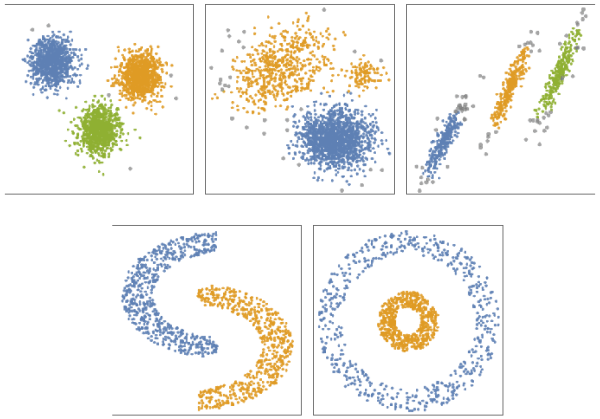
Dynamic Density-Based Clustering

Bogyeong Kim, Kyoseung Koo, Undraa Enkhbat, and Bongki Moon

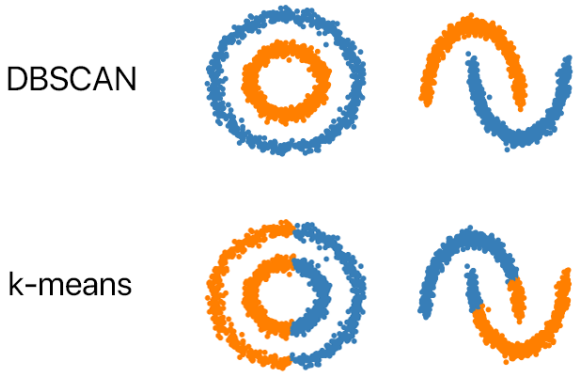
Density-Based Clustering

- **D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise (**DBSCAN)**
  - ✓ Martin Ester, et al. (KDD 1996)
- Unlike other clustering algorithms such as K-means or BIRCH, DBSCAN can detect clusters of ***arbitrary shapes***.



DBSCAN

(https://reference.wolfram.com/language/ref/method/DBSCAN.html)

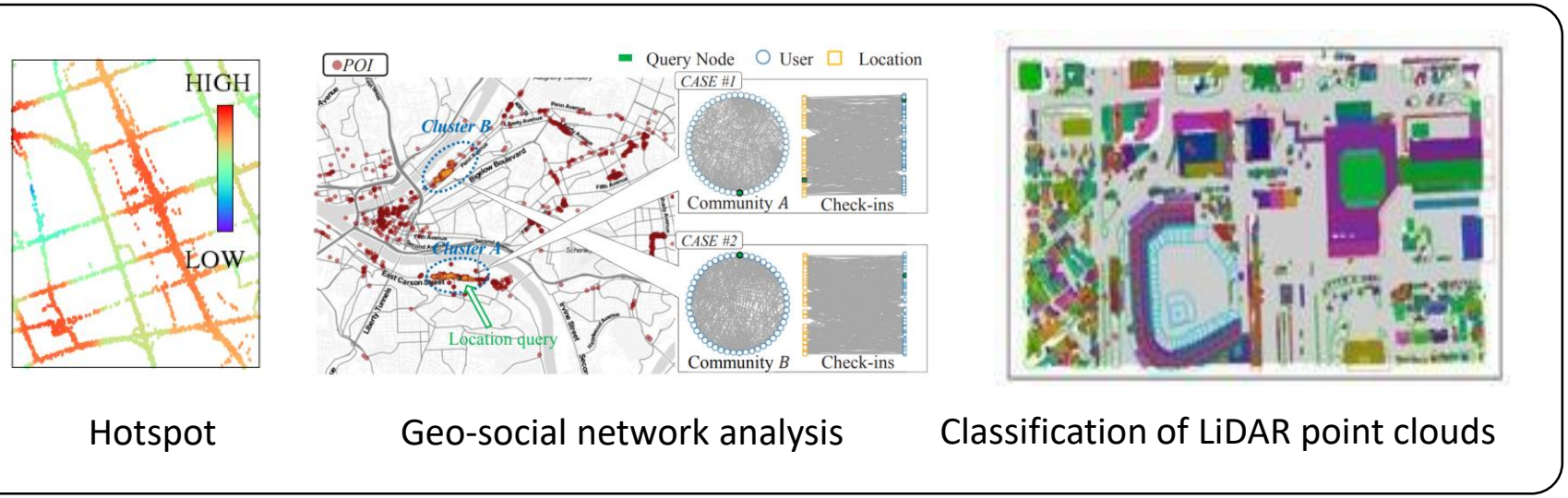


DBSCAN vs K-means

https://scikit-learn.org/

Density-Based Clustering

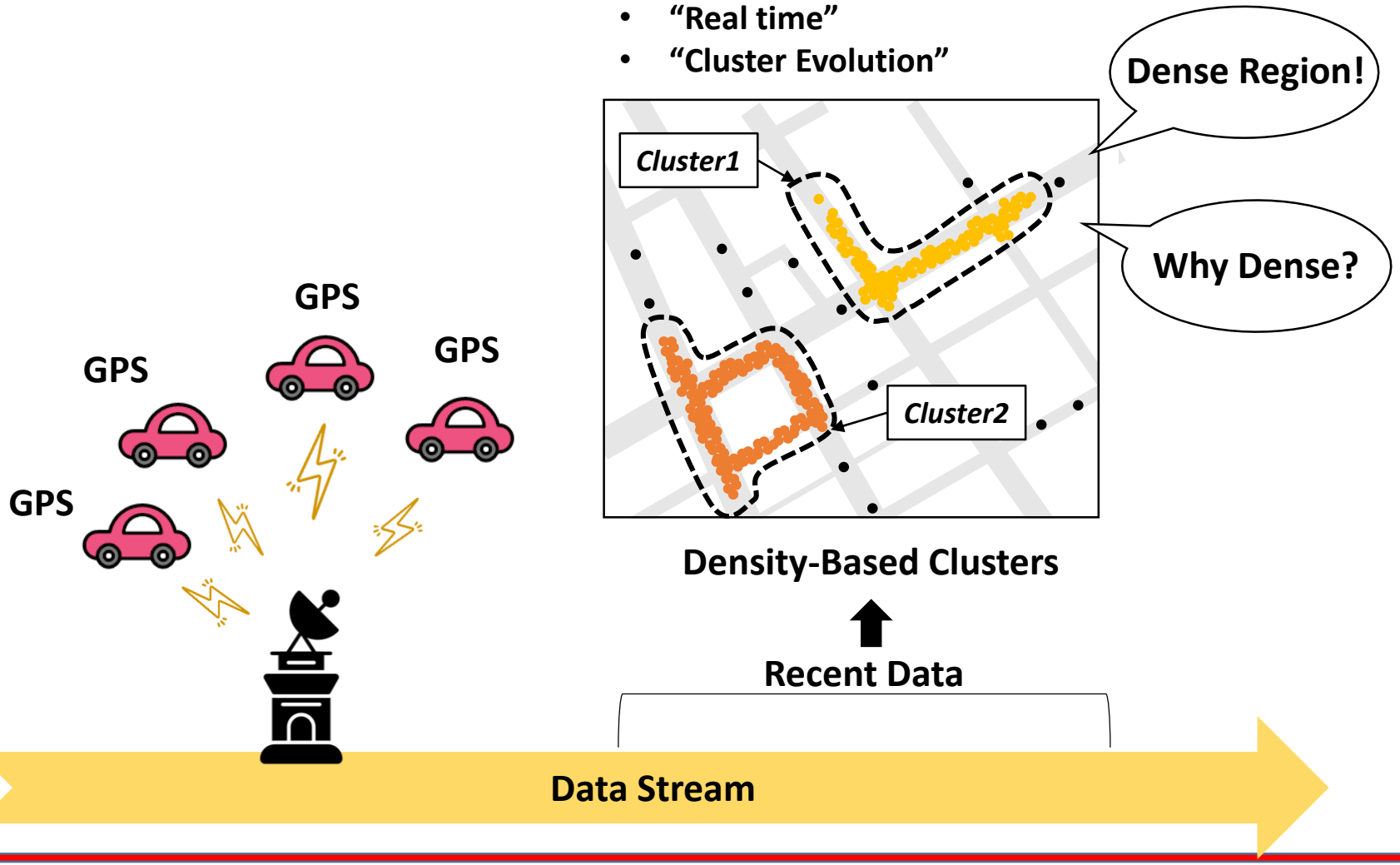
- **Applications of Density-Based Clustering**
  - ✓ Hotspot or Segmented Regions
  - ✓ Geo-social network analysis
  - ✓ Classification of LiDAR point clouds
  - ✓ Mining events by clustering text messages



Hotspot | Geo-social network analysis | Classification of LiDAR point clouds

Dynamic Clustering |over Streaming Data

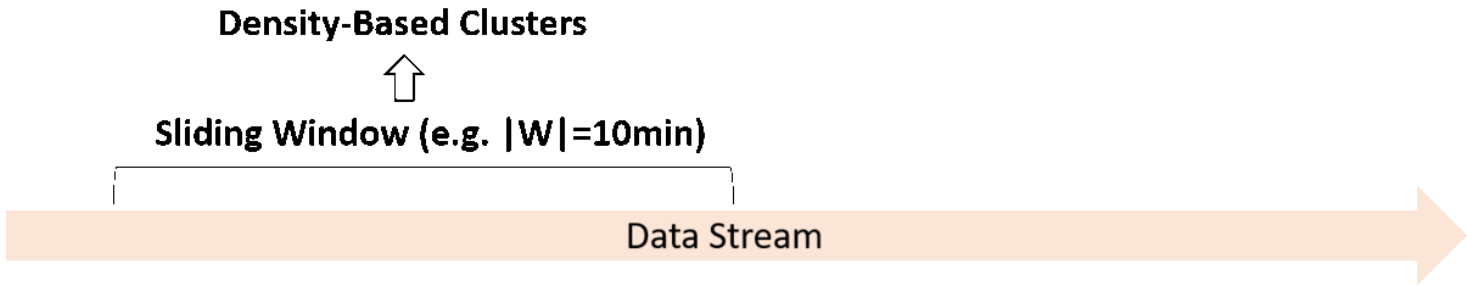
- "Real time"
- "Cluster Evolution"



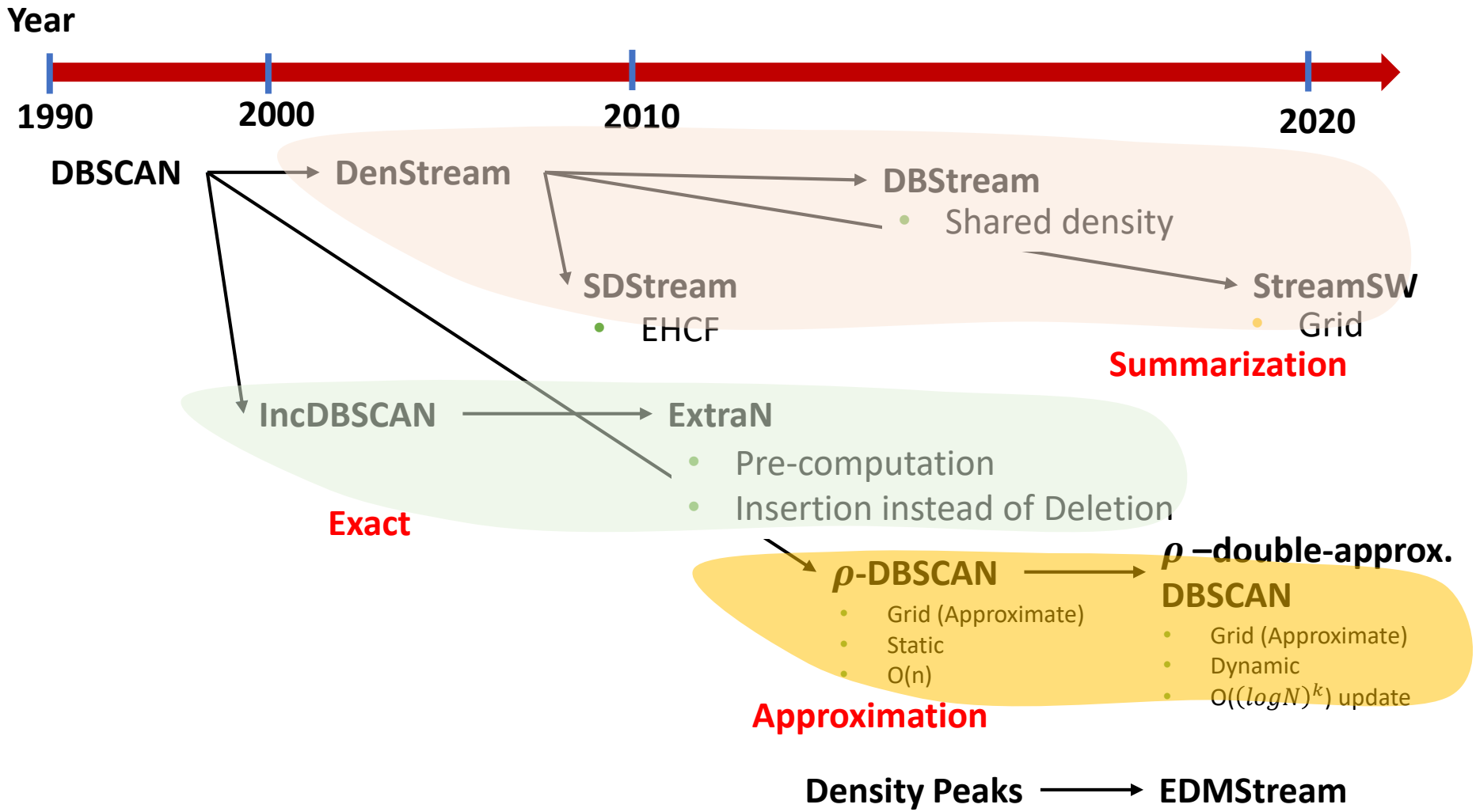
Introduction |Density-Based Clustering over Sliding Windows

- **Sliding Window Model**
  - **Window** is a set of consecutive data points in the data stream**.**
  - Window **slides** over data streams **periodically**.
  - **The window size |W|** is *time-based* or *count-based*.

