Karlsruher Institut für Technologie

Theoretische Grundlagen der Informatik

Map Labeling

INSTITUT FÜR THEORETISCHE INFORMATIK

Problem MAP LABELING 1

Karlsruher Institut für Technologie

Problem MAP LABELING 1

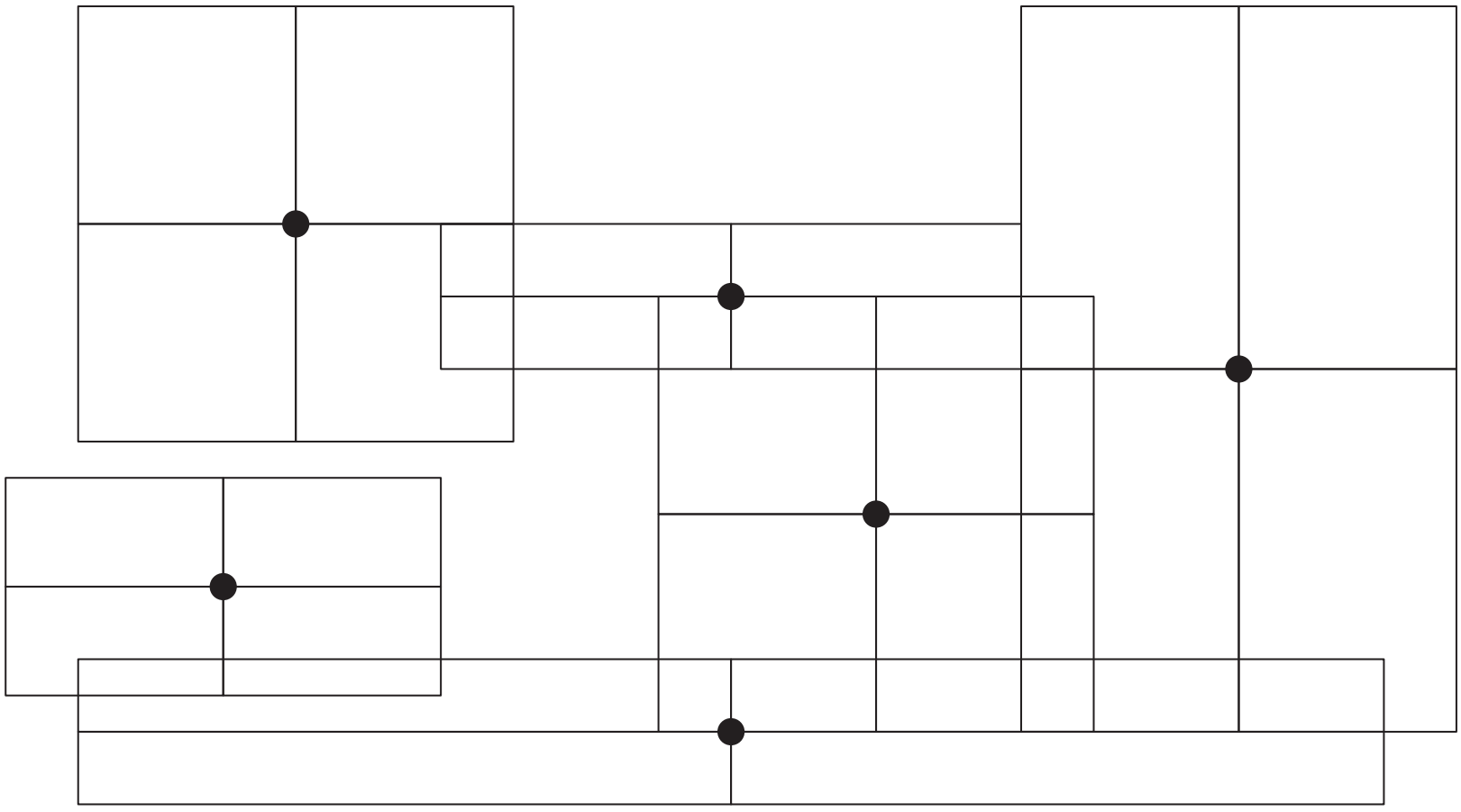
Gegeben: Punkte $\{p_1, \ldots, p_n\}$ paarweise verschieden in $\mathbb{R}^2$
Zu jedem Punkt p_i ein achsenparalleles Rechteck R_i

Frage: Können diese Rechtecke so horizontal und vertikal verschoben werden, dass

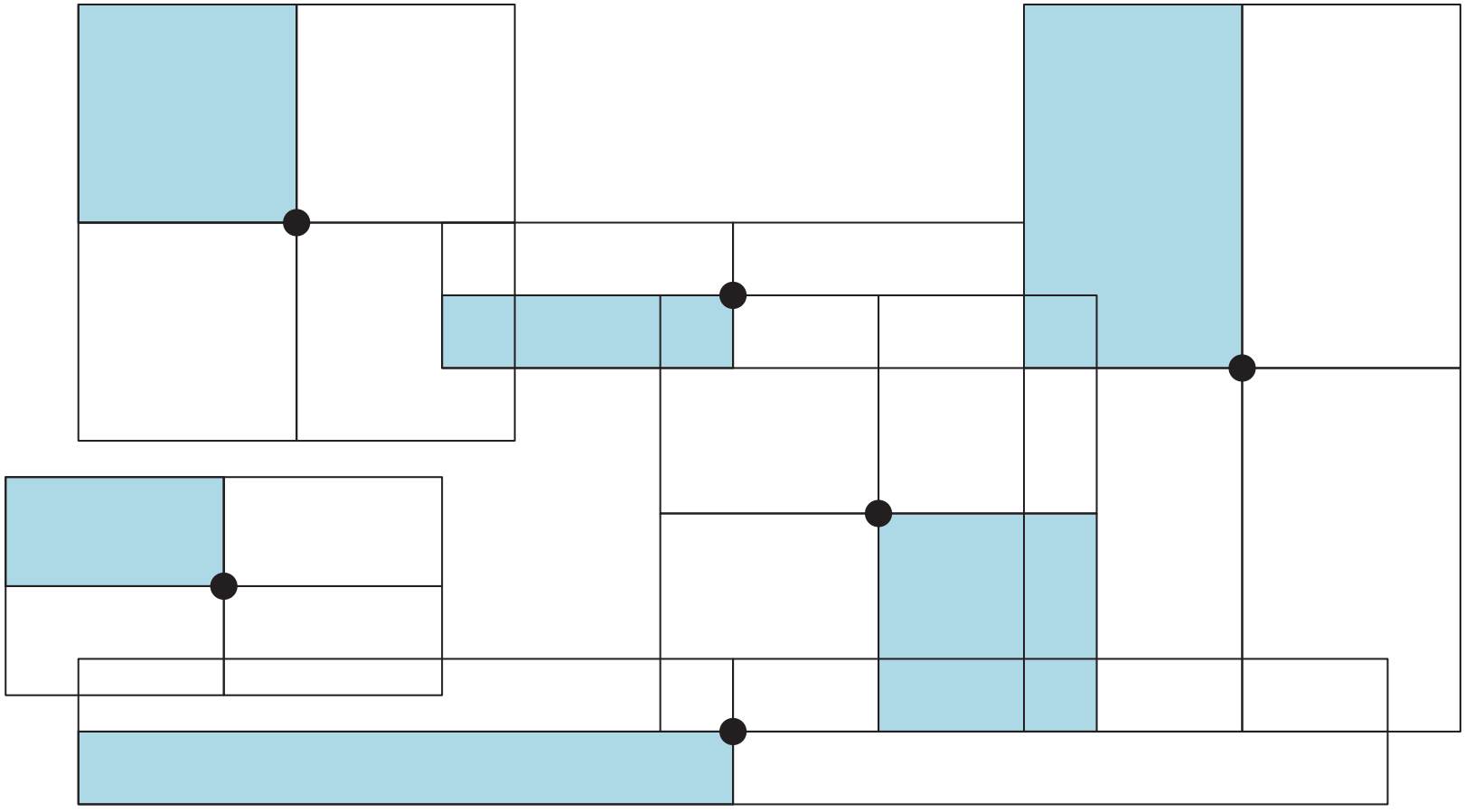
- alle Rechtecke paarweise disjunkt sind
- ein Eckpunkt von R_i gerade p_i ist für $1 \leq i \leq n$

- Es gibt für jedes Rechteck R_i also vier mögliche Positionen.
- **Optimierungsversion:**
  Maximiere die Anzahl der zulässig positionierten Rechtecke.

Beispiel



Beispiel



Problem MAP LABELING 2

Karlsruher Institut für Technologie

Problem MAP LABELING 2

Gegeben: Punkte $\{p_1, \ldots, p_n\}$ paarweise verschieden in $\mathbb{R}^2$
Zahl $q \in \mathbb{R}$

Frage: Können an den Punkten achsenparallele abgeschlossene Quadrate mit Seitenlänge q positioniert werden, so dass

- alle Quadrate paarweise disjunkt sind
- p_i Eckpunkt genau eines Quadrates ist für $1 \leq i \leq n$

- **Optimierungsversion:**
  Maximiere q.

Fahrplan

- Wir beweisen, dass **MAP LABELING 2 $\mathcal{NP}$-vollständig** ist.
- Damit ist **MAP LABELING 1 $\mathcal{NP}$-vollständig**.
- Für **MAP LABELING 2** - Optimierungsvariante geben wir einen **Approximationsalgorithmus** mit relativer Gütegarantie 2 an.
- Wir zeigen: Unter $\mathcal{P} \neq \mathcal{NP}$ gibt es **keinen Approximationsalgorithmus für MAP LABELING 2** - Optimierungsvariante mit relativer Gütegarantie kleiner 2.
- Für einen **Spezialfall von MAP LABELING 1** geben wir einen **polynomialen Algorithmus** an.
- Wir leiten einen **Approximationsalgorithmus für MAP LABELING 2** her.

NP-Vollständigkeit von MAP-LABELING 2

Karlsruher Institut für Technologie

Satz:
Problem MAP LABELING 2 ist $\mathcal{NP}$-vollständig.

NP-Vollständigkeit von MAP-LABELING 2

Karlsruher Institut für Technologie

Satz:
Problem MAP LABELING 2 ist $\mathcal{NP}$-vollständig.

