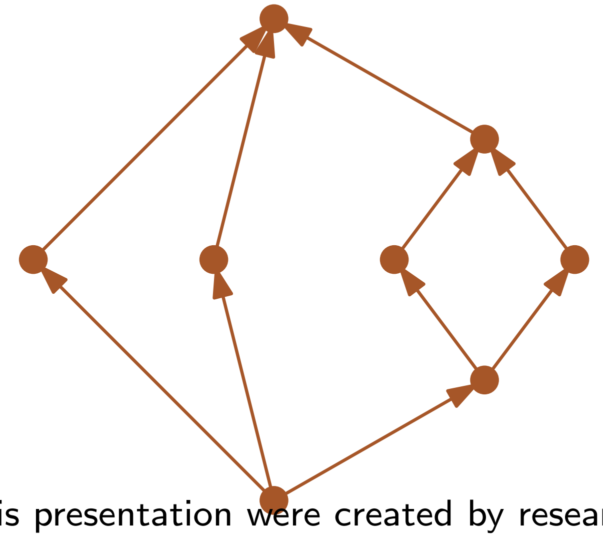
Visualisation of graphs

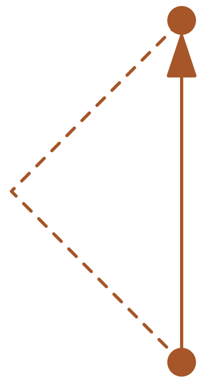
Drawing series-parallel graphs

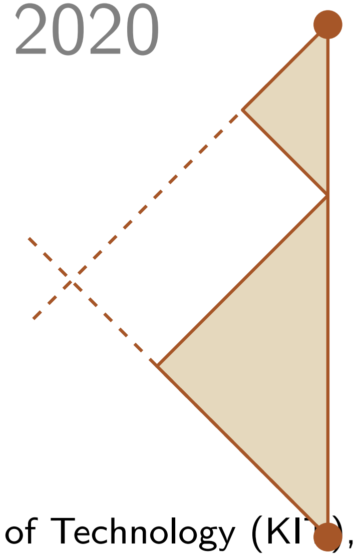
Divide and conquer methods

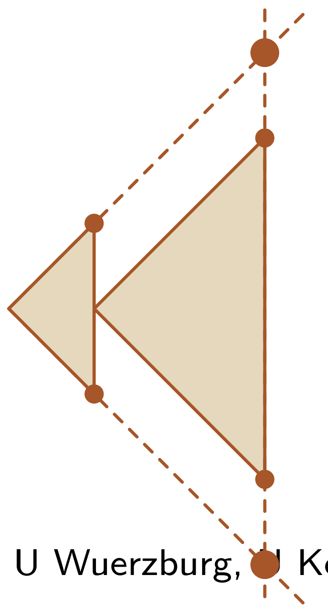
Antonios Symvonis · Chrysanthi Raftopoulou

Fall semester 2020







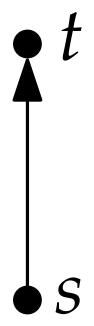


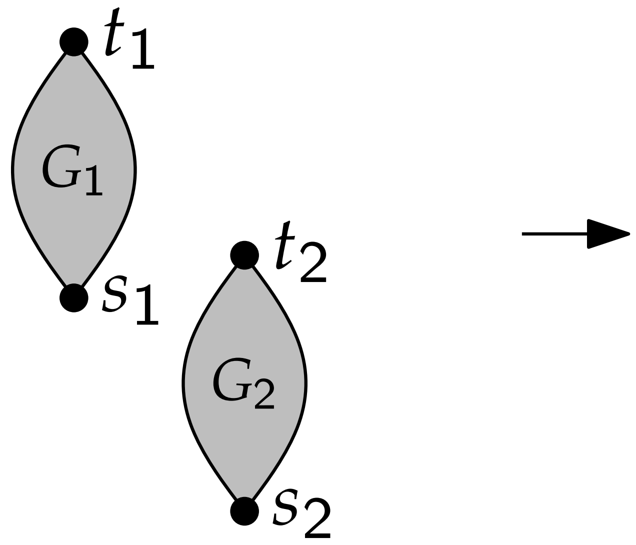
The original slides of this presentation were created by researchers at Karlsruhe Institute of Technology (KIT), TU Wien, U Wuerzburg, U Konstanz, ...
The original presentation was modified/updated by A. Symvonis and C. Raftopoulou

Series-parallel graphs

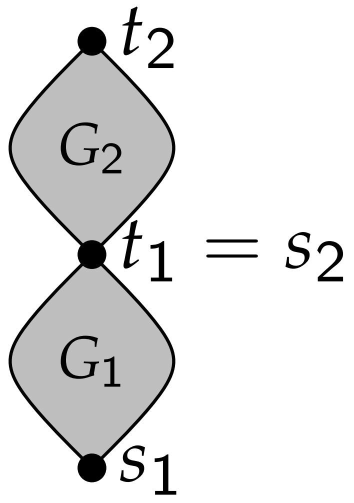
A graph G is **series-parallel**, if

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

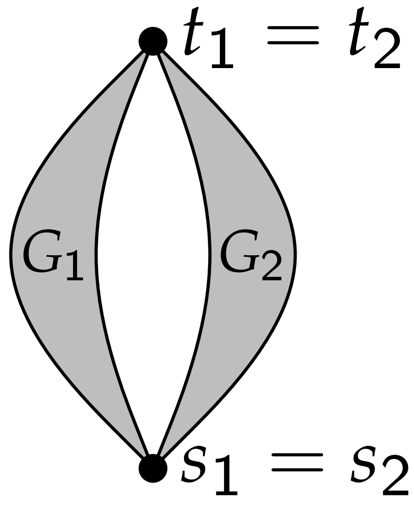




Series composition

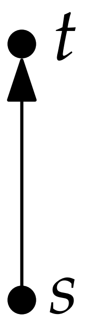


Parallel composition



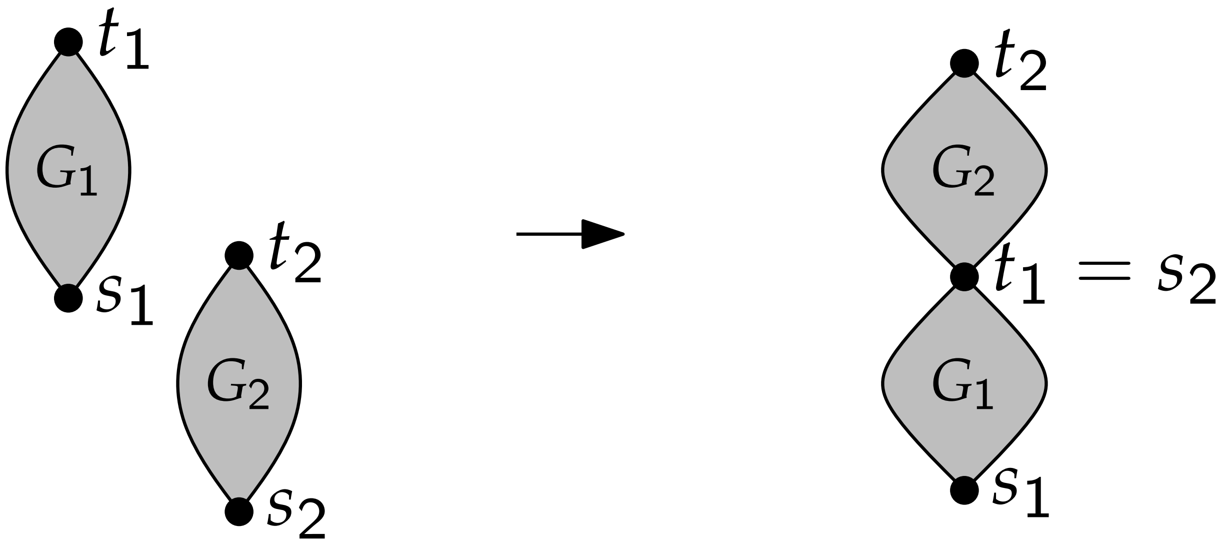
Series-parallel graphs

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

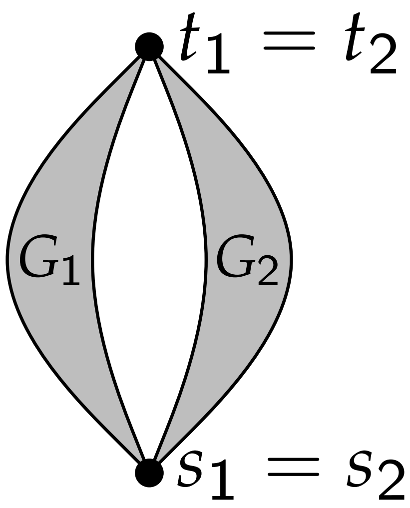


Observations:
- $|E| \leq 2|V| - 4$
- Series-parallel graphs are planar

Series composition



Parallel composition



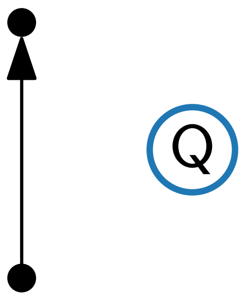
Series-parallel graphs – decomposition tree

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

Series-parallel graphs – decomposition tree

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

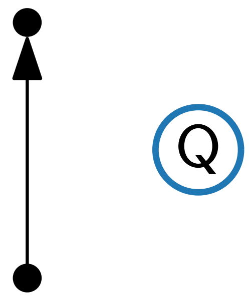
- A Q-node represents a single edge

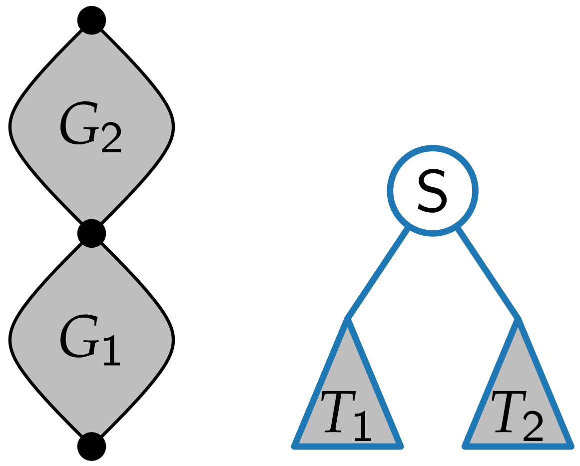


Series-parallel graphs – decomposition tree

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

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

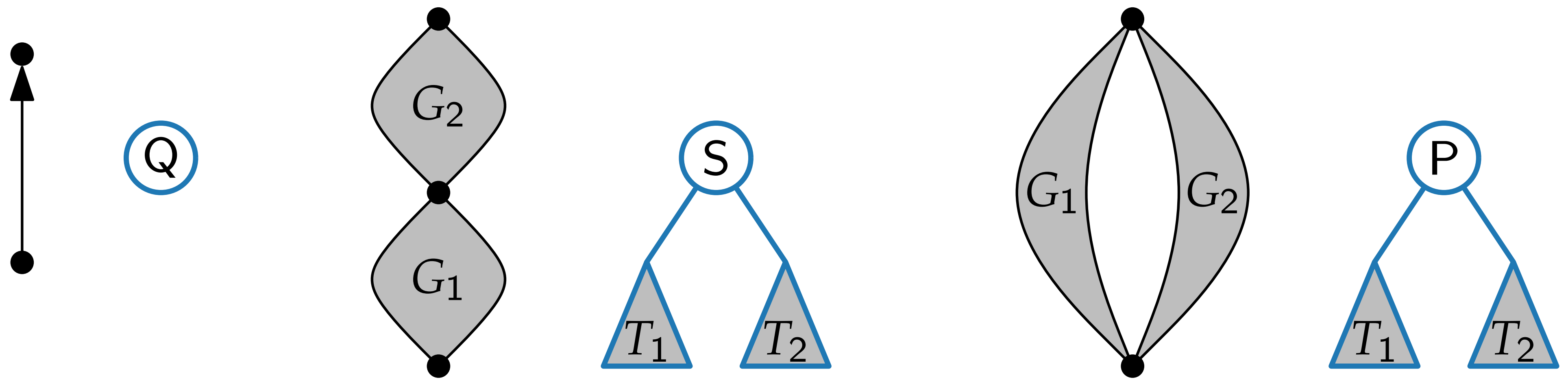




Series-parallel graphs – decomposition tree

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

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



Series-parallel graphs – decomposition tree

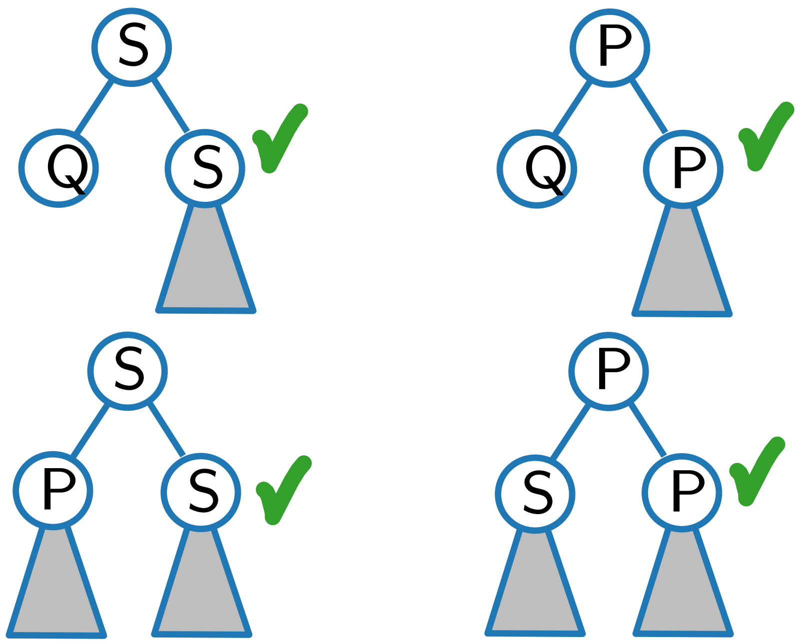
We further require:

- if a node μ and its parent ν have the same type, then μ is the **right** child of ν.

Series-parallel graphs – decomposition tree

We further require:

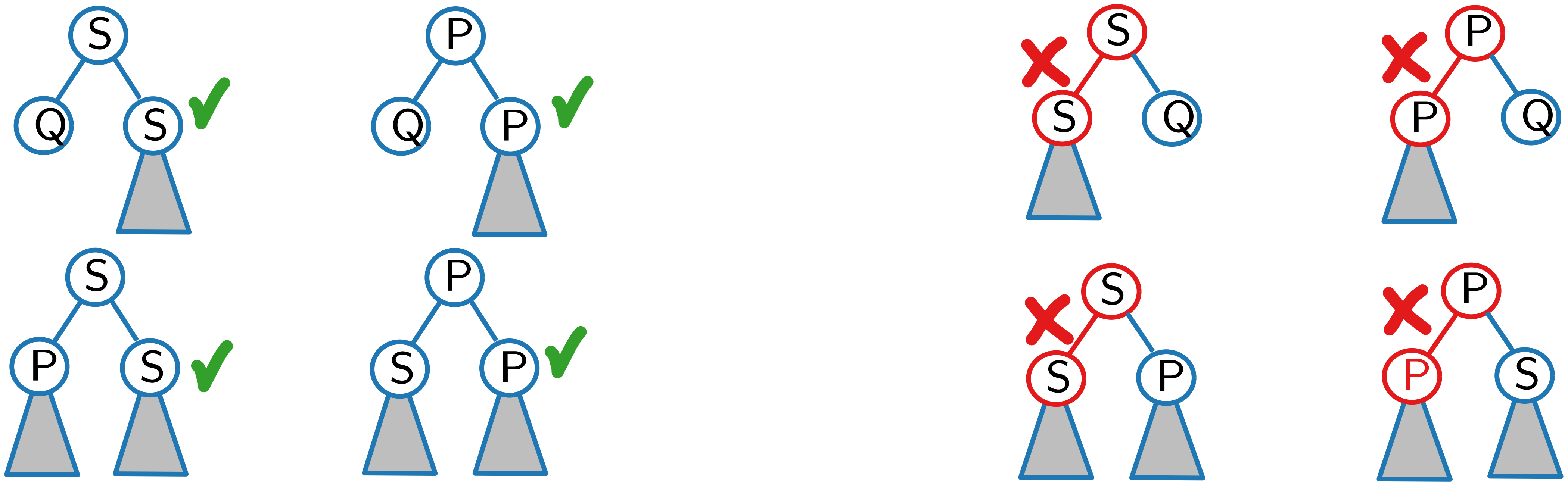
- if a node μ and its parent ν have the same type, then μ is the **right** child of ν.



Series-parallel graphs – decomposition tree

We further require:

- if a node μ and its parent ν have the same type, then μ is the **right** child of ν.



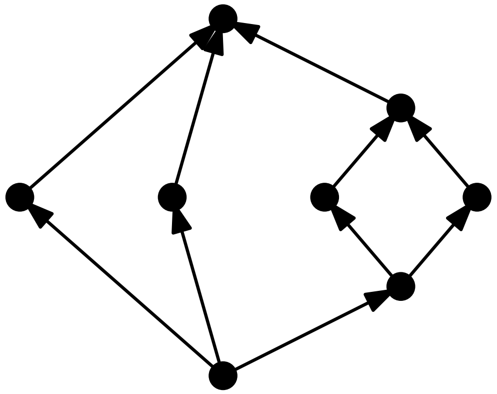
Series-parallel graphs – decomposition tree

We further require:

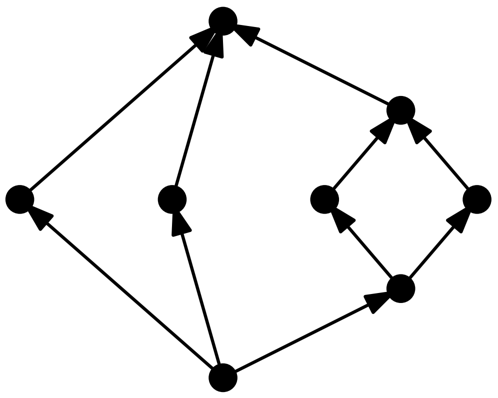
- if a node μ and its parent ν have the same type, then μ is the **right** child of ν.

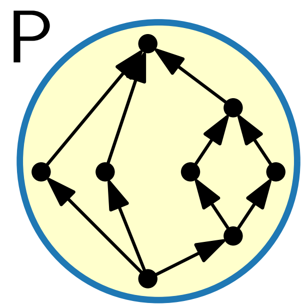
- Unique decomposition tree
- The order of the children (Q or S) define the graph embedding

