SECLEDS: Sequence Clustering in Evolving Data Streams via Multiple Medoids and Medoid Voting

Azqa Nadeem, Sicco Verwer

Delft University of Technology, The Netherlands

ECML/PKDD 2022

TU Delft

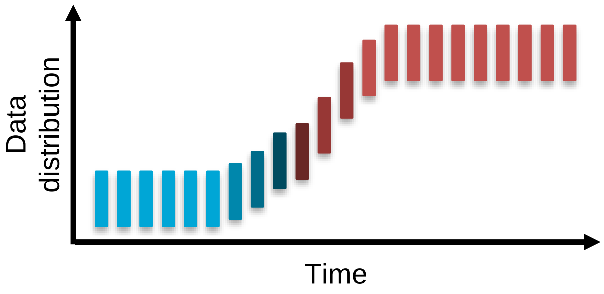
Cyber Analytics Lab

Sequence clustering in streaming environments

- Stream clustering
  - Cluster infinitely many data items

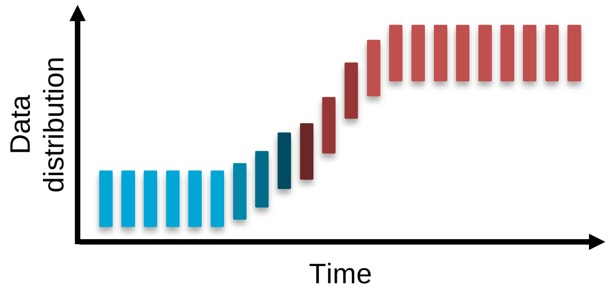
Sequence clustering in streaming environments

- Stream clustering
  - Cluster infinitely many data items
  - Concept drift



Sequence clustering in streaming environments

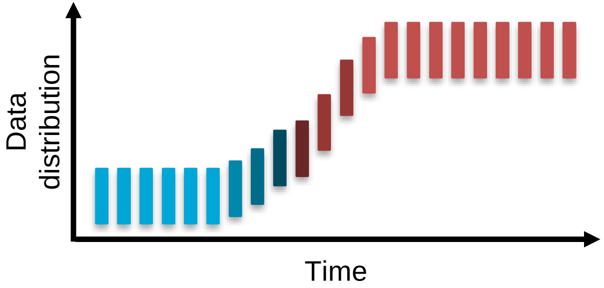
- Stream clustering
  - Cluster infinitely many data items
  - Concept drift
  - Heuristic-based solutions



TU Delft

Sequence clustering in streaming environments

- Stream clustering
  - Cluster infinitely many data items
  - Concept drift
  - Heuristic-based solutions

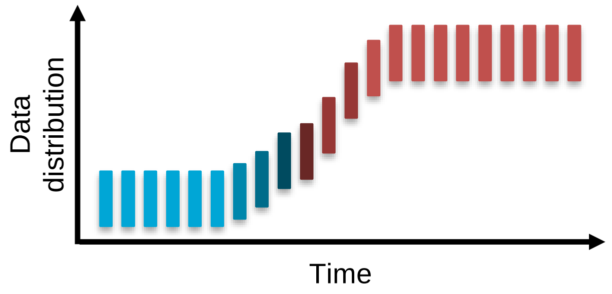


- Sequence clustering while streaming

TU Delft

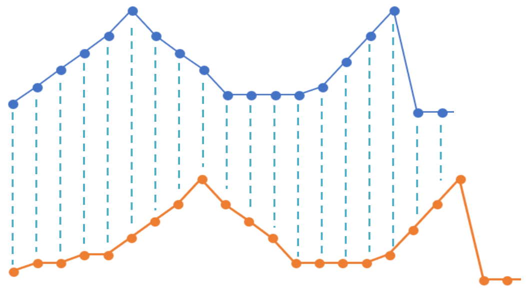
Sequence clustering in streaming environments

- Stream clustering
  - Cluster infinitely many data items
  - Concept drift
  - Heuristic-based solutions



- Sequence clustering while streaming

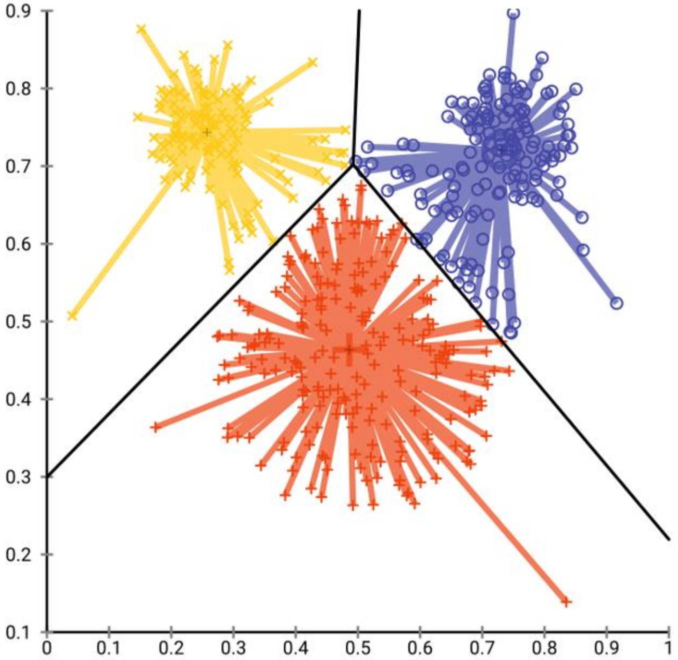
Out-of-sync sequences
  - Alignment-based distances, e.g., DTW
  - Expensive and limited support



TU Delft

K-medoids or Partitioning Around Medoids (PAM)

- K-centroids → actual data items
- Supports non-metric distances

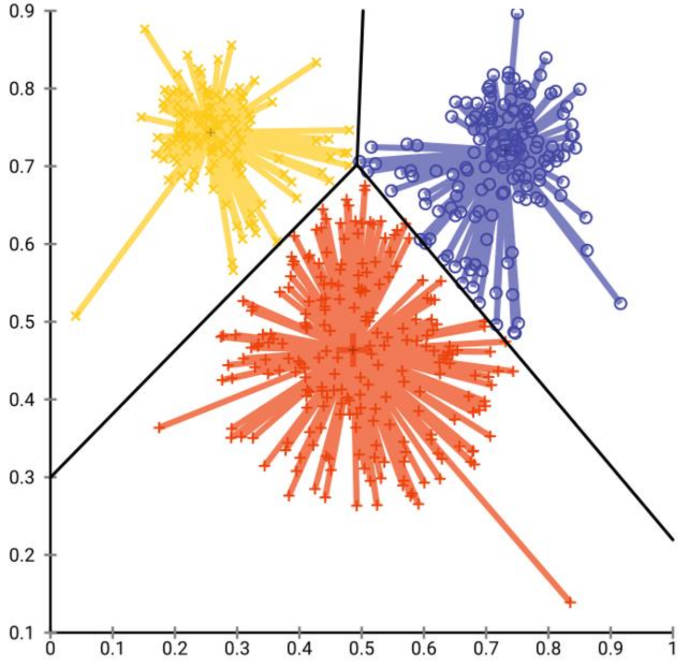


TU Delft

K-medoids or Partitioning Around Medoids (PAM)

- K-centroids → actual data items
- Supports non-metric distances

Computationally expensive for streaming



By Chire - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=82938091

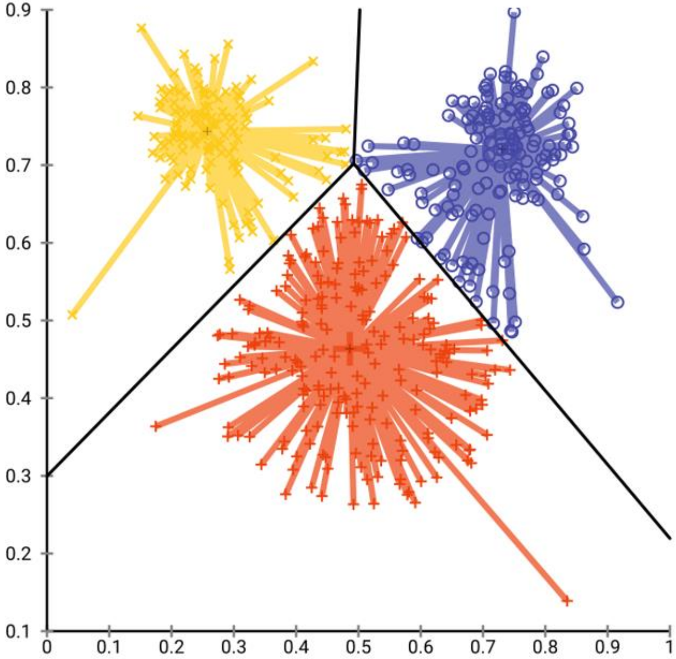
TUDelft

K-medoids or Partitioning Around Medoids (PAM)

- K-centroids → actual data items
- Supports non-metric distances

❌ Computationally expensive for streaming

❌ No support for concept drift



By Chire - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=82938091

TU Delft

SECLEDS: SEquence CLustering in Evolving Data Streams

- Lightweight streaming variant of k-medoids with p-medoids

SECLEDS: SEquence CLustering in Evolving Data Streams

- Lightweight streaming variant of k-medoids with p-medoids

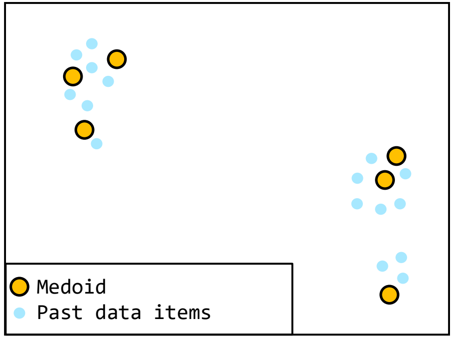
- Important properties
  1. Constant memory footprint
  2. Multiple medoids
  3. Medoid voting scheme