✓A task of detecting density-based clusters for the **data points in the current window.**

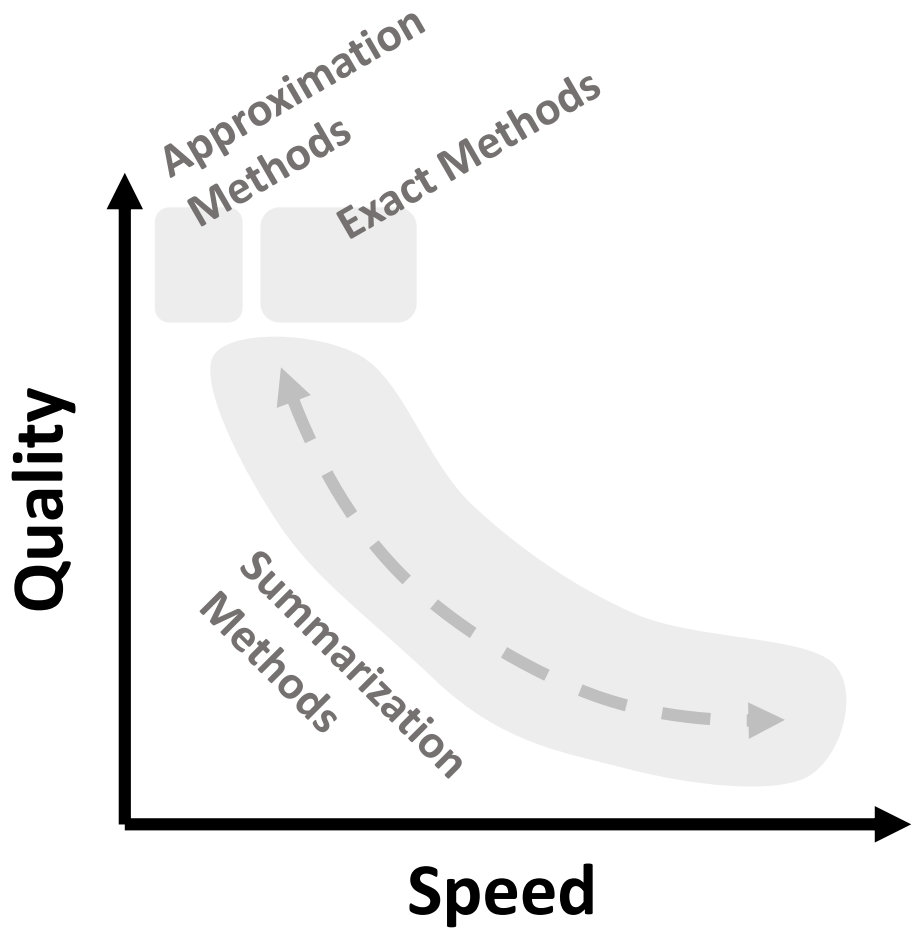


Related Work

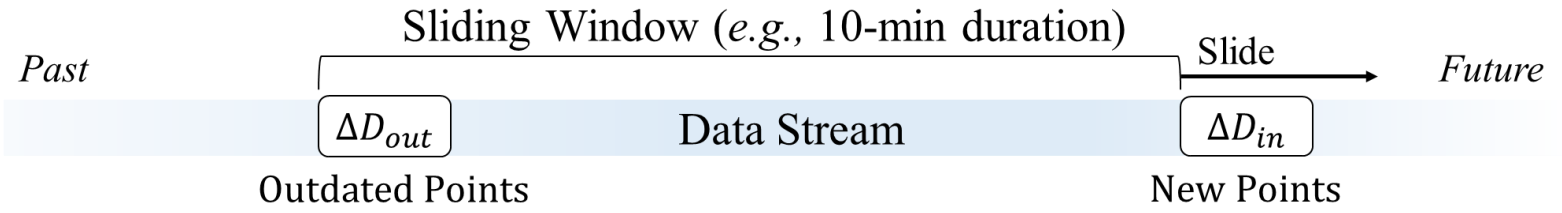


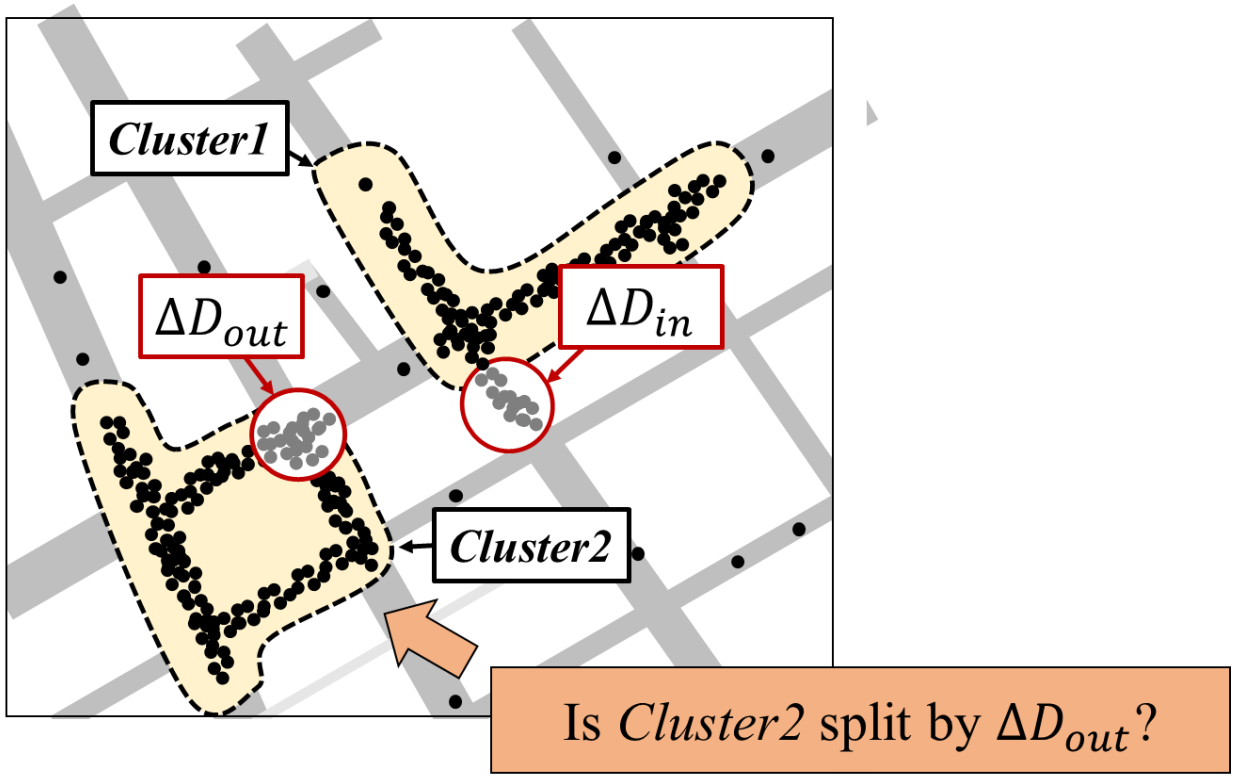
Related Work : Trade-offs

- **Exact Methods**
  - ✓ **Good** Quality
  - ✓ **Bad** Speed
- **Approximation Methods**
  - ✓ **Good** Quality
  - ✓ **Good** Theoretical Speed
  - ✓ **Bad** Practical Speed
- **Summarization Methods**
  - ✓ Speed - Quality **Tradeoff**



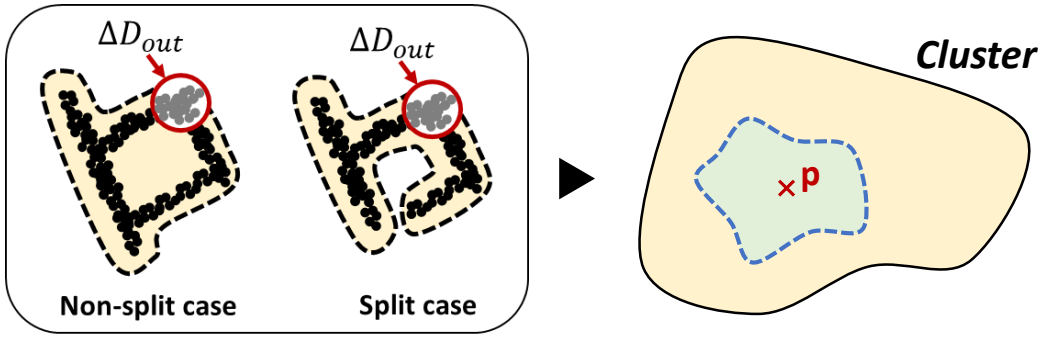
Major Challenge





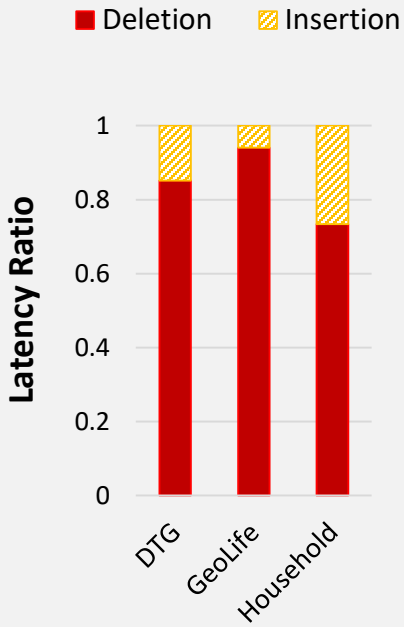
Major Challenge : Slow Deletion

- ***Split Check***



✓ *Determining "Split or Not" may require exploring a large number of surrounding points.*

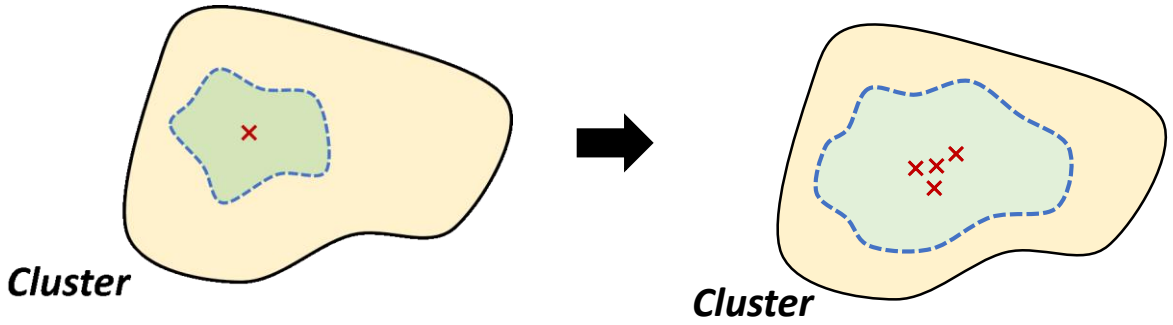
✓ ***Major performance bottleneck → Slow Deletion***



