Near-optimal computation of temporal matching

Julien Baste
Binh-Minh Bui-Xuan
Ngoc-Trung Nguyen
Timothé Picavet
Antoine Roux

Glasgow, July 2021

Motivations

Real world network analysis:

– web log, analytics

– CRM statistics, BI reporting

– criminology

⇒ timestamped information!

Cooperation scheduling:

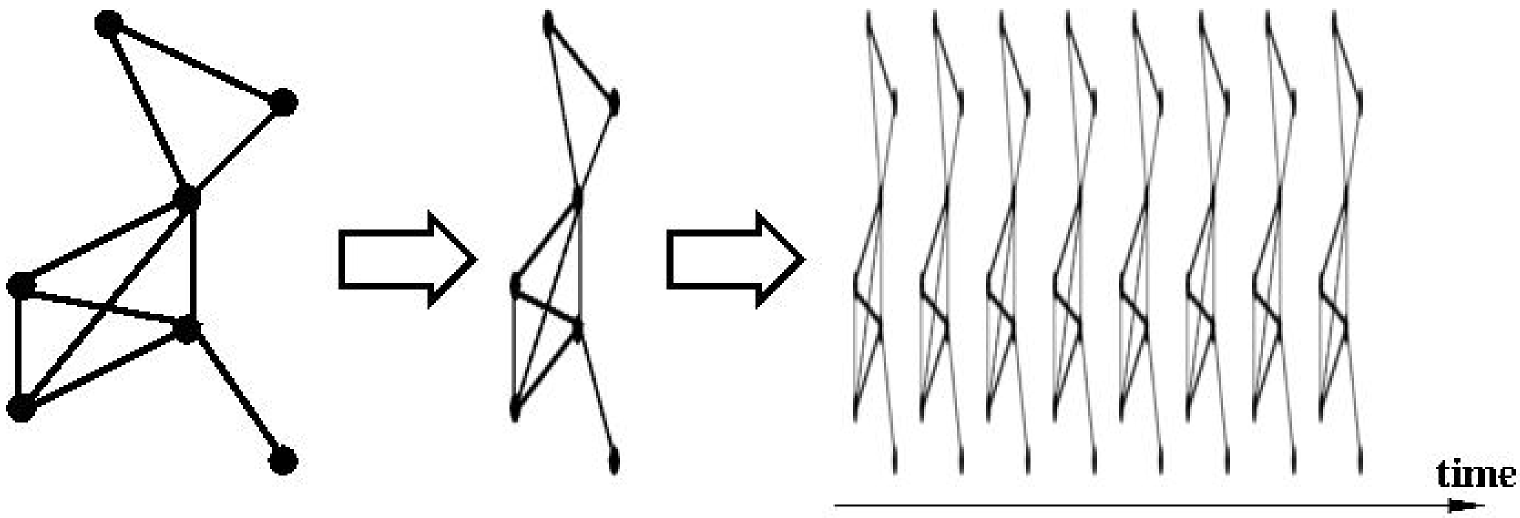
– XP, peer programming, coworker

– edges, instant edges, interval edges

– matching (independent edges)

⇒ temporal matching!