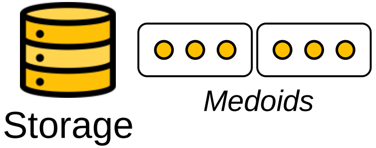
TUDelft

Constant memory footprint

- Only stores the medoids $\rightarrow \mathcal{O}(kp)$

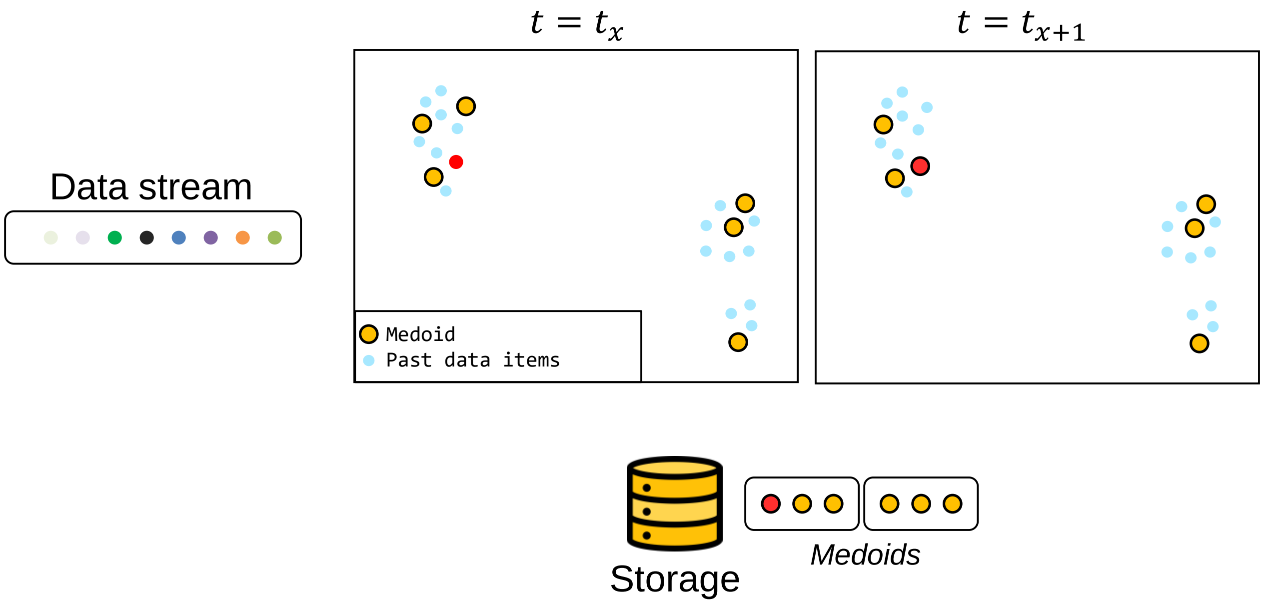
$t = t_{x-1}$





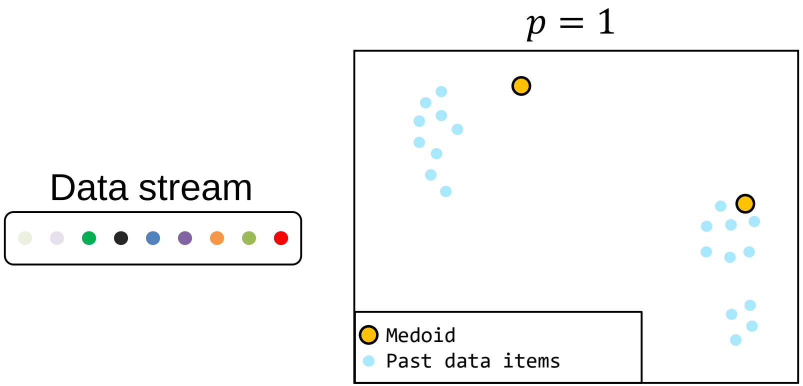
Constant memory footprint

- Only stores the medoids $\rightarrow \mathcal{O}(kp)$



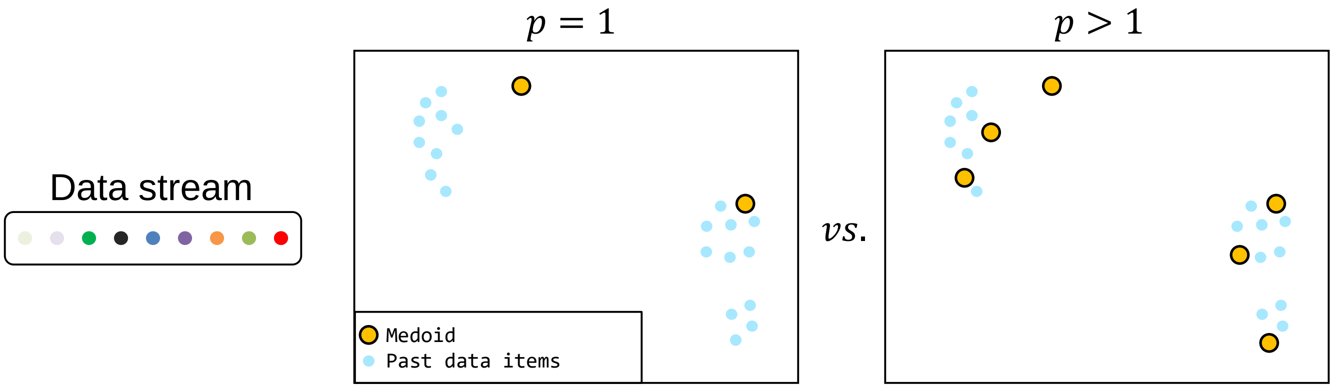
Multiple medoids

- p-medoids represent each cluster
  - *p = configurable parameter*



Multiple medoids

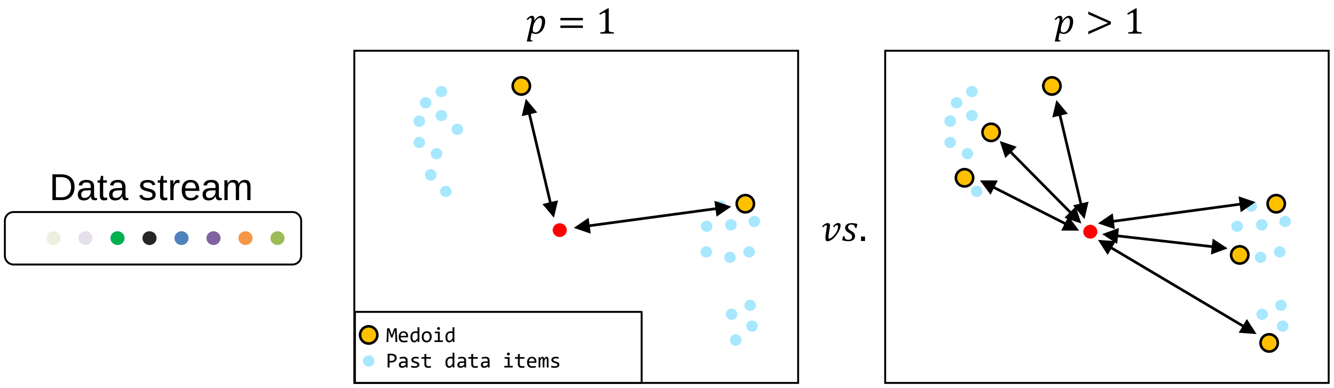
- p-medoids represent each cluster
  - *p = configurable parameter*



Multiple medoids

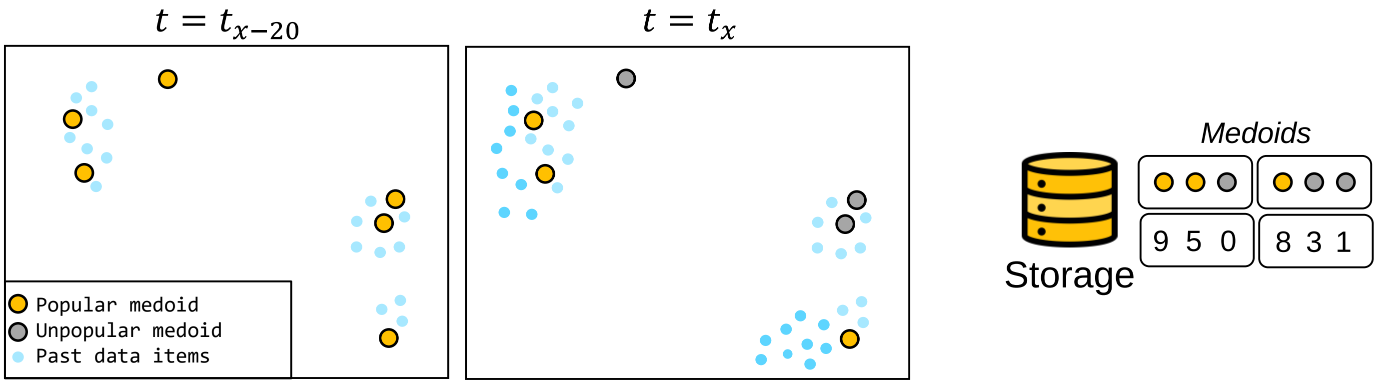
- p-medoids represent each cluster
  - *p = configurable parameter*

- Assignment based on minimum average distance
  - Reduces influence of irrelevant medoids



