Karlsruher Institut für Technologie

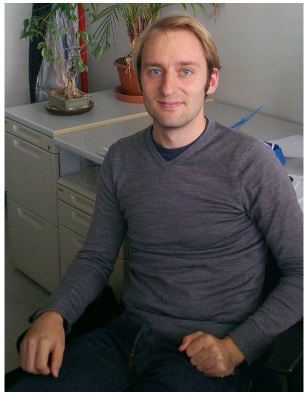
Algorithmische Geometrie
Übung am 16.04.2014

Institut für Theoretische Informatik · Prof. Dr. Dorothea Wagner

Übung für algorithmische Geometrie

Karlsruher Institut für Technologie

Übungsleiter



- Benjamin Niedermann
- `niedermann@kit.edu`
- Raum 322
- Sprechzeiten: individuell per Mail vereinbaren

Termine

- Vorlesung: Di 9:45 – 11:15 Uhr, Raum 301
- Übung: Mi 15:45 – 17:15 Uhr, Raum 301

Ablauf des Übungsbetrieb

Ausgabe Blatt 1

for *Woche* $i = 2 \ldots 14$ **do**

  if *Tag == Dienstag* **then**

    Ausgabe Blatt i

    Abgabe Blatt $(i-1)$

  if *Tag == Mittwoch & übungFindetStatt(Tag)* **then**

    Besprechung Blatt $(i-1)$

- Bearbeitung der Aufgaben und Abgabe der Lösungen in Zweiergruppen erwünscht
- Übung in der Regel wöchentlich ca. 45-60 Minuten
- abweichende Regelung werden rechtzeitig bekannt gegeben.

Nächste Woche (23. April) findet keine Übung statt!

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Karlsruher Institut für Technologie

Grundlegende Datenstrukturen und Techniken

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Unterteilung der Ebene

Quelle: Google Earth

Unterteilung der Ebene

Quelle: Google Earth

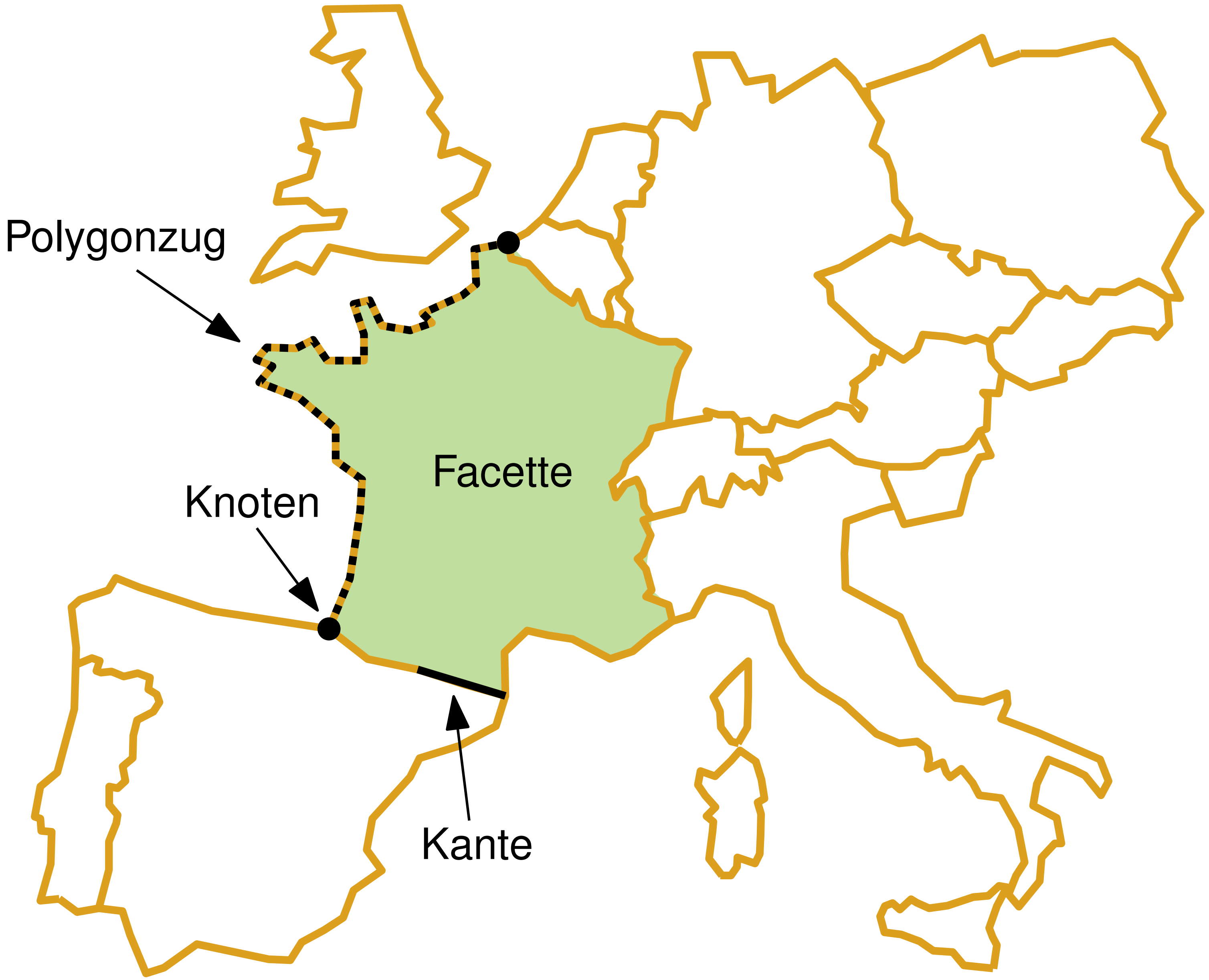
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Unterteilung der Ebene

Unterteilung der Ebene

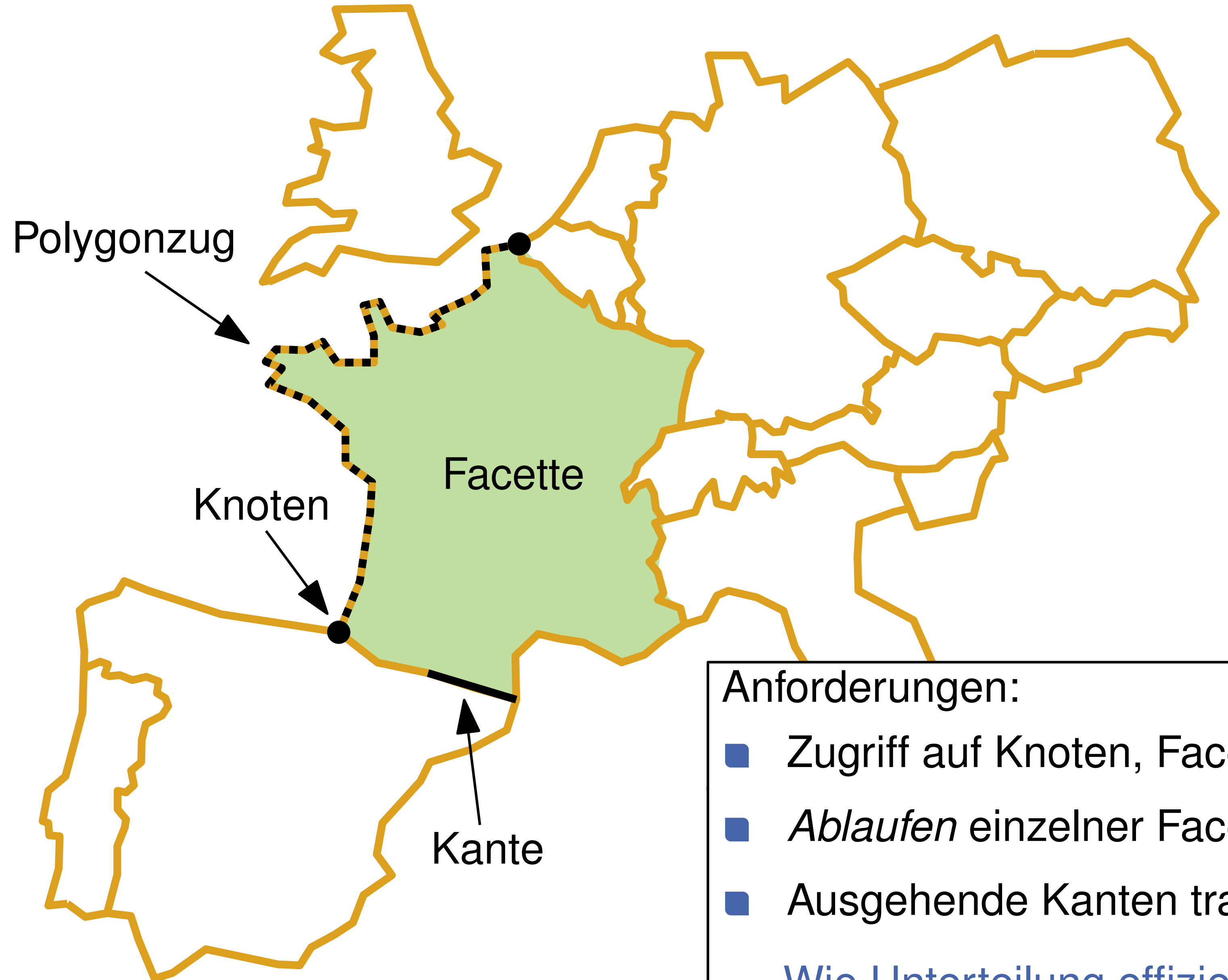
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Unterteilung der Ebene



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Unterteilung der Ebene



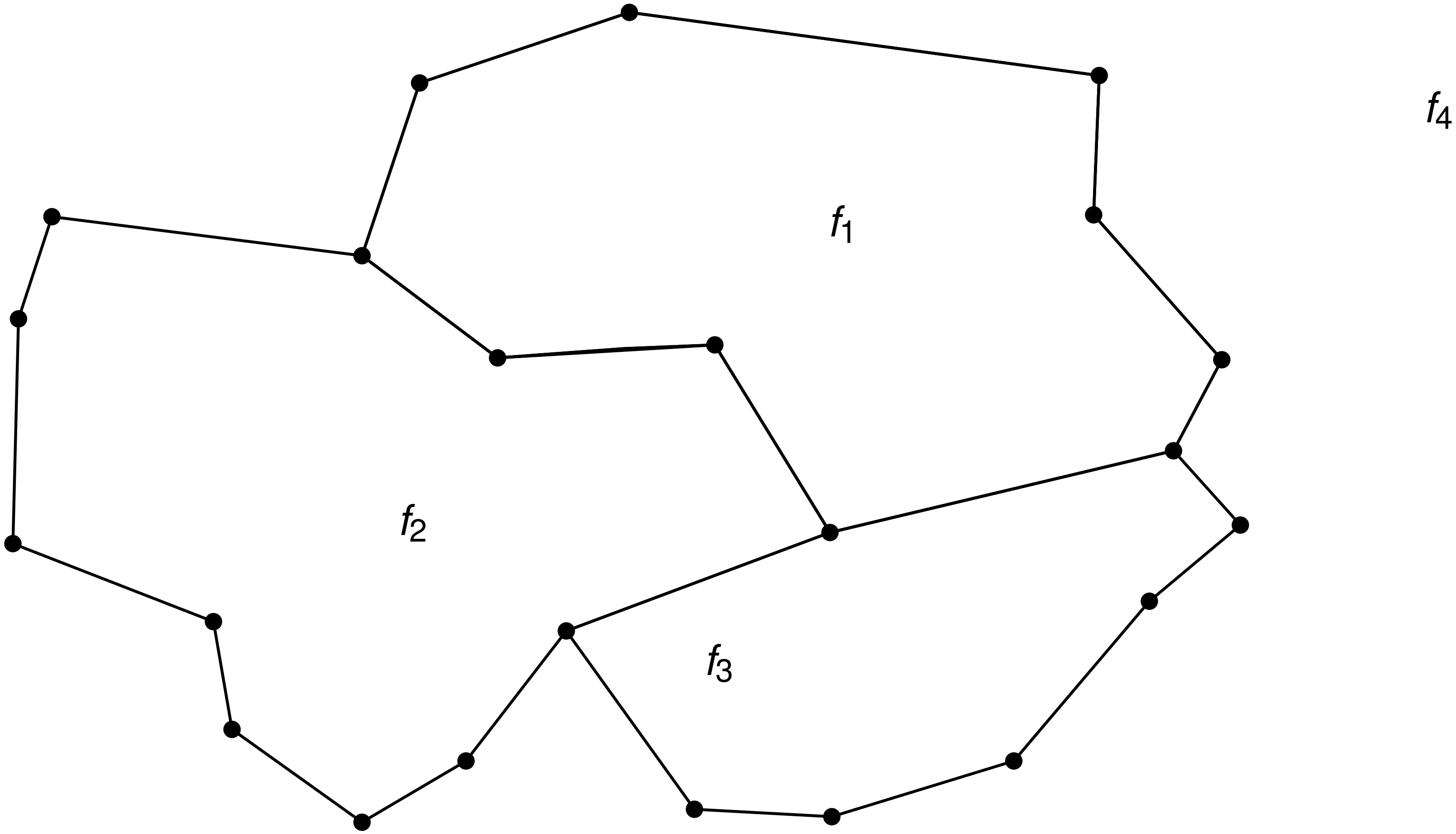
Anforderungen:

- Zugriff auf Knoten, Facetten und Kanten
- *Ablaufen* einzelner Facetten.
- Ausgehende Kanten traversieren.

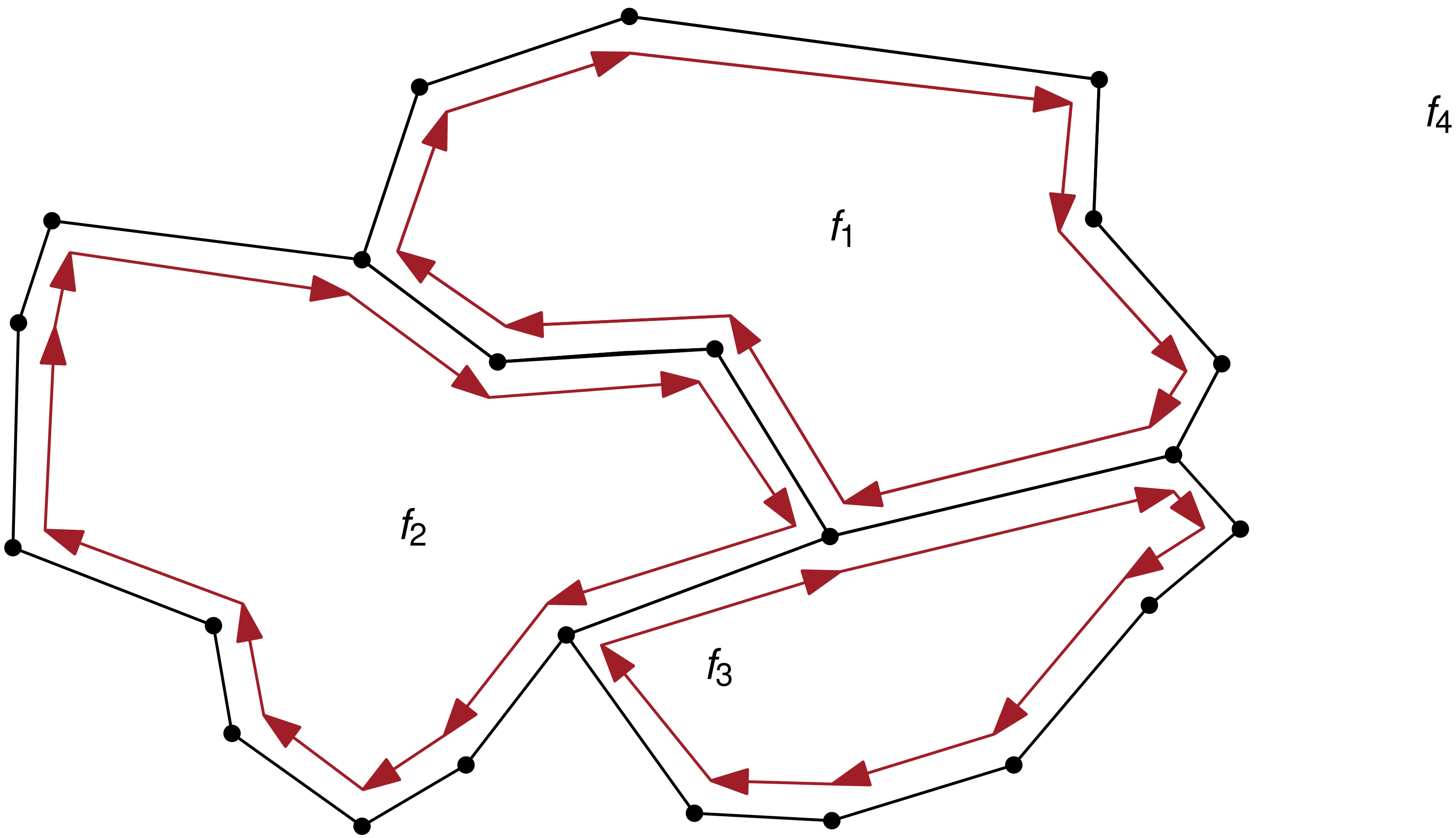
Wie Unterteilung effizient speichern?

