Temporal matchings

Julien Baste

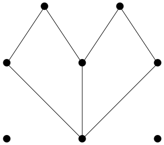
University of Lille

July 12, 2021

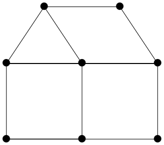
Joint work with Binh-Minh Bui-Xuan and Antoine Roux

Temporal graphs are life!

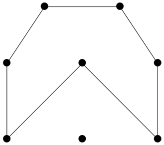
A temporal graph is a collection of graphs, on the same vertex set, indexed by time.



t=1



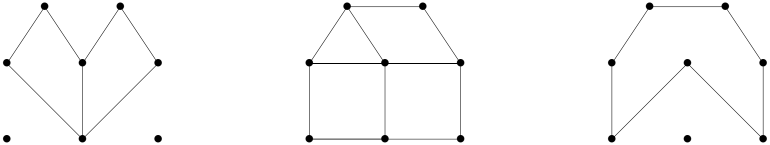
t=2



t=3

Temporal graphs are life!

A temporal graph is a collection of graphs, on the same vertex set, indexed by time.



t=1 t=2 t=3

Some examples of used temporal graphs:

- Enron
  - Enron Email Dataset
  - 150 users
  - 0.5M messages
- Rollernet
  - Rollerblading tour in Paris
  - 62 participants
  - Proximity detection every 15 sec

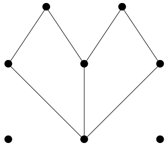
Who will match with me?

Goal: Generalize the MATCHING problem to temporal graphs.

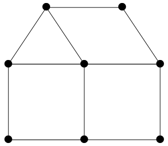
Who will match with me?

Goal: Generalize the MATCHING problem to temporal graphs.

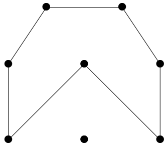
Question: How to do it in an interesting way?



t=1



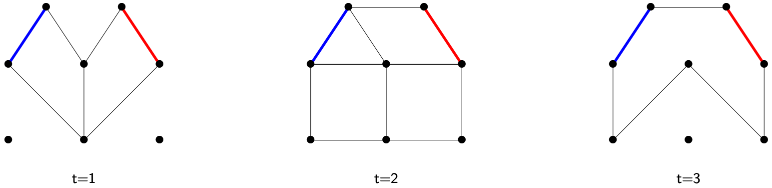
t=2



t=3

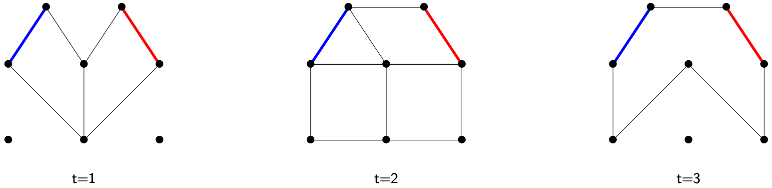
Who will match with me?

Goal: Generalize the MATCHING problem to temporal graphs.
Question: How to do it in an interesting way?

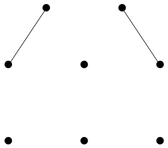


Answer: A set of edges that, at any time, induces a matching and such that each edge exists at each time.

Goal: Generalize the MATCHING problem to temporal graphs.
Question: How to do it in an interesting way?

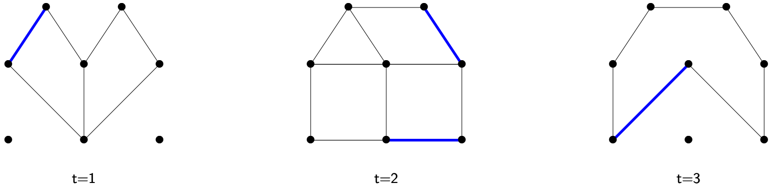


Answer: A set of edges that, at any time, induces a matching and such that each edge exists at each time.
Issue: This is MATCHING in the intersection graph.



Who will match with me?

