On Strict (Outer-)Confluent Graphs

Henry Förster, Robert Ganian, Fabian Klute, Martin Nöllenburg

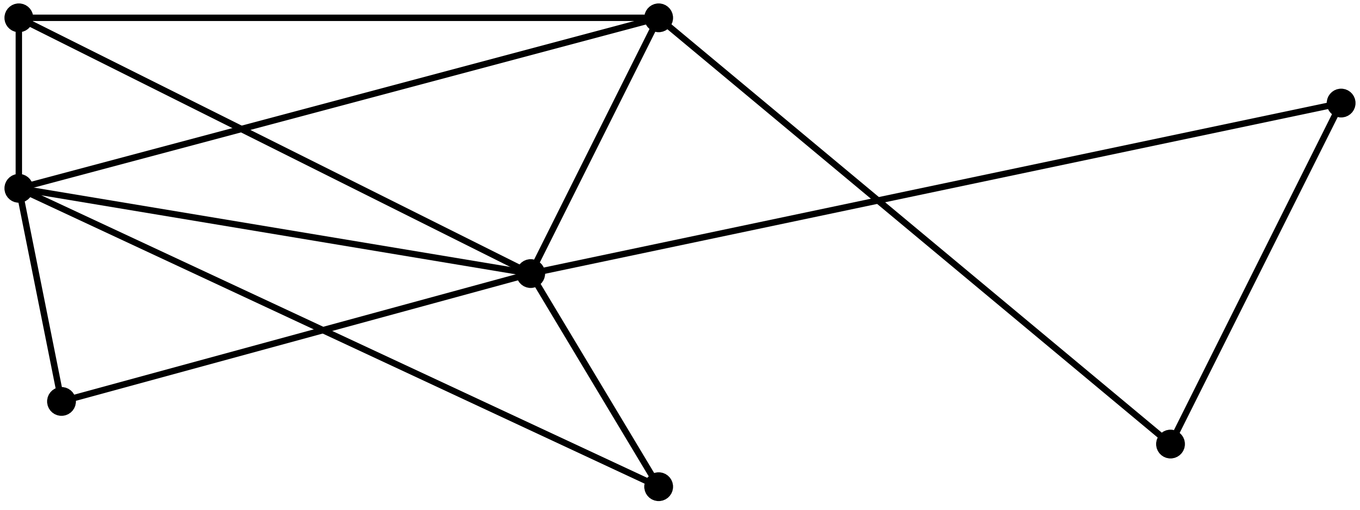
Graph Drawing 2019 · September 18, 2019

ALGORITHMS AND COMPLEXITY GROUP

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

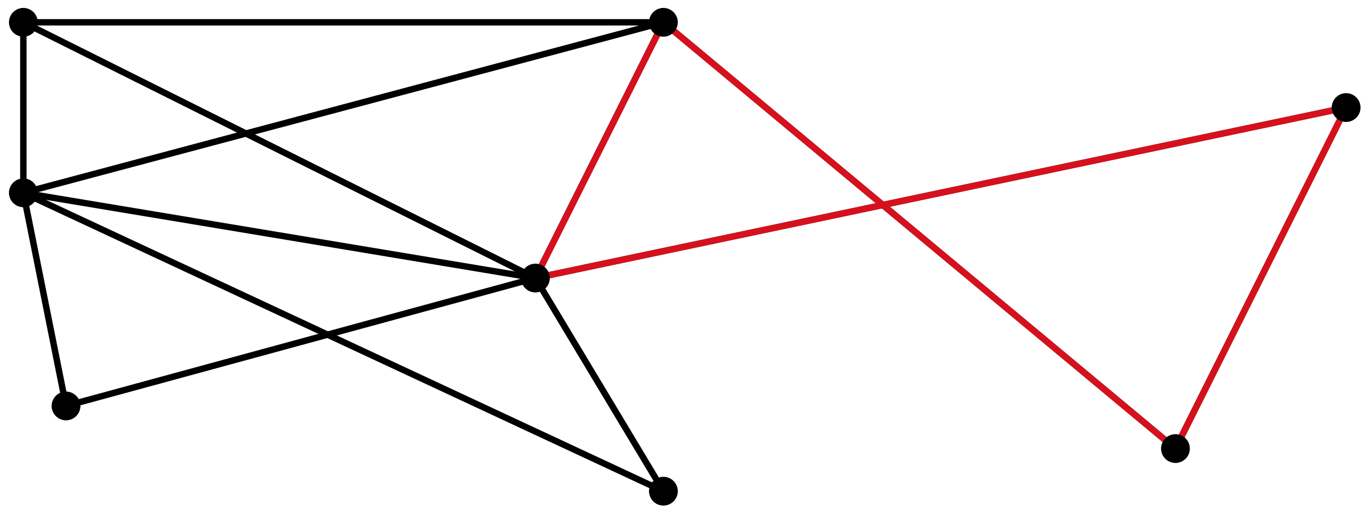
Confluence

Technique to bundle edges



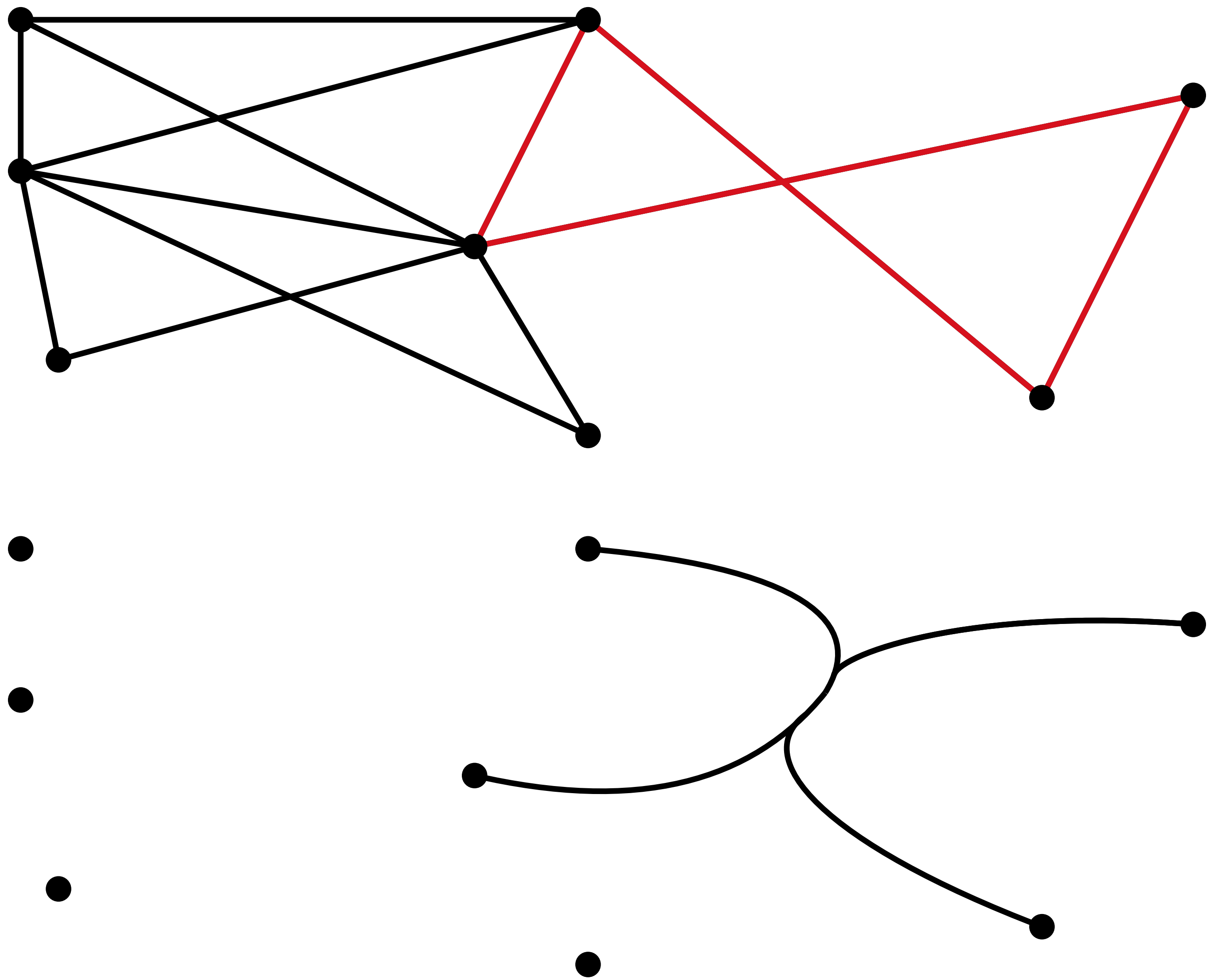
Confluence

Technique to bundle edges



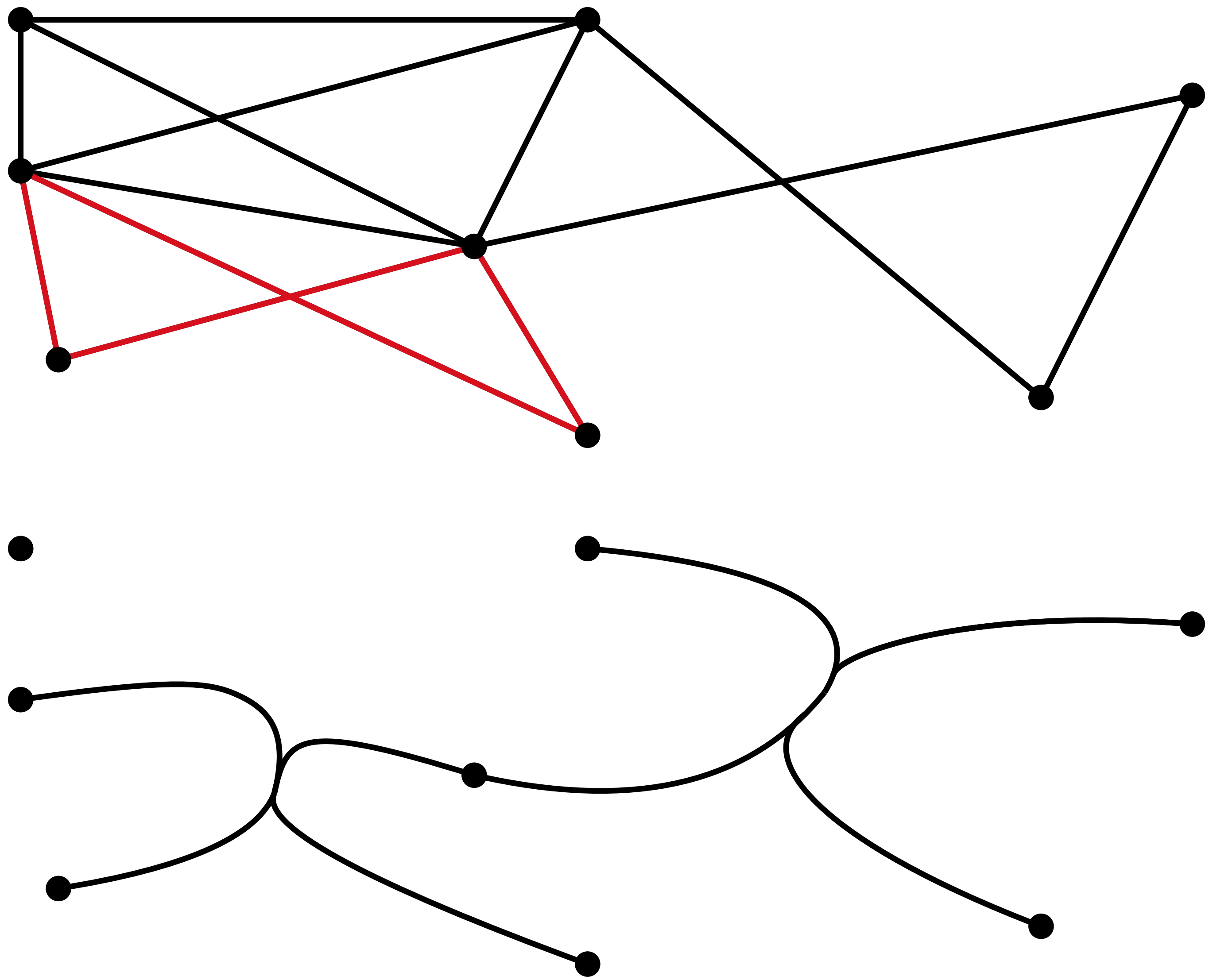
Confluence

Technique to bundle edges



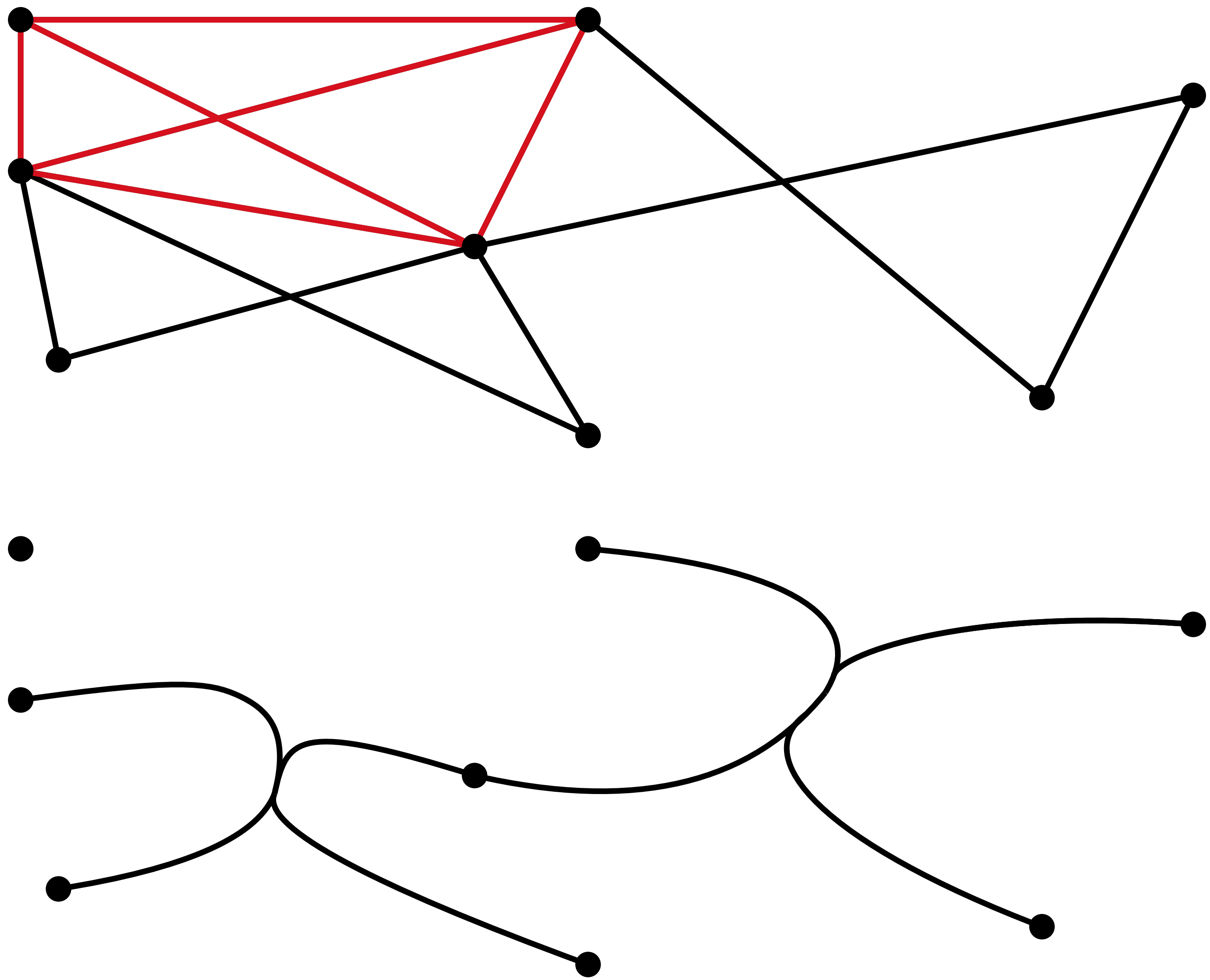
Confluence

Technique to bundle edges



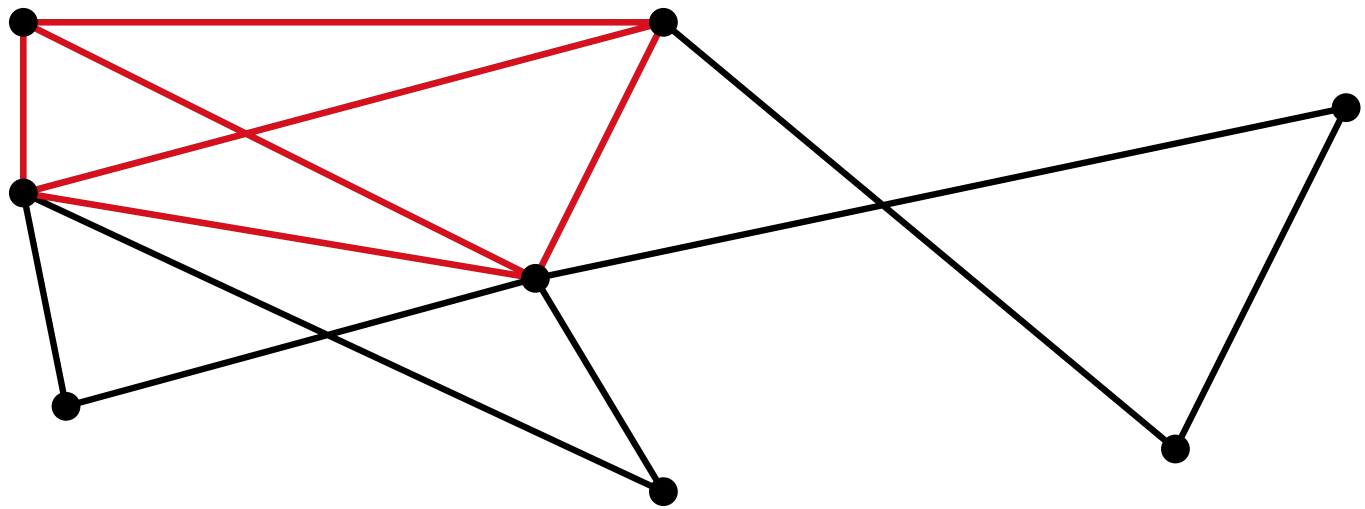
Confluence

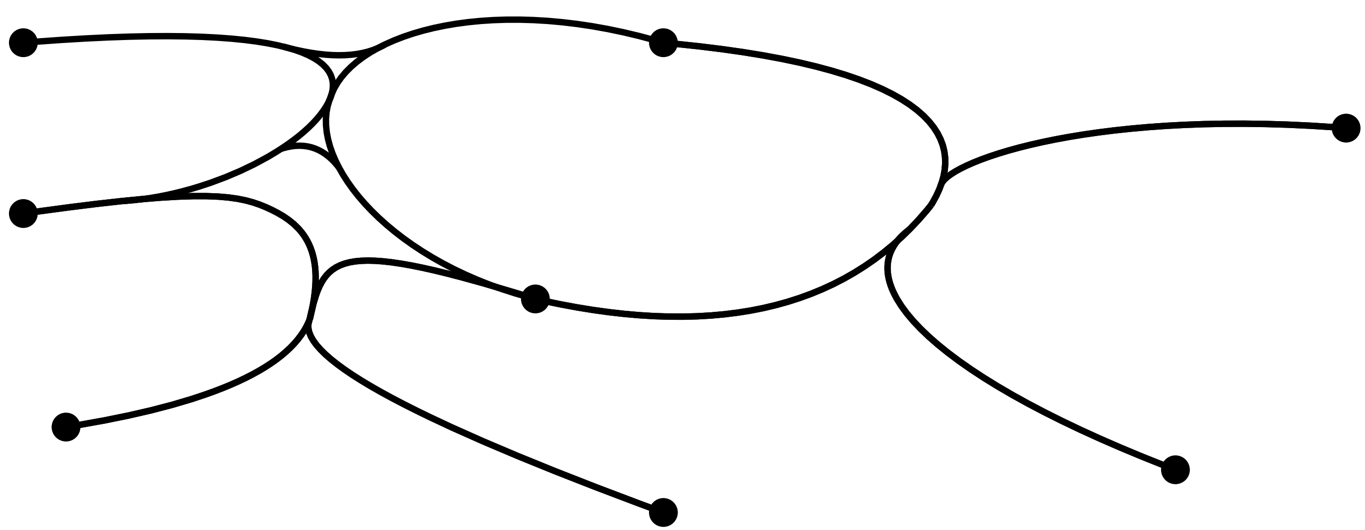
Technique to bundle edges



Confluence

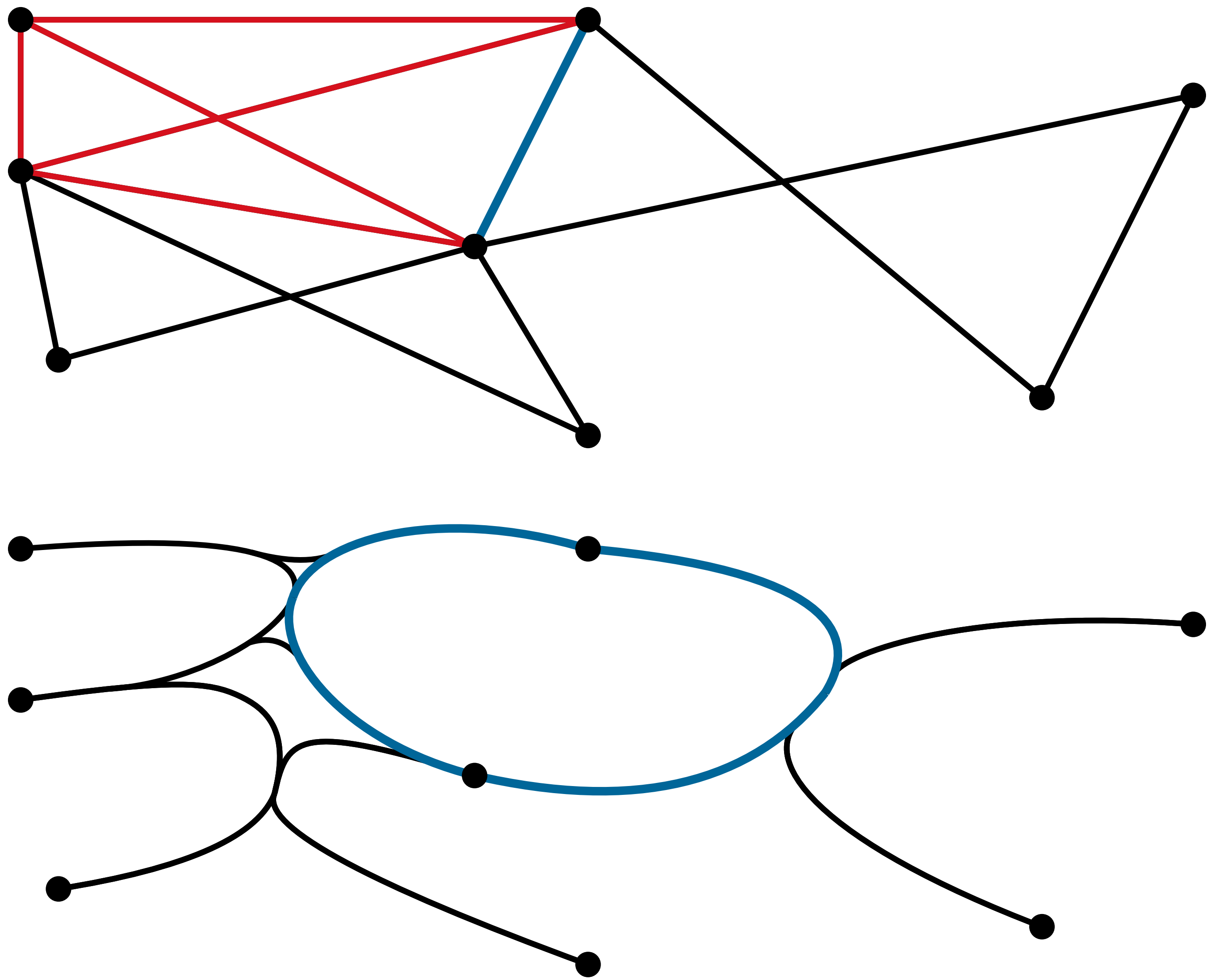
Technique to bundle edges





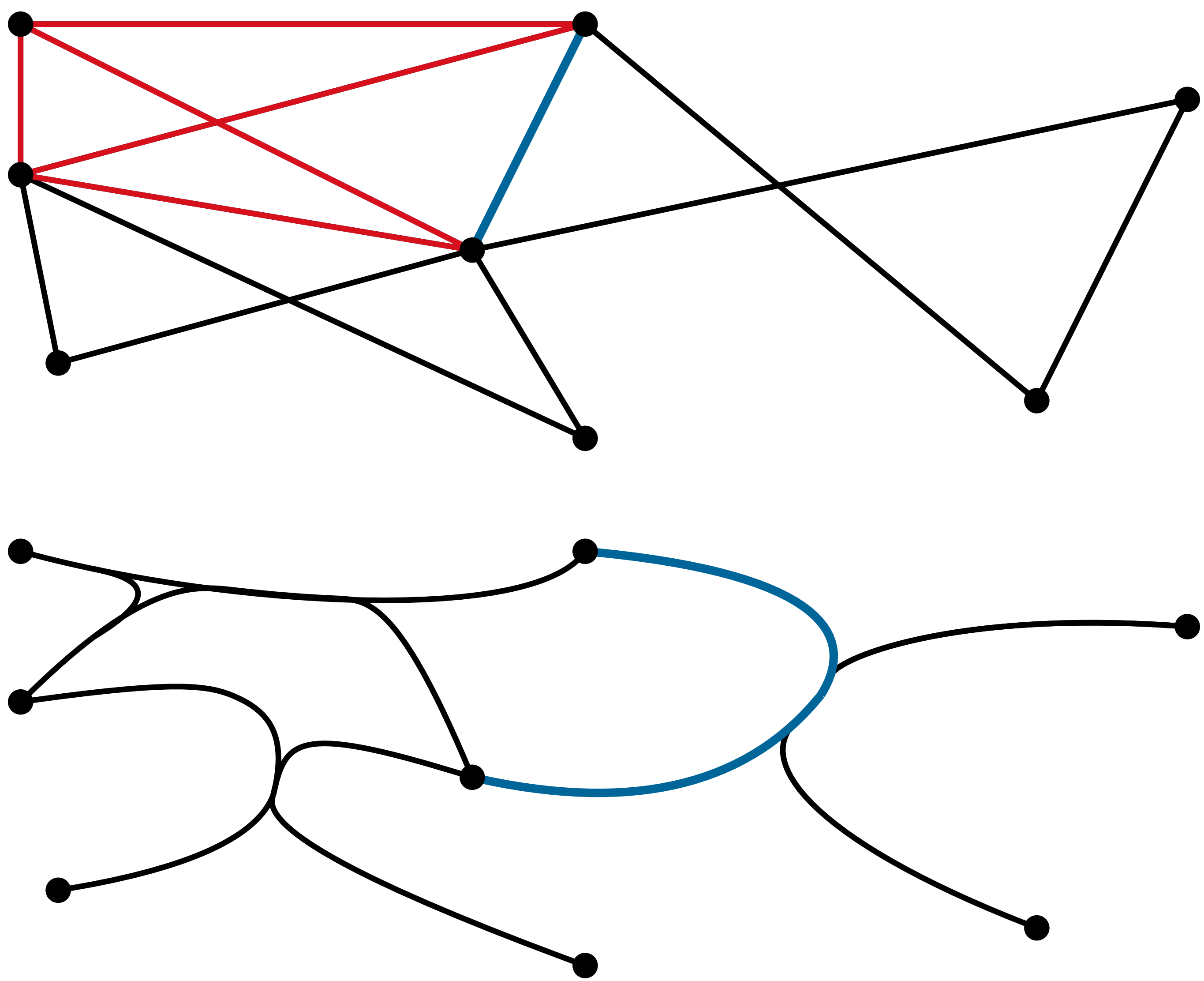
Confluence

Technique to bundle edges

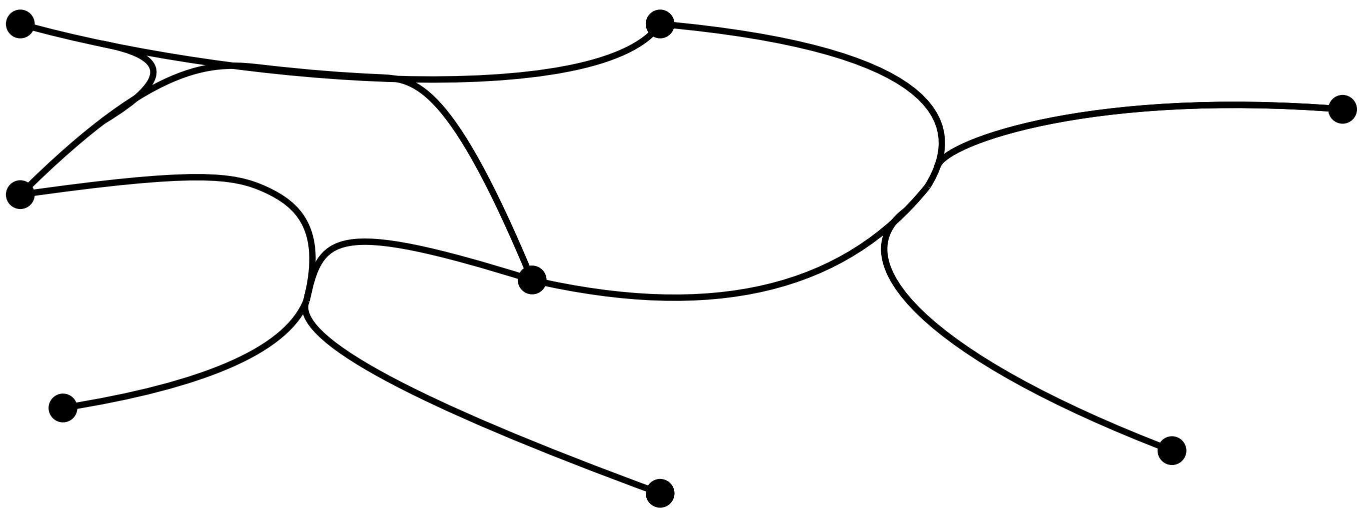


Confluence

Technique to bundle edges



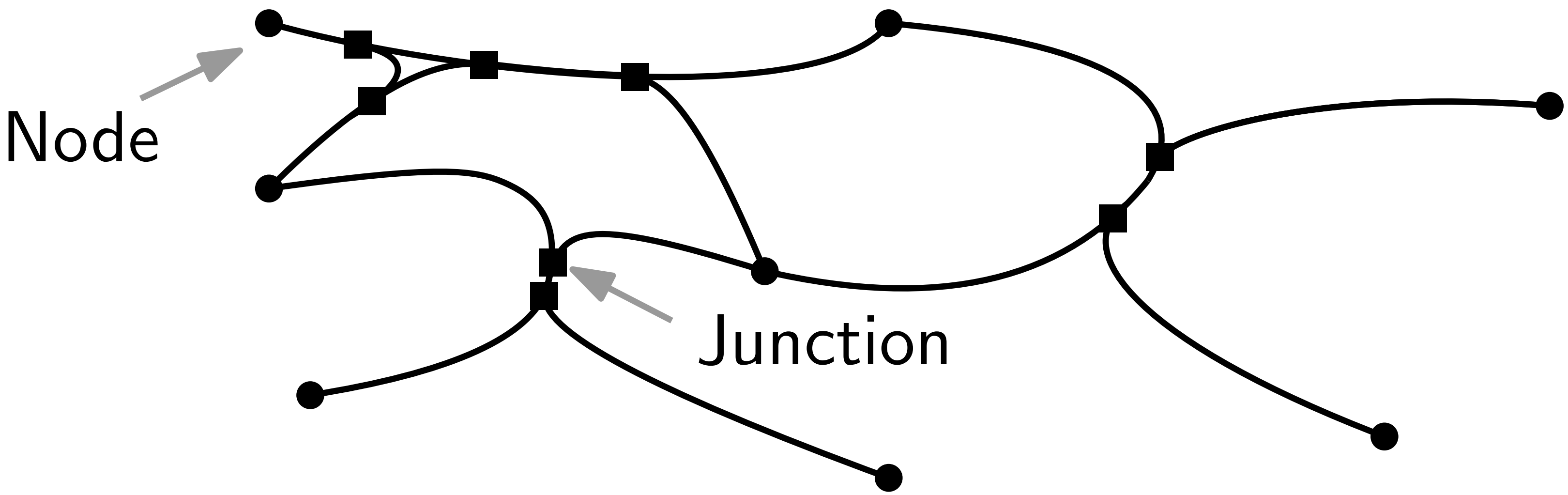
Definitions



Plane drawing, i.e. no “real” intersections
No wrong adjacencies

No double paths $\Rightarrow$ strict confluency
All vertices on a circle $\Rightarrow$ outer confluency

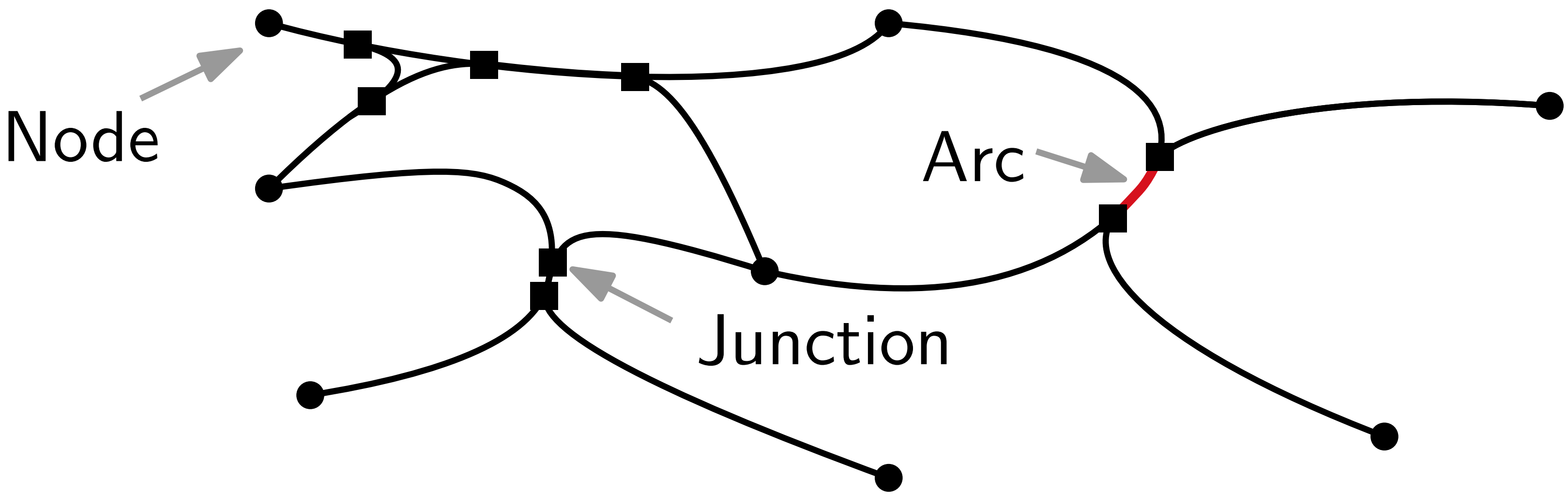
Definitions



Plane drawing, i.e. no "real" intersections
No wrong adjacencies

No double paths $\Rightarrow$ strict confluency
All vertices on a circle $\Rightarrow$ outer confluency

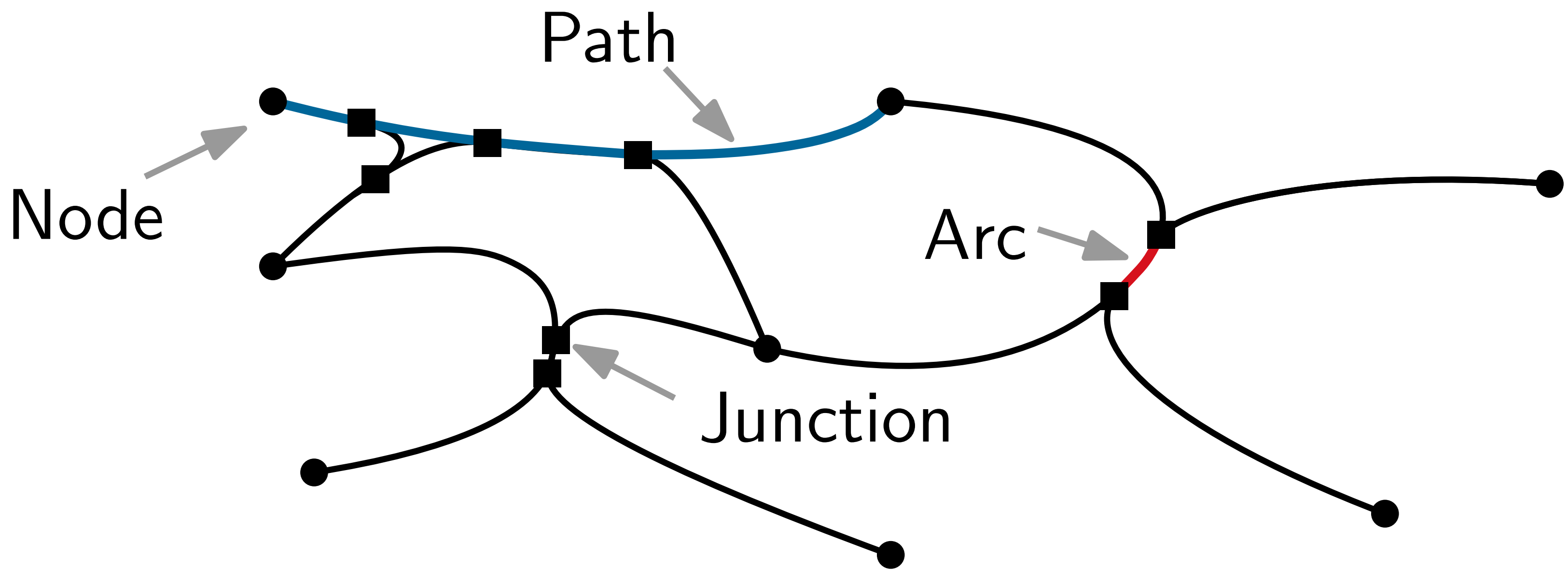
Definitions



Plane drawing, i.e. no "real" intersections
No wrong adjacencies

No double paths ⇒ strict confluency
All vertices on a circle ⇒ outer confluency

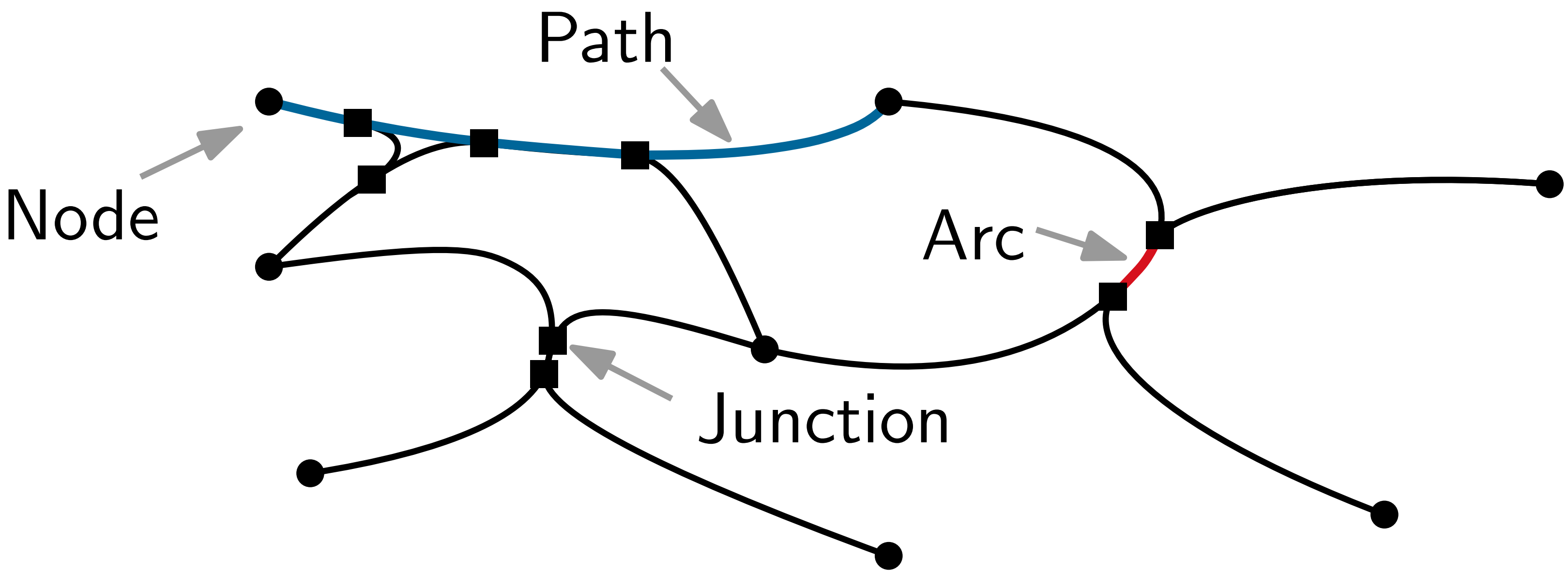
Definitions



Plane drawing, i.e. no “real” intersections
No wrong adjacencies

No double paths ⇒ strict confluency
All vertices on a circle ⇒ outer confluency

Definitions



Plane drawing, i.e. no “real” intersections
No wrong adjacencies

No double paths ⇒ strict confluency
All vertices on a circle ⇒ outer confluency

Every plane drawing is (strict) confluent
⇒ Questions mainly interesting for non-planar graphs

Known Results

Does a graph G admit a confluent drawing?

- Only known for a few classes of graphs
  E.g. Interval graphs, bipartite permutation graphs
  [Dickerson et al 2005, Hui et al. 2007]
- Negative case also only known for a few classes of graphs
  E.g. Petersen graph, Chordal graphs [Dickerson et al. 2005]
- Recognizing strict-confluent graphs is NP-hard
  [Eppstein et al. 2016]

Known Results

Does a graph G admit a confluent drawing?

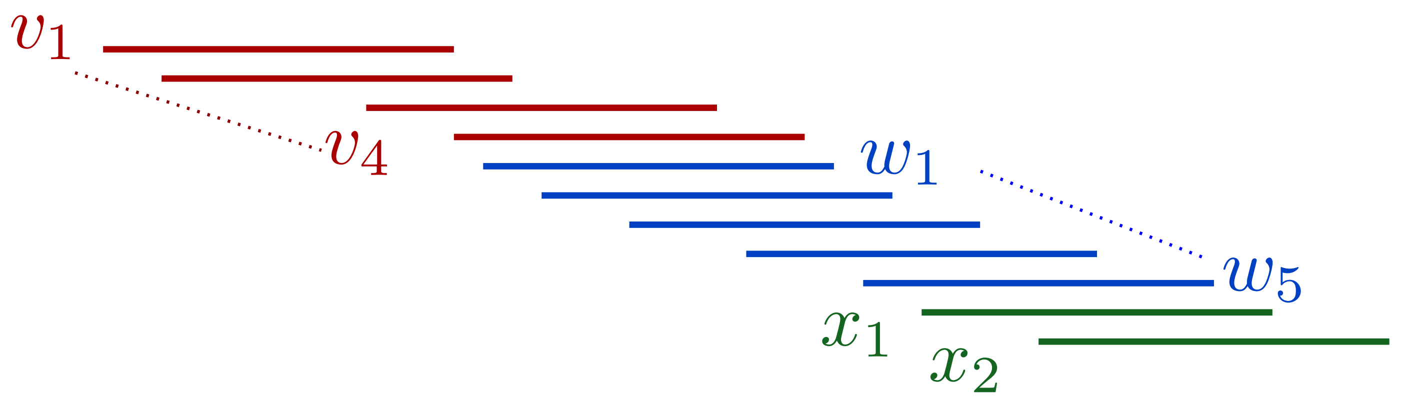
- Only known for a few classes of graphs
  E.g. Interval graphs, bipartite permutation graphs
  [Dickerson et al 2005, Hui et al. 2007]
- Negative case also only known for a few classes of graphs
  E.g. Petersen graph, Chordal graphs [Dickerson et al. 2005]
- Recognizing strict-confluent graphs is NP-hard
  [Eppstein et al. 2016]

One main open question

Which graphs have a strict outer-confluent drawing?

