Finite Model Theory
Lecture 13: Color Refinement and k-Weisfeiler-Leman

Spring 2025

Announcement

- HW3 was due on Friday.
- HW4 (the last one!!) to be posted this week.
- Topics this week: Color Refinement, k-Weisfeiler-Leman, FO^k, C^k, applications to GNN and graph kernels.
- Next week: descriptive complexity.

Overview

Overview

- Color refinement and the Weisfeiler-Leman algorithm are used as test for graph isomorphism
- They have strong connections to logic, more precisely to C^k, where C^k is FO^k extended with "counting quantifiers"
- More recently: strong connection to Graph Neural Networks (GNN) and graph kernels
- Today: color refinement and WL
- Next lecture: connection to C^k and brief discussion of GNN and graph kernels

This and the following lecture are based on [Morris et al., 2023, Grohe, 2021, Morris et al., 2019, Grohe, 2020]

Graphs and their Isomorphisms

We consider undirected graphs $G = (V, E)$ without self-loops:

$$(uv) \in E \text{ implies } (vu) \in E \text{ and } u \neq v$$

$N(v)$ = the set of neighbors of v

Homomorphism $h : G \to G'$: a function $f : V(G) \to V(G')$ s.t. $(uv) \in E(G)$ implies $(h(u)h(v)) \in E(G')$.

Isomorphism: a bijection h such that both h, h^{-1} are homomorphisms.

G, G' are isomorphic, $G \approx G'$, if there exists an isomorphism $G \to G'$.

The problem given G, G', check if $G \approx G'$ is in NP.
It is believe to be neither in PTIME, nor NP-complete.

Graphs and their Isomorphisms

We consider undirected graphs $G = (V, E)$ without self-loops:

$$(uv) \in E \text{ implies } (vu) \in E \text{ and } u \neq v$$

$N(v)$ = the set of neighbors of v

Homomorphism $h : G \to G'$: a function $f : V(G) \to V(G')$ s.t. $(uv) \in E(G)$ implies $(h(u)h(v)) \in E(G')$.

Isomorphism: a bijection h such that both h, h^{-1} are homomorphisms.

G, G' are isomorphic, $G \approx G'$, if there exists an isomorphism $G \to G'$.

The problem given G, G', check if $G \approx G'$ is in NP.
It is believe to be neither in PTIME, nor NP-complete.

Graphs and their Isomorphisms

We consider undirected graphs $G = (V, E)$ without self-loops:

$$(uv) \in E \text{ implies } (vu) \in E \text{ and } u \neq v$$

$N(v)$ = the set of neighbors of v

Homomorphism $h : G \to G'$: a function $f : V(G) \to V(G')$ s.t. $(uv) \in E(G)$ implies $(h(u)h(v)) \in E(G')$.

Isomorphism: a bijection h such that both h, h^{-1} are homomorphisms.

G, G' are isomorphic, $G \approx G'$, if there exists an isomorphism $G \to G'$.

The problem given G, G', check if $G \approx G'$ is in NP.
It is believe to be neither in PTIME, nor NP-complete.

Graphs and their Isomorphisms

We consider undirected graphs $G = (V, E)$ without self-loops:

$$(uv) \in E \text{ implies } (vu) \in E \text{ and } u \neq v$$

$N(v)$ = the set of neighbors of v

Homomorphism $h : G \to G'$: a function $f : V(G) \to V(G')$ s.t. $(uv) \in E(G)$ implies $(h(u)h(v)) \in E(G')$.

Isomorphism: a bijection h such that both h, h^{-1} are homomorphisms.

G, G' are isomorphic, $G \approx G'$, if there exists an isomorphism $G \to G'$.

The problem given G, G', check if $G \approx G'$ is in NP.
It is believe to be neither in PTIME, nor NP-complete.

Graphs and their Isomorphisms

We consider undirected graphs $G = (V, E)$ without self-loops:

$$(uv) \in E \text{ implies } (vu) \in E \text{ and } u \neq v$$

$N(v)$ = the set of neighbors of v

Homomorphism $h : G \to G'$: a function $f : V(G) \to V(G')$ s.t. $(uv) \in E(G)$ implies $(h(u)h(v)) \in E(G')$.

