Aalto University
School of Science

The network-untangling problem: From interactions to activity timelines

Polina Rozenshtein (Nordea DS Lab, Finland)

Nikolaj Tatti (University of Helsinki, Finland)

Aristides Gionis (Aalto University, Finland)

ECML/PKDD’17 + journal extension

Temporal networks

- Temporal graph $G = (V, E)$
- V – set of entities (e.g. people, sensors, locations..)

- Edges $(u, v, t) \in E$ – instantaneous interactions over entities
- $u, v \in \mathrm{V}$
- t is the time of interaction
- tweets, emails, comments on social networks..

Problem setting

- consider a set of entities
- entities can become *active* or *inactive*
- entities interact over time, forming a temporal network
- each interaction is attributed to an active entity

Problem setting

- consider a set of entities
- entities can become *active* or *inactive*
- entities interact over time, forming a temporal network
- each interaction is attributed to an active entity

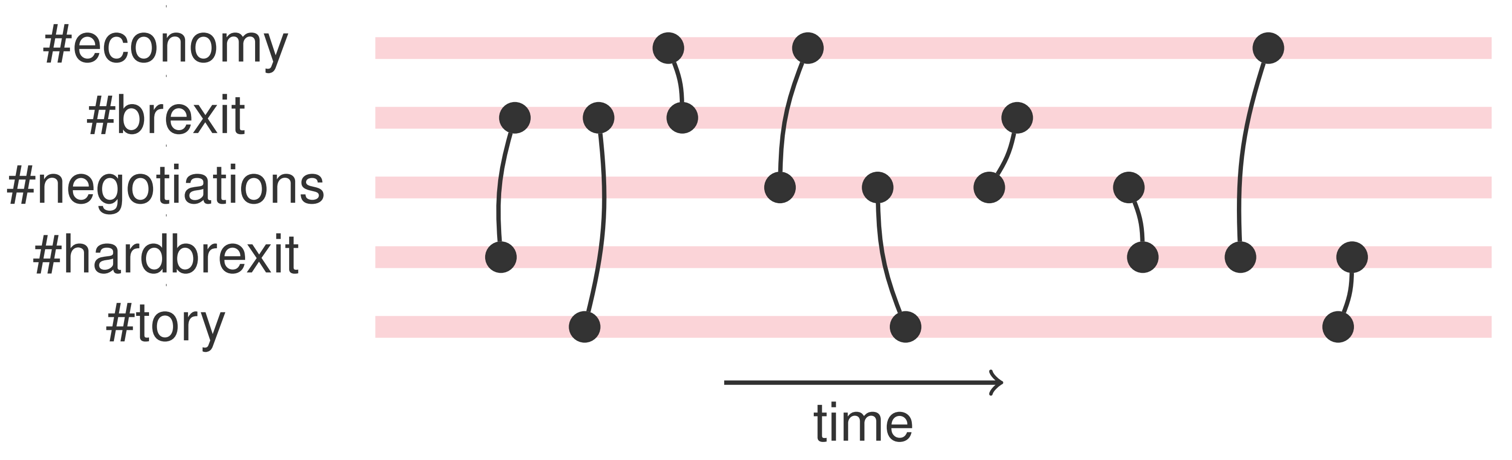
- can we reconstruct the activity timeline that explains best the observed temporal network?

- assumption: being active is more costly, thus we want to minimize total activity time

