Statistically significant graph evolution rules in temporal networks

Alessia Galdeman, Matteo Zignani,
And Sabrina Gaito

COMPLEX NETWORKS

Connets LAB
Computer Science
UNIVERSITY OF MILAN

Introduction

RESEARCH GROUP WORKS

Network evolution

Graph evolution rules
Change point/Anomaly detection

Graph Machine Learning

Social network analysis using GNN and LLM
Temporal Graph Learning for heterogeneous networks

User behaviour

Multilayer community detection
Influence of hubs in a user migration context
User strategies in a reward-based platform

Web3 platform behavioral and network analysis

Blackchain-based online social network

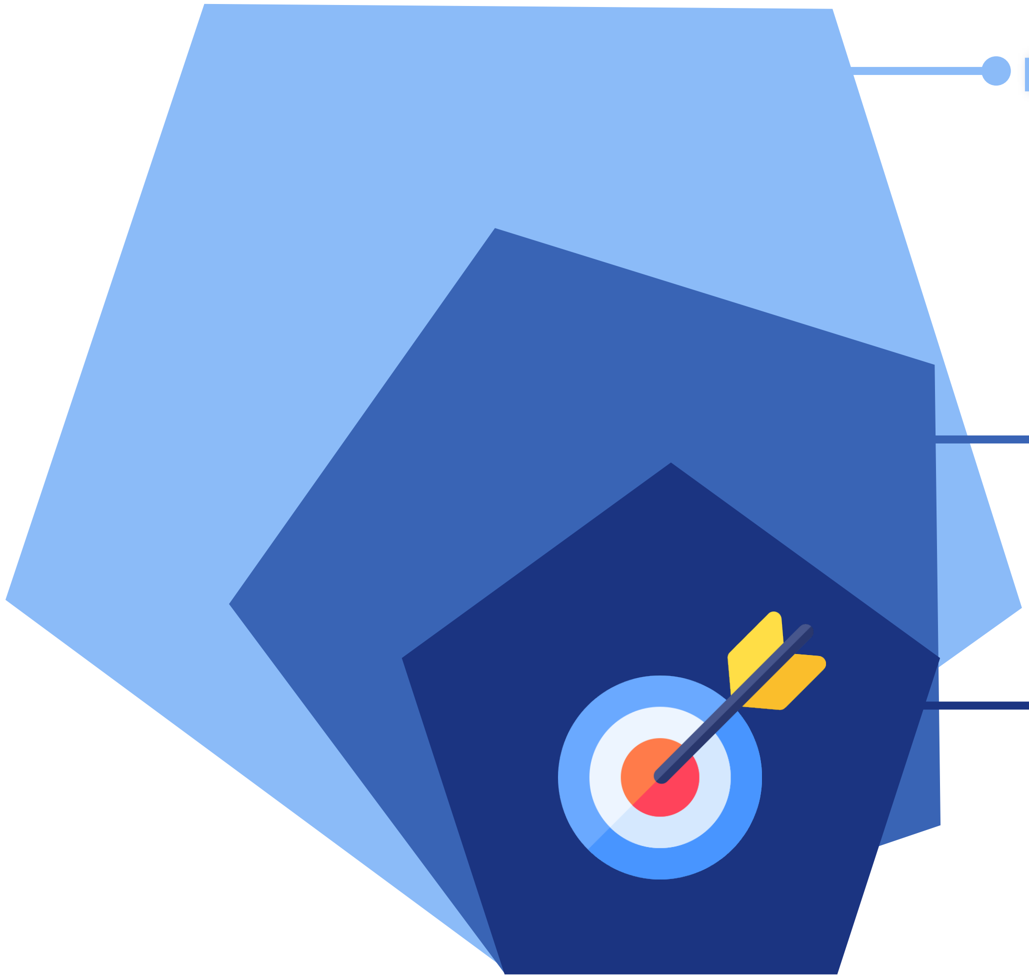
NFT networks

Cryptocurrency networks (Luna, Steem, Ethereum, Sarafu)

To see our works visit

https://connets.di.unimi.it/

The goal



NETWORK EVOLUTION

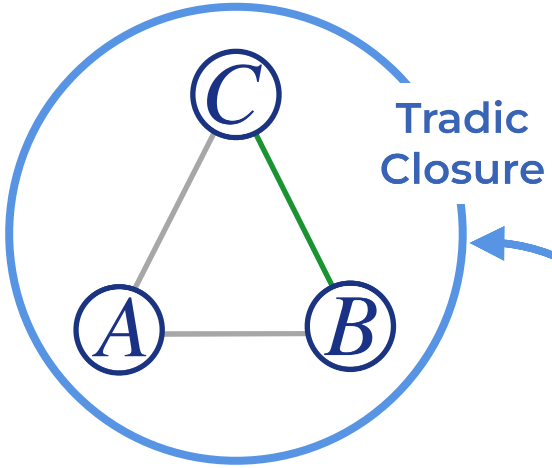
GRAPH EVOLUTION RULES

STATISTICALLY SIGNIFICANT GRAPH EVOLUTION RULES

GRAPH
EVOLUTION RULES

Background

MICROCANONICAL
RANDOMIZED
REFERENCE MODELS



Background

GRAPH EVOLUTION RULES (GER)

