

Offline Bewegungsplanung: Red-Blue Merge

Elmar Langetepe
University of Bonn

Zurück zur Aufgabe: Konfigurationsraum

Zurück zur Aufgabe: Konfigurationsraum

- n Bögen

Zurück zur Aufgabe: Konfigurationsraum

- n Bögen
- Je zwei schneiden sich s mal

Zurück zur Aufgabe: Konfigurationsraum

- n Bögen
- Je zwei schneiden sich s mal
- X -monoton,

Zurück zur Aufgabe: Konfigurationsraum

- n Bögen
- Je zwei schneiden sich s mal
- X -monoton, eventuell erzeugen

Zurück zur Aufgabe: Konfigurationsraum

- n Bögen
- Je zwei schneiden sich s mal
- X -monoton, eventuell erzeugen
- Startpunkt x

Zurück zur Aufgabe: Konfigurationsraum

- n Bögen
- Je zwei schneiden sich s mal
- X -monoton, eventuell erzeugen
- Startpunkt x
- Komplexität der Zelle Z_x :

Zurück zur Aufgabe: Konfigurationsraum

- n Bögen
- Je zwei schneiden sich s mal
- X -monoton, eventuell erzeugen
- Startpunkt x
- Komplexität der Zelle Z_x : $\lambda_{s+2}(4n)$

Zurück zur Aufgabe: Konfigurationsraum

- n Bögen
- Je zwei schneiden sich s mal
- X -monoton, eventuell erzeugen
- Startpunkt x
- Komplexität der Zelle Z_x : $\lambda_{s+2}(4n)$
- Divide and Conquer Ansatz sinnvoll

Alg. 2.4: Zelle Z_x !

Alg. 2.4: Zelle Z_x !

Gegeben: Punkt x , n X -monotone Bögen.

Alg. 2.4: Zelle Z_x !

Gegeben: Punkt x , n X -monotone Bögen.

Gesucht: Zelle Z_x im Arrangement der Bögen, die x enthält.

Alg. 2.4: Zelle Z_x !

Gegeben: Punkt x , n X -monotone Bögen.

Gesucht: Zelle Z_x im Arrangement der Bögen, die x enthält.

- Zerlege Menge der Bögen in gleichgroße Teilmengen R und B .

Alg. 2.4: Zelle Z_x !

Gegeben: Punkt x , n X -monotone Bögen.

Gesucht: Zelle Z_x im Arrangement der Bögen, die x enthält.

- Zerlege Menge der Bögen in gleichgroße Teilmengen R und B .
- Berechne rek. im Arrangement $A(R)$ Zelle $Z(R)_x$, die x enthält

Alg. 2.4: Zelle Z_x !

Gegeben: Punkt x , n X -monotone Bögen.

Gesucht: Zelle Z_x im Arrangement der Bögen, die x enthält.

- Zerlege Menge der Bögen in gleichgroße Teilmengen R und B .
- Berechne rek. im Arrangement $A(R)$ Zelle $Z(R)_x$, die x enthält.
- Berechne rek. im Arrangement $A(B)$ Zelle $Z(B)_x$, die x enthält.

Alg. 2.4: Zelle Z_x !

Gegeben: Punkt x , n X -monotone Bögen.

Gesucht: Zelle Z_x im Arrangement der Bögen, die x enthält.

- Zerlege Menge der Bögen in gleichgroße Teilmengen R und B .
- Berechne rek. im Arrangement $A(R)$ Zelle $Z(R)_x$, die x enthält.
- Berechne rek. im Arrangement $A(B)$ Zelle $Z(B)_x$, die x enthält.
- Berechne Zusammenhangskomponente Z_x von $Z(R)_x \cap Z(B)_x$, die x enthält und berichte diese!

Alg. 2.4: Zelle Z_x !

Gegeben: Punkt x , n X -monotone Bögen.

Gesucht: Zelle Z_x im Arrangement der Bögen, die x enthält.

- Zerlege Menge der Bögen in gleichgroße Teilmengen R und B .
- Berechne rek. im Arrangement $A(R)$ Zelle $Z(R)_x$, die x enthält.
- Berechne rek. im Arrangement $A(B)$ Zelle $Z(B)_x$, die x enthält.
- Berechne Zusammenhangskomponente Z_x von $Z(R)_x \cap Z(B)_x$, die x enthält und berichte diese! **RED-BLUE Merge**

Alg. 2.4: Zelle Z_x !

Gegeben: Punkt x , n X -monotone Bögen.

Gesucht: Zelle Z_x im Arrangement der Bögen, die x enthält.

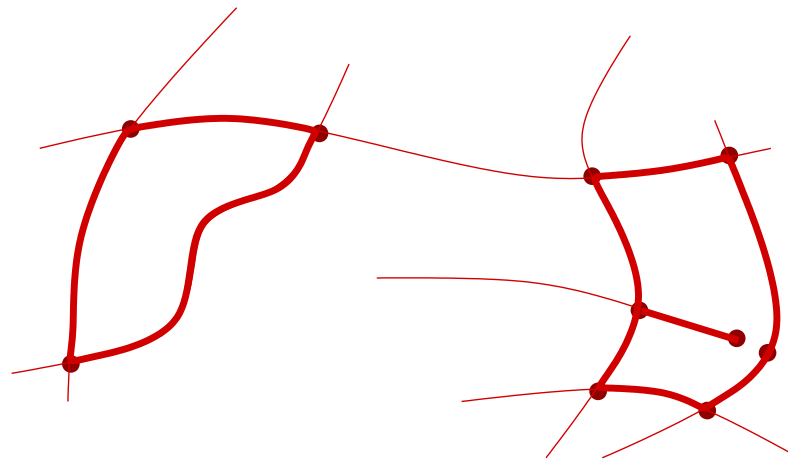
- Zerlege Menge der Bögen in gleichgroße Teilmengen R und B .
- Berechne rek. im Arrangement $A(R)$ Zelle $Z(R)_x$, die x enthält.
- Berechne rek. im Arrangement $A(B)$ Zelle $Z(B)_x$, die x enthält.
- Berechne Zusammenhangskomponente Z_x von $Z(R)_x \cap Z(B)_x$, die x enthält und berichte diese! **RED-BLUE Merge**

Zuerst **RED-BLUE Merge** betrachten, dann zurück!

Allgemeiner RED-BLUE Merge

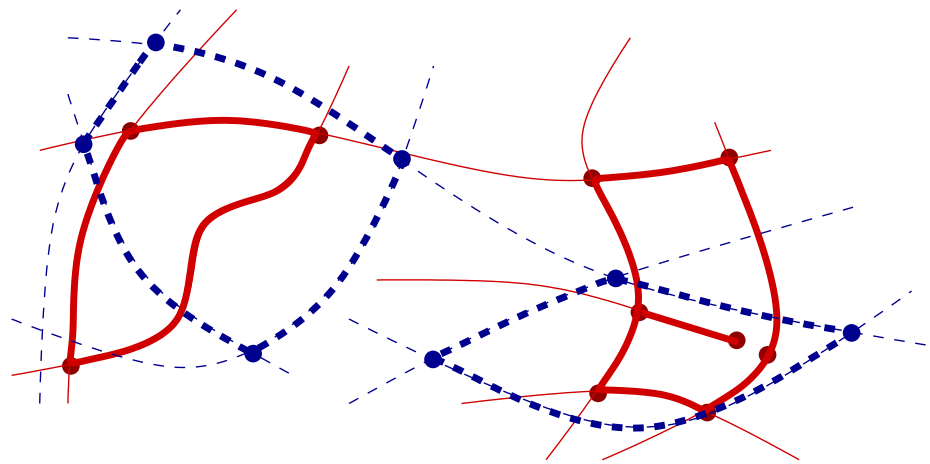
Allgemeiner RED-BLUE Merge

- Rotes Arrangement R mit Zellen R_1, \dots, R_{m_R} , r Ecken.



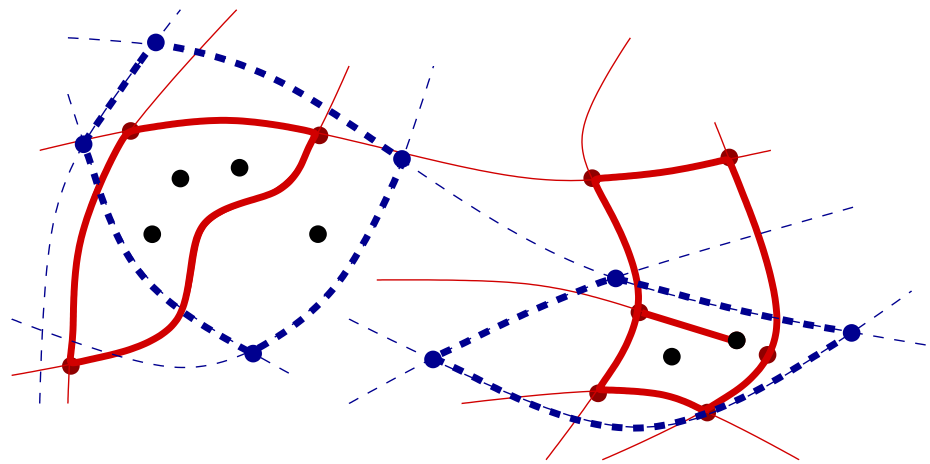
Allgemeiner RED-BLUE Merge

- Rotes Arrangement R mit Zellen R_1, \dots, R_{m_R} , r Ecken.
- Blaues Arrangement B mit Zellen B_1, \dots, B_{m_B} , b Ecken.



