Grundlagen

Ziel

Konstruktion einer Hierarchie von Clustern (meist repräsentiert durch ein sog. *Dendrogramm*), so dass immer die Cluster mit minimaler Distanz verschmolzen werden

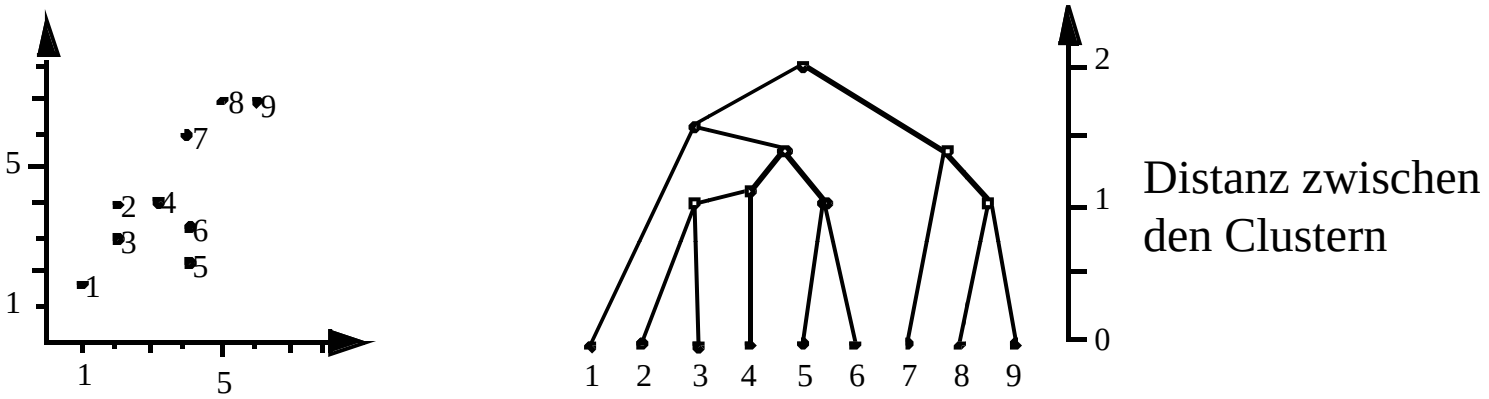
Dendrogramm

ein Baum, dessen Knoten jeweils einen Cluster repräsentieren, mit folgenden Eigenschaften:

- die Wurzel repräsentiert die ganze DB
- die Blätter repräsentieren einzelne Objekte
- ein innerer Knoten repräsentiert einen Cluster bestehend aus allen Objekten des darunter liegenden Teilbaums

Grundlagen

Beispiel eines Dendrogramms



Typen von hierarchischen Verfahren (meistens best-first Heuristiken)

- Bottom-Up Konstruktion des Dendrogramms (*agglomerative*)
- Top-Down Konstruktion des Dendrogramms (*divisive*)

Agglomeratives hierarchisches Clustering

1. Bilde initiale Cluster, die anfangs jeweils aus einem Objekt bestehen, und bestimme die Distanzen zwischen allen Paaren dieser Cluster.
2. Bilde einen neuen Cluster aus den zwei Clustern, welche die geringste Distanz zueinander haben.
3. Bestimme die Distanz zwischen dem neuen Cluster und allen anderen Clustern.
4. Wenn sich alle Objekte in einem einzigen Cluster befinden: Fertig, andernfalls wiederhole ab Schritt 2.

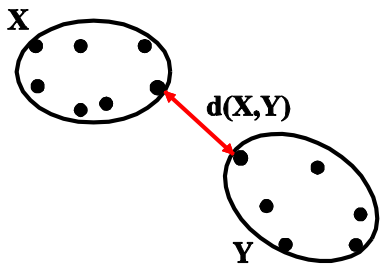
Distanzfunktionen für Cluster

- Die Verfahren unterscheiden sich anhand ihrer Distanzfunktionen für Cluster.
- Festlegung auf bestimmte Distanz macht oft algorithmische Finessen möglich.
- Beliebte Distanzfunktionen:
  - Single Link
  - Complete Link
  - Average Link – UPGMA (Unweighted Pair Group Method with Arithmetic Mean)
- Im Folgenden gegeben:
  - Distanzfunktion $dist(x,y)$ für Paare von Objekten
  - Seien X, Y Cluster, d.h. Mengen von Objekten.

Single-Link Distanz

Definition:

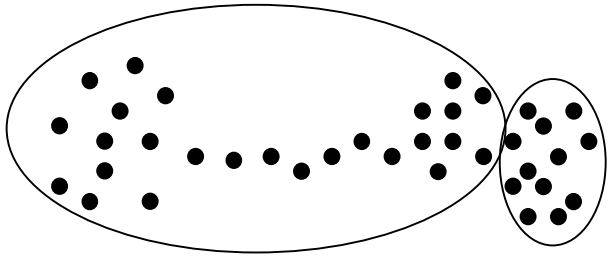
$$distSL(X,Y) = \min_{x \in X, y \in Y} dist(x,y)$$



Single-Link

Eigenschaften:

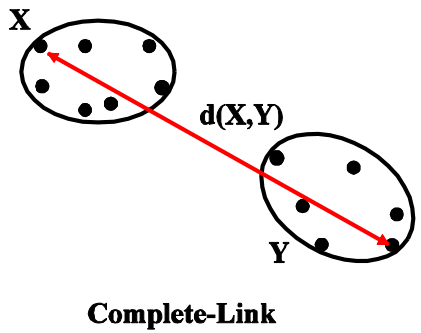
- Effiziente Implementierung (z.B. SLINK): $O(n^2)$
- Single-Link Effekt: „kettenförmige“ Cluster, Cluster werden durch wenige, kettenförmig verteilte Objekte vereinigt
  - Cluster mit starker Streuung
  - Cluster mit langgezogener Struktur



Complete-Link Distanz

Definition:

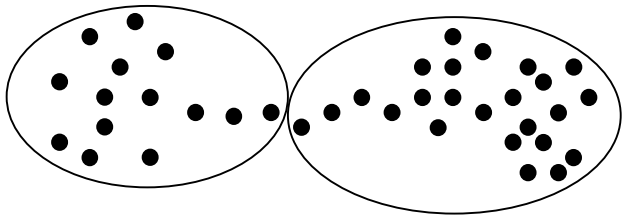
$$distCL(X,Y) = \max_{x \in X, y \in Y} dist(x,y)$$



Complete-Link

Eigenschaften:

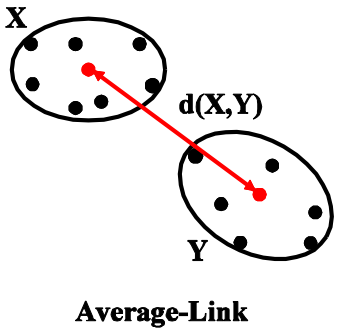
- Effiziente Implementierung (z.B. CLINK): $O(n^2)$
- Complete-Link Effekt
  - Kleine, stark abgegrenzte Cluster
  - Gleichgroße, konvexe Cluster



Average-Link Distanz

Definition:

$$distAL(X,Y) = \frac{1}{|X| \cdot |Y|} \cdot \sum_{x \in X, y \in Y} dist(x,y)$$



Eigenschaften:

- Kompromiss zwischen Single- und Complete-Link Ansatz

Diskussion

+ erfordert keine Kenntnis der Anzahl *k* der Cluster
+ findet nicht nur ein flaches Clustering, sondern eine ganze Hierarchie
+ ein einzelnes Clustering kann aus dem Dendrogramm gewonnen werden, z.B. mit Hilfe eines horizontalen Schnitts durch das Dendrogramm (erfordert aber wieder Anwendungswissen: wo ist ein sinnvoller Schnitt?)

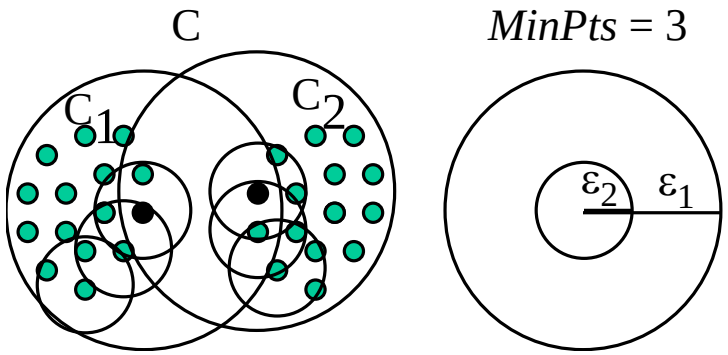
- Entscheidungen können nicht zurückgenommen werden (best first search)
- Single-Link-Effekte, Complete-Link-Effekte
- Ineffizienz: Laufzeitkomplexität von mindestens $O(n^2)$ für n Objekte

Hierarchische Verfahren

Dichtebasiertes hierarchisches Clustering

[Ankerst, Breunig, Kriegel & Sander 1999]

- für einen konstanten *MinPts*-Wert sind dichte-basierte Cluster bzgl. eines kleineren ε vollständig in Clustern bzgl. eines größeren ε enthalten



- in einem DBSCAN-ähnlichen Durchlauf gleichzeitig das Clustering für verschiedene Dichte-Parameter bestimmen
- zuerst die dichteren Teil-Cluster, dann den dünneren Rest-Cluster
- kein Dendrogramm, sondern eine auch noch bei sehr großen Datenmengen übersichtliche Darstellung der Cluster-Hierarchie
- Clustermodell: Dichte