Several models, mechanisms and measures have been proposed to describe the network growth

BUT

- They assume that the growth is guided by a single parameterized mechanism
- Identifying which mechanism plays a more important role is challenging

Graph evolution rules mining can detect evolutionary behaviors, while avoiding any a-priori mechanism

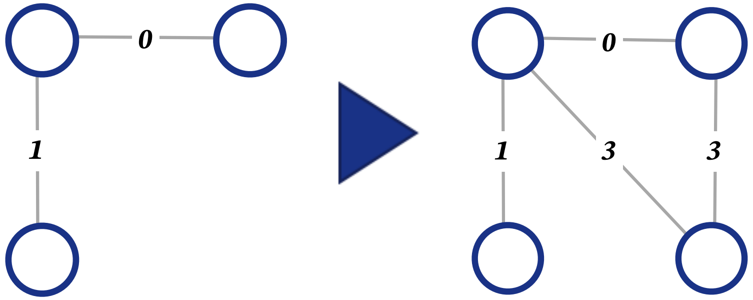
Graph evolution rules

Precondition

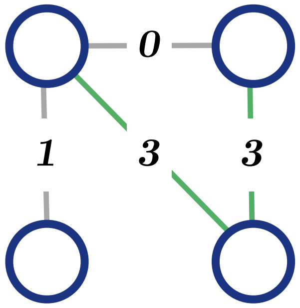
Body

Postcondition

Head



Alternative visualization:

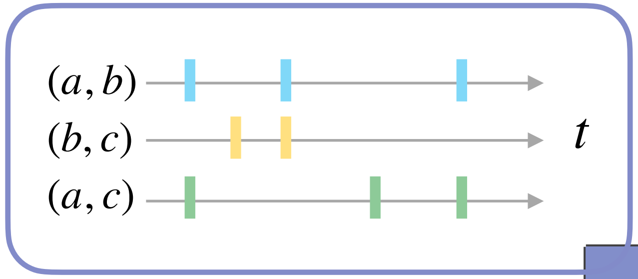


- Similar to association rules in data mining, a GER consists of a precondition (referred to as the *body*) and a postcondition (referred to as the *head)*
- *A* subgraph matching the body is likely to evolve into one matching the head

Explore structural evolutionary properties

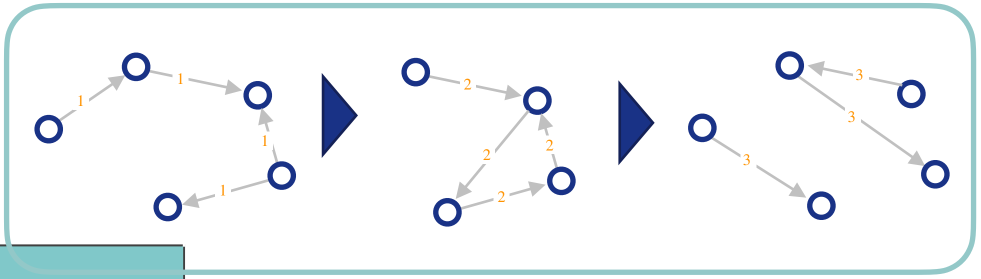
Lack of a null model to measure the significance of rules

Microcanonical Randomized Reference Models[1]



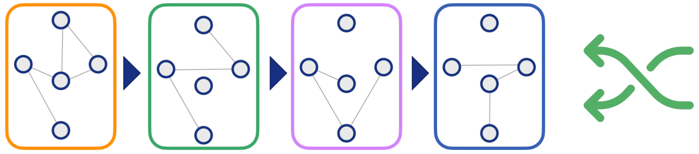
MRRM

Graph representation

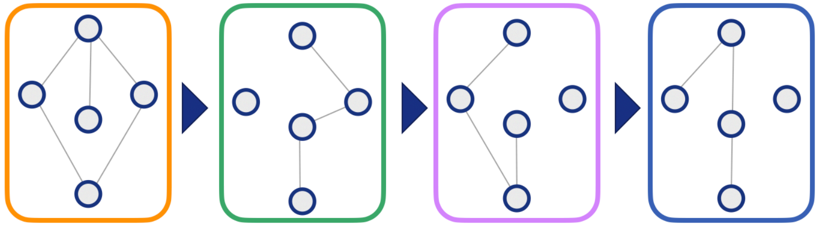


MRMMs categories		Timeline representation	Snapshot representation
What's preserving	Topology	*Timeline shuffling*	*Sequence shuffling*
	Temporal distribution	*Link shuffling*	*Snapshot shuffling*