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Unterteilung



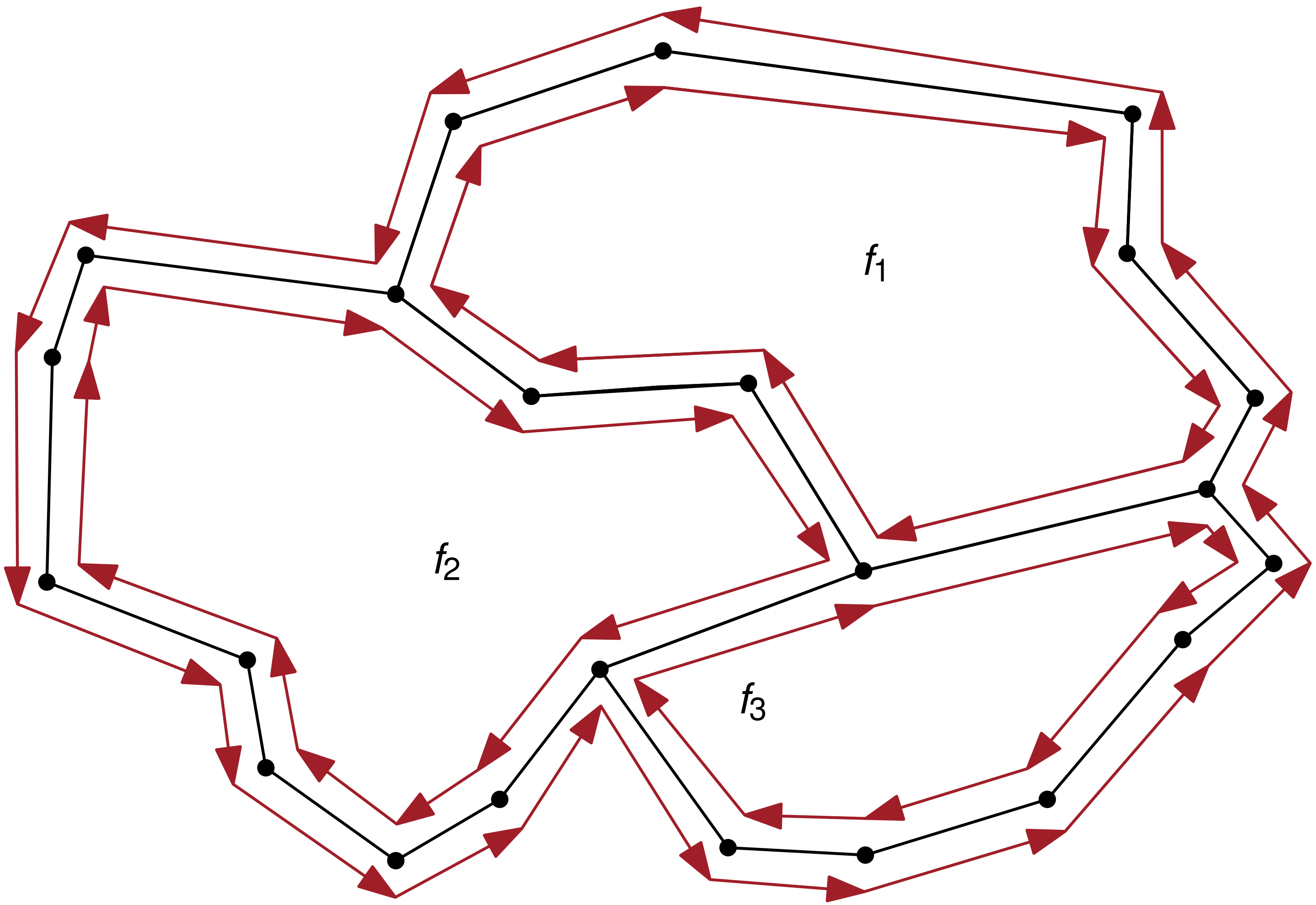
Unterteilung



Für jede Kante einer inneren Facette führe gerichtete Halbkante im Uhrzeigersinn ein.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Unterteilung

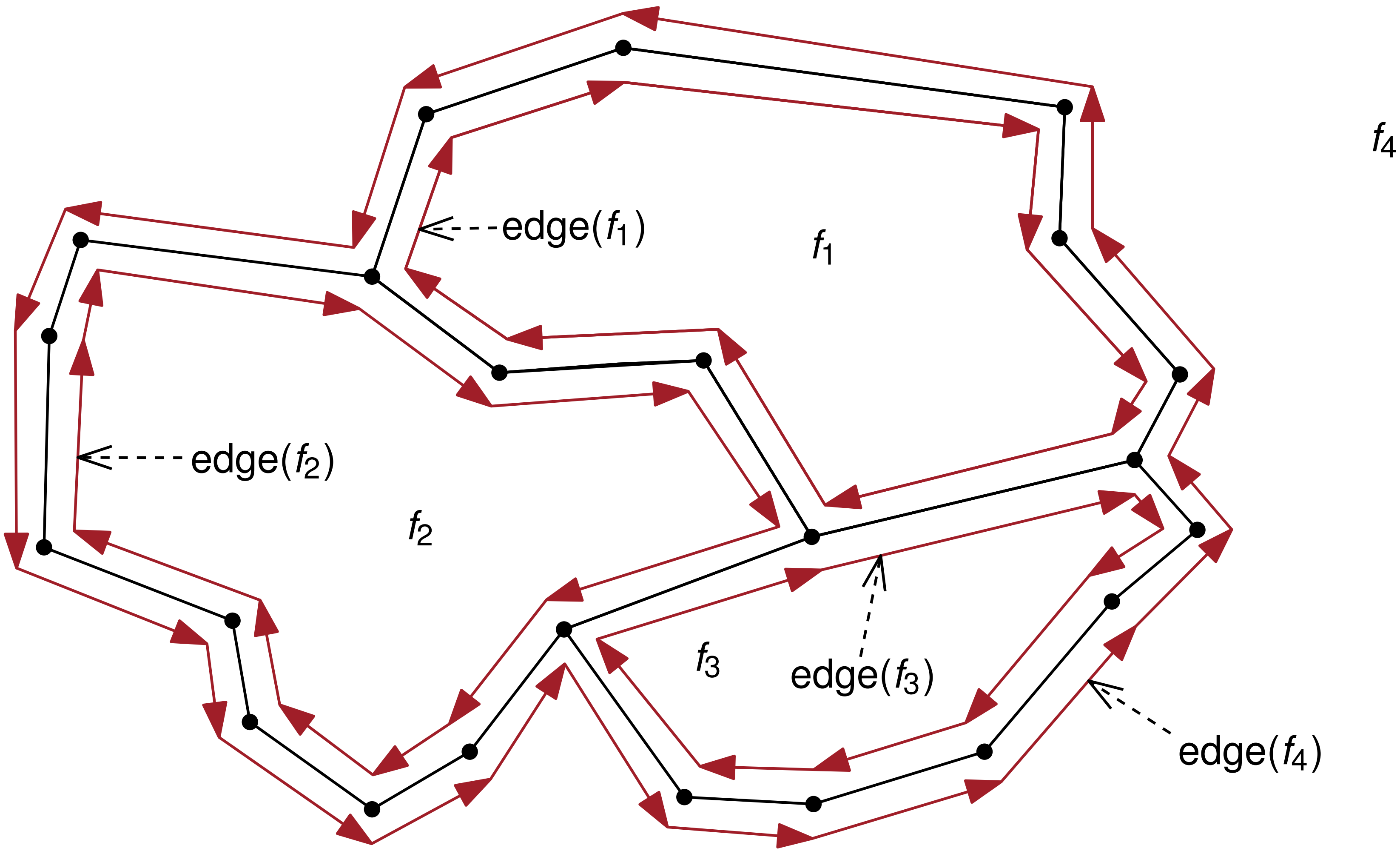


Für jede Kante einer inneren Facette führe gerichtete Halbkante im Uhrzeigersinn ein.

Für jede Kante der äußeren Facette führe gerichtete Halbkante gegen den Uhrzeigersinn ein.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

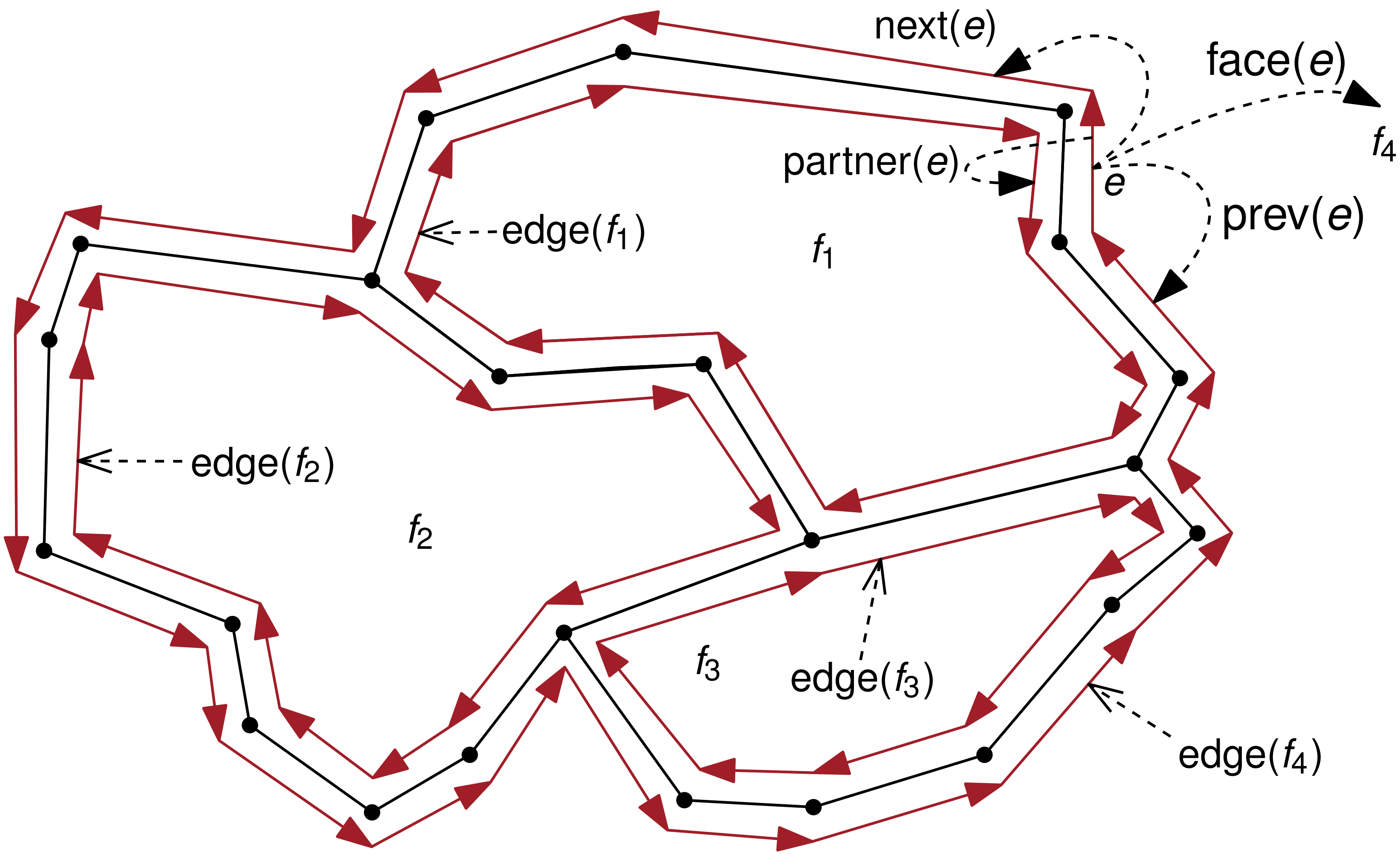
Unterteilung



Speichere für jede Facette eine beliebige angrenzende Halbkante.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

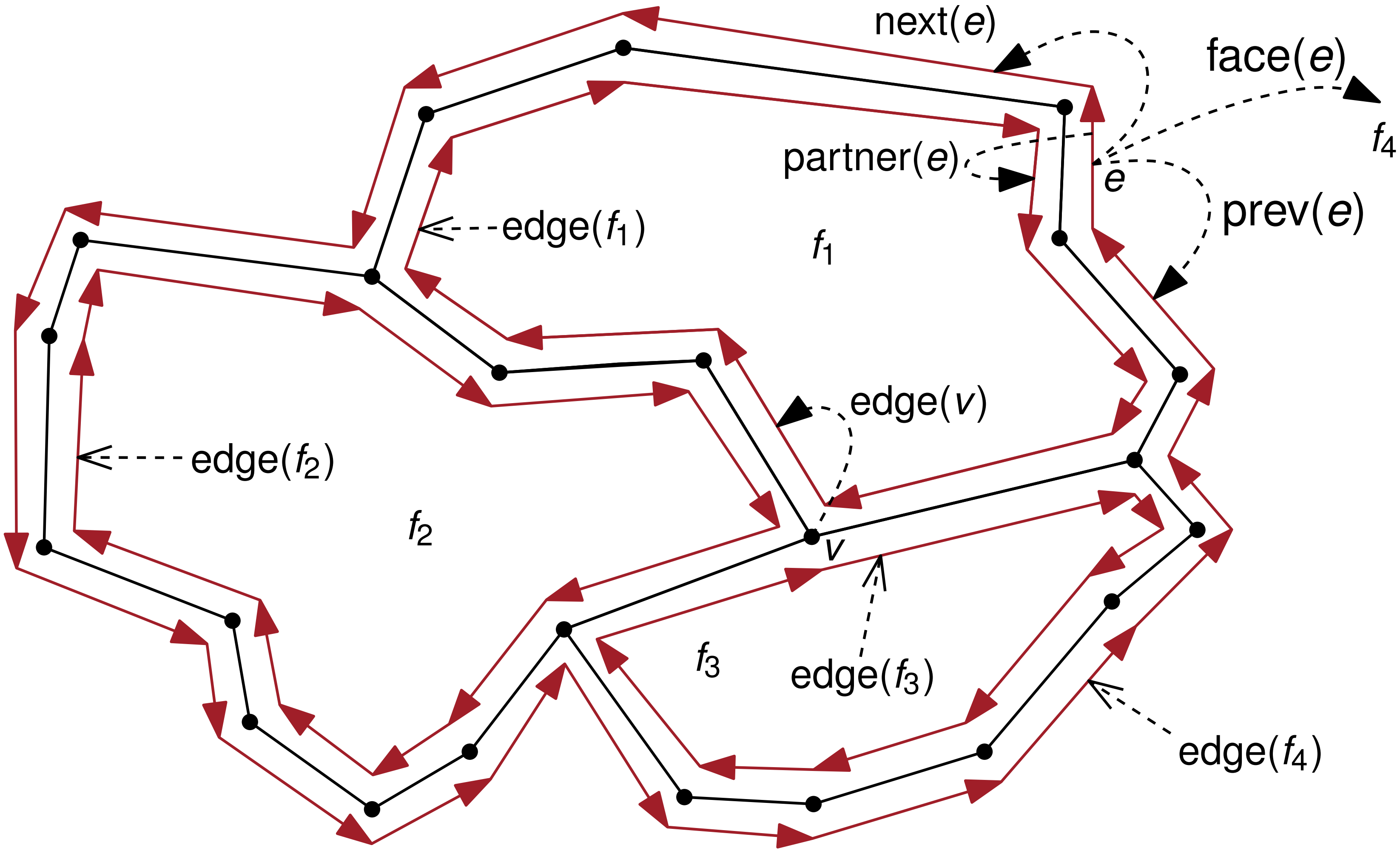
Unterteilung



Speichere für jede Facette eine beliebige angrenzende Halbkante.

