Karlsruhe Institute of Technology

A New Perspective on Clustered Planarity as a Combinatorial Embedding Problem

Würzburg · GD 2014 · September 26

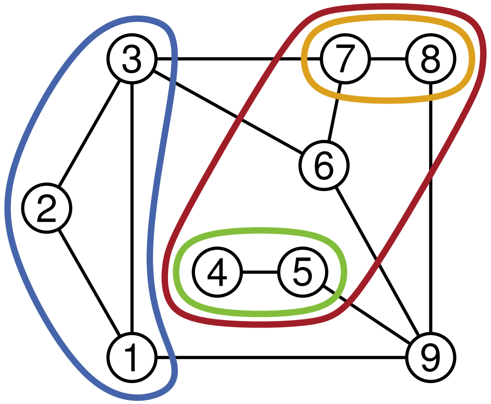
Thomas Bläsius · Ignaz Rutter

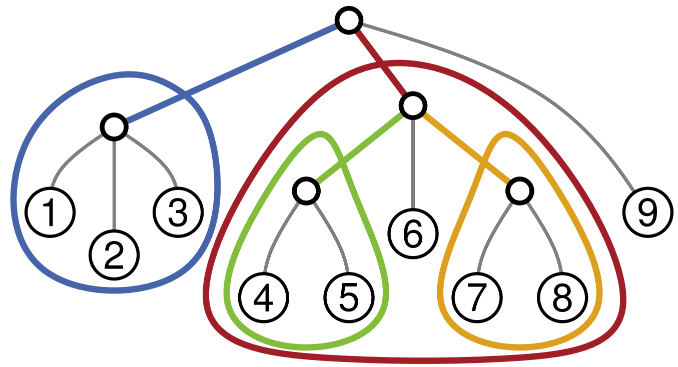
INSTITUTE OF THEORETICAL INFORMATICS · PROF. DR. DOROTHEA WAGNER

Clustered Planarity

[**Lengauer** 1989]
[**Feng, Cohen, Eades** 1995]

Input: graph $G = (V, E)$ tree T with leaves V



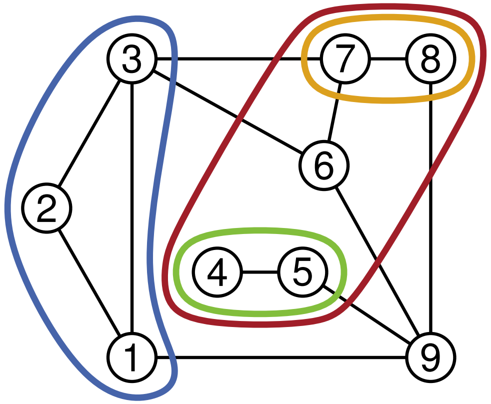


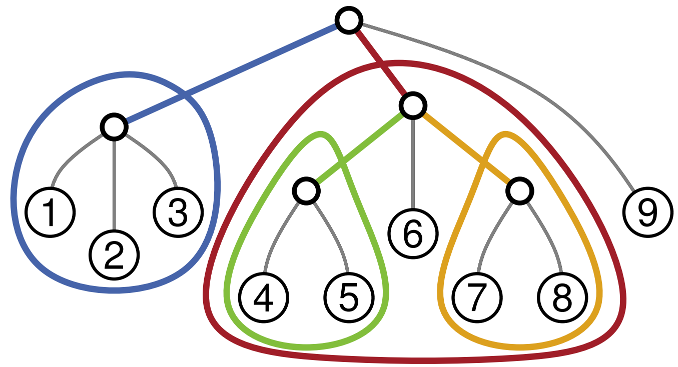
every subtree of T induces a cluster

Clustered Planarity

[**Lengauer** 1989]
[**Feng, Cohen, Eades** 1995]

Input: graph $G = (V, E)$ tree T with leaves V

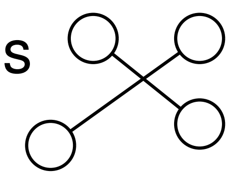




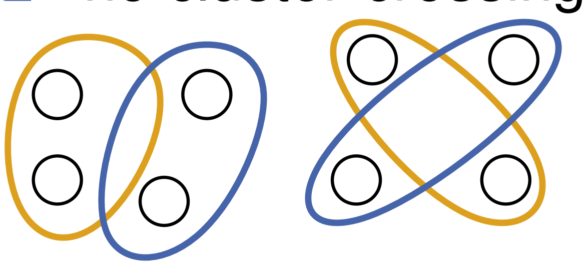
every subtree of T induces a cluster

Find: drawing of G together with regions representing the clusters

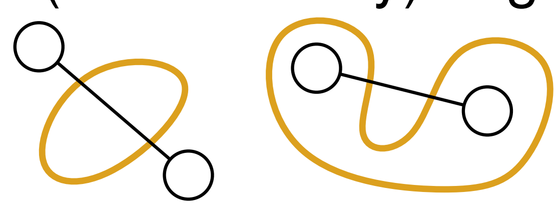
- no edge-crossings



- no cluster-crossings



- no (unnecessary) edge-cluster-crossings



My Goals for this Talk

Give you an Understanding of our Perspective on C-Planarity

- the cd-tree and a characterization
- flat clusterings and constrained planarity
- related work from the new perspective

no new c-planarity variants will be solved in this part

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

My Goals for this Talk

Give you an Understanding of our Perspective on C-Planarity

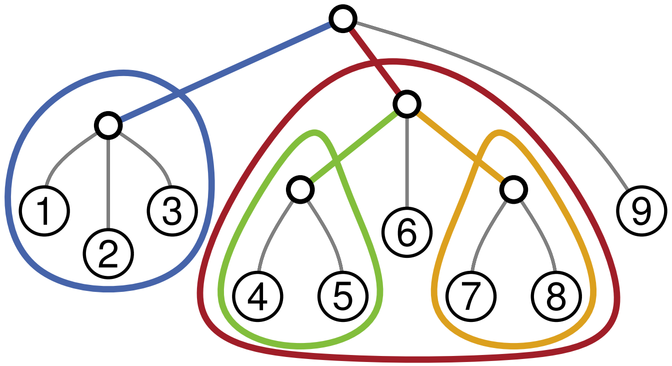
- the cd-tree and a characterization
- flat clusterings and constrained planarity
- related work from the new perspective

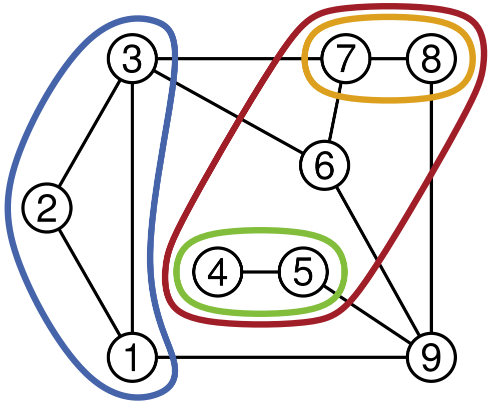
no new c-planarity variants will be solved in this part

Final Remarks

- new cases we can solve

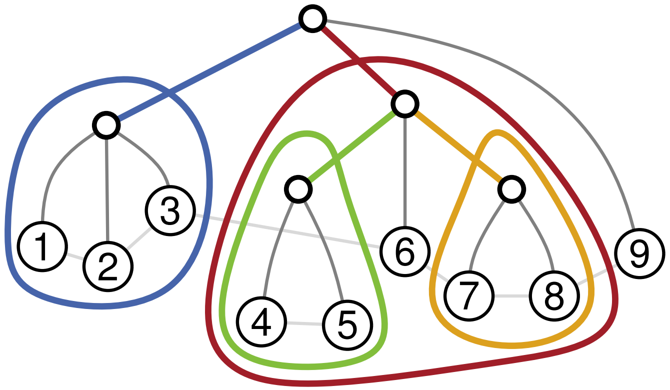
The CD-Tree – A (New) Perspective

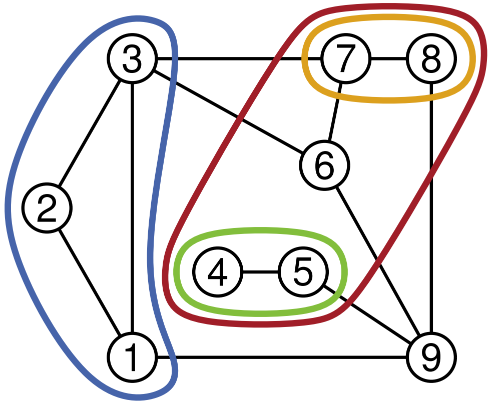




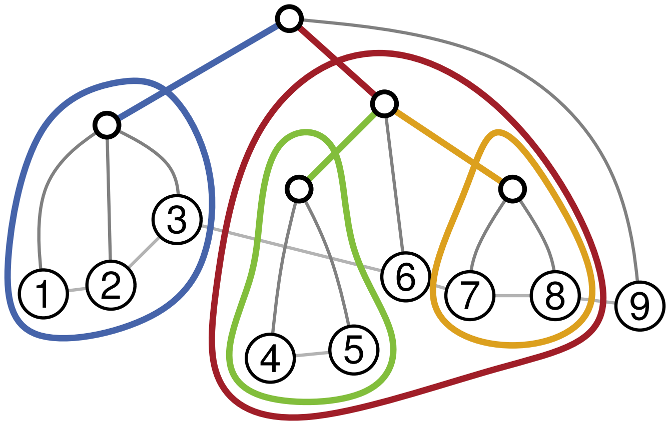
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

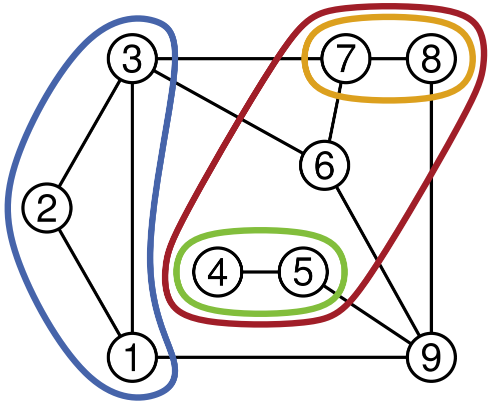
The CD-Tree – A (New) Perspective



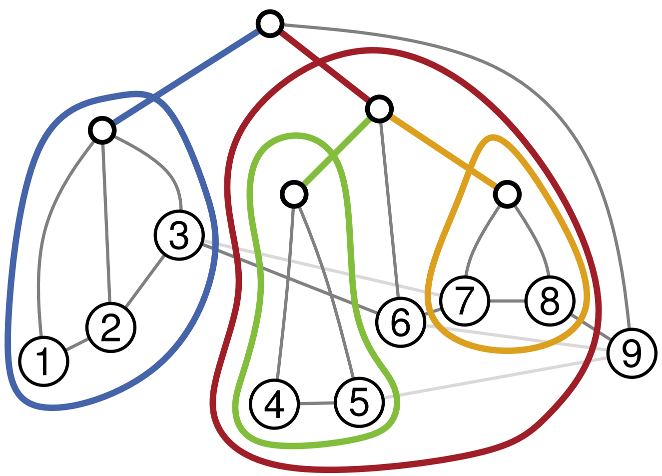


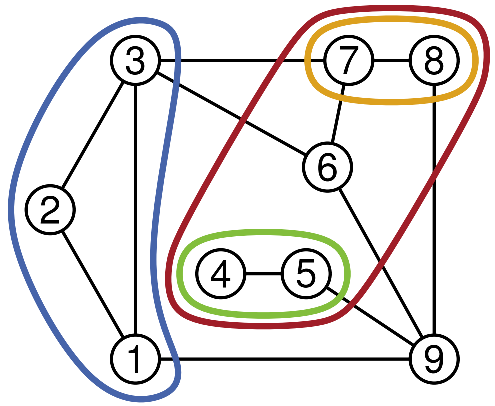
The CD-Tree – A (New) Perspective



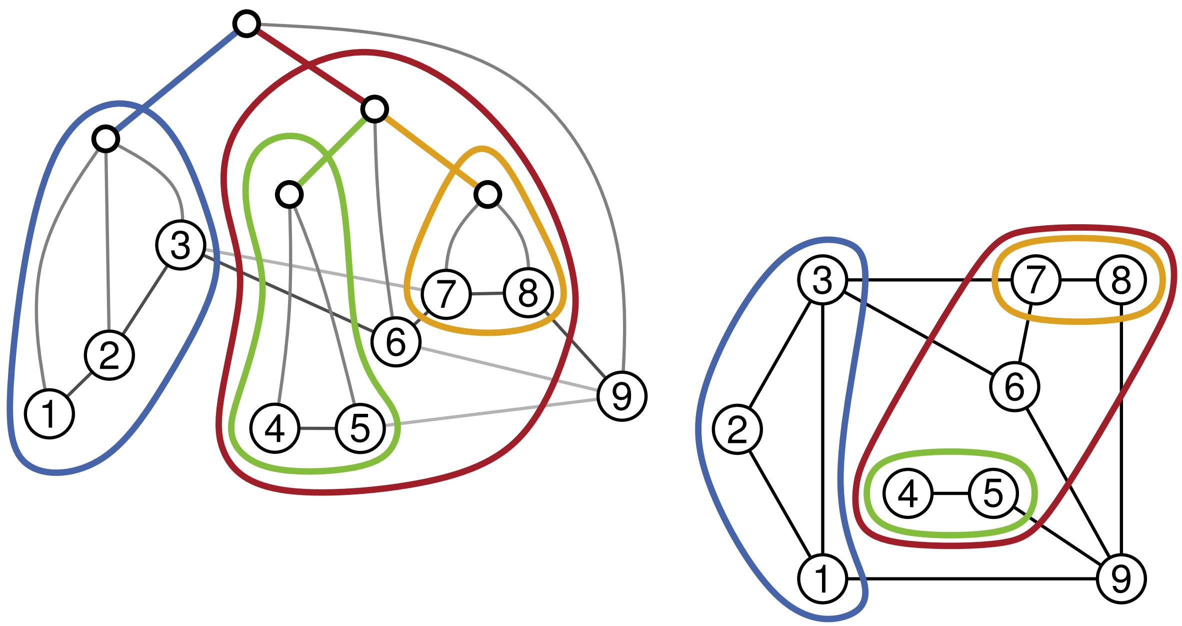


The CD-Tree – A (New) Perspective



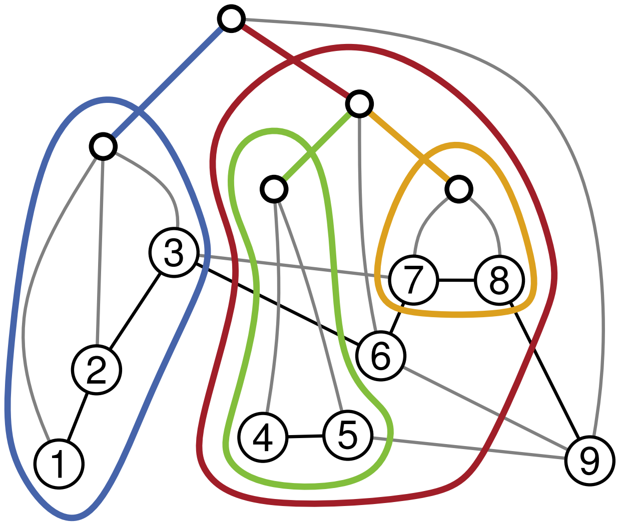


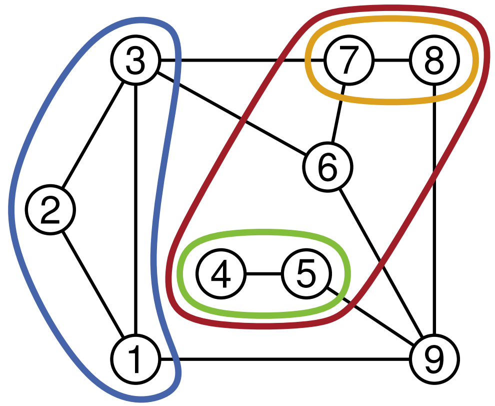
The CD-Tree – A (New) Perspective



Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

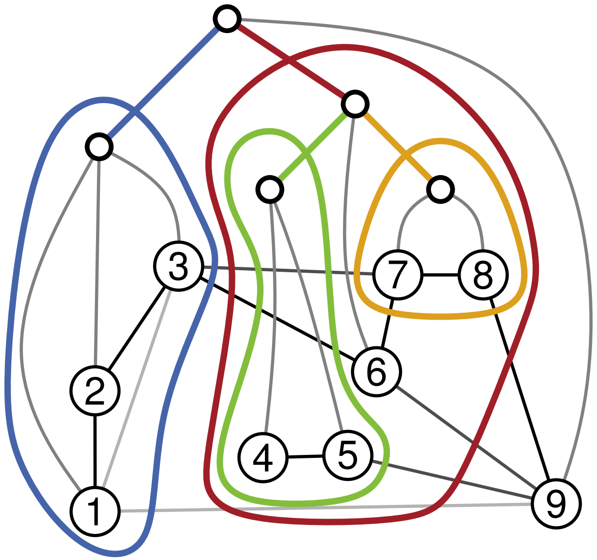
The CD-Tree – A (New) Perspective

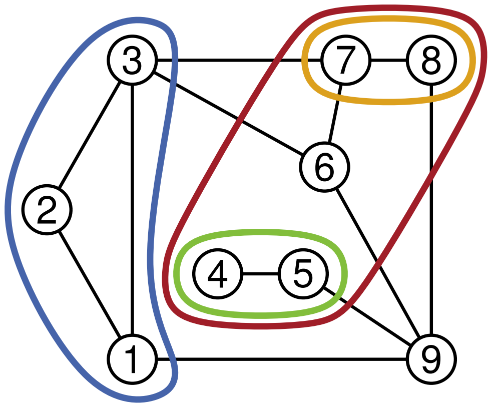




Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

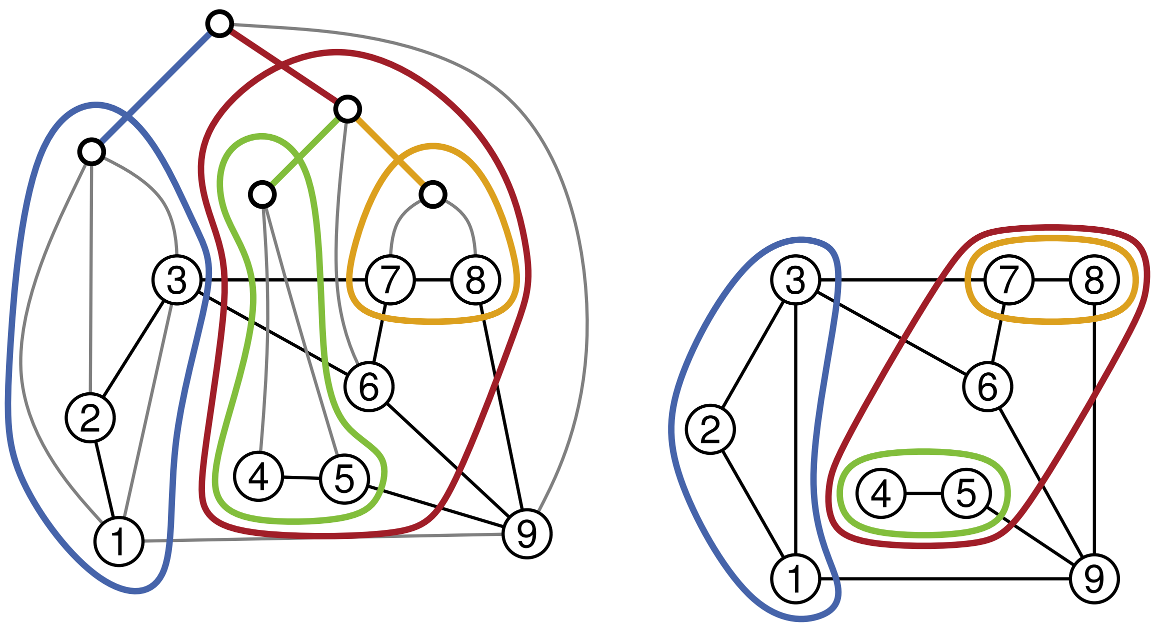
The CD-Tree – A (New) Perspective



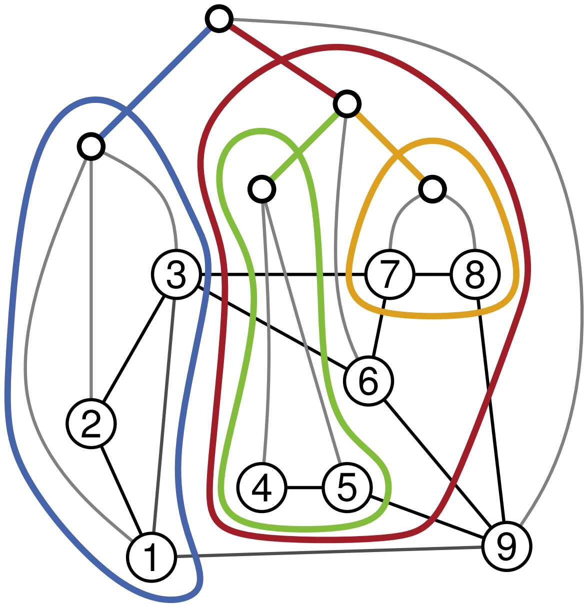


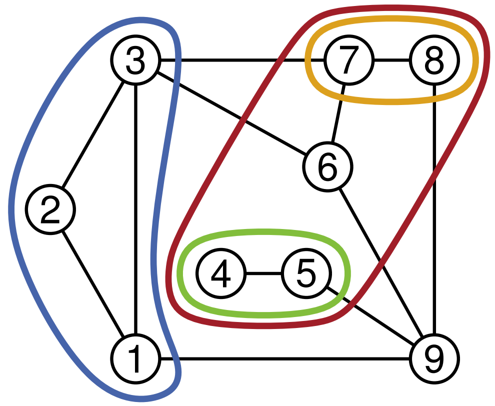
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective



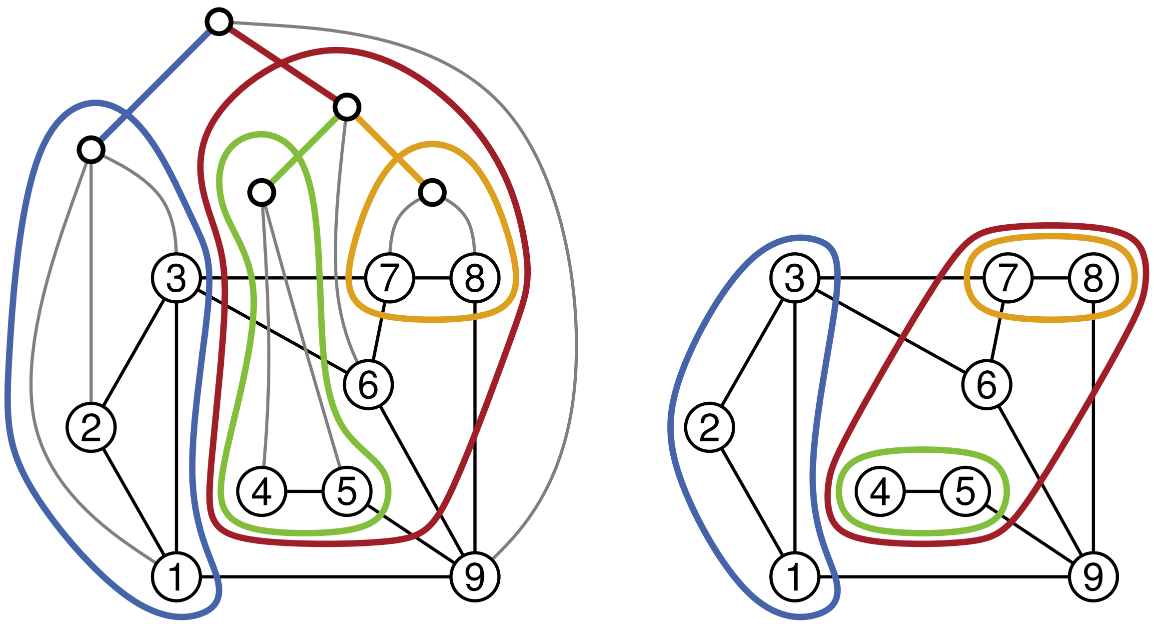
The CD-Tree – A (New) Perspective





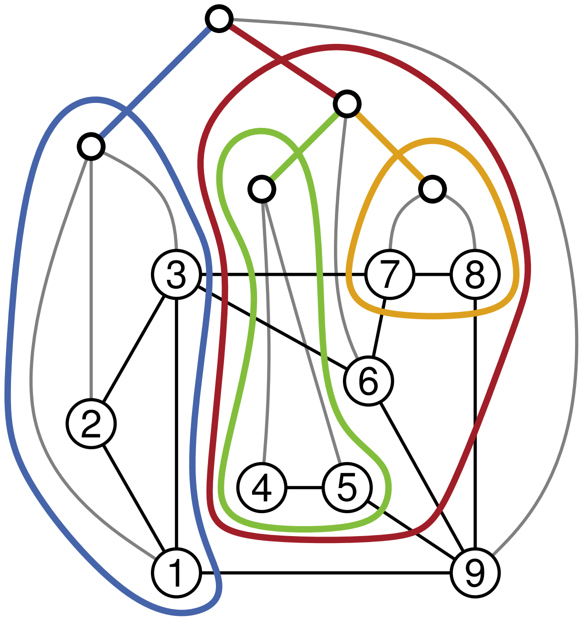
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective

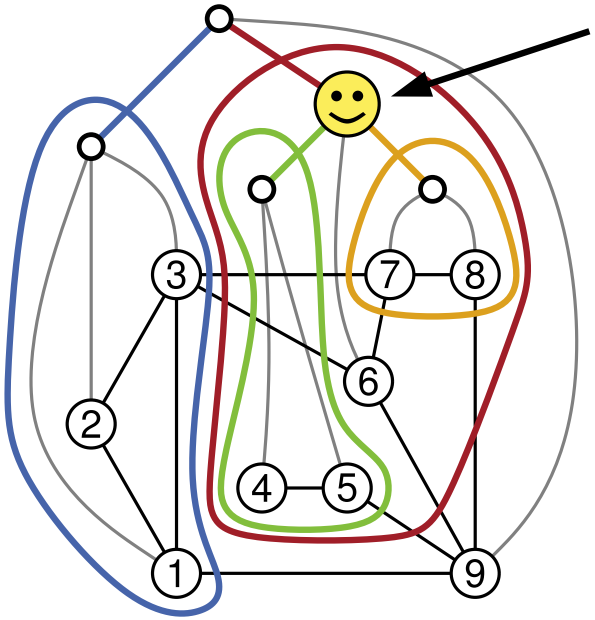


Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective



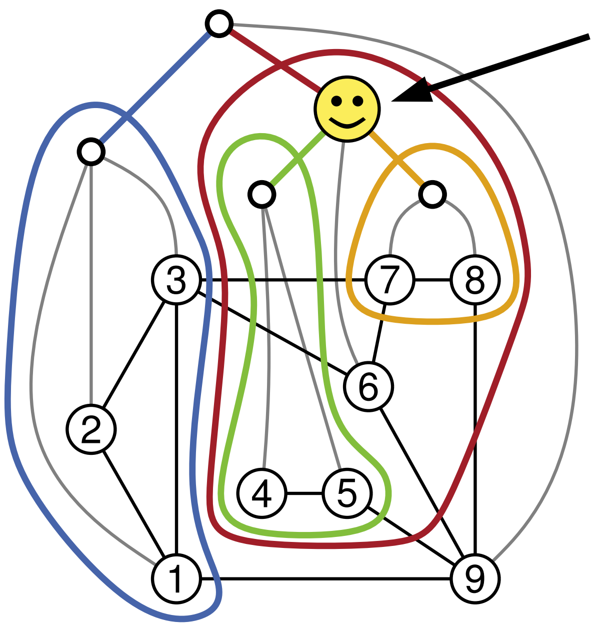
The CD-Tree – A (New) Perspective



From the perspective of this node, there are

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective

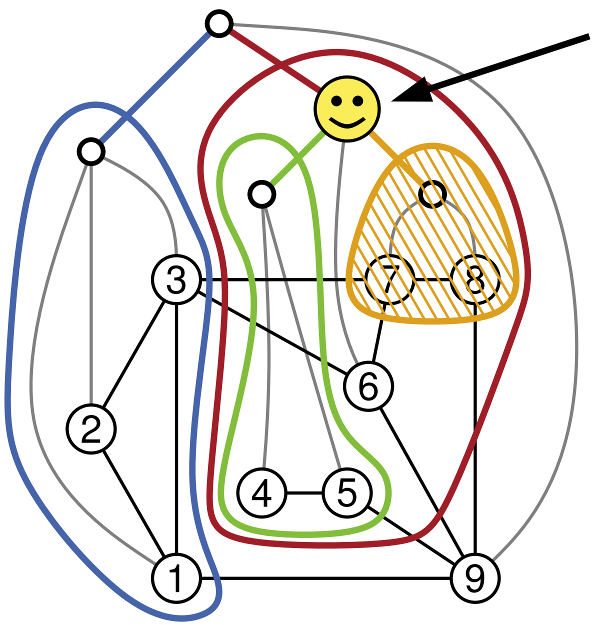


From the perspective of this node, there are

- node 6

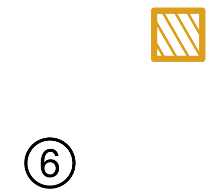
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective



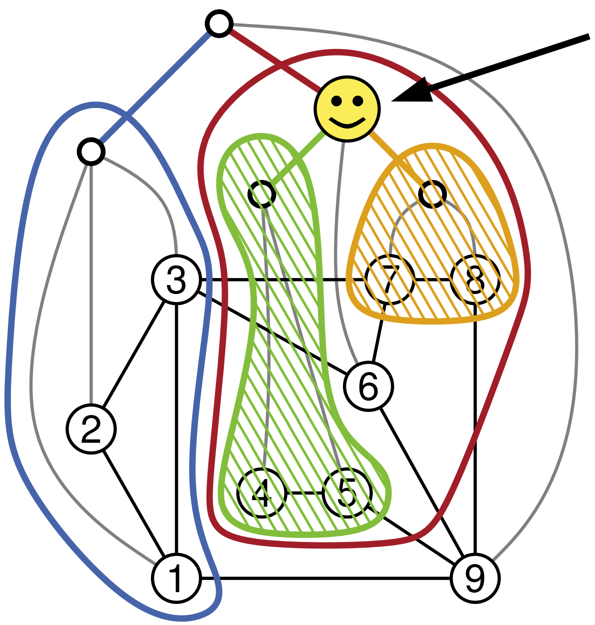
From the perspective of this node, there are

- node 6
- orange part



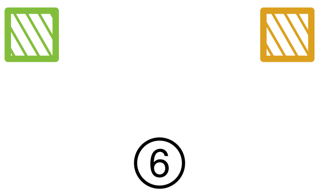
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective



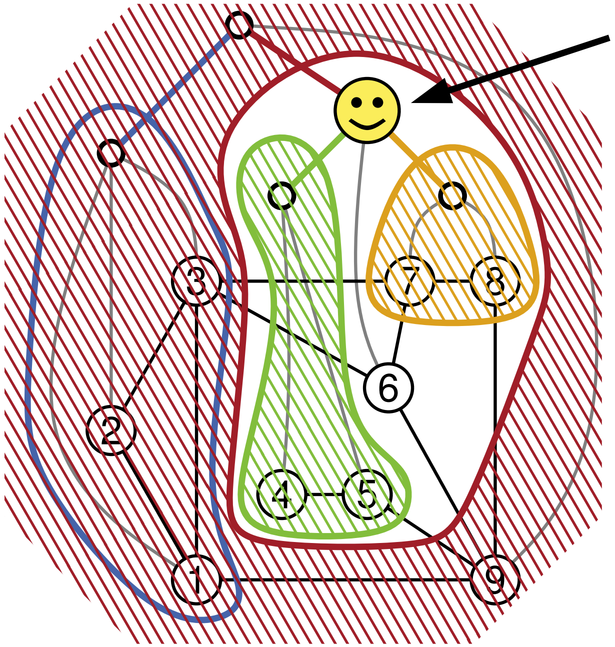
From the perspective of this node, there are

- node 6
- orange part
- green part



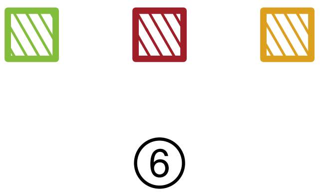
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective



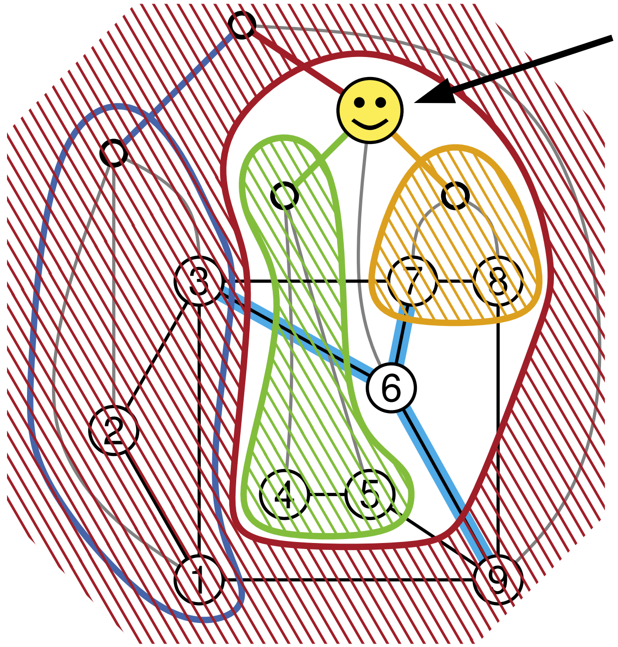
From the perspective of this node, there are

- node 6
- orange part
- green part
- red part



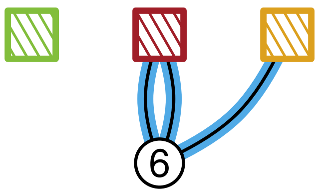
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective



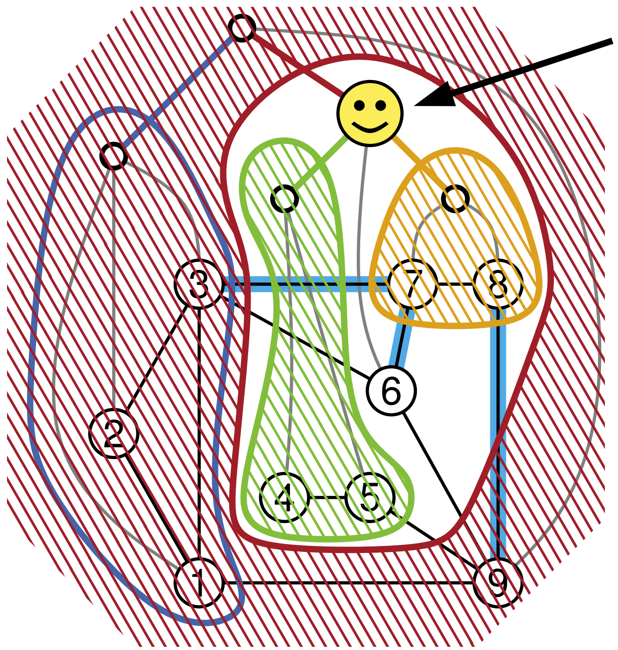
From the perspective of this node, there are

- node 6
- orange part
- green part
- red part
- some edges



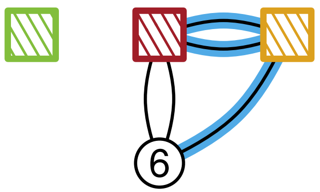
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective



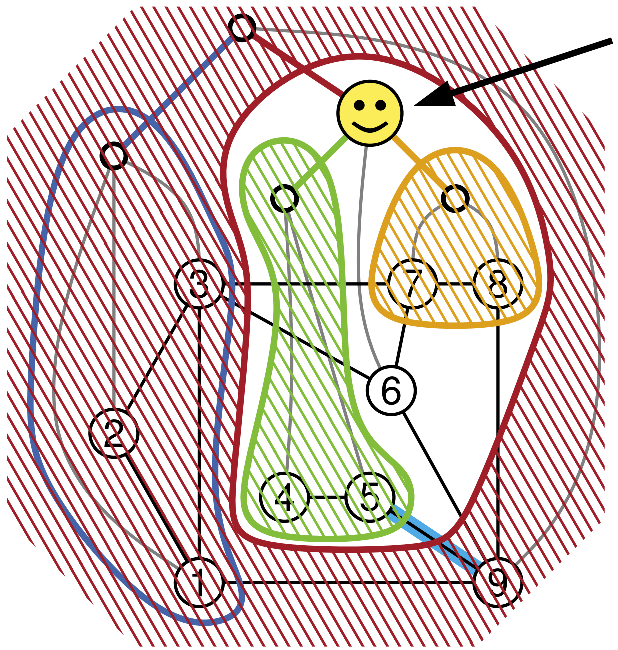
From the perspective of this node, there are

- node 6
- orange part
- green part
- red part
- some edges



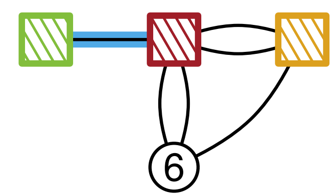
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective



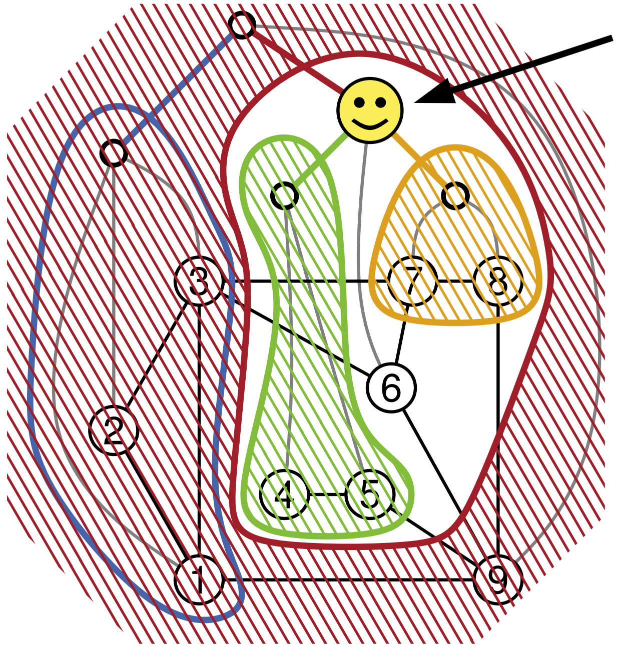
From the perspective of this node, there are

- node 6
- orange part
- green part
- red part
- some edges



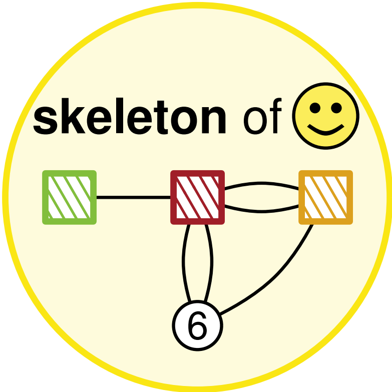
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective



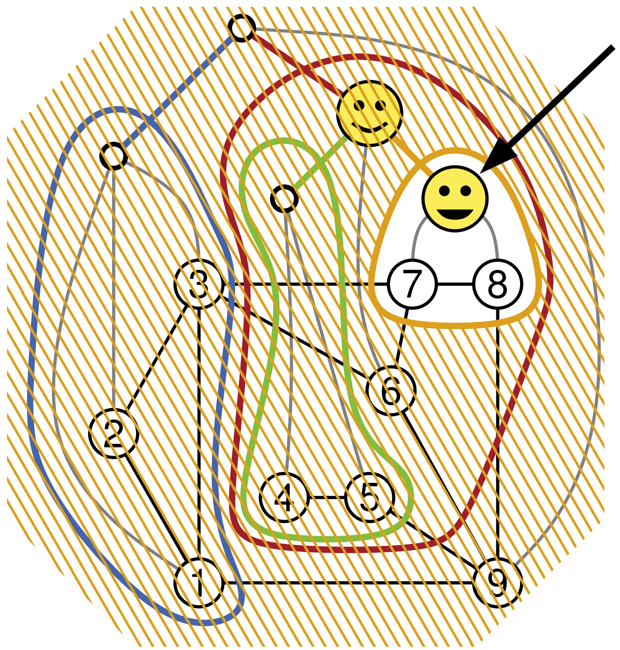
From the perspective of this node, there are

- node 6
- orange part
- green part
- red part
- some edges

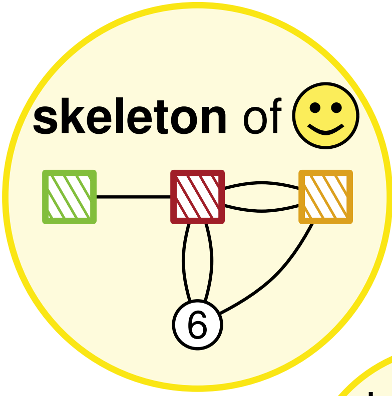


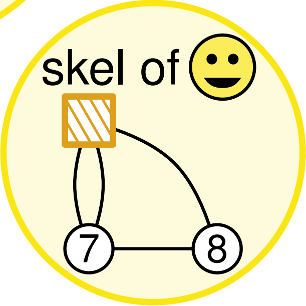
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective

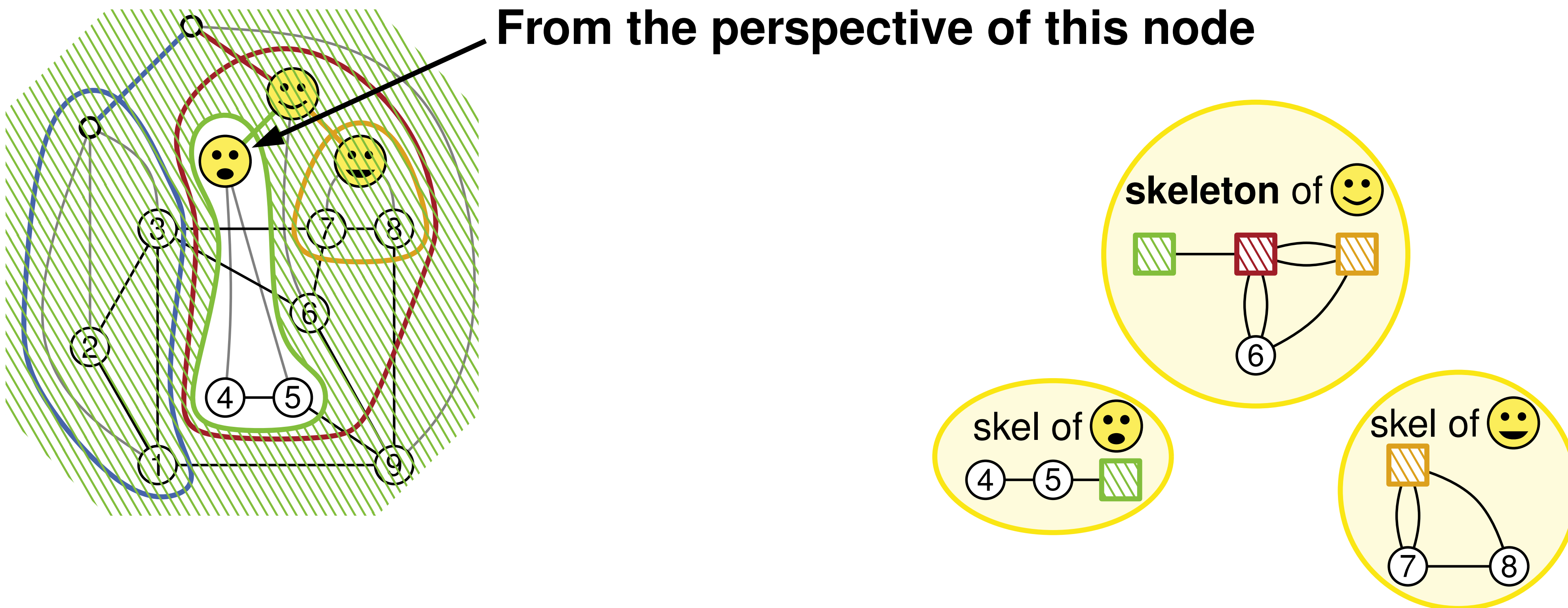


From the perspective of this node



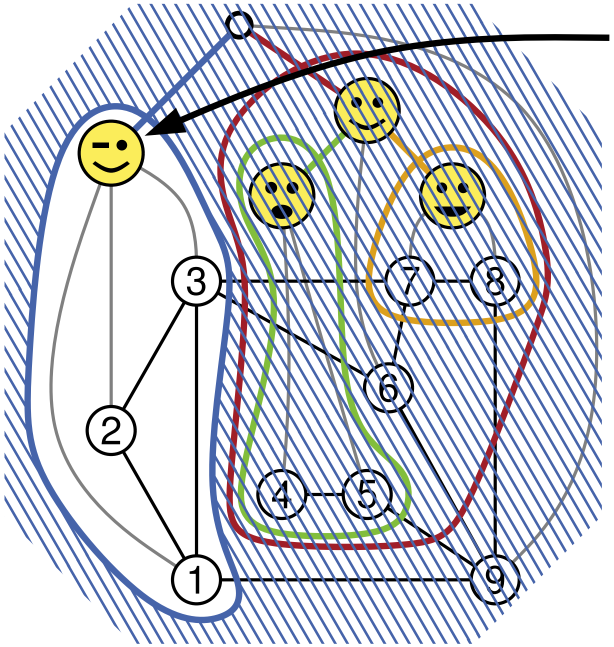


The CD-Tree – A (New) Perspective

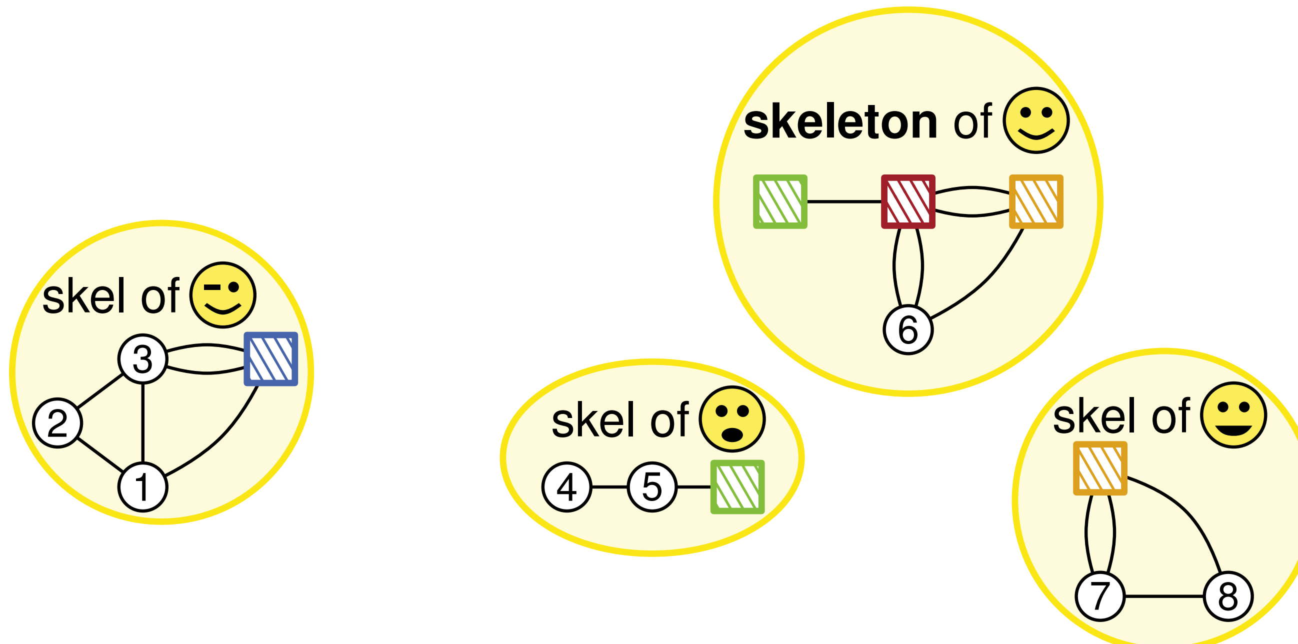


Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective

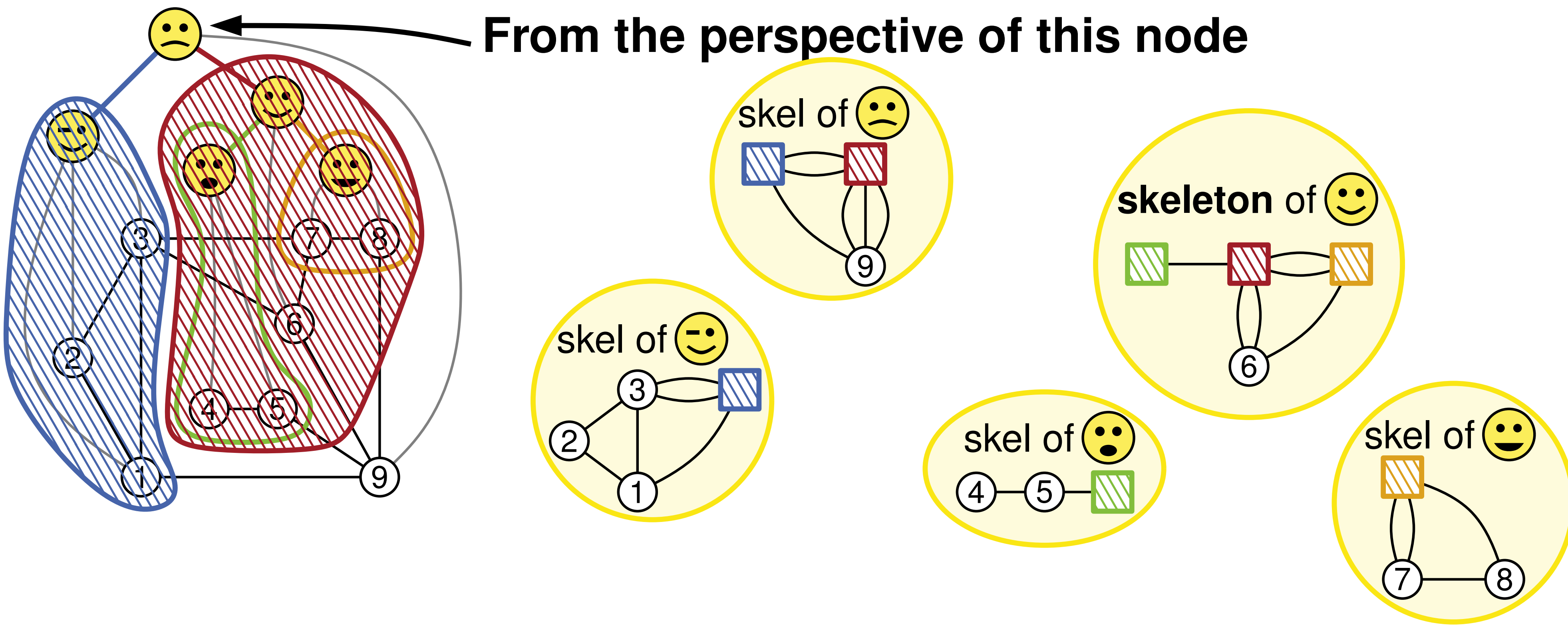


From the perspective of this node



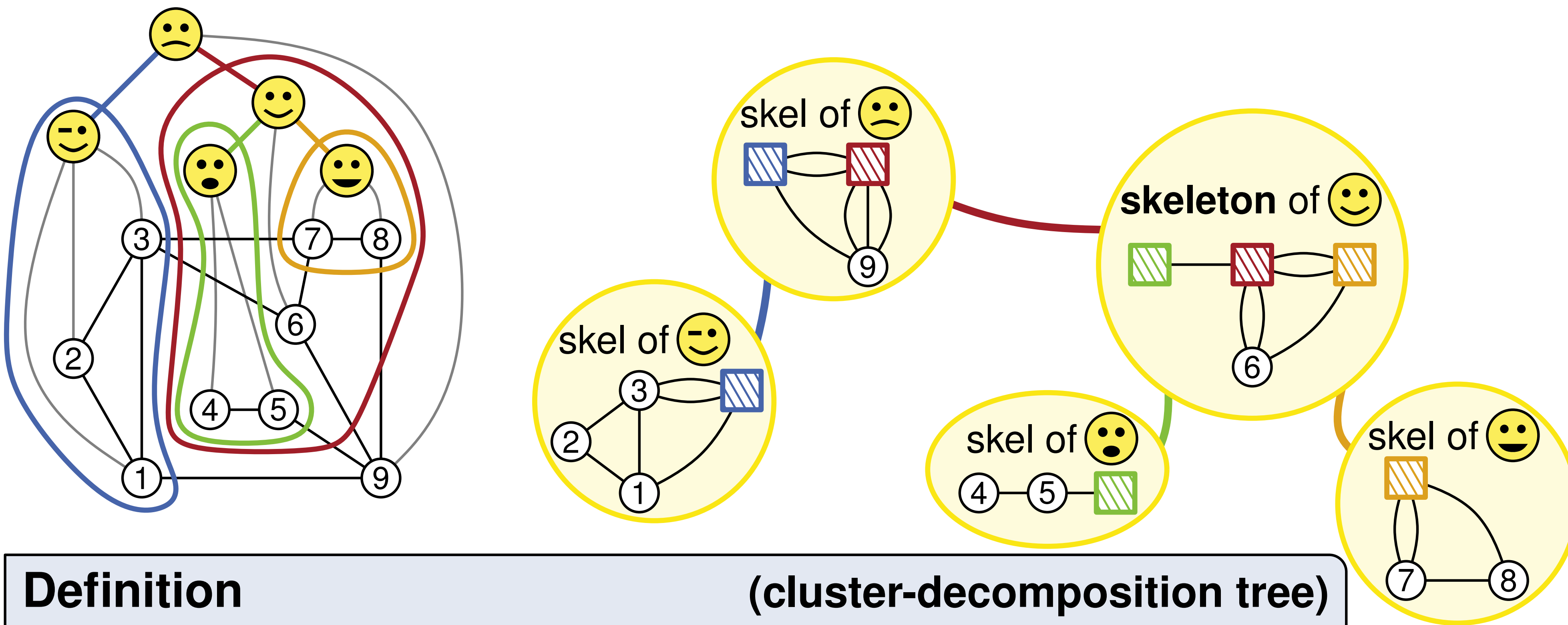
The CD-Tree – A (New) Perspective

From the perspective of this node



Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

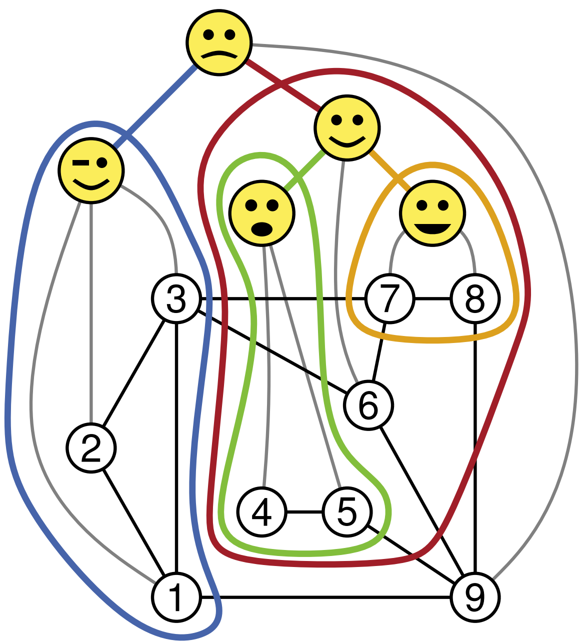
The CD-Tree – A (New) Perspective

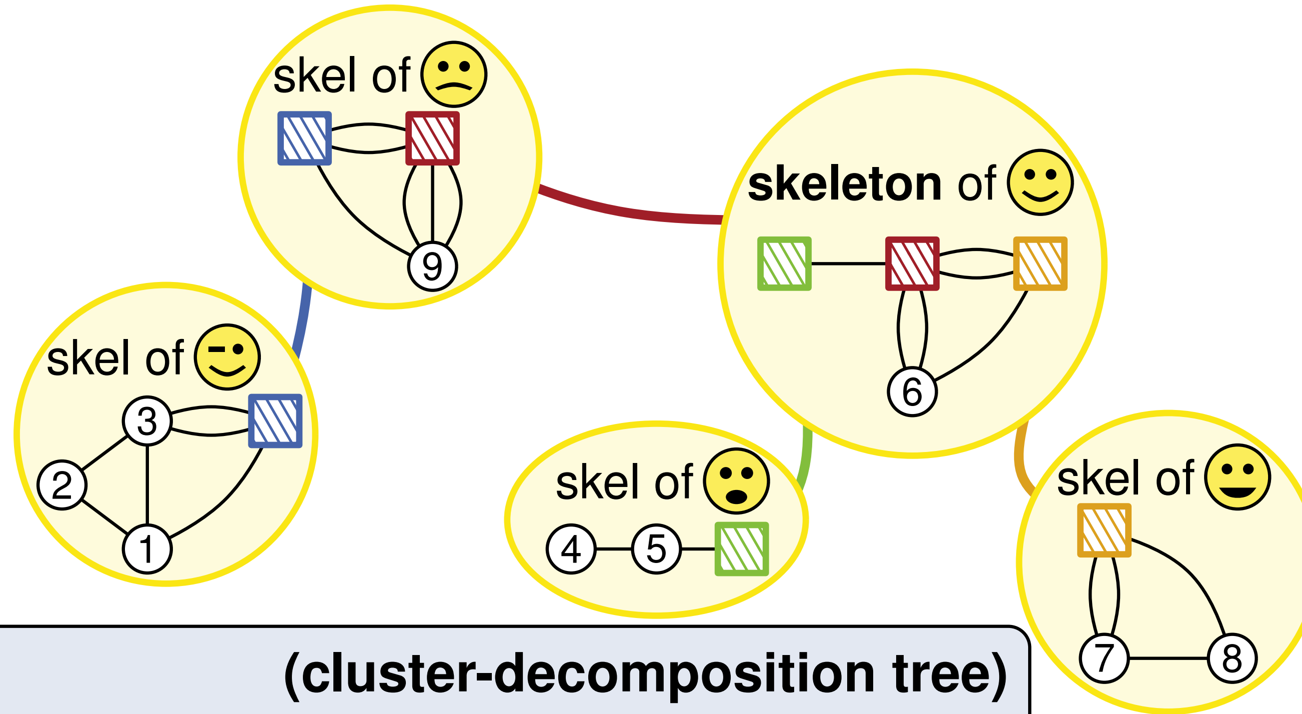


Definition **(cluster-decomposition tree)**

The **cd-tree** of a clustered graph (G, T) is T together with a bijection between T's nodes and their skeletons.

The CD-Tree – A (New) Perspective





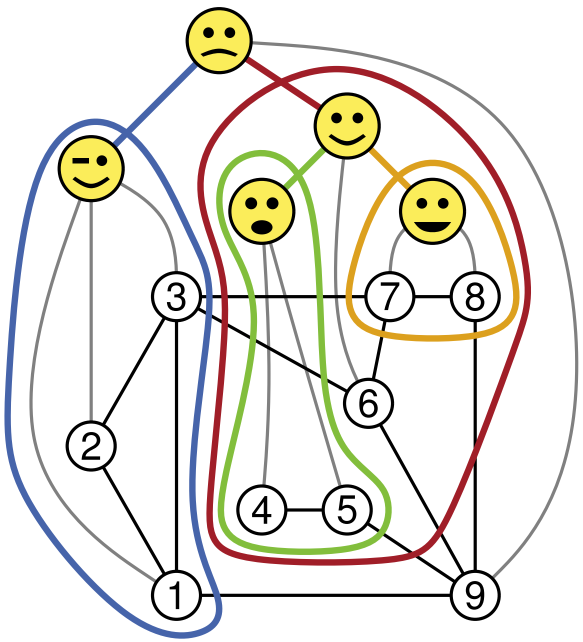
Definition **(cluster-decomposition tree)**

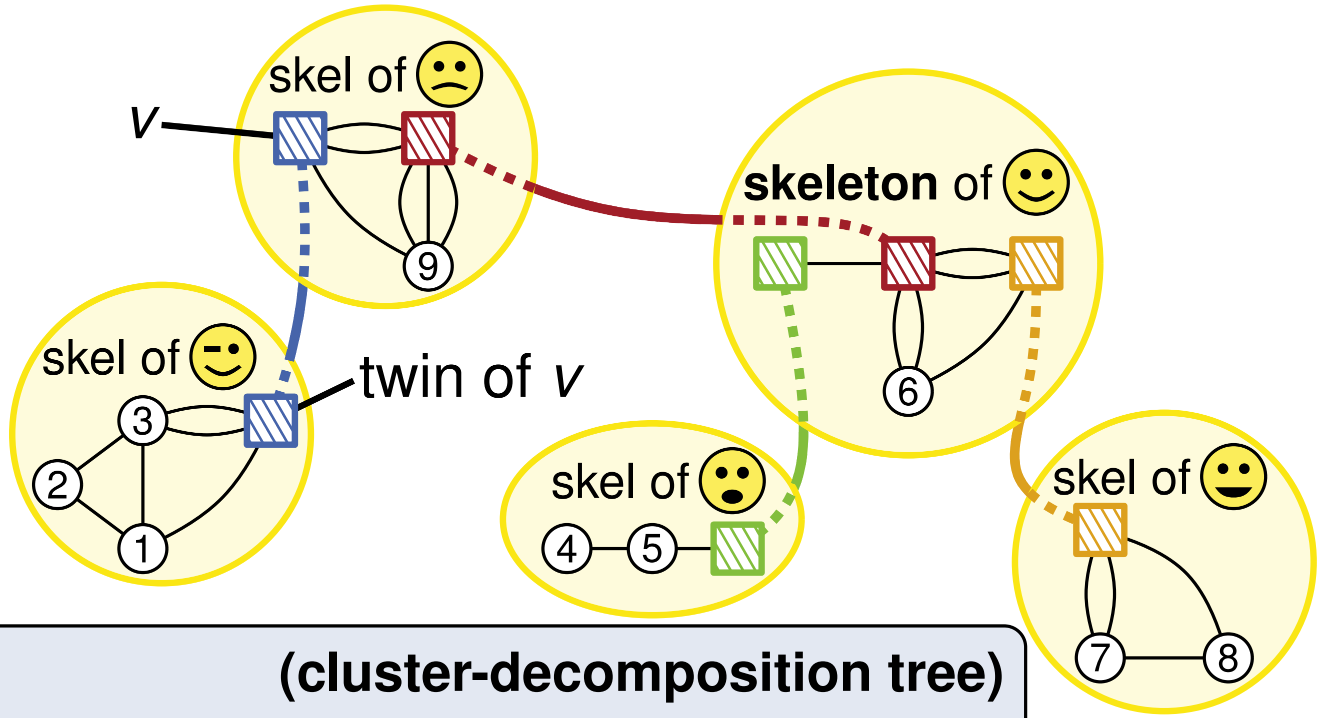
The **cd-tree** of a clustered graph (G, T) is T together with a bijection between T's nodes and their skeletons.

- The new vertices in skeletons (▨) are **virtual vertices**.

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective





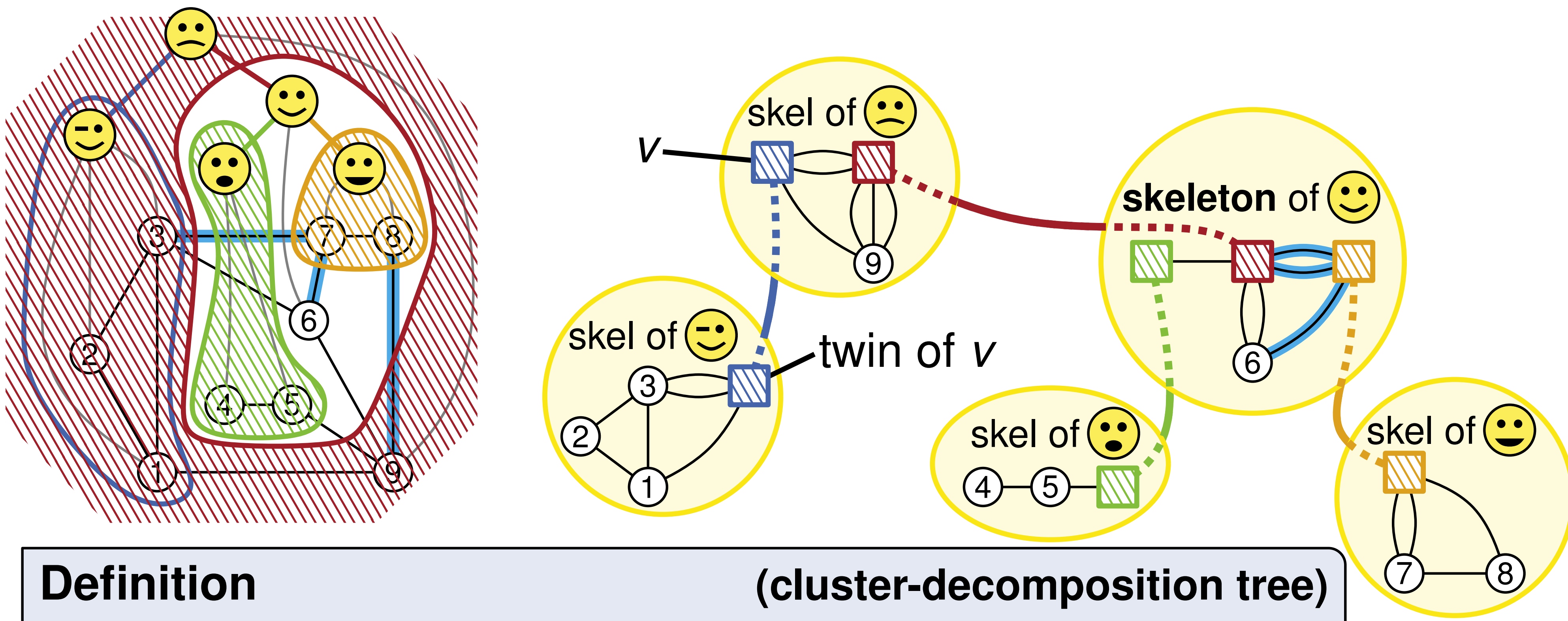
Definition **(cluster-decomposition tree)**

The **cd-tree** of a clustered graph (G, T) is T together with a bijection between T's nodes and their skeletons.

