RPS: A Generic Reservoir Patterns Sampler

Lamine DIOP*, Marc PLANTEVIT*, Arnaud SOULET+

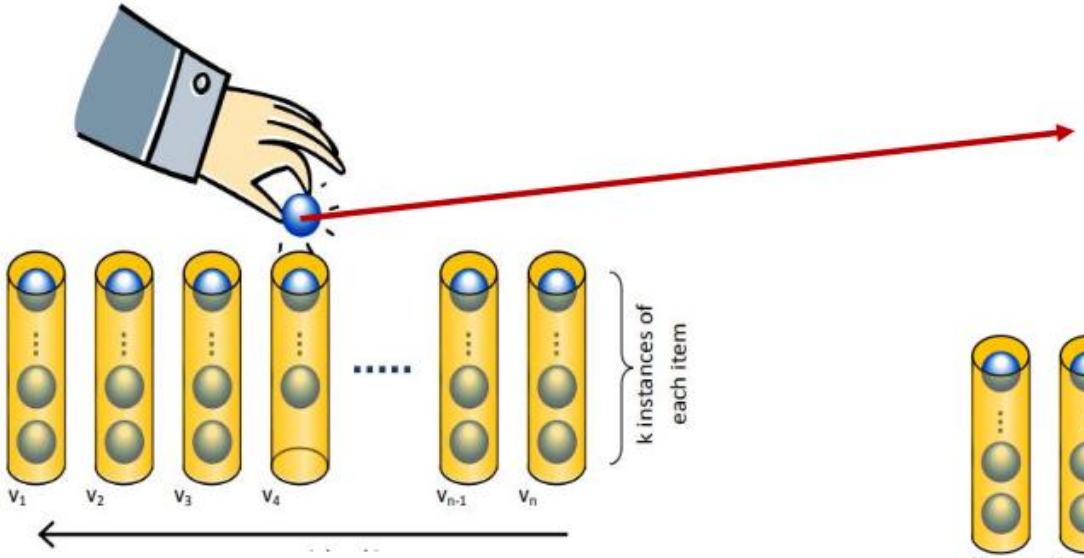
(*) EPITA Research Laboratory (LRE), France

(+) University of Tours, LIFAT, France

Combinatorics and Life Sciences
University of Lyon 1
October 2, 2025

Reservoir Pattern Sampling Problem

WRS-k-W: Weighted Random Sampling over Data Streams



[Pavlos S. Efraimidis, 2015]

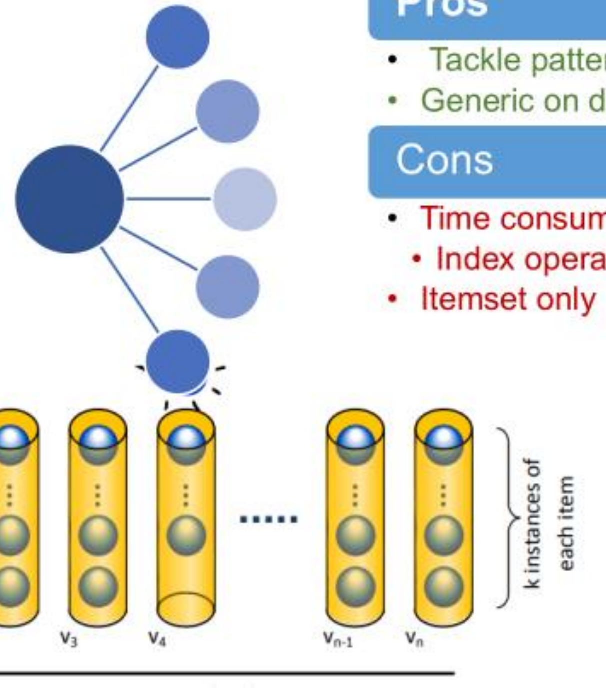
ResPat: Reservoir **Patterns** Sampling in Data Streams

Pros

- Tackle patterns
- Generic on damping factor

Cons

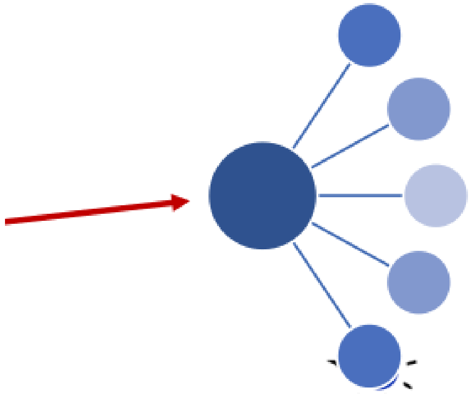
- Time consuming
  - Index operator
- Itemset only



[Giacometti & Soulet, ECMLPKDD'22]

Batch-based Reservoir Pattern Sampling

ResPat: Reservoir **Patterns** Sampling in Data Streams

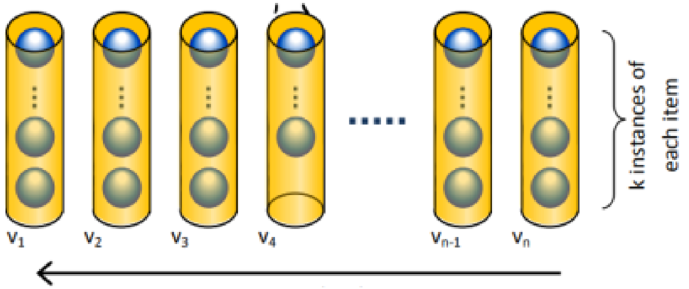


Pros

- Tackle patterns
- Generic on damping factor

Cons

- Time consuming
  - Index operator
- Itemset only



[Giacometti & Soulet, ECMLPKDD'22]

RPS: Our contribution

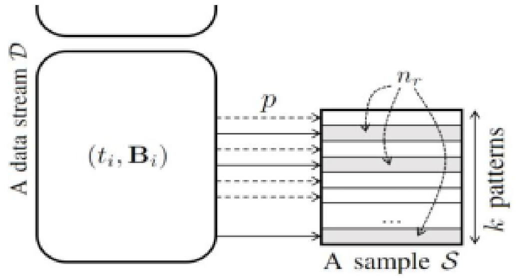
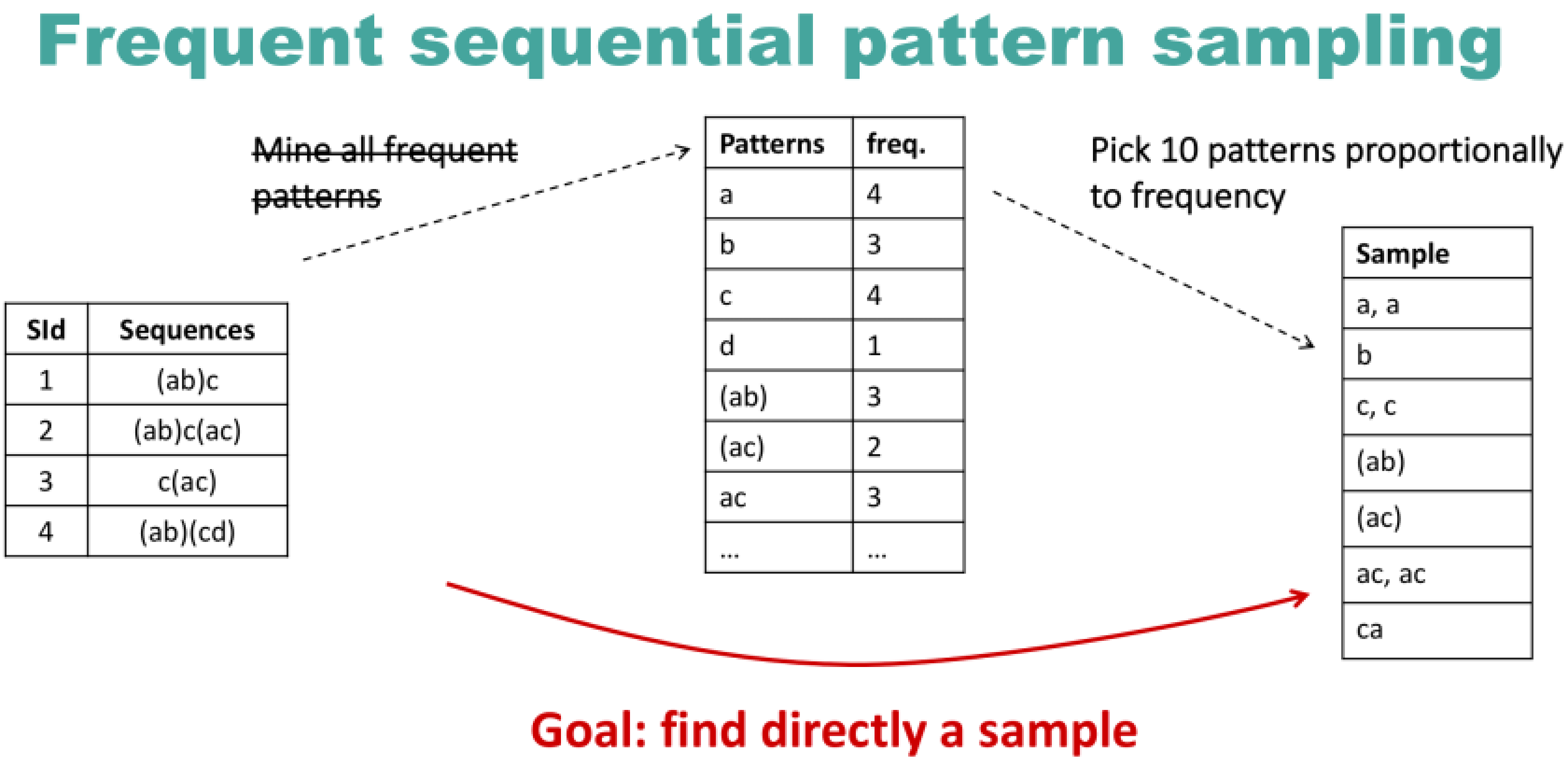


