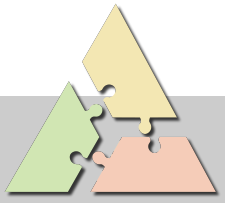
Vorstellung der bisherigen Forschungsgebiete

Gebietsüberwachung, Funktionsapproximation

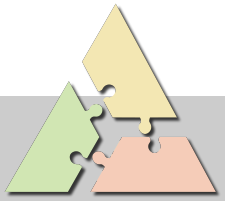
Dennis Schieferdecker (schiefer@ira.uka.de)

ITI Sanders, Universität Karlsruhe (TH)

29.09.2008



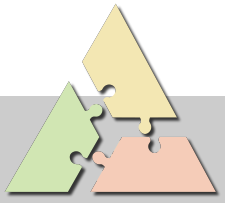
Inhaltsverzeichnis



– I –

Gebietsüberwachung

energie-effizient, sensorbasiert

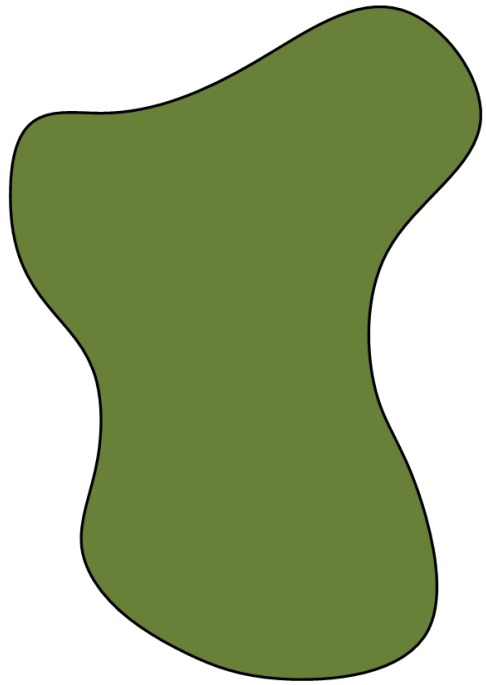


Überblick

Problemformulierung - 1

Überwachung eines Gebietes

- Gebiet F soll permanent überwacht werden (z.B. Temperatur, unerlaubtes Betreten, ...)
- Ausbringung von N Sensorknoten
  - ↪ N wesentlich größer als nötig für vollständige Überwachung von F
- Aktiviere zu jedem Zeitpunkt nur soviele Sensoren wie nötig
  - ↪ Überwachungsdauer T maximieren

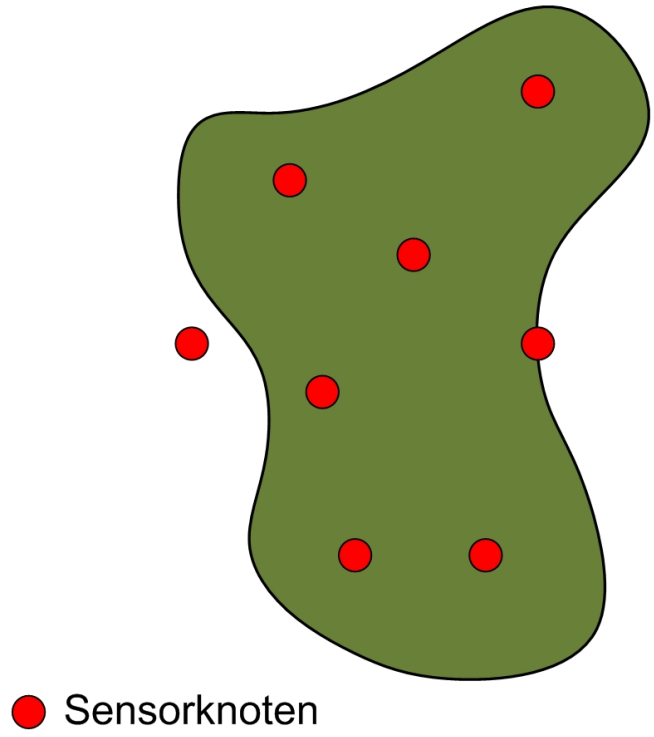


Überblick

Problemformulierung - 1

Überwachung eines Gebietes

- Gebiet F soll permanent überwacht werden (z.B. Temperatur, unerlaubtes Betreten, ...)
- Ausbringung von N Sensorknoten
  - ↪ N wesentlich größer als nötig für vollständige Überwachung von F
- Aktiviere zu jedem Zeitpunkt nur soviele Sensoren wie nötig
  - ↪ Überwachungsdauer T maximieren

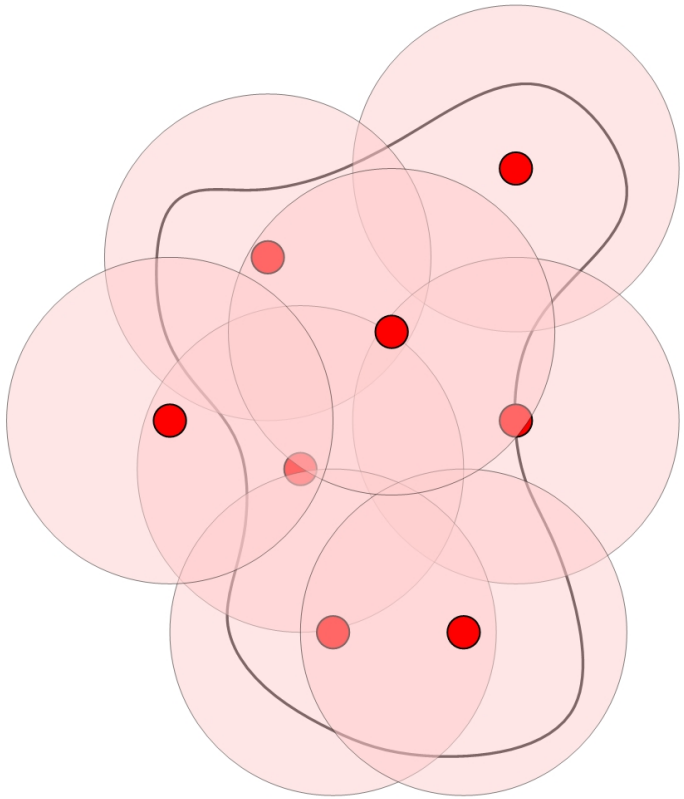


Überblick

Problemformulierung - 1

Überwachung eines Gebietes

- Gebiet F soll permanent überwacht werden (z.B. Temperatur, unerlaubtes Betreten, ...)
- Ausbringung von N Sensorknoten
  - ↪ N wesentlich größer als nötig für vollständige Überwachung von F
- Aktiviere zu jedem Zeitpunkt nur soviele Sensoren wie nötig
  - ↪ Überwachungsdauer T maximieren

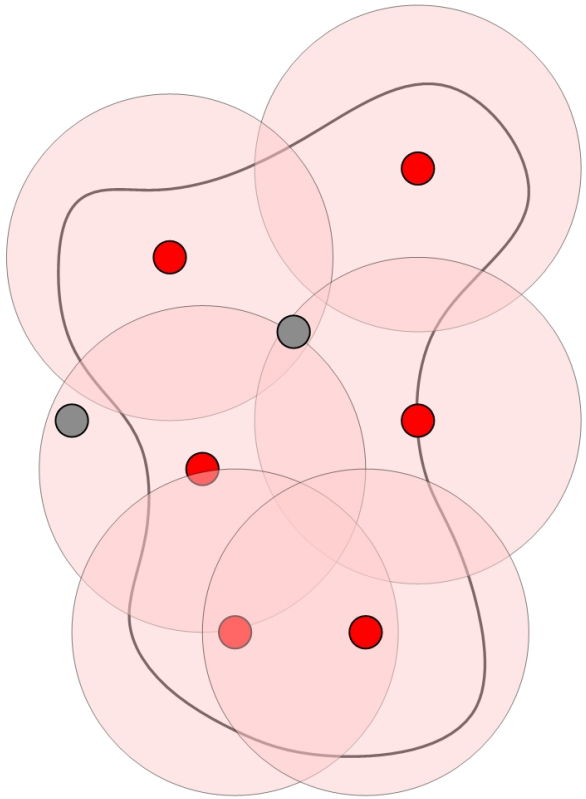


Überblick

Problemformulierung - 1

Überwachung eines Gebietes

- Gebiet F soll permanent überwacht werden (z.B. Temperatur, unerlaubtes Betreten, ...)
- Ausbringung von N Sensorknoten
  - ↪ N wesentlich größer als nötig für vollständige Überwachung von F
- Aktiviere zu jedem Zeitpunkt nur soviele Sensoren wie nötig
  - ↪ Überwachungsdauer T maximieren



Überblick

Problemformulierung - 2

Problembezeichnung

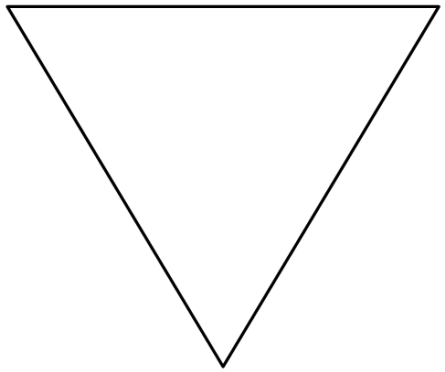
Scheduling of nodes for **L**ifetime maximization of area **C**overage **(SLC)**

Beispiel

- ≫ Sensoren A, B, C mit Kapazität 1
- ≫ 3 mögliche Überdeckungen: AB, BC, AC

(a) Lasse AB für t=1 aktiv
↪ $T = 1.0$, keine weitere Überdeckung

(b) Lasse AB für t=0.5 aktiv,
danach BC für t=0.5 aktiv,
danach CA für t=0.5 aktiv
↪ $T = 1.5$, 50% längere Laufzeit



Überblick

Problemformulierung - 2

Problembezeichnung

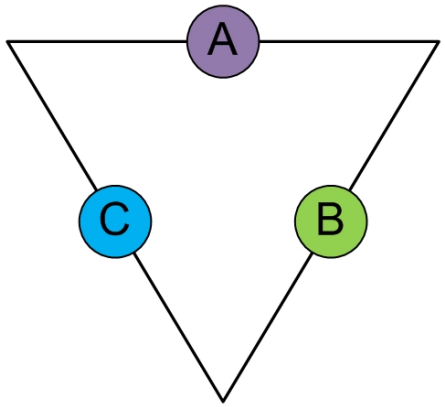
Scheduling of nodes for **L**ifetime maximization of area **C**overage **(SLC)**

Beispiel

- ≫ Sensoren A, B, C mit Kapazität 1
- ≫ 3 mögliche Überdeckungen: AB, BC, AC

(a) Lasse AB für t=1 aktiv
↪ $T = 1.0$, keine weitere Überdeckung

(b) Lasse AB für t=0.5 aktiv,
danach BC für t=0.5 aktiv,
danach CA für t=0.5 aktiv
↪ $T = 1.5$, 50% längere Laufzeit



Überblick

Problemformulierung - 2

Problembezeichnung

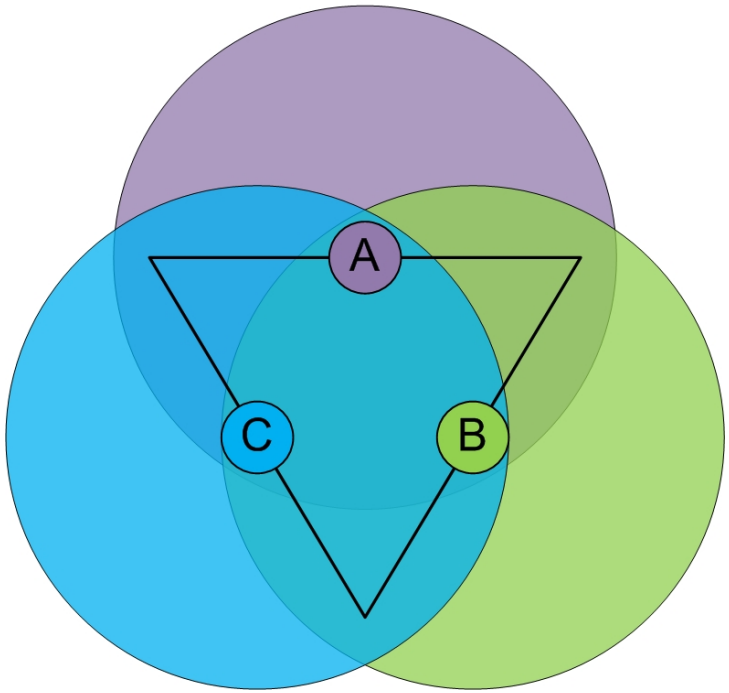
Scheduling of nodes for **L**ifetime maximization of area **C**overage **(SLC)**

Beispiel

- Sensoren A, B, C mit Kapazität 1
- 3 mögliche Überdeckungen: AB, BC, AC

(a) Lasse AB für t=1 aktiv
↪ $T = 1.0$, keine weitere Überdeckung

(b) Lasse AB für t=0.5 aktiv,
danach BC für t=0.5 aktiv,
danach CA für t=0.5 aktiv
↪ $T = 1.5$, 50% längere Laufzeit



Überblick

Problemformulierung - 2

Problembezeichnung

Scheduling of nodes for **L**ifetime maximization of area **C**overage **(SLC)**

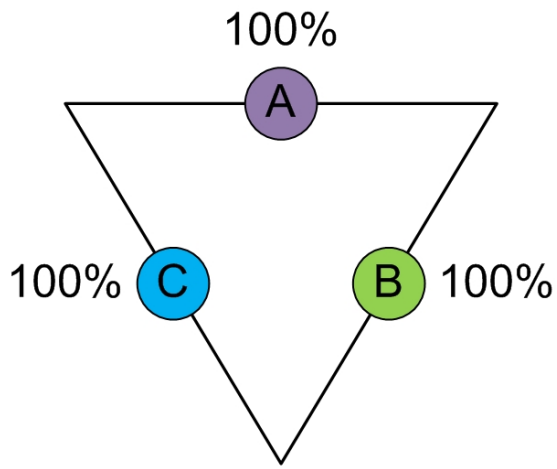
Beispiel

- ≫ Sensoren A, B, C mit Kapazität 1
- ≫ 3 mögliche Überdeckungen: AB, BC, AC

(a) Lasse AB für t=1 aktiv
↪ $T = 1.0$, keine weitere Überdeckung

(b) Lasse AB für t=0.5 aktiv,
danach BC für t=0.5 aktiv,
danach CA für t=0.5 aktiv
↪ $T = 1.5$, 50% längere Laufzeit