Grundbegriffe

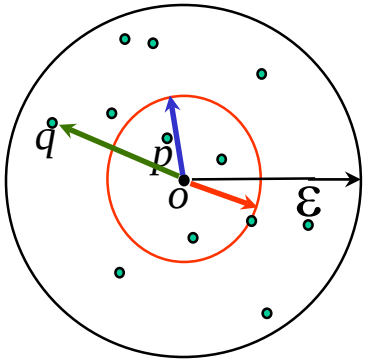
Kerndistanz eines Objekts *o* bzgl. ε und *MinPts*

$$Kerndistanz_{\varepsilon,MinPts}(o) = \begin{cases} UNDEFINIERT, & wenn\ |RQ(o,\varepsilon)| < MinPts \\ MinPtsDistanz(o), & sonst \end{cases}$$

Erreichbarkeitsdistanz eines Objekts *p* relativ zu einem Objekt *o*

$$Erreichbarkeitsdistanz_{\varepsilon,MinPts}(p,o) = \begin{cases} UNDEFINIERT & wenn\ |RQ(o,\varepsilon)| < MinPts \\ \max\{Kerndistanz(o), dist(o,p)\}, & sonst \end{cases}$$

MinPts = 5



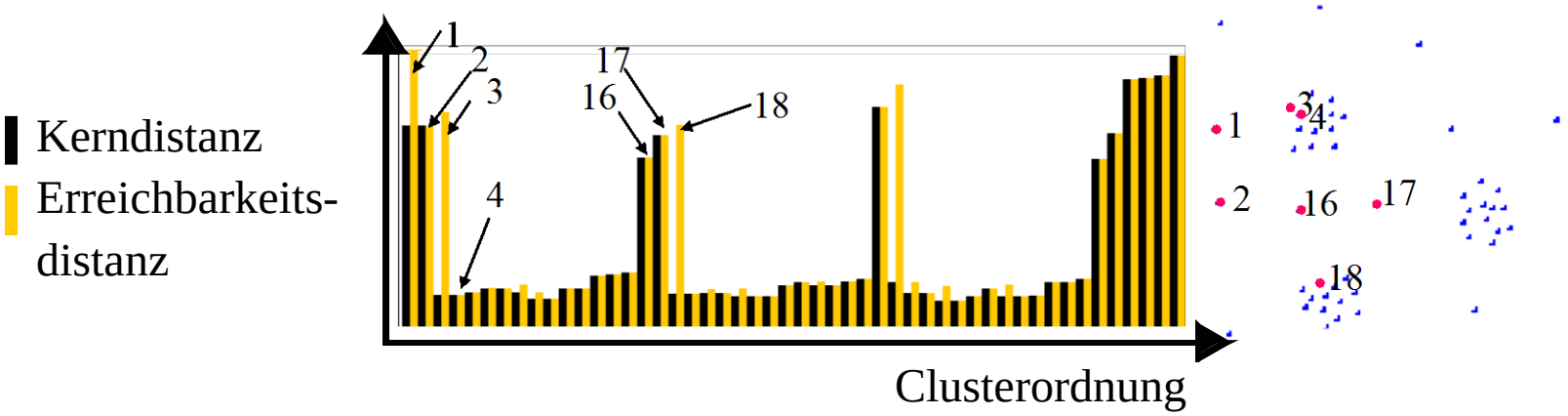
→ *Kerndistanz*(*o*)

→ *Erreichbarkeitsdistanz*(*p*,*o*)

→ *Erreichbarkeitsdistanz*(*q*,*o*)

Clusterordnung

- OPTICS liefert nicht direkt ein (hierarchisches) Clustering, sondern eine „Clusterordnung“ bzgl. ε und *MinPts*
- *Clusterordnung* bzgl. ε und *MinPts*
  - beginnt mit einem beliebigen Objekt
  - als nächstes wird das Objekt besucht, das zur Menge der bisher besuchten Objekte die minimale Erreichbarkeitsdistanz besitzt



Algorithmus OPTICS

Datenstrukturen

- SeedList
  - speichert Punkte mit „aktueller“ Erreichbarkeitsdistanz aufsteigend sortiert
- ClusterOrder
  - resultierende Clusterordnung wird schrittweise aufgebaut

Hauptschleife:

```
SeedList = ∅;
WHILE es gibt noch unmarkierte Objekte in DB DO
   IF SeedList = ∅
   THEN
     füge beliebiges noch unmarkiertes Objekt in ClusterOrder ein mit
     Erreichbarkeitsdistanz ∞;
   ELSE
     füge erstes Objekt aus der SeedList mit aktueller Erreichbarkeitsdistanz in
     ClusterOrder ein;
   // sei obj das zuletzt in ClusterOrder eingefügte Objekt
   markiere obj als bearbeitet;
   FOR ALL neighbor ∈ RQ(obj, ε) DO
     // insert/update neighbor in SeedList mit referenzobjekt obj
     SeedList.update(neighbor, obj);
```

Algorithmus OPTICS

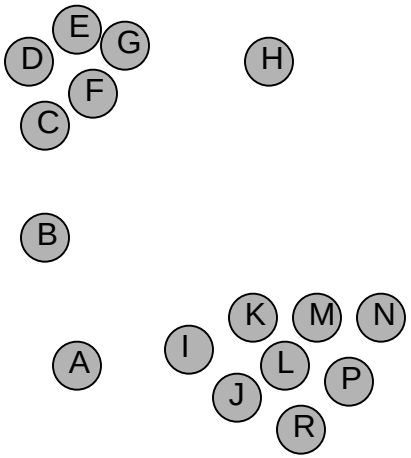
Einfügen/Updaten eines Objekts *o* in SeedList

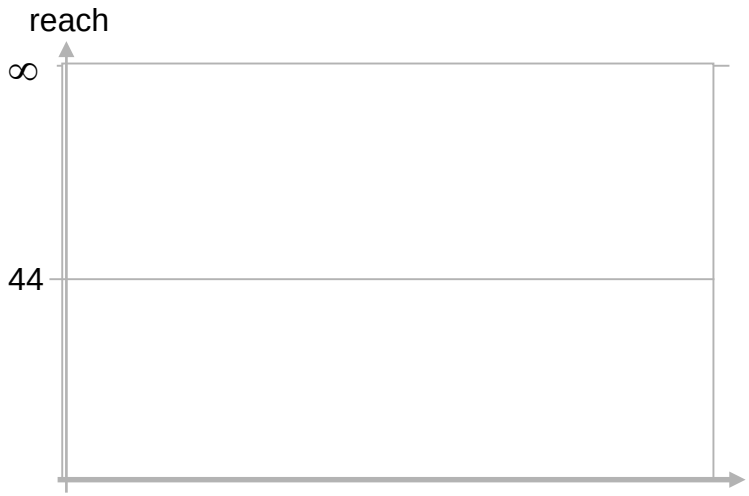
- Beachte: Für alle Objekte *p* in SeedList ist die „aktuelle“ Erreichbarkeitsdistanz *p*.rdist gespeichert.
- SeedList ist nach *p*.rdist aufsteigend sortiert (als Heap organisiert)
- Referenzobjekt: *obj*

```
SeedList :: update(o, obj)
Berechne Erreichbarkeitsdistanz_{ε,MinPts}(o, obj) =: rdistneu_o;
IF o ist bereits in SeedList THEN
   IF rdistneu_o ≤ o.rdist THEN
     o.rdist := rdistneu_o;
     verschiebe o in SeedList (nach vorne); // aufsteigen im Heap
ELSE
   // o ist noch nicht in SeedList => normales Einfügen in Heap
   füge o mit o.rdist := rdistneu_o in SeedList ein;
```

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

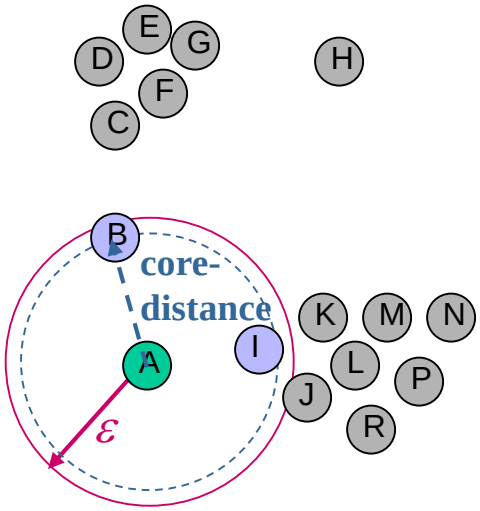


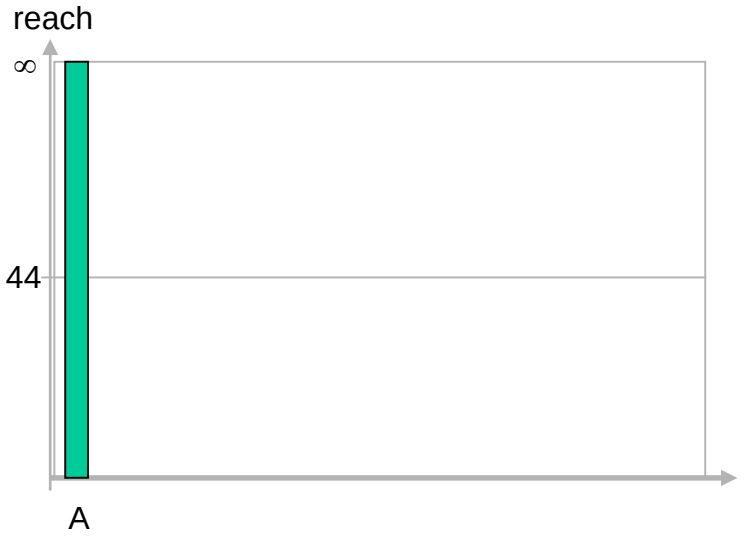


seed list:

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

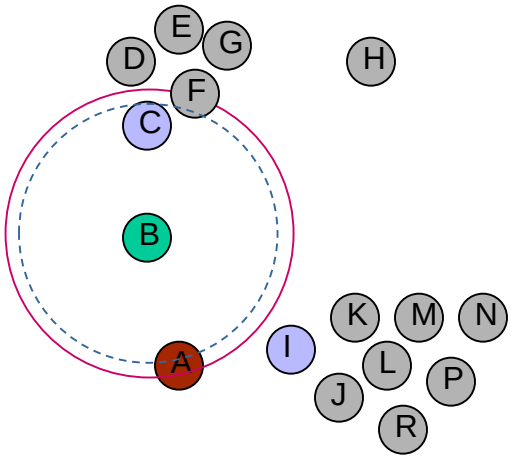


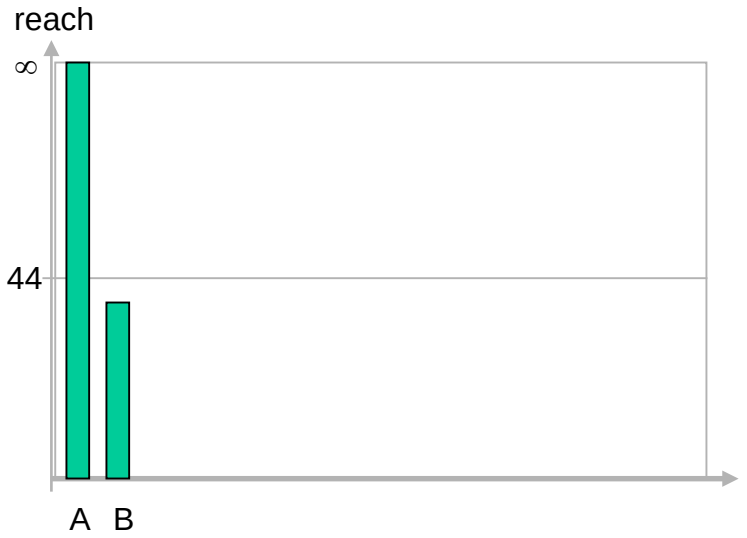


seed list: (B,40) (I, 40)

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

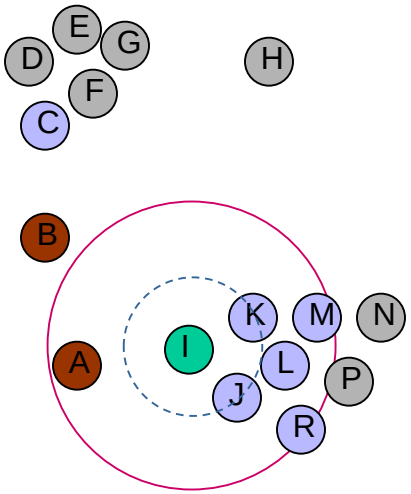




seed list: (I, 40) (C, 40)

Hierarchische Verfahren

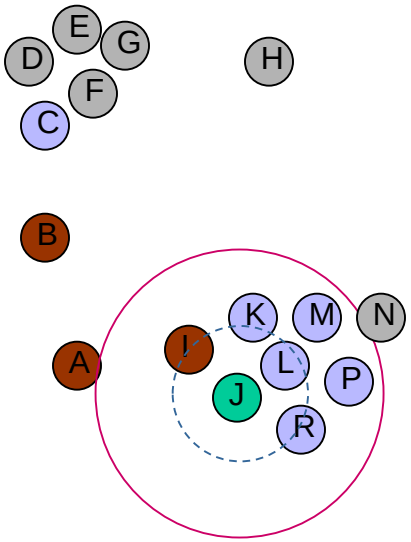
- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

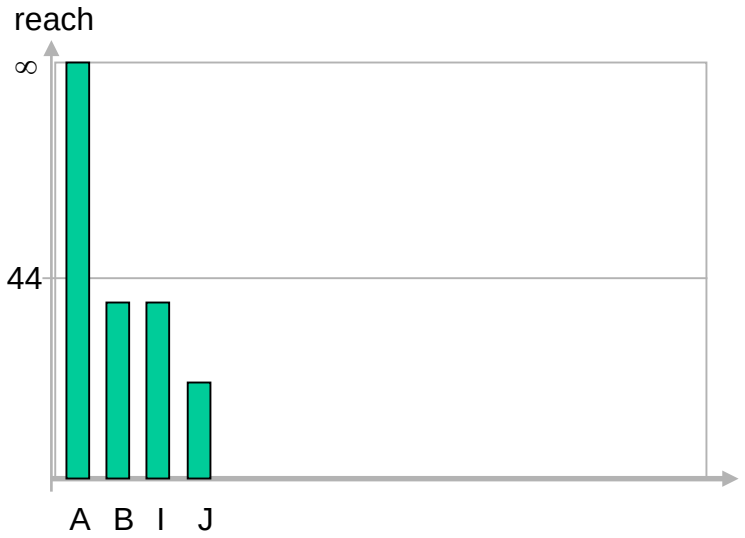


seed list: (J, 20) (K, 20) (L, 31) (C, 40) (M, 40) (R, 43)

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

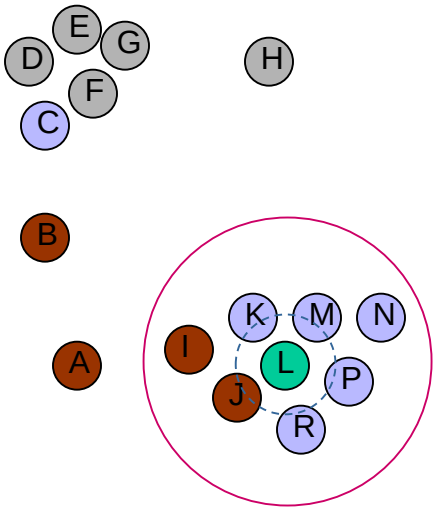


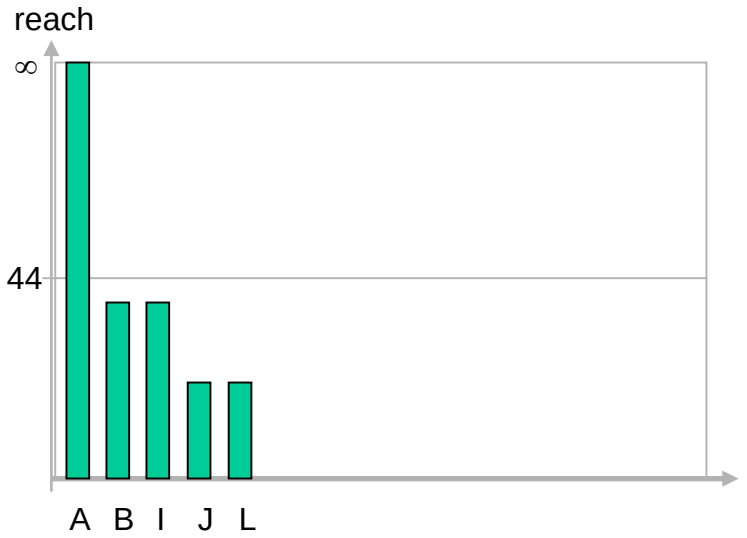


seed list: (L, 19) (K, 20) (R, 21) (M, 30) (P, 31) (C, 40)

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

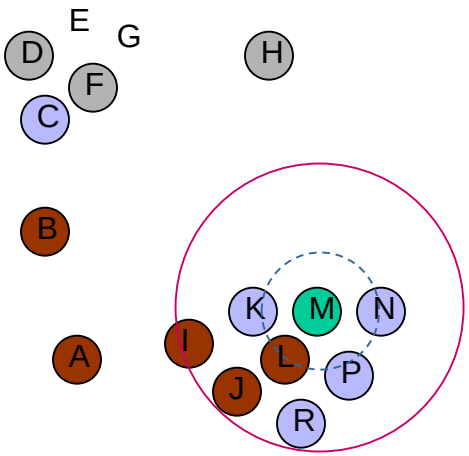


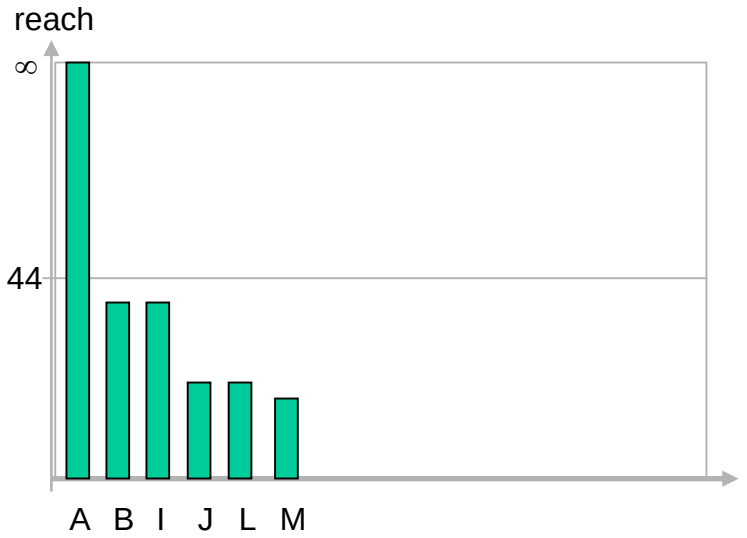


seed list: (M, 18) (K, 18) (R, 20) (P, 21) (N, 35) (C, 40)

