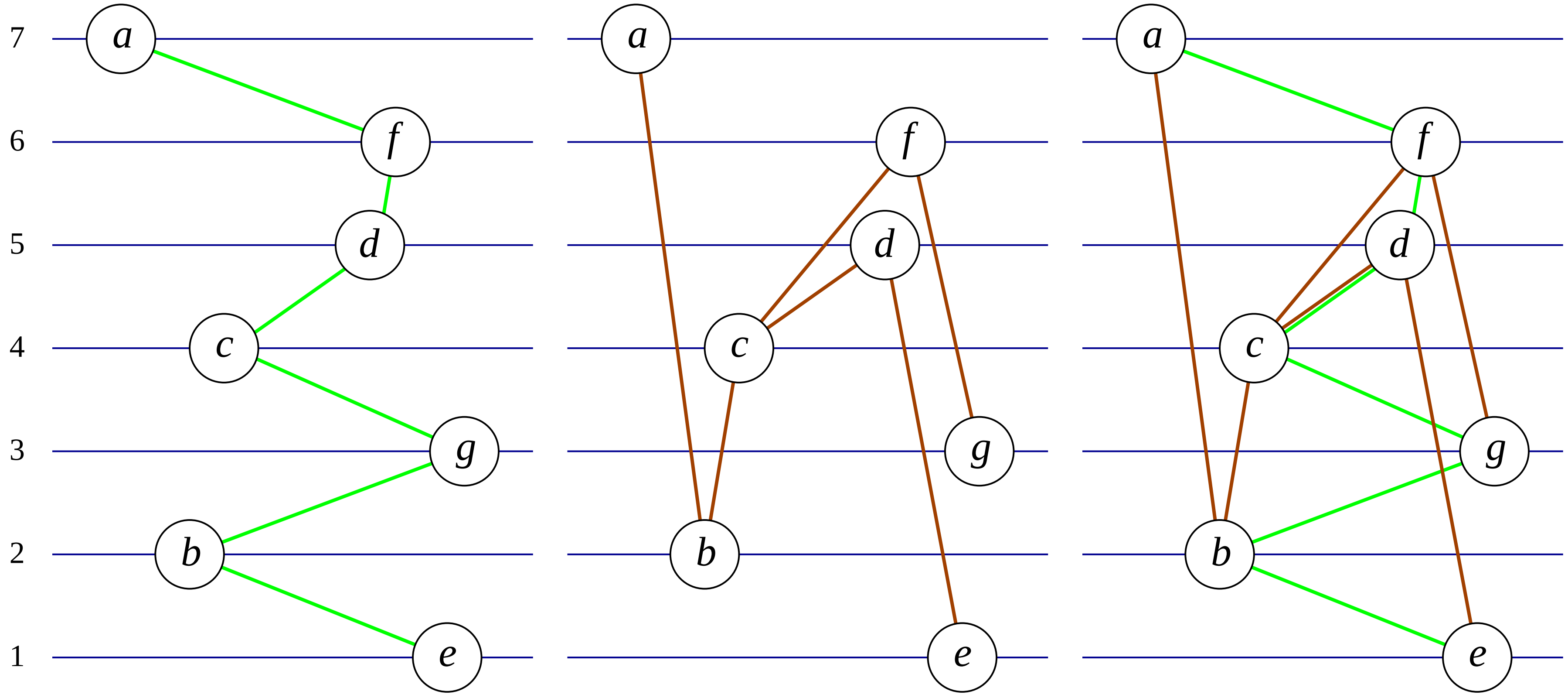
Definitions – Simultaneous Embedding

- Embedding multiple planar graphs on the same vertex set V
  - Desire straight-line edges and each layer is planar
- Simultaneously embed monotone path with any ULP graph
  - Graphs overlaid so that vertices with same label have same position



Definitions – Simultaneous Embedding

- Embedding multiple planar graphs on the same vertex set V
  - Desire straight-line edges and each layer is planar
- Simultaneously embed monotone path with any ULP graph
  - Has to work for *any* of the $n!$ labelings between graphs



Previous Work – Level Planarity

- $O(n)$ time algorithms for level graphs

Previous Work – Level Planarity

- $O(n)$ time algorithms for level graphs
  - Jünger, Leipert, and Mutzel gave a level planarity testing algorithm in 1998
    - Uses PQ-trees

Previous Work – Level Planarity

- $O(n)$ time algorithms for level graphs
  - Jünger, Leipert, and Mutzel gave a level planarity testing algorithm in 1998
  - Jünger and Leipert achieved level planar embedding in 1999
    - A *level embedding* is the left-to-right ordering of vertices along a level

Previous Work – Level Planarity

- $O(n)$ time algorithms for level graphs
  - Jünger, Leipert, and Mutzel gave a level planarity testing algorithm in 1998
  - Jünger and Leipert achieved level planar embedding in 1999
  - Eades, Feng, Lin, and Nagamochi devised a straight-line level planar drawing algorithm given an embedding in 1997
    - Shows any level planar graph drawn with bends can be drawn with straight-line edges instead

Previous Work – Level Planarity

- $O(n)$ time algorithms for level graphs
  - Jünger, Leipert, and Mutzel gave a level planarity testing algorithm in 1998
  - Jünger and Leipert achieved level planar embedding in 1999
  - Eades, Feng, Lin, and Nagamochi devised a straight-line level planar drawing algorithm given an embedding in 1997
- Characterizations of level graphs

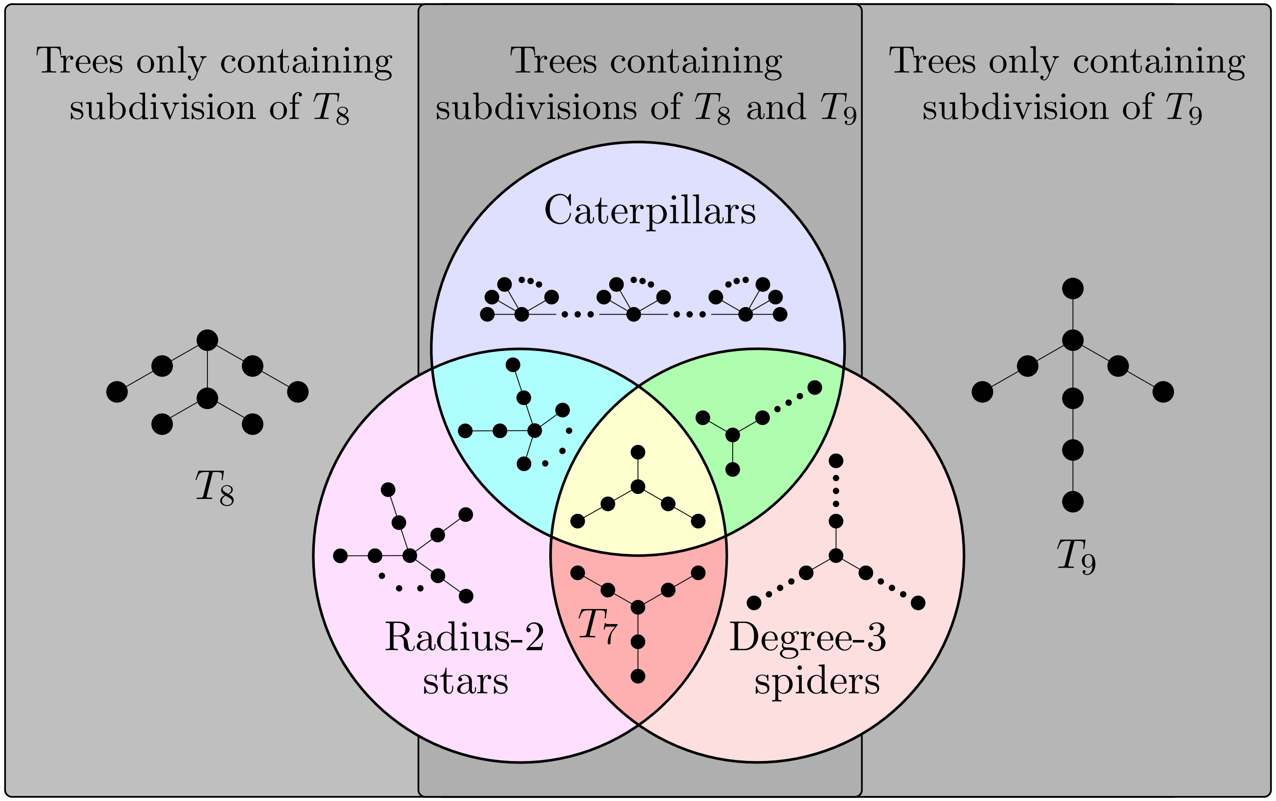
Previous Work – Level Planarity

- $O(n)$ time algorithms for level graphs
  - Jünger, Leipert, and Mutzel gave a level planarity testing algorithm in 1998
  - Jünger and Leipert achieved level planar embedding in 1999
  - Eades, Feng, Lin, and Nagamochi devised a straight-line level planar drawing algorithm given an embedding in 1997
- Characterizations of level graphs
  - Di Battista and Nardelli characterized hierarchies in 1988
    - Uses level non-planar (LNP) patterns

Previous Work – Level Planarity

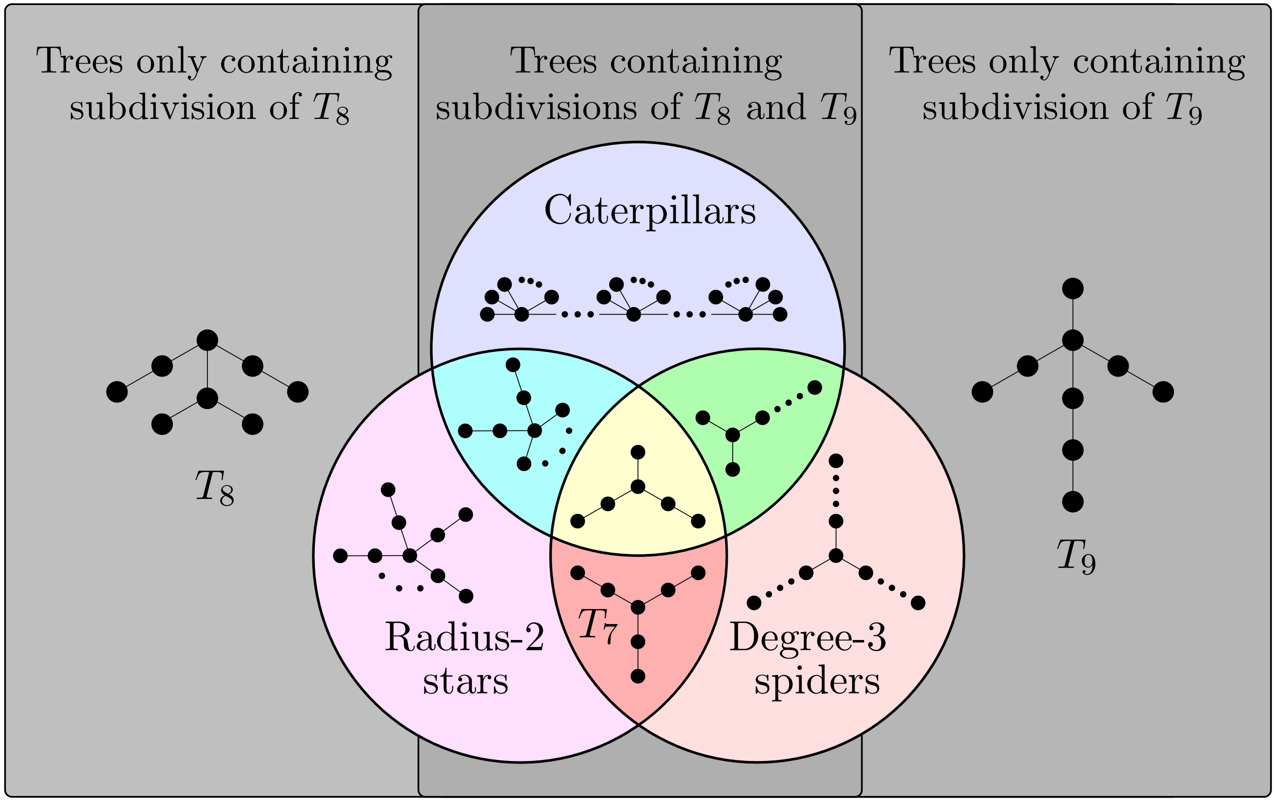
- $O(n)$ time algorithms for level graphs
  - Jünger, Leipert, and Mutzel gave a level planarity testing algorithm in 1998
  - Jünger and Leipert achieved level planar embedding in 1999
  - Eades, Feng, Lin, and Nagamochi devised a straight-line level planar drawing algorithm given an embedding in 1997
- Characterizations of level graphs
  - Di Battista and Nardelli characterized hierarchies in 1988
  - Healy, Kuusik, and Leipert found minimal LNP subgraph patterns in 2000
    - Incomplete – do not match all forbidden ULP graphs

Previous Work – ULP Trees

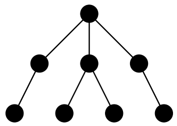


- Characterization of ULP trees by two forbidden subdivisions

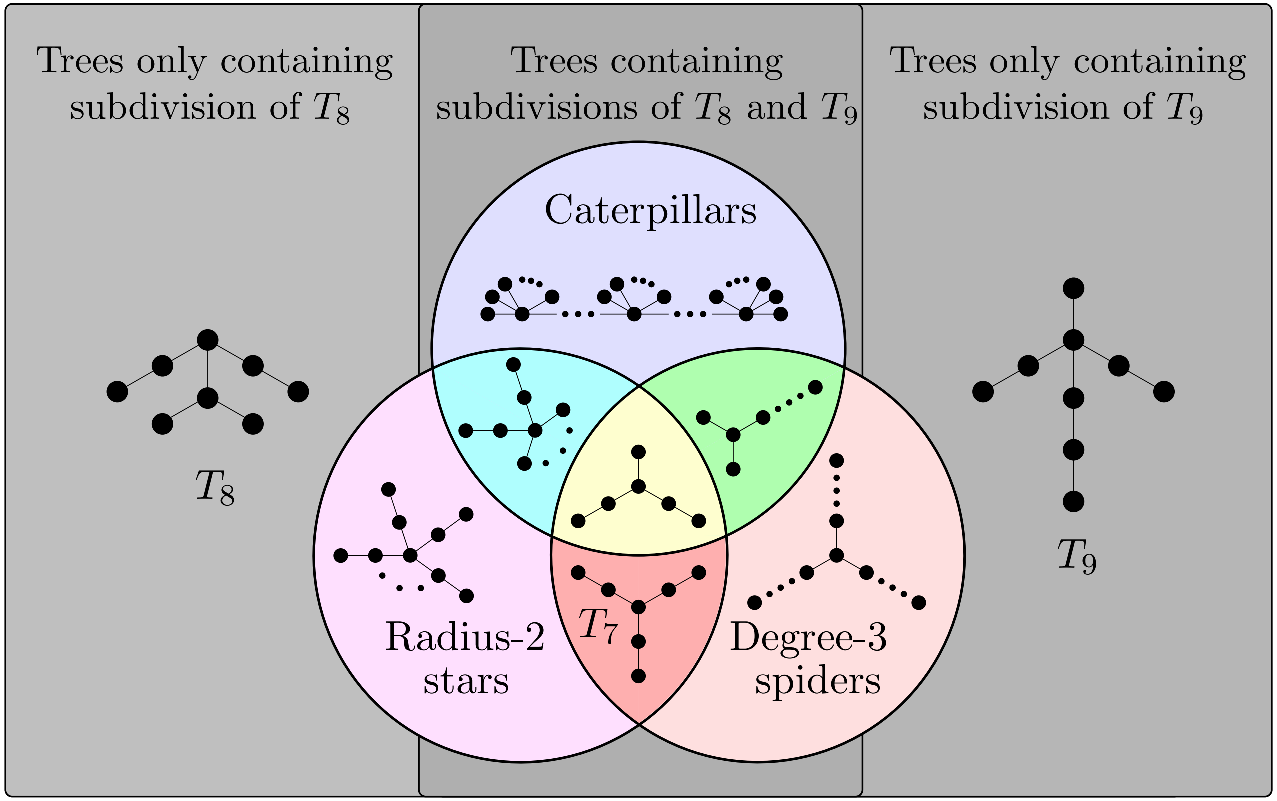
Previous Work – ULP Trees



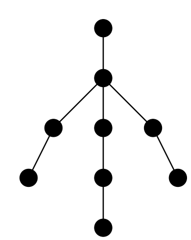
- Characterization of ULP trees by two forbidden subdivisions
  - Tree T_8 with 8 vertices and two nodes of degree 3



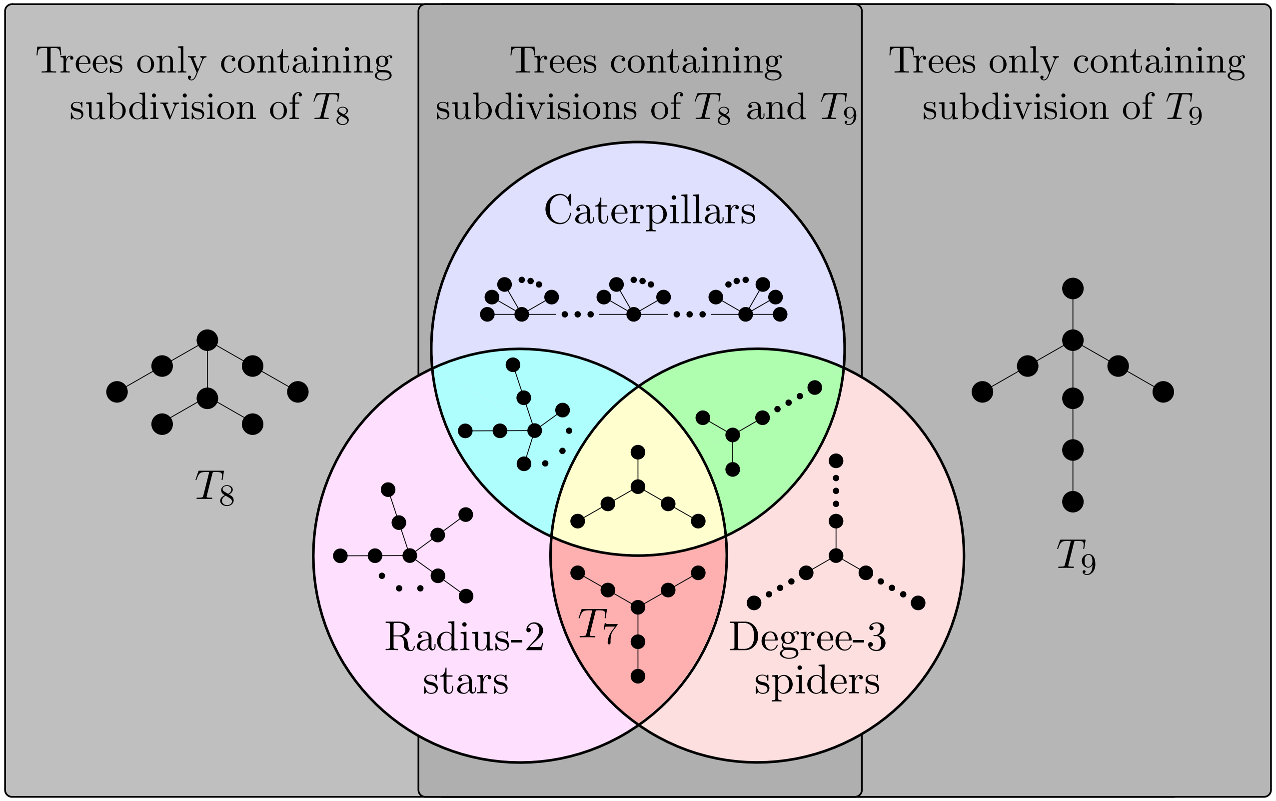
Previous Work – ULP Trees



- Characterization of ULP trees by two forbidden subdivisions
  - Tree T_8 with 8 vertices and two nodes of degree 3
  - Tree T_9 with 9 vertices and one node of degree 4

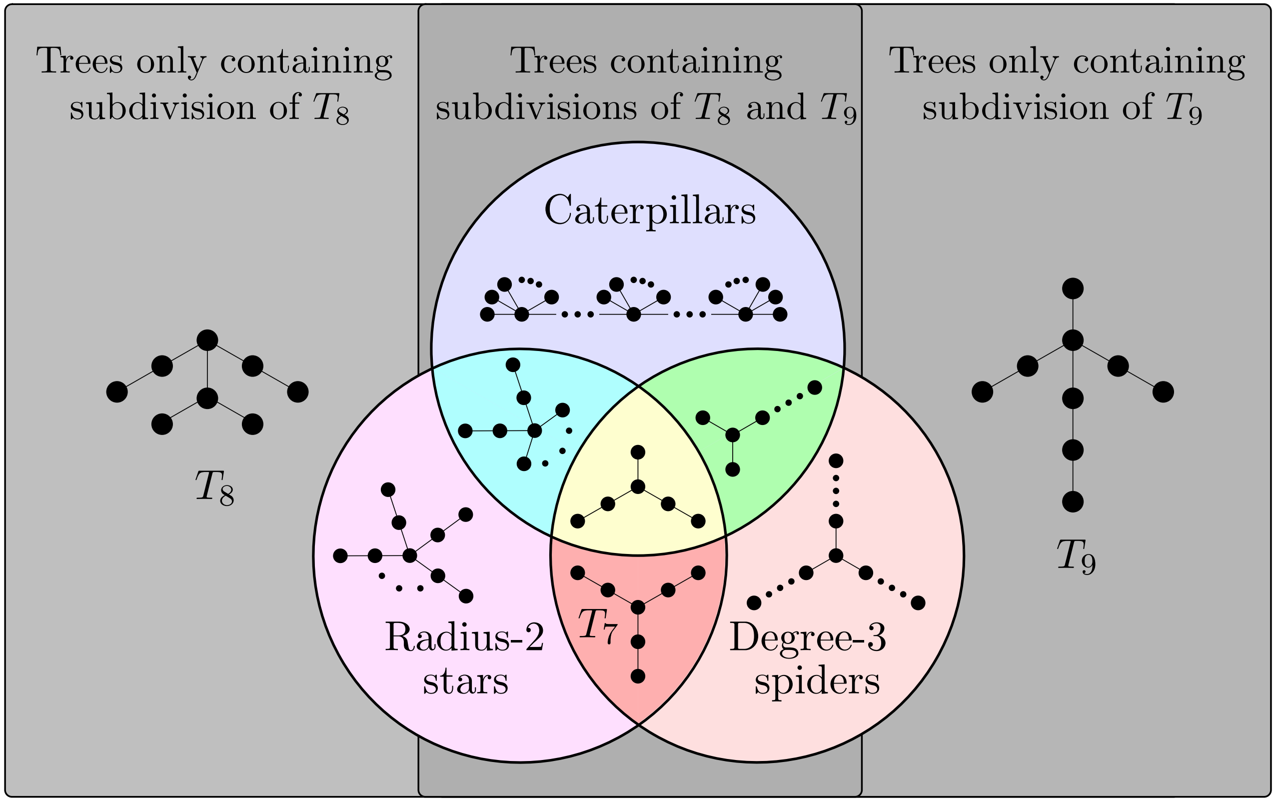


Previous Work – ULP Trees

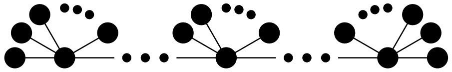


- All ULP trees fall into one of three categories:

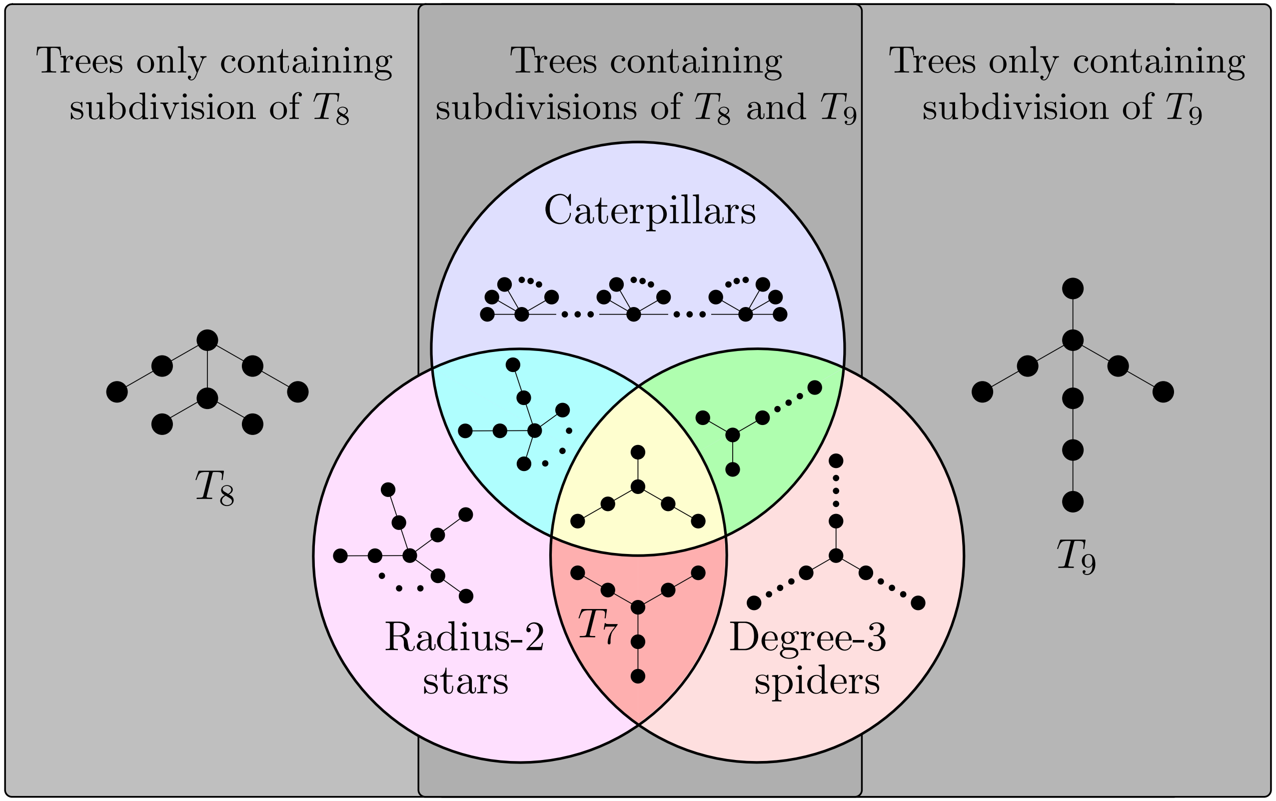
Previous Work – ULP Trees



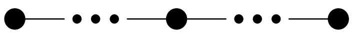
- All ULP trees fall into one of three categories:
  - **Caterpillars**

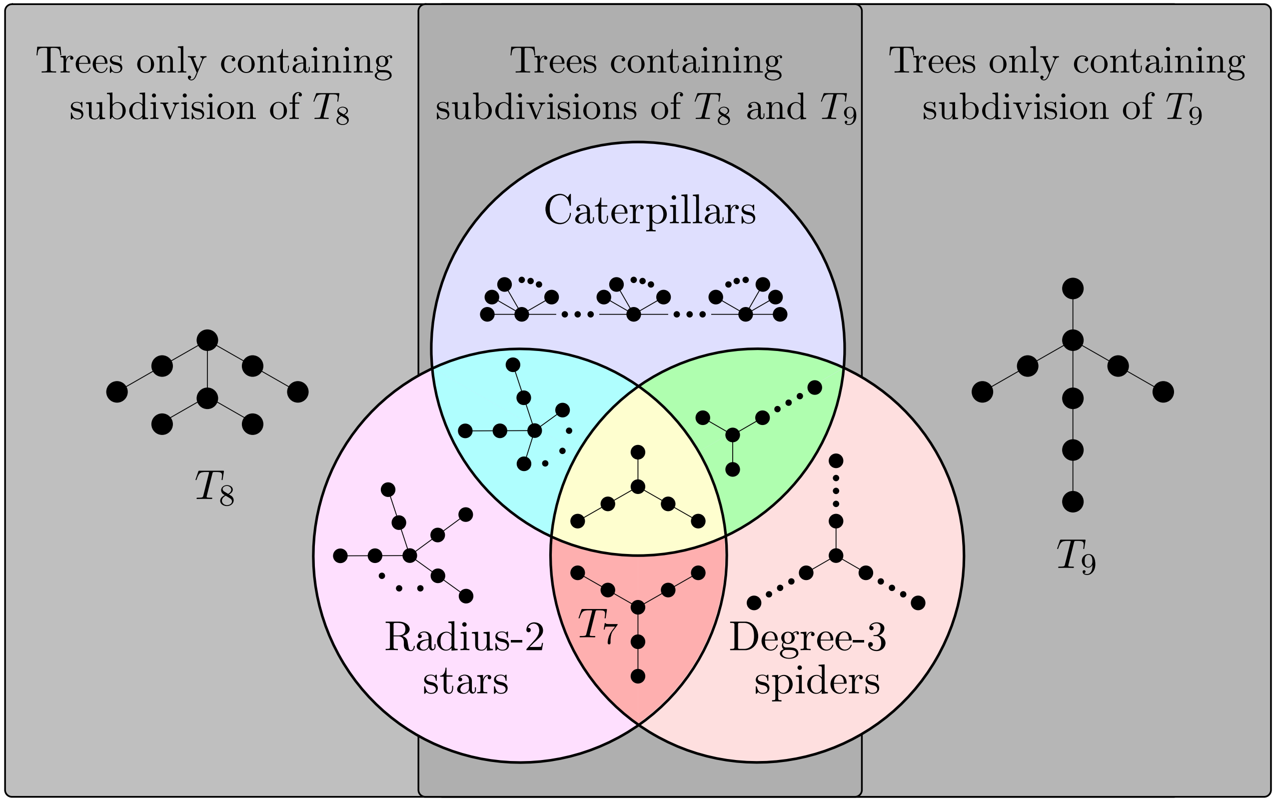


Previous Work – ULP Trees

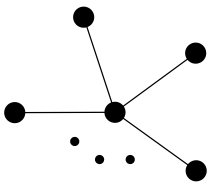


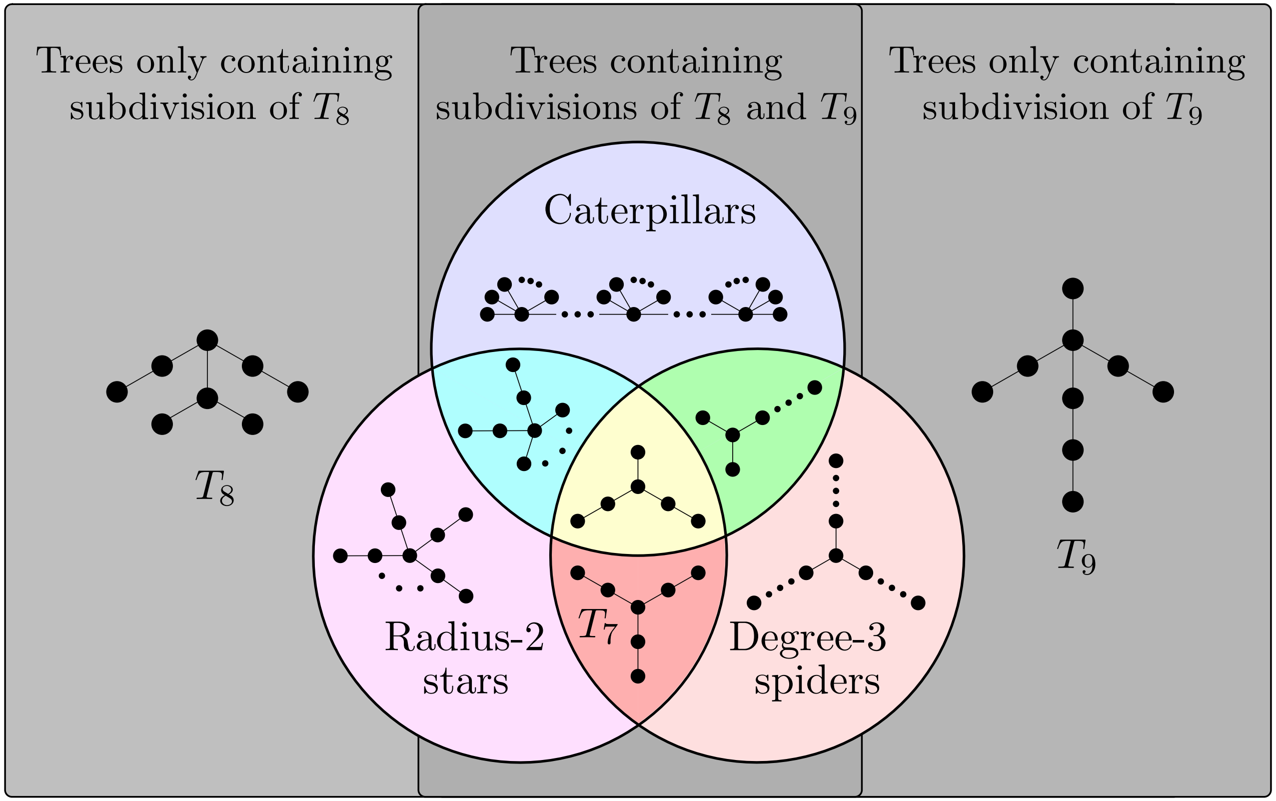
- All ULP trees fall into one of three categories:
  - **Caterpillars**



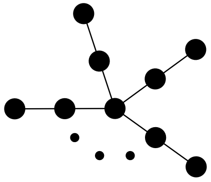


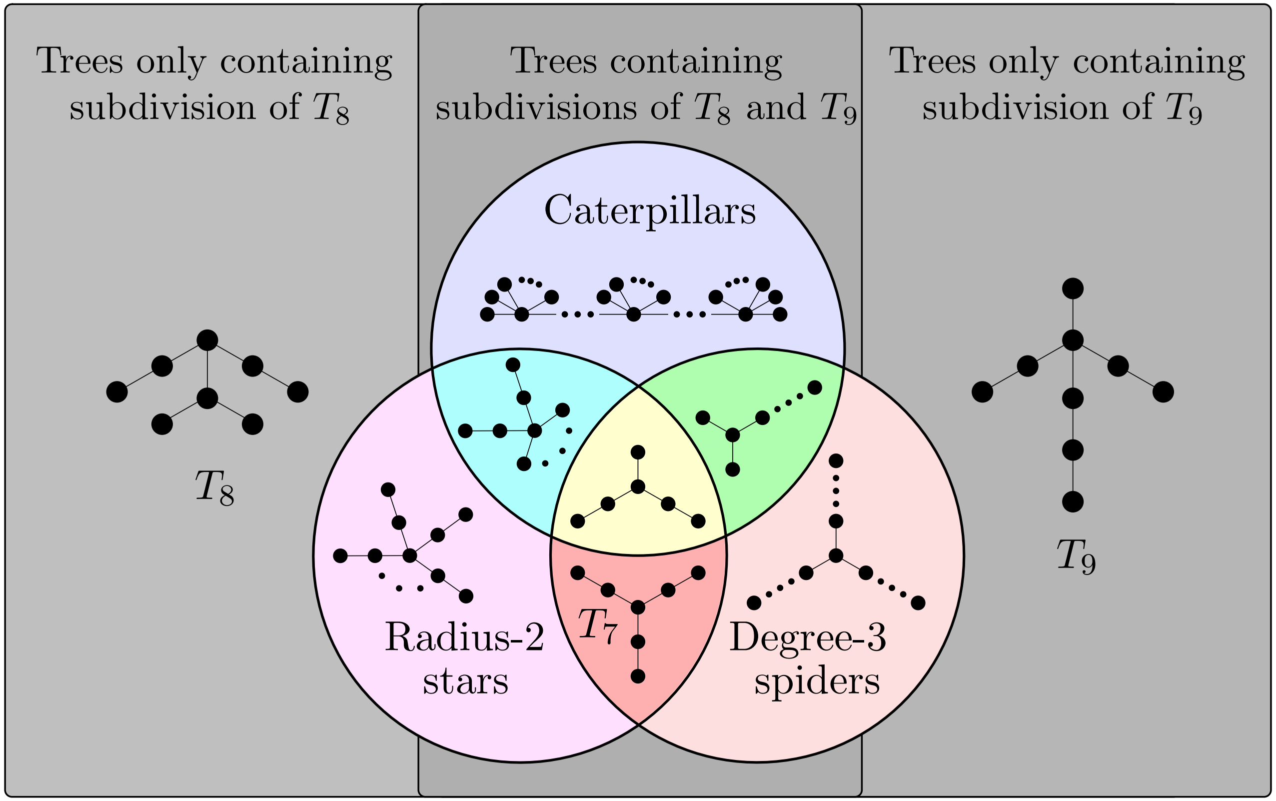
- All ULP trees fall into one of three categories:
  - Caterpillars
  - **Radius-2 stars**



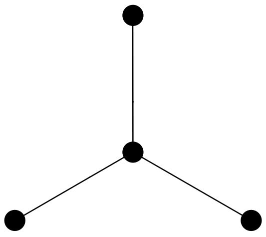


- All ULP trees fall into one of three categories:
  - Caterpillars
  - **Radius-2 stars**

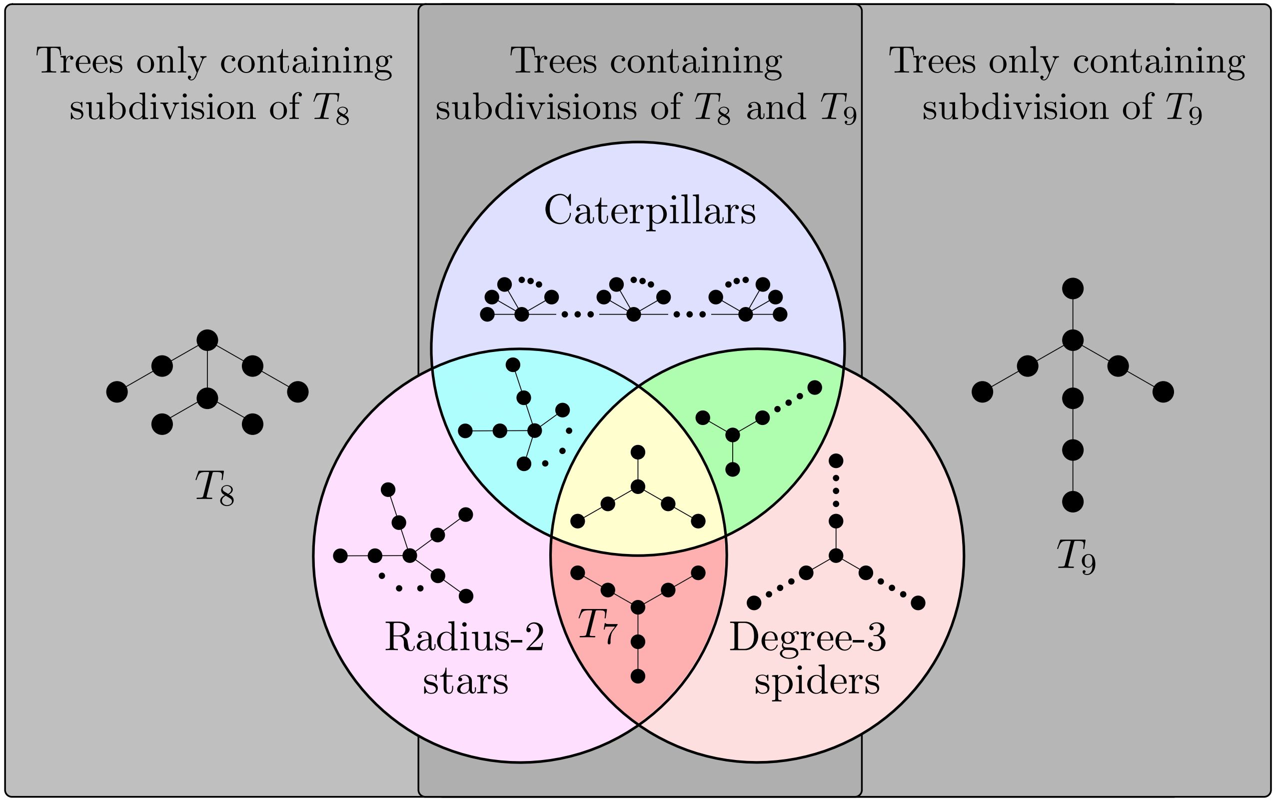




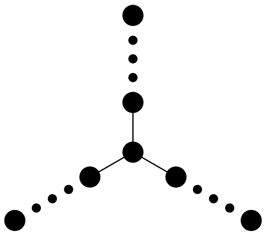
- All ULP trees fall into one of three categories:
  - Caterpillars
  - Radius-2 stars
  - **Degree-3 spiders**



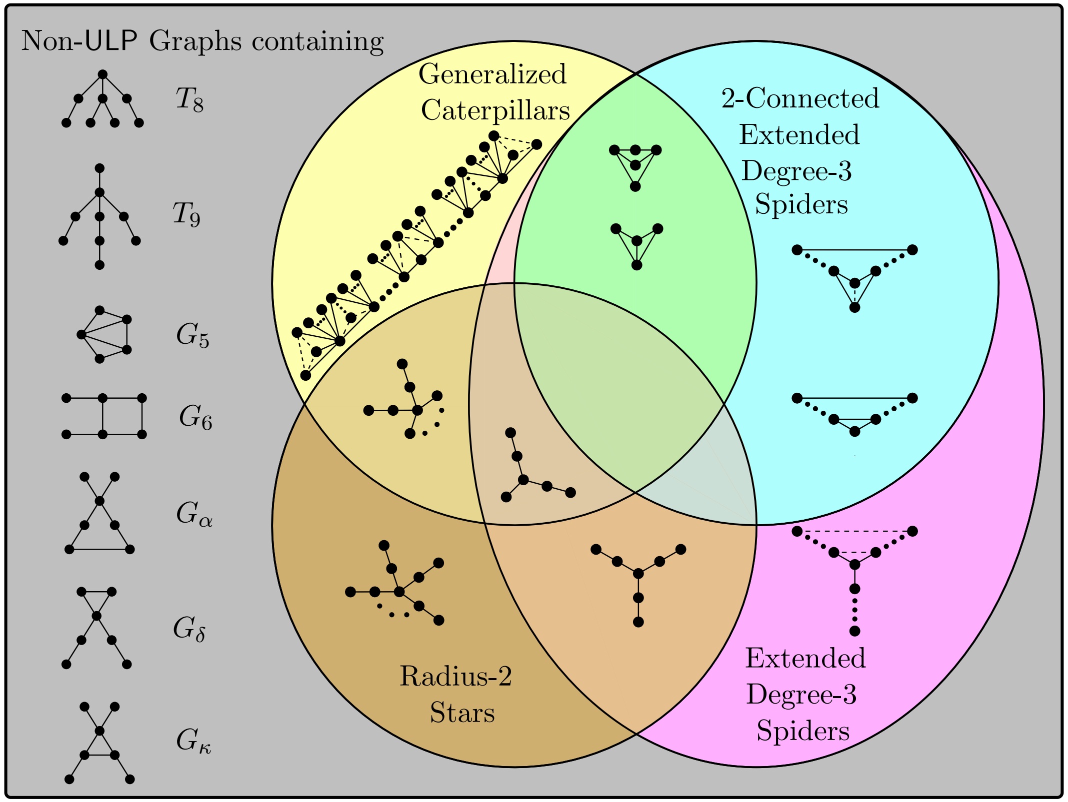
Previous Work – ULP Trees



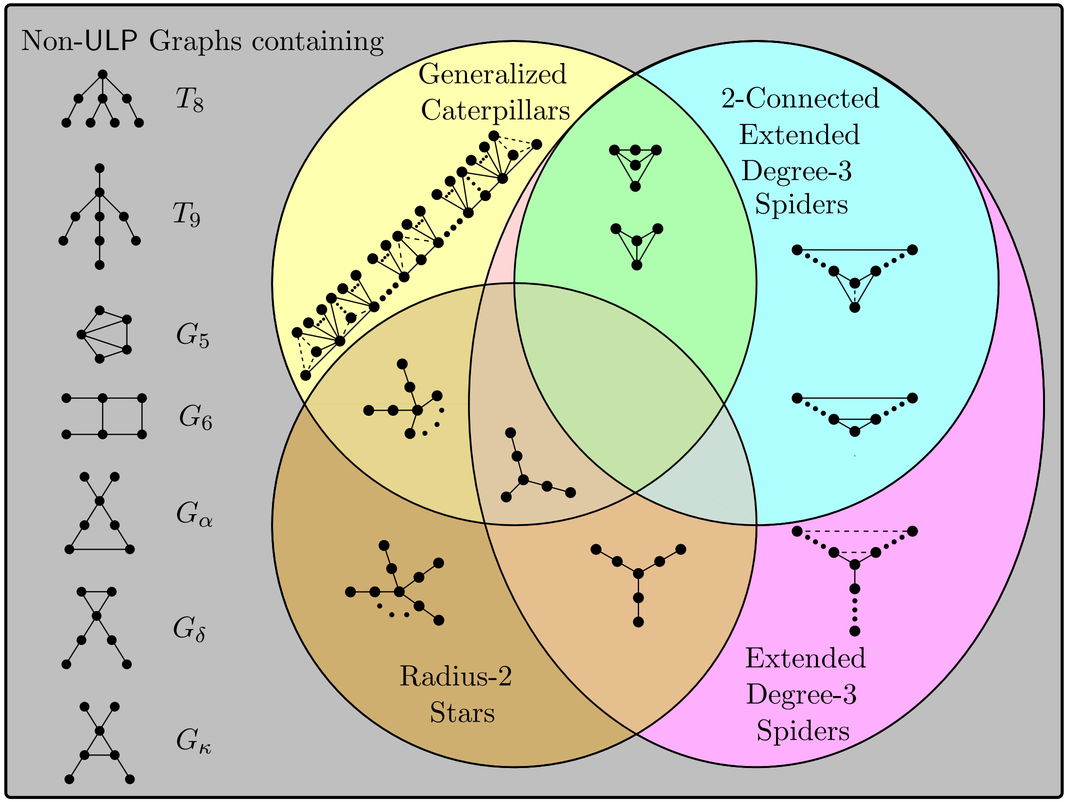
- All ULP trees fall into one of three categories:
  - Caterpillars
  - Radius-2 stars
  - **Degree-3 spiders**

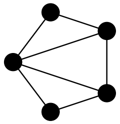


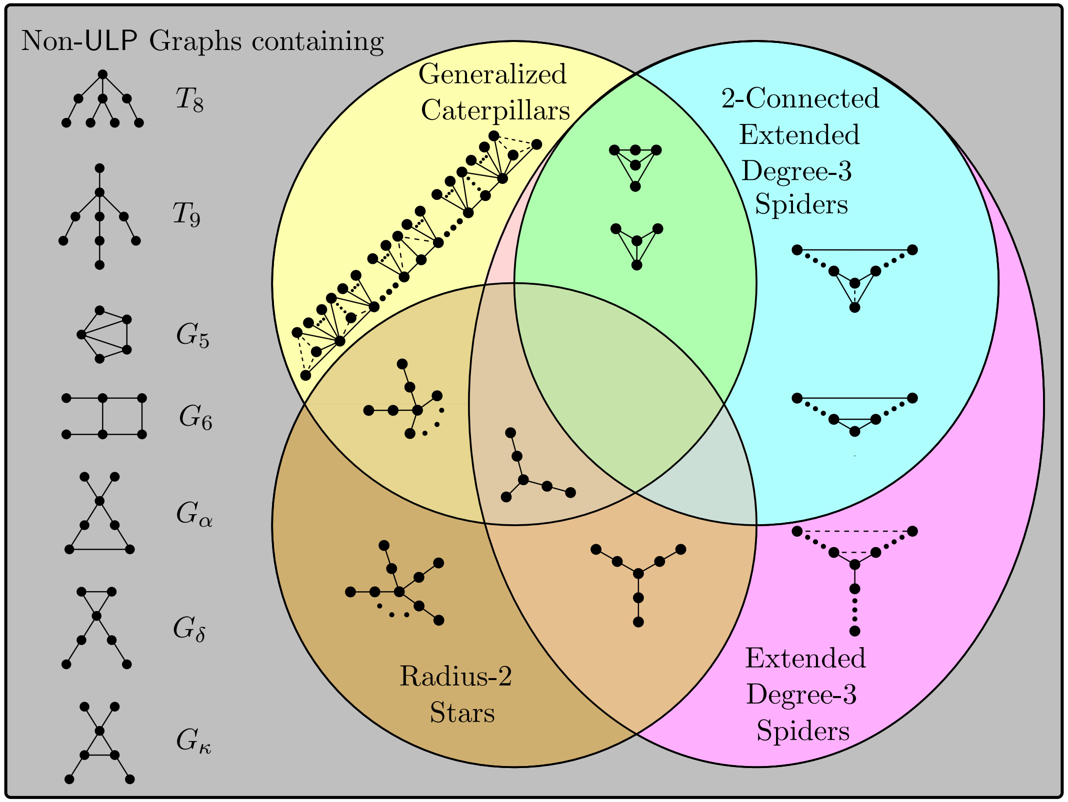
New Results – ULP Graphs



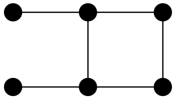
- Characterization of ULP graphs by 7 forbidden subdivisions



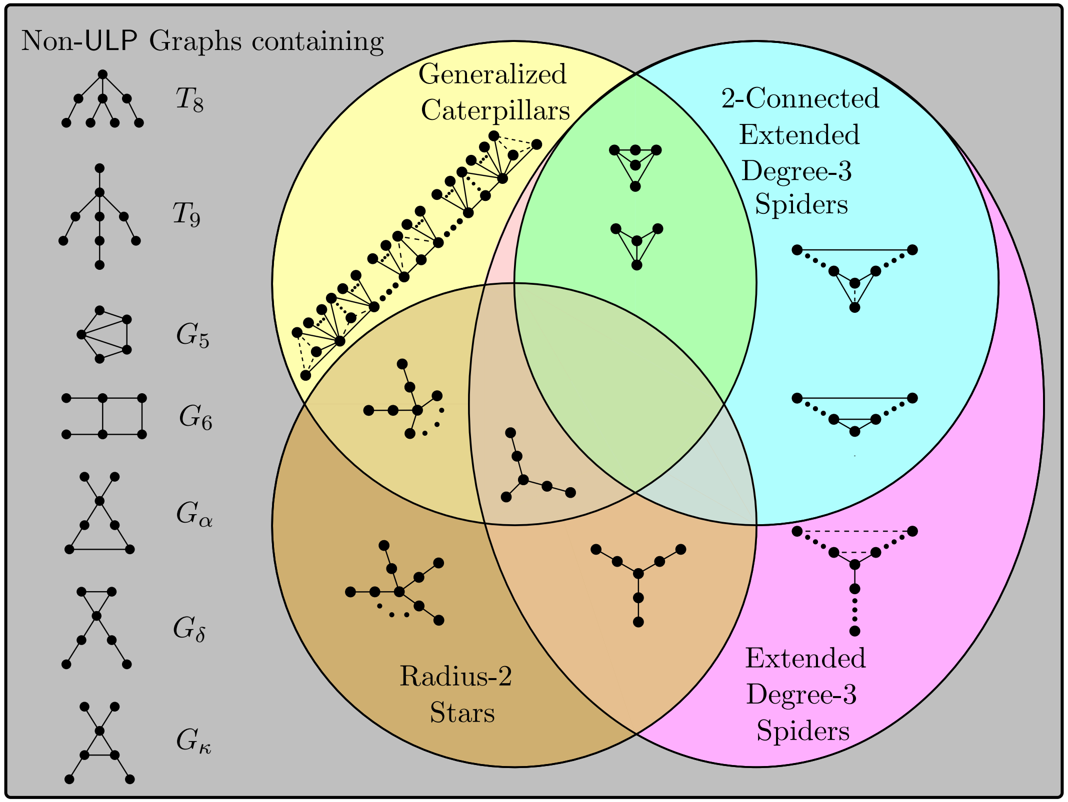
- Characterization of ULP graphs by 7 forbidden subdivisions
  - Graph G_5 with 5 vertices and three nodes of degree 4 



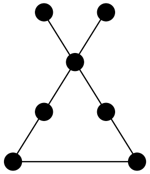
- Characterization of ULP graphs by 7 forbidden subdivisions
  - Graph G_6 with 6 vertices and two nodes of degree 3

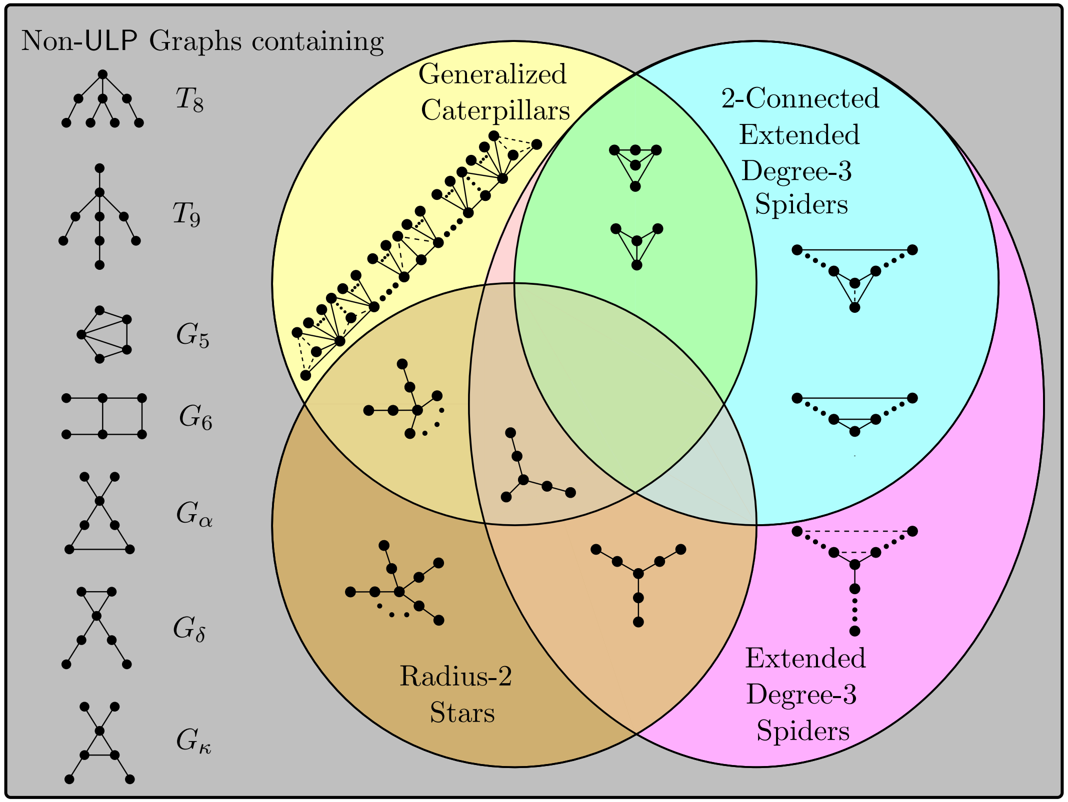


New Results – ULP Graphs

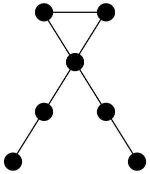


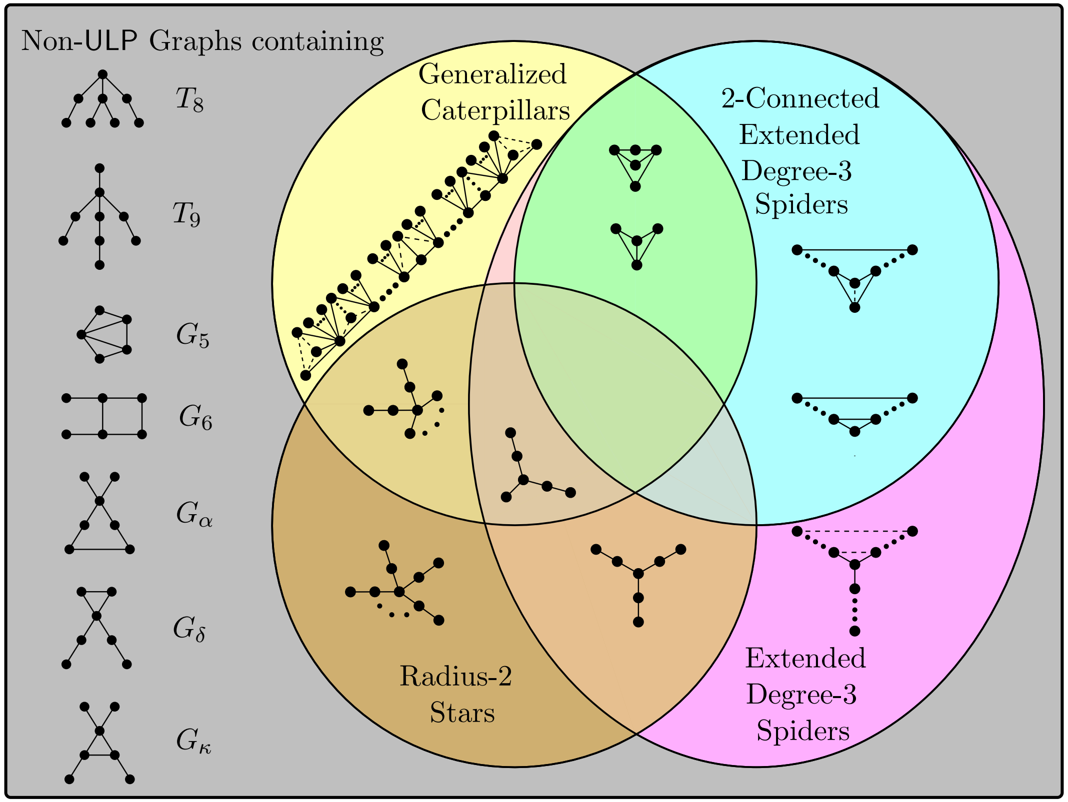
- Characterization of ULP graphs by 7 forbidden subdivisions
  - Graph G_α with 7 vertices and one node of degree 4