(a) triviale Lösung



Überblick

Problemformulierung - 2

Problembezeichnung

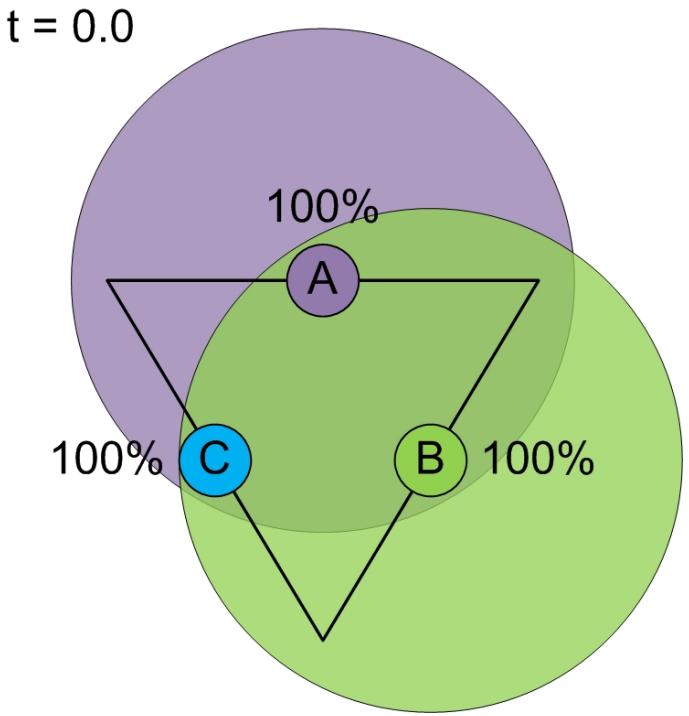
Scheduling of nodes for **L**ifetime maximization of area **C**overage **(SLC)**

Beispiel

- Sensoren A, B, C mit Kapazität 1
- 3 mögliche Überdeckungen: AB, BC, AC

(a) Lasse AB für t=1 aktiv
↪ $T = 1.0$, keine weitere Überdeckung

(b) Lasse AB für t=0.5 aktiv,
danach BC für t=0.5 aktiv,
danach CA für t=0.5 aktiv
↪ $T = 1.5$, 50% längere Laufzeit



Überblick

Problemformulierung - 2

Problembezeichnung

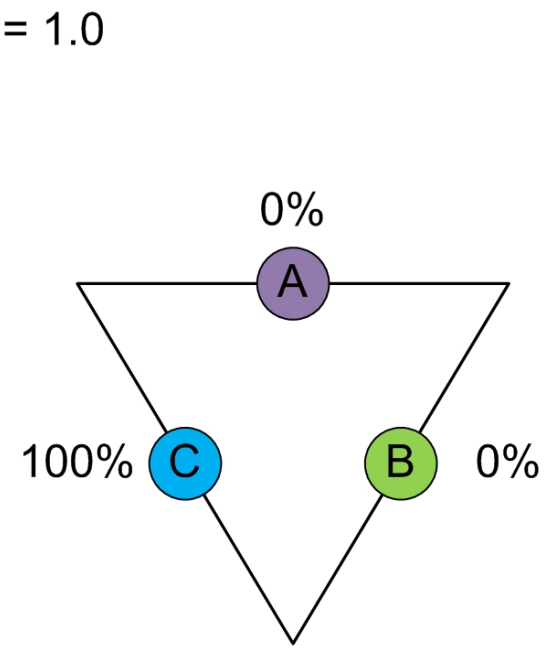
Scheduling of nodes for **L**ifetime maximization of area **C**overage **(SLC)**

Beispiel

- Sensoren A, B, C mit Kapazität 1
- 3 mögliche Überdeckungen: AB, BC, AC

(a) Lasse AB für t=1 aktiv
↪ $T = 1.0$, keine weitere Überdeckung

(b) Lasse AB für t=0.5 aktiv,
danach BC für t=0.5 aktiv,
danach CA für t=0.5 aktiv
↪ $T = 1.5$, 50% längere Laufzeit



Überblick

Problemformulierung - 2

Problembezeichnung

Scheduling of nodes for **L**ifetime maximization of area **C**overage **(SLC)**

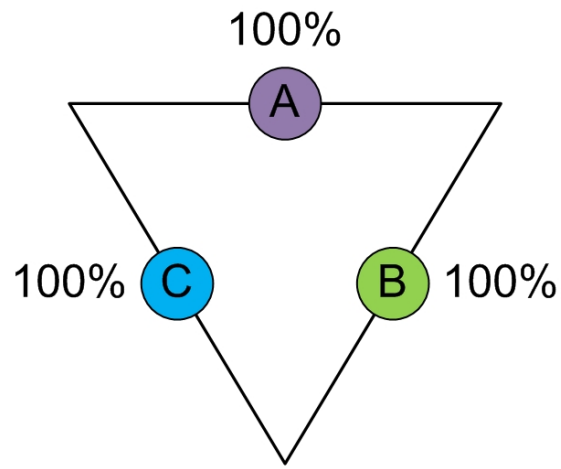
Beispiel

- ≫ Sensoren A, B, C mit Kapazität 1
- ≫ 3 mögliche Überdeckungen: AB, BC, AC

(a) Lasse AB für t=1 aktiv
↪ $T = 1.0$, keine weitere Überdeckung

(b) Lasse AB für t=0.5 aktiv,
danach BC für t=0.5 aktiv,
danach CA für t=0.5 aktiv
↪ $T = 1.5$, 50% längere Laufzeit

(b) optimale Lösung



Überblick

Problemformulierung - 2

Problembezeichnung

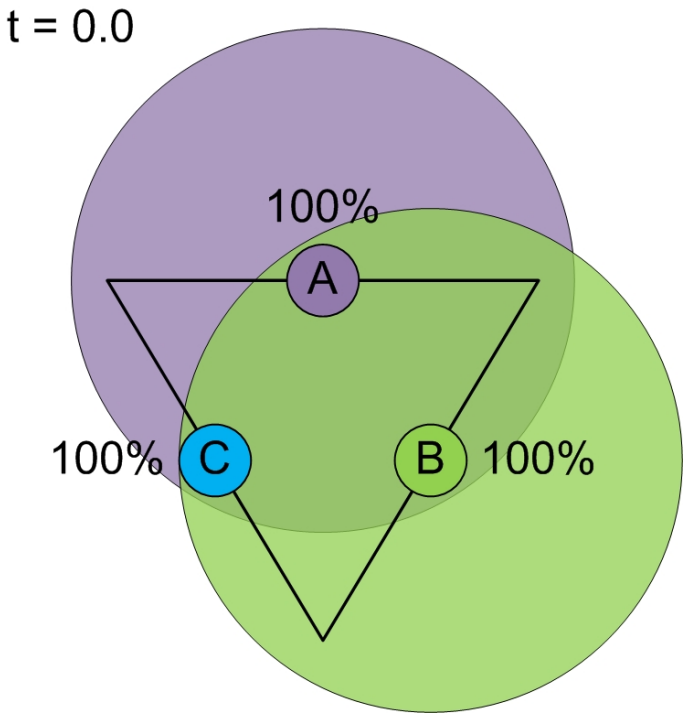
Scheduling of nodes for **L**ifetime maximization of area **C**overage **(SLC)**

Beispiel

- Sensoren A, B, C mit Kapazität 1
- 3 mögliche Überdeckungen: AB, BC, AC

(a) Lasse AB für t=1 aktiv
↪ $T = 1.0$, keine weitere Überdeckung

(b) Lasse AB für t=0.5 aktiv,
danach BC für t=0.5 aktiv,
danach CA für t=0.5 aktiv
↪ $T = 1.5$, 50% längere Laufzeit



Überblick

Problemformulierung - 2

Problembezeichnung

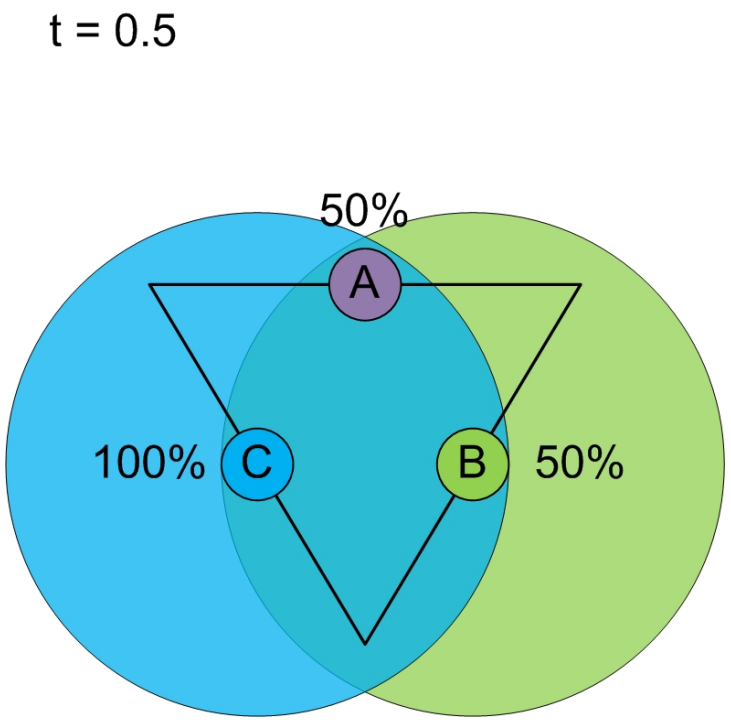
Scheduling of nodes for **L**ifetime maximization of area **C**overage **(SLC)**

Beispiel

- Sensoren A, B, C mit Kapazität 1
- 3 mögliche Überdeckungen: AB, BC, AC

(a) Lasse AB für t=1 aktiv
↪ $T = 1.0$, keine weitere Überdeckung

(b) Lasse AB für t=0.5 aktiv,
danach BC für t=0.5 aktiv,
danach CA für t=0.5 aktiv
↪ $T = 1.5$, 50% längere Laufzeit



Überblick

Problemformulierung - 2

Problembezeichnung

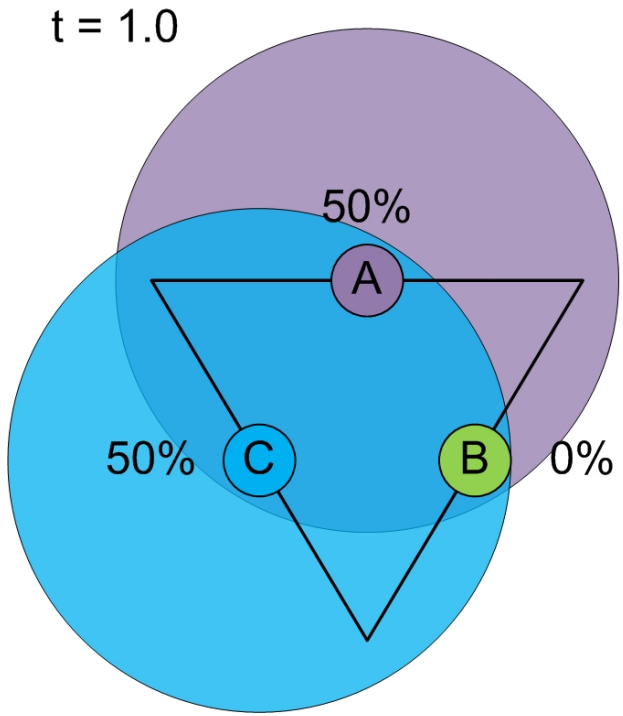
Scheduling of nodes for **L**ifetime maximization of area **C**overage **(SLC)**

Beispiel

- Sensoren A, B, C mit Kapazität 1
- 3 mögliche Überdeckungen: AB, BC, AC

(a) Lasse AB für t=1 aktiv
↪ $T = 1.0$, keine weitere Überdeckung

(b) Lasse AB für t=0.5 aktiv,
danach BC für t=0.5 aktiv,
danach CA für t=0.5 aktiv
↪ $T = 1.5$, 50% längere Laufzeit



Überblick

Problemformulierung - 2

Problembezeichnung

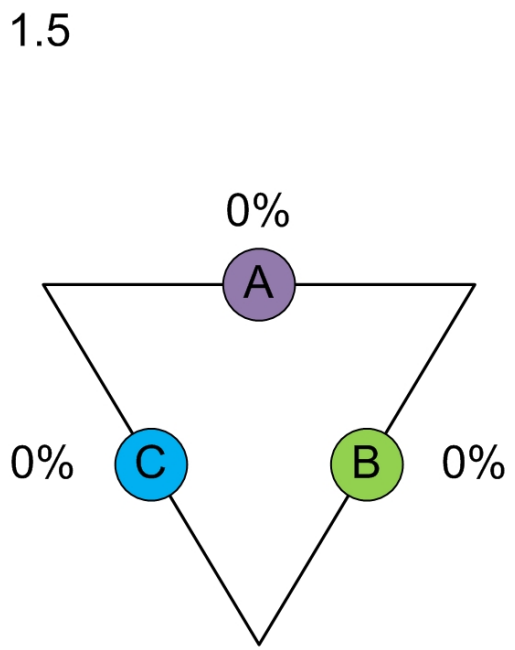
Scheduling of nodes for **L**ifetime maximization of area **C**overage **(SLC)**

Beispiel

- Sensoren A, B, C mit Kapazität 1
- 3 mögliche Überdeckungen: AB, BC, AC

(a) Lasse AB für t=1 aktiv
↪ $T = 1.0$, keine weitere Überdeckung

(b) Lasse AB für t=0.5 aktiv,
danach BC für t=0.5 aktiv,
danach CA für t=0.5 aktiv
↪ $T = 1.5$, 50% längere Laufzeit



