Algorithmen zur Visualisierung von Graphen

Flussmethoden Teil 2: Kompaktierung und Aufwärtsplanarität

INSTITUT FÜR THEORETISCHE INFORMATIK · FAKULTÄT FÜR INFORMATIK

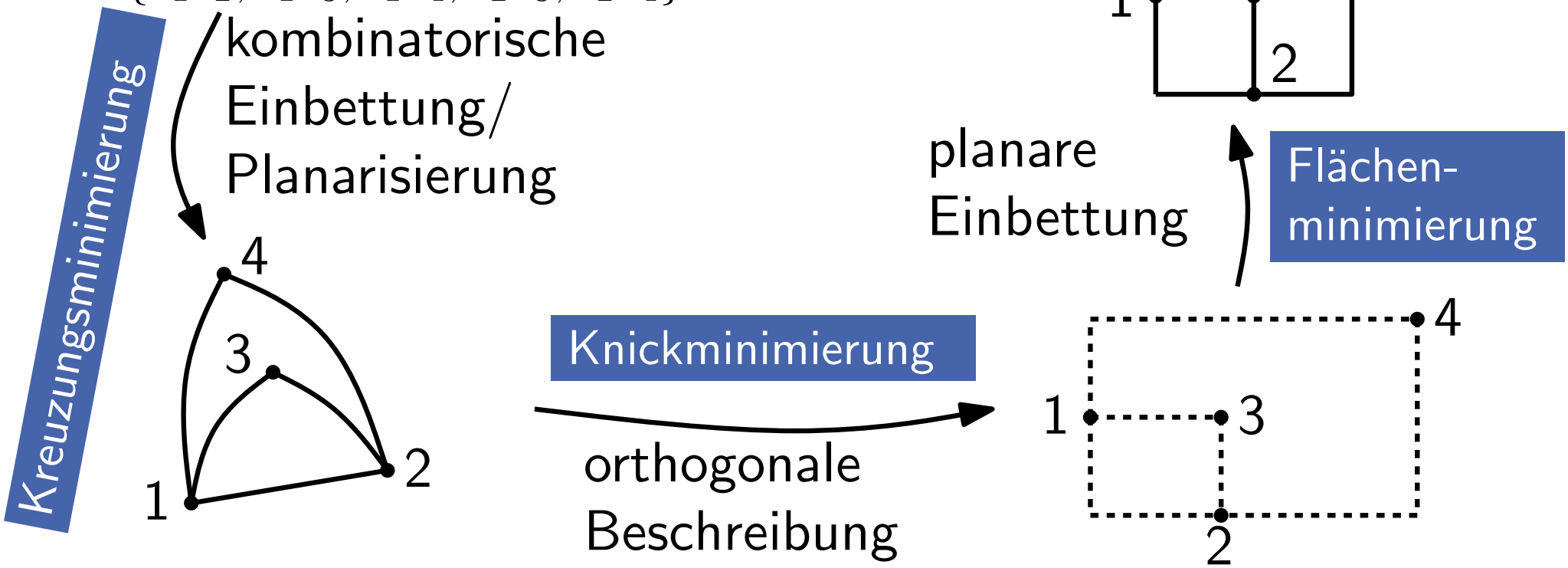
Tamara Mchedlidze · **Martin Nöllenburg**

14.01.2015

(Planare) Orthogonale Zeichnungen

Dreistufiger Ansatz: *Topology – Shape – Metrics*

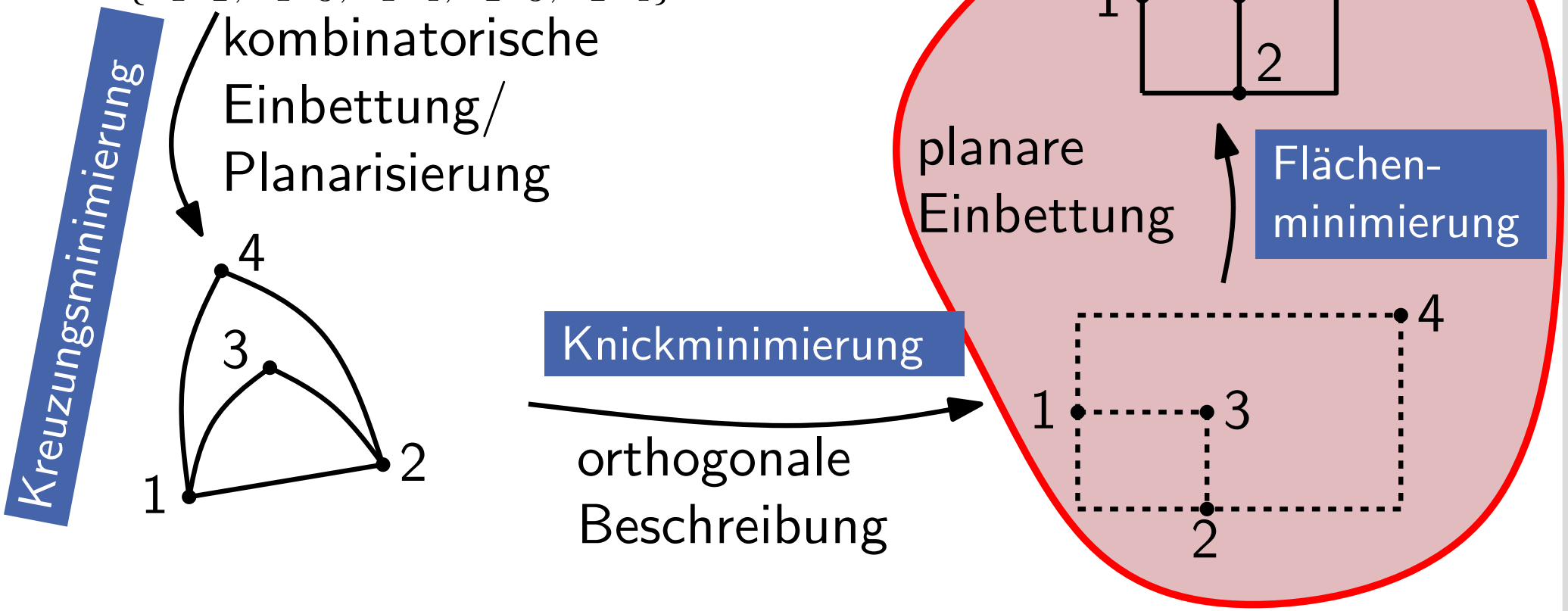
$V = \{v_1, v_2, v_3, v_4\}$
$E = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$



(Planare) Orthogonale Zeichnungen

Dreistufiger Ansatz: *Topology – Shape – Metrics*

$V = \{v_1, v_2, v_3, v_4\}$
$E = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$



Kompaktierung

Problem Kompaktierung

Geg:
- planarer Graph $G = (V, E)$ mit Maximalgrad 4
- orthogonale Beschreibung $H(G)$

Ges: kompaktes orthogonales Layout von G, das $H(G)$ realisiert

Kompaktierung

Karlsruhe Institute of Technology

Problem Kompaktierung

Geg:
- planarer Graph $G = (V, E)$ mit Maximalgrad 4
- orthogonale Beschreibung $H(G)$

Ges: kompaktes orthogonales Layout von G, das $H(G)$ realisiert

Spezialfall: alle Facetten sind Rechtecke

$\rightarrow$ Garantien möglich
- minimale Gesamtkantenlänge
- minimale Fläche

Kompaktierung

Karlsruhe Institute of Technology

Problem Kompaktierung

Geg:
- planarer Graph $G = (V, E)$ mit Maximalgrad 4
- orthogonale Beschreibung $H(G)$

Ges: kompaktes orthogonales Layout von G, das $H(G)$ realisiert

Spezialfall: alle Facetten sind Rechtecke

$\rightarrow$ Garantien möglich
- minimale Gesamtkantenlänge
- minimale Fläche

Eigenschaften:

- Knicke höchstens als Ecken der äußeren Facette
- gegenüberliegende Seiten einer Facette gleich lang

Kompaktierung

Problem Kompaktierung

Geg:
- planarer Graph $G = (V, E)$ mit Maximalgrad 4
- orthogonale Beschreibung $H(G)$

Ges: kompaktes orthogonales Layout von G, das $H(G)$ realisiert

Spezialfall: alle Facetten sind Rechtecke

$\rightarrow$ Garantien möglich
- minimale Gesamtkantenlänge
- minimale Fläche

Eigenschaften:

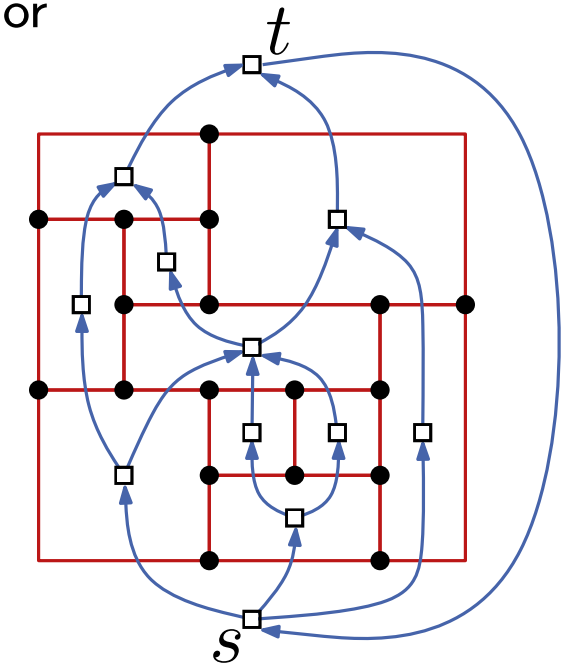
- Knicke höchstens als Ecken der äußeren Facette
- gegenüberliegende Seiten einer Facette gleich lang

Wie geht das mit Flussnetzwerken?

Flussnetzwerk Längenzuweisung

Def: Flussnetzwerk $N_{\text{hor}} = ((W_{\text{hor}}, A_{\text{hor}}); \ell; u; b; \text{cost})$

- $W_{\text{hor}} = \mathcal{F} \setminus \{f_0\} \cup \{s, t\}$
- $A_{\text{hor}} = \{(f, g) \mid f, g$ besitzen gemeinsames horizontales Kantensegment und f liegt unterhalb von $g\} \cup \{(t, s)\}$
- $\ell(a) = 1 \quad \forall a \in A_{\text{hor}}$
- $u(a) = \infty \quad \forall a \in A_{\text{hor}}$
- $\text{cost}(a) = 1 \quad \forall a \in A_{\text{hor}}$
- $b(f) = 0 \quad \forall f \in W_{\text{hor}}$

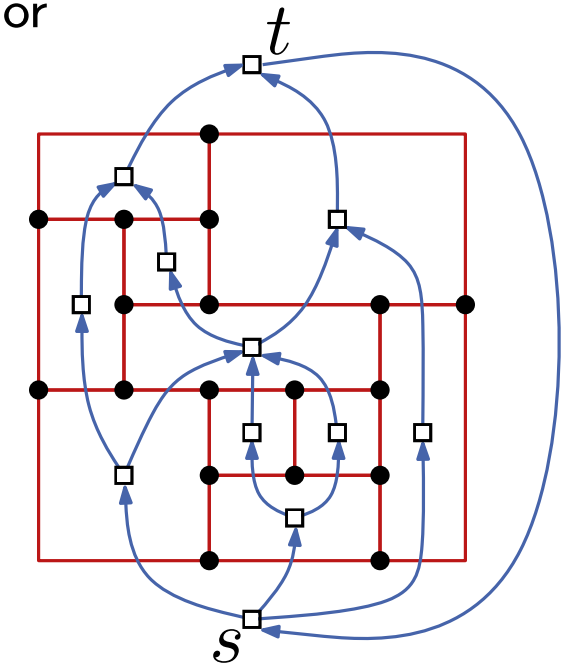


Flussnetzwerk Längenzuweisung

Def: Flussnetzwerk $N_{\text{hor}} = ((W_{\text{hor}}, A_{\text{hor}}); \ell; u; b; \text{cost})$

- $W_{\text{hor}} = \mathcal{F} \setminus \{f_0\} \cup \{s, t\}$
- $A_{\text{hor}} = \{(f, g) \mid f, g$ besitzen gemeinsames horizontales Kantensegment und f liegt unterhalb von $g\} \cup \{(t, s)\}$
- $\ell(a) = 1 \quad \forall a \in A_{\text{hor}}$
- $u(a) = \infty \quad \forall a \in A_{\text{hor}}$
- $\text{cost}(a) = 1 \quad \forall a \in A_{\text{hor}}$
- $b(f) = 0 \quad \forall f \in W_{\text{hor}}$

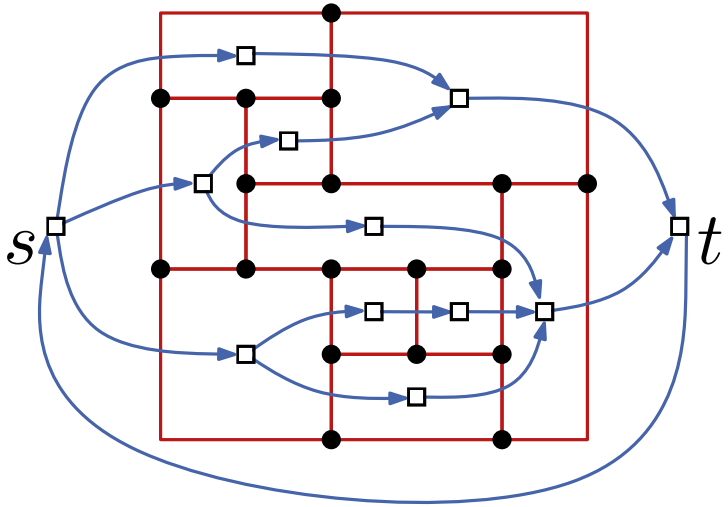
s und t repräsentieren untere und obere Hälfte von f_0



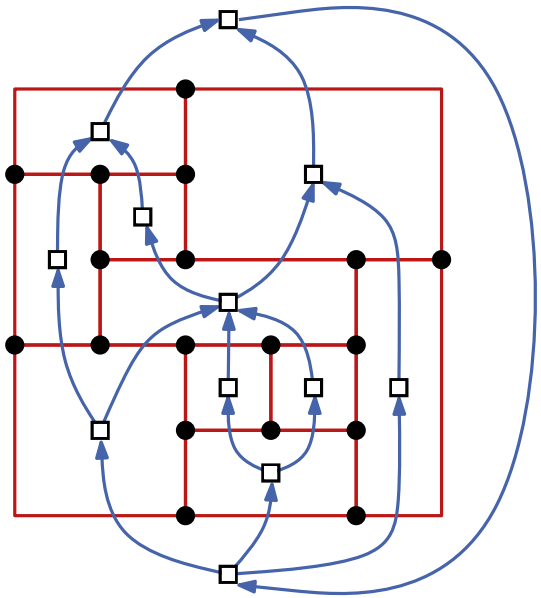
Flussnetzwerk Längenzuweisung

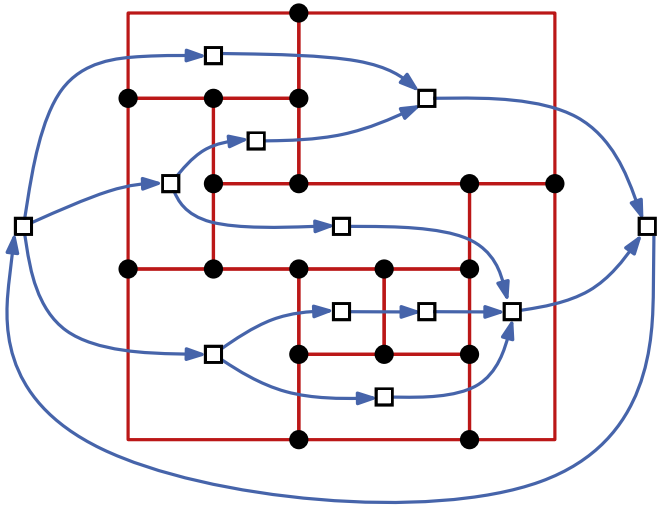
Def: Flussnetzwerk $N_{\text{ver}} = ((W_{\text{ver}}, A_{\text{ver}}); \ell; u; b; \text{cost})$

- $W_{\text{ver}} = \mathcal{F} \setminus \{f_0\} \cup \{s, t\}$
- $A_{\text{ver}} = \{(f, g) \mid f, g$ besitzen gemeinsames vertikales Kantensegment und f liegt links von $g\} \cup \{(t, s)\}$
- $\ell(a) = 1 \quad \forall a \in A_{\text{hor}}$
- $u(a) = \infty \quad \forall a \in A_{\text{hor}}$
- $\text{cost}(a) = 1 \quad \forall a \in A_{\text{hor}}$
- $b(f) = 0 \quad \forall f \in W_{\text{hor}}$



Optimales Layout

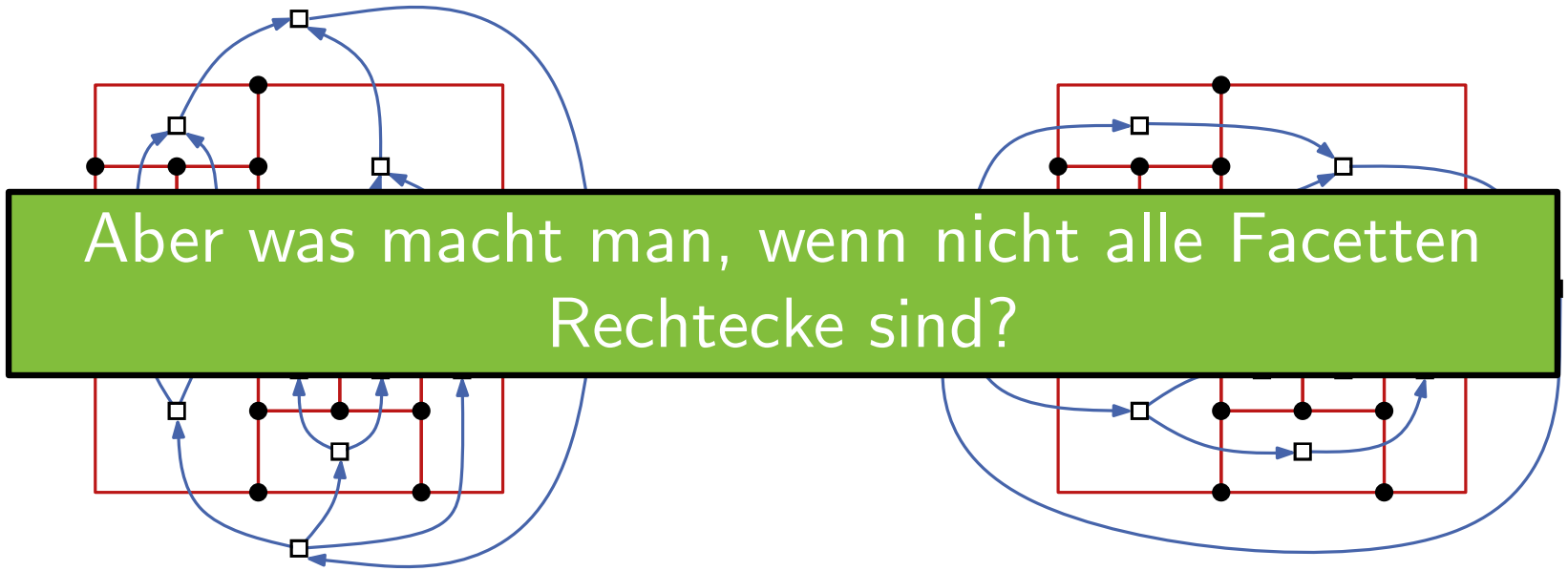




Satz 1: Ganzzahlige Flüsse x_{hor} und x_{ver} in N_{hor} und N_{ver} mit minimalen Kosten liefern:

- zugeh. Seitenlängen induzieren gültiges Layout
- $|x_{\text{hor}}(t,s)|$ und $|x_{\text{ver}}(t,s)|$ entsprechen minimaler Breite und Höhe des Layouts
- $\sum_{a \in A_{\text{hor}}} x_{\text{hor}}(a) \cdot \text{cost}(a) + \sum_{a \in A_{\text{ver}}} x_{\text{ver}}(a) \cdot \text{cost}(a)$ entsprechen minimaler Gesamtkantenlänge

Optimales Layout

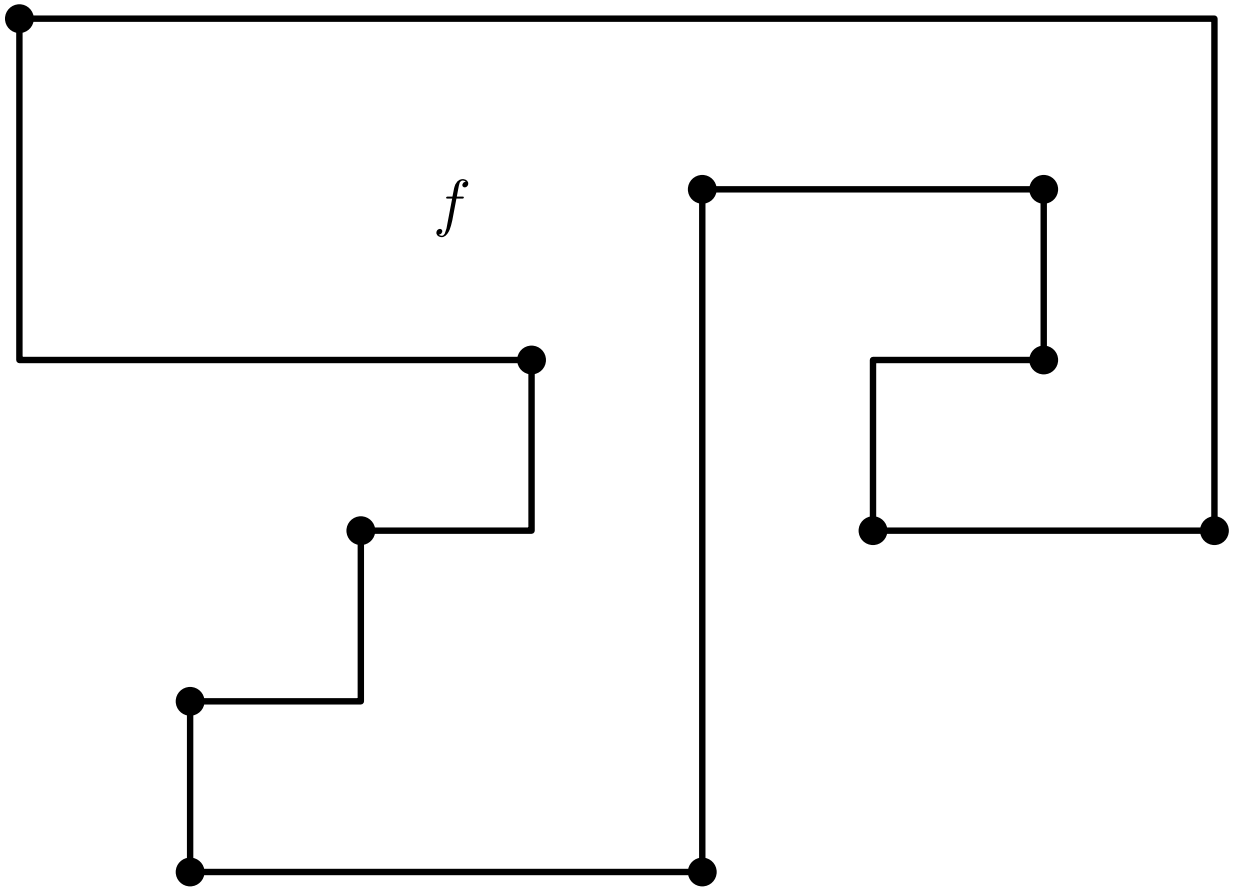


Aber was macht man, wenn nicht alle Facetten Rechtecke sind?

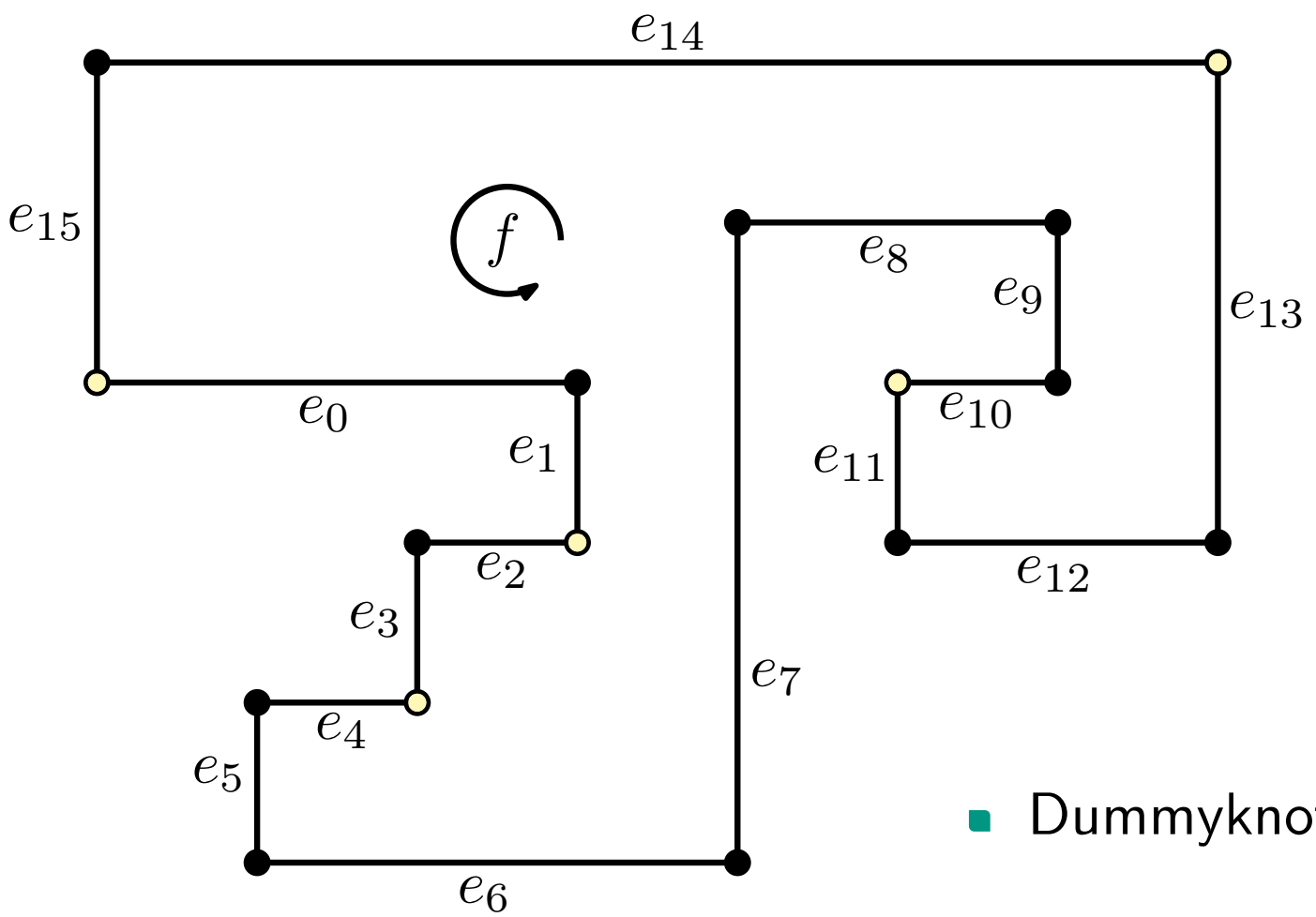
Satz 1: Ganzzahlige Flüsse x_{hor} und x_{ver} in N_{hor} und N_{ver} mit minimalen Kosten liefern:

- zugeh. Seitenlängen induzieren gültiges Layout
- $|x_{\text{hor}}(t,s)|$ und $|x_{\text{ver}}(t,s)|$ entsprechen minimaler Breite und Höhe des Layouts
- $\sum_{a \in A_{\text{hor}}} x_{\text{hor}}(a) \cdot \text{cost}(a) + \sum_{a \in A_{\text{ver}}} x_{\text{ver}}(a) \cdot \text{cost}(a)$ entsprechen minimaler Gesamtkantenlänge

