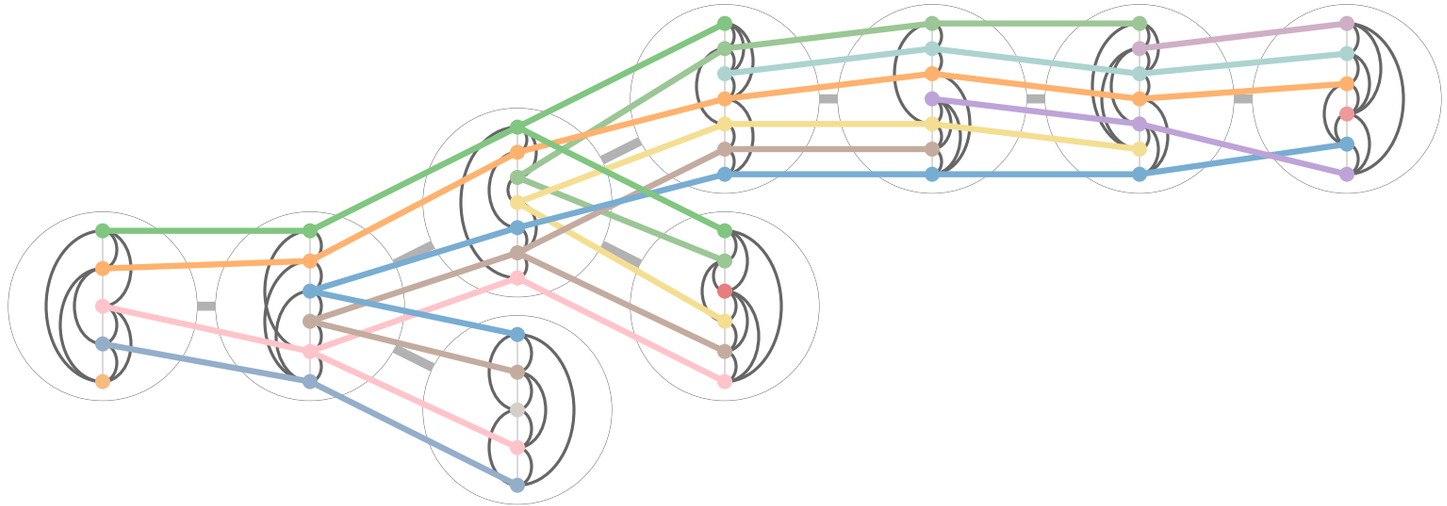


Visualizing Treewidth

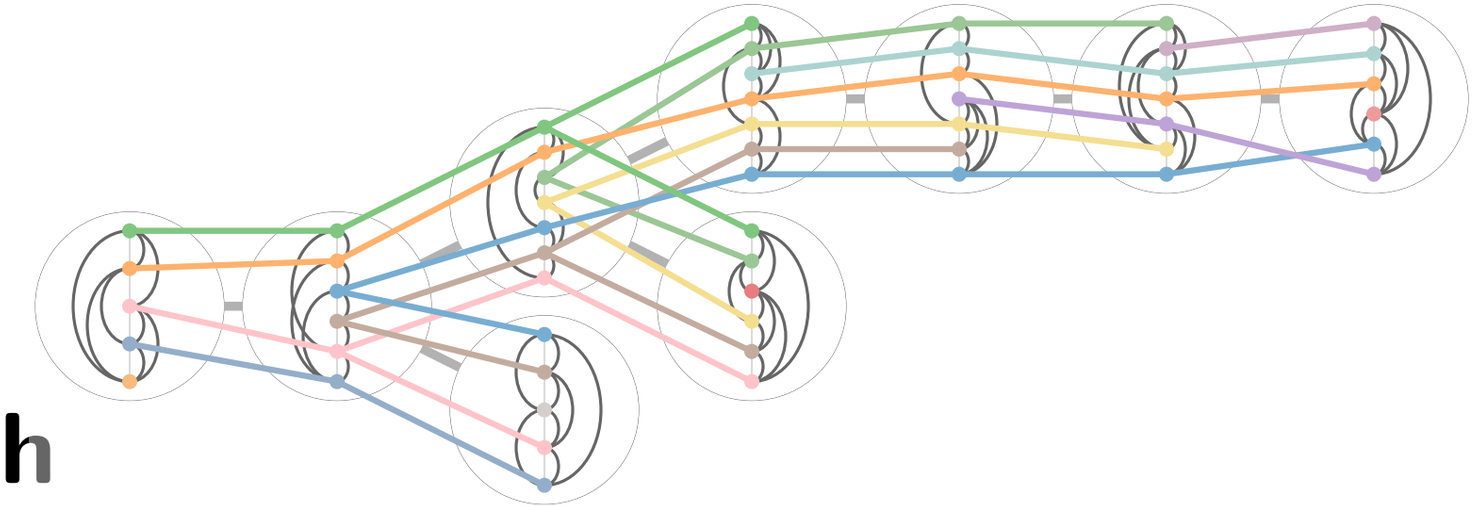
Alvin Chiu[(1)], **Thomas Depian**[(2)], David Eppstein[(1)],
Michael T. Goodrich[(1)], Martin Nöllenburg[(2)]

24. – 26. September · GD 2025

(1) UC Irvine

(2) TU WIEN ac ALGORITHMS AND COMPLEXITY GROUP

WWTF Vienna Science and Technology Fund



Visualizing **Treewidth**

Alvin Chiu[(1)], **Thomas Depian**[(2)], David Eppstein[(1)],
Michael T. Goodrich[(1)], Martin Nöllenburg[(2)]

24. – 26. September · GD 2025

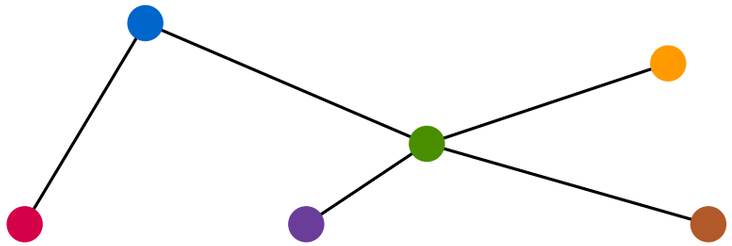
(1) UC Irvine

(2) TU WIEN ac ALGORITHMS AND COMPLEXITY GROUP

WWTF Vienna Science and Technology Fund

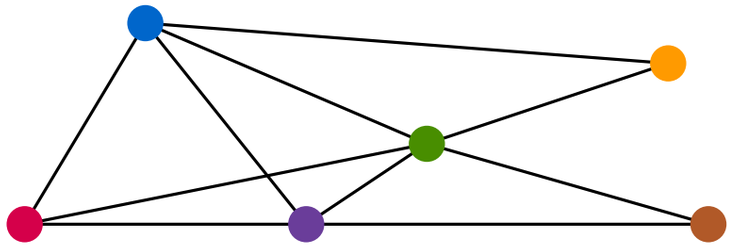
Tree Decompositions & Treewidth

$G = (V, E)$



Tree Decompositions & Treewidth

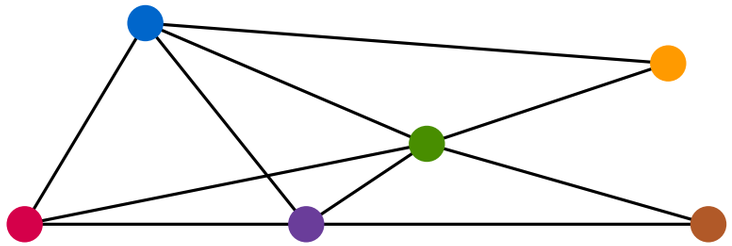
$G = (V, E)$



Tree Decompositions & Treewidth

Treewidth $\approx$ how "tree-like" is G

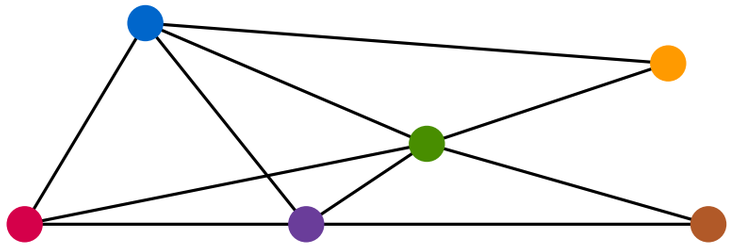
$G = (V, E)$



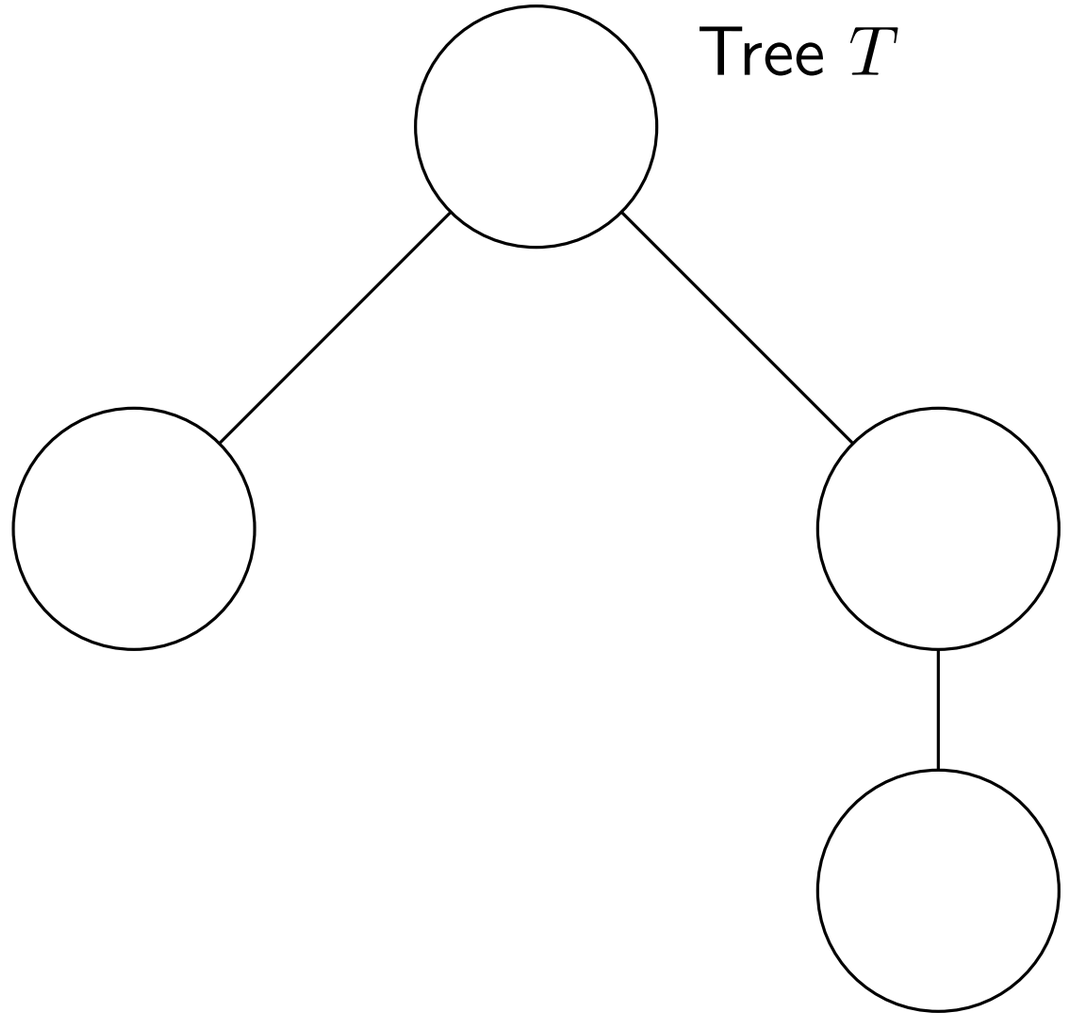
Tree Decompositions & Treewidth

Treewidth $\approx$ how "tree-like" is G

$G = (V, E)$



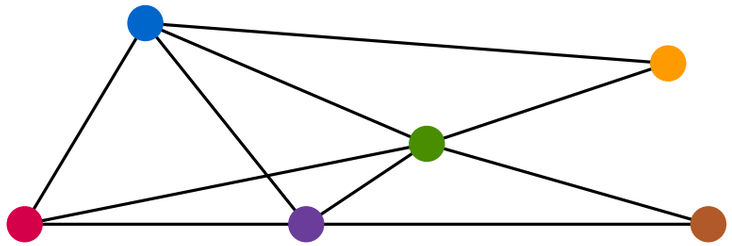
Tree Decomposition:



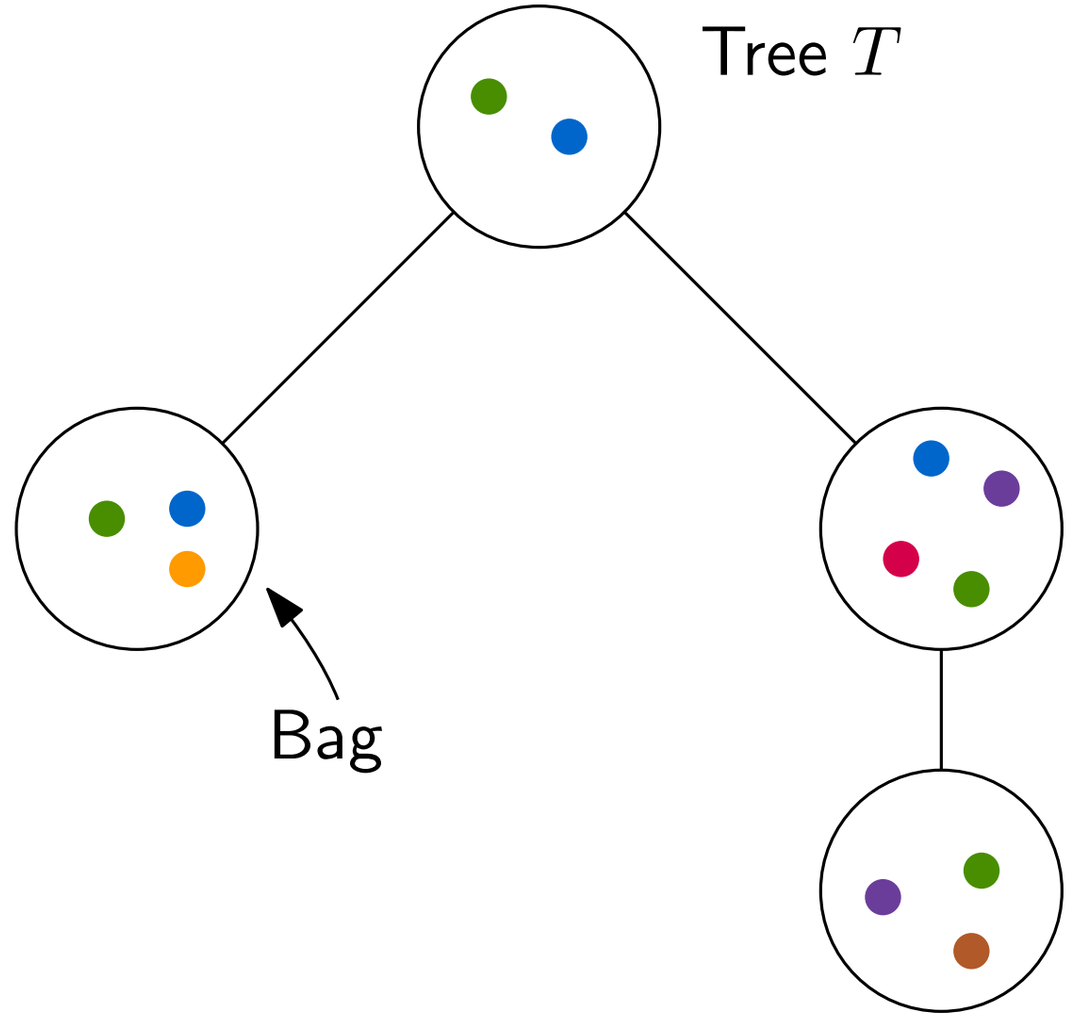
Tree Decompositions & Treewidth

Treewidth $\approx$ how "tree-like" is G

$G = (V, E)$



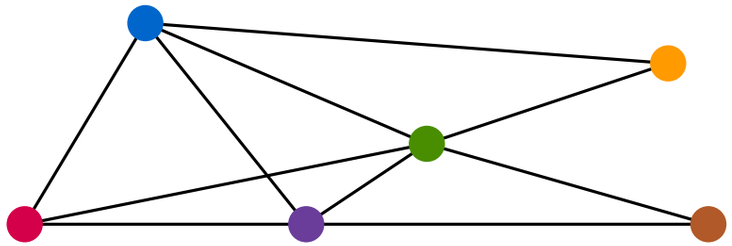
Tree Decomposition:



Tree Decompositions & Treewidth

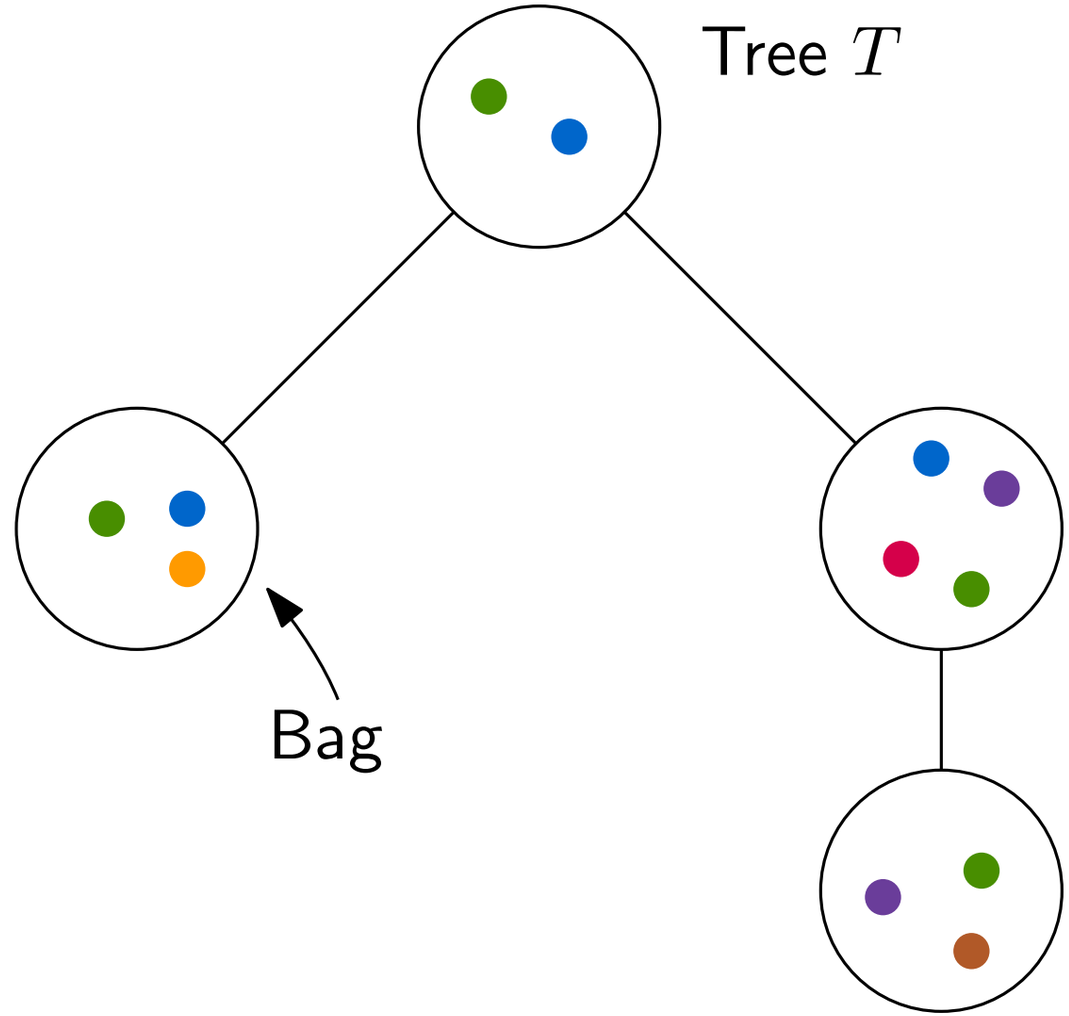
Treewidth $\approx$ how "tree-like" is G

$G = (V, E)$



Tree Decomposition:

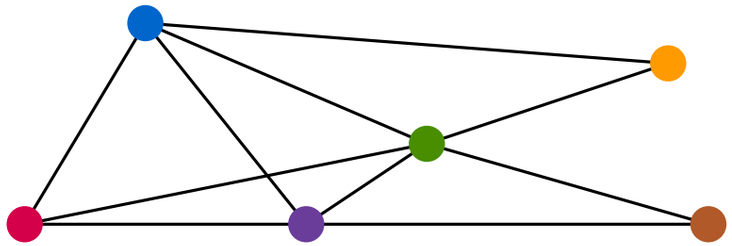
1. For each vertex v, there is a bag containing v



Tree Decompositions & Treewidth

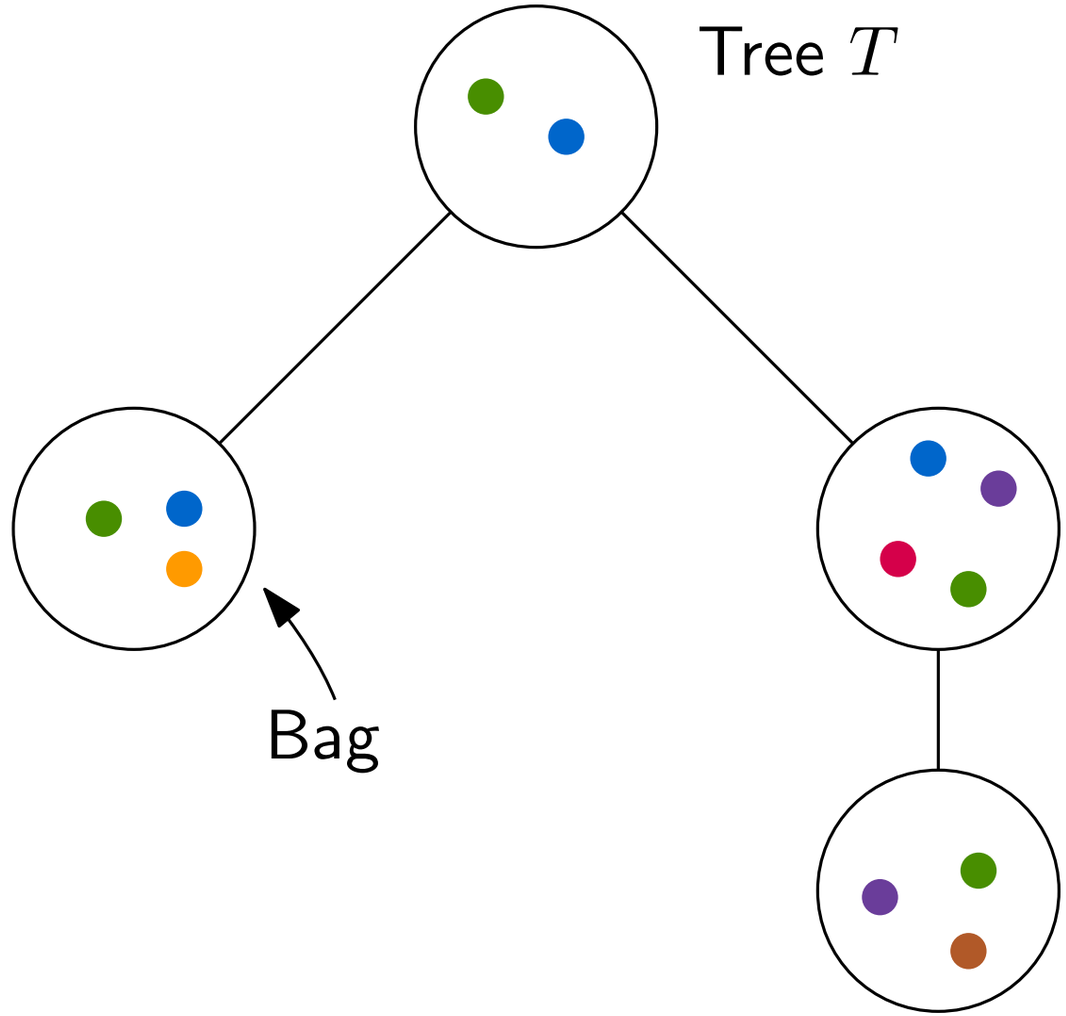
Treewidth $\approx$ how "tree-like" is G

$G = (V, E)$



Tree Decomposition:

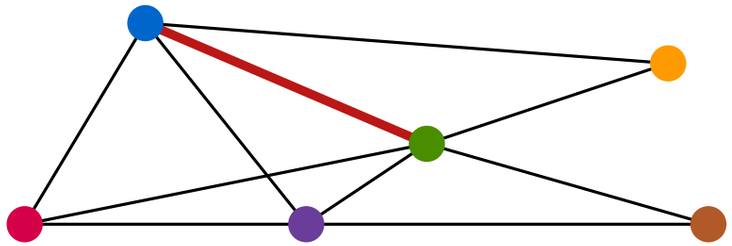
1. For each vertex v, there is a bag containing v
2. For each edge uv, there is a bag containing u and v



Tree Decompositions & Treewidth

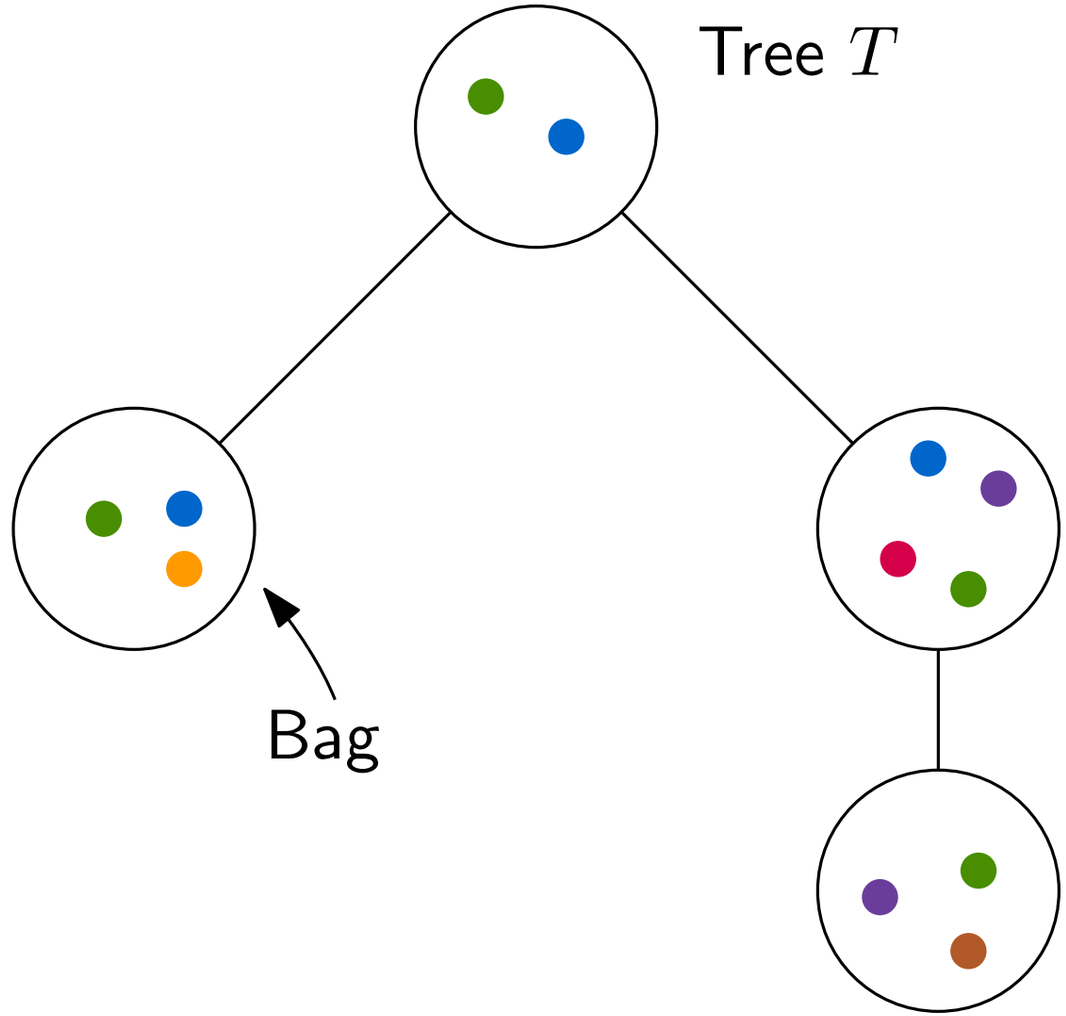
Treewidth $\approx$ how "tree-like" is G

$G = (V, E)$



Tree Decomposition:

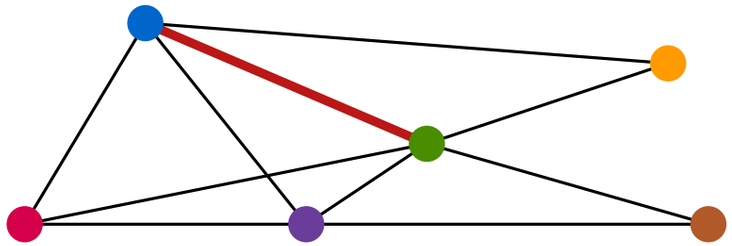
1. For each vertex v, there is a bag containing v
2. For each edge uv, there is a bag containing u and v



Tree Decompositions & Treewidth

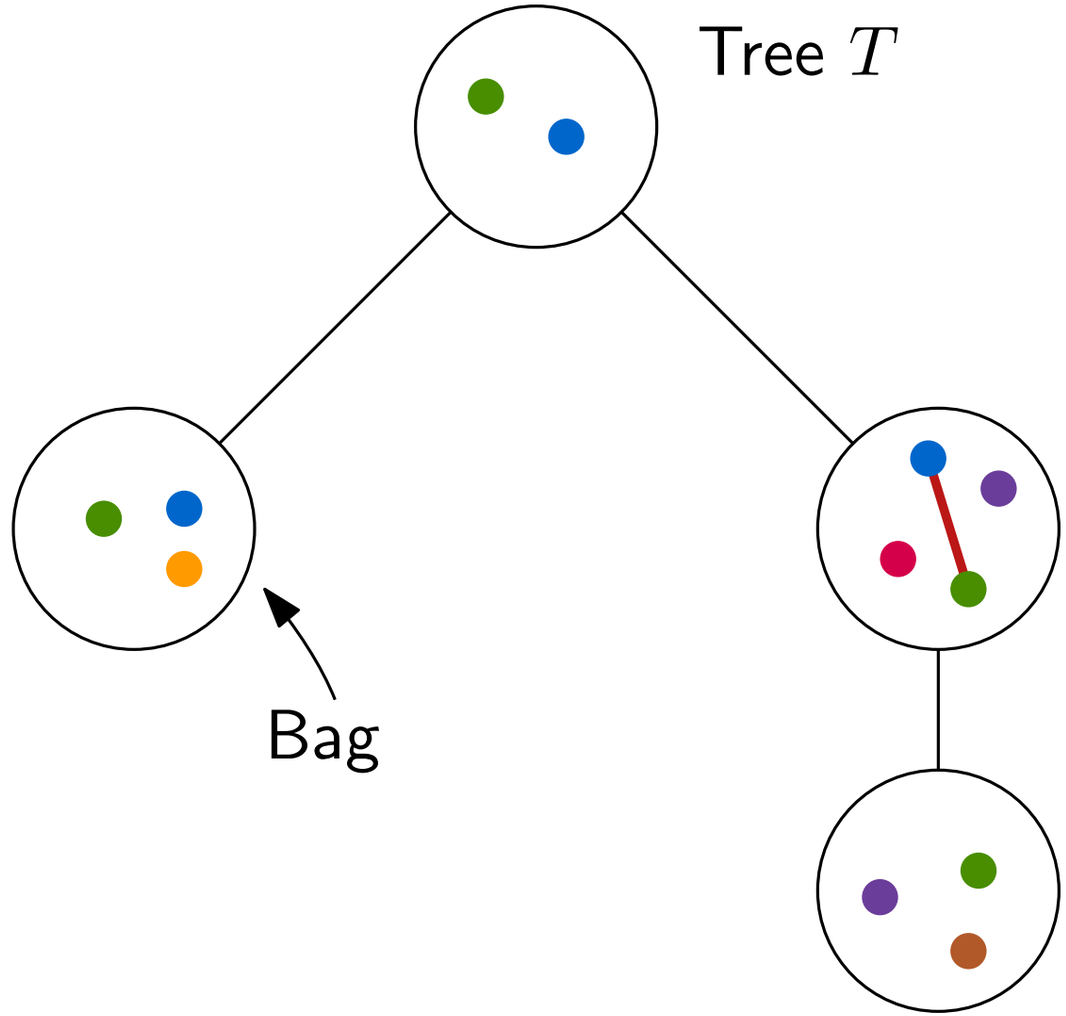
Treewidth $\approx$ how "tree-like" is G

$G = (V, E)$



Tree Decomposition:

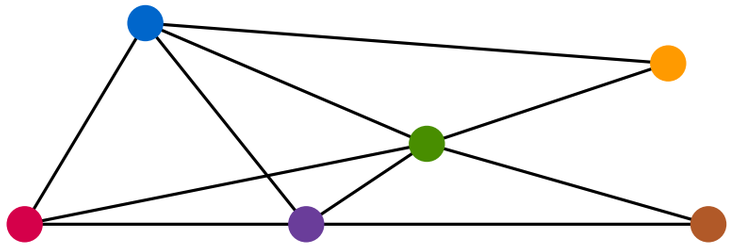
1. For each vertex v, there is a bag containing v
2. For each edge uv, there is a bag containing u and v



Tree Decompositions & Treewidth

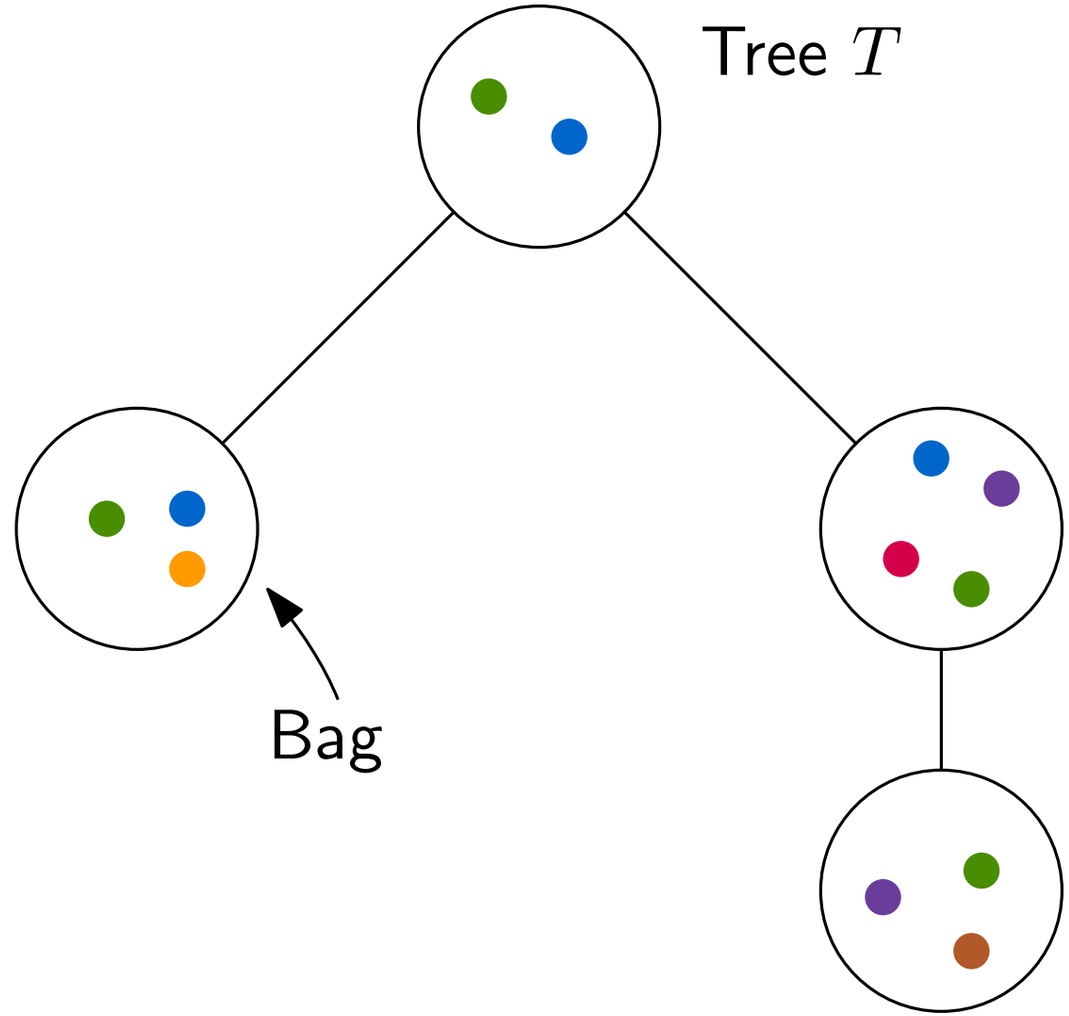
Treewidth $\approx$ how "tree-like" is G

$G = (V, E)$



Tree Decomposition:

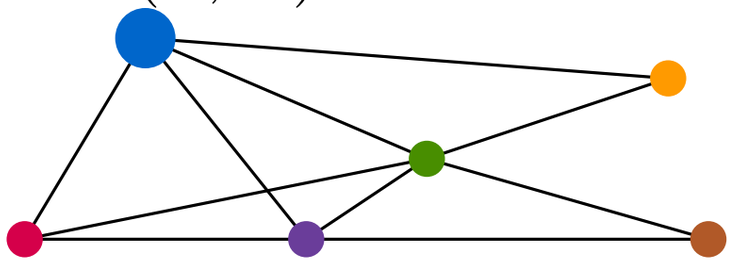
1. For each vertex v, there is a bag containing v
2. For each edge uv, there is a bag containing u and v
3. For each v, bags containing v form subtree of T



Tree Decompositions & Treewidth

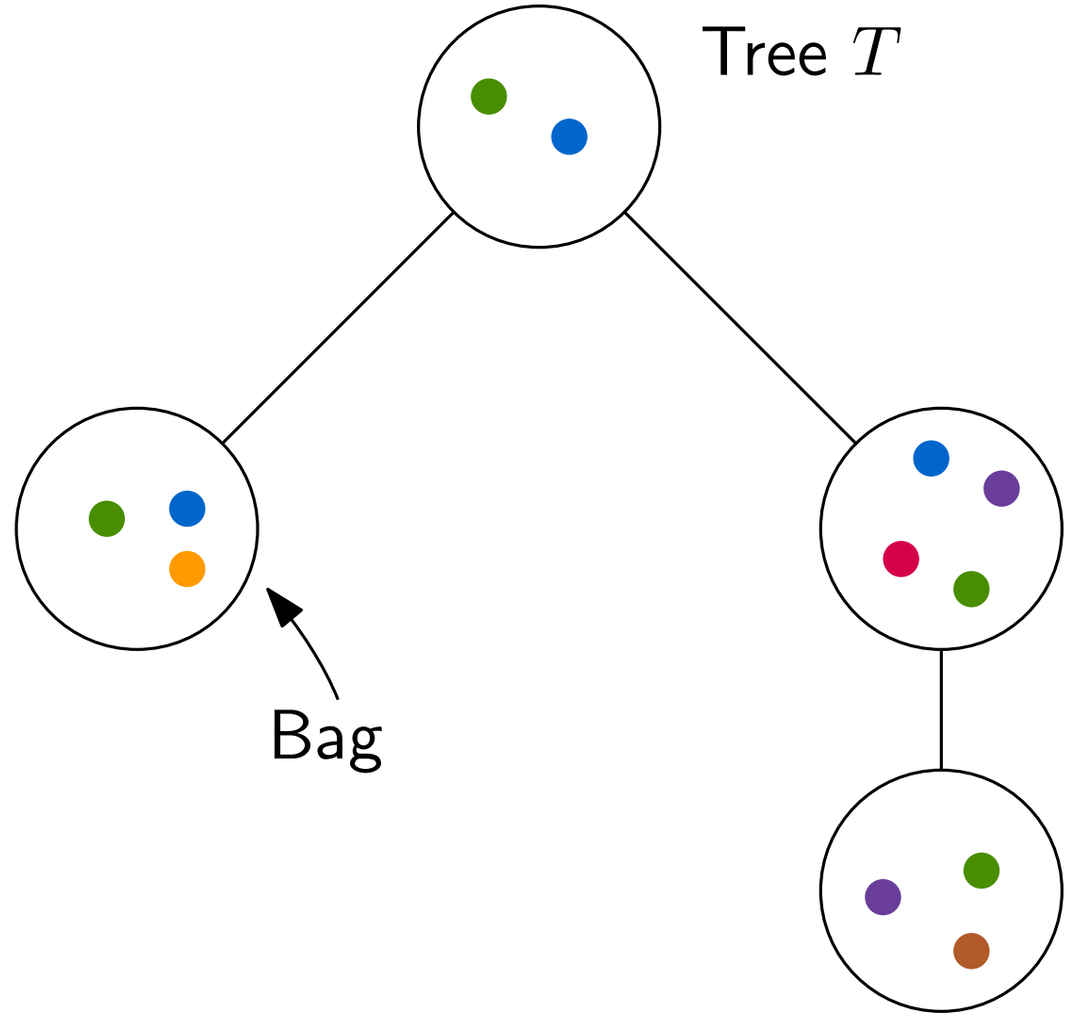
Treewidth $\approx$ how "tree-like" is G

$G = (V, E)$



Tree Decomposition:

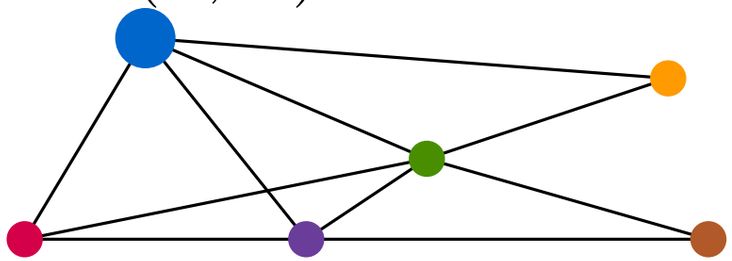
1. For each vertex v, there is a bag containing v
2. For each edge uv, there is a bag containing u and v
3. For each v, bags containing v form subtree of T



Tree Decompositions & Treewidth

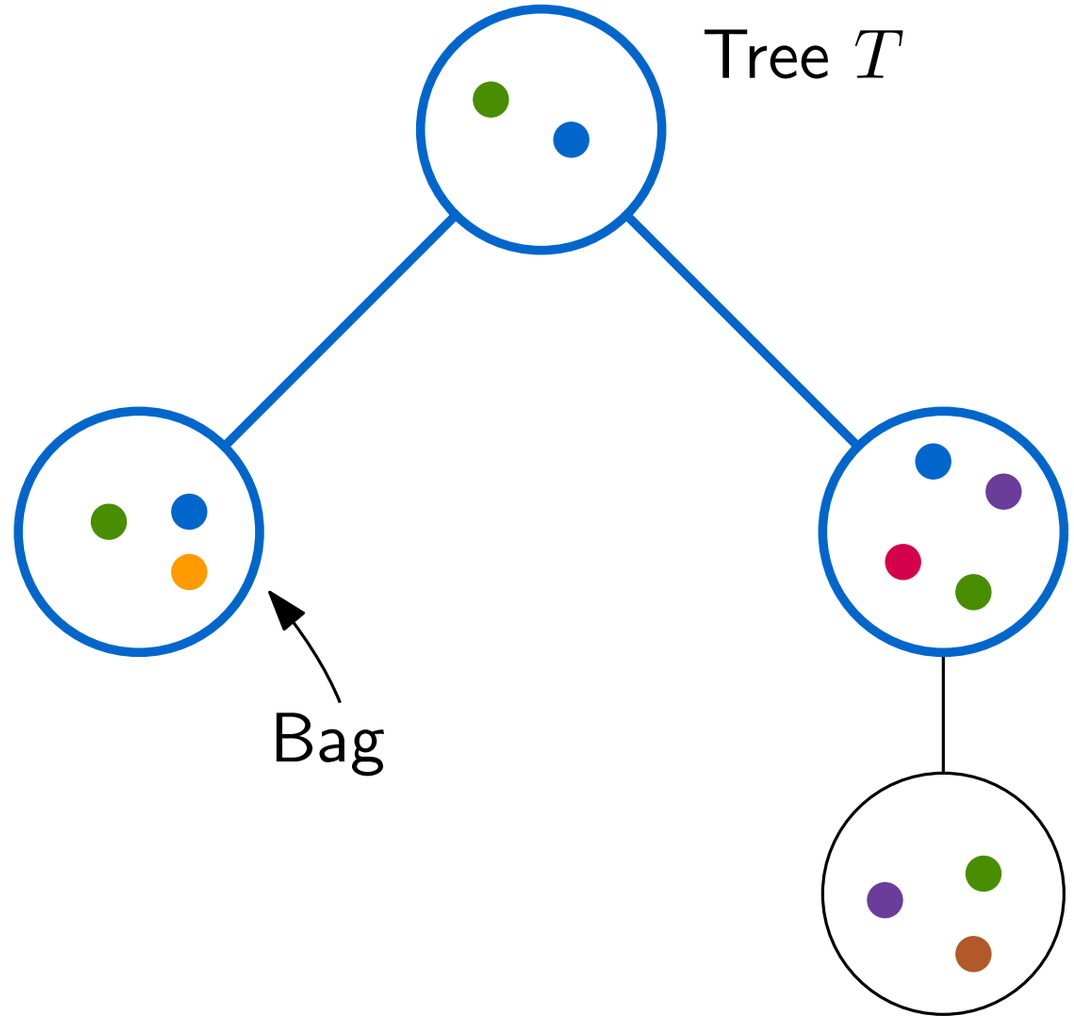
Treewidth $\approx$ how "tree-like" is G