Beweis:

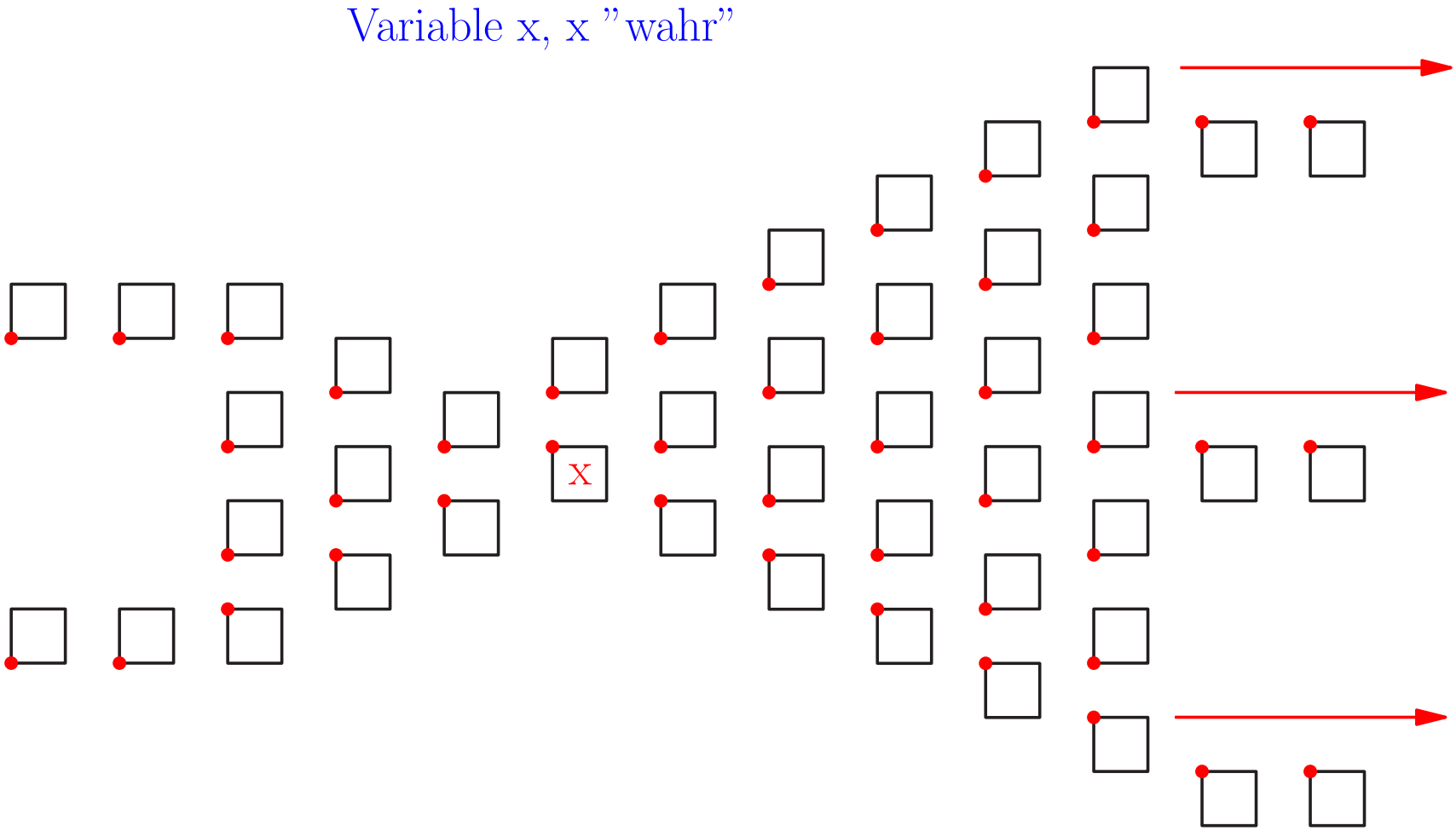
- Offensichtlich ist MAP LABELING 2 in $\mathcal{NP}$.
- Wir zeigen: 3SAT $\propto$ MAP LABELING 2

Beweisskizze

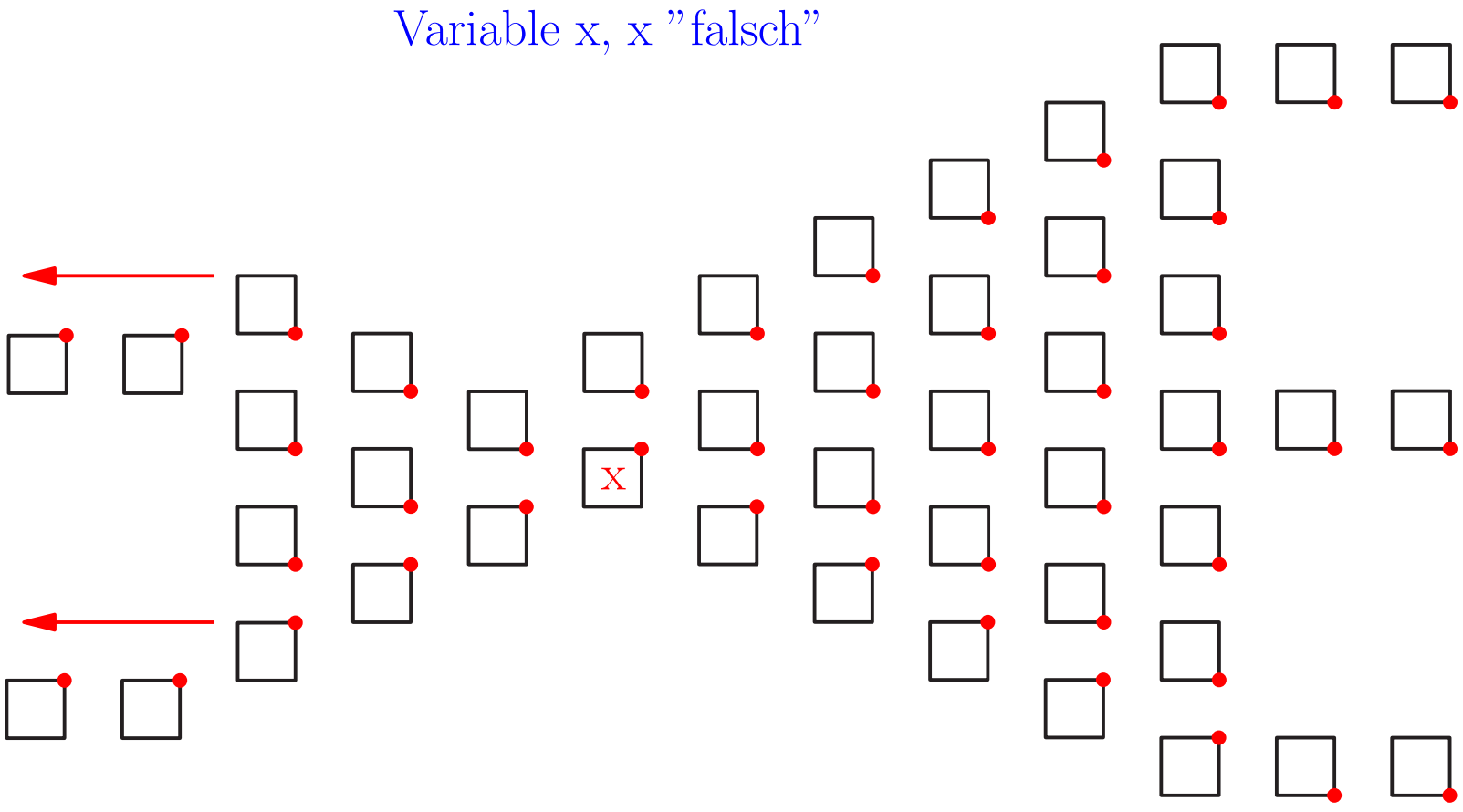
- Für jede Variable x führe
  - Punkt p_x
  - Menge von zusätzlichen Punkten

  ein
- Lage so, dass Quadrat an p_x nur links unten oder rechts unten
  - links unten: x falsch
  - rechts unten: x wahr