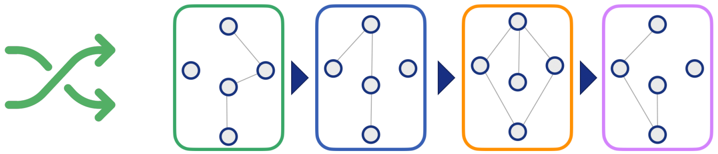
Preserving Temp. Distribution



Original graph



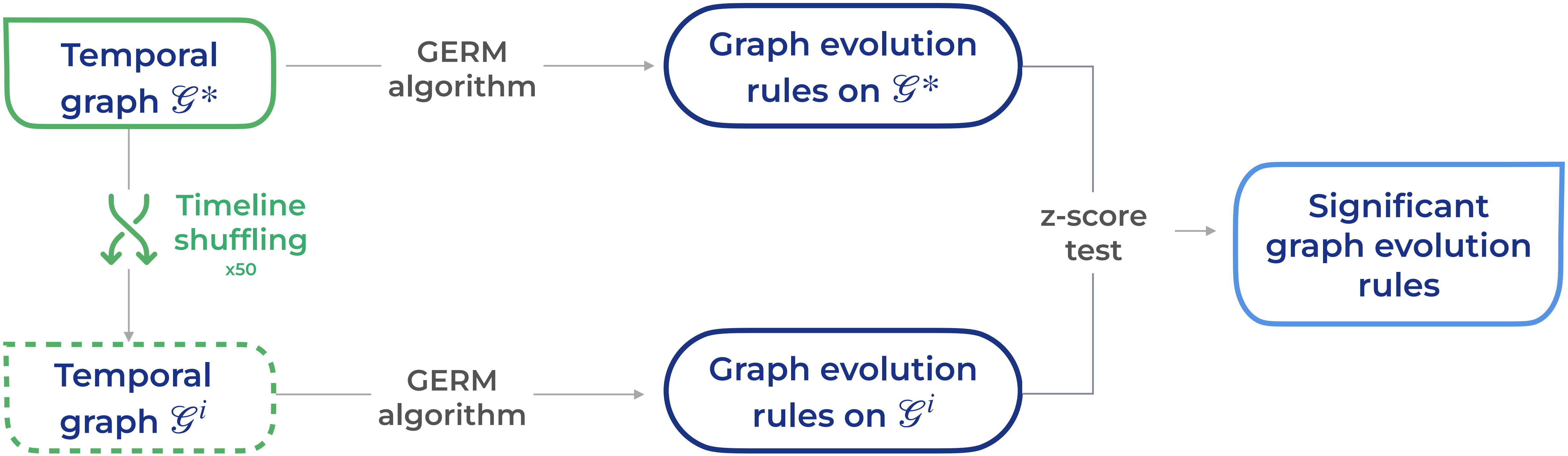
Preserving Topology



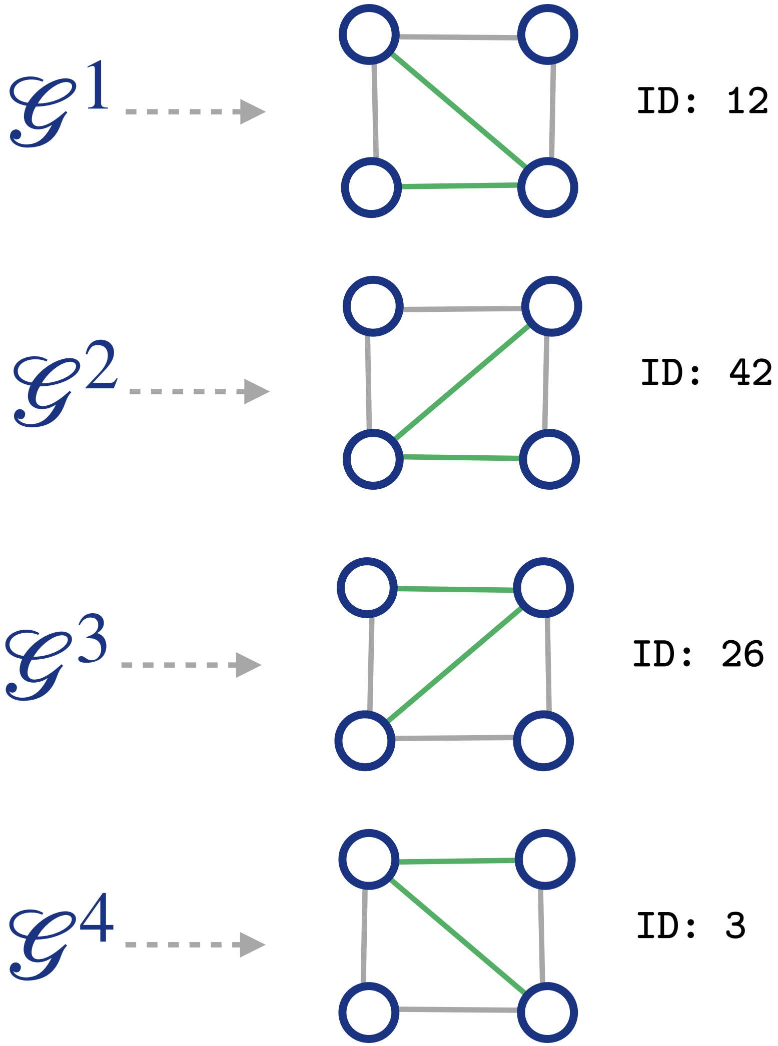
[1] Gauvin, Laetitia, et al. "Randomized reference models for temporal networks." *SIAM Review* 64.4 (2022): 763-830.