Series-parallel graphs – decomposition example

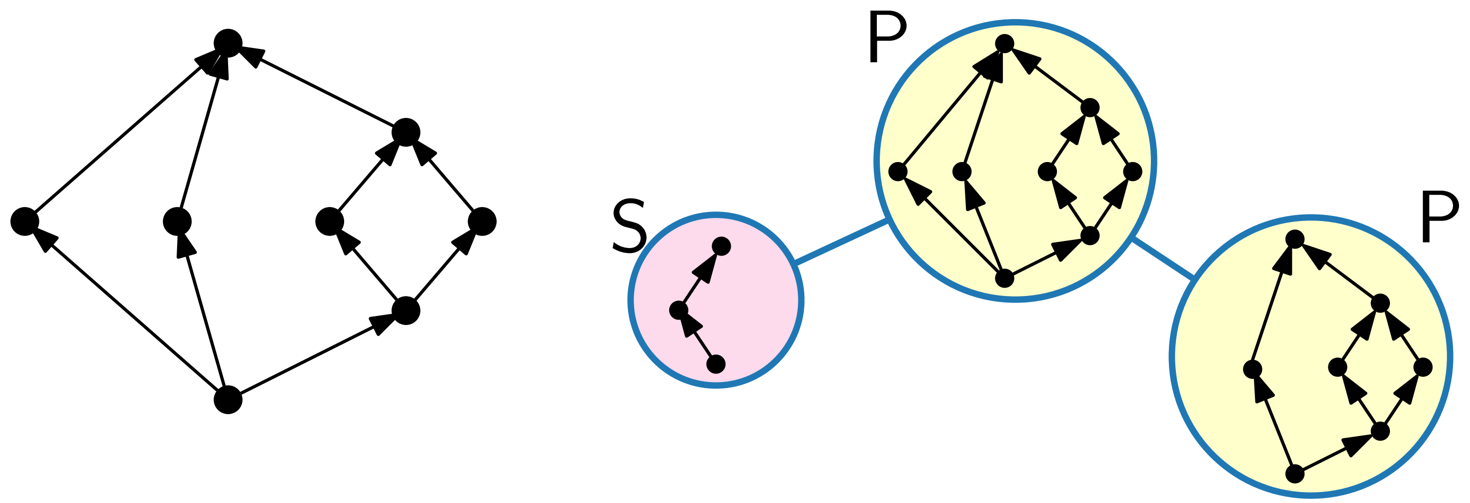


Series-parallel graphs – decomposition example

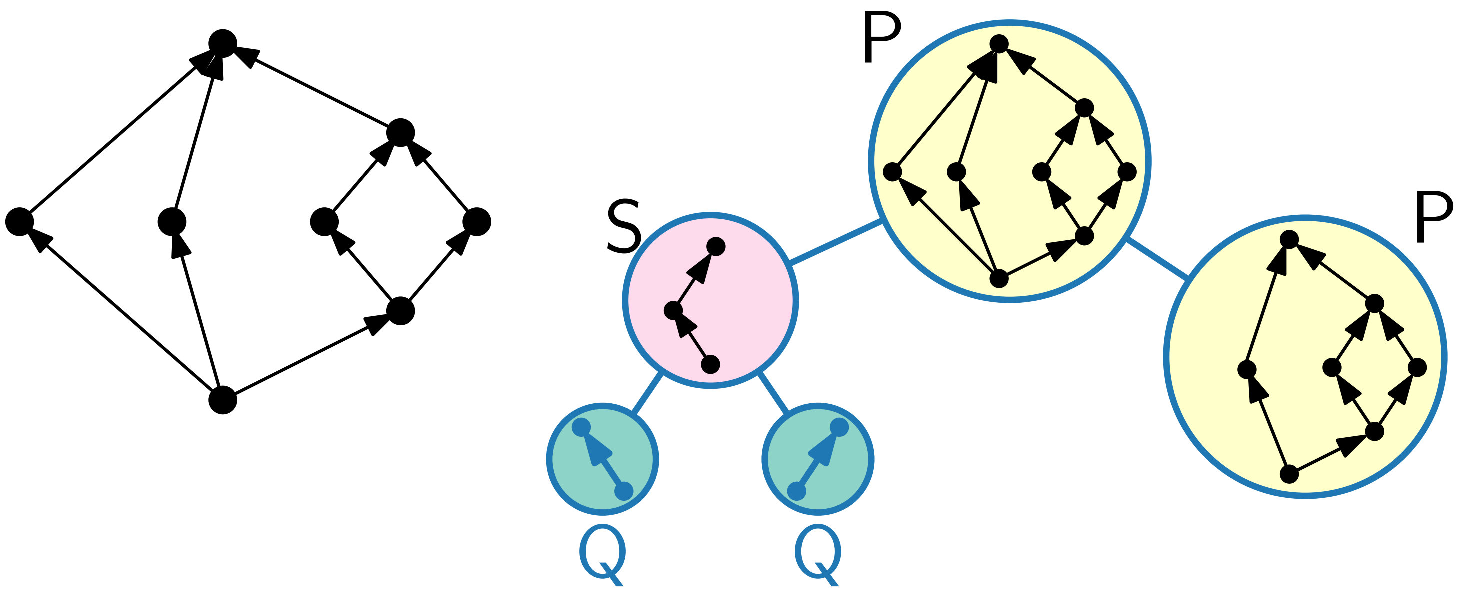




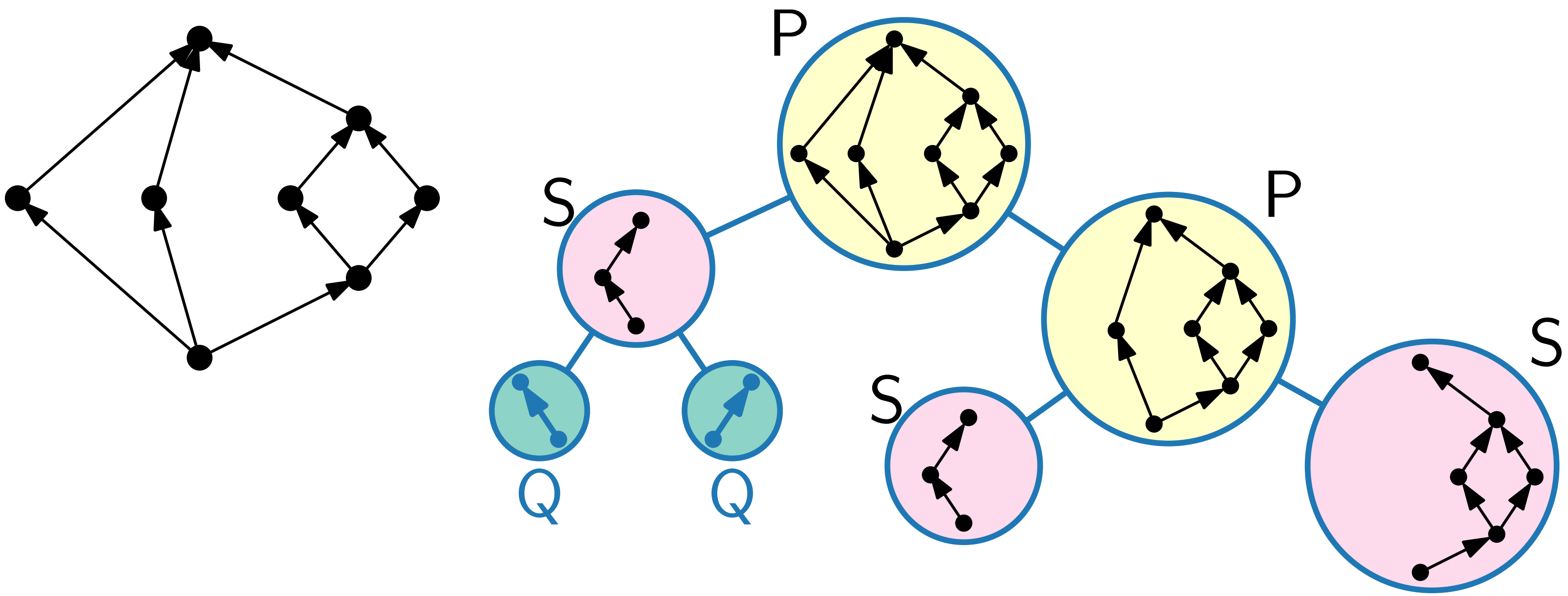
Series-parallel graphs – decomposition example



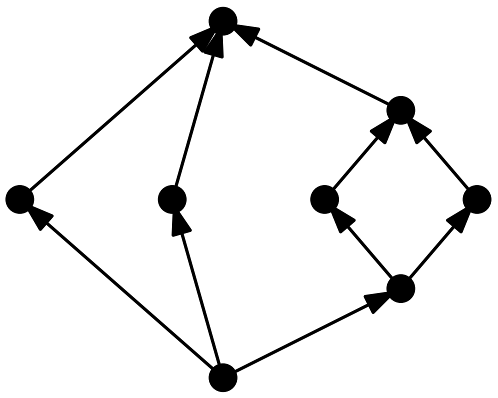
Series-parallel graphs – decomposition example

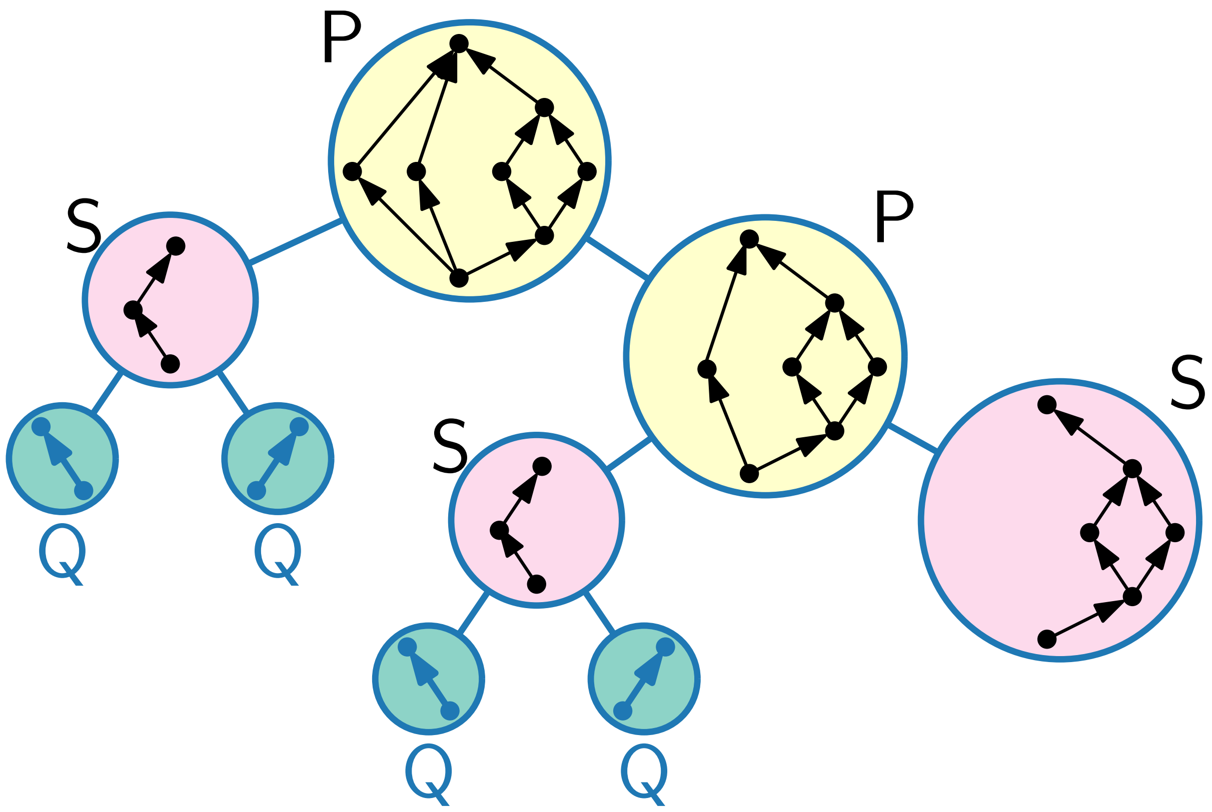


Series-parallel graphs – decomposition example

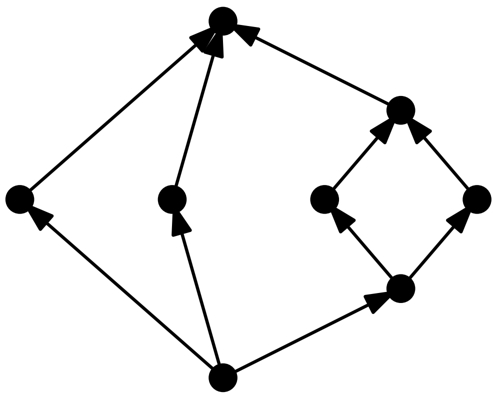


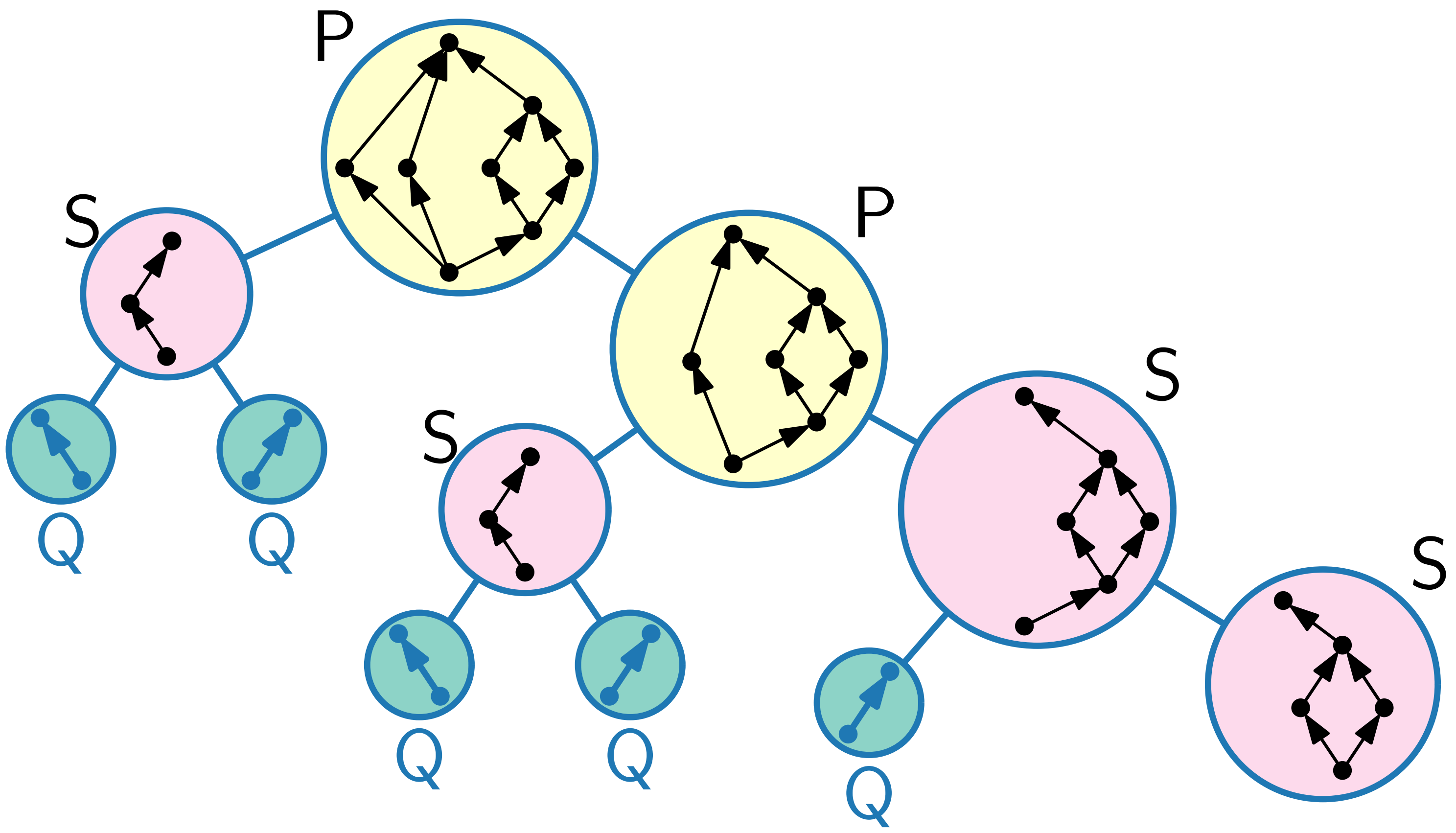
Series-parallel graphs – decomposition example



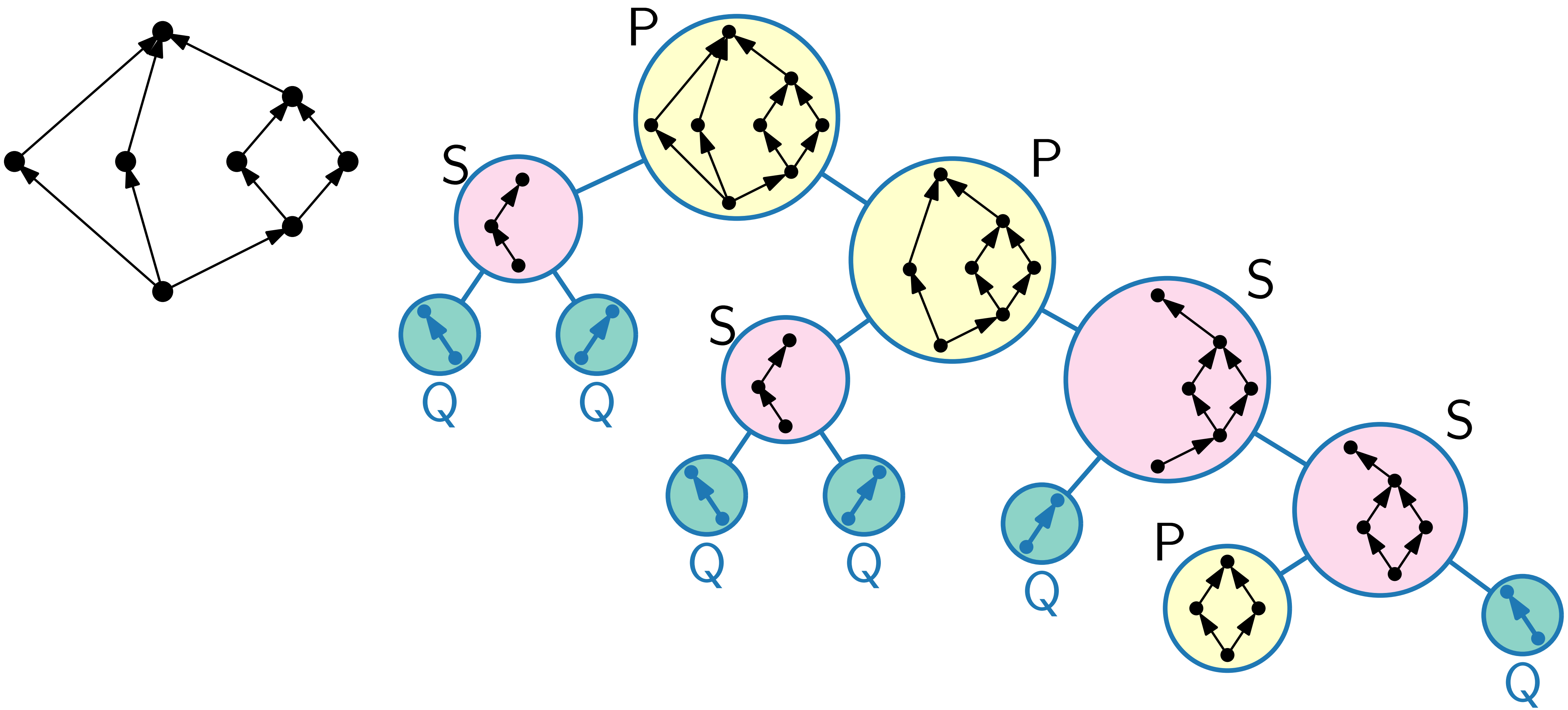


Series-parallel graphs – decomposition example

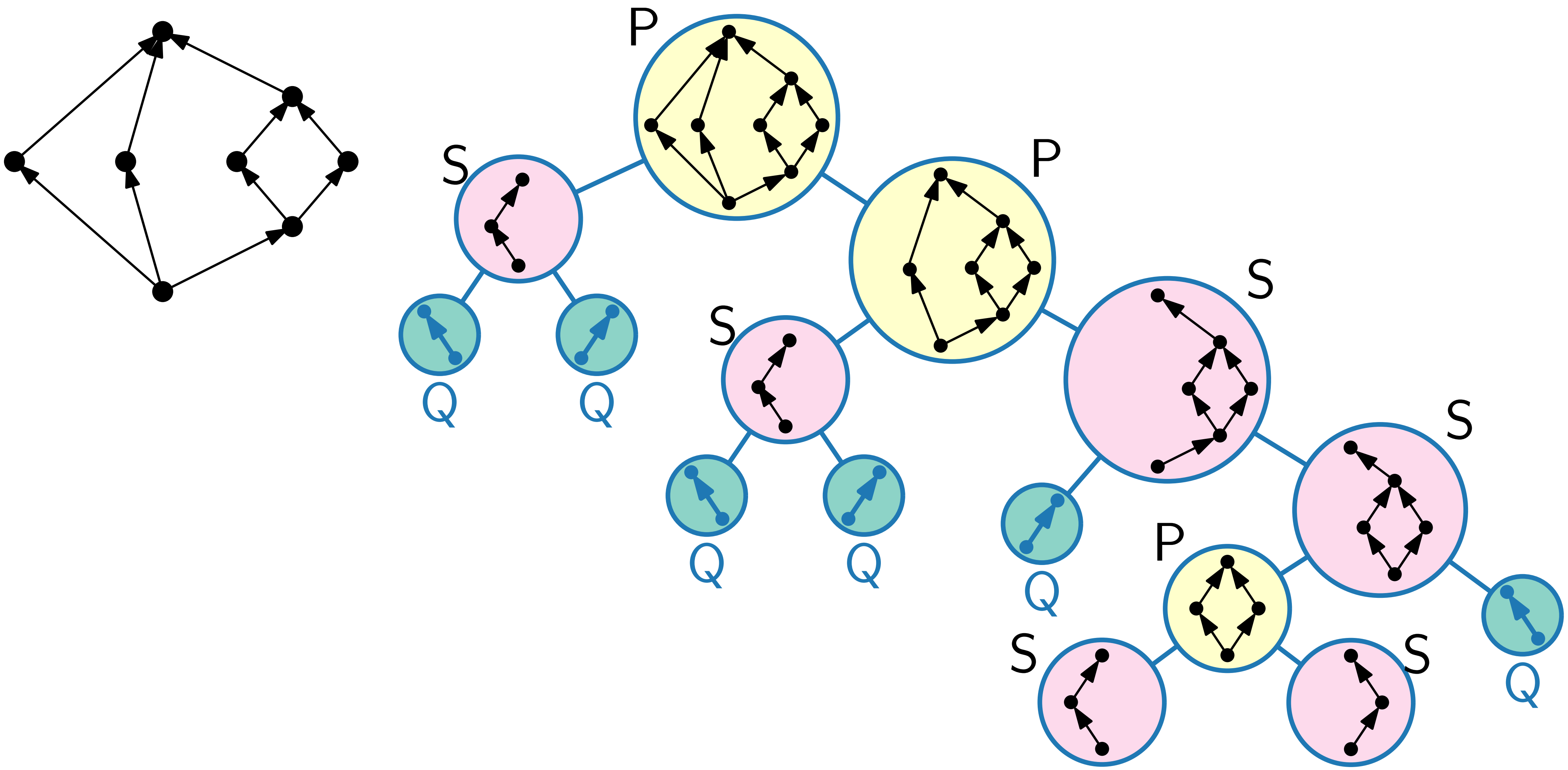




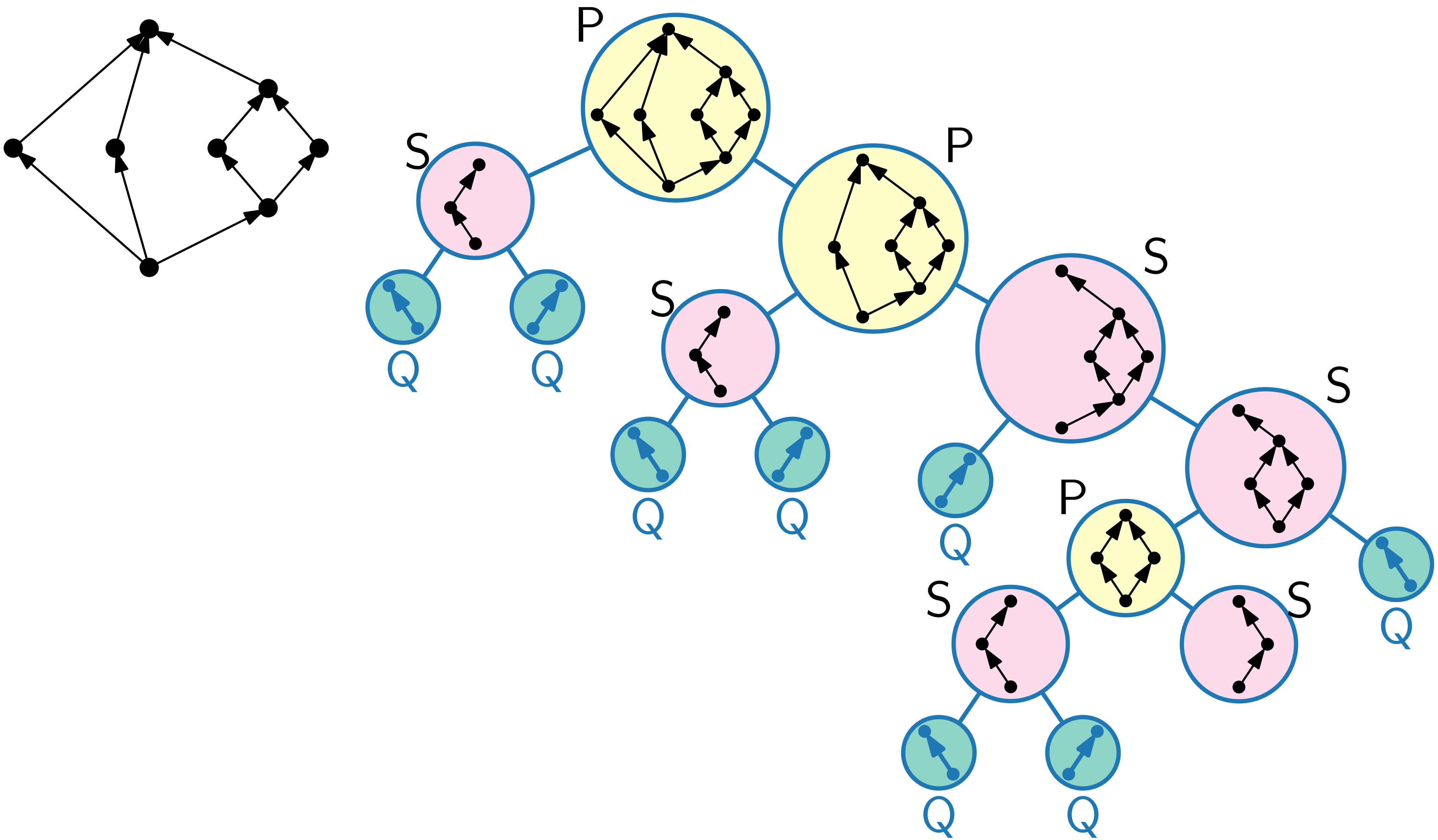
Series-parallel graphs – decomposition example