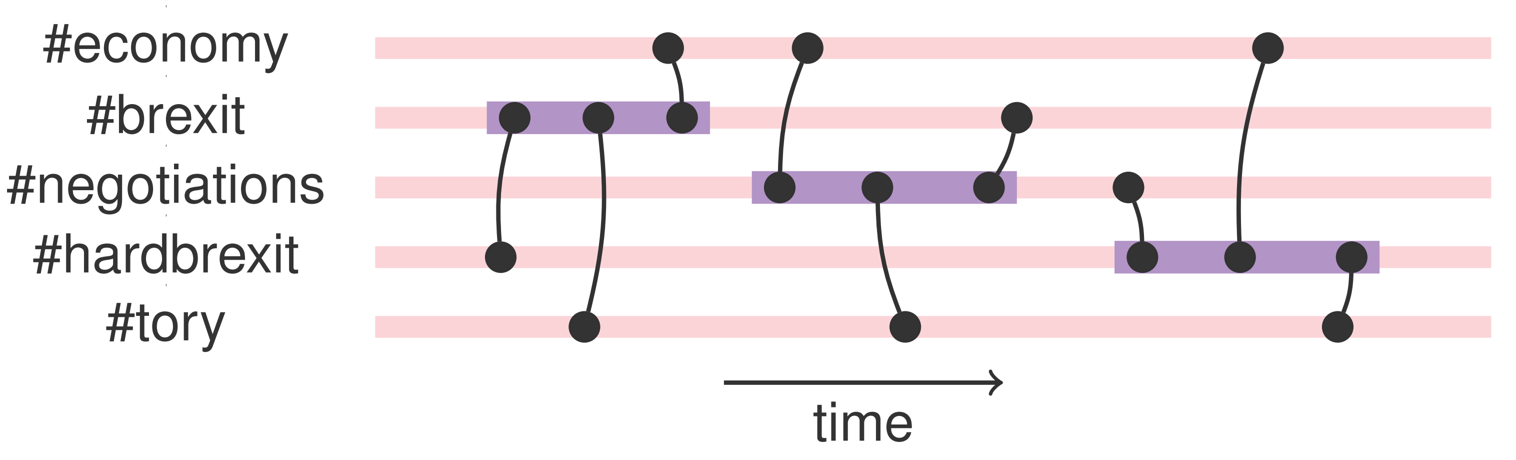
Motivating example

- analyze a discussion in twitter about a topic (e.g., brexit)
- entities are hashtags
- two hashtags interact if they appear in the same tweet
- summarize the discussion by reconstructing a timeline
- pick a set of important hashtags and the time intervals they are active

Motivating example



Motivating example



Aalto University
School of Science

Problem formulation

- given a temporal network $G = (V, E)$ with $E = \{(u, v, t)\}$
- $I_u = [s_u, e_u]$ – activity interval of $u \in V$ (starts at s_u and ends at e_u)
- find a set of activity intervals for all nodes
- at most k per each node $u \in V$

Problem formulation: preliminaries

- given a temporal network $G = (V, E)$ with $E = \{(u, v, t)\}$
- $I_u = [s_u, e_u]$ – activity interval of $u \in V$ (starts at s_u and ends at e_u)
- find a set of activity intervals for all nodes
- at most k per each node $u \in V$

- Activity timeline of G is a set of activity intervals $\mathcal{T} = \{I_{ui}\}_{u \in V, i \in [1,k]}$
- The timeline $\mathcal{T}$ covers temporal network G, if for each edge $(u, v, t) \in E$ we have $t \in I_{ui}$ or $t \in I_{vi}$ for some $i \in [1, k]$.

Problem formulation

Problem 1. (Sum-Span)

- Find a timeline $\mathcal{T} = \{I_{ui}\}_{u \in V, i \in [1,k]}$ that covers G and minimizes total length of $\mathcal{T}$.

Problem 2. (Max-Span)

- Find a timeline $\mathcal{T} = \{I_{ui}\}_{u \in V, i \in [1,k]}$ that covers G and minimizes maximum length of intervals in $\mathcal{T}$.

- For the ease of analysis consider $k = 1$ and $k > 1$ separately

1-Sum-Span

Problem **1-Sum-Span** is NP-hard

Consider subproblem **Coalesce**:

- Assume we are given one active time point m_v for each vertex $v \in V$.
- Find an optimal activity timeline $\mathcal{T}$, which contains the corresponding active time points $\{m_v\}_{v \in V}$.

1-Sum-Span

- **Coalesce** can be solved in linear time with factor 2 approximation, based on Binary LP-formulation.

- Define a variable $x_{vt} \in \{0,1\}$ for each vertex $v \in V$ and time stamp $t \in T(v)$ (moments of interactions of v).
- $x_{vt} = 1$ indicates that t is either the beginning or end of the active interval of v.
- Binary LP:
  - Cost function $\min \sum_{v,t} |t - m_v| \, x_{vt}$
  - Constraints to ensure feasibility

1-Sum-Span

- Relax the integrality and write the dual
- Maximal solution to the dual program is a 2-approximation for **Coalesce**
- Maximal solution can be found in one pass ($O(m)$, Alg. `Maximal`)

Iterate to solve **1-Sum-Span** (Alg. `Inner`):

- Start with $m_v = (min\, T(v) + max\, T(v))/2$
- Run `Maximal` and update m_v
- Repeat until no improvement.

k-Sum-Span

k-Sum-Span is are inapproximable

Consider subproblem **k-Coalesce**:

- Assume we are also given k active time points m_{vi} for each vertex $v \in V$
- One for each of activity intervals of v
- Find an optimal activity timeline $\mathcal{T}$, which contains the corresponding active time points $\{m_{vi}\}_{v \in V, i \in [1,k]}$.
- Similar BLP and Alg. `k-Maximal`, $O(m)$

k-Sum-Span

Iterate to solve **k-Sum-Span** (Alg. `k-Inner`):

- Start with m_{vj} as centroids of a k-clustering algorithm
- Run `k-Maximal` and update m_v
- Repeat until no improvement

1-Max-Span

1-Max-Span can be solved efficiently

Subproblem **Budget**:

- Assume we are also given a set of budgets $\{b_v\}_{v \in V}$ of interval durations for each vertex.
- Find an optimal activity timeline $\mathcal{T} = \{I_v\}_{v \in V}$, such that length of each activity interval I_v is at most b_v.

