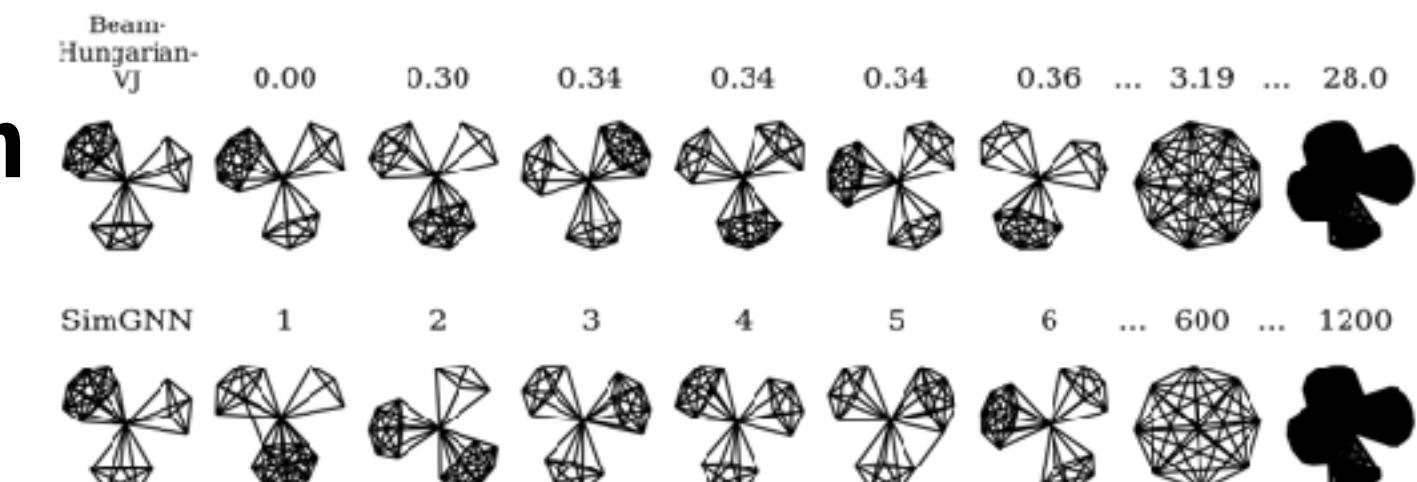
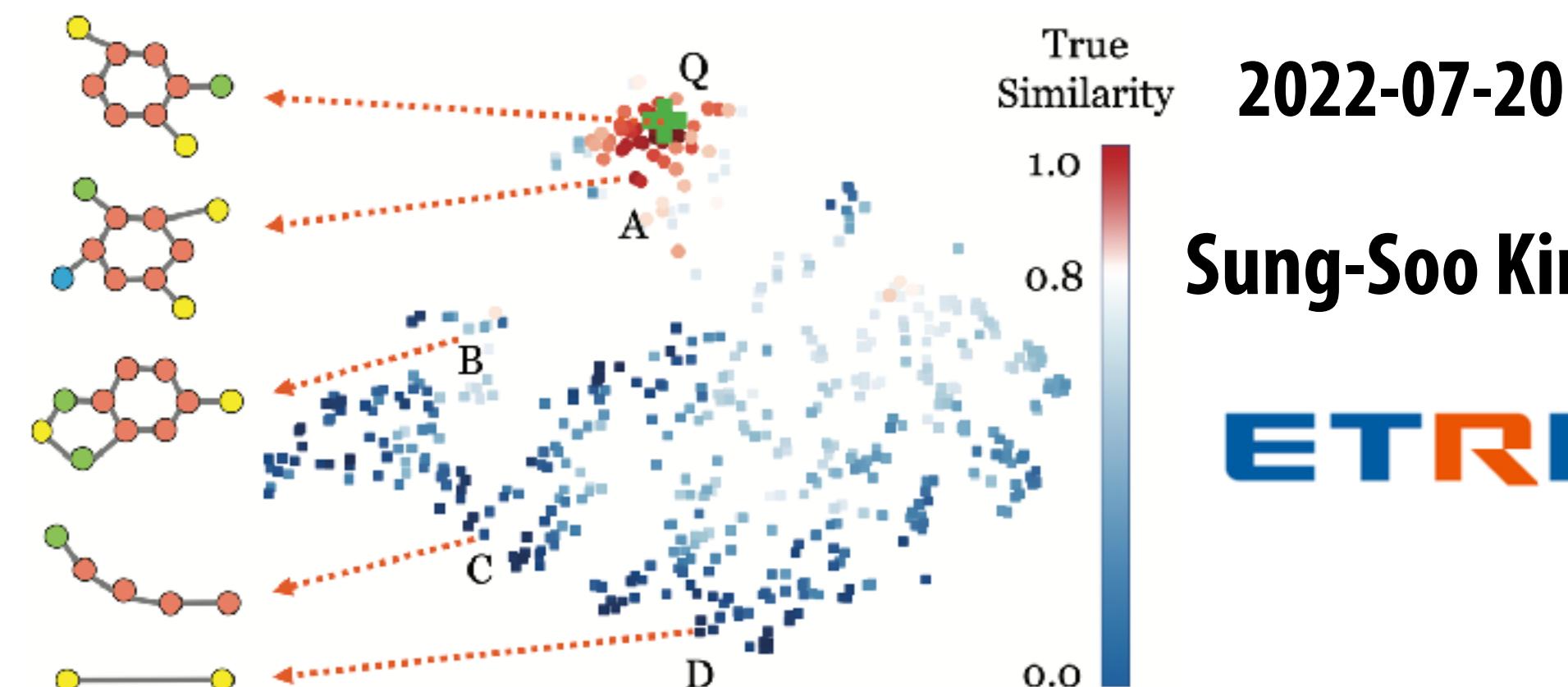


Paper Analysis related to Graph Similarity Search

SimGNN: A Neural Network Approach to Fast Graph Similarity Computation



Outline

- Introduction
- SimGNN: A Neural Network Approach to Fast Graph Similarity Computation
- Summary

SimGNN: A Neural Network Approach to Fast Graph Similarity Computation

Yunsheng Bai¹, Hao Ding¹, Song Bian³, Ting Chen¹, Yizhou Sun¹, Wei Wang¹
¹University of California, Los Angeles, CA, USA
²Purdue University, IN, USA
³Zhejiang University, China
yba@ucl.edu, ding209@purdue.edu, bianscnghts@gmail.com, [tingchen,yzsun,weiwang]@cs.ucla.edu

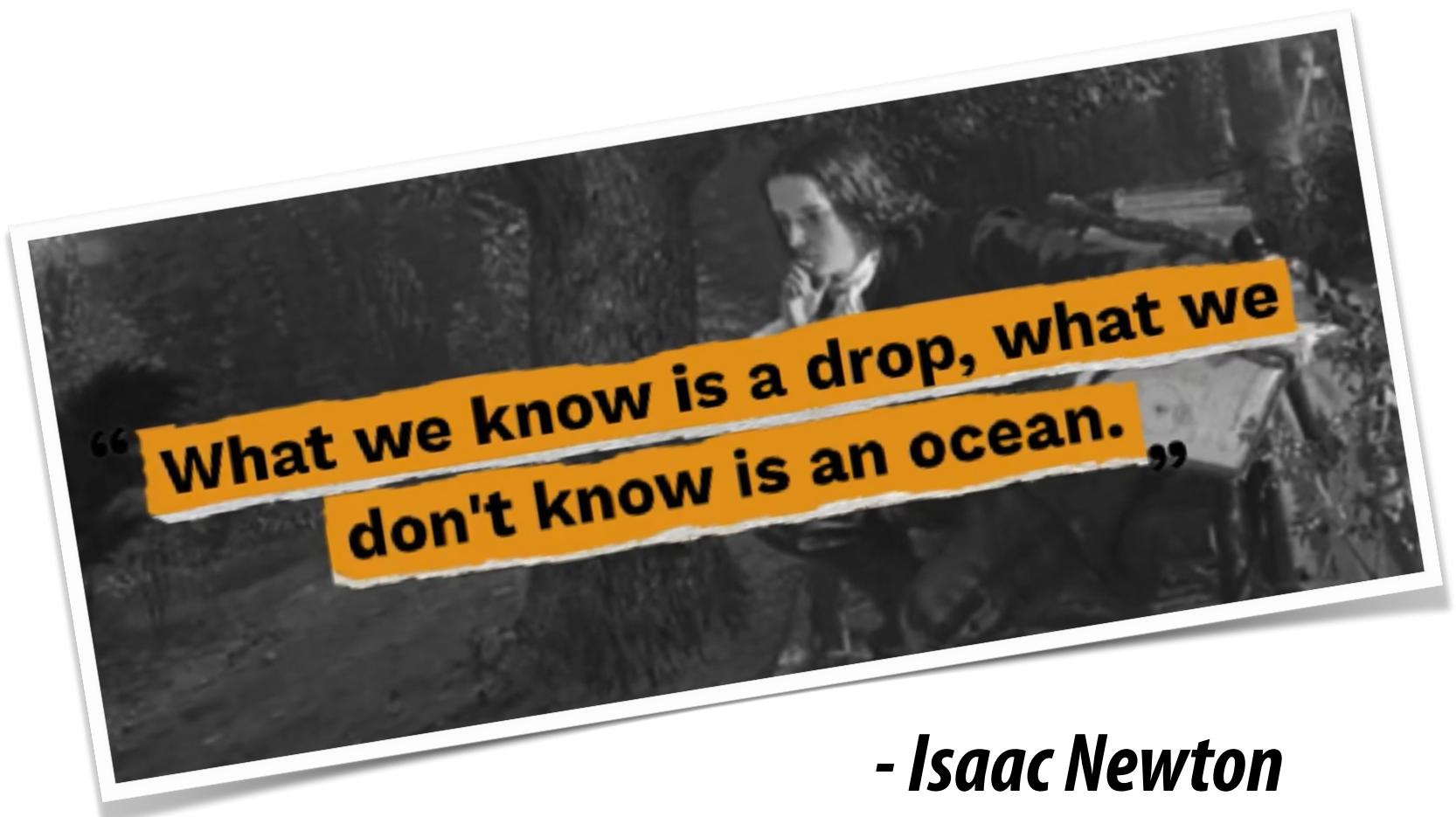
ABSTRACT
Graph similarity search is among the most important graph-based applications, e.g. finding the chemical compounds that are most similar to a query compound. Graph similarity/distance computation, such as Graph Edit Distance (GED) and Maximum Common Subgraph (MCS), is the core operation of graph similarity search and many other applications, but very costly to compute in practice. Inspired by the recent success of neural network approaches to several graph applications, such as node or graph classification, we propose a novel neural network based approach to address this classic yet challenging graph problem, aiming to alleviate the computational burden while preserving a good performance.

The proposed approach, called SimGNN, combines two strategies. First, we design a learnable embedding function that maps every graph into an embedding vector, which provides a global summary of a graph. A novel attention mechanism is proposed to emphasize the important nodes with respect to a specific similarity metric. Second, we design a pairwise node comparison method to supplement the graph-level embeddings with fine-grained node-level

KEYWORDS
network embedding, neural networks, graph similarity computation, graph edit distance

ACM Reference Format:
Yunsheng Bai, Hao Ding, Song Bian, Ting Chen, Yizhou Sun, Wei Wang. 2019. SimGNN: A Neural Network Approach to Fast Graph Similarity Computation. In The Twelfth ACM International Conference on Web Search and Data Mining (WSDM'19), February 11–15, 2019, Melbourne, VIC, Australia. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3289400.3289987>