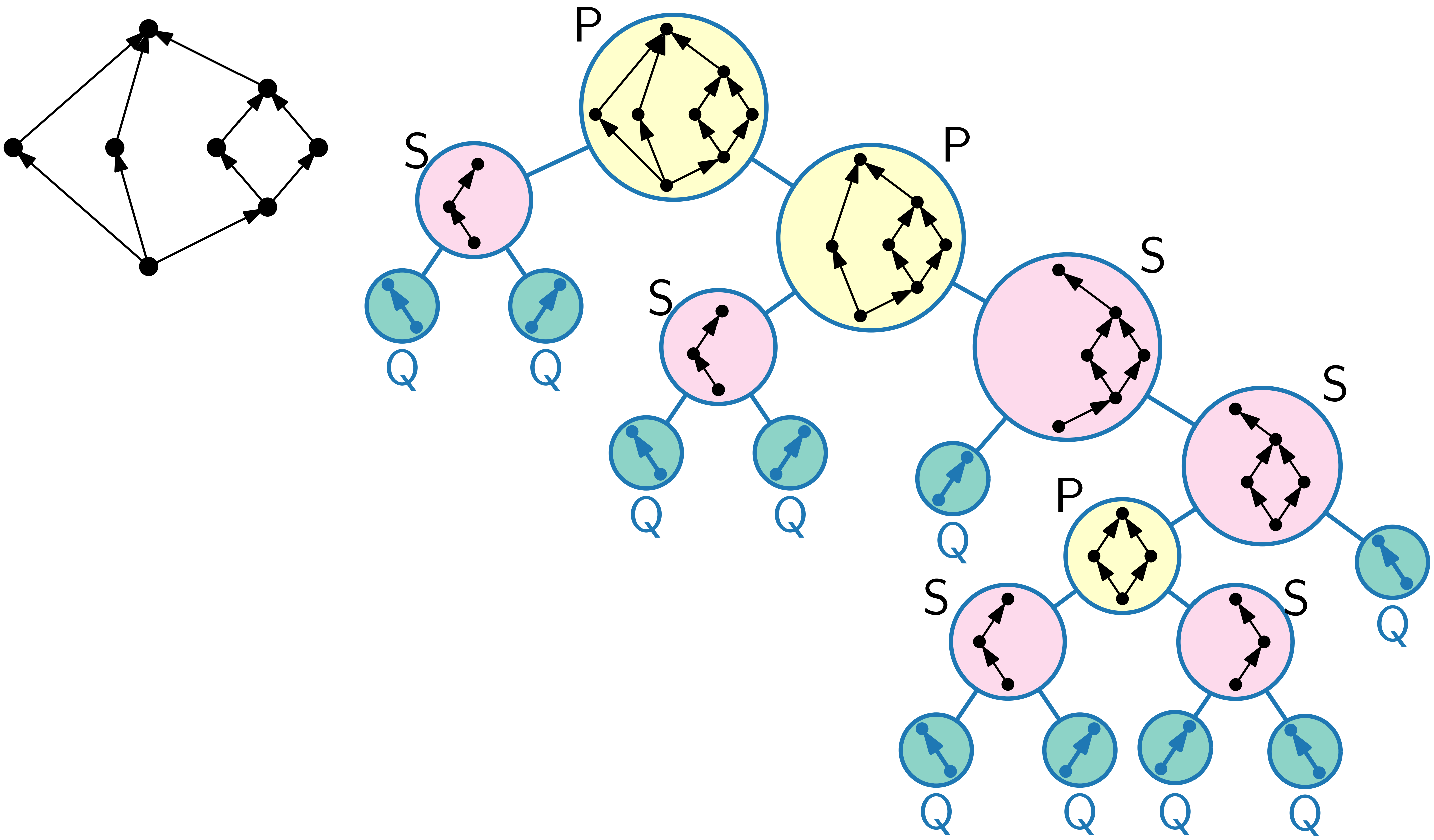
Series-parallel graphs – decomposition example



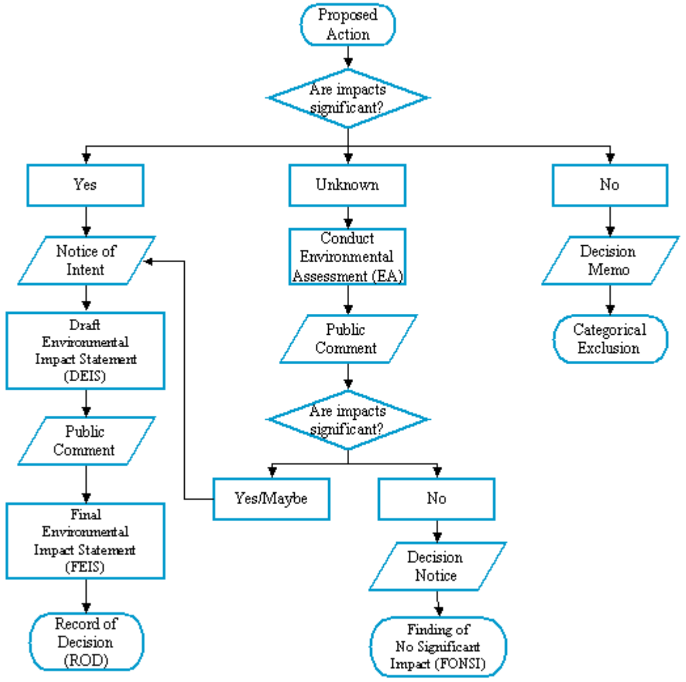
Series-parallel graphs – decomposition example



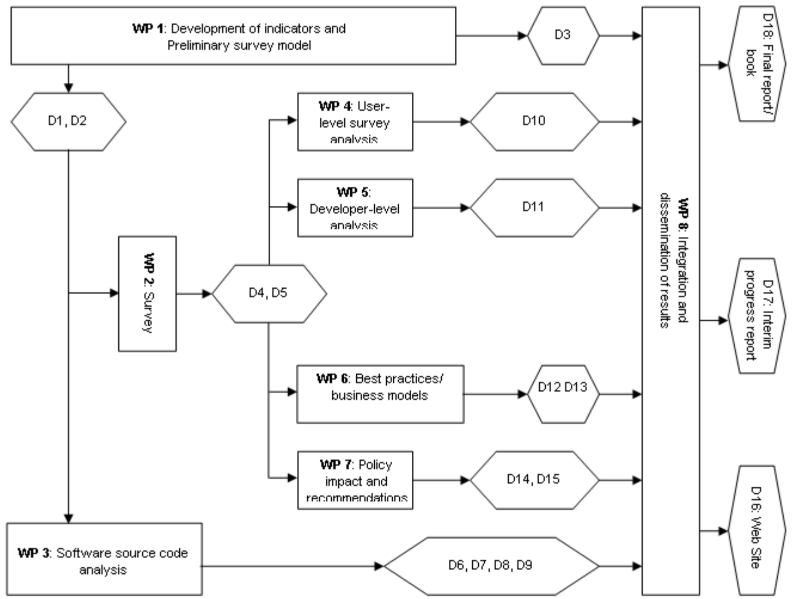
Series-parallel graphs – decomposition example



Series-parallel graphs – applications



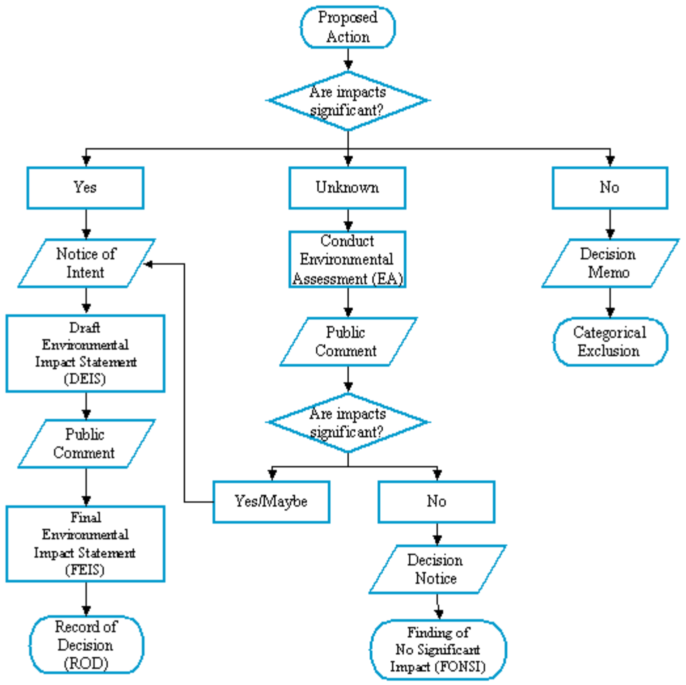
Flowcharts



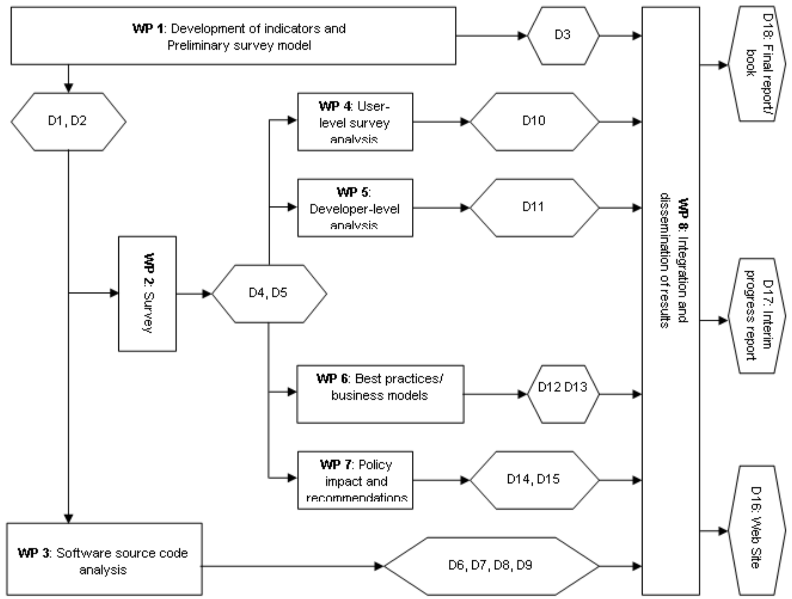
PERT-Diagrams

(Program Evaluation and Review Technique)

Series-parallel graphs – applications



Flowcharts



PERT-Diagrams

(Program Evaluation and Review Technique)

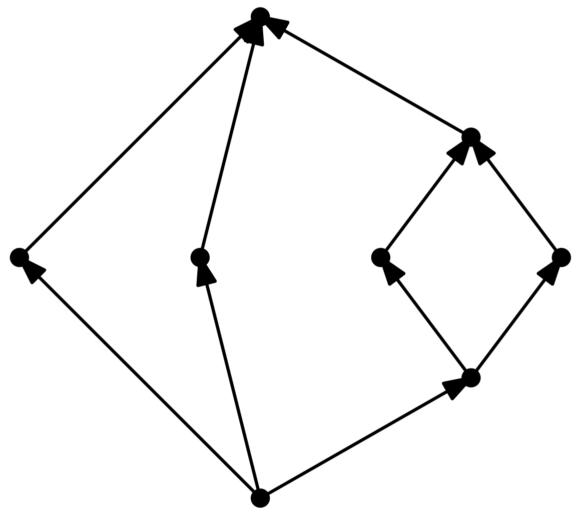
Computational complexity:

Linear time algorithms for $\mathcal{NP}$-hard problems

(e.g. Maximum Matching, MIS, Hamiltonian Completion)

Series-parallel graphs – drawing style

Drawing conventions

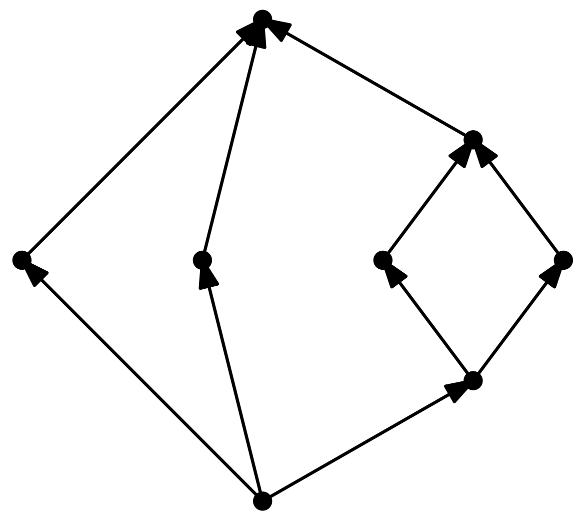


Drawing aesthetics

Series-parallel graphs – drawing style

Drawing conventions

- Planarity
- Straight-line edges
- Upward



Drawing aesthetics

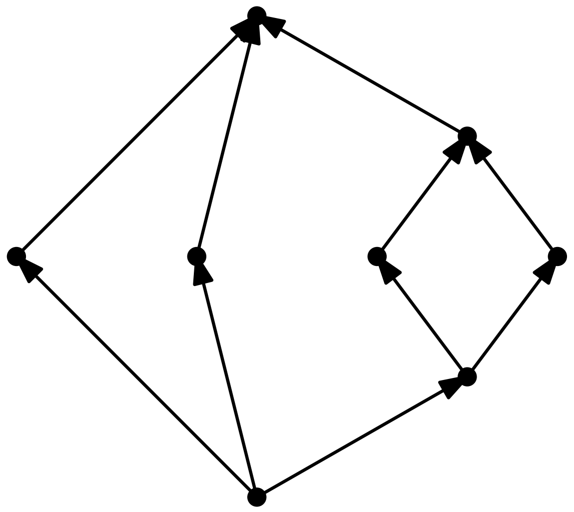
Series-parallel graphs – drawing style

Drawing conventions

- Planarity
- Straight-line edges
- Upward

Drawing aesthetics

- Area
- Symmetry



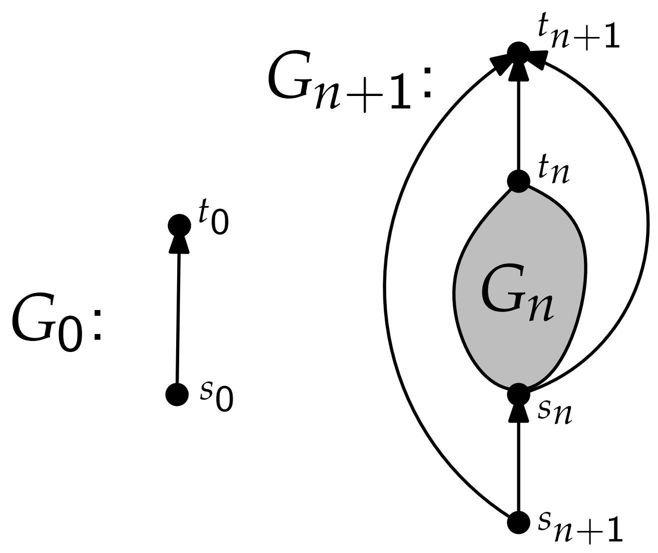
Series-parallel graphs – An exponential area bound

- **A class of graphs that requires exponential area for its upward drawing**

Theorem [Bertolazzi et al. 1994] Any upward drawing of the $2n$-vertex embedded graph G_n that preserves the embedding requires area $\Omega(4^n)$, under any resolution rule.

Series-parallel graphs – An exponential area bound

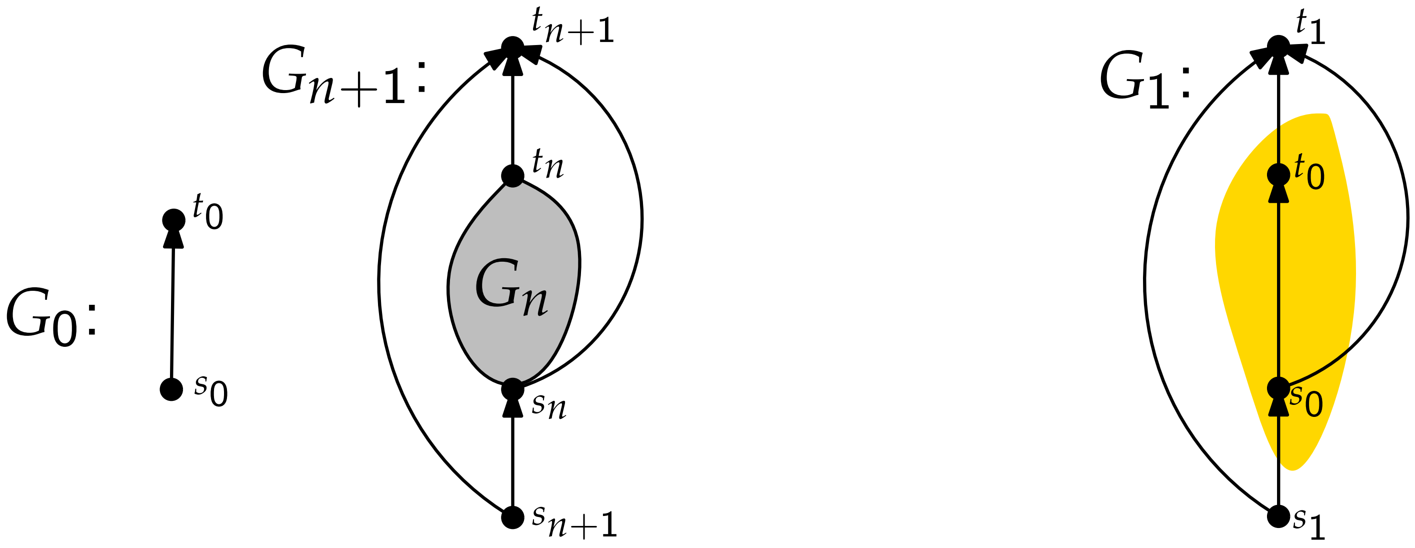
- **A class of graphs that requires exponential area for its upward drawing**



Theorem [Bertolazzi et al. 1994] Any upward drawing of the $2n$-vertex embedded graph G_n that preserves the embedding requires area $\Omega(4^n)$, under any resolution rule.

Series-parallel graphs – An exponential area bound

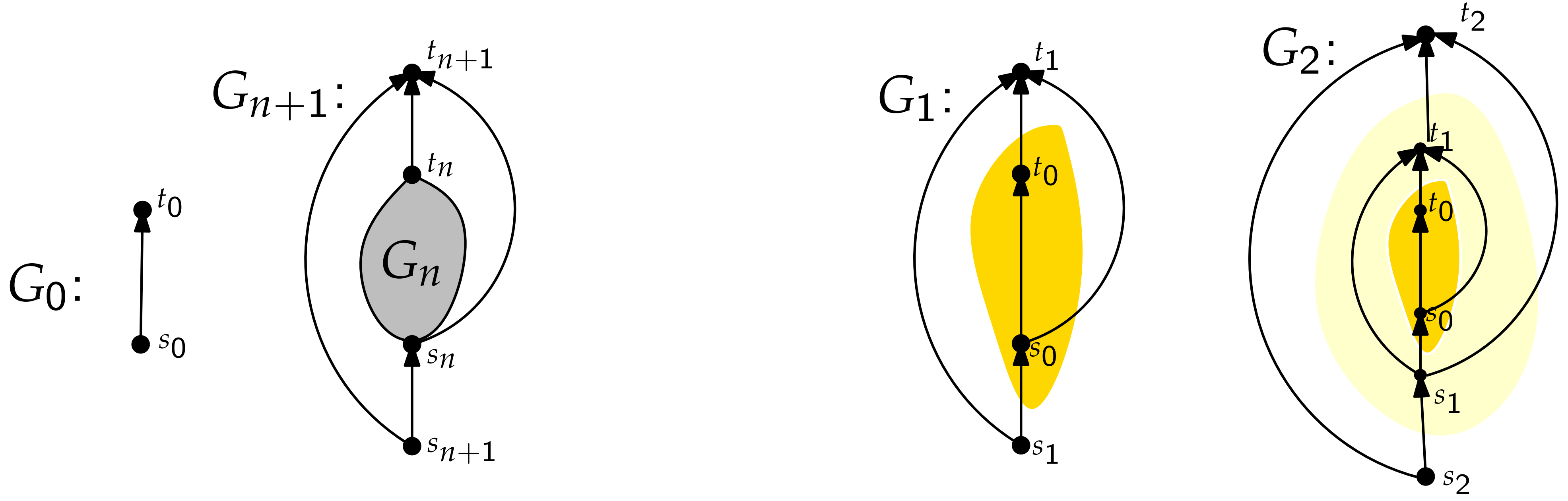
A class of graphs that requires exponential area for its upward drawing



Theorem [Bertolazzi et al. 1994] Any upward drawing of the $2n$-vertex embedded graph G_n that preserves the embedding requires area $\Omega(4^n)$, under any resolution rule.