Fig. 1: Overview of the approach *(the incomplete block denotes next batch)*

Property 2 (From CBPD to IBF). *Let k be the total number of trials, $n_r \leq k$ the minimum number of times the event must occur, and p the probability of the event occurring in a single trial. The Cumulative Binomial Probability Distribution can be computed as follows:*

$$\mathbb{P}(n_r, k) \equiv \sum^{k} \binom{k}{j} p^j (1-p)^{k-j} = \mathbf{I}_p(n_r, k-n_r+1). \quad (2)$$

Direct Local Pattern Sampling



Frequent sequential pattern sampling

SId	Sequences
1	(ab)c
2	(ab)c(ac)
3	c(ac)
4	(ab)(cd)

~~Mine all frequent patterns~~

Patterns	freq.
a	4
b	3
c	4
d	1
(ab)	3
(ac)	2
ac	3
…	…

Pick 10 patterns proportionally to frequency

Sample
a, a
b
c, c
(ab)
(ac)
ac, ac
ca

Goal: find directly a sample

Pattern sampling state-of-the-art

Language

	MCMC	SAT	Two-step random
Graphs	[Hasan & Zaki, 2009]		[Diop et Plantevit, IEEEBigData'24]
Sequences			[Diop et al., ICDM'2018] [Diop et al., KAIS'2019] [Diop & Ba, IEEEBigData'2021]
Transactions	[Boley *et al.*, 2010] [Li et Zaki, 2012] [Bendimerad et al.,2020]	[Dzyuba et al., 2017] [Gueguen *et al.*, 2019]	[Boley et al., SIGKDD'2011] [Boley *et al.*, 2012] [Moens & Boley, 2014] [Diop et al., ADBIS'2020] [Diop et al., IS'2022]
Numerical data			[Giacometti & Soulet, 2018] [Diop, PAKDD2022] [Djawad et al., DKE'25 (under submission)]

Approach

Challenges in Complex Data Structure

Challenges for subsequences

Challenge 1: compute the number of subsequences

SId	Sequences
1	(ab)c
2	**(ab)c(ac)**
3	c(ac)
4	(ab)(cd)

#subseq.
6
12
5
10

1. Pick a sequence proportionally to its number of subsequences

Challenge 2: do not favor subsequence having multiple occurrences

TId	Subsequences
1	a, b, (ab), ac, bc
2	**a, b, c, (ab), (ac), ac, bc,…**
3	c, a, (ac), ca, cc
4	a, b, c, d, (ab), (cd),…

Sample
a, a
b
c, c
(ab)
(ac)
ac, ac
ca

2. Pick a subsequence uniformly within the drawn sequence

Example

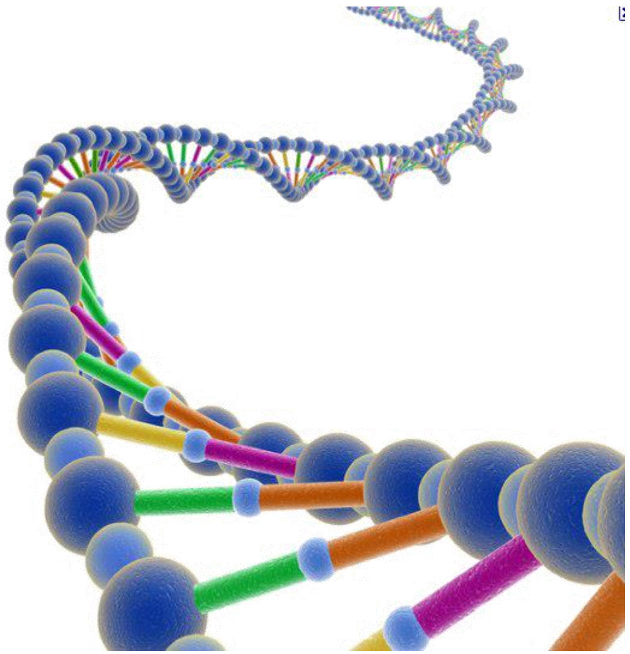
$$\phi_{\le 2}(\langle(ab)c\rangle \circ (c)) \neq \phi_{\le 2}(\langle(ab)c\rangle)\times\binom{1}{0} + \phi_{\le 1}(\langle(ab)c\rangle)\times\binom{1}{1}$$

With repetitions, some subsequences are counted multiple times

$$\left.\begin{array}{c}\langle\quad\rangle,\\ \langle a\rangle,\langle b\rangle,\langle c\rangle,\\ \langle(ab)\rangle,\langle ac\rangle,\langle bc\rangle\end{array}\right\}\circ\emptyset \quad\Big|\quad \left.\begin{array}{c}\langle\quad\rangle\\ \langle a\rangle,\langle b\rangle,\langle c\rangle\end{array}\right\}\circ(c) = \langle c\rangle,\langle ac\rangle,\langle bc\rangle,\langle cc\rangle$$

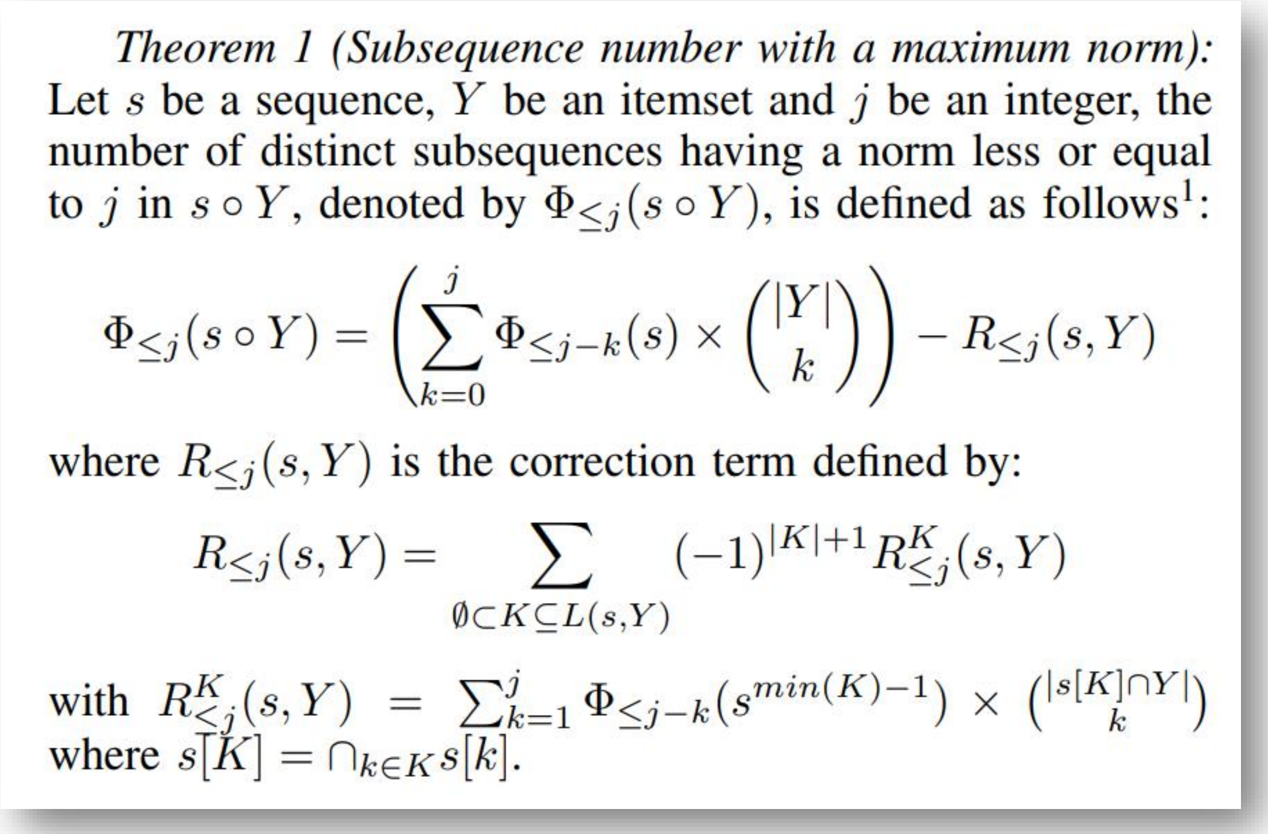
$$\begin{aligned}&\neq 7\times 1 + 4\times 1\\ &\neq 11\\ &= 11-(-1)^2\,\phi_{\le 1}(\langle(ab)\rangle\circ(c)) = 11-3=8\end{aligned}$$