Methodology

PIPELINE



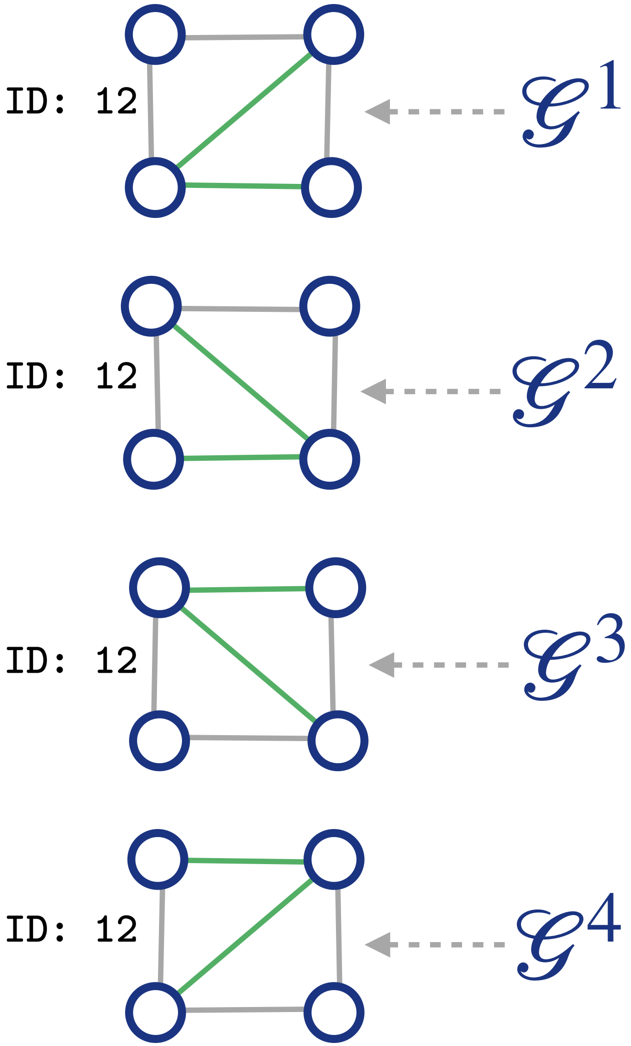
General mapping



The application of GERM on each realization generates patterns with their respective edge lists and supports, identified by independent incremental IDs.

This can create isomorphic rules with different IDs, making impossible the comparis among different random realizations

Create a **general mapping**, starting from the set of edge list (rules are consistent in number among realizations, so there's a change they're equals), and then checking the isomorphism for each realization



Datasets

Each operation is a tuple
(u, v, t)
That records a relationship (co-authorship or online interaction) between users u and v at timestamp t

DBLP

$|E| = 277081$

$|N| = 129073$

11-years period

UC-SOCIAL

$|E| = 12916$

$|N| = 1280$

Timestamps (originally in microseconds) are aggregated at different granularities