Series-parallel graphs – An exponential area bound

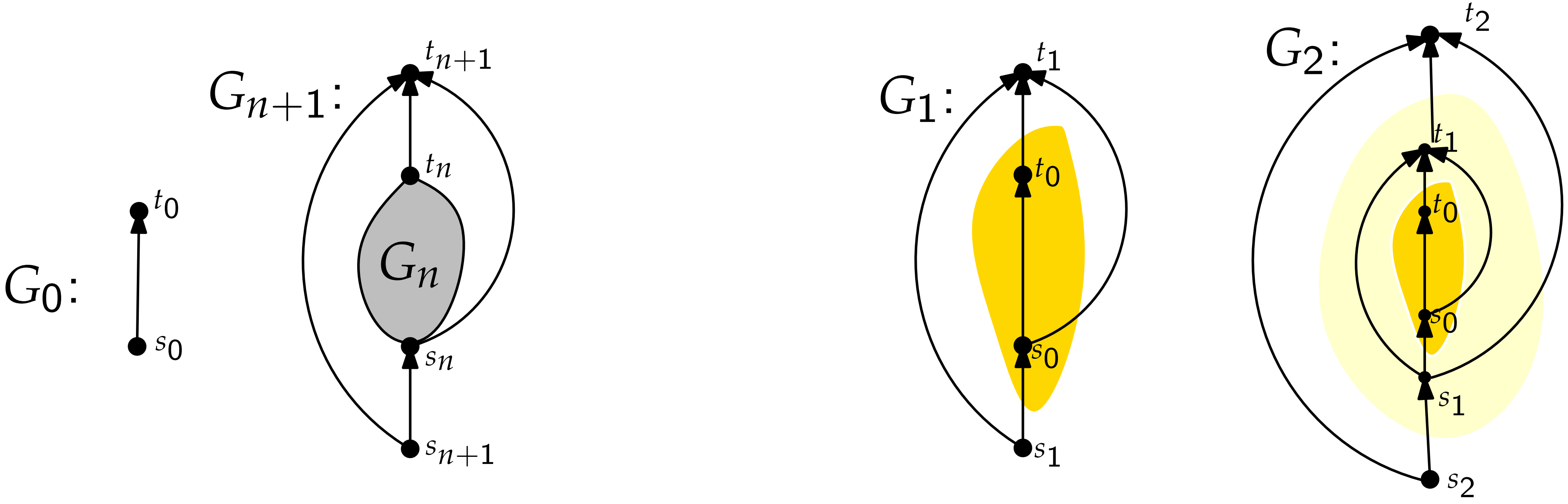
A class of graphs that requires exponential area for its upward drawing



Theorem [Bertolazzi et al. 1994] Any upward drawing of the $2n$-vertex embedded graph G_n that preserves the embedding requires area $\Omega(4^n)$, under any resolution rule.

Series-parallel graphs – An exponential area bound

- **A class of graphs that requires exponential area for its upward drawing**

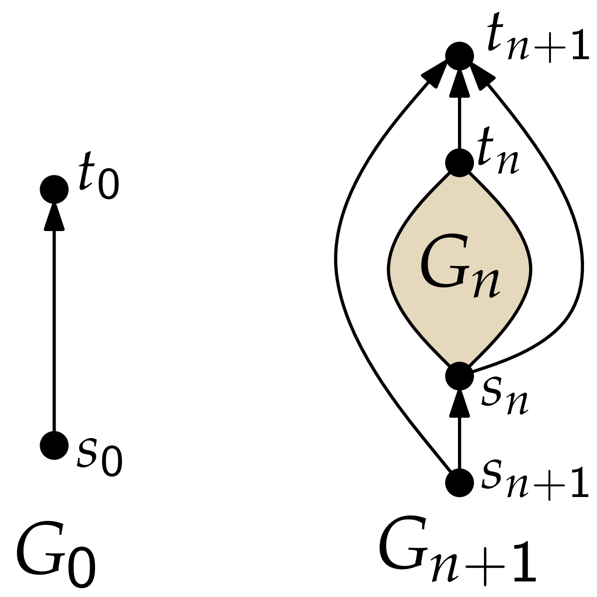


Theorem [Bertolazzi et al. 1994] Any upward drawing of the $2n$-vertex embedded graph G_n that preserves the embedding requires area $\Omega(4^n)$, under any resolution rule.

Series-parallel graphs – fixed embedding

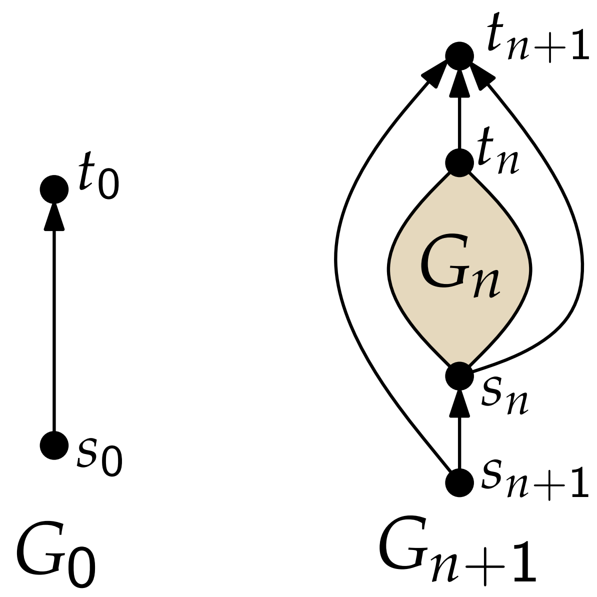
Series-parallel graphs – fixed embedding

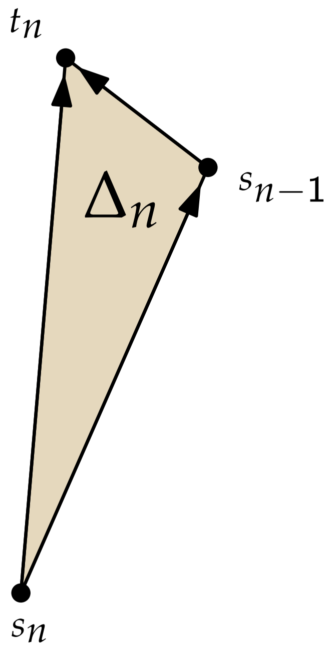
Proof:



Series-parallel graphs – fixed embedding

Proof:

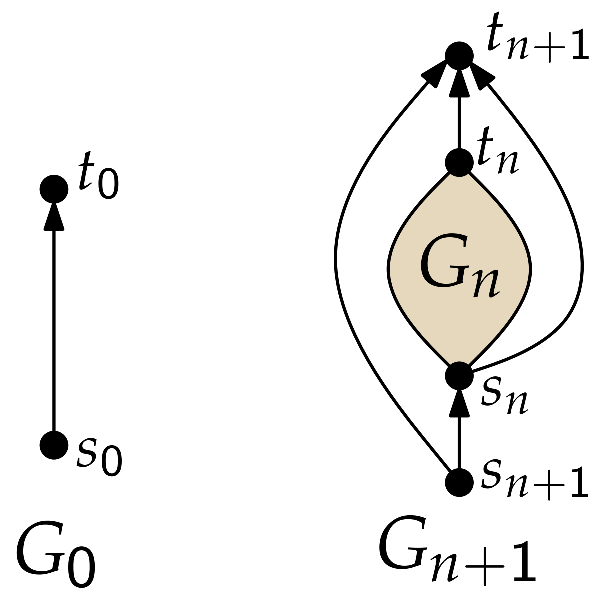


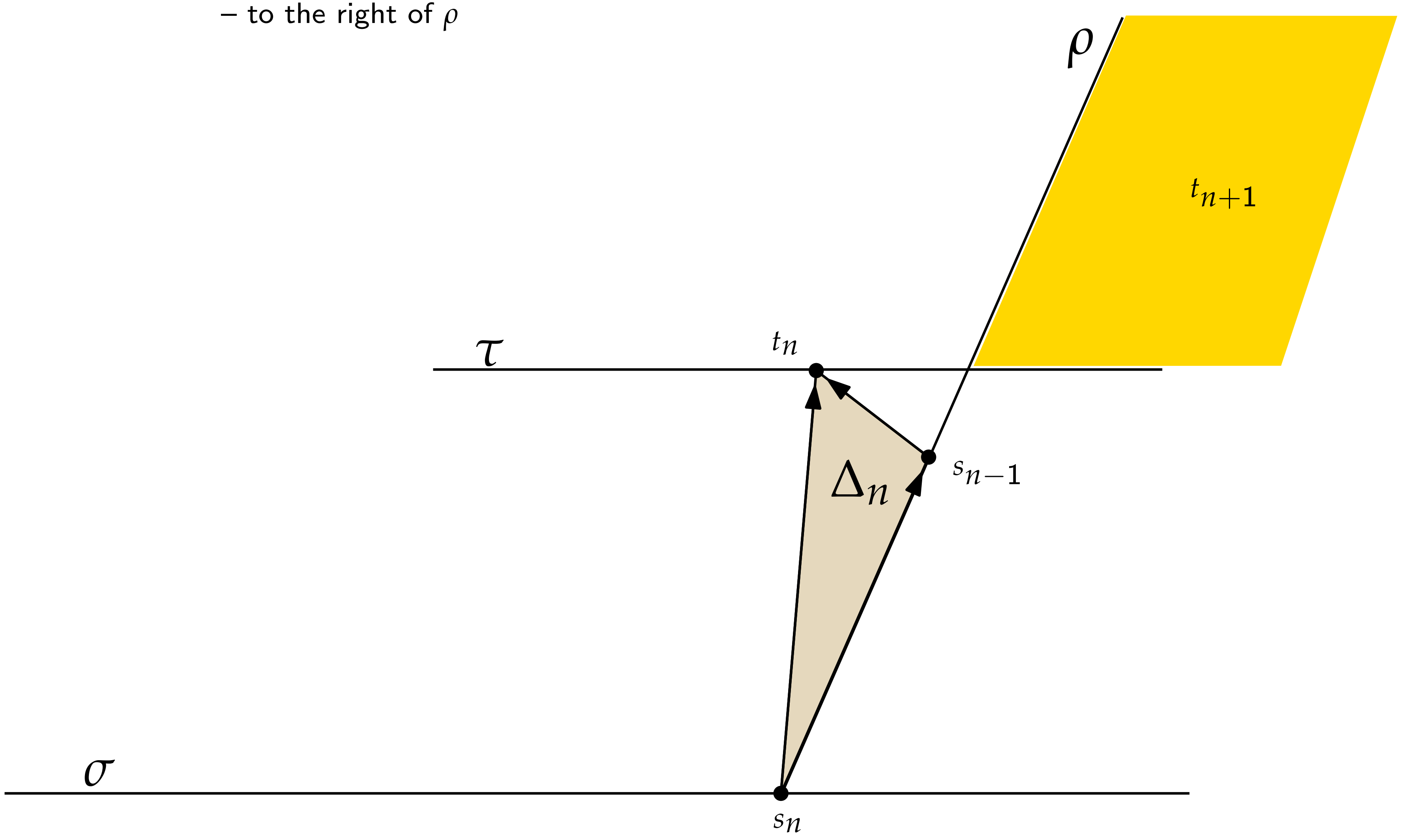


Series-parallel graphs – fixed embedding

Proof:

t_{n+1}: – above τ
– to the right of ρ



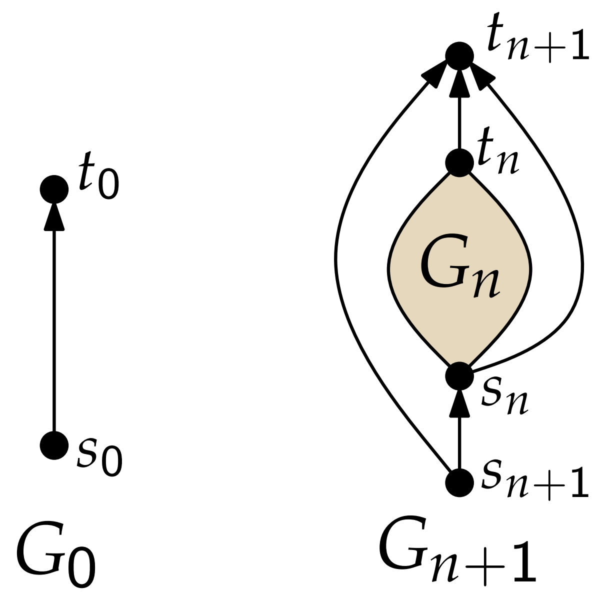


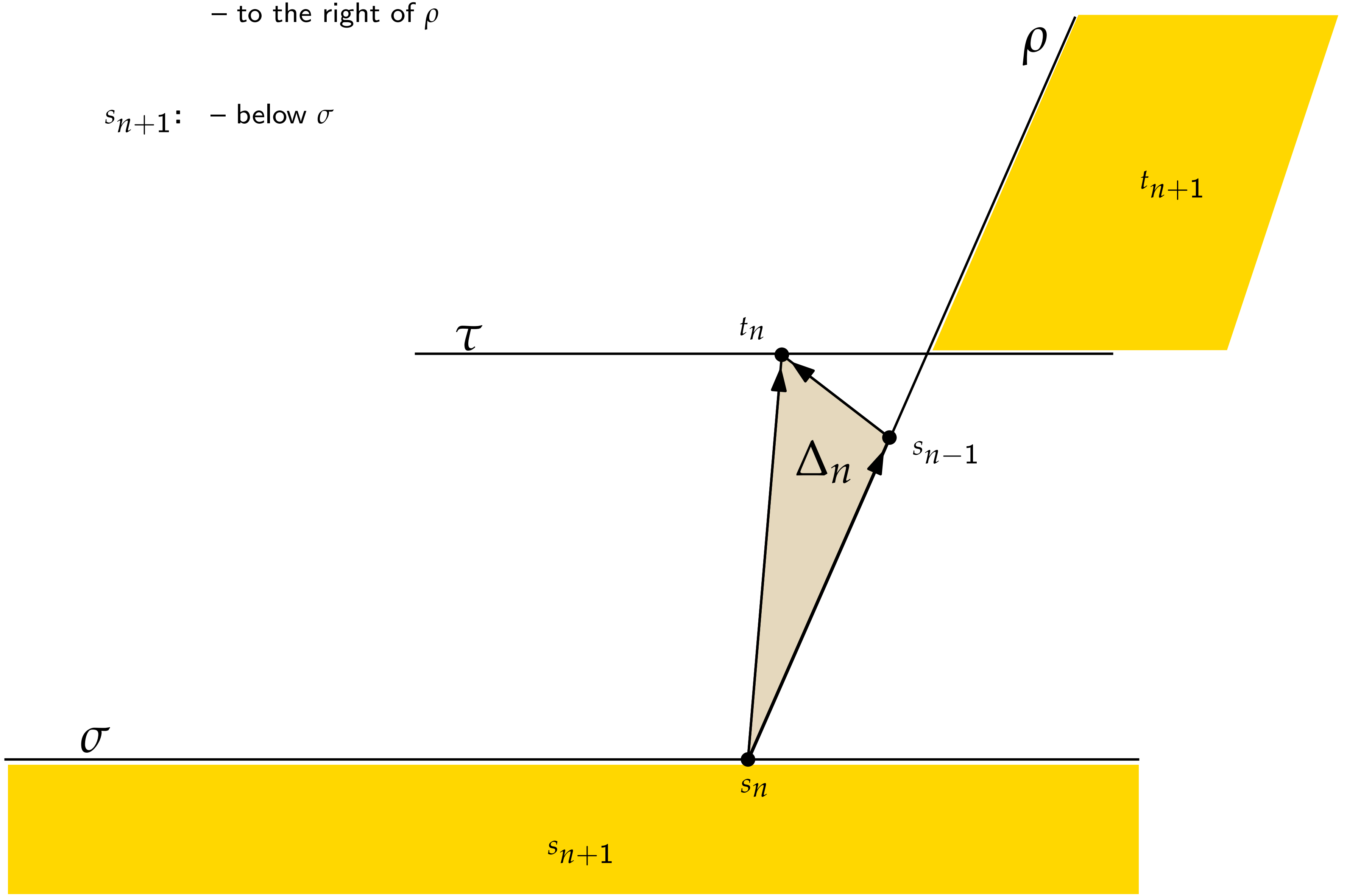
Series-parallel graphs – fixed embedding

Proof:

t_{n+1}: – above τ
– to the right of ρ

s_{n+1}: – below σ



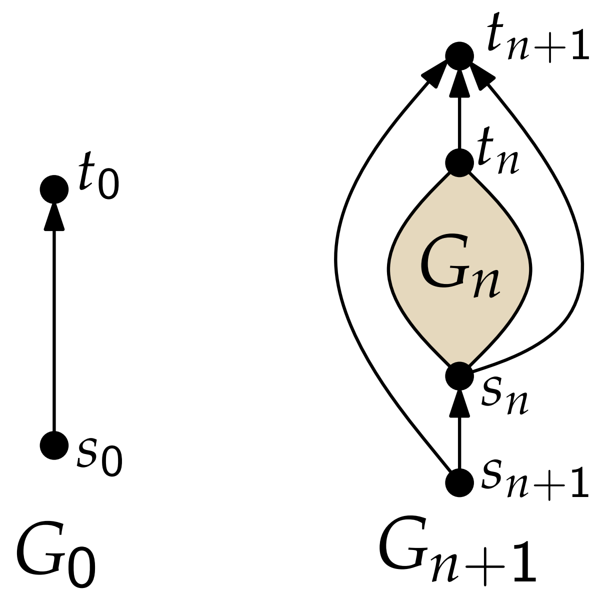


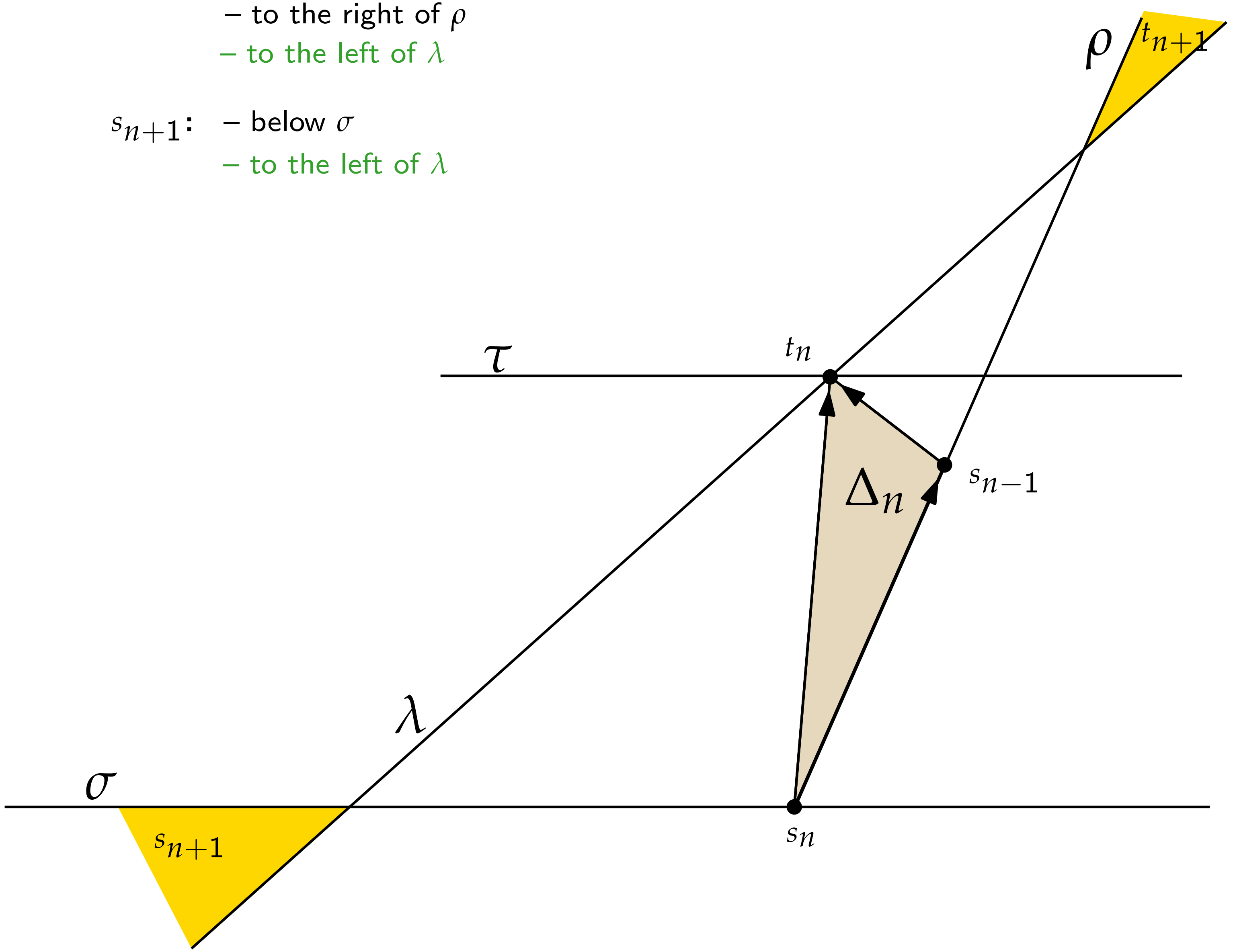
Series-parallel graphs – fixed embedding

Proof:

t_{n+1}:
- above τ
- to the right of ρ
- to the left of λ

s_{n+1}:
- below σ
- to the left of λ





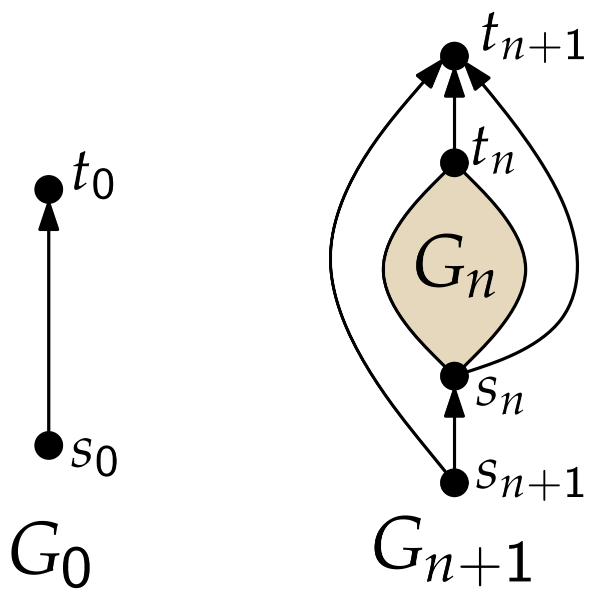
Series-parallel graphs – fixed embedding