$G = (V, E)$



Tree Decomposition:

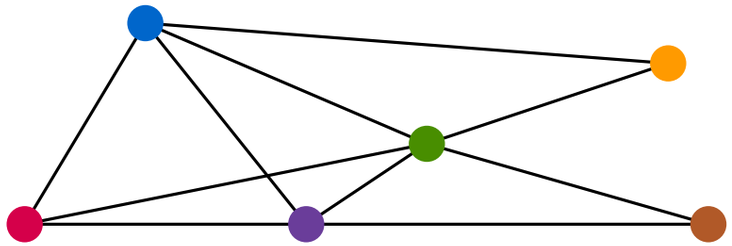
1. For each vertex v, there is a bag containing v
2. For each edge uv, there is a bag containing u and v
3. For each v, bags containing v form subtree of T



Tree Decompositions & Treewidth

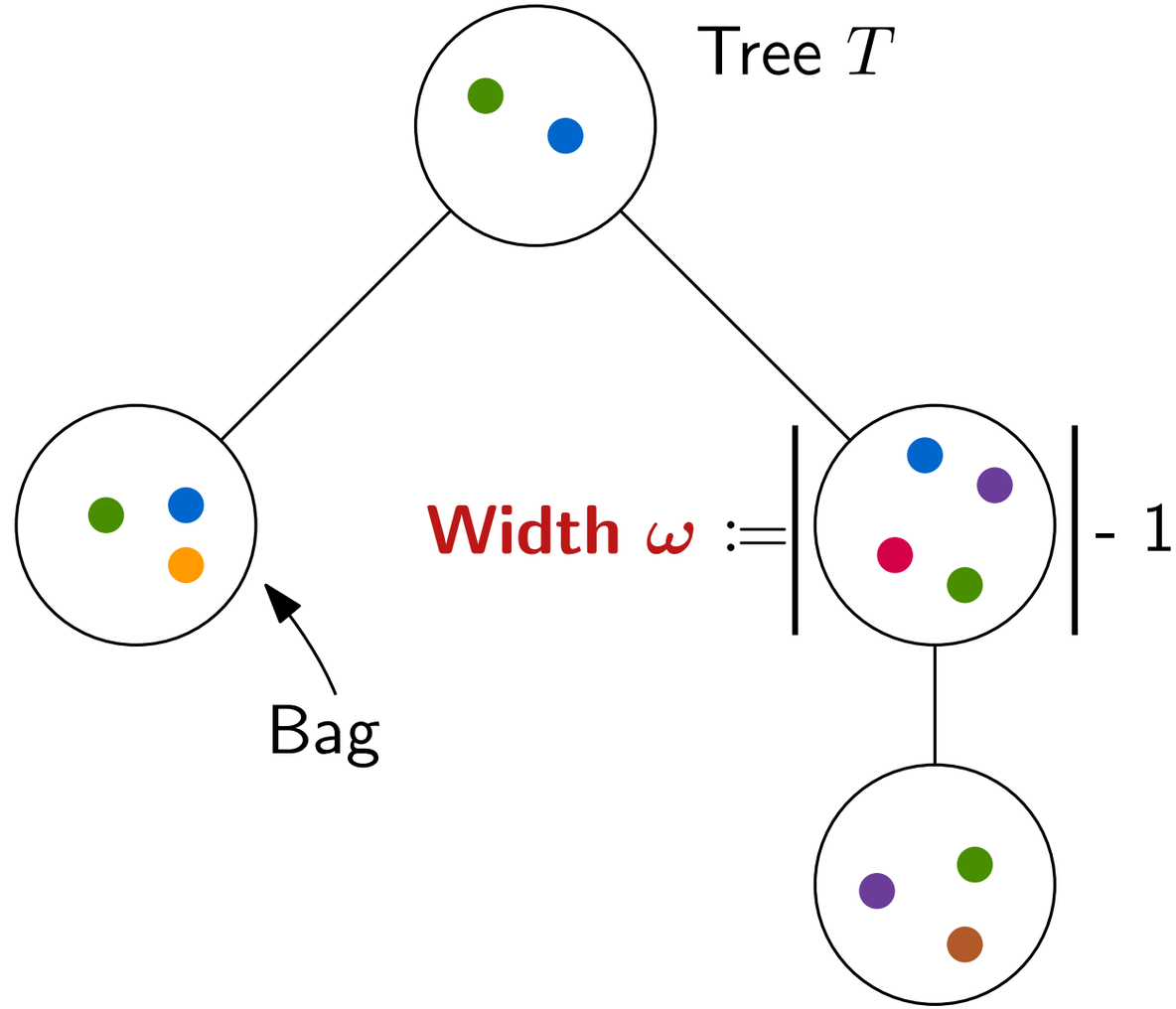
Treewidth $\approx$ how "tree-like" is G

$G = (V, E)$



Tree Decomposition:

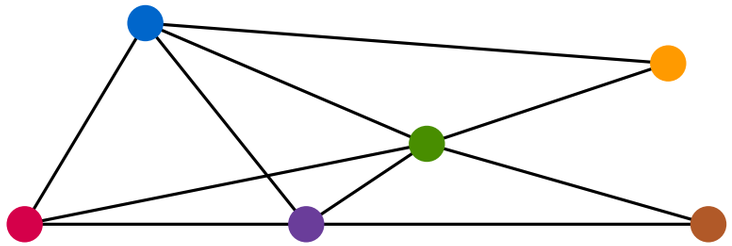
1. For each vertex v, there is a bag containing v
2. For each edge uv, there is a bag containing u and v
3. For each v, bags containing v form subtree of T



Tree Decompositions & Treewidth

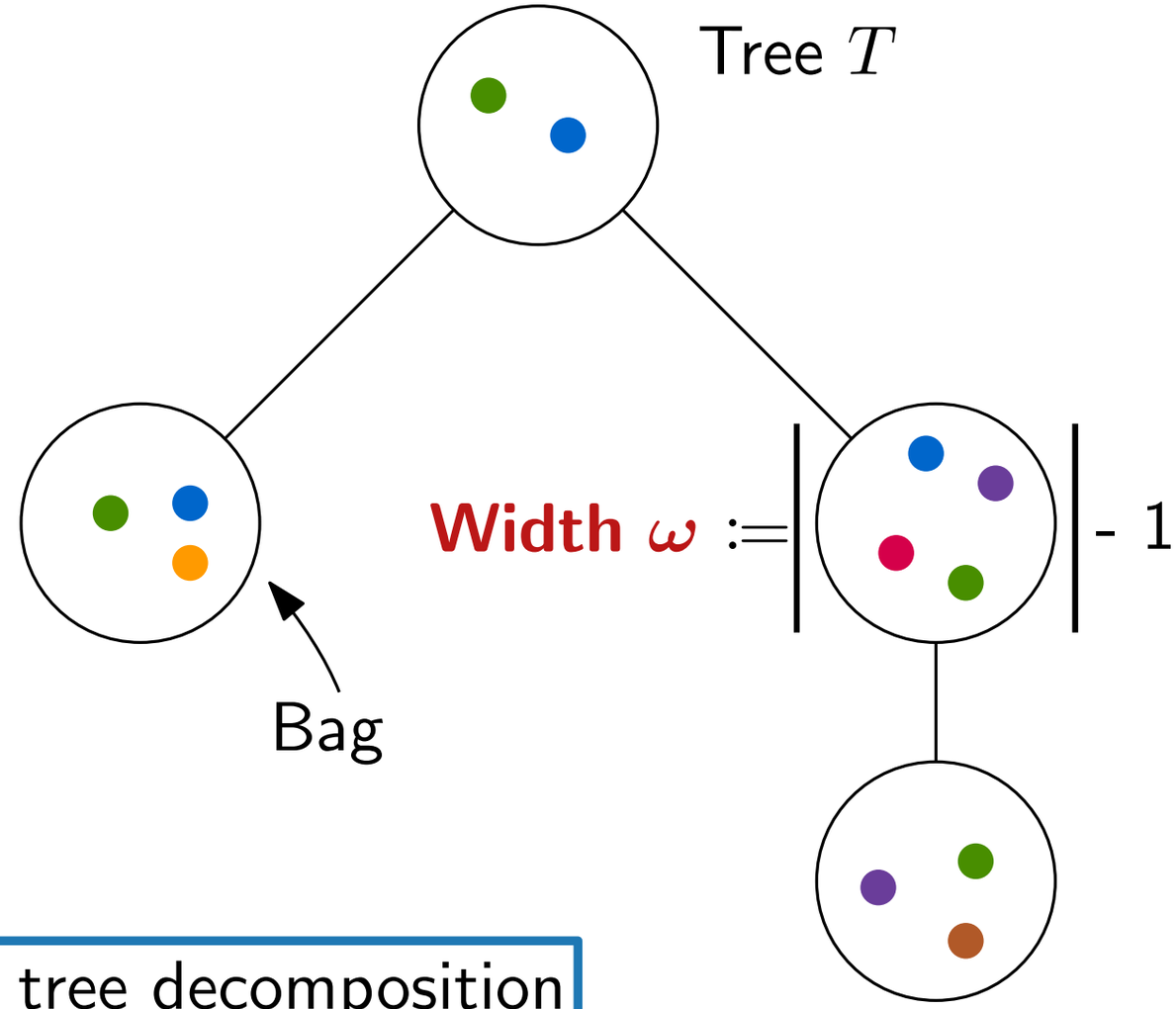
Treewidth $\approx$ how "tree-like" is G

$G = (V, E)$



Tree Decomposition:

1. For each vertex v, there is a bag containing v
2. For each edge uv, there is a bag containing u and v
3. For each v, bags containing v form subtree of T



Treewidth of G: Smallest width of tree decomposition

Why Treewidth?

Many problems: NP-hard on general graphs, poly-time solvable on trees
(Vertex Cover, 3-Colorability, ...)

Why Treewidth?

Many problems: NP-hard on general graphs, poly-time solvable on trees
(Vertex Cover, 3-Colorability, ...)

Hope: Still poly-time solvable for graphs of small (constant) treewidth

Why Treewidth?

Many problems: NP-hard on general graphs, poly-time solvable on trees
(Vertex Cover, 3-Colorability, ...)

Hope: Still poly-time solvable for graphs of small (constant) treewidth
$\Rightarrow$ treewidth is an important parameter in parameterized complexity

[Bodlaender, WG'06] [Dujmović, Eppstein, & Wood, JoDM'17] [Dujmović, Eppstein, & Wood, JoC'05] [Korach and Solel, Dis. Appl. Math.'93] . . .

Why Treewidth?

Many problems: NP-hard on general graphs, poly-time solvable on trees
(Vertex Cover, 3-Colorability, ...)

Hope: Still poly-time solvable for graphs of small (constant) treewidth
$\Rightarrow$ treewidth is an important parameter in parameterized complexity

[Bodlaender, WG'06] [Dujmović, Eppstein, & Wood, JoDM'17] [Dujmović, Eppstein, & Wood, JoC'05] [Korach and Solel, Dis. Appl. Math.'93] . . .

Session 6, 11:00 — 12:20	**Sponsored by yWorks, Chair: Maarten Löffler, *Hemerycksalen***
11:00 — 11:20	Md. Jawaherul Alam, Michael Bekos, Martin Gronemann and Michael Kaufmann. **The Page Number of Monotone Directed Acyclic Outerplanar Graphs is Four or Five** [T1]
11:20 — 11:40	Michael Bekos, Giordano Da Lozzo, Fabrizio Frati, Giuseppe Liotta and Antonios Symvonis. **Internally-Convex Drawings of Outerplanar Graphs in Small Area** [T1]
11:40 — 12:00	Rafał Pyzik. **Treewidth of Outer k-Planar Graphs** [T1]
12:00 — 12:20	Alvin Chiu, Thomas Depian, David Eppstein, Michael T. Goodrich and Martin Nöllenburg. **Visualizing Treewidth** [T2]
12:20 — 14:00	Lunch, *Trozelli Gallery*
14:00 — 15:00	**Invited Talk**, ***Hemerycksalen*** **Prof. Dr. Hans Bodlaender**. A Sketch of Parameterized Complexity, **Chair: Vida Dujmovic**
15:00 — 15:30	**Coffee Break, *Trozelli Lounge***

Why Treewidth?

Many problems: NP-hard on general graphs, poly-time solvable on trees
(Vertex Cover, 3-Colorability, ...)

Hope: Still poly-time solvable for graphs of small (constant) treewidth
$\Rightarrow$ treewidth is an important parameter in parameterized complexity

[Bodlaender, WG'06] [Dujmović, Eppstein, & Wood, JoDM'17] [Dujmović, Eppstein, & Wood, JoC'05] [Korach and Solel, Dis. Appl. Math.'93] . . .

Session 6, 11:00 — 12:20	**Sponsored by yWorks, Chair: Maarten Löffler, *Hemerycksalen***
11:00 — 11:20	Md. Jawaherul Alam, Michael Bekos, Martin Gronemann and Michael Kaufmann. **The Page Number of Monotone Directed Acyclic Outerplanar Graphs is Four or Five** [T1]
11:20 — 11:40	Michael Bekos, Giordano Da Lozzo, Fabrizio Frati, Giuseppe Liotta and Antonios Symvonis. **Internally-Convex Drawings of Outerplanar Graphs in Small Area** [T1]
11:40 — 12:00	Rafał Pyzik. **Treewidth of Outer k-Planar Graphs** [T1]
12:00 — 12:20	Alvin Chiu, Thomas Depian, David Eppstein, Michael T. Goodrich and Martin Nöllenburg. **Visualizing Treewidth** [T2]
12:20 — 14:00	Lunch, *Trozelli Gallery*
14:00 — 15:00	**Invited Talk**, ***Hemerycksalen*** **Prof. Dr. Hans Bodlaender**. A Sketch of Parameterized Complexity, **Chair: Vida Dujmovic**
15:00 — 15:30	**Coffee Break, *Trozelli Lounge***

Why Treewidth?

Many problems: NP-hard on general graphs, poly-time solvable on trees
(Vertex Cover, 3-Colorability, ...)

Hope: Still poly-time solvable for graphs of small (constant) treewidth
$\Rightarrow$ treewidth is an important parameter in parameterized complexity

[Bodlaender, WG'06] [Dujmović, Eppstein, & Wood, JoDM'17] [Dujmović, Eppstein, & Wood, JoC'05] [Korach and Solel, Dis. Appl. Math.'93] . . .

Session 6, 11:00 — 12:20	**Sponsored by yWorks, Chair: Maarten Löffler, *Hemerycksalen***
11:00 — 11:20	Md. Jawaherul Alam, Michael Bekos, Martin Gronemann and Michael Kaufmann. **The Page Number of Monotone Directed Acyclic Outerplanar Graphs is Four or Five** [T1]
11:20 — 11:40	Michael Bekos, Giordano Da Lozzo, Fabrizio Frati, Giuseppe Liotta and Antonios Symvonis. **Internally-Convex Drawings of Outerplanar Graphs in Small Area** [T1]
11:40 — 12:00	Rafał Pyzik. **Treewidth of Outer k-Planar Graphs** [T1]
12:00 — 12:20	Alvin Chiu, Thomas Depian, David Eppstein, Michael T. Goodrich and Martin Nöllenburg. **Visualizing Treewidth** [T2]
12:20 — 14:00	Lunch, *Trozelli Gallery*
14:00 — 15:00	**Invited Talk, *Hemerycksalen*** **Prof. Dr. Hans Bodlaender**. A Sketch of Parameterized Complexity, **Chair: Vida Dujmovic**
15:00 — 15:30	**Coffee Break, *Trozelli Lounge***

?

Why Treewidth?

Many problems: NP-hard on general graphs, poly-time solvable on trees
(Vertex Cover, 3-Colorability, ...)

Hope: Still poly-time solvable for graphs of small (constant) treewidth
$\Rightarrow$ treewidth is an important parameter in parameterized complexity

[Bodlaender, WG'06] [Dujmović, Eppstein, & Wood, JoDM'17] [Dujmović, Eppstein, & Wood, JoC'05] [Korach and Solel, Dis. Appl. Math.'93] . . .

Session 6, 11:00 – 12:20	**Sponsored by yWorks, Chair: Maarten Löffler, *Hemerycksalen***
11:00 – 11:20	Md. Jawaherul Alam, Michael Bekos, Martin Gronemann and Michael Kaufmann. **The Page Number of Monotone Directed Acyclic Outerplanar Graphs is Four or Five** [T1]
11:20 – 11:40	Michael Bekos, Giordano Da Lozzo, Fabrizio Frati, Giuseppe Liotta and Antonios Symvonis. **Internally-Convex Drawings of Outerplanar Graphs in Small Area** [T1]
11:40 – 12:00	Rafał Pyzik. **Treewidth of Outer k-Planar Graphs** [T1]
12:00 – 12:20	Alvin Chiu, Thomas Depian, David Eppstein, Michael T. Goodrich and Martin Nöllenburg. **Visualizing Treewidth** [T2]
12:20 – 14:00	Lunch, *Trozelli Gallery*
14:00 – 15:00	**Invited Talk, *Hemerycksalen*** **Prof. Dr. Hans Bodlaender**. A Sketch of Parameterized Complexity, **Chair: Vida Dujmovic**
15:00 – 15:30	**Coffee Break, *Trozelli Lounge***

?

Bounding the Treewidth of Outer k-Planar Graphs via Triangulations [GD'24]

Oksana Firman
Universität Würzburg, Germany

Grzegorz Gutowski
Institute of Theoretical Computer Science, Faculty of Mathematics and Computer Science, Jagiellonian University, Kraków, Poland

Myroslav Kryven
University of Manitoba, Canada

Yuto Okada
Nagoya University, Japan

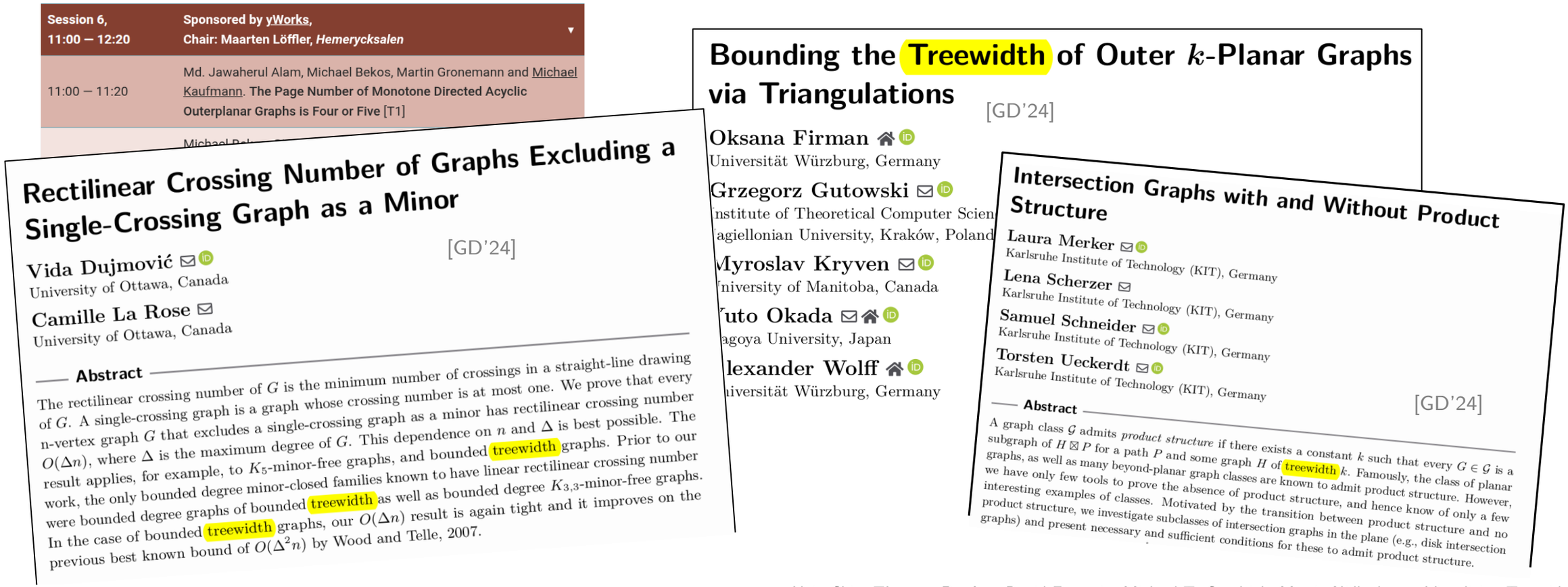
Alexander Wolff
Universität Würzburg, Germany

Why Treewidth?

Many problems: NP-hard on general graphs, poly-time solvable on trees
(Vertex Cover, 3-Colorability, ...)

Hope: Still poly-time solvable for graphs of small (constant) treewidth
$\Rightarrow$ treewidth is an important parameter in parameterized complexity

[Bodlaender, WG'06] [Dujmović, Eppstein, & Wood, JoDM'17] [Dujmović, Eppstein, & Wood, JoC'05] [Korach and Solel, Dis. Appl. Math.'93] ...



Session 6, 11:00 — 12:20 | **Sponsored by yWorks, Chair: Maarten Löffler, *Hemerycksalen***

11:00 — 11:20 | Md. Jawaherul Alam, Michael Bekos, Martin Gronemann and Michael Kaufmann. **The Page Number of Monotone Directed Acyclic Outerplanar Graphs is Four or Five** [T1]

Rectilinear Crossing Number of Graphs Excluding a Single-Crossing Graph as a Minor [GD'24]

Vida Dujmović
University of Ottawa, Canada

Camille La Rose
University of Ottawa, Canada

Abstract

The rectilinear crossing number of G is the minimum number of crossings in a straight-line drawing of G. A single-crossing graph is a graph whose crossing number is at most one. We prove that every n-vertex graph G that excludes a single-crossing graph as a minor has rectilinear crossing number $O(\Delta n)$, where Δ is the maximum degree of G. This dependence on n and Δ is best possible. The result applies, for example, to K_5-minor-free graphs, and bounded treewidth graphs. Prior to our work, the only bounded degree minor-closed families known to have linear rectilinear crossing number were bounded degree graphs of bounded treewidth as well as bounded degree $K_{3,3}$-minor-free graphs. In the case of bounded treewidth graphs, our $O(\Delta n)$ result is again tight and it improves on the previous best known bound of $O(\Delta^2 n)$ by Wood and Telle, 2007.

Bounding the Treewidth of Outer k-Planar Graphs via Triangulations [GD'24]

Oksana Firman
Universität Würzburg, Germany

Grzegorz Gutowski
nstitute of Theoretical Computer Scien
agiellonian University, Kraków, Poland

Myroslav Kryven
niversity of Manitoba, Canada

uto Okada
agoya University, Japan

lexander Wolff
niversität Würzburg, Germany

Intersection Graphs with and Without Product Structure [GD'24]

Laura Merker
Karlsruhe Institute of Technology (KIT), Germany

Lena Scherzer
Karlsruhe Institute of Technology (KIT), Germany

Samuel Schneider
Karlsruhe Institute of Technology (KIT), Germany

Torsten Ueckerdt
Karlsruhe Institute of Technology (KIT), Germany

Abstract

A graph class $\mathcal{G}$ admits *product structure* if there exists a constant k such that every $G \in \mathcal{G}$ is a subgraph of $H \boxtimes P$ for a path P and some graph H of treewidth k. Famously, the class of planar graphs, as well as many beyond-planar graph classes are known to admit product structure. However, we have only few tools to prove the absence of product structure, and hence know of only a few interesting examples of classes. Motivated by the transition between product structure and no product structure, we investigate subclasses of intersection graphs in the plane (e.g., disk intersection graphs) and present necessary and sufficient conditions for these to admit product structure.

2-Layer k-Planar Graphs
Density, Crossing Lemma, Relationships, and Pathwidth

Patrizio Angelini[1], Giordano Da Lozzo[2], Henry Förster[3(✉)], and Thomas Schneck[3]

[1] John Cabot University, Rome, Italy
pangelini@johncabot.edu
[2] Roma Tre University, Rome, Italy
giordano.dalozzo@uniroma3.it
[3] University of Tübingen, Tübingen, Germany
{foersth,schneck}@informatik.uni-tuebingen.de

[GD'24]

Abstract. The 2-layer drawing model is a well-established paradigm to visualize bipartite graphs. Several beyond-planar graph classes have been studied under this model. Surprisingly, however, the fundamental class of k-planar graphs has been considered only for $k = 1$ in this context. We provide several contributions that address this gap in the literature. First, we show tight density bounds for the classes of 2-layer k-planar graphs with $k \in \{2, 3, 4, 5\}$. Based on these results, we provide a Crossing Lemma for 2-layer k-planar graphs, which then implies a general density bound for 2-layer k-planar graphs. We prove this bound to be almost optimal with a corresponding lower bound construction. Finally, we study relationships between k-planarity and h-quasiplanarity in the 2-layer model and show that 2-layer k-planar graphs have pathwidth at most $k+1$.

The Local Queue Number of Graphs with Bounded Treewidth

[GD'20]

Laura Merker(✉) and Torsten Ueckerdt

Institute of Theoretical Informatics, Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
laura.merker@student.kit.edu, torsten.ueckerdt@kit.edu

Abstract. A queue layout of a graph G consists of a vertex ordering of G and a partition of the edges into so-called queues such that no two edges in the same queue nest, i.e., have their endpoints ordered in an ABBA-pattern. Continuing the research on local ordered covering numbers, we introduce the local queue number of a graph G as the minimum ℓ such that G admits a queue layout with each vertex having incident edges

Upward and Orthogonal Planarity are W[1]-Hard Parameterized by Treewidth

Bart M. P. Jansen[1], Liana Khazaliya[2(✉)], Philipp Kindermann[3], Giuseppe Liotta[4], Fabrizio Montecchiani[4], and Kirill Simonov[5]

[1] Eindhoven University of Technology, Eindhoven, The Netherlands
b.m.p.jansen@tue.nl
[2] Technische Universität Wien, Vienna, Austria
lkhazaliya@ac.tuwien.ac.at
[3] Universität Trier, Trier, Germany
kindermann@uni-trier.de
[4] University of Perugia, Perugia, Italy
{giuseppe.liotta,fabrizio.montecchiani}@unipg.it
[5] Hasso Plattner Institute, University of Potsdam, Potsdam, Germany

[GD'23]

Appl. Math.'93] . . .

Eppstein, Martin Gronemann and Michael

Outerplanar Graphs is Four or Five [T1]

Bounding the Treewidth of Outer k-Planar Graphs via Triangulations

[GD'24]

Oksana Firman
Universität Würzburg, Germany

Grzegorz Gutowski
nstitute of Theoretical Computer Scien
agiellonian University, Kraków, Poland

Myroslav Kryven
niversity of Manitoba, Canada

uto Okada
agoya University, Japan

lexander Wolff
niversität Würzburg, Germany

Rectilinear Crossing Number of Graphs Excluding a Single-Crossing Graph as a Minor

[GD'24]

Vida Dujmović
University of Ottawa, Canada

Camille La Rose
University of Ottawa, Canada

Abstract

The rectilinear crossing number of G is the minimum number of crossings in a straight-line drawing of G. A single-crossing graph is a graph whose crossing number is at most one. We prove that every n-vertex graph G that excludes a single-crossing graph as a minor has rectilinear crossing number $O(\Delta n)$, where Δ is the maximum degree of G. This dependence on n and Δ is best possible. The result applies, for example, to K_5-minor-free graphs, and bounded treewidth graphs. Prior to our work, the only bounded degree minor-closed families known to have linear rectilinear crossing number were bounded degree graphs of bounded treewidth as well as bounded degree $K_{3,3}$-minor-free graphs. In the case of bounded treewidth graphs, our $O(\Delta n)$ result is again tight and it improves on the previous best known bound of $O(\Delta^2 n)$ by Wood and Telle, 2007.

Intersection Graphs with and Without Product Structure

Laura Merker
Karlsruhe Institute of Technology (KIT), Germany

Lena Scherzer
Karlsruhe Institute of Technology (KIT), Germany

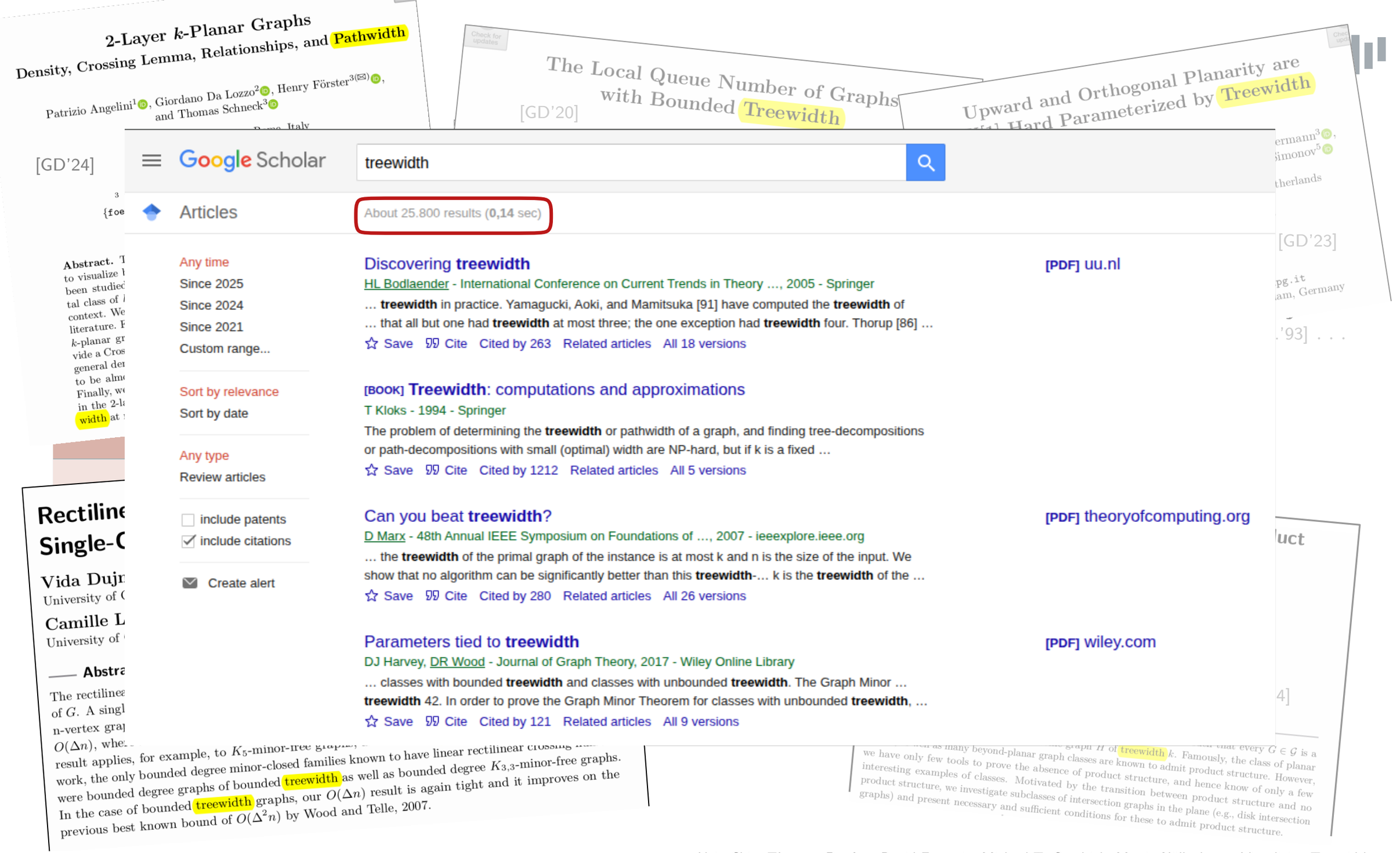
Samuel Schneider
Karlsruhe Institute of Technology (KIT), Germany

Torsten Ueckerdt
Karlsruhe Institute of Technology (KIT), Germany

[GD'24]

Abstract

A graph class $\mathcal{G}$ admits *product structure* if there exists a constant k such that every $G \in \mathcal{G}$ is a subgraph of $H \boxtimes P$ for a path P and some graph H of treewidth k. Famously, the class of planar graphs, as well as many beyond-planar graph classes are known to admit product structure. However, we have only few tools to prove the absence of product structure, and hence know of only a few interesting examples of classes. Motivated by the transition between product structure and no product structure, we investigate subclasses of intersection graphs in the plane (e.g., disk intersection graphs) and present necessary and sufficient conditions for these to admit product structure.


2-Layer k-Planar Graphs
Density, Crossing Lemma, Relationships, and Pathwidth
Patrizio Angelini[1], Giordano Da Lozzo[2], Henry Förster[3](✉), and Thomas Schneck[3]
[GD'24]
The Local Queue Number of Graphs with Bounded Treewidth
[GD'20]
Upward and Orthogonal Planarity are Hard Parameterized by Treewidth
[GD'23]
Google Scholar
treewidth
Articles
About 25.800 results (0,14 sec)
Any time
Since 2025
Since 2024
Since 2021
Custom range...
Sort by relevance
Sort by date
Any type
Review articles
include patents
include citations
Create alert
Discovering treewidth
[PDF] uu.nl
HL Bodlaender - International Conference on Current Trends in Theory ..., 2005 - Springer
... treewidth in practice. Yamagucki, Aoki, and Mamitsuka [91] have computed the treewidth of
... that all but one had treewidth at most three; the one exception had treewidth four. Thorup [86] ...
Save Cite Cited by 263 Related articles All 18 versions
[BOOK] Treewidth: computations and approximations
T Kloks - 1994 - Springer
The problem of determining the treewidth or pathwidth of a graph, and finding tree-decompositions or path-decompositions with small (optimal) width are NP-hard, but if k is a fixed ...
Save Cite Cited by 1212 Related articles All 5 versions
Can you beat treewidth?
[PDF] theoryofcomputing.org
D Marx - 48th Annual IEEE Symposium on Foundations of ..., 2007 - ieeexplore.ieee.org
... the treewidth of the primal graph of the instance is at most k and n is the size of the input. We show that no algorithm can be significantly better than this treewidth-... k is the treewidth of the ...
Save Cite Cited by 280 Related articles All 26 versions
Parameters tied to treewidth
[PDF] wiley.com
DJ Harvey, DR Wood - Journal of Graph Theory, 2017 - Wiley Online Library
... classes with bounded treewidth and classes with unbounded treewidth. The Graph Minor ...
treewidth 42. In order to prove the Graph Minor Theorem for classes with unbounded treewidth, ...
Save Cite Cited by 121 Related articles All 9 versions
Rectilinear Single-C
Vida Dujmović
Camille L
Abstract
result applies, for example, to K_5-minor-free graphs,
work, the only bounded degree minor-closed families known to have linear rectilinear crossing
were bounded degree graphs of bounded treewidth as well as bounded degree $K_{3,3}$-minor-free graphs.
In the case of bounded treewidth graphs, our $O(\Delta n)$ result is again tight and it improves on the
previous best known bound of $O(\Delta^2 n)$ by Wood and Telle, 2007.
we have only few tools to prove the absence of product structure, and hence know of only a few
interesting examples of classes. Motivated by the transition between product structure and no
product structure, we investigate subclasses of intersection graphs in the plane (e.g., disk intersection
graphs) and present necessary and sufficient conditions for these to admit product structure.

Why Visualizing Treewidth?

Explain and understand concepts and properties of graphs

[Bodlaender, WG'06] [Maniu et al., ICDT'19] [Eppstein, Notices of AMS'25]

Why Visualizing Treewidth?

Explain and understand concepts and properties of graphs

[Bodlaender, WG'06] [Maniu et al., ICDT'19] [Eppstein, Notices of AMS'25]

Wikipedia The Free Encyclopedia

Search Wikipedia | Search

Donate Create account Log in

Treewidth

7 languages

Article Talk | Read Edit View history Tools

From Wikipedia, the free encyclopedia

In graph theory, the **treewidth** of an undirected graph is an integer number which specifies, informally, how far the graph is from being a tree. The smallest treewidth is 1; the graphs with treewidth 1 are exactly the trees and the forests. An example of graphs with treewidth at most 2 are the series–parallel graphs. The maximal graphs with treewidth exactly k are called *k-trees*, and the graphs with treewidth at most k are called *partial k-trees*.[1] Many other well-studied graph families also have bounded treewidth.

Treewidth may be formally defined in several equivalent ways: in terms of the size of the largest vertex set in a tree decomposition of the graph, in terms of the size of the largest clique in a chordal completion of the graph, in terms of the maximum order of a haven describing a strategy for a pursuit–evasion game on the graph, or in terms of the maximum order of a bramble, a collection of connected subgraphs that all touch each other.

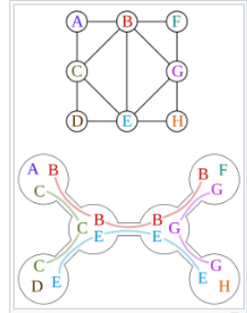
Treewidth is commonly used as a parameter in the parameterized complexity analysis of graph algorithms. Many algorithms that are NP-hard for general graphs, become easier when the treewidth is bounded by a constant.

The concept of treewidth was originally introduced by Umberto Bertelè and Francesco Brioschi (1972) under the name of *dimension*. It was later rediscovered by Rudolf Halin (1976), based on properties that it shares with a different graph parameter, the Hadwiger number. Later it was again rediscovered by Neil Robertson and Paul Seymour (1984) and has since been studied by many other authors.[2]

Definition [edit]

A tree decomposition of a graph $G = (V, E)$ is a tree T in which each node is associated with a subset of vertices called a "bag". (The term *node* is used to refer to a vertex of T to avoid confusion with vertices of G). The bags $X_1, \dots X_t$ must satisfy the following properties:[3]

1. Each graph vertex is contained in at least one bag: $\bigcup_i X_i = V$
2. If bags X_i and X_j both contain a vertex v, then all bags X_k associated with nodes in the (unique) path of T between X_i and X_j also contain v as well. Equivalently, the bags containing vertex v are associated with a connected subtree of T.
3. For every edge (v, w) in the graph, there is at least one bag X_i that contains both v and w. That is, vertices are adjacent in the graph only when their corresponding subtrees have a node in common. (However, two vertices may belong to a bag without being adjacent to each other.)



A graph with eight vertices, and a tree decomposition of it onto a tree with six nodes. Each graph edge connects two vertices that are listed together at some tree node,

The *width* of a tree decomposition is the size of its largest bag X_i minus one. The **treewidth** $\mathrm{tw}(G)$ of a graph G is the minimum width among all possible tree decompositions of G. In this definition, the size of the largest set is diminished by one in order to make the treewidth of a tree equal to one.

Equivalently, the treewidth of G is one less than the size of the largest clique in the chordal graph containing G with the smallest clique number. A chordal graph with this clique size may be obtained by adding to G an edge between every two vertices for which at least one bag contains both vertices.

Treewidth may also be characterized in terms of havens, functions describing an evasion strategy for a certain pursuit–evasion game defined on a graph. A graph G has treewidth k if and only if it has a haven of order $k + 1$ but of no higher order, where a haven of order $k + 1$ is a function β that maps each set X of at most k vertices in G into one of the connected components of $G \setminus X$ and that obeys the monotonicity property that $\beta(Y) \subset \beta(X)$ whenever $X \subset Y$

Why Visualizing Treewidth?

Explain and understand concepts and properties of graphs

[Bodlaender, WG'06] [Maniu et al., ICDT'19] [Eppstein, Notices of AMS'25]

WIKIPEDIA
The Free Encyclopedia

Search Wikipedia | Search

Donate | Create account | Log in

Treewidth

7 languages

Article | Talk | Read | Edit | View history | Tools

From Wikipedia, the free encyclopedia

In graph theory, the **treewidth** of an undirected graph is an integer number which specifies, informally, how far the graph is from being a tree. The smallest treewidth is 1; the graphs with treewidth 1 are exactly the trees and the forests. An example of graphs with treewidth at most 2 are the series–parallel graphs. The maximal graphs with treewidth exactly k are called *k-trees*, and the graphs with treewidth at most k are called *partial k-trees*.[1] Many other well-studied graph families also have bounded treewidth.

Treewidth may be formally defined in several equivalent ways: in terms of the size of the largest vertex set in a tree decomposition of the graph, in terms of the size of the largest clique in a chordal completion of the graph, in terms of the maximum order of a haven describing a strategy for a pursuit–evasion game on the graph, or in terms of the maximum order of a bramble, a collection of connected subgraphs that all touch each other.

Treewidth is commonly used as a parameter in the parameterized complexity analysis of graph algorithms. Many algorithms that are NP-hard for general graphs, become easier when the treewidth is bounded by a constant.

The concept of treewidth was originally introduced by Umberto Bertelè and Francesco Brioschi (1972) under the name of *dimension*. It was later rediscovered by Rudolf Halin (1976), based on properties that it shares with a different graph parameter, the Hadwiger number. Later it was again rediscovered by Neil Robertson and Paul Seymour (1984) and has since been studied by many other authors.[2]

Definition [edit]

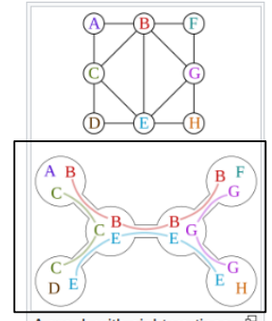
A tree decomposition of a graph $G = (V, E)$ is a tree T in which each node is associated with a subset of vertices called a "bag". (The term *node* is used to refer to a vertex of T to avoid confusion with vertices of G). The bags $X_1, \dots X_t$ must satisfy the following properties:[3]

1. Each graph vertex is contained in at least one bag: $\bigcup_i X_i = V$
2. If bags X_i and X_j both contain a vertex v, then all bags X_k associated with nodes in the (unique) path of T between X_i and X_j also contain v as well. Equivalently, the bags containing vertex v are associated with a connected subtree of T.
3. For every edge (v, w) in the graph, there is at least one bag X_i that contains both v and w. That is, vertices are adjacent in the graph only when their corresponding subtrees have a node in common. (However, two vertices may belong to a bag without being adjacent to each other.)

The *width* of a tree decomposition is the size of its largest bag X_i minus one. The **treewidth** $\mathrm{tw}(G)$ of a graph G is the minimum width among all possible tree decompositions of G. In this definition, the size of the largest set is diminished by one in order to make the treewidth of a tree equal to one.

Equivalently, the treewidth of G is one less than the size of the largest clique in the chordal graph containing G with the smallest clique number. A chordal graph with this clique size may be obtained by adding to G an edge between every two vertices for which at least one bag contains both vertices.

Treewidth may also be characterized in terms of havens, functions describing an evasion strategy for a certain pursuit–evasion game defined on a graph. A graph G has treewidth k if and only if it has a haven of order $k + 1$ but of no higher order, where a haven of order $k + 1$ is a function β that maps each set X of at most k vertices in G into one of the connected components of $G \setminus X$ and that obeys the monotonicity property that $\beta(Y) \subset \beta(X)$ whenever $X \subset Y$

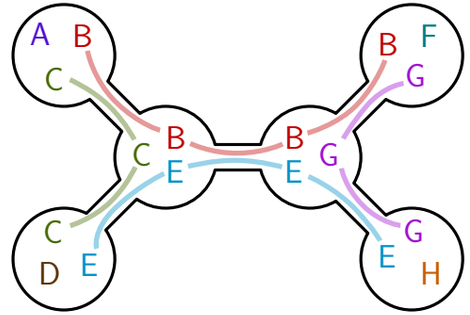


A graph with eight vertices, and a tree decomposition of it onto a tree with six nodes. Each graph edge connects two vertices that are listed together at some tree node,

Why Visualizing Treewidth?