UC-WEEKLY

28 timestamps

UC-MONTHLY

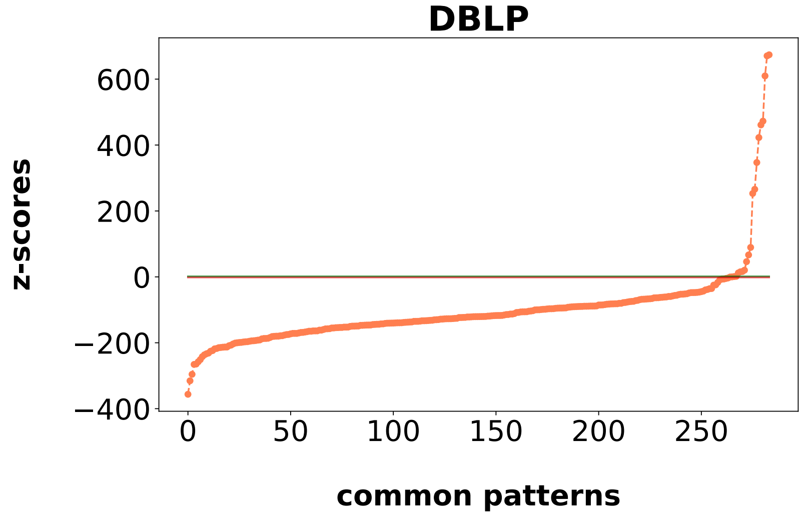
7 timestamps

Findings

Outcomes on real and randomize graphs

		Graph representation		
	Number of frequent rules	$\mathscr{G}*$	$\overline{\mathscr{G}^i}$ Mean shuffle	$\bigcup \mathscr{G}^i$ Union shuffle
Graphs	DBLP	*296*	*3795*	*3871*
	UC-monthly	*266*	*1269*	*1378*
	UC-weekly	*999*	*1039*	*1235*

The rules obtained in the 50 realizations of the null model is stable as the mean and union shuffle count are similar while being very distant from the real graph count

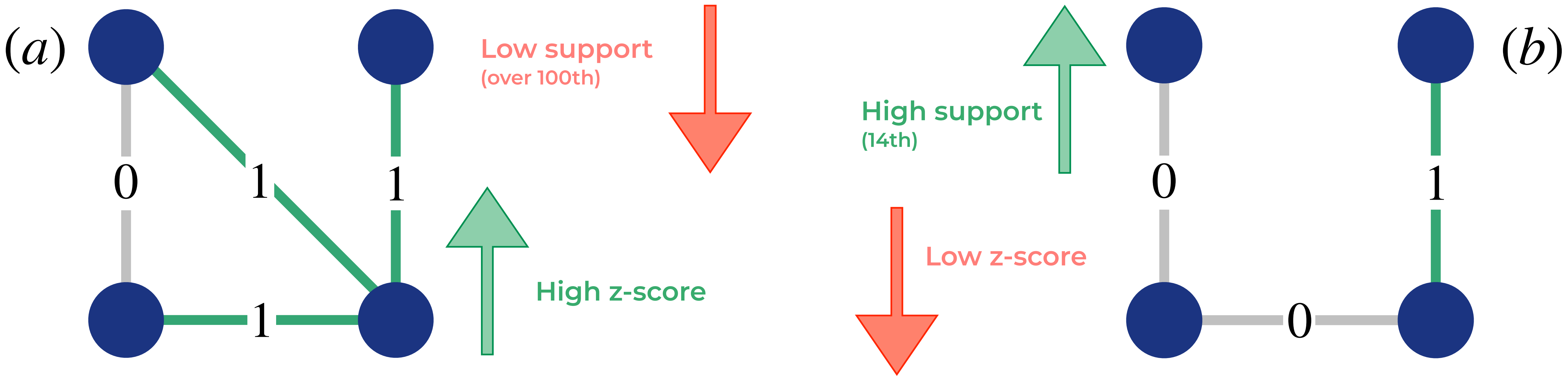


16
over-represented (significant) rules

264
under-represented rules

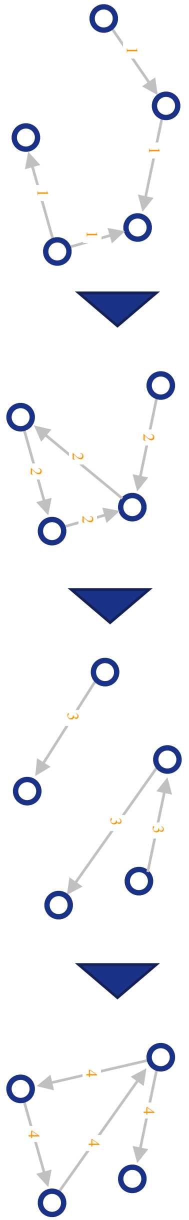
Discussion

WHAT IF WE DO NOT IMPLEMENT THE NULL MODEL?



Rule apparently ***NOT*** important but actually ***worthy*** of attention

Rule apparently ***important*** but actually ***NOT worthy*** of attention



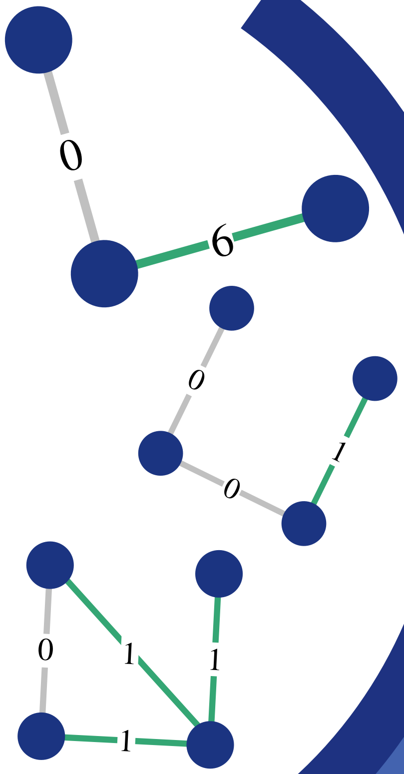
Conclusions

1. Extract GERs from temporal graph
2. Extract GERs from null model realizations
3. Obtain statistically significant rules
4. Comprehensive analysis of results

