Motifs in Temporal Networks

Ashwin Paranjape, Austin Benson, & Jure Leskovec
{ashwinpp,arbenson,jure}@stanford.edu
Code & data available at http://snap.stanford.edu/temporal-motifs/

Stanford University

Overview

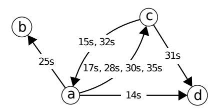
Temporal networks model dynamic complex systems such as telecommunications, credit card payments, and social interactions.

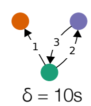
Two common ways that people study temporal networks are
1. ***Growth models*** consider how nodes and edges enter a network (e.g., how does the internet infrastructure grow over time)
2. ***Snapshot analysis*** creates a sequence of static graphs by aggregating links in coarse-grained intervals (e.g., daily phone call graph)

These existing analyses do not capture the rich temporal information of complex systems that are constantly in motion.

Temporal network motifs

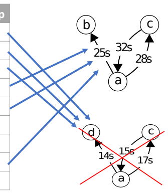
We propose temporal network motifs, or small temporal subgraph patterns as an analytical tool for temporal networks. These are analogous to network motifs, which are small subgraph patterns, used to study static graphs.





r-node, k-edge temporal network motif
1. Directed multigraph with r nodes and k edges
2. Edge ordering
3. Maximum time span δ

source	destination	timestamp
a	d	14s
c	a	15s
a	c	17s
a	b	25s
a	c	28s
a	c	30s
c	d	31s
c	a	32s
a	c	35s



Motif instance k temporal edges that match the pattern that all occur within δ time

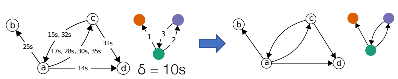
Wrong order!
(c, a) before (a, c)

Efficient counting algorithms

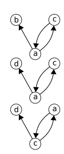
Given a temporal network and a motif, we want to efficiently count the number of instances of the motif in the temporal network.

General algorithm for any motif

1. Ignore timestamps to get a static graph and a static motif.



2. Find instances of the static motif in the static graph (using known algorithms).
3. For each static motif instance, fetch time-ordered temporal edges.

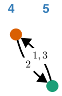


15s	17s	25s	28s	30s	32s	35s
(c,a)	(a,c)	(a,b)	(a,c)	(a,c)	(c,a)	(a,c)
14s	15s	17s	28s	30s	32s	35s
(a,d)	(c,a)	(a,c)	(a,c)	(a,c)	(c,a)	(a,c)
15s	17s	28s	30s	31s	32s	35s
(c,a)	(a,c)	(a,c)	(a,c)	(c,d)	(c,a)	(a,c)

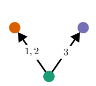
4. Count temporal motif instances in each temporal edge list using a dynamic programming algorithm that maintains subsequence counts. **Runs in linear time in the number of timestamps.**

	15s (c,a)	17s (a,c)	25s (a,b)	28s (a,c)	30s (a,c)	32s (c,a)	35s (a,c)
counts$[(a,b)]$	0	0	1	1	1	1	1
counts$[(a,c)]$	0	1	1	1	2	2	3
counts$[(c,a)]$	1	1	1	0	0	1	1
counts$[(a,b)(a,c)]$	0	0	0	1	2	2	3
counts$[(c,a)(a,c)]$	0	1	1	0	0	0	1
counts$[(a,b)(a,c)(c,a)]$	0	0	0	0	0	2	2
start	1	1	1	3	3	3	3
end	1	2	3	4	5	6	7

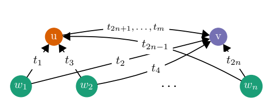
Special case analysis. Runs in $O(k^2m)$ time for 2-node, k-edge motifs, where m is the total number of temporal edges.
This is **optimal**, i.e., linear in the size of the data for constant k.

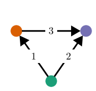


Faster algorithms for special cases



3-node, 3-edge stars
Problem have to enumerate pairs of neighbors of high-degree nodes
Improvement count for all neighbors simultaneously
Runtime complexity is *O(m)*, where m is the total number of edges (**optimal**)



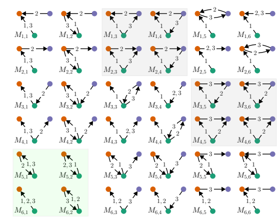


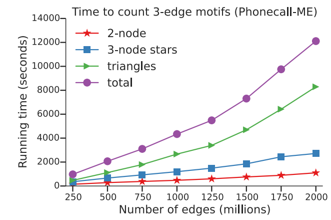
3-node, 3-edge triangles
Problem a static edge with many timestamps may appear in several triangles → *O(Tm)* complexity, where T is the number of static triangles
Improvement simultaneously count triangles for edges with many timestamps
Runtime complexity is $O(T^{1/2}m)$, a significant improvement

