Graphlets over Time: A New Lens for Temporal Network Analysis

Deukyeol Yoon, Dongjin Lee, Minyoung Choe, and Kijung Shin

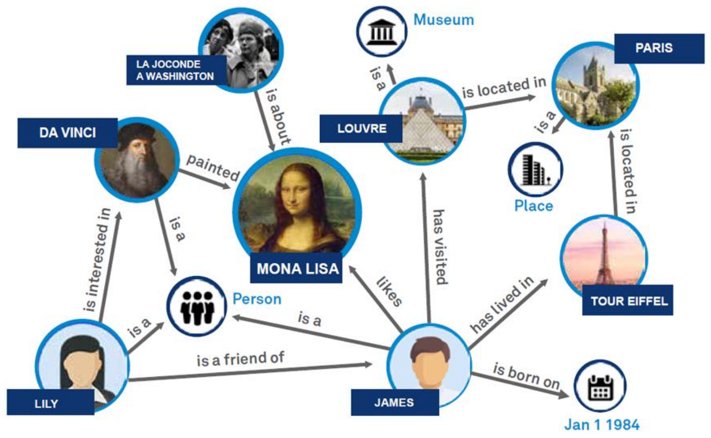
Kim Jaechul Graduate School of AI
KAIST

Graphs are Everywhere!

- A graph consists of a set of nodes and a set of edges
- Graphs are used to represent various types of data

Online social networks

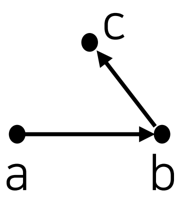
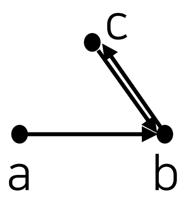
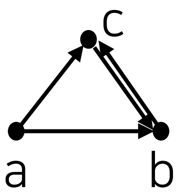
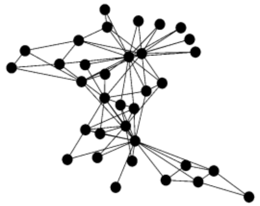
Purchase history



Knowledge graphs

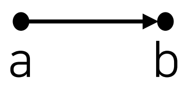
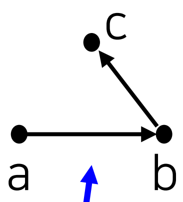
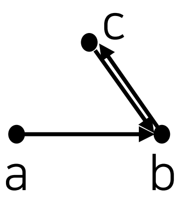
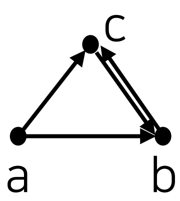
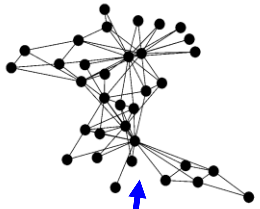
Real-world Graphs are Evolving over Time

- Many real-world graphs evolve over time, especially with newly arriving nodes and edges
- They are naturally represented as a stream of edges

Time t	**1**	**2**	**3**	**4**	**…**	***T***
Arriving Edge	a → b	b → c	c → b	a → c	…	…
Graph (Directed)					…	

Real-world Graphs are Evolving over Time

- Many real-world graphs evolve over time, especially with newly arriving nodes and edges
- They are naturally represented as a stream of edges

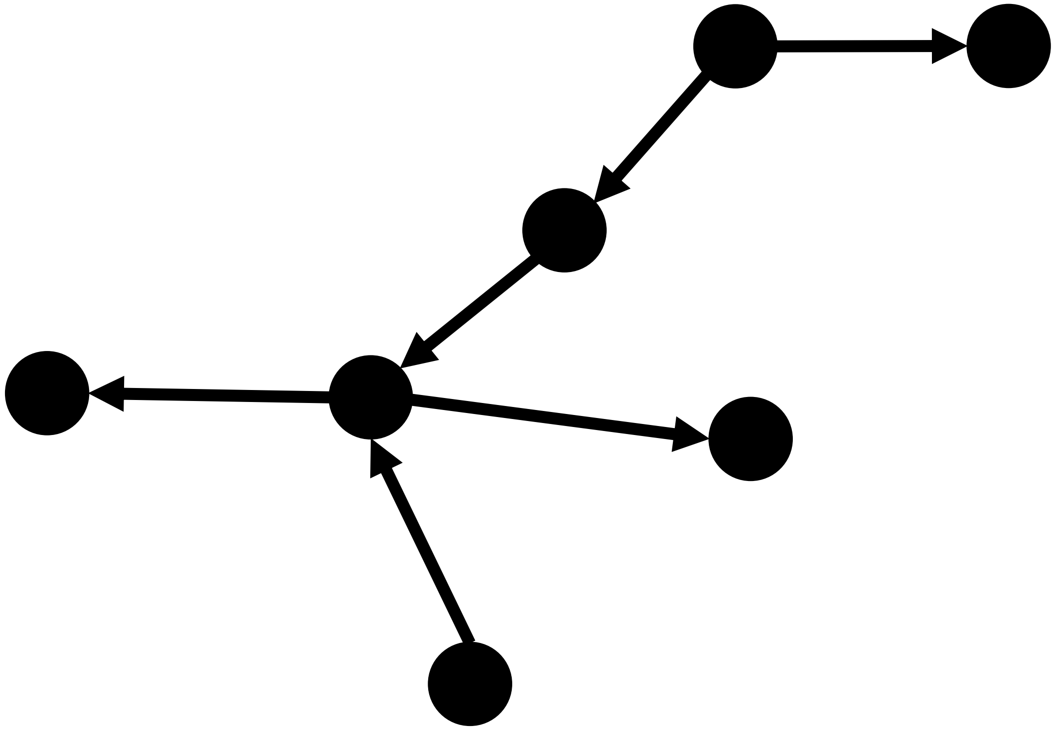
Time t	1	2	3	4	…	T
Arriving Edge	a → b	b → c	c → b	a → c	…	…
Graph (Directed)					…	

Snapshot at time 2

Snapshot at time T

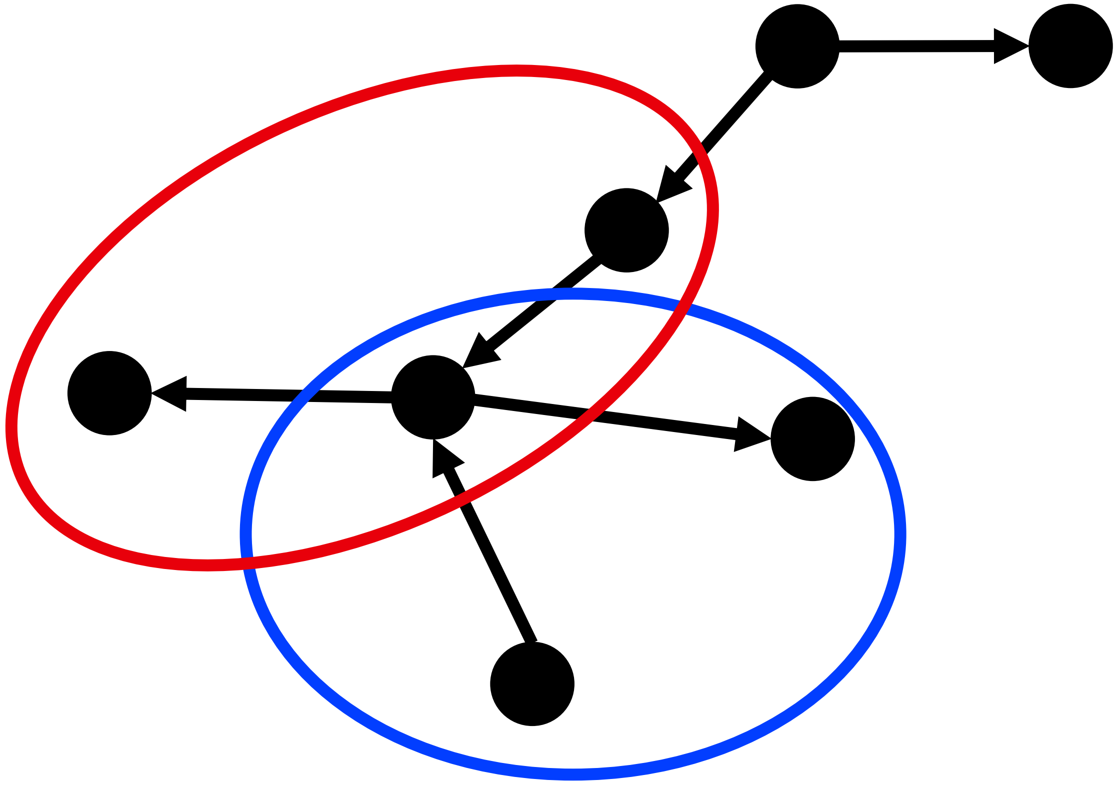
Graphlet: Example

- Consider a snapshot of an edge stream



Graphlet: Example

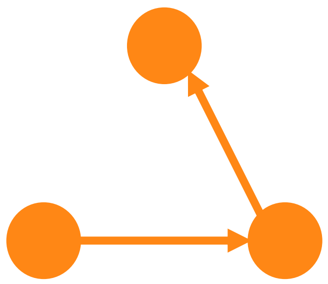
- There are induced subgraphs that are isomorphic to each other

