Planar Packing of Binary Trees

Markus Geyer[1] Michael Hoffmann[2] Michael Kaufmann[1]
Vincent Kusters[2] Csaba D. Tóth[3]

Universität Tübingen

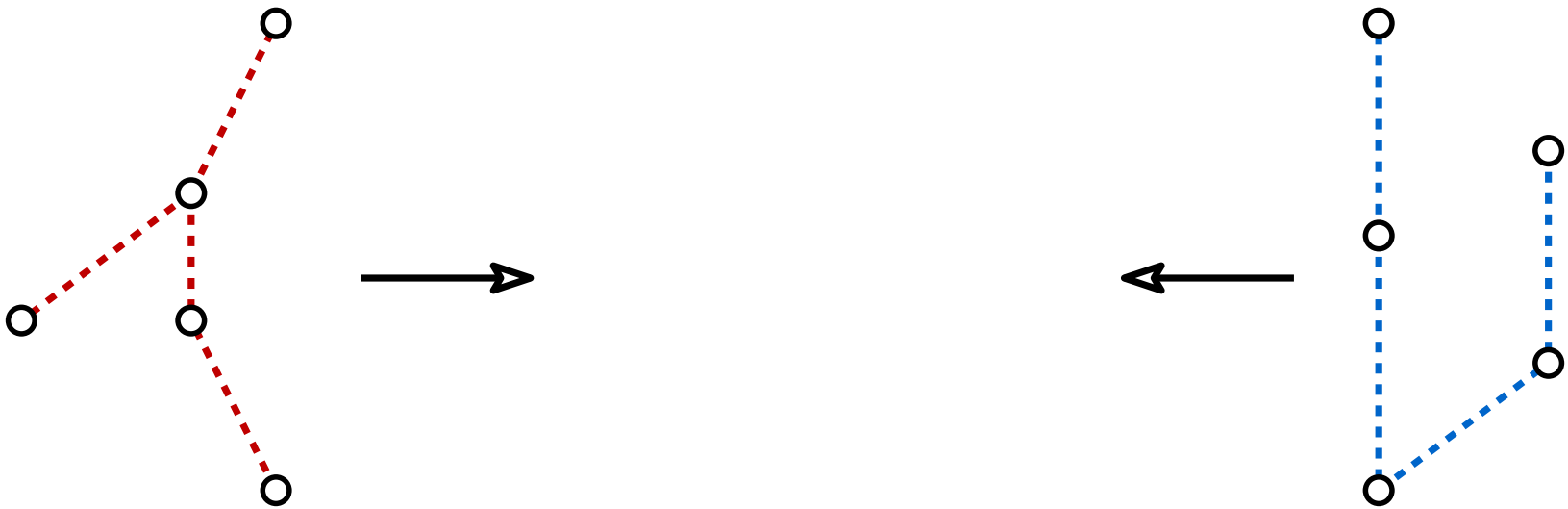
ETH Zürich

University of Calgary, California State University Northridge, and Tufts University Boston

February 18, 2014

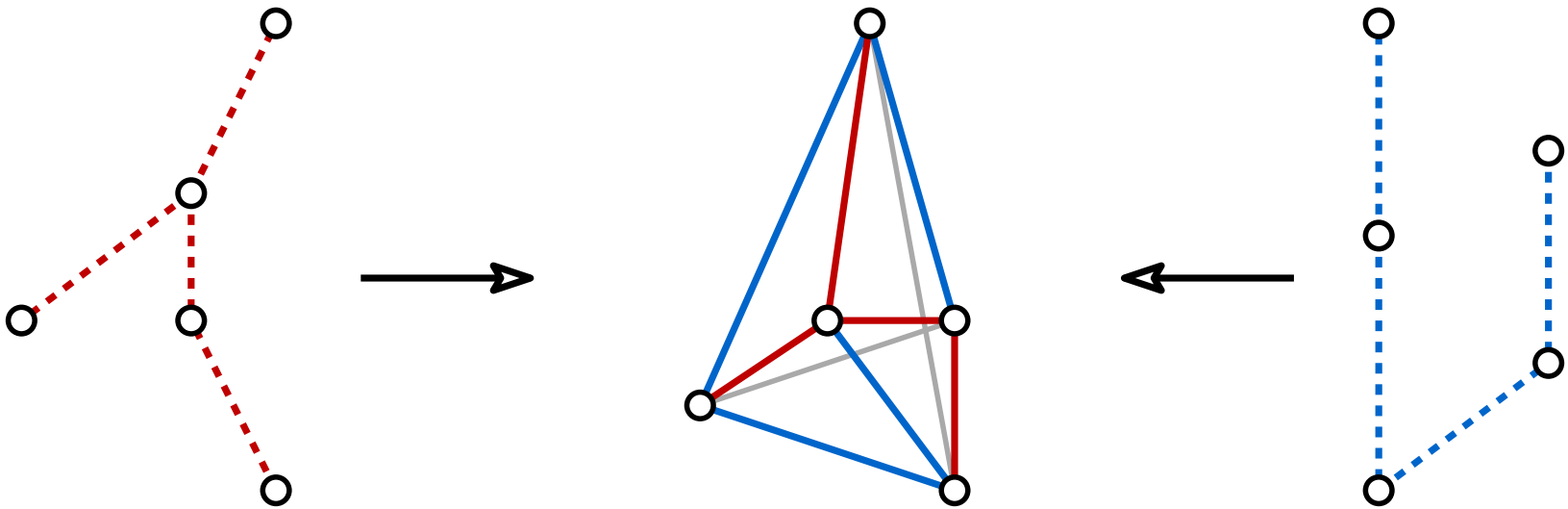
Introduction

Graph packing: given k graphs on n vertices, is there a graph on n vertices that contains them as edge-disjoint subgraphs?



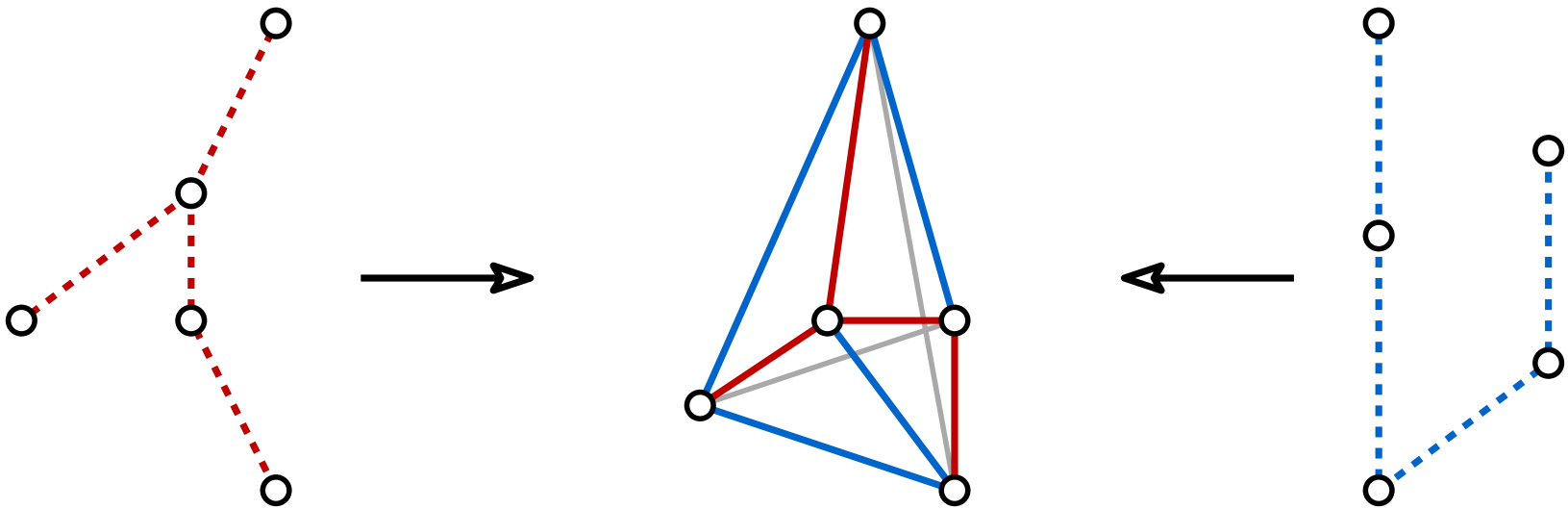
Introduction

Graph packing: given k graphs on n vertices, is there a graph on n vertices that contains them as edge-disjoint subgraphs?



Introduction

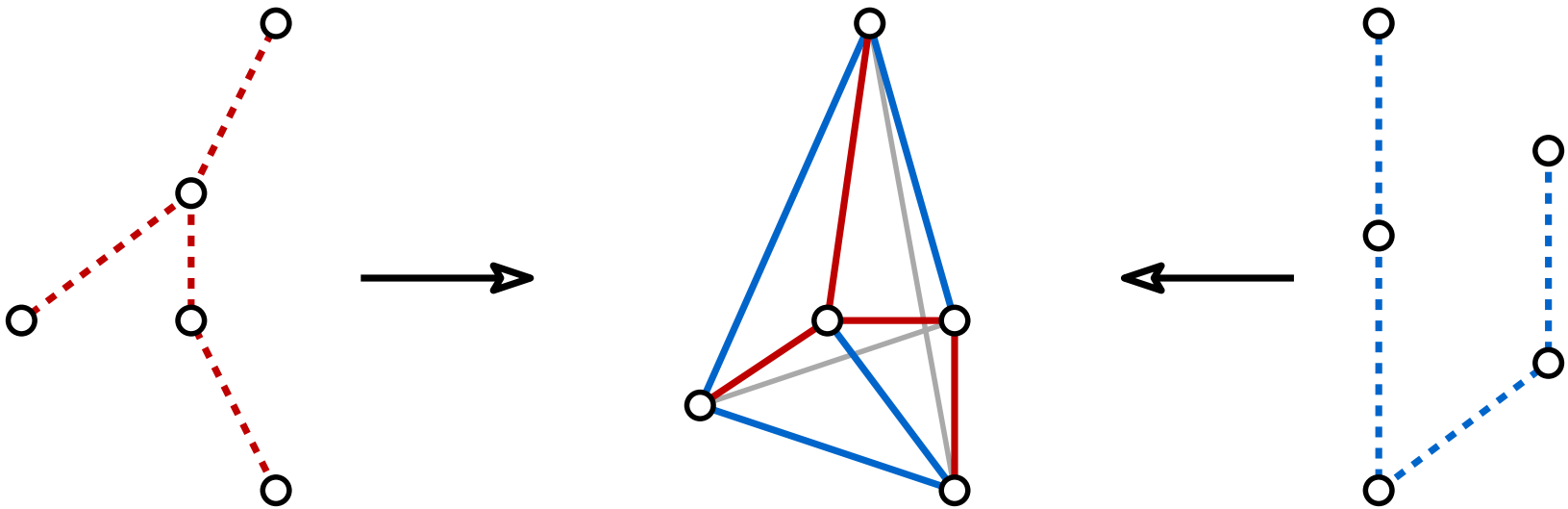
Graph packing: given k graphs on n vertices, is there a graph on n vertices that contains them as edge-disjoint subgraphs?



- Characterization of the pairs of trees that admit a packing into K_n. (Hedetniemi, Hedetniemi, and Slater 1981)

Introduction

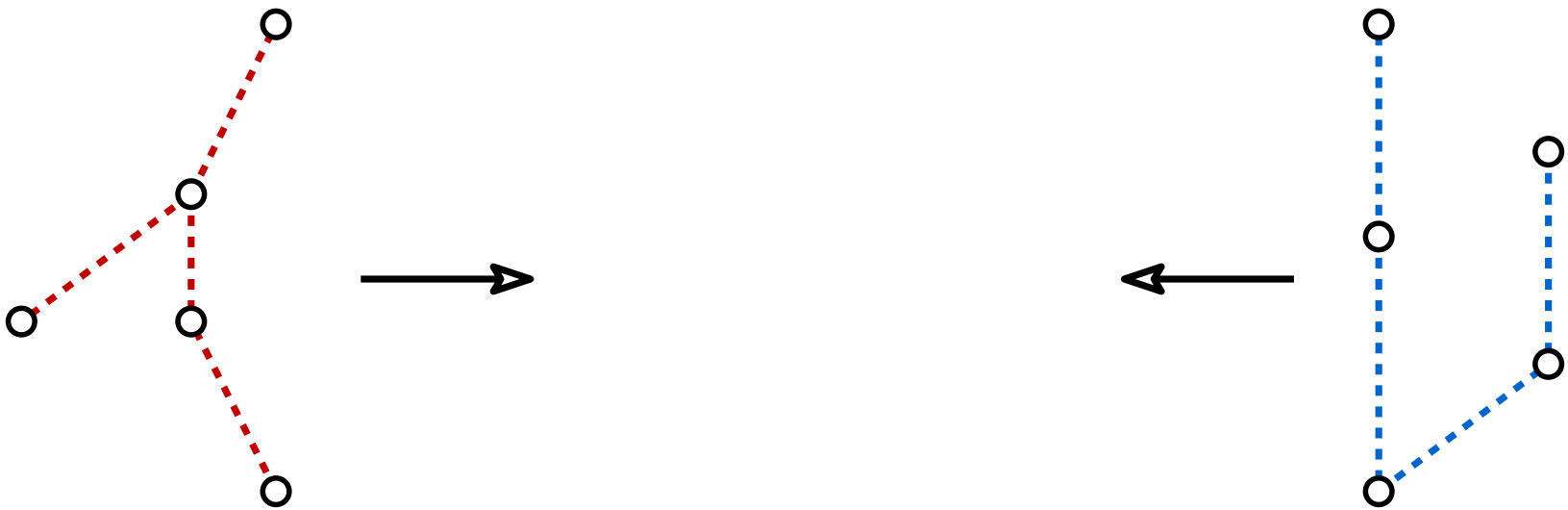
Graph packing: given k graphs on n vertices, is there a graph on n vertices that contains them as edge-disjoint subgraphs?



- Characterization of the pairs of trees that admit a packing into K_n. (Hedetniemi, Hedetniemi, and Slater 1981)
- Characterization of the triples of trees that admit a packing into K_n. (Maheo, Saclé, and Woźniak 1996)

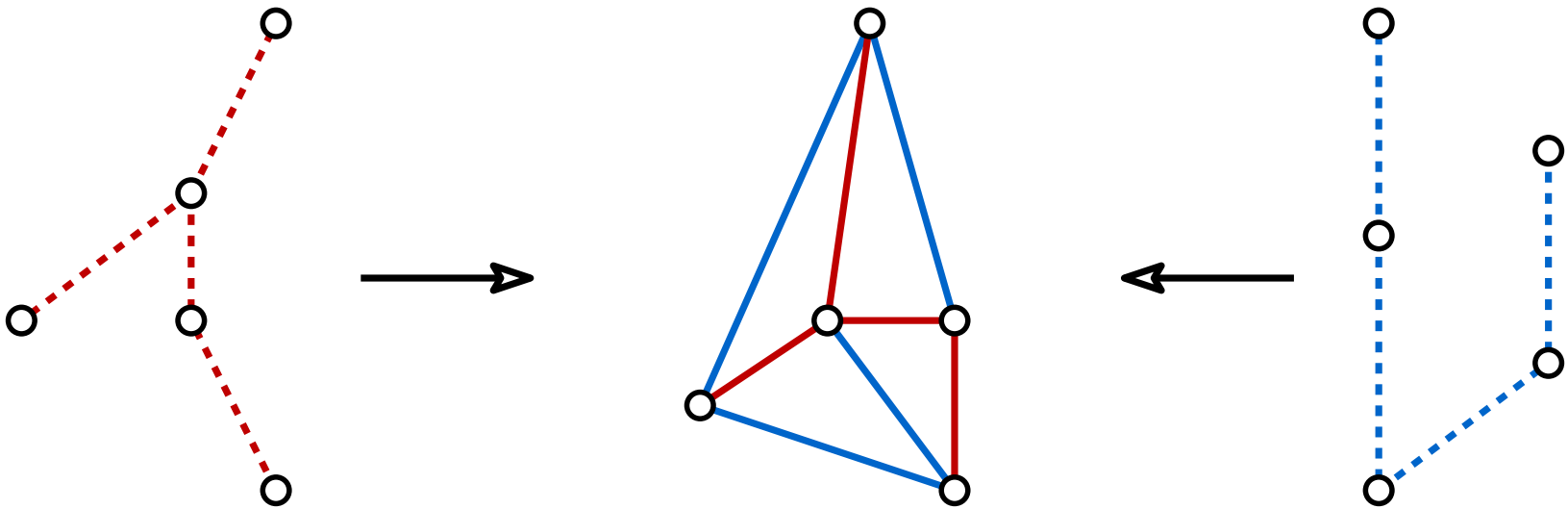
Introduction

Planar packing: given k graphs on n vertices, is there a planar graph on n vertices that contains them as edge-disjoint subgraphs?