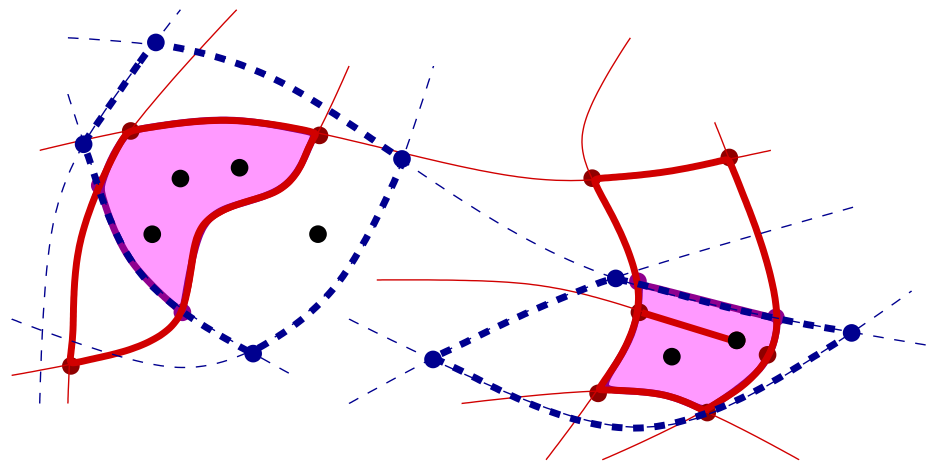
Allgemeiner RED-BLUE Merge

- Rotes Arrangement R mit Zellen R_1, \dots, R_{m_R} , r Ecken.
- Blaues Arrangement B mit Zellen B_1, \dots, B_{m_B} , b Ecken.
- Punktmenge p_i $i = 1, \dots, k$



Allgemeiner RED-BLUE Merge

- Rotes Arrangement R mit Zellen R_1, \dots, R_{m_R} , r Ecken.
- Blaues Arrangement B mit Zellen B_1, \dots, B_{m_B} , b Ecken.
- Punktmenge p_i $i = 1, \dots, k$
- Schnittzellen $Z_j = R_{\mu_j} \cap B_{\nu_j}$, $j = 1, \dots, l$, die mind. ein p_i enthalten



Komplexität der Schnittzellen: **Lem. 2.21**

Komplexität der Schnittzellen: **Lem. 2.21**

Kombinationslemma: Guibas, Sharir, Sifrony 1989 (DSS *Bibel*)

Komplexität der Zellen Z_1, \dots, Z_ℓ , $\ell \leq k$:

$$|Z_1| + |Z_2| + \dots + |Z_\ell| \in O(r + b + k)$$

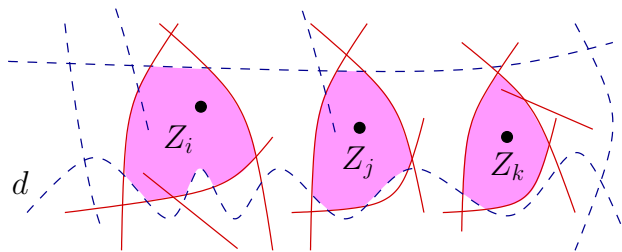
Komplexität der Schnittzellen: **Lem. 2.21**

Kombinationslemma: Guibas, Sharir, Sifrony 1989 (DSS *Bibel*)

Komplexität der Zellen Z_1, \dots, Z_ℓ , $\ell \leq k$:

$$|Z_1| + |Z_2| + \dots + |Z_\ell| \in O(r + b + k)$$

Nicht trivial:



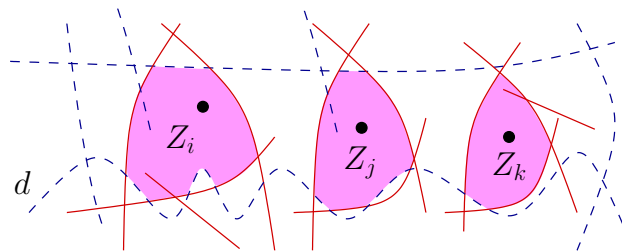
Komplexität der Schnittzellen: **Lem. 2.21**

Kombinationslemma: Guibas, Sharir, Sifrony 1989 (DSS *Bibel*)

Komplexität der Zellen Z_1, \dots, Z_ℓ , $\ell \leq k$:

$$|Z_1| + |Z_2| + \dots + |Z_\ell| \in O(r + b + k)$$

Nicht trivial:



Zur Analyse der Berechnung verwenden!!!

Berechnung: **Th. 2.22**

Berechnung: Th. 2.22

- Rotes Arrangement R (r), Blaues Arrangement B (b), Punktmenge P (k), Schnittzellen Z_i

Berechnung: Th. 2.22

- Rotes Arrangement R (r), Blaues Arrangement B (b), Punktmenge P (k), Schnittzellen Z_i
- $|Z_1| + |Z_2| + \dots + |Z_\ell|$ in $O((r + b + k) \log(r + b + k))$ berechnen!

Berechnung: Th. 2.22

- Rotes Arrangement R (r), Blaues Arrangement B (b), Punktmenge P (k), Schnittzellen Z_i
- $|Z_1| + |Z_2| + \dots + |Z_\ell|$ in $O((r + b + k) \log(r + b + k))$ berechnen!
- Sweep Algorithmus

Berechnung: Th. 2.22

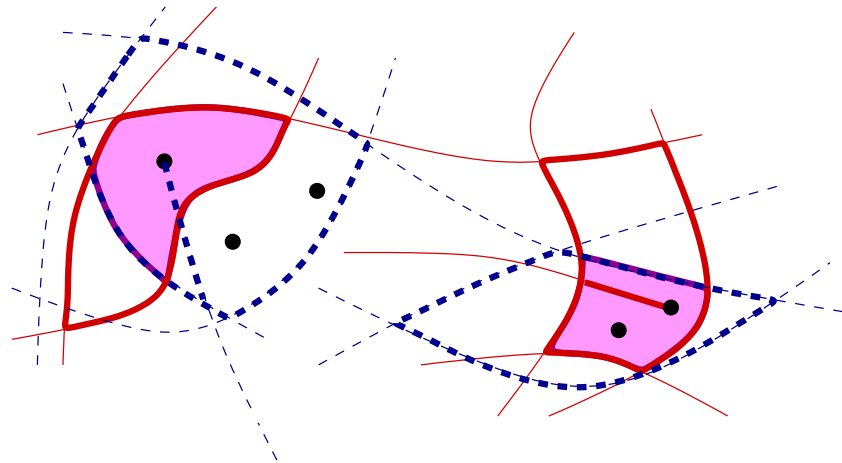
- Rotes Arrangement R (r), Blaues Arrangement B (b), Punktmenge P (k), Schnittzellen Z_i
- $|Z_1| + |Z_2| + \dots + |Z_\ell|$ in $O((r + b + k) \log(r + b + k))$ berechnen!
- Sweep Algorithmus
- P erweitern: *Innere* Endknoten von R und B : Wichtig!!

Berechnung: Th. 2.22

- Rotes Arrangement R (r), Blaues Arrangement B (b), Punktmenge P (k), Schnittzellen Z_i
- $|Z_1| + |Z_2| + \dots + |Z_\ell|$ in $O((r + b + k) \log(r + b + k))$ berechnen!
- Sweep Algorithmus
- P erweitern: *Innere* Endknoten von R und B : Wichtig!!
- Q : P und *innere* Endknoten

Berechnung: Th. 2.22

- Rotes Arrangement R (r), Blaues Arrangement B (b), Punktmenge P (k), Schnittzellen Z_i
- $|Z_1| + |Z_2| + \dots + |Z_\ell|$ in $O((r + b + k) \log(r + b + k))$ berechnen!
- Sweep Algorithmus
- P erweitern: *Innere* Endknoten von R und B : Wichtig!!
- Q : P und *innere* Endknoten



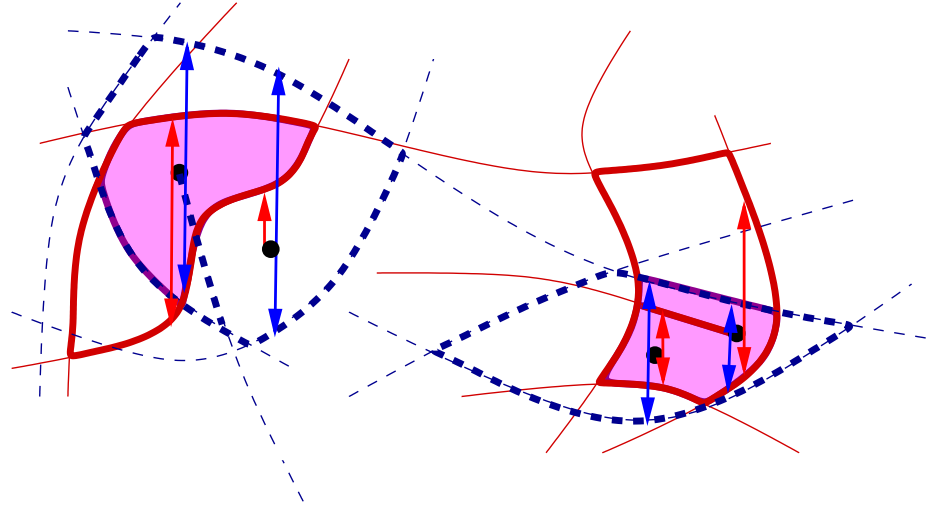
Alg. 2.5: Preprocessing!

Alg. 2.5: Preprocessing!

Für alle $q \in Q$ darüber/darunter-liegende Kante in R und B

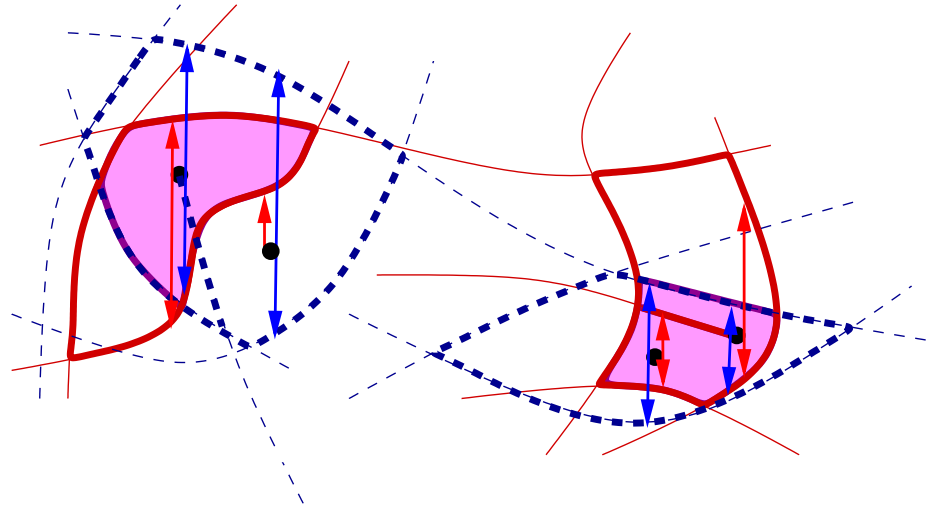
Alg. 2.5: Preprocessing!