Get over the Slow Deletion Challenge

✓ Efficient "Split" Check for multiple updates

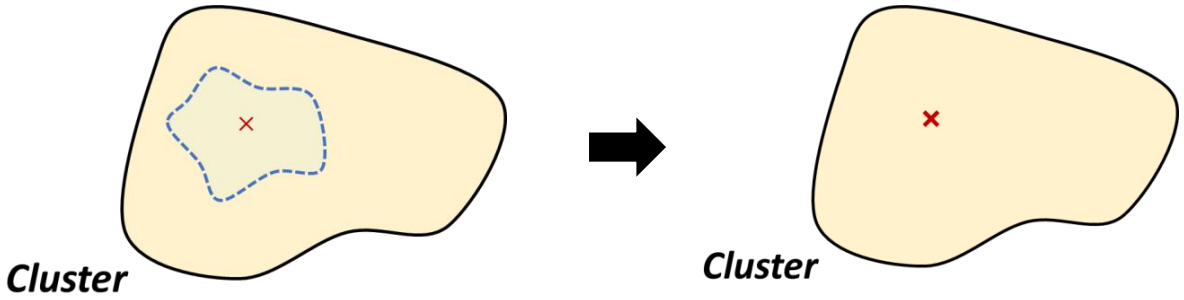
DISC algorithm (ICDE 2021)



- Avoid **redundant exploration** of surrounding objects

✓ Efficient "Split" Check with spanning trees

DenForest algorithm (SIGMOD 2022)



- Do **no exploration** of surrounding objects

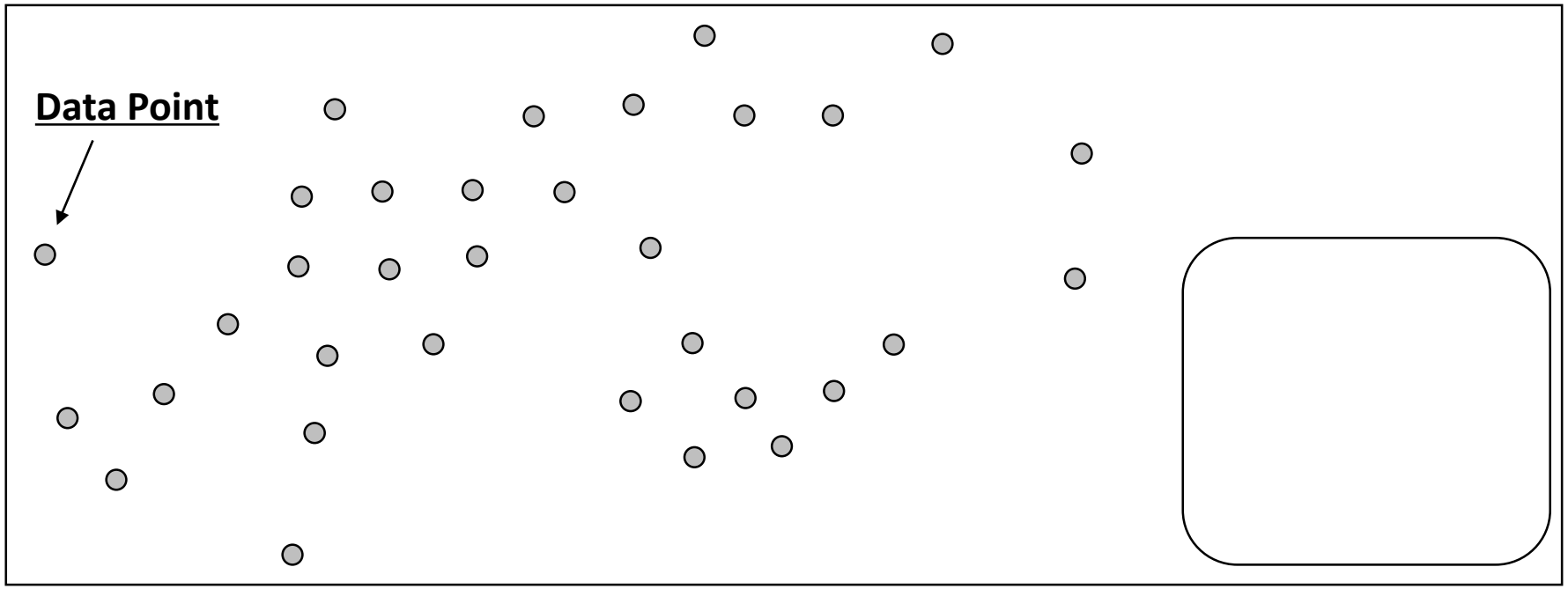
Background

$N_\epsilon(p)$: a set of ϵ - neighbors of p

τ : Density Threshold

ϵ : Distance Threshold

- **Density-Based Cluster (DBSCAN)**
  - ✓ Points are classified into 3 categories, **Core, Border, and Noise**



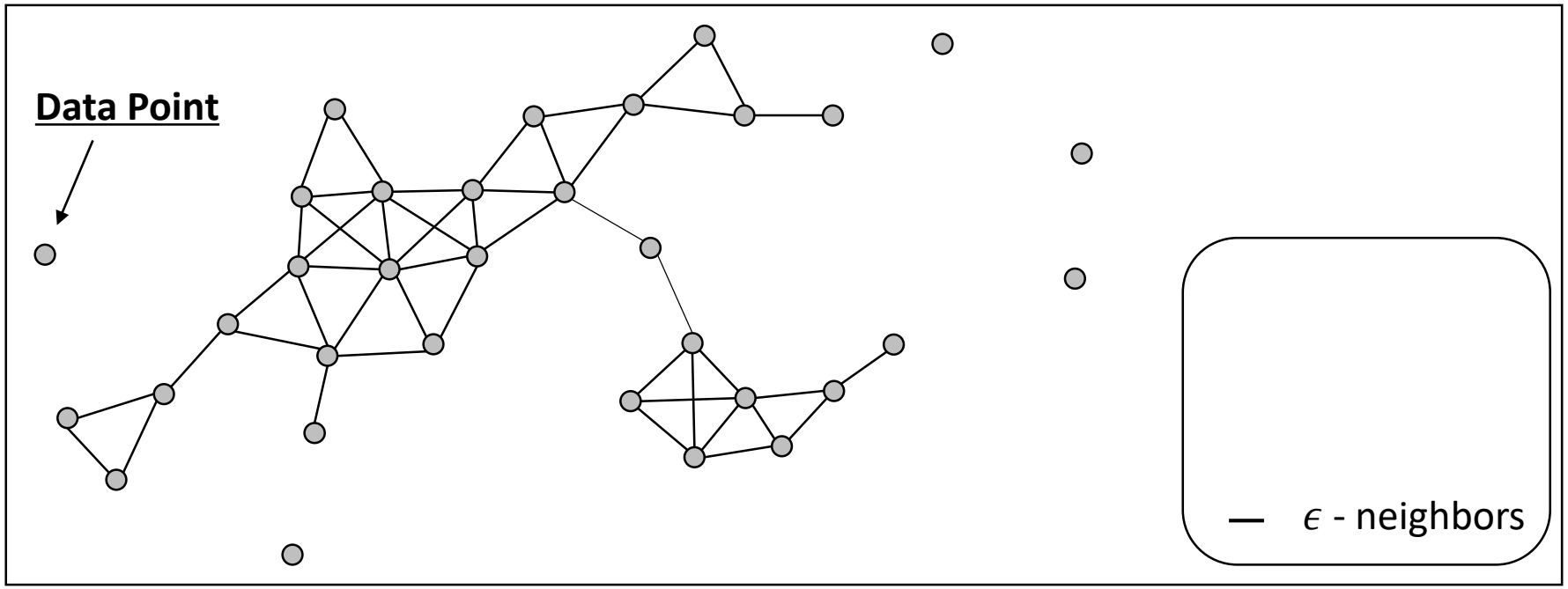
Background

$N_\epsilon(p)$: a set of ϵ - neighbors of p

τ : Density Threshold

ϵ : Distance Threshold

- **Density-Based Cluster (DBSCAN)**
  - ✓ Points are classified into 3 categories, **Core, Border, and Noise**



Background

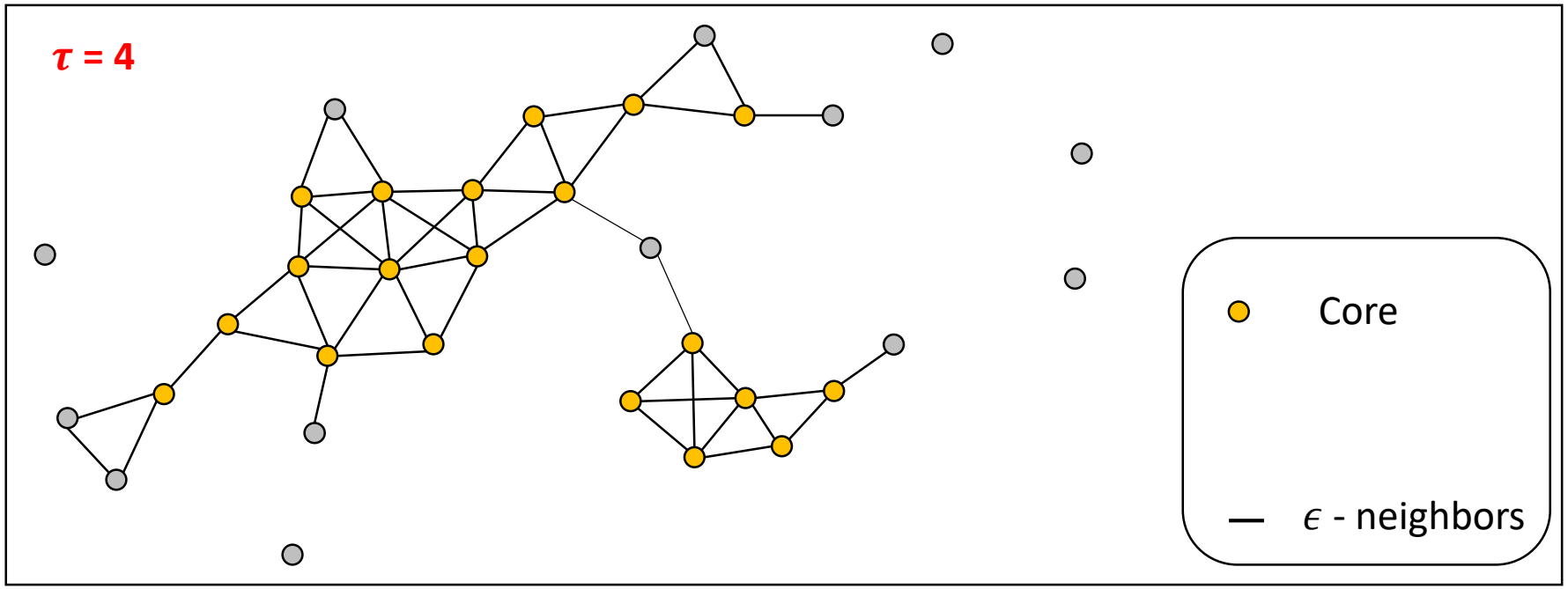
$N_\epsilon(p)$: a set of ϵ - neighbors of p

τ : Density Threshold

ϵ : Distance Threshold

- **Density-Based Cluster (DBSCAN)**
  - ✓ Points are classified into 3 categories, **Core, Border, and Noise**

Point p is **core** if $|N_\epsilon(p)| \geq \tau$



Background

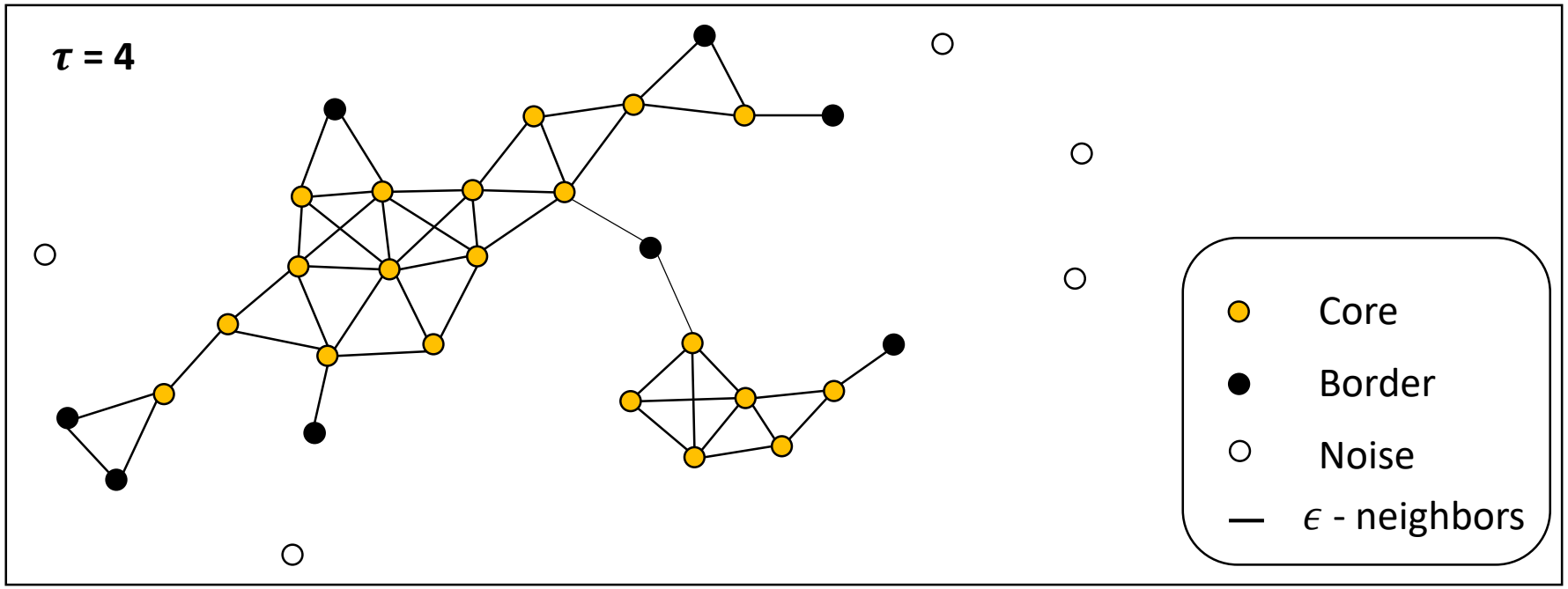
$N_\epsilon(p)$: a set of ϵ - neighbors of p