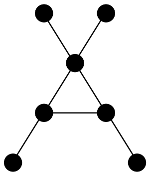


- Characterization of ULP graphs by 7 forbidden subdivisions
  - Graph G_δ with 7 vertices and one node of degree 4

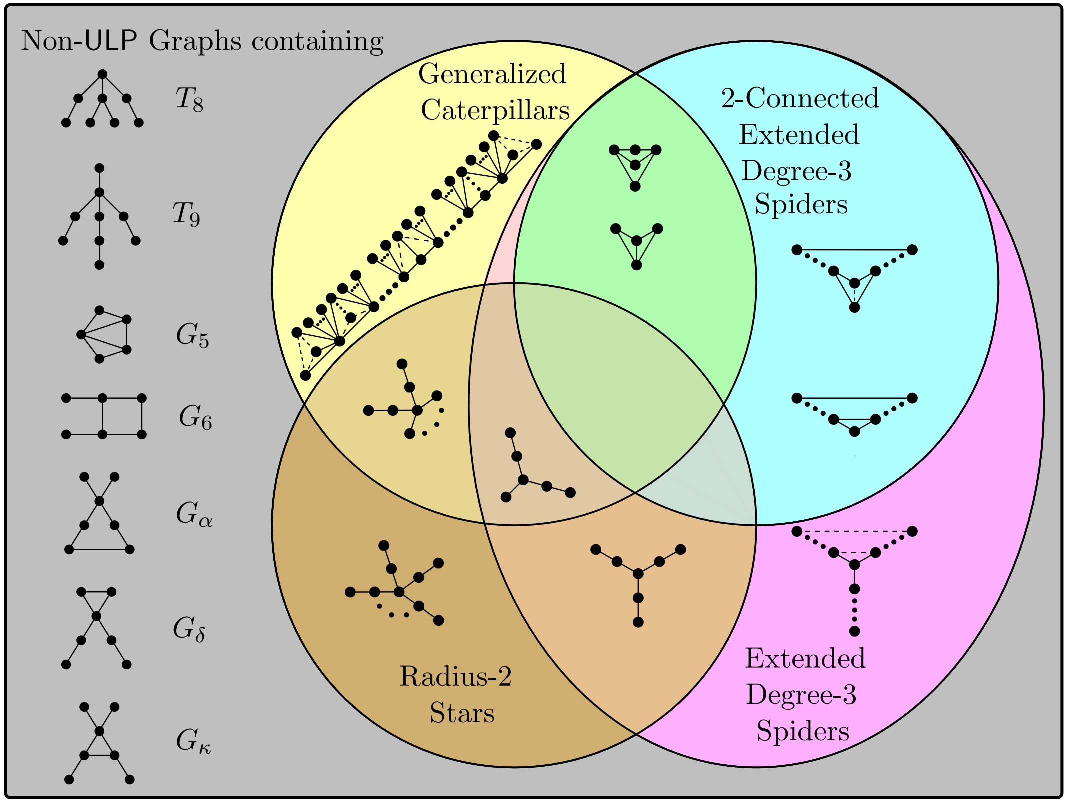




- Characterization of ULP graphs by 7 forbidden subdivisions
  - Graph G_κ with 7 vertices and three nodes of degree 4

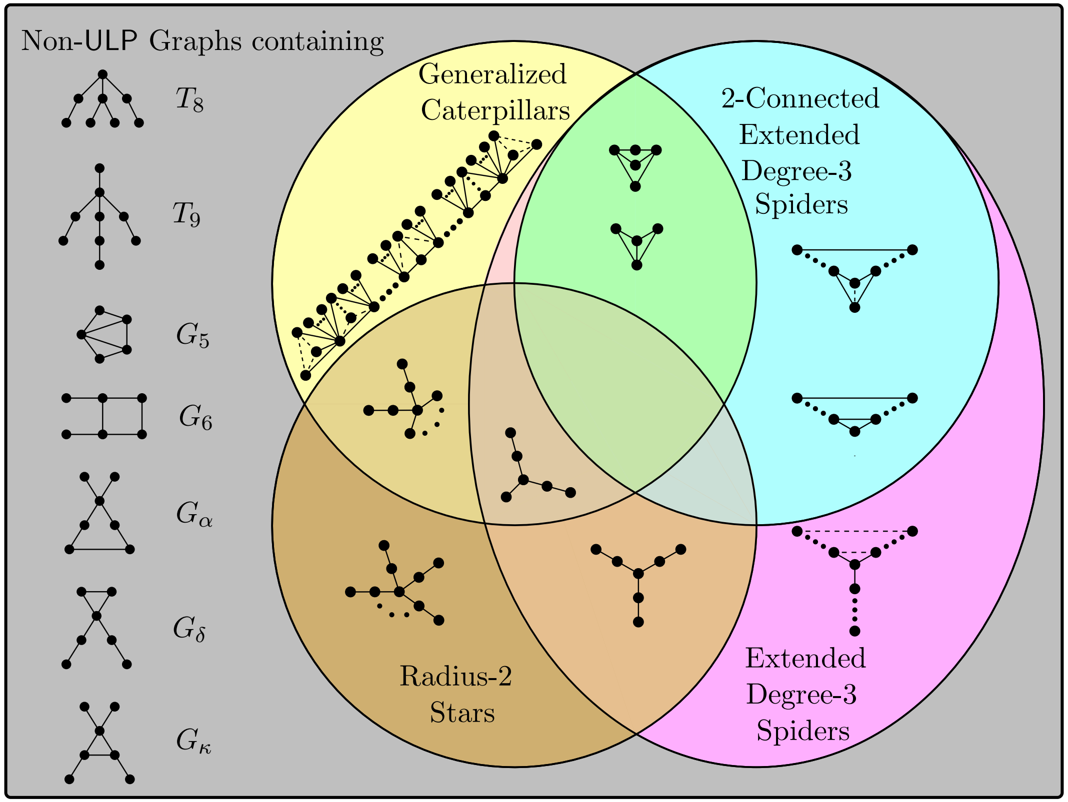


New Results – ULP Graphs

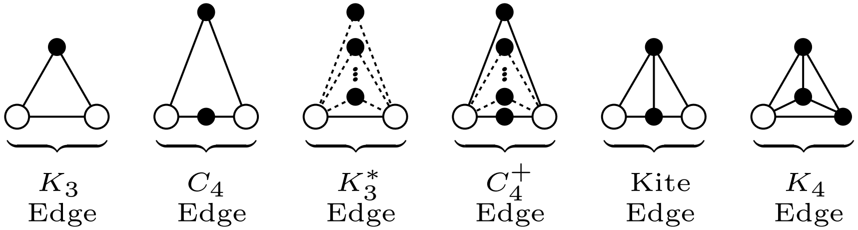


- Characterization of ULP graphs by 7 forbidden subdivisions
- All ULP graphs fall into one of three categories:

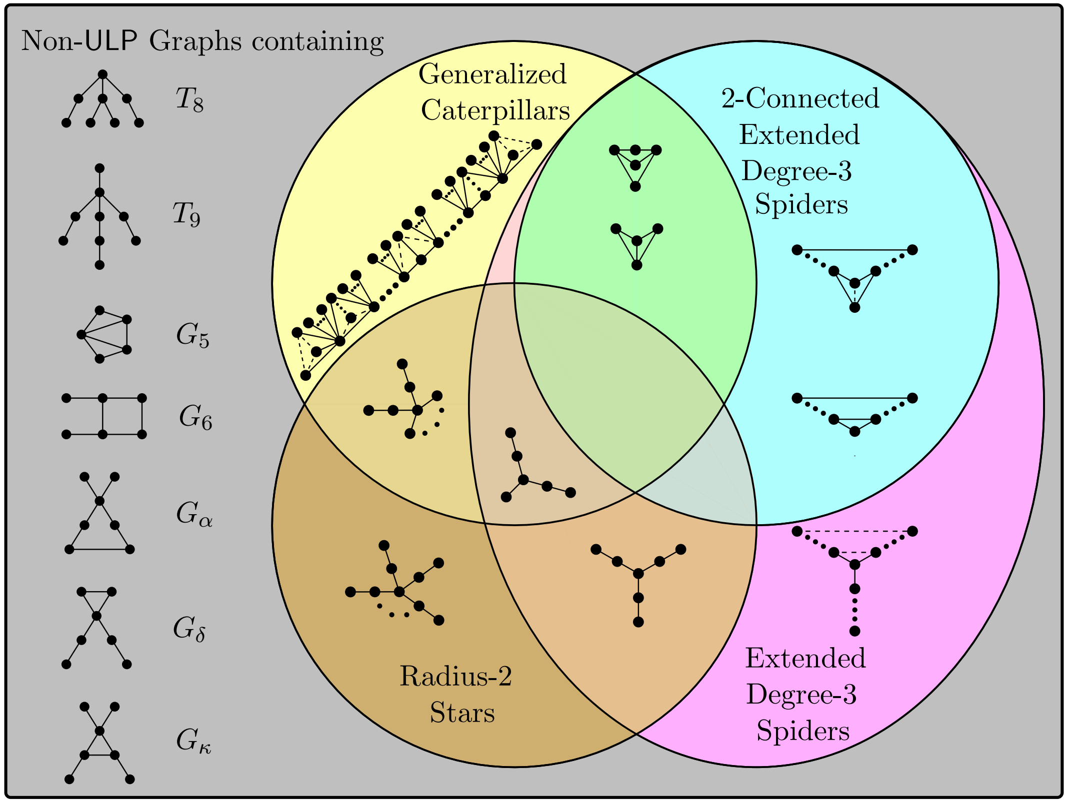
New Results – ULP Graphs



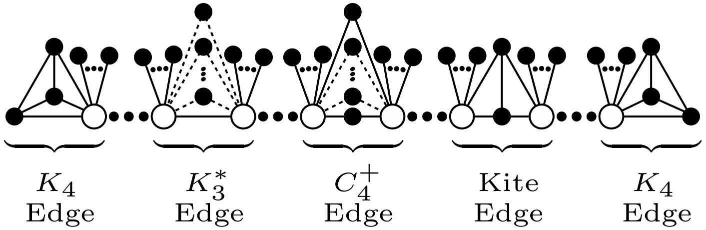
- Characterization of ULP graphs by 7 forbidden subdivisions
- All ULP graphs fall into one of three categories:
  - **Generalized Caterpillars**

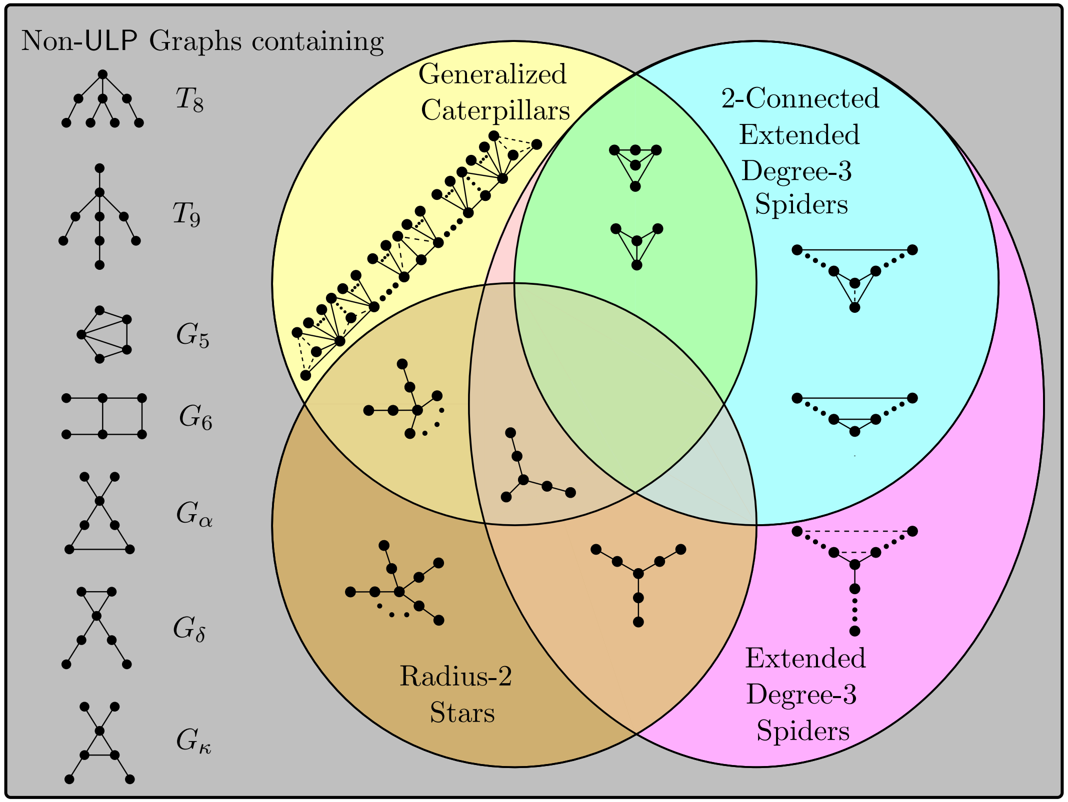


New Results – ULP Graphs

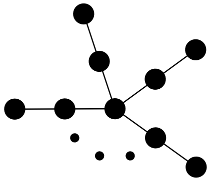


- All ULP graphs fall into one of three categories:
  - **Generalized Caterpillars**

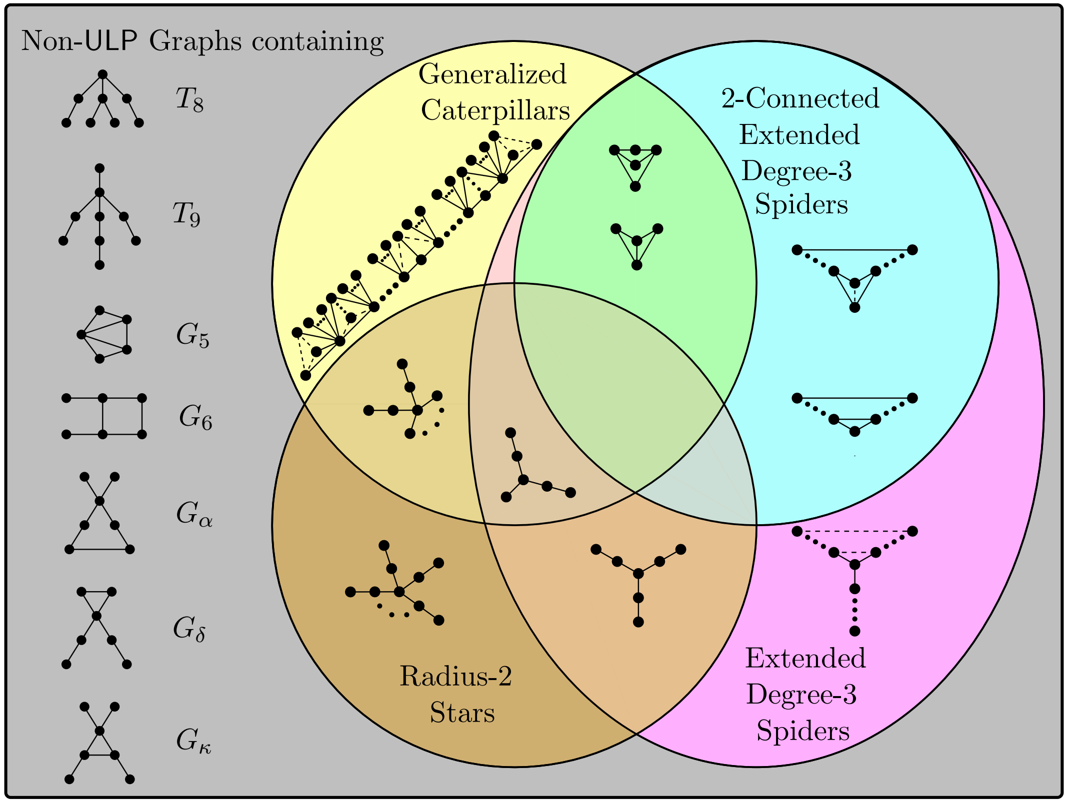




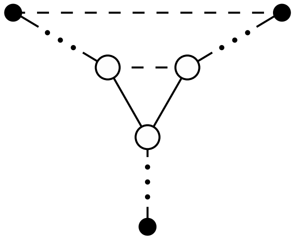
- All ULP graphs fall into one of three categories:
  - Generalized Caterpillars
  - **Radius-2 Stars**



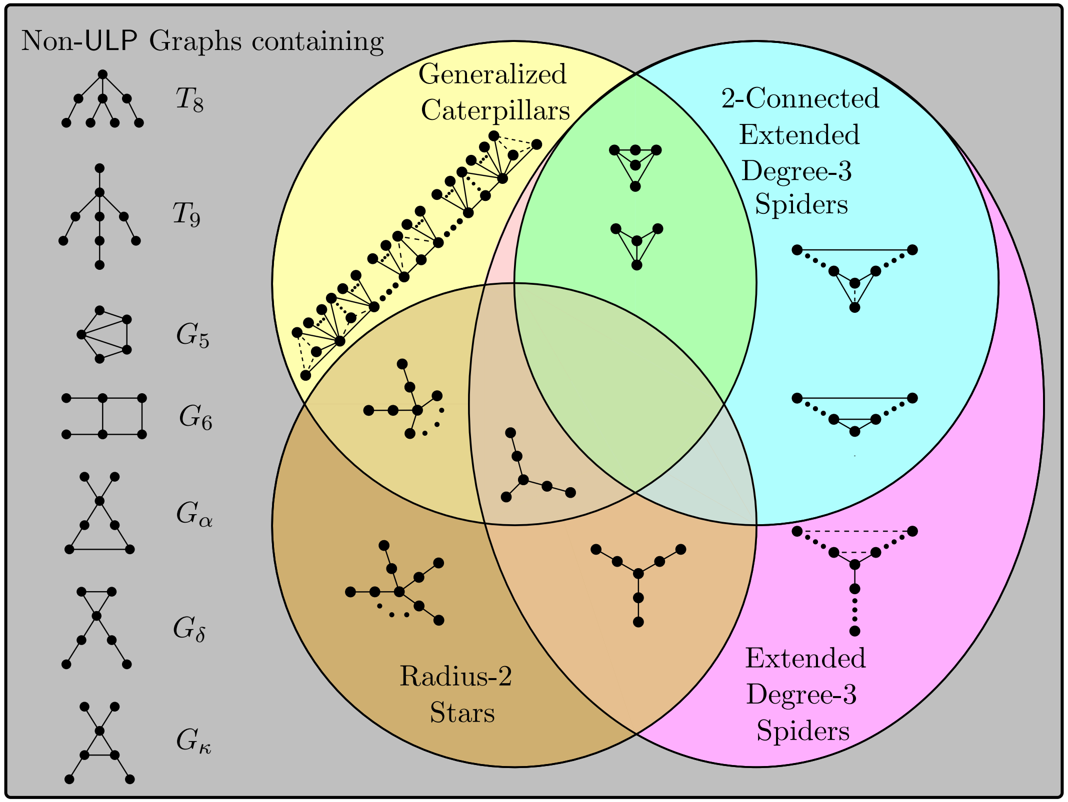
New Results – ULP Graphs



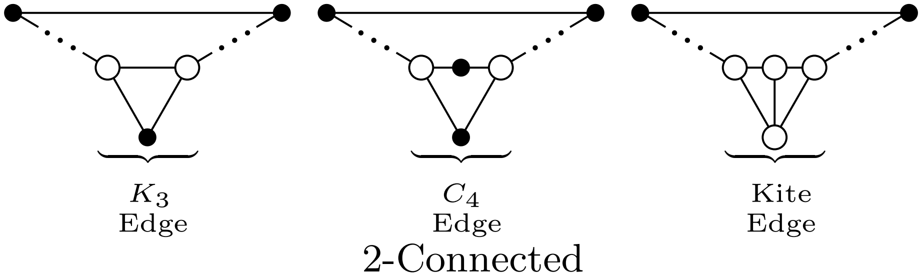
- All ULP graphs fall into one of three categories:
  - Generalized Caterpillars
  - Radius-2 Stars
  - **Extended Degree-3 Spiders**



New Results – ULP Graphs

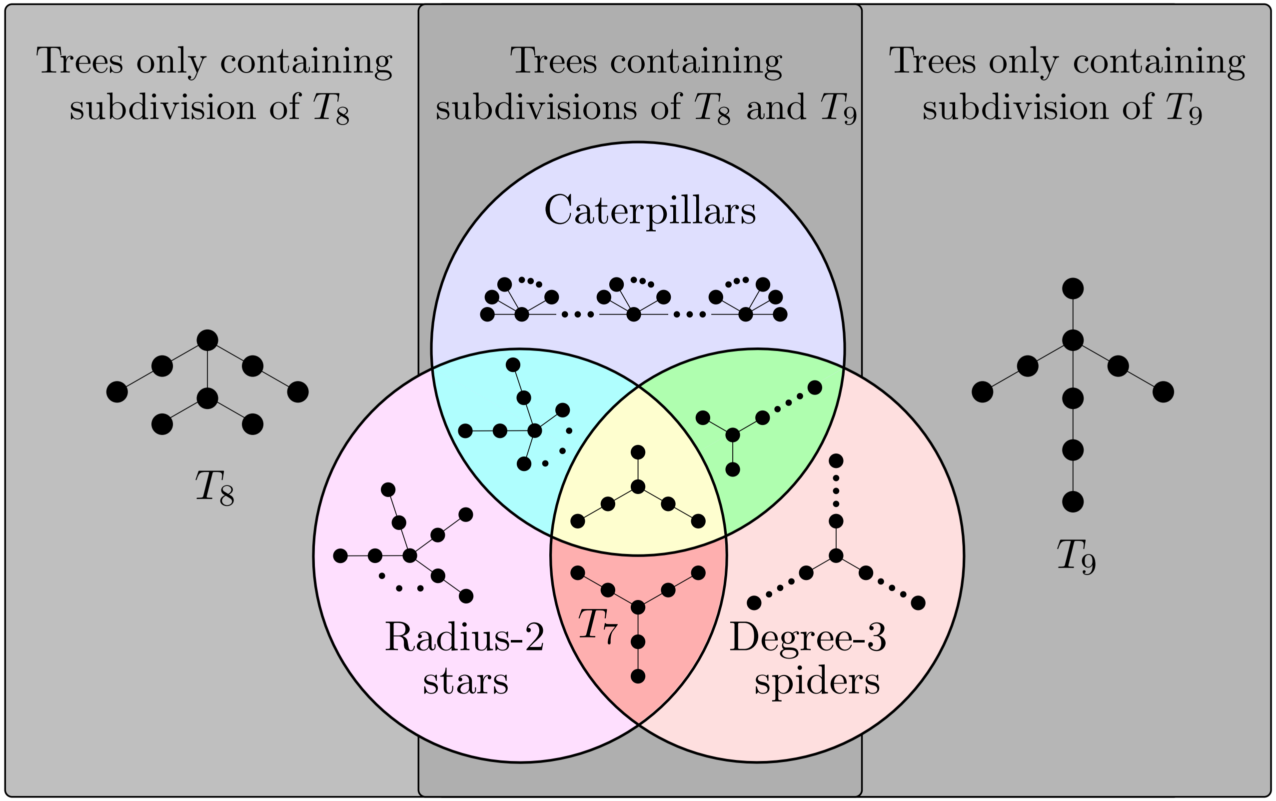


- All ULP graphs fall into one of three categories:
  - Generalized Caterpillars
  - Radius-2 Stars
  - **Extended Degree-3 Spiders**



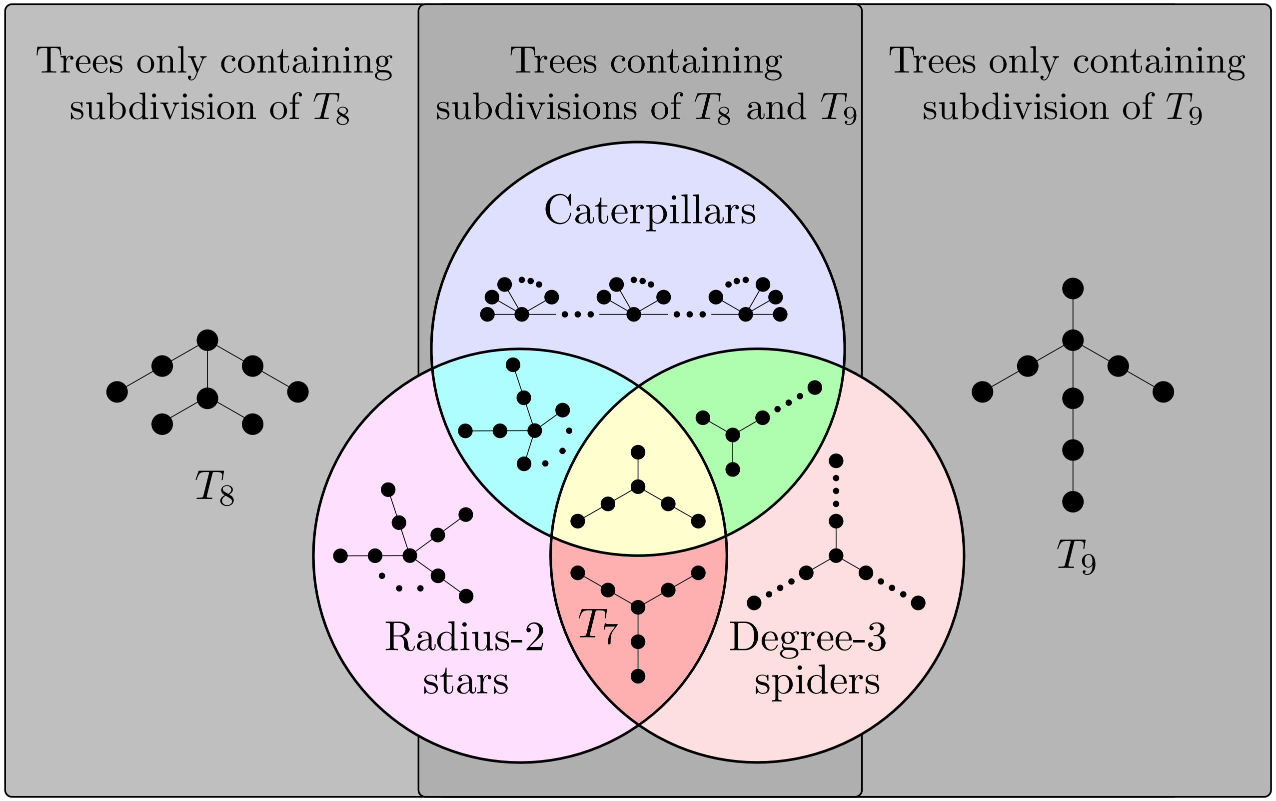
Outline – Unlabeled Level Planar Trees

- Background
- Unlabeled Level Planar Trees



Outline – Unlabeled Level Planar Trees

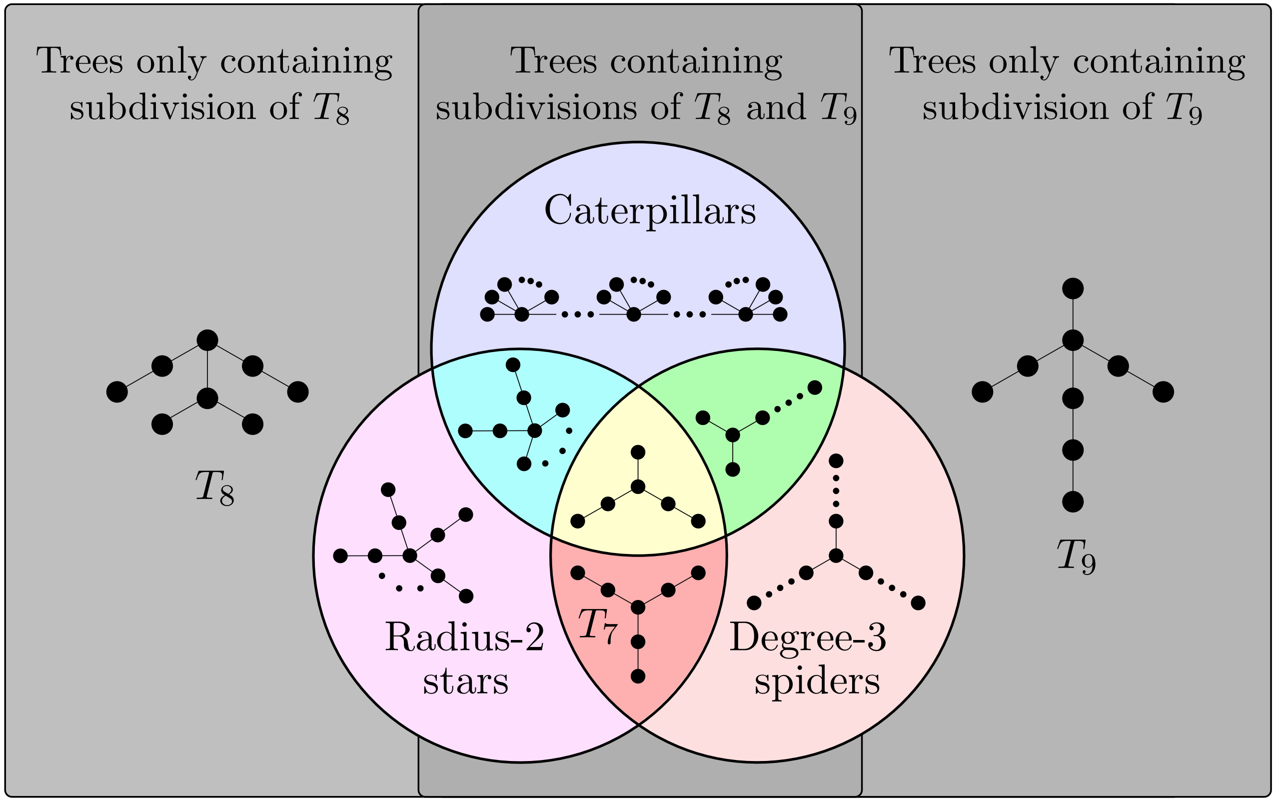
- Background
- Unlabeled Level Planar Trees



  - First show that forbidden trees are not ULP

Outline – Unlabeled Level Planar Trees

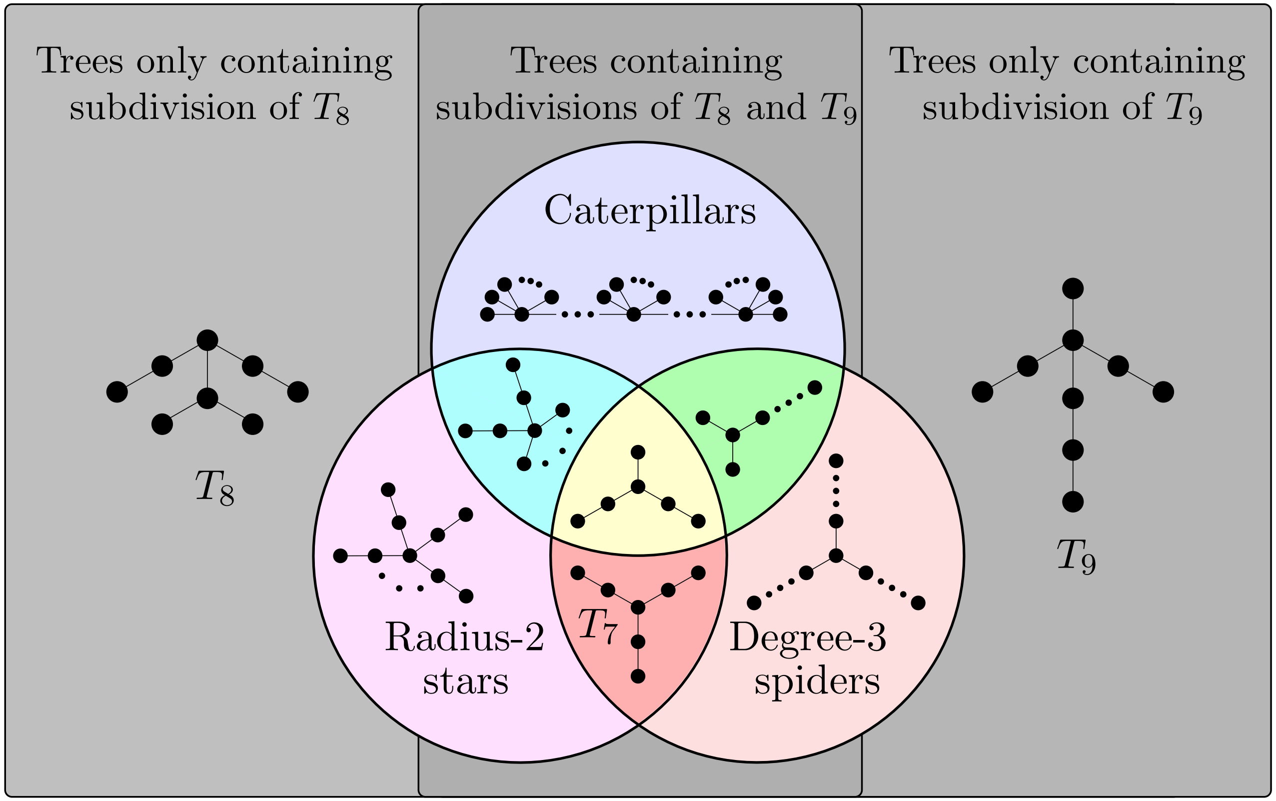
- Background
- Unlabeled Level Planar Trees



- Then show how to draw ULP trees

Outline – Unlabeled Level Planar Trees

- Background
- Unlabeled Level Planar Trees



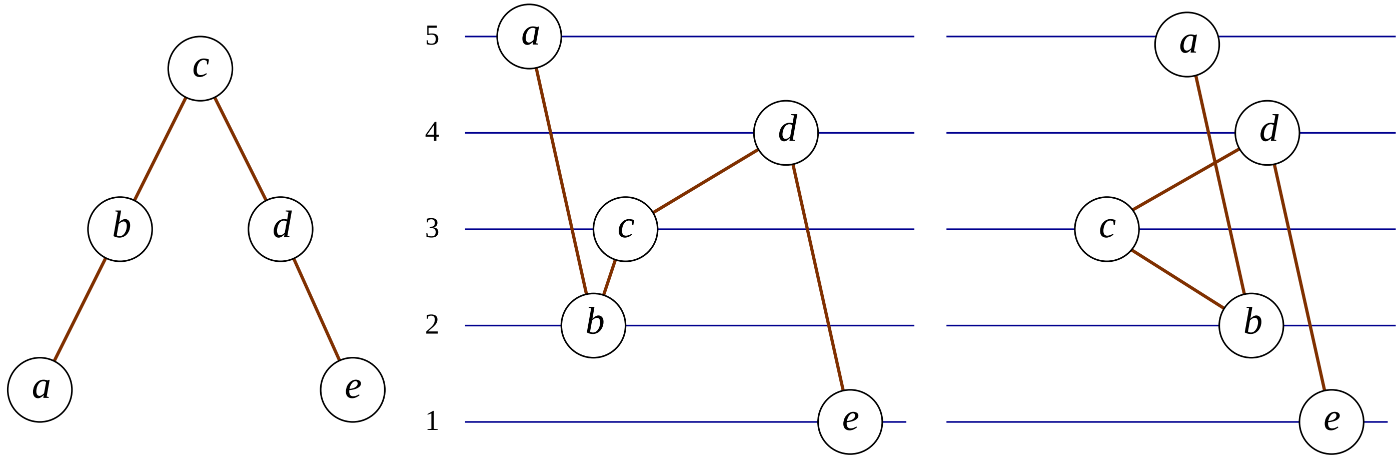
- Finally describe how all trees either
  - Contain a subdivision of T_8 or T_9 OR
  - Are one of the three ULP trees

Key Observation

- Let C be some chain $a{-}b{-}c{-}d{-}e$

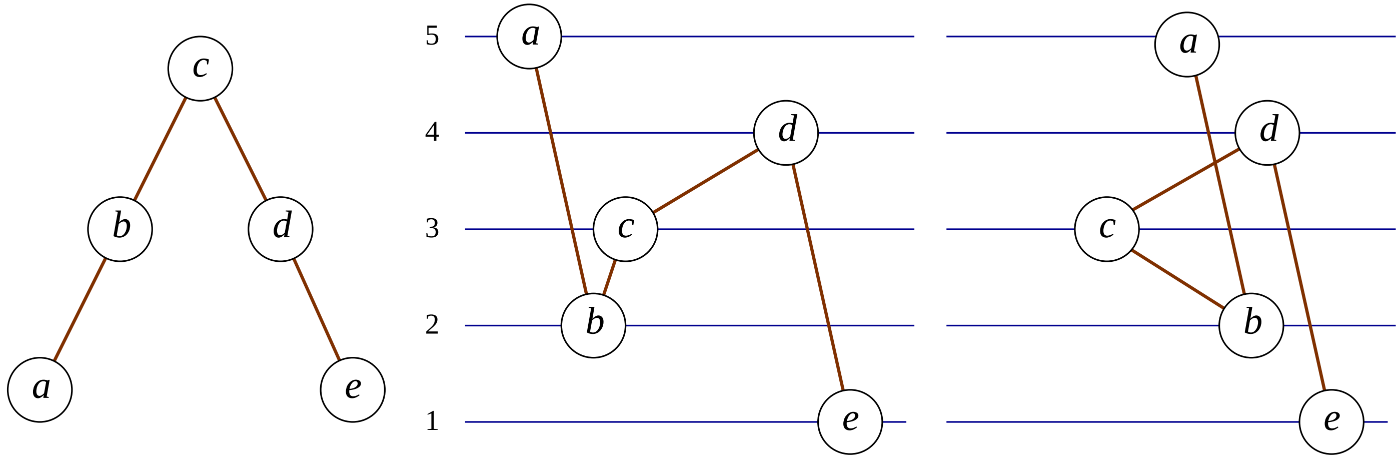
Key Observation

- Let C be some chain $a{-}b{-}c{-}d{-}e$
  - $\phi(a) < \phi(d) < \phi(c) < \phi(b) < \phi(e)$



Key Observation

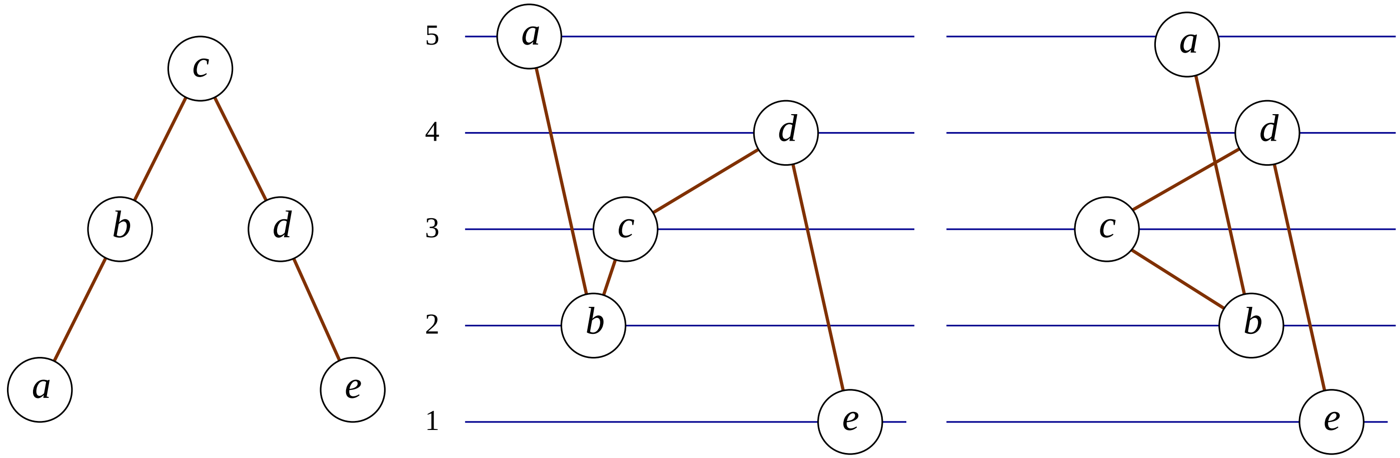
- Let C be some chain a—b—c—d—e
  - $\phi(a) < \phi(d) < \phi(c) < \phi(b) < \phi(e)$



- c cannot be leftmost or rightmost without forcing a crossing

Key Observation

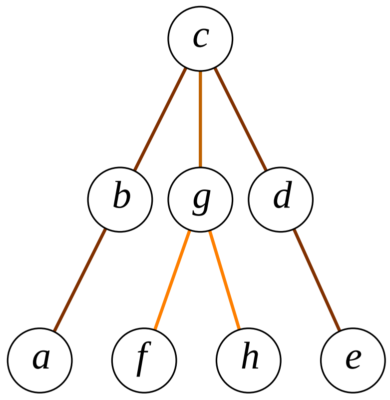
- Let C be some chain a—b—c—d—e
  - $\phi(a) < \phi(d) < \phi(c) < \phi(b) < \phi(e)$



- c cannot be leftmost or rightmost without forcing a crossing
  - Can use this property to prove T_8 and T_9 are not ULP

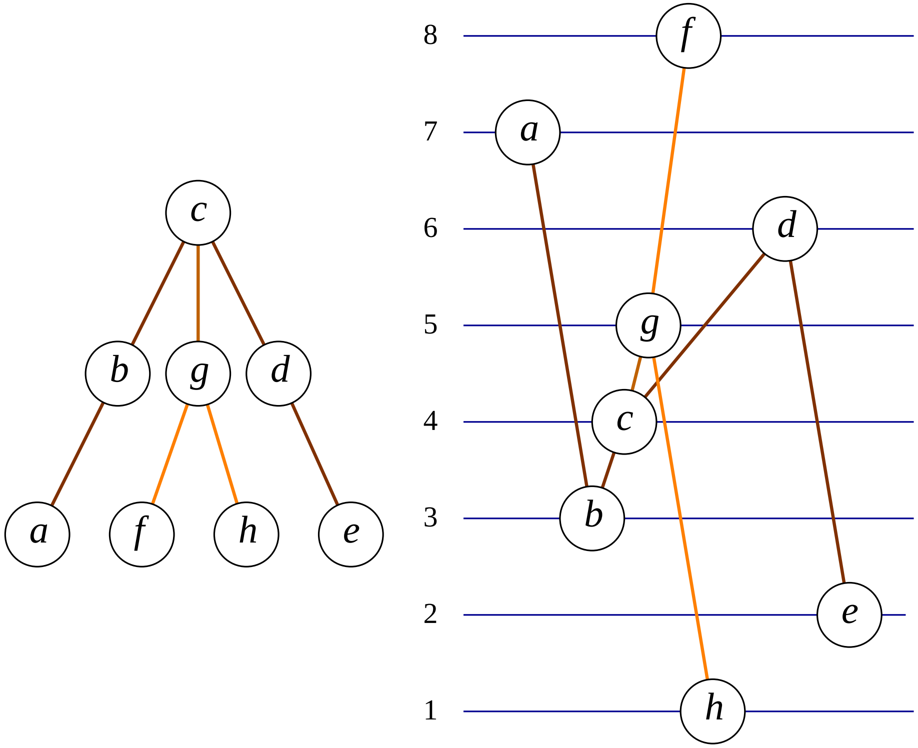
Forbidden Trees – T_8

- Let C be the chain a–b–c–d–e



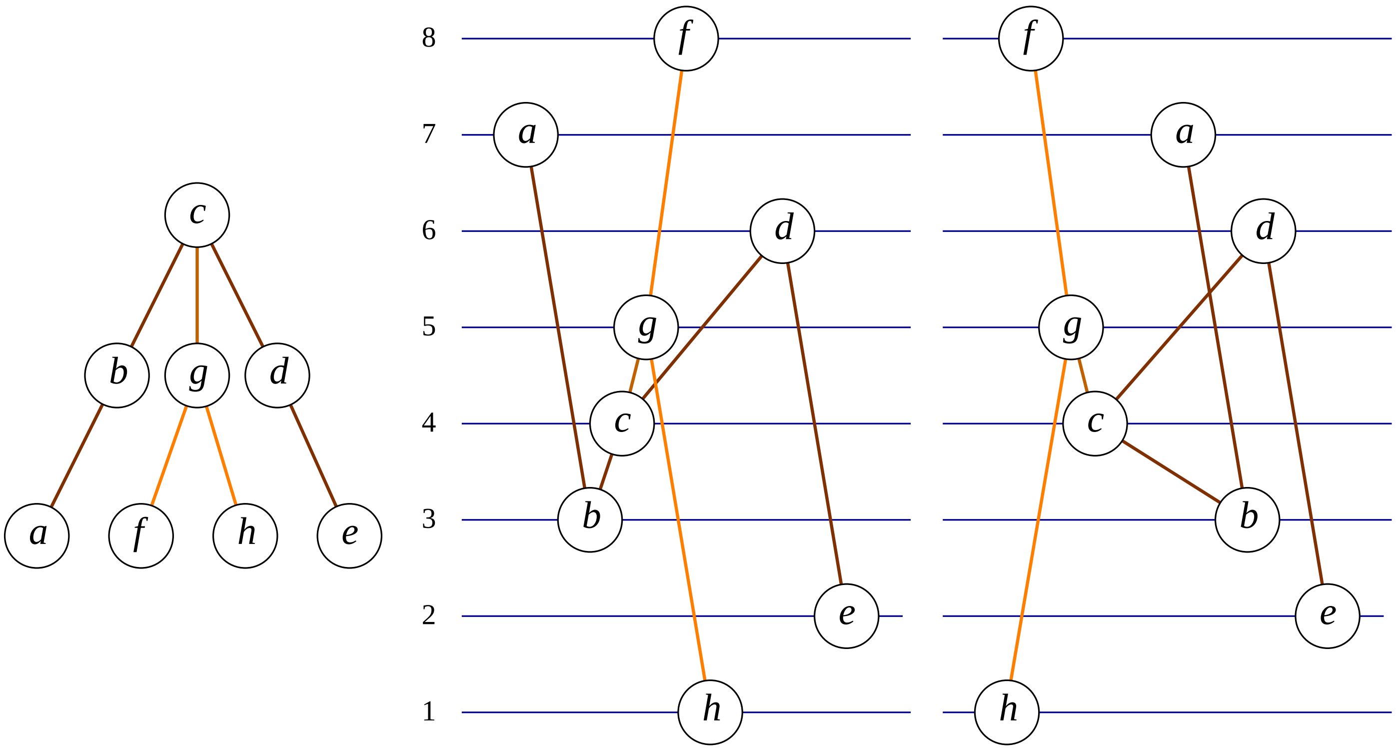
Forbidden Trees – T_8

- Let C be the chain $a{-}b{-}c{-}d{-}e$
  - $\phi(\{a, f\}) < \phi(d) < \phi(\{c, g\}) < \phi(b) < \phi(\{e, h\})$



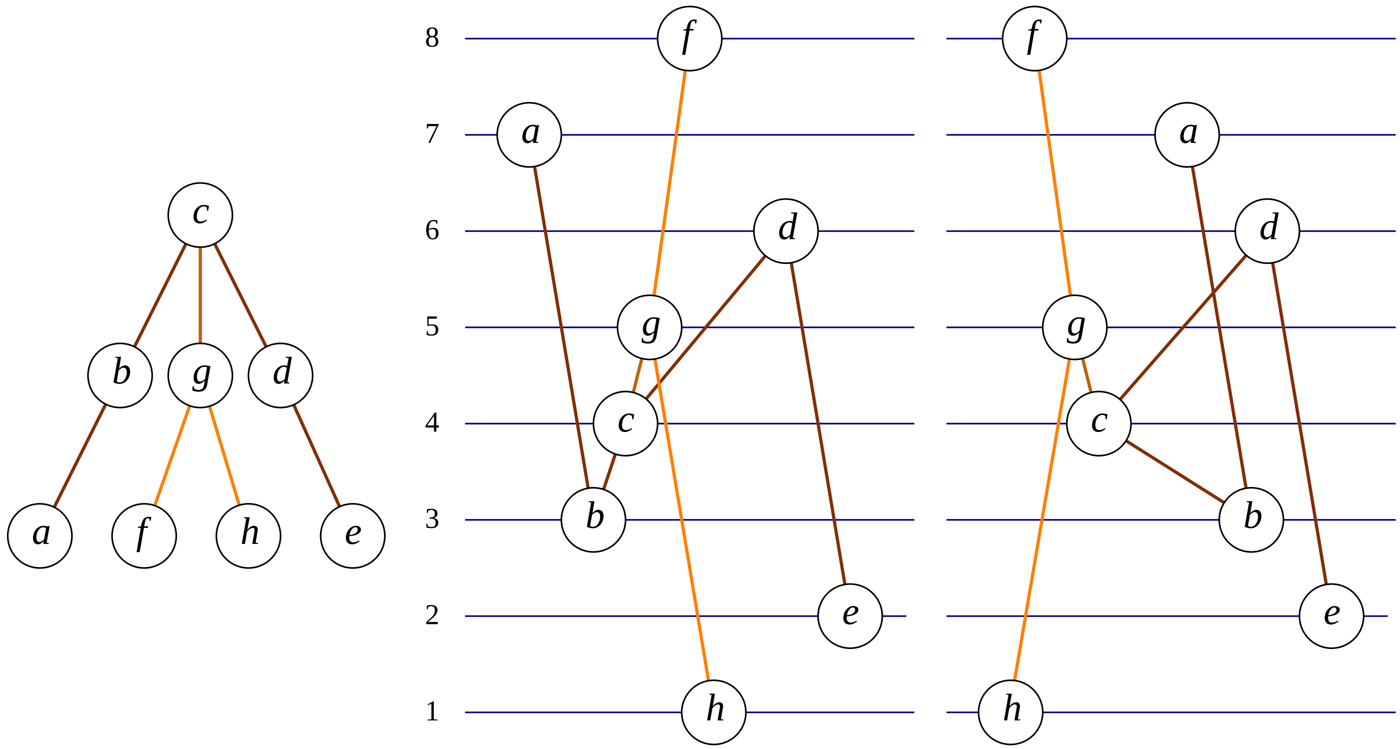
Forbidden Trees – T_8

- Let C be the chain $a–b–c–d–e$
  - $\phi(\{a, f\}) < \phi(d) < \phi(\{c, g\}) < \phi(b) < \phi(\{e, h\})$



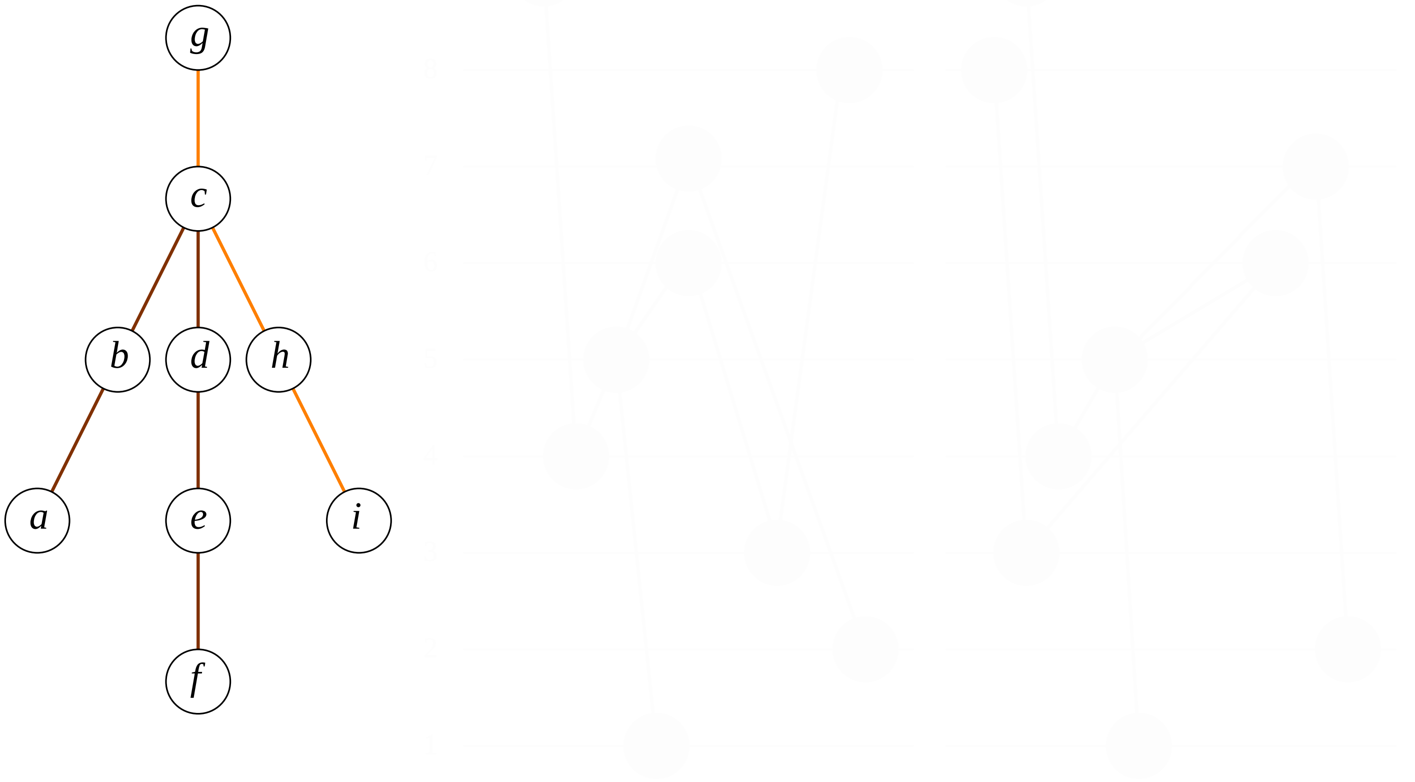
Forbidden Trees – T_8

- Let C be the chain a–b–c–d–e
  - $\phi(\{a,f\}) < \phi(d) < \phi(\{c,g\}) < \phi(b) < \phi(\{e,h\})$



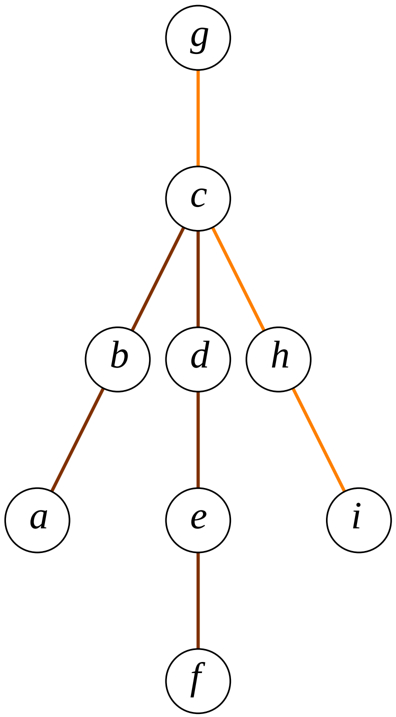
Lemma 1 *Any tree containing a subdivision of* T_8 *cannot be ULP.*

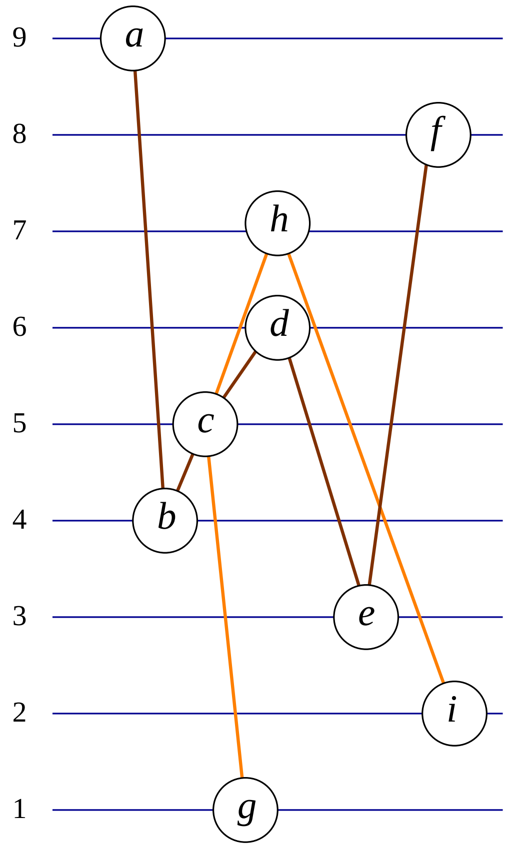
- Let C be the chain $a{-}b{-}c{-}d{-}e$