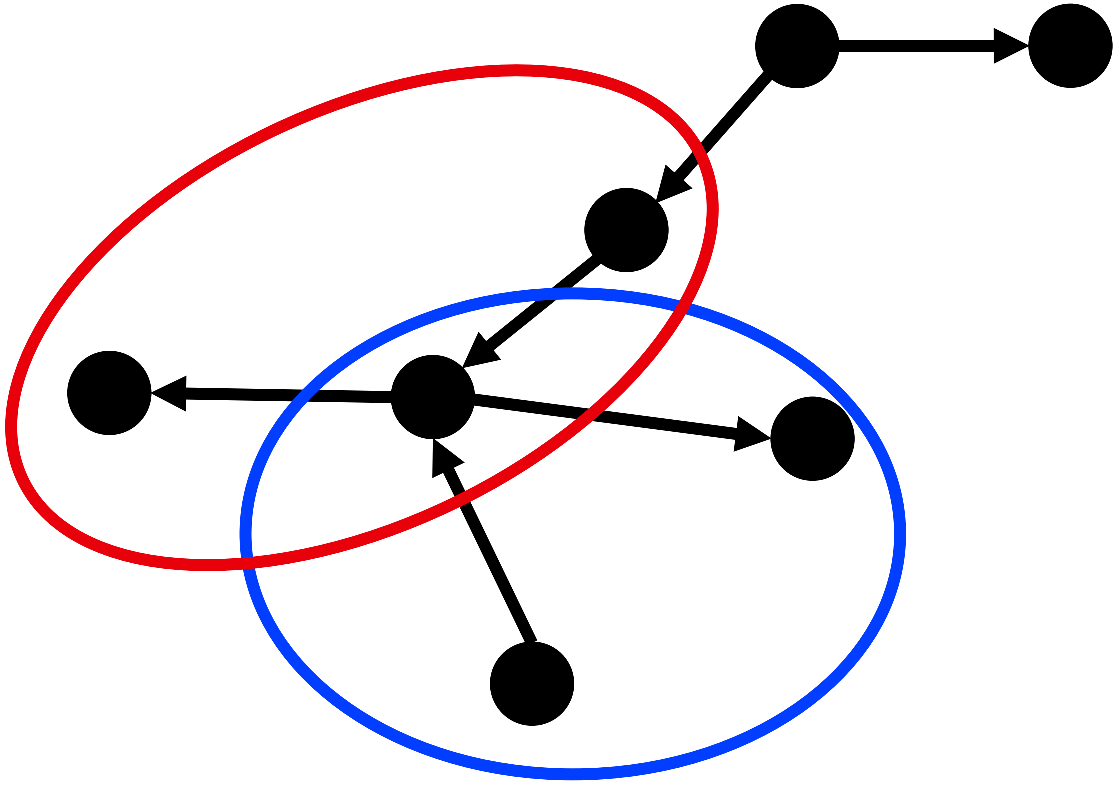


Graphlet: Example

- There are induced subgraphs that are isomorphic to each other

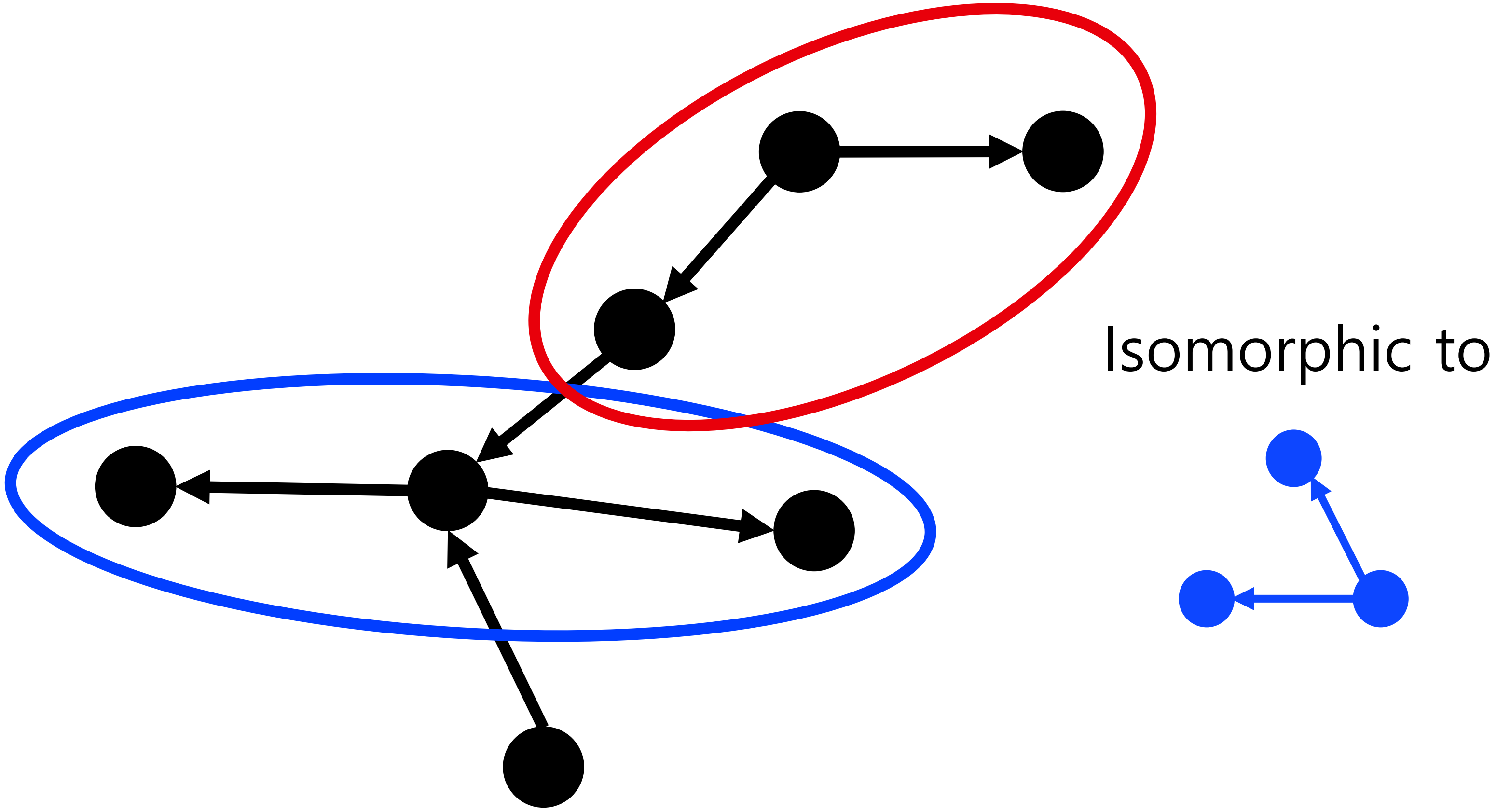
Isomorphic to





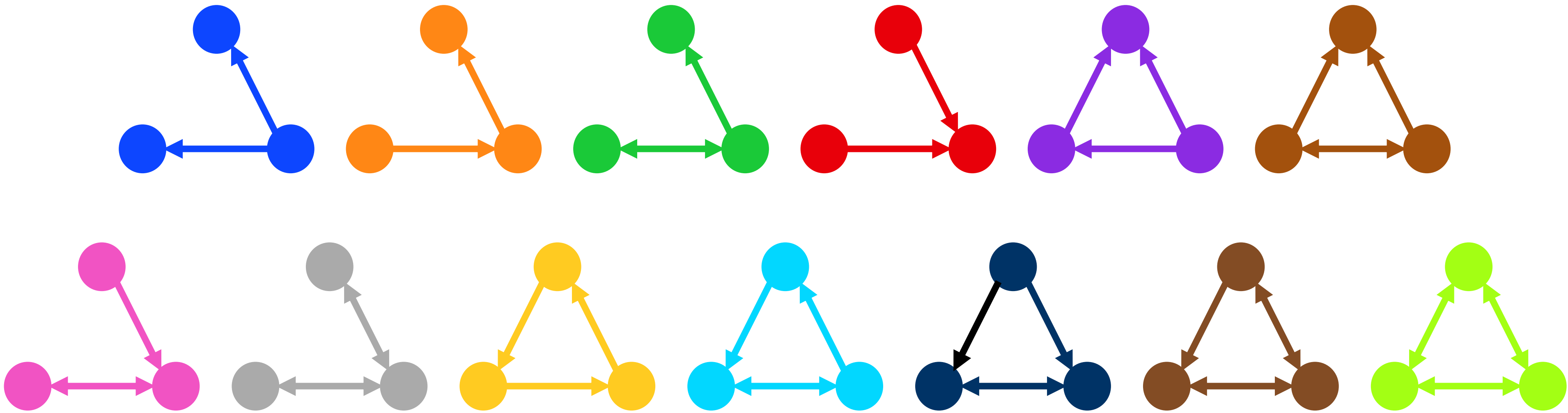
Graphlet: Example

- There are induced subgraphs that are isomorphic to each other



Graphlet

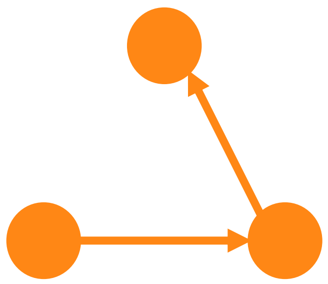
- A graphlet is a class (set) of isomorphic induced subgraphs
- Graphlets are used to analyze “higher-order” connectivity of graphs
- This work focuses on 13 (weakly) connected graphlets of size 3 defined on directed graphs

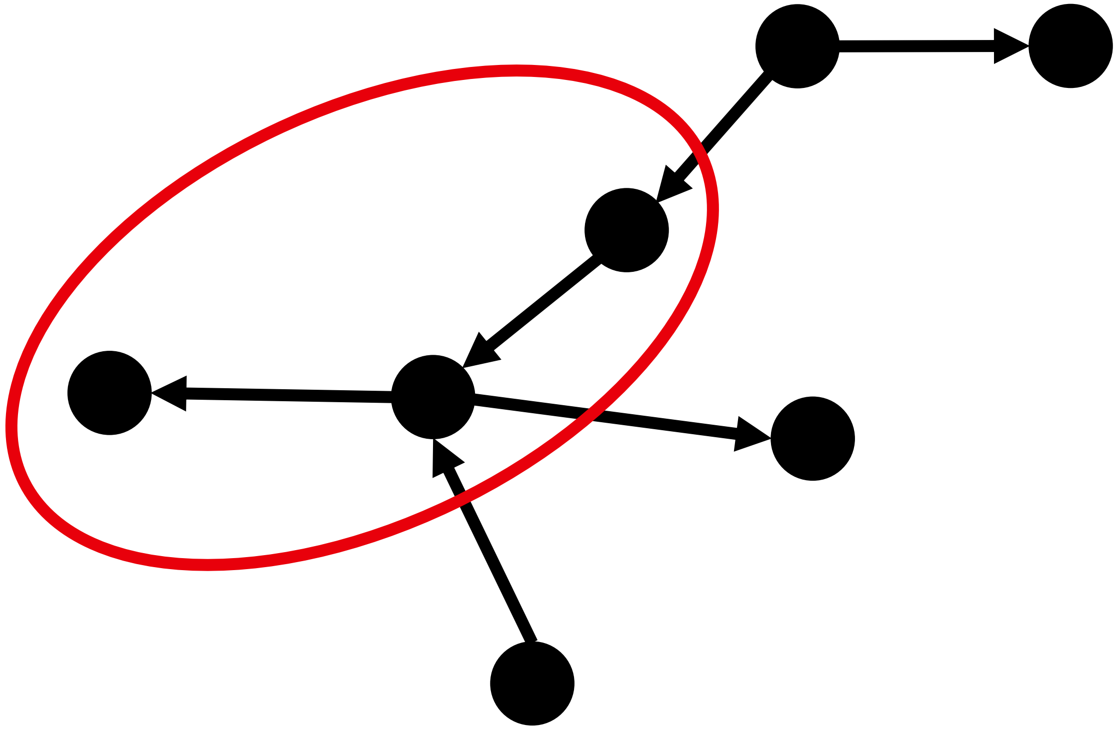


Graphlet Instance

- An induced subgraph corresponding to a graphlet is called instance

Instance of graphlet:

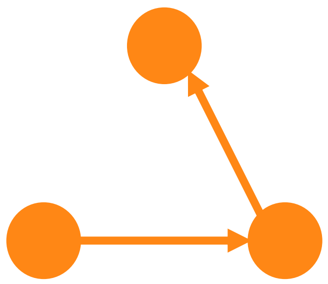


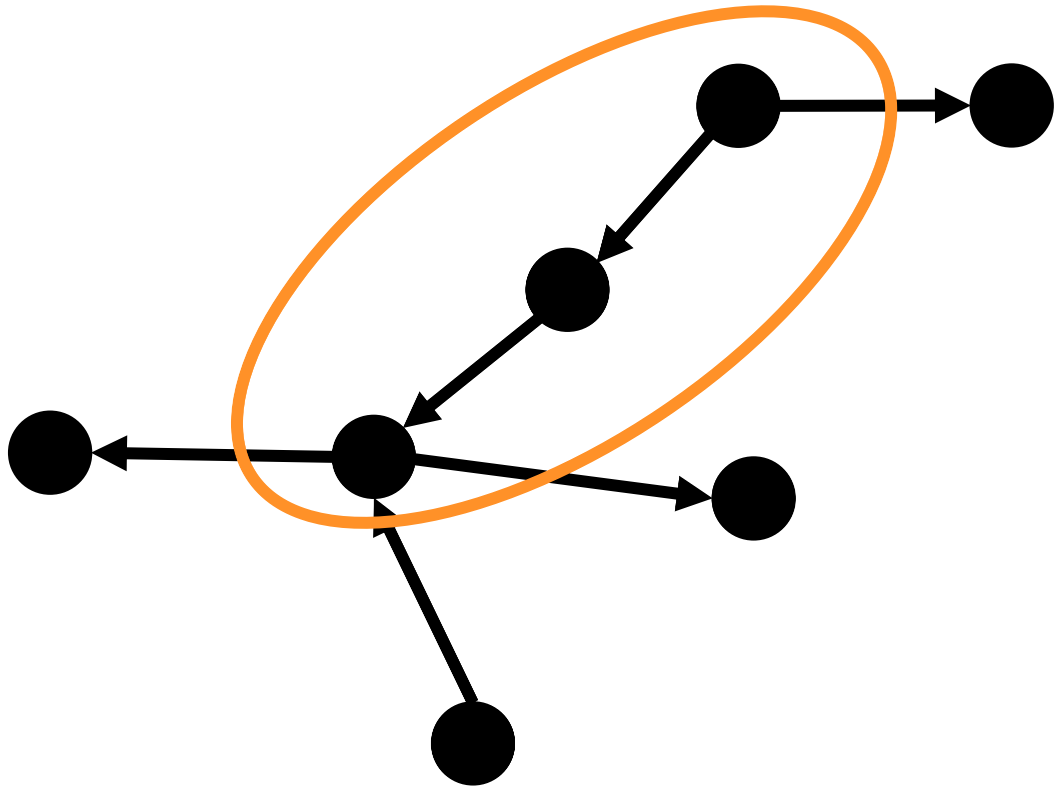


Graphlet Instance

- An induced subgraph corresponding to a graphlet is called instance

Instance of graphlet:

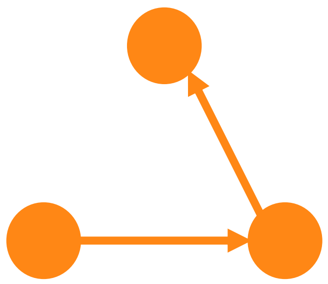


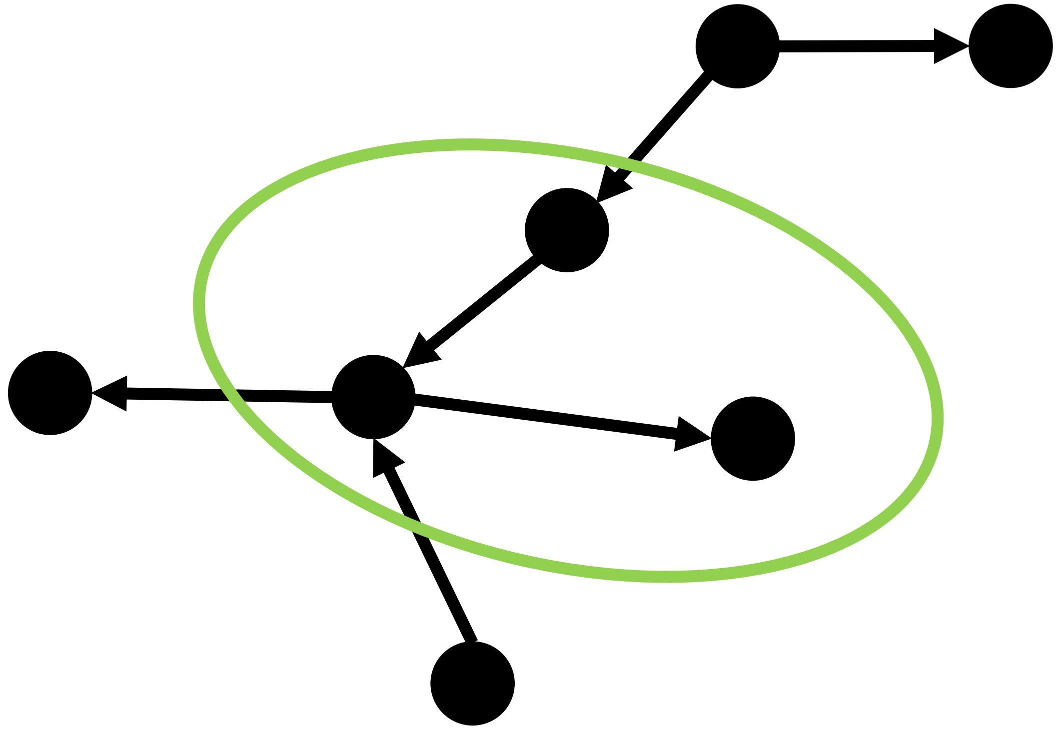


Graphlet Instance

- An induced subgraph corresponding to a graphlet is called instance

Instance of graphlet:

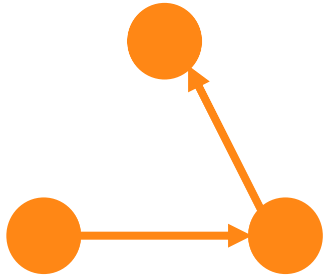


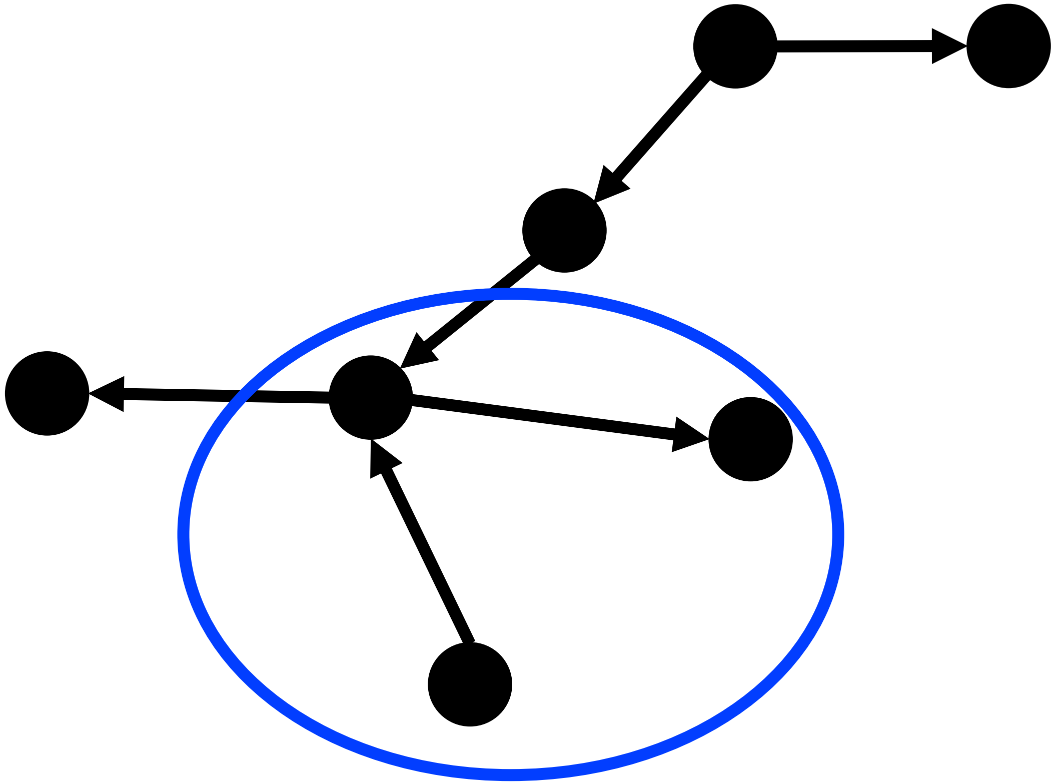


Graphlet Instance

- An induced subgraph corresponding to a graphlet is called instance

Instance of graphlet:

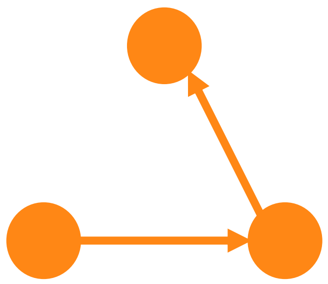


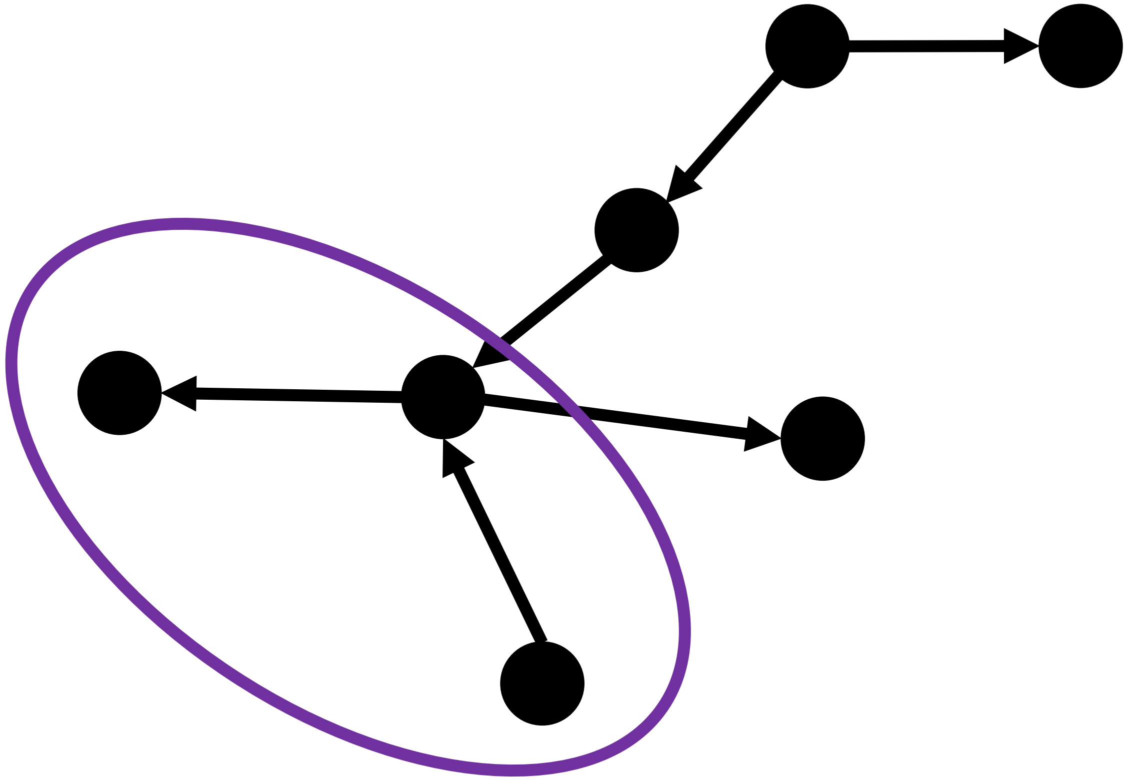


Graphlet Instance

- An induced subgraph corresponding to a graphlet is called instance

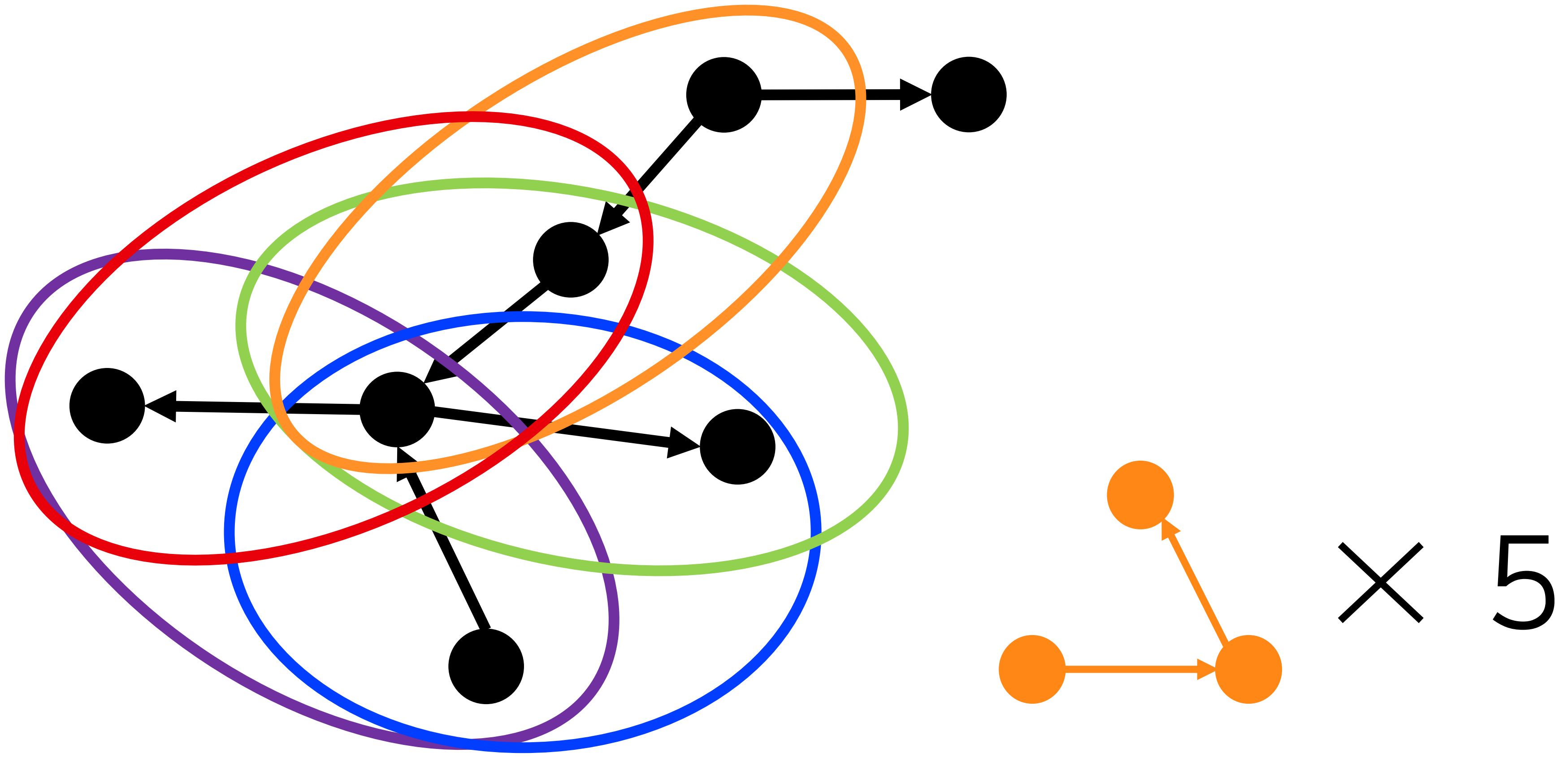
Instance of graphlet:





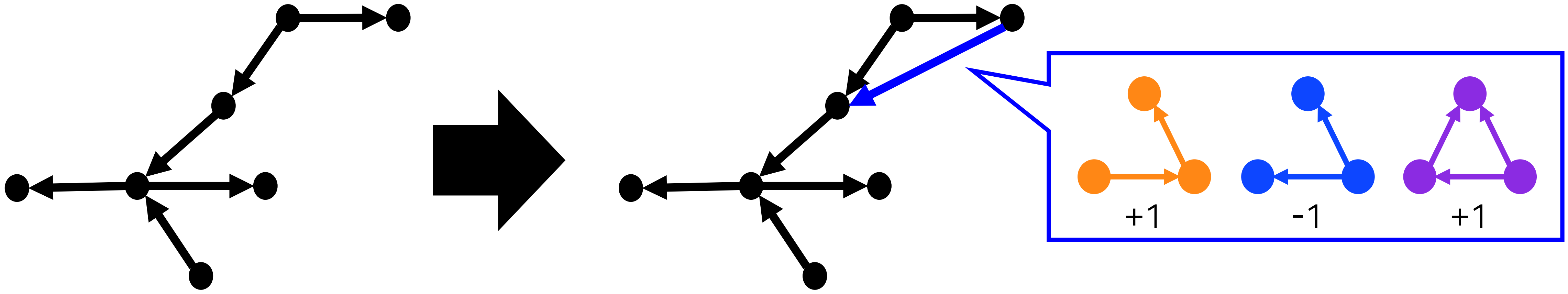
Graphlet Instance

- An induced subgraph corresponding to a graphlet is called instance



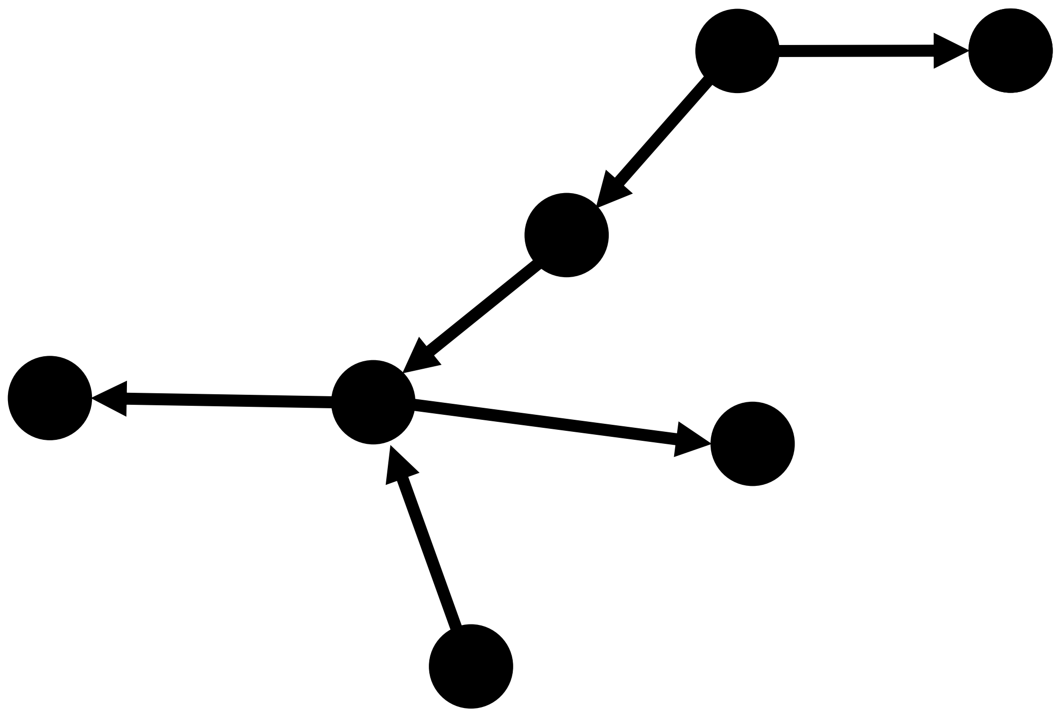
Summary of Our Contributions

- **Research questions:** *"How do real-world graphs evolve over time?"*
  - We focus on local structures rather than global structures (e.g., diameter)
- **Method**: to examine the changes of graphlet instances over time
- **Outcome**: patterns and analysis tools