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

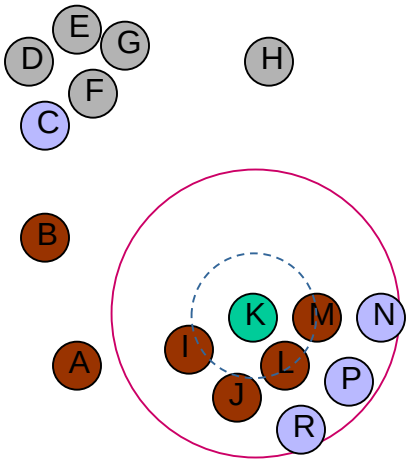


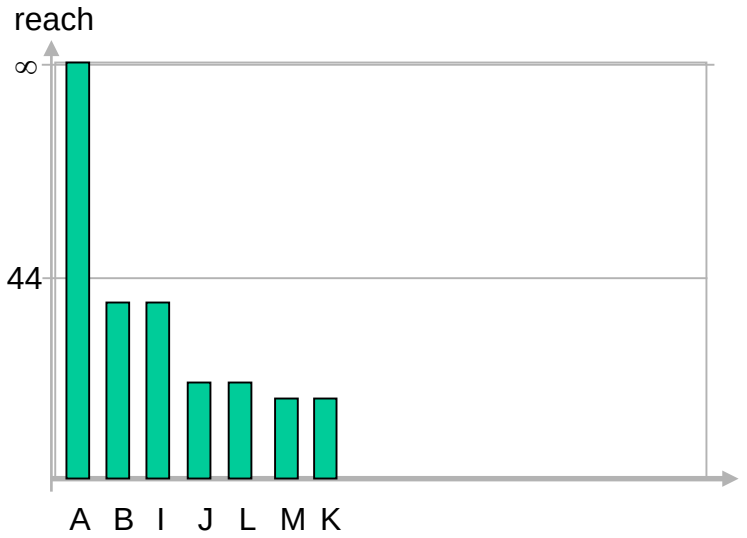


seed list: (K, 18) (N, 19) (R, 20) (P, 21) (C, 40)

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

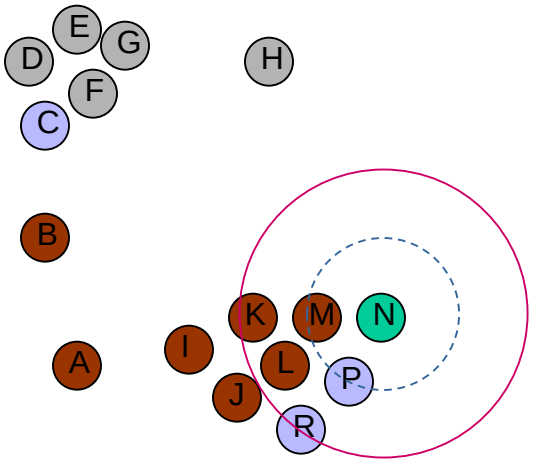


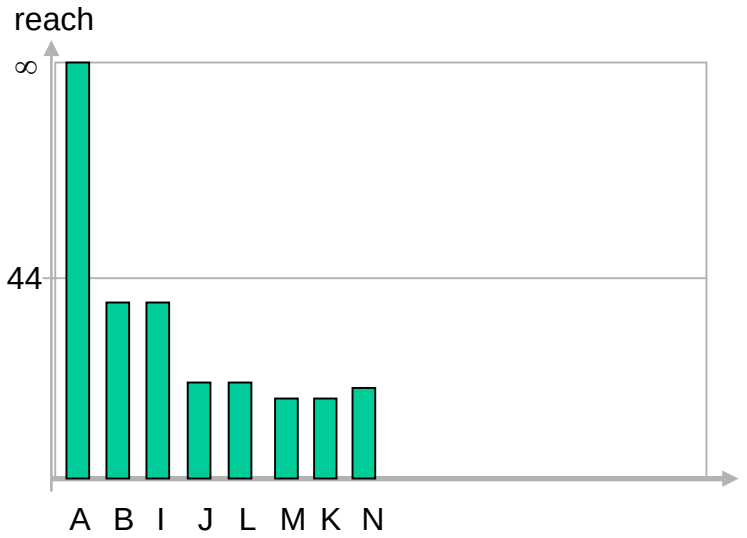


seed list: (N, 19) (R, 20) (P, 21) (C, 40)

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

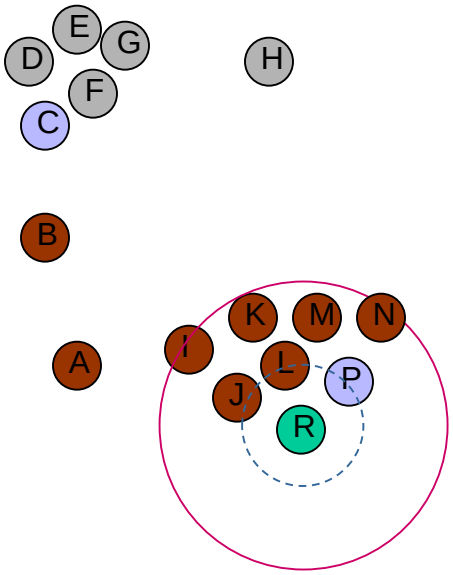


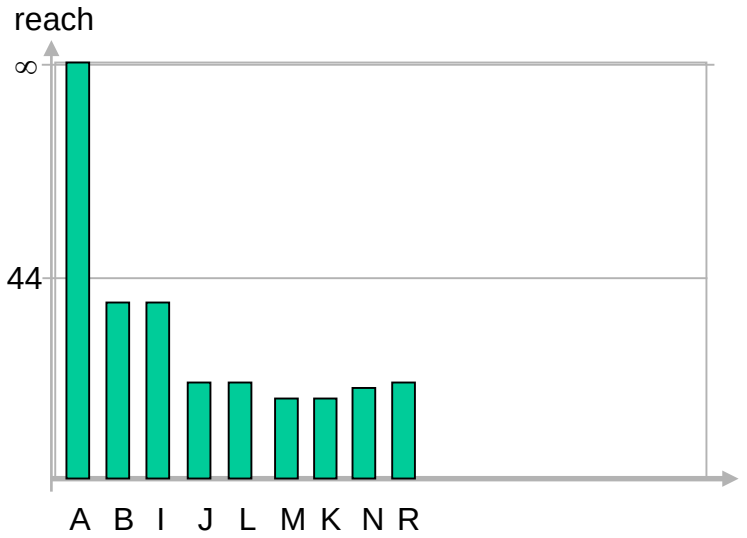


seed list: (R, 20) (P, 21) (C, 40)

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

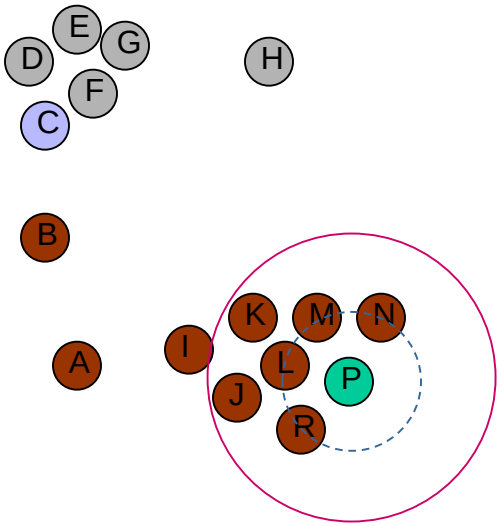




seed list: (P, 21) (C, 40)

Hierarchische Verfahren

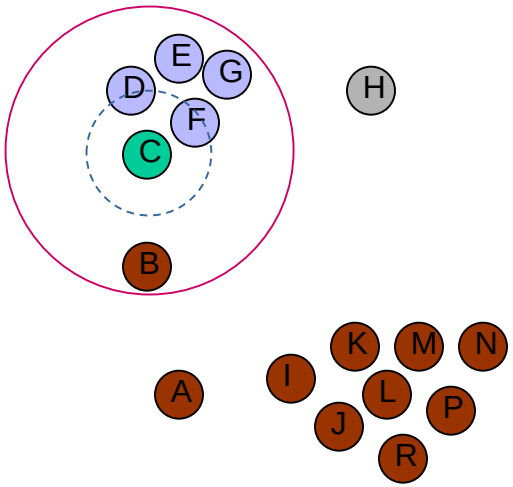
- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

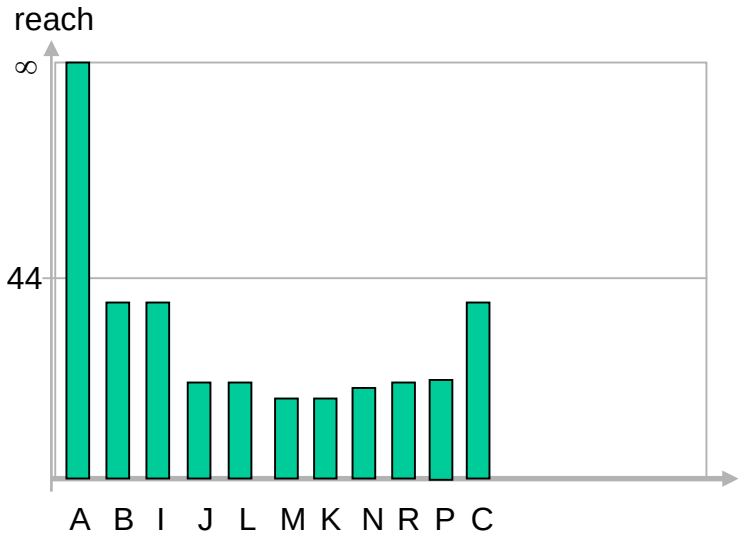


seed list: (C, 40)

