The Isomorphism Problem of Power Graphs and a Question of Cameron

Bireswar Das
Indian Institute of Technology Gandhinagar, India

Jinia Ghosh
Indian Institute of Technology Gandhinagar, India

Anant Kumar
Indian Institute of Technology Gandhinagar, India

Abstract

We study the isomorphism problem of graphs that are defined in terms of groups, namely power graphs, directed power graphs, and enhanced power graphs. We design polynomial-time algorithms for the isomorphism problems for the power graphs, the directed power graphs and the enhanced power graphs arising from finite nilpotent groups. In contrast, no polynomial-time algorithm is known for the group isomorphism problem, even for nilpotent groups of class 2.

Our algorithms do not require the underlying groups of the input graphs to be given. A crucial step in our algorithms is to reconstruct the directed power graph from the given power graph or the enhanced power graph. The problem of efficiently computing the directed power graph from a power graph or an enhanced power graph is due to Cameron [IJGT'22]. Bubboloni and Pinzauti [Arxiv'22] gave a polynomial-time algorithm to reconstruct the directed power graph from a power graph. We give an efficient algorithm to compute the directed power graph from an enhanced power graph. The tools and techniques that we design are general enough to give a different algorithm to compute the directed power graph from a power graph as well.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Graph algorithms analysis

Keywords and phrases Graph Isomorphism, Graphs defined on Groups, Power Graphs, Enhanced Power Graphs, Directed Power Graphs, Nilpotent Groups

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2024.20

Related Version *Full Version*: `https://arxiv.org/pdf/2305.18936` [12]

1 Introduction

Given two graphs as input, the graph isomorphism problem (GI) is to check if the graphs are isomorphic. Despite extensive research, the complexity status of GI is still open. The graph isomorphism problem is in NP but is very unlikely to be NP-hard as it is also in coAM [6].

Efficient algorithms for the graph isomorphism problem are known for several restricted graph classes, for example, graphs with bounded genus [28, 16], graphs with bounded degree [26], graphs with bounded eigenvalue multiplicity [3], graphs with bounded tree-width [5], graphs with bounded rank-width [20, 17].

Group-theoretic tools have played an important role in the design of efficient algorithms for the GI. Some of the early works on GI using the structure of groups non-trivially include the isomorphism algorithm for bounded degree graphs by Luks [26], and the graph canonization framework developed by Babai and Luks [4]. Babai developed sophisticated new techniques to give a quasipolynomial time isomorphism algorithm [2]. Currently, it is the best known isomorphism algorithm for general graphs. Group-theoretic machinery has been used to design faster isomorphism algorithms for bounded degree graphs by Grohe,

Neuen and Schweitzer [18]; for graphs with bounded tree-width by Grohe, Neuen, Schweitzer, Wiebking [19] and Wiebking [36]; for bounded rank-width graphs by Grohe and Schweitzer [20] and graphs excluding small topological subgraphs [32].

In this paper, we study the isomorphism problem of graphs defined on finite groups. More precisely, we study the class of power graphs, directed power graphs, and enhanced power graphs. For two elements x and y in a finite group G, we say that y is a power of x if $y = x^i$ for some integer i. For a group G, the vertex set of the *power graph* $Pow(G)$ of G consists of elements of G. Two vertices x and y are adjacent in $Pow(G)$ if x is a power of y or y is a power of x. We refer to G as the *underlying group* of $Pow(G)$. The definition of directed power graphs and enhanced power graphs can be found in Section 2.

Kelarev and Quinn defined the concept of directed power graphs of semigroups [24]. Power graphs were defined by Chakrabarty et al. [11] again for semigroups. Cameron in [9] discusses several graph classes defined in terms of groups and surveys many interesting results on these graphs. Kumar et al. [25] gave a survey on the power graphs of finite groups. Questions related to isomorphism of graphs defined on groups have also been studied [34, 13, 30, 14].

Our motivation for studying the isomorphism of graphs defined in terms of groups is to explore if the group structure can be exploited to give efficient algorithms for the isomorphism problems of these graphs. There are two versions of the isomorphism problem for each class of graphs defined on groups. Let us consider the case for the class of power graphs. In the first version of the problem, two groups G_1 and G_2 are given by their Cayley tables, and the task is to check if $Pow(G_1)$ is isomorphic to $Pow(G_2)$. In the second version, two power graphs Γ_1 and Γ_2 are given and we need to check if Γ_1 is isomorphic to Γ_2. Note that in the second version, the underlying groups are not given as input.

