The Iteration Number of Colour Refinement

Sandra Kiefer
RWTH Aachen University, Germany
kiefer@cs.rwth-aachen.de

Brendan D. McKay
Australian National University, Canberra, Australia
brendan.mckay@anu.edu.au

Abstract

The Colour Refinement procedure and its generalisation to higher dimensions, the Weisfeiler-Leman algorithm, are central subroutines in approaches to the graph isomorphism problem. In an iterative fashion, Colour Refinement computes a colouring of the vertices of its input graph.

A trivial upper bound on the iteration number of Colour Refinement on graphs of order n is $n-1$. We show that this bound is tight. More precisely, we prove via explicit constructions that there are infinitely many graphs G on which Colour Refinement takes $|G|-1$ iterations to stabilise. Modifying the infinite families that we present, we show that for every natural number $n \geq 10$, there are graphs on n vertices on which Colour Refinement requires at least $n-2$ iterations to reach stabilisation.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Mathematics of computing → Combinatorial algorithms; Mathematics of computing → Graph theory

Keywords and phrases Colour Refinement, iteration number, Weisfeiler-Leman algorithm, quantifier depth

Digital Object Identifier 10.4230/LIPIcs.ICALP.2020.73

Category Track A: Algorithms, Complexity and Games

Related Version A full version of the paper is available at [21], `http://arxiv.org/abs/2005.10182`.

1 Introduction

Colour Refinement, which is also known as Naïve Vertex Classification or the 1-dimensional Weisfeiler-Leman algorithm (1-WL), is an important combinatorial algorithm in theoretical and practical approaches to the graph isomorphism problem. In an iterative fashion, it refines an isomorphism-invariant partition of the vertex set of the input graph. This process stabilises at some point and the final partition can often be used to distinguish non-isomorphic graphs [3]. Colour Refinement can be implemented to run in time $O((m+n)\log n)$, where n is the order of the input graph and m is its number of edges [6, 29]. Most notably, its efficient implementations are used in all competitive graph isomorphism solvers (such as `Nauty` and `Traces` [30], `Bliss` [19] and `saucy` [7]).

Colour Refinement has been rediscovered many times, one of its first occurences being in a paper on chemical information systems from the 1960s [31]. The procedure is applied in plenty of other fields, for example, it can be modified to reduce the dimension of linear programs significantly [14]. Other applications are in the context of graph kernels [34] or static program analysis [27]. A recently discovered connection to deep learning shows that the expressive power of Colour Refinement is captured by graph neural networks [32].

As described above, Colour Refinement computes a stable colouring of its input graph. It is known that two given graphs result in equal colourings, i.e. are not distinguished by Colour Refinement, if and only if there is a fractional isomorphism between them [12, 33, 35]. Moreover, the graphs which Colour Refinement identifies up to isomorphism (i.e. distinguishes from all non-isomorphic ones) have been completely characterised [2, 24].

To obtain its final colouring, the algorithm proceeds in iterations. In this paper, we investigate how many iterations it takes for the algorithm to terminate. More specifically, for $n \in \mathbb{N}$, we are interested in $\mathrm{WL}_1(n)$, the maximum number of iterations required to reach stabilisation of Colour Refinement among all graphs of order n.

While not directly linked to the running time on a sequential machine, the iteration number corresponds to the parallel running time of Colour Refinement (on a standard PRAM model) [17, 25]. Furthermore, via a connection to counting logics, a bound on the iteration number for graphs of a fixed size directly translates into a bound on the descriptive complexity of the difference between the two graphs, namely into a bound on the quantifier depth of a distinguishing formula in the extension of the 2-variable fragment of first-order logic by counting quantifiers [5, 18]. Moreover, the iteration number of 1-WL equals the depth of a graph neural network that outputs the stable vertex colouring of the underlying graph with respect to Colour Refinement [32].

Considering paths, one quickly determines that $\mathrm{WL}_1(n) \geq \frac{n}{2} - 1$ holds for every $n \in \mathbb{N}$. By contrast, on random graphs, the iteration number is asymptotically almost surely 2 [3]. The best published lower bound on the iteration number of Colour Refinement on n-vertex graphs is $n - O(\sqrt{n})$ [26]. Concerning the upper bound, the trivial inequality $\mathrm{WL}_1(n) \leq n - 1$ holds for every repeated partitioning of a set of size n and it does not take into account any further properties of the input graph or of the algorithm used to execute the partitioning. Still, no improvement over this upper bound has been established.

Our first main result reads as follows.

▶ **Theorem 1.** *For every $n \in \mathbb{N}_{\geq 10}$ with $n = 12$ or $n \bmod 18 \notin \{6, 12\}$, it holds that $\mathrm{WL}_1(n) = n - 1$.*

Thus, there are infinitely many $n \in \mathbb{N}$ with $\mathrm{WL}_1(n) = n - 1$. We can even determine the iteration number up to an additive constant of 1 for all $n \in \mathbb{N}$ (where the precise numbers for $n \leq 9$ can easily be determined computationally), as stated in our second main result.

▶ **Theorem 2.** *For every $n \in \mathbb{N}_{\geq 10}$, it holds that $\mathrm{WL}_1(n) \in \{n - 2, n - 1\}$.*

We obtain our bounds via an empirical approach. More precisely, we have designed a procedure that enables us to systematically generate for all $n \leq 64$ graphs of order n that obey certain constraints (to render the procedure tractable) and on which Colour Refinement takes $n - 1$ iterations to stabilise. Analysing the graphs, we have determined the connections between colour classes during the execution of the algorithm in detail. If the vertex degrees that are present in the graph are low, then the connections between colour classes of size 2 are restricted. This allows us to develop an elegant graphical visualisation and a compact string representation of the graphs with low vertex degrees that take $n - 1$ iterations to stabilise. Using these encodings, we are able to provide infinite families with $n - 1$ Colour Refinement iterations until stabilisation.

Our analysis enables a deep understanding of the families that we present. Via slight modifications of the graph families, we can then cover a large portion of graph sizes and, allowing to go from connected graphs to general graphs, we can construct the graphs that yield Theorem 2.

Due to space limits, we omit some of the proof details here and defer the reader to the full version for them [21].

Related work

Colour Refinement is the 1-dimensional version of the so-called Weisfeiler-Leman algorithm. For every $k \in \mathbb{N}$, there exists a generalisation of it (k-WL), which colours vertex k-tuples in the input graph instead of single vertices only. See [20] for an in-depth study of the main parameters of Colour Refinement and k-WL.

Similarly as for Colour Refinement, one can consider the number $\mathrm{WL}_k(n)$ of iterations of k-WL on graphs of order n. Notably, contrasting our results for Colour Refinement, in [22], it was first proved that the trivial upper bound of $\mathrm{WL}_2(n) \leq n^2 - 1$ is not even asymptotically tight (see also the journal version [23]). This foundation fostered further work, leading to an astonishingly good new upper bound of $O(n \log n)$ on the iteration number of 2-WL [28].

For fixed $k > 1$, it is already non-trivial to show linear lower bounds on $\mathrm{WL}_k(n)$. Modifying a construction of Cai, Fürer, and Immerman [5], this was achieved by Fürer [9], who showed that $\mathrm{WL}_k(n) \in \Omega(n)$, remaining to date the best known lower bound when the input is supposed to be a graph. Only when considering structures with relations of higher arity than 2 as input, better lower bounds on the iteration number of k-WL have been proved [4].

For $k > 2$, regarding upper bounds on the iteration number of k-WL, without further knowledge about the input graph, no significant improvements over the trivial $n^k - 1$ are known.[1] Still, when the input graph has bounded treewidth or is a 3-connected planar graph, polylogarithmic bounds on the iteration number of k-WL needed to identify the graph have been published [17, 36].

Although for every k, there are non-isomorphic graphs that are not distinguished by k-WL [5], it is known that for every graph class with a forbidden minor, a sufficiently high-dimensional Weisfeiler-Leman algorithm correctly decides isomorphism [13]. Recent results give new upper bounds on the dimension needed for certain interesting graph classes [15, 16]. A closely-related direction of research investigates what properties the Weisfeiler-Leman algorithm can detect in graphs [1, 8, 10].

2 Preliminaries

By $\mathbb{N}$, we denote the set of natural numbers, i.e. $\{1, 2, \dots\}$. We set $\mathbb{N}_0 \coloneqq \mathbb{N} \cup \{0\}$ and, for $k, \ell \in \mathbb{N}_0$, we define $[k, \ell] \coloneqq \{n \in \mathbb{N}_0 \mid k \leq n \leq \ell\}$ and $[k] \coloneqq [1, k]$. For a set S, a *partition* of S is a set Π of non-empty sets such that $\bigcup_{M \in \Pi} M = S$ and for all $M, M' \in \Pi$ with $M \neq M'$, it holds that $M \cap M' = \emptyset$. For two partitions Π and Π' of the same set S, we say that Π' is *finer* than Π (or Π' *refines* Π) if every element of Π' is a (not necessarily proper) subset of an element of Π. We write $\Pi \succeq \Pi'$ (and equivalently $\Pi' \preceq \Pi$) to express that Π' is finer than Π. Concurrently, we say that Π is *coarser* than Π'. Note that if both $\Pi \succeq \Pi'$ and $\Pi' \succeq \Pi$ hold, then $\Pi = \Pi'$.

For $S \neq \emptyset$, the partition $\{S\}$ is the *unit* partition of S. The partition $\big\{\{s\} \mid s \in S\big\}$ is called the *discrete partition* of S. A set of cardinality 1 is a *singleton*.

All graphs that we consider in this paper are finite and simple, i.e. undirected without self-loops at vertices. For a graph with vertex set $V(G)$ and edge set $E(G)$, its *order* is $|G| \coloneqq |V(G)|$. For a vertex $v \in V(G)$, we denote by $N(v)$ the neighbourhood of v in G, i.e. the set $\{w \mid \{v, w\} \in E(G)\}$. Similarly, for a vertex set W, we set $N(W) \coloneqq \big\{v \mid v \notin W, \exists w \in W \colon v \in N(w)\big\}$. The *degree* of a vertex v is $\deg(v) \coloneqq |N(v)|$. (Since the graph G

[1] The bound $n^k - 1$ is not tight, since the initial partition of the k-tuples already has multiple classes, for example, one consisting of all tuples of the form $(v, v, \dots, v)$.

will be clear from the context, we do not need to include it in our notation.) We also set $\deg(G) \coloneqq \{\deg(v) \mid v \in V(G)\}$. If there is a $d \in \mathbb{N}_0$ such that $\deg(G) = \{d\}$, the graph G is d-regular. A *regular* graph is a graph that is d-regular for some $d \in \mathbb{N}_0$. By a *matching*, we mean a 1-regular graph.

