Improved Weighted Matching in the Sliding Window Model

Cezar-Mihail Alexandru, Pavel Dvořák, Christian Konrad, Kheeran K. Naidu

Presented by Cezar-Mihail Alexandru at STACS 2023

March 2023

Table of Contents

Table of Contents

The world of sliding-window streaming.

Let A be a binary array and $L > 0$ be a positive integer.

We insert elements into A.

Unable to store the entire input.

Goal: Infer properties of the L (the window length) most recent elements of the array using a small amount of memory.

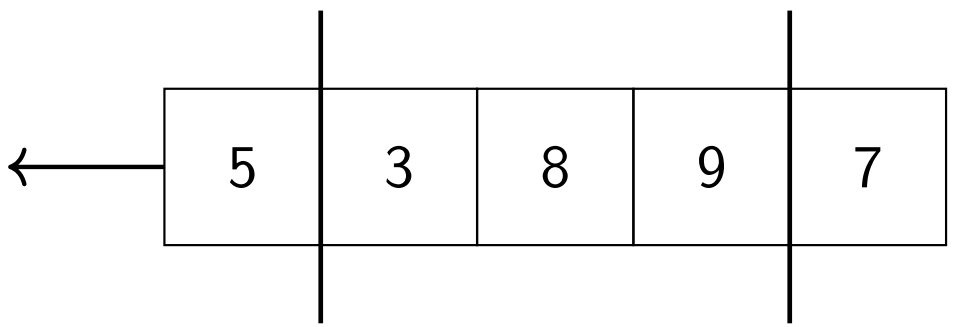


Figure: **A stream of window length** $L = 3$

More about the sliding window model

Submodels:

- Fixed window
- Variable window
- Burst model (timestamp based).

Some of the main **techniques** are:

- Exponential histogram ([**Datar et al. 2002**])
- Smooth histogram ([**Braverman et al. 2007**])

If the length of the window is the length of the whole stream, then we are in the insertion-only model.

Streaming on graphs

Edge arrival model: Stream elements consist of individual edges of the graph.

Goal: Semi-streaming memory $\tilde{O}(n)$ where n is the number of vertices.

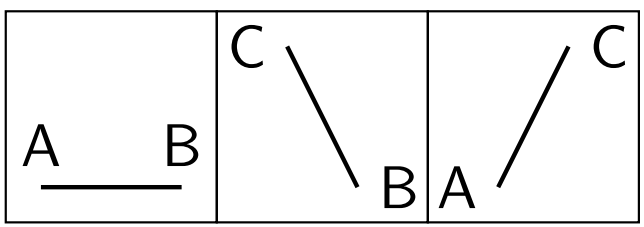


Figure: **Edge arrival stream for 3-cycle graph**

Maximum matching problem

A **matching** is a set of edges with unique endpoints.

A **maximal** matching is a matching which is maximal under the $\subseteq$ relation

A **maximum** matching is a matching with the largest size.

Goal: Find a matching as large as possible.

Figure: **A maximal non-maximum matching**

Maximum matching problem and streaming

Many results are based on Greedy Algorithm.

Greedy always gives a maximal matching (2.0 approximation) in insertion-only model.

Doing better than 2.0 in insertion-only model remains an open problem.

Maximum matching in streaming research

Maximum matching and its variants are one of the most widely studied problems in streaming on graphs. There is a wide array of results on:

- Multi-pass matching
- Random order matching
- Weighted matching
- Matching in sliding window
- Memory lower bounds on matching
- And many more.....

[**Dark et al 2020**],[**Konrad et al 2021**], [**Feldman et al 2021**], [**Levin et al. 2021**], [**Biabani et al. 2021**], [**Kapralov 2021**], [**Bezenhad et al 2022**], [**Fischer et al 2022**] [**Assadi 2022**], [**Naidu et al 2023**], etc.

Maximum weighted matching

Each edge has a positive integer associated with it (weight)

Weight of a matching = Sum of weights of its edges.

Maximum weighted matching (MWM): Find the matching with maximum weight.

Best result in insertion-only model: $2 + \varepsilon$ approximation factor for any $\varepsilon > 0$ (Paz-Schwartzmann algorithm).

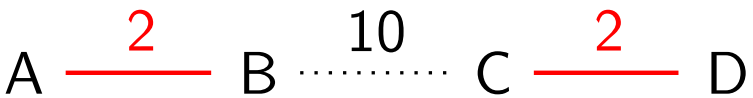


Figure: **A matching which is not MWM**

Matching in sliding window

Maintaining a maximal matching is no longer straightforward.

State-of the art semistreaming results before our work:

1. Unweighted matching: 3.0 ([**Crouch et al. 2013**])
2. Weighted matching: $3.5 + \varepsilon$ ([**Biabani et al. 2021**])

Both results use the smooth histogram method by exploiting a "lookahead" property of the matchings.