Explain and understand concepts and properties of graphs

[Bodlaender, WG'06] [Maniu et al., ICDT'19] [Eppstein, Notices of AMS'25]

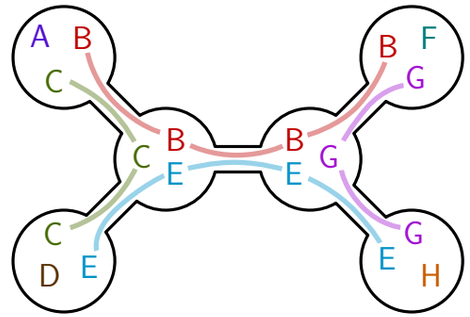


[Eppstein, Wikipedia'10]

Why Visualizing Treewidth?

Explain and understand concepts and properties of graphs

[Bodlaender, WG'06] [Maniu et al., ICDT'19] [Eppstein, Notices of AMS'25]



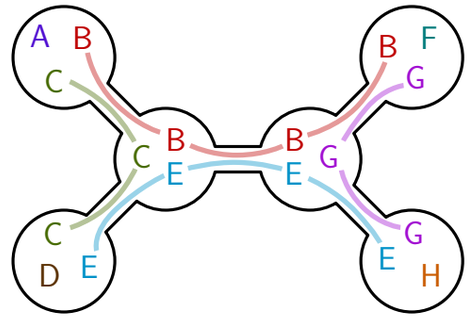
[Eppstein, Wikipedia'10]

Provide a **visual proof** for bounded treewidth graphs

Why Visualizing Treewidth?

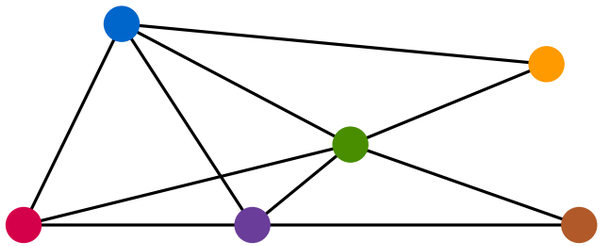
Explain and understand concepts and properties of graphs

[Bodlaender, WG'06] [Maniu et al., ICDT'19] [Eppstein, Notices of AMS'25]



[Eppstein, Wikipedia'10]

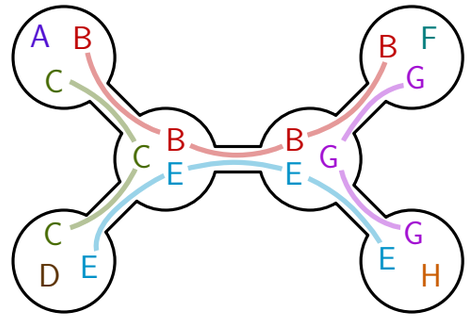
Provide a **visual proof** for bounded treewidth graphs



Why Visualizing Treewidth?

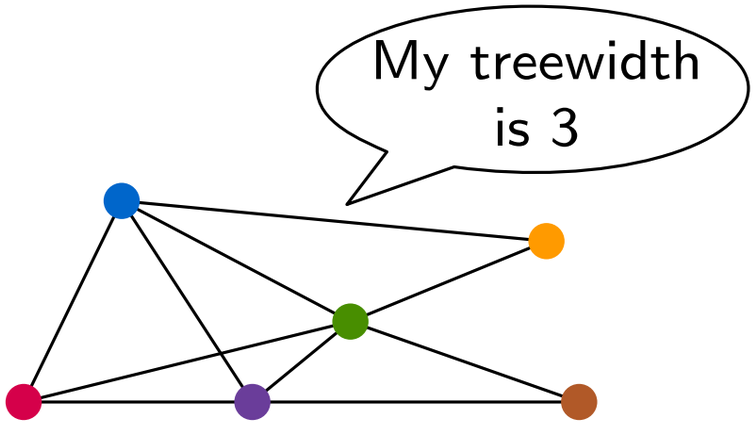
Explain and understand concepts and properties of graphs

[Bodlaender, WG'06] [Maniu et al., ICDT'19] [Eppstein, Notices of AMS'25]



[Eppstein, Wikipedia'10]

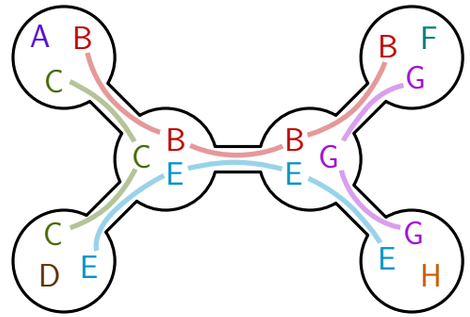
Provide a **visual proof** for bounded treewidth graphs



Why Visualizing Treewidth?

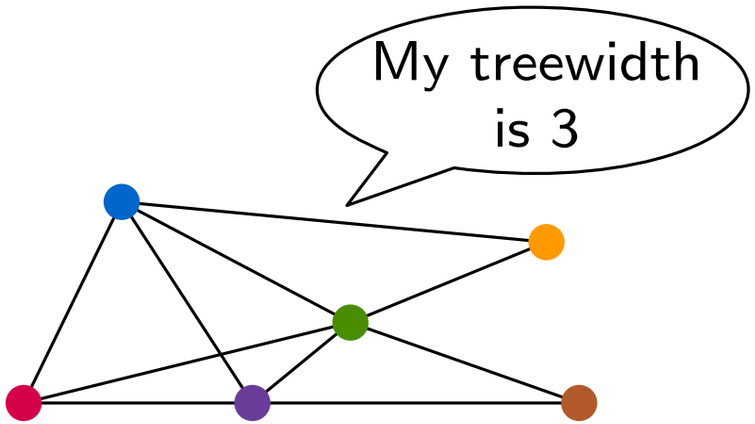
Explain and understand concepts and properties of graphs

[Bodlaender, WG'06] [Maniu et al., ICDT'19] [Eppstein, Notices of AMS'25]



[Eppstein, Wikipedia'10]

Provide a **visual proof** for bounded treewidth graphs

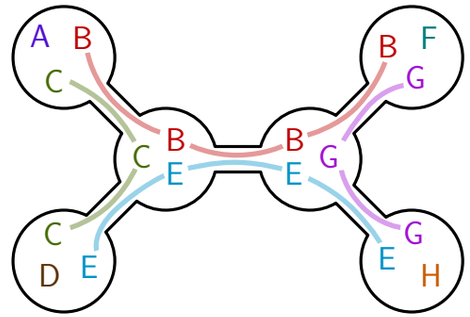


$$T = (\{ V_1 = \{\bullet\bullet\}, V_2 = \{\bullet\bullet\bullet\}, V_3 = \{\bullet\bullet\bullet\bullet\}, V_4 = \{\bullet\bullet\bullet\}\}, \{V_1V_2, V_1V_3, V_3V_4\})$$

Why Visualizing Treewidth?

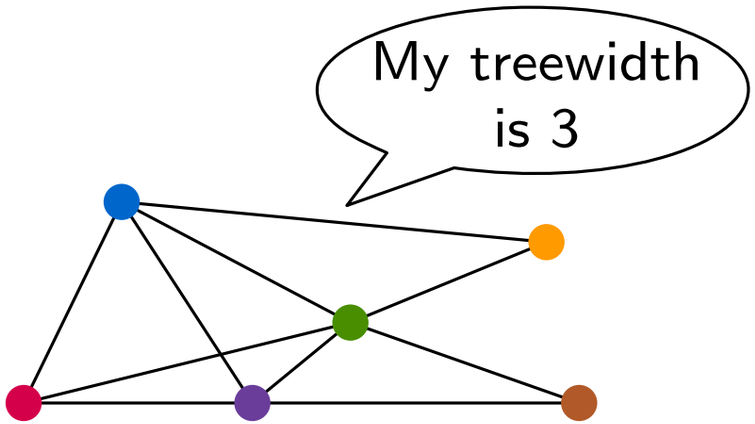
Explain and understand concepts and properties of graphs

[Bodlaender, WG'06] [Maniu et al., ICDT'19] [Eppstein, Notices of AMS'25]



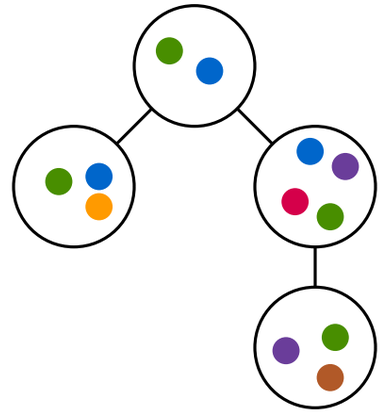
[Eppstein, Wikipedia'10]

Provide a **visual proof** for bounded treewidth graphs



$$T = (\{ V_1 = \{\bullet\bullet\}, V_2 = \{\bullet\bullet\bullet\}, V_3 = \{\bullet\bullet\bullet\bullet\}, V_4 = \{\bullet\bullet\bullet\}\}, \{V_1V_2, V_1V_3, V_3V_4\})$$

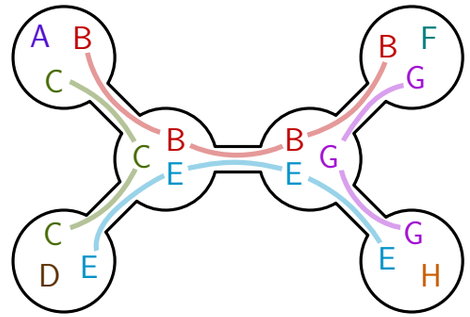
$\Leftrightarrow$



Why Visualizing Treewidth?

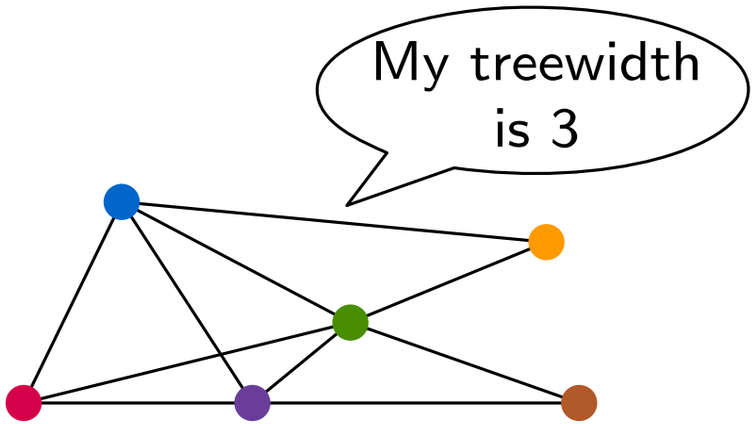
Explain and understand concepts and properties of graphs

[Bodlaender, WG'06] [Maniu et al., ICDT'19] [Eppstein, Notices of AMS'25]



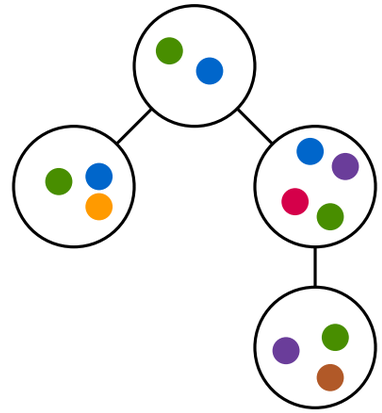
[Eppstein, Wikipedia'10]

Provide a **visual proof** for bounded treewidth graphs



$$T = (\{ V_1 = \{\bullet\bullet\}, V_2 = \{\bullet\bullet\bullet\}, V_3 = \{\bullet\bullet\bullet\bullet\}, V_4 = \{\bullet\bullet\bullet\}\}, \{V_1V_2, V_1V_3, V_3V_4\})$$

$\Leftrightarrow$

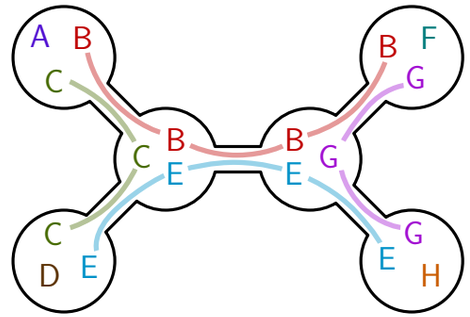


$\Rightarrow$ GraphTrials Framework [Förster et al., GD'24]

Why Visualizing Treewidth?

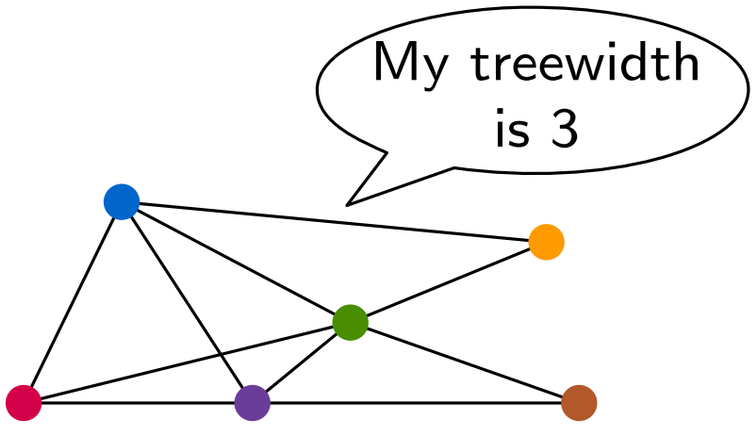
Explain and understand concepts and properties of graphs

[Bodlaender, WG'06] [Maniu et al., ICDT'19] [Eppstein, Notices of AMS'25]



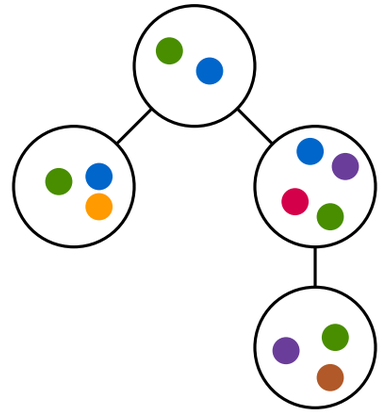
[Eppstein, Wikipedia'10]

Provide a **visual proof** for bounded treewidth graphs



$$T = (\{ V_1 = \{\bullet\bullet\}, V_2 = \{\bullet\bullet\bullet\}, V_3 = \{\bullet\bullet\bullet\bullet\}, V_4 = \{\bullet\bullet\bullet\}\}, \{V_1V_2, V_1V_3, V_3V_4\})$$

$\Leftrightarrow$



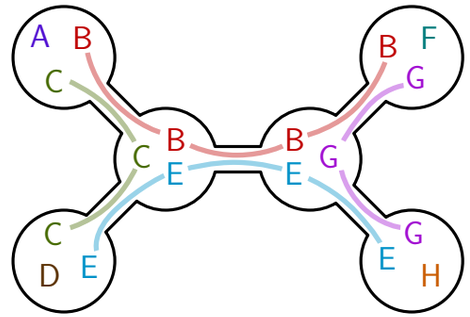
$\Rightarrow$ GraphTrials Framework [Förster et al., GD'24]

Algorithmic problem, related to crossing minimization in metro maps, book drawings, ...

[Argyriou et al., JGAA'10]
[Klawitter et al., GD'18]

Our Contributions

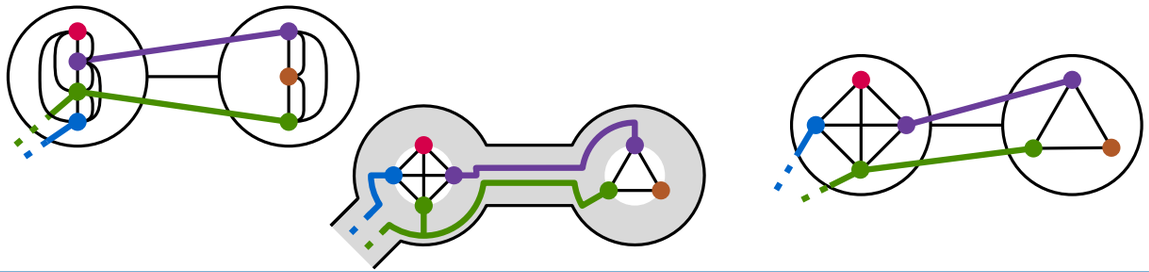
Drawing Paradigms for Treewidth

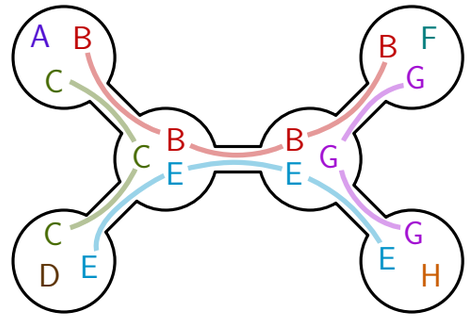


[Eppstein, Wikipedia'10]

Our Contributions

Drawing Paradigms for Treewidth

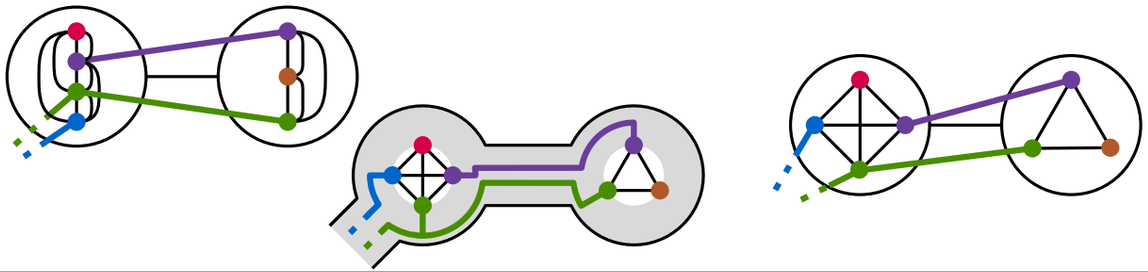


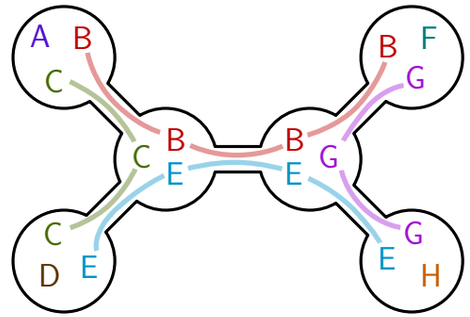


[Eppstein, Wikipedia'10]

Our Contributions

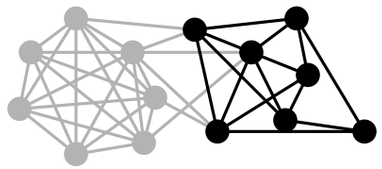
Drawing Paradigms for Treewidth

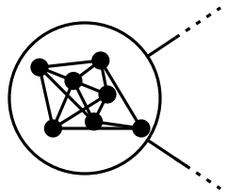




[Eppstein, Wikipedia'10]

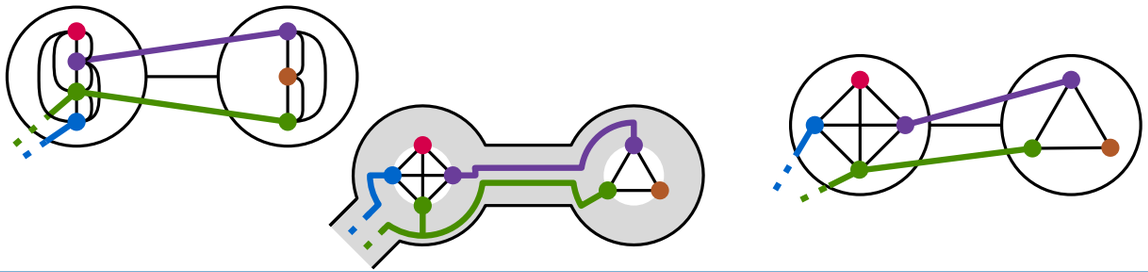
Complexity and Algorithms for Minimizing Crossings

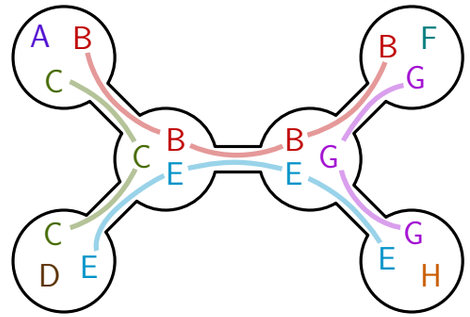




Our Contributions

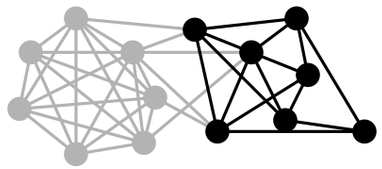
Drawing Paradigms for Treewidth

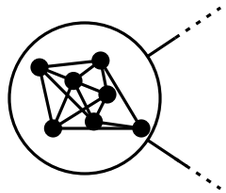




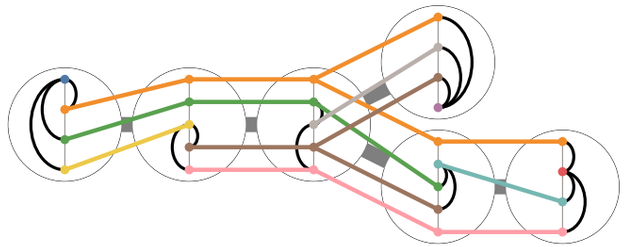
[Eppstein, Wikipedia'10]

Complexity and Algorithms for Minimizing Crossings



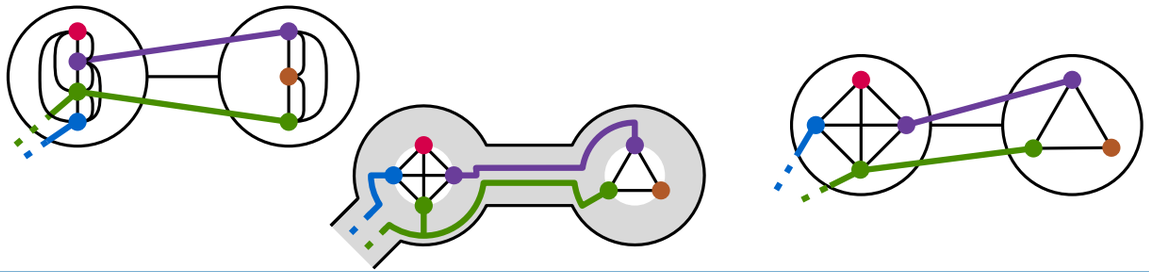


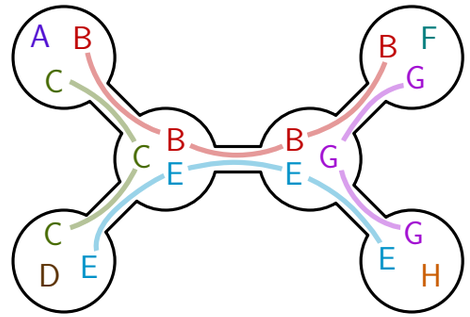
Experimental Evaluation



Our Contributions

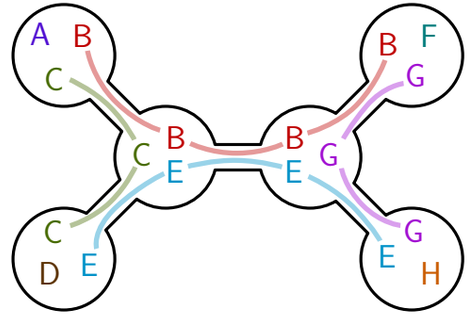
Drawing Paradigms for Treewidth





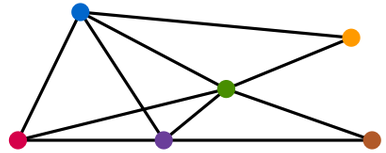
[Eppstein, Wikipedia'10]

Witness Drawing Styles



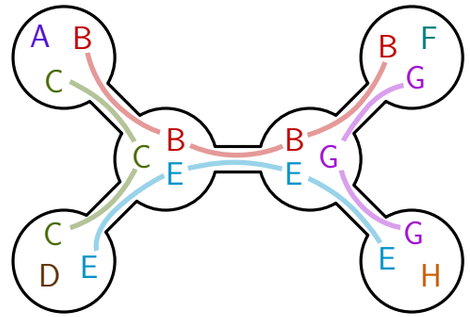
[Eppstein, Wikipedia'10]

Witness Drawing Styles



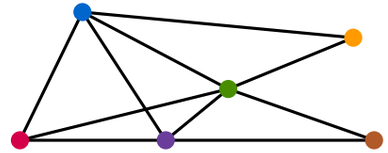
Tree T where bags are subsets of V s.t.

1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree



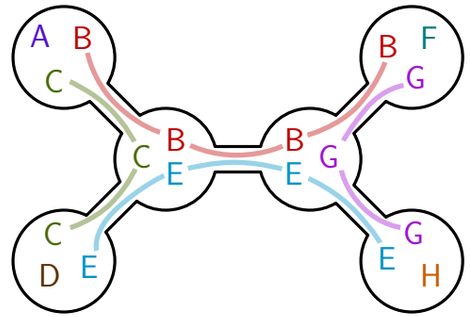
[Eppstein, Wikipedia'10]

Witness Drawing Styles

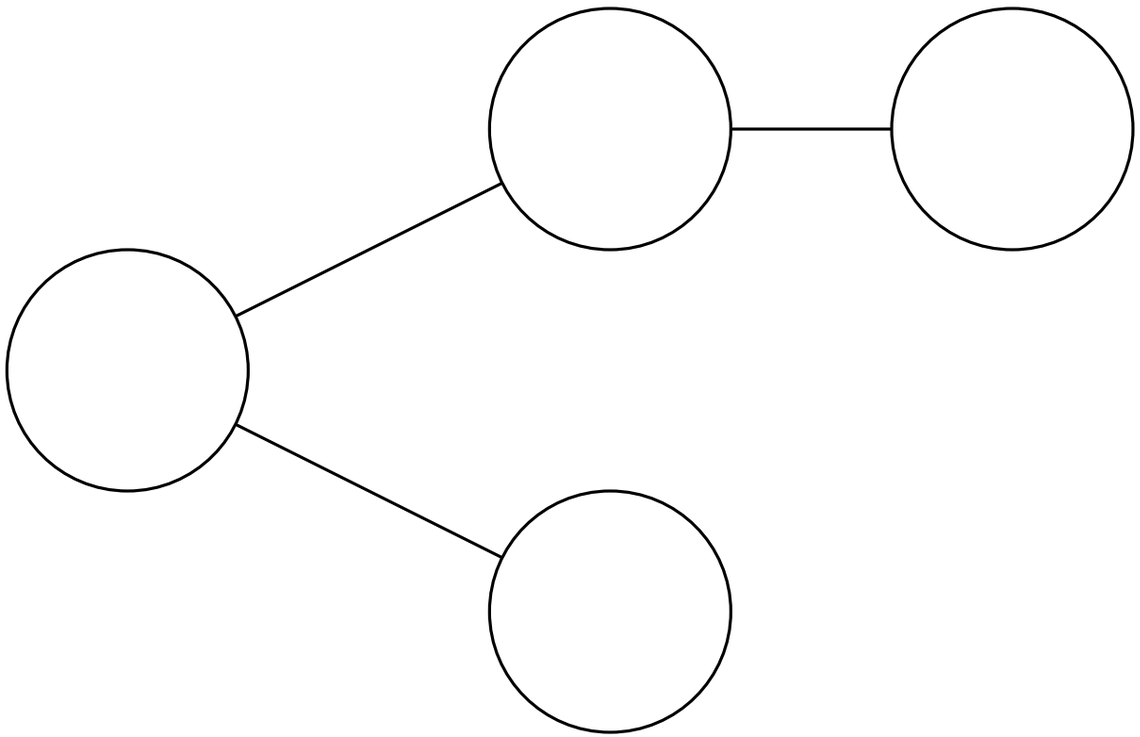


Tree T where bags are subsets of V s.t.

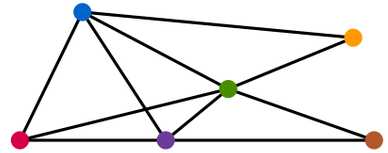
1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree



[Eppstein, Wikipedia'10]

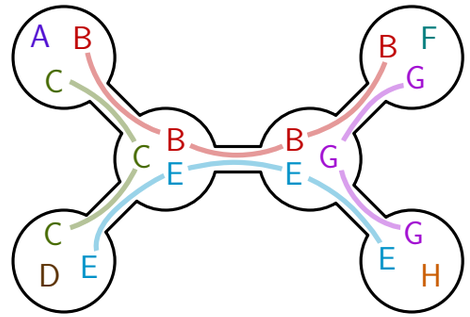


Witness Drawing Styles

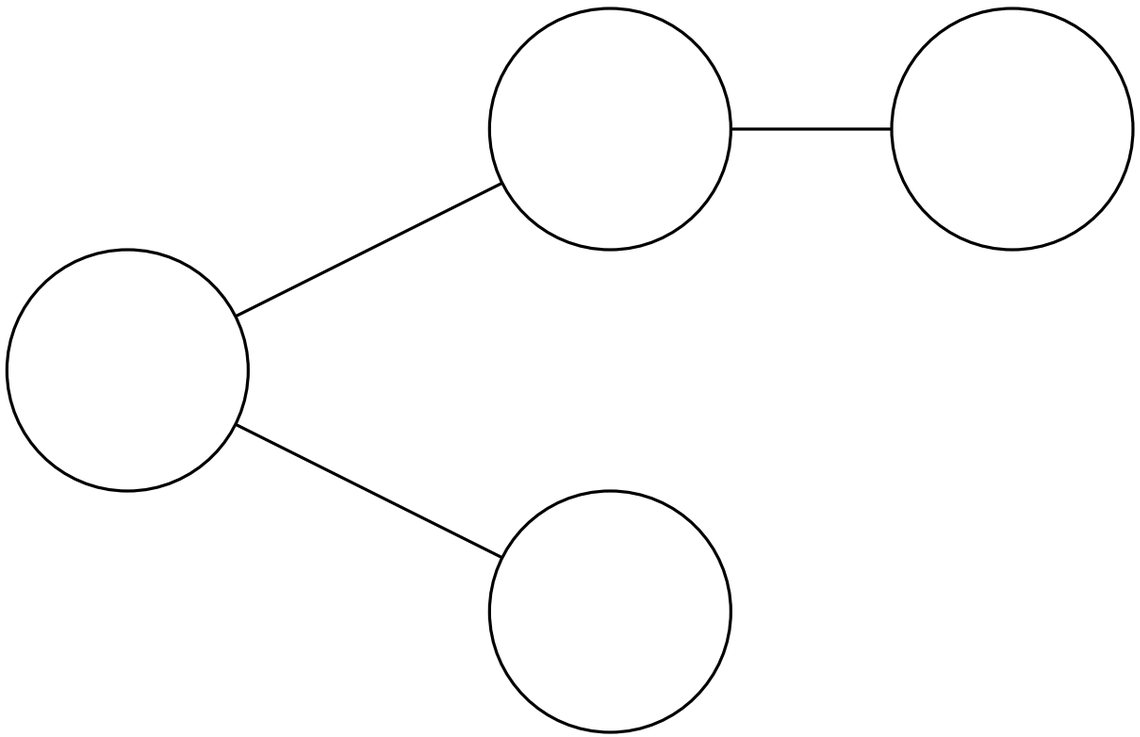


Tree T where bags are subsets of V s.t.

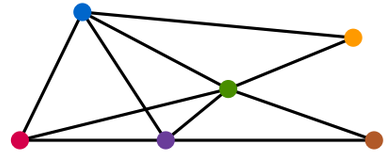
1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree



[Eppstein, Wikipedia'10]

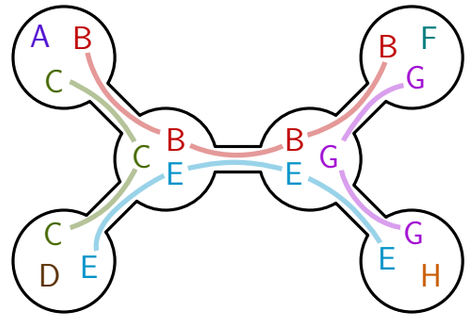


Witness Drawing Styles

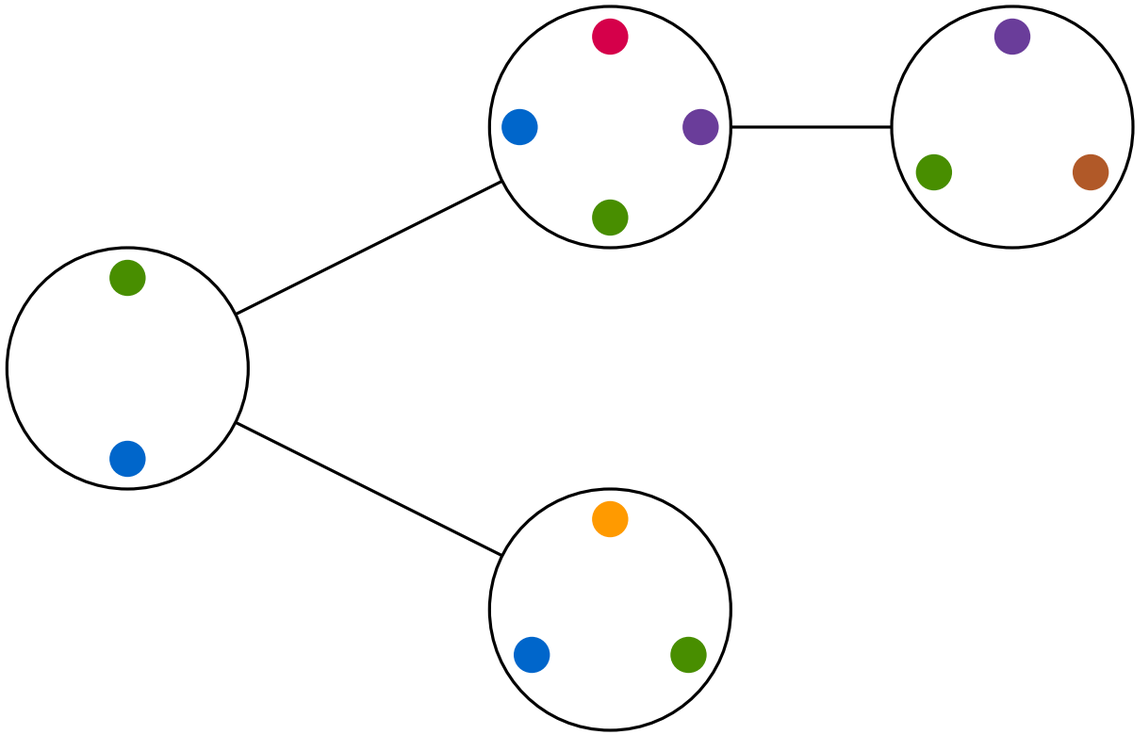


Tree T where bags are subsets of V s.t.

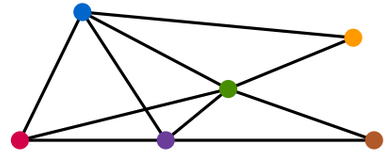
1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree



[Eppstein, Wikipedia'10]

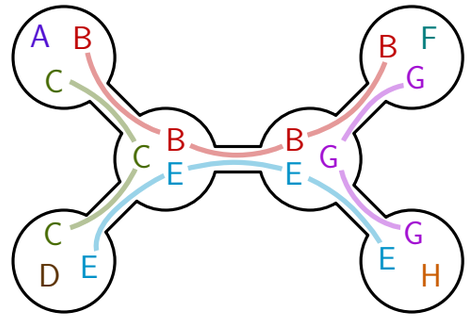


Witness Drawing Styles

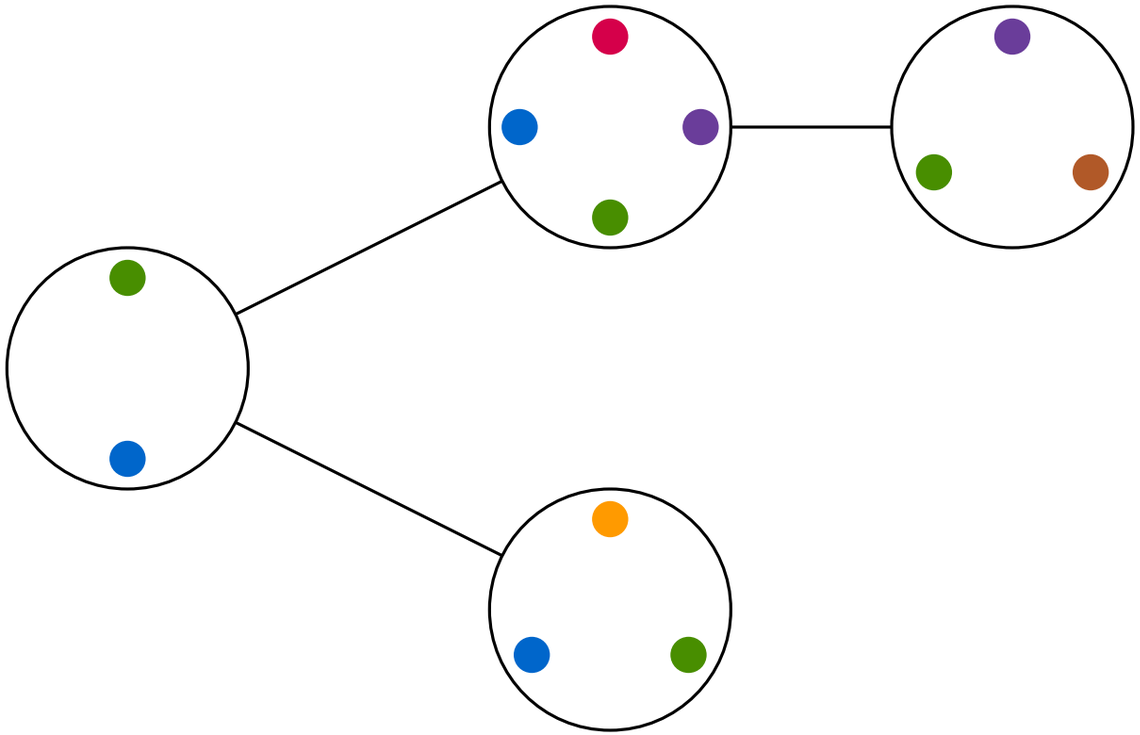


Tree T where bags are subsets of V s.t.

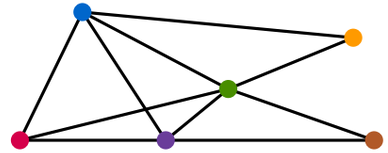
1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree



[Eppstein, Wikipedia'10]

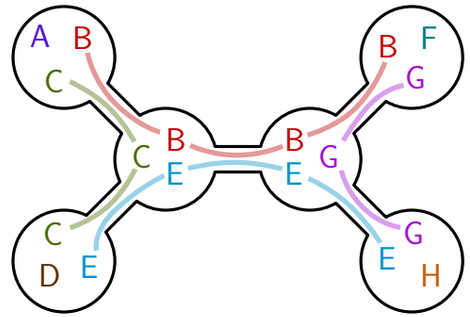


Witness Drawing Styles

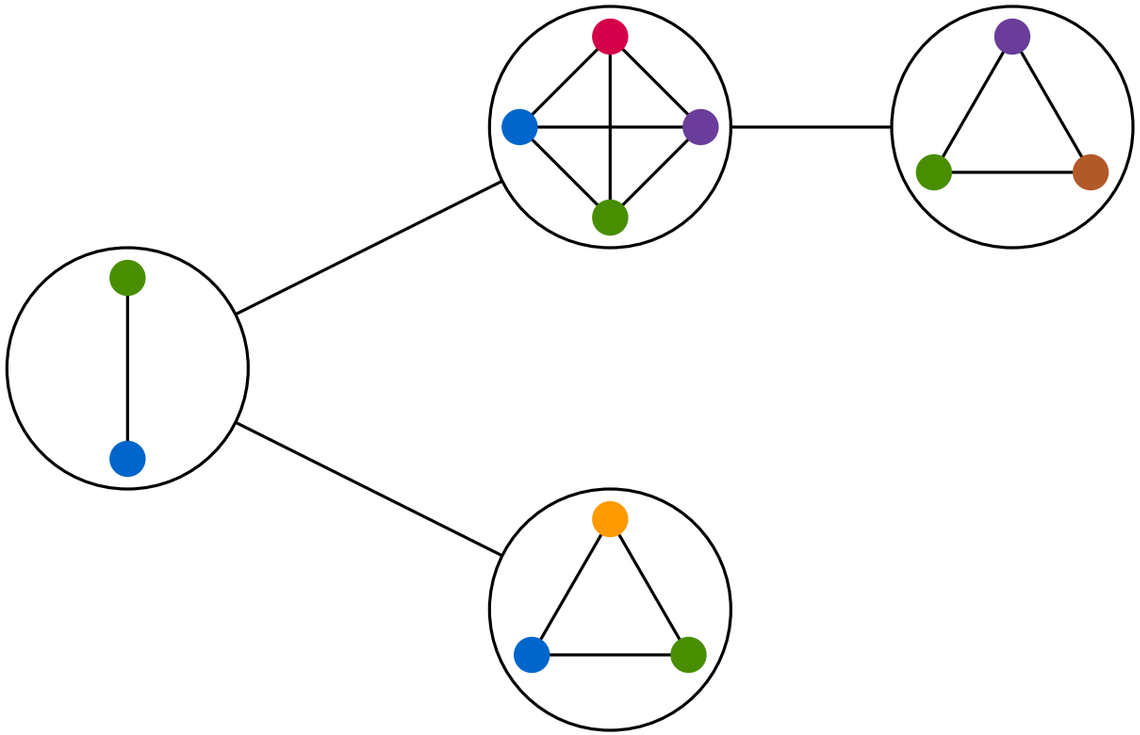


Tree T where bags are subsets of V s.t.

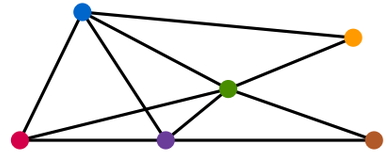
1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree



[Eppstein, Wikipedia'10]

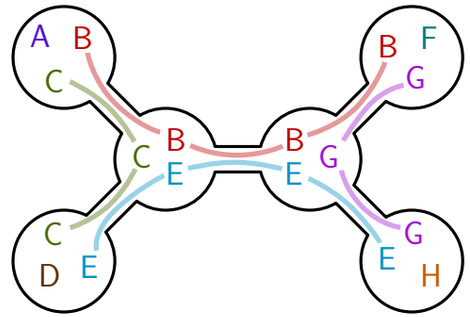


Witness Drawing Styles

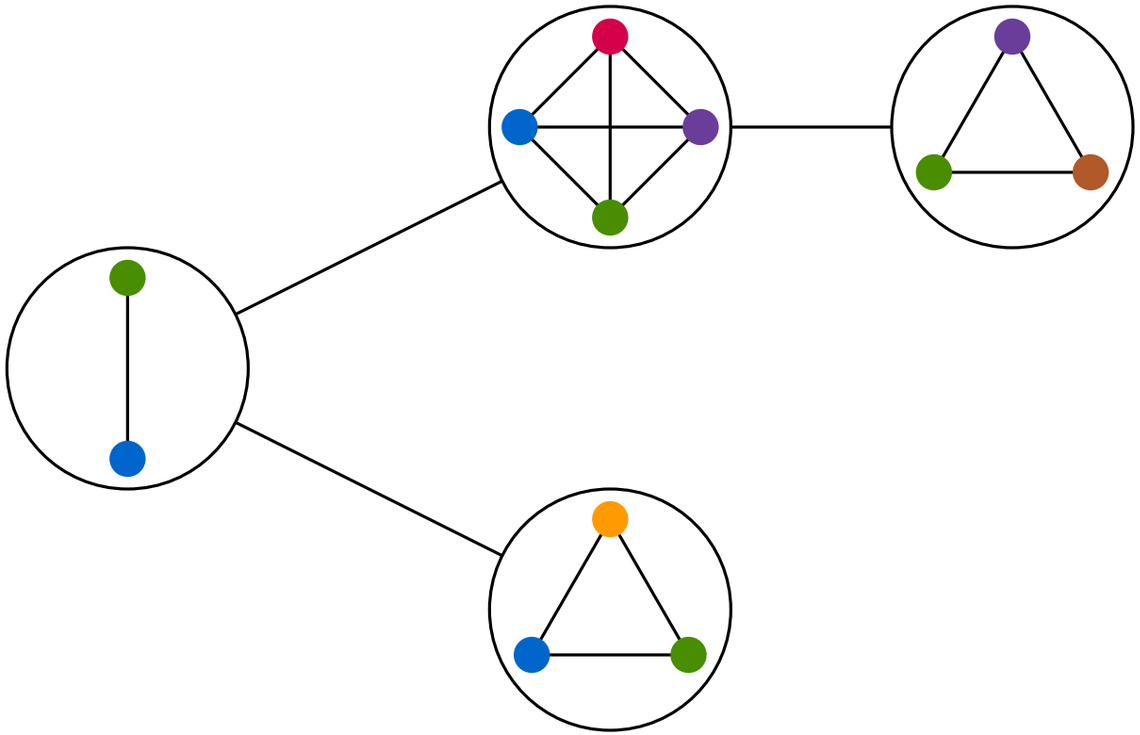


Tree T where bags are subsets of V s.t.