Triangle speedups	Wiki. edits	Stack Overflow	Bitcoin	Texts	Phone calls
# temp. edges	10M	63M	123M	800M	2.04B
speedup	1.92x	1.29x	56.5x	2.28x	1.42x

The 36 motifs we can count quickly

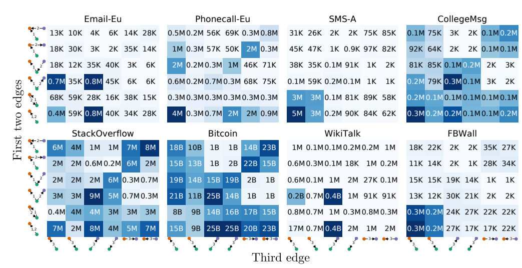
With our algorithms, we can count 2-node and 3-node, 3-edge motifs efficiently.
It takes a couple hours to count all 36 of these motifs for a phone call network with 2B edges.



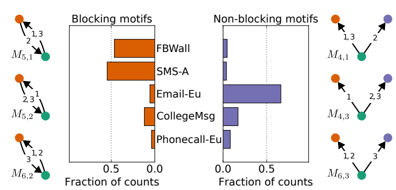


Empirical observations (δ = 1 hour)

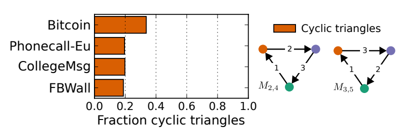
- **Email-Eu** E-mails between researchers.
- **Phonecall-Eu** Phone call records.
- **SMS-A** Text messages.
- **College-Msg** Online private messages.
- **StackOverflow** Answers to questions and comments on questions and answers.
- **Bitcoin** transactions between addresses.
- **WikiTalk** edits of user talk pages.
- **FBWall** Facebook wall posts.



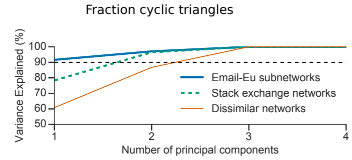
Counts of all 2- and 3-node, 3-edge temporal motifs in our datasets (δ =1 hour). Along each row, the first two edges are the same. Along each column, the third edge is the same.



Blocking and non-blocking communication.
Fraction of all 2 and 3-node, 3-edge motif counts that correspond to two groups of motifs. Motifs on the left capture "blocking" behavior (common in SMS) and motifs on the right capture "non-blocking" behavior (common in email).

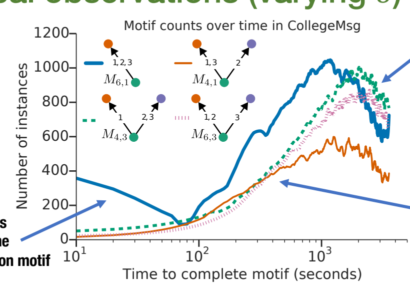


Cyclic triangles in Bitcoin.
Fraction of 3-edge temporal triangle motifs corresponding to cyclic triangles. Bitcoin has a much higher fraction compared to all other datasets.



Networks from the same domain have similar motif count distributions.
Variance explained by number of principal components for three groups of networks.

Empirical observations (varying δ)



Fewer switches in target → more common

Early time scales dominated by the single-destination motif

Overall growth from 1 to 20 minutes

References

Motifs for static networks
1. Milo, Ron, et al. Network motifs: simple building blocks of complex networks. Science 298.5594 (2002): 824-827.

Similar temporal network pattern counting ideas
2. Kovanen, et al. Temporal motifs in time-dependent networks. JSTAT, 2011.
3. Zhao, et al. Communication motifs: a tool to characterize social communications. CIKM, 2010.
4. Gurukar, et al. Commit: A scalable approach to mining communication motifs from dynamic networks. SIGMOD, 2015.

Acknowledgements This research has been supported in part by NSF IIS- 1149837, ARO MURI, DARPA SIMPLEX and NGS2, Boeing, Bosch, Huawei, Lightspeed, SAP, Tencent, Volkswagen, Stanford Data Science Initiative, and a Stanford Graduate Fellowship.