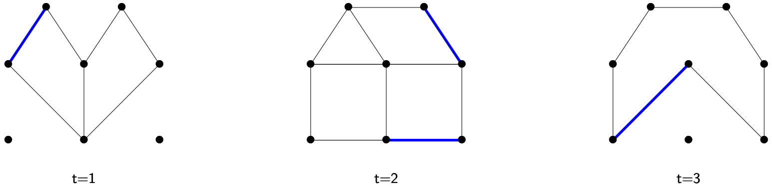
Goal: Generalize the MATCHING problem to temporal graphs.
Question: How to do it in an interesting way?



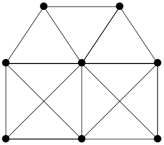
Answer: A matching in the graphs of the link steams such that the same vertex is not used twice.

Who will match with me?

Goal: Generalize the MATCHING problem to temporal graphs.
Question: How to do it in an interesting way?

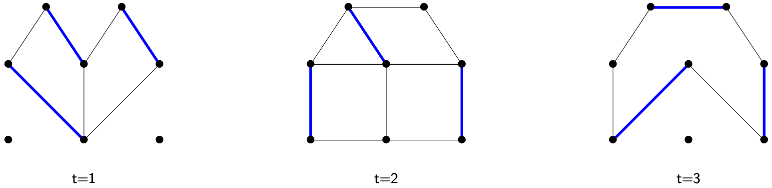


Answer: A matching in the graphs of the link steams such that the same vertex is not used twice.
Issue: This is MATCHING in the union graph.



Who will match with me?

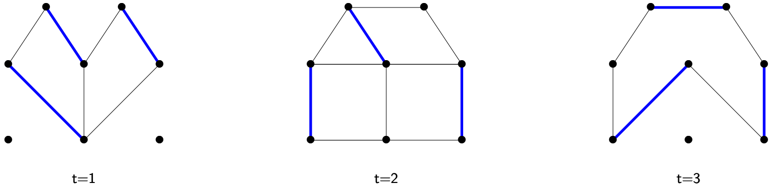
Goal: Generalize the MATCHING problem to temporal graphs.
Question: How to do it in an interesting way?



Answer: A matching in the disjoint union of the graphs of the temporal graph.

Who will match with me?

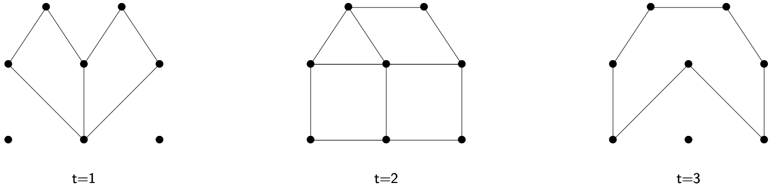
Goal: Generalize the MATCHING problem to temporal graphs.
Question: How to do it in an interesting way?



Answer: A matching in the disjoint union of the graphs of the temporal graph.
Issue: This is MATCHING in the disjoint union graph.

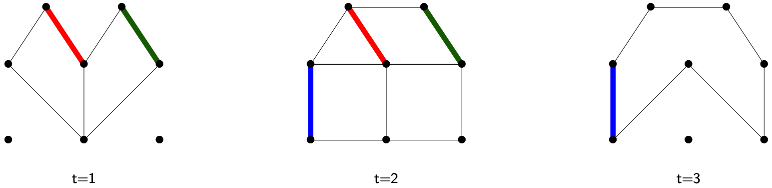
Who will match with me?

Goal: Generalize the MATCHING problem to temporal graphs.
Question: How to do it in an interesting way?



Answer: The edge of our γ-matching should exist during γ consecutive times and at each time, this should be a matching.

Goal: Generalize the MATCHING problem to temporal graphs.
Question: How to do it in an interesting way?



Answer: The edge of our γ-matching should exist during γ consecutive times and at each time, this should be a matching.

γ-Matching is NP-hard

Theorem

Given $\gamma \geq 2$, γ-Matching is NP-hard.