Medoid voting scheme

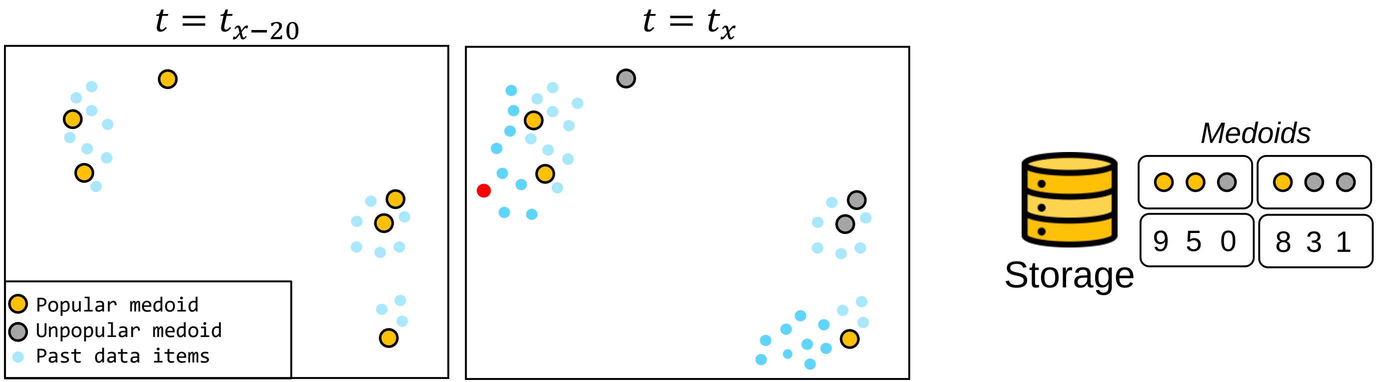
- Votes maintained for each medoid
  - Captures the fraction of close recent data
  - Exponential decay to forget past votes



Medoid voting scheme

- Votes maintained for each medoid
  - Captures the fraction of close recent data
  - Exponential decay to forget past votes

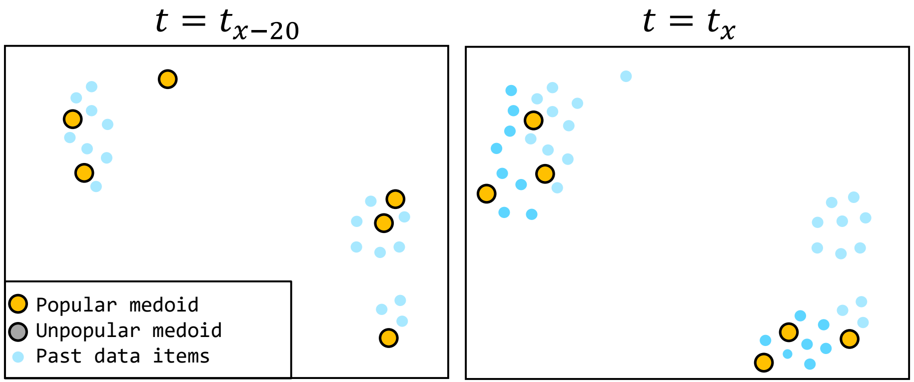
- Which medoids to replace?
  - The one with the least votes → irrelevant medoid

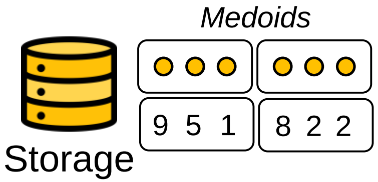


Medoid voting scheme

- Votes maintained for each medoid
  - Captures the fraction of close recent data
  - Exponential decay to forget past votes

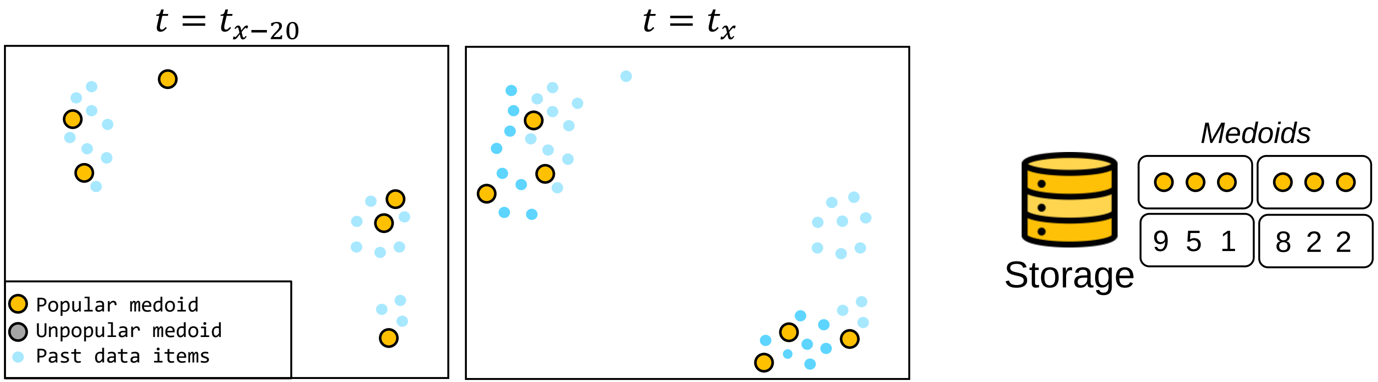
- Which medoids to replace?
  - The one with the least votes → irrelevant medoid





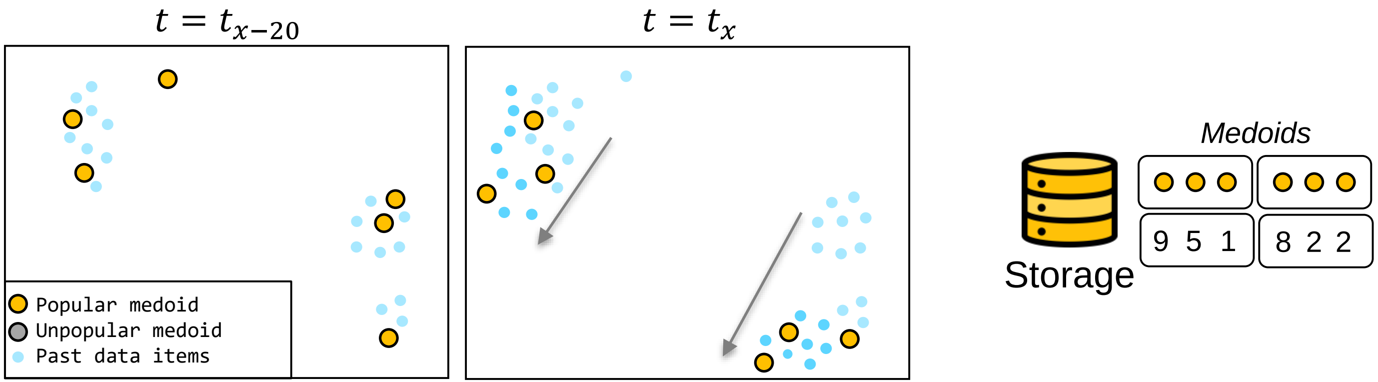
Medoid voting scheme

- Estimates center of mass without distance computations
  - Enables the use of expensive distance measures



Medoid voting scheme

- Estimates center of mass without distance computations
  - Enables the use of expensive distance measures
- Concept drift with k-evolving clusters
  - Keep relevant medoids | Replace irrelevant medoids



SECLEDS

Algorithm 1: SECLEDS for clustering sequences in evolving streams

Input: Data stream, nclusters, nprototypes: $S,\ k,\ p$

function SECLEDS(S, k, p)
  $b \leftarrow 1.5 \cdot k \cdot p$
  $\mathbb{B} \leftarrow$ Collect b items from S
  $\mathcal{C} \leftarrow$ INIT($\mathbb{B},\ k,\ p$) // INIT
  forall s *in* $S[b:]$ **do**
    $cid \leftarrow \arg\min_{1 \leq cid \leq k} \frac{1}{p} \cdot \sum_{j=1}^{p} d(s, m_{cid,j})$ // ASSIGN
    $j \leftarrow \arg\min_j d(s, m_{cid,j})$ for all $1 \leq j \leq p$
    $v_{cid,j} \leftarrow (v_{cid,j} + 1),\ v_{cid,j'} \leftarrow v_{cid,j'} \cdot (1 - \lambda)$ for $j' \neq j$
    $j \leftarrow \arg\min_j v_{cid,j}$ where $m_{cid,j} \neq \eta_{cid}$ for all $1 \leq j \leq p$ // UPDATE
    $m_{cid,j} \leftarrow s,\ v_{cid,j} \leftarrow 0$
    yield cid

function INIT($\mathbb{B}, k, p$)
  Choose $m_{1,1} \in \mathbb{B}$ arbitrarily. Let $C_1 \leftarrow \{(m_{1,1}, 0)\}$
  for $i \leftarrow 2 \ldots k$ **do**
    Choose $m_{i,1} \in \mathbb{B}$ with probability $d(m_{i,1}, m_{1,1})^2$, $m_{i,1} \neq m_{1,1}$
    Let $C_i \leftarrow \{(m_{i,1}, 0)\}$
  for $i \leftarrow 1 \ldots k$ **do**
    $dist \leftarrow d(b, m_{i,1})$ for all $b \in \mathbb{B}$ and $b \neq m_{i,1}$
    Choose $\{m_{i,2} \ldots m_{i,p}\}$ having smallest values in $dist$
    Update $C_i \leftarrow \{(m_{i,1}, 0) \ldots (m_{i,p}, 0)\}$
  return $\{C_1, \ldots, C_k\}$

TU Delft

SECLEDS

- From a batch, initialize clusters using a non-uniform sampling strategy

Algorithm 1: SECLEDS for clustering sequences in evolving streams

Input: Data stream, nclusters, nprototypes: $S,\ k,\ p$

function SECLEDS(S, k, p)
  $b \leftarrow 1.5 \cdot k \cdot p$
  $\mathbb{B} \leftarrow$ Collect b items from S
  $\mathcal{C} \leftarrow$ INIT$(\mathbb{B},\ k,\ p)$ // INIT
  forall s *in* $S[b:]$ **do**
    $cid \leftarrow \arg\min_{1 \le cid \le k} \frac{1}{p} \cdot \sum_{j=1}^{p} d(s, m_{cid,j})$ // ASSIGN
    $j \leftarrow \arg\min_j d(s, m_{cid,j})$ for all $1 \le j \le p$
    $v_{cid,j} \leftarrow (v_{cid,j} + 1),\ v_{cid,j'} \leftarrow v_{cid,j'} \cdot (1 - \lambda)$ for $j' \neq j$
    $j \leftarrow \arg\min_j v_{cid,j}$ where $m_{cid,j} \neq \eta_{cid}$ for all $1 \le j \le p$ // UPDATE
    $m_{cid,j} \leftarrow s,\ v_{cid,j} \leftarrow 0$
    yield cid

function INIT$(\mathbb{B}, k, p)$
  Choose $m_{1,1} \in \mathbb{B}$ arbitrarily. Let $C_1 \leftarrow \{(m_{1,1}, 0)\}$
  for $i \leftarrow 2 \ldots k$ **do**
    Choose $m_{i,1} \in \mathbb{B}$ with probability $d(m_{i,1}, m_{1,1})^2$, $m_{i,1} \neq m_{1,1}$
    Let $C_i \leftarrow \{(m_{i,1}, 0)\}$
  for $i \leftarrow 1 \ldots k$ **do**
    $dist \leftarrow d(b, m_{i,1})$ for all $b \in \mathbb{B}$ and $b \neq m_{i,1}$
    Choose $\{m_{i,2} \ldots m_{i,p}\}$ having smallest values in $dist$
    Update $C_i \leftarrow \{(m_{i,1}, 0) \ldots (m_{i,p}, 0)\}$
  return $\{C_1, \ldots, C_k\}$