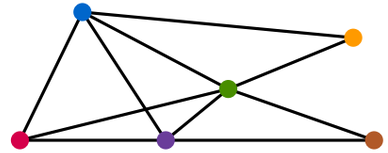
1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree



[Eppstein, Wikipedia'10]

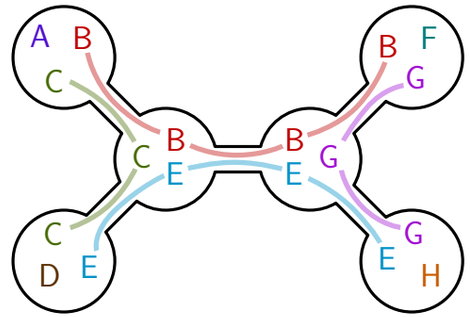


Witness Drawing Styles

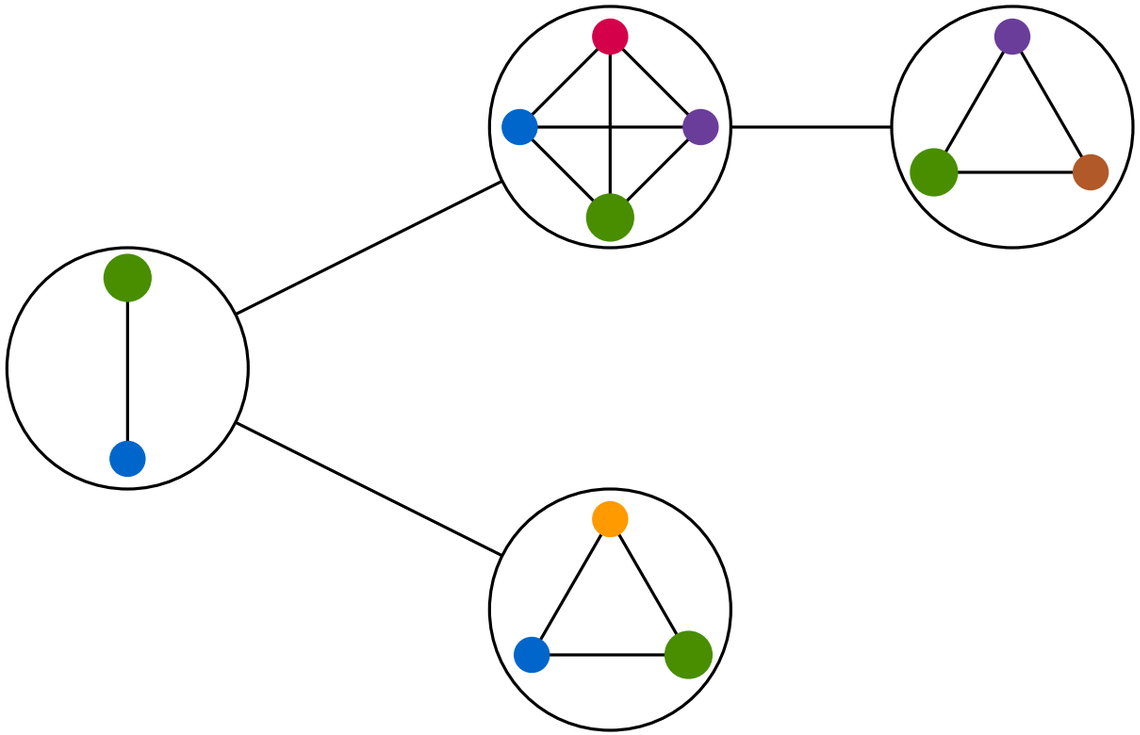


Tree T where bags are subsets of V s.t.

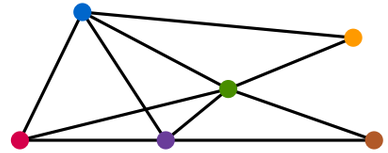
1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree



[Eppstein, Wikipedia'10]

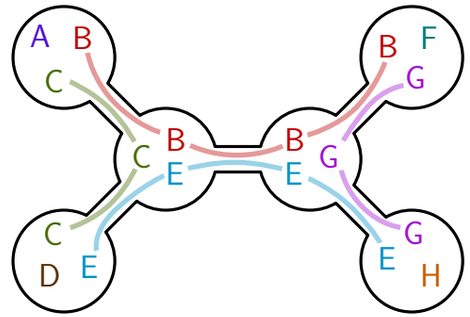


Witness Drawing Styles

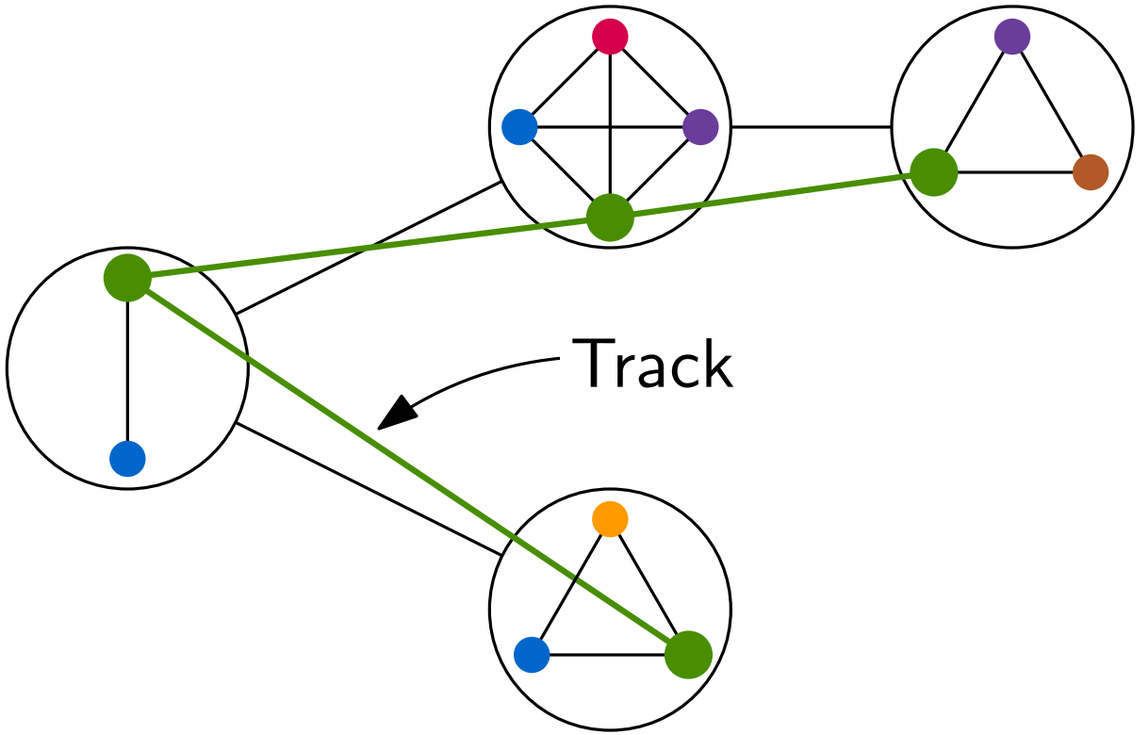


Tree T where bags are subsets of V s.t.

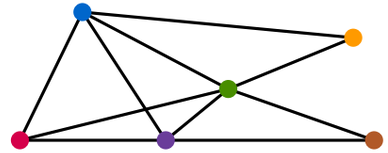
1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree



[Eppstein, Wikipedia'10]

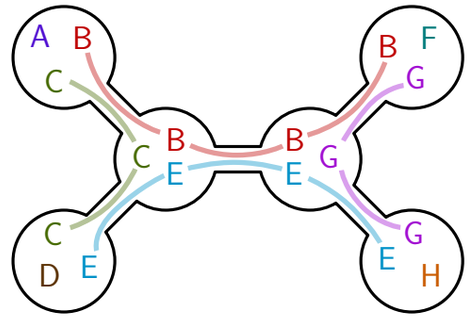


Witness Drawing Styles

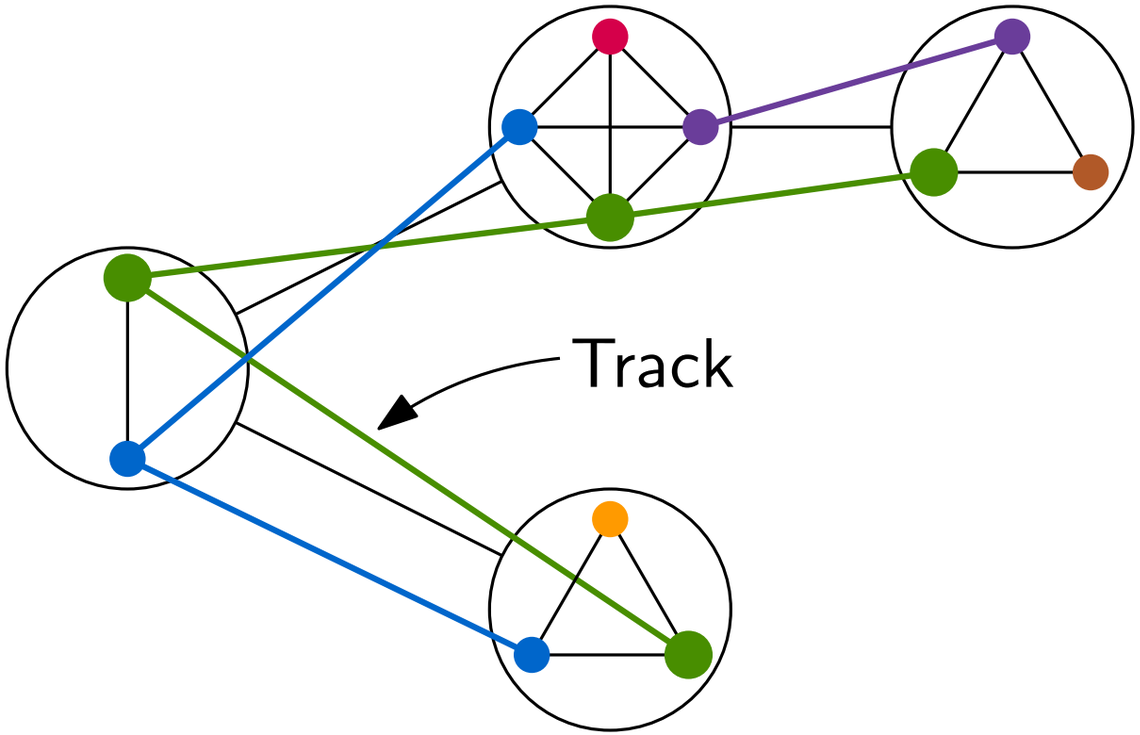


Tree T where bags are subsets of V s.t.

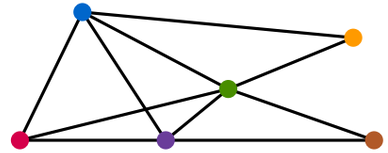
1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree



[Eppstein, Wikipedia'10]

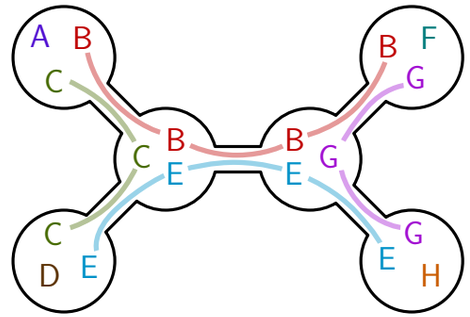


Witness Drawing Styles

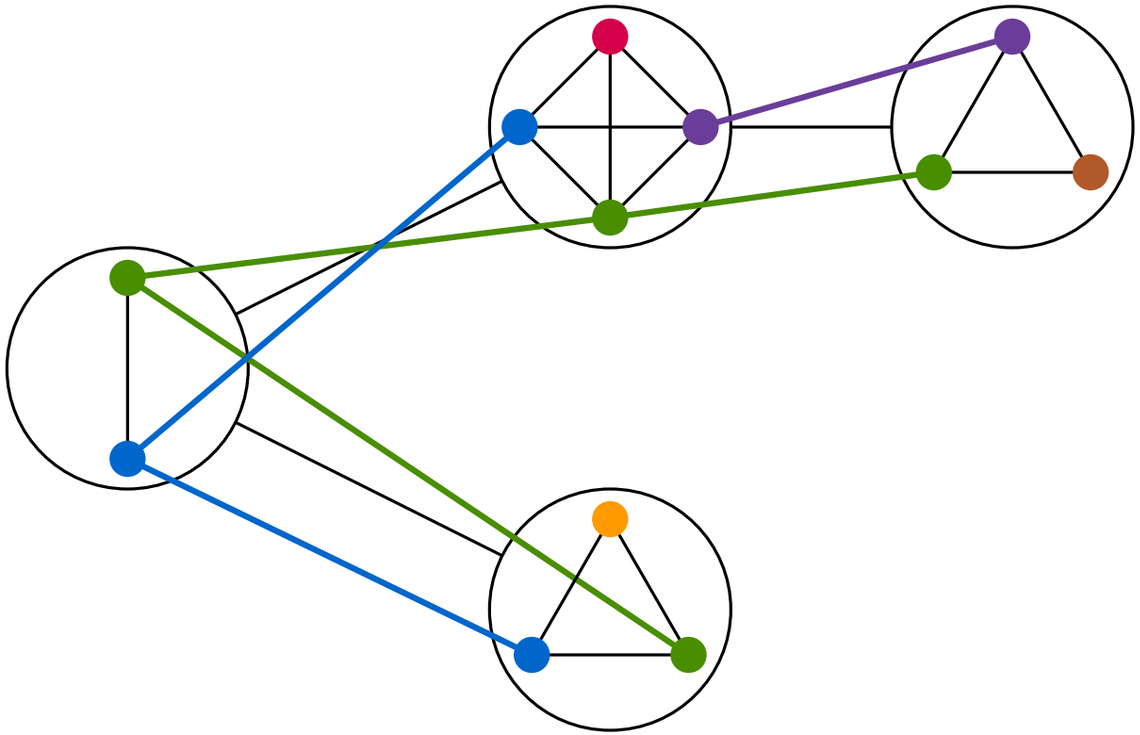


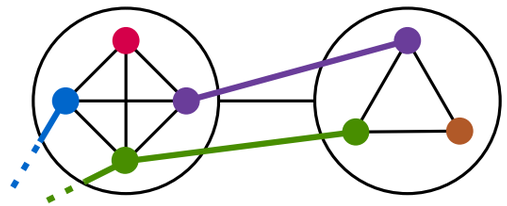
Tree T where bags are subsets of V s.t.

1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree



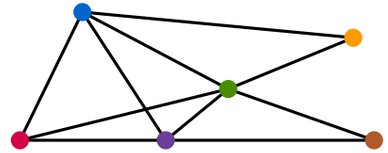
[Eppstein, Wikipedia'10]





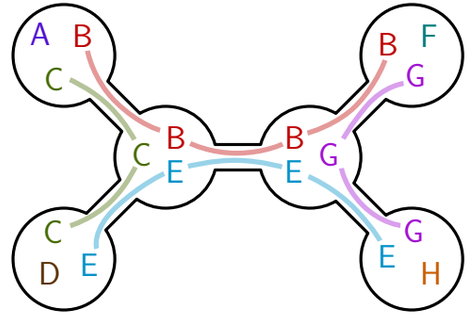
Circular

Witness Drawing Styles

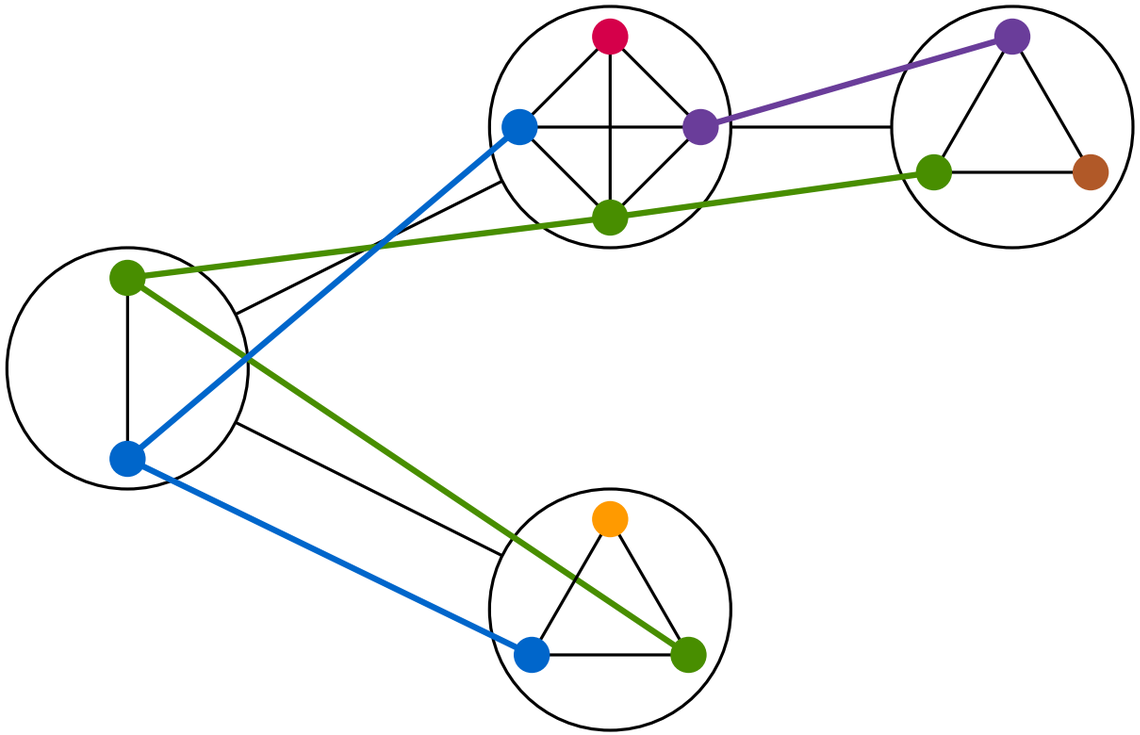


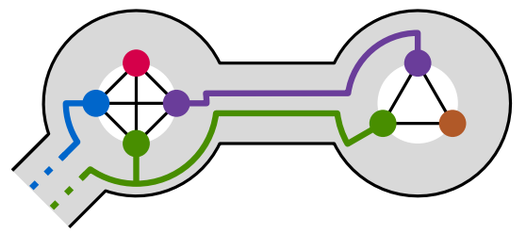
Tree T where bags are subsets of V s.t.

1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree

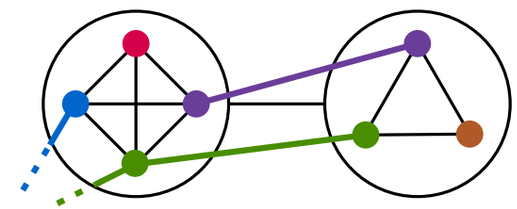


[Eppstein, Wikipedia'10]



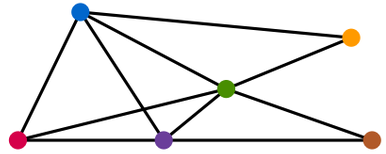


Orbital



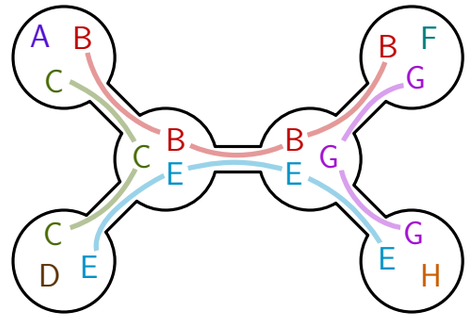
Circular

Witness Drawing Styles

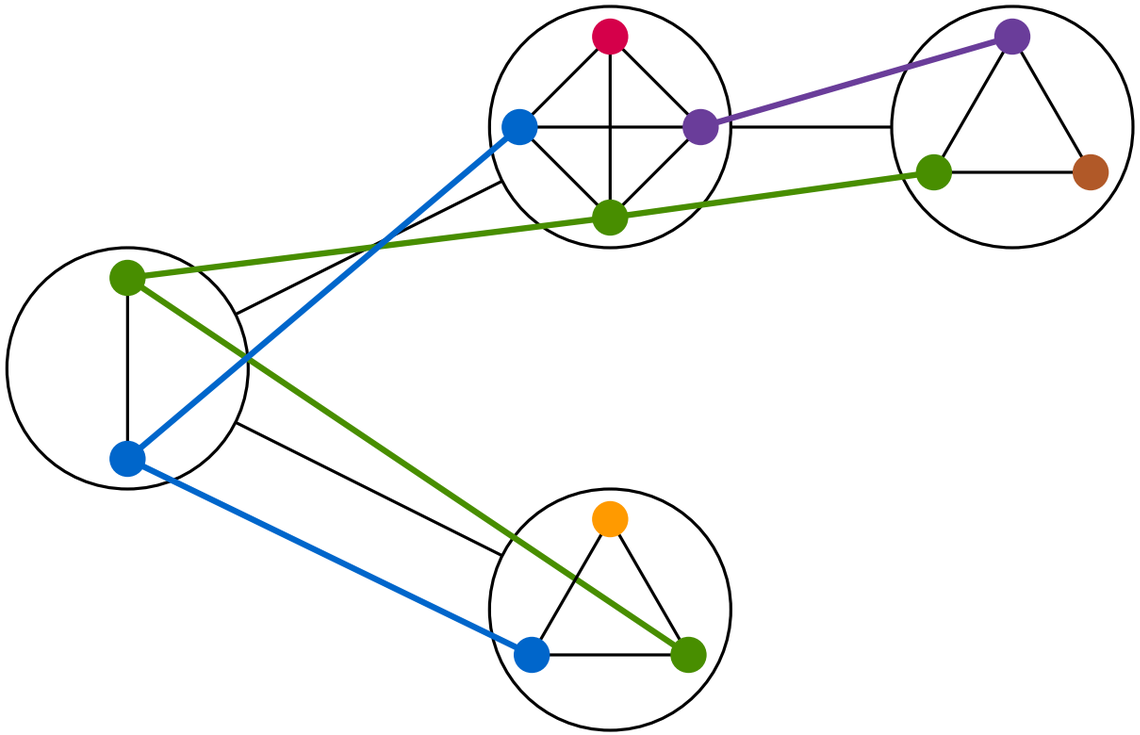


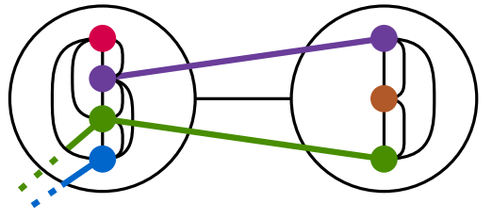
Tree T where bags are subsets of V s.t.

1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree

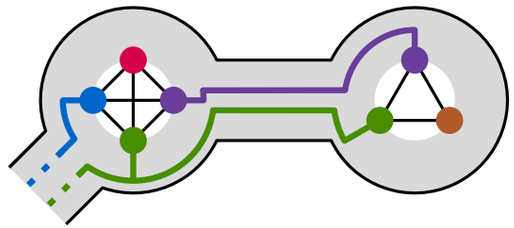


[Eppstein, Wikipedia'10]

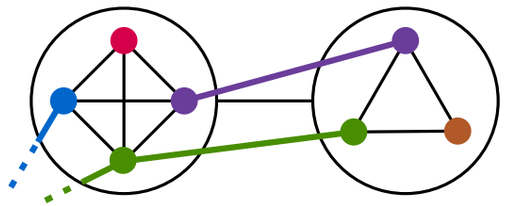




Linear

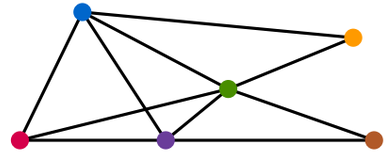


Orbital



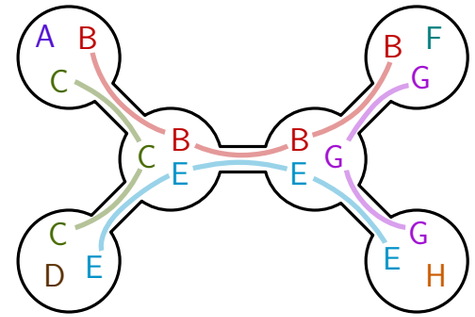
Circular

Witness Drawing Styles

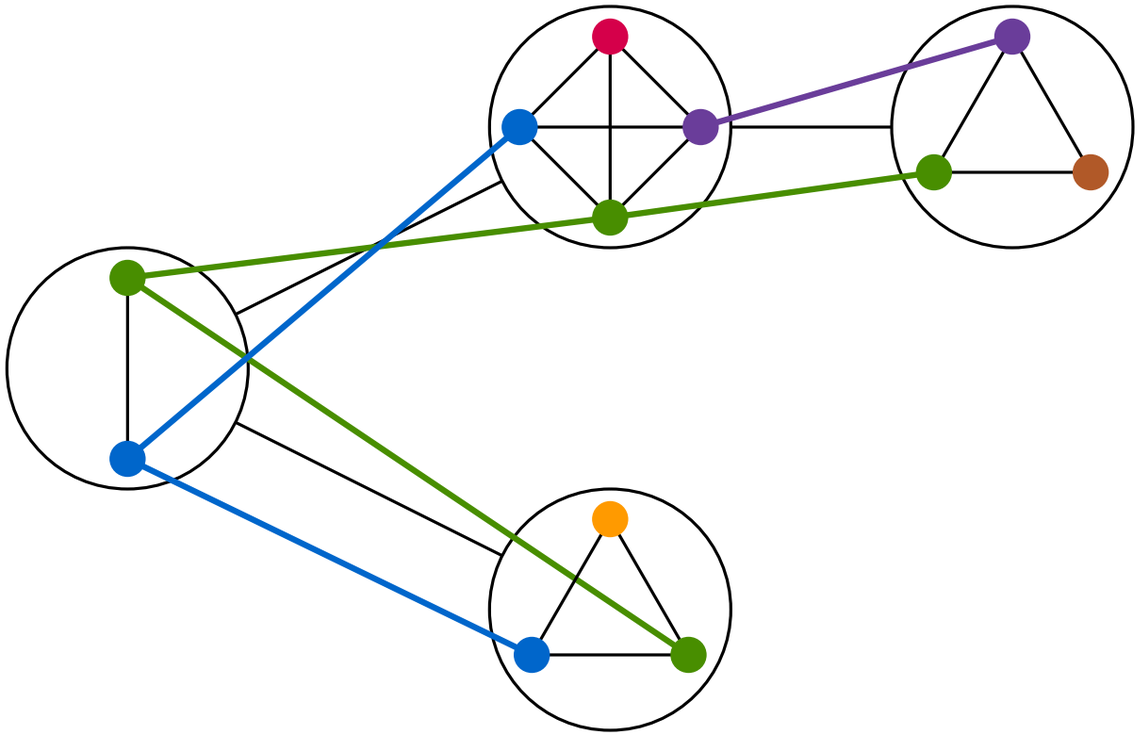


Tree T where bags are subsets of V s.t.

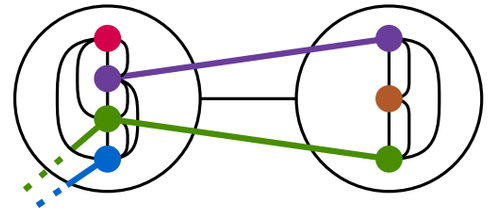
1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree



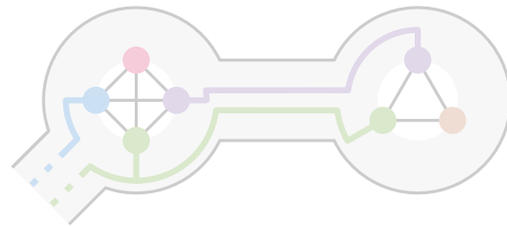
[Eppstein, Wikipedia'10]



Linear

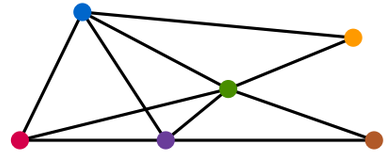


Orbital



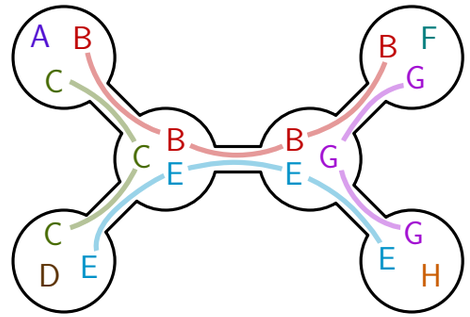
Circular

Witness Drawing Styles

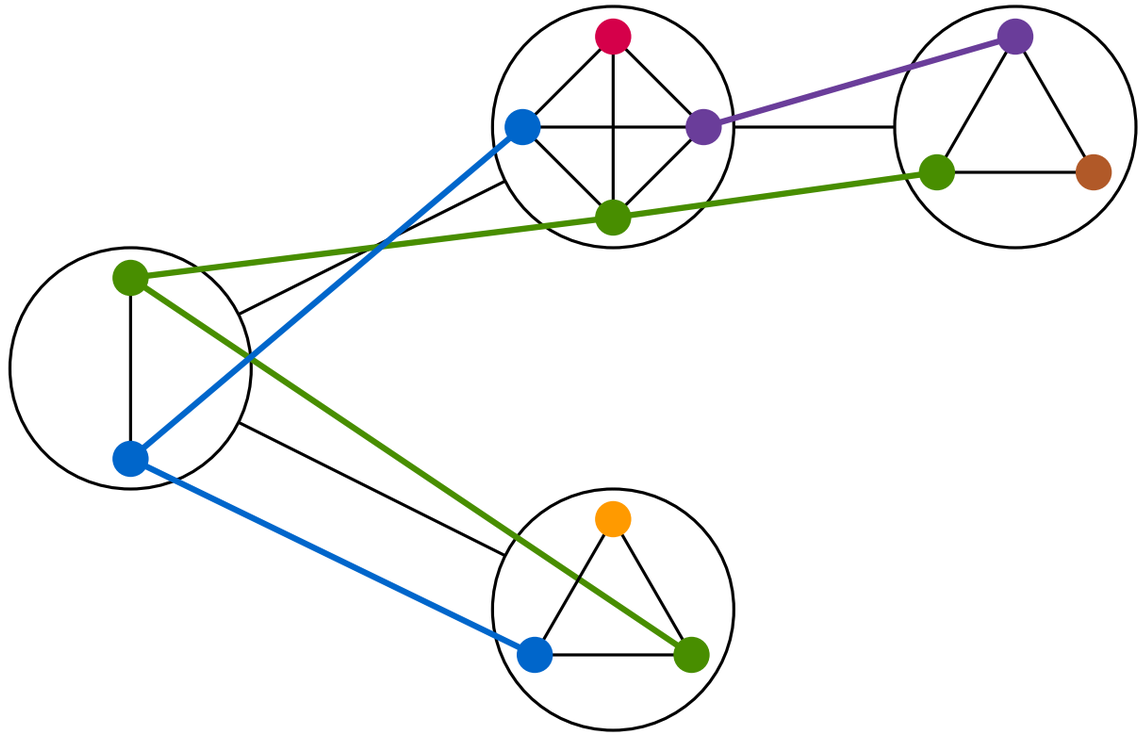


Tree T where bags are subsets of V s.t.

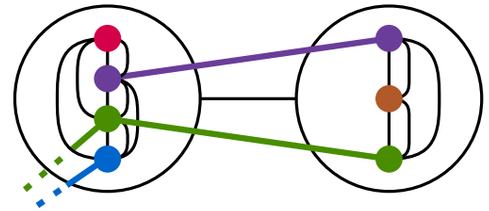
1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree



[Eppstein, Wikipedia'10]



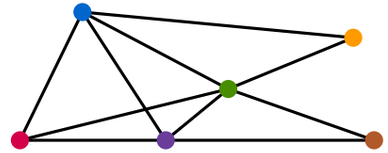
Linear



This talk:

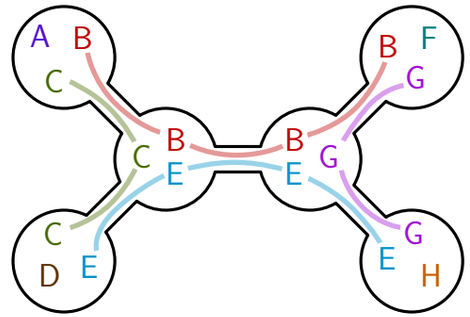
Crossing minimization in linear drawings

Witness Drawing Styles



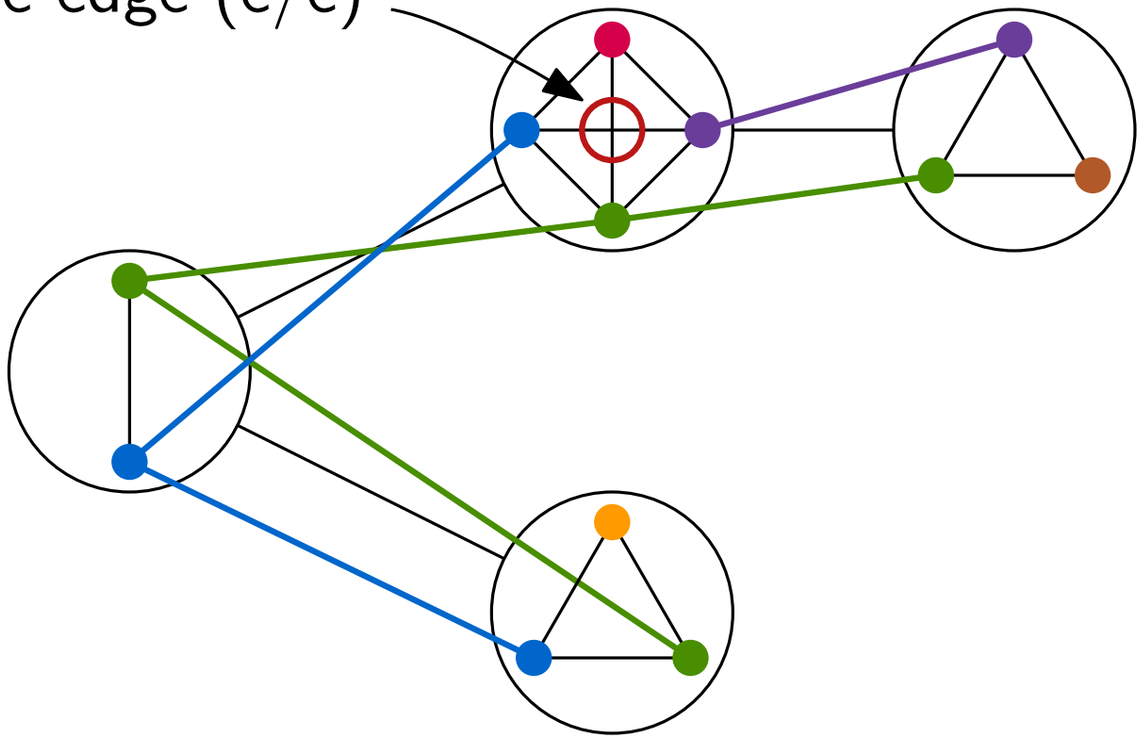
Tree T where bags are subsets of V s.t.

1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree

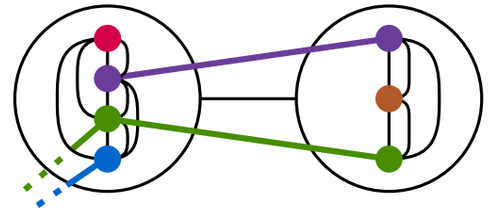


[Eppstein, Wikipedia'10]

Edge-edge (e/e)



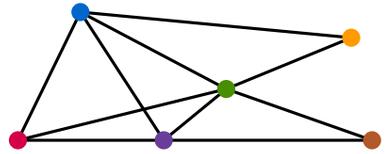
Linear



This talk:

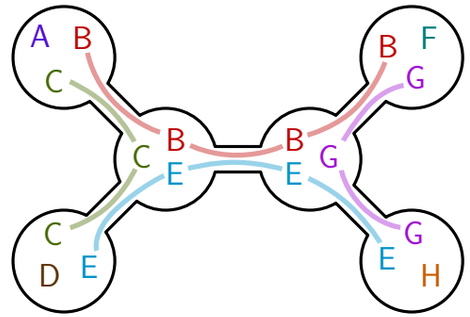
Crossing minimization in linear drawings

Witness Drawing Styles



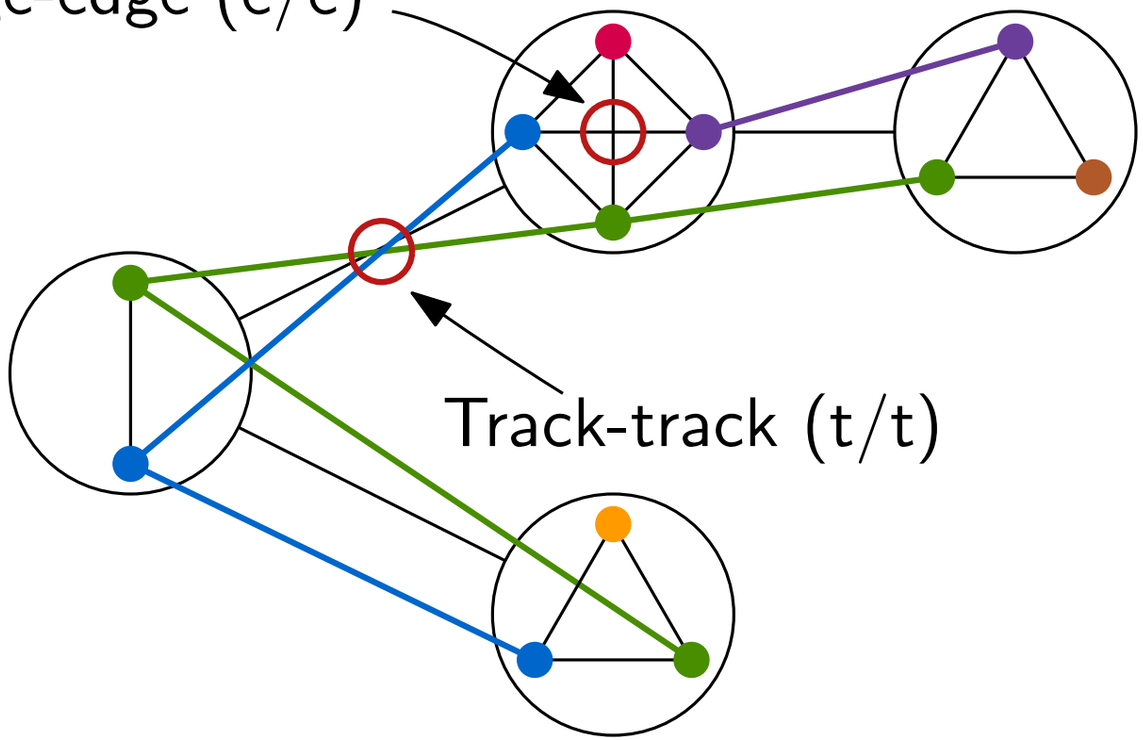
Tree T where bags are subsets of V s.t.

1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree

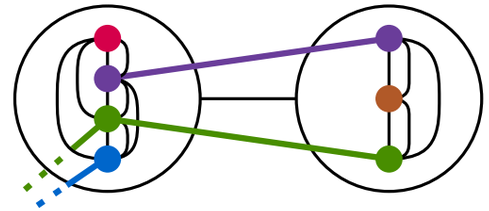


[Eppstein, Wikipedia'10]

Edge-edge (e/e)



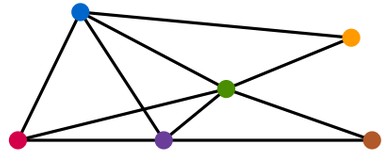
Linear



This talk:

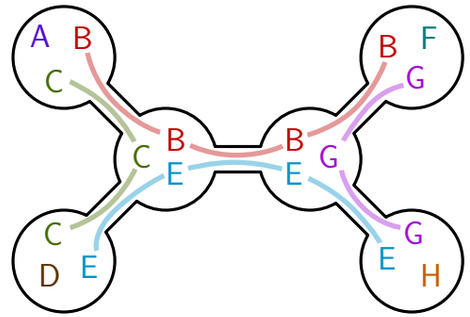
Crossing minimization in linear drawings

Witness Drawing Styles



Tree T where bags are subsets of V s.t.

1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree

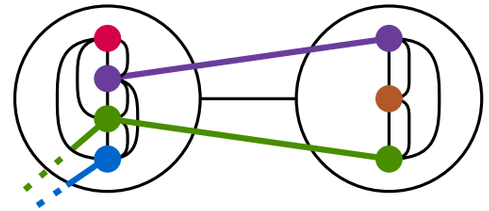


[Eppstein, Wikipedia'10]

Edge-edge (e/e)

Track-track (t/t)

Track-edge (t/e)

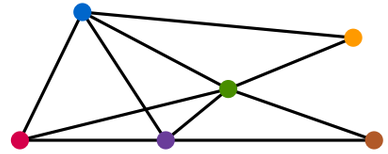


Linear

This talk:

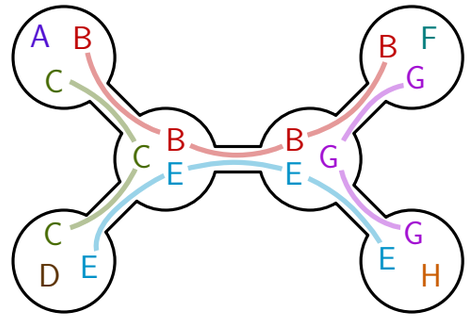
Crossing minimization in linear drawings

Witness Drawing Styles



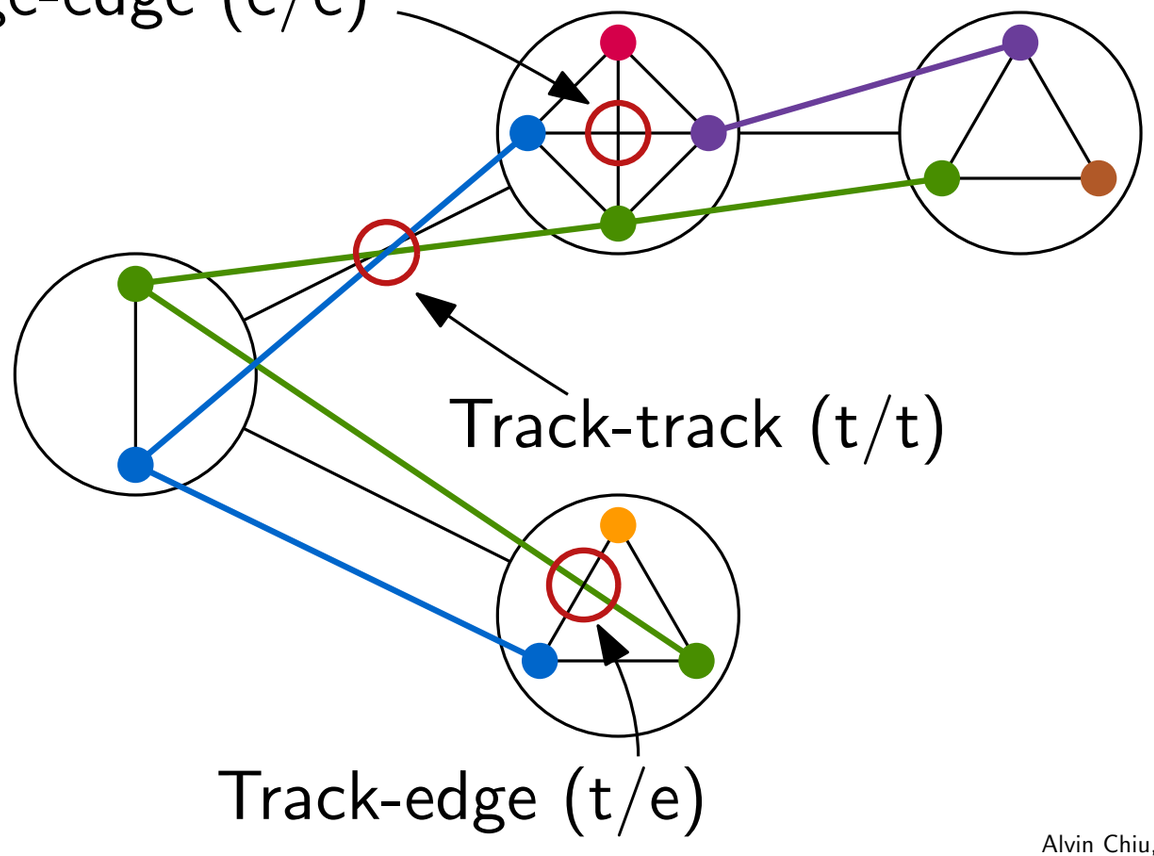
Tree T where bags are subsets of V s.t.