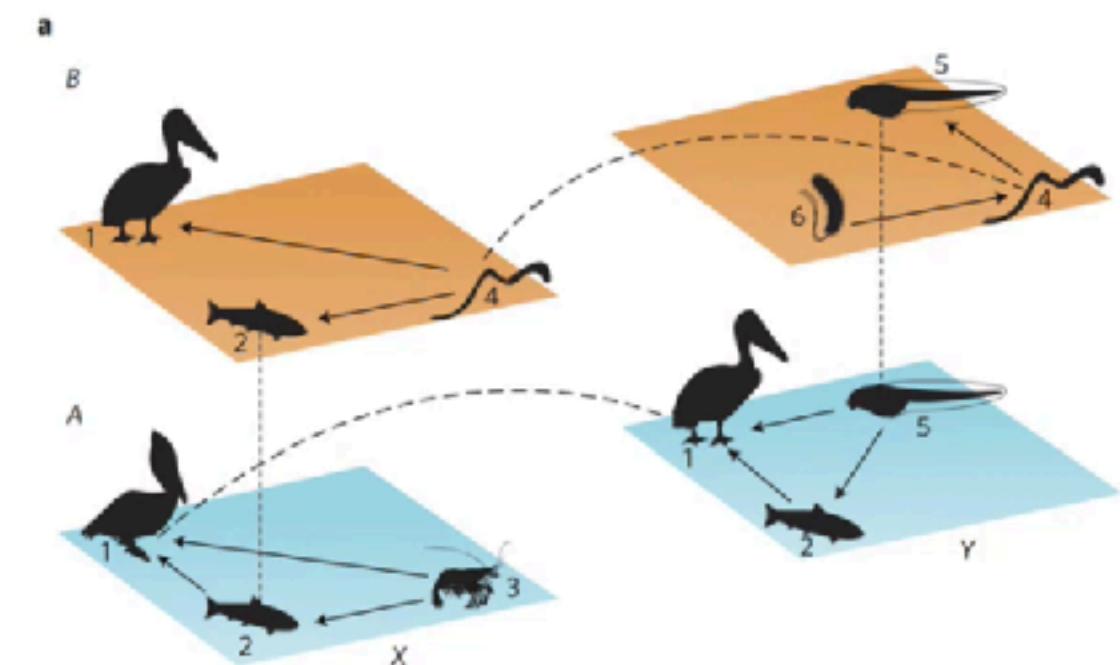
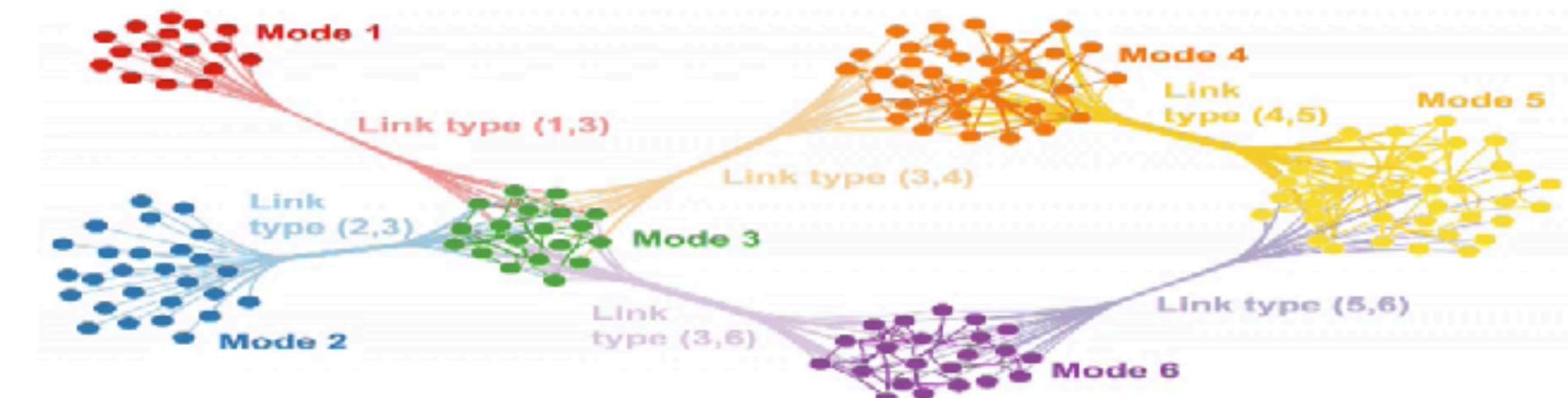
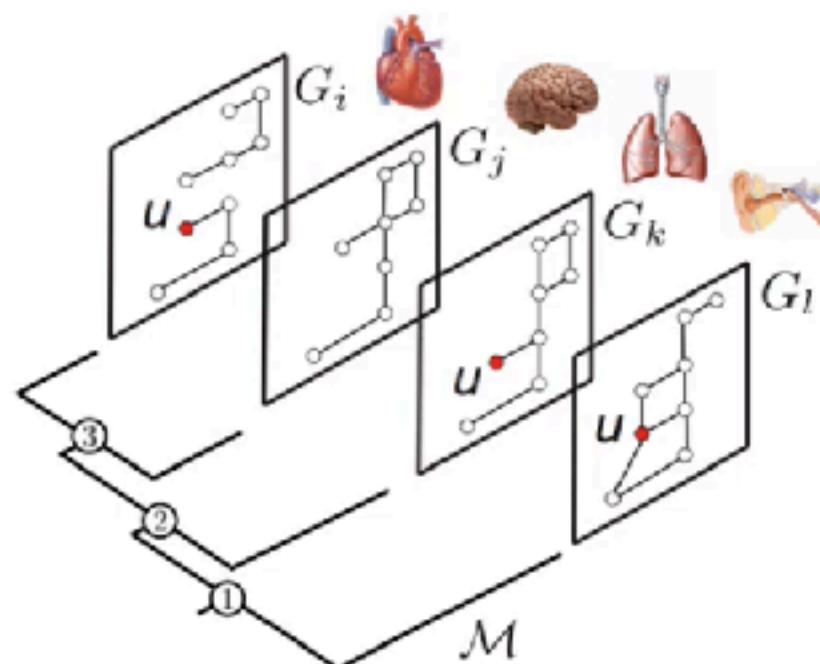
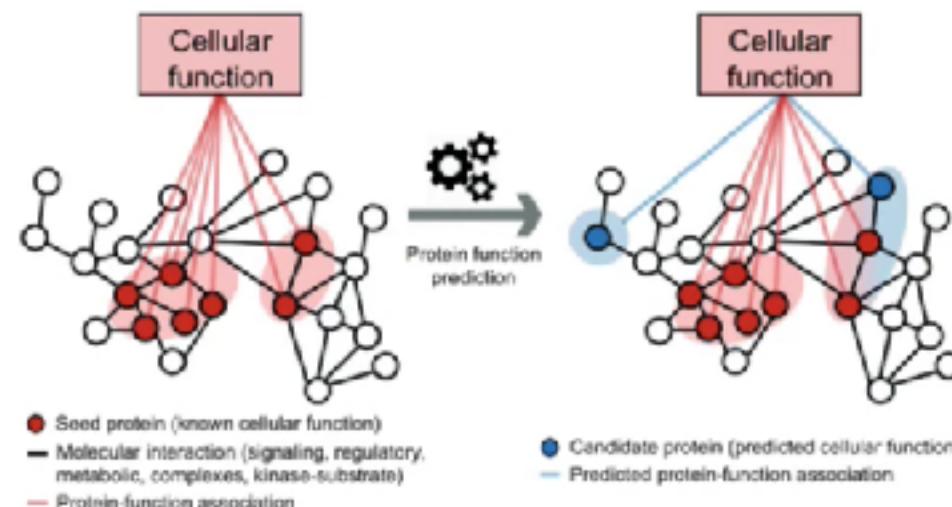
1 INTRODUCTION
Graphs are ubiquitous nowadays and have a wide range of applications in bioinformatics, chemistry, recommender systems, social network study, program static analysis, etc. Among them, one of the fundamental problems is to retrieve a set of similar graphs from a database given a user query. Different graph similarity/distance metrics are defined, such as Graph Edit Distance (GED) [3], Maximum



- Isaac Newton

Graph Applications: Biology

- Biology
- Chemistry
- Social Network Analysis
- Software Analysis
- Circuit Design



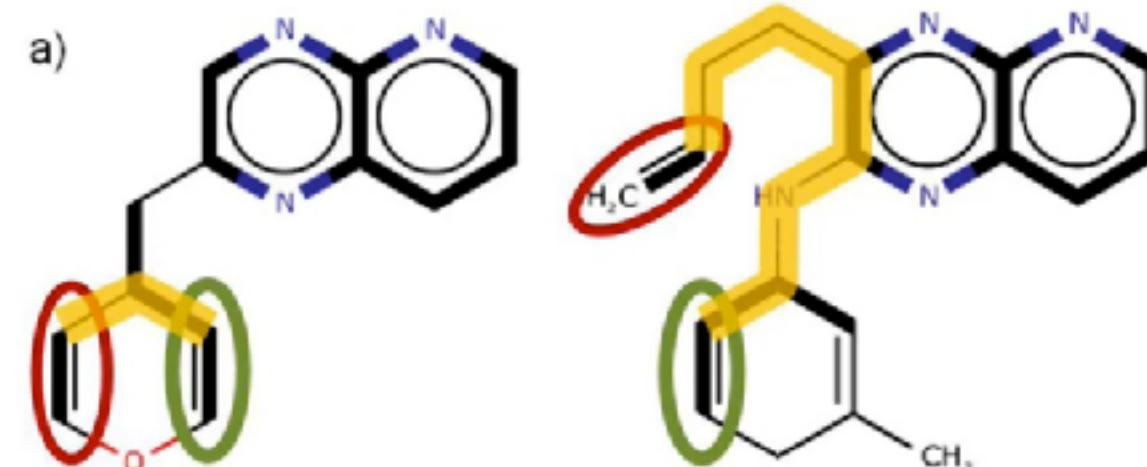
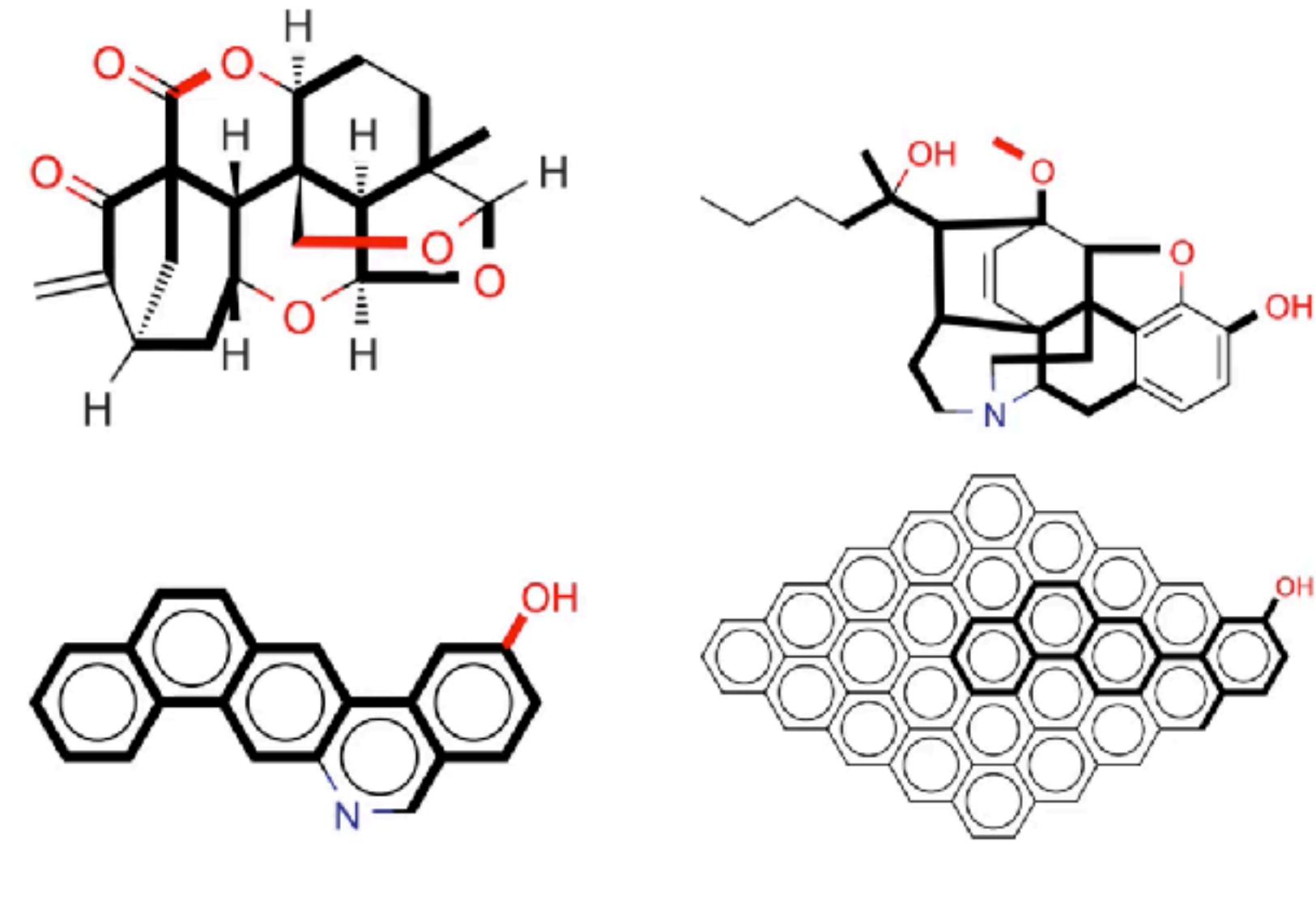
Pilosof, Shai, et al. "The multilayer nature of ecological networks." *Nature Ecology & Evolution* 1.4 (2017): 0101.

Zitnik, Marinka, and Jure Leskovec. "Predicting multicellular function through multi-layer tissue networks." *Bioinformatics* 33.14 (2017): i190-i198.

Zitnik, Marinka, et al. "Machine learning for integrating data in biology and medicine: Principles, practice, and opportunities." *Information Fusion* 50 (2019): 71-91.

Graph Applications: Chemistry

- Biology
- **Chemistry**
- Social Network Analysis
- Software Analysis
- Circuit Design



Duesbury, Edmund. *Applications and Variations of the Maximum Common Subgraph for the Determination of Chemical Similarity*. Diss. University of Sheffield, 2015.

Duesbury, Edmund, John Holliday, and Peter Willett. "Comparison of Maximum Common Subgraph Isomorphism Algorithms for the Alignment of 2D Chemical Structures." *ChemMedChem* 13.6 (2018): 588-598.