Our main contributions

1. There exists a deterministic sliding-window algorithm which gives a $2+\varepsilon$ approximation on the MWM problem by using $O(\sqrt{nL})$ space where L is the sliding window length.

2. State-of-the-art semi-streaming approximation factor of MWM in sliding window model can be improved to $3+\varepsilon$.

Table of Contents

MWM as a linear programming problem

We can interpret MWM as a linear (integer) programming problem as follows:

$$\begin{array}{ll} \text{maximize} & \sum_{e \in E} w(e) \cdot x_e \\ \text{subject to} & \forall v \in V, \sum_{e \ni v} x_e \leq 1 \\ & \forall e \in E, x_e \geq 0 \end{array}$$

Its dual problem is:

$$\begin{array}{ll} \text{minimize} & \sum_{v \in V} x_v \\ \text{subject to} & \forall e = uv \in E, x_v + x_u \geq w(e) \\ & \forall v \in V, x_v \geq 0 \end{array}$$

Motivation behind Paz-Schwartzman algorithm

Fact (Weak duality theorem)

Let P be a linear programming problem in standard form with its feasible solution p and D be its dual with its feasible solution d. Then, $p \leq d$.

In our setting, a dual solution is always an upper bound of the maximum matching

Strategy: Construct a potential function $\phi : V \to \mathbb{R}$ which represents a "good" dual solution.

Building a "good" matching in tandem with a "good" potential should reinforce each other.

Outline of Paz-Schwartzman algorithm ($2+\varepsilon$ approximation)

- Before the stream, set ϕ to be 0 for all vertices.
- When an edge $e = uv$ arrives in the stream, if $w(e) < (1+\varepsilon)(\phi(v)+\phi(u))$, then ignore e.
- Otherwise, memorize e and define its reduced weight $w'(e) = w(e) - \phi(v) - \phi(u)$.
- Update $\phi(u)$ and $\phi(v)$ by adding to them $w'(e)$.
- At the end of the stream, do reverse greedy on the memorized edges.

Table of Contents

$O(\sqrt{nL})$ memory result

Open problem: Can we maintain a 2-approximation for the maximum matching in the sliding window model using semi-streaming space?

Progress: $2+\varepsilon$ approximation possible if we allow $O(\sqrt{nL})$ memory (at worst $O(n\sqrt{n})$ assuming the graph is simple at each timestamp).

This result applies for both unweighted and weighted matching!

Theorem (simplified)

Let $\varepsilon > 0$. Then, there exists a **deterministic sliding window** *algorithm which calculates a $2+\varepsilon$ approximation of MWM using $\tilde{O}(\sqrt{nL})$ space.*

Reverse Paz-Schwartzman

Let us apply Paz-Schwartzmann algorithm in reverse order instead of forward order.

Advantage: Easy to maintain in sliding window model.

Disadvantage: Reverse matching requires storing $\Omega(n^2)$ edges.

The best of two worlds

Neither forward matching nor reverse matching are fully satisfactory.