Problemformulierung

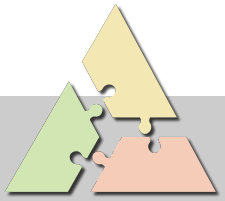
Beschreibung des Modells

gegeben:

- Gebiet F
- N Sensorknoten $S = \{s_i\}$ mit
  - fester Position im Gebiet F
  - kreisförmigem Überwachungsbereich (Radius r)
  - beschränkter Kapazität c_i

gesucht:

- maximale Zeit T, die das gesamte Gebiet überwacht werden kann
- Lösung umfasst:
  - Einteilung der Sensoren in M Covers $\{\mathcal{C}_j\}$, die das Gebiet überwachen
  - Zeiten $\{t_j\}$ die jedes Cover $\mathcal{C}_j$ aktiv ist (Scheduling)



Problemformulierung

Beschreibung mit Linear Programming (LP)

Zu maximieren: Überwachungszeitraum

$$T = max\{\mathbf{b}^T\mathbf{t}|\mathbf{t} \in \mathbb{R}^M\}$$

Nebenbedingungen: begrenzte Knotenkapazitäten

$$\sum_{j=1}^{M} A_{i,j}\ t_j \leq c_i \qquad i = 1, \ldots N$$

- t_j: Dauer für Cover $\mathcal{C}_j$ aktiv
- b_j: 1
- $A_{i,j}$: 1, wenn Knoten s_i in Cover $\mathcal{C}_j$ aktiv, sonst 0
- c_i: Kapazität von Knoten s_i

Linear Programming

Nützliche Eigenschaften

Duales Problem

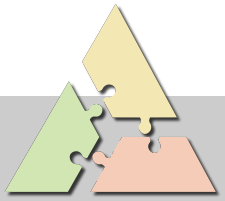
Zu jedem primalen Problem

$$max\{\mathbf{b}^T\mathbf{t}|A\mathbf{t} \leq \mathbf{c}, \mathbf{t} \in \mathbb{R}^M\}$$

existiert ein duales Problem

$$min\{\mathbf{c}^T\mathbf{w}|A^T\mathbf{w} \geq \mathbf{b}, \mathbf{w} \in \mathbb{R}^N\}$$

- w_i: neue Variable für duales Problem
  Interpretation für SLC: "Kosten" eines Knotens s_i



Schwere des Problems

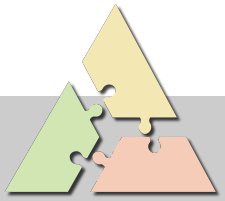
Skizze des NP-Vollständigkeits-Beweises - 1

(1) **Separationsproblem für das duale Problem von SLC:**
gleiche Komplexität wie primales Problem [GrötschelLoSc81]

$$\textit{Gegeben } w, \textit{ existiert ein Cover } C_j \textit{ mit Kosten } \sum_{i,\ A_{i,j}=1} w_i < b_j \ ?$$

(2) **Minimum Dominating Set (MDS) für Unit Disk Graphen:**
bewiesenermaßen NP-hart [MasuyamaIbHa81]

Gegeben ein Unit Disk Graph G = (S,E), finde D ⊆ S mit |D| minimal und f.a. d ∈ D: d ∈ S oder (d, s) ∈ E mit s ∈ S



Schwere des Problems

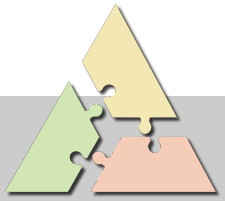
Skizze des NP-Vollständigkeits-Beweises - 2

- Man kann zeigen, dass (2) ein Spezialfall von (1) ist
  - Sensornetzwerk als Unit Disk Graph interpretieren
  - zu überwachende Fläche auf Positionen der Knoten reduzieren
  - alle Knoten-Kosten gleich wählen

⇒ **SLC ist NP-hart**

- Eine potentielle Lösung kann in polynomieller Zeit überprüft werden

⇒ **SLC ist NP-vollständig**



Approximationsalgorithmus

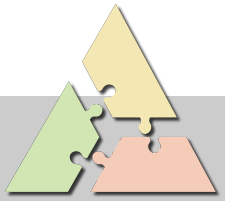
Vorbemerkungen

Definitionen

- Sei T_r eine Lösung für eine Instanz von SLC mit Sensorradien r,
- $T_r = opt_r$ sei die optimale Lösung

Ansatz

- Relaxiere zwei Eigenschaften, um eine approximierte Lösung schnell berechnen zu können
  - Sensorradien r
  - max. Überwachungsdauer T



Approximationsalgorithmus

Vorgehen - erste Approximation

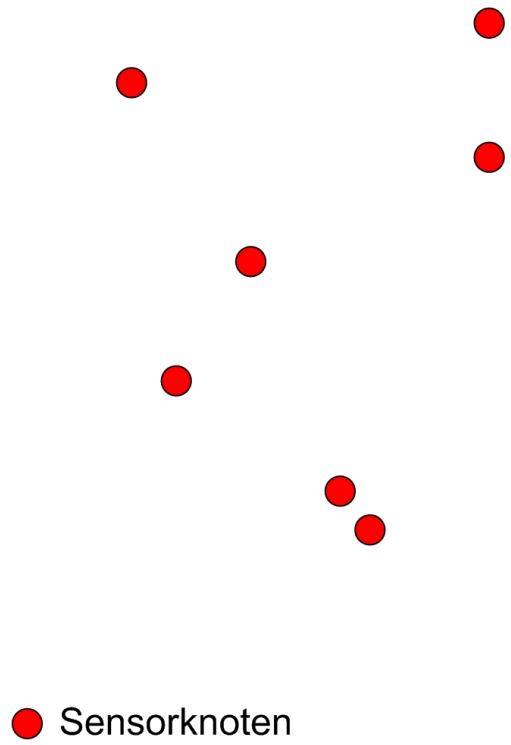
Sensorradien

≫ Diskretisierung der Sensorpositionen auf Gitterpunkte eines $r \cdot \delta/2$-Gitters

≫ Algorithmus $\mathcal{A}$ liefere eine α-Approximation für dieses Problem

→ $\mathcal{A}$ liefert Lösungen $T_r \geq \alpha \cdot opt_{(1-\delta)r}$ für das Orginalproblem

→ Relaxierung der festen Sensorradien: entspricht Vergrößerung um δ



Approximationsalgorithmus

Vorgehen - erste Approximation

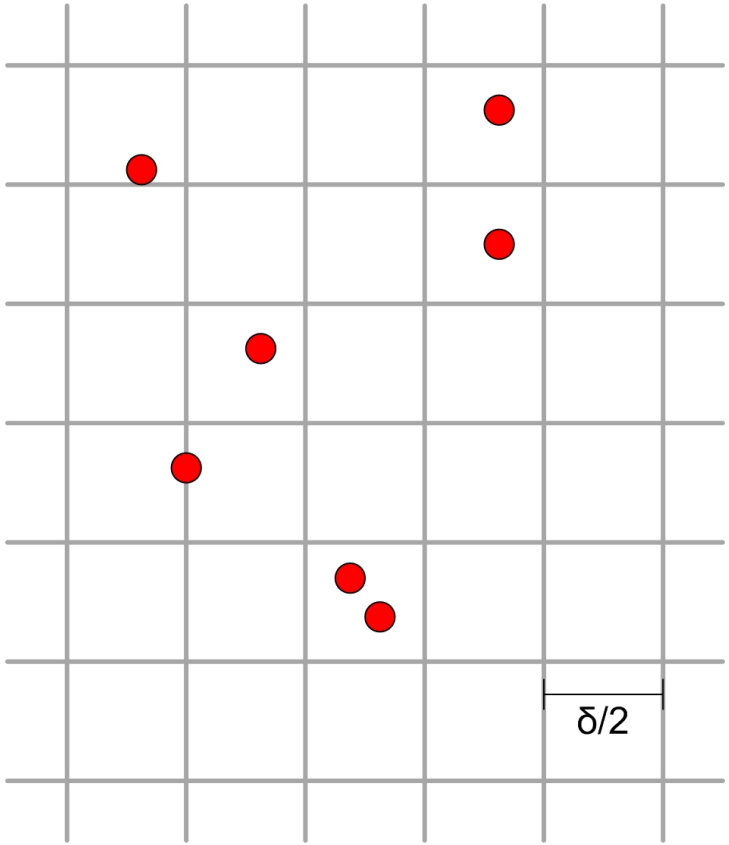
Sensorradien

≫ Diskretisierung der Sensorpositionen auf Gitterpunkte eines $r \cdot \delta/2$-Gitters

≫ Algorithmus $\mathcal{A}$ liefere eine α-Approximation für dieses Problem

→ $\mathcal{A}$ liefert Lösungen $T_r \geq \alpha \cdot opt_{(1-\delta)r}$ für das Orginalproblem

→ Relaxierung der festen Sensorradien: entspricht Vergrößerung um δ



Approximationsalgorithmus

Vorgehen - erste Approximation

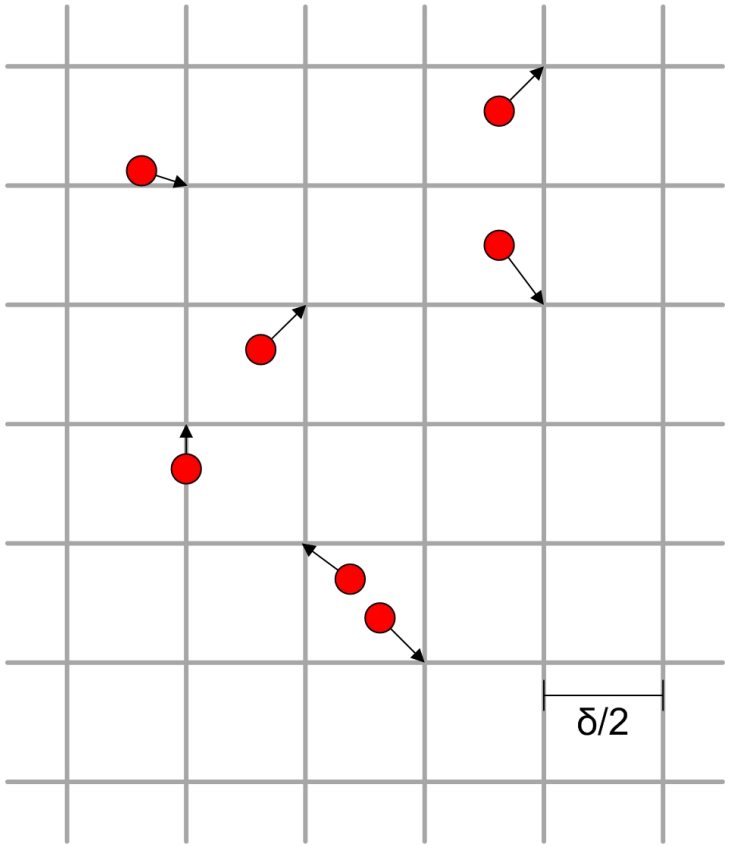
Sensorradien

≫ Diskretisierung der Sensorpositionen auf Gitterpunkte eines $r \cdot \delta/2$-Gitters

≫ Algorithmus $\mathcal{A}$ liefere eine α-Approximation für dieses Problem

→ $\mathcal{A}$ liefert Lösungen $T_r \geq \alpha \cdot opt_{(1-\delta)r}$ für das Orginalproblem

→ Relaxierung der festen Sensorradien: entspricht Vergrößerung um δ



Approximationsalgorithmus

Vorgehen - erste Approximation

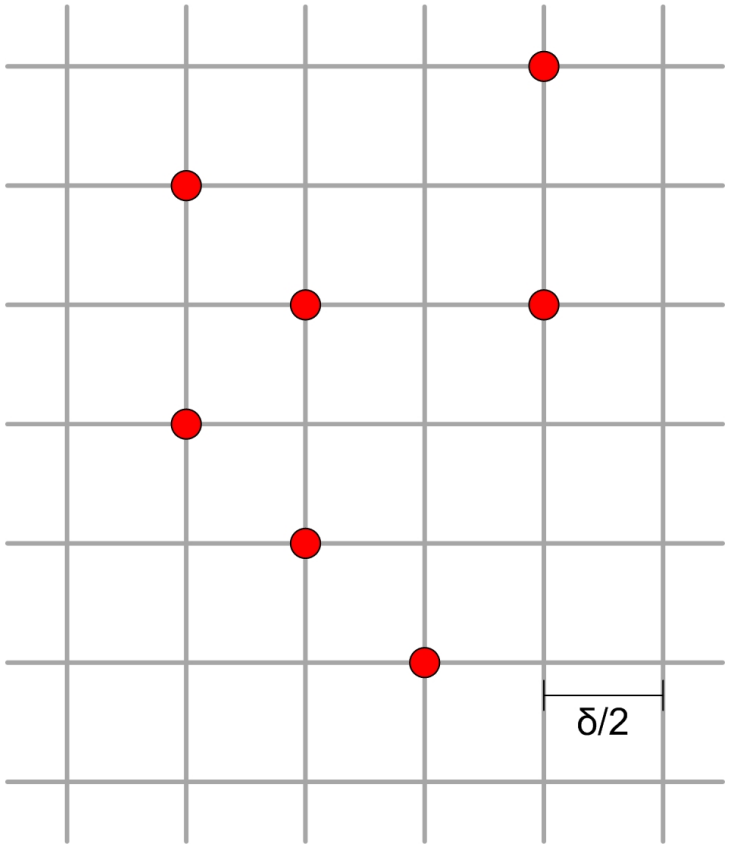
Sensorradien

≫ Diskretisierung der Sensorpositionen auf Gitterpunkte eines $r \cdot \delta/2$-Gitters

≫ Algorithmus $\mathcal{A}$ liefere eine α-Approximation für dieses Problem

→ $\mathcal{A}$ liefert Lösungen $T_r \geq \alpha \cdot opt_{(1-\delta)r}$ für das Orginalproblem