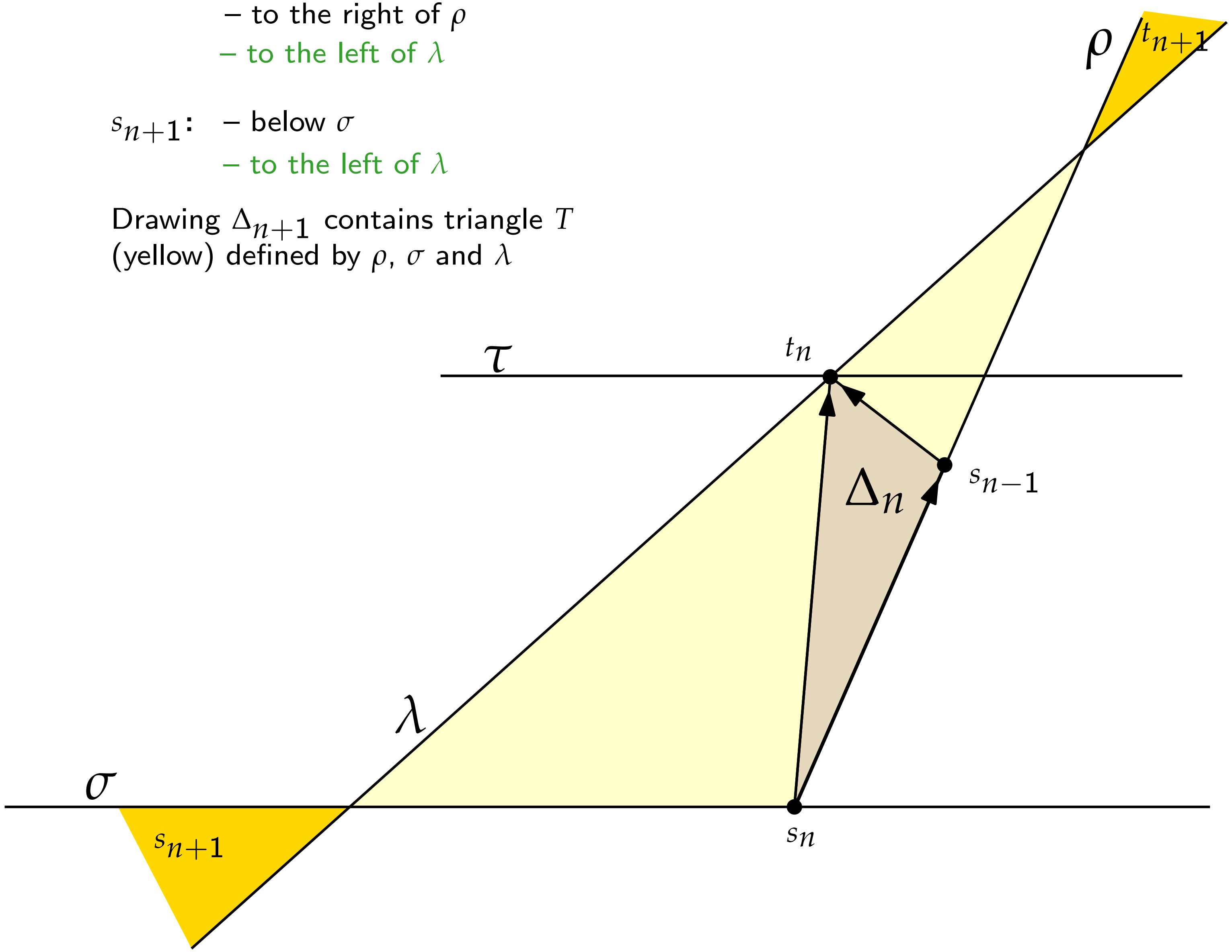
Proof:

t_{n+1}: – above τ
– to the right of ρ
– to the left of λ

s_{n+1}: – below σ
– to the left of λ

Drawing Δ_{n+1} contains triangle T (yellow) defined by ρ, σ and λ





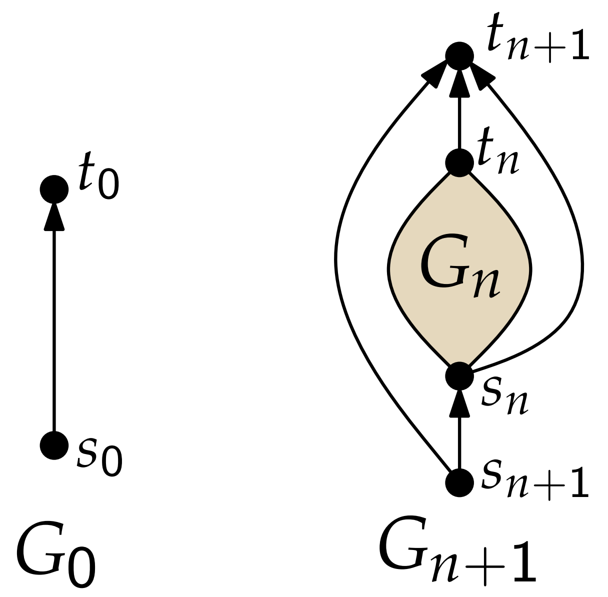
Series-parallel graphs – fixed embedding

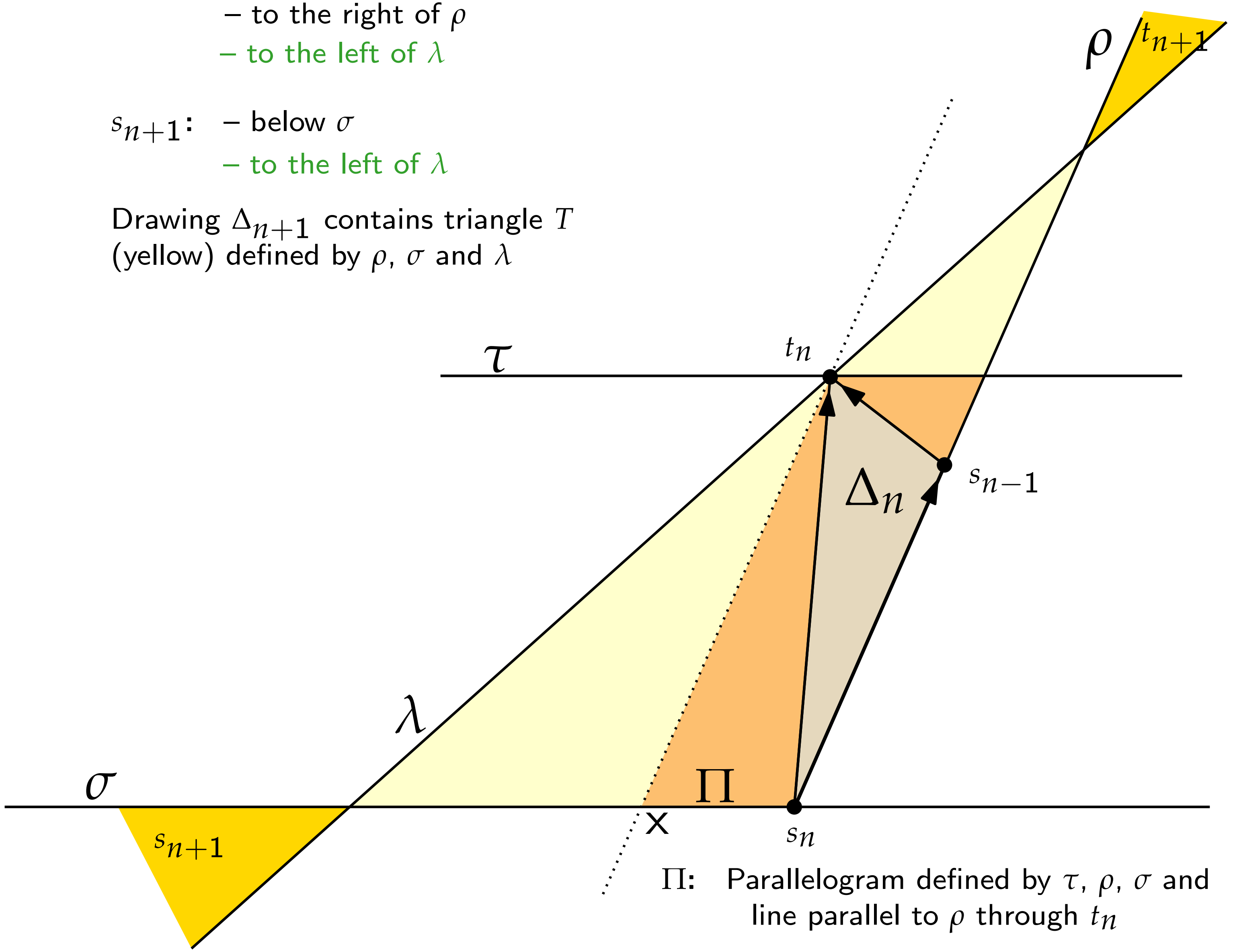
Proof:

t_{n+1}: – above τ
– to the right of ρ
– to the left of λ

s_{n+1}: – below σ
– to the left of λ

Drawing Δ_{n+1} contains triangle T (yellow) defined by ρ, σ and λ





Π: Parallelogram defined by τ, ρ, σ and line parallel to ρ through t_n

Series-parallel graphs – fixed embedding

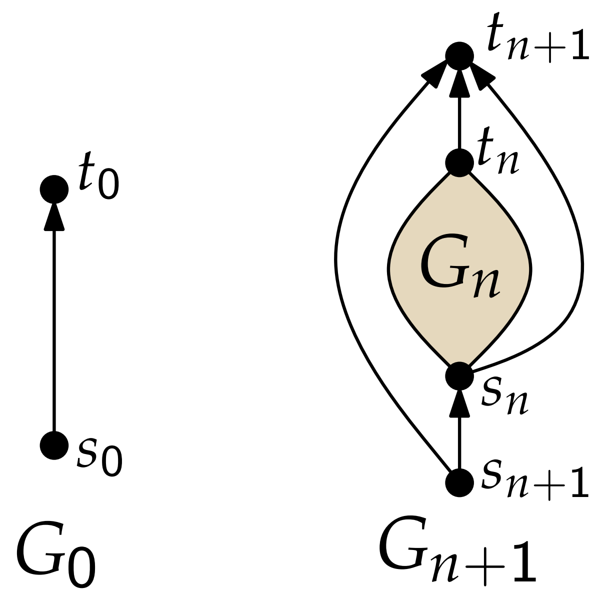
Proof:

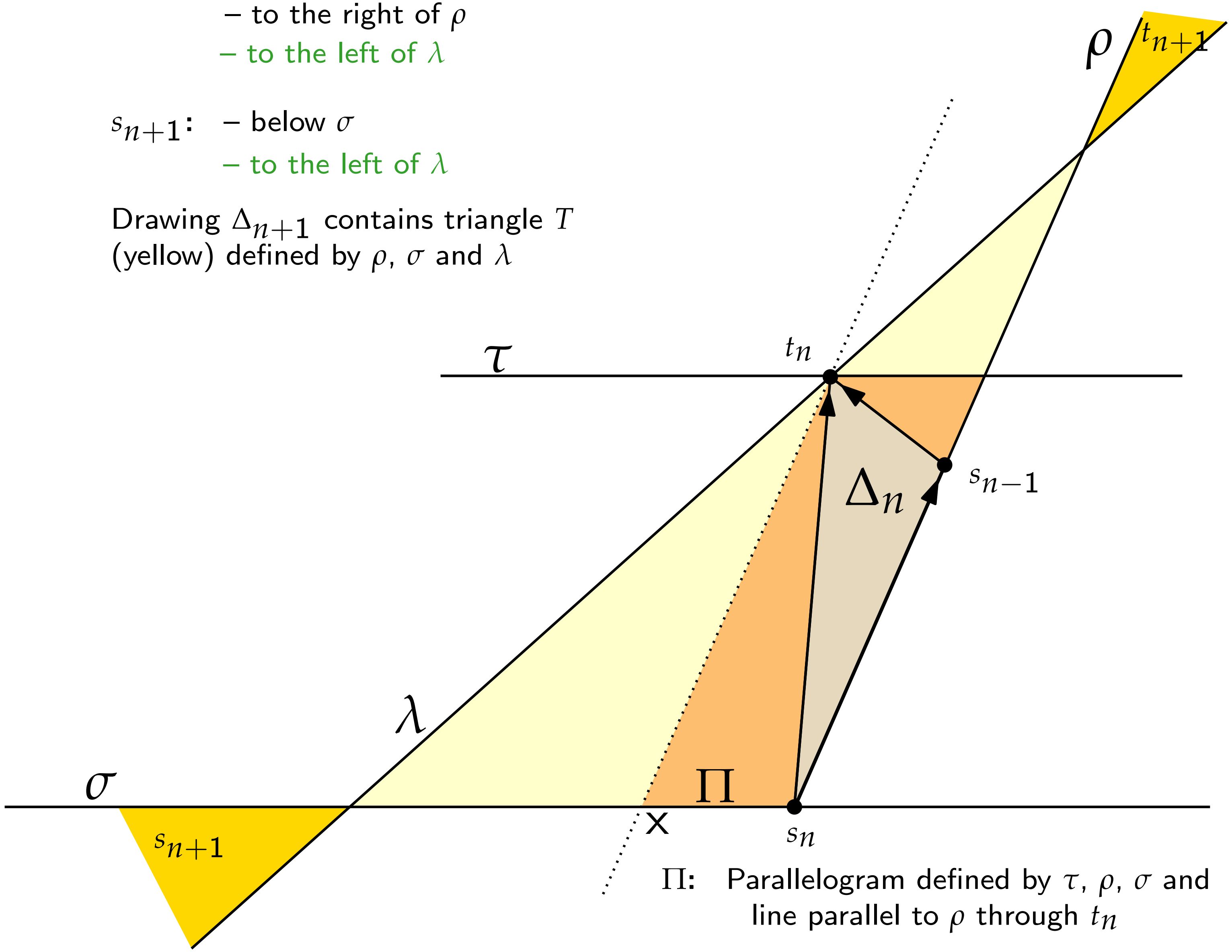
- $2 \cdot Area(\Delta_n) < Area(\Pi)$
  [$\overline{s_n, t_n}$ is the diagonal of Π]

t_{n+1}: – above τ
– to the right of ρ
– to the left of λ

s_{n+1}: – below σ
– to the left of λ

Drawing Δ_{n+1} contains triangle T (yellow) defined by ρ, σ and λ





Π: Parallelogram defined by τ, ρ, σ and line parallel to ρ through t_n

Series-parallel graphs – fixed embedding

Proof:

- $2 \cdot Area(\Delta_n) < Area(\Pi)$

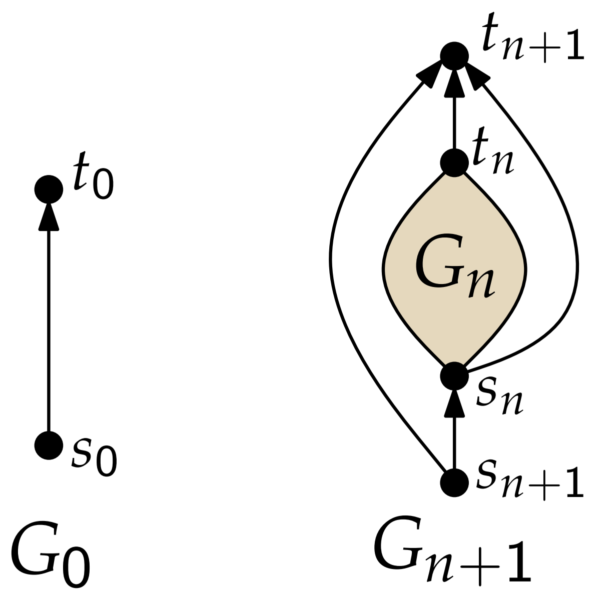
  [$\overline{s_n, t_n}$ is the diagonal of Π]

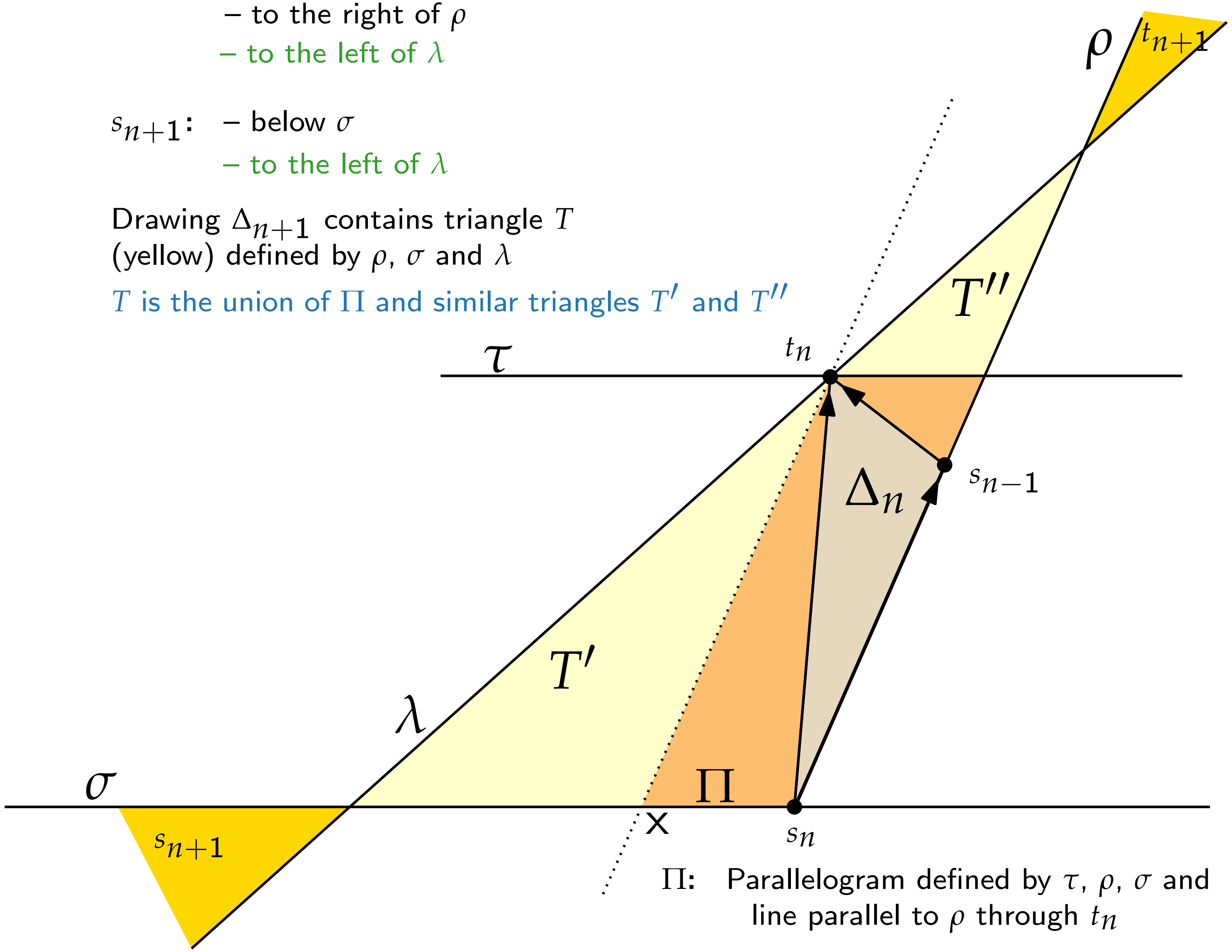
t_{n+1}: – above τ
– to the right of ρ
– to the left of λ

s_{n+1}: – below σ
– to the left of λ

Drawing Δ_{n+1} contains triangle T (yellow) defined by ρ, σ and λ

T is the union of Π and similar triangles T' and T''



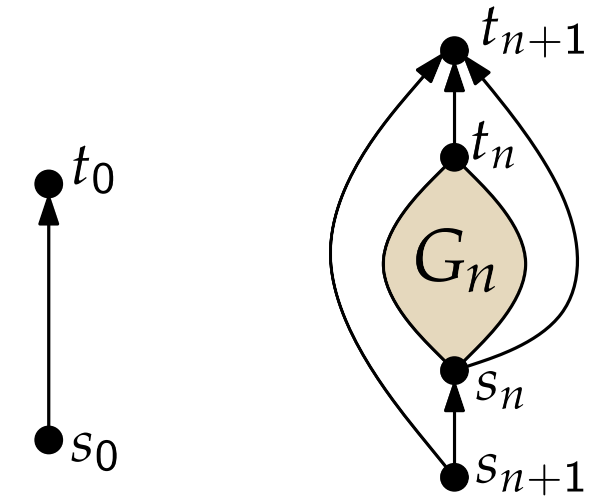


Π: Parallelogram defined by τ, ρ, σ and line parallel to ρ through t_n

Series-parallel graphs – fixed embedding

Proof:

- $2 \cdot Area(\Delta_n) < Area(\Pi)$
  [$\overline{s_n, t_n}$ is the diagonal of Π]