τ : Density Threshold

ϵ : Distance Threshold

- **Density-Based Cluster (DBSCAN)**
  - ✓ Points are classified into 3 categories, **Core, Border, and Noise**

Point p is **core** if $|N_\epsilon(p)| > \tau$ Point p is **border** or **noise** if $|N_\epsilon(p)| < \tau$



Background

$N_\epsilon(p)$: a set of ϵ - neighbors of p

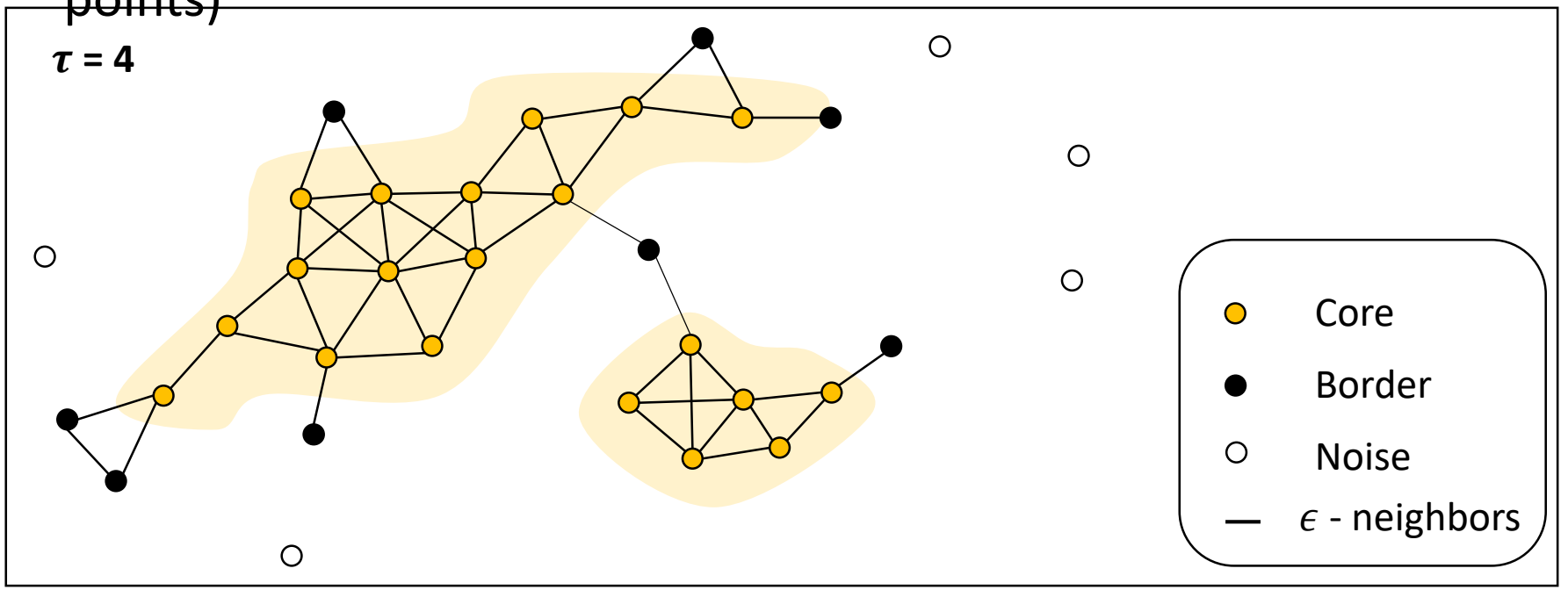
τ : Density Threshold

ϵ : Distance Threshold

- **Density-Based Cluster (DBSCAN)**

  = **A connected component of cores, ignoring all non-core point**

  + non-cores adjacent to the connected component (*i.e.*, border points)



Background

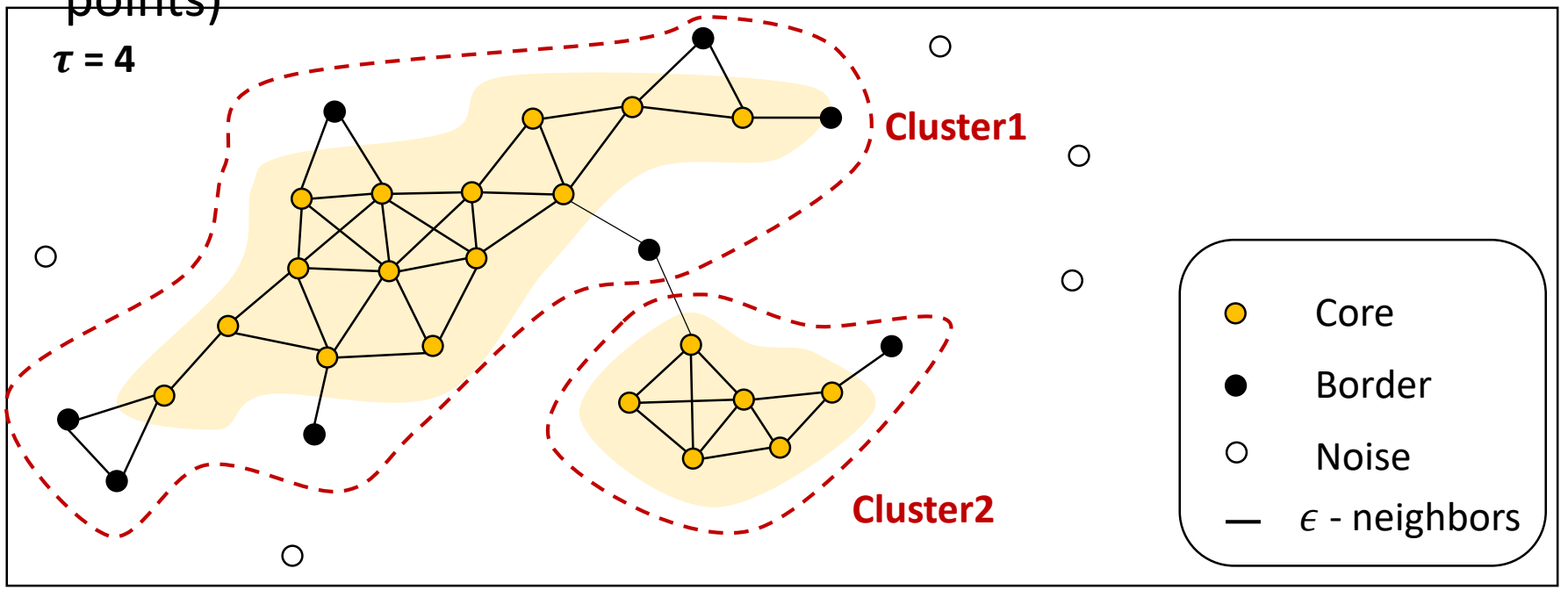
$N_\epsilon(p)$: a set of ϵ - neighbors of p

τ : Density Threshold

ϵ : Distance Threshold

- **Density-Based Cluster (DBSCAN)**

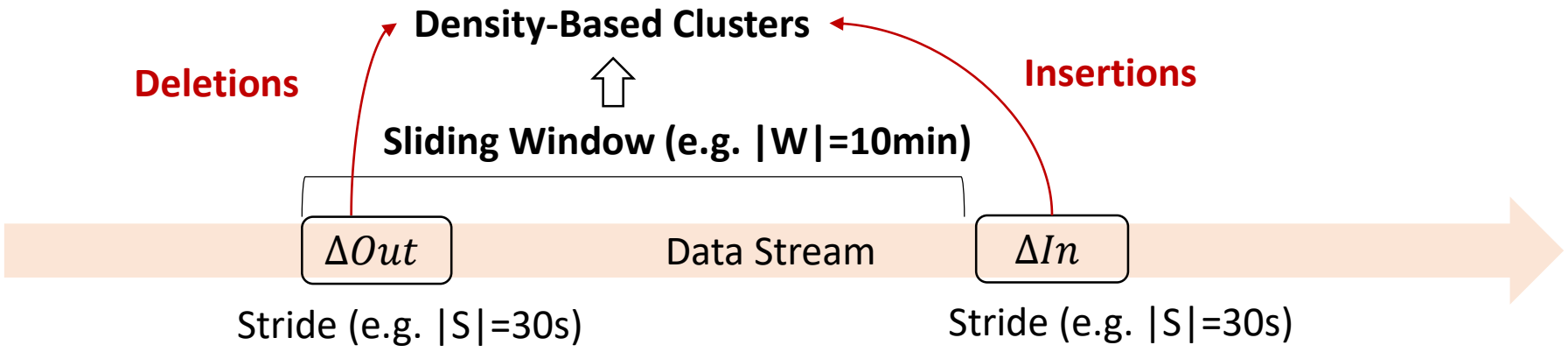
  = A connected component of cores, ignoring all non-core point
  + non-cores adjacent to the connected component (*i.e.*, border points)



Incremental Clustering over Sliding Window

- Sliding windows are updated **periodically.**
  - ✓ ***Stride*** determines the update interval of the sliding window.
  - ✓ The stride size |S| is also time-based or count-based.

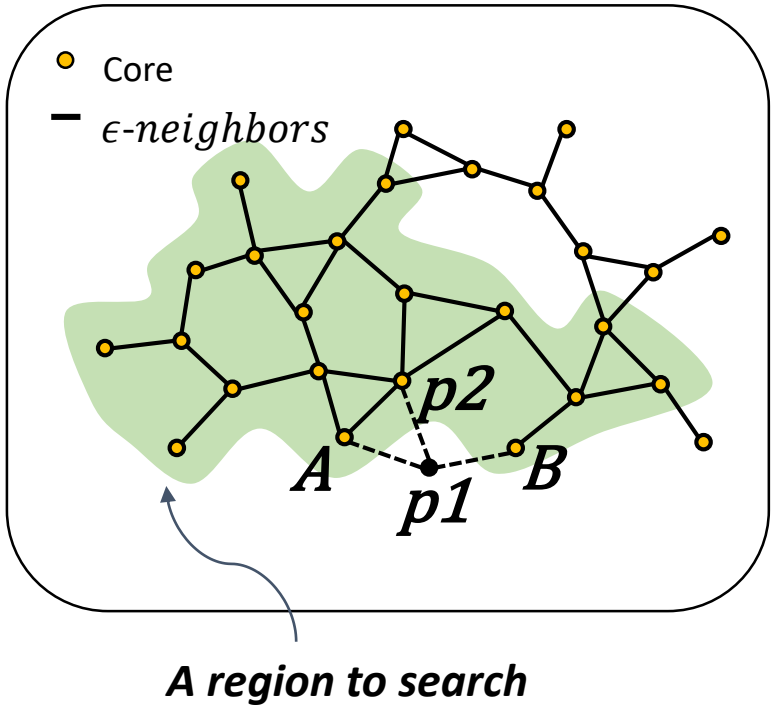
- **Multiple** points are inserted and deleted together.



Problem : Redundant Work in Deletion.

- Determining “split or not” requires graph traversals.
- Redundant work can be done when deleting multiple points individually.

Example.