1. Every vertex is in a bag
2. For edge uv, there is bag with u and v
3. Bags containing $v \in V$ form subtree

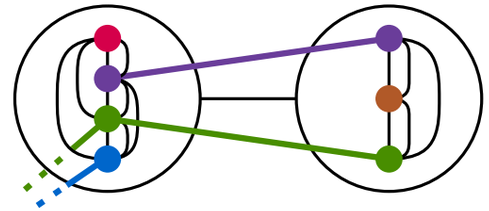


[Eppstein, Wikipedia'10]

Edge-edge (e/e)



Linear



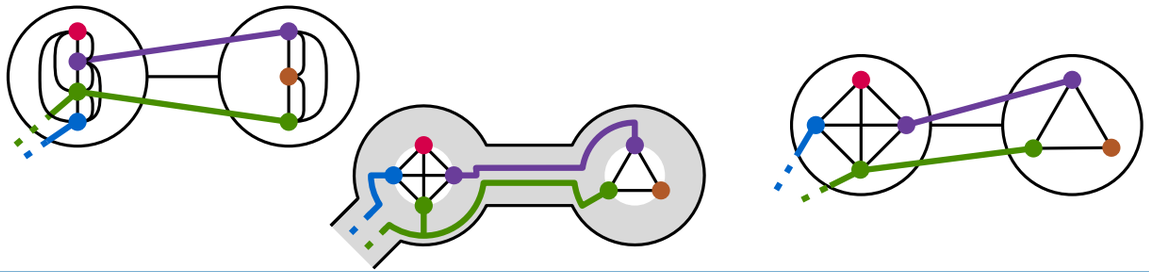
This talk:
Crossing minimization in linear drawings

Assumptions:
$\mathcal{O}(|V|)$ bags
Tree has maximum degree 3

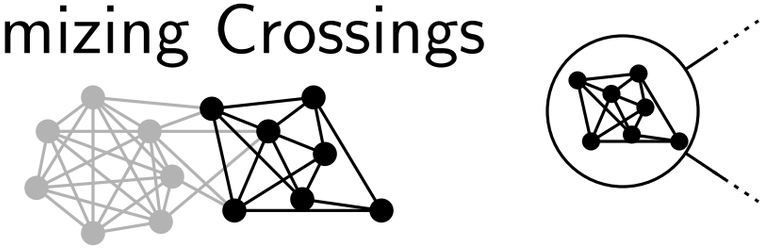
[Bodlaender & Kloks, J.Alg.'96]

Our Contributions

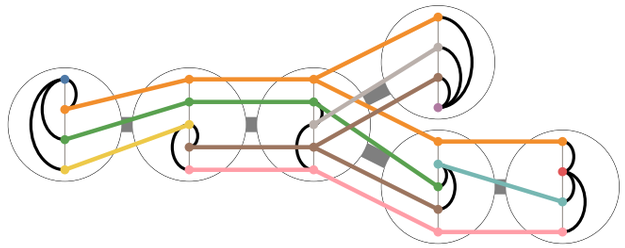
Drawing Paradigms for Treewidth



Complexity and Algorithms for Minimizing Crossings

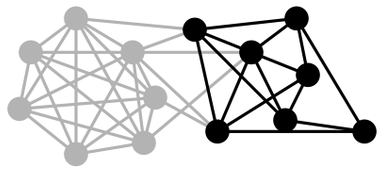


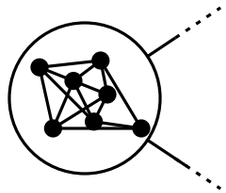
Experimental Evaluation



Our Contributions

Complexity and Algorithms for Minimizing Crossings





Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

Crossing Minimization for Linear Drawings

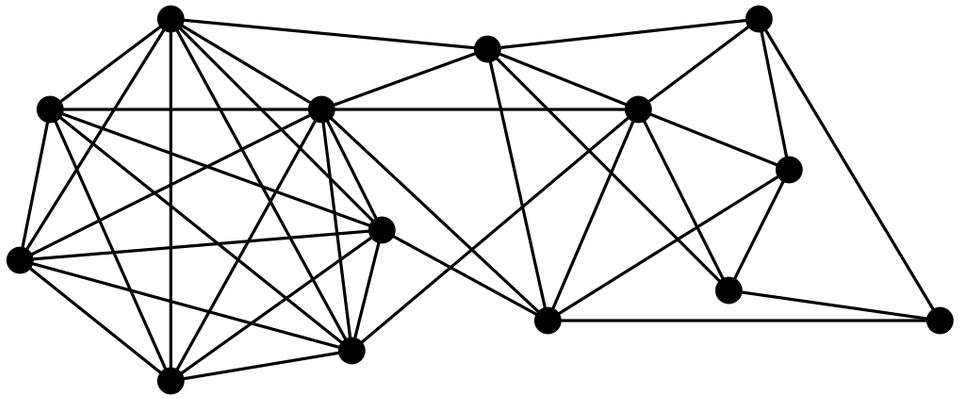
Goal: Minimize Σ (t/t, t/e, e/e-crossings)

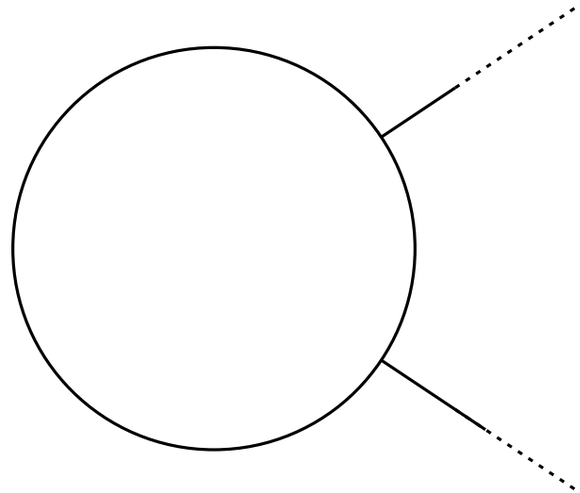
Caveat: Involves crossing minimization inside the bags

Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

Caveat: Involves crossing minimization inside the bags

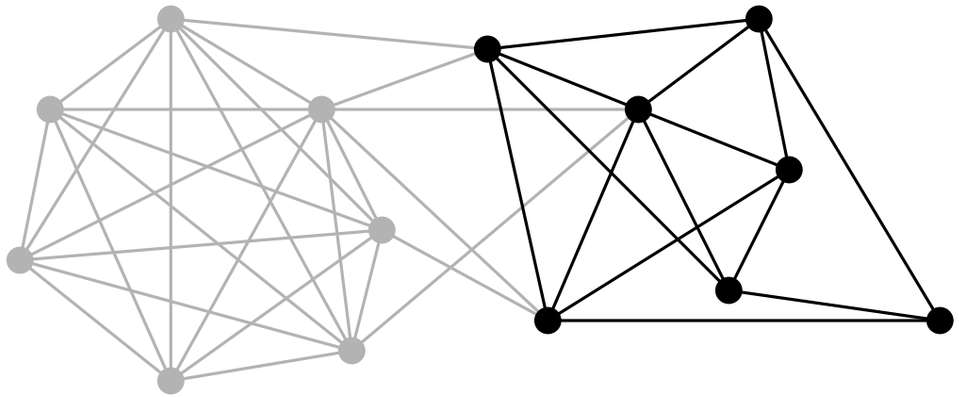


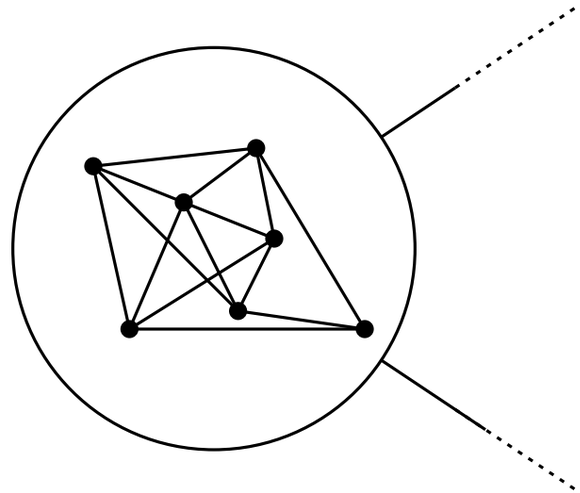


Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

Caveat: Involves crossing minimization inside the bags



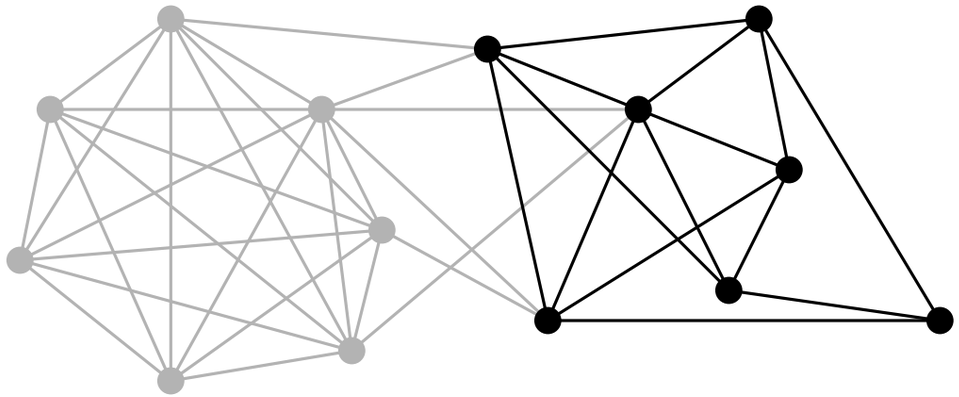


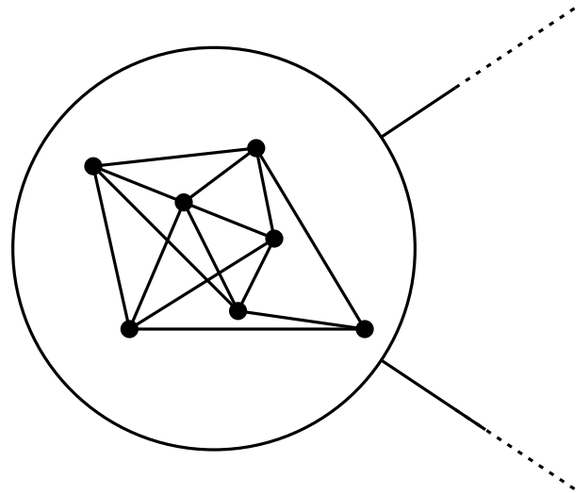
Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

Caveat: Involves crossing minimization inside the bags $\Rightarrow$ NP-hard

[Masuda et al., Symp. Circuits and Systems'87]



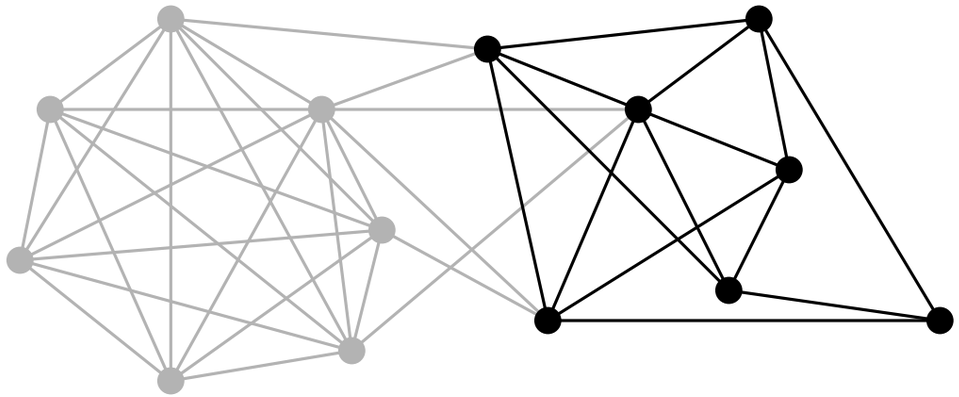


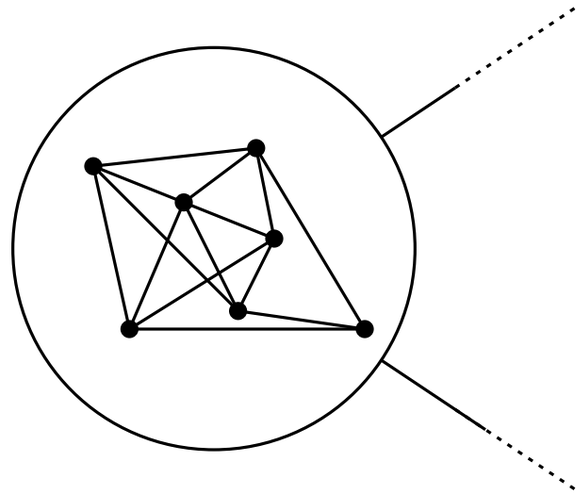
Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

Caveat: Involves crossing minimization inside the bags $\Rightarrow$ NP-hard

[Masuda et al., Symp. Circuits and Systems'87]





Theorem:
Deciding if a tree decomposition has a witness drawing with at most c crossings is NP-hard.

Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

Theorem:
Deciding if a tree decomposition has a witness drawing with at most c crossings is NP-hard.

Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

Theorem:
Deciding if a tree decomposition has a witness drawing with at most c crossings is NP-hard.

Theorem:
A crossing-minimal linear witness drawing of a tree decomposition of width ω can be computed in $\mathcal{O}(f(\omega) \cdot |V|)$ time.

Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

Theorem:
Deciding if a tree decomposition has a witness drawing with at most c crossings is NP-hard.

Theorem:
A crossing-minimal linear witness drawing of a tree decomposition of width ω can be computed in $\mathcal{O}(f(\omega) \cdot |V|)$ time.

$f(\omega)$ exponential in ω

Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

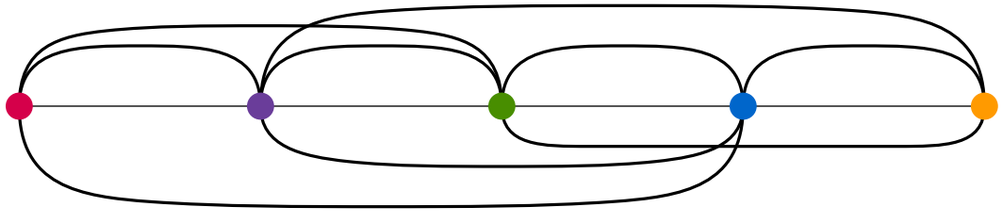
Theorem:
Deciding if a tree decomposition has a witness drawing with at most c crossings is NP-hard.

Theorem:
A crossing-minimal linear witness drawing of a tree decomposition of width ω can be computed in $\mathcal{O}(f(\omega) \cdot |V|)$ time.

$f(\omega)$ exponential in $\omega \Rightarrow$ Heuristics!

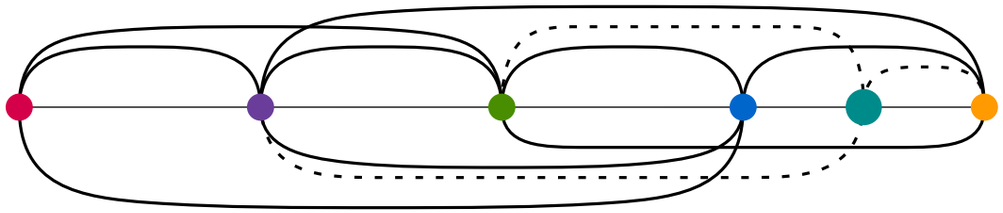
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]



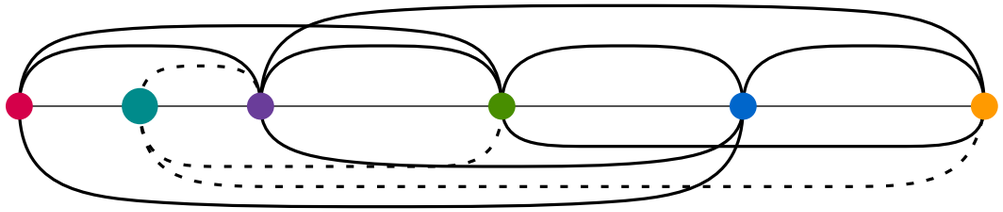
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]



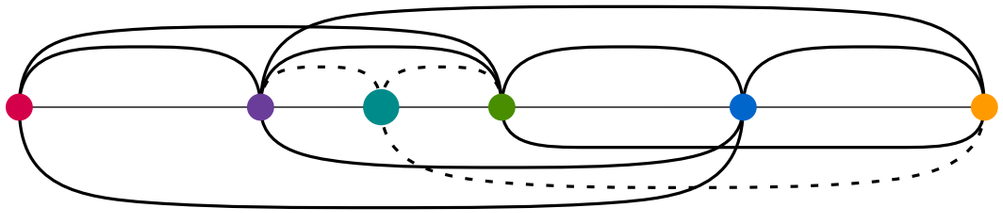
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]



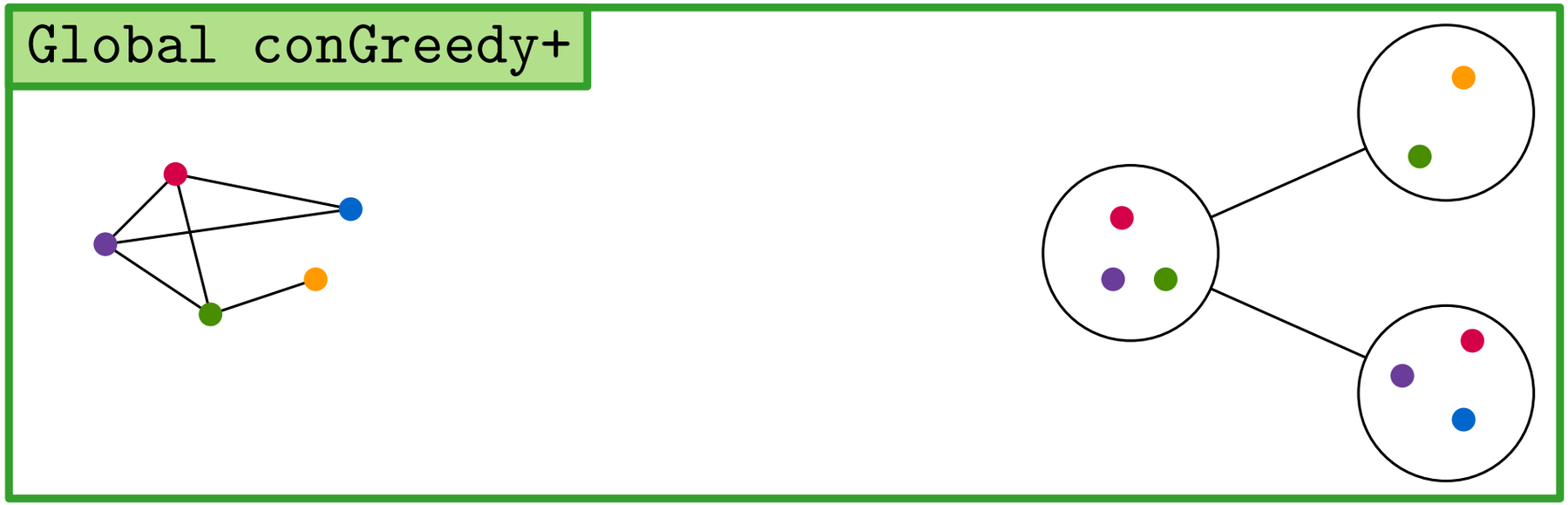
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]



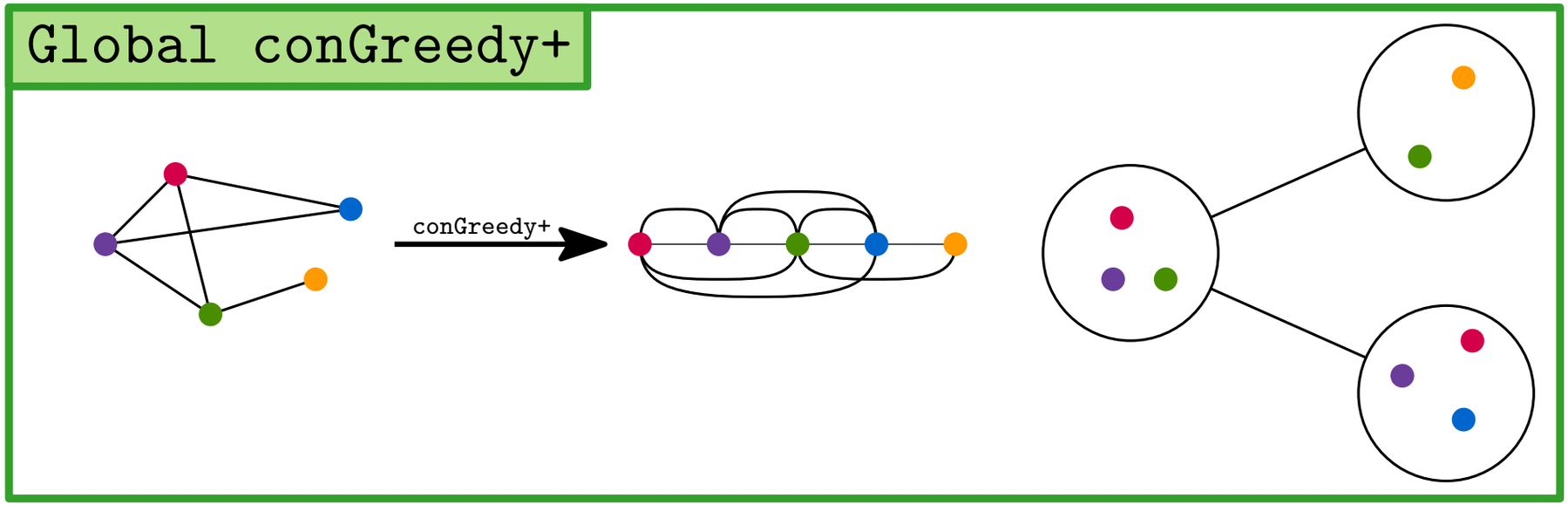
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]



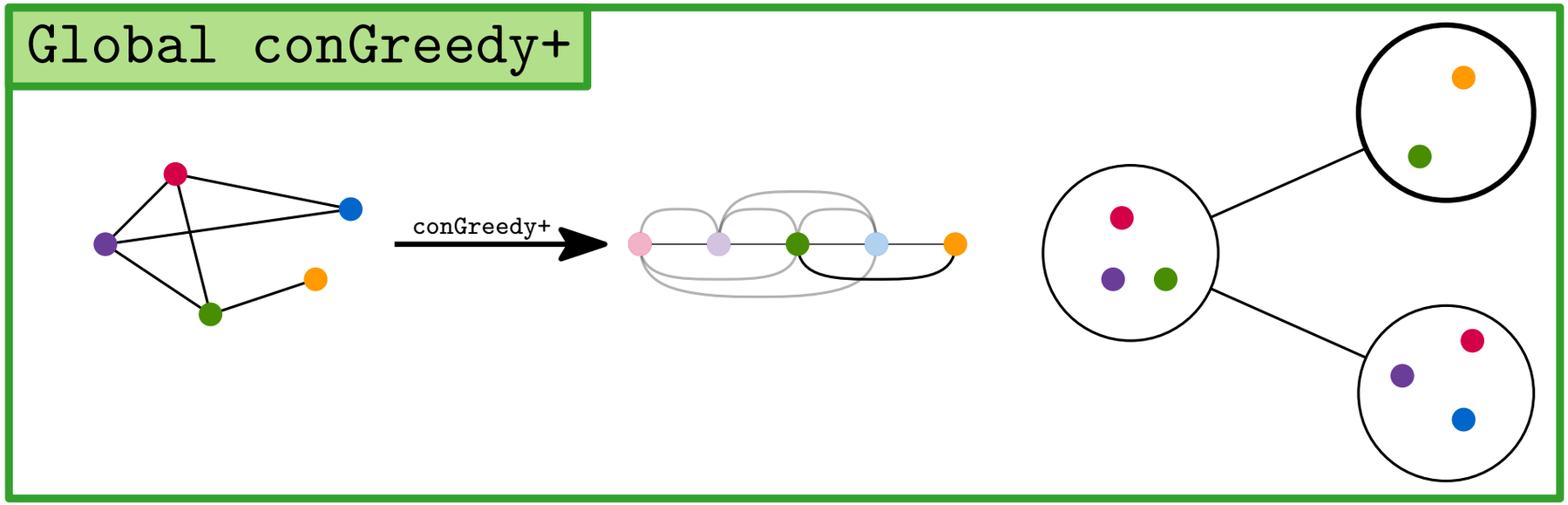
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]



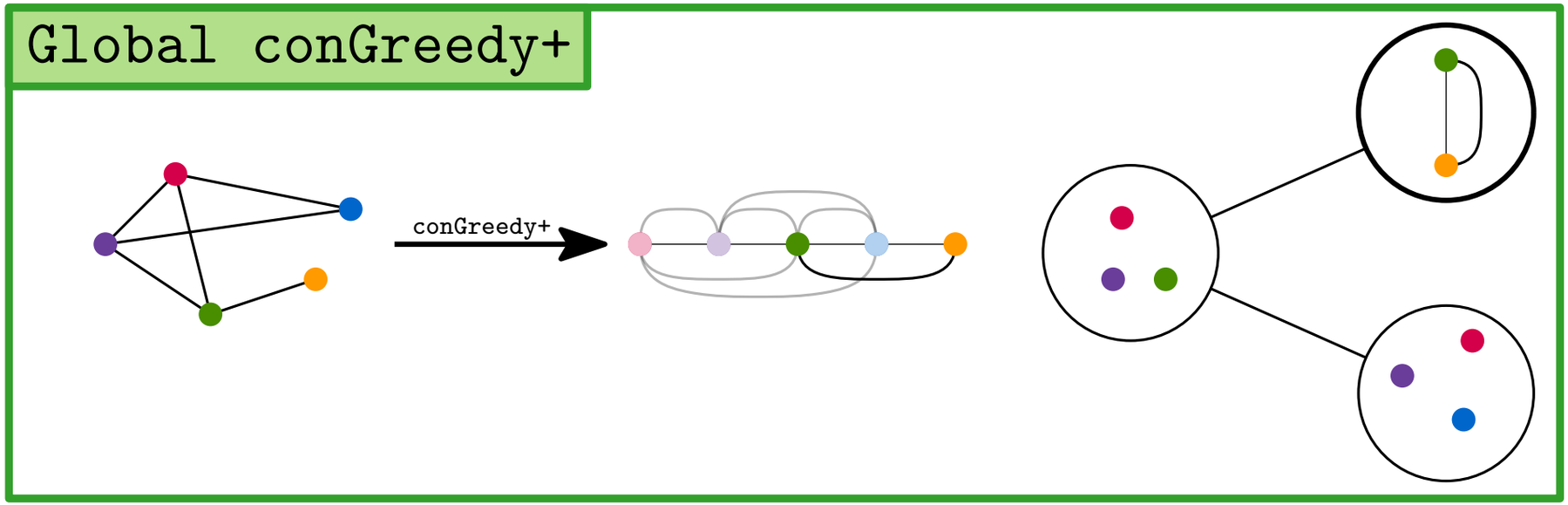
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]



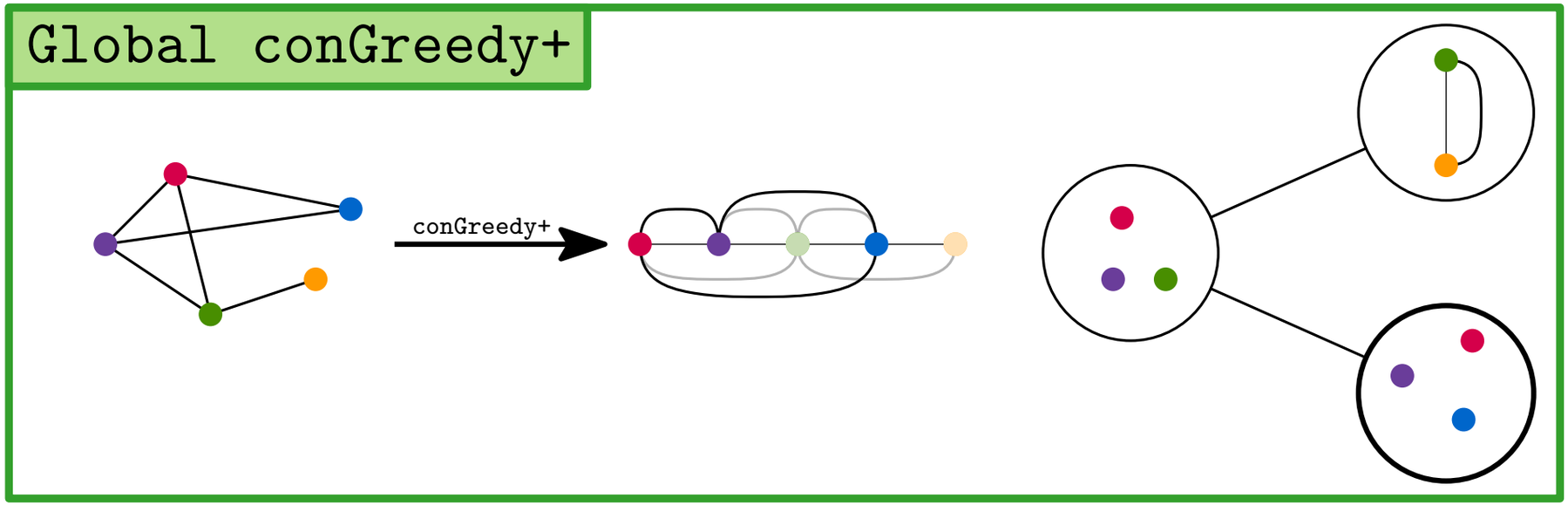
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]



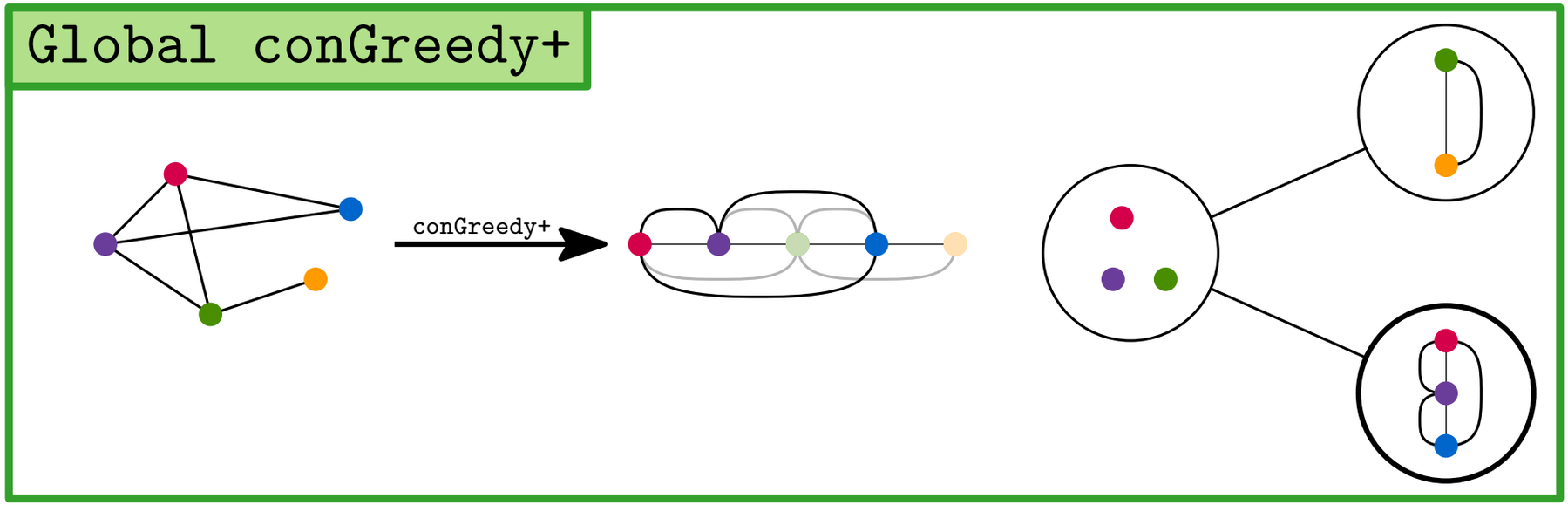
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]



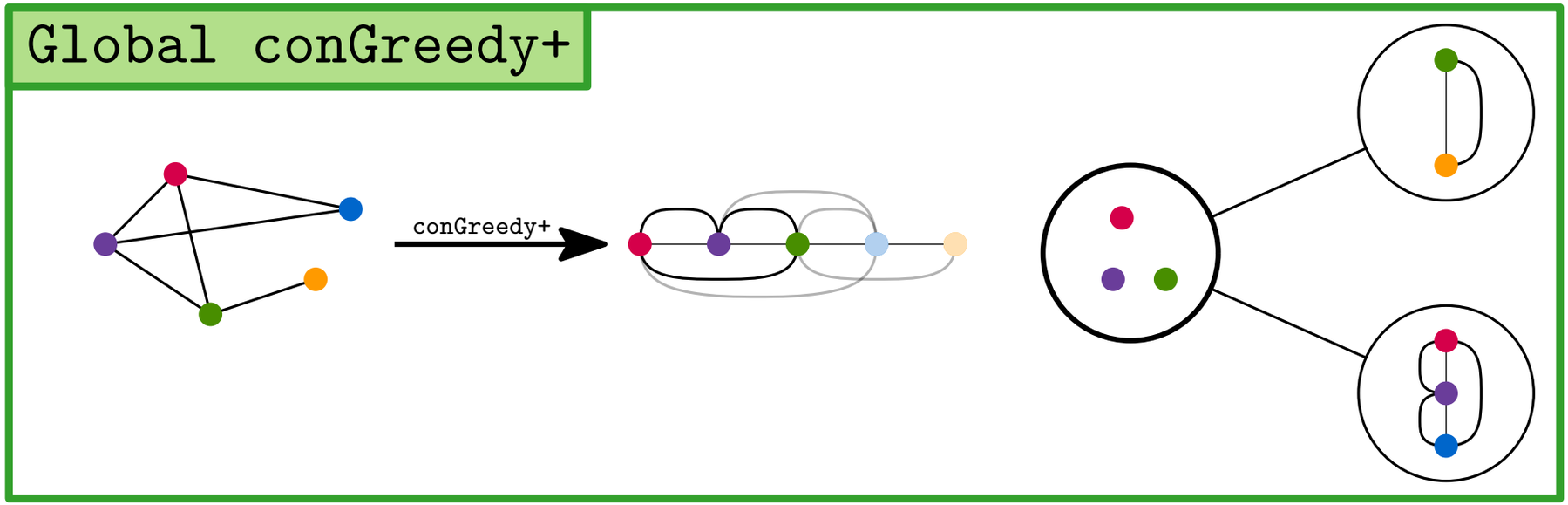
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]



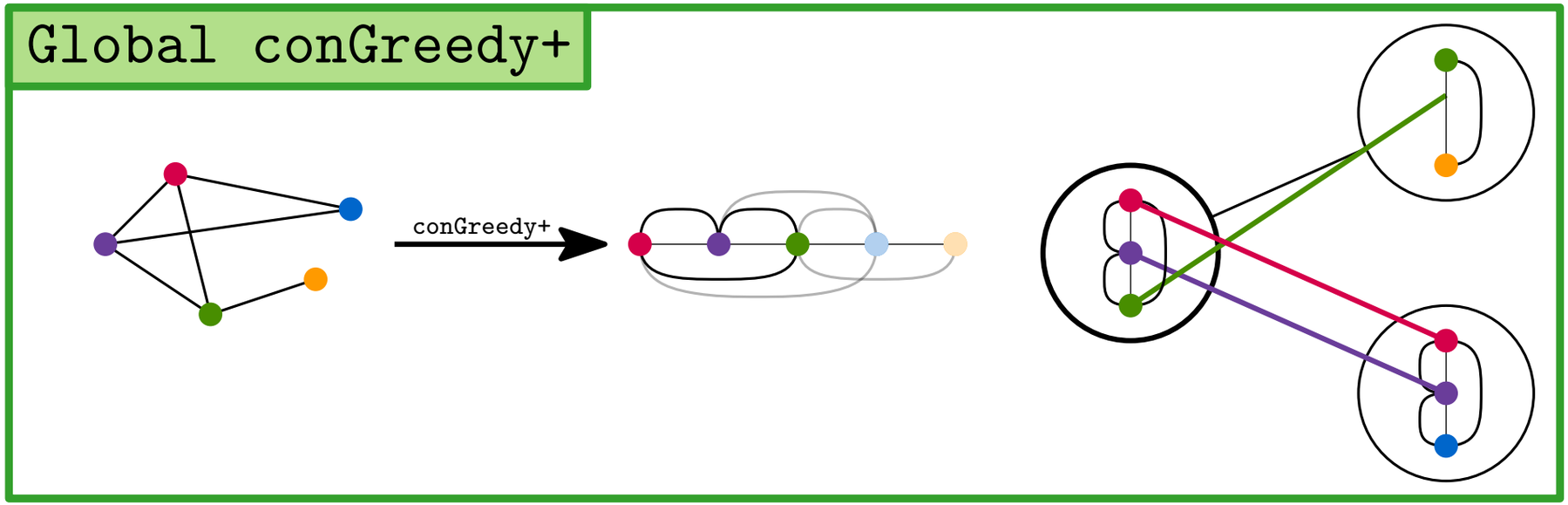
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]



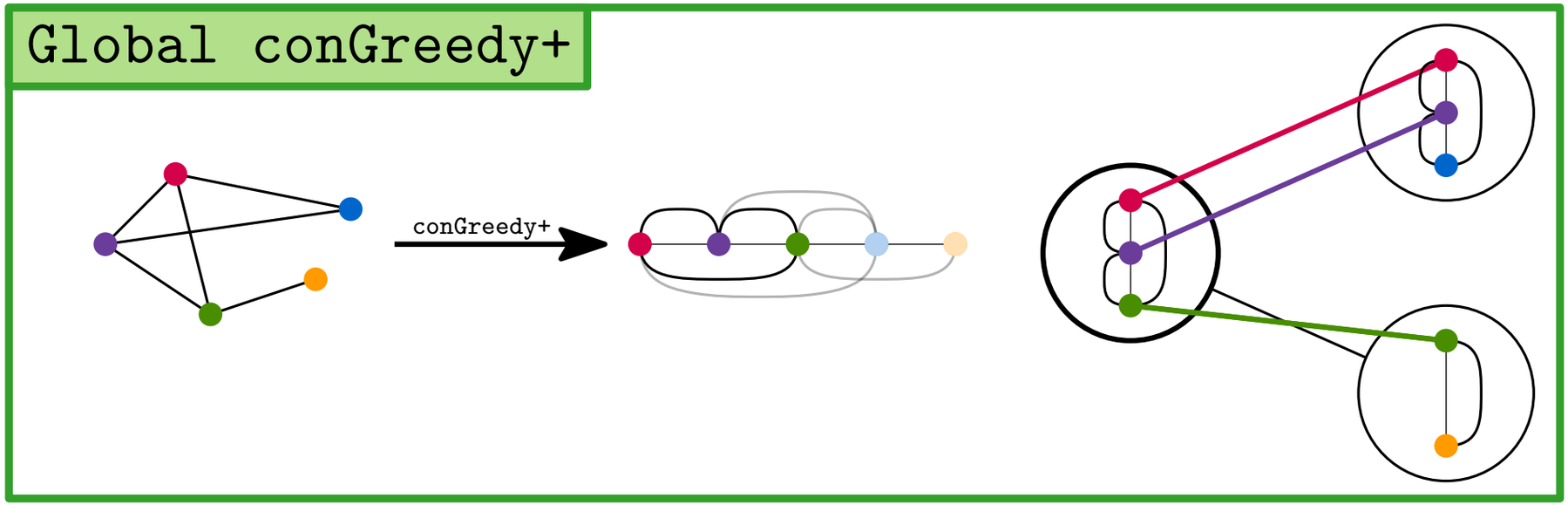
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]



Heuristics for Linear Witness Drawings

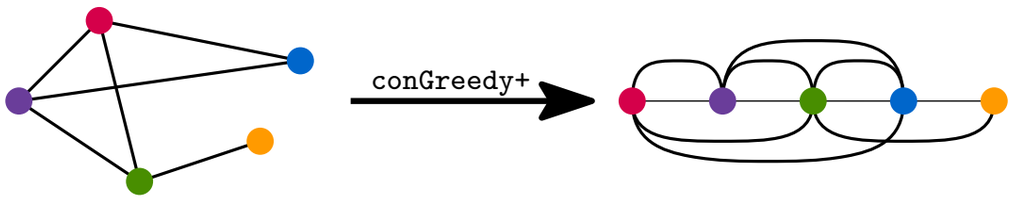
Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]

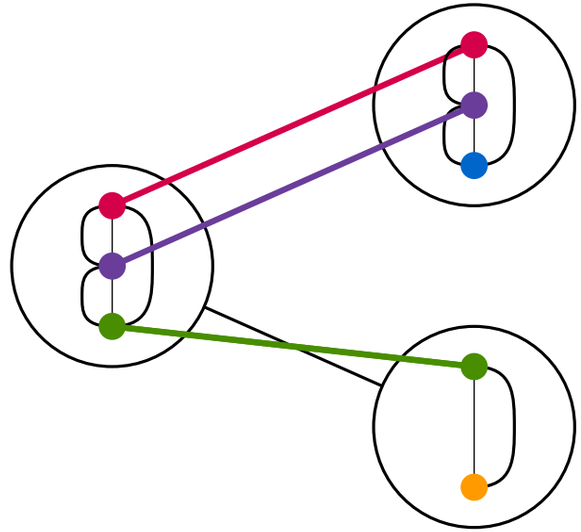


Heuristics for Linear Witness Drawings

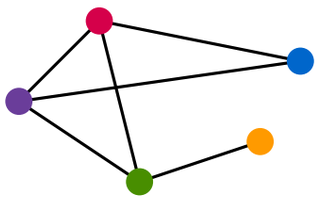
Building Block: Book drawing heuristic conGreedy+ [Klawitter, Mchedlidze, & Nöllenburg, GD'17]