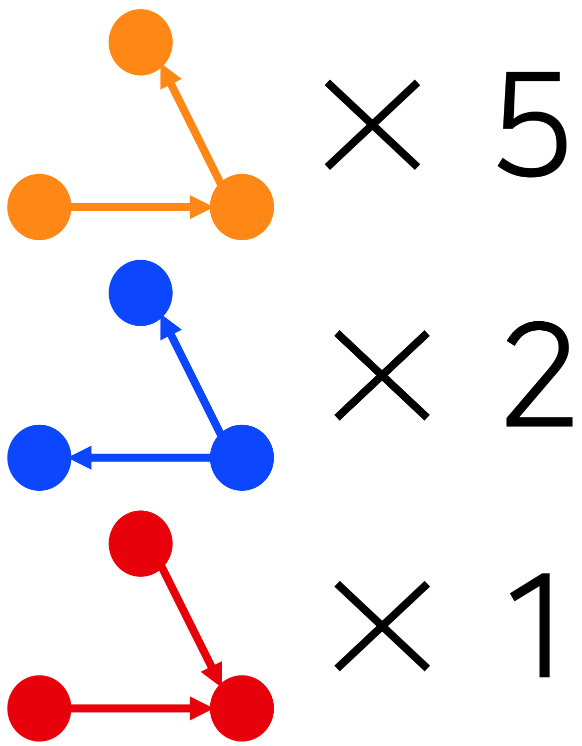


Graphlets over Time: Example

- *How does the graphlet instance distribution change as a graph grows?*

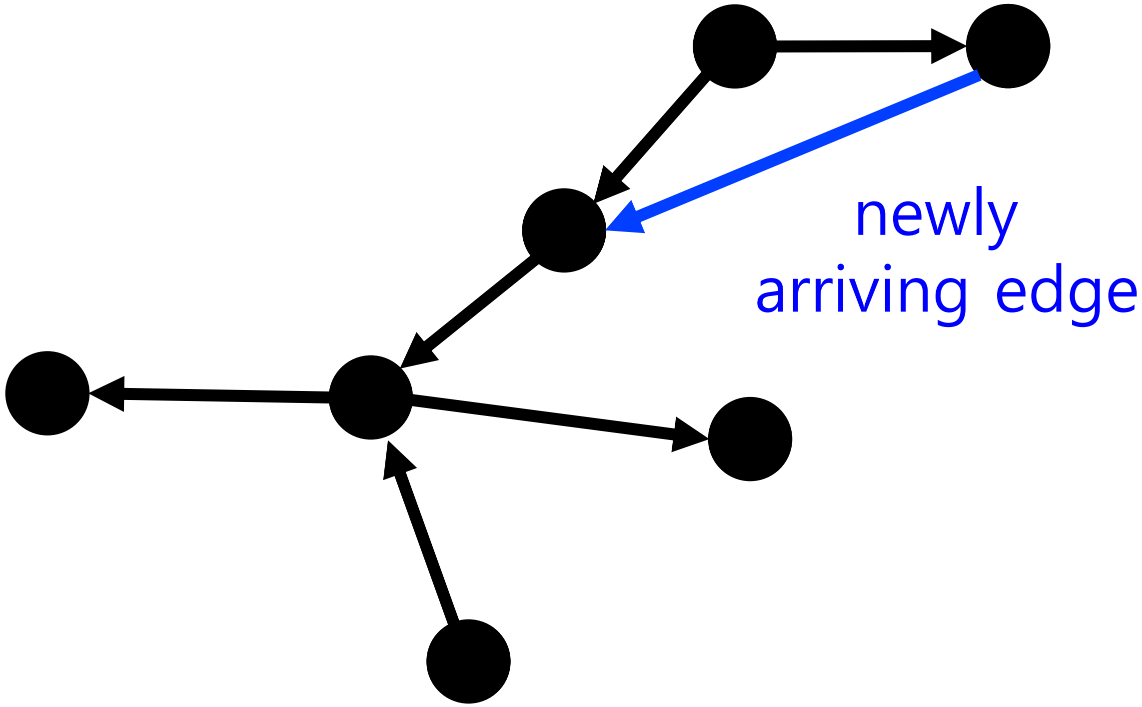


Snapshot at time t

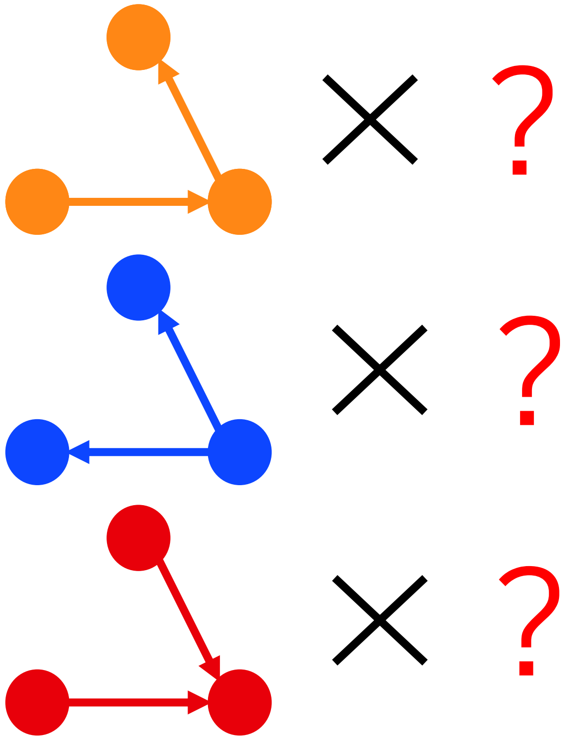


Graphlets over Time: Example

- *How does the graphlet instance distribution change as a graph grows?*

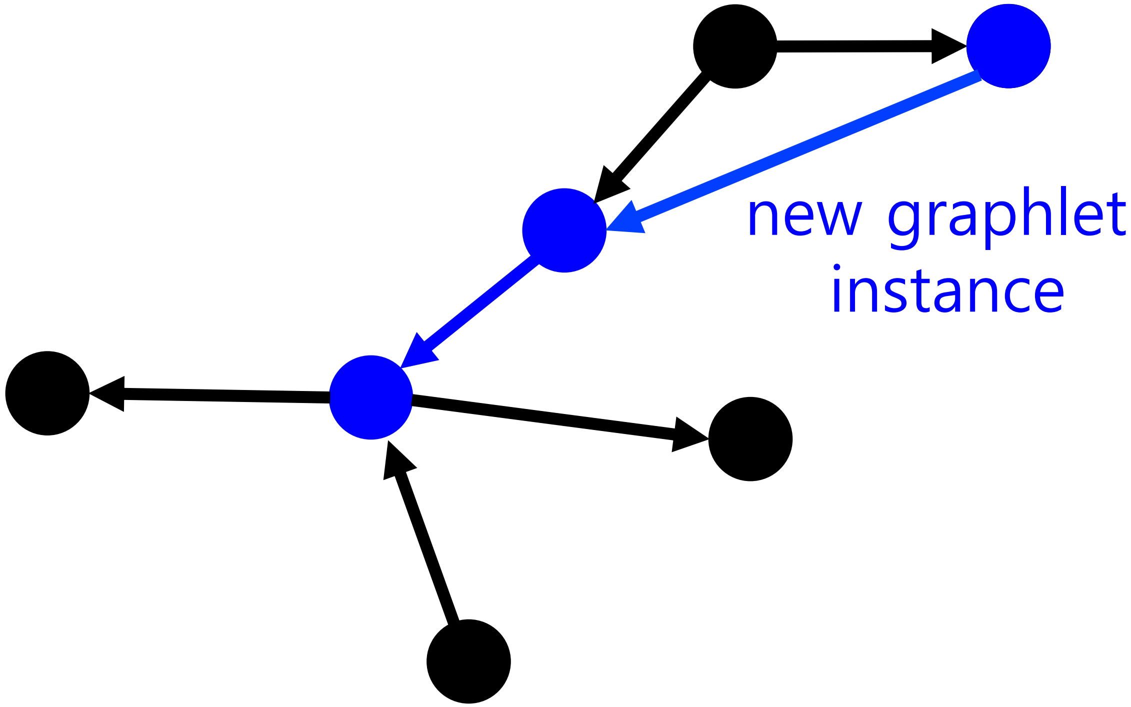


Snapshot at time $t + 1$

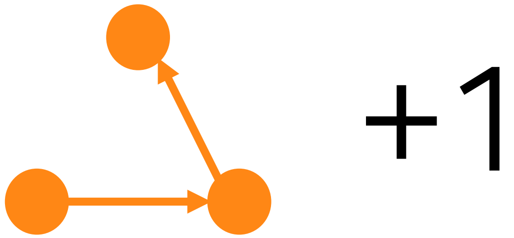


Graphlets over Time: Example

- *How does the graphlet instance distribution change as a graph grows?*

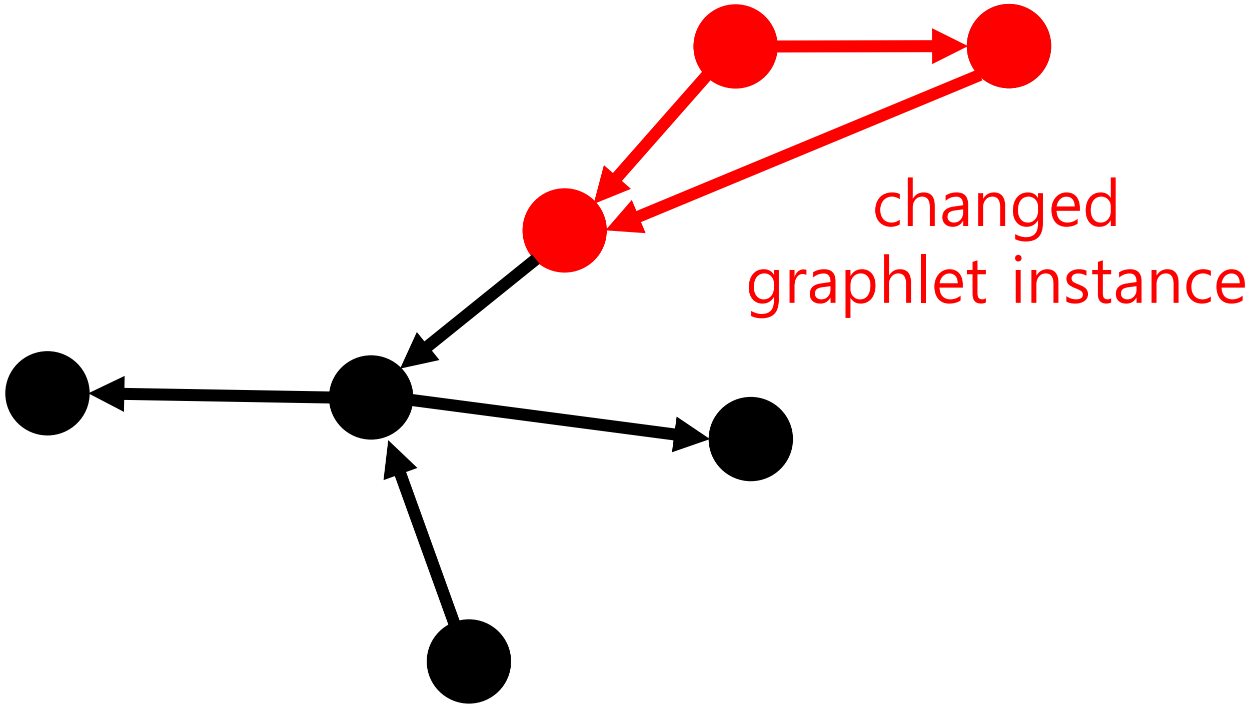


Snapshot at time $t+1$

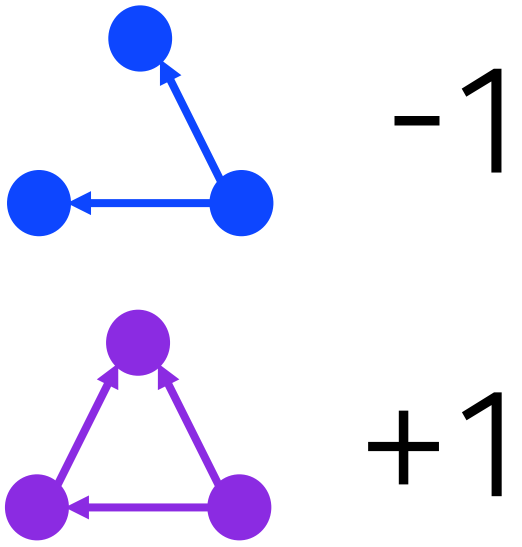


Graphlets over Time: Example

- *How does the graphlet instance distribution change as a graph grows?*

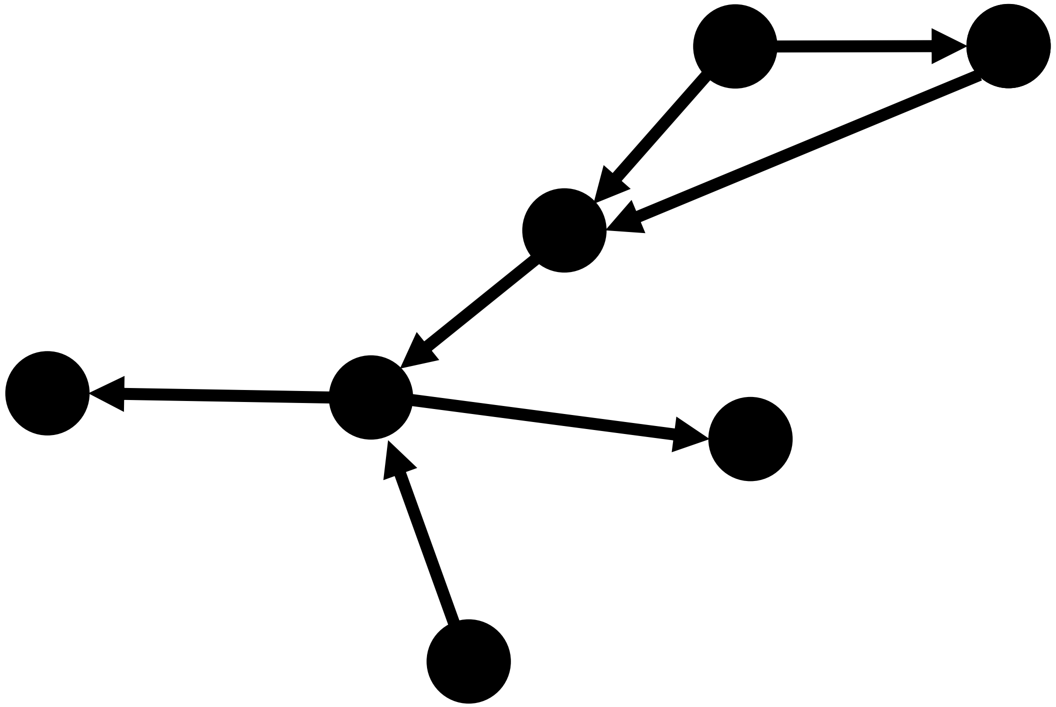


Snapshot at time $t + 1$

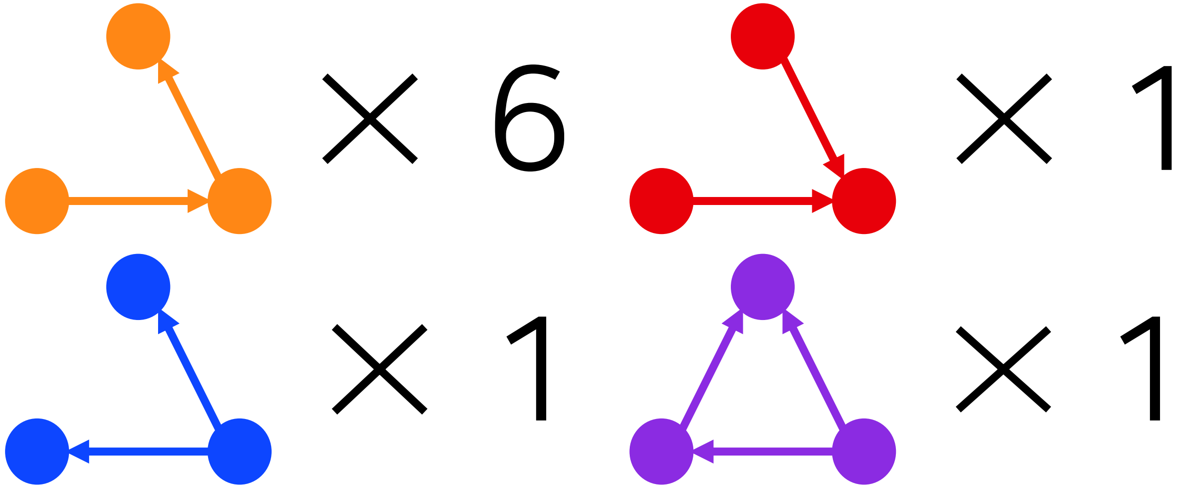


Graphlets over Time: Example

- *How does the graphlet instance distribution change as a graph grows?*

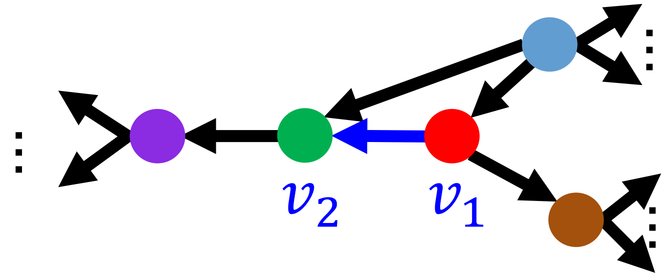


Snapshot at time $t + 1$



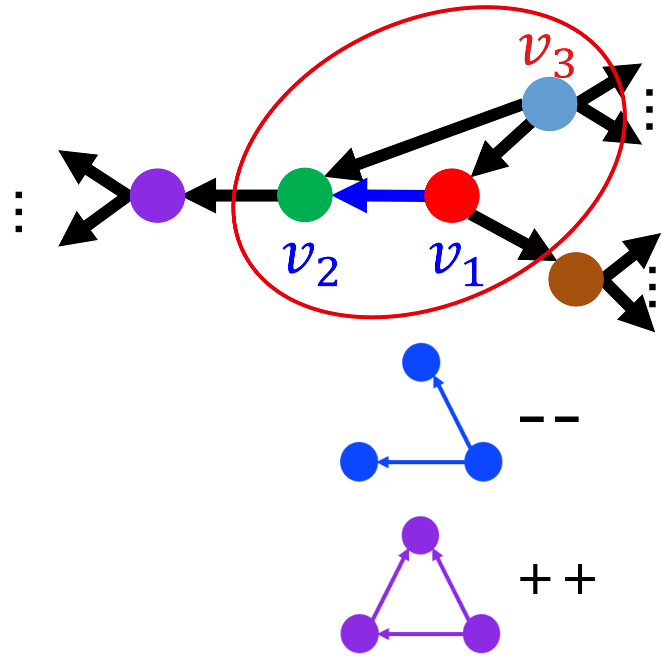
Graphlets over Time: Algorithm

- For each newly arriving edge $v_1 \rightarrow v_2$:
  - For each neighbor v_3 of v_1 or v_2



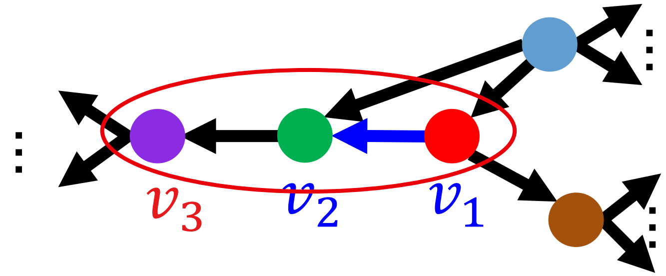
Graphlets over Time: Algorithm

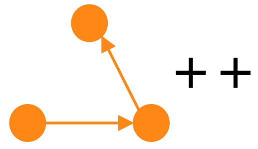
- For each newly arriving edge $v_1 \rightarrow v_2$:
  - For each neighbor v_3 of v_1 or v_2:
    - If v_1, v_2, v_3 is previously connected:
