Einbettungs-Constraints

- Modell, Planaritätstest und optimales Kanteneinfügen -

Karsten Klein

Vorlesung

Automatisches Zeichnen von Graphen

WS 07/08 - 07. Januar 2008

TU Dortmund, Fakultät für Informatik,Ls11 Algorithm Engineering

Überblick

- Kurzwiederholung: Planarität
- Einbettungs-Constraints (*ec*) Modell
- ec-Planaritätstest

Dienstag:

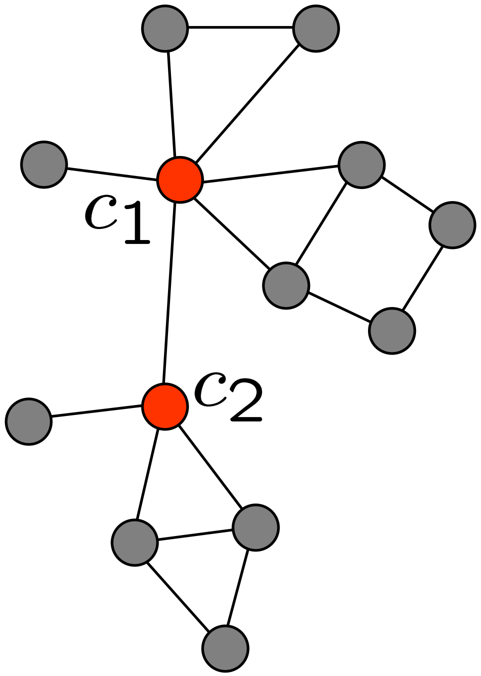
- (ec-)Kanteneinfügen

Planarität

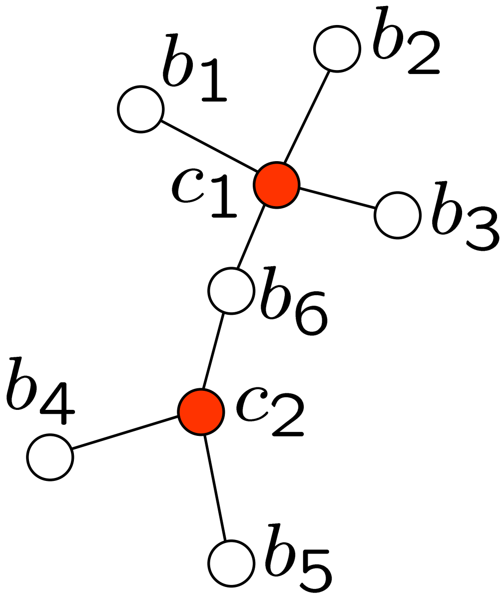
Teil I – Planarität

- Was ist Planarität?
- Test?
- Preprocessing möglich?
- Datenstrukturen?

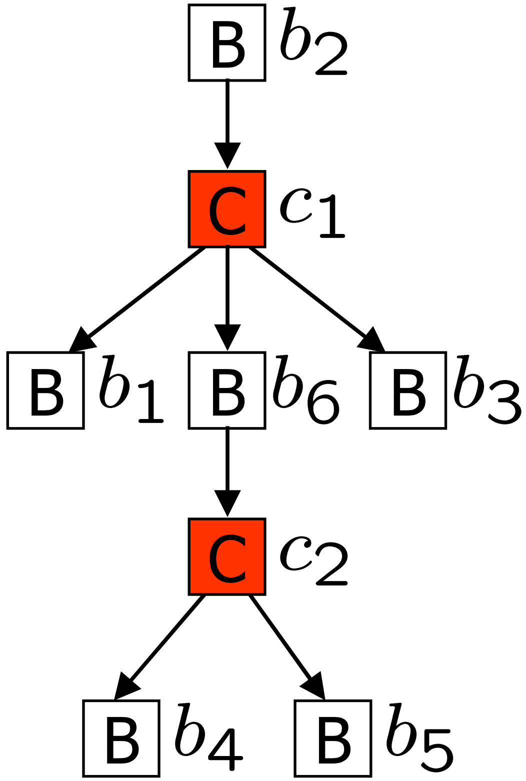
Blockgraph



Graph



Block-Graph



BC-Baum

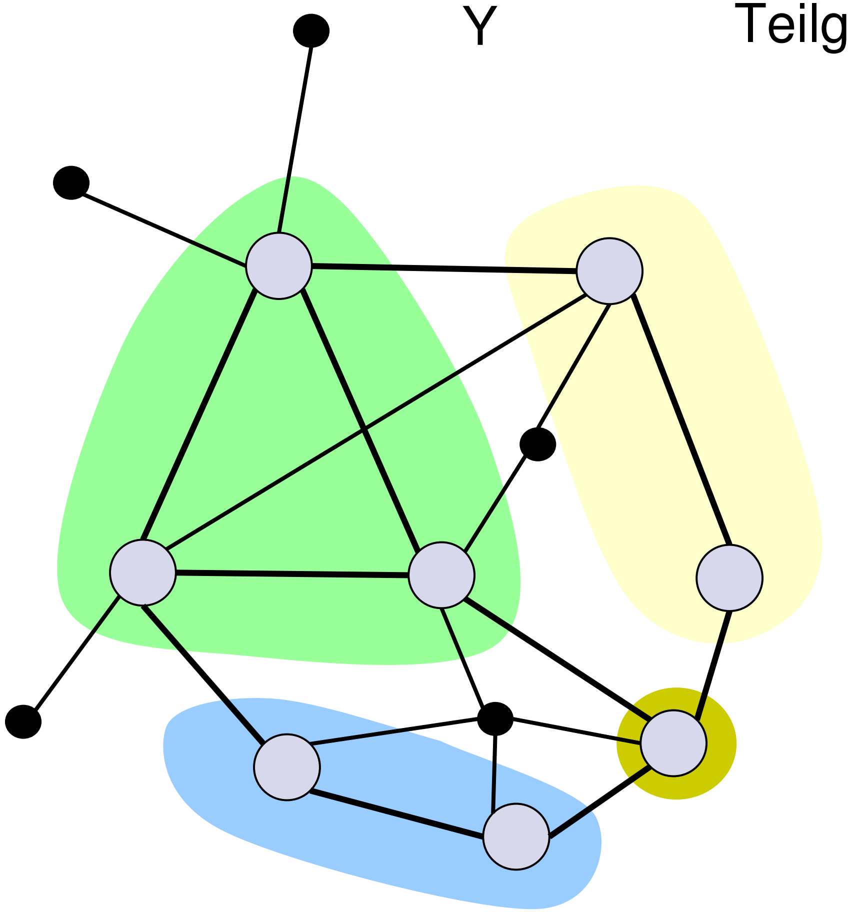
Eigenschaften planarer Graphen

- Kantenzahl
- Minoren (Verteilung?)
- ...