Graph Applications: Social Network Analysis

- Biology
- Chemistry
- **Social Network Analysis**
- Software Analysis
- Circuit Design

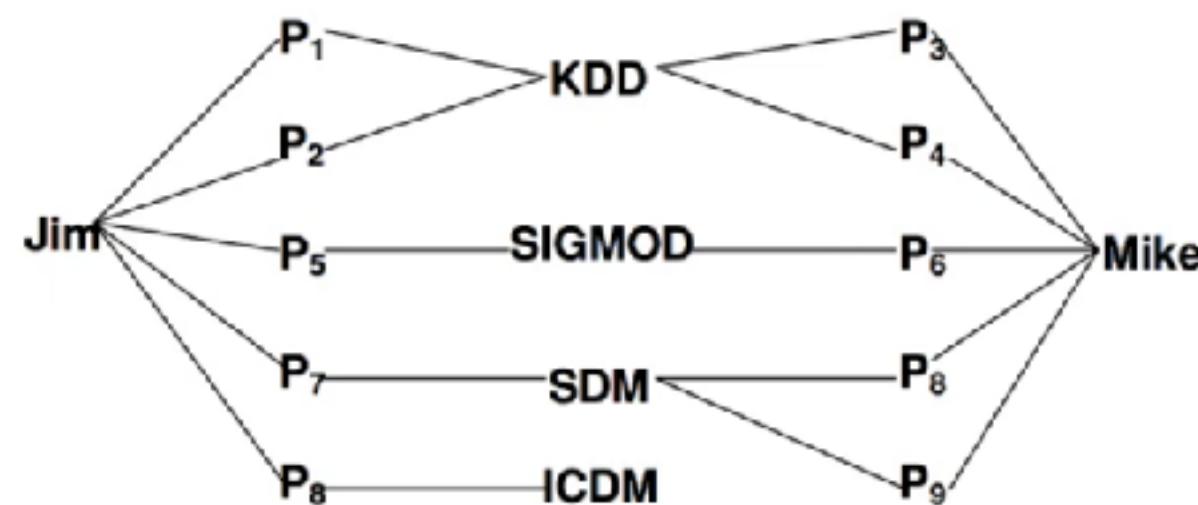
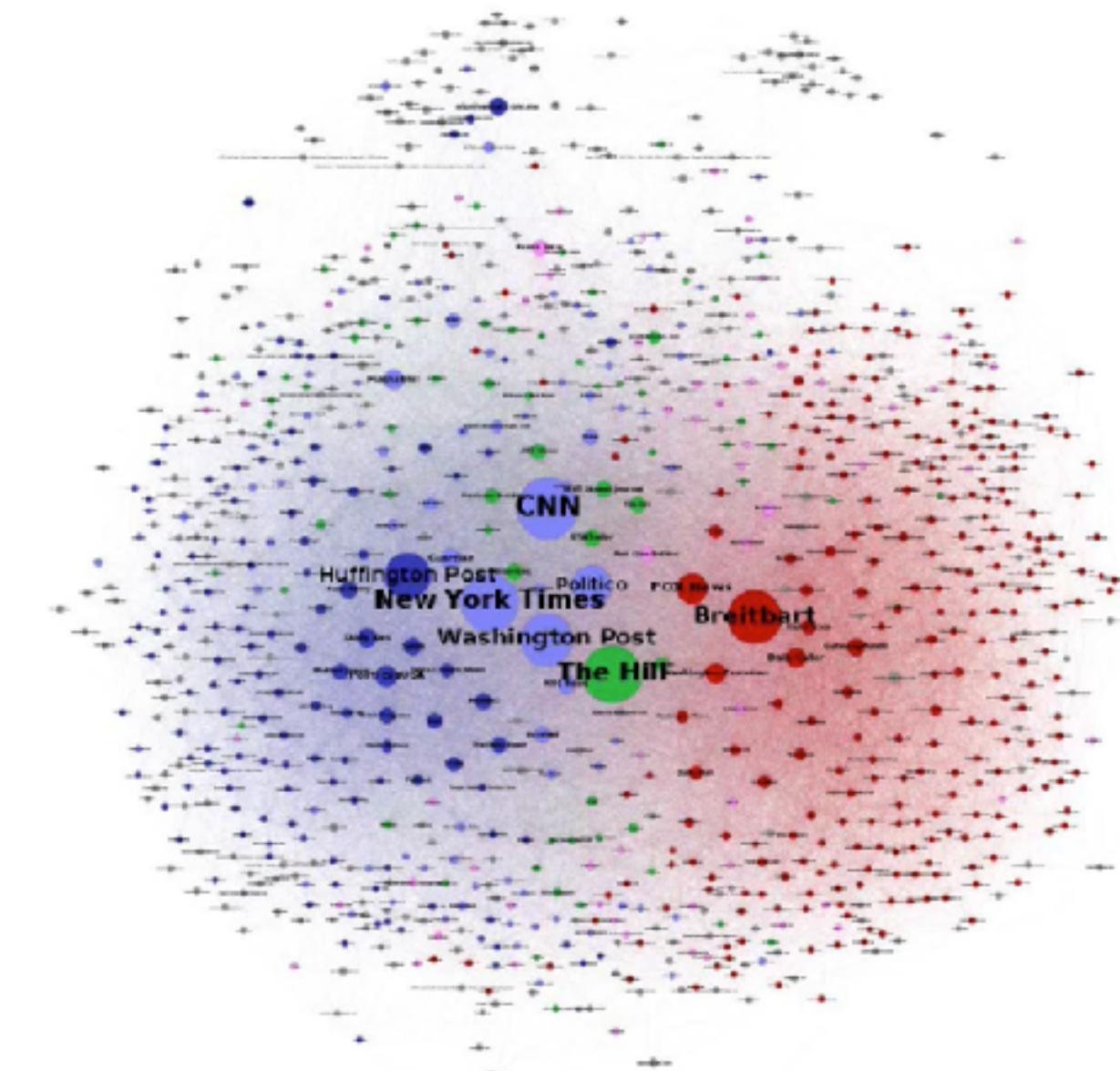


Figure 3. An Example of A-P-V-P-A Paths Between Two Authors



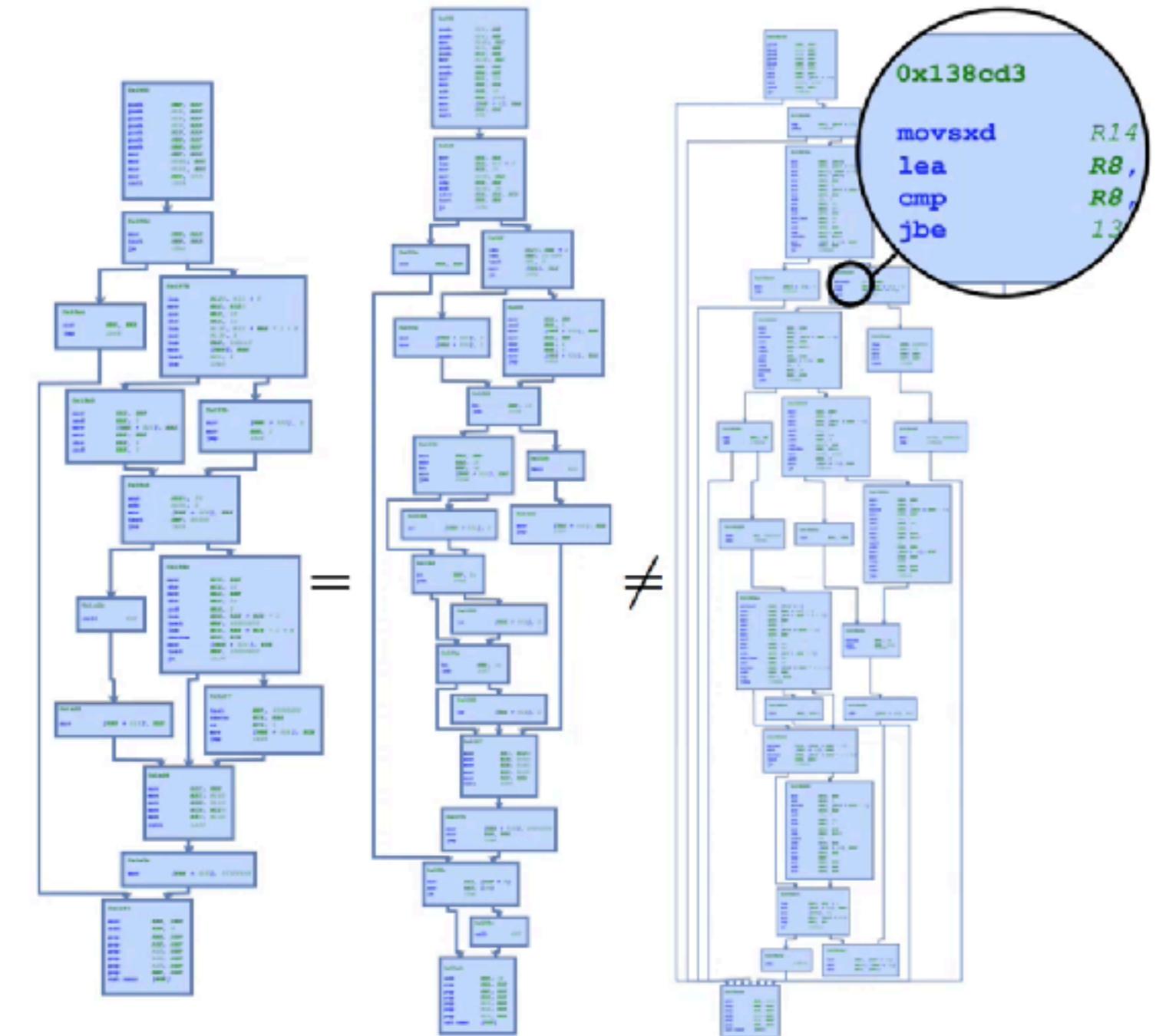
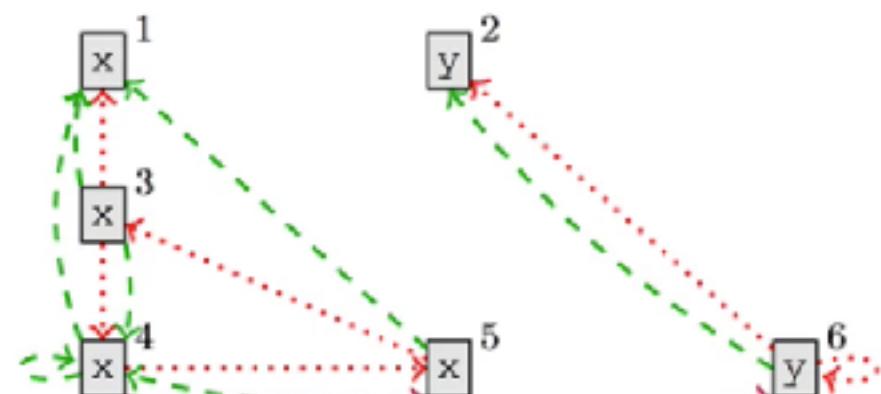
Sun, Yizhou, et al. "Co-author relationship prediction in heterogeneous bibliographic networks." *2011 International Conference on Advances in Social Networks Analysis and Mining*. IEEE, 2011.

<https://cyber.harvard.edu/publications/2017/08/mediacloud>

Graph Applications: Software Analysis

- Biology
- Chemistry
- Social Network Analysis
- **Software Analysis**
- Circuit Design