      - num_prev_graphlet_instance--
      - num_new_graphlet_instance++



Graphlets over Time: Algorithm

- For each newly arriving edge $v_1 \rightarrow v_2$:
  - For each neighbor v_3 of v_1 or v_2:
    - If v_1, v_2, v_3 is previously connected:
      - num_prev_graphlet_instance--
      - num_new_graphlet_instance++
    - Else:
      - num_new_graphlet_instance++





Graphlets over Time: Algorithm

- Our algorithm achieves the lowest complexity possible for any enumeration-based method

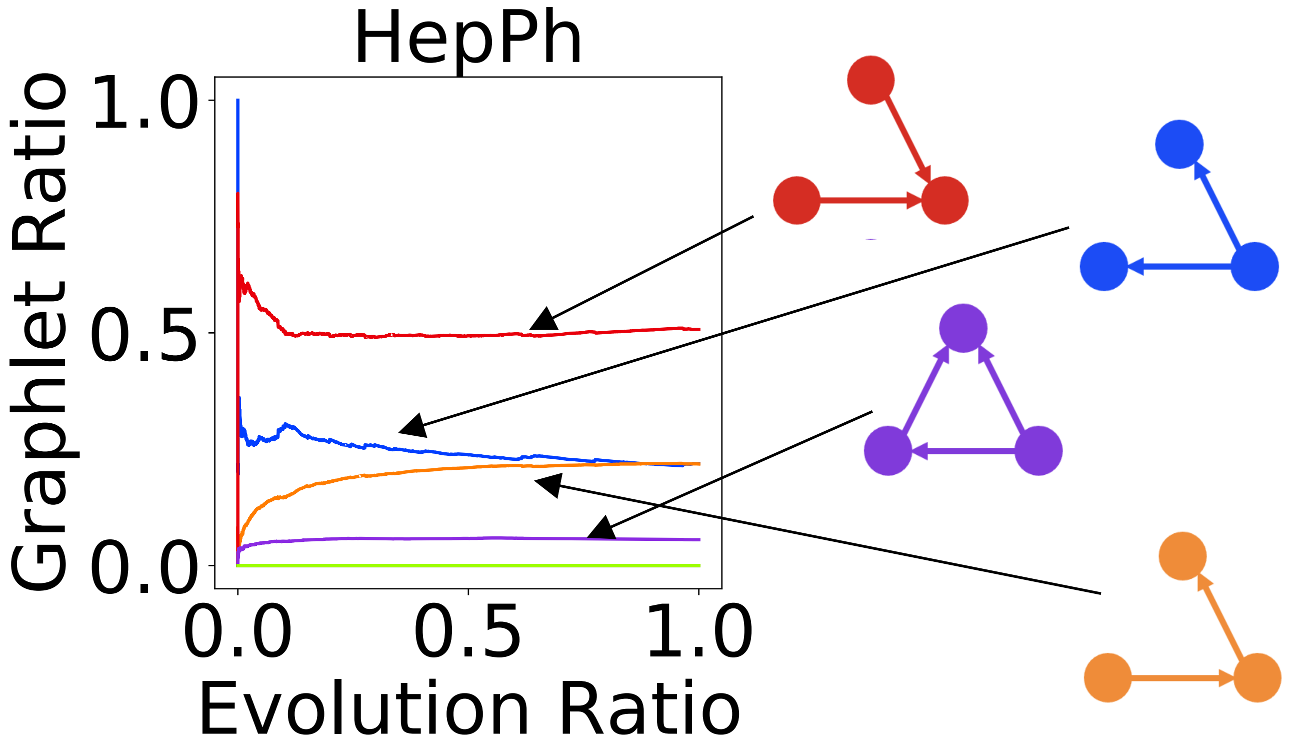
- ***Theorem***. *Given an edge stream, the time complexity of counting the instances of every graphlet in all snapshot by our algorithm is*

$$\Theta\left(\Sigma_{v\in\mathcal{V}}(d(v))^2\right) = \Theta(\#graphlet\ instances\ in\ the\ last\ snapshot),$$

where $d(v)$ *is the degree of node* v *in the last snapshot*

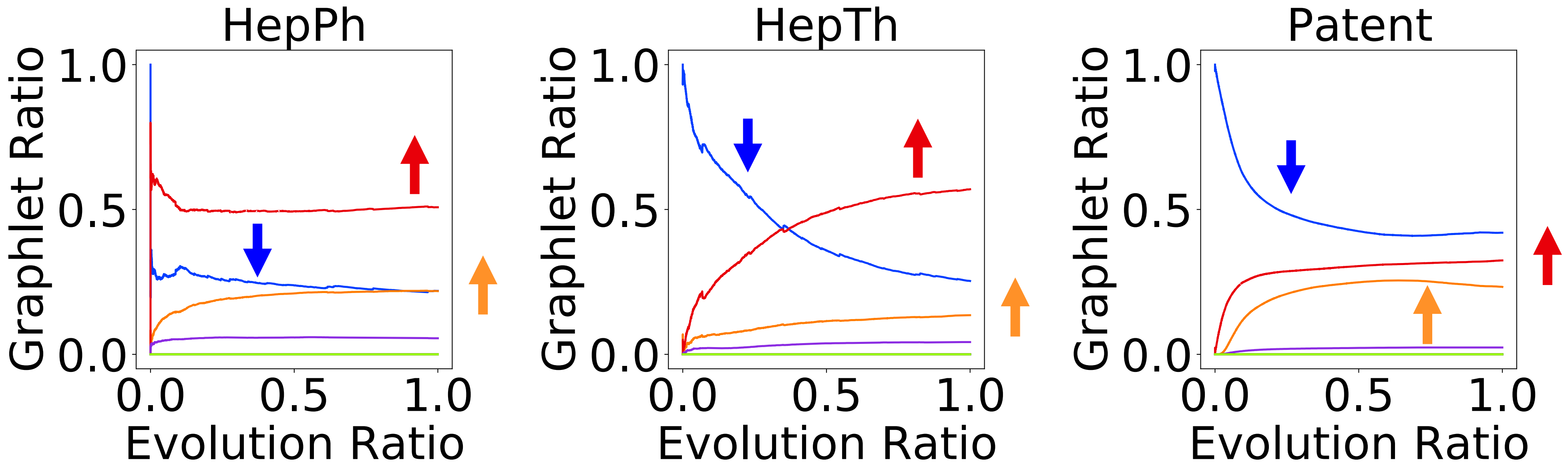
Graphlets over Time: Observations

- We examine the graphlet instance distribution change in real graphs
- *Example: a citation network (ArXiv HepPh)*



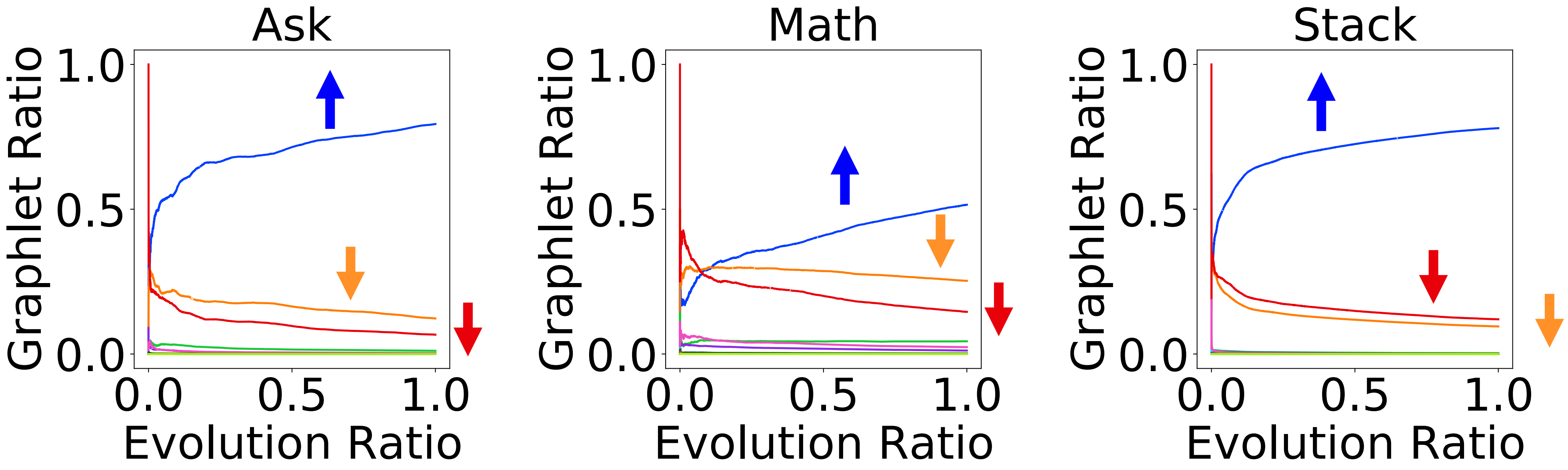
Graphlets over Time: Observations

- We can observe domain-based patterns
- Graphs from the same domain share similar patterns in their evolution
- *Example: citation networks*



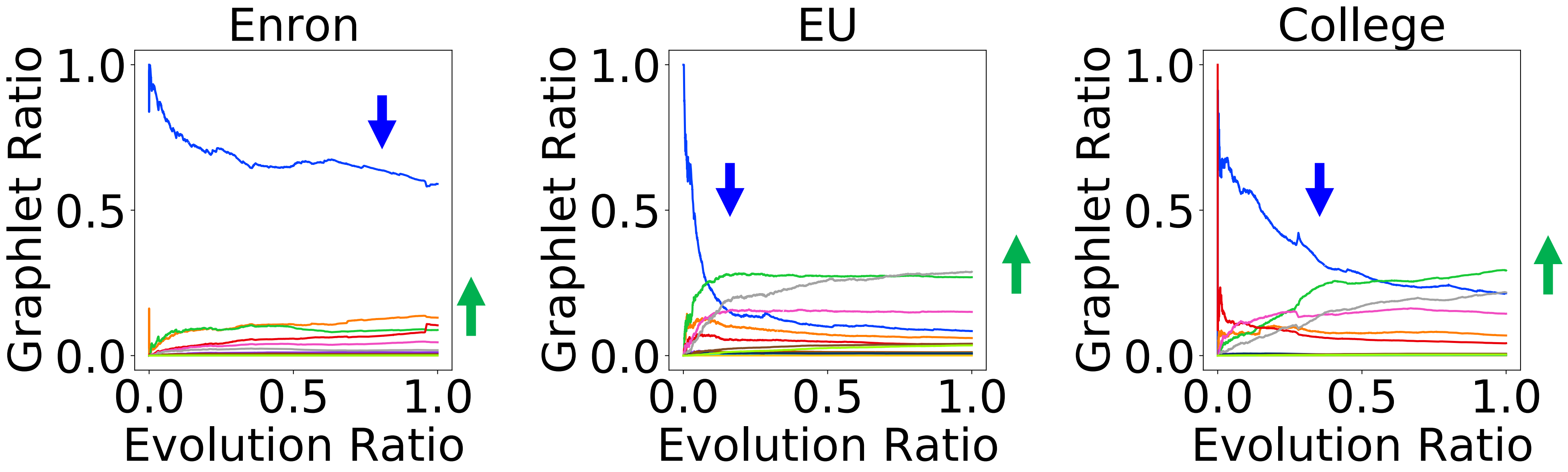
Graphlets over Time: Observations

- We can observe domain-based patterns
- Graphs from the same domain share similar patterns in their evolution
- *Example: online Q/A networks*