Speichere für jede Halbkante den Nachfolger/Vorgänger, die gegenüberliegende Halbkante und die angrenzende Facette.

Unterteilung

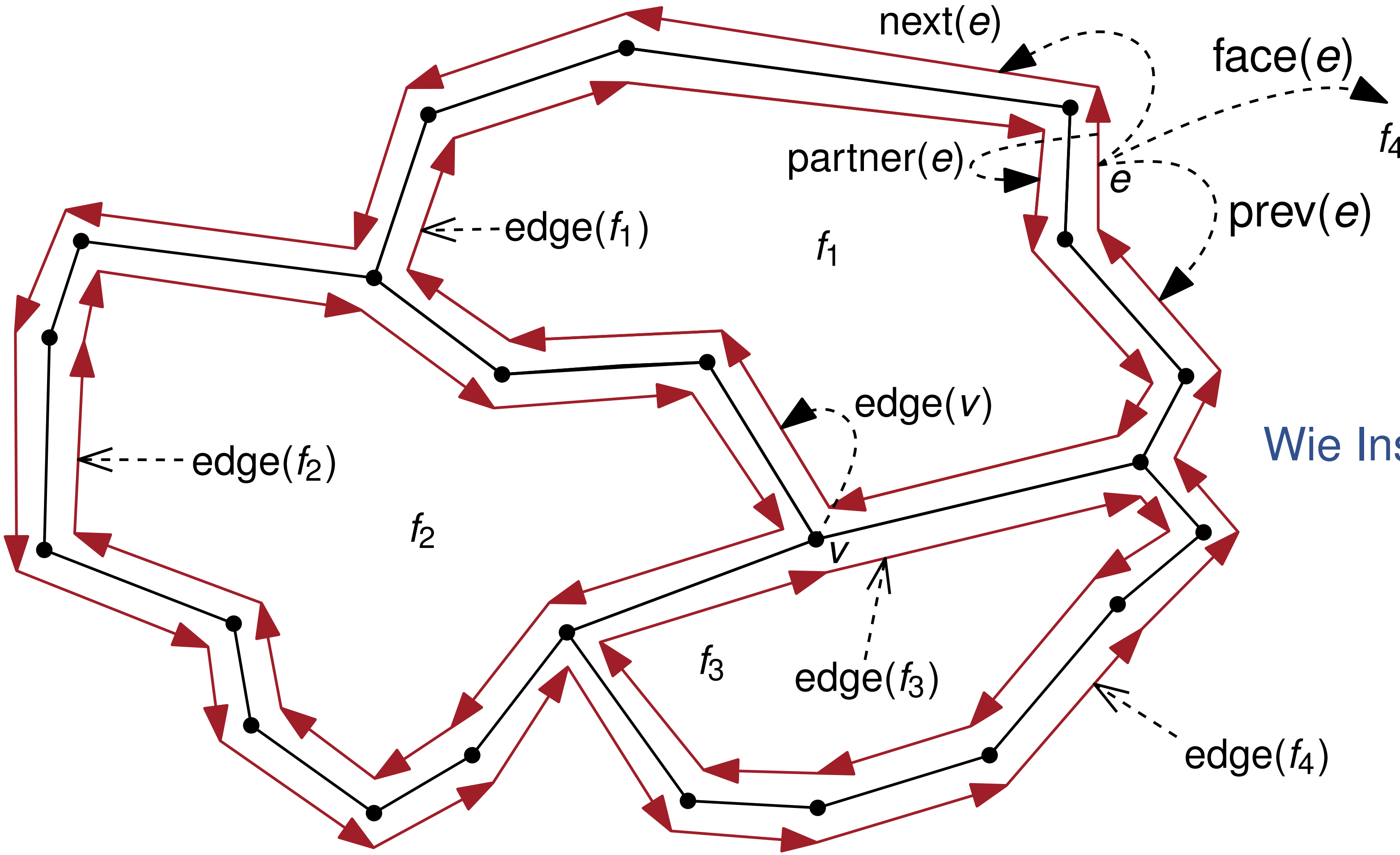


Speichere für jede Facette eine beliebige angrenzende Halbkante.

Speichere für jede Halbkante den Nachfolger/Vorgänger, die gegenüberliegende Halbkante und die angrenzende Facette.

Speichere für jeden Knoten eine beliebige inzidente ausgehende Halbkante.

Unterteilung



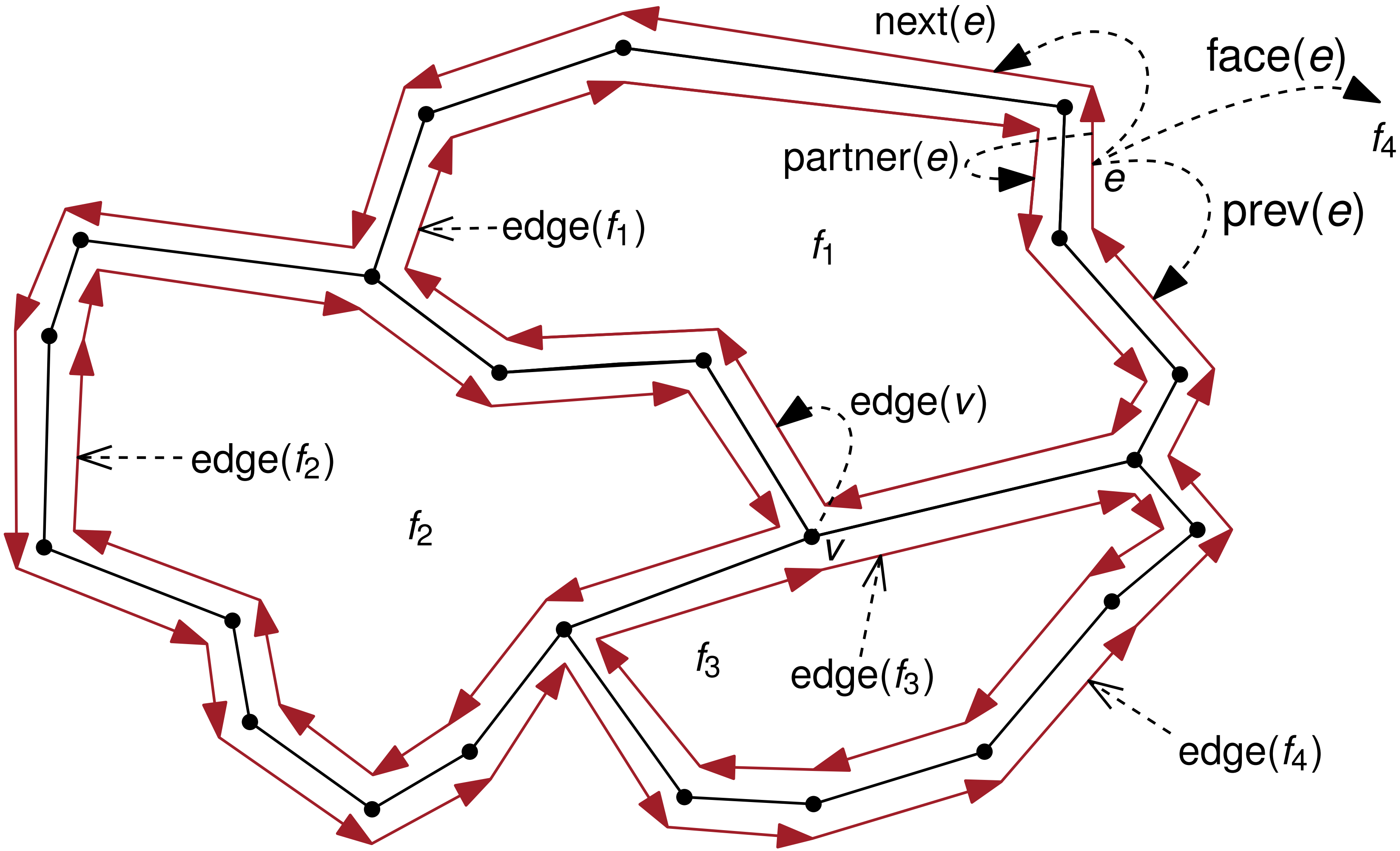
Wie Inseln behandeln?

Speichere für jede Facette eine beliebige angrenzende Halbkante.

Speichere für jede Halbkante den Nachfolger/Vorgänger, die gegenüberliegende Halbkante und die angrenzende Facette.

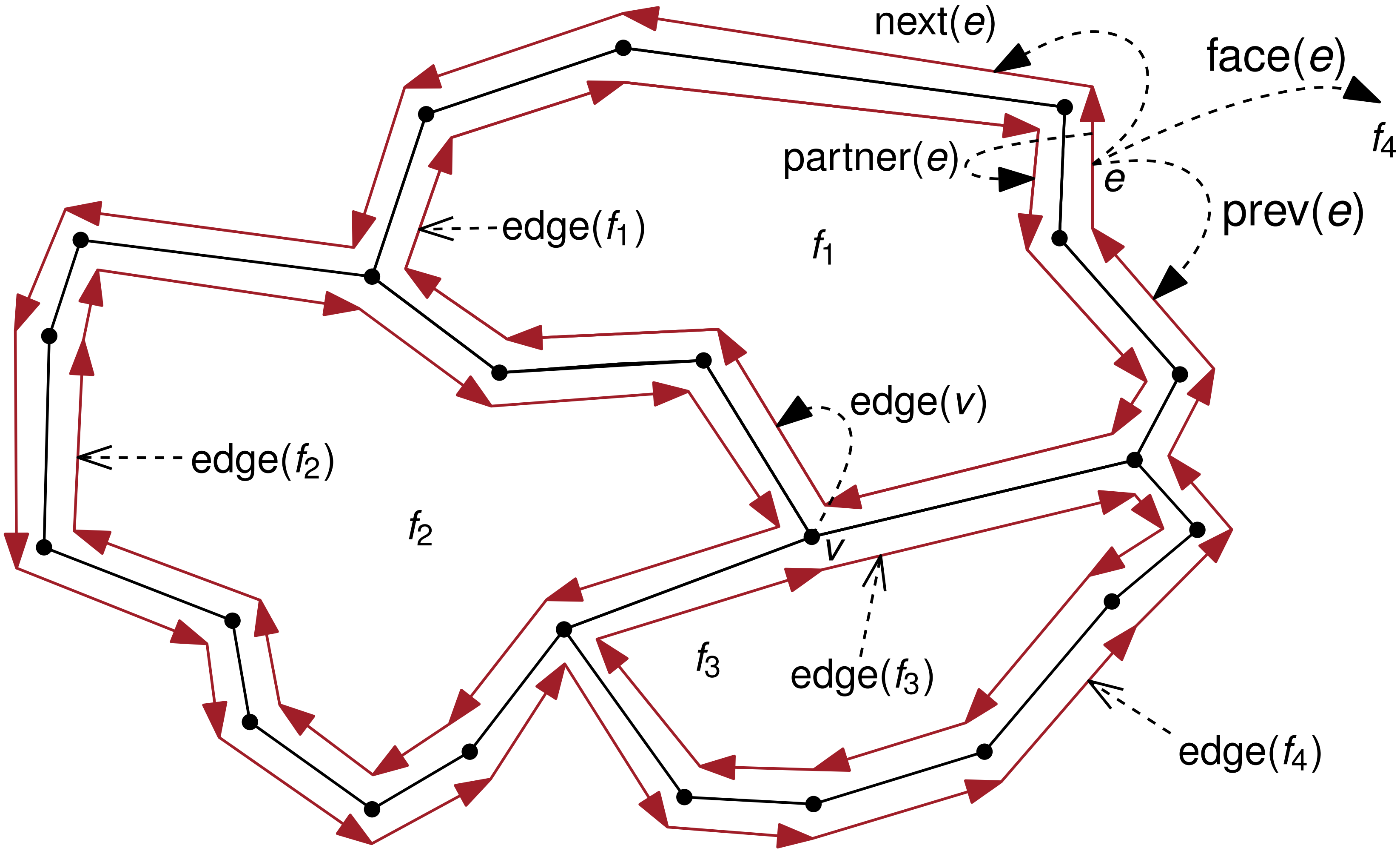
Speichere für jeden Knoten eine beliebige inzidente ausgehende Halbkante.

Unterteilung



- Zugriff auf Knoten, Facetten und Kanten
- *Ablaufen* einzelner Facetten.
- Ausgehende Kanten eines Knoten traversieren.

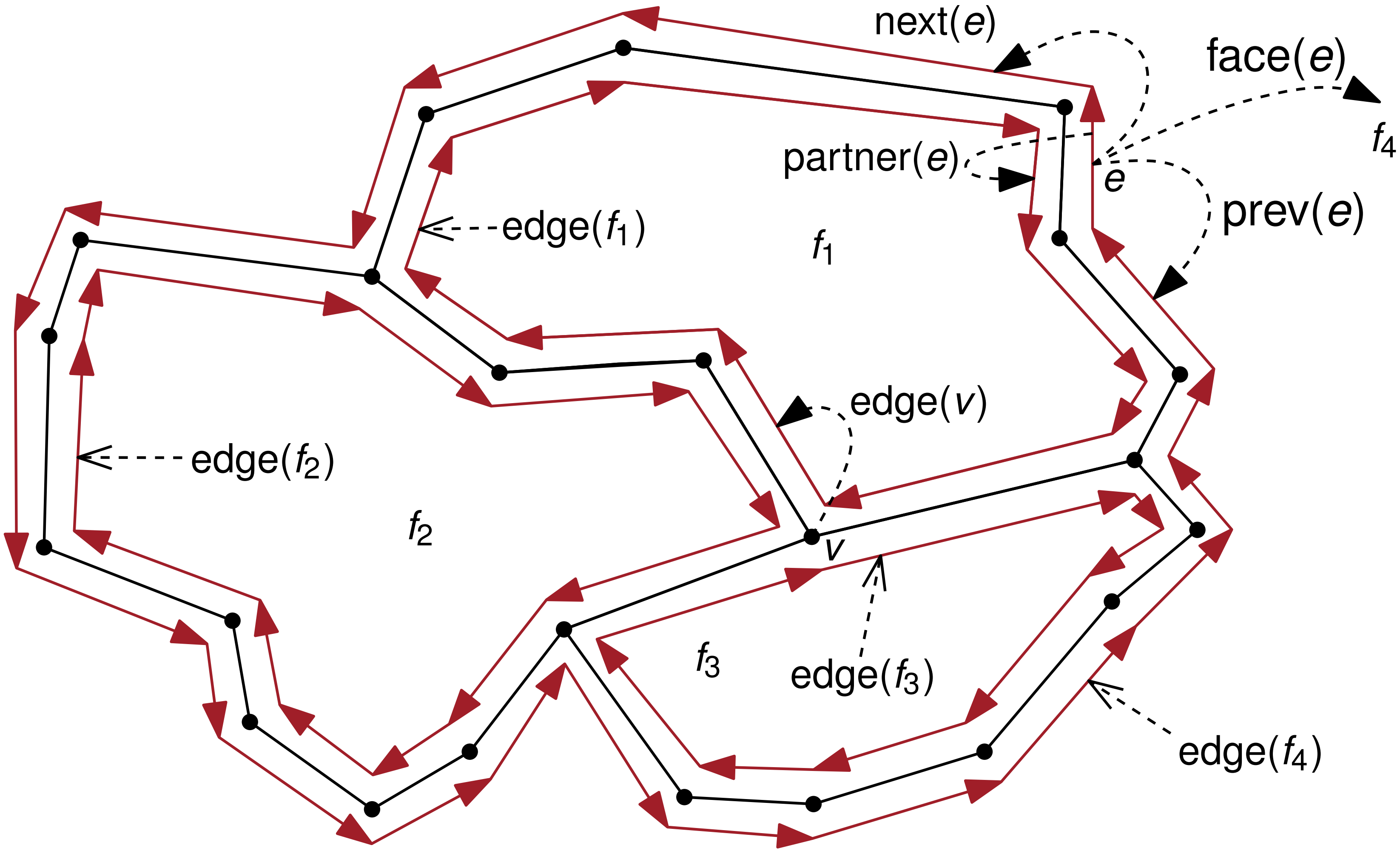
Unterteilung



- Zugriff auf Knoten, Facetten und Kanten ✓
- *Ablaufen* einzelner Facetten.
- Ausgehende Kanten eines Knoten traversieren.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