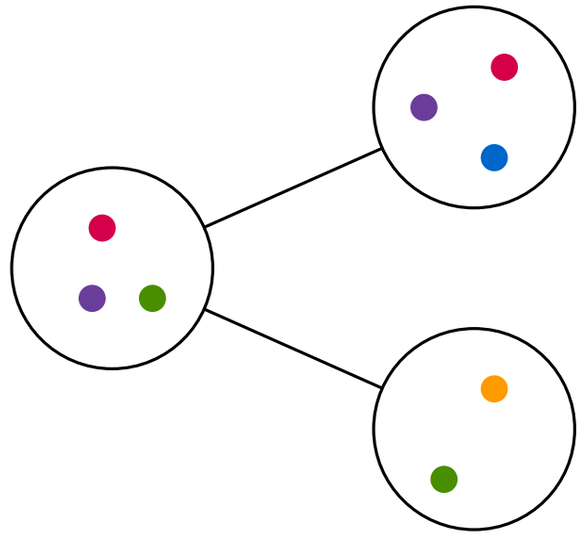
Global conGreedy+





Local conGreedy+



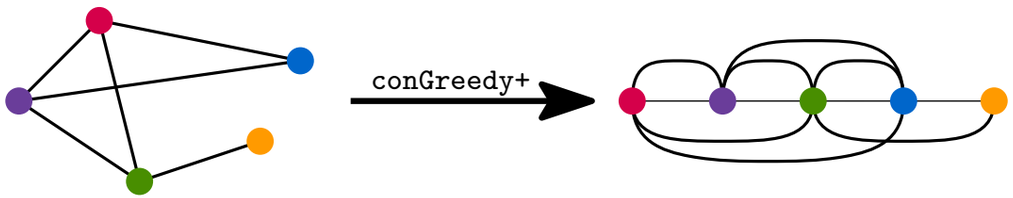


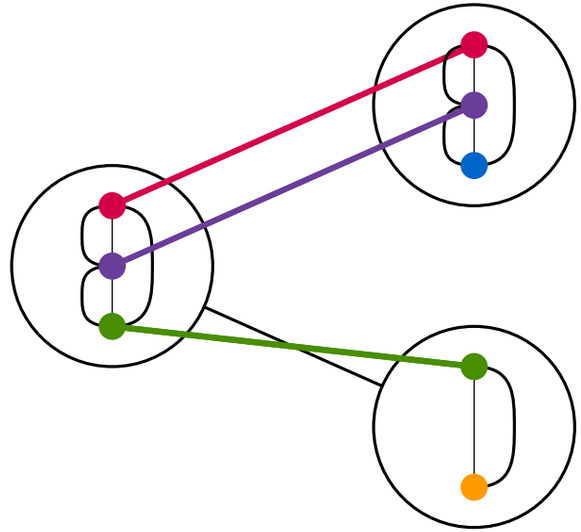
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic conGreedy+

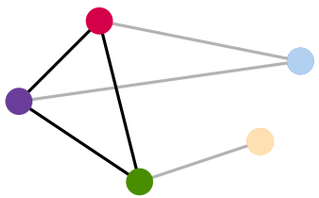
[Klawitter, Mchedlidze, & Nöllenburg, GD'17]

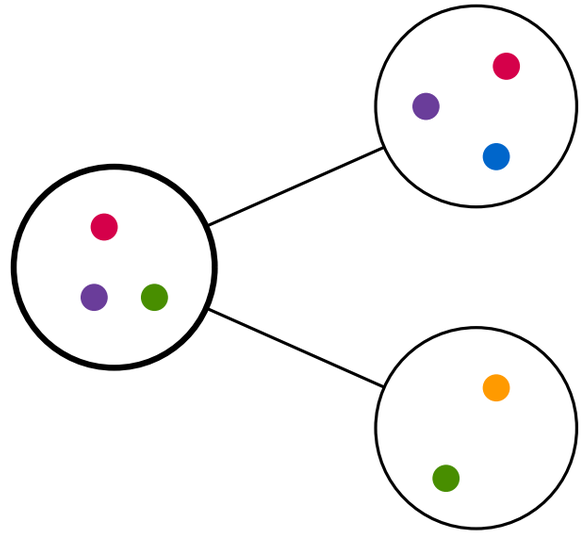
Global conGreedy+





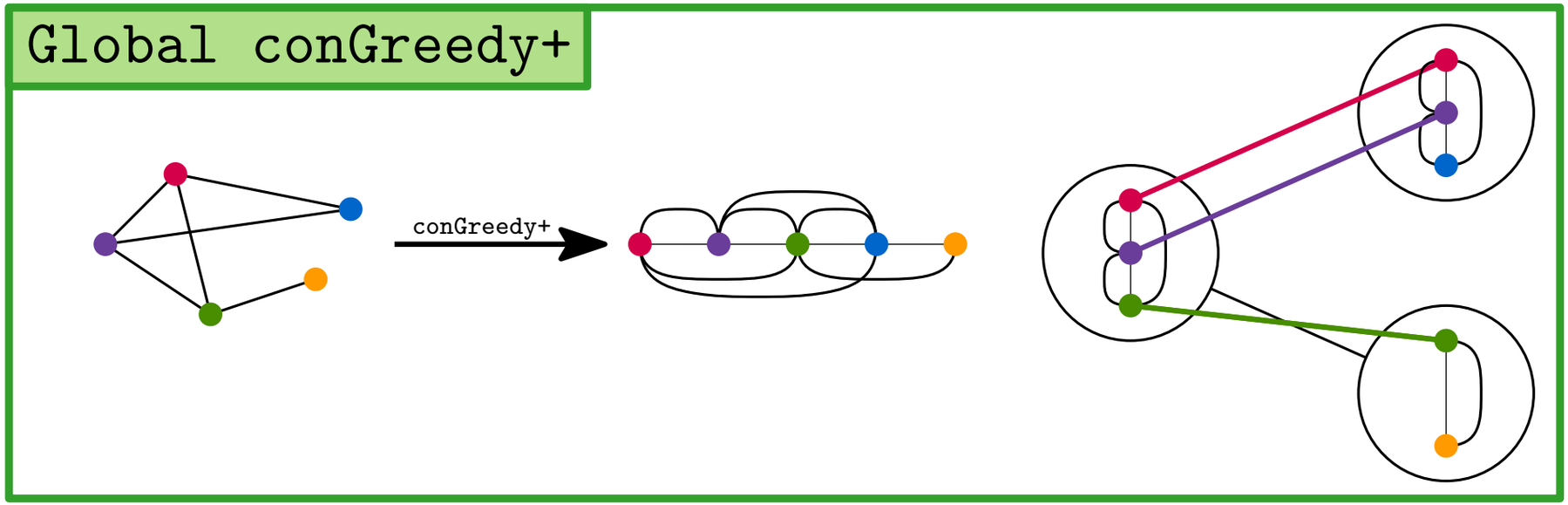
Local conGreedy+

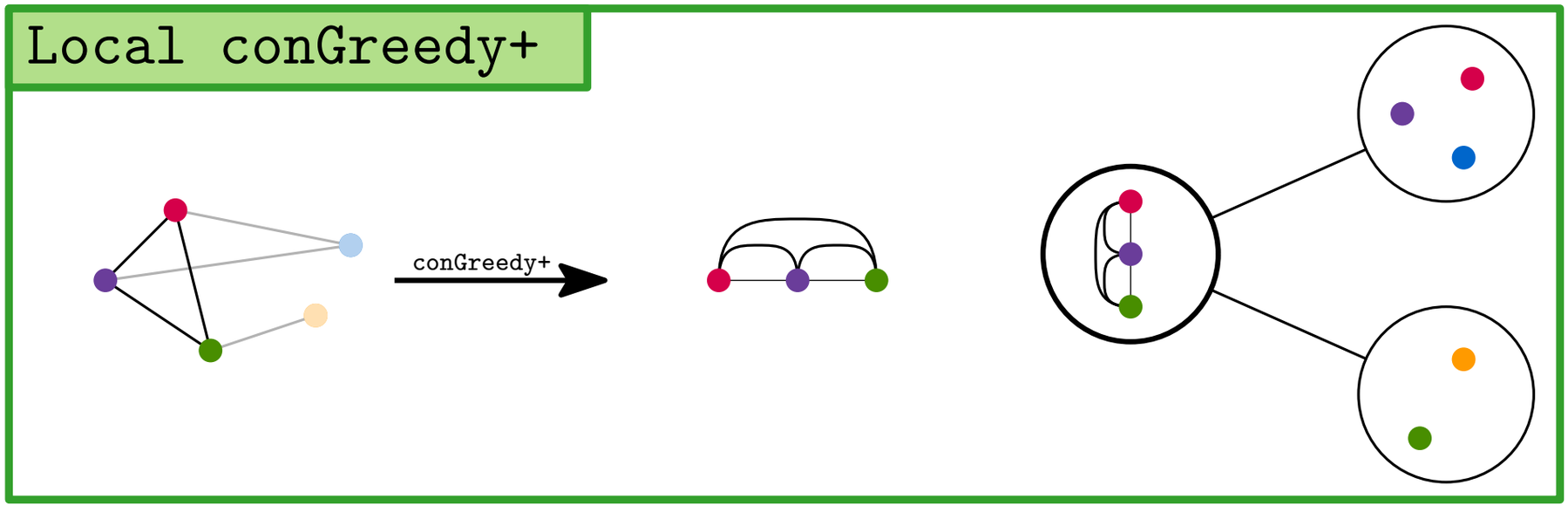




Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]



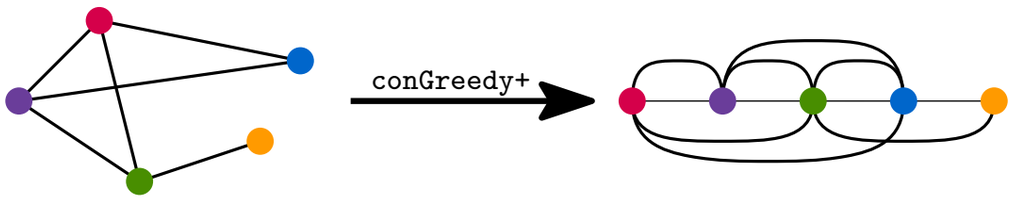


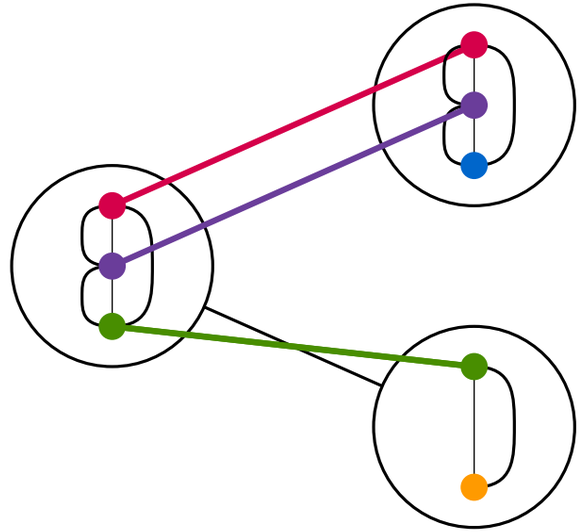
Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic conGreedy+

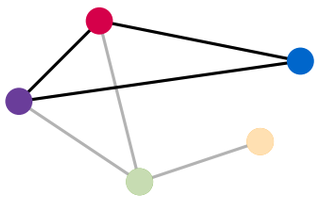
[Klawitter, Mchedlidze, & Nöllenburg, GD'17]

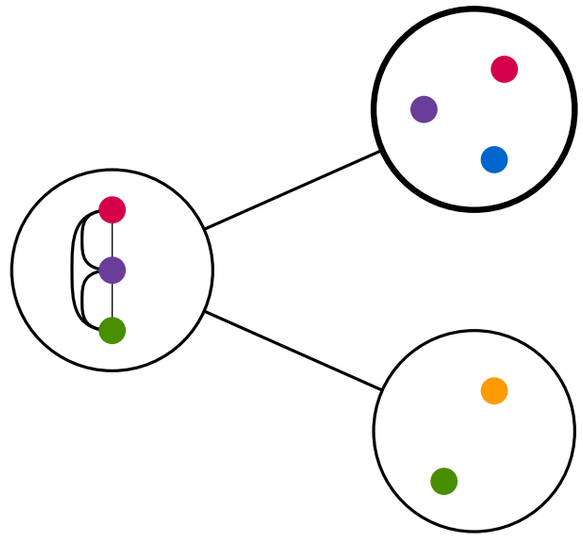
Global conGreedy+





Local conGreedy+



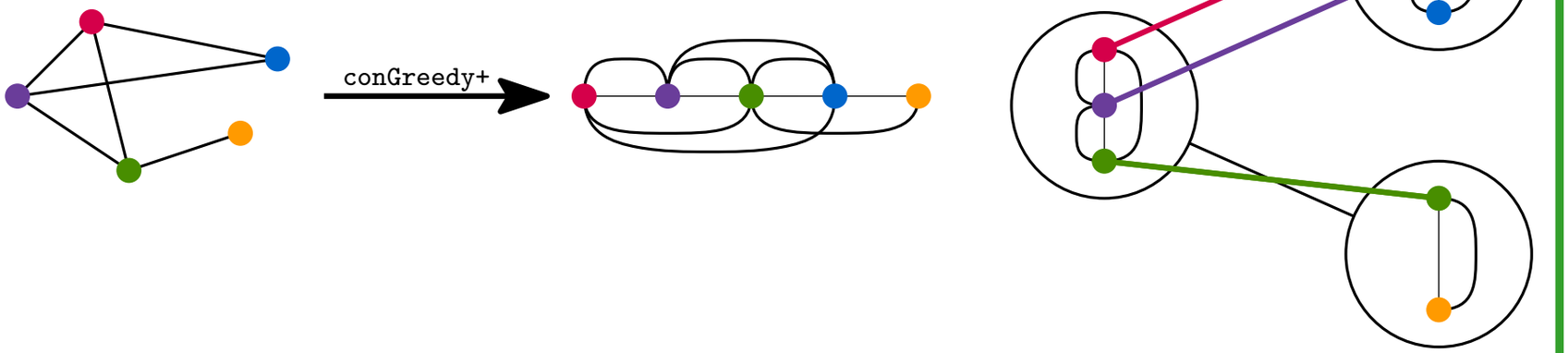


Heuristics for Linear Witness Drawings

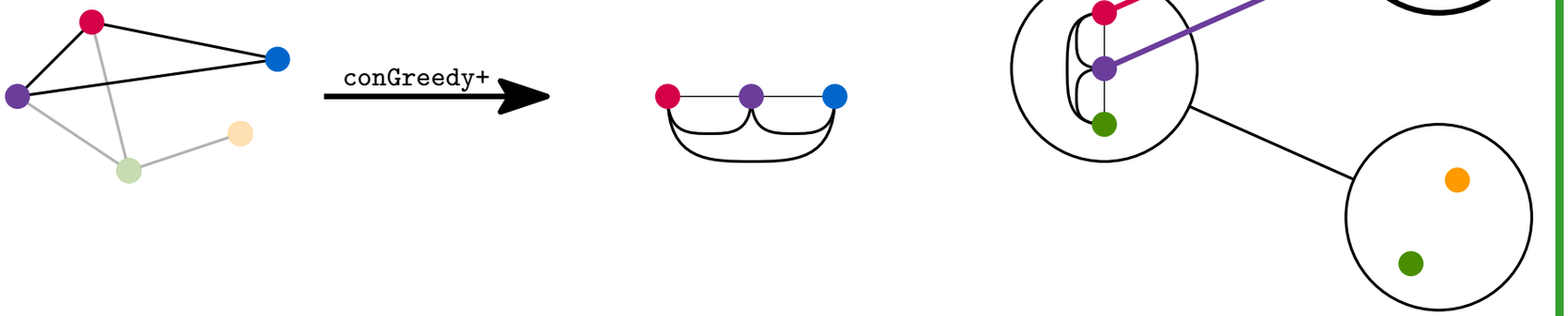
Building Block: Book drawing heuristic `conGreedy+`

[Klawitter, Mchedlidze, & Nöllenburg, GD'17]

`Global conGreedy+`



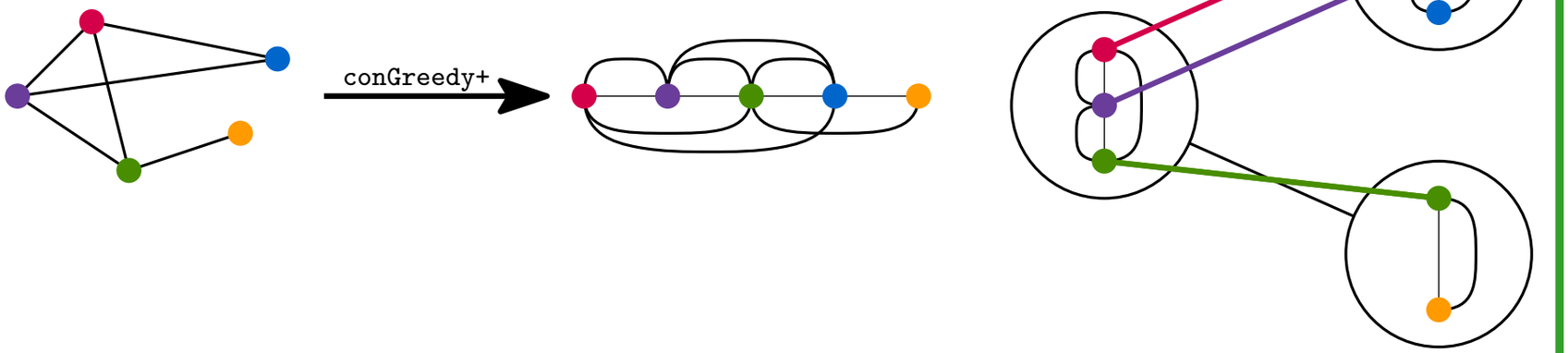
`Local conGreedy+`



Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic conGreedy+ [Klawitter, Mchedlidze, & Nöllenburg, GD'17]

Global conGreedy+



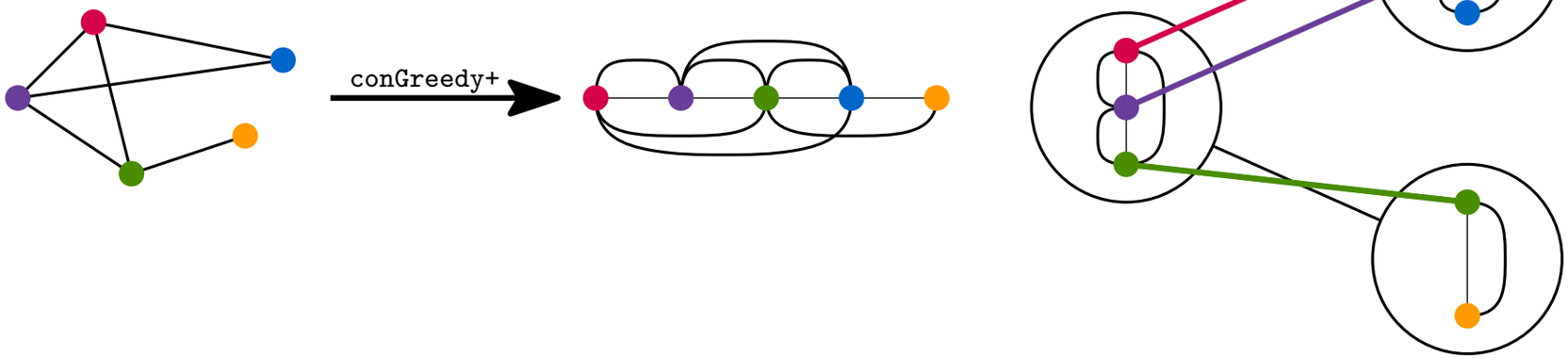
Local conGreedy+

Heuristics for Linear Witness Drawings

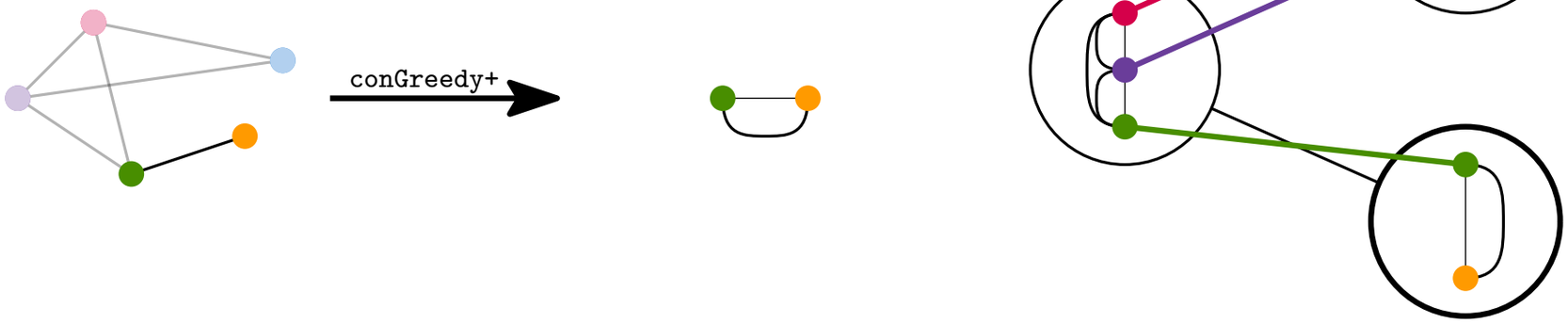
Building Block: Book drawing heuristic conGreedy+

[Klawitter, Mchedlidze, & Nöllenburg, GD'17]

Global conGreedy+



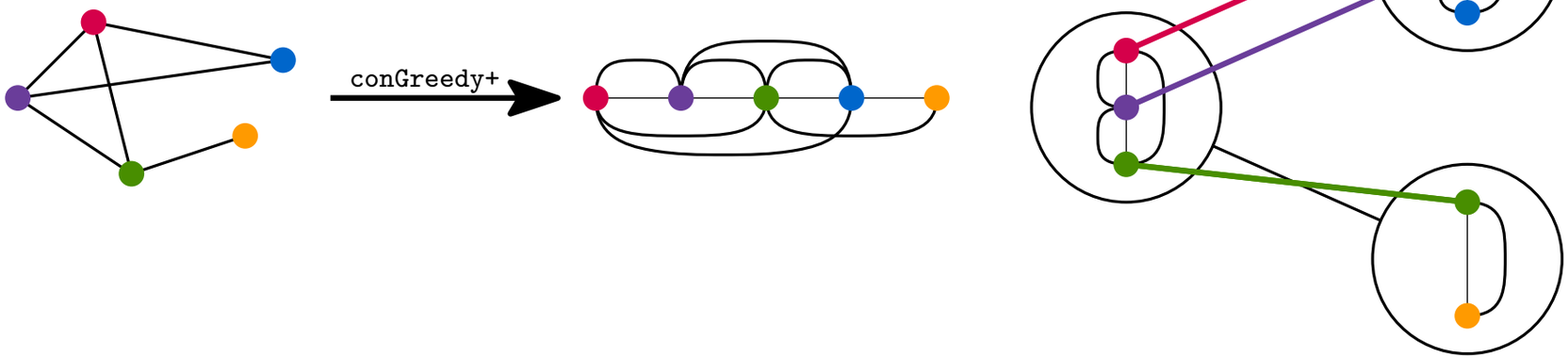
Local conGreedy+



Heuristics for Linear Witness Drawings

Building Block: Book drawing heuristic `conGreedy+` [Klawitter, Mchedlidze, & Nöllenburg, GD'17]

Global conGreedy+



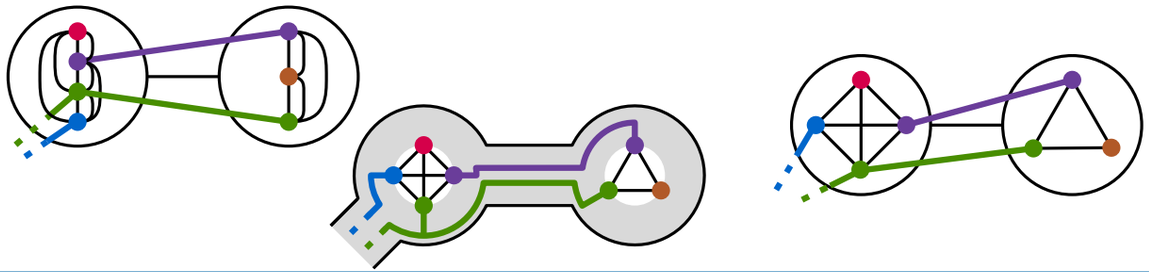
Local conGreedy+

Local Search

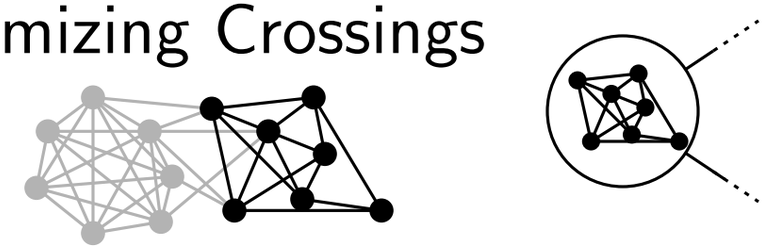
- Vertex-Swap
- Edge-Swap
- Edge-Flip
- Embedding-Flip

Our Contributions

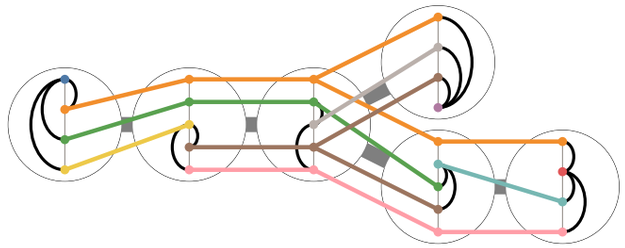
Drawing Paradigms for Treewidth



Complexity and Algorithms for Minimizing Crossings

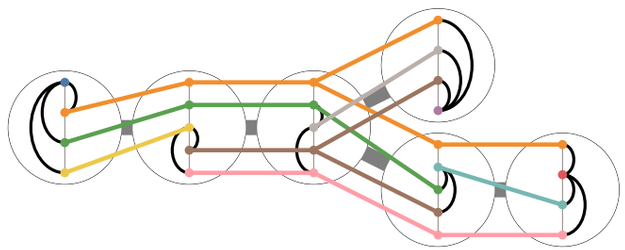


Experimental Evaluation



Our Contributions

Experimental Evaluation

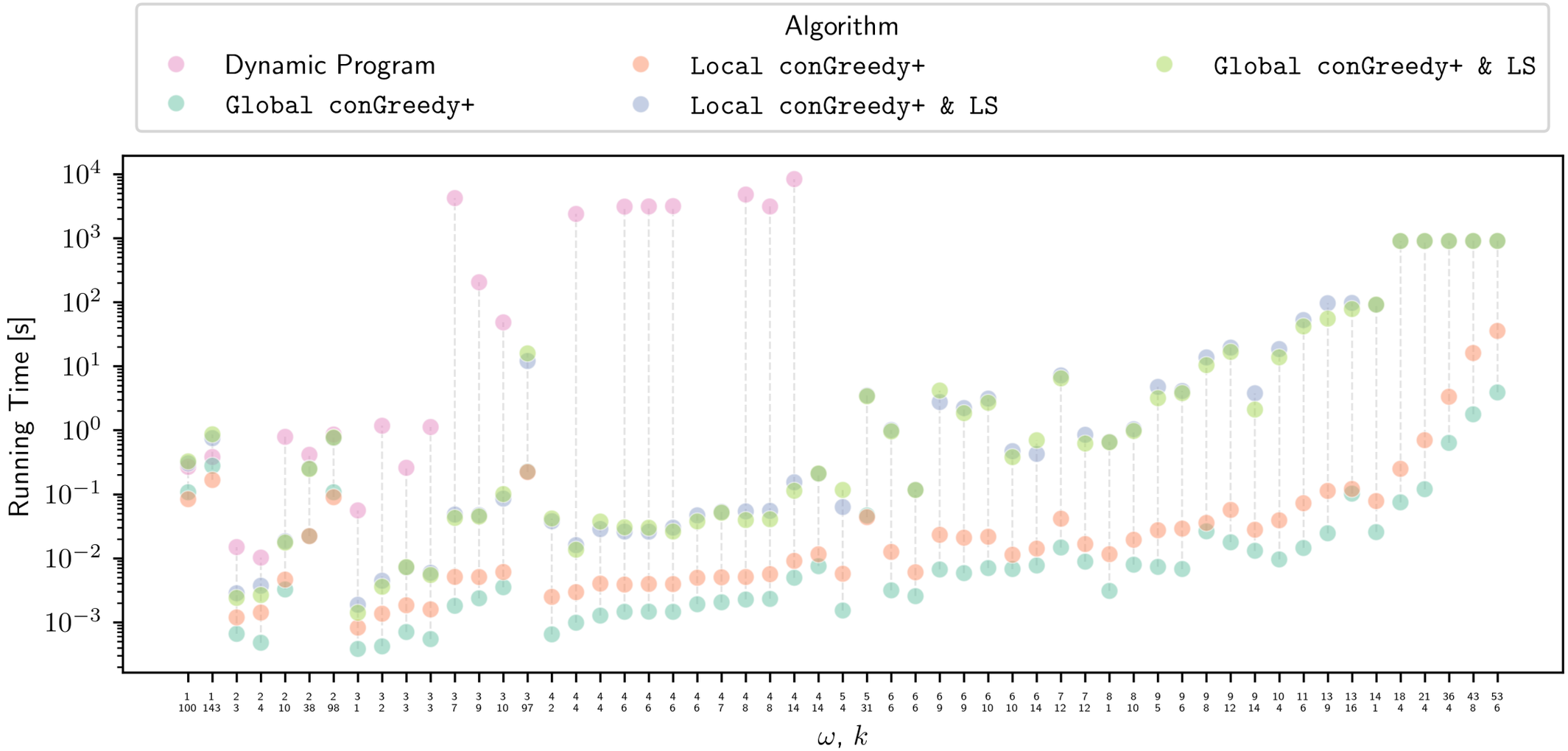


Experimental Evaluation – Running Time

Dataset: 55 graphs from “Sage” with tree decomposition, 15min time limit

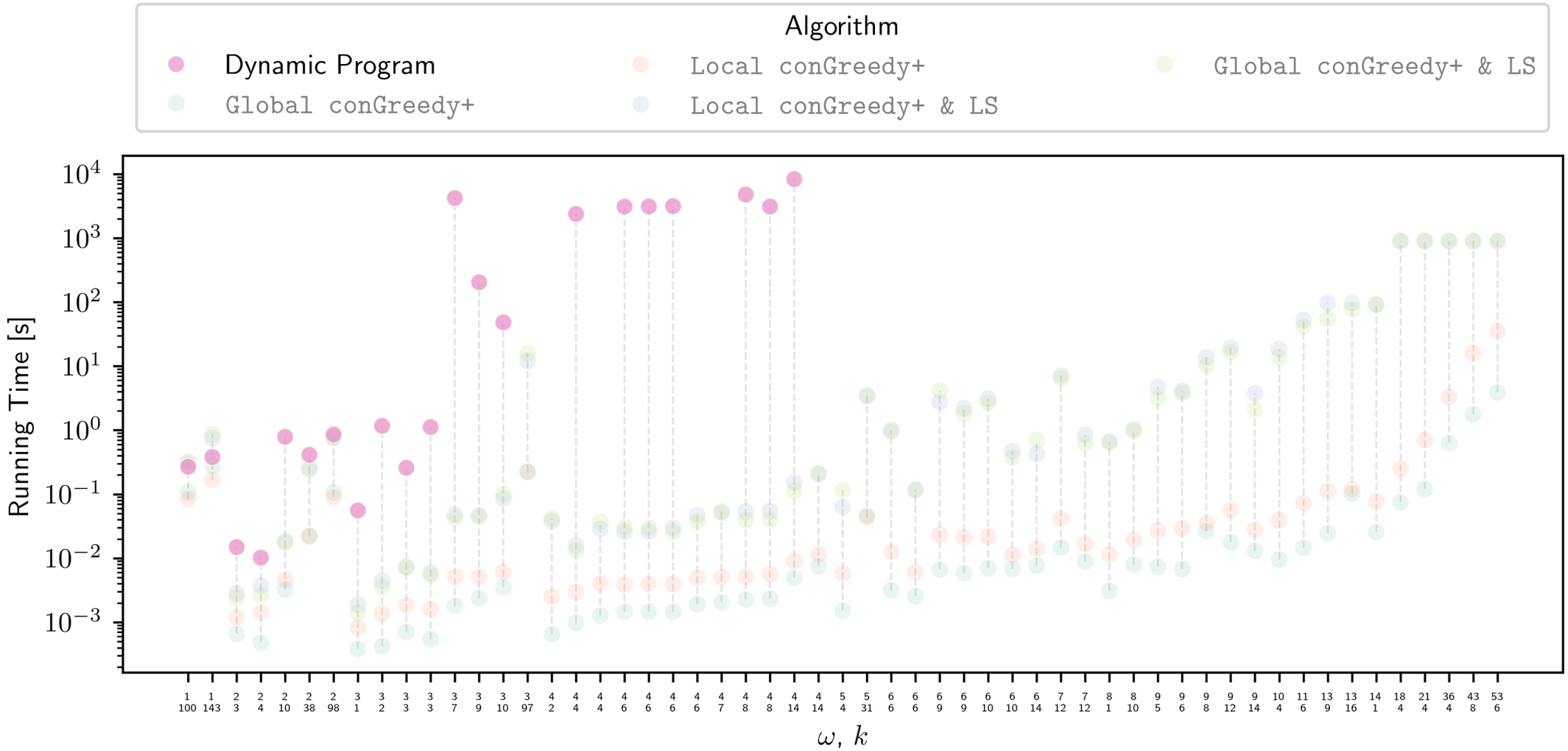
Experimental Evaluation – Running Time

Dataset: 55 graphs from “Sage” with tree decomposition, 15min time limit



Experimental Evaluation – Running Time

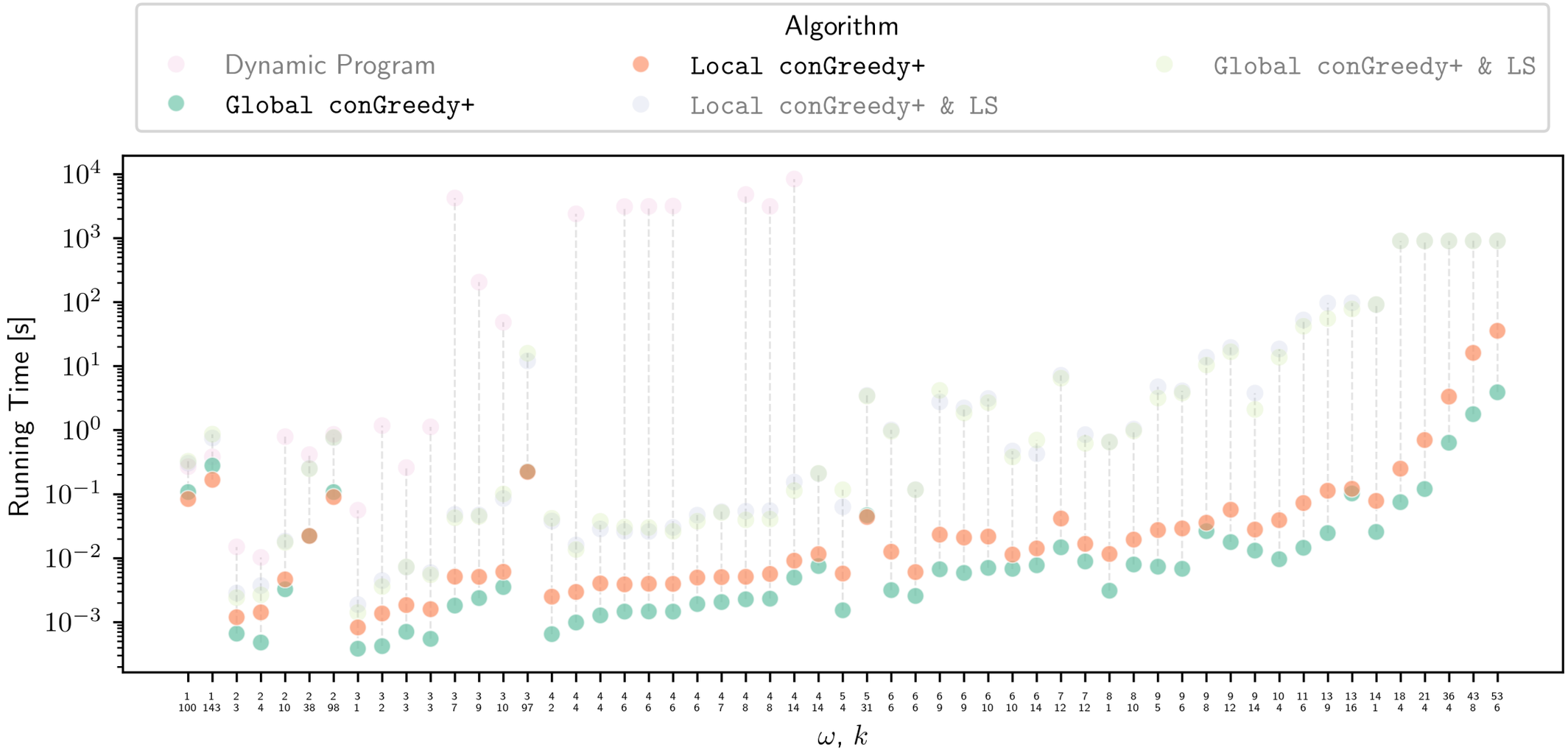
Dataset: 55 graphs from “Sage” with tree decomposition, 15min time limit



DP only feasible for small decompositions (width ≤ 4)

Experimental Evaluation – Running Time

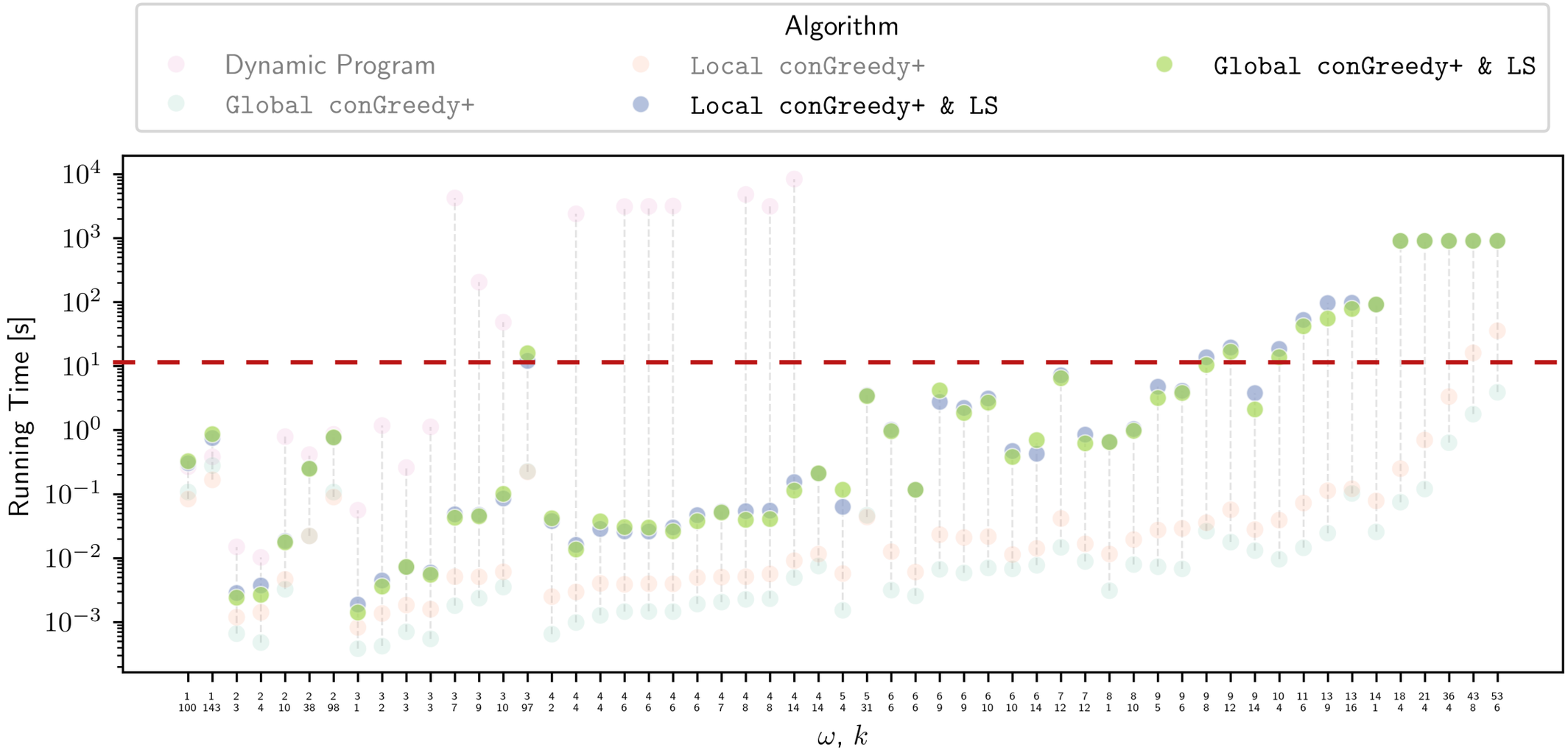
Dataset: 55 graphs from “Sage” with tree decomposition, 15min time limit



`Global conGreedy+` faster than `Local conGreedy+`

Experimental Evaluation – Running Time

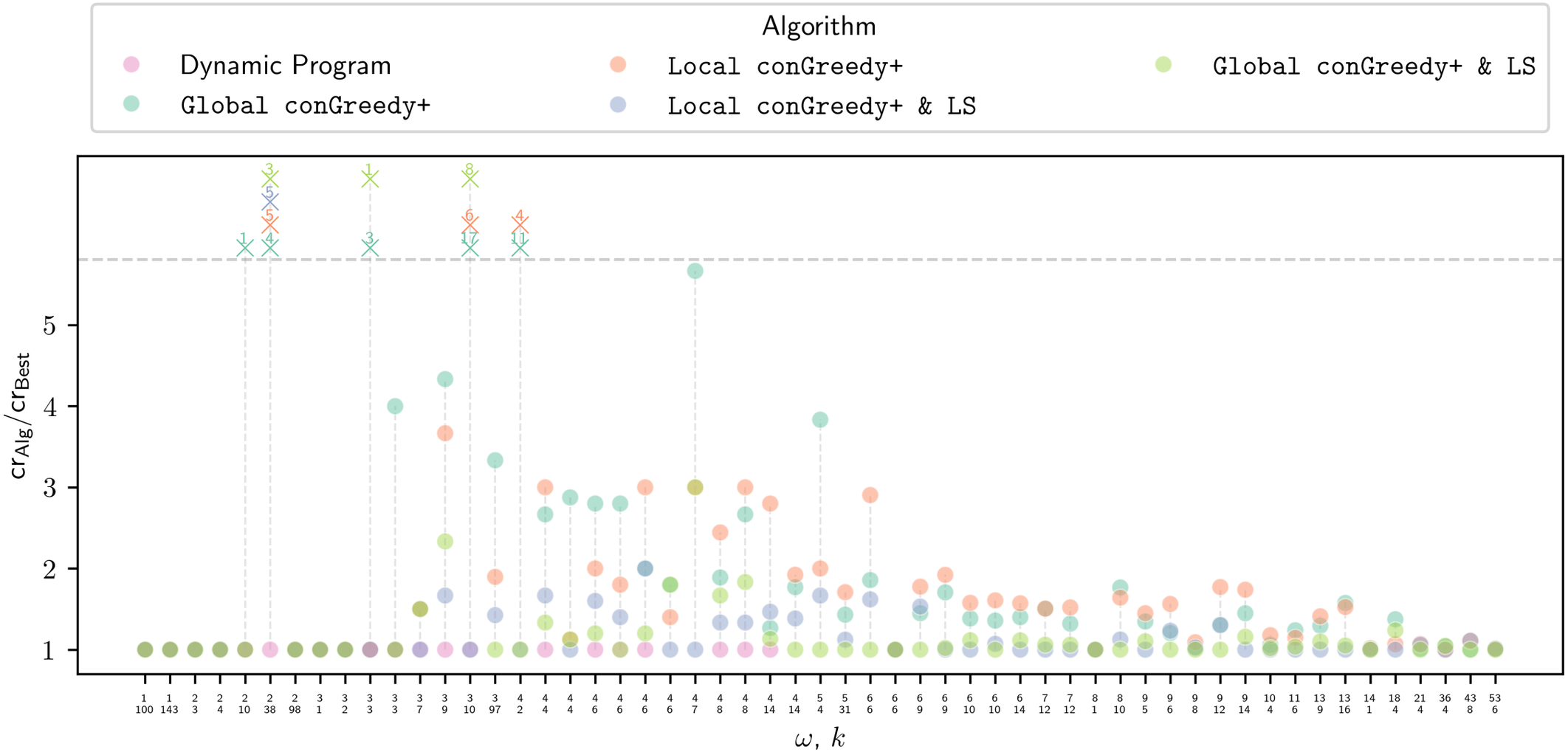
Dataset: 55 graphs from “Sage” with tree decomposition, 15min time limit



