Interval Selection in Data Streams: Weighted Intervals and the Insertion-deletion Setting

Jacques Dark, Adithya Diddapur[1], Christian Konrad[1]

[1]University of Bristol

The Streaming Setting

The Streaming Setting

- We have an input $\mathcal{I} = (I_1, \ldots, I_m)$.

The Streaming Setting

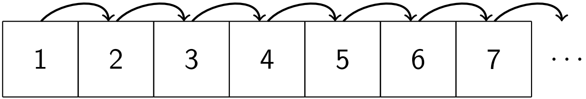
- We have an input $\mathcal{I} = (I_1, \ldots, I_m)$.
- Unable to store the entire input.

The Streaming Setting

- We have an input $\mathcal{I} = (I_1, \ldots, I_m)$.
- Unable to store the entire input.

Goal

Given the input elements one at a time, compute an (approximate) solution using as **little space** as possible.



The Streaming Setting (Contd.)

The Streaming Setting (Contd.)

- **Insertion-only**: stream elements arrive, and never disappear.
- **Insertion-deletion:** stream elements may either be elements arriving, or previous elements begin deleted.

The Streaming Setting (Contd.)

- **Insertion-only**: stream elements arrive, and never disappear.
- **Insertion-deletion:** stream elements may either be elements arriving, or previous elements begin deleted.

The Streaming Setting (Contd.)

- **Insertion-only**: stream elements arrive, and never disappear.
- **Insertion-deletion:** stream elements may either be elements arriving, or previous elements begin deleted.

- Length of the stream is by n.
- Optimum solution is denoted OPT.
- OPT^* denotes the largest solution over all prefixes.

The Streaming Setting (Contd.)

- **Insertion-only**: stream elements arrive, and never disappear.
- **Insertion-deletion:** stream elements may either be elements arriving, or previous elements begin deleted.

- Length of the stream is by n.
- Optimum solution is denoted OPT.
- OPT^* denotes the largest solution over all prefixes.

Any problem can be solved in space $O(n)$ and requires $\Omega(|OPT|)$.

The Streaming Setting (Contd.)

- **Insertion-only**: stream elements arrive, and never disappear.
- **Insertion-deletion:** stream elements may either be elements arriving, or previous elements begin deleted.

- Length of the stream is by n.
- Optimum solution is denoted OPT.
- OPT^* denotes the largest solution over all prefixes.

Any problem can be solved in space $O(n)$ and requires $\Omega(|OPT|)$.

- **Key Question:** What can we achieve across this regime?

Interval Selection Problem

Interval Selection Problem

- An interval is a set $I = [a, b] \subset \mathbb{R}$.
- I_1, I_2 are independent if $I_1 \cap I_2 = \emptyset$.

Interval Selection Problem

- An interval is a set $I = [a, b] \subset \mathbb{R}$.
- I_1, I_2 are independent if $I_1 \cap I_2 = \emptyset$.

Problem (Interval Selection)

Given a set of intervals, presented as a stream, return a set of pairwise independent intervals which is (approximately) as large as possible.

Interval Selection Problem

- An interval is a set $I = [a, b] \subset \mathbb{R}$.
- I_1, I_2 are independent if $I_1 \cap I_2 = \emptyset$.

Problem (Interval Selection)

Given a set of intervals, presented as a stream, return a set of pairwise independent intervals which is (approximately) as large as possible.

Interval Selection Problem

- An interval is a set $I = [a, b] \subset \mathbb{R}$.
- I_1, I_2 are independent if $I_1 \cap I_2 = \emptyset$.

Problem (Interval Selection)

Given a set of intervals, presented as a stream, return a set of pairwise independent intervals which is (approximately) as large as possible.

Interval Selection Problem

- An interval is a set $I = [a, b] \subset \mathbb{R}$.
- I_1, I_2 are independent if $I_1 \cap I_2 = \emptyset$.

Problem (Interval Selection)

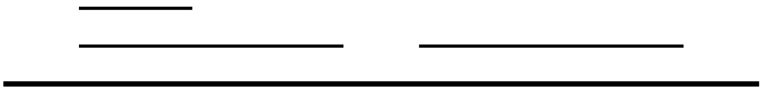
Given a set of intervals, presented as a stream, return a set of pairwise independent intervals which is (approximately) as large as possible.

Interval Selection Problem

- An interval is a set $I = [a, b] \subset \mathbb{R}$.
- I_1, I_2 are independent if $I_1 \cap I_2 = \emptyset$.

Problem (Interval Selection)

Given a set of intervals, presented as a stream, return a set of pairwise independent intervals which is (approximately) as large as possible.

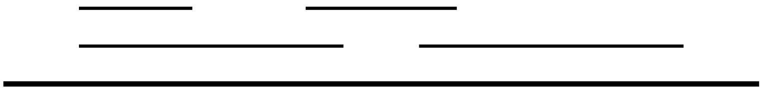


Interval Selection Problem

- An interval is a set $I = [a, b] \subset \mathbb{R}$.
- I_1, I_2 are independent if $I_1 \cap I_2 = \emptyset$.

Problem (Interval Selection)

Given a set of intervals, presented as a stream, return a set of pairwise independent intervals which is (approximately) as large as possible.

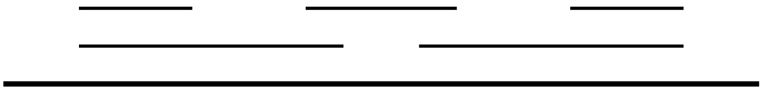


Interval Selection Problem

- An interval is a set $I = [a, b] \subset \mathbb{R}$.
- I_1, I_2 are independent if $I_1 \cap I_2 = \emptyset$.

Problem (Interval Selection)

Given a set of intervals, presented as a stream, return a set of pairwise independent intervals which is (approximately) as large as possible.

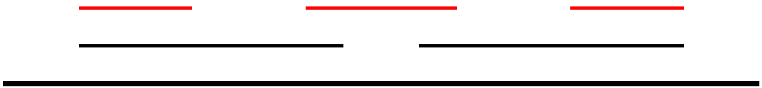


Interval Selection Problem

- An interval is a set $I = [a, b] \subset \mathbb{R}$.
- I_1, I_2 are independent if $I_1 \cap I_2 = \emptyset$.