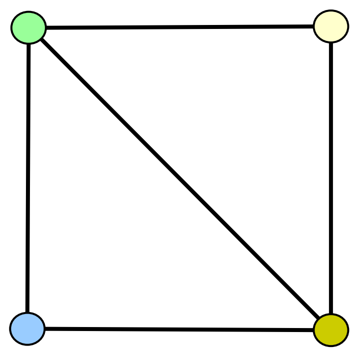
Minor

Y

Teilgraph G



X



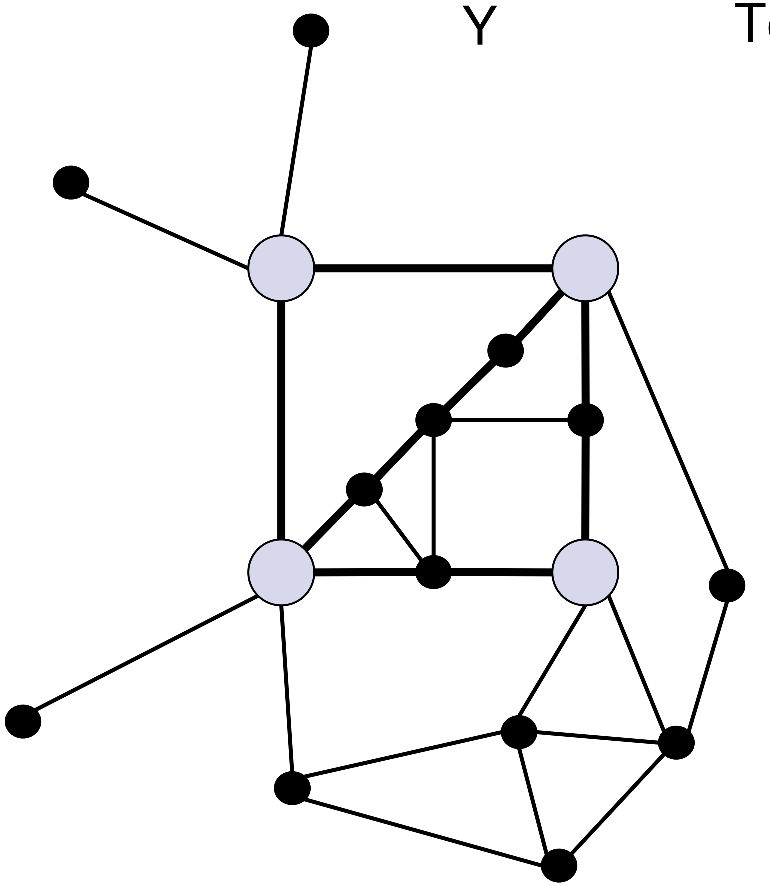
X ist Minor von Y

Partition zusammenhängender Teilmengen + Kontraktion

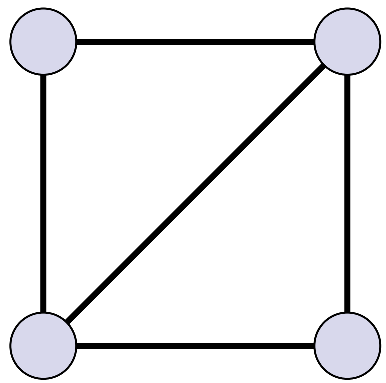
Topologischer Minor

Y

Teilgraph G



X



X ist topologischer Minor von Y

Subdivision: Pfade als Kantenunterteilungen

Planarität

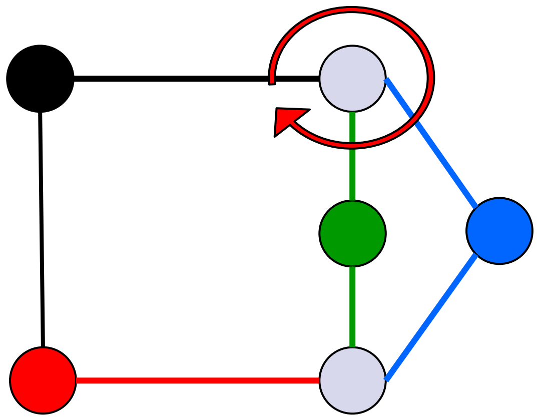
Theorem (Kuratowski '30, Wagner '37):

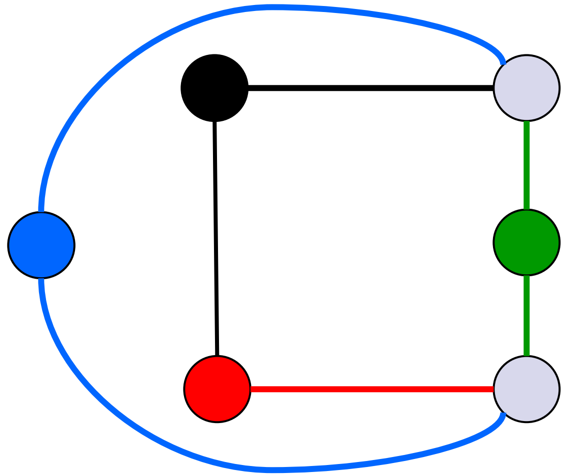
Folgende Aussagen sind äquivalent für einen Graph G:

- G ist planar
- G enthält weder einen K_5 noch einen $K_{3,3}$ als Minor
- G enthält weder einen K_5 noch einen $K_{3,3}$ als topologischen Minor

Einbettungen

- Kombinatorische Einbettung
- Planare Einbettung: Kombinatorische Einbettung und Wahl eines äußeren Faces





- Eulersche Polyederformel n-m+f=2

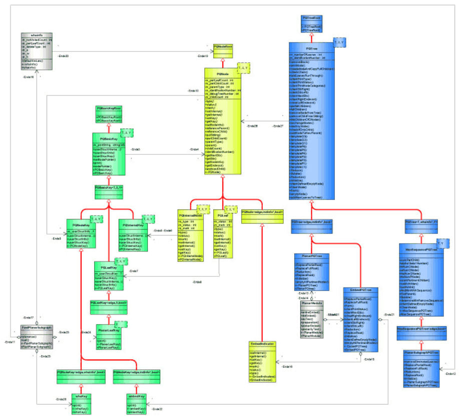
Wir benutzen hier…

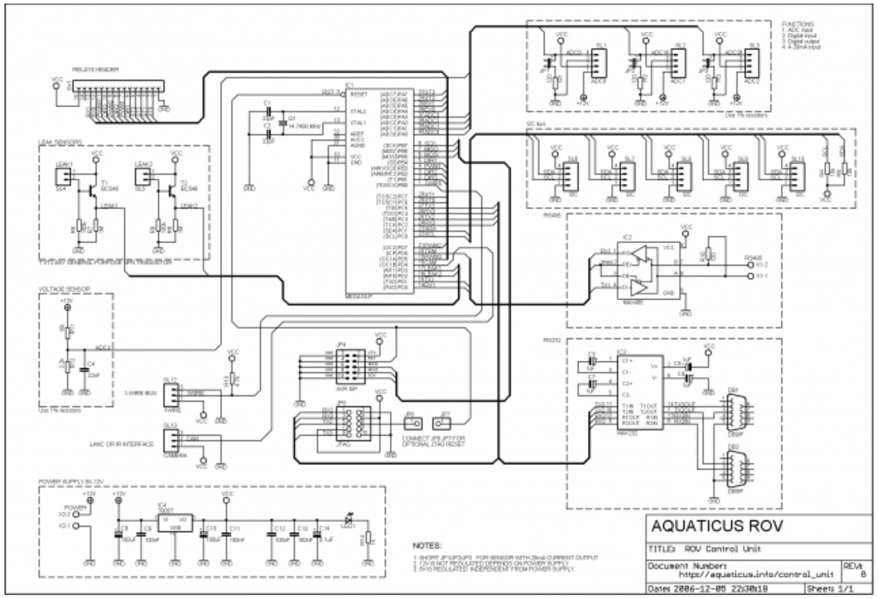
- SPQR – Bäume (Di Battista + Tamassia ´96) zur Optimierung über alle Einbettungen und zur Zerlegung
- Blockgraphen (Block-Cutvertex – Bäume)
- Standard – Planaritätstest als Blackbox

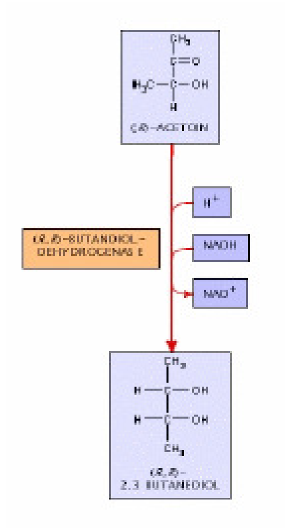
Beschränkung der möglichen Einbettungen

Was sind Einbettungsconstraints?

- Beschränkung der möglichen Kanten-reihenfolge um die Knoten
- WICHTIG für Anwendung von Graphen-zeichnen in praktischen Anwendungen







Was sind Einbettungsconstraints?

- Beschränkung der möglichen Kanten-reihenfolge um die Knoten
- WICHTIG für Anwendung von Graphen-zeichnen in praktischen Anwendungen
  - UML Diagramme – Seitenconstraints
  - Schaltkreisschemata – Portconstraints
  - Biologische und chemische Netzwerke – Einbettung abhängig von Semantik
  - Datenbankschemata
  - Interaktives Graphenzeichnen – „Mental Map“
  - Berücksichtigung allgemeiner oder persönlicher Zeichenkonventionen
  - …

Constraints in Anwendungen

Trotzdem selten betrachtet

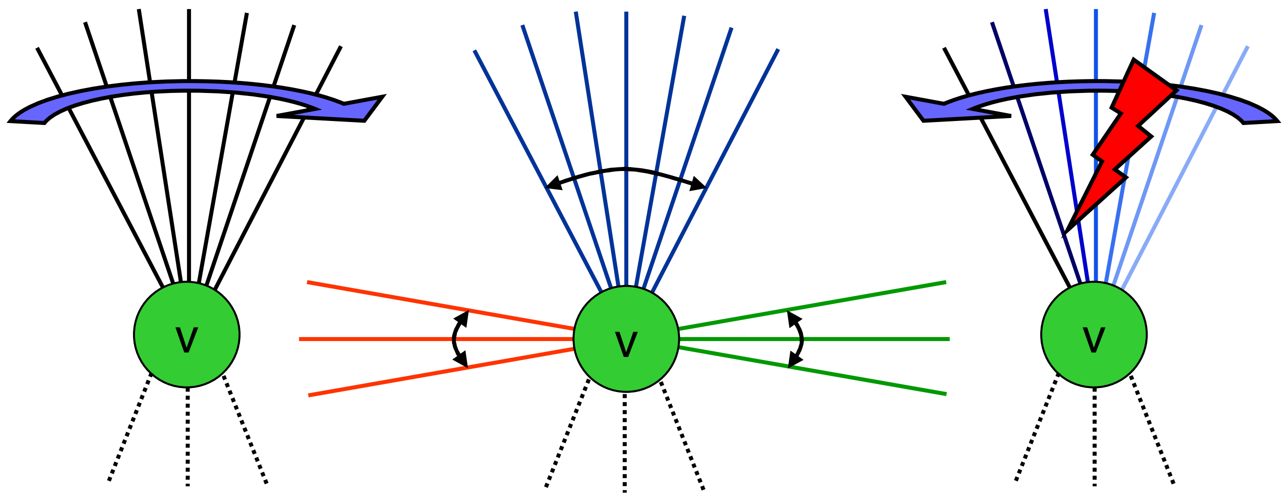
Ansätze bisher
• Database schema drawings [Di Battista et al. `02]
• Topological constraints [Dornheim `02]
• Embedding constraints [Gutwenger et al. `06]
• Level Planarity Testing with Embedding Constraints [Harrigan and Healy `07]