TU Delft

SECLEDS

- From a batch, initialize clusters using a non-uniform sampling strategy

Stream loop starts

Algorithm 1: SECLEDS for clustering sequences in evolving streams

Input: Data stream, nclusters, nprototypes: $S,\ k,\ p$

function SECLEDS(S, k, p)
- $b \leftarrow 1.5 \cdot k \cdot p$
- $\mathbb{B} \leftarrow$ Collect b items from S
- $\mathcal{C} \leftarrow$ Init$(\mathbb{B},\ k,\ p)$ // Init
- **forall** s *in* $S[b:]$ **do**
  - $cid \leftarrow \arg\min_{1 \le cid \le k} \frac{1}{p} \cdot \sum_{j=1}^{p} d(s, m_{cid,j})$ // Assign
  - $j \leftarrow \arg\min_j d(s, m_{cid,j})$ for all $1 \le j \le p$
  - $v_{cid,j} \leftarrow (v_{cid,j} + 1),\ v_{cid,j'} \leftarrow v_{cid,j'} \cdot (1 - \lambda)$ for $j' \neq j$
  - $j \leftarrow \arg\min_j v_{cid,j}$ where $m_{cid,j} \neq \eta_{cid}$ for all $1 \le j \le p$ // Update
  - $m_{cid,j} \leftarrow s,\ v_{cid,j} \leftarrow 0$
  - **yield** *cid*

function Init$(\mathbb{B}, k, p)$
- Choose $m_{1,1} \in \mathbb{B}$ arbitrarily. Let $C_1 \leftarrow \{(m_{1,1}, 0)\}$
- **for** $i \leftarrow 2 \ldots k$ **do**
  - Choose $m_{i,1} \in \mathbb{B}$ with probability $d(m_{i,1}, m_{1,1})^2$, $m_{i,1} \neq m_{1,1}$
  - Let $C_i \leftarrow \{(m_{i,1}, 0)\}$
- **for** $i \leftarrow 1 \ldots k$ **do**
  - $dist \leftarrow d(b, m_{i,1})$ for all $b \in \mathbb{B}$ and $b \neq m_{i,1}$
  - Choose $\{m_{i,2} \ldots m_{i,p}\}$ having smallest values in $dist$
  - Update $C_i \leftarrow \{(m_{i,1}, 0) \ldots (m_{i,p}, 0)\}$
- **return** $\{C_1, \ldots, C_k\}$

TU Delft

SECLEDS

- From a batch, initialize clusters using a non-uniform sampling strategy

Stream loop starts

- Assign
  - Assign data item to the cluster with the least average distance to the medoids
  - Cast a vote for the closest medoid

Algorithm 1: SECLEDS for clustering sequences in evolving streams

Input: Data stream, nclusters, nprototypes: S, k, p

function SECLEDS(S, k, p)
  $b \leftarrow 1.5 \cdot k \cdot p$
  $\mathbb{B} \leftarrow$ Collect b items from S
  $\mathcal{C} \leftarrow \textsc{Init}(\mathbb{B}, k, p)$ // Init
  forall s *in* $S[b:]$ **do**
    $cid \leftarrow \arg\min_{1 \le cid \le k} \frac{1}{p} \cdot \sum_{j=1}^{p} d(s, m_{cid,j})$ // Assign
    $j \leftarrow \arg\min_j d(s, m_{cid,j})$ for all $1 \le j \le p$
    $v_{cid,j} \leftarrow (v_{cid,j} + 1)$, $v_{cid,j'} \leftarrow v_{cid,j'} \cdot (1 - \lambda)$ for $j' \neq j$
    $j \leftarrow \arg\min_j v_{cid,j}$ where $m_{cid,j} \neq \eta_{cid}$ for all $1 \le j \le p$ // Update
    $m_{cid,j} \leftarrow s$, $v_{cid,j} \leftarrow 0$
    yield cid

function $\textsc{Init}(\mathbb{B}, k, p)$
  Choose $m_{1,1} \in \mathbb{B}$ arbitrarily. Let $C_1 \leftarrow \{(m_{1,1}, 0)\}$
  for $i \leftarrow 2 \ldots k$ **do**
    Choose $m_{i,1} \in \mathbb{B}$ with probability $d(m_{i,1}, m_{1,1})^2$, $m_{i,1} \neq m_{1,1}$
    Let $C_i \leftarrow \{(m_{i,1}, 0)\}$
  for $i \leftarrow 1 \ldots k$ **do**
    $dist \leftarrow d(b, m_{i,1})$ for all $b \in \mathbb{B}$ and $b \neq m_{i,1}$
    Choose $\{m_{i,2} \ldots m_{i,p}\}$ having smallest values in $dist$
    Update $C_i \leftarrow \{(m_{i,1}, 0) \ldots (m_{i,p}, 0)\}$
  return $\{C_1, \ldots, C_k\}$

SECLEDS

- From a batch, initialize clusters using a non-uniform sampling strategy

Stream loop starts

- Assign
  - Assign data item to the cluster with the least average distance to the medoids
  - Cast a vote for the closest medoid
- Update
  - Upgrade data item as a medoid by replacing the one with the least votes

Algorithm 1: SECLEDS for clustering sequences in evolving streams

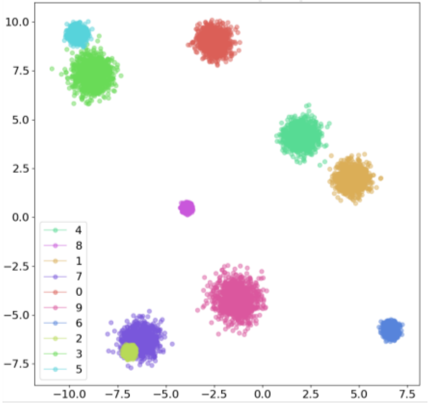
Input: Data stream, nclusters, nprototypes: $S,\ k,\ p$

function SECLEDS(S, k, p)
  $b \leftarrow 1.5 \cdot k \cdot p$
  $\mathbb{B} \leftarrow$ Collect b items from S
  $\mathcal{C} \leftarrow \textsc{Init}(\mathbb{B},\ k,\ p)$ // Init
  forall s *in* $S[b:]$ **do**
    $cid \leftarrow \arg\min_{1 \le cid \le k} \frac{1}{p} \cdot \sum_{j=1}^{p} d(s, m_{cid,j})$ // Assign
    $j \leftarrow \arg\min_j d(s, m_{cid,j})$ for all $1 \le j \le p$
    $v_{cid,j} \leftarrow (v_{cid,j} + 1),\ v_{cid,j'} \leftarrow v_{cid,j'} \cdot (1 - \lambda)$ for $j' \neq j$
    $j \leftarrow \arg\min_j v_{cid,j}$ where $m_{cid,j} \neq \eta_{cid}$ for all $1 \le j \le p$ // Update
    $m_{cid,j} \leftarrow s,\ v_{cid,j} \leftarrow 0$
    yield cid

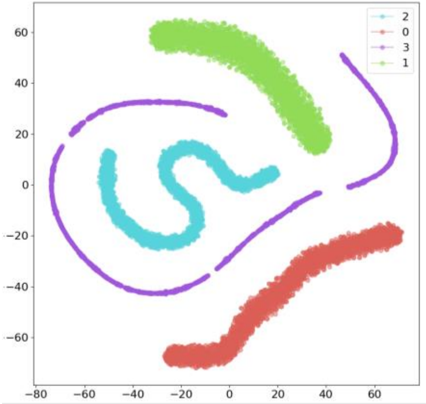
function $\textsc{Init}(\mathbb{B}, k, p)$
  Choose $m_{1,1} \in \mathbb{B}$ arbitrarily. Let $C_1 \leftarrow \{(m_{1,1}, 0)\}$
  for $i \leftarrow 2 \ldots k$ **do**
    Choose $m_{i,1} \in \mathbb{B}$ with probability $d(m_{i,1}, m_{1,1})^2$, $m_{i,1} \neq m_{1,1}$
    Let $C_i \leftarrow \{(m_{i,1}, 0)\}$
  for $i \leftarrow 1 \ldots k$ **do**
    $dist \leftarrow d(b, m_{i,1})$ for all $b \in \mathbb{B}$ and $b \neq m_{i,1}$
    Choose $\{m_{i,2} \ldots m_{i,p}\}$ having smallest values in $dist$
    Update $C_i \leftarrow \{(m_{i,1}, 0) \ldots (m_{i,p}, 0)\}$
  return $\{C_1, \ldots, C_k\}$