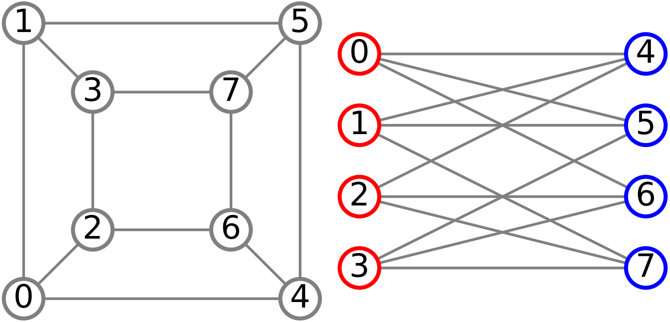
Isomorphism: a bijection h such that both h, h^{-1} are homomorphisms.

G, G' are isomorphic, $G \approx G'$, if there exists an isomorphism $G \to G'$.

The problem given G, G', check if $G \approx G'$ is in NP.
It is believe to be neither in PTIME, nor NP-complete.

Example

From
`https://en.wikipedia.org/wiki/Graph_isomorphism_problem`



Color Refinement or 1-WL

Color Refinement

Iterative process that assigns a "color" to each node.

[1] $\{\!\{\ldots\}\!\}$ is a bag, as in $\{\!\{a, a, b, c, c, c\}\!\}$.

Color Refinement

Iterative process that assigns a "color" to each node.

- Initially all nodes have the same color:

$$\boxed{\mathtt{cr}^0(v) \stackrel{\text{def}}{=} ()}$$

[1] $\{\!\{\dots\}\!\}$ is a bag, as in $\{\!\{a, a, b, c, c, c\}\!\}$.

Color Refinement

Iterative process that assigns a "color" to each node.

- Initially all nodes have the same color:

$$\boxed{\mathtt{cr}^0(v) \stackrel{\text{def}}{=} ()}$$

- For $t \geq 0$, assign colors based on the neighbors's colors:[1]

$$\boxed{\mathtt{cr}^{t+1}(v) \stackrel{\text{def}}{=} (\mathtt{cr}^t(v), \{\!\{\mathtt{cr}^t(w) \mid w \in N(v)\}\!\})}$$

[1] $\{\!\{\ldots\}\!\}$ is a bag, as in $\{\!\{a, a, b, c, c, c\}\!\}$.

Color Refinement

Iterative process that assigns a "color" to each node.

- Initially all nodes have the same color:

$$\boxed{\mathtt{cr}^0(v) \stackrel{\text{def}}{=} ()}$$

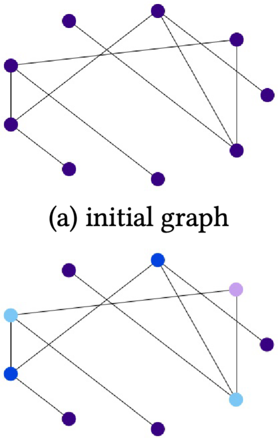
- For $t \geq 0$, assign colors based on the neighbors's colors:[1]

$$\boxed{\mathtt{cr}^{t+1}(v) \stackrel{\text{def}}{=} (\mathtt{cr}^t(v), \{\{\mathtt{cr}^t(w) \mid w \in N(v)\}\})}$$

- The stable coloring is $\mathtt{cr}^\infty \stackrel{\text{def}}{=} \mathtt{cr}^t$ where t is s.t. $\mathtt{cr}^{t+1} = \mathtt{cr}^t$.

[1] $\{\{\ldots\}\}$ is a bag, as in $\{\{a, a, b, c, c, c\}\}$.

Example from [Grohe, 2020]



(a) initial graph

(b) colouring after round 1

(c) colouring after round 2

(d) stable colouring after round 3

Figure 3: A run of 1-WL

Discussion

- The actual "color" doesn't matter: all that matter is whether two nodes have the same color or different colors:

$$\boxed{v \equiv^t w} \quad \text{if} \quad \boxed{\mathtt{cr}^t(v) = \mathtt{cr}^t(w)}$$

Discussion

- The actual "color" doesn't matter: all that matter is whether two nodes have the same color or different colors:

$$\boxed{v \equiv^t w} \quad \text{if} \quad \boxed{\mathtt{cr}^t(v) = \mathtt{cr}^t(w)}$$

- Observe that $\equiv^0 = V(G) \times V(G)$ and $\equiv^t \supseteq \equiv^{t+1}$.