1-Max-Span

Budget can be solved optimally in linear time

Map **Budget** into 2-SAT:

- Variable x_{vt} for each vertex v and timestamp $t \in T(v)$.
- Clause $(x_{vt} \vee x_{ut})$ – cover each edge (u, v, t).
- Clause $(\overline{x_{vs}} \vee \overline{x_{vt}})$ – ensure budget:
  for each $s, t \in T(v)$, such that $|s - t| > b_v$

- Solution for **Budget** : time intervals where all boolean variables are **`True`**.

1-Max-Span

Linear time:

- 2-SAT is solved in linear-time of the number of clauses (Aspvall et all [1]). We have $O(m^2)$ clauses.
- Bottleneck: SCC decomposition $O(m^2 + m)$
- algorithm by Kosaraju [2] for SCC decomposition
- Use of temporal structure → perform DFS in $O(m)$.

Solve **1-Max-Span** by binary search to find the optimal maximum length for intervals (Algorithm `Budget`, $O(m \log(m))$).

k-Max-Span

k-Max-Span inapproximable

- consider two nested subproblems

Subproblem **k-Partition**:

- Assume we are also given k-1 **inactive** time points g_{vi} for each vertex $v \in V$
- One for each of gap between the activity intervals of v
- Find an optimal activity timeline $\mathcal{T}$, which interleaves with corresponding gap points $\{g_{vj}\}_{v \in V, j=[1,k-1]}$

k-Max-Span

- Problem **k-Partition** can be solved in polynomial time through iteration of Problem k-Budget, which sets a budget for each interval.

Subproblem **k-Budget**:

- Assume we are given a set of budgets $\{b_v\}_{v \in V}$ of interval durations for each vertex;
- k-1 **inactive** time points g_{vj} for each vertex
- Find an optimal activity timeline $\mathcal{T} = \{I_v\}_{v \in V}$, such that length of each activity interval I_{vj} is at most b_{vj} and the gap points are interleaved

k-Budget can be solved $O(m)$, similarly to **Budget**

k-Max-Span

Iterate to solve **k-Sum-Span** (Alg. `k-Budget`):

- Start with g_{vj} as mean points of the largest intervals with no activity of node v
- Solve **k-Partition**:
  - do binary search on budgets with solving **k-Budget**
  - update g_{vj}
- Repeat until no improvement

Summary

Problem 1: Sum-Span

- $k = 1$ NP-hard
- $k > 1$ inapproximable
- Subproblem **(k-)Partition** with inner points
- 2-approximation in linear time via BLP dual for **(k-)Partition**