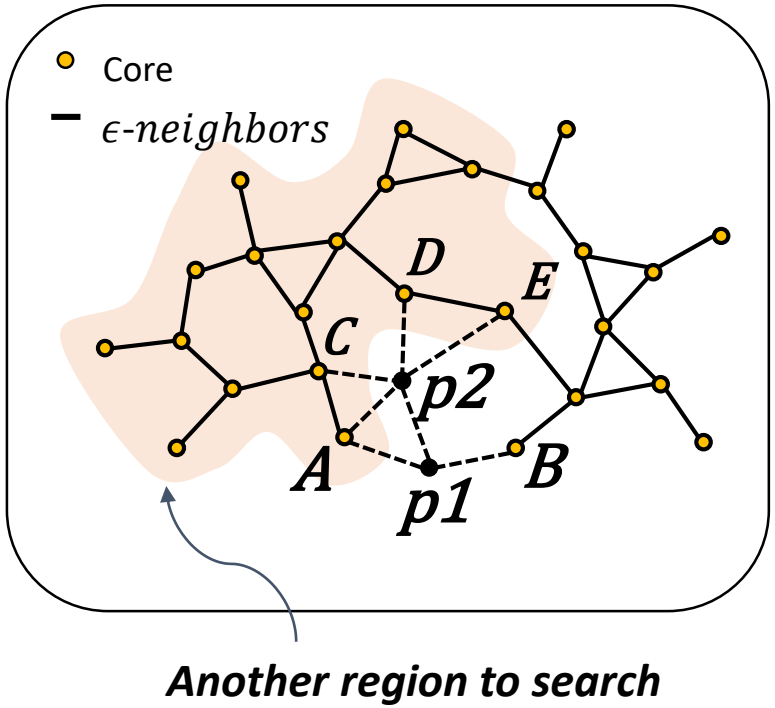


- **p1** becomes non-core due to nearby points removed.
- The connectivity of **p1**'s neighbors (A, B, and p2) needs to be checked.
- It can be checked by a graph traversal, for example, by a BFS from A to check the reachability to B and p2.

Problem : Redundant Work in Deletion.

- Determining “split or not” requires graph traversals.
- Redundant work can be done when deleting multiple points individually.

Example.

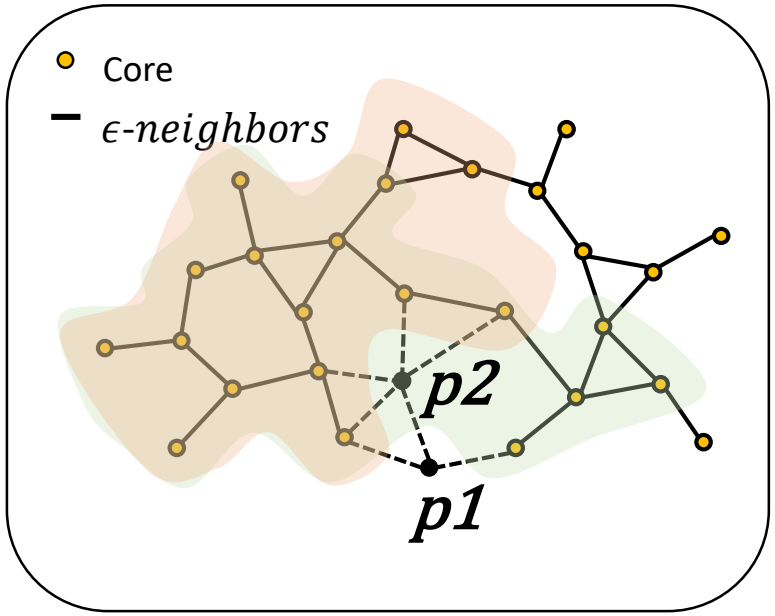


- **p2** also becomes non-core due to more points removed.
- The connectivity of **p2**'s neighbors (A, C, D and E) needs to be checked.
- It can be checked by a graph traversal, for example, by a BFS from A to check the reachability to C, D and E.

Problem : Redundant Work in Deletion.

- Determining “split or not” requires graph traversals.
- Redundant work can be done when deleting multiple points individually.

Example.

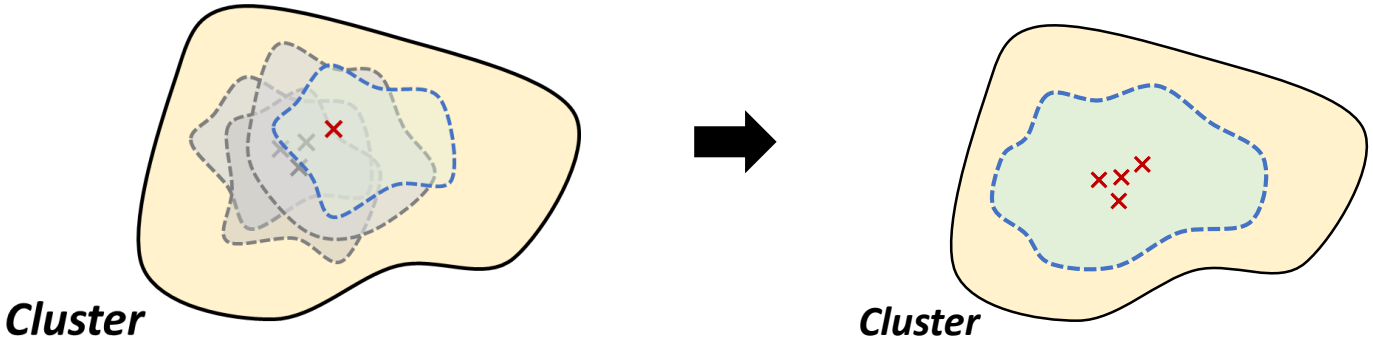


Two search regions are overlapped
→ Redundant work

DISC Approach

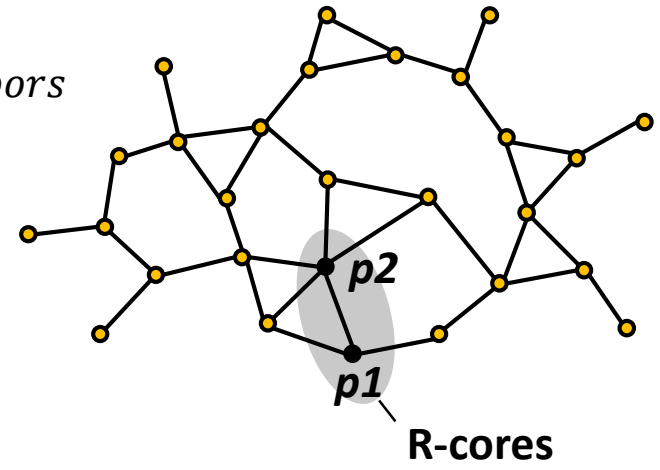
To avoid redundant work, DISC introduces **"Minimal Bonding Cores"**

- Further optimized with **Epoch-Based Probe** and **MS-BFS**
- Up to ***3x faster*** than the *Incremental DBSCAN*
- Produce exactly the same result as DBSCAN



DISC | Minimal Bonding Cores

- **Minimal Bonding Cores:** a set of cores adjacent to **R-cores**
  - **R-cores**: a connected component of *retro-reachable ex-cores*



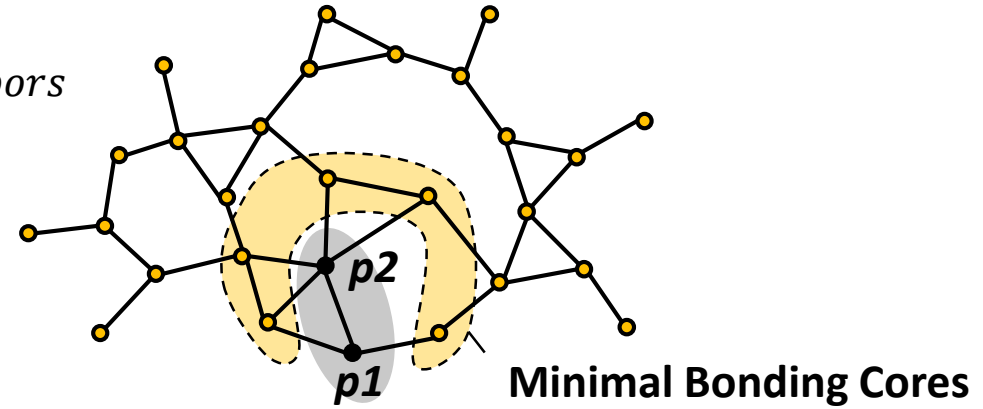
  - **p1** and **p2** were cores and were density-reachable from each other.

DISC | Minimal Bonding Cores

- **Minimal Bonding Cores:** a set of cores adjacent to **R-cores**
  - **R-cores**: a connected component of *retro-reachable ex-cores*

Core

ϵ-neighbors



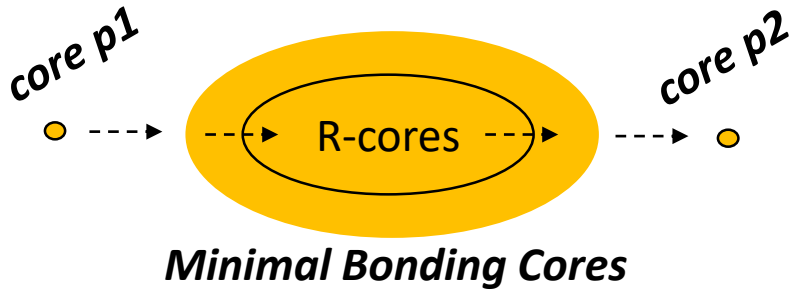
Minimal Bonding Cores

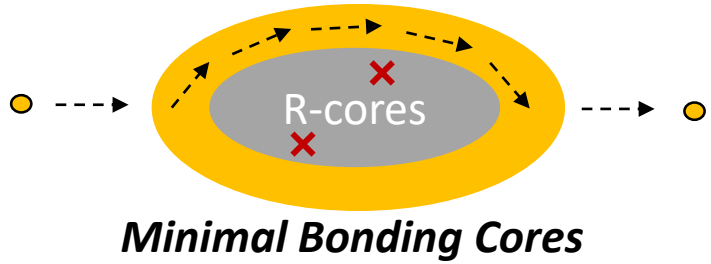
- Minimal set of cores that need to be checked for the “connectivity” of clusters.
  - ✓ Connectedness of all minimum bonding cores ensures connectedness of clusters after the deletion.

DISC | Minimal Bonding Cores

How does it ensure connectedness?

- Any path between two cores passing through the R-cores also passes through the minimal bonding cores.





<Before deletion>

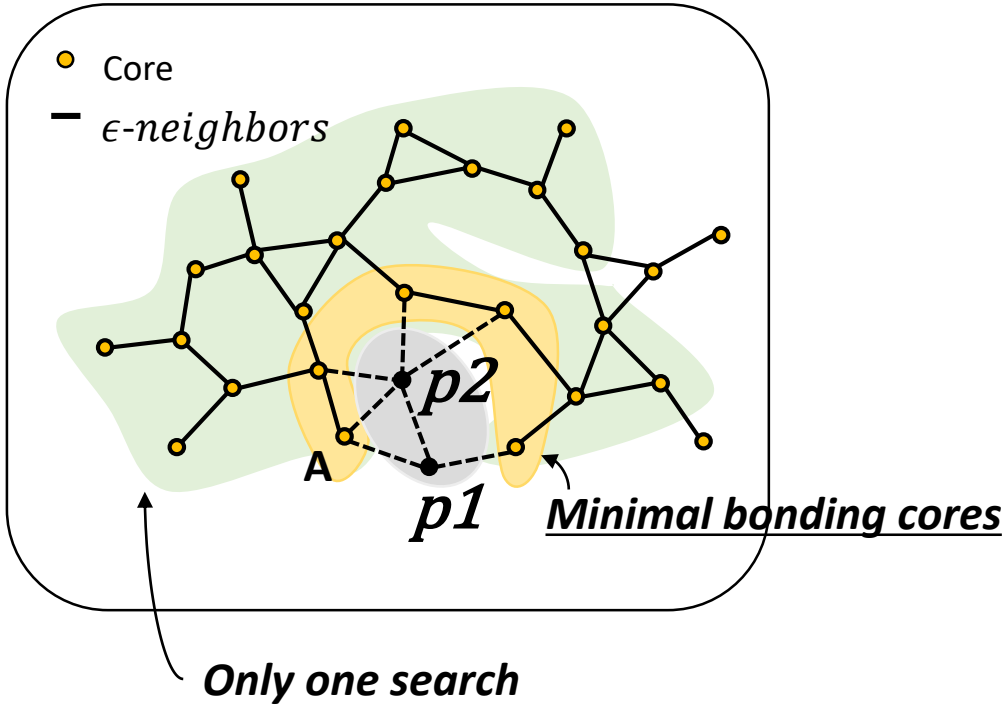
<After Deletion>

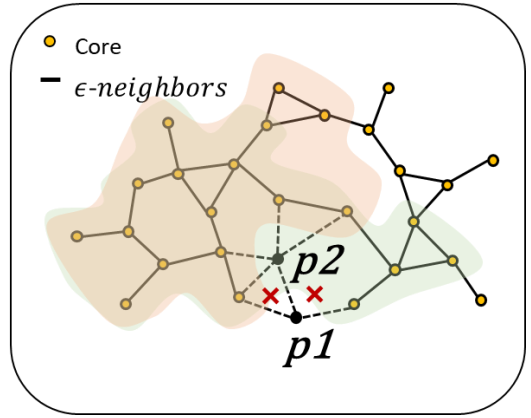
DISC | Minimal Bonding Cores

- **STEP1. Find R-cores (Retro-reachable cores)**
- **STEP2. Find minimal bonding cores.**
- **STEP3. Check connectivity.**

Ex) Start BFS from A
→ check reachability to minimal bonding cores

Example.