Discussion

- The actual "color" doesn't matter: all that matter is whether two nodes have the same color or different colors:

$$\boxed{v \equiv^t w} \quad \text{if} \quad \boxed{\mathtt{cr}^t(v) = \mathtt{cr}^t(w)}$$

- Observe that $\equiv^0 = V(G) \times V(G)$ and $\equiv^t \supseteq \equiv^{t+1}$.

- $\equiv^\infty$ is the greatest fixpoint of the refinement procedure:

$$\equiv^0 \supset \equiv^1 \supset \equiv^2 \supset \cdots \supset \equiv^t = \equiv^{t+1}$$

Discussion

- The actual "color" doesn't matter: all that matter is whether two nodes have the same color or different colors:

$$\boxed{v \equiv^t w} \quad \text{if} \quad \boxed{\mathtt{cr}^t(v) = \mathtt{cr}^t(w)}$$

- Observe that $\equiv^0 = V(G) \times V(G)$ and $\equiv^t \supseteq \equiv^{t+1}$.

- $\equiv^\infty$ is the greatest fixpoint of the refinement procedure:

$$\equiv^0 \supset \equiv^1 \supset \equiv^2 \supset \cdots \supset \equiv^t = \equiv^{t+1}$$

- Better: stable refinement can be computed in time $O((m+n)\log n)$, where $m = |E(G)|$, $n = |V(G)|$.

Viewing Colors as Tree Unfoldings

One can view the set of colors at round t as the set of all trees of depths t. Example from [Grohe, 2020]:

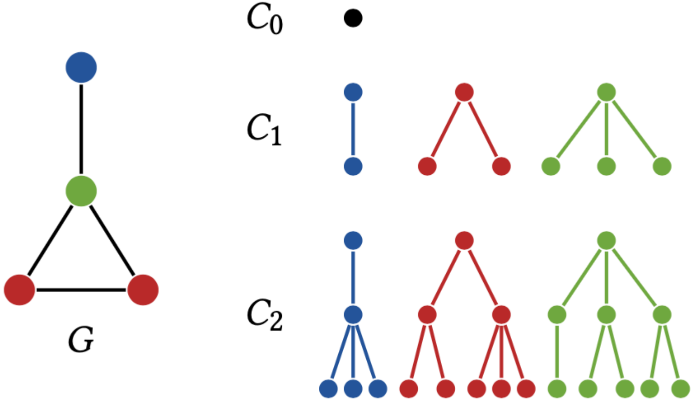


Figure 5: Viewing colours of WL as trees

Color Refinement for Isomorphism Test

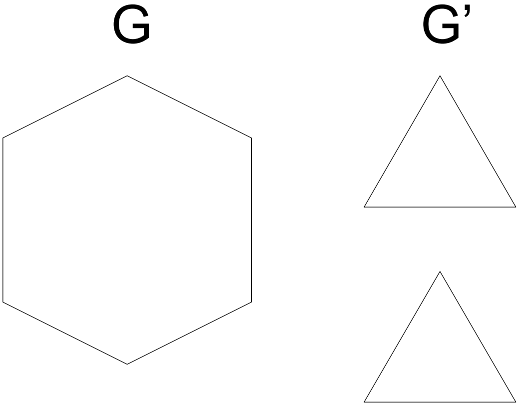
Input: G, G'

Take their disjoint union $G \cup G'$ and compute its stable color $\mathtt{cr}^\infty$.

If for any color, the two graphs a have different number of nodes of that color, then they are not isomorphic.

Other, we need to run an expensive isomorphism test.

Graph Coloring is Insufficient



All nodes have the same color, yet they are not isomorphic.

Canonical Labeling of a Graph

A canonical labeling is a graph G is a labeling of its nodes with numbers $1, 2, \ldots, n$ s.t. $G \approx G'$ iff their canonical labelings are identical.

Canonical Labeling of a Graph

A canonical labeling is a graph G is a labeling of its nodes with numbers $1, 2, \dots, n$ s.t. $G \approx G'$ iff their canonical labelings are identical.

A canonical labeling gives us immediately an isomorphism test: simply check that the one-one mapping $1 \mapsto 1$, $2 \mapsto 2$, etc is an isomorphism.

Canonical Labeling of a Graph

A canonical labeling is a graph G is a labeling of its nodes with numbers $1, 2, \ldots, n$ s.t. $G \approx G'$ iff their canonical labelings are identical.