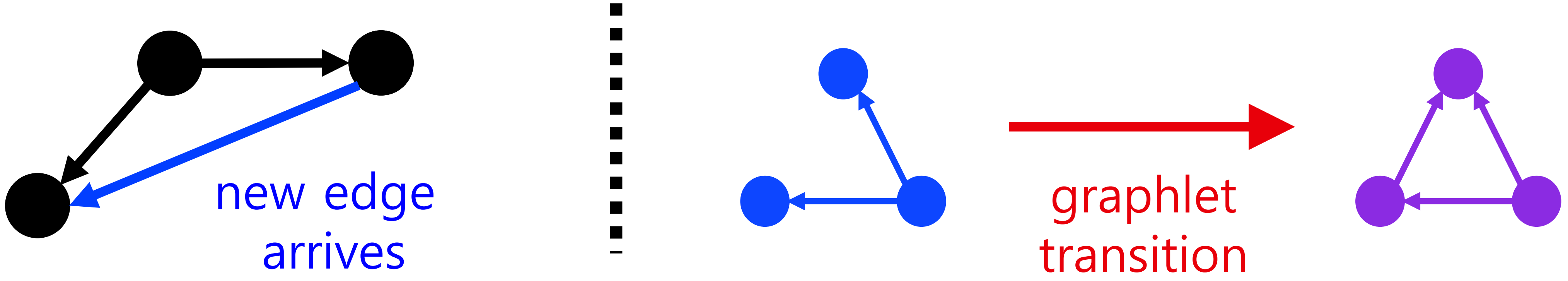
Graphlets over Time: Observations

- We can observe domain-based patterns
- Graphs from the same domain share similar patterns in their evolution
- *Example: email/message networks*



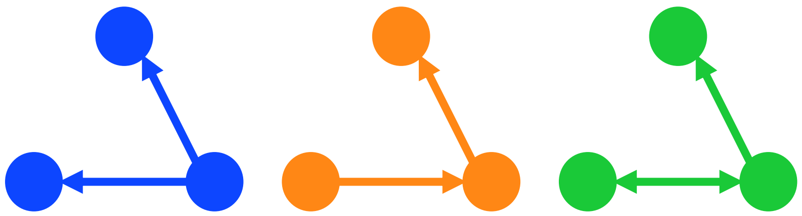
Graphlet Transition Graph: Motivation

- How can we compare evolutionary patterns in greater detail?
- Idea: to count the occurrences of graphlet transitions
  - Our counting algorithm is easily extended to count the transitions under the same time complexity

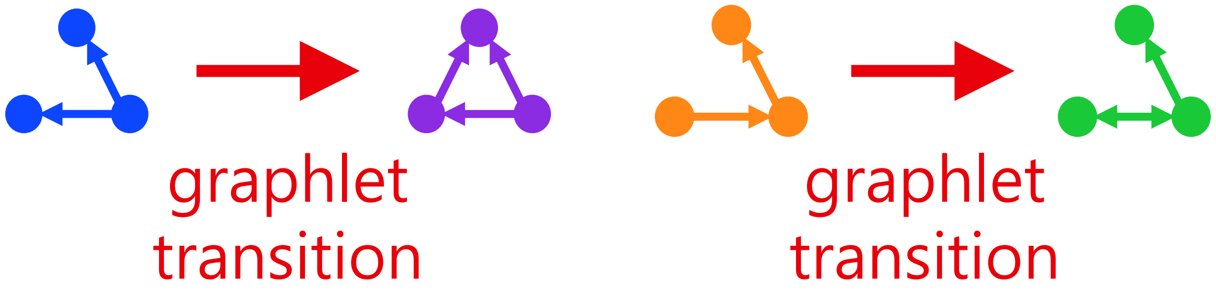


Graphlet Transition Graph: Definition

- Graphlet transition graph (GTG) of an edge stream is a graph where
  - Nodes: graphlets
  - Edges: transitions between graphlets
  - Weight: ratio of the transitions occurring in the edge stream

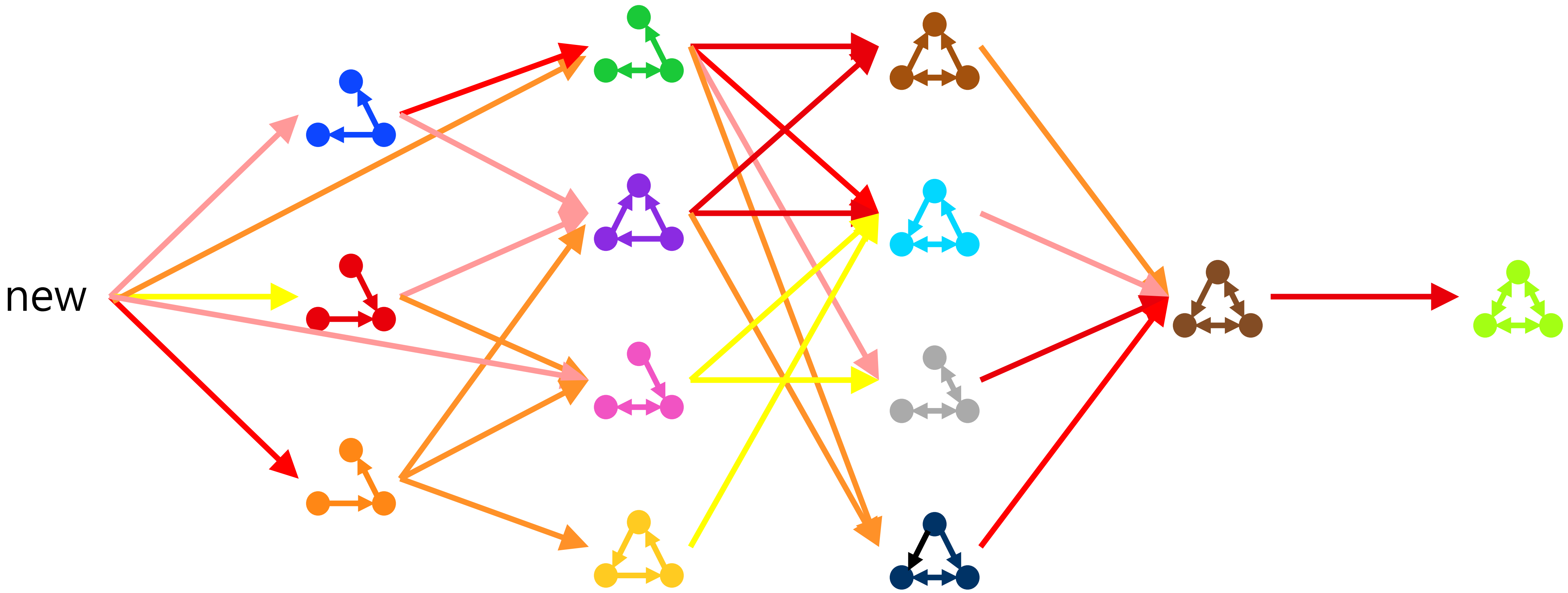


Nodes



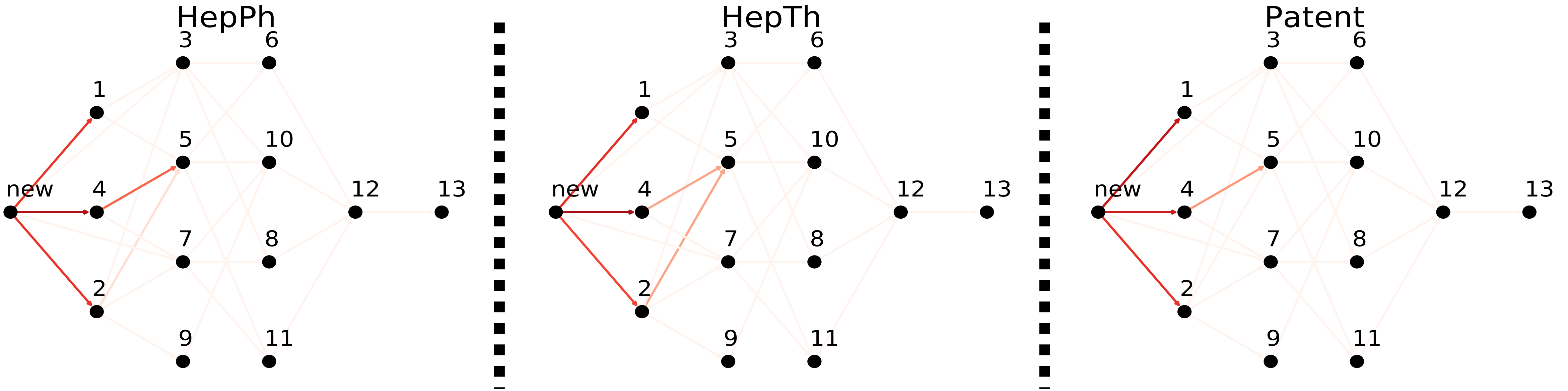
Edges

Graphlet Transition Graph: Example



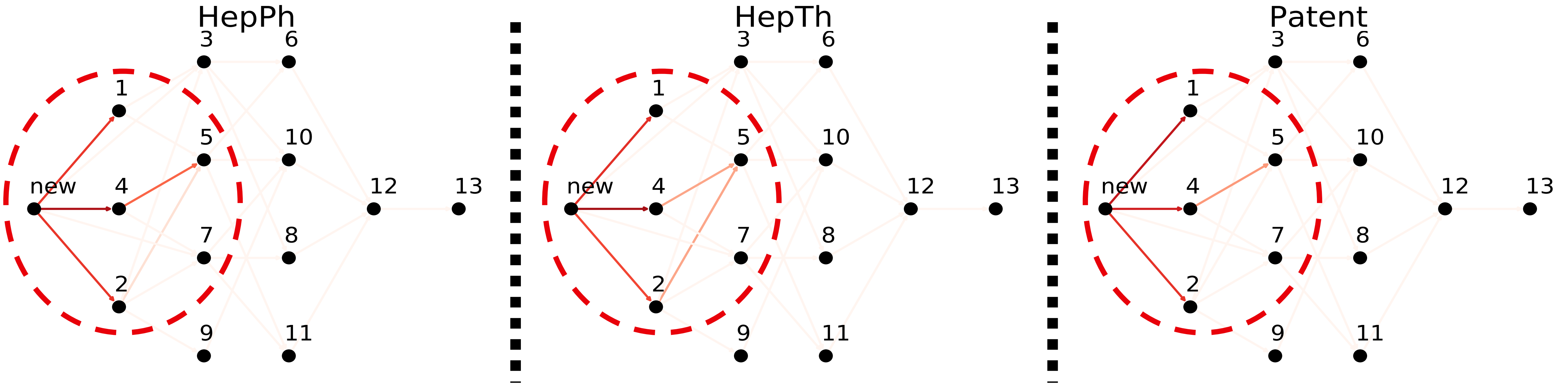
Graphlet Transition Graph: Observations

- We can observe domain-based patterns in GTGs
- Graphs from the same domain share similar edge-weight distributions
- *Example: citation networks*



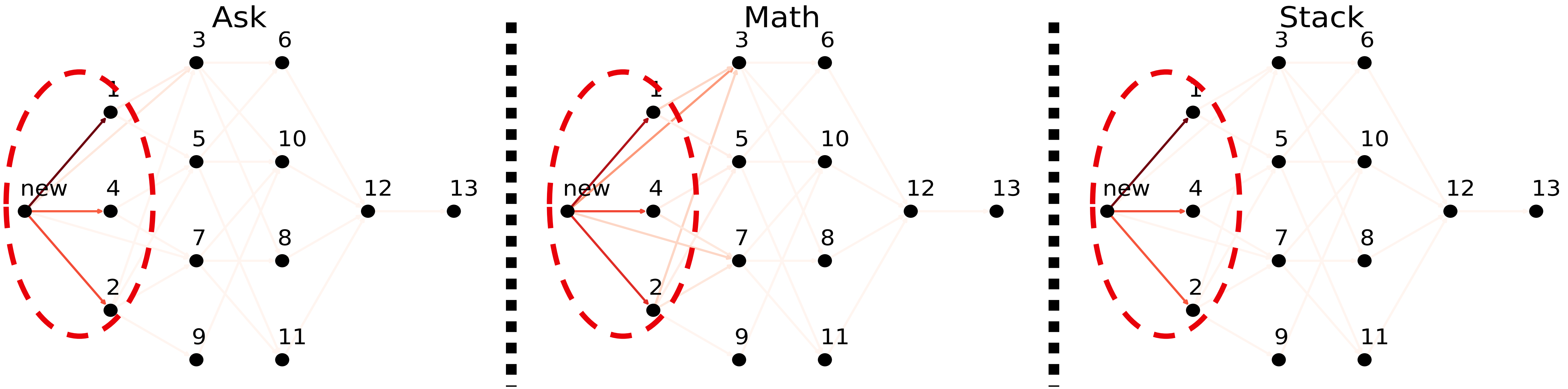
Graphlet Transition Graph: Observations

- We can observe domain-based patterns in GTGs
- Graphs from the same domain share similar edge-weight distributions
- *Example: citation networks*