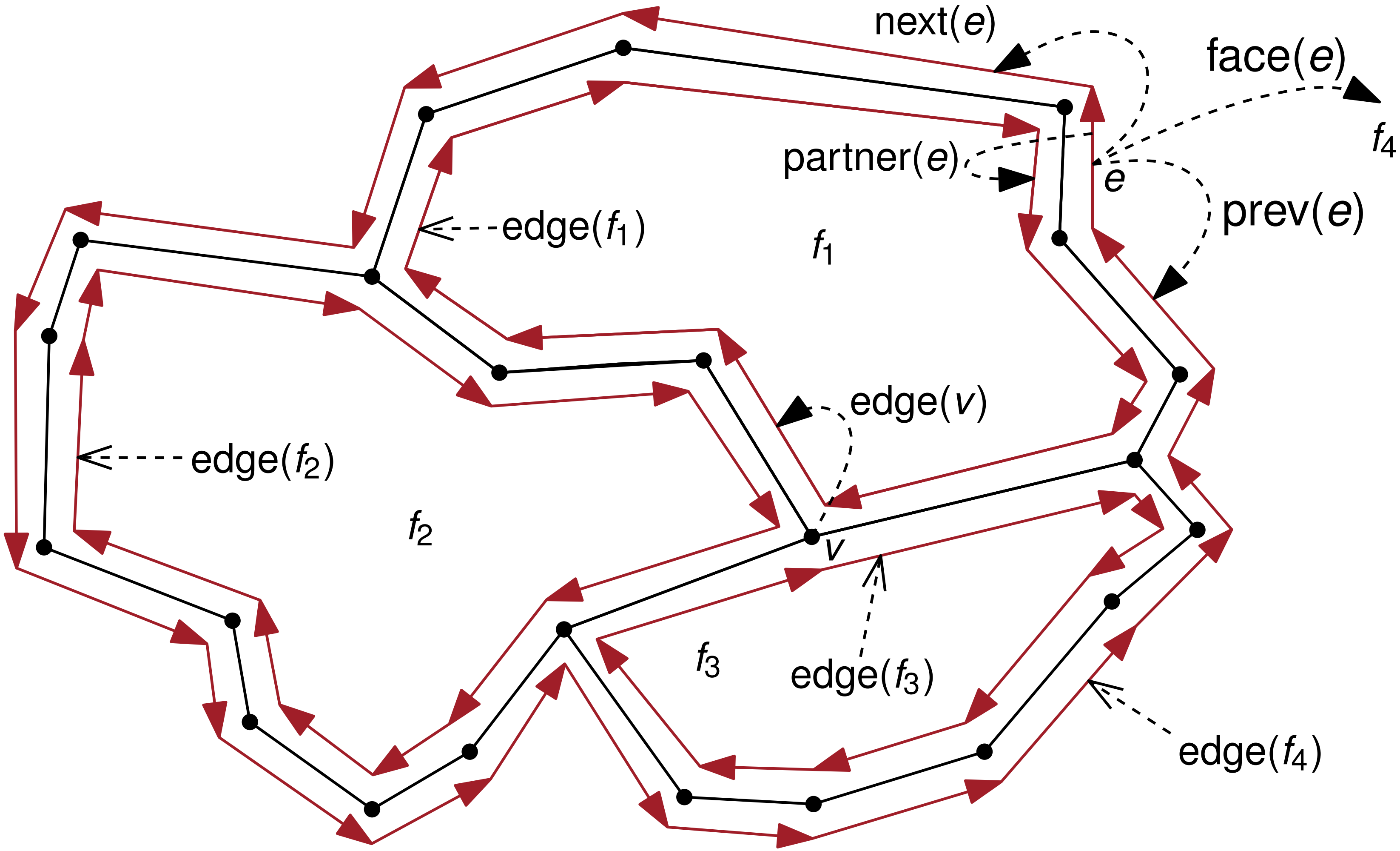
Unterteilung



- Zugriff auf Knoten, Facetten und Kanten
- *Ablaufen* einzelner Facetten.
- Ausgehende Kanten eines Knoten traversieren.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

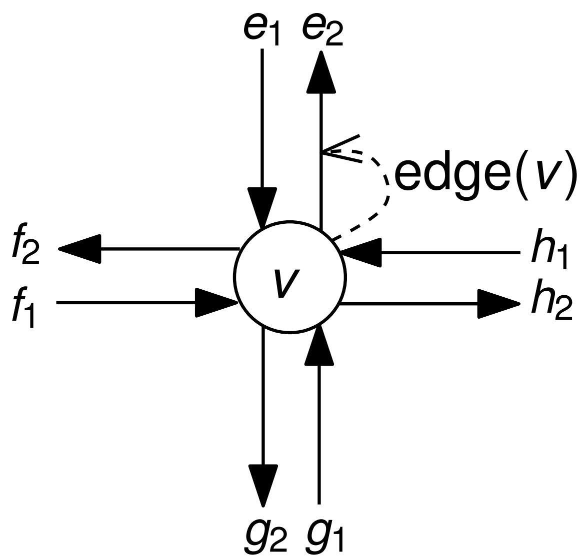
Unterteilung



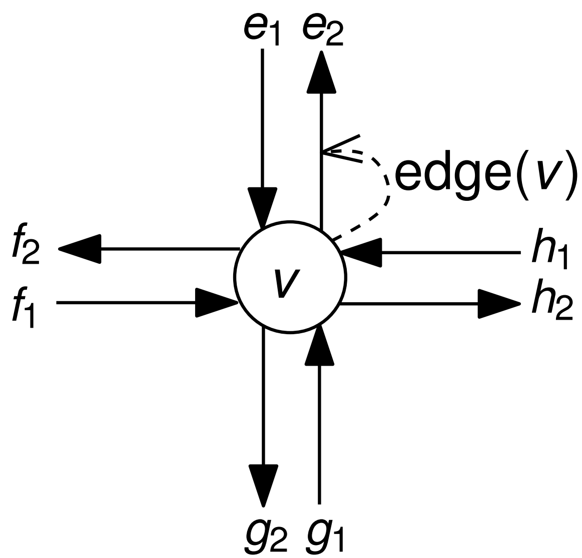
- Zugriff auf Knoten, Facetten und Kanten ✓
- *Ablaufen* einzelner Facetten. ✓
- Ausgehende Kanten eines Knoten traversieren. ?

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Traversierung von inzidenten Kanten



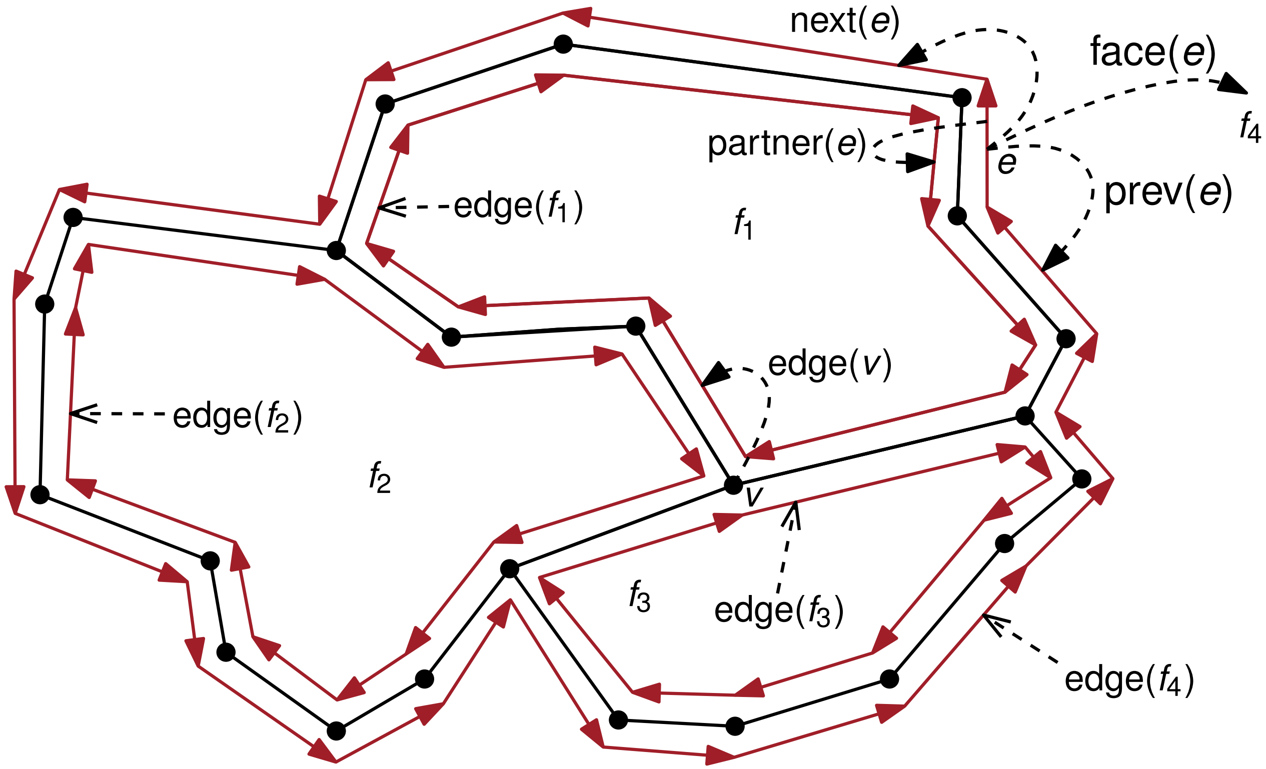
Traversierung von inzidenten Kanten



Traversierung im Uhrzeigersinn:

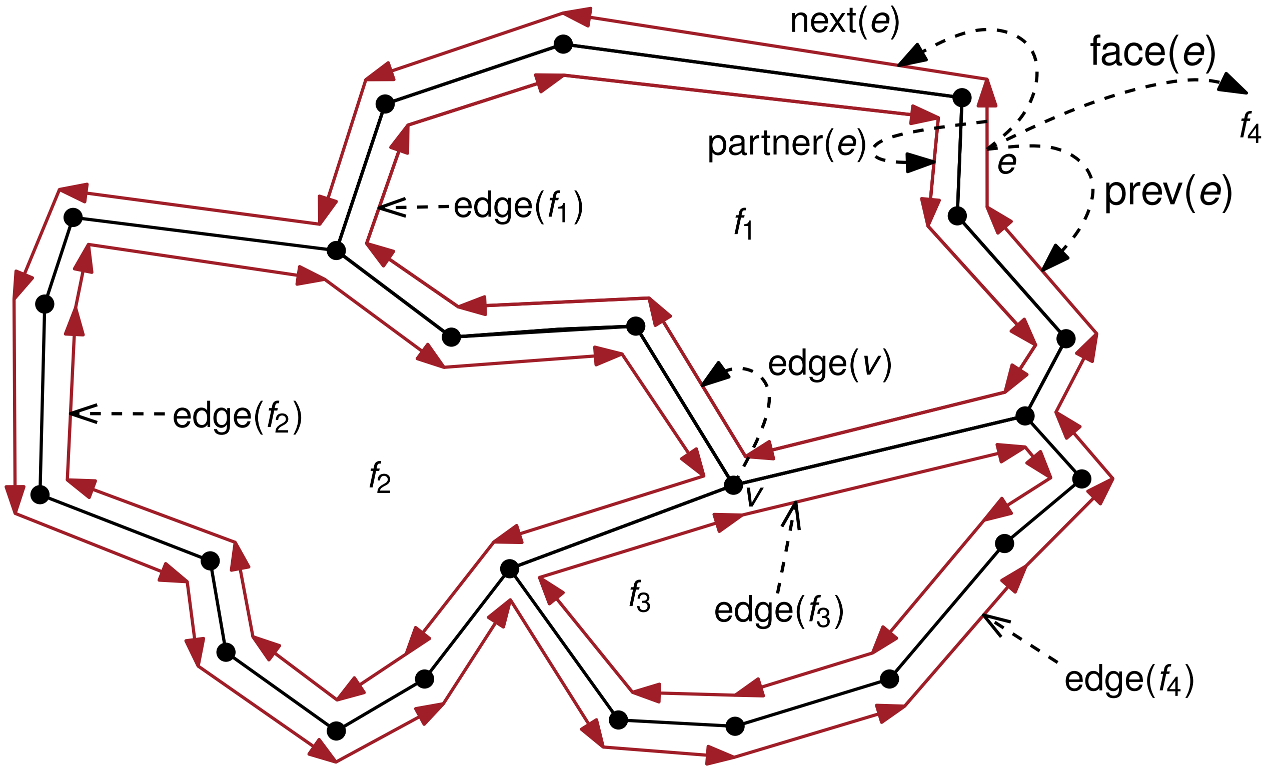
$$f_2 = \text{next}(\text{partner}(e_2))$$
$$g_2 = \text{next}(\text{partner}(f_2))$$
$$h_2 = \text{next}(\text{partner}(g_2))$$
$$e_2 = \text{next}(\text{partner}(h_2))$$

Welche Äquivalenzen sind immer wahr?



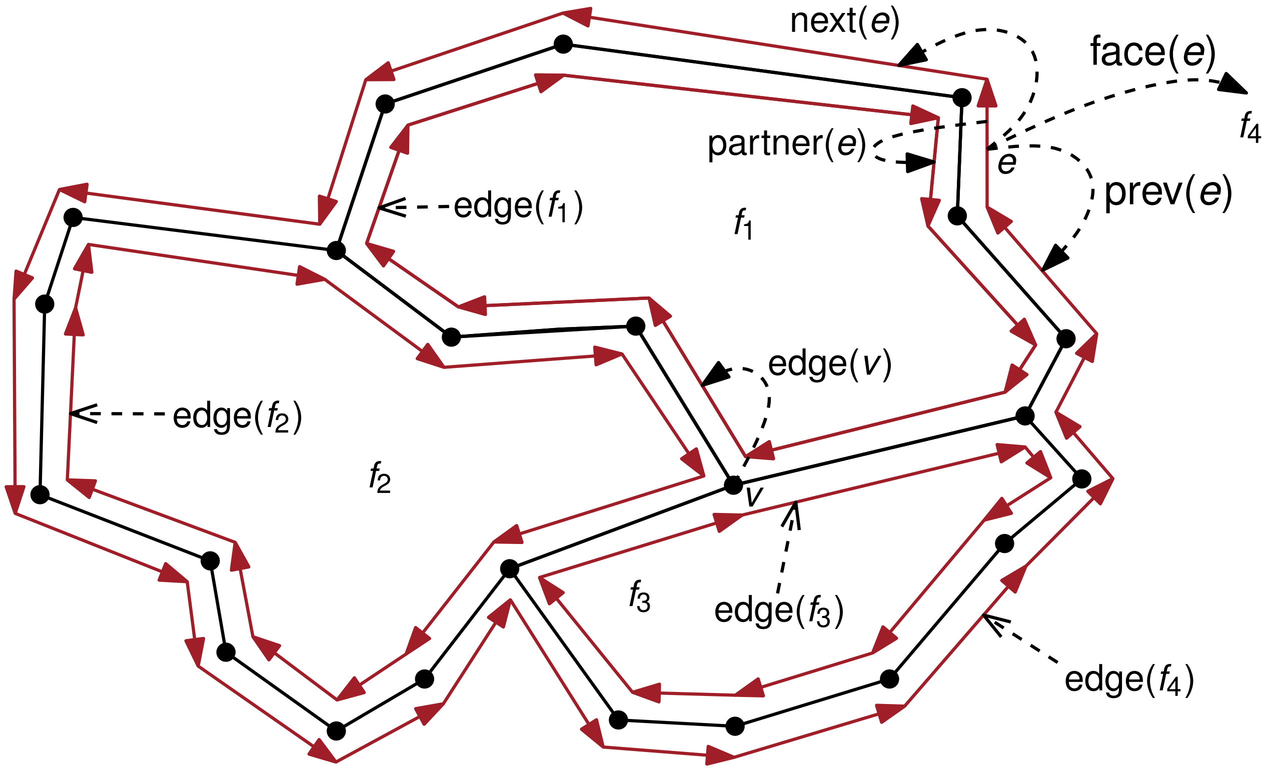
- partner(partner(e)) = e
- next(prev(e)) = e
- partner(prev(partner(e))) = next(e)
- face(e) = face(next(e))

Welche Äquivalenzen sind immer wahr?