Verfeinerung von (G, H) – innere Facette

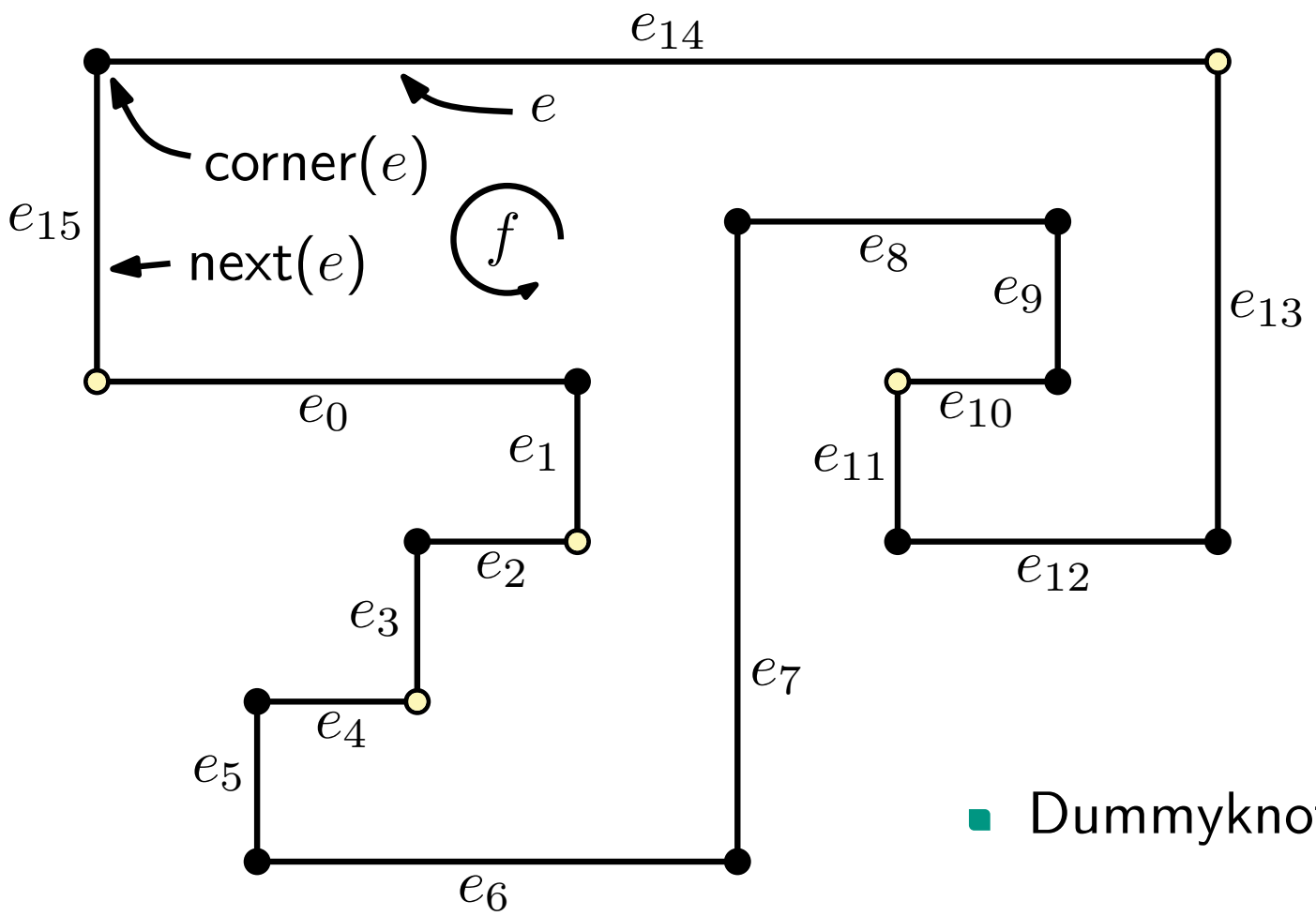


Verfeinerung von (G, H) – innere Facette



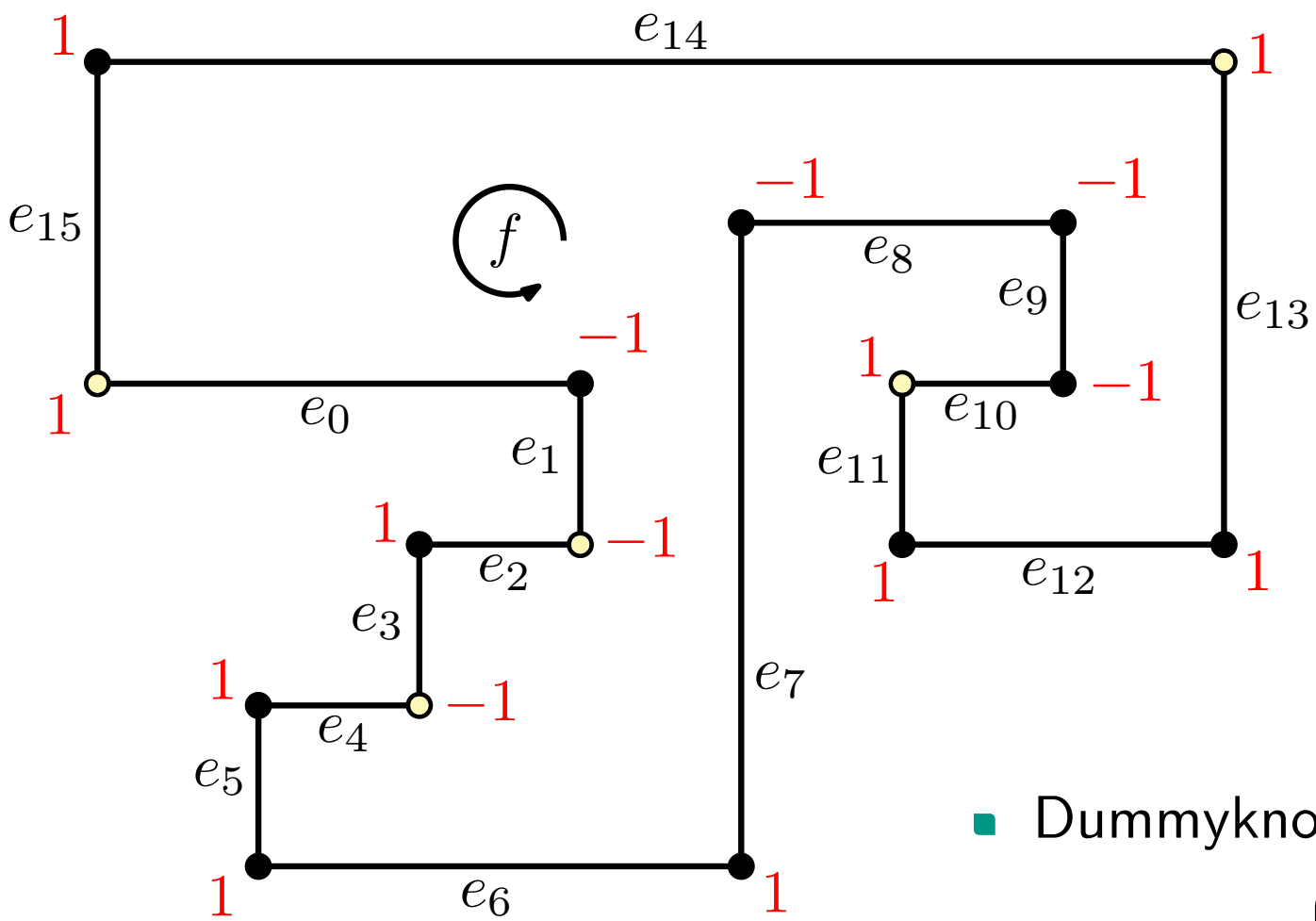
- Dummyknoten für Knicke

Verfeinerung von (G, H) – innere Facette



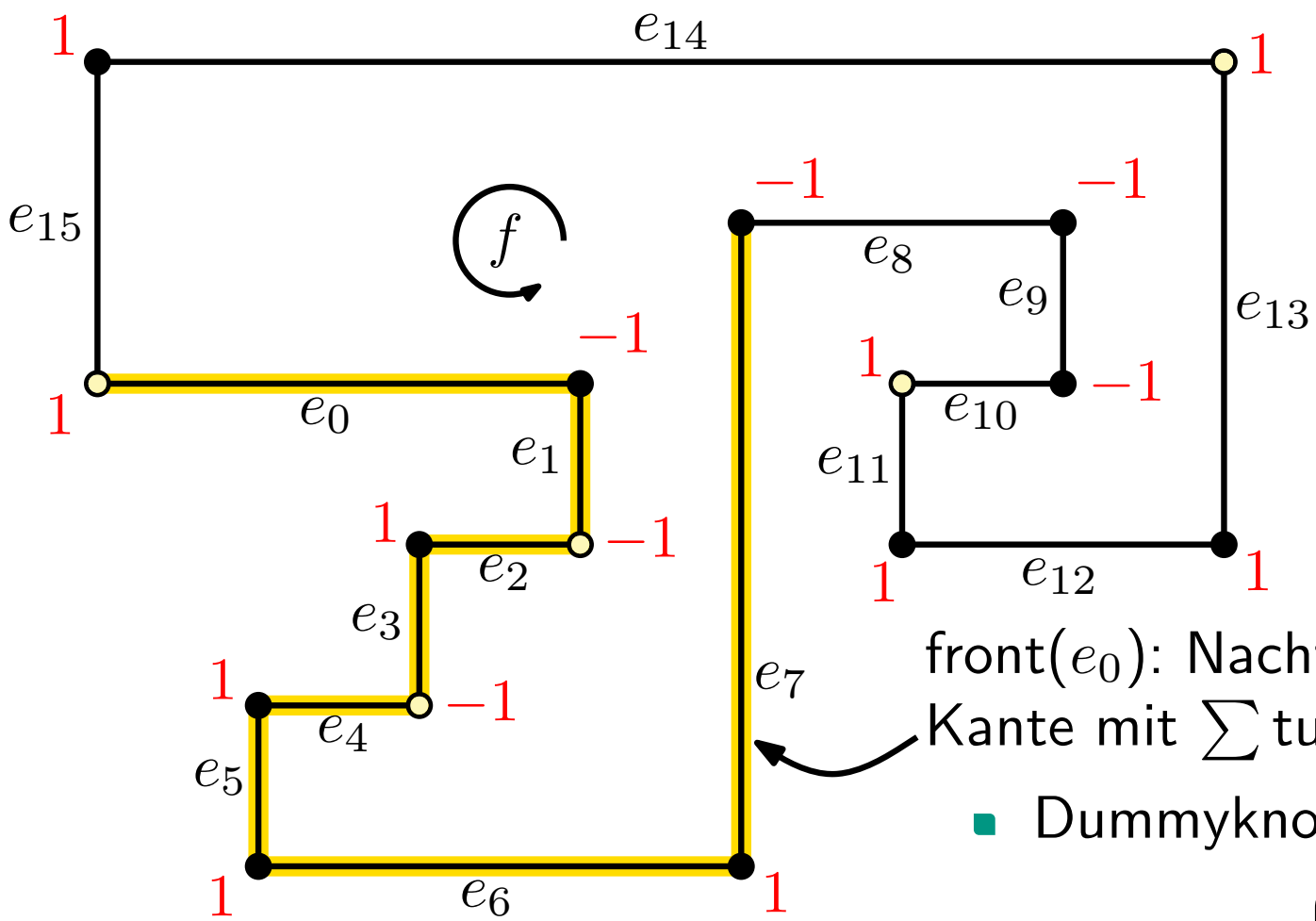
- Dummyknoten für Knicke

Verfeinerung von (G, H) – innere Facette



- Dummyknoten für Knicke
- $\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$

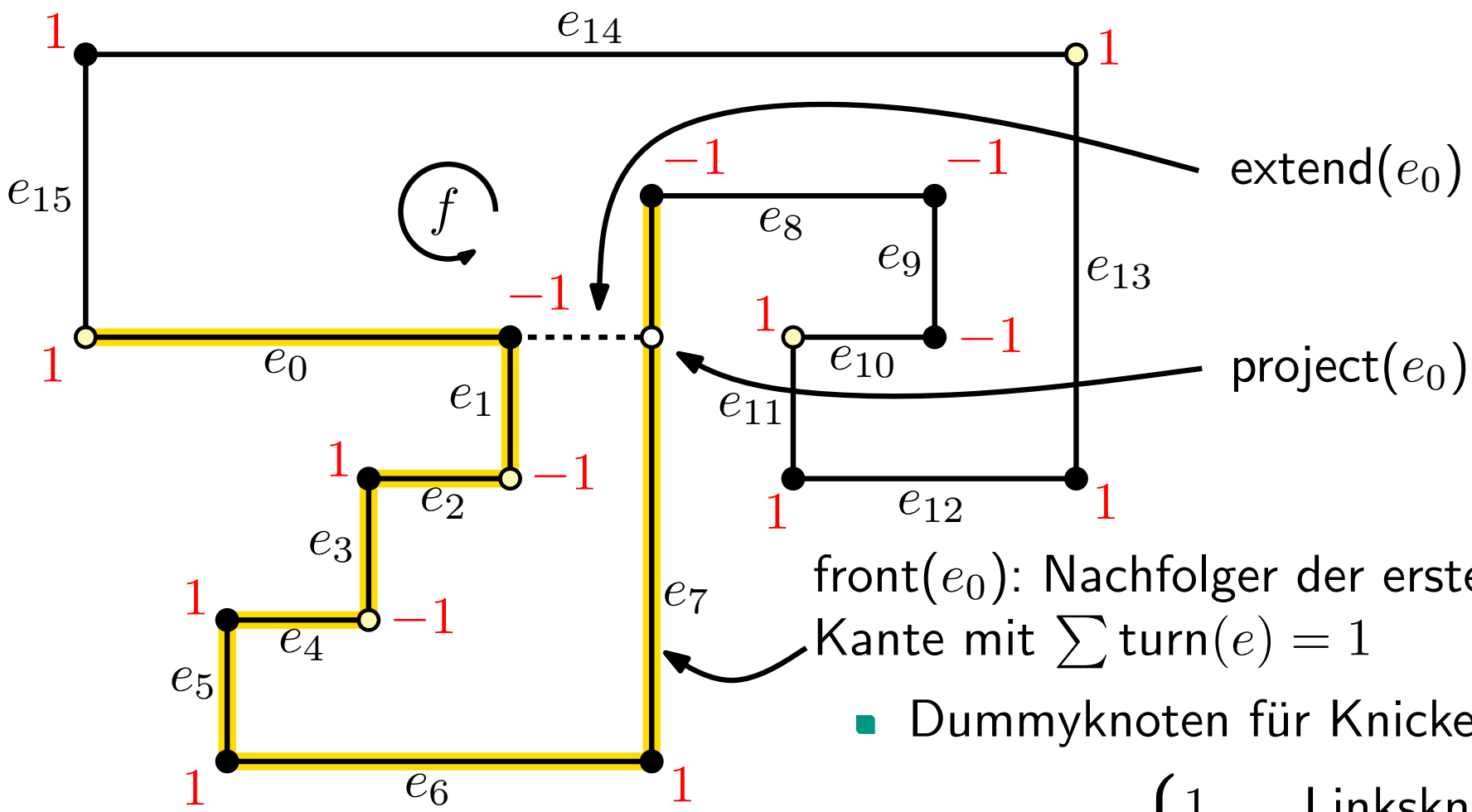
Verfeinerung von (G, H) – innere Facette



front(e_0): Nachfolger der ersten Kante mit $\sum \text{turn}(e) = 1$

- Dummyknoten für Knicke
- $\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$

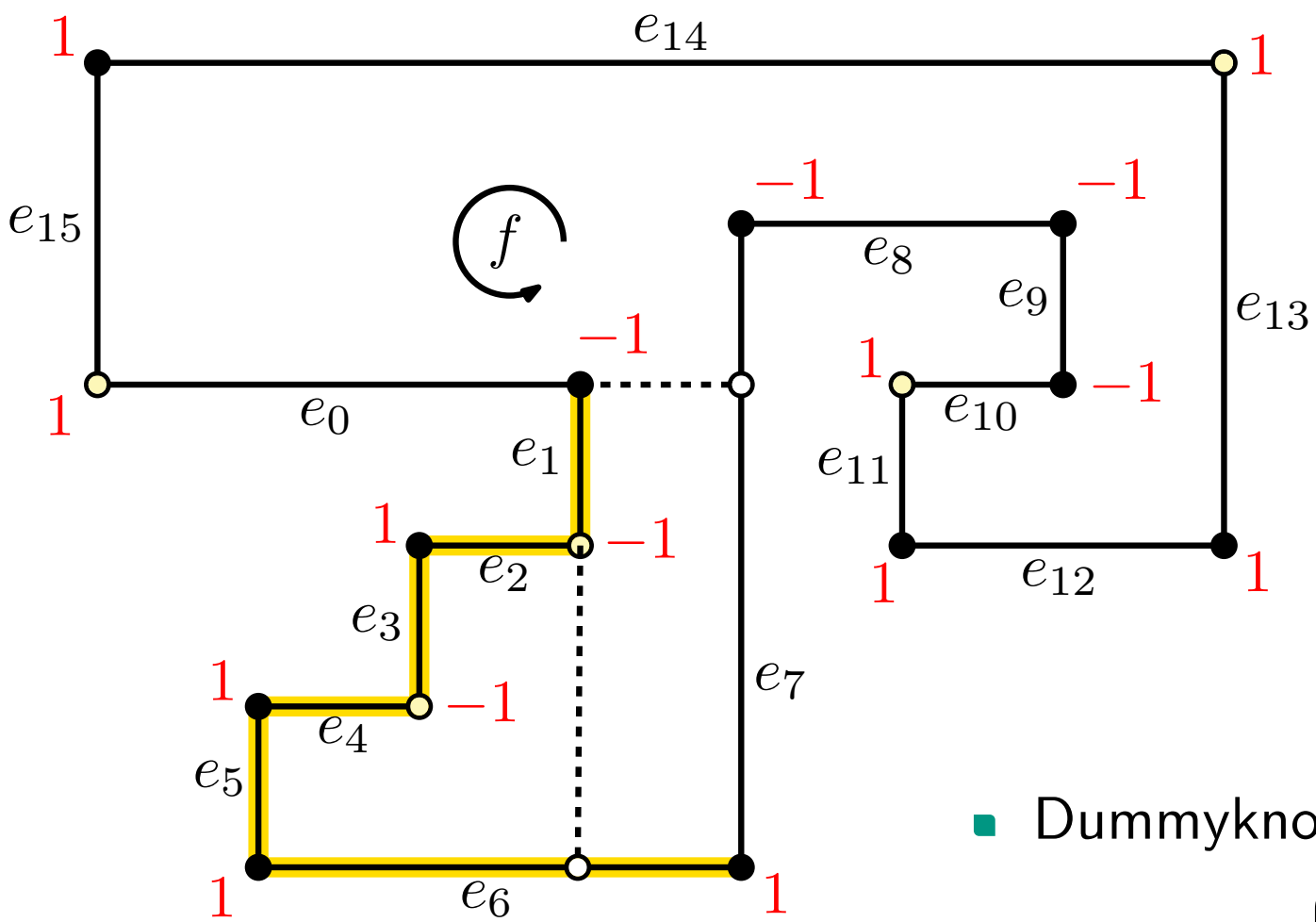
Verfeinerung von (G, H) – innere Facette



front(e_0): Nachfolger der ersten Kante mit $\sum \text{turn}(e) = 1$

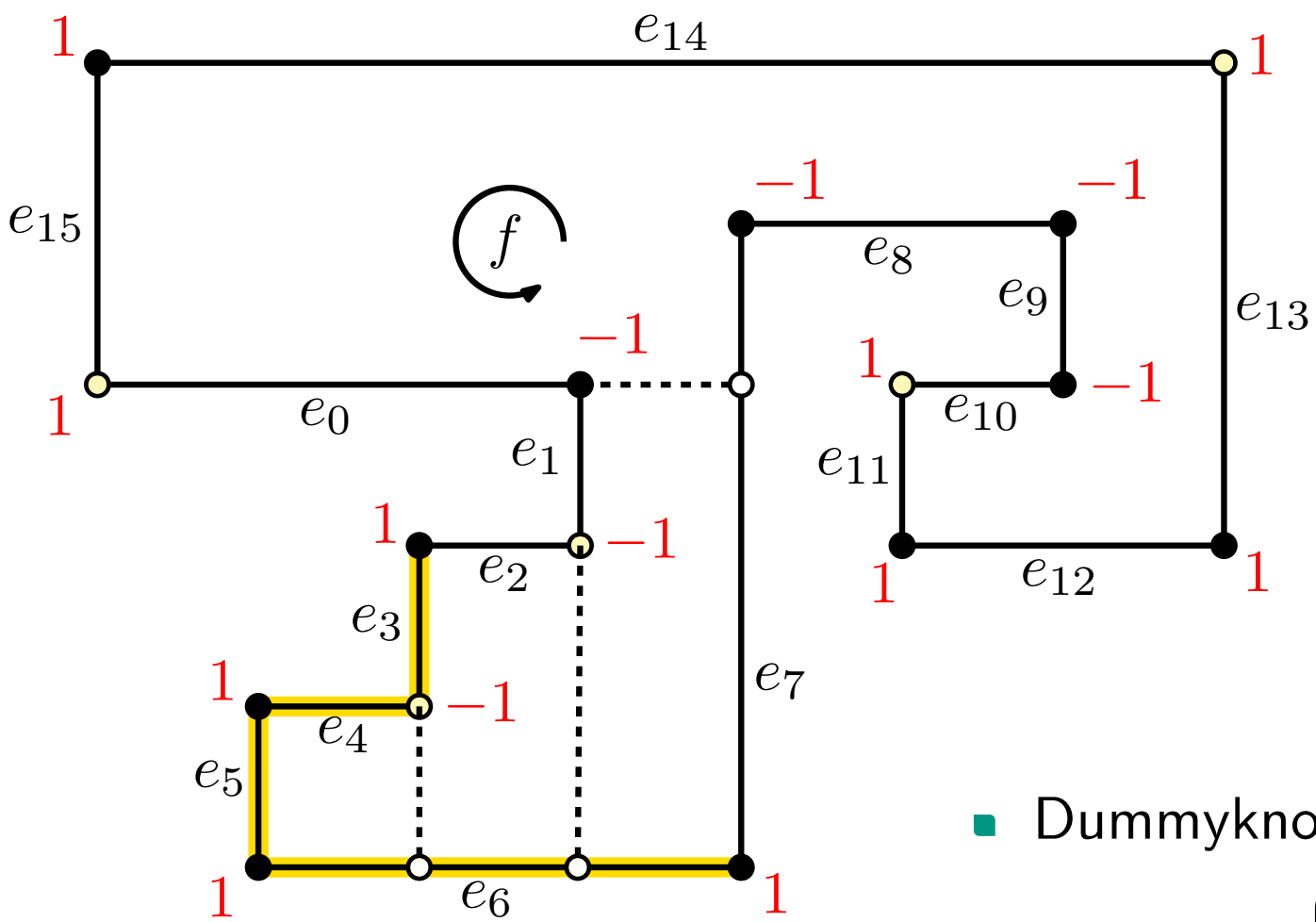
- Dummyknoten für Knicke
- $\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$

Verfeinerung von (G, H) – innere Facette



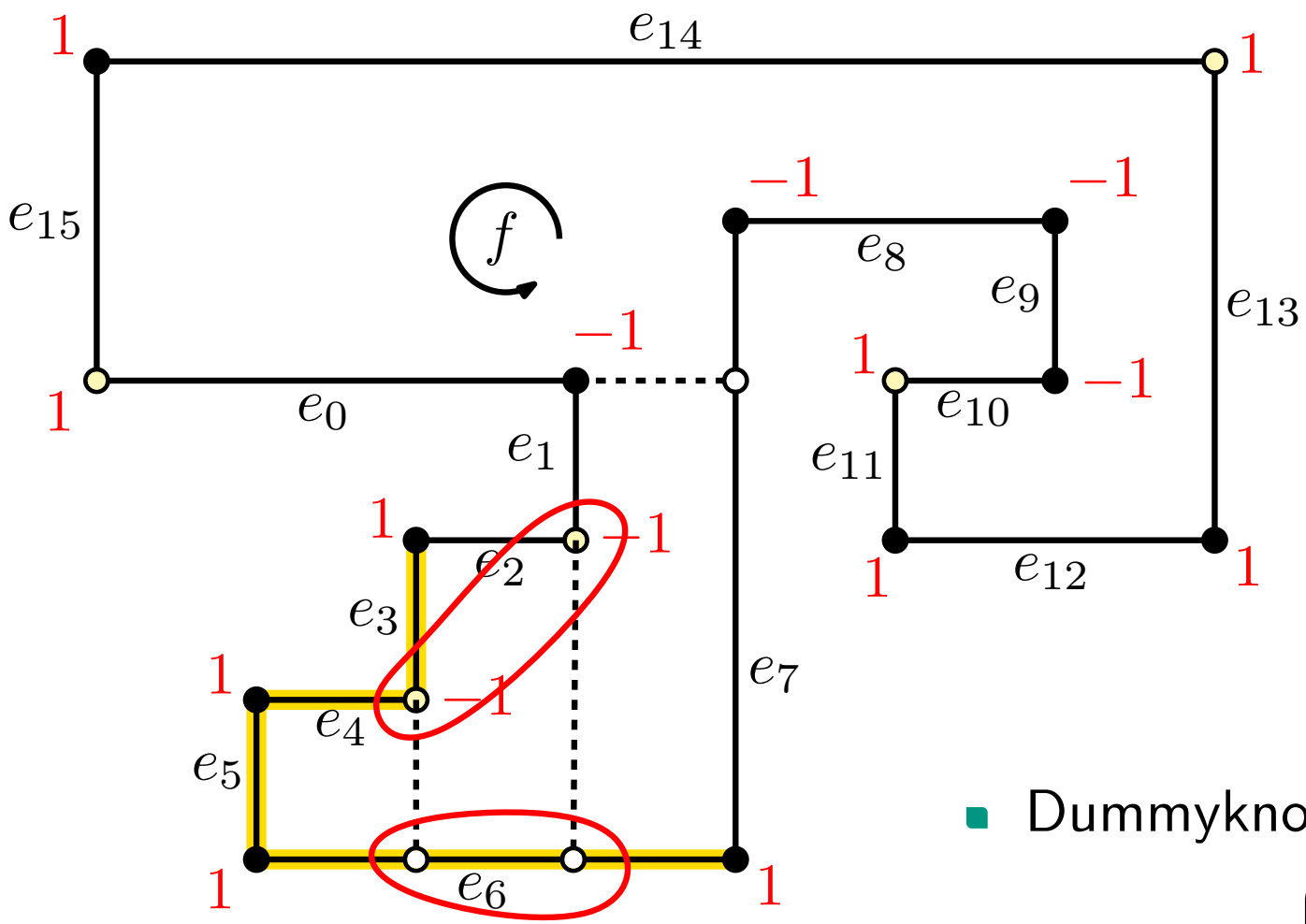
- Dummyknoten für Knicke
- $\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$

Verfeinerung von (G, H) – innere Facette



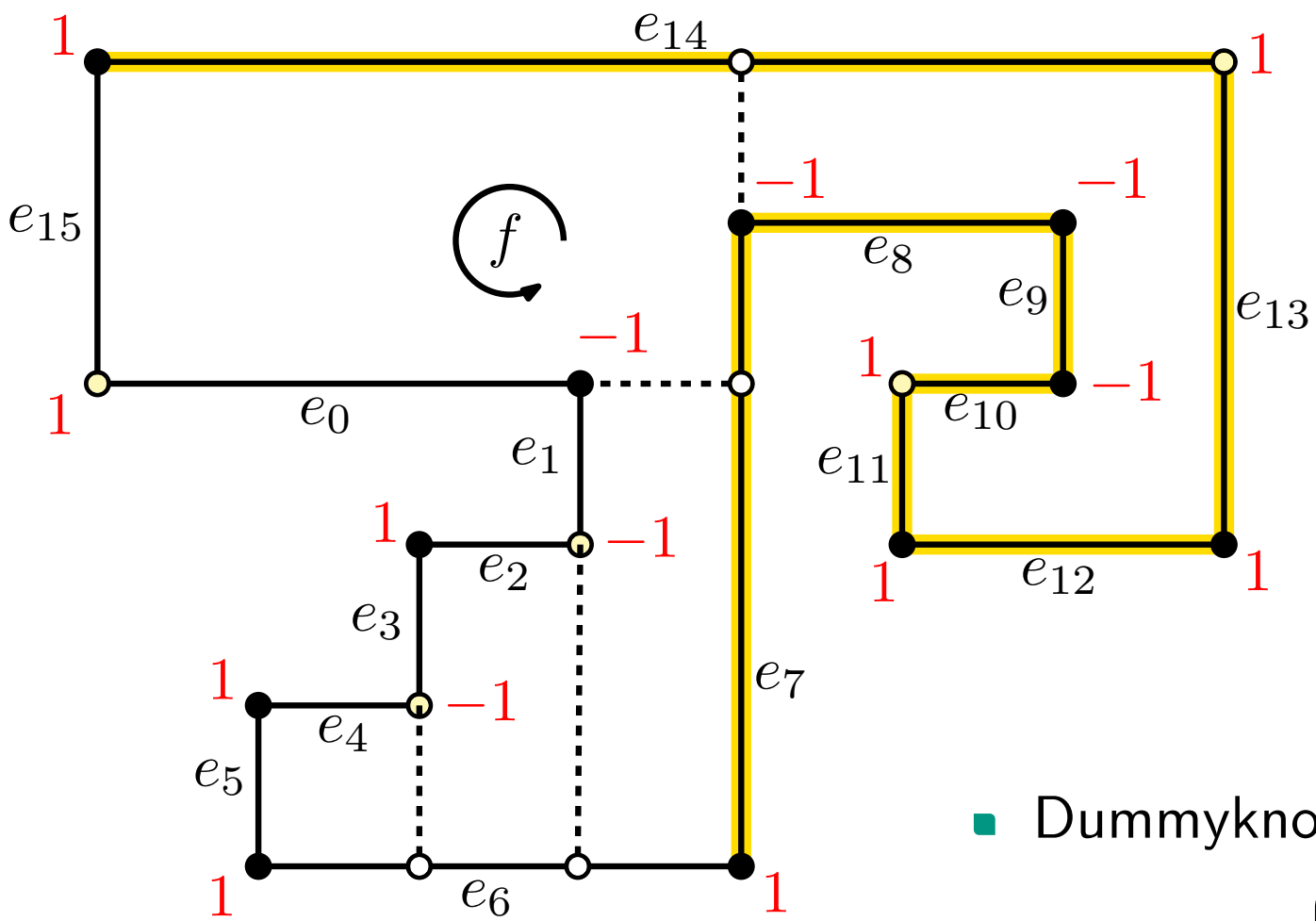
- Dummyknoten für Knicke
- $\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$

Verfeinerung von (G, H) – innere Facette



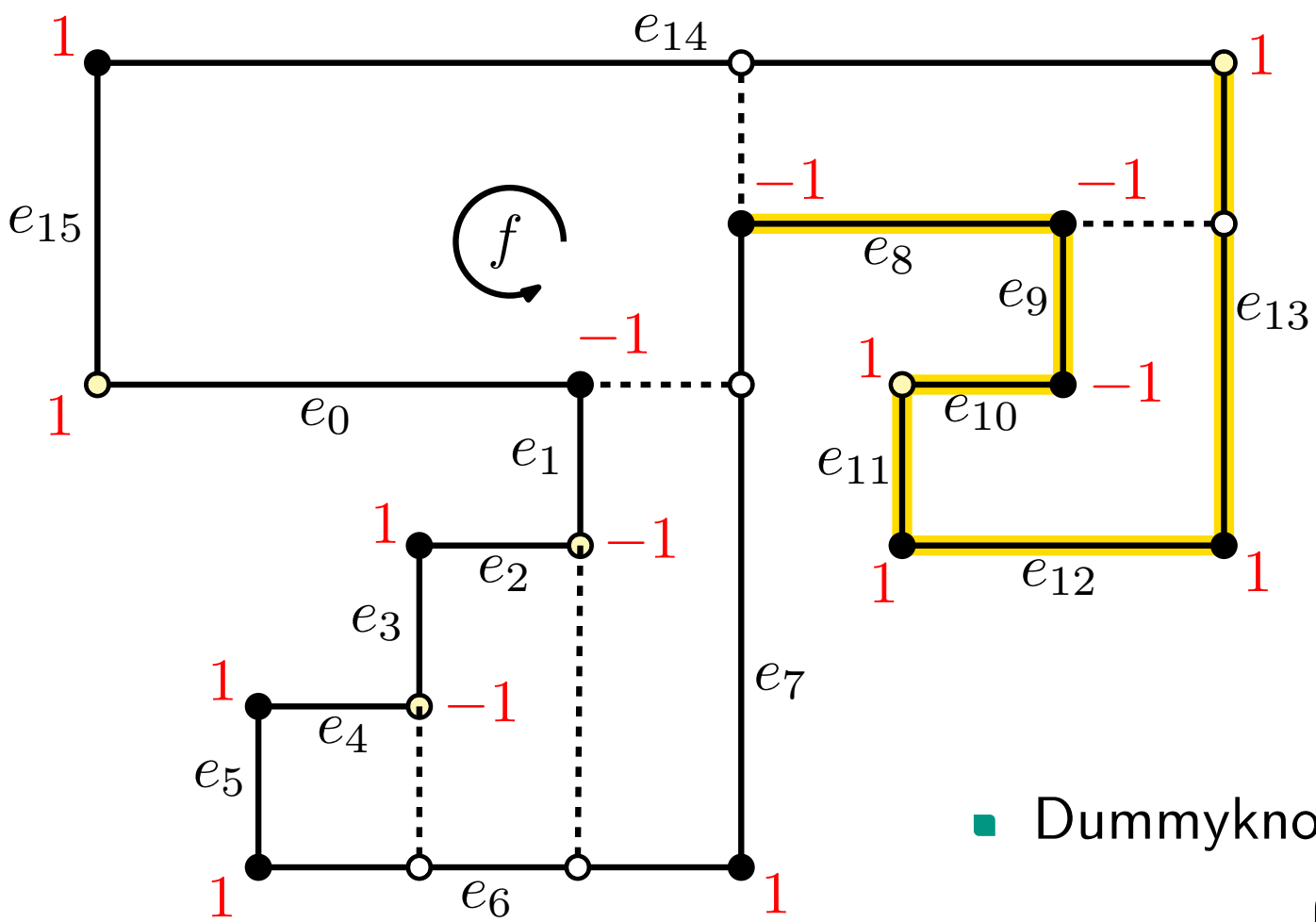
- Dummyknoten für Knicke
- $\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$

Verfeinerung von (G, H) – innere Facette



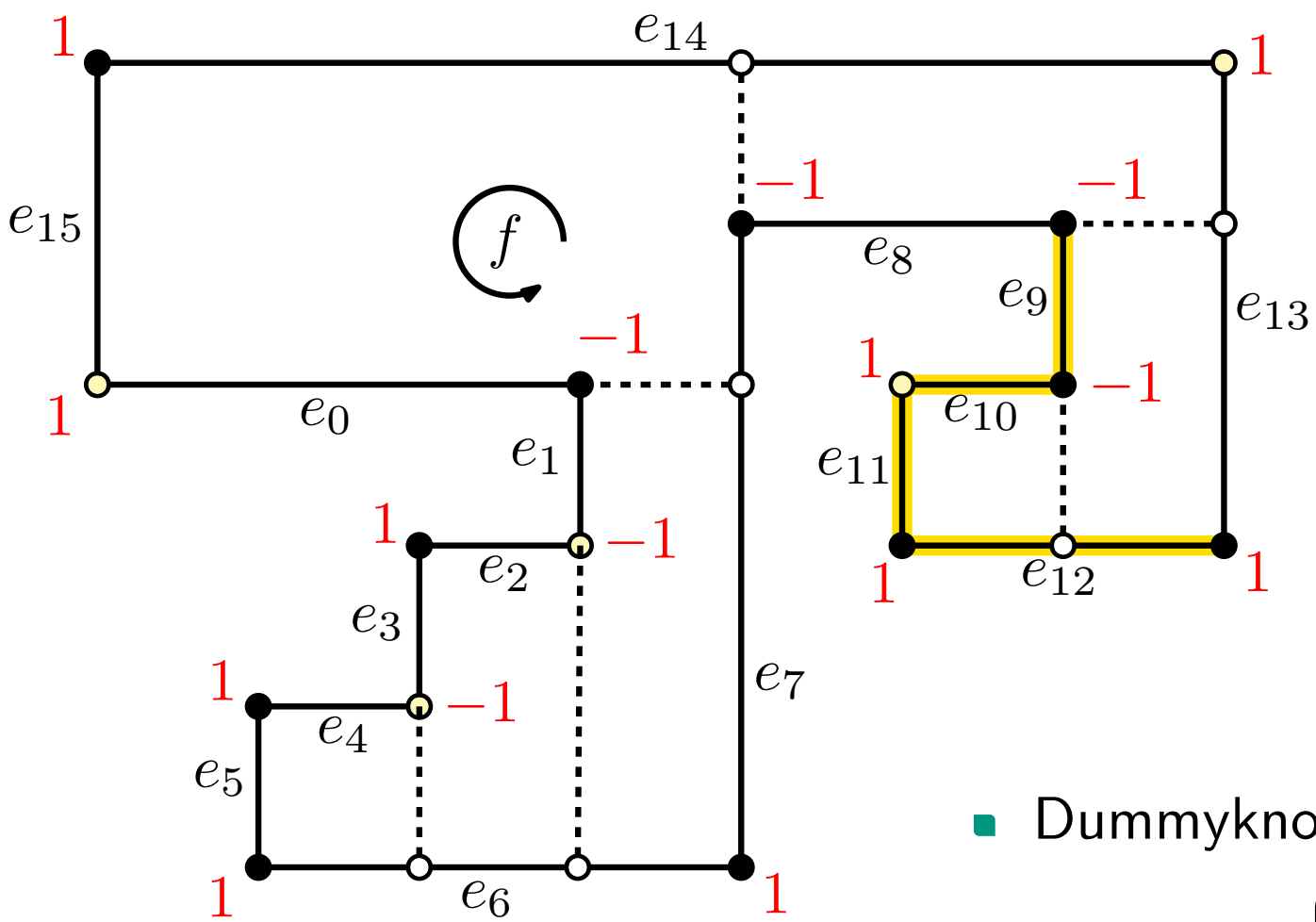
- Dummyknoten für Knicke
- $\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$