Forbidden Trees – T_9

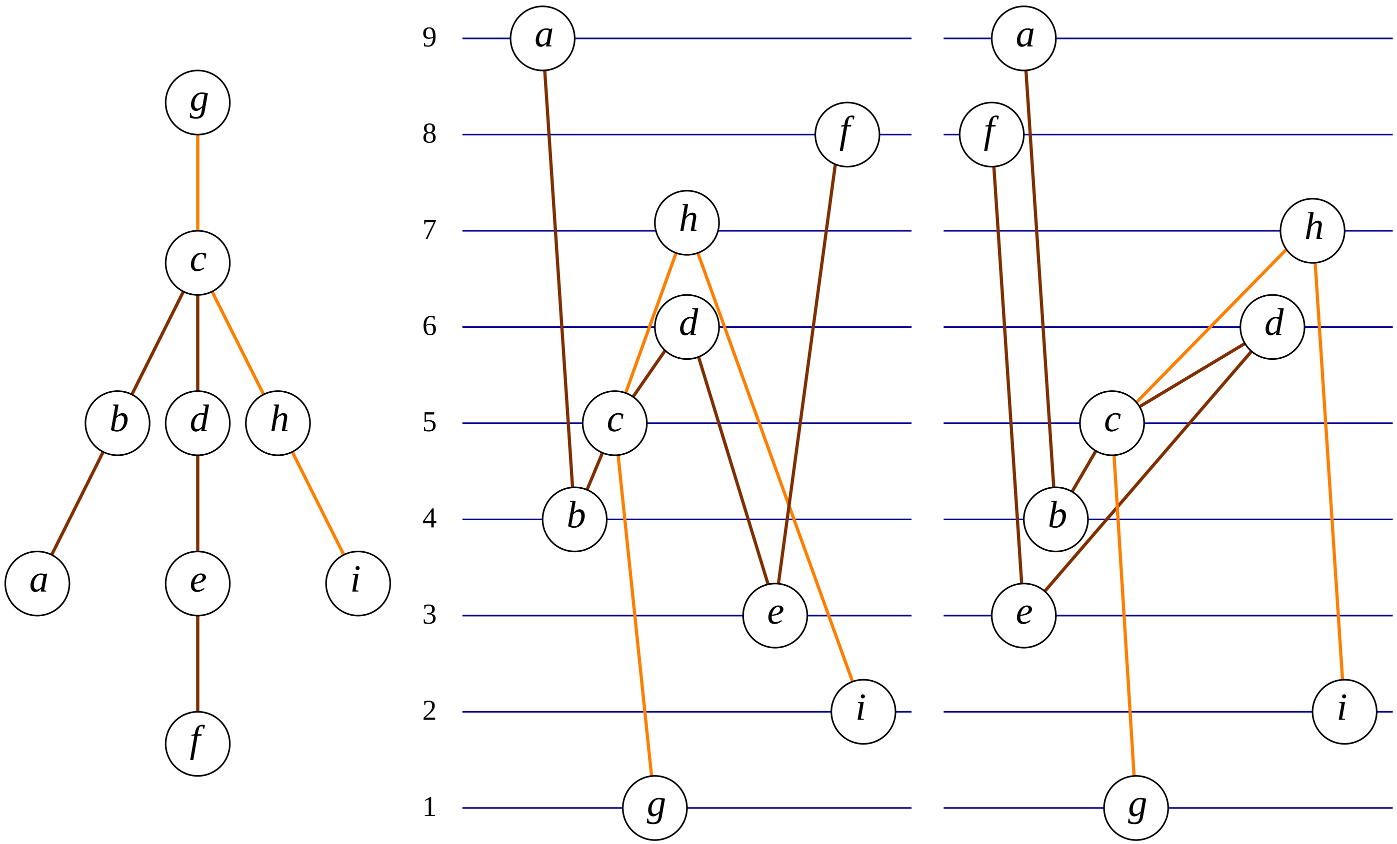
- Let C be the chain $a{-}b{-}c{-}d{-}e$
  - $\phi(\{a, f\}) < \phi(h) < \phi(d) < \phi(c) < \phi(b) < \phi(e) < \phi(\{g, i\})$





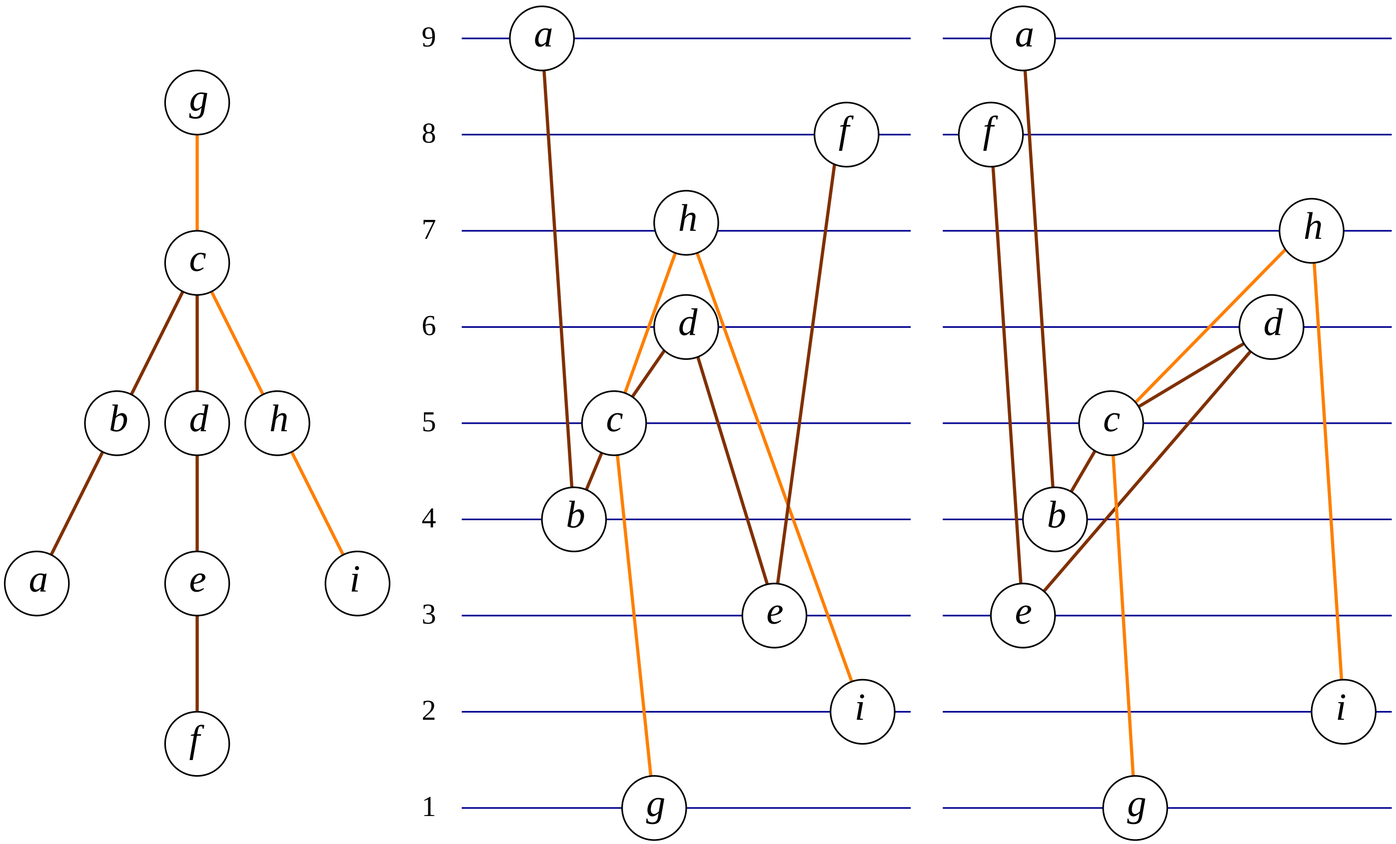
Forbidden Trees – T_9

- Let C be the chain $a–b–c–d–e$
  - $\phi(\{a, f\}) < \phi(h) < \phi(d) < \phi(c) < \phi(b) < \phi(e) < \phi(\{g, i\})$

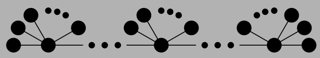


Forbidden Trees – T_9

- Let C be the chain $a–b–c–d–e$
  - $\phi(\{a,f\}) < \phi(h) < \phi(d) < \phi(c) < \phi(b) < \phi(e) < \phi(\{g,i\})$



Lemma 2 *Any tree containing T_9 cannot be ULP.*

Drawing Algorithms – Caterpillar 

- Drawing a caterpillar in linear time:

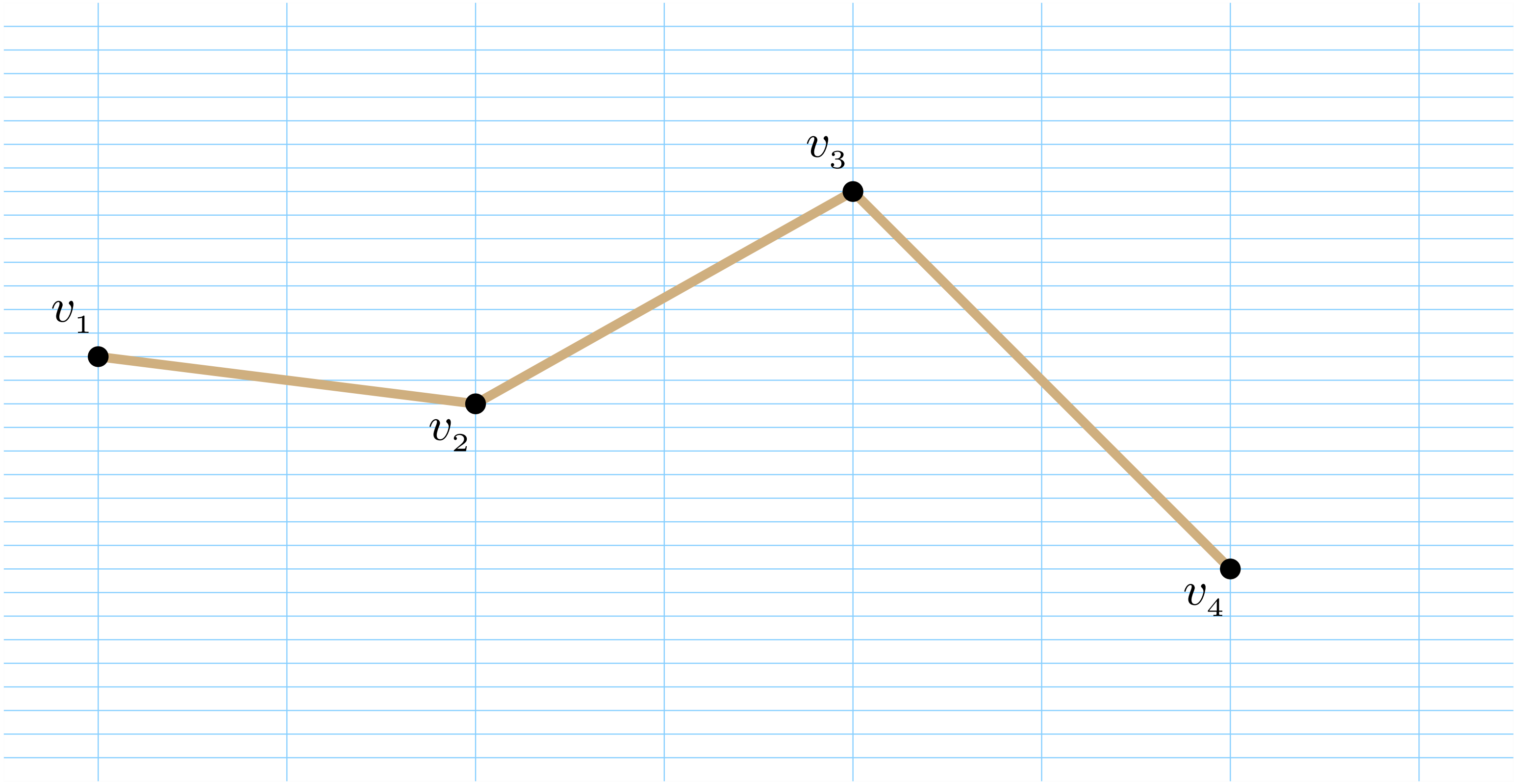
Drawing Algorithms – Caterpillar

- Drawing a caterpillar in linear time:

Lemma 3 *(Brass* et al.*, 2003) A plane drawing of n-vertex caterpillar $T(V, E)$ with an m-vertex spine can be drawn in $O(n)$ time on a $2m \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

- Drawing a caterpillar in linear time:

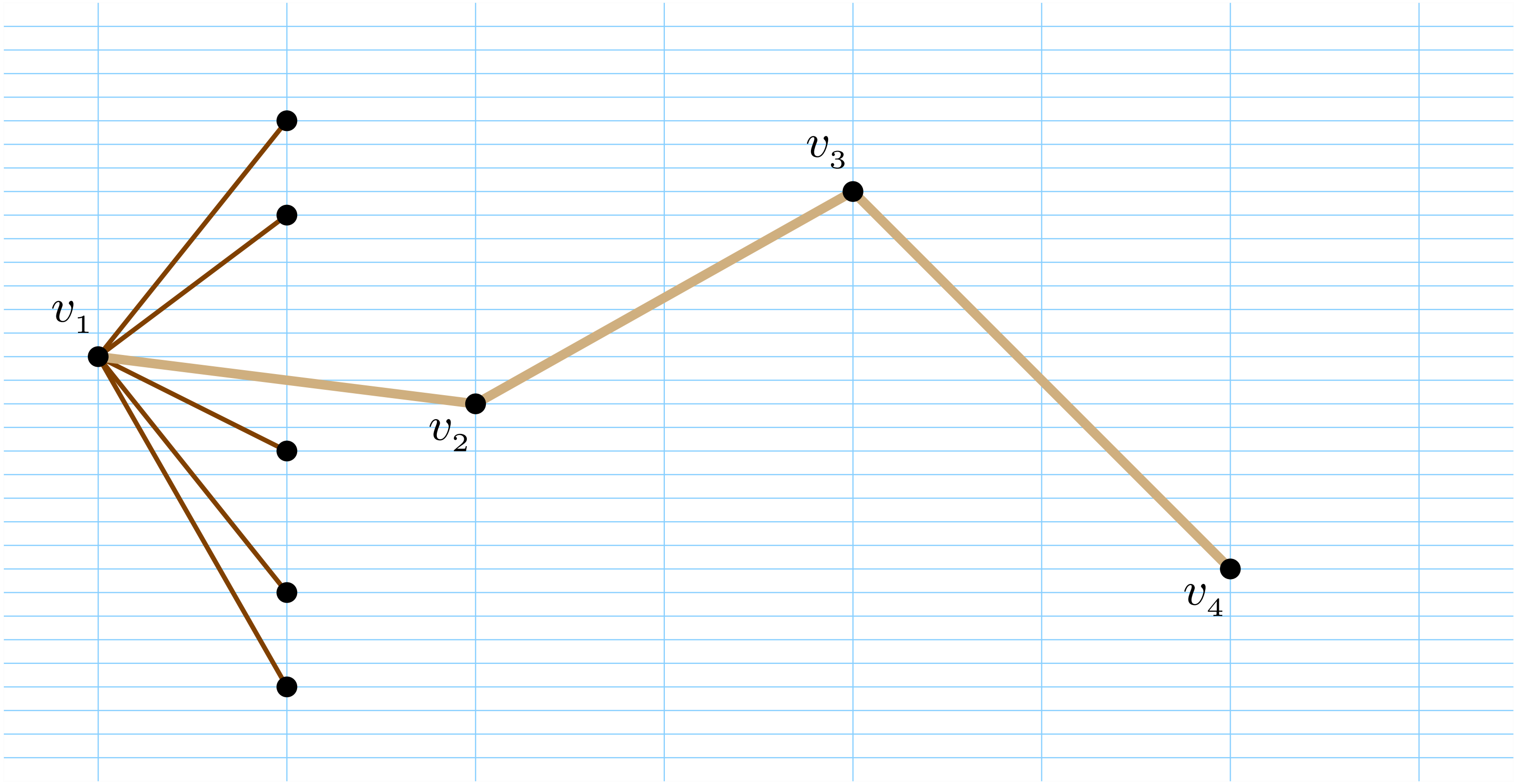
Lemma 3 *(Brass et al., 2003) A plane drawing of n-vertex caterpillar $T(V, E)$ with an m-vertex spine can be drawn in $O(n)$ time on a $2m \times n$ grid for any leveling* $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.



Drawing Algorithms – Caterpillar

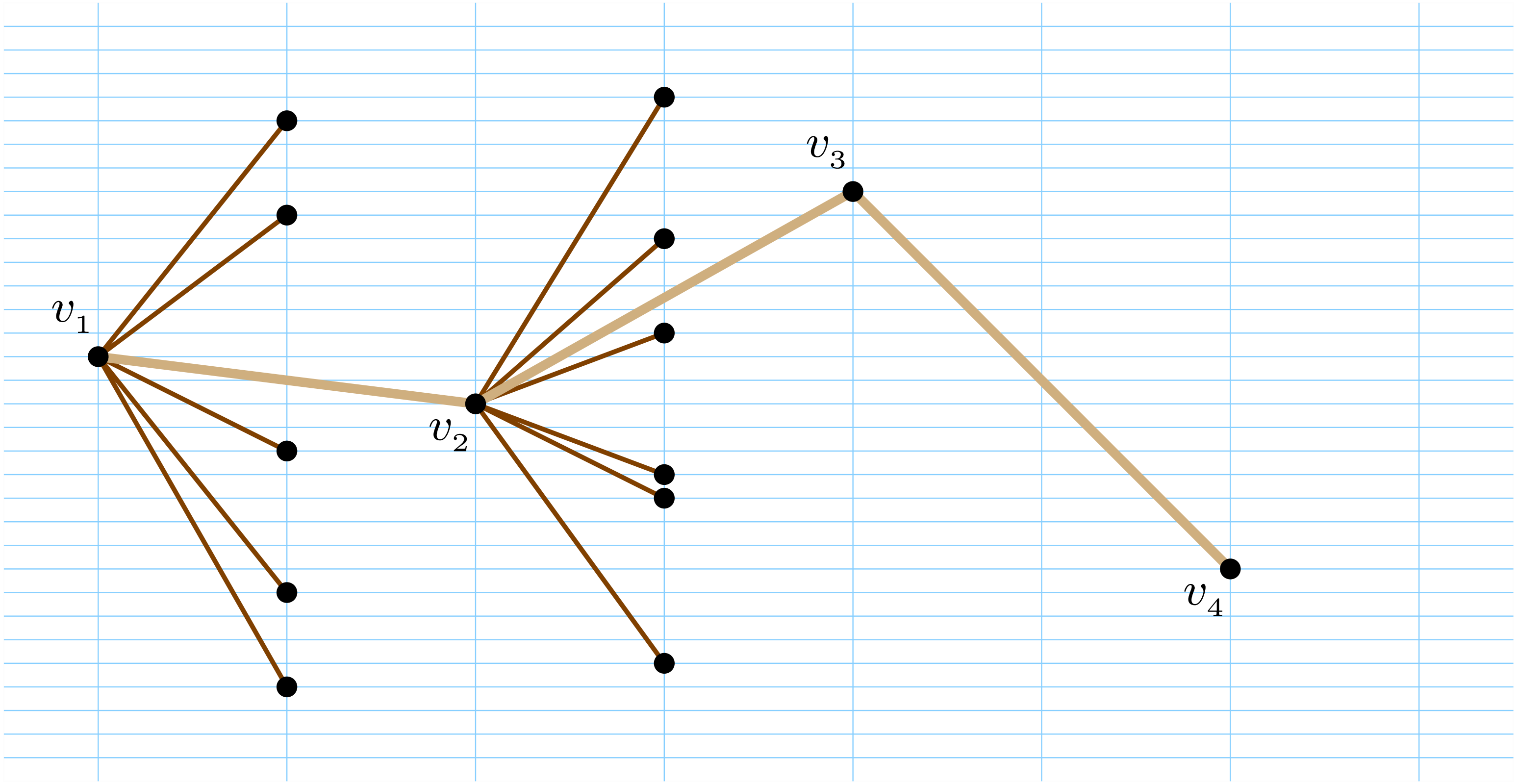
- Drawing a caterpillar in linear time:

Lemma 3 *(Brass et al., 2003) A plane drawing of n-vertex caterpillar $T(V, E)$ with an m-vertex spine can be drawn in $O(n)$ time on a $2m \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



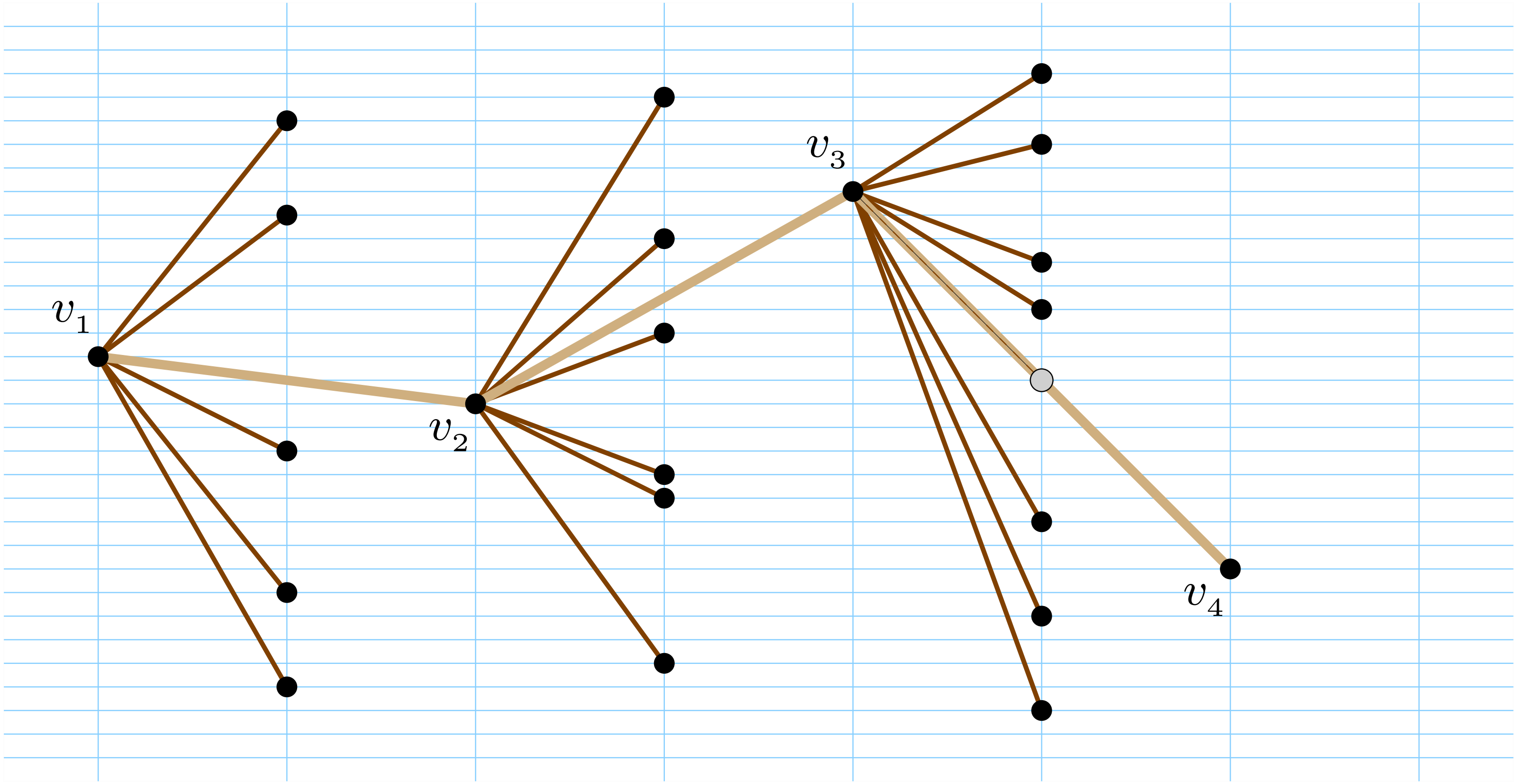
- Drawing a caterpillar in linear time:

Lemma 3 *(Brass et al., 2003) A plane drawing of n-vertex caterpillar $T(V, E)$ with an m-vertex spine can be drawn in $O(n)$ time on a $2m \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



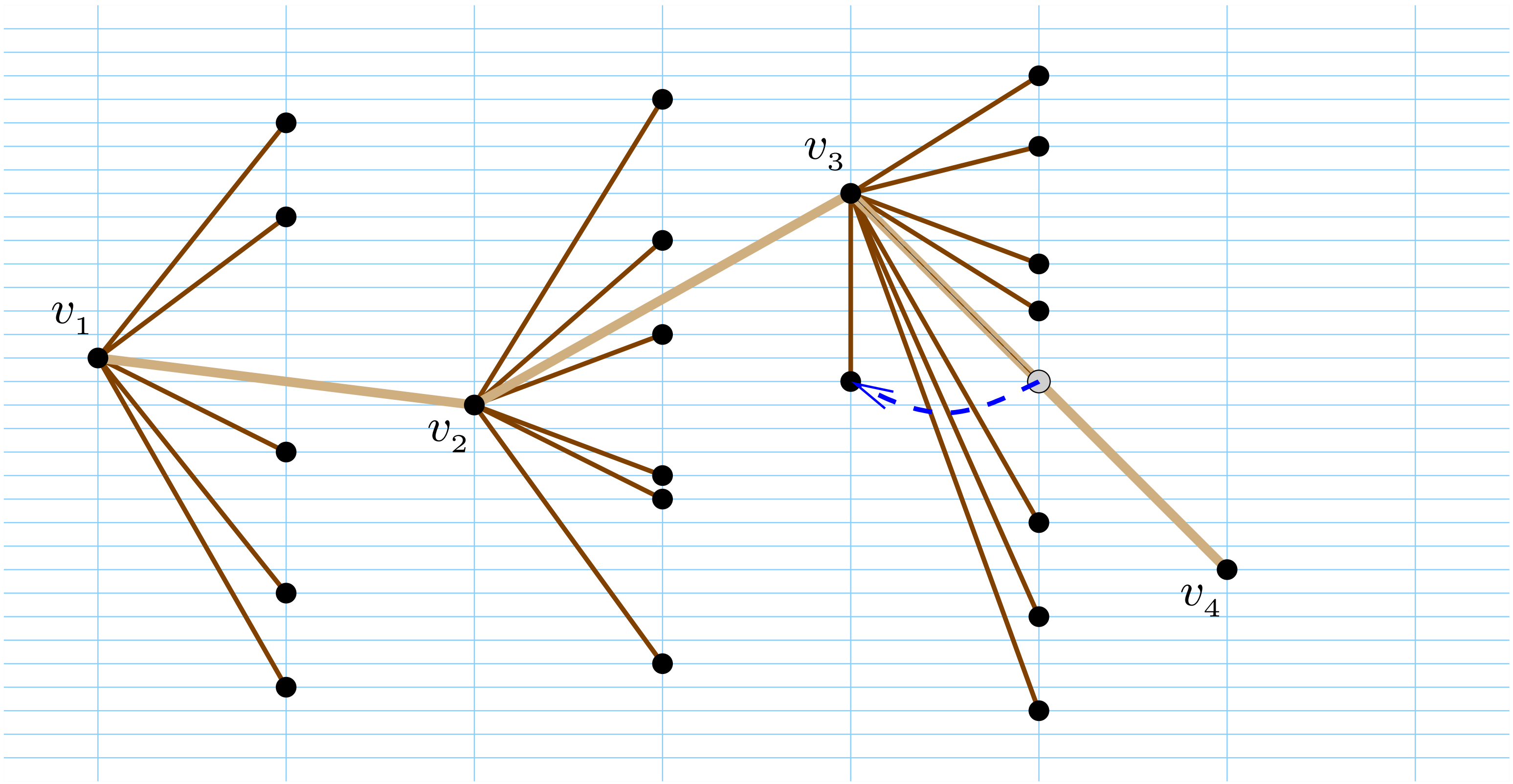
- Drawing a caterpillar in linear time:

Lemma 3 *(Brass et al., 2003) A plane drawing of n-vertex caterpillar $T(V, E)$ with an m-vertex spine can be drawn in $O(n)$ time on a $2m \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



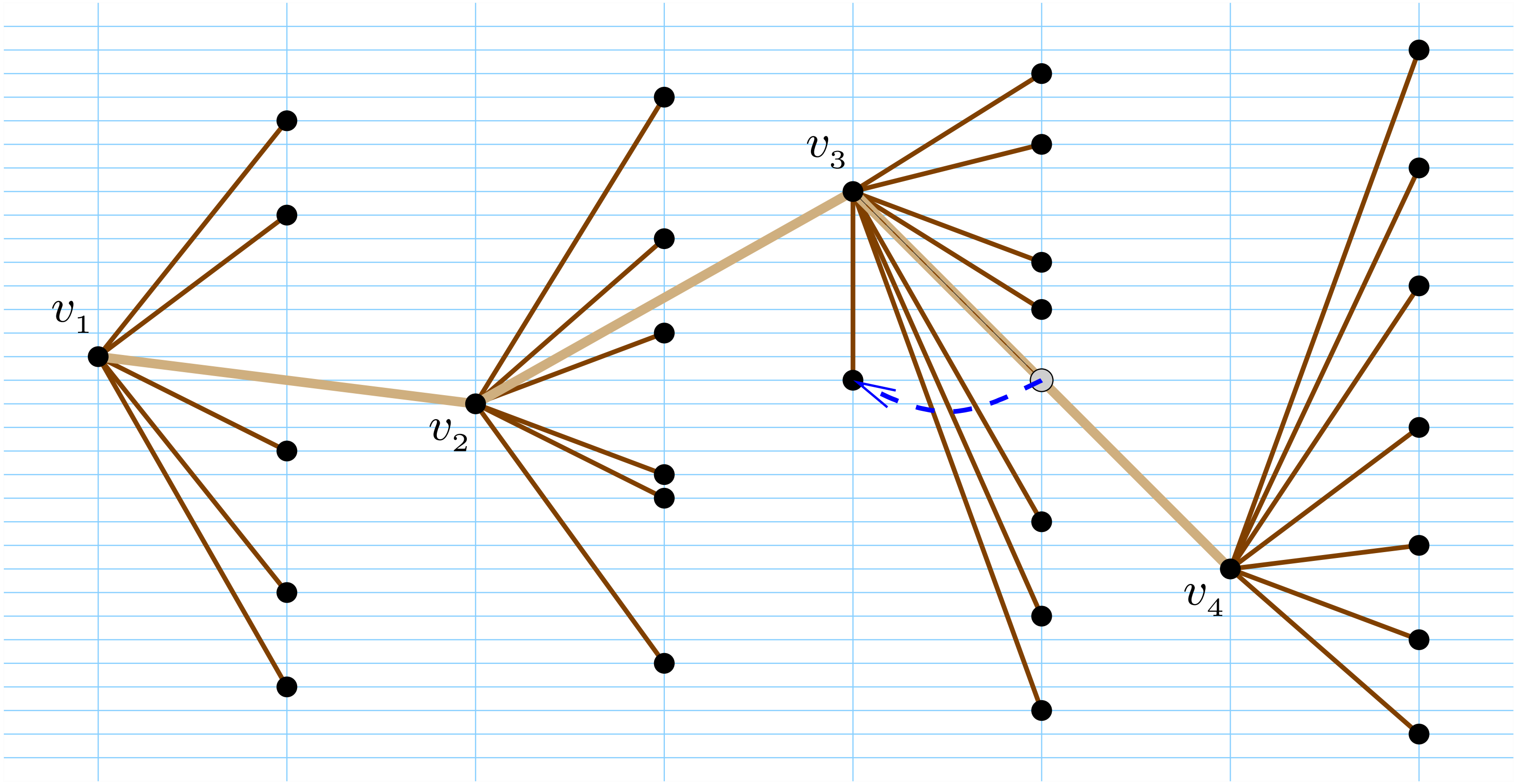
- Drawing a caterpillar in linear time:

Lemma 3 *(Brass et al., 2003) A plane drawing of n-vertex caterpillar $T(V, E)$ with an m-vertex spine can be drawn in $O(n)$ time on a $2m \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



- Drawing a caterpillar in linear time:

Lemma 3 *(Brass et al., 2003) A plane drawing of n-vertex caterpillar $T(V, E)$ with an m-vertex spine can be drawn in $O(n)$ time on a $2m \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



Drawing Algorithms – Radius-2 Stars

- Drawing a radius-2 star in linear time:

Drawing Algorithms – Radius-2 Stars

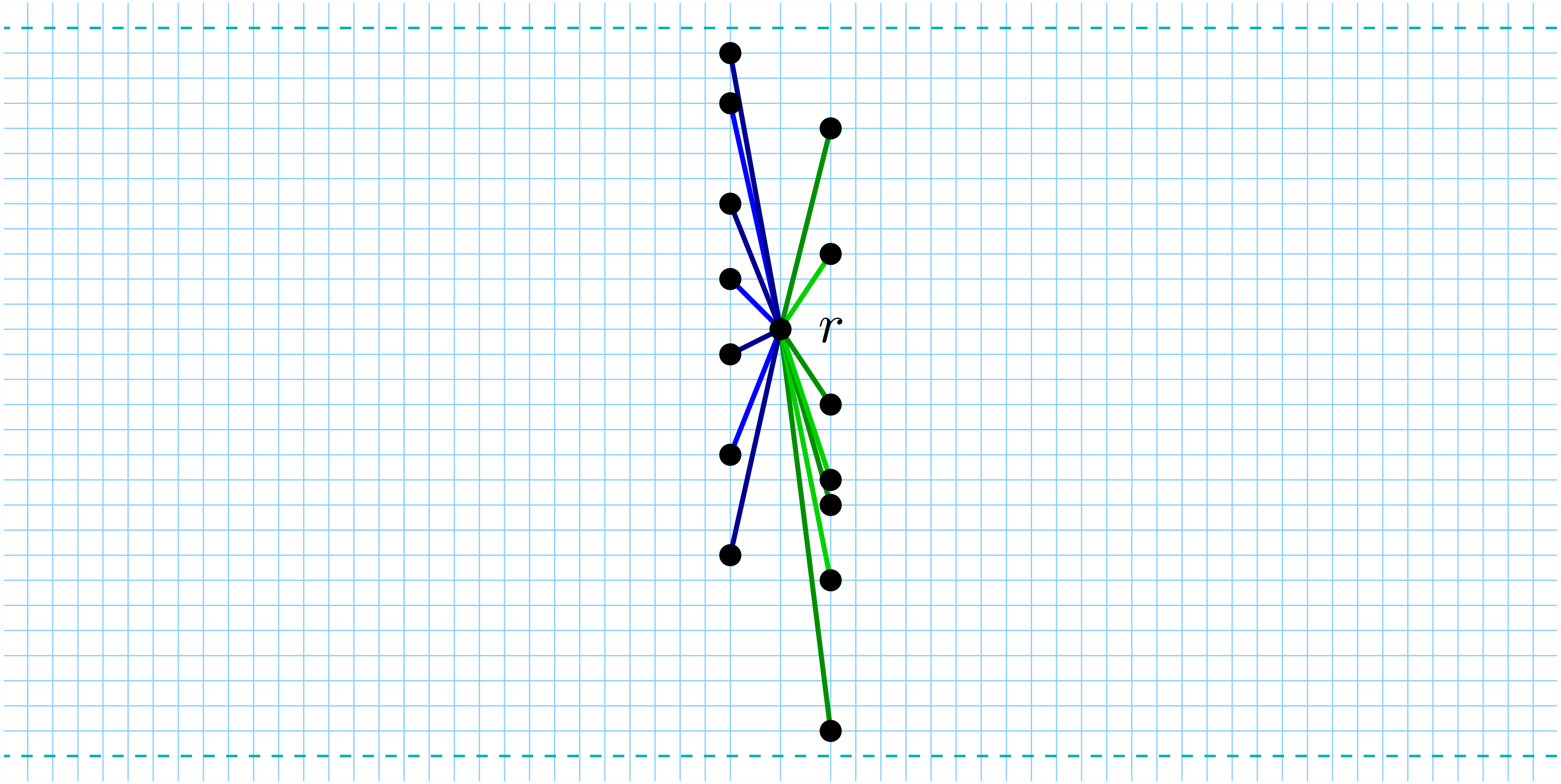
- Drawing a radius-2 star in linear time:

Lemma 4 *A plane drawing of a n-vertex radius-2 star $T(V, E)$ can be drawn in $O(n)$ time on a $(2n+1) \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

Drawing Algorithms – Radius-2 Stars

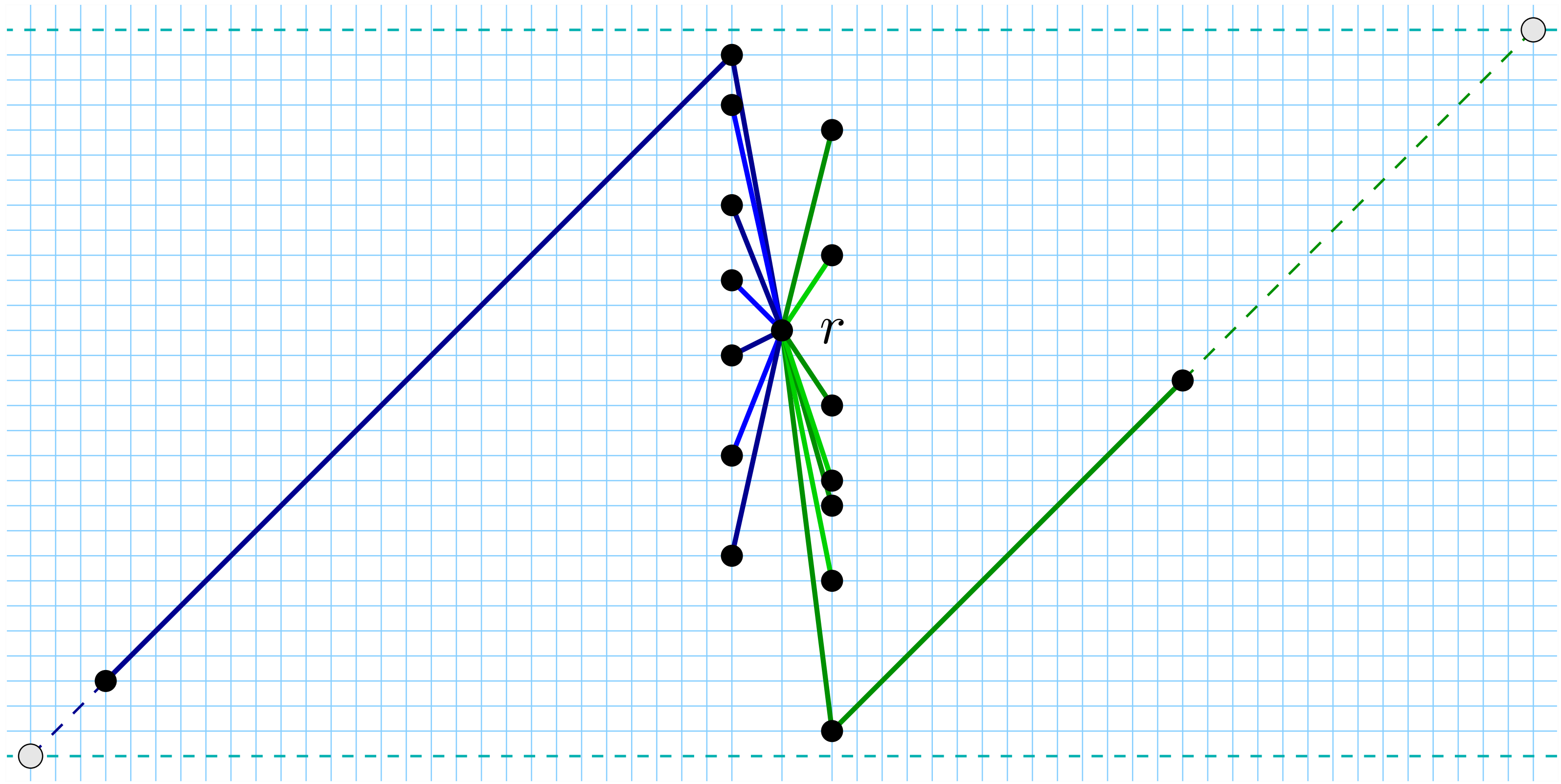
- Drawing a radius-2 star in linear time:

Lemma 4 *A plane drawing of a n-vertex radius-2 star $T(V, E)$ can be drawn in $O(n)$ time on a $(2n+1) \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



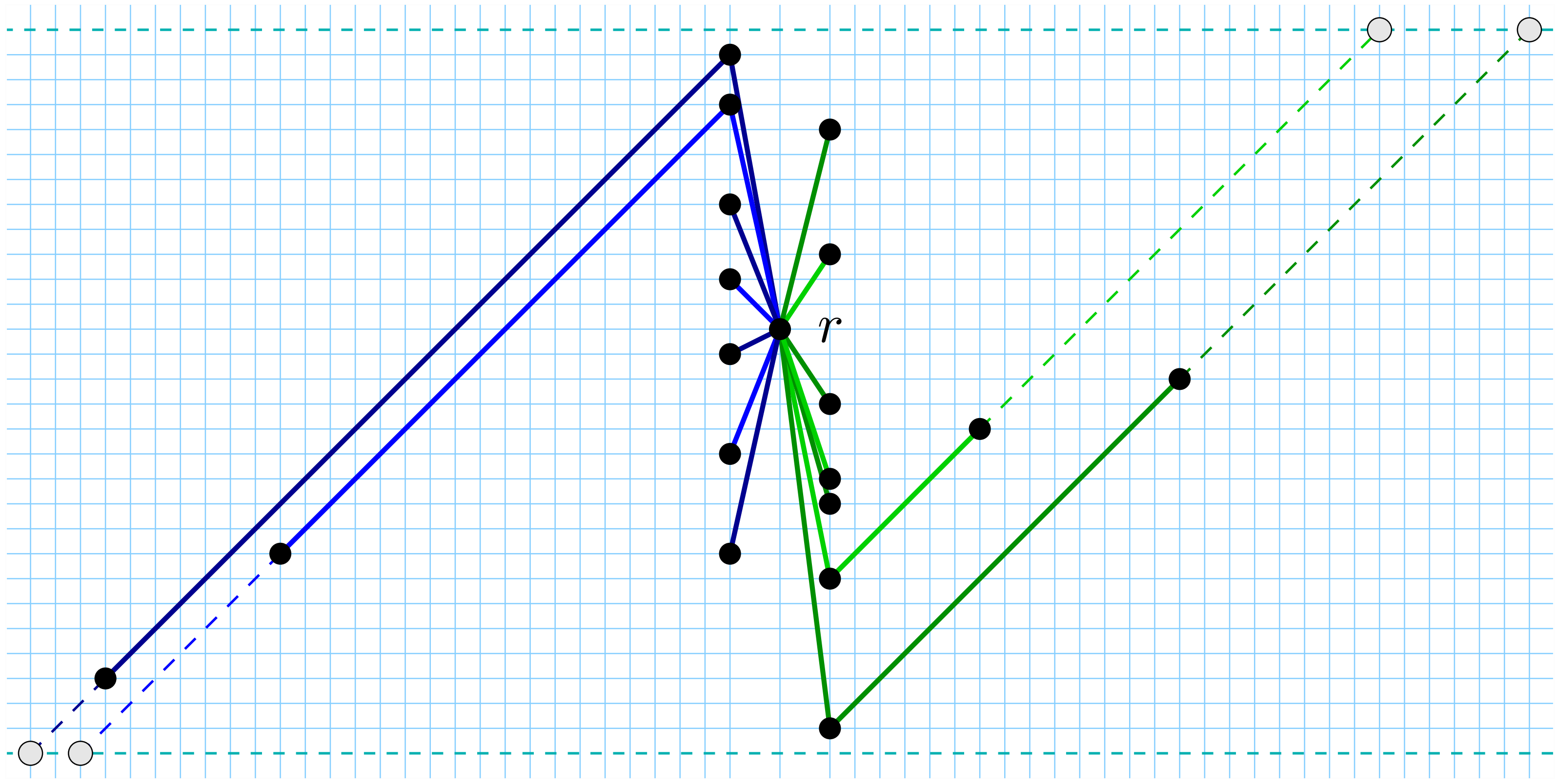
- Drawing a radius-2 star in linear time:

Lemma 4 *A plane drawing of a n-vertex radius-2 star $T(V, E)$ can be drawn in $O(n)$ time on a $(2n+1) \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



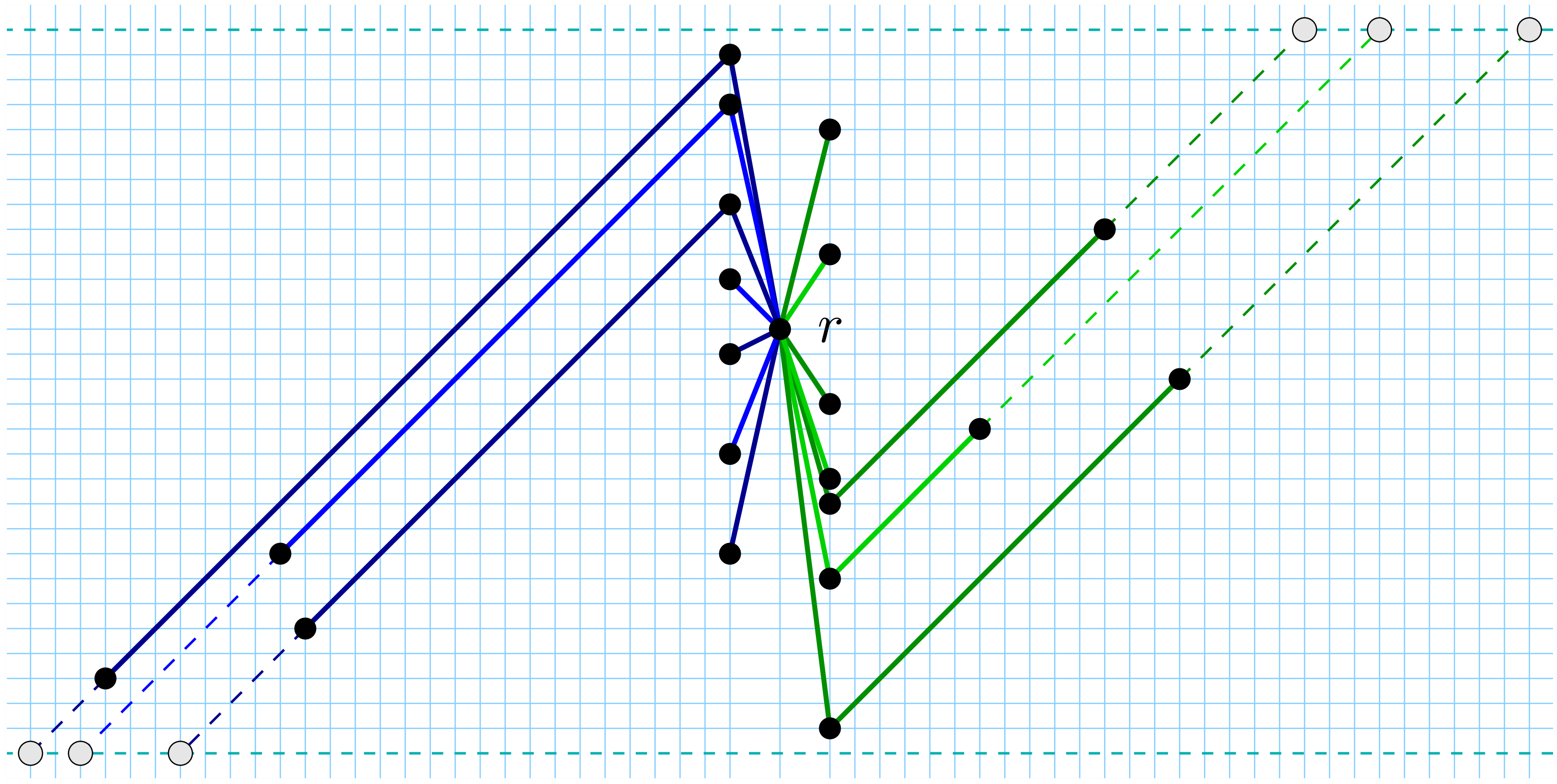
- Drawing a radius-2 star in linear time:

Lemma 4 *A plane drawing of a n-vertex radius-2 star $T(V, E)$ can be drawn in $O(n)$ time on a $(2n+1) \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



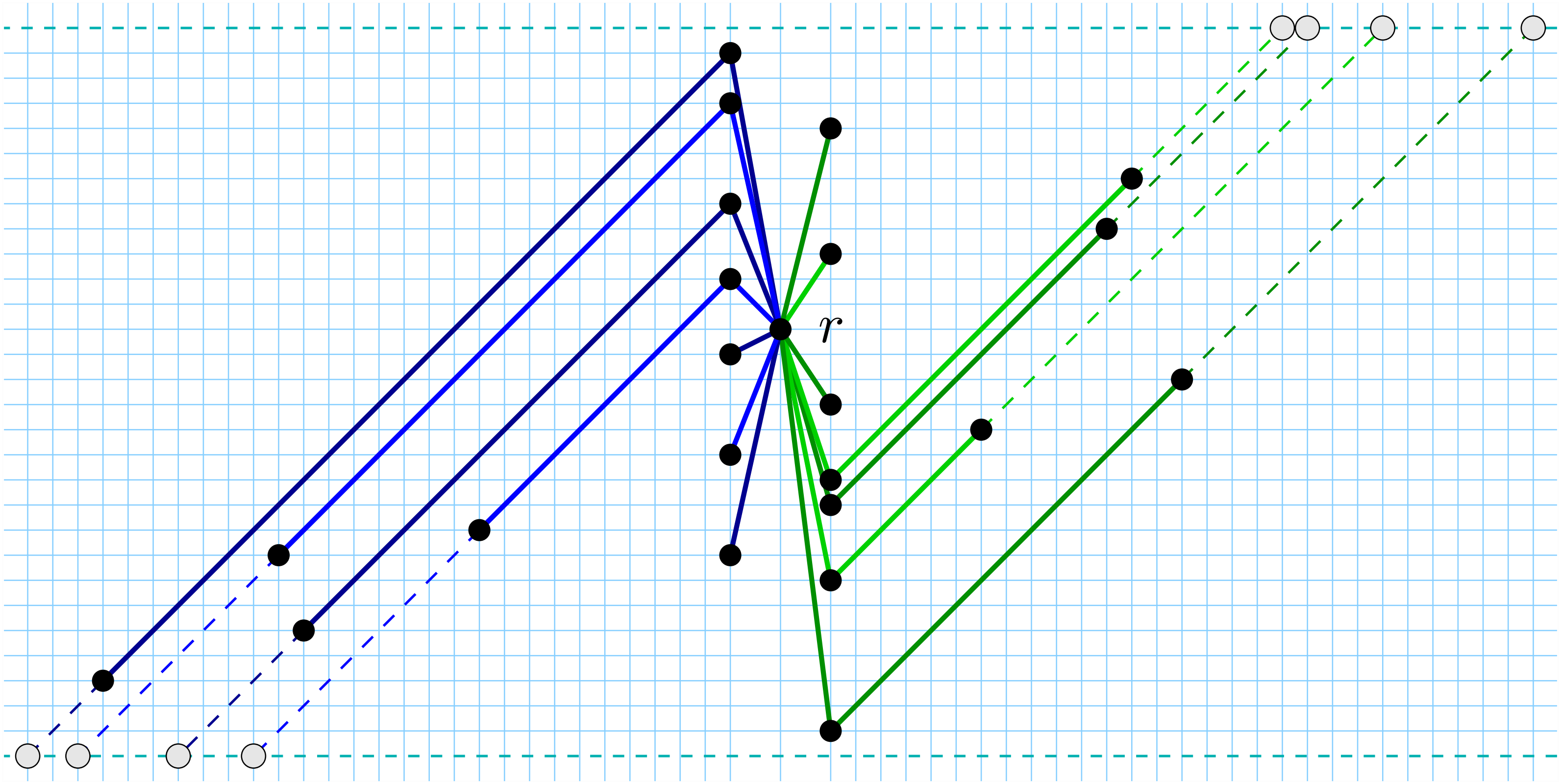
- Drawing a radius-2 star in linear time:

Lemma 4 *A plane drawing of a n-vertex radius-2 star $T(V, E)$ can be drawn in $O(n)$ time on a $(2n+1) \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



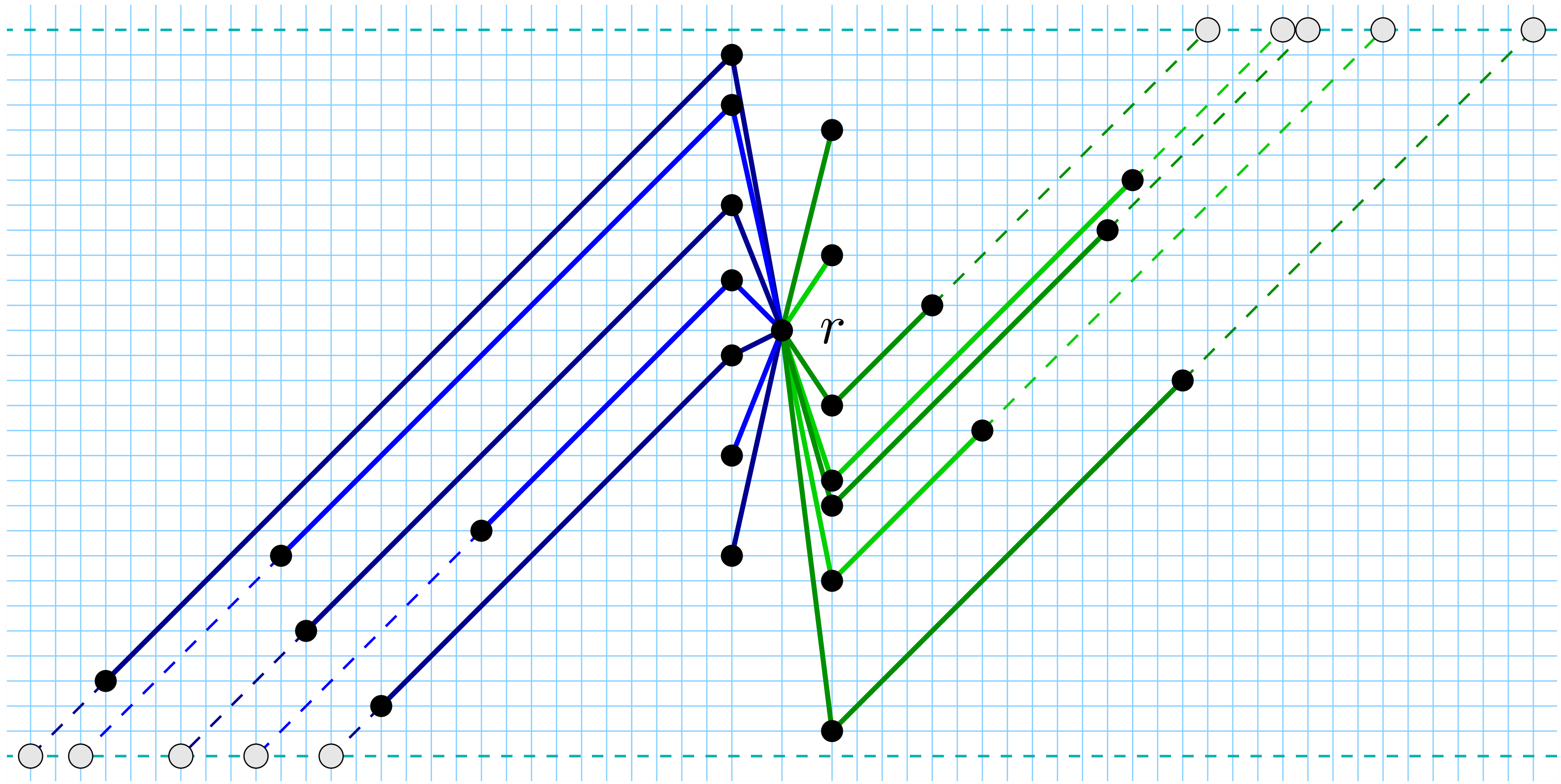
- Drawing a radius-2 star in linear time:

Lemma 4 *A plane drawing of a n-vertex radius-2 star $T(V, E)$ can be drawn in $O(n)$ time on a $(2n + 1) \times n$ grid for any leveling $\phi : V \xrightarrow[onto]{1:1} \{1, 2, \ldots, n\}$.*