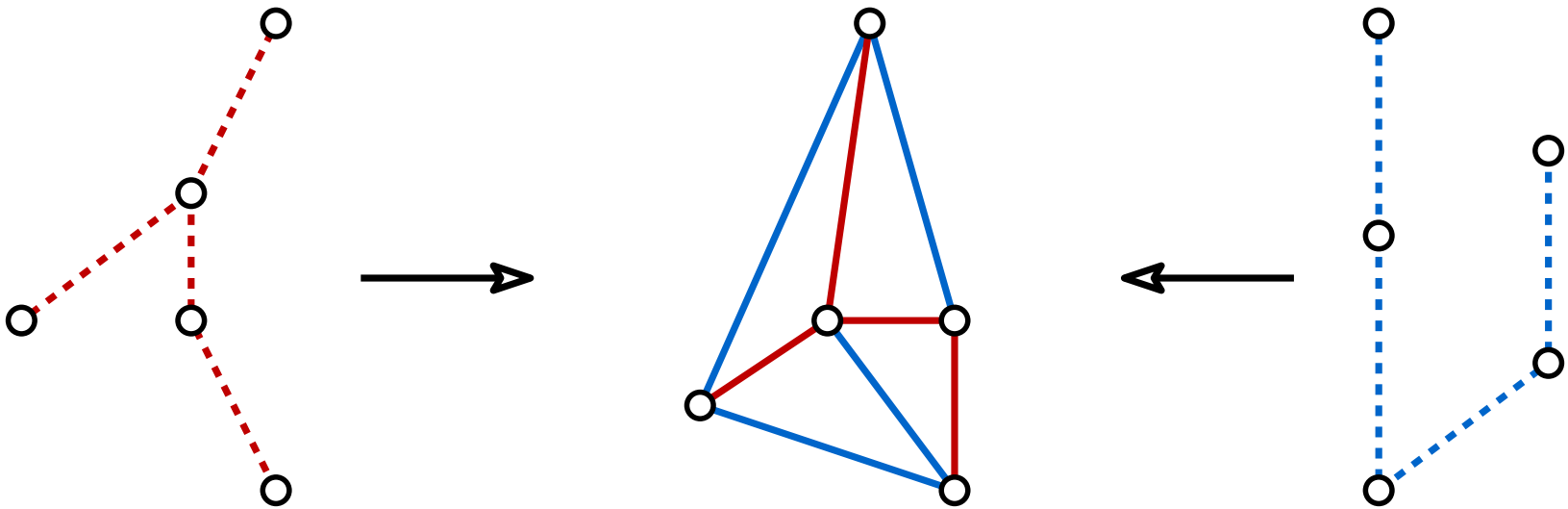
Introduction

Planar packing: given k graphs on n vertices, is there a planar graph on n vertices that contains them as edge-disjoint subgraphs?



Introduction

Planar packing: given *k* graphs on *n* vertices, is there a planar graph on *n* vertices that contains them as edge-disjoint subgraphs?

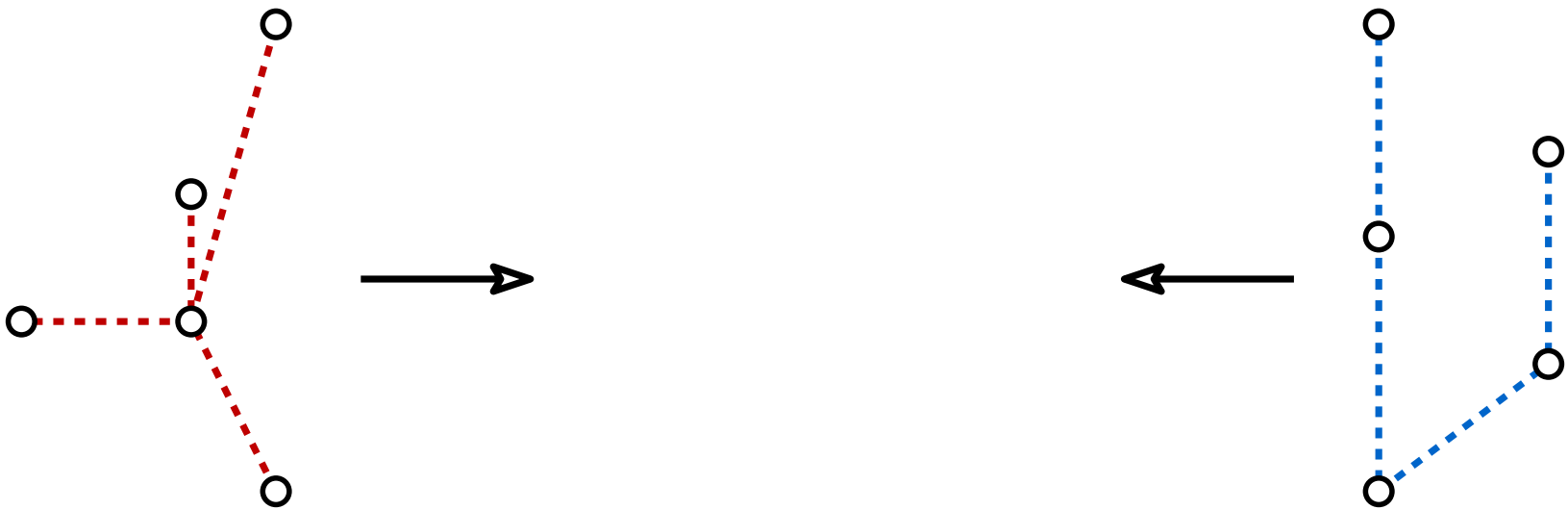


Conjecture (García, Hernando, Hurtado, Noy, and Tejel 2002)

*Every two trees on n vertices admit a planar packing.**

Introduction

Planar packing: given *k* graphs on *n* vertices, is there a planar graph on *n* vertices that contains them as edge-disjoint subgraphs?

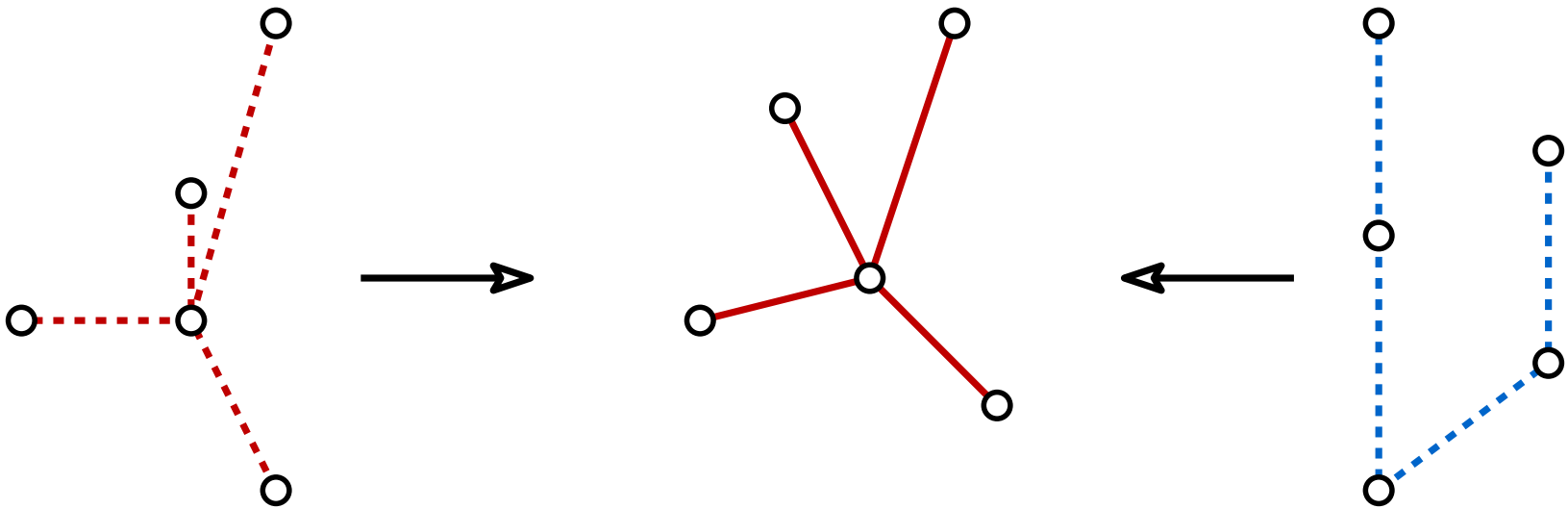


Conjecture (García, Hernando, Hurtado, Noy, and Tejel 2002)

*Every two trees on n vertices admit a planar packing.**

Introduction

Planar packing: given *k* graphs on *n* vertices, is there a planar graph on *n* vertices that contains them as edge-disjoint subgraphs?

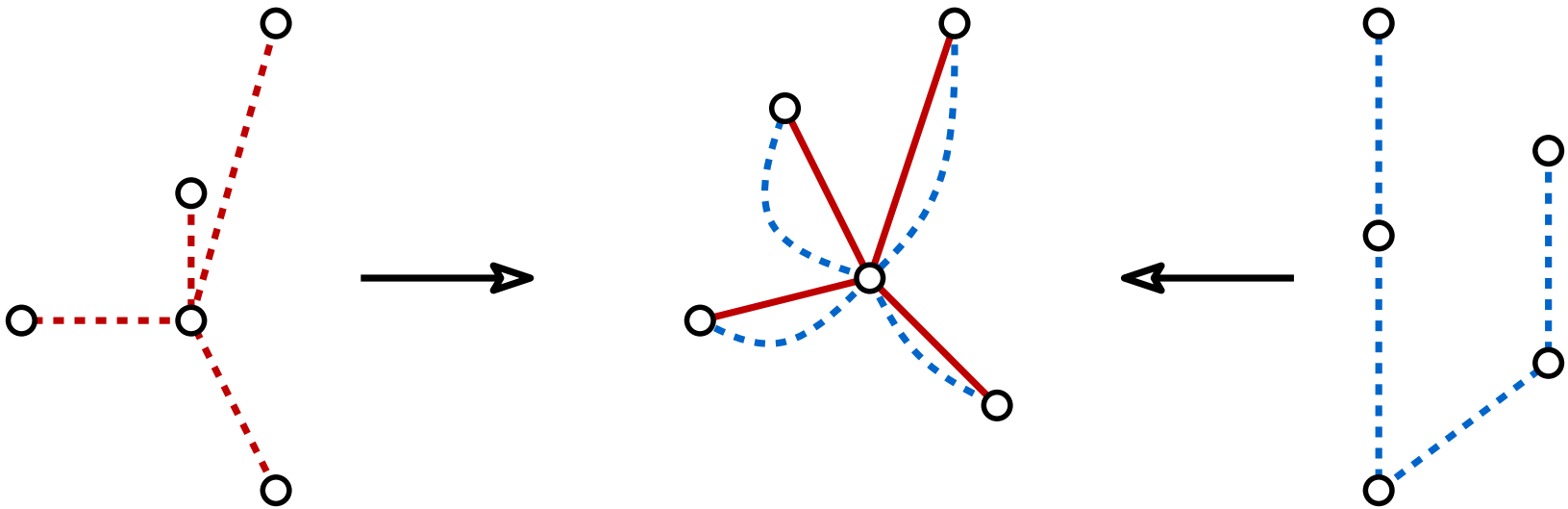


Conjecture (García, Hernando, Hurtado, Noy, and Tejel 2002)

*Every two trees on n vertices admit a planar packing.**

Introduction

Planar packing: given *k* graphs on *n* vertices, is there a planar graph on *n* vertices that contains them as edge-disjoint subgraphs?

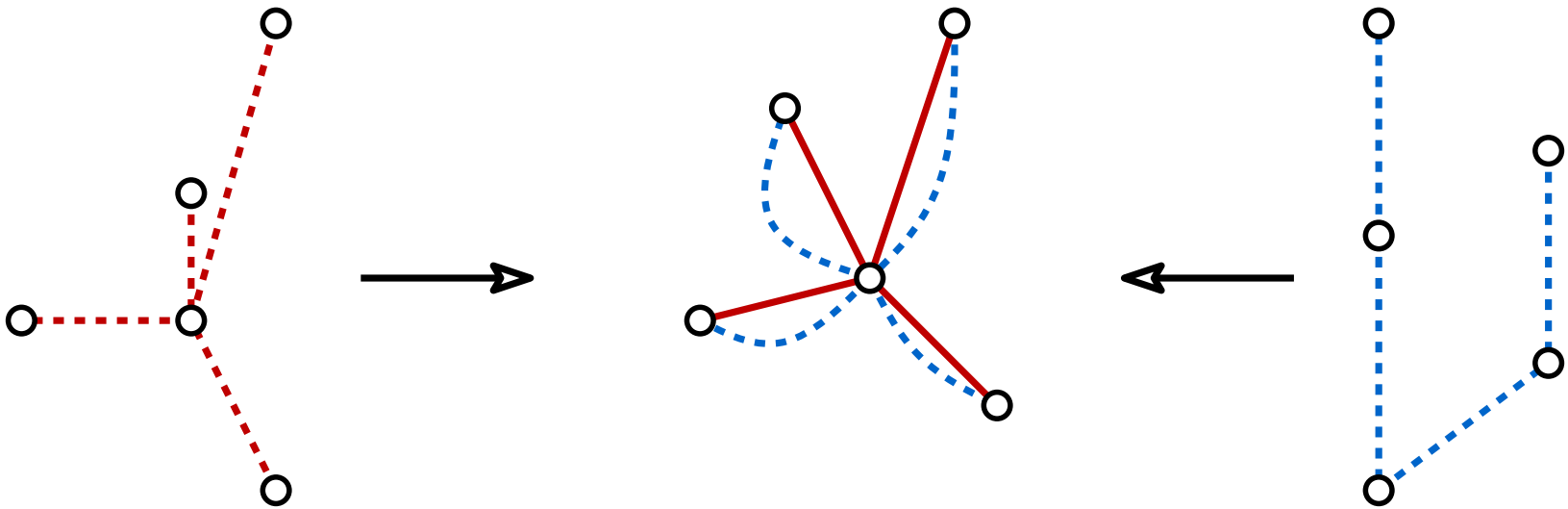


Conjecture (García, Hernando, Hurtado, Noy, and Tejel 2002)

*Every two trees on n vertices admit a planar packing.**

Introduction

Planar packing: given *k* graphs on *n* vertices, is there a planar graph on *n* vertices that contains them as edge-disjoint subgraphs?



Conjecture (García, Hernando, Hurtado, Noy, and Tejel 2002)

Every two non-star trees on n vertices admit a planar packing.

Planar tree packing

Conjecture (García, Hernando, Hurtado, Noy, and Tejel 2002)

Every two non-star trees on n vertices admit a planar packing.

Solved special cases:

- Isomorphic trees (García, Hernando, Hurtado, Noy, and Tejel 2002).
- One tree is a path (García, Hernando, Hurtado, Noy, and Tejel 2002).
- One tree is a caterpillar or a spider of diameter at most four (Oda and Ota 2006).
- One tree is a spider (Frati, Geyer, and Kaufmann 2009).
- Both trees have diameter at most four (Frati 2009).

Theorem

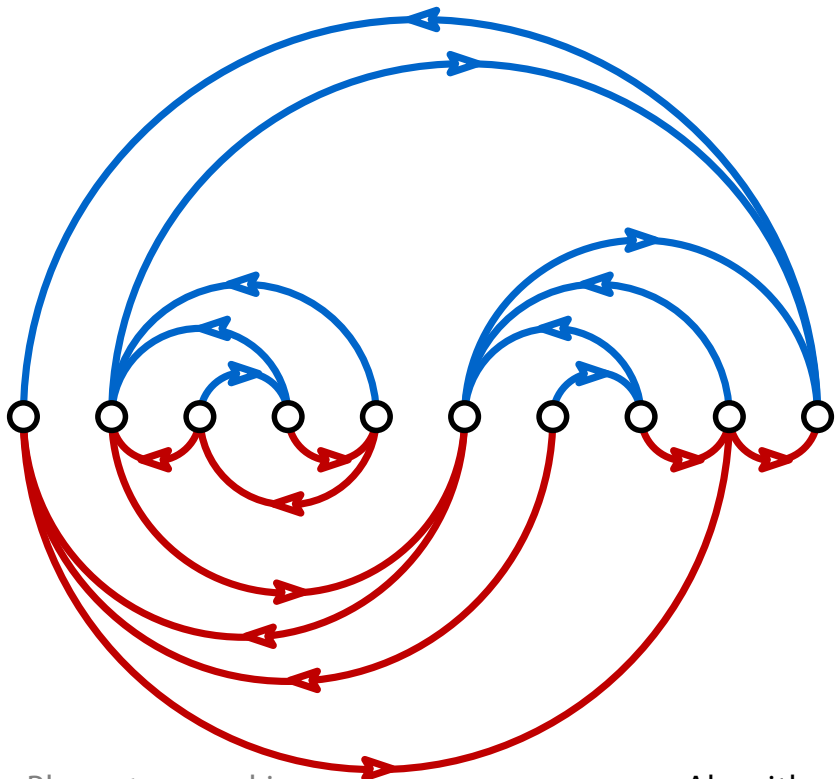
Every two non-star binary trees on n vertices admit a planar packing.