- The new vertices in skeletons (▧) are **virtual vertices**.
- Every virtual vertex has a twin.

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective

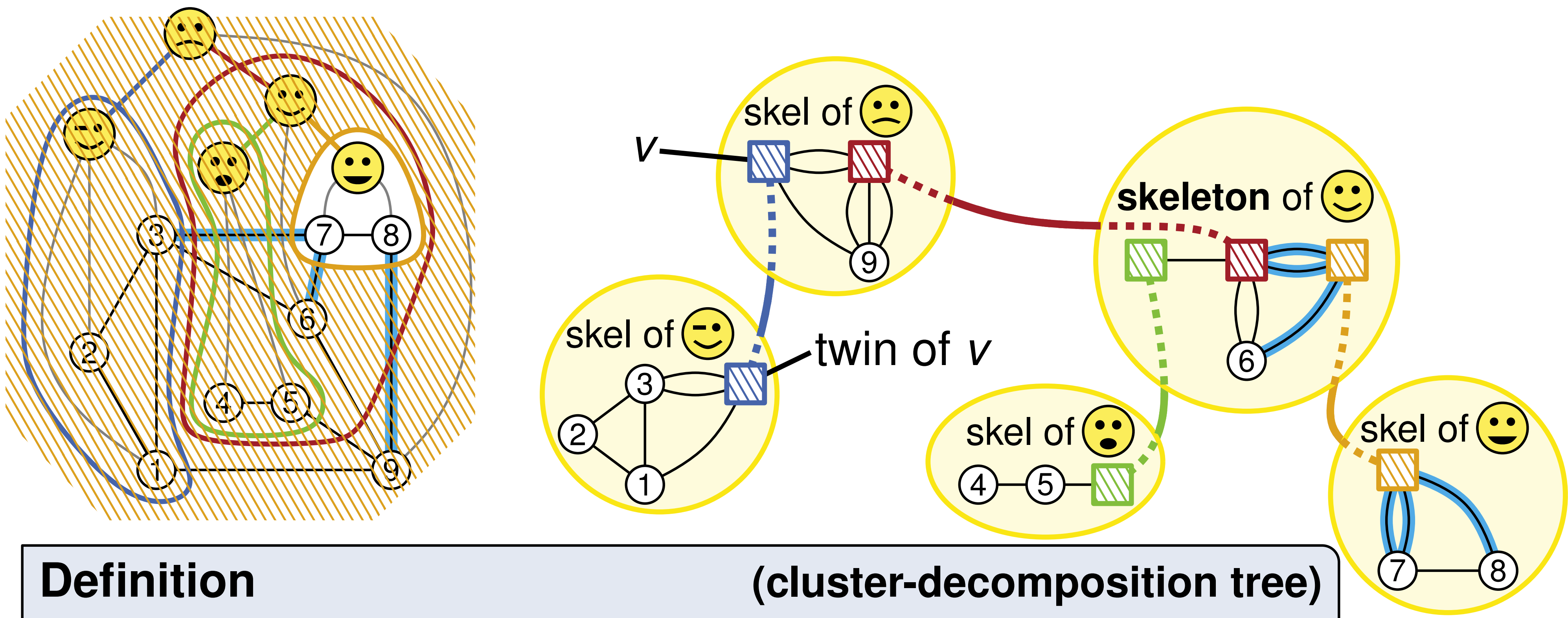


Definition **(cluster-decomposition tree)**

The **cd-tree** of a clustered graph (G, T) is T together with a bijection between T's nodes and their skeletons.

- The new vertices in skeletons (▧) are **virtual vertices**.
- Every virtual vertex has a twin.
- Twins have the same incident edges.

The CD-Tree – A (New) Perspective

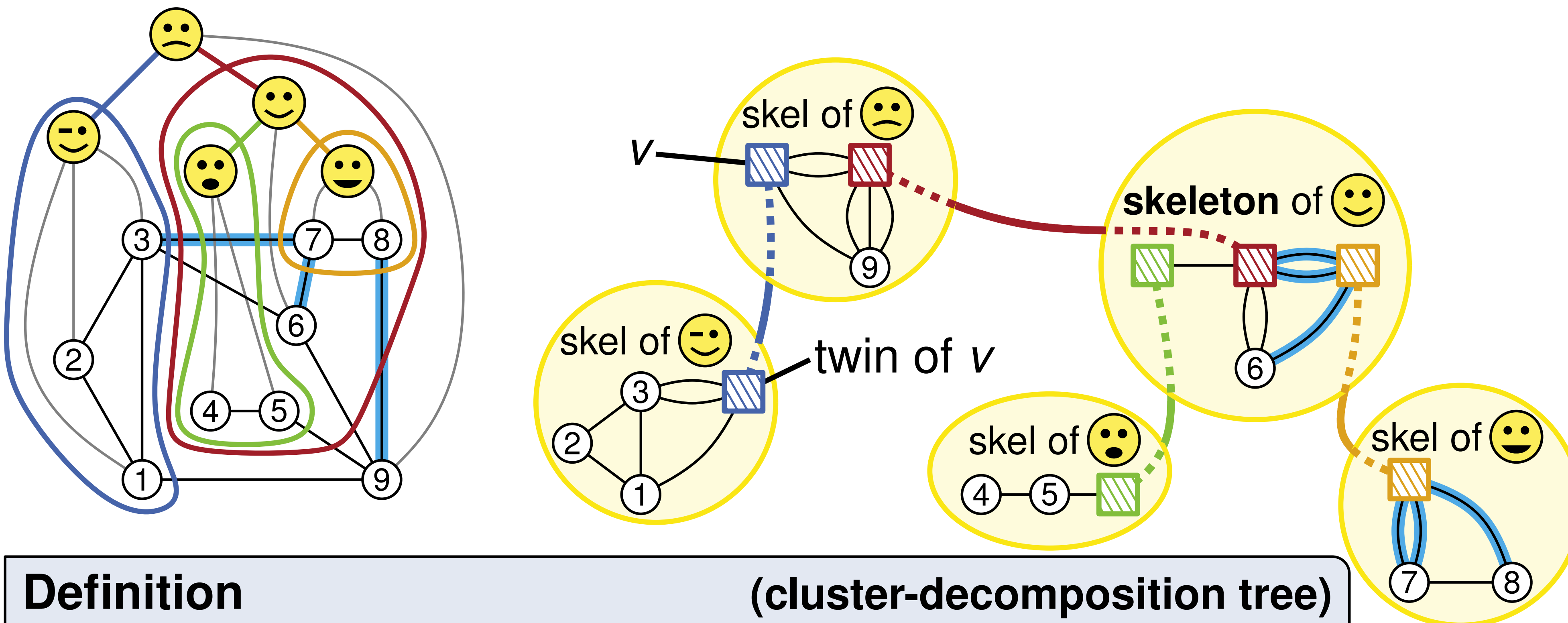


Definition **(cluster-decomposition tree)**
The **cd-tree** of a clustered graph (G, T) is T together with a bijection between T's nodes and their skeletons.

- The new vertices in skeletons (▧) are **virtual vertices**.
- Every virtual vertex has a twin.
- Twins have the same incident edges.

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

The CD-Tree – A (New) Perspective



Definition **(cluster-decomposition tree)**

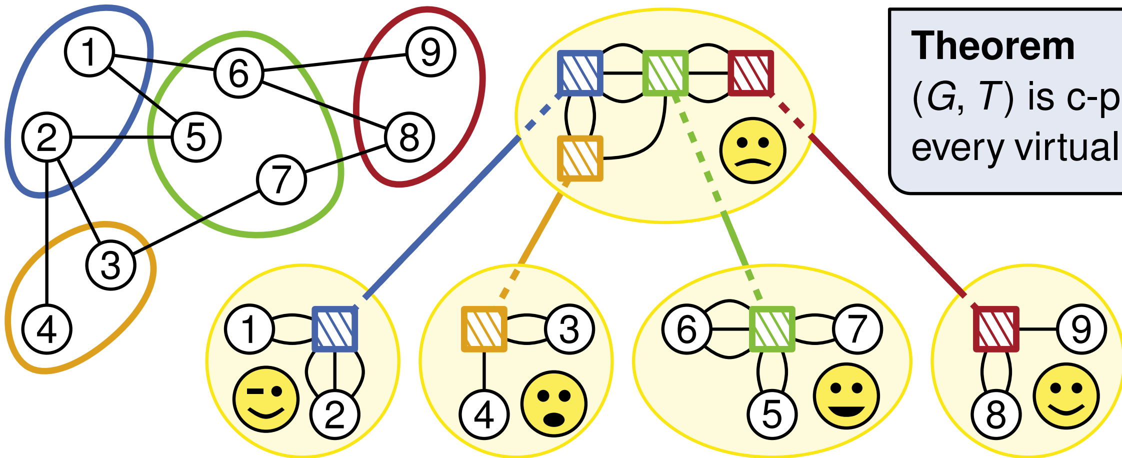
The **cd-tree** of a clustered graph (G, T) is T together with a bijection between T's nodes and their skeletons.

Theorem **(characterization)**

(G, T) is c-planar $\Leftrightarrow$ one can embed the skeletons such that every virtual vertex and its twin have the same edge-ordering.

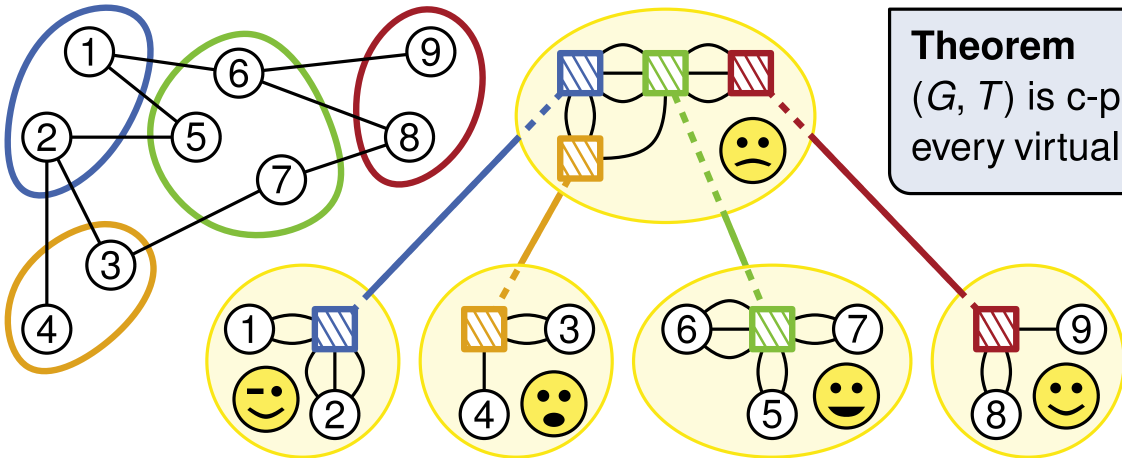
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – Isolated Vertices



Theorem **(characterization)**
(G, T) is c-planar $\Leftrightarrow$ one can embed the skeletons such that every virtual vertex and its twin have the same edge-ordering.

Flat-Clustered Graphs – Isolated Vertices



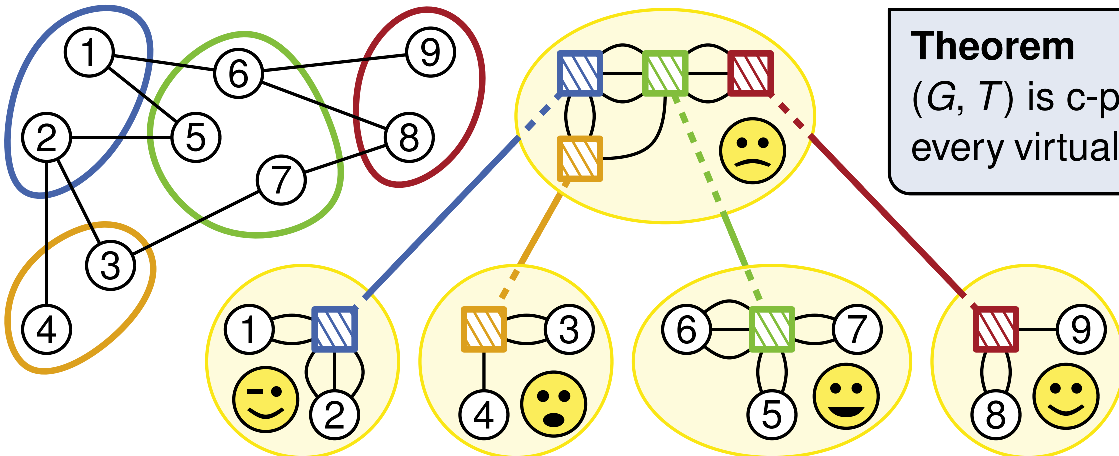
Theorem **(characterization)**
(G, T) is c-planar $\Leftrightarrow$ one can embed the skeletons such that every virtual vertex and its twin have the same edge-ordering.

Why is this instance special?

- flat hierarchy
- cluster = isolated vertices

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – Isolated Vertices



Theorem **(characterization)**
(G, T) is c-planar $\Leftrightarrow$ one can embed the skeletons such that every virtual vertex and its twin have the same edge-ordering.

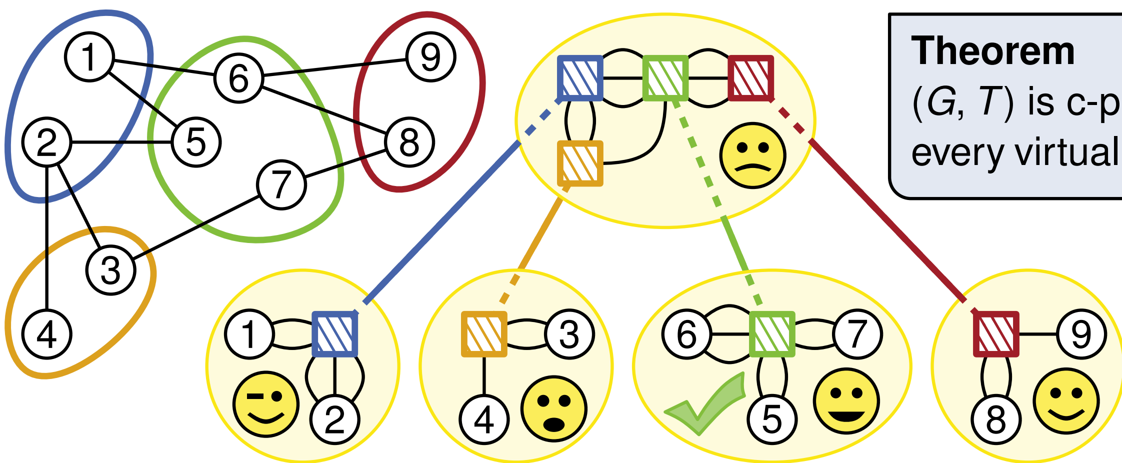
Why is this instance special?

- flat hierarchy
- cluster = isolated vertices

Which edge-orderings around does 's skeleton allow?

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – Isolated Vertices



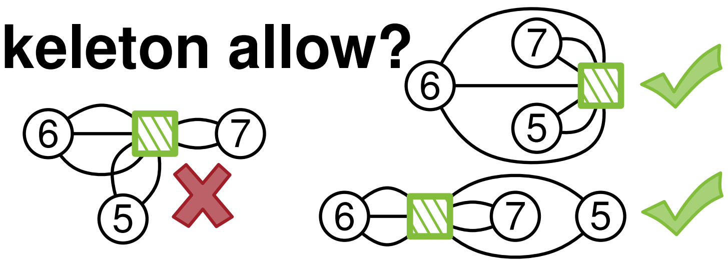
Theorem **(characterization)**
(G, T) is c-planar $\Leftrightarrow$ one can embed the skeletons such that every virtual vertex and its twin have the same edge-ordering.

Why is this instance special?

- flat hierarchy
- cluster = isolated vertices

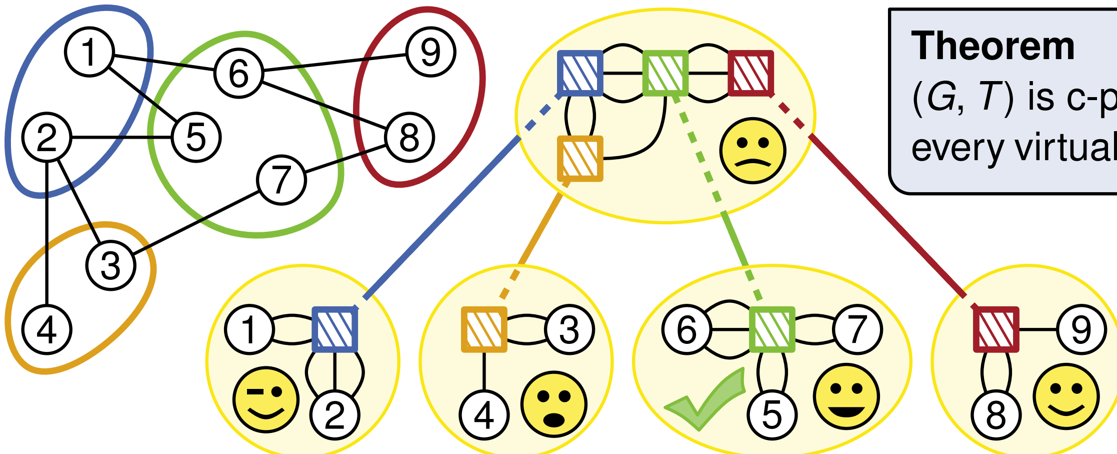
Which edge-orderings around ▧ does 😐's skeleton allow?

- vertices ⑤, ⑥, and ⑦ partition the edges
- partitions must not alternate



Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – Isolated Vertices



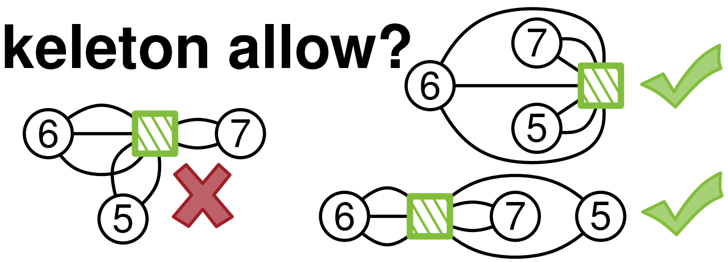
Theorem **(characterization)**
(G, T) is c-planar $\Leftrightarrow$ one can embed the skeletons such that every virtual vertex and its twin have the same edge-ordering.

Why is this instance special?

- flat hierarchy
- cluster = isolated vertices

Which edge-orderings around does 's skeleton allow?

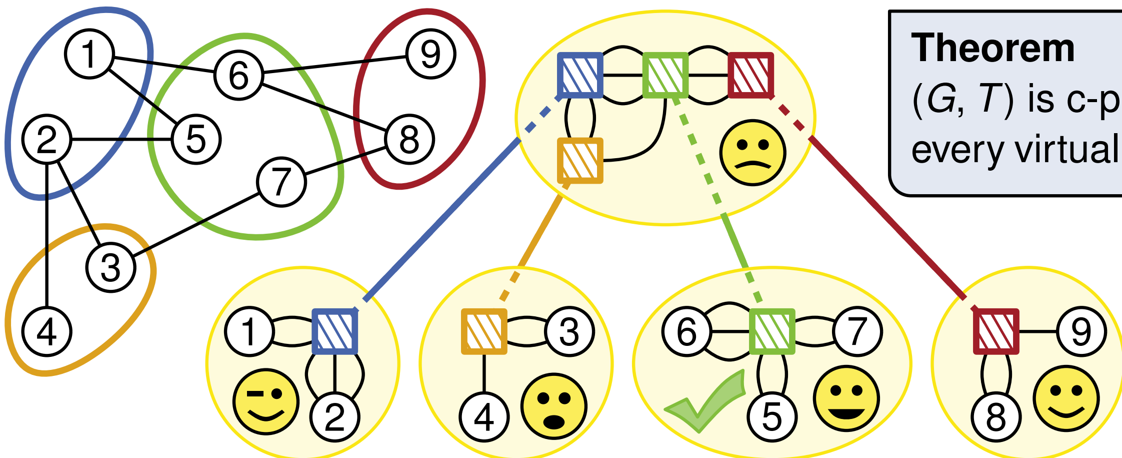
- vertices 5, 6, and 7 partition the edges
- partitions must not alternate



Definition **(partition-constraint)**
We call such a restriction of edge-orderings a **partition-constraint**.

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – Isolated Vertices



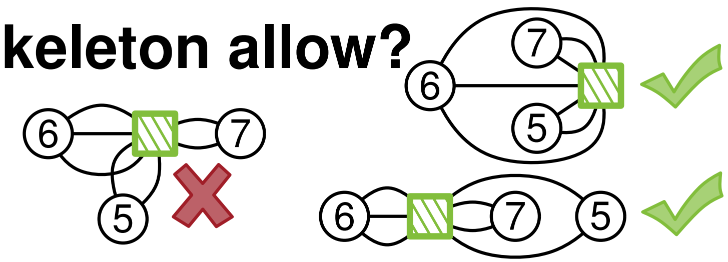
Theorem **(characterization)**
(G, T) is c-planar $\Leftrightarrow$ one can embed the skeletons such that every virtual vertex and its twin have the same edge-ordering.

Why is this instance special?

- flat hierarchy
- cluster = isolated vertices

Which edge-orderings around does 's skeleton allow?