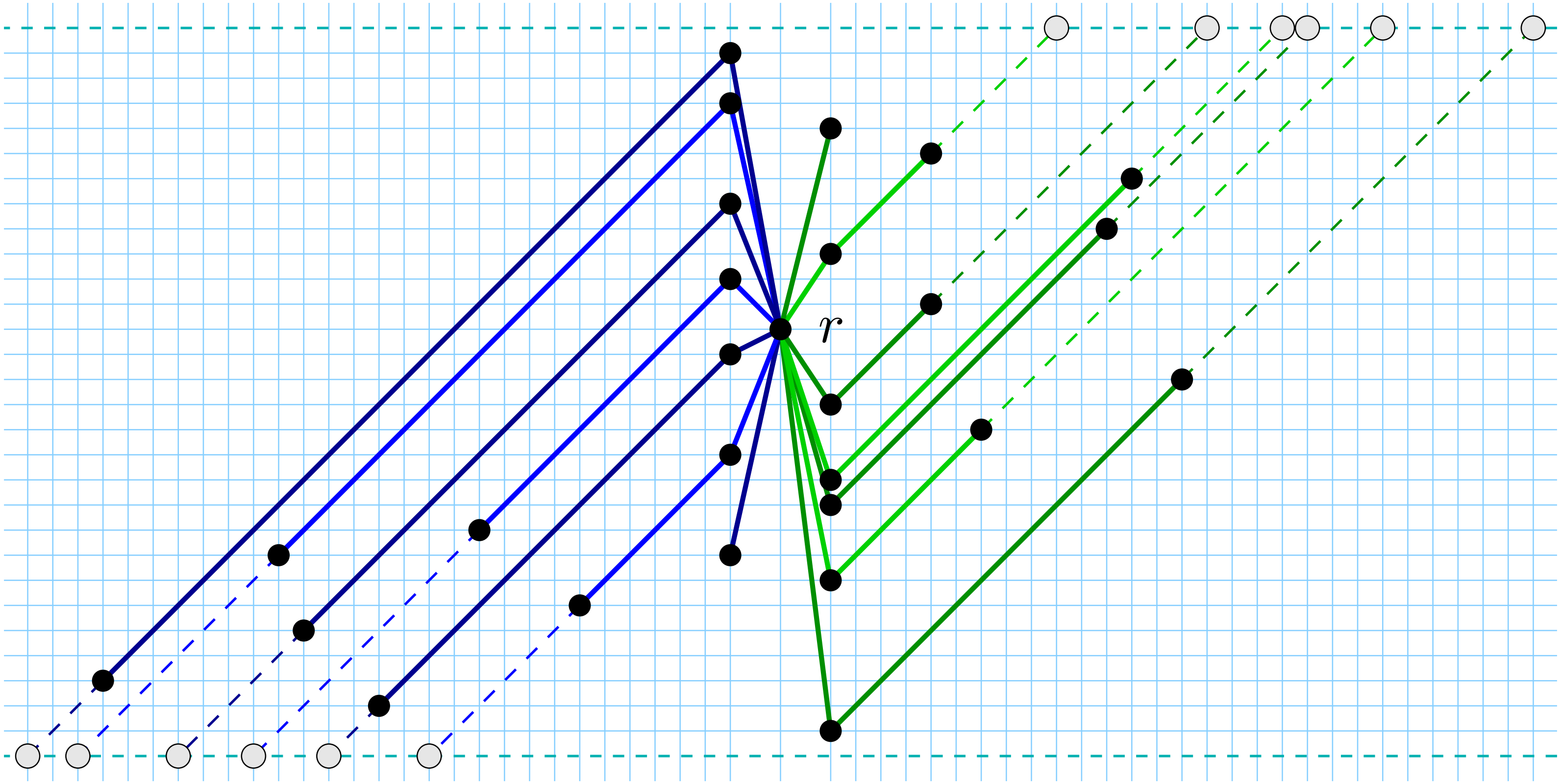
- Drawing a radius-2 star in linear time:

Lemma 4 *A plane drawing of a n-vertex radius-2 star $T(V, E)$ can be drawn in $O(n)$ time on a $(2n+1) \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



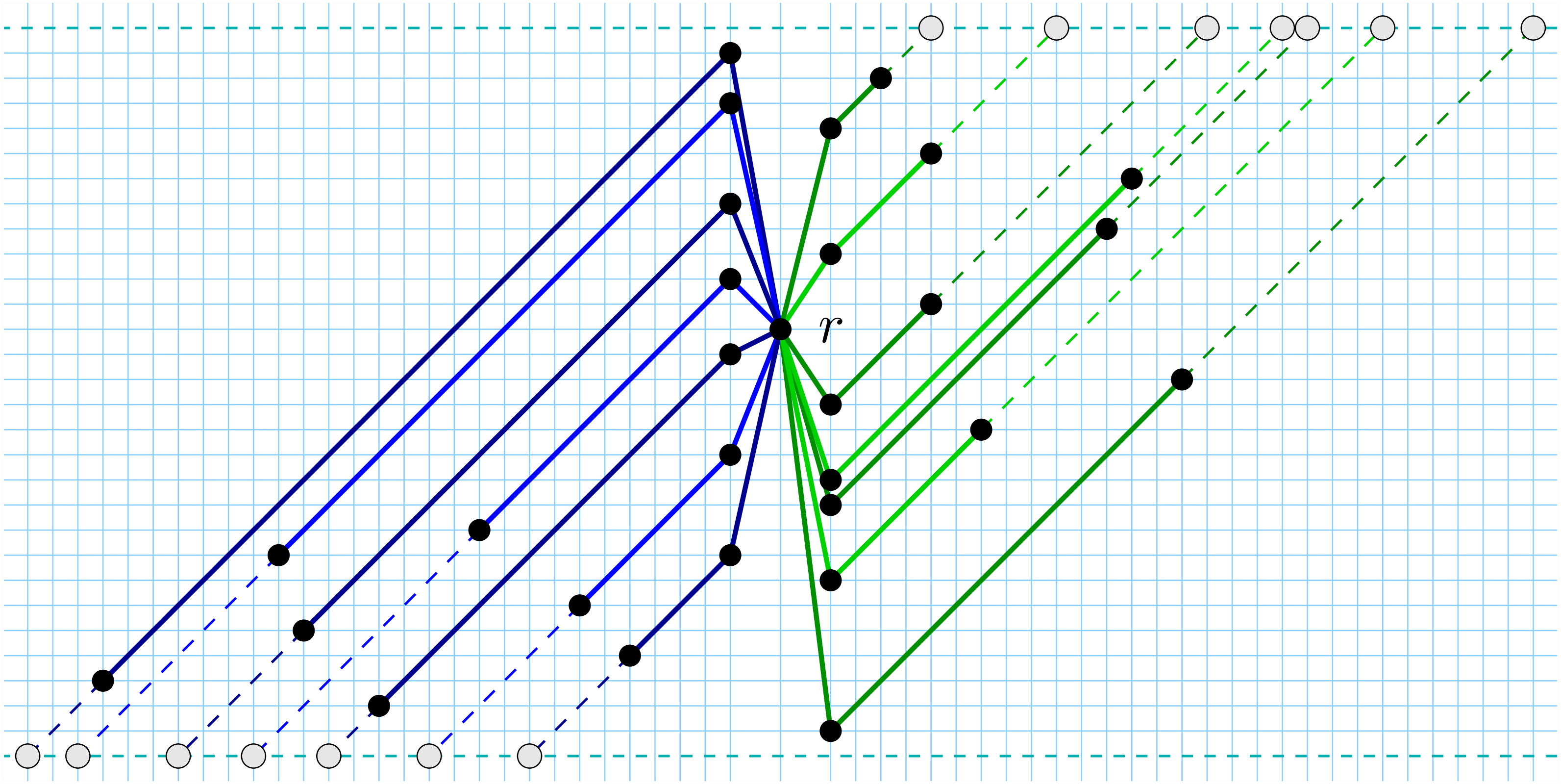
- Drawing a radius-2 star in linear time:

Lemma 4 *A plane drawing of a n-vertex radius-2 star $T(V, E)$ can be drawn in $O(n)$ time on a $(2n+1) \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



- Drawing a radius-2 star in linear time:

Lemma 4 *A plane drawing of a n-vertex radius-2 star $T(V, E)$ can be drawn in $O(n)$ time on a $(2n+1) \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



Drawing Algorithms – Degree-3 Spiders

- Drawing a degree-3 spider in linear time:

Drawing Algorithms – Degree-3 Spiders

- Drawing a degree-3 spider in linear time:

Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

Drawing Algorithms – Degree-3 Spiders

- Drawing a degree-3 spider in linear time:

  Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

- Proof idea:
  - Want to main two invariants

Drawing Algorithms – Degree-3 Spiders

- Drawing a degree-3 spider in linear time:

  Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

- Proof idea:
  - Want to main two invariants
    1. Two leaves of tree drawn so far are extreme

Drawing Algorithms – Degree-3 Spiders

- Drawing a degree-3 spider in linear time:

  Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

- Proof idea:
  - Want to main two invariants
    1. Two leaves of tree drawn so far are extreme
    2. Other leaf has a way to proceed left or right

Drawing Algorithms – Degree-3 Spiders

- Drawing a degree-3 spider in linear time:

  Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

- Proof idea:
  - Want to main two invariants
    1. Two leaves of tree drawn so far are extreme
    2. Other leaf has a way to proceed left or right
  - Choose the non-extreme leaf and extend it until it becomes extreme, repeat

Drawing Algorithms – Degree-3 Spiders

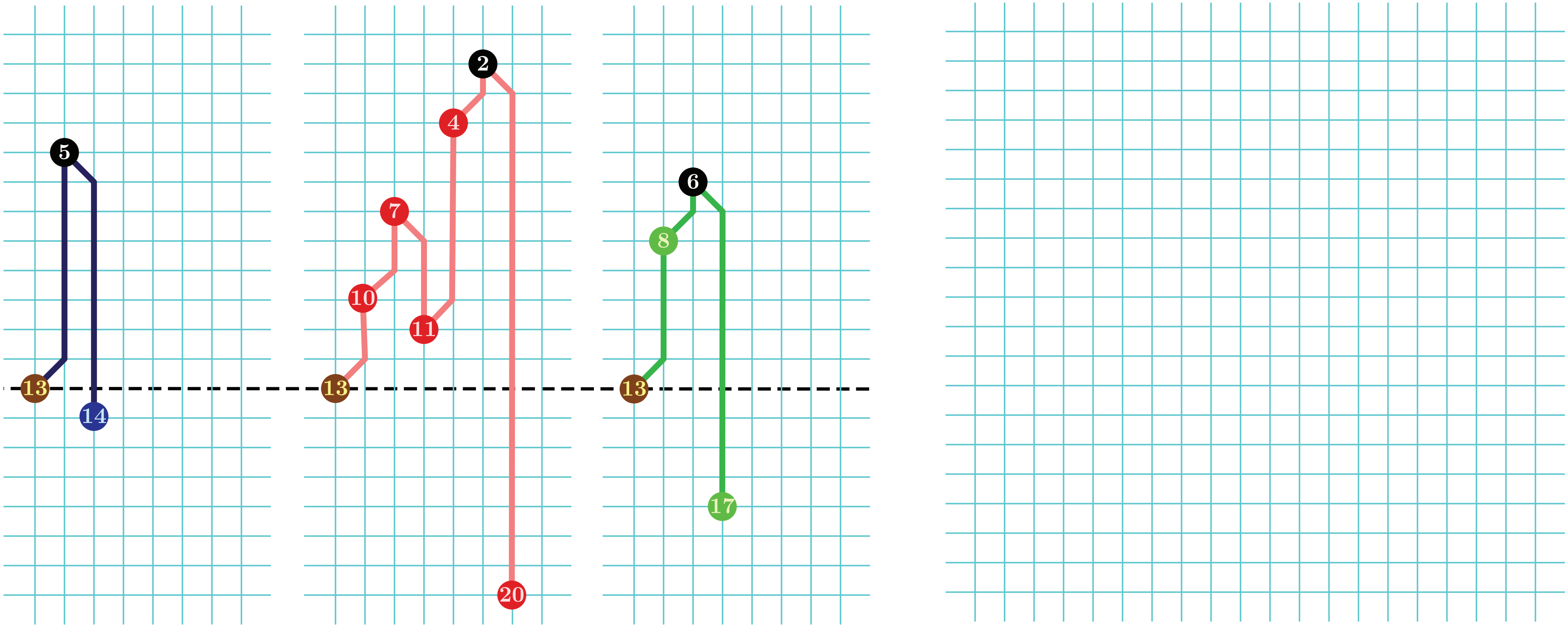
- Drawing a degree-3 spider in linear time:

  Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

- Proof idea:
  - Want to main two invariants
    1. Two leaves of tree drawn so far are extreme
    2. Other leaf has a way to proceed left or right
  - Choose the non-extreme leaf and extend it until it becomes extreme, repeat
  - Tricky part is getting degree-3 spider started

- Drawing a degree-3 spider in linear time:

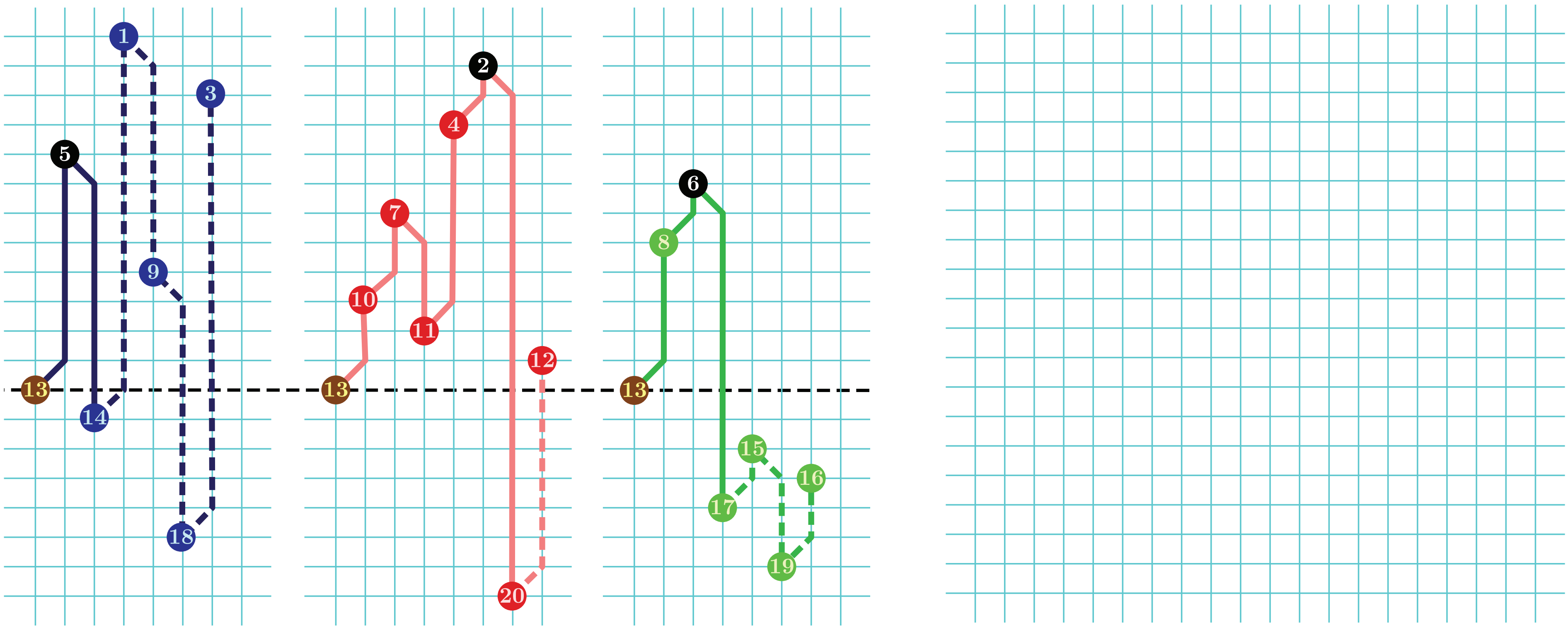
Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



Drawing Algorithms – Degree-3 Spiders

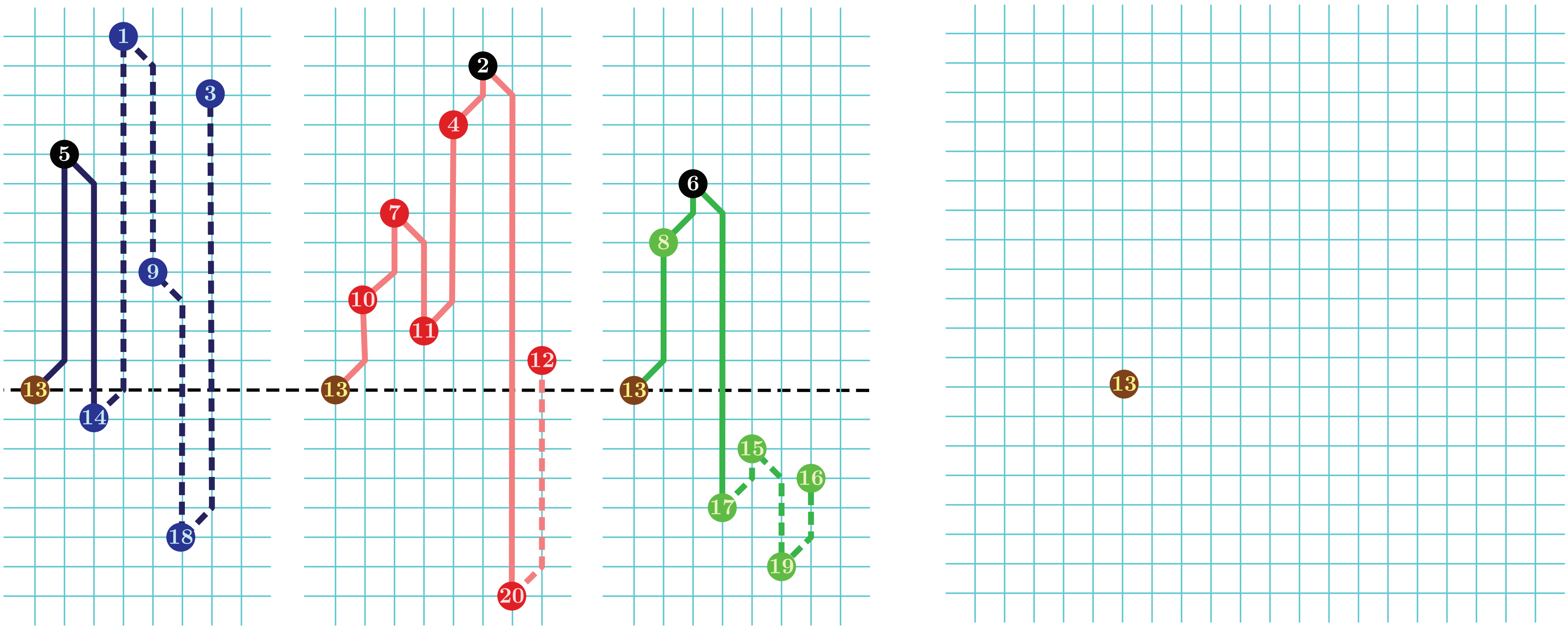
- Drawing a degree-3 spider in linear time:

  Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



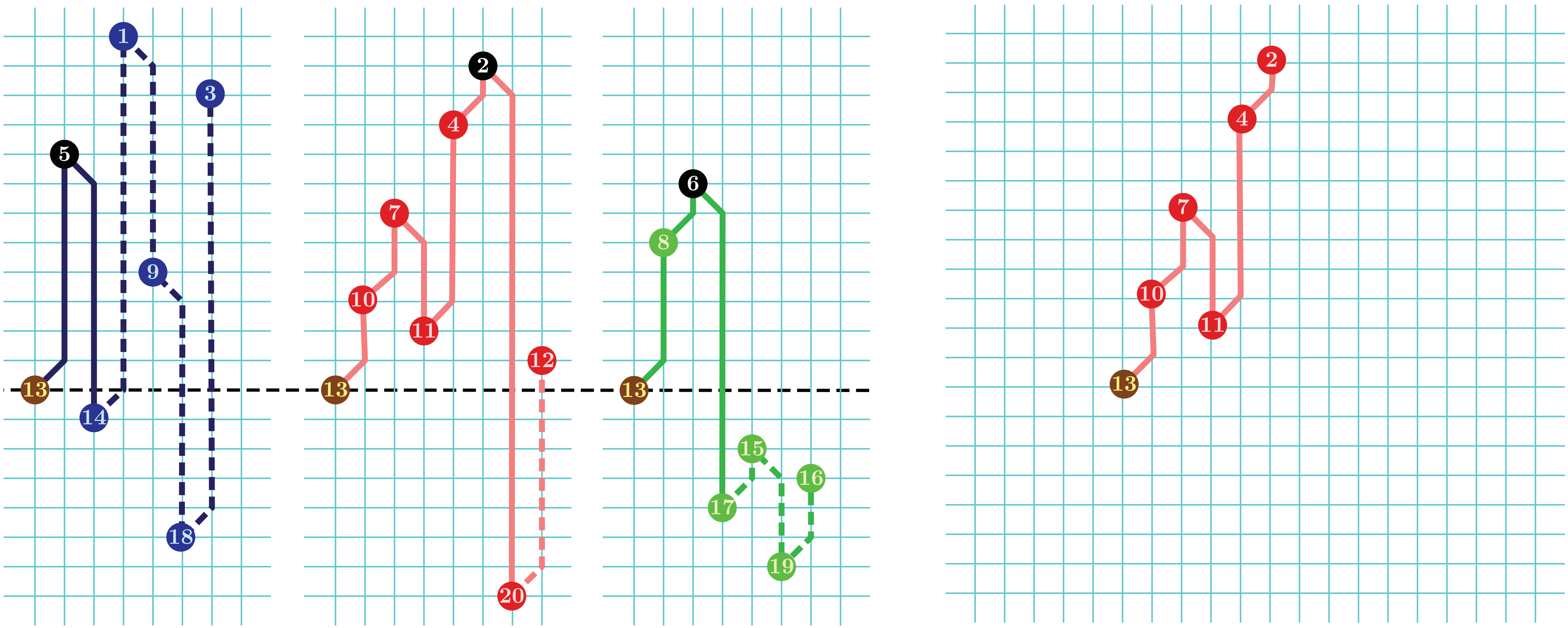
- Drawing a degree-3 spider in linear time:

Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



- Drawing a degree-3 spider in linear time:

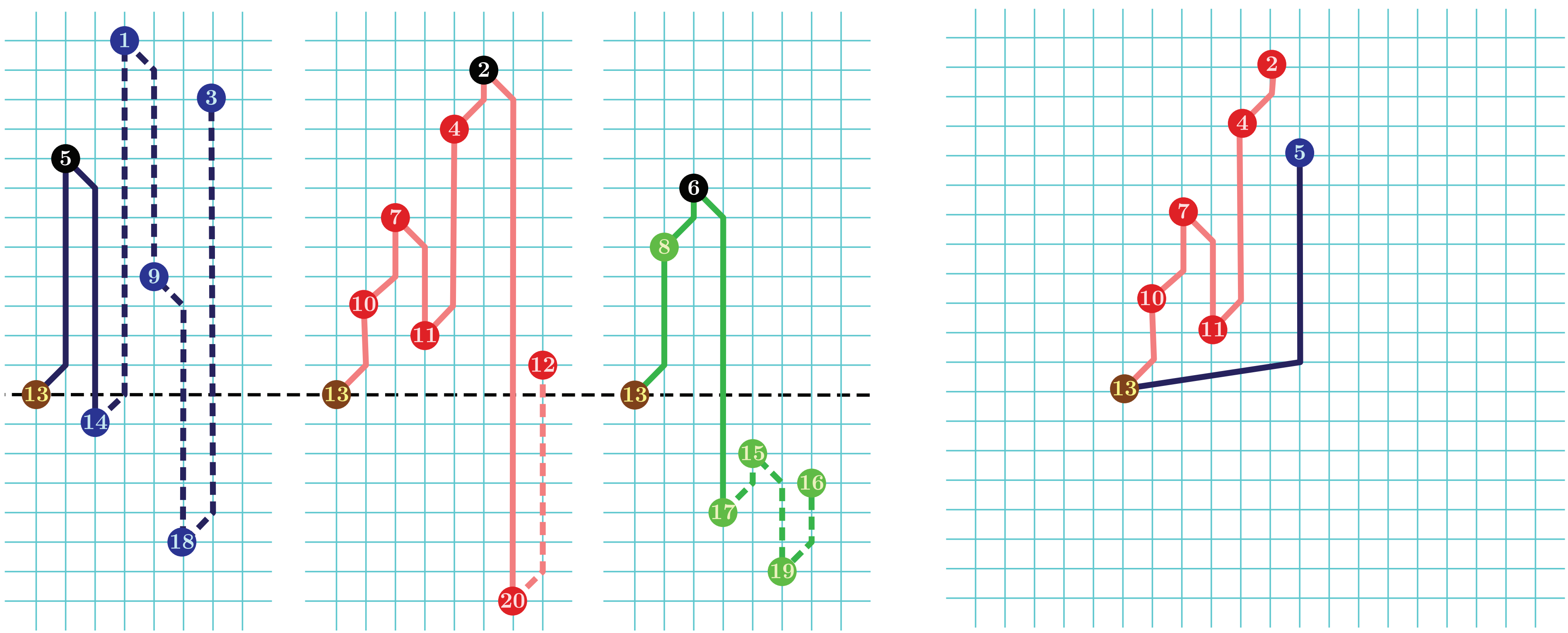
Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



Drawing Algorithms – Degree-3 Spiders

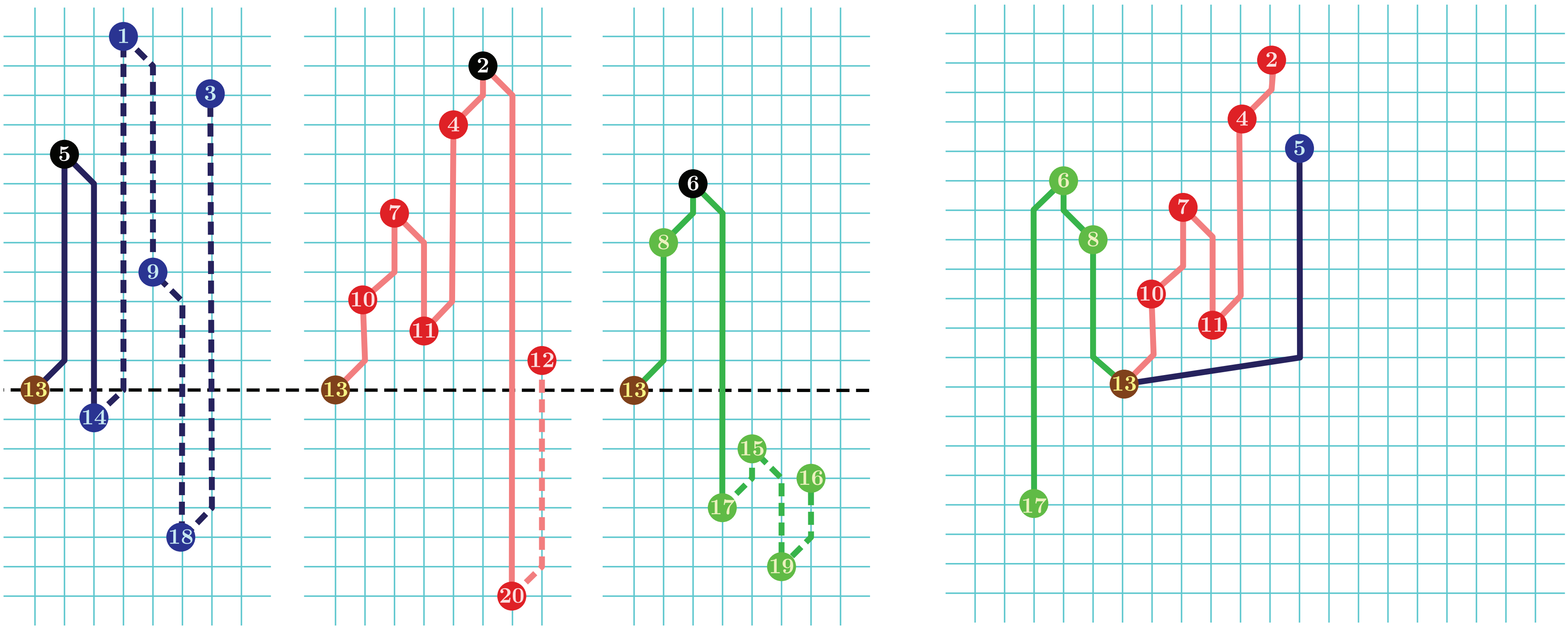
- Drawing a degree-3 spider in linear time:

Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



- Drawing a degree-3 spider in linear time:

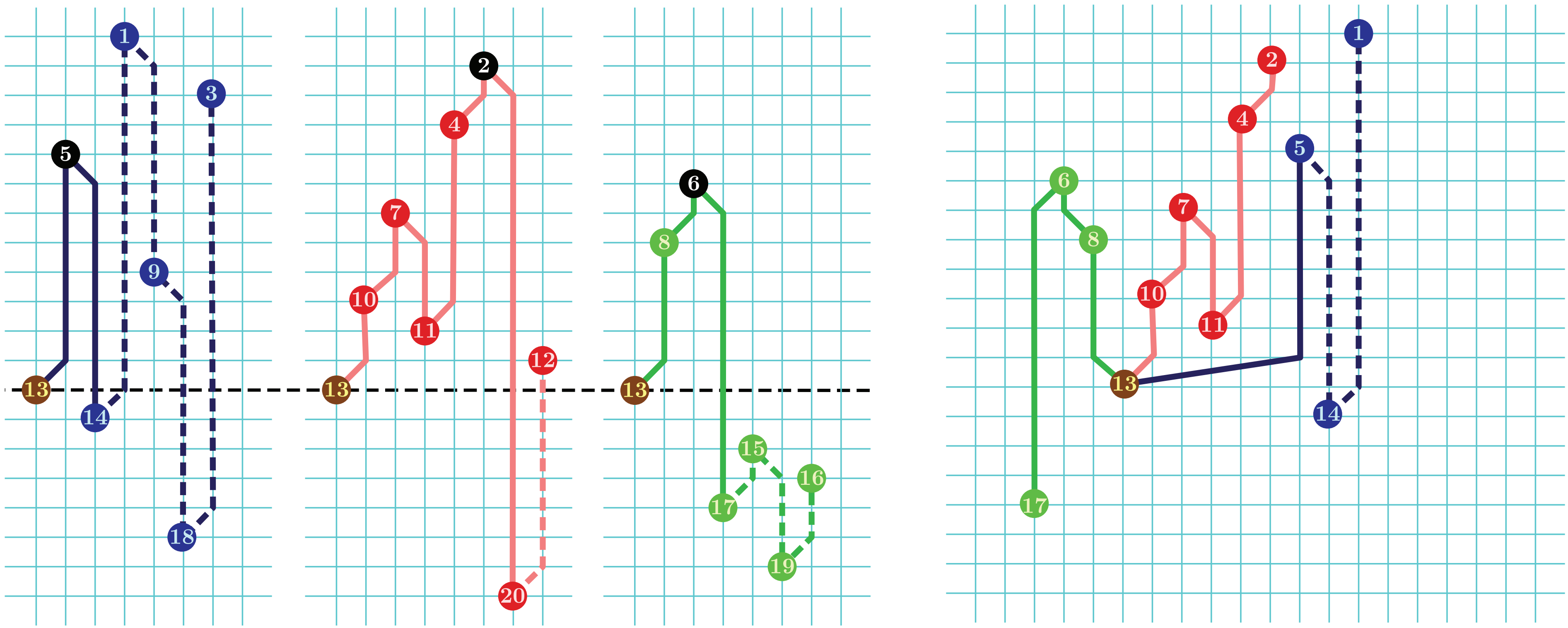
Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V,E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



Drawing Algorithms – Degree-3 Spiders

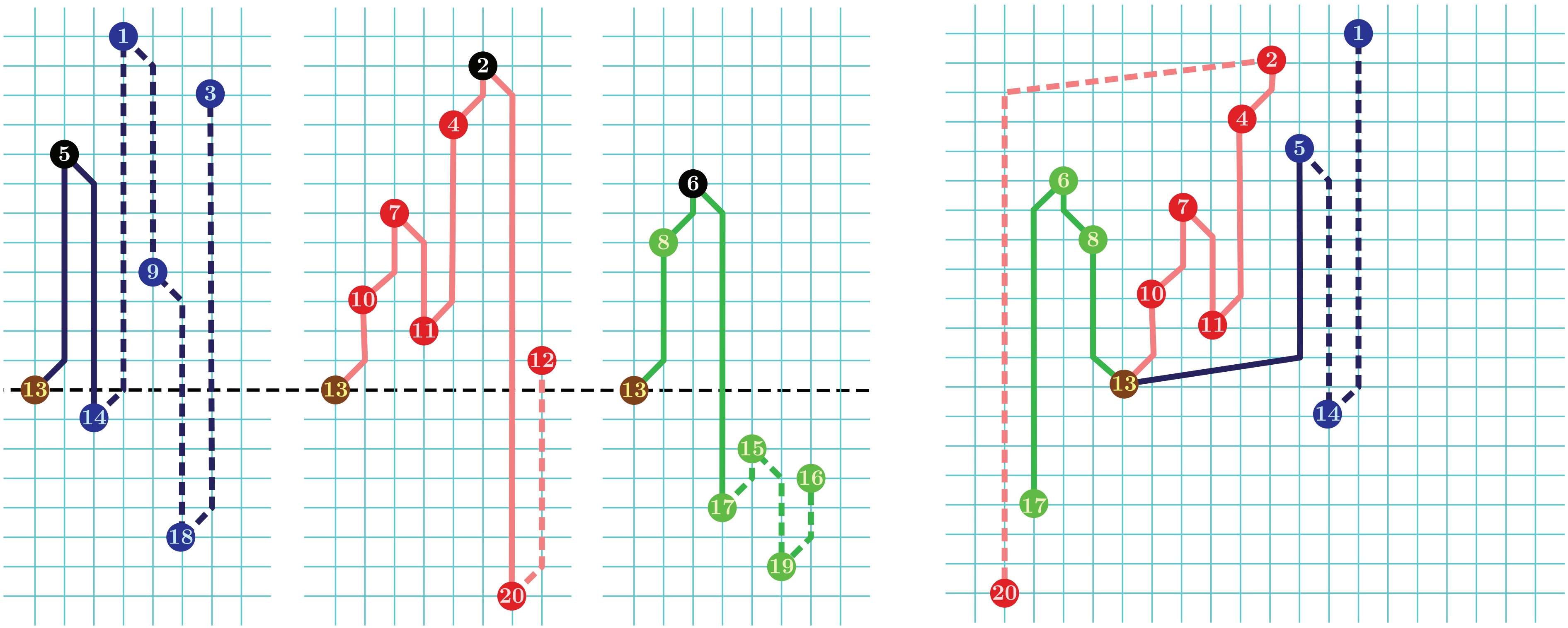
- Drawing a degree-3 spider in linear time:

Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



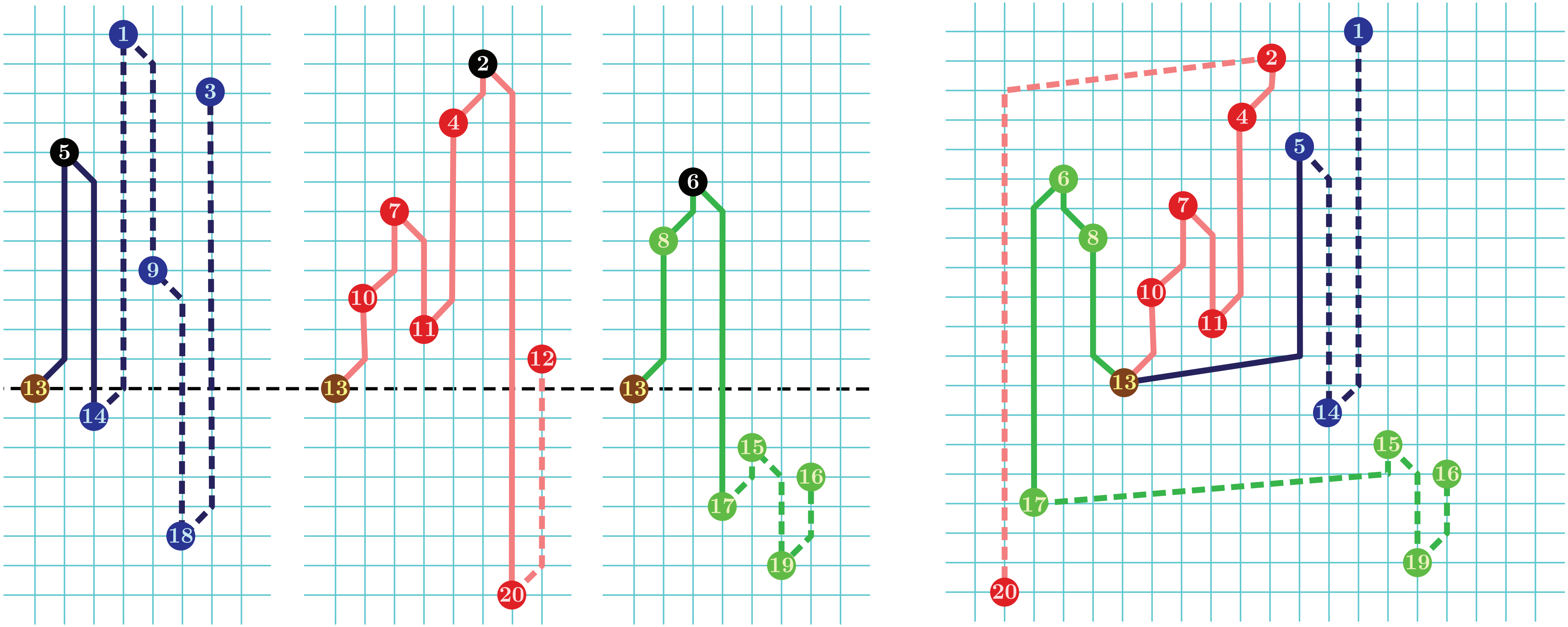
- Drawing a degree-3 spider in linear time:

Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



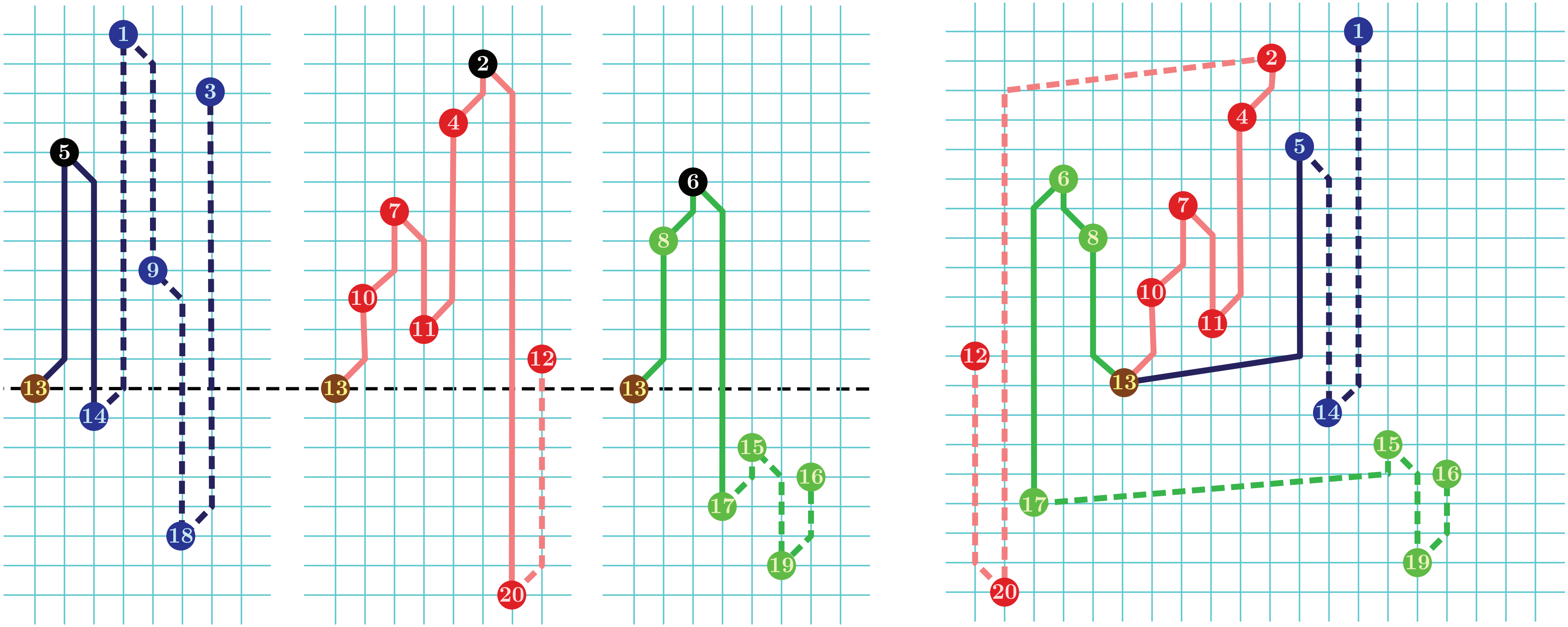
- Drawing a degree-3 spider in linear time:

  Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



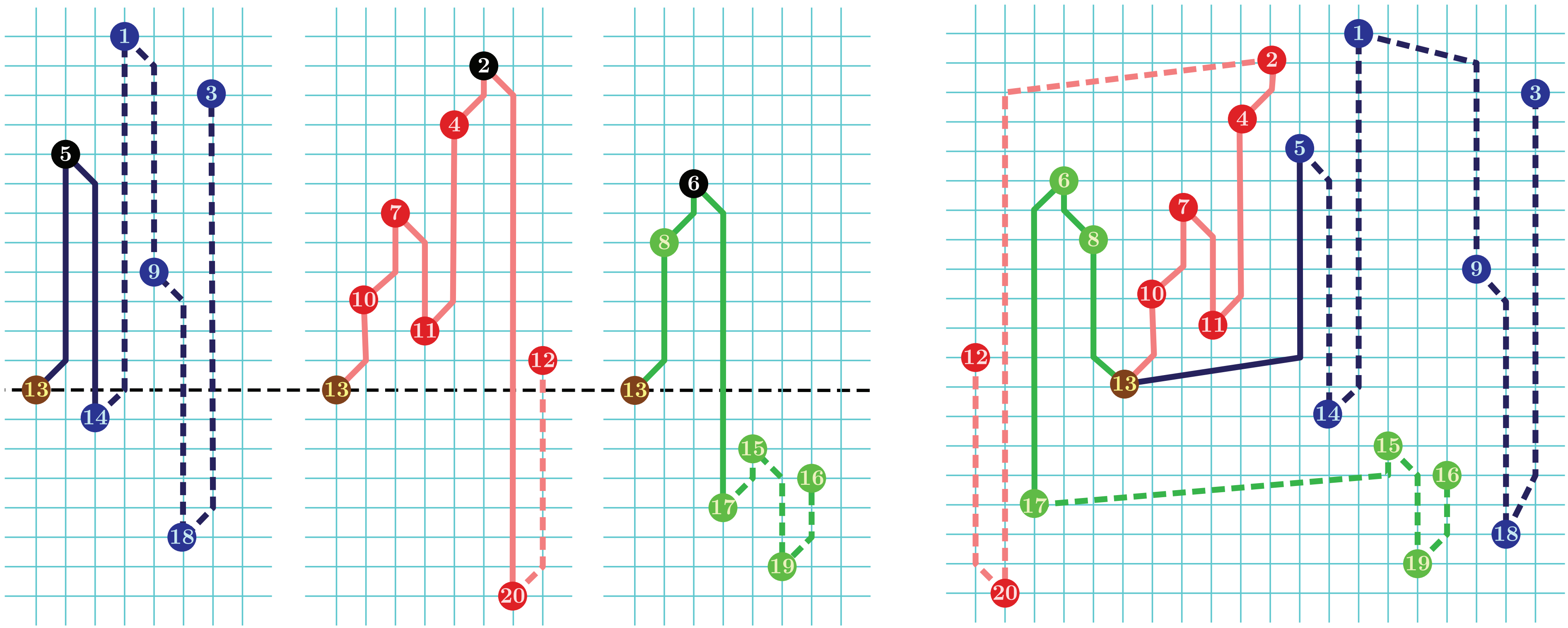
- Drawing a degree-3 spider in linear time:

Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



- Drawing a degree-3 spider in linear time:

Lemma 5 *A planar drawing of an n-vertex degree-3 spider $T(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



Characterization – ULP Trees

- A minimal lobster argument gives the next theorem:

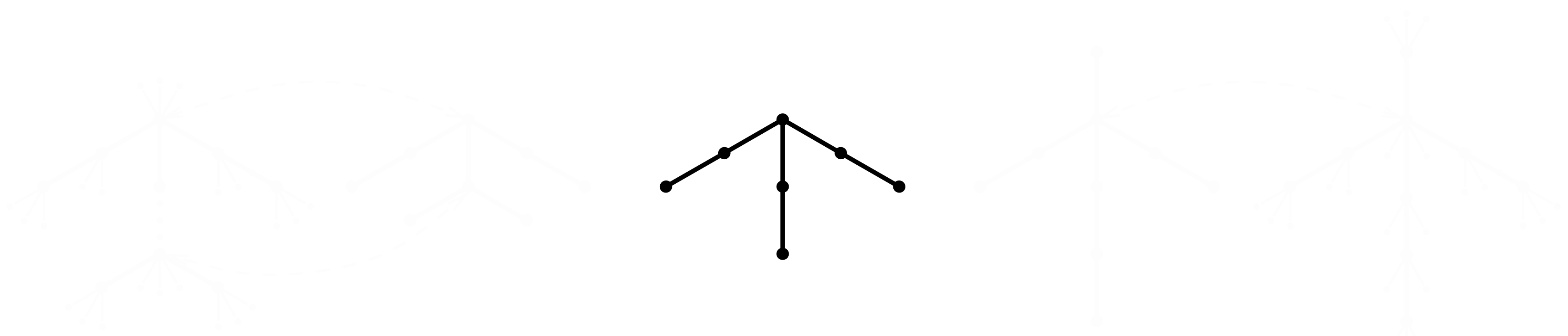
  Theorem 6 *Every tree either contains a subdivision of* T_8 *or* T_9 *in which case it is not ULP, or it is a caterpillar, a radius-*2 *star, or a degree-*3 *spider in which case it is ULP.*

Characterization – ULP Trees

- A minimal lobster argument gives the next theorem:

  Theorem 6 *Every tree either contains a subdivision of T_8 or T_9 in which case it is not ULP, or it is a caterpillar, a radius-2 star, or a degree-3 spider in which case it is ULP.*

- Proof idea:
  - A tree that is not a caterpillar has a minimal lobster T_7



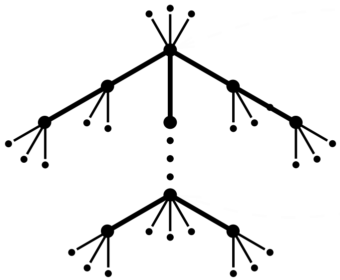
T_7

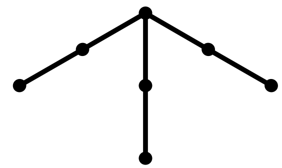
- A minimal lobster argument gives the next theorem:

Theorem 6 *Every tree either contains a subdivision of T_8 or T_9 in which case it is not ULP, or it is a caterpillar, a radius-2 star, or a degree-3 spider in which case it is ULP.*

- Proof idea:
  - A tree that is not a degree-3 spider or radius-2 star has two cases

Characterization – ULP Trees





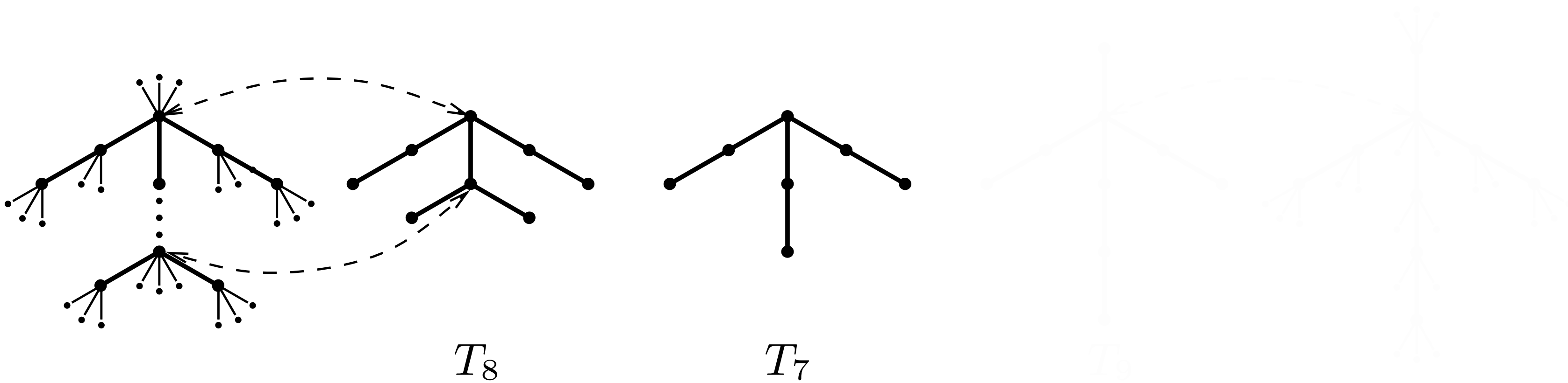
T_7

- A minimal lobster argument gives the next theorem:

  Theorem 6 *Every tree either contains a subdivision of T_8 or T_9 in which case it is not ULP, or it is a caterpillar, a radius-2 star, or a degree-3 spider in which case it is ULP.*

- Proof idea:
  - A tree that is not a degree-3 spider or radius-2 star has two cases
    - Has at least two vertices of degree-3

Characterization – ULP Trees

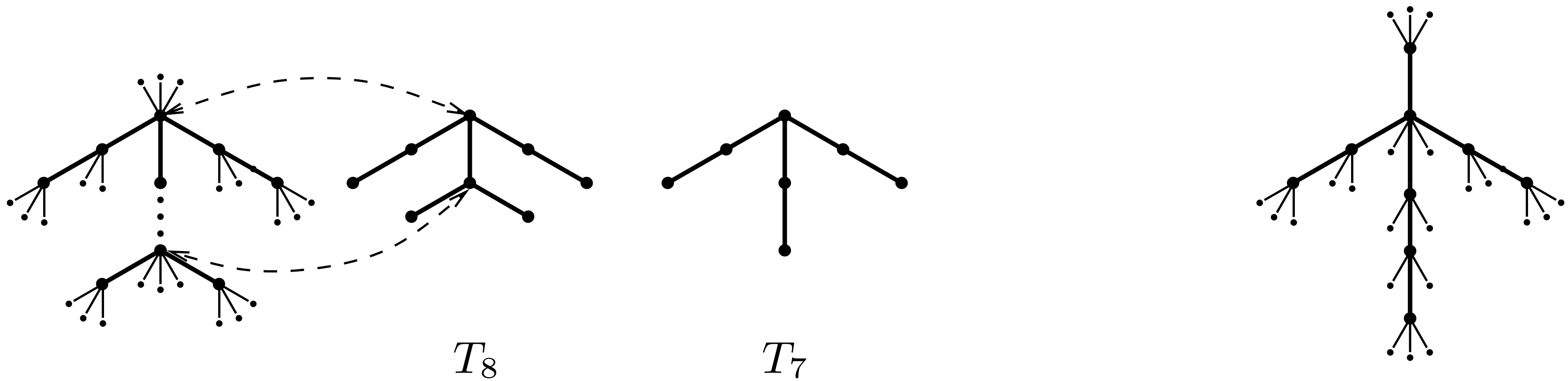


- A minimal lobster argument gives the next theorem:

  Theorem 6 *Every tree either contains a subdivision of T_8 or T_9 in which case it is not ULP, or it is a caterpillar, a radius-2 star, or a degree-3 spider in which case it is ULP.*

- Proof idea:
  - A tree that is not a degree-3 spider or radius-2 star has two cases
    - Has at least two vertices of degree-3 – contains T_8 subdivision

Characterization – ULP Trees

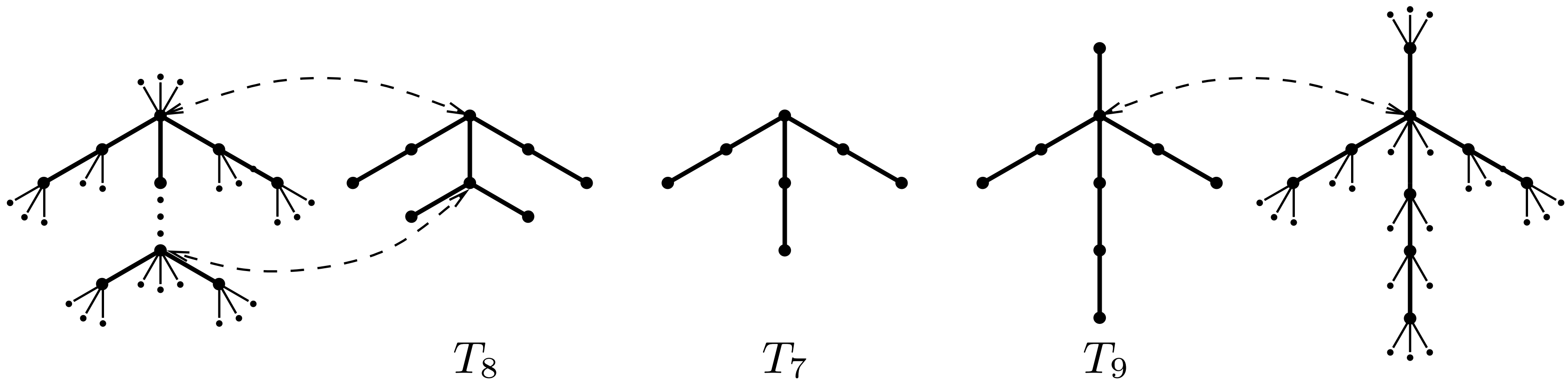


- A minimal lobster argument gives the next theorem:

Theorem 6 *Every tree either contains a subdivision of T_8 or T_9 in which case it is not ULP, or it is a caterpillar, a radius-2 star, or a degree-3 spider in which case it is ULP.*

- Proof idea:
  - A tree that is not a degree-3 spider or radius-2 star has two cases
    - Has at least one vertex of degree-4

Characterization – ULP Trees



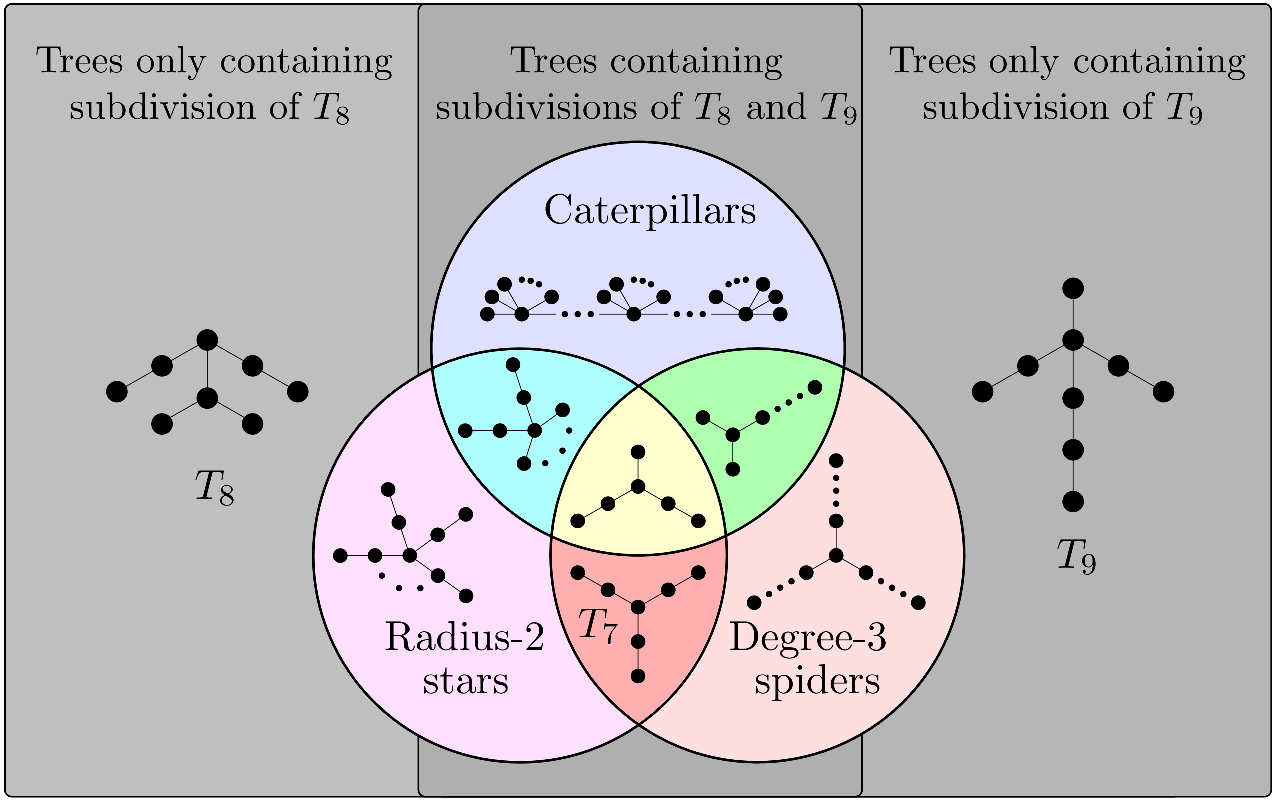
- A minimal lobster argument gives the next theorem:

  Theorem 6 *Every tree either contains a subdivision of T_8 or T_9 in which case it is not ULP, or it is a caterpillar, a radius-2 star, or a degree-3 spider in which case it is ULP.*

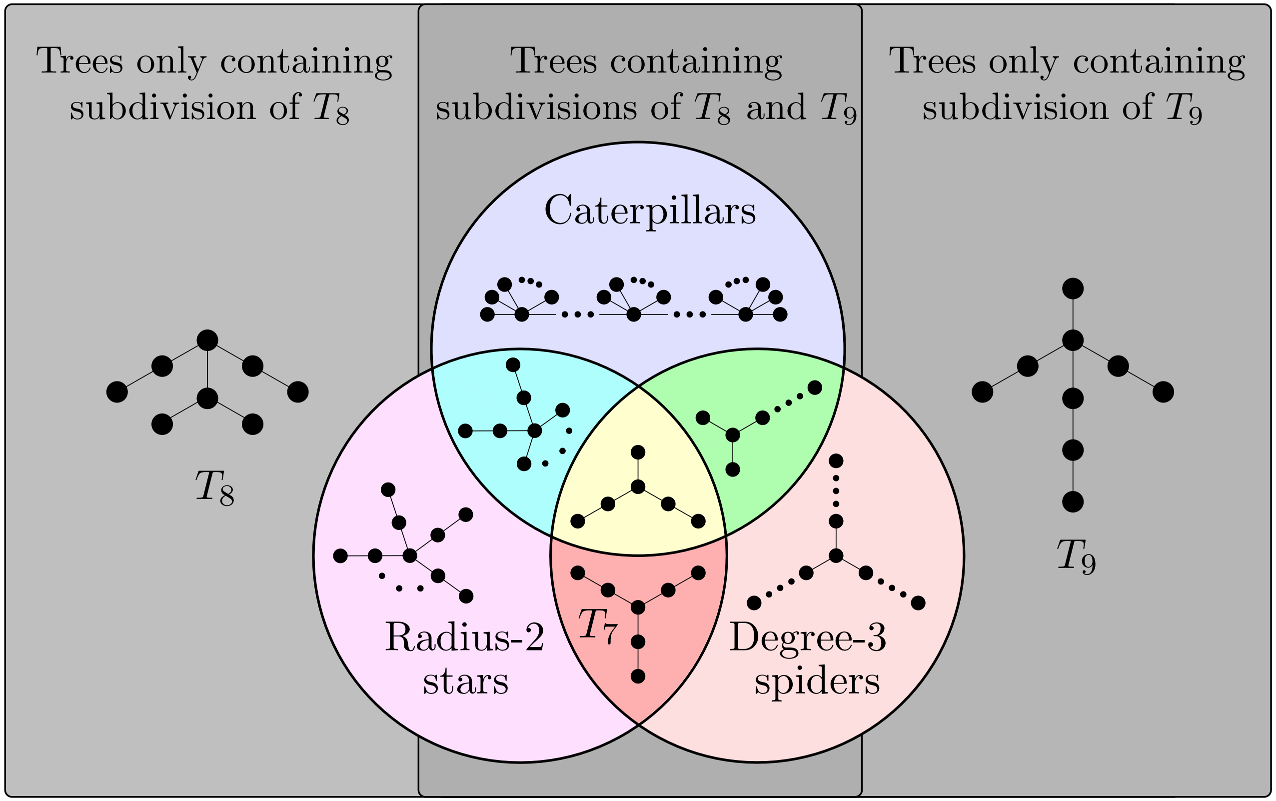
- Proof idea:
  - A tree that is not a degree-3 spider or radius-2 star has two cases
    - Has at least one vertex of degree-4 – contains T_9

Outline – Unlabeled Level Planar Trees

- Background
- Unlabeled Level Planar Trees



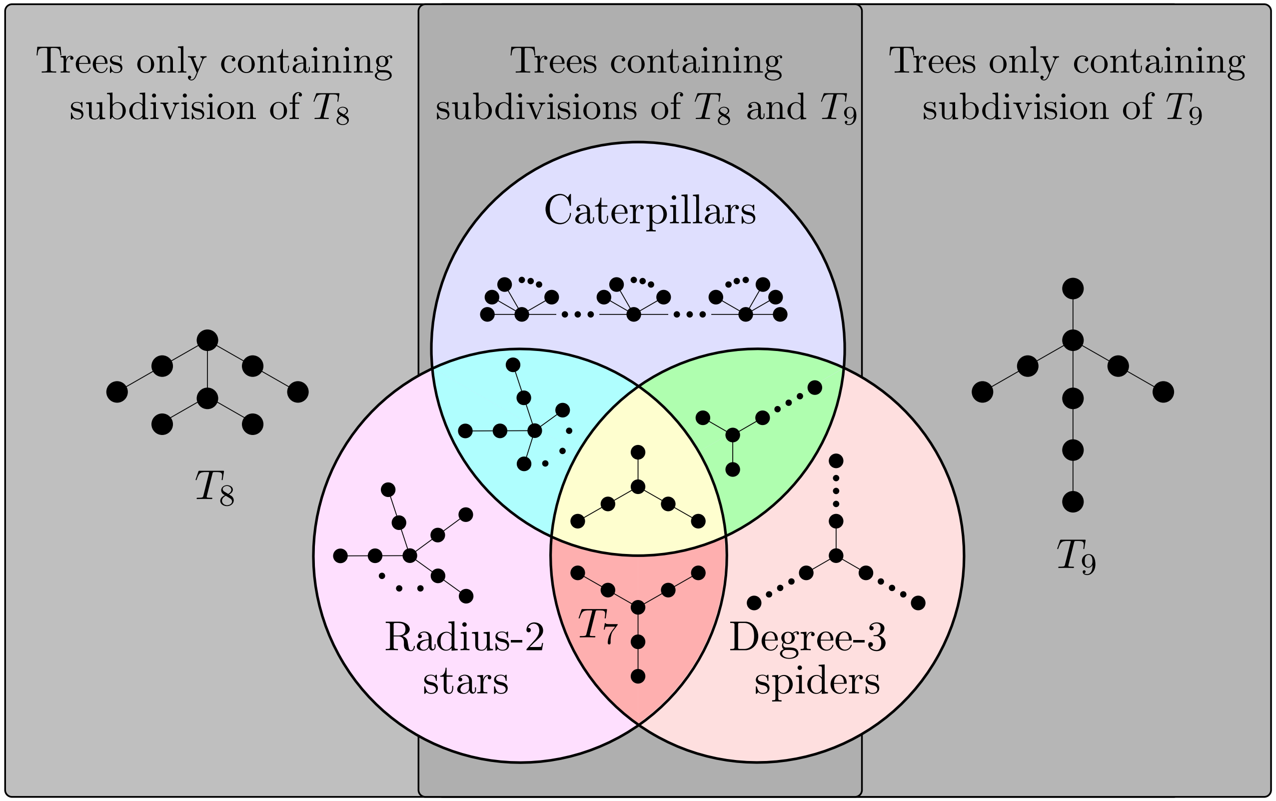
- Background
- Unlabeled Level Planar Trees



- First showed T_8 and T_9 are not ULP

Outline – Unlabeled Level Planar Trees

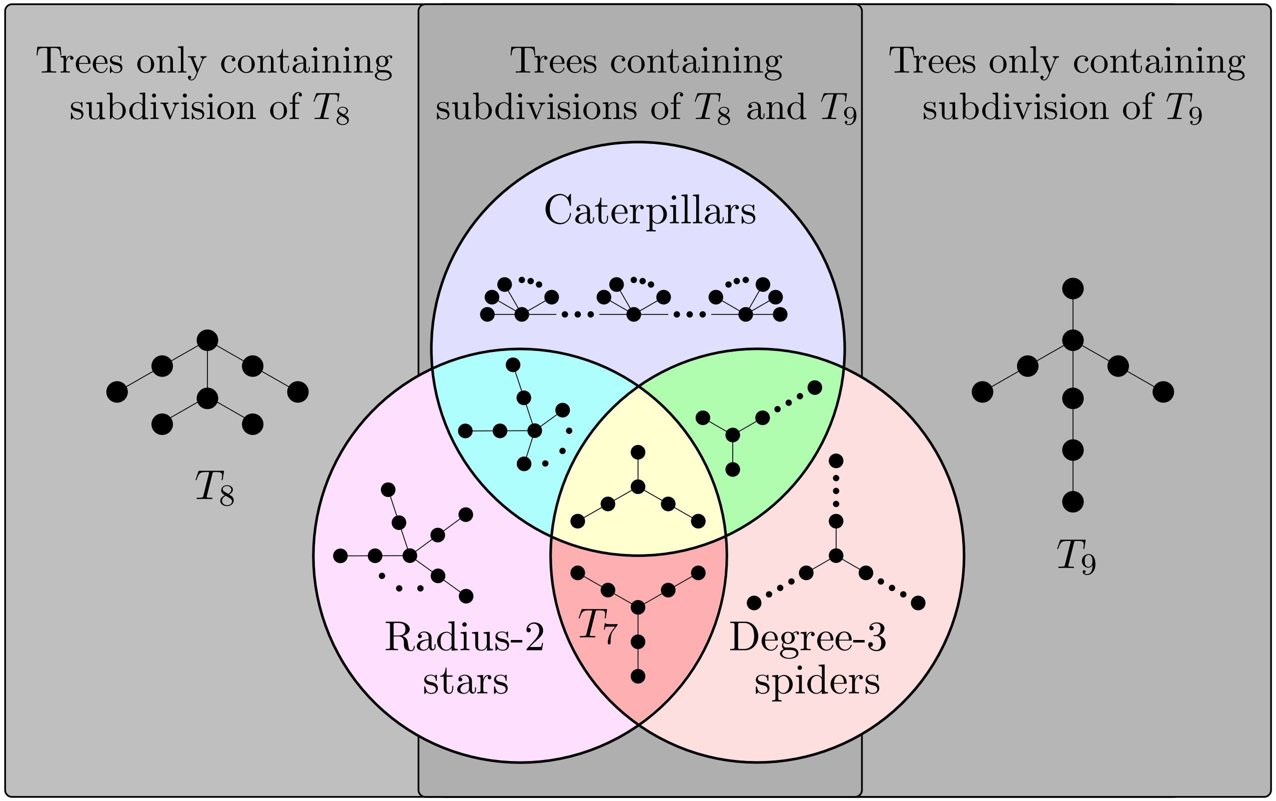
- Background
- Unlabeled Level Planar Trees



  - Showed caterpillars, radius-2 stars and degree-3 spiders are ULP

Outline – Unlabeled Level Planar Trees

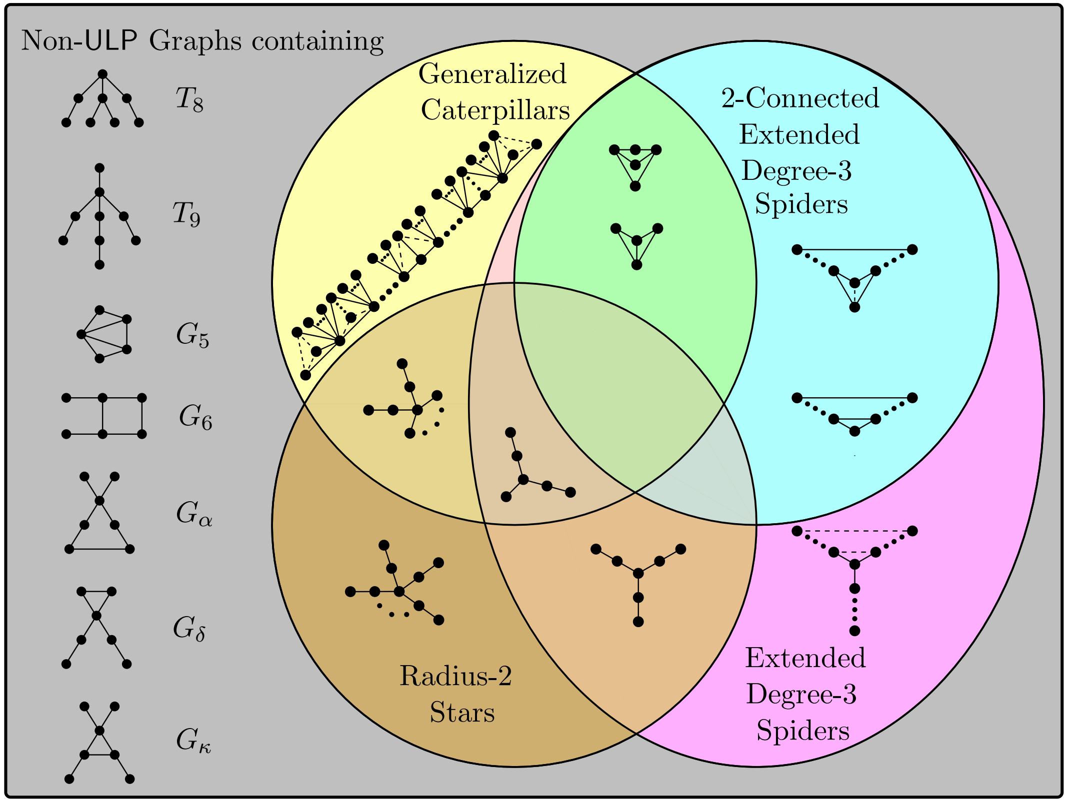
- Background
- Unlabeled Level Planar Trees



- Last showed all trees fall into one the above categories

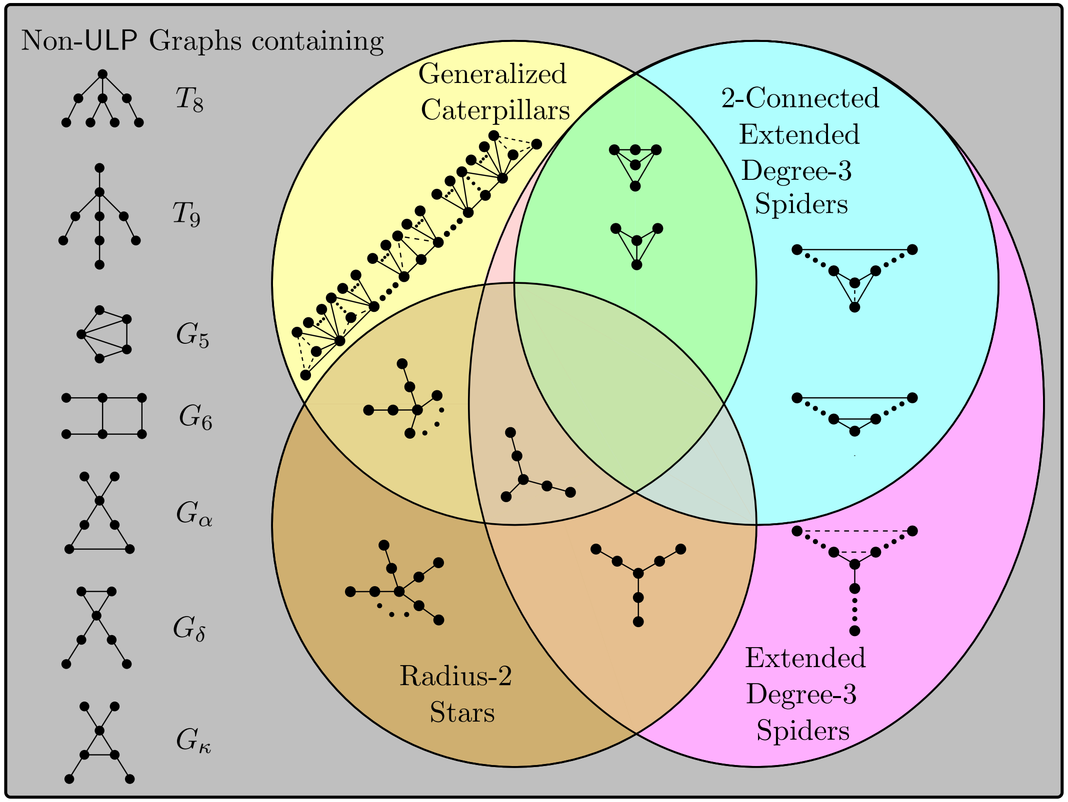
Outline – Unlabeled Level Planar Graphs

- Background
- Unlabeled Level Planar Trees
- Unlabeled Level Planar Graphs



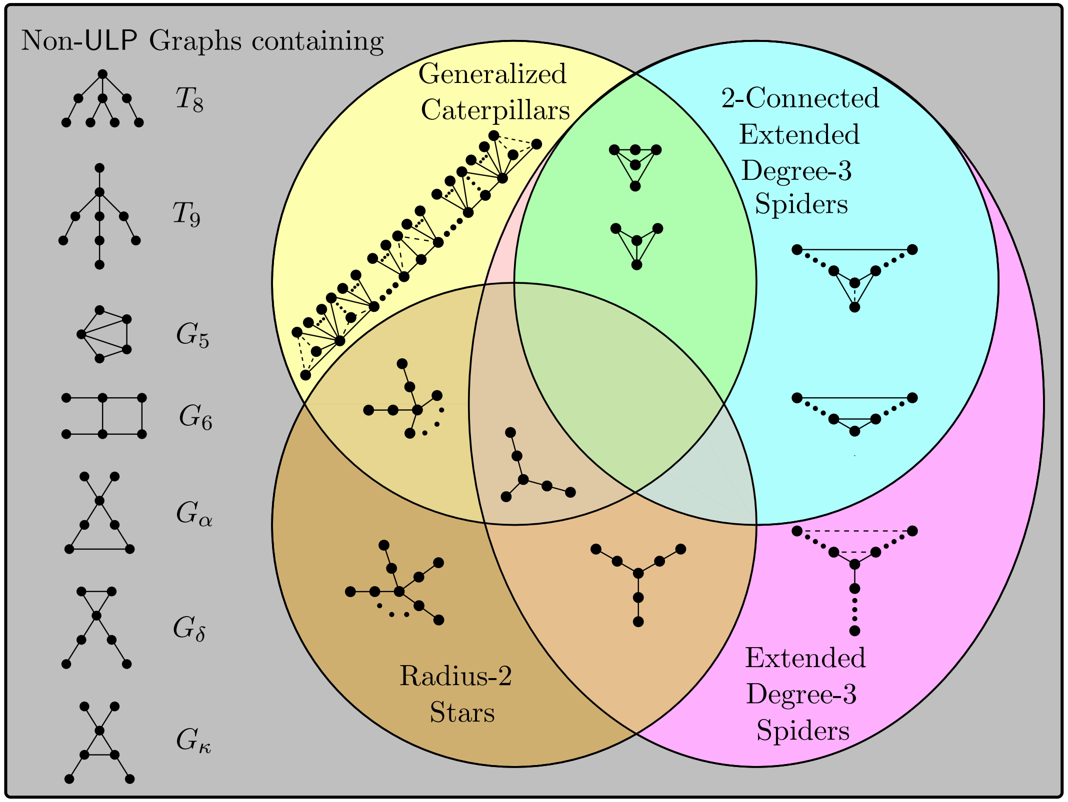
Outline – Unlabeled Level Planar Graphs

- Background
- Unlabeled Level Planar Trees
- Unlabeled Level Planar Graphs



- First show forbidden graphs are not ULP

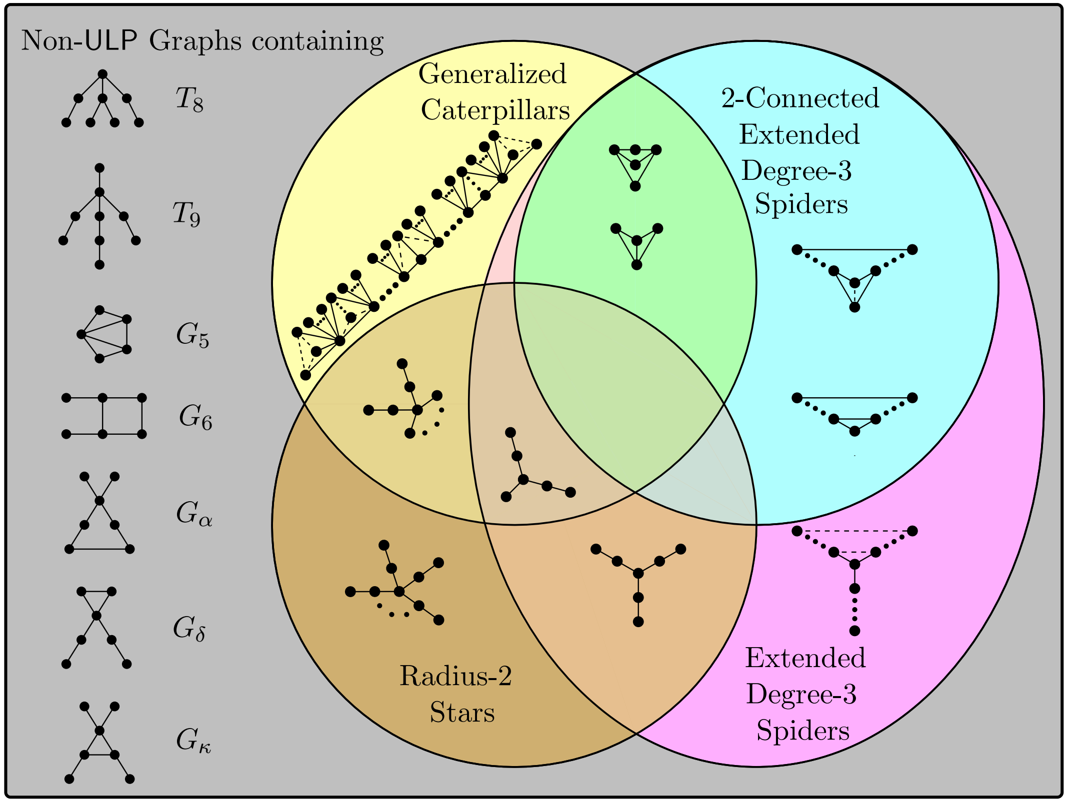
- Background
- Unlabeled Level Planar Trees
- Unlabeled Level Planar Graphs



  - ► Then extend drawing for ULP graphs with cycles

Outline – Unlabeled Level Planar Graphs

- Background
- Unlabeled Level Planar Trees
- Unlabeled Level Planar Graphs



- Finally describe how all graphs fall into one of the above categories

Forbidden Graphs – G_5, G_6, G_α, G_κ, G_δ

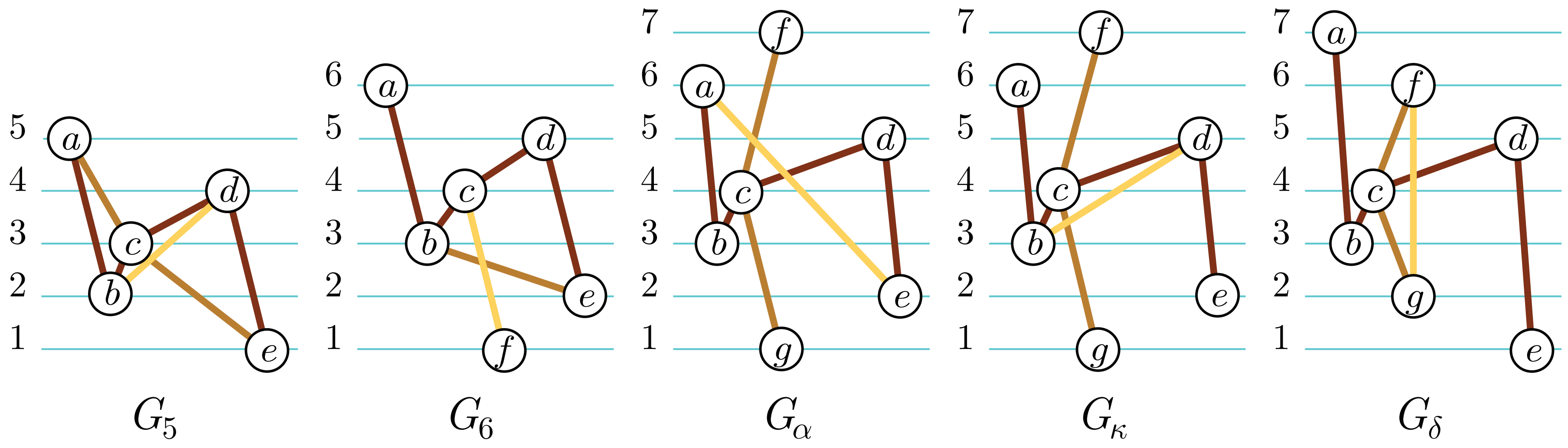
- For each ULP graph let C be the chain a—b—c—d—e

Forbidden Graphs – G_5, G_6, G_α, G_κ, G_δ

- For each ULP graph let C be the chain a—b—c—d—e
  - $\phi(a) < \phi(d) < \phi(c) < \phi(b) < \phi(e)$

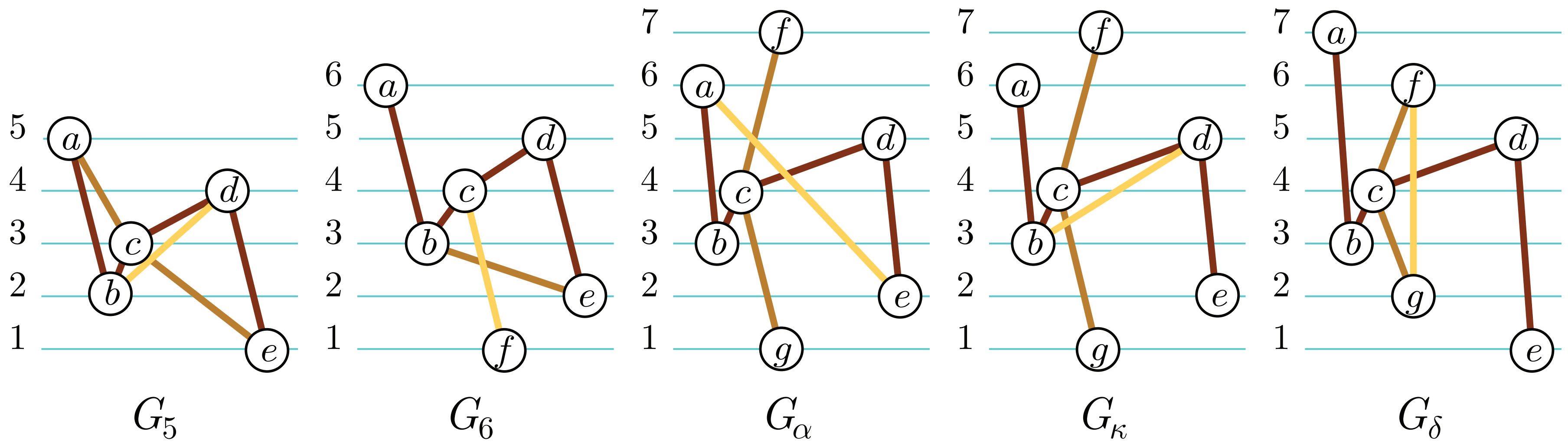
Forbidden Graphs – G_5, G_6, G_α, G_κ, G_δ

- For each ULP graph let C be the chain $a—b—c—d—e$
  - $\phi(a) < \phi(d) < \phi(c) < \phi(b) < \phi(e)$



Forbidden Graphs – G_5, G_6, G_α, G_κ, G_δ

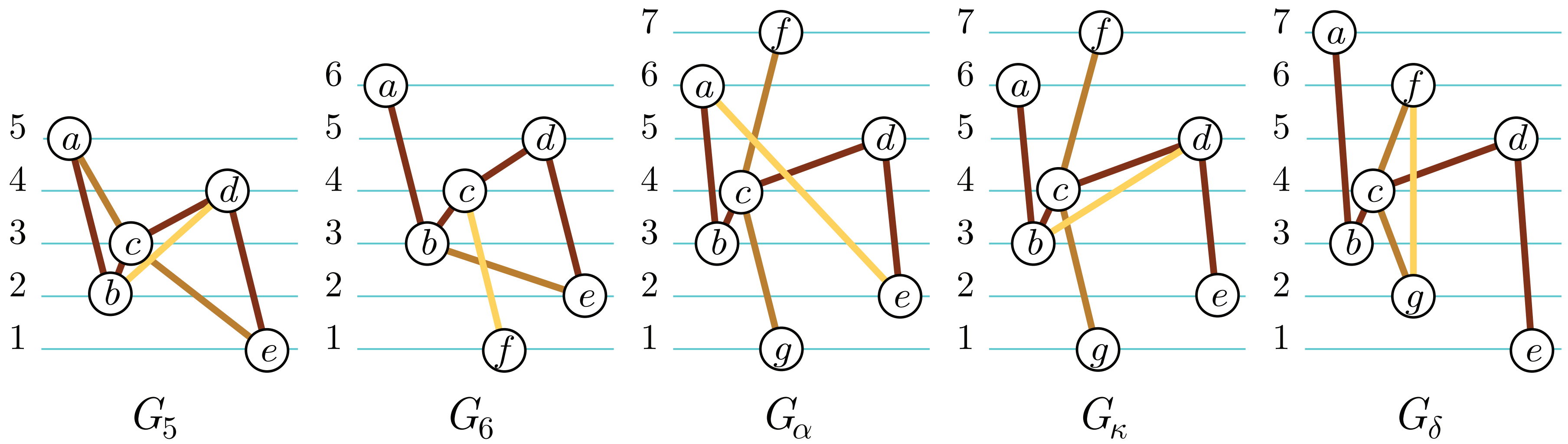
- For each ULP graph let C be the chain a—b—c—d—e
  - $\phi(a) < \phi(d) < \phi(c) < \phi(b) < \phi(e)$



  - Use similar arguments as with T_8 and T_9

Forbidden Graphs – G_5, G_6, G_α, G_κ, G_δ

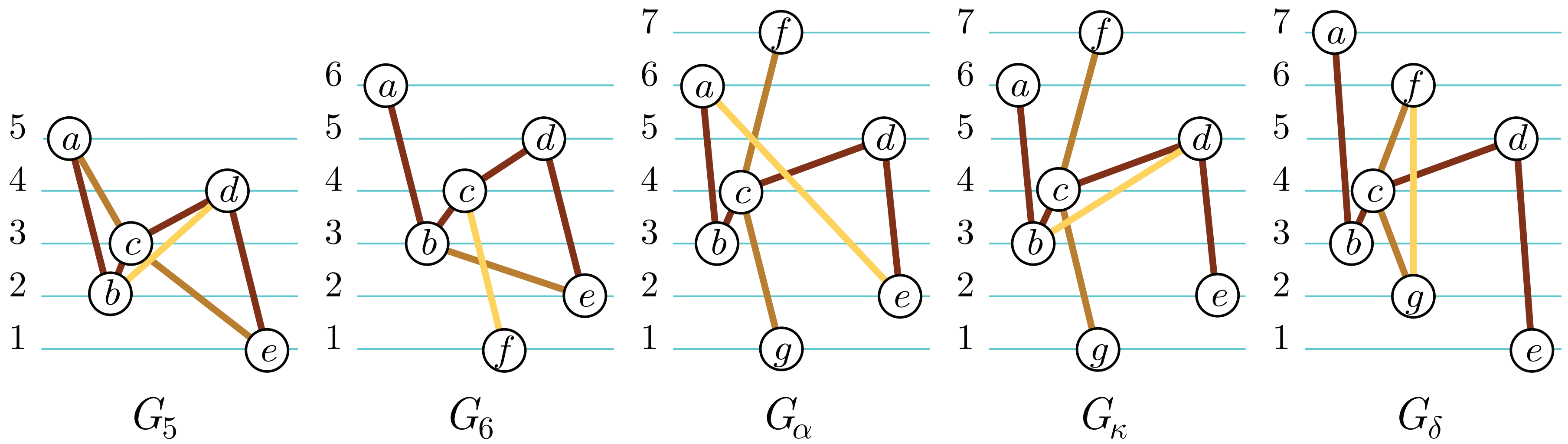
- For each ULP graph let C be the chain $a{-}b{-}c{-}d{-}e$
  - $\phi(a) < \phi(d) < \phi(c) < \phi(b) < \phi(e)$



  - Implies one of the other edges must cross C

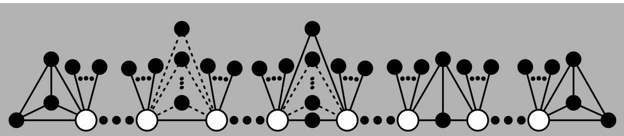
Forbidden Graphs – $G_5, G_6, G_\alpha, G_\kappa, G_\delta$

- For each ULP graph let C be the chain $a{-}b{-}c{-}d{-}e$
  - $\phi(a) < \phi(d) < \phi(c) < \phi(b) < \phi(e)$



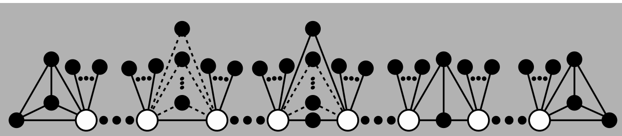
Lemma 7 *Any graph with a cycle containing a subdivision of G_5, G_6, G_α, G_κ, or G_δ cannot be ULP.*

Generalized Caterpillar



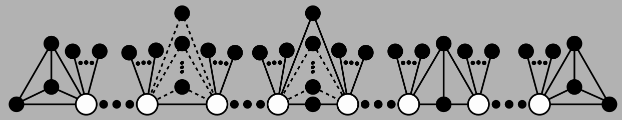
- Drawing a generalized caterpillar in linear time:

Generalized Caterpillar



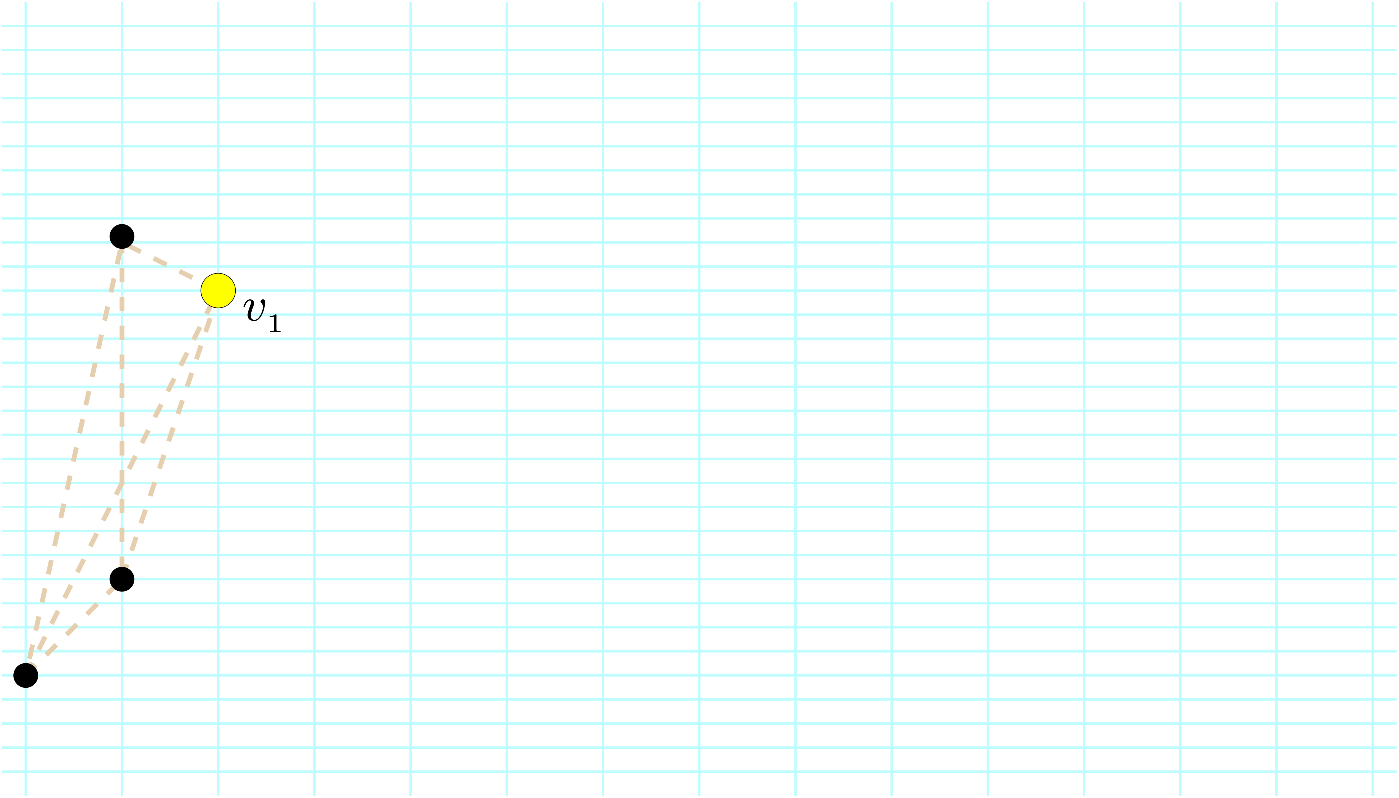
- Drawing a generalized caterpillar in linear time:

  Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\textit{onto}]{1:1} \{1, 2, \ldots, n\}$.*



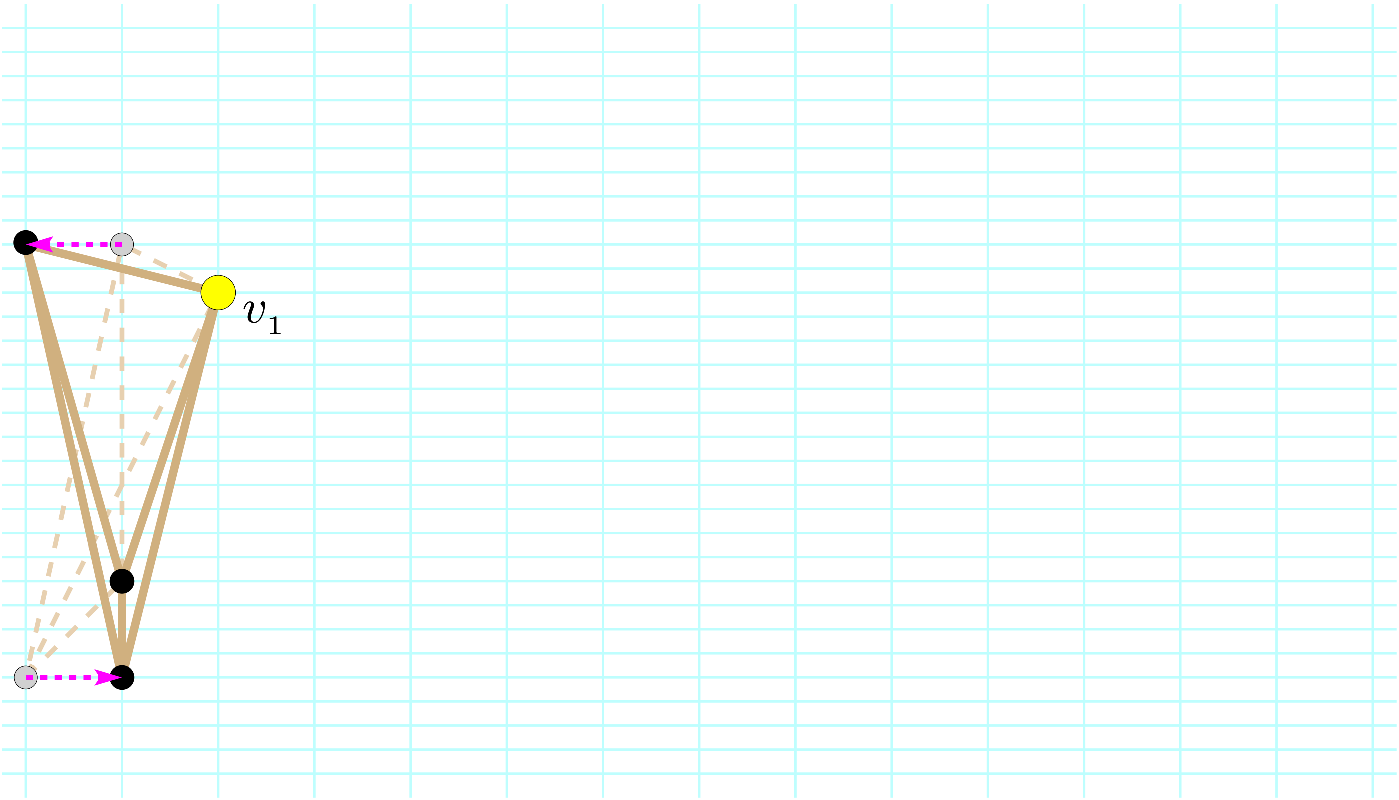
- Drawing a generalized caterpillar in linear time:

  Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



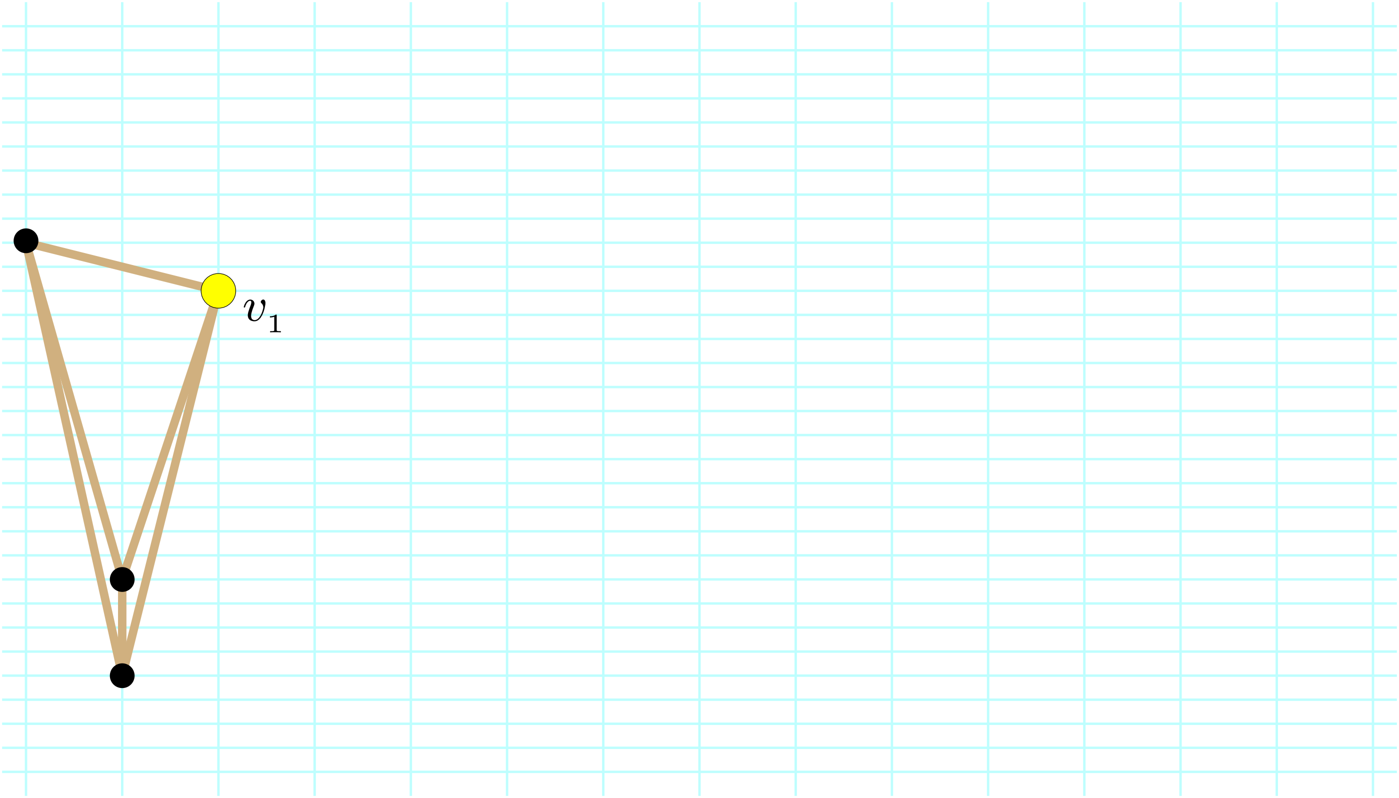
- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



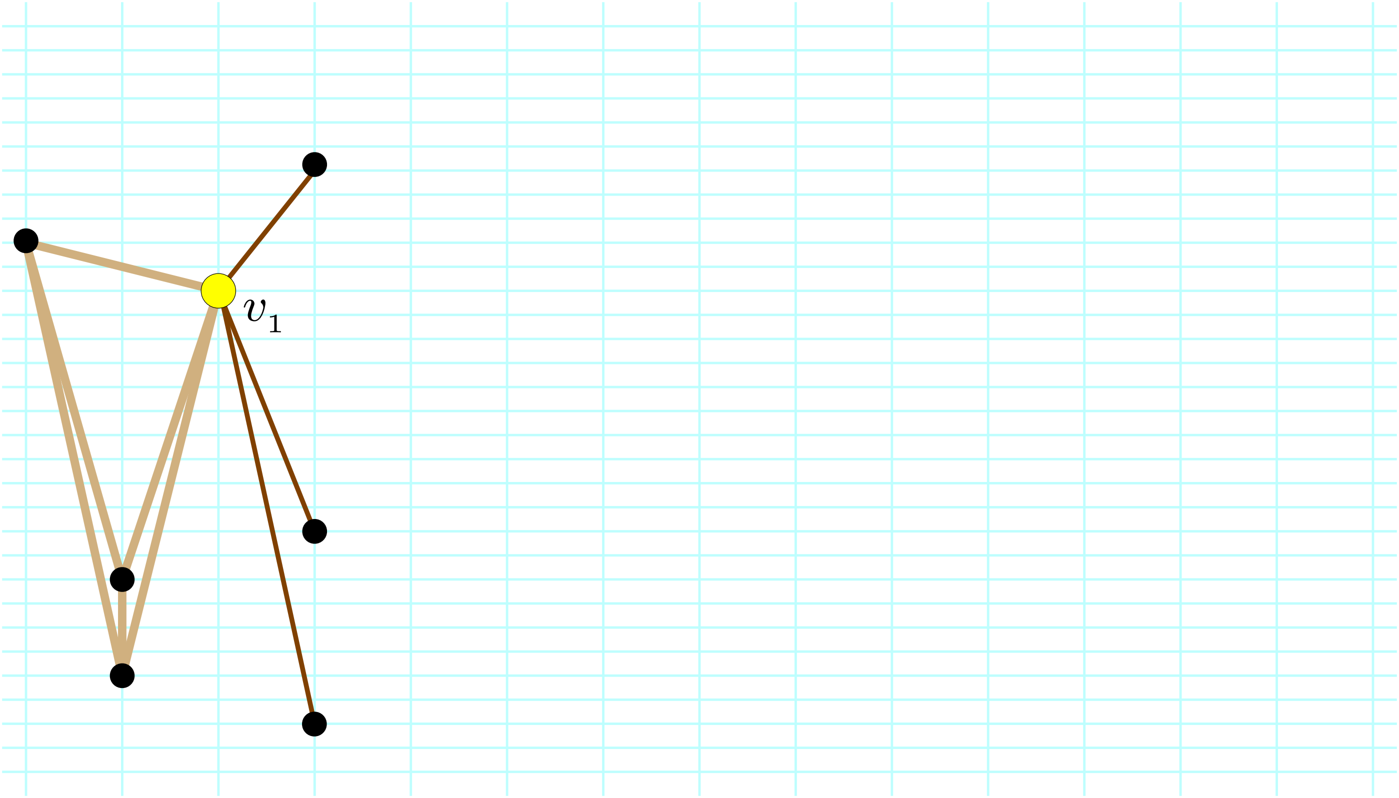
- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



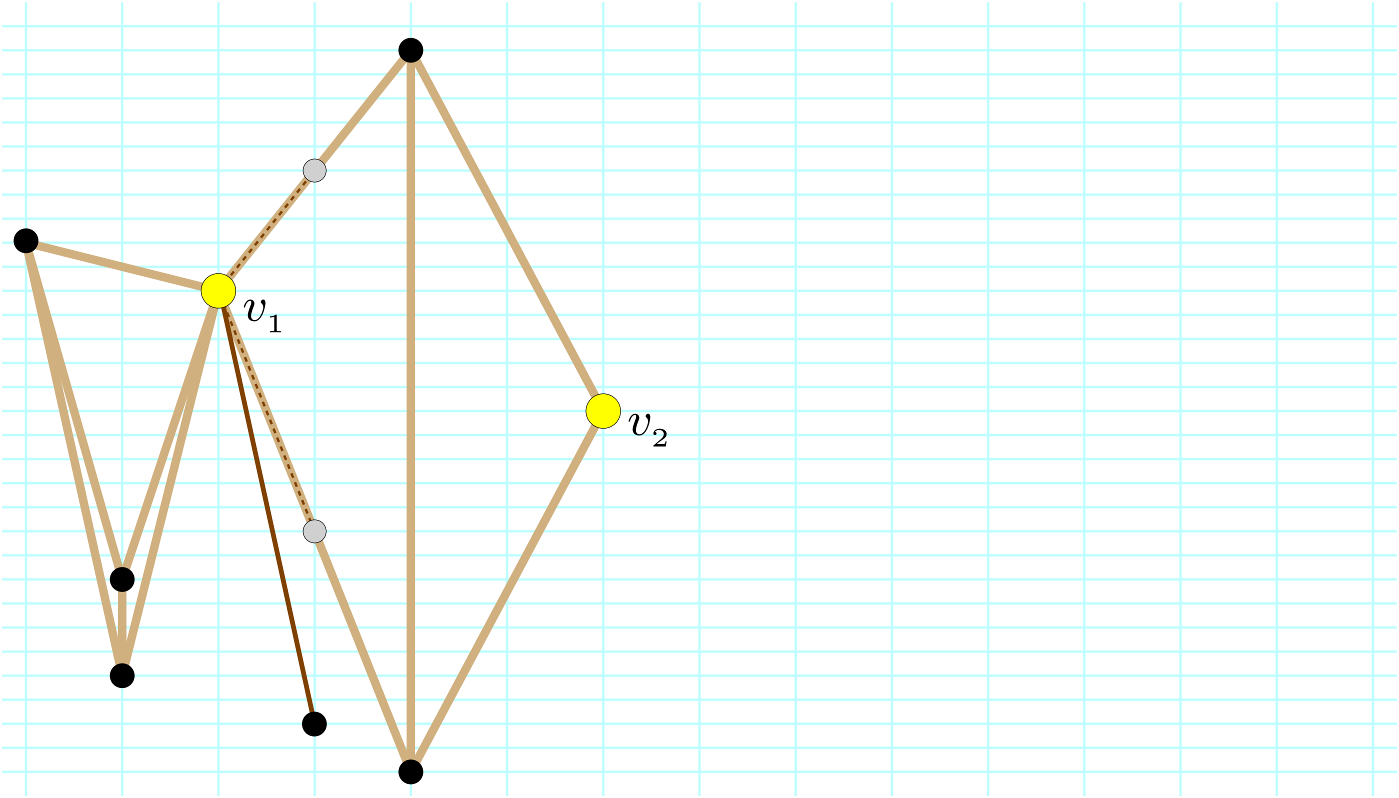
- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



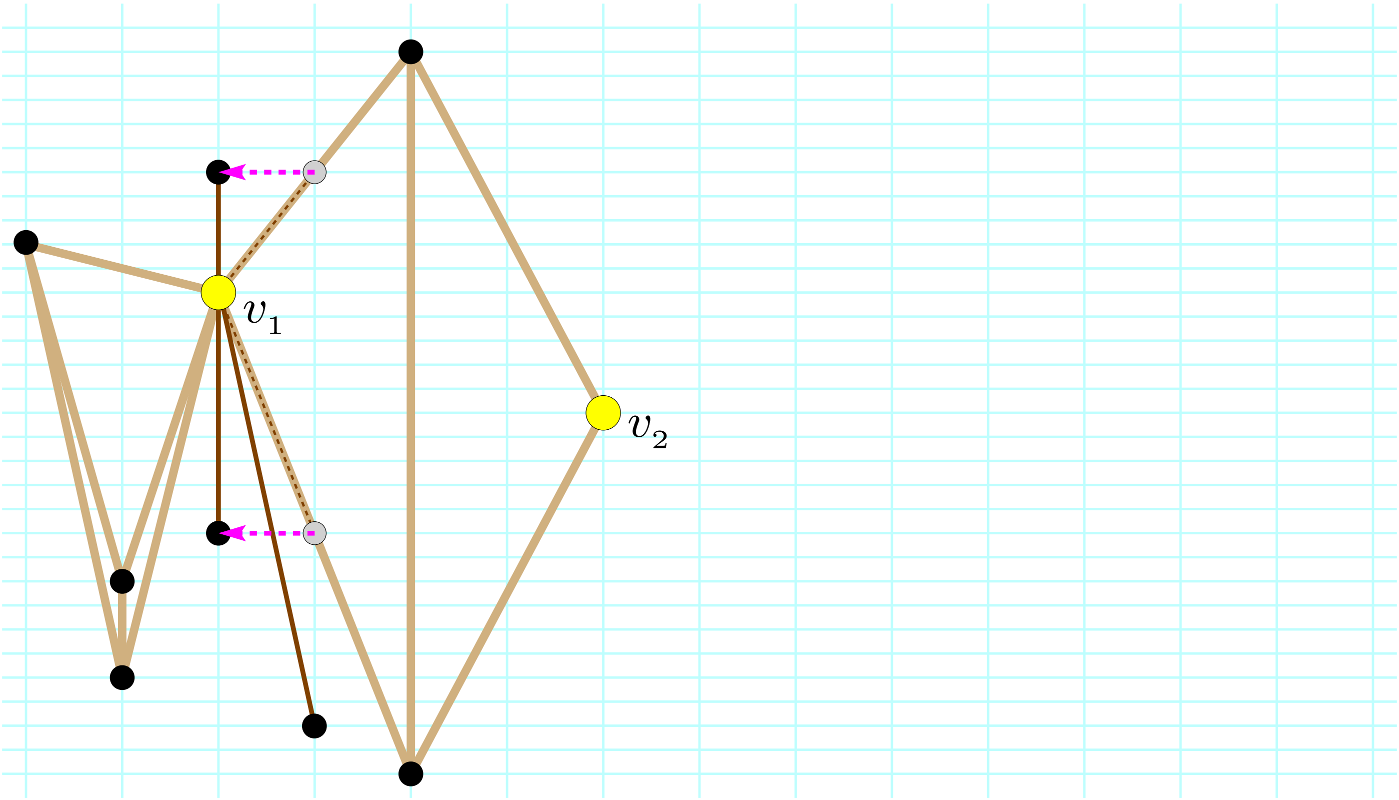
- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



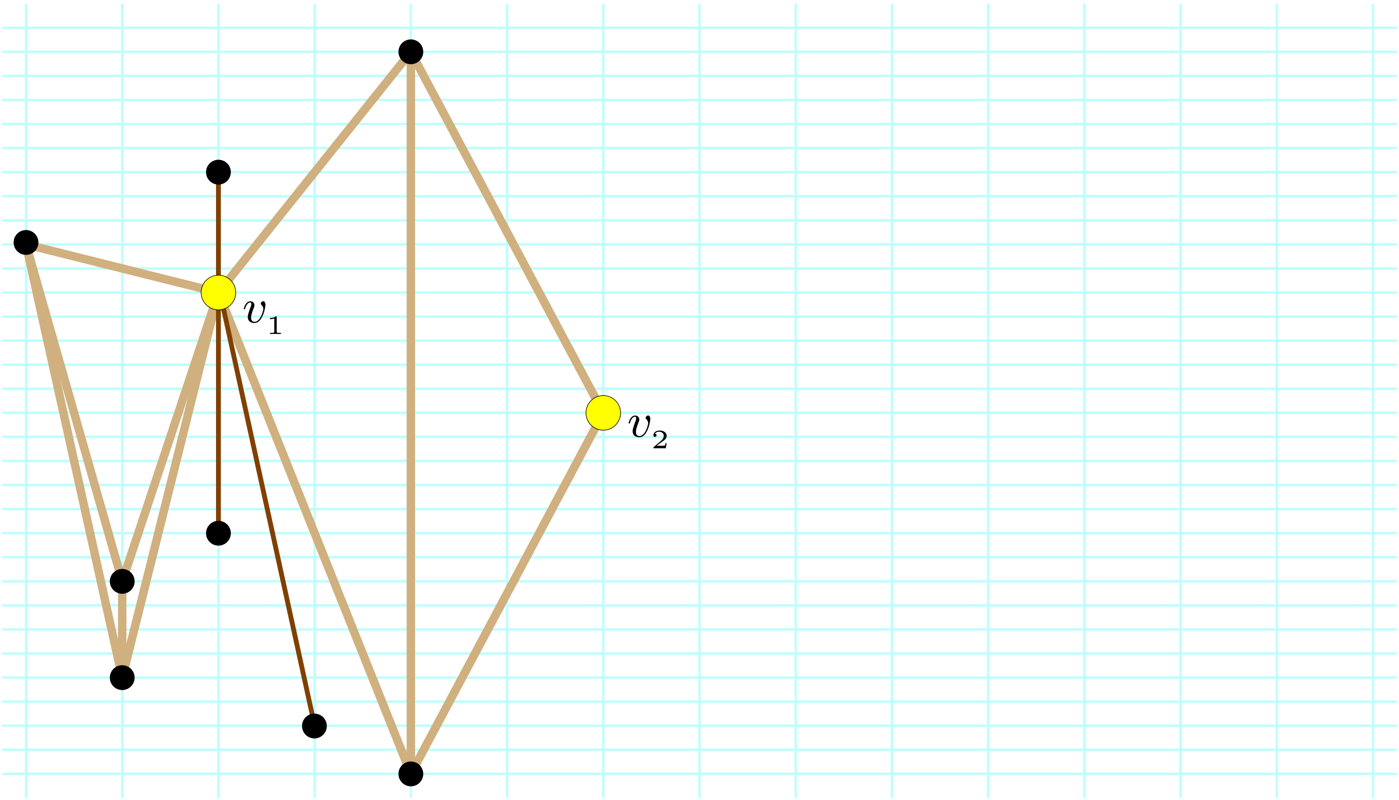
- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