General scheme

1. Place n points on the x-axis.
2. Root the blue tree at one of its leaves.
   Recursively embed the blue tree in the upper halfplane.
3. Root the red tree at one of its leaves.
   Recursively embed the red tree in the lower halfplane.

General scheme

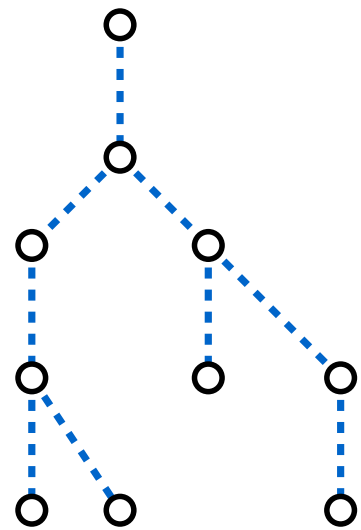
1. Place n points on the x-axis.
2. Root the blue tree at one of its leaves.
   Recursively embed the blue tree in the upper halfplane.
3. Root the red tree at one of its leaves.
   Recursively embed the red tree in the lower halfplane.



Embedding the blue tree

Embed the blue tree according to two rules:

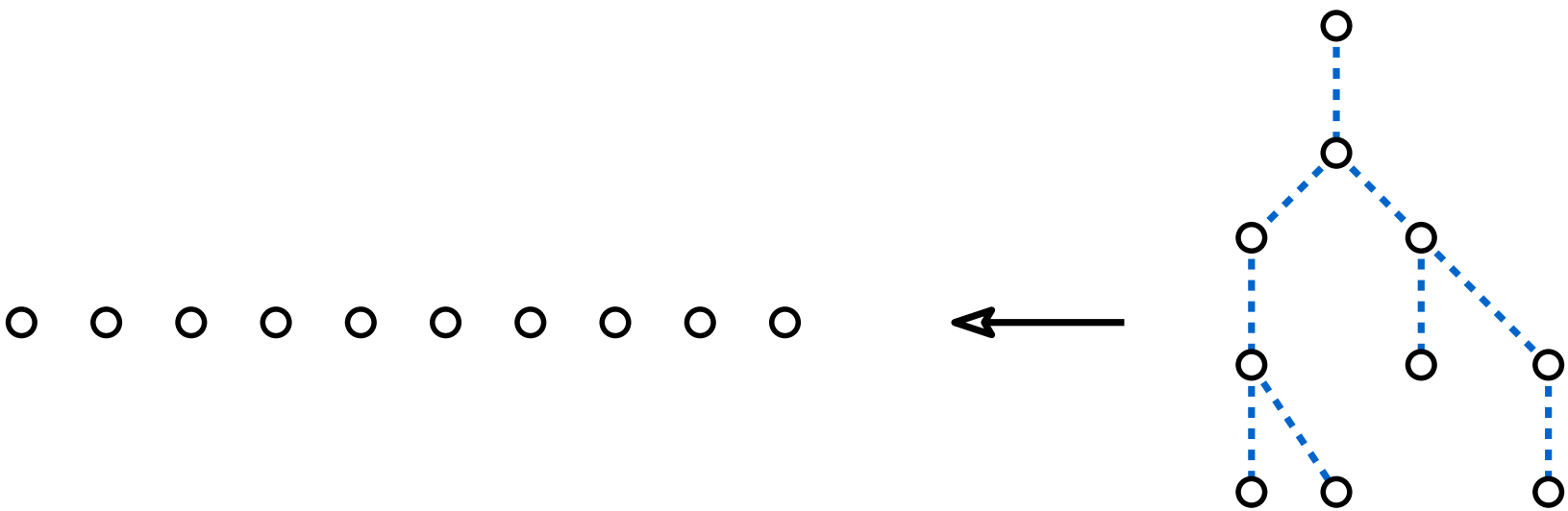
- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.



Embedding the blue tree

Embed the blue tree according to two rules:

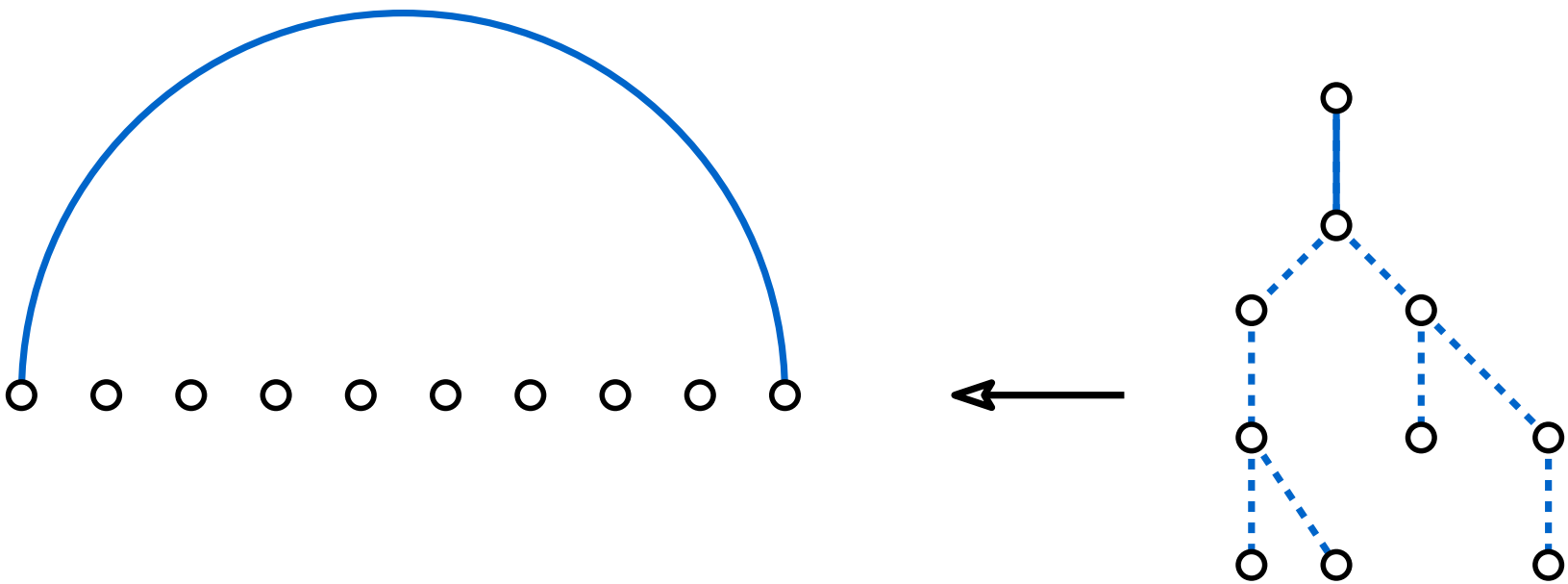
- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.



Embedding the blue tree

Embed the blue tree according to two rules:

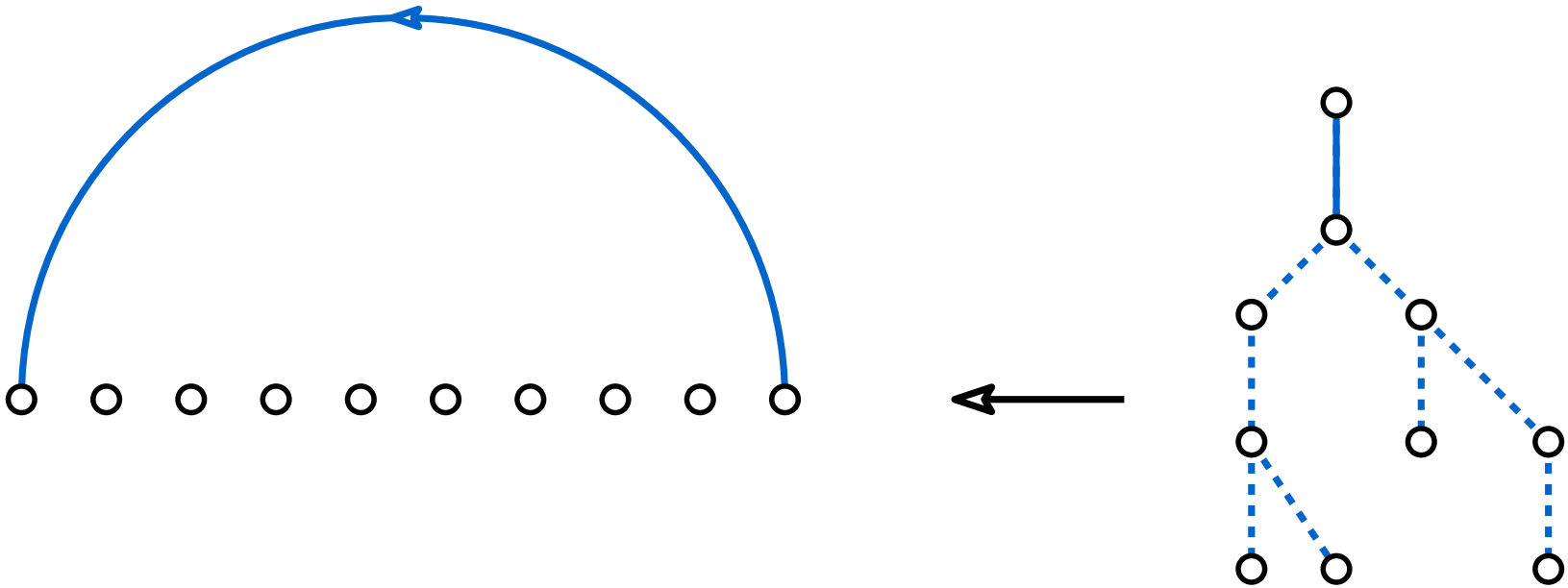
- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.



Embedding the blue tree

Embed the blue tree according to two rules:

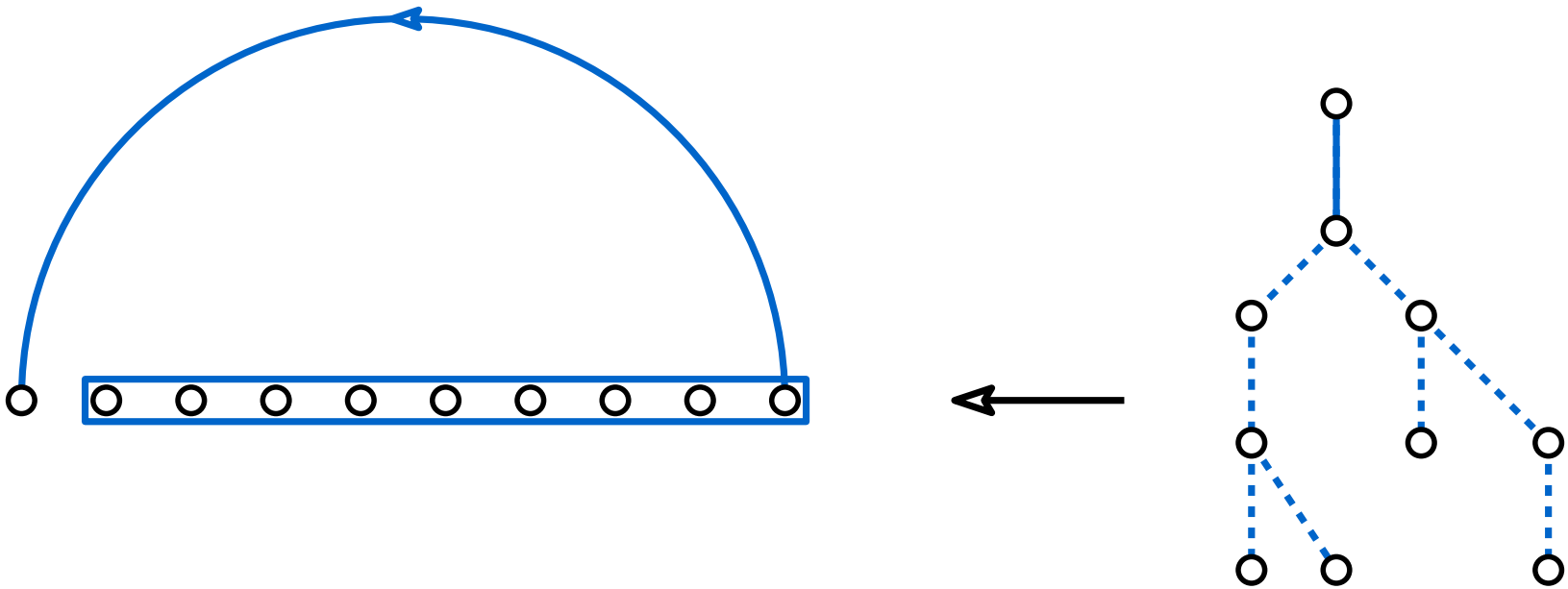
- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.



Embedding the blue tree

Embed the blue tree according to two rules:

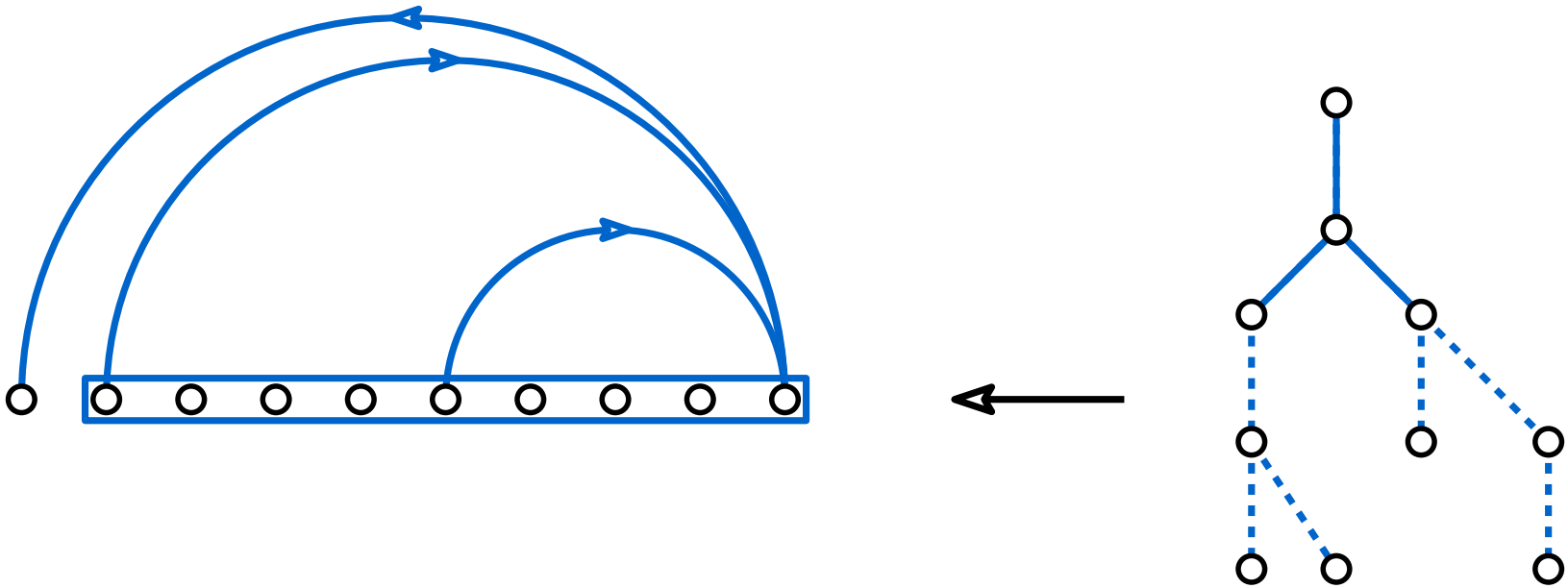
- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.