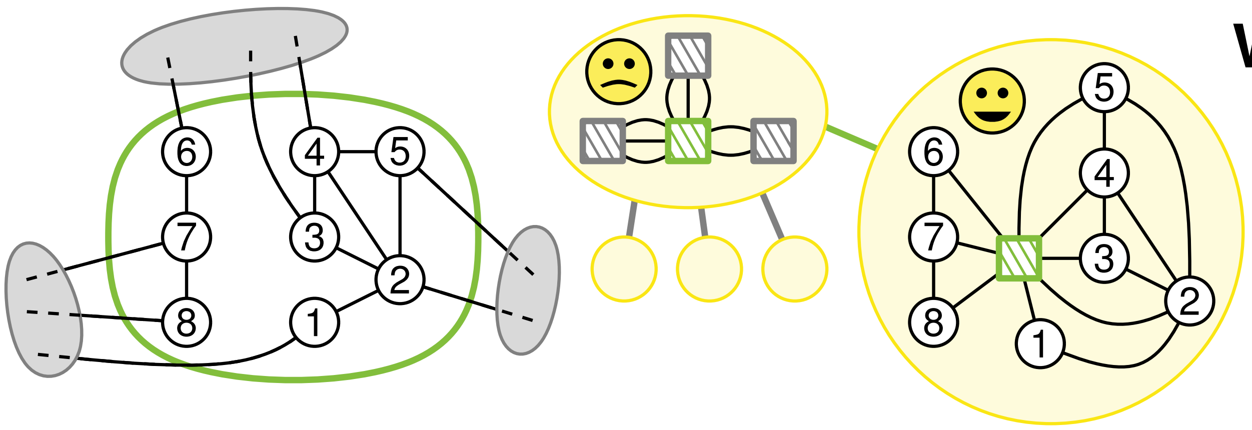
- vertices ⑤, ⑥, and ⑦ partition the edges
- partitions must not alternate



Definition **(partition-constraint)**
We call such a restriction of edge-orderings a **partition-constraint**.

Theorem
C-planarity for flat-clustered graphs where every cluster is a set of isolated vertices is equivalent to **planarity with partition-constraints**.

Flat-Clustered Graphs – Fixed Embedding

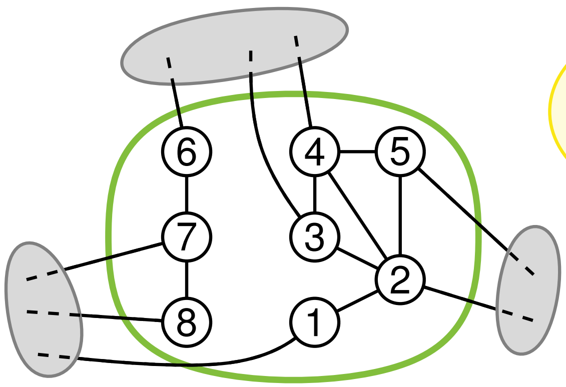


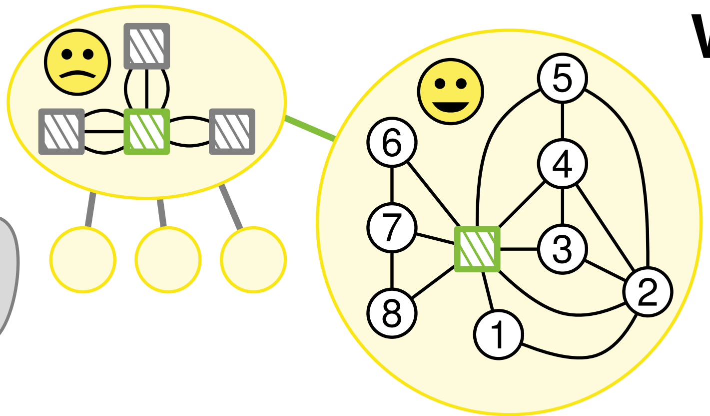
What is fixed?

- embedding of G

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – Fixed Embedding



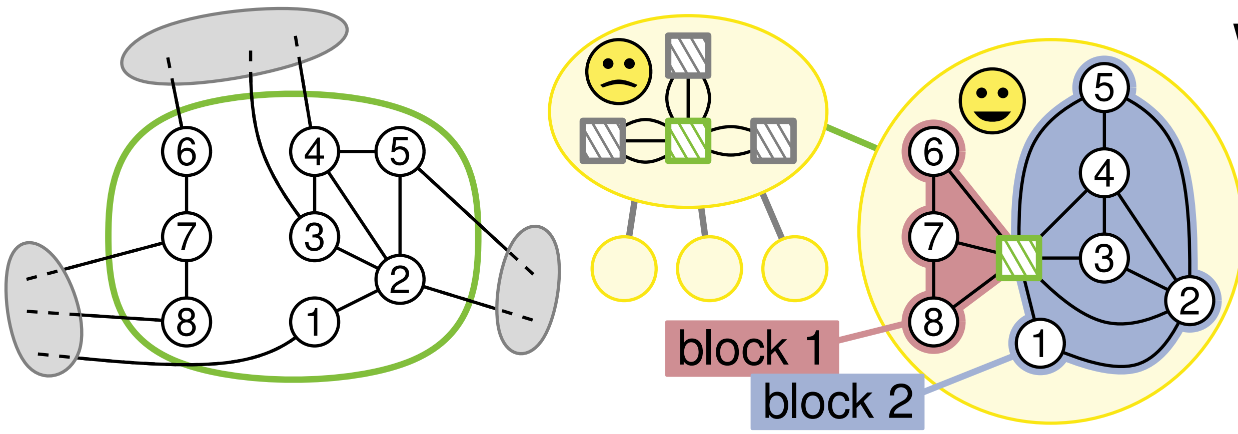


What is fixed?

- embedding of G
- edge-orderings of non-virtual vertices

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – Fixed Embedding

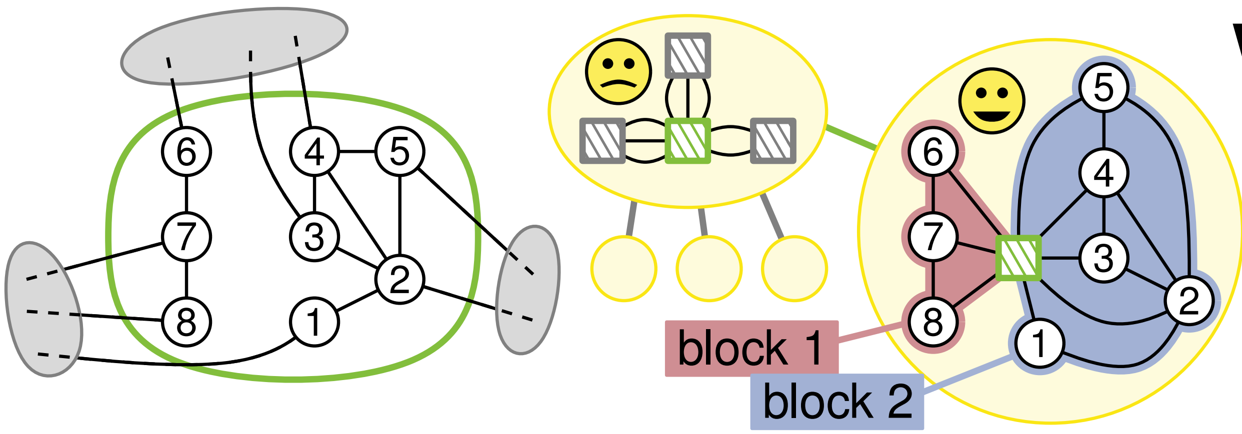


What is fixed?

- embedding of G
- edge-orderings of non-virtual vertices
- embeddings of blocks

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – Fixed Embedding



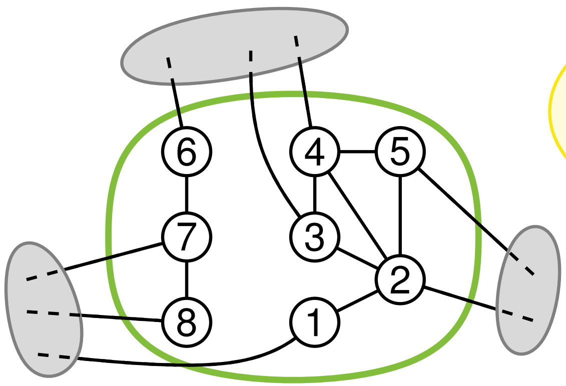
What is fixed?

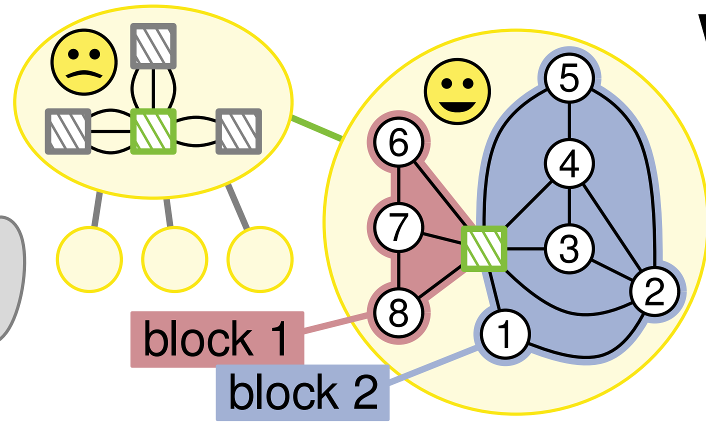
- embedding of G
- edge-orderings of non-virtual vertices
- embeddings of blocks

Which edge-orderings around  does 's skeleton allow?

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – Fixed Embedding





What is fixed?

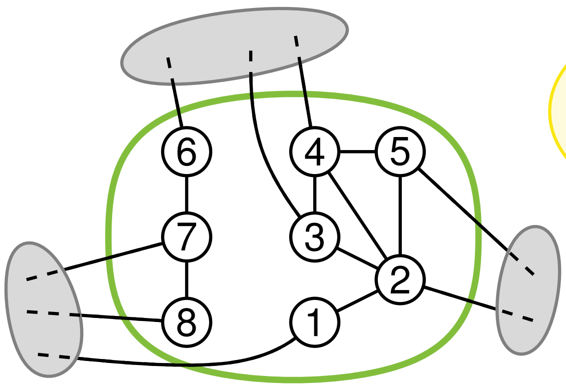
- embedding of G
- edge-orderings of non-virtual vertices
- embeddings of blocks

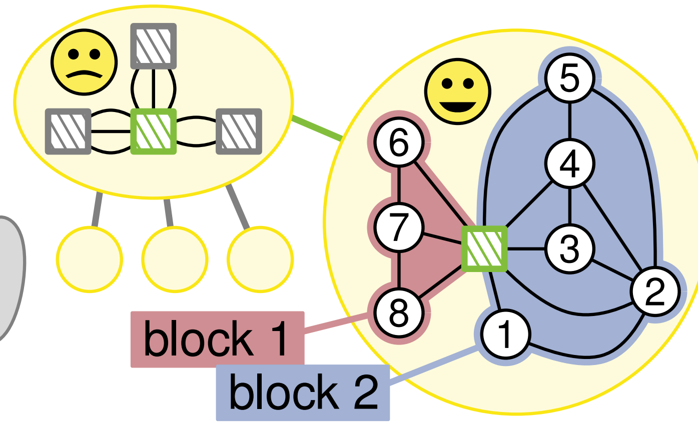
Which edge-orderings around does 's skeleton allow?

- blocks in 's skeleton partition the edges

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – Fixed Embedding





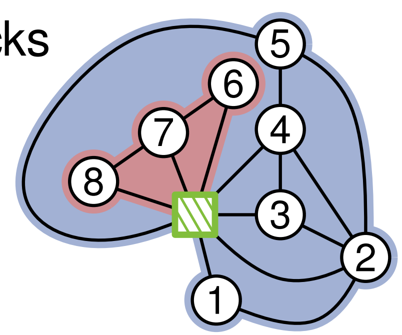
What is fixed?

- embedding of *G*
- edge-orderings of non-virtual vertices
- embeddings of blocks

Which edge-orderings around does 's skeleton allow?

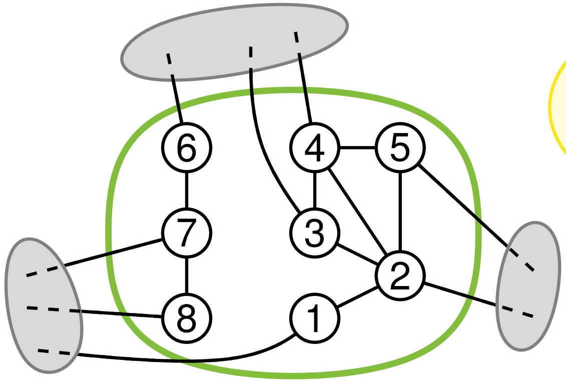
- blocks in 's skeleton partition the edges
- partitions must not alternate → **partition-constraint**

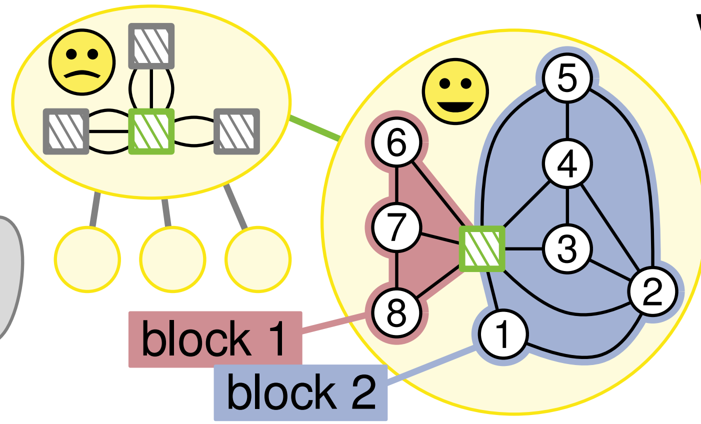
Nesting the blocks is possible:



Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – Fixed Embedding





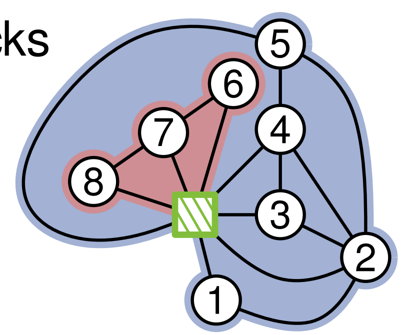
What is fixed?

- embedding of *G*
- edge-orderings of non-virtual vertices
- embeddings of blocks

Which edge-orderings around ▧ does 🙂's skeleton allow?

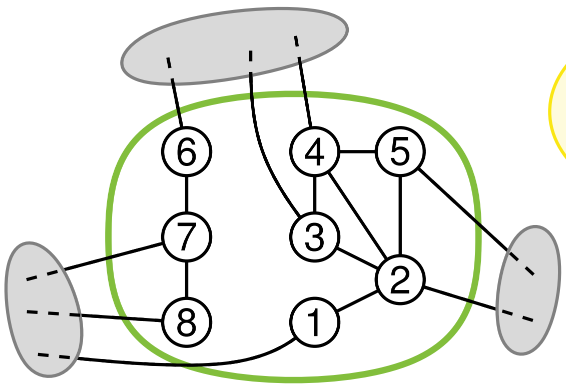
- blocks in 🙂's skeleton partition the edges
- partitions must not alternate → **partition-constraint**
- order in each partition is fixed → **full-constraint**

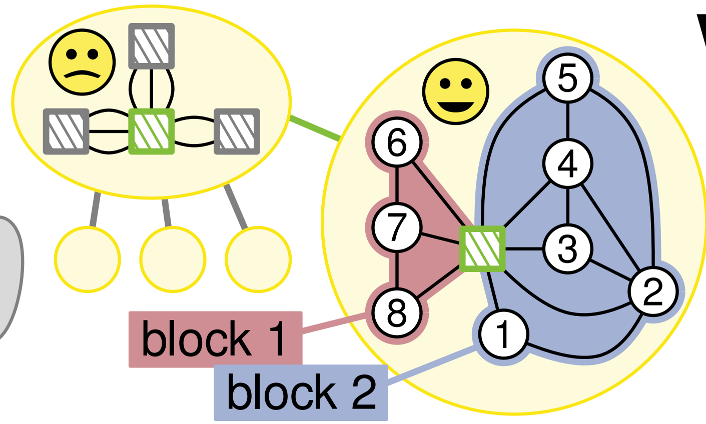
Nesting the blocks is possible:



Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – Fixed Embedding





What is fixed?

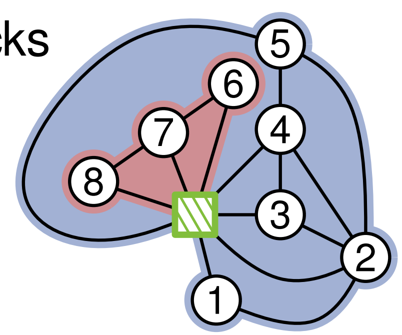
- embedding of *G*
- edge-orderings of non-virtual vertices
- embeddings of blocks

Which edge-orderings around does 's skeleton allow?

- blocks in 's skeleton partition the edges
- partitions must not alternate → **partition-constraint**
- order in each partition is fixed → **full-constraint**

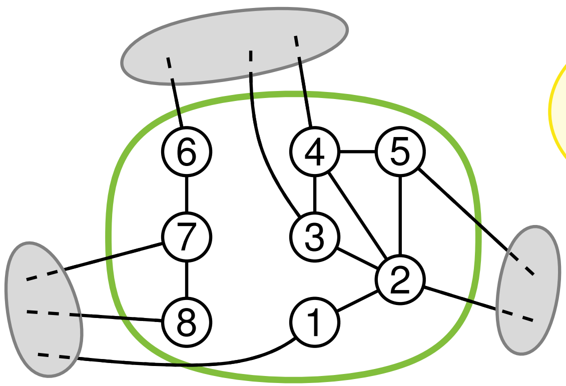
partitioned full-constraint

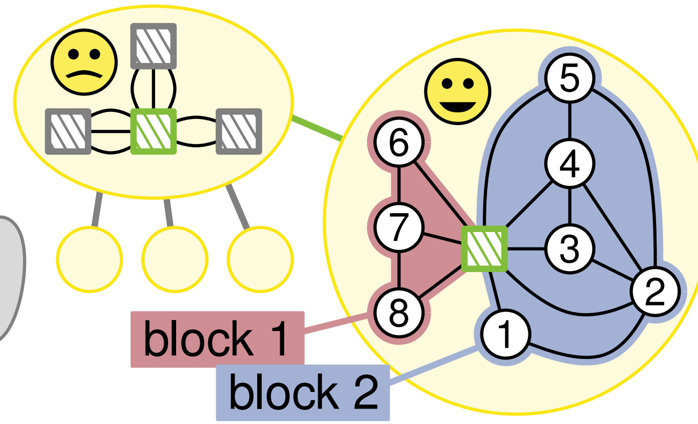
Nesting the blocks is possible:



Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – Fixed Embedding





What is fixed?

- embedding of G
- edge-orderings of non-virtual vertices
- embeddings of blocks

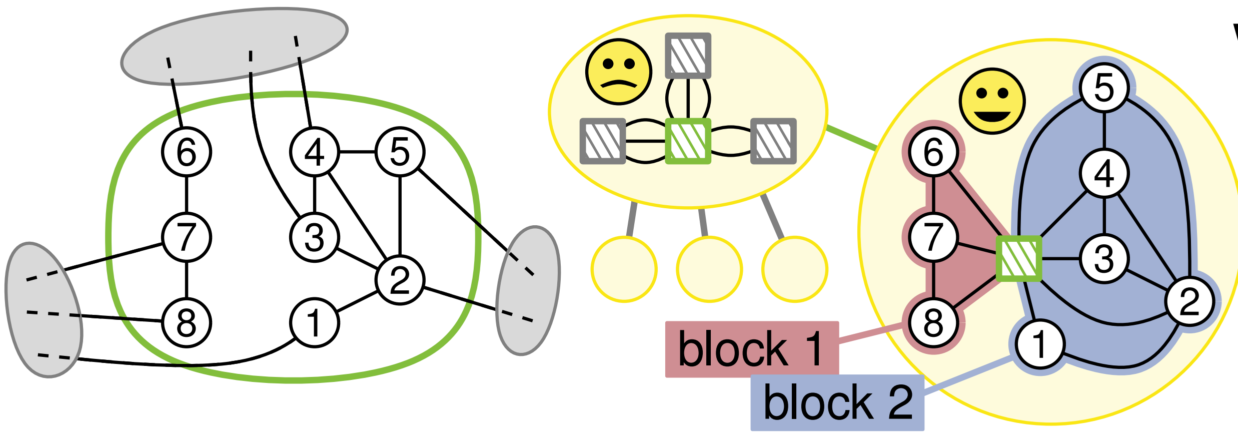
Which edge-orderings around does 's skeleton allow?

- blocks in 's skeleton partition the edges
- partitions must not alternate → **partition-constraint**
- order in each partition is fixed → **full-constraint**

partitioned full-constraint

Theorem
C-planarity for flat-clustered graphs with fixed planar embedding is equivalent to **planarity with partitioned full-constraints**.

Flat-Clustered Graphs – ~~Fixed Embedding~~



What is fixed?

- ~~embedding of *G*~~
- ~~edge-orderings of non-virtual vertices~~
- ~~embeddings of blocks~~

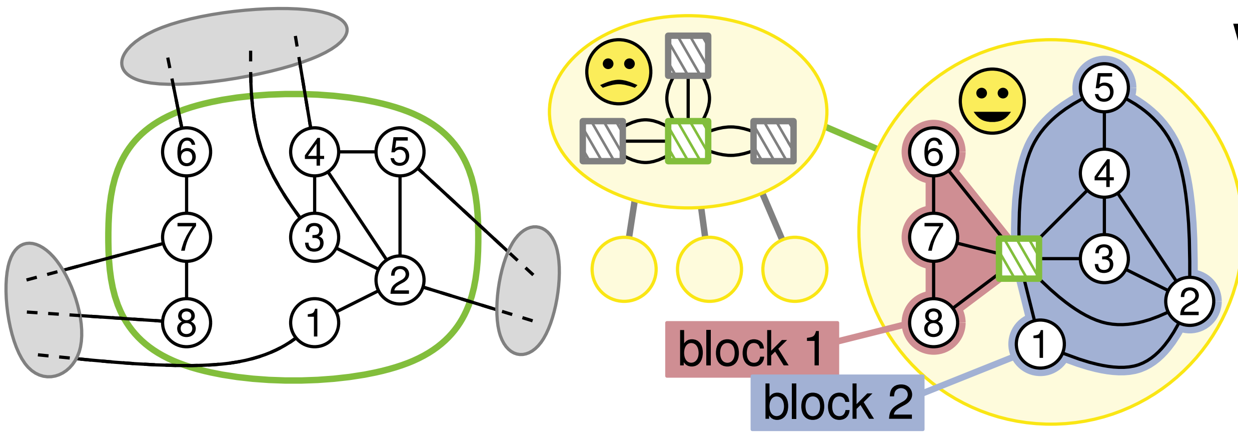
Which edge-orderings around does 's skeleton allow?