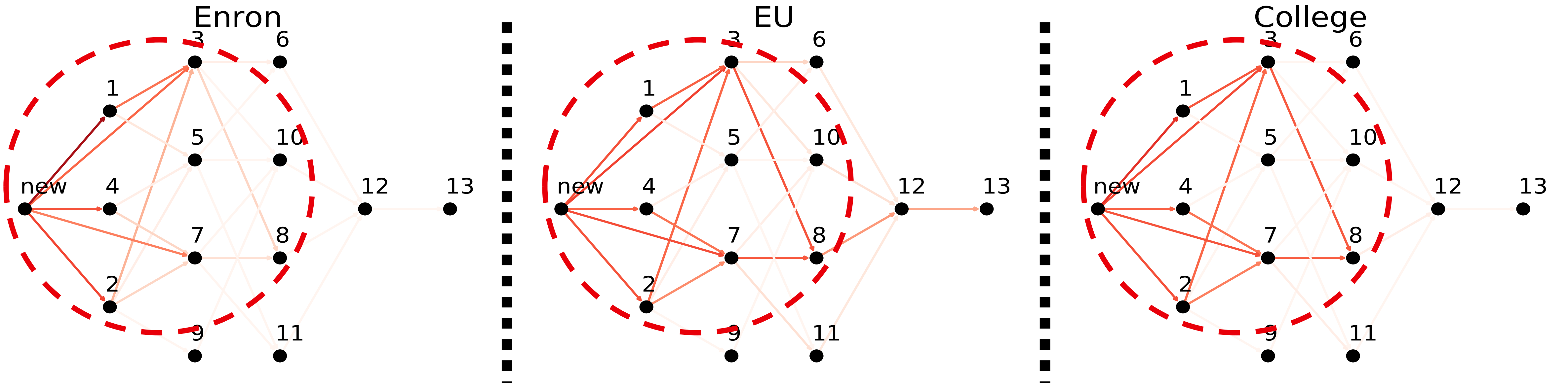
Graphlets over Time: Observations

- We can observe domain-based patterns in GTGs
- Graphs from the same domain share similar edge-weight distributions
- *Example: online Q/A networks*



Graphlets over Time: Observations

- We can observe domain-based patterns in GTGs
- Graphs from the same domain share similar edge-weight distributions
- *Example: email/message networks*



Characterization Profile: Motivation

- How can we compare graphlet-transition patterns numerically?
- Can we compare them for graphs with different scales?
- Idea: to measure the significance of graphlet transitions

dataset	$\# \, nodes$	$\# \, edges$
HepPh	18,477	136,190
Patent	3,774,362	16,512,782

Characterization Profile: Definition

- Significance of a graph transition i is defined as:

$$SP_i := \frac{w_i - \widetilde{w_i}}{w_i + \widetilde{w_i} + \epsilon}$$

  - w_i: occurrences of i in a given edge stream
  - $\widetilde{w_i}$: (expected) occurrences of i in a randomized edge stream
  - ϵ: a small constant
- $SP_i > 0$: the transition i is more frequent than expectation
- $SP_i < 0$: the transition i is less frequent than expectation

Characterization Profile: Definition

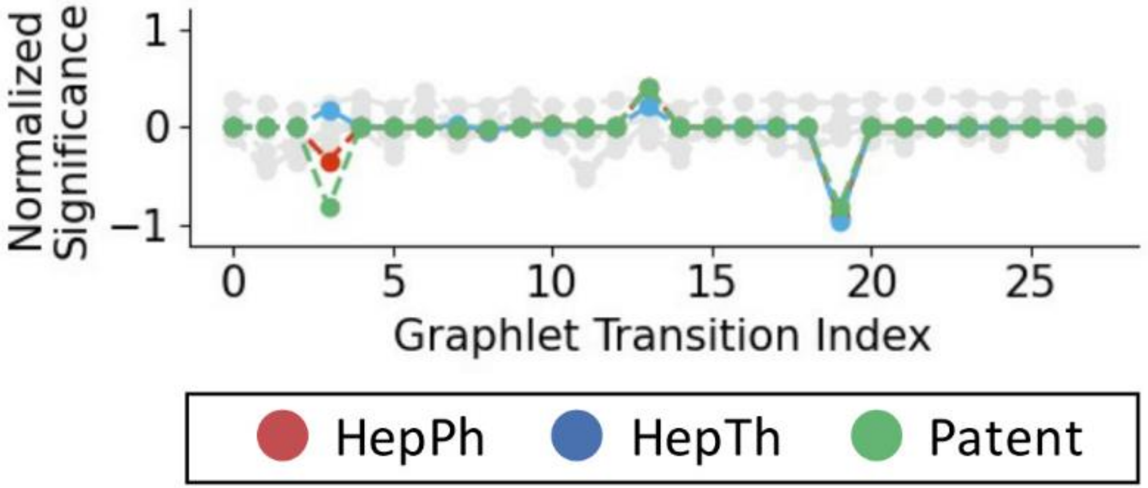
- Characteristic profile (CP) of an edge stream is the normalized significance vector of all transitions

$$[CP_1, CP_2, \ldots, CP_{28}], \qquad where \quad CP_i \coloneqq \frac{SP_i}{\sqrt{\sum_{i=1}^{|E|} SP_i^2}}$$

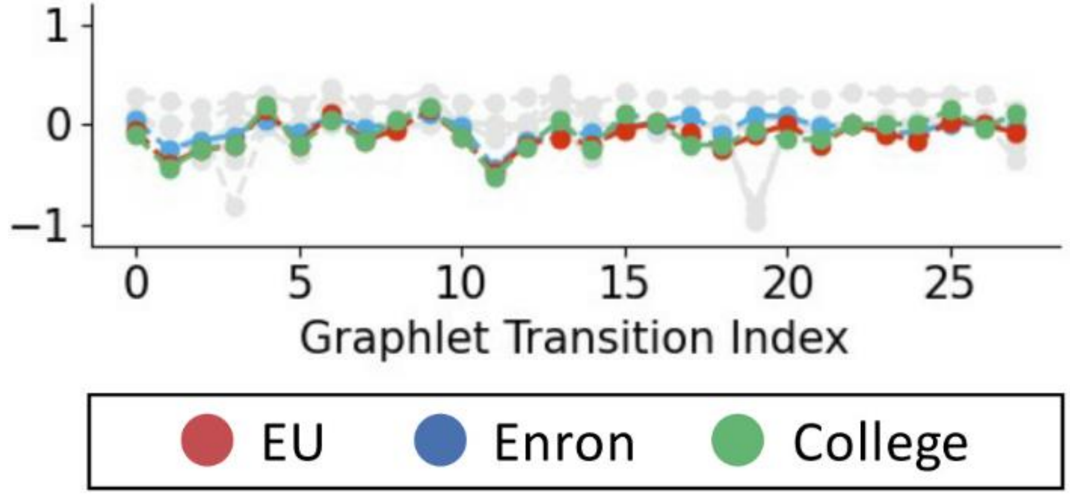
- CP is a 28-diemensional vector that summarizes the evolutionary pattern of an edge stream

Characterization Profile: Example

- We can observe domain-based patterns in GTGs
- *Example: citations networks* **vs** *email/messaging networks*

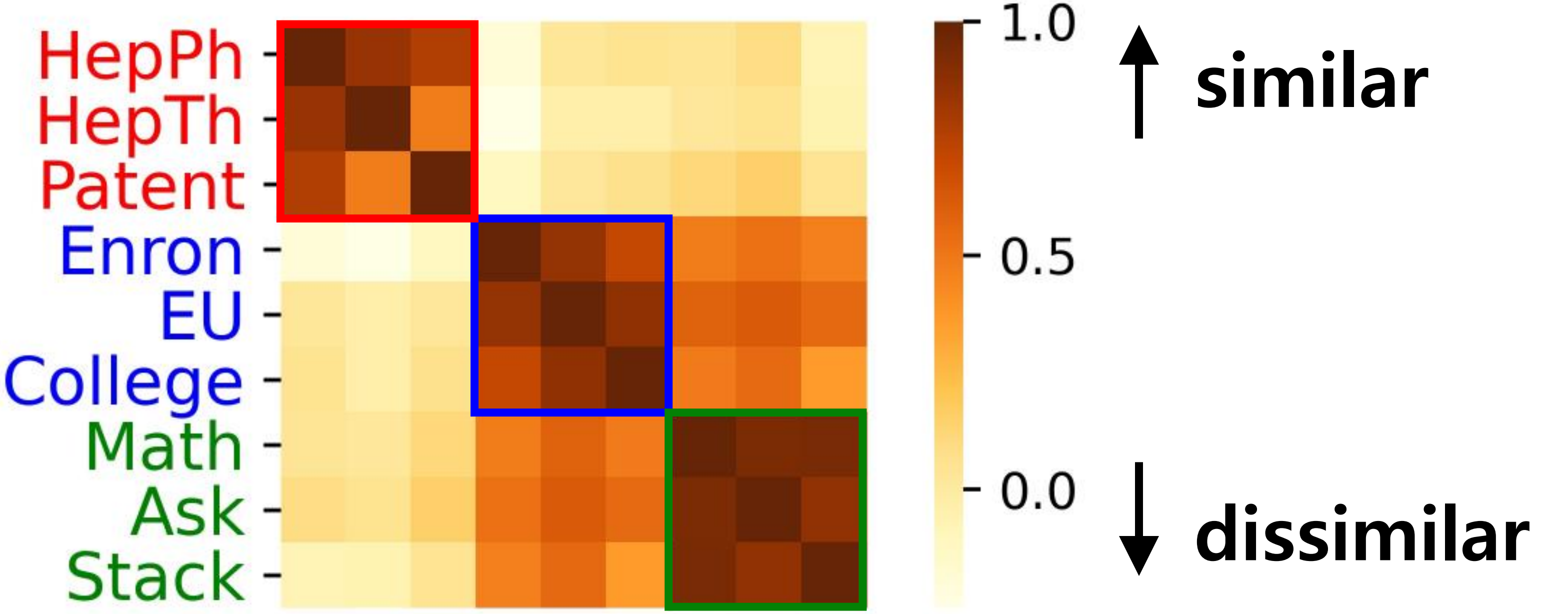


vs



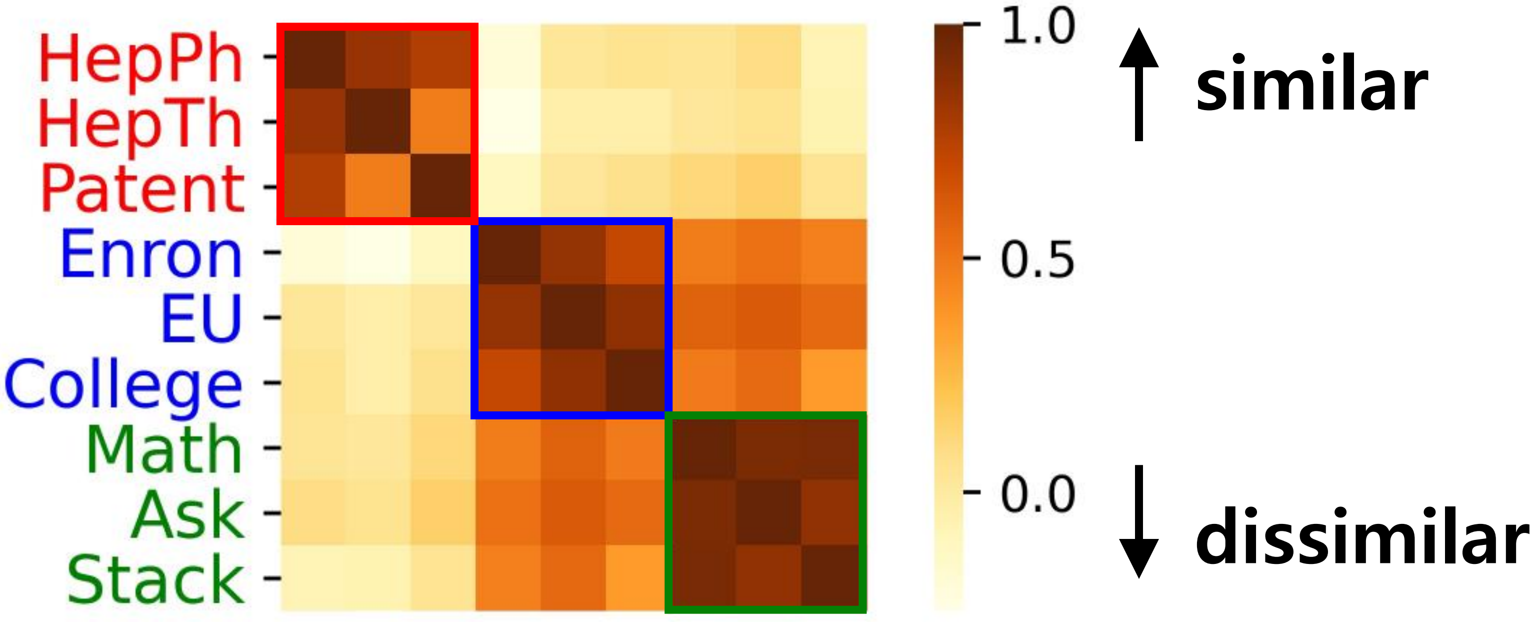
Characterization Profiles: Numerical Similarity

- We measure the numerical similarity between two edge streams using the correlation coefficients between their CPs
- Similarity is higher within each domain and lower across domains



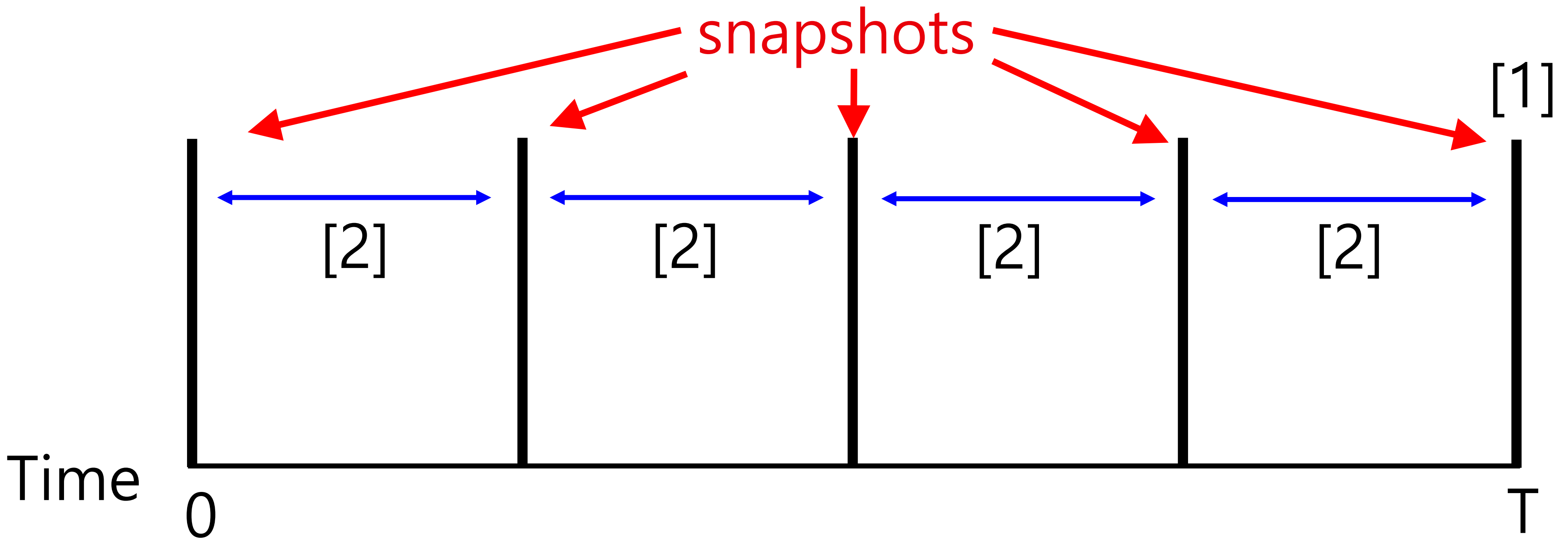
Characterization Profiles: Numerical Similarity