- Hier: Einige Typen von Einschränkungen in einem hierarchischen Modell

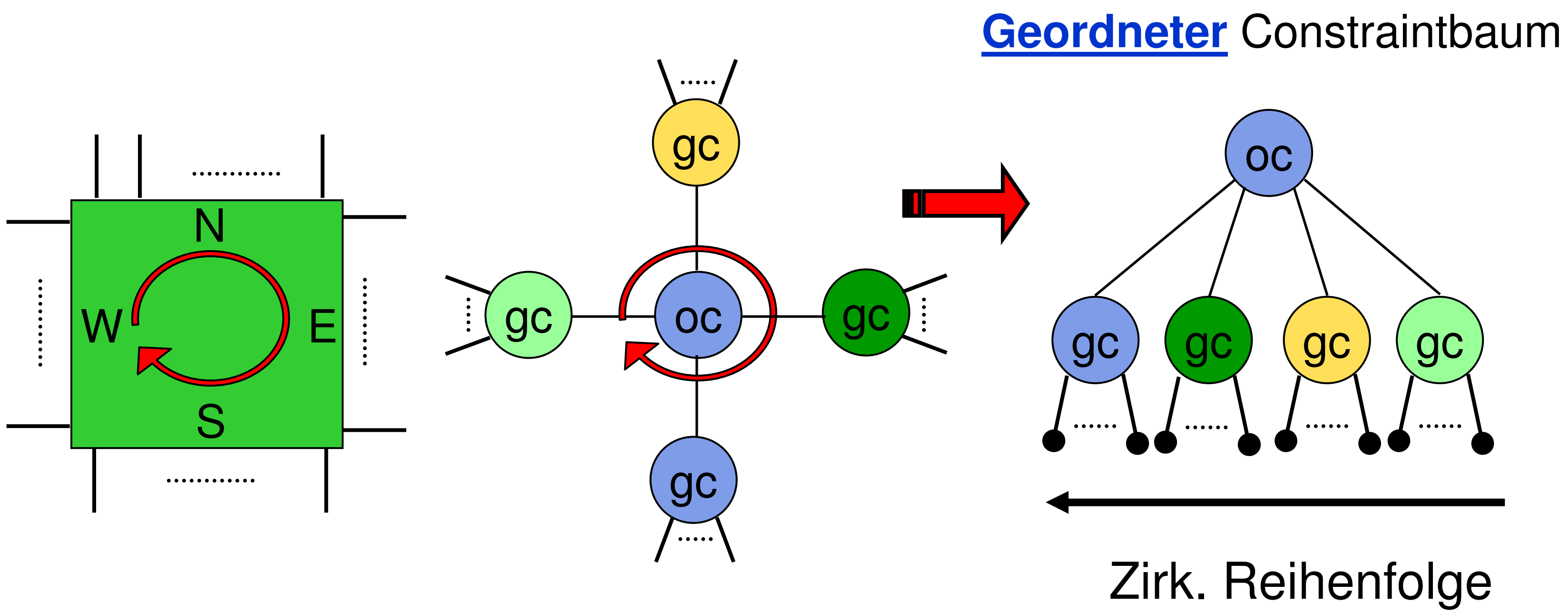
Teil II – Einbettungs-Constraints

- *grouping constraint* (gc): Disjunkte Kantenmengen mit beliebiger Reihenfolge
- *mirror constraint* (mc): Feste Reihenfolge der Kanten
- *oriented constraint* (oc): Feste zirkuläre Reihenfolge

Diese Constraints dürfen auch kombiniert werden



Beispiel: Seiten-Constraints



Achtung: Einbettung sichert nur gültige Reihenfolge, nicht die Orientierung!

Repräsentation von Permutationsreihenfolgen auch in PQ-/PC-Bäumen

Einbettungs-Constraints

Definition: Einbettungs-Constraint

Einbettungs-Constraint an einem Knoten v ist def. als

- Gewurzelter, geordneter Baum T_v
- Blätter sind die zu v inzidenten Kanten
- Innere Baumknoten repräsentieren Constraints

  gc mc oc

- Constraint bestimmt mögliche Reihenfolge der Kinder
- Blätterfront wird interpretiert als zirkuläre Kantenreihenfolge um v

Knoten mit einem Kind sind offensichtlich obsolet, deshalb nicht erlaubt

Pro Knoten wird nur ein Einbettungsconstraint erlaubt

Einbettungs-Constraints

Definition: ec-planare Einbettung

Eine planare Einbettung Γ eines Graphen G, die die Einbettungsconstraints in einer Menge C erfüllt, heißt *ec-planare Einbettung* von G bezüglich C.

Definition: ec-Planarität

(G,C) *ec-planar* $\Leftrightarrow \exists$ ec-planare Einbettung Γ von G bezüglich C.

Zusammenfassung

- Drei grundlegende Constraints:
  Grouping, *Mirror*, *Oriented*
- Einbettungsconstraints an Knoten: Baum von Constraints
- ec-planare Einbettungen: Mit Constraints planar
- ec-Planarität für (G,C)

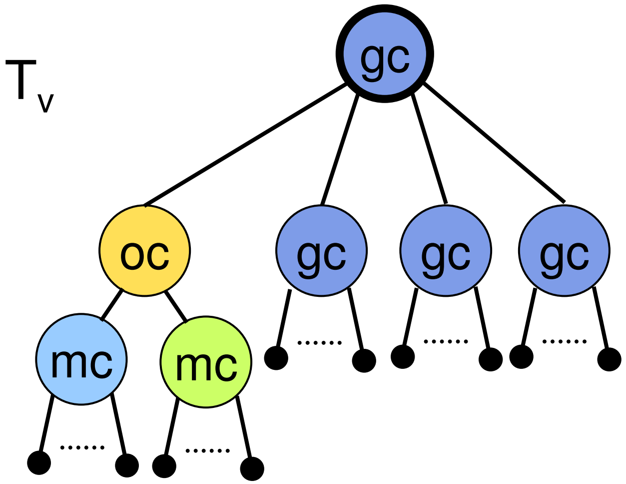
Teil II – ec-Planaritätstest

Eingabe: Graph *G* mit Menge *C* von Einbettungs-Constraints
Ausgabe: Ist (*G*, *C*) ec-planar?

1. Strukturelle Transformation: *ec-Expansion*
2. Prüfen zusätzlicher Bedingungen

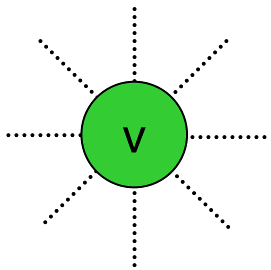
ec-Expansion

- Transformation des Eingabegraphen G mit Constraintmenge C in den *Expansionsgraph* $E(G, C)$
- Grundlegende Operation, die Planaritätstest vereinfacht
- Orientiert sich an der Struktur der Constraintbäume