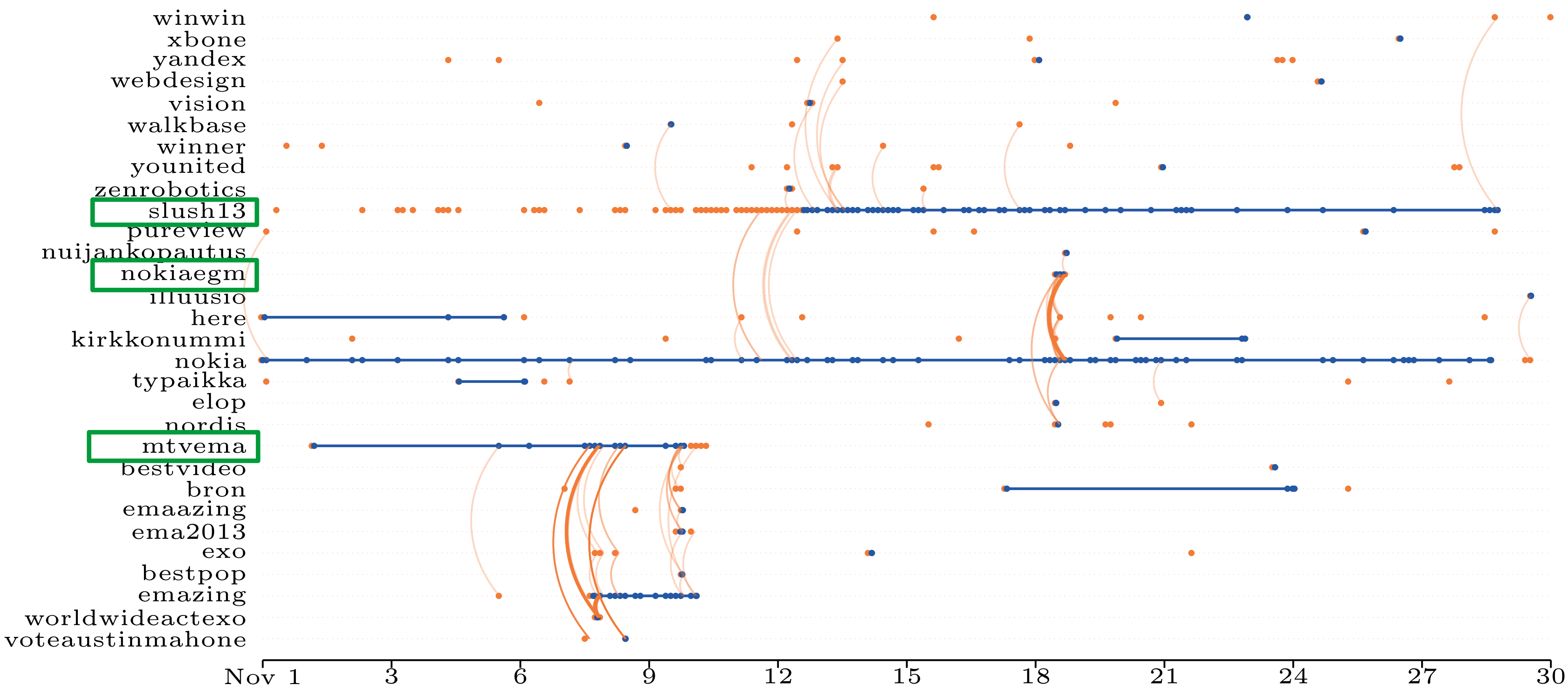
Summary

Problem 2: Max-Span

- $k = 1$ polynomially solvable
- $k > 1$ inapproximable
- Subproblem **(k-)Budget** with budgets
- Exact solution in linear time via 2-SAT for **(k-)Budget**