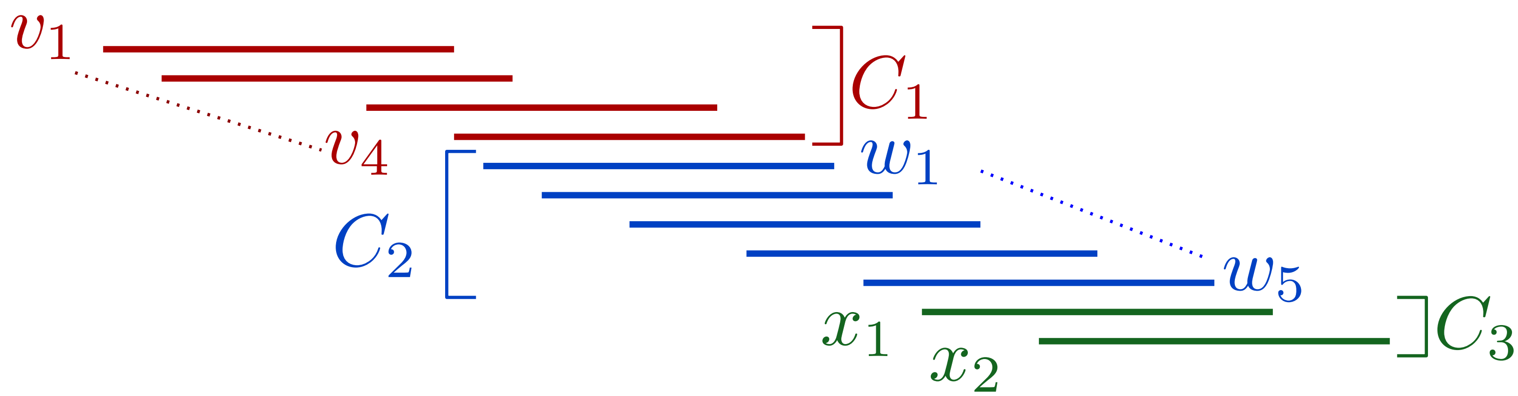
SC Drawings of Unit Interval Graphs

Graphs that can be represented as intersection of unit intervals



SC Drawings of Unit Interval Graphs

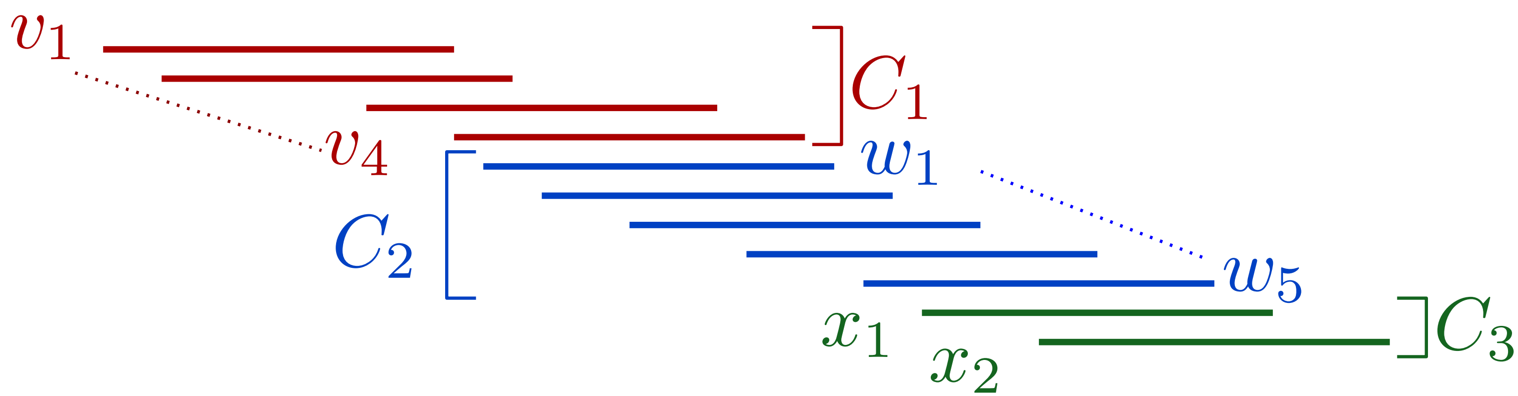
Graphs that can be represented as intersection of unit intervals



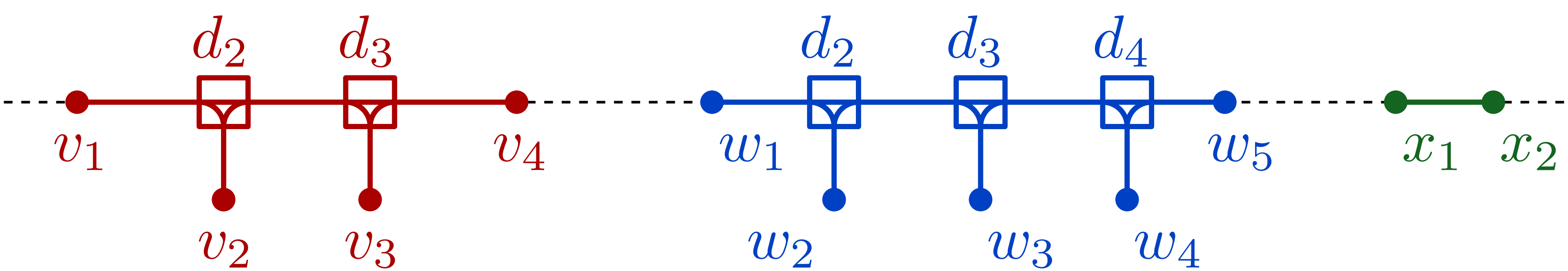
Decompose into cliques

SC Drawings of Unit Interval Graphs

Graphs that can be represented as intersection of unit intervals

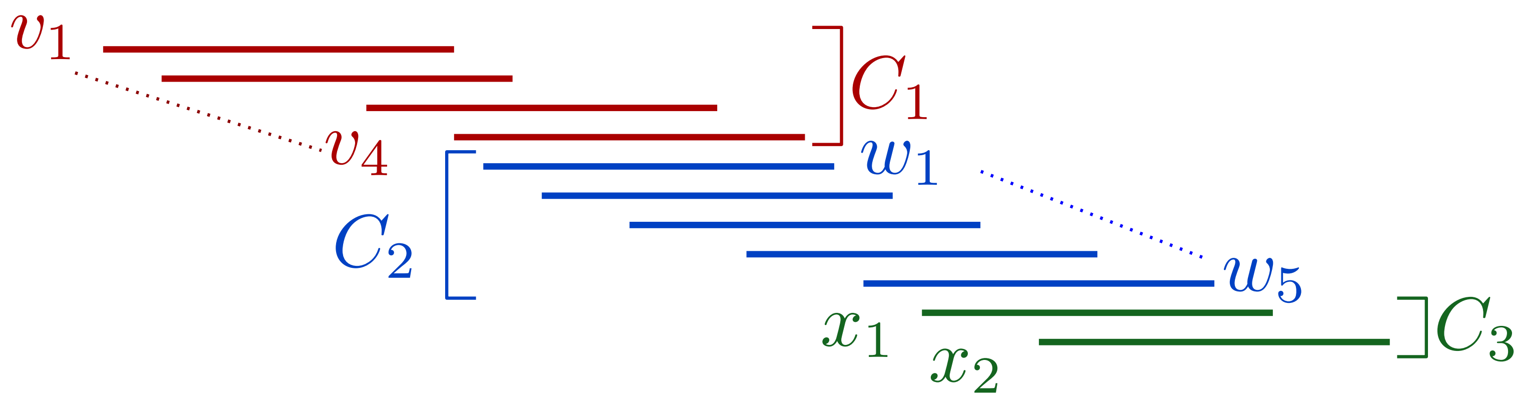


Decompose into cliques

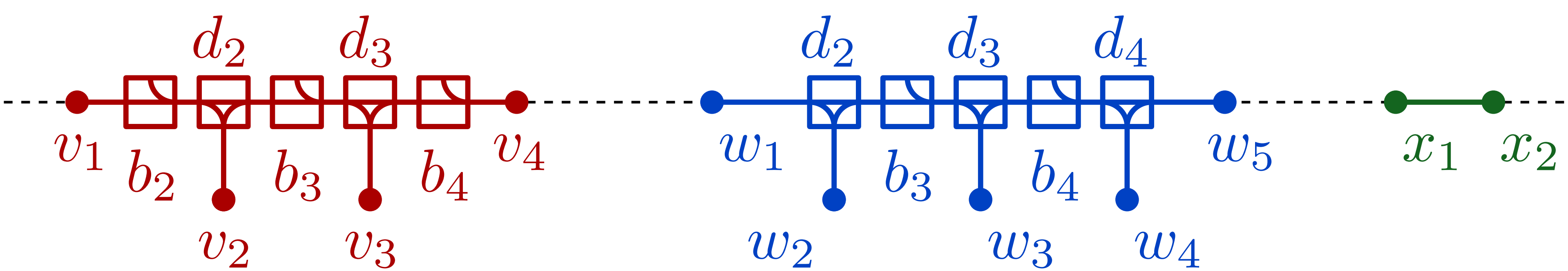


SC Drawings of Unit Interval Graphs

Graphs that can be represented as intersection of unit intervals

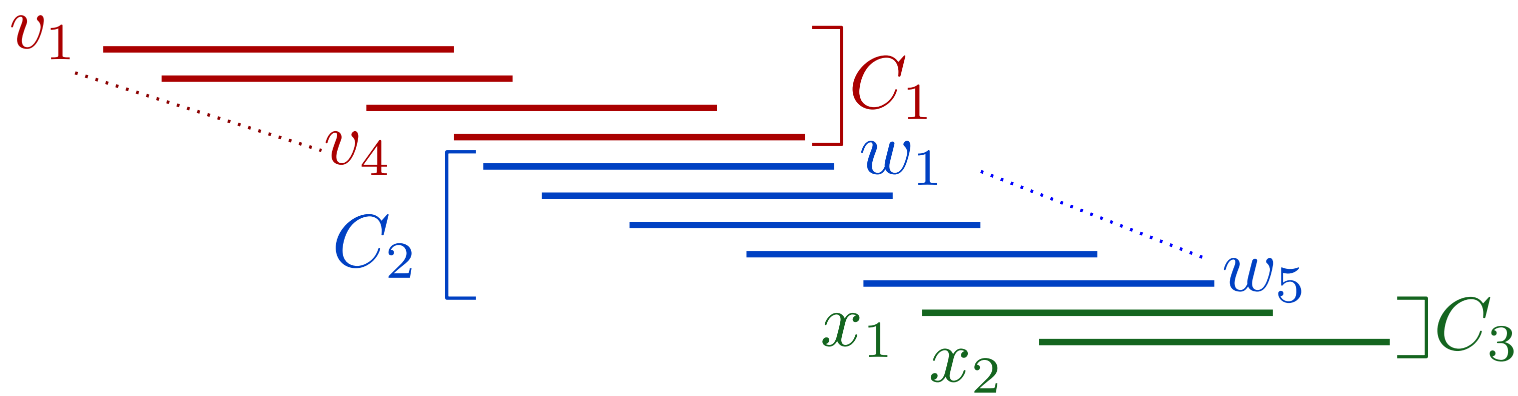


Decompose into cliques



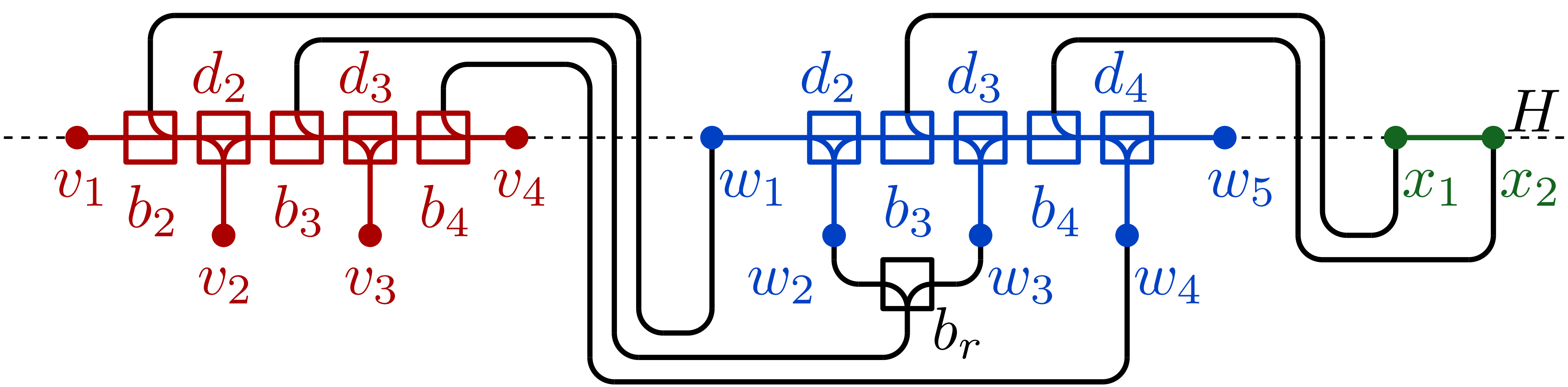
SC Drawings of Unit Interval Graphs

Graphs that can be represented as intersection of unit intervals

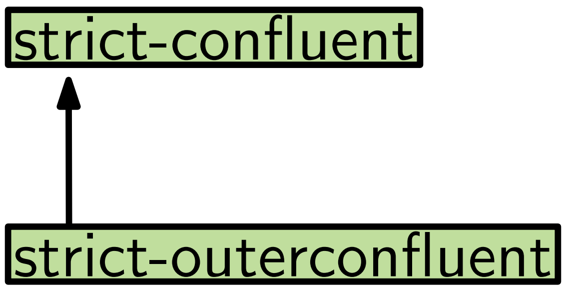


Decompose into cliques

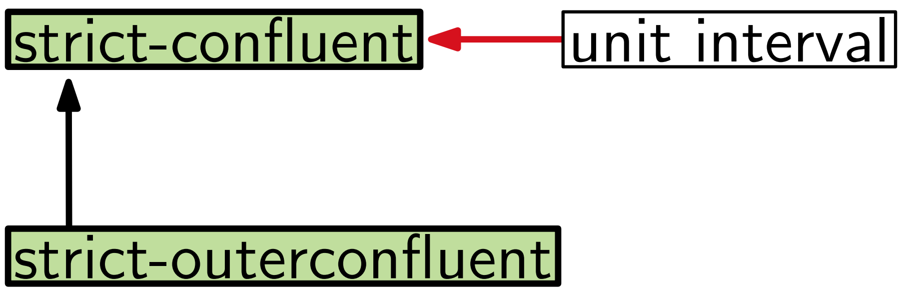
Connect the cliques with confluent paths



Our Results



Our Results



Our Results

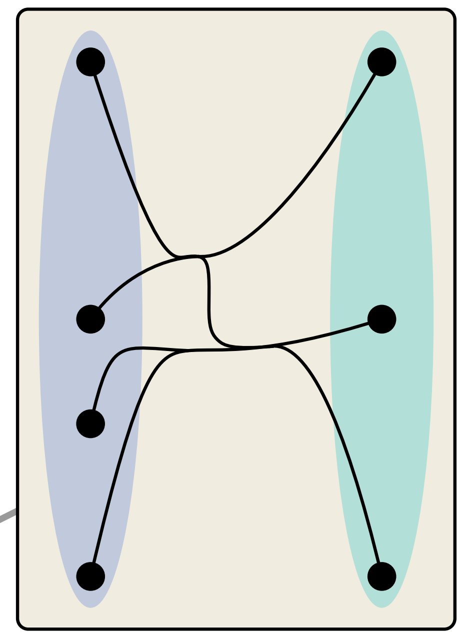
strict-confluent ← unit interval

↑

strict-outerconfluent

↑

strict bipartite outerconfluent



↕

bipartite permutation ∩ domino-free

Our Results

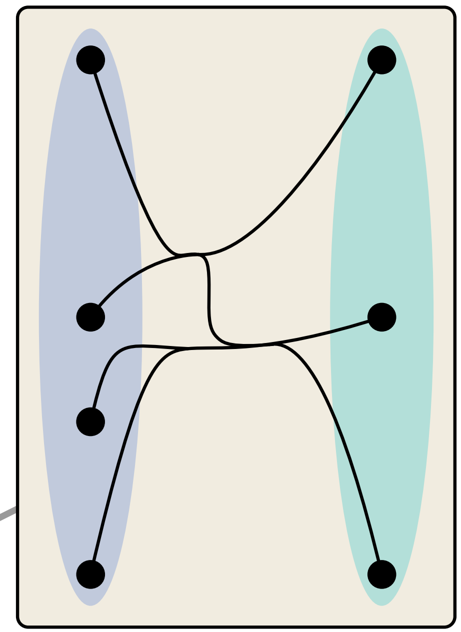
strict-confluent ← unit interval

↑

strict-outerconfluent

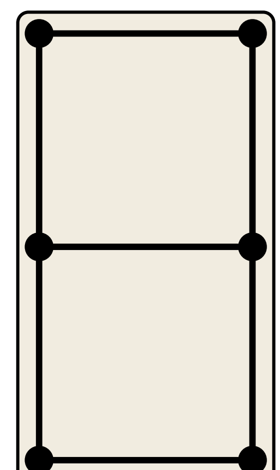
↑

strict bipartite outerconfluent

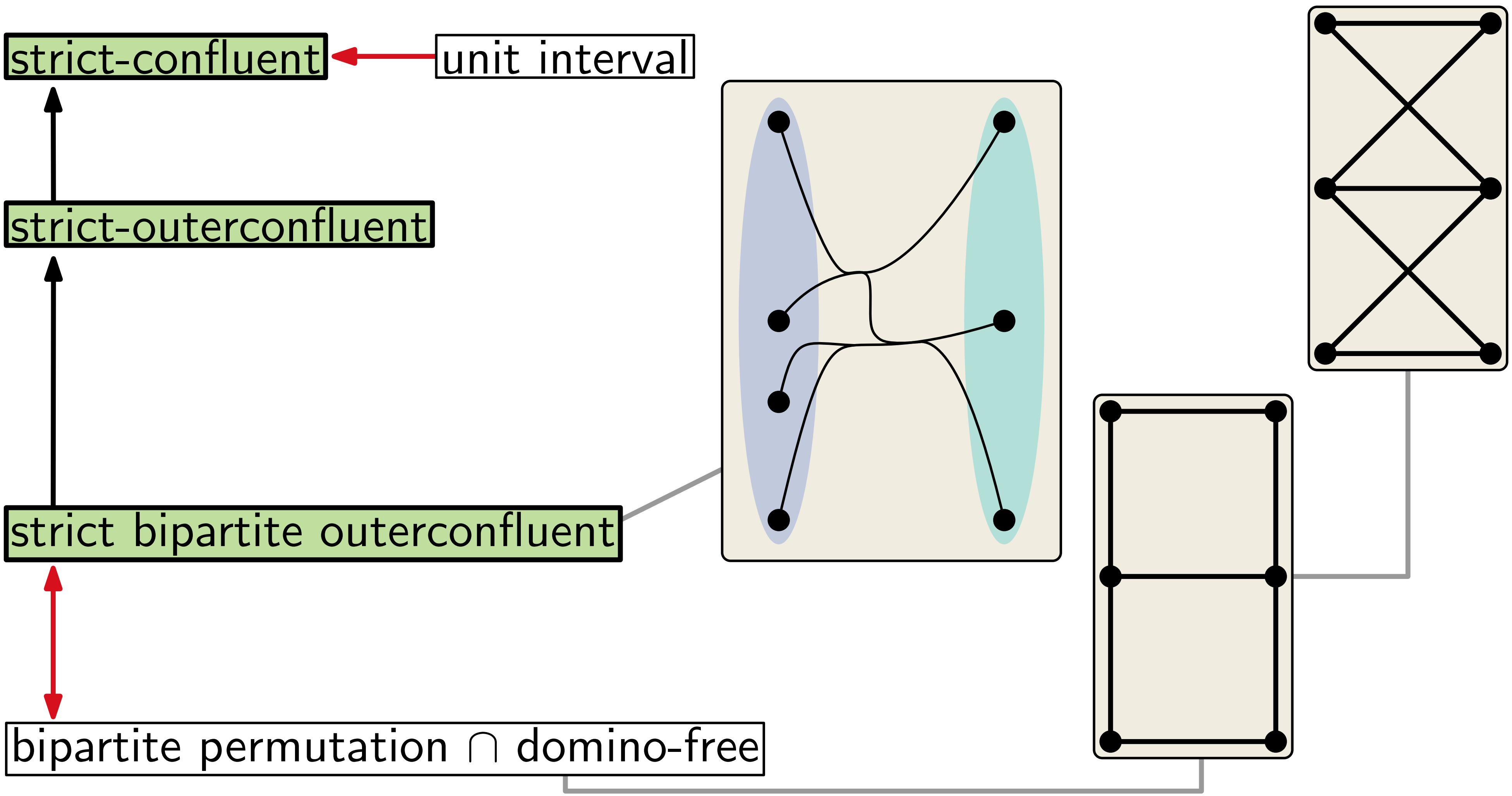


↕

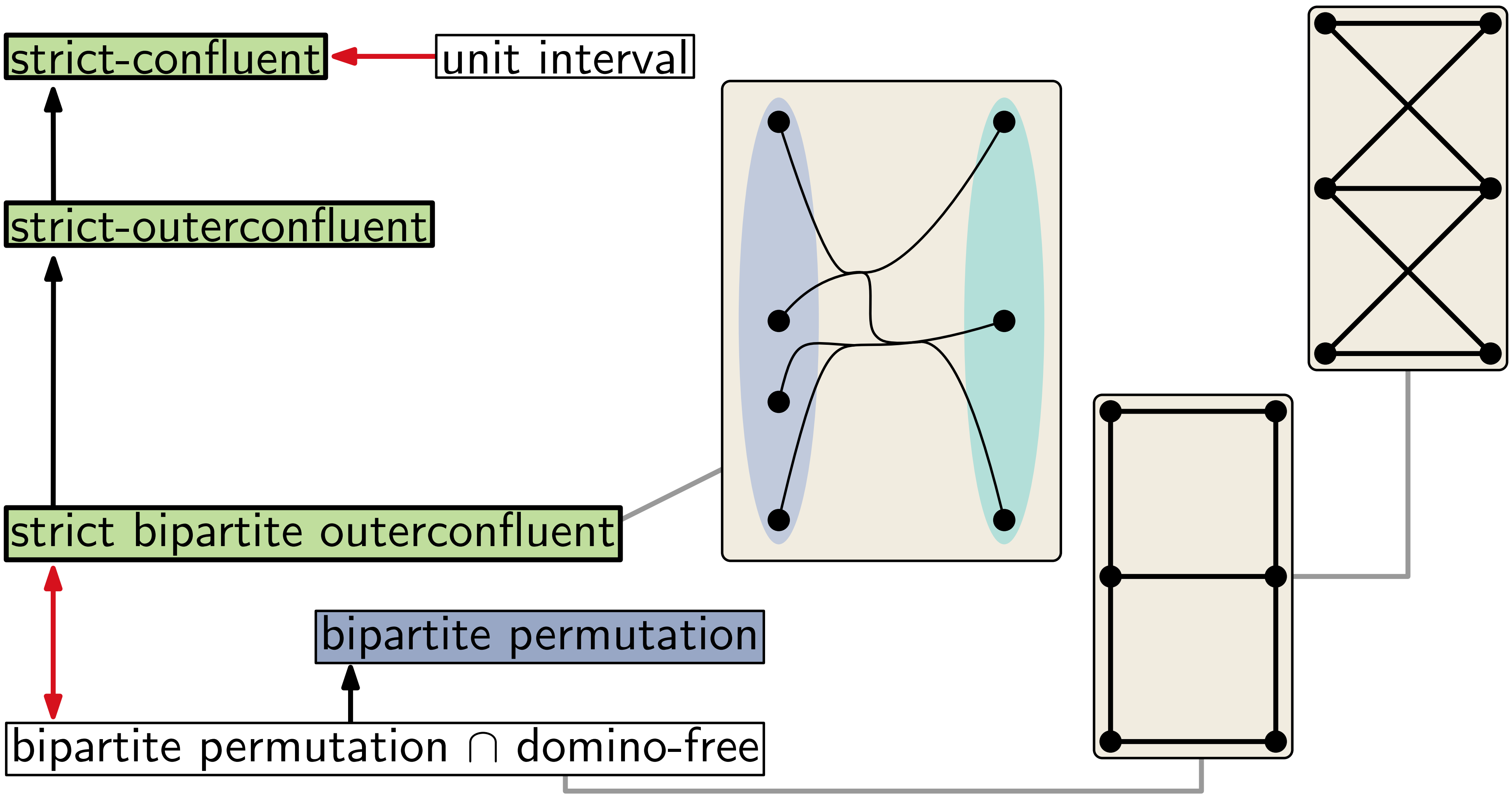
bipartite permutation ∩ domino-free



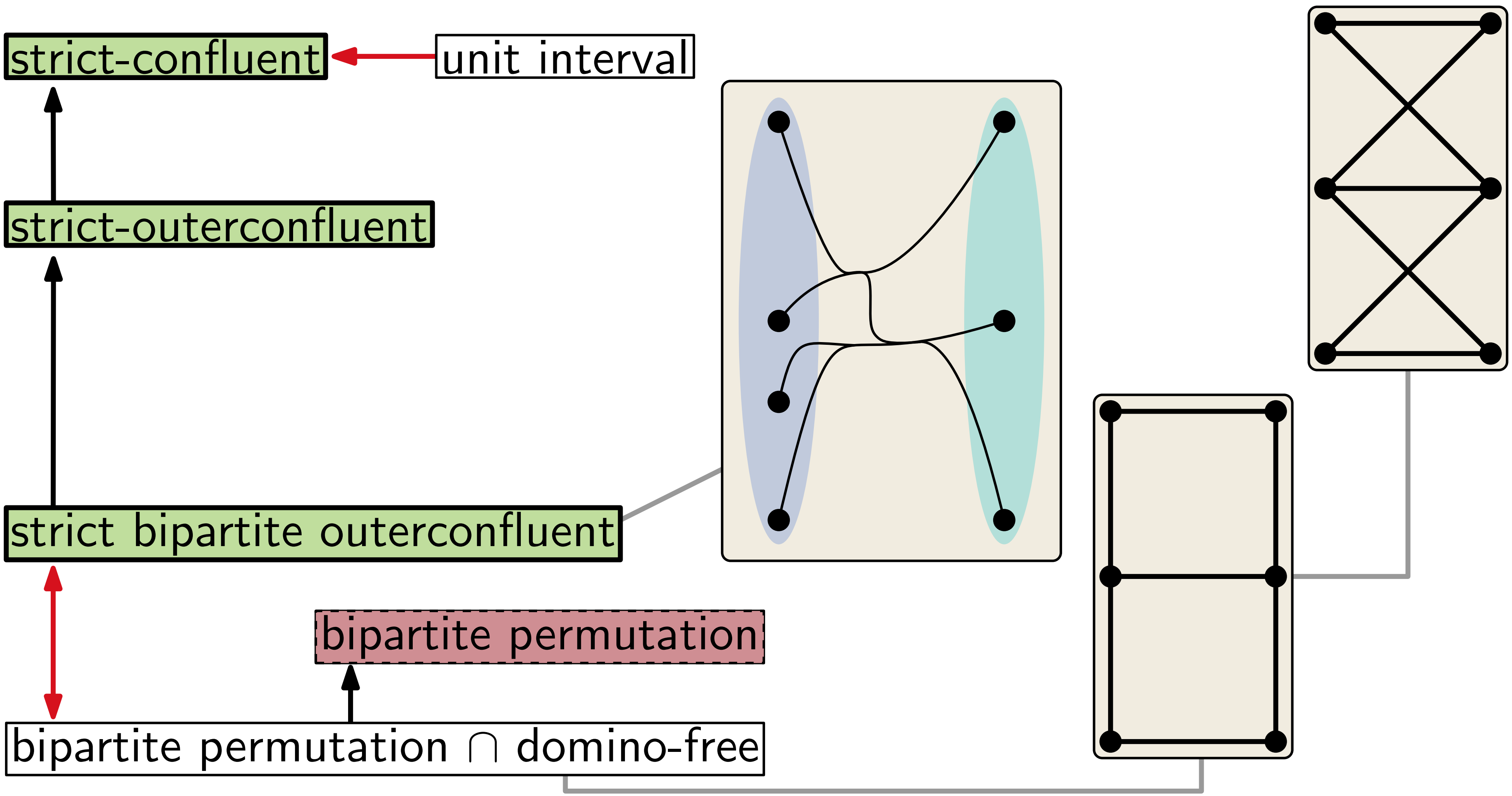
Our Results


strict-confluent
unit interval
strict-outerconfluent
strict bipartite outerconfluent
bipartite permutation ∩ domino-free

Our Results

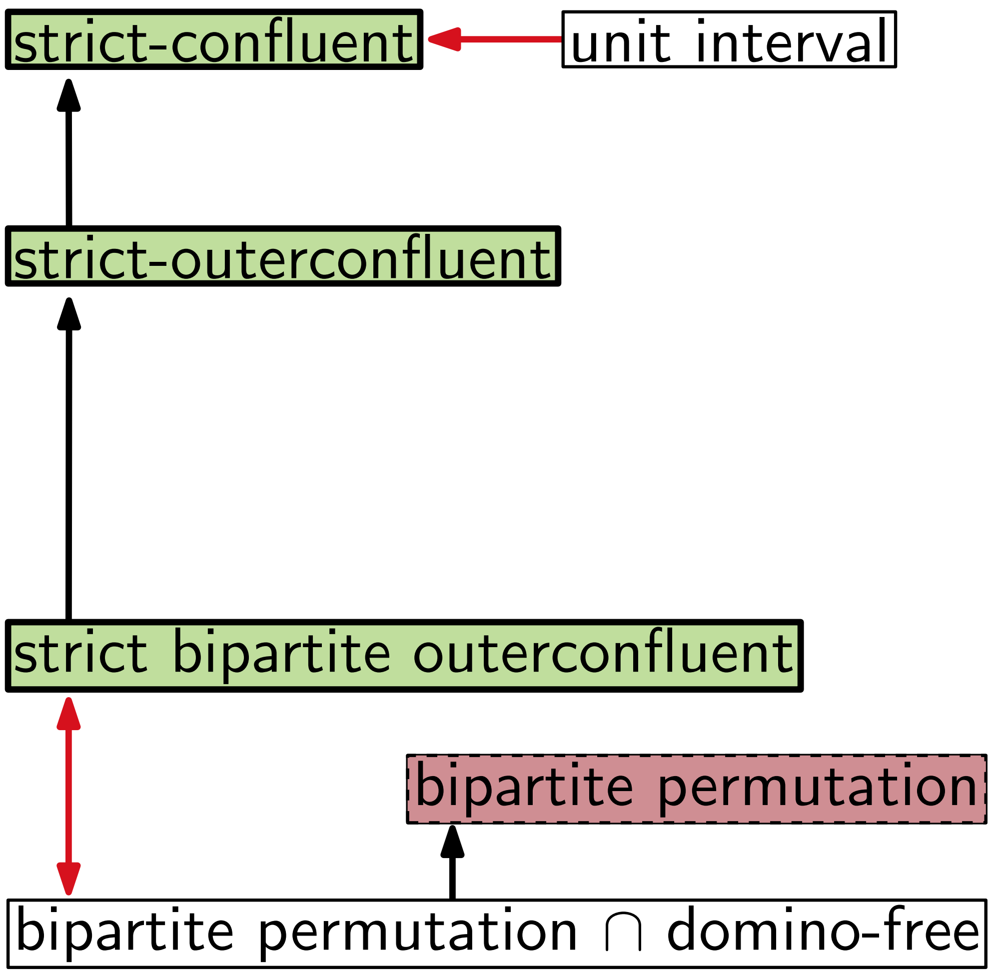

strict-confluent
unit interval
strict-outerconfluent
strict bipartite outerconfluent
bipartite permutation
bipartite permutation ∩ domino-free

Our Results



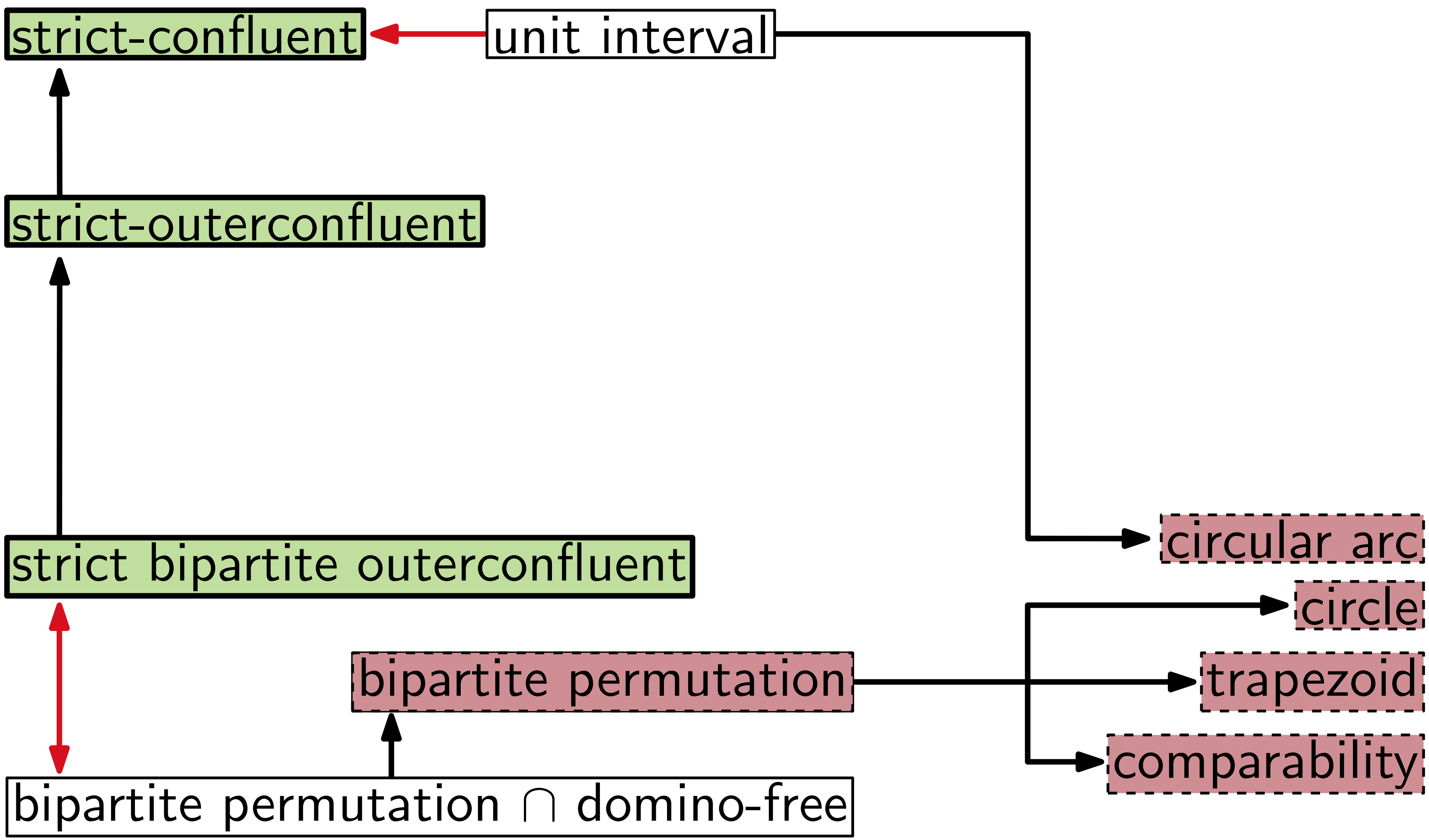
strict-confluent
unit interval
strict-outerconfluent
strict bipartite outerconfluent
bipartite permutation
bipartite permutation ∩ domino-free

Our Results

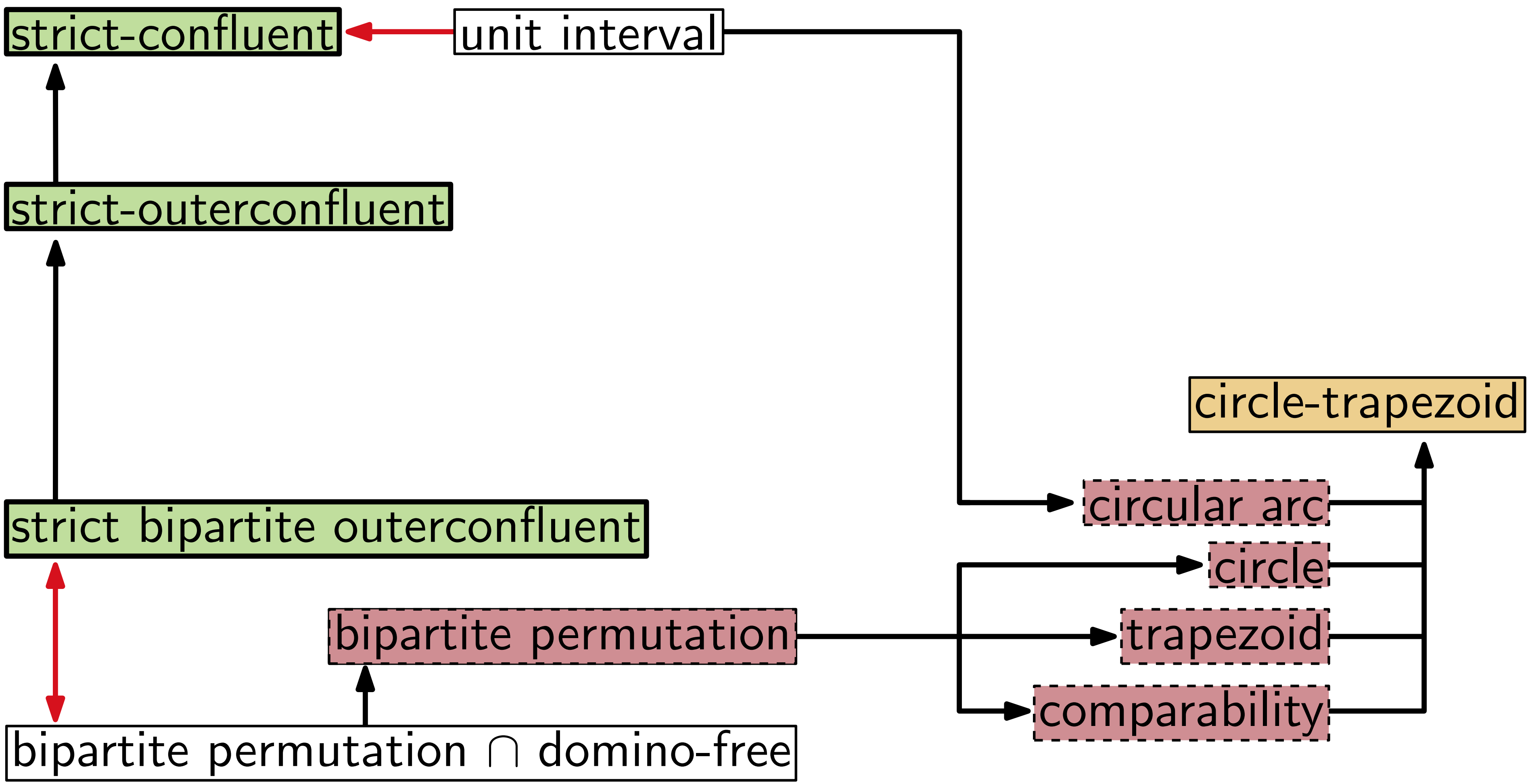


strict-confluent
unit interval
strict-outerconfluent
strict bipartite outerconfluent
bipartite permutation
bipartite permutation ∩ domino-free

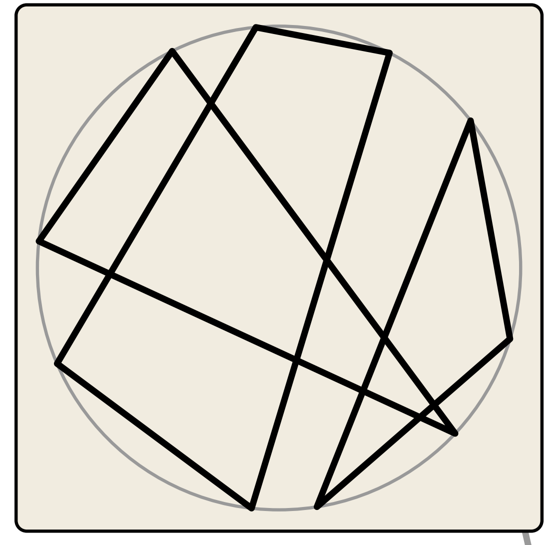
Our Results

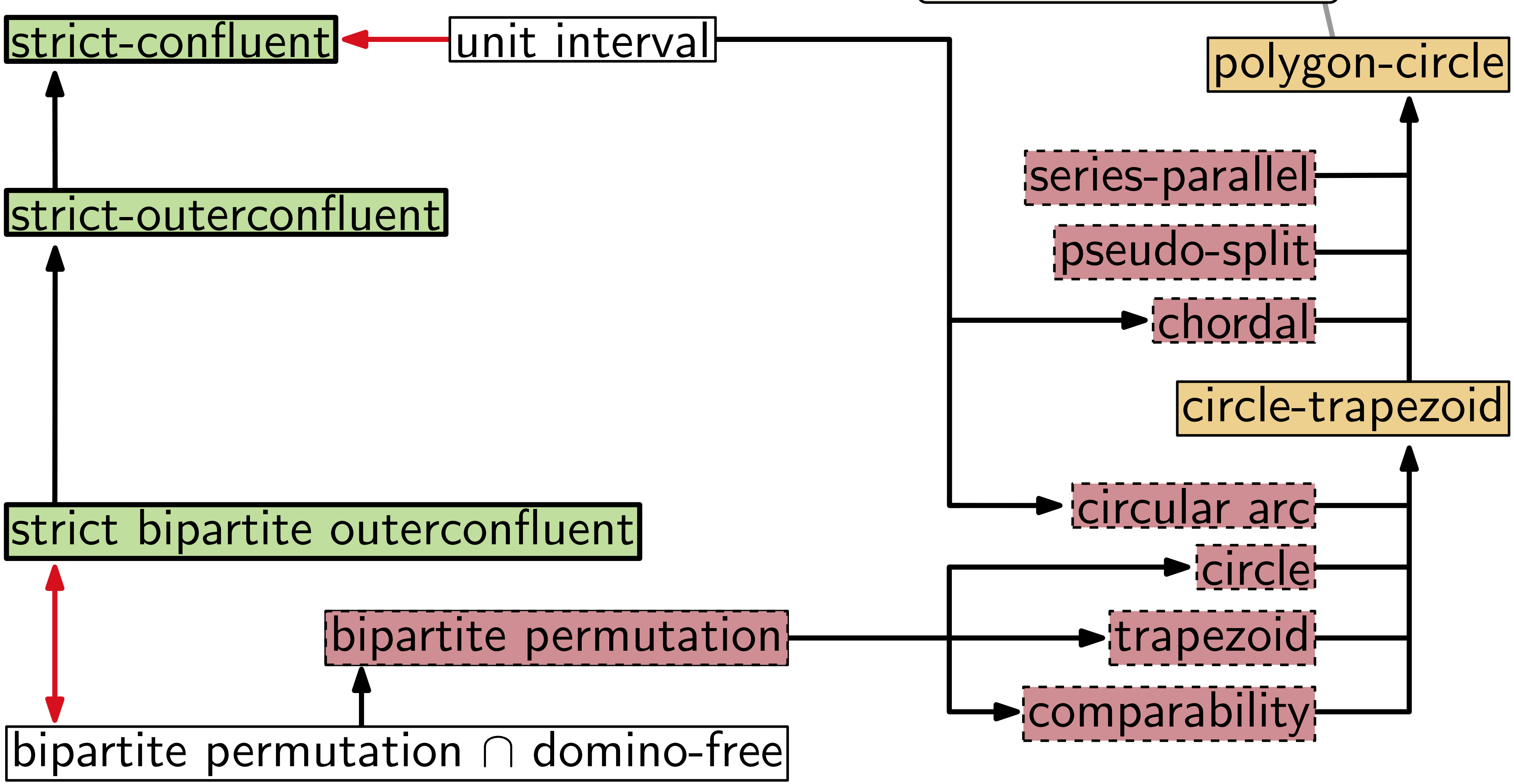

strict-confluent
unit interval
strict-outerconfluent
strict bipartite outerconfluent
circular arc
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

Our Results

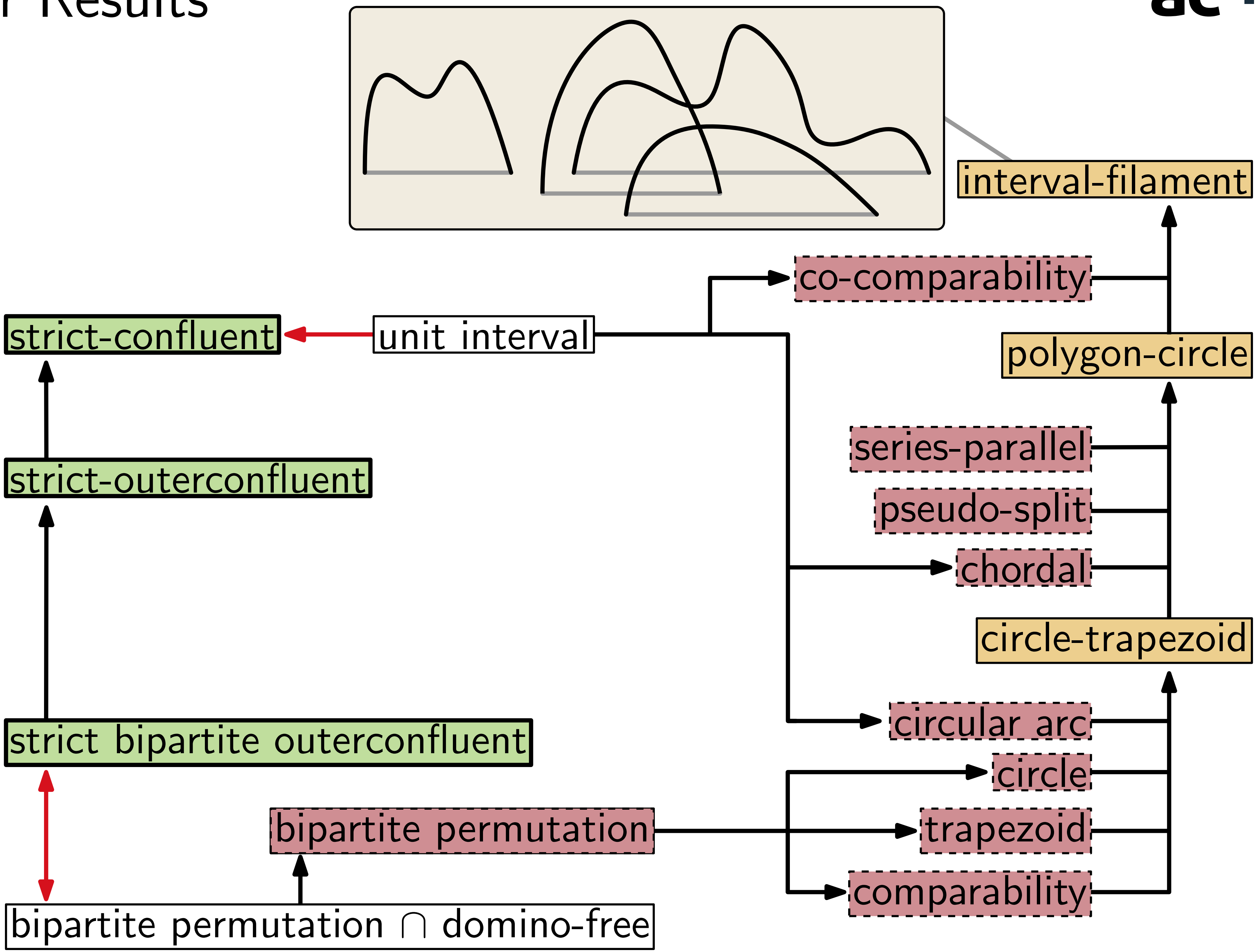

strict-confluent
unit interval
strict-outerconfluent
circle-trapezoid
circular arc
circle
strict bipartite outerconfluent
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

Our Results

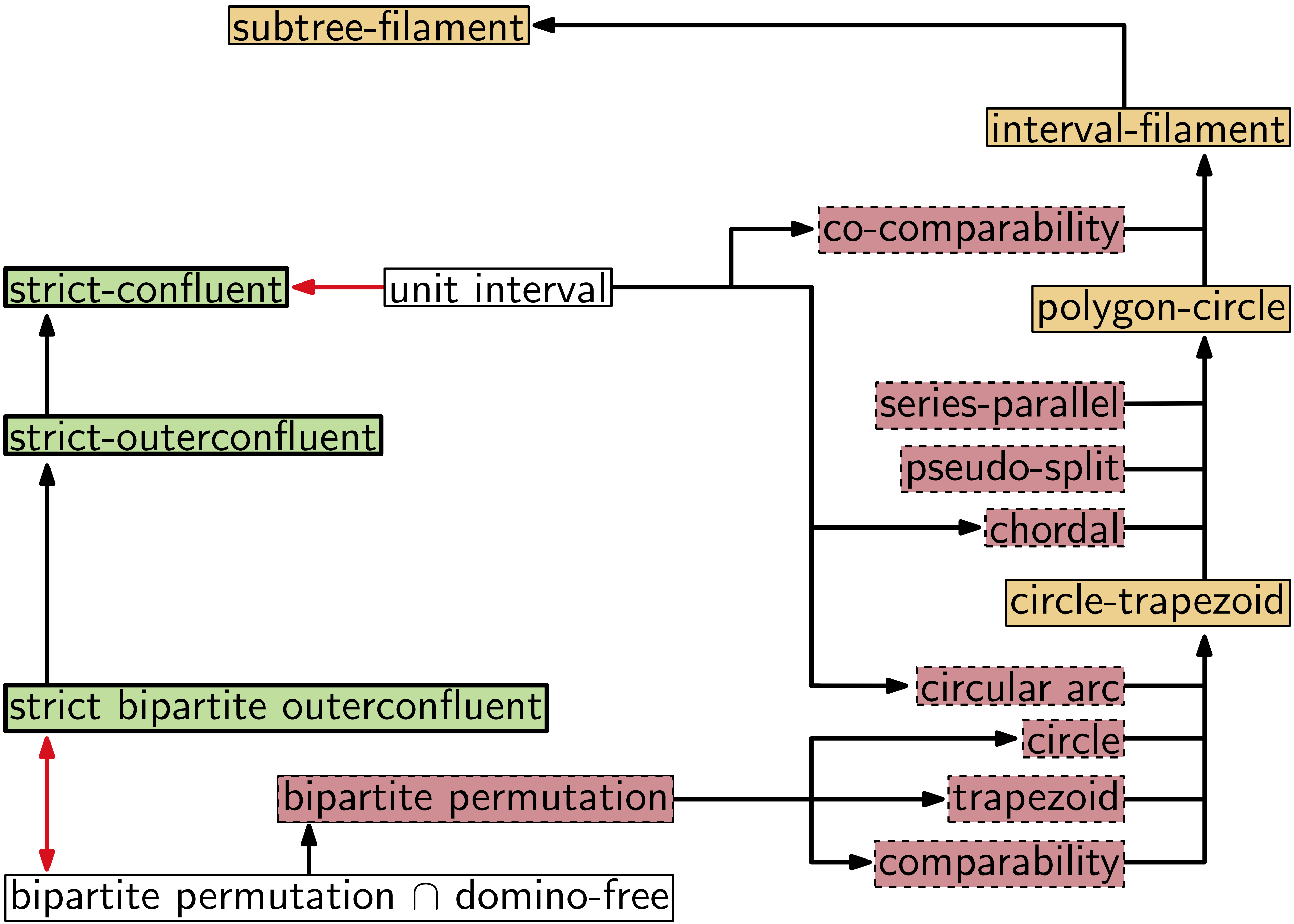



strict-confluent
unit interval
polygon-circle
series-parallel
strict-outerconfluent
pseudo-split
chordal
circle-trapezoid
circular arc
strict bipartite outerconfluent
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