- partner(partner(e)) = e ✓
- next(prev(e)) = e
- partner(prev(partner(e))) = next(e)
- face(e) = face(next(e))

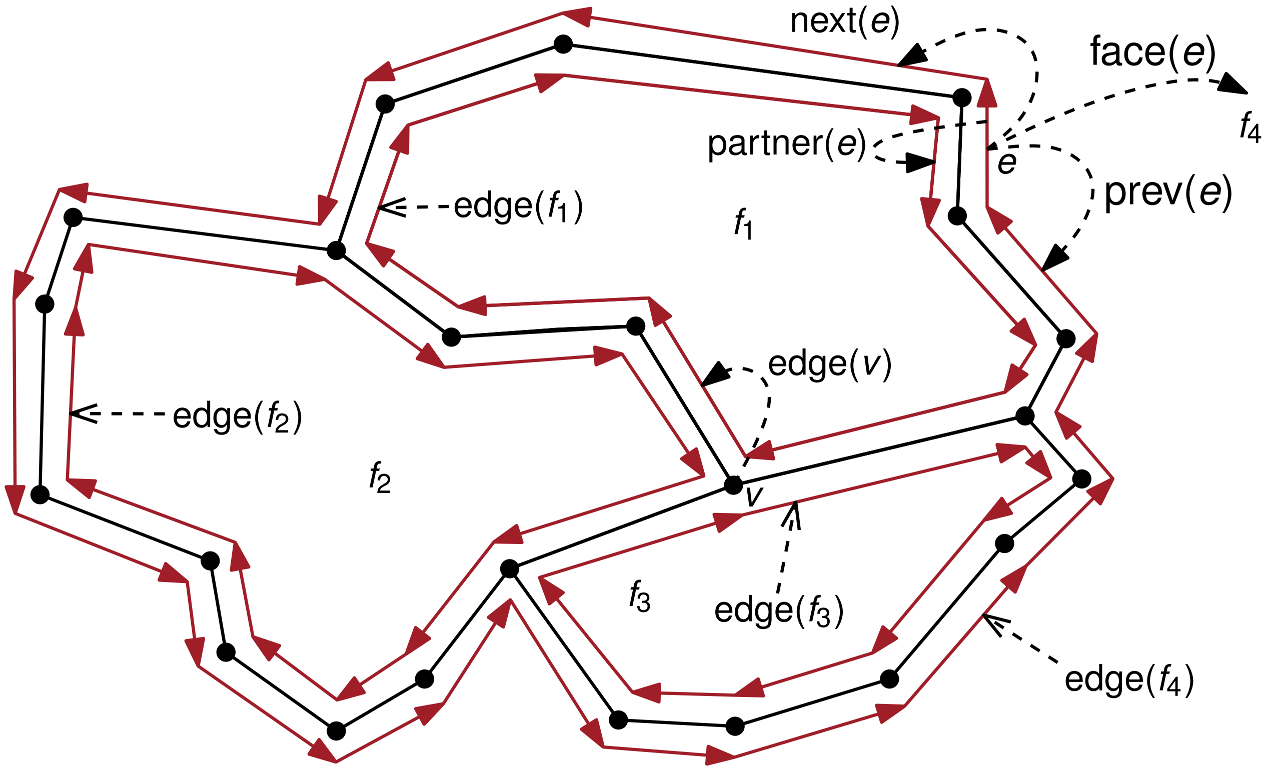
Welche Äquivalenzen sind immer wahr?



- partner(partner(e)) = e ✓
- next(prev(e)) = e ✓
- partner(prev(partner(e))) = next(e)
- face(e) = face(next(e))

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

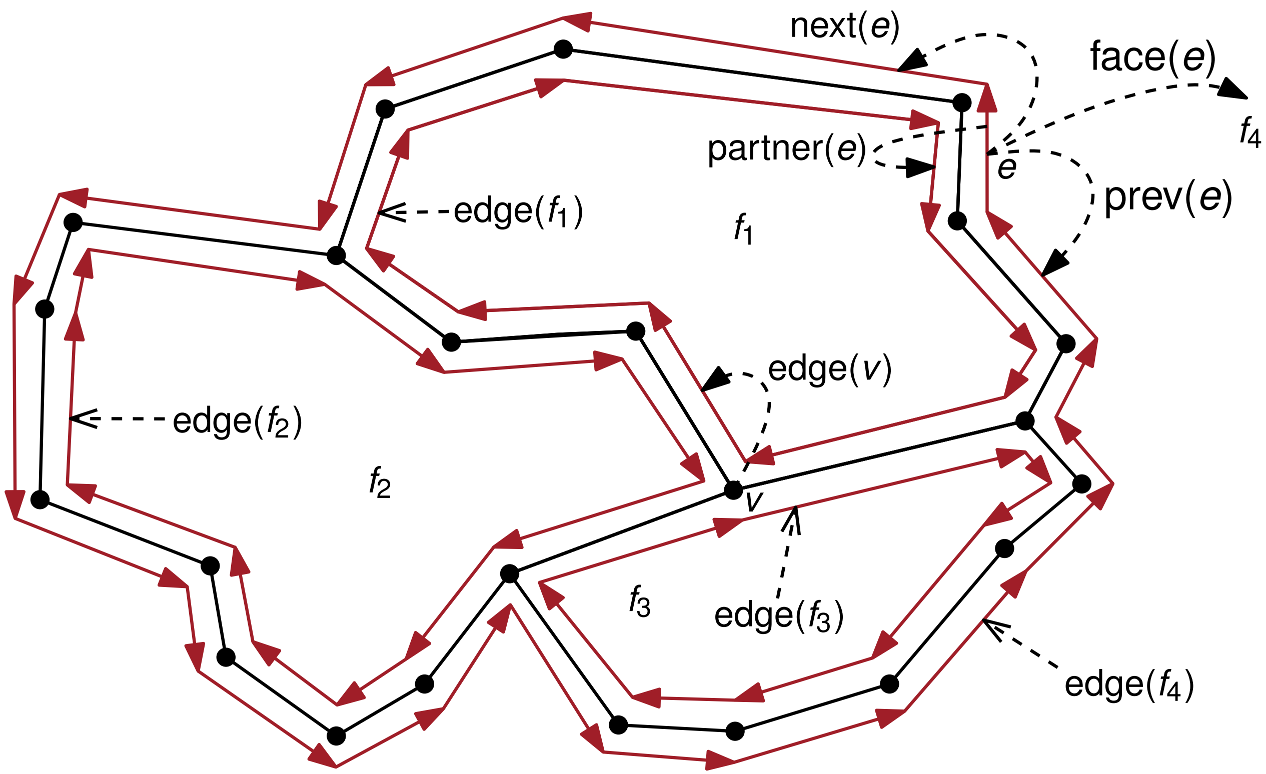
Welche Äquivalenzen sind immer wahr?



- partner(partner(e)) = e ✓
- next(prev(e)) = e ✓
- partner(prev(partner(e))) = next(e) ✗
- face(e) = face(next(e))

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Welche Äquivalenzen sind immer wahr?



- partner(partner(e)) = e ✓
- next(prev(e)) = e ✓
- partner(prev(partner(e))) = next(e) ✗
- face(e) = face(next(e)) ✓

Komplexität

n: Anzahl Knoten
e: Anzahl Kanten
f: Anzahl Facetten

Welchen Platzbedarf besitzt eine zusammenhängende Unterteilung?

Komplexität

n: Anzahl Knoten
e: Anzahl Kanten
f: Anzahl Facetten

Welchen Platzbedarf besitzt eine zusammenhängende Unterteilung?

- Jeder Knoten besitzt einen Zeiger.
- Jede Kante wird in zwei Halbkanten aufgeteilt, die jeweils vier Zeiger besitzen.
- Jede Facette besitzt einen Zeiger.

Damit ergibt sich $O(n + e + f)$ Speicher.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Komplexität

n: Anzahl Knoten
e: Anzahl Kanten
f: Anzahl Facetten

Welchen Platzbedarf besitzt eine zusammenhängende Unterteilung?

- Jeder Knoten besitzt einen Zeiger.
- Jede Kante wird in zwei Halbkanten aufgeteilt, die jeweils vier Zeiger besitzen.
- Jede Facette besitzt einen Zeiger.

Damit ergibt sich $O(n + e + f)$ Speicher.

Satz von Euler besagt: $n - m + f = 2$

Komplexität

n: Anzahl Knoten
e: Anzahl Kanten
f: Anzahl Facetten

Welchen Platzbedarf besitzt eine zusammenhängende Unterteilung?

- Jeder Knoten besitzt einen Zeiger.
- Jede Kante wird in zwei Halbkanten aufgeteilt, die jeweils vier Zeiger besitzen.
- Jede Facette besitzt einen Zeiger.

Damit ergibt sich $O(n + e + f)$ Speicher.

Satz von Euler besagt: $n - m + f = 2$

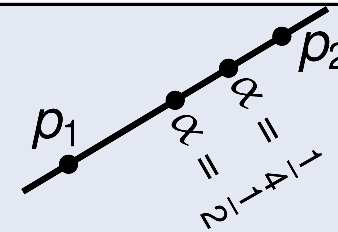
Daraus ergibt sich: Jeder planare Graph besitzt maximal $3n - 6$ Kanten und $2n - 4$ Facetten.

$O(n)$ Speicher für Unterteilung der Ebene

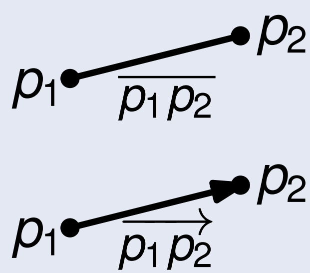
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Strecken

Definition: Konvexkombination
Für zwei Punkte p_1 und p_2 (in der Ebene) und $\alpha \in [0, 1]$ ist der Punkt $p = \alpha p_1 + (1 - \alpha) p_2$ eine *Konvexkombination* von p_1 und p_2.

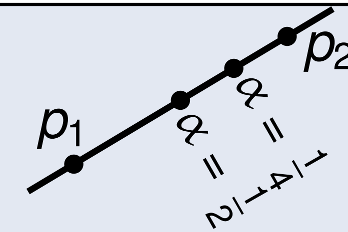


Definition: Strecke
Die Menge aller Konvexkombinationen zweier Punkte p_1, p_2 ist die *Strecke* $\overline{p_1 p_2}$ zwischen p_1 und p_2. Ist die Reihenfolge von p_1 und p_2 von Bedeutung, so sprechen wir von der *gerichteten Strecke* $\overrightarrow{p_1 p_2}$.

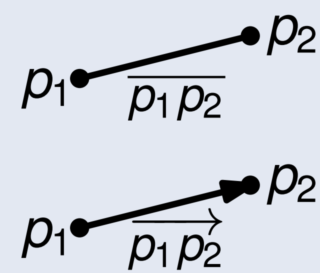


Strecken

Definition: Konvexkombination
Für zwei Punkte p_1 und p_2 (in der Ebene) und $\alpha \in [0, 1]$ ist der Punkt $p = \alpha p_1 + (1 - \alpha)p_2$ eine *Konvexkombination* von p_1 und p_2.

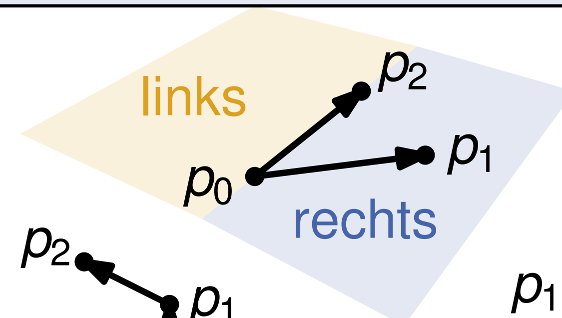


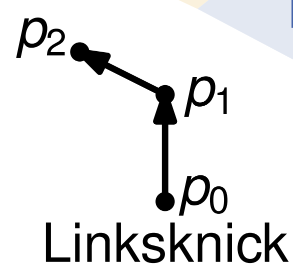
Definition: Strecke
Die Menge aller Konvexkombinationen zweier Punkte p_1, p_2 ist die *Strecke* $\overline{p_1 p_2}$ zwischen p_1 und p_2. Ist die Reihenfolge von p_1 und p_2 von Bedeutung, so sprechen wir von der *gerichteten Strecke* $\overrightarrow{p_1 p_2}$.

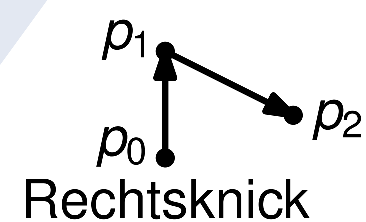


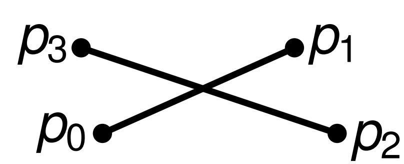
Grundlegende Fragestellungen:

1. Gegeben drei Punkte p_0, p_1, p_2 in der Ebene. Liegt $\overrightarrow{p_0 p_1}$ rechts oder links von $\overrightarrow{p_0, p_2}$?
2. Bilden die beiden Strecken $\overrightarrow{p_0, p_1}$ und $\overrightarrow{p_1, p_2}$ bei p_1 einen Rechts- oder einen Linksknick?
3. Schneiden sich zwei gegebene Strecken $\overline{p_0 p_1}$ und $\overline{p_2 p_3}$?