- blocks in 's skeleton partition the edges
- partitions must not alternate → **partition-constraint**
- order in each partition **??** → **??-constraint**

partitioned ??-constraint

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – ~~Fixed Embedding~~



What is fixed?

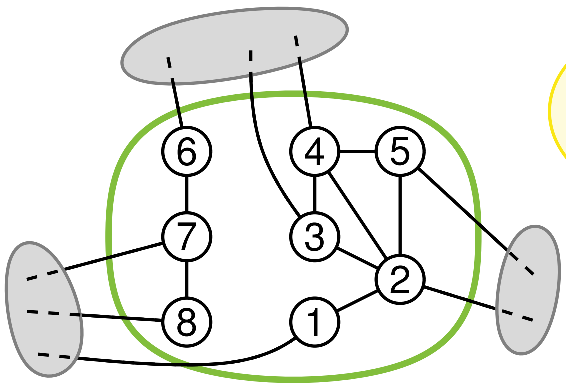
- ~~embedding of *G*~~
- ~~edge-orderings of non-virtual vertices~~
- ~~embeddings of blocks~~

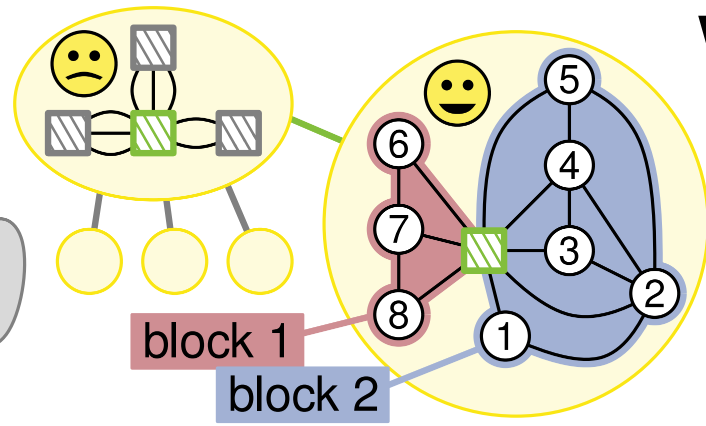
Which edge-orderings around ▧ does 🙂's skeleton allow?

- blocks in 🙂's skeleton partition the edges
- partitions must not alternate → **partition-constraint**
- order in each partition **is restricted by a PQ-tree** → **PQ-constraint**

partitioned PQ-constraint

Flat-Clustered Graphs – ~~Fixed Embedding~~





What is fixed?

- ~~embedding of *G*~~
- ~~edge-orderings of non-virtual vertices~~
- ~~embeddings of blocks~~

Which edge-orderings around ▧ does 🙂's skeleton allow?

- blocks in 🙂's skeleton partition the edges
- partitions must not alternate → **partition-constraint**
- order in each partition **is restricted by a PQ-tree** → **PQ-constraint**

partitioned PQ-constraint

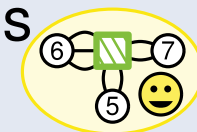
> **Theorem**
> C-planarity for flat-clustered graphs is equivalent to **planarity with partitioned PQ-constraints**.

Conclusion

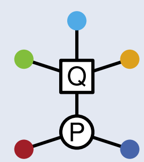
Theorem **(flat-clustered graphs)**

These variants of c-planarity and constrained embedding are equivalent:

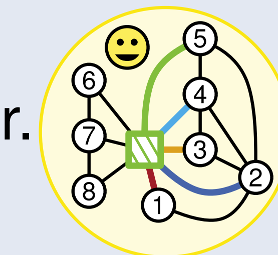
- flat, isolated vertices
  ⟷ partition-constr.



- flat
  ⟷ partitioned PQ-constr.



- flat, fixed embedding
  ⟷ partitioned full-constr.



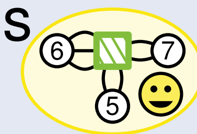
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Conclusion

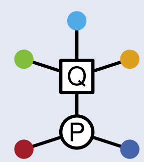
Theorem **(flat-clustered graphs)**

These variants of c-planarity and constrained embedding are equivalent:

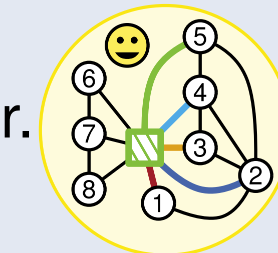
- flat, isolated vertices ⟷ partition-constr.



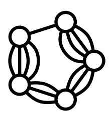
- flat ⟷ partitioned PQ-constr.



- flat, fixed embedding ⟷ partitioned full-constr.



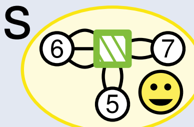
solved variants

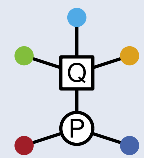
graph class	constraints	
multi-cycle 	partition constraints partitions of size 2 	[Cortese, Di Battista, Patrignani, Pizzonia 2005]

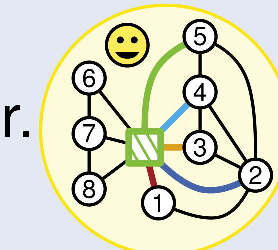
Conclusion

Theorem **(flat-clustered graphs)**

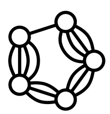
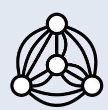
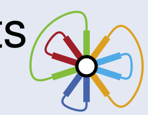
These variants of c-planarity and constrained embedding are equivalent:

- flat, isolated vertices ⟷ partition-constr. 

- flat ⟷ partitioned PQ-constr. 

- flat, fixed embedding ⟷ partitioned full-constr. 

solved variants

graph class	constraints	
multi-cycle 	partition constraints partitions of size 2 	**[Cortese, Di Battista, Patrignani, Pizzonia** 2005]
fixed embedding (up to reordering multi-edges) 	partition constraints partitions of size 2 	**[Cortese, Di Battista, Patrignani, Pizzonia** 2009]

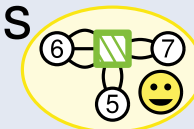
Conclusion

Karlsruhe Institute of Technology

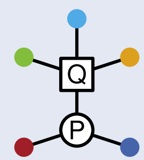
Theorem **(flat-clustered graphs)**

These variants of c-planarity and constrained embedding are equivalent:

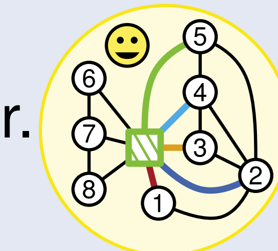
- flat, isolated vertices ⟷ partition-constr.



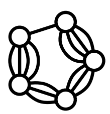
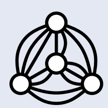
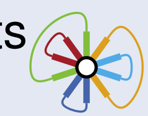
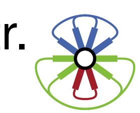
- flat ⟷ partitioned PQ-constr.



- flat, fixed embedding ⟷ partitioned full-constr.



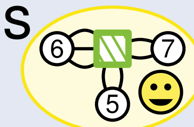
solved variants

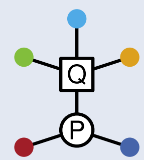
graph class	**constraints**	
multi-cycle 	partition constraints partitions of size 2 	[Cortese, Di Battista, Patrignani, Pizzonia 2005]
fixed embedding (up to reordering multi-edges) 	partition constraints partitions of size 2 	[Cortese, Di Battista, Patrignani, Pizzonia 2009]
complicated restriction (not very strong)	partitioned-full constr. at most 3 partitions 	[Jelínková, Kára, Kratochvíl, Pergel, Suchý, Vyskočil 2009]

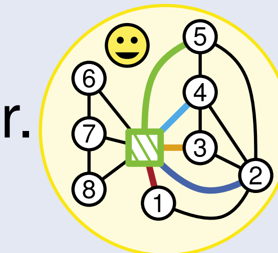
Conclusion

Theorem **(flat-clustered graphs)**

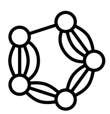
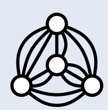
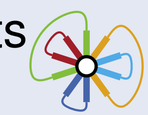
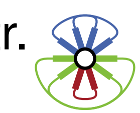
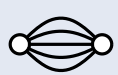
These variants of c-planarity and constrained embedding are equivalent:

- flat, isolated vertices ⟷ partition-constr. 

- flat ⟷ partitioned PQ-constr. 

- flat, fixed embedding ⟷ partitioned full-constr. 

solved variants

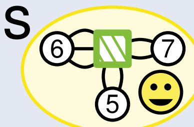
graph class	constraints	
multi-cycle 	partition constraints partitions of size 2 	**[Cortese, Di Battista, Patrignani, Pizzonia** 2005]
fixed embedding (up to reordering multi-edges) 	partition constraints partitions of size 2 	**[Cortese, Di Battista, Patrignani, Pizzonia** 2009]
complicated restriction (not very strong)	partitioned-full constr. at most 3 partitions 	**[Jelínková, Kára, Kratochvíl, Pergel, Suchý, Vyskočil** 2009]
only two vertices 	partitioned PQ-constraints	**[Biedl, Kaufmann, Mutzel** 1998] **[Hong, Nagamochi** 2014]

Conclusion

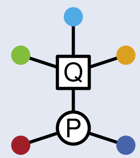
Theorem **(flat-clustered graphs)**

These variants of c-planarity and constrained embedding are equivalent:

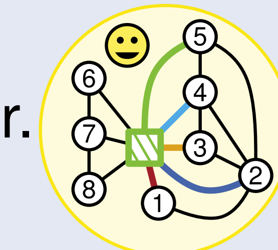
- flat, isolated vertices ⟷ partition-constr.



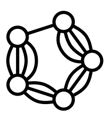
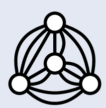
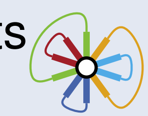
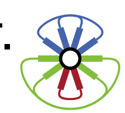
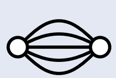
- flat ⟷ partitioned PQ-constr.



- flat, fixed embedding ⟷ partitioned full-constr.



solved variants

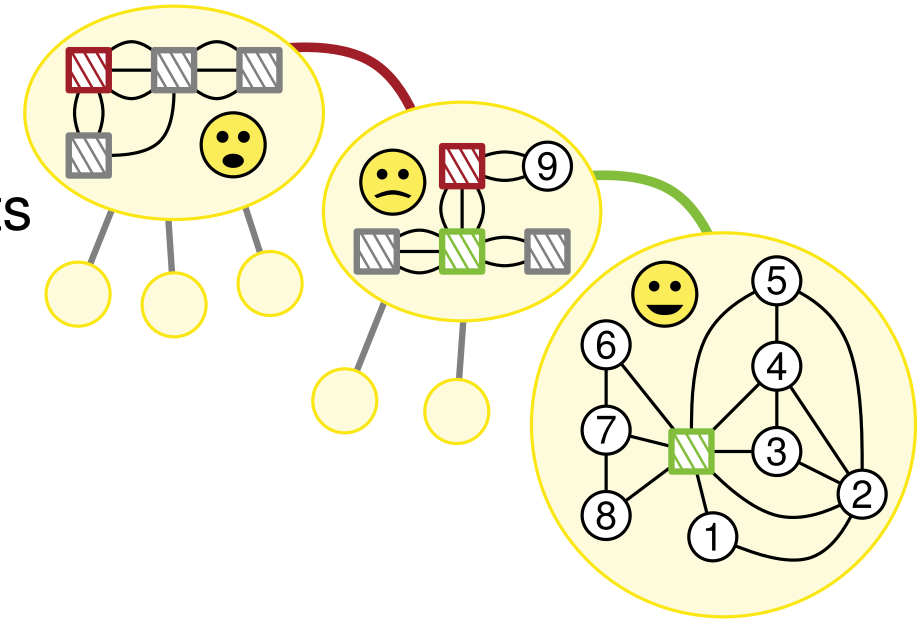
graph class	**constraints**	
multi-cycle 	partition constraints partitions of size 2 	**[Cortese, Di Battista, Patrignani, Pizzonia** 2005]
fixed embedding (up to reordering multi-edges) 	partition constraints partitions of size 2 	**[Cortese, Di Battista, Patrignani, Pizzonia** 2009]
complicated restriction (not very strong)	partitioned-full constr. at most 3 partitions 	**[Jelínková, Kára, Kratochvíl, Pergel, Suchý, Vyskočil** 2009]
only two vertices 	partitioned PQ-constraints	**[Biedl, Kaufmann, Mutzel** 1998] **[Hong, Nagamochi** 2014]

open problem: extend this table

Final Remarks

Things become more complicated, when the clustering is not flat.

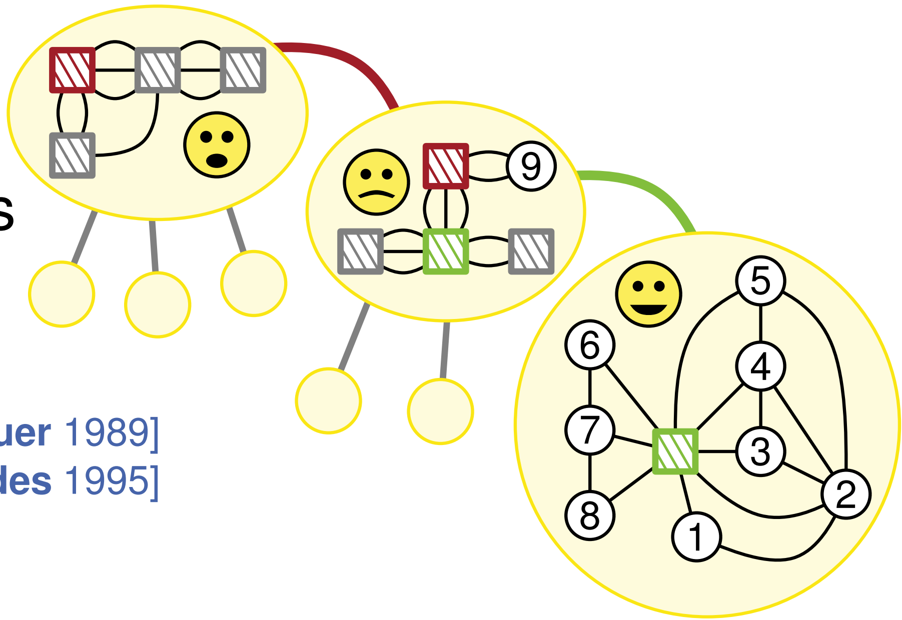
- lowest level: same constraints as in the flat-clustered case
- higher levels: complicated constraints



Final Remarks

Things become more complicated, when the clustering is not flat.

- lowest level: same constraints as in the flat-clustered case
- higher levels: complicated constraints

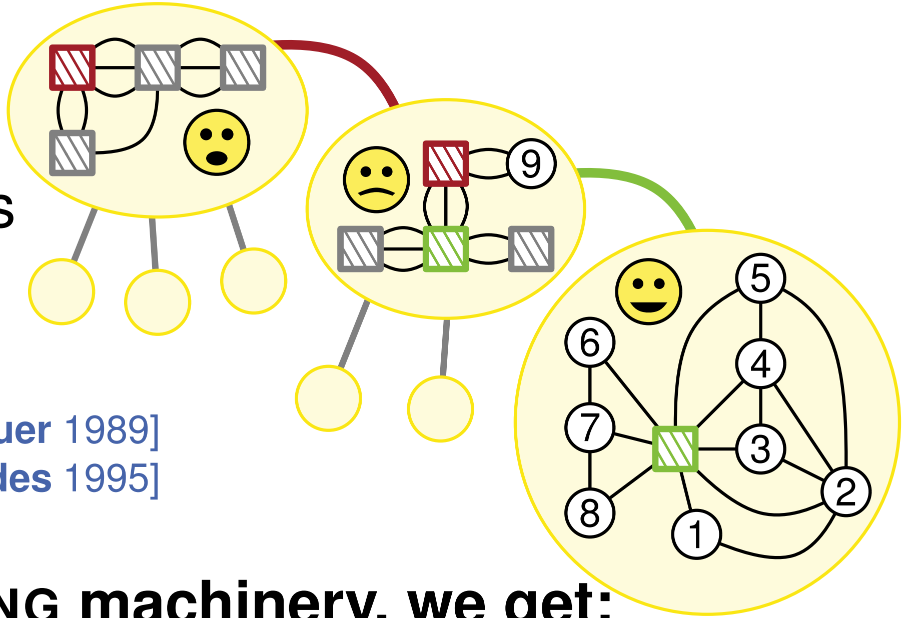


exception: every cluster is connected
$\Rightarrow$ PQ-constraints on every level [**Lengauer** 1989] [**Feng, Cohen, Eades** 1995]

Final Remarks

Things become more complicated, when the clustering is not flat.

- lowest level: same constraints as in the flat-clustered case
- higher levels: complicated constraints



exception: every cluster is connected
$\Rightarrow$ PQ-constraints on every level [**Lengauer** 1989]
[**Feng, Cohen, Eades** 1995]

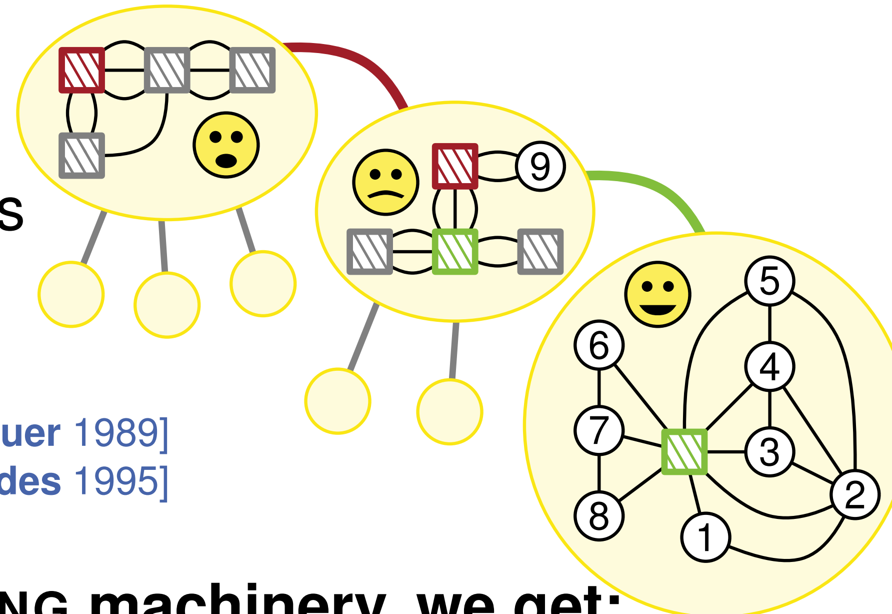
Using the SIMULTANEOUS PQ-ORDERING machinery, we get:

> **Theorem**
> C-planarity can be solved efficiently if each virtual vertex in a skeleton of the cd-tree is incident to at most two non-trivial blocks.

Final Remarks

Things become more complicated, when the clustering is not flat.

- lowest level: same constraints as in the flat-clustered case
- higher levels: complicated constraints



exception: every cluster is connected
$\Rightarrow$ PQ-constraints on every level [**Lengauer** 1989]
[**Feng, Cohen, Eades** 1995]

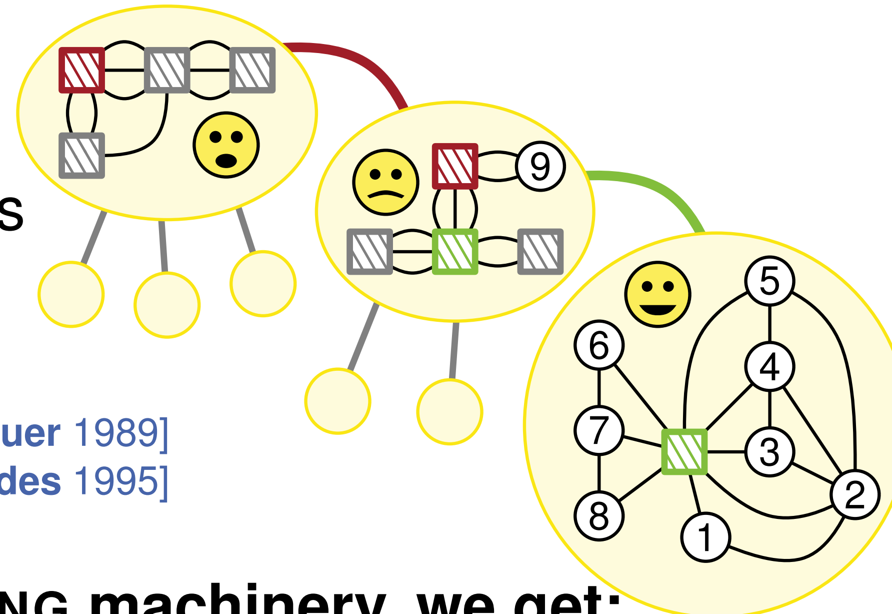
Using the SIMULTANEOUS PQ-ORDERING machinery, we get:

> **Theorem**
> C-planarity can be solved efficiently if each virtual vertex in a skeleton of the cd-tree is incident to at most two non-trivial blocks. This includes:
> - clusters and co-clusters have at most 2 connected components
> - clusters have at most 5 outgoing edges

Final Remarks

Things become more complicated, when the clustering is not flat.

- lowest level: same constraints as in the flat-clustered case
- higher levels: complicated constraints



exception: every cluster is connected
$\Rightarrow$ PQ-constraints on every level [**Lengauer** 1989]
[**Feng, Cohen, Eades** 1995]

Using the SIMULTANEOUS PQ-ORDERING machinery, we get:

Theorem
C-planarity can be solved efficiently if each virtual vertex in a skeleton of the cd-tree is incident to at most two non-trivial blocks. This includes:

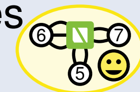
- clusters and co-clusters have at most 2 connected components
- clusters have at most 5 outgoing edges

formerly known for 4 instead of 5
[**Jelínek, Suchý, Tesař, Vyskočil** 2009]

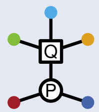
Conclusion

Theorem **(flat-clustered graphs)**
These variants of c-planarity and constrained embedding are equivalent:

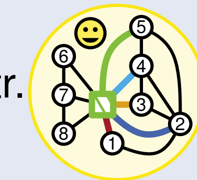
- flat, isolated vertices ⟷ partition-constr.