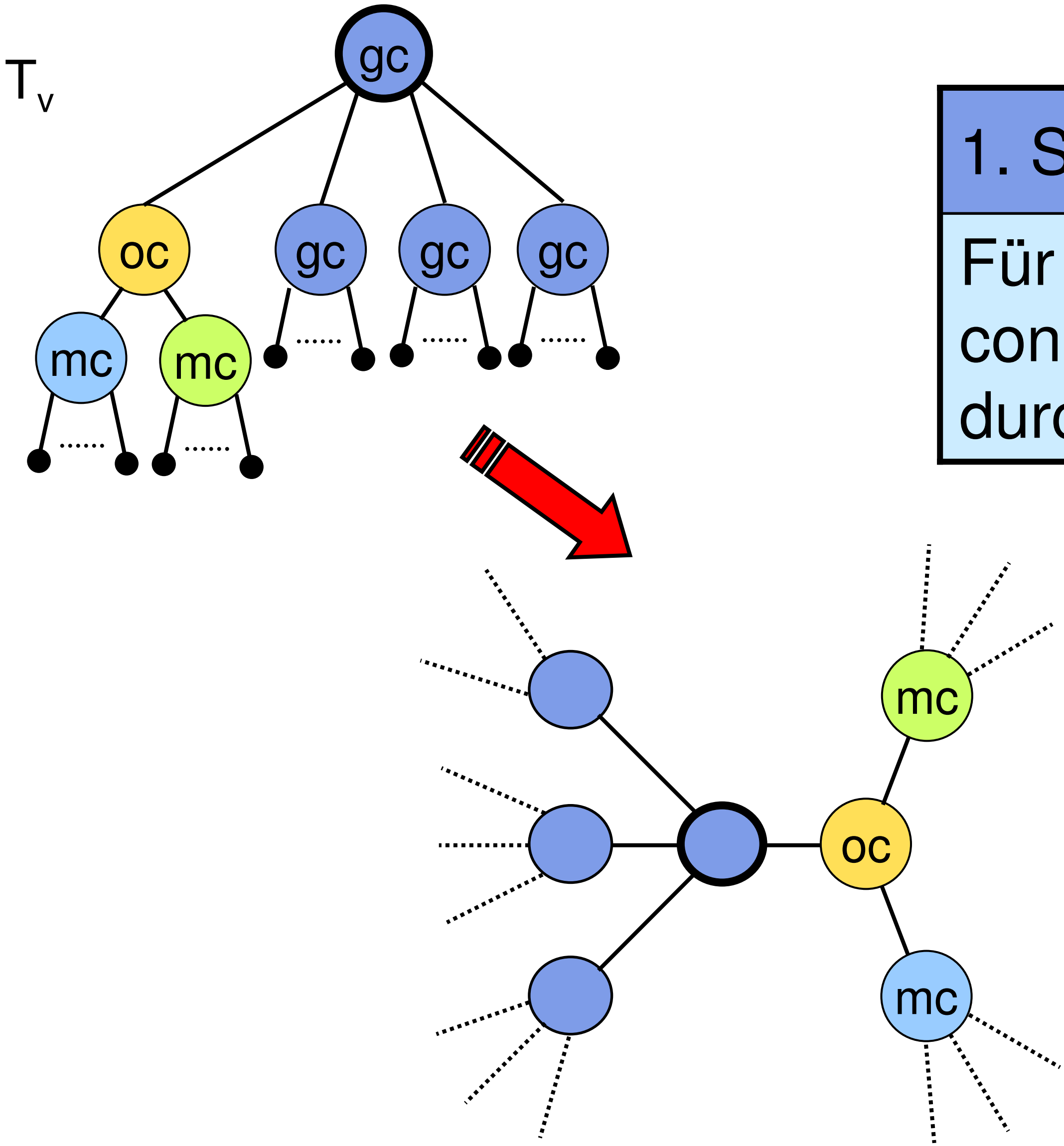


1. Schritt

Für jeden Einbettungs-constraint T_v ersetze v durch T_v ohne die Blätter



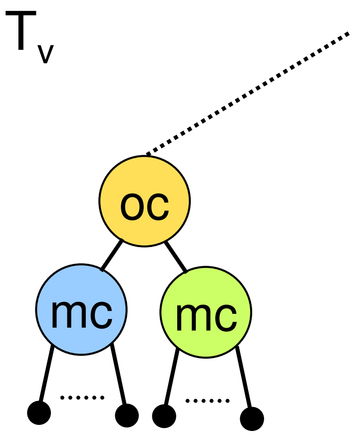
Ziel: Gruppierung der Kanten, erfüllt grouping constraints



1. Schritt

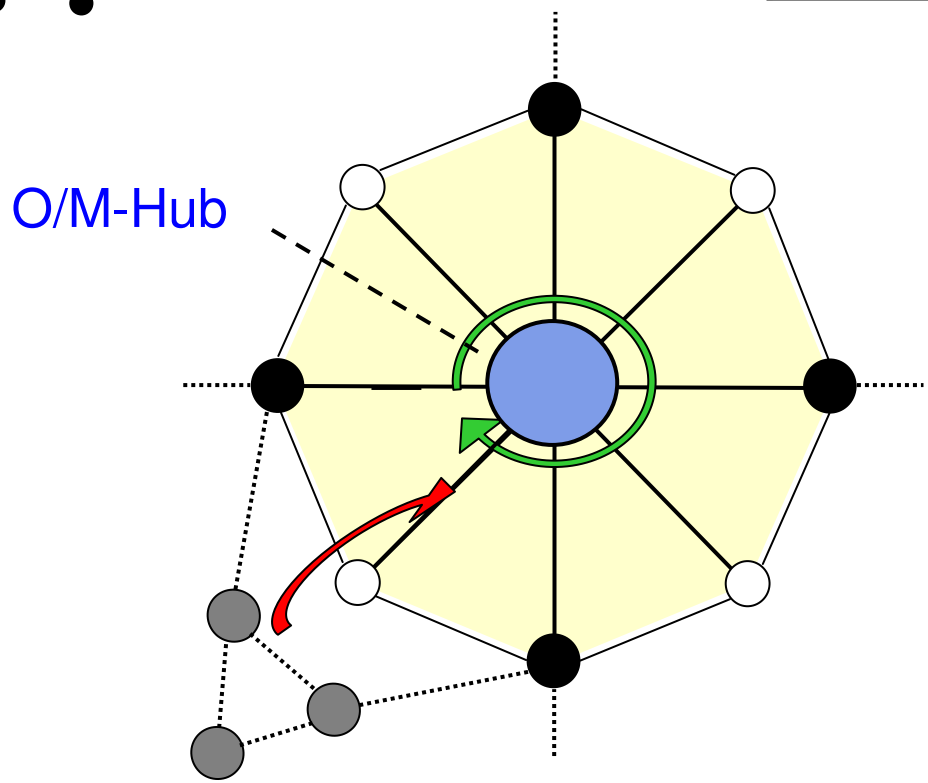
Für jeden Einbettungs-constraint T_v ersetze v durch T_v ohne die Blätter