A canonical labeling gives us immediately an isomorphism test: simply check that the one-one mapping $1 \mapsto 1$, $2 \mapsto 2$, etc is an isomorphism.

If the stable coloring assigns distinct colors to each node, then it is a canonical labeling.

Canonical Labeling of a Graph

A canonical labeling is a graph G is a labeling of its nodes with numbers $1, 2, \ldots, n$ s.t. $G \approx G'$ iff their canonical labelings are identical.

A canonical labeling gives us immediately an isomorphism test: simply check that the one-one mapping $1 \mapsto 1$, $2 \mapsto 2$, etc is an isomorphism.

If the stable coloring assigns distinct colors to each node, then it is a canonical labeling.

Theorem

[Babai et al., 1980] In a random graph (where $pr(uv) = 1/2$ for all u, v), all colors $cr^2(v)$ are distinct, with probability $\geq 1 - \sqrt[7]{1/n}$.

Discussion

- For almost all graphs, the stable color uniquely identifies the graph up to isomorphism.
- Even more: only two iterations of stable coloring suffice!
- However, stable coloring fails to differentiate any two regular[2] graphs with the same degree: e.g. C_6 v.s. $C_3 \cup C_3$.
- The Weisfeiler-Leman procedure refines stable coloring to identify more graphs.

[2] Meaning: any two nodes have the same degree.

Weisfeiler-Leman

Initially Colored Graph

For the Weisfeiler-Leman algorithm it will be convenient to considered labeled graphs:

$$G = (V, E, P_1, \ldots, P_m)$$

where $P_1, \ldots, P_m$ form a partition on V.

Initially Colored Graph

For the Weisfeiler-Leman algorithm it will be convenient to considered labeled graphs:

$$G = (V, E, P_1, \dots, P_m)$$

where $P_1, \dots, P_m$ form a partition on V.

The atomic type, $\mathtt{atp}_k(\boldsymbol{v})$, of a k-tuples $\boldsymbol{v} \in V(G)^k$ is the isomorphism type of the subgraph induced by $\boldsymbol{v}$.

$\mathtt{atp}_k(\boldsymbol{v}) = \mathtt{atp}_k(\boldsymbol{u})$ iff the mapping $(v_1, \dots, v_k) \mapsto (u_1, \dots, u_k)$ is a partial isomomorphism.

k-Dimensional Weisfeiler-Leman

k-WL colors k-tuples of nodes of the graph.

k-Dimensional Weisfeiler-Leman

k-WL colors k-tuples of nodes of the graph.

Initially, each vector $\boldsymbol{v} \in V(G)^k$ is colored with $\mathtt{atp}_k(\boldsymbol{v})$.

k-Dimensional Weisfeiler-Leman

k-WL colors k-tuples of nodes of the graph.

Initially, each vector $\boldsymbol{v} \in V(G)^k$ is colored with $\mathtt{atp}_k(\boldsymbol{v})$.

At step $t+1$, the tuple $\boldsymbol{v}$ is colored with:

$$\mathtt{wl}_k^{t+1}(\boldsymbol{v}) \stackrel{\text{def}}{=} (\mathtt{wl}_k^t(\boldsymbol{v}), M(\boldsymbol{v}))$$

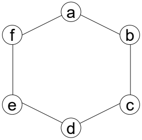
where M is:

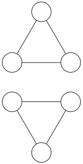
$$M(\boldsymbol{v}) \stackrel{\text{def}}{=} \{\{(\mathtt{atp}_{k+1}(\boldsymbol{v}w), \mathtt{wl}_k^t(\boldsymbol{v}[w/v_1]), \ldots, \mathtt{wl}_k^t(\boldsymbol{v}[w/v_k])) \mid w \in V(G)\}\}$$

note that $\mathtt{atp}_{k+1}(\boldsymbol{v}w)$ considers all colors of $\mathtt{wl}_k^t$.

Example

Consider $k = 2$.



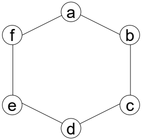


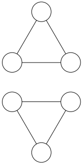
Example

Consider $k = 2$.

$t = 0$ there are three colors:

Same node: $\mathtt{wl}_2^0(aa) = \cdots = \mathtt{wl}_2^0(ff)$





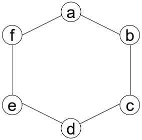
Example

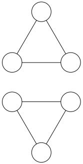
Consider $k = 2$.