Problem (Interval Selection)

Given a set of intervals, presented as a stream, return a set of pairwise independent intervals which is (approximately) as large as possible.



State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\|OPT\|)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\|OPT^*\|)$	LB - $\Omega(n)$
Unweighted	Unit-Length				
	Variable-Length				
Weighted	Unit-Length				
	Variable-Length				

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\lvert OPT\rvert)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\lvert OPT^*\rvert)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2-\epsilon$ [EHR12]		$2-\epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2-\epsilon$ [EHR12]		
Weighted	Unit-Length		$3/2-\epsilon$ [EHR12]		$2-\epsilon$ [BCW20]
	Variable-Length				

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\|OPT\|)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\|OPT^*\|)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2-\epsilon$ [EHR12]		$2-\epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2-\epsilon$ [EHR12]		
Weighted	Unit-Length	$\mathbf{3/2}+\epsilon$	$3/2-\epsilon$ [EHR12]		$2-\epsilon$ [BCW20]
	Variable-Length				

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\|OPT\|)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\|OPT^*\|)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2-\epsilon$ [EHR12]		$2-\epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2-\epsilon$ [EHR12]		
Weighted	Unit-Length	$\mathbf{3/2}+\epsilon$	$3/2-\epsilon$ [EHR12]		$2-\epsilon$ [BCW20]
	Variable-Length	$*$	$\mathbf{\Theta(1)}$		

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\|OPT\|)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\|OPT^*\|)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2-\epsilon$ [EHR12]	$\mathbf{2}$	$2-\epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2-\epsilon$ [EHR12]		
Weighted	Unit-Length	$\mathbf{3/2}+\epsilon$	$3/2-\epsilon$ [EHR12]	$\mathbf{2}+\epsilon$	$2-\epsilon$ [BCW20]
	Variable-Length	$*$	$\mathbf{\Theta(1)}$		

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\lvert OPT\rvert)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\lvert OPT^*\rvert)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2-\epsilon$ [EHR12]	$\mathbf{2}$	$2-\epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2-\epsilon$ [EHR12]	$*$	$\boldsymbol{\Theta(1)}$
Weighted	Unit-Length	$\mathbf{3/2}+\epsilon$	$3/2-\epsilon$ [EHR12]	$\mathbf{2}+\epsilon$	$2-\epsilon$ [BCW20]
	Variable-Length	$*$	$\boldsymbol{\Theta(1)}$	$*$	$\boldsymbol{\Theta(1)}$

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\lvert OPT \rvert)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\lvert OPT^* \rvert)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2 - \epsilon$ [EHR12]	$\mathbf{2}$	$2 - \epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2 - \epsilon$ [EHR12]	$*$	$\boldsymbol{\Theta(1)}$
Weighted	Unit-Length	$\mathbf{3/2} + \epsilon$	$3/2 - \epsilon$ [EHR12]	$\mathbf{2} + \epsilon$	$2 - \epsilon$ [BCW20]
	Variable-Length	$*$	$\boldsymbol{\Theta(1)}$	$*$	$\boldsymbol{\Theta(1)}$

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\lvert OPT\rvert)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\lvert OPT^*\rvert)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2-\epsilon$ [EHR12]	$\mathbf{2}$	$2-\epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2-\epsilon$ [EHR12]	$*$	$\boldsymbol{\Theta(1)}$
Weighted	Unit-Length	$\mathbf{3/2}+\epsilon$	$3/2-\epsilon$ [EHR12]	$\mathbf{2}+\epsilon$	$2-\epsilon$ [BCW20]
	Variable-Length	$*$	$\boldsymbol{\Theta(1)}$	$*$	$\boldsymbol{\Theta(1)}$

Weighted Unit-Length Intervals

Weighted Unit-Length Intervals

- A window of length γ is a set of the form $[a, a + \gamma)$.

Weighted Unit-Length Intervals

- A window of length γ is a set of the form $[a, a+\gamma)$.

Lemma

If there is a unit-length algorithm for a restricted window of length γ, then there is an algorithm for unrestricted domains whose approximation factor increases by $\gamma/(\gamma-1)$ and space by $O(|OPT|)$.

Weighted Unit-Length Intervals

- A window of length γ is a set of the form $[a, a + \gamma)$.

Lemma

If there is a unit-length algorithm for a restricted window of length γ, then there is an algorithm for unrestricted domains whose approximation factor increases by $\gamma/(\gamma - 1)$ and space by $O(|OPT|)$.

Weighted Unit-Length Intervals

- A window of length γ is a set of the form $[a, a+\gamma)$.

Lemma

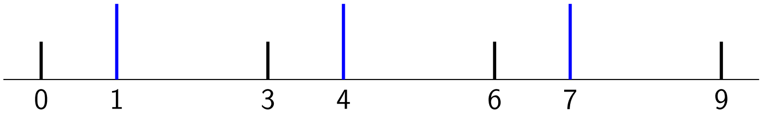
If there is a unit-length algorithm for a restricted window of length γ, then there is an algorithm for unrestricted domains whose approximation factor increases by $\gamma/(\gamma-1)$ and space by $O(|OPT|)$.

Weighted Unit-Length Intervals

- A window of length γ is a set of the form $[a, a+\gamma)$.

Lemma

If there is a unit-length algorithm for a restricted window of length γ, then there is an algorithm for unrestricted domains whose approximation factor increases by $\gamma/(\gamma-1)$ and space by $O(|OPT|)$.

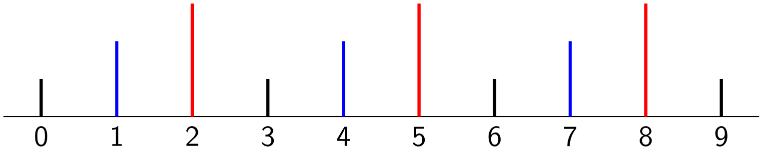


Weighted Unit-Length Intervals

- A window of length γ is a set of the form $[a, a+\gamma)$.

Lemma

If there is a unit-length algorithm for a restricted window of length γ, then there is an algorithm for unrestricted domains whose approximation factor increases by $\gamma/(\gamma-1)$ and space by $O(|OPT|)$.