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

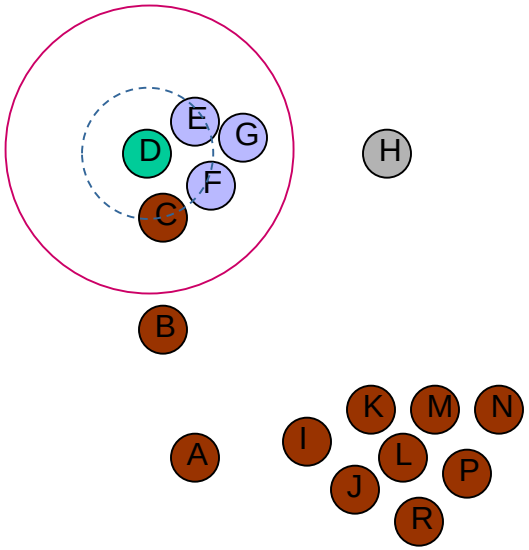


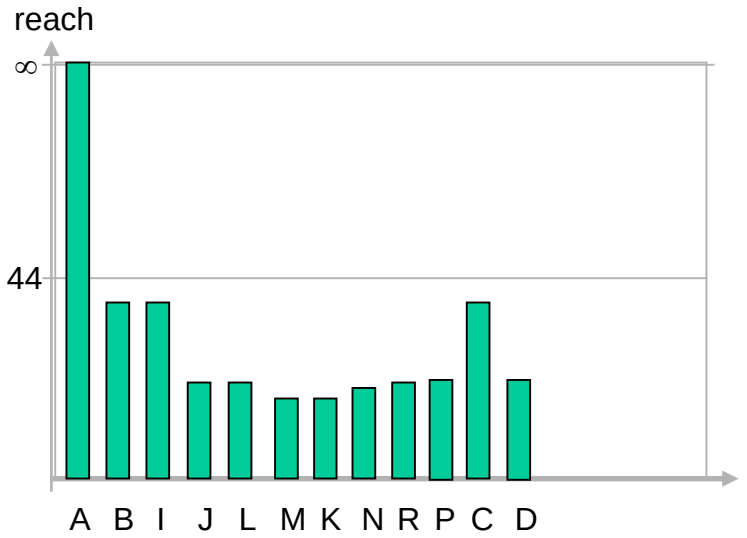


seed list: (D, 22) (F, 22) (E, 30) (G, 35)

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

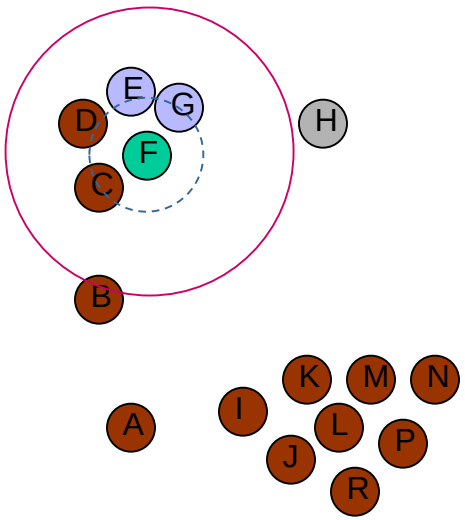


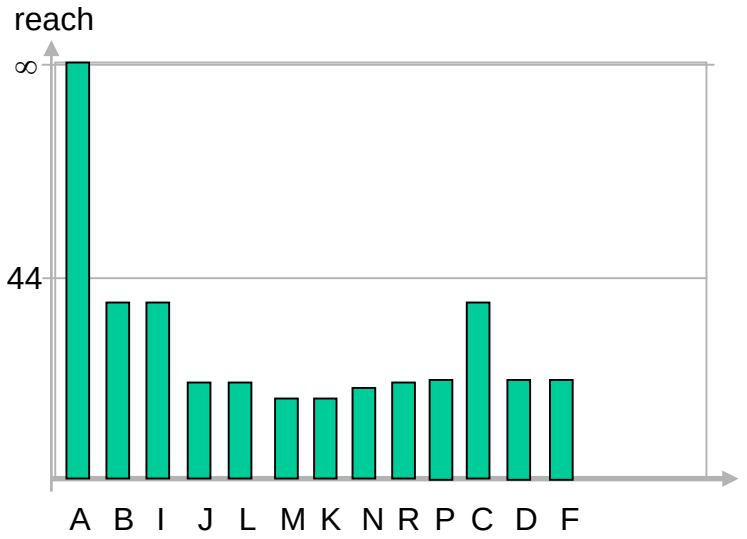


seed list: (F, 22) (E, 22) (G, 32)

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

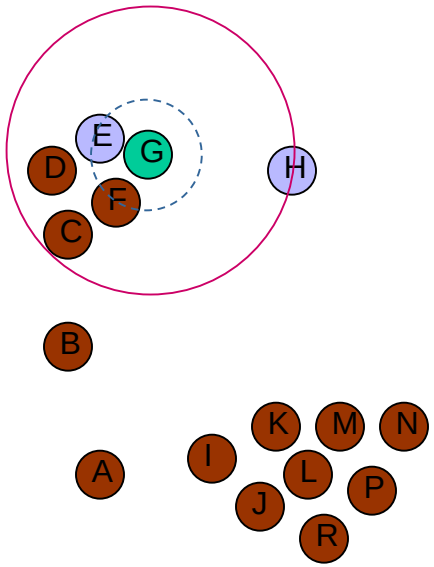


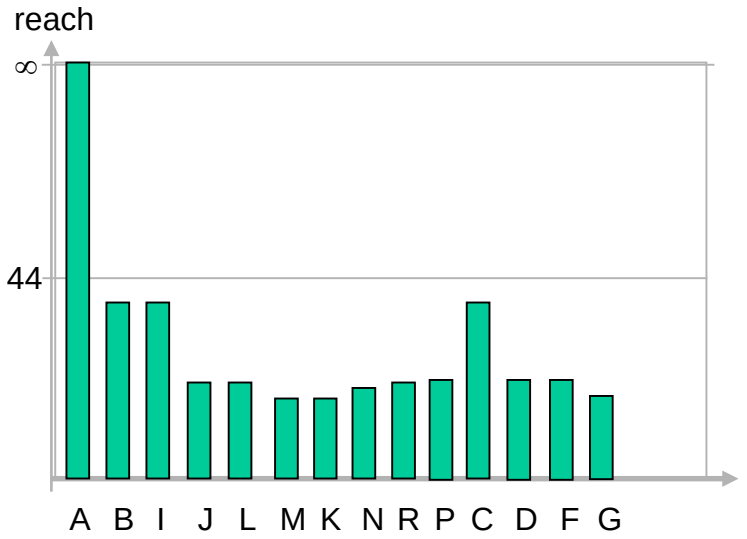


seed list: (G, 17) (E, 22)

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

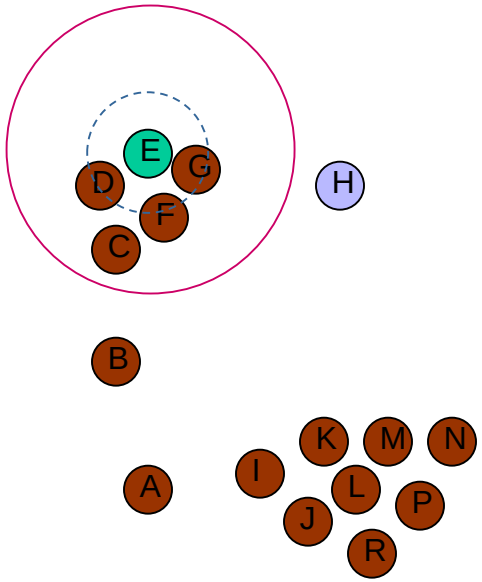


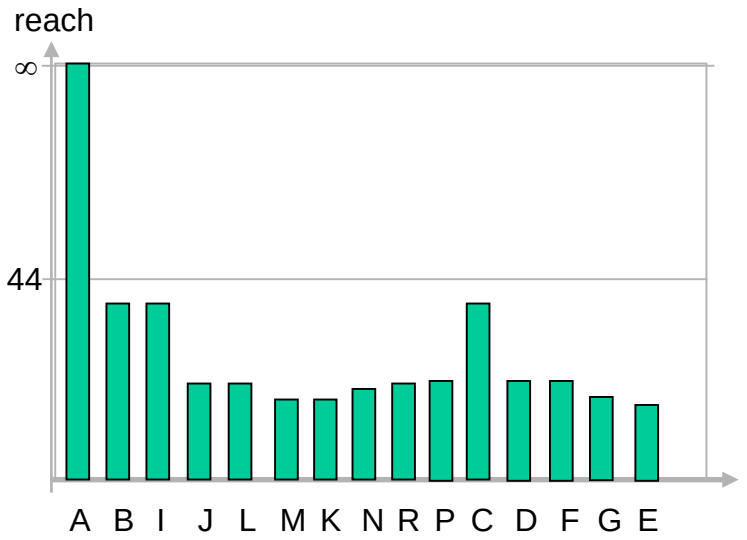


seed list: (E, 15) (H, 43)