Für alle $q \in Q$ darüber/darunter-liegende Kante in R und B



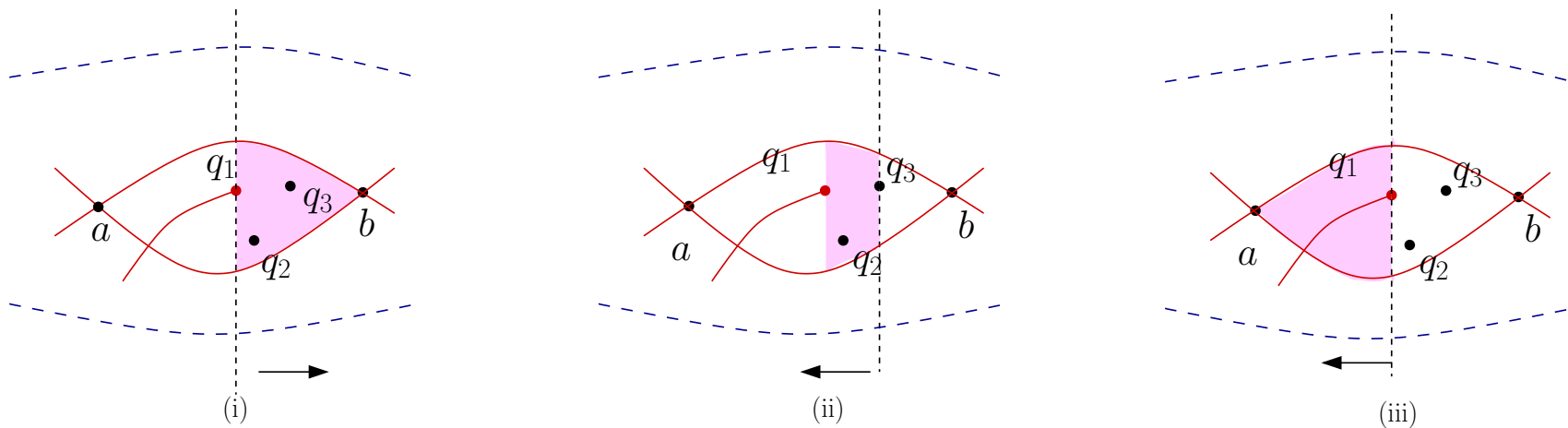
Alg. 2.5: Preprocessing!

Für alle $q \in Q$ darüber/darunter-liegende Kante in R und B



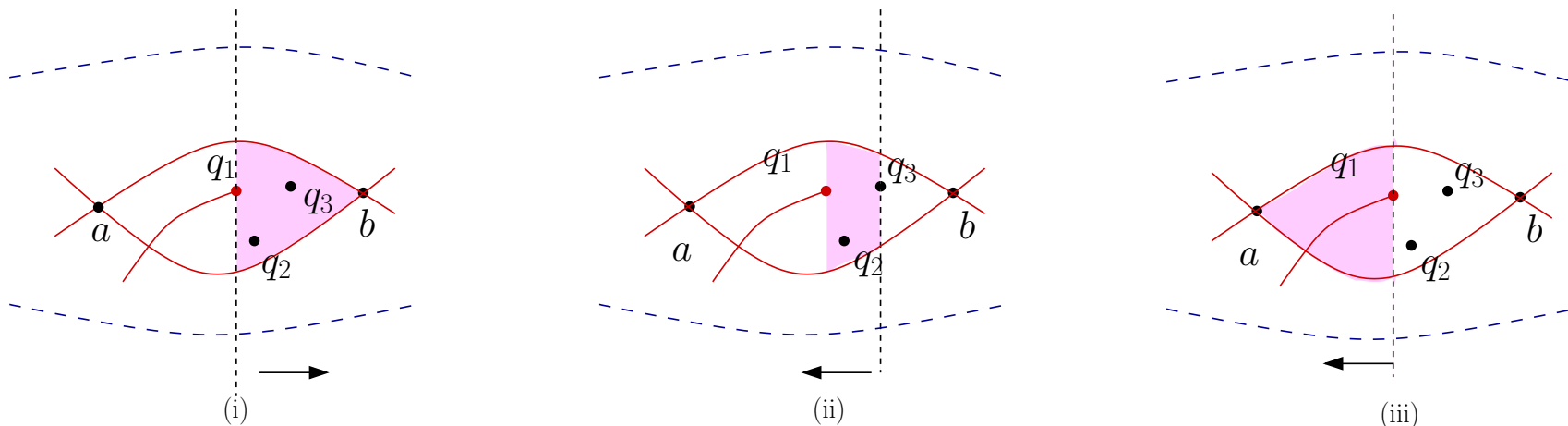
Durch Sweep in jedem Arrangement: Übung
 $O((r + b + k) \log(r + b + k))$

Alg. 2.5: Sweep in zwei Richtungen!



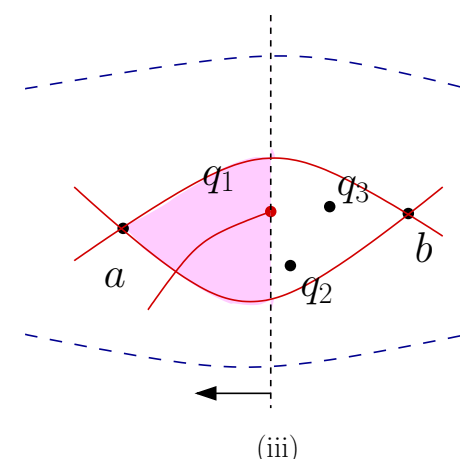
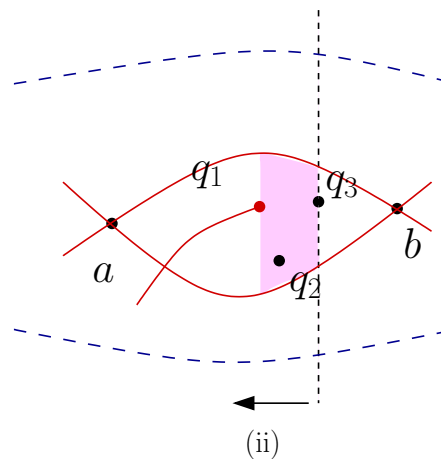
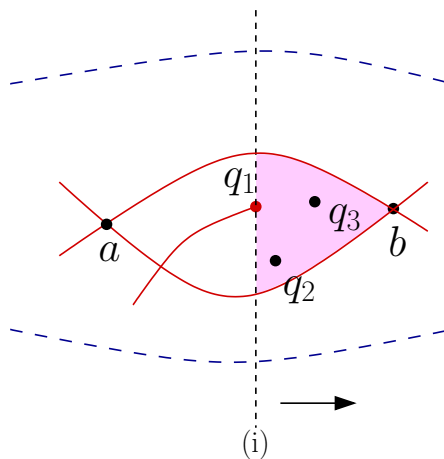
Alg. 2.5: Sweep in zwei Richtungen!

- i) Teile der Ergebniszellen: Rechts vom am weitesten links liegenden $q \in Q$ beginnen



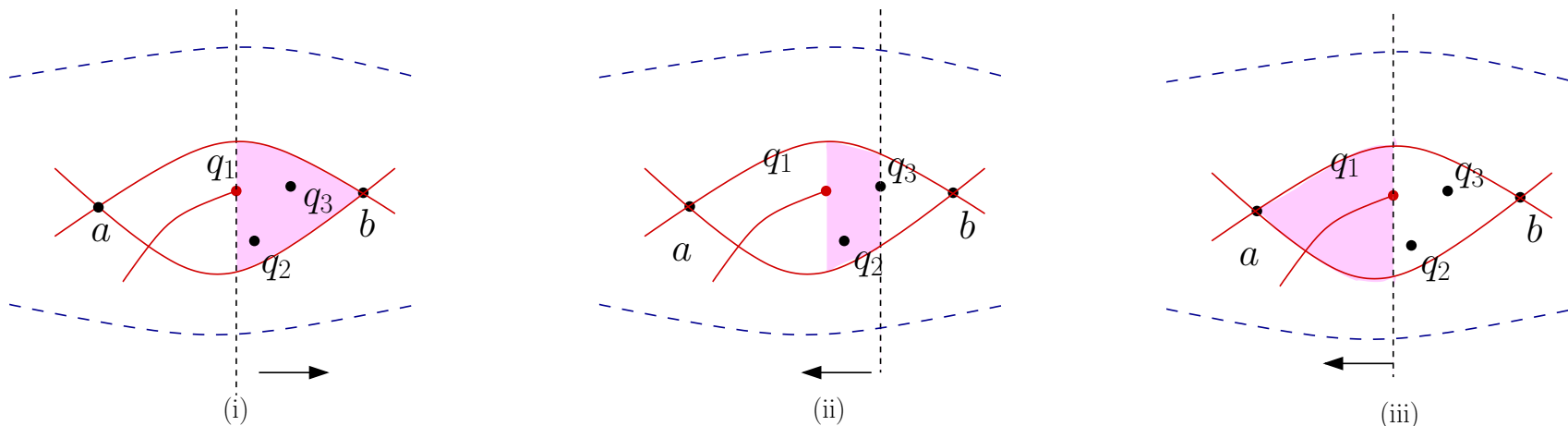
Alg. 2.5: Sweep in zwei Richtungen!

- i) Teile der Ergebniszellen: Rechts vom am weitesten links liegenden $q \in Q$ beginnen
- ii) Teile der Ergebniszellen: Links vom am weitesten rechts liegenden $q \in Q$ beginnen



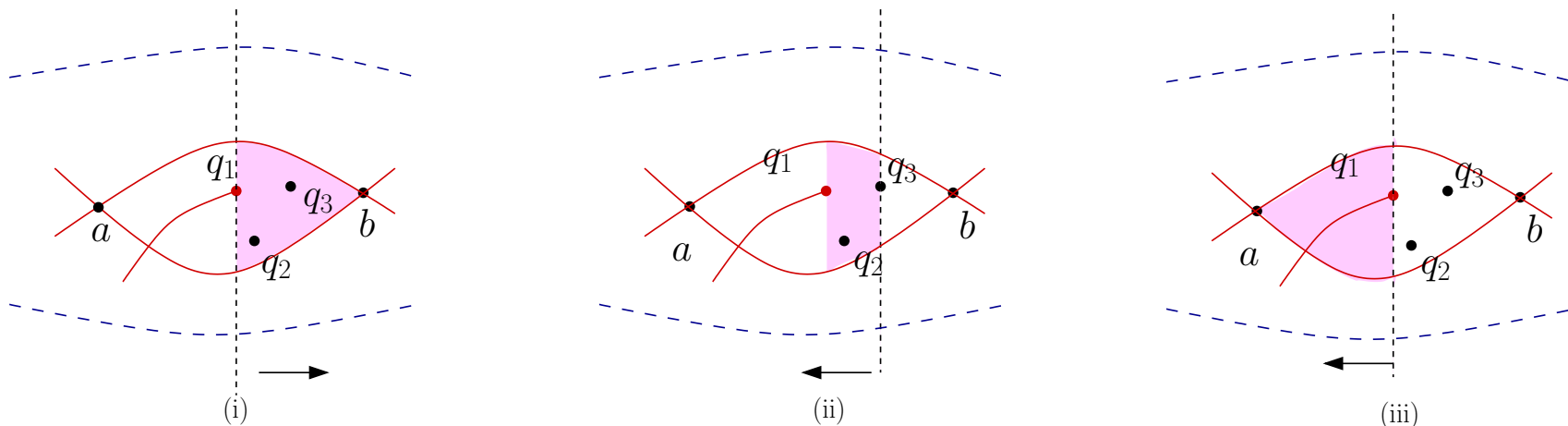
Alg. 2.5: Sweep in zwei Richtungen!