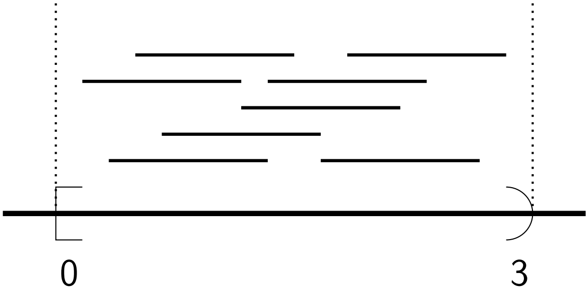
Weighted Unit-Length Intervals

Weighted Unit-Length Intervals

- Consider a single window of length 3.

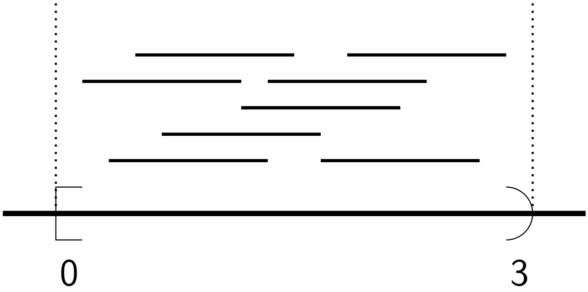
Weighted Unit-Length Intervals

- Consider a single window of length 3.



Weighted Unit-Length Intervals

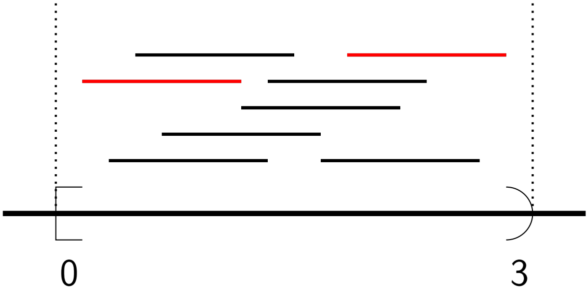
- Consider a single window of length 3.



- **Unweighted Setting:** Find the left and right most intervals.

Weighted Unit-Length Intervals

- Consider a single window of length 3.



- **Unweighted Setting:** Find the left and right most intervals.

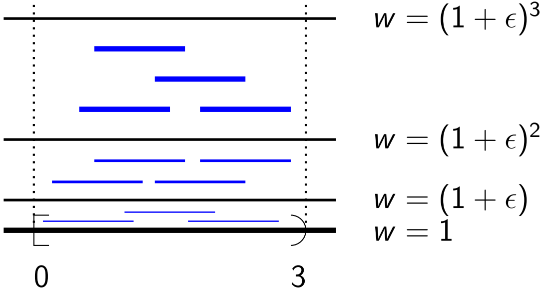
Weighted Unit-Length Intervals

Weighted Unit-Length Intervals

- **Weighted Setting:** Sub-divide the window into geometric weight classes.

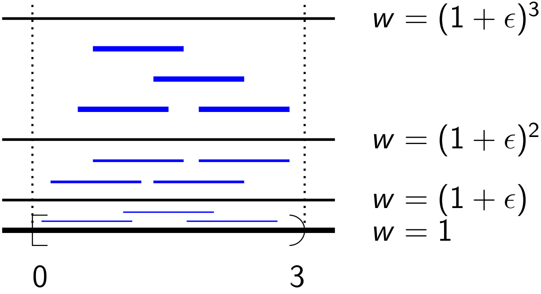
Weighted Unit-Length Intervals

- **Weighted Setting:** Sub-divide the window into geometric weight classes.



Weighted Unit-Length Intervals

- **Weighted Setting:** Sub-divide the window into geometric weight classes.



- This gives a $(1+\epsilon)$-approximation within the heaviest $\log(1/\epsilon)/\epsilon$ weight classes.

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\lvert OPT \rvert)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\lvert OPT^* \rvert)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2 - \epsilon$ [EHR12]	$\mathbf{2}$	$2 - \epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2 - \epsilon$ [EHR12]	$*$	$\mathbf{\Theta(1)}$
Weighted	Unit-Length	$\mathbf{3/2} + \epsilon$	$3/2 - \epsilon$ [EHR12]	$\mathbf{2} + \epsilon$	$2 - \epsilon$ [BCW20]
	Variable-Length	$*$	$\mathbf{\Theta(1)}$	$*$	$\mathbf{\Theta(1)}$

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\|OPT\|)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\|OPT^*\|)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2-\epsilon$ [EHR12]	$\mathbf{2}$	$2-\epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2-\epsilon$ [EHR12]	$*$	$\mathbf{\Theta(1)}$
Weighted	Unit-Length	$\mathbf{3/2}+\epsilon$	$3/2-\epsilon$ [EHR12]	$\mathbf{2}+\epsilon$	$2-\epsilon$ [BCW20]
	Variable-Length	$*$	$\mathbf{\Theta(1)}$	$*$	$\mathbf{\Theta(1)}$

Unweighted Variable-Length Intervals

Unweighted Variable-Length Intervals

- LB via communication complexity.

Unweighted Variable-Length Intervals

- LB via communication complexity.

Chained Index Problem - $\mathrm{CHAIN_p(k)}$ [CDK19]

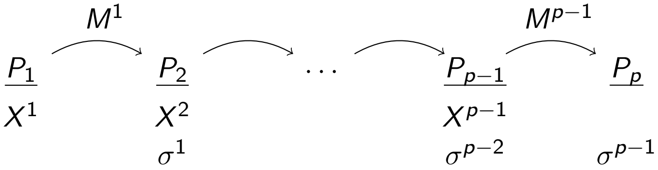
$$\begin{array}{ccccccccc} \underline{P_1} & \xrightarrow{M^1} & \underline{P_2} & \longrightarrow & \cdots & \longrightarrow & \underline{P_{p-1}} & \xrightarrow{M^{p-1}} & \underline{P_p} \\ X^1 & & X^2 & & & & X^{p-1} & & \\ & & \sigma^1 & & & & \sigma^{p-2} & & \sigma^{p-1} \end{array}$$

- $X^i \in \{0,1\}^k$, $\sigma^i \in [k]$.
- **Promise:** $X^i[\sigma^i] = z \in \{0,1\}$ for each i.
- **Goal:** Recover z using small messages.

Unweighted Variable-Length Intervals

- LB via communication complexity.