Embedding the blue tree

Embed the blue tree according to two rules:

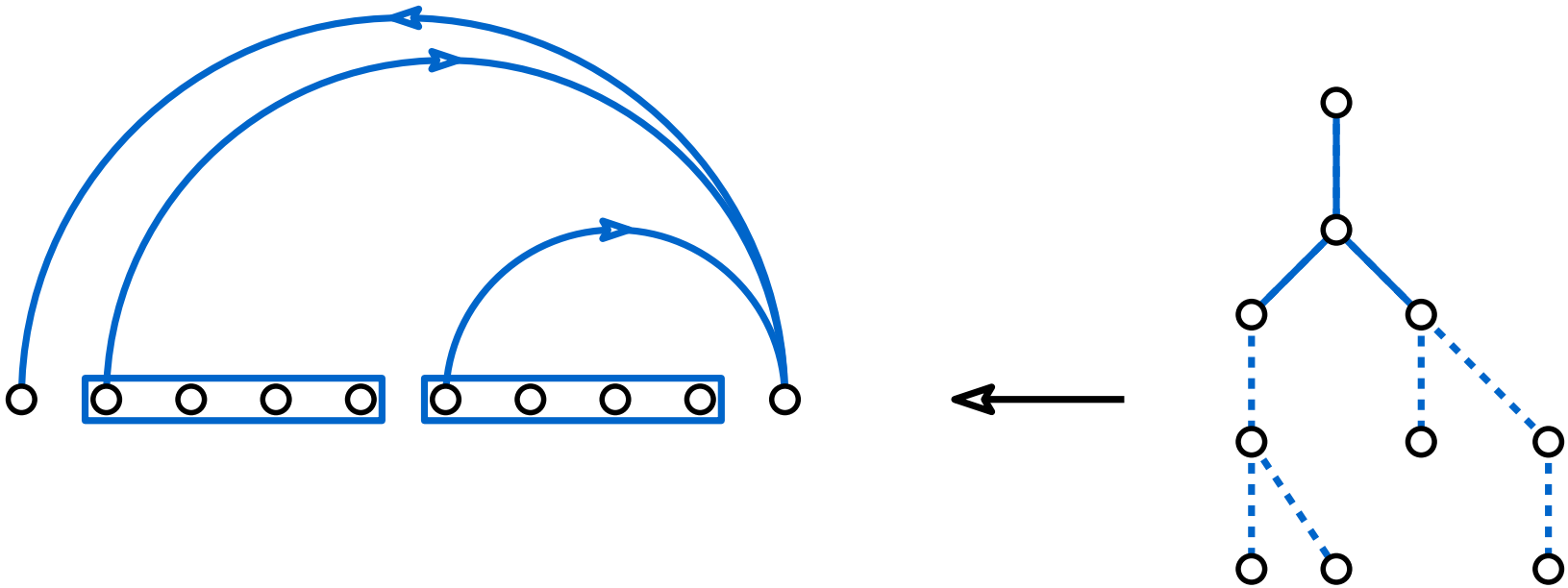
- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.



Embedding the blue tree

Embed the blue tree according to two rules:

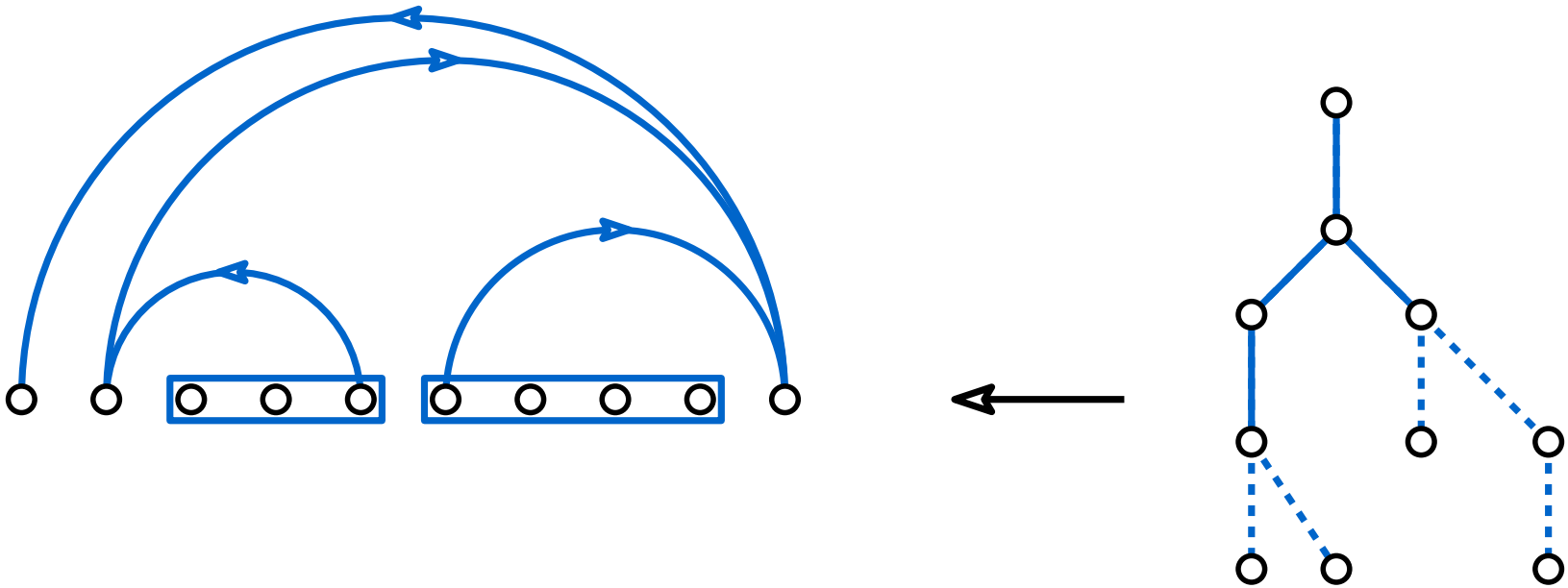
- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.



Embedding the blue tree

Embed the blue tree according to two rules:

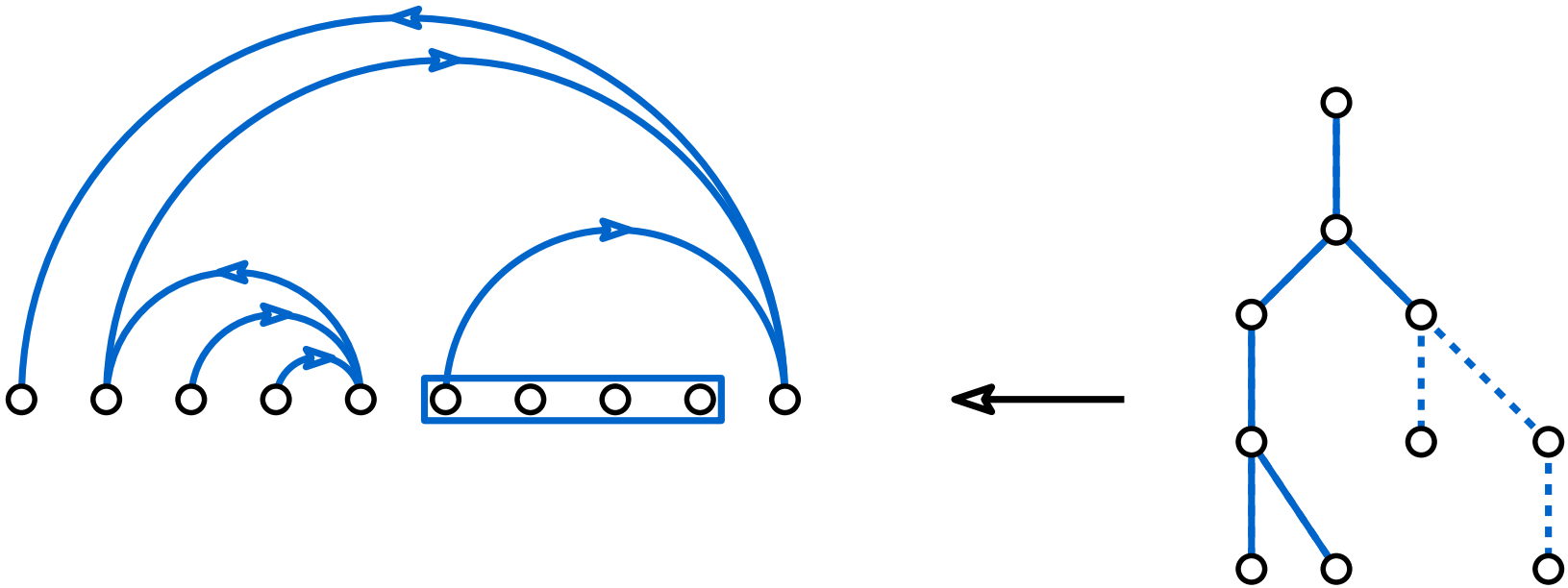
- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.



Embedding the blue tree

Embed the blue tree according to two rules:

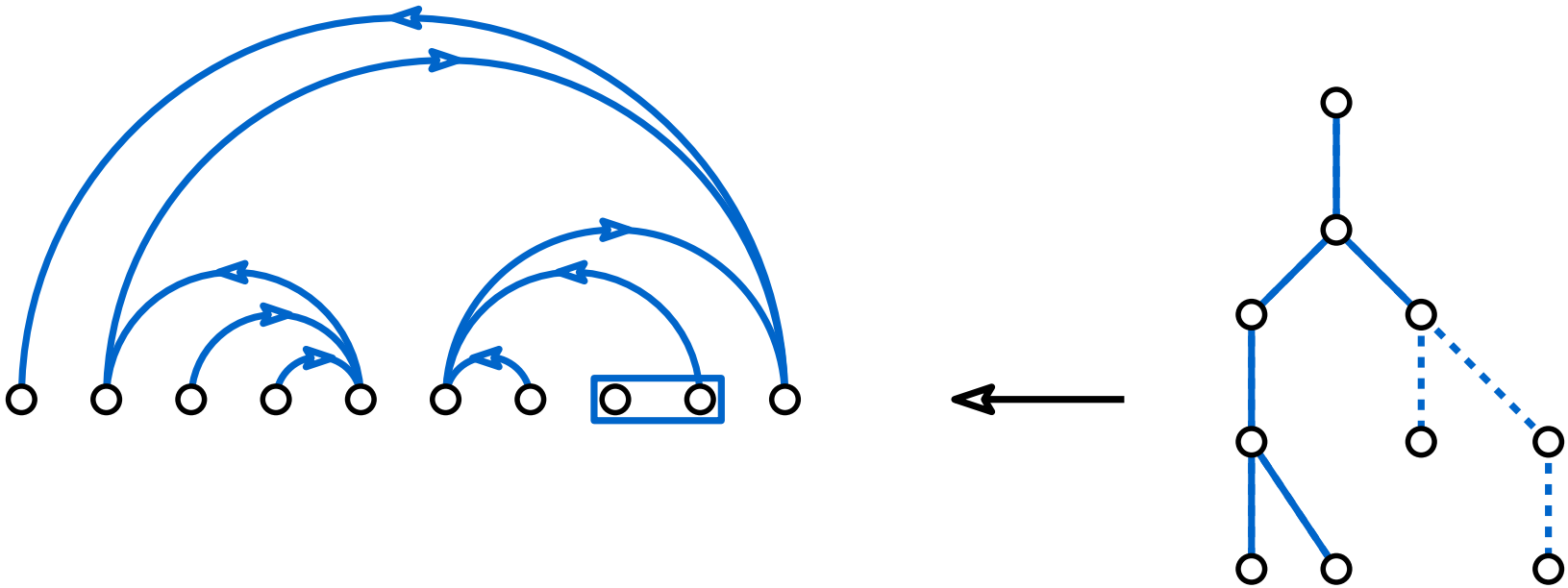
- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.



Embedding the blue tree

Embed the blue tree according to two rules:

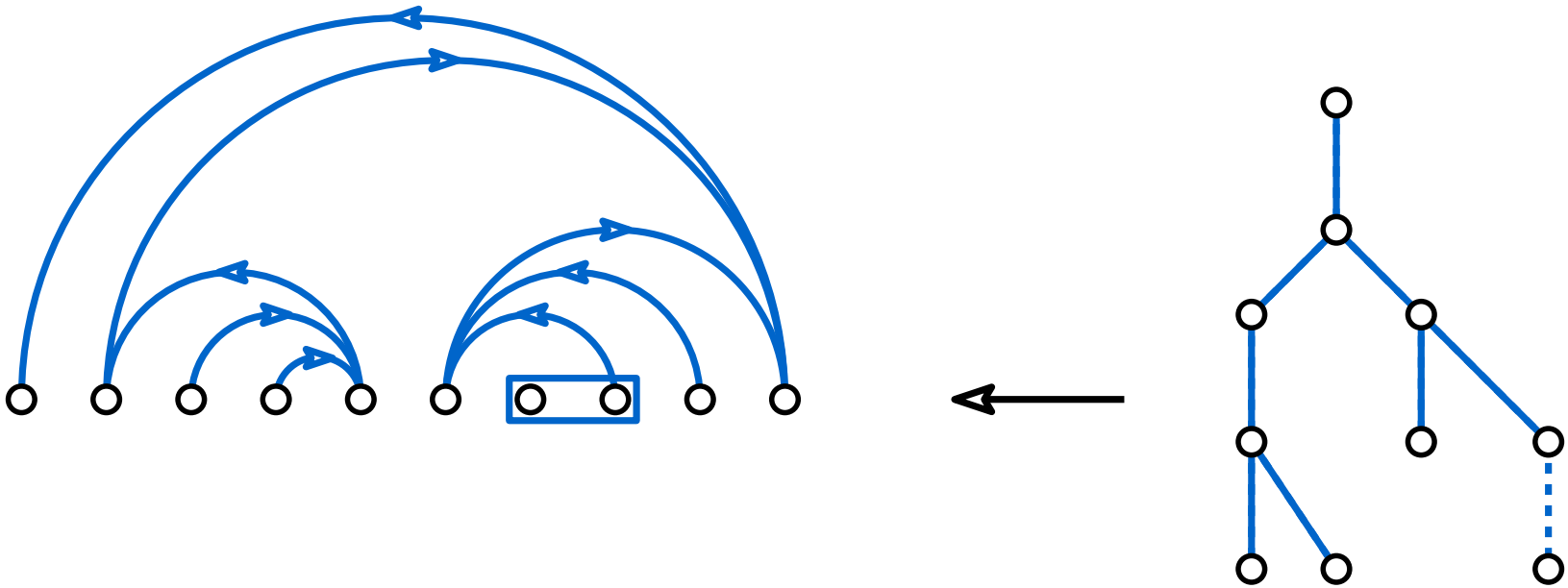
- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.



Embedding the blue tree

Embed the blue tree according to two rules:

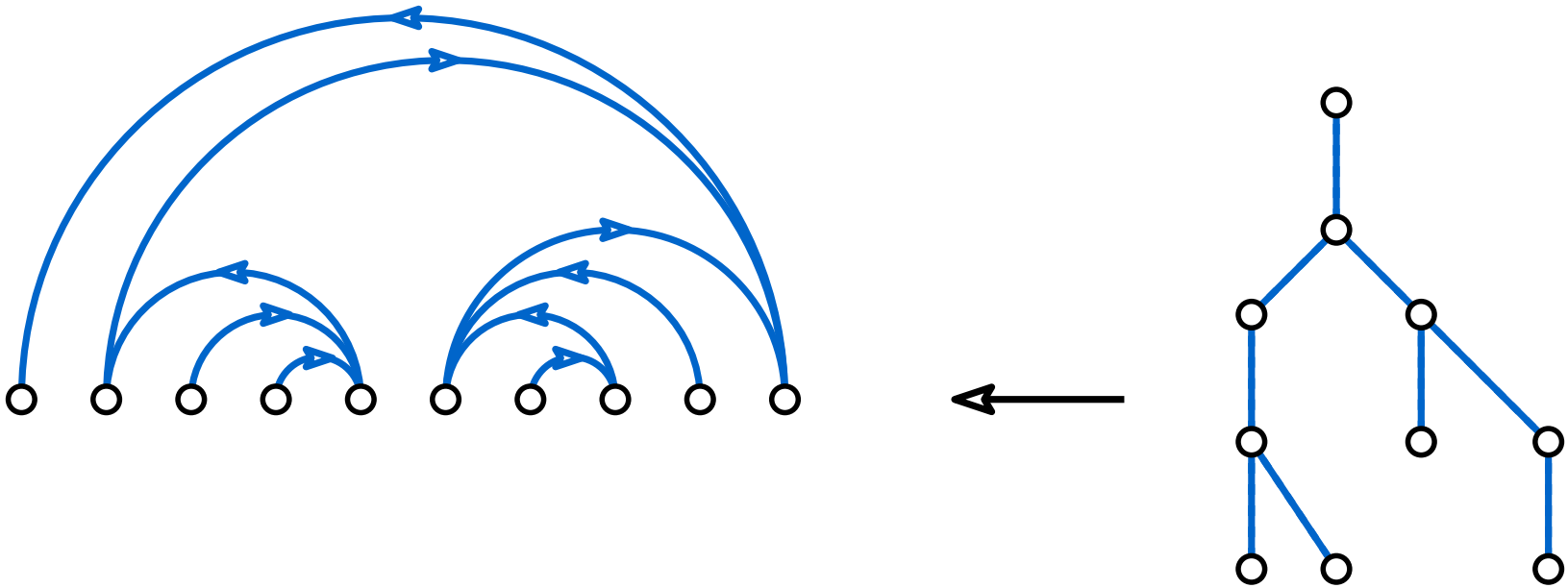
- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.