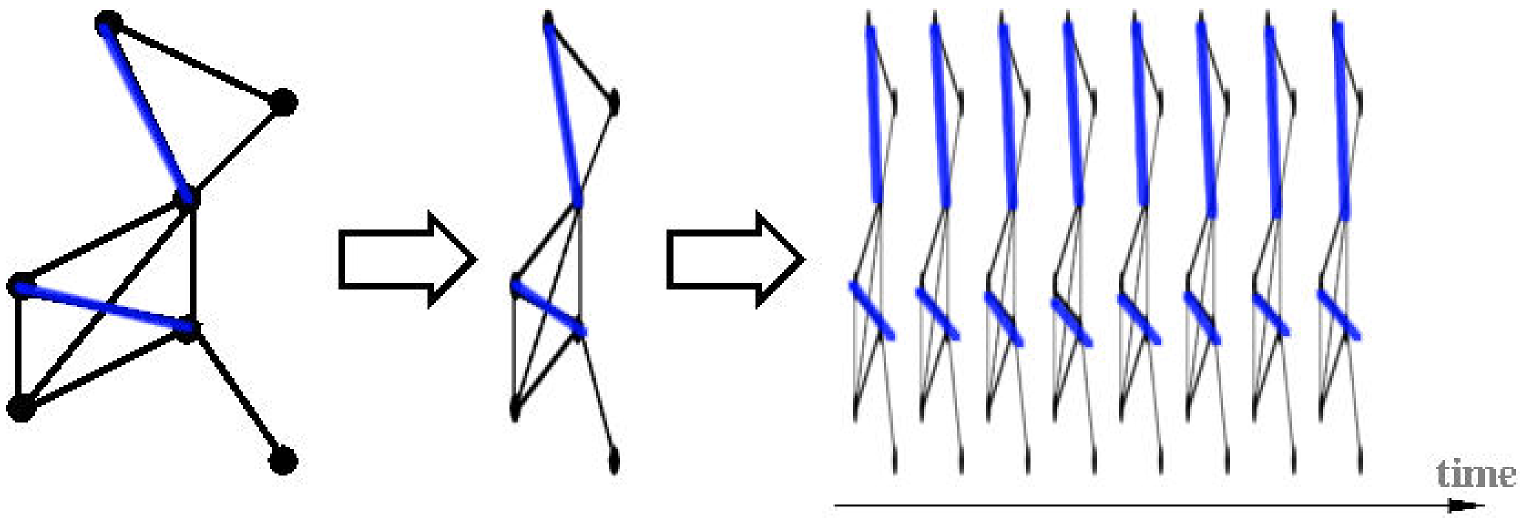
Timestamped edges and matching



graph

link stream

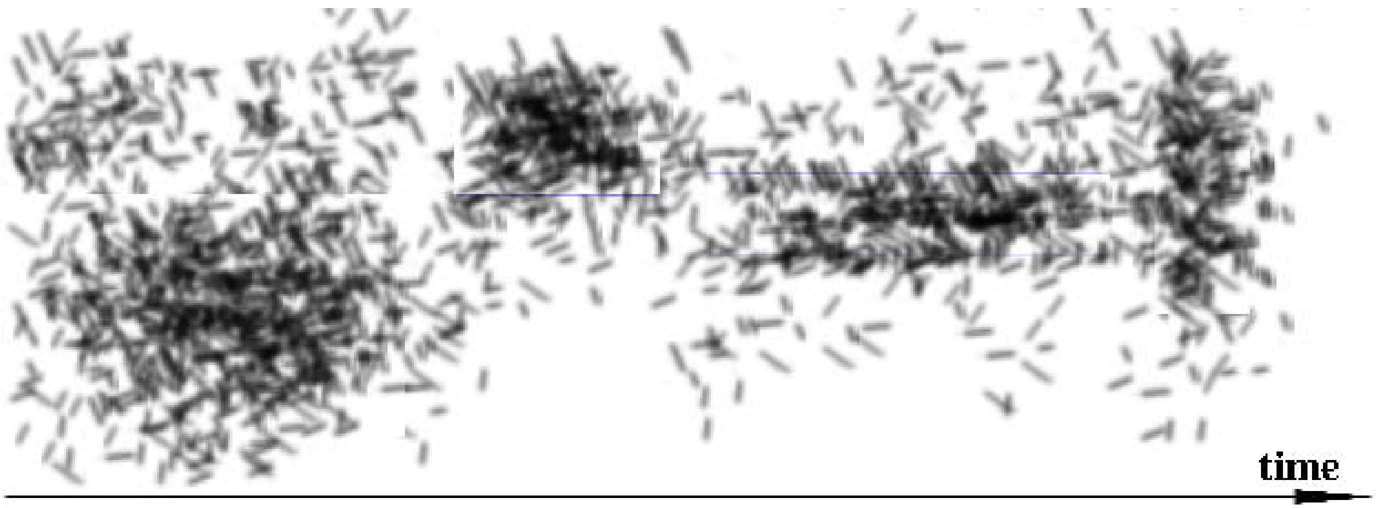
Timestamped edges and matching



graph

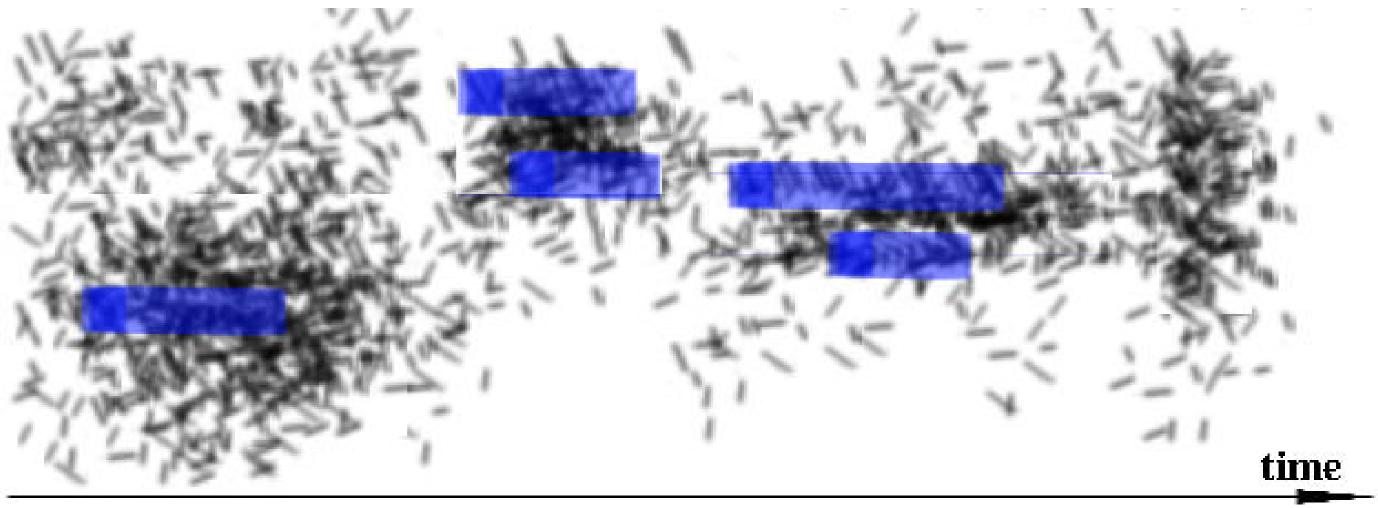
link stream

Timestamped edges and matching



Real world link stream

Our question: maximize collaborations?



QUESTION: given a minimum coworking duration, maximize the number of collaborations?