Chained Index Problem - $\mathrm{CHAIN_p(k)}$ [CDK19]



- $X^i \in \{0,1\}^k$, $\sigma^i \in [k]$.
- **Promise:** $X^i[\sigma^i] = z \in \{0,1\}$ for each i.
- **Goal:** Recover z using small messages.

Theorem

[FNSZ20] Any (randomised) protocol for $CHAIN_p(k)$ with success probability at least $2/3$ must send a message of size $\Omega(k/p^2)$.

Unweighted Variable-Length Intervals

Unweighted Variable-Length Intervals

- A protocol for Interval Selection can be used to solve $\mathsf{CHAIN}_p(k)$.

Unweighted Variable-Length Intervals

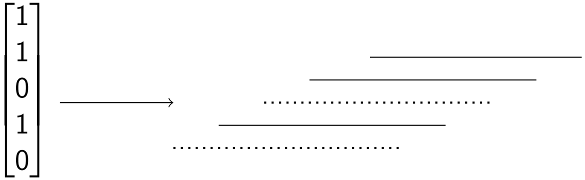
- A protocol for Interval Selection can be used to solve $\mathsf{CHAIN_p(k)}$.

1. Encode $X^i \in \{0,1\}^k$ into a weighted clique gadget with weight $w = 2^{p-i}$.

Unweighted Variable-Length Intervals

- A protocol for Interval Selection can be used to solve $\mathrm{CHAIN_p(k)}$.

1. Encode $X^i \in \{0,1\}^k$ into a weighted clique gadget with weight $w = 2^{p-i}$.



Unweighted Variable-Length Intervals

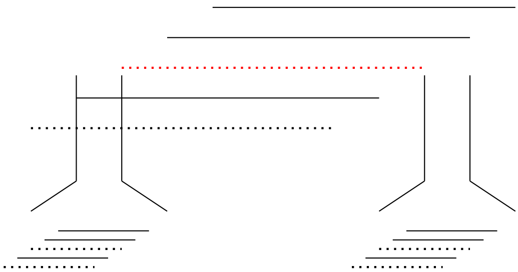
- A algorithm for Interval Selection can be used to solve $\mathsf{CHAIN}_p(k)$.

2. Use the special indices to nest clique gadgets.

Unweighted Variable-Length Intervals

- A algorithm for Interval Selection can be used to solve $\mathsf{CHAIN}_p(k)$.

2. Use the special indices to nest clique gadgets.



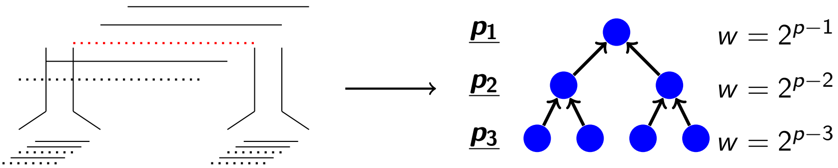
Unweighted Variable-Length Intervals

Unweighted Variable-Length Intervals

- Model the independent sets via a weighted balanced binary tree.

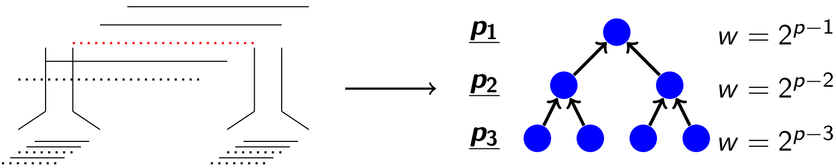
Unweighted Variable-Length Intervals

- Model the independent sets via a weighted balanced binary tree.



Unweighted Variable-Length Intervals

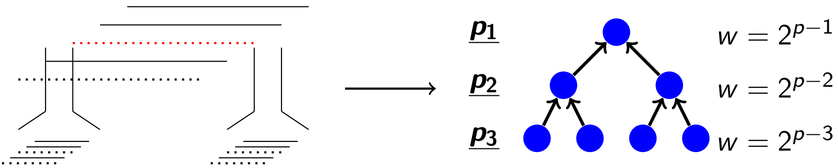
- Model the independent sets via a weighted balanced binary tree.



- If $Z = 1$, then $w(OPT) = p \cdot 2^p$.
- If $Z = 0$, then $w(OPT) < 2^p$.

Unweighted Variable-Length Intervals

- Model the independent sets via a weighted balanced binary tree.



- If $Z = 1$, then $w(OPT) = p \cdot 2^p$.
- If $Z = 0$, then $w(OPT) < 2^p$.

Theorem

A $(p - \epsilon)$-approximate alg. solves the $CHAIN_p(k)$ instance, and so requires space $\Omega(k/p) = \Omega(n)$ for constant p.

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\lvert OPT \rvert)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\lvert OPT^* \rvert)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2 - \epsilon$ [EHR12]	$\mathbf{2}$	$2 - \epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2 - \epsilon$ [EHR12]	$*$	$\mathbf{\Theta(1)}$
Weighted	Unit-Length	$\mathbf{3/2} + \epsilon$	$3/2 - \epsilon$ [EHR12]	$\mathbf{2} + \epsilon$	$2 - \epsilon$ [BCW20]
	Variable-Length	$*$	$\mathbf{\Theta(1)}$	$*$	$\mathbf{\Theta(1)}$

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\|OPT\|)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\|OPT^*\|)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2-\epsilon$ [EHR12]	$\mathbf{2}$	$2-\epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2-\epsilon$ [EHR12]	$*$	$\mathbf{\Theta(1)}$
Weighted	Unit-Length	$\mathbf{3/2}+\epsilon$	$3/2-\epsilon$ [EHR12]	$\mathbf{2}+\epsilon$	$2-\epsilon$ [BCW20]
	Variable-Length	$*$	$\mathbf{\Theta(1)}$	$*$	$\mathbf{\Theta(1)}$

Going from $O(|OPT|)$ to $O(|OPT^*|)$

Going from $O(|OPT|)$ to $O(|OPT^*|)$

- $|OPT|$ is the cardinality of the optimum solution at the end of the stream.

Going from $O(|OPT|)$ to $O(|OPT^*|)$

- $|OPT|$ is the cardinality of the optimum solution at the end of the stream.
- $|OPT^*|$ is the cardinality of the optimum solution over all prefixes of the stream.

