Franka Bause, Nils M. Kriege

University of Vienna

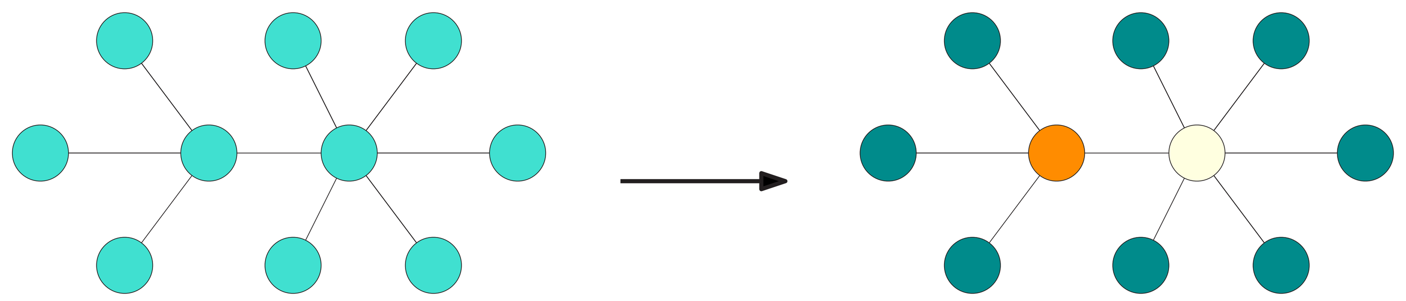
Gradual Weisfeiler-Leman: Slow and Steady Wins the Race

universität wien

Problem

Weisfeiler-Leman colors diverge too quickly!

- Node with degree 4 **as similar** to degree 1 **as** to degree 6
- WL-based graph kernels: only **few** iterations used



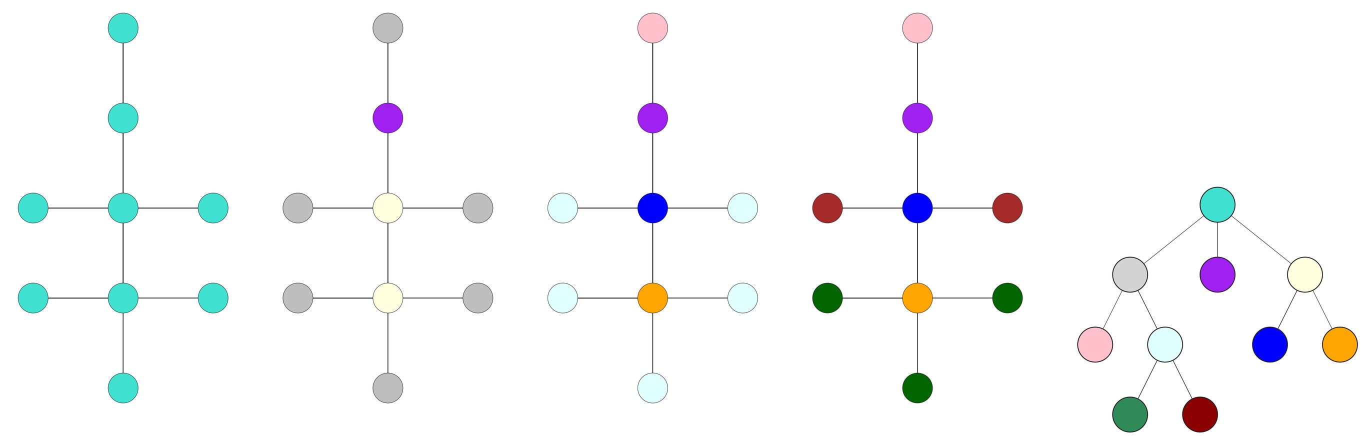
- **Idea:** restrict number of new colors per iteration
- Reach stable coloring slower

Weisfeiler-Leman/Color Refinement

- Initial coloring: uniform/depending on node label
- Update color of nodes:

$$c_{i+1}(v) = z(c_i(v), \{\!\{c_i(u) | u \in N(v)\}\!\}),$$

z : injective



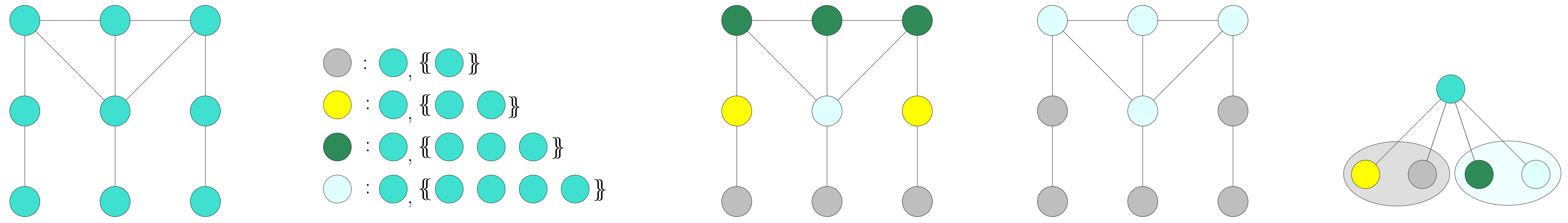
- Colors can be represented in a hierarchy