Embedding the blue tree

Embed the blue tree according to two rules:

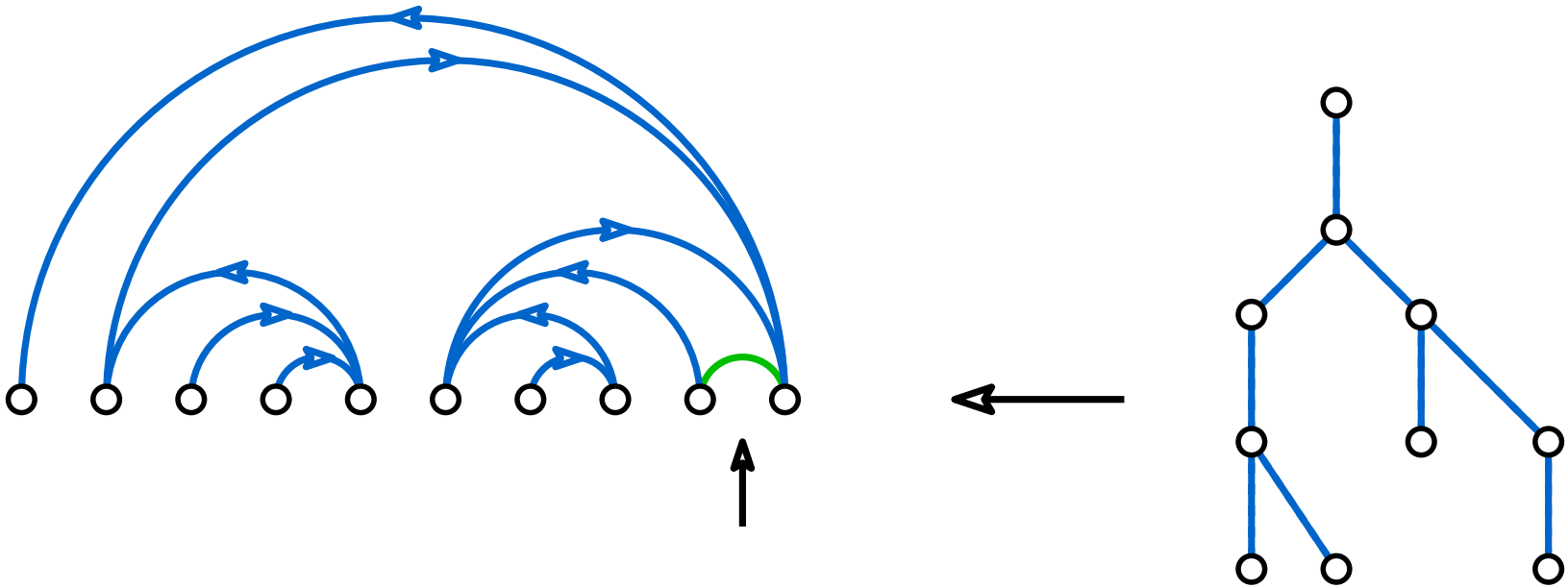
- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.



Embedding the blue tree

Embed the blue tree according to two rules:

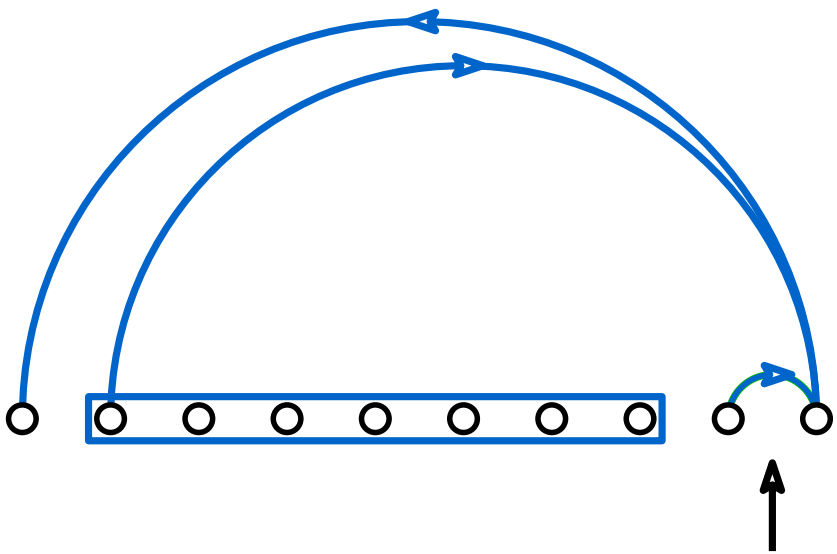
- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.



Embedding the blue tree

Embed the blue tree according to two rules:

- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.

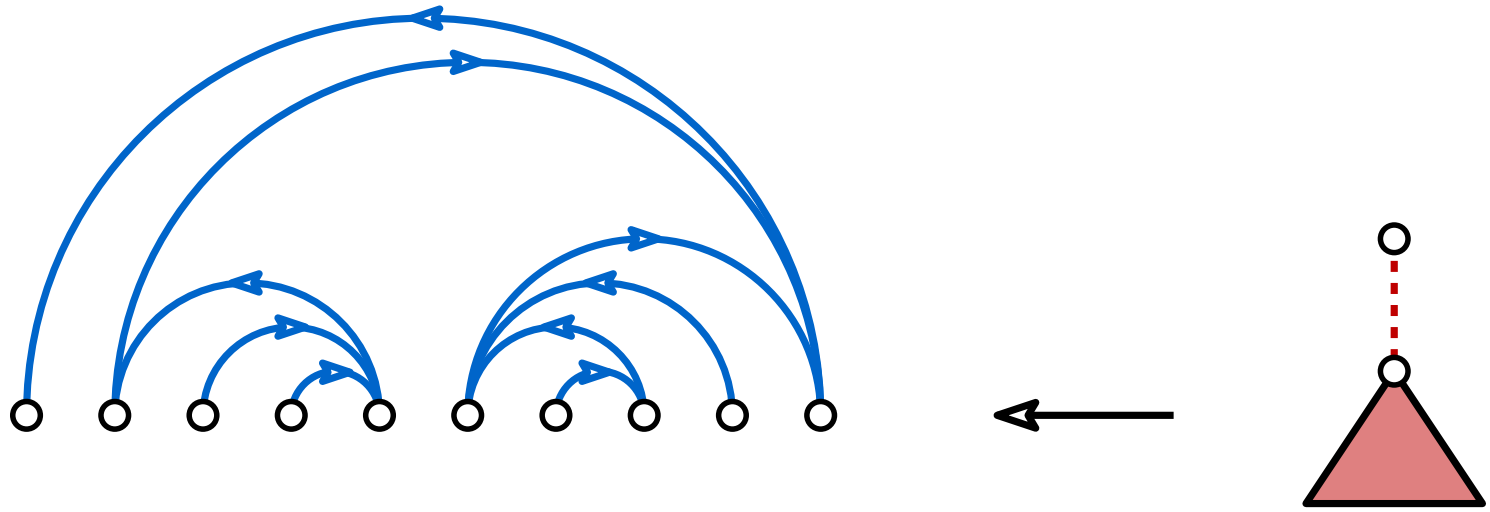


General scheme

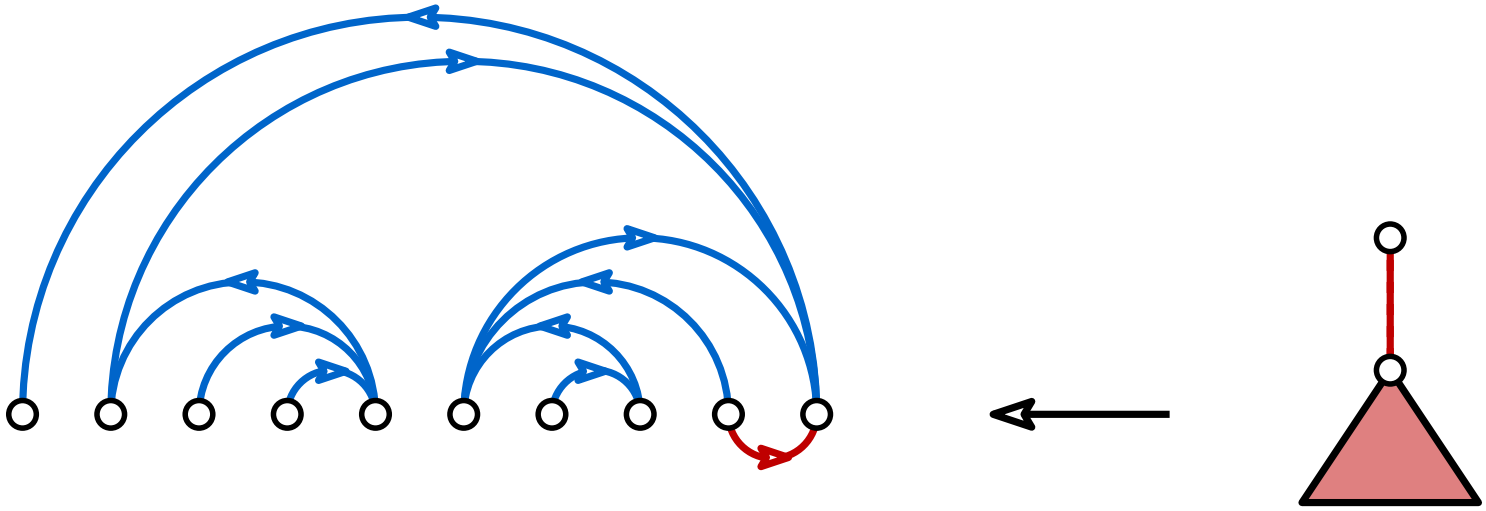
Recall:

1. Place n points on the x-axis.
2. Root the blue tree at one of its leaves.
   Recursively embed the blue tree in the upper halfplane.
3. Root the red tree at one of its leaves.
   Recursively embed the red tree in the lower halfplane.