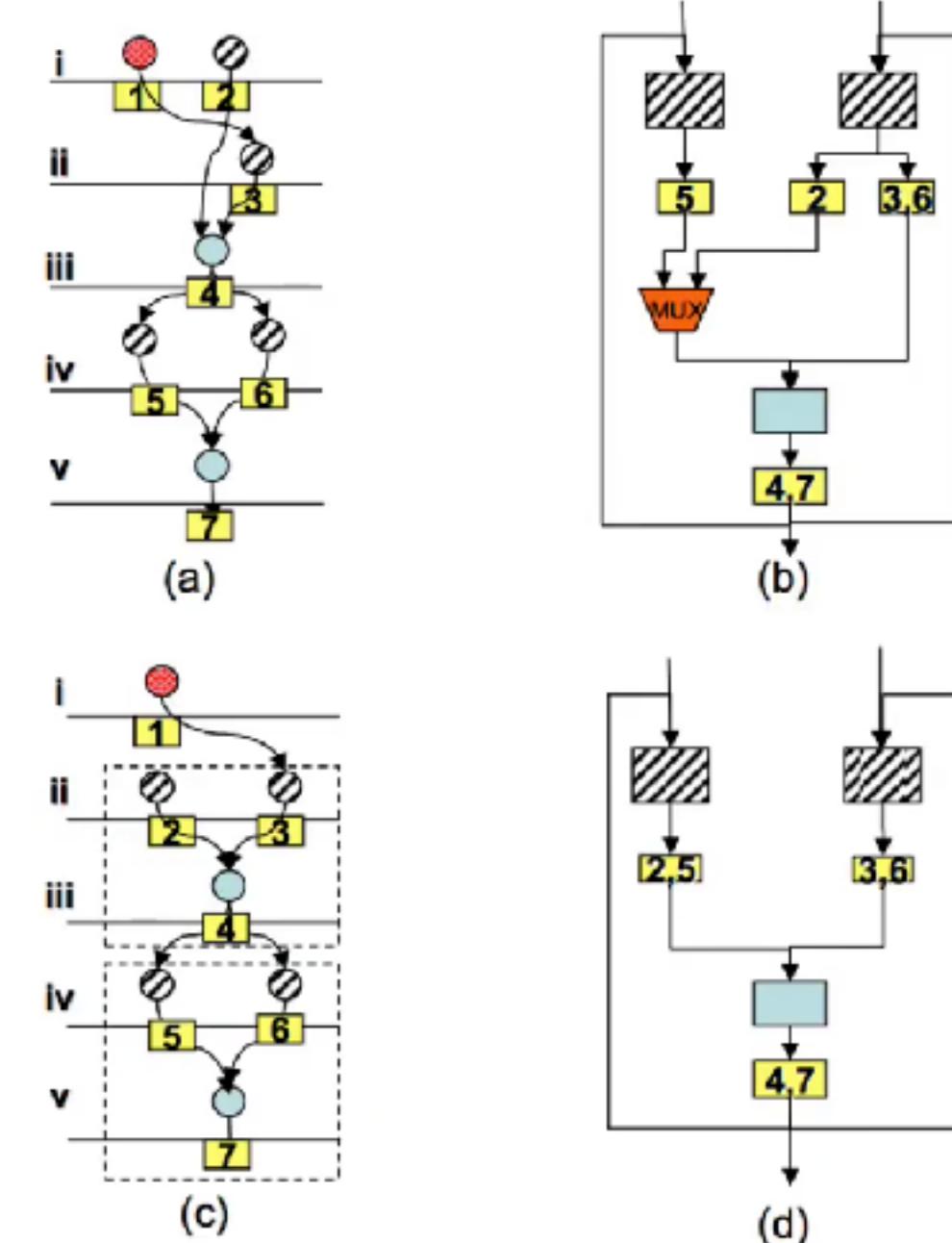
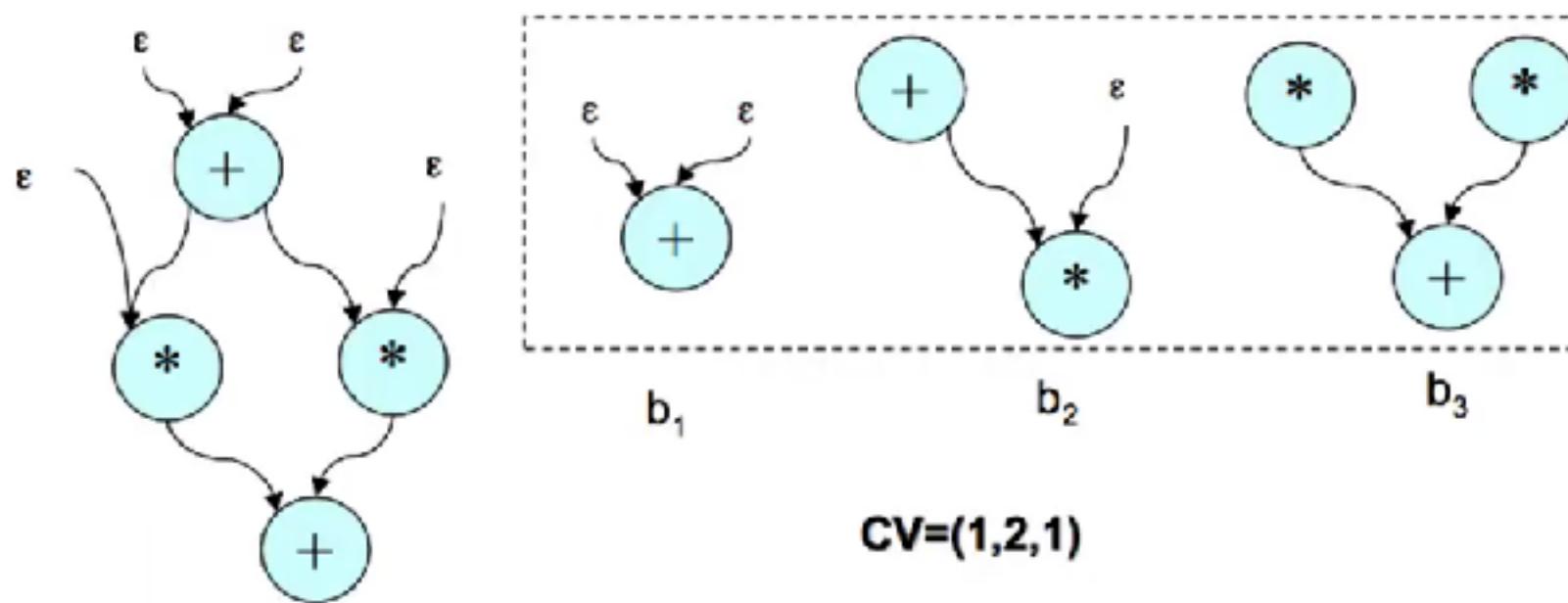
```
(x1,y2) = Foo();  
while (x3 > 0)  
    x4 = x5 + y6
```



Allamanis, Miltiadis, Marc Brockschmidt, and Mahmoud Khademi. "Learning to represent programs with graphs." *ICLR* (2018).
Li, Yujia, et al. "Graph Matching Networks for Learning the Similarity of Graph Structured Objects." *ICML* (2019).

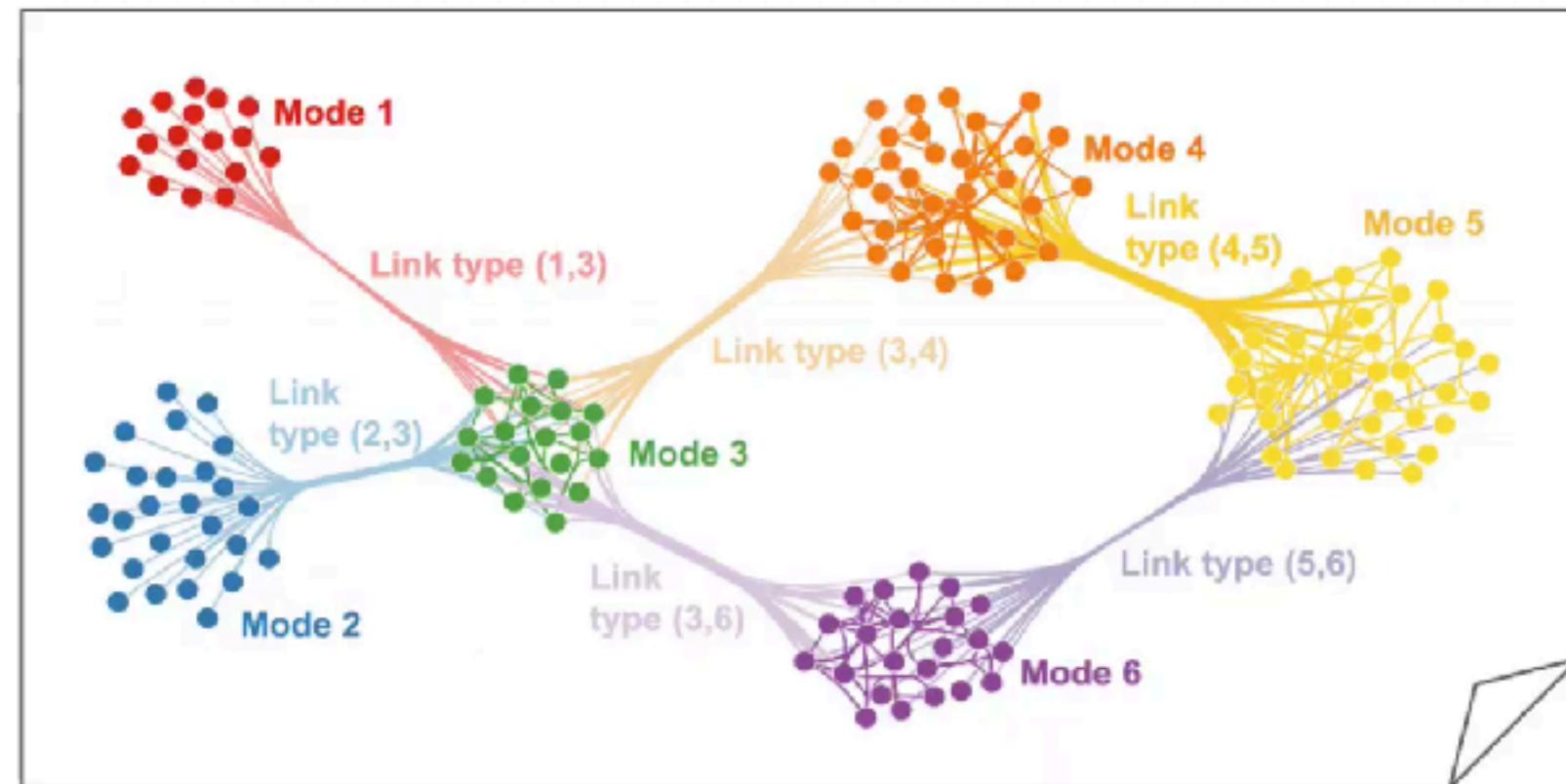
Graph Applications: Circuit Design

- Biology
- Chemistry
- Social Network Analysis
- Software Analysis
- **Circuit Design**

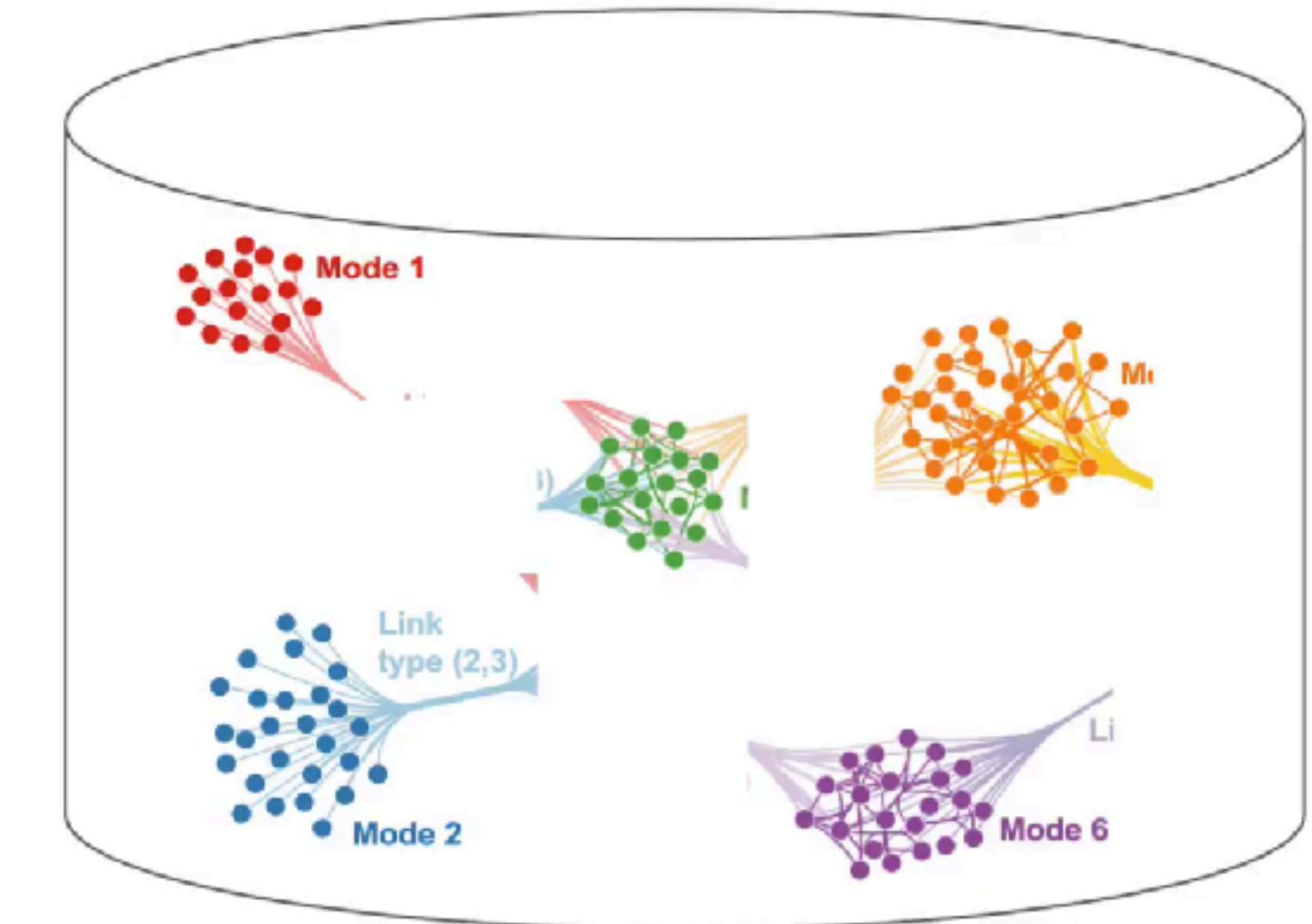


Cong, Jason, and Wei Jiang. "Pattern-based behavior synthesis for FPGA resource reduction." *Proceedings of the 16th international ACM/SIGDA symposium on Field programmable gate arrays*. ACM, 2008.