Datasets

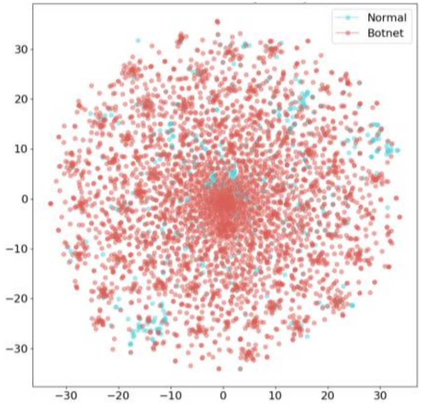
2D: Blobs *(k=10)*



Seq: Sine curves *(k=4)*

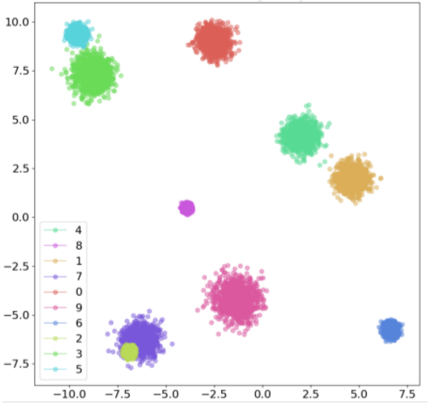


Seq: Network traffic *(k=2)*

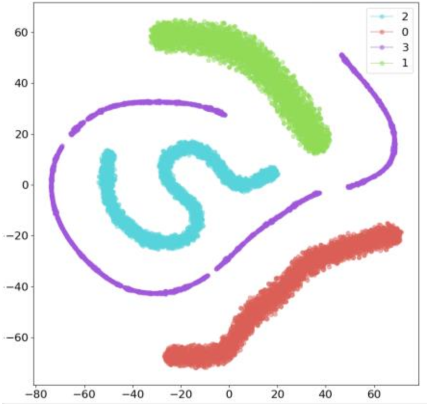


Datasets

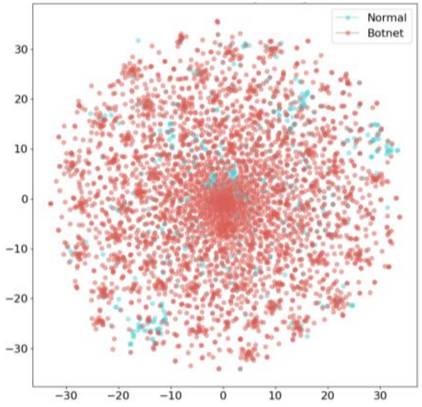
2D: Blobs *(k=10)*



Seq: Sine curves *(k=4)*



Seq: Network traffic *(k=2)*



Baselines

Streaming: CluStream[1], StreamKM++

Batch: Minibatch k-means

Offline: BanditPAM[2]

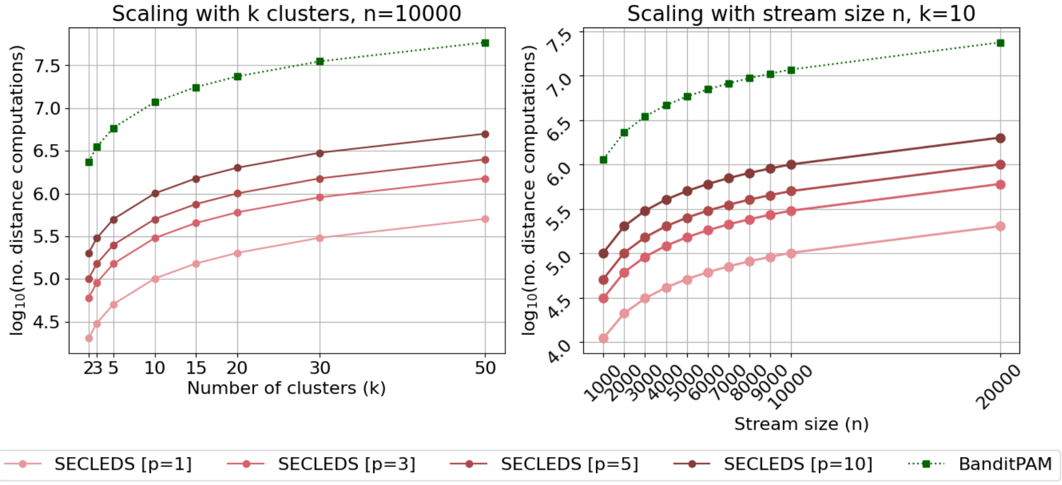
TU Delft

1. Aggarwal, Charu C., et al. "A framework for clustering evolving data streams." Proceedings 2003 VLDB conference. Morgan Kaufmann, 2003.
2. Tiwari, Mo, et al. "Banditpam: Almost linear time k-medoids clustering via multi-armed bandits." Advances in Neural Information Processing Systems 33 (2020): 10211-10222.

Results [1/4]: Distance computations

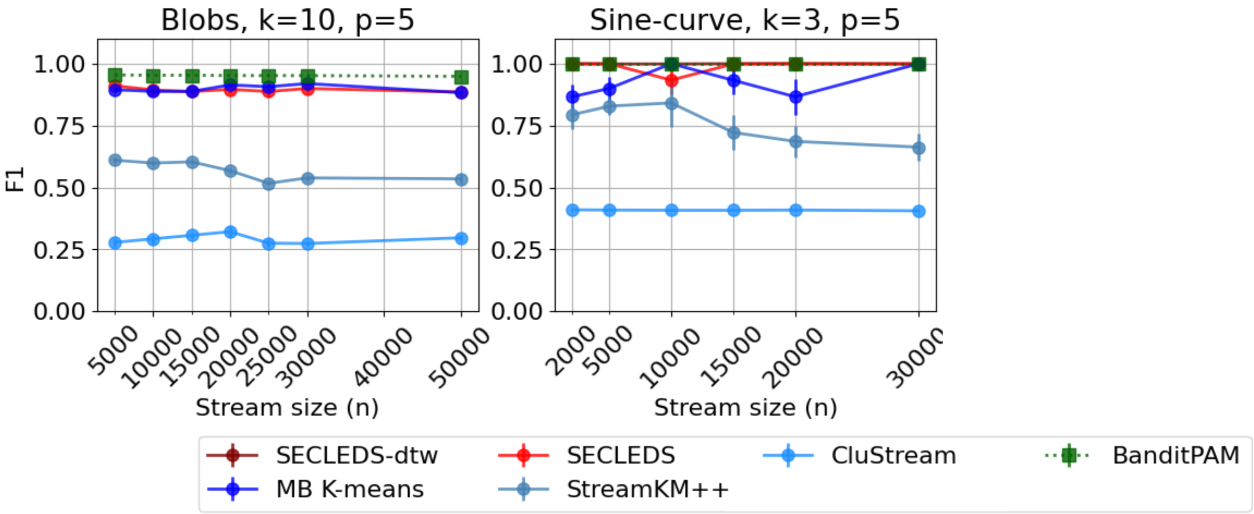
Results [1/4]: Distance computations

- Reduces required distance computations by **83.7%**



Results [2/4]: Performance

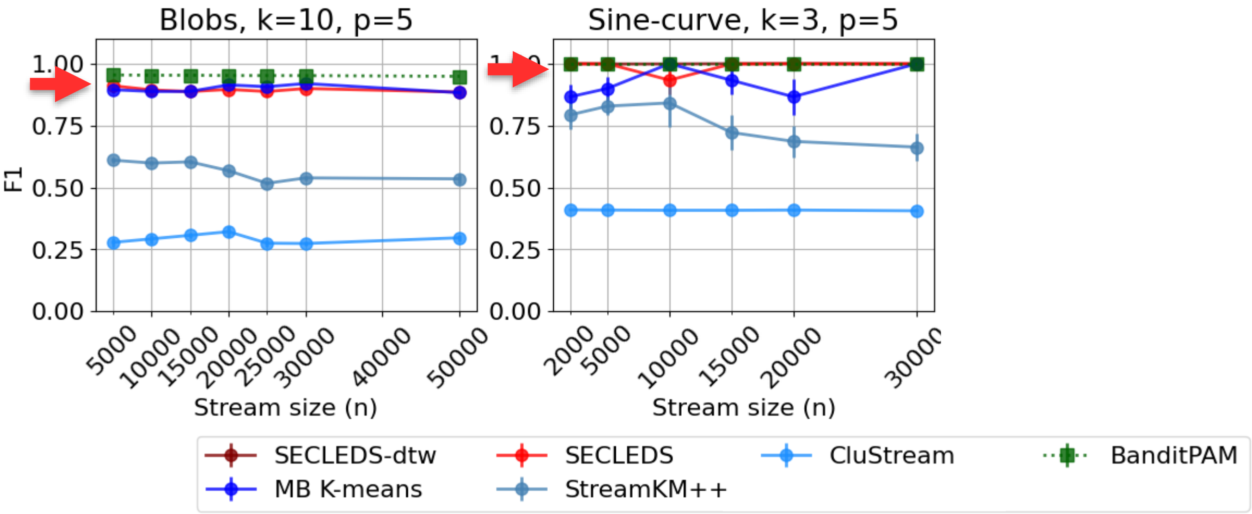
- Outperforms all streaming baselines without concept drift
  - Competitive performance with BanditPAM