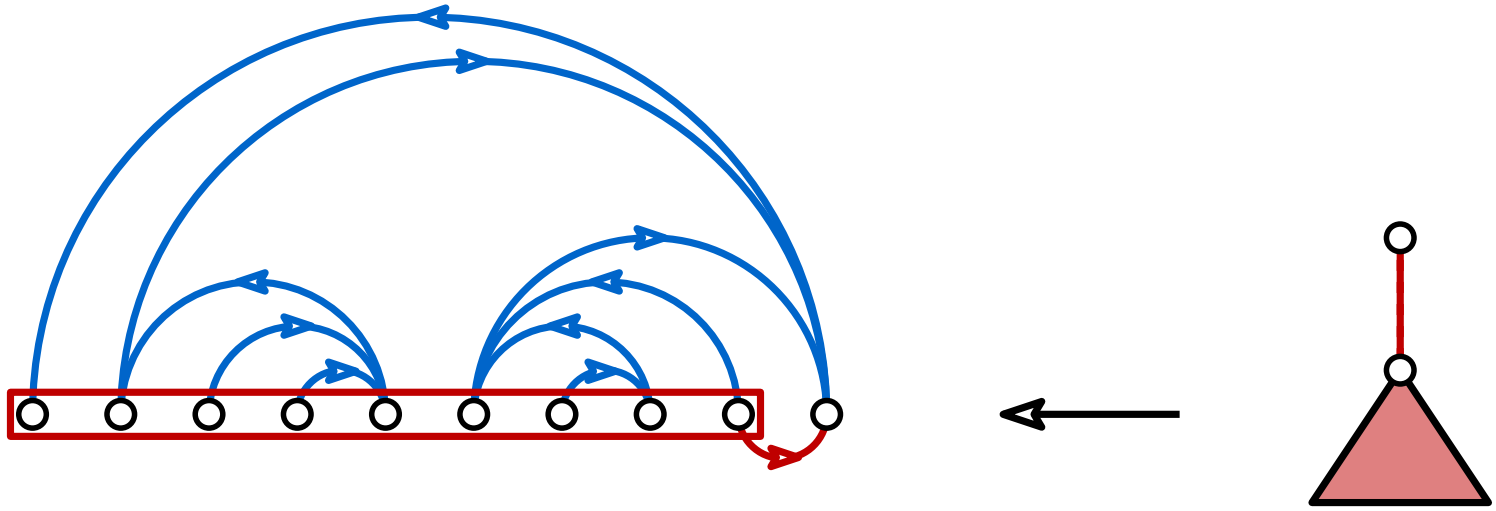
Embedding the first red edge



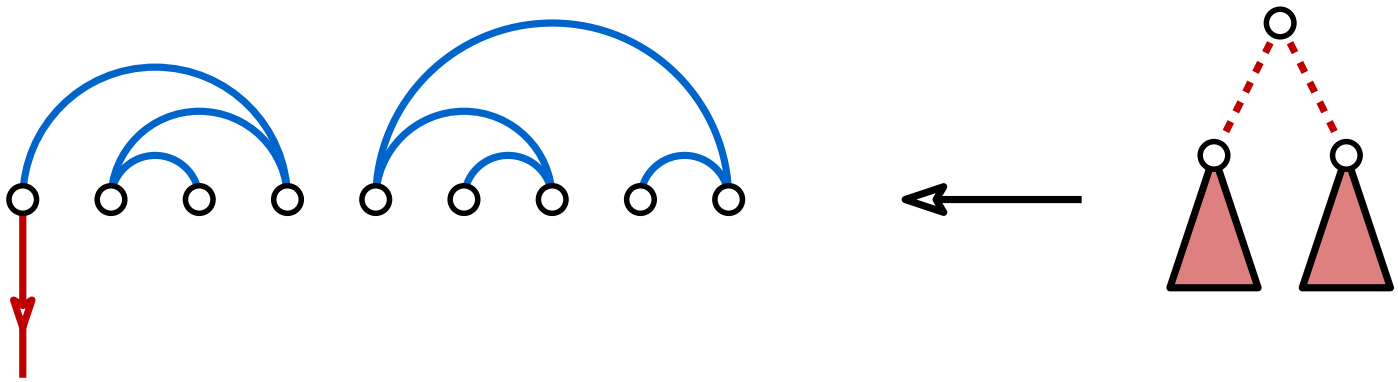
Embedding the first red edge



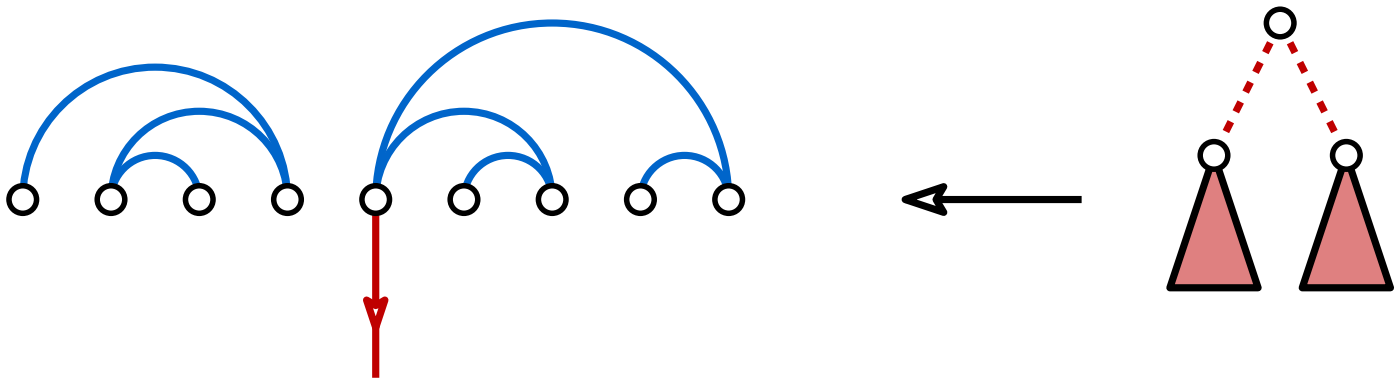
Embedding the first red edge



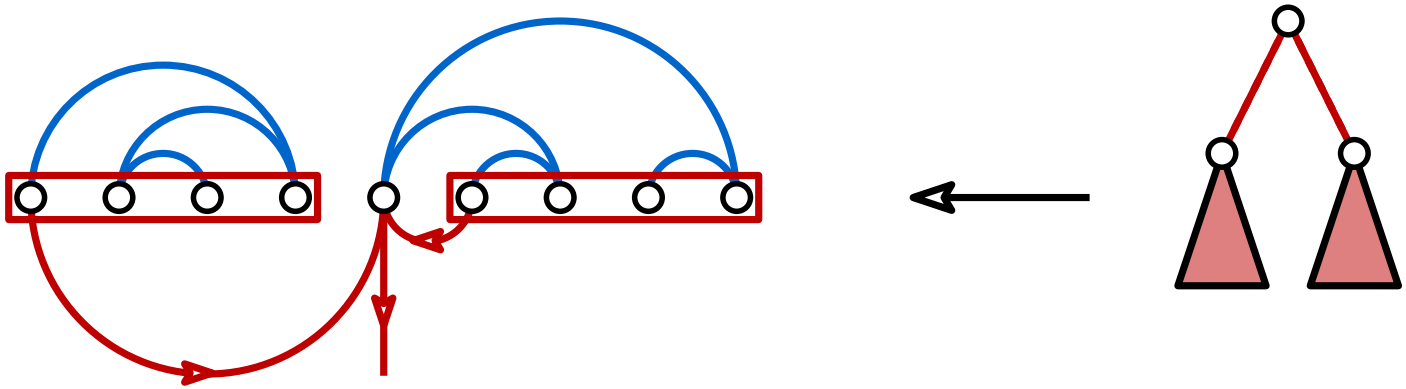
Embedding the rest



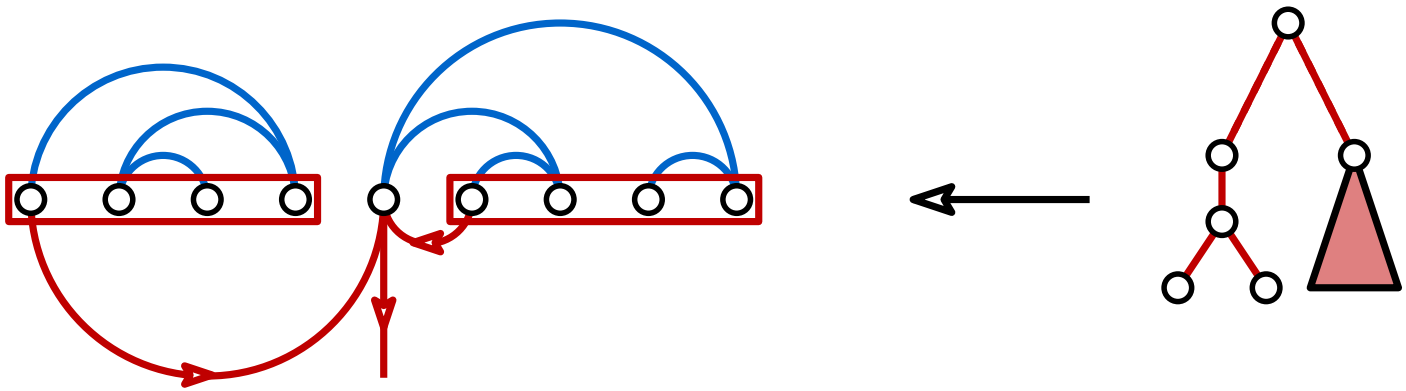
Embedding the rest



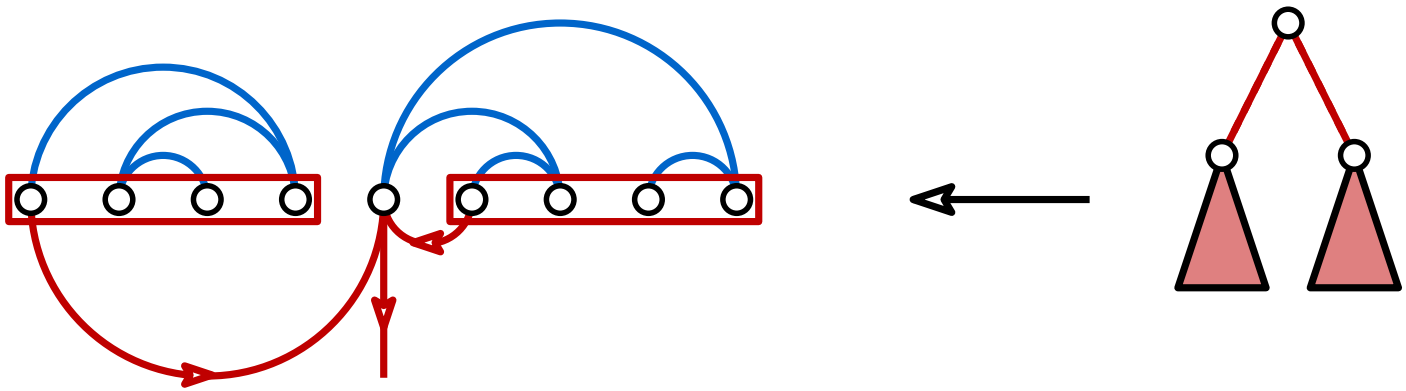
Embedding the rest



Embedding the rest

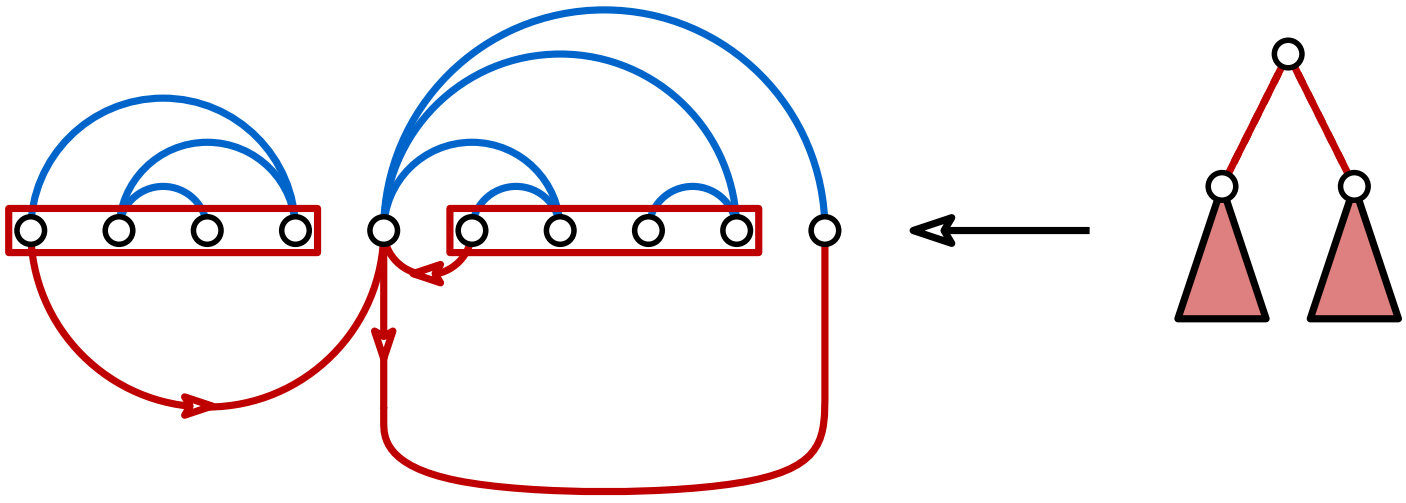


Embedding the rest



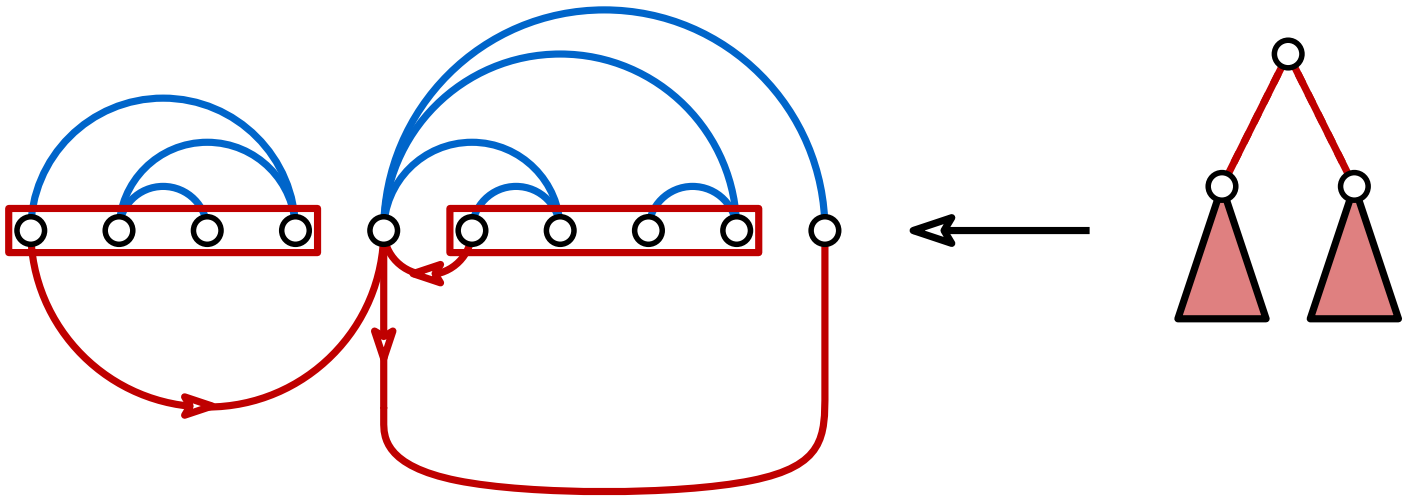
- Be careful with small red or blue stars.

Embedding the rest



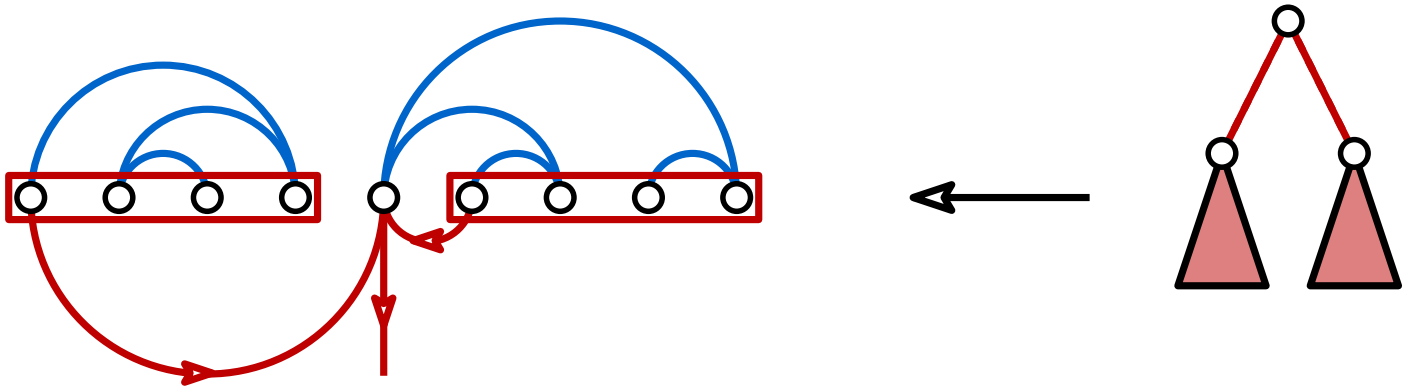
- Be careful with small red or blue stars.

Embedding the rest



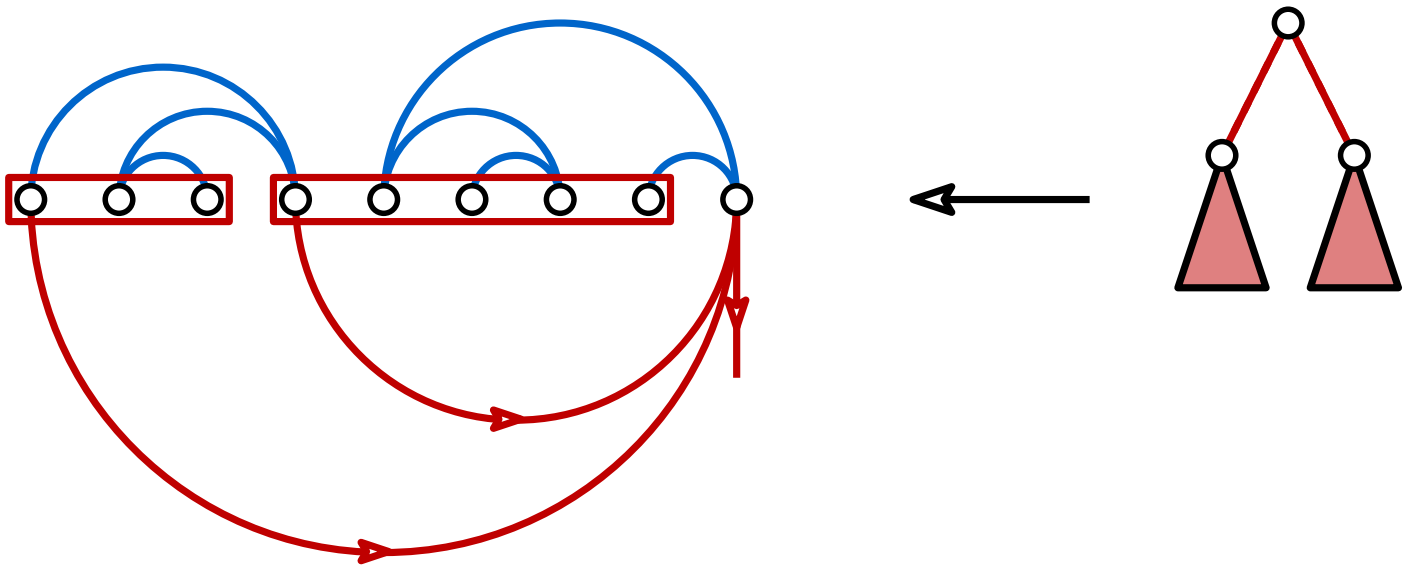
- Be careful with small red or blue stars.
- A vertex is in conflict if it is connected to the parent of the current root: do not embed the current root there.

Embedding the rest



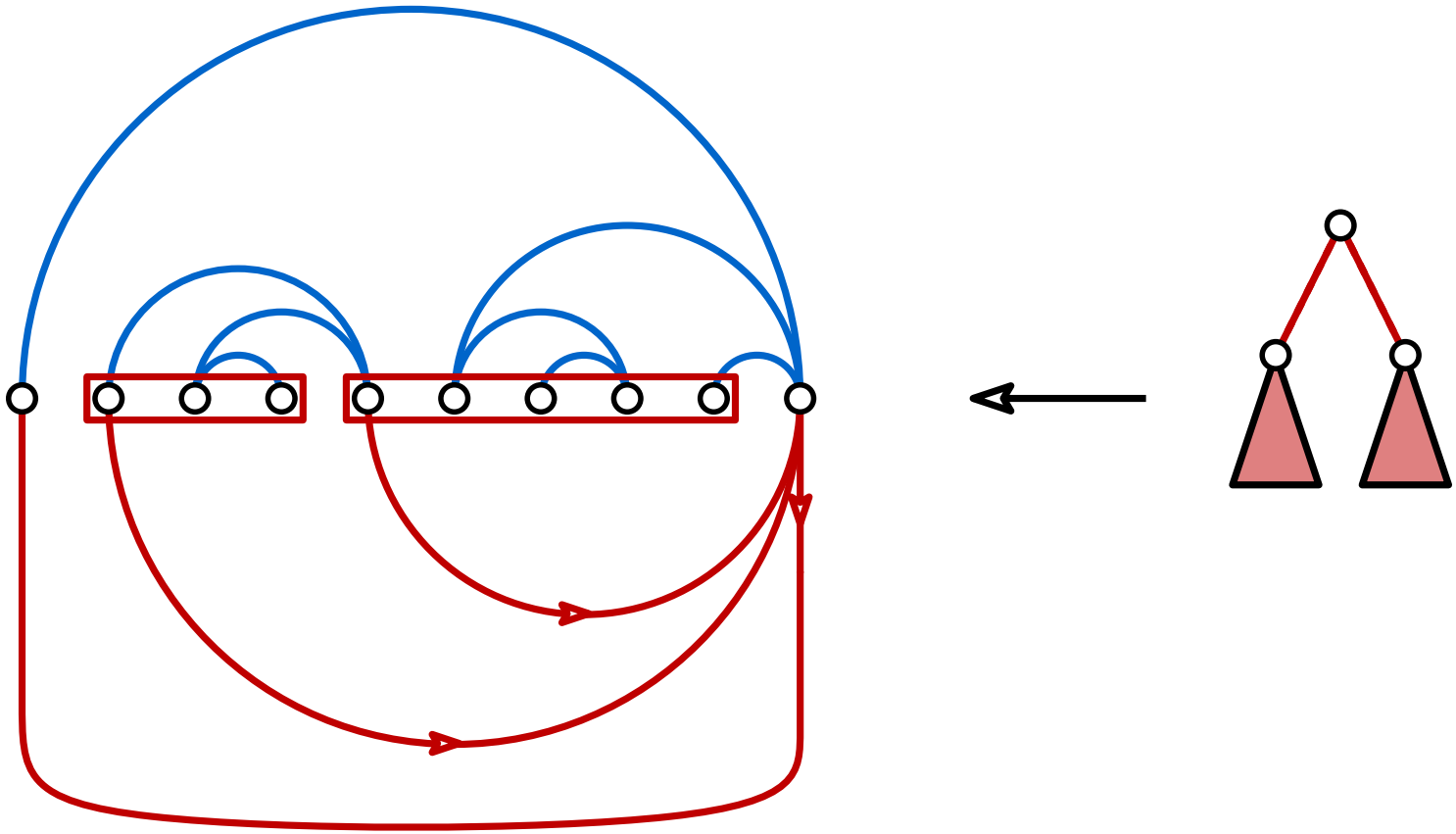
- Be careful with small red or blue stars.
- A vertex is in conflict if it is connected to the parent of the current root: do not embed the current root there.