G_0 G_{n+1}

t_{n+1}: – above τ
– to the right of ρ
– to the left of λ

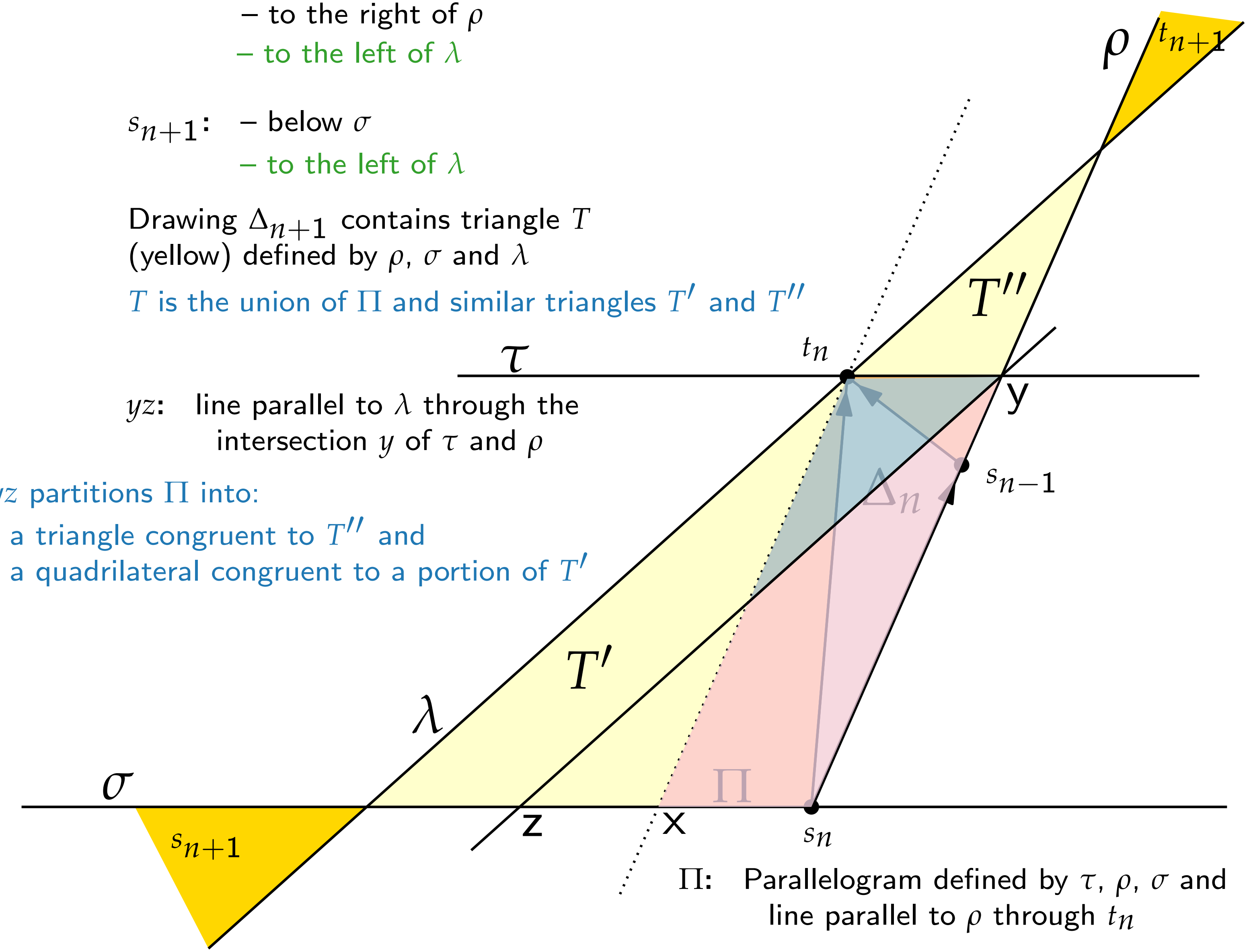
s_{n+1}: – below σ
– to the left of λ

Drawing Δ_{n+1} contains triangle T (yellow) defined by ρ, σ and λ

T is the union of Π and similar triangles T' and T''

yz: line parallel to λ through the intersection y of τ and ρ

yz partitions Π into:
a triangle congruent to T'' and
a quadrilateral congruent to a portion of T'



Π: Parallelogram defined by τ, ρ, σ and line parallel to ρ through t_n

Series-parallel graphs – fixed embedding

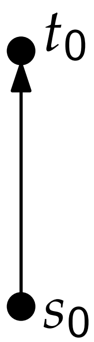
Proof:

- $2 \cdot Area(\Delta_n) < Area(\Pi)$

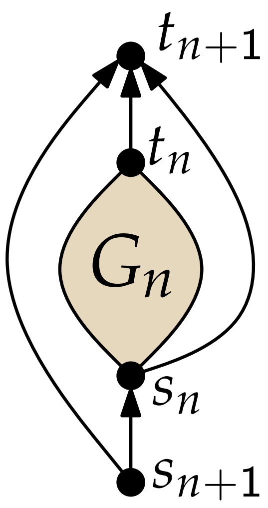
  [$\overline{s_n, t_n}$ is the diagonal of Π]

- $2 \cdot Area(\Pi) \leq Area(\Delta_{n+1})$

  $Area(T) \leq Area(\Delta_{n+1})$
  $Area(T) \geq 2\dot{A}rea(\Pi)$



G_0



G_{n+1}

t_{n+1}: – above τ
– to the right of ρ
– to the left of λ

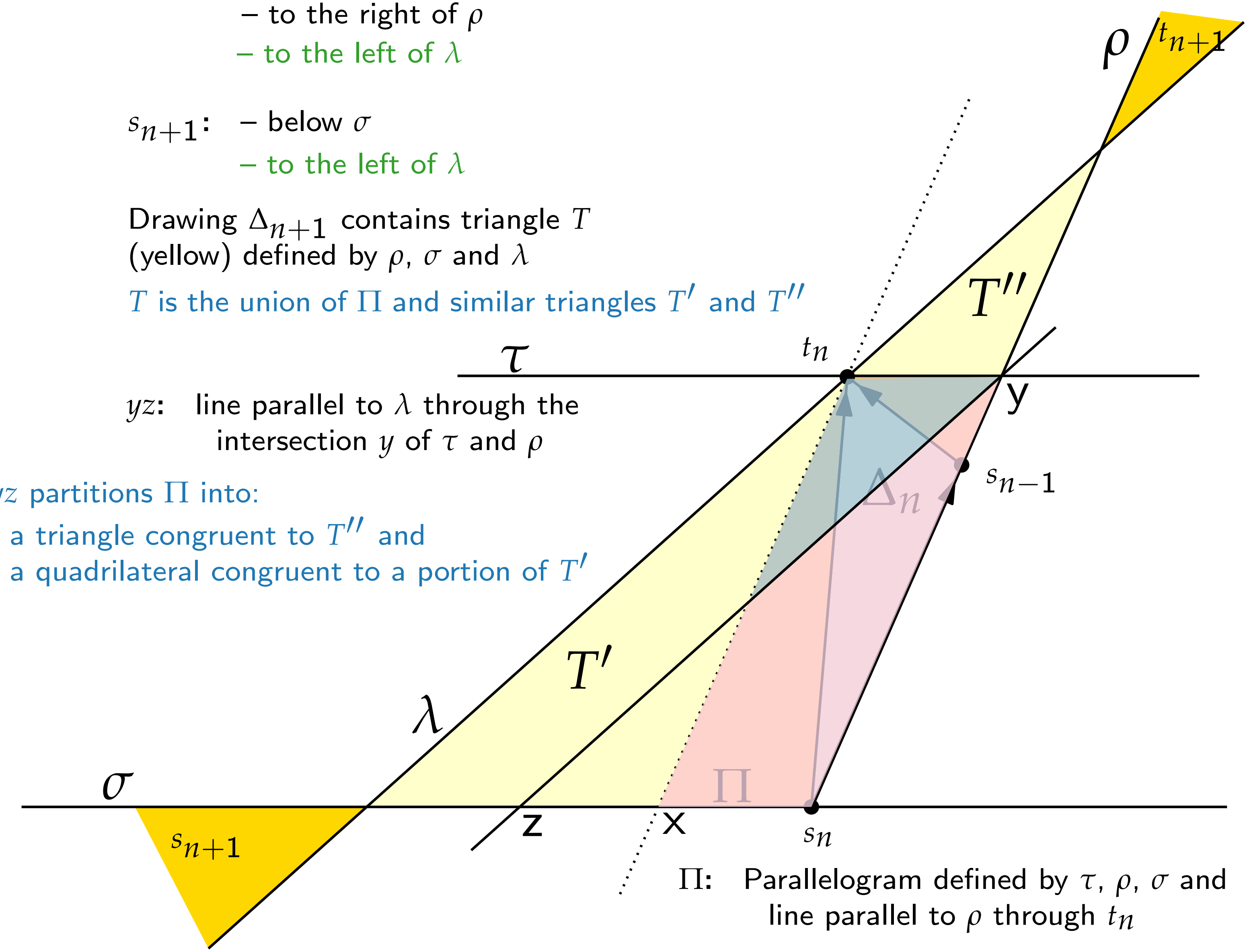
s_{n+1}: – below σ
– to the left of λ

Drawing Δ_{n+1} contains triangle T (yellow) defined by ρ, σ and λ

T is the union of Π and similar triangles T' and T''

yz: line parallel to λ through the intersection y of τ and ρ

yz partitions Π into:
a triangle congruent to T'' and
a quadrilateral congruent to a portion of T'



Π: Parallelogram defined by τ, ρ, σ and line parallel to ρ through t_n

Series-parallel graphs – fixed embedding

Proof:

- $2 \cdot Area(\Delta_n) < Area(\Pi)$

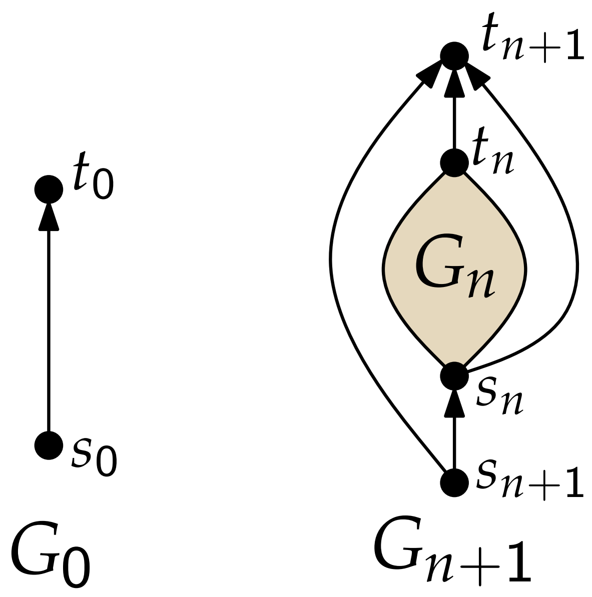
  [$\overline{s_n, t_n}$ is the diagonal of Π]

- $2 \cdot Area(\Pi) \leq Area(\Delta_{n+1})$

  $Area(T) \leq Area(\Delta_{n+1})$

  $Area(T) \geq 2\dot{A}rea(\Pi)$

- $4 \cdot Area(\Delta_n) \leq Area(\Delta_{n+1})$



t_{n+1}: – above τ
– to the right of ρ
– to the left of λ

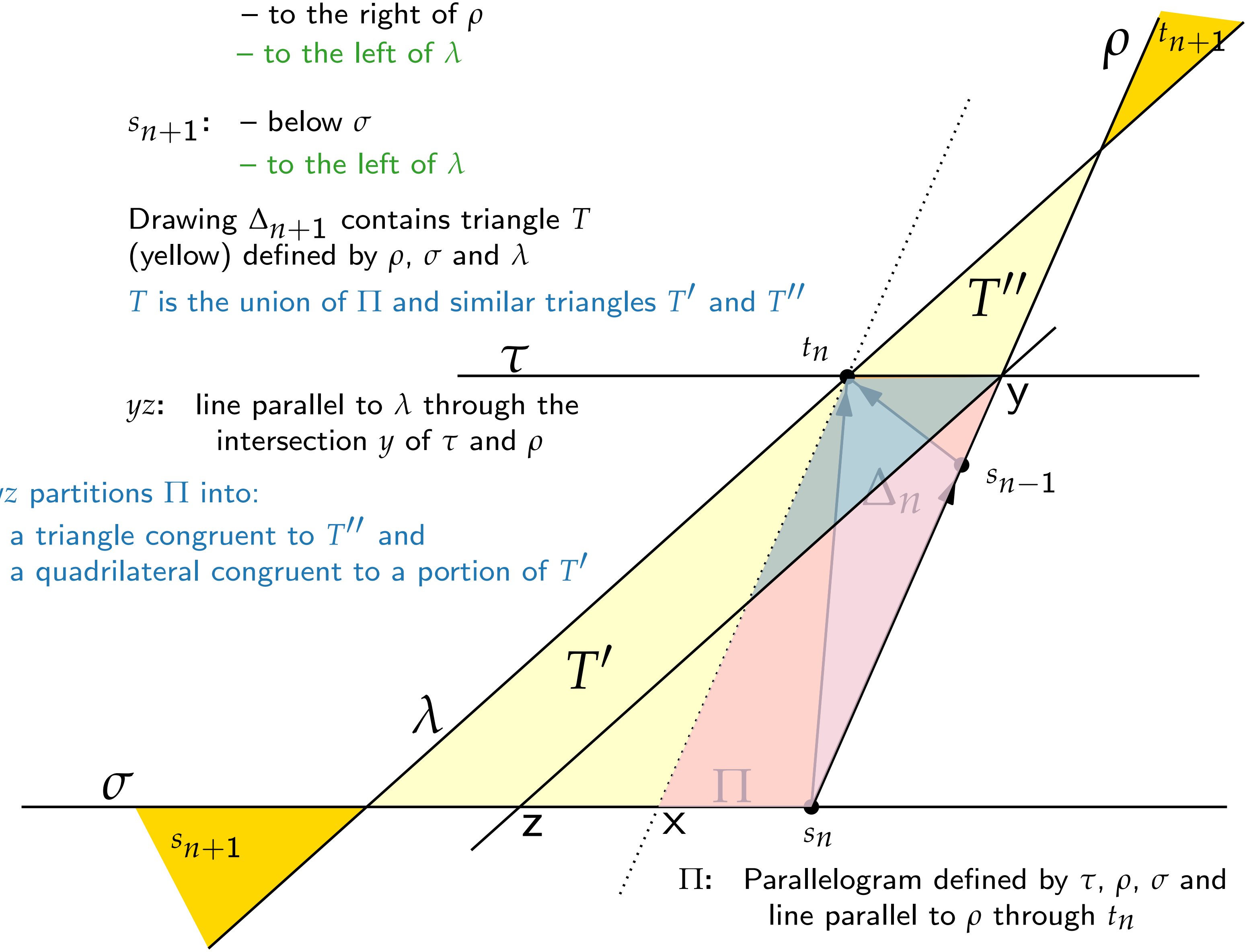
s_{n+1}: – below σ
– to the left of λ

Drawing Δ_{n+1} contains triangle T (yellow) defined by ρ, σ and λ

T is the union of Π and similar triangles T' and T''

yz: line parallel to λ through the intersection y of τ and ρ

yz partitions Π into:
a triangle congruent to T'' and
a quadrilateral congruent to a portion of T'



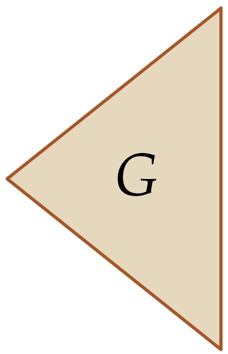
Π: Parallelogram defined by τ, ρ, σ and line parallel to ρ through t_n

∎

Series-parallel graphs – straight-line drawings

Divide & conquer algorithm using the decomposition tree

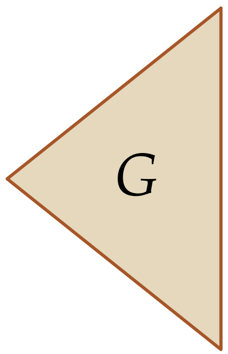
- Draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with no vertex placed at its right corner



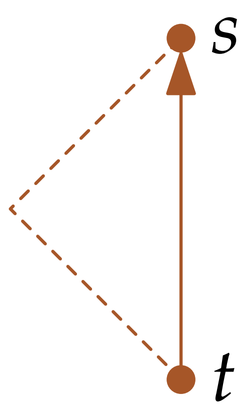
Series-parallel graphs – straight-line drawings

Divide & conquer algorithm using the decomposition tree

- Draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with no vertex placed at its right corner

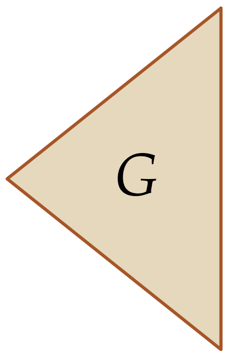


Base case: Q-nodes



Series-parallel graphs – straight-line drawings

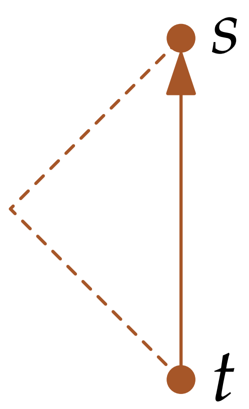
Divide & conquer algorithm using the decomposition tree



- Draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with no vertex placed at its right corner

Base case: Q-nodes

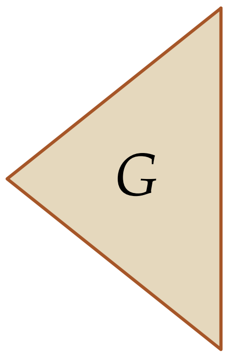
Divide: Draw G_1 and G_2 first



Series-parallel graphs – straight-line drawings

Divide & conquer algorithm using the decomposition tree

- Draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with no vertex placed at its right corner

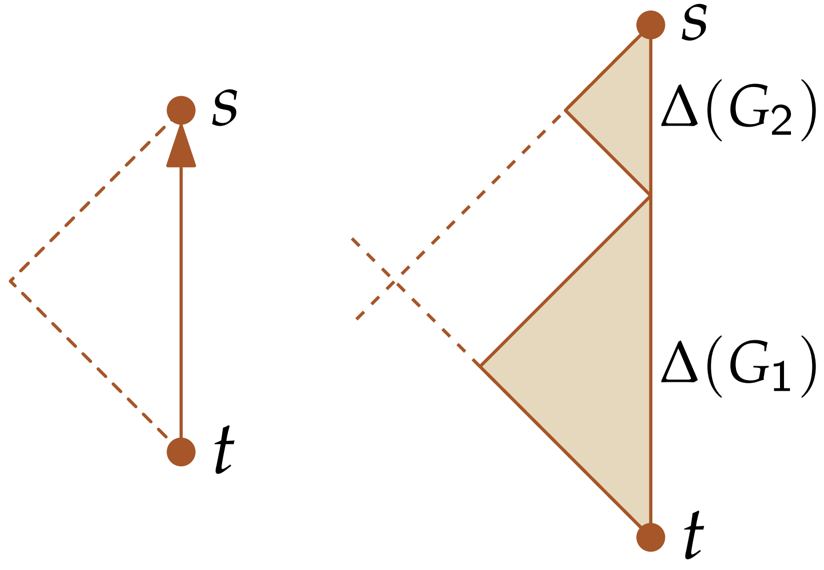


Base case: Q-nodes

Divide: Draw G_1 and G_2 first

Conquer:

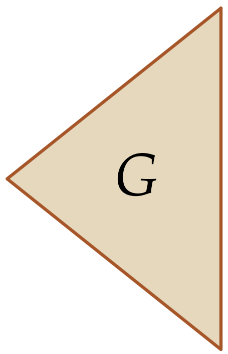
- S-nodes / series composition



Series-parallel graphs – straight-line drawings

Divide & conquer algorithm using the decomposition tree

- Draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with no vertex placed at its right corner

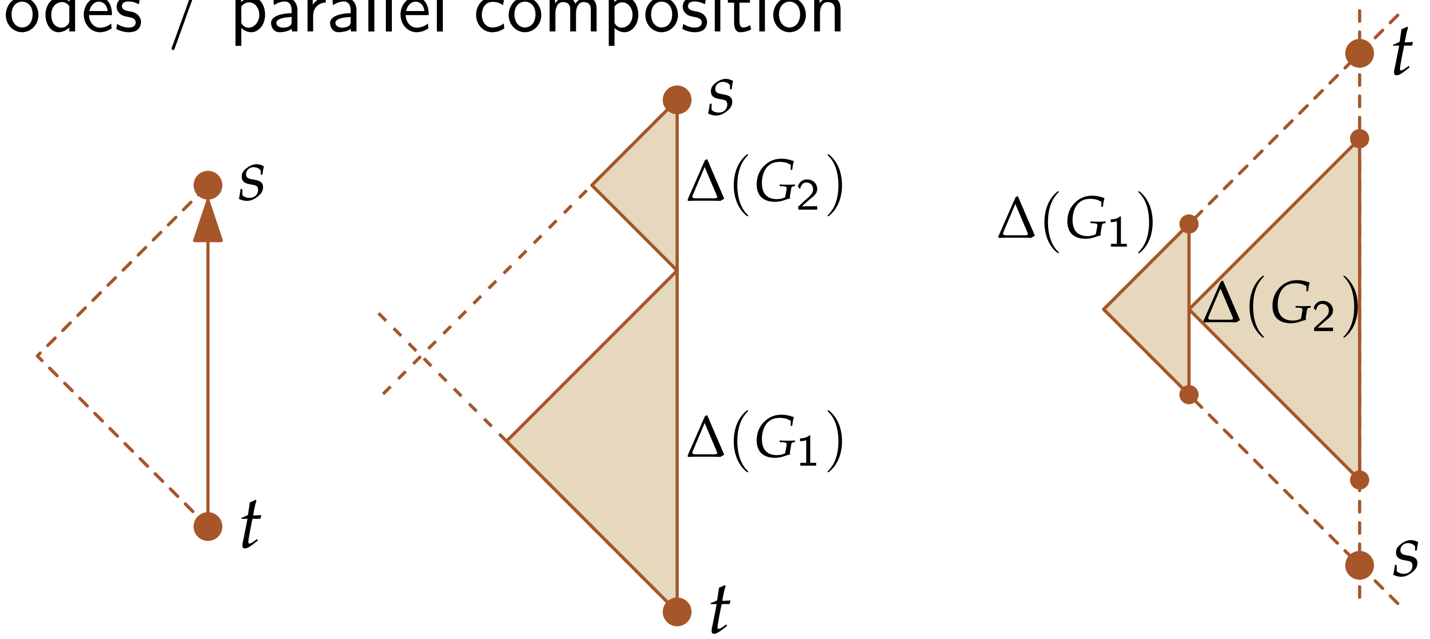


Base case: Q-nodes

Divide: Draw G_1 and G_2 first

Conquer:

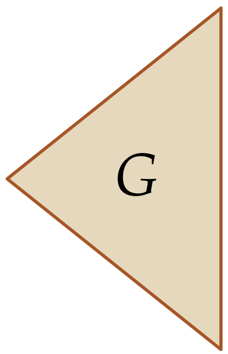
- S-nodes / series composition
- P-nodes / parallel composition