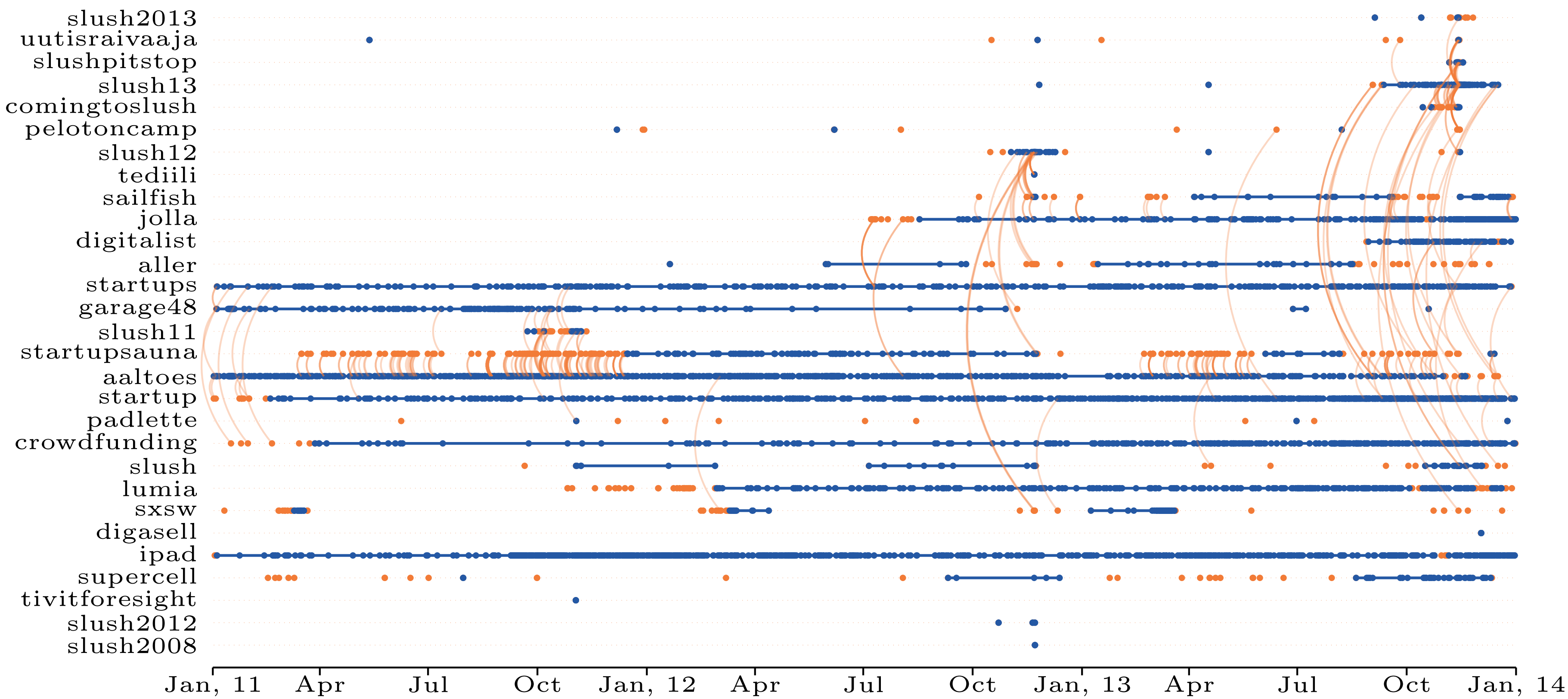
Experiments: case study

- Tweets from Helsinki region, November 2013
- **`Inner`** algorithm (**1-Sum-Span**)



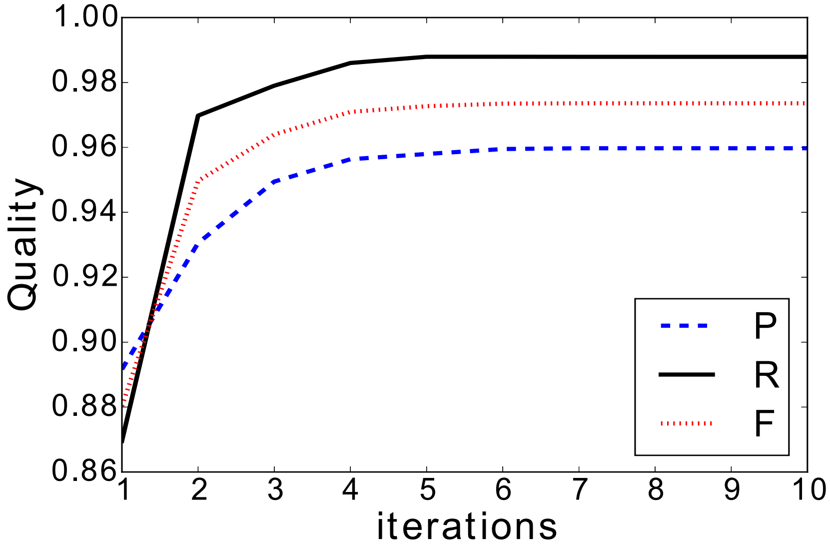
Experiments: case study

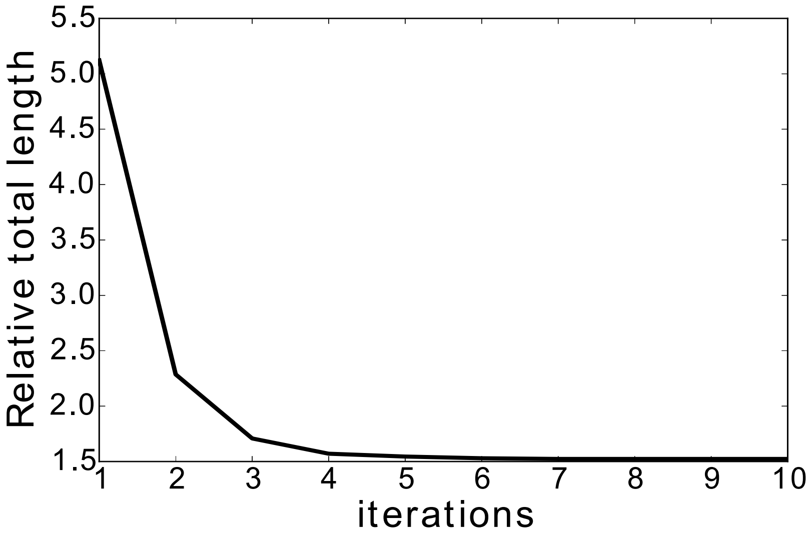
- Helsinki Twitter, years 2011-2013
- **`k-Inner`** algorithm with k = 3 (**k-Max-S**)