TU Delft

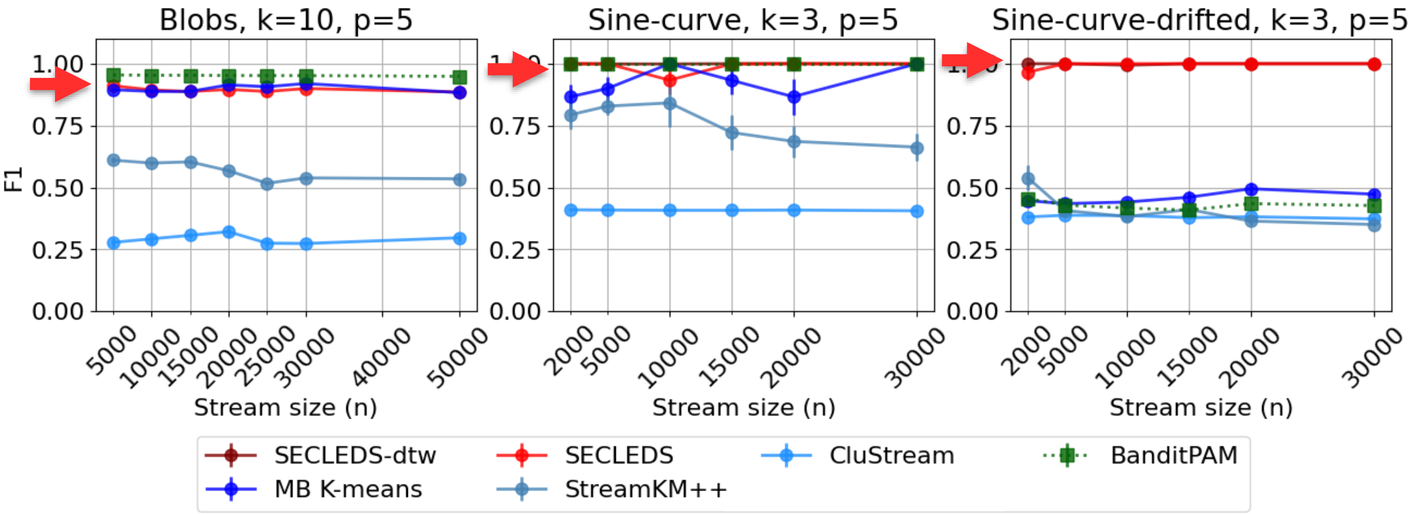
Results [2/4]: Performance

- Outperforms all streaming baselines without concept drift
  - Competitive performance with BanditPAM



Results [2/4]: Performance

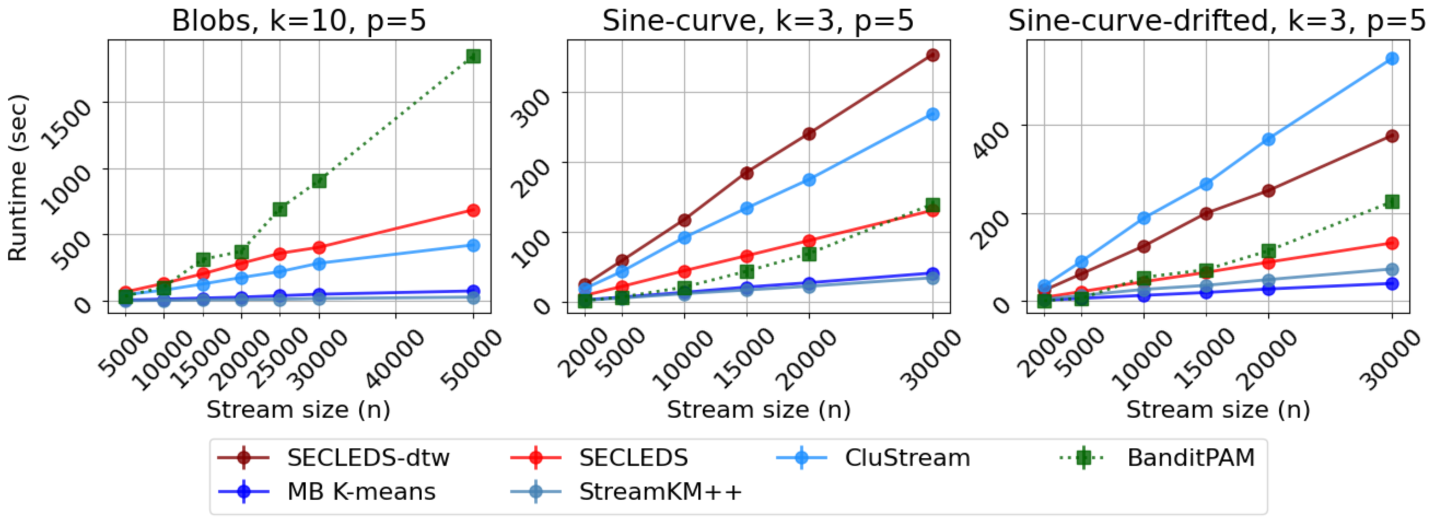
- Outperforms all streaming baselines without concept drift
  - Competitive performance with BanditPAM
- Outperforms all baselines by **138.7%** with concept drift



TU Delft

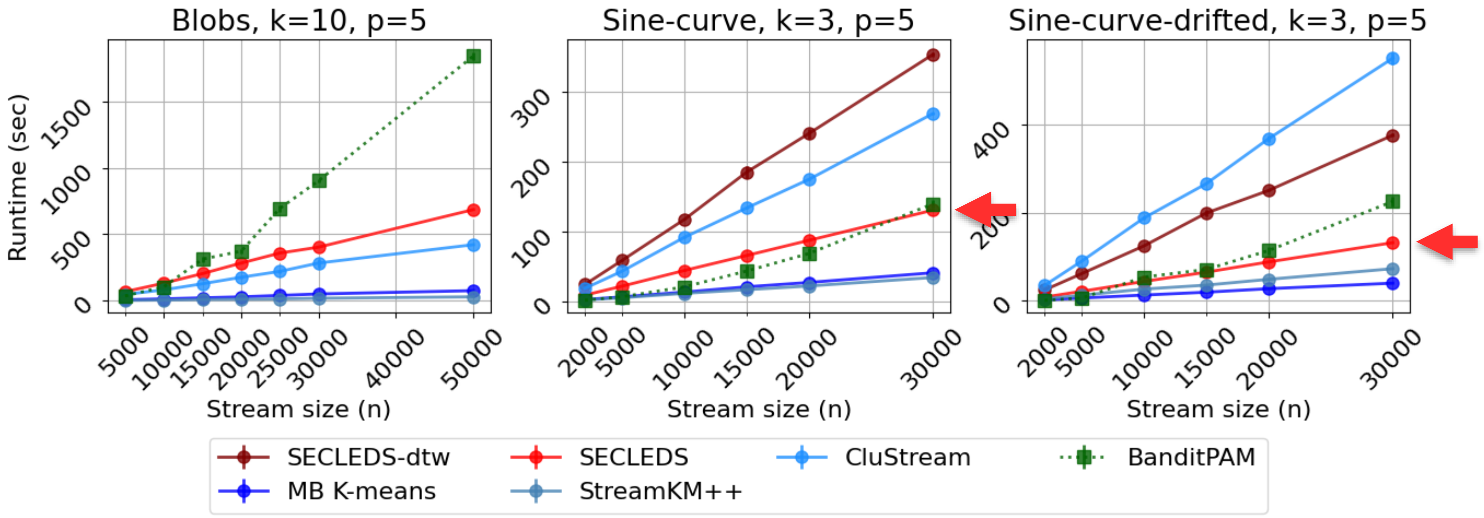
Results [3/4]: Runtime

- Runtime scales almost linearly *w.r.t.* stream size → $\mathcal{O}(nkp)$



Results [3/4]: Runtime

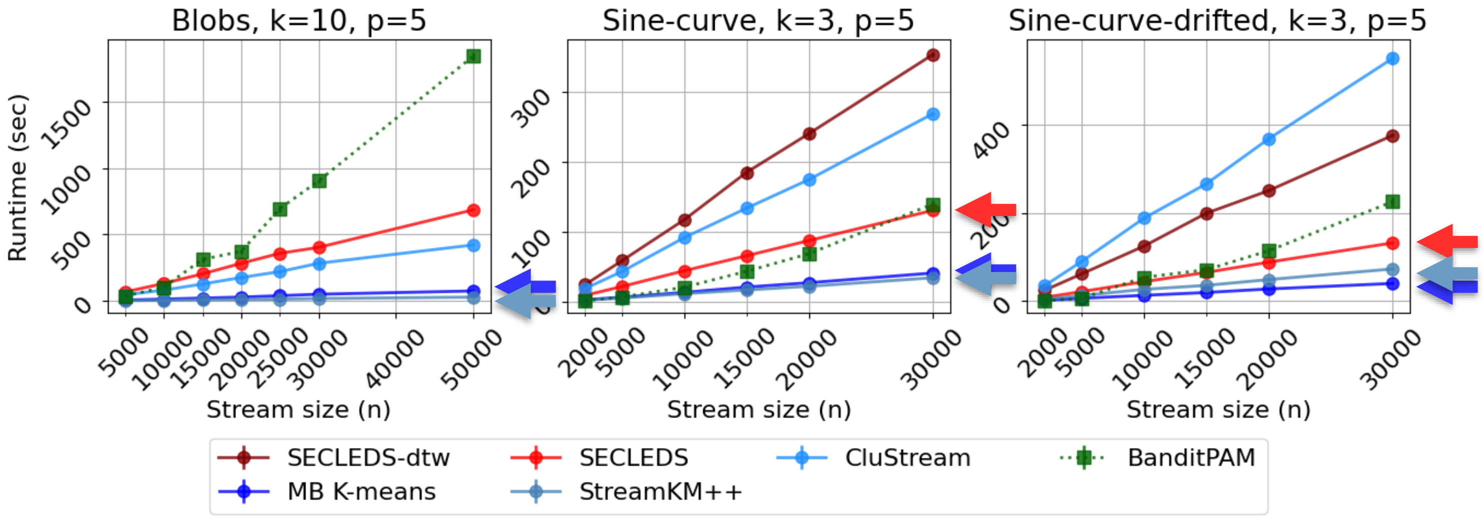
- Runtime scales almost linearly *w.r.t.* stream size → $\mathcal{O}(nkp)$
- Faster than banditPAM and CluStream for sequence clustering