Applications: Node-Level vs. Graph-Level



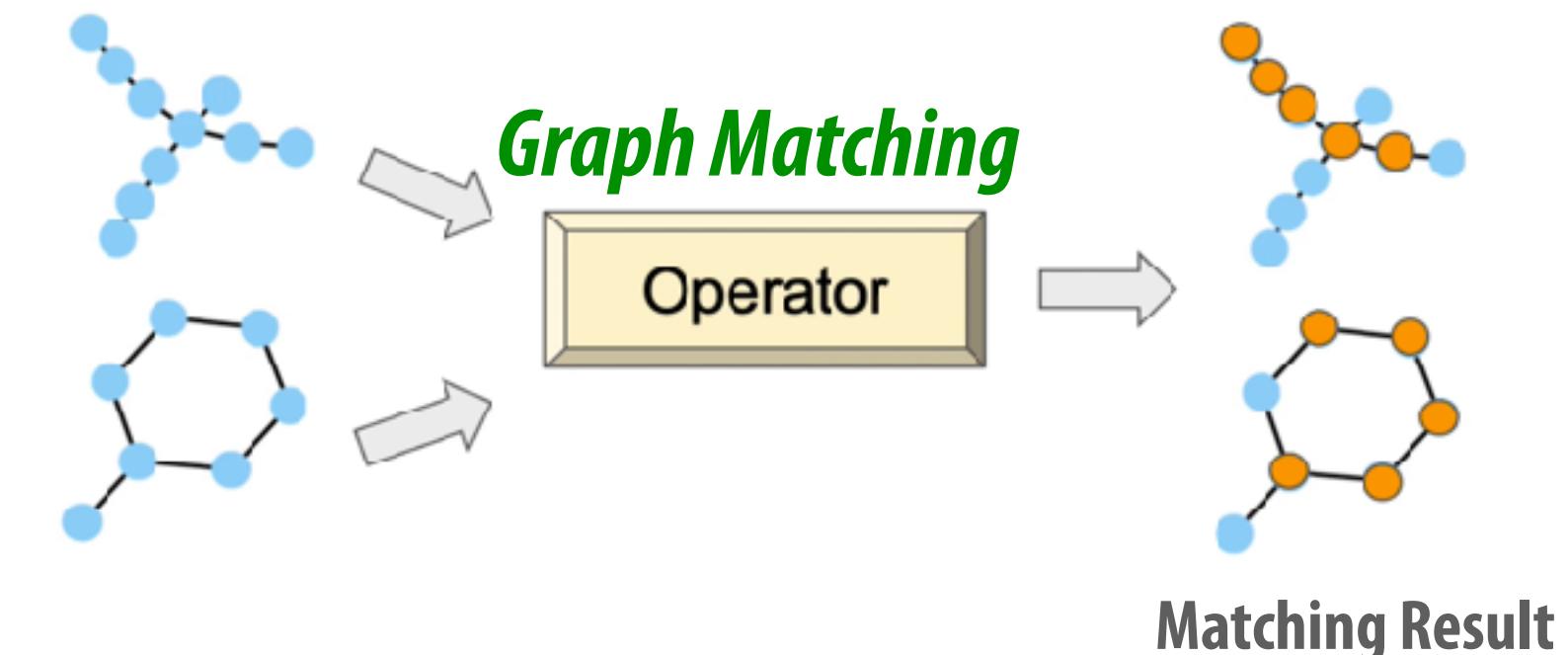
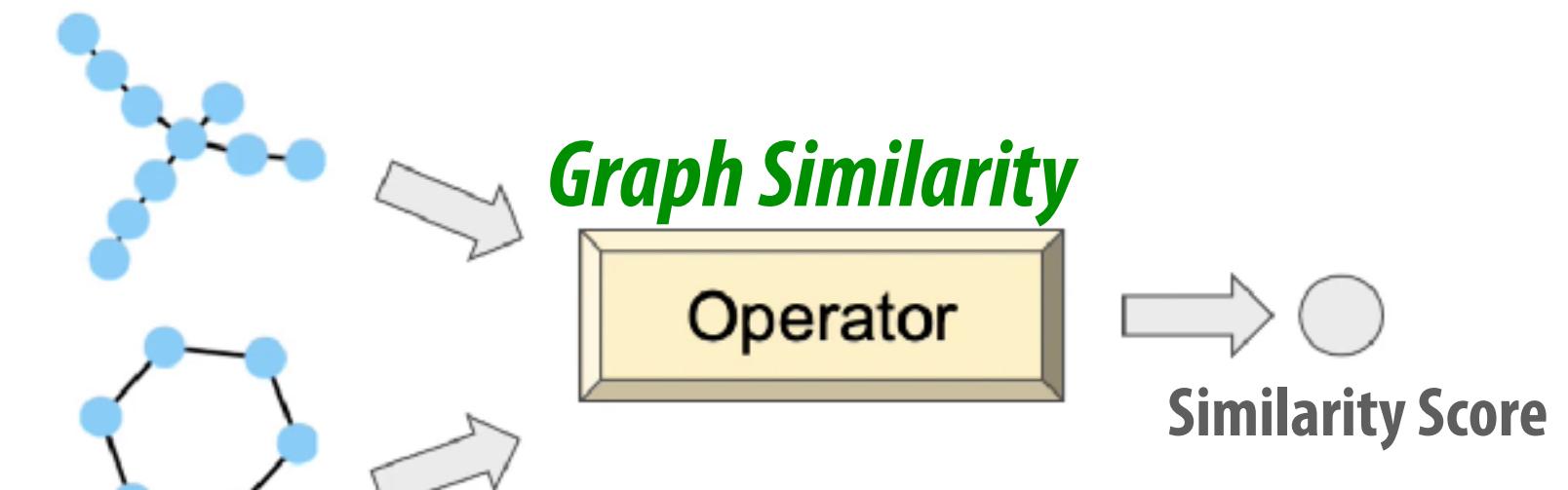
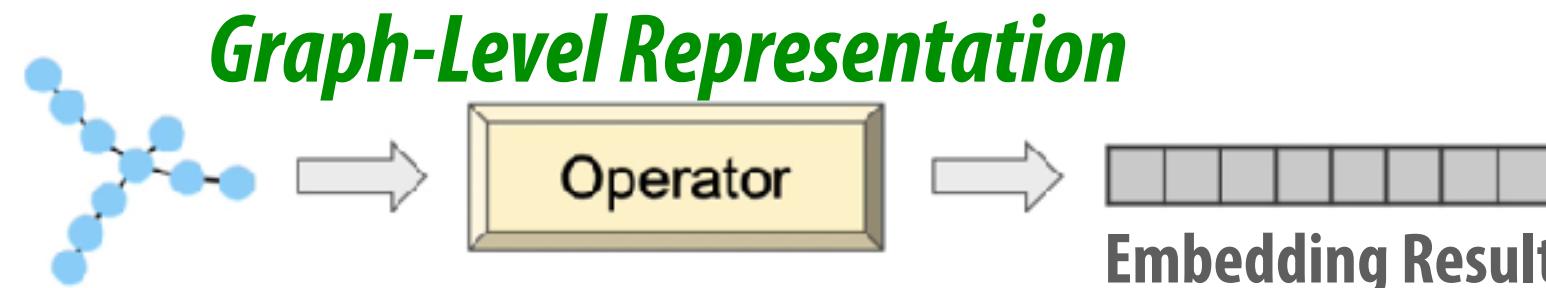
One Single Graph



A Collection of Graphs

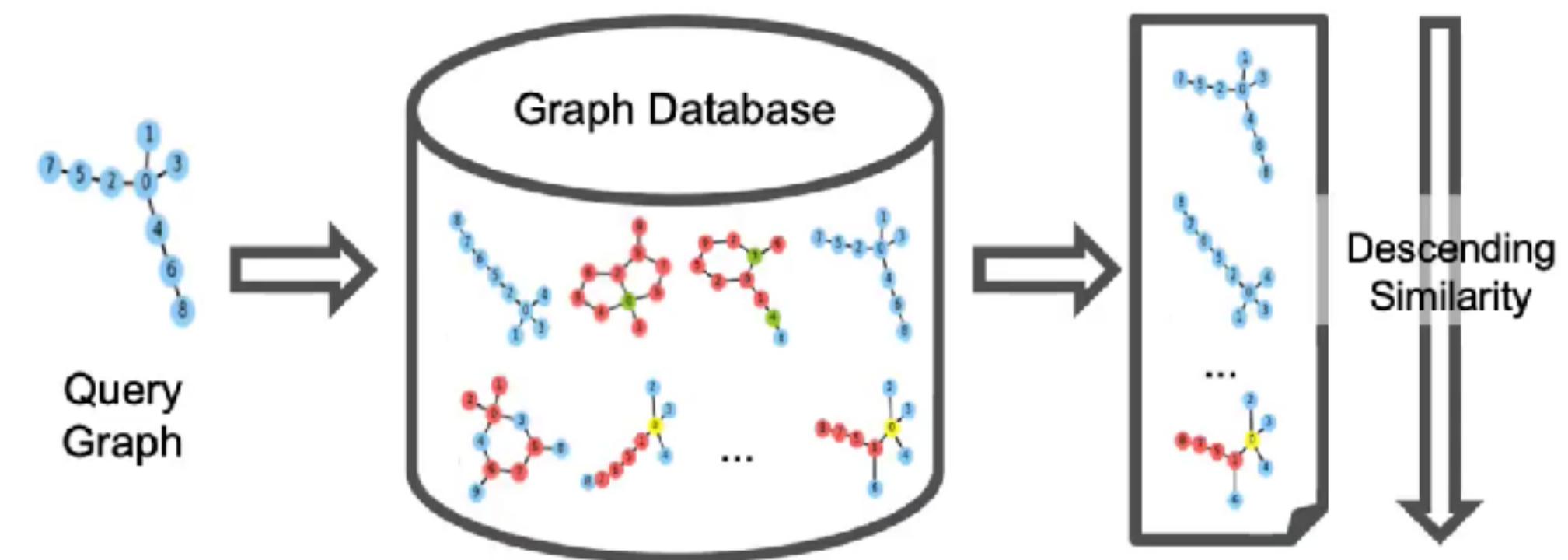
Graph-Level Operators

- Graph Similarity
- Graph Matching
- Graph-Level Representation
- ...



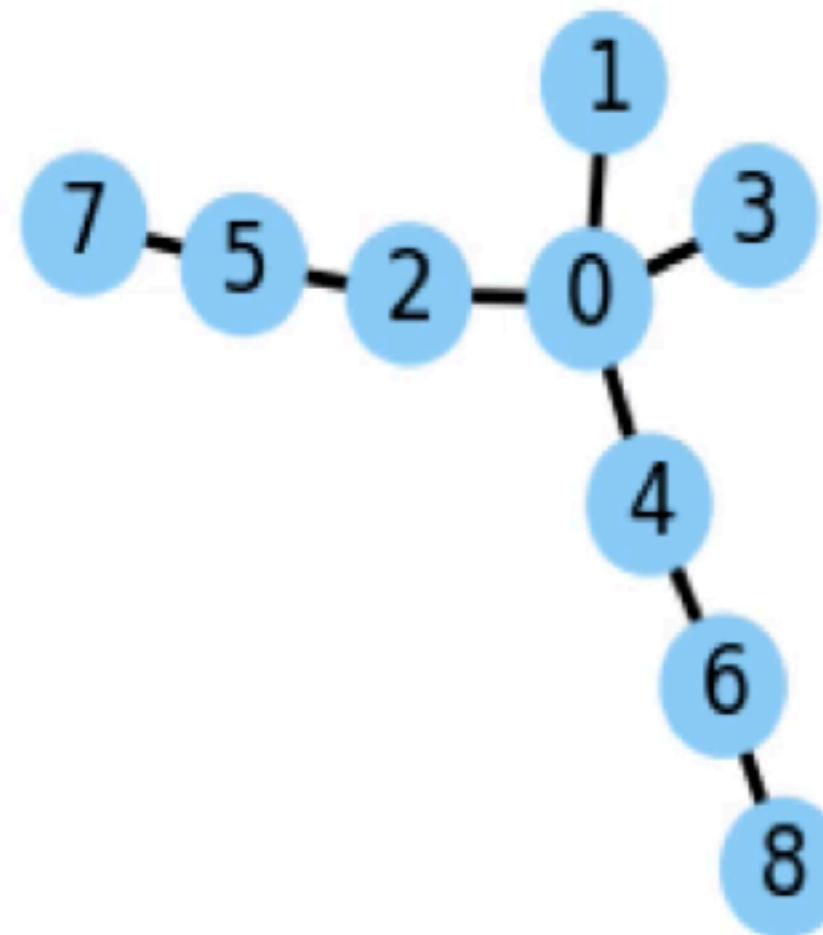
SimGNN: Introduction

- Neural Network Based Approach for Graph Analytics
- Graph-Graph Similarity
 - Applications:
 - Drug design
 - Computer security
 - Social network analysis
 - Anomaly detection
 - Challenging (**NP-hard**)