Beweisskizze



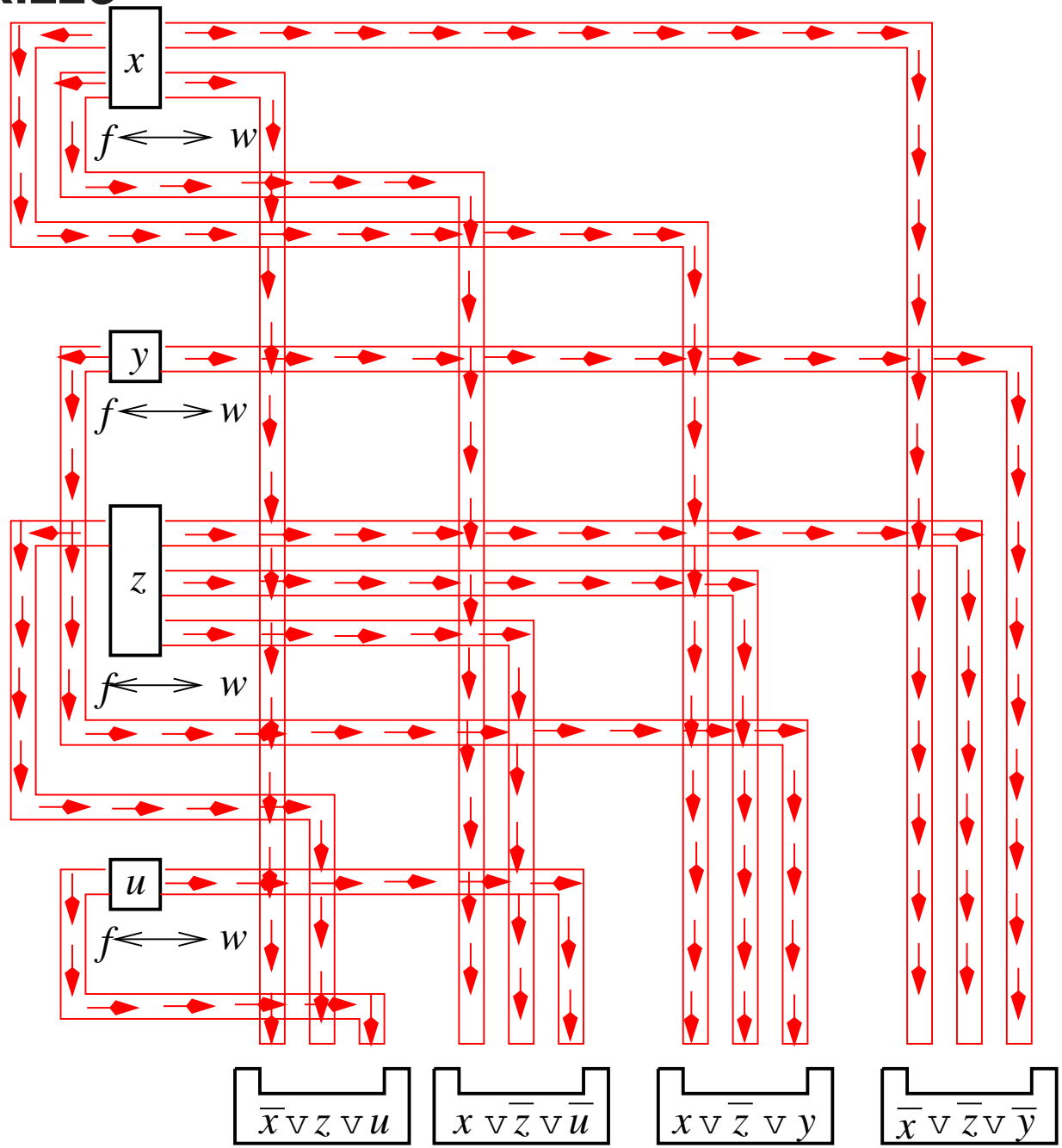
Beweisskizze



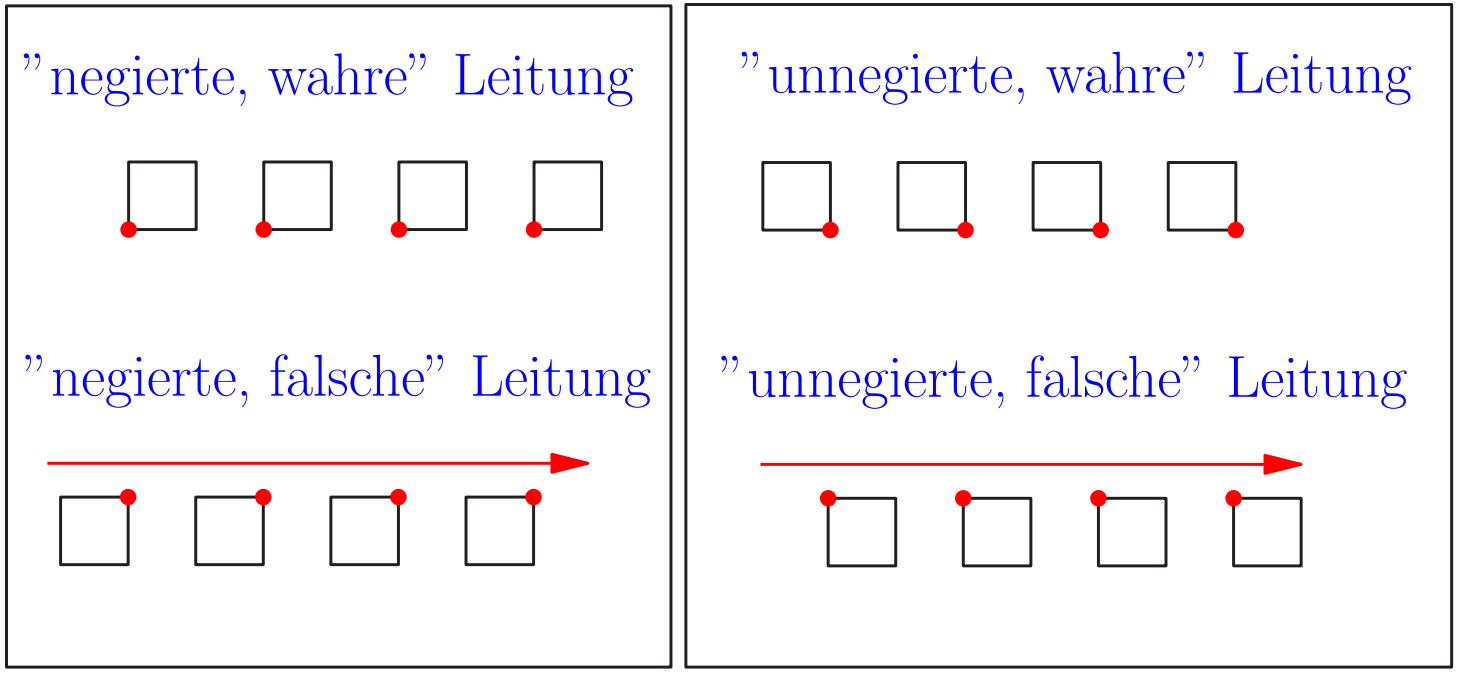
Beweisskizze

- Von p_X aus gibt es zu jeder Klausel in der x vorkommt eine „Leitung“
- Die Punkte einer „Leitung“ liegen so, dass die Lage des Quadrats von p_X die Lage aller Quadrate der Leitung festlegt.