- flat ⟷ partitioned PQ-constr.



- flat, fixed embedding ⟷ partitioned full-constr.



solved variants

graph class	**constraints**	
multi-cycle	partition constraints partitions of size 2	[**Cortese, Di Battista, Patrignani, Pizzonia** 2005]
fixed embedding (up to reordering multi-edges)	partition constraints partitions of size 2	[**Cortese, Di Battista, Patrignani, Pizzonia** 2009]
complicated restriction (not very strong)	partitioned-full constr. at most 3 partitions	[Jelínková, Kára, Kratochvíl, Pe…
only two vertices	partitioned PQ-constraints	[Bie…

open problem: extend this table

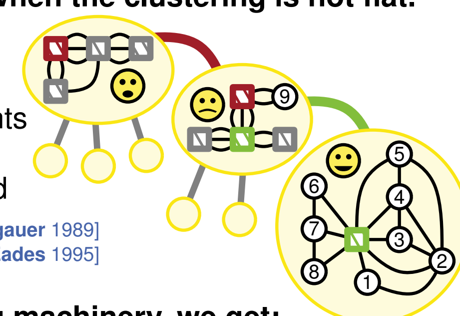
Thank you!

Questions?

Final Remarks

Things become more complicated, when the clustering is not flat.

- lowest level: same constraints as in the flat-clustered case
- higher levels: complicated constraints



exception: every cluster is connected
⇒ PQ-constraints on every level [**Lengauer** 1989] [**Feng, Cohen, Eades** 1995]

Using the simultaneous PQ-ordering machinery, we get:

Theorem
C-planarity can be solved efficiently if every virtual in the skeletons of the cd-tree is incident to at most two non-trivial blocks. This includes:

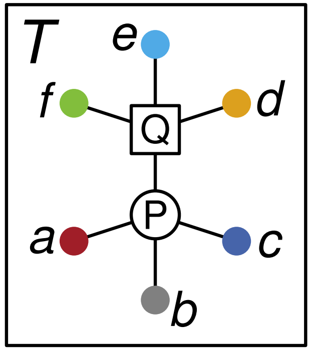
- clusters and co-clusters have at most 2 connected components
- clusters have at most 5 outgoing edges

formerly known for 4 instead of 5 [**Jelínek, Suchý, Tesař, Vyskočil** 2009]

PQ-Trees

Every inner node in a PQ-tree is either a P-node or a Q-node.

- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal

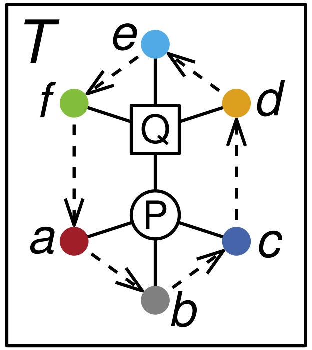


PQ-Trees

Every inner node in a PQ-tree is either a P-node or a Q-node.

- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal

Every embedding induces a (cyclic) ordering on the leaves.

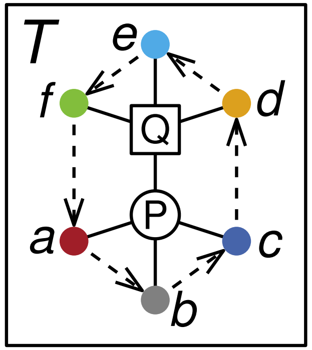


Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

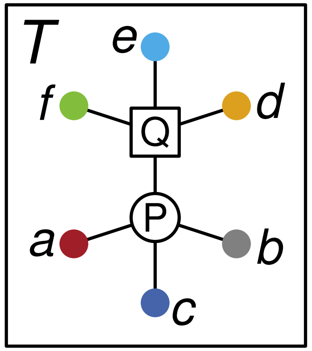
PQ-Trees

Every inner node in a PQ-tree is either a P-node or a Q-node.

- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal



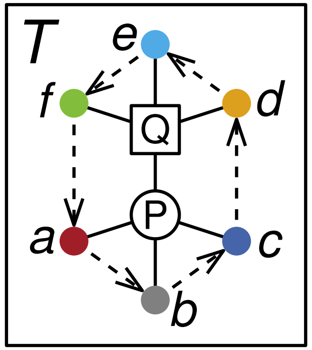
Every embedding induces a (cyclic) ordering on the leaves.



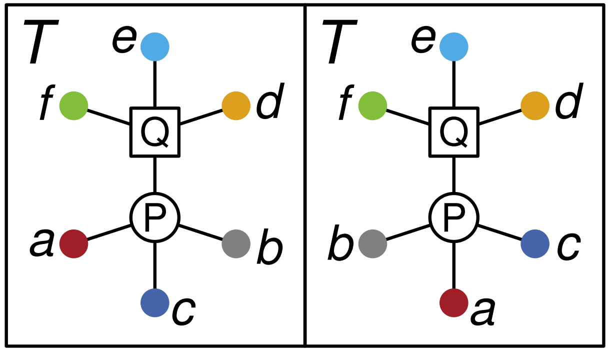
PQ-Trees

Every inner node in a PQ-tree is either a P-node or a Q-node.

- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal



Every embedding induces a (cyclic) ordering on the leaves.

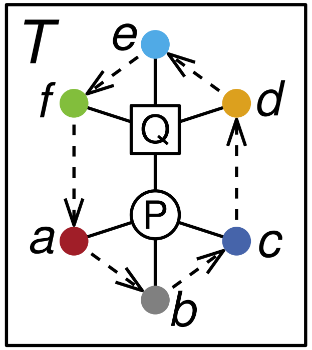


Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

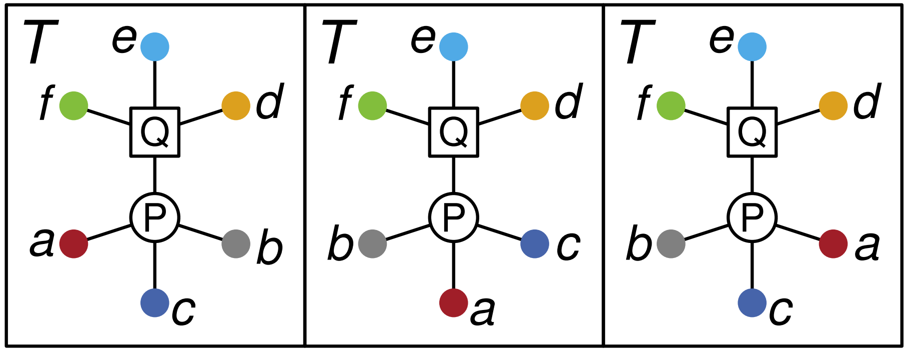
PQ-Trees

Every inner node in a PQ-tree is either a P-node or a Q-node.

- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal



Every embedding induces a (cyclic) ordering on the leaves.

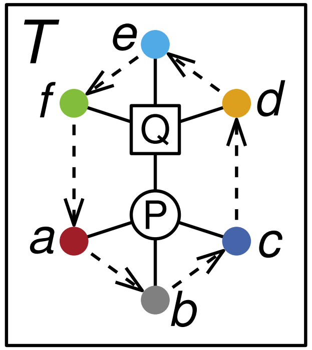


Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

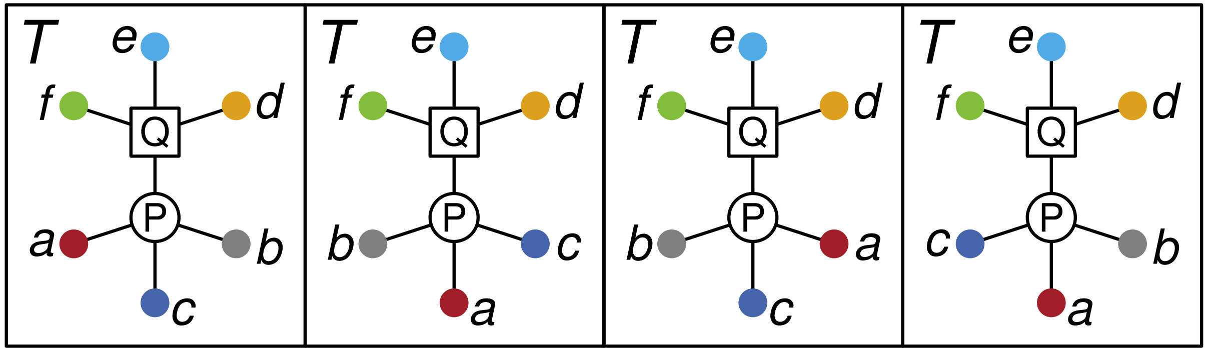
PQ-Trees

Every inner node in a PQ-tree is either a P-node or a Q-node.

- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal

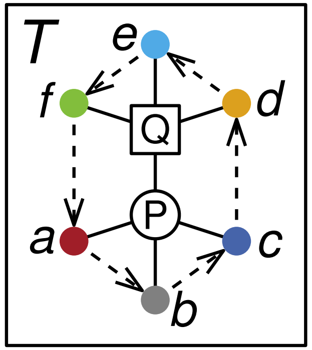


Every embedding induces a (cyclic) ordering on the leaves.



Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

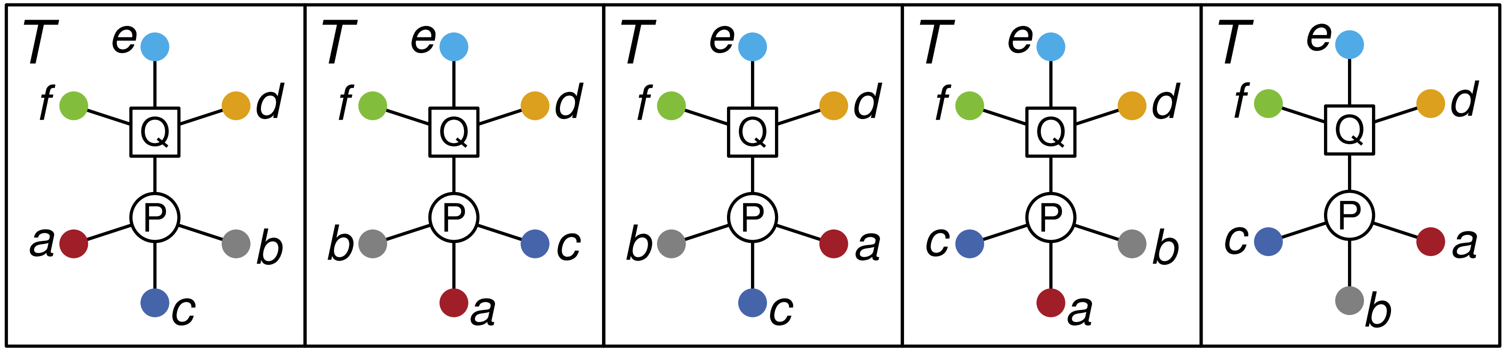
PQ-Trees



Every inner node in a PQ-tree is either a P-node or a Q-node.

- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal

Every embedding induces a (cyclic) ordering on the leaves.

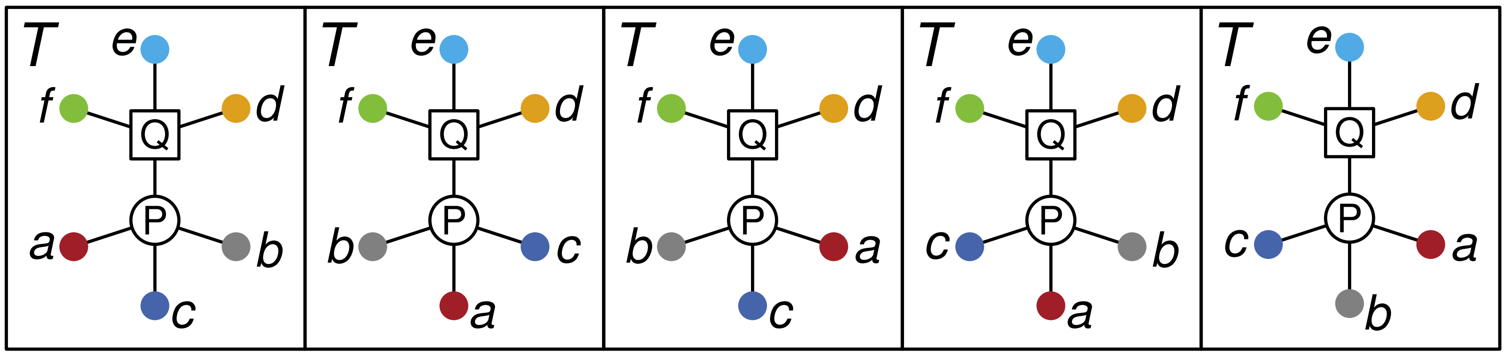


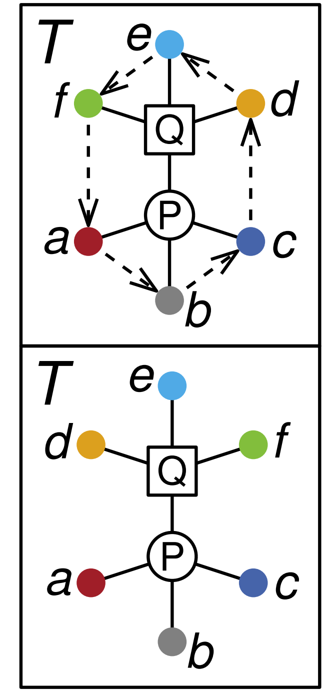
PQ-Trees

Every inner node in a PQ-tree is either a P-node or a Q-node.

- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal

Every embedding induces a (cyclic) ordering on the leaves.



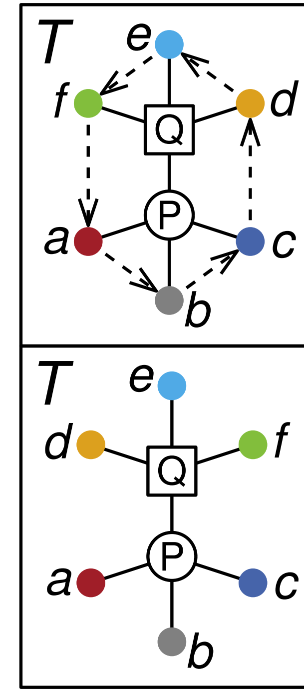


Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

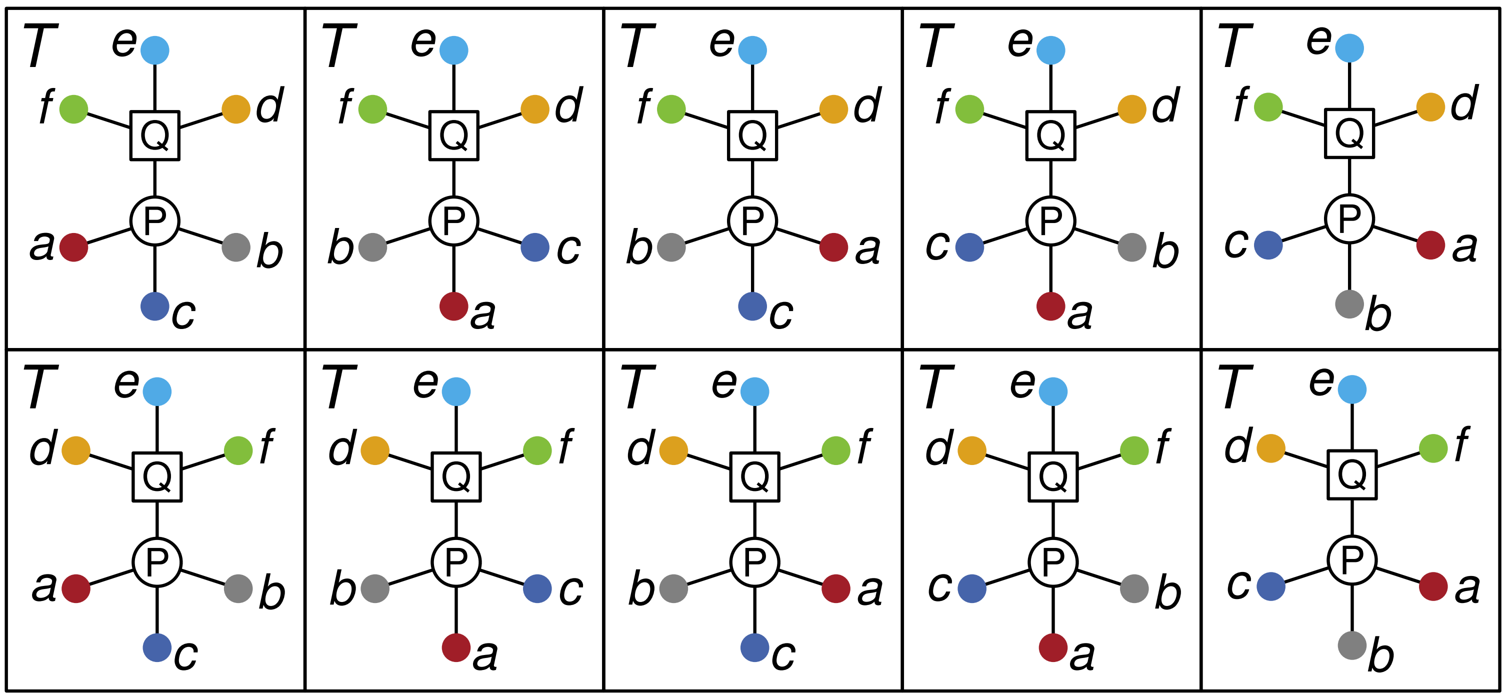
PQ-Trees

Every inner node in a PQ-tree is either a P-node or a Q-node.

- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal



Every embedding induces a (cyclic) ordering on the leaves.

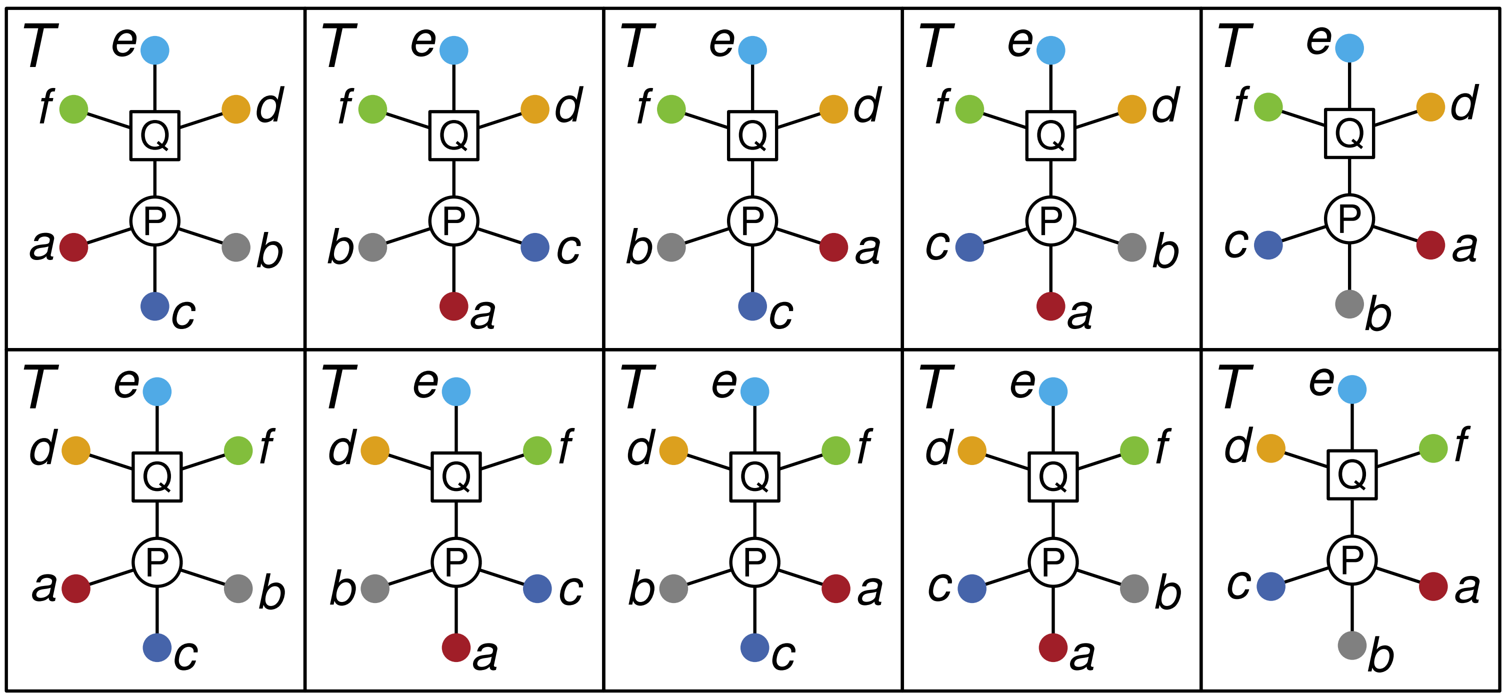


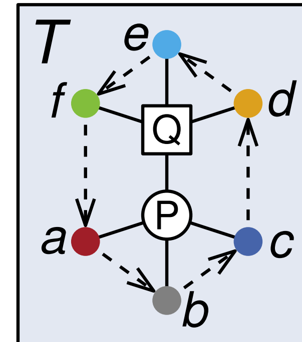
PQ-Trees

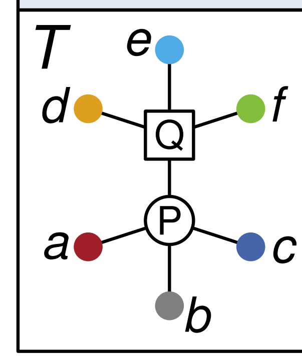
Every inner node in a PQ-tree is either a P-node or a Q-node.