Let G be a graph with at least two vertices. If there are sets $\emptyset \neq P, Q \subseteq V(G)$ such that $V(G) = P \dot\cup Q$ and $P \cap Q = \emptyset$ and $E(G) \cap \{\{v, w\} \mid v, w \in P\} = \emptyset = E(G) \cap \{\{v, w\} \mid v, w \in Q\}$, then G is *bipartite (on bipartition (P, Q))*. If, additionally, $\big\{\{v, w\} \mid v \in P, w \in Q\big\} = E(G)$, the graph G is *complete bipartite*.

For $k, \ell \in \mathbb{N}_0$, a *(k, ℓ)-biregular graph (on bipartition (P, Q))* is a bipartite graph on bipartition (P, Q) such that for every $v \in P$, it holds that $|N(v)| = k$, and for every $w \in Q$, it holds that $|N(w)| = \ell$. A *biregular* graph is a graph G for which there are $P, Q \subseteq V(G)$ and $k, \ell \in \mathbb{N}_0$ such that G is (k, ℓ)-biregular on bipartition (P, Q).

For a graph G and a set $V' \subseteq V(G)$, we let $G[V']$ be the *induced subgraph of G on V'*, i.e. the subgraph of G with vertex set V' and edge set $E(G) \cap \{\{v, w\} \mid v, w \in V'\}$. We define $G - V' \coloneqq G[V(G) \setminus V']$. Furthermore, for vertex sets $V_1, V_2 \subseteq V(G)$, we denote by $G[V_1, V_2]$ the graph with vertex set $V_1 \cup V_2$ and edge set $E(G) \cap \{\{v_1, v_2\} \mid v_1 \in V_1, v_2 \in V_2\}$.

A *coloured graph* is a tuple (G, λ), where G is a graph and $\lambda \colon V(G) \to \mathcal{C}$ is a function that assigns colours (i.e. elements from a particular set $\mathcal{C}$) to the vertices. We interpret all graphs treated in this paper as coloured graphs. If the colouring is not specified, we assume a monochromatic colouring, i.e. all vertices have the same colour.

For a coloured graph G with colouring λ, a *(vertex) colour class* of G is a maximal set of vertices that all have the same λ-colour. Every graph colouring λ induces a partition $\pi(\lambda)$ of $V(G)$ into the vertex colour classes with respect to λ.

3 Colour Refinement

Colour Refinement proceeds by iteratively refining a partition of the vertex set of its input graph until the partition is stable with respect to the refinement criterion.

▶ **Definition 3** (Colour Refinement). *Let $\lambda \colon V(G) \to \mathcal{C}$ be a colouring of the vertices of a graph G, where $\mathcal{C}$ is some set of colours. The colouring computed by Colour Refinement on input (G, λ) is defined recursively: we set $\chi_G^0 \coloneqq \lambda$, i.e. the initial colouring is λ. For $i \in \mathbb{N}$, the colouring χ_G^i computed by Colour Refinement after i iterations on G is defined as $\chi_G^i(v) \coloneqq \big(\chi_G^{i-1}(v), \{\!\{\chi_G^{i-1}(w) \mid w \in N(v)\}\!\}\big)$.*

That is, $\chi_G^i(v)$ consists of the colour of v from the previous iteration as well as the multiset of colours of neighbours of v from the previous iteration. It is not difficult to see that $\pi(\chi_G^{i-1}) \succeq \pi(\chi_G^i)$ holds for every graph G and every $i \in \mathbb{N}$. Therefore, there is a unique minimal integer j such that $\pi(\chi_G^j) = \pi(\chi_G^{j+1})$. For this value j, we define the *output* of Colour Refinement on input G to be $\chi_G \coloneqq \chi_G^j$ and call χ_G and $\pi(\chi_G)$ *the stable colouring* and *the stable partition*, respectively, of G. Accordingly, executing *i Colour Refinement iterations* on G means computing the colouring χ_G^i. We call a graph G with initial colouring λ and the induced partition $\pi(\lambda)$ *stable* if $\pi(\lambda) = \pi(\chi_G)$. Note that if λ is stable, then for all $P, Q \in \pi(\lambda)$ with $P \neq Q$, the graph $G[P]$ is regular and the graph $G[P, Q]$ is biregular.

Colour Refinement can be used to check whether two given graphs G and G' are non-isomorphic by computing the stable colouring on the disjoint union of the two. If there is a colour C such that, in the stable colouring, the numbers of vertices of colour C differ in G and G', they are non-isomorphic. However, even if they agree in every colour class size in the stable colouring, the graphs might not be isomorphic. It is non-trivial to describe for which graphs this isomorphism test is always successful (see [2, 24]).

▶ **Notation 4.** *We write* $\mathrm{WL}_1(G)$ *for the number of iterations of Colour Refinement on input* G*, that is,* $\mathrm{WL}_1(G) = j$*, where* j *is the minimal integer for which* $\pi(\chi_G^j) = \pi(\chi_G^{j+1})$*. Similarly, for* $n \in \mathbb{N}$*, we write* $\mathrm{WL}_1(n)$ *to denote the maximum number of iterations that Colour Refinement needs to reach stabilisation on an* n*-vertex graph.*

We call every graph G with $\mathrm{WL}_1(G) = |G| - 1$ a *long-refinement graph.*

▶ **Fact 5.** *Let* G *be an uncoloured path with* n *vertices. Then* $\mathrm{WL}_1(G) = \lfloor \frac{n-1}{2} \rfloor$.

Proof sketch. In the first iteration, the two end vertices are distinguished from all others because they are the only ones with degree 1. Then in each iteration, the information of being adjacent to a "special" vertex, i.e. the information about the distance to a vertex of degree 1, is propagated one step closer to the vertices in the centre of the path. This procedure takes $\lfloor \frac{n-1}{2} \rfloor$ iterations. ◀

In 2015, Krebs and Verbitsky improved on the explicit linear lower bound for graphs of order n given by Fact 5 by constructing a family of pairs of graphs whose members of order n can only be distinguished after $n - 8\sqrt{n}$ Colour Refinement iterations (see [26, Theorem 4.6]). Hence, since for a set $\{v_1, \ldots, v_n\} =: S$ and every sequence $\pi_1, \ldots, \pi_\ell$ of partitions of S that satisfy

$$\pi_1 \gneqq \pi_2 \gneqq \cdots \gneqq \big\{\{v_1\}, \ldots \{v_n\}\big\} = \pi_\ell,$$

it holds that $\ell \leq n - 1$, we obtain the following corollary.

▶ **Corollary 6.** *For every* $n \in \mathbb{N}$*, it holds that* $n - 8\sqrt{n} \leq \mathrm{WL}_1(n) \leq n - 1$.

It has remained open whether any of the two bounds is tight. In preliminary research conducted together with Gödicke and Schweitzer, towards improving the lower bound, the first author took up an approach to reverse-engineer the splitting of colour classes. Gödicke's implementation of those split procedures led to the following result.

▶ **Theorem 7** ([11])**.** *For every* $n \in \{1, 10, 11, 12\}$*, it holds that* $\mathrm{WL}_1(n) = n - 1$*. For* $n \in [2, 9]$*, it holds that* $\mathrm{WL}_1(n) < n - 1$.

Unfortunately, due to computational exhaustion, it was not possible to test for larger graph sizes. Also, the obtained graphs do not exhibit any structural properties that would lend themselves for a generalisation to larger orders. Using a fast implementation of Colour Refinement, we could verify that there are exactly 16 long-refinement graphs of order 10, 24 long-refinement graphs of order 11, 32 of order 12, and 36 of order 13. However, again, with simple brute-force approaches, we could not go beyond this number exhaustively.

4 Compact Representations of Long-Refinement Graphs

In the light of the previous section, the question whether the lower bound obtained by Krebs and Verbitsky is asymptotically tight has remained open. With the brute-force approach, it becomes infeasible to test all graphs of orders much larger than 10 exhaustively for their number of Colour Refinement iterations until stabilisation. Still, knowing that there exist long-refinement graphs, it is natural to ask whether the ones presented in [11] are exceptions or whether there are infinitely many such graphs. We show that the latter is the case.

When the input is a coloured graph with at least two vertex colours, the initial partition already has two elements. Hence, all long-refinement graphs are monochromatic. Therefore, in the following, all initial input graphs are considered to be monochromatic.

▶ **Proposition 8.** *Let G be a graph and let $n \coloneqq |G|$. If there exists an $i \in \mathbb{N}_0$ such that $|\{\chi_G^{i+1}(v) \mid v \in V(G)\}| - |\{\chi_G^i(v) \mid v \in V(G)\}| \geq 2$ holds, then G is not a long-refinement graph.*

Proof. Every pair of partitions π, π' with $\pi \supsetneq \pi'$ satisfies $|\pi'| \geq |\pi| + 1$. Thus, every sequence of partitions of the form

$$\pi_1 \coloneqq \big\{\{v_1, \ldots, v_n\}\big\} \supsetneq \pi_2 \supsetneq \cdots \supsetneq \big\{\{v_1\}, \ldots \{v_n\}\big\} =: \pi_n$$

must satisfy $|\pi_i| = |\pi_{i-1}| + 1$ for all $i \in [2, n]$. ◀

The proposition implies that, in order to find long-refinement graphs, we have to look for graphs in which, in every Colour Refinement iteration, only one additional colour class appears. That is, in each iteration, only one colour class is split and the splitting creates exactly two new colour classes.