Beweisskizze

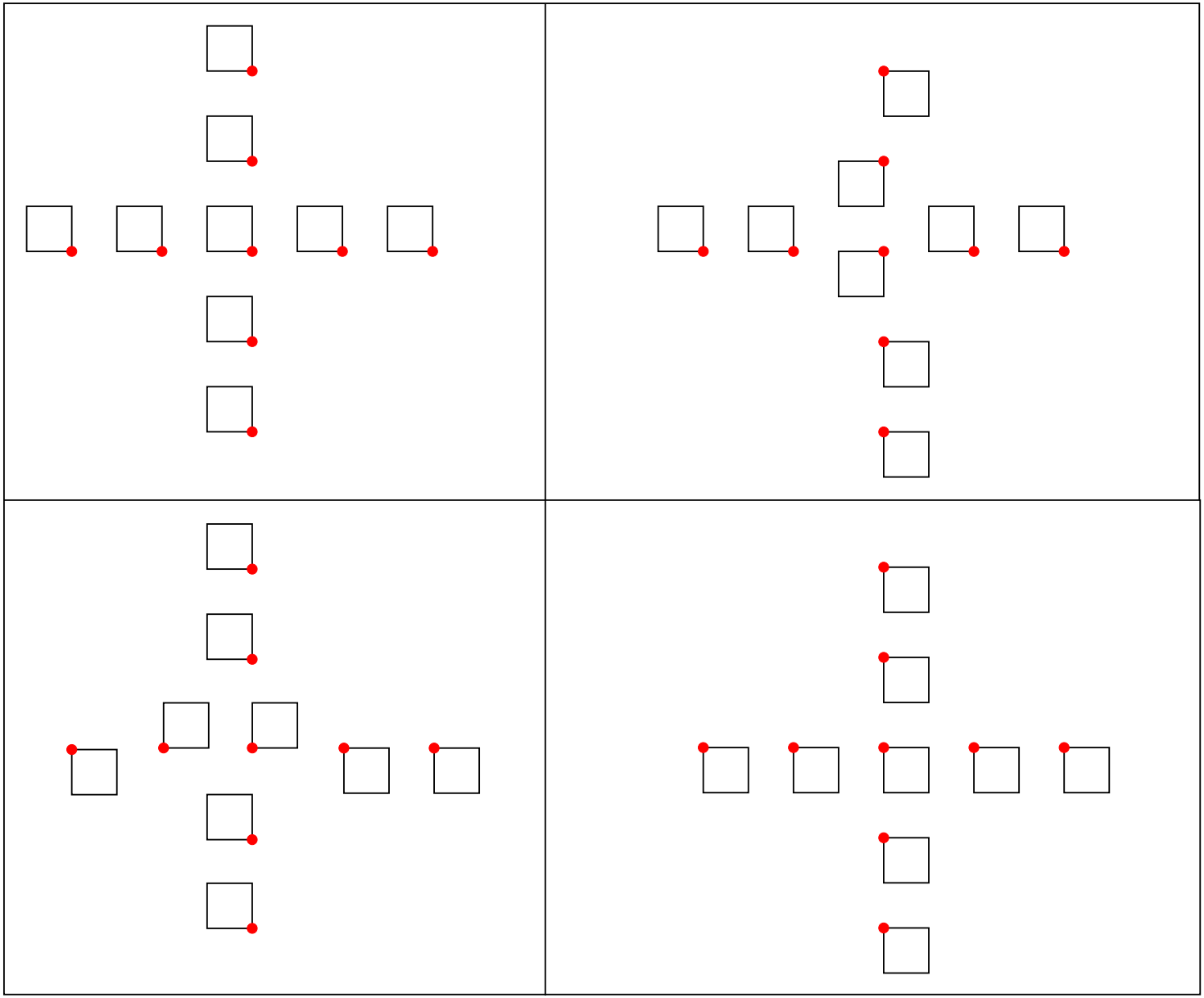


Beweisskizze



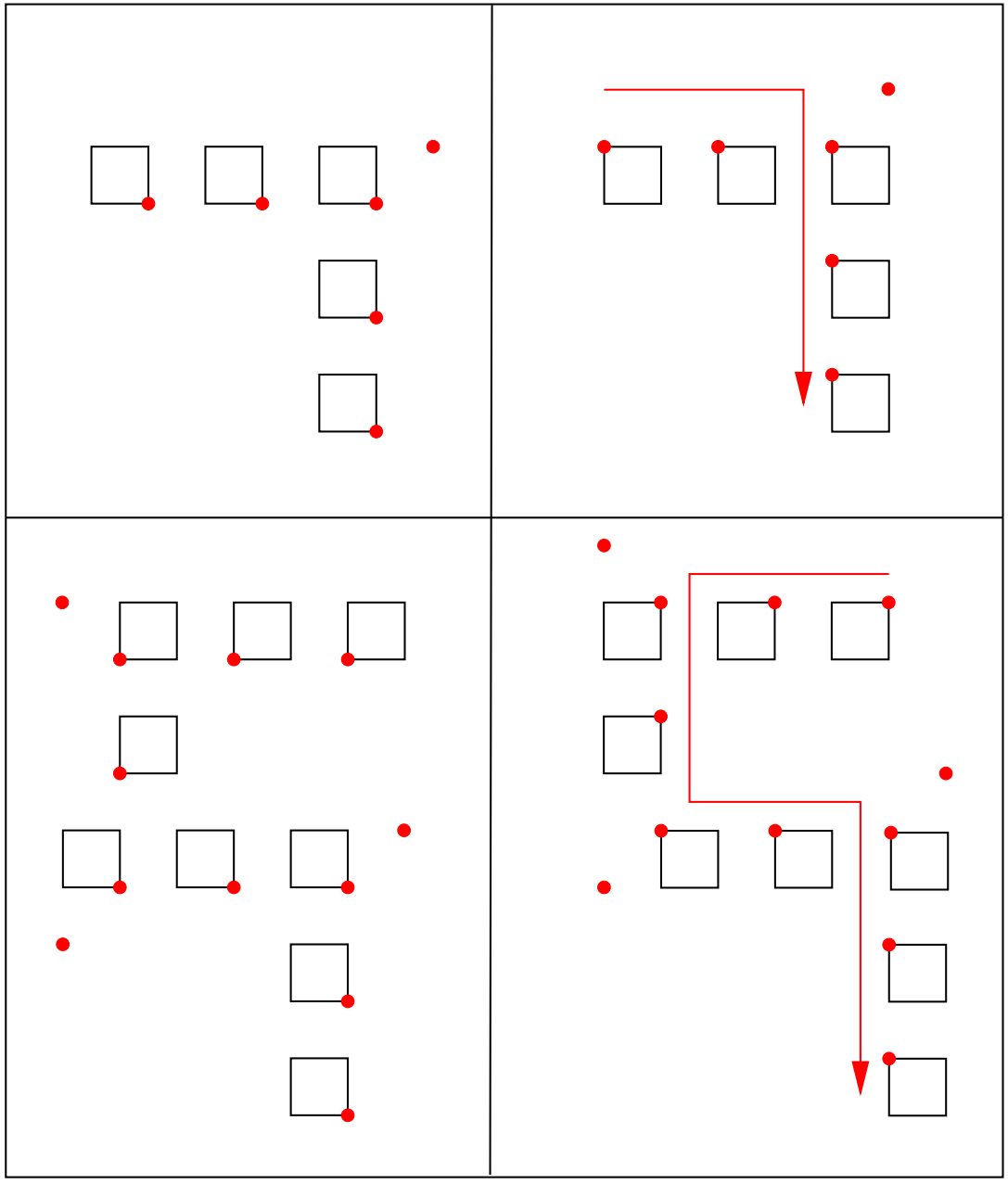
Beweisskizze

Kreuzungen



Beweisskizze

Knicke



Beweisskizze

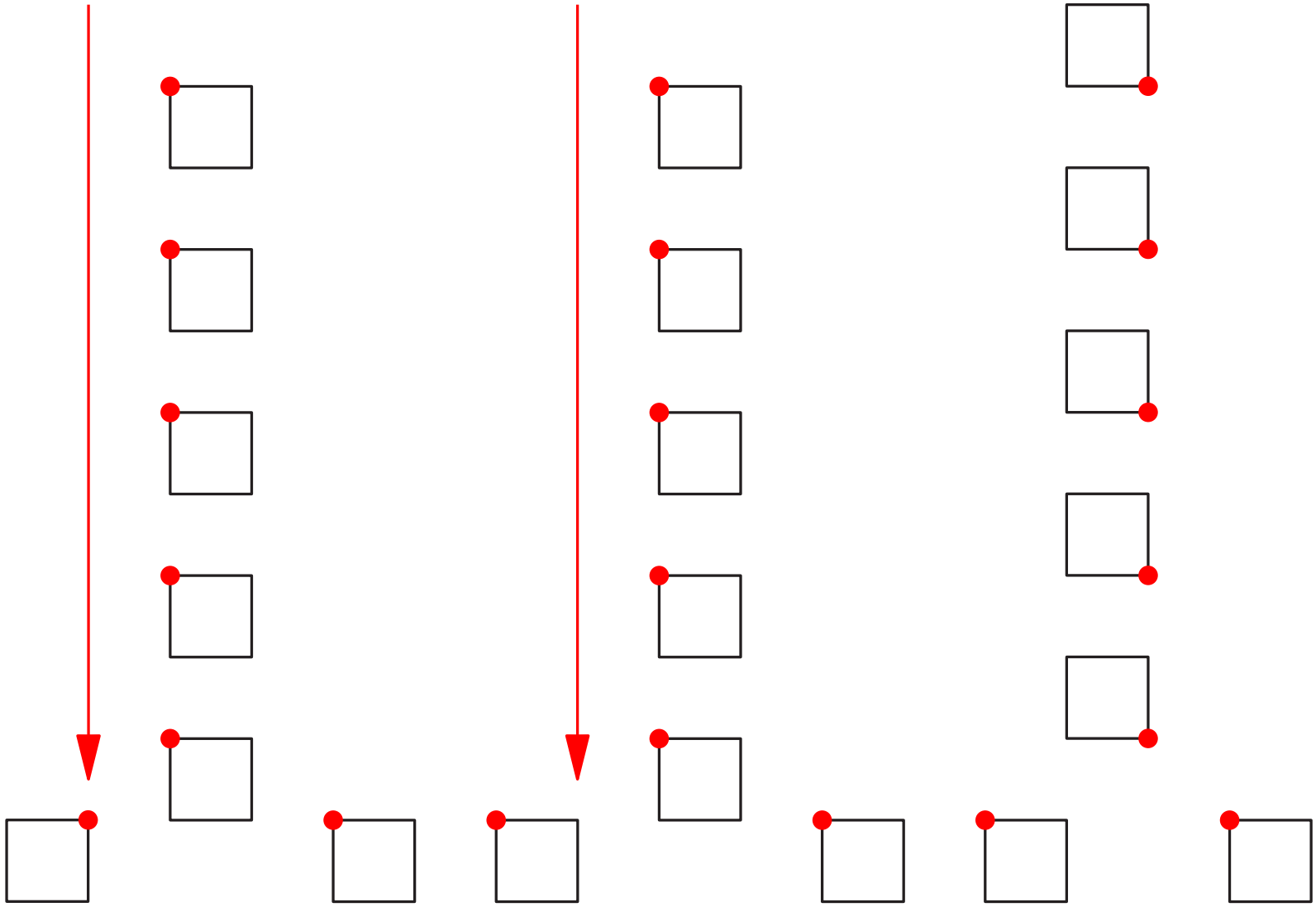
- Für jede Klausel gibt es eine Ansammlung von Punkten, deren Quadrate von den einkommenden Leitungen abhängt.

Beweisskizze

- Setzung des Quadrats an p_x induziert Position der Quadrate der Leitung.
- Position der Quadrate der Leitung induziert Position der Quadrate der Klauseln.
- Quadrat zu p_x unten links (x falsch) bewirkt „Druck“ auf Variablen in denen $\overline{x}$ vorkommt.
- Quadrat zu p_x unten rechts (x wahr) bewirkt „Druck“ auf Variablen in denen x vorkommt.
- Klauseln halten höchstens zwei Einheiten Druck aus.

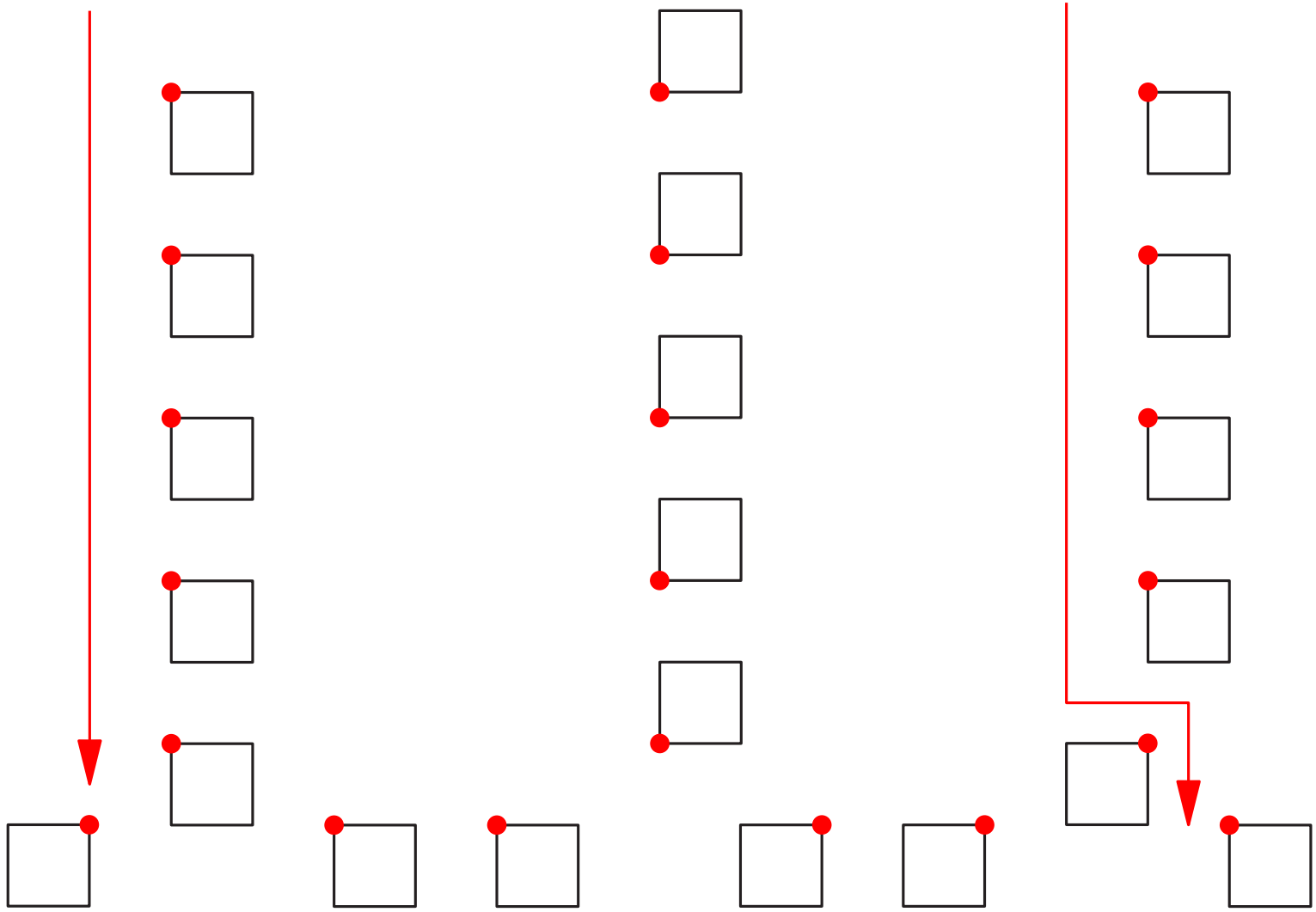
Beweisskizze

Klausel mit zwei "falschen" Literalen



Beweisskizze

Klausel mit zwei ”falschen ” Literalen



Nichtapproximierbarkeit von MAP LABELING 2

Karlsruher Institut für Technologie

Satz:
Falls $\mathcal{P} \neq \mathcal{NP}$, so gibt es keinen relativen Approximationsalgorithmus für MAP LABELING 2 Optimierungsversion mit Gütegarantie $R_{\mathcal{A}} < 2$.

Beweis:

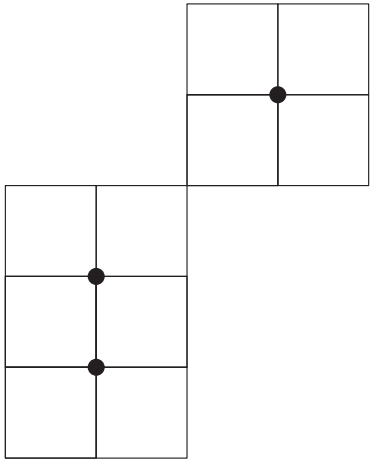
- Wir zeigen, dass mit $\mathcal{A}$ beliebige Instanzen von 3SAT entschieden werden können, im Widerspruch zu $\mathcal{P} \neq \mathcal{NP}$.
- Konstruiere, wie im letzten Beweis, eine der 3SAT-Instanz I entsprechende Instanz I' für MAP LABELING 2.

Nichtapproximierbarkeit von MAP LABELING 2

Satz:
Falls $\mathcal{P} \neq \mathcal{NP}$, so gibt es keinen relativen Approximationsalgorithmus für MAP LABELING 2 Optimierungsversion mit Gütegarantie $R_{\mathcal{A}} < 2$.

Beobachtung:

Es gibt für I' Lösung mit Kantenlänge 1
$\Leftrightarrow$ Es gibt für I' Lösung mit Kantenlänge $2 - \epsilon$ für beliebig kleines ϵ

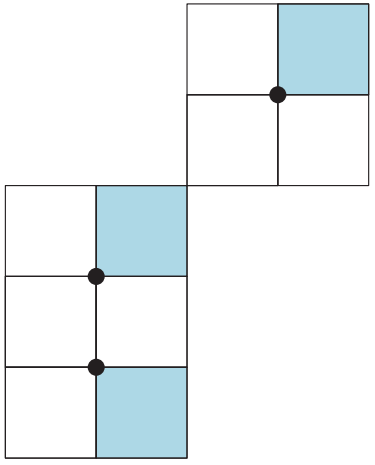


Nichtapproximierbarkeit von MAP LABELING 2

Satz:
Falls $\mathcal{P} \neq \mathcal{NP}$, so gibt es keinen relativen Approximationsalgorithmus für MAP LABELING 2 Optimierungsversion mit Gütegarantie $R_{\mathcal{A}} < 2$.

Beobachtung:

Es gibt für I' Lösung mit Kantenlänge 1
$\Leftrightarrow$ Es gibt für I' Lösung mit Kantenlänge $2 - \epsilon$ für beliebig kleines ϵ

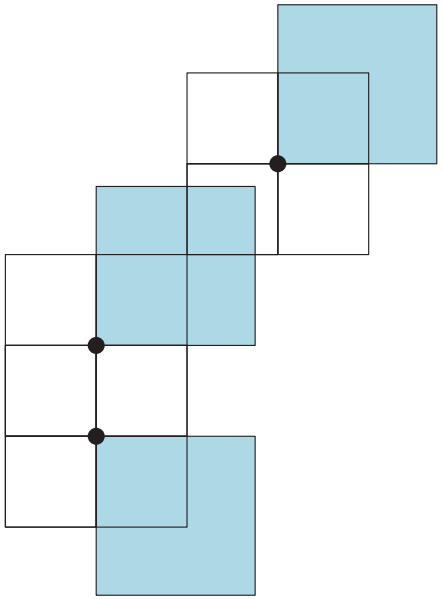


Nichtapproximierbarkeit von MAP LABELING 2

Satz:
Falls $\mathcal{P} \neq \mathcal{NP}$, so gibt es keinen relativen Approximationsalgorithmus für MAP LABELING 2 Optimierungsversion mit Gütegarantie $R_{\mathcal{A}} < 2$.

Beobachtung:

Es gibt für I' Lösung mit Kantenlänge 1
$\Leftrightarrow$ Es gibt für I' Lösung mit Kantenlänge $2 - \epsilon$ für beliebig kleines ϵ



Nichtapproximierbarkeit von MAP LABELING 2

Satz:
Falls $\mathcal{P} \neq \mathcal{NP}$, so gibt es keinen relativen Approximationsalgorithmus für MAP LABELING 2 Optimierungsversion mit Gütegarantie $R_{\mathcal{A}} < 2$.

Beobachtung:

Es gibt für I' Lösung mit Kantenlänge 1
$\Leftrightarrow$ Es gibt für I' Lösung mit Kantenlänge $2 - \epsilon$ für beliebig kleines ϵ

Damit gilt für beliebig kleines ϵ

$$opt(I') \begin{cases} > 2 - \epsilon & \text{, 3SAT-Instanz } I \text{ erfüllbar} \\ < 1 & \text{, 3SAT-Instanz } I \text{ nicht erfüllbar} \end{cases}$$

Nichtapproximierbarkeit von MAP LABELING 2

Satz:
Falls $\mathcal{P} \neq \mathcal{NP}$, so gibt es keinen relativen Approximationsalgorithmus für MAP LABELING 2 Optimierungsversion mit Gütegarantie $R_{\mathcal{A}} < 2$.