We saw that:

- By shuffling timestamps, we weaken some temporal constraints on the original graph
- Some rules are frequent but not significative and viceversa

Future and ongoing works



Factors of under/over representation

Identifying the mechanisms and the factors acting on the over and under-representation of the GERs

Extension to other GER algorithms

Apply the same methodology to other state-of-the-art GER algorithms

Python toolkit for GER easy usage: GERANIO

Recently released a python toolkit that helps applying, drawing, and analysing graph evolution rules.

GERANIO

Graph Evolution Rules ANalytics vIsualization tOolkit

GitHub

Alessia Galdeman

Phd Student

alessia.galdeman@unimi.it

@AlessiaGaldeman

Alessia Galdeman

https://alessiaatunimi.github.io/

Thanks for your attention

Research Lab website:

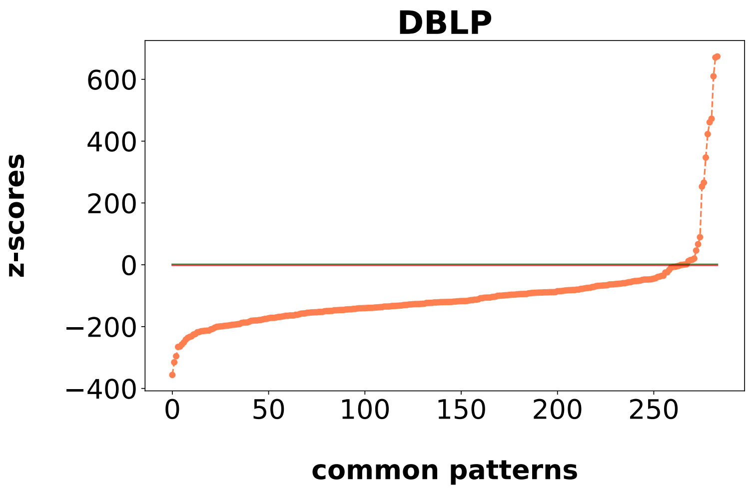
https://connets.di.unimi.it/

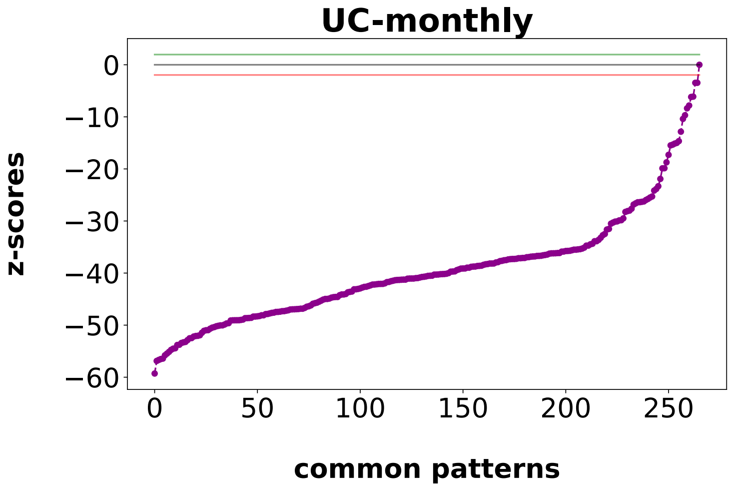
This work is a short version of the paper

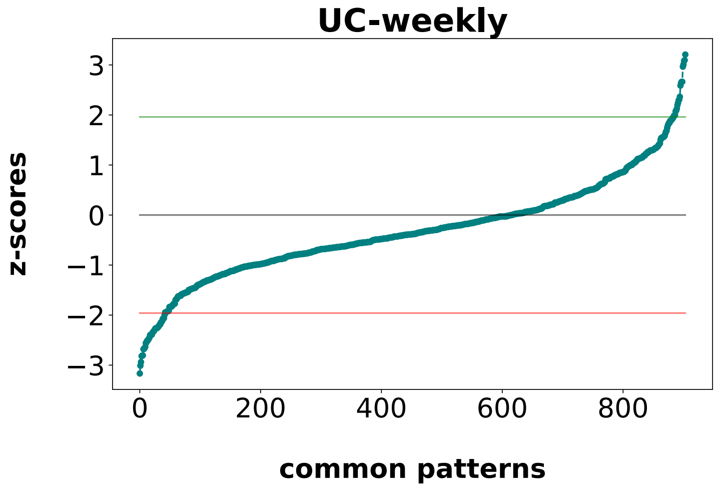
"Unfolding temporal networks through statistically significant graph evolution rules"

accepted at DSAA2023

z-score results







General mapping

Algorithm 2 General mapping: set of edges

Input: results of GERM on each realization of the null model $germ$

Output: $edges\_set$

$edges\_set = dict()$
$m = 0$
for $model \in germ$ **do**
  for $p, pattern \in model$ **do**
    push($edges\_set[pattern_{edges}]$, (p, m))
  end for
  $m \leftarrow m + 1$
end for