Going from $O(|OPT|)$ to $O(|OPT^*|)$

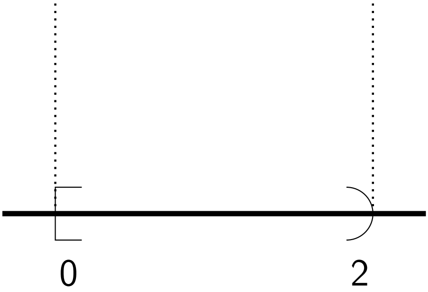
- $|OPT|$ is the cardinality of the optimum solution at the end of the stream.
- $|OPT^*|$ is the cardinality of the optimum solution over all prefixes of the stream.

Observation

In an insertion-only stream $|OPT^*| \leq |OPT|$.
This is not guaranteed in an insertion-deletion stream.

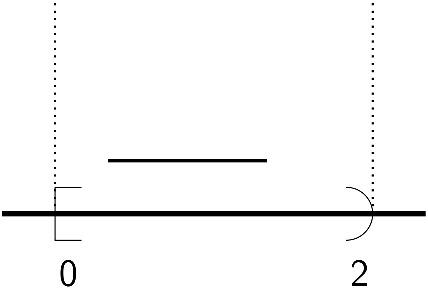
Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.



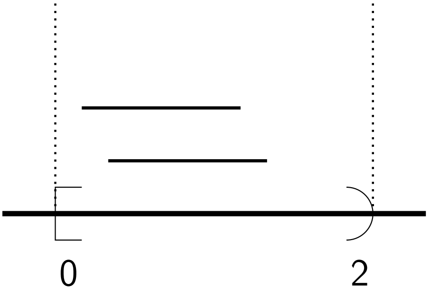
Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.



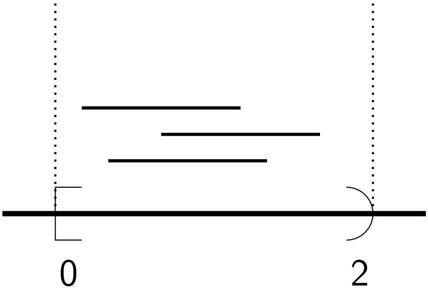
Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.



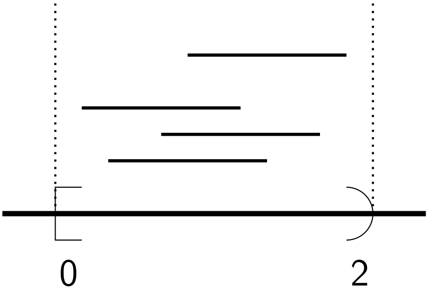
Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.



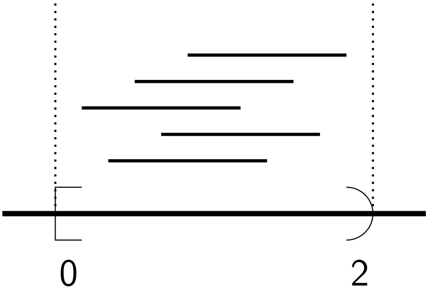
Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.



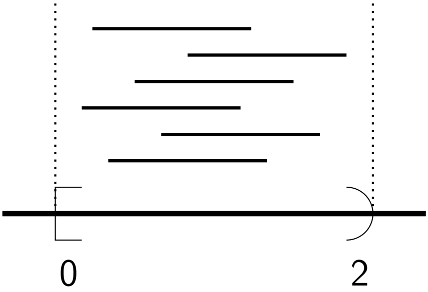
Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.



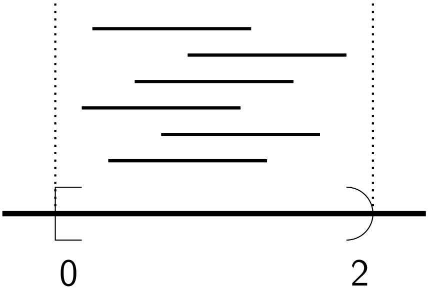
Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.



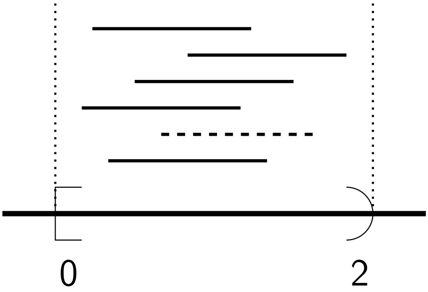
Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.



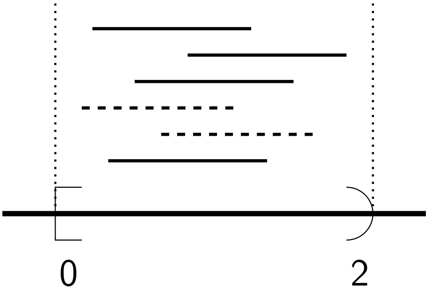
Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.



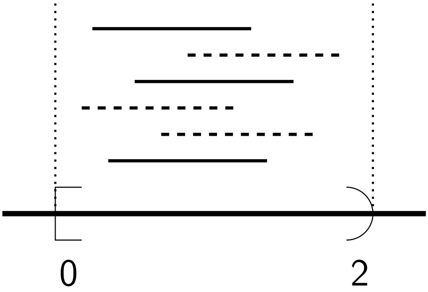
Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.



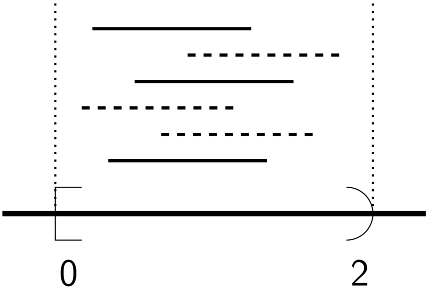
Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.



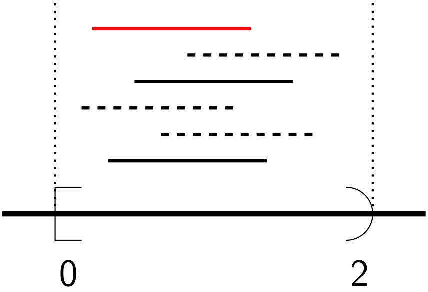
Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.
- **Problem:** Recover a single surviving interval.



Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.
- **Problem:** Recover a single surviving interval.