Formal definition: γ-matching

A link stream is a triple $L = (T, V, E)$ s.t.:

- $T = [\![1, \tau]\!]$
- V is a finite set of vertices
- $E \subseteq \{(t, uv) : t \in T \wedge u \in V \wedge v \in V \wedge u \neq v\}$

A γ-edge Γ is a set of one edge repeated γ times:

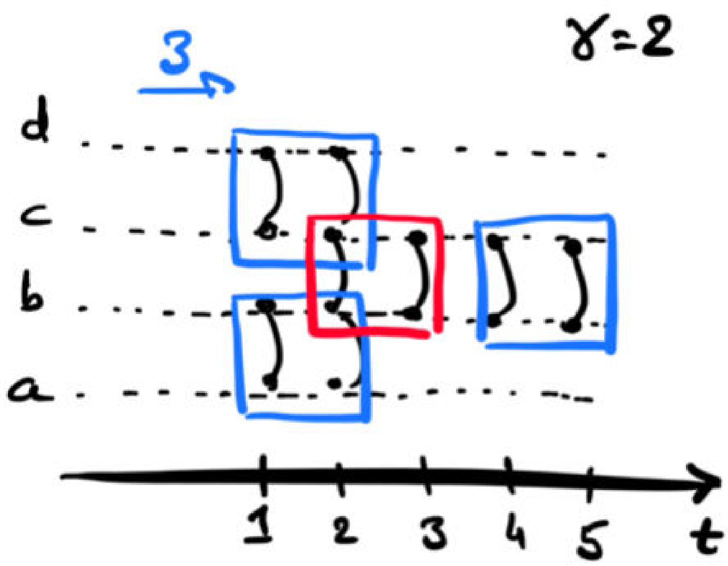
- $\Gamma = \{(t, uv) \in E : t_0 \leq t \leq t_0 + \gamma - 1\}$