In the first version of the problem, it is tempting to use the isomorphism of the underlying groups in the hope that it might yield an easier [1] quasipolynomial time algorithm because, unlike graphs, the quasipolynomial time algorithm for groups attributed to Tarjan by Miller [29] is technically and conceptually much easier. However, we note that it is not enough to check the isomorphism of the underlying groups, as two nonisomorphic groups can have isomorphic power graphs[2].

The second version looks more challenging since we do not have the underlying groups. As one of our *main results*, we show that the isomorphism problem of power graphs of nilpotent groups can be tested in polynomial-time even in the second version of the isomorphism problem (Section 6). Thus, we do not need the underlying groups to be given. In contrast, the group isomorphism problem for nilpotent groups, even for class 2, is still unresolved and is considered to be a bottleneck for the group isomorphism problem [15].

Our algorithm for solving the isomorphism problem of power graphs works by first computing the directed power graphs of the input power graphs. Next, we use the algorithm for the isomorphism problem of directed power graphs for nilpotent groups that we design in Section 6.

The question of efficiently computing the directed power graph from the power graph (or the enhanced power graph) was asked by Cameron [9]: "Question 2: Is there a simple algorithm for constructing the directed power graph or the enhanced power graph from the power graph, or the directed power graph from the enhanced power graph?" Recently,

[1] compared to Babai's quasipolynomial time isomorphism algorithm.

[2] To see this, we can take the elementary abelian group of order 27 and the non-abelian group of order 27 with exponent 3 ([10]). In general, consider the power graphs of two nonisomorphic groups of order p^i for any $i \geq 2$ and exponent p for some prime p. One can check that the power graphs are isomorphic while the groups are not.

Bubboloni and Pinzauti [7] gave a polynomial-time algorithm to reconstruct the directed power graph from the power graph. We give a different solution to the problem of efficiently computing the directed power graph from the power graph. Moreover, we also design an efficient algorithm to compute the directed power graph from the enhanced power graph. This fully resolves Cameron's question.

Cameron [8], and Zahirović et al. [37] proved that for two finite groups, the power graphs are isomorphic if and only if the directed power graphs of the groups are isomorphic if and only if the enhanced power graphs of the groups are isomorphic. The algorithms to solve Cameron's question provide a complete algorithmic proof of this result.

This paper makes contributions in three algebraic and combinatorial techniques, which form the foundation of our algorithms. Firstly, we introduce a simple yet effective concept of a certain type of generating sets of a group which we call covering cycle generating sets (CCG-sets). In essence, these are defined in terms of the set of maximal cyclic subgroups (Section 3). Secondly, we present a set of results concerning the structure of closed-twins within specific subgraphs of power graphs. While Cameron previously explored the structure of closed-twins in a power graph [8], we extend this investigation to focus on the subgraph induced by the closed neighbourhood of a vertex (Section 4). These structures form the basis of our algorithm to determine whether a vertex in a power graph can be part of a CCG-set. Lastly, we introduce a series of reduction rules that facilitate the simplification of the structure of a directed power graph while preserving its isomorphism-invariant properties (Section 6).

Related work on Cameron's question. The algorithm to reconstruct the directed power graph from a power graph by Bubboloni and Pinzauti works by first considering the notion of *plain* and *compound* closed-twin classes in a power graph[3]. The other notion is that of *critical* closed-twin classes. Depending on whether a closed-twin class is critical or non-critical they design efficient tests to determine if the class is plain or compound. Once this is done, they put directions to the edges of the power graph to reconstruct the directed power graph. The details can be found in [7]. In contrast, our algorithm identifies a CCG-set in the power graph by using the properties and algorithms associated with the graph reductions that we give in this paper.

2 Preliminaries

For a simple graph X, the vertex set of X is denoted by $V(X)$, and the edge set of X is denoted by $E(X)$. For basic definitions and notations from graph theory, an interested reader can refer to any standard textbook (for example, [35]). A *subgraph* of X is a graph Y, where $V(Y) \subseteq V(X)$ and $E(Y) \subseteq E(X)$. The subgraph with the vertex set $S \subseteq V(X)$, and all such edges in $E(X)$ whose both endpoints are in S, is called the *induced subgraph* of X on S, and it is denoted by $X[S]$.