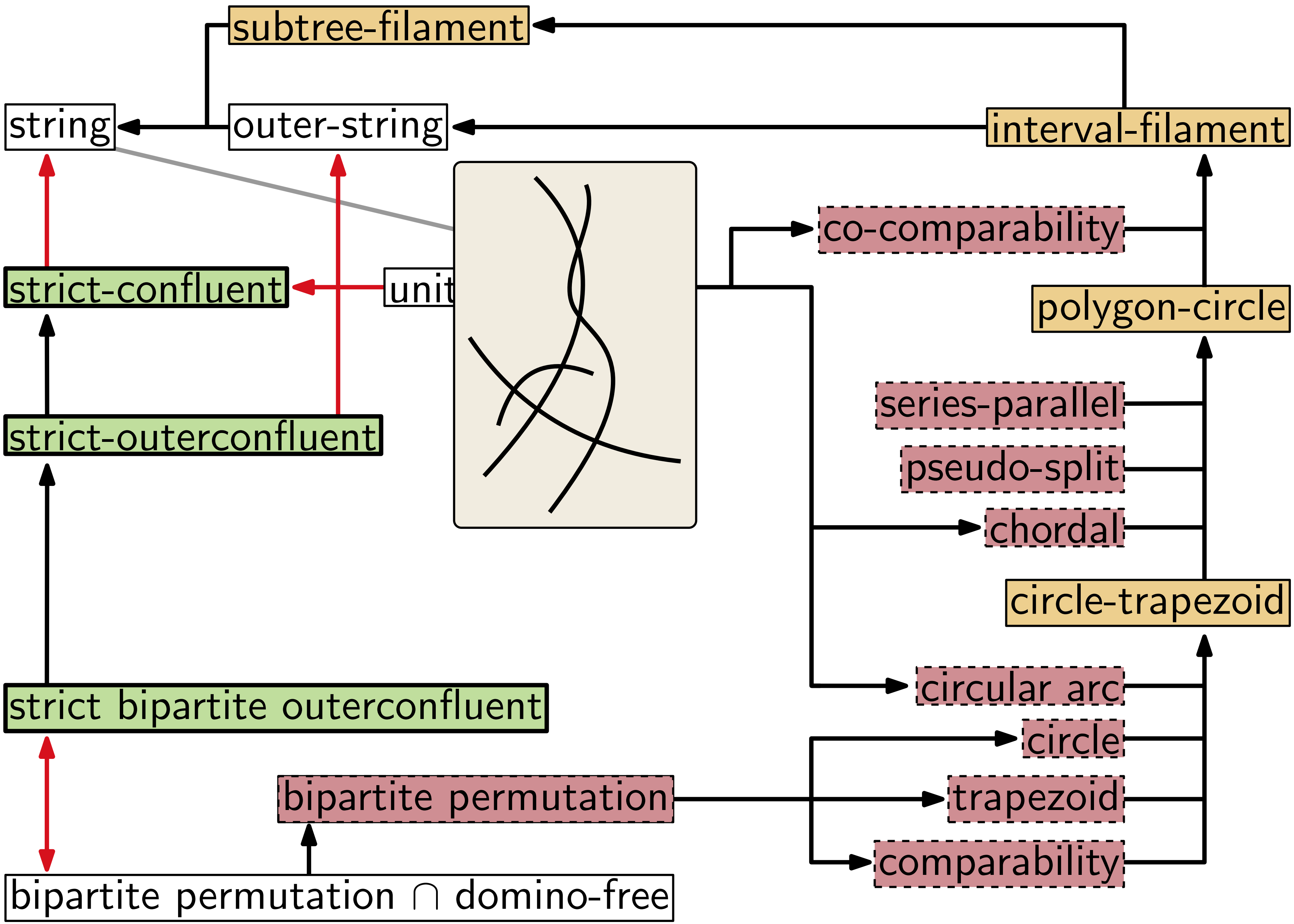
Our Results


interval-filament
co-comparability
unit interval
strict-confluent
polygon-circle
series-parallel
strict-outerconfluent
pseudo-split
chordal
circle-trapezoid
circular arc
strict bipartite outerconfluent
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

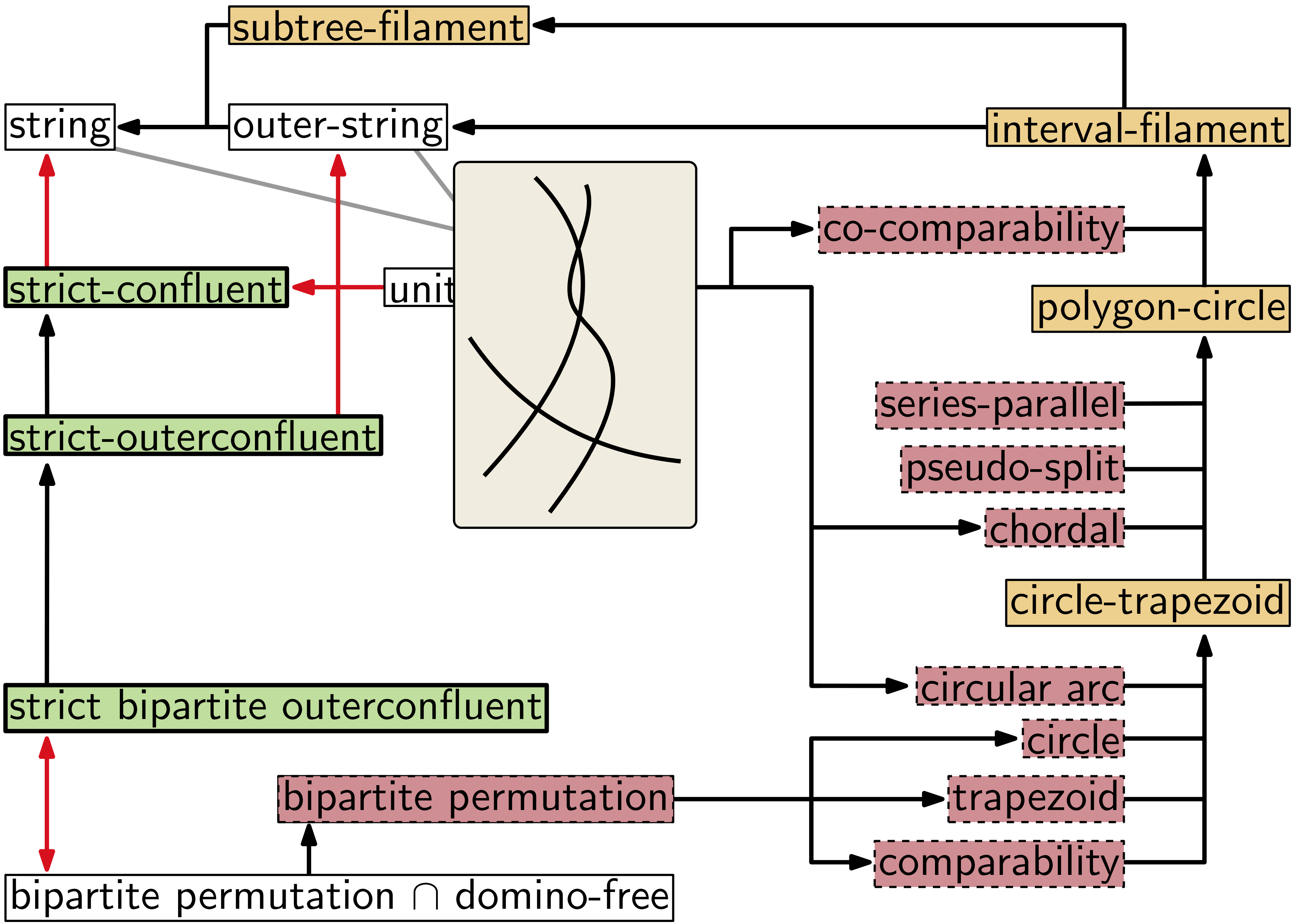
Our Results


subtree-filament
interval-filament
co-comparability
strict-confluent
unit interval
polygon-circle
strict-outerconfluent
series-parallel
pseudo-split
chordal
circle-trapezoid
circular arc
strict bipartite outerconfluent
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

Our Results

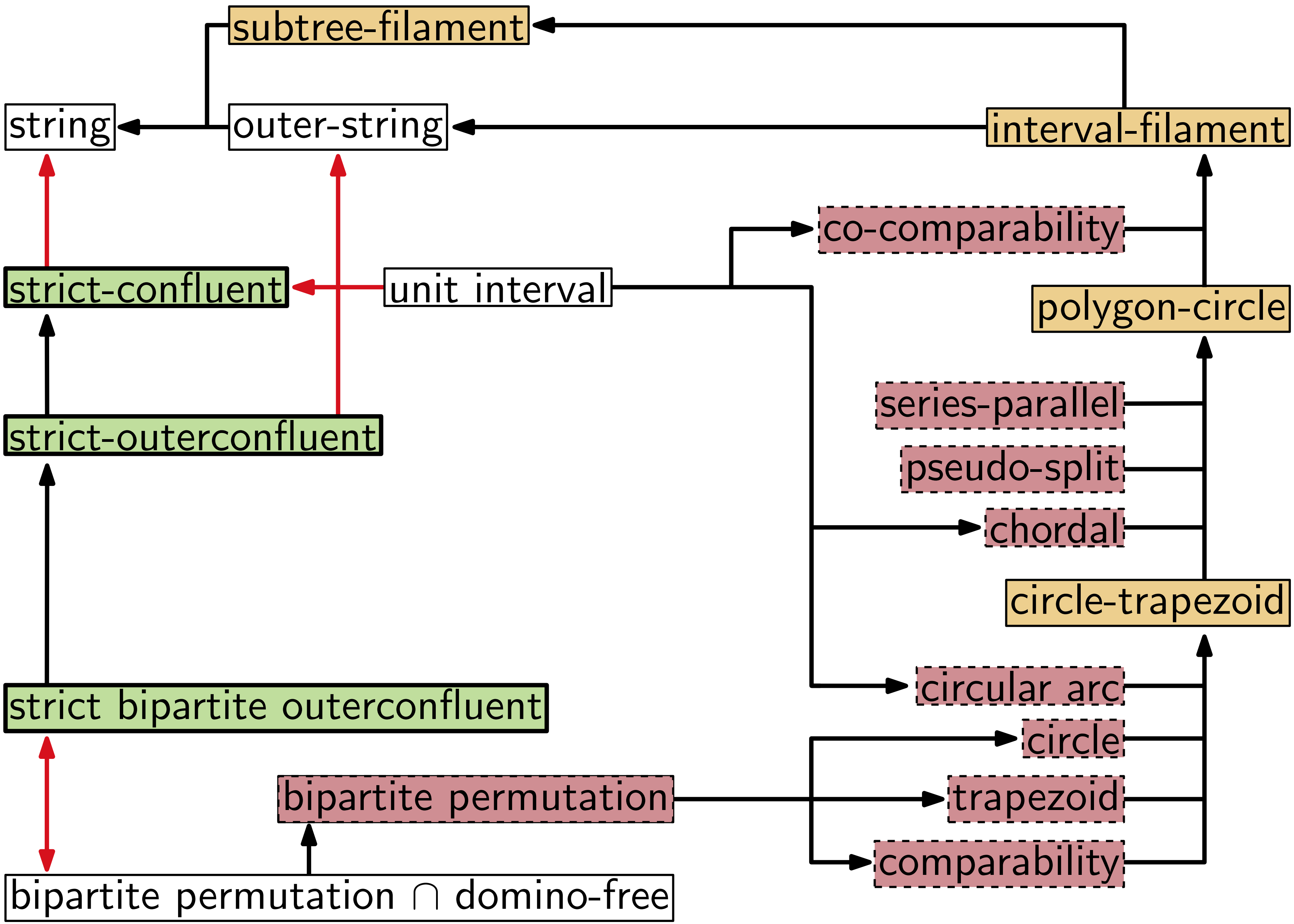

subtree-filament
string
outer-string
interval-filament
co-comparability
strict-confluent
unit
polygon-circle
strict-outerconfluent
series-parallel
pseudo-split
chordal
circle-trapezoid
strict bipartite outerconfluent
circular arc
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

Our Results

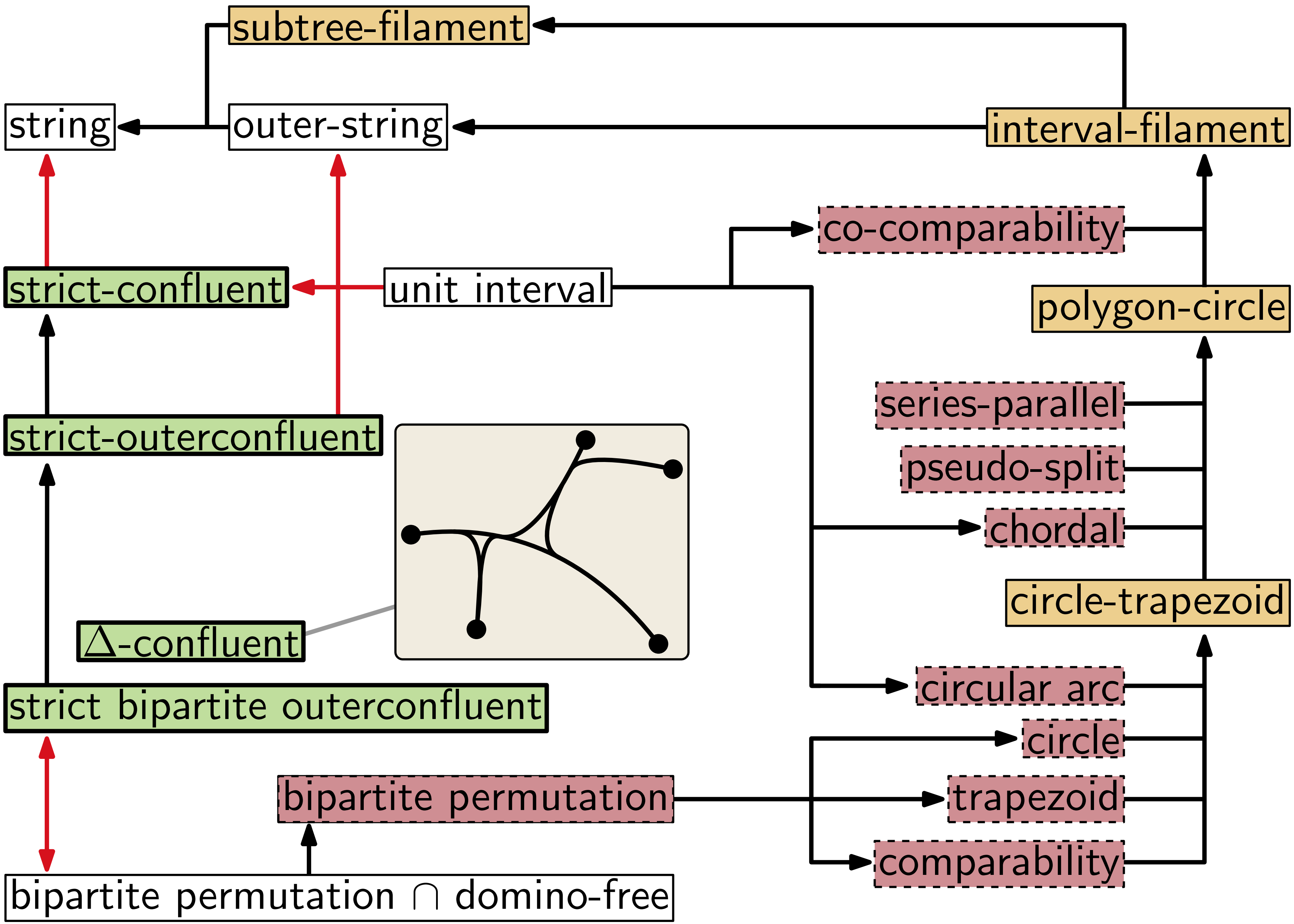


subtree-filament
string
outer-string
interval-filament
co-comparability
strict-confluent
unit
polygon-circle
strict-outerconfluent
series-parallel
pseudo-split
chordal
circle-trapezoid
strict bipartite outerconfluent
circular arc
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

Our Results

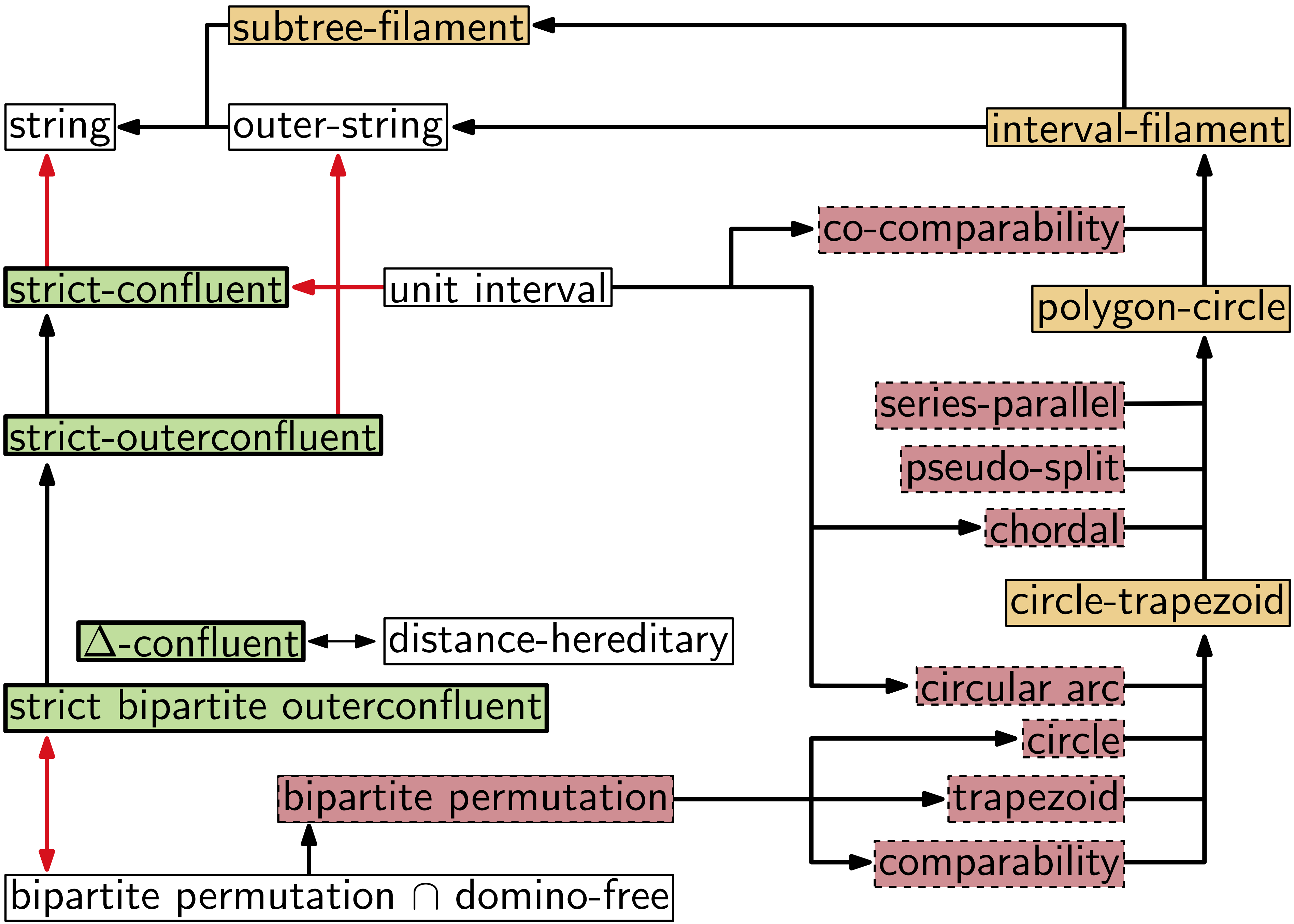

subtree-filament
string
outer-string
interval-filament
co-comparability
strict-confluent
unit interval
polygon-circle
series-parallel
strict-outerconfluent
pseudo-split
chordal
circle-trapezoid
strict bipartite outerconfluent
circular arc
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

Our Results

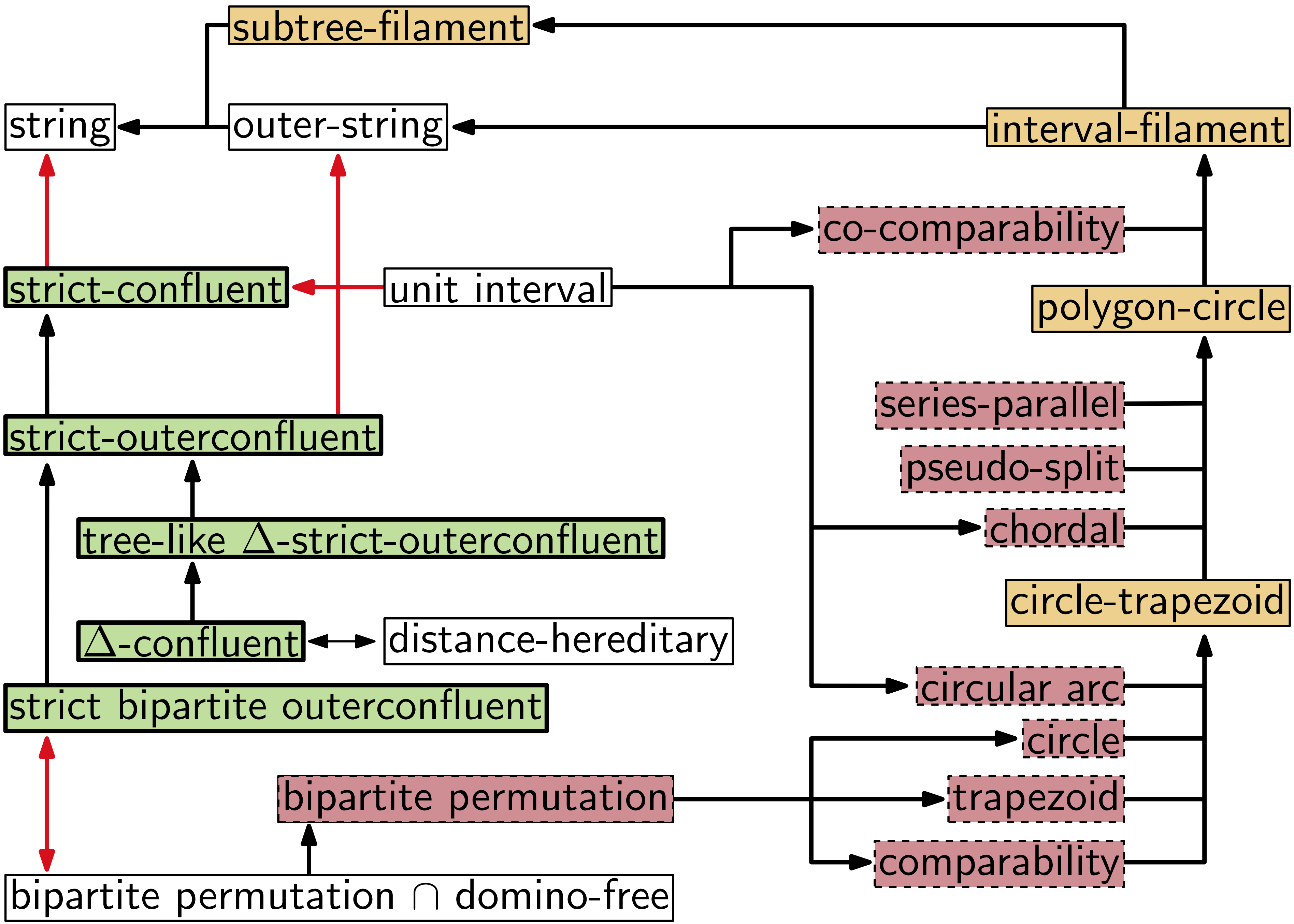


subtree-filament
string
outer-string
interval-filament
co-comparability
strict-confluent
unit interval
polygon-circle
series-parallel
strict-outerconfluent
pseudo-split
chordal
circle-trapezoid
Δ-confluent
circular arc
strict bipartite outerconfluent
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

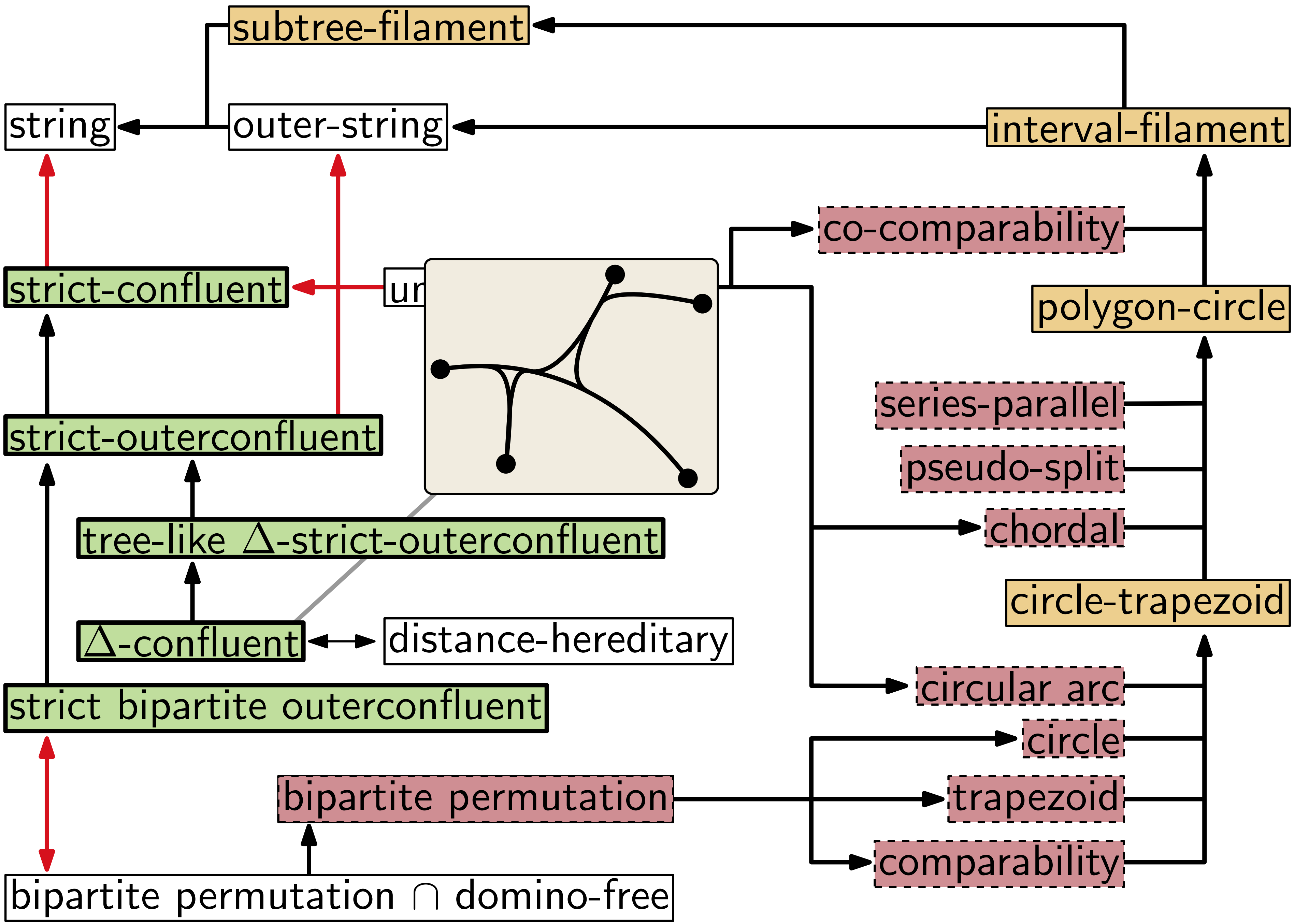
Our Results


subtree-filament
string
outer-string
interval-filament
co-comparability
strict-confluent
unit interval
polygon-circle
series-parallel
strict-outerconfluent
pseudo-split
chordal
circle-trapezoid
Δ-confluent
distance-hereditary
circular arc
strict bipartite outerconfluent
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

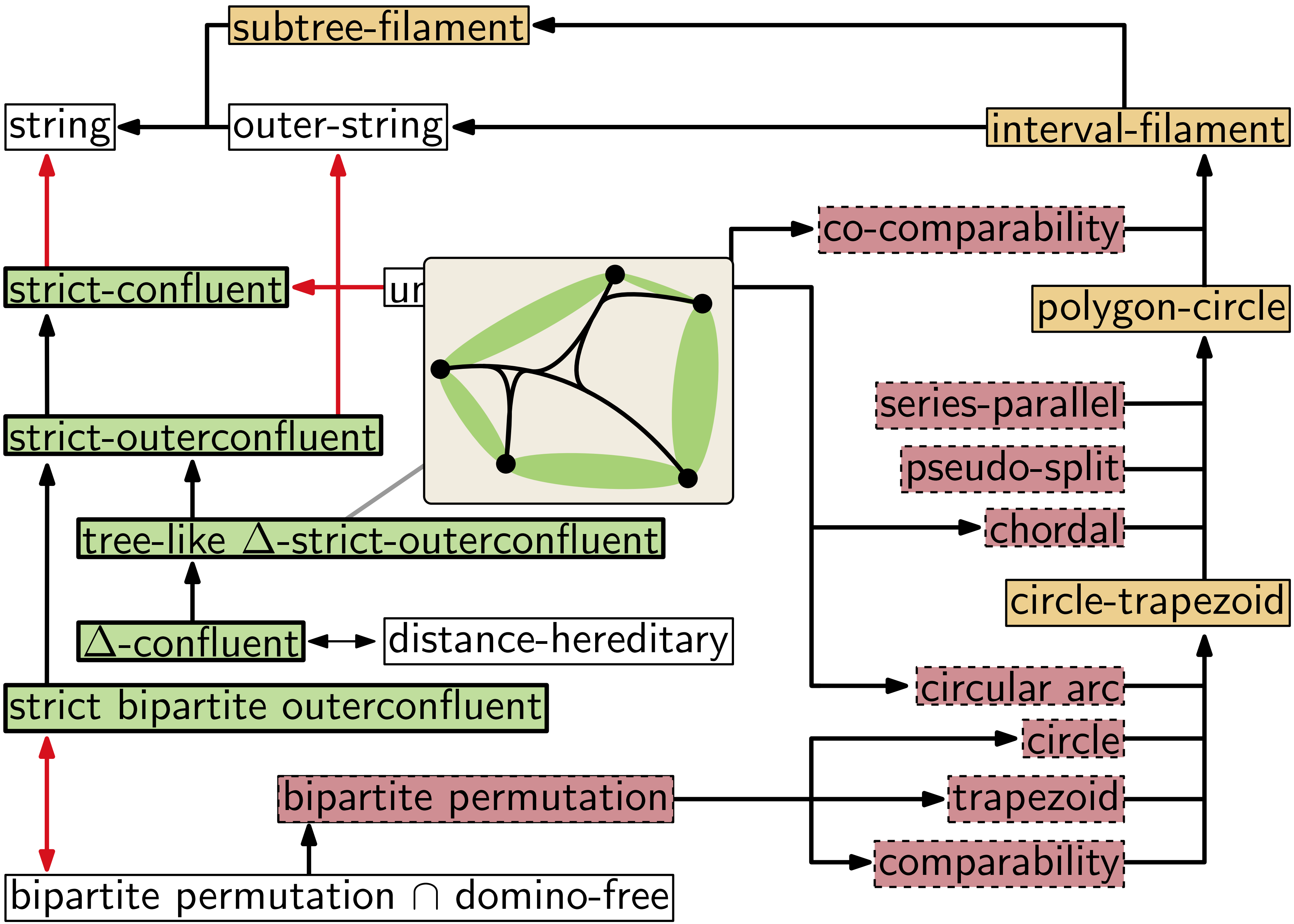
Our Results


subtree-filament
string
outer-string
interval-filament
co-comparability
strict-confluent
unit interval
polygon-circle
series-parallel
strict-outerconfluent
pseudo-split
tree-like Δ-strict-outerconfluent
chordal
circle-trapezoid
Δ-confluent
distance-hereditary
strict bipartite outerconfluent
circular arc
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

Our Results

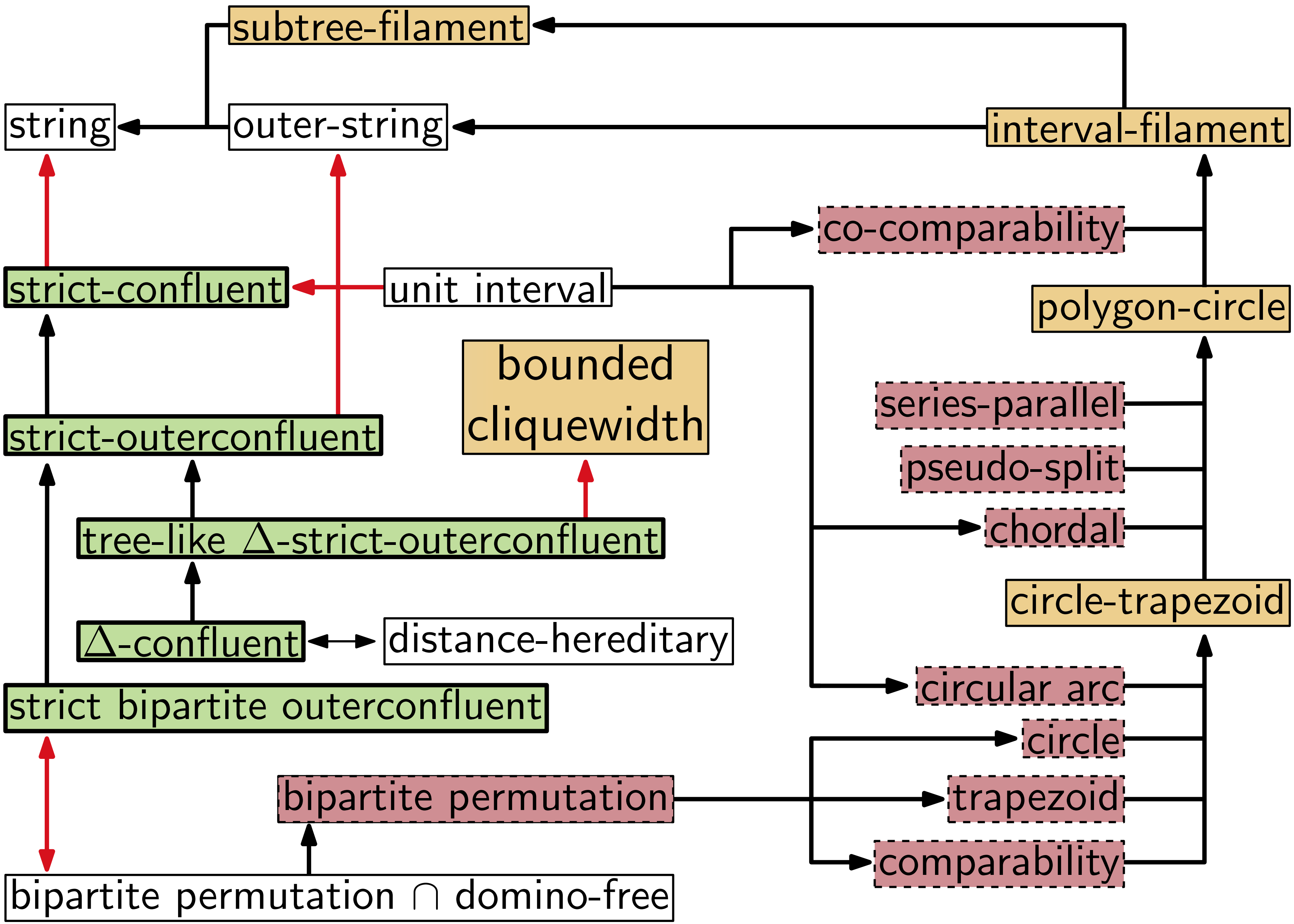

subtree-filament
string
outer-string
interval-filament
co-comparability
strict-confluent
ur
polygon-circle
strict-outerconfluent
series-parallel
pseudo-split
chordal
tree-like Δ-strict-outerconfluent
circle-trapezoid
Δ-confluent
distance-hereditary
strict bipartite outerconfluent
circular arc
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

Our Results

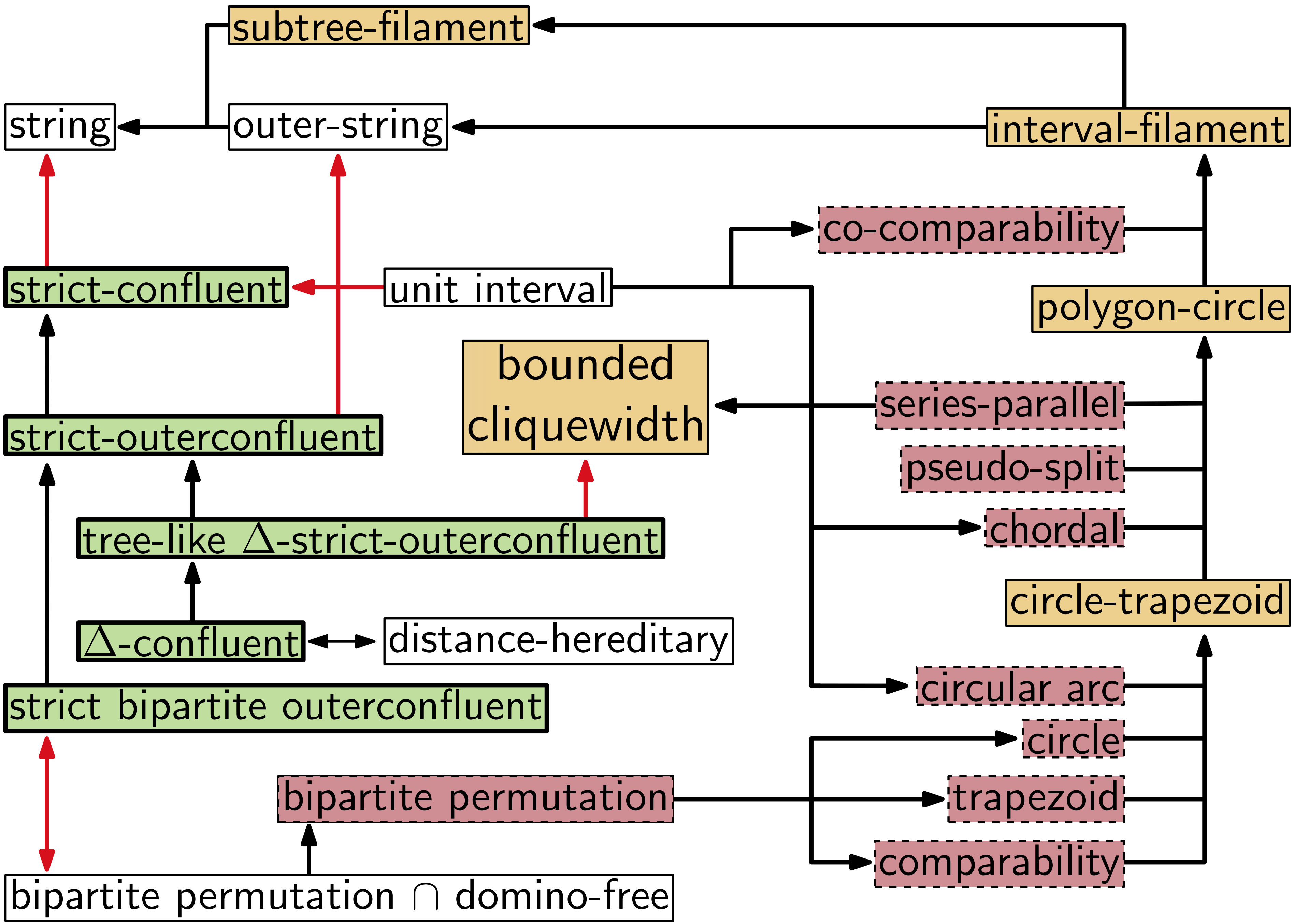


subtree-filament
string
outer-string
interval-filament
co-comparability
strict-confluent
polygon-circle
series-parallel
pseudo-split
chordal
strict-outerconfluent
tree-like Δ-strict-outerconfluent
Δ-confluent
distance-hereditary
circle-trapezoid
strict bipartite outerconfluent
circular arc
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

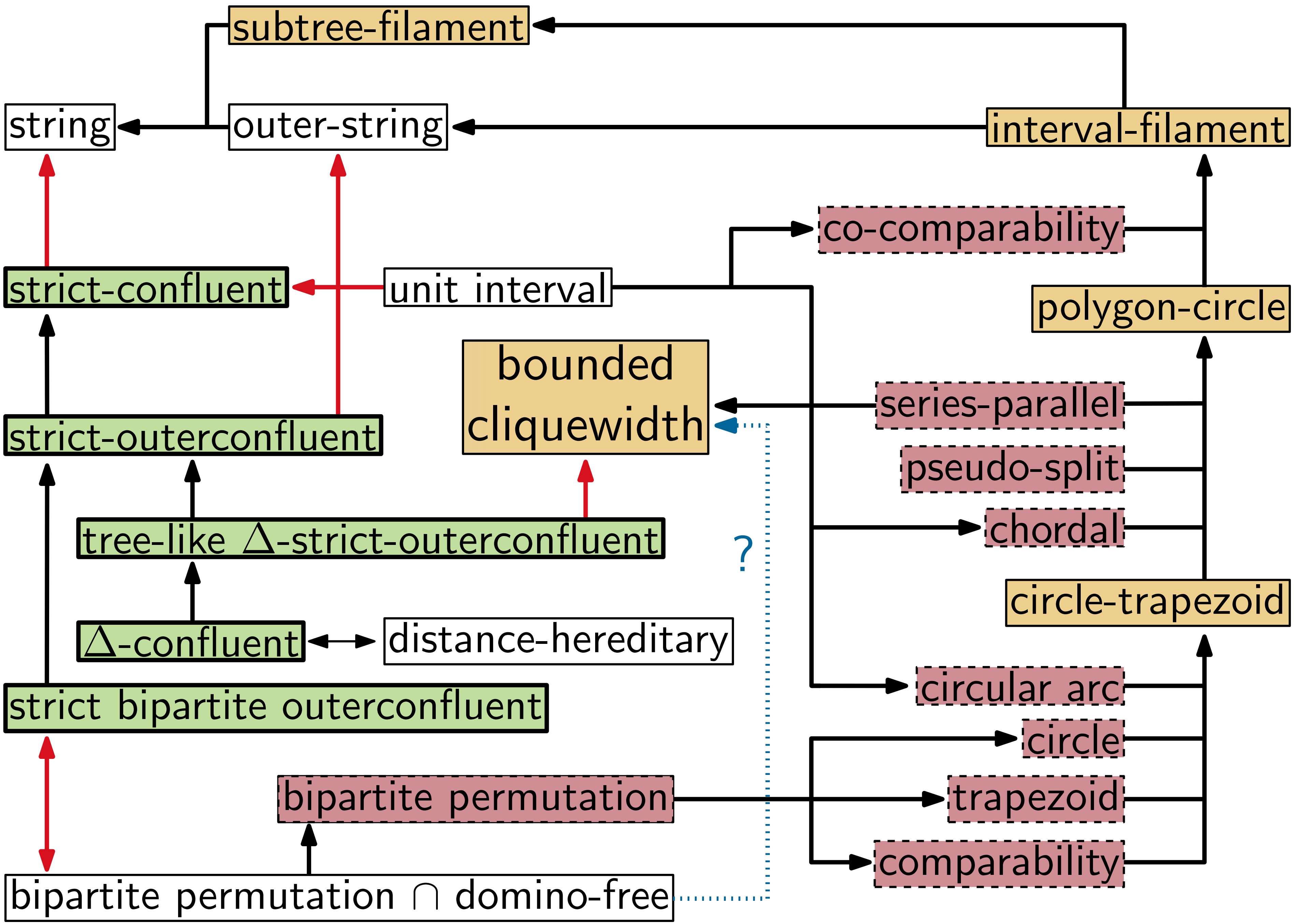
Our Results


subtree-filament
string
outer-string
interval-filament
co-comparability
strict-confluent
unit interval
polygon-circle
bounded cliquewidth
series-parallel
strict-outerconfluent
pseudo-split
tree-like Δ-strict-outerconfluent
chordal
circle-trapezoid
Δ-confluent
distance-hereditary
strict bipartite outerconfluent
circular arc
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

Our Results


subtree-filament
string
outer-string
interval-filament
co-comparability
strict-confluent
unit interval
polygon-circle
bounded cliquewidth
series-parallel
strict-outerconfluent
pseudo-split
chordal
tree-like Δ-strict-outerconfluent
circle-trapezoid
Δ-confluent
distance-hereditary
circular arc
strict bipartite outerconfluent
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

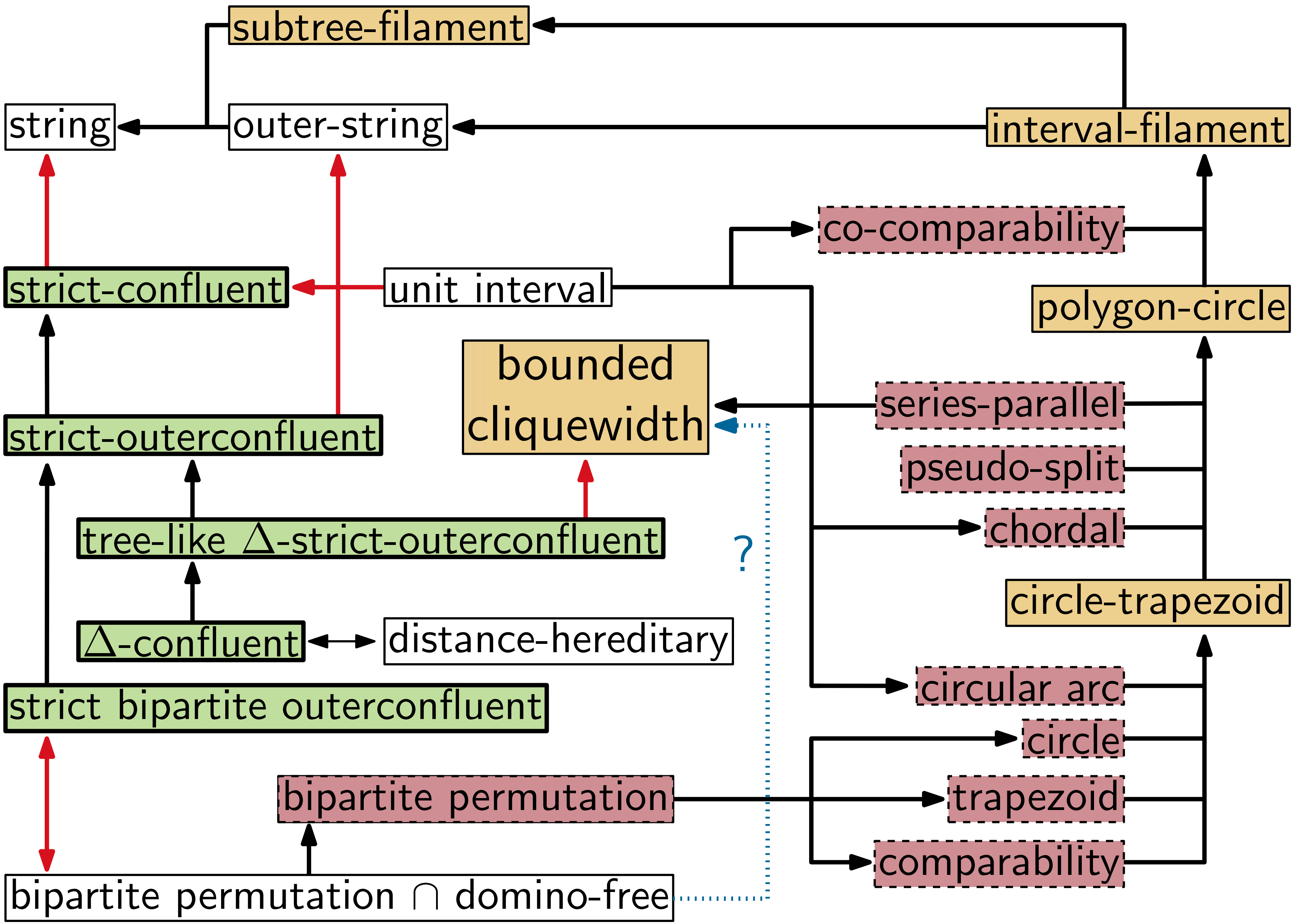
Our Results



subtree-filament
string
outer-string
interval-filament
co-comparability
strict-confluent
unit interval
polygon-circle
bounded cliquewidth
series-parallel
strict-outerconfluent
pseudo-split
chordal
tree-like Δ-strict-outerconfluent
?
circle-trapezoid
Δ-confluent
distance-hereditary
strict bipartite outerconfluent
circular arc
circle
bipartite permutation
trapezoid
comparability
bipartite permutation ∩ domino-free