- i) Teile der Ergebniszellen: Rechts vom am weitesten links liegenden $q \in Q$ beginnen
- ii) Teile der Ergebniszellen: Links vom am weitesten rechts liegenden $q \in Q$ beginnen
- iii) Nochmals Aufteilen in Teilzellen

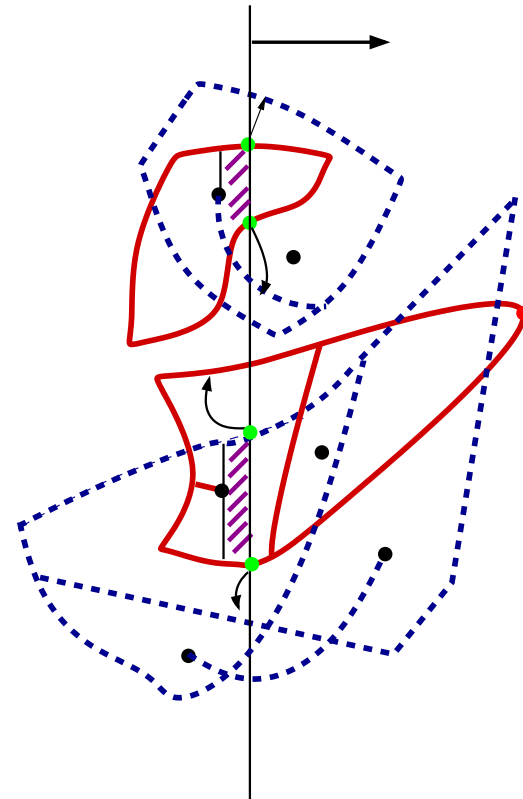


Alg. 2.5: Sweep in zwei Richtungen!

- i) Teile der Ergebniszellen: Rechts vom am weitesten links liegenden $q \in Q$ beginnen
- ii) Teile der Ergebniszellen: Links vom am weitesten rechts liegenden $q \in Q$ beginnen
- iii) Nochmals Aufteilen in Teilzellen
Dann Vereinigung!

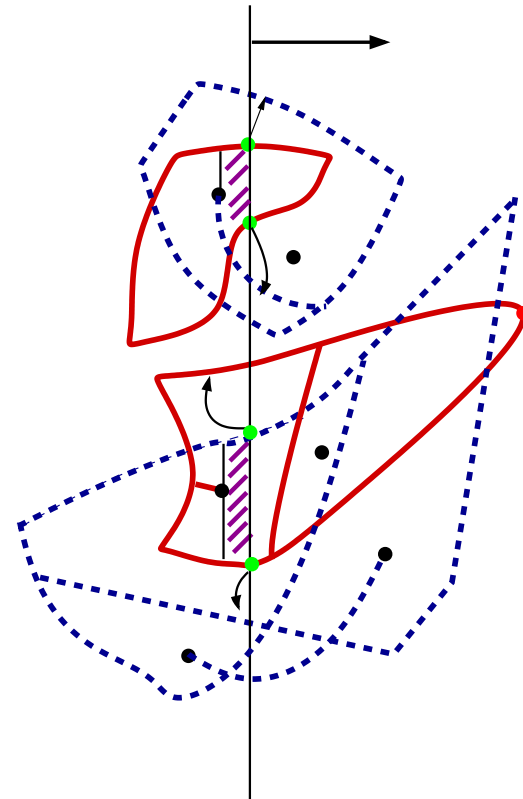


Alg. 2.6: Sweep (eine Richtung!)



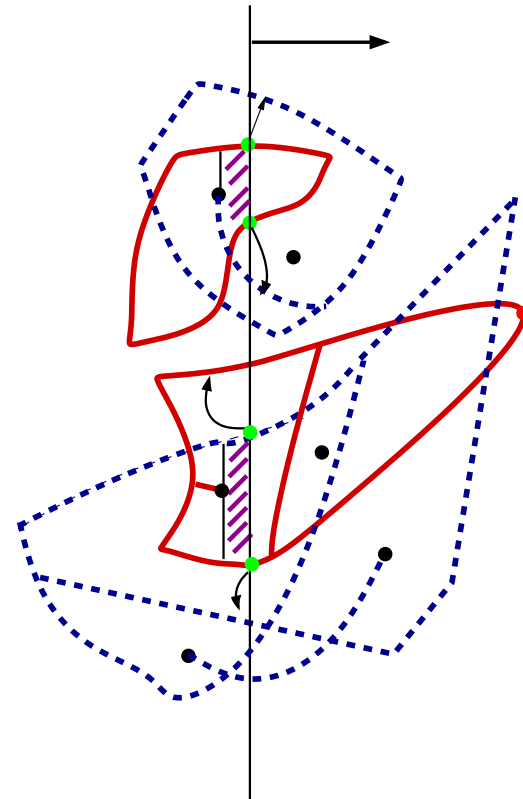
Alg. 2.6: Sweep (eine Richtung!)

- ES: nach X -Koord. sort.
Punkte aus Q + zusätzl. Ecken
von R und B .



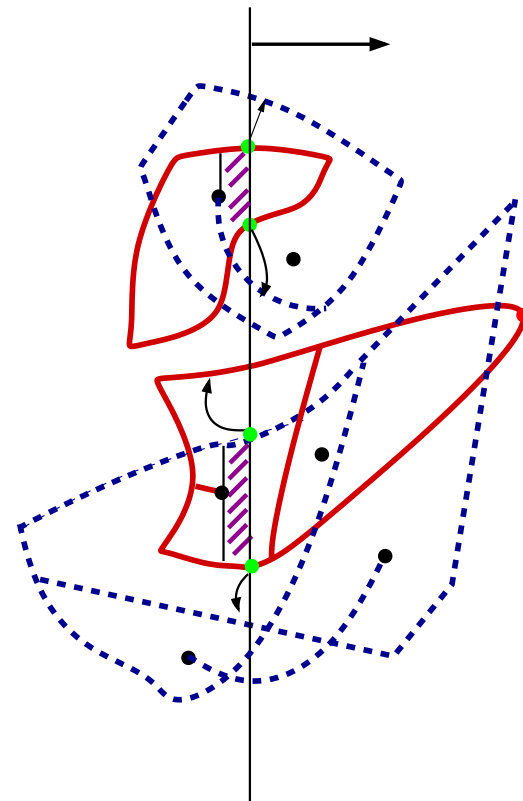
Alg. 2.6: Sweep (eine Richtung!)

- ES: nach X -Koord. sort.
Punkte aus Q + zusätzl. Ecken
von R und B .
- SSS: Zu jedem Zeitpunkt:



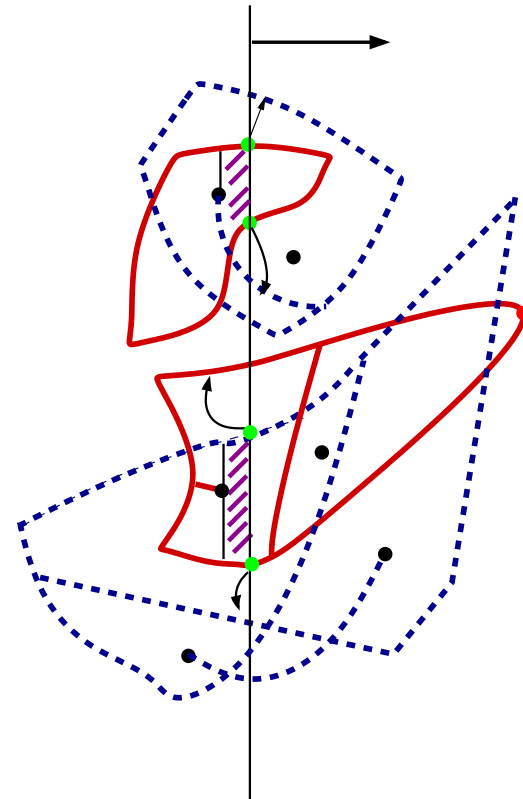
Alg. 2.6: Sweep (eine Richtung!)

- ES: nach X -Koord. sort.
Punkte aus Q + zusätzl. Ecken
von R und B .
- SSS: Zu jedem Zeitpunkt:
 - sortierte Folge der
Schnittzellen entlang der
Sweepeline



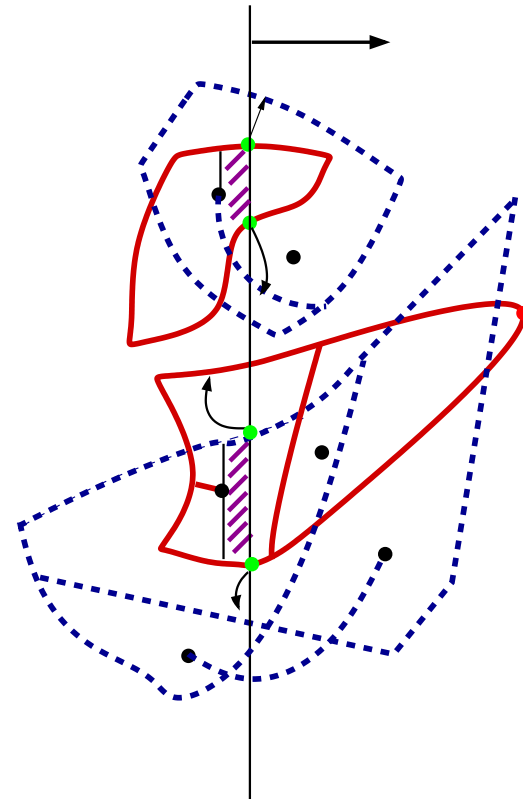
Alg. 2.6: Sweep (eine Richtung!)

- ES: nach X -Koord. sort.
Punkte aus Q + zusätzl. Ecken von R und B .
- SSS: Zu jedem Zeitpunkt:
 - sortierte Folge der Schnittzellen entlang der Sweepline
 - Schnittzellen haben Zeiger auf O/U Kanten

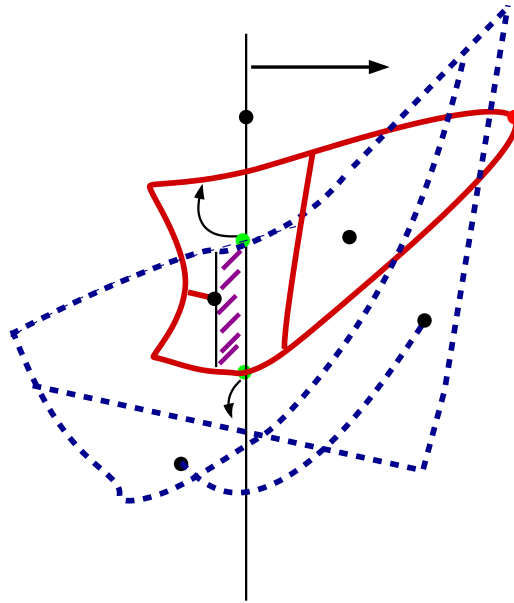


Alg. 2.6: Sweep (eine Richtung!)

- ES: nach X -Koord. sort.
Punkte aus Q + zusätzl. Ecken von R und B .
- SSS: Zu jedem Zeitpunkt:
 - sortierte Folge der Schnittzellen entlang der Sweepline
 - Schnittzellen haben Zeiger auf O/U Kanten
 - Scout läuft auf Schnittkante und bewacht mitentscheidene Kanten!!