[Diop et al., ICDM'2018]



DNA sequential data

Theorem for Distinct Subsequences Counting



Theorem 1 (Subsequence number with a maximum norm): Let s be a sequence, Y be an itemset and j be an integer, the number of distinct subsequences having a norm less or equal to j in $s \circ Y$, denoted by $\Phi_{\leq j}(s \circ Y)$, is defined as follows[1]:

$$\Phi_{\leq j}(s \circ Y) = \left(\sum_{k=0}^{j} \Phi_{\leq j-k}(s) \times \binom{|Y|}{k} \right) - R_{\leq j}(s, Y)$$

where $R_{\leq j}(s, Y)$ is the correction term defined by:

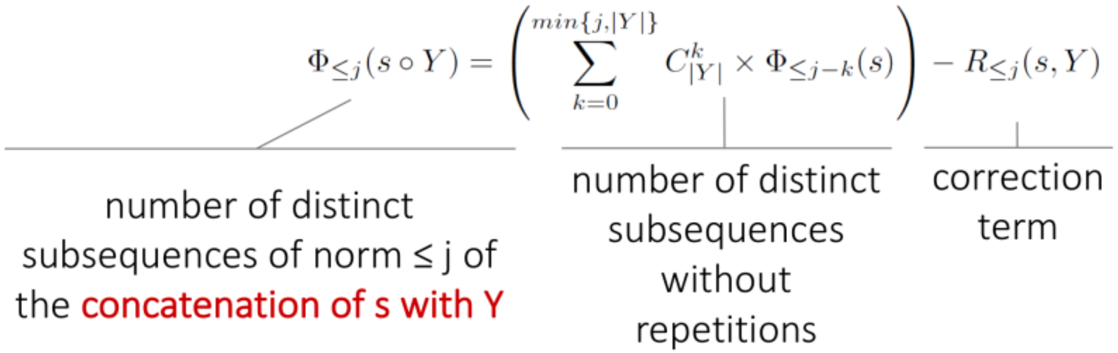
$$R_{\leq j}(s, Y) = \sum_{\emptyset \subset K \subseteq L(s,Y)} (-1)^{|K|+1} R^{K}_{\leq j}(s, Y)$$

with $R^{K}_{\leq j}(s, Y) = \sum_{k=1}^{j} \Phi_{\leq j-k}(s^{min(K)-1}) \times \binom{|s[K] \cap Y|}{k}$ where $s[K] = \cap_{k \in K} s[k]$.

Using the inclusion-exclusion principle

[Diop et al., ICDM'2018]

1. Number of subsequences with a norm ≤ j



$$\underbrace{\Phi_{\leq j}(s \circ Y)}_{\text{number of distinct subsequences of norm} \leq \text{j of the concatenation of s with Y}} = \left(\underbrace{\sum_{k=0}^{min\{j,|Y|\}} C^{k}_{|Y|} \times \Phi_{\leq j-k}(s)}_{\text{number of distinct subsequences without repetitions}} \right) - \underbrace{R_{\leq j}(s, Y)}_{\text{correction term}}$$

2. Drawing uniformly a subsequence

- **Algorithm for drawing a subsequence within s:**
  1. Draw randomly an occurrence o from s
  2. If o is not the first occurrence in s of its subsequence, repeat step 1.
  3. Return the subsequence stemming from o

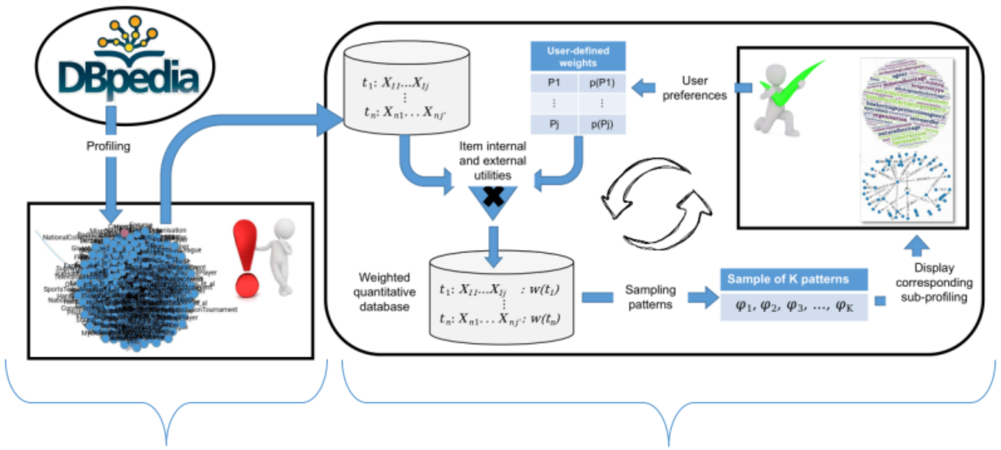
Example: 2 occurrences of **ac** in **(ab)c(ac):**

(ab)c(ac) ~~(ab)c(ac)~~

Pattern Counting in Weighted Itemsets (1/2)

High utility pattern mining & Knowledge graph profiling

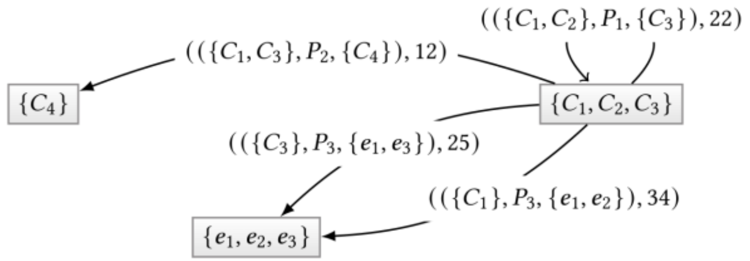
Knowledge graph profiling [Principe et al, VLDBJ'21]



[Principe et al., VLDBJ'21; Diop et al., JOCCH'23]

Our contribution

High utility pattern sampling [Diop, PAKDD'22]



$$X_1 = (\{C_1, C_2\}, P_1, \{C_3\}),\ X_2 = (\{C_1, C_3\}, P_2, \{C_4\}),$$
$$X_3 = (\{C_3\}, P_3, \{e_1, e_3\}),\ X_4 = (\{C_1\}, P_3, \{e_1, e_2\})$$

$p(X_1) = 2$, $p(X_2) = 1$, $p(X_3) = p(X_4) = 3$.

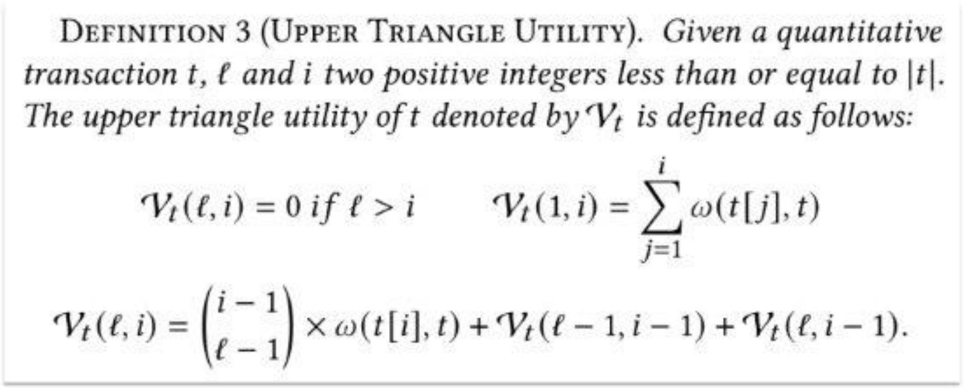
$$t_1 = \{X_1 : 22, X_2 : 12, X_3 : 25, X_4 : 34\} \quad t_2 = \{X_1 : 22\}$$
$$t_3 = \{X_2{:}12\} \text{ and } t_4 = \{X_3{:}25, X_4{:}34\} \quad \omega(t_1[2], t_1) = 12$$

t_1 : {	X_1	,	X_2		,	X_3			,	X_4				}
	44		12	56		75	206	131		34	233	364	165	
	0		44	0		56	56	0		131	262	131	0	