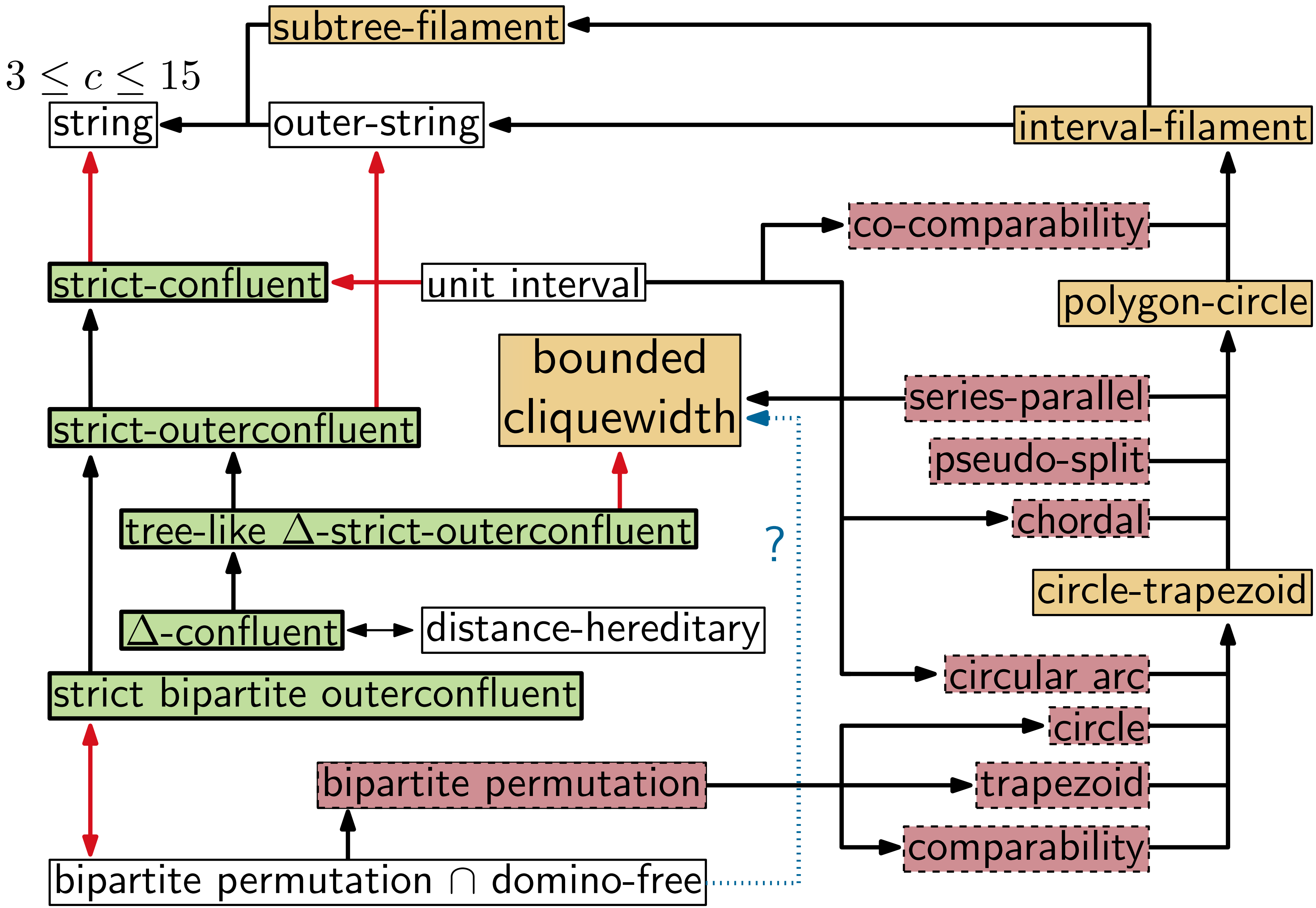
Our Results

Cop number: c [Gavenčiak et al. 18]



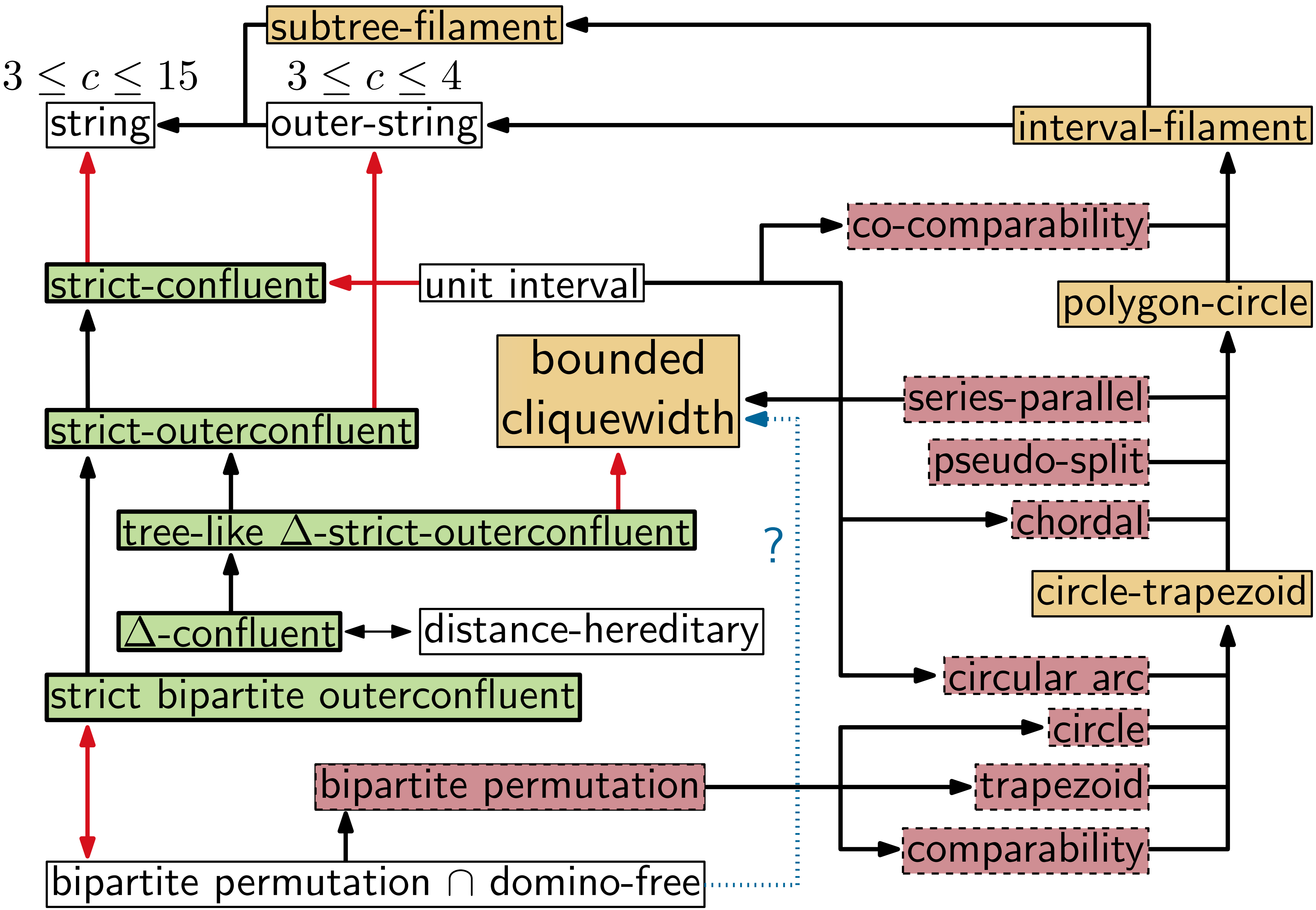
Our Results

Cop number: c [Gavenčiak et al. 18]



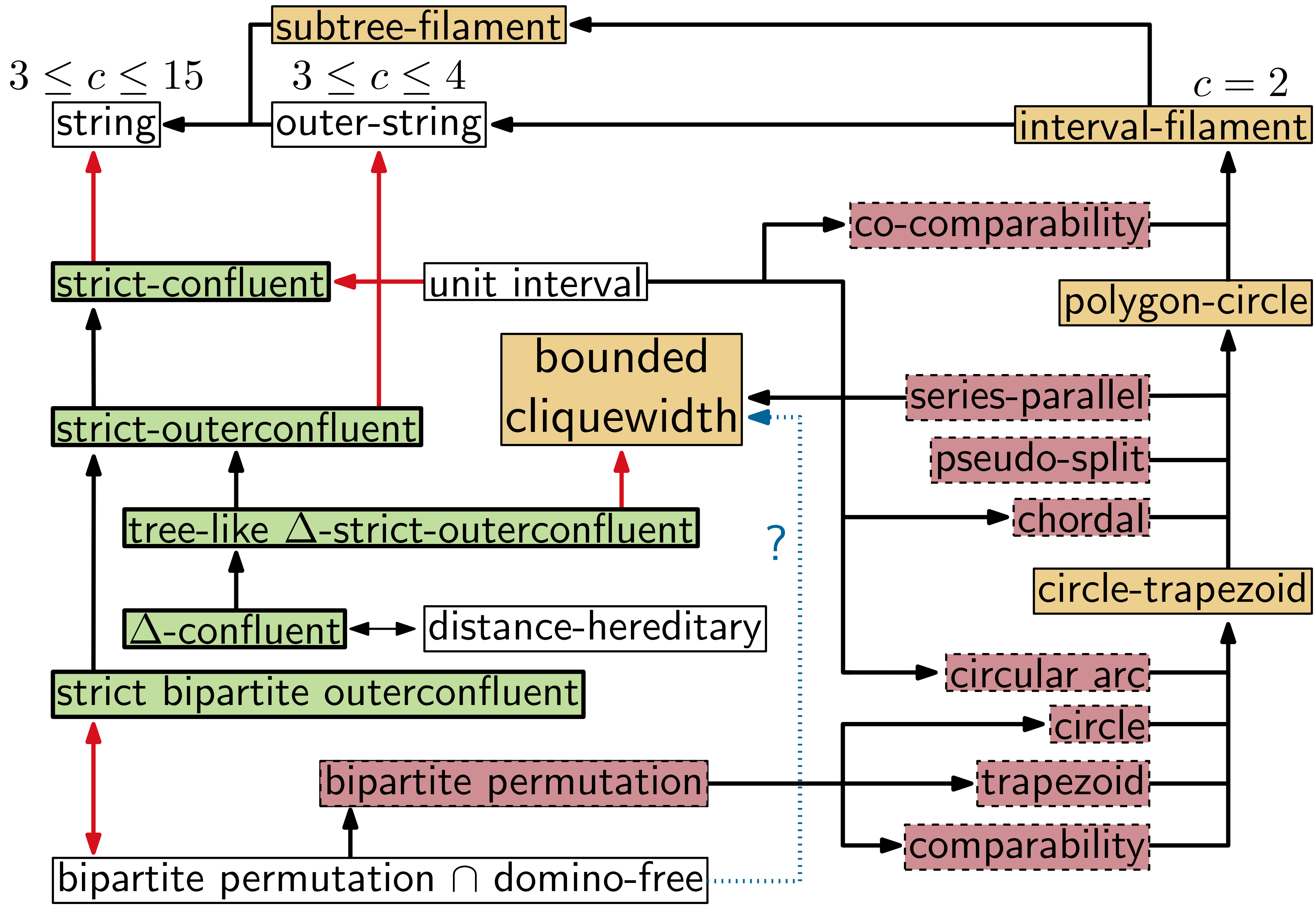
Our Results

Cop number: c [Gavenčiak et al. 18]



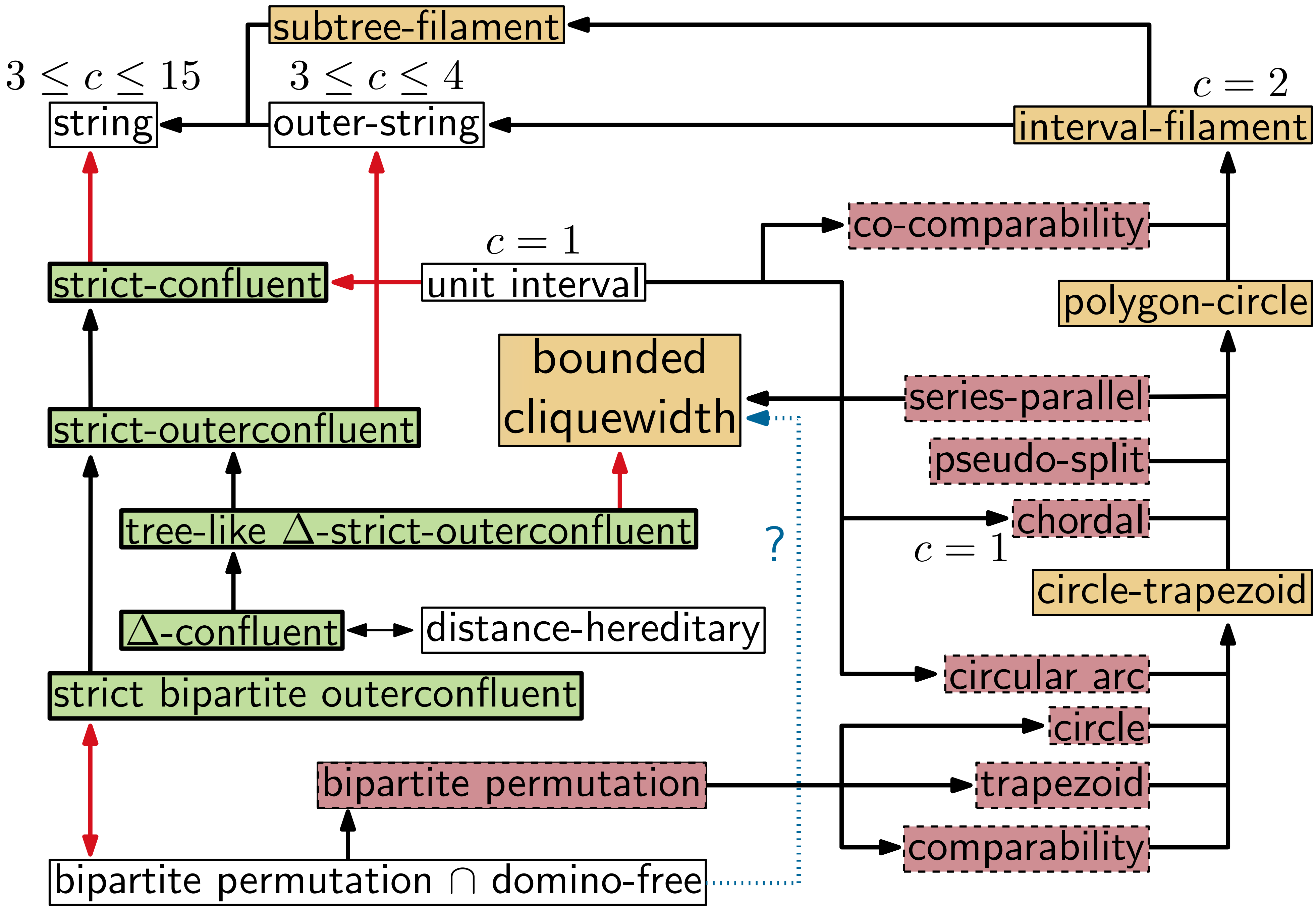
Our Results

Cop number: c [Gavenčiak et al. 18]



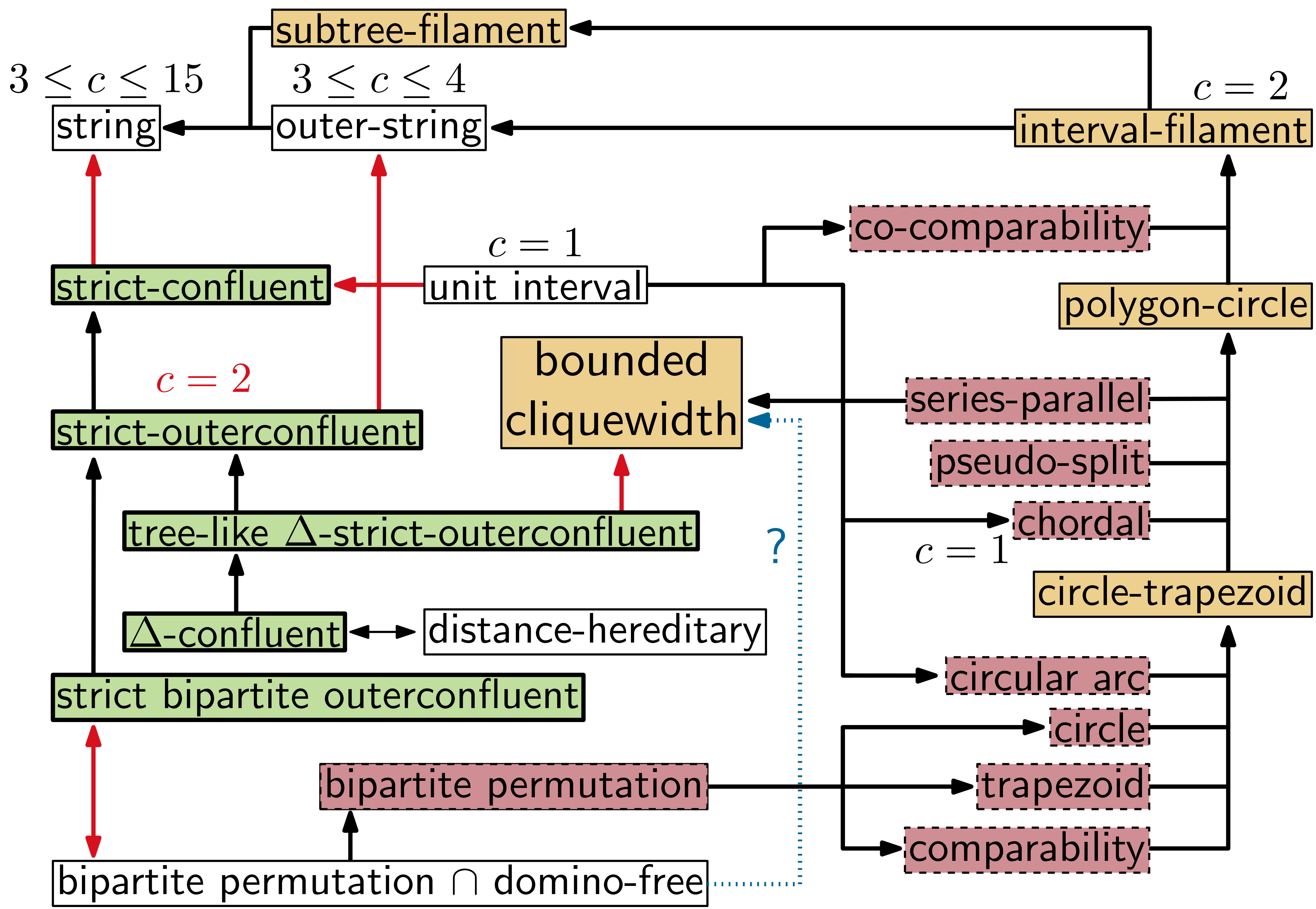
Our Results

Cop number: c [Gavenčiak et al. 18]



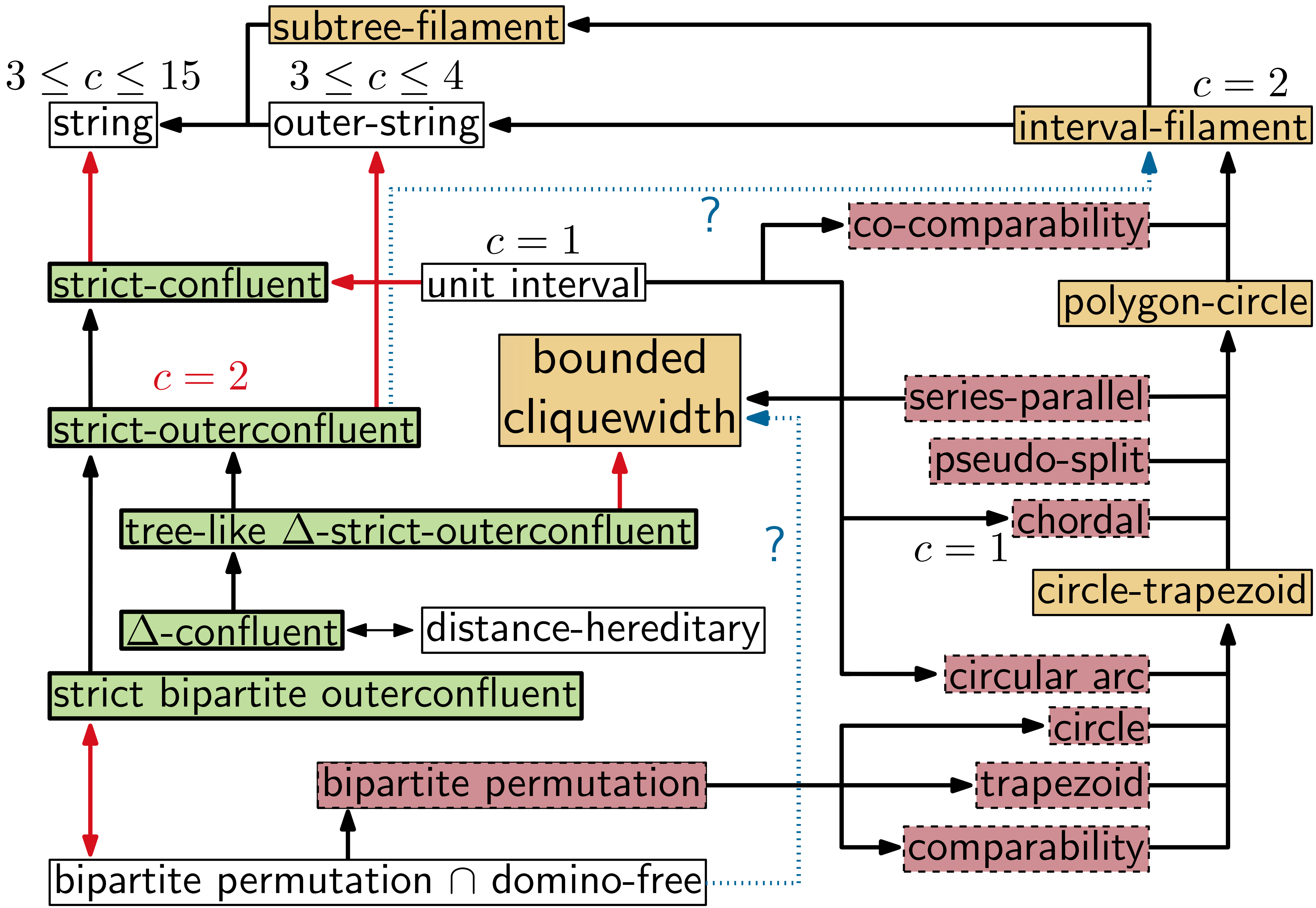
Our Results

Cop number: c [Gavenčiak et al. 18]



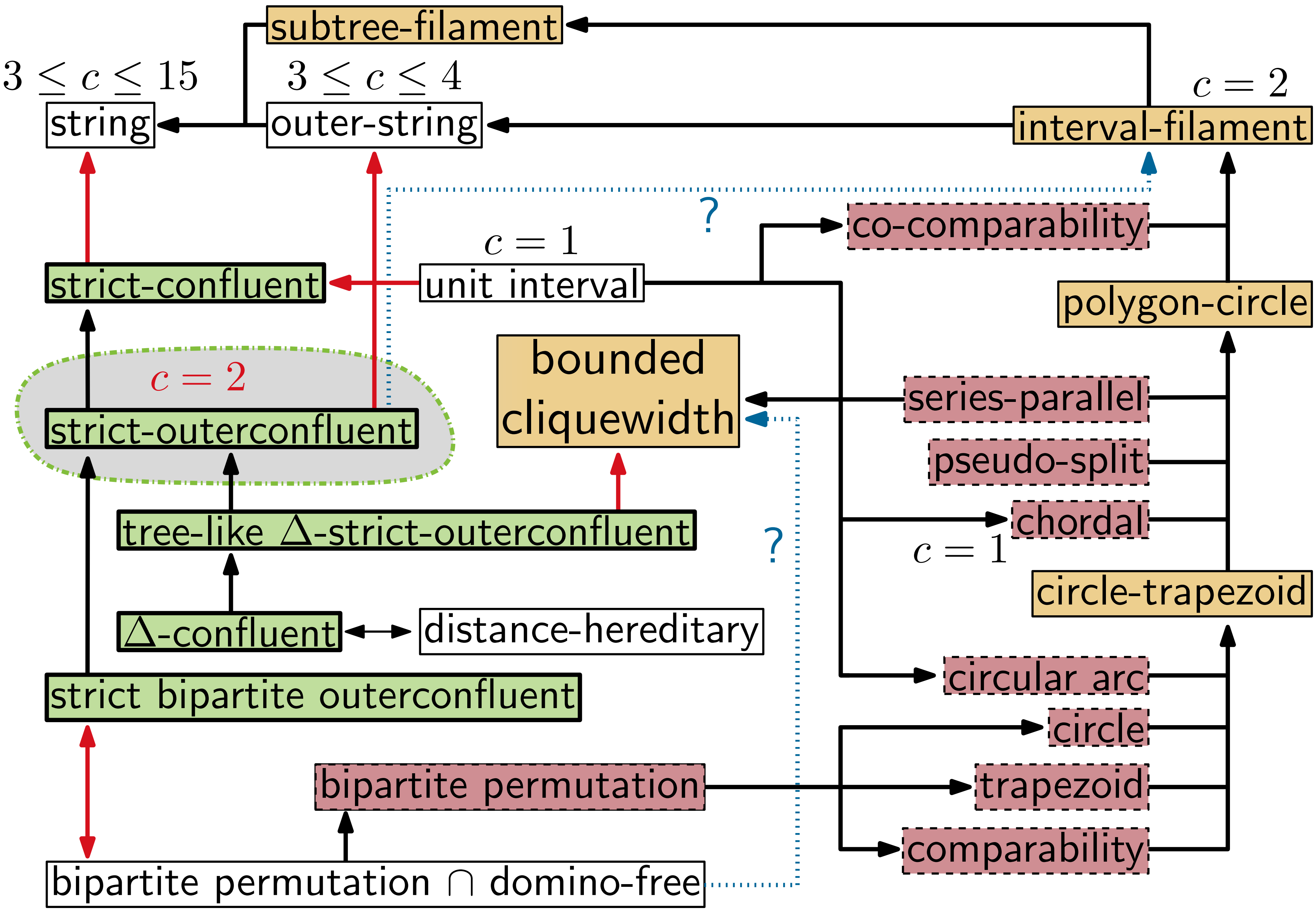
Our Results

Cop number: c [Gavenčiak et al. 18]



Our Results

Cop number: c [Gavenčiak et al. 18]



Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

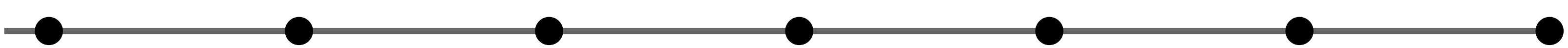
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A path



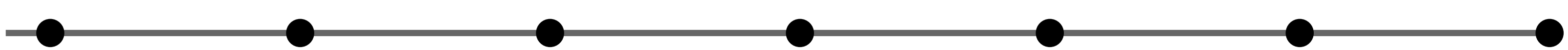
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A path



Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A path

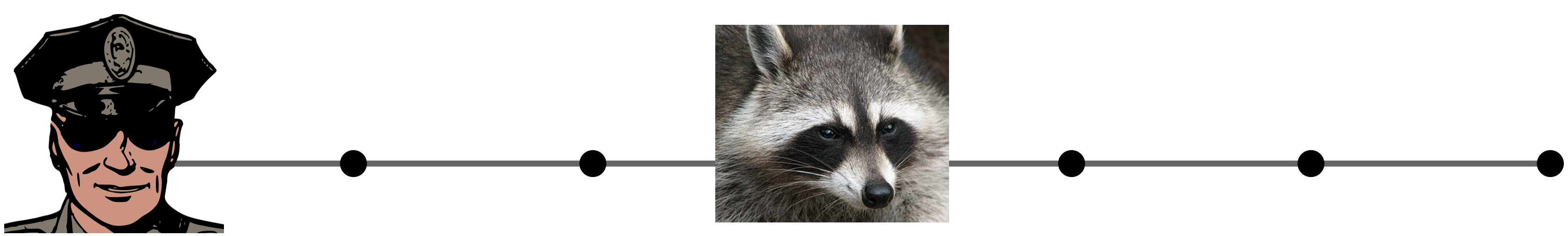
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A path



Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A path

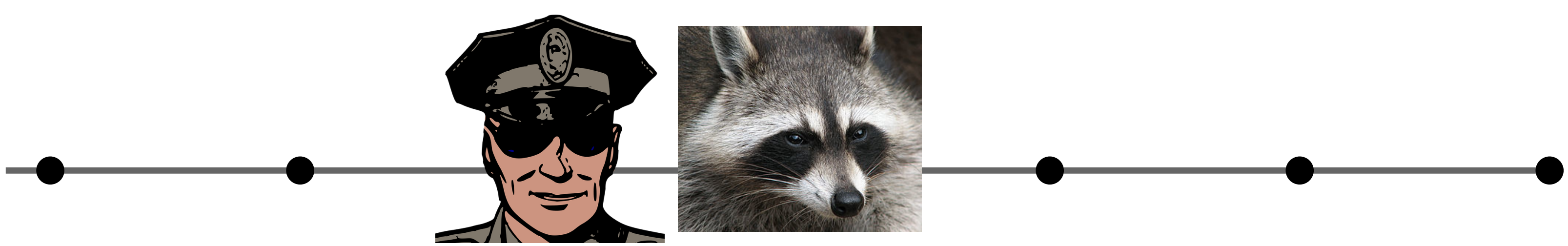
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A path



Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A path

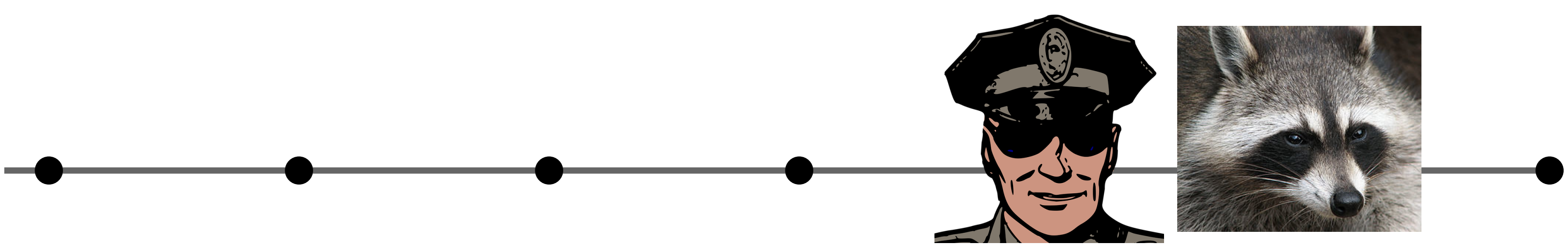
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A path



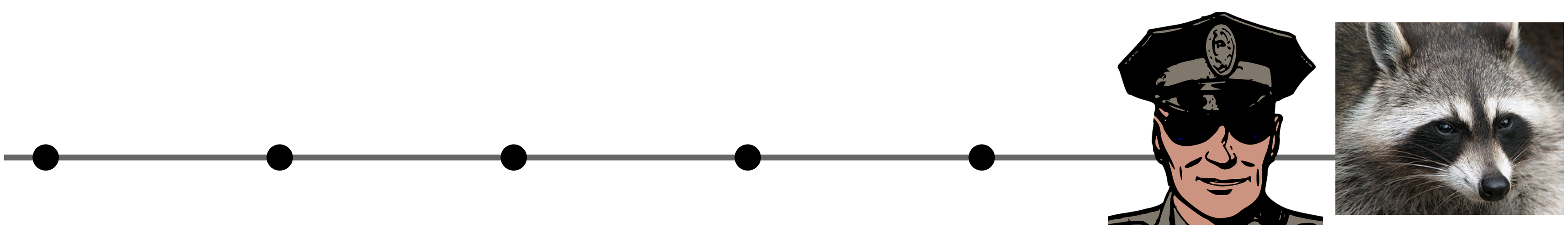
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A path



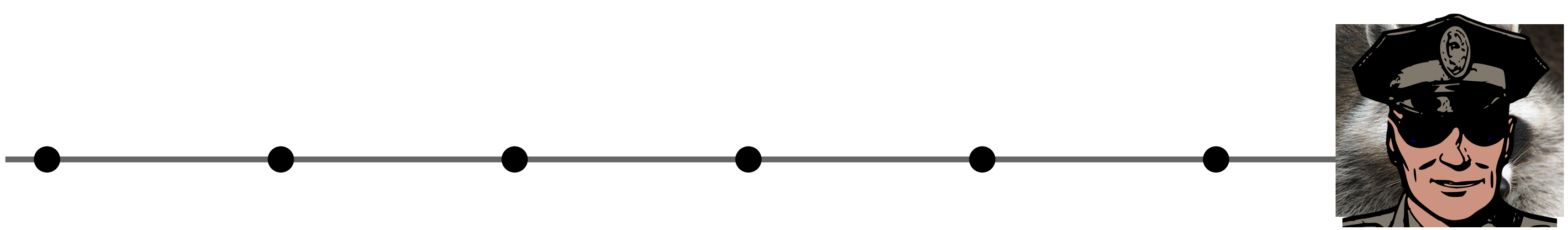
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A path



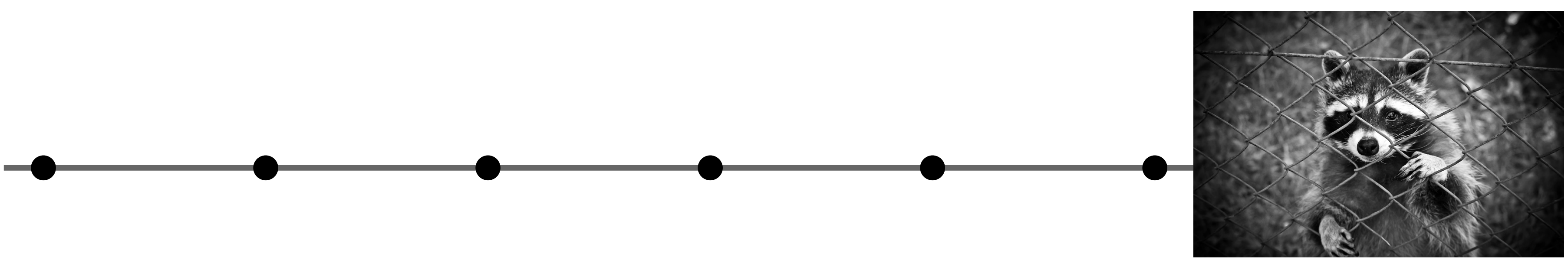
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A path



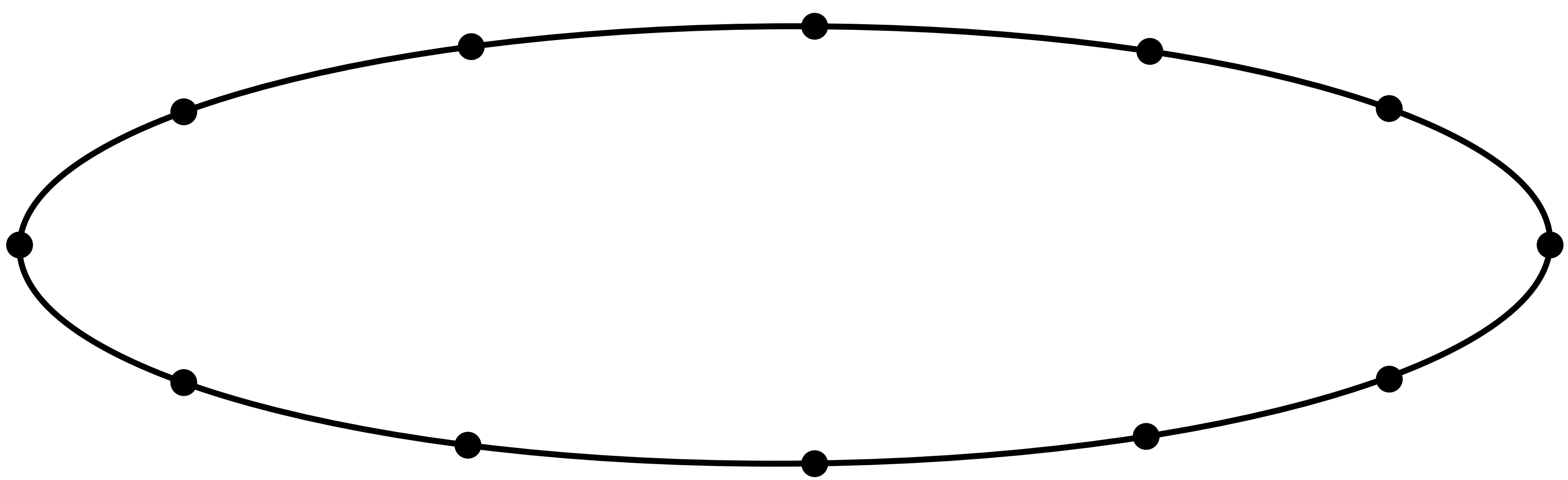
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



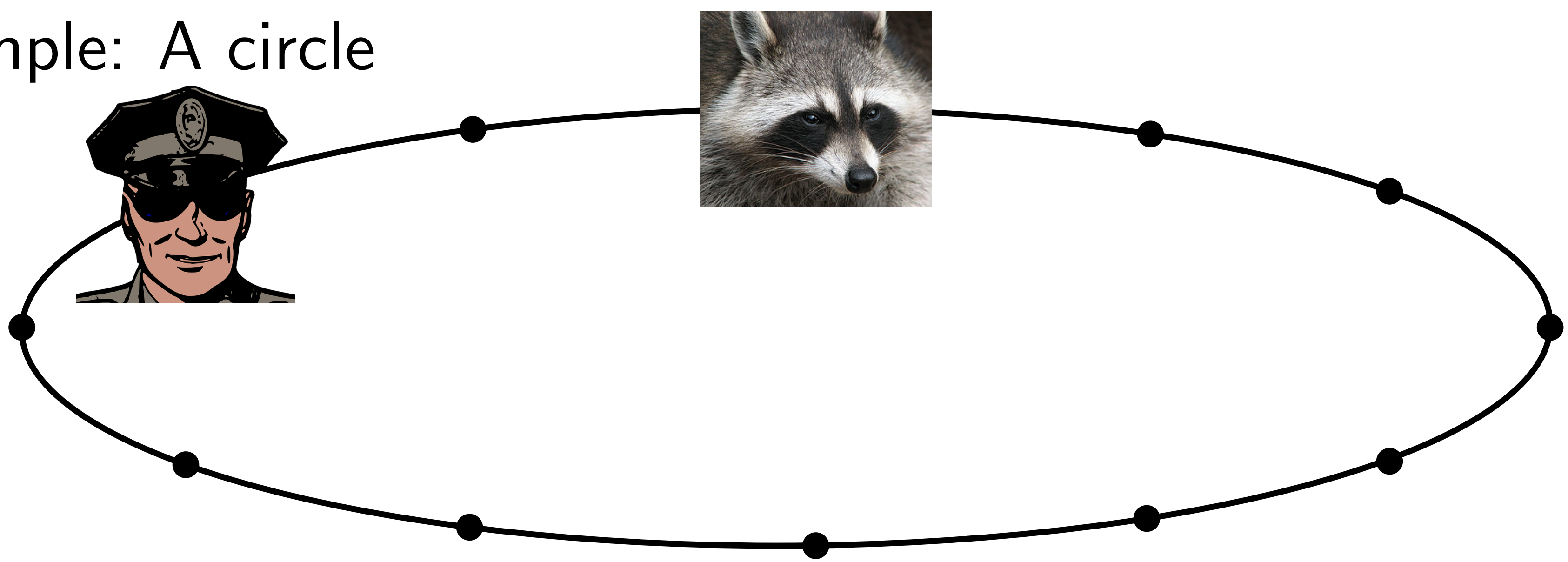
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



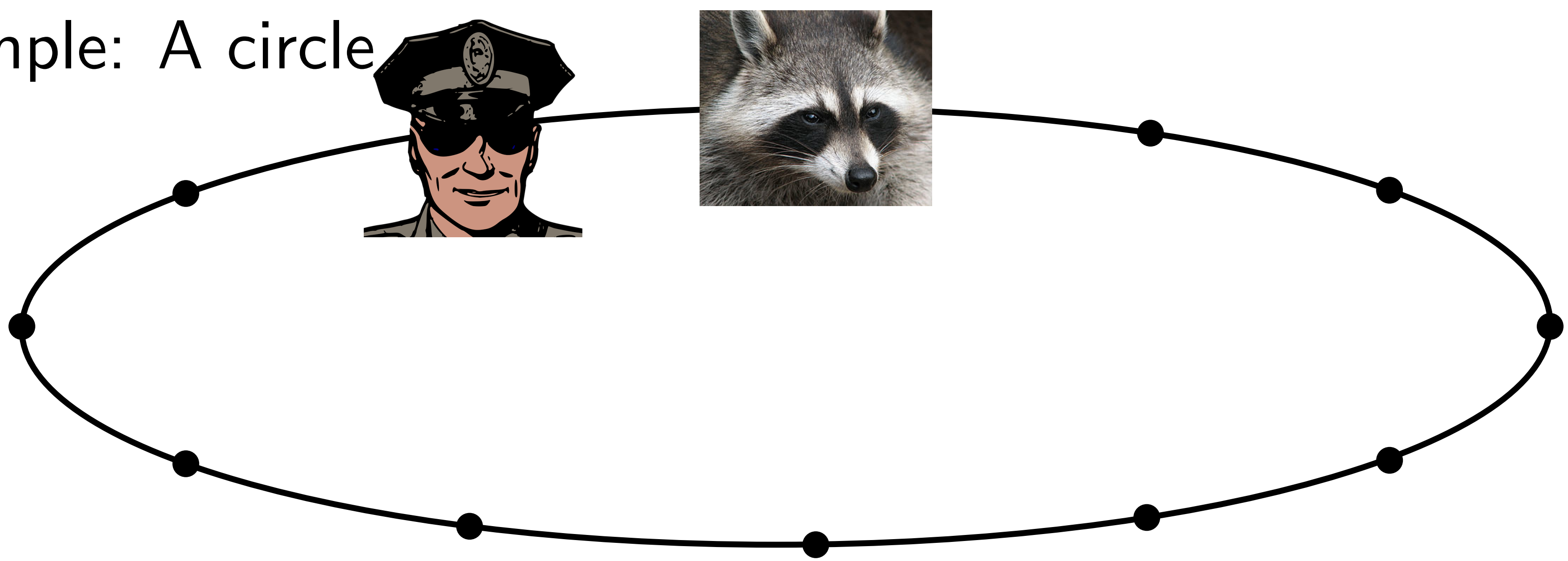
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



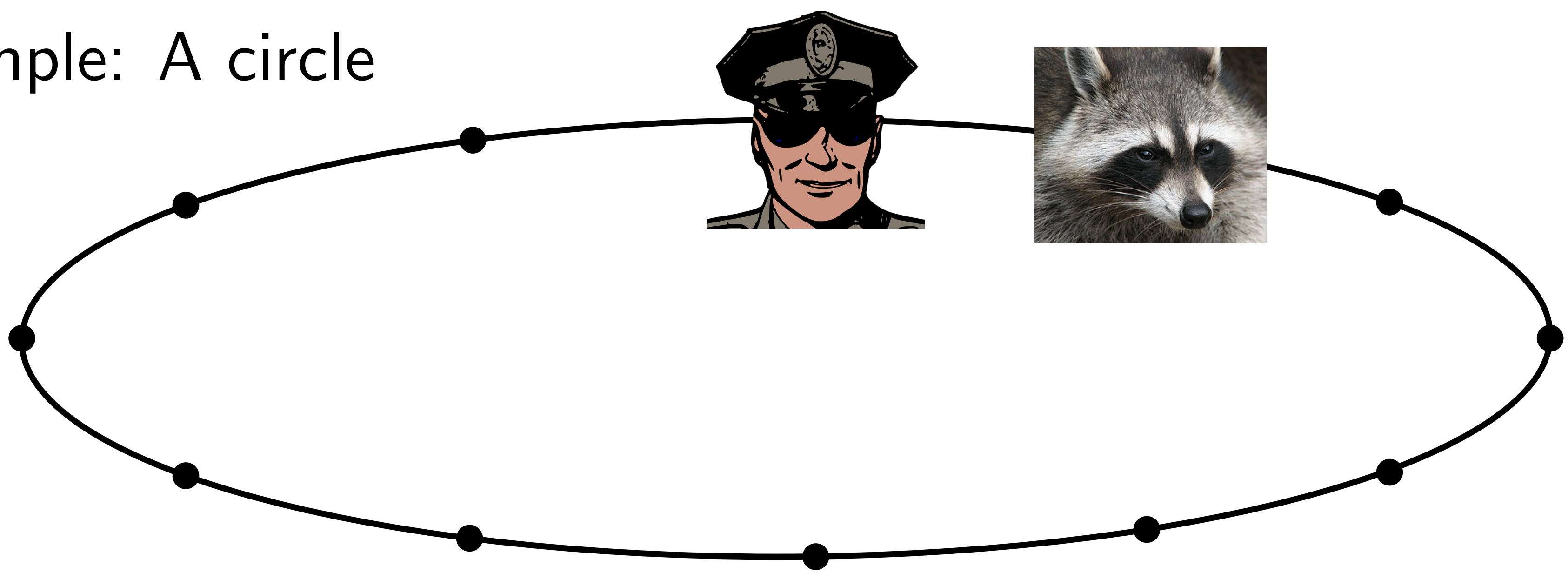
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



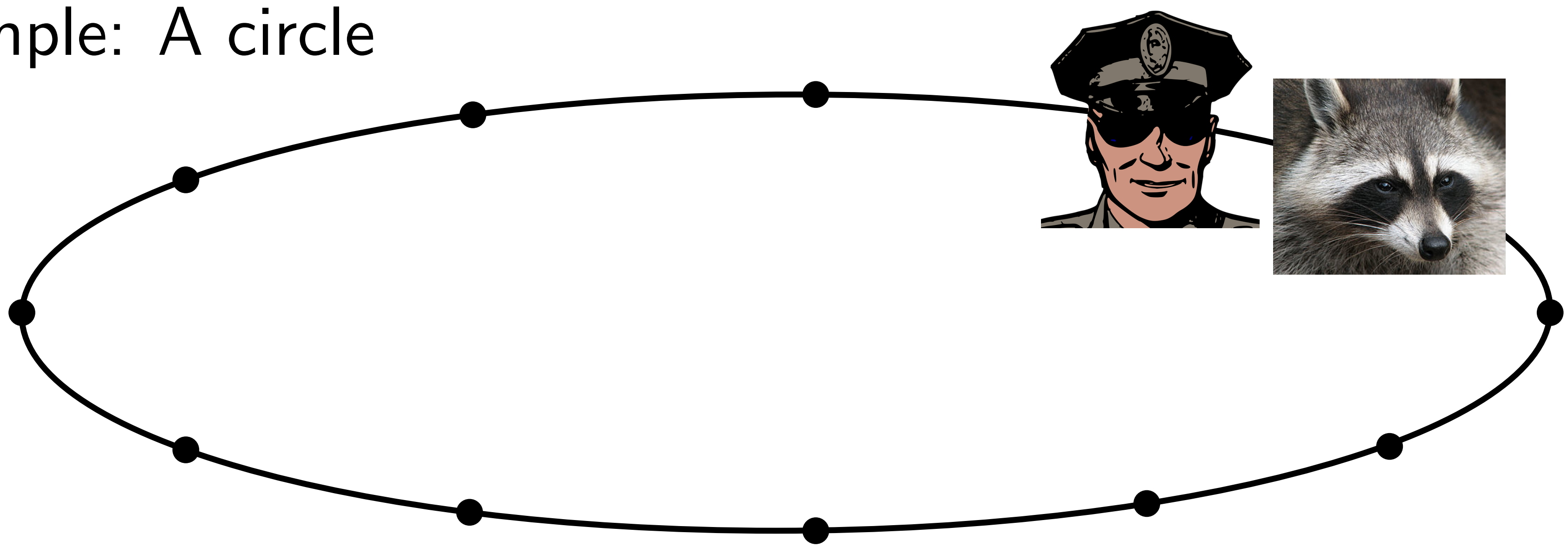
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