▶ **Corollary 9.** *Let G be a long-refinement graph with at least two vertices. Then there exist $d_1, d_2 \in \mathbb{N}_0$ with $d_1 \neq d_2$ and such that $\deg(G) = \{d_1, d_2\}$.*

Proof. This is a direct consequence of Proposition 8: every (monochromatic) regular graph G satisfies $\mathrm{WL}_1(G) = 0$ and if there were more than two vertex degrees present in G, we would have $|\{\chi_G^1(v) \mid v \in V(G)\}| - |\{\chi_G^0(v) \mid v \in V(G)\}| \geq 3 - 1 \geq 2$. ◀

We can thus restrict ourselves to graphs with exactly two vertex degrees.

▶ **Notation 10.** *For a graph G and $i \in \mathbb{N}_0$, we let π_G^i denote the partition induced by χ_G^i on $V(G)$, i.e. after i Colour Refinement iterations on G. If G is clear from the context, we omit it in the expression.*

As a result of the regularity conditions that must hold for the bipartite graph $G[V_1, V_2]$, we make the following observation. It implies that, in a long-refinement graph, to determine the class C that is split in iteration i, it suffices to consider the neighbourhood of an arbitrary class obtained in the preceding iteration.

▶ **Lemma 11.** *Let G be a graph. Suppose there are $i \in \mathbb{N}$ and C_1, C_2, C' with $\pi^i \setminus \pi^{i-1} = \{C_1, C_2\}$ and $C' \in \pi^i \setminus \pi^{i+1}$. Then there are vertices $v_1', v_2' \in C'$ such that $|N(v_1') \cap C_1| \neq |N(v_2') \cap C_1|$.*

Proof. Note that there must be a $C \in \pi^{i-1} \setminus \pi^i$ with $C_1 \cup C_2 = C$. Since $C' \in \pi^i$ and $C \in \pi^{i-1}$, there is a $d \in \mathbb{N}_0$ such that for every $v \in C'$, it holds that $d = |N(v) \cap C|$. Since $\{C_1, C_2\} = \pi^i \setminus \pi^{i-1}$ and $C' \notin \pi^{i+1}$, there are vertices $v_1', v_2' \in C'$ such that $|N(v_1') \cap C_1| \neq |N(v_2') \cap C_1|$ or $|N(v_1') \cap C_2| \neq |N(v_2') \cap C_2|$. In the first case, we are done. In the second case, we obtain $|N(v_1') \cap C_1| = d - |N(v_1') \cap C_2| \neq d - |N(v_2') \cap C_2| = |N(v_2') \cap C_1|$. ◀

Note that the validity of the lemma depends on the assumption $\{C_1, C_2\} = \pi^i \setminus \pi^{i-1}$, which by Proposition 8 is always fulfilled in long-refinement graphs as long as $\pi^{i-1} \neq \pi^i$.

▶ **Corollary 12.** *No graph with more than one connected component is a long-refinement graph.*

Proof. By Corollary 9, the graph cannot have isolated vertices, since the subgraph formed by the other connected components would have to be d-regular for some $d \geq 1$. Therefore, the stable partition would not be discrete.

Suppose G is a long-refinement graph in which every connected component has size at least 2. Then there is an iteration i for which there is a unique connected component with vertex set V' such that the current partition is discrete on V' but not discrete on the other connected components. Then the vertex sets C_1, C_2 from Lemma 11 must be subsets of V'. However, the lemma implies that no further classes are split because none is adjacent to C_1 or C_2. ◀

We can therefore restrict ourselves to connected graphs. The only connected graphs G with $\deg(G) = \{1, 2\}$ are paths and, by Fact 5, they are not long-refinement graphs. Thus, the smallest degree pairs for a search for candidates are $\{1, 3\}$ and $\{2, 3\}$.

▶ **Lemma 13.** *Let G be a long-refinement graph. Then $|\{v \in V(G) \mid \deg(v) = 1\}| \leq 2$.*

Table 1 displays the adjacency lists of two long-refinement graphs on 12 and 14 vertices, respectively, which each have exactly one vertex of degree 1.

Table 1 Adjacency lists of long-refinement graphs G with $\deg(G) = \{1, 5\}$ (left) and $\deg(G) = \{1, 3\}$ (right), respectively.

v	$N(v)$
0	1
1	0,2,3,4,5
2	1,3,5,7,10
3	1,2,4,6,10
4	1,3,5,9,11
5	1,2,4,8,11

v	$N(v)$
6	3,7,8,9,11
7	2,6,8,9,10
8	5,6,7,10,11
9	4,6,7,10,11
10	2,3,7,8,9
11	4,5,6,8,9

v	$N(v)$
0	1
1	0,2,3
2	1,11,13
3	1,10,12
4	5,7,10
5	4,6,10
6	5,9,11

v	$N(v)$
7	4,8,11
8	7,9,13
9	6,8,12
10	3,4,5
11	2,6,7
12	3,9,13
13	2,8,12

The lemma allows us to reduce the decision problem whether there are infinitely many long-refinement graph with degrees in $\{1, 2, 3\}$ to the question whether there are such families with degrees in $\{2, 3\}$. The proof of the lemma as well as this reduction can be found in [21]. With the help of the tool `Nauty` [29], our quest for long-refinement graphs with degrees 2 and 3 was successful. Exploiting the degree restrictions and some other conditions that we imposed to render the search tractable, it was possible to test for graphs up to order 64. We found graphs G with $n - 1$ Colour Refinement iterations, where $n = |G|$, for all even $n \in [10, 64] \setminus \{24, 30, 42, 48, 60\}$ and for all odd $n \in [11, 63] \setminus \{21, 27, 39, 45, 57, 63\}$.

In the following, in order to generalise the results, we analyse the obtained graphs. Among our computational results, the even-order graphs G have the following property in common: there is an iteration j such that for every $c \in \chi^j_G(V(G))$, it holds that $|\{v \in V(G) \mid \chi^j_G(v) = c\}| = 2$. That is, with respect to their assigned colours, the vertices remain in pairs until there are no larger colour classes left. Then the first such pair is split into singletons, which must induce a splitting of another pair, and so on, until the discrete partition is obtained. (Similar statements hold for the odd-order long-refinement graphs, but are more technical.) In the following, a *pair* is a set of two vertices which occurs as a colour class during the execution of Colour Refinement. That is, vertices v, v' form a pair if and only if $\{v, v'\}$ is an element of π^i for some $i \in \mathbb{N}_0$.

As just argued, there is a splitting order on the pairs, i.e., a linear order $\prec$ induced by the order in which pairs are split into singletons. We now examine the possible connections between pairs.

From now on, we make the following assumption.

▶ **Assumption 14.** *G is a long-refinement graph with* $\deg(G) = \{2, 3\}$ *and such that there is an* $i \in \mathbb{N}_0$ *for which* π^i *contains only pairs. Let* $\prec$ *be the splitting order of these pairs.*

We call pairs $P_1, P_2 \subseteq V(G)$ *successive* if P_2 is the successor of P_1 with respect to $\prec$. Note that for successive pairs P_1, P_2, in the graph $G[P_1, P_2]$, every $v_2 \in P_2$ must have the same number of neighbours in P_1, otherwise it would hold that $P_2 \prec P_1$. By a simple case analysis, together with an application of Lemma 11, this rules out all connections but matchings for successive pairs.

▶ **Corollary 15.** *Let P_1 and P_2 be successive pairs. Then $G[P_1, P_2]$ is a matching.*

Towards a compact representation of the graphs, we further examine the connections between pairs P_1 and P_2 with $S(P_1) \prec P_2$, where $S(P_1)$ is the successor of P_1 with respect to $\prec$.

▶ **Lemma 16.** *Let P_1 be a pair. Then exactly one of the following holds.*

- *$P_1 \neq \min(\prec)$ and for every pair P_2 with $S(P_1) \prec P_2$, it holds that $E(G[P_1, P_2]) = \emptyset$.*
- *$P_1 = \min(\prec)$ and there are exactly two choices P_2, P_2' for a pair P' with $S(P_1) \prec P'$ such that $E(G[P_1, P']) \neq \emptyset$. Furthermore, there is a vertex $v_1 \in P_1$ such that $G[\{v_1\}, P_2]$ and $G[P_1 \setminus \{v_1\}, P_2']$ are complete bipartite and $E(G[\{v_1\}, P_2']) = E(G[P_1 \setminus \{v_1\}, P_2]) = \emptyset$.*

Proof. Suppose $P_1 \neq \min(\prec)$. If $P_1 = \max(\prec)$, the statement trivially holds. Otherwise, by Corollary 15, every vertex $v_1 \in P_1$ has exactly one neighbour in $S(P_1)$ and exactly one neighbour in the predecessor of P_1, i.e. in the unique pair $A(P_1)$ such that $P_1 = S(A(P_1))$. Thus, due to the degree restrictions, v_1 can have at most one additional neighbour in a pair P' with $P_1 \prec P'$ and $P' \neq S(P_1)$. However, if v_1 had a neighbour in such a P', the graph $G[\{v_1\}, P']$ would not be biregular, implying that $P' = S(P_1)$, a contradiction. Therefore, $N(v_1) \subseteq A(P_1) \cup P_1 \cup S(P_1)$ and thus, $N(P_1) \subseteq A(P_1) \cup S(P_1)$. In particular, for every pair P_2 with $P_1 \prec P_2$ and $P_2 \neq S(P_1)$, it holds that $E(G[P_1, P_2]) = \emptyset$.

Now suppose that $P_1 = \min(\prec)$. Since the splitting of P_1 must be induced by a splitting of a union of two pairs and $G[P_1, S(P_1)]$ is biregular and $G[P_1]$ is regular, we cannot have $N(P_1) \subseteq S(P_1)$. Thus, there is a pair P_2 with $S(P_1) \prec P_2$ and such that $E(G[P_1, P_2]) \neq \emptyset$. Let $v_1 \in P_1$ be a vertex with $N(v_1) \cap P_2 \neq \emptyset$. Then $P_2 \subseteq N(v_1)$, otherwise $P_2 = S(P_1)$. Thus, $G[\{v_1\}, P_2]$ is complete bipartite. Therefore and due to the degree restrictions, v_1 has exactly three neighbours: one in $S(P_1)$ and two in P_2. In particular, for every pair P_2' with $P_2 \neq P_2' \neq S(P_1)$, it holds that $E(G[\{v_1\}, P_2']) = \emptyset$.