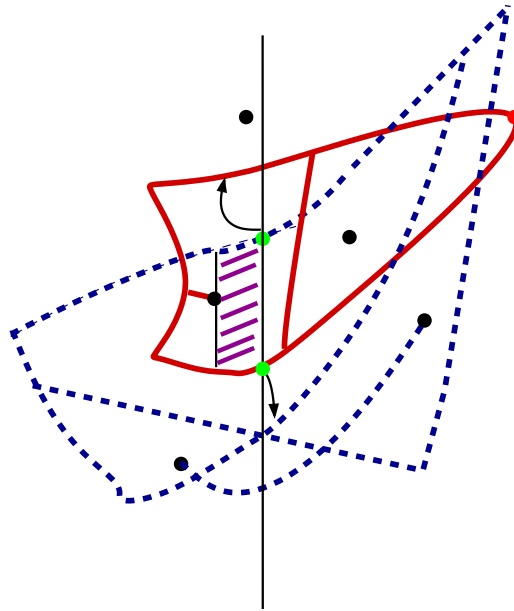


Alg. 2.6: Ereignisse!!



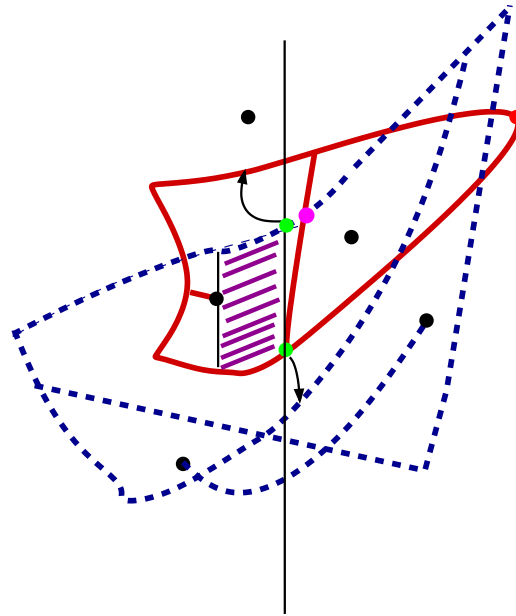
Alg. 2.6: Ereignisse!!

- Bewachte Kante wechselt! (Schnitte: Mit Bewacher?)



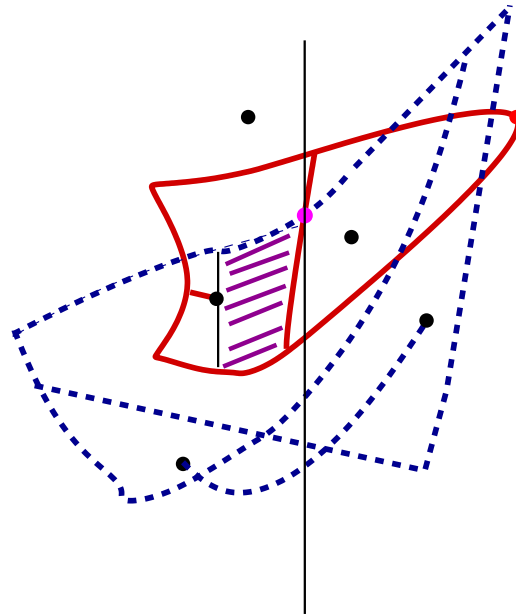
Alg. 2.6: Ereignisse!!

- Bewachte Kante wechselt! (Schnitte: Mit Bewacher?)
- Randkante wechselt! (Schnitte: Mit Bewacher/Mit Randkante?)



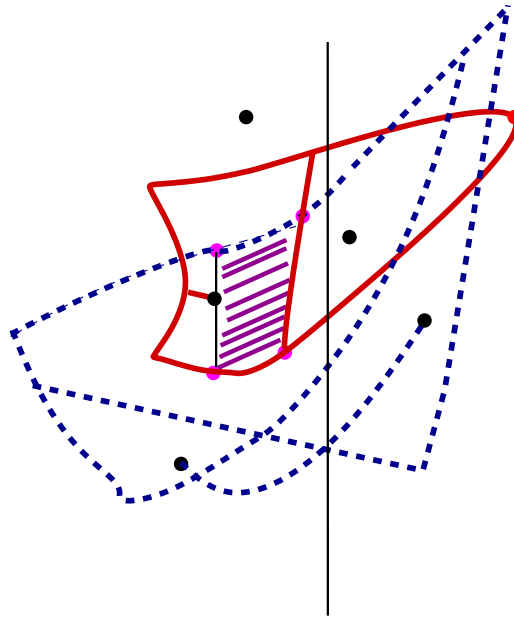
Alg. 2.6: Ereignisse!!

- Bewachte Kante wechselt! (Schnitte: Mit Bewacher?)
- Randkante wechselt! (Schnitte: Mit Bewacher/Mit Randkante?)
- Red/Blue Schnittpunkt: Region Ende oder Wechsel
Rand/Bewachte Kante



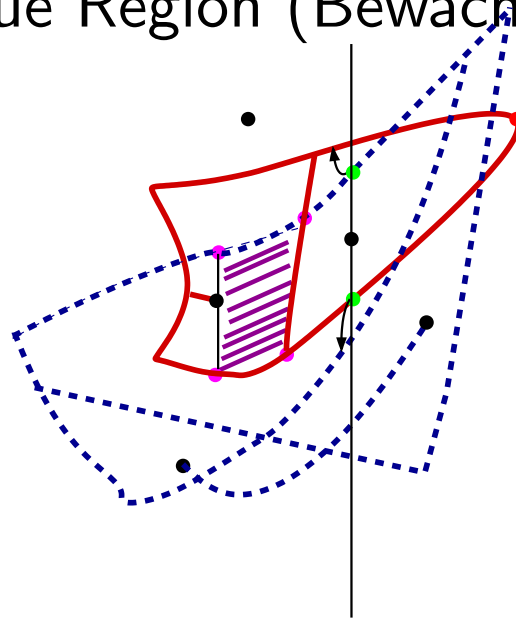
Alg. 2.6: Ereignisse!!

- Bewachte Kante wechselt! (Schnitte: Mit Bewacher?)
- Randkante wechselt! (Schnitte: Mit Bewacher/Mit Randkante?)
- Red/Blue Schnittpunkt: Region Ende oder Wechsel
Rand/Bewachte Kante



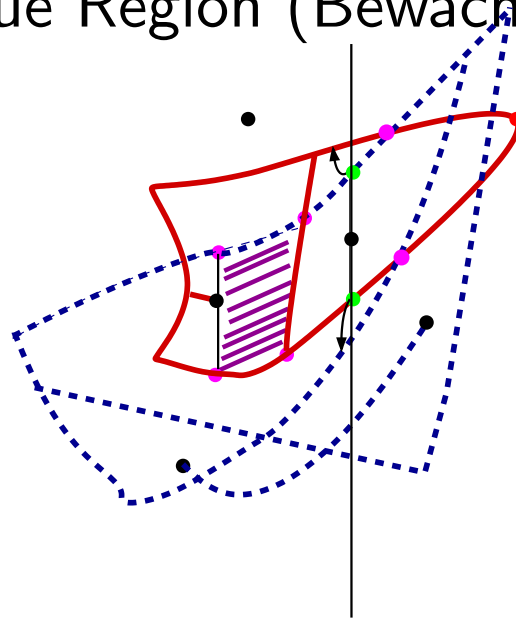
Alg. 2.6: Ereignisse!!

- Bewachte Kante wechselt! (Schnitte: Mit Bewacher?)
- Randkante wechselt! (Schnitte: Mit Bewacher/Mit Randkante?)
- Red/Blue Schnittpunkt: Region Ende oder Wechsel
Rand/Bewachte Kante
- Endpunkt aus Q : Neue Region (Bewachung/Schnitte)!!



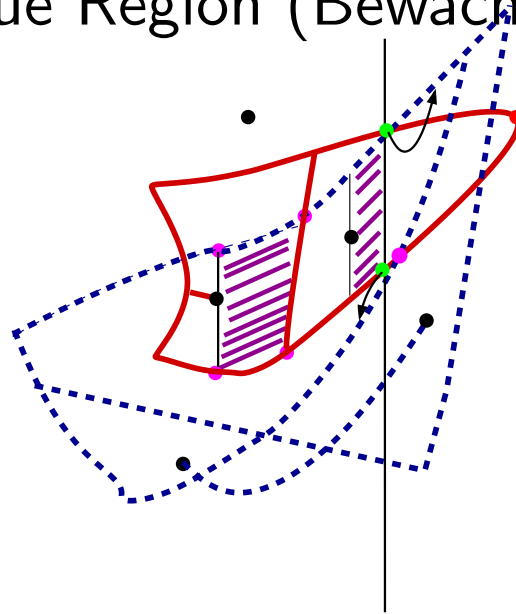
Alg. 2.6: Ereignisse!!

- Bewachte Kante wechselt! (Schnitte: Mit Bewacher?)
- Randkante wechselt! (Schnitte: Mit Bewacher/Mit Randkante?)
- Red/Blue Schnittpunkt: Region Ende oder Wechsel
Rand/Bewachte Kante
- Endpunkt aus Q : Neue Region (Bewachung/Schnitte)!!



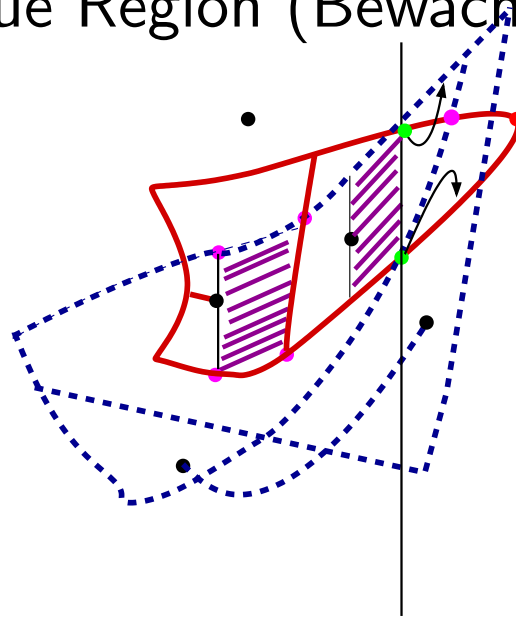
Alg. 2.6: Ereignisse!!

- Bewachte Kante wechselt! (Schnitte: Mit Bewacher?)
- Randkante wechselt! (Schnitte: Mit Bewacher/Mit Randkante?)
- Red/Blue Schnittpunkt: Region Ende oder Wechsel
Rand/Bewachte Kante
- Endpunkt aus Q : Neue Region (Bewachung/Schnitte)!!