Heuristics usually take only few seconds

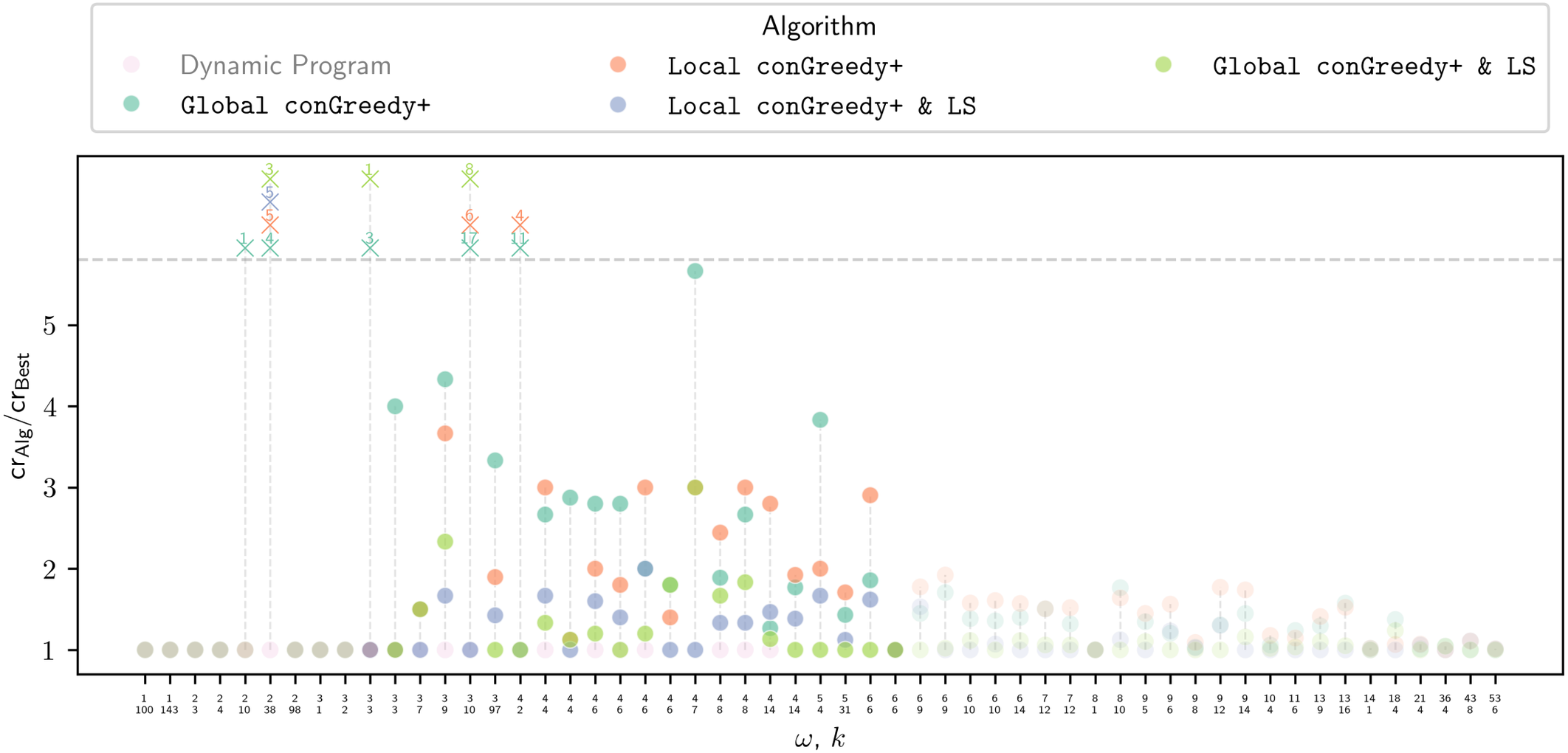
Experimental Evaluation – Crossings

Dataset: 55 graphs from “Sage” with tree decomposition, 15min time limit



Experimental Evaluation – Crossings

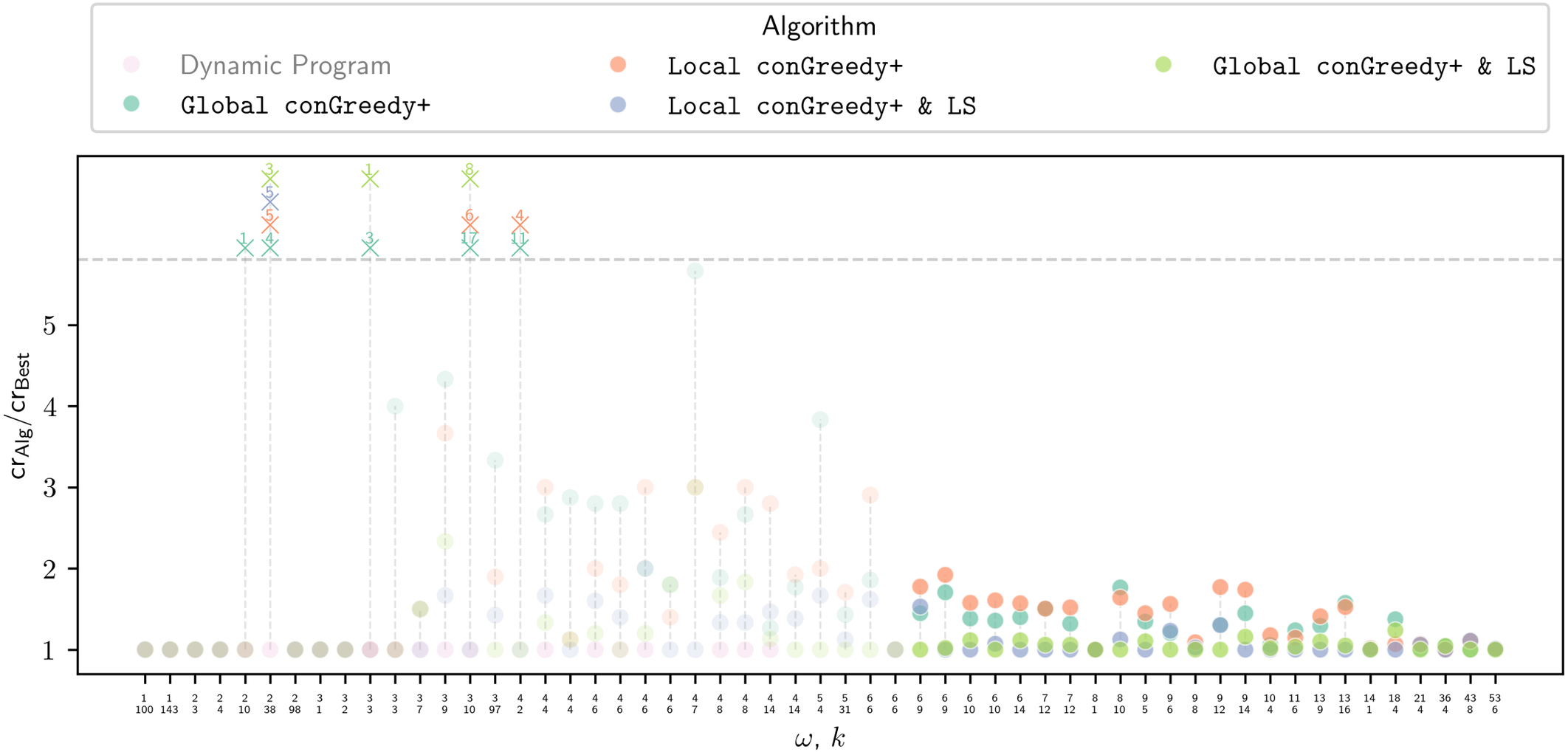
Dataset: 55 graphs from “Sage” with tree decomposition, 15min time limit



Medium-sized instances: local search helps (a lot)

Experimental Evaluation – Crossings

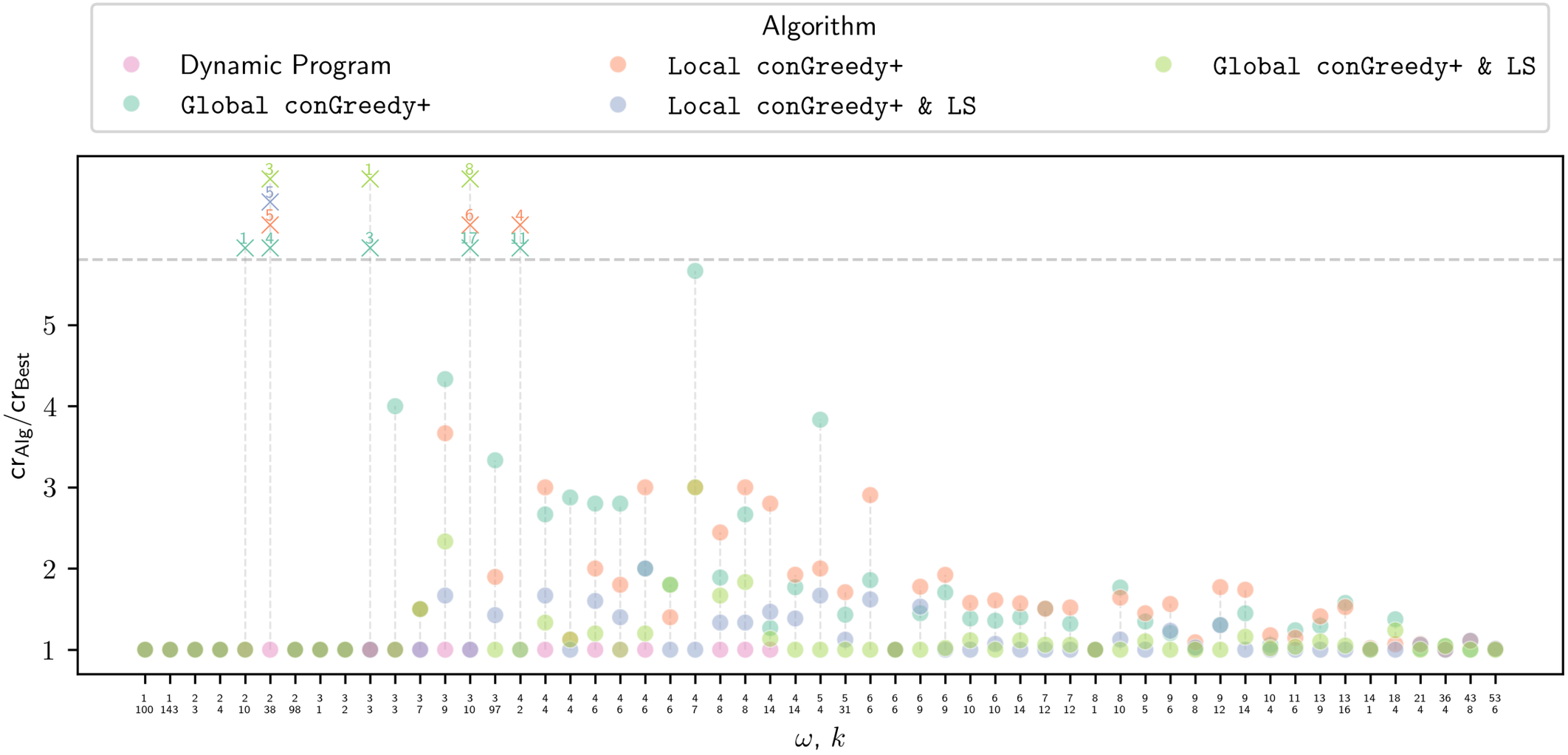
Dataset: 55 graphs from “Sage” with tree decomposition, 15min time limit



Large instances: small/no benefit of local search

Experimental Evaluation – Crossings

Dataset: 55 graphs from “Sage” with tree decomposition, 15min time limit



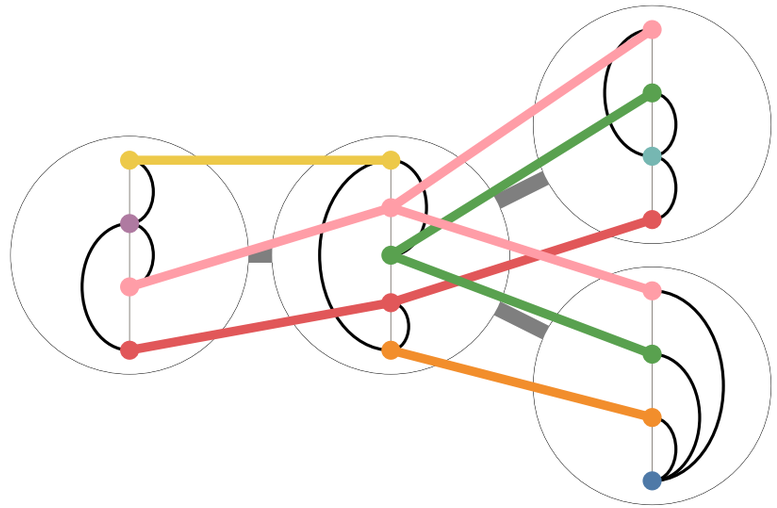
Time for drawings!

Witness Drawings for the Wagner Graph

Witness Drawings for the Wagner Graph

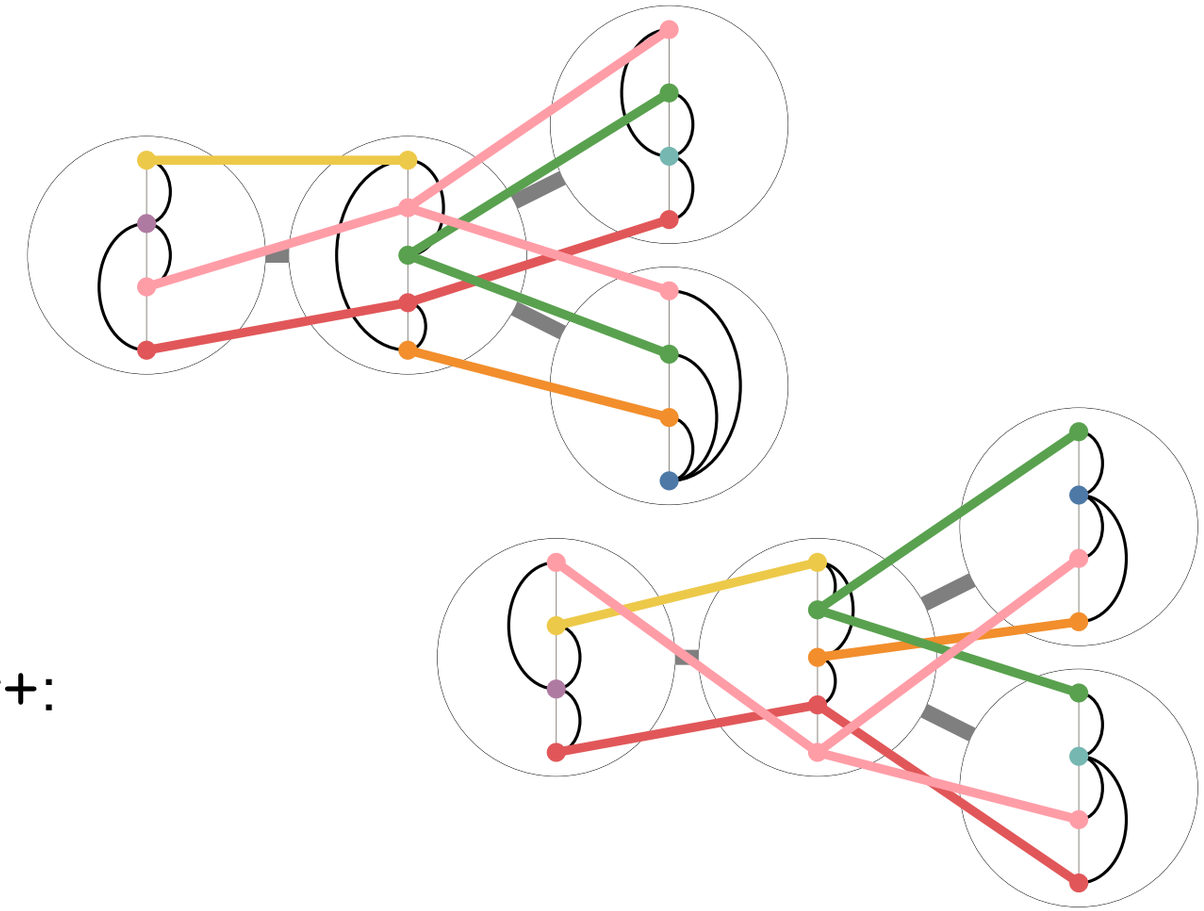
Witness Drawings for the Wagner Graph

`Global conGreedy+`:
8 crossings
1ms



Witness Drawings for the Wagner Graph

`Global conGreedy+`:
8 crossings
1ms



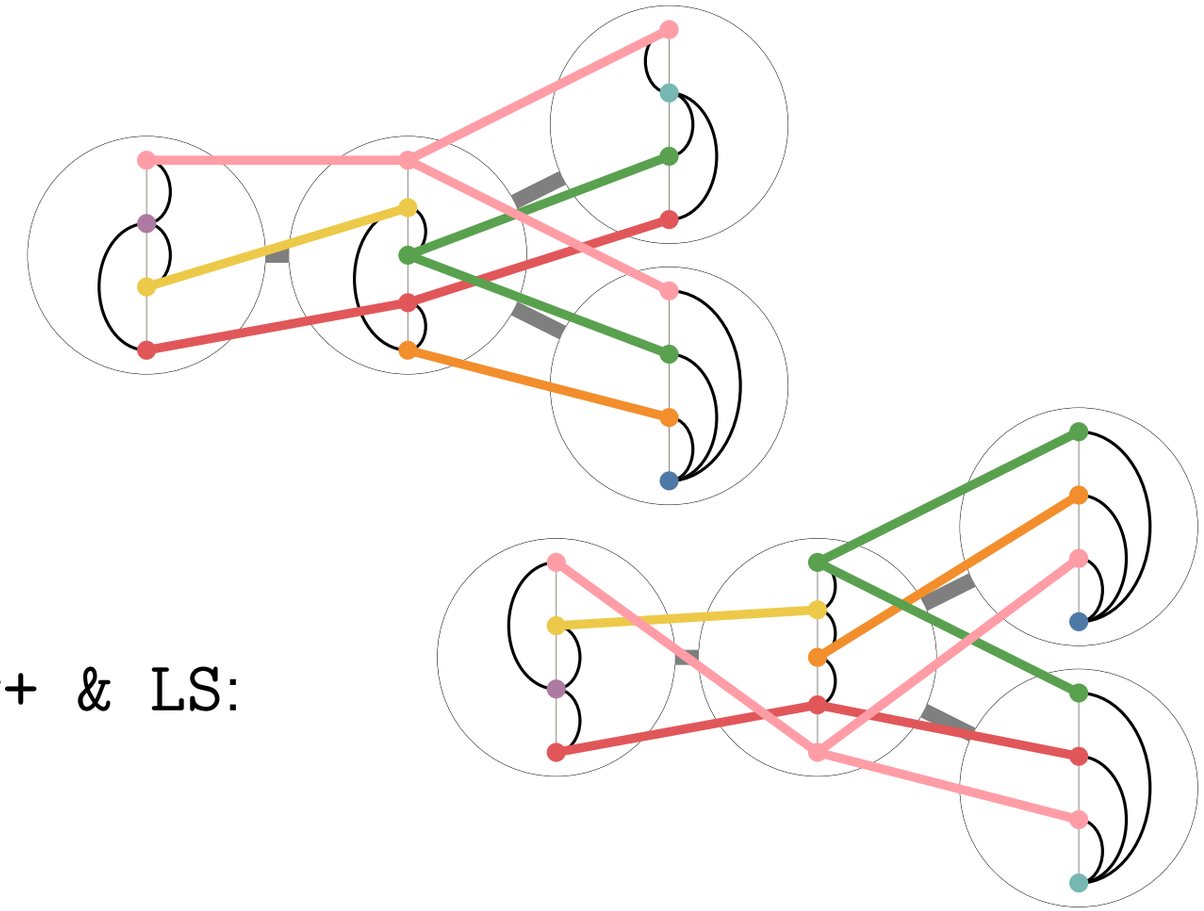
`Local conGreedy+`:
9 crossings
3ms

Witness Drawings for the Wagner Graph

`Global conGreedy+ & LS:`

~~8~~ $\rightarrow$ 4 crossings

~~1~~ $\rightarrow$ 135ms

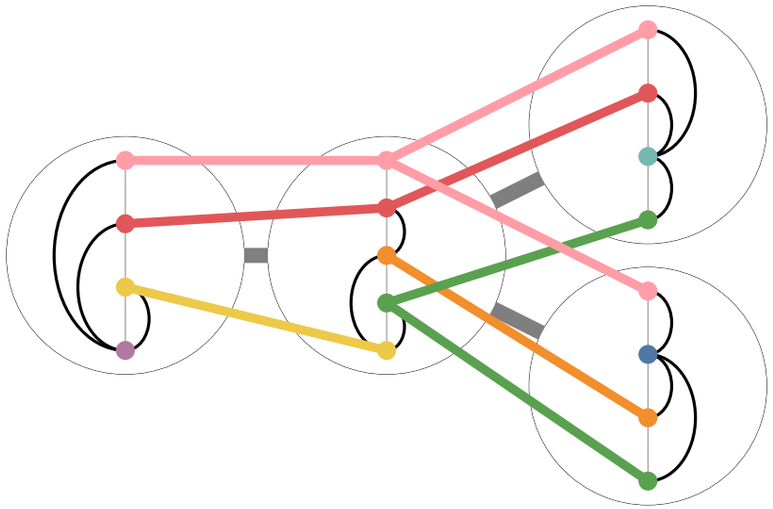


`Local conGreedy+ & LS:`

~~9~~ $\rightarrow$ 5 crossings

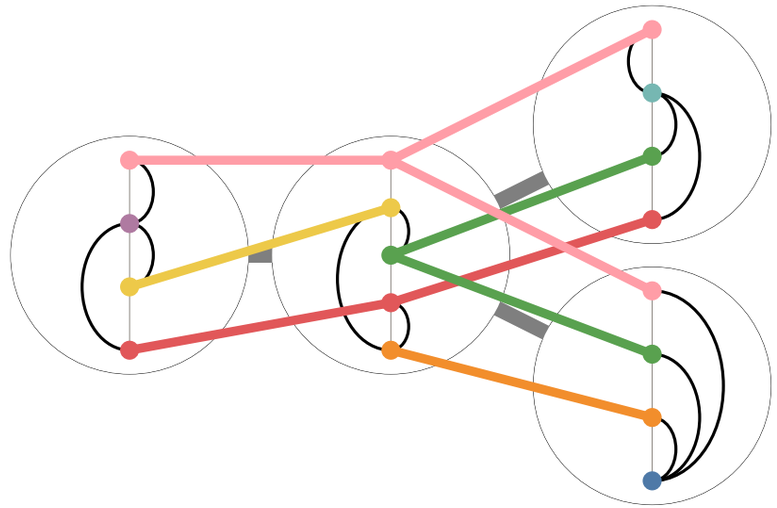
~~3~~ $\rightarrow$ 160ms

Witness Drawings for the Wagner Graph

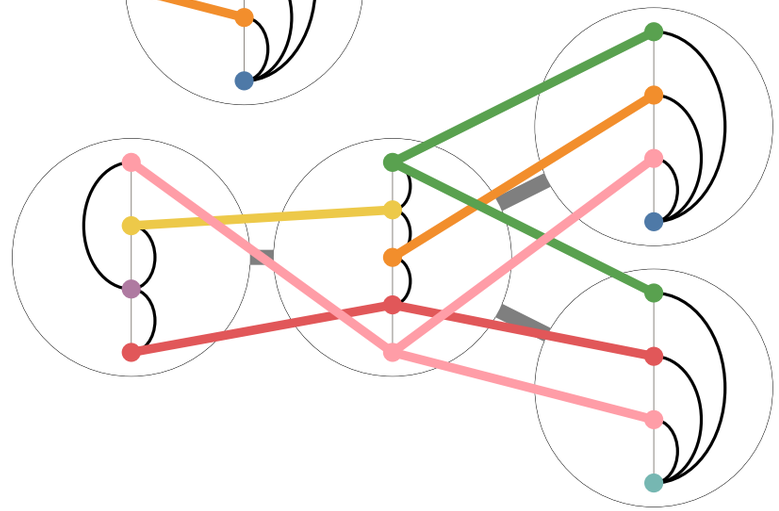


DP:
3 crossings
40min

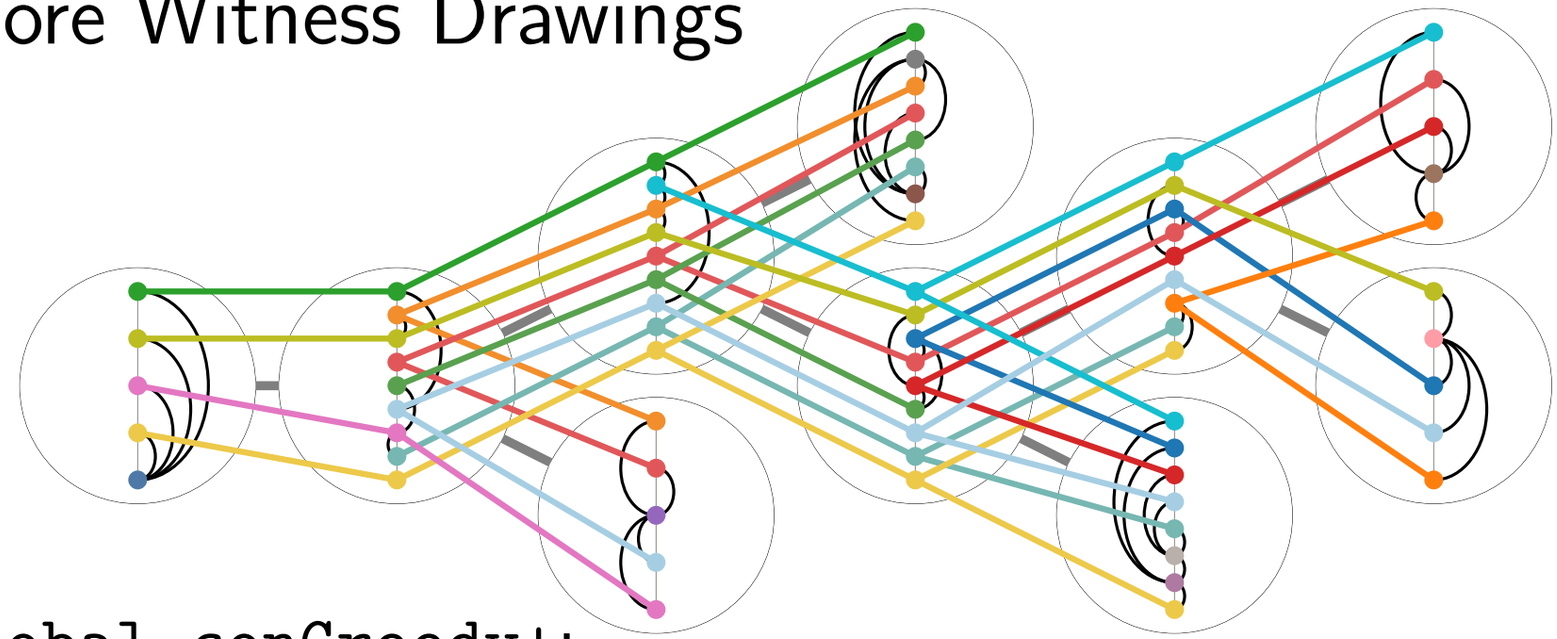
Global conGreedy+ & LS:
~~8~~ → 4 crossings
~~1~~ → 135ms

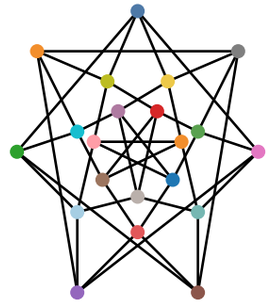


Local conGreedy+ & LS:
~~9~~ → 5 crossings
~~3~~ → 160ms

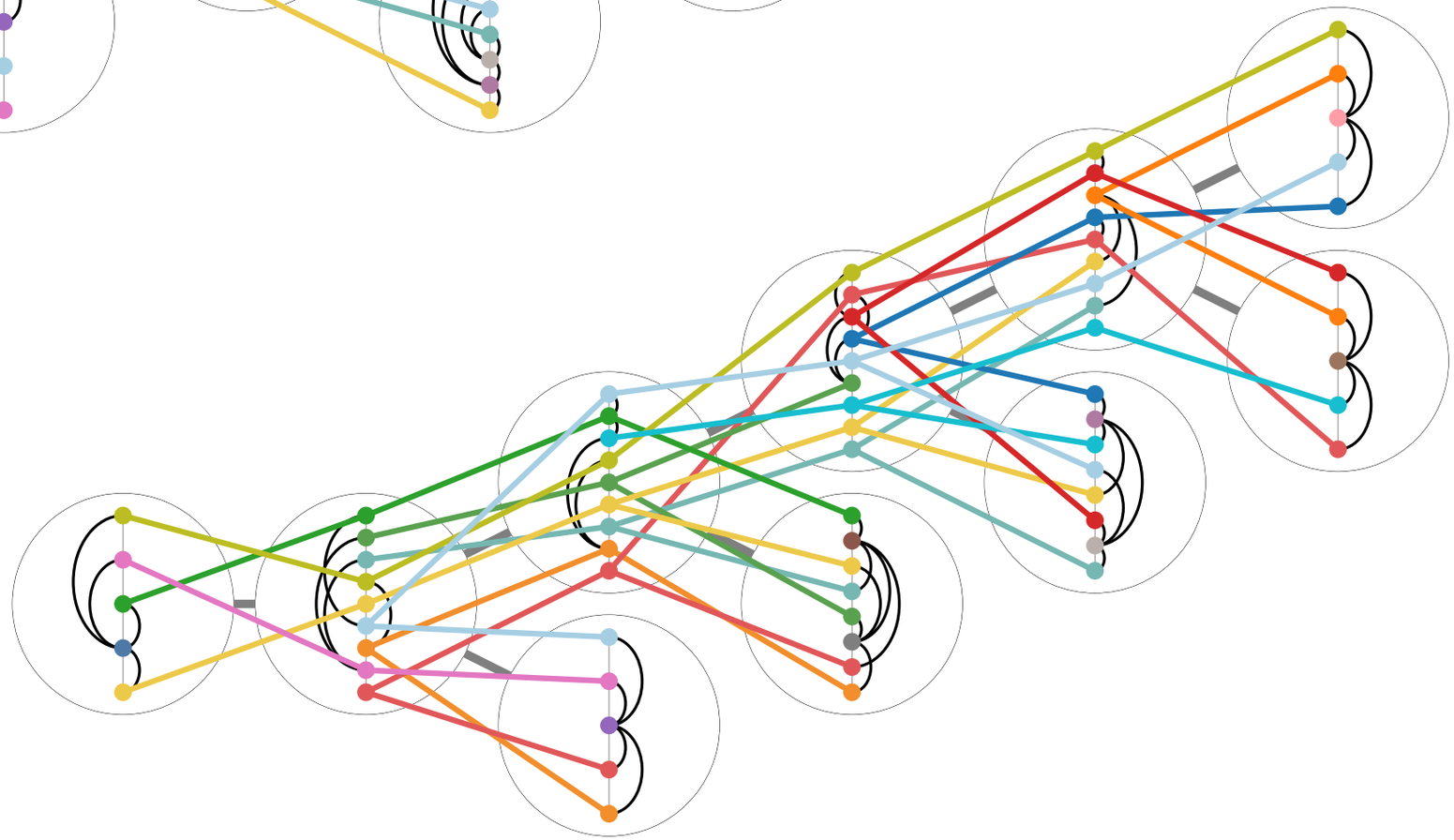


More Witness Drawings



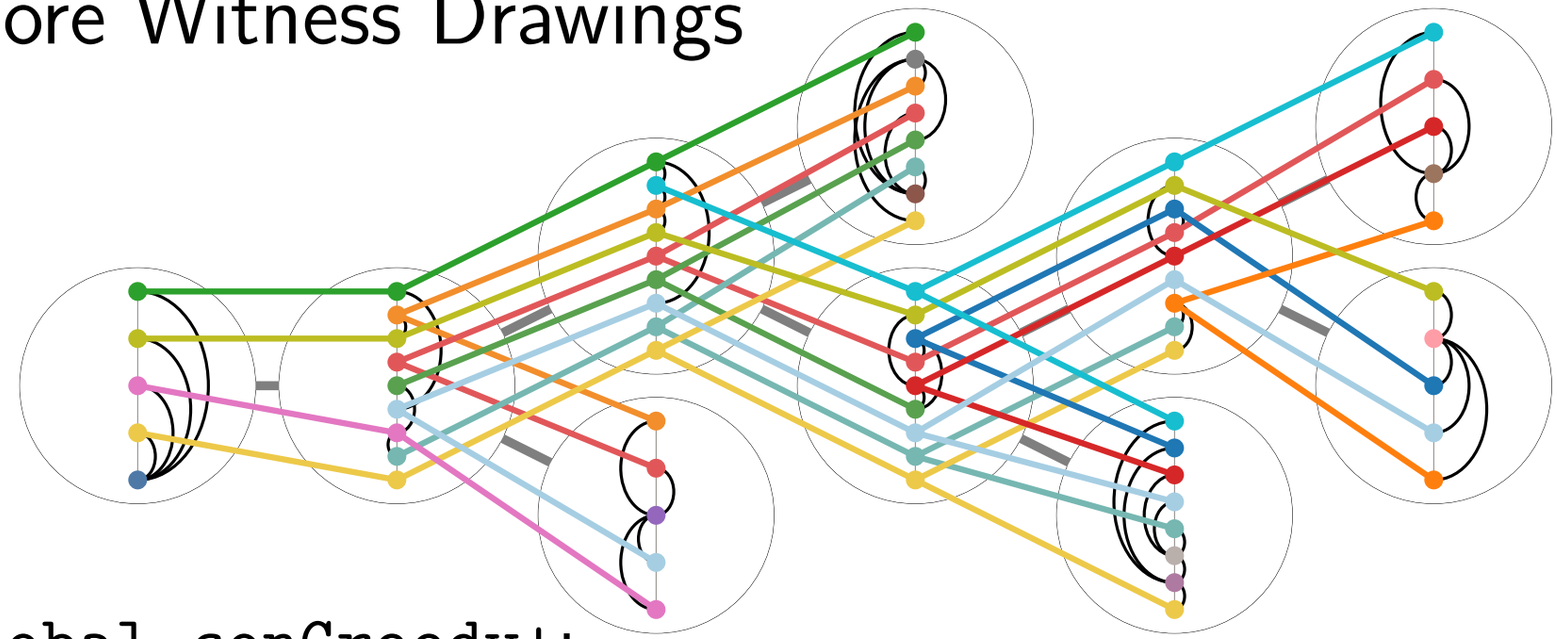


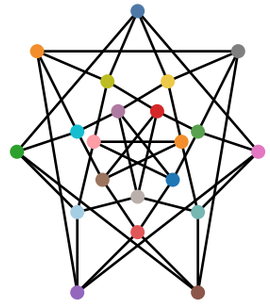
`Global conGreedy+`:
113 crossings
8ms



`Local conGreedy+`:
105 crossings
19ms

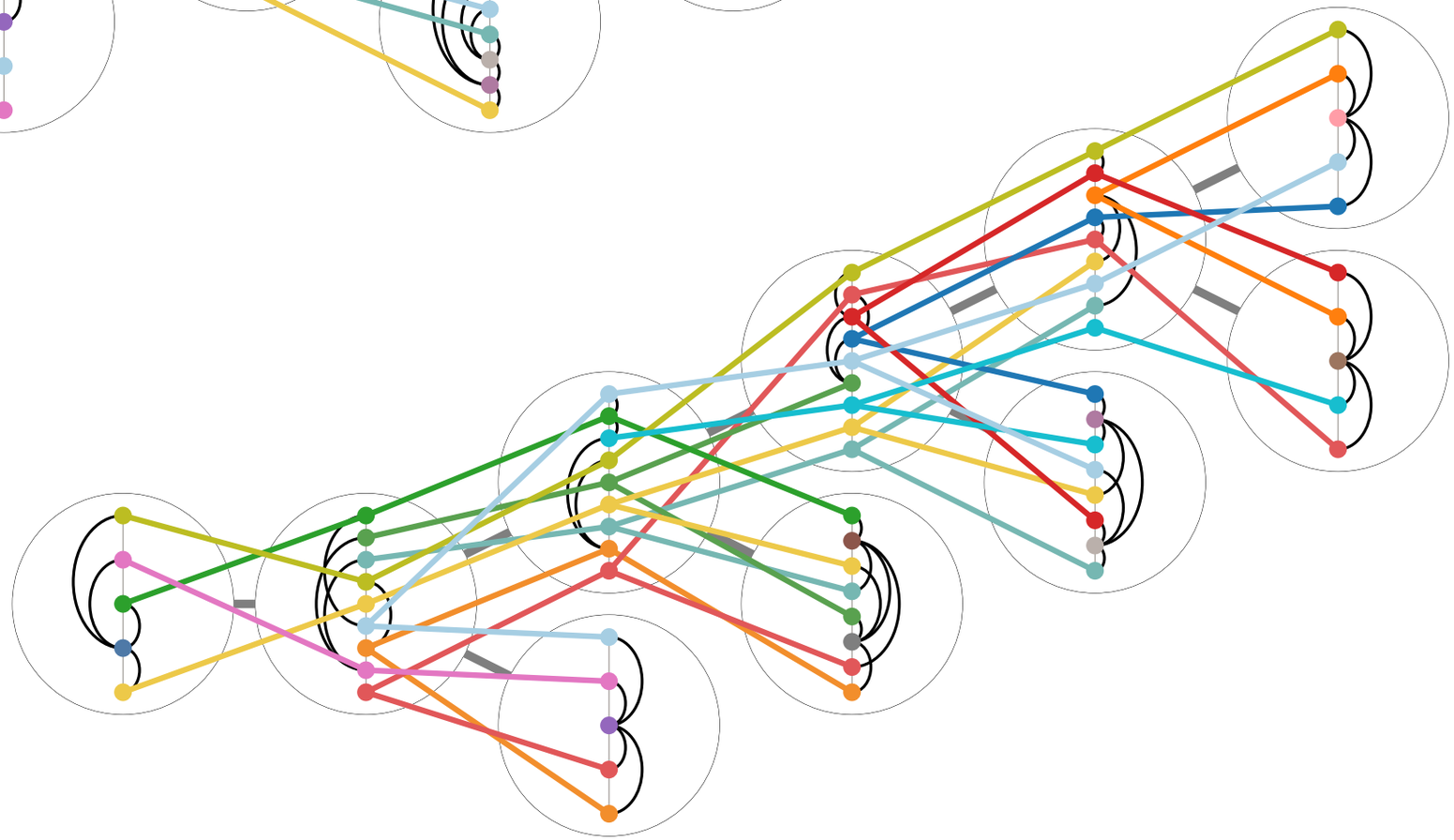
More Witness Drawings





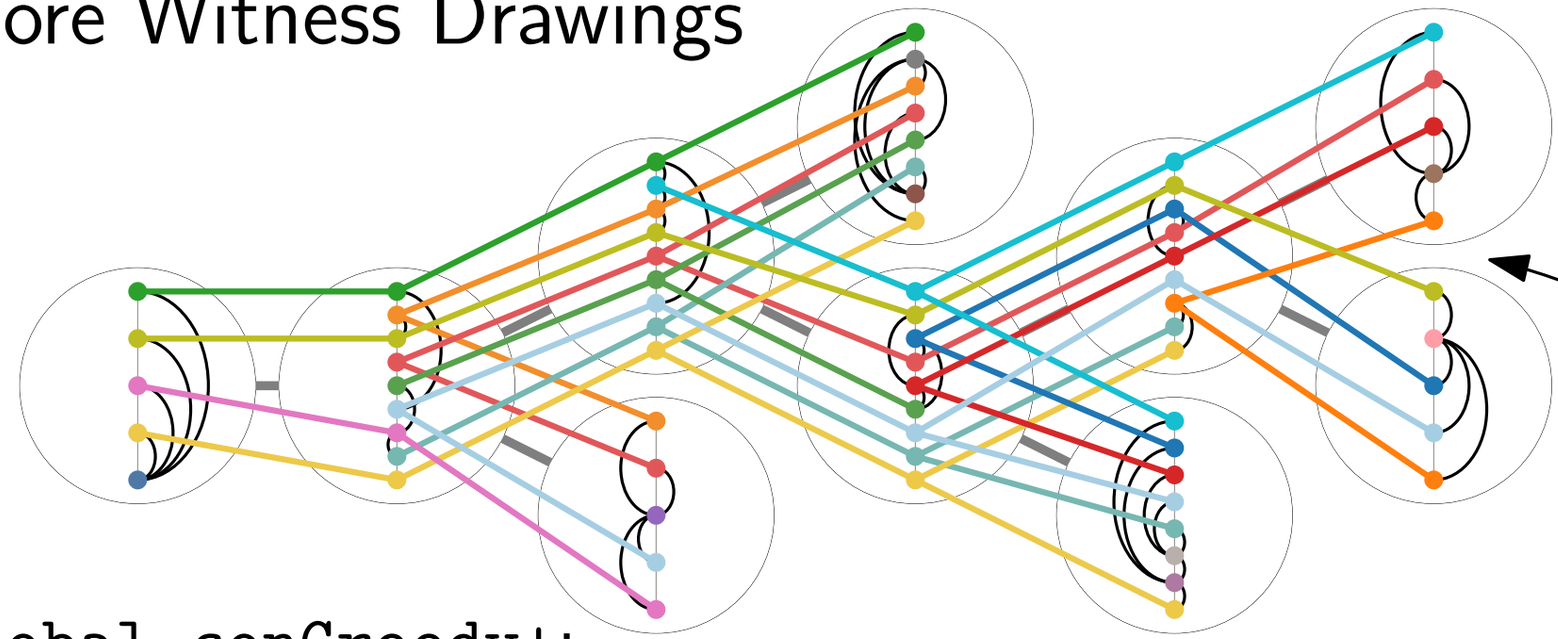
More readable?

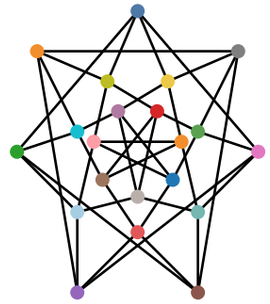
`Global conGreedy+`:
113 crossings
8ms



`Local conGreedy+`:
105 crossings
19ms

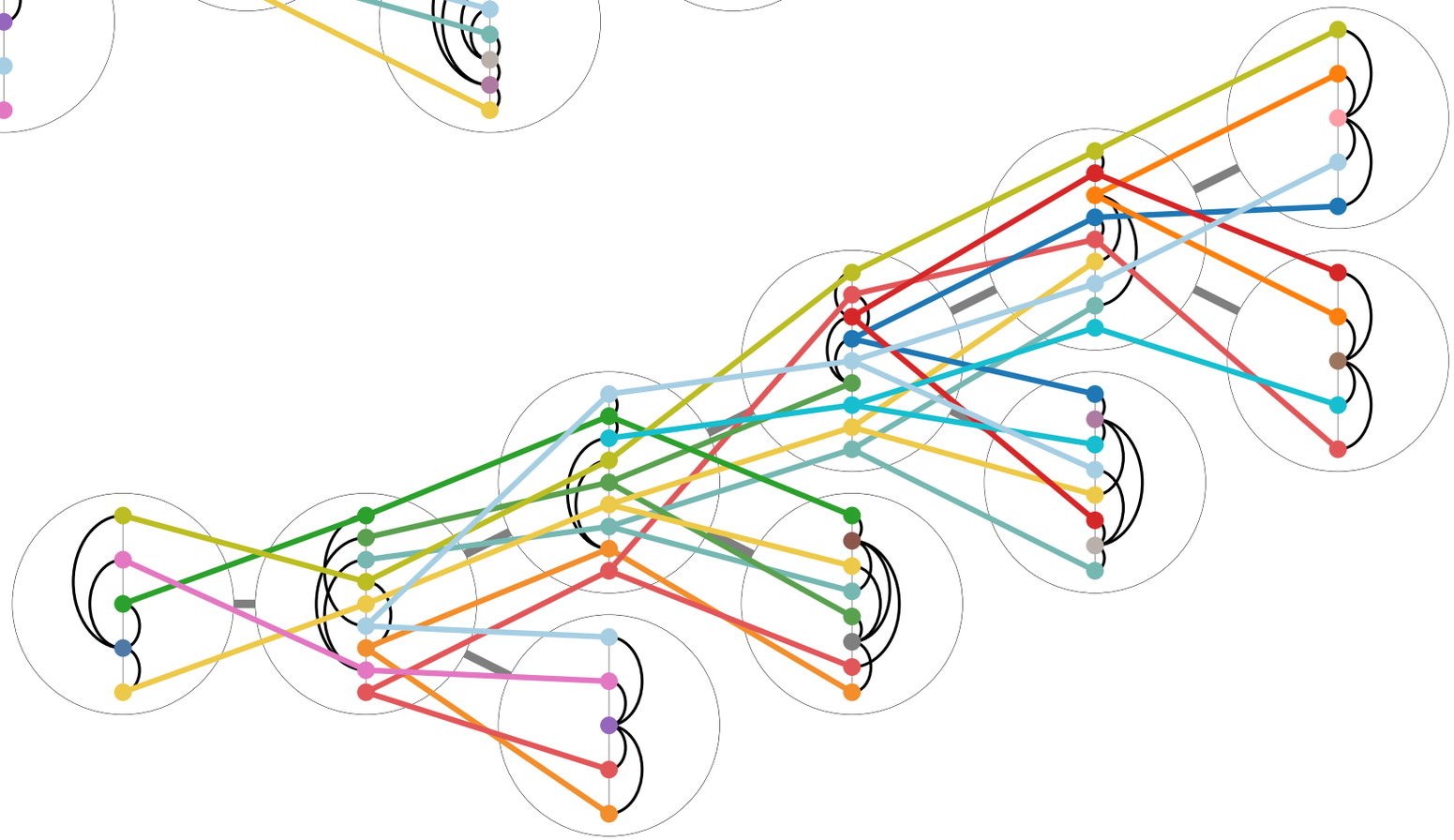
More Witness Drawings





More readable?