Ziel: Gruppierung der Kanten, erfüllt grouping constraints



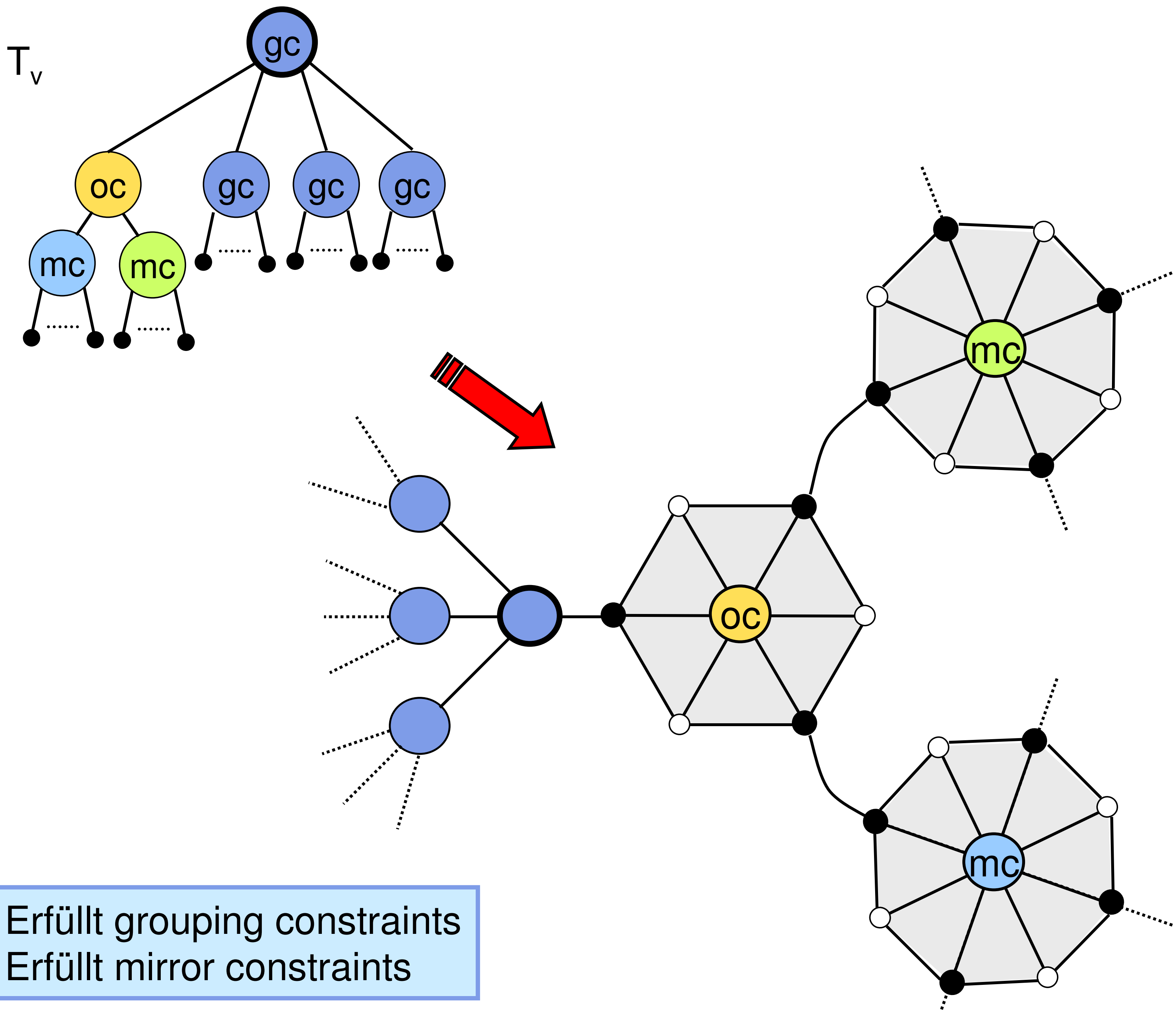
2. Schritt

Weitere Ersetzung für oriented / mirror Constraints



Wheel gadget

Ziel: Festlegung der Kantenreihenfolge, erfüllt Mirrorconstraints



Erfüllt grouping constraints
Erfüllt mirror constraints

ec-Expansion

Lemma 1:

Die ec-Expansion $E(G, C)$ eines Graphen $G=(V, E)$ mit Constraintmenge C hat Größe $O(|V|+|E|)$ und kann in Zeit $O(|V|+|E|)$ aufgebaut werden.

Beweis:

Es reicht, einzelne ec-Bäume T_v zu betrachten, $|C| \leq |V|$

- T_v ist linear größenbeschränkt durch $deg(v)$
- Wheels für Constraint c haben $4 \cdot deg(c)$ Kanten
- Expansion erzeugt $O(deg(v))$ Kanten

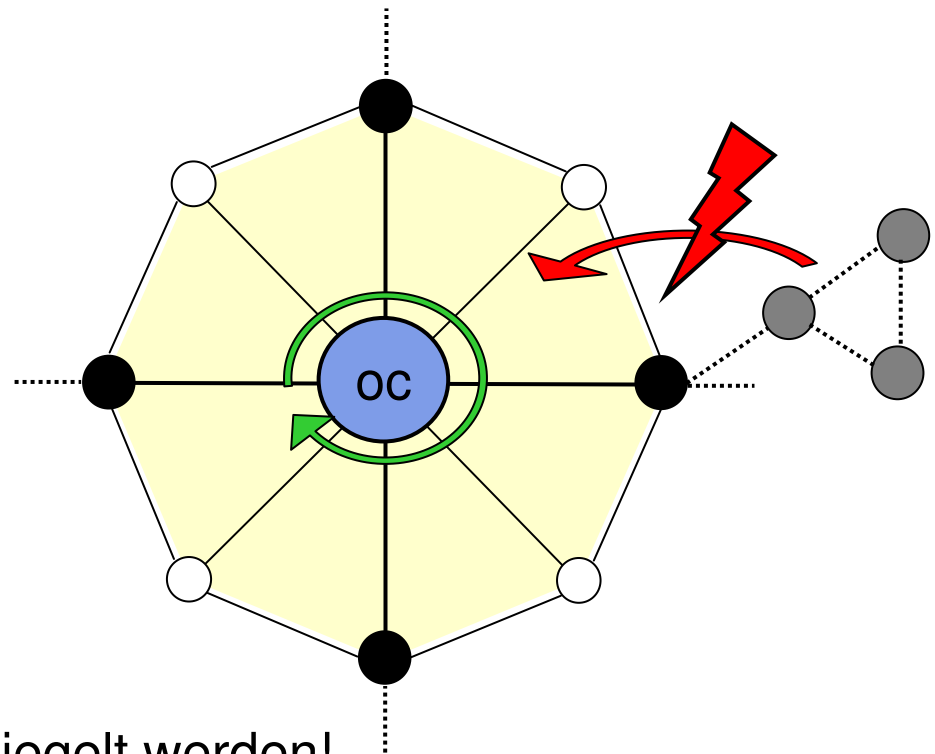
Die Gesamtsumme der neuen Kanten ist damit begrenzt durch $\sum_{v \in V} O(deg(v)) = O(|E|)$

ec-Expansion und ec-planare Einbettungen

- ec-Expansion erzwingt Einhaltung der grouping und mirror-Constraints
- Nicht jede Einbettung der ec-Expansion läßt sich auch als korrekte Einbettung von *(G, C)* interpretieren:
    1. Oriented constraints müssen nicht erfüllt sein
    2. Expansion wheels machen die Sache komplizierter

ec-planare Einbettungen: Zusätzliche Bedingungen

1. Jeder O-hub ist korrekt orientiert
2. Das äußere Face darf keinen Hub enthalten
3. Faces an einem Hub sind Wheel Dreiecke



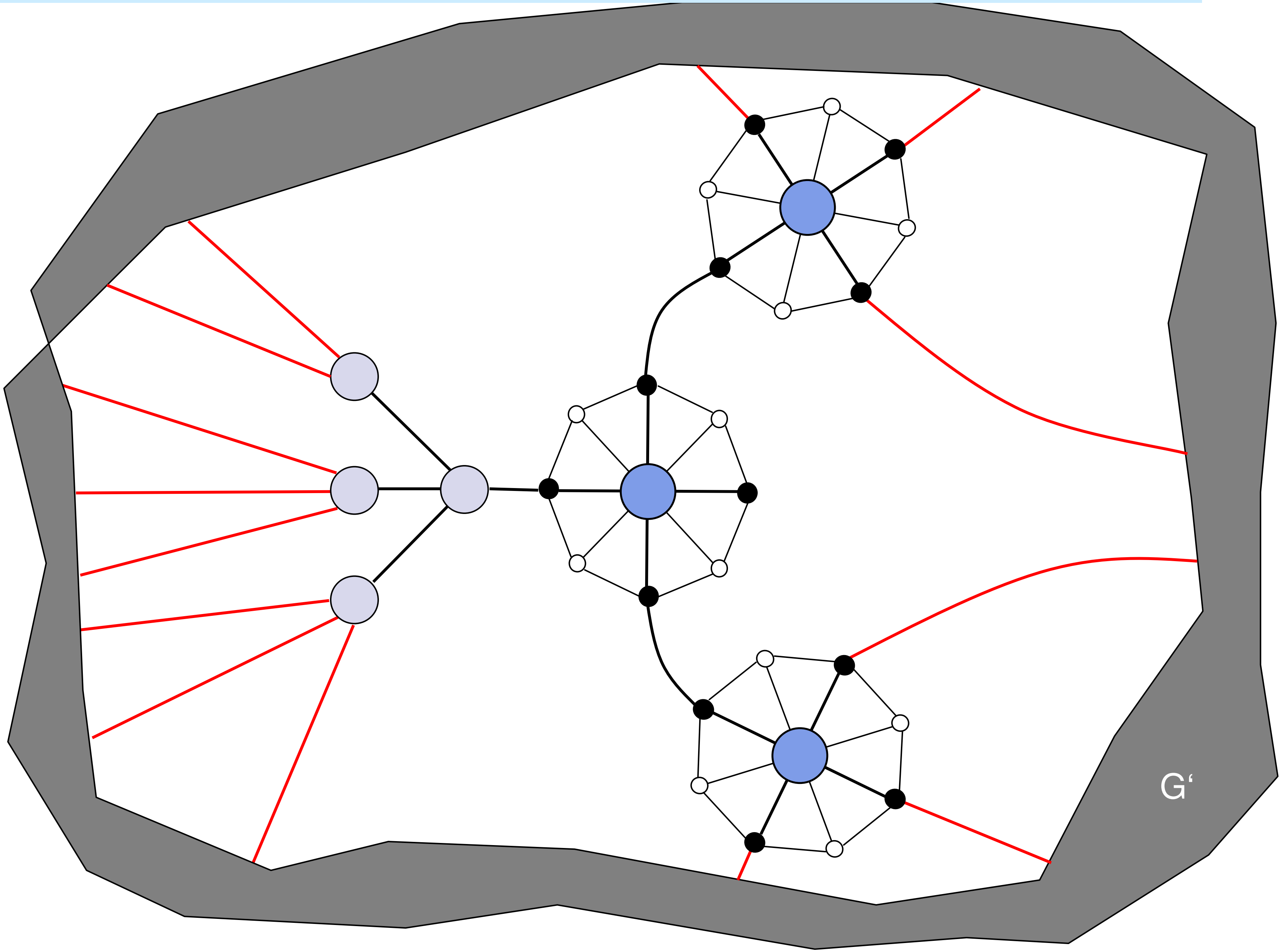
Komb. Einb. kann gespiegelt werden!

