Graph Homomorphism Features: Why Not Sample?
Paul Beaujean 1  Florian Sikora 1  Florian Yger 1
1LAMSADE, Université Paris-Dauphine

**What is a graph morphism?**

Graph morphisms are functions that map the set of nodes from a pattern graph \( F \) to the set of nodes of a target graph \( G \). Graph morphisms that preserve adjacency are called homomorphisms. We say that \( G_1 \cong G_2 \) if and only if there exists an isomorphism between \( G_1 \) and \( G_2 \). This is a natural way to express that two graphs are identical up to a permutation.

**Computational Complexity**

- Finding a graph homomorphism is \( \text{NP} \)-hard because of its relation to finding cliques.
- Counting graph homomorphisms is \( \text{AP} \)-hard however.
- Diaz et al. (2002) give a slice-wise polynomial algorithm depending on the treewidth of \( F \).
- Allowing some multiplicative approximation does not make the problem easier.

**Homomorphism statistics: a canonical vector representation of graphs**

Combining homomorphism statistics from several pattern graphs reveals deep information which characterizes graphs and can be used as a permutation-invariant vector representation.

- Homomorphism numbers: absolute counts (between 0 and \( n^k \))
- Homomorphism densities: ratio of homomorphisms to morphisms (between 0 and 1)

**Lovász on homomorphism densities**

**Theorem**

Given two undirected graphs \( G_1 \) and \( G_2 \) with the same number of nodes \( n \) and \( \gamma_n \), the set of all simple graphs with at most \( n \) nodes, we have:

\[
G_1 \cong G_2 \iff \varphi(G_1) = \varphi(G_2).
\]

\( \varphi(F, G) = \) number of homomorphisms from \( F \) to \( G \) / number of morphisms from \( F \) to \( G \)

\( \varphi(G_1) = \varphi(F, G_1) \setminus \{ F, G \} \), where \( \gamma_n \) is the set of graphs on at most \( n \) nodes.

**Approximate homomorphism densities for large-scale graph learning**

To handle large datasets containing large graphs we propose to use approximate homomorphism densities as permutation-invariant feature descriptors parameterized by a given family of pattern graphs.

**Experimental results**

We compare two implementations of the sGHD algorithm with the standard implementation of the graph homomorphism numbers-based GHC algorithm of NT & Maehara (2020). Experiments are conducted on Erdős-Rényi random graphs with an edge density that guarantees the presence of several subgraph patterns. We obtain a quasi-constant runtime for sGHD with Bloom filter.

**Running time (ms)**

- sGHD: sample graph homomorphism density
- GHC: homlib impl. of Diaz et al., 2002
- sGHD (\( \varepsilon = 5 \times 10^{-3} \)) with adjacency list
- sGHD (\( \varepsilon = 5 \times 10^{-3} \)) with Bloom filter
- sGHD (\( \varepsilon = 1 \times 10^{-2} \)) with Bloom filter

**Additional insights**

- Low runtime increase from computing homomorphism statistics from larger cliques \( K_5, K_6 \)
- Actual precision obtained is 2 orders of magnitude above requested precision
- Graph sparsity short-circuits edge membership queries leading to good performance

**Future work**

1. Additional experiments on currently available large datasets such as OGB (small graphs), MaNet (large graphs), and new social network datasets currently being extracted
2. Representation learning via selection of the family of pattern graphs
3. Investigation into polynomial models and their relationship with graph algebras

**References**