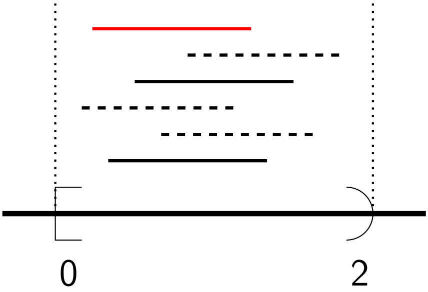


Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.
- **Problem:** Recover a single surviving interval.

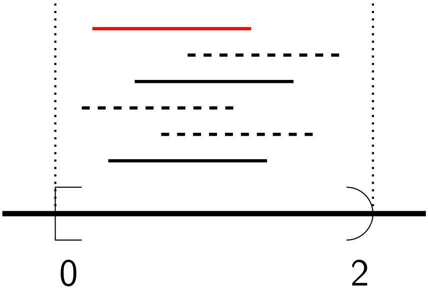
Problem

ℓ_0-Sampling Sample a uniform random surviving element from an insertion-deletion stream.



Insertion-Deletion: Unit-Length Intervals

- Now consider a window of length 2.
- **Problem:** Recover a single surviving interval.



Problem

ℓ_0-Sampling Sample a uniform random surviving element from an insertion-deletion stream.

Theorem

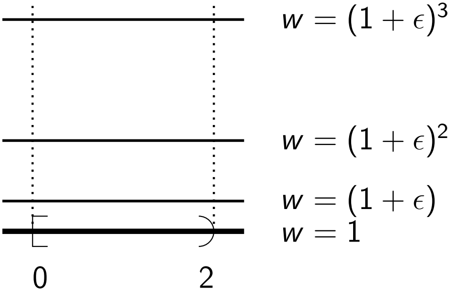
[JST11] There is an algorithm for ℓ_0-Sampling which succeeds w.h.p. in space $O(\log^3 n)$.

Insertion-Deletion: Unit-Length Intervals

- **Weighted Setting:**
  Sub-divide the window into geometric weight classes.

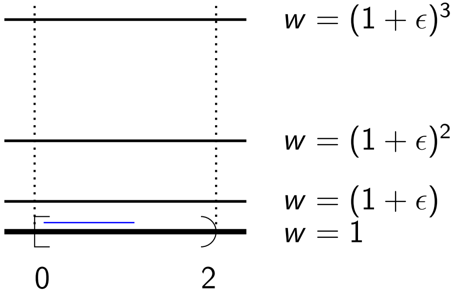
Insertion-Deletion: Unit-Length Intervals

- **Weighted Setting:**
  Sub-divide the window into geometric weight classes.



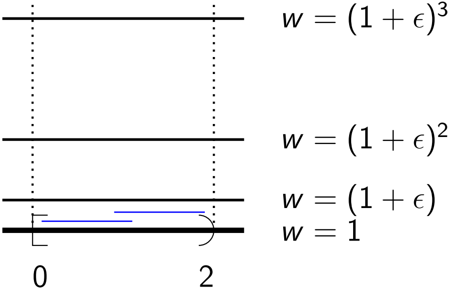
Insertion-Deletion: Unit-Length Intervals

- **Weighted Setting:**
  Sub-divide the window into geometric weight classes.



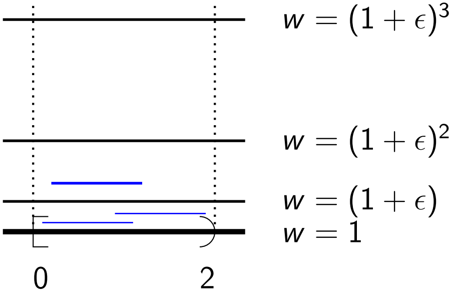
Insertion-Deletion: Unit-Length Intervals

- **Weighted Setting:**
  Sub-divide the window into geometric weight classes.



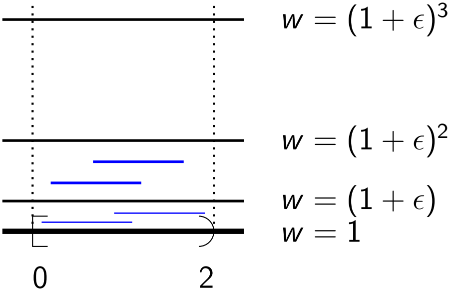
Insertion-Deletion: Unit-Length Intervals

- **Weighted Setting:** Sub-divide the window into geometric weight classes.



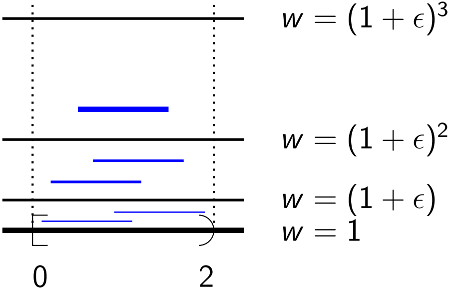
Insertion-Deletion: Unit-Length Intervals

- **Weighted Setting:** Sub-divide the window into geometric weight classes.



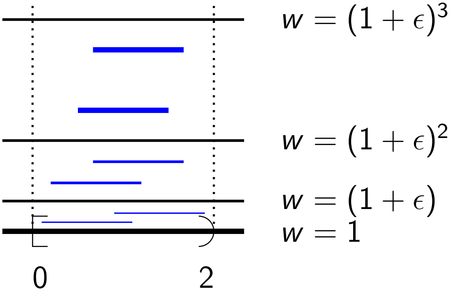
Insertion-Deletion: Unit-Length Intervals

- **Weighted Setting:** Sub-divide the window into geometric weight classes.



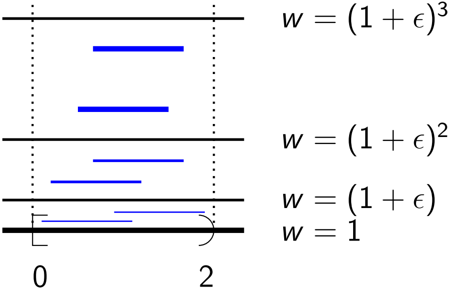
Insertion-Deletion: Unit-Length Intervals

- **Weighted Setting:**
  Sub-divide the window into geometric weight classes.