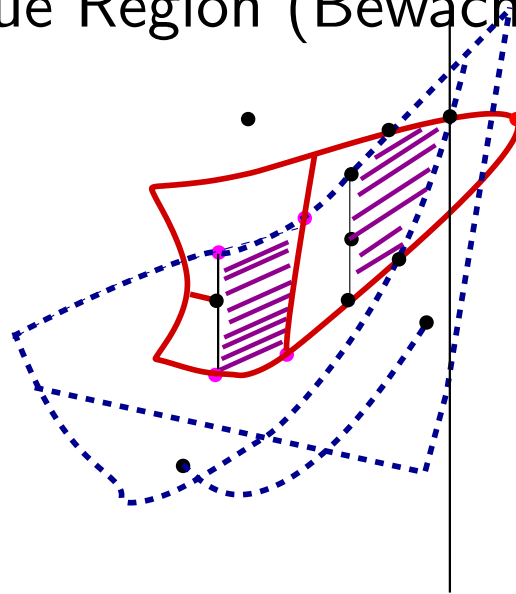
Alg. 2.6: Ereignisse!!

- Bewachte Kante wechselt! (Schnitte: Mit Bewacher?)
- Randkante wechselt! (Schnitte: Mit Bewacher/Mit Randkante?)
- Red/Blue Schnittpunkt: Region Ende oder Wechsel
Rand/Bewachte Kante
- Endpunkt aus Q : Neue Region (Bewachung/Schnitte)!!



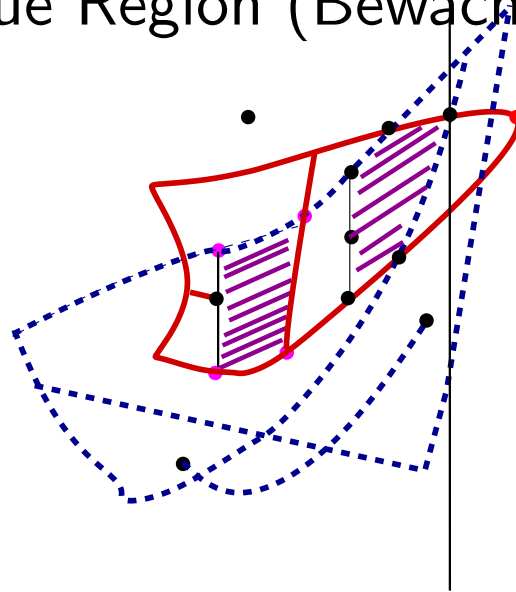
Alg. 2.6: Ereignisse!!

- Bewachte Kante wechselt! (Schnitte: Mit Bewacher?)
- Randkante wechselt! (Schnitte: Mit Bewacher/Mit Randkante?)
- Red/Blue Schnittpunkt: Region Ende oder Wechsel
Rand/Bewachte Kante
- Endpunkt aus Q : Neue Region (Bewachung/Schnitte)!!



Alg. 2.6: Ereignisse!!

- Bewachte Kante wechselt! (Schnitte: Mit Bewacher?)
- Randkante wechselt! (Schnitte: Mit Bewacher/Mit Randkante?)
- Red/Blue Schnittpunkt: Region Ende oder Wechsel
Rand/Bewachte Kante
- Endpunkt aus Q : Neue Region (Bewachung/Schnitte)!!



Beispiel (Tafel): **Alg. 2.6**: Ereignisse!!

Beispiel (Tafel): Alg. 2.6: Ereignisse!!

$$n = r + b + k$$

Beispiel (Tafel): Alg. 2.6: Ereignisse!!

$$n = r + b + k$$

1. Rot/Roter Schnittpunkt: Wechsel! Schnitte! $O(1)$
2. Blau/Blauer Schnittpunkt: Wechsel! Schnitte! $O(1)$
3. Rot/Blauer Schnittpunkt: Wechsel! Schnitte! $O(1)$
4. Neuer Punkt: Regionstart $O(1)$ Preprocessing! Einfügen in SSS: $O(\log n)$; Schnitte! $O(1)$

Beispiel (Tafel): Alg. 2.6: Ereignisse!!

$$n = r + b + k$$

1. Rot/Roter Schnittpunkt: Wechsel! Schnitte! $O(1)$
 2. Blau/Blauer Schnittpunkt: Wechsel! Schnitte! $O(1)$
 3. Rot/Blauer Schnittpunkt: Wechsel! Schnitte! $O(1)$
 4. Neuer Punkt: Regionstart $O(1)$ Preprocessing! Einfügen in SSS: $O(\log n)$; Schnitte! $O(1)$
- Schnitte Blau/Rot berechnen: 1), 2), 3), 4)!
 - A) Nächsten berechnet in $O(1)$
 - B) Einfügen in ES: $O(\log n)$: Begründung!

Analyse: Red-Blue Merge **Th. 2.22**

Analyse: Red-Blue Merge **Th. 2.22**

$$n = r + b + k$$

Analyse: Red-Blue Merge **Th. 2.22**

$$n = r + b + k$$

- Nie mehr als $O(r + b + k)$ Punkte in ES

Analyse: Red-Blue Merge **Th. 2.22**

$$n = r + b + k$$

- Nie mehr als $O(r + b + k)$ Punkte in ES
- Nie mehr als $O(r + b + k)$ Regionen in SSS

Analyse: Red-Blue Merge **Th. 2.22**

$$n = r + b + k$$

- Nie mehr als $O(r + b + k)$ Punkte in ES
- Nie mehr als $O(r + b + k)$ Regionen in SSS
- Einfügen in ES: $O(\log(r + b + k))$

Analyse: Red-Blue Merge **Th. 2.22**

$$n = r + b + k$$

- Nie mehr als $O(r + b + k)$ Punkte in ES
- Nie mehr als $O(r + b + k)$ Regionen in SSS
- Einfügen in ES: $O(\log(r + b + k))$
- Einfügen in SSS: $O(\log(r + b + k))$

Analyse: Red-Blue Merge **Th. 2.22**

$$n = r + b + k$$

- Nie mehr als $O(r + b + k)$ Punkte in ES
- Nie mehr als $O(r + b + k)$ Regionen in SSS
- Einfügen in ES: $O(\log(r + b + k))$
- Einfügen in SSS: $O(\log(r + b + k))$
- Nicht mehr als $O(r + b + k)$ Ereignisse:

Analyse: Red-Blue Merge **Th. 2.22**

$$n = r + b + k$$

- Nie mehr als $O(r + b + k)$ Punkte in ES
- Nie mehr als $O(r + b + k)$ Regionen in SSS
- Einfügen in ES: $O(\log(r + b + k))$
- Einfügen in SSS: $O(\log(r + b + k))$
- Nicht mehr als $O(r + b + k)$ Ereignisse:
 - Rot/Rot, Blau/Blau:

Analyse: Red-Blue Merge **Th. 2.22**

$$n = r + b + k$$

- Nie mehr als $O(r + b + k)$ Punkte in ES
- Nie mehr als $O(r + b + k)$ Regionen in SSS
- Einfügen in ES: $O(\log(r + b + k))$
- Einfügen in SSS: $O(\log(r + b + k))$
- Nicht mehr als $O(r + b + k)$ Ereignisse:
 - Rot/Rot, Blau/Blau: r und b
 - Rot/Blau, neu aber in $O(r + b + k)$

Analyse: Red-Blue Merge **Th. 2.22**

$$n = r + b + k$$

- Nie mehr als $O(r + b + k)$ Punkte in ES
- Nie mehr als $O(r + b + k)$ Regionen in SSS
- Einfügen in ES: $O(\log(r + b + k))$
- Einfügen in SSS: $O(\log(r + b + k))$
- Nicht mehr als $O(r + b + k)$ Ereignisse:
 - Rot/Rot, Blau/Blau: r und b
 - Rot/Blau, neu aber in $O(r + b + k)$
 - Neue Punkte:

Analyse: Red-Blue Merge **Th. 2.22**

$$n = r + b + k$$

- Nie mehr als $O(r + b + k)$ Punkte in ES
- Nie mehr als $O(r + b + k)$ Regionen in SSS
- Einfügen in ES: $O(\log(r + b + k))$
- Einfügen in SSS: $O(\log(r + b + k))$
- Nicht mehr als $O(r + b + k)$ Ereignisse:
 - Rot/Rot, Blau/Blau: r und b
 - Rot/Blau, neu aber in $O(r + b + k)$
 - Neue Punkte: max. $k + r + b$

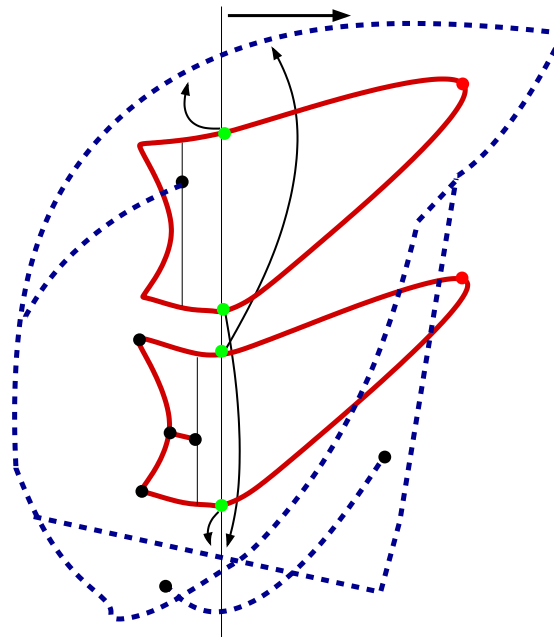
Analyse: Red-Blue Merge **Th. 2.22**

$$n = r + b + k$$

- Nie mehr als $O(r + b + k)$ Punkte in ES
- Nie mehr als $O(r + b + k)$ Regionen in SSS
- Einfügen in ES: $O(\log(r + b + k))$
- Einfügen in SSS: $O(\log(r + b + k))$
- Nicht mehr als $O(r + b + k)$ Ereignisse:
 - Rot/Rot, Blau/Blau: r und b
 - Rot/Blau, neu aber in $O(r + b + k)$
 - Neue Punkte: max. $k + r + b$