The set of vertices adjacent to a vertex u in an undirected graph X is called the open neighborhood of u in X and is denoted by $N_X(u)$. The cardinality of $N_X(u)$ is called the *degree* of u in X, denoted by $deg_X(u)$. The *closed neighborhood* of a vertex u in X is denoted by $N_X[u]$ and defined by $N_X[u] = N_X(u) \cup \{u\}$. Two vertices in X are called *closed-twins*[4] in X if their closed neighborhoods in X are the same.

[3] These notions are similar to the closed-twin classes that Cameron calls type (a) and type (b) (Proposition 5, [8]).

[4] In the previous related literature, both the terms *'false twins'* and *'closed twins'* are used. However, in this paper, we follow the terminology used by Cameron and other authors in their works on graphs defined on groups.

For a directed graph X (with no multiple edges), the *out-neighborhood* of a vertex u in X is the set $\{v \in V(X) \ : \ (u,v) \ \in E(X)\}$ and *out-degree* of u in X, denoted by $\textit{out-deg}_X(u)$, is the size of the out-neighborhood of u in X. Similarly, the *in-neighborhood* of a vertex u in X is the set $\{v \in V(X) \ : \ (v,u) \ \in E(X)\}$ and *in-degree* of u in X, denoted by $\textit{in-deg}_X(u)$, is the size of the in-neighborhood of u in X.[5] Two vertices in a directed graph X are called the *closed-twins* in X if their closed-out-neighborhoods in X are the same and also the closed-in-neighborhoods in X are the same. An edge of the form (u,u) in a directed graph is called a *self-loop*.

In any graph X, the *closed-twin-class* of a vertex u in X is the set of all closed-twins of u in X. In this paper, if the underlying graphs are colored, then by isomorphism we mean color preserving isomorphism only.

A graph is called *prime* with respect to strong product if it cannot be represented as a strong product of two non-trivial graphs.

▶ **Definition 1.** *Two graphs X and Y are called* isomorphic *if and only if there exists a bijection f from $V(X)$ to $V(Y)$ such that $\{u,v\} \in E(X)$ if and only if $\{f(u), f(v)\} \in E(Y)$. Moreover, if X and Y are vertex-colored, then an isomorphism f is called a* color preserving isomorphism *if for all $u \in V(X)$, the color of u and the color of $f(u)$ are the same.*

▶ **Definition 2** (see for example [22])**.** *Let X and Y be two directed graphs. The* strong product *$(X \boxtimes Y)$ of X and Y is the graph with the vertex set $V(X) \times V(Y)$, where there is an edge from (u,u') to a distinct vertex (v,v') in $X \boxtimes Y$ if and only if one of the following holds: (i) $u = v$ and there is an edge from u' to v' in Y. (ii) $u' = v'$ and there is an edge from u is to v in X. (iii) There is an edge from u to v in X and an edge from u' to v' in Y.*

▶ **Definition 3** (see e.g., [23])**.** *Vertex identification of a pair of vertices v_1 and v_2 of a graph is the operation that produces a graph in which the two vertices v_1 and v_2 are replaced with a new vertex v such that v is adjacent to the union of the vertices to which v_1 and v_2 were originally adjacent. In vertex identification, it doesn't matter whether v_1 and v_2 are connected by an edge or not.*

All the groups considered in this paper are finite. The basic definitions and properties from group theory can be found in any standard book (see, for example, [33]). A subset H of a group G is called a *subgroup* of G if H forms a group under the binary operation of G; it is denoted by $H \leq G$. The number of elements in G is called the *order* of the group and it is denoted by $|G|$. The *order* of an element g in G, denoted by $o(g)$, is the smallest positive integer m such that $g^m = e$, where e is the identity element. The set $\{g, g^2, g^3, \ldots, g^{m-1}, e\}$ is the set of all group elements that are *generated* by g, where $m = o(g)$. Moreover, this set forms a subgroup of G and is called the *cyclic subgroup* generated by g and denoted by $\langle g \rangle$. The number of generators of a cyclic subgroup $\langle g \rangle$ is $\phi(o(g))$, where ϕ is the Euler's totient function. A group G is called *cyclic* if $G = \langle g \rangle$, for some $g \in G$. In a finite cyclic group G, for any factor m of $|G|$, G has a unique subgroup of order m (known as the converse of Langrance's theorem).