Theorem

Given $\gamma \geq 2$, γ-MATCHING is NP-hard.

The reduction is from 3-SAT.

$\varphi = (\overline{w} \vee x \vee \overline{y}) \wedge (\overline{w} \vee \overline{x} \vee z)$; $\gamma = 3$

γ-Matching is NP-hard

Theorem

Given $\gamma \geq 2$, γ-Matching is NP-hard.

The reduction is from 3-SAT.

0

1

2

3

4

5

6

7

8

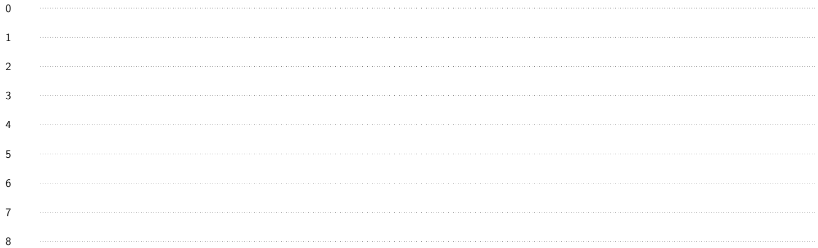
$$\varphi = (\overline{w} \vee x \vee \overline{y}) \wedge (\overline{w} \vee \overline{x} \vee z);\ \gamma = 3$$

γ-Matching is NP-hard

Theorem

Given $\gamma \geq 2$, γ-Matching is NP-hard.

The reduction is from 3-SAT.



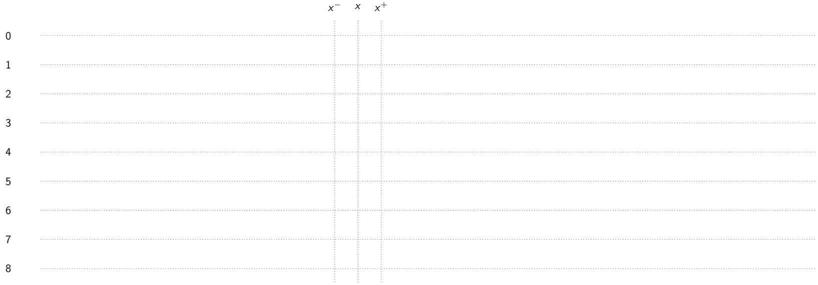
$$\varphi = (\overline{w} \vee x \vee \overline{y}) \wedge (\overline{w} \vee \overline{x} \vee z);\ \gamma = 3$$

γ-MATCHING is NP-hard

Theorem

Given $\gamma \geq 2$, γ-MATCHING is NP-hard.

The reduction is from 3-SAT.



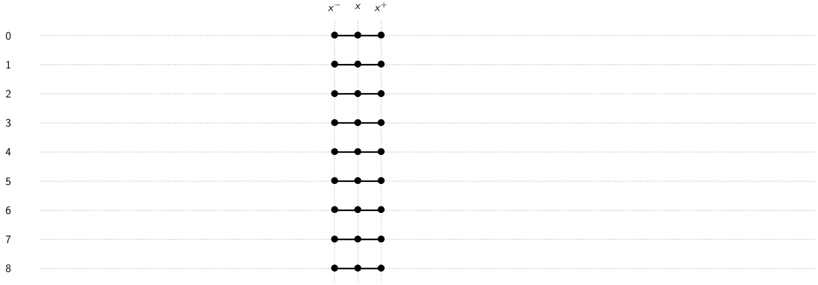
$$\varphi = (\overline{w} \vee x \vee \overline{y}) \wedge (\overline{w} \vee \overline{x} \vee z);\ \gamma = 3$$

γ-Matching is NP-hard

Theorem

Given $\gamma \geq 2$, γ-Matching is NP-hard.

The reduction is from 3-SAT.



