OPTICS: Ordering Points To Identify the Clustering Structure

Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, Jörg Sander

Presented by Chris Mueller
November 4, 2004
Clustering

Goal: Group objects into meaningful subclasses as part of an exploratory process to insight into data or as a preprocessing step for other algorithms.

Clustering Strategies
- Hierarchical
- Partitioning
  - k-means
  - **Density Based**

Density Based clustering requires a distance metric between points and works well on high dimensional data and data that forms irregular clusters.
DBSCAN: Density Based Clustering

An object $p$ is in the $\epsilon$-neighborhood of $q$ if the distance from $p$ to $q$ is less than $\epsilon$.

A core object has at least $MinPts$ in its $\epsilon$-neighborhood.

An object $p$ is directly density-reachable from object $q$ if $q$ is a core object and $p$ is in the $\epsilon$-neighborhood of $q$.

An object $p$ is density-reachable from object $q$ if there is a chain of objects $p_1, \ldots, p_n$, where $p_1 = q$ and $p_n = p$ such that $p_{i+1}$ is directly density-reachable from $p_i$.

An object $p$ is density-connected to object $q$ if there is an object $o$ such that both $p$ and $q$ are density-reachable from $o$.

A cluster is a set of density-connected objects which is maximal with respect to density-reachability.

Noise is the set of objects not contained in any cluster.
OPTICS: Density-Based Cluster Ordering

OPTICS generalizes DB clustering by creating an ordering of the points that allows the extraction of clusters with arbitrary values for $\varepsilon$.

The **generating-distance** $\varepsilon$ is the largest distance considered for clusters. Clusters can be extracted for all $\varepsilon_i$ such that $0 \leq \varepsilon_i \leq \varepsilon$.

The **core-distance** is the smallest distance $\varepsilon'$ between $p$ and an object in its $\varepsilon$-neighborhood such that $p$ would be a core object.

The **reachability-distance** of $p$ is the smallest distance such that $p$ is density-reachable from a core object $o$.

---

```python
1 OPTICS(Objects, e, MinPts, OrderFile):
2    for each unprocessed obj in objects:
3        neighbors = Objects.getNeighbors(obj, e)
4        obj.setCoreDistance(neighbors, e, MinPts)
5        OrderFile.write(obj)
6    if obj.coreDistance != NULL:
7        orderSeeds.update(neighbors, obj)
8    for obj in orderSeeds:
9        neighbors = Objects.getNeighbors(obj, e)
10       obj.setCoreDistance(neighbors, e, MinPts)
11       OrderFile.write(obj)
12       if obj.coreDistance != NULL:
13           orderSeeds.update(neighbors, obj)
```

---

```python
1 OrderSeeds::update(neighbors, centerObj):
2    d = centerObj.coreDistance
3    for each unprocessed obj in neighbors:
4        newRdist = max(d, dist(obj, centerObj))
5        if obj.reachability == NULL:
6            obj.reachability = newRdist
7            insert(obj, newRdist)
8        elif newRdist < obj.reachability:
9            obj.reachability = newRdist
10           decrease(obj, newRdist)
```
Get Neighbors, Calc Core Distance, Save Current Object

=> pt1 $\varepsilon' \cdot 10$ rd:NULL

=> pt2 5 10

=> pt3 7 5
Get Neighbors, Calc Core Distance, Save Current Object

Calc/Update Reachability Distances

Update Processing Order

$\Rightarrow$ pt1 $\varepsilon'$:10 rd:NULL

$\Rightarrow$ pt2 5 10

$\Rightarrow$ pt3 7 5

$\ldots$
Get Neighbors, Calc Core Distance, Save Current Object

Calc/Update Reachability Distances

Update Processing Order

=> pt1 \( \epsilon' : 10 \) rd:NULL

=> pt2 5 10

=> pt3 7 5

...
Get Neighbors, Calc Core Distance, Save Current Object

Calc/Update Reachability Distances

Update Processing Order

=> pt1 $\epsilon':10$ rd:NULL

=> pt2 5 10

=> pt3 7 5...
Get Neighbors, Calc Core Distance, Save Current Object

Calc/Update Reachability Distances

Update Processing Order

=> pt1 $\varepsilon':10$ rd:NULL

=> pt2 5 10

=> pt3 7 5
Get Neighbors, Calc Core Distance, Save Current Object

Calc/Update Reachability Distances

Update Processing Order

$\Rightarrow pt1 \, \epsilon' : 10 \, rd: NULL$

$\Rightarrow pt2 \, 5 \, 10$

$\Rightarrow pt3 \, 7 \, 5$

$\ldots$
Get Neighbors, Calc Core Distance, Save Current Object

Calc/Update Reachability Distances

Update Processing Order

=> pt1 $\epsilon' \cdot 10$ rd:NULL

=> pt2 5 10

=> pt3 7 5

\[ \ldots \]
Get Neighbors, Calc Core Distance, Save Current Object

Calc/Update Reachability Distances

Update Processing Order

=> pt1 $\varepsilon':10$ rd:NULL

=> pt2 5 10

=> pt3 7 5
Get Neighbors, Calc Core Distance, Save Current Object

Calc/Update Reachability Distances

Update Processing Order

=> pt1 \( \varepsilon' :10 \) rd:NULL

=> pt2 5 10

=> pt3 7 5
Reachability Plots

A **reachability plot** is a bar chart that shows each object’s reachability distance in the order the object was processed. These plots clearly show the cluster structure of the data.
Automatic Cluster Extraction

Retrieving DBSCAN clusters

1. \texttt{ExtractDBSCAN(OrderedPoints, ei, MinPts)}:
2. \hspace{1em} \texttt{clusterId} = \texttt{NOISE}
3. \hspace{1em} \texttt{for each obj in OrderedPoints:}
4. \hspace{2em} \texttt{if obj.reachability} > \texttt{ei:}
5. \hspace{3em} \texttt{if obj.coreDistance} <= \texttt{ei:}
6. \hspace{4em} \texttt{clusterId} = \texttt{nextId(clusterId)}
7. \hspace{4em} \texttt{obj.clusterId} = \texttt{clusterId}
8. \hspace{3em} \texttt{else:}
9. \hspace{4em} \texttt{obj.clusterId} = \texttt{NOISE}
10. \hspace{2em} \texttt{else:}
11. \hspace{3em} \texttt{obj.clusterId} = \texttt{clusterId}

Extracting hierarchical clusters

A **steep upward point** is a point that is \( t \% \) lower that its successor. A **steep downward point** is similarly defined.

A **steep upward area** is a region from \([s, e]\) such that \( s \) and \( e \) are both steep upward points, each successive point is at least as high as its predecessors, and the region does not contain more than \( MinPts \) successive points that are not steep upward.

A **cluster**:  
- Starts with a steep downward area  
- Ends with a steep upward area  
- Contains at least \( MinPts \)  
- The reachability values in the cluster are at least \( t \% \) lower than the first point in the cluster.

1. \texttt{HierarchicalCluster(objects)}:
2. \hspace{1em} \texttt{for each index:}
3. \hspace{2em} \texttt{if start of down area D:}
4. \hspace{3em} \texttt{add D to steep down areas}
5. \hspace{3em} \texttt{index} = \texttt{end of D}
6. \hspace{2em} \texttt{elif start of steep up area U:}
7. \hspace{3em} \texttt{index} = \texttt{end of U}
8. \hspace{2em} \texttt{for each steep down area D:}
9. \hspace{3em} \texttt{if D and U form a cluster:}
10. \hspace{4em} \texttt{add [start(D), end(U)] to set of clusters}
References