→ Relaxierung der festen Sensorradien: entspricht Vergrößerung um δ



Approximationsalgorithmus

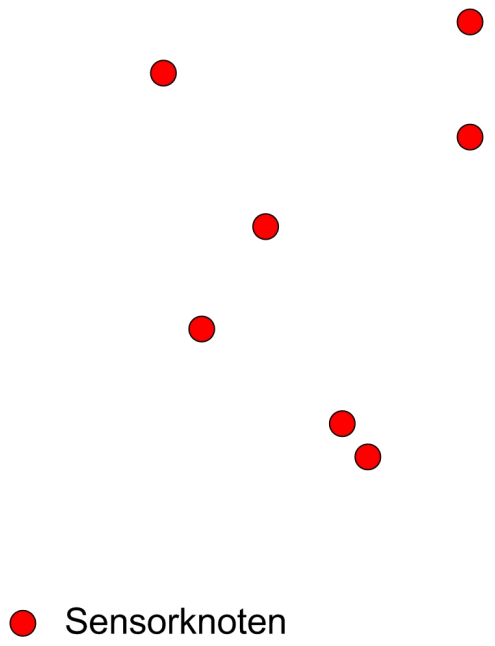
Vorgehen - zweite Approximation

Überwachungsdauer

- Erzeuge Unterteilung $\mathcal{T}$ des Gebiets in Kacheln mit Kantenlänge $k = \lceil 10/\epsilon \rceil$
- Erzeuge Verschiebungen $\mathcal{T}_i$ von $\mathcal{T}$ um (i, i) mit $i \in \mathbb{Z}_k$

Beobachtung für r = 1:

- für jede Sensorfläche gilt:
  - geschnitten von max. 2 der $\mathcal{T}_i$,
  - dann verteilt auf max. 4 Kacheln

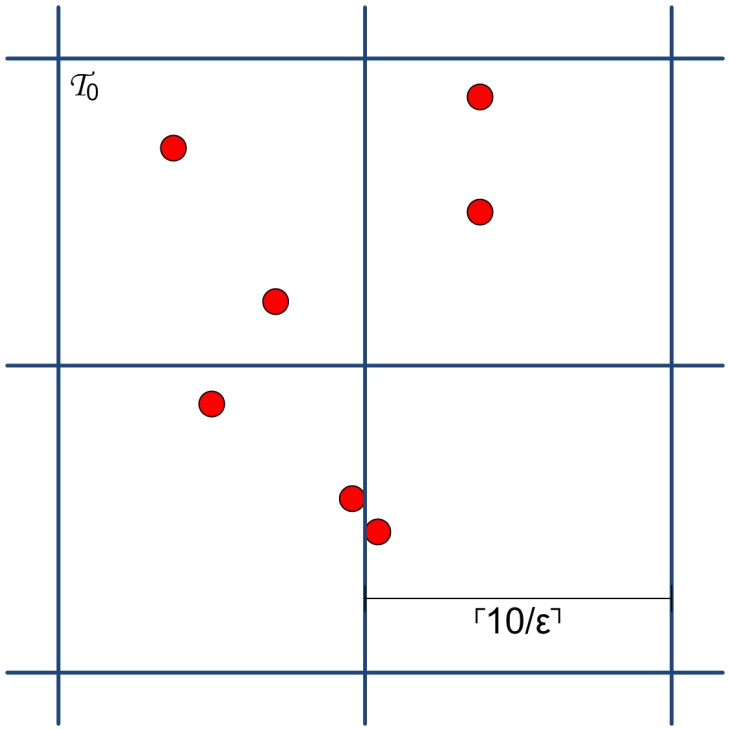


Approximationsalgorithmus

Vorgehen - zweite Approximation

Überwachungsdauer

- Erzeuge Unterteilung $\mathcal{T}$ des Gebiets in Kacheln mit Kantenlänge $k = \lceil 10/\epsilon \rceil$
- Erzeuge Verschiebungen $\mathcal{T}_i$ von $\mathcal{T}$ um (i, i) mit $i \in \mathbb{Z}_k$



Beobachtung für r = 1:

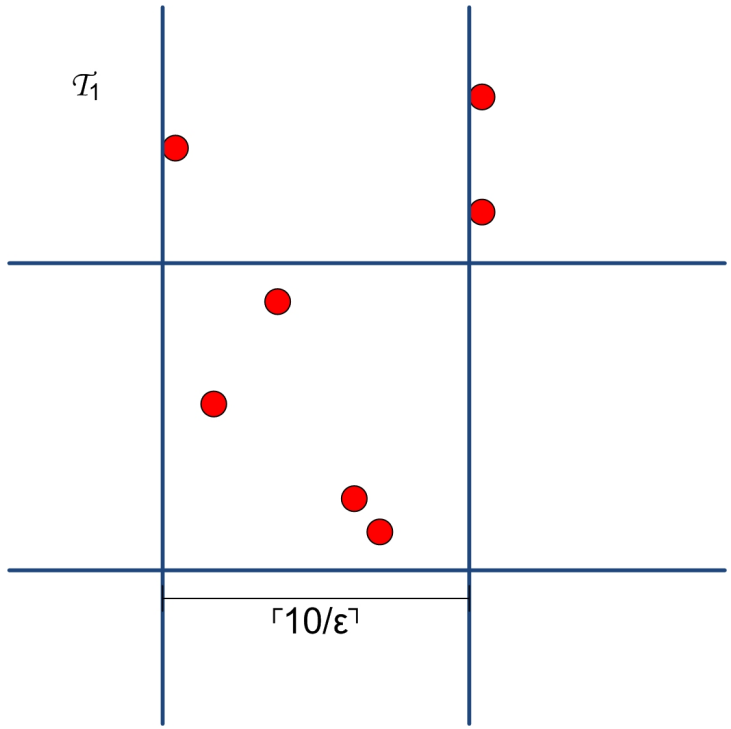
- für jede Sensorfläche gilt:
  - geschnitten von max. 2 der $\mathcal{T}_i$,
  - dann verteilt auf max. 4 Kacheln

Approximationsalgorithmus

Vorgehen - zweite Approximation

Überwachungsdauer

- Erzeuge Unterteilung $\mathcal{T}$ des Gebiets in Kacheln mit Kantenlänge $k = \lceil 10/\epsilon \rceil$
- Erzeuge Verschiebungen $\mathcal{T}_i$ von $\mathcal{T}$ um (i, i) mit $i \in \mathbb{Z}_k$



Beobachtung für r = 1:

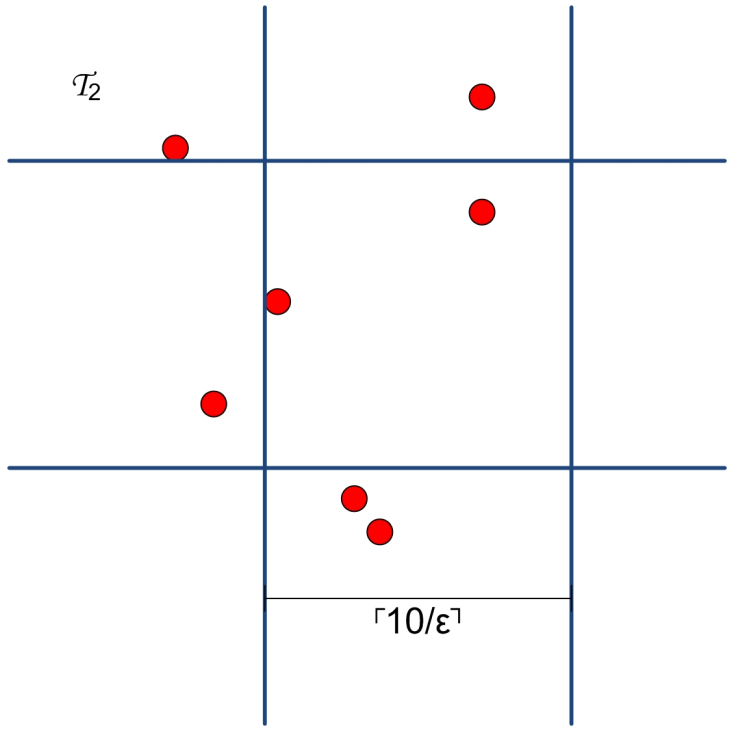
- für jede Sensorfläche gilt:
  - geschnitten von max. 2 der $\mathcal{T}_i$,
  - dann verteilt auf max. 4 Kacheln

Approximationsalgorithmus

Vorgehen - zweite Approximation

Überwachungsdauer

- Erzeuge Unterteilung $\mathcal{T}$ des Gebiets in Kacheln mit Kantenlänge $k = \lceil 10/\epsilon \rceil$
- Erzeuge Verschiebungen $\mathcal{T}_i$ von $\mathcal{T}$ um (i, i) mit $i \in \mathbb{Z}_k$



Beobachtung für $r = 1$:

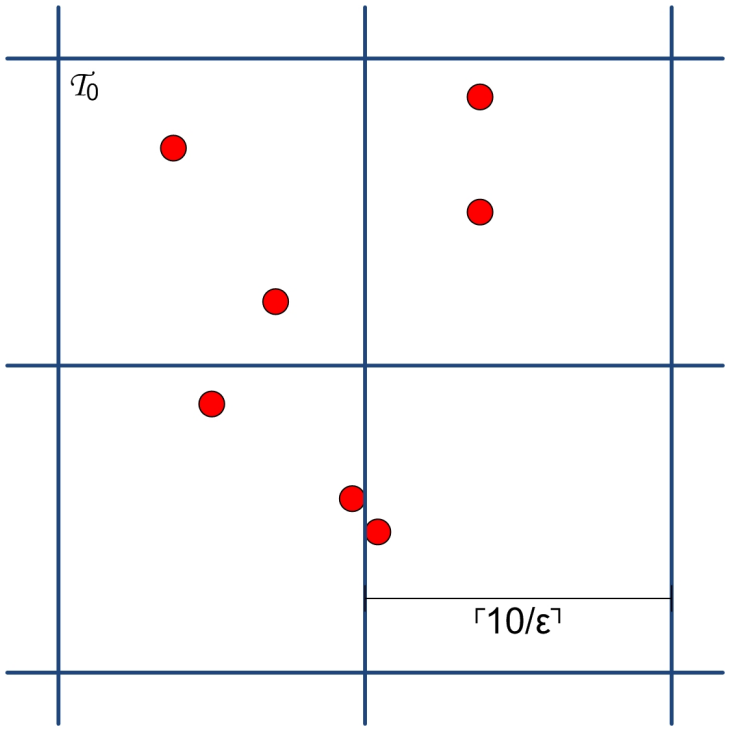
- für jede Sensorfläche gilt:
  - geschnitten von max. 2 der $\mathcal{T}_i$,
  - dann verteilt auf max. 4 Kacheln

Approximationsalgorithmus

Vorgehen - zweite Approximation

Überwachungsdauer

- Erzeuge Unterteilung $\mathcal{T}$ des Gebiets in Kacheln mit Kantenlänge $k = \lceil 10/\epsilon \rceil$
- Erzeuge Verschiebungen $\mathcal{T}_i$ von $\mathcal{T}$ um (i, i) mit $i \in \mathbb{Z}_k$



Beobachtung für r = 1:

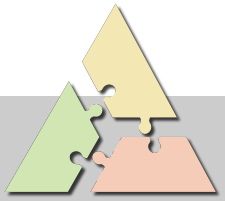
- für jede Sensorfläche gilt:
  - geschnitten von max. 2 der $\mathcal{T}_i$,
  - dann verteilt auf max. 4 Kacheln

Approximationsalgorithmus

Vorgehen - zweite Approximation

- Algorithmus $\mathcal{A}$ liefere α-Approximation für SLC Probleme beschränkt auf Gebiete der Größe $k \times k$
  - Anwendung von $\mathcal{A}$ auf jede Kachel von $\mathcal{T}_i$ liefert Lösung mit:
    - $T_1 = \alpha \cdot opt_1$
    - max. $4x$ Überlastung jedes Sensors
  - Vereinigung der Lösungen $\{t_j\}_i$ aller $\mathcal{T}_i$ nach $\{t_j\} = \frac{1-\epsilon}{k} \sum_{i \in \mathbb{Z}_k} \{t_j\}_i$ liefert Gesamtlösung mit:
    - $T_1 = (1-\epsilon) \cdot \alpha \cdot opt_1$
    - keine Überlastung der Sensoren

→ Relaxierung der maximalen Überwachungsdauer: entspricht Verkleinerung um ϵ

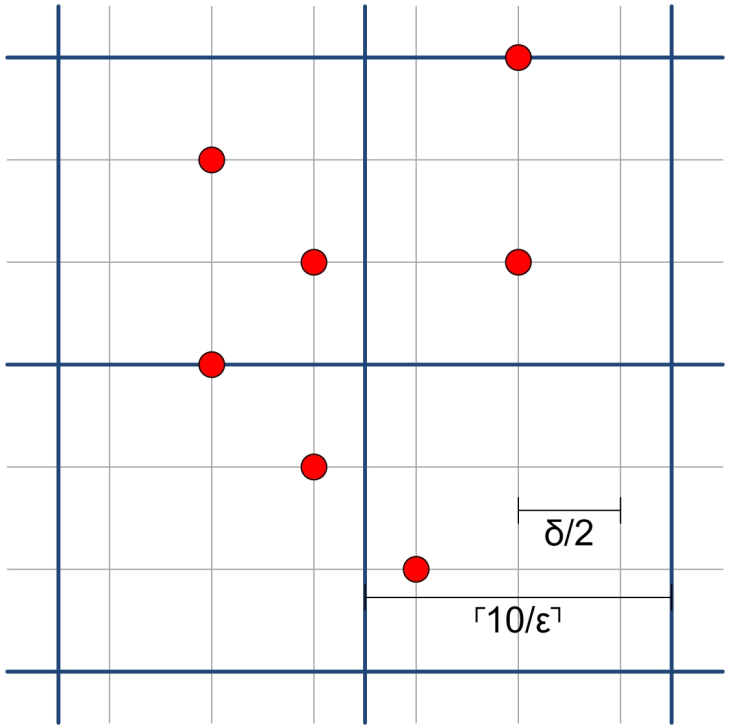


Approximationsalgorithmus

Vorgehen - gesamt

Verbindung beider Approximationen

- benötige Algorithmus $\mathcal{A}$, der α-Approximation für SLC berechnet
  - für Gebiete der Größe $k \times k$, und
  - für diskretisierte Sensorpositionen auf Gitterpunkte mit Abstand $\delta/2$



Beobachtung:

- Jede Kachel wird von maximal $O(1/\delta^2\epsilon^2)$ Sensorknoten berührt

→ unabhängig von N !