Verfeinerung von (G, H) – innere Facette



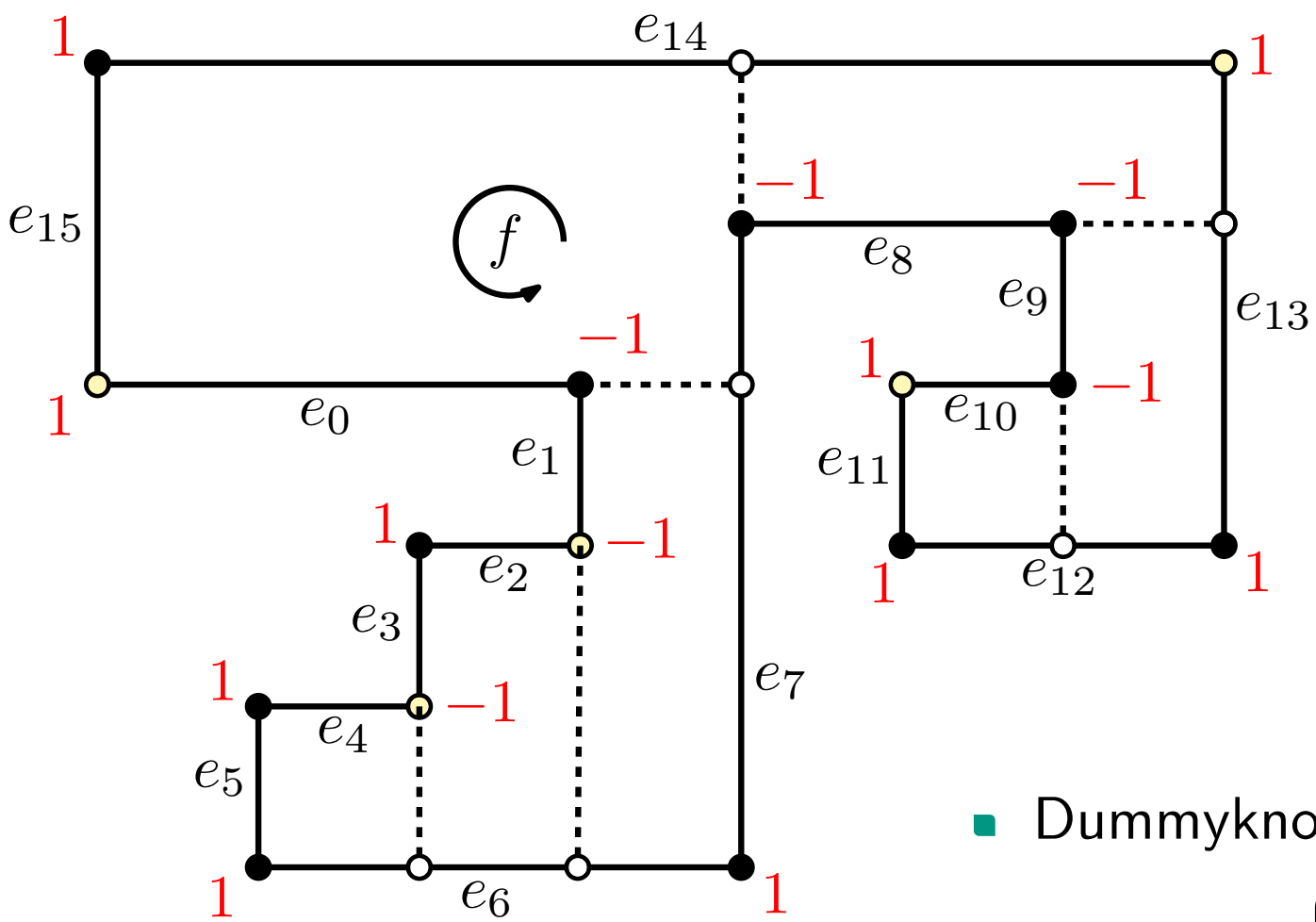
- Dummyknoten für Knicke
- $\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$

Verfeinerung von (G, H) – innere Facette



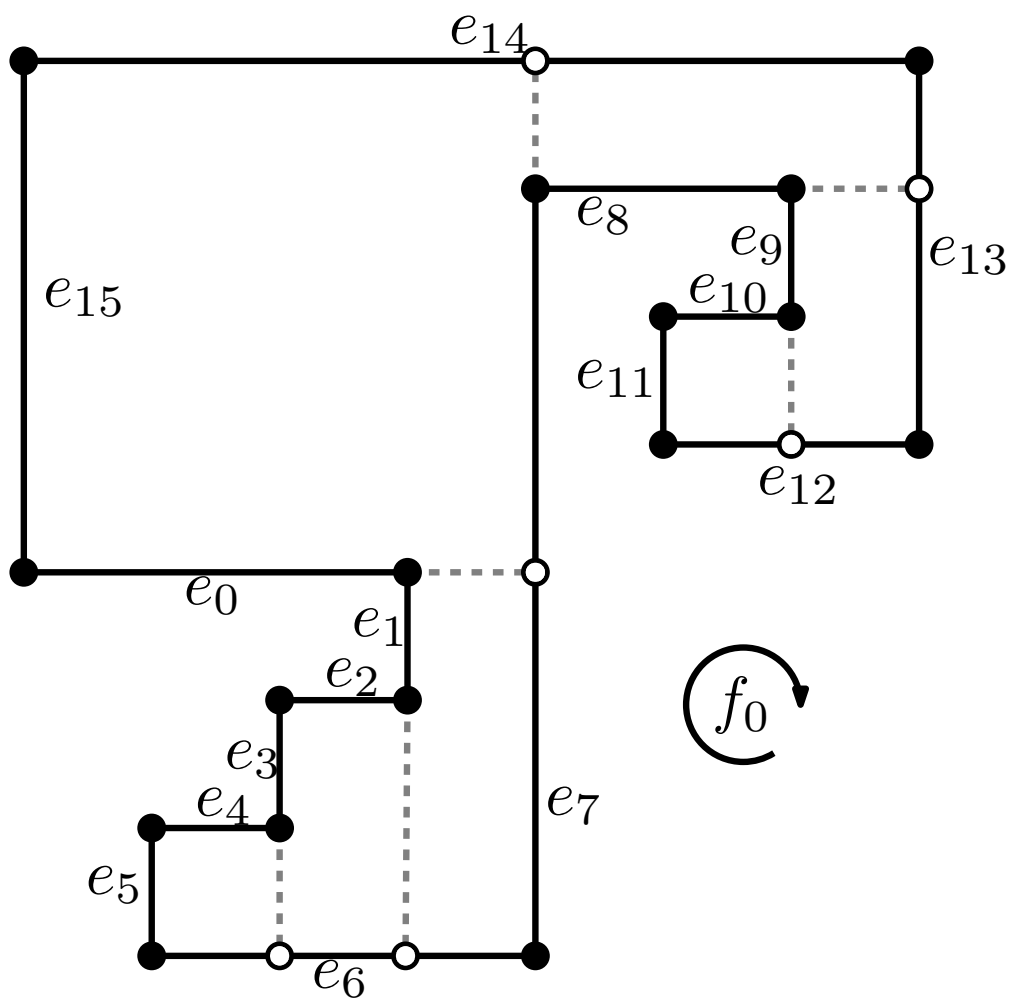
- Dummyknoten für Knicke
- $\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$

Verfeinerung von (G, H) – innere Facette

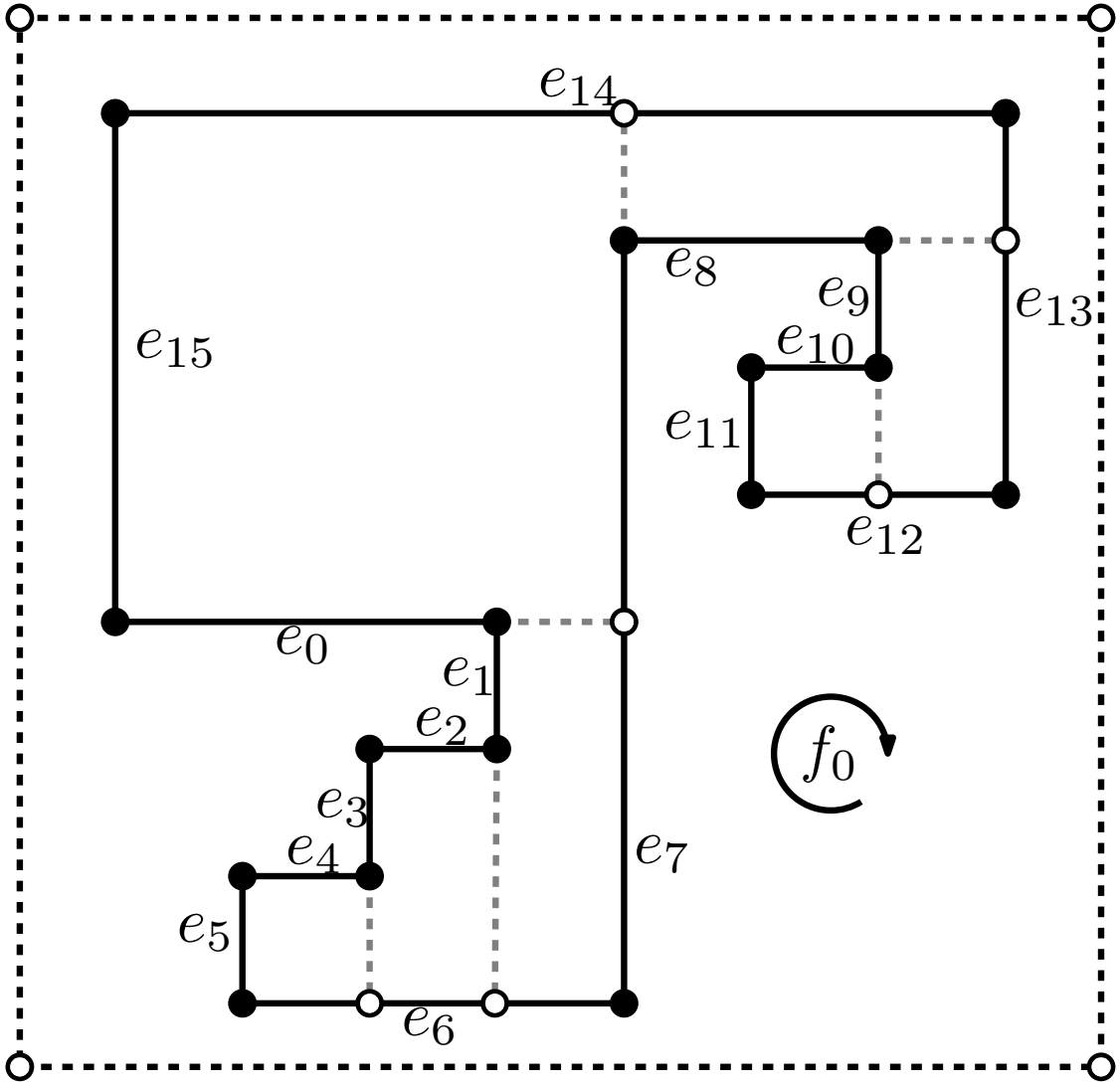


- Dummyknoten für Knicke
- $\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$

Verfeinerung von (G, H) – äußere Facette

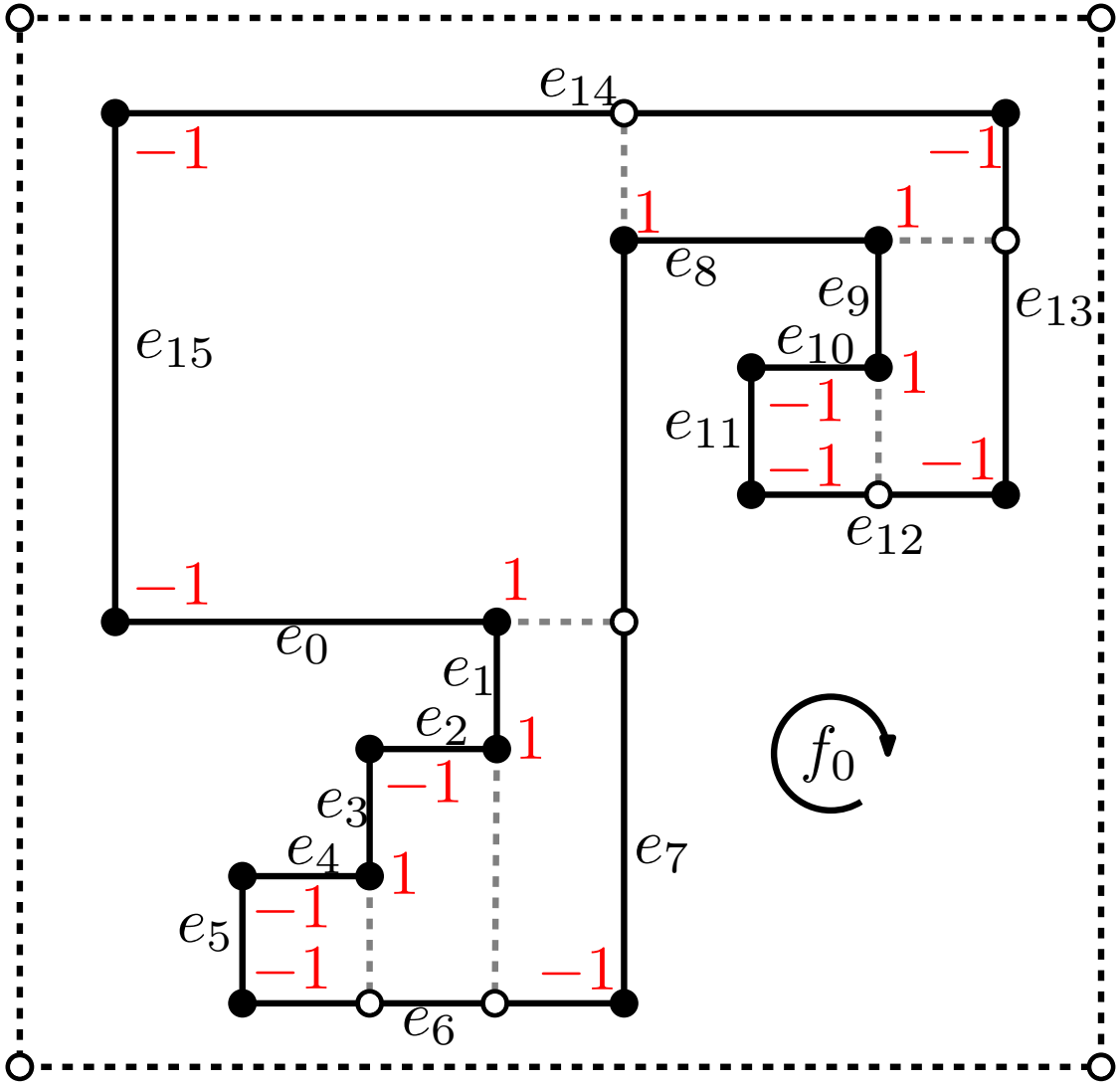


Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen

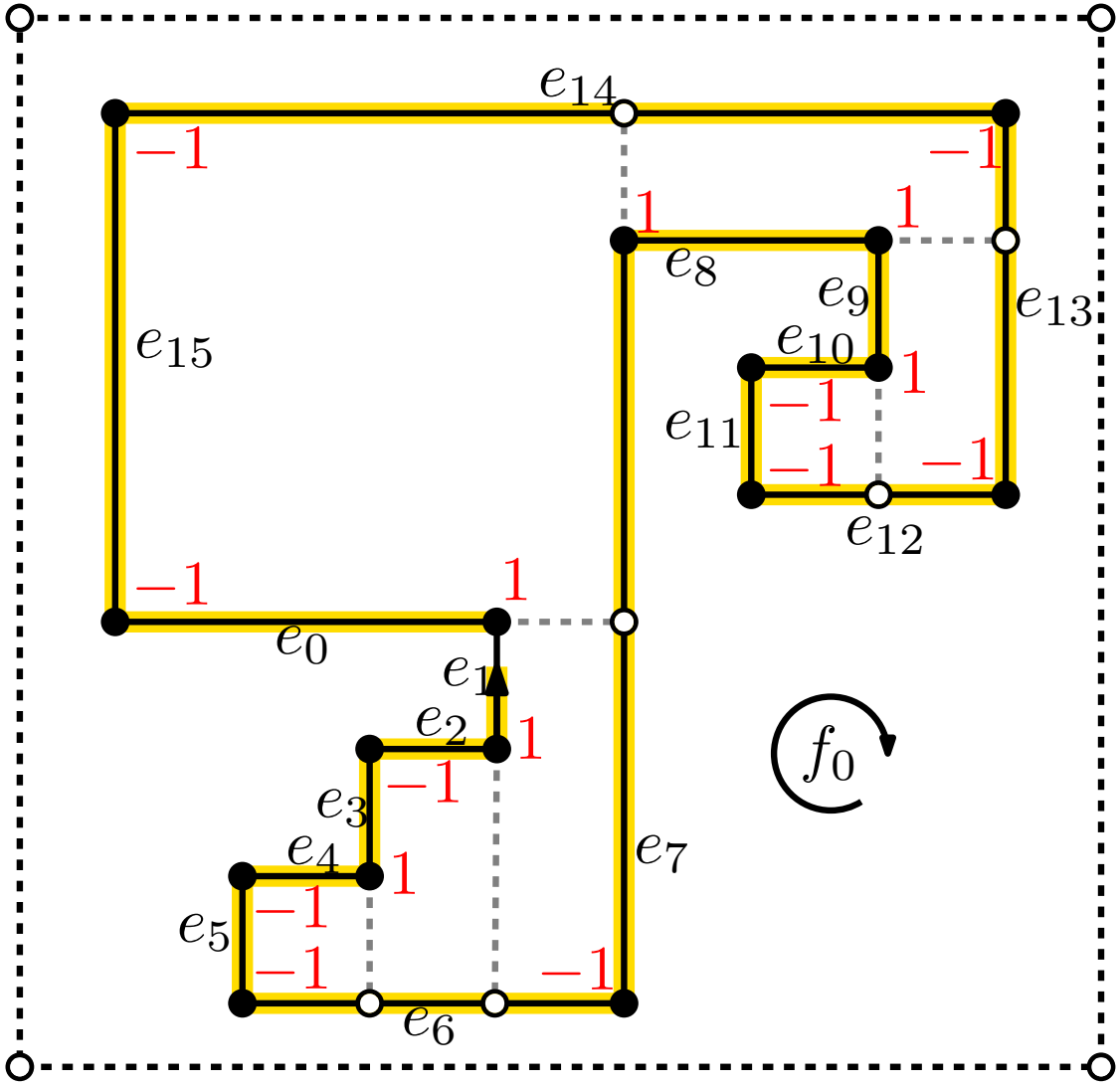
Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen

$$\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$$

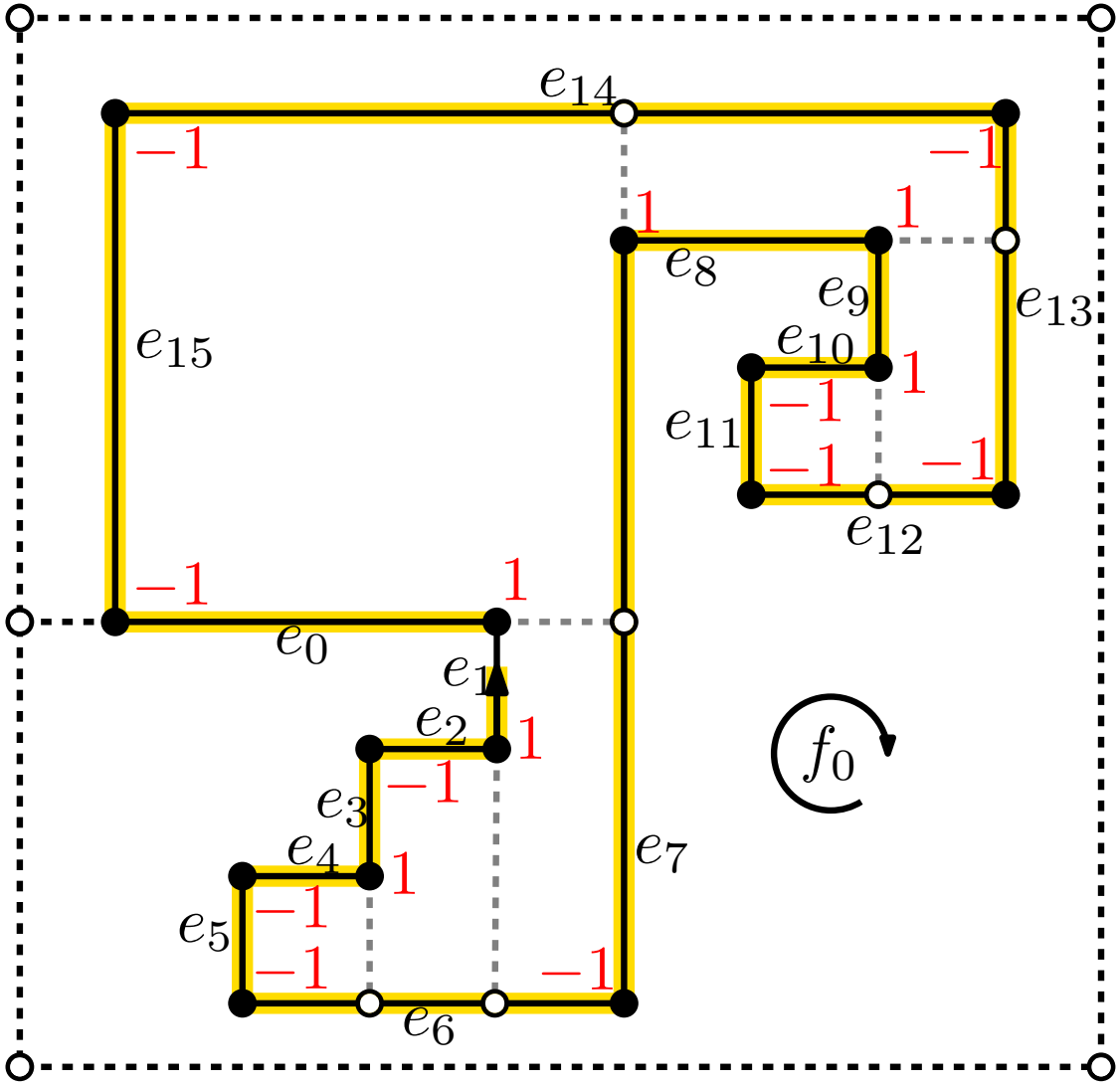
Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen

$$\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$$

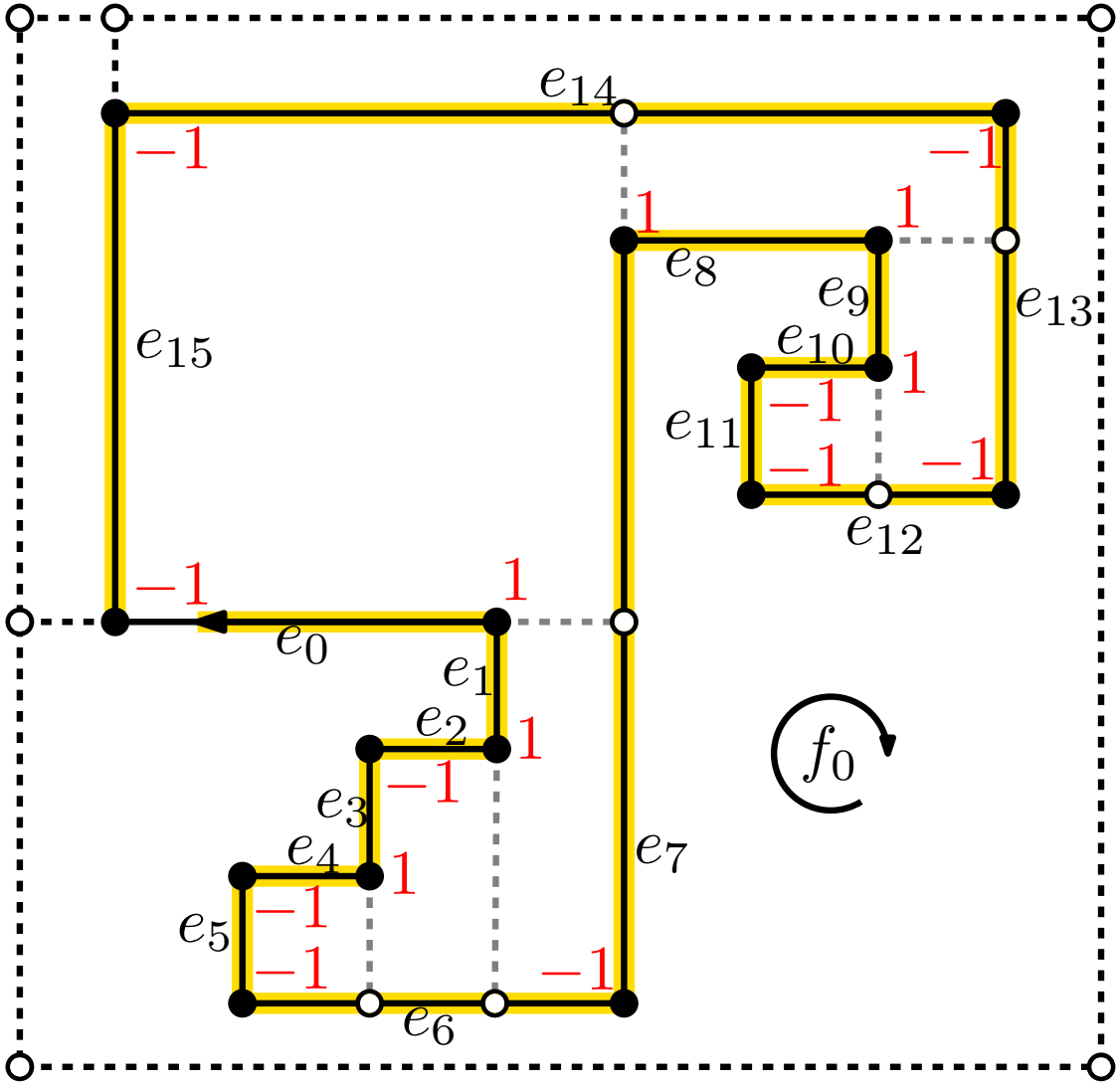
Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen
- falls $\sum \text{turn}(e) < 1$ für komplette Umrundung von f_0, projiziere auf R

$$\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$$

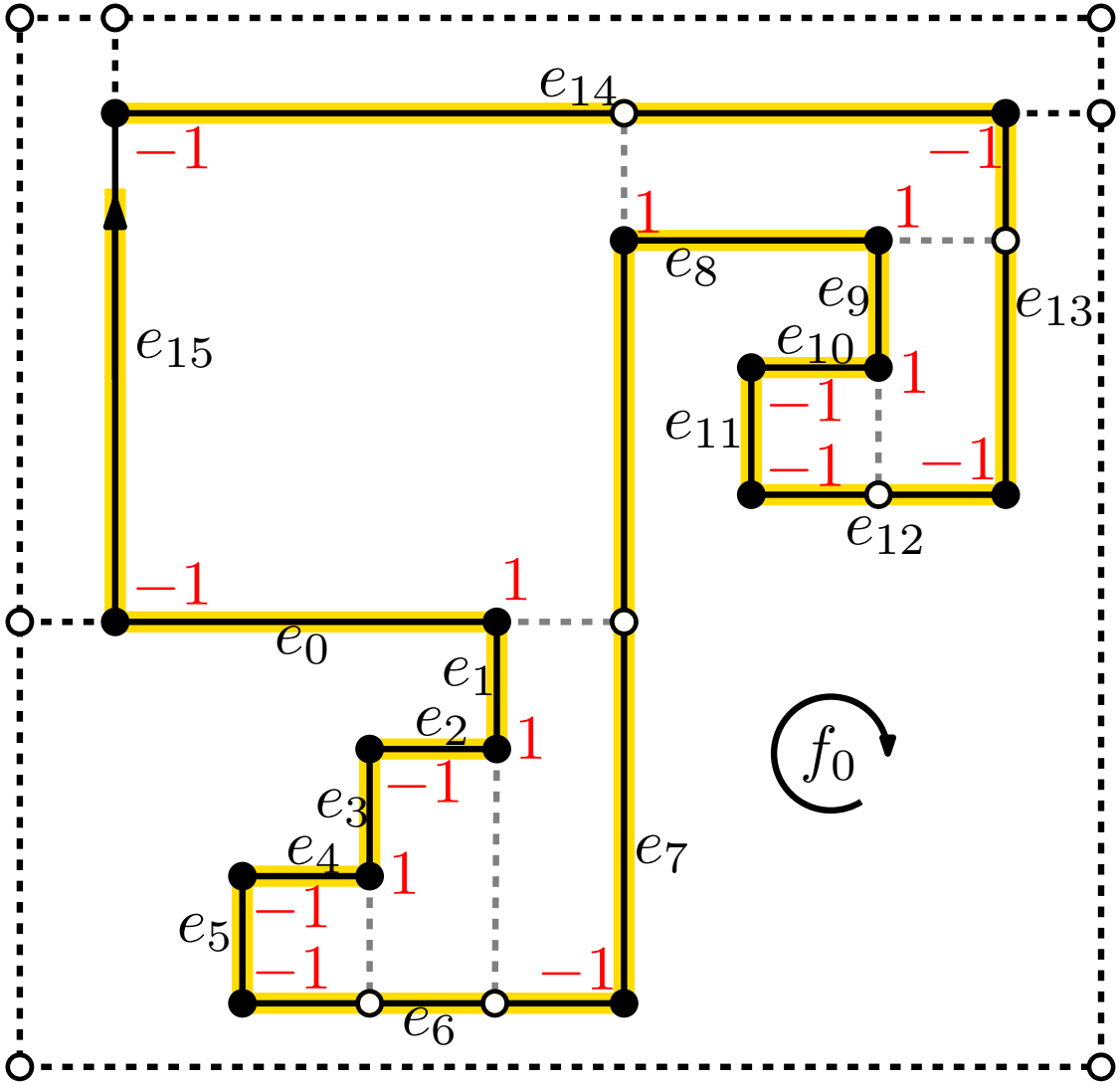
Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen
- falls $\sum \text{turn}(e) < 1$ für komplette Umrundung von f_0, projiziere auf R

$$\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$$

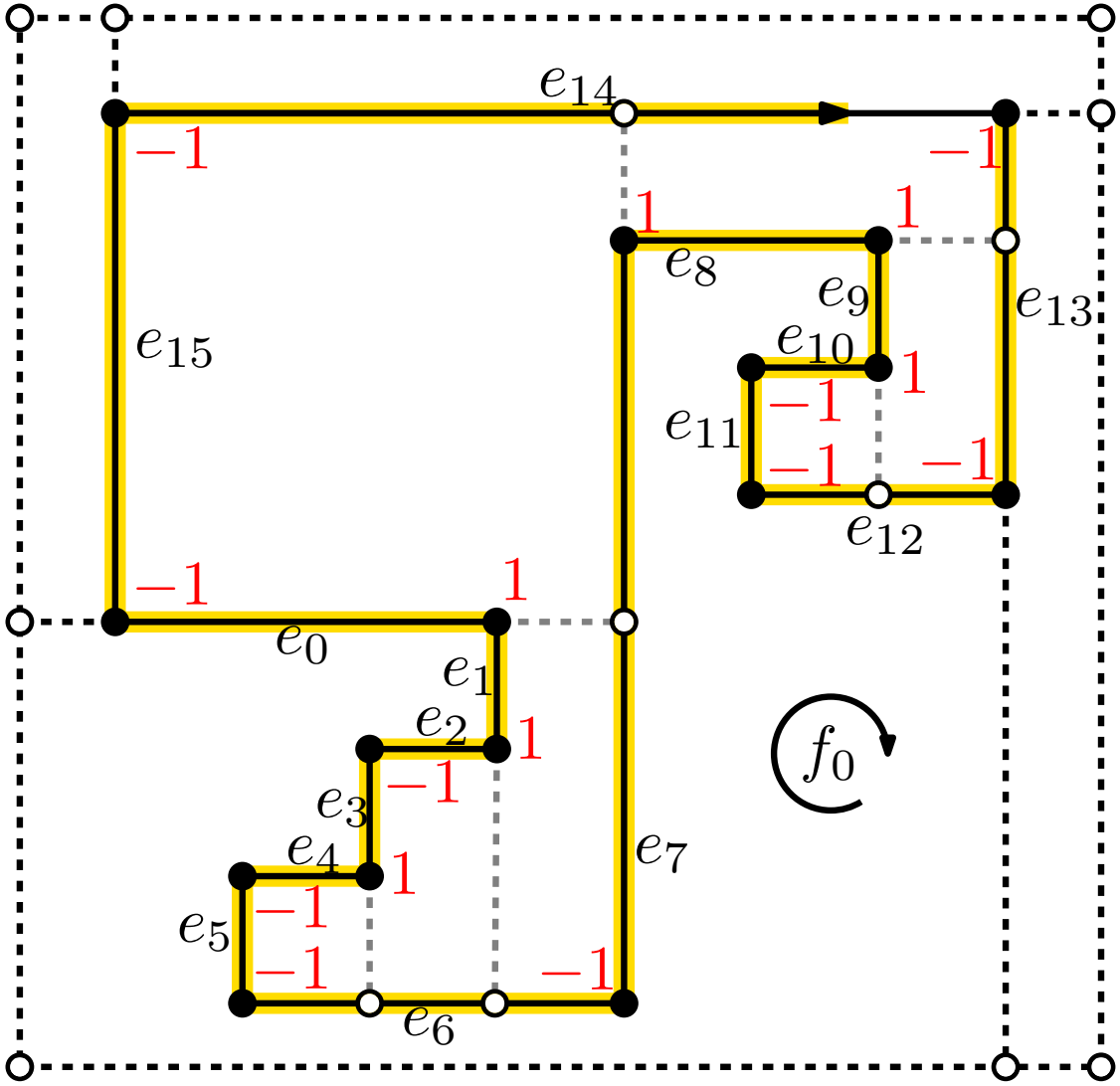
Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen
- falls $\sum \text{turn}(e) < 1$ für komplette Umrundung von f_0, projiziere auf R