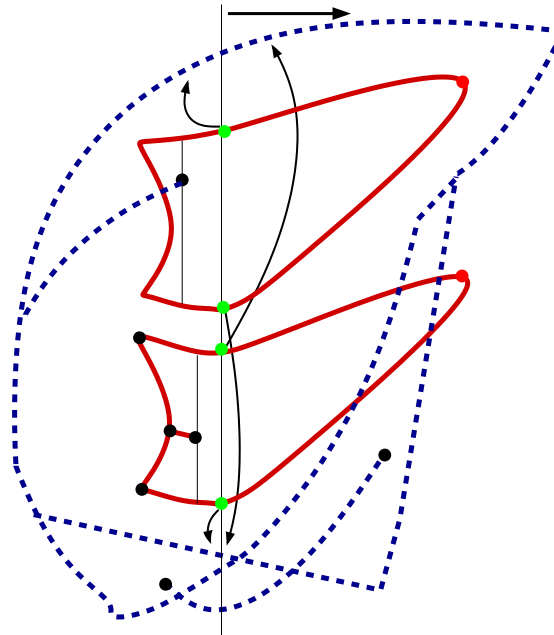
Insgesamt: $O(n \log n)$

Analyse: Red-Blue Merge **Th. 2.22**



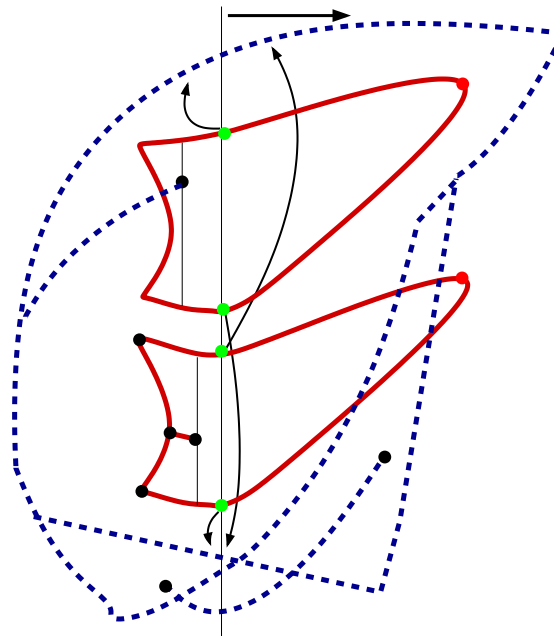
Analyse: Red-Blue Merge **Th. 2.22**

- Problem (Besonderheiten):



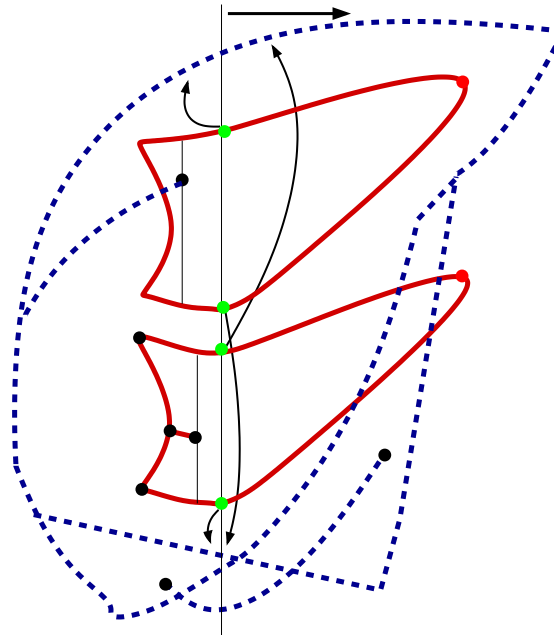
Analyse: Red-Blue Merge **Th. 2.22**

- Problem (Besonderheiten):
- Eine bewachte Kante gehört zu vielen Zellen!



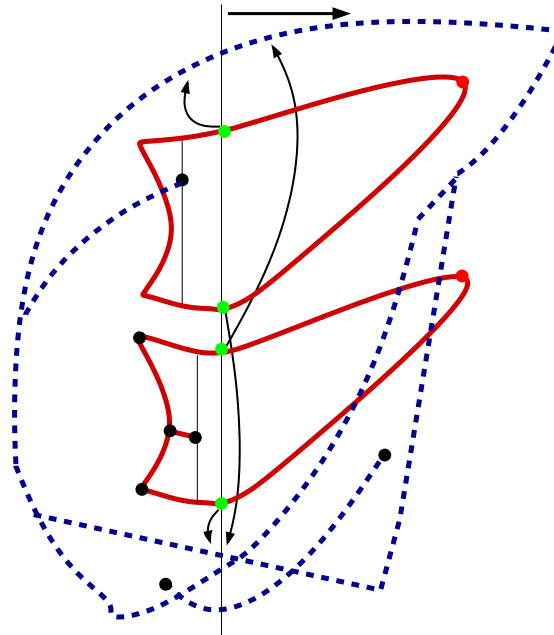
Analyse: Red-Blue Merge **Th. 2.22**

- Problem (Besonderheiten):
- Eine bewachte Kante gehört zu vielen Zellen!
- Nicht für alle Zellen prüfen!!



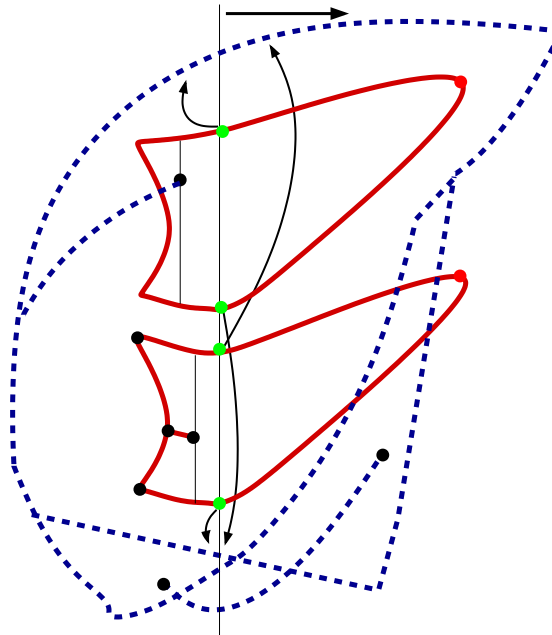
Analyse: Red-Blue Merge **Th. 2.22**

- Problem (Besonderheiten):
- Eine bewachte Kante gehört zu vielen Zellen!
- Nicht für alle Zellen prüfen!!
- Abhilfe: In Listen zusammenfassen!



Analyse: Red-Blue Merge **Th. 2.22**

- Problem (Besonderheiten):
- Eine bewachte Kante gehört zu vielen Zellen!
- Nicht für alle Zellen prüfen!!
- Abhilfe: In Listen zusammenfassen!
- Nur mit oberen/unteren den Schnitt testen!



Initialisierung Red-Blue Merge **Th. 2.22**

Initialisierung Red-Blue Merge **Th. 2.22**

- Weitere Besonderheiten

Initialisierung Red-Blue Merge **Th. 2.22**

- Weitere Besonderheiten
- Initialisierung

Initialisierung Red-Blue Merge **Th. 2.22**

- Weitere Besonderheiten
- Initialisierung
- Zu Beginn nur zwei einzelne Bögen

Initialisierung Red-Blue Merge **Th. 2.22**

- Weitere Besonderheiten
- Initialisierung
- Zu Beginn nur zwei einzelne Bögen
- Natürliche Begrenzung (häufig)

Initialisierung Red-Blue Merge **Th. 2.22**

- Weitere Besonderheiten
- Initialisierung
- Zu Beginn nur zwei einzelne Bögen
- Natürliche Begrenzung (häufig)
- Unendliche Zellen

Initialisierung Red-Blue Merge **Th. 2.22**

- Weitere Besonderheiten
- Initialisierung
- Zu Beginn nur zwei einzelne Bögen
- Natürliche Begrenzung (häufig)
- Unendliche Zellen
- Beispiel: Tafel!

Zellenberechnung: **Th. 2.23**

Zellenberechnung: **Th. 2.23**

n X -monotone Kurvenstücke von denen sich zwei nur s mal schneiden, x gegeben. Zelle Z_x kann in Zeit $O(\lambda_{s+2}(n)) \log^2 n$ berechnet werden.

Fahrplan!!

- Divide and Conquer!!
- Teile Segmente in zwei gleichgroße Mengen Z_1, Z_2
- Berechne Z_{1x} und Z_{2x}
- Merge zu $\{Z_1 \cup Z_2\}_x$
- Spezieller Merge wegen Schnitt mit x
- RED BLUE Merge
- Merge: Komplexität des Ergebnisses

Beweis: **Th. 2.22**

Beweis: Th. 2.22

Divide and Conquer

Beweis: Th. 2.22

Divide and Conquer

$$T(n) \leq 2T\left(\frac{n}{2}\right)$$

Beweis: Th. 2.22

Divide and Conquer

$$T(n) \leq 2T\left(\frac{n}{2}\right) \text{ (Rek.)}$$

Beweis: Th. 2.22

Divide and Conquer

$$T(n) \leq 2T\left(\frac{n}{2}\right) \text{ (Rek.)} + C \times \lambda_{s+2}(n) \log(\lambda_{s+2}(n))$$

Beweis: Th. 2.22

Divide and Conquer

$$T(n) \leq 2T\left(\frac{n}{2}\right) \text{ (Rek.)} + C \times \lambda_{s+2}(n) \log(\lambda_{s+2}(n)) \text{ (R/B-Merge)}$$

Beweis: Th. 2.22

Divide and Conquer

$$\begin{aligned} T(n) &\leq 2T\left(\frac{n}{2}\right) \text{ (Rek.)} + C \times \lambda_{s+2}(n) \log(\lambda_{s+2}(n)) \text{ (R/B-Merge)} \\ &\leq 2\left(2T\left(\frac{n}{4}\right) + C\lambda_{s+2}\left(\frac{n}{2}\right) \log\frac{n}{2}\right) + C \times \lambda_{s+2}(n) \log n \end{aligned}$$

Beweis: Th. 2.22

Divide and Conquer

$$\begin{aligned} T(n) &\leq 2T\left(\frac{n}{2}\right) \text{ (Rek.)} + C \times \lambda_{s+2}(n) \log(\lambda_{s+2}(n)) \text{ (R/B-Merge)} \\ &\leq 2\left(2T\left(\frac{n}{4}\right) + C\lambda_{s+2}\left(\frac{n}{2}\right) \log\frac{n}{2}\right) + C \times \lambda_{s+2}(n) \log n \\ &\vdots \quad \vdots \end{aligned}$$

Beweis: Th. 2.22

Divide and Conquer

$$\begin{aligned} T(n) &\leq 2T\left(\frac{n}{2}\right) \text{ (Rek.)} + C \times \lambda_{s+2}(n) \log(\lambda_{s+2}(n)) \text{ (R/B-Merge)} \\ &\leq 2\left(2T\left(\frac{n}{4}\right) + C\lambda_{s+2}\left(\frac{n}{2}\right) \log\frac{n}{2}\right) + C \times \lambda_{s+2}(n) \log n \\ &\quad \vdots \quad \vdots \\ &\leq (n)(T(1) + C) + C \sum_{i=0}^{\log n} \left(\lambda_{s+2}(n) \log \frac{n}{2^i}\right) \end{aligned}$$

Beweis: Th. 2.22

Divide and Conquer

$$\begin{aligned} T(n) &\leq 2T\left(\frac{n}{2}\right) \text{ (Rek.)} + C \times \lambda_{s+2}(n) \log(\lambda_{s+2}(n)) \text{ (R/B-Merge)} \\ &\leq 2\left(2T\left(\frac{n}{4}\right) + C\lambda_{s+2}\left(\frac{n}{2}\right) \log\frac{n}{2}\right) + C \times \lambda_{s+2}(n) \log n \\ &\quad \vdots \\ &\leq (n) (T(1) + C) + C \sum_{i=0}^{\log n} \left(\lambda_{s+2}(n) \log \frac{n}{2^i}\right) \\ &\in O(\lambda_{s+2}(n) \log^2 n) \end{aligned}$$