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

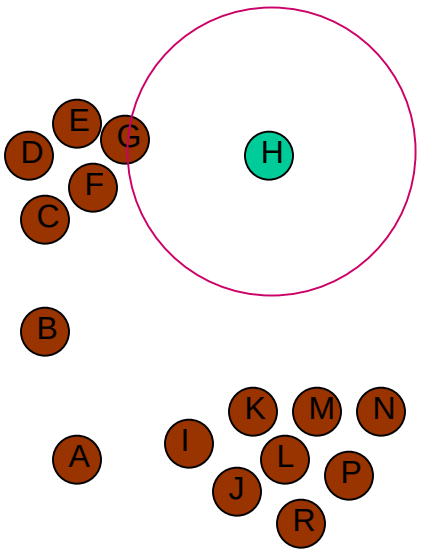


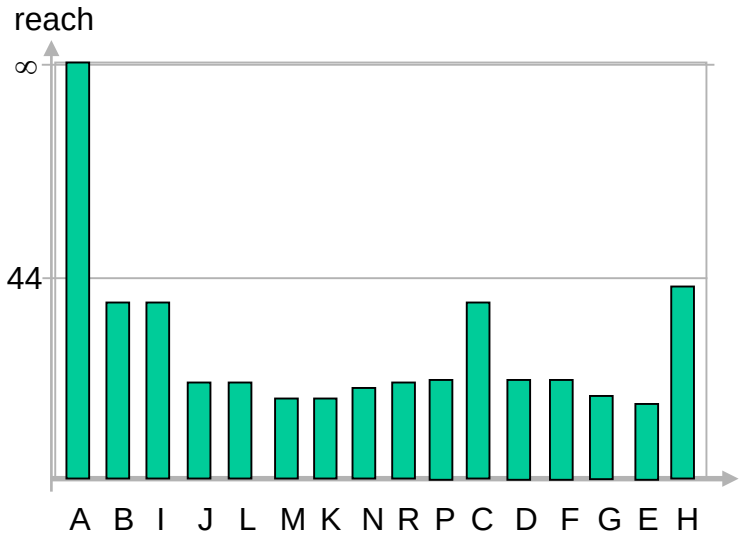


seed list: (H, 43)

Hierarchische Verfahren

- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3

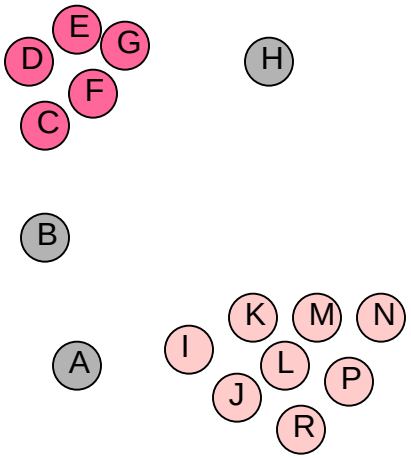


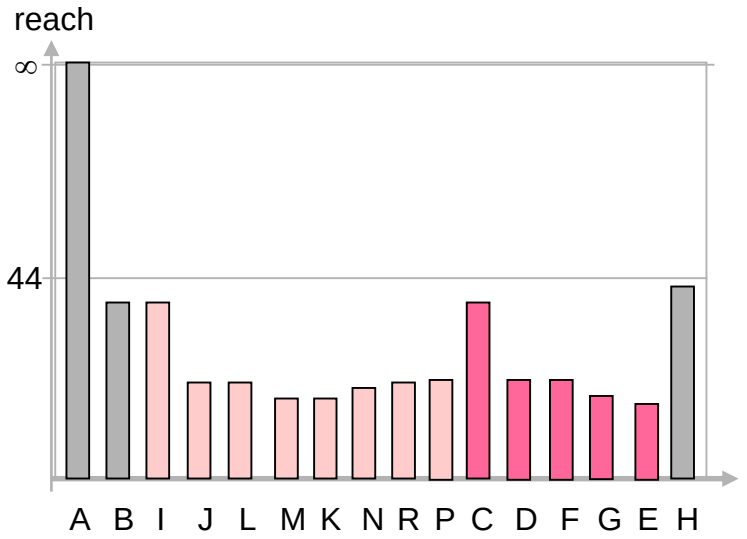


seed list: -

Hierarchische Verfahren

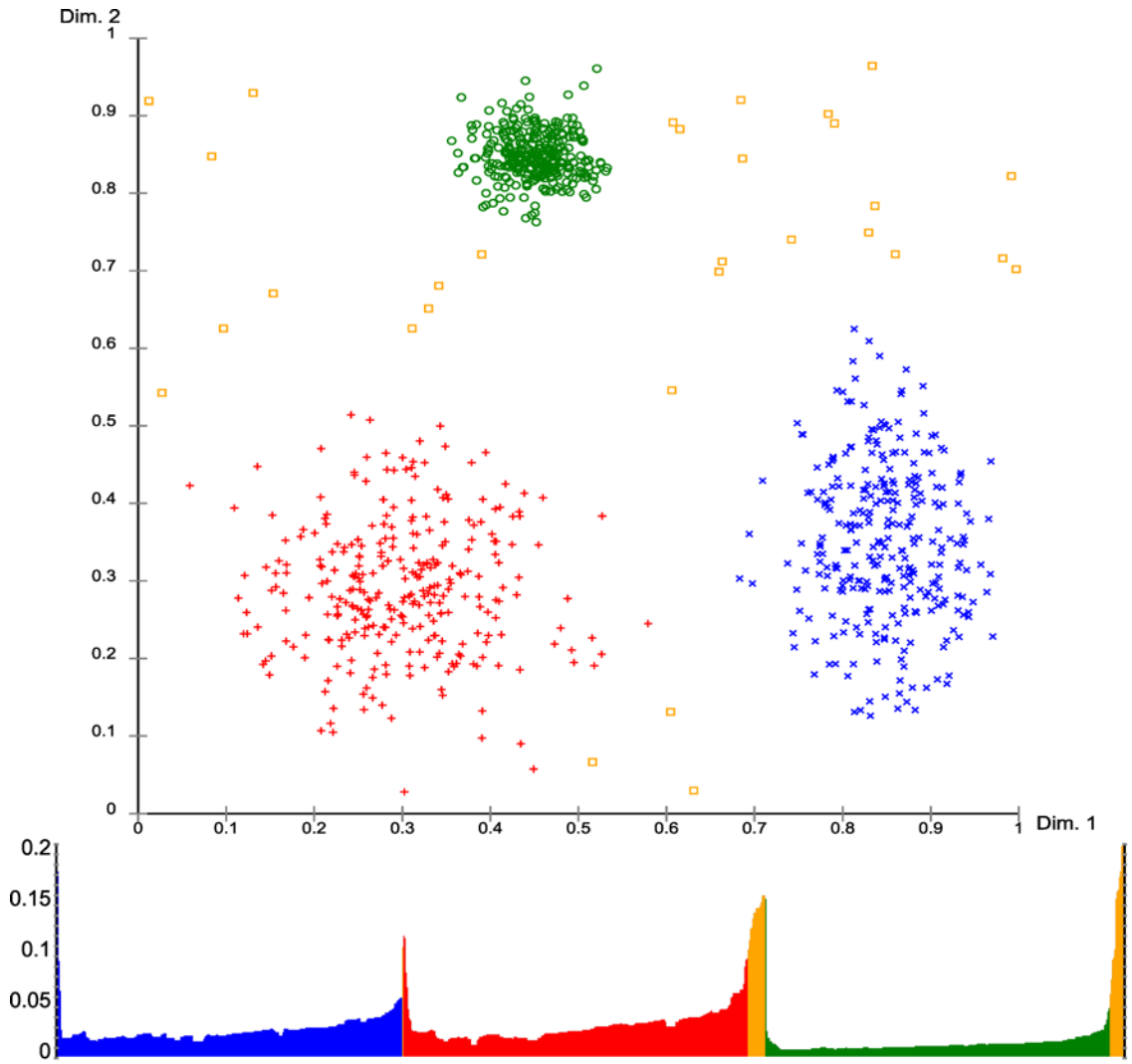
- Example Database (2-dimensional, 16 points)
- ε= 44, *MinPts* = 3