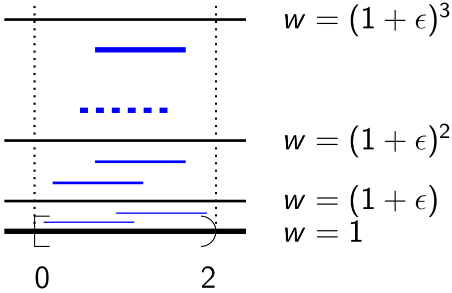
Insertion-Deletion: Unit-Length Intervals

- **Weighted Setting:** Sub-divide the window into geometric weight classes.



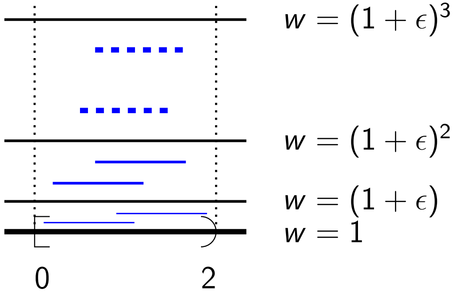
Insertion-Deletion: Unit-Length Intervals

- **Weighted Setting:** Sub-divide the window into geometric weight classes.



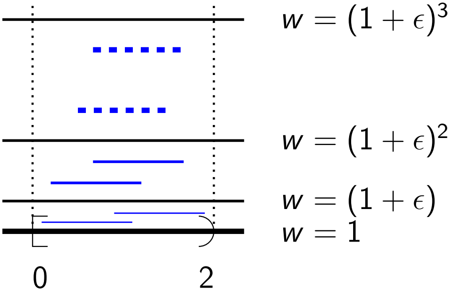
Insertion-Deletion: Unit-Length Intervals

- **Weighted Setting:** Sub-divide the window into geometric weight classes.



Insertion-Deletion: Unit-Length Intervals

- **Weighted Setting:** Sub-divide the window into geometric weight classes.
- Need all $\log(W)/\epsilon$ weight classes this time.



State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\|OPT\|)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\|OPT^*\|)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2 - \epsilon$ [EHR12]	$\mathbf{2}$	$2 - \epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2 - \epsilon$ [EHR12]	$*$	$\mathbf{\Theta(1)}$
Weighted	Unit-Length	$\mathbf{3/2} + \epsilon$	$3/2 - \epsilon$ [EHR12]	$\mathbf{2} + \epsilon$	$2 - \epsilon$ [BCW20]
	Variable-Length	$*$	$\mathbf{\Theta(1)}$	$*$	$\mathbf{\Theta(1)}$

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\|OPT\|)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\|OPT^*\|)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2 - \epsilon$ [EHR12]	$\mathbf{2}$	$2 - \epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2 - \epsilon$ [EHR12]	$*$	$\boldsymbol{\Theta(1)}$
Weighted	Unit-Length	$\mathbf{3/2} + \epsilon$	$3/2 - \epsilon$ [EHR12]	$\mathbf{2} + \epsilon$	$2 - \epsilon$ [BCW20]
	Variable-Length	$*$	$\boldsymbol{\Theta(1)}$	$*$	$\boldsymbol{\Theta(1)}$

Insertion-Deletion: Variable-Length Intervals

Insertion-Deletion: Variable-Length Intervals

Augmented Chained Index Problem - AUG-CHAIN$_p$(k)

$$\begin{array}{ccccccccc} & \xrightarrow{M^1} & & \longrightarrow & & \longrightarrow & & \xrightarrow{M^{p-1}} & \\ \underline{P_1} & & \underline{P_2} & & \cdots & & \underline{P_{p-1}} & & \underline{P_p} \\ X^1 & & X^2 & & & & X^{p-1} & & \\ & & \sigma^1 & & & & \sigma^{p-2} & & \sigma^{p-1} \\ & & X_+^1 & & & & X_+^{p-2} & & X_+^{p-2} \end{array}$$

- $X^i \in \{0,1\}^k$, $\sigma^i \in [k]$, $X_+^i = X^i[0 : \sigma^i - 1]$.
- **Promise:** $X^i[\sigma^i] = z \in \{0,1\}$ for each i.
- **Goal:** Recover z using small messages.

Insertion-Deletion: Variable-Length Intervals

Augmented Chained Index Problem - AUG-CHAIN$_\mathrm{p}$(k)

$$
\begin{tikzcd}
\underline{P_1} \arrow[r, bend left, "M^1"] & \underline{P_2} \arrow[r, bend left] & \cdots \arrow[r, bend left] & \underline{P_{p-1}} \arrow[r, bend left, "M^{p-1}"] & \underline{P_p} \\
X^1 & X^2 & & X^{p-1} & \\
 & \sigma^1 & & \sigma^{p-2} & \sigma^{p-1} \\
 & X_+^1 & & X_+^{p-2} & X_+^{p-2}
\end{tikzcd}
$$

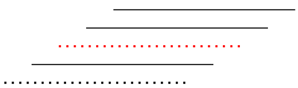
- $X^i \in \{0,1\}^k$, $\sigma^i \in [k]$, $X_+^i = X^i[0 : \sigma^i - 1]$.
- **Promise:** $X^i[\sigma^i] = z \in \{0,1\}$ for each i.
- **Goal:** Recover z using small messages.

Theorem

Any (randomised) protocol for AUG-CHAIN$_p$(k) with success probability at least $2/3$ must send a message of size $\Omega(k/p^2)$.

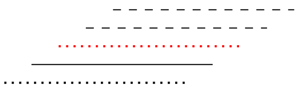
Insertion-Deletion: Variable-Length Intervals

- **Key Idea:** Prefixes allow players to introduce valid deletions in the stream.



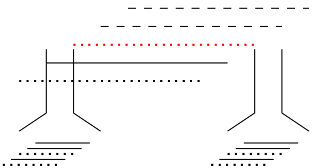
Insertion-Deletion: Variable-Length Intervals

- **Key Idea:** Prefixes allow players to introduce valid deletions in the stream.



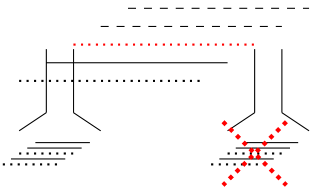
Insertion-Deletion: Variable-Length Intervals

- **Key Idea:** Prefixes allow players to introduce valid deletions in the stream.