Approximationsalgorithmus

Ergebnis - 1

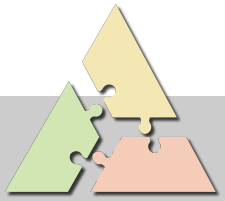
Approximationsgarantie

$$T_1 \geq (1-\epsilon) \cdot \alpha \cdot opt_{1-\delta}$$

- $(1-\epsilon)$: Aufteilung des zu überwachenden Gebietes F in kleinere Kacheln
- α: Approximationsgarantie von $\mathcal{A}$
- $opt_{1-\delta}$: Diskretisierung der Sensorpositionen auf Gitterpunkte

Vorgenommene Relaxationen:

- Sensorradien dürfen größer als r sein
- Gesamtzeit T darf kleiner als das Optimum sein



Approximationsalgorithmus

Ergebnis - 2

Laufzeitverhalten

$$O\left(N + 1/\epsilon \cdot \frac{\epsilon^2 \cdot N}{opt_{1-\delta}} \cdot f\left(O\left(1/\delta^2\epsilon^2\right)\right)\right)$$

- $O(N)$: Kosten für Verschiebung der Sensorknoten auf Gitterpunkte
- $O(1/\epsilon)$: Anzahl an Unterteilungen $\mathcal{T}_i$ des Gebiets F
- $O(\frac{\epsilon^2 \cdot N}{opt_{1-\delta}})$: Anzahl zu betrachtender Kacheln
- $O\left(f\left(O\left(1/\delta^2\epsilon^2\right)\right)\right)$: Laufzeit von Algorithmus $\mathcal{A}$

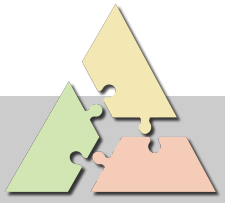
Bemerkungen:

- Laufzeit linear in N
- $\mathcal{A}$ darf exponentielle Zeit benötigen, da unabhängig von N

– II –

Funktionsapproximation

Gaussian Mixture Reduction



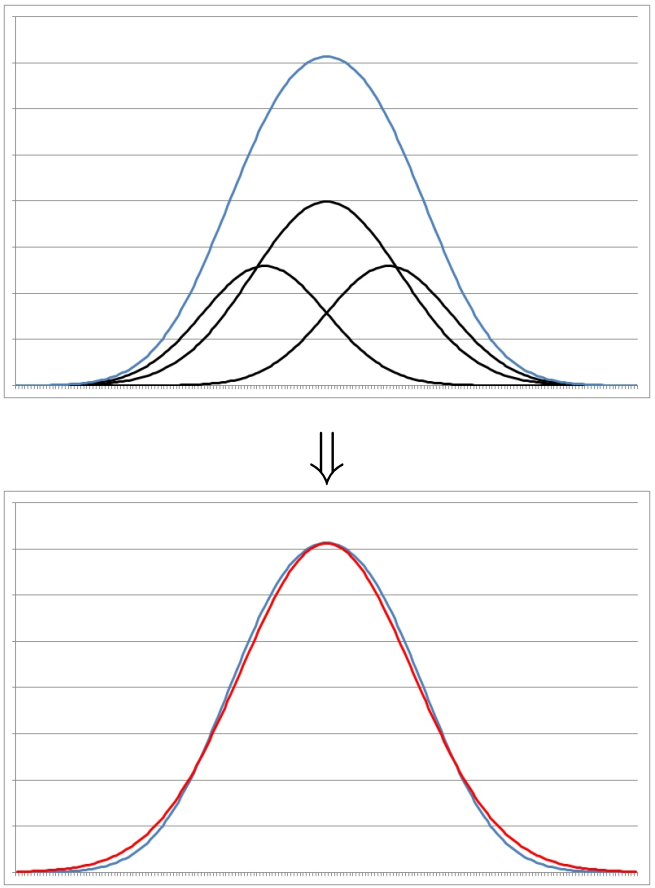
Funktionsapproximation

Problemformulierung - 1

Gaussian Mixture Reduction

- *gegeben*:
  Gaussian-Mixture $G = \sum_{i=1}^{N} w_i \cdot g_i$
- *gesucht*:
  Approximation $A = \sum_{j=1}^{K} w_j \cdot a_j$
  mit minimaler integrierter
  quadratischer Distanz (ISD) zu G

→ inspiriert durch Marco Huber (ISAS)



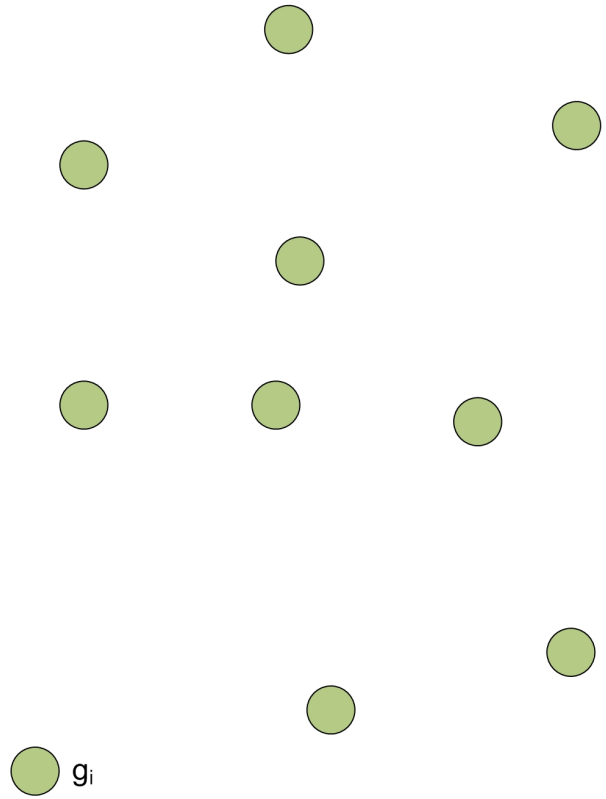
Funktionsapproximation

Problemformulierung - 2

Algorithmik-Ansatz

Ansatz: Clustering / Facility Location

- Betrachte $g_{1..N}, a_{1..K}$ als Punkte in einer Ebene
- Teile $g_{1..N}$ in K Gruppen (Clustering) mit je einem Cluster-Zentrum a_j ein
- **Ziel:** Minimierung eines Distanzmaßes $d(g_i, a_j)$ innerhalb jeder Gruppe j für alle Gruppenmitglieder g_i



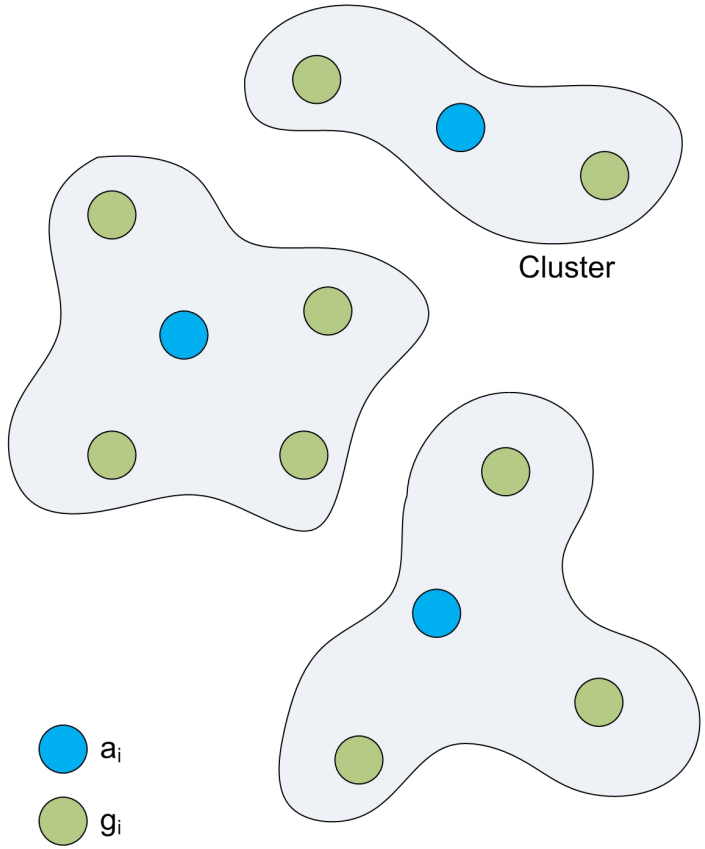
Funktionsapproximation

Problemformulierung - 2

Algorithmik-Ansatz

Ansatz: Clustering / Facility Location

- Betrachte $g_{1..N}, a_{1..K}$ als Punkte in einer Ebene
- Teile $g_{1..N}$ in K Gruppen (Clustering) mit je einem Cluster-Zentrum a_j ein
- **Ziel:** Minimierung eines Distanzmaßes $d(g_i, a_j)$ innerhalb jeder Gruppe j für alle Gruppenmitglieder g_i



Funktionsapproximation

Beschreibung des Algorithmus - 1

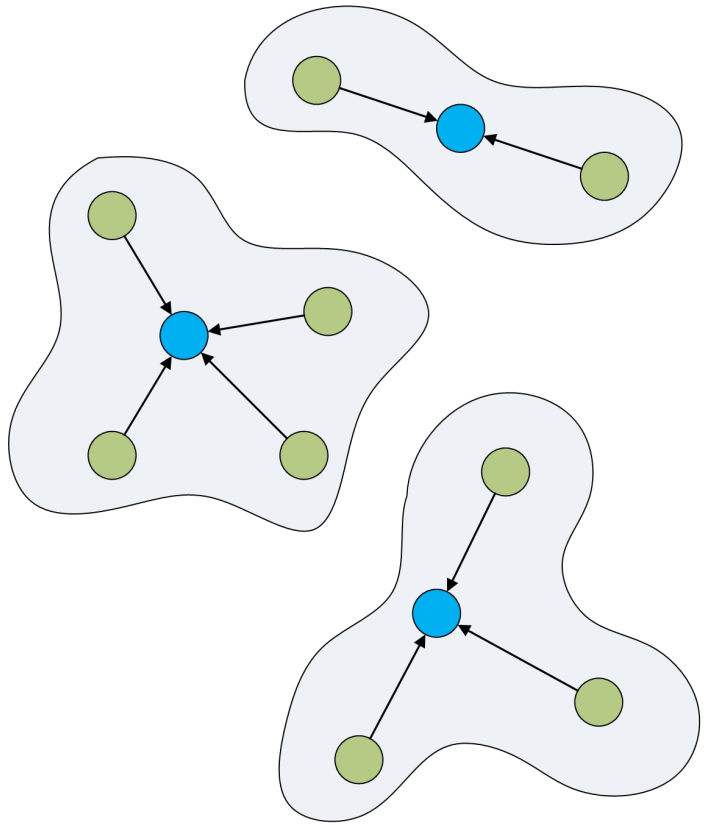
Clustering-Algorithmus

≫ basiert auf Llyods Algorithmus

Initialisierung

≫ Generiere Anfangslösung A_{init}
z.B. mit einfachen Algorithmus
(**hier:** Runnalls Algorithmus)

≫ liefert initiale Zentren $\{a_{j,init}\}$ und Zuordnungen $center(\cdot)_{init} : g_i \rightarrow a_{j,init}$

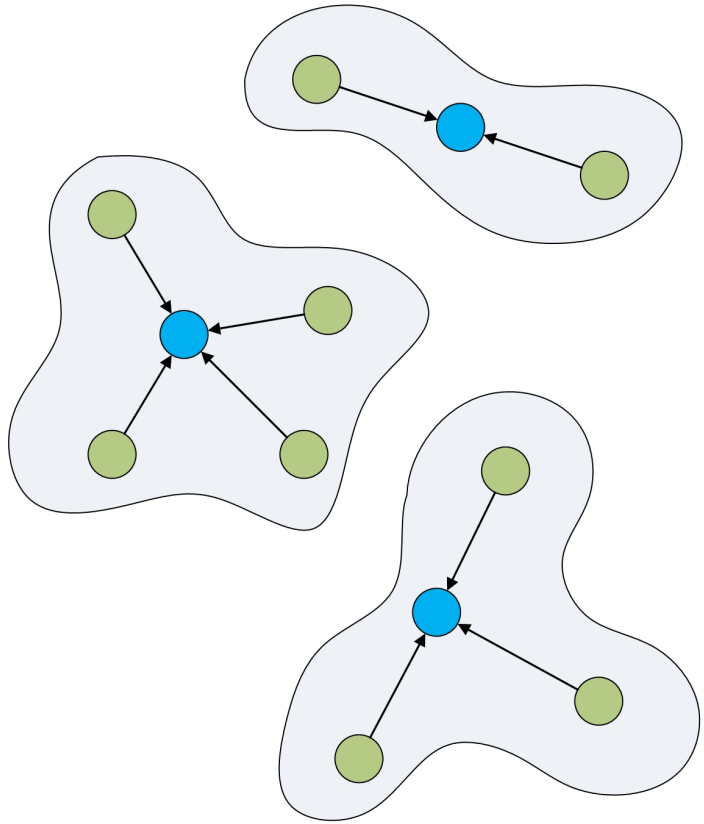


Funktionsapproximation

Beschreibung des Algorithmus - 2

Iteration *it*

- f.a. g_i, berechne Distanz $d(g_i, a_{j,it})$ zu jedem Zentrum $a_{j,it}$
- Ordne jedes g_i dem Zentrum $a_{j,it}$ mit minimaler Distanz zu
  $\rightarrow$ neue Zuordnungen: $center_{it+1}(\cdot)$
- Berechne $a_{j,it+1}$ als Gaussfunktion, die Gewicht, Mittelwert und Varianz von $\sum_{center_{it+1}(g_i)=a_{j,it}} w_i \cdot g_i$ erhält
  $\rightarrow$ neue Zentren: $a_{j,it+1}$