Hierarchische Verfahren

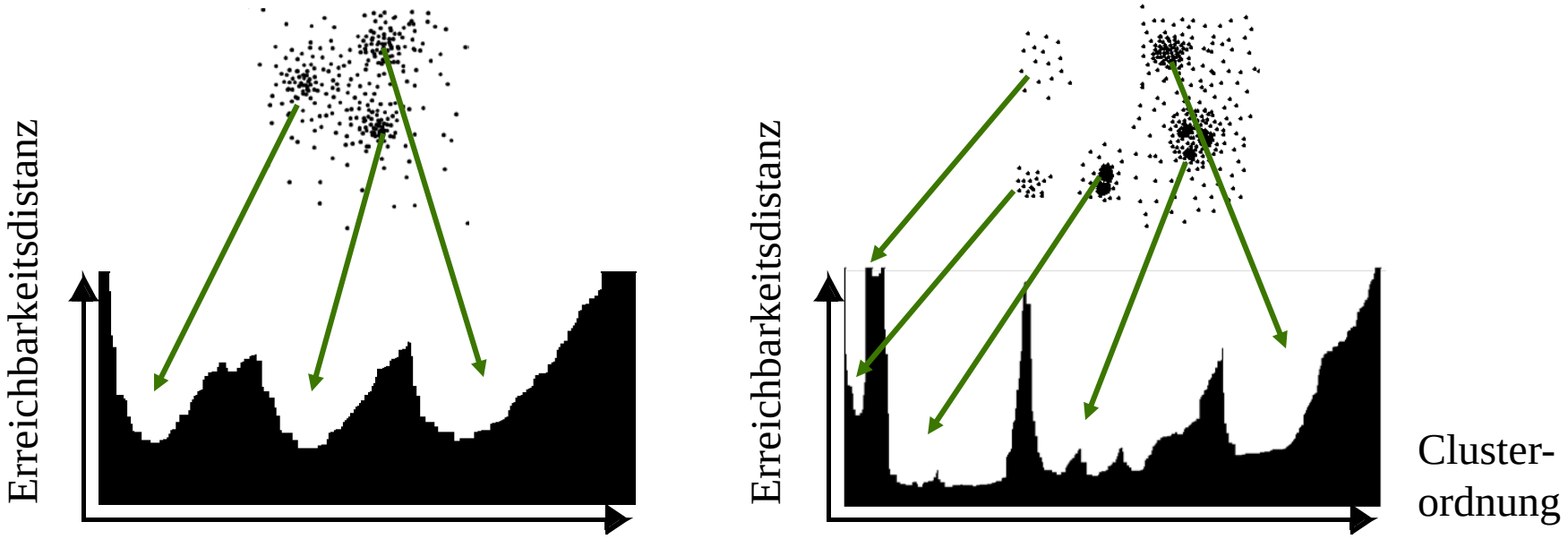
Erreichbarkeits-Diagramm:



Erreichbarkeits-Diagramm

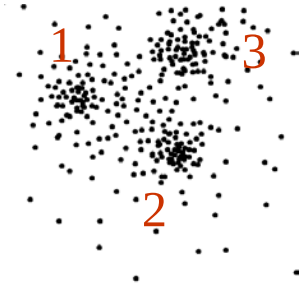
Zeigt die Erreichbarkeitsdistanzen (bzgl. ε und *MinPts*) der Objekte als senkrechte, nebeneinanderliegende Balken

in der durch die Clusterordnung der Objekte gegebenen Reihenfolge

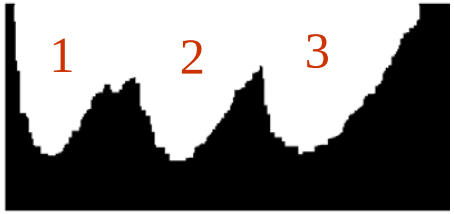


Schnitt: ε für DBSCAN: Dichte-Niveau

Parameter-Sensitivität



MinPts = 10, ε = 10

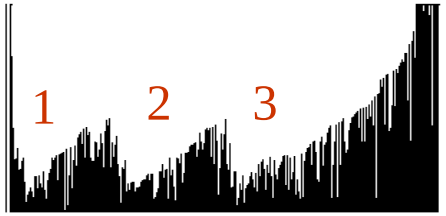


optimale Parameter

MinPts = 10, ε = 5

kleineres ε

MinPts = 2, ε = 10



kleineres *MinPts*

Clusterordnung ist robust gegenüber den Parameterwerten

gute Resultate wenn Parameterwerte „groß genug“

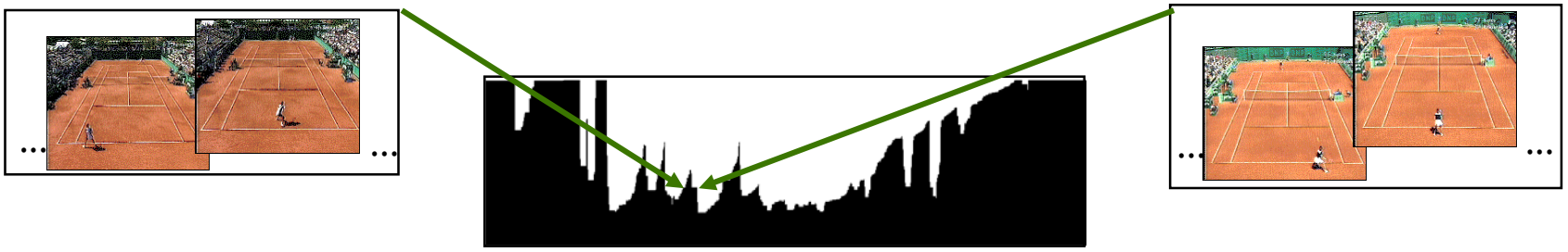
Heuristische Parameter-Bestimmung

ε

- wähle größte *MinPts*-Distanz aus einem Sample oder
- berechne durchschnittliche *MinPts*-Distanz für gleichverteilte Daten

MinPts

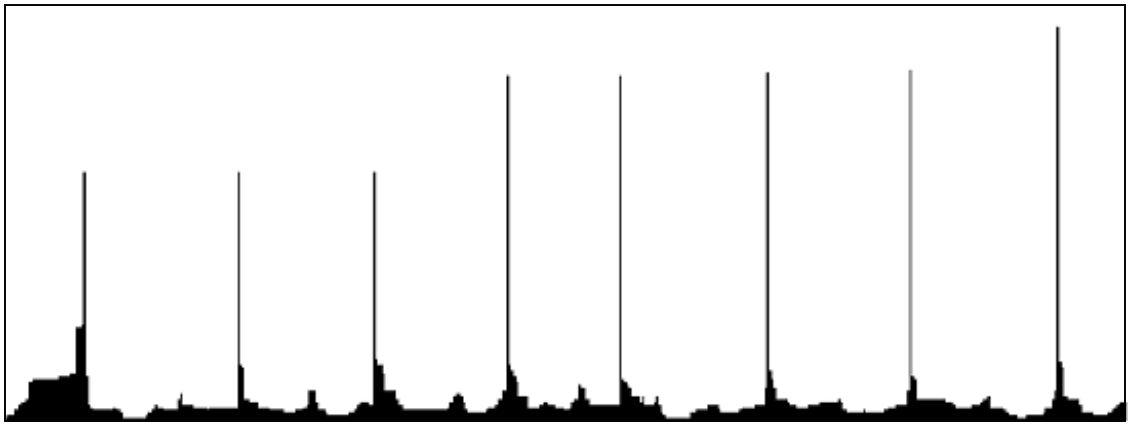
- glätte Erreichbarkeits-Diagramm
- vermeide "single-" bzw. "*MinPts*-link" Effekt



Manuelle Analyse der Cluster

Mit Erreichbarkeits-Diagramm

- gibt es Cluster?
- wieviele Cluster?
- sind die Cluster hierarchisch geschachtelt?
- wie groß sind die Cluster?



Erreichbarkeits-Diagramm

Automatisches Entdecken von Clustern

- ξ-Cluster [Ankerst, Breunig, Kriegel, Sander 99]

  Cluster = Teilsequenz der Clusterordnung mit mindestens *MinPts* Punkte
  - beginnt in einem Gebiet x-steil *abfallender* Erreichbarkeitsdistanzen
  - endet in einem Gebiet x-steil *steigender* Erreichbarkeitsdistanzen bei etwa demselben absoluten Wert
- ClusterTree [Sander, Qin, Lu, Niu, Kovarsky 03]

  Cluster sind geteilt durch "signifikante" lokale Maxima
- "Hierarchical" DBSCAN [Campello, Moulavi, Sander 13]

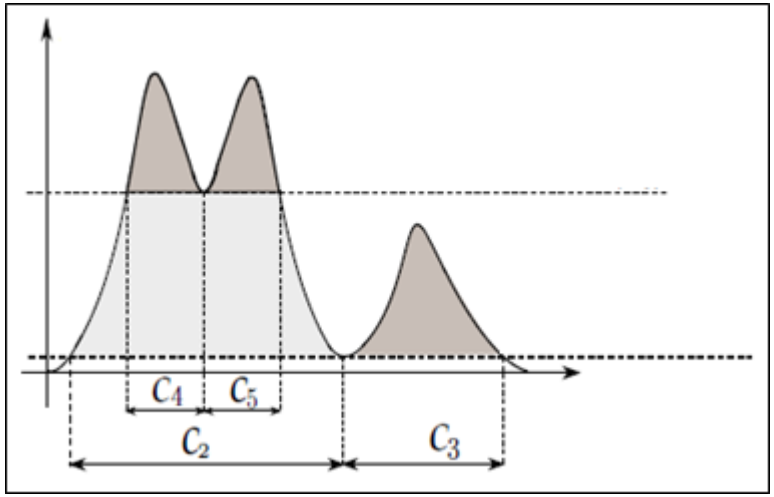
  formuliert Kriterium der Cluster-Stabilität, optimiert die Gesamt-Stabilität der durch lokale Schnitte aus der Hierarchie extrahierten Lösung
- Framework zur Extraktion von Clustern aus einer Cluster-Hierarchie [Campello, Moulavi, Zimek, Sander 13]