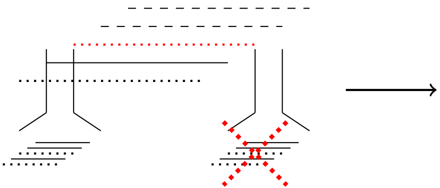
Insertion-Deletion: Variable-Length Intervals

- **Key Idea:** Prefixes allow players to introduce valid deletions in the stream.



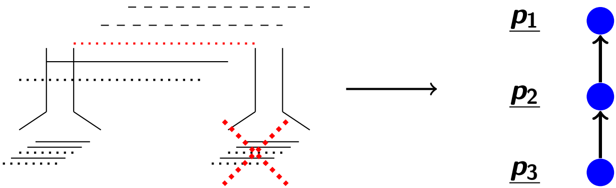
Insertion-Deletion: Variable-Length Intervals

- **Key Idea:** Prefixes allow players to introduce valid deletions in the stream.



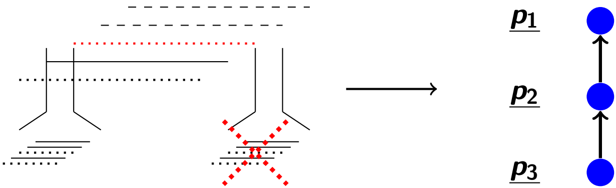
Insertion-Deletion: Variable-Length Intervals

- **Key Idea:** Prefixes allow players to introduce valid deletions in the stream.



Insertion-Deletion: Variable-Length Intervals

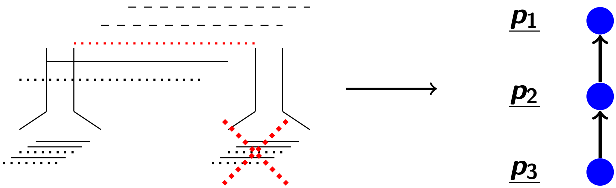
- **Key Idea:** Prefixes allow players to introduce valid deletions in the stream.



- If $Z = 1$, then $|OPT| = p - 1$.
- If $Z = 0$, then $|OPT| = 1$

Insertion-Deletion: Variable-Length Intervals

- **Key Idea:** Prefixes allow players to introduce valid deletions in the stream.



- If $Z = 1$, then $|OPT| = p - 1$.
- If $Z = 0$, then $|OPT| = 1$

Theorem

Any $(p - \epsilon)$-approximate alg. solves the AUG-CHAIN$_p(k)$ instance, and so requires space $\Omega(k/p) = \Omega(n)$ for constant p.

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\|OPT\|)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\|OPT^*\|)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2 - \epsilon$ [EHR12]	$\mathbf{2}$	$2 - \epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2 - \epsilon$ [EHR12]	$*$	$\mathbf{\Theta(1)}$
Weighted	Unit-Length	$\mathbf{3/2} + \epsilon$	$3/2 - \epsilon$ [EHR12]	$\mathbf{2} + \epsilon$	$2 - \epsilon$ [BCW20]
	Variable-Length	$*$	$\mathbf{\Theta(1)}$	$*$	$\mathbf{\Theta(1)}$

State of the Art

		Insertion-Only		Insertion-Deletion	
		UB - $O_\epsilon(\lvert OPT \rvert)$	LB - $\Omega(n)$	UB - $\tilde{O}_\epsilon(\lvert OPT^* \rvert)$	LB - $\Omega(n)$
Unweighted	Unit-Length	$3/2$ [EHR12]	$3/2-\epsilon$ [EHR12]	$\mathbf{2}$	$2-\epsilon$ [BCW20]
	Variable-Length	2 [EHR12]	$2-\epsilon$ [EHR12]	$*$	$\boldsymbol{\Theta(1)}$
Weighted	Unit-Length	$\mathbf{3/2}+\epsilon$	$3/2-\epsilon$ [EHR12]	$\mathbf{2}+\epsilon$	$2-\epsilon$ [BCW20]
	Variable-Length	$*$	$\boldsymbol{\Theta(1)}$	$*$	$\boldsymbol{\Theta(1)}$

Further Work

Further Work

- The interval setting is now well understood!

Further Work

- The interval setting is now well understood!
- Similar open questions remain in higher dimensions.

Further Work

- The interval setting is now well understood!
- Similar open questions remain in higher dimensions.
- [CDK18] ask about a gap for unit squares:

Further Work

- The interval setting is now well understood!
- Similar open questions remain in higher dimensions.
- [CDK18] ask about a gap for unit squares:
  - **UB:** 3-approx.
  - **LB:** 5/2-approx.

References I

Ainesh Bakshi, Nadiia Chepurko, and David P. Woodruff.
Weighted maximum independent set of geometric objects in turnstile streams.
In Jaroslaw Byrka and Raghu Meka, editors, Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, August 17-19, 2020, Virtual Conference, volume 176 of LIPIcs, pages 64:1–64:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

Graham Cormode, Jacques Dark, and Christian Konrad.
Approximating the caro-wei bound for independent sets in graph streams.
In Jon Lee, Giovanni Rinaldi, and Ali Ridha Mahjoub, editors, Combinatorial Optimization - 5th International Symposium, ISCO 2018, Marrakesh, Morocco, April 11-13, 2018, Revised

References II

Selected Papers, volume 10856 of Lecture Notes in Computer Science, pages 101–114. Springer, 2018.

Graham Cormode, Jacques Dark, and Christian Konrad. Independent Sets in Vertex-Arrival Streams. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, 46th International Colloquium on Automata, Languages, and Programming (ICALP 2019), volume 132 of Leibniz International Proceedings in Informatics (LIPIcs), pages 45:1–45:14, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

Yuval Emek, Magnus M. Halldorsson, and Adi Rosen. Space-Constrained Interval Selection. In 39th International Colloquium on Automata, Languages, and Prograr July 2012.

References III

Moran Feldman, Ashkan Norouzi-Fard, Ola Svensson, and Rico Zenklusen.
The one-way communication complexity of submodular maximization with applications to streaming and robustness.
In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020, pages 1363–1374. ACM, 2020.

Hossein Jowhari, Mert Sağlam, and Gábor Tardos.
Tight bounds for lp samplers, finding duplicates in streams, and related problems.
In Proceedings of the Thirtieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS '11, pages 49–58, New York, NY, USA, 2011. Association for Computing Machinery.