Algorithm 3 General mapping: isomorphism check

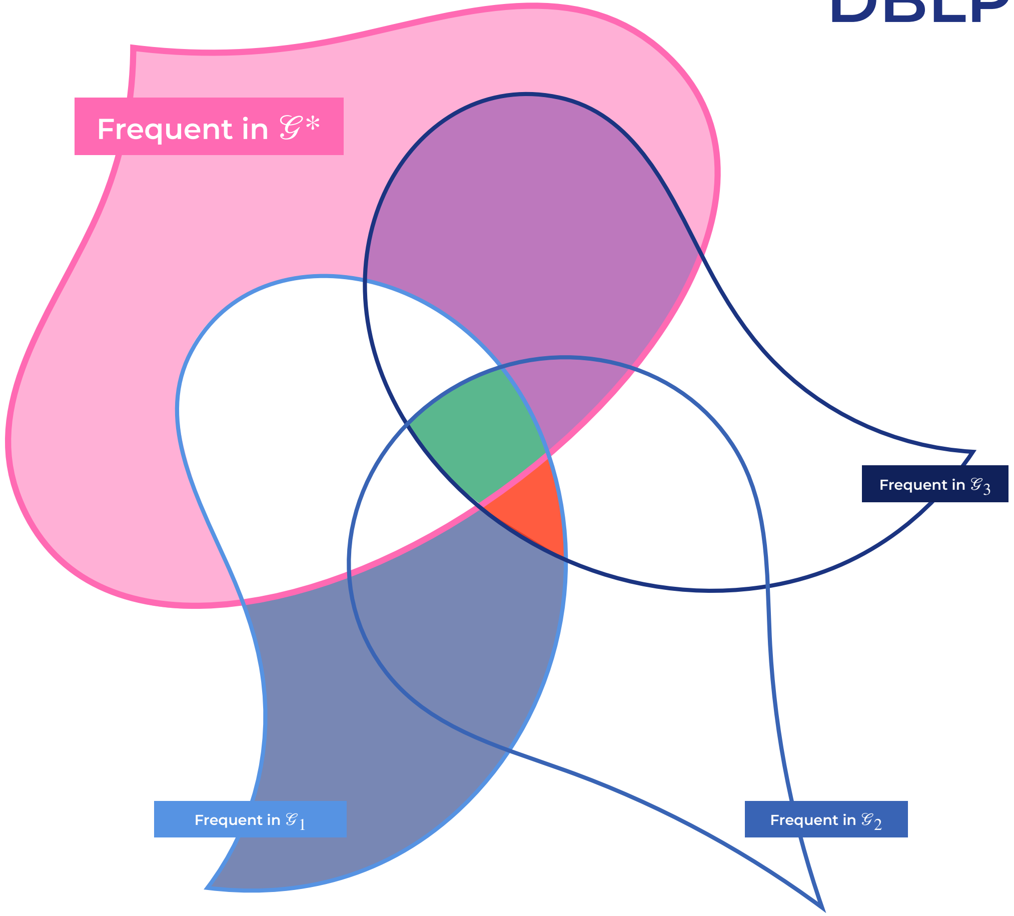
Input: $edges\_set$

Output: $new\_shuffle\_germ$

$mapping = dict()$
$new\_shuffle\_germ = dict()$ ▷ Inspired by Python, dict() creates an associative map
$i = max(mapping.keys)$
for $edges, occurrences \in edges\_set$ **do**
  $id \leftarrow 0$
  for $p, pattern \in mapping$ **do**
    if $G(edges)$ $is\_isomorphic$ $G(pattern)$ **then**
      $id \leftarrow p$
      $continue$
    end if
  end for
  if $id = 0$ **then**
    $i \leftarrow i + 1$
    $id \leftarrow i$
    $mapping[i] = G(edges)$
  end if
  for $p, m \in occurrences$ **do**
    $new\_shuffle\_germ[m][id] = germ[m][p]$
  end for
end for