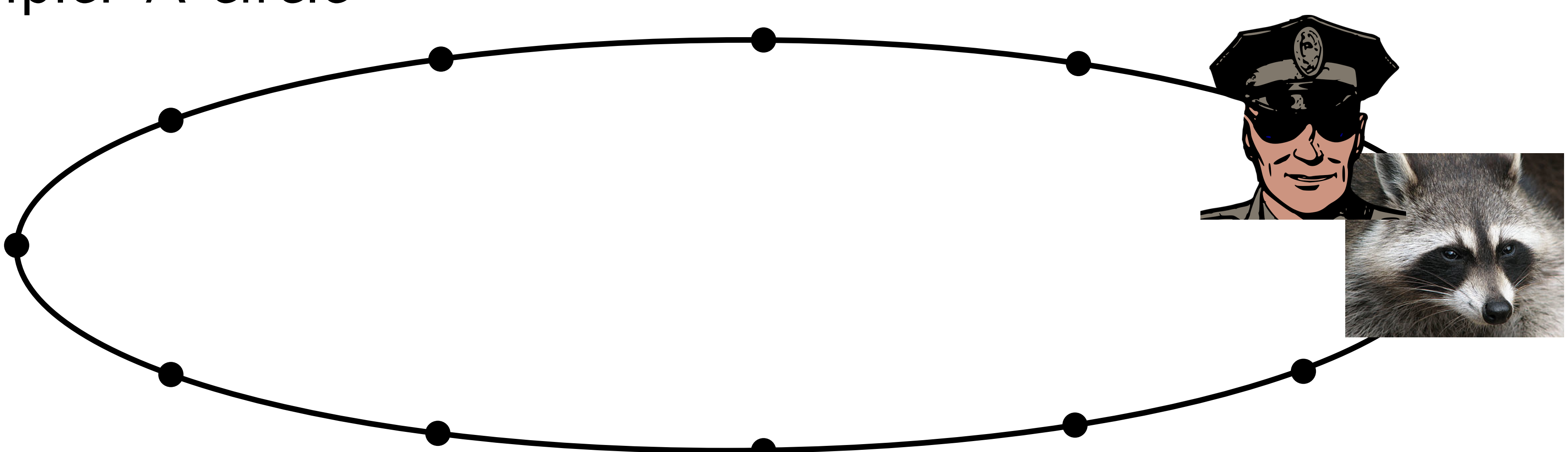
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



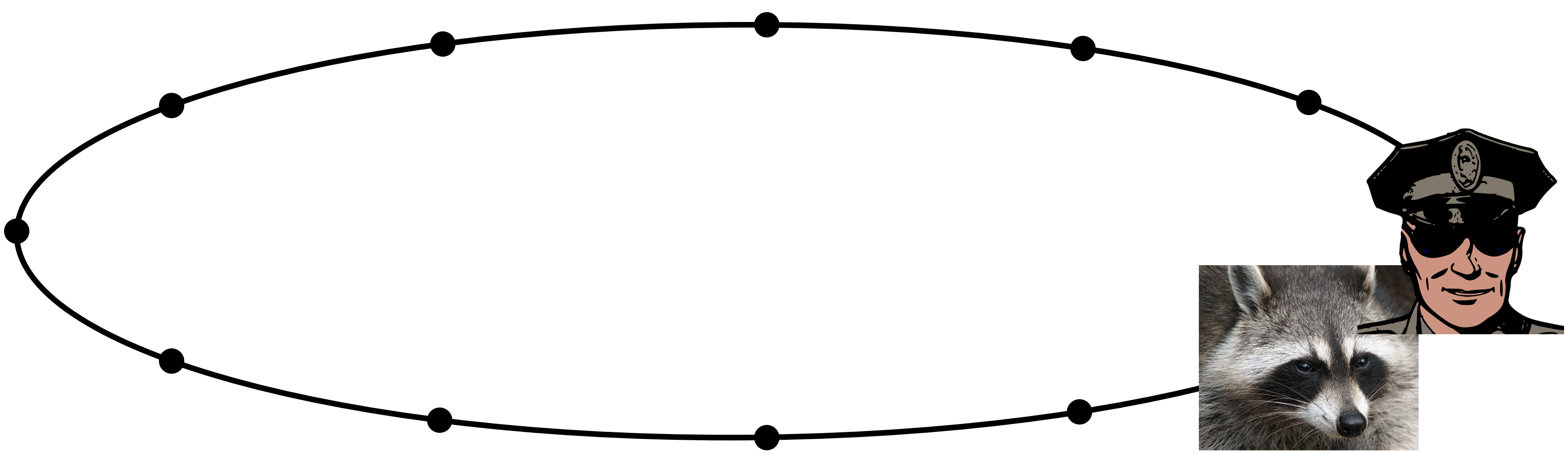
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



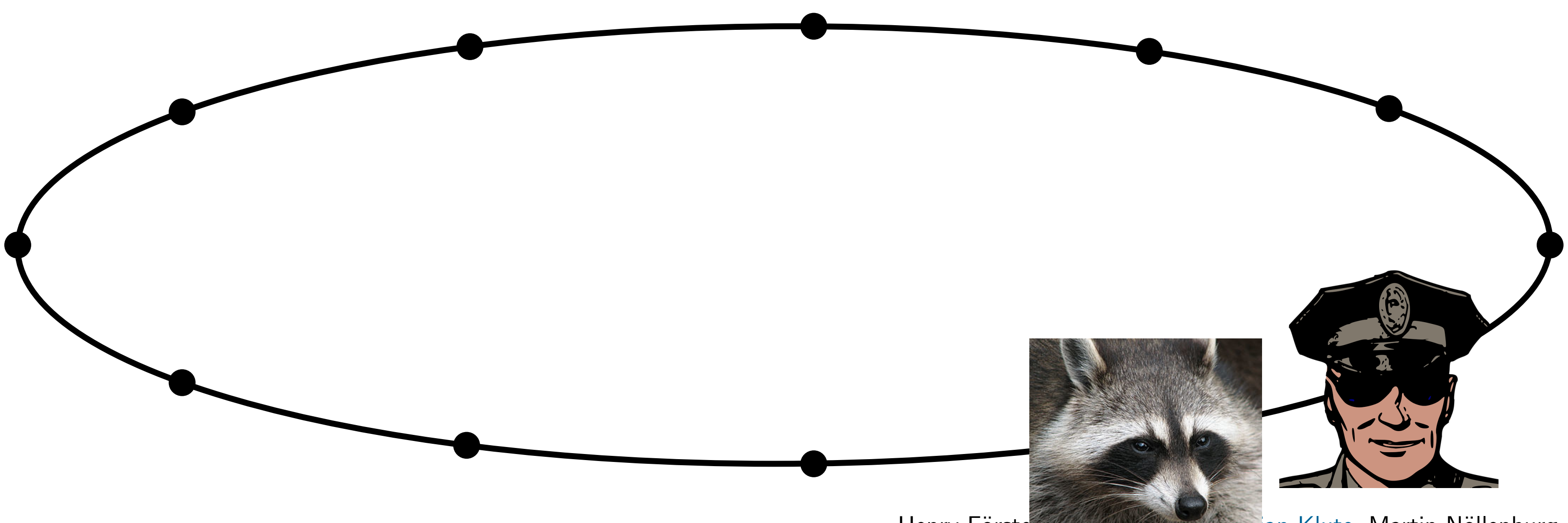
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



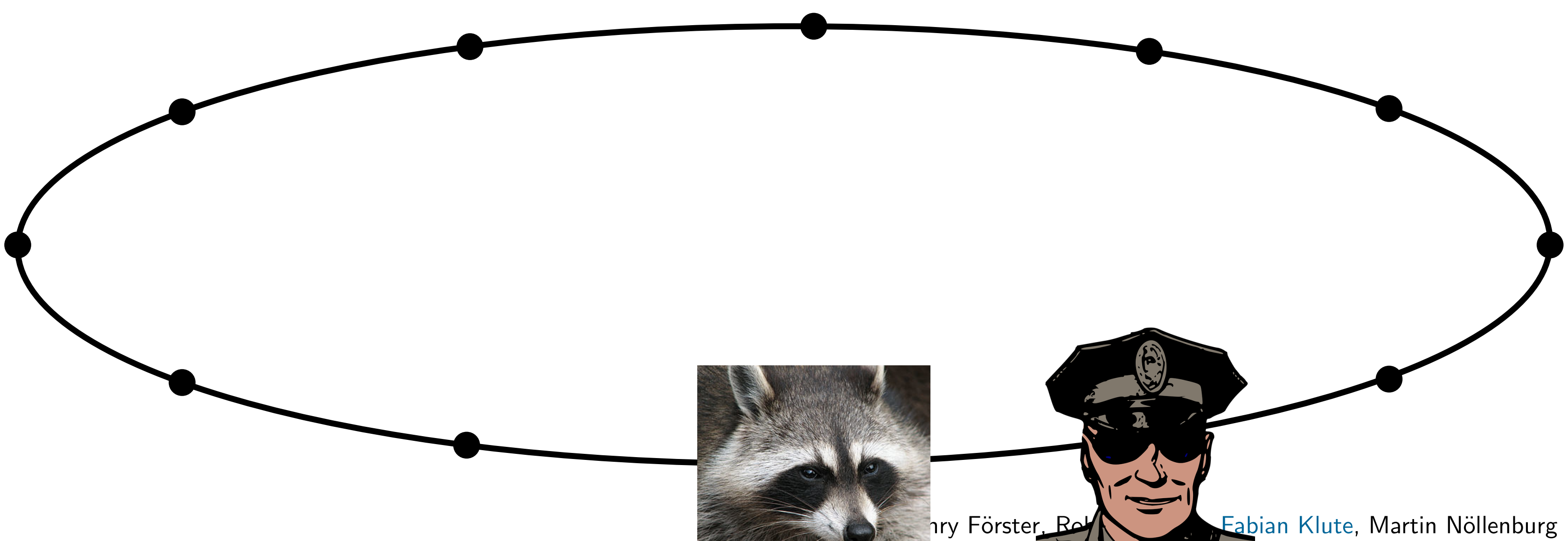
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



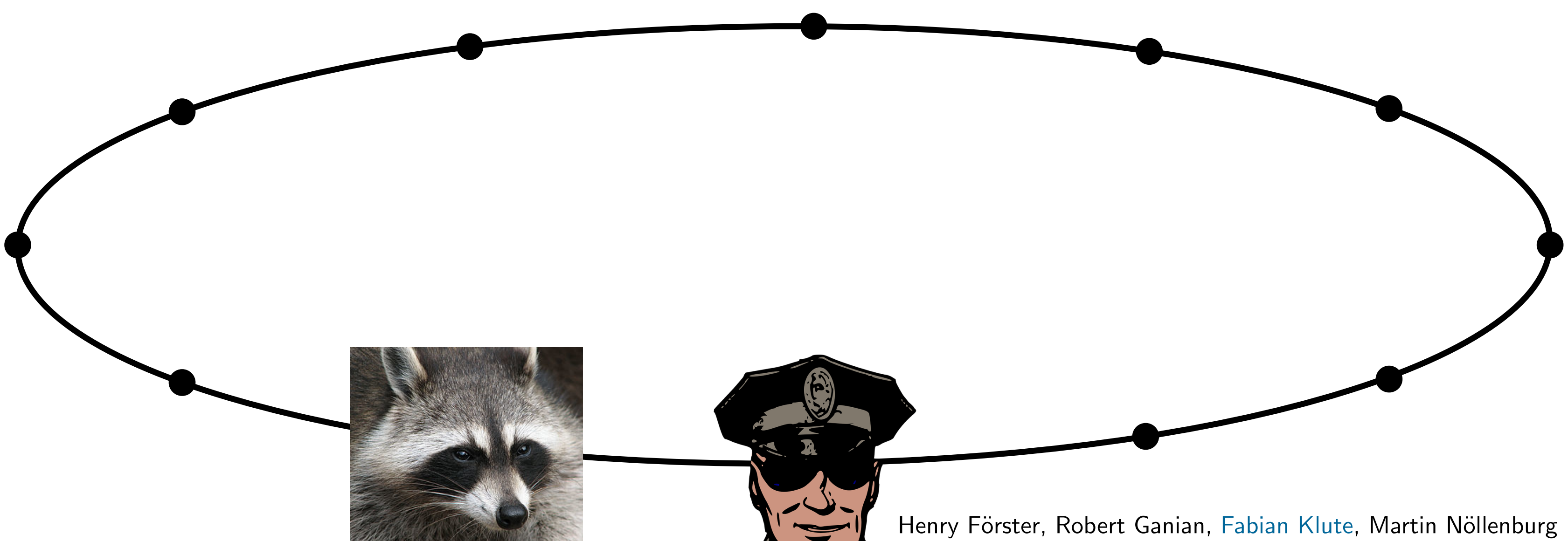
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



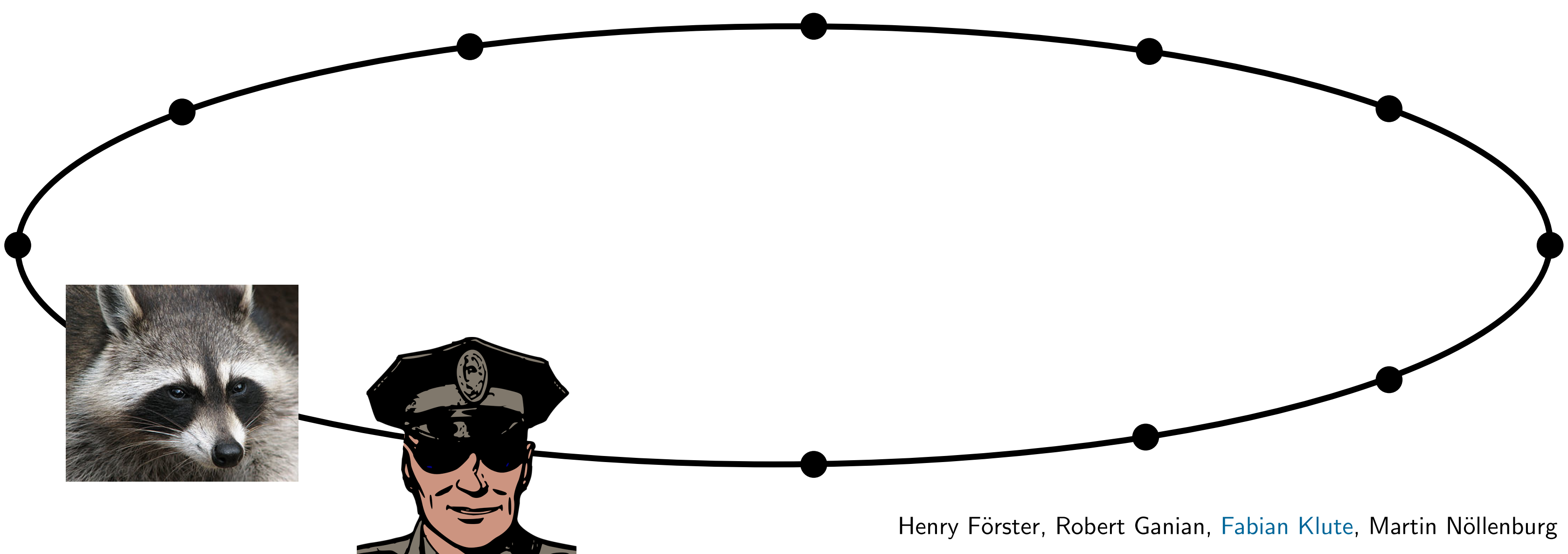
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



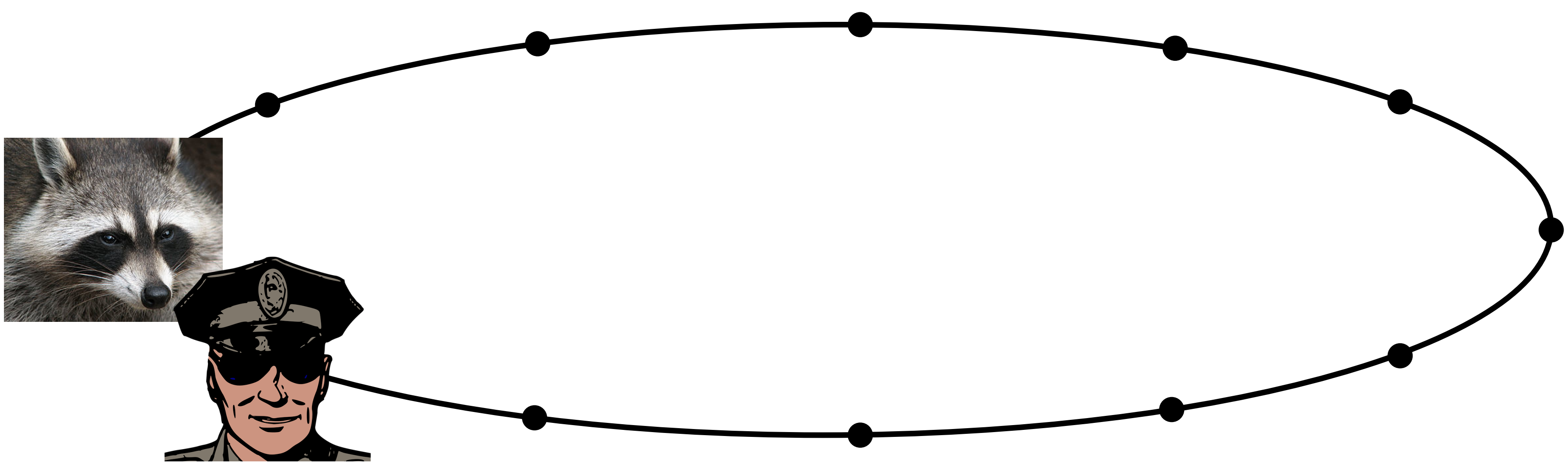
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



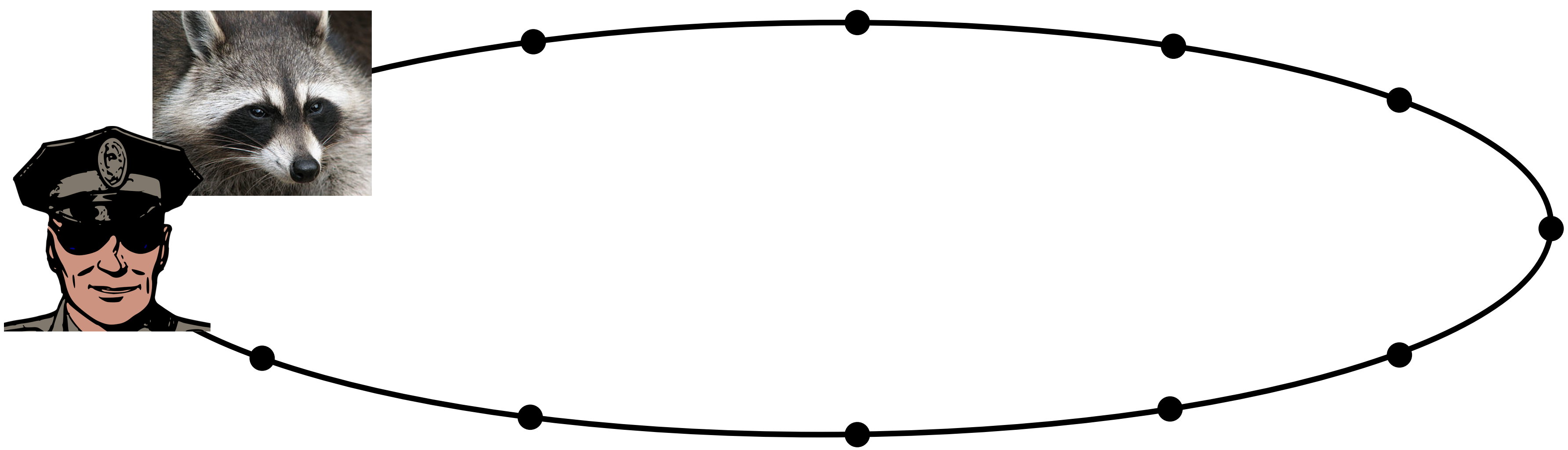
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



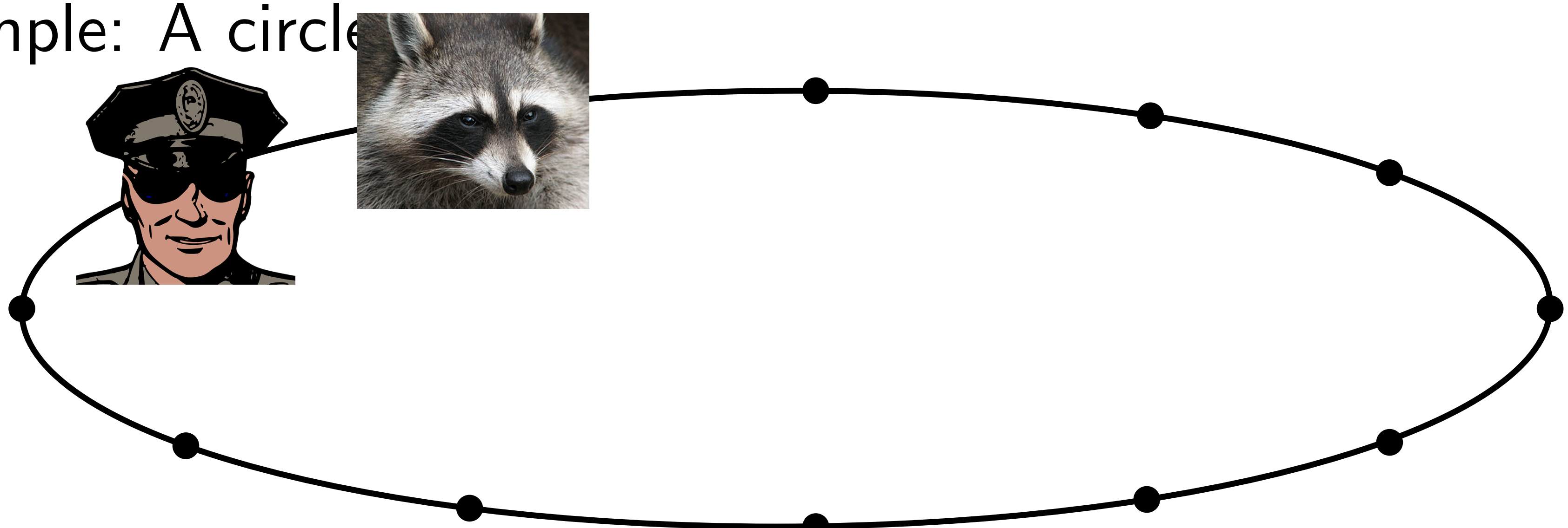
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



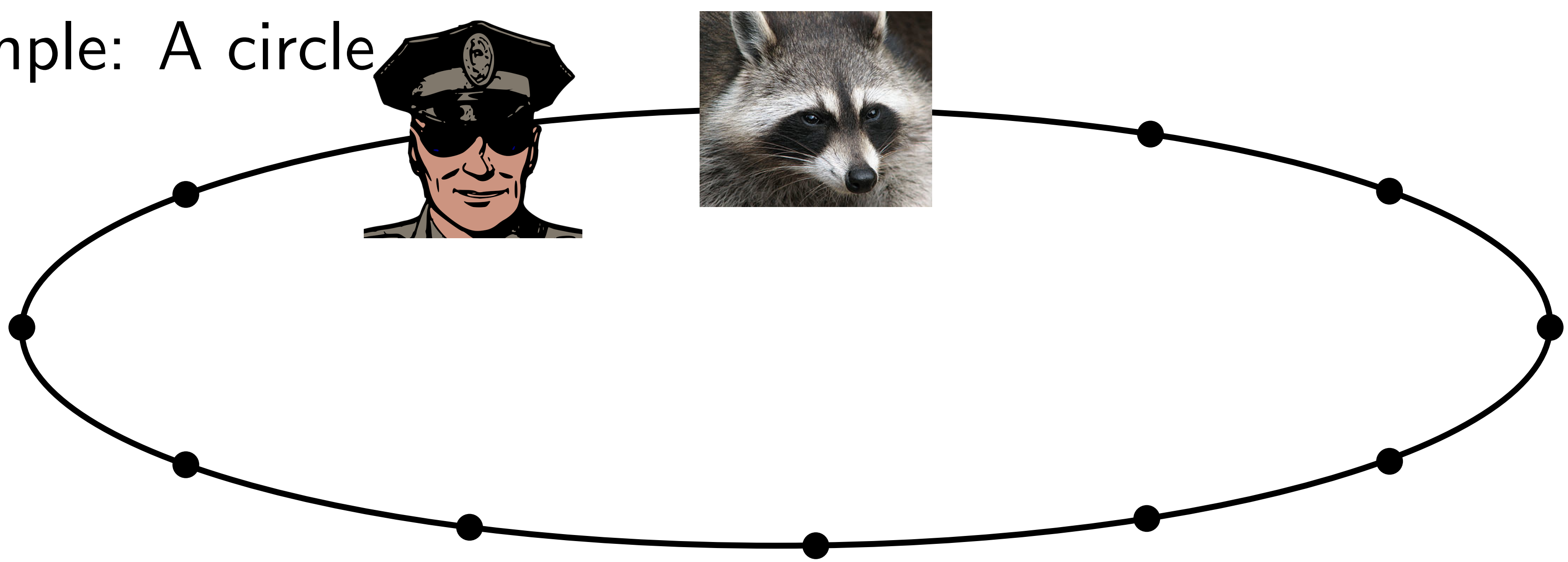
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



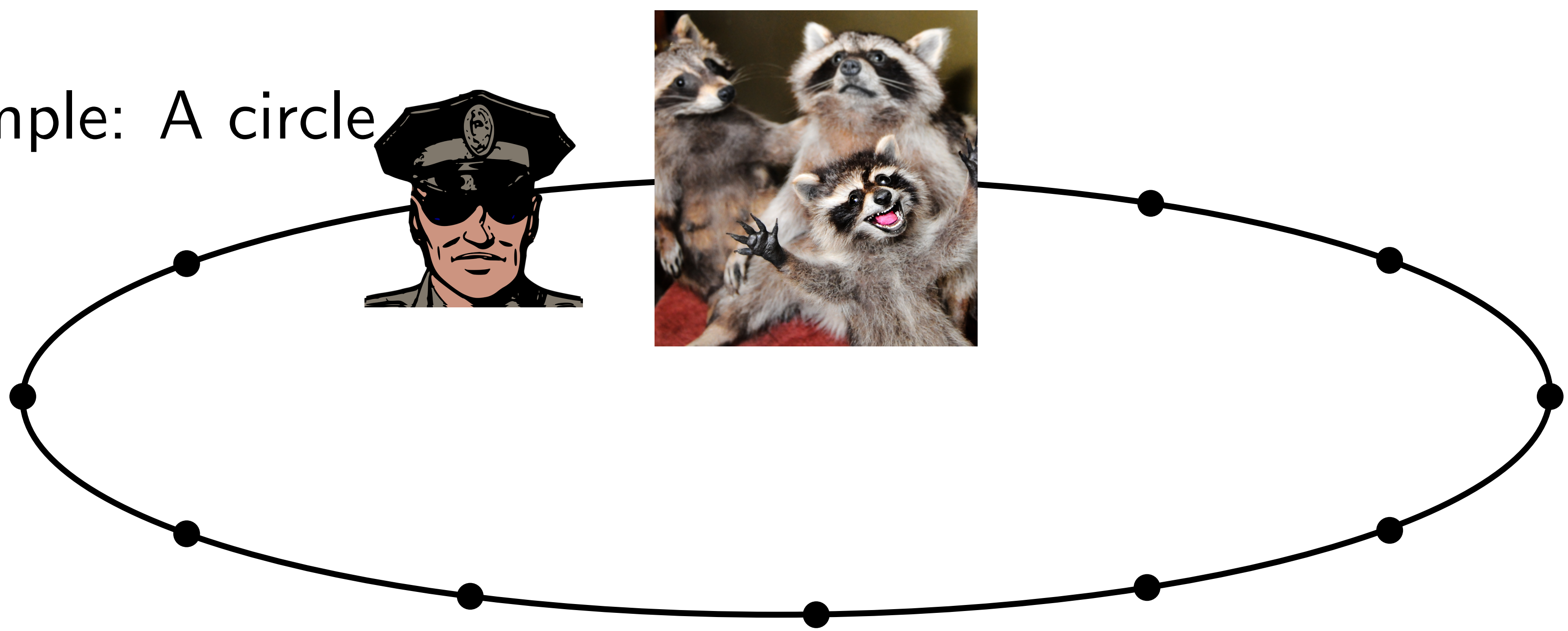
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



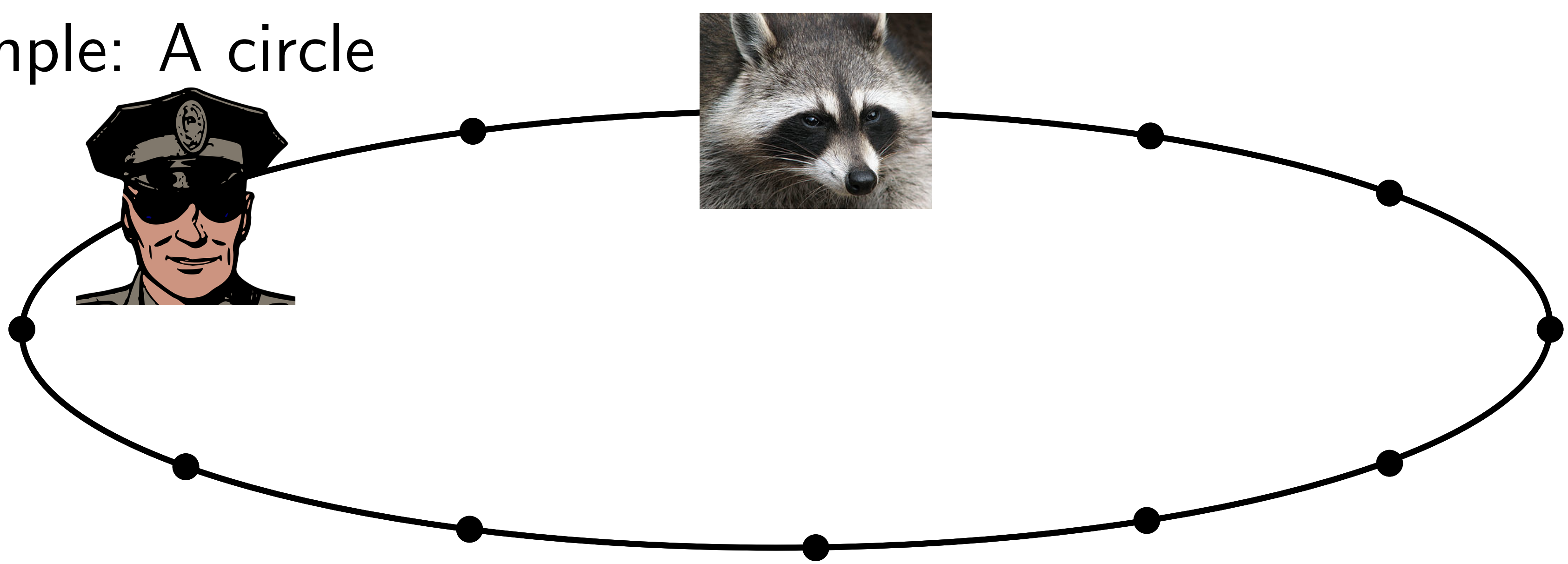
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



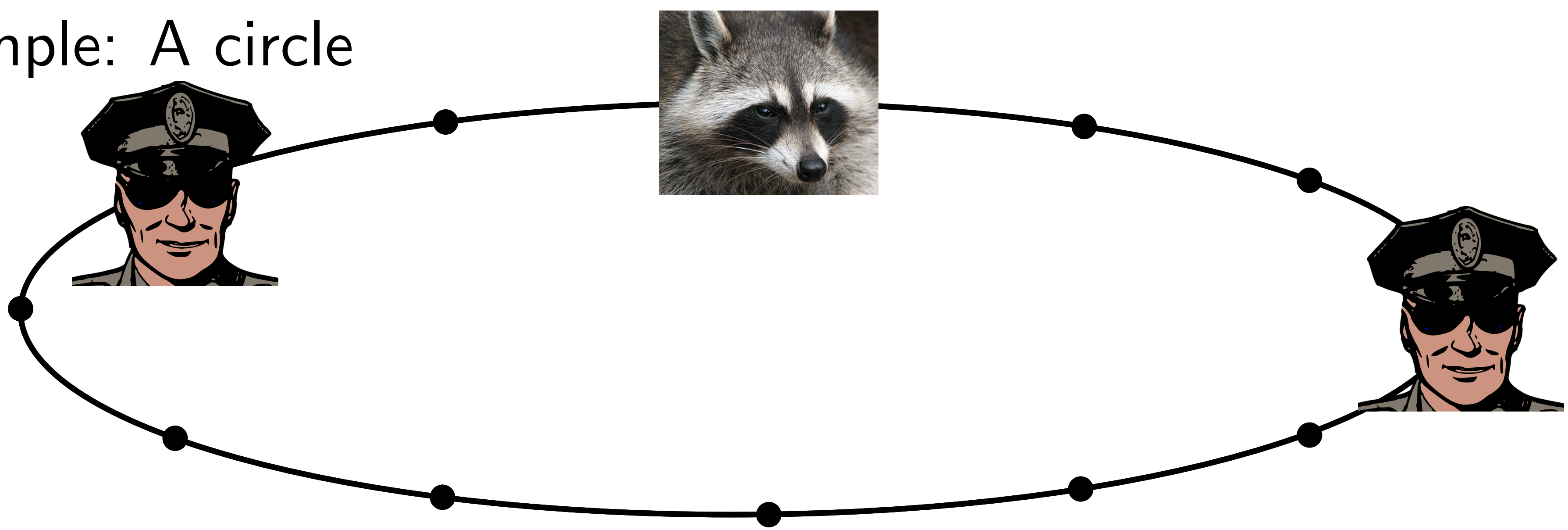
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



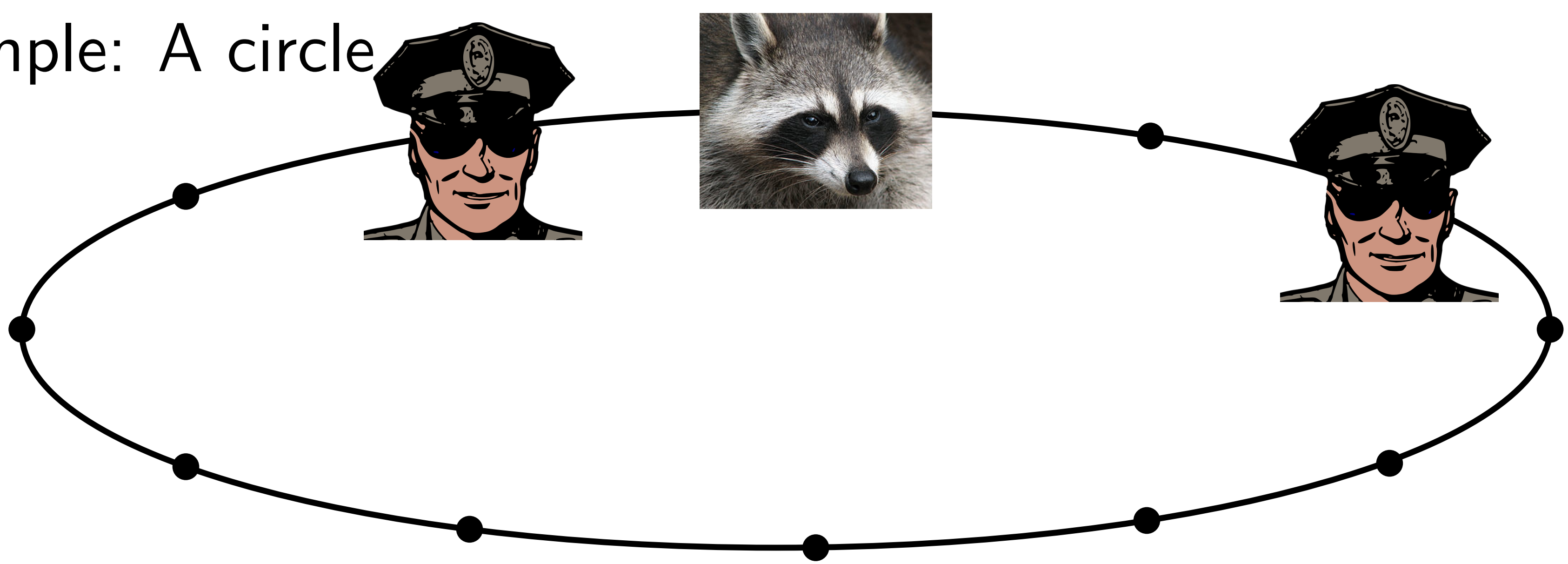
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



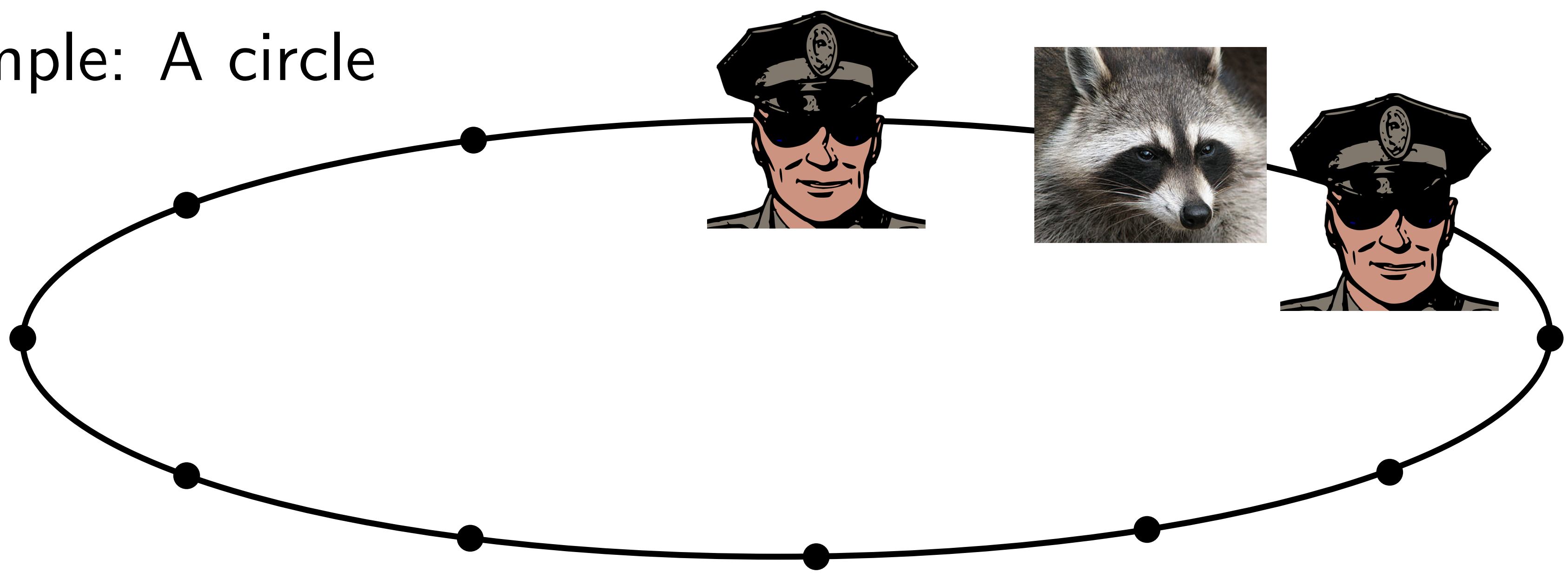
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



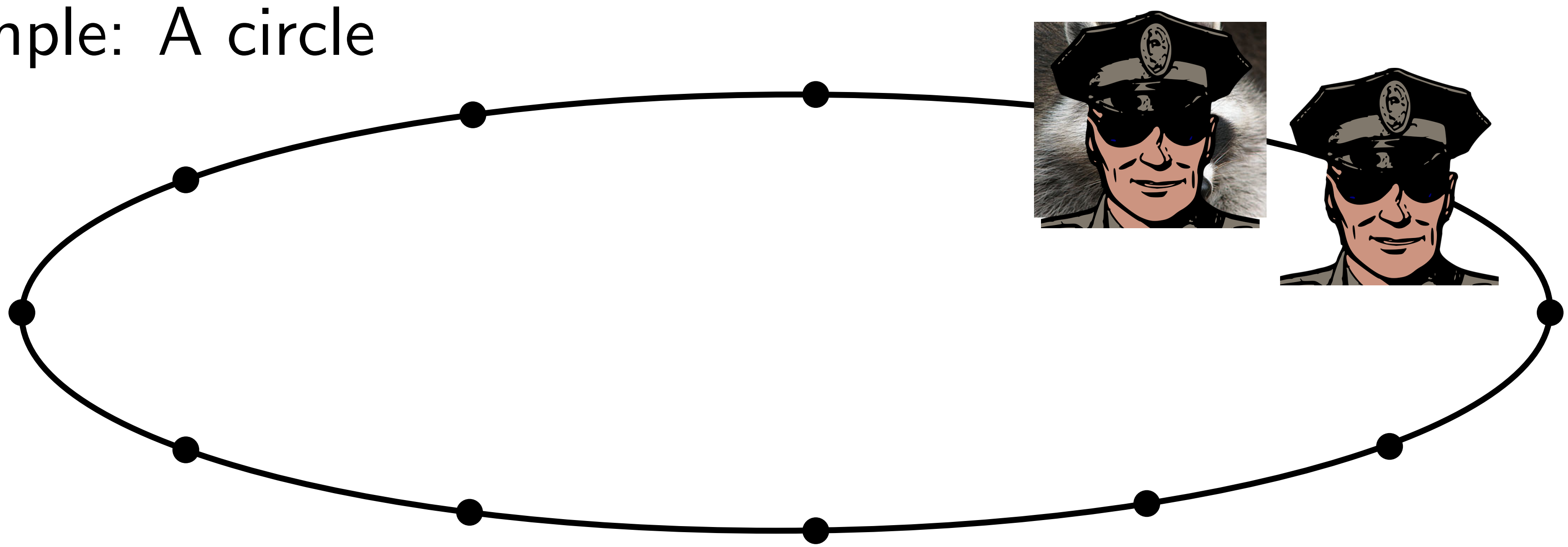
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle



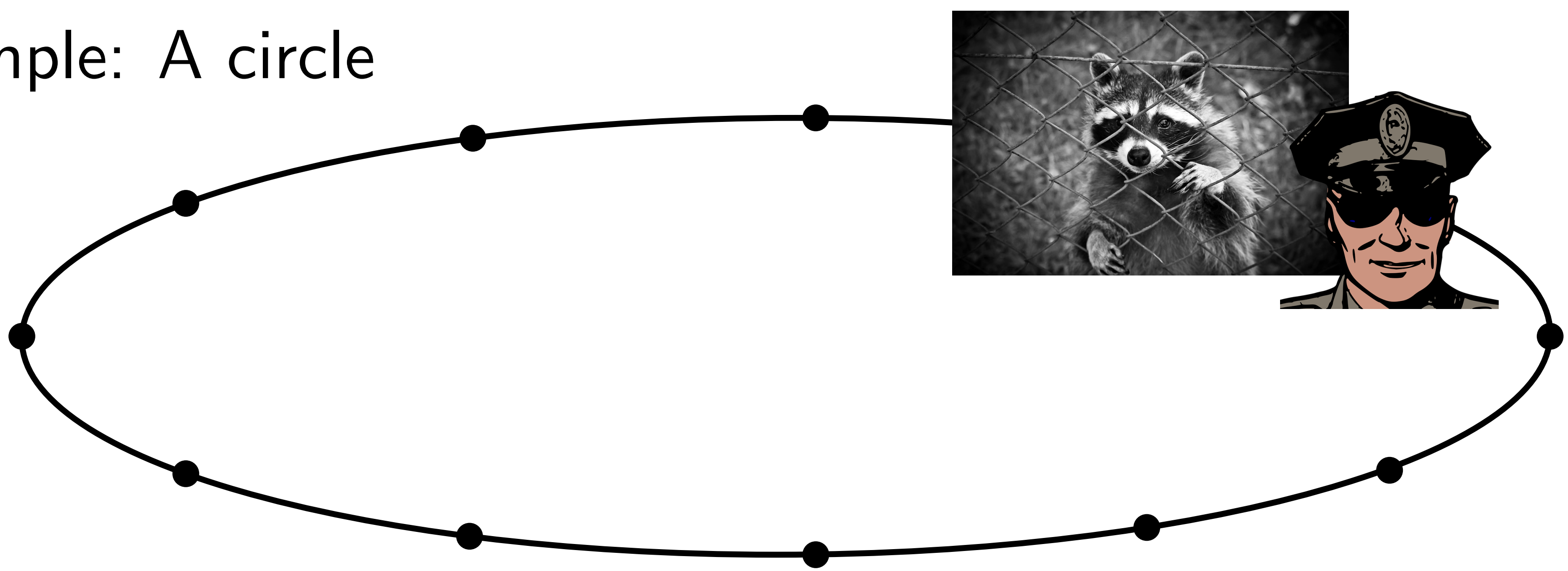
Cops and Robbers

Simple two player game on a graph

- Cop player has k cop tokens
- Robber player has 1 robber token
- In a turn player can move their token to adjacent vertices
- Robber is caught if it coincides with a cop

Cop-number: What is the smallest k such that cop player wins?