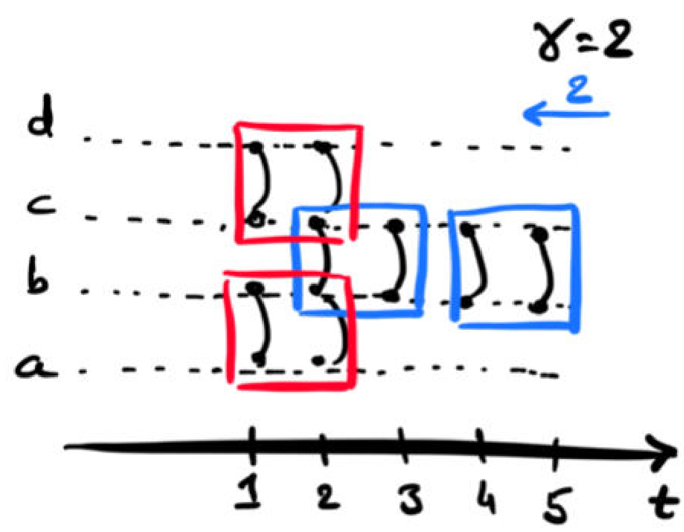
Two γ-edges Γ and Γ' are dependent if:

- $\exists t, u, v, w : (t, uv) \in \Gamma \wedge (t, uw) \in \Gamma'$

TEMPORAL MATCHING: *a γ-matching is a set of independent γ-edges*

Formal definition: γ-matching





Our question: maximum γ-matching?

TEMPORAL MATCHING: *a γ-matching is a set of independent γ-edges*

PROBLEM γ-MATCHING:
INPUT: link stream L, integer k
OUTPUT: boolean stating if there is a γ-matching in L of size k

THEOREM:
- ***NP-completeness for*** $\gamma > 1$ [BASTE, BX.]
- ***NP-completeness for very retricted link streams (with underlying graph being a path)*** [MERTZIOS, MOLTER, NIEDERMEIER, ZAMARAEV, ZSCHOCHE]

Our results:

TEMPORAL MATCHING: *a γ-matching is a set of independent γ-edges*

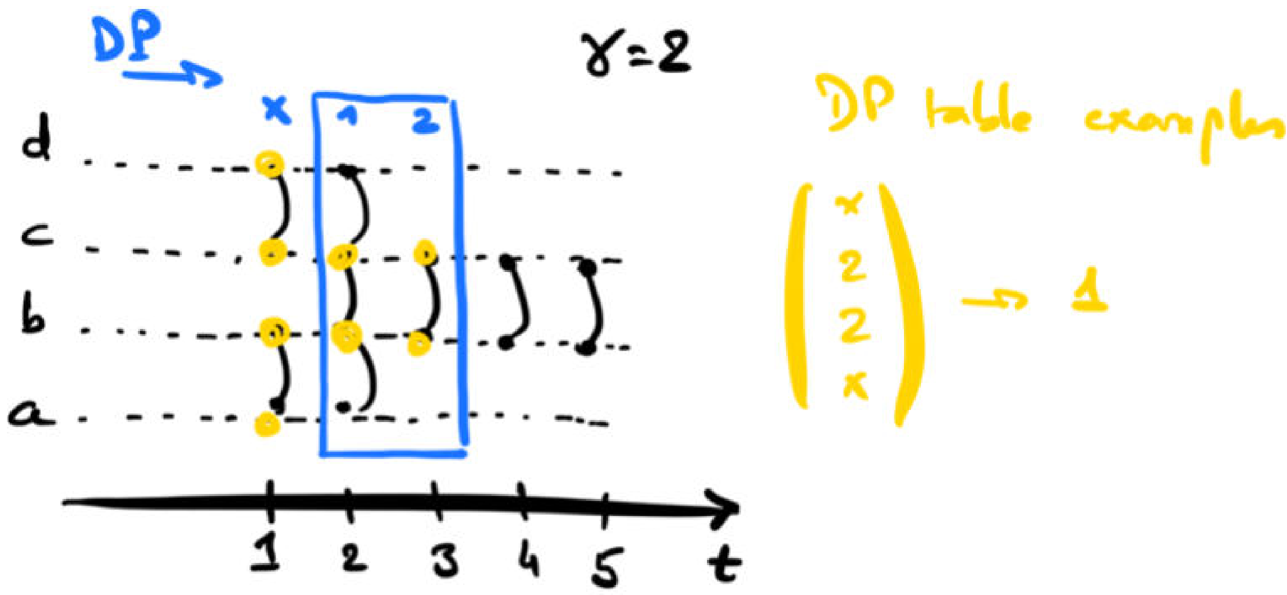
THEOREM [BX., NGUYEN, PICAVET]:

- *γ-matching in time $O^*((\gamma+1)^n)$ by dynamic programming (DP)*
- *PTAS for geometric link streams of bounded velocity and density*

NUMERICAL ANALYSIS:

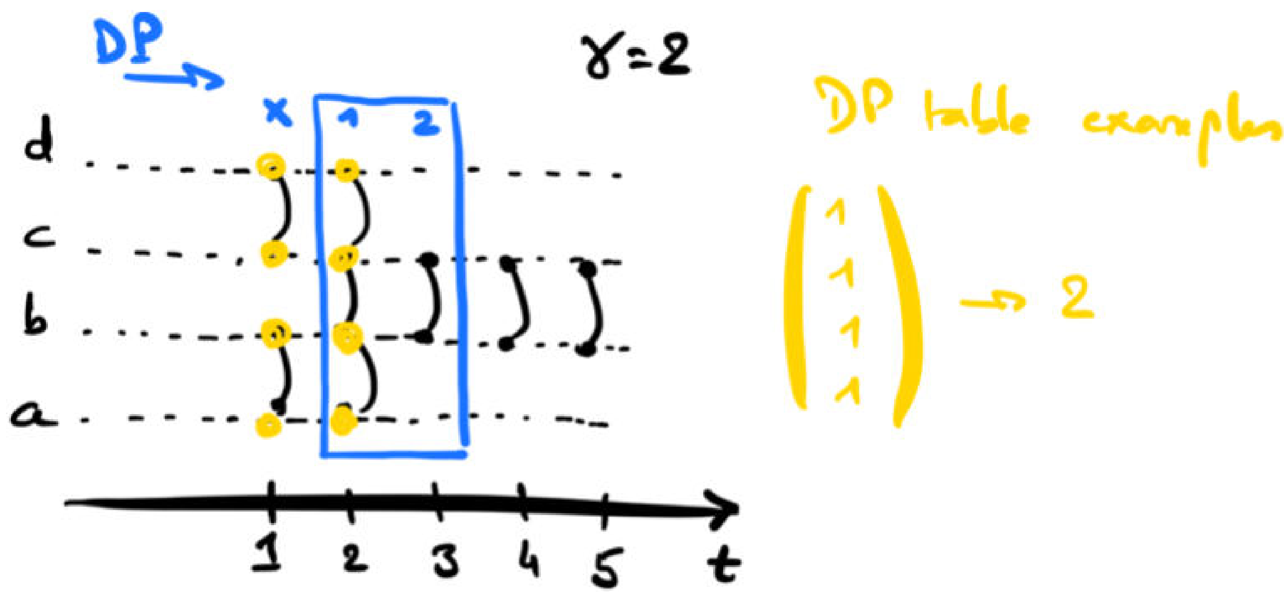
- 2-approximation with greedy [BASTE, BX., ROUX]
  https://github.com/antoinedimitriroux
- DP for general case and PTAS for geometric case
  https://github.com/Talessseed/

DP formula



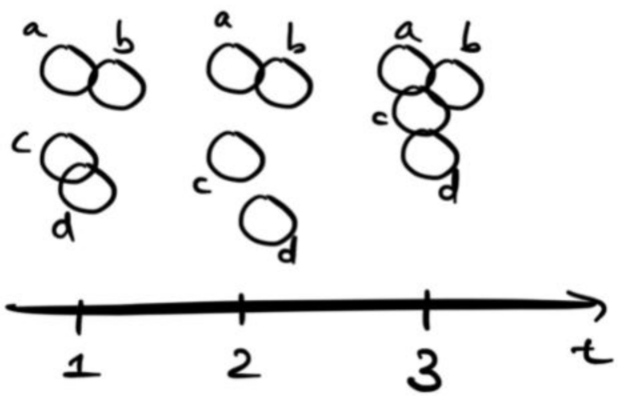
⇒ for every vertex, store the last used position in the sliding window