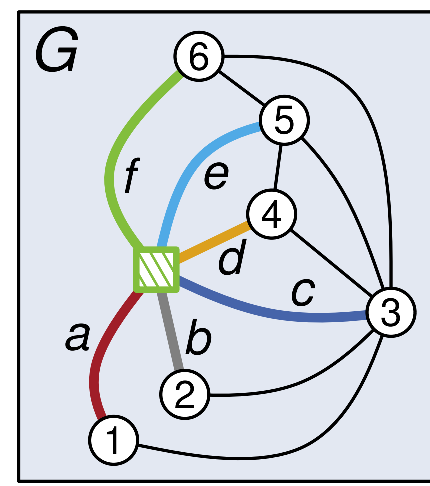
- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal

Every embedding induces a (cyclic) ordering on the leaves.







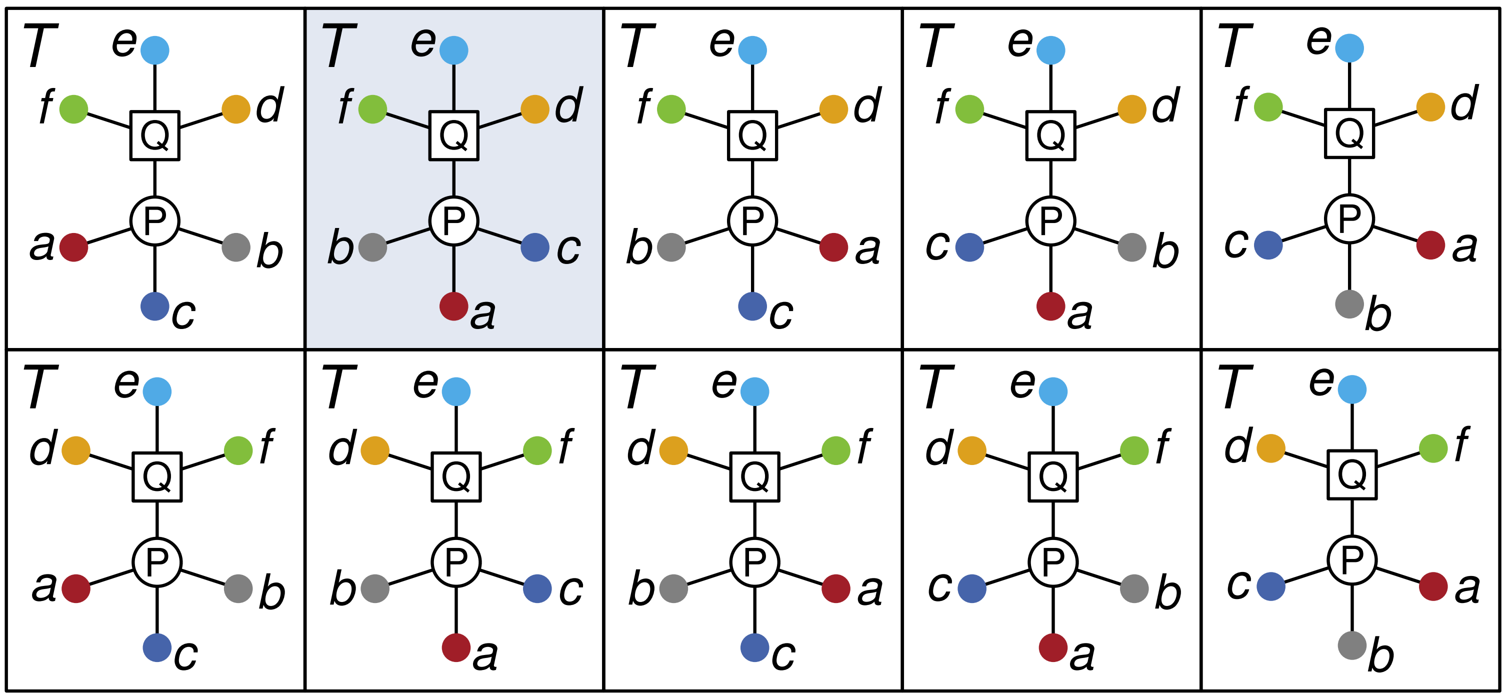


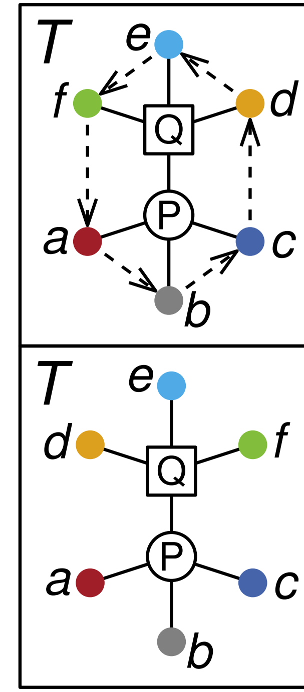
PQ-Trees

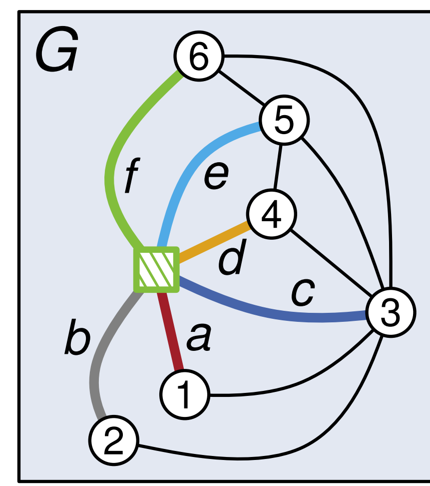
Every inner node in a PQ-tree is either a P-node or a Q-node.

- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal

Every embedding induces a (cyclic) ordering on the leaves.







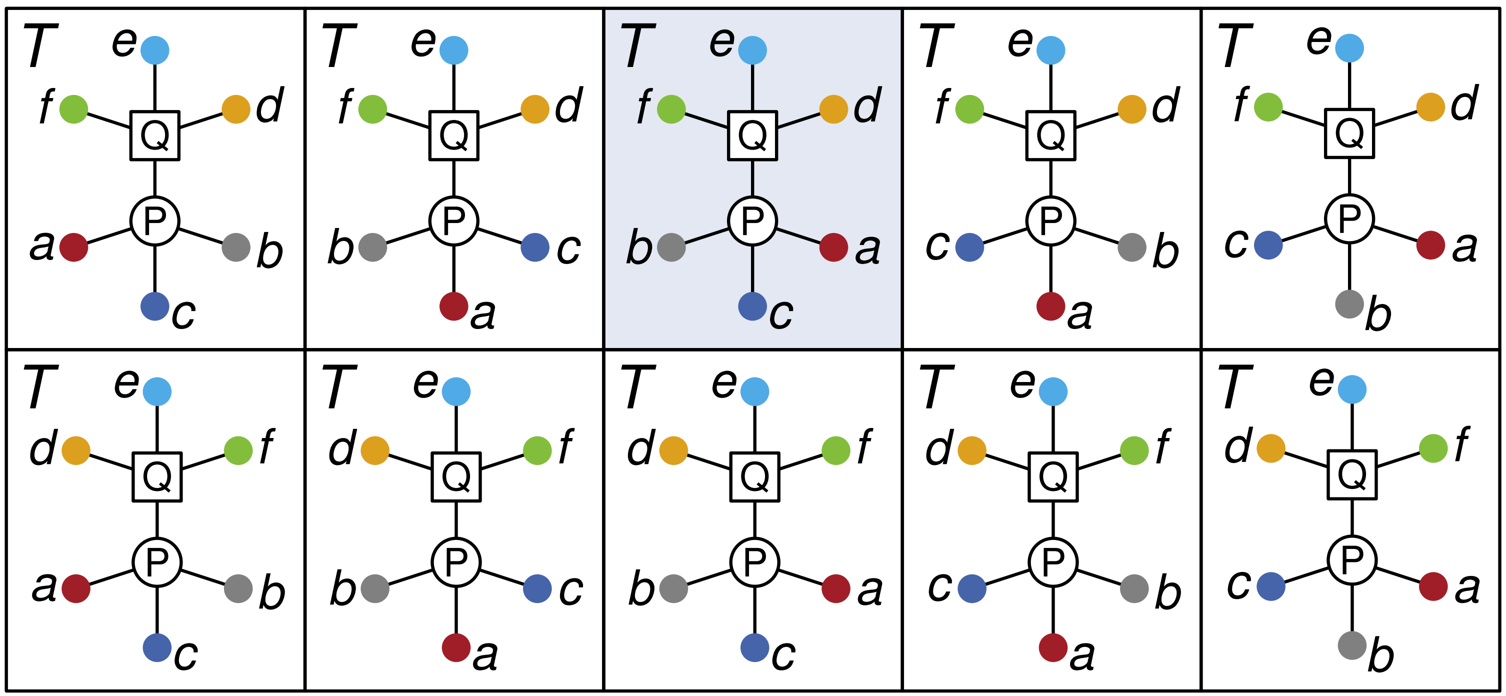
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

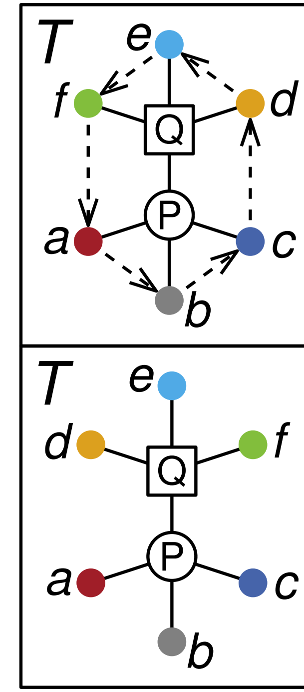
PQ-Trees

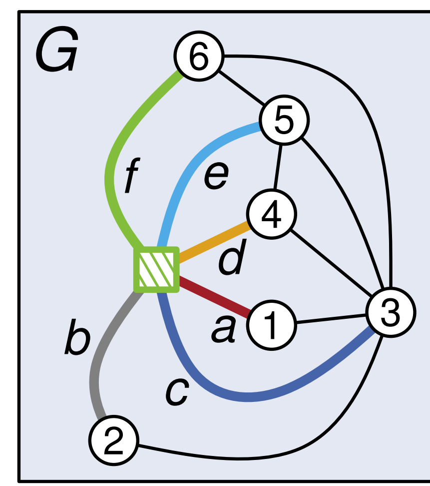
Every inner node in a PQ-tree is either a P-node or a Q-node.

- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal

Every embedding induces a (cyclic) ordering on the leaves.





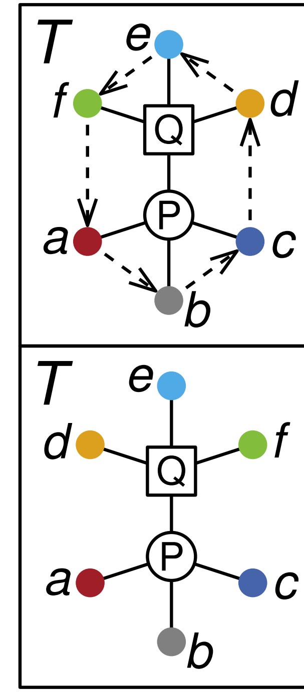


Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

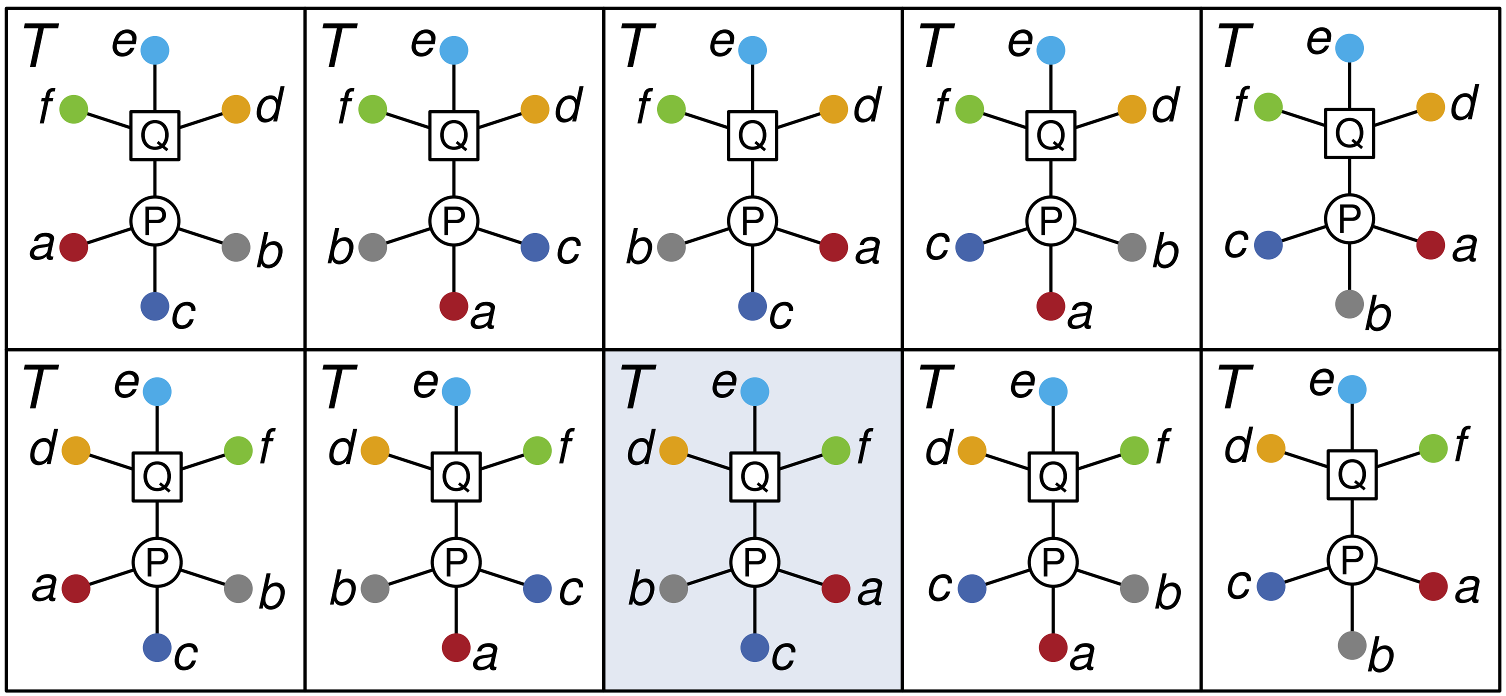
PQ-Trees

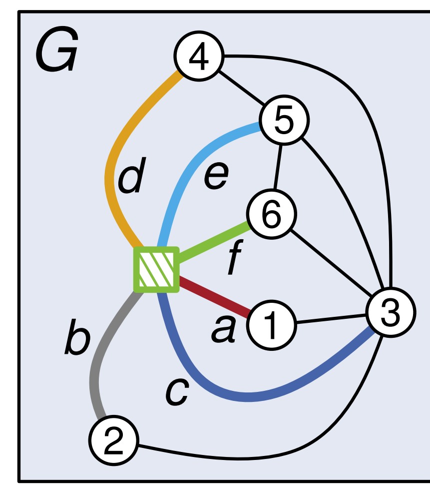
Every inner node in a PQ-tree is either a P-node or a Q-node.

- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal



Every embedding induces a (cyclic) ordering on the leaves.



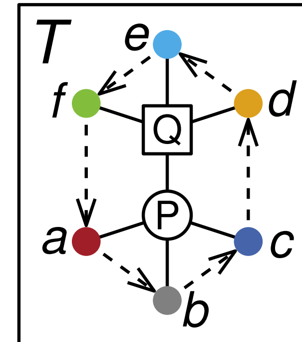


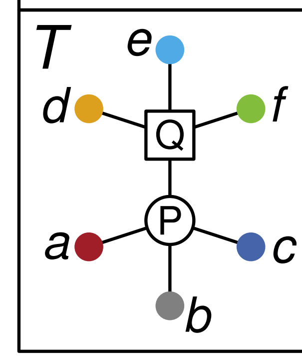
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

PQ-Trees

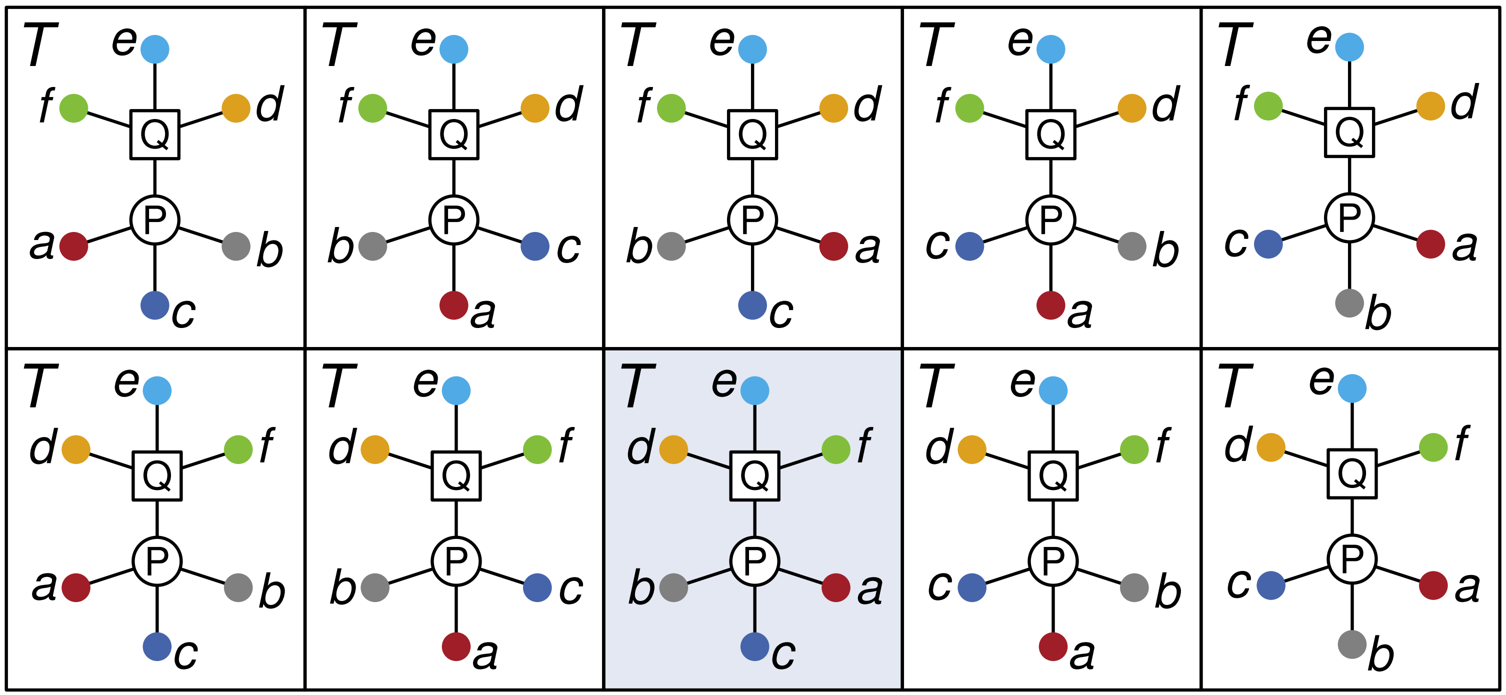
Every inner node in a PQ-tree is either a P-node or a Q-node.

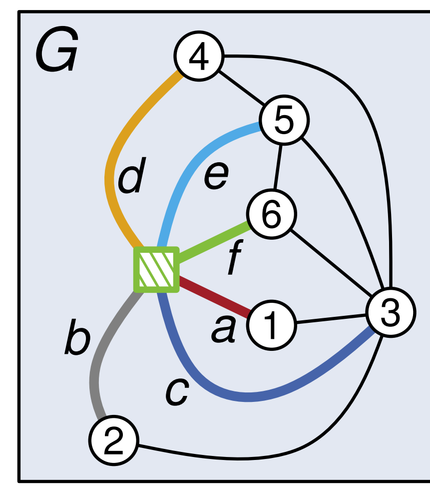
- **P-nodes:** choose arbitrary edge-ordering
- **Q-nodes:** edge-ordering is fixed up to reversal





Every embedding induces a (cyclic) ordering on the leaves.

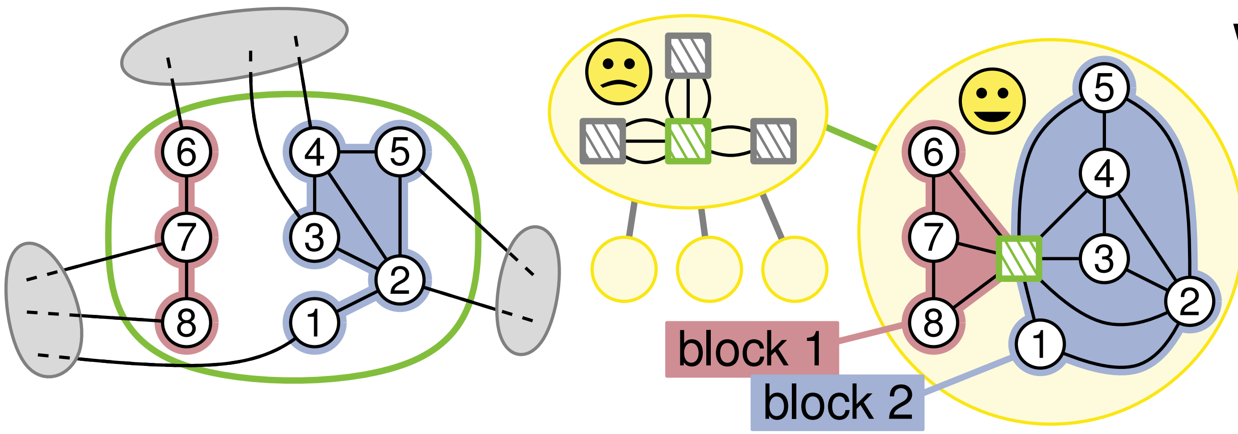




T represents the possible edge-orderings around ▨ in *G*.

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Flat-Clustered Graphs – ~~Fixed Embedding~~



What is fixed?

- ~~embedding of *G*~~
- ~~edge-orderings of non-virtual vertices~~
- ~~embeddings of blocks~~

Which edge-orderings around does 's skeleton allow?

- blocks in 's skeleton partition the edges
- partitions must not alternate → **partition-constraint**
- order in each partition **is restricted by a PQ-tree** → **PQ-constraint**

partitioned PQ-constraint

> **Theorem**
> C-planarity for flat-clustered graphs is equivalent to **planarity with partitioned PQ-constraints**.