A group G is called a *p-group* if the order of each element is some power of p, where p is a prime. For a prime p, if p^m is the highest power of p such that p^m divides $|G|$, then a subgroup $H \leq G$ with the property $|H| = p^m$ is called a *Sylow p-subgroup* of G. The *direct product* of two groups G and H, denoted by $G \times H$, is the group with elements (g,h) where

[5] When the graph is clear from the context, we drop the suffixes.

$g \in G$ and $h \in H$. The group operation of $G \times H$ is given by $(g_1, h_1)(g_2, h_2) = (g_1 g_2, h_1 h_2)$, where the co-ordinate wise operations are the group operations of G and H respectively. A finite group is called a *nilpotent group* if it is a direct product of its Sylow p-subgroups.

We now give the definitions of graphs defined on groups that we discuss in this paper (see [9]).

▶ **Definition 4.** *The* directed power graph *of a group G ($DPow(G)$), is a directed graph with vertex set G, and edge set $E = \{(x, y) : y = x^m$ for some integer $m\}$. The* power graph *of a group G, denoted by $Pow(G)$, is the undirected version $DPow(G)$.*

If (x, y) is an edge in $DPow(G)$, then $o(y)$ divides $o(x)$ whereas if $\{x, y\}$ is an edge in $Pow(G)$, then $o(x)|o(y)$ or $o(y)|o(x)$. Let $\mathcal{DP}ow$ denote the set $\{DPow(G) : G$ is a finite group $\}$. Let $\mathcal{P}ow$ denote the set $\{Pow(G) : G$ is a finite group $\}$.

▶ **Definition 5.** *The enhanced power graph of a group G, denoted $EPow(G)$, is an undirected graph with vertex set G, in which two vertices x and y are adjacent if and only if they are in a common cyclic subgroup of G, i.e., there exists z in G such that $x, y \in \langle z \rangle$.*

Let $\mathcal{EP}ow$ denote the set $\{EPow(G) : G$ is a finite group $\}$.

3 Cyclic cover of a group and organization of the paper

In this section, we introduce the notion of minimal cyclic cover and covering cycle generating set. We start with the following definitions.

▶ **Definition 6.** *We say that a proper cyclic subgroup C of G is a* maximal cyclic subgroup *if for all cyclic subgroups C', $C \leq C'$ implies $C = C'$ or $C' = G$.*

▶ **Definition 7.** *Let G be a finite group. Let $\{C_1, \ldots, C_m\}$ be a set of the cyclic subgroups of G. We say that $\{C_1, \ldots, C_m\}$ is a* minimal cyclic cover (MCC) *if $G = \cup_{i=1}^{m} C_i$ and $\cup_{i \neq j} C_i \neq G$ for all $j = 1, \ldots, m$.*

It is not hard to see from the following lemma that the set of all maximal cyclic subgroups forms the minimum cyclic cover of a non-cyclic group.

▶ **Lemma 8.** *For a cyclic group G the only MCC is $\{G\}$. For non-cyclic groups the set of all maximal cyclic subgroups forms the unique minimal cyclic cover.*

▶ **Definition 9.** *Let $\{C_1, \ldots, C_m\}$ be the minimum cyclic cover (MCC) of G. Each C_i is called a* covering cycle. *For a cyclic group C, let $gen(C)$ be the set of generators of C. An element in $\cup_{i=1}^{m} gen(C_i)$ is called a covering cycle generator or CC-generator. We call a set $\{g_1, g_2, \ldots, g_m\}$ a* covering cycle generating set *(CCG-set) if $\{\langle g_1 \rangle, \langle g_2 \rangle, \ldots, \langle g_m \rangle\} = \{C_1, C_2, \ldots, C_m\}$.*

The above definition includes the case when $m = 1$, i.e., G is cyclic.

Organization of the paper. With the notion of CCG-set defined above, we are now ready to describe the organization of the paper. For the sake of clarity we give the organization in a somewhat nonlinear manner.

How to identify a CCG-set in a power graph or in an enhanced power graph when the underlying group is not given? We design algorithms in Section 5 to solve this problem. These are iterative algorithms that take one of the potential vertices and decide if that vertex can be safely marked as a member of the CCG-set.

The correctness of the algorithm, in the case of power graphs, crucially depends on the structure of closed-twins in the subgraph induced by the closed neighbourhood of the potential vertex that the algorithm examines in each iterative step. In Section 4, we derive a collection of results that characterizes these structures.