Let $v_1' \neq v_1$ be the second vertex in P_1. Since the splitting of P_1 induces the splitting of $S(P_1)$, by Proposition 8, for every pair P' with $P_1 \neq P' \neq S(P_1)$, the graph $G[\{v_1'\}, P']$ must be biregular, i.e. either empty or complete bipartite.

Moreover, since $\deg(v_1) = 3$, also $\deg(v_1') = 3$. By Corollary 15, $|N(v_1') \cap S(P_1)| = 1$. Therefore, there is exactly one pair P_2' such that $G[\{v_1'\}, P_2']$ is complete bipartite and for all other pairs P' with $P_1 \neq P' \neq S(P_1)$, the graph $G[\{v_1'\}, P']$ is empty.

Suppose $P_2' = P_2$. Choose i such that $\pi^i \setminus \pi^{i+1} = \{P_1\}$. Then the unique element in $\pi^{i-1} \setminus \pi^i$ is a union of two pairs, whose splitting induces the splitting of P_1. However, $N(P_1) = S(P_1) \cup P_2$ and both graphs $G[P_1, S(P_1)]$ and $G[P_1, P_2]$ are biregular.

Thus, $P_2' \neq P_2$, which concludes the proof. ◀

Corollary 15 and Lemma 16 characterise $G[P_1, P_2]$ for all pairs $P_1 \neq P_2$. Thus, all additional edges must be between vertices from the same pair. Hence, we can use the following compact graphical representation to fully describe the graphs of order at least 12

that we found. As the set of nodes, we take the pairs. We order them according to $\prec$ and connect successive pairs with an edge representing the matching. If the two vertices of a pair are adjacent, we indicate this with a loop at the corresponding node. The only other type of connection between pairs is constituted by the edges from $\min(\prec)$ to two other pairs which form the last colour class of size 4, i.e. a colour class of size 4 in the partition π^i for which $\pi^{i+1} \setminus \pi^{i+2} = \{\min(\prec)\}$. We indicate this type of edge with a dotted curve.

An example graph as well as the evolution of the colour classes computed by Colour Refinement on the graph is depicted in Figure 1.

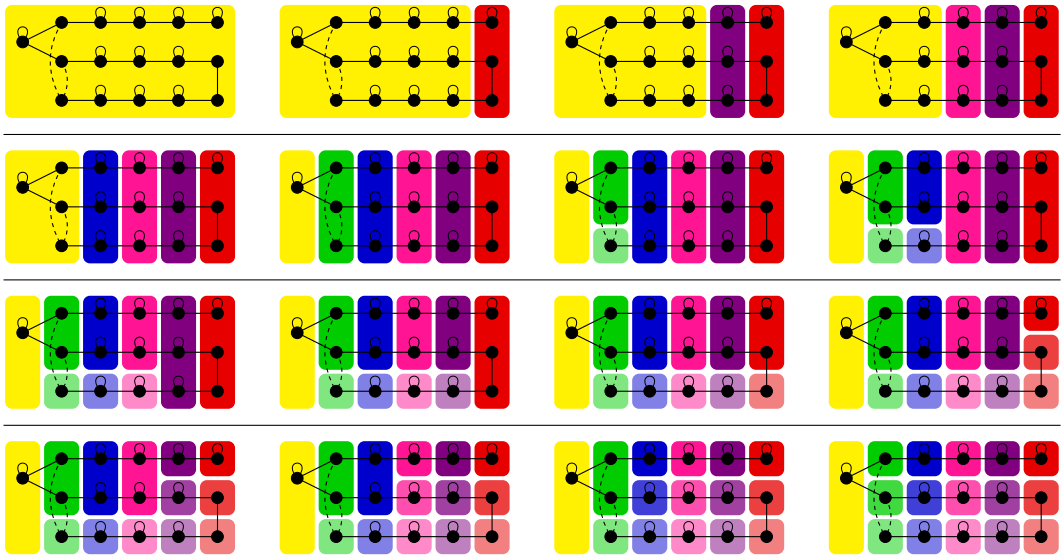

Figure 1 Top left: A long-refinement graph G on 32 vertices. The subsequent pictures show the partitions of $V(G)$ after the first 15 Colour Refinement iterations. There are 16 further iterations not depicted here, which consist in the splitting of the pairs into singletons.

In fact, there is an even more compact representation for our even-order long-refinement graphs.

▶ **Notation 17.** *Since $\prec$ is a linear order, we can also use a string representation to fully describe the graphs. For this, we introduce the following notation, letting $A(P)$ and $S(P)$ be the predecessor and successor of P, respectively, with respect to $\prec$.*

- *0 represents a pair of vertices of degree 2.*
- *1 represents a pair P of vertices of degree 3 that is not the minimum of $\prec$ and for which $N(P) \subseteq A(P) \cup S(P)$. (This implies that $P \in E(G)$.)*
- *X represents a pair P of vertices of degree 3 that is not the minimum of $\prec$ and for which $N(P) \not\subseteq A(P) \cup S(P)$.*
- *S represents the minimum of $\prec$.*

Thus, by Lemma 16, there are exactly two pairs of type X, namely P_2 and P_2' from the lemma. Now we can use the alphabet $\Sigma = \{0, 1, \mathrm{S}, \mathrm{X}\}$ and the order $\prec$ to encode the graphs in strings. The i-th letter of a string is the i-th element of $\prec$. Note that S is always a pair of non-adjacent vertices of degree 3 due to the degree restrictions. For example, the string representation for the graph in Figure 1 is S11100111X1X1110.

Formally, for every $\ell \geq 3$ and every string $\Xi\colon [\ell] \to \{0,1,\mathrm{S},\mathrm{X}\}$ with $\Xi(1) = \mathrm{S}$ and $\Xi^{-1}(\mathrm{X}) = \{r, r'\}$ for some $r, r' \in [\ell]$ with $r < r'$, we define the corresponding graph $G \coloneqq G(\Xi)$ with $V(G) = \{v_{i,j} \mid i \in [\ell], j \in [2]\}$ and

$$\begin{aligned} E(G) = &\big\{\{v_{i,1}, v_{i,2}\} \,\big|\, i \in [\ell], \Xi(i) = 1\big\} \cup \\ &\big\{\{v_{i,j}, v_{i+1,j}\} \,\big|\, i \in [\ell-1], j \in [2]\big\} \cup \\ &\big\{\{v_{r,j}, v_{1,1}\} \,\big|\, j \in [2]\big\} \cup \\ &\big\{\{v_{r',j}, v_{1,2}\} \,\big|\, j \in [2]\big\}. \end{aligned}$$

We use this encoding in the next section, which contains our main results.

5 Infinite Families of Long-Refinement Graphs

In this section, we present infinite families of long-refinement graphs. We adapt them further to deduce that $\mathrm{WL}_1(n) \geq n - 2$ holds for all $n \in \mathbb{N}_{\geq 10}$.

For $w \in \{0,1\}^*$, the notation $(w)^k$ abbreviates the k-fold concatenation of w. We let $1^k \coloneqq (1)^k$.

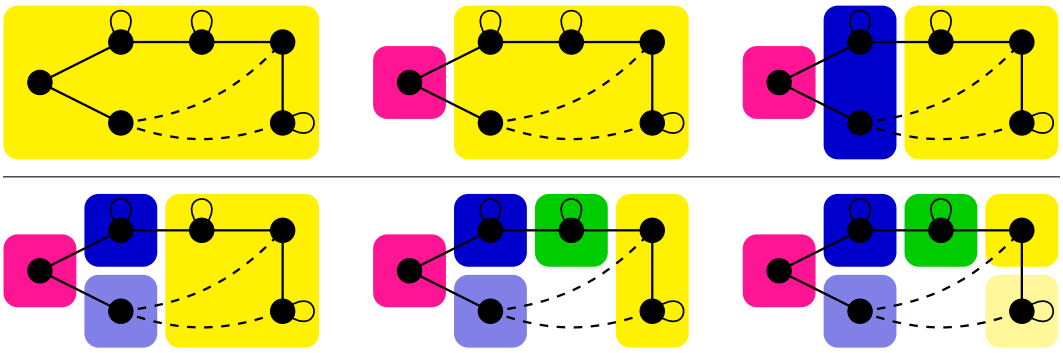

Figure 2 A visualisation of the graph with string representation S011XX and the evolution of the colour classes in the first 5 Colour Refinement iterations on the graph.

▶ **Theorem 18.** *For every string Ξ contained in the following sets, the graph $G(\Xi)$ is a long-refinement graph.*

- $\{\mathrm{S011XX}\}$
- $\{\mathrm{S}1^k001^k\mathrm{X}1\mathrm{X}1^k0 \mid k \in \mathbb{N}_0\}$
- $\{\mathrm{S}1^k11001^k\mathrm{XX}1^k0 \mid k \in \mathbb{N}_0\}$
- $\{\mathrm{S}1^k0011^k\mathrm{XX}1^k10 \mid k \in \mathbb{N}_0\}$
- $\{\mathrm{S}011(011)^k00(110)^k\mathrm{XX}(011)^k0 \mid k \in \mathbb{N}_0\}$
- $\{\mathrm{S}(011)^k00(110)^k1\mathrm{X}0\mathrm{X}1(011)^k0 \mid k \in \mathbb{N}_0\}$

We only present the parts of the proof that are most essential for an intuition why the graphs are indeed long-refinement graphs. The full proof can be found in [21].