$t = 0$ there are three colors:

Same node: $\mathtt{wl}_2^0(aa) = \cdots = \mathtt{wl}_2^0(ff)$

Edges: $\mathtt{wl}_2^0(ab) = \cdots = \mathtt{wl}_2^0(fa)$





Example

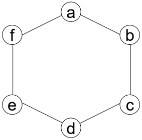
Consider $k = 2$.

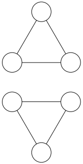
$t = 0$ there are three colors:

Same node: $\texttt{wl}_2^0(aa) = \cdots = \texttt{wl}_2^0(ff)$

Edges: $\texttt{wl}_2^0(ab) = \cdots = \texttt{wl}_2^0(fa)$

Non-edges: $\texttt{wl}_2^0(ac) = \texttt{wl}_2^0(ad) = \cdots = \texttt{wl}_2^0(fb)$





Example

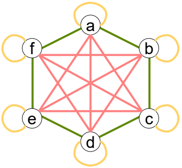
Consider $k = 2$.

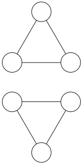
$t = 0$ there are three colors:

Same node: $\mathtt{wl}_2^0(aa) = \cdots = \mathtt{wl}_2^0(ff)$

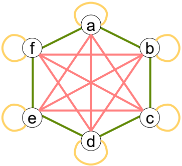
Edges: $\mathtt{wl}_2^0(ab) = \cdots = \mathtt{wl}_2^0(fa)$

Non-edges: $\mathtt{wl}_2^0(ac) = \mathtt{wl}_2^0(ad) = \cdots = \mathtt{wl}_2^0(fb)$





Example



Consider $k = 2$.

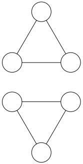
$t = 0$ there are three colors:

Same node: $\mathtt{wl}_2^0(aa) = \cdots = \mathtt{wl}_2^0(ff)$

Edges: $\mathtt{wl}_2^0(ab) = \cdots = \mathtt{wl}_2^0(fa)$

Non-edges: $\mathtt{wl}_2^0(ac) = \mathtt{wl}_2^0(ad) = \cdots = \mathtt{wl}_2^0(fb)$

$t = 1$ distinguishes between ac and ad:



Example

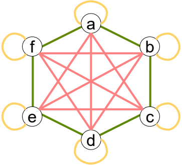
Consider $k = 2$.

$t = 0$ there are three colors:

Same node: $\mathtt{wl}_2^0(aa) = \cdots = \mathtt{wl}_2^0(ff)$

Edges: $\mathtt{wl}_2^0(ab) = \cdots = \mathtt{wl}_2^0(fa)$

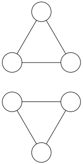
Non-edges: $\mathtt{wl}_2^0(ac) = \mathtt{wl}_2^0(ad) = \cdots = \mathtt{wl}_2^0(fb)$



$t = 1$ distinguishes between ac and ad:

$\mathtt{wl}_2^1(ad) = (\mathtt{wl}_2^0(ad), M(ad))$

$M(ad) = \{\{(\mathtt{atp}_3(ad{\color{red}v}), \mathtt{wl}_2^0({\color{red}v}d), \mathtt{wl}_2^0(a{\color{red}v})) \mid {\color{red}v} \in V(G)\}\}$



Example

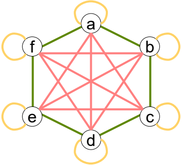
Consider $k = 2$.

$t = 0$ there are three colors:

Same node: $\mathtt{wl}_2^0(aa) = \cdots = \mathtt{wl}_2^0(ff)$

Edges: $\mathtt{wl}_2^0(ab) = \cdots = \mathtt{wl}_2^0(fa)$

Non-edges: $\mathtt{wl}_2^0(ac) = \mathtt{wl}_2^0(ad) = \cdots = \mathtt{wl}_2^0(fb)$



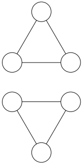
$t = 1$ distinguishes between ac and ad:

$\mathtt{wl}_2^1(ad) = (\mathtt{wl}_2^0(ad), M(ad))$

$M(ad) = \{\!\{(\mathtt{atp}_3(ad{\color{red}v}), \mathtt{wl}_2^0({\color{red}v}d), \mathtt{wl}_2^0(a{\color{red}v})) \mid {\color{red}v} \in V(G)\}\!\}$

$\mathtt{wl}_2^1(ac) = (\mathtt{wl}_2^0(ac), M(ac))$

$M(ac) = \{\!\{(\mathtt{atp}_3(ac{\color{red}w}), \mathtt{wl}_2^0({\color{red}w}c), \mathtt{wl}_2^0(a{\color{red}w})) \mid {\color{red}w} \in V(G)\}\!\}$



Example

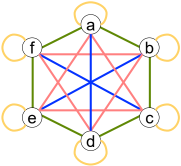
Consider $k = 2$.