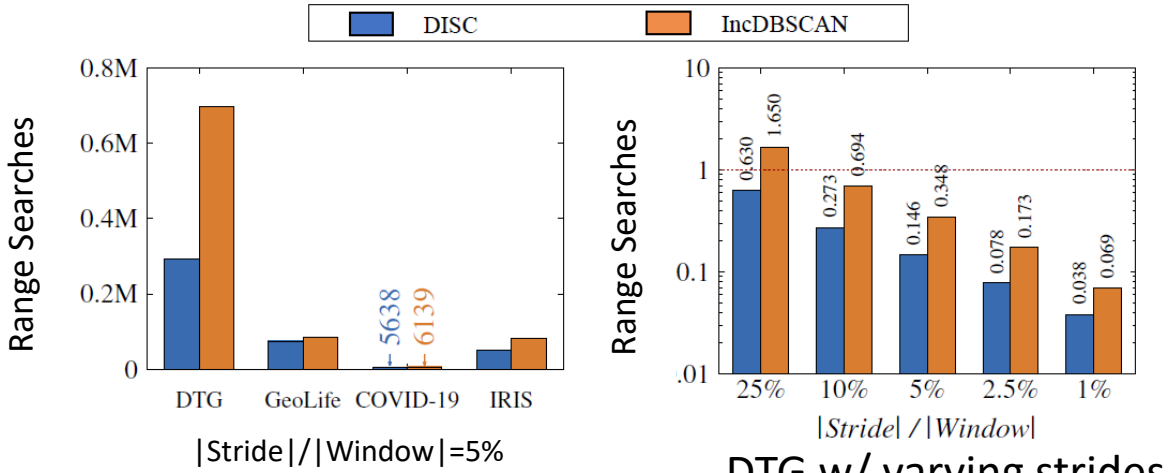
Evaluation

Real-world Datasets

Dataset	density (τ)	distance (ϵ)	$\|window\|$
DTG	372	0.002	2M (∼10 min)
$GeoLife$	7	0.01	200K (∼fortnight)
$COVID$-19	5	1.2	15K (∼fortnight)
$IRIS$	9	2	200K (∼decade)

Effect of minimal bonding cores

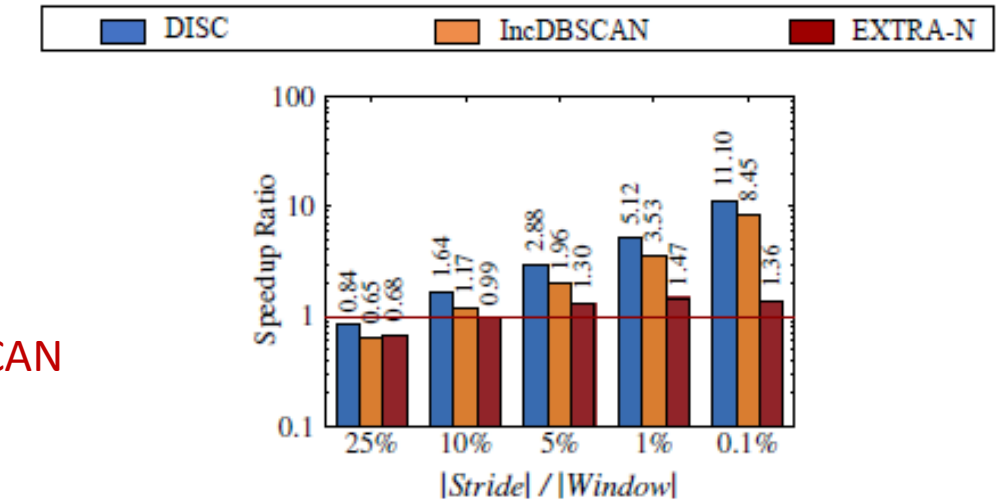
Reduce range searches by about 10~50%



DTG w/ varying strides
(relatively in comp with DBSCAN)

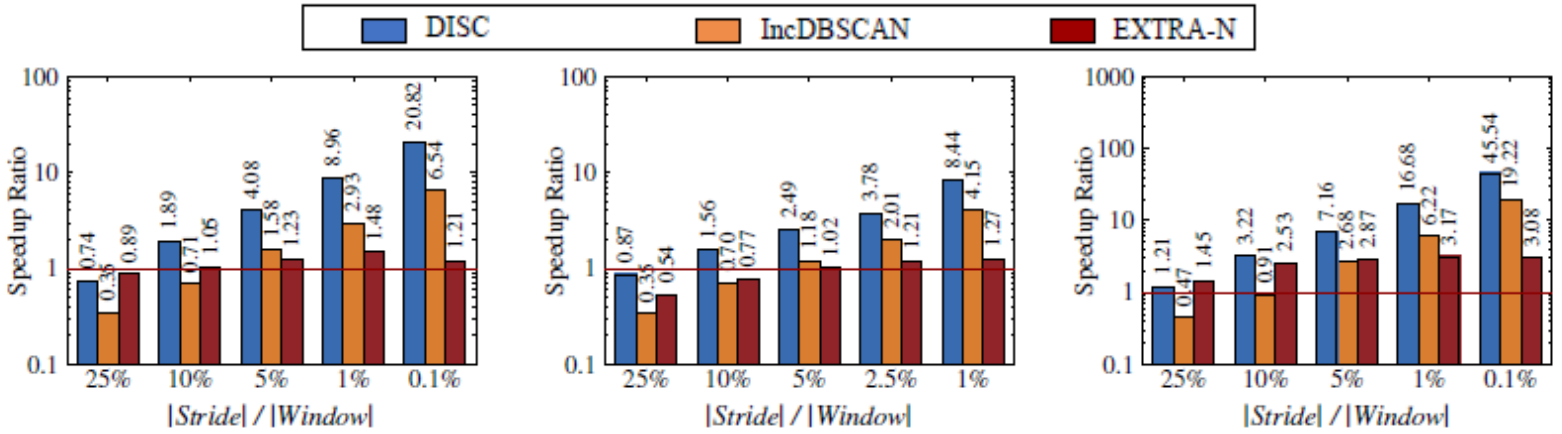
DISC performance for the DTG dataset

Up to 40% faster than IncDBSCAN
(11x faster than DBSCAN)



DTG w/ varying strides
(relatively in comp with DBSCAN)

DISC performance with the other datasets

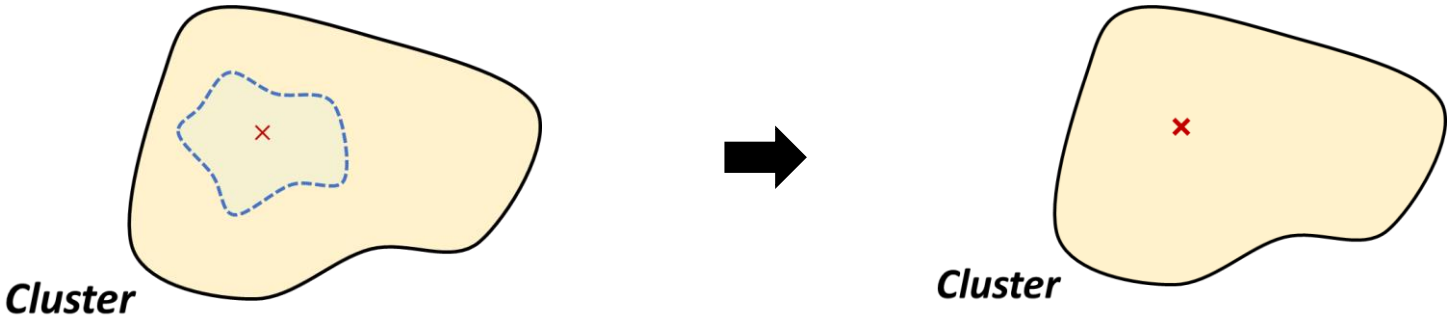


GeoLife **Covid-19** **IRIS**

DISC tends to achieve larger speedup over IncDBSCAN when the stride is smaller

DenForest Approach

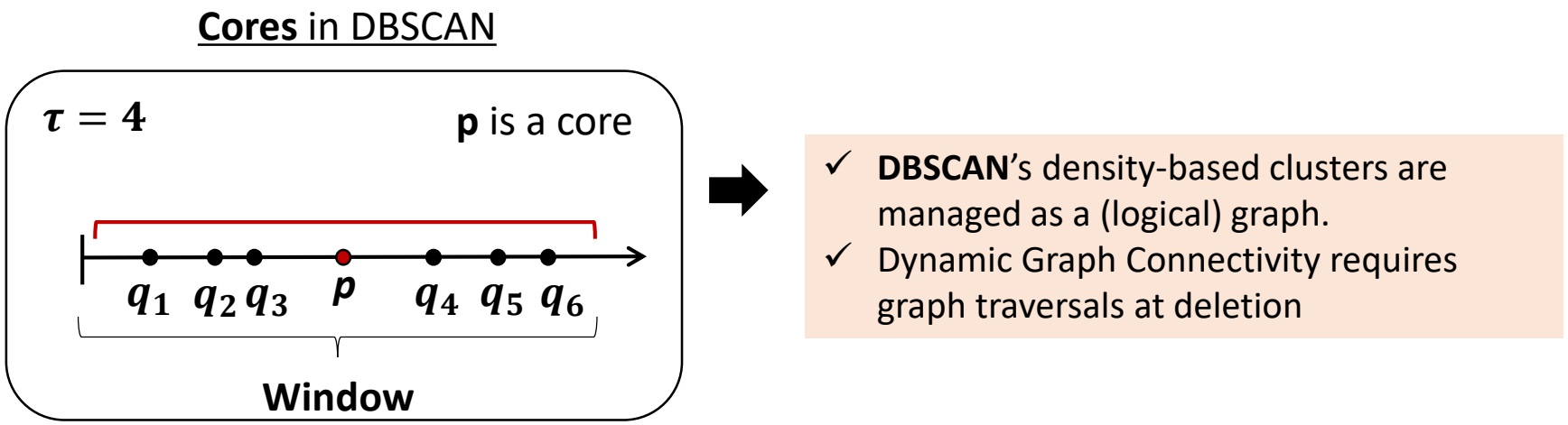
✓ Efficient “Split” Check with spanning trees



No exploration of surrounding objects

DenForest | Problem

- In **DBSCAN**, data points can change their core status repeatedly.
- Cannot predict when **p** gains and loses its core status



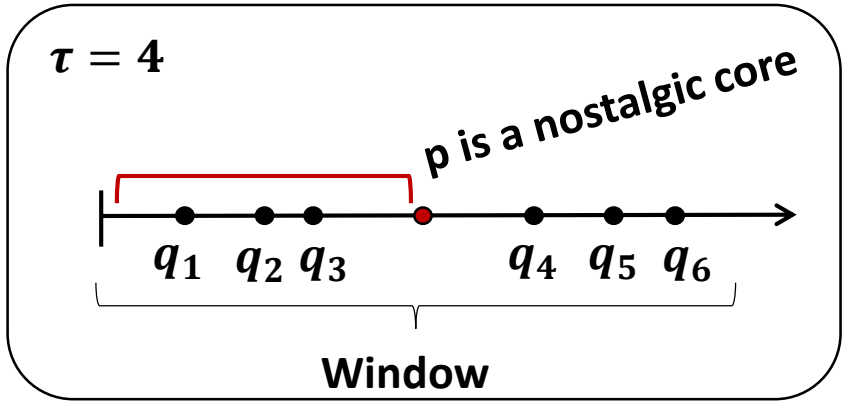
✓ **DBSCAN**'s density-based clusters are managed as a (logical) graph.
✓ Dynamic Graph Connectivity requires graph traversals at deletion

✓ *What if we can predict exactly when p loses its core status?*

DenForest | Nostalgic Core

Point p is a ***nostalgic core***, if the number of ϵ-neighbors is above the density threshold *when* it enters the window.