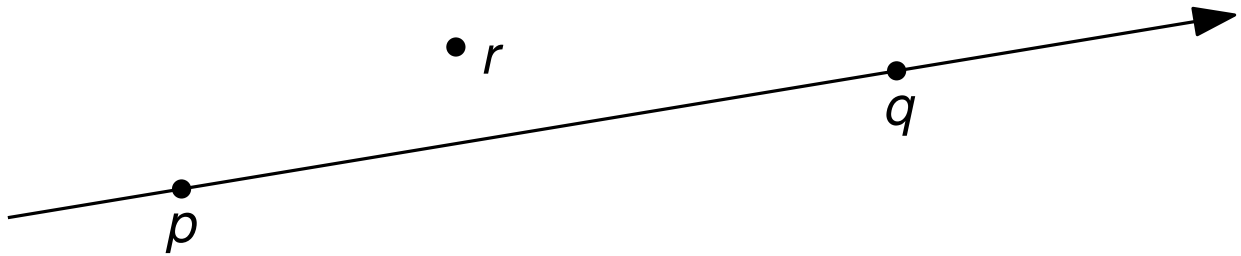








Position eines Punktes

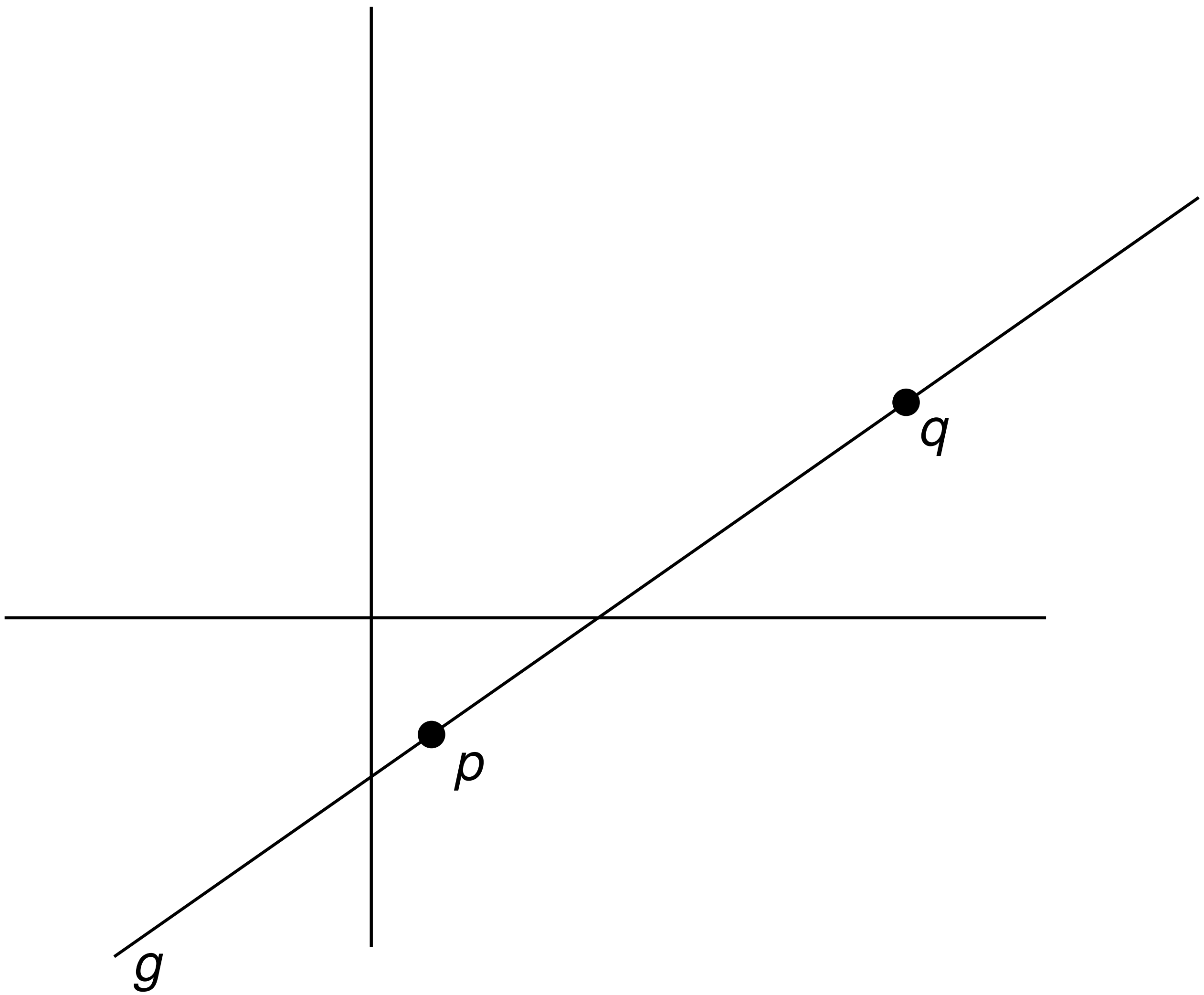


Gerichtete Gerade g durch $p = (p_x, p_y)$ und $q = (q_x, q_y)$

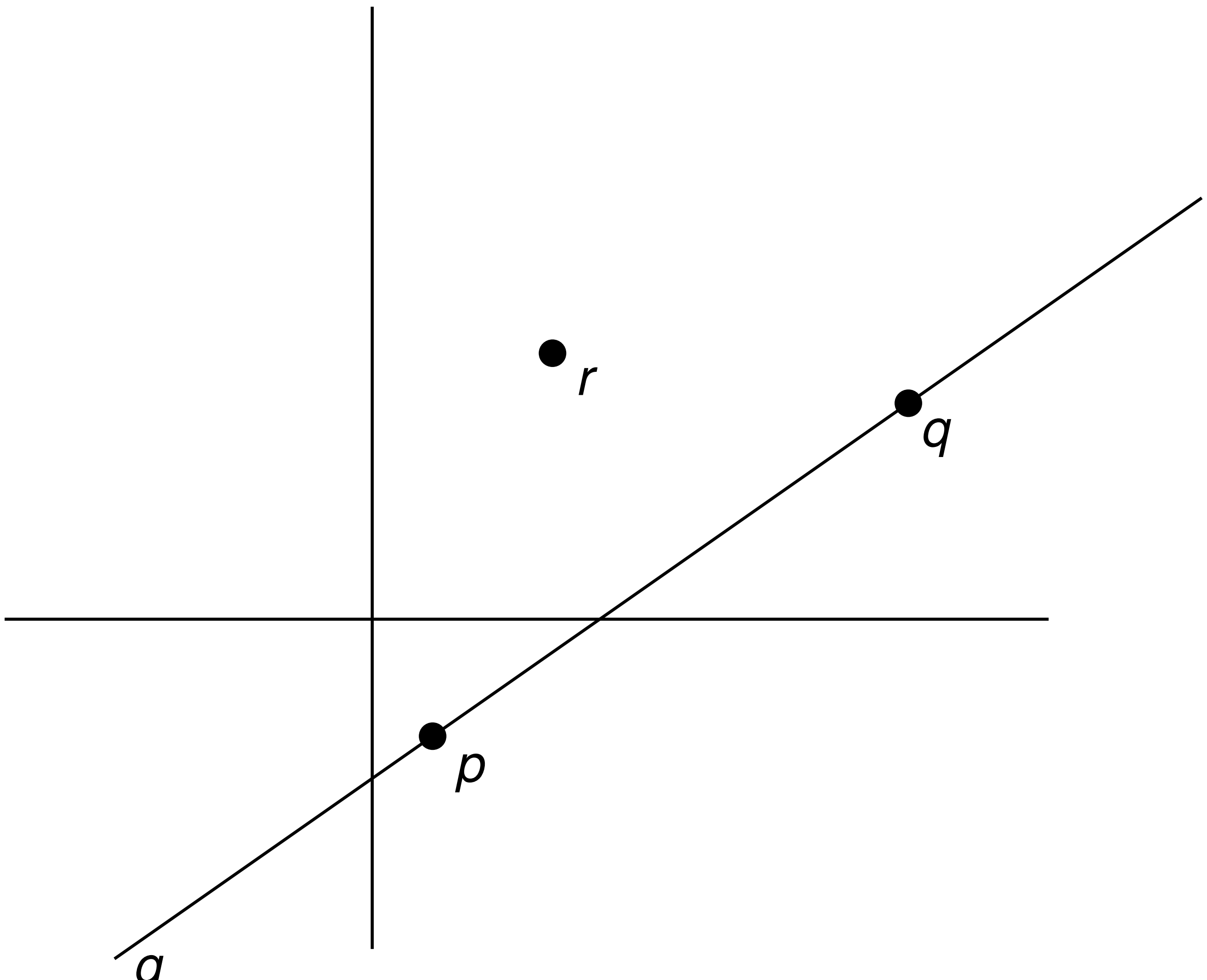
Punkt $r = (r_x, r_y)$

Liegt der Punkt r links oder rechts der Geraden g?

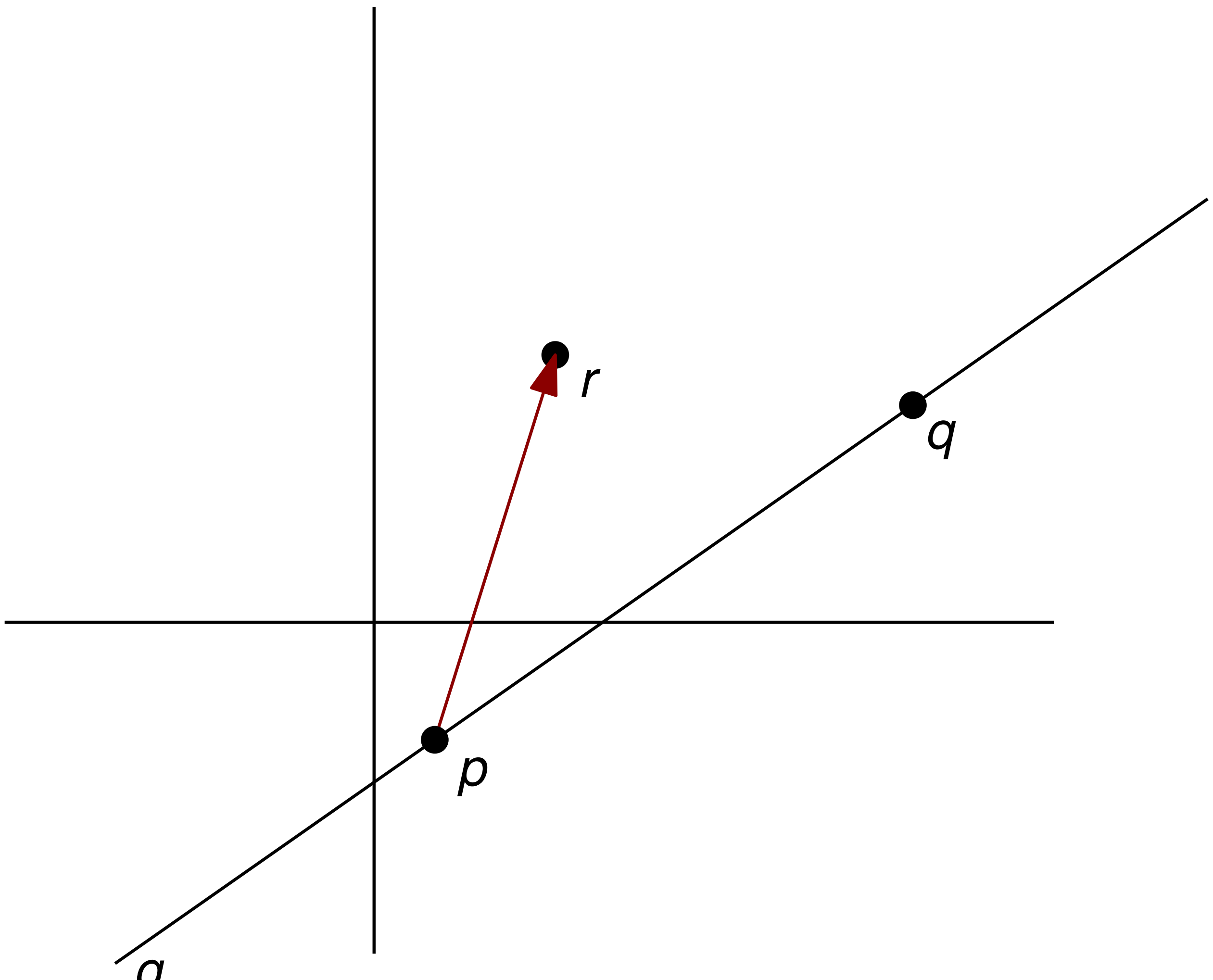
Position eines Punktes



Position eines Punktes

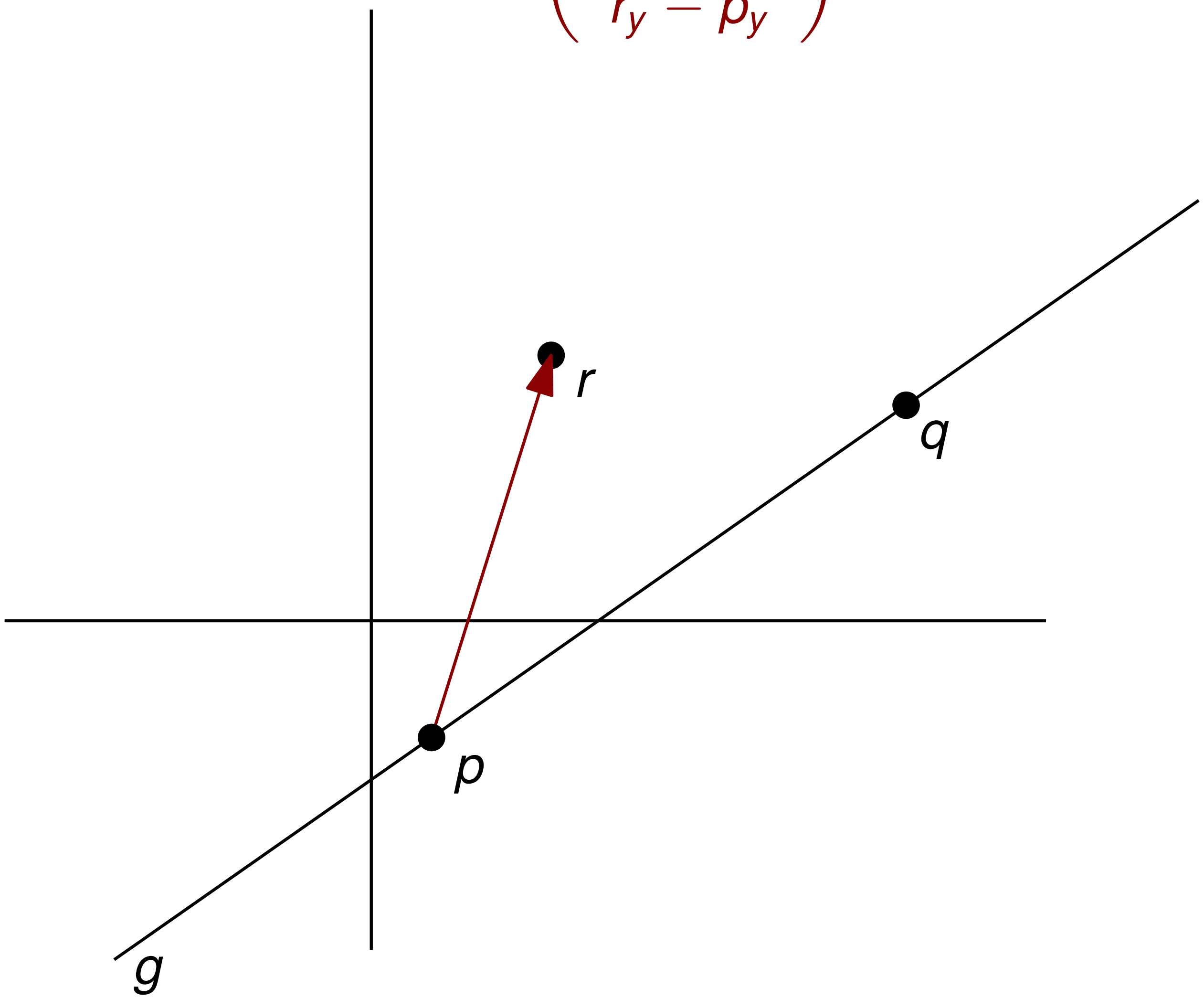


Position eines Punktes



Position eines Punktes

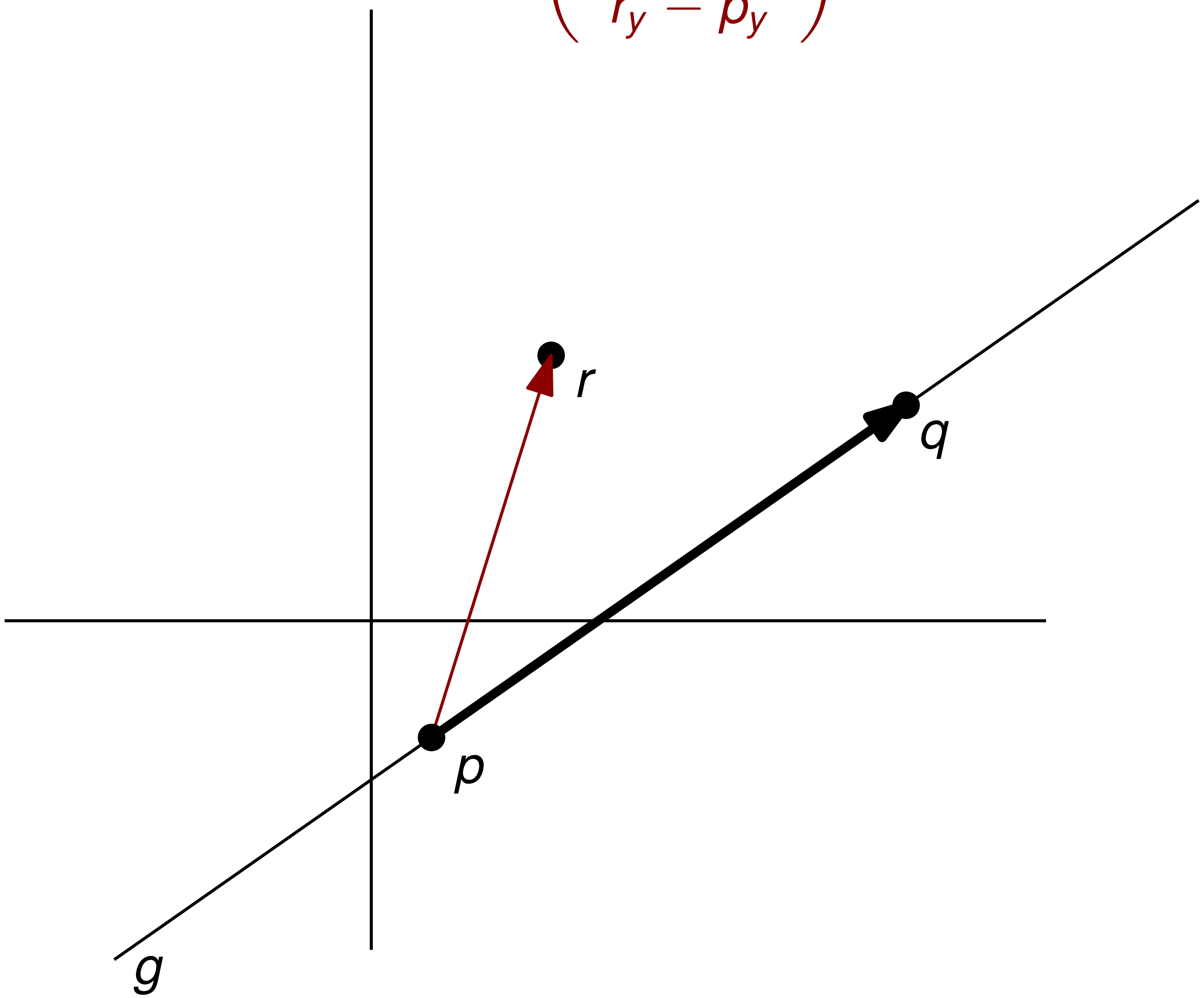
$$v = \begin{pmatrix} r_x - p_x \\ r_y - p_y \end{pmatrix}$$