Our Contribution

- **Generalizing** color refinement: refining, neighborhood preserving (**renep**) functions
- Connections to **original** Weisfeiler-Leman algorithm and other vertex refinement strategies
- Two new **graph kernels** based on renep functions
- Application to approximating the **graph edit distance**

Gradual Weisfeiler-Leman Refinement

Idea: **Slow** down **convergence** of color refinement using a non-injective data-dependent update function **preserving** Weisfeiler-Leman **expressivity**



Update function:

- New coloring is a **refinement** of previous coloring
  - ⇒ Nodes with different old colors get different new colors
- Nodes with equal old colors and equal neighbor label multiset get equal new colors
- New coloring is **equal** to old coloring ⇔ **stable** (WL) coloring reached

Color Update using Clustering

- Interpret neighbor color multisets as vectors
- Cluster vectors for each old color separately (clusters **imply** new colors)
- k-means clustering: **number** of new colors can be **controlled** easily

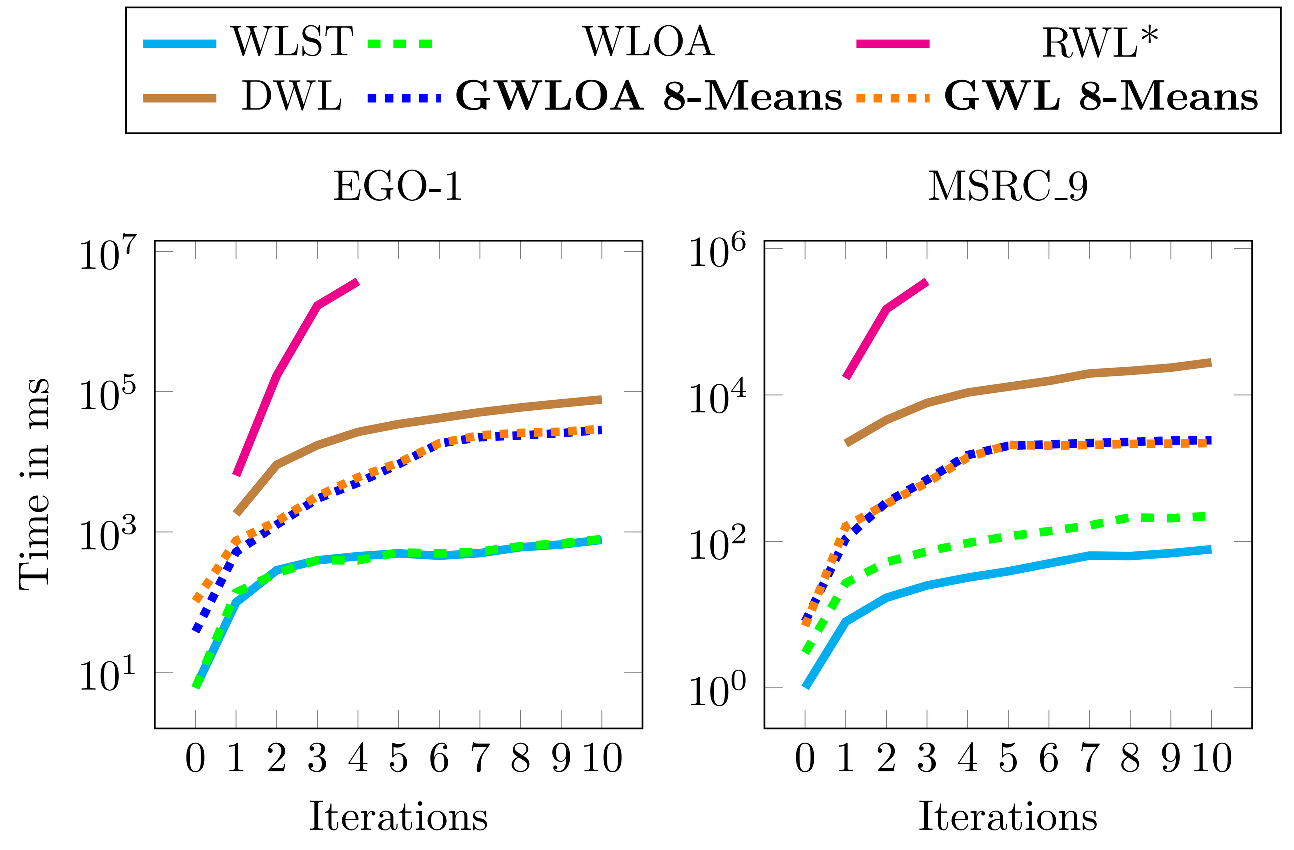
Classification Accuracy

Kernel	PTC_FM	KKI	EGO-1	EGO-2
WLST [1]	64.16 ±1.30	49.97 ±2.88	51.30 ±2.42	57.15 ±1.61
DWL [2]	64.18 ±1.46	50.93 ±2.87	55.80 ±1.35	56.50 ±1.64
RWL* [3]	62.43 ±1.46	46.54 ±4.03	65.60 ±2.74	70.20 ±1.36
WLOA [4]	62.34 ±1.39	48.72 ±4.05	55.95 ±1.11	60.30 ±2.00
GWL	62.61 ±1.94	**57.79** ±3.95	67.95 ±2.05	**73.65** ±1.86
GWLOA	**64.58** ±1.77	47.47 ±2.41	**69.80** ±1.65	72.40 ±2.52