[Diop et Plantevit., EEEBigData'2024]

Pattern Counting in Weighted Itemsets (2/2)

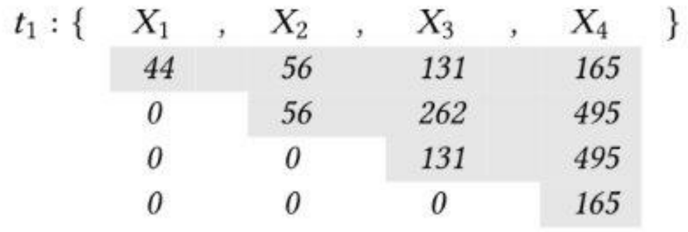
1. Weighting approach



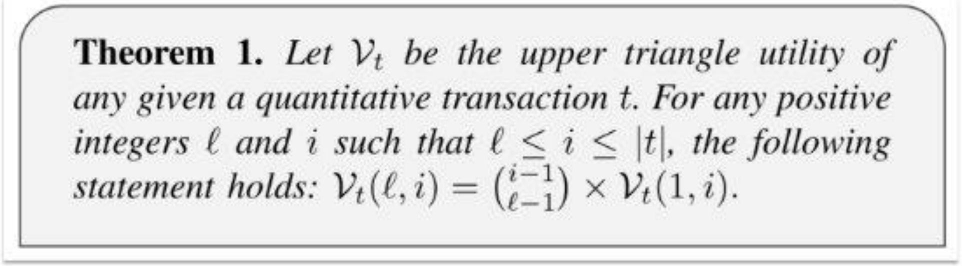
Definition 3 (Upper Triangle Utility). *Given a quantitative transaction t, ℓ and i two positive integers less than or equal to $|t|$. The upper triangle utility of t denoted by $\mathcal{V}_t$ is defined as follows:*

$$\mathcal{V}_t(\ell, i) = 0 \text{ if } \ell > i \qquad \mathcal{V}_t(1, i) = \sum_{j=1}^{i} \omega(t[j], t)$$

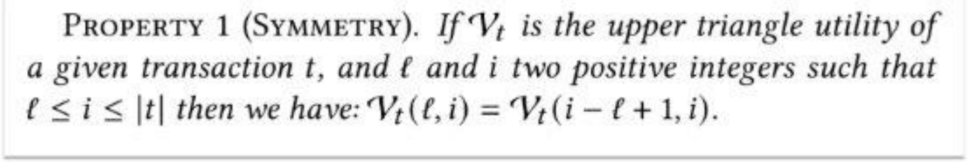
$$\mathcal{V}_t(\ell, i) = \binom{i-1}{\ell-1} \times \omega(t[i], t) + \mathcal{V}_t(\ell-1, i-1) + \mathcal{V}_t(\ell, i-1).$$



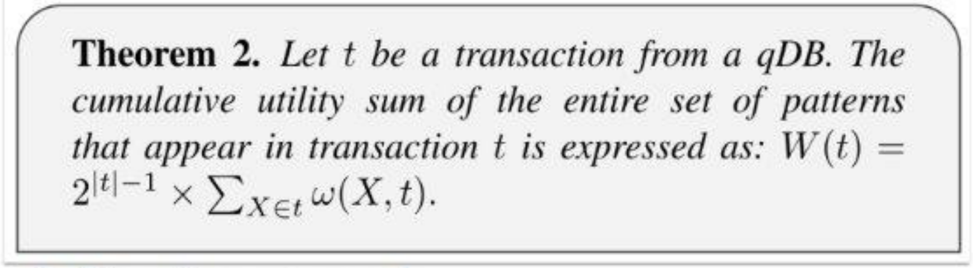
t_1 : {	X_1 ,	X_2 ,	X_3 ,	X_4	}
	44	56	131	165	
	0	56	262	495	
	0	0	131	495	
	0	0	0	165	



Theorem 1. *Let $\mathcal{V}_t$ be the upper triangle utility of any given a quantitative transaction t. For any positive integers ℓ and i such that $\ell \leq i \leq |t|$, the following statement holds: $\mathcal{V}_t(\ell, i) = \binom{i-1}{\ell-1} \times \mathcal{V}_t(1, i)$.*

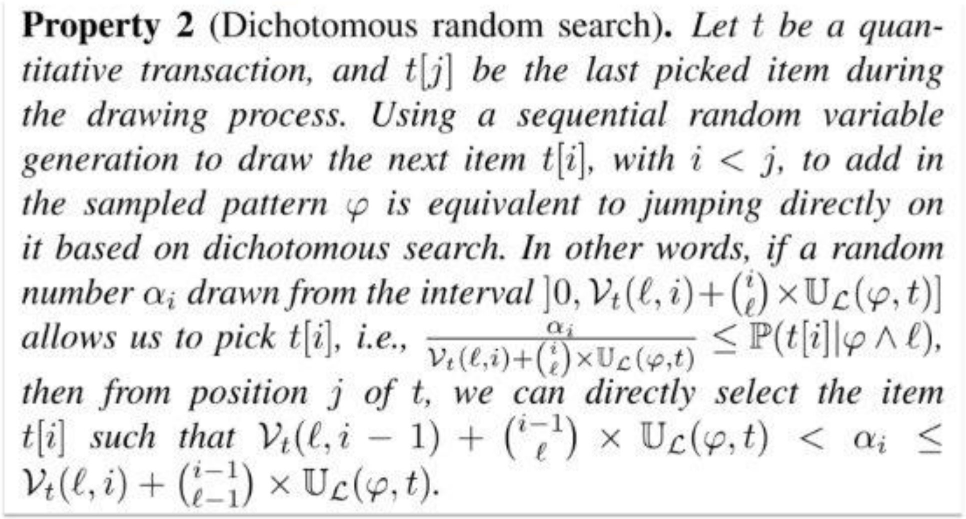


Property 1 (Symmetry). *If $\mathcal{V}_t$ is the upper triangle utility of a given transaction t, and ℓ and i two positive integers such that $\ell \leq i \leq |t|$ then we have: $\mathcal{V}_t(\ell, i) = \mathcal{V}_t(i - \ell + 1, i)$.*



Theorem 2. *Let t be a transaction from a qDB. The cumulative utility sum of the entire set of patterns that appear in transaction t is expressed as: $W(t) = 2^{|t|-1} \times \sum_{X \in t} \omega(X, t)$.*

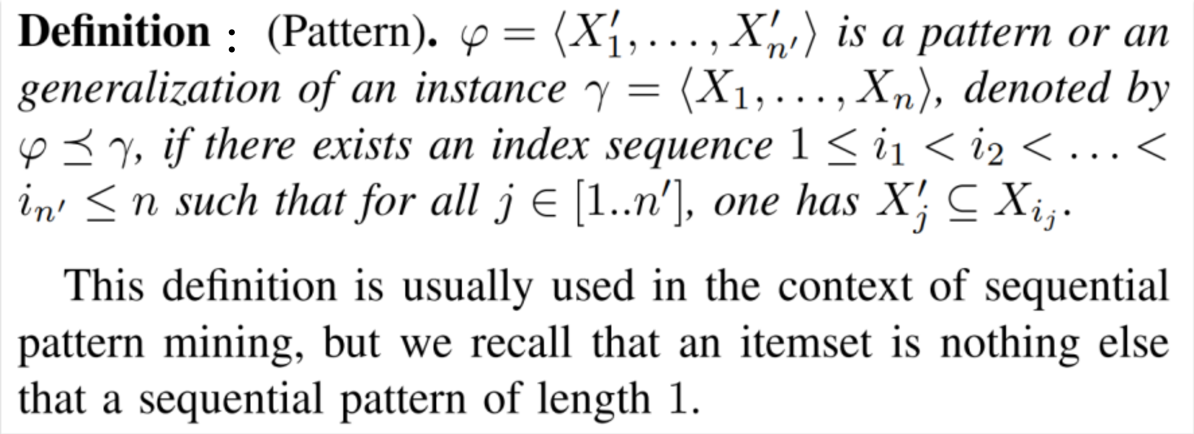
2. Drawing approach



Property 2 (Dichotomous random search)**.** *Let t be a quantitative transaction, and $t[j]$ be the last picked item during the drawing process. Using a sequential random variable generation to draw the next item $t[i]$, with $i < j$, to add in the sampled pattern φ is equivalent to jumping directly on it based on dichotomous search. In other words, if a random number α_i drawn from the interval $]0, \mathcal{V}_t(\ell, i) + \binom{i}{\ell} \times \mathbb{U}_{\mathcal{L}}(\varphi, t)]$ allows us to pick $t[i]$, i.e., $\frac{\alpha_i}{\mathcal{V}_t(\ell,i) + \binom{i}{\ell} \times \mathbb{U}_{\mathcal{L}}(\varphi,t)} \leq \mathbb{P}(t[i] | \varphi \wedge \ell)$, then from position j of t, we can directly select the item $t[i]$ such that $\mathcal{V}_t(\ell, i-1) + \binom{i-1}{\ell} \times \mathbb{U}_{\mathcal{L}}(\varphi, t) < \alpha_i \leq \mathcal{V}_t(\ell, i) + \binom{i-1}{\ell-1} \times \mathbb{U}_{\mathcal{L}}(\varphi, t)$.*