Anwendungen: **Kap. 2.2.2**

Anwendungen: Kap. 2.2.2

- Polygonaler Roboter R mit $|R| = m$

Anwendungen: Kap. 2.2.2

- Polygonaler Roboter R mit $|R| = m$
- Polygonale Szene n Ecken

Anwendungen: Kap. 2.2.2

- Polygonaler Roboter R mit $|R| = m$
- Polygonale Szene n Ecken
- Reine Translationsbewegung

Anwendungen: Kap. 2.2.2

- Polygonaler Roboter R mit $|R| = m$
- Polygonale Szene n Ecken
- Reine Translationsbewegung
- Startposition s , Endposition t

Anwendungen: Kap. 2.2.2

- Polygonaler Roboter R mit $|R| = m$
- Polygonale Szene n Ecken
- Reine Translationsbewegung
- Startposition s , Endposition t
- Kollisionsfreie Bahn von s nach t

Alg. 2.7

Preprocessing:

Preprocessing:

- Arrang. $2mn$ Linienseg.(Ecke/Kante)

Preprocessing:

- Arrang. $2mn$ Linienseg.(Ecke/Kante)
- Ber. Zelle Z , die s enthält:

Preprocessing:

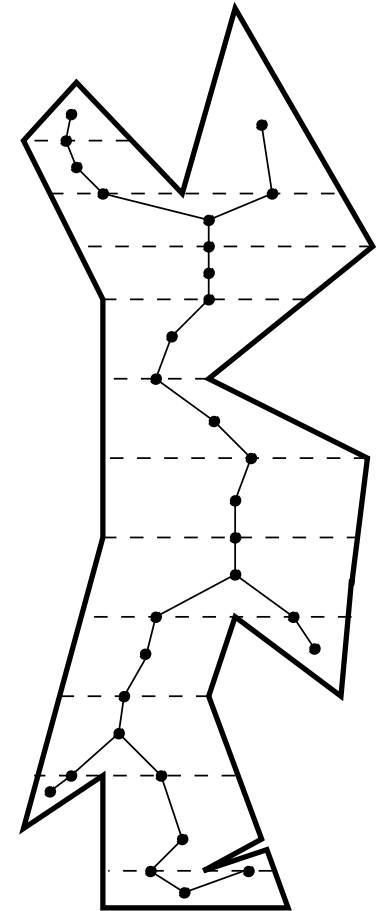
- Arrang. $2mn$ Linienseg.(Ecke/Kante)
- Ber. Zelle Z , die s enthält:
Komplexität $O(\lambda_{(1+2)}(mn))$,

Preprocessing:

- Arrang. $2mn$ Linienseg.(Ecke/Kante)
- Ber. Zelle Z , die s enthält:
Komplexität $O(\lambda_{(1+2)}(mn))$,
Laufzeit $O(\lambda_{(1+2)}(mn) \log^2(mn))$

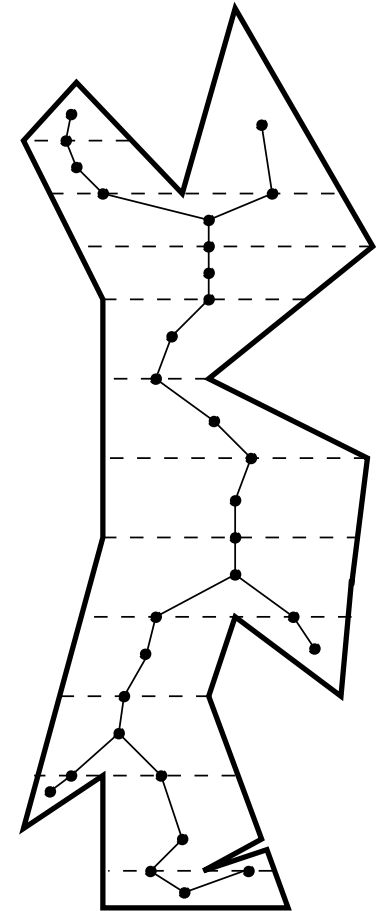
Preprocessing:

- Arrang. $2mn$ Linienseg.(Ecke/Kante)
- Ber. Zelle Z , die s enthält:
Komplexität $O(\lambda_{(1+2)}(mn))$,
Laufzeit $O(\lambda_{(1+2)}(mn) \log^2(mn))$
- Trapezzerlegung,



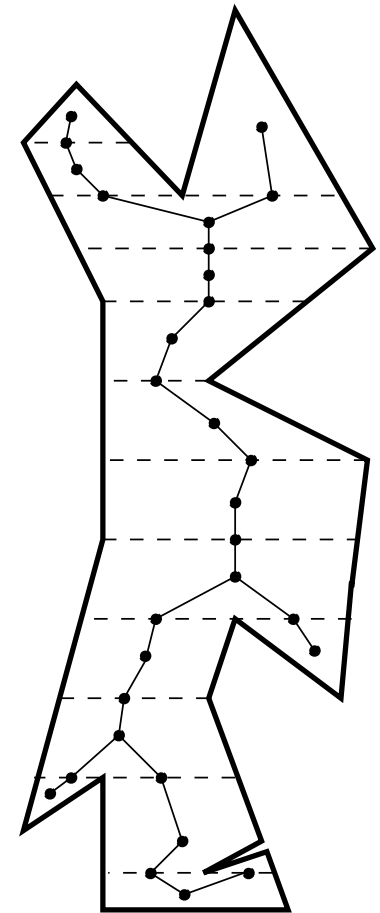
Preprocessing:

- Arrang. $2mn$ Linienseg.(Ecke/Kante)
- Ber. Zelle Z , die s enthält:
Komplexität $O(\lambda_{(1+2)}(mn))$,
Laufzeit $O(\lambda_{(1+2)}(mn) \log^2(mn))$
- Trapezzerlegung,
Zusammenhangsgraph: Seidel



Preprocessing:

- Arrang. $2mn$ Linienseg.(Ecke/Kante)
- Ber. Zelle Z , die s enthält:
Komplexität $O(\lambda_{(1+2)}(mn))$,
Laufzeit $O(\lambda_{(1+2)}(mn) \log^2(mn))$
- Trapezzerlegung,
Zusammenhangsgraph: Seidel
 $O(\lambda_{(1+2)}(mn) \log^*(mn))$ (Sweep)

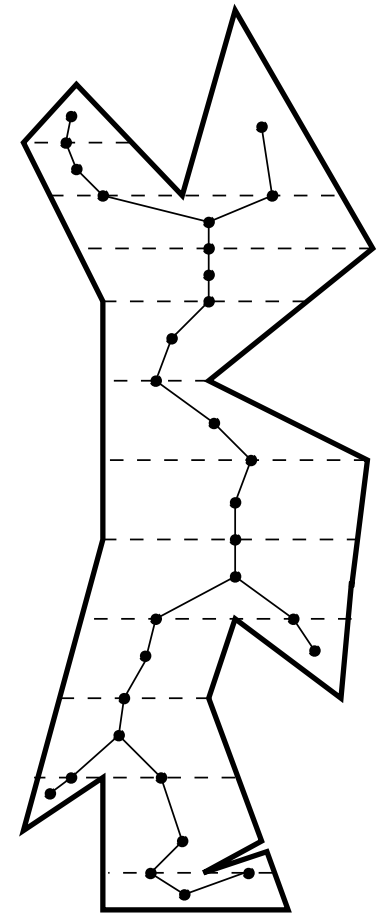


Preprocessing:

- Arrang. $2mn$ Linienseg.(Ecke/Kante)
- Ber. Zelle Z , die s enthält:
Komplexität $O(\lambda_{(1+2)}(mn))$,
Laufzeit $O(\lambda_{(1+2)}(mn) \log^2(mn))$
- Trapezzerlegung,
Zusammenhangsgraph: Seidel
 $O(\lambda_{(1+2)}(mn) \log^*(mn))$ (Sweep)

Query: gegebenes t :

- Trapez, das t enthält: $O(\log(mn))$

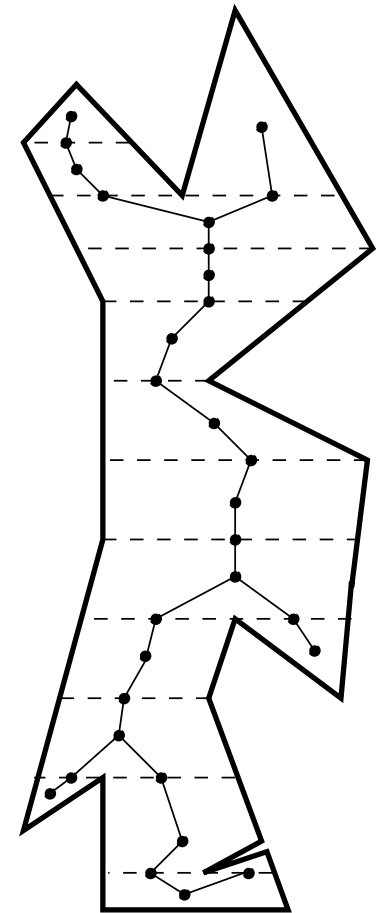


Preprocessing:

- Arrang. $2mn$ Linienseg.(Ecke/Kante)
- Ber. Zelle Z , die s enthält:
Komplexität $O(\lambda_{(1+2)}(mn))$,
Laufzeit $O(\lambda_{(1+2)}(mn) \log^2(mn))$
- Trapezzerlegung,
Zusammenhangsgraph: Seidel
 $O(\lambda_{(1+2)}(mn) \log^*(mn))$ (Sweep)

Query: gegebenes t :

- Trapez, das t enthält: $O(\log(mn))$
- Pfad s nach t im Zusammenhangsgraph:
 $O(\lambda_{(1+2)}(mn))$



Theorem 2.24

Theorem 2.24

Translation von R polygonaler Roboter mit m Ecken, in einer Umgebung mit polygonalen Hindernissen P_i mit insgesamt n Ecken.

Theorem 2.24

Translation von R polygonaler Roboter mit m Ecken, in einer Umgebung mit polygonalen Hindernissen P_i mit insgesamt n Ecken.

Gegeben seien Start- und Zielposition s, t .

Theorem 2.24

Translation von R polygonaler Roboter mit m Ecken, in einer Umgebung mit polygonalen Hindernissen P_i mit insgesamt n Ecken.

Gegeben seien Start- und Zielposition s, t .

Dann kann in Zeit $O(mn \alpha(mn) \log^2(mn))$ eine kollisionsfreie Translation von s nach t bestimmt werden oder festgestellt werden, dass keine solche existiert.