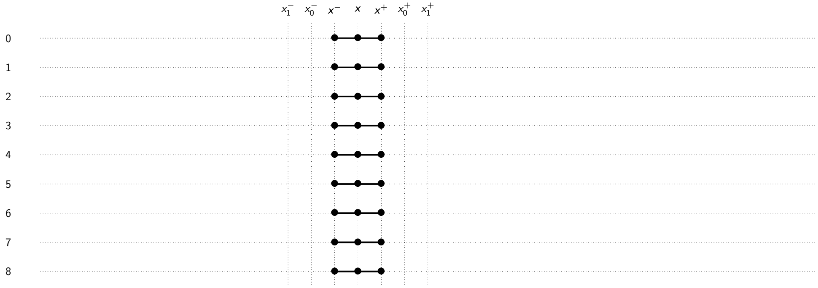
$$\varphi = (\overline{w} \vee x \vee \overline{y}) \wedge (\overline{w} \vee \overline{x} \vee z);\ \gamma = 3$$

γ-Matching is NP-hard

Theorem

Given $\gamma \geq 2$, γ-Matching is NP-hard.

The reduction is from 3-SAT.



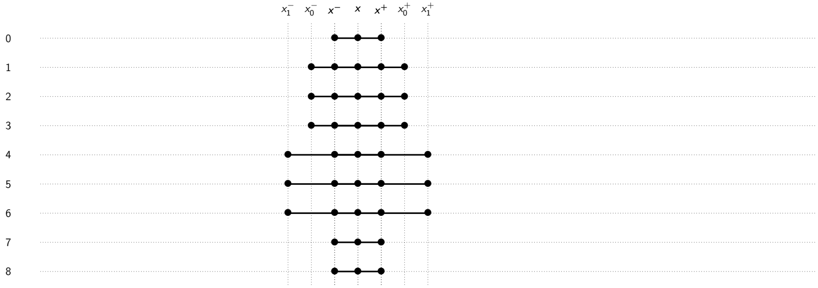
$$\varphi = (\overline{w} \vee x \vee \overline{y}) \wedge (\overline{w} \vee \overline{x} \vee z);\ \gamma = 3$$

γ-Matching is NP-hard

Theorem

Given $\gamma \geq 2$, γ-Matching is NP-hard.

The reduction is from 3-SAT.

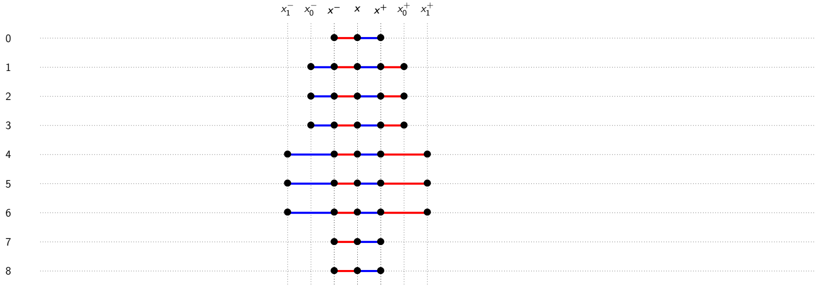


$\varphi = (\overline{w} \vee x \vee \overline{y}) \wedge (\overline{w} \vee \overline{x} \vee z)$; $\gamma = 3$

Theorem

Given $\gamma \geq 2$, γ-MATCHING is NP-hard.

The reduction is from 3-SAT.

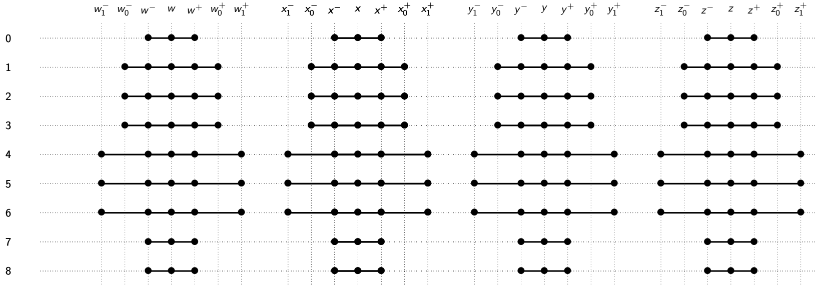


$$\varphi = (\overline{w} \vee x \vee \overline{y}) \wedge (\overline{w} \vee \overline{x} \vee z);\ \gamma = 3$$

Theorem

Given $\gamma \geq 2$, γ-Matching is NP-hard.