Proof. Let $G \coloneqq G(\mathrm{S011XX})$ (cf. Figure 2). The vertices $v_{2,1}$ and $v_{2,2}$ are the only ones of degree 2. Thus,

$$\begin{aligned} \pi^1 &= \big\{\{v_{2,1}, v_{2,2}\}, V(G) \setminus \{v_{2,1}, v_{2,2}\}\big\}, \\ \pi^2 &= \big\{\{v_{2,1}, v_{2,2}\}, \{v_{i,j} \mid i \in \{1,3\}, j \in [2]\}, \{v_{i,j} \mid i \in [4,6], j \in [2]\}\big\}. \end{aligned}$$

Then

$$\pi^3 = \big\{\{v_{1,1}, v_{1,2}\}, \{v_{2,1}, v_{2,2}\}, \{v_{3,1}, v_{3,2}\}, \{v_{i,j} \mid i \in [4,6], j \in [2]\}\big\},$$

since the vertices in the S-pair have no neighbours in $\{v_{i,j} \mid i \in \{1,3\}, j \in [2]\}$. Similarly,

$$\pi^4 = \big\{\{v_{1,1}, v_{1,2}\}, \{v_{2,1}, v_{2,2}\}, \{v_{3,1}, v_{3,2}\}, \{v_{4,1}, v_{4,2}\}, \{v_{i,j} \mid i \in [5,6], j \in [2]\}\big\},$$
$$\pi^5 = \big\{\{v_{i,j} \mid j \in [2]\} \mid i \in [6]\big\}.$$

Now the splitting of the last colour class of size 4 into two X-pairs induces the splitting of the S-pair into singletons, which is propagated linearly according to $\prec$, adding 6 further iterations, thus summing up to 11 iterations.

We now consider the various infinite families of graphs. The proofs for them work similarly by induction over k. Therefore, we only present the full detailed proof for the family $\{\mathrm{S1}^k001^k\mathrm{X1X1}^k0 \mid k \in \mathbb{N}_0\}$, which includes the graph from Figure 1.

For $k = 0$, the graph $G_0 := G(\mathrm{S00X1X0})$ has 14 vertices. It is easy to verify that it indeed takes 13 Colour Refinement iterations to stabilise. We sketch how Colour Refinement processes the graph: for this, for $i \in \mathbb{N}_0$, we let π_0^i denote the partition of $V(G_0)$ induced by $\chi_{G_0}^i$, i.e. after i iterations of Colour Refinement on G_0. First, vertices are assigned colours indicating their degrees. That is,

$$\pi_0^1 = \big\{\{v_{i,j} \mid i \in \{2,3,7\}, j \in [2]\}, \{v_{i,j} \mid i \in \{1,4,5,6\}, j \in [2]\}\big\}.$$

Now

$$\pi_0^2 = \big\{\{v_{i,j} \mid i \in \{2,3,7\}, j \in [2]\}, \{v_{i,j} \mid i \in \{1,4,6\}, j \in [2]\}, \{v_{5,i} \mid i \in [2]\}\big\},$$

since the vertices contained in the 1-pair are not adjacent to vertices from 0-pairs. Since no vertex contained in the S-pair is adjacent to any vertex from the 1-pair, we obtain

$$\pi_0^3 = \big\{\{v_{i,j} \mid i \in \{2,3,7\}, j \in [2]\}, \{v_{i,j} \mid i \in \{4,6\}, j \in [2]\}, \{v_{5,i} \mid j \in [2]\},$$
$$\{v_{1,j} \mid j \in [2]\}\big\}.$$

Furthermore,

$$\pi_0^4 = \big\{\{v_{i,j} \mid i \in \{3,7\}, j \in [2]\}, \{v_{i,j} \mid i \in \{4,6\}, j \in [2]\}, \{v_{5,i} \mid j \in [2]\},$$
$$\{v_{1,j} \mid j \in [2]\}, \{v_{2,j} \mid j \in [2]\}\big\},$$
$$\pi_0^5 = \big\{\{v_{7,j} \mid j \in [2]\}, \{v_{i,j} \mid i \in \{4,6\}, j \in [2]\}, \{v_{5,i} \mid j \in [2]\},$$
$$\{v_{1,j} \mid j \in [2]\}, \{v_{2,j} \mid j \in [2]\}, \{v_{3,j} \mid j \in [2]\}\big\},$$
$$\pi_0^6 = \big\{\{v_{i,j} \mid j \in [2]\} \mid i \in [7]\big\},$$

i.e. with respect to the order $\prec$ induced by the string representation, the first 0-pair, the second 0-pair, and the first X-pair are separated from the others. Once the two X-pairs form separate colour classes, this induces the splitting of S into two singletons, which is propagated linearly through the entire string, adding 7 further iterations, thus summing up to 13 iterations.

For general $k \geq 1$, let $G_k := G(\mathrm{S1}^k001^k\mathrm{X1X1}^k0)$. To count the iterations of Colour Refinement, we introduce some vocabulary for the pairs in G_k (see also Figure 1). We let $V := \{v_{i,j} \mid i \in [2, k{+}1] \cup [k{+}4, 2k{+}3] \cup [2k{+}7, 3k{+}6], j \in [2]\}$. Note that V is the set of vertices contained in the subgraphs corresponding to the substrings 1^k in the string representation.

Furthermore, for all $i \in [k+2]$, we call the set $\{v_{i',j} \mid i' \in \{i, 2k+5-i, 2k+5+i\}, j \in [2]\}$ the *i-th column* and denote it by V_i. The *0-th column* is the set $\{v_{2k+5,j} \mid j \in [2]\}$. Thus,

$$V = \bigcup_{2 \leq i \leq k+1} V_i.$$

For every $j \in [2]$, the sets $\{v_{i,j} \mid i \in [1, k+2]\}$, $\{v_{i,j} \mid i \in [k+3, 2k+4]\}$, and $\{v_{i,j} \mid i \in [2k+6, 3k+7]\}$ are called *rows*. In accordance with Figure 1, we fix an ordering on the rows: the *first row* is $\{v_{i,1} \mid i \in [2k+6, 3k+7]\}$, the *second row* is $\{v_{i,2} \mid i \in [2k+6, 3k+7]\}$, ..., the *sixth row* is $\{v_{i,2} \mid i \in [1, k+2]\}$. To be able to refer to the vertices in V and the adjacent columns more easily, we relabel them: for $i \in [k+2], j \in [6]$, the vertex $w_{i,j}$ is defined to be the unique vertex in the i-th column and the j-th row.

The following observation is the crucial insight for counting the iterations of Colour Refinement on G_k. We will use it to show that, informally stated, the subgraph $G_k[V]$ delays the propagation of the splitting of the colour classes in the remainder of the graph by k iterations whenever the splitting of a colour class contained in V_1 or V_{k+2} initiates a splitting of a colour class contained in V.

▷ Claim 19. Consider a colouring λ of G_k and its induced partition π_k of $V(G_k)$. For $t \in \mathbb{N}_0$, let π_k^t be the partition induced by $\chi_{G_k}^t$ on input (G_k, λ). Suppose G_k, λ, π_k satisfy the following conditions.

1. There exist $\ell \in [6]$ and $I_1, \ldots, I_\ell \subseteq [6]$ such that $\bigcup_{i \in [\ell]} I_i = [6]$ and $I_i \cap I_j = \emptyset$ for $1 \leq i < j \leq \ell$ and for every $i \in [\ell]$, it holds that $\{w_{k+2,j'} \mid j' \in I_i\} \in \pi_k^0$. That is, V_{k+2} is a union of colour classes with respect to λ.
2. $\pi_k^1 = \big\{\{w_{k+1,j'} \mid j' \in I_i\} \,\big|\, i \in [\ell]\big\} \cup \big\{C \setminus V_{k+1} \,\big|\, C \in \pi_k, C \setminus V_{k+1} \neq \emptyset\big\}$.
3. For all $C, C' \subseteq V_{k+1}$ with $C, C' \in \pi_k^1$, the graph $G_k[C]$ is regular and $G_k[C, C']$ is biregular.

Then for every $t \in [k]$, it holds that

$$\pi_k^t = \bigcup_{i' \in [t]} \big\{\{w_{k+2-i',j'} \mid j' \in I_i\} \,\big|\, i \in [\ell]\big\} \cup \Big\{C \setminus \Big(\bigcup_{i' \in [t]} V_{k+2-i'}\Big) \,\Big|\, C \in \pi_k^0, C \setminus \Big(\bigcup_{i' \in [t]} V_{k+2-i'}\Big) \neq \emptyset\Big\}.$$

The proof of the claim follows by induction on t. For $t = 1$, the statement is exactly the second item from the assumptions. The induction step is quite technical and we defer the reader to the full version for the details [21]. Informally speaking, the idea is to show that, within one iteration of Colour Refinement, if there is a a column V_i with $i \in [2, k+1]$ whose partition into row indices is strictly coarser than the partition of the column V_{i+1}, then V_i copies the partition from V_{i+1} and all other colour classes remain intact.

We call the property described in the claim *path propagation from right to left*. Modifying the indices, a similar statement yields *path propagation from left to right*, as formulated in the following claim.

▷ Claim 20. Consider a colouring λ of G_k and its induced partition π_k of $V(G_k)$. For $t \in \mathbb{N}_0$, let π_k^t be the partition induced by $\chi_{G_k}^t$ on input (G_k, λ). Suppose G_k, λ, π_k satisfy the following conditions.

1. There exist $\ell \in [6]$ and $I_1, \ldots, I_\ell \subseteq [6]$ such that $\bigcup_{i \in [\ell]} I_i = [6]$ and $I_i \cap I_j = \emptyset$ for $1 \leq i < j \leq \ell$ and for every $i \in [\ell]$, it holds that $\{w_{1,j'} \mid j' \in I_i\} \in \pi_k$. That is, the first column is a union of colour classes with respect to λ.

2. $\pi_k^1 = \big\{\{w_{2,j'} \mid j' \in I_i\} \,\big|\, i \in [\ell]\big\} \cup \big\{C \setminus V_2 \,\big|\, C \in \pi_k, C \setminus V_2 \neq \emptyset\big\}$.
3. For all $C, C' \subseteq V_2$ with $C, C' \in \pi_k^1$, the graph $G_k[C]$ is regular and $G_k[C, C']$ is biregular.

Then for every $t \in [k]$, it holds that

$$\pi_k^t = \bigcup_{i' \in [t]} \big\{\{w_{i'+1,j'} \mid j' \in I_i\} \,\big|\, i \in [\ell]\big\} \cup \Big\{C \setminus \Big(\bigcup_{i' \in [t]} V_{i'+1}\Big) \,\Big|\, C \in \pi_k^0, C \setminus \Big(\bigcup_{i' \in [t]} V_{i'+1}\Big) \neq \emptyset\Big\}.$$

Again, we defer the reader to the full version for the proof details for the path propagation. We are now ready to analyse the run of Colour Refinement on input G_k. Recall that π_0^t denotes the partition induced by $\chi_{G_0}^t$ on $V(G_0) \subseteq V(G_k)$. For the following arguments, see also Figure 1.