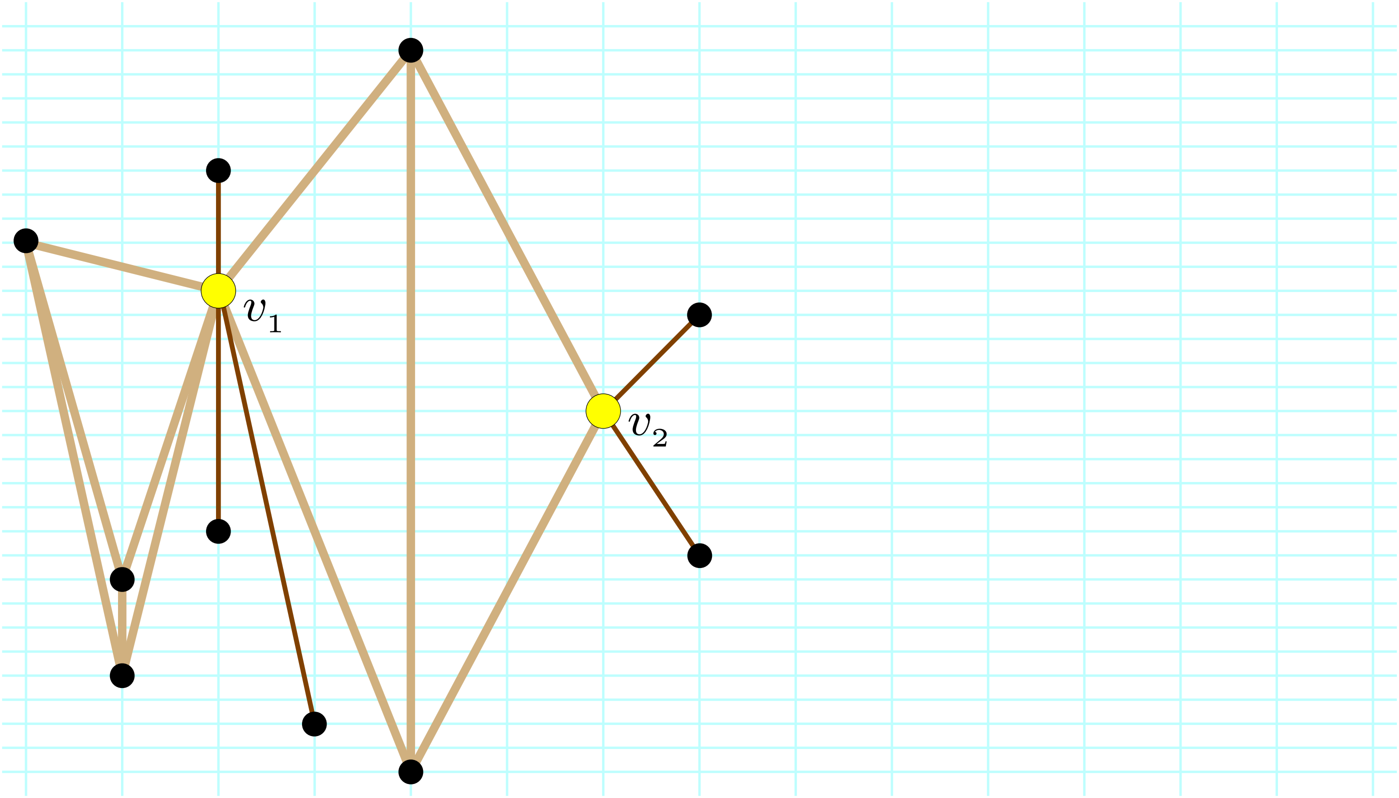
- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



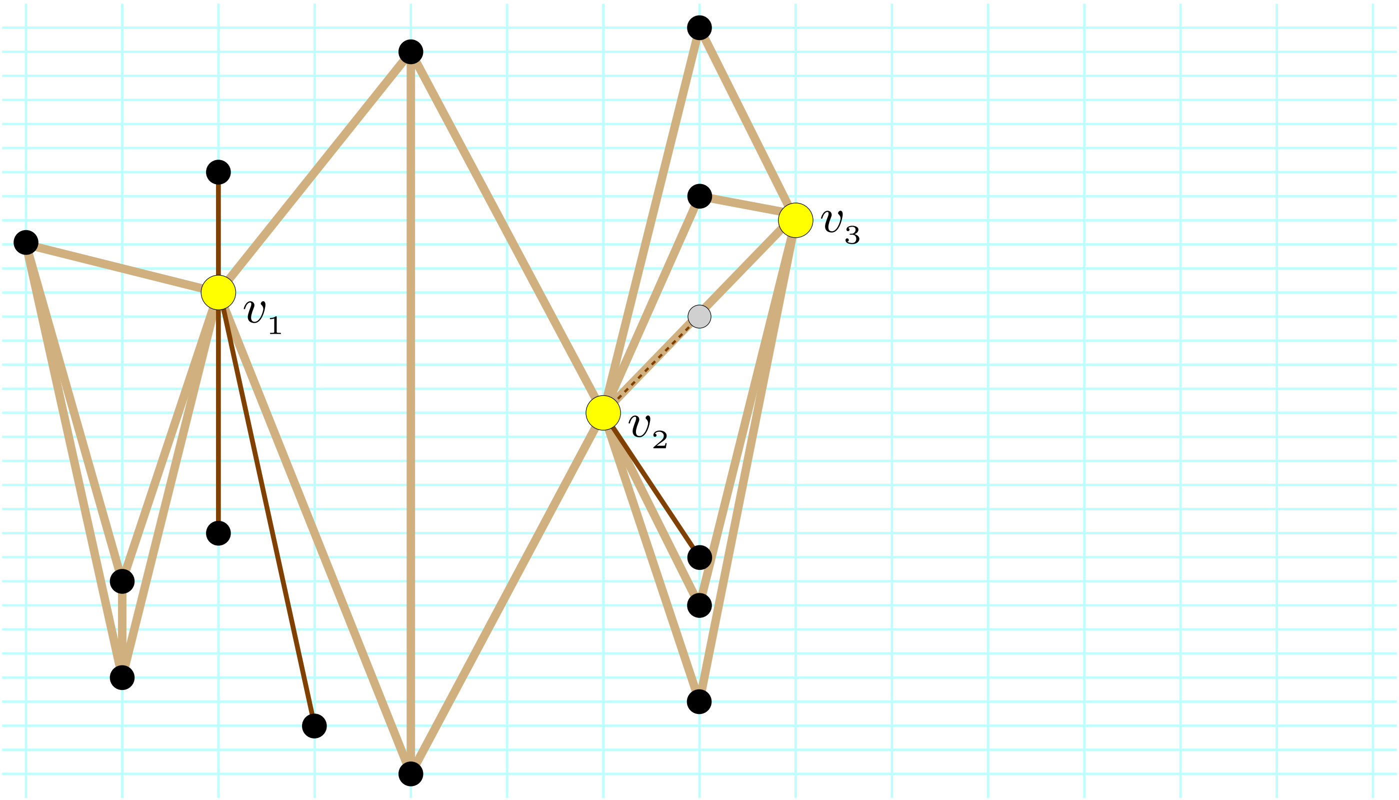
- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



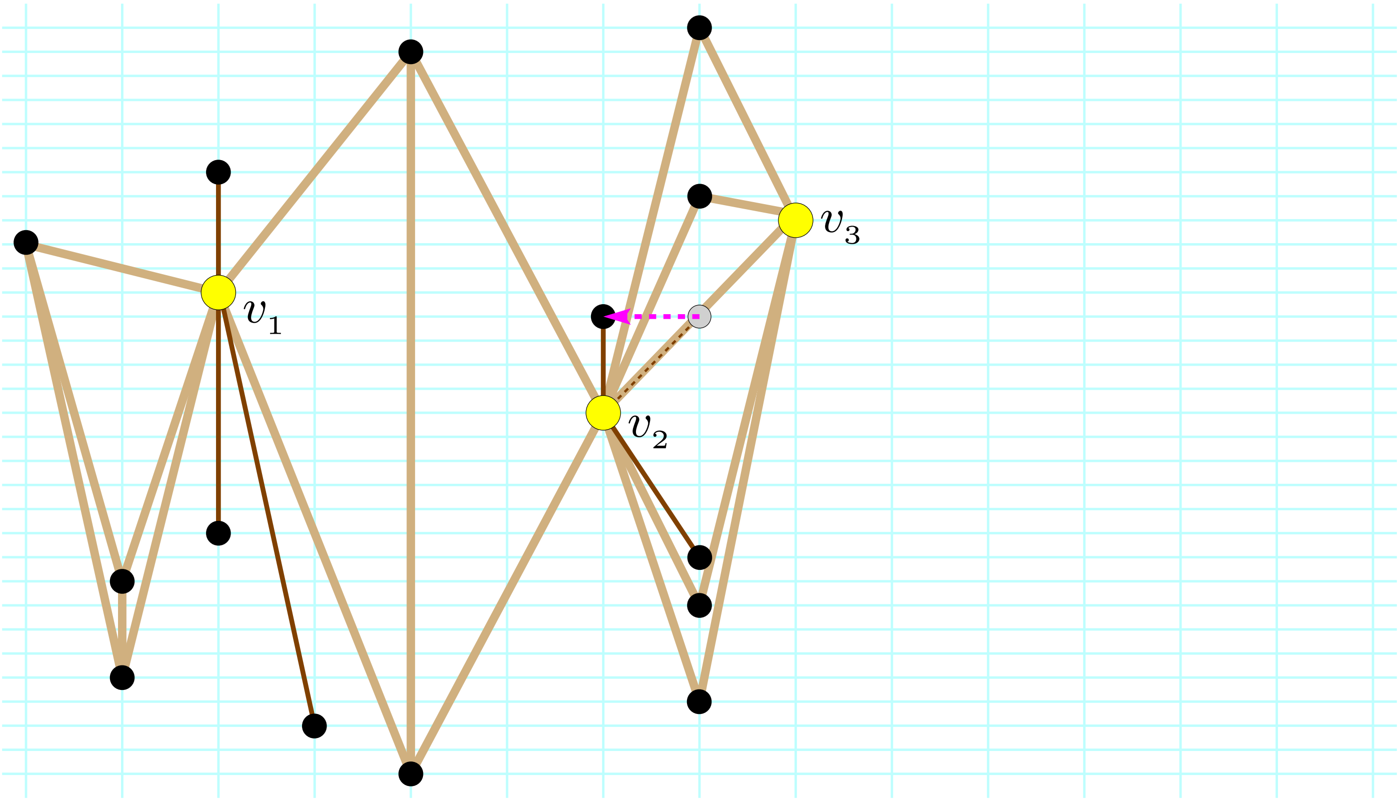
- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



- Drawing a generalized caterpillar in linear time:

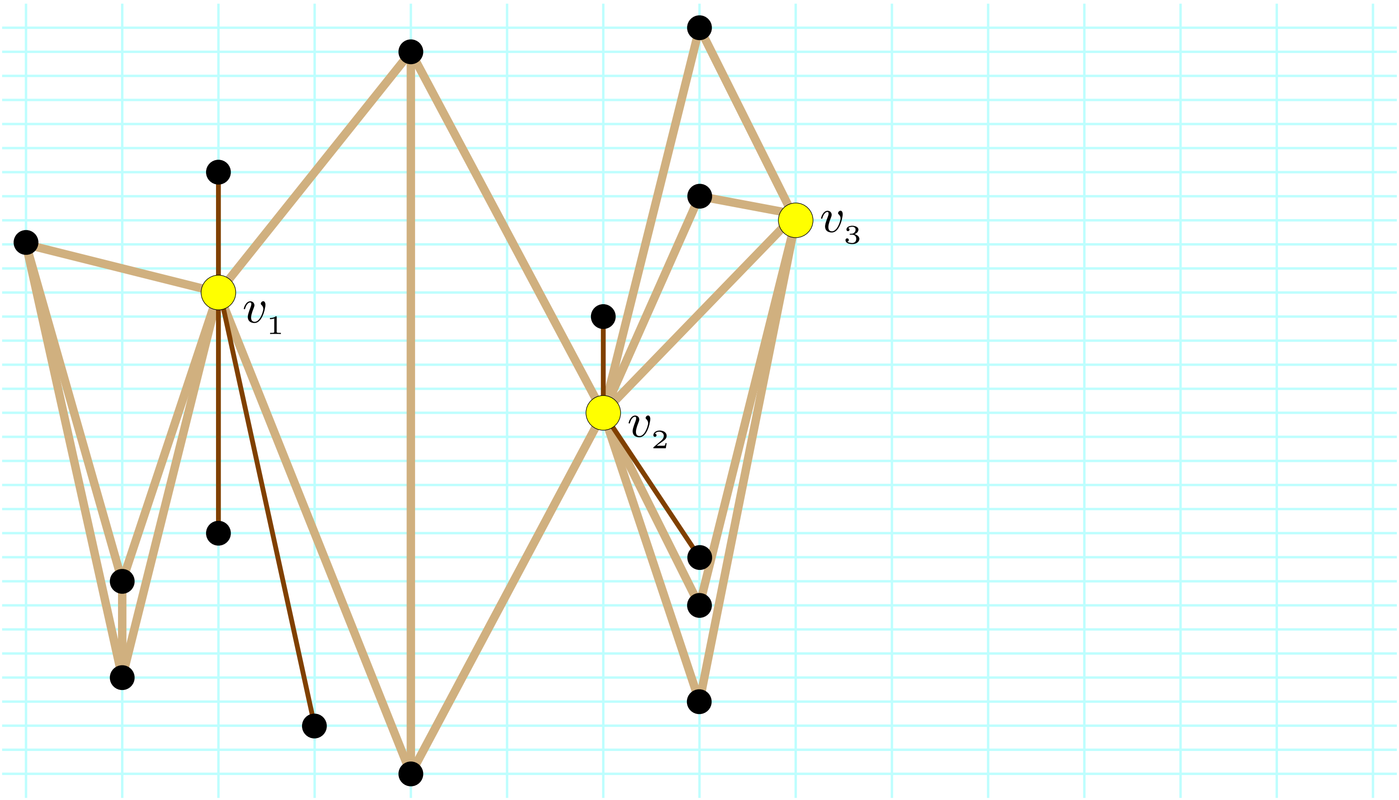
Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



Generalized Caterpillar

- Drawing a generalized caterpillar in linear time:

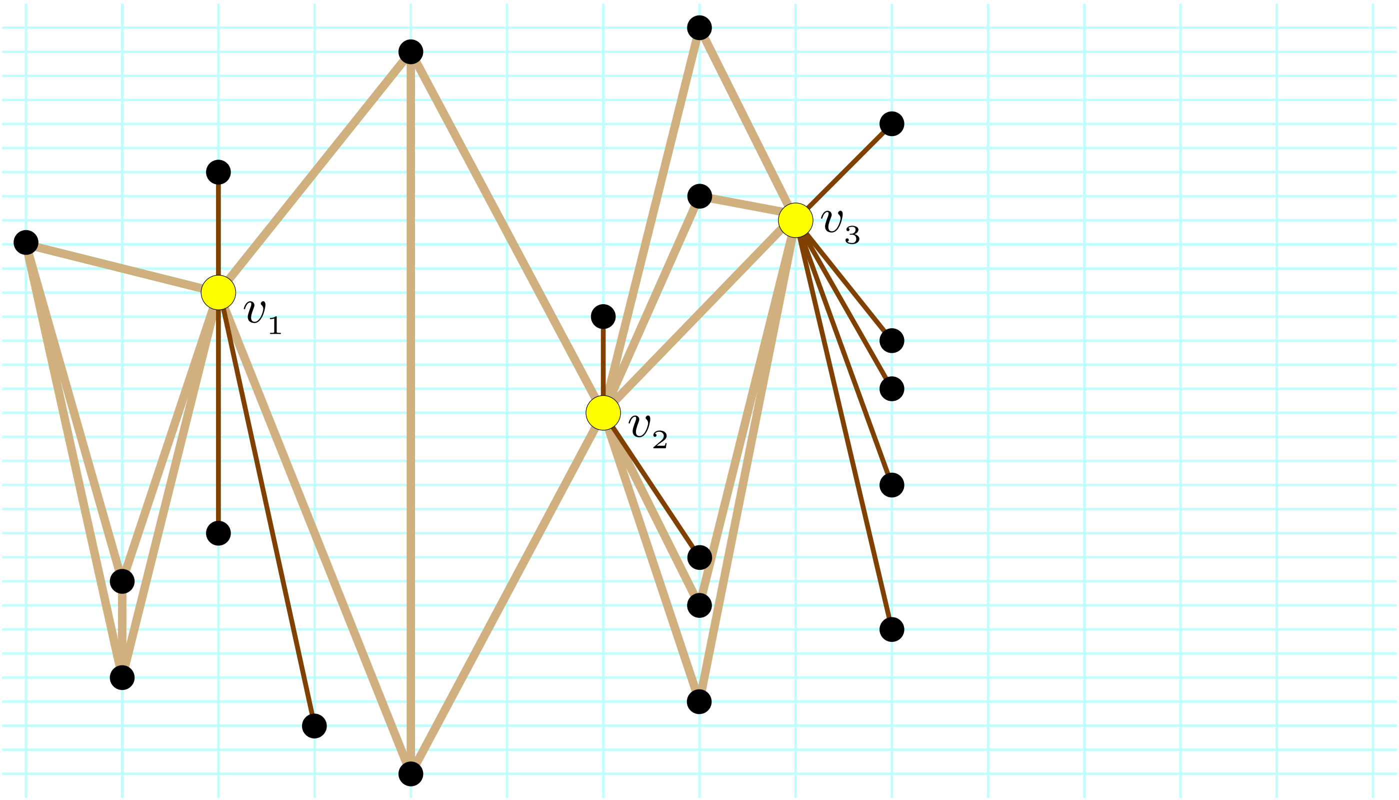
Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



Generalized Caterpillar

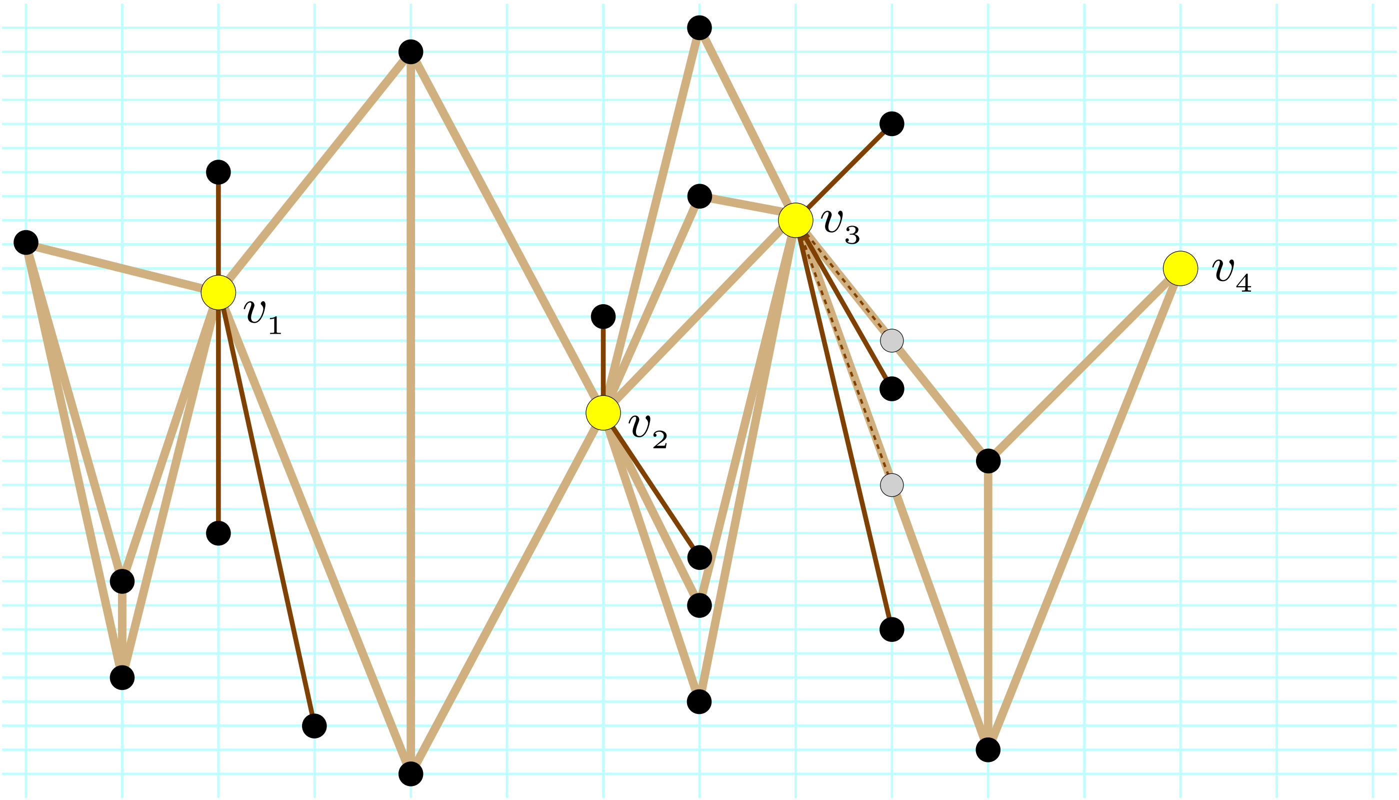
- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

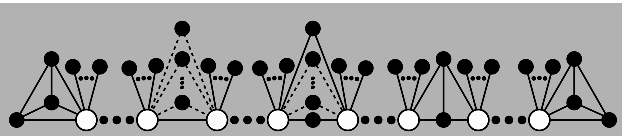


- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

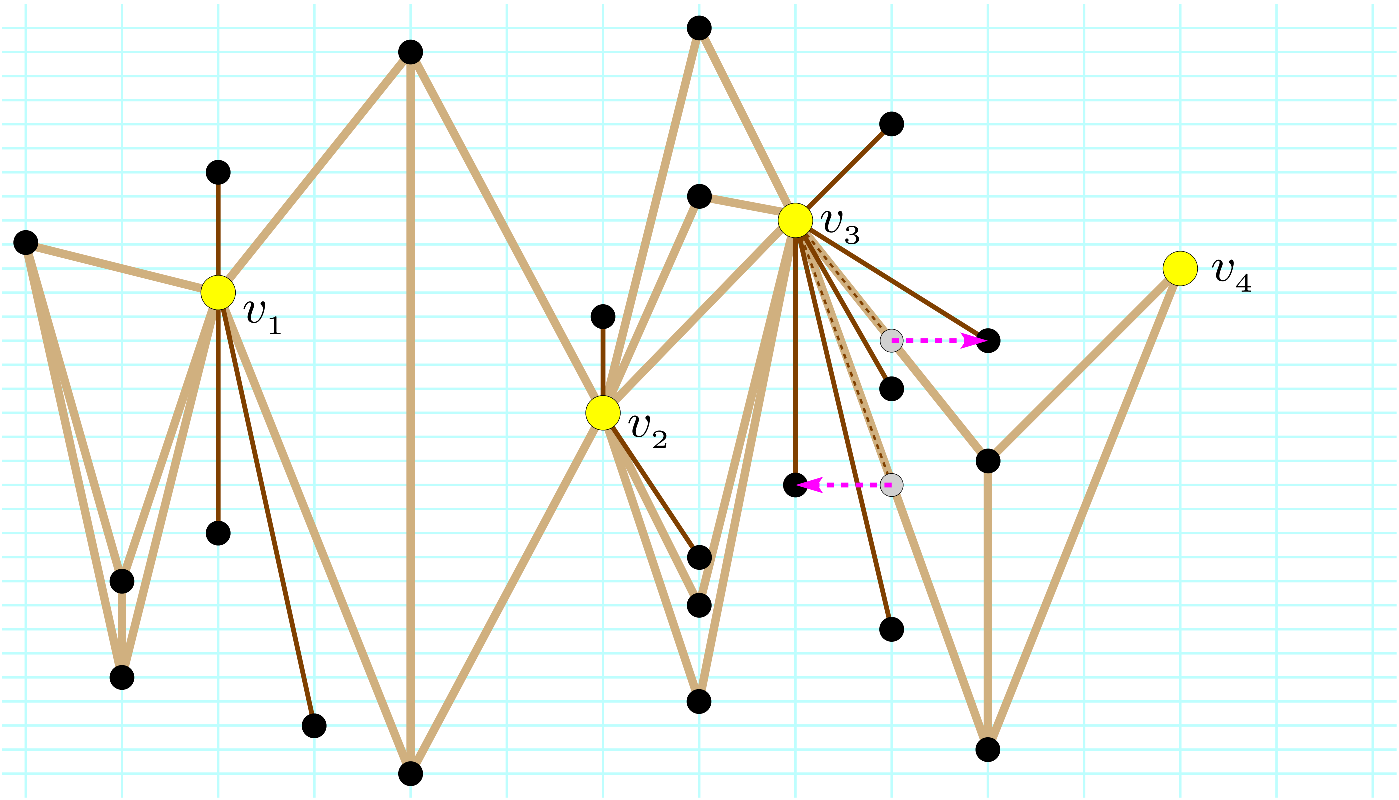


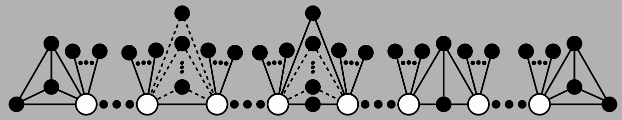
Generalized Caterpillar



- Drawing a generalized caterpillar in linear time:

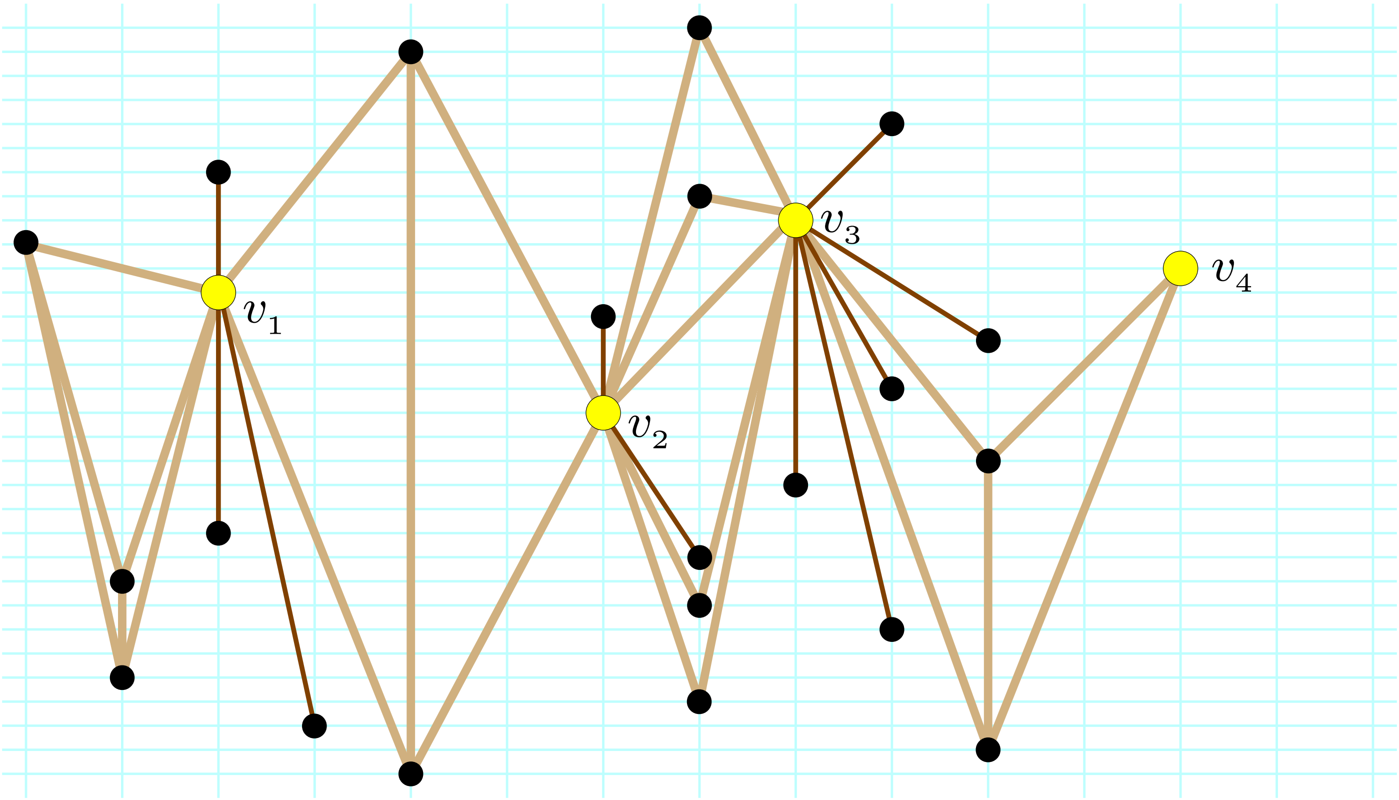
  Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*





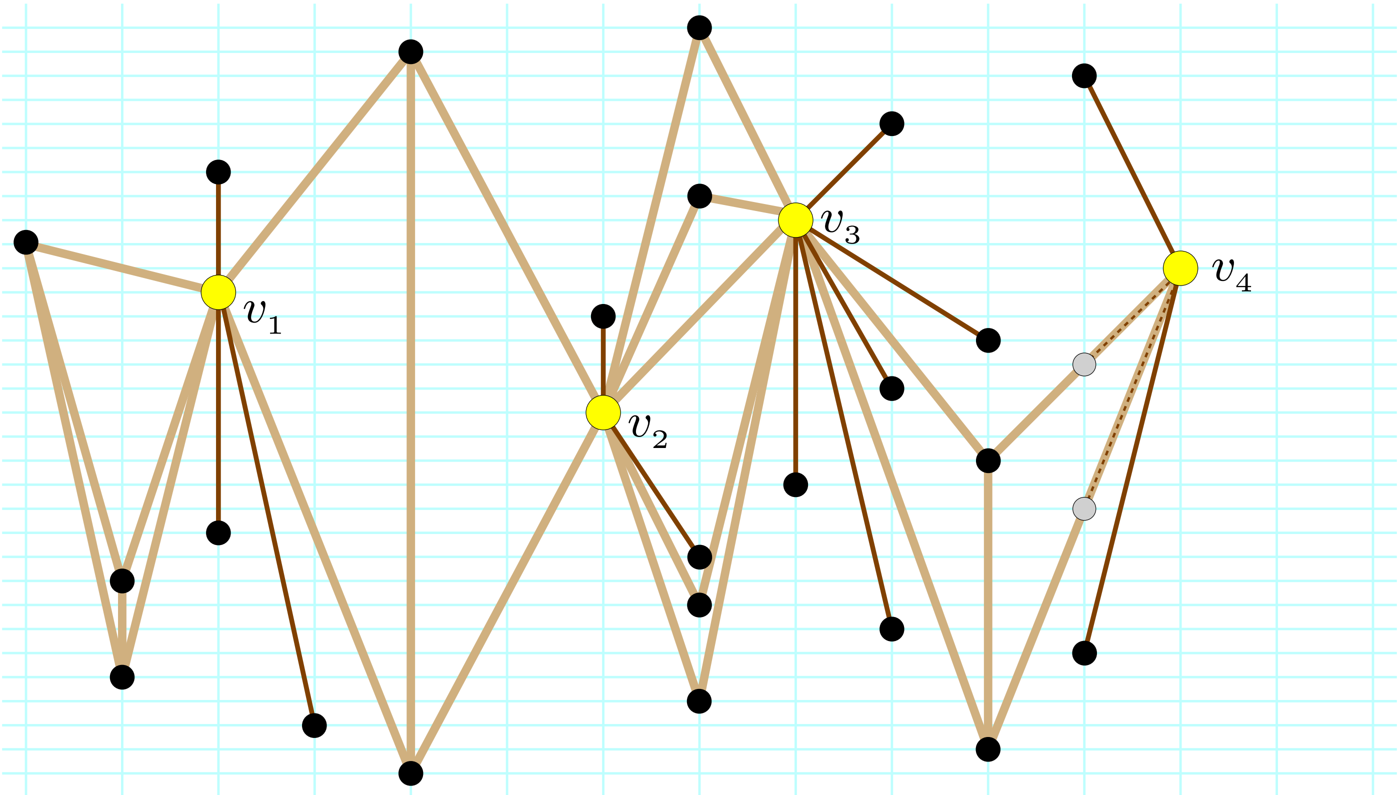
- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



- Drawing a generalized caterpillar in linear time:

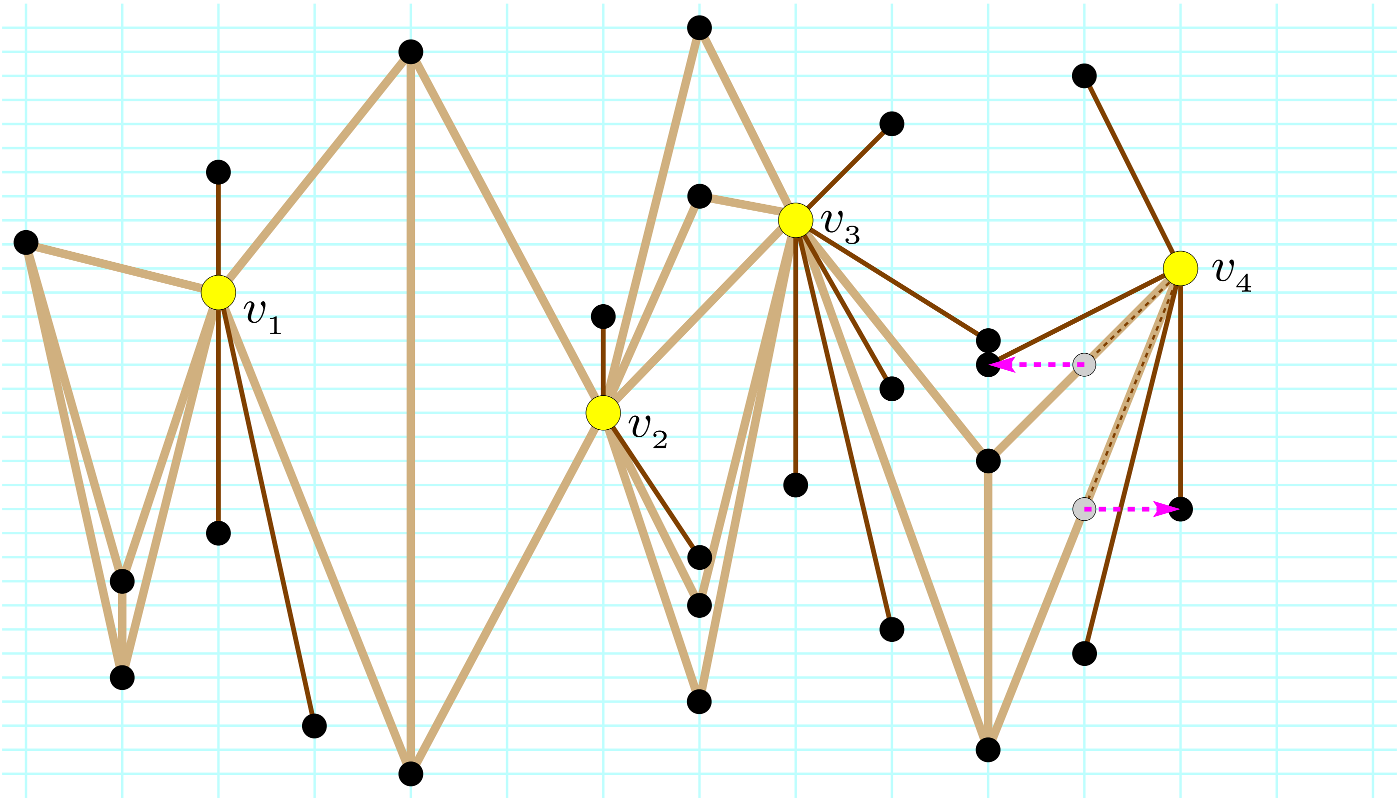
Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



Generalized Caterpillar

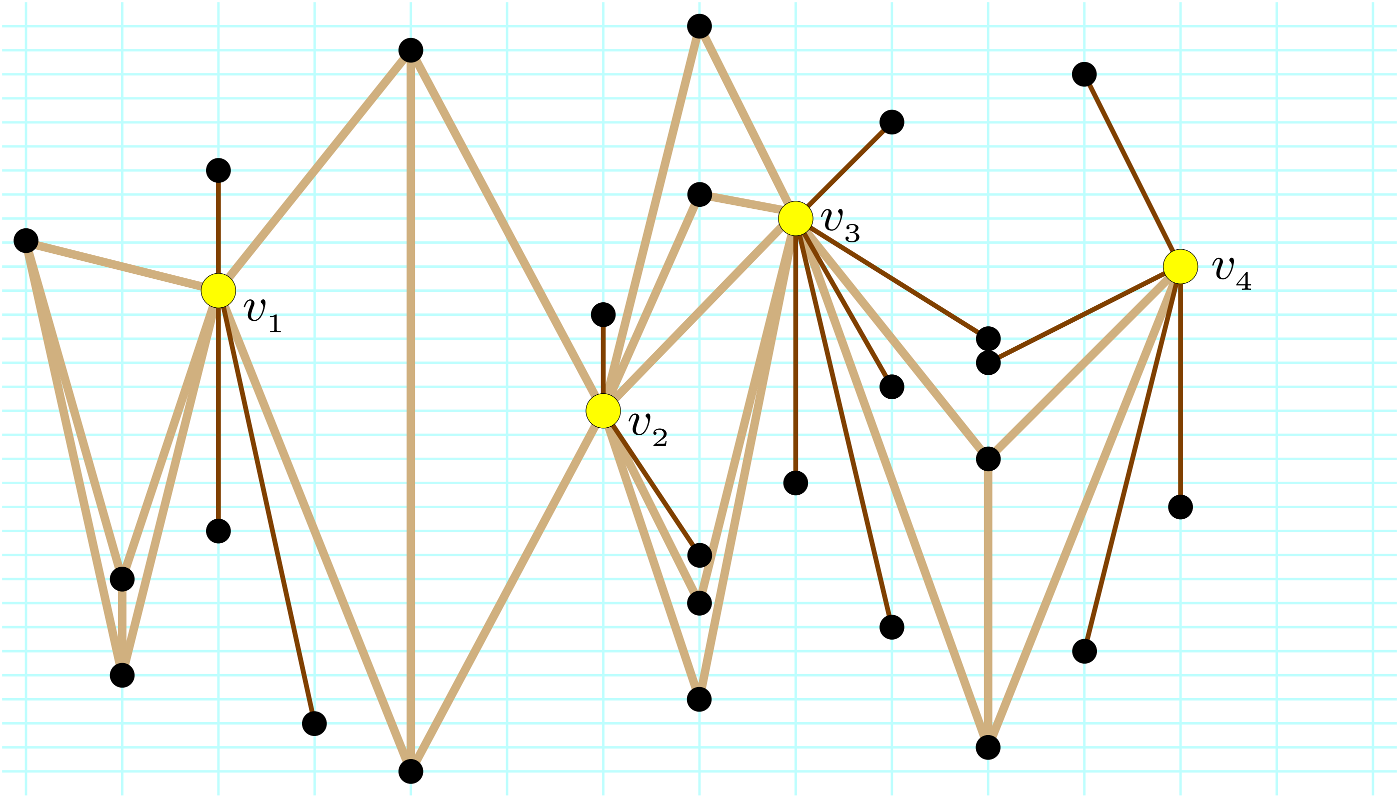
- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



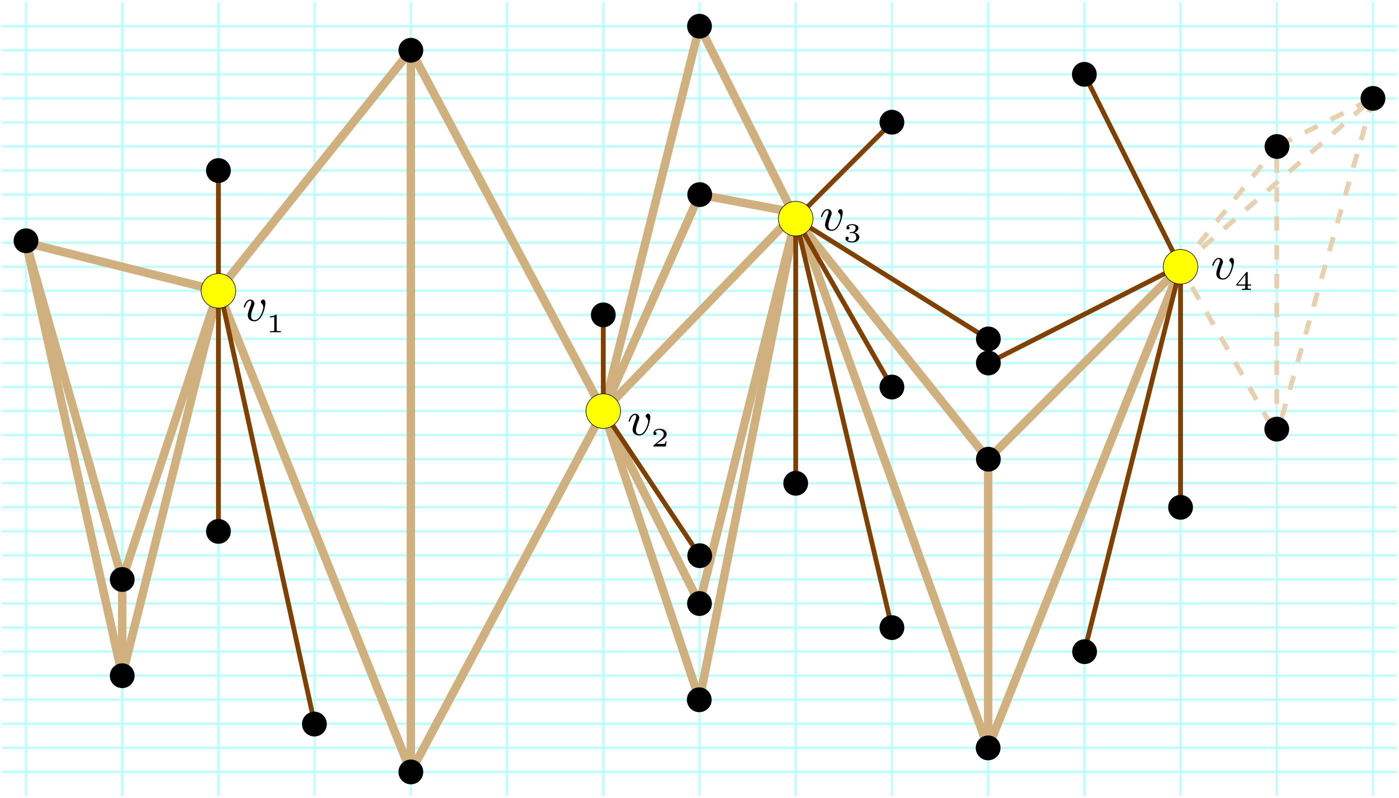
- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



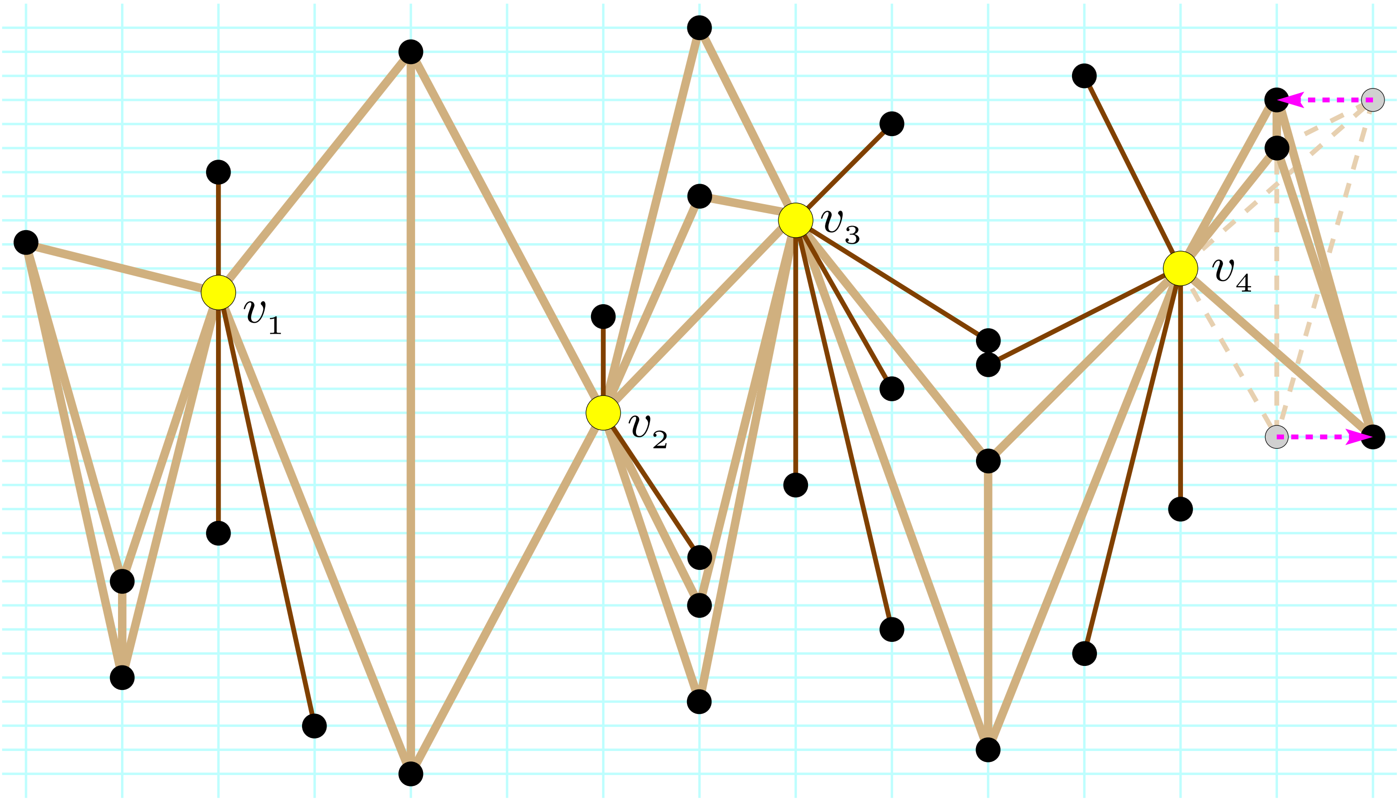
- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



- Drawing a generalized caterpillar in linear time:

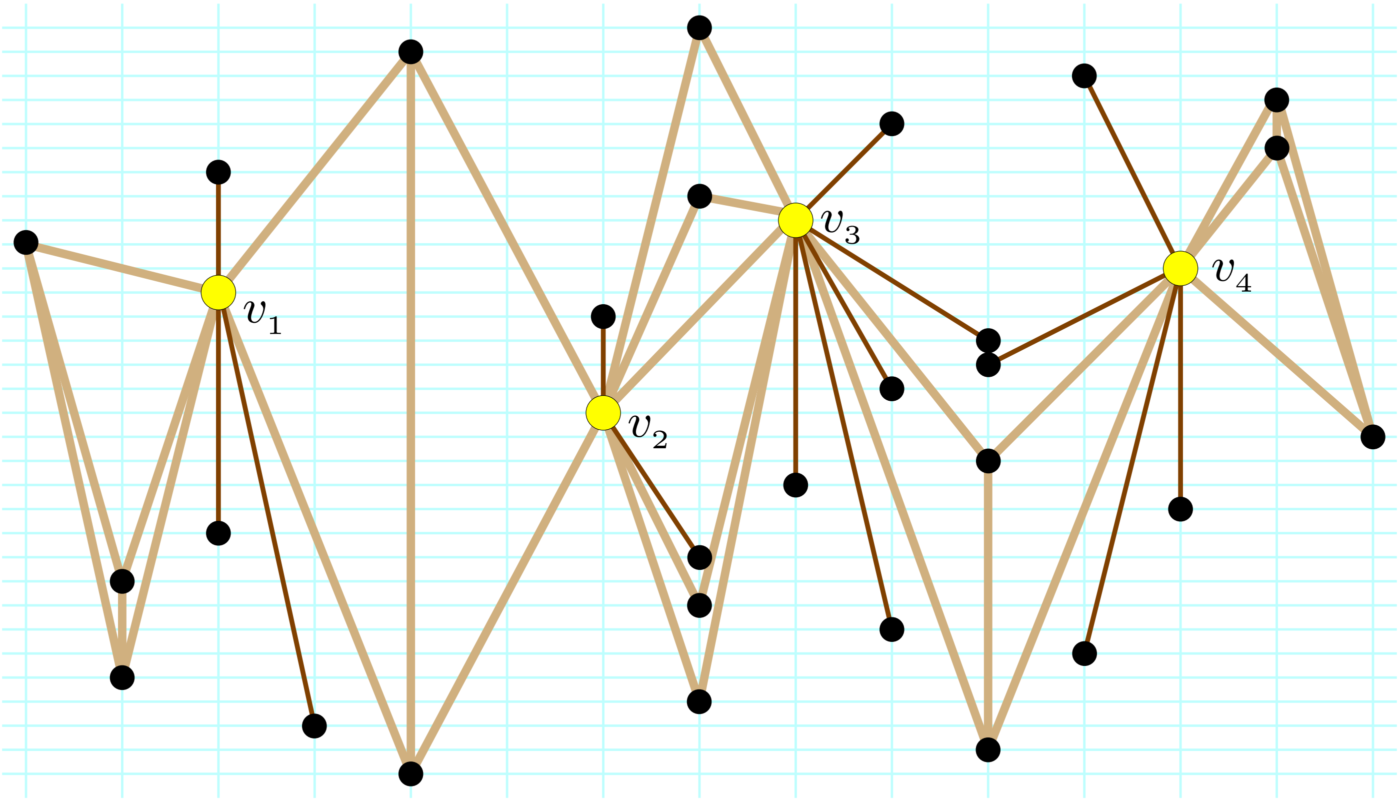
Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V,E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



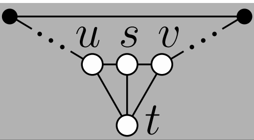
Generalized Caterpillar

- Drawing a generalized caterpillar in linear time:

Lemma 8 *A plane drawing of n-vertex generalized caterpillar $T(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

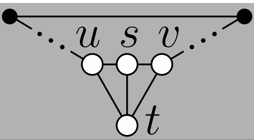


2-Connected Extended 3-Spider



- Drawing a 2-connected extended degree-3 spider:

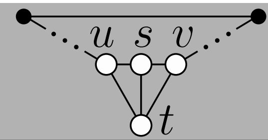
2-Connected Extended 3-Spider



- Drawing a 2-connected extended degree-3 spider:

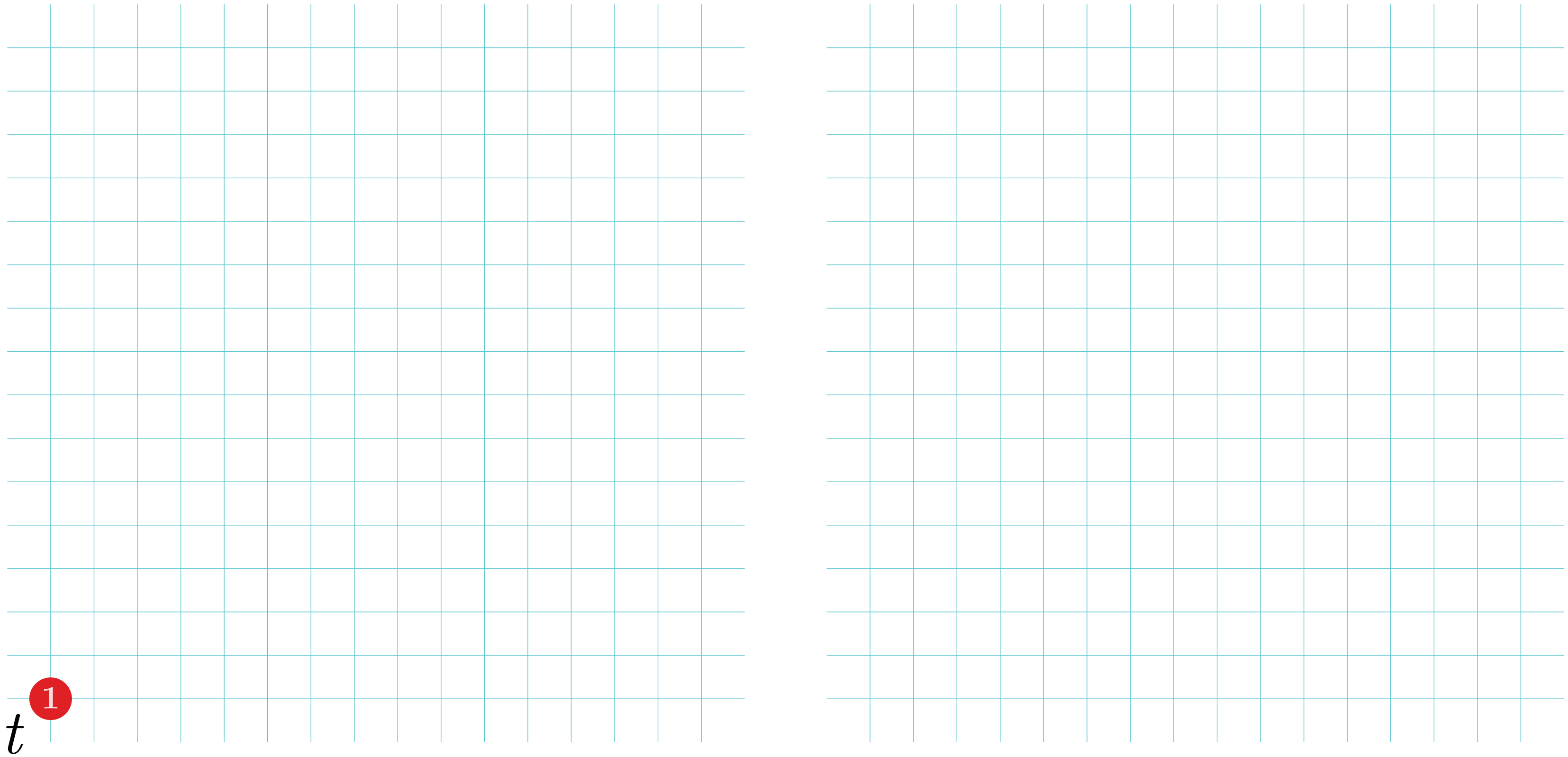
  Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

2-Connected Extended 3-Spider

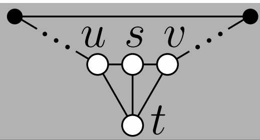


- Drawing a 2-connected extended degree-3 spider:

Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

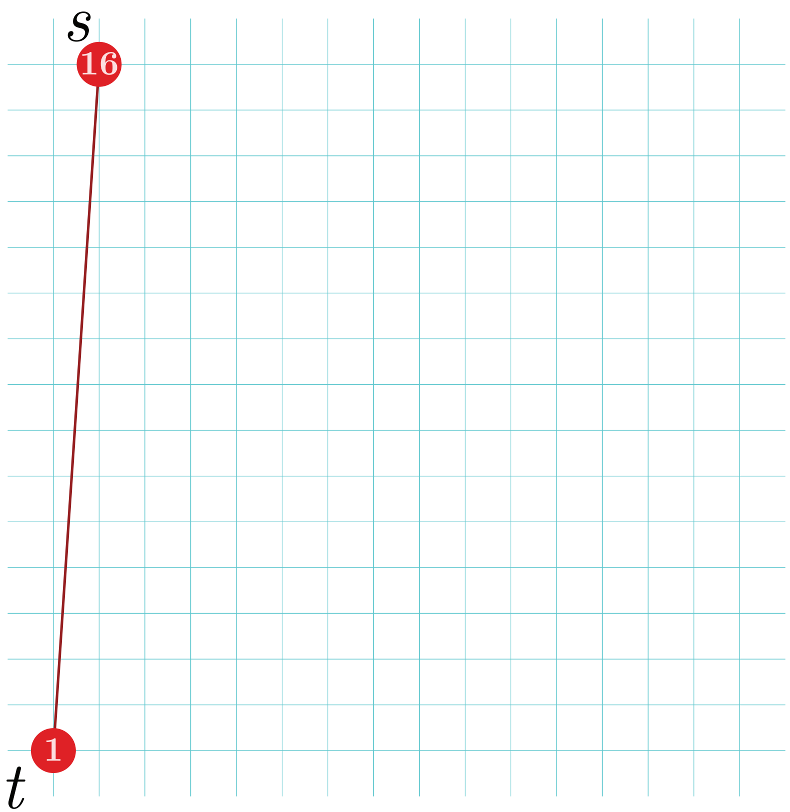


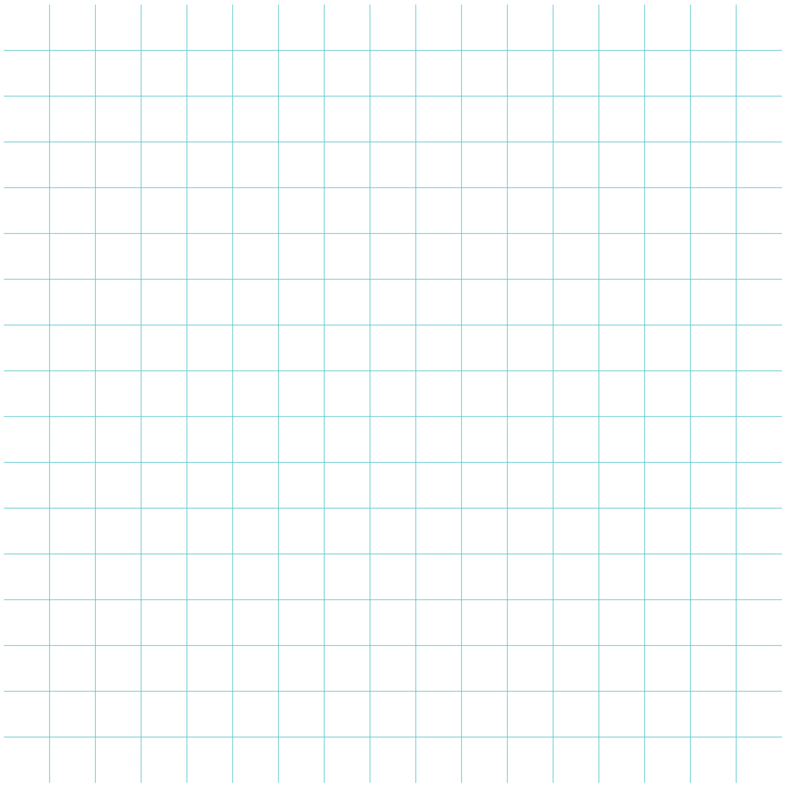
2-Connected Extended 3-Spider



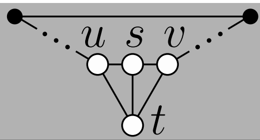
- Drawing a 2-connected extended degree-3 spider:

  Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



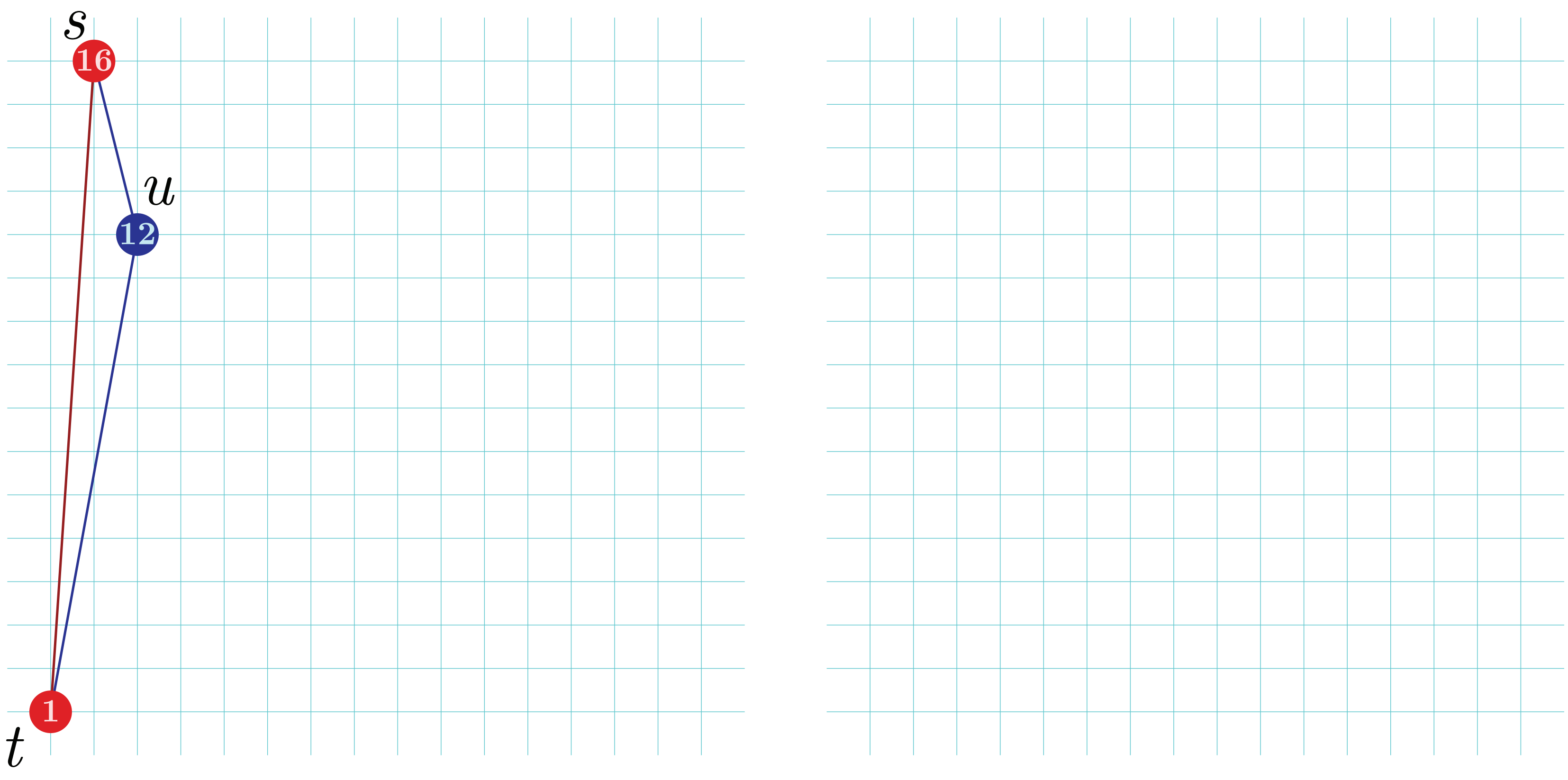


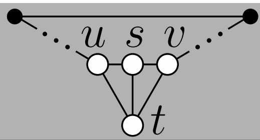
2-Connected Extended 3-Spider



- Drawing a 2-connected extended degree-3 spider:

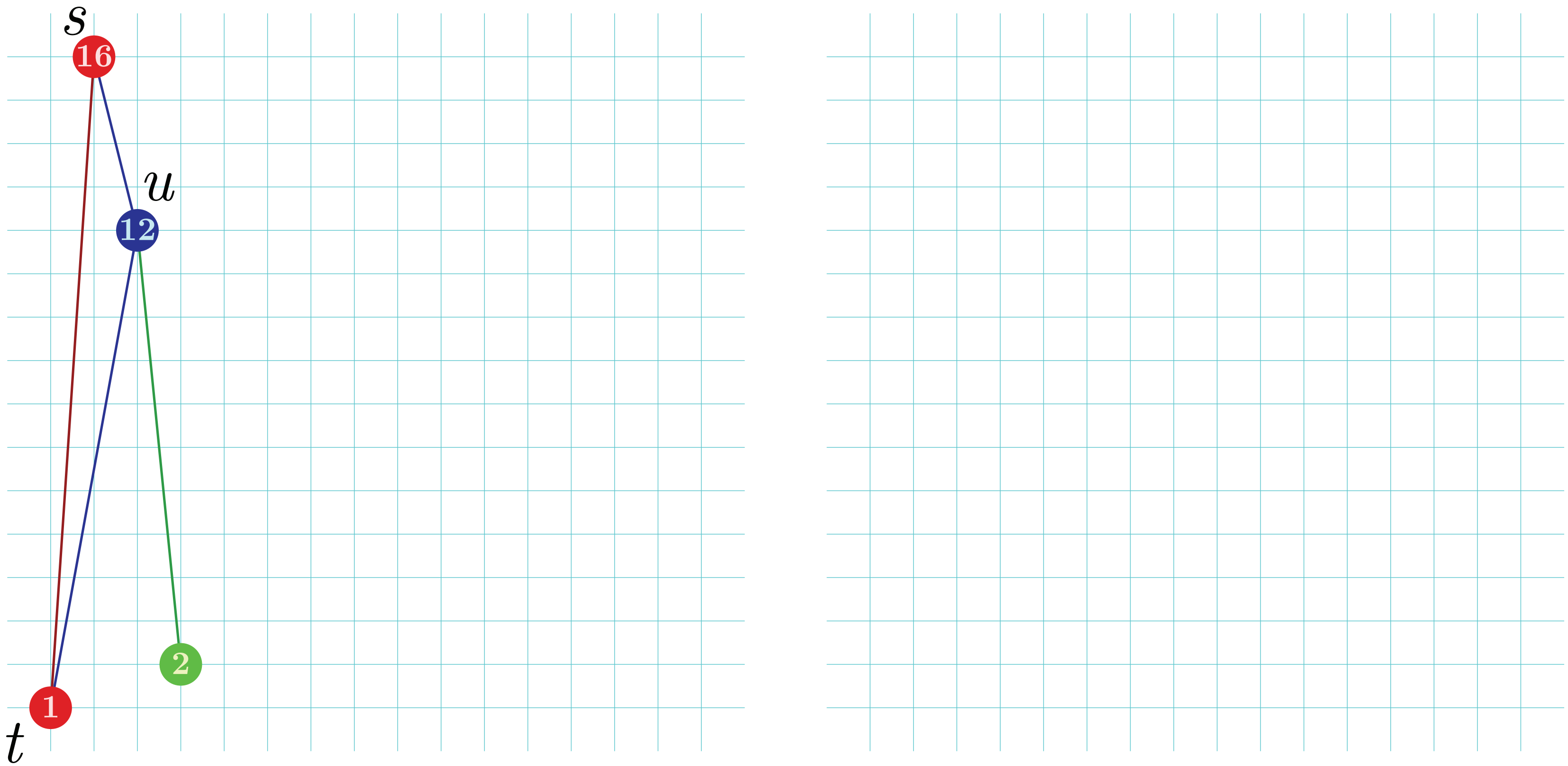
Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



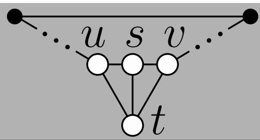


- Drawing a 2-connected extended degree-3 spider:

Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

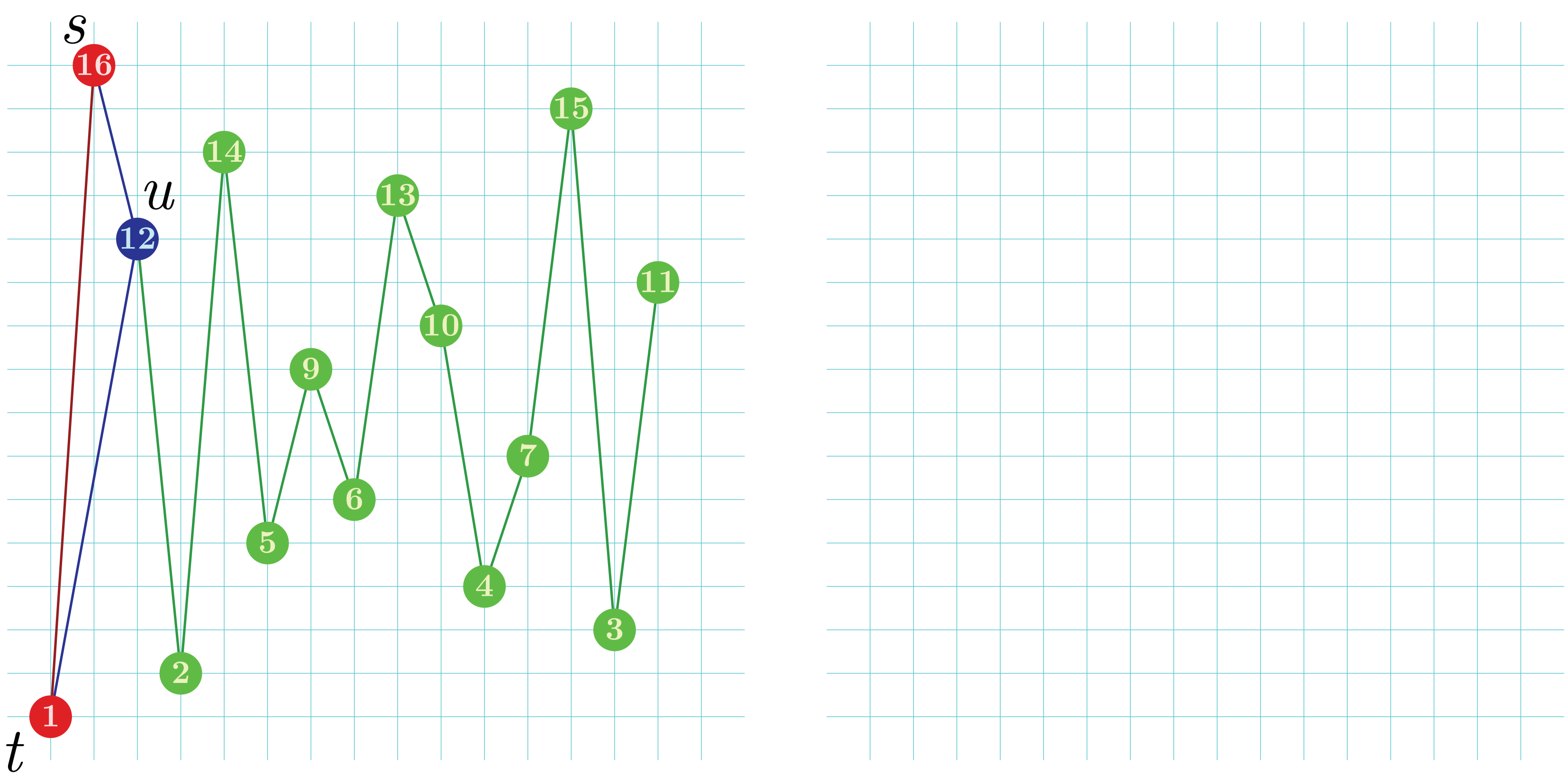


2-Connected Extended 3-Spider

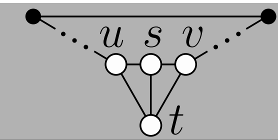


- Drawing a 2-connected extended degree-3 spider:

  Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

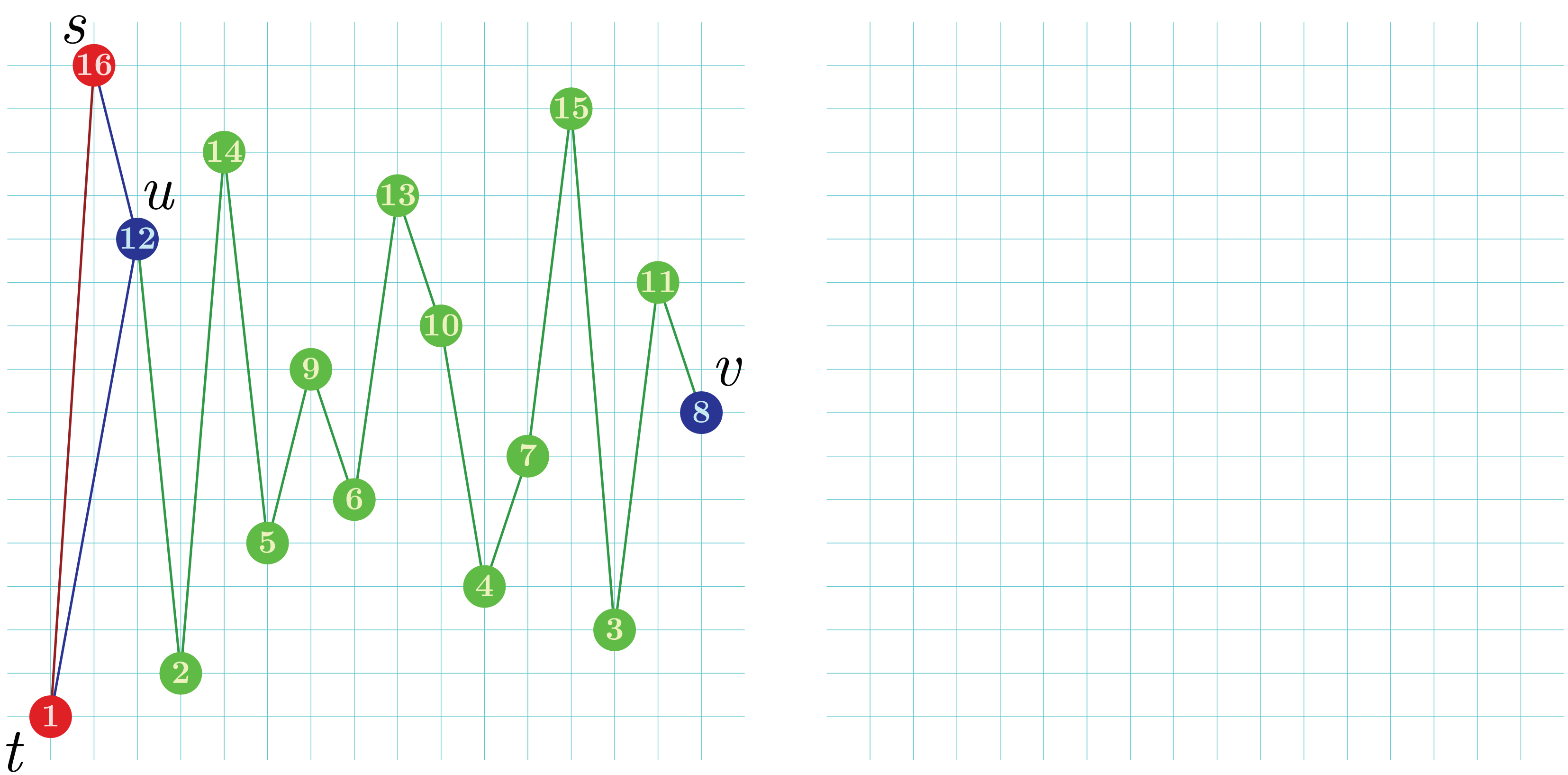


2-Connected Extended 3-Spider

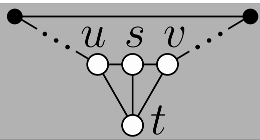


- Drawing a 2-connected extended degree-3 spider:

Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

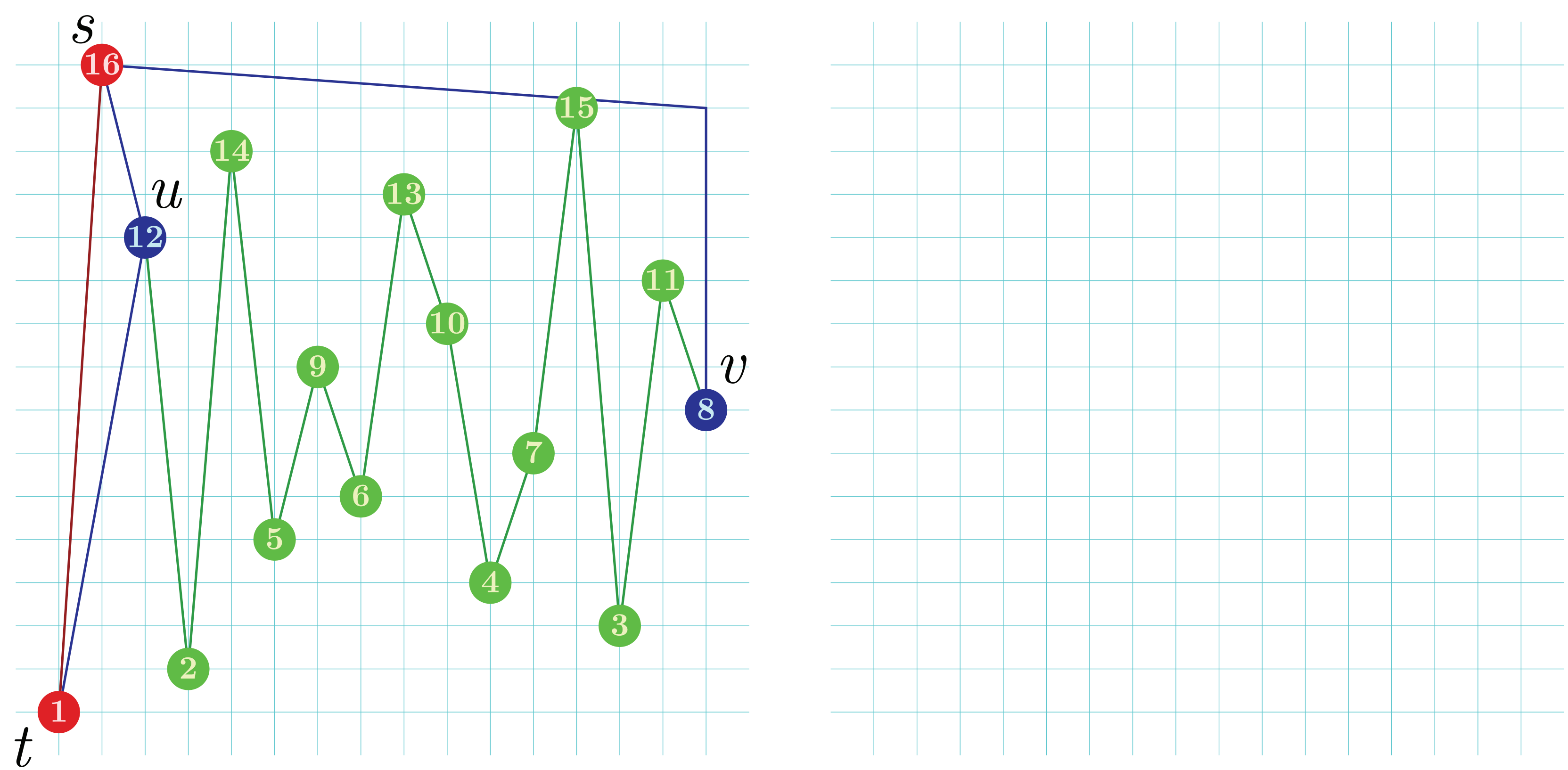


2-Connected Extended 3-Spider

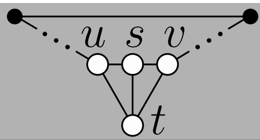


- Drawing a 2-connected extended degree-3 spider:

  Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

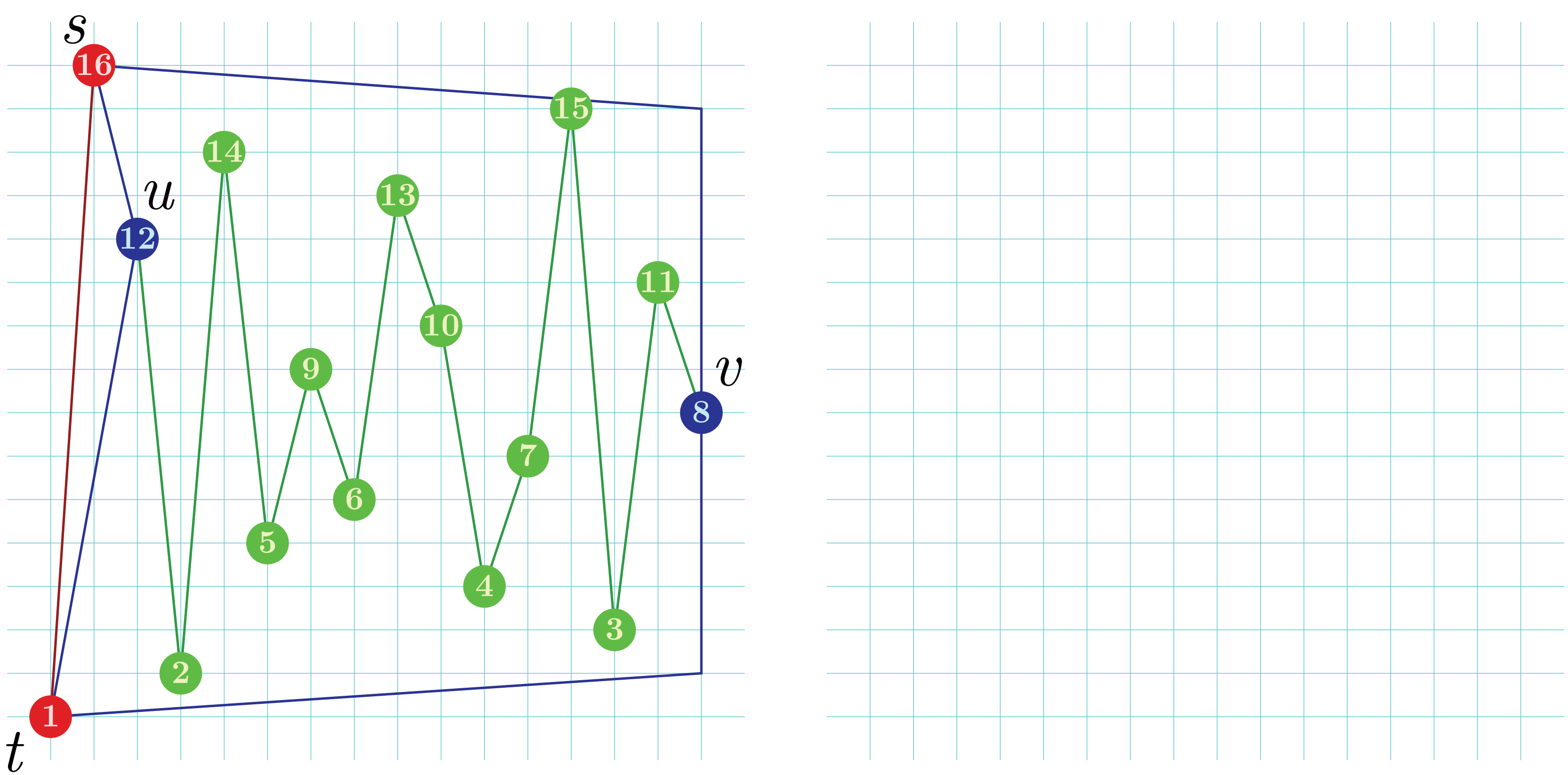


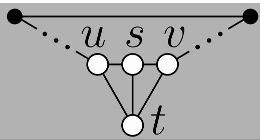
2-Connected Extended 3-Spider



- Drawing a 2-connected extended degree-3 spider:

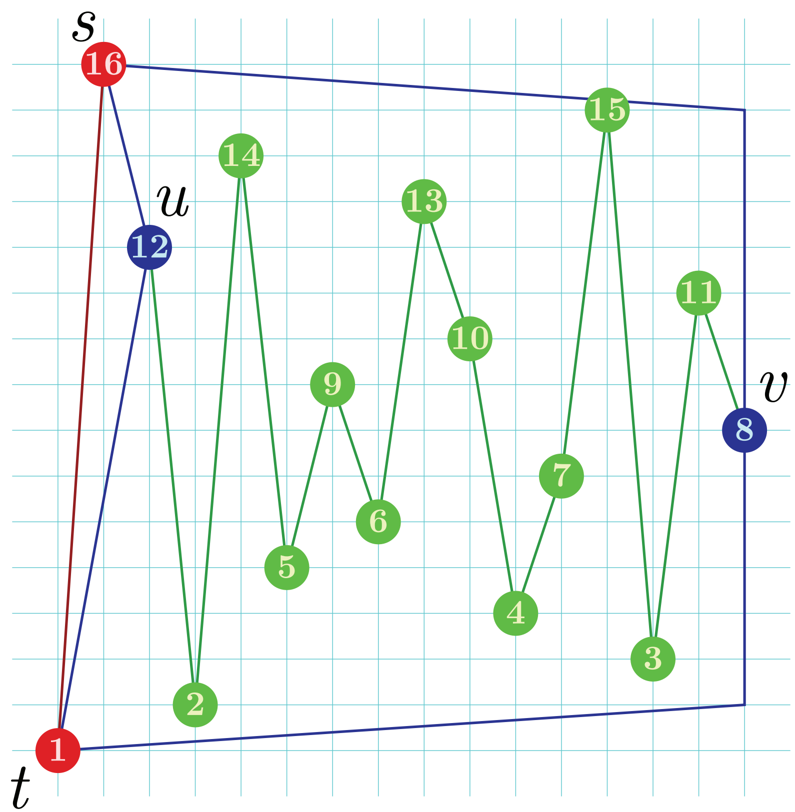
  Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

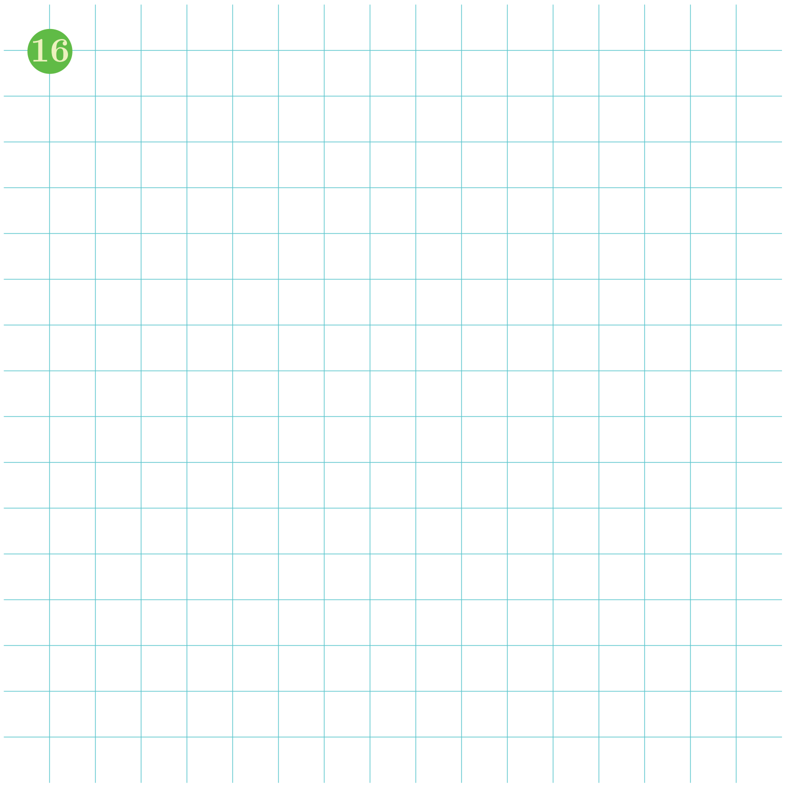




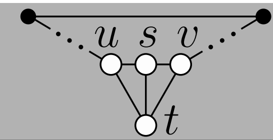
- Drawing a 2-connected extended degree-3 spider:

Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



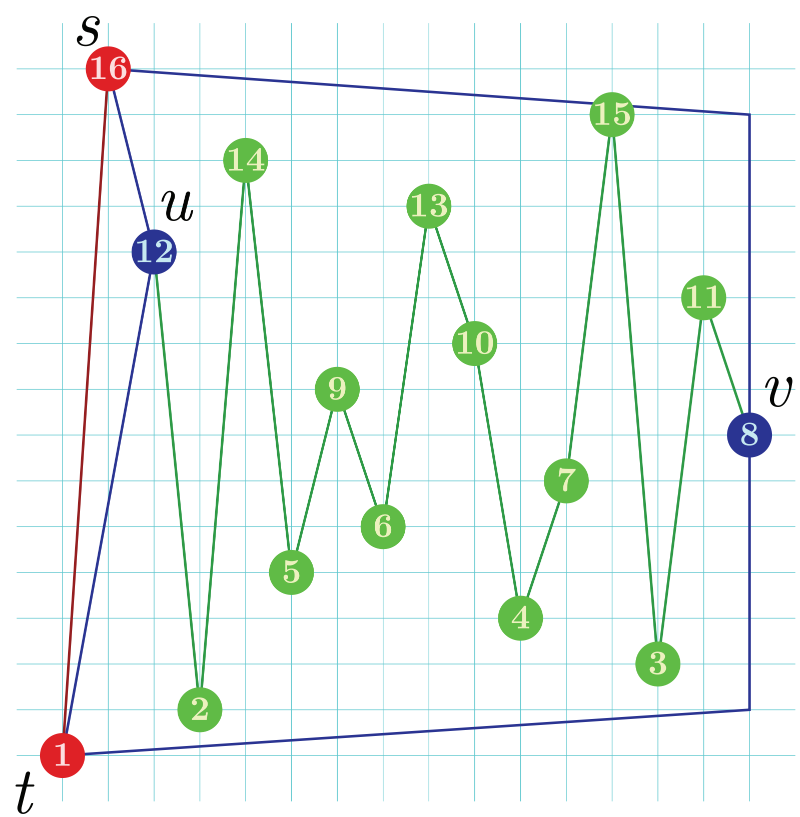


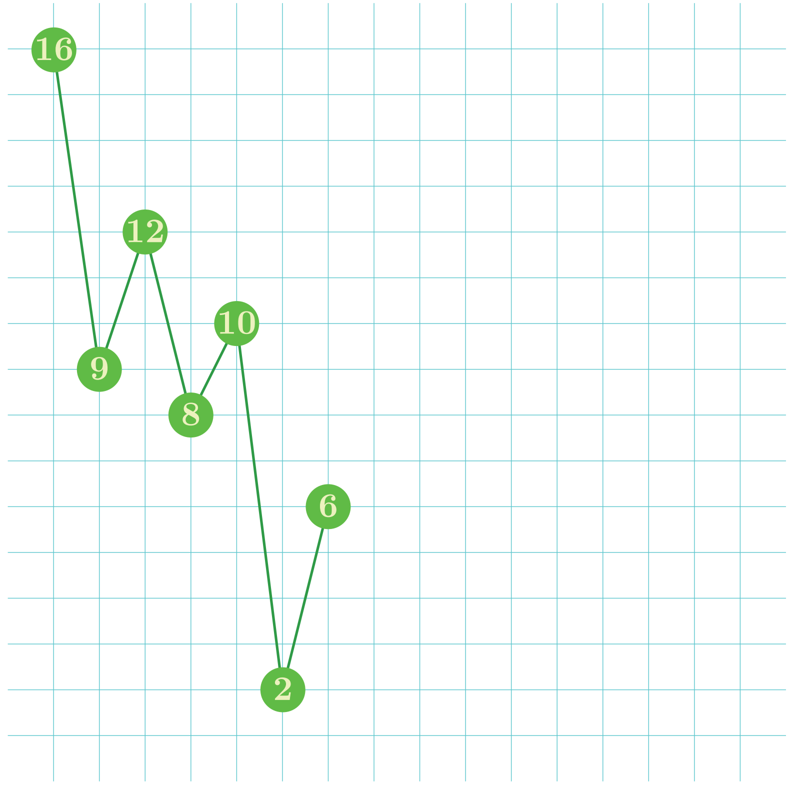
2-Connected Extended 3-Spider



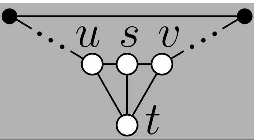
- Drawing a 2-connected extended degree-3 spider:

Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



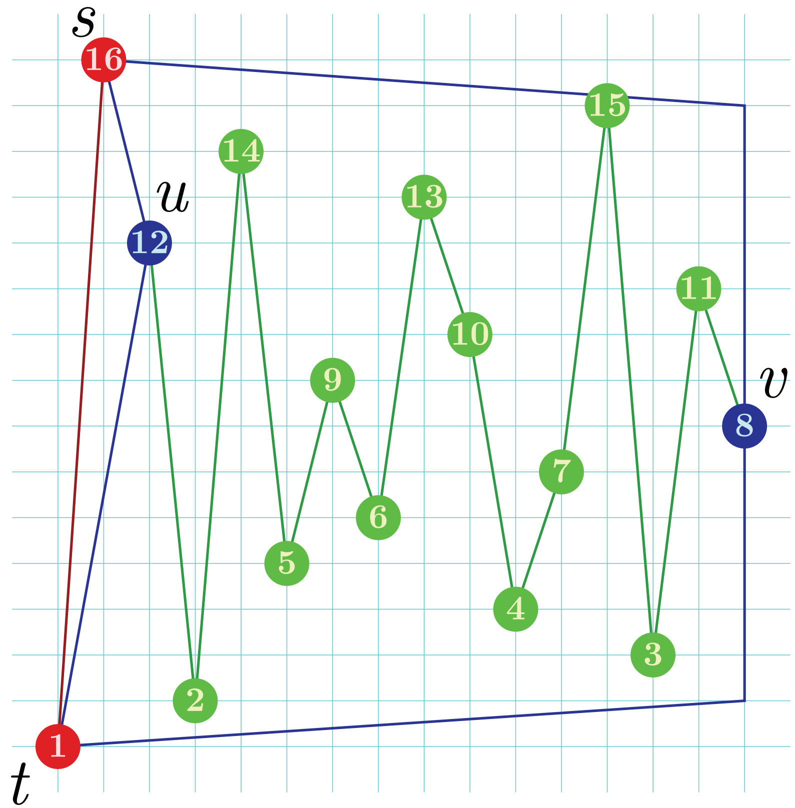


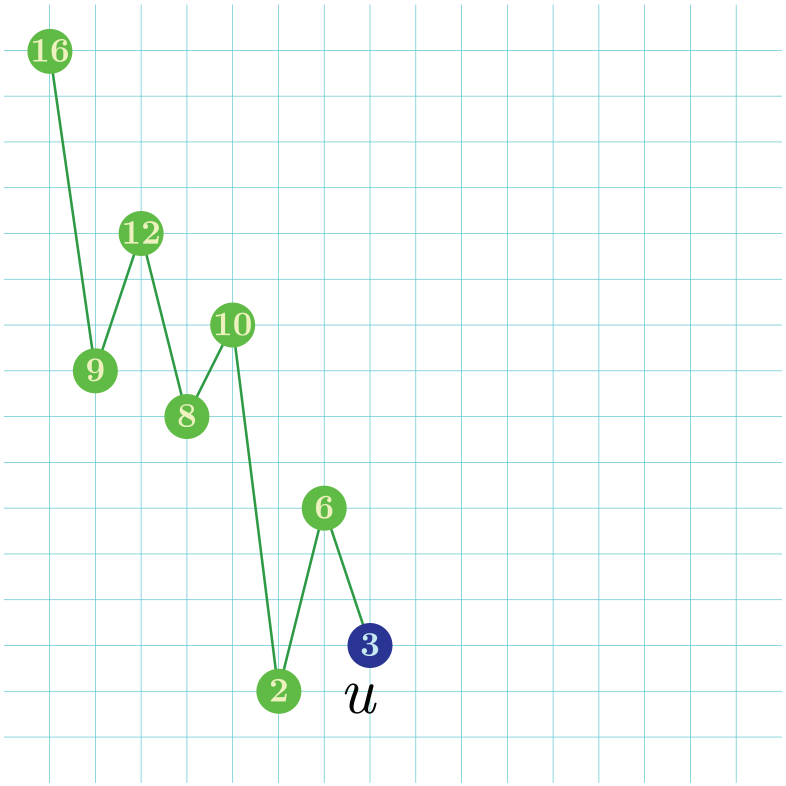
2-Connected Extended 3-Spider



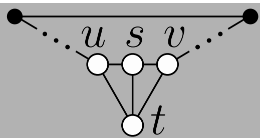
- Drawing a 2-connected extended degree-3 spider:

Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



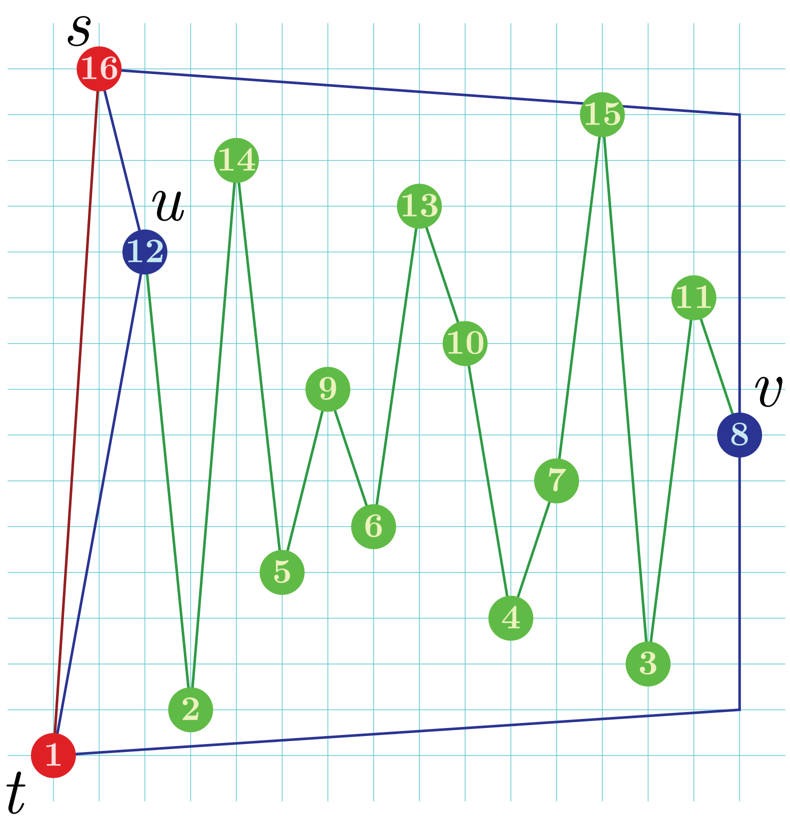


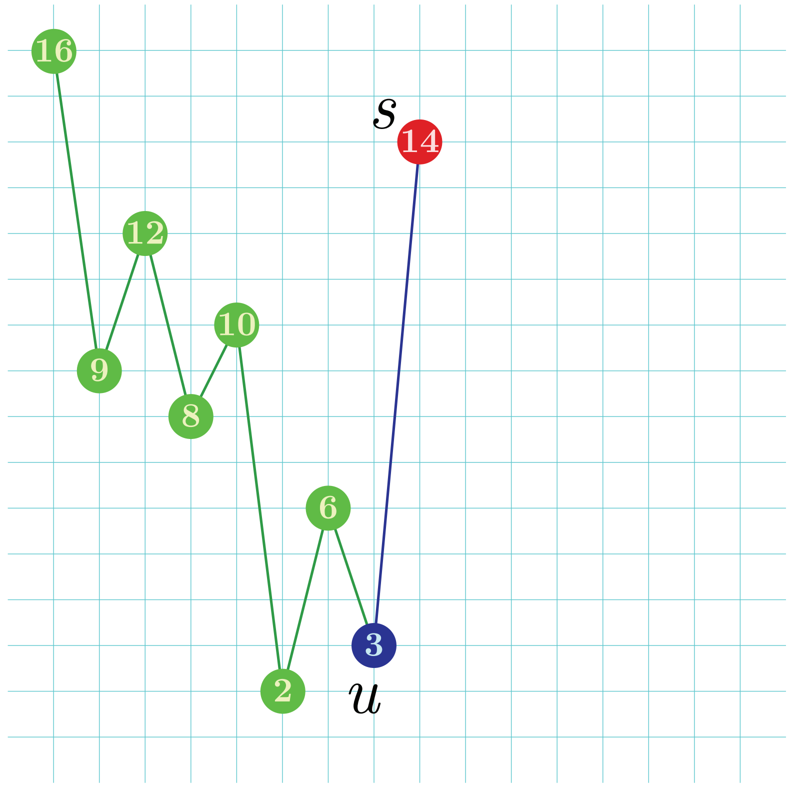
2-Connected Extended 3-Spider



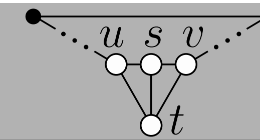
- Drawing a 2-connected extended degree-3 spider:

Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



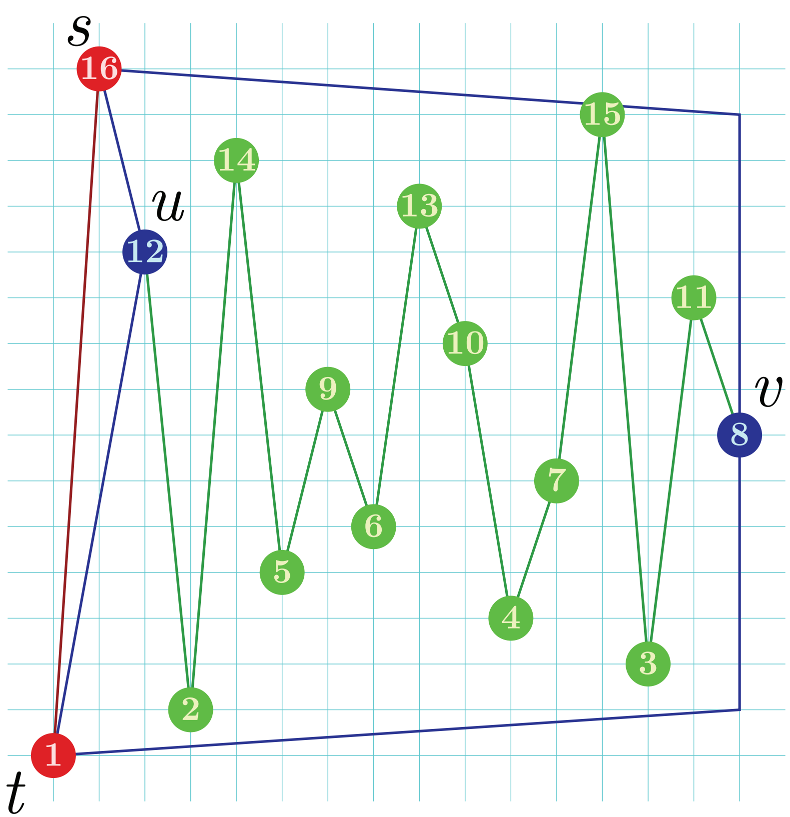


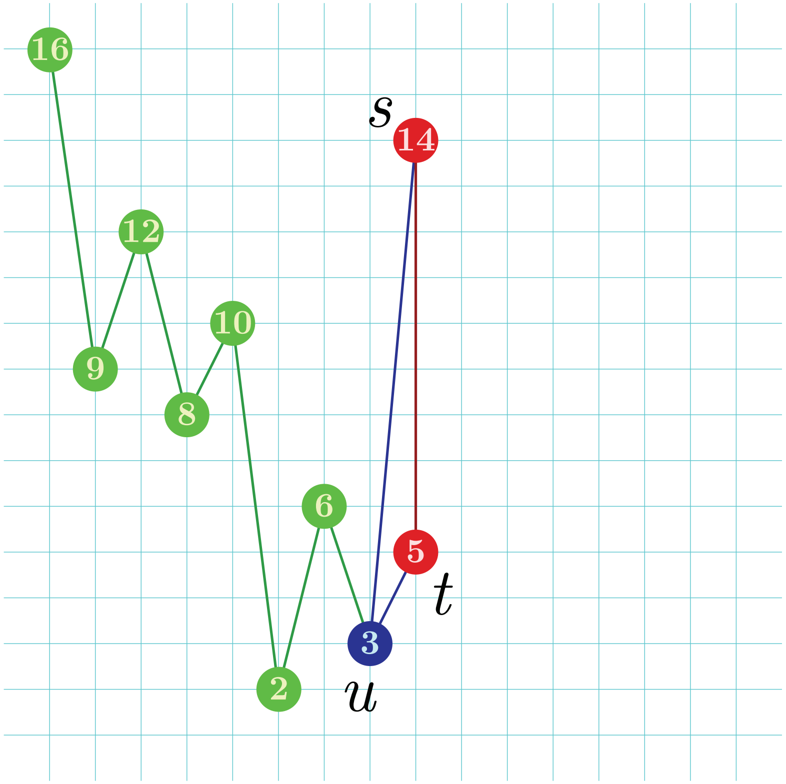
2-Connected Extended 3-Spider

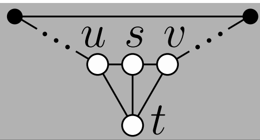


- Drawing a 2-connected extended degree-3 spider:

Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

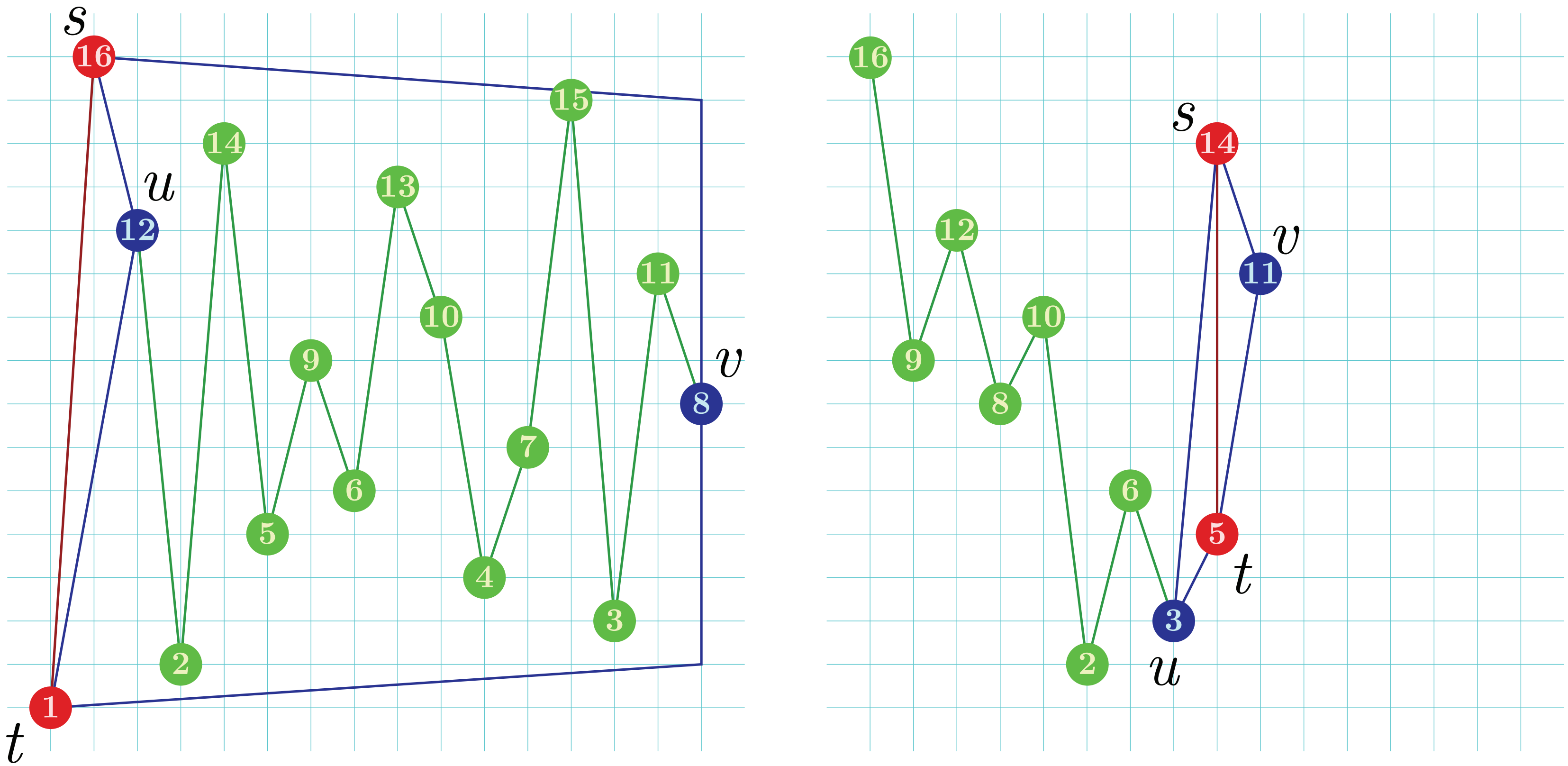




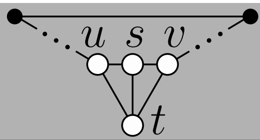


- Drawing a 2-connected extended degree-3 spider:

Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*

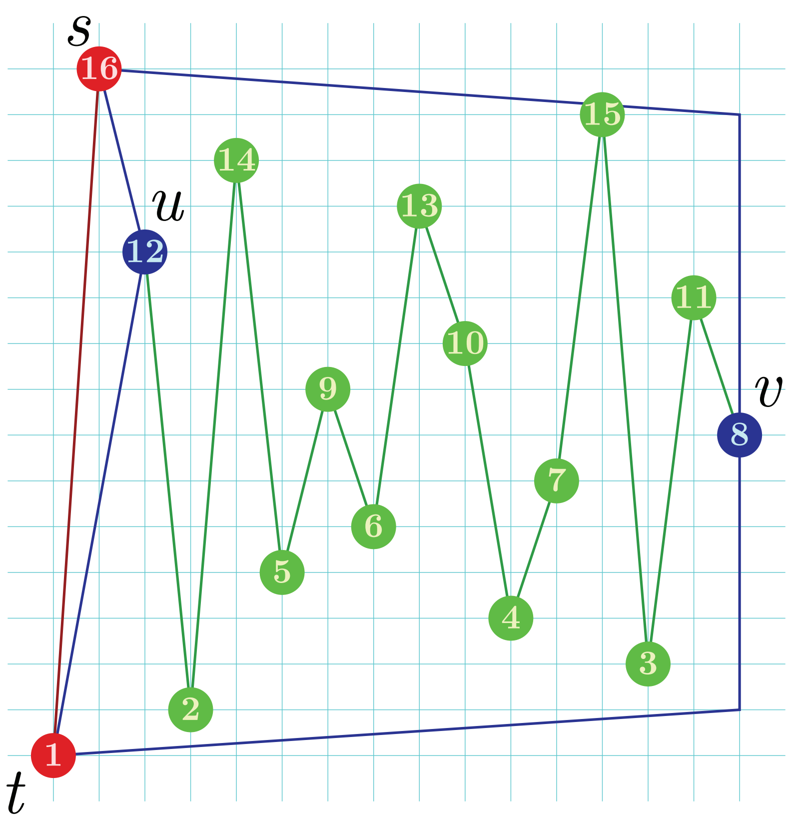


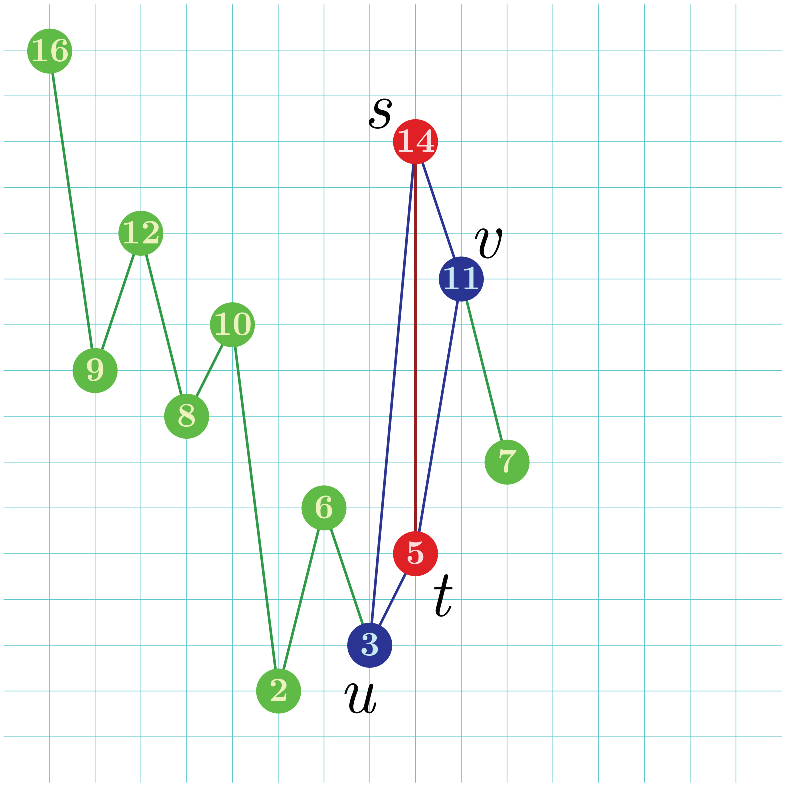
2-Connected Extended 3-Spider



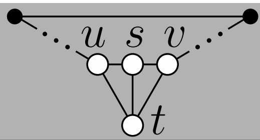
- Drawing a 2-connected extended degree-3 spider:

  Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



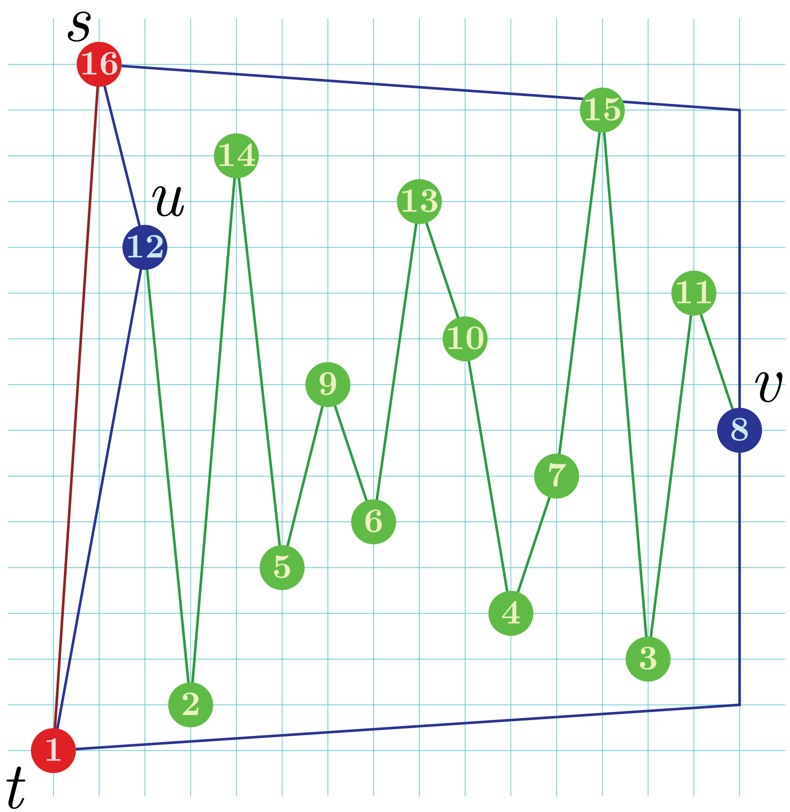


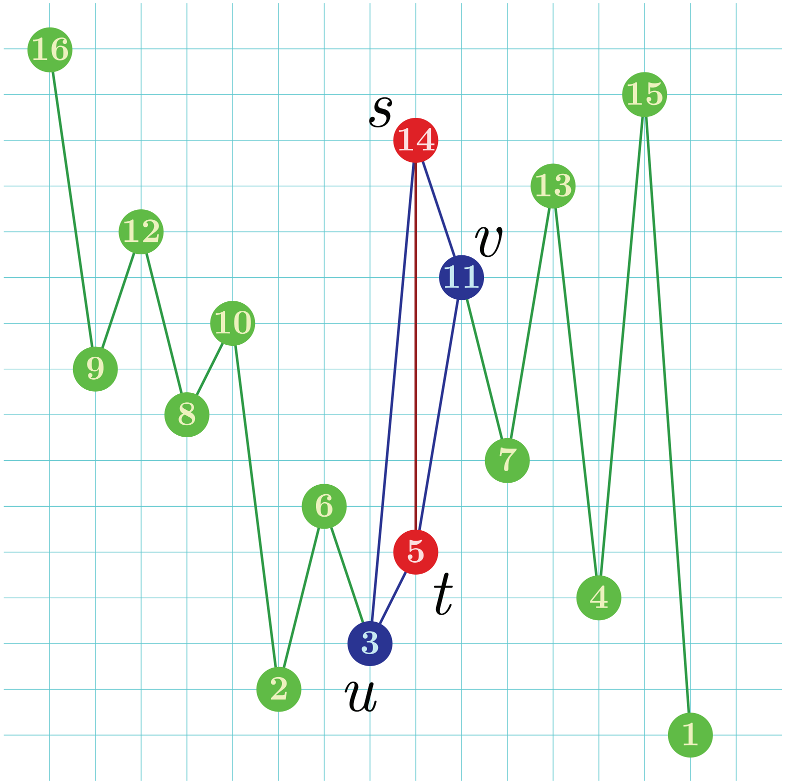
2-Connected Extended 3-Spider



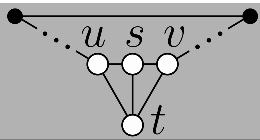
- Drawing a 2-connected extended degree-3 spider:

Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



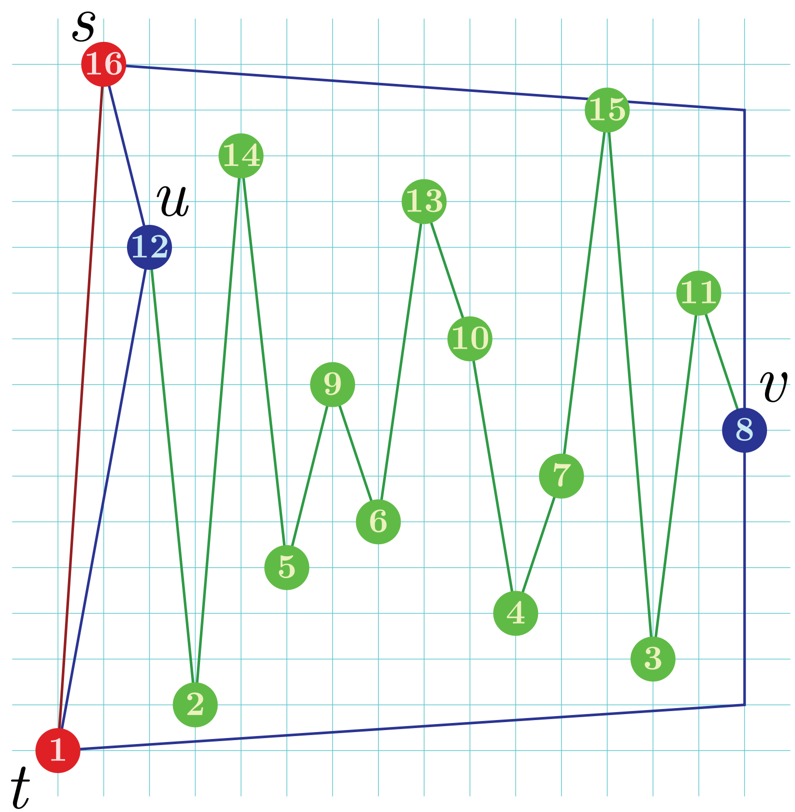


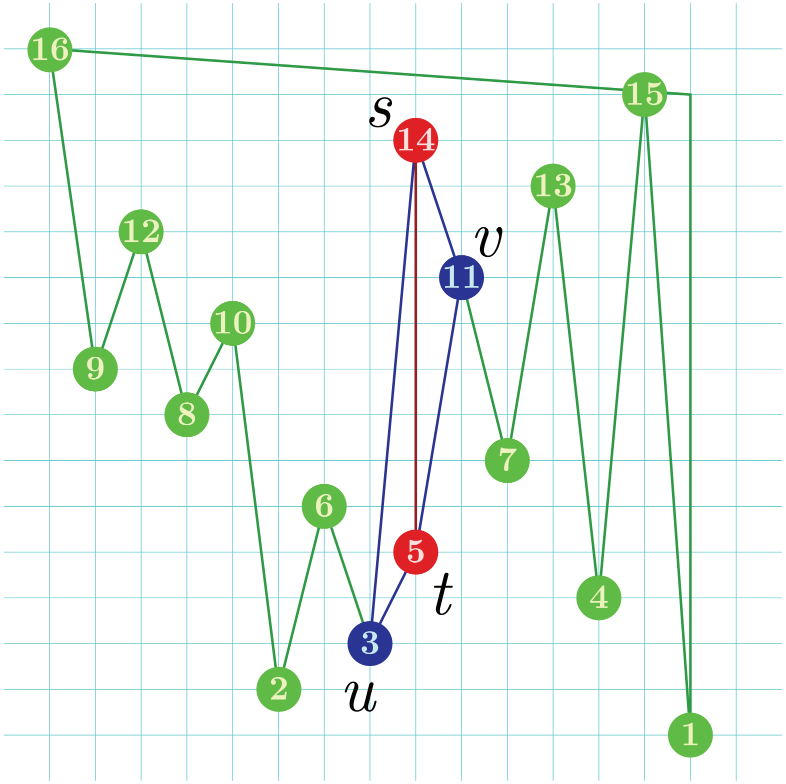
2-Connected Extended 3-Spider



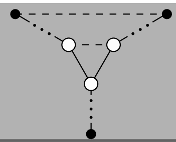
- Drawing a 2-connected extended degree-3 spider:

Lemma 9 *A plane drawing of n-vertex 2-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on a $n \times n$ grid for any leveling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$.*



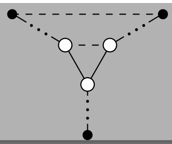


1-Connected Extended 3-Spider



- Drawing a 1-connected extended degree-3 spider:

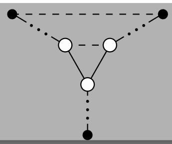
1-Connected Extended 3-Spider



- Drawing a 1-connected extended degree-3 spider:

Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

1-Connected Extended 3-Spider

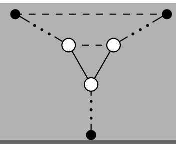


- Drawing a 1-connected extended degree-3 spider:

  Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

- Proof idea:
  - Modify degree-3 spider algorithm, same invariants

1-Connected Extended 3-Spider

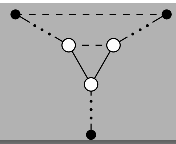


- Drawing a 1-connected extended degree-3 spider:

  Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

- Proof idea:
  - Modify degree-3 spider algorithm, same invariants
  - Break large cycle so one endpoint is extreme

1-Connected Extended 3-Spider



- Drawing a 1-connected extended degree-3 spider:

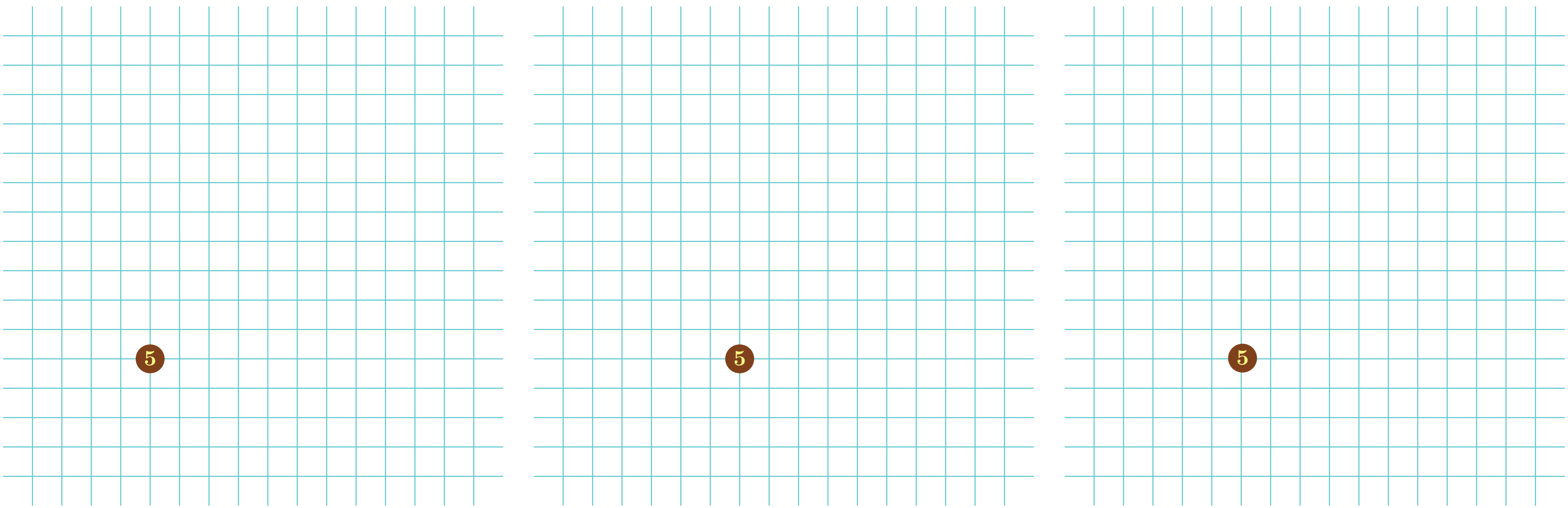
  Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

- Proof idea:
  - Modify degree-3 spider algorithm, same invariants
  - Break large cycle so one endpoint is extreme
  - Getting degree-3 spider started is more involved

1-Connected Extended 3-Spider

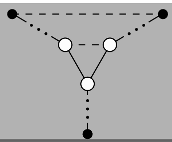
- Drawing a 1-connected extended degree-3 spider:

  Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



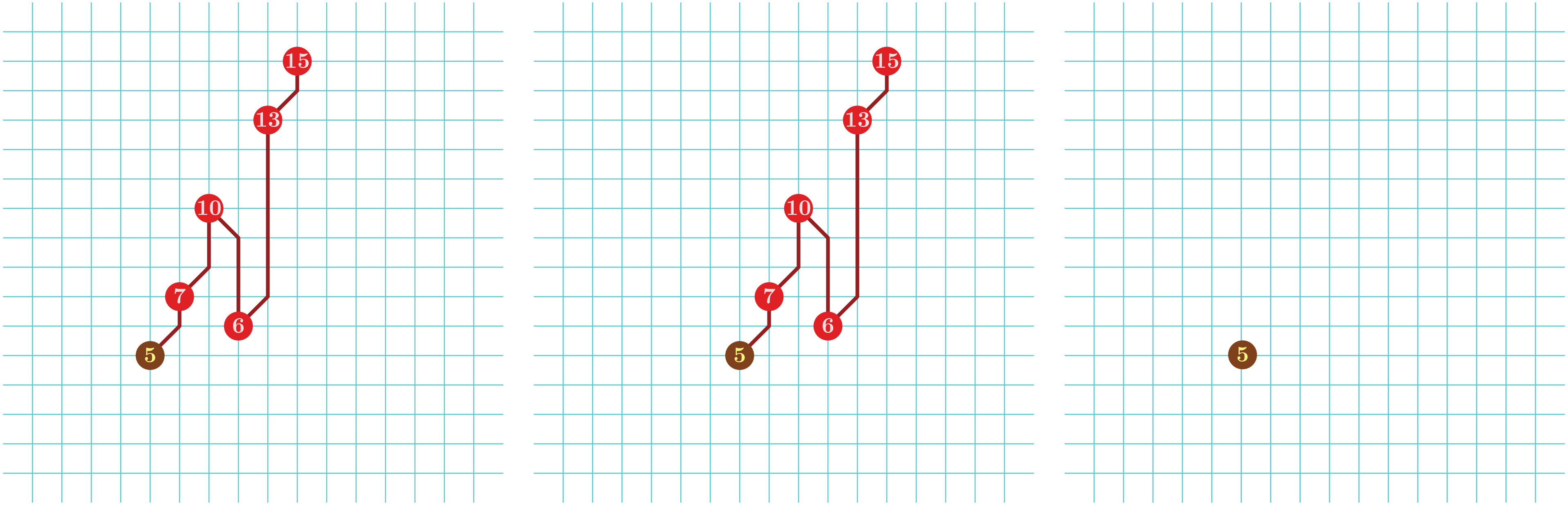
  - Two cases of starting spider for extra edge that forms a K_3

1-Connected Extended 3-Spider



- Drawing a 1-connected extended degree-3 spider:

  Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

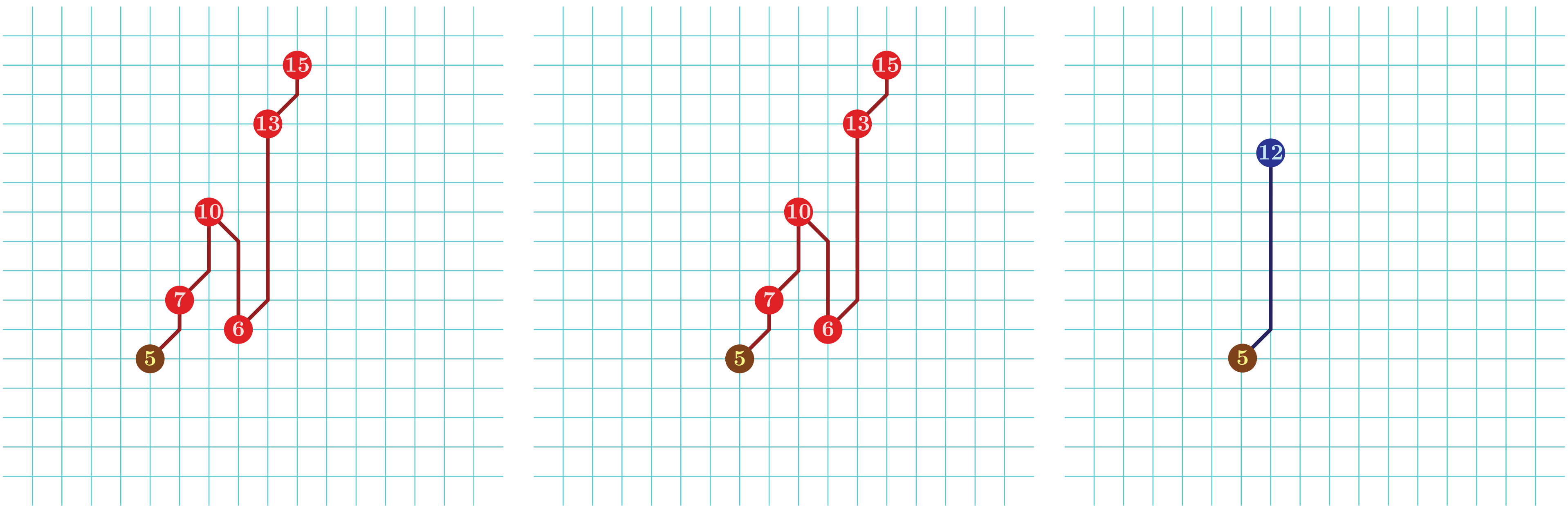


  - In the first case red path is first drawn to its maximum

1-Connected Extended 3-Spider

- Drawing a 1-connected extended degree-3 spider:

  Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

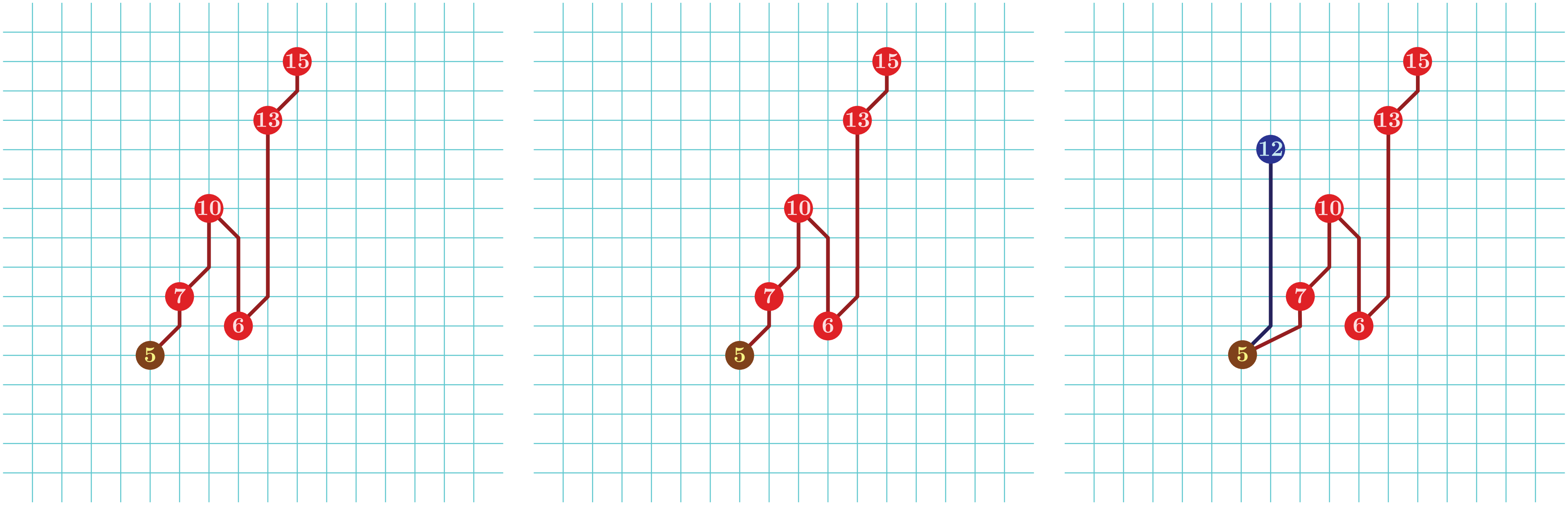


  - In the second case blue path is drawn first

1-Connected Extended 3-Spider

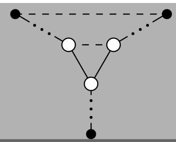
- Drawing a 1-connected extended degree-3 spider:

  Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



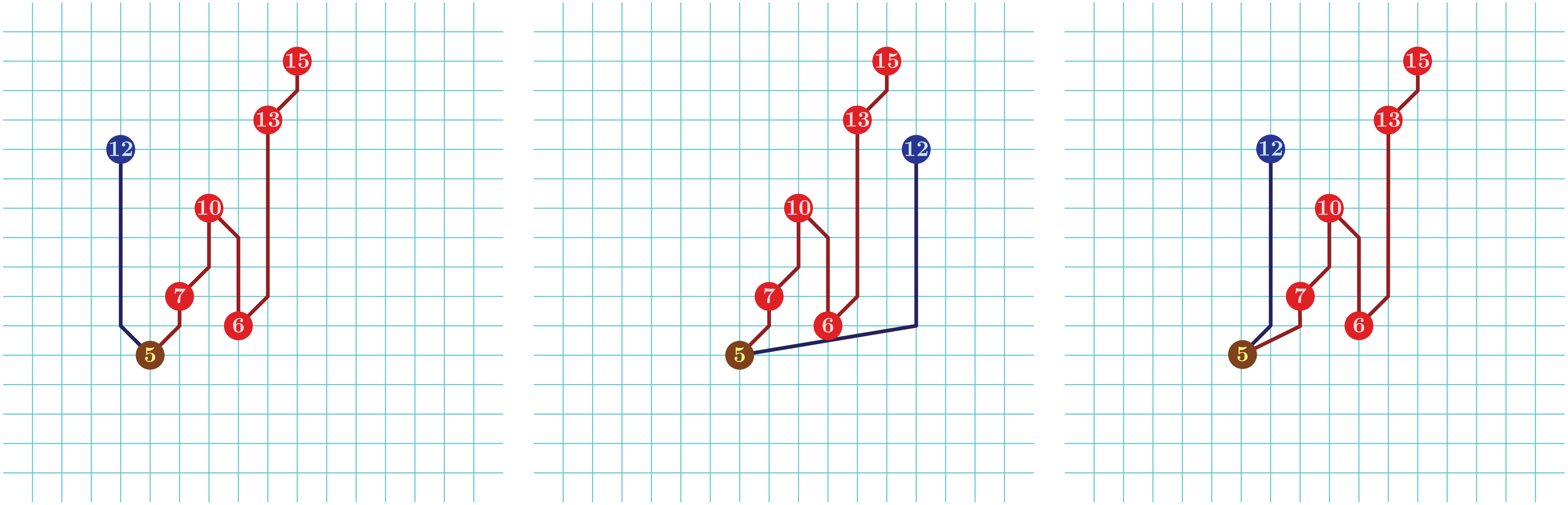
  - Then the red path is drawn next

1-Connected Extended 3-Spider



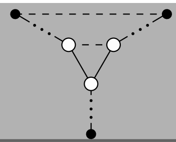
- Drawing a 1-connected extended degree-3 spider:

  Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



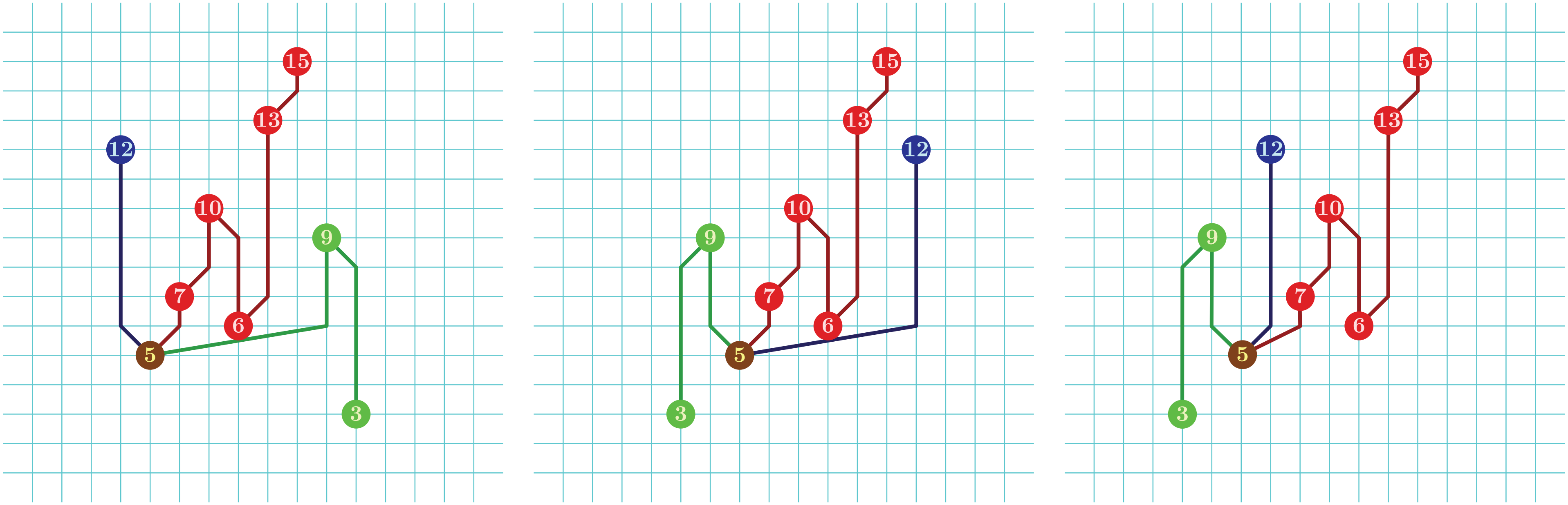
  - Whereas in the first case the blue path comes second

1-Connected Extended 3-Spider



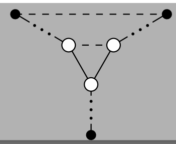
- Drawing a 1-connected extended degree-3 spider:

  Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



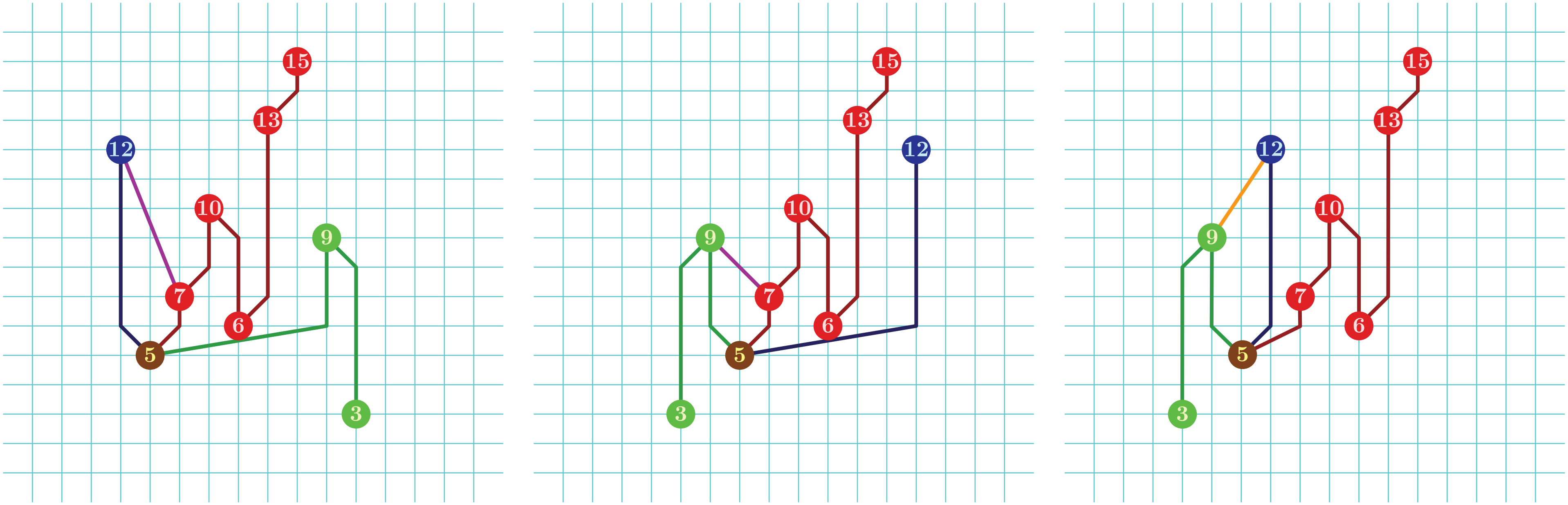
  - Then the green paths are drawn to their minimum

1-Connected Extended 3-Spider



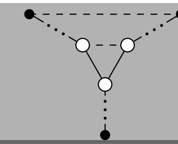
- Drawing a 1-connected extended degree-3 spider:

  Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*



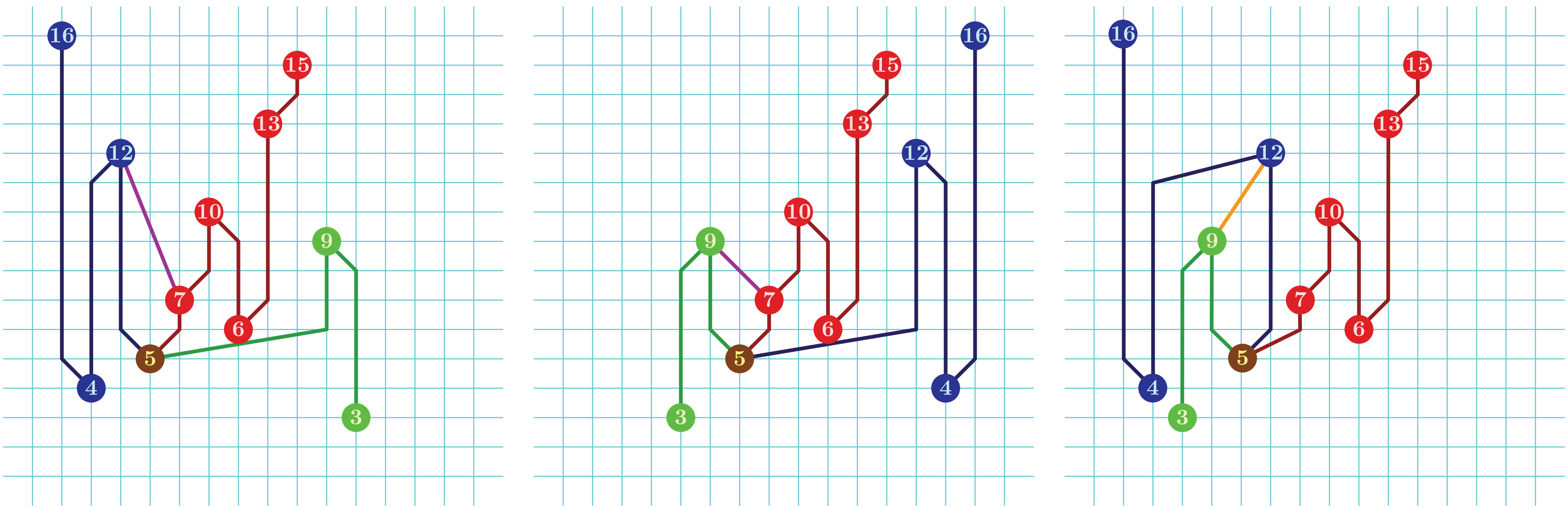
  - Can then accommodate the extra edge

1-Connected Extended 3-Spider



- Drawing a 1-connected extended degree-3 spider:

  Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

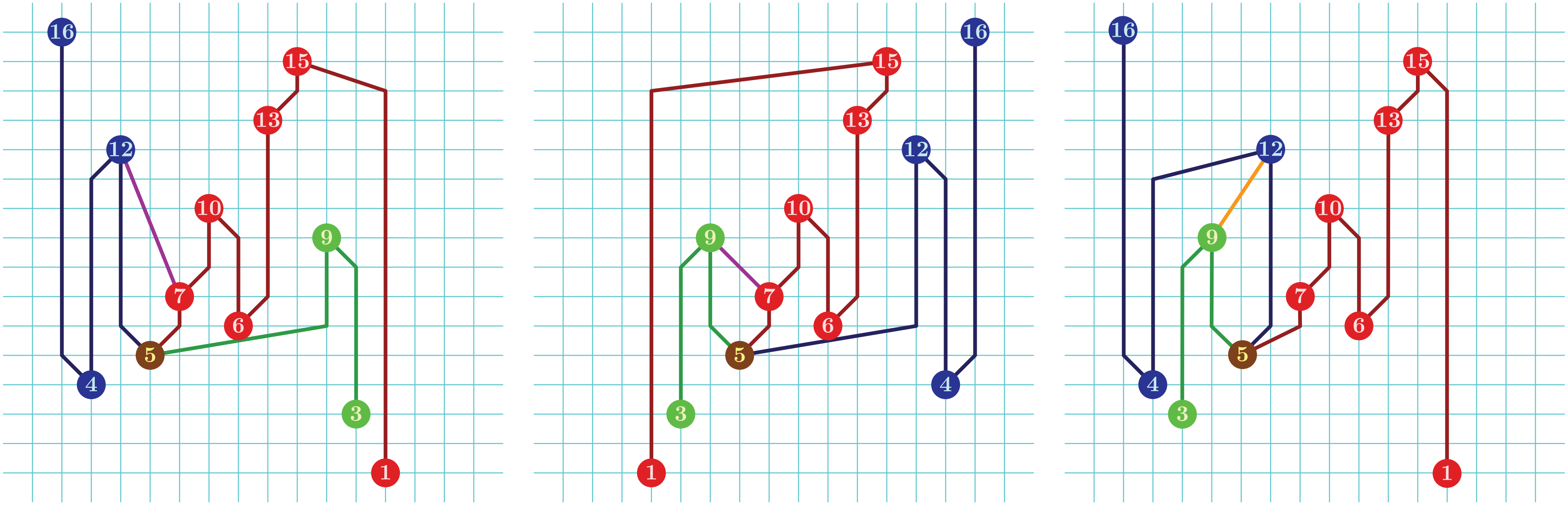


  - Can then accommodate the extra edge

1-Connected Extended 3-Spider

- Drawing a 1-connected extended degree-3 spider:

  Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

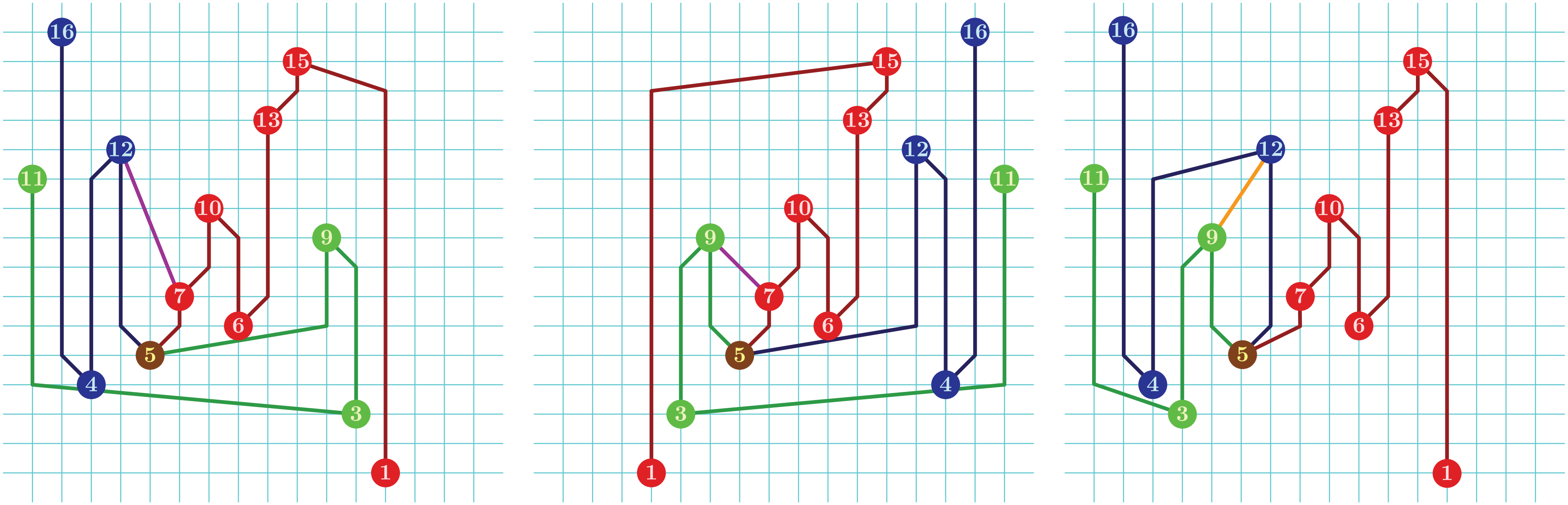


  - Can then accommodate the extra edge

1-Connected Extended 3-Spider

- Drawing a 1-connected extended degree-3 spider:

Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

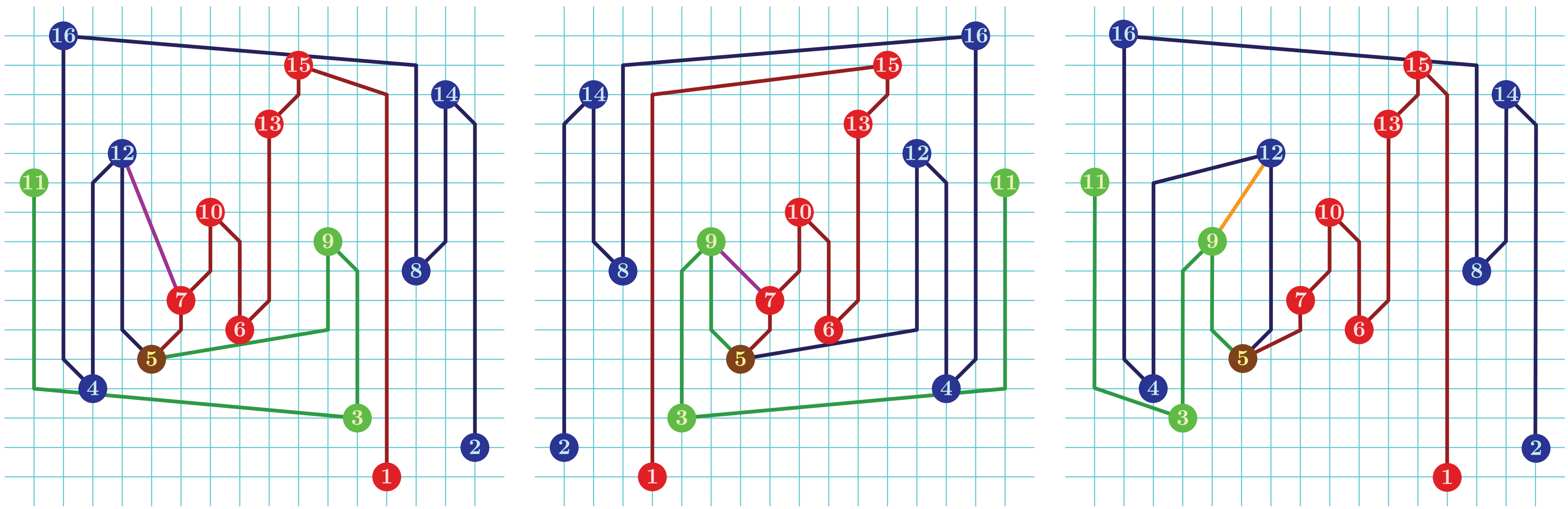


  - Can then accommodate the extra edge

1-Connected Extended 3-Spider

- Drawing a 1-connected extended degree-3 spider:

  Lemma 10 *A planar drawing of an n-vertex 1-connected extended degree-3 spider $G(V, E)$ can be drawn in $O(n)$ time on an $n \times n$ grid for any vertex labeling $\phi : V \xrightarrow[\text{onto}]{1:1} \{1, 2, \ldots, n\}$ with one bend per edge.*

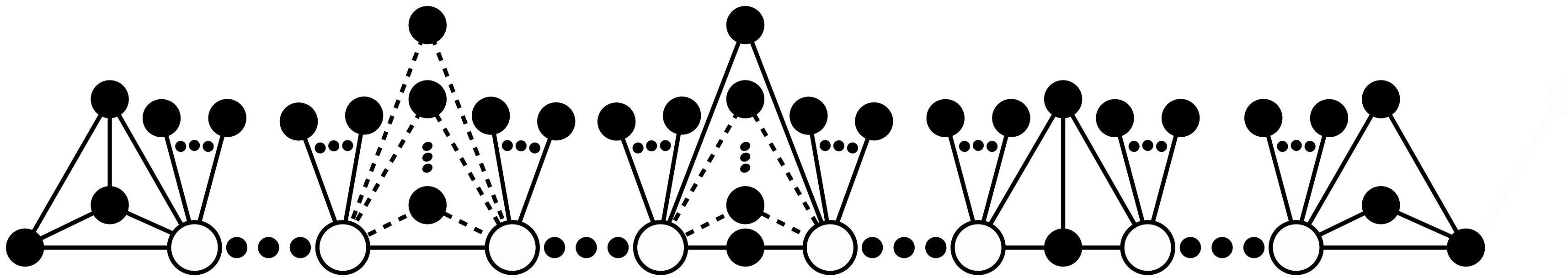


  - ► Can then accommodate the extra edge

Characterization – ULP Graphs

- Proof by induction gives the next theorem:

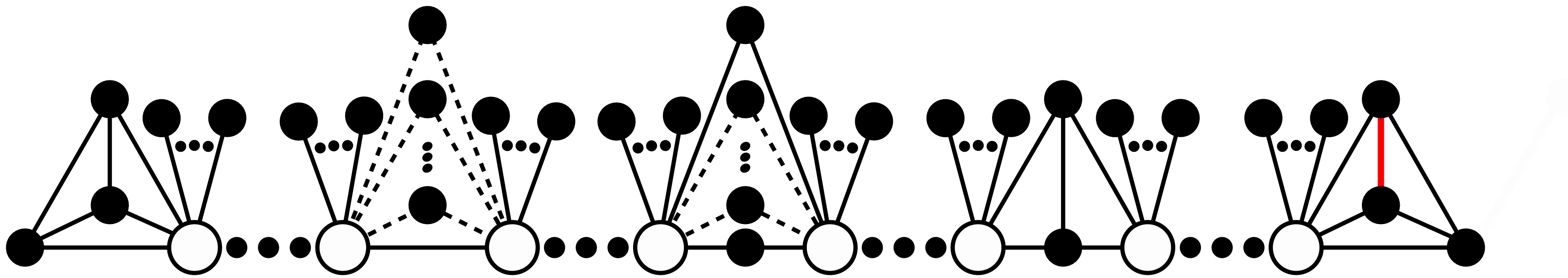
  Theorem 11 *Every graph either contains one of the seven forbidden graphs in which case it is not ULP, or it is a generalized caterpillar, a radius-2 star, or a extended degree-3 spider in which case it is ULP.*



- Proof by induction gives the next theorem:

  Theorem 11 *Every graph either contains one of the seven forbidden graphs in which case it is not ULP, or it is a generalized caterpillar, a radius-2 star, or a extended degree-3 spider in which case it is ULP.*

- Proof idea:
  - Assume graph with n edges is ULP

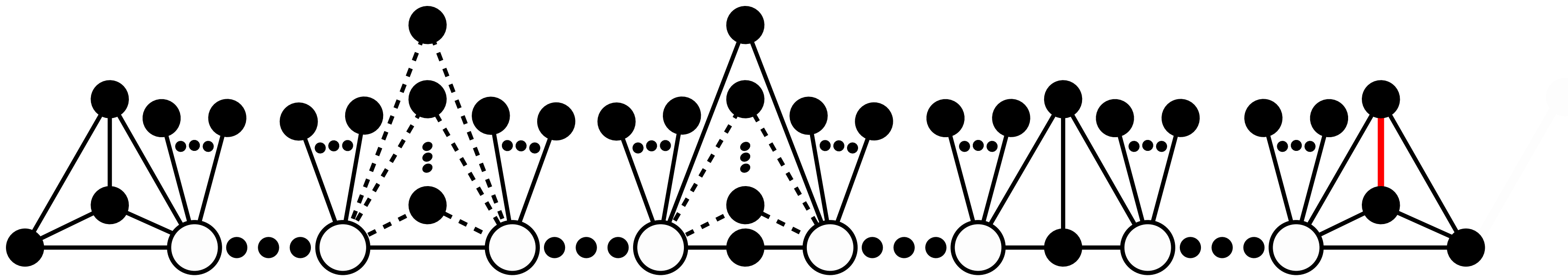


- Proof by induction gives the next theorem:

  Theorem 11 *Every graph either contains one of the seven forbidden graphs in which case it is not ULP, or it is a generalized caterpillar, a radius-2 star, or a extended degree-3 spider in which case it is ULP.*

- Proof idea:
  - Consider all possible ways of adding an extra edge

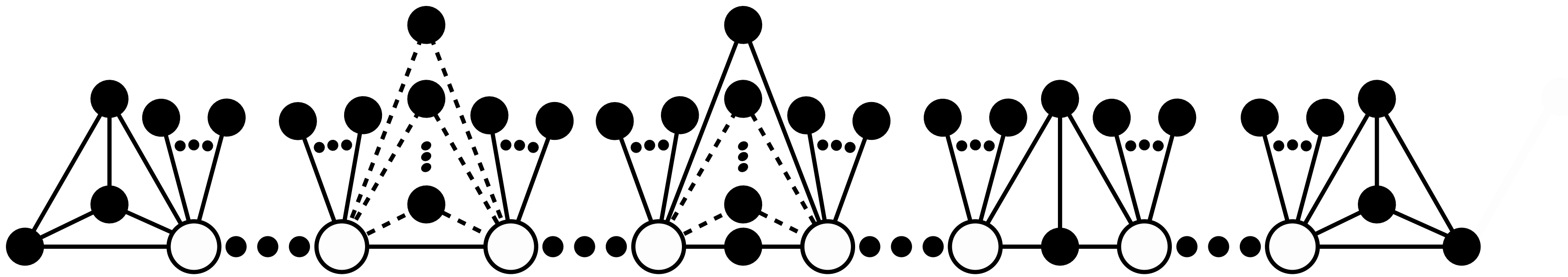
Characterization – ULP Graphs



- Proof by induction gives the next theorem:

Theorem 11 *Every graph either contains one of the seven forbidden graphs in which case it is not ULP, or it is a generalized caterpillar, a radius-2 star, or a extended degree-3 spider in which case it is ULP.*

- Proof idea:
  - Either two possibilities

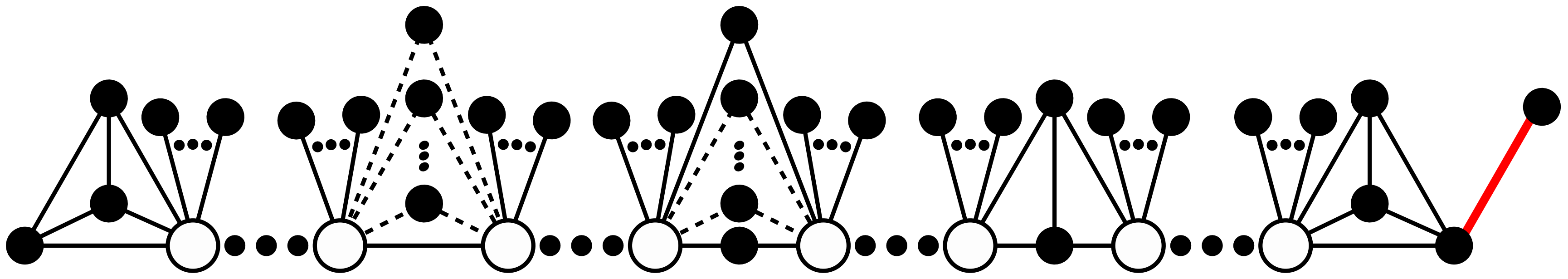


- Proof by induction gives the next theorem:

  Theorem 11 *Every graph either contains one of the seven forbidden graphs in which case it is not ULP, or it is a generalized caterpillar, a radius-2 star, or a extended degree-3 spider in which case it is ULP.*

- Proof idea:
  - Either two possibilities
    - Graph remains ULP

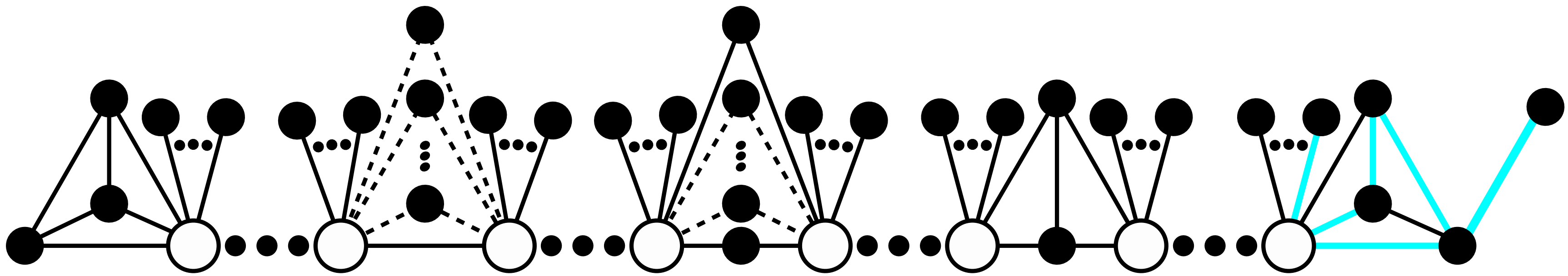
Characterization – ULP Graphs



- Proof by induction gives the next theorem:

Theorem 11 *Every graph either contains one of the seven forbidden graphs in which case it is not ULP, or it is a generalized caterpillar, a radius-2 star, or a extended degree-3 spider in which case it is ULP.*

- Proof idea:
  - Either two possibilities
    - OR



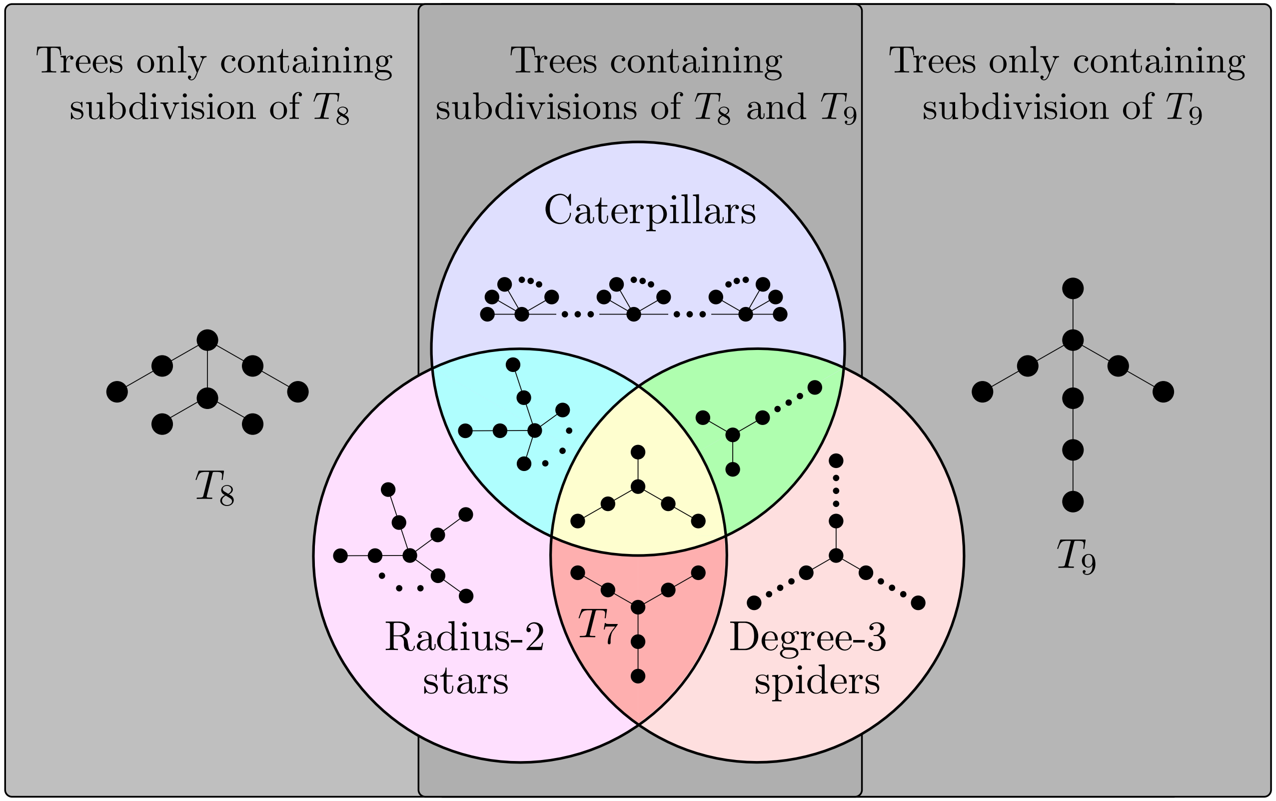
- Proof by induction gives the next theorem:

  Theorem 11 *Every graph either contains one of the seven forbidden graphs in which case it is not ULP, or it is a generalized caterpillar, a radius-2 star, or a extended degree-3 spider in which case it is ULP.*

- Proof idea:
  - Either two possibilities
    - Graph now contains one of the forbidden graphs – G_6 in this case

Summary

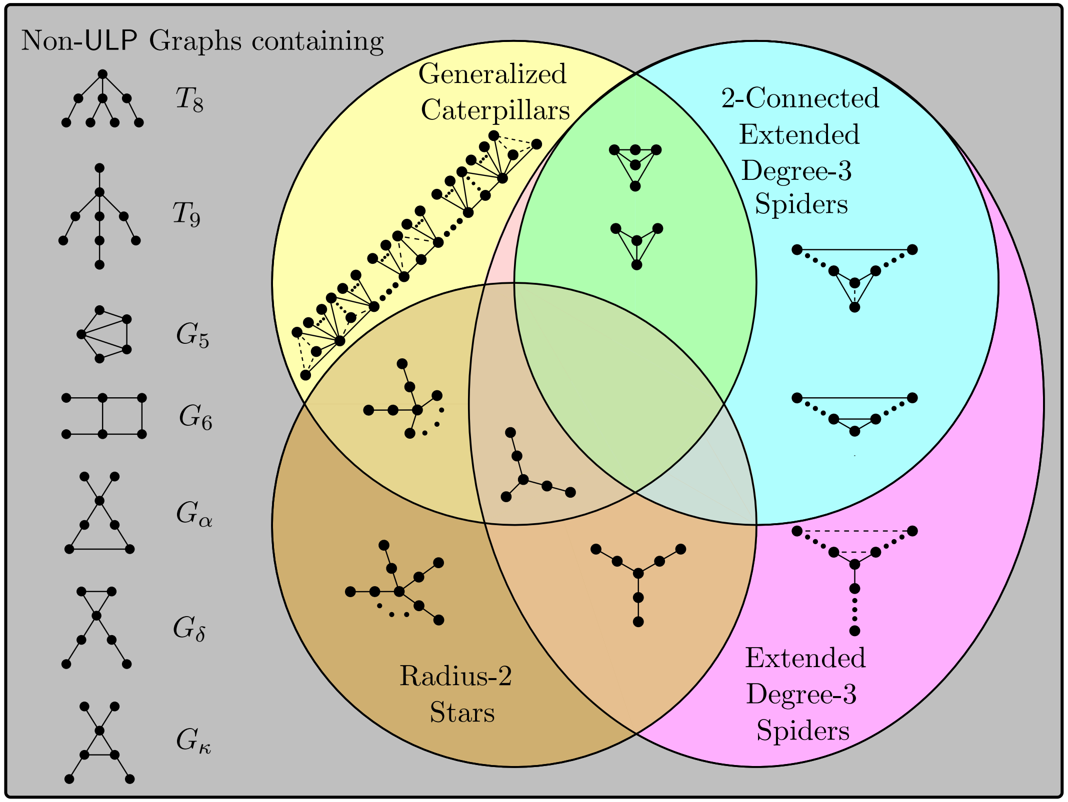
- Background
- Unlabeled Level Planar Trees



  - Previous results for trees

Summary

- Background
- Unlabeled Level Planar Trees
- Unlabeled Level Planar Graphs



  - New results for all graphs

Future Work

- Provide recognition algorithm for all ULP graphs

Future Work

- Provide recognition algorithm for all ULP graphs
- Provide certificate of unlabeled level non-planarity

Future Work

- Provide recognition algorithm for all ULP graphs
- Provide certificate of unlabeled level non-planarity
  - I.e., find a copy of a forbidden ULP graph

Thank You!