Beobachtung:

Es gibt für I' Lösung mit Kantenlänge 1
$\Leftrightarrow$ Es gibt für I' Lösung mit Kantenlänge $2 - \epsilon$ für beliebig kleines ϵ

Damit gilt für beliebig kleines ϵ

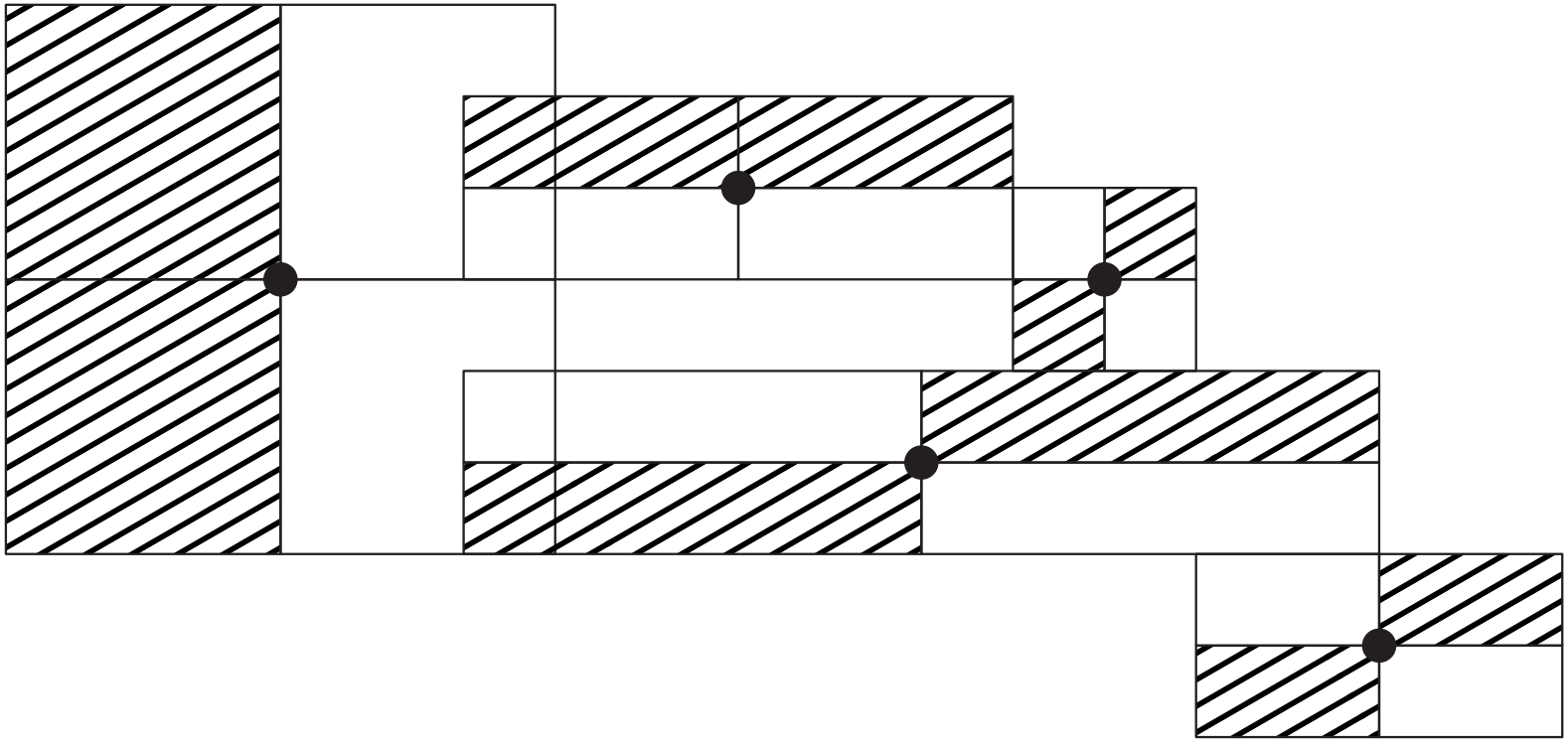
$$opt(I') \begin{cases} > 2 - \epsilon & \text{, 3SAT-Instanz } I \text{ erfüllbar} \\ < 1 & \text{, 3SAT-Instanz } I \text{ nicht erfüllbar} \end{cases}$$

Also

$$\mathcal{A}(I') \begin{cases} > 1 & \text{, 3SAT-Instanz } I \text{ erfüllbar} \\ < 1 & \text{, 3SAT-Instanz } I \text{ nicht erfüllbar} \end{cases}$$

Algo A1 für Spezialfall von MAP LABELING 1

Spezialfall MAP LABELING 1_2:
Für jeden Punkt *p* sind nur höchstens zwei der 4 möglichen Positionen zulässig.



Algo A1 für Spezialfall von MAP LABELING 1

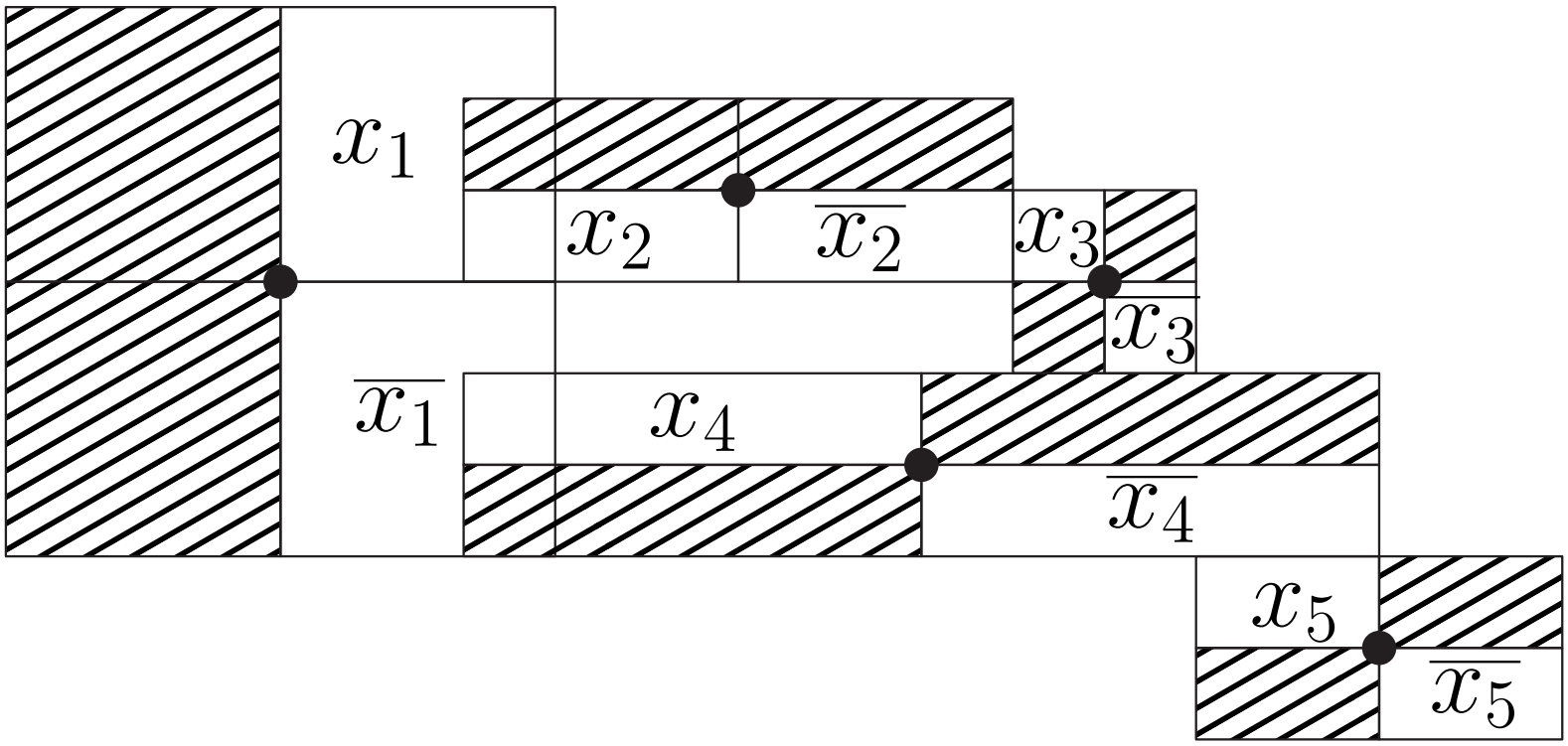
Idee: Modelliere MAP LABELING 1_2-Instanz als 2SAT-Instanz
Eingabe: $p_1, \ldots, p_n$ mit je zwei Positionen und Rechtecken $r_1, \ldots, r_n$

- Betrachte Punkte p_i als boolsche Variablen.
- Benenne die zugelassenen Positionen x_i bzw $\overline{x}_i$.
- Betrachte die Paare von Positionen verschiedener Punkte p_i und p_j, die sich überlappen. Stelle Klauseln auf:

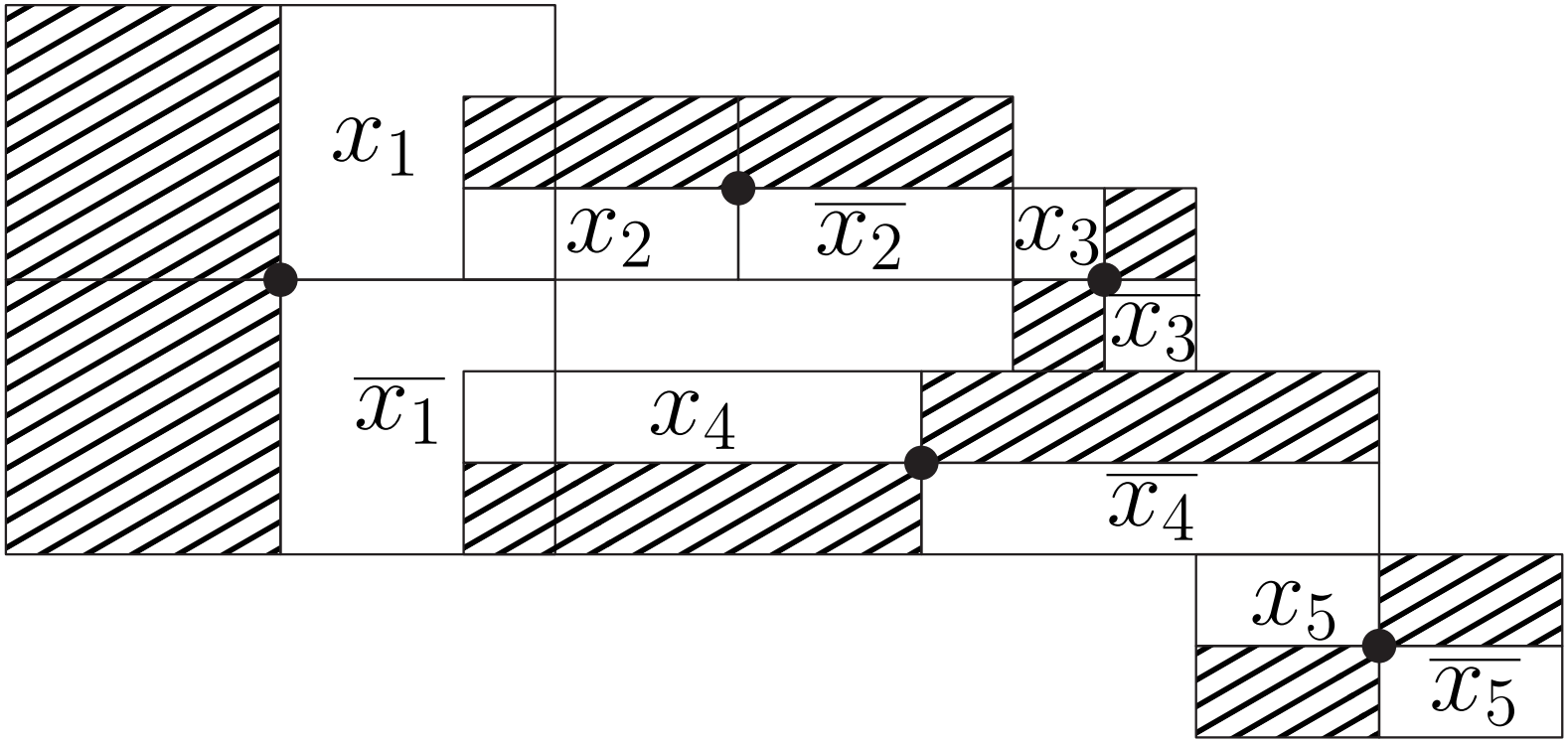
$$\begin{aligned}
\overline{(x_i \wedge x_j)} &= \overline{x_i} \vee \overline{x_j} && \text{bzw} \\
\overline{(\overline{x}_i \wedge \overline{x}_j)} &= x_i \vee x_j && \text{bzw} \\
\overline{(\overline{x}_i \wedge x_j)} &= x_i \vee \overline{x_j} && \text{bzw} \\
\overline{(x_i \wedge \overline{x}_j)} &= \overline{x_i} \vee x_j
\end{aligned}$$

- Überprüfe ob es eine erfüllende Belegung für I gibt.
- Falls ja, wähle die Positionen entsprechend.