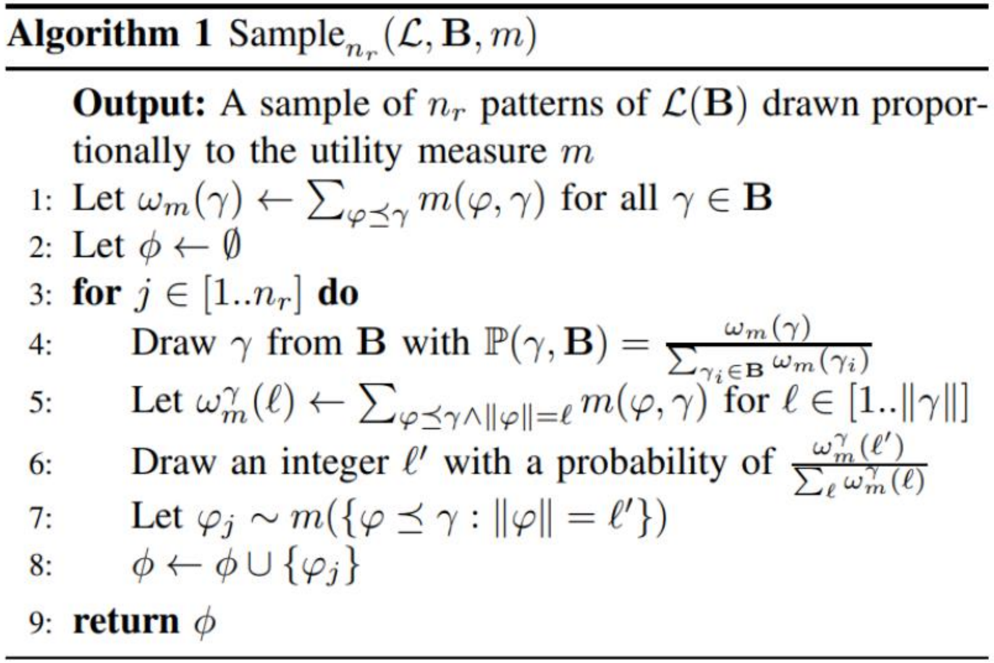
[Diop et Plantevit., EEEBigData'2024]

Generic Problem Reformulation



Definition : (Pattern). *$\varphi = \langle X'_1, \ldots, X'_{n'} \rangle$ is a pattern or an generalization of an instance $\gamma = \langle X_1, \ldots, X_n \rangle$, denoted by $\varphi \preceq \gamma$, if there exists an index sequence $1 \leq i_1 < i_2 < \ldots < i_{n'} \leq n$ such that for all $j \in [1..n']$, one has $X'_j \subseteq X_{i_j}$.*

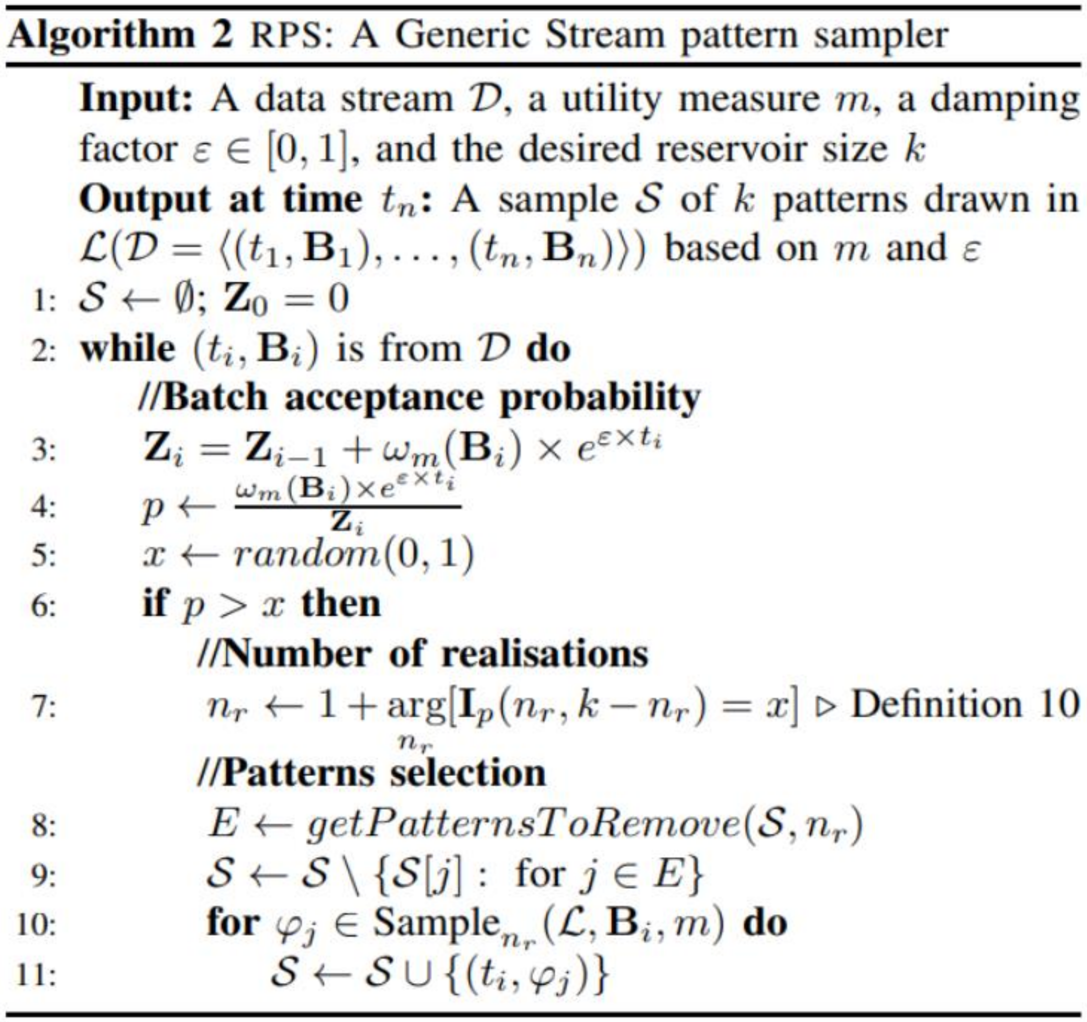
This definition is usually used in the context of sequential pattern mining, but we recall that an itemset is nothing else that a sequential pattern of length 1.



Algorithm 1 Sample$_{n_r}(\mathcal{L}, \mathbf{B}, m)$

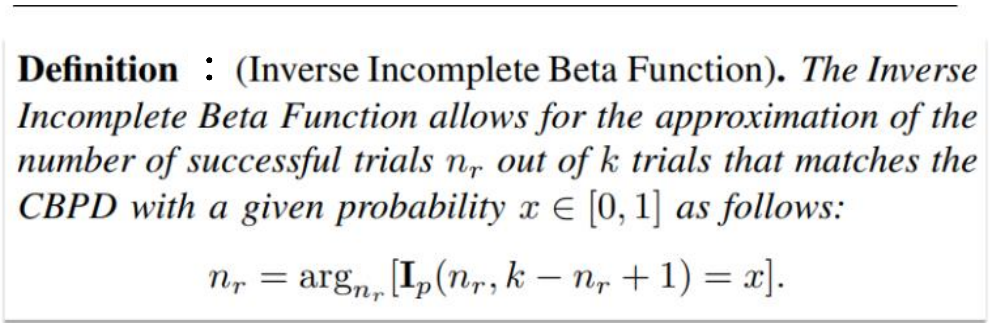
```
**Output:** A sample of $n_r$ patterns of $\mathcal{L}(\mathbf{B})$ drawn proportionally to the utility measure $m$
Let $\omega_m(\gamma) \leftarrow \sum_{\varphi \preceq \gamma} m(\varphi, \gamma)$ for all $\gamma \in \mathbf{B}$
Let $\phi \leftarrow \emptyset$
for $j \in [1..n_r]$ do
    Draw $\gamma$ from $\mathbf{B}$ with $\mathbb{P}(\gamma, \mathbf{B}) = \frac{\omega_m(\gamma)}{\sum_{\gamma_i \in \mathbf{B}} \omega_m(\gamma_i)}$
    Let $\omega_m^\gamma(\ell) \leftarrow \sum_{\varphi \preceq \gamma \wedge \|\varphi\| = \ell} m(\varphi, \gamma)$ for $\ell \in [1..\|\gamma\|]$
    Draw an integer $\ell'$ with a probability of $\frac{\omega_m^\gamma(\ell')}{\sum_\ell \omega_m^\gamma(\ell)}$
    Let $\varphi_j \sim m(\{\varphi \preceq \gamma : \|\varphi\| = \ell'\})$
    $\phi \leftarrow \phi \cup \{\varphi_j\}$
return $\phi$
```