Kernel	COLLAB	DD	IMDB-B	MSRC_9
WLST	78.98 ±0.22	79.00 ±0.52	72.01 ±0.80	90.13 ±0.75
DWL	78.93 ±0.18	78.92 ±0.40	72.36 ±0.56	90.50 ±0.76
RWL*	77.94 ±0.38	77.52 ±0.65	72.96 ±0.86	88.86 ±0.89
WLOA	80.81 ±0.22	**79.44** ±0.31	72.60 ±0.89	90.68 ±0.92
GWL	80.62 ±0.33	79.00 ±0.81	**73.66** ±1.25	88.32 ±1.20
GWLOA	**81.30** ±0.29	78.49 ±0.57	72.88 ±0.79	**91.27** ±1.06

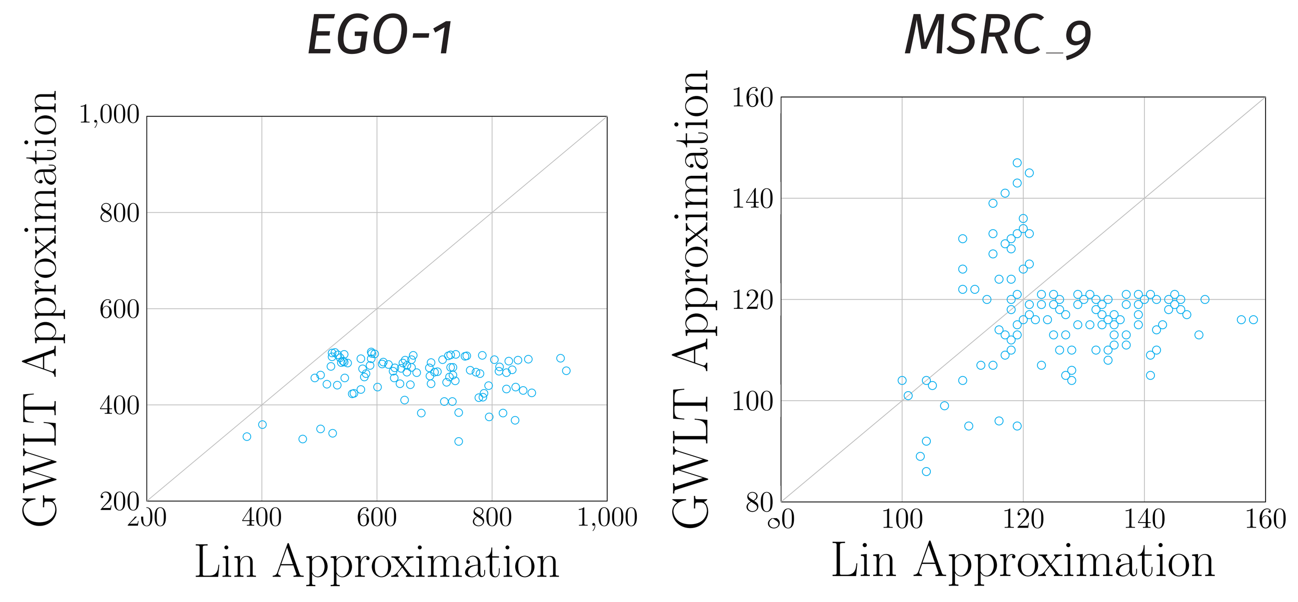
Running Time

- **GWL**: WLST kernel with GWL coloring
- **GWLOA**: WLOA kernel with GWL coloring



Approximating the Graph Edit Distance

- **GED**: cost of transforming one graph into another
- Use tree metric from GWL for **node similarity**
- Find optimal assignment between the nodes
- Cost of (sub-optimal) edit path derived from assignment ≙ **upper bound** for GED
- **Lin** [5]: underlying node similarity based on WL
- **GWLT**: underlying node similarity based on GWL



Conclusion and Future Work

Better measure for vertex similarity!

Future Work

- Explore **other** possible update functions
- Incorporate **continuous** attributes

preprint on arXiv ⟶

https://github.com/frareba

@frar3ba

References

[1] Shervashidze et al., Weisfeiler-Lehman graph kernels. J. Mach. Learn. Res, 2011.
[2] Yanardag and Vishwanathan, Deep graph kernels. SIGKDD, 2015.
[3] Schulz et al., A generalized Weisfeiler-Lehman graph kernel. Mach Learn, 2022.
[4] Kriege et al., On valid optimal assignment kernels and applications to graph classification. NIPS, 2016.
[5] Kriege et al., Computing optimal assignments in linear time for approximate graph matching. ICDM, 2019.

1st Learning on Graphs Conference (LoG 2022) — December 9 – December 12, 2022