Beispiel


x_1
x_2
$\overline{x_2}$
x_3
$\overline{x_3}$
$\overline{x_1}$
x_4
$\overline{x_4}$
x_5
$\overline{x_5}$

Beispiel

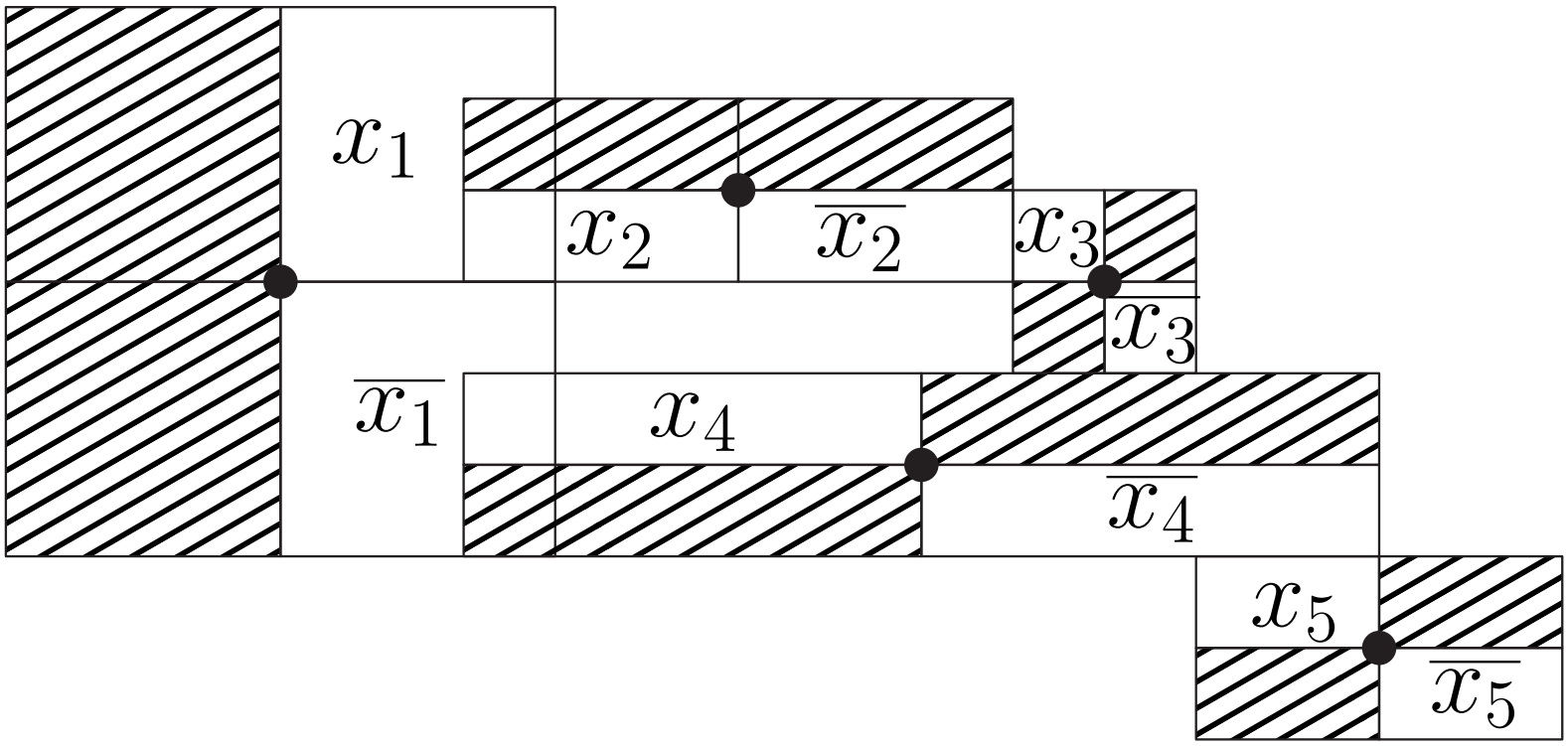


$$(x_1 \vee \overline{x}_1) \wedge (x_2 \vee \overline{x}_2) \wedge (x_3 \vee \overline{x}_3) \wedge (x_4 \vee \overline{x}_4) \wedge (x_5 \vee \overline{x}_5)$$

$$\overline{(x_1 \wedge \overline{x}_1)} \wedge \overline{(x_2 \wedge \overline{x}_2)} \wedge \overline{(x_3 \wedge \overline{x}_3)} \wedge \overline{(x_4 \wedge \overline{x}_4)} \wedge \overline{(x_5 \wedge \overline{x}_5)}$$

$$\overline{(x_1 \wedge x_2)} \wedge \overline{(\overline{x}_2 \wedge x_3)} \wedge \overline{(\overline{x}_1 \wedge x_4)} \wedge \overline{(\overline{x}_4 \wedge x_5)}$$

Beispiel

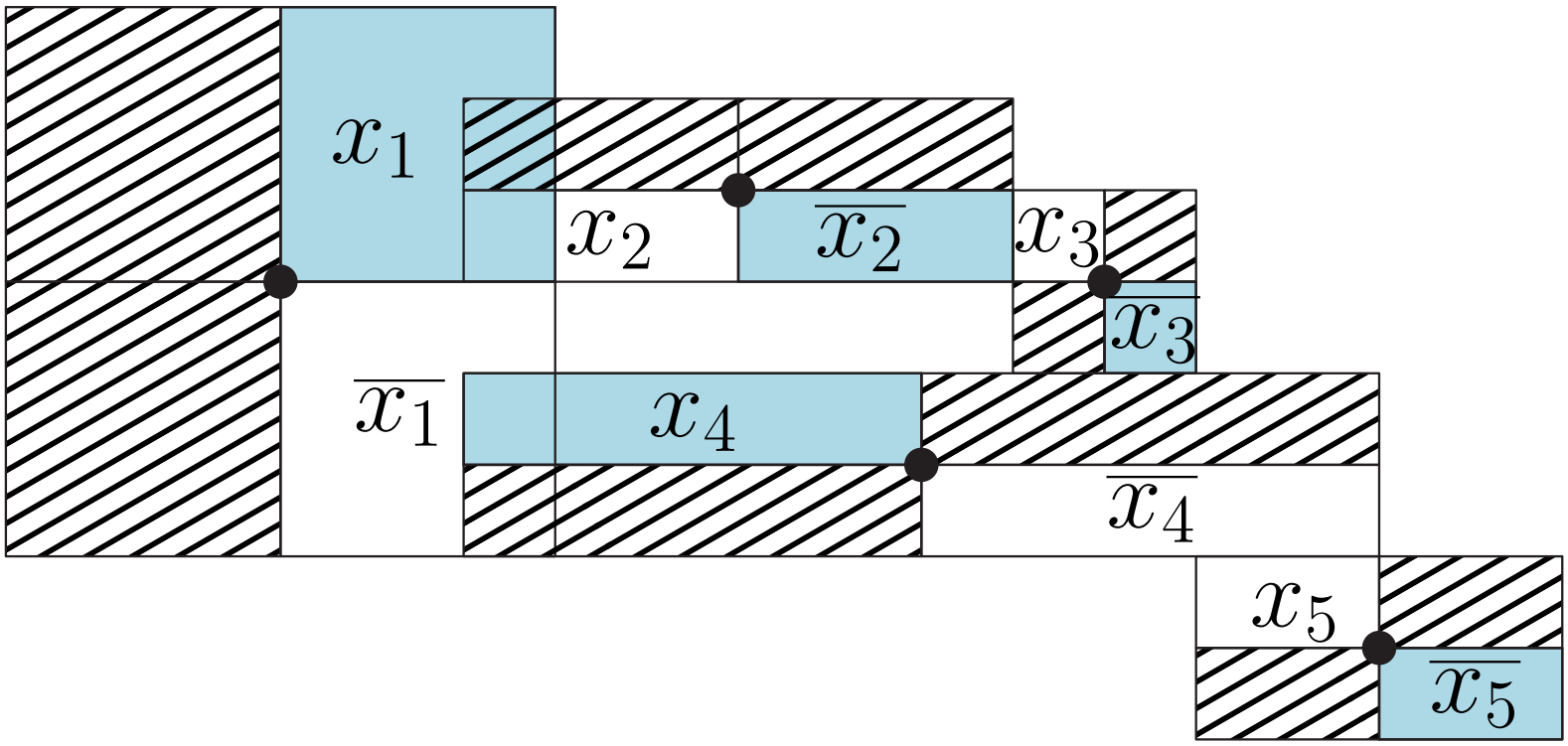


$$(x_1 \vee \overline{x}_1) \wedge (x_2 \vee \overline{x}_2) \wedge (x_3 \vee \overline{x}_3) \wedge (x_4 \vee \overline{x}_4) \wedge (x_5 \vee \overline{x}_5)$$

$$(\overline{x}_1 \vee x_1) \wedge (\overline{x}_2 \vee x_2) \wedge (\overline{x}_3 \vee x_3) \wedge (\overline{x}_4 \vee x_4) \wedge (\overline{x}_5 \vee x_5)$$

$$(\overline{x}_1 \vee \overline{x}_2) \vee (x_2 \vee \overline{x}_3) \wedge (x_1 \vee \overline{x}_4) \wedge (x_4 \vee \overline{x}_5)$$

Beispiel



$$(x_1 \vee \overline{x}_1) \wedge (x_2 \vee \overline{x}_2) \wedge (x_3 \vee \overline{x}_3) \wedge (x_4 \vee \overline{x}_4) \wedge (x_5 \vee \overline{x}_5)$$

$$(\overline{x}_1 \vee x_1) \wedge (\overline{x}_2 \vee x_2) \wedge (\overline{x}_3 \vee x_3) \wedge (\overline{x}_4 \vee x_4) \wedge (\overline{x}_5 \vee x_5)$$

$$(\overline{x}_1 \vee \overline{x}_2) \vee (x_2 \vee \overline{x}_3) \wedge (x_1 \vee \overline{x}_4) \wedge (x_4 \vee \overline{x}_5)$$

Lemma

Gegeben

- Menge $P = \{p_1, \ldots, p_n\}$ von Punkten
- Quadrat Q fester Größe
- Zu jedem p_i höchstens zwei achsenparallele Positionen für Q an denen ein Eckpunkt gerade p_i ist.