$$\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$$

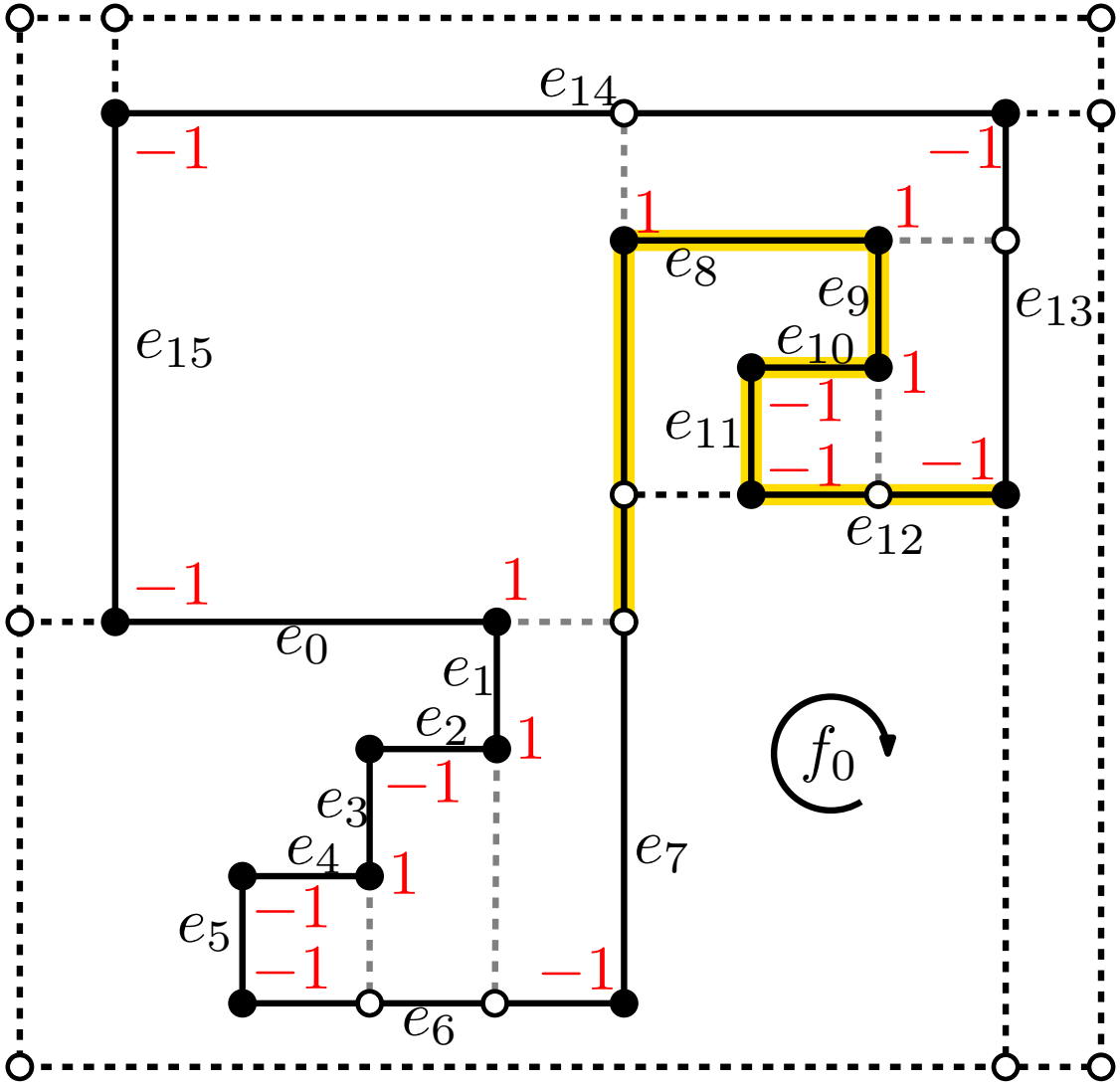
Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen
- falls $\sum \text{turn}(e) < 1$ für komplette Umrundung von f_0, projiziere auf R

$$\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$$

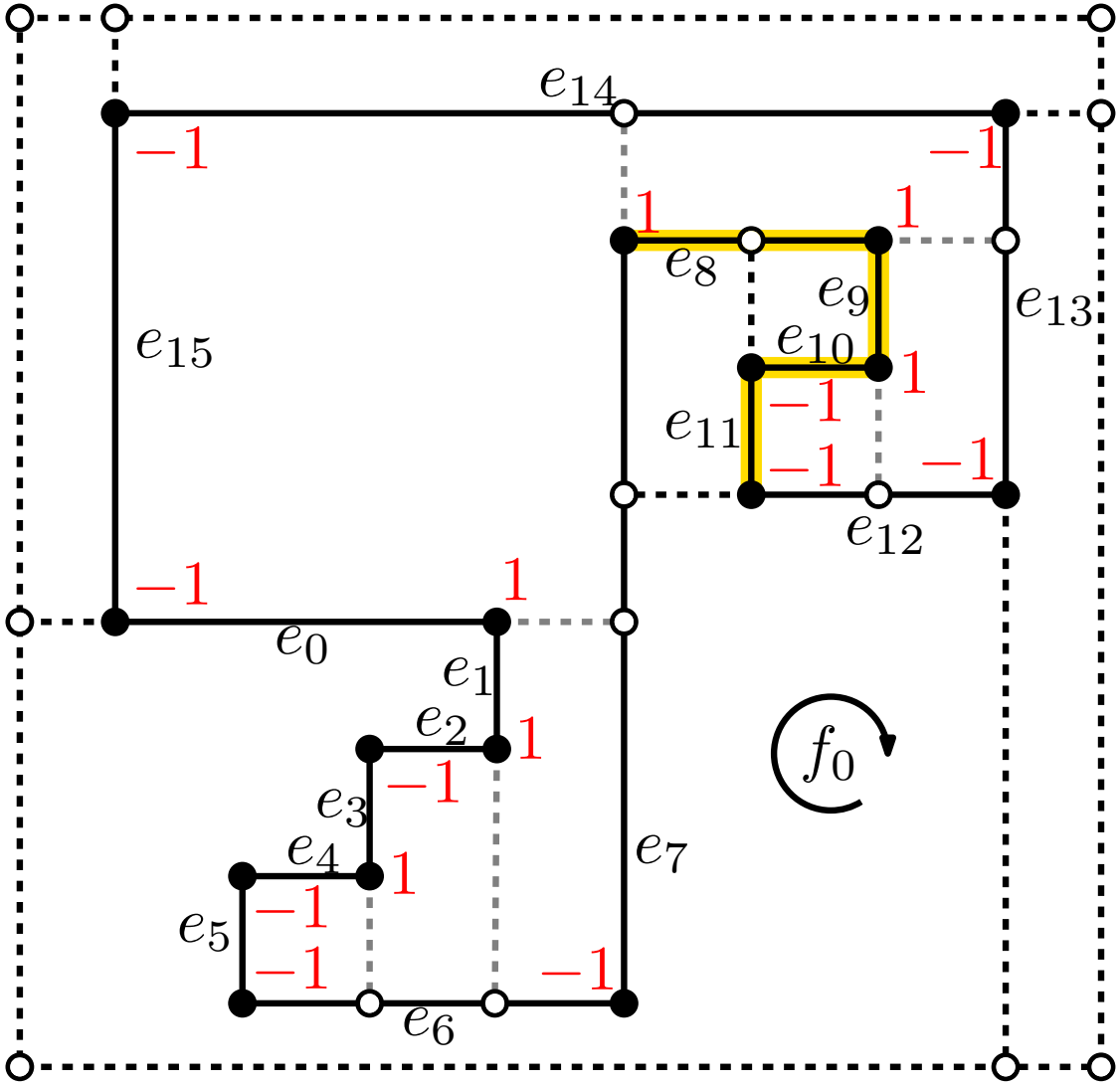
Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen
- falls $\sum \text{turn}(e) < 1$ für komplette Umrundung von f_0, projiziere auf R

$$\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$$

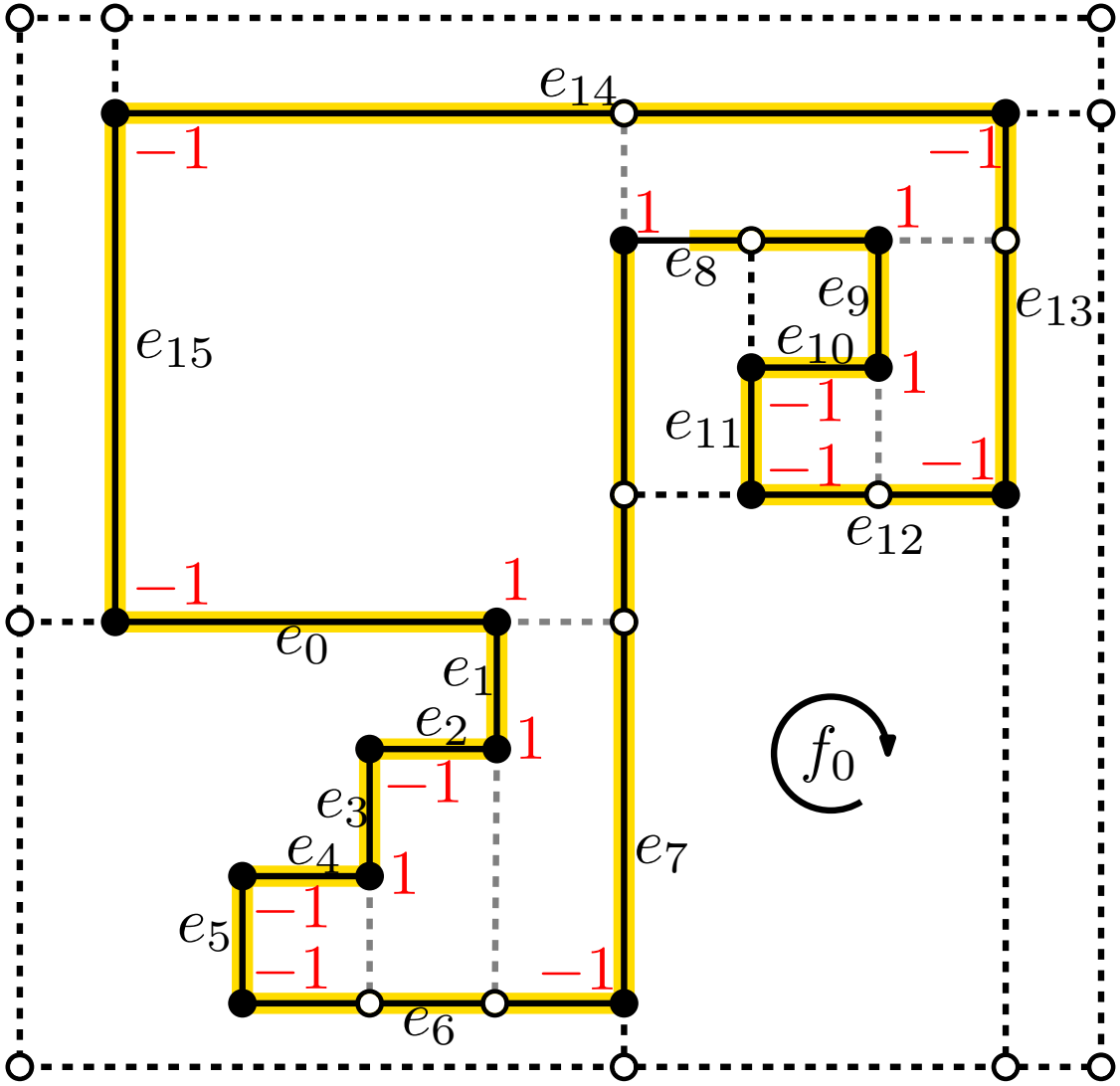
Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen
- falls $\sum \text{turn}(e) < 1$ für komplette Umrundung von f_0, projiziere auf R

$$\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$$

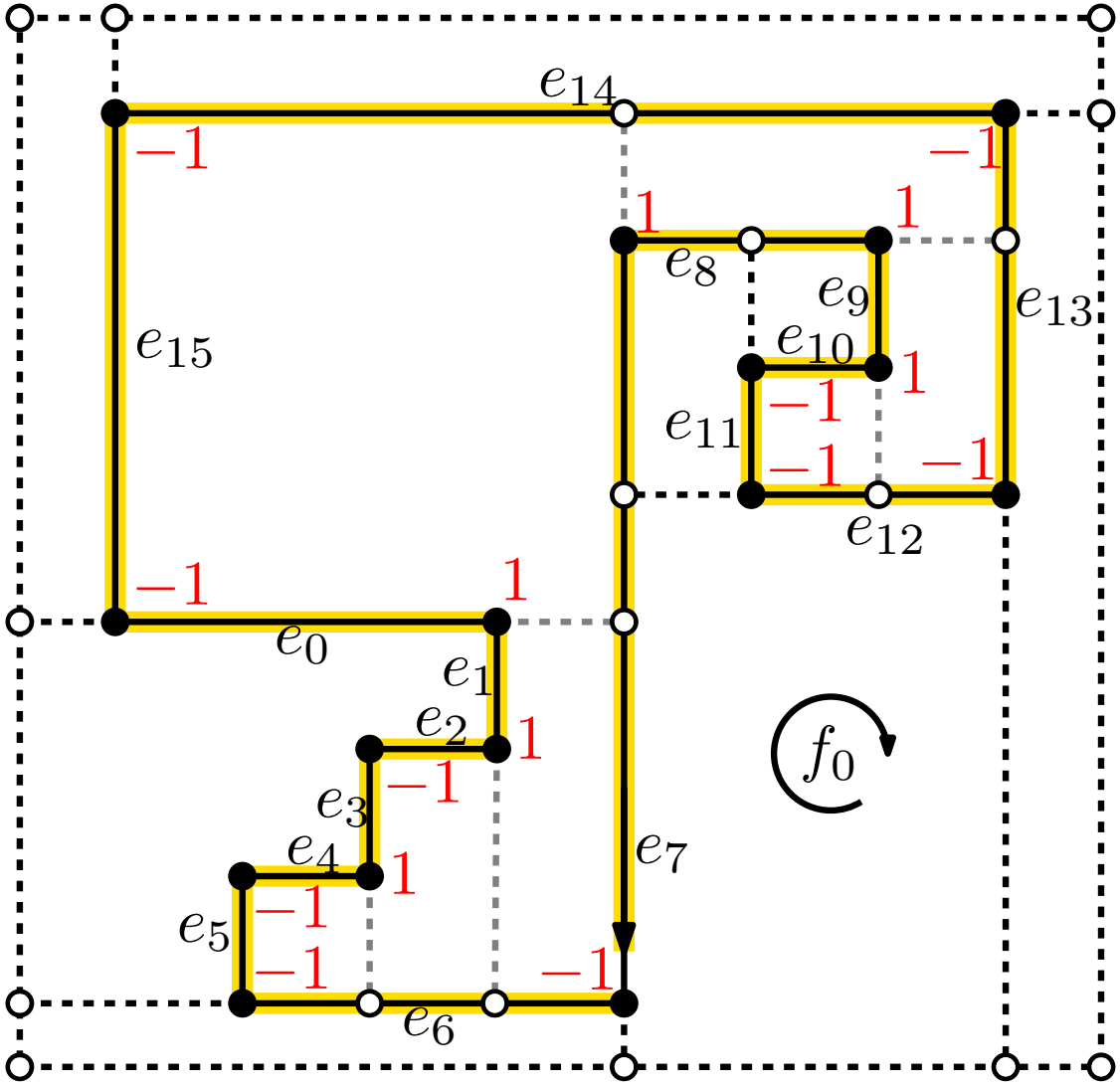
Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen
- falls $\sum \text{turn}(e) < 1$ für komplette Umrundung von f_0, projiziere auf R

$$\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$$

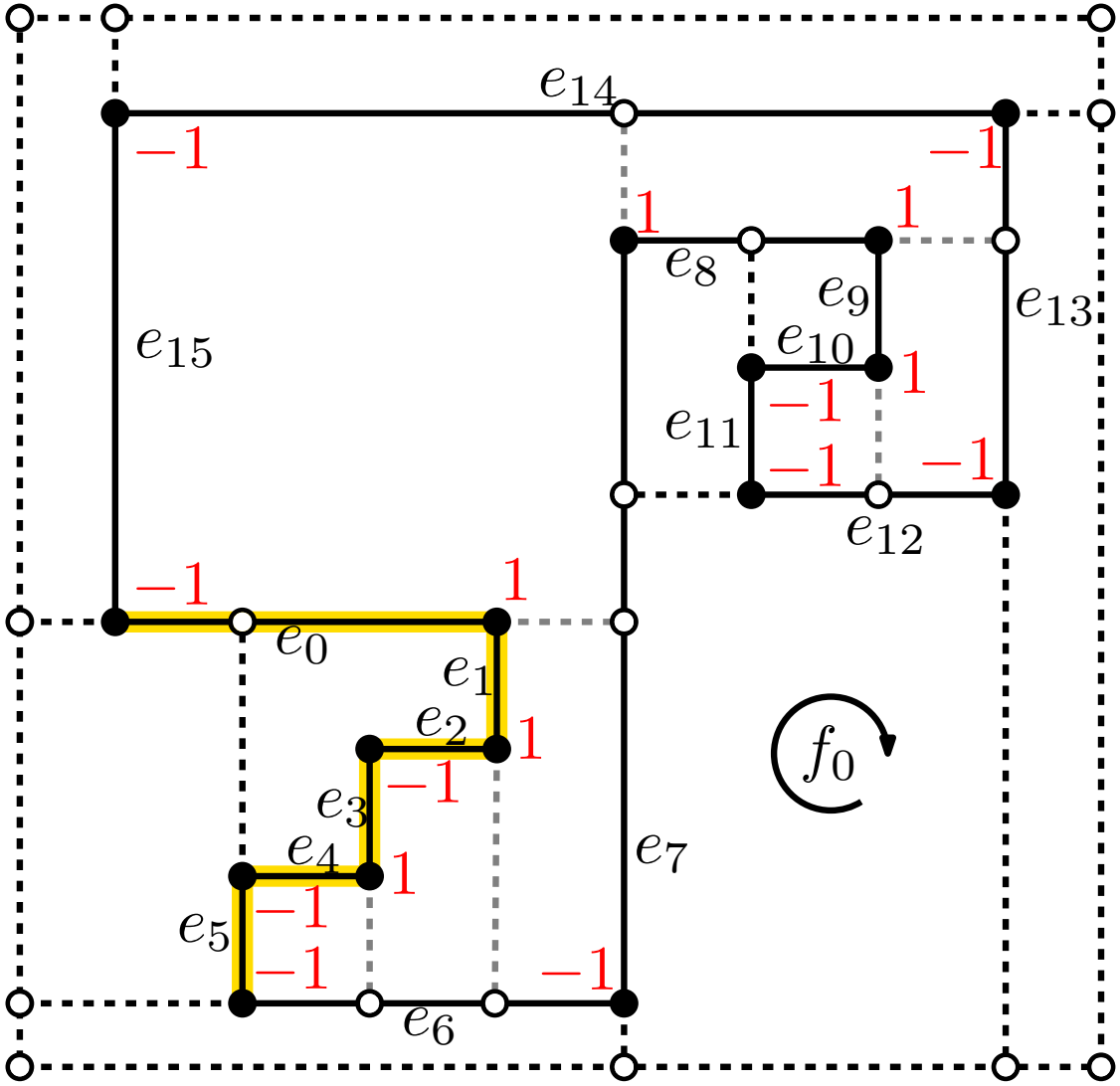
Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen
- falls $\sum \text{turn}(e) < 1$ für komplette Umrundung von f_0, projiziere auf R

$$\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$$

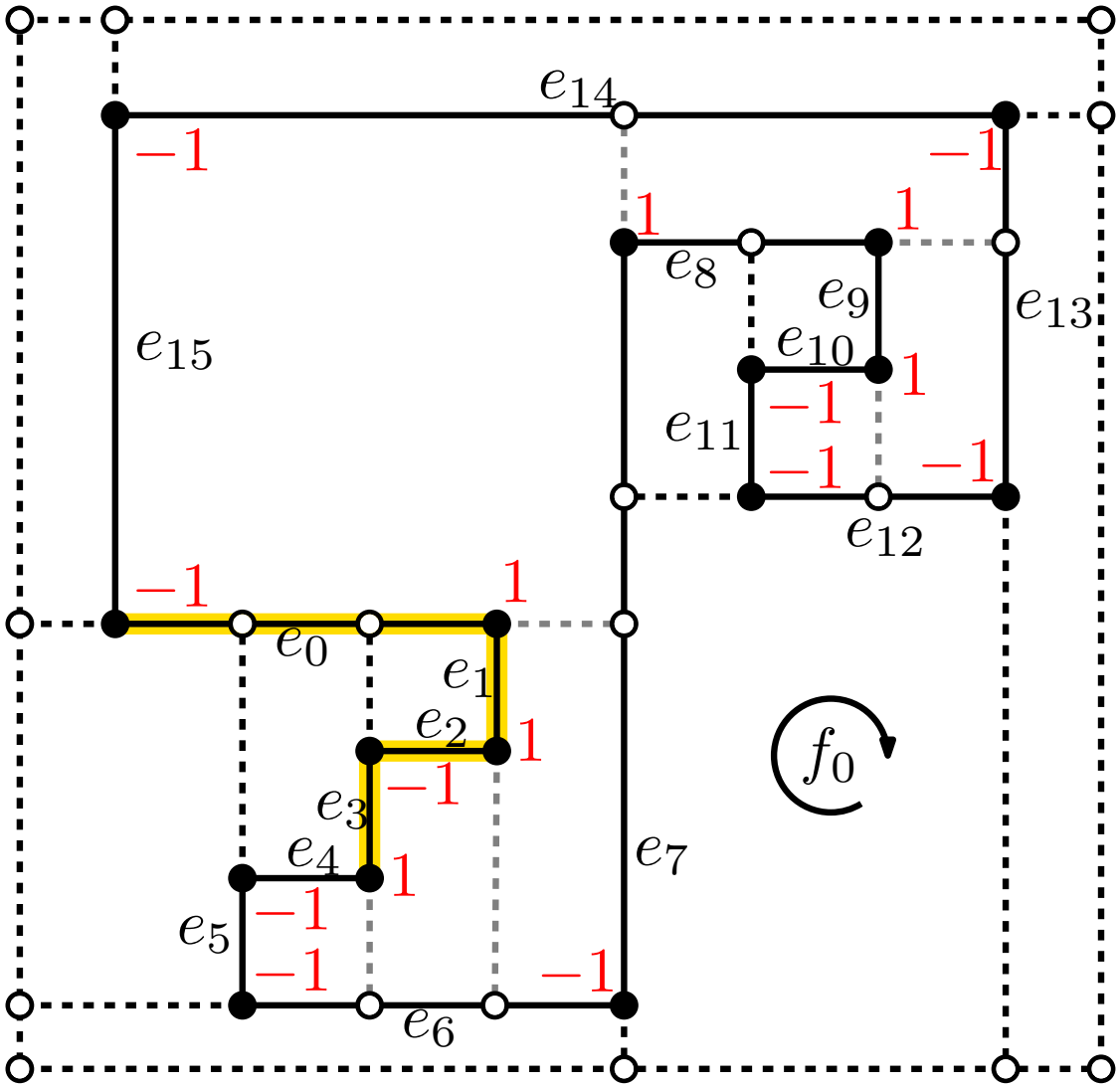
Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen
- falls $\sum \text{turn}(e) < 1$ für komplette Umrundung von f_0, projiziere auf R

$$\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$$

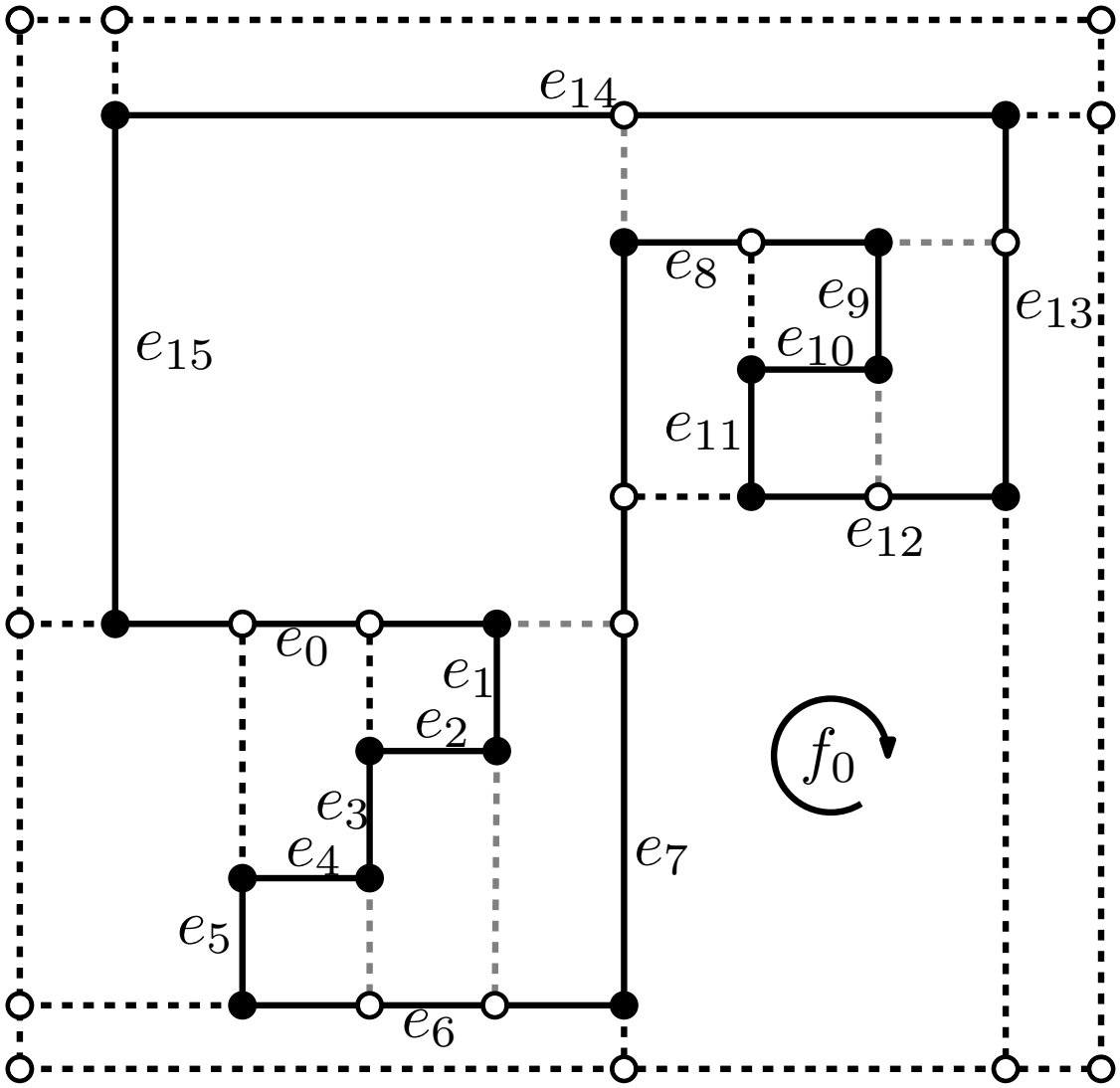
Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen
- falls $\sum \text{turn}(e) < 1$ für komplette Umrundung von f_0, projiziere auf R

$$\text{turn}(e) = \begin{cases} 1 & \text{Linksknick} \\ 0 & \text{kein Knick} \\ -1 & \text{Rechtsknick} \end{cases}$$

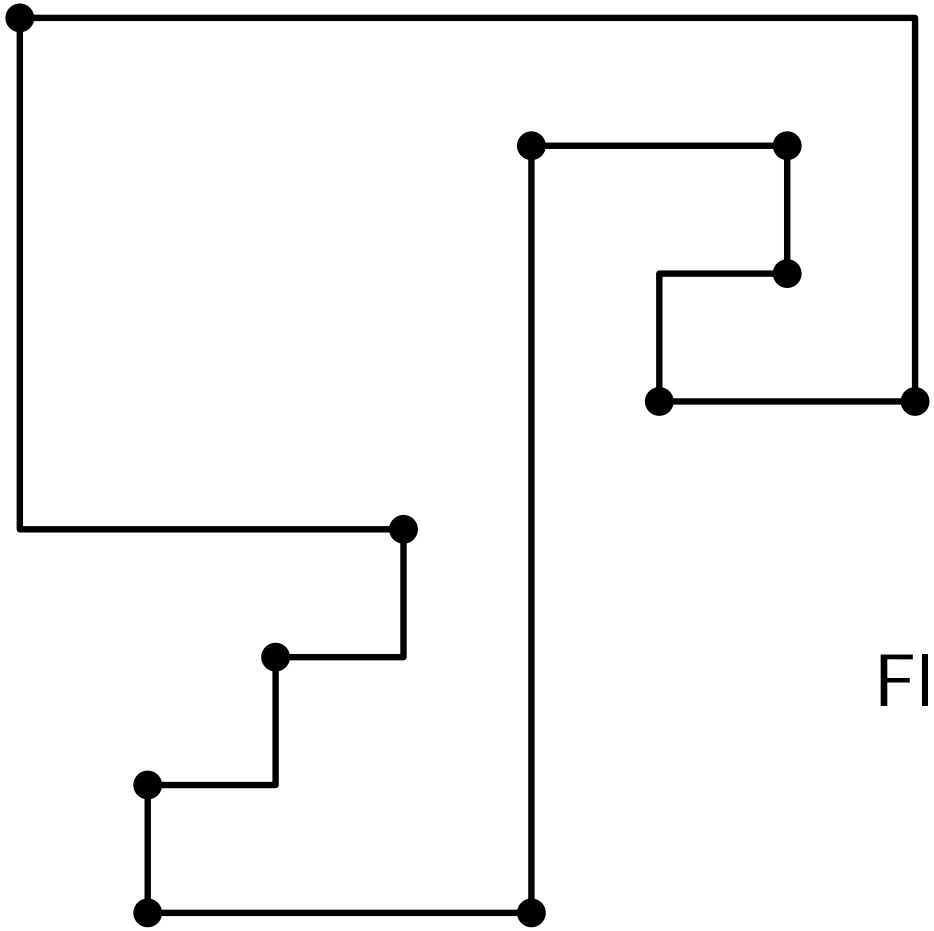
Verfeinerung von (G, H) – äußere Facette



- Rechteck R um die äußere Facette legen
- falls $\sum \text{turn}(e) < 1$ für komplette Umrundung von f_0, projiziere auf R

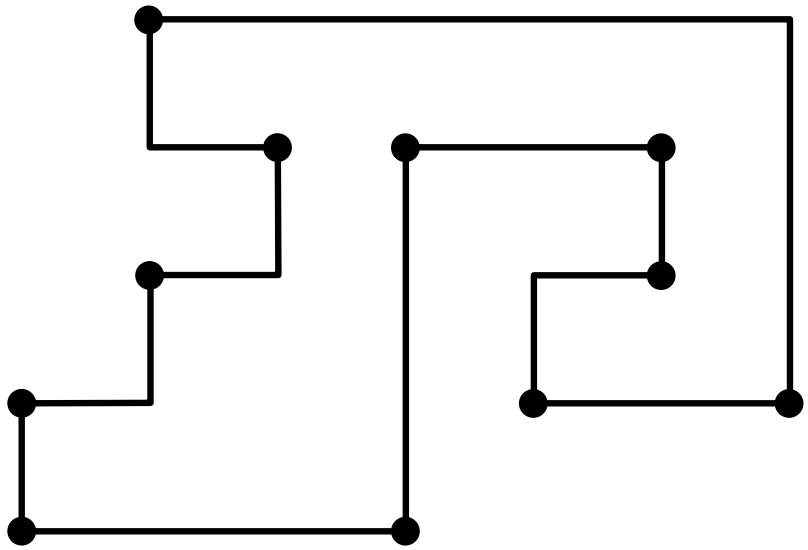
alle Facetten sind Rechtecke
$\rightarrow$ wende Flussnetzwerke an

Verfeinerung von (G, H) – äußere Facette



Flächenminimal?

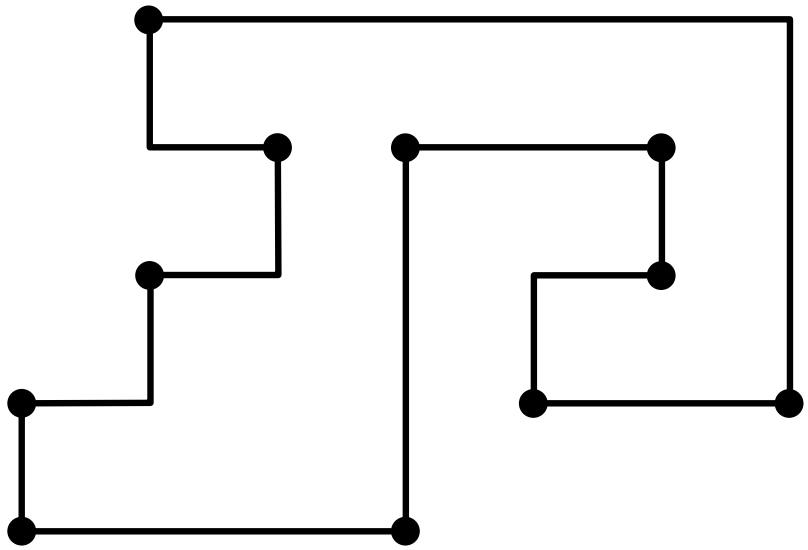
Verfeinerung von (G, H) – äußere Facette



Flächenminimal?