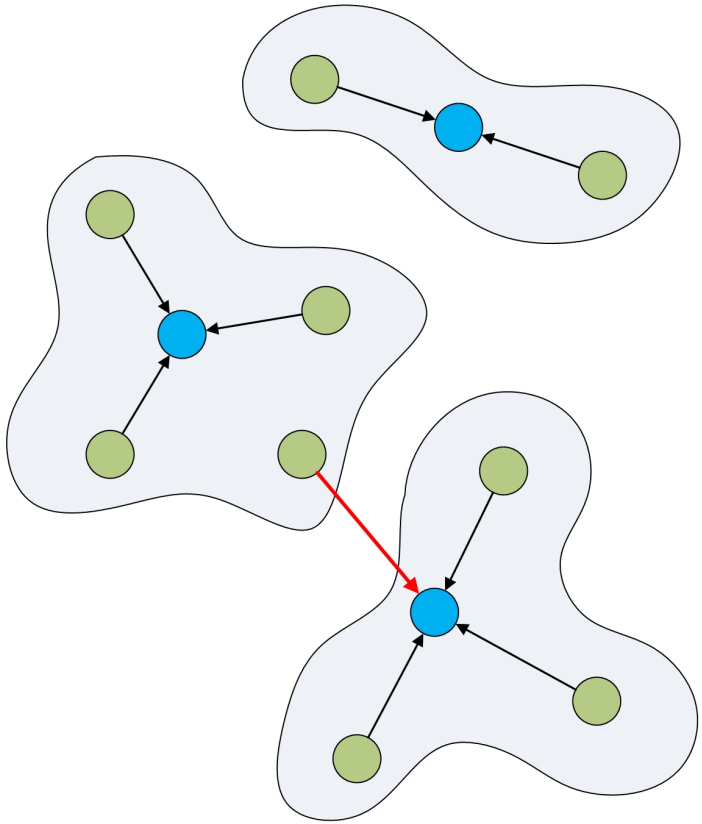


Funktionsapproximation

Beschreibung des Algorithmus - 2

Iteration *it*

- f.a. g_i, berechne Distanz $d(g_i, a_{j,it})$ zu jedem Zentrum $a_{j,it}$
- Ordne jedes g_i dem Zentrum $a_{j,it}$ mit minimaler Distanz zu
  $\rightarrow$ neue Zuordnungen: $center_{it+1}(\cdot)$
- Berechne $a_{j,it+1}$ als Gaussfunktion, die Gewicht, Mittelwert und Varianz von $\sum_{center_{it+1}(g_i)=a_{j,it}} w_i \cdot g_i$ erhält
  $\rightarrow$ neue Zentren: $a_{j,it+1}$

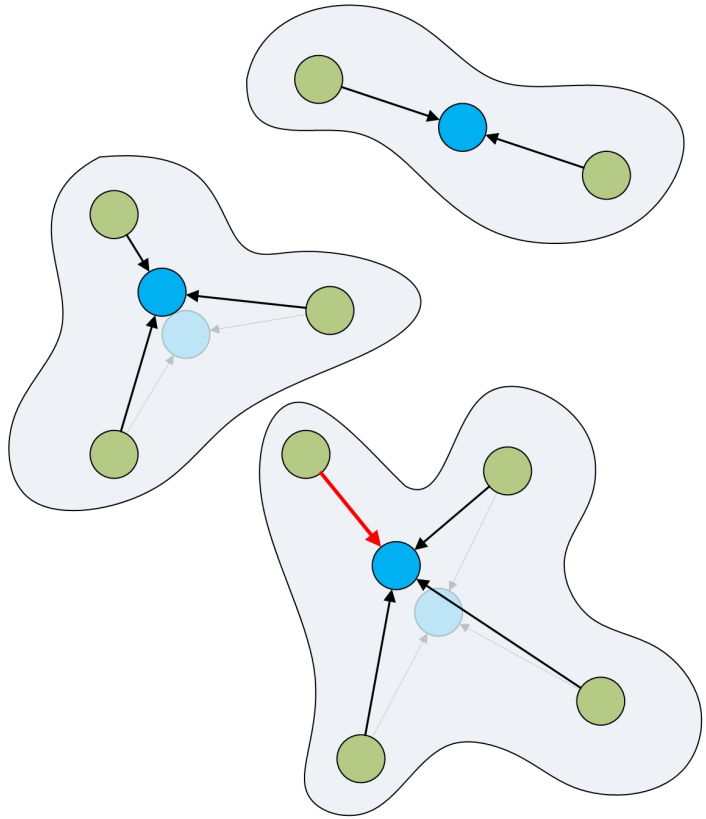


Funktionsapproximation

Beschreibung des Algorithmus - 2

Iteration *it*

- f.a. g_i, berechne Distanz $d(g_i, a_{j,it})$ zu jedem Zentrum $a_{j,it}$
- Ordne jedes g_i dem Zentrum $a_{j,it}$ mit minimaler Distanz zu
  $\rightarrow$ neue Zuordnungen: $center_{it+1}(\cdot)$
- Berechne $a_{j,it+1}$ als Gaussfunktion, die Gewicht, Mittelwert und Varianz von $\sum_{center_{it+1}(g_i)=a_{j,it}} w_i \cdot g_i$ erhält
  $\rightarrow$ neue Zentren: $a_{j,it+1}$



Funktionsapproximation

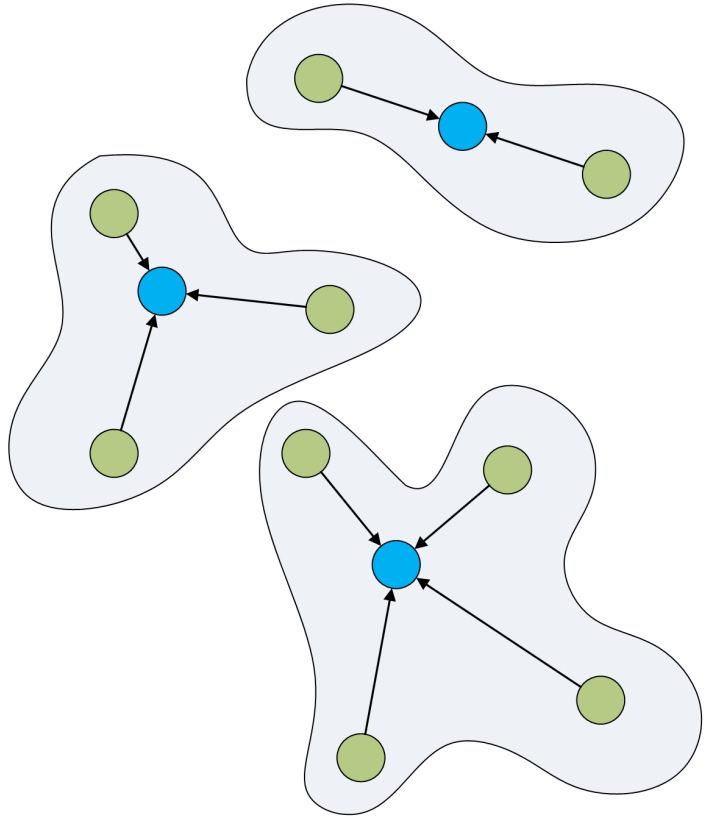
Beschreibung des Algorithmus - 3

Abbruchbedingung

- Beende die Iterationen, wenn
  - Approximationsqualität genügt
  - Zuordnungen konstant bleiben

verwendetes Distanzmaß $d(g_i, a_j)$

- ISD zwischen Orginal-Mixture G und einer Approximation A'
- A' entspricht A_{it}, außer dass g_i dem Zentrum $a_{j,it}$ zugerechnet wird



Funktionsapproximation

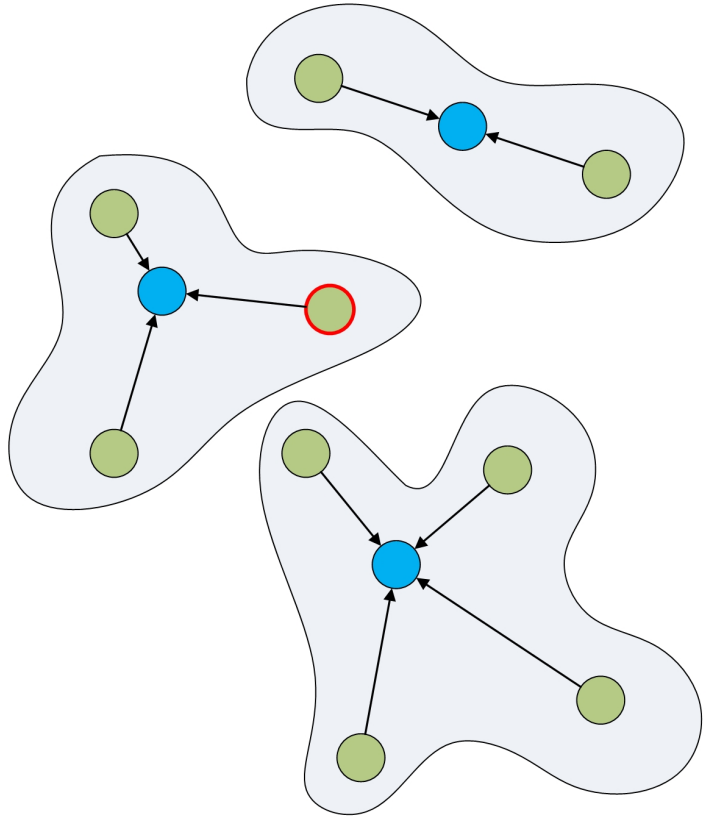
Beschreibung des Algorithmus - 3

Abbruchbedingung

- Beende die Iterationen, wenn
  - Approximationsqualität genügt
  - Zuordnungen konstant bleiben

verwendetes Distanzmaß $d(g_i, a_j)$

- ISD zwischen Orginal-Mixture G und einer Approximation A'
- A' entspricht A_{it}, außer dass g_i dem Zentrum $a_{j,it}$ zugerechnet wird



Funktionsapproximation

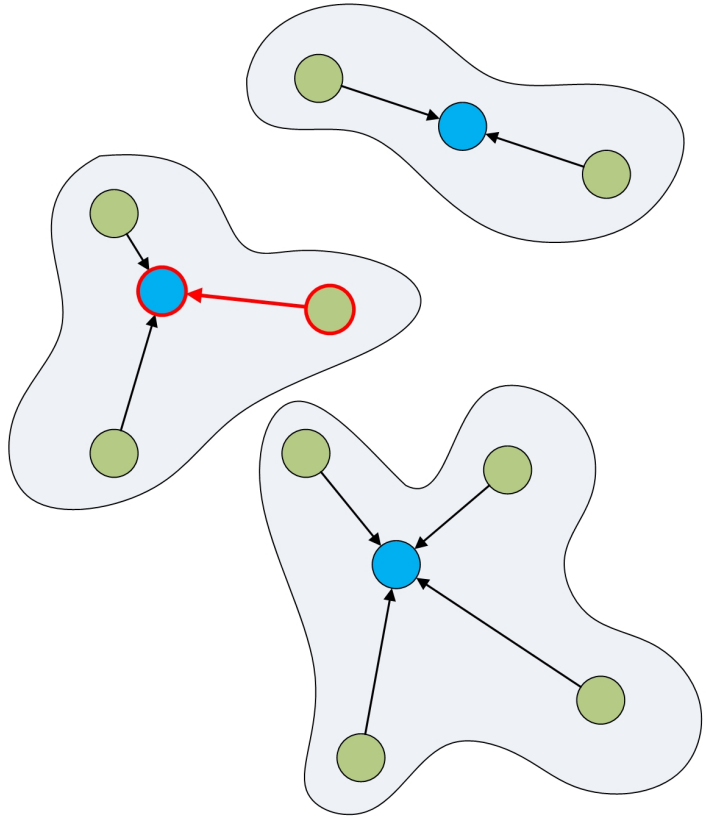
Beschreibung des Algorithmus - 3

Abbruchbedingung

- Beende die Iterationen, wenn
  - Approximationsqualität genügt
  - Zuordnungen konstant bleiben

verwendetes Distanzmaß $d(g_i, a_j)$

- ISD zwischen Orginal-Mixture G und einer Approximation A'
- A' entspricht A_{it}, außer dass g_i dem Zentrum $a_{j,it}$ zugerechnet wird



Funktionsapproximation

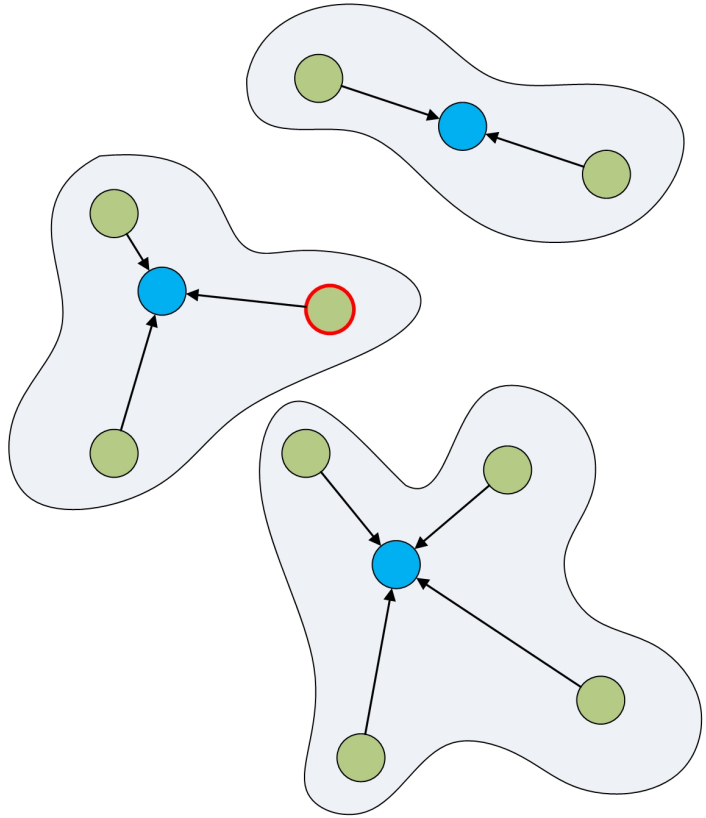
Beschreibung des Algorithmus - 3

Abbruchbedingung

- Beende die Iterationen, wenn
  - Approximationsqualität genügt
  - Zuordnungen konstant bleiben

verwendetes Distanzmaß $d(g_i, a_j)$

- ISD zwischen Orginal-Mixture G und einer Approximation A'
- A' entspricht A_{it}, außer dass g_i dem Zentrum $a_{j,it}$ zugerechnet wird