Other set of rules to consider

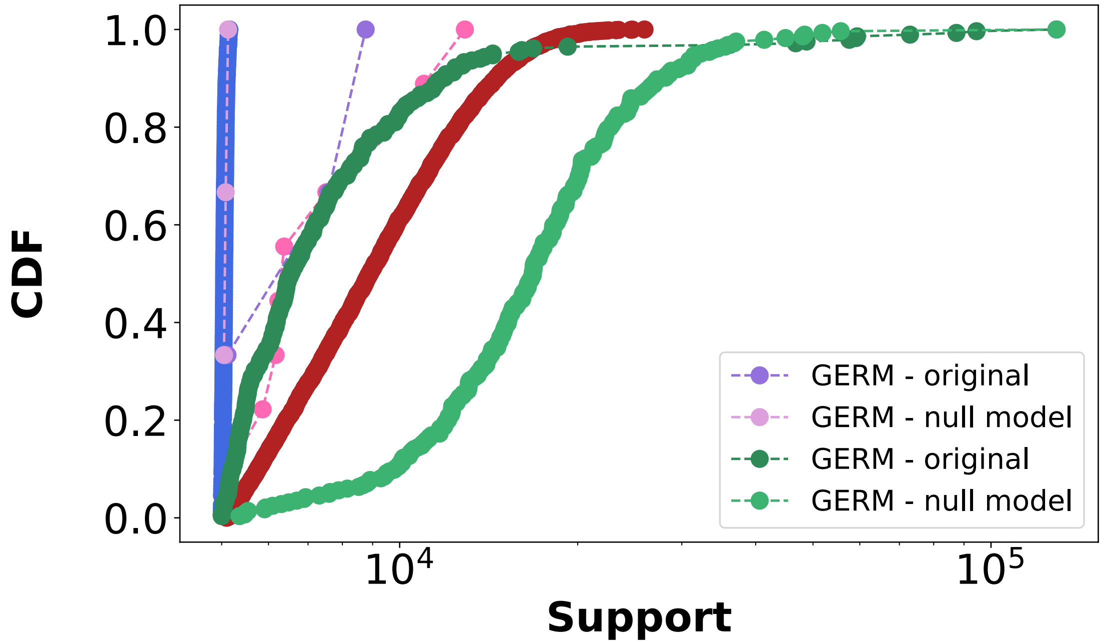
DBLP CASE STUDY

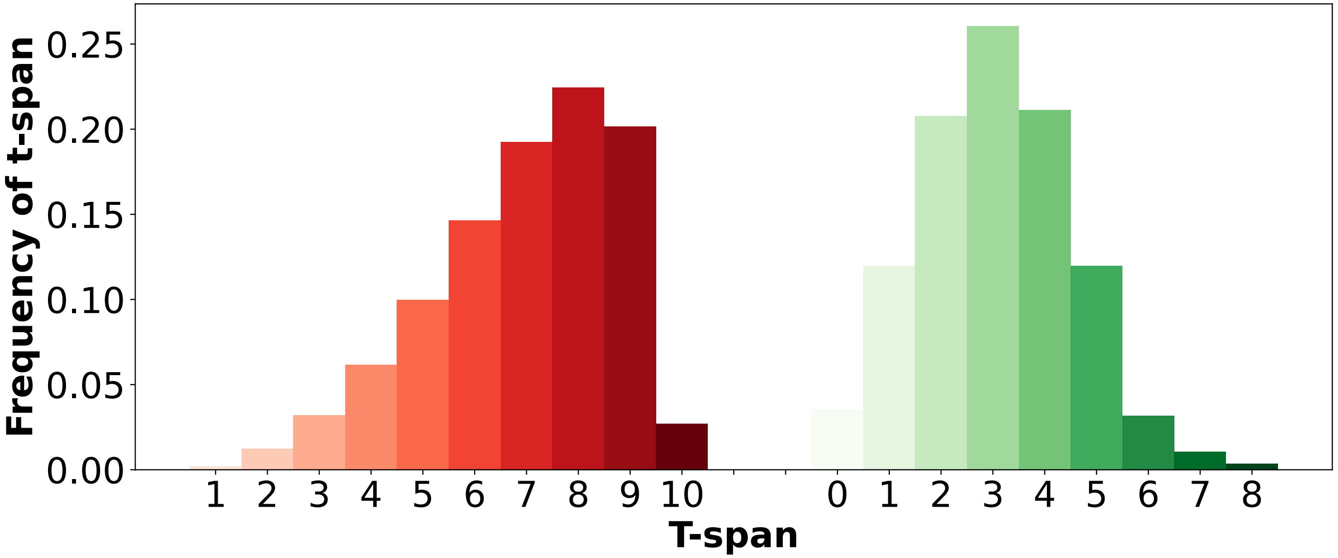


		Frequent in GERM		
	DBLP	Frequent	Not frequent	*TOT*
Frequent in how many realizations of the null model	Frequent in all	284	3431	**3715**
	Frequent in some	3	153	**156**
	Not frequent	9		
	TOT	**3871**	**296**	

Do other set tell something about evolution?

DBLP CASE STUDY





References

[1] *E. Scharwa¨chter, E. Mu¨ller, J. Donges, M. Hassani, and T. Seidl, “Detecting change processes in dynamic networks by frequent graph evolution rule mining,” in 2016 IEEE 16th International Conference on Data Mining (ICDM). IEEE, 2016, pp. 1191–1196.*