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Position eines Punktes

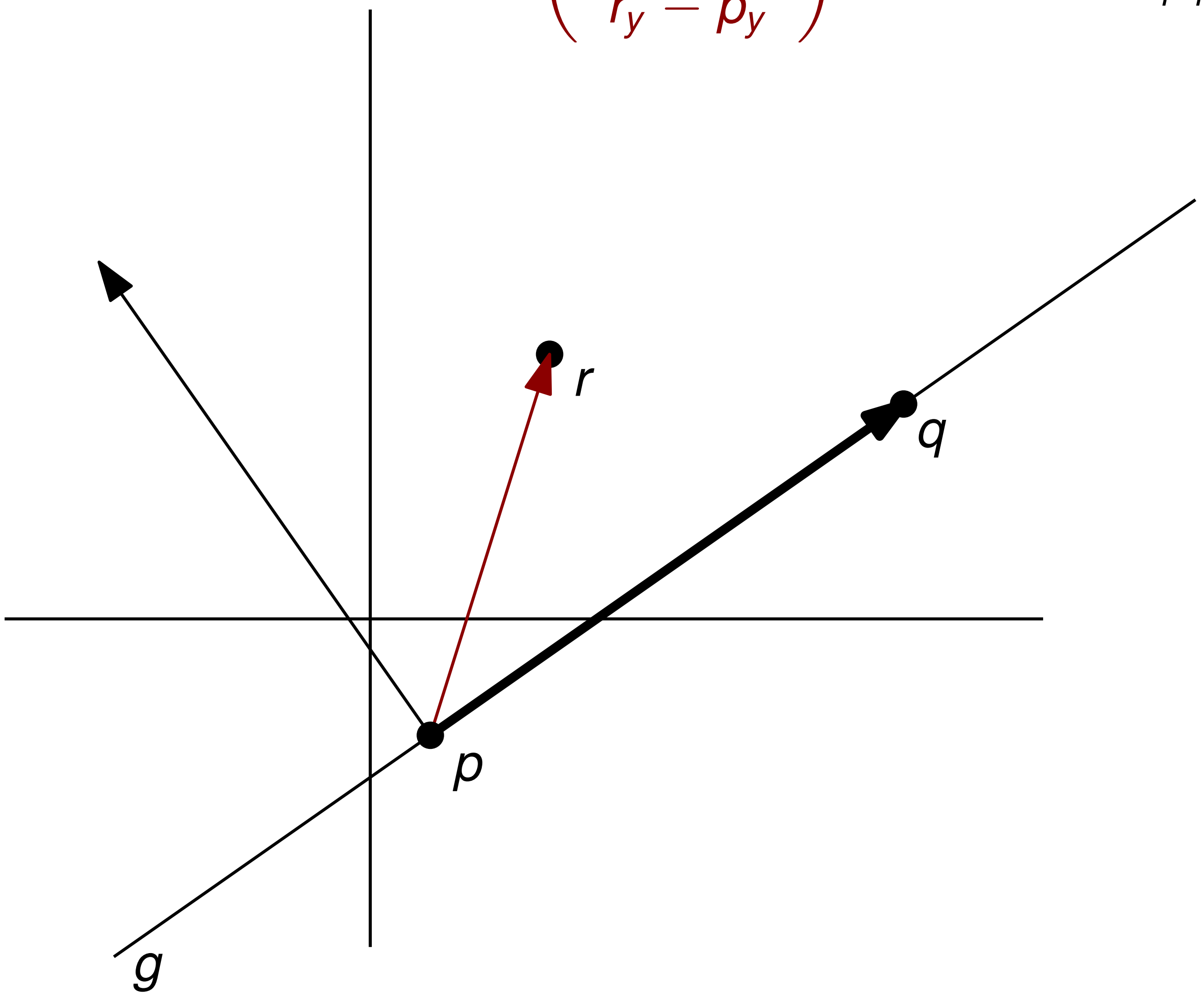
$$v = \begin{pmatrix} r_x - p_x \\ r_y - p_y \end{pmatrix}$$



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

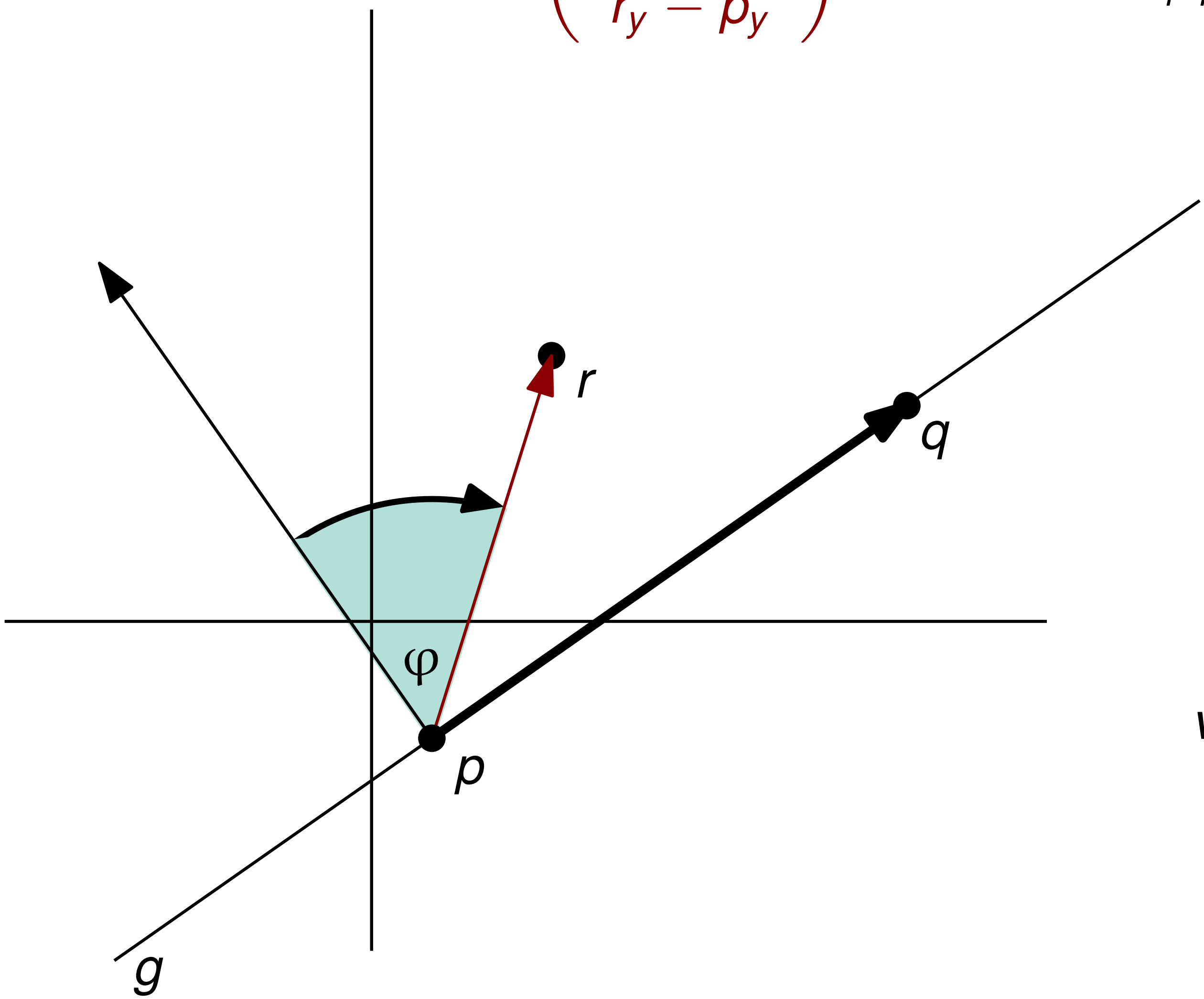
Position eines Punktes

$$v = \begin{pmatrix} r_x - p_x \\ r_y - p_y \end{pmatrix} \qquad n_{pq} = \begin{pmatrix} p_y - q_y \\ q_x - p_x \end{pmatrix}$$



Position eines Punktes

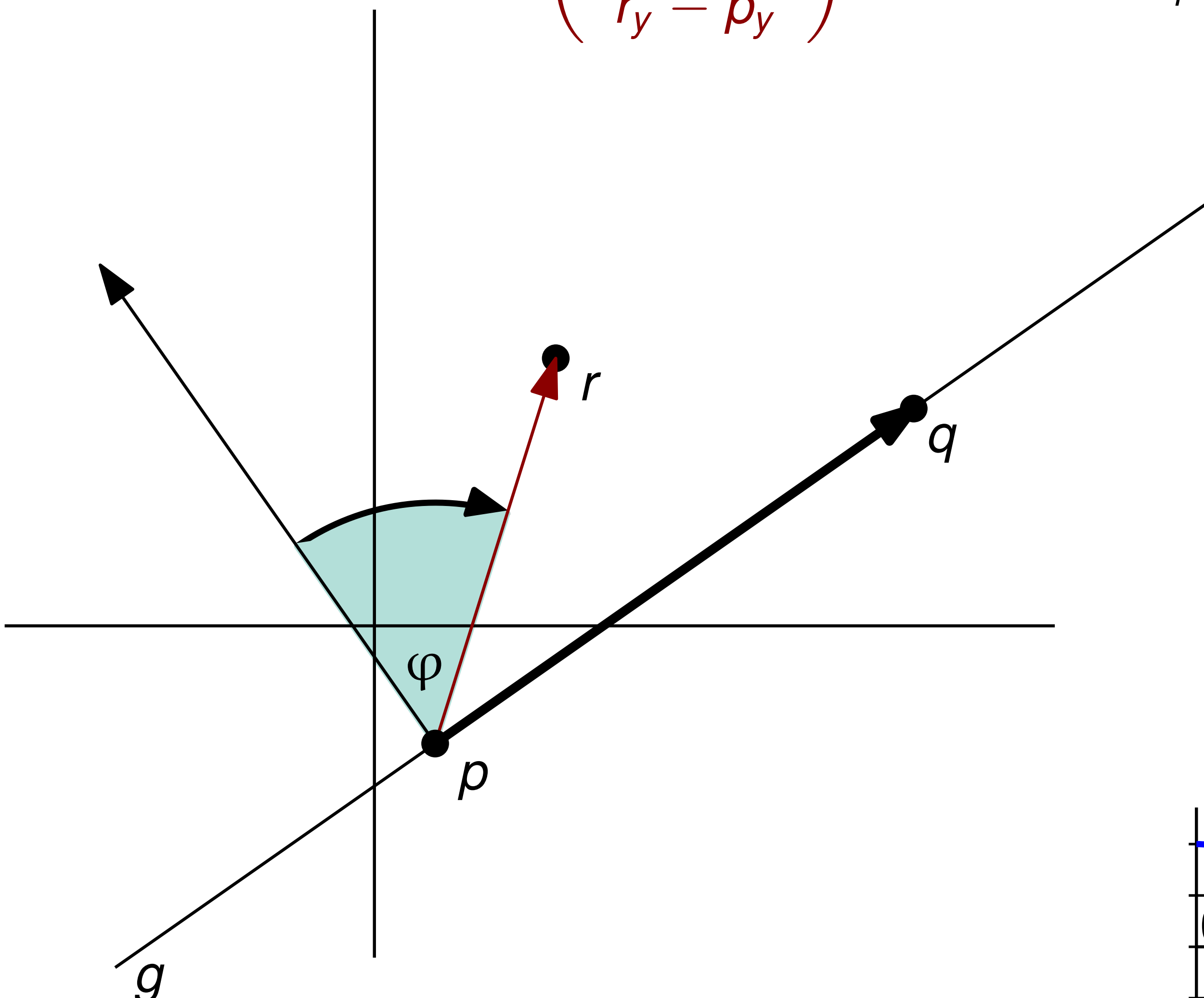
$$v = \begin{pmatrix} r_x - p_x \\ r_y - p_y \end{pmatrix} \qquad n_{pq} = \begin{pmatrix} p_y - q_y \\ q_x - p_x \end{pmatrix}$$



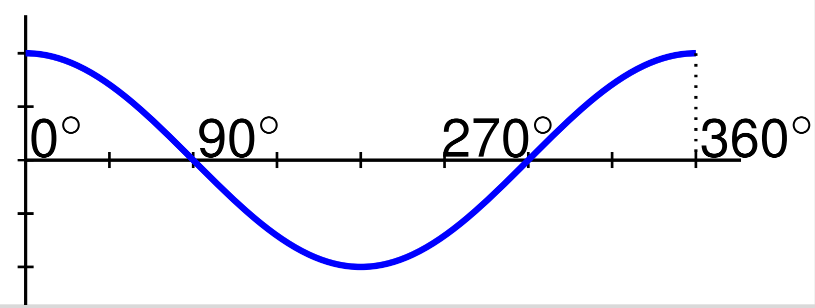
$$v^T \cdot n_{pq} = |v| \cdot |n_{pq}| \cos \varphi$$

Position eines Punktes

$$v = \begin{pmatrix} r_x - p_x \\ r_y - p_y \end{pmatrix} \qquad n_{pq} = \begin{pmatrix} p_y - q_y \\ q_x - p_x \end{pmatrix}$$