$t = 0$ there are three colors:

Same node: $\mathtt{wl}_2^0(aa) = \cdots = \mathtt{wl}_2^0(ff)$

Edges: $\mathtt{wl}_2^0(ab) = \cdots = \mathtt{wl}_2^0(fa)$

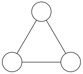
Non-edges: $\mathtt{wl}_2^0(ac) = \mathtt{wl}_2^0(ad) = \cdots = \mathtt{wl}_2^0(fb)$



$t = 1$ distinguishes between ac and ad:

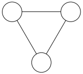
$\mathtt{wl}_2^1(ad) = (\mathtt{wl}_2^0(ad), M(ad))$

$M(ad) = \{\!\{(\mathtt{atp}_3(adv), \mathtt{wl}_2^0(vd), \mathtt{wl}_2^0(av)) \mid v \in V(G)\}\!\}$



$\mathtt{wl}_2^1(ac) = (\mathtt{wl}_2^0(ac), M(ac))$

$M(ac) = \{\!\{(\mathtt{atp}_3(acw), \mathtt{wl}_2^0(wc), \mathtt{wl}_2^0(aw)) \mid w \in V(G)\}\!\}$



Example

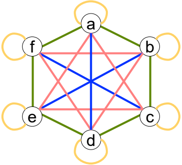
Consider $k = 2$.

$t = 0$ there are three colors:

Same node: $\mathtt{wl}_2^0(aa) = \cdots = \mathtt{wl}_2^0(ff)$

Edges: $\mathtt{wl}_2^0(ab) = \cdots = \mathtt{wl}_2^0(fa)$

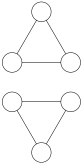
Non-edges: $\mathtt{wl}_2^0(ac) = \mathtt{wl}_2^0(ad) = \cdots = \mathtt{wl}_2^0(fb)$



$t = 1$ distinguishes between ac and ad:

$\mathtt{wl}_2^1(ad) = (\mathtt{wl}_2^0(ad), M(ad))$

$M(ad) = \{\!\{(\mathtt{atp}_3(ad{\color{red}v}), \mathtt{wl}_2^0({\color{red}v}d), \mathtt{wl}_2^0(a{\color{red}v})) \mid {\color{red}v} \in V(G)\}\!\}$



$\mathtt{wl}_2^1(ac) = (\mathtt{wl}_2^0(ac), M(ac))$

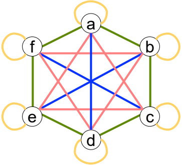
$M(ac) = \{\!\{(\mathtt{atp}_3(ac{\color{red}w}), \mathtt{wl}_2^0({\color{red}w}c), \mathtt{wl}_2^0(a{\color{red}w})) \mid {\color{red}w} \in V(G)\}\!\}$

When ${\color{red}w} = b$ then $\mathtt{atp}_3(acb) \neq \mathtt{atp}_3(ad{\color{red}v})$, for all ${\color{red}v}$.

Example

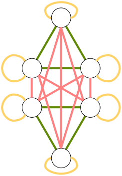
Consider $k = 2$.
$t = 0$ there are three colors:

Same node: $\mathtt{wl}_2^0(aa) = \cdots = \mathtt{wl}_2^0(ff)$
Edges: $\mathtt{wl}_2^0(ab) = \cdots = \mathtt{wl}_2^0(fa)$
Non-edges: $\mathtt{wl}_2^0(ac) = \mathtt{wl}_2^0(ad) = \cdots = \mathtt{wl}_2^0(fb)$



$t = 1$ distinguishes between ac and ad:
$\mathtt{wl}_2^1(ad) = (\mathtt{wl}_2^0(ad), M(ad))$
$M(ad) = \{\!\{(\mathtt{atp}_3(ad{\color{red}v}), \mathtt{wl}_2^0({\color{red}v}d), \mathtt{wl}_2^0(a{\color{red}v})) \mid {\color{red}v} \in V(G)\}\!\}$