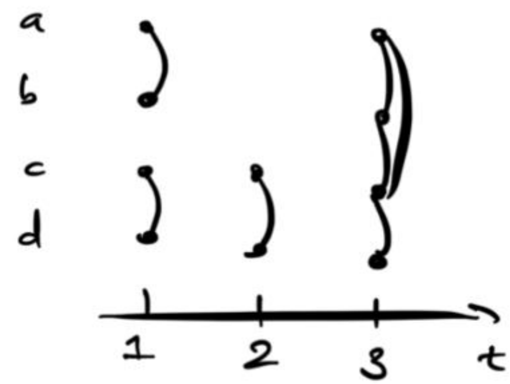
DP formula



⇒ for every vertex, store the last used position in the sliding window

Geometric link streams



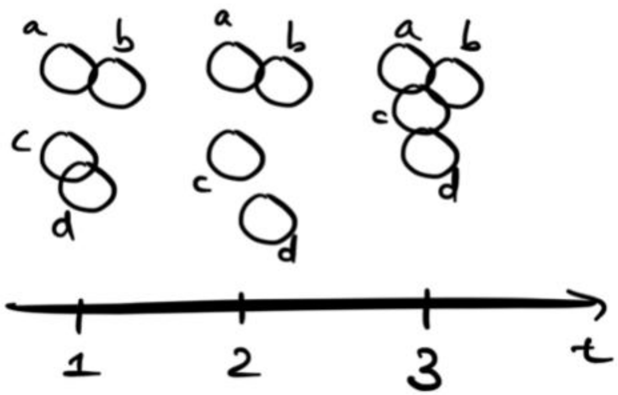


A unit disk graph is:

– the intersection graph of unit disks in the plane

– embedded vertices in the plane; edges exist between vertices of distance at most 1

Geometric link streams



A geometric link stream is s.t.:

– every snapshot at time t is a unit disk graph

– moving (embedded) vertices in the plane; edges exist between vertices of distance at most 1

– velocity ≈ derivative of (embedded) vertex's position

– density ≈ number of (embedded) vertices per square

PTAS for the geometric case

Essential ideas:

– time dimension ≈ sliding window like previous DP

– each snapshot ≈ use linear pathwith DP on stripes of the plane

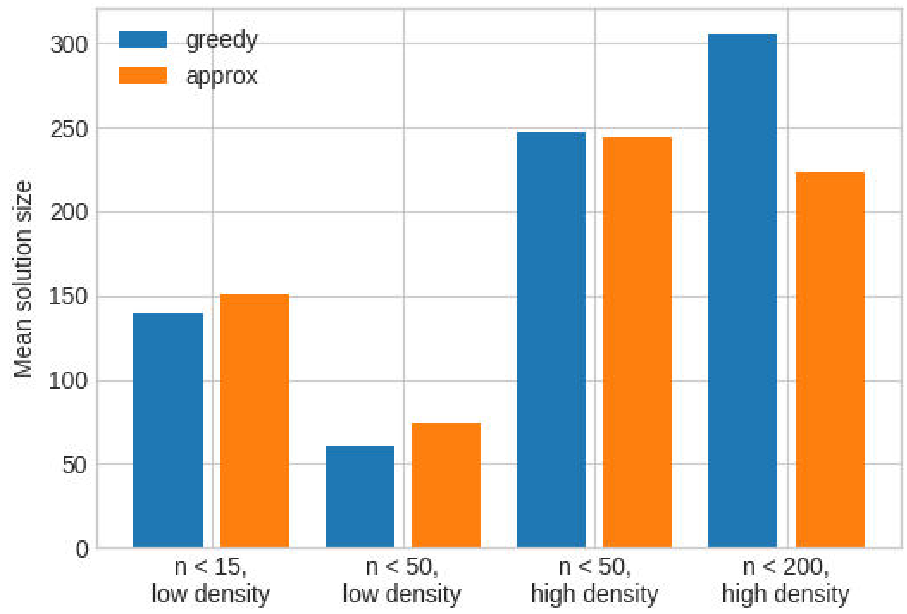
Numerical analysis:

– DP: exact approximation ratio is 1

– PTAS: expected approximation ratio between 1.3 and 1.4

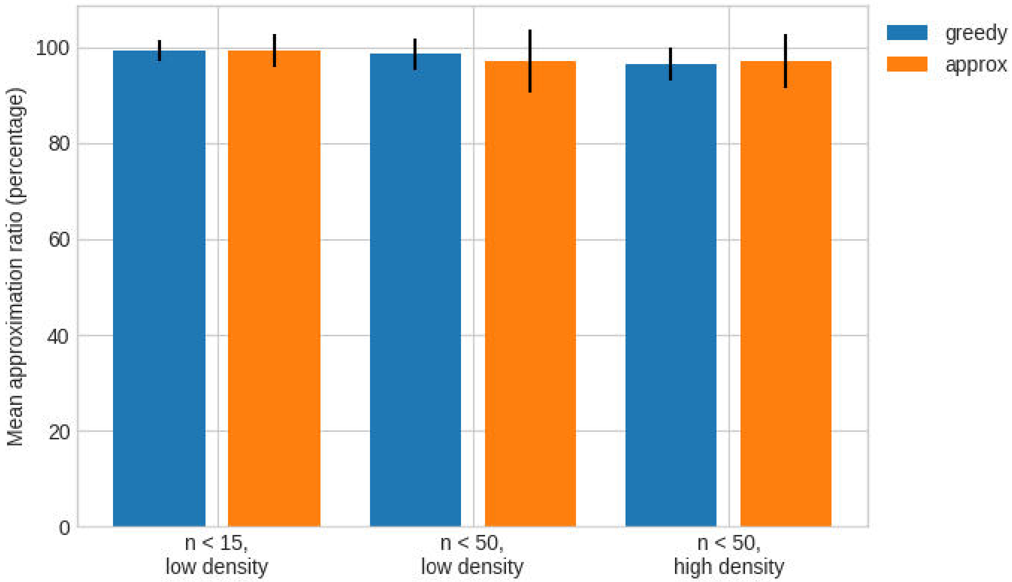
– greedy: expected approximation ratio is 2

PTAS vs. greedy on artificial data



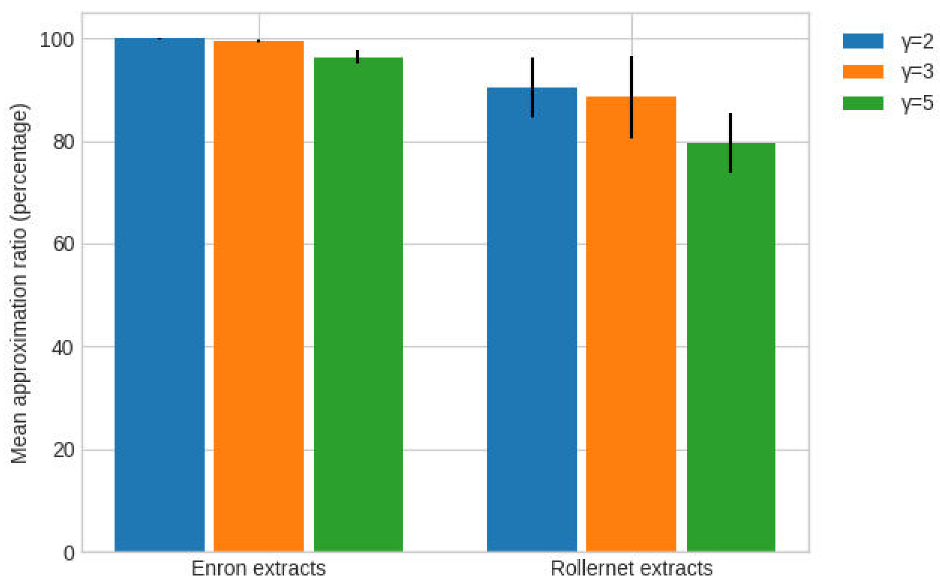
Mean of the outputted size of γ-matchings

Ratio on artificial data (when available)



Mean of the approximation ratio (compared to exact DP)

Ratio on extracts from experiments



Mean of the approximation ratio (compared to exact DP)