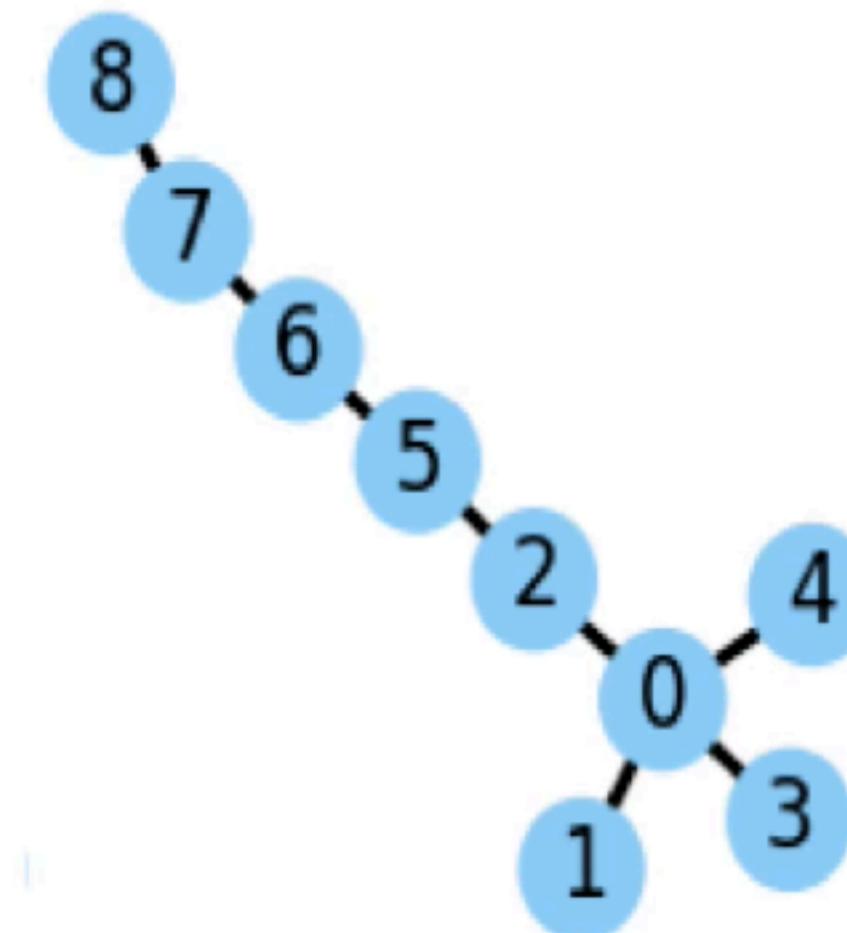


Graph Similarity Computation

\mathcal{G}_1



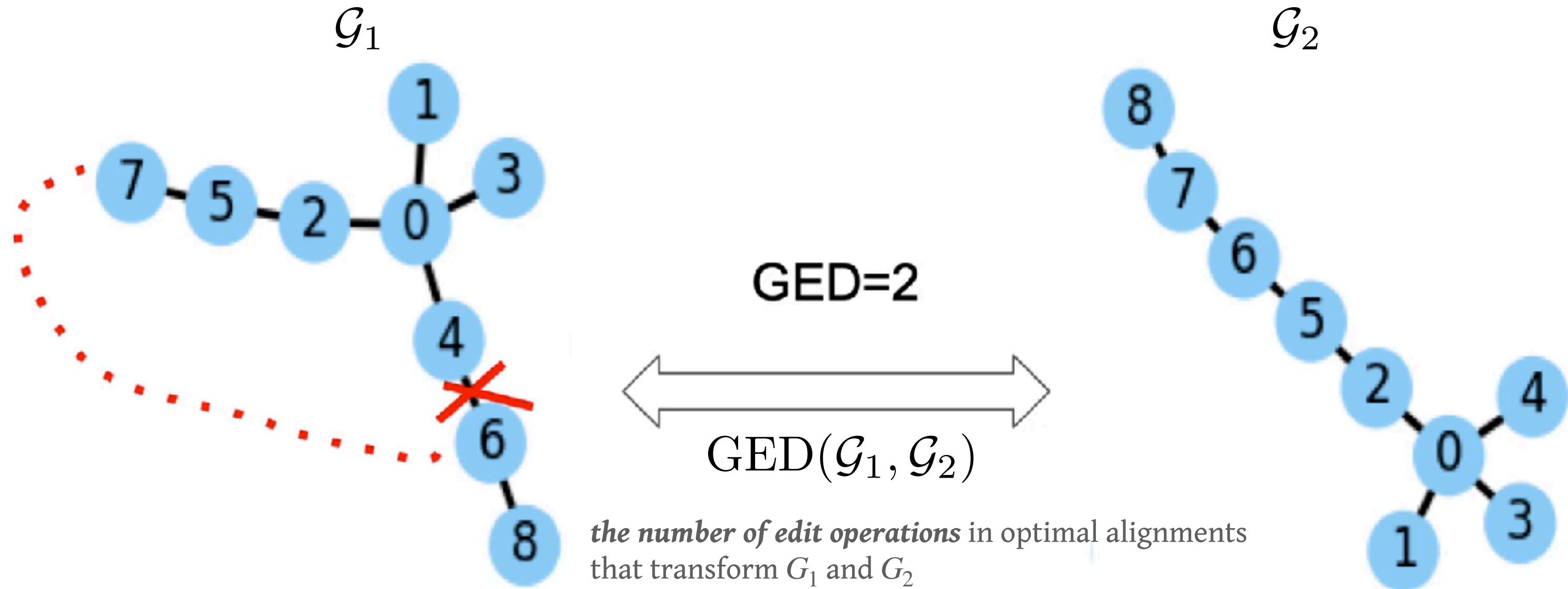
\mathcal{G}_2



How similar are
they?



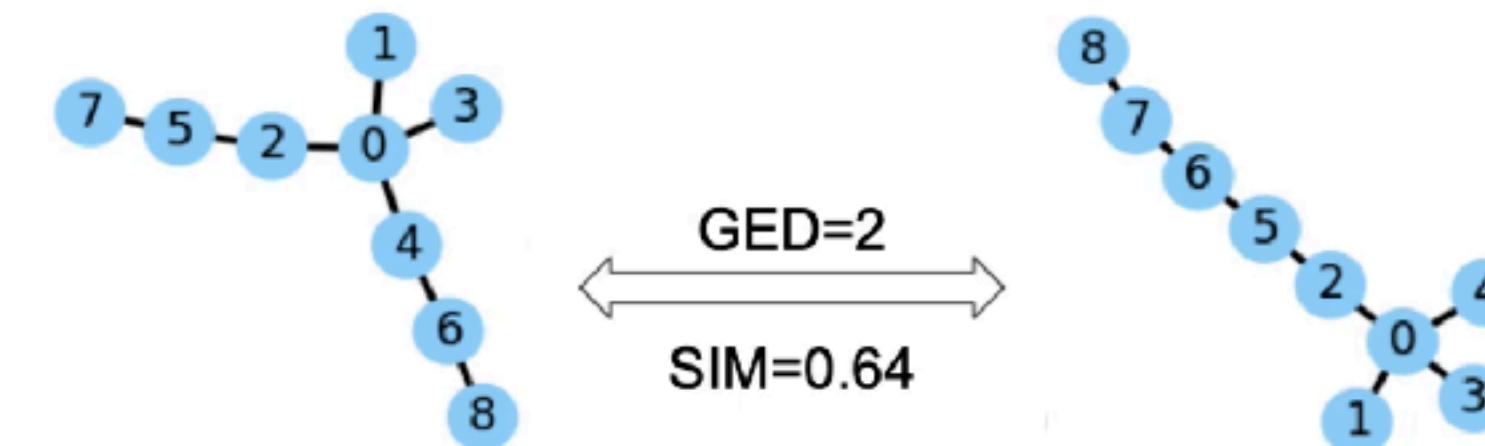
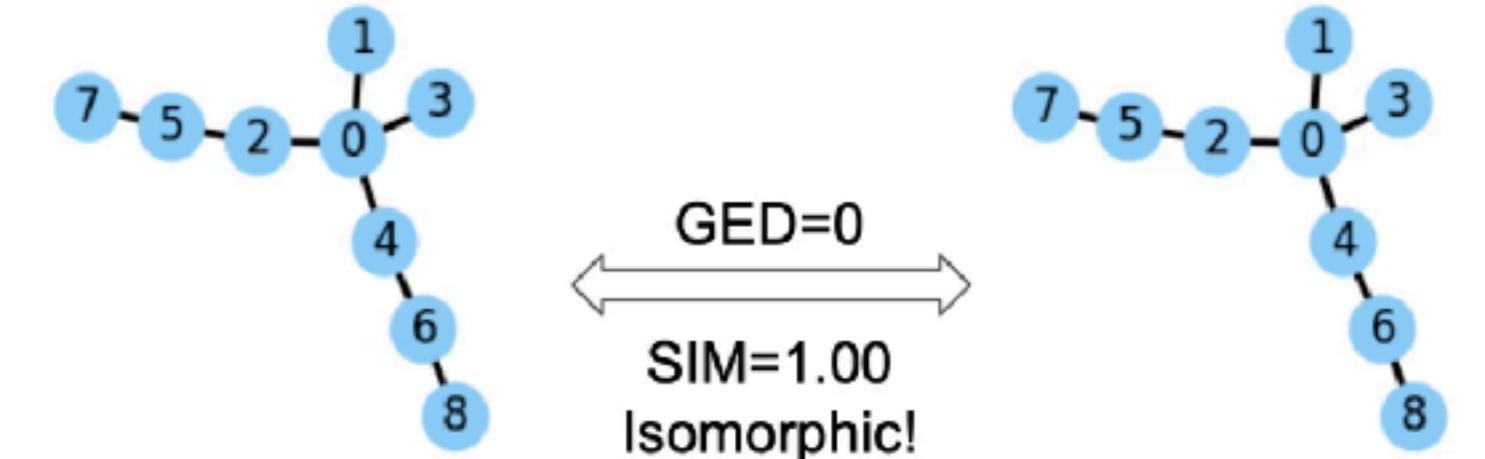
Graph Edit Distance (GED)



* edit operations: insertion, deletion of vertex/edge, relabeling

Examples of GED-Based Similarity Computation

The GED and similarity can be *transformed* to one another via a **bijective mapping**:
 $\text{sim} = \exp(-\text{GED})$



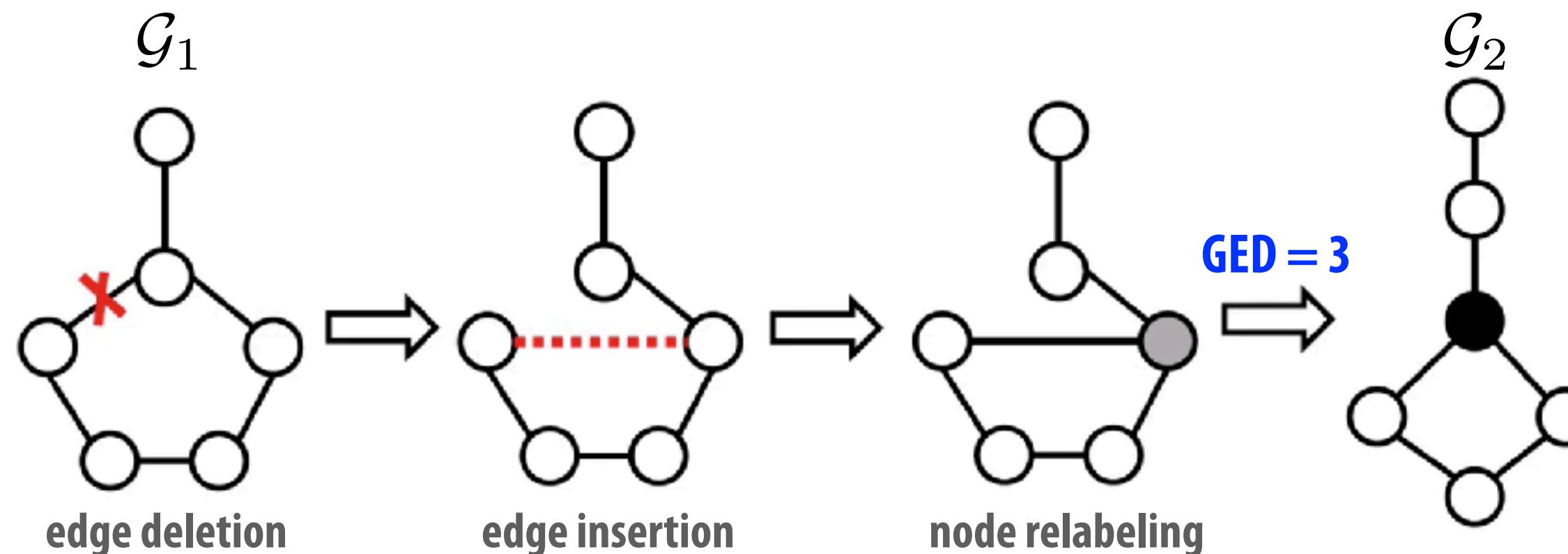
Existing Work on Graph Similarity Computation

- Computation of exact GED between two graphs: **NP-hard!**

- Search/Combinatorial optimization approaches:**

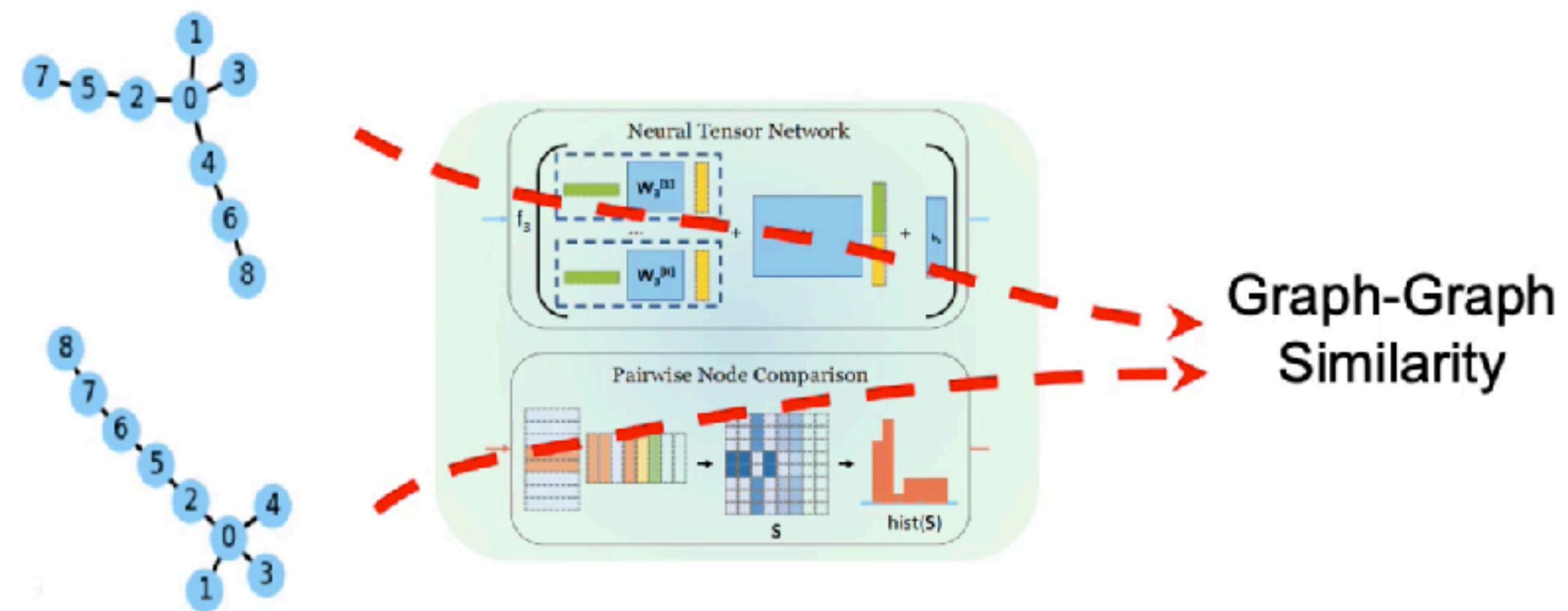
- Domain knowledge and heuristics
- Difficult to design