TU Delft

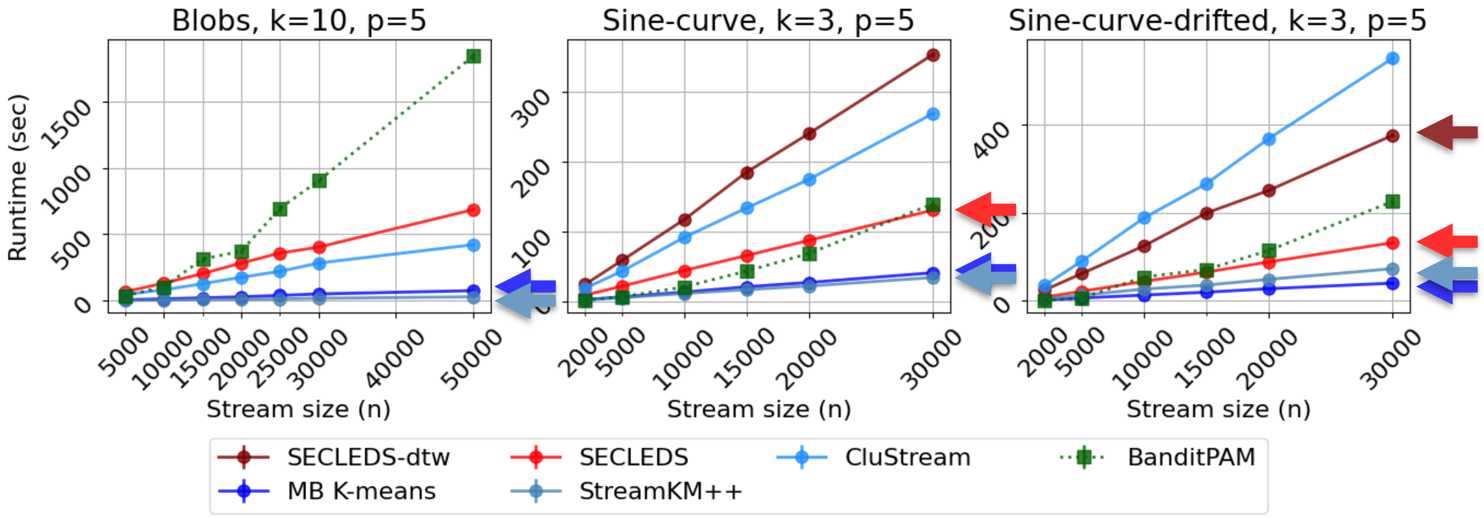
Results [3/4]: Runtime

- Runtime scales almost linearly *w.r.t.* stream size → $\mathcal{O}(nkp)$
- Faster than banditPAM and CluStream for sequence clustering



Results [3/4]: Runtime

- Runtime scales almost linearly *w.r.t.* stream size → $\mathcal{O}(nkp)$
- Faster than banditPAM and CluStream for sequence clustering



Results [4/4]: Network traffic use case

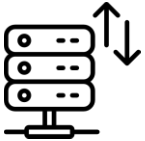
- Temporal pattern preserving network traffic sampling
  - Periodically store medoids

	# distances (10^6)	Runtime (sec)	F1 (k=2)	F1 (k=5)
BanditPAM	10.3	984.8	0.64	0.38
SECLEDS-eu	**2.1**	**631.8**	0.79	0.76
SECLEDS-dtw	**2.1**	1626.89	**0.81**	**0.80**

TU Delft

Results [4/4]: Network traffic use case

- Temporal pattern preserving network traffic sampling
  - Periodically store medoids
- Clusters 5.5h network traffic in **< 30 minutes**
  - Supporting network bandwidth of up to **1.08 Gb/s** with DTW



	# distances (10^6)	**Runtime (sec)**	**F1 (k=2)**	**F1 (k=5)**
BanditPAM	10.3	984.8	0.64	0.38
SECLEDS-eu	**2.1**	**631.8**	0.79	0.76
SECLEDS-dtw	**2.1**	1626.89	**0.81**	**0.80**

TU Delft

Future work

- Medoid-aware clustering
  - Better coverage, but additional distance computations required
- Theoretical properties of SECLEDS
  - Empirically works but limited theoretical understanding
- Optimized implementation
  - Runtime efficient C implementation

Takeways

- SECLEDS → lightweight streaming k-medoids with multiple medoids
  - Medoid voting → concept drift using k-evolving clusters
- Cluster sequences in high-bandwidth data streams
  - Reduces required distance computations
- Exceptional clustering performance with concept drift
  - Outperforms all streaming baselines regardless of concept drift
- SECLEDS can be used to intelligently sample network traffic

TU Delft

Thank you! Questions?

- SECLEDS is a lightweight streaming variant of k-medoids.
- Efficiently clusters sequences in high-bandwidth data streams with concept drift.
- Can be used for temporal pattern-preserving network traffic sampling.

SECLEDS is open-source!

azqa.nadeem@tudelft.nl @azqa_nadeem https://cyber-analytics.nl/

TUDelft

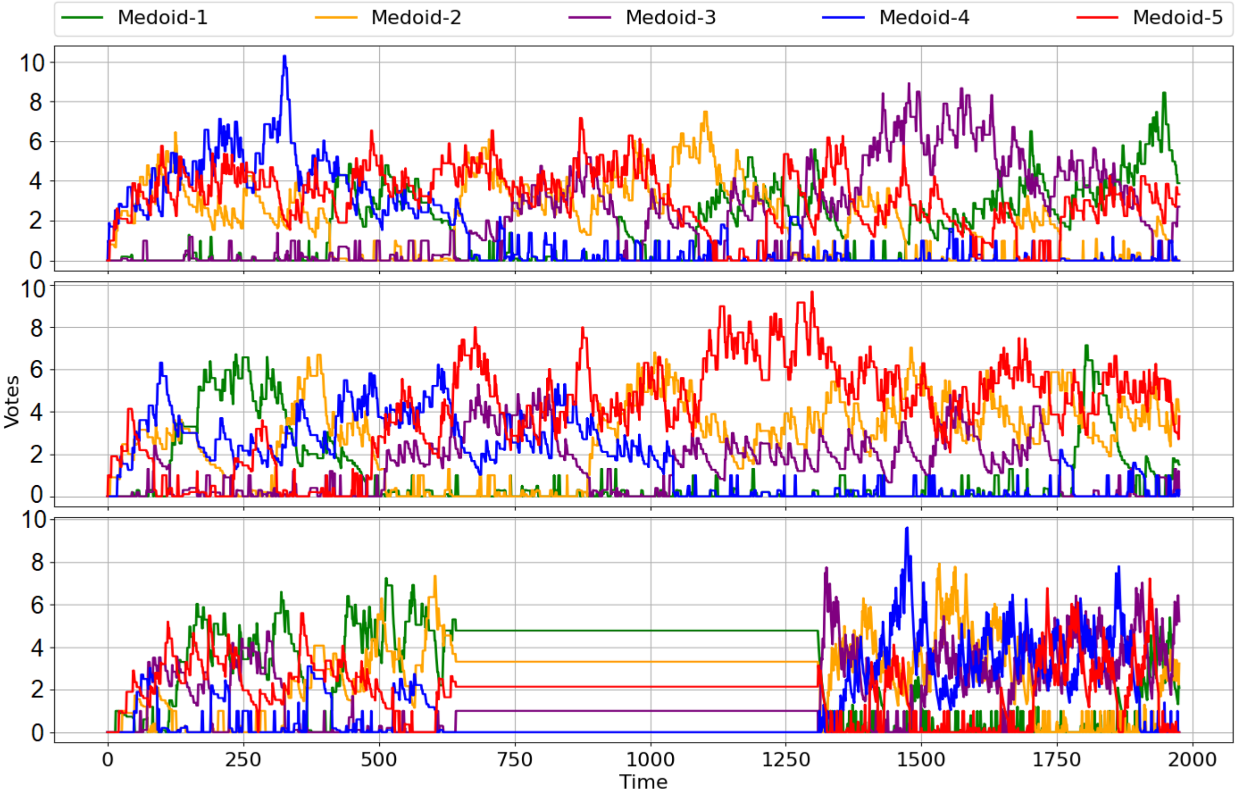
*Icons courtesy Smashicons, Pixel perfect, Freepik, Roundicons from www.flaticons.com