Funktionsapproximation

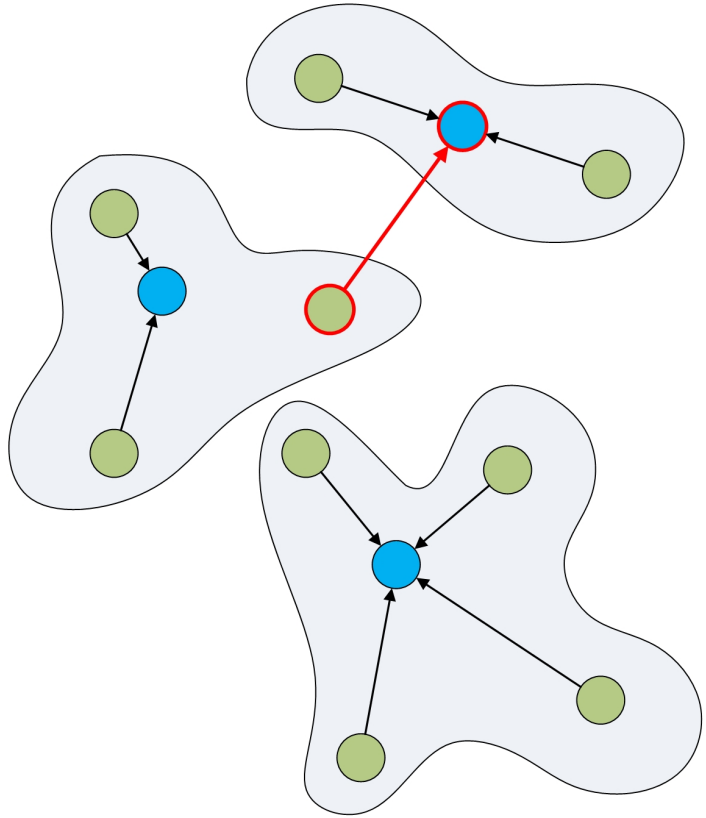
Beschreibung des Algorithmus - 3

Abbruchbedingung

- Beende die Iterationen, wenn
  - Approximationsqualität genügt
  - Zuordnungen konstant bleiben

verwendetes Distanzmaß $d(g_i, a_j)$

- ISD zwischen Orginal-Mixture G und einer Approximation A'
- A' entspricht A_{it}, außer dass g_i dem Zentrum $a_{j,it}$ zugerechnet wird



Funktionsapproximation

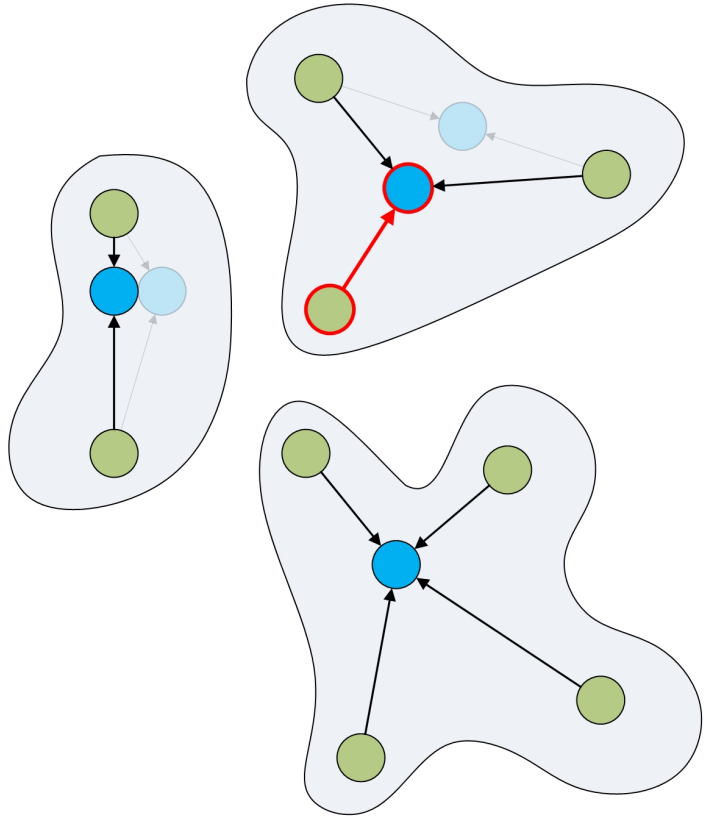
Beschreibung des Algorithmus - 3

Abbruchbedingung

- Beende die Iterationen, wenn
  - Approximationsqualität genügt
  - Zuordnungen konstant bleiben

verwendetes Distanzmaß $d(g_i, a_j)$

- ISD zwischen Orginal-Mixture G und einer Approximation A'
- A' entspricht A_{it}, außer dass g_i dem Zentrum $a_{j,it}$ zugerechnet wird



Funktionsapproximation

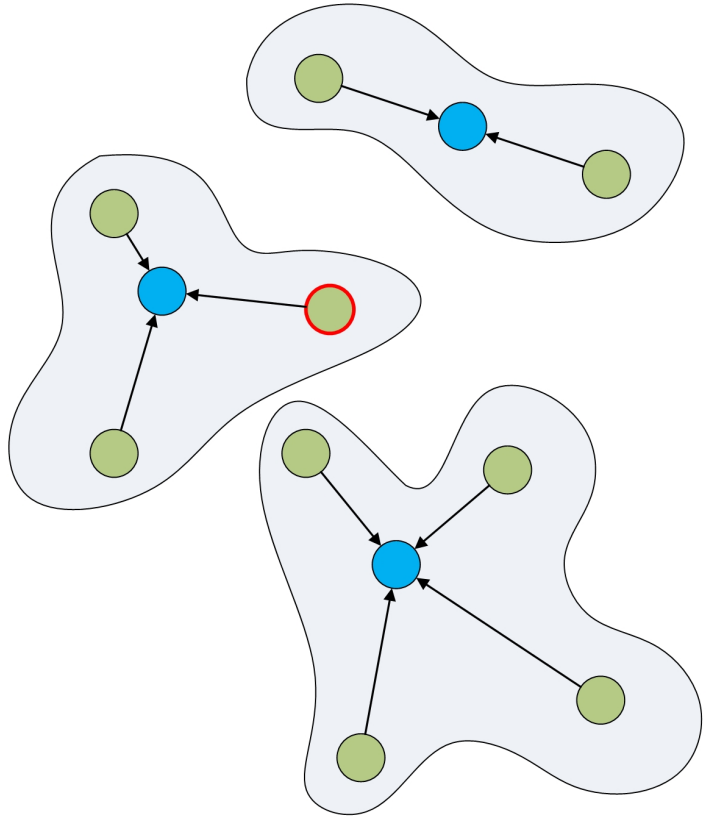
Beschreibung des Algorithmus - 3

Abbruchbedingung

- Beende die Iterationen, wenn
  - Approximationsqualität genügt
  - Zuordnungen konstant bleiben

verwendetes Distanzmaß $d(g_i, a_j)$

- ISD zwischen Orginal-Mixture G und einer Approximation A'
- A' entspricht A_{it}, außer dass g_i dem Zentrum $a_{j,it}$ zugerechnet wird



Funktionsapproximation

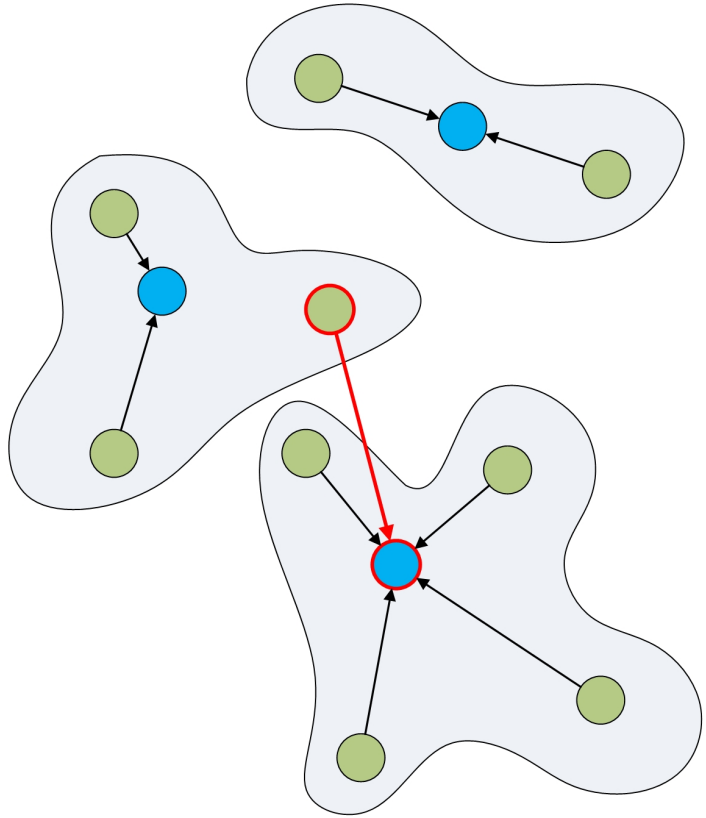
Beschreibung des Algorithmus - 3

Abbruchbedingung

- Beende die Iterationen, wenn
  - Approximationsqualität genügt
  - Zuordnungen konstant bleiben

verwendetes Distanzmaß $d(g_i, a_j)$

- ISD zwischen Orginal-Mixture G und einer Approximation A'
- A' entspricht A_{it}, außer dass g_i dem Zentrum $a_{j,it}$ zugerechnet wird



Funktionsapproximation

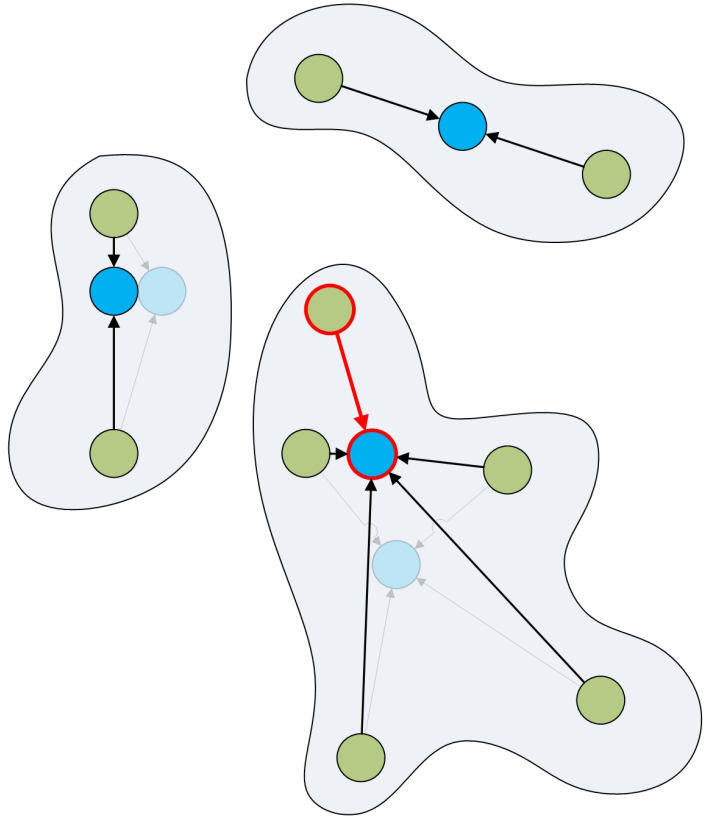
Beschreibung des Algorithmus - 3

Abbruchbedingung

- Beende die Iterationen, wenn
  - Approximationsqualität genügt
  - Zuordnungen konstant bleiben

verwendetes Distanzmaß $d(g_i, a_j)$

- ISD zwischen Orginal-Mixture G und einer Approximation A'
- A' entspricht A_{it}, außer dass g_i dem Zentrum $a_{j,it}$ zugerechnet wird