Series-parallel graphs – straight-line drawings

Divide & conquer algorithm using the decomposition tree

- Draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with no vertex placed at its right corner

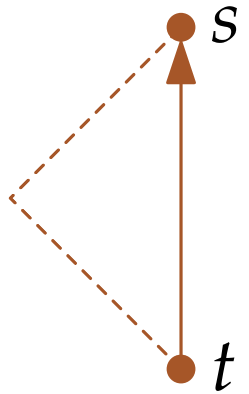


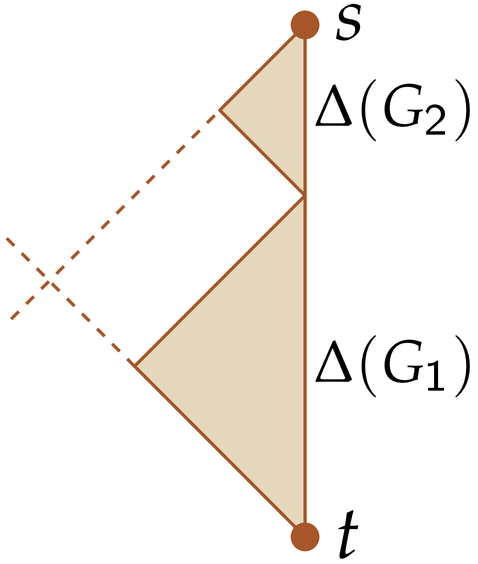
Base case: Q-nodes

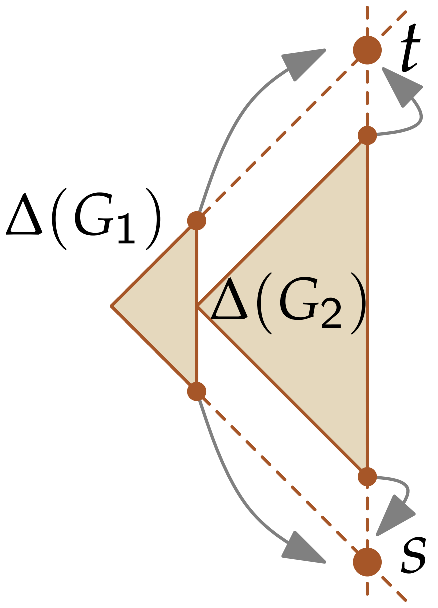
Divide: Draw G_1 and G_2 first

Conquer:

- S-nodes / series composition
- P-nodes / parallel composition



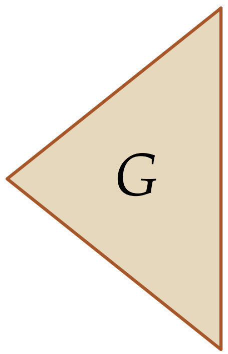




Series-parallel graphs – straight-line drawings

Divide & conquer algorithm using the decomposition tree

- Draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with no vertex placed at its right corner

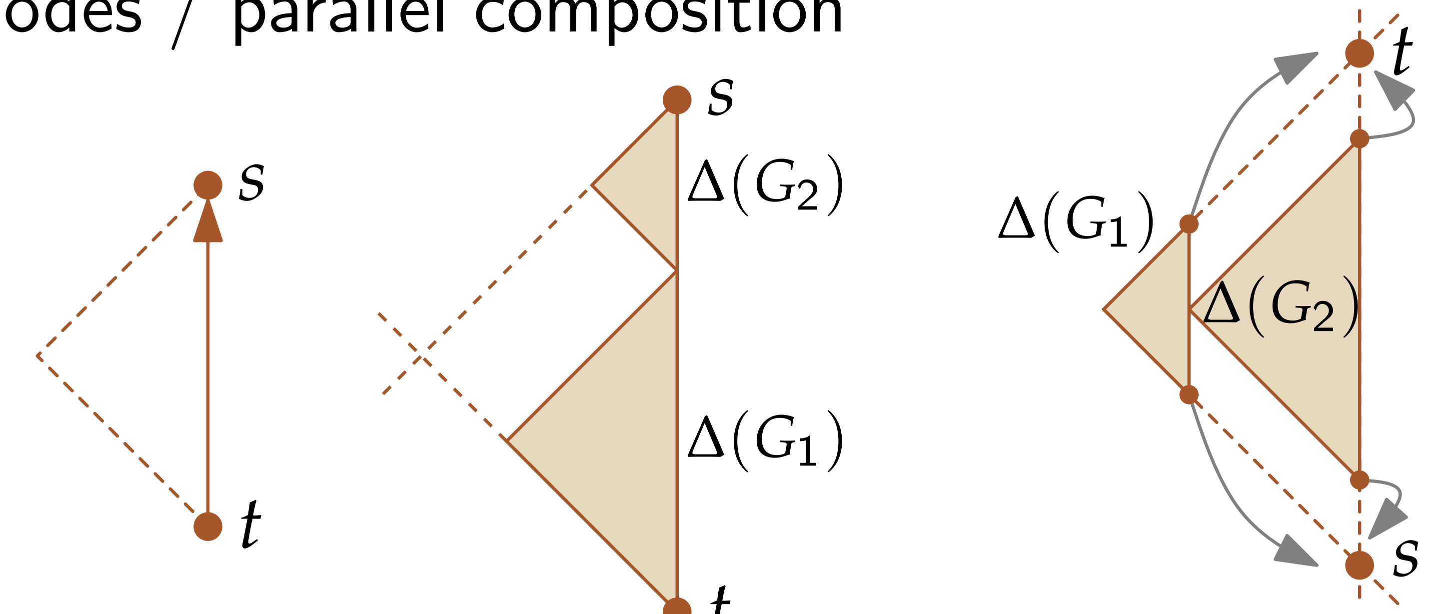


Base case: Q-nodes

Divide: Draw G_1 and G_2 first

Conquer:

- S-nodes / series composition
- P-nodes / parallel composition

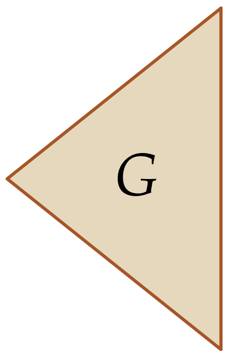


Do you see any problem?

Series-parallel graphs – straight-line drawings

Divide & conquer algorithm using the decomposition tree

- Draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with no vertex placed at its right corner

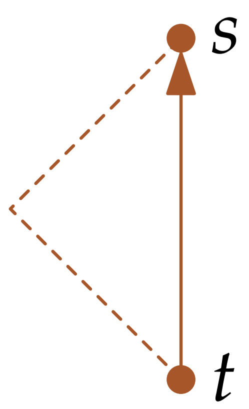


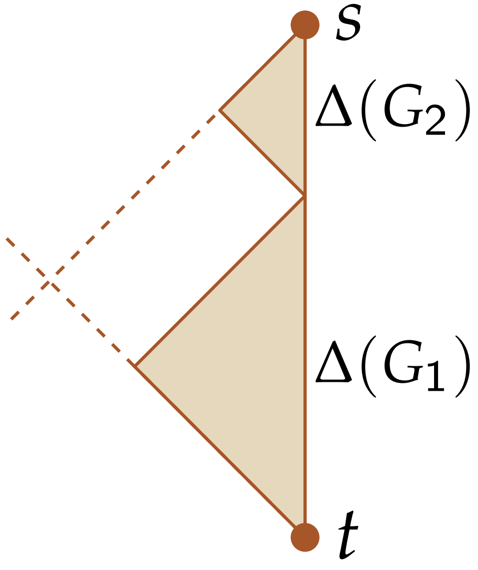
Base case: Q-nodes

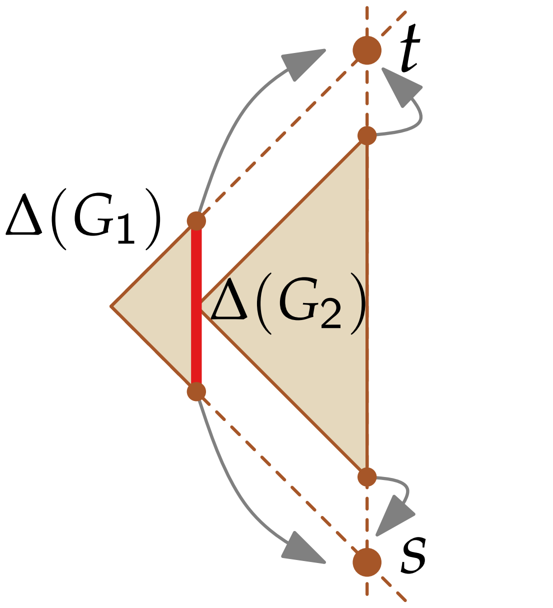
Divide: Draw G_1 and G_2 first

Conquer:

- S-nodes / series composition
- P-nodes / parallel composition



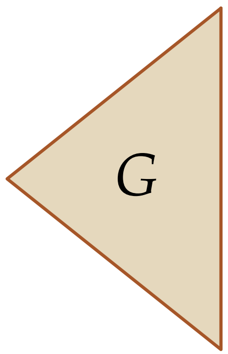




Series-parallel graphs – straight-line drawings

Divide & conquer algorithm using the decomposition tree

- Draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with no vertex placed at its right corner

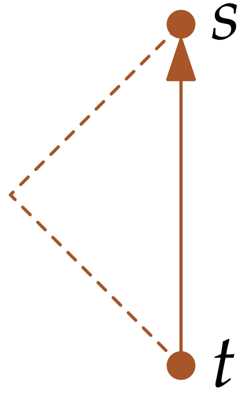


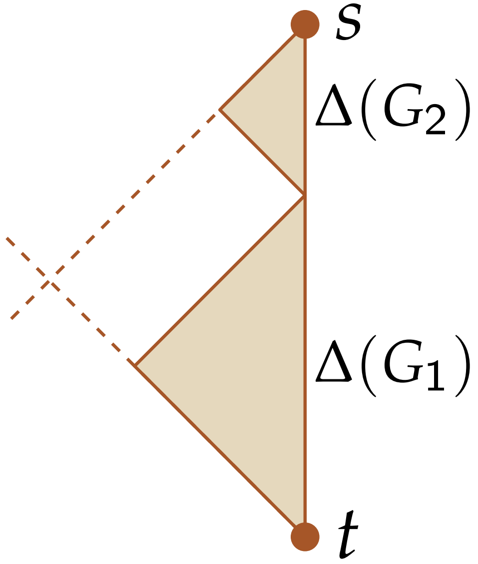
Base case: Q-nodes

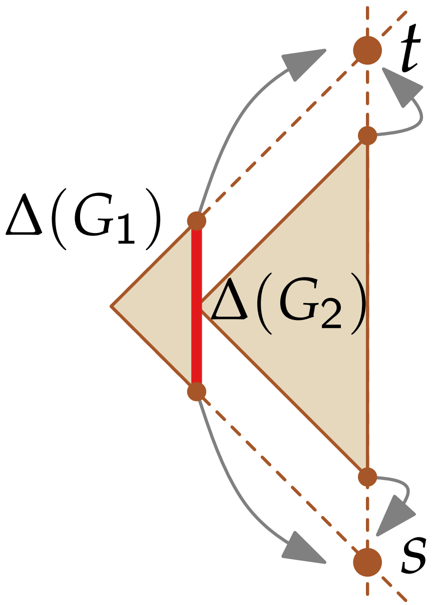
Divide: Draw G_1 and G_2 first

Conquer:

- S-nodes / series composition
- P-nodes / parallel composition





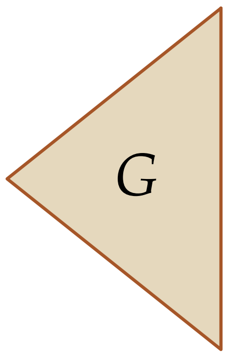


change embedding!

Series-parallel graphs – straight-line drawings

Divide & conquer algorithm using the decomposition tree

- Draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with no vertex placed at its right corner

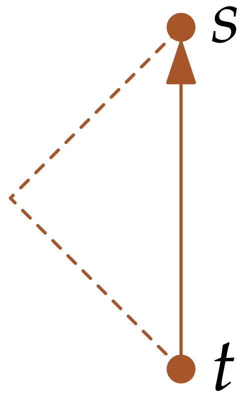


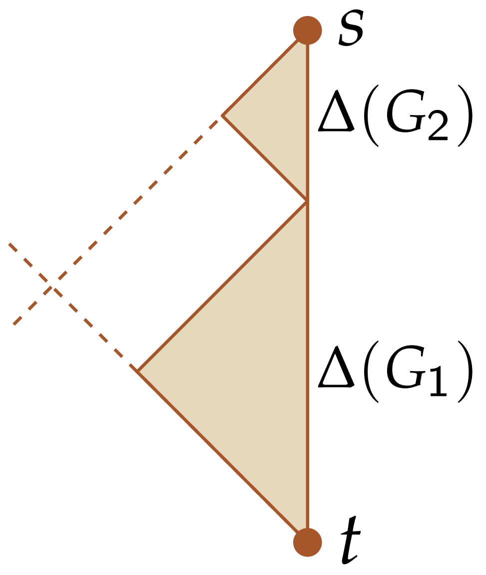
Base case: Q-nodes

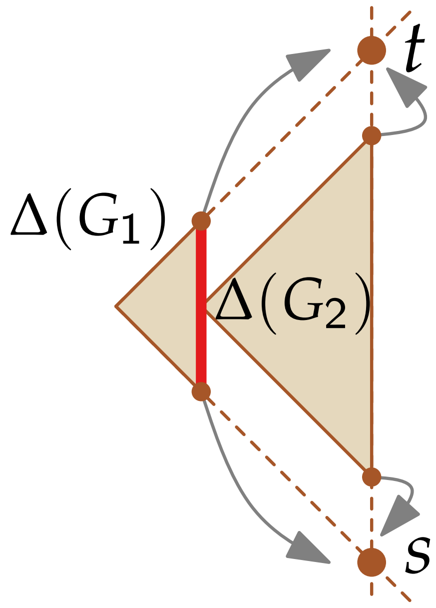
Divide: Draw G_1 and G_2 first

Conquer:

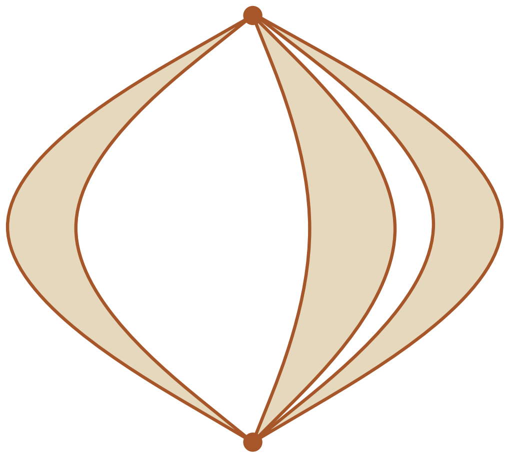
- S-nodes / series composition
- P-nodes / parallel composition







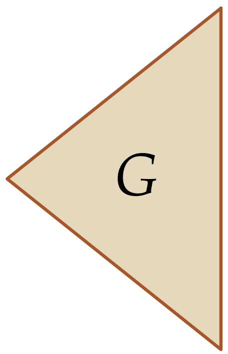
change embedding!



Series-parallel graphs – straight-line drawings

Divide & conquer algorithm using the decomposition tree

- Draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with no vertex placed at its right corner

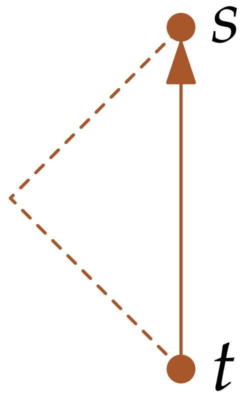


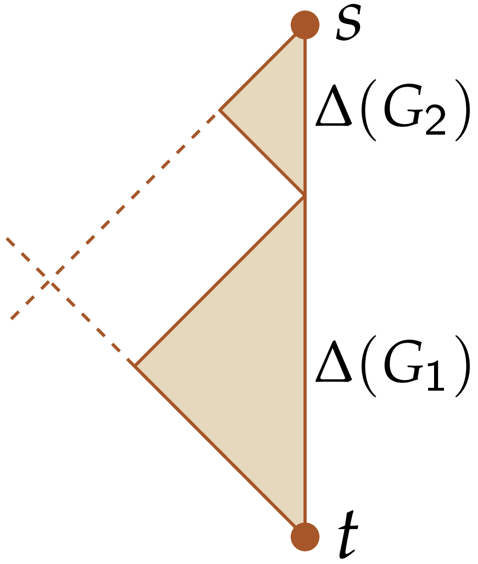
Base case: Q-nodes

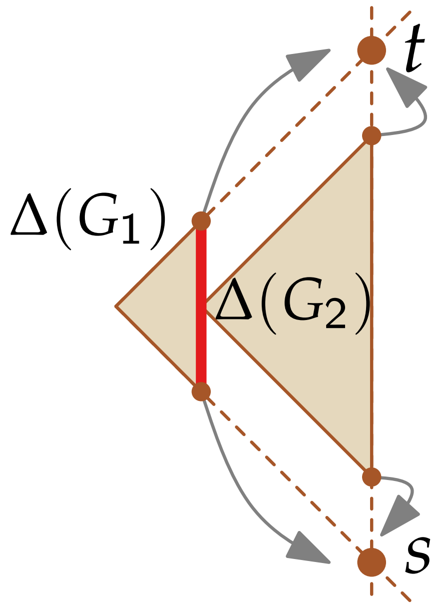
Divide: Draw G_1 and G_2 first

Conquer:

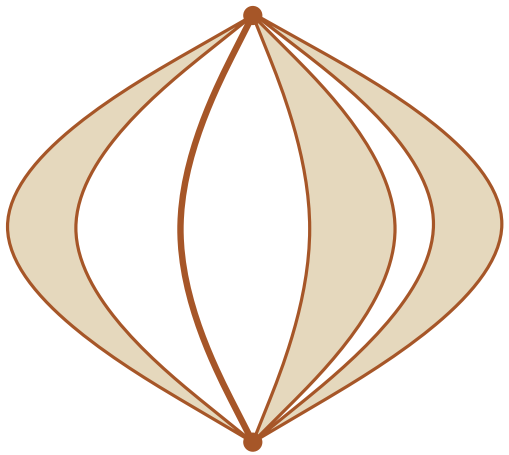
- S-nodes / series composition
- P-nodes / parallel composition







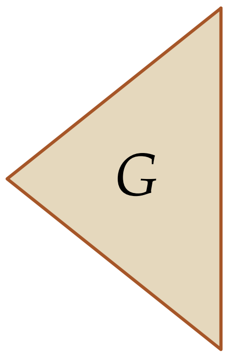
change embedding!



Series-parallel graphs – straight-line drawings

Divide & conquer algorithm using the decomposition tree

- Draw G inside a right-angled isosceles bounding triangle $\Delta(G)$ with no vertex placed at its right corner

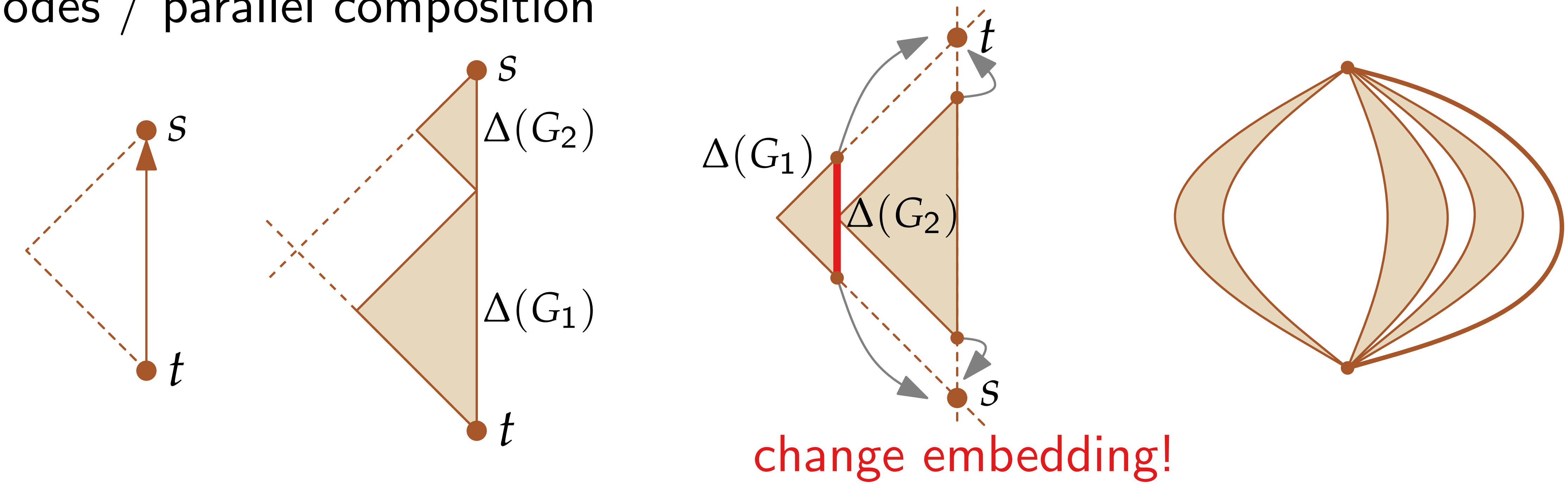


Base case: Q-nodes

Divide: Draw G_1 and G_2 first

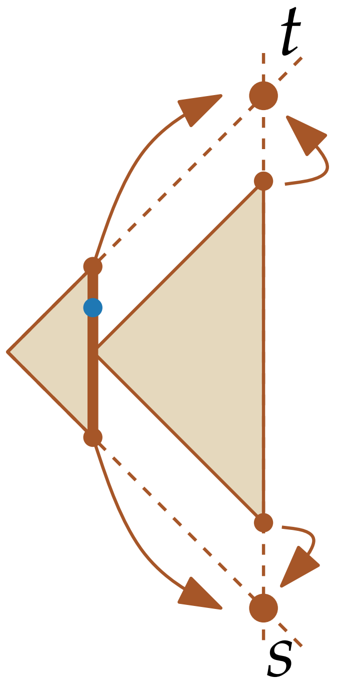
Conquer:

- S-nodes / series composition
- P-nodes / parallel composition



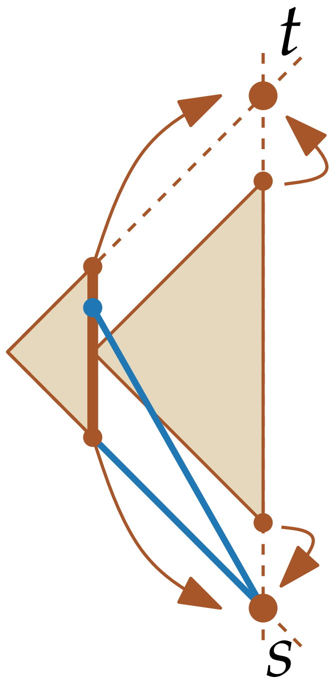
Series-parallel graphs – straight-line drawings

- What makes parallel composition possible without creating crossings?