Extras

Uniformly distributed stream

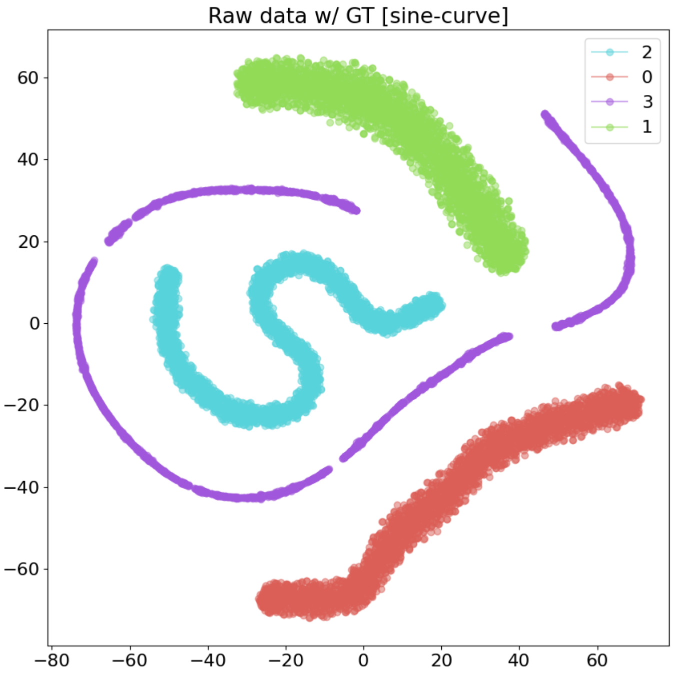
Incrementally drifted stream

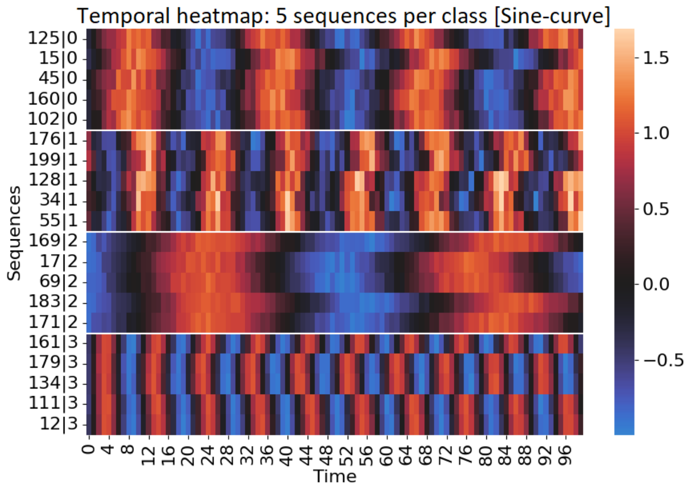
Class-ordered stream

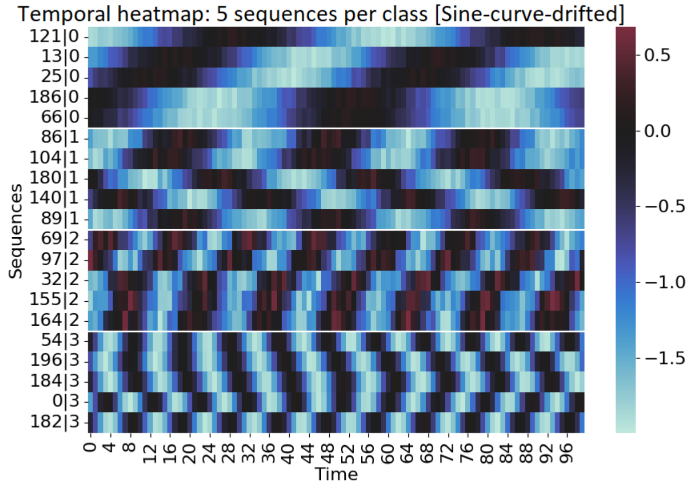


TU Delft

Extras

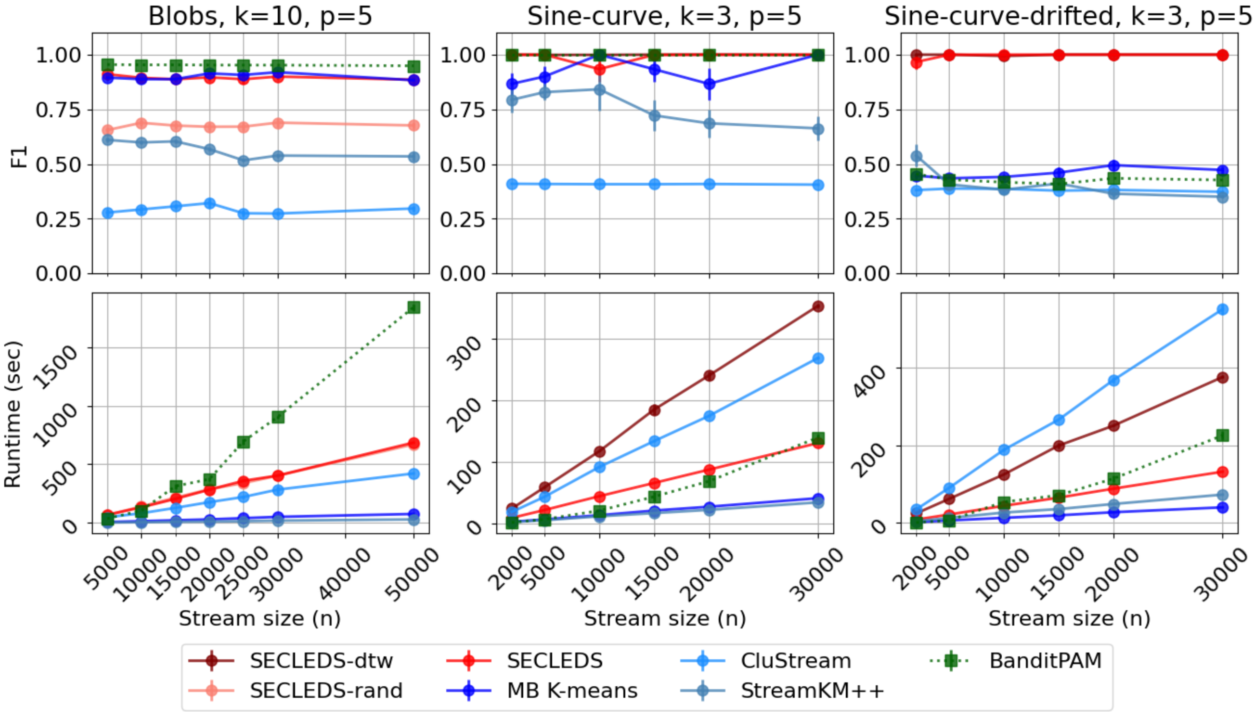






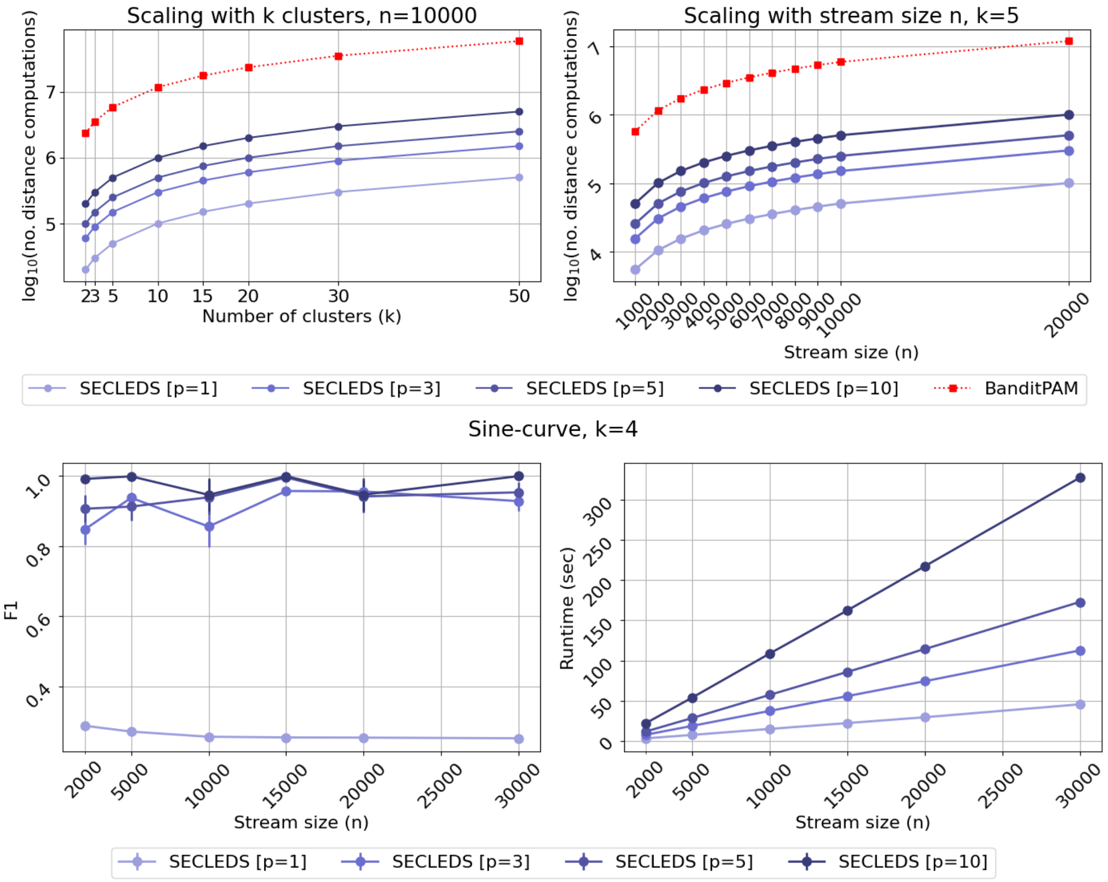
TU Delft

Extras



Blobs, k=10, p=5
Sine-curve, k=3, p=5
Sine-curve-drifted, k=3, p=5
F1
Runtime (sec)
Stream size (n)
SECLEDS-dtw
SECLEDS-rand
SECLEDS
MB K-means
CluStream
StreamKM++
BanditPAM

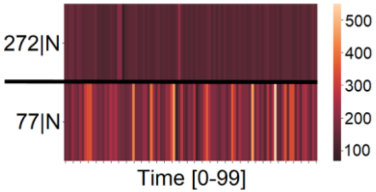
Extras



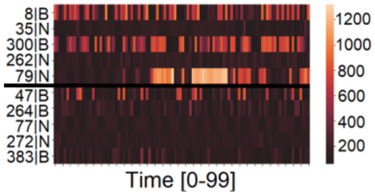
Scaling with k clusters, n=10000
log10(no. distance computations)
Number of clusters (k)
Scaling with stream size n, k=5
log10(no. distance computations)
Stream size (n)
SECLEDS [p=1]
SECLEDS [p=3]
SECLEDS [p=5]
SECLEDS [p=10]
BanditPAM
Sine-curve, k=4
F1
Stream size (n)
Runtime (sec)
Stream size (n)
SECLEDS [p=1]
SECLEDS [p=3]
SECLEDS [p=5]
SECLEDS [p=10]

Extras

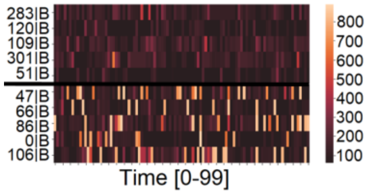
k=2



BanditPAM

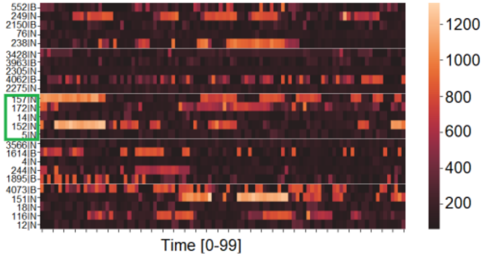


SECLEDS-eu

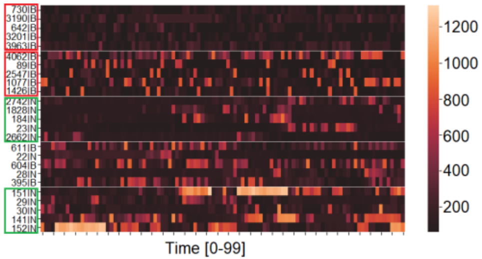


SECLEDS-dtw

k=5



SECLEDS-eu



SECLEDS-dtw

TU Delft