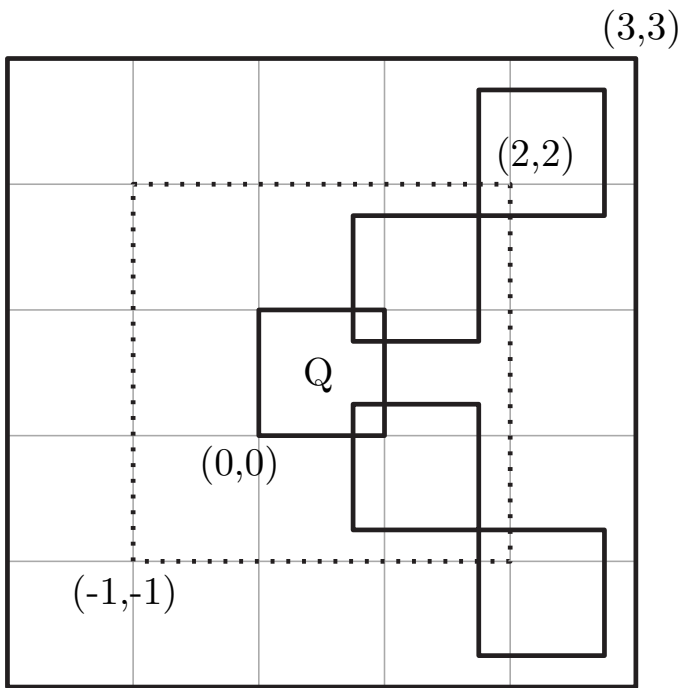
Falls für eine dieser Positionen gilt:

- Das entsprechende Quadrat wird von mehr als 24 Quadraten an anderen Positionen geschnitten,

so existiert keine Lösung für die gegebene Instanz von MAP LABELING 1.

Beweis:

- Sei Q ein achsenparalleles Quadrat
- O.B.d.A habe Q Einheitsgröße und Koordinaten $(0, 0)$ der linken unteren Ecke
- Jedes achsenparallele Quadrat gleicher Größe, dass Q schneidet, muss in $[-1, 2] \times [-1, 2]$ liegen
- Deren andere Positionen müssen im Quadrat $[-2, 3] \times [-2, 3]$ liegen.
- Dieses Quadrat hat Größe 25.



Approximationsalgo für MAP LABELING 2

Karlsruher Institut für Technologie

Idee:

- Starte mit unendlich kleinen, gleichgroßen Quadraten an allen 4 Positionen für jeden Punkt
- Iterativ vergrößere diese gleichzeitig
- Wenn Quadrate aneinanderstoßen entferne diese so, dass eine zulässige Lösung erhalten bleibt

Approximationsalgo für MAP LABELING 2

Karlsruher Institut für Technologie

Idee:

- Starte mit unendlich kleinen, gleichgroßen Quadraten an allen 4 Positionen für jeden Punkt
- Iterativ vergrößere diese gleichzeitig
- Wenn Quadrate aneinanderstoßen entferne diese so, dass eine zulässige Lösung erhalten bleibt

Plan:

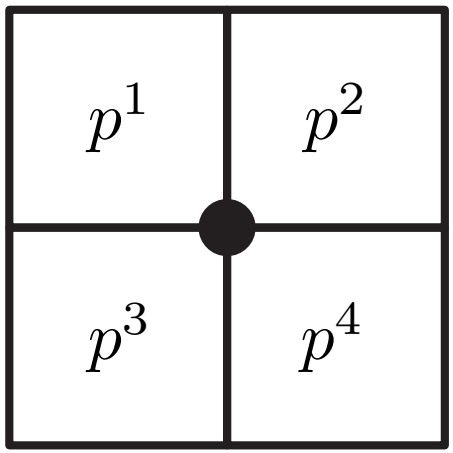
- Lösche Positionen so, dass nur zwei Positionen pro Punkt bleiben
- Diese können mit Algorithmus A1 festgelegt werden

Benennung

Für Punkt p sei p_i eine achsenparallele Posistion eines Quadrates mit Eckpunkt p, wobei

- p^1 bedeutet p rechts unten
- p^2 bedeutet p links unten
- p^3 bedeutet p rechts oben
- p^4 bedeutet p links oben

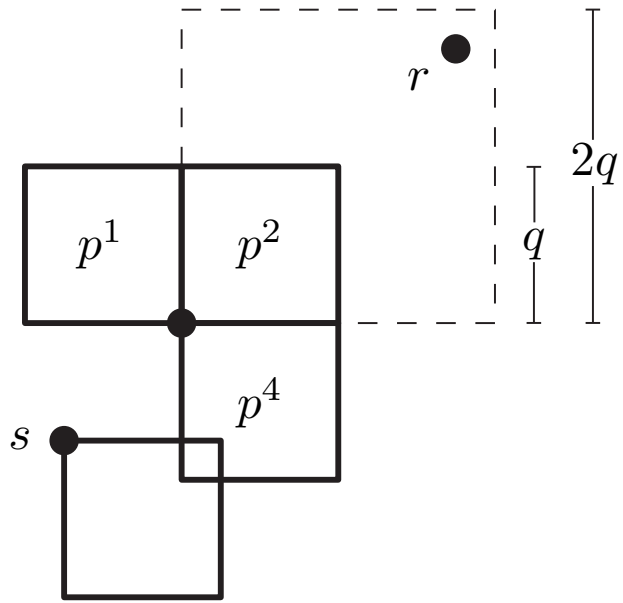
Zu Zahl q bezeichne qp^i entsprechendes Quadrat mit Seitenlänge q.



Benennung - 2

Für Menge P von Punkten, $p \in P$ und Zahl q, heißt Position p^j

- **q-erledigt,** falls $(2q)p^j$ einen anderen Punkt aus P enthält.
- **q-unerledigt,** falls
  - p^j nicht q-erledigt und
  - qp^j hat einen nichtleeren Schnitt mit einem qr^j, wobei r^j eine andere nicht q-erledigte Position ist $(r \in P)$.
- **q-aktiv,** falls p^j weder q-erledigt, noch q-unerledigt ist.



p^1 ist q-aktiv
p^2 ist q-erledigt
p^4 ist q-unerledigt

Anmerkung

A2 ist so entworfen, dass folgende Invarianten gelten:

- **Invariante 1:** Nur q-erledigte Quadrate sind nach Zeitpunkt q entfernt.
- **Invariante 2:** Es kann zu jedem Zeitpunkt q in polynomieller Zeit eine Lösung mit Quadraten mit Seitenlänge q konstruiert werden.

Vorbemerkung

- Im folgenden Algorithmus A2 wird für Schritt 2.2 vorausgesetzt, dass zu jedem aktuellen q höchstens zwei p^i pro $p \in P$ unerledigt sind.
- Wir werden später beweisen, dass diese Annahme zulässig ist.

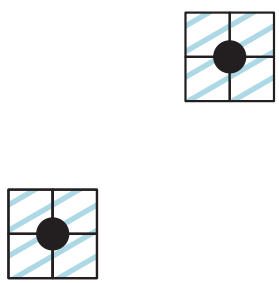
Der Algorithmus

Eingabe: $P := \{p_1, \ldots, p_n\}$

1. Starte mit Seitenlänge $q = 0$ und allen 4 Positionen p^i für jedes $p \in P$.
2. Für wachsendes q führe aus
   1. Entferne alle q-erledigten Positionen.
   2. Für die Menge der Punkte $P' \subseteq P$, die keine q-aktive Position mehr haben führe Algo A1 mit den q-unerledigten Positionen aus.
   3. Stoppe, falls ein Punkt keine (q-unerledigte oder q-aktive Position mehr hat, oder falls A1 in Schritt 2.2 Antwort „nein“ gibt.
3. Verkleinere den Stop-Wert q um einen beliebig kleineren Wert auf q_S.
4. Konstruiere eine Lösung mit Seitenlänge q_S indem
   1. für jeden Punkt, der noch q_S aktive Positionen hat eine von diesen ausgewählt wird.
   2. für die restlichen Punkte die Lösung von A1 genommen wird.

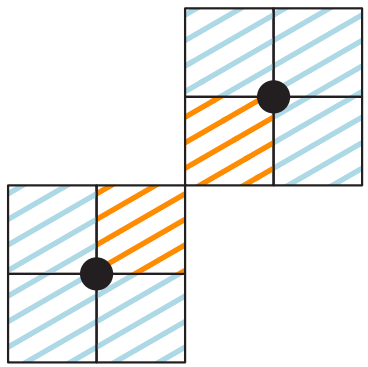
Bemerkung

- Es kann sein, dass A2 niemals stoppt, wenn $|P| \leq 4$ und alle Punkte auf dem Rand der konvexen Hülle von P liegen.
- Dies kann leicht vor der Ausführung von A2 erkannt werden.



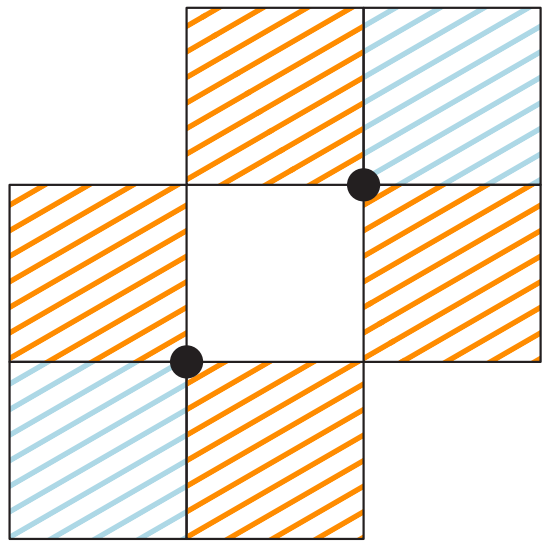
Bemerkung

- Es kann sein, dass A2 niemals stoppt, wenn $|P| \leq 4$ und alle Punkte auf dem Rand der konvexen Hülle von P liegen.
- Dies kann leicht vor der Ausführung von A2 erkannt werden.



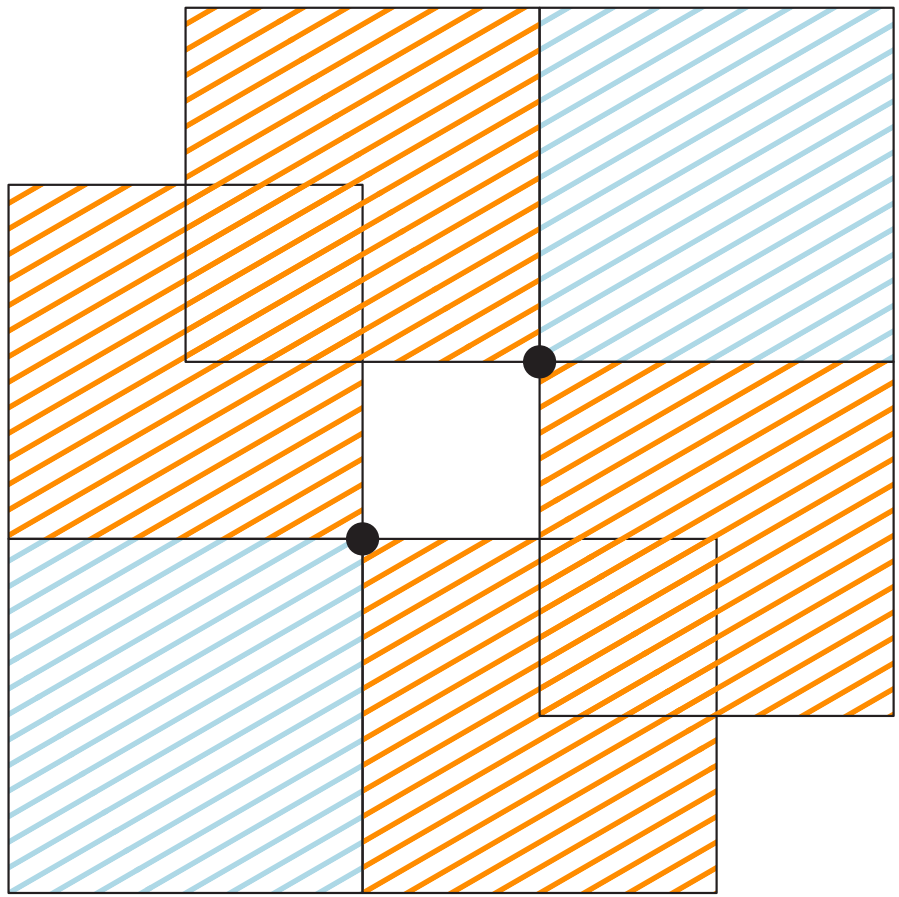
Bemerkung

- Es kann sein, dass A2 niemals stoppt, wenn $|P| \leq 4$ und alle Punkte auf dem Rand der konvexen Hülle von P liegen.
- Dies kann leicht vor der Ausführung von A2 erkannt werden.