RPS: Generic Reservoir Pattern Sampler



Algorithm 2 RPS: A Generic Stream pattern sampler

```
Input: A data stream $\mathcal{D}$, a utility measure $m$, a damping factor $\varepsilon \in [0, 1]$, and the desired reservoir size $k$
Output at time $t_n$: A sample $\mathcal{S}$ of $k$ patterns drawn in $\mathcal{L}(\mathcal{D} = \langle (t_1, \mathbf{B}_1), \ldots, (t_n, \mathbf{B}_n) \rangle)$ based on $m$ and $\varepsilon$
$\mathcal{S} \leftarrow \emptyset$; $\mathbf{Z}_0 = 0$
while $(t_i, \mathbf{B}_i)$ is from $\mathcal{D}$ do
    //Batch acceptance probability
    $\mathbf{Z}_i = \mathbf{Z}_{i-1} + \omega_m(\mathbf{B}_i) \times e^{\varepsilon \times t_i}$
    $p \leftarrow \frac{\omega_m(\mathbf{B}_i) \times e^{\varepsilon \times t_i}}{\mathbf{Z}_i}$
    $x \leftarrow random(0, 1)$
    if $p > x$ then
        //Number of realisations
        $n_r \leftarrow 1 + \arg_{n_r}[\mathbf{I}_p(n_r, k - n_r) = x]$   ▷ Definition 10
        //Patterns selection
        $E \leftarrow getPatternsToRemove(\mathcal{S}, n_r)$
        $\mathcal{S} \leftarrow \mathcal{S} \setminus \{\mathcal{S}[j] : \text{ for } j \in E\}$
        for $\varphi_j \in \text{Sample}_{n_r}(\mathcal{L}, \mathbf{B}_i, m)$ do
            $\mathcal{S} \leftarrow \mathcal{S} \cup \{(t_i, \varphi_j)\}$
```



Definition : (Inverse Incomplete Beta Function). *The Inverse Incomplete Beta Function allows for the approximation of the number of successful trials n_r out of k trials that matches the CBPD with a given probability $x \in [0, 1]$ as follows:*

$$n_r = \arg_{n_r}[\mathbf{I}_p(n_r, k - n_r + 1) = x].$$

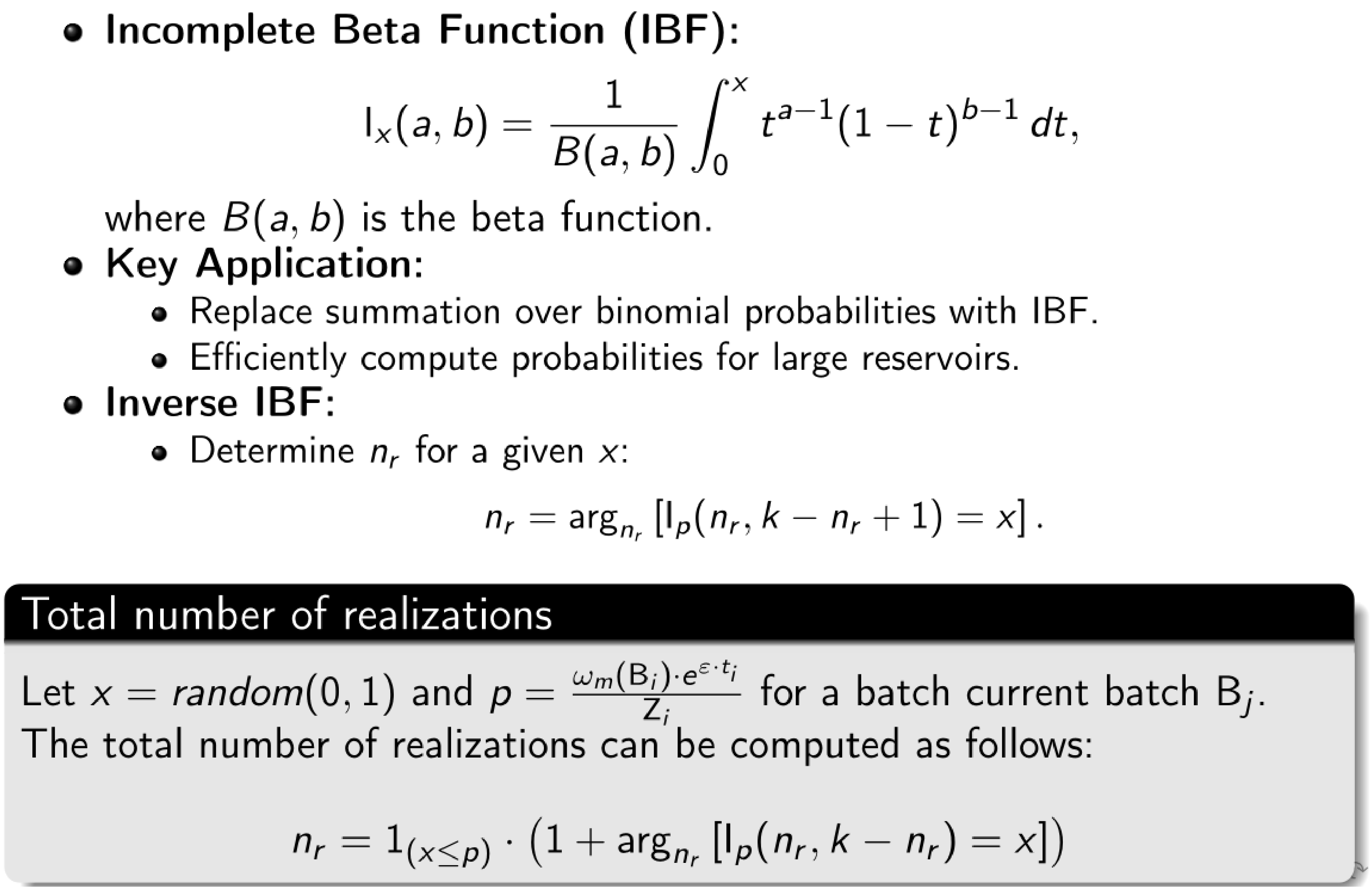
Computing Number of Realizations (1/2)

- **Context:** Selecting patterns for a reservoir of size k.
- **Goal:** Draw a set of n_r patterns proportionally to their weights from the current batch B_j, with $n_r \leq k$.
- **Method:**
  - Use n_r copies of batch B_j to simulate independent Bernoulli trials.
  - Simplify probability computation using **Cumulative Binomial Probability Distribution (CBPD):**

    $$\mathbb{P}(n_r, k) \equiv \sum_{j=n_r}^{k} \binom{k}{j} p^j (1-p)^{k-j}$$

    with $n_r \leq k$ the minimum number of times the event must occur, and p the probability of the event occurring in a single trial.
  - **Efficiency:** Leverage the **Incomplete Beta Function (IBF)** to reduce computational overhead.

Computing Number of Realizations (2/2)



- **Incomplete Beta Function (IBF):**

$$\mathrm{I}_x(a,b) = \frac{1}{B(a,b)} \int_0^x t^{a-1}(1-t)^{b-1}\,dt,$$

where $B(a,b)$ is the beta function.