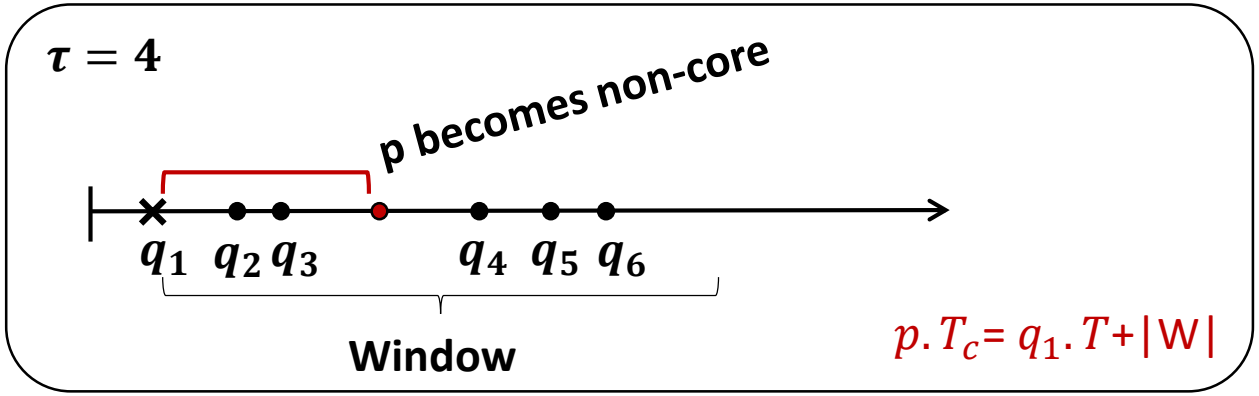
- **p** gains the core status due to q_1, q_2, q_3 when it enters.



DenForest | Core-expiration Time ($p.T_c$)

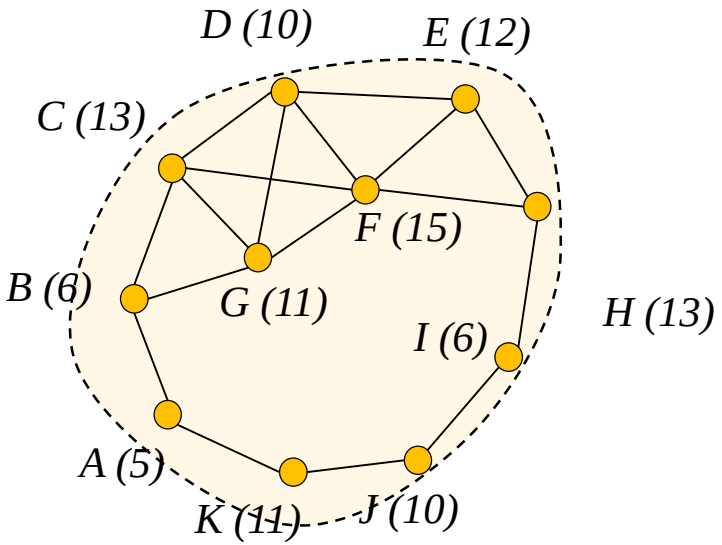
When does a nostalgic core **p** lose its core status?

- **p** loses its core status when $\mathbf{q_1}$ exits the window.
- $\mathbf{q_1}$ exits the window at time $q_1.T+|W|$.
- Thus, **p** loses its core status at time $q_1.T+|W|$.
- All is determined when **p** enters the window.



DenForest | DenGraph of Nostalgic Cores

- Nostalgic cores



A (5) denotes $A.T_c = 5$

$$w_{pq} = \min(p.T_c, q.T_c)$$

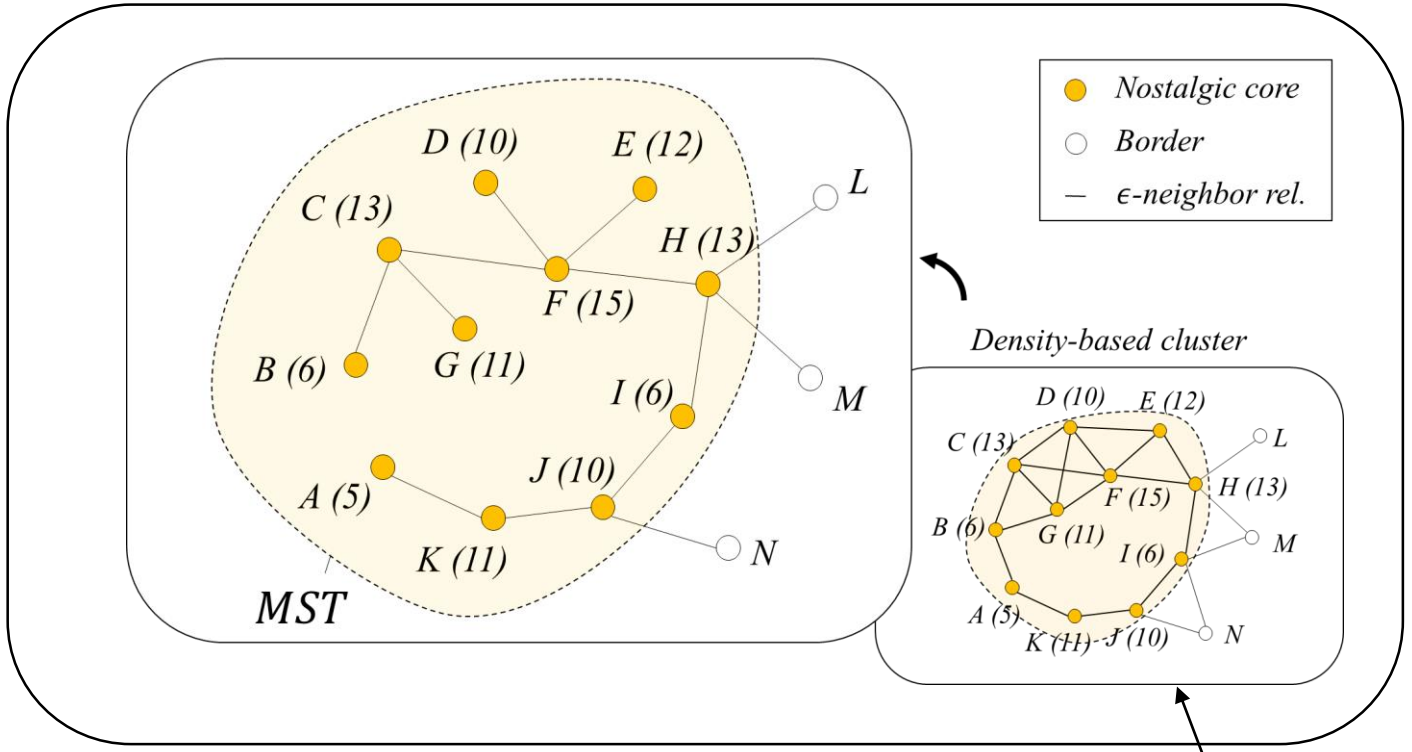
DenGraph

- *Each vertex is a nostalgic core labeled with its T_c time.*
- *Each edge denotes ϵ-neighbors whose weight is the smaller T_c time of the two vertices.*

✓ *Each connected component of DenGraph is a DenForest's cluster.*

DenForest | Maximum Spanning Tree

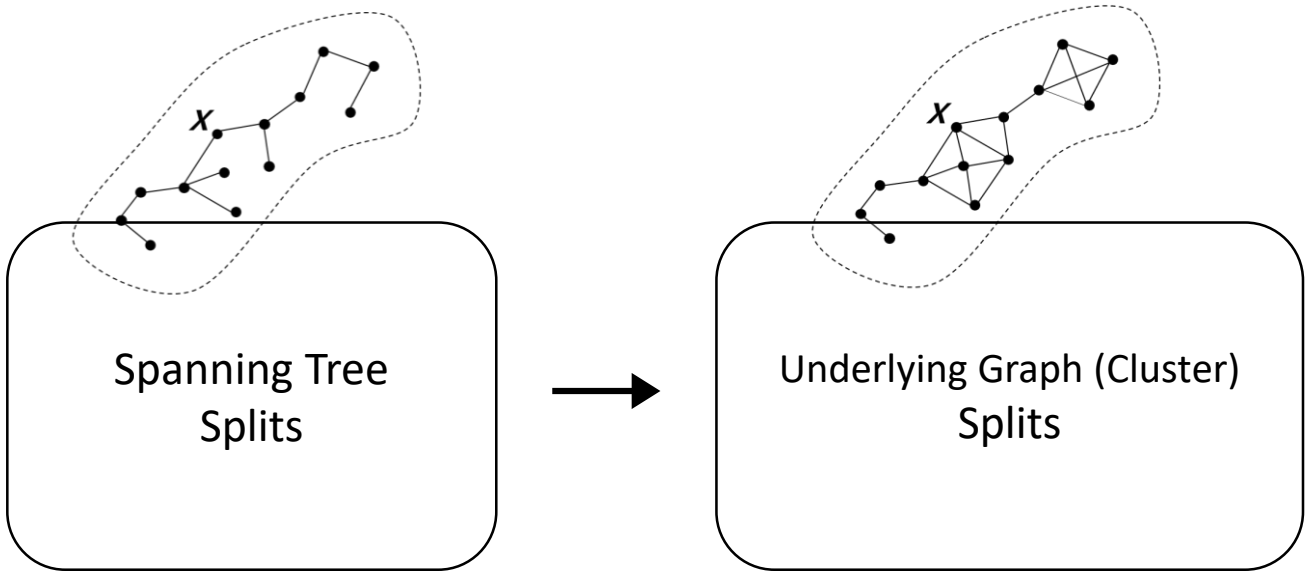
- *Property: If the maximum spanning tree splits, the underlying graph also splits*



Note that DenGraph is not physically built.

DenForest | No Range Search

✓ Property: If the maximum spanning tree splits, the underlying graph also splits.
✓ ***No graph traversals*** are required, which enables ***fast deletion*** of cores.



[is it internal node?]
- Yes ▶ tree split ▶ Graph split (= Cluster split)
- No ▶ No tree split ▶ No Graph split

Evaluation |Dataset and Competitors

- **Four real datasets**

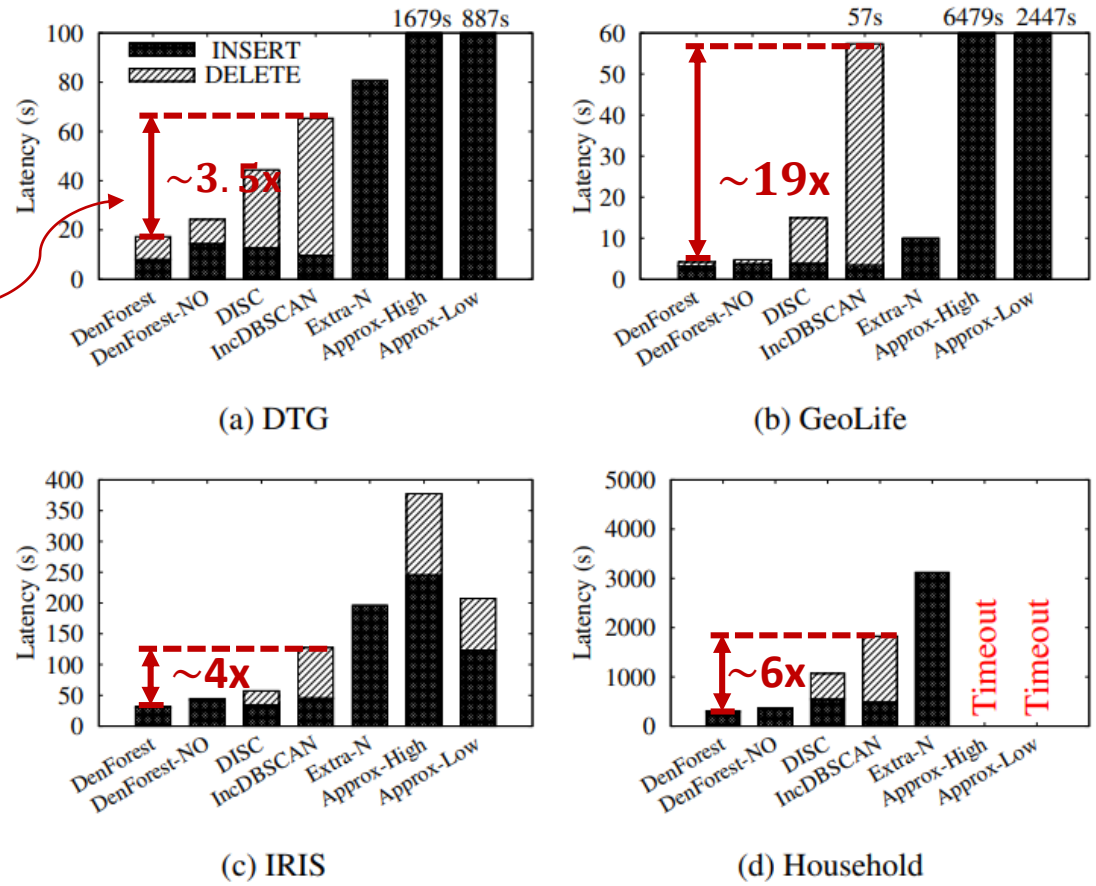
Dataset	dim	density (τ)	distance (ϵ)	$\lvert window \rvert$
DTG	2D	372	0.002	2M (~10 min)
GeoLife	3D	765	0.002	0.1M (~ week)
IRIS	4D	8	2	0.2M (~ decade)
Household	7D	14	0.3	0.5M (~ year)