Nein!

Verfeinerung von (G, H) – äußere Facette



Flächenminimal?

Nein!

Die Flächenkompaktierung bei gegebener orthogonaler Beschreibung ist im Allgemeinen NP-schwer.

Komplexität der Kompaktierung

Satz 3: Für einen Graphen G mit gegebener orthogonaler Beschreibung $H(G)$ und ein $K \in \mathbb{N}$ ist es NP-schwer zu entscheiden, ob sich $(G, H(G))$ auf einem Gitter der Größe höchstens K zeichnen lässt.

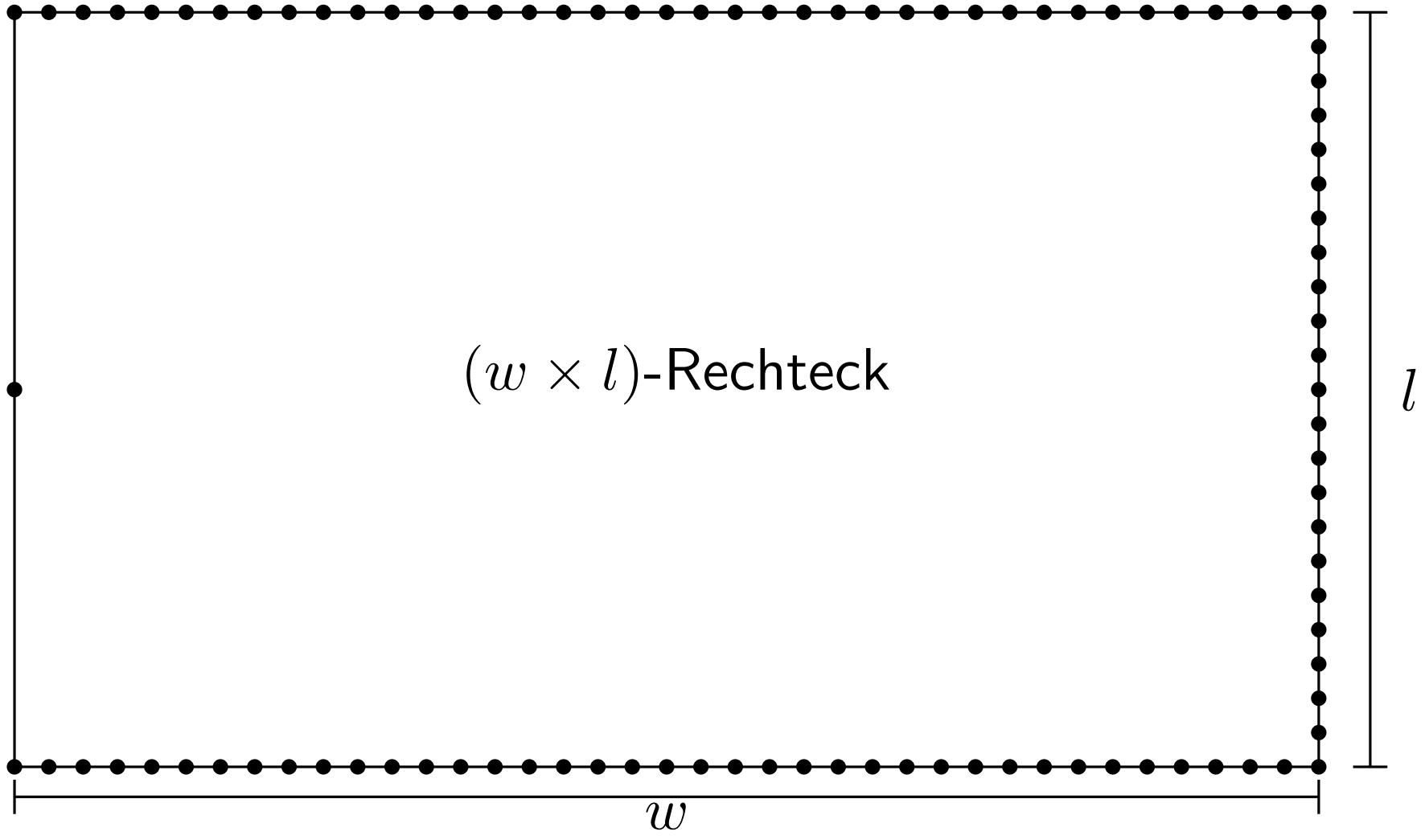
Komplexität der Kompaktierung

Satz 3: Für einen Graphen G mit gegebener orthogonaler Beschreibung $H(G)$ und ein $K \in \mathbb{N}$ ist es NP-schwer zu entscheiden, ob sich $(G, H(G))$ auf einem Gitter der Größe höchstens K zeichnen lässt.

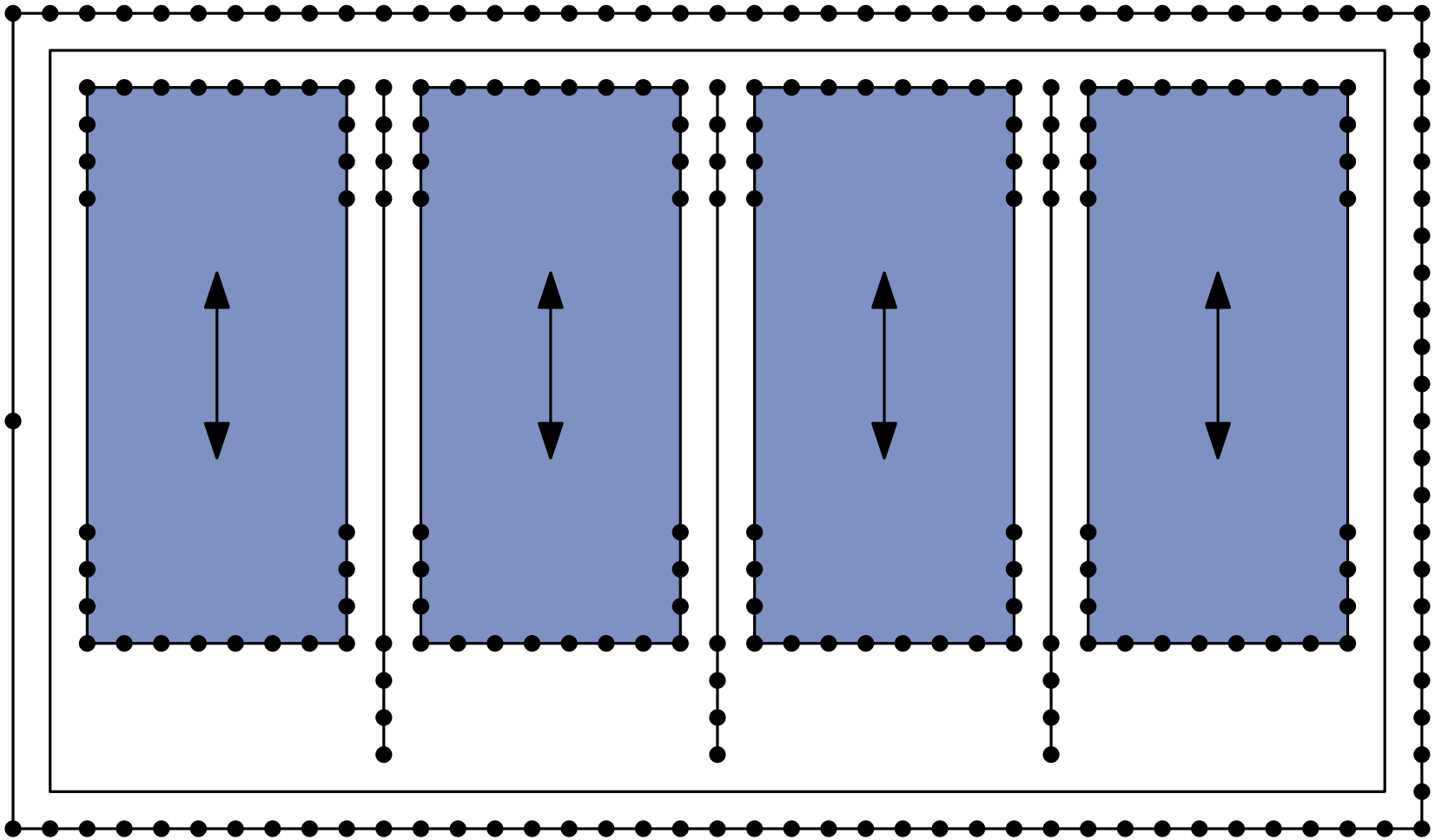
Beweisskizze:

- Reduktion von SAT, d.h. Erfüllbarkeitstest für Boolesche Formel $\Phi = c_1 \wedge \cdots \wedge c_m$, wobei jedes c_i Klausel über Variablenmenge $\{x_1, \ldots, x_n\}$
- geometrische Struktur der Reduktion
  - Klauselgadgets
  - Variablengadgets
- Bestimme geeigneten Wert K, so dass sich (G, H) in Fläche K zeichnen lässt gdw. Φ erfüllbar

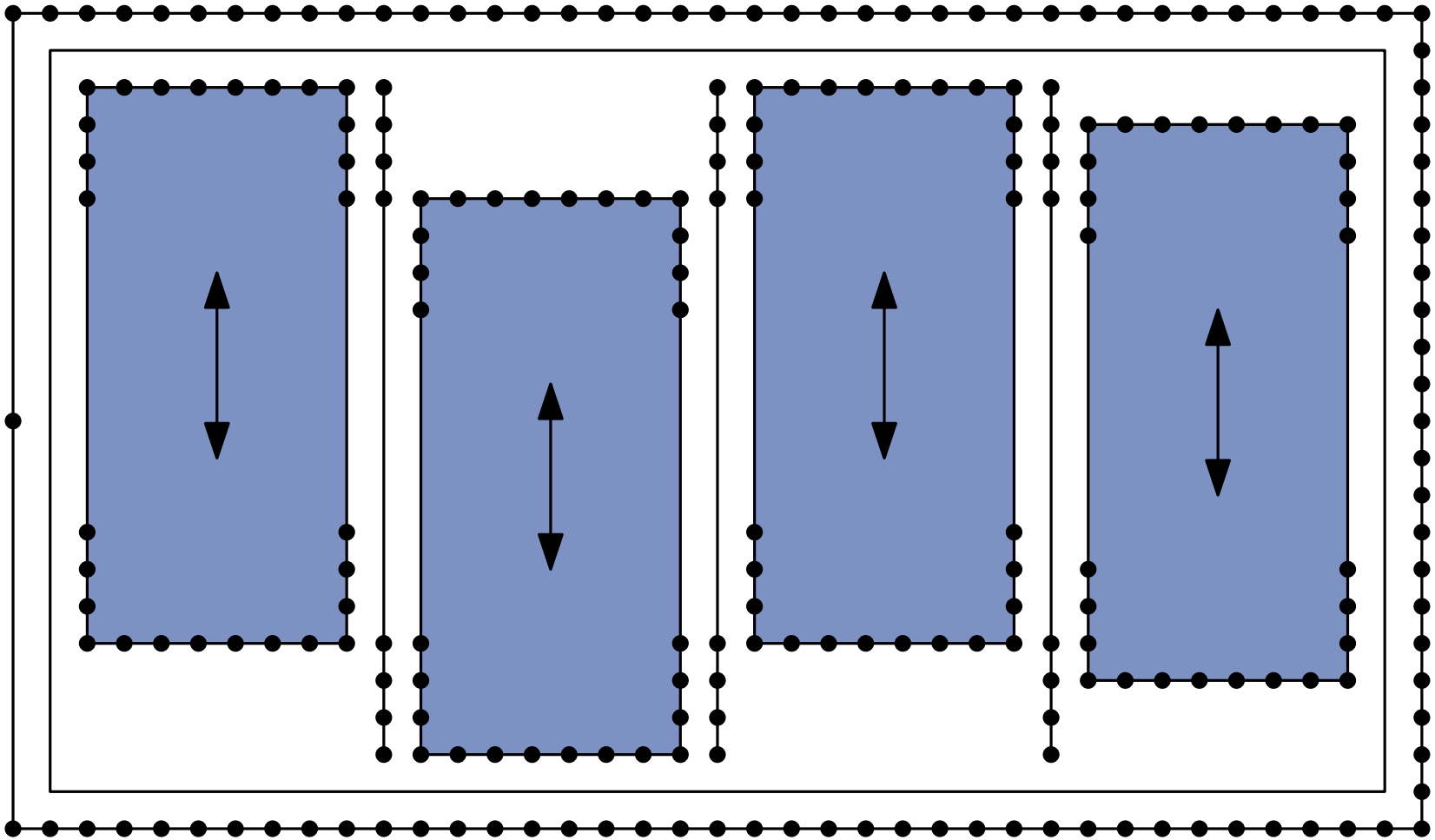
Variablengadget



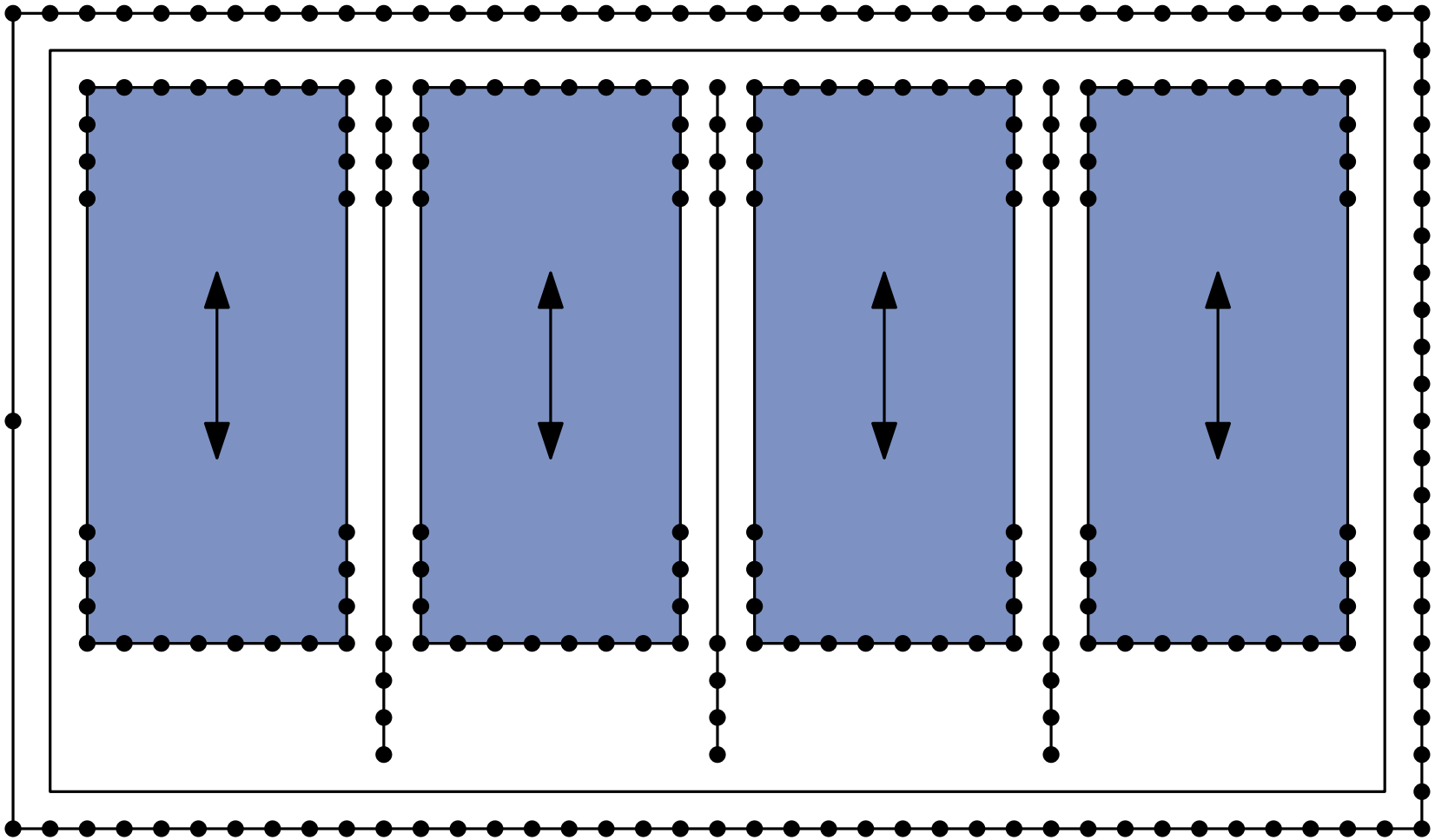
Variablengadget



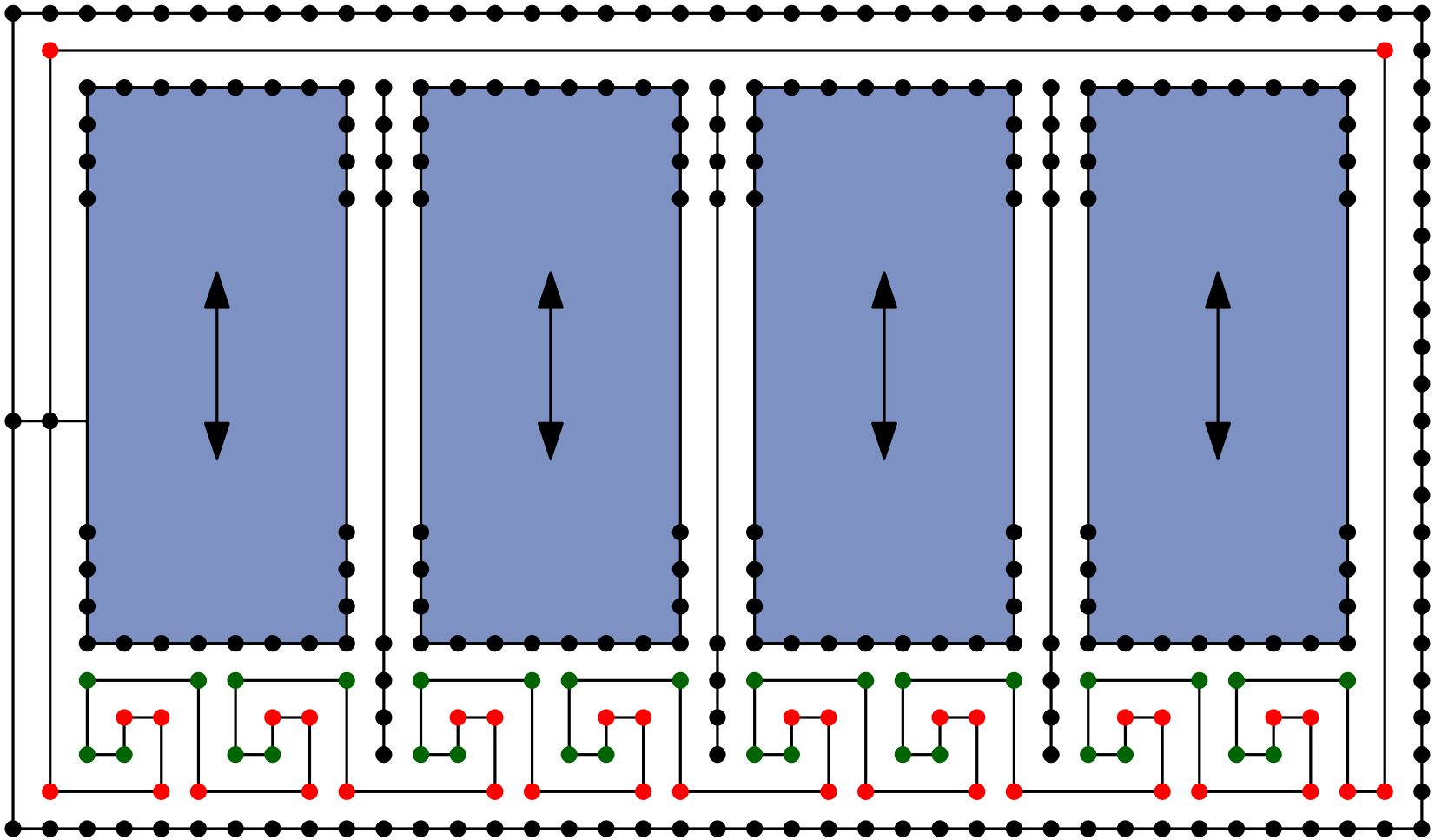
Variablengadget



Variablengadget



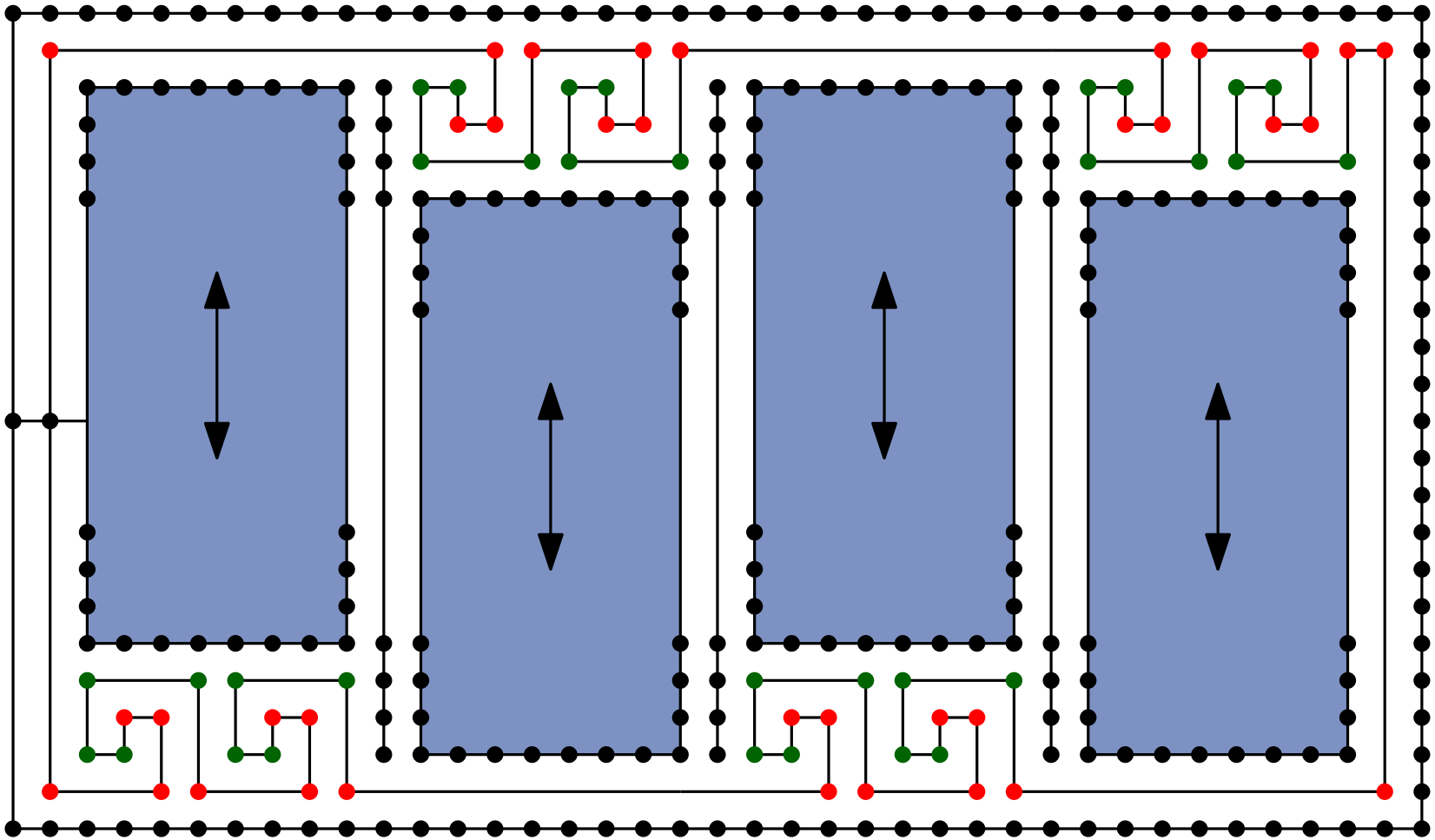
Variablengadget



Gürtel ist eine Kante mit Beschreibung $(r^4 l^4)^{2n} r^4$

- Gürtelzellen lassen sich innerhalb des Rahmens verschieben

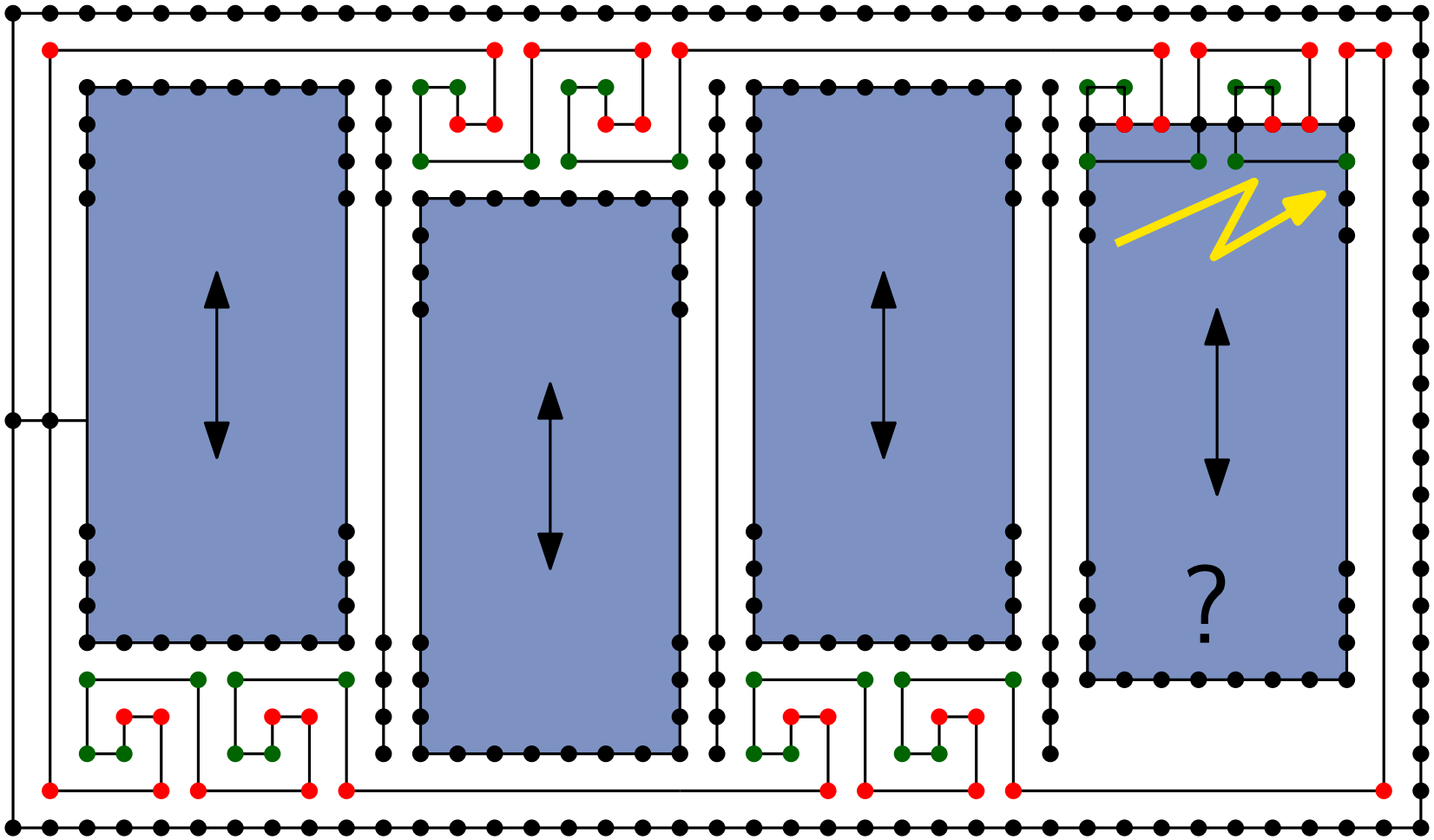
Variablengadget



Gürtel ist eine Kante mit Beschreibung $(r^4 l^4)^{2n} r^4$

- Gürtelzellen lassen sich innerhalb des Rahmens verschieben
- jeder Kolben entspricht einer Variablen mit zwei Belegungen

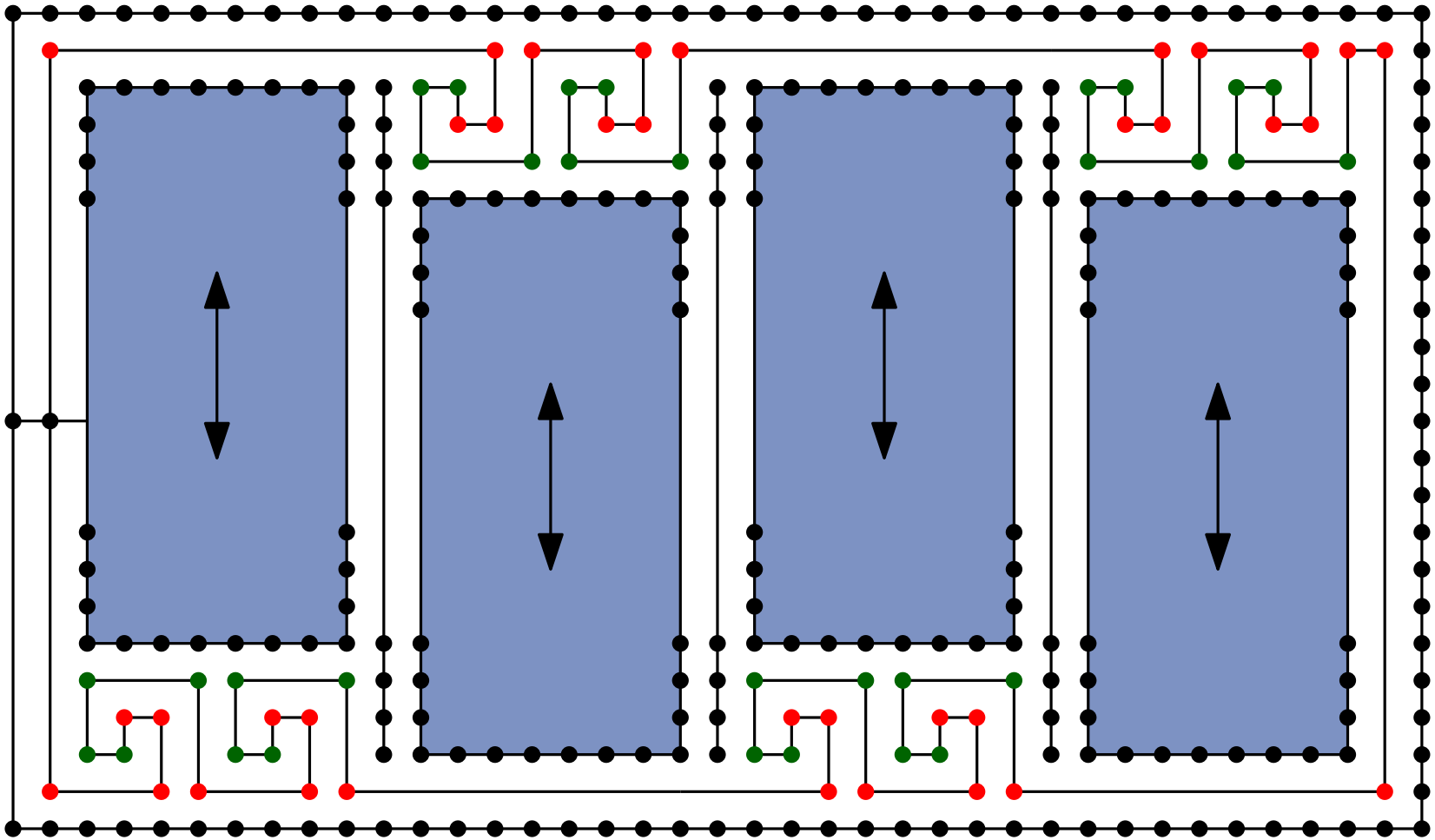
Variablengadget



Gürtel ist eine Kante mit Beschreibung $(r^4 l^4)^{2n} r^4$

- Gürtelzellen lassen sich innerhalb des Rahmens verschieben
- jeder Kolben entspricht einer Variablen mit zwei Belegungen