Funktionsapproximation

Vorläufige Ergebnisse - 1

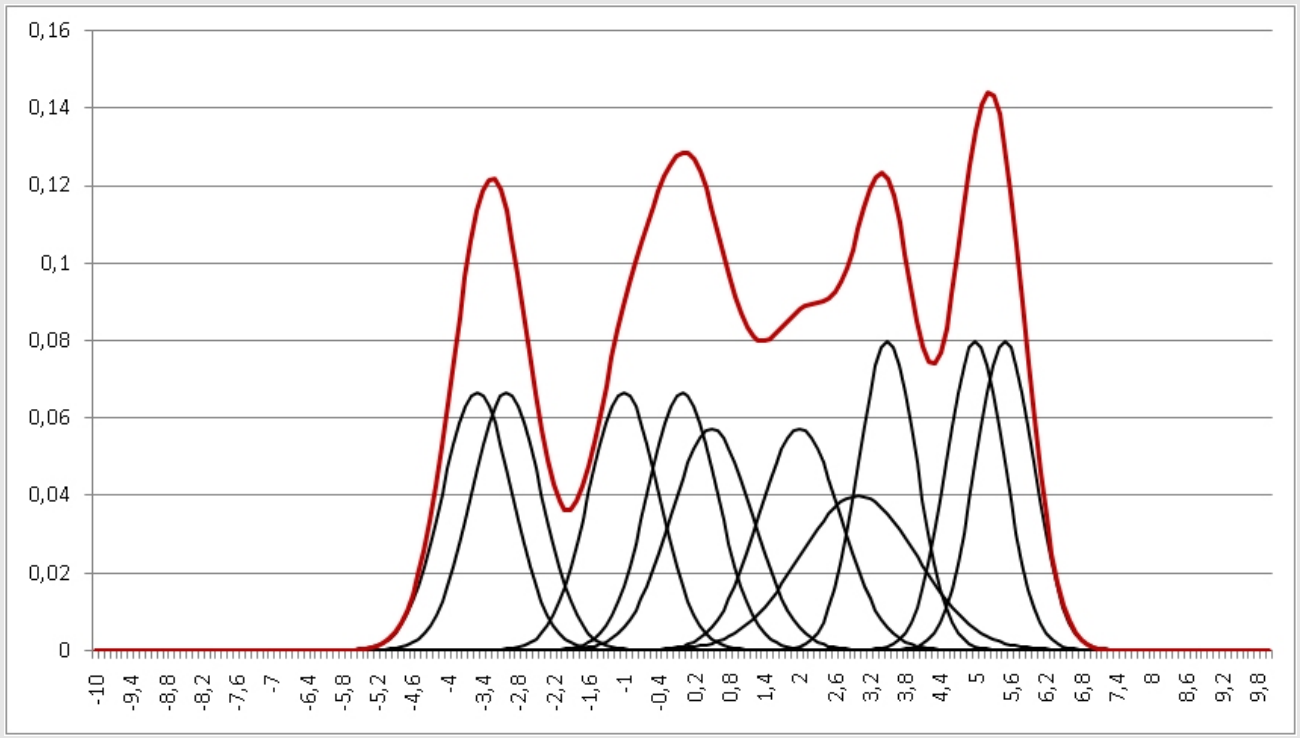
Verfahren	N	K	Dauer*	Abweichung
Clustering	100	5	0,08s	2,07
	100	10	0,29s	0,96
	200	5	0,38s	1,47
	200	10	1,36s	0,81
	400	5	1,82s	1,11
	400	10	7,56s	0,62
West**	200	10	0,04s	3,81
Williams**	200	10	57,92s	1,03
PGMR**	200	5	4,78s	0,64

* Zeiten nur bedingt vergleichbar: Clustering in C++, Rest in Matlab

** Werte übernommen aus [HuberHa08]

Funktionsapproximation

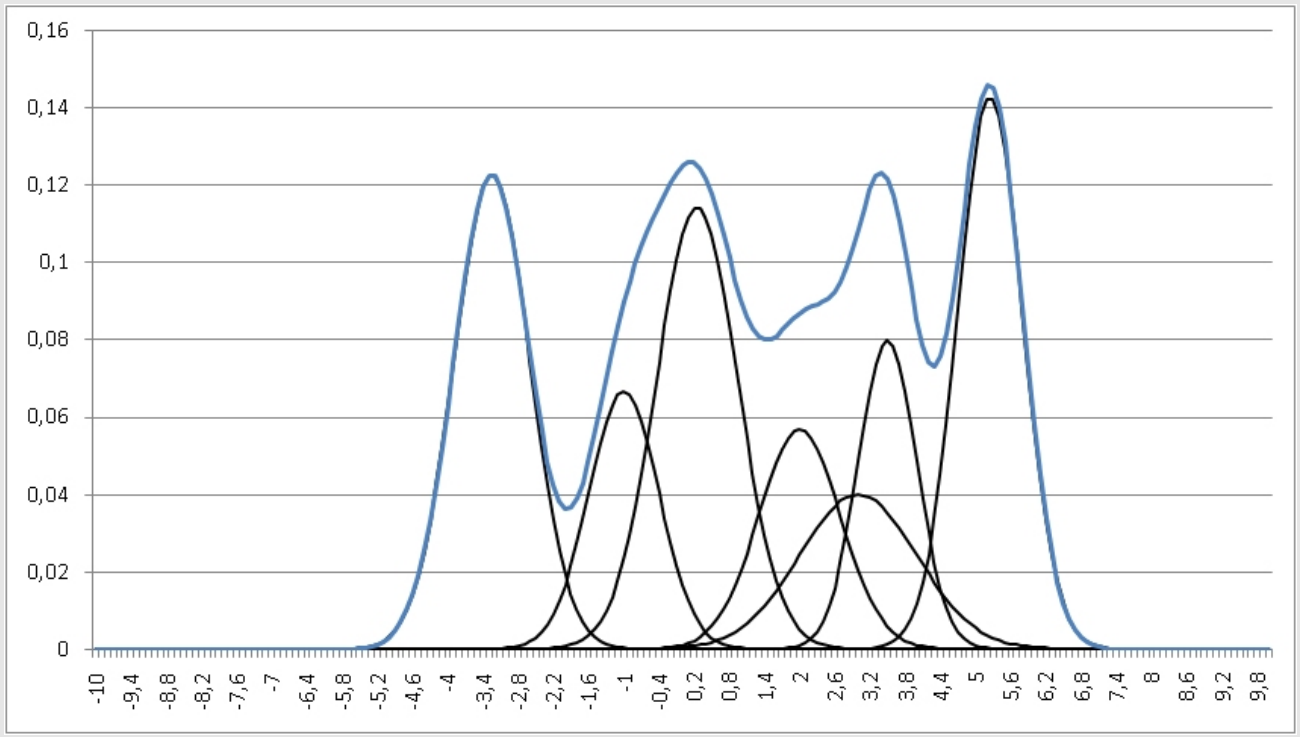
Vorläufige Ergebnisse - 2



Orginal Gaussian Mixture (10 Komponenten)

Funktionsapproximation

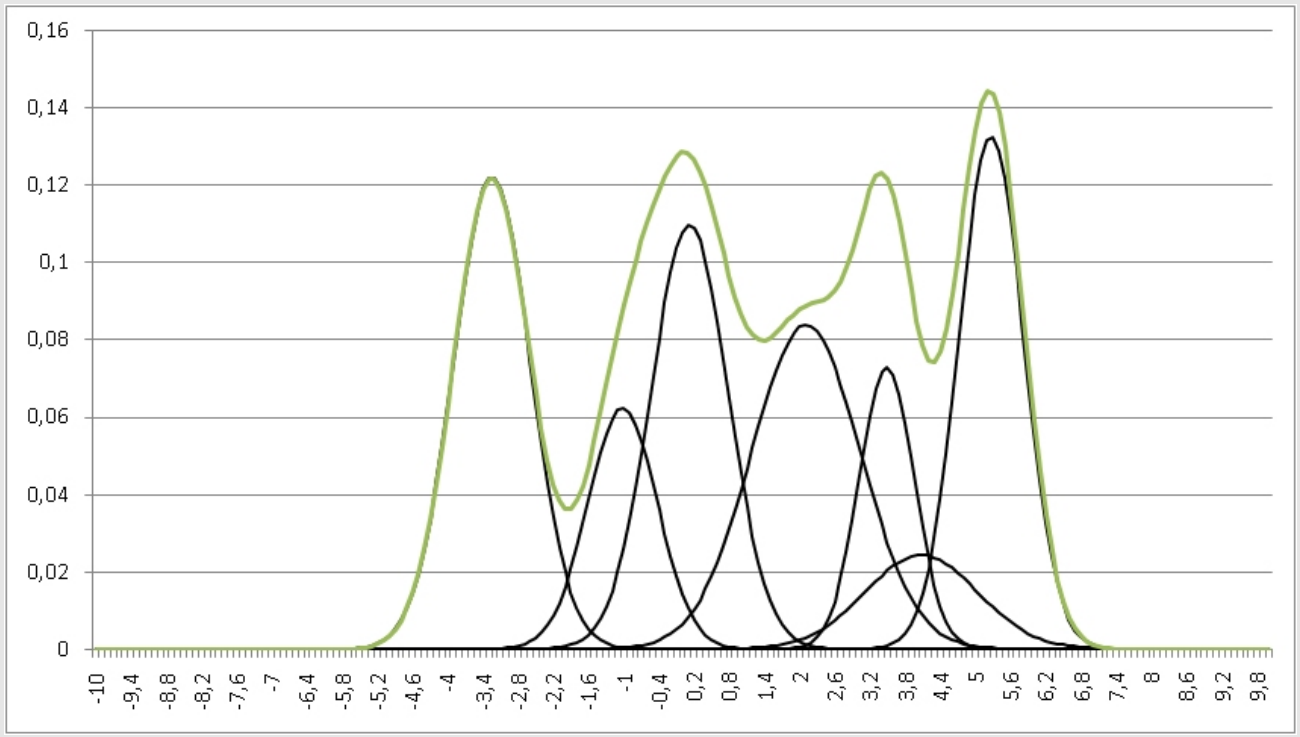
Vorläufige Ergebnisse - 2



Clustering Approximation (7 Komponenten) - Fehler: 1,18%

Funktionsapproximation

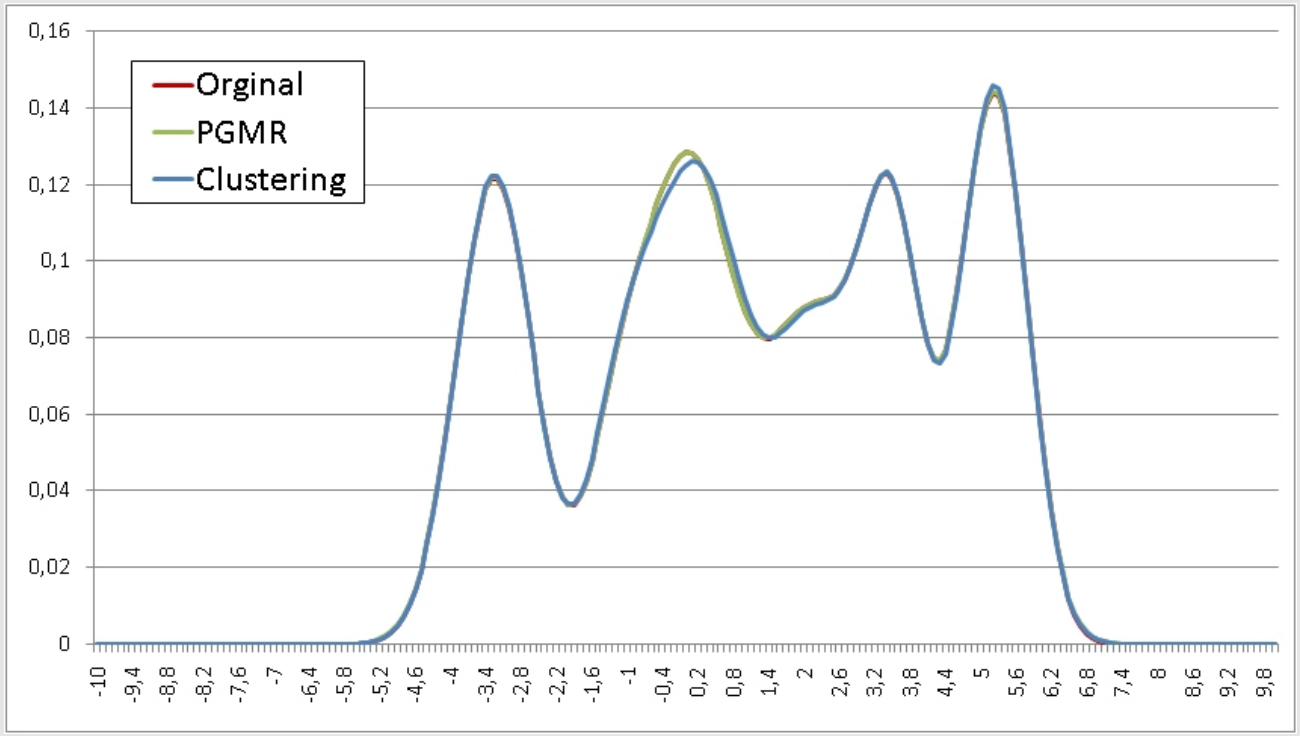
Vorläufige Ergebnisse - 2



PGMR Approximation (7 Komponenten) - Fehler: 0,22%

Funktionsapproximation

Vorläufige Ergebnisse - 2

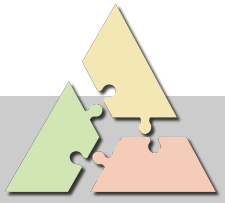


Vergleich

– III –

Schlussteil

Zusammenfassung und Ausblick



Zusammenfassung und Ausblick

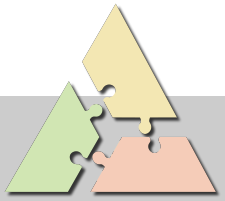
Zusammenfassung

Gebietsüberwachung

- Beweis der Schwere des Problems: NP vollständig
- Schneller Approximationsalgorithmus

Funktionsapproximation

- Blickwinkel der Algorithmik: Clustering-Ansatz
- erste Ergebnisse, noch unbefriedigend



Zusammenfassung und Ausblick

Ausblick

Gebietsüberwachung

- Implementierung des Algorithmus
- Verallgemeinerung auf beliebige (konvexe) Sensorformen und allgemeinere Metriken (David Steurer - Princeton University)

Funktionsapproximation

- Verbesserung der Approximationsergebnisse
- Verbesserung der Anfangslösung: hierarchischer Ansatz
- Beschleunigung der Iterationen: andere Distanzfunktionen

Weitere Gebiete

Referenzen

[GrötschelLoSc81] M. Grötschel, L. Lovász, and A. Schrijver, "The ellipsoid method and its consequences in combinatorial optimization", in *Combinatorica 1 (1981) no. 2*, 1981

[MasuyamaIbHa81] S. Masuyama, T. Ibaraki, and T. Hasegawa, "The computational complexity of the m-center problems on the plane", in *IEICE Transaction 64 (1981) no. 2*, 1981

[HuberHa08] M. Huber and U. D. Hanebeck, "Progressive Gaussian Mixture Reduction", in *Proceedings of the 11th International Conference on Information Fusion (Fusion 2008)*, Cologne, Germany, July, 2008