Example: A circle

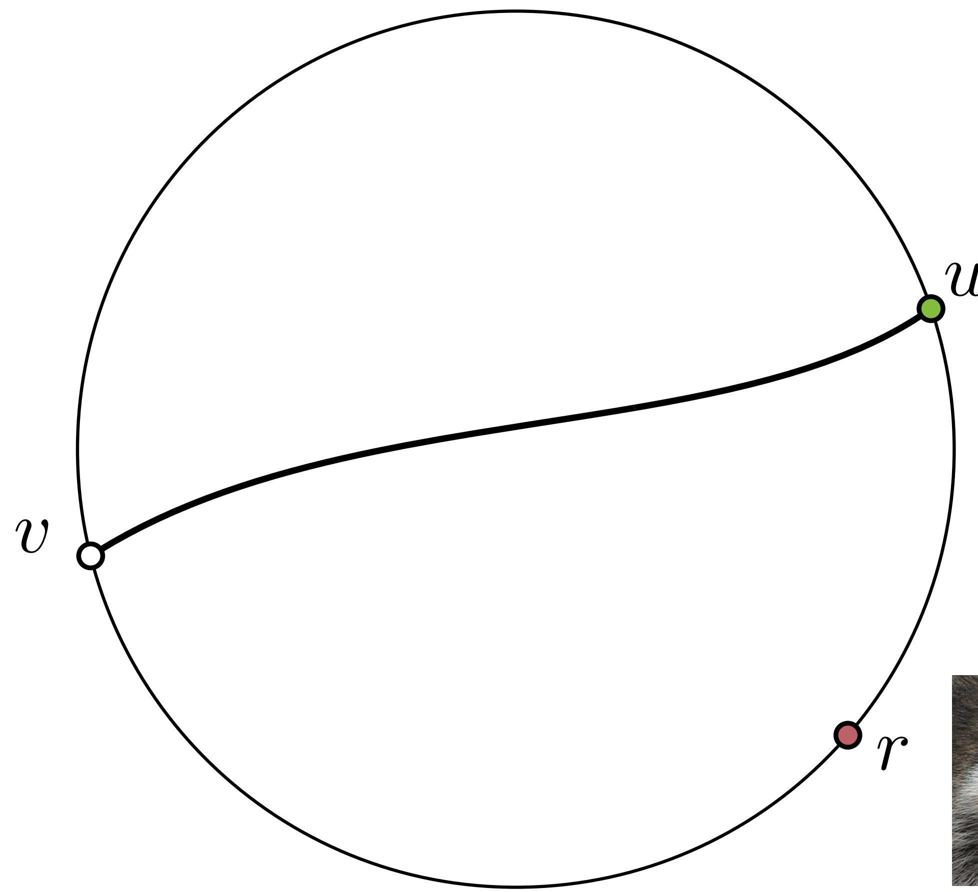


Cops and Robbers meeting SOC Graphs

Result: Strict outer-confluent graphs have cop-number 2

Cops and Robbers meeting SOC Graphs

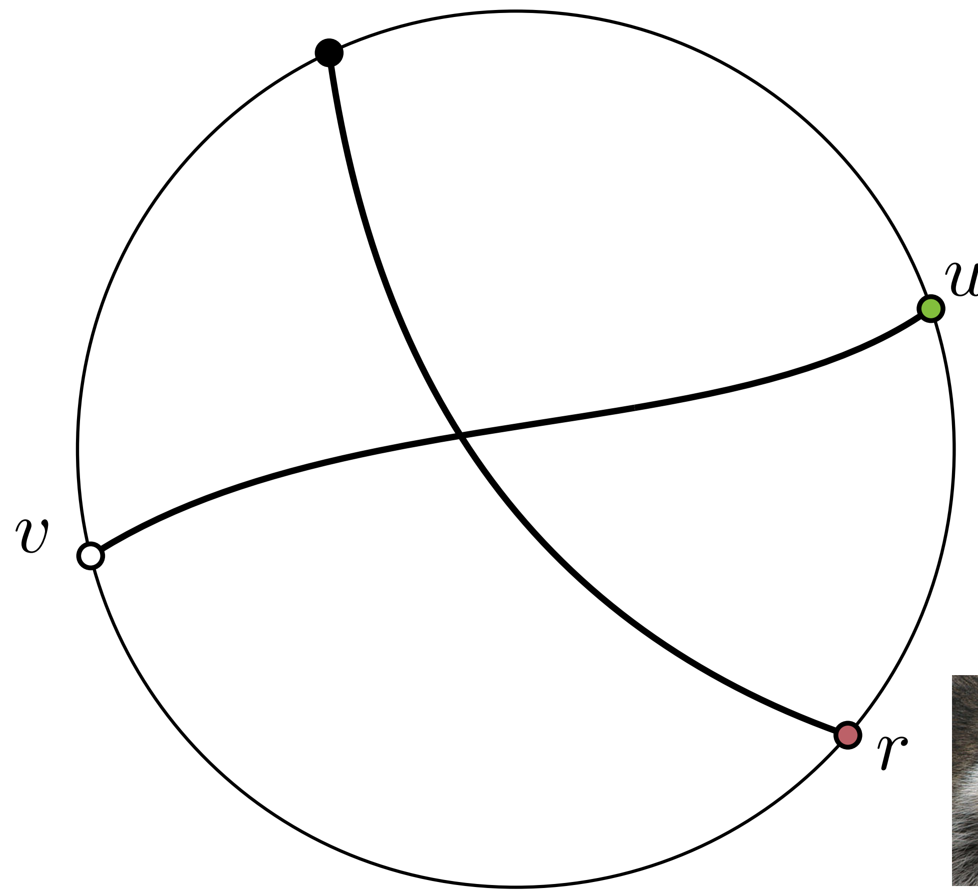
Result: Strict outer-confluent graphs have cop-number 2



Using one cop, robber is “locked” between u and v

Cops and Robbers meeting SOC Graphs

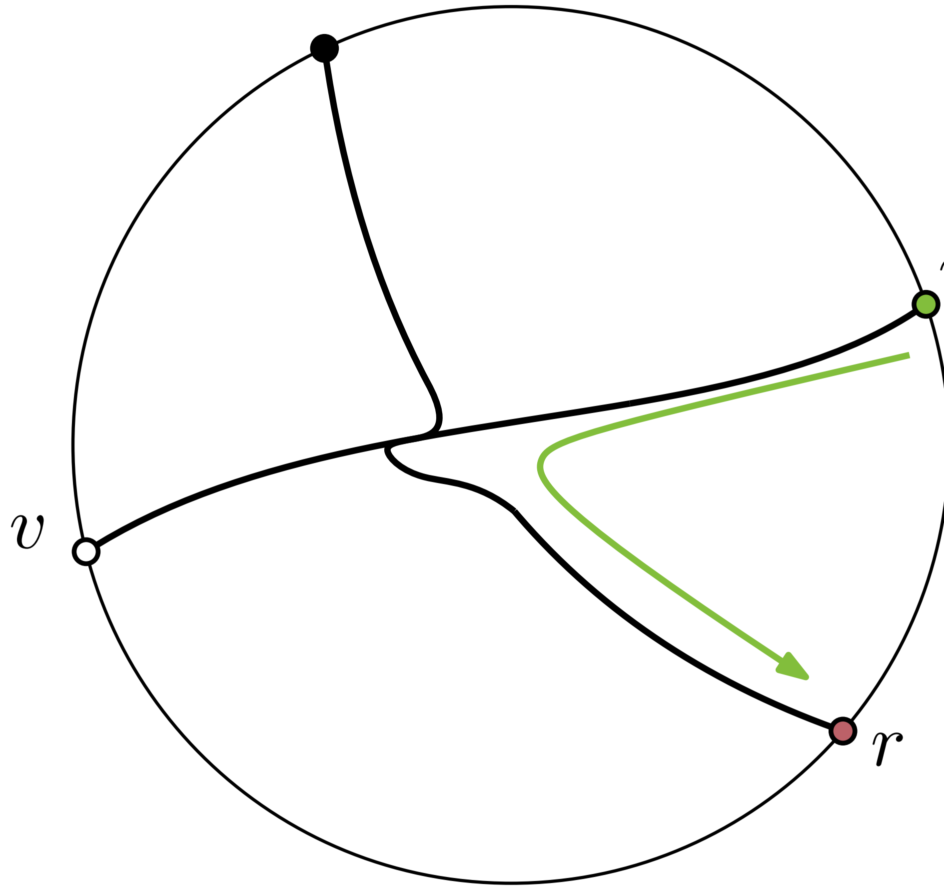
Result: Strict outer-confluent graphs have cop-number 2



Using one cop, robber is “locked” between u and v

Cops and Robbers meeting SOC Graphs

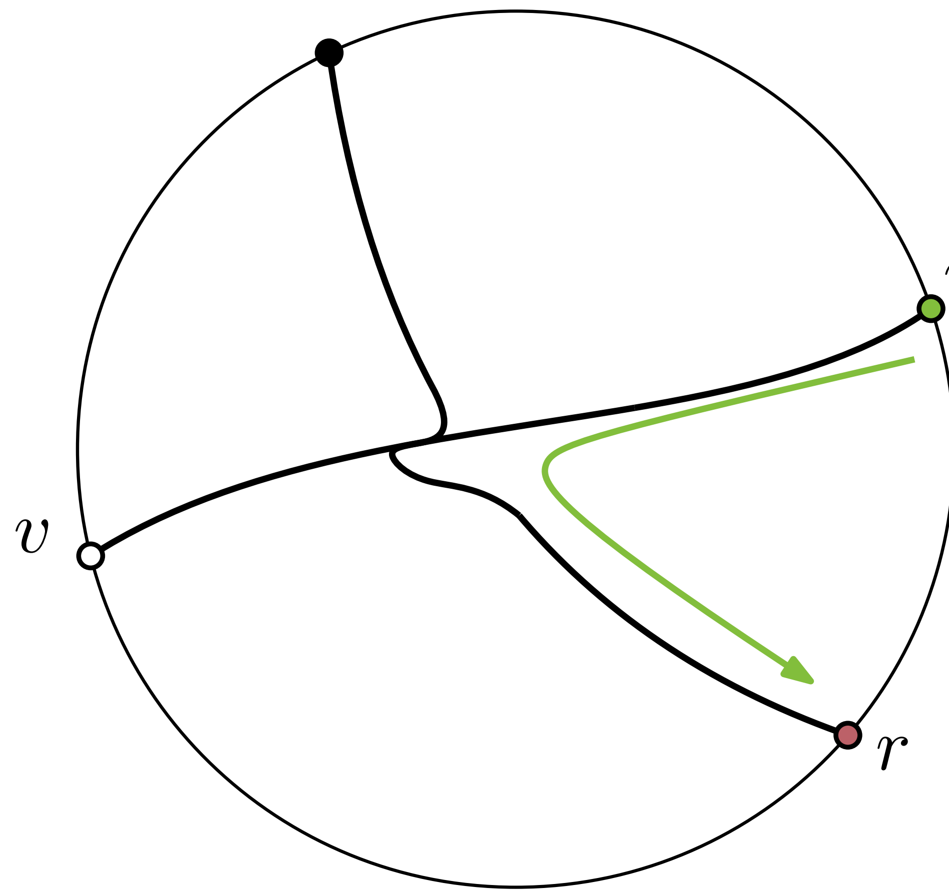
Result: Strict outer-confluent graphs have cop-number 2



Using one cop, robber is “locked” between u and v

Cops and Robbers meeting SOC Graphs

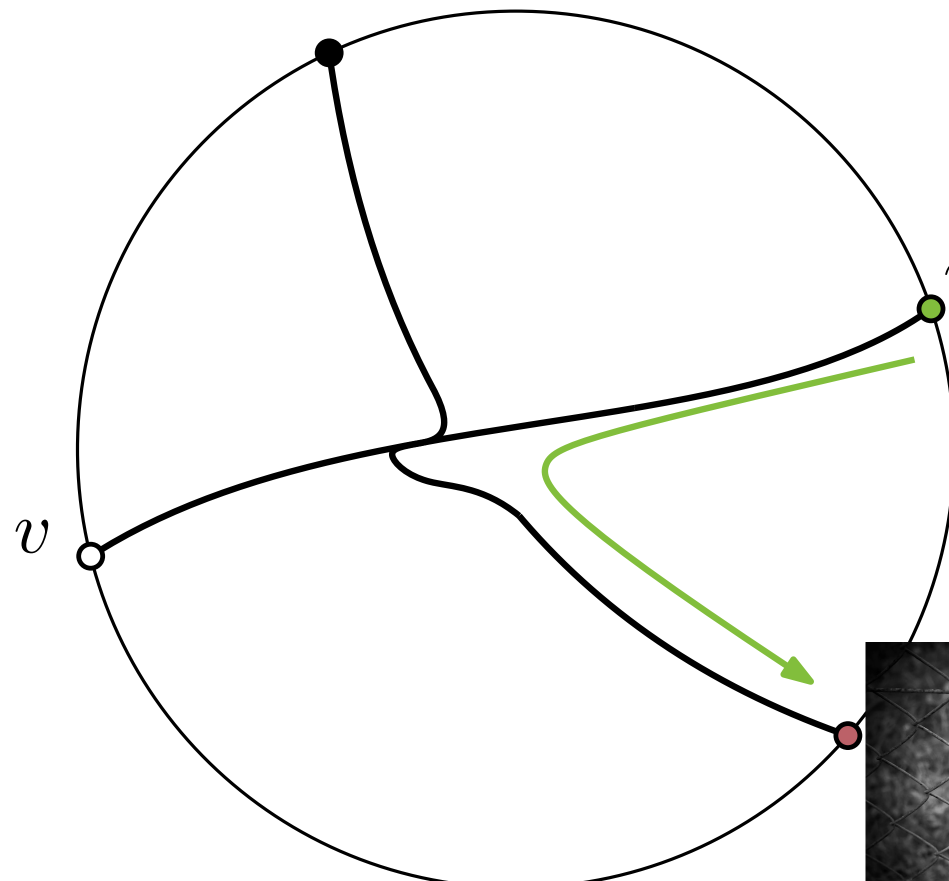
Result: Strict outer-confluent graphs have cop-number 2



Using one cop, robber is “locked” between u and v

Cops and Robbers meeting SOC Graphs

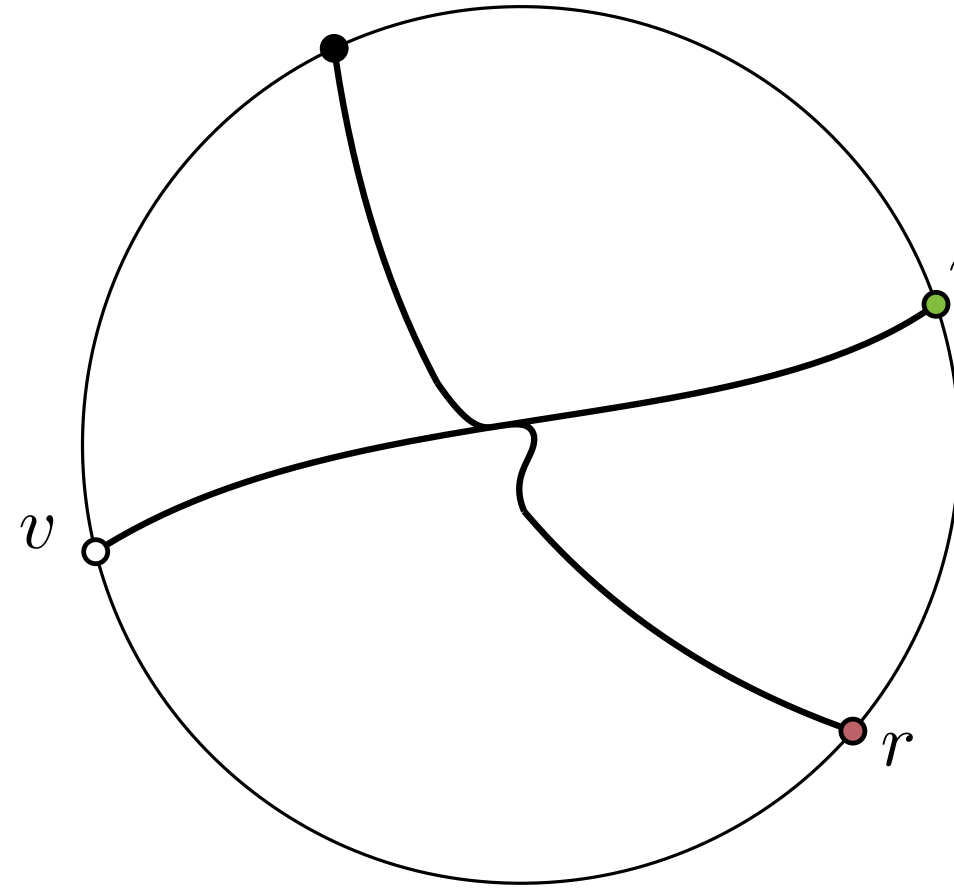
Result: Strict outer-confluent graphs have cop-number 2



Using one cop, robber is "locked" between u and v

Cops and Robbers meeting SOC Graphs

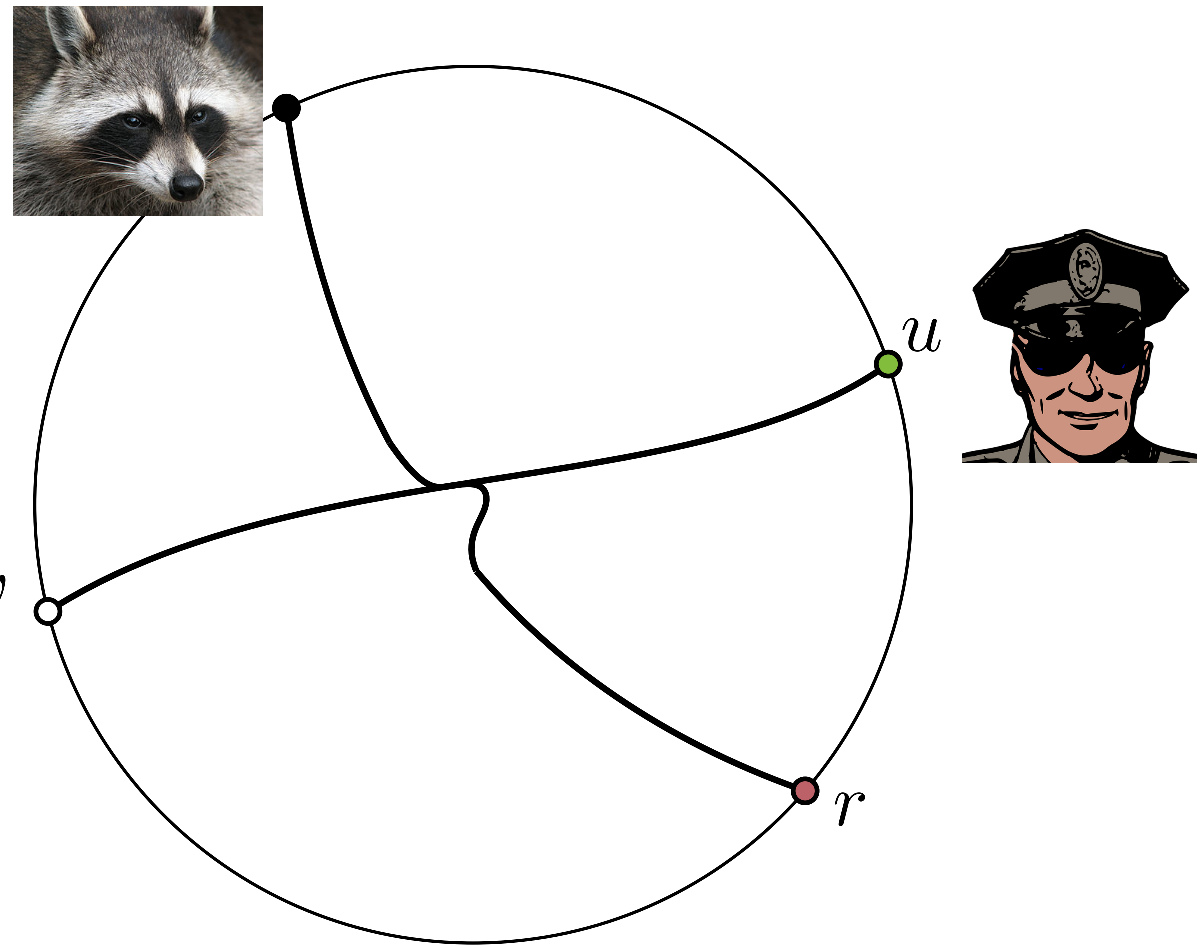
Result: Strict outer-confluent graphs have cop-number 2



Using one cop, robber is “locked” between u and v

Cops and Robbers meeting SOC Graphs

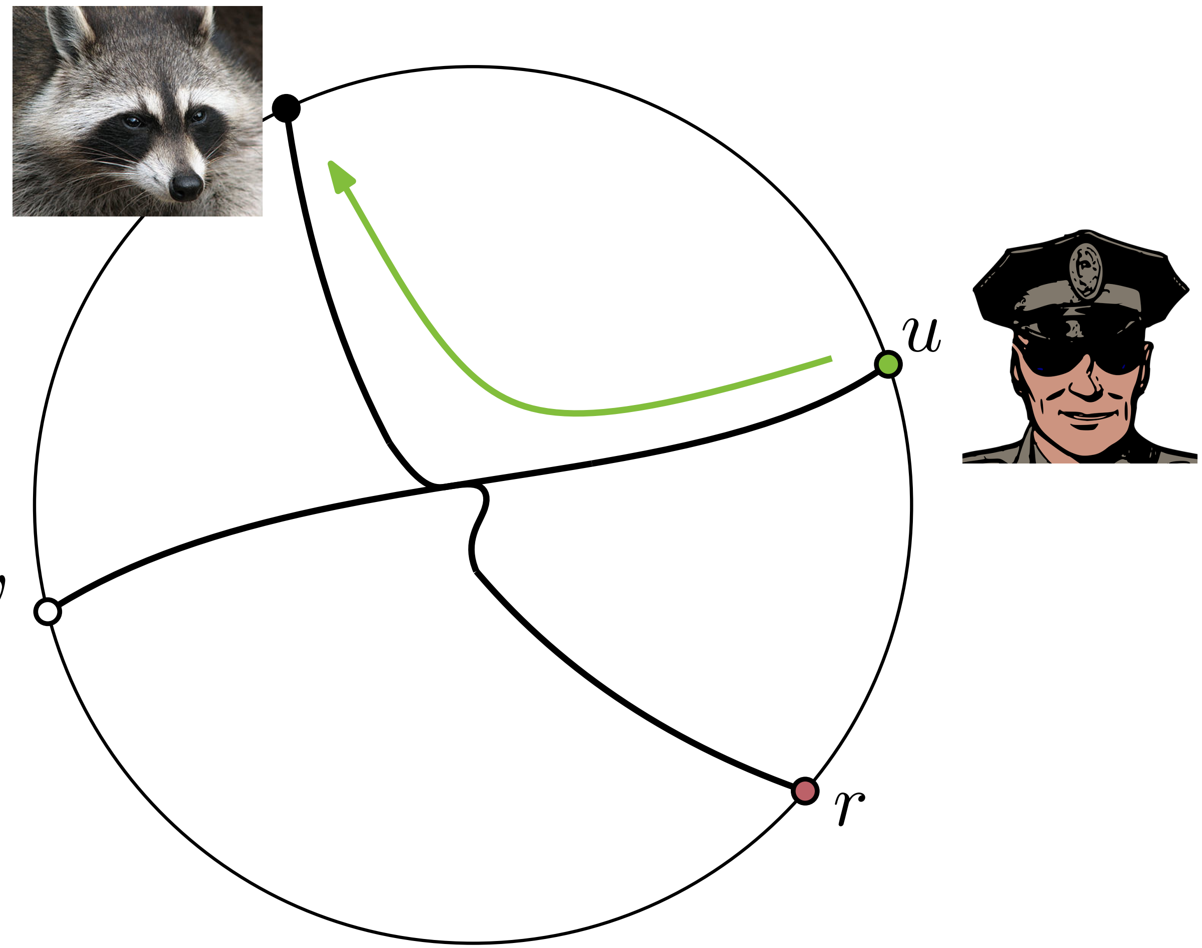
Result: Strict outer-confluent graphs have cop-number 2



Using one cop, robber is “locked” between u and v

Cops and Robbers meeting SOC Graphs

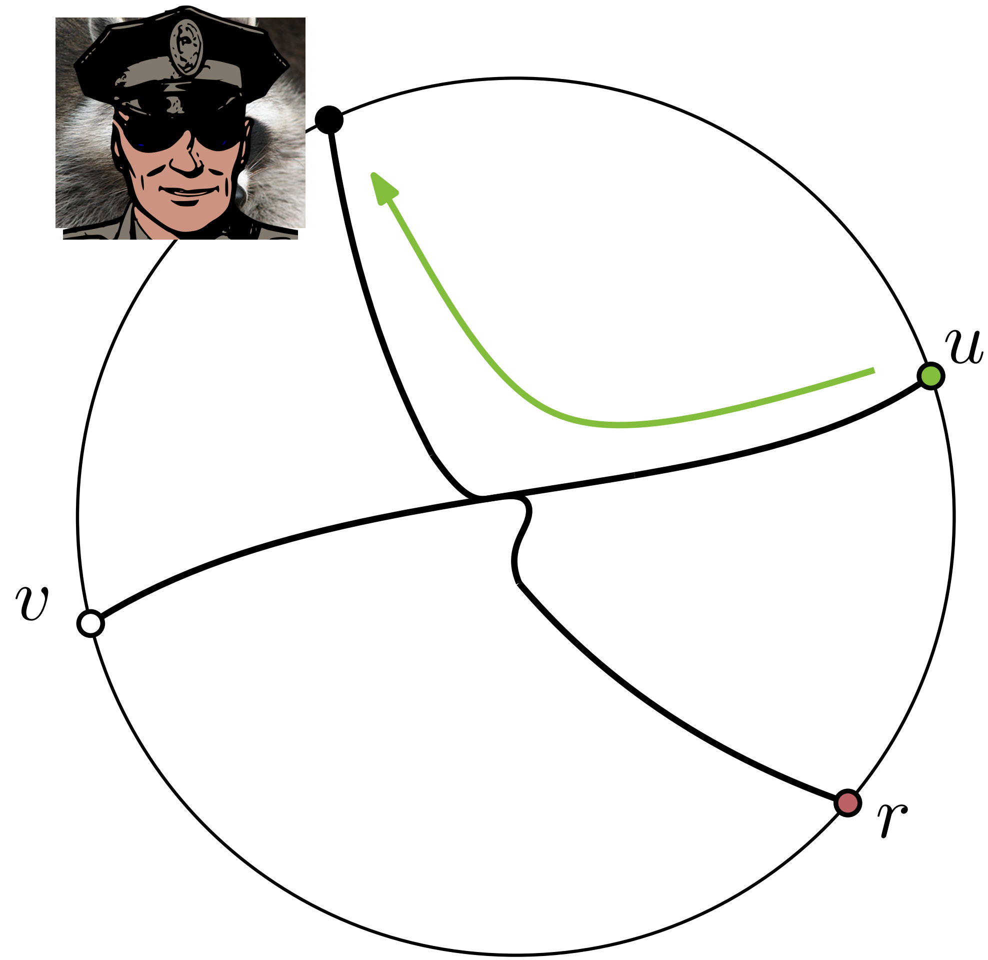
Result: Strict outer-confluent graphs have cop-number 2



Using one cop, robber is “locked” between u and v

Cops and Robbers meeting SOC Graphs

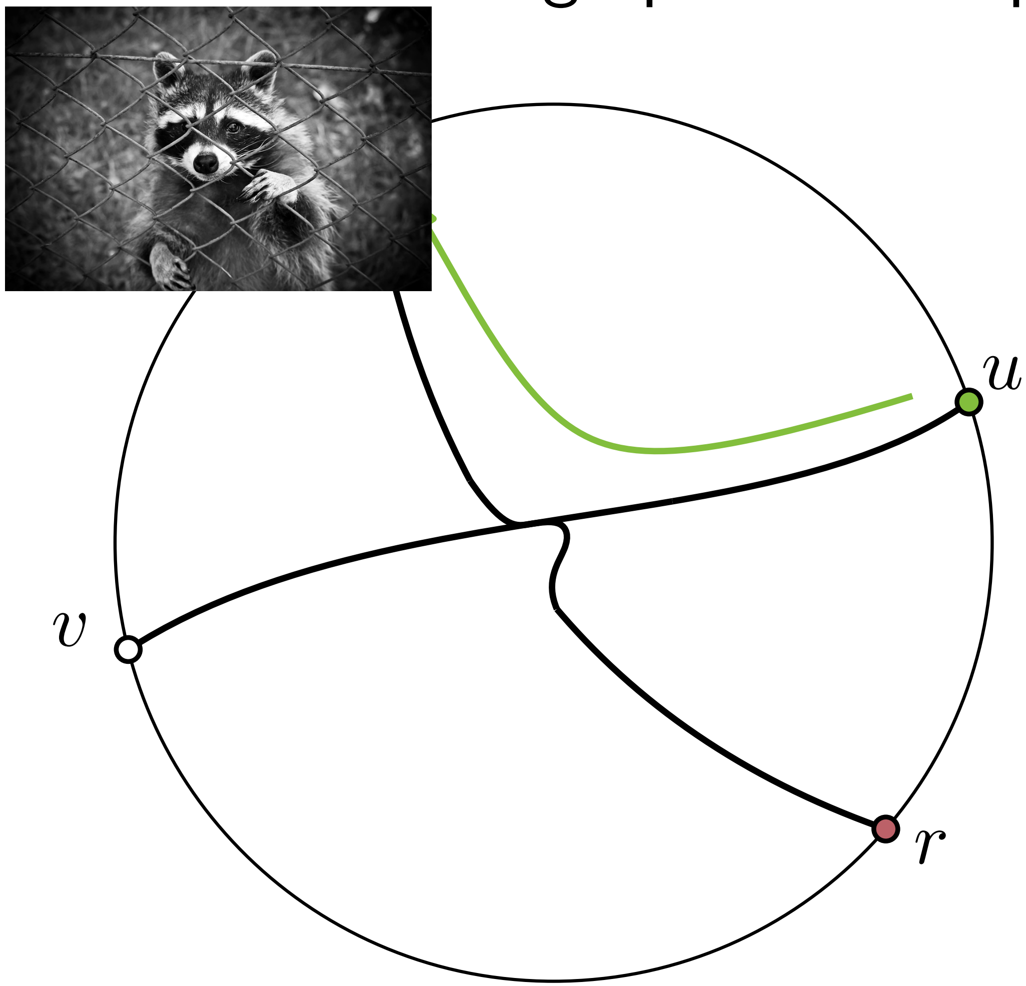
Result: Strict outer-confluent graphs have cop-number 2



Using one cop, robber is “locked” between u and v

Cops and Robbers meeting SOC Graphs

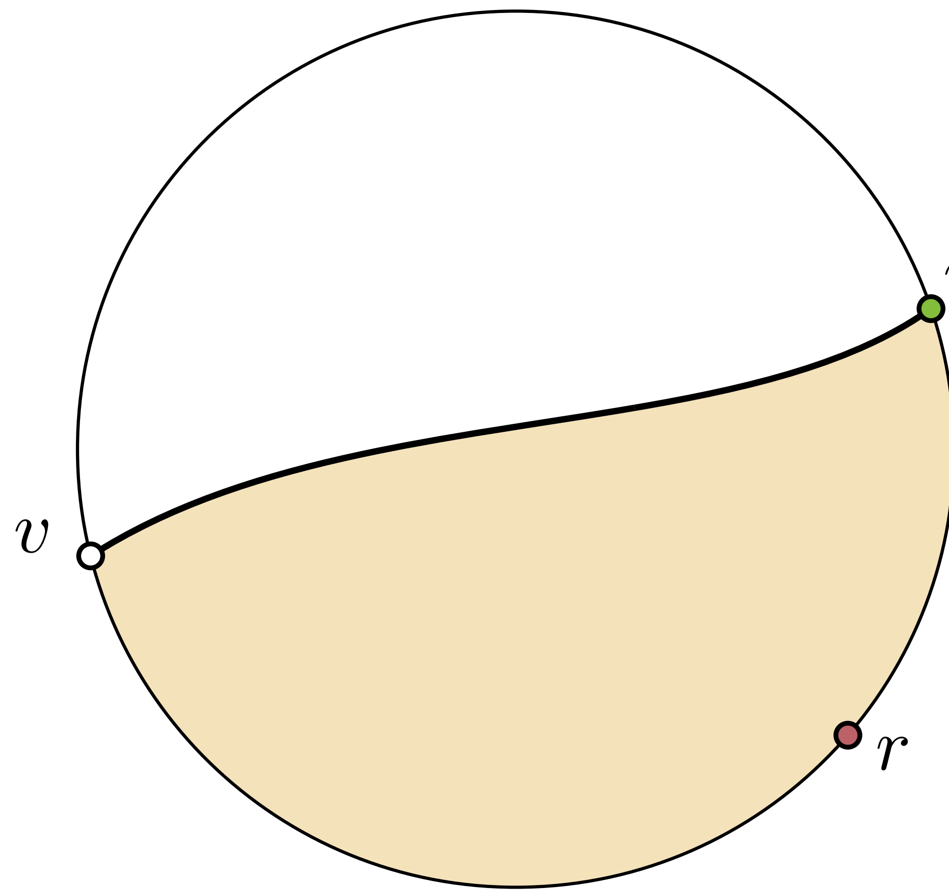
Result: Strict outer-confluent graphs have cop-number 2



Using one cop, robber is “locked” between u and v

Cops and Robbers meeting SOC Graphs

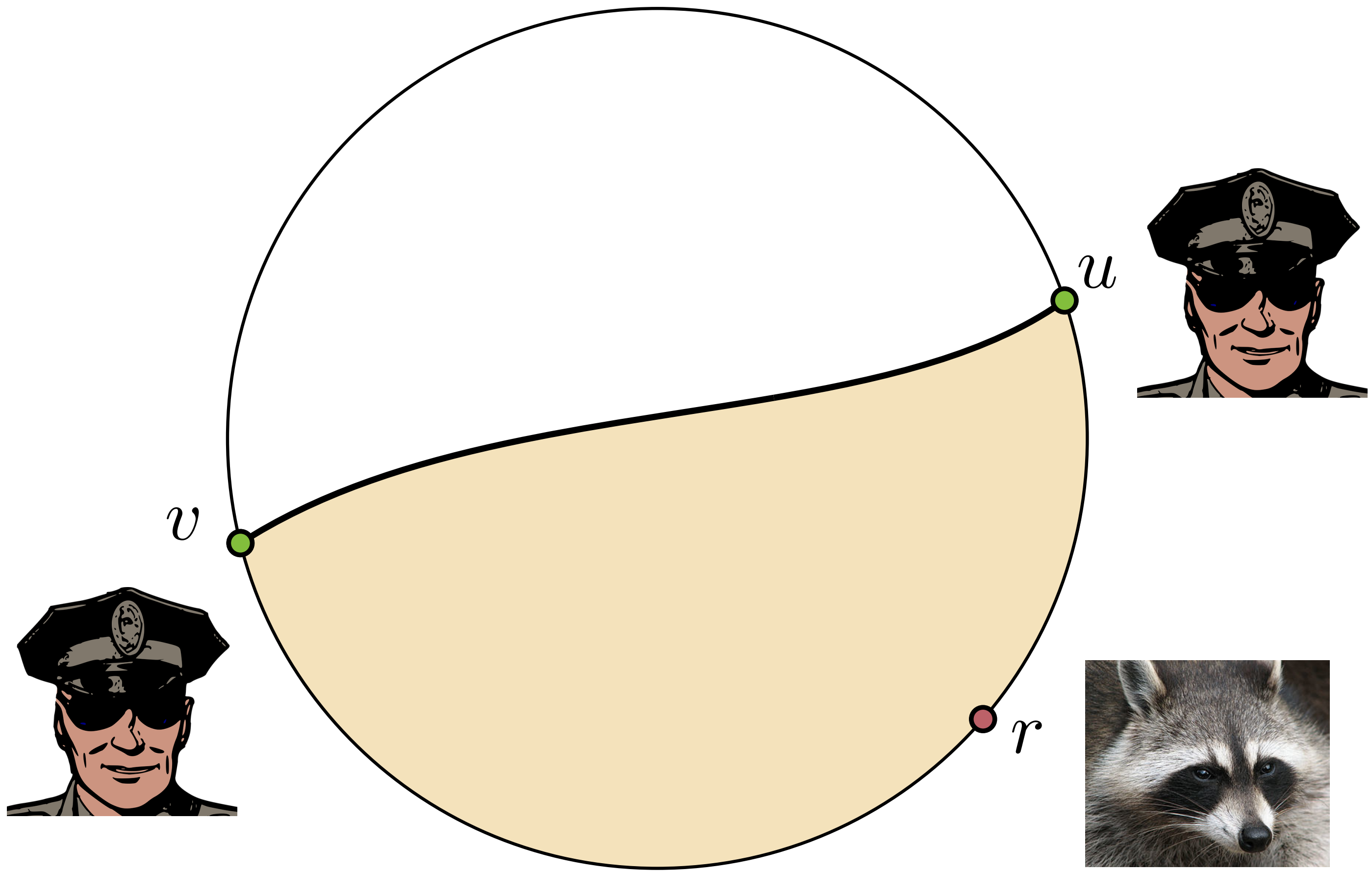
Result: Strict outer-confluent graphs have cop-number 2



Using one cop, robber is “locked” between u and v

Cops and Robbers meeting SOC Graphs

Result: Strict outer-confluent graphs have cop-number 2

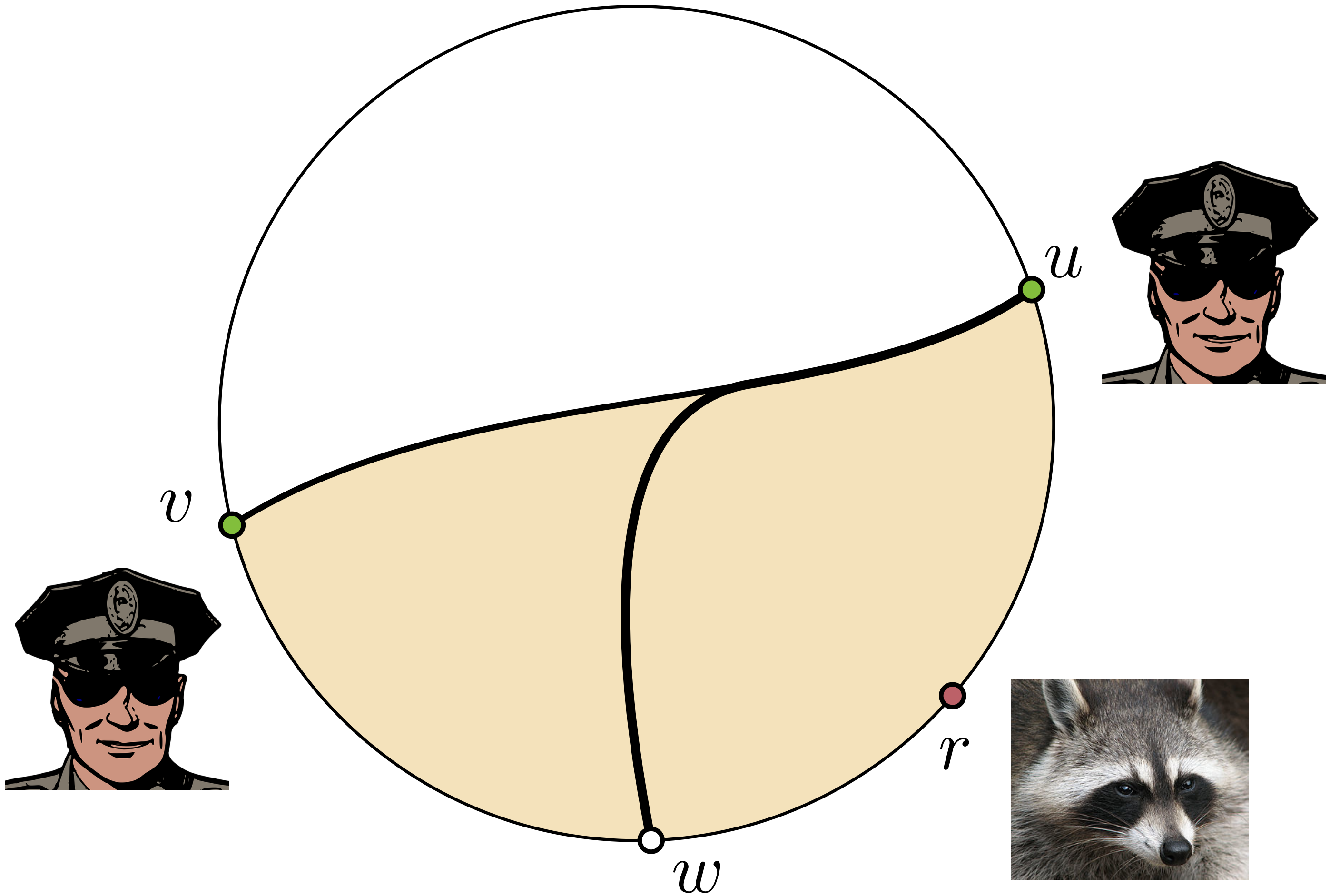


Using one cop, robber is “locked” between u and v

How to lock robber to smaller set of vertices?

Moving cops – Case 1

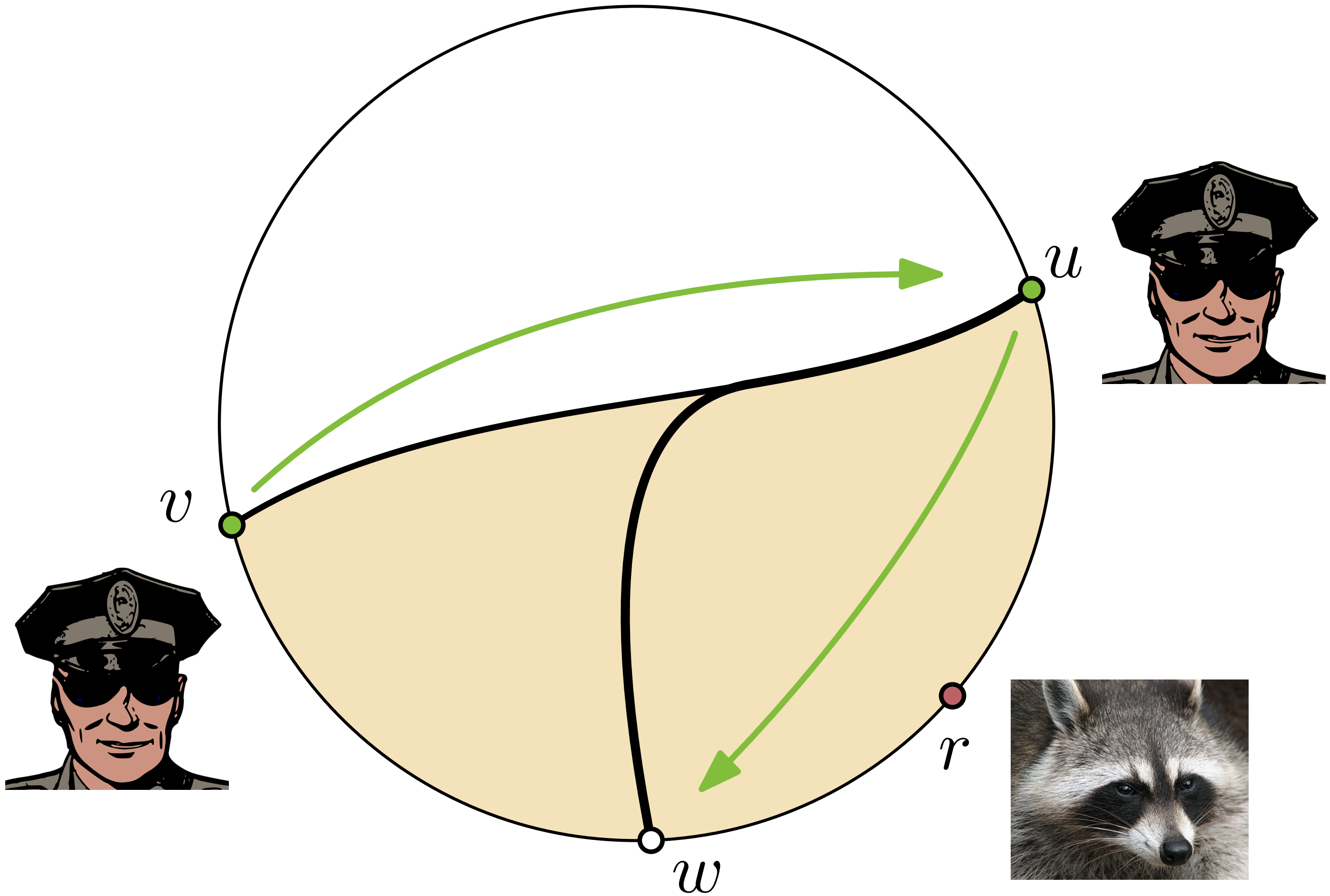
Result: Strict outer-confluent graphs have cop-number 2



$\exists uw$-path under which we can lock the robber

Moving cops – Case 1

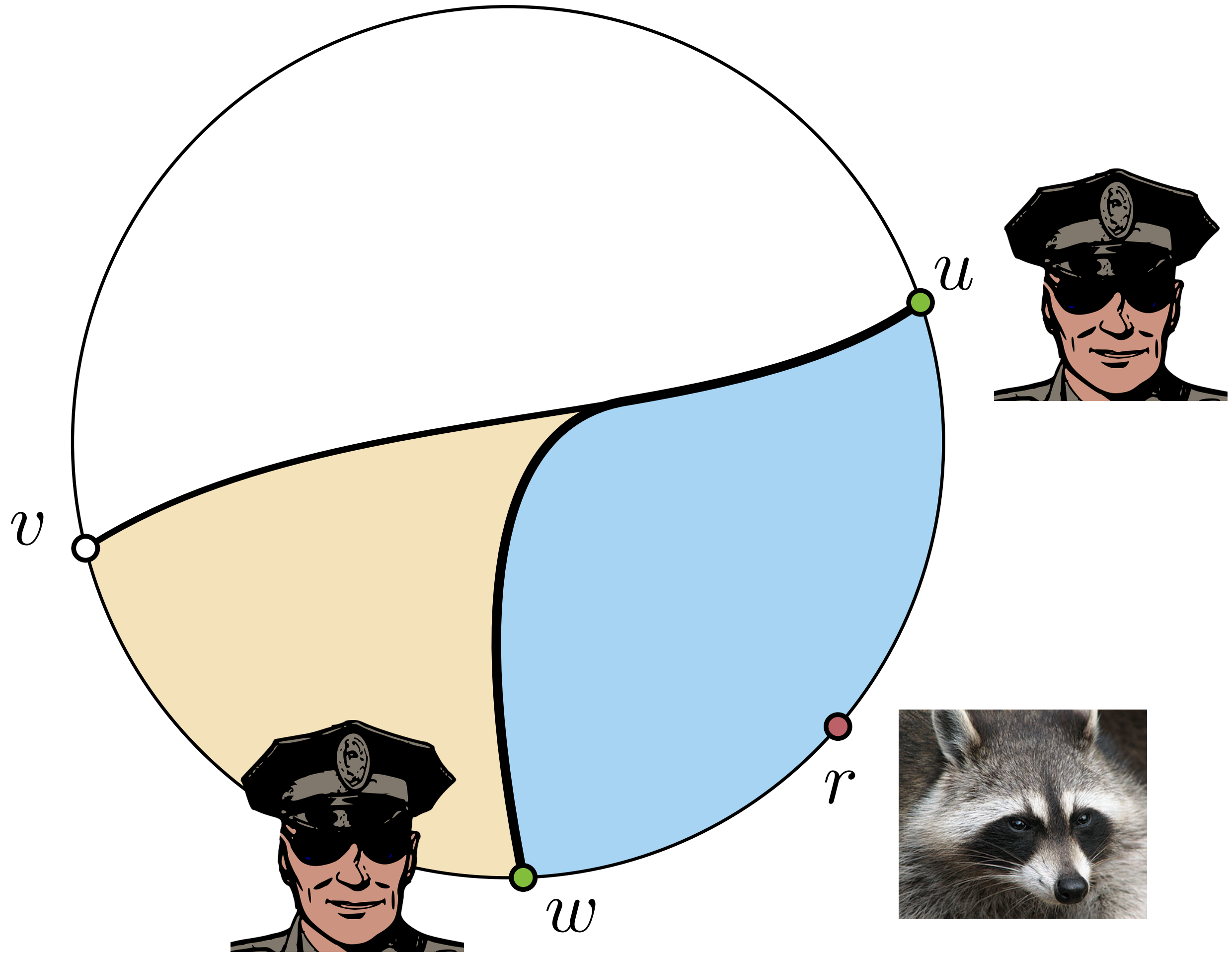
Result: Strict outer-confluent graphs have cop-number 2



$\exists uw$-path under which we can lock the robber

Moving cops – Case 1

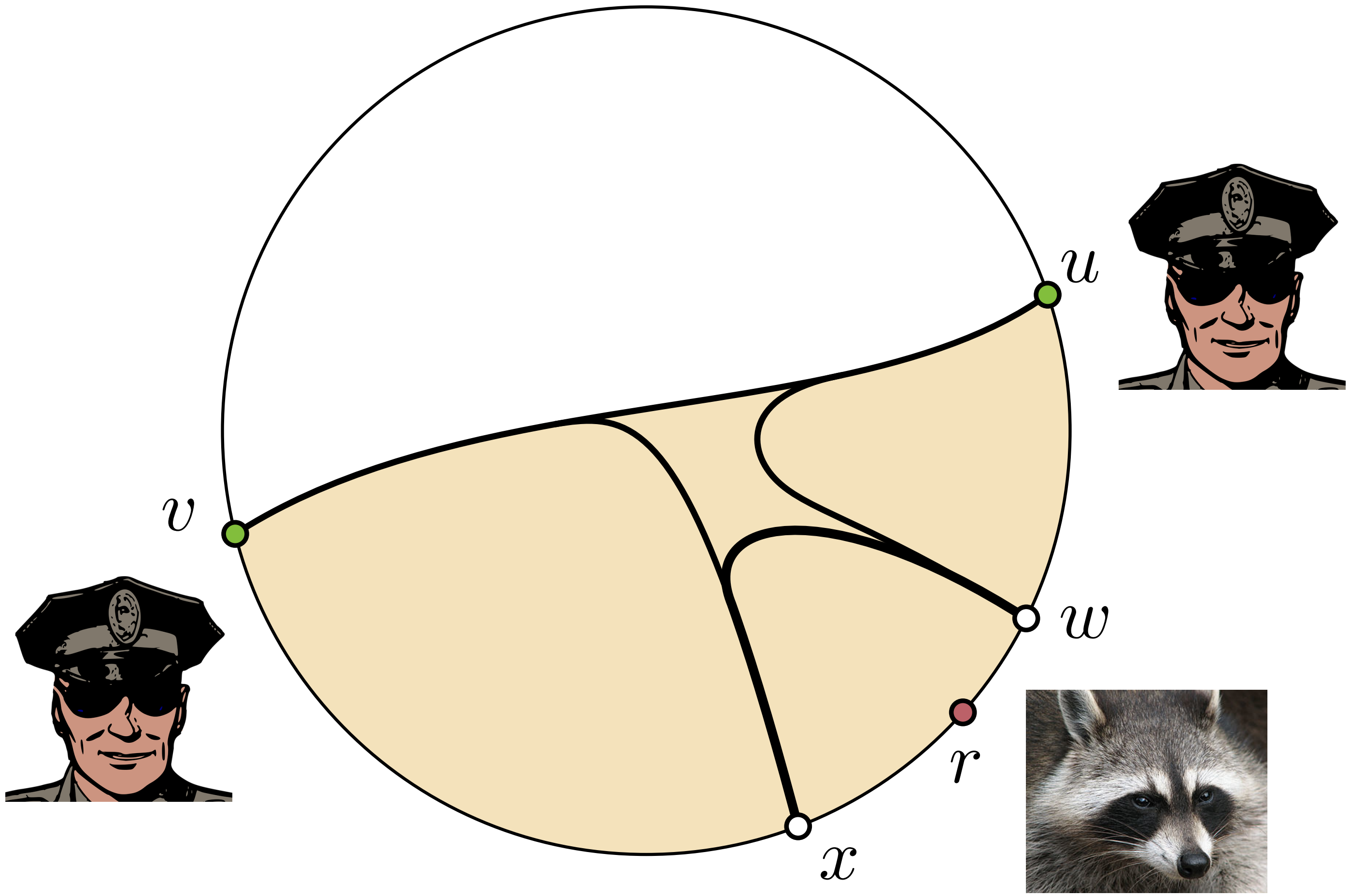
Result: Strict outer-confluent graphs have cop-number 2