- **Competitors**
  - Incremental DBSCAN (IncDBSCAN in short)
  - Extra-N
  - DISC
  - ρ-*double-Approx-DBSCAN*
    - Approx-High (ρ = 0.001)
    - Approx-Low (ρ = 0.1)

Update (Insert & Delete) Latency

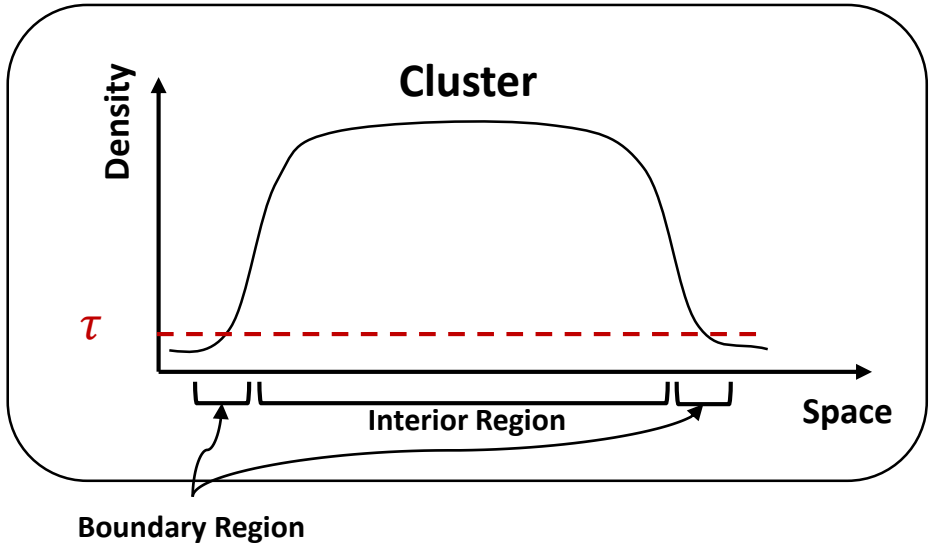
Update Latency of DenForest

Speedup ratio of DenForest in comparison with the baseline algorithm (IncDBSCAN)



DenForest |Quality

- DBSCAN and DenForest define cores *differently*.
- Nonetheless, they produce similar clusters at dense regions.



- **Quality evaluation with various datasets**

Dataset	*DenForest* vs. Label			DBSCAN vs. Label			*DenForest* vs. DBSCAN		
	ARI	AMI	NMI	ARI	AMI	NMI	ARI	AMI	NMI
Spiral [15]	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
R15 [80]	0.98	0.98	0.98	0.99	0.99	0.99	0.98	0.98	0.98
Aggr. [36]	0.97	0.96	0.97	0.99	0.99	0.99	0.97	0.96	0.97
Comp. [89]	0.94	0.87	0.90	0.94	0.85	0.91	0.98	0.88	0.94
G2-2-30 [61]	0.95	0.88	0.90	0.96	0.93	0.93	0.96	0.92	0.94
G2-4-30 [61]	0.99	0.97	0.99	1.00	1.00	1.00	0.99	0.97	0.99
G2-8-30 [61]	0.99	0.99	0.99	1.00	1.00	1.00	0.99	0.99	0.99
Average	**0.97**	**0.95**	**0.96**	**0.98**	**0.96**	**0.97**	**0.98**	**0.95**	**0.97**

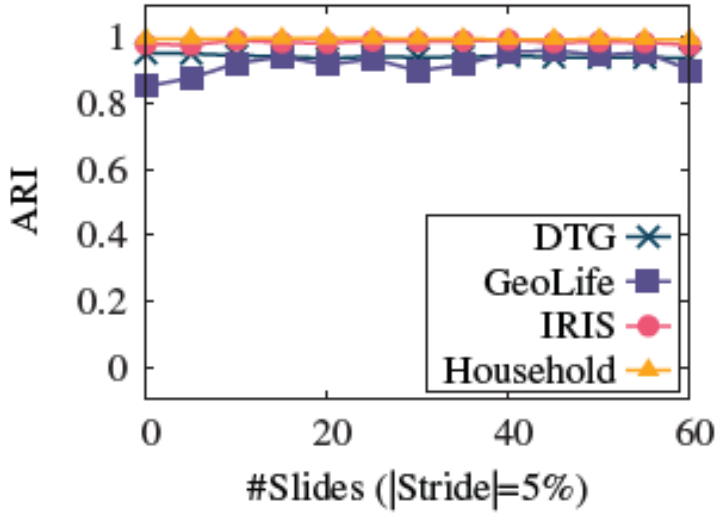
✓ **Interior Region (*Density* $\gg \tau$) :** $DBSCAN \cong DenForest$

✓ **Boundary Region (*Density* $\sim \tau$) :** Could be $DBSCAN \neq DenForest$ but gray area anyway

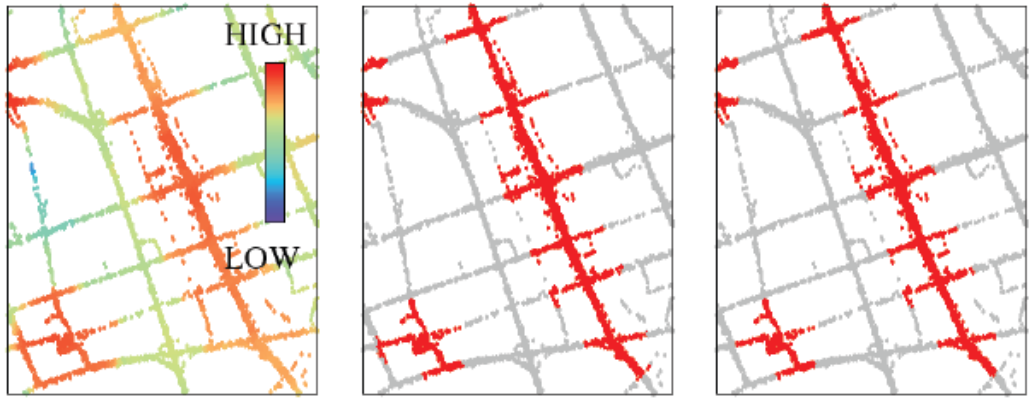
- ✓ ARI (Adjusted Rand Index)
- ✓ AMI (Adjusted Mutual Information)
- ✓ NMI (Normalized Mutual Information)

Quality

- Quality change over sliding window



- Heatmap and clusters from each method

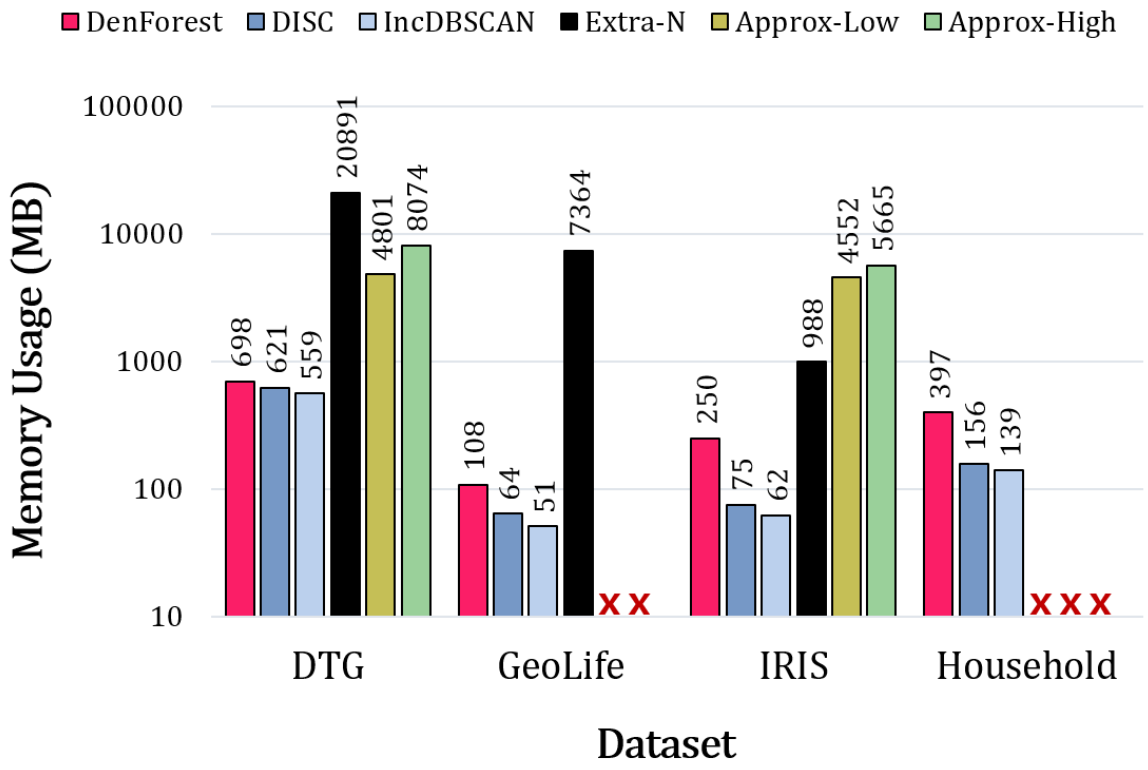


(a) Heatmap (b) DBSCAN (c) *DenForest*

Memory Usage | Four Datasets

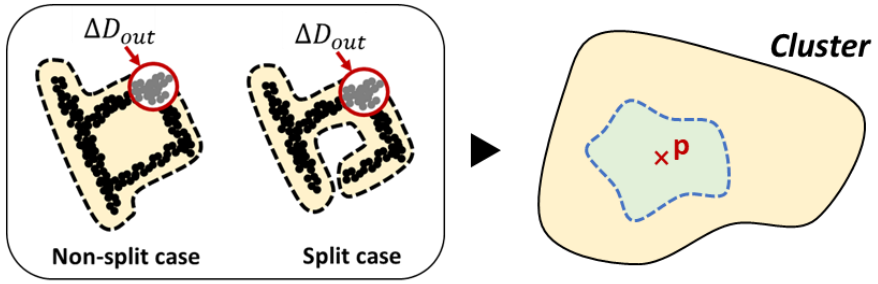
Datasets

Dataset	Window Size
DTG	2M
GeoLife	0.4M
IRIS	0.4M
Household	0.5M



Conclusion |Summary

- ***Split Check***



✓ *Determining "Split or Not" may require exploring a large number of surrounding points.*

✓ ***Major performance bottleneck → Slow Deletion***

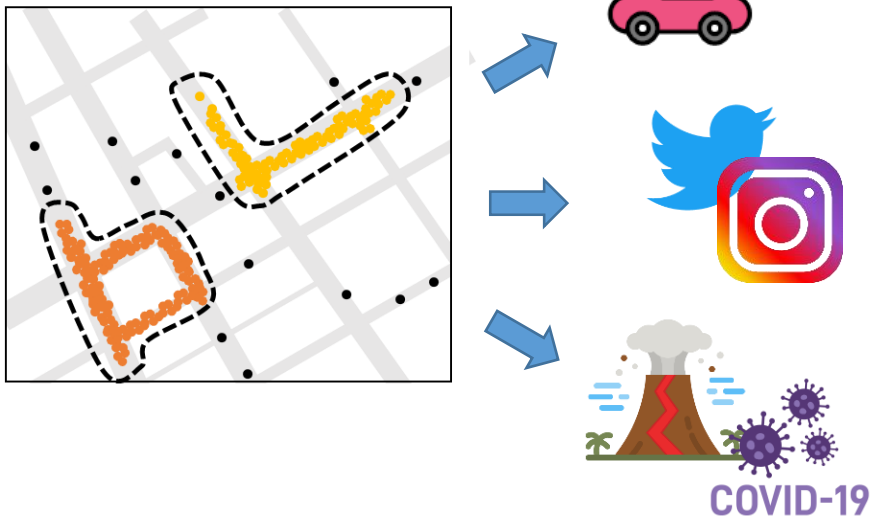
DISC

- Minimal bonding cores
  → Avoid redundant work in deletion

DenForest

- Maximal spanning tree of nostalgic cores
  → Avoid graph traversal in deletion

Conclusion | Expected Applications



- ✓ Traffic Monitoring
- ✓ Traffic Signal System

- ✓ Real-time Social Event Detection
- ✓ Tracking Trending Places

- ✓ Anomaly Detection in Volcanic Activity
- ✓ Temporal Analysis for Major COVID-19 Outbreaks

- DISC and DenForest are expected to support many data analytic tasks in the streaming environment at low computational cost.

Thank you!!