$\mathtt{wl}_2^1(ac) = (\mathtt{wl}_2^0(ac), M(ac))$
$M(ac) = \{\!\{(\mathtt{atp}_3(ac{\color{red}w}), \mathtt{wl}_2^0({\color{red}w}c), \mathtt{wl}_2^0(a{\color{red}w})) \mid {\color{red}w} \in V(G)\}\!\}$



When ${\color{red}w} = b$ then $\mathtt{atp}_3(acb) \neq \mathtt{atp}_3(ad{\color{red}v})$, for all ${\color{red}v}$.

Color Refinement same as 1-WL

Fact

Color refinement and 1-WL coincide: $\forall t \geq 0,\ cr^t(v) = wl_1^t(v)$.

Color Refinement same as 1-WL

Fact

Color refinement and 1-WL coincide: $\forall t \geq 0,\ \mathcal{cr}^t(v) = \mathcal{wl}_1^t(v)$.

$\mathtt{cr}^{t+1}(v) =$
$(\mathtt{cr}^t(v), \{\!\{\mathtt{cr}^t(w) \mid w \in N(v)\}\!\})$

Color Refinement same as 1-WL

Fact

Color refinement and 1-WL coincide: $\forall t \geq 0,\ cr^t(v) = wl_1^t(v)$.

$\mathtt{cr}^{t+1}(v) =$
$(\mathtt{cr}^t(v), \{\!\{\mathtt{cr}^t(w) \mid w \in N(v)\}\!\})$

$\mathtt{wl}_1^{t+1}(v) \stackrel{\text{def}}{=} (\mathtt{wl}_1^t(v), M(v)),$ where
$M(v) \stackrel{\text{def}}{=} \{\!\{(\mathtt{atp}_2(vw), \mathtt{wl}_1^t(w)) \mid w \in V(G)\}\!\}$

Color Refinement same as 1-WL

Fact

Color refinement and 1-WL coincide: $\forall t \geq 0,\ cr^t(v) = wl_1^t(v)$.

$\mathtt{cr}^{t+1}(v) =$
$(\mathtt{cr}^t(v), \{\!\{\mathtt{cr}^t(w) \mid w \in N(v)\}\!\})$

$\mathtt{wl}_1^{t+1}(v) \stackrel{\text{def}}{=} (\mathtt{wl}_1^t(v), M(v))$, where
$M(v) \stackrel{\text{def}}{=} \{\!\{(\mathtt{atp}_2(vw), \mathtt{wl}_1^t(w)) \mid w \in V(G)\}\!\}$

M consists of three bags:

Color Refinement same as 1-WL

Fact

Color refinement and 1-WL coincide: $\forall t \geq 0$, $cr^t(v) = wl_1^t(v)$.

$\mathtt{cr}^{t+1}(v) =$
$(\mathtt{cr}^t(v), \{\!\{\mathtt{cr}^t(w) \mid w \in N(v)\}\!\})$

$\mathtt{wl}_1^{t+1}(v) \stackrel{\text{def}}{=} (\mathtt{wl}_1^t(v), M(v))$, where
$M(v) \stackrel{\text{def}}{=} \{\!\{(\mathtt{atp}_2(vw), \mathtt{wl}_1^t(w)) \mid w \in V(G)\}\!\}$

M consists of three bags:

$v = w$: then $M_0(v) = \{\!\{(\mathtt{wl}_1^t(v))\}\!\}$

Color Refinement same as 1-WL

Fact

Color refinement and 1-WL coincide: $\forall t \geq 0,\ cr^t(v) = wl_1^t(v)$.

$\mathtt{cr}^{t+1}(v) =$
$(\mathtt{cr}^t(v), \{\!\{\mathtt{cr}^t(w) \mid w \in N(v)\}\!\})$

$\mathtt{wl}_1^{t+1}(v) \stackrel{\text{def}}{=} (\mathtt{wl}_1^t(v), M(v))$, where
$M(v) \stackrel{\text{def}}{=} \{\!\{(\mathtt{atp}_2(vw), \mathtt{wl}_1^t(w)) \mid w \in V(G)\}\!\}$

M consists of three bags:

$v = w$: then $M_0(v) = \{\!\{(\mathtt{wl}_1^t(v))\}\!\}$

$vw \in E(G)$: then $M_1(v) = \{\!\{\mathtt{wl}_1^t(w) \mid w \in N(v)\}\!\}$

Color Refinement same as 1-WL

Fact

Color refinement and 1-WL coincide: $\forall t \geq 0,\ cr^t(v) = wl_1^t(v)$.