The reduction is from 3-SAT.

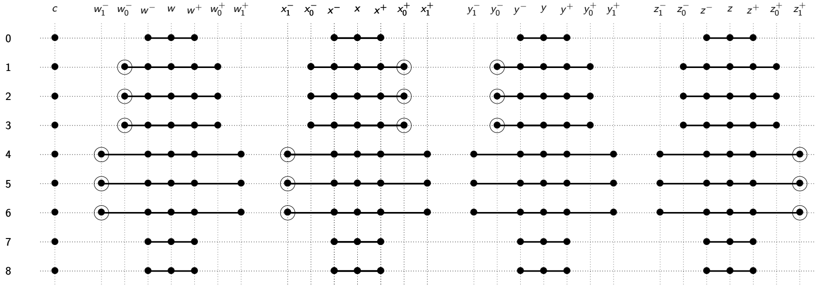


$$\varphi = (\overline{w} \vee x \vee \overline{y}) \wedge (\overline{w} \vee \overline{x} \vee z);\ \gamma = 3$$

Theorem

Given $\gamma \geq 2$, γ-MATCHING is NP-hard.

The reduction is from 3-SAT.



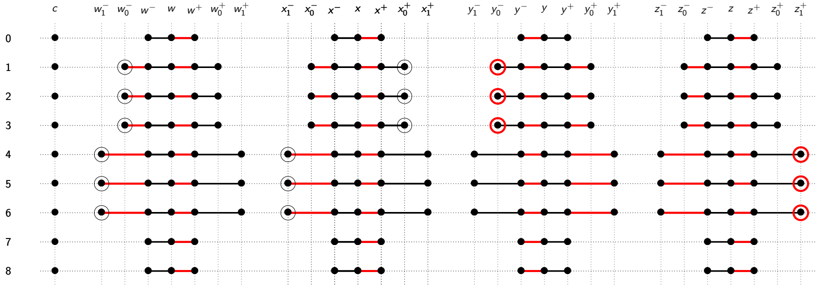
$$\varphi = (\overline{w} \vee x \vee \overline{y}) \wedge (\overline{w} \vee \overline{x} \vee z);\ \gamma = 3$$

γ-Matching is NP-hard

Theorem

Given $\gamma \geq 2$, γ-Matching is NP-hard.

The reduction is from 3-SAT.



$$\varphi = (\overline{w} \vee x \vee \overline{y}) \wedge (\overline{w} \vee \overline{x} \vee z);\ \gamma = 3$$

Solution: w, x, $\overline{y}$, z

In normal graphs: Any maximal matching is a 2-approximation.

Trivial approximation of γ-Matching

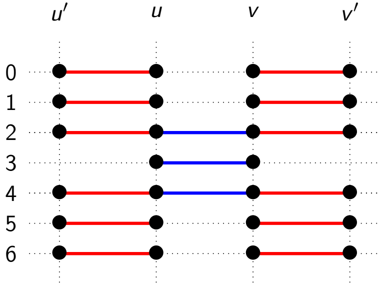
In normal graphs: Any maximal matching is a 2-approximation.

In temporal graphs: Any maximal γ-matching is a 4-approximation.

Trivial approximation of γ-MATCHING

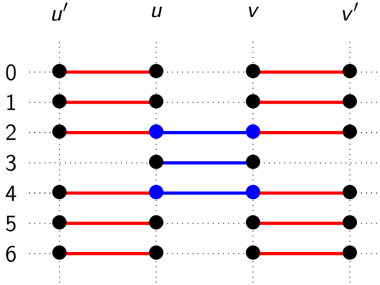
In normal graphs: Any maximal matching is a 2-approximation.

In temporal graphs: Any maximal γ-matching is a 4-approximation.