Method	Time Complexity
A* [19]	$O(N_1^{N_2})$
Beam [11]	subexponential
Hungarian [13]	$O((N_1 + N_2)^3)$
VJ [12]	$O((N_1 + N_2)^3)$
HED [14]	$O((N_1 + N_2)^2)$

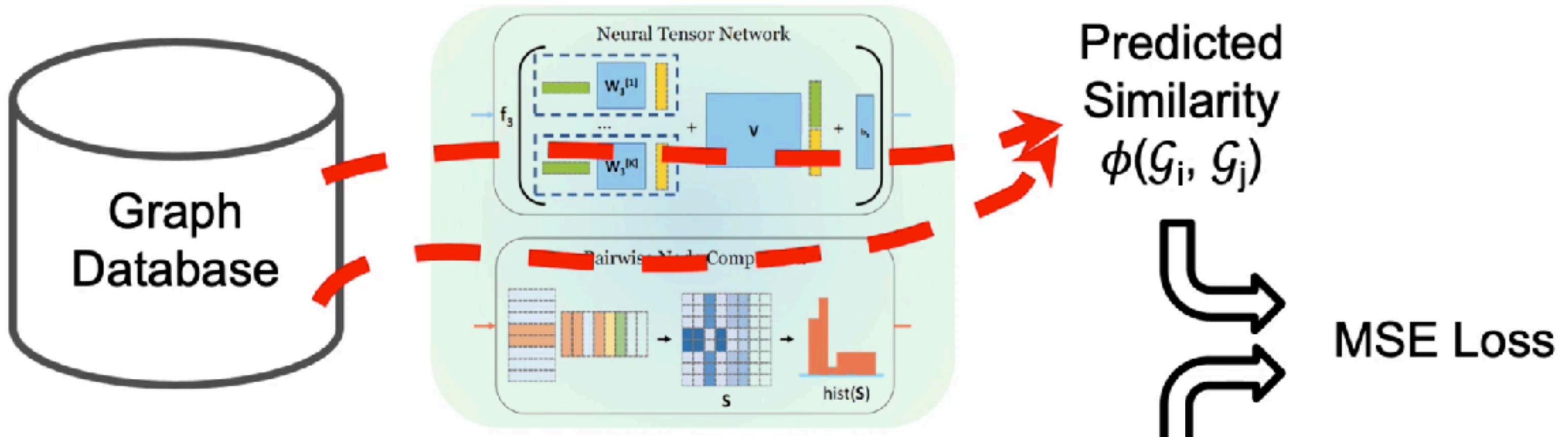


A General Learning Framework for Graph Similarity

- **Goal:** Learn a *neural network*-based similarity function $\phi(G_i, G_j)$
 - **Input:** A pair of graph, G_i and G_j
 - **Output:** The estimated graph-graph similarity
 - **Desired properties of ϕ :**
 - Representation-invariant
 - Inductive
 - Learnable



Training Phase: Minimize $(\phi(\mathcal{G}_i, \mathcal{G}_j) - s_{ij})^2$



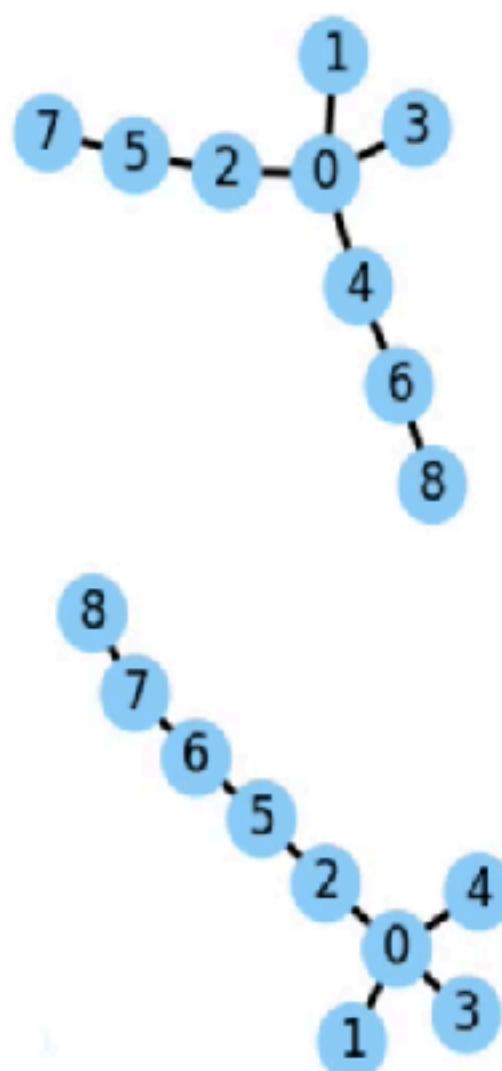
$$\mathcal{L} = \frac{1}{|\mathcal{D}|} \sum_{(i,j) \in \mathcal{D}} (\phi(\mathcal{G}_i, \mathcal{G}_j) - s_{ij})^2$$

\mathcal{D} : the set of training graph pairs

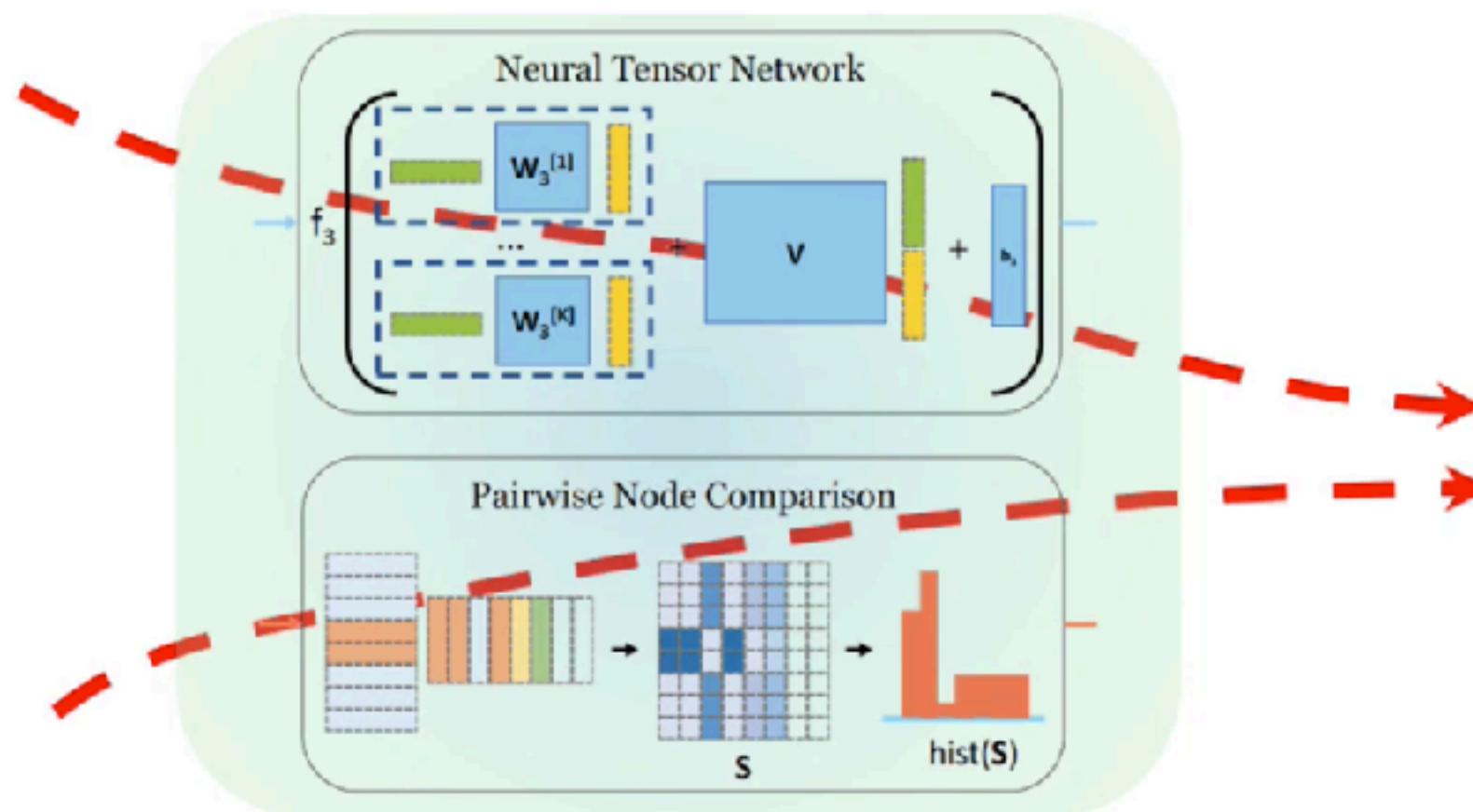
1.00	0.79	0.53	0.4	0.41	0.10	...	0.14
0.71	1.00	0.7	0.7	0.7	0.7	...	0.15
0.68	0.73	1.00	0.88	0.46	0.23	...	0.27
0.41	0.39	0.75	0.85	1.00	0.81	...	0.28
0.10	0.12	0.23	0.1	0.01	0.00	...	0.91
...
0.14	0.15	0.27	0.28	0.86	0.91	...	1.00

True
Similarity
 S_{ij}

Testing Phase: Graph Similarity Computation



$$g(\mathbf{h}_i, \mathbf{h}_j) = f_3(\mathbf{h}_i^T \mathbf{W}_3^{[1:K]} \mathbf{h}_j + V \begin{bmatrix} \mathbf{h}_i \\ \mathbf{h}_j \end{bmatrix} + \mathbf{b}_3)$$

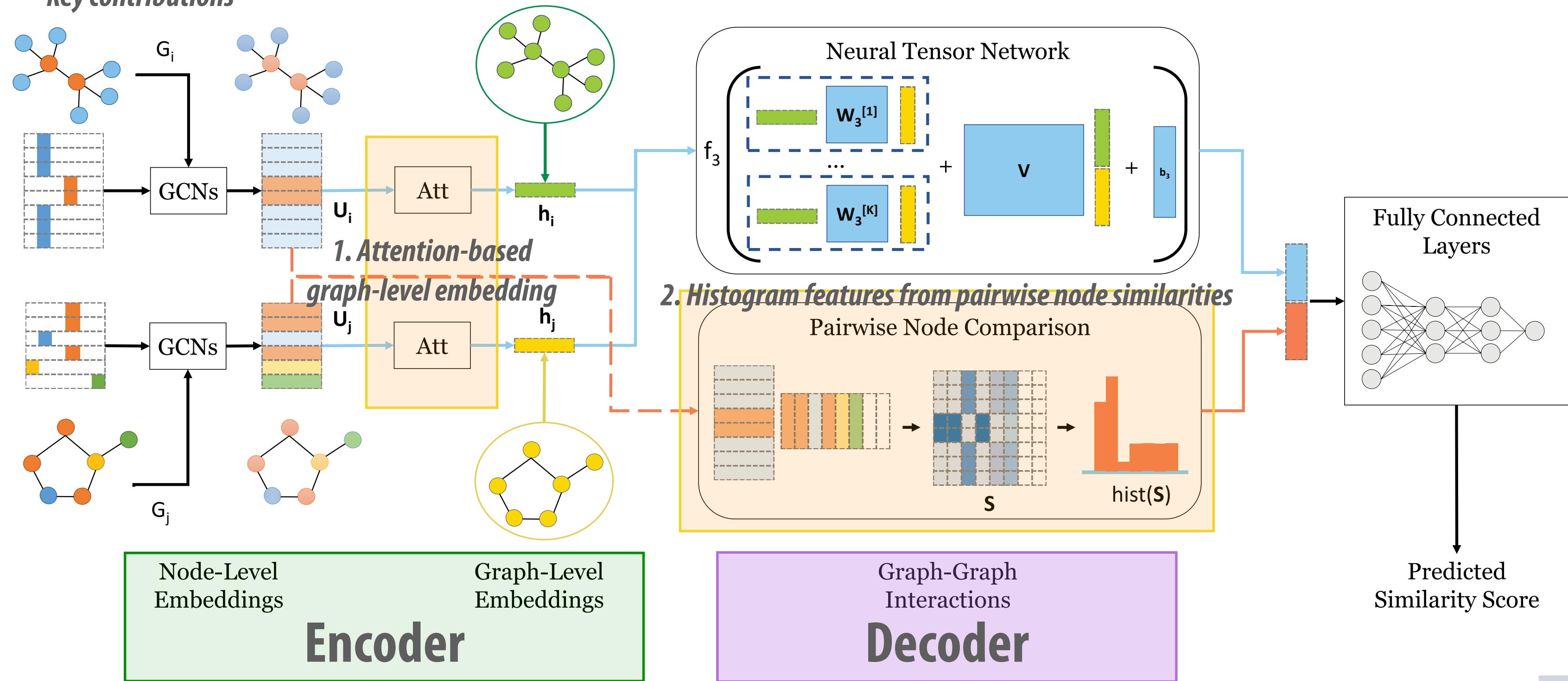


Graph-Graph
Similarity

$$\text{Node attentive mechanism } \mathbf{h} = \sum_{n=1}^N f_2(\mathbf{u}_n^T \mathbf{c}) \mathbf{u}_n = \sum_{n=1}^N f_2(\mathbf{u}_n^T \tanh((\frac{1}{N} \sum_{m=1}^N \mathbf{u}_m) \mathbf{W}_2)) \mathbf{u}_n$$