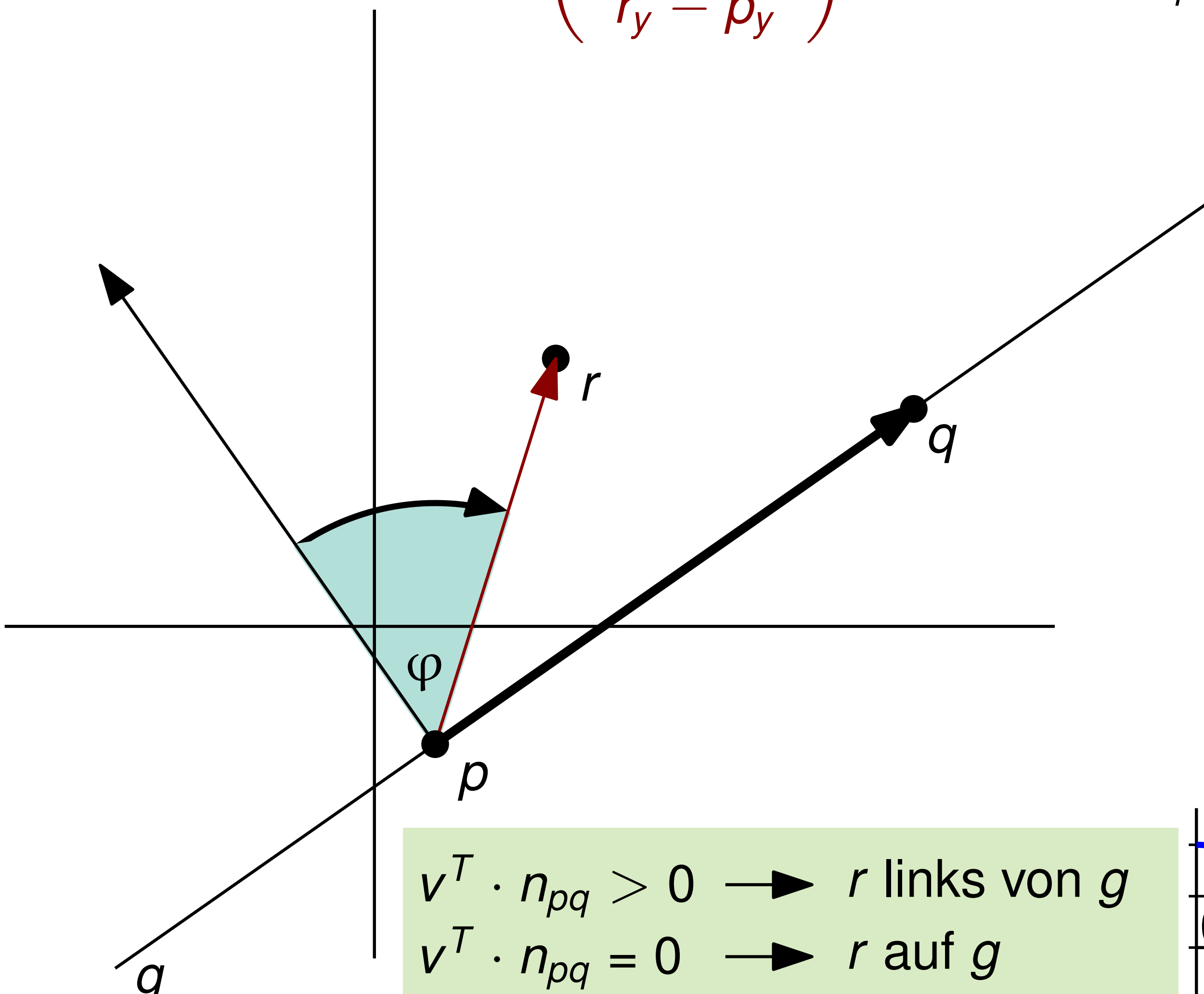


$$v^T \cdot n_{pq} = |v| \cdot |n_{pq}| \cos \varphi$$

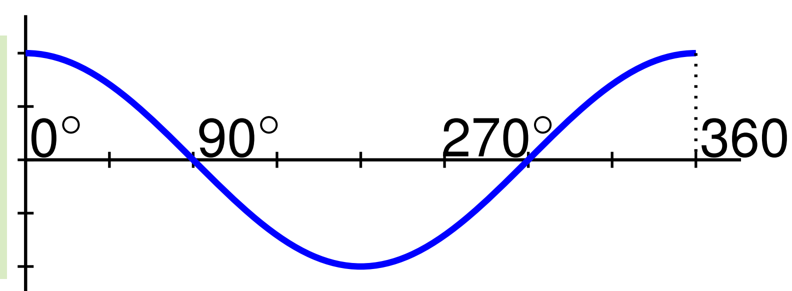


Position eines Punktes

$$v = \begin{pmatrix} r_x - p_x \\ r_y - p_y \end{pmatrix} \qquad n_{pq} = \begin{pmatrix} p_y - q_y \\ q_x - p_x \end{pmatrix}$$



$$v^T \cdot n_{pq} = |v| \cdot |n_{pq}| \cos \varphi$$



$v^T \cdot n_{pq} > 0 \longrightarrow$ r links von g
$v^T \cdot n_{pq} = 0 \longrightarrow$ r auf g
$v^T \cdot n_{pq} < 0 \longrightarrow$ r rechts von g

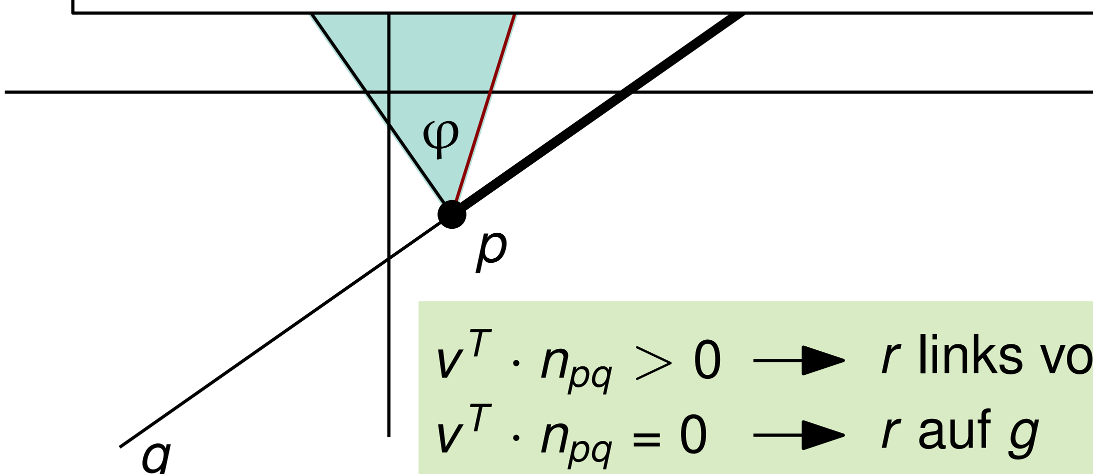
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Position eines Punktes

$$v = \begin{pmatrix} r_x - p_x \\ r_y - p_y \end{pmatrix} \qquad n_{pq} = \begin{pmatrix} p_y - q_y \\ q_x - p_x \end{pmatrix}$$

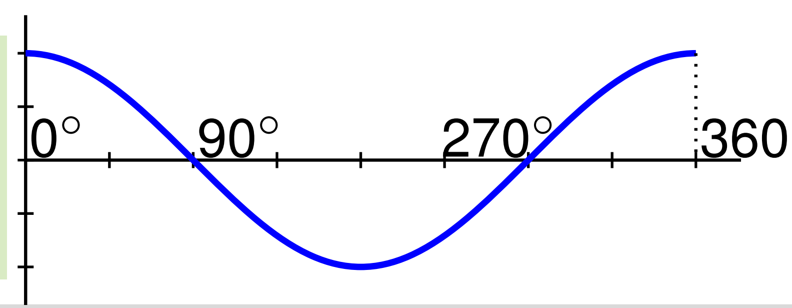
Vorteile dieser Methode:

- Es werden nur Additionen, Subtraktionen, Multiplikationen und Vergleiche durchgeführt.
- Keine Divisionen, Wurzeln, etc.
- Wenn Eingabe ganzzahlig, dann sind alle Zwischenschritte ganzzahlig.
  $\Rightarrow$ Keine Fehler durch Ungenauigkeiten bei Gleitkommaarithmetik.



$$v^T \cdot n_{pq} = |v| \cdot |n_{pq}| \cos \varphi$$

$v^T \cdot n_{pq} > 0 \longrightarrow r$ links von g
$v^T \cdot n_{pq} = 0 \longrightarrow r$ auf g
$v^T \cdot n_{pq} < 0 \longrightarrow r$ rechts von g



Schneiden sich zwei Strecken?

Definition: Umschließendes Rechteck (bounding box)
Das *Umschließende Rechteck (bounding box)* eines geometrischen Objekts ist das kleinste achsenparallele Rechteck, das diese Objekt enthält.

Notwendige Bedingung: Zwei Strecken können sich nur schneiden, wenn sich auch ihre umschließenden Rechtecke schneiden.

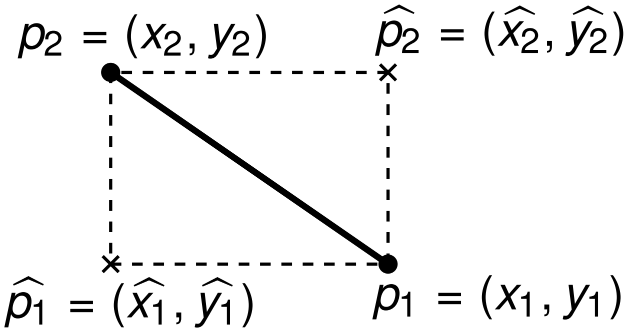
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Schneiden sich zwei Strecken?

Definition: Umschließendes Rechteck (bounding box)
Das *Umschließende Rechteck (bounding box)* eines geometrischen Objekts ist das kleinste achsenparallele Rechteck, das diese Objekt enthält.

Notwendige Bedingung: Zwei Strecken können sich nur schneiden, wenn sich auch ihre umschließenden Rechtecke schneiden.

Das umschließende Rechteck $(\widehat{p_1}, \widehat{p_2})$ zu $\overline{p_1 p_2}$ ist beschrieben durch den linken unteren Punkt $\widehat{p_1} = (\widehat{x_1}, \widehat{y_1})$ und den rechten oberen Punkte $\widehat{p_2} = (\widehat{x_2}, \widehat{y_2})$ mit:



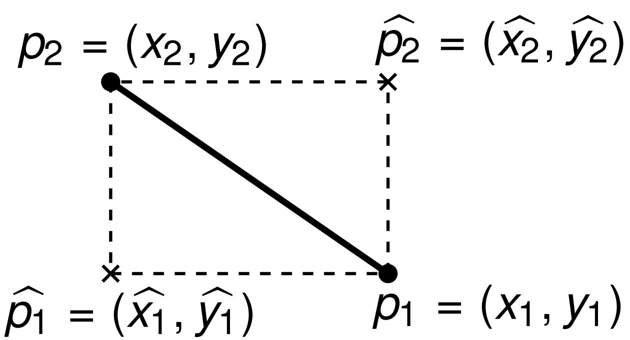
$$\widehat{x_1} = \min(x_1, x_2) \qquad \widehat{y_1} = \min(y_1, y_2)$$
$$\widehat{x_2} = \max(x_1, x_2) \qquad \widehat{y_2} = \max(y_1, y_2)$$

Schneiden sich zwei Strecken?

Definition: Umschließendes Rechteck (bounding box)
Das *Umschließende Rechteck (bounding box)* eines geometrischen Objekts ist das kleinste achsenparallele Rechteck, das diese Objekt enthält.

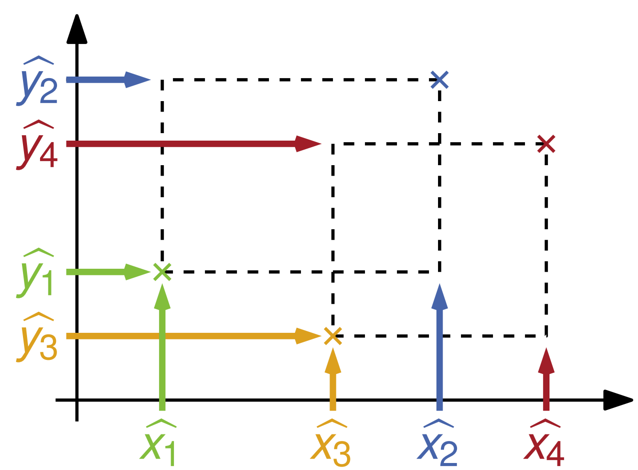
Notwendige Bedingung: Zwei Strecken können sich nur schneiden, wenn sich auch ihre umschließenden Rechtecke schneiden.

Das umschließende Rechteck $(\widehat{p_1}, \widehat{p_2})$ zu $\overline{p_1 p_2}$ ist beschrieben durch den linken unteren Punkt $\widehat{p_1} = (\widehat{x_1}, \widehat{y_1})$ und den rechten oberen Punkte $\widehat{p_2} = (\widehat{x_2}, \widehat{y_2})$ mit:



$$\widehat{x_1} = \min(x_1, x_2) \qquad \widehat{y_1} = \min(y_1, y_2)$$
$$\widehat{x_2} = \max(x_1, x_2) \qquad \widehat{y_2} = \max(y_1, y_2)$$

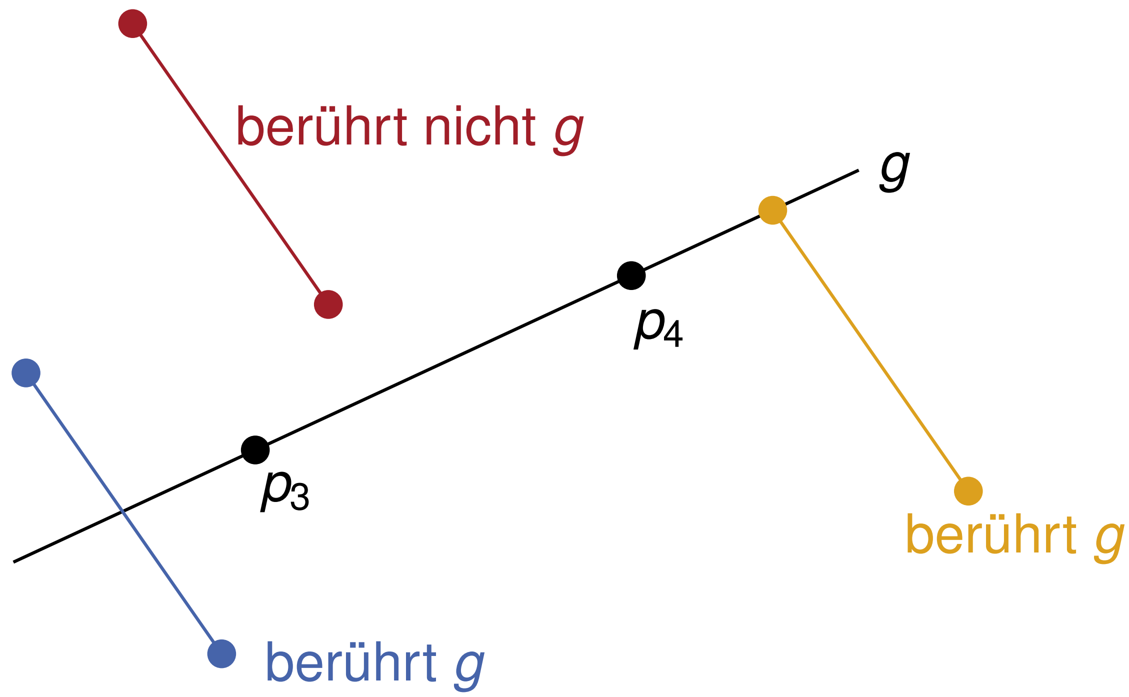
Zwei Rechtecke $(\widehat{p_1}, \widehat{p_2})$ und $(\widehat{p_3}, \widehat{p_4})$ schneiden sich genau dann, wenn
$\widehat{x_2} \geq \widehat{x_3} \wedge \widehat{x_4} \geq \widehat{x_1} \wedge \widehat{y_2} \geq \widehat{y_3} \wedge \widehat{y_4} \geq \widehat{y_1}$.



Schneiden sich zwei Strecken?

Definition
$\overline{p_1 p_2}$ *berührt* die Gerade durch $\overline{p_3 p_4}$ falls p_1 und p_2 auf unterschiedlichen Seiten von $\overline{p_3 p_4}$ liegen oder einer von ihnen auf der Gerade liegt.

Ob eine Strecke die Gerade durch zwei Punkte berührt kann mithilfe von Positionstests überprüft werden.



Schneiden sich zwei Strecken?

Karlsruher Institut für Technologie

Definition
$\overline{p_1 p_2}$ *berührt* die Gerade durch $\overline{p_3 p_4}$ falls p_1 und p_2 auf unterschiedlichen Seiten von $\overline{p_3 p_4}$ liegen oder einer von ihnen auf der Gerade liegt.

Ob eine Strecke die Gerade durch zwei Punkte berührt kann mithilfe von Positionstests überprüft werden.

Lemma
Zwei Strecken schneiden sich genau dann, wenn ihre umschließenden Rechtecke sich schneiden und jede der beiden Strecken die Gerade durch die jeweils andere Strecke berührt.

Vorteile dieser Methode:
Wie zuvor müssen Additionen, Subtraktionen, Multiplikationen und Vergleiche durchgeführt werden (insbesondere keine Divisionen, Wurzeln, etc.)
$\Rightarrow$ Bei Ganzzahligen Koordinaten ist das Ergebnis und alle Zwischenschritte Ganzzahlig.
$\Rightarrow$ Keine Fehler durch Ungenauigkeiten bei Gleitkommaarithmetik.