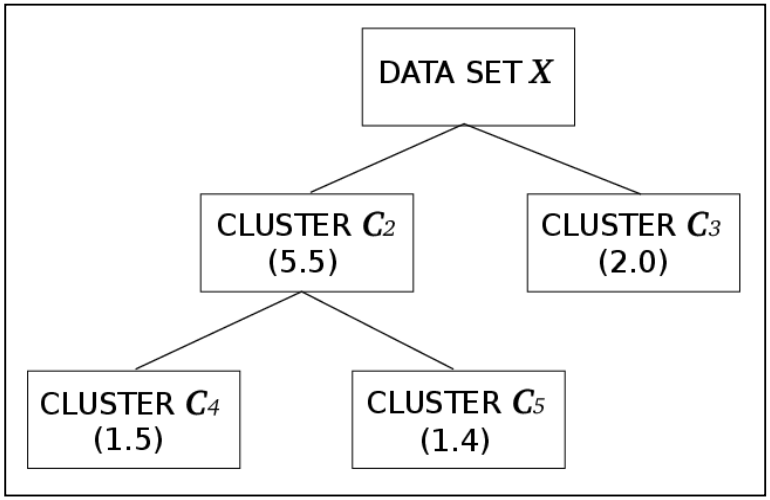
  finden global-optimaler lokaler Schnitte in der Hierarchie, basierend auf zusätzlichen Kriterien (z.B. Cluster-Qualität, Constraints) – HDBSCAN* wird zum Spezialfall mit Kriterium der Cluster-Stabilität

Hierarchical DBSCAN

[Campello, Moulavi, Sander 13]

- Voraussetzung: DBSCAN* (Randpunkte gehören nicht zum Cluster)
- erzeugt Cluster-Hierarchie ähnlich OPTICS, aber vereinfacht die Hierarchie
  - Grundidee: wo ändern sich Cluster signifikant durch
    - Verlust von zu vielen Objekten (Cluster wird zu klein)
    - Verschmelzen mit anderem Cluster
  - dabei: Festhalten der Clusterstabilität



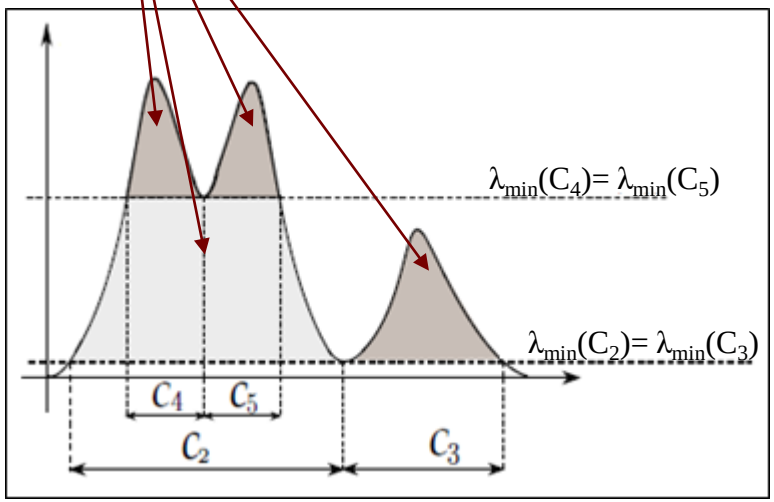


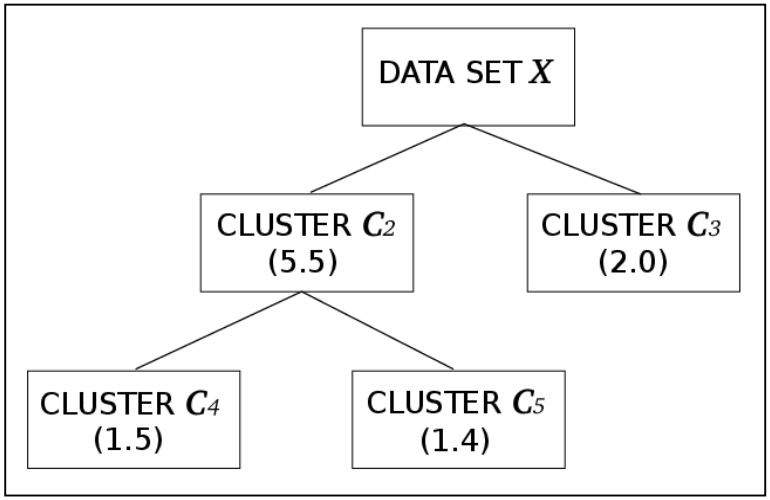
Hierarchische Verfahren

Hierarchical DBSCAN

- Clusterstabilität: Relative Excess of Mass
- betrachtet unterschiedliches Dichte-Profil von Objekten im Cluster

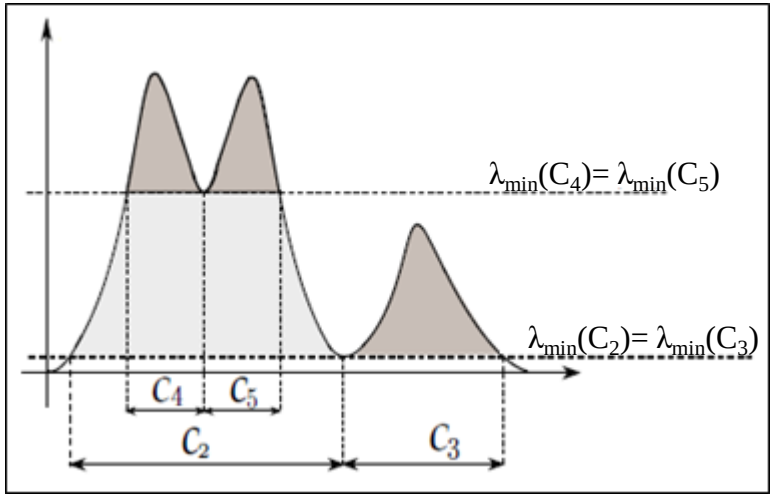
$$S(\mathbf{C}_i) = \sum_{\mathbf{x}_j \in \mathbf{C}_i} \left(\lambda_{max}(\mathbf{x}_j, \mathbf{C}_i) - \lambda_{min}(\mathbf{C}_i) \right) = \sum_{\mathbf{x}_j \in \mathbf{C}_i} \left(\frac{1}{\varepsilon_{min}(\mathbf{x}_j, \mathbf{C}_i)} - \frac{1}{\varepsilon_{max}(\mathbf{C}_i)} \right)$$

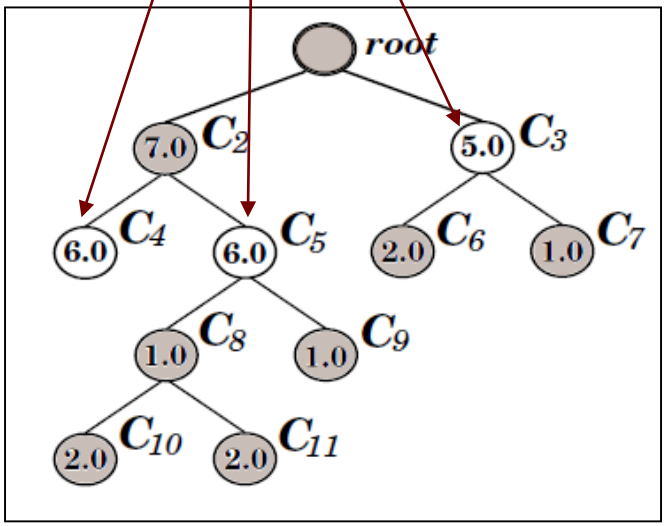




Hierarchical DBSCAN

- Ermöglicht Extraktion eines flachen, nicht-überlappenden Clustering
- Optimierung: Maximiere Gesamt-Cluster-Stabilität $(\sum S(C_i) \mid C_i \text{ Teil der Lösung})$
- Eltern-/Kind-Cluster schließen sich gegenseitig als Lösung aus
- Global-optimale Lösung durch dynamische Programmierung:
  lokale Schnitte in der Hierarchie





Hierarchische Verfahren

Literatur

M. Ankerst, M. M. Breunig, H.-P. Kriegel, J. Sander: OPTICS: Ordering Points To Identify the Clustering Structure. SIGMOD Conference 1999: 49-60.

R. J. G. B. Campello, D. Moulavi, J. Sander: Density-Based Clustering Based on Hierarchical Density Estimates. PAKDD 2013: 160-172.

R. J. G. B. Campello, D. Moulavi, A. Zimek, J. Sander: A framework for semi-supervised and unsupervised optimal extraction of clusters from hierarchies. Data Mining and Knowledge Discovery, 2013.

J. Sander, X. Qin, Z. Lu, N. Niu, A. Kovarsky: Automatic Extraction of Clusters from Hierarchical Clustering Representations. PAKDD 2003: 75-87.

Hierarchische Verfahren: Was haben Sie gelernt?

- Dendrogramm zur Repräsentation hierarchischer Clusterings
- Distanzen für Cluster und Objekte
- Dichte-basiertes hierarchisches Clustering
  - Formalisierung
  - Algorithmus OPTICS
  - Cluster-Ordnung (Erreichbarkeits-Diagramm)
    - Konstruktion
    - Interpretation
    - Parameter

Clustering: Was haben Sie gelernt?

- Clustering: Problem und Lösungsansätze
- Kategorien von Clustering-Verfahren:
  - partitionierende Verfahren
  - dichte-basiertes Clustering
  - hierarchisches Clustering
- Prinzipielle Unterschiede der Verfahren:
  - parametrisch vs. nicht-parametrisch
  - Abhängigkeit von Parametern (Anzahl Cluster, Dichte-Grenzwert)?
  - nicht-deterministisch (lokal optimierend) vs. deterministisch
  - definiert für euklidischen Vektorraum oder andere Räume?