- How well do our CPs characterize evolutionary patterns?
- We need a comparison with competitors



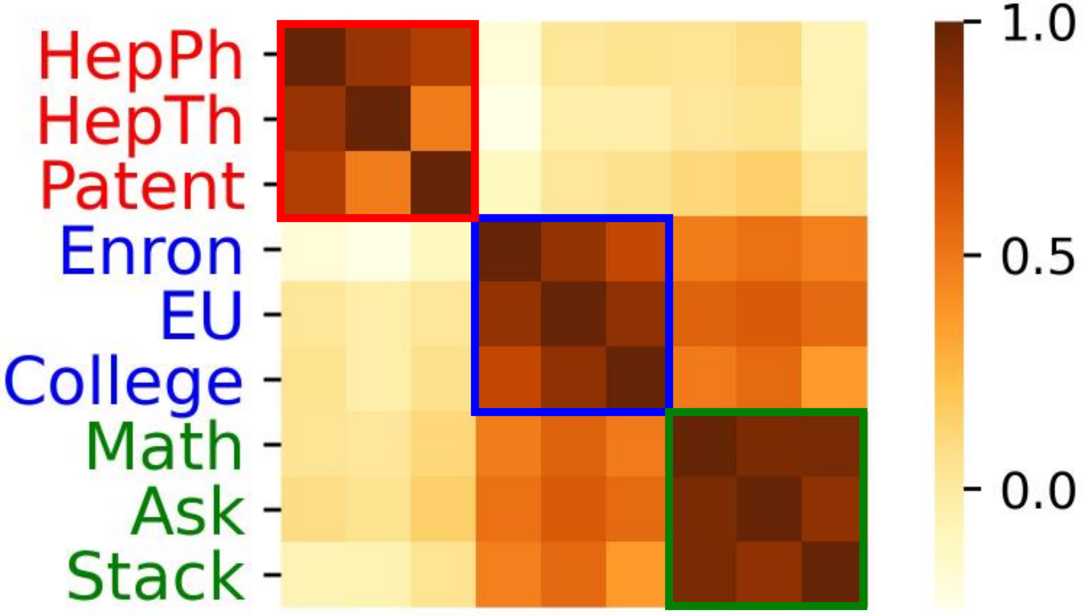
Comparison with Competitors

- Competitors compute CPs based on the following statistics
  - Graphlet count [1]: count of graphlet instances in the last snapshot
  - OTA & GoT [2]: count of transitions between multiple snapshots

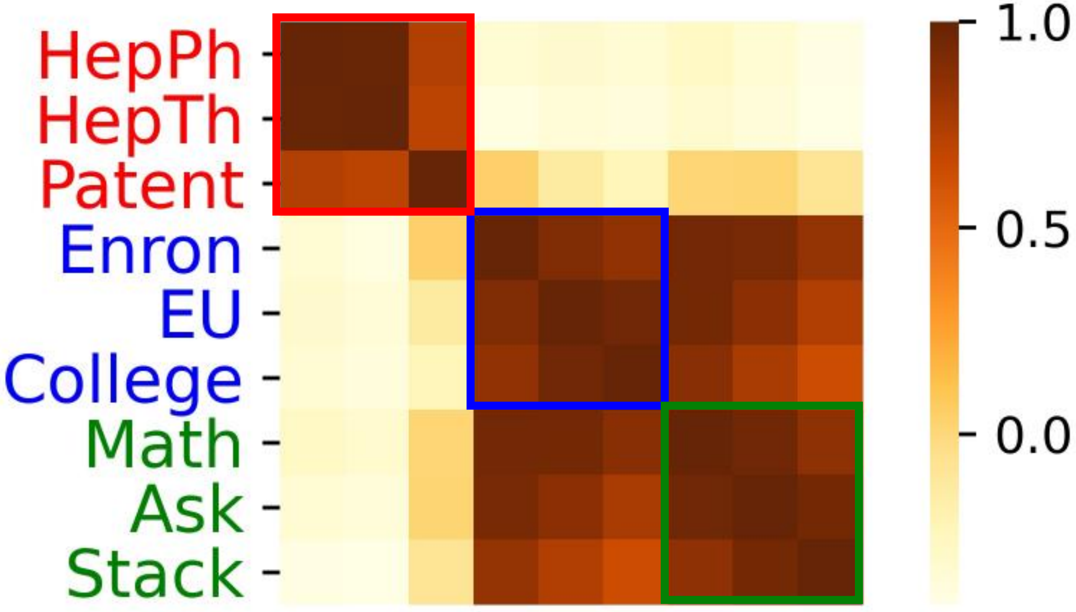


Comparison with Competitors: Result

- Domain-based similarity was less clear in the competitors



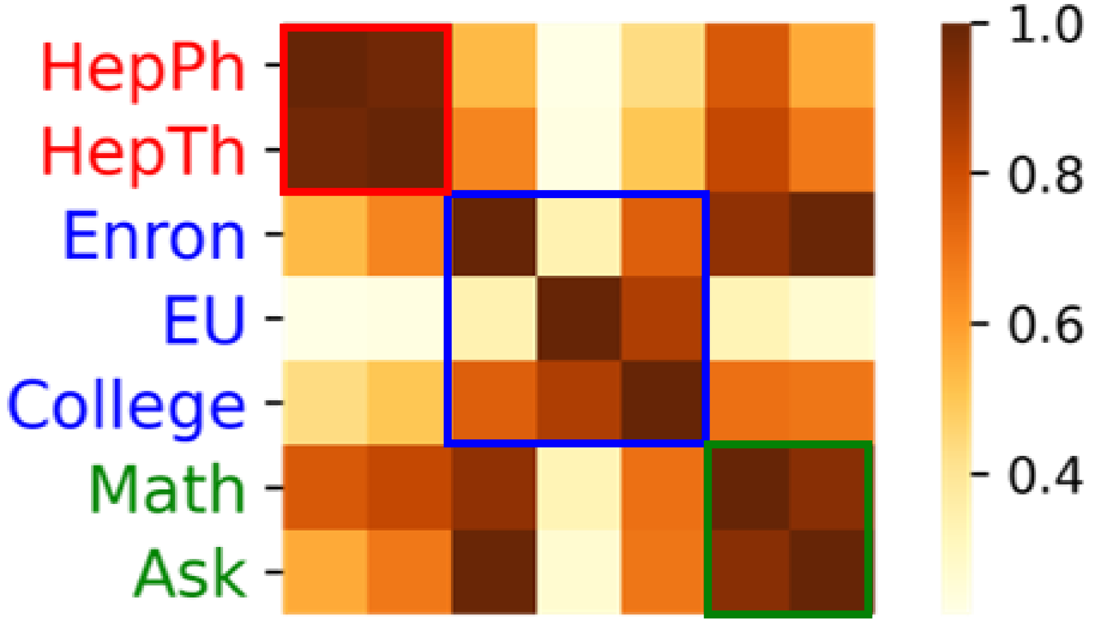
Ours



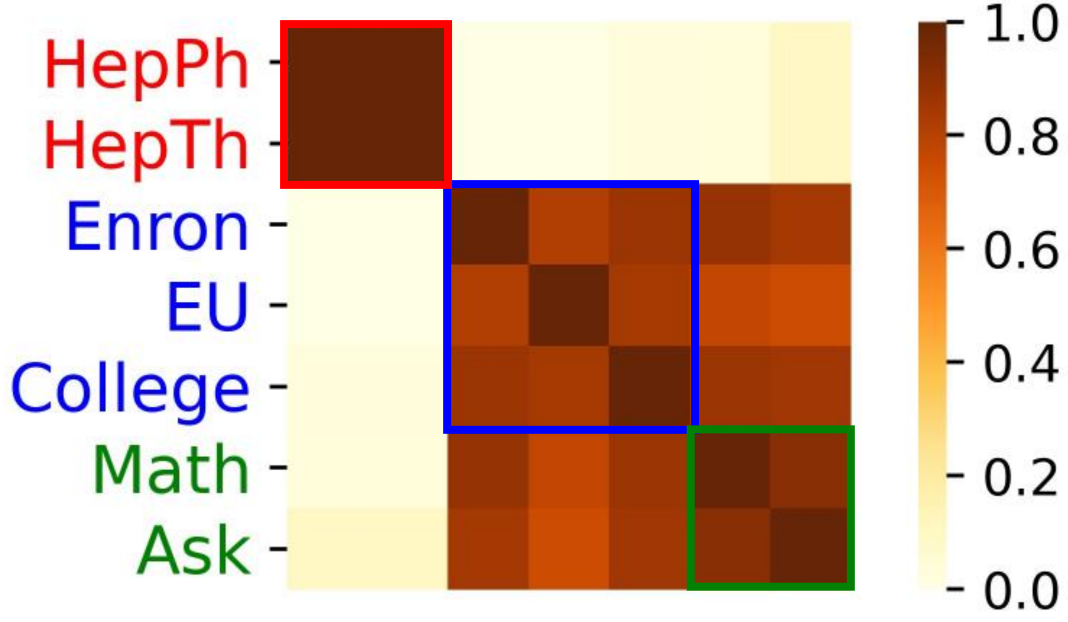
Graphlet count [1]

Comparison with Competitors: Result

- Domain-based similarity was less clear in the competitors
- GoT [2] and OTA [2] run out of memory on large-scale streams



GoT [2]



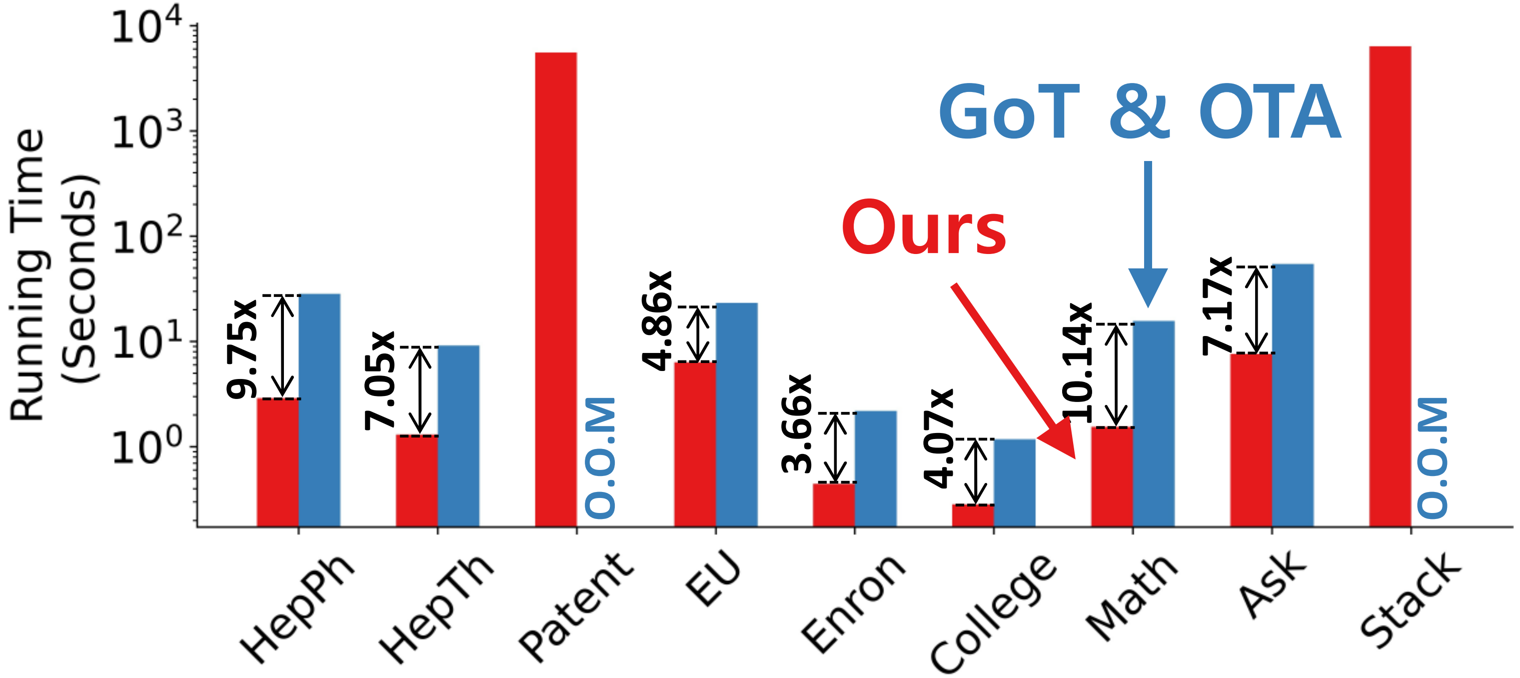
OTA [2]

Comparison with Competitors: Result

- Consider a binary classification problem:
  - **Given:** two edge streams
  - **Classify:** whether they belong to the same domain or not
- With the best similarity threshold:
  - Ours correctly classifies **97.2%** of the pairs
  - Graphlet count [1] , GoT [2], and OTA [3] correctly classify 83.3%, 81.0, 85.7% of the pairs, respectively

Comparison with Competitors: Result

- Ours is up to 10 times faster than GoT [2] and OTA [2]
  - GoT and OTA involve duplicated counting in multiple snapshots
  - They also need to store graphlet instances and track their changes



Conclusions

- **Research questions:** *"How do real-world graphs evolve over time?"*
- **Method**: to examine the changes of graphlet instances over time
- **Outcome**: patterns (e.g., domain-based similarity) & analysis tools (e.g., graph transition graphs)

https://github.com/deukryeol-yoon/graphlets-over-time