ec-planare Einbettungen: Zusätzliche Bedingungen

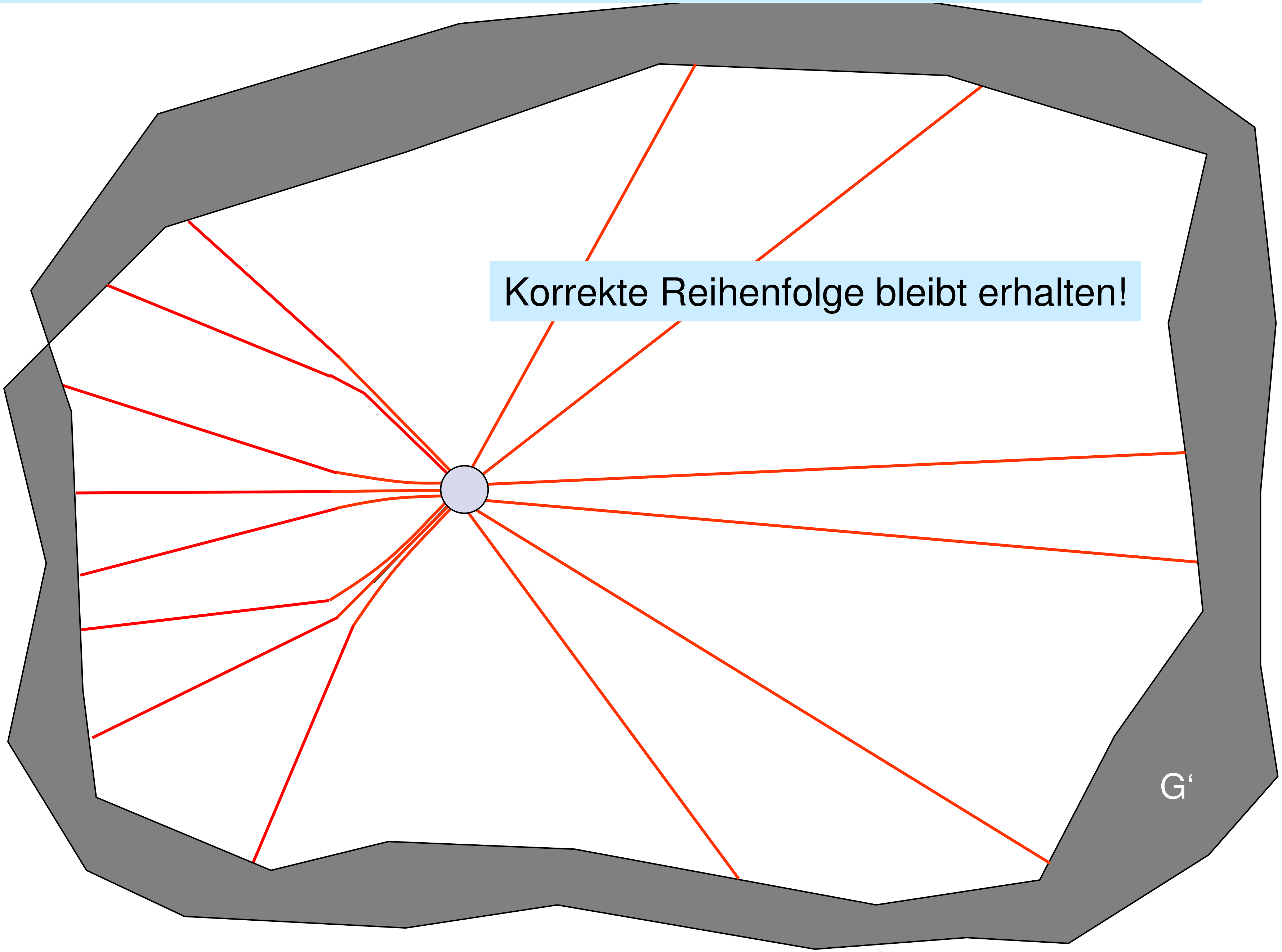
1. Jeder O-hub ist korrekt orientiert
2. Das äußere Face darf keinen Hub enthalten
3. Faces an einem Hub sind Wheel Dreiecke

Definition:
Planarität +1.+2.+3. = *ec-planare Einbettung* von *E(G,C)*

ec-planare Einbettung von *E(G, C)* $\Rightarrow$ ec-planare Einbettung von *(G, C)*



ec-planare Einbettung von *E(G, C)* $\Rightarrow$ ec-planare Einbettung von *(G, C)*



ec-Expansion & ec-planare Einbettungen

Lemma 2:

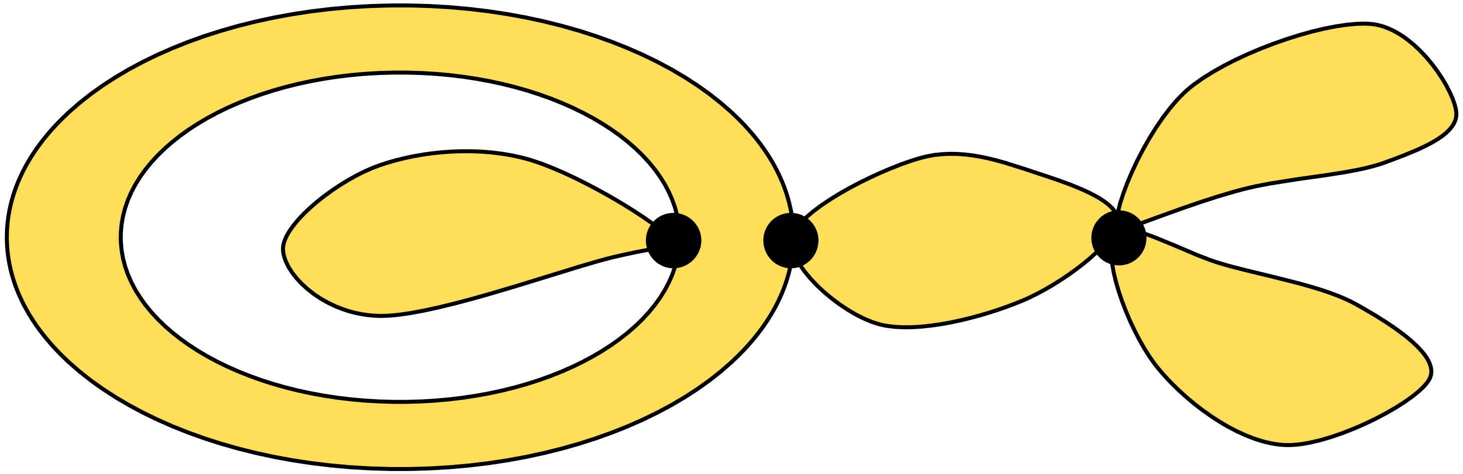
Sei *G* ein Graph mit Einbettungsconstraints *C*.

- *(G,C)* ist ec-planar $\Leftrightarrow$ *$E(G,C)$* ist ec-planar.
- Jede ec-planare Einbettung von *$E(G,C)$* entspricht einer ec-planaren Einbettung von *(G,C)*.

Unser Problem liegt jetzt also darin, ec-Planarität von *$E(G,C)$* zu bestimmen

ec-Expansion und ec-Planarität

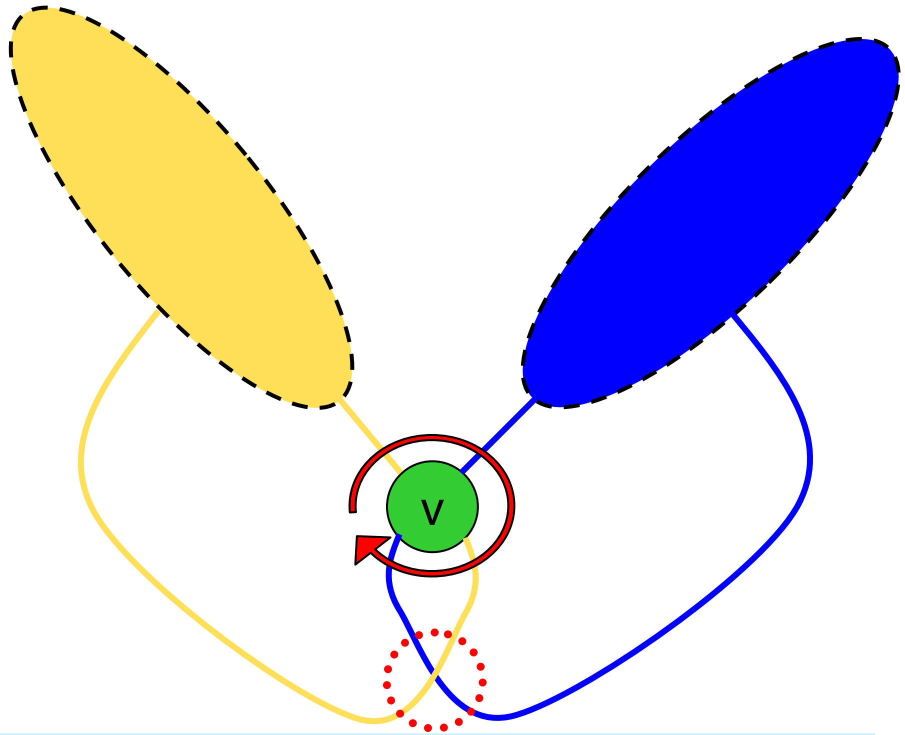
Planaritätstest für normale Graphen auf Blöcken