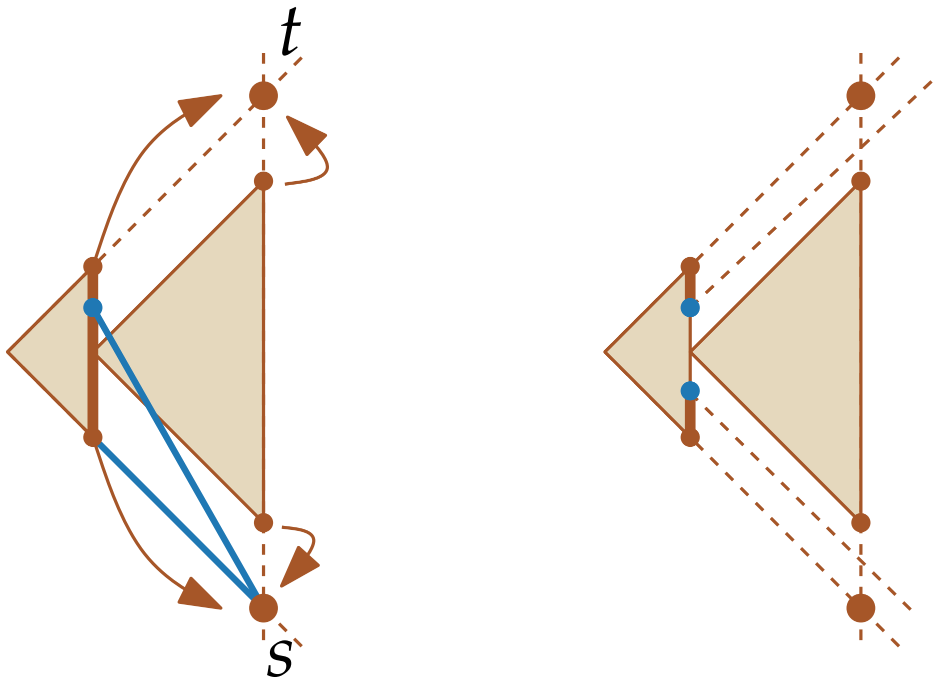
Series-parallel graphs – straight-line drawings

- What makes parallel composition possible without creating crossings?



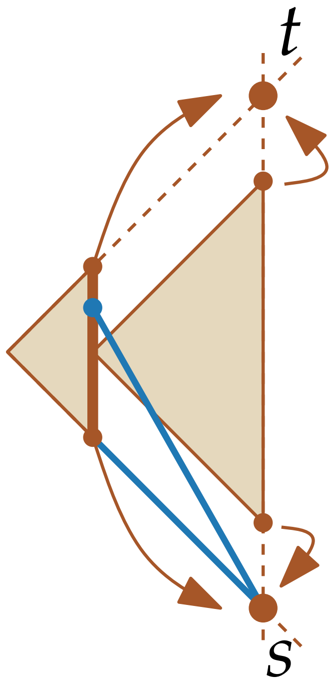
Series-parallel graphs – straight-line drawings

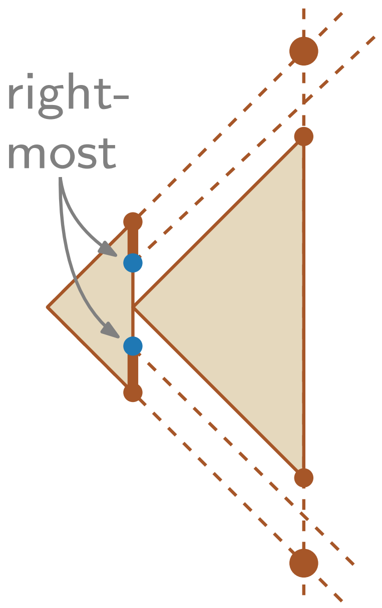
- What makes parallel composition possible without creating crossings?



Series-parallel graphs – straight-line drawings

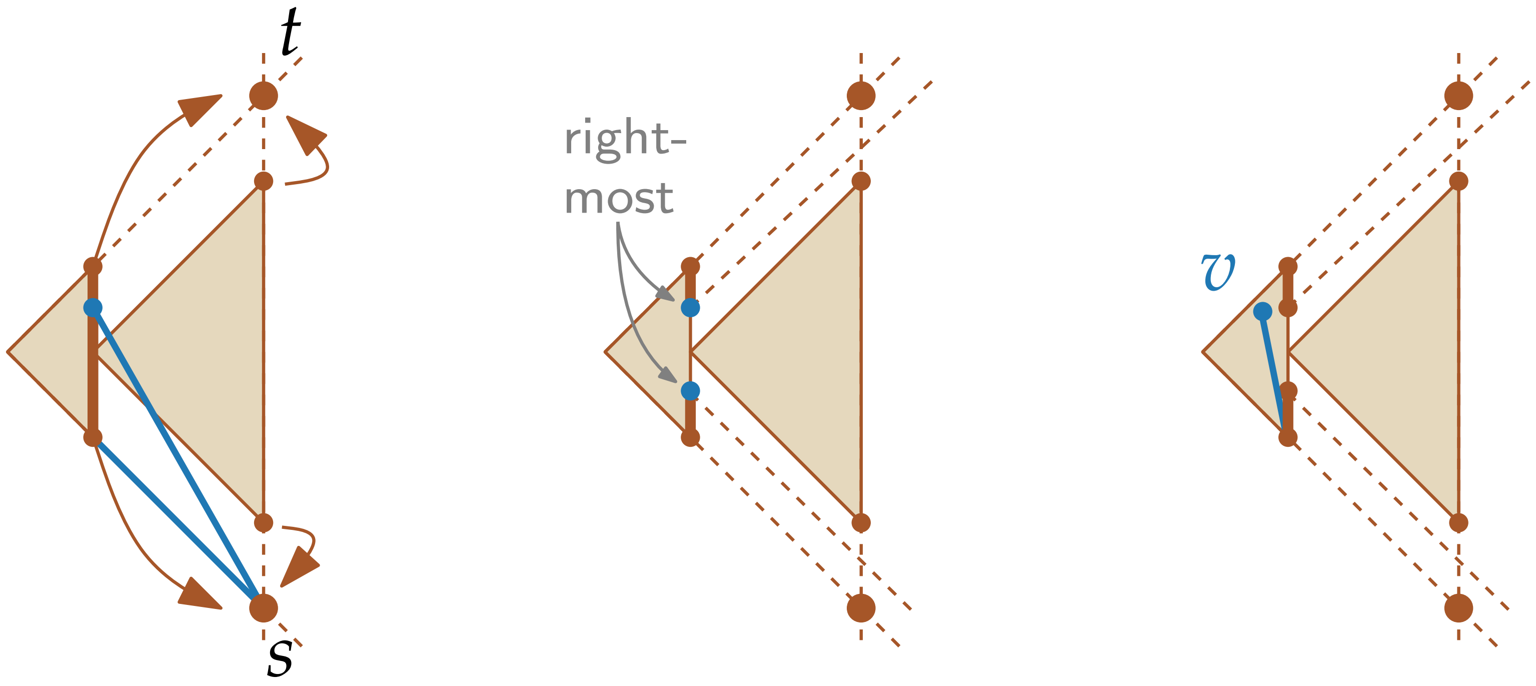
- What makes parallel composition possible without creating crossings?





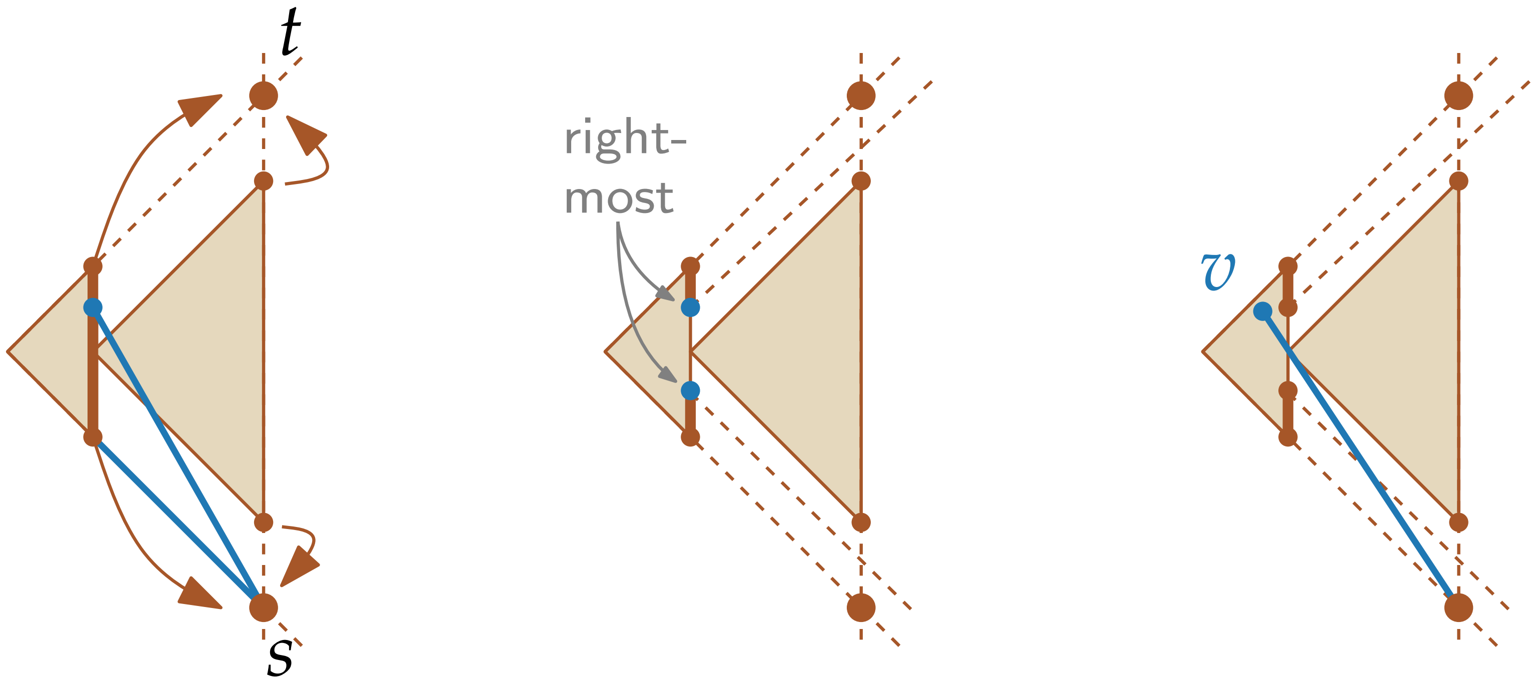
Series-parallel graphs – straight-line drawings

- What makes parallel composition possible without creating crossings?



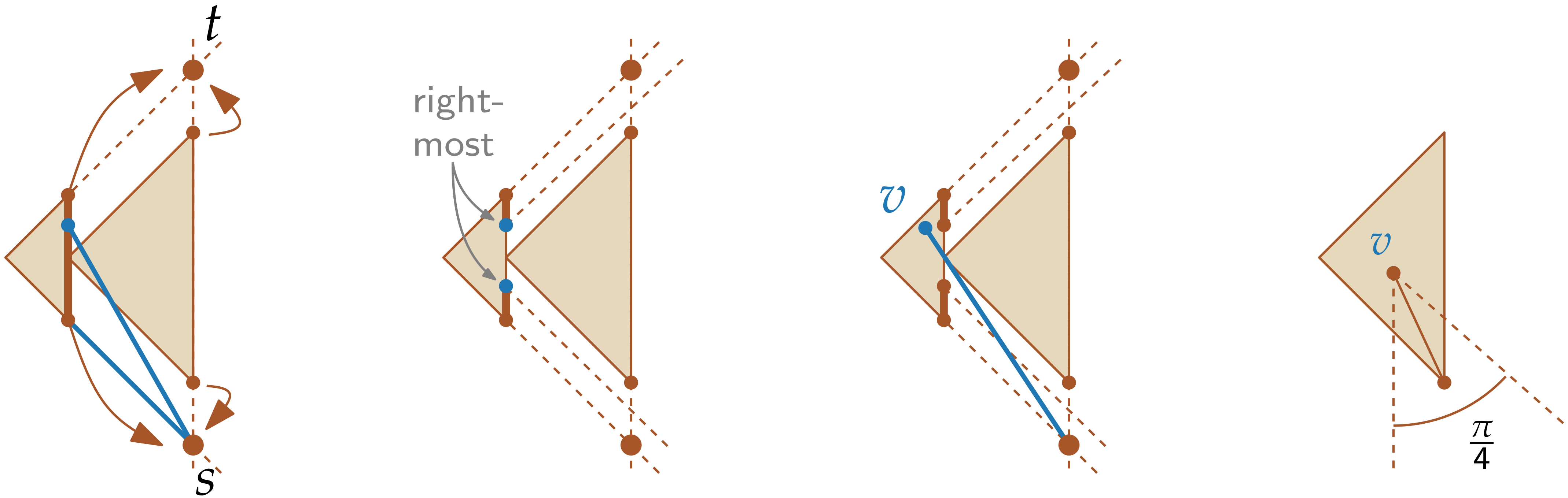
Series-parallel graphs – straight-line drawings

- What makes parallel composition possible without creating crossings?



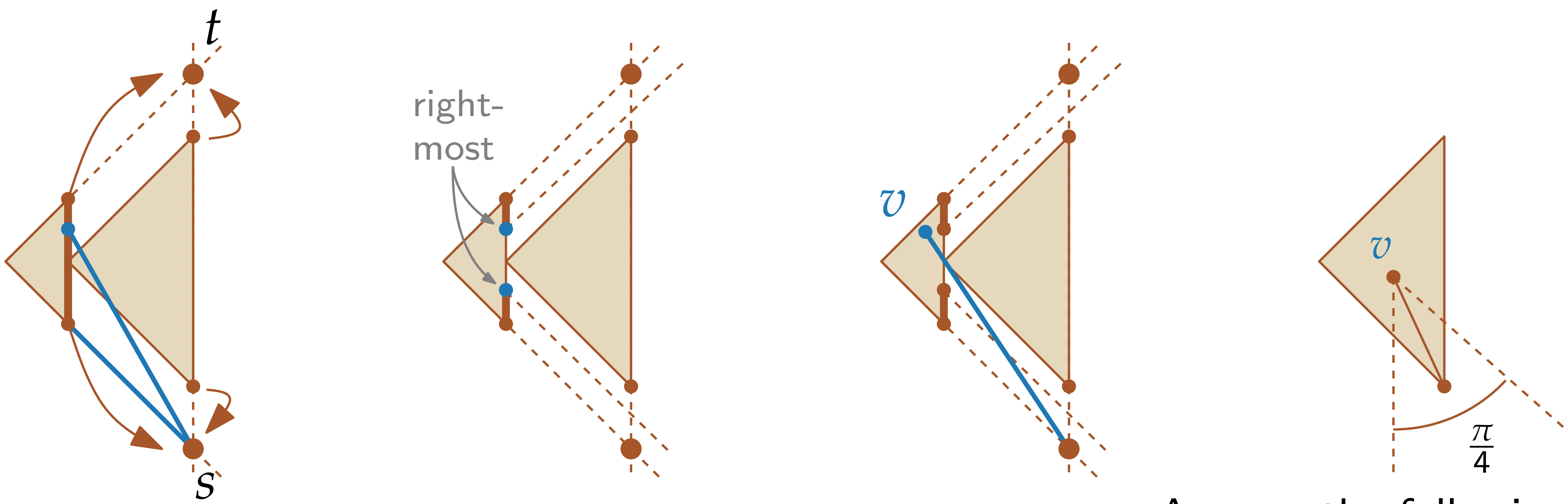
Series-parallel graphs – straight-line drawings

- What makes parallel composition possible without creating crossings?



Series-parallel graphs – straight-line drawings

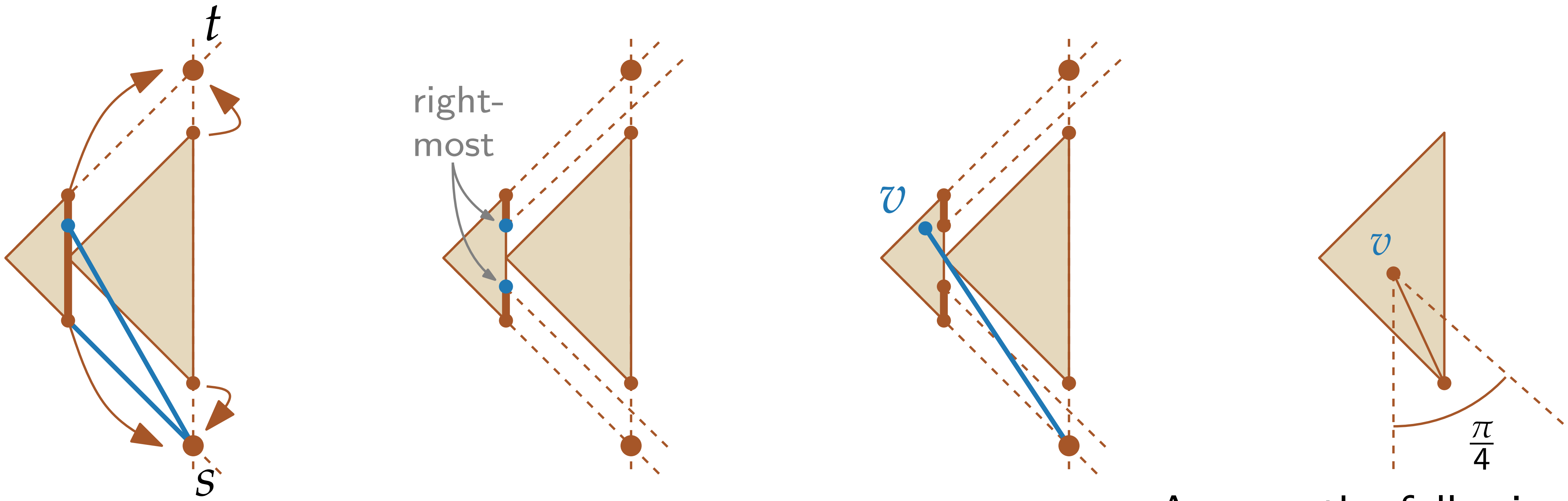
- What makes parallel composition possible without creating crossings?



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

Series-parallel graphs – straight-line drawings

- What makes parallel composition possible without creating crossings?

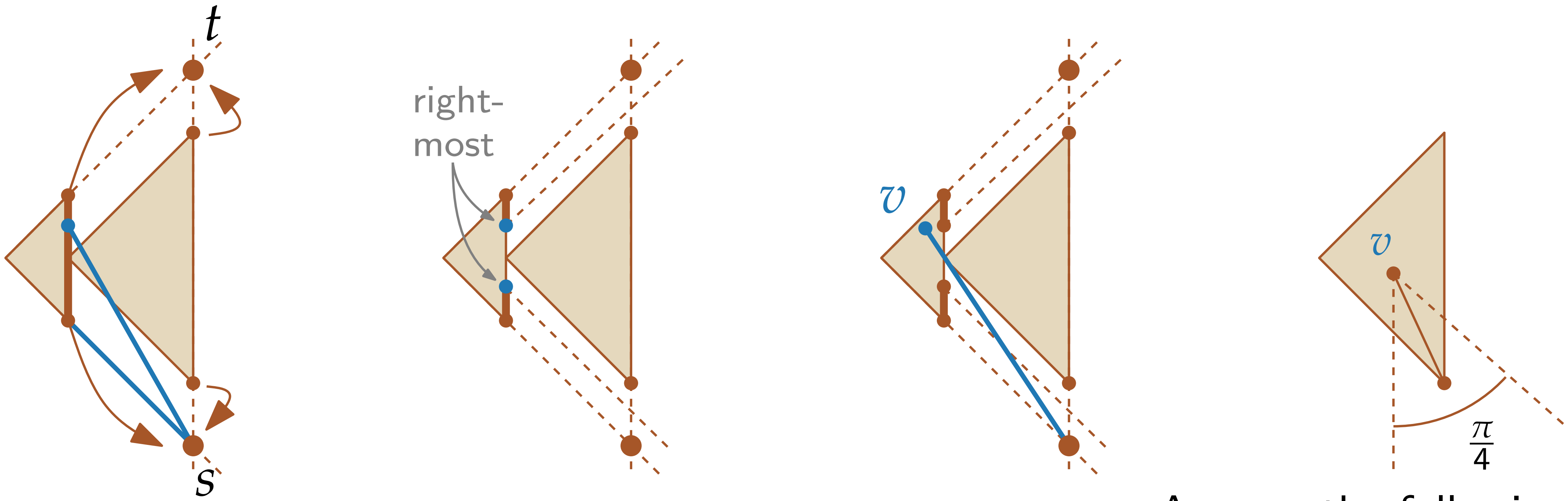


Assume the following holds:
the only vertex in angle(v) is s

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

Series-parallel graphs – straight-line drawings

- What makes parallel composition possible without creating crossings?



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

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

Lemma.
The drawing produced by the algorithm is planar.

Series-parallel graphs – result

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

- is upward planar and
- a straight-line drawing
- with area in $\mathcal{O}(n^2)$
  [$m \times 2m$, where m is the number of edges of G]
- Isomorphic components of G have congruent drawings up to translation.

Γ can be computed in $\mathcal{O}(n)$ time.

Literature

- [GD Ch. 3.2] for divide an conquer mehtods for series-parallel graphs.
- [BC+94] Bertolazzi, Cohen, Di Battista, Tamassia and Tollis, "How to draw a series-parallel digraph", Int. J. of Computational Geometry and Applications, Vol. 4, pp. 385-402, 1994.