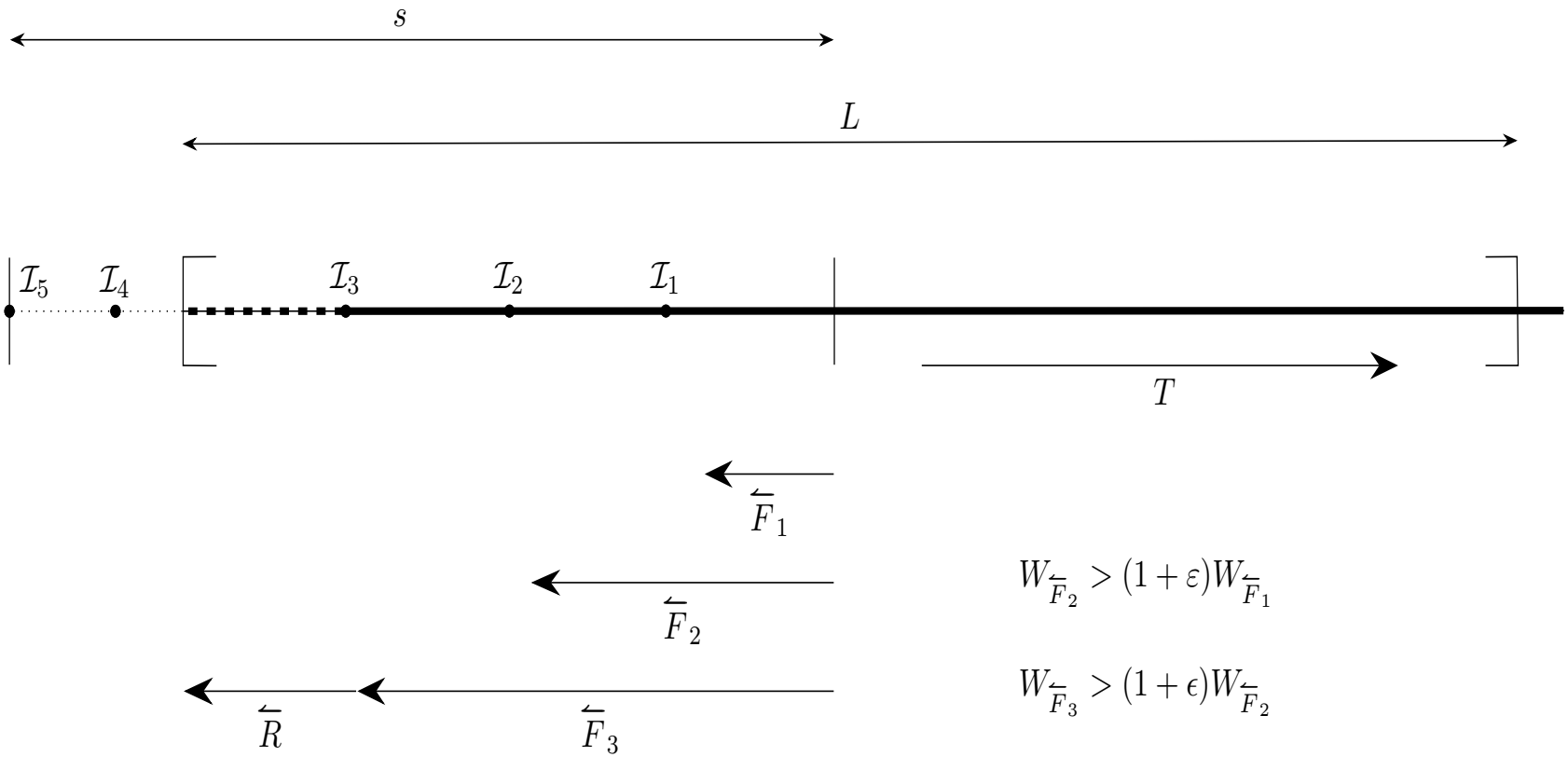
Suggestion: Do reverse matching at the beginning on a small enough block and continue with forward matching after the block ends!

Turns out this is the right and effective approach, making the best of two worlds .

Outline of the algorithm

- Divide the stream into blocks of size $s = O(\sqrt{nL})$ where L is the window length.
- For each block, when it becomes fully available, apply Paz-Schwartzman on the edges of the block in reverse order.
- Mark a checkpoint (i.e bucket) whenever the sum of all the reduced weights reaches $(1+\varepsilon)^k$ for some natural k.
- Then, for each checkpoint, continue the matching until the end of the stream.
- Output the largest checkpoint with unexpired elements

Schematic of algorithm



Proof outline: Goal and some notation

We need to make sure the still active residue not considered by the output checkpoint ($\overleftarrow{R}$ in the figure) does not compromoise the current matching ($\overleftarrow{F_3}\,T$ in the figure).

Let $\overleftarrow{F}$ be the reverse stream from the end of the block to the biggest fully active checkpoint.

Let T be the forward stream after the block as in the figure.

Proof outline: Linear programming strikes back

- We will use the dual equation of MWM.
- By giving a suitable dual solution related to sums of reduced weights, we can prove that the residual part R yields an insignificat part of the whole stream $\overleftarrow{F}RT$.
- Therefore, we can "forget" the residue R without harming the approximation factor too much.
- In this manner, we can obtain a $2+\varepsilon$ approximation of the optimal matching.

Table of Contents

"Lookahead" property

Paz-Schwartzman has a **"lookahead"** property.

Assume $S = ABC$ is a stream and the outputs of the algorithm on prefixes AB and B are "close enough" .

Then, the run BC yields an approximation factor realtively "close" to the run on ABC.

By using this observation and a bucketing technique called the "smooth histogram" method, Biabani et al. obtained a $3.5 + \varepsilon$ approximation

$3.0 + \varepsilon$ approximation result

Instead of looking at the direct output of the algorithm, we inspect a quantity called the sum of **reduced weights**.

Sums of reduced weights still maintain the "lookahead" property.

Employing a slightly modified smooth histogram, we yield the following.

Theorem (simplified)

There exists a **deterministic sliding window** *algorithm which gives a* $3.0 + \varepsilon$ *approximation on the Maximum Weighted matching using* $\tilde{O}(n)$ *space.*

Further directions

Can we get below 3.0 in sliding window model for maximum unweighted matching using semi-streaming space?

Can we establish memory lower bounds for maximum matching problem in sliding window model?

Thank you!

End of presentation! Thank you!

References