In normal graphs: Any maximal matching is a 2-approximation.

In temporal graphs: Any maximal γ-matching is a 4-approximation.



NP-hard problem? What's next?

Theorem

Given $\gamma \geq 2$, there exists a 2-approximation for γ-Matching.

NP-hard problem? What's next?

Theorem

*Given $\gamma \geq 2$, there exists a 2-approximation for γ-*Matching.

A greedy algorithm will do the job:

- Take the first (in time) γ-edge available.
- Remove this γ-edge and every edge incident to it.
- Repeat.

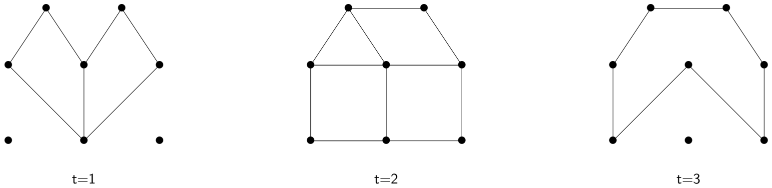
NP-hard problem? What's next?

Theorem

*Given $\gamma \geq 2$, there exists a 2-approximation for γ-*MATCHING*.*

A greedy algorithm will do the job:

- Take the first (in time) γ-edge available.
- Remove this γ-edge and every edge incident to it.
- Repeat.



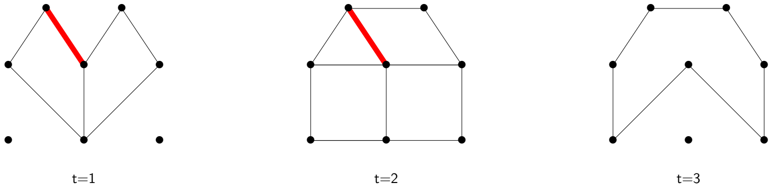
NP-hard problem? What's next?

Theorem

Given $\gamma \geq 2$, there exists a 2-approximation for γ-$\textsc{Matching}$.

A greedy algorithm will do the job:

- Take the first (in time) γ-edge available.
- Remove this γ-edge and every edge incident to it.
- Repeat.



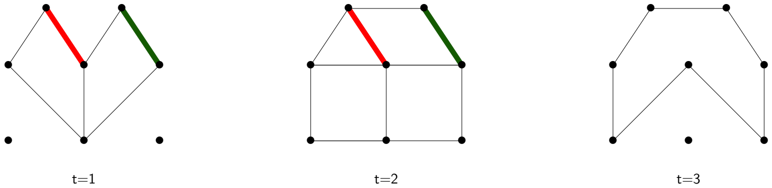
NP-hard problem? What's next?

Theorem

Given $\gamma \geq 2$, there exists a 2-approximation for γ-MATCHING.

A greedy algorithm will do the job:

- Take the first (in time) γ-edge available.
- Remove this γ-edge and every edge incident to it.
- Repeat.



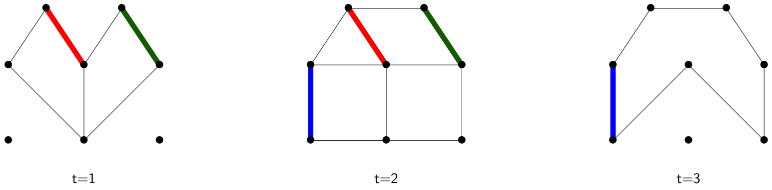
NP-hard problem? What's next?

Theorem

*Given $\gamma \geq 2$, there exists a 2-approximation for γ-*MATCHING.

A greedy algorithm will do the job:

- Take the first (in time) γ-edge available.
- Remove this γ-edge and every edge incident to it.
- Repeat.