- **Key Application:**
  - Replace summation over binomial probabilities with IBF.
  - Efficiently compute probabilities for large reservoirs.
- **Inverse IBF:**
  - Determine n_r for a given x:

$$n_r = \arg_{n_r}\left[\mathrm{I}_p(n_r, k - n_r + 1) = x\right].$$

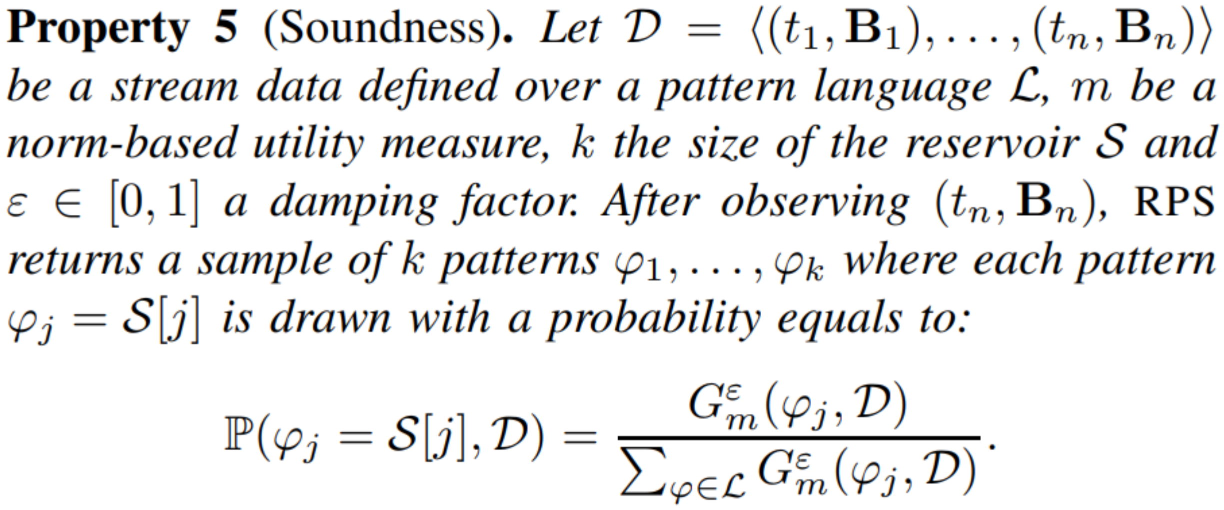
Total number of realizations

Let $x = random(0,1)$ and $p = \frac{\omega_m(\mathrm{B}_i)\cdot e^{\varepsilon \cdot t_i}}{\mathrm{Z}_i}$ for a batch current batch B_j. The total number of realizations can be computed as follows:

$$n_r = 1_{(x \leq p)} \cdot \left(1 + \arg_{n_r}\left[\mathrm{I}_p(n_r, k - n_r) = x\right]\right)$$

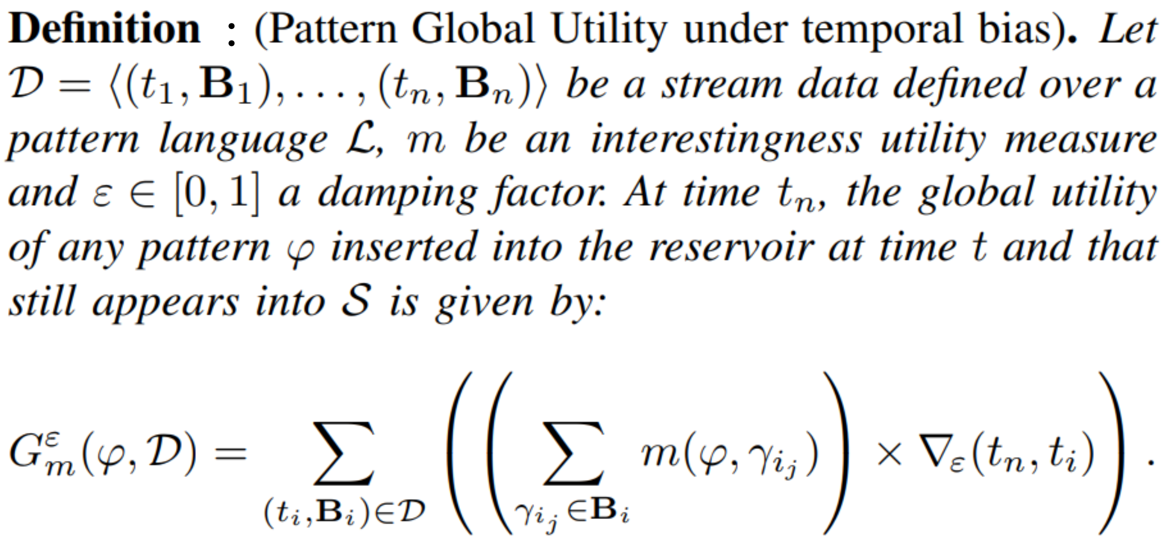
Soundness of RPS

For any Two-Step pattern sampler, regardless the data structure



Property 5 (Soundness)**.** *Let* $\mathcal{D} = \langle (t_1, \mathbf{B}_1), \ldots, (t_n, \mathbf{B}_n) \rangle$ *be a stream data defined over a pattern language* $\mathcal{L}$*,* m *be a norm-based utility measure,* k *the size of the reservoir* $\mathcal{S}$ *and* $\varepsilon \in [0, 1]$ *a damping factor. After observing* $(t_n, \mathbf{B}_n)$*,* RPS *returns a sample of* k *patterns* $\varphi_1, \ldots, \varphi_k$ *where each pattern* $\varphi_j = \mathcal{S}[j]$ *is drawn with a probability equals to:*

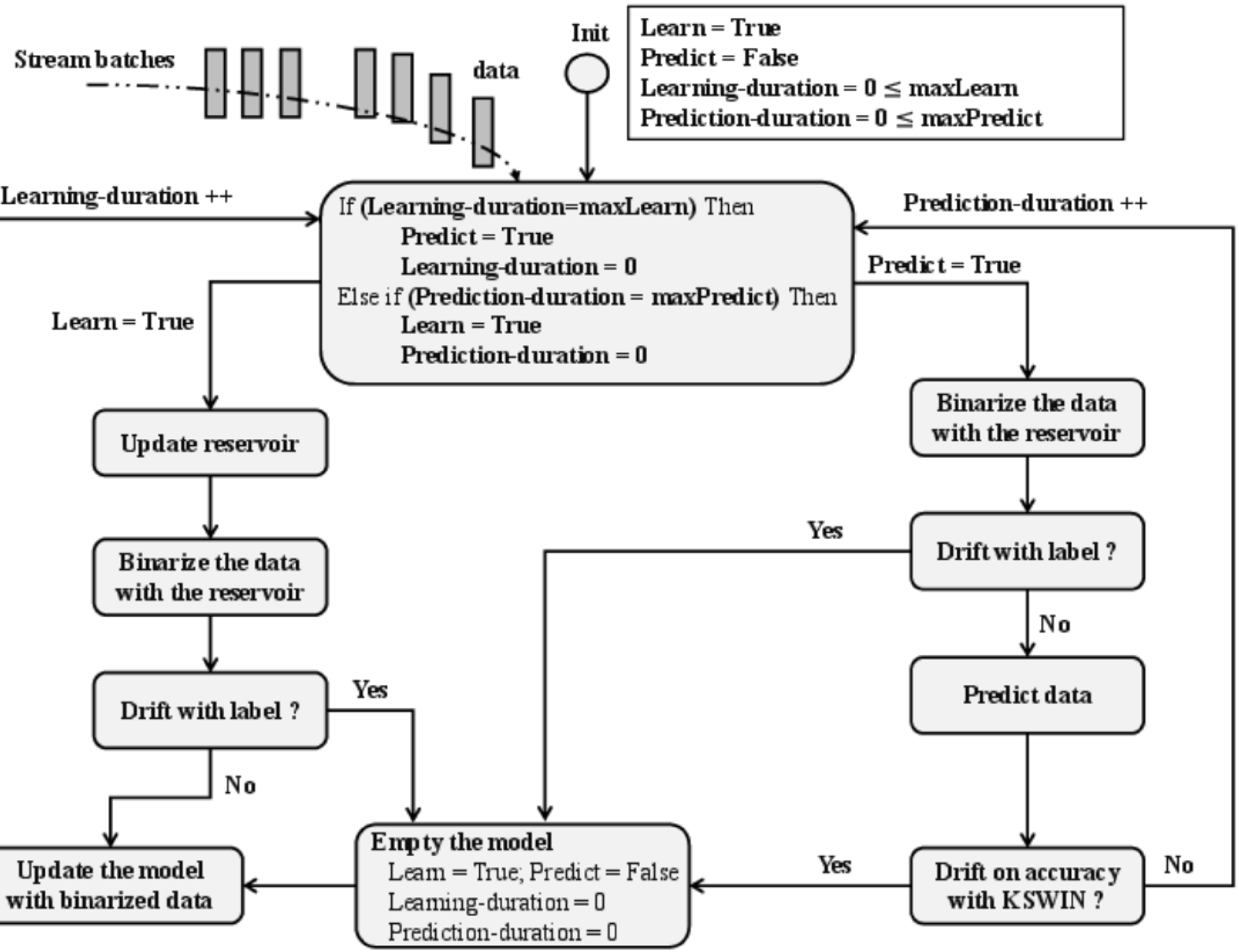
$$\mathbb{P}(\varphi_j = \mathcal{S}[j], \mathcal{D}) = \frac{G_m^{\varepsilon}(\varphi_j, \mathcal{D})}{\sum_{\varphi \in \mathcal{L}} G_m^{\varepsilon}(\varphi_j, \mathcal{D})}.$$