SimGNN: A Novel Framework

*Key contributions



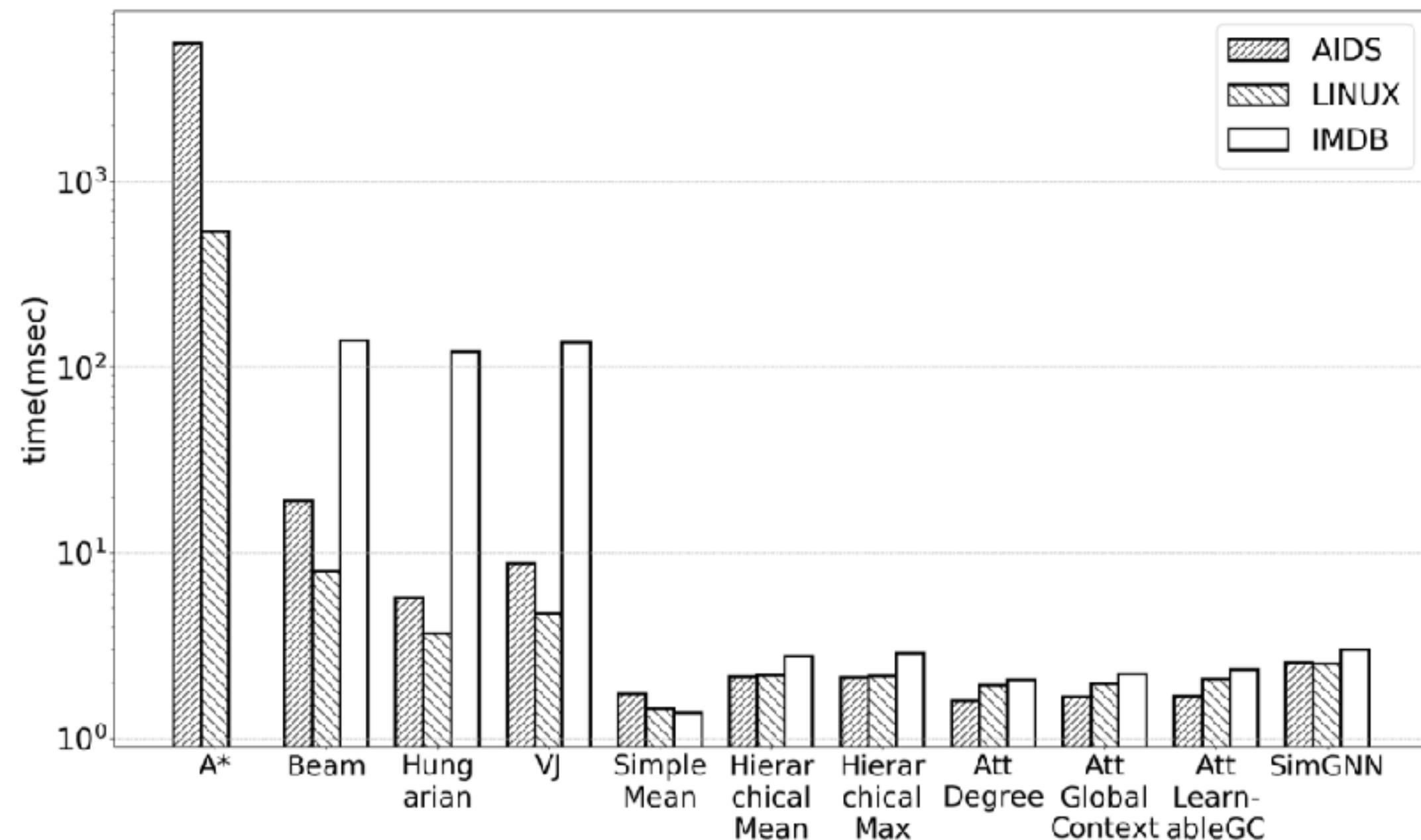
Evaluation: Effectiveness

Method	Spearman's Rank Correlation Coefficient (ρ)			Kendall's Rank Correlation Coefficient (τ)	
	mse(10^{-3})	ρ	τ	p@10	p@20
Beam	12.090	0.609	0.463	0.481	0.493
Hungarian	25.296	0.510	0.378	0.360	0.392
VJ	29.157	0.517	0.383	0.310	0.345
SimpleMean	3.115	0.633	0.480	0.269	0.279
HierarchicalMean	3.046	0.681	0.629	0.246	0.340
HierarchicalMax	3.396	0.655	0.505	0.222	0.295
AttDegree	3.338	0.628	0.478	0.209	0.279
AttGlobalContext	1.472	0.813	0.653	0.376	0.473
AttLearnableGC	1.340	0.825	0.667	0.400	0.488
SimGNN	1.189	0.843	0.690	0.421	0.514

More **accurate** than *most* the existing approximate GED algorithms.

Precision at k (p@k) is computed by taking the intersection of the predicted top k results and the ground-truth top k results divided by k .

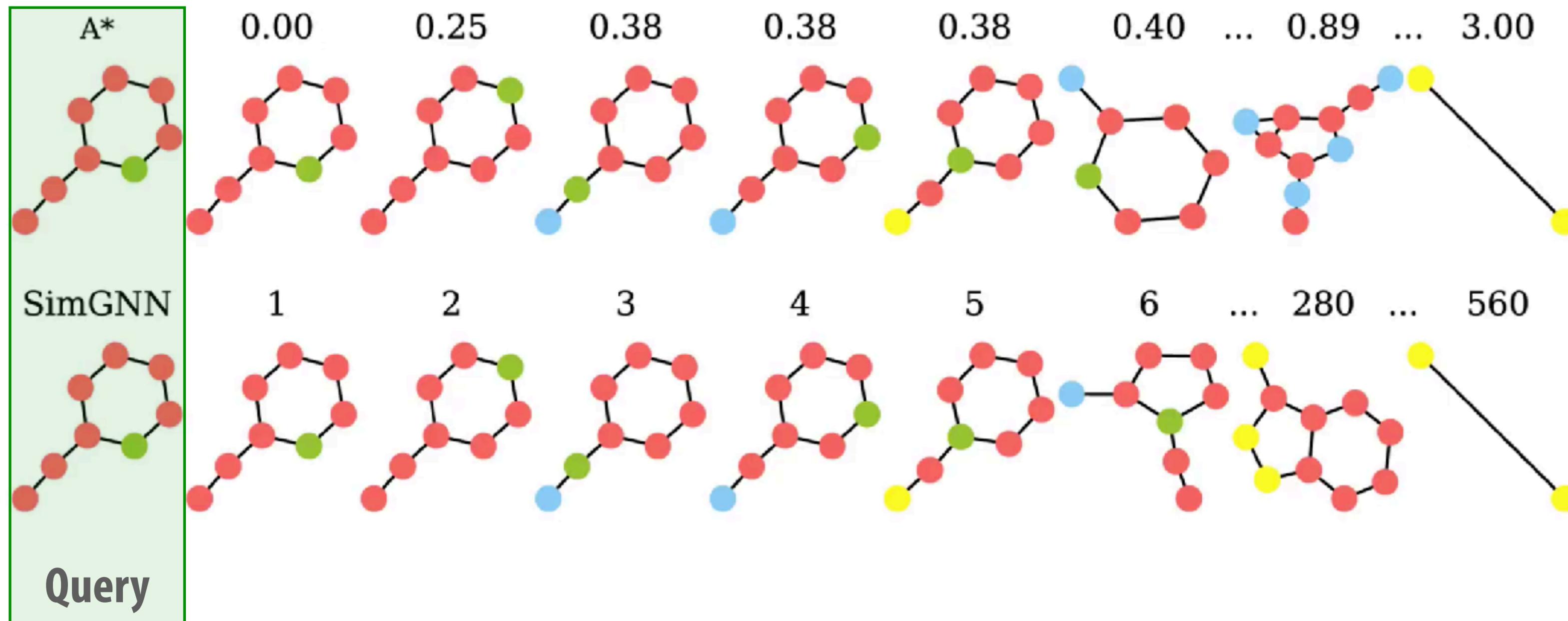
Evaluation: Efficiency



Other NN approaches are **fast** but **less accurate** than SimGNN.
Beam is better, but is **much slower** (log-scale).

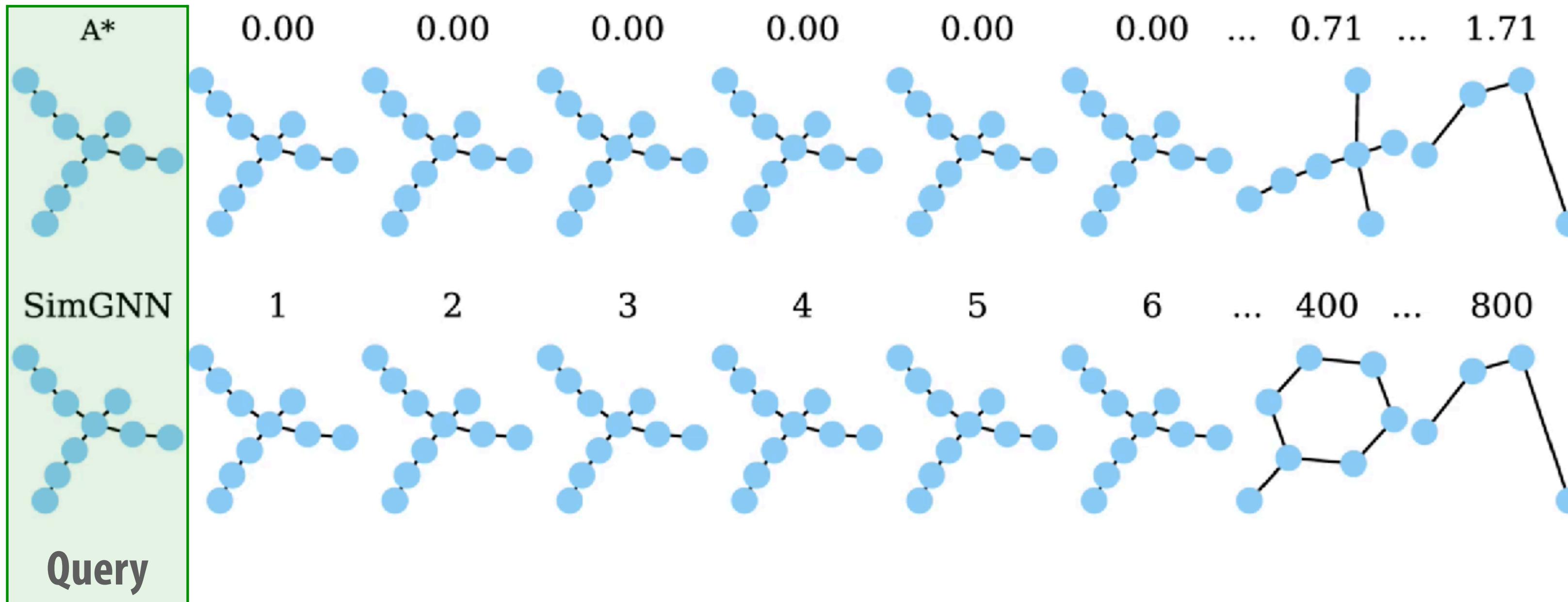
Case Study: AIDS

A*: the *exact* algorithm



Case Study: LINUX

A*: the *exact* algorithm



Key Takeaways

- SimGNN enjoys the key advantage of **efficiency** due to the nature of neural network computation.
- **Representation-invariant.** The same graph can be represented by different adjacency matrices by *permuting* the order of nodes. The computed similarity score *should be invariant* to such changes.
- **Inductive.** The similarity computation should *generalize to unseen graphs*, i.e. compute the similarity score for graphs outside the training graph pairs.
- **Learnable.** The model should be adaptive to any similarity metric, by *adjusting its parameters through training*.

Summary

- The intersection of *graph deep learning* and *graph search* problem
- Tackling the core operation of *graph similarity* computation via a novel neural network based approach
- Key idea is to learn a *neural network based function* that is *representation-invariant*, *inductive*, and *adaptive* to the specific similarity metric
- SimGNN runs very *fast* compared to existing classic algorithms on approximate Graph Edit Distance computation, and achieves very competitive *accuracy*.

Thank
you