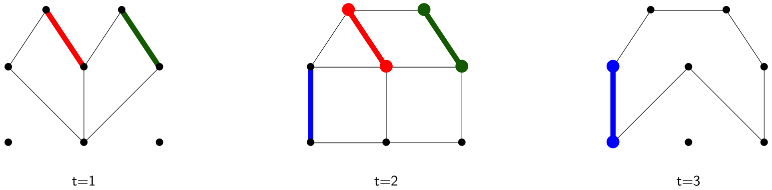


Theorem

*Given $\gamma \geq 2$, there exists a 2-approximation for γ-*MATCHING.

A greedy algorithm will do the job:

- Take the first (in time) γ-edge available.
- Remove this γ-edge and every edge incident to it.
- Repeat.



NP-hard problem? What's next?

γ-MATCHING
Input: A temporal graph L and an integer k.
Question: Is L contains a γ-matching of size at least k?

Theorem

There exists a polynomial-time algorithm that for each instance (L, k), either correctly determines if L contains a γ-matching of size k, or returns an equivalence instance (L', k) such that the number of edges of L' is $2(k-1)(2k-1)\gamma^2$.

NP-hard problem? What's next?

γ-MATCHING
Input: A temporal graph L and an integer k.
Question: Is L contains a γ-matching of size at least k?

Theorem

There exists a polynomial-time algorithm that for each instance (L, k), either correctly determines if L contains a γ-matching of size k, or returns an equivalence instance (L', k) such that the number of edges of L' is $2(k-1)(2k-1)\gamma^2$.

This can be done as follows:

NP-hard problem? What's next?

γ-Matching
Input: A temporal graph L and an integer k.
Question: Is L contains a γ-matching of size at least k?

Theorem

There exists a polynomial-time algorithm that for each instance (L, k), either correctly determines if L contains a γ-matching of size k, or returns an equivalence instance (L', k) such that the number of edges of L' is $2(k-1)(2k-1)\gamma^2$.

This can be done as follows:

- Run the greedy algorithm $\mathcal{A}$.

NP-hard problem? What's next?

γ-Matching
Input: A temporal graph L and an integer k.
Question: Is L contains a γ-matching of size at least k?

Theorem

There exists a polynomial-time algorithm that for each instance (L, k), either correctly determines if L contains a γ-matching of size k, or returns an equivalence instance (L', k) such that the number of edges of L' is $2(k-1)(2k-1)\gamma^2$.

This can be done as follows:

- Run the greedy algorithm $\mathcal{A}$.
- If $\mathcal{A}$ returns a value at least k, then answer YES.

NP-hard problem? What's next?

γ-Matching
Input: A temporal graph L and an integer k.
Question: Is L contains a γ-matching of size at least k?

Theorem

There exists a polynomial-time algorithm that for each instance (L, k), either correctly determines if L contains a γ-matching of size k, or returns an equivalence instance (L', k) such that the number of edges of L' is $2(k-1)(2k-1)\gamma^2$.

This can be done as follows:

- Run the greedy algorithm $\mathcal{A}$.
- If $\mathcal{A}$ returns a value at least k, then answer YES.
- If $\mathcal{A}$ returns a value at most $\frac{k-1}{2}$, then answer NO.

NP-hard problem? What's next?

γ-MATCHING
Input: A temporal graph L and an integer k.
Question: Is L contains a γ-matching of size at least k?

Theorem

There exists a polynomial-time algorithm that for each instance (L, k), either correctly determines if L contains a γ-matching of size k, or returns an equivalence instance (L', k) such that the number of edges of L' is $2(k-1)(2k-1)\gamma^2$.

This can be done as follows:

- Run the greedy algorithm $\mathcal{A}$.
- If $\mathcal{A}$ returns a value at least k, then answer YES.
- If $\mathcal{A}$ returns a value at most $\frac{k-1}{2}$, then answer NO.
- Otherwise, for each bottom vertex returned by $\mathcal{A}$, keep at most $2k-1$ γ-edges incident to it.

Conclusion

- γ-Matching is NP-hard.
- Any maximal γ-matching is a 4-approximation.
- There exists a 2-approximation for γ-Matching.
- γ-Matching has a kernel of size $2(k-1)(2k-1)\gamma^2$.

Thanks for your attention