In π_k^1, the vertices are distinguished according to their degrees. We can then use path propagation from right to left to deduce that

$$\pi_k^{k+1} = \pi_0^1 \cup \big\{V_i \,\big|\, i \in [2, k+1]\big\} \quad \text{and thus} \quad \pi_k^{k+2} = \pi_0^2 \cup \big\{V_i \,\big|\, i \in [2, k+1]\big\},$$
$$\pi_k^{k+3} = \pi_0^3 \cup \big\{V_i \,\big|\, i \in [2, k+1]\big\} \quad \text{and also} \quad \pi_k^{k+4} = \pi_0^4 \cup \big\{V_i \,\big|\, i \in [2, k+1]\big\}.$$

Now path propagation from left to right yields that

$$\pi_k^{2k+4} = \pi_0^4 \cup \big\{\{w_{i,j} \mid j \in [4]\} \,\big|\, i \in [2, k+1]\big\} \cup \big\{\{w_{i,j} \mid j \in \{5,6\}\} \,\big|\, i \in [2, k+1]\big\}$$

and $\pi_k^{2k+5} = \pi_0^5 \cup (\pi_k^{2k+4} \setminus \pi_0^4)$. With three more combinations of path propagation from right to left and path propagation from left to right, we obtain

$$\pi_k^{6k+13} = \pi_0^{13} \cup \big\{\{w_{i,j}\} \,\big|\, i \in [2, k+1], j \in [6]\big\},$$

which is the discrete partition by the induction assumption for $k = 0$.

This implies that on input G_k, Colour Refinement takes $6k + 13$ iterations to stabilise and, since $|G_k| = 6k + 14$, it holds that $\mathrm{WL}_1(G_k) = n - 1$, where $n = |G_k|$. ◀

▶ **Corollary 21.** *There are infinitely many $n \in \mathbb{N}$ with $\mathrm{WL}_1(n) = n - 1$.*

▶ **Corollary 22.** *For every even $n \in \mathbb{N}_{\geq 12}$ such that $n = 12$ or $n \bmod 18 \notin \{6, 12\}$, there is a long-refinement graph G with $|G| = n$. The graph G can be chosen to satisfy $\deg(G) = \{2, 3\}$.*

Proof. The string representation S011XX covers $n = 12$. The first infinite family from Theorem 18 covers all even $n \in \mathbb{N}_{\geq 14}$ with $n = 2 \mod 6$, i.e. with $n \bmod 18 \in \{2, 8, 14\}$. The second and the third infinite family both cover all even $n \in \mathbb{N}_{\geq 16}$ with $n = 4 \mod 6$, i.e. with $n \bmod 18 \in \{4, 10, 16\}$. The fourth and the fifth infinite family cover all even $n \in \mathbb{N}_{\geq 18}$ with $n = 0 \mod 18$. Thus, among the even graph orders larger than 12, only the ones with $n \bmod 18 \in \{6, 12\}$ remain not covered. ◀

We now turn to the long-refinement graphs G of odd order with vertex degrees in $\{2, 3\}$. If the graph has odd order, we cannot represent it just with pairs. For this, we relax Assumption 14 as follows.

▶ **Assumption 23.** *G is a long-refinement graph with $\deg(G) = \{2, 3\}$ and such that there is an $i \in \mathbb{N}_0$ for which π^i contains only pairs and at most one singleton.*

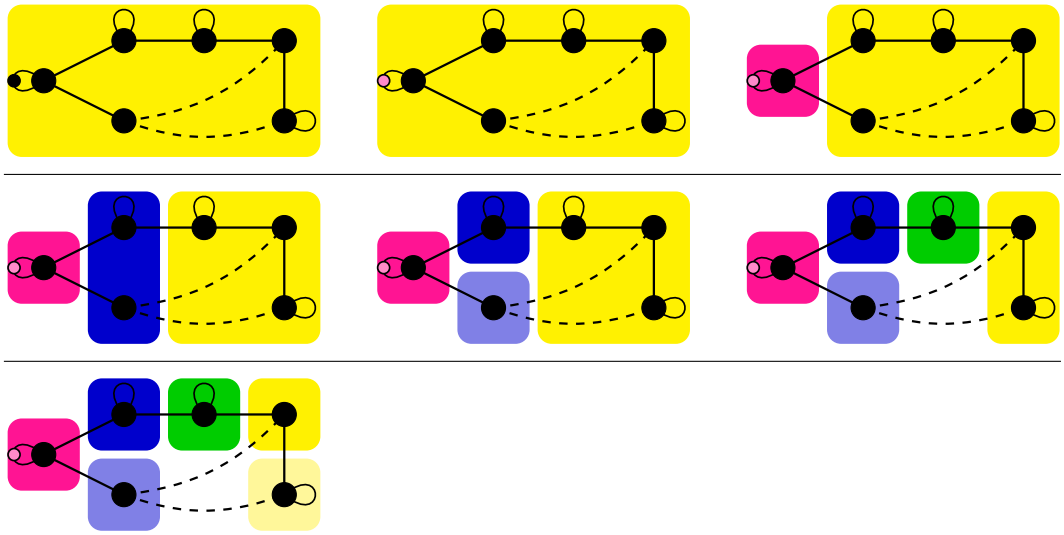

Figure 3 A visualisation of the graph with string representation SÎ11XX and the evolution of the colour classes in the first 6 Colour Refinement iterations on the graph.

We maintain the vocabulary and notation from the long-refinement graphs of even order, i.e. 0, 1, S, X will be used in the same way as before. However, in order to fully describe the odd-order graphs via strings, we have to extend the string alphabet by fresh letters $\hat{1}$ and $\hat{\mathrm{X}}$, which represent particular pairs with attached vertices as follows. For a string $\hat{\Xi}\colon [\ell] \to \{0, 1, \mathrm{S}, \mathrm{X}, \hat{1}, \hat{\mathrm{X}}\}$, we define the *base string* Ξ as the string obtained by removing hats. More precisely, we replace every $\hat{\mathrm{X}}$ with an X, and we replace every $\hat{1}$ with a 1 if it is non-terminal (i.e. if it is not at position ℓ) and with a 0 otherwise. Let $I(\hat{\Xi}) \subseteq [\ell]$ be the set of positions i with $\hat{\Xi}(i) \in \{\hat{1}, \hat{X}\}$. If, in the base graph $G(\Xi)$, every vertex pair corresponding to a position in $I(\hat{\Xi})$ (a *hat vertex pair*) is adjacent, we call $\hat{\Xi}$ a *hat string*.

Similarly as for the even-order long-refinement graphs, to every hat string $\hat{\Xi}$, we assign a graph $G(\hat{\Xi})$. We obtain the graph $G(\hat{\Xi})$ by subdividing in $G(\Xi)$ each edge connecting a hat vertex pair with a new fresh vertex, which we call a *hat*. Additionally, if the vertices of the hat vertex pair have degree 2, we now insert another edge between them. (This can only happen if the hat vertex pair corresponds to a terminal 0.) For a hat $\hat{v}$, we call the neighbourhood $N(\hat{v}) \subseteq V\big(G(\hat{\Xi})\big)$ the *hat base* of $\hat{v}$. Note that every vertex in the hat base has degree 3 by construction. Also, a hat always has degree 2 and thus, with respect to $\chi^1_{G(\hat{\Xi})}$, it has a different colour than its hat base.

Graphically, we represent a hat by a loop attached to the corresponding hat vertex pair, which we subdivide by inserting a small vertex (see Figure 3). It is not difficult to see that every graph $G(\hat{\Xi})$ corresponding to a hat string $\hat{\Xi}$ has exactly one hat and that the hat is the first vertex forming a singleton colour class during the execution of Colour Refinement on $G(\hat{\Xi})$. Thus, $|G(\hat{\Xi})| = |G(\Xi)| + 1$.

▶ **Theorem 24.** *For every string $\hat{\Xi}$ contained in the following sets, the graph $G(\hat{\Xi})$ is a long-refinement graph.*

- $\{\mathrm{S}\hat{1}11\mathrm{XX}\}$
- $\{\mathrm{S0X1}\hat{\mathrm{X}}\} \cup \{\mathrm{S}1^k1011^k\mathrm{X1X}1^k\hat{1} \mid k \in \mathbb{N}_0\}$
- $\{\mathrm{S110X}\hat{\mathrm{X}}\} \cup \{\mathrm{S}111^k1011^k\mathrm{XX}1^k\hat{1} \mid k \in \mathbb{N}_0\}$
- $\{\mathrm{S}1^k01^k1\mathrm{XX}1^k\hat{1} \mid k \in \mathbb{N}_0\}$
- $\{\mathrm{S}(011)^k00(110)^k\mathrm{X}\hat{1}\mathrm{X}(011)^k0 \mid k \in \mathbb{N}_0\}$

Proof sketch. The proof techniques for the infinite families are very similar to the ones presented for Theorem 18. Therefore, we only sketch the proof on two concrete examples containing $\hat{1}$ and $\hat{\mathrm{X}}$, respectively. To be able to refer to vertices explicitly, recall the formal definition of $G := G(\Xi)$ from Section 4. Since for a hat string $\hat{\Xi}$, it holds that $|G(\hat{\Xi})| = |G(\Xi)| + 1$, we can use the same indexing of vertices, additionally letting $\hat{v}$ be the unique hat in $G(\hat{\Xi})$.

In the graph $G := G(\mathrm{S\hat{1}11XX})$ (cf. Figure 3), the only vertex of degree 2 is $\hat{v}$. Thus, in π^1, it forms a singleton colour class. In π^2, the hat base $\{v_{2,1}, v_{2,2}\}$ forms a new colour class.

In general, for $i \in \mathbb{N}$, we have that $\pi^i_G = \pi^{i-1}_{G'} \cup \{\hat{v}\}$, where $G' = G(\mathrm{S011XX})$ (cf. Theorem 18 and Figure 2).