Definition : (Pattern Global Utility under temporal bias)**.** *Let* $\mathcal{D} = \langle (t_1, \mathbf{B}_1), \ldots, (t_n, \mathbf{B}_n) \rangle$ *be a stream data defined over a pattern language* $\mathcal{L}$*,* m *be an interestingness utility measure and* $\varepsilon \in [0, 1]$ *a damping factor. At time* t_n*, the global utility of any pattern* φ *inserted into the reservoir at time* t *and that still appears into* $\mathcal{S}$ *is given by:*

$$G_m^{\varepsilon}(\varphi, \mathcal{D}) = \sum_{(t_i, \mathbf{B}_i) \in \mathcal{D}} \left(\left(\sum_{\gamma_{i_j} \in \mathbf{B}_i} m(\varphi, \gamma_{i_j}) \right) \times \nabla_{\varepsilon}(t_n, t_i) \right).$$

Use Case on Classification of Stream Sequential Data



Stream batches

data

Init

Learn = True
Predict = False
Learning-duration = 0 ≤ maxLearn
Prediction-duration = 0 ≤ maxPredict

Learning-duration ++

If (Learning-duration=maxLearn) Then
    Predict = True
    Learning-duration = 0
Else if (Prediction-duration = maxPredict) Then
    Learn = True
    Prediction-duration = 0

Prediction-duration ++

Learn = True

Predict = True

Update reservoir

Binarize the data with the reservoir

Drift with label ?

Yes

No

Update the model with binarized data

Empty the model
Learn = True; Predict = False
Learning-duration = 0
Prediction-duration = 0

Binarize the data with the reservoir

Drift with label ?

Yes

No

Predict data

Drift on accuracy with KSWIN ?

Yes

No

Snapshots of RPS-Classifiers Evolution per batch

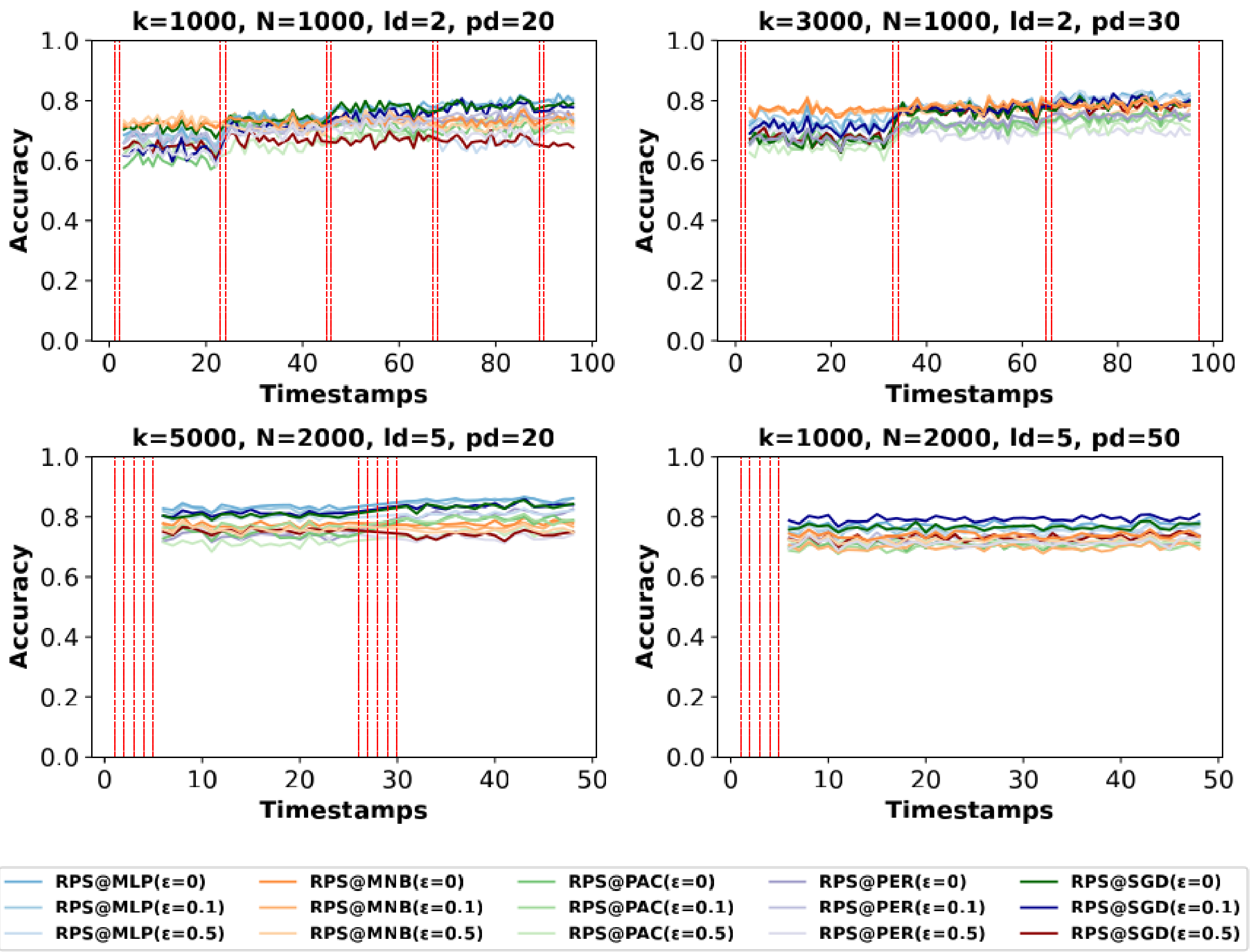


Figure: Evolution of the accuracy per batch with different parameters on `Books`. The learning timestamps are colored red. *k*: reservoir size, *N*: batch size, *ld*: learning duration, *pd*: predict duration

Online Classifier Accuracies

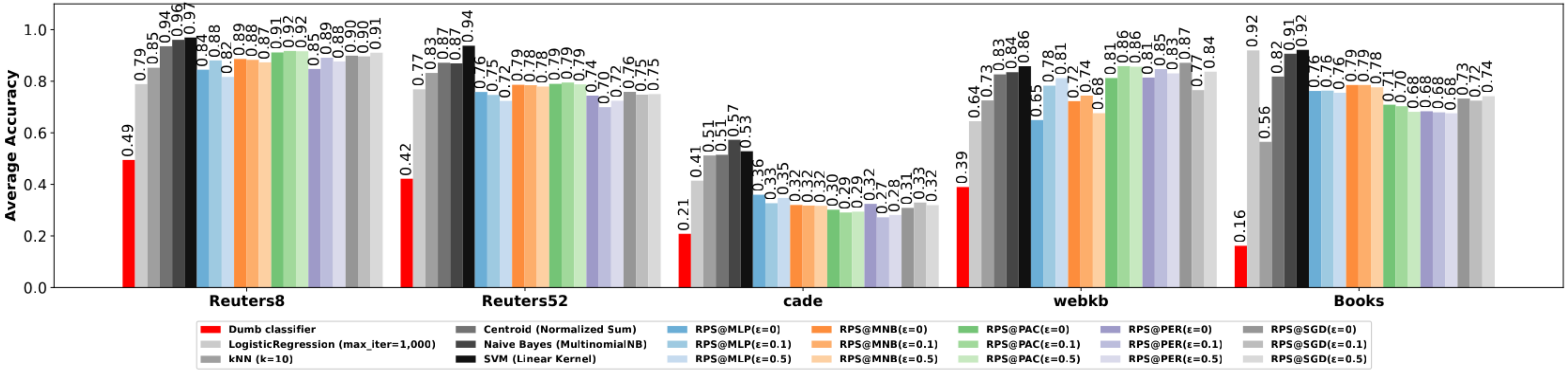


Figure: Comparison between RPS-based classifiers (with reservoir size k =10,000; batch size=1,000; learning duration=2 time-units, predict duration=52 time-units) vs cheater classifiers (with 50% train and 50% test)

Conclusion

Summary:

- Introduced a novel weighted reservoir sampling approach.
- Demonstrated scalability and effectiveness in complex streams.

Impact:

- Enhances scalability for large-scale systems.
- Reduces computational complexity for real-time tasks.

Future Work:

- Extension to multi-stream environments.
- For xAI: Capturing patterns from latent space during model training