$\mathtt{cr}^{t+1}(v) =$
$(\mathtt{cr}^t(v), \{\!\{\mathtt{cr}^t(w) \mid w \in N(v)\}\!\})$

$\mathtt{wl}_1^{t+1}(v) \stackrel{\text{def}}{=} (\mathtt{wl}_1^t(v), M(v)),$ where
$M(v) \stackrel{\text{def}}{=} \{\!\{(\mathtt{atp}_2(vw), \mathtt{wl}_1^t(w)) \mid w \in V(G)\}\!\}$

M consists of three bags:

$v = w$: then $M_0(v) = \{\!\{(\mathtt{wl}_1^t(v))\}\!\}$

$vw \in E(G)$: then $M_1(v) = \{\!\{\mathtt{wl}_1^t(w) \mid w \in N(v)\}\!\}$

$vw \notin E(G)$: then $M_2(v) = \{\!\{\mathtt{wl}_1^t(w) \mid w \notin N(v)\}\!\}$

Color Refinement same as 1-WL

Fact

Color refinement and 1-WL coincide: $\forall t \geq 0,\ cr^t(v) = wl_1^t(v)$.

$\mathtt{cr}^{t+1}(v) =$
$(\mathtt{cr}^t(v), \{\!\{\mathtt{cr}^t(w) \mid w \in N(v)\}\!\})$

$\mathtt{wl}_1^{t+1}(v) \stackrel{\text{def}}{=} (\mathtt{wl}_1^t(v), M(v)),$ where
$M(v) \stackrel{\text{def}}{=} \{\!\{(\mathtt{atp}_2(vw), \mathtt{wl}_1^t(w)) \mid w \in V(G)\}\!\}$

M consists of three bags:

$v = w$: then $M_0(v) = \{\!\{(\mathtt{wl}_1^t(v))\}\!\}$

$vw \in E(G)$: then $M_1(v) = \{\!\{\mathtt{wl}_1^t(w) \mid w \in N(v)\}\!\}$

$vw \notin E(G)$: then $M_2(v) = \{\!\{\mathtt{wl}_1^t(w) \mid w \notin N(v)\}\!\}$

Color refinement counts the colors of the neighbors.

1-WL counts both neighbors and non-neighbors. Same thing!

Discussion

- k-WL identifies a graph G if it is isomorphic to any other graph with the same k-colors histogram.
- The WL-dimension of G is the smallest k that identifies it. Notice: $k \leq |V(G)|$.
- 1-WL identifies a random graph with high probability. But fails on any regular graph[3]
- 2-WL identifies a random regular graph with high probability. But fails on any strongly regular graph[4]
- For any k, there exists two graphs G_k, H_k that are not isomorphic but are indistinguishable by k-WL [Cai et al., 1992].
- Lots of classes of graphs have low WL-dimension, see [Morris et al., 2023].

[3] G is regular if all nodes have the same degree.

[4] `https://en.wikipedia.org/wiki/Strongly_regular_graph`

Babai, L., Erdös, P., and Selkow, S. M. (1980).
Random graph isomorphism.
SIAM J. Comput., 9(3):628–635.

Cai, J., Fürer, M., and Immerman, N. (1992).
An optimal lower bound on the number of variables for graph identification.
Comb., 12(4):389–410.

Grohe, M. (2020).
word2vec, node2vec, graph2vec, x2vec: Towards a theory of vector embeddings of structured data.
In Suciu, D., Tao, Y., and Wei, Z., editors, *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020*, pages 1–16. ACM.

Grohe, M. (2021).
The logic of graph neural networks.
In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pages 1–17. IEEE.

Morris, C., Lipman, Y., Maron, H., Rieck, B., Kriege, N. M., Grohe, M., Fey, M., and Borgwardt, K. M. (2023).
Weisfeiler and leman go machine learning: The story so far.
J. Mach. Learn. Res., 24:333:1–333:59.

Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G., and Grohe, M. (2019).
Weisfeiler and leman go neural: Higher-order graph neural networks.
In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 4602–4609. AAAI Press.