Thus,

$$\pi^1 = \big\{\{\hat{v}\}, V(G) \setminus \{\hat{v}\}\big\},$$
$$\pi^2 = \big\{\{\hat{v}\}, \{v_{2,1}, v_{2,2}\}, V(G) \setminus \{\hat{v}, v_{2,1}, v_{2,2}\}\big\},$$
$$\pi^3 = \big\{\{\hat{v}\}, \{v_{2,1}, v_{2,2}\}, \{v_{i,j} \mid i \in \{1,3\}, j \in [2]\}, \{v_{i,j} \mid i \in [4,6], j \in [2]\}\big\},$$
$$\pi^4 = \big\{\{\hat{v}\}, \{v_{1,1}, v_{1,2}\}, \{v_{2,1}, v_{2,2}\}, \{v_{3,1}, v_{3,2}\}, \{v_{i,j} \mid i \in [4,6], j \in [2]\}\big\},$$
$$\pi^5 = \big\{\{\hat{v}\}, \{v_{1,1}, v_{1,2}\}, \{v_{2,1}, v_{2,2}\}, \{v_{3,1}, v_{3,2}\}, \{v_{4,1}, v_{4,2}\}, \{v_{i,j} \mid i \in [5,6], j \in [2]\}\big\},$$
$$\pi^6 = \big\{\{\hat{v}\}\big\} \cup \big\{\{v_{i,j} \mid j \in [2]\} \mid i \in [6]\big\}.$$

Now in 6 further iterations, the splitting of the pairs is propagated linearly according to the order $\prec$.

Next, let G be the graph $G(\mathrm{S0X1\hat{X}})$, i.e. a member of the first infinite family from Theorem 24. It has three vertices of degree 2, namely $\hat{v}$, $v_{2,1}$, and $v_{2,2}$, which therefore form a colour class in π^1. Also, the vertices contained in the 1-pair are the only vertices of degree 3 that are not adjacent to any vertex of degree 2. Thus,

$$\pi^2 = \big\{\{\hat{v}, v_{2,1}, v_{2,2}\}, \{v_{4,1}, v_{4,2}\}, V(G) \setminus \{\hat{v}, v_{2,1}, v_{2,2}, v_{4,1}, v_{4,2}\}\big\},$$

and similarly,

$$\pi^3 = \big\{\{\hat{v}, v_{2,1}, v_{2,2}\}, \{v_{4,1}, v_{4,2}\}, \{v_{1,1}, v_{1,2}\}, \{v_{i,j} \mid i \in \{3,5\}, j \in [2]\}\big\}.$$

Now the hat forms a singleton since, in contrast to the vertices of the 0-pair, it is not adjacent to any vertex in the S-pair. We obtain:

$$\pi^4 = \big\{\{\hat{v}\}, \{v_{2,1}, v_{2,2}\}, \{v_{4,1}, v_{4,2}\}, \{v_{1,1}, v_{1,2}\}, \{v_{i,j} \mid i \in \{3,5\}, j \in [2]\}\big\},$$
$$\pi^5 = \big\{\{\hat{v}\}\big\} \cup \big\{\{v_{i,j} \mid j \in [2]\} \mid i \in [5]\big\}.$$

Then in 5 further iterations, the splitting of the pairs is propagated linearly according to the order $\prec$. ◀

As an example, Figure 3 shows the evolution of the colour classes of the graph with string representation S$\hat{1}$11XX.

▶ **Corollary 25.** *For every odd $n \in \mathbb{N}_{\geq 11}$ with $n \bmod 18 \notin \{3, 9\}$, there is a long-refinement graph G with $|G| = n$. The graph G can be chosen to satisfy $\deg(G) = \{2, 3\}$.*

Proof. The string representation S$\hat{1}$11XX covers $n = 13$. The first infinite family covers all odd $n \in \mathbb{N}_{\geq 11}$ with $n = 5 \mod 6$, i.e. with $n \bmod 18 \in \{5, 11, 17\}$. The second and the third infinite family both cover all all odd $n \in \mathbb{N}_{\geq 13}$ with $n = 1 \mod 6$, i.e. with $n \bmod 18 \in \{1, 7, 13\}$. The fourth infinite family covers all odd $n \in \mathbb{N}_{\geq 15}$ with $n = 15 \mod 18$. Thus, among the odd orders larger than 10, only the ones with $n \bmod 18 \in \{3, 9\}$ are skipped. ◀

We summarise the results from Corollaries 22 and 25.

▶ **Corollary 26.** *For every $n \in \mathbb{N}_{\geq 11}$ such that $n = 12$ or $n \bmod 18 \notin \{3, 6, 9, 12\}$, there is a long-refinement graph G with $|G| = n$. The graph G can be chosen to satisfy* $\deg(G) = \{2, 3\}$.

The following lemma allows to cover more graph sizes.

▶ **Lemma 27.** *Let $n \in \mathbb{N}$ be arbitrary. Suppose there is a long-refinement graph G such that $|G| = n$. If there is a $d \in \mathbb{N}$ with $\deg(G) = \{d, d+1\}$ such that $|\{v \in V(G) \mid \deg(v) = d\}| \neq d + 1$, then there is also a long-refinement graph G' with $|G'| = n + 1$.*

Proof. We can insert an isolated vertex w into G and insert edges from w to every vertex $v \in V(G)$ with $\deg(v) = d$. In the new graph G', the vertex w has a degree other than $d + 1$, whereas all other vertices have degree $d + 1$. Thus, the colour classes in $\pi^1_{G'}$ are $\{w\}$ and $V(G') \setminus \{w\}$. After the second iteration, the neighbours of w are distinguished from all other vertices, just like they are in π^1_G. Inductively, it is easy to see that for $i \in \mathbb{N}$, it holds that $\pi^i_{G'} = \pi^{i-1}_G \cup \{\{w\}\}$. Thus, Colour Refinement takes $n - 1 + 1 = n$ iterations to stabilise on G'. ◀

▶ **Corollary 28.** *For every odd $n \in \mathbb{N}_{\geq 11}$, there is a long-refinement graph G with $|G| = n$.*

Proof. By Corollary 25, it suffices to provide long-refinement graphs of order n for every odd $n \in \mathbb{N}_{\geq 11}$ with $n \bmod 18 \in \{3, 9\}$. We will accomplish this by applying Lemma 27 to suitable graphs of orders n' with $n' \bmod 18 \in \{2, 8\}$. Every graph G with a string representation contained in one of the infinite families from Theorem 18 has an even number of vertices of degree 2. In particular, it satisfies $|\{v \in V(G) \mid \deg(v) = 2\}| \neq 3$. Furthermore, every even graph order larger than 10 not covered by Corollary 22 is a multiple of 6. Hence, since $N \coloneqq \{n \in \mathbb{N}_{\geq 18} \mid n \bmod 18 \in \{2, 8\}\}$ contains only even numbers and no multiples of 6, for every $n' \in N$, there is a graph of order n' that satisfies the prerequisites of Lemma 27 with $d = 2$. (Actually, we can cover all of these graph orders with the family $\{\mathrm{S1}^k001^k\mathrm{X1X1}^k0 \mid k \in \mathbb{N}_0\}$.)

Thus, applying the lemma, we can construct for every $n \in \mathbb{N}_{\geq 18}$ with $n \bmod 18 \in \{3, 9\}$ a long-refinement graph G' of order n. ◀

Note that, since we apply Lemma 27 to close the gaps, we cannot guarantee anymore that the vertex degrees are 2 and 3, as we could in Corollary 26.

We are ready to prove Theorem 1.

Proof of Theorem 1. The theorem follows from combining Corollaries 22 and 28 with Theorem 7. ◀

Although the theorem leaves some gaps, we can show that, starting from $n = 10$, the worst-case number of Colour Refinement iterations until stabilisation on graphs of order n is always either $n - 2$ or $n - 1$.

Proof of Theorem 2. By Theorem 1, we only need to consider the numbers $n \geq 24$ for which $n \bmod 18 \in \{6, 12\}$. Since these numbers are all even, for every such n, we can use Corollary 28 to obtain a long-refinement graph G' with $|G'| = n - 1$. Let v be a fresh vertex not contained in $V(G')$. Now define $G \coloneqq (V, E)$ with $V \coloneqq V(G') \cup \{v\}$ and $E \coloneqq E(G')$. Then $\pi^i_G = \pi^i_{G'} \cup \{\{v\}\}$ for every $i \in \mathbb{N}$. In particular, $\mathrm{WL}_1(G) = \mathrm{WL}_1(G') = n - 2$. ◀

6 Conclusion

With Theorem 2, it holds for all $n \in \mathbb{N}_{\geq 10}$ that $\mathrm{WL}_1(n) \geq n-2$. In particular, this proves that the trivial upper bound $\mathrm{WL}_1(n) = n-1$ is tight, up to an additive constant of 1.

For infinitely many graph orders, the graph G can even be chosen to have vertex degrees 2 and 3, as Theorems 18 and 24 show. We applied Lemma 27 to cover some of the remaining sizes. However, no order $n \in \mathbb{N}_{\geq 18}$ with $n \bmod 18 \in \{6, 12\}$ is covered by Theorem 18. Also, for $|G| \in \{n \in \mathbb{N}_{\geq 18} \mid n \bmod 18 \in \{5, 11\}\}$, all the long-refinement graphs G we have found satisfy $|\{v \in V(G) \mid \deg(v) = 2\}| = 3$ (see the first infinite family in Theorem 24). Thus, these graphs do not satisfy the prerequisites of Lemma 27. Note that we cannot apply the construction from the proof if $|\{v \in V(G) \mid \deg(v) = d\}| = d+1$, since the new graph G' would be $(d+1)$-regular and would thus satisfy $\mathrm{WL}_1(G') = 0$. Hence, it is not clear how to apply our techniques to construct a long-refinement graph of order 24. Altogether, the values $n \in \mathbb{N}_{\geq 24}$ with $n \bmod 18 \in \{6, 12\}$ are precisely the graph orders for which it remains open whether there is a graph G with $\mathrm{WL}_1(G) = |G| - 1$.

A related question is for which values $d_1 \neq d_2$ there are long-refinement graphs G with $\deg(G) = \{d_1, d_2\}$. It would be nice to know whether we have actually found all long-refinement graphs G with $\deg(G) = \{2, 3\}$. Similarly, we can ask for long-refinement graphs when fixing other parameters. For example, since all long-refinement graphs that we found have girth at most 4, it would be interesting to know whether there exists any long-refinement graph with larger girth, or infinite families with unbounded girth.