Bemerkung

- Es kann sein, dass A2 niemals stoppt, wenn $|P| \leq 4$ und alle Punkte auf dem Rand der konvexen Hülle von P liegen.
- Dies kann leicht vor der Ausführung von A2 erkannt werden.



Invariante 2

Karlsruher Institut für Technologie

Invariante 2
Es kann zu jedem Zeitpunkt q in polynomieller Zeit eine Lösung mit Quadraten mit Seitenlänge q konstruiert werden.

Invariante 2

Invariante 2
Es kann zu jedem Zeitpunkt q in polynomieller Zeit eine Lösung mit Quadraten mit Seitenlänge q konstruiert werden.

- Ein Quadrat an einer q-aktiven schneidet kein anderes Quadrat mit Seitenlänge q.
- D.h. für alle Punkte mit q-aktiven Positionen kann eine Lösung leicht angegeben werden.
- Für alle anderen Punkte gibt es nach Abbruchbedingung aus 2.3. jeweils mindestens eine q-unerledigte Position.

Lemma

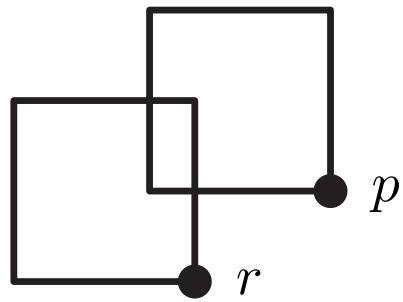
Zu jedem Zeitpunkt q des Algorithmus A2 gibt es für jeden Punkt höchstens zwei q-unerledigte Positionen.

Lemma

Zu jedem Zeitpunkt q des Algorithmus A2 gibt es für jeden Punkt höchstens zwei q-unerledigte Positionen.

Beweis:

- Angenommen zu p sei p^1 q-unerledigt.
- Dann existiert ein Punkt r und Position r^j mit qp^1 und qr^j haben nichtleeren Schnitt
- Nach Definition: p^1 und r^j nicht q-erledigt
- Also enthalten $2qp^1$ und $2qr^j$ keinen anderen Punkt.
- Dann liegt aber r in p^3 oder p^2.



Lemma

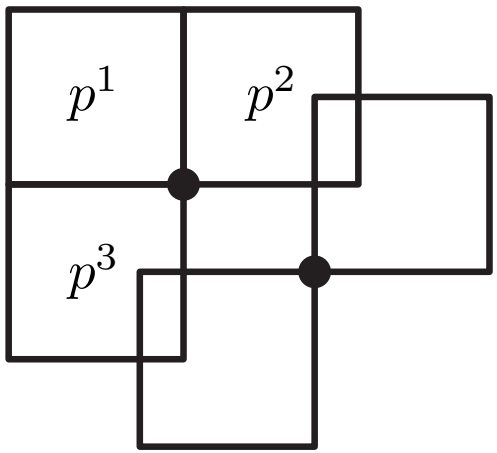
Zu jedem Zeitpunkt q des Algorithmus A2 gibt es für jeden Punkt höchstens zwei q-unerledigte Positionen.

- Beachte, dass r^1 ebenfalls q-unerledigt sein muss.
- Die Fälle p^2, p^3, p^4 sind analog.

Lemma

Zu jedem Zeitpunkt q des Algorithmus A2 gibt es für jeden Punkt höchstens zwei q-unerledigte Positionen.

- Angenommen es gäbe nun drei Positionen zu p, o.B.d.A p^1, p^2, p^3, die q-unerledigt sind.
- Dann müssen deren q-unerledigte Partner in Quadraten zu q-unerledigten Positionen von p liegen.
- Dies ist höchstens für die Partner zu p^2 und p^3 möglich.



Satz

Satz:
Algorithmus A2 hat relative Gütegarantie 2.

Satz

Satz:
Algorithmus A2 hat relative Gütegarantie 2.

- Betrachte optimale Lösung mit Quadraten der Seitenlänge OPT.
- Durch schrumpfen der Seitenlänge kann eine Lösung der Seitenlänge $OPT/2$ konstruiert werden.
- Keine der darin enthaltenen Positionen wird q-erledigt, solange $q \leq OPT/2$.
- Invariante 1: A2 entfernt keine Position, die für eine Lösung der Größe $OPT/2$ benötigt wird.

Laufzeitanalyse

- Diskretisiere das Anwachsen von q.
- Es sind nur Werte relevant, an denen sich für eine Position der Zustand ändert.

Laufzeitanalyse

- Diskretisiere das Anwachsen von q.
- Es sind nur Werte relevant, an denen sich für eine Position der Zustand ändert.

Position wird q-erledigt.

- Eine Position kann nur einmal q-erledigt werden.
- Für jede Position hängt das entsprechende q nur von der Lage der anderen Punkte ab.
- Es gibt maximal $4n$ Werte von q, an denen eine der Positionen q-erledigt wird.
- Diese können mit 4 sogenannten Plane-Sweeps in $O(n \log n)$ berechnet werden.

Laufzeitanalyse

- Diskretisiere das Anwachsen von q.
- Es sind nur Werte relevant, an denen sich für eine Position der Zustand ändert.

Position wird q-unerledigt.

- Situation komplizierter
- Frage für p^j und q:
  Wieviele andere q-unerledigte Quadrate gibt es, die qp^j überlappen.

Lemma

Sei P eine Punktmenge und Q ein Quadrat mit Seitenlänge q.
Falls $\exists$ mehr als acht verschiedene Punkte $p \in P$ mit Positionen p^i,

- die q-unerledigt sind
- für die qp^i Quadrat q scheidet

so existiert keine Lösung der Größe q zu P für MAPLABELING 2.

Lemma

Sei P eine Punktmenge und Q ein Quadrat mit Seitenlänge q.
Falls $\exists$ mehr als acht verschiedene Punkte $p \in P$ mit Positionen p^i,

- die q-unerledigt sind
- für die qp^i Quadrat q scheidet

so existiert keine Lösung der Größe q zu P für MAPLABELING 2.

Beweis:

- O.B.d.A sei
  - Position von Q an $p \in P$ genau p^2
  - Koordinate von p sei $(0, 0)$.
- Dann muss eine q-unerledigte Position, die Q schneidet,
  - Eckpunkt r in qp^1 oder qp^4 haben
  - vom Typ r^1 oder r^4 sein.

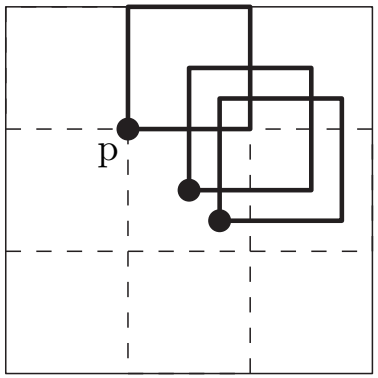
Lemma

Sei P eine Punktmenge und Q ein Quadrat mit Seitenlänge q.
Falls $\exists$ mehr als acht verschiedene Punkte $p \in P$ mit Positionen p^i,

- die q-unerledigt sind
- für die qp^i Quadrat q scheidet

so existiert keine Lösung der Größe q zu P für MAPLABELING 2.

- Die Quadrate r^1, r^2, r^3, r^4 in qp^4 liegen in dem $3q \times 3q$ Quadrat, das von $(-1, -2)$ und $(2, 1)$ aufgespannt wird.
- Analog liegen die 4 Quadrate zu einem solchen Punkt r in qp^1 in dem $3q \times 3q$ Quadrat zwischen $(-2, -1)$ und $(1, 2)$.



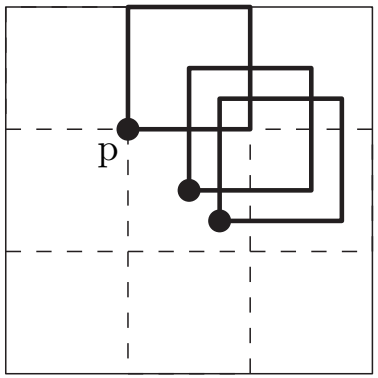
Lemma

Sei P eine Punktmenge und Q ein Quadrat mit Seitenlänge q.
Falls $\exists$ mehr als acht verschiedene Punkte $p \in P$ mit Positionen p^i,

- die q-unerledigt sind
- für die qp^i Quadrat q scheidet

so existiert keine Lösung der Größe q zu P für MAPLABELING 2.

- In ein solches Quadrat passen jeweils höchstens 4 weitere abgeschlossene Quadrate der Seitenlänge q.
- Es können also höchstes 4 Punkte aus P in qp^1 und 4 Punkte aus P in qp^4 liegen, damit eine Lösung möglich ist.



Folgerung

- Für beliebies $p \in P$ und q gibt es zu Position p^i höchstens 8 Punkte, die q'-unerledigter Partner zu $q'p^i$ mit $q' \leq q$ sein können.
- Dazu brauchen nur die jeweils 4 nächstgelegenen Punkte in den Positionen p^i berechnet werden.
- Dies ist in Zeit $O(n \log n)$ für alle $p \in P$ möglich.

Beachte:

- Zu p^i kann ein r^j existieren, das
  - q-unerledigter Partner von qp^i ist, aber
  - q'-erledigt ist für $q' \geq q$
  - damit $q'p^i$ wieder q'-aktiv ist.
- Trotzdem sind insgesamt nur 8 Punkte relevant.
- Diese können wieder in $O(n \log n)$ berechnet werden.

Implementation von Anwachsen von q

Die q-Werte, für die irgendeine Position eventuell ihren Status wechselt, werden

- vorabberechnet,
- sortiert,
- gemäß ihrer Sortierung durchlaufen.

Laufzeit

- Es werden $O(n)$ q-Werte durchlaufen.
- Also wird $O(n)$ mal Algorithmus A1 aufgerufen.
- Dies führt zu einer Laufzeit in $O(n^2)$.

Laufzeit

- Es ist möglich Algo A2 mit Laufzeit in $O(n \log n)$ zu implementieren
- Dazu rufe A1 nur $O(\log n)$ mal auf:
  - Führe A2 vorab ohne Aufruf von A1 aus
  - Stoppe mit Wert q wenn ein Punkt nur noch q-erledigte Positionen hat
  - Mit Binärer Suche auf den q-Werten wird ermittelt, wann A2 gestoppt hätte.
- Es kann gezeigt werden, dass MAPLABELING 2 in $\Theta(n \log n)$ liegt.