Und für Einbettungsconstraints?

ec-Expansion und ec-Planarität

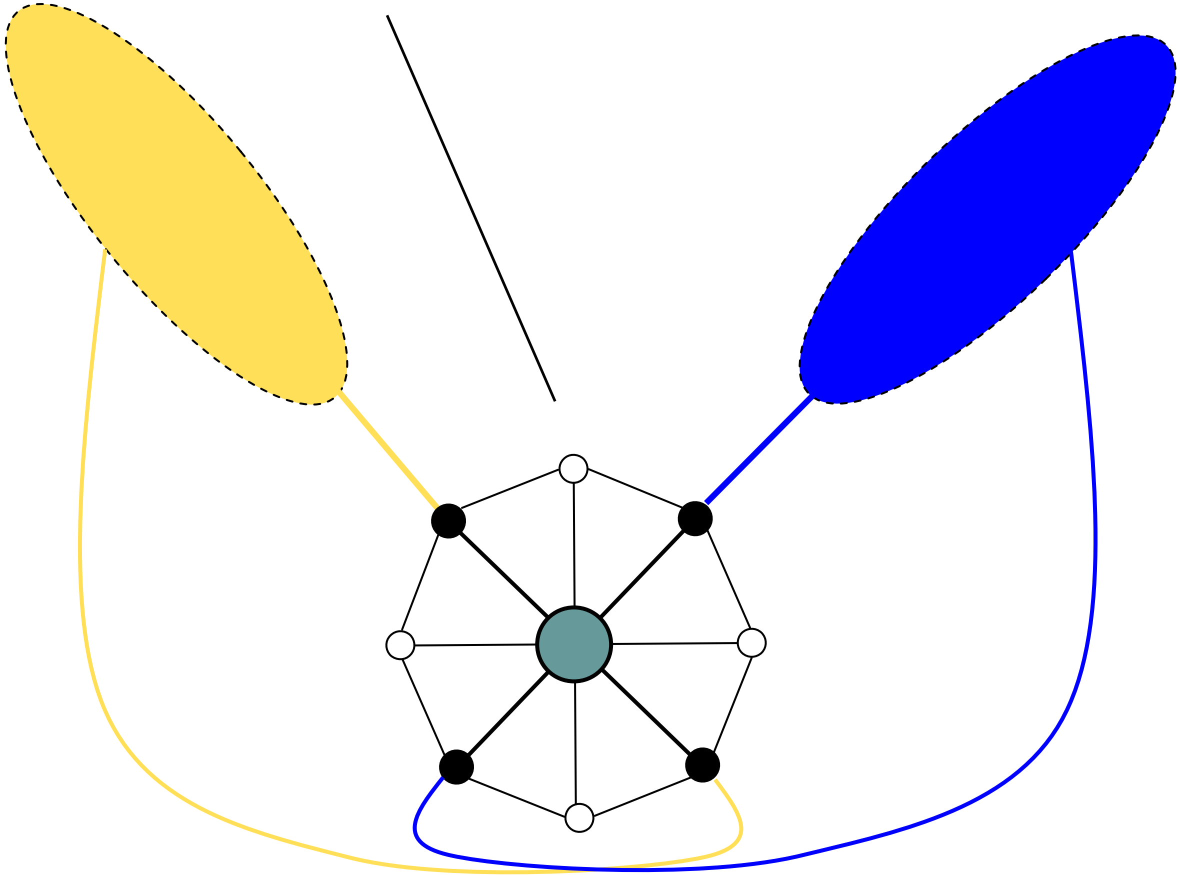
Test von einzelnen Blöcken reicht für ec-Planarität nicht aus!



Kein Blocktest für *(G,C)*. Was ist mit *E(G,C)*?

ec-Expansion und ec-Planarität

Expansion verbindet die Blöcke an Wheels



ec-Expansion & ec-Planarität

Lemma 3:

Sei G ein Graph mit Embeddingconstraints C.

- $E(G,C)$ ist ec-planar $\Leftrightarrow$ Jeder Block von $E(G,C)$ ist ec-planar.

Beweis:

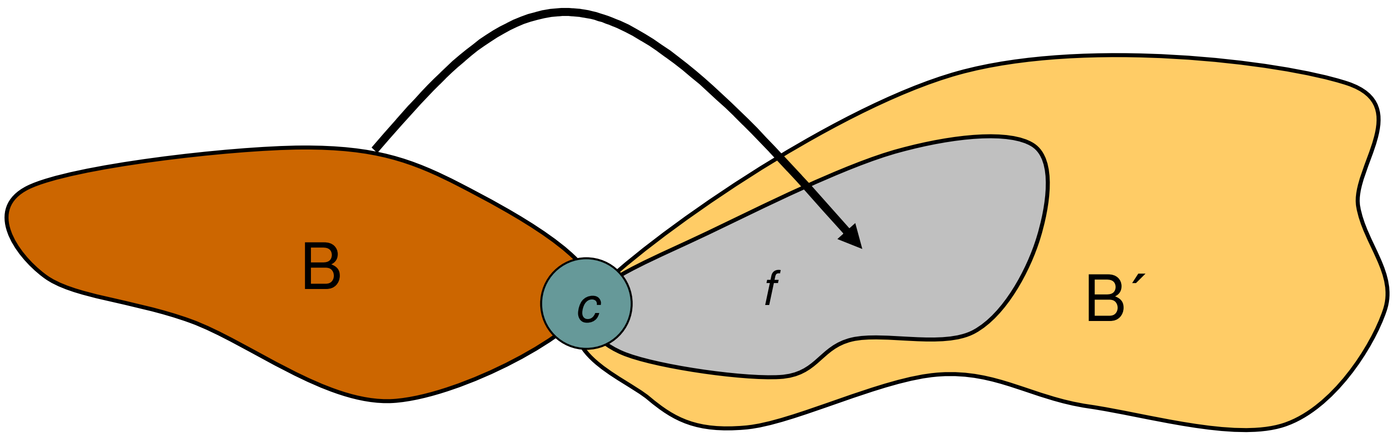
„$\Rightarrow$“: klar

„$\Leftarrow$“: 1. Jedes Wheel Gadget liegt vollständig in einem Block B
2. Jedes Wheelface ist ein Dreieckface in jeder Einbettung von B
3. Hubs sind keine Artikulationen von $E(G,C)$…

Konstruktion einer ec-planaren Einbettung

B und B´ ec-planar eingebettet

f ist kein Wheelface
c kann kein Hub sein



Identifiziere äußeres Face von B mit *f*

ec-Expansion & ec-Planarität

Beweis (Fortsetzung):

Wir konstruieren eine ec-planare Einbettung von $E(G,C)$:

Starte mit beliebigem Block B von $E(G,C)$

Sei Π ec-planare Einbettung von B (ext. Face nicht in Wheel)

Füge weitere Blöcke hinzu: Sei Block B' mit B über Knoten c verbunden und habe ec-planare Einbettung Π'

Wähle nicht Wheel-Faces f und f' adjazent zu c (c ist kein Hub) in Π und Π'

Füge Π' mit f' als externem Face in Face f von Π ein…

ec-Expansion & ec-Planarität

Beweis (Fortsetzung):

Wir haben nun eine ec-planare Einbettung von $B \cup B'$!