$\exists uw$-path under which we can lock the robber

Moving Cops – Case 2

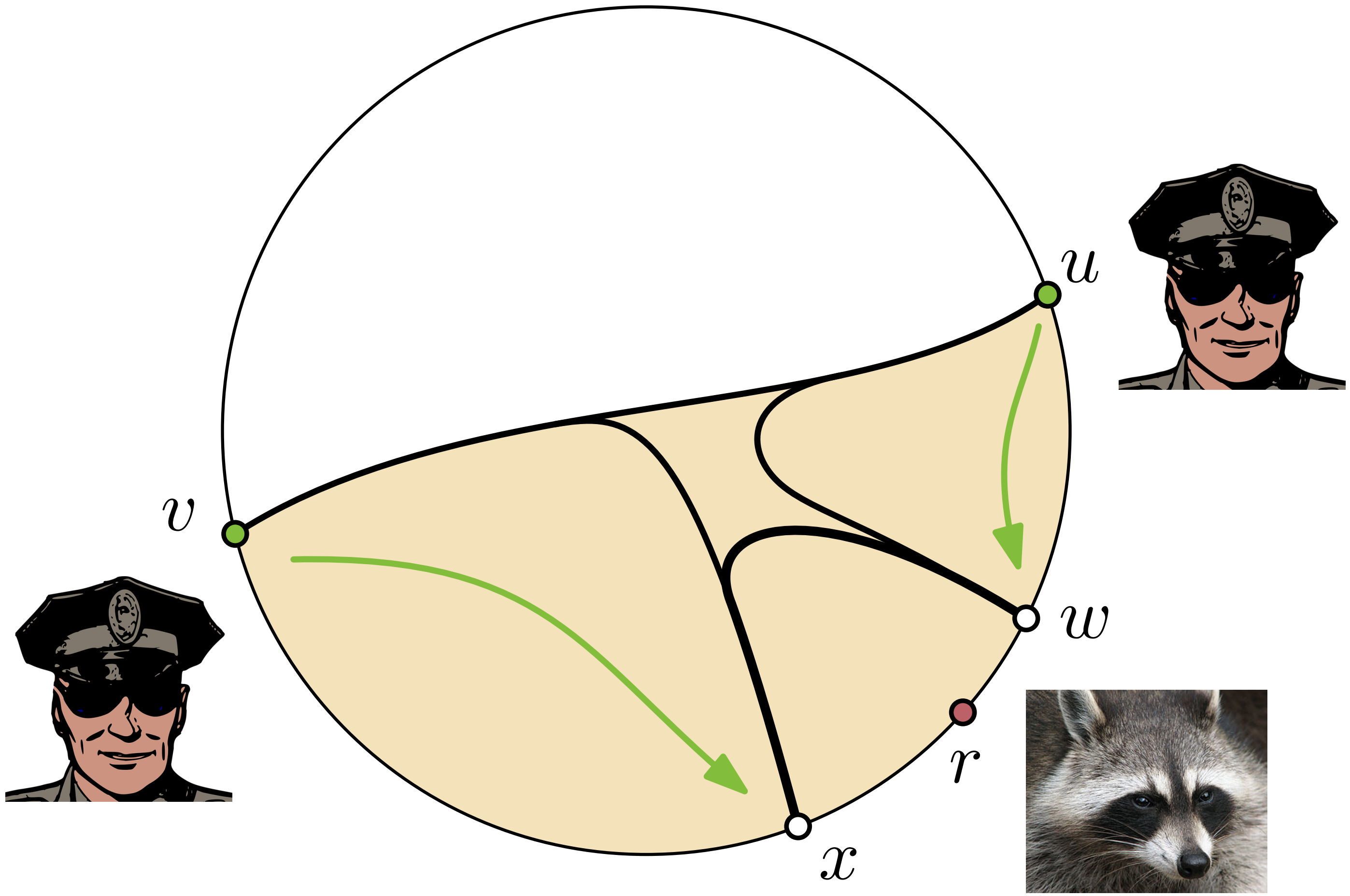
Result: Strict outer-confluent graphs have cop-number 2



$\exists xw$-path with x, w neighbors of u, v under which we can lock the robber

Moving Cops – Case 2

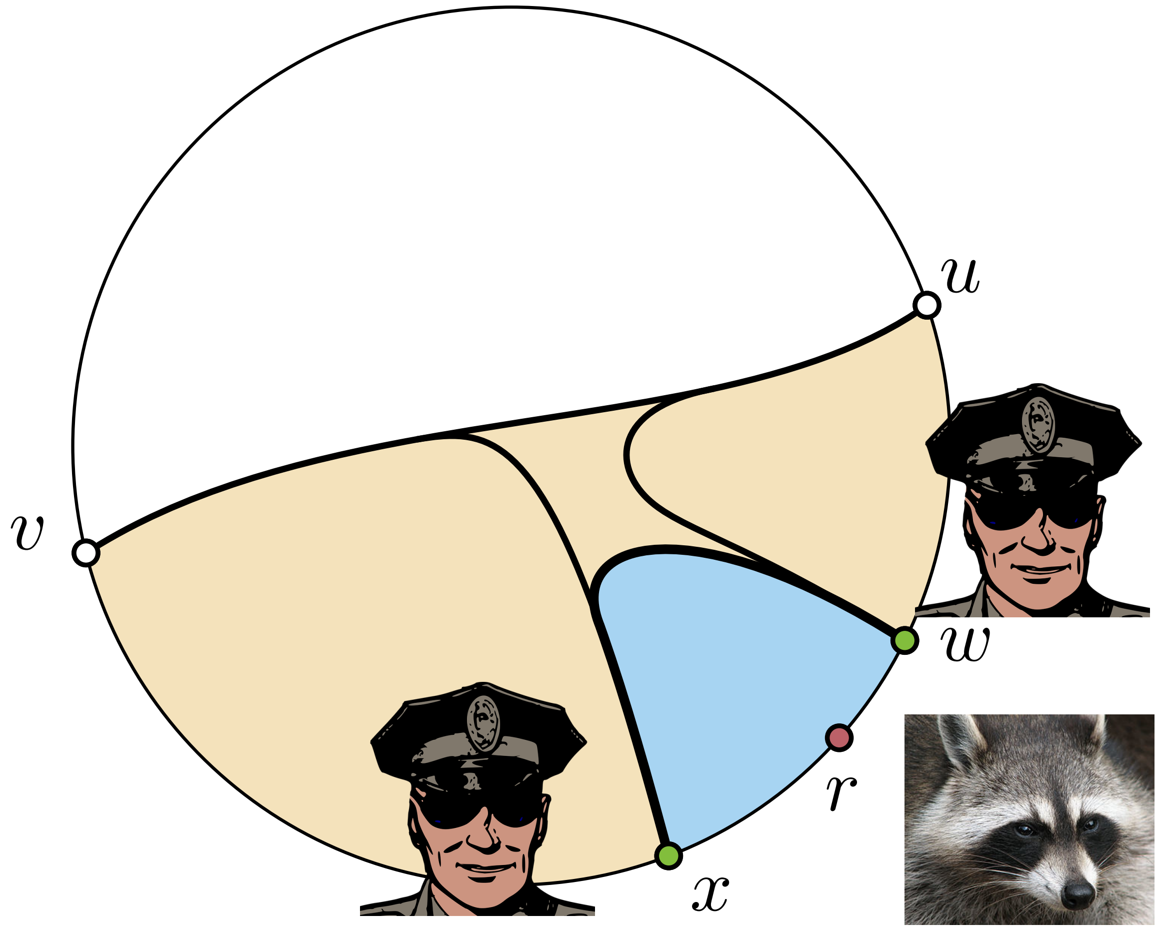
Result: Strict outer-confluent graphs have cop-number 2



$\exists xw$-path with x, w neighbors of u, v under which we can lock the robber

Moving Cops – Case 2

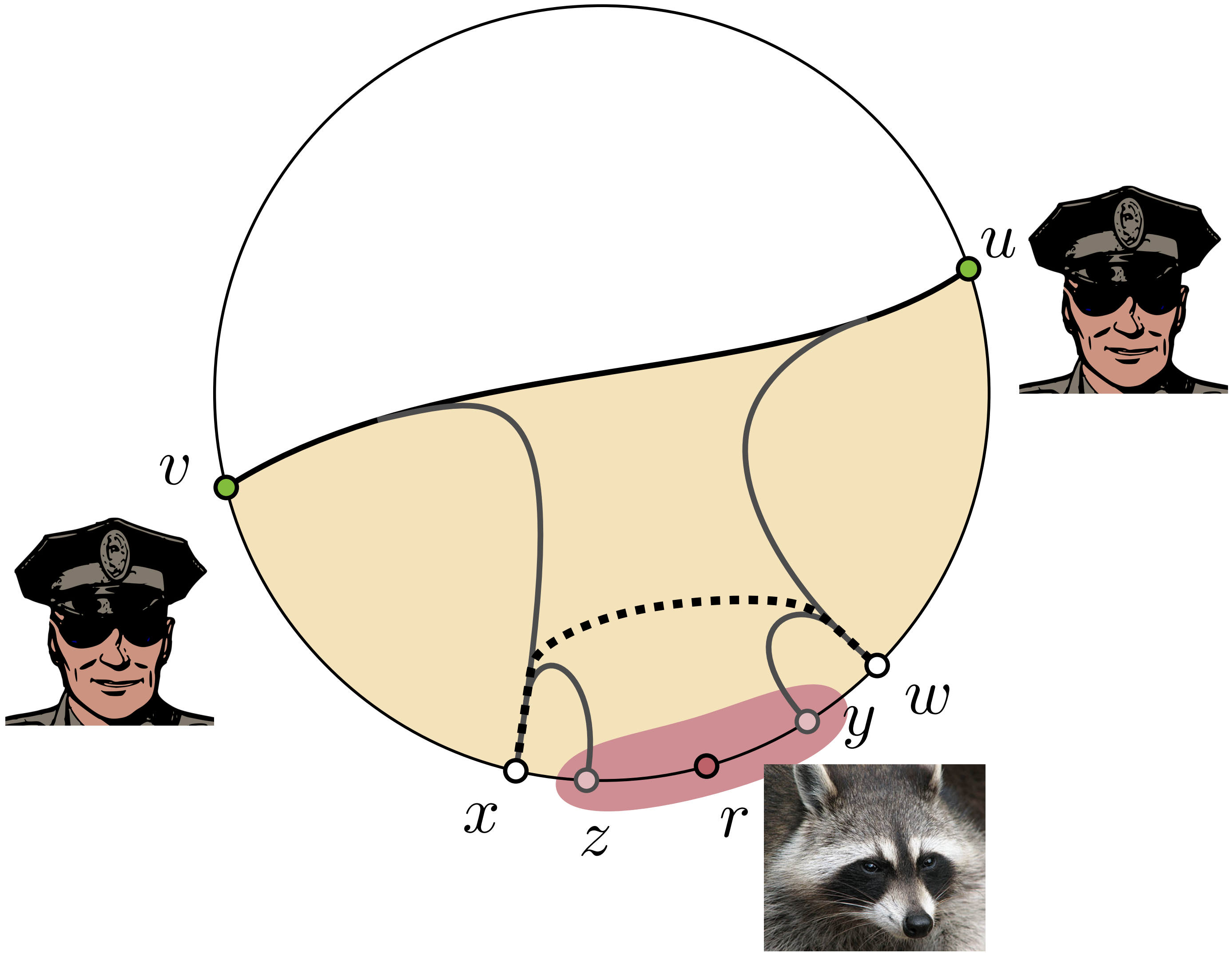
Result: Strict outer-confluent graphs have cop-number 2



$\exists xw$-path with x, w neighbors of u, v under which we can lock the robber

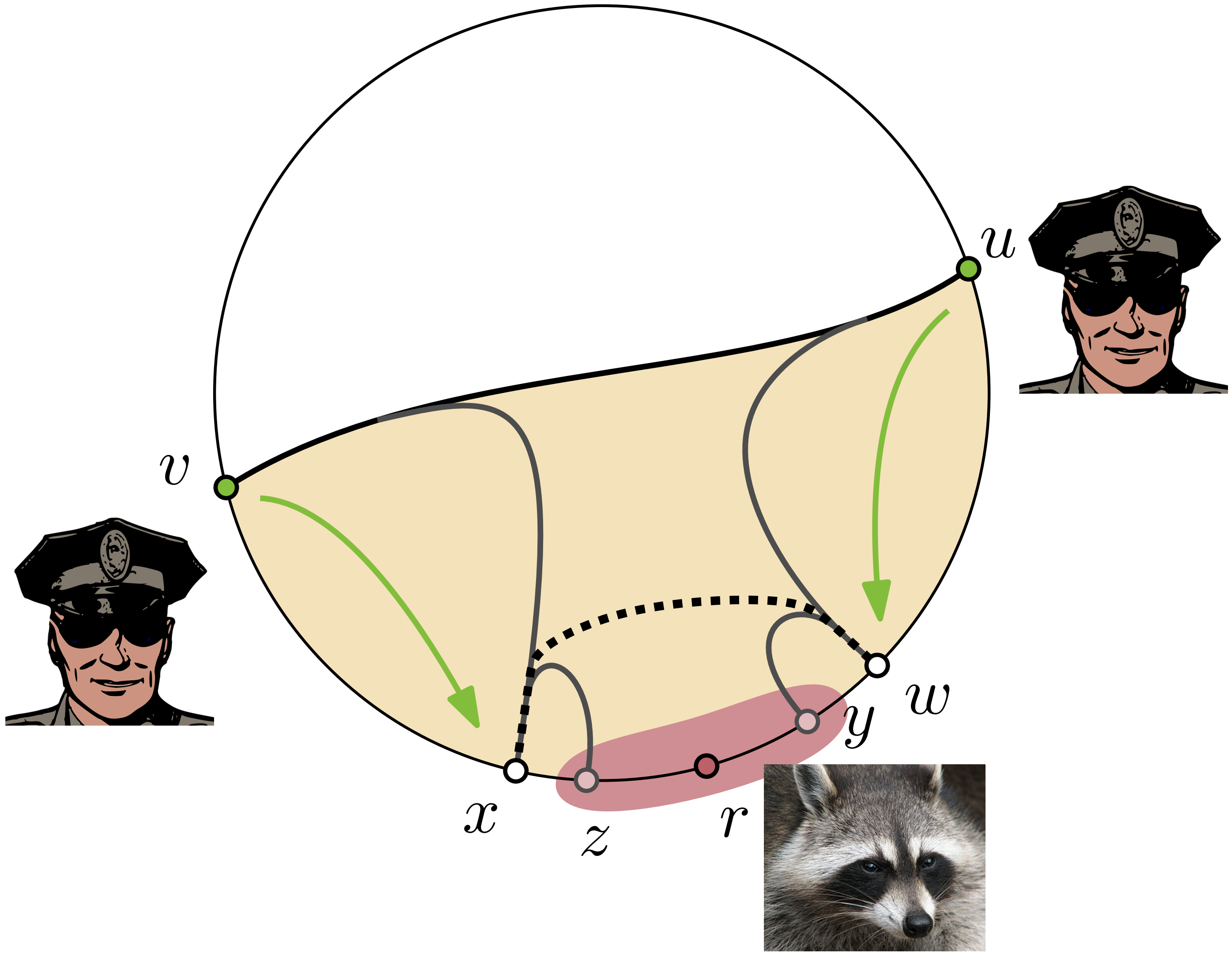
Moving Cops – Case 3

Result: Strict outer-confluent graphs have cop-number 2



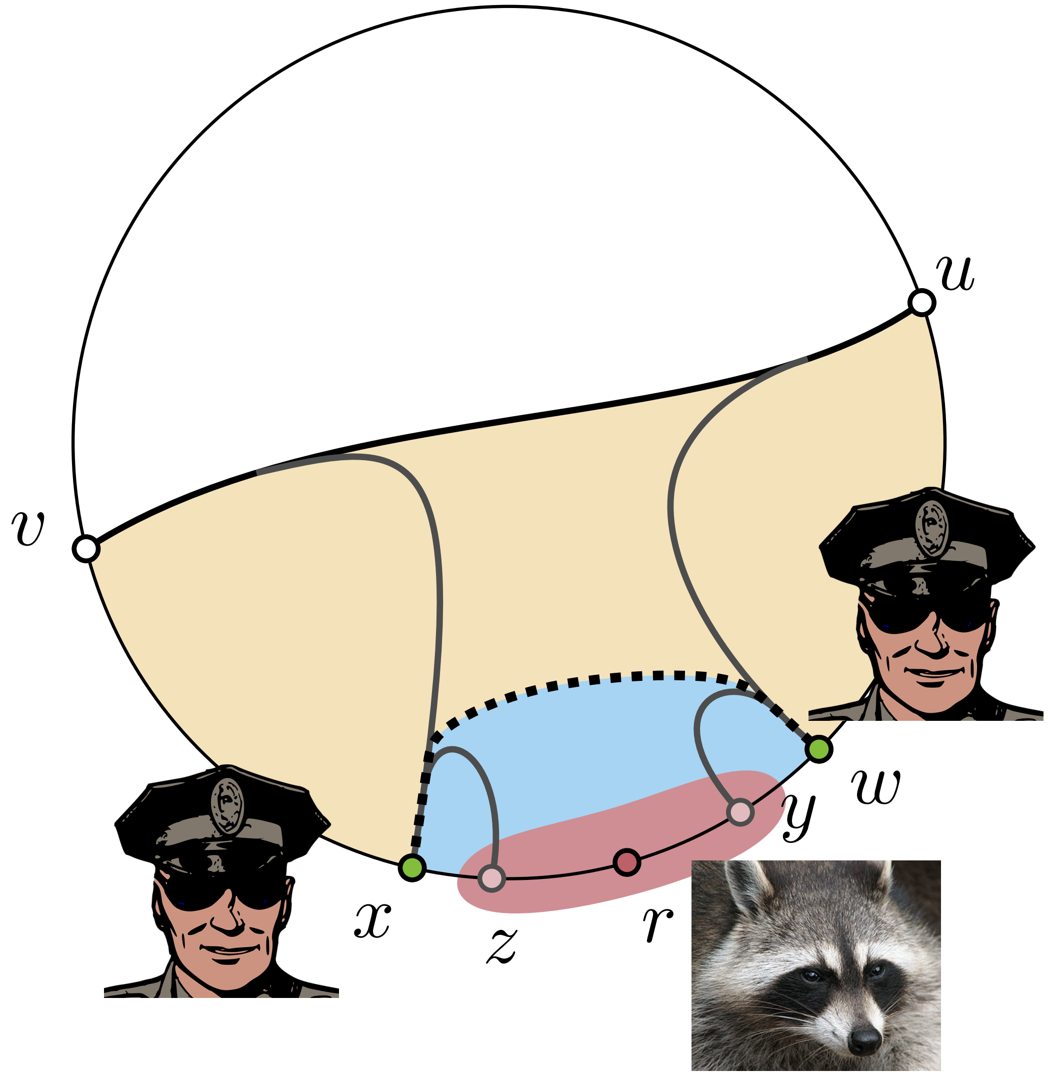
Moving Cops – Case 3

Result: Strict outer-confluent graphs have cop-number 2



Moving Cops – Case 3

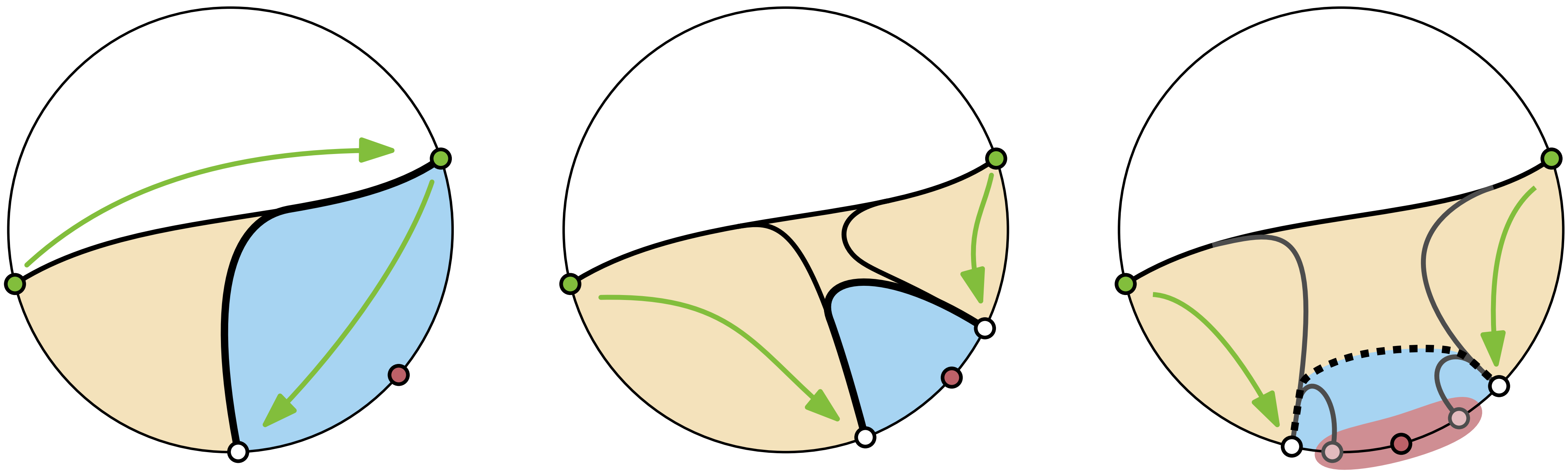
Result: Strict outer-confluent graphs have cop-number 2



If no wx-path exists, we find y, z such that we can lock a robber that is inside the red region

Catching the Robber

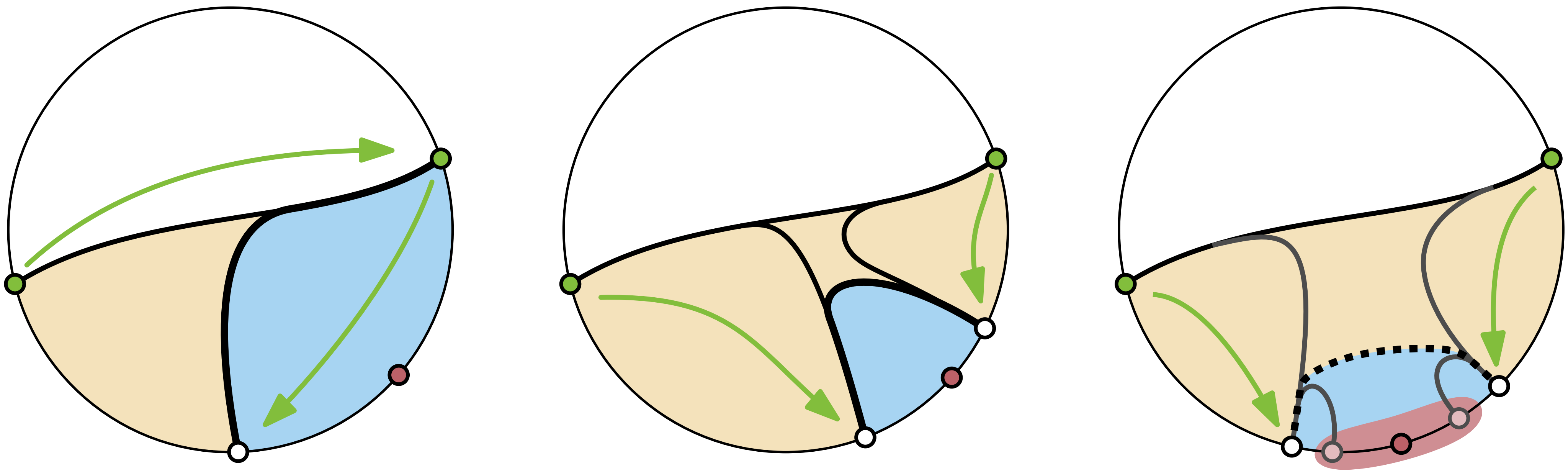
Result: Strict outer-confluent graphs have cop-number 2



Each case makes the set of nodes the robber can use smaller

Catching the Robber

Result: Strict outer-confluent graphs have cop-number 2



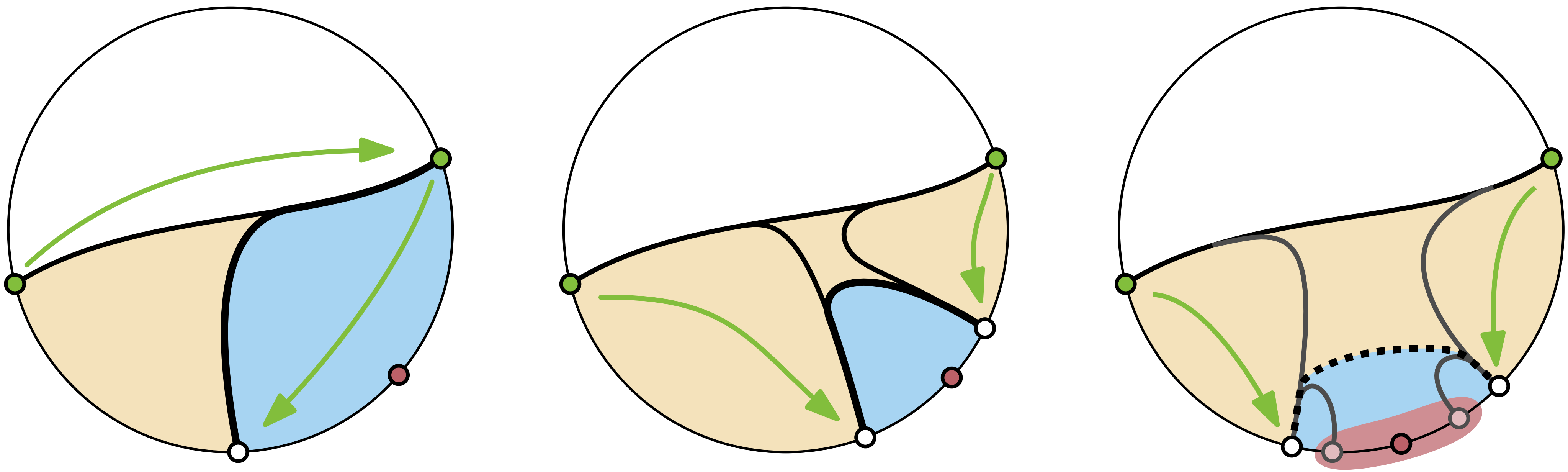
Each case makes the set of nodes the robber can use smaller

We show that one case always applies

$\Rightarrow$ Two cops suffice to catch the roober

Catching the Robber

Result: Strict outer-confluent graphs have cop-number 2



Each case makes the set of nodes the robber can use smaller

We show that one case always applies

$\Rightarrow$ Two cops suffice to catch the roober

Conclusions

One main open question

Which graphs have a strict outer-confluent drawing?

Conclusions

One main open question

Which graphs have a strict outer-confluent drawing?

The question remains open

Not even good intuition if the problem is in P or NP-hard

Conclusions

One main open question

Which graphs have a strict outer-confluent drawing?

The question remains open

Not even good intuition if the problem is in P or NP-hard

Many other questions

Do strict outer-confluent graphs have bounded cliquewidth?

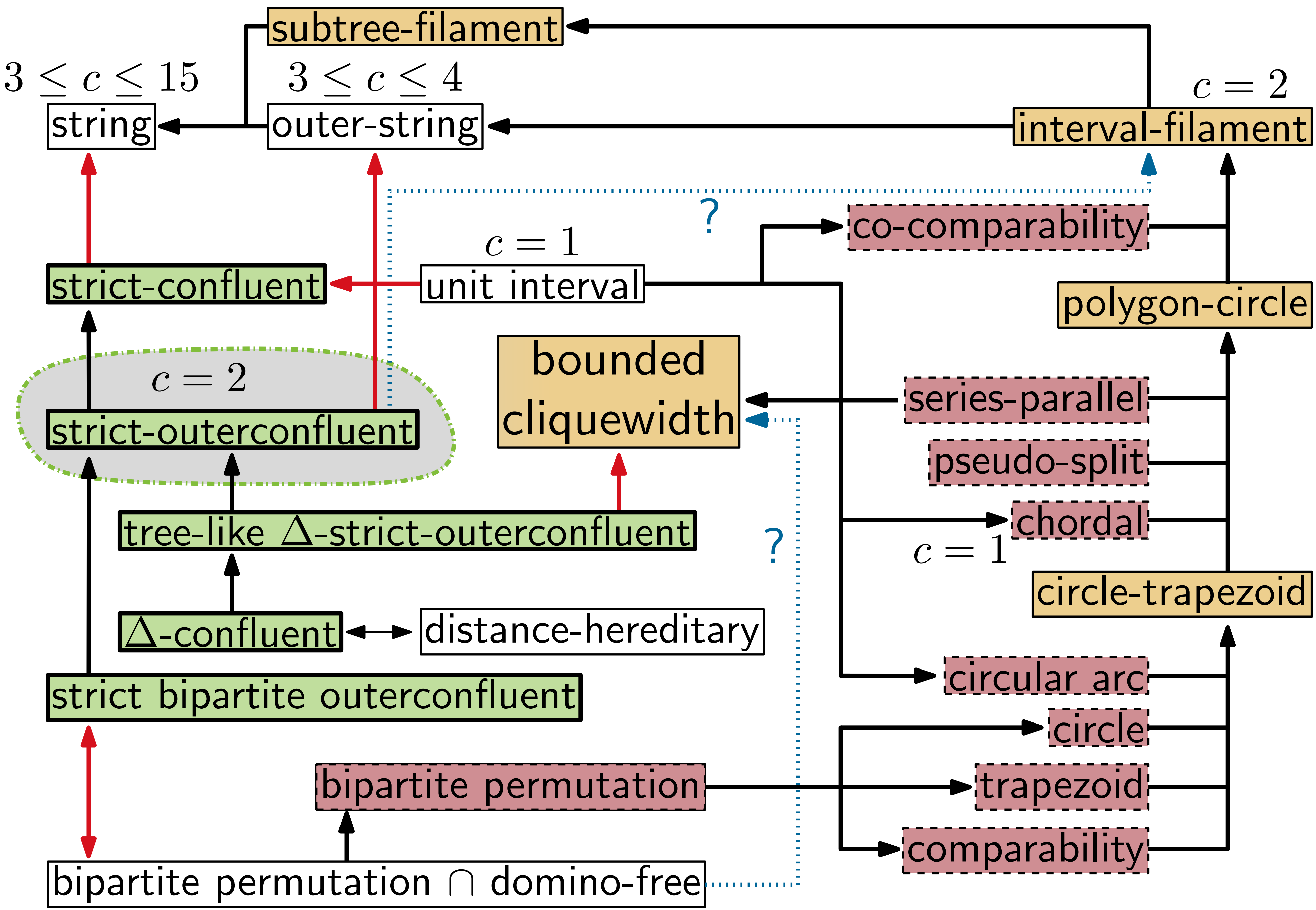
Complexity of other types of confluence?

What other graph classes admit (strict) confluent drawings?

Confluence, but with some allowed crossings? [Bach et al. 16]

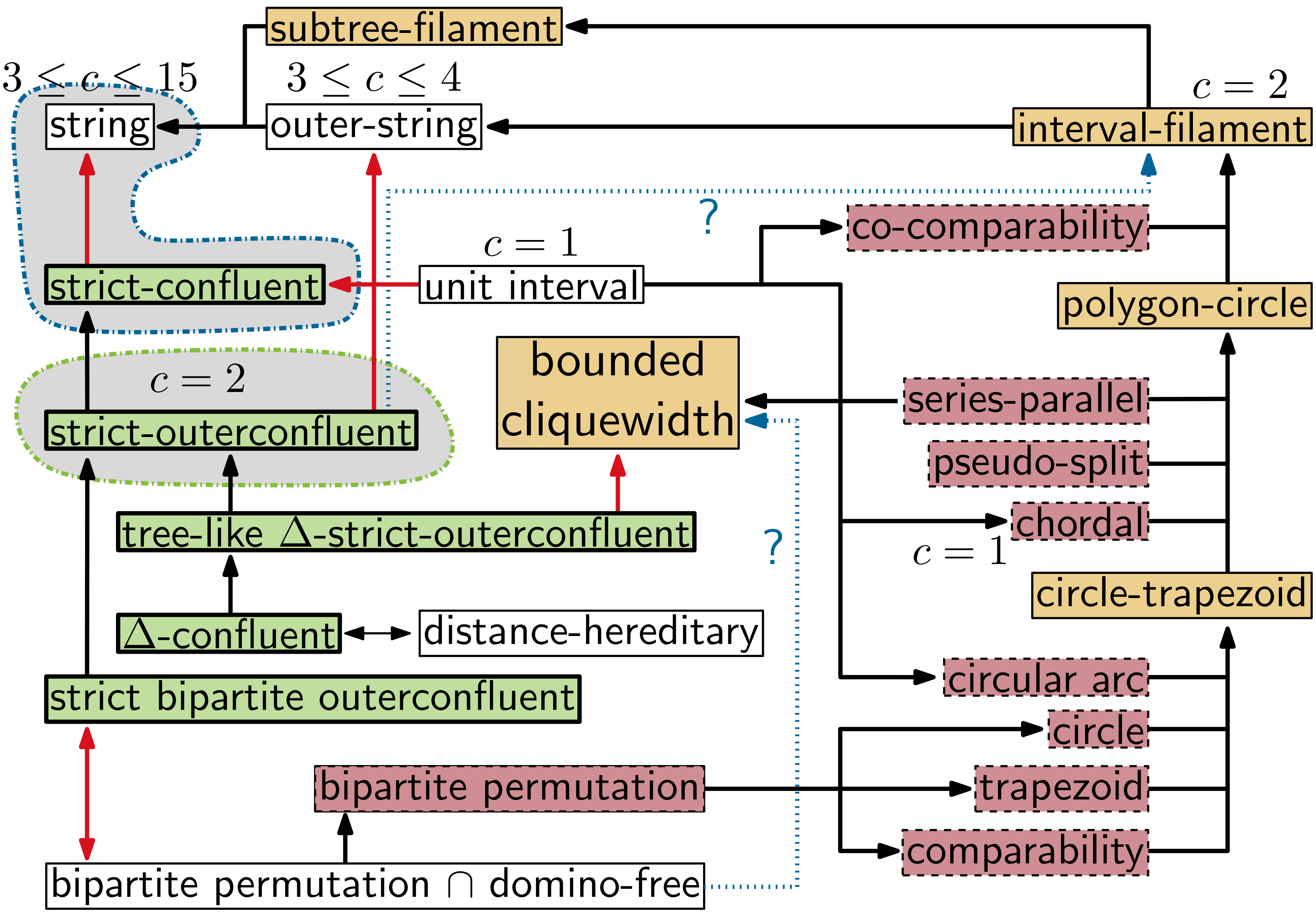
Our Results

Cop number: c



Our Results

Cop number: c



Construction of Traces

Given strict confluent drawing D

For each u define trace $t(u)$ (these will be the strings)

Each trace starts at u in D

Viewed from u, $t(u)$ stays on the left side of the paths

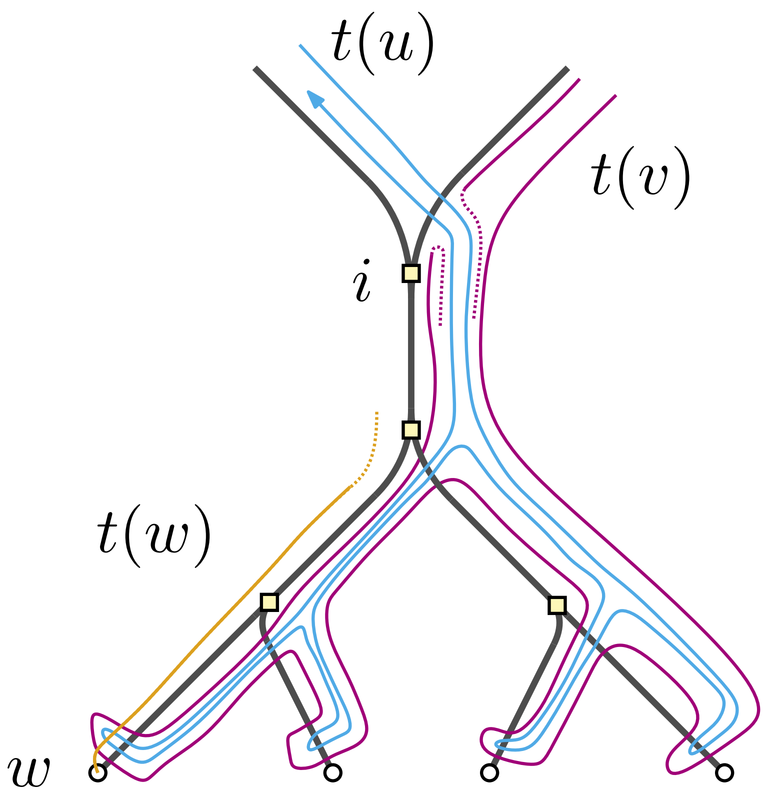
Construction of Traces

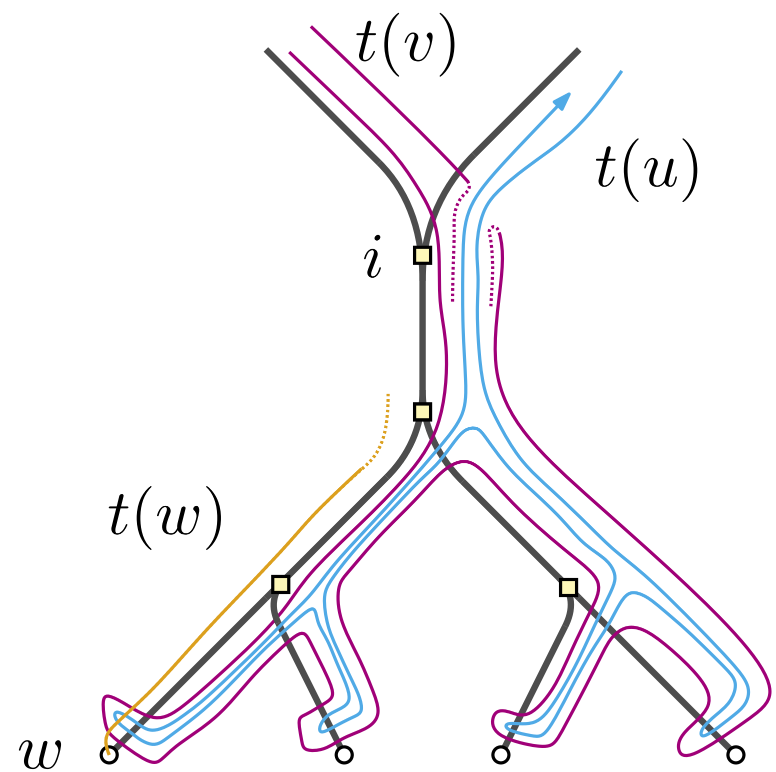
Given strict confluent drawing D

For each u define trace $t(u)$ (these will be the strings)

Each trace starts at u in D

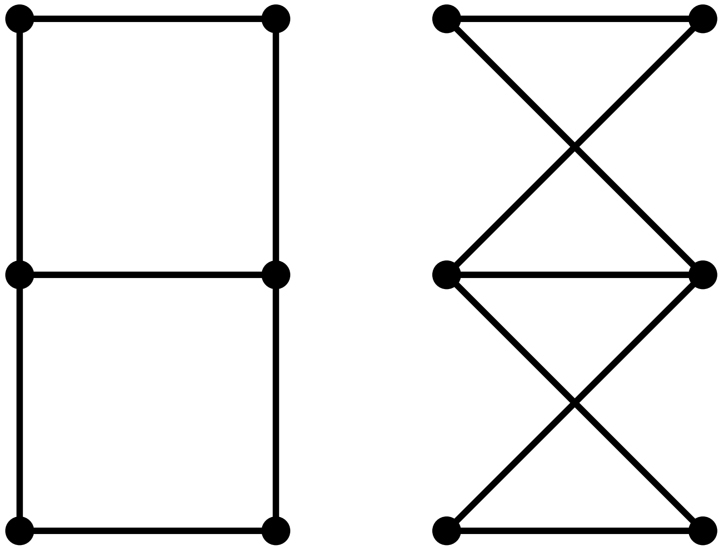
Viewed from u, $t(u)$ stays on the left side of the paths