Also, in the light of the graph isomorphism problem, it is a natural follow-up task to find for each order n pairs of non-isomorphic graphs G, H for which it takes $n-1$ Colour Refinement iterations to distinguish the graphs from each other. A first step towards this goal is the search for pairs of long-refinement graphs of equal order. It is easy to see that for infinitely many n, Theorems 18 and 24 yield such pairs of graphs. Still, for example, when evaluating the colourings computed by Colour Refinement on the graphs with string representations S1100XX0 and S001XX10, they differ after less than $n-1$ iterations. To see this, observe that in G(S1100XX0), all vertices of degree 2 have paths of length 3 to a vertex of degree 2 whose inner vertices only have degrees other than 2. This is not the case for G(S001XX10) and this property is detected by Colour Refinement after at most 4 iterations. Thus, finding for $n \in \mathbb{N}_{\geq 10}$ two graphs of order n which Colour Refinement only distinguishes after $n-1$ iterations remains a challenge.

References

1 Vikraman Arvind, Frank Fuhlbrück, Johannes Köbler, and Oleg Verbitsky. On Weisfeiler-Leman invariance: Subgraph counts and related graph properties. In *Fundamentals of Computation Theory – 22nd International Symposium, FCT 2019, Copenhagen, Denmark, August 12–14, 2019, Proceedings*, pages 111–125, 2019. doi:10.1007/978-3-030-25027-0_8.

2 Vikraman Arvind, Johannes Köbler, Gaurav Rattan, and Oleg Verbitsky. Graph isomorphism, color refinement, and compactness. *Computational Complexity*, 26(3):627–685, 2017. doi:10.1007/s00037-016-0147-6.

3 László Babai, Paul Erdős, and Stanley M. Selkow. Random graph isomorphism. *SIAM Journal on Computing*, 9(3):628–635, 1980. doi:10.1137/0209047.

4 Christoph Berkholz and Jakob Nordström. Near-optimal lower bounds on quantifier depth and Weisfeiler-Leman refinement steps. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 267–276, 2016. doi:10.1145/2933575.2934560.

5 Jin-Yi Cai, Martin Fürer, and Neil Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12(4):389–410, 1992.

6 Alain Cardon and Maxime Crochemore. Partitioning a graph in $O(|A| \log |A|)$. *Theoretical Computer Science*, 19:85–98, 1982. doi:10.1016/0304-3975(82)90016-0.

7 Paul T. Darga, Mark H. Liffiton, Karem A. Sakallah, and Igor L. Markov. Exploiting structure in symmetry detection for CNF. In *Proceedings of the 41th Design Automation Conference, DAC 2004, San Diego, CA, USA, June 7-11, 2004*, pages 530–534. ACM, 2004. doi:10.1145/996566.996712.

8 Frank Fuhlbrück, Johannes Köbler, and Oleg Verbitsky. Local WL invariance and hidden shades of regularity. *CoRR*, abs/2002.04590, 2020. arXiv:2002.04590.

9 Martin Fürer. Weisfeiler-Lehman refinement requires at least a linear number of iterations. In *Automata, Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001, Proceedings*, volume 2076 of *Lecture Notes in Computer Science*, pages 322–333. Springer, 2001. doi:10.1007/3-540-48224-5_27.

10 Martin Fürer. On the combinatorial power of the Weisfeiler-Lehman algorithm. In Dimitris Fotakis, Aris Pagourtzis, and Vangelis Th. Paschos, editors, *Proceedings of the Tenth International Conference on Algorithms and Complexity*, volume 10236 of *Lecture Notes in Computer Science*, pages 260–271. Springer Verlag, 2017. doi:10.1007/978-3-319-57586-5_22.

11 Maximilian Gödicke. The iteration number of the Weisfeiler-Lehman-algorithm. Master's thesis, RWTH Aachen University, 2015.

12 Christopher D. Godsil. Compact graphs and equitable partitions. *Linear Algebra Appl.*, 255:259–266, 1997. doi:10.1016/S0024-3795(97)83595-1.

13 Martin Grohe. Fixed-point definability and polynomial time on graphs with excluded minors. *J. ACM*, 59(5):27, 2012. doi:10.1145/2371656.2371662.

14 Martin Grohe, Kristian Kersting, Martin Mladenov, and Erkal Selman. Dimension reduction via colour refinement. In *Algorithms - ESA 2014 - 22th Annual European Symposium, Wroclaw, Poland, September 8-10, 2014. Proceedings*, volume 8737 of *Lecture Notes in Computer Science*, pages 505–516. Springer, 2014. doi:10.1007/978-3-662-44777-2_42.

15 Martin Grohe and Sandra Kiefer. A linear upper bound on the Weisfeiler-Leman dimension of graphs of bounded genus. In *Proceedings of the Forty-Sixth International Colloquium on Automata, Languages, and Programming*, pages 117:1–117:15, July 2019. doi:10.4230/LIPIcs.ICALP.2019.117.

16 Martin Grohe and Daniel Neuen. Canonisation and definability for graphs of bounded rank width. In *Proceedings of the Thirty-Fourth Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 1–13, June 2019. doi:10.1109/LICS.2019.8785682.

17 Martin Grohe and Oleg Verbitsky. Testing graph isomorphism in parallel by playing a game. In *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10–14, 2006, Proceedings, Part I*, pages 3–14, 2006. doi:10.1007/11786986_2.

18 Neil Immerman and Eric Lander. Describing graphs: A first-order approach to graph canonization. In Alan L. Selman, editor, *Complexity theory retrospective*, pages 59–81. Springer, 1990. doi:10.1007/978-1-4612-4478-3_5.

19 Tommi A. Junttila and Petteri Kaski. Engineering an efficient canonical labeling tool for large and sparse graphs. In *Proceedings of the Nine Workshop on Algorithm Engineering and Experiments, ALENEX 2007, New Orleans, Louisiana, USA, January 6, 2007*. SIAM, 2007. doi:10.1137/1.9781611972870.13.

20 Sandra Kiefer. *Power and Limits of the Weisfeiler-Leman Algorithm*. PhD thesis, RWTH Aachen University, Aachen, 2020. doi:10.18154/RWTH-2020-03508.

21 Sandra Kiefer and Brendan D. McKay. The iteration number of Colour Refinement. *CoRR*, 2020. arXiv:2005.10182.

22 Sandra Kiefer and Pascal Schweitzer. Upper bounds on the quantifier depth for graph differentiation in first order logic. In *Proceedings of the Thirty-First Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 287–296, July 2016. doi:10.1145/2933575.2933595.

23 Sandra Kiefer and Pascal Schweitzer. Upper bounds on the quantifier depth for graph differentiation in first-order logic. *Logical Methods in Computer Science*, 15(2), 2019. URL: https://lmcs.episciences.org/5522, doi:10.23638/LMCS-15(2:19)2019.

24 Sandra Kiefer, Pascal Schweitzer, and Erkal Selman. Graphs identified by logics with counting. In *Proceedings of the Fortieth International Symposium on Mathematical Foundations of Computer Science*, volume 9234 of *Lecture Notes in Computer Science*, pages 319–330. Springer, August 2015. doi:10.1007/978-3-662-48057-1_25.

25 Johannes Köbler and Oleg Verbitsky. From invariants to canonization in parallel. In *Computer Science - Theory and Applications, Third International Computer Science Symposium in Russia, CSR 2008, Moscow, Russia, June 7-12, 2008, Proceedings*, volume 5010 of *Lecture Notes in Computer Science*, pages 216–227. Springer, 2008. doi:10.1007/978-3-540-79709-8_23.

26 Andreas Krebs and Oleg Verbitsky. Universal covers, color refinement, and two-variable counting logic: Lower bounds for the depth. In *Proceedings of the Thirtieth Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 689–700, 2015. doi:10.1109/LICS.2015.69.

27 Wenchao Li, Hossein Saidi, Huascar Sanchez, Martin Schäf, and Pascal Schweitzer. Detecting similar programs via the Weisfeiler-Leman graph kernel. In *Software Reuse: Bridging with Social-Awareness - 15th International Conference, ICSR 2016, Limassol, Cyprus, June 5-7, 2016, Proceedings*, pages 315–330, 2016. doi:10.1007/978-3-319-35122-3_21.

28 Moritz Lichter, Ilia N. Ponomarenko, and Pascal Schweitzer. Walk refinement, walk logic, and the iteration number of the Weisfeiler-Leman algorithm. In *Proceedings of the Thirty-Fourth Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 1–13, June 2019. doi:10.1109/LICS.2019.8785694.

29 Brendan D. McKay. Practical graph isomorphism. *Congressus Numerantium*, 30:45–87, 1981.

30 Brendan D. McKay and Adolfo Piperno. Practical graph isomorphism, II. *Journal of Symbolic Computation*, 60:94–112, 2014. doi:10.1016/j.jsc.2013.09.003.

31 Harry L. Morgan. The generation of a unique machine description for chemical structures – a technique developed at chemical abstracts service. *Journal of Chemical Documentation*, 5(2):107–113, 1965. doi:10.1021/c160017a018.

32 Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan E. Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and Leman go neural: Higher-order graph neural networks. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, January 2019. doi:10.1609/aaai.v33i01.33014602.

33 Motakuri V. Ramana, Edward R. Scheinerman, and Daniel Ullman. Fractional isomorphism of graphs. *Discrete Mathematics*, 132(1–3):247–265, 1994. doi:10.1016/0012-365X(94)90241-0.

34 Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. Weisfeiler-Lehman graph kernels. *Journal of Machine Learning Research*, 12:2539–2561, 2011. URL: http://dl.acm.org/citation.cfm?id=2078187.

35 Gottfried Tinhofer. A note on compact graphs. *Discrete Applied Mathematics*, 30(2–3):253–264, 1991. doi:10.1016/0166-218X(91)90049-3.

36 Oleg Verbitsky. Planar graphs: Logical complexity and parallel isomorphism tests. In *STACS 2007, 24th Annual Symposium on Theoretical Aspects of Computer Science, Aachen, Germany, February 22–24, 2007, Proceedings*, pages 682–693, 2007. doi:10.1007/978-3-540-70918-3_58.