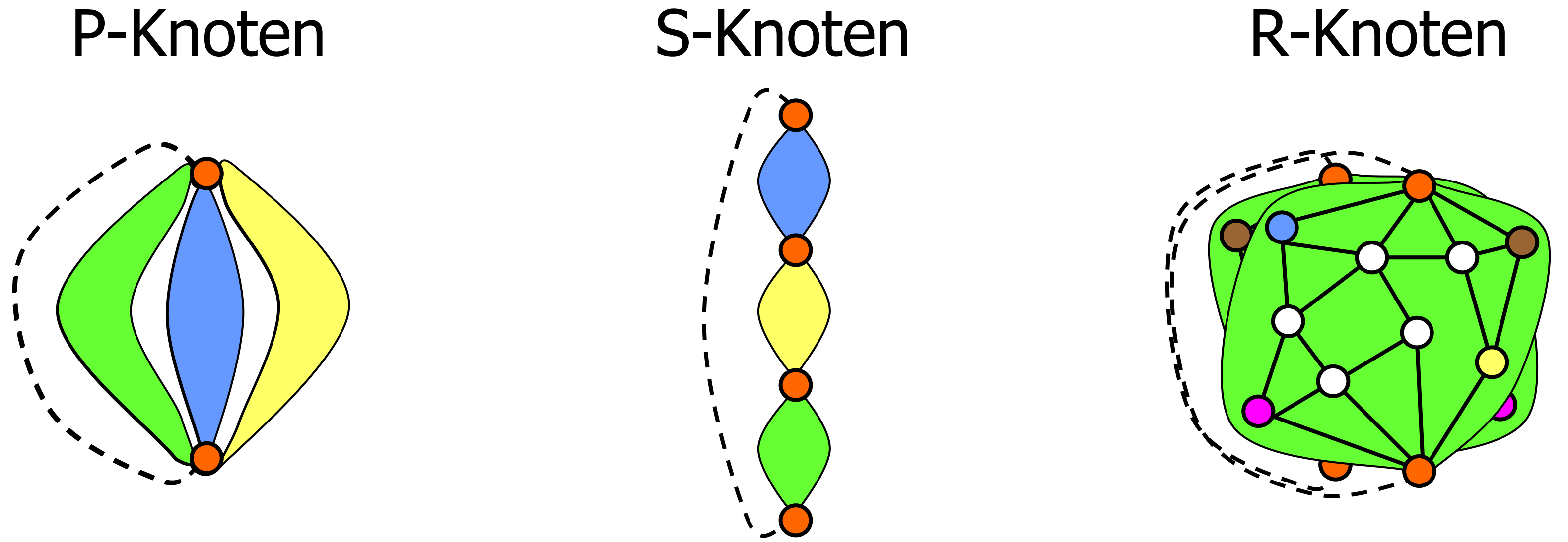
Weitere Blöcke können ebenso in diese Einbettung eingefügt werden

Dadurch erhalten wir eine ec-planare Einbettung von $E(G,C)$

$\Rightarrow$ Wir brauchen „nur“ ec-Planaritätstest für jeden Block von $E(G,C)$

Charakterisierung von ec-planaren Einbettungen

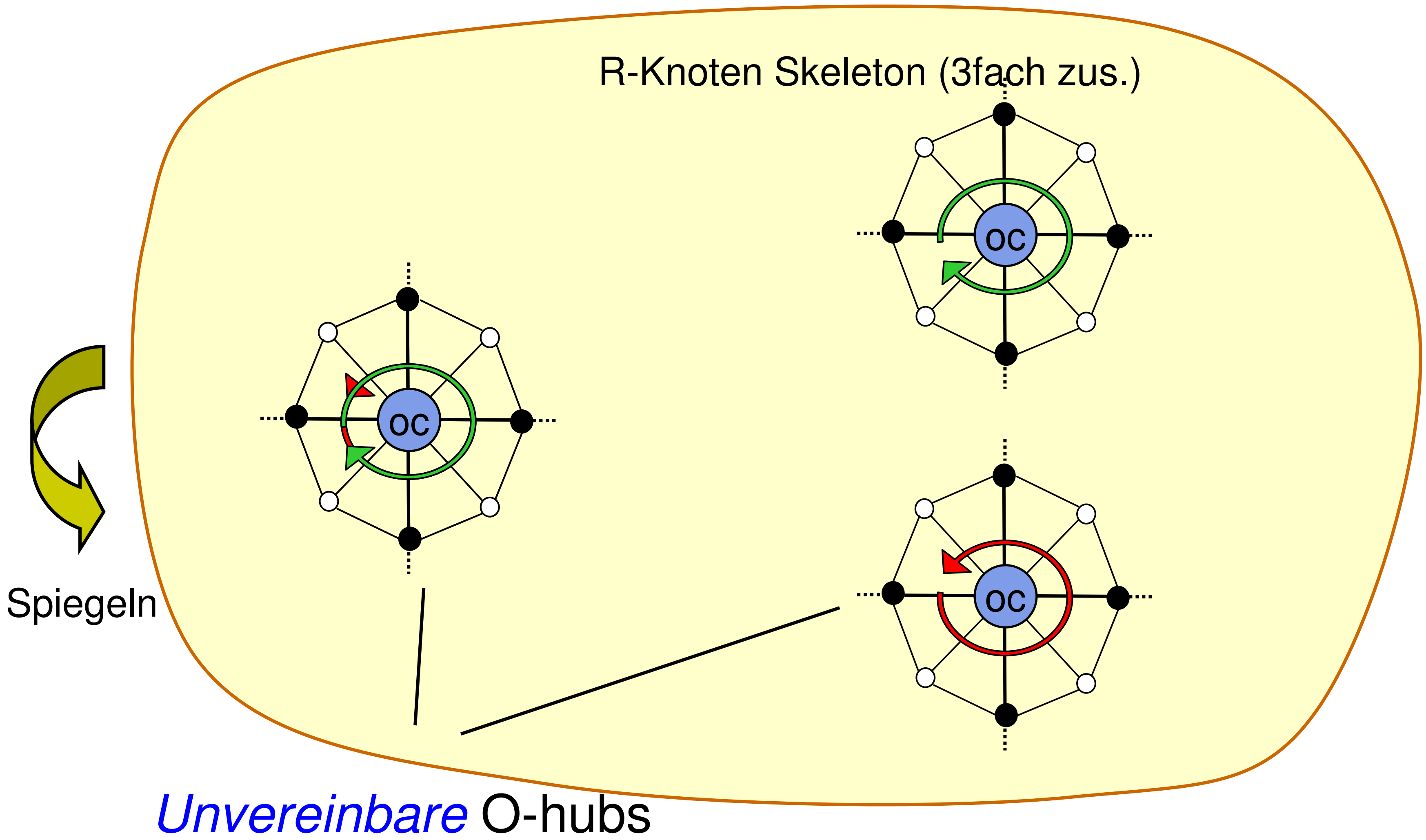
- Nutzen SPQR-Bäume um alle kombinatorischen Einbettungen eines Blocks von *E(G,C)* zu repräsentieren



Beobachtung: O-hubs liegen in genau einem R-node Skeleton

R-Knoten Skeletons mit O-hubs

Falls *B* planar, ist Skeleton auch planar mit zwei Einbettungen



Charakterisierung von ec-planaren Einbettungen

Theorem:

Sei G ein Graph with Einbettungsconstraints C.

Sei B ein Block von $E(G, C)$ und T der SPQR-Baum von B.

- B ec-planar $\Leftrightarrow$ B ist planar + kein Skeleton eines R-Knotens von T enthält unvereinbare O-hubs.
- Falls B ec-planar ist, ergeben die Einbettungen der Skeletons von T eine ec-planare Einbettung von B **iff** jeder O-hub im Skeleton eines R-Knotens ist korrekt orientiert.

Charakterisierung von ec-planaren Einbettungen

Beweis:

„$\Rightarrow$“: *B* ec-planar, klar.

„$\Leftarrow$“: *B* planar und keine unvereinbaren O-Hubs

Wir wählen für R-Knoten Skeletons mit O-Hubs jeweils die Einbettung, die O-Hubs korrekt orientiert, für alle anderen eine beliebige Einbettung.

Innere Wheel-Faces sind Dreiecke (Block!).

Wir wählen als externes Face ein beliebiges Nicht-Wheel-Face (existiert immer!)

Insgesamt erhalten wir damit eine ec-planare Einbettung von *B*.

ec-Planaritätstest Algorithmus

Gegeben Graph *G*, Einbettungsconstraints *C*

Konstruiere ec-Expansion *E(G, C)*

if *E* is not planar return false

for each block *B* of *E* do

  Konstruiere SPQR-tree *T* of *B*

  for each R-node $\mu \in T$ do

    if skeleton(μ) enthält unvereinbare O-hubs

      return false

return true

Zeitkomplexität

- Aufbau der ec-Expansion
- Planaritätstest
- Aufbau von SPQR-Bäumen [Gutwenger+Mutzel‘01]
- Erkennung unvereinbarer O-Hubs: Berechne Einbettung für R-Knoten Skeletons

Testen von ec-Planarität in Linearzeit

Zusammenfassung

- Einschränkung der Einbettung durch drei grundlegende Constraints:
  Grouping, *Mirror*, *Oriented*
- Einbettungsconstraints an Knoten: Baum von Constraints
- ec-planare Einbettungen und ec-Planarität für (G,C)
- Transformation in ec-Expansion $E(G,C)$
- Planaritätstest mit Nebenbedingungen testet ec-Planarität für (G,C)
- Linearzeit