Performance: `Inner`

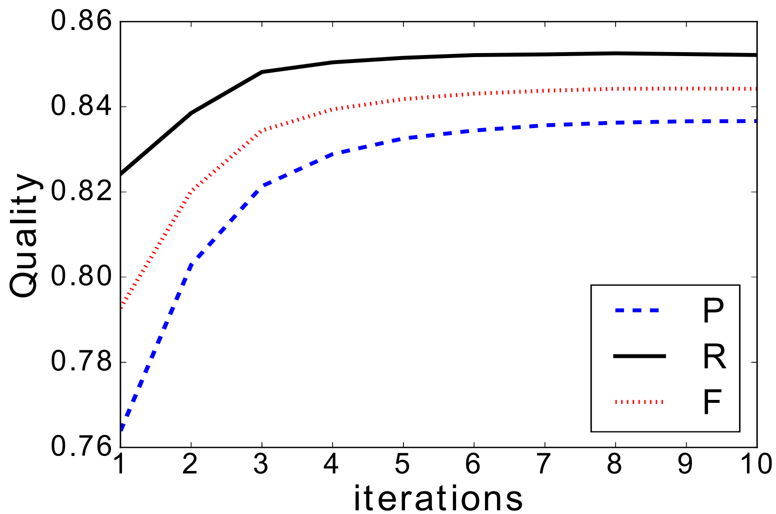
- Synthetic dataset, with planted ground truth
- overlap p is set to 0.5
- values are averaged over 100 runs.