Embedding the rest



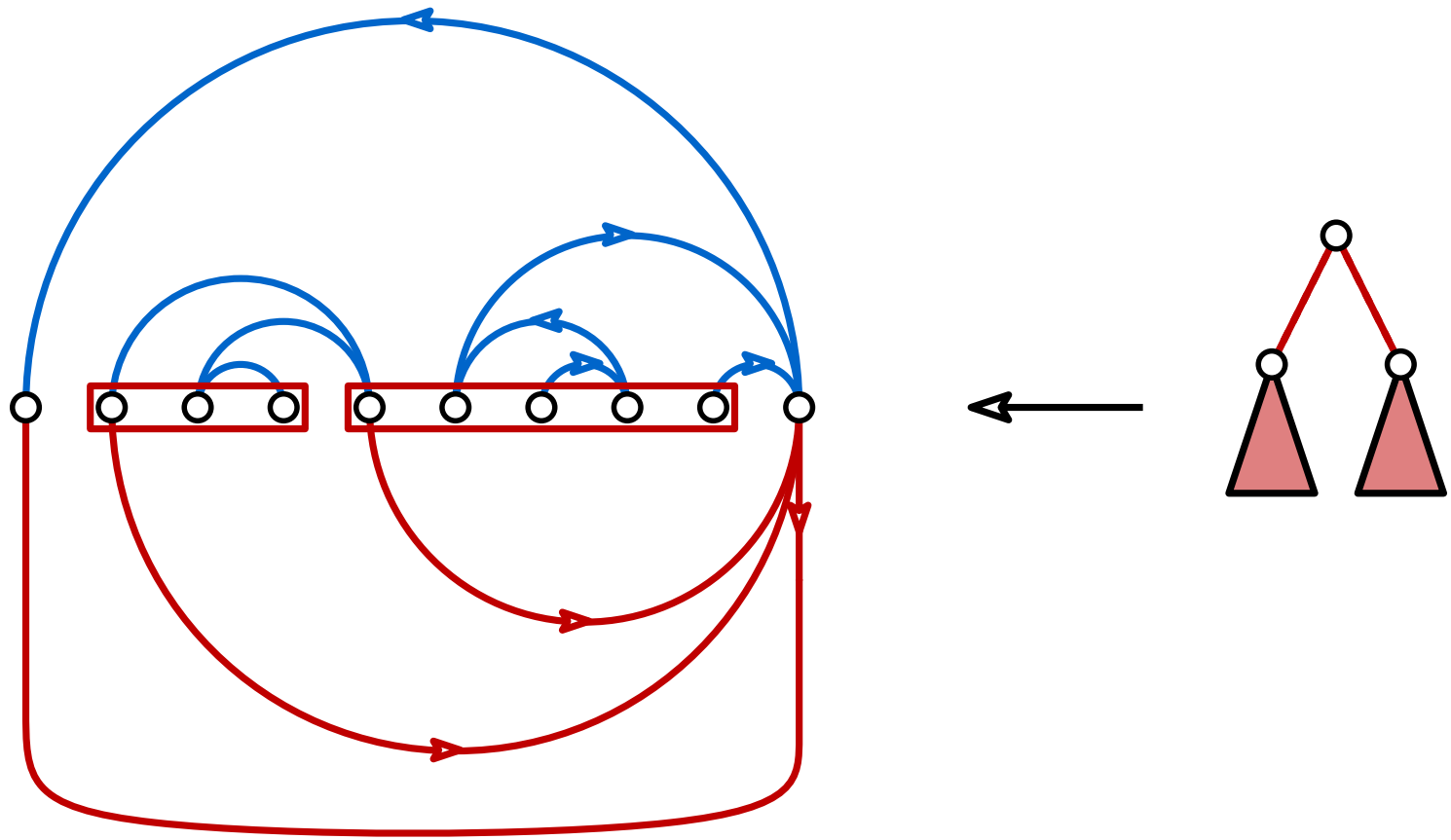
- Be careful with small red or blue stars.
- A vertex is in conflict if it is connected to the parent of the current root: do not embed the current root there.

Embedding the rest



- Be careful with small red or blue stars.
- A vertex is in conflict if it is connected to the parent of the current root: do not embed the current root there.

Embedding the rest



- Be careful with small red or blue stars.
- A vertex is in conflict if it is connected to the parent of the current root: do not embed the current root there.

Embedding the rest

Recall:

- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.

Embedding the rest

Recall:

- **One-side rule**: blue edges incident to each vertex either all go left or all go right.
- **Larger-subtree-first rule**: the larger (if any) subtree is embedded next to its parent.

When we embed a subtree on an interval $[i, j]$:

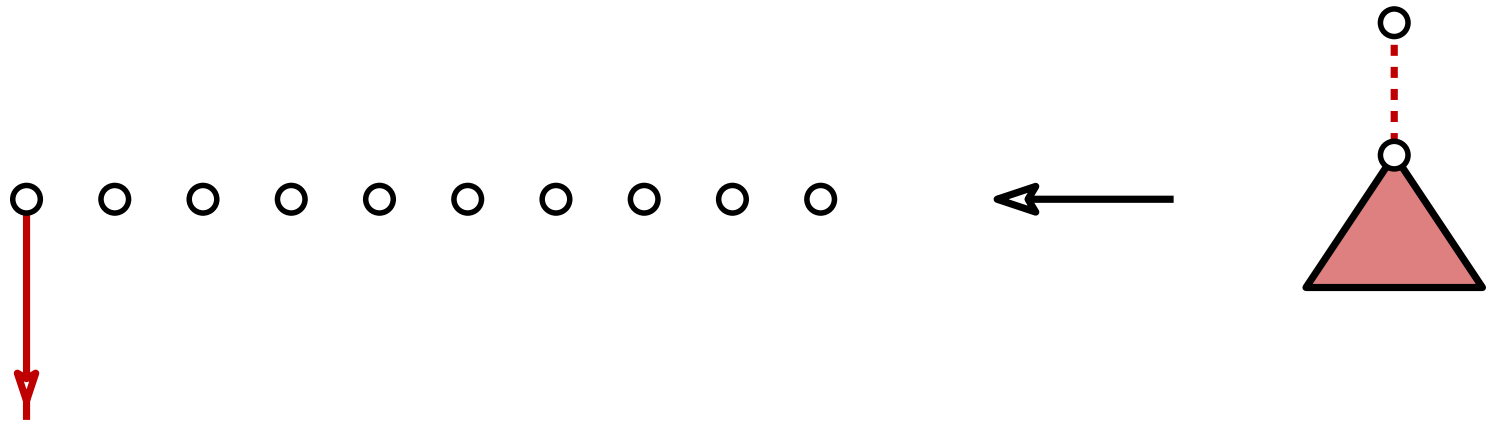
- **Placement invariant**: position i is not in conflict.
- **Local conformance**: the forest on $[i, j]$ satisfies the two rules.
- **No star**: the blue embedding on $[i, j]$ is not a star.

Overview of the cases

We distinguish the following cases in the recursion:

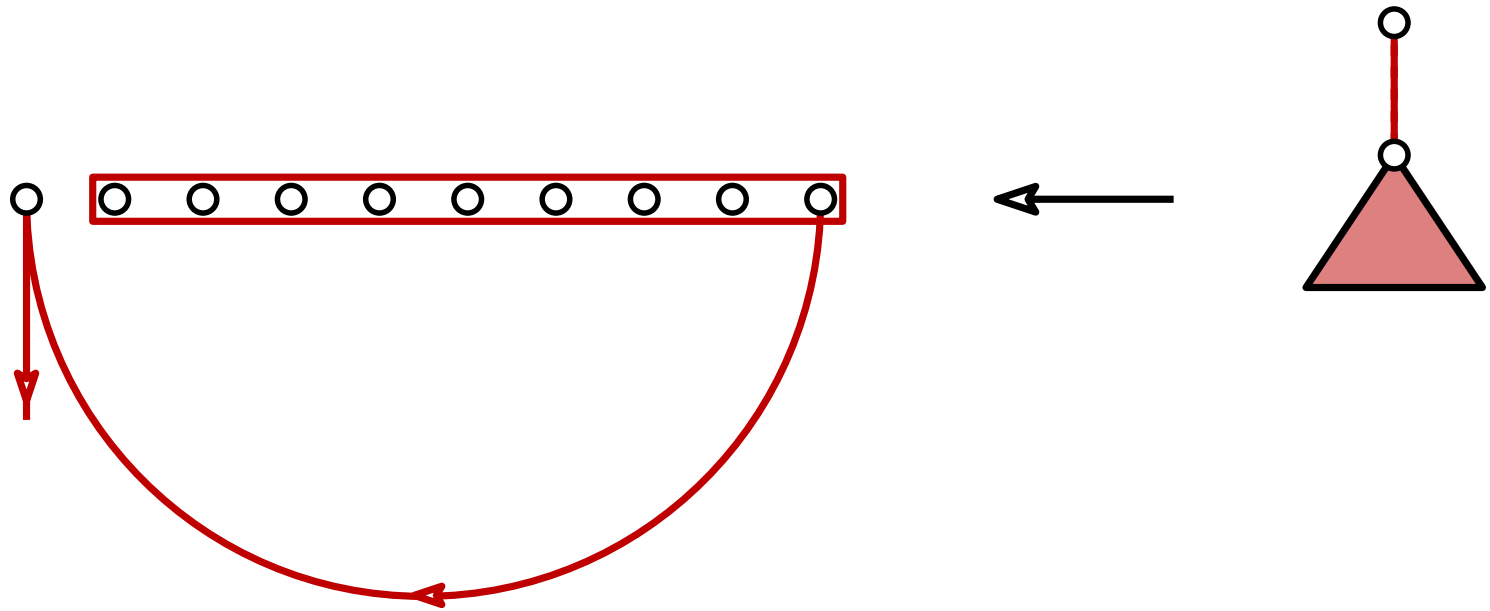
- The red tree is really small ($n \leq 5$).
- The red tree is small ($5 < n \leq 9$).
- The red tree is large and its root has one child.
- The red tree is large and its root has two children.

Embedding a large red tree with one child



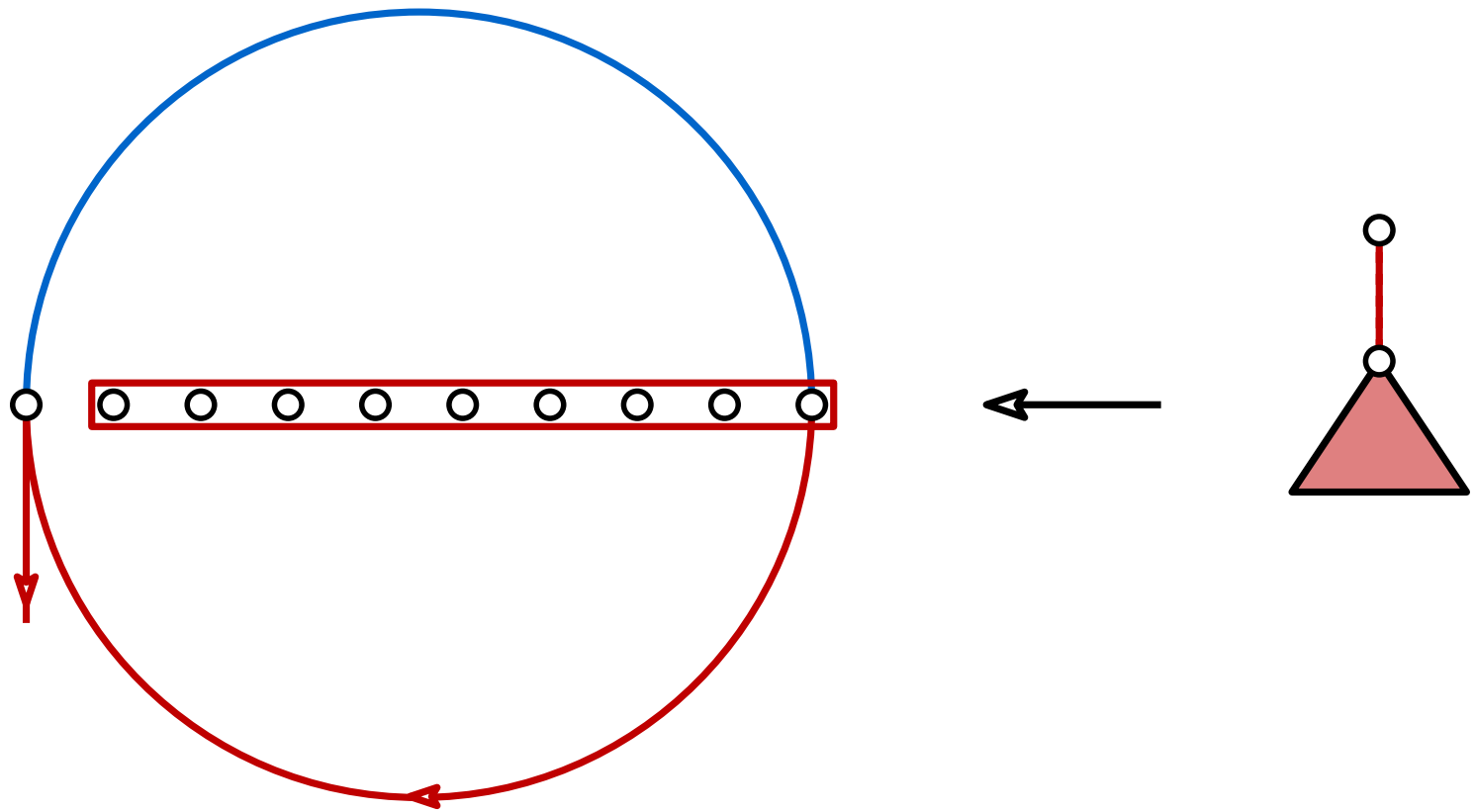
- **Placement invariant**: position i is not in conflict.
- **Local conformance**: the forest on $[i, j]$ satisfies the two rules.
- **No star**: the blue embedding on $[i, j]$ is not a star.

Embedding a large red tree with one child



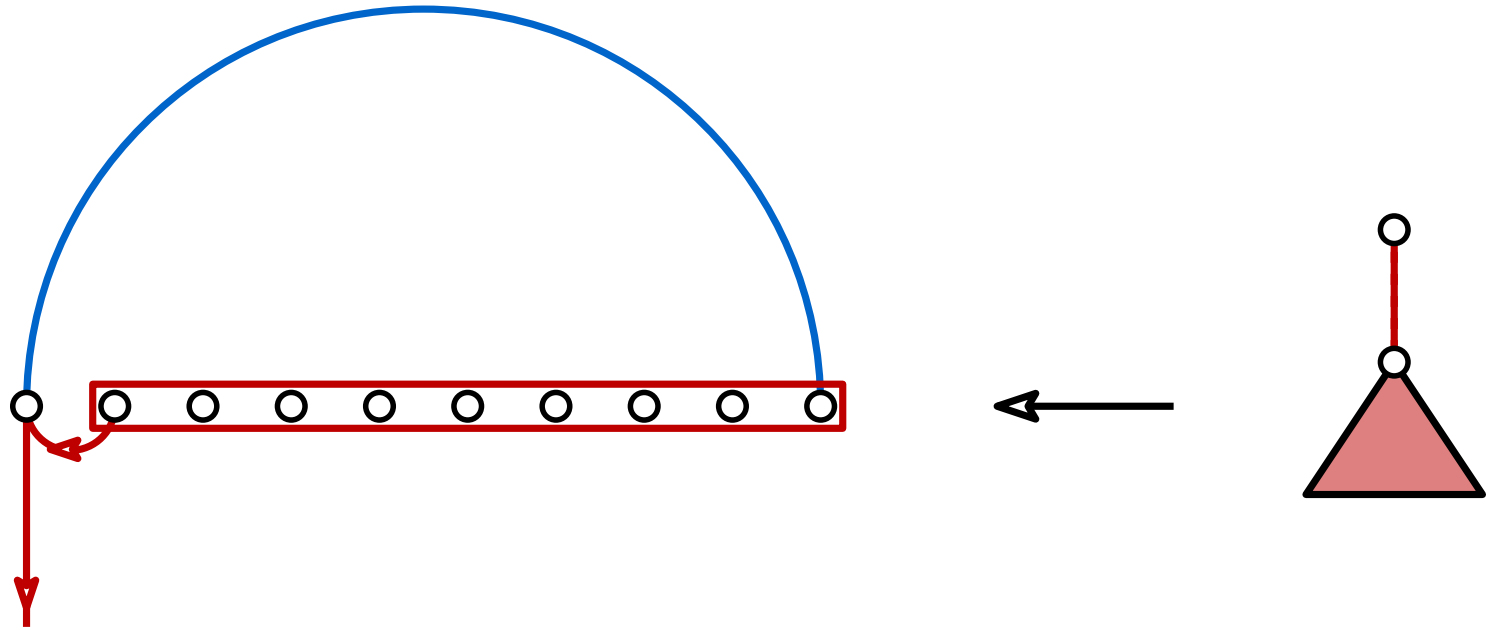
- **Placement invariant**: position i is not in conflict.
- **Local conformance**: the forest on $[i, j]$ satisfies the two rules.
- **No star**: the blue embedding on $[i, j]$ is not a star.