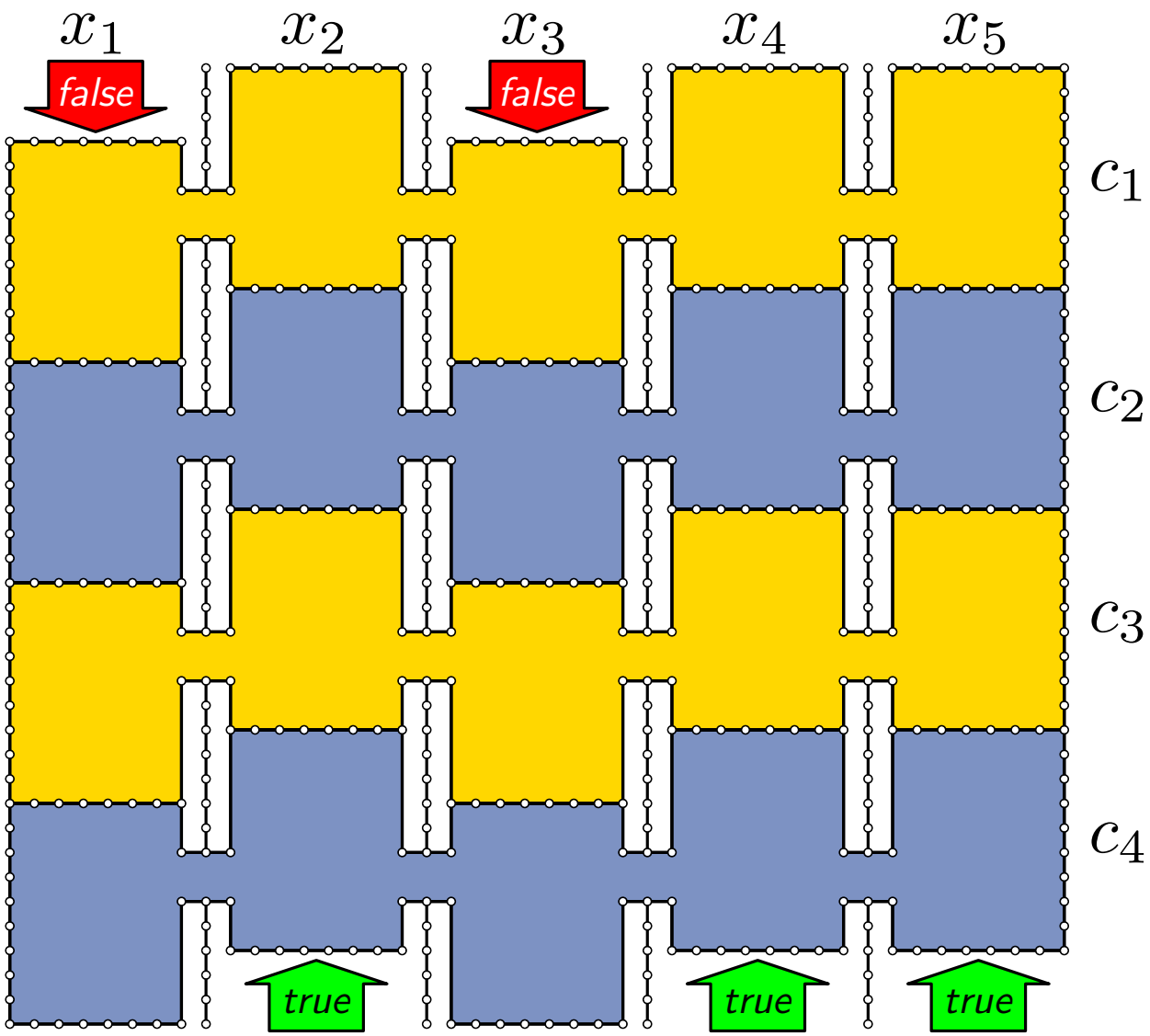
Variablengadget



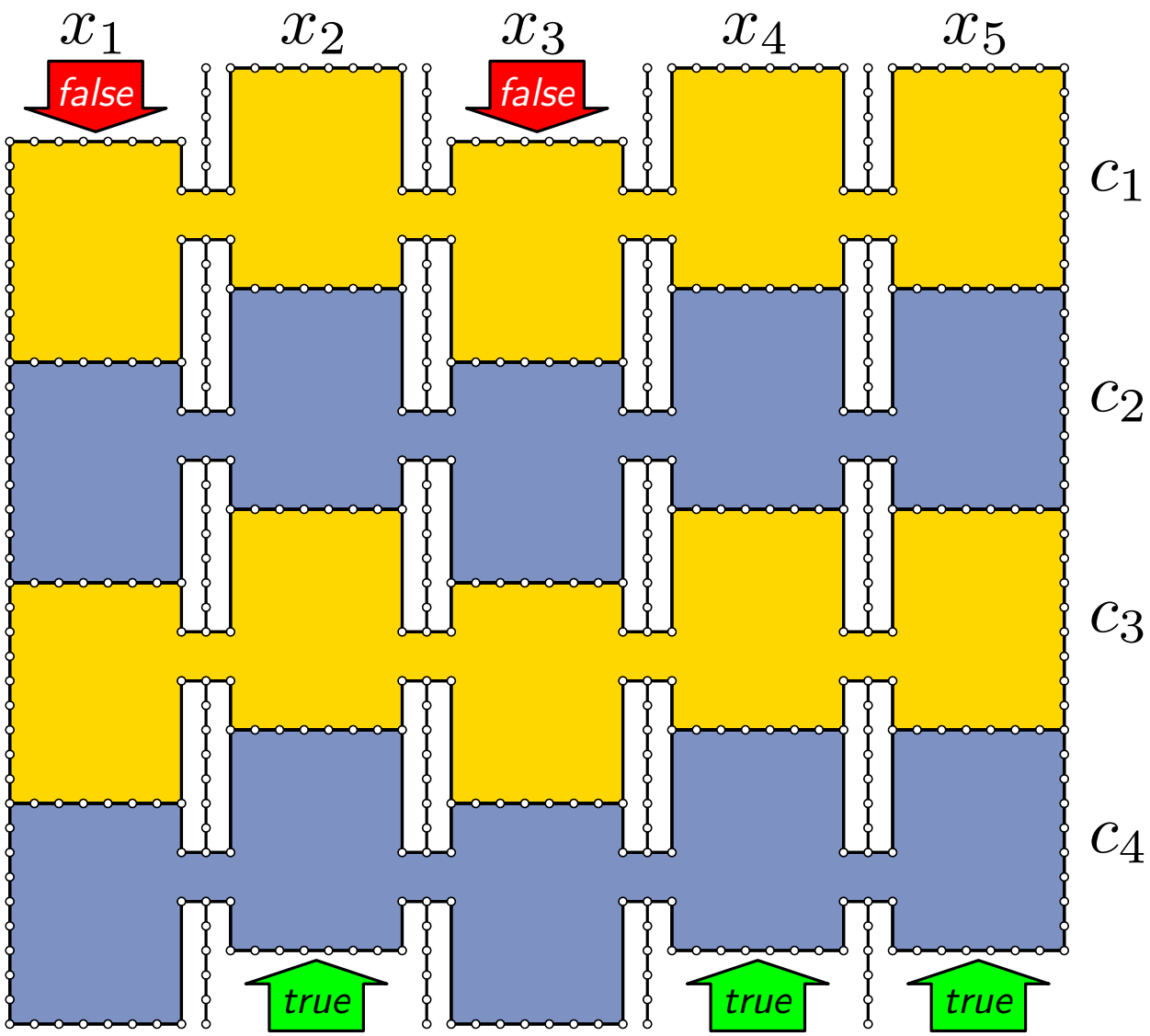
Gürtel ist eine Kante mit Beschreibung $(r^4 l^4)^{2n} r^4$

- Gürtelzellen lassen sich innerhalb des Rahmens verschieben
- jeder Kolben entspricht einer Variablen mit zwei Belegungen

Klauselgadgets



Klauselgadgets



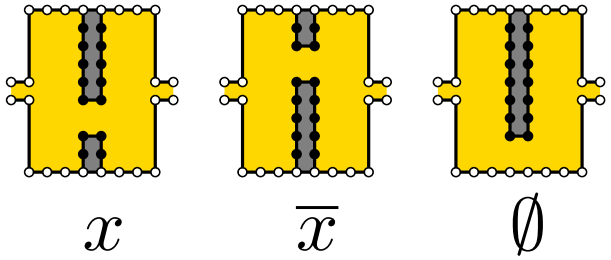
Beispiel:

$c_1 = x_2 \vee \overline{x_4}$

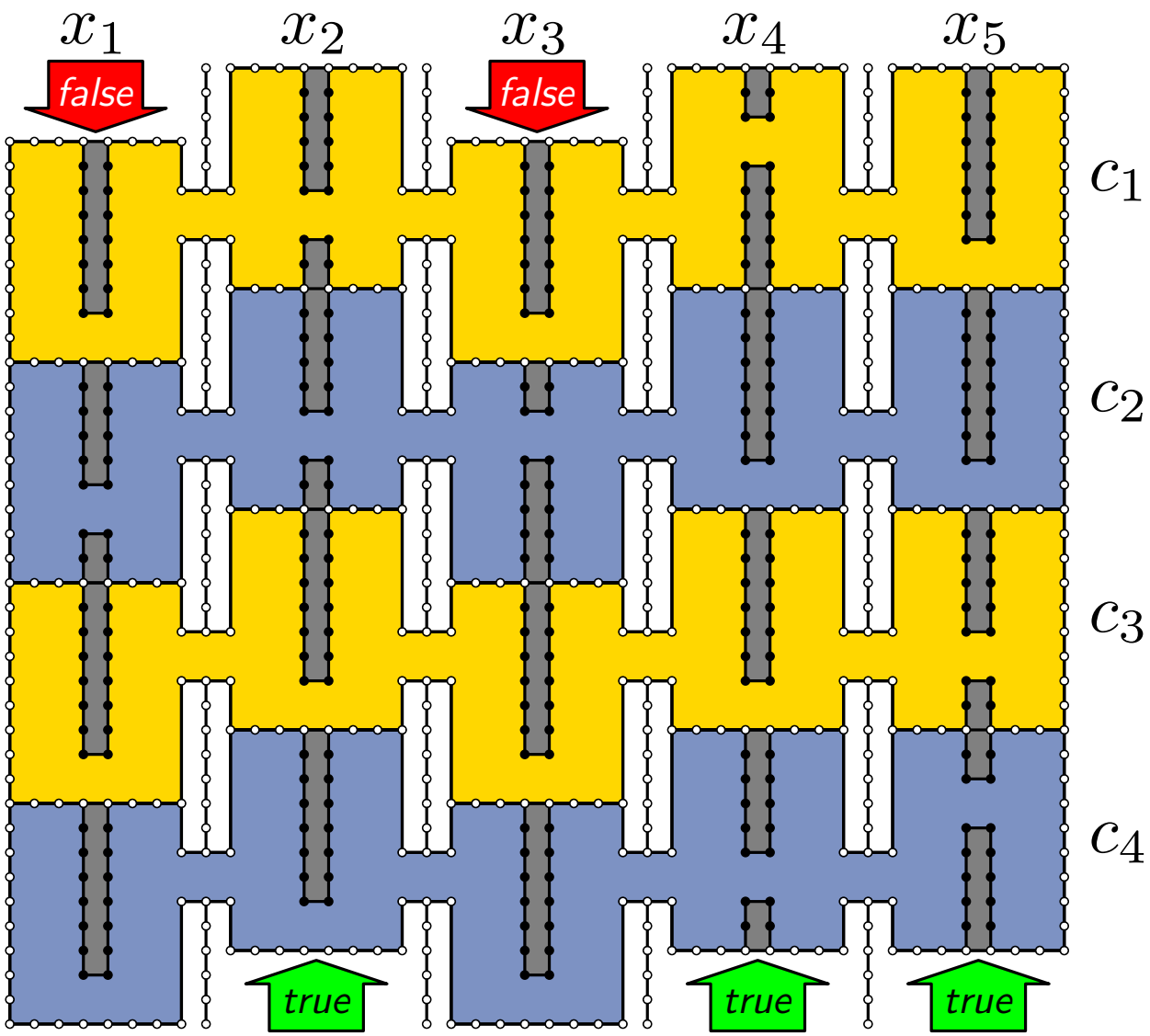
$c_2 = x_1 \vee x_2 \vee \overline{x_3}$

$c_3 = x_5$

$c_4 = x_4 \vee \overline{x_5}$



Klauselgadgets



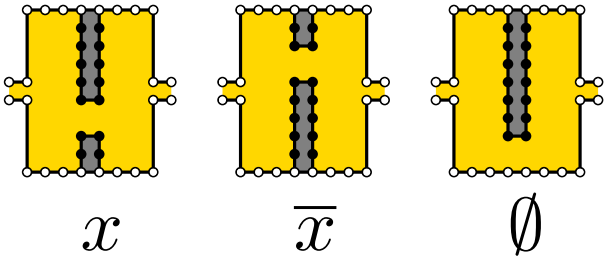
Beispiel:

$c_1 = x_2 \vee \overline{x_4}$

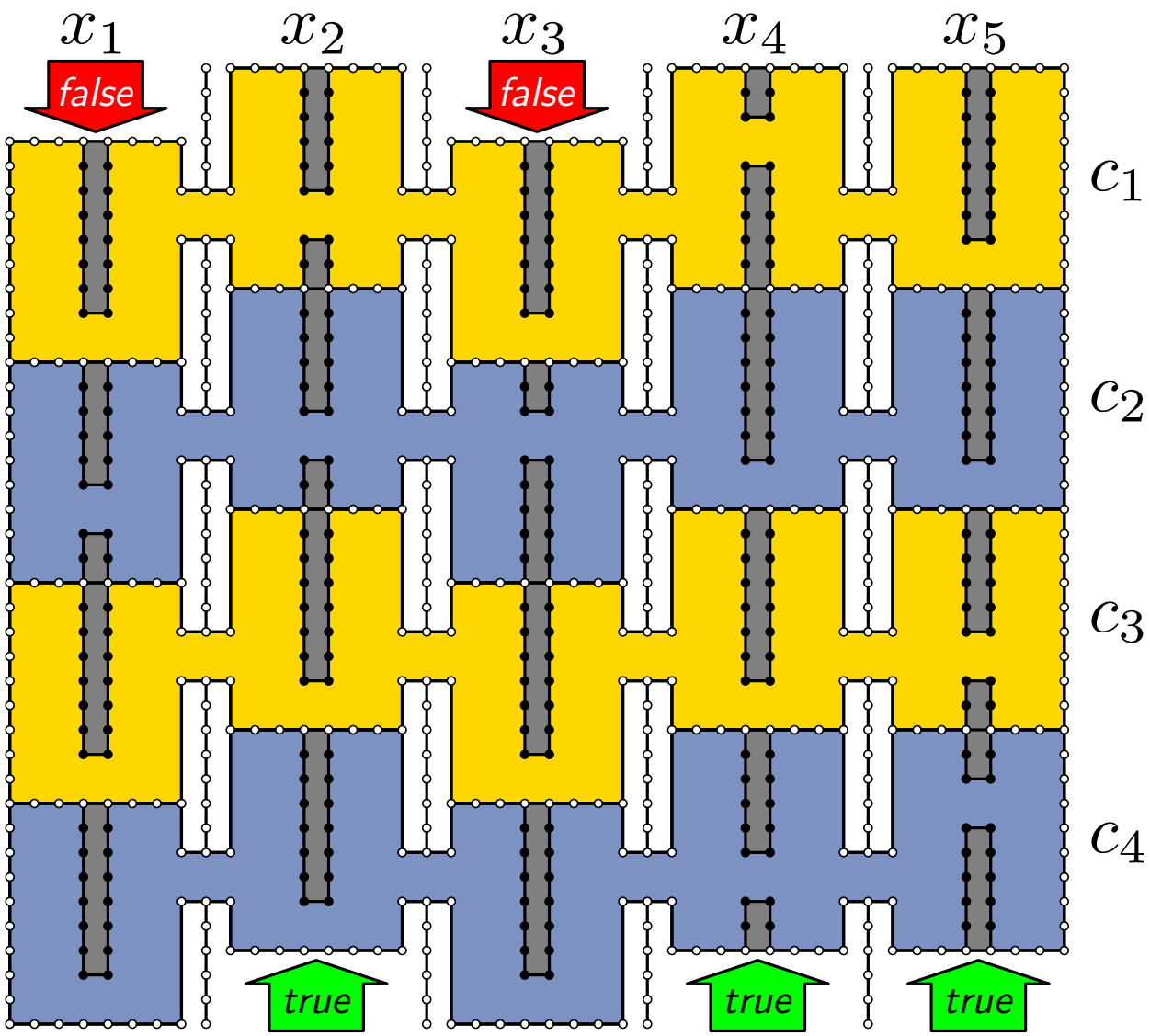
$c_2 = x_1 \vee x_2 \vee \overline{x_3}$

$c_3 = x_5$

$c_4 = x_4 \vee \overline{x_5}$



Klauselgadgets



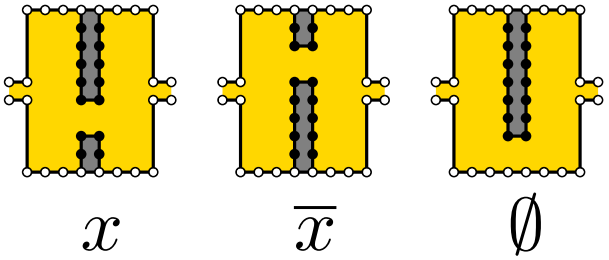
Beispiel:

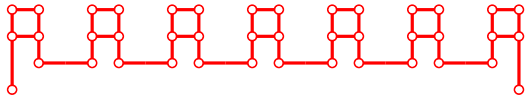
$c_1 = x_2 \vee \overline{x_4}$

$c_2 = x_1 \vee x_2 \vee \overline{x_3}$

$c_3 = x_5$

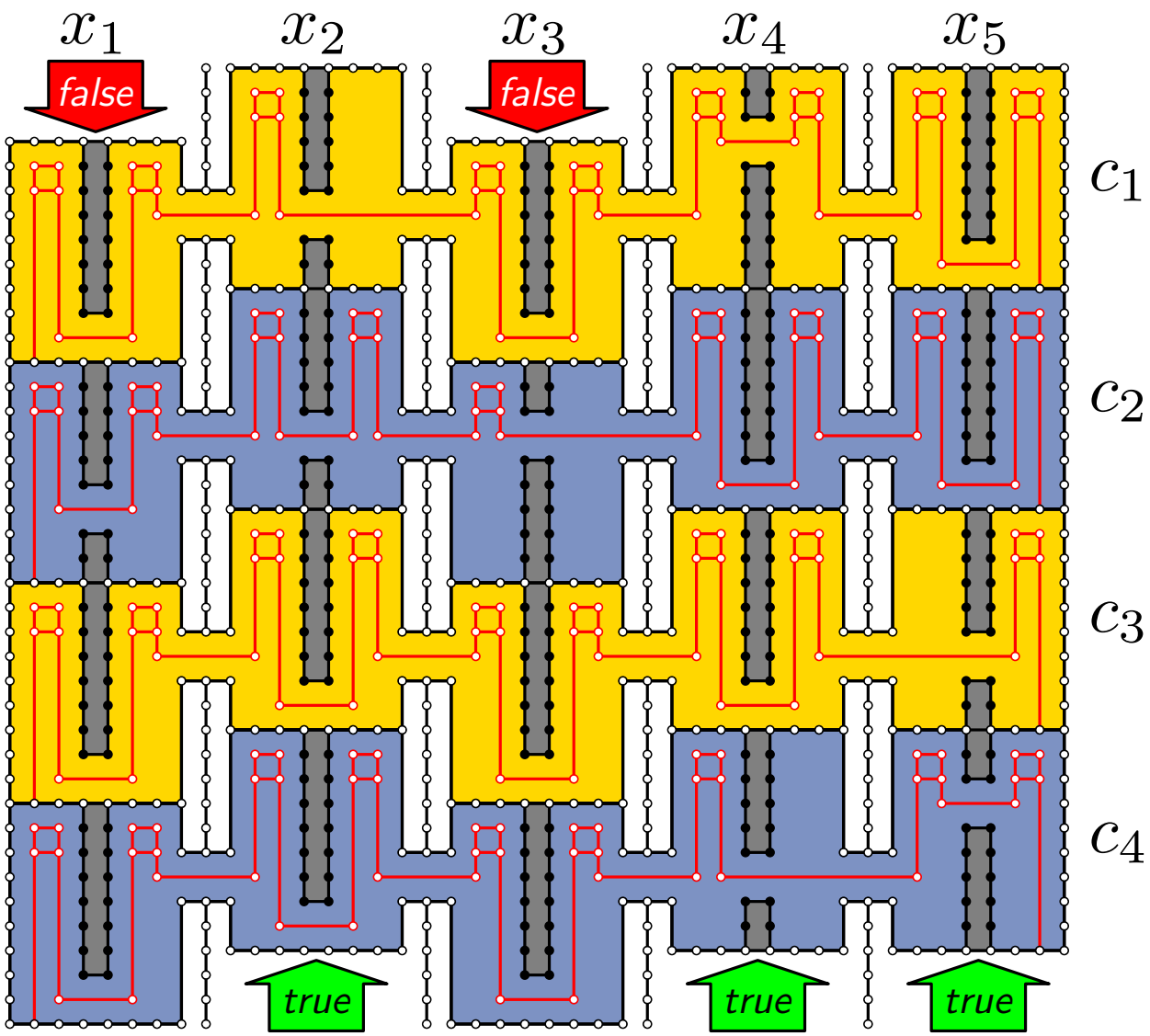
$c_4 = x_4 \vee \overline{x_5}$





lege A^{2n-1}-Kette durch jede Klausel

Klauselgadgets



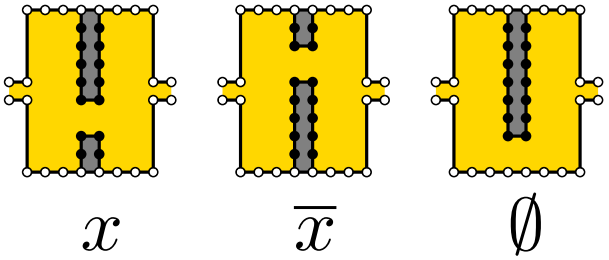
Beispiel:

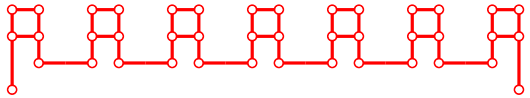
$c_1 = x_2 \vee \overline{x_4}$

$c_2 = x_1 \vee x_2 \vee \overline{x_3}$

$c_3 = x_5$

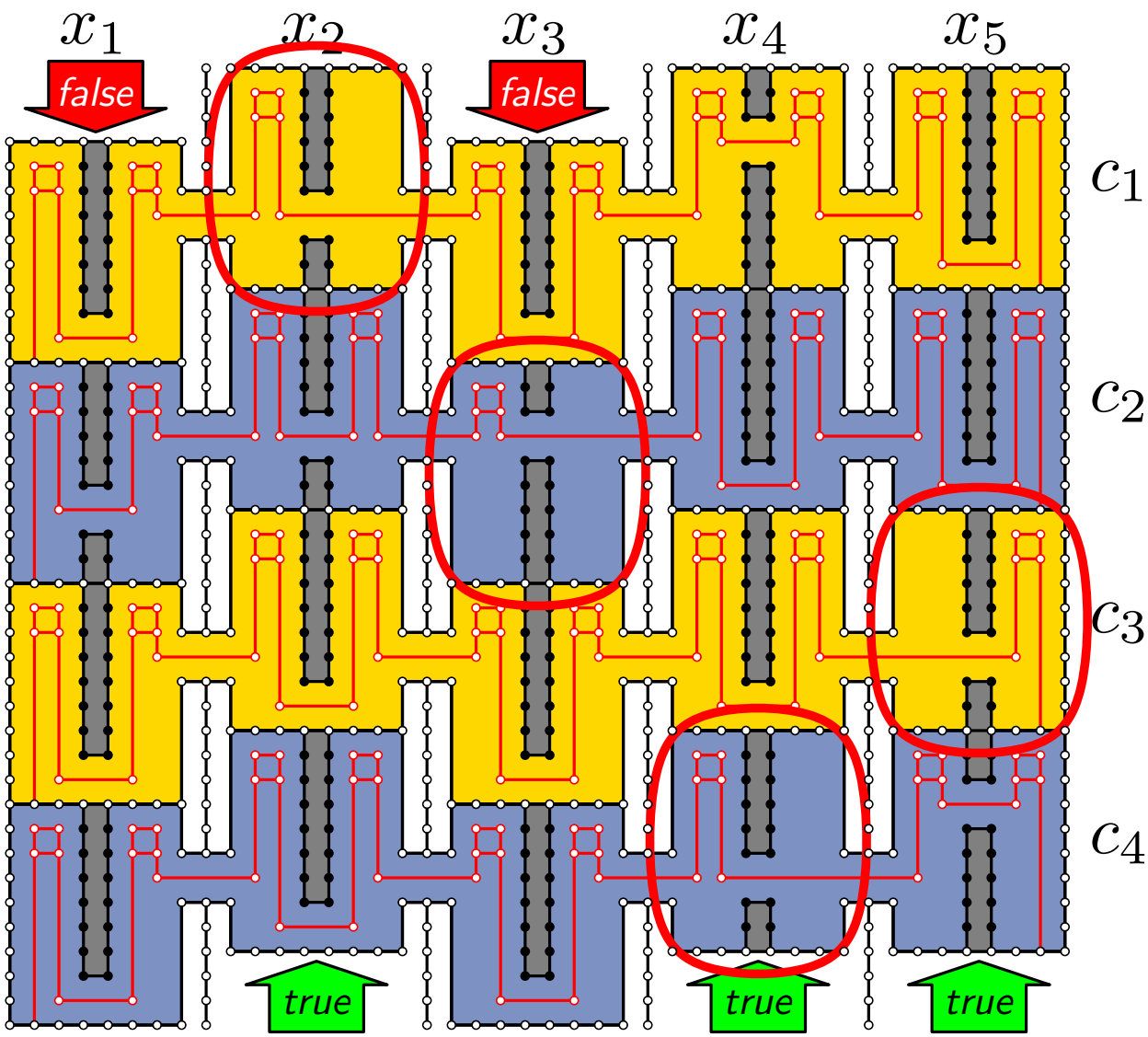
$c_4 = x_4 \vee \overline{x_5}$





lege A^{2n-1}-Kette durch jede Klausel

Klauselgadgets



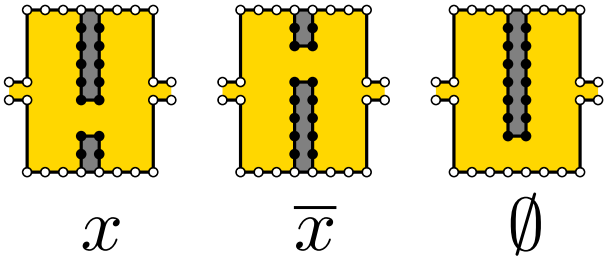
Beispiel:

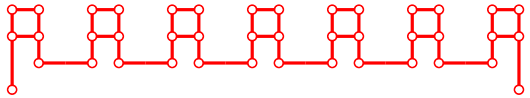
$c_1 = x_2 \vee \overline{x_4}$

$c_2 = x_1 \vee x_2 \vee \overline{x_3}$

$c_3 = x_5$

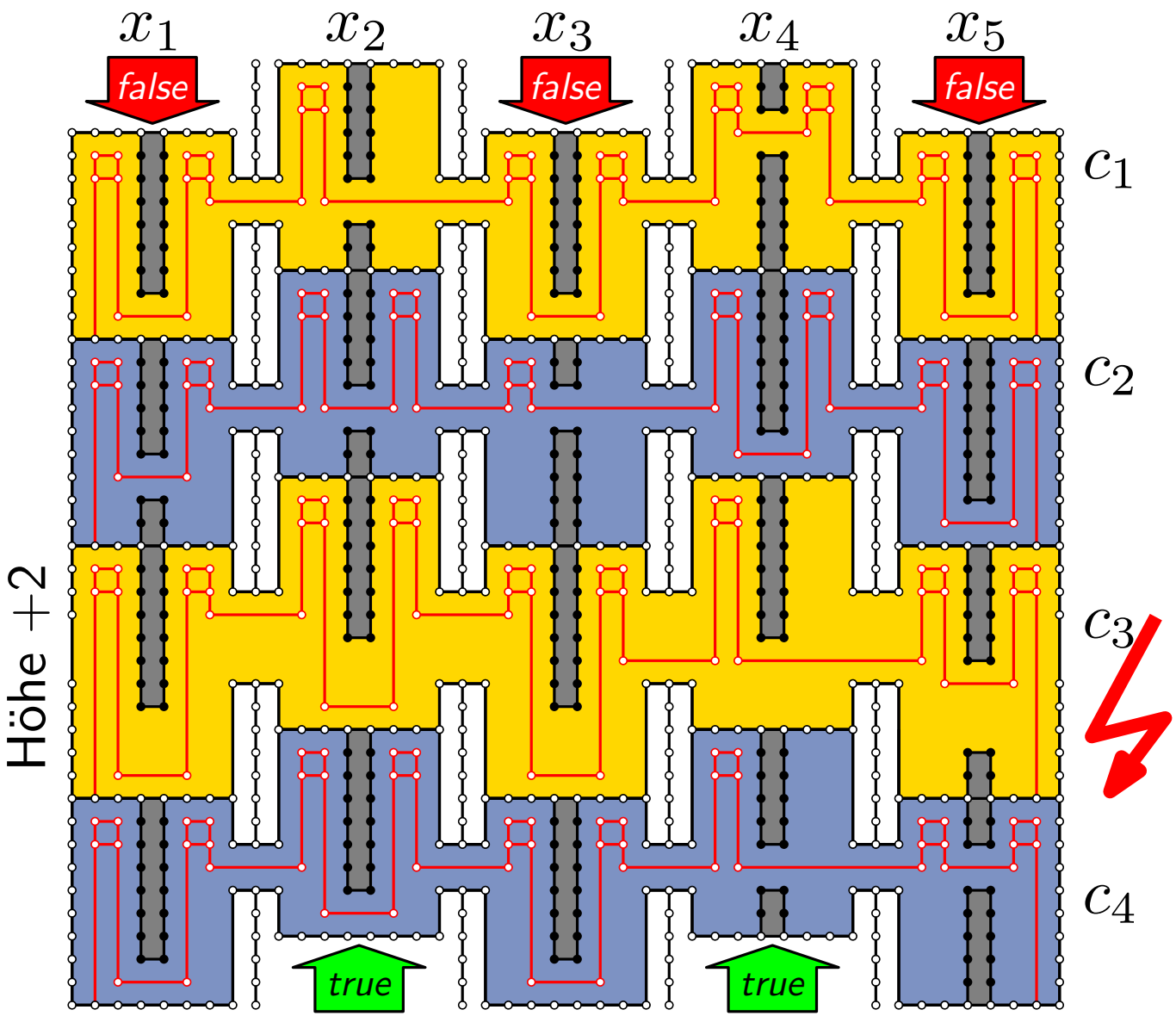
$c_4 = x_4 \vee \overline{x_5}$





lege A^{2n-1}-Kette durch jede Klausel

Klauselgadgets



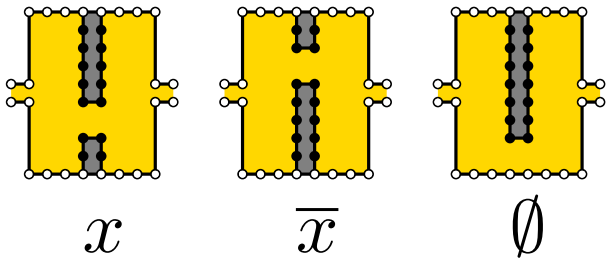
Beispiel:

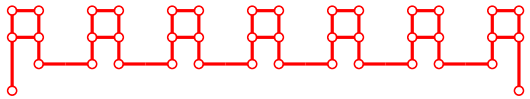
$c_1 = x_2 \vee \overline{x_4}$

$c_2 = x_1 \vee x_2 \vee \overline{x_3}$

$c_3 = x_5$

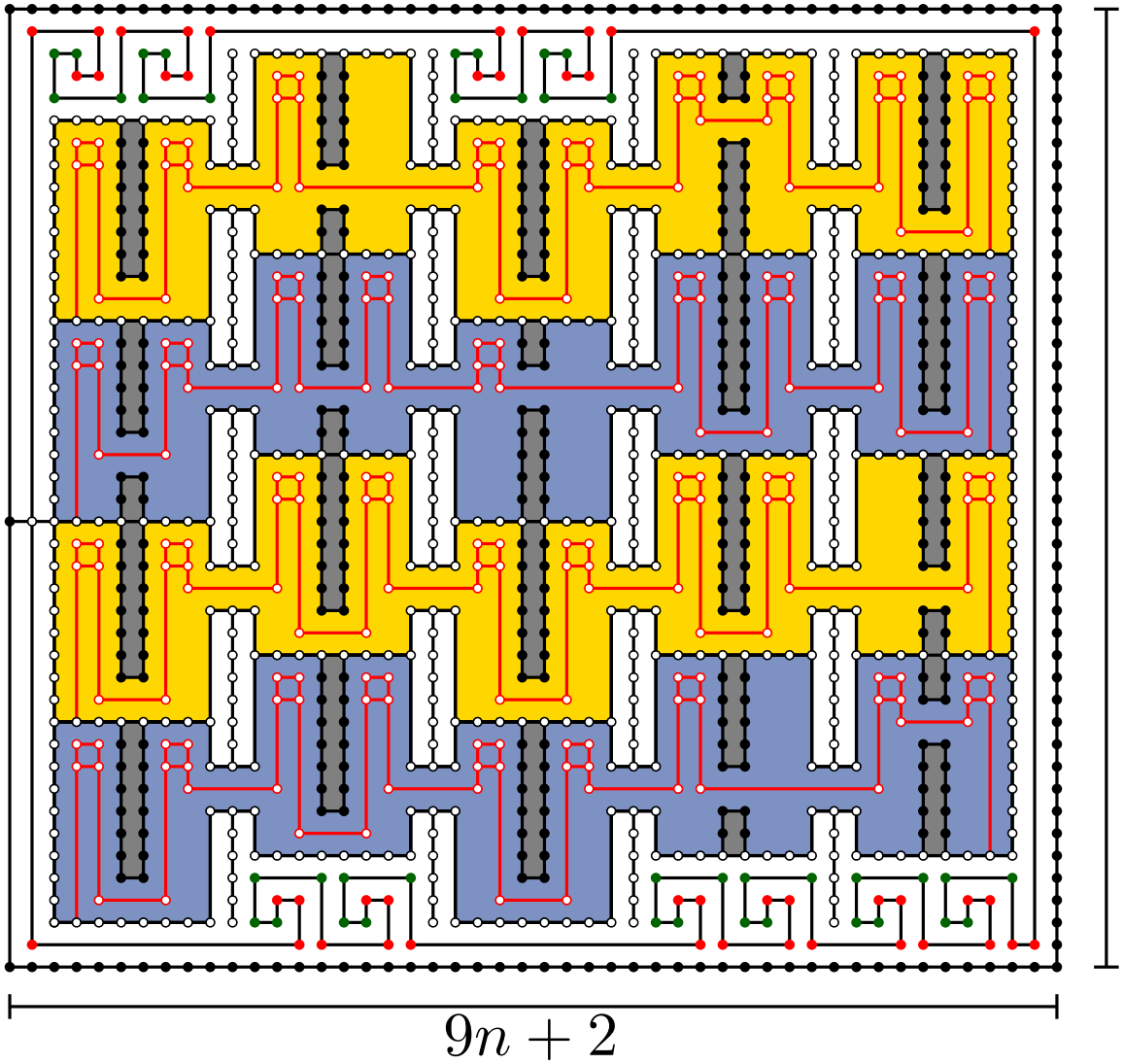
$c_4 = x_4 \vee \overline{x_5}$





lege A^{2n-1}-Kette durch jede Klausel

Komplette Reduktion



Setze

$$K = (9n+2)\cdot(9m+7)$$

Es gilt:

(G, H) auf Fläche K zeichenbar

$\Leftrightarrow$

Φ erfüllbar

$\square$

Zusammenfassung

- Flächenkompaktierung bei gegebener orthogonaler Beschreibung ist im Allgemeinen NP-schwer
- falls alle Facetten Rechtecke sind ist Kompaktierung durch Flussmodell effizient möglich
- heuristisch durch Verfeinerung nicht-rechteckiger Facetten zu Rechtecken lösbar

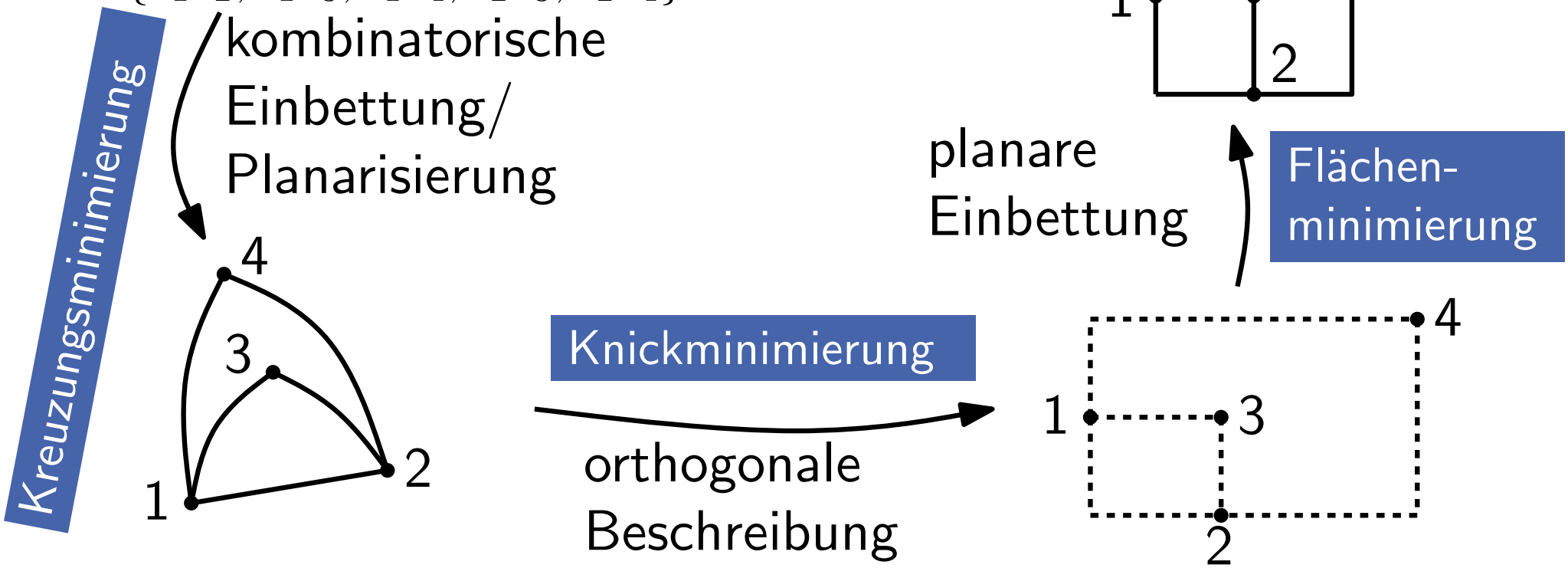
Zusammenfassung

- Flächenkompaktierung bei gegebener orthogonaler Beschreibung ist im Allgemeinen NP-schwer
- falls alle Facetten Rechtecke sind ist Kompaktierung durch Flussmodell effizient möglich
- heuristisch durch Verfeinerung nicht-rechteckiger Facetten zu Rechtecken lösbar
- im allgemeinen Fall durch ganzzahlige lineare Programmierung (ILP) lösbar [Klau, Mutzel IPCO 1999]
- experimenteller Vergleich verschiedener heuristischer Methoden bzgl. Laufzeit und Qualität [Klau, Klein, Mutzel GD 2001]
- für nicht-planare Graphen sogar schwer zu approximieren [Bannister, Eppstein, Simons JGAA 2012]

(Planare) Orthogonale Zeichnungen

Dreistufiger Ansatz: *Topology – Shape – Metrics*

$V = \{v_1, v_2, v_3, v_4\}$
$E = \{v_1v_2, v_1v_3, v_1v_4, v_2v_3, v_2v_4\}$



Aufwärtsplanare Zeichnungen

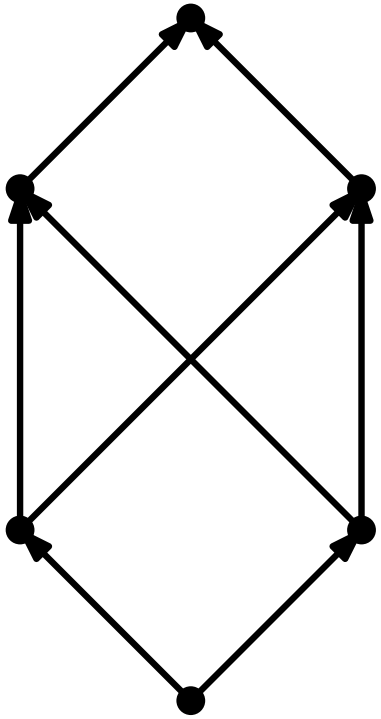
Aufwärtsplanarität

Def: Ein gerichteter azyklischer Graph $D = (V, A)$ heißt **aufwärtsplanar**, wenn es eine planare Einbettung von D in die Ebene gibt, bei der alle Kanten aufwärts gerichtet (y-monoton) sind.

Aufwärtsplanarität

Def: Ein gerichteter azyklischer Graph $D = (V, A)$ heißt **aufwärtsplanar**, wenn es eine planare Einbettung von D in die Ebene gibt, bei der alle Kanten aufwärts gerichtet (y-monoton) sind.

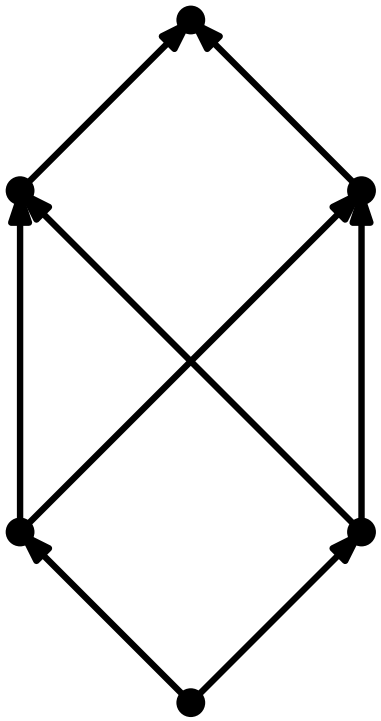
Beispiel:

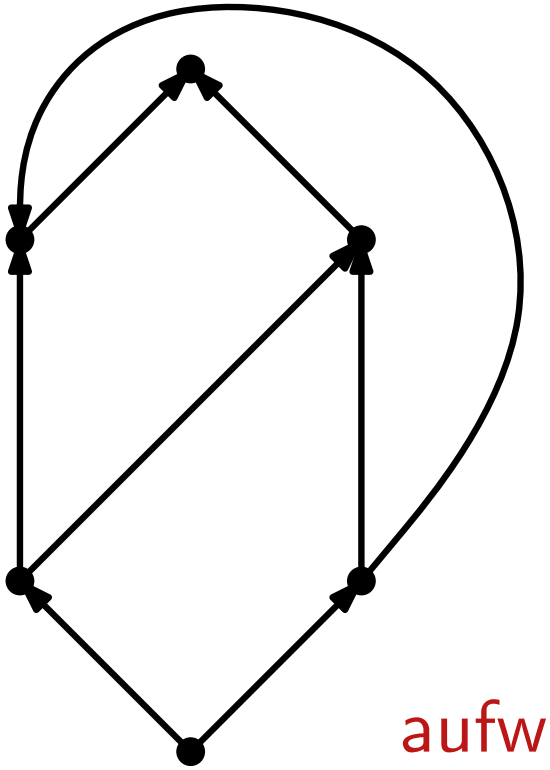


Aufwärtsplanarität

Def: Ein gerichteter azyklischer Graph $D = (V, A)$ heißt **aufwärtsplanar**, wenn es eine planare Einbettung von D in die Ebene gibt, bei der alle Kanten aufwärts gerichtet (y-monoton) sind.

Beispiel:





planar!

aufwärtsplanar? – Nein!

Komplexität

Satz 4: Für einen gerichteten azyklischen Graphen $D = (V, A)$ ist es NP-vollständig zu entscheiden, ob D aufwärtsplanar ist. [Garg, Tamassia GD 1995]

Komplexität

Satz 4: Für einen gerichteten azyklischen Graphen $D = (V, A)$ ist es NP-vollständig zu entscheiden, ob D aufwärtsplanar ist. [Garg, Tamassia GD 1995]

Dennoch kann man die aufwärtsplanaren Graphen wie folgt charakterisieren:

Satz 5: Für einen gerichteten Graphen $D = (V, A)$ sind folgende Aussagen äquivalent:

1. D ist aufwärtsplanar
2. D hat ein geradliniges aufwärtsplanares Layout
3. D ist aufspannender Subgraph eines planaren st-Graphen [Di Battista, Tamassia TCS 1988]

Komplexität

Satz 4: Für einen gerichteten azyklischen Graphen $D = (V, A)$ ist es NP-vollständig zu entscheiden, ob D aufwärtsplanar ist. [Garg, Tamassia GD 1995]

Dennoch kann man die aufwärtsplanaren Graphen wie folgt charakterisieren:

Satz 5: Für einen gerichteten Graphen $D = (V, A)$ sind folgende Aussagen äquivalent:

1. D ist aufwärtsplanar
2. D hat ein geradliniges aufwärtsplanares Layout
3. D ist aufspannender Subgraph eines planaren st-Graphen [Di Battista, Tamassia TCS 1988]

st-Graph: (i) eindeutige Quelle s und Senke t, (ii) Kante $(s, t) \in E$

Komplexität

Satz 4: Für einen gerichteten azyklischen Graphen $D = (V, A)$ ist es NP-vollständig zu entscheiden, ob D aufwärtsplanar ist. [Garg, Tamassia GD 1995]

Dennoch kann man die aufwärtsplanaren Graphen wie folgt charakterisieren:

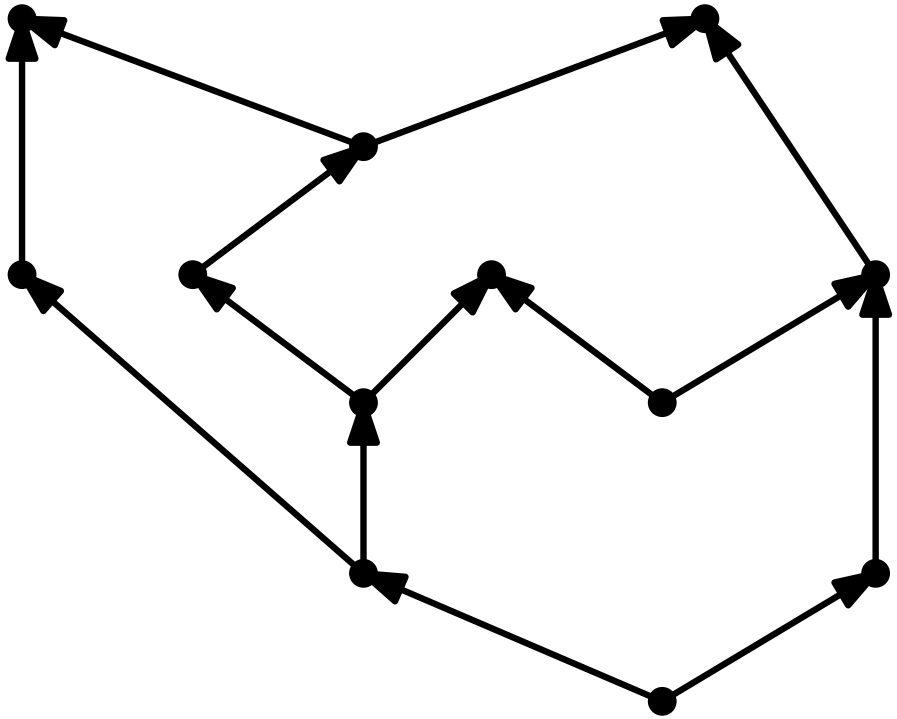
Satz 5: Für einen gerichteten Graphen $D = (V, A)$ sind folgende Aussagen äquivalent:

1. D ist aufwärtsplanar
2. D hat ein geradliniges aufwärtsplanares Layout
3. D ist aufspannender Subgraph eines planaren st-Graphen [Di Battista, Tamassia TCS 1988]

Beweis:
- (2) $\Rightarrow$ (1) klar
- (1) $\Leftrightarrow$ (3) relativ einfach
- (3) $\Rightarrow$ (2) etwas aufwändiger

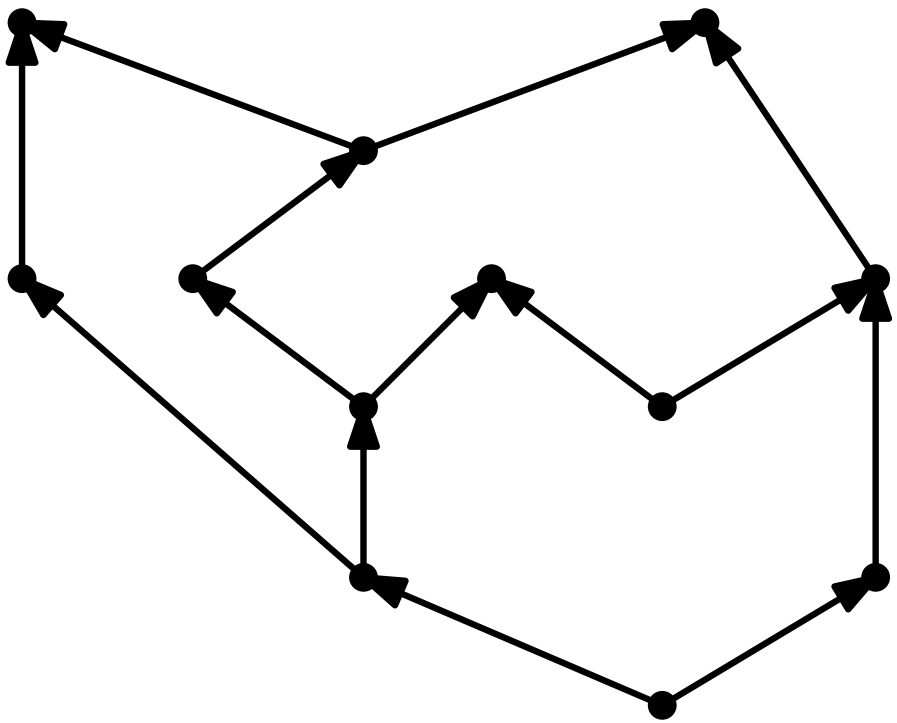
Einschränkung: feste Einbettung

Problem: Gegeben ein gerichteter azyklischer Graph $D = (V, A)$ mit Einbettung $\mathcal{F}, f_0$. Teste, ob $D, \mathcal{F}, f_0$ aufwärtsplanar ist und konstruiere ggf. ein entsprechendes Layout.

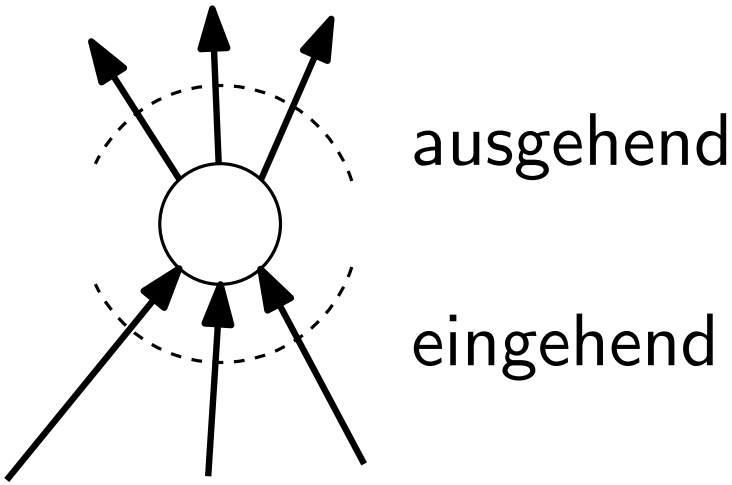


Einschränkung: feste Einbettung

Problem: Gegeben ein gerichteter azyklischer Graph $D = (V, A)$ mit Einbettung $\mathcal{F}, f_0$. Teste, ob $D, \mathcal{F}, f_0$ aufwärtsplanar ist und konstruiere ggf. ein entsprechendes Layout.

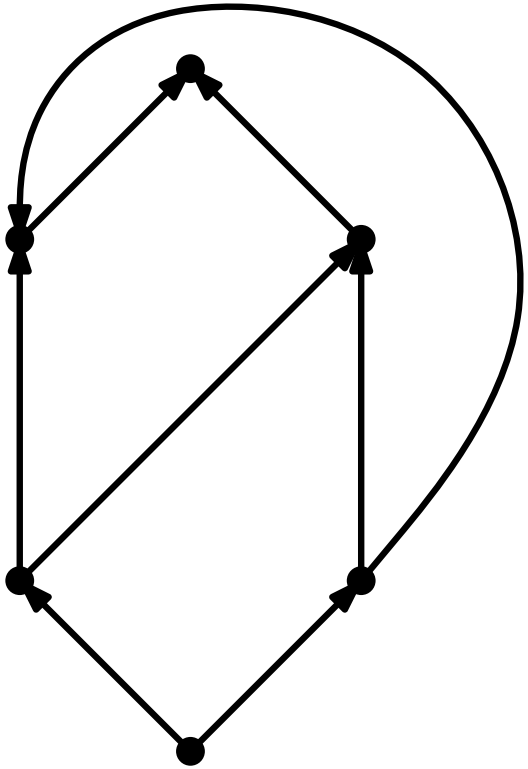


Einbettung ist **bimodal** für jeden Knoten



Beobachtungen

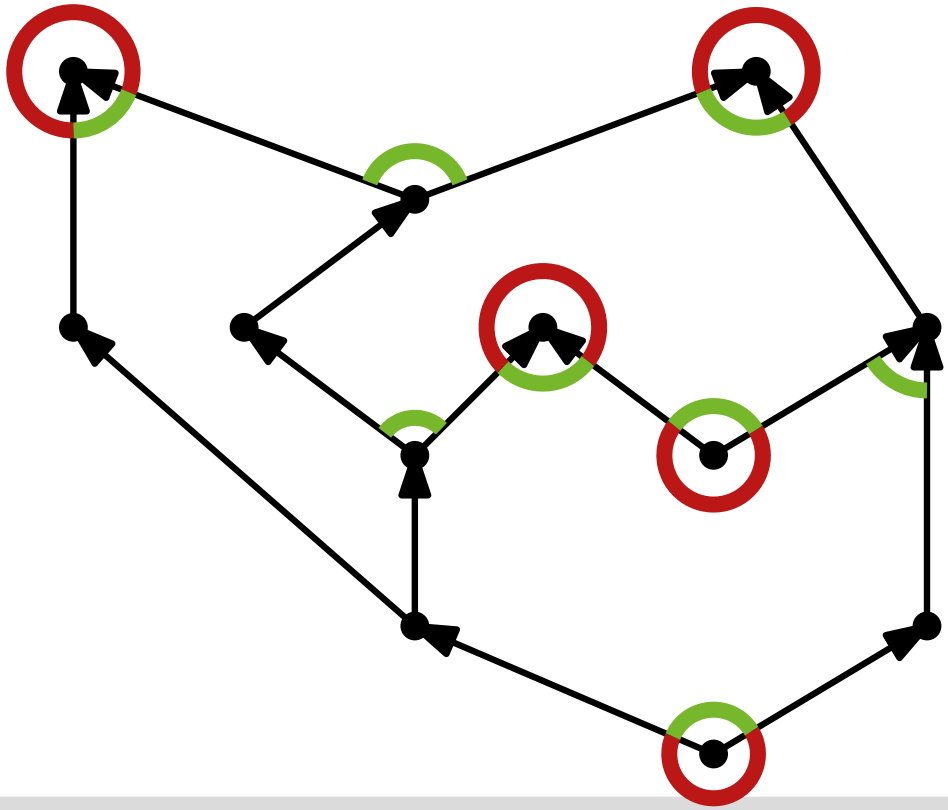
- Bimodalität ist notwendig (aber nicht hinreichend)



Beobachtungen

- Bimodalität ist notwendig (aber nicht hinreichend)
- betrachte Winkel zwischen zwei ein-/ausgehenden Kanten
  Winkel α ist **groß** wenn $\alpha > \pi$, **klein** sonst
  $L(v) :=$ # große Winkel an Knoten v
  $L(f) :=$ # große Winkel in Facette f
  $S(v)$ bzw. $S(f)$: Anzahl **kleiner** Winkel

nur zw. ein- bzw. ausgehenden Kanten!



Beobachtungen

- Bimodalität ist notwendig (aber nicht hinreichend)
- betrachte Winkel zwischen zwei ein-/ausgehenden Kanten
  Winkel α ist **groß** wenn $\alpha > \pi$, **klein** sonst
  $L(v) := \#$ große Winkel an Knoten v
  $L(f) := \#$ große Winkel in Facette f
  $S(v)$ bzw. $S(f)$: Anzahl **kleiner** Winkel

nur zw. ein- bzw. ausgehenden Kanten!

Lemma 1: In jedem Aufwärts-Layout von D gilt:

$$(1)\ \forall v \in V : L(v) = \begin{cases} 0 & v \text{ innerer Knoten} \\ 1 & v \text{ Quelle/Senke} \end{cases}$$

$$(2)\ \forall f \in \mathcal{F} : L(f) - S(f) = \begin{cases} -2 & f \neq f_0 \\ 2 & f = f_0 \end{cases}$$

Beobachtungen

- $A(f) := \#$ Winkel zwischen zwei eingehenden Kanten an f
  Es gilt stets: $L(f) + S(f) = 2A(f)$ für alle Facetten.

- in jedem Aufwärts-Layout von D gilt:
  $$\forall f \in \mathcal{F} : L(f) = \begin{cases} A(f) - 1 & f \neq f_0 \\ A(f) + 1 & f = f_0 \end{cases}$$

Beobachtungen

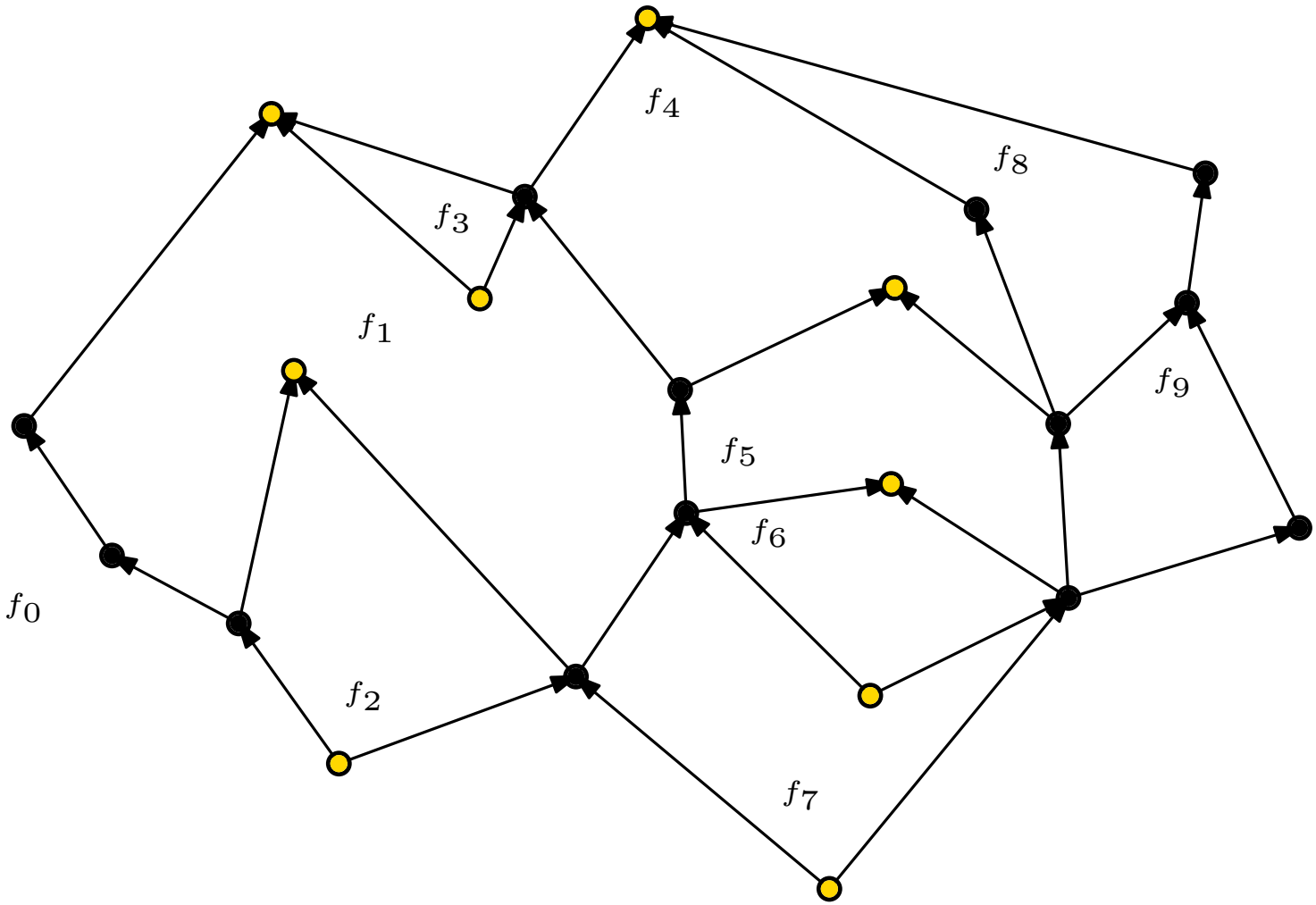
- $A(f) := \#$ Winkel zwischen zwei eingehenden Kanten an f
  Es gilt stets: $L(f) + S(f) = 2A(f)$ für alle Facetten.

- in jedem Aufwärts-Layout von D gilt:
  $$\forall f \in \mathcal{F} : L(f) = \begin{cases} A(f) - 1 & f \neq f_0 \\ A(f) + 1 & f = f_0 \end{cases}$$

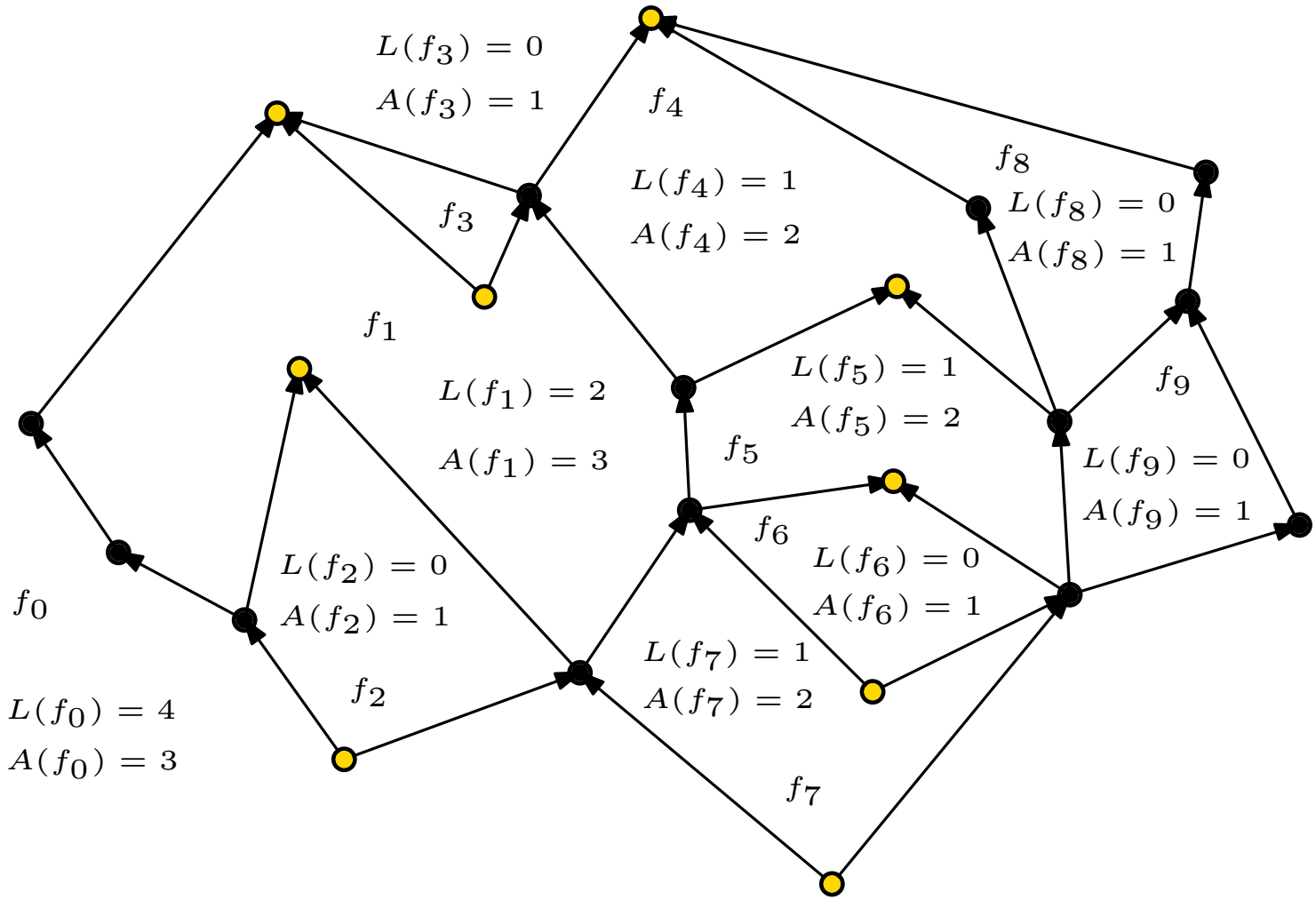
- Definiere Abbildung $\Phi : Q \cup S \to \mathcal{F}$
  (Q Menge der Quellen, S Menge der Senken), wobei
  $\Phi : v \mapsto$ inzidente Facette mit großem Winkel an v

- Φ heißt **konistent**, falls $|\Phi^{-1}(f)| = \begin{cases} A(f) - 1 & f \neq f_0 \\ A(f) + 1 & f = f_0 \end{cases}$