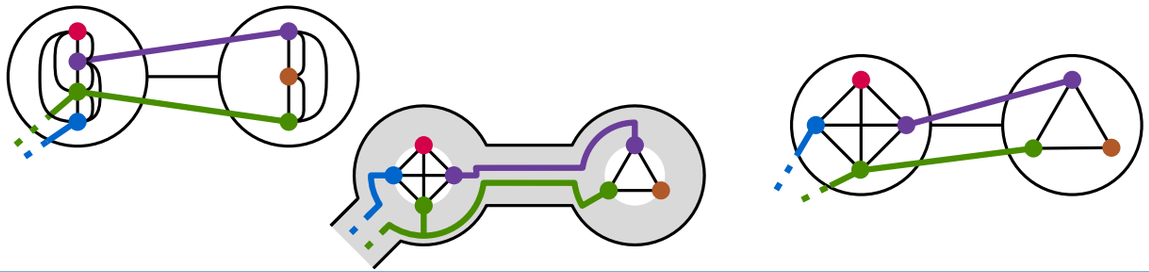
`Global conGreedy+`:
113 crossings
8ms



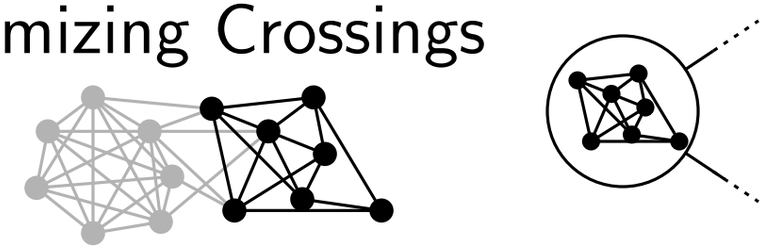
`Local conGreedy+`:
105 crossings
19ms

Conclusion

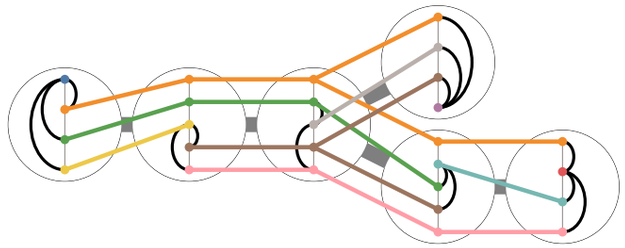
Drawing Paradigms for Treewidth



Complexity and Algorithms for Minimizing Crossings

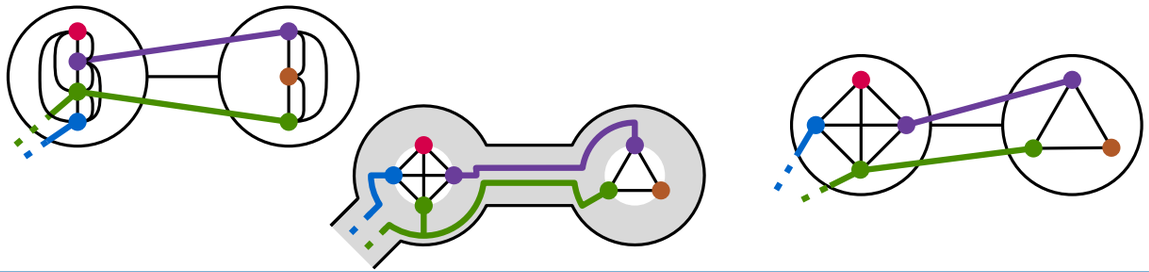


Experimental Evaluation

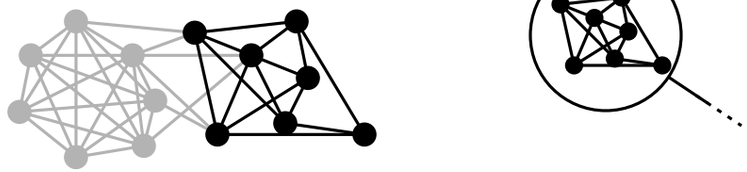


Conclusion

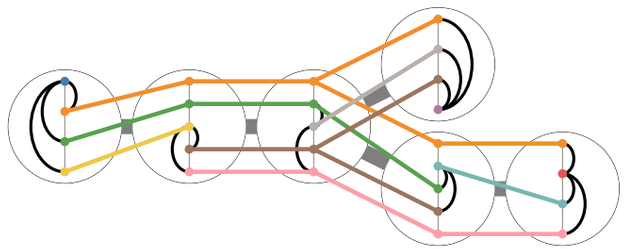
Drawing Paradigms for Treewidth



Complexity and Algorithms for Minimizing Crossings



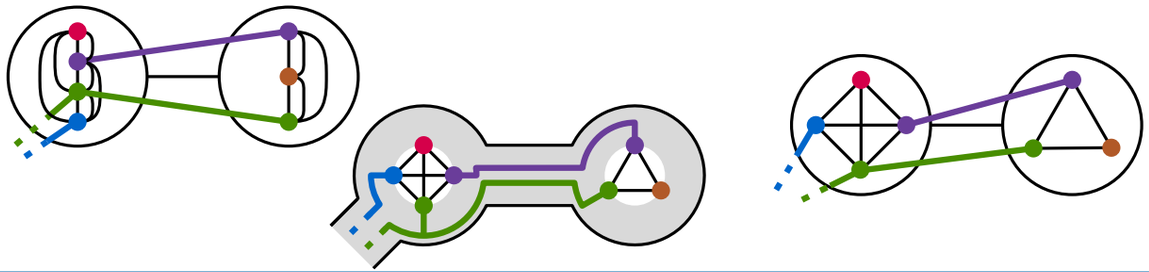
Experimental Evaluation



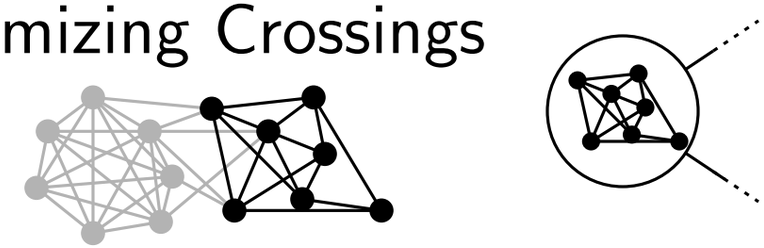
Prototype

Conclusion

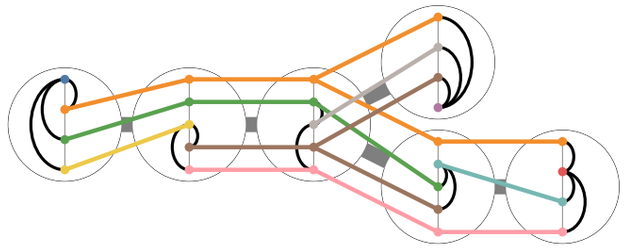
Drawing Paradigms for Treewidth



Complexity and Algorithms for Minimizing Crossings



Experimental Evaluation

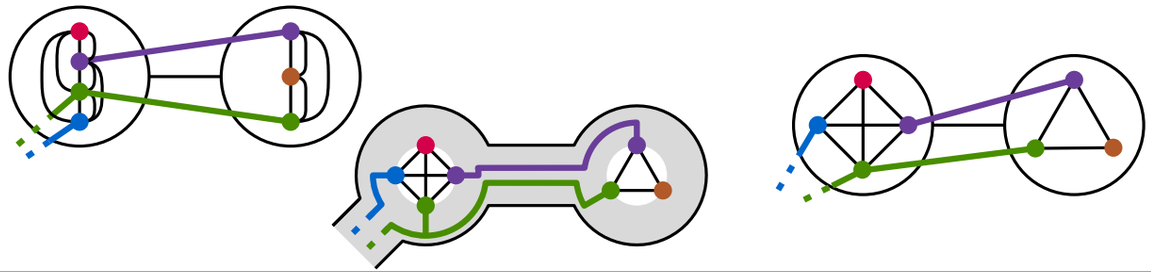


Prototype

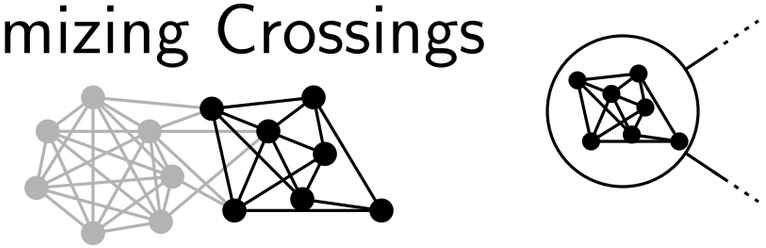
Future Work

Conclusion

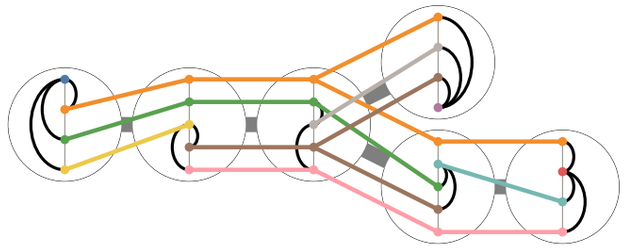
Drawing Paradigms for Treewidth



Complexity and Algorithms for Minimizing Crossings



Experimental Evaluation



Prototype

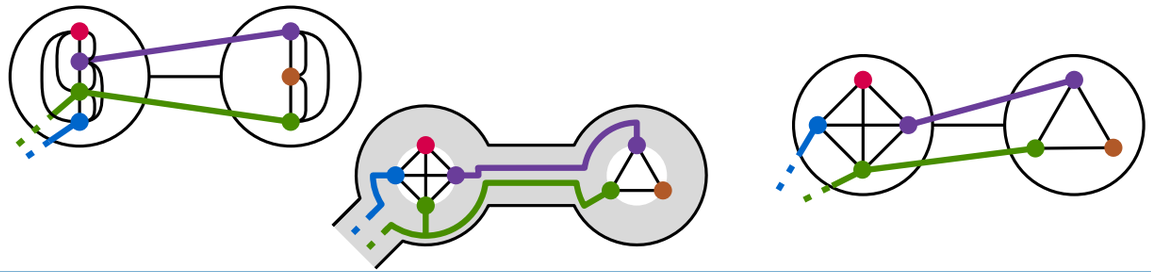
Future Work

Further Explore Design Space

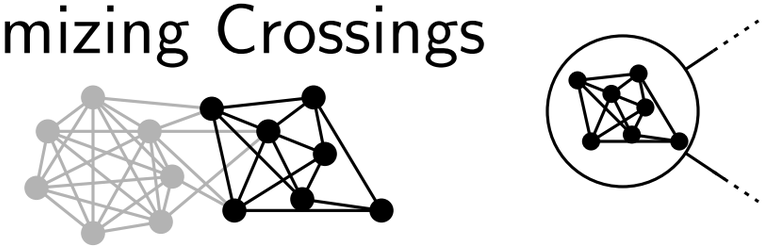
Other Optimization Criteria

Conclusion

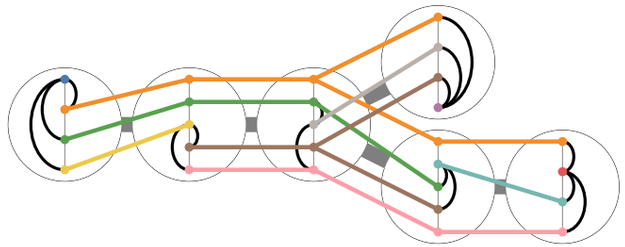
Drawing Paradigms for Treewidth



Complexity and Algorithms for Minimizing Crossings



Experimental Evaluation



Prototype

Future Work

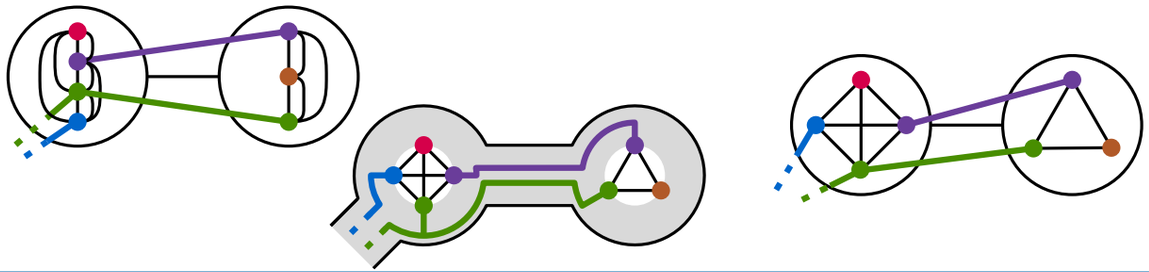
Further Explore Design Space

Other Optimization Criteria

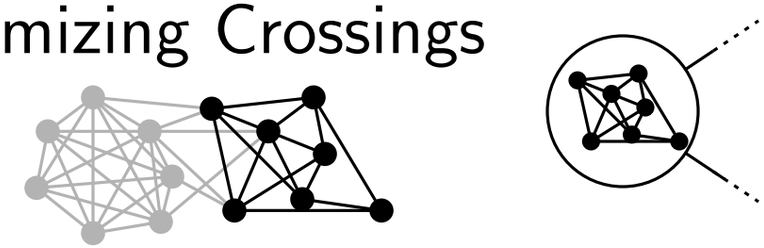
Trees of Larger Degree

Conclusion

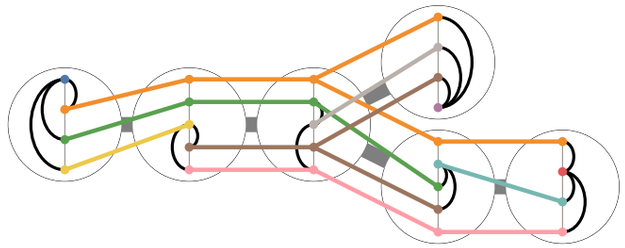
Drawing Paradigms for Treewidth



Complexity and Algorithms for Minimizing Crossings



Experimental Evaluation



Prototype

Future Work

Further Explore Design Space

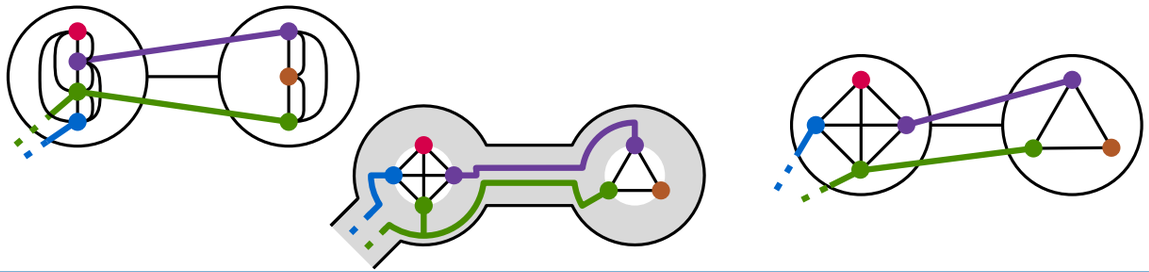
Other Optimization Criteria

Trees of Larger Degree

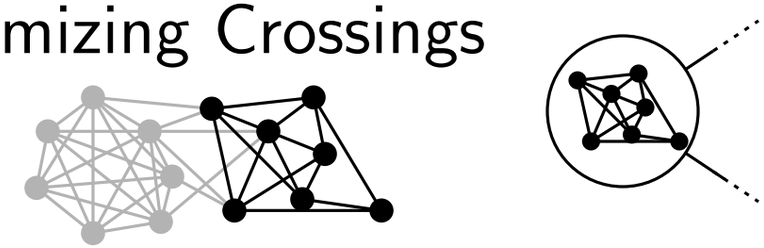
Witness Drawings for other Graph Parameters

Conclusion

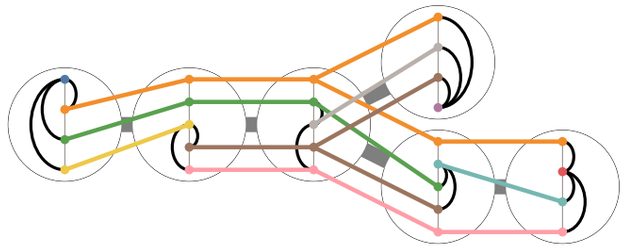
Drawing Paradigms for Treewidth



Complexity and Algorithms for Minimizing Crossings



Experimental Evaluation



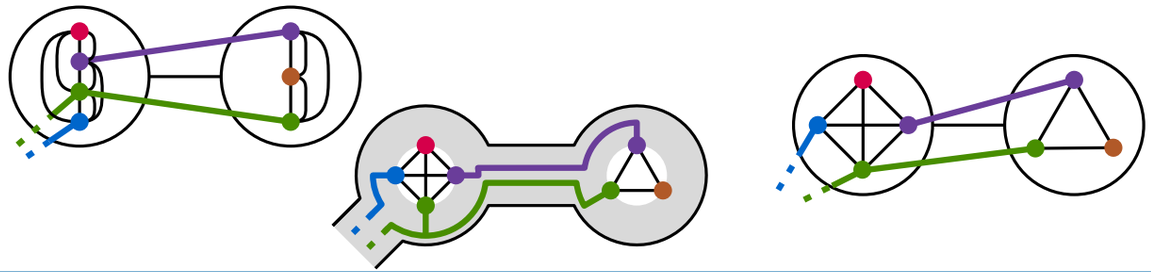
Prototype

Future Work

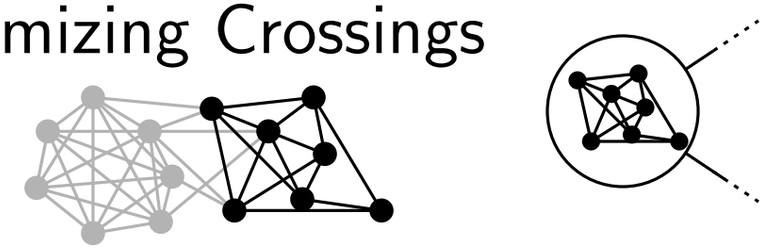
- Further Explore Design Space
- Other Optimization Criteria
- Trees of Larger Degree
- Witness Drawings for other Graph Parameters
- User Study

Conclusion

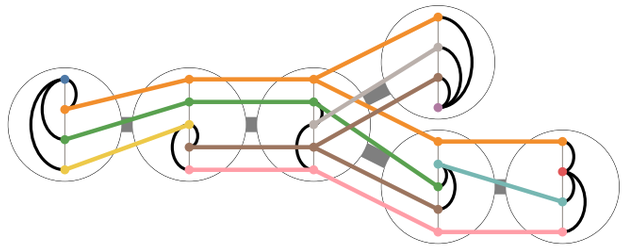
Drawing Paradigms for Treewidth



Complexity and Algorithms for Minimizing Crossings



Experimental Evaluation



Prototype

Future Work

- Further Explore Design Space
- Other Optimization Criteria
- Trees of Larger Degree
- Witness Drawings for other Graph Parameters
- User Study

Thank you for your attention!

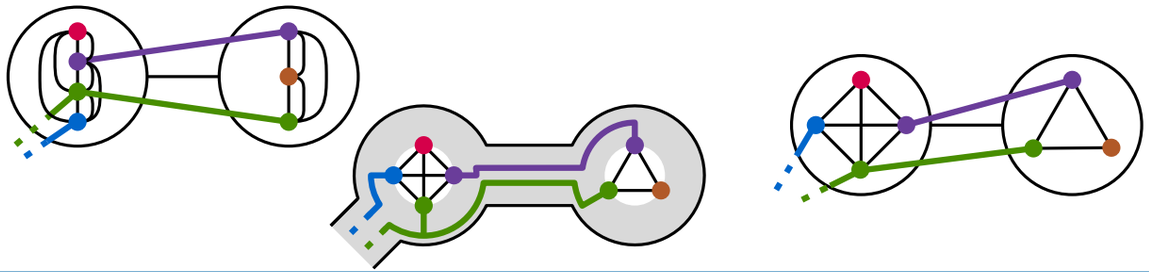
Invitation:

Participate in a **User Study** on Treewidth Visualizations

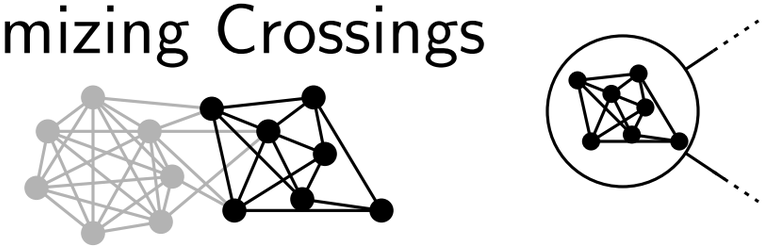
https://url.tuwien.at/yqgbg

Conclusion

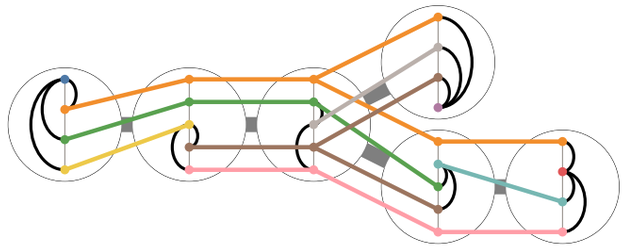
Drawing Paradigms for Treewidth



Complexity and Algorithms for Minimizing Crossings



Experimental Evaluation



Prototype

Future Work

Further Explore Design Space

Other Optimization Criteria

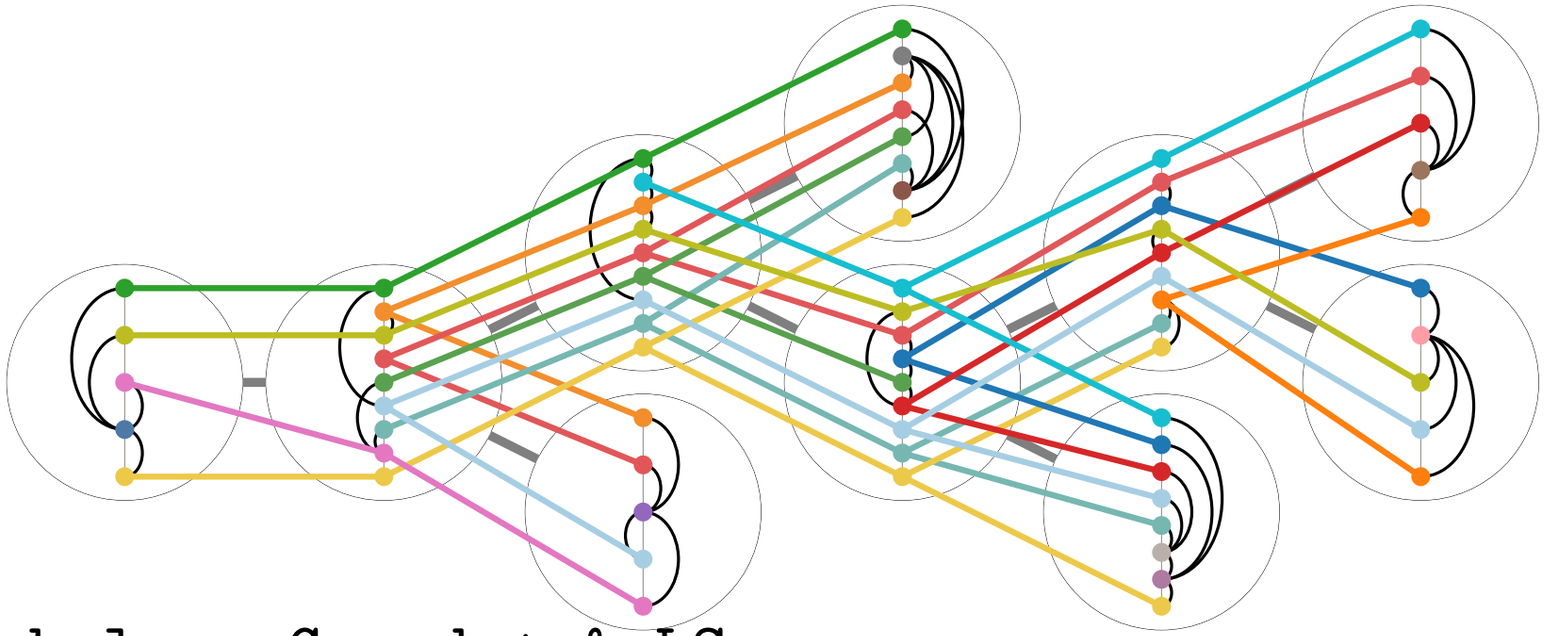
Trees of Larger Degree

Witness Drawings for other Graph Parameters

User Study →

Thank you for your attention!

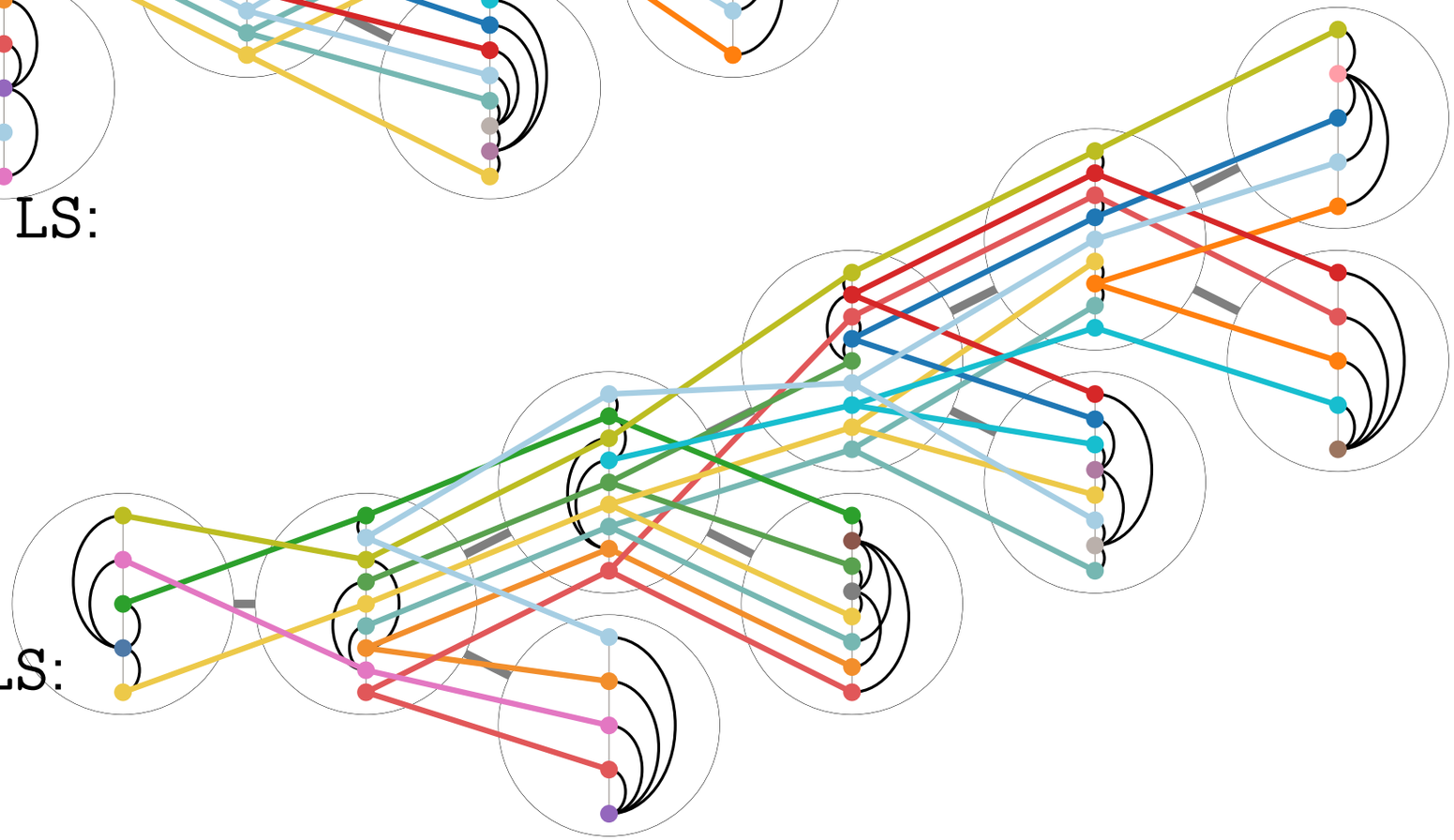
More Witness Drawings (LS)



`Global conGreedy+ & LS:`

64 crossings

970ms



`Local conGreedy+ & LS:`

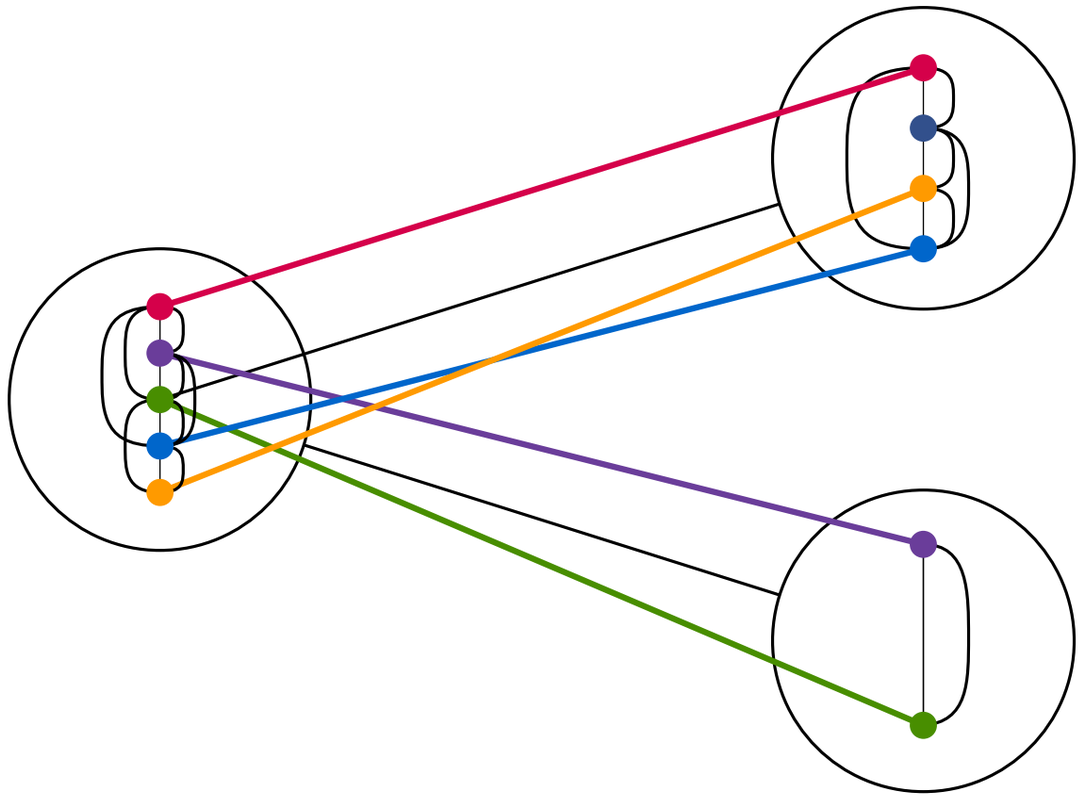
72 crossings

1s

Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

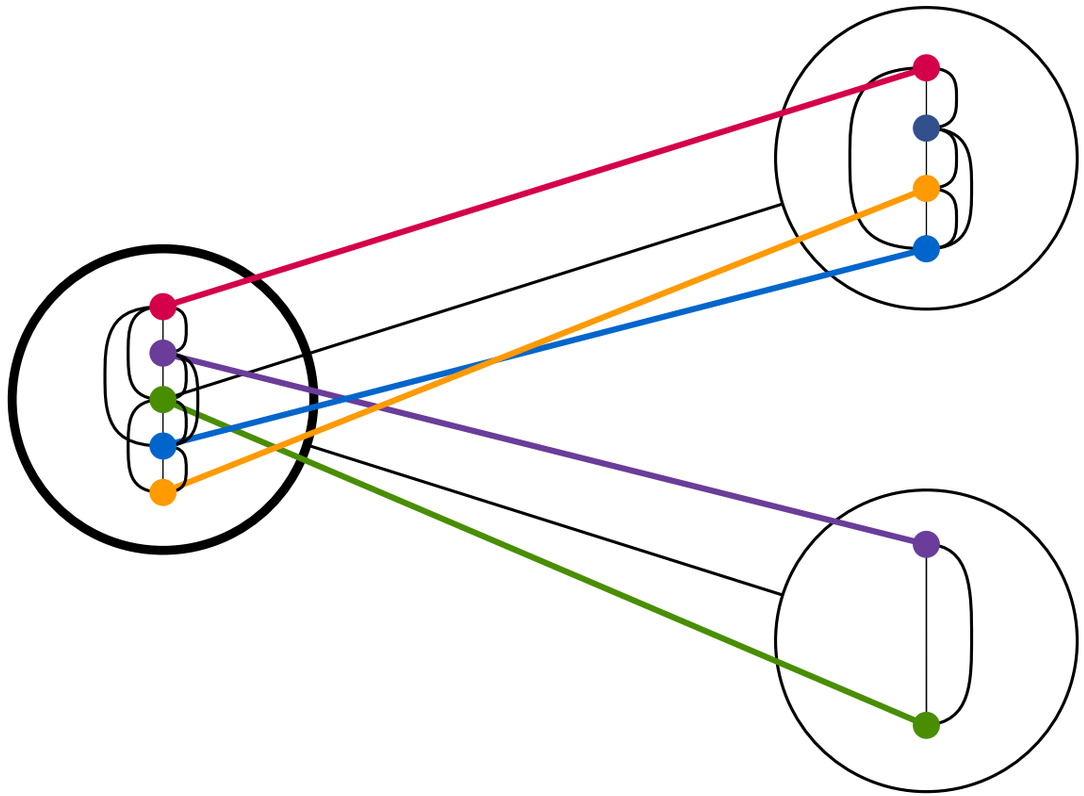
Theorem:
A crossing-minimal linear witness drawing of a tree decomposition of width ω can be computed in $\mathcal{O}(f(\omega) \cdot n)$ time.



Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

Theorem:
A crossing-minimal linear witness drawing of a tree decomposition of width ω can be computed in $\mathcal{O}(f(\omega) \cdot n)$ time.

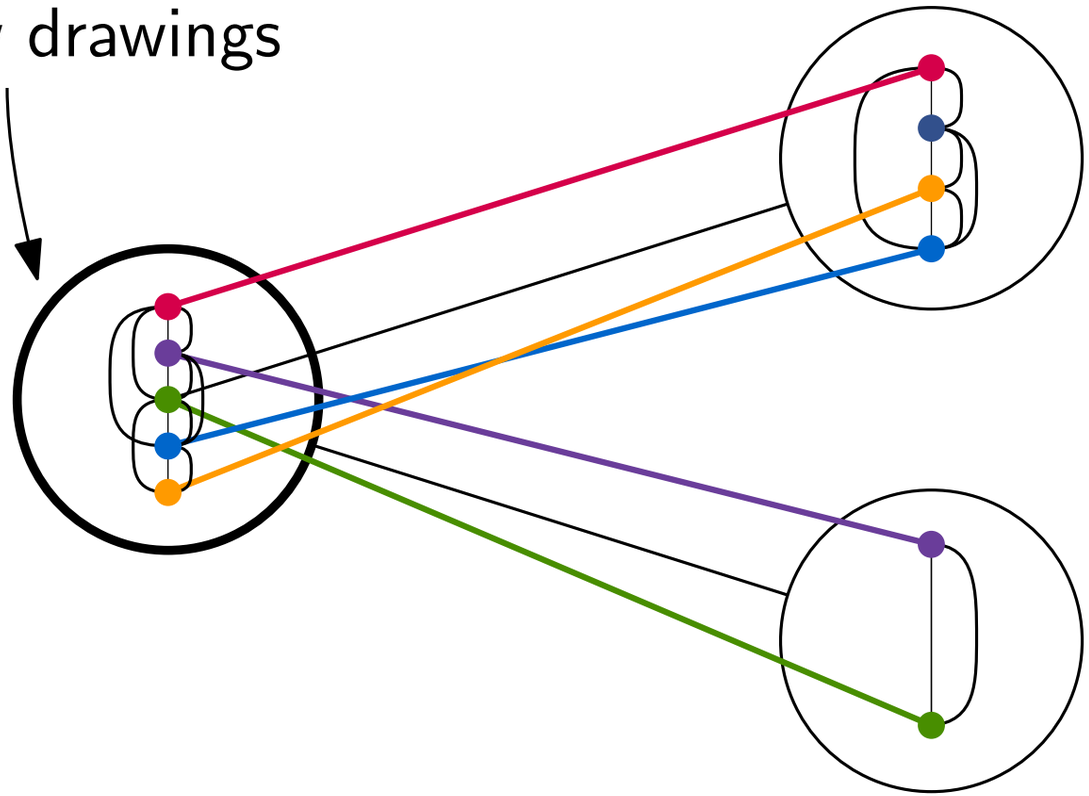


Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

Theorem:
A crossing-minimal linear witness drawing of a tree decomposition of width ω can be computed in $\mathcal{O}(f(\omega) \cdot n)$ time.

Few drawings

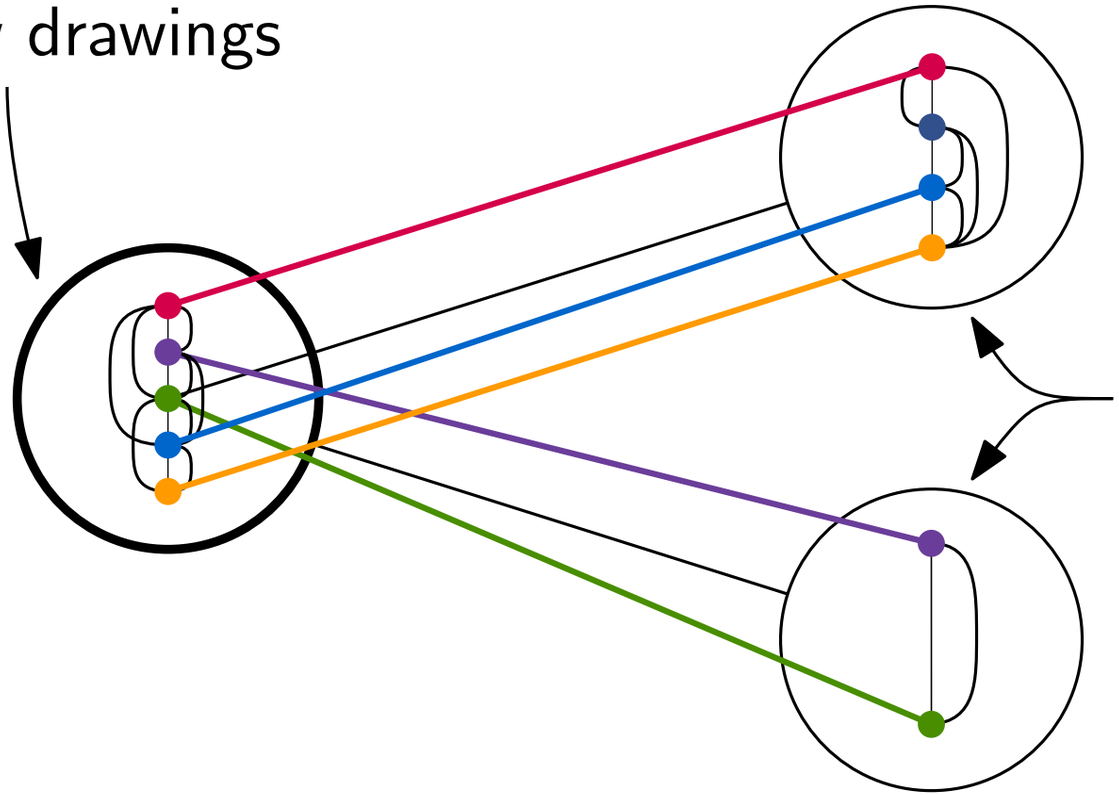


Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

Theorem:
A crossing-minimal linear witness drawing of a tree decomposition of width ω can be computed in $\mathcal{O}(f(\omega) \cdot n)$ time.

Few drawings

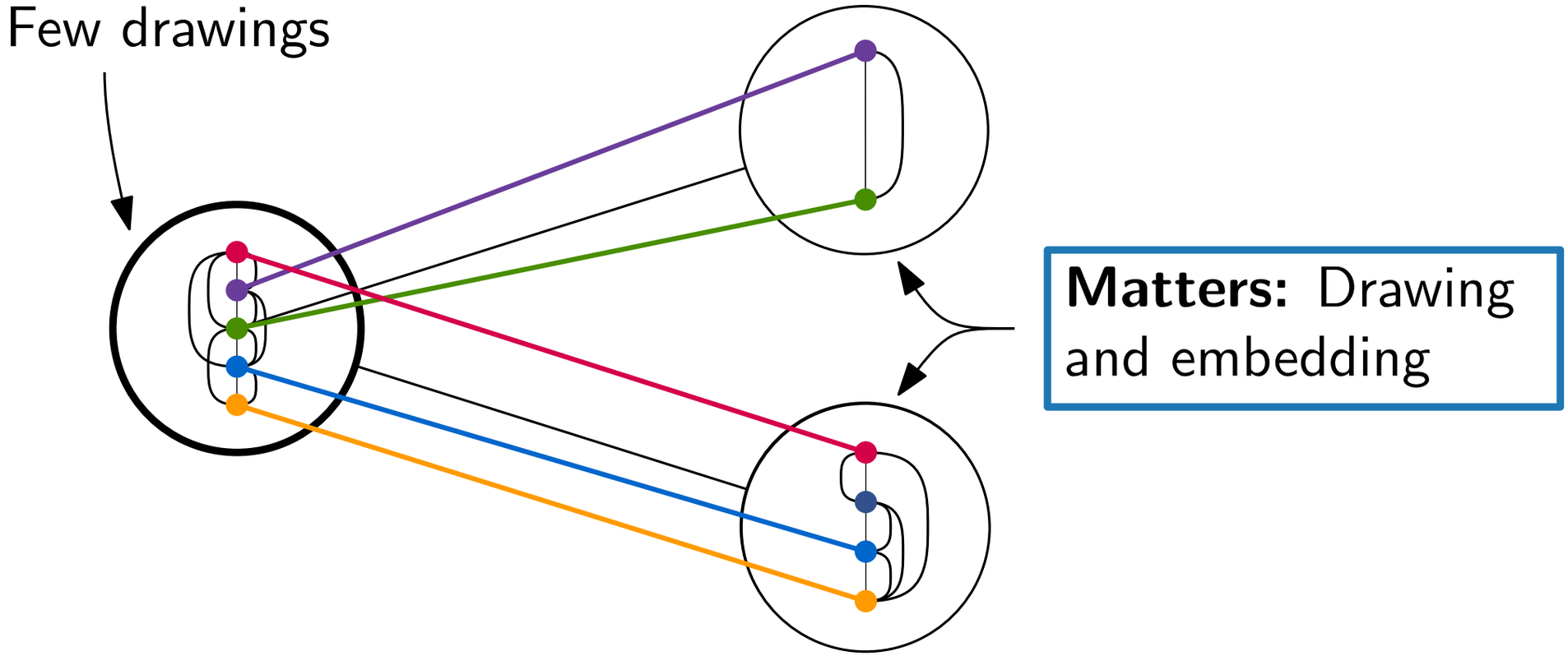


Matters: Drawing

Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

Theorem:
A crossing-minimal linear witness drawing of a tree decomposition of width ω can be computed in $\mathcal{O}(f(\omega) \cdot n)$ time.



Crossing Minimization for Linear Drawings

Goal: Minimize Σ (t/t, t/e, e/e-crossings)

Theorem:
A crossing-minimal linear witness drawing of a tree decomposition of width ω can be computed in $\mathcal{O}(f(\omega) \cdot n)$ time.