Embedding a large red tree with one child



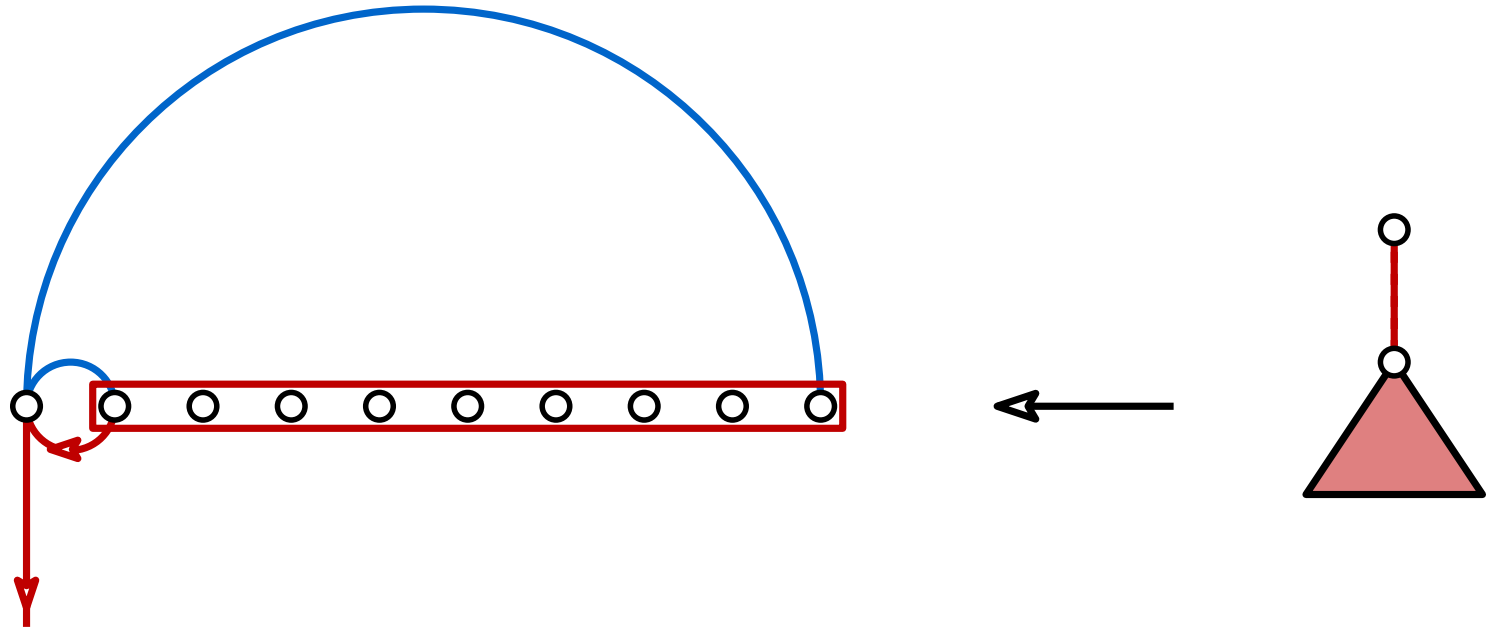
- **Placement invariant**: position i is not in conflict.
- **Local conformance**: the forest on $[i, j]$ satisfies the two rules.
- **No star**: the blue embedding on $[i, j]$ is not a star.

Embedding a large red tree with one child



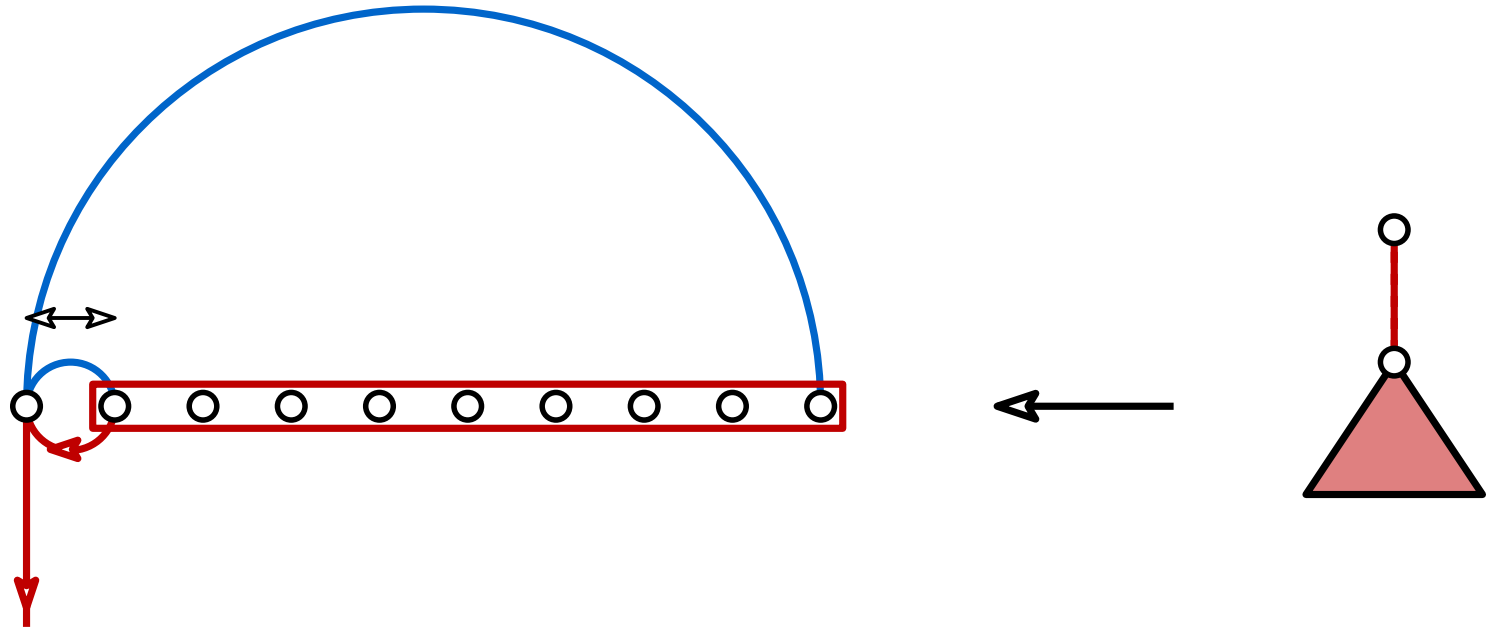
- **Placement invariant**: position i is not in conflict.
- **Local conformance**: the forest on $[i, j]$ satisfies the two rules.
- **No star**: the blue embedding on $[i, j]$ is not a star.

Embedding a large red tree with one child



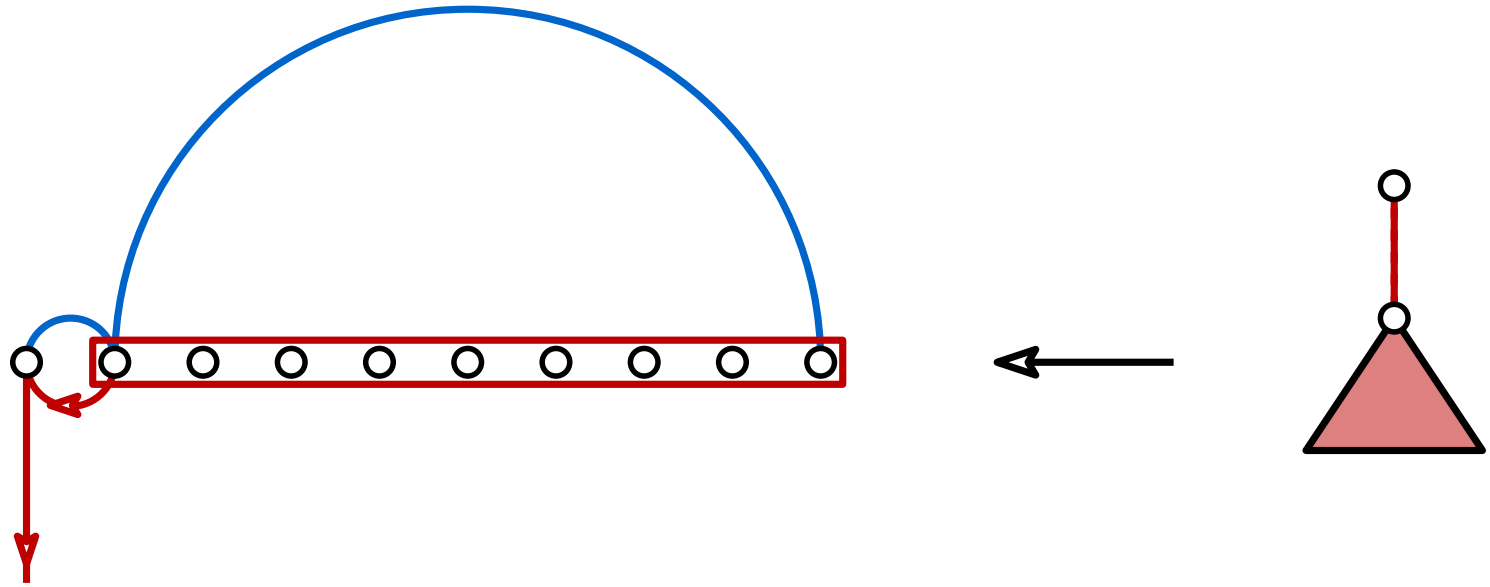
- **Placement invariant**: position i is not in conflict.
- **Local conformance**: the forest on $[i, j]$ satisfies the two rules.
- **No star**: the blue embedding on $[i, j]$ is not a star.

Embedding a large red tree with one child



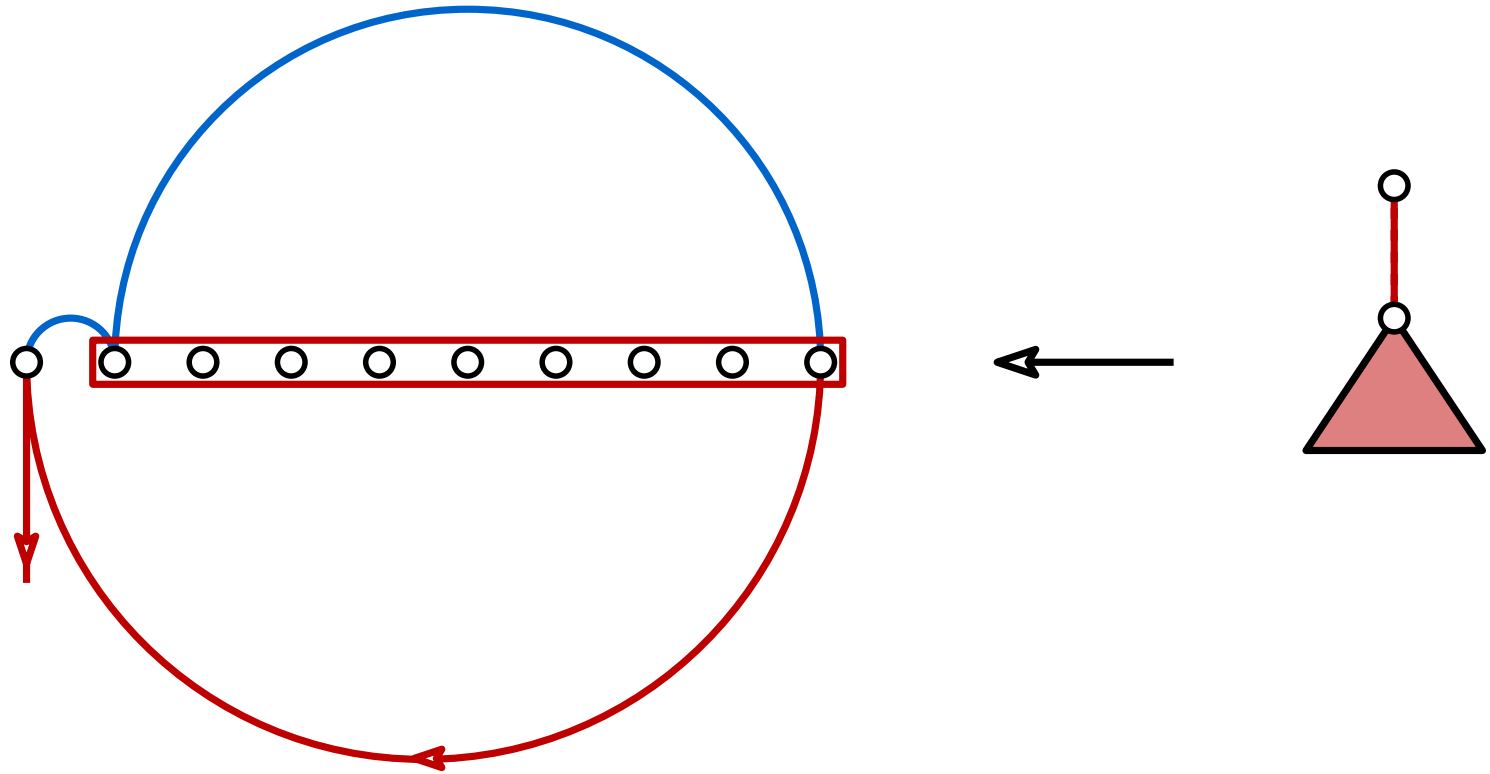
- **Placement invariant**: position i is not in conflict.
- **Local conformance**: the forest on $[i, j]$ satisfies the two rules.
- **No star**: the blue embedding on $[i, j]$ is not a star.

Embedding a large red tree with one child



- **Placement invariant**: position i is not in conflict.
- **Local conformance**: the forest on $[i, j]$ satisfies the two rules.
- **No star**: the blue embedding on $[i, j]$ is not a star.

Embedding a large red tree with one child



- **Placement invariant**: position i is not in conflict.
- **Local conformance**: the forest on $[i, j]$ satisfies the two rules.
- **No star**: the blue embedding on $[i, j]$ is not a star.

Conclusion

Theorem

Every two non-star binary trees on n vertices admit a planar packing.

Conjecture (García, Hernando, Hurtado, Noy, and Tejel 2002)

Every two non-star trees on n vertices admit a planar packing.

Conclusion

Theorem

Every two non-star binary trees on n vertices admit a planar packing.

Conjecture (García, Hernando, Hurtado, Noy, and Tejel 2002)

Every two non-star trees on n vertices admit a planar packing.

Thanks!

References I

- Cardinal, Jean, Michael Hoffmann, and Vincent Kusters (2013). "On Universal Point Sets for Planar Graphs". In: *Proc. Thailand-Japan Joint Conference on Computational Geometry and Graphs*. Vol. 8296. Lecture Notes in Computer Science. Springer-Verlag, pp. 30–41. DOI: 10.1007/978-3-642-45281-9_3. eprint: http://arxiv.org/abs/1209.3594.
- Cardinal, Jean and Vincent Kusters (n.d.). "The Complexity of Simultaneous Geometric Graph Embedding".
- Frati, F. (2009). "Planar Packing of Diameter-Four Trees". In: *21st Canadian Conference on Computational Geometry (CCCG '09)*, pp. 95–98.
- Frati, F., M. Geyer, and M. Kaufmann (Feb. 2009). "Planar Packings of Trees and Spider Trees". In: *Information Processing Letters* 109.6, pp. 301–307. DOI: http://dx.doi.org/10.1016/j.ipl.2008.11.002.

References II

- García, A., C. Hernando, F. Hurtado, M. Noy, and J. Tejel (2002). "Packing trees into planar graphs". In: *J. Graph Theory*, pp. 172–181.
- Geyer, Markus, Michael Kaufmann, Michael Hoffmann, Vincent Kusters, and Csaba D. Tóth (2013). "Planar Packing of Binary Trees". In: *Algorithms and Data Structures*. Ed. by Frank Dehne, Roberto Solis-Oba, and Jörg-Rüdiger Sack. Vol. 8037. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 353–364. DOI: 10.1007/978-3-642-40104-6_31.
- Hedetniemi, SM, ST Hedetniemi, and PJ Slater (1981). "A note on packing two trees into KN". In: *Ars Combin* 11, pp. 149–153.
- Hoffmann, Michael, Vincent Kusters, and Tillmann Miltzow (n.d.). "Halving Balls in Deterministic Linear Time".
- Hoffmann, Michael, Vincent Kusters, Günter Rote, Maria Saumell, and Rodrigo I. Silveira (n.d.). "Convex hull alignment through translation". In: *CCCG'13*, pp. 295–300.

References III

- Kusters, Vincent and Bettina Speckmann (n.d.). "Characterizing Graphs With A Sliceable Rectangular Dual".
- Maheo, Maryvonne, Jean-François Saclé, and Mariusz Woźniak (1996). "Edge-disjoint placement of three trees". In: *European Journal of Combinatorics* 17.6, pp. 543–563. DOI: 10.1006/eujc.1996.0047.
- Oda, Y. and K. Ota (2006). "Tight planar packings of two trees". In: *European Workshop on Computational Geometry*, pp. 215–216.