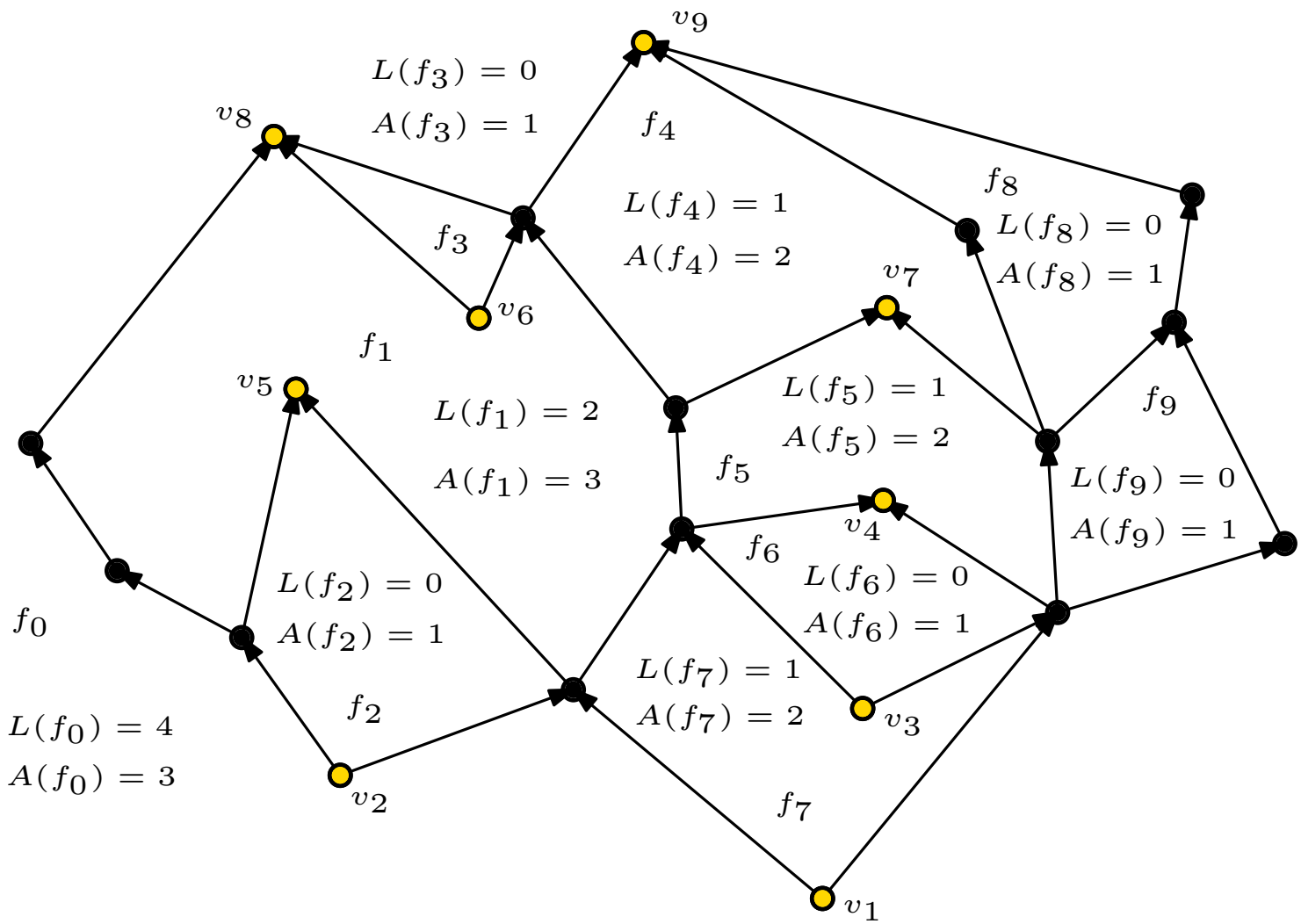
Beispiel Facettenzuordnung



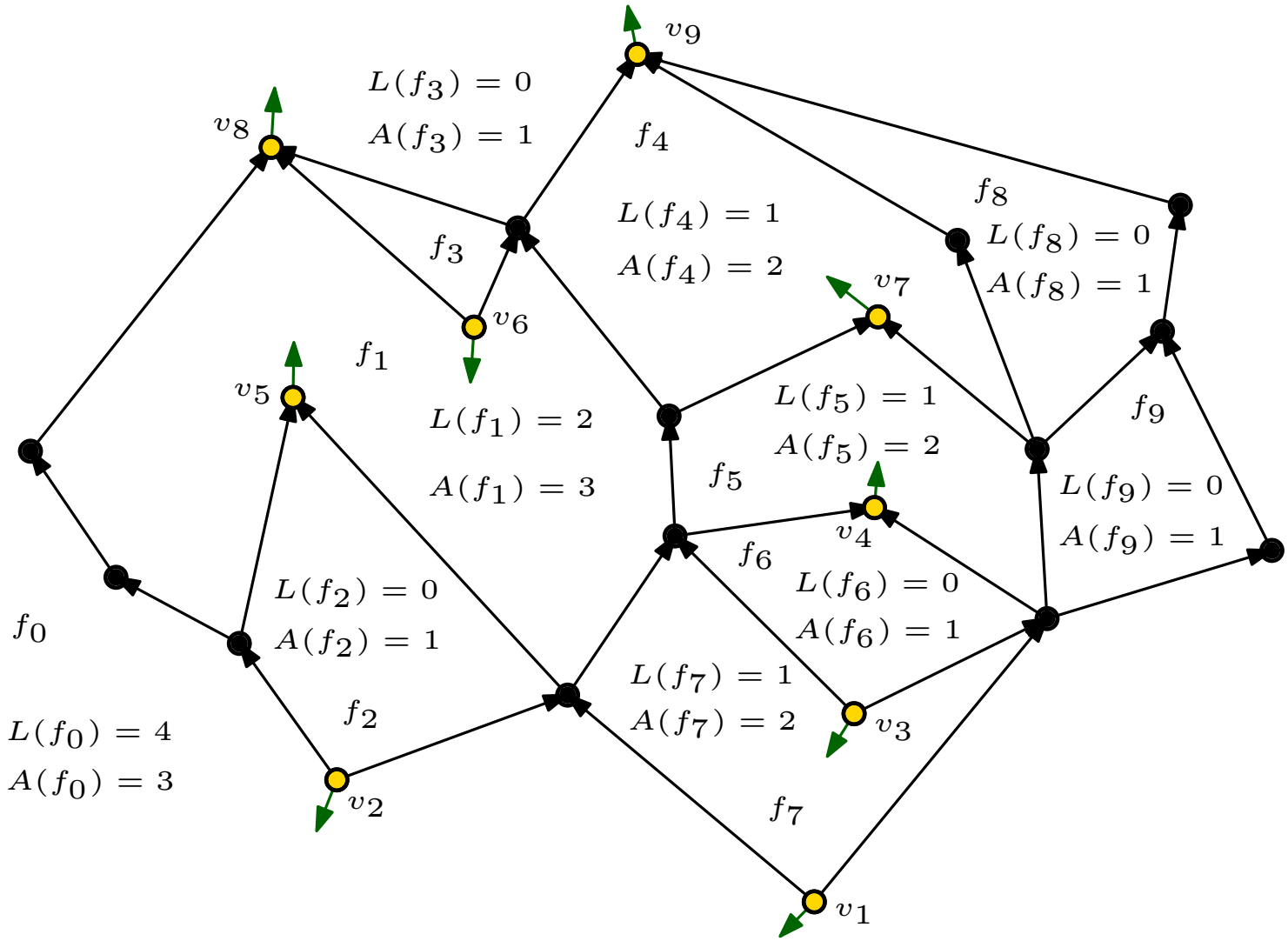
Beispiel Facettenzuordnung



Beispiel Facettenzuordnung



Beispiel Facettenzuordnung



$$\Phi(v_1) = f_0$$
$$\Phi(v_2) = f_0$$
$$\Phi(v_3) = f_7$$
$$\Phi(v_4) = f_5$$
$$\Phi(v_5) = f_1$$
$$\Phi(v_6) = f_1$$
$$\Phi(v_7) = f_4$$
$$\Phi(v_8) = f_0$$
$$\Phi(v_9) = f_0$$

Charakterisierung

Satz 6: Für einen gerichteten azyklischen Graphen $D = (V, A)$ mit kombinatorischer Einbettung $\mathcal{F}, f_0$ gilt:
D aufwärtsplanar $\Leftrightarrow$ D bimodal und $\exists$ konsistentes Φ

Charakterisierung

Satz 6: Für einen gerichteten azyklischen Graphen $D = (V, A)$ mit kombinatorischer Einbettung $\mathcal{F}, f_0$ gilt:
D aufwärtsplanar $\Leftrightarrow$ D bimodal und $\exists$ konsistentes Φ

Beweis:

$\Rightarrow$ gerade hergeleitet

Charakterisierung

Satz 6: Für einen gerichteten azyklischen Graphen $D = (V, A)$ mit kombinatorischer Einbettung $\mathcal{F}, f_0$ gilt:
D aufwärtsplanar $\Leftrightarrow$ D bimodal und $\exists$ konsistentes Φ

Beweis:

$\Rightarrow$ gerade hergeleitet

$\Leftarrow$ konstruiere enthaltenden planaren st-Graphen

Charakterisierung

Satz 6: Für einen gerichteten azyklischen Graphen $D = (V, A)$ mit kombinatorischer Einbettung $\mathcal{F}, f_0$ gilt:
D aufwärtsplanar $\Leftrightarrow$ D bimodal und $\exists$ konsistentes Φ

Beweis:

$\Rightarrow$ gerade hergeleitet

$\Leftarrow$ konstruiere enthaltenden planaren st-Graphen

- Frage zunächst: Gibt es für $D, \mathcal{F}, f_0$ ein konsistentes Φ?

Charakterisierung

Satz 6: Für einen gerichteten azyklischen Graphen $D = (V, A)$ mit kombinatorischer Einbettung $\mathcal{F}, f_0$ gilt:
D aufwärtsplanar $\Leftrightarrow$ D bimodal und $\exists$ konsistentes Φ

Beweis:

$\Rightarrow$ gerade hergeleitet

$\Leftarrow$ konstruiere enthaltenden planaren st-Graphen

- Frage zunächst: Gibt es für $D, \mathcal{F}, f_0$ ein konsistentes Φ?

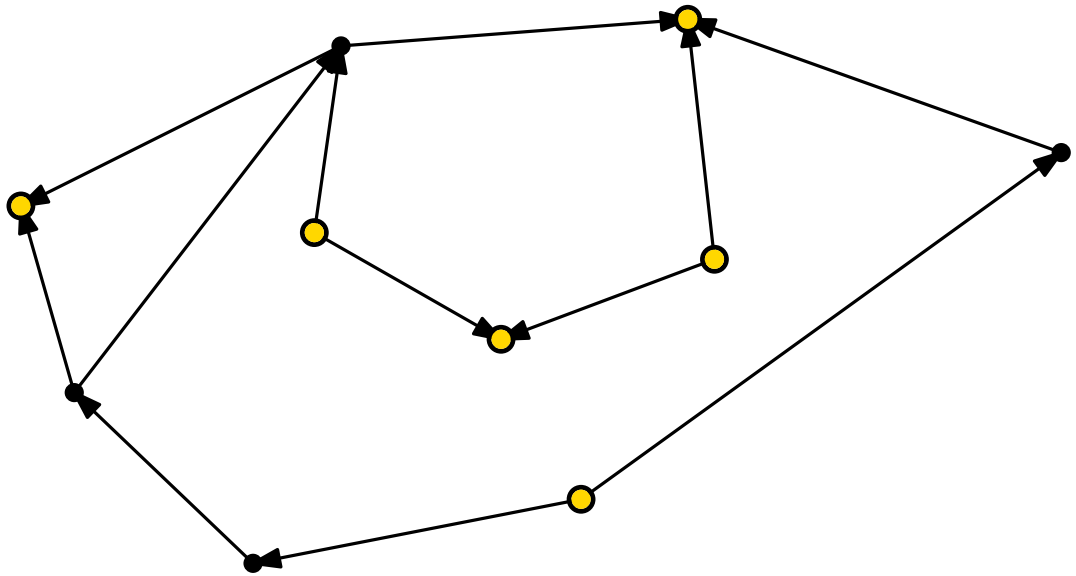
Antwort: Flussnetzwerk!

Flussnetzwerk

Def: Flussnetzwerk $N(D, \mathcal{F}, f_0) = ((W, A_N); \ell; u; b)$

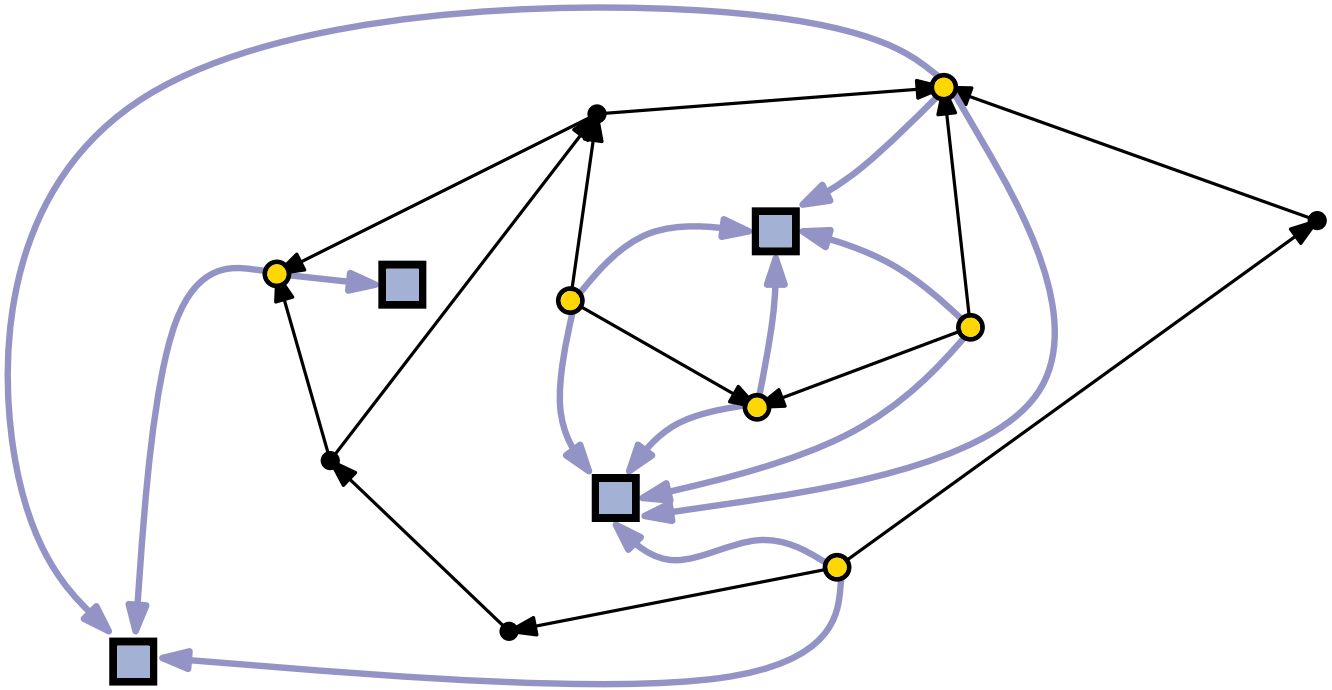
- $W = \{v \in V \mid v \text{ ist Quelle oder Senke}\} \cup \mathcal{F}$
- $A_N = \{(v, f) \mid v \text{ inzident zu } f\}$
- $l(a) = 0 \quad \forall a \in A_N$
- $u(a) = 1 \quad \forall a \in A_N$
- $b(q) = \begin{cases} 1 & \forall q \in W \cap V \\ -(A(q) - 1) & \forall q \in \mathcal{F} \setminus \{f_0\} \\ -(A(q) + 1) & q = f_0 \end{cases}$

Beispielnetzwerk



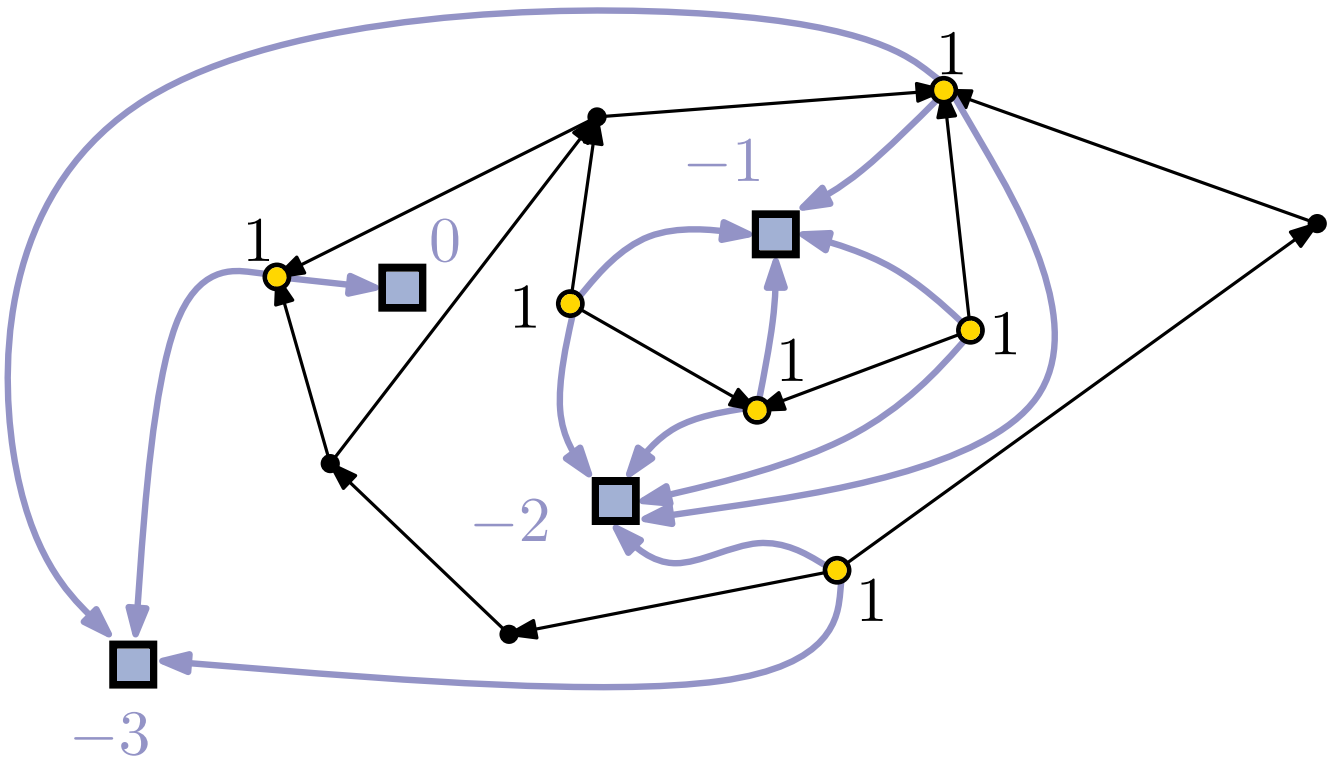
- normaler Knoten
- Quelle / Senke

Beispielnetzwerk



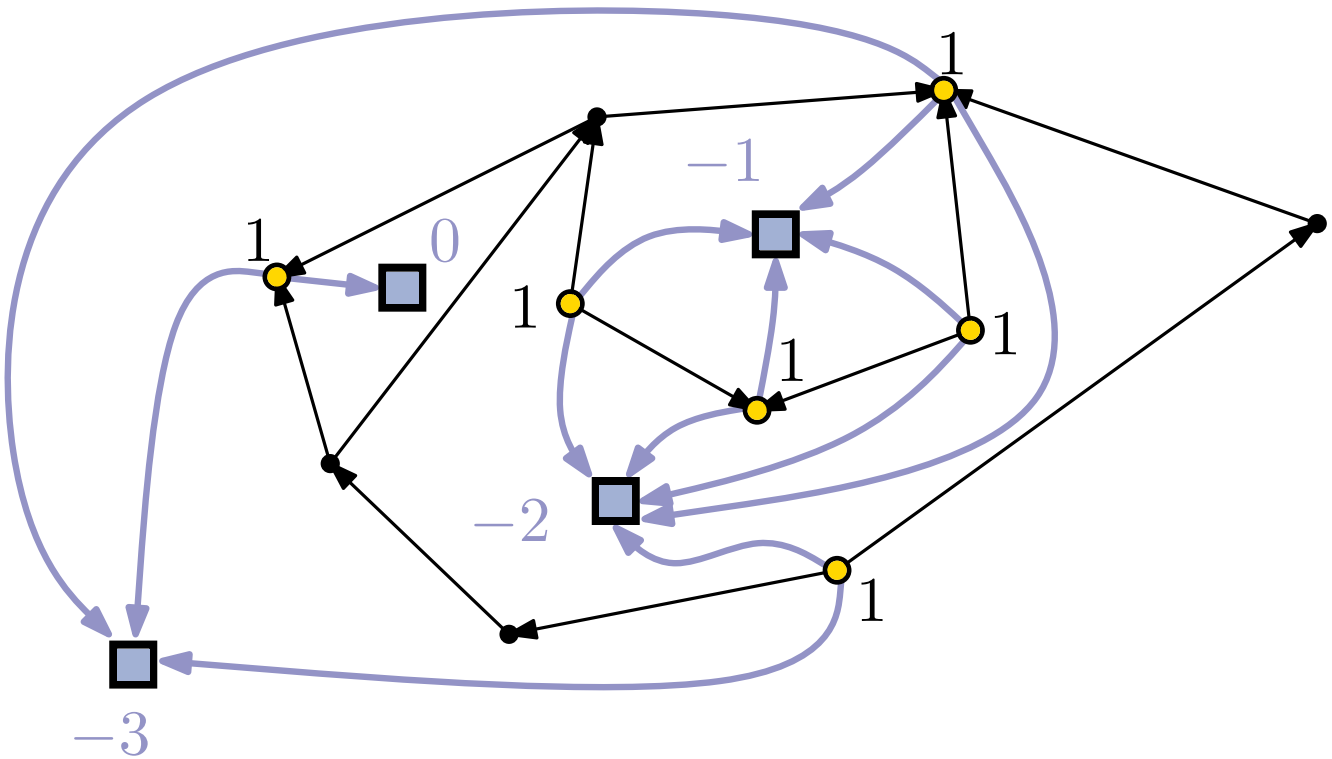
- normaler Knoten
- Quelle / Senke
- Facettenknoten

Beispielnetzwerk



- normaler Knoten
- Quelle / Senke
- Facettenknoten

Beispielnetzwerk

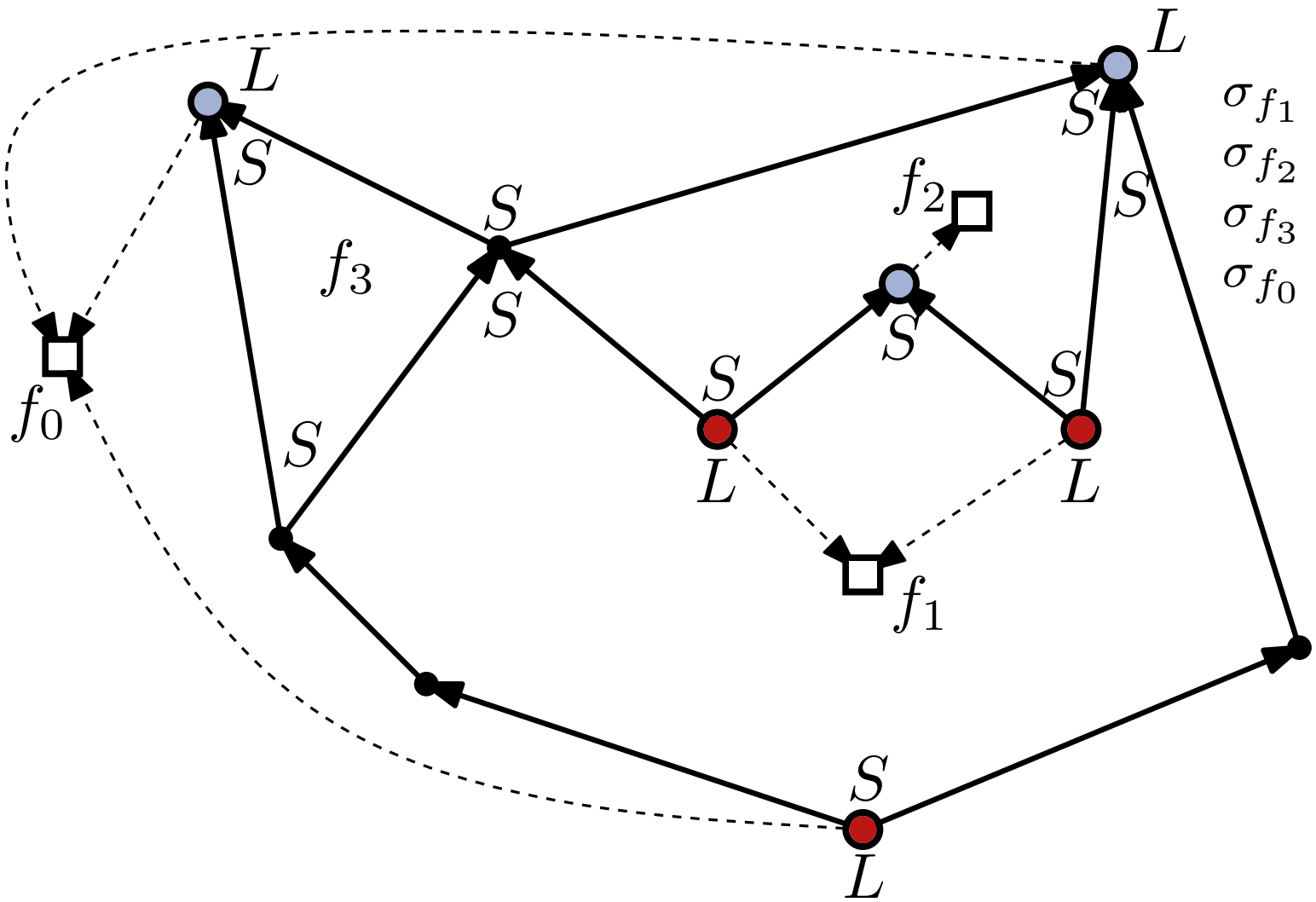


- normaler Knoten
- Quelle / Senke
- Facettenknoten

- starte mit Nullfluss
- suche erhöhende Wege (r-mal für insg. r Quellen/Senken)
- geht auch ohne festgelegtes f_0: zunächst alle Facetten als intern betrachten, danach alle Facetten für gültige Flusserhöhung um 2 Einheiten prüfen

Winkelfolgen an Facetten

Betrachte Folge σ_f von Winkelmarkierungen L, S im UZS an lokalen Quellen und Senken von f



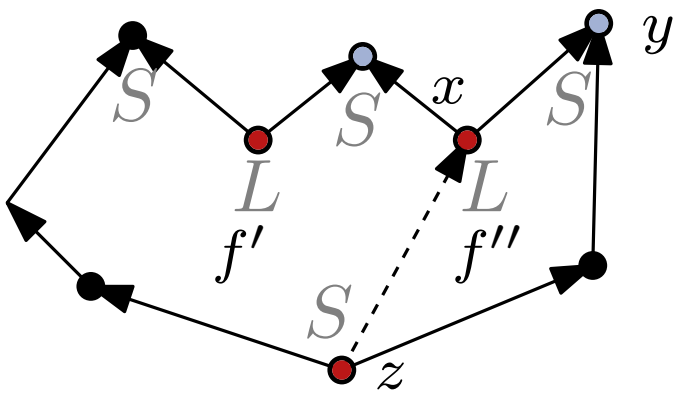
$$\sigma_{f_1} := (S, S, L, S, L, S)$$
$$\sigma_{f_2} := (L, S, S, S)$$
$$\sigma_{f_3} := (S, S)$$
$$\sigma_{f_0} := (L, L, S, L)$$

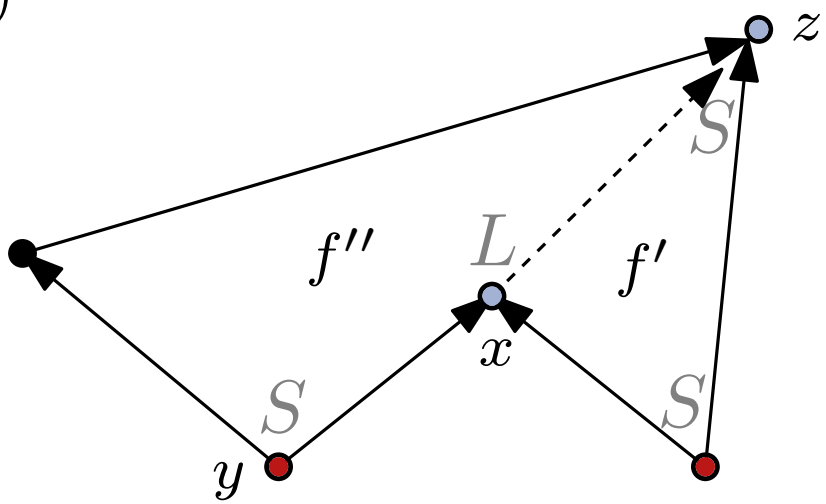
Konstruktion planarer st-Graph

Gegeben ein planarer, bimodaler, azyklischer, gerichteter Graph $D = (V, A)$ mit Einbettung $\mathcal{F}, f_0$ und konsistenter Facettenzuweisung Φ, konstruiere planaren st-Graph, der D enthält.

Ziel: entferne alle Quellen und Senken

- nach Lemma 1 muss σ_f mit $|\sigma_f| > 2$ für $f \neq f_0$ eine Teilsequenz LSS an drei Knoten x, y, z enthalten
- ist x Quelle $\Rightarrow$ verfeinere mit (z, x)
- ist x Senke $\Rightarrow$ verfeinere mit (x, z)



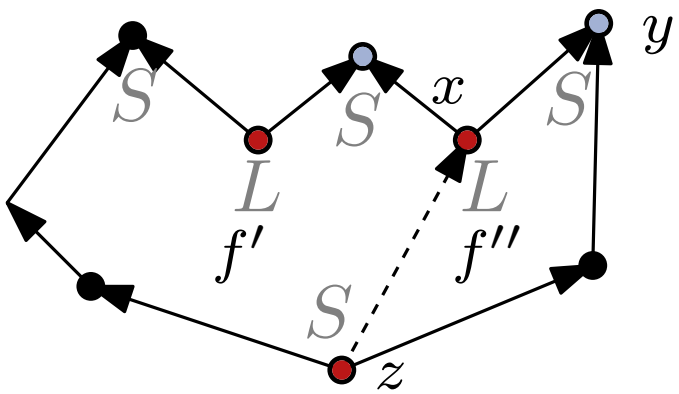


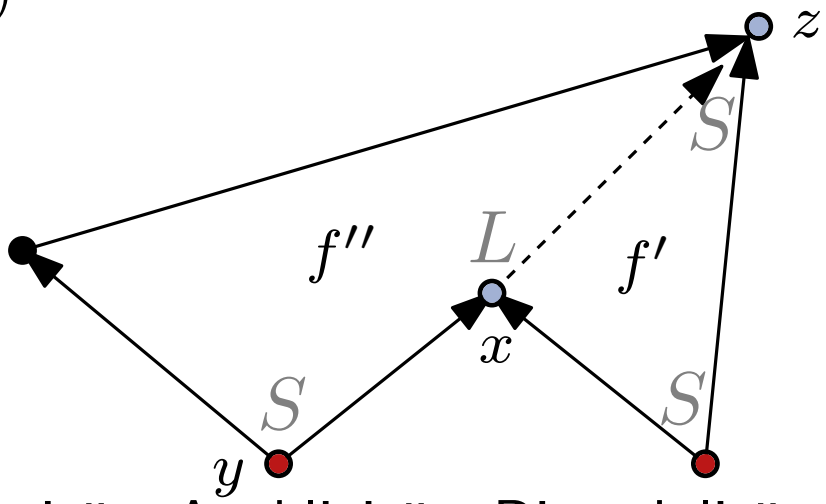
Konstruktion planarer st-Graph

Gegeben ein planarer, bimodaler, azyklischer, gerichteter Graph $D = (V, A)$ mit Einbettung $\mathcal{F}, f_0$ und konsistenter Facettenzuweisung Φ, konstruiere planaren st-Graph, der D enthält.

Ziel: entferne alle Quellen und Senken

- nach Lemma 1 muss σ_f mit $|\sigma_f| > 2$ für $f \neq f_0$ eine Teilsequenz LSS an drei Knoten x, y, z enthalten
- ist x Quelle $\Rightarrow$ verfeinere mit (z, x)
- ist x Senke $\Rightarrow$ verfeinere mit (x, z)





- Invarianten der Konstruktion: Planarität, Azyklizität, Bimodalität; verbleibende Quellen/Senken von f liegen in f'
- wähle verbleibende Superquelle s und Supersenke t auf f_0 und füge (s, s') für alle Quellen $s' \neq s$, (t', t) für alle Senken $t' \neq t$ und (s, t) ein

Charakterisierung

Satz 6: Für einen gerichteten azyklischen Graphen $D = (V, A)$ mit kombinatorischer Einbettung $\mathcal{F}, f_0$ gilt:
D aufwärtsplanar $\Leftrightarrow$ D bimodal und $\exists$ konsistentes Φ

Beweis:

$\Rightarrow$ vorhin hergeleitet

$\Leftarrow$ konstruiere enthaltenden planaren st-Graphen ✓

- mit Satz 5 folgt, dass D als aufspannender Teilgraph eines planaren st-Graphen auch aufwärtsplanar ist

$\square$

Zusammenfassung

Satz 4: Gegeben ein gerichteter azyklischer Graph $D = (V, A)$, ist es NP-vollständig zu entscheiden, ob D aufwärtsplanar ist. [Garg, Tamassia GD 1995]

Satz 6: Für einen gerichteten azyklischen Graphen $D = (V, A)$ mit kombinatorischer Einbettung $\mathcal{F}, f_0$ gilt:
D aufwärtsplanar $\Leftrightarrow$ D bimodal und $\exists$ konsistentes Φ

[Bertolazzi et al. Algorithmica 1994]

Zusammenfassung

Satz 4: Gegeben ein gerichteter azyklischer Graph $D = (V, A)$, ist es NP-vollständig zu entscheiden, ob D aufwärtsplanar ist. [Garg, Tamassia GD 1995]

Satz 6: Für einen gerichteten azyklischen Graphen $D = (V, A)$ mit kombinatorischer Einbettung $\mathcal{F}, f_0$ gilt:
D aufwärtsplanar $\Leftrightarrow$ D bimodal und $\exists$ konsistentes Φ
[Bertolazzi et al. Algorithmica 1994]

- Es gibt Festparameter-Algorithmen zum Testen der Aufwärtsplanarität [Healy, Lynch SOFSEM 2005]
- Die Entscheidung in Satz 6 kann in $O(n + r^{1.5})$ Zeit getroffen werden, wobei $r = \#$ Quellen/Senken
[Abbasi, Healy, Rextin IPL 2010]
- viele verwandte Konzepte wurden in der Folge untersucht, z.B. Quasi-Aufwärtsplanarität, mixed Graphs etc.