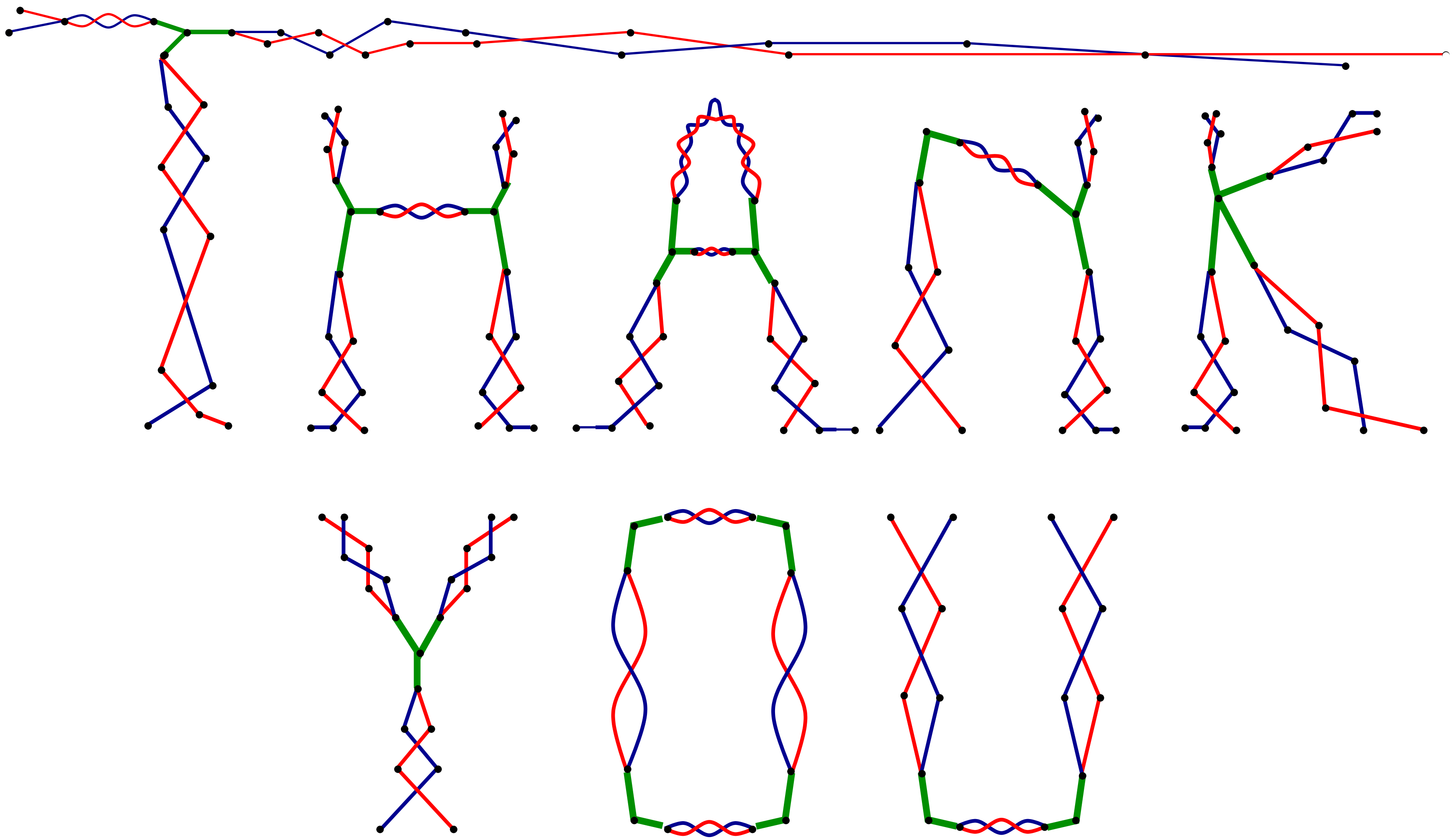
Conclusion and questions

CONCLUSION:

- exact $O^*((\gamma+1)^n)$ dynamic programming
- PTAS for geometric cases
- greedy 2-approximation
- https://github.com/antoinedimitriroux
- https://github.com/Talessseed/

QUESTION:

– *dynamic programming in* $O^*(2^n)$*?*


THANK
YOU