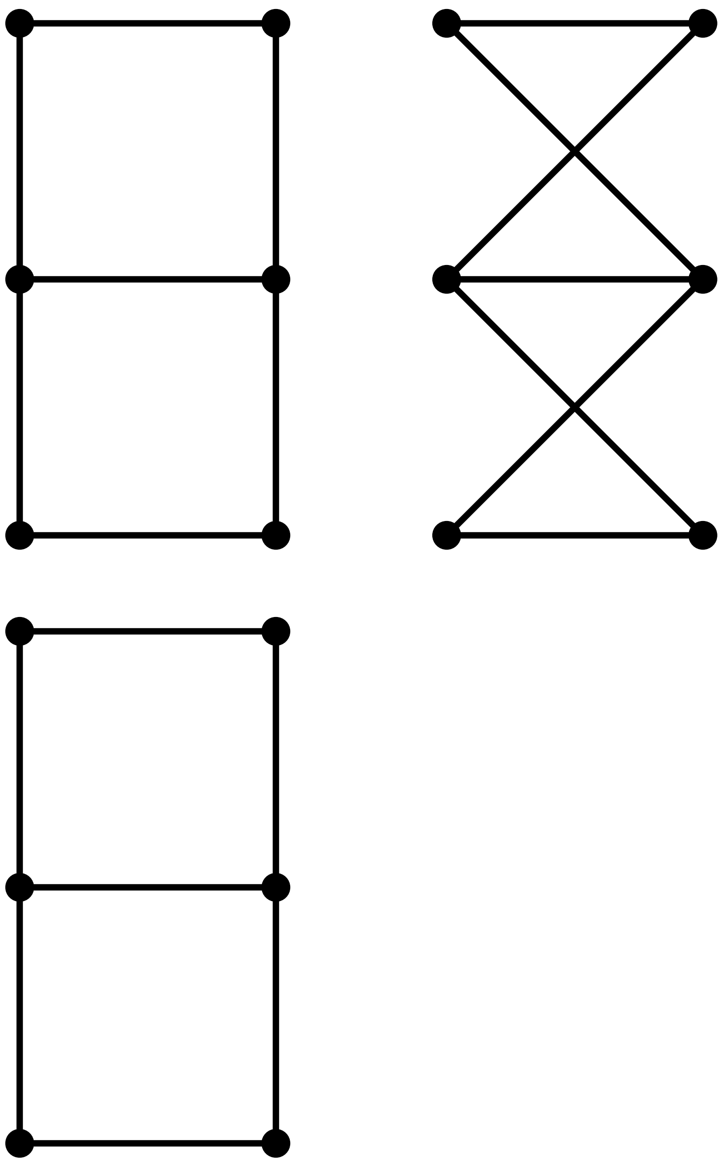
Two Problems for Strictness

Domino



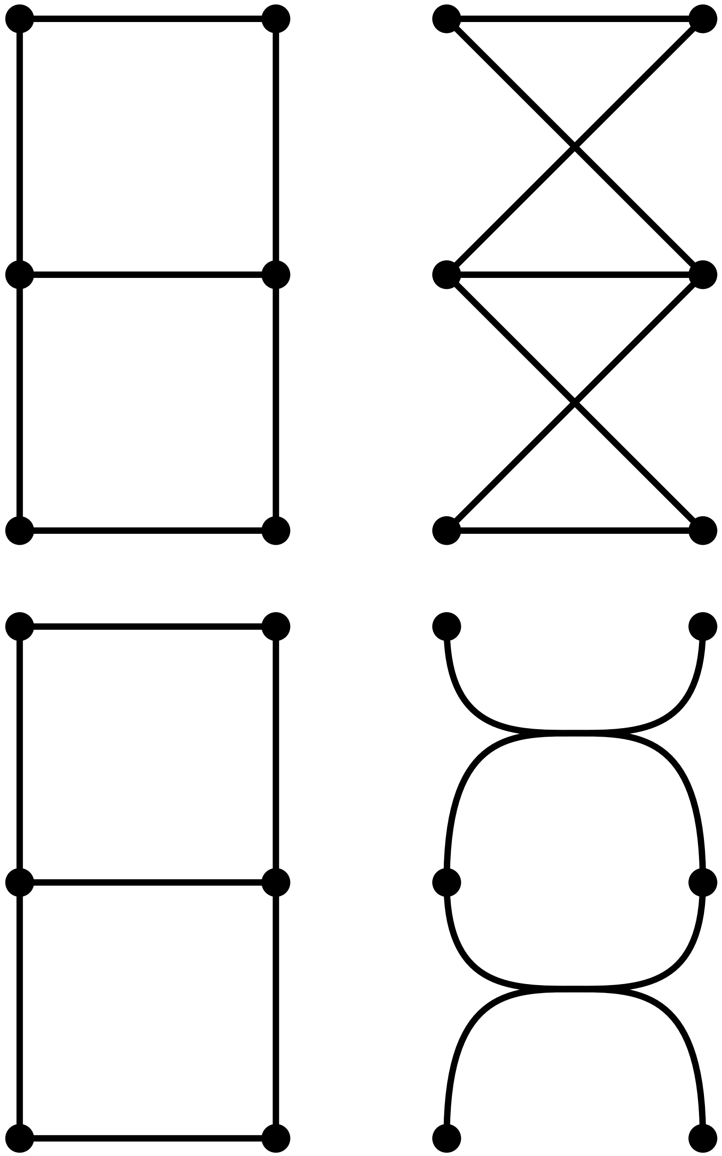
Two Problems for Strictness

Domino



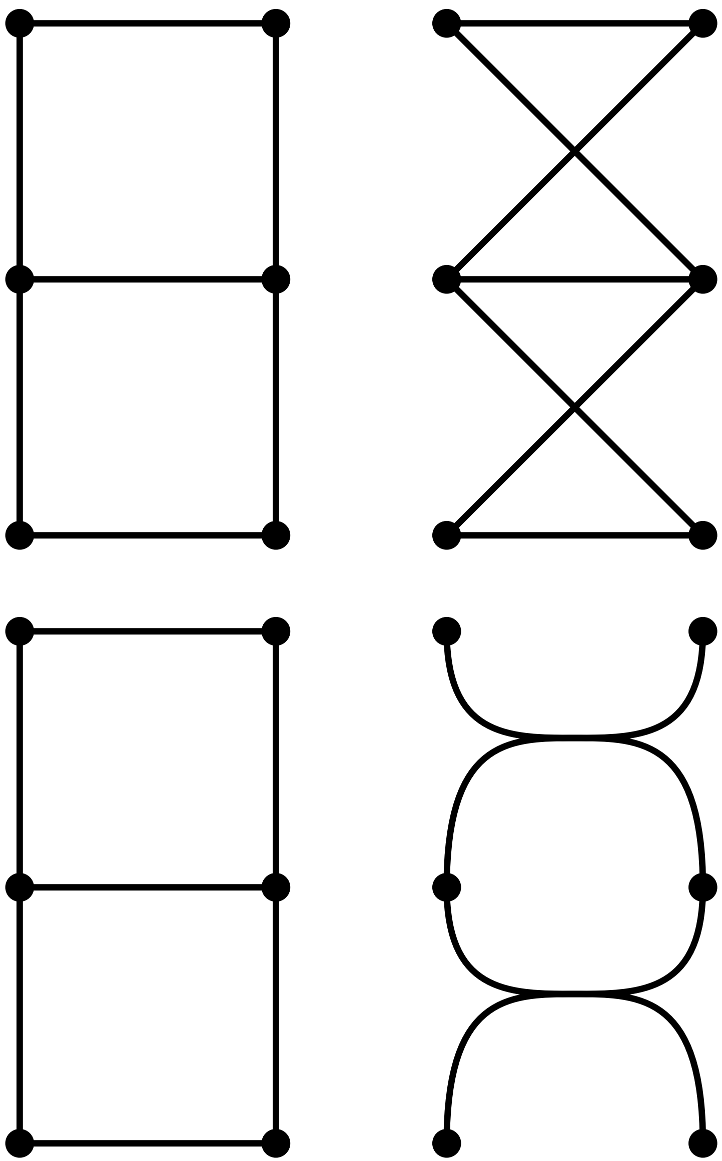
Two Problems for Strictness

Domino

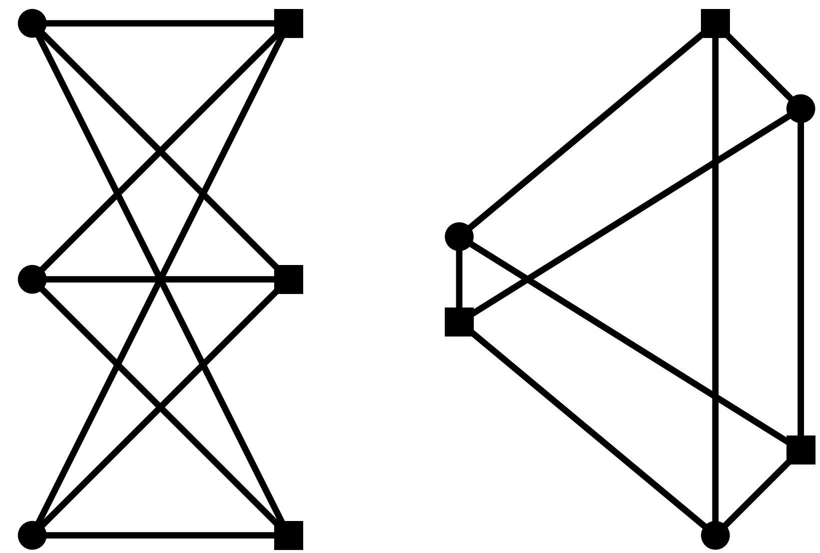


Two Problems for Strictness

Domino

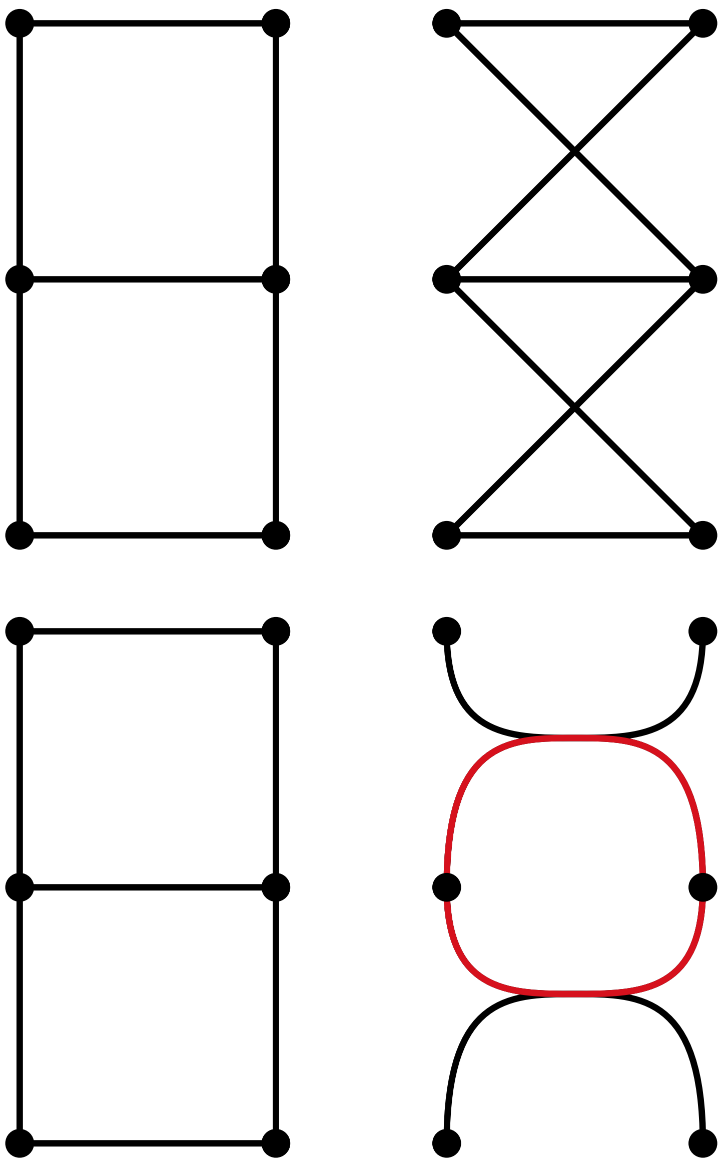


$K_{3,3}$

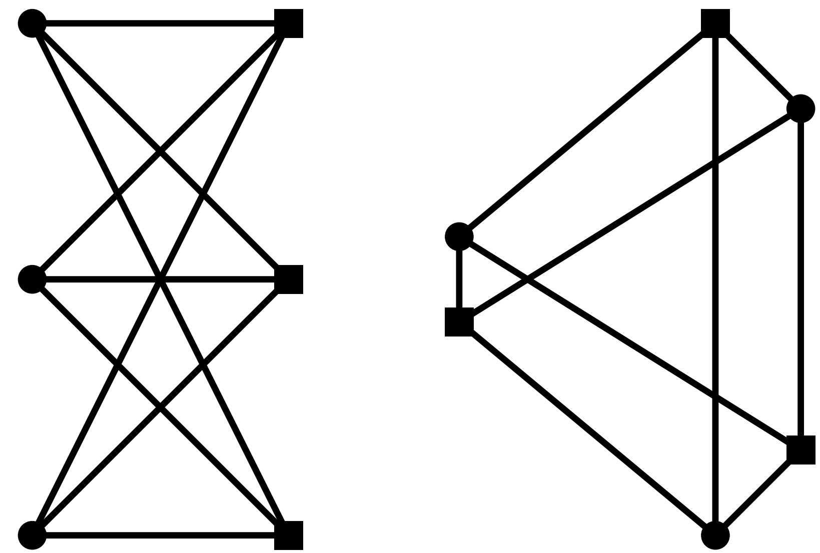


Two Problems for Strictness

Domino

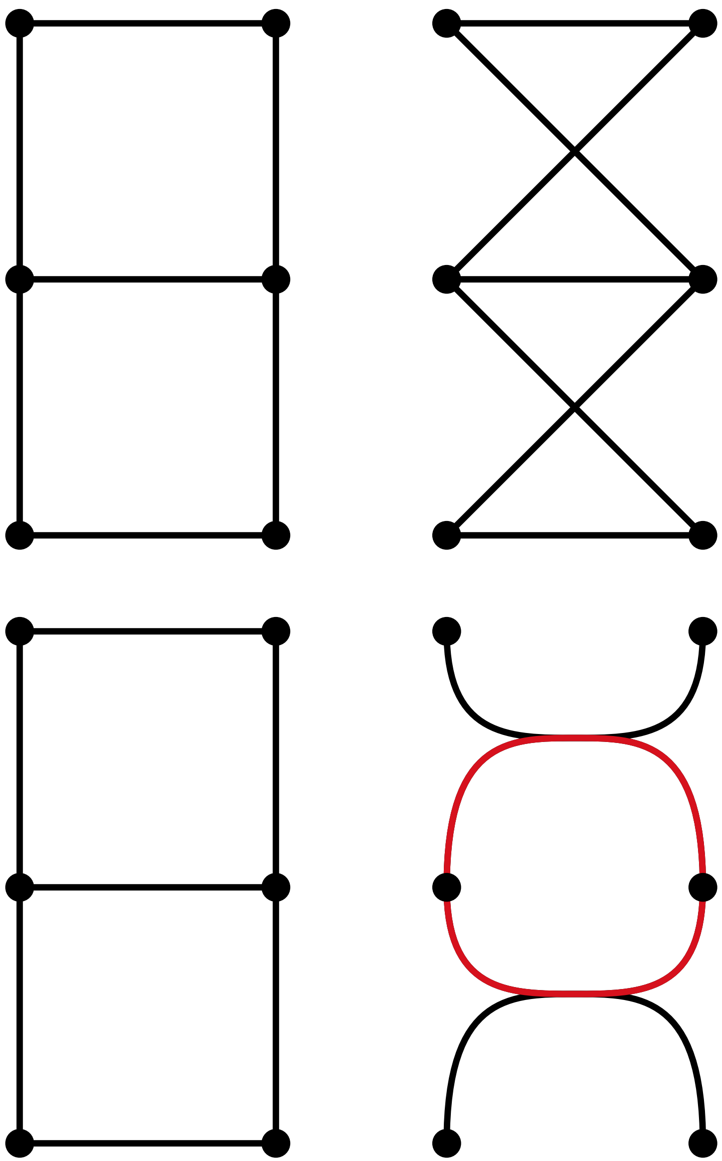


$K_{3,3}$

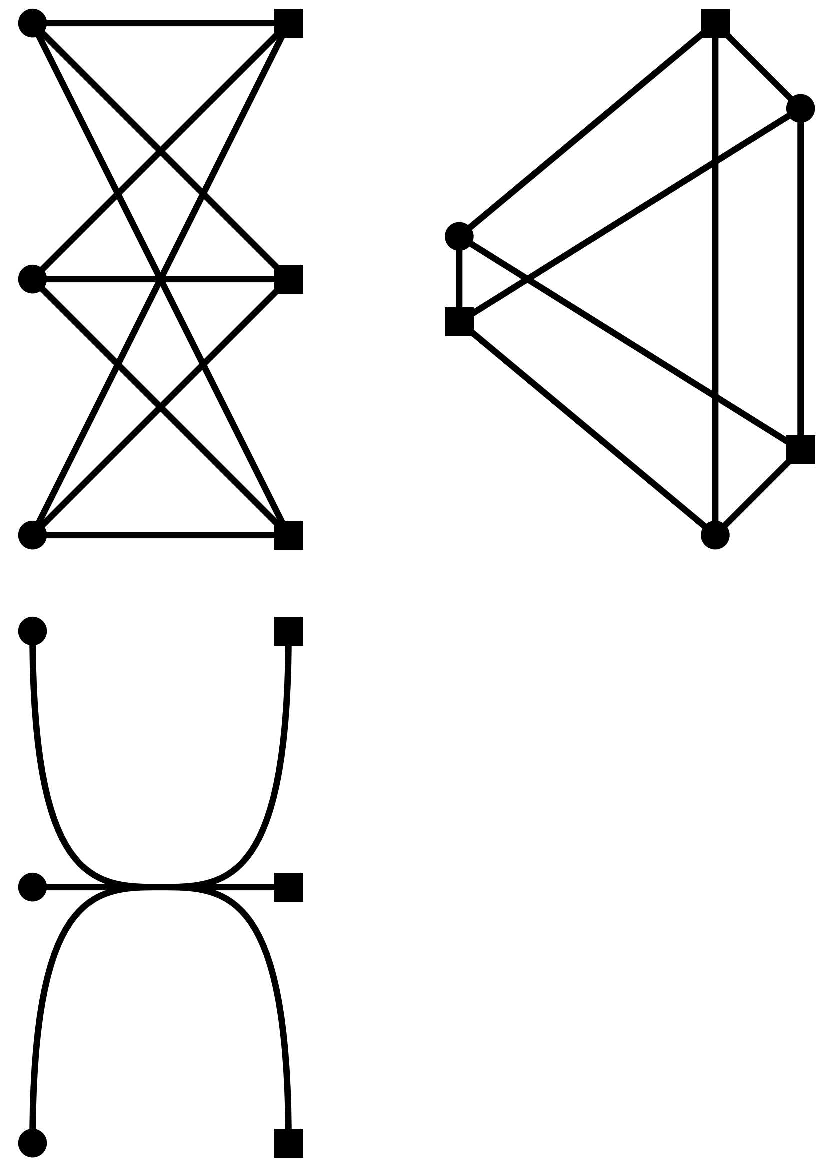


Two Problems for Strictness

Domino

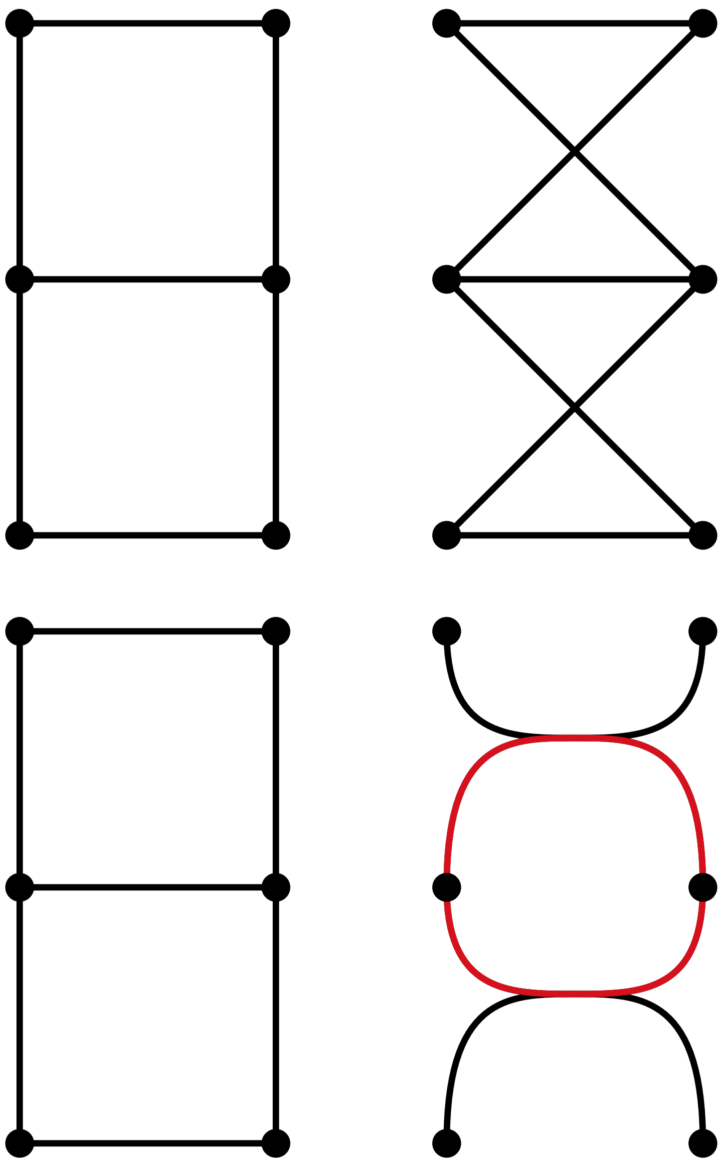


$K_{3,3}$

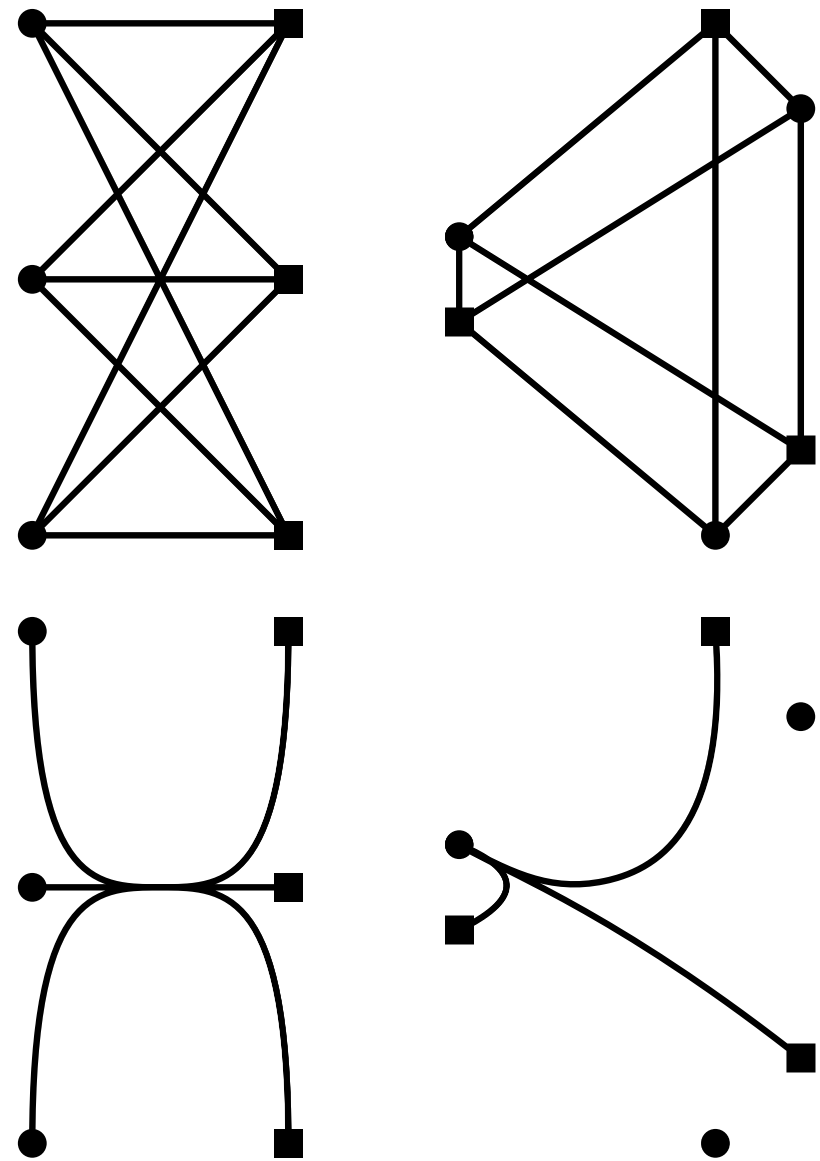


Two Problems for Strictness

Domino

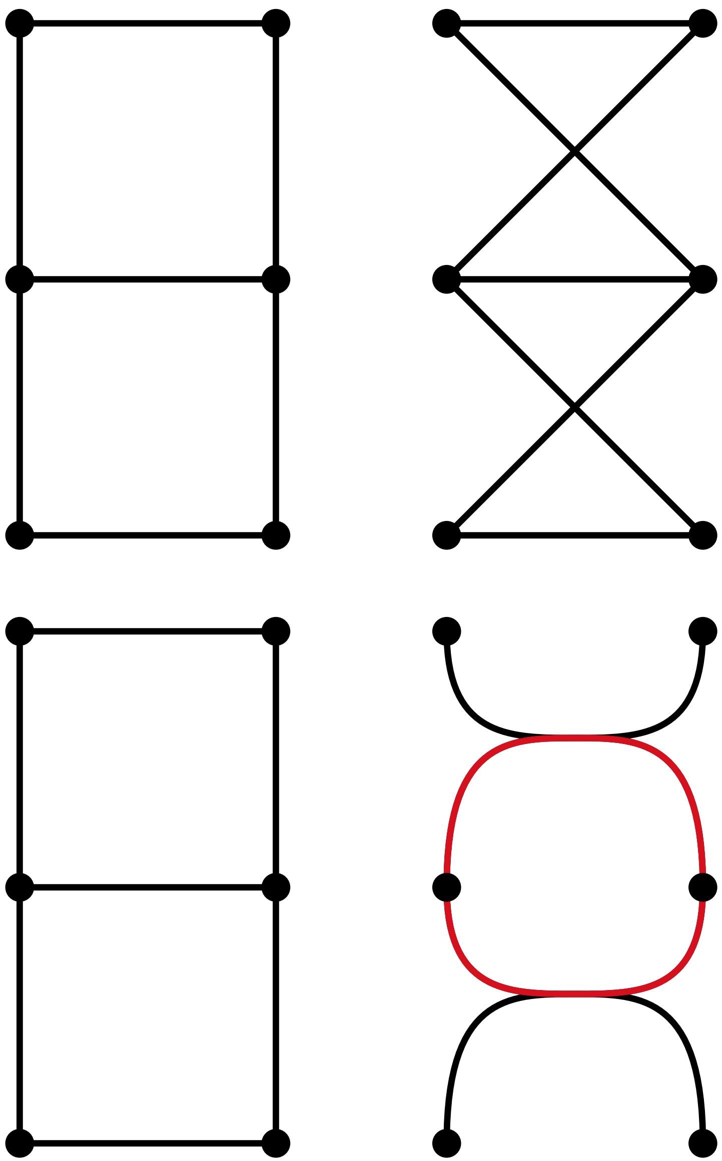


$K_{3,3}$

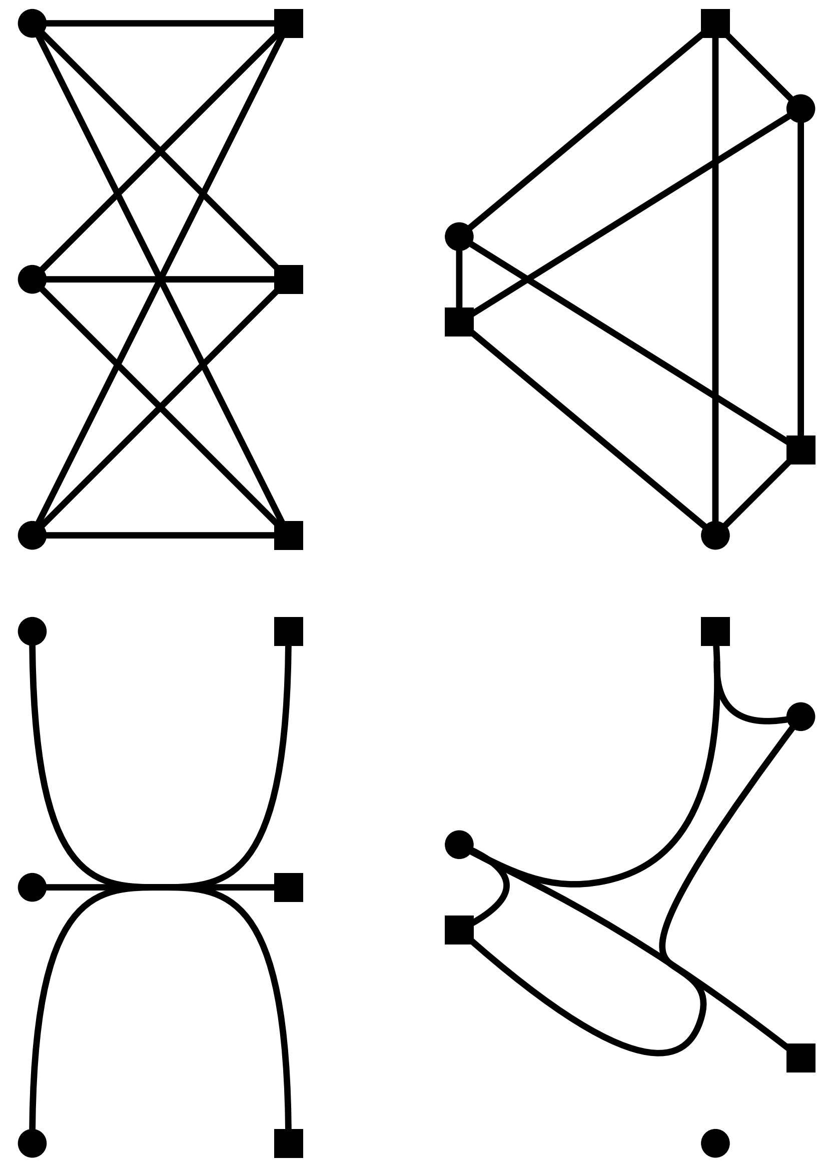


Two Problems for Strictness

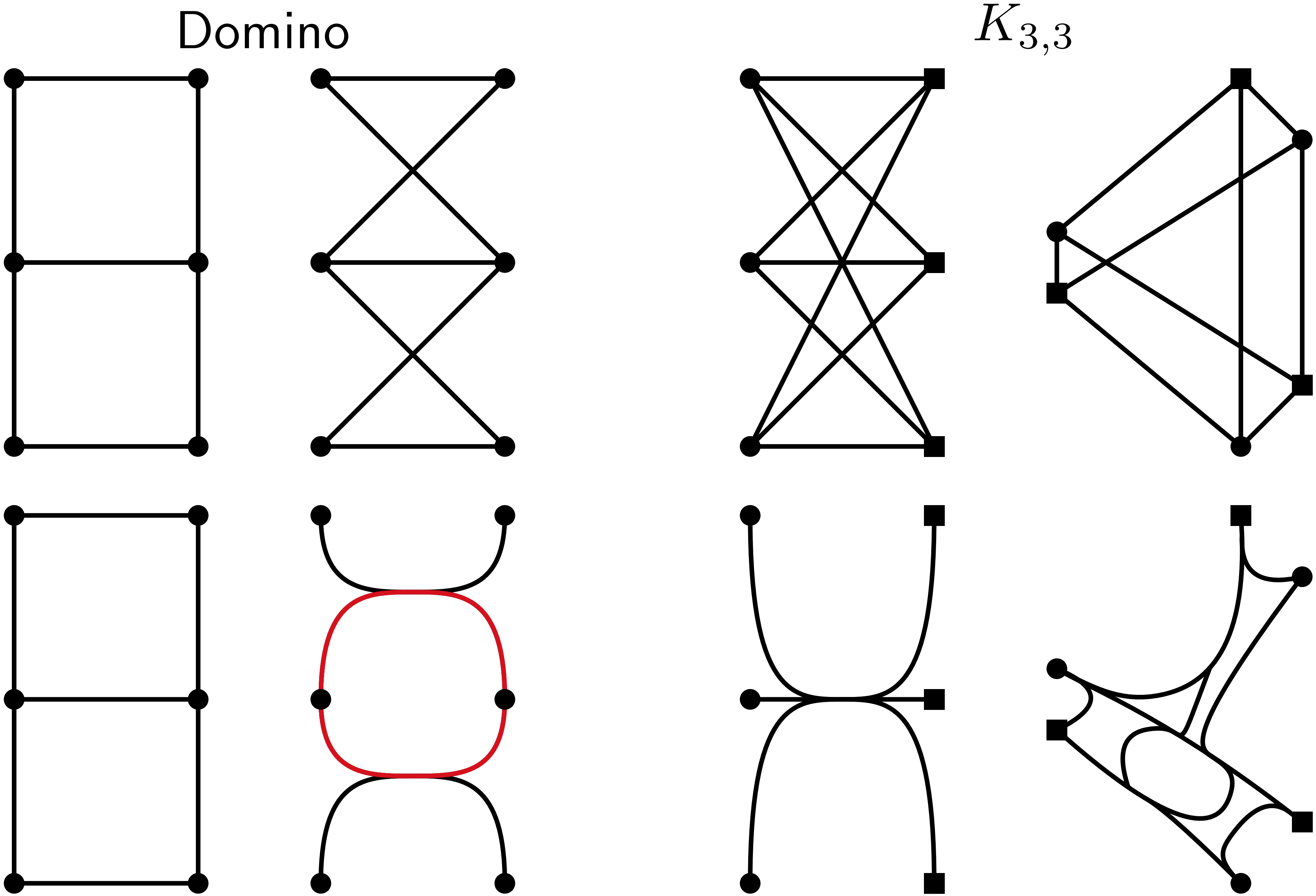
Domino



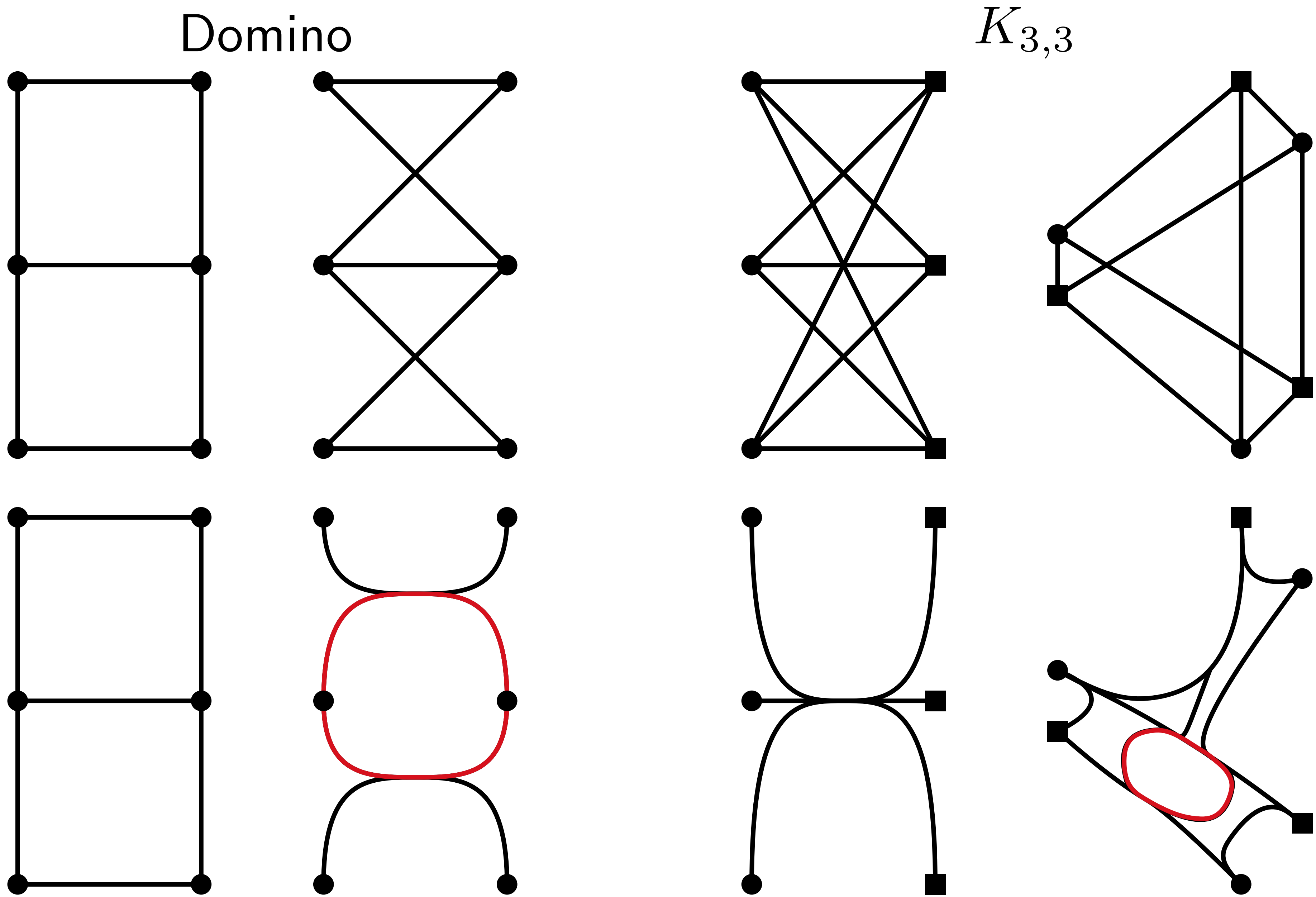
$K_{3,3}$



Two Problems for Strictness

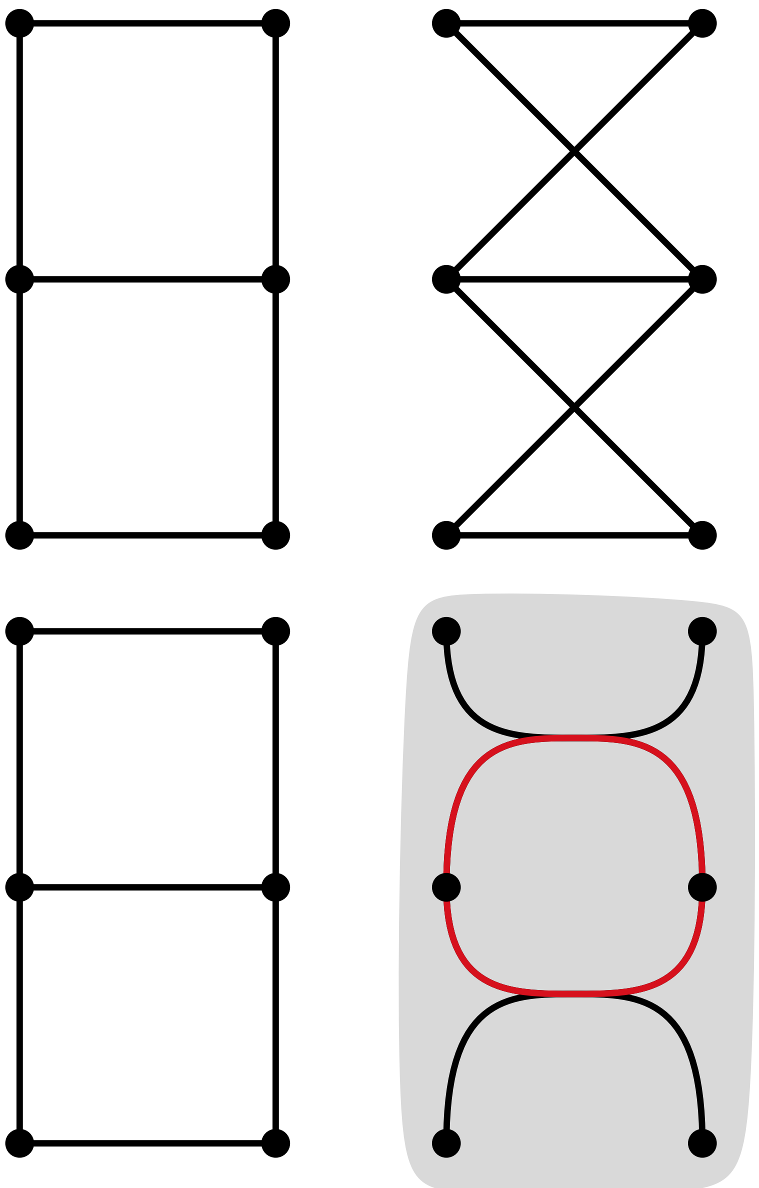


Two Problems for Strictness



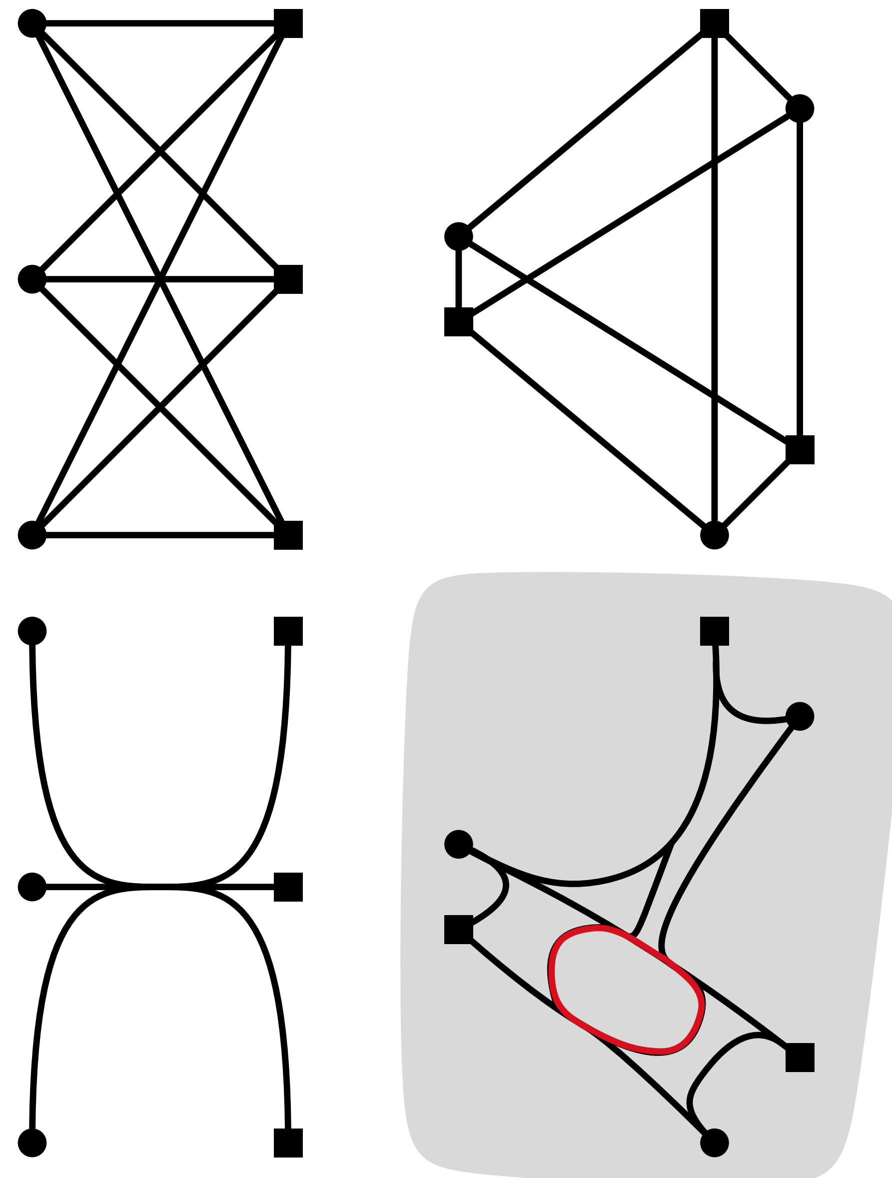
Two Problems for Strictness

Domino



Twisted domino

$K_{3,3}$



Alternating $K_{3,3}$

SOC Drawings of Bipartite Permutation Graphs

Bipartite Permutation graphs (BP) are graphs that are

SOC Drawings of Bipartite Permutation Graphs

Bipartite Permutation graphs (BP) are graphs that are

- Bipartite

SOC Drawings of Bipartite Permutation Graphs

Bipartite Permutation graphs (BP) are graphs that are

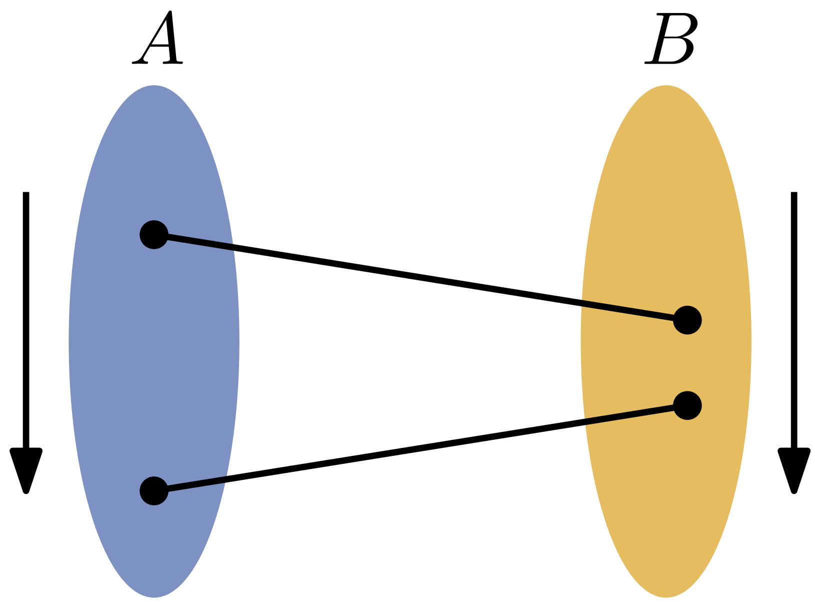
- Bipartite
- Permutation

SOC Drawings of Bipartite Permutation Graphs

Bipartite Permutation graphs (BP) are graphs that are

- Bipartite
- Permutation

Better characerization for us:

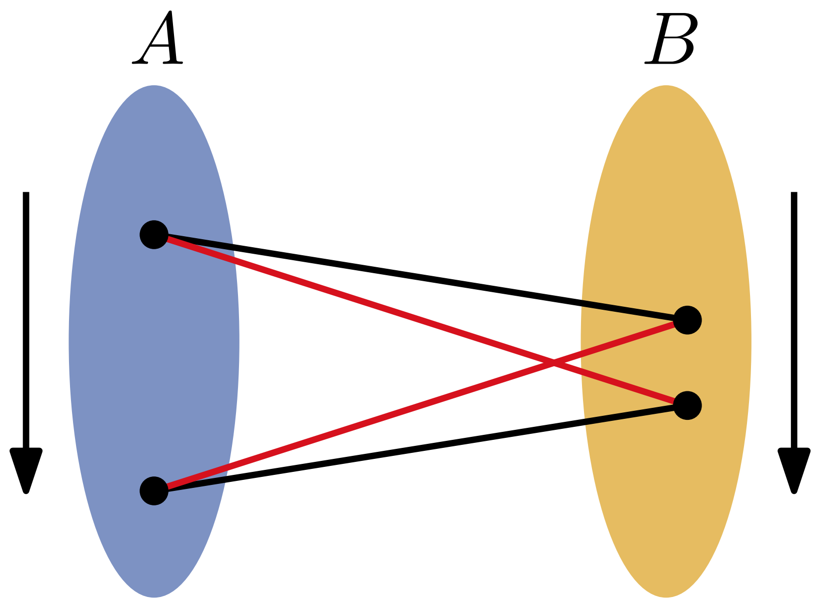


SOC Drawings of Bipartite Permutation Graphs

Bipartite Permutation graphs (BP) are graphs that are

- Bipartite
- Permutation

Better characerization for us:



$\Rightarrow$ Confluent drawing easily possible (Formal proof [Hui et al. 09])

Domino graph is a Bipartite Permutation graph $\Rightarrow$ strictness?

BP Without Dominos

BP without dominos have soc drawings

BP Without Dominos

BP without dominos have soc drawings

Draw given graph with algorithm by Hui et al.

BP Without Dominos

BP without dominos have soc drawings

Draw given graph with algorithm by Hui et al.

Observations:

- $K_{3,3}$ is never alternating
- Dominos might be twisted

BP Without Dominos

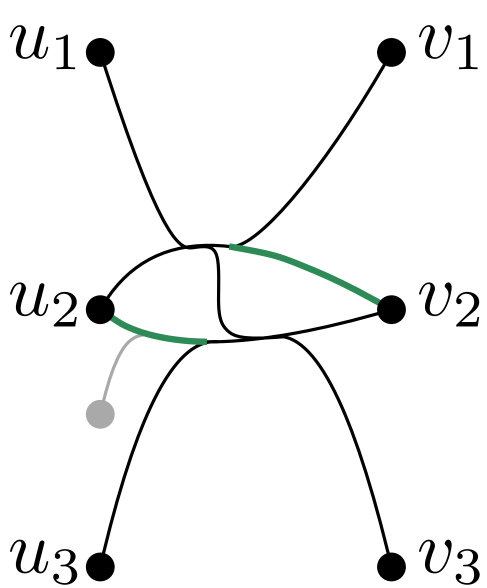
BP without dominos have soc drawings

Draw given graph with algorithm by Hui et al.

Observations:

- $K_{3,3}$ is never alternating
- Dominos might be twisted

We can always untwist dominos



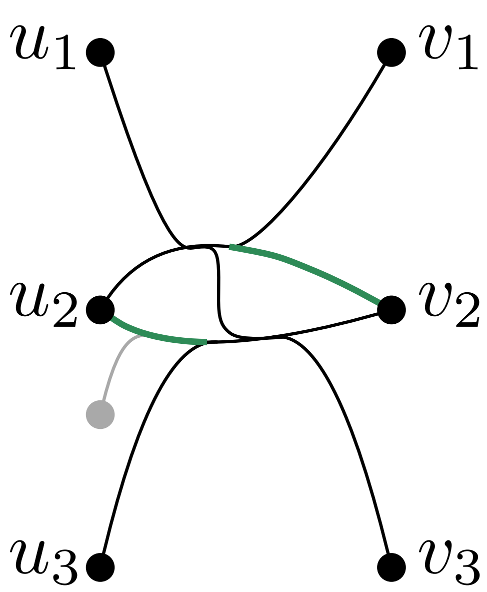
BP without dominos have soc drawings

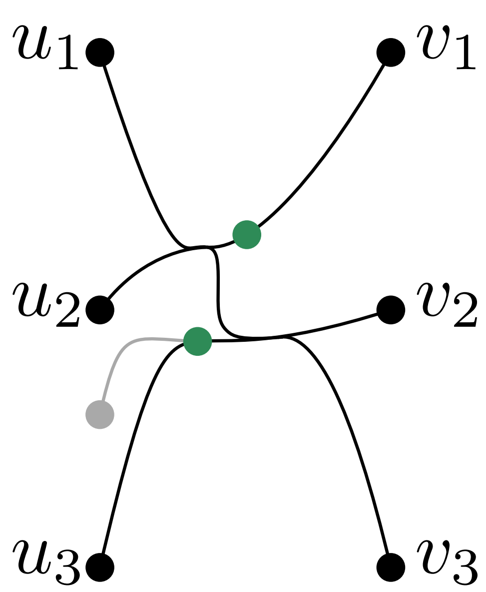
Draw given graph with algorithm by Hui et al.

Observations:

- $K_{3,3}$ is never alternating
- Dominos might be twisted

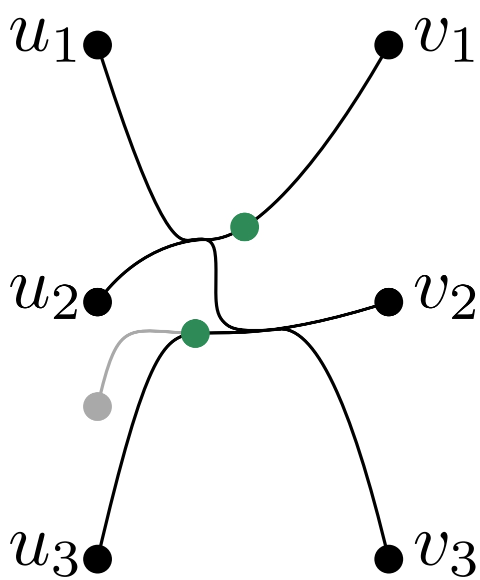
We can always untwist dominos





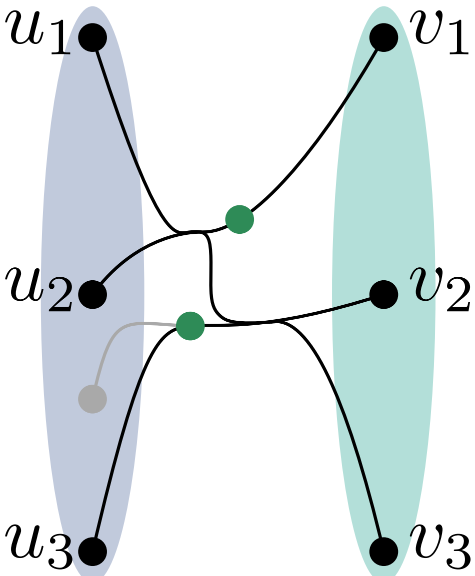
Bipartite Strict Confluent Drawings

Drawings such that



Bipartite Strict Confluent Drawings

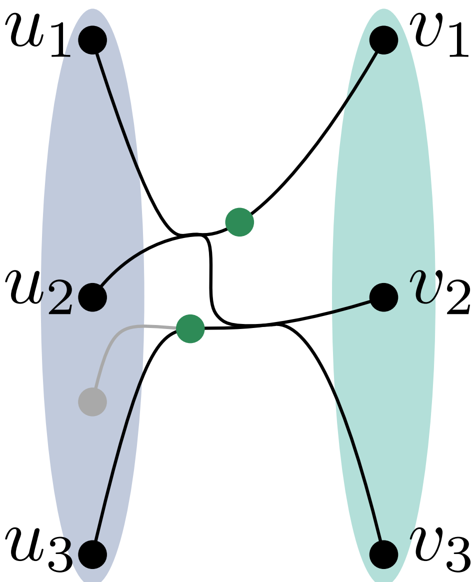
Drawings such that



- Vertices drawn in bipartite "manner"

Bipartite Strict Confluent Drawings

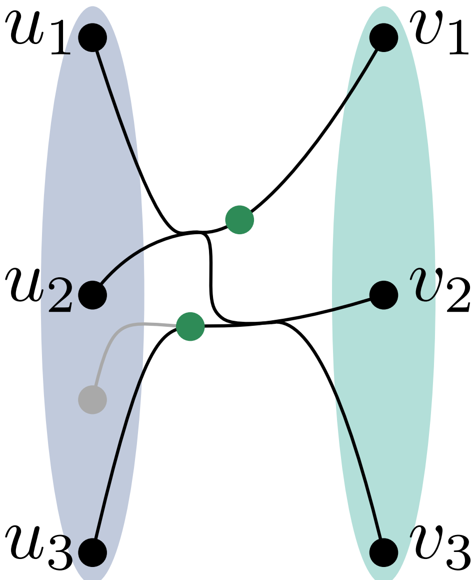
Drawings such that



- Vertices drawn in bipartite “manner”
- Edges drawn as strict confluent network between them

Bipartite Strict Confluent Drawings

Drawings such that



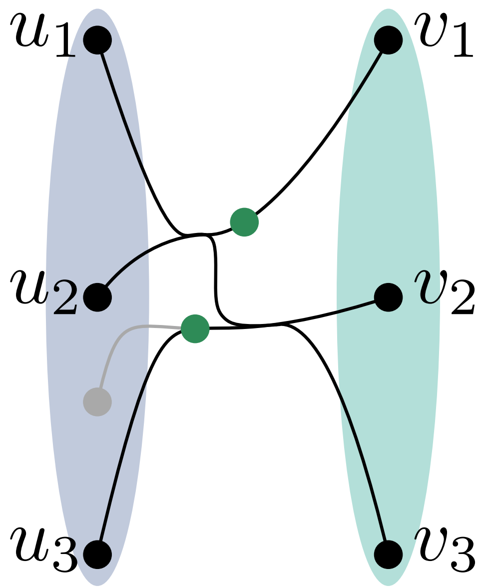
- Vertices drawn in bipartite “manner”
- Edges drawn as strict confluent network between them

Slide before

Bipartite permutation graphs without domino have such drawings

Bipartite Strict Confluent Drawings

Drawings such that



- Vertices drawn in bipartite "manner"
- Edges drawn as strict confluent network between them

Slide before

Bipartite permutation graphs without domino have such drawings

Such drawings are bipartite permutation graphs without dominos:

- Twisted domino is only order admitting bipartite confluent drawing
- But twisted domino can not be drawn strict outer-confluent

Counterexample for BP $\subset$ SOC