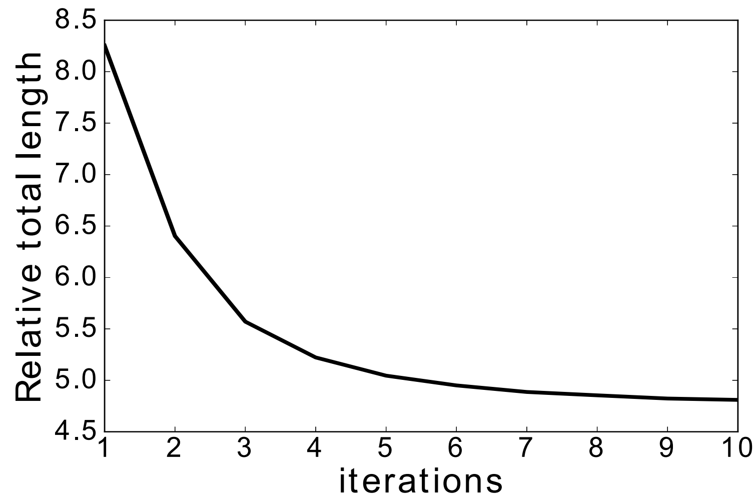




Performance : `k-Inner`

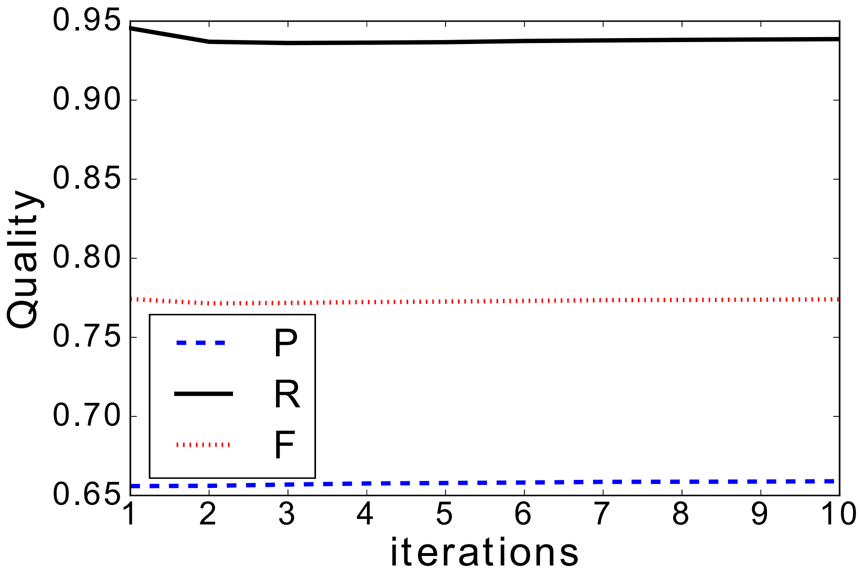
- Synthetic dataset, k=10 intervals

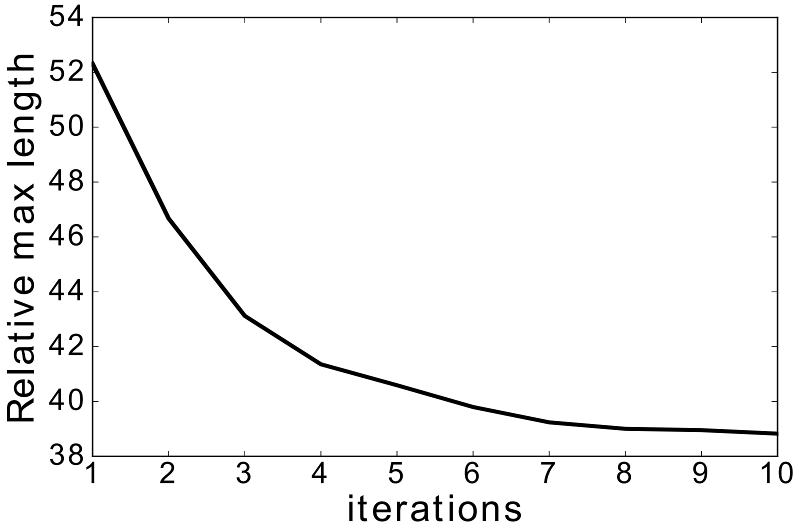




Performance : `k-Budget`

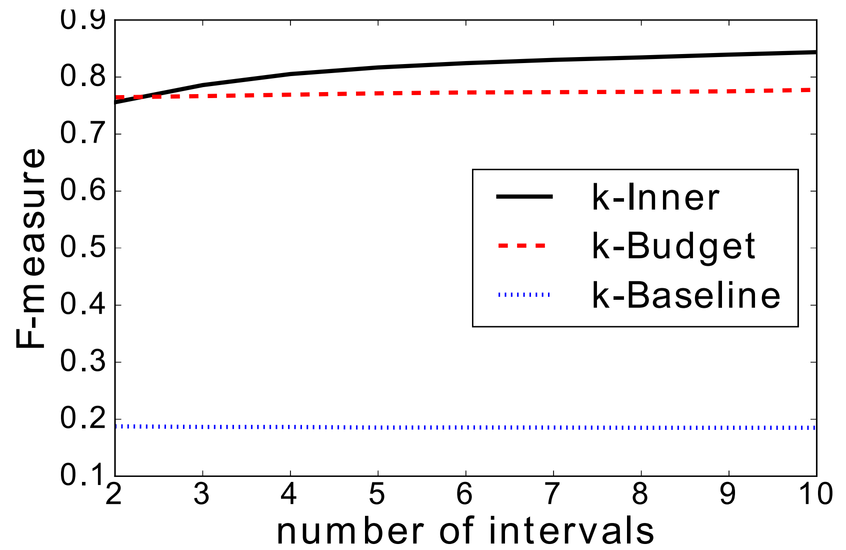
- Synthetic dataset, k=10 intervals

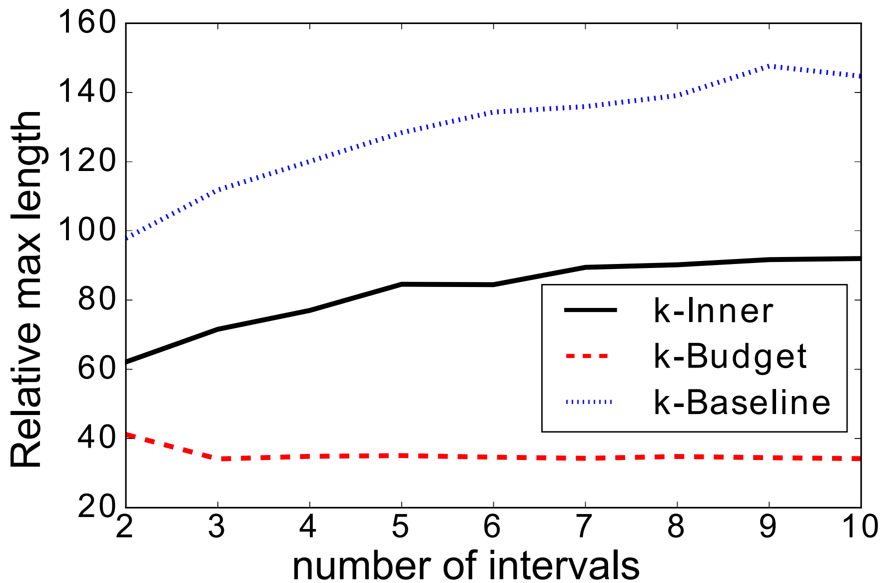


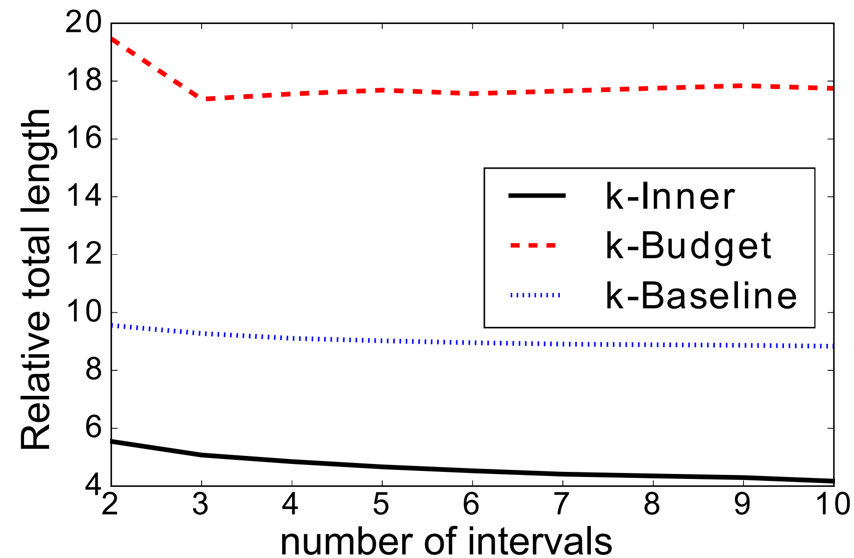


Baseline comparison

- Baseline: greedily ’cover’ the longest activity intervals of the nodes.

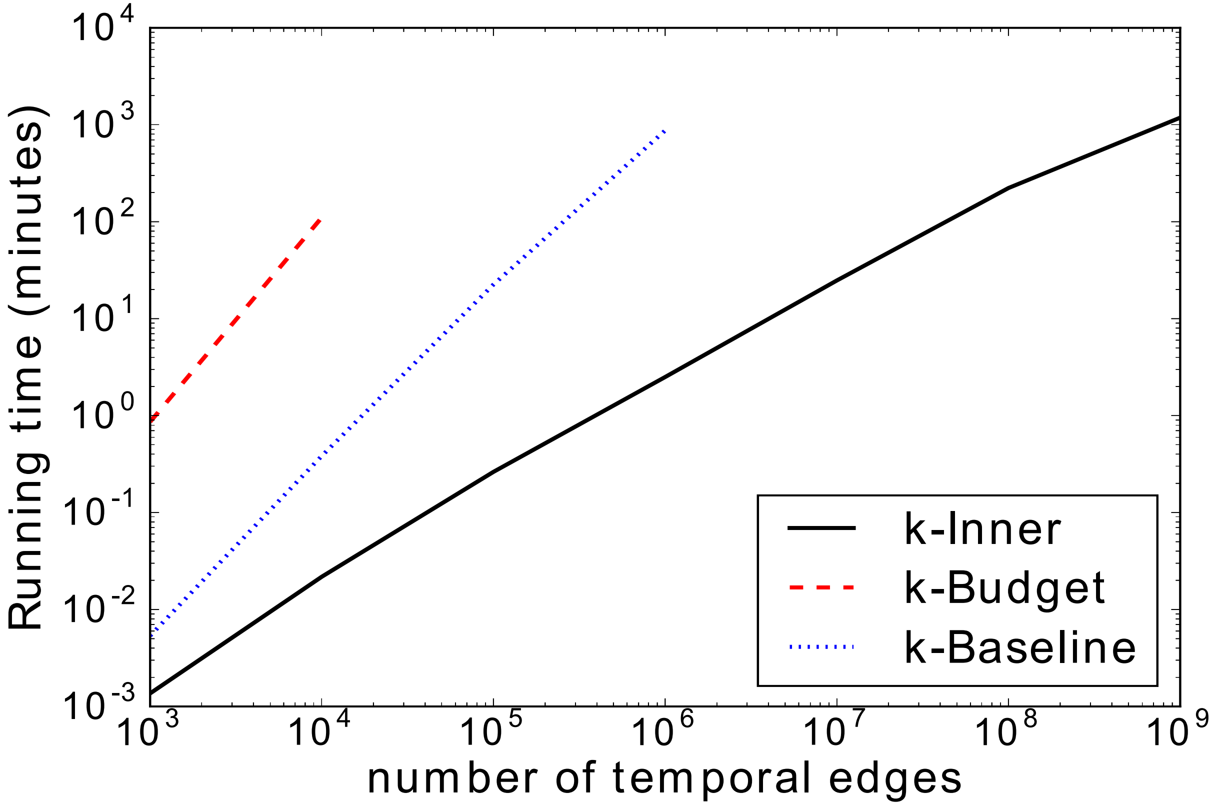






Running time

- k=10, synthetic dataset



Conclusions

- Novel problem of network untangling:

Discover activity time intervals for the network entities to explain the observed interactions.

- A possible Temporal extension of Vertex Cover Problem

- Two settings: **(k-)Sum-Span** (minimize sum of interval lengths) and **(k-)Max-Span** (minimize maximum length).

- Some hardness and inapproximability results
- Efficient algorithms

Future work

- Approximation for **1-Sum-Span**?
- Consider different activity levels for each entity.
- Consider hyperedges.

References

1. B. Aspvall, M. F. Plass, and R. E. Tarjan. A linear-time algorithm for testing the truth of certain quantified Boolean formulas. 1982.

2. J. E. Hopcroft and J. D. Ullman. Data structures and algorithms.1983.

Thank you!