In Section 6, we define a set of reduction rules that simplifies the structure of the directed power graph of a group G while retaining all its isomorphism-invariant properties. There are four reductions, Reduction 1, 2, 3, and 4, and they are applied one after the other in that order. The graph obtained after i-th reduction is denoted by $R_i(G)$. We also show that these reductions can be efficiently reversed. At the end of Section 6 we design an isomorphism algorithm for the directed power graphs of nilpotent groups using the structure of R_3.

In Section 7, we show how to obtain the reduced graph R_4 from an input power graph (or an enhanced power) graph given along with a CCG-set.

Combining the above, we have an efficient way of going from a power graph (or an enhanced power graph) to R_4 to R_3 to the directed power graph. This answers Cameron's question positively.

The isomorphism of the power graphs (or the enhanced power graphs) of nilpotent groups can be done as follows: compute the directed power graphs from the input graphs and apply the algorithm developed in Section 6.

4 Structure of closed-twins in a power graph

The structure of closed-twins in a power graph has been studied by Cameron [8], and by Bubboloni and Pinzauti [7]. In this section, we explore the structures of closed-twins in the subgraph of a power graph induced by the closed neighborhood of a vertex. We show in Section 5.1 that these structures can be used to find a CCG-set of a group from the corresponding power graph, even when the group is not given.

First, we note an easy fact about the closed-twins in any graph.

▶ **Lemma 10.** *Let X be a graph and let $v \in V(X)$. Suppose x and y are closed-twins in X. If $x \in N[v]$, then $y \in N[v]$ and x and y are closed-twins in $X[N[v]]$.*

In a group G, an element $x \in G$ and any generator of $\langle x \rangle$ are closed-twins in $\Gamma = Pow(G)$. Therefore, by Lemma 10, we have the following corollary.

▶ **Corollary 11.** *Let $v \in V(\Gamma)$. If $x \in N[v]$, then all the generators of $\langle x \rangle$ are in $N[v]$. Moreover, they are closed-twins in Γ_v, where $\Gamma_v = \Gamma[N[v]]$.*

Now consider a vertex $v \in V(\Gamma)$, where $\Gamma \in \mathcal{P}ow$ and the subgraph $\Gamma_v = \Gamma[N[v]]$ induced on the closed neighborhood of v. For any vertex x in Γ_v, $o(x)|o(v)$ or $o(v)|o(x)$. We partition $V(\Gamma_v)$ according to the order of the vertices in the following way:

$$U_v = \{x \in V(\Gamma_v) : o(x) > o(v)\},\ E_v = \{x \in V(\Gamma_v) : o(x) = o(v)\},$$
$$L_v = \{x \in V(\Gamma_v) : o(x) < o(v)\}$$

For a vertex $x \in U_v$, we have $o(v)|o(x)$ and for a vertex $x \in L_v$, we have $o(x)|o(v)$.

▶ **Definition 12.** *For a prime p, an element x in a group is called* a p-power element *if $o(x) = p^i$ for some $i \geq 0$, x is a* nontrivial p-power element *if $i > 0$.*

▶ **Lemma 13.** *Suppose $v \in V(\Gamma)$ is not a p-power element for any prime p and $x \in U_v$ is a closed-twin of v in Γ_v. Then, $deg_\Gamma(x) > deg_\Gamma(v)$.*

Proof. In this case, there exists prime q and positive integer s such that $q^s|o(x)$ but $q^s \nmid o(v)$. Then, x has a neighbor $z = x^{\frac{o(x)}{q^s}}$ of order q^s (by the converse of Lagrange's theorem in finite cyclic groups). Note that z is not a neighbor of v as $o(z) \nmid o(v)$ and also $o(v) \nmid o(z)$. The latter is true as $o(v)$ is divisible by at least two distinct primes. ◀

The proofs of the next lemma is in Section A.1.

▶ **Lemma 14.** *Let $v \in V(\Gamma)$ be a CC-generator such that $o(v)$ is not a prime power. Let $u \in V(\Gamma_v)$. If $u = e$ or u is a generator of $\langle v \rangle$, then the closed-twins of u in Γ_v are exactly the generators of $\langle v \rangle$ and e; otherwise, the closed-twins of u in Γ_v are exactly the generators of $\langle u \rangle$.*

▶ Remark 15. If $a|b$, $a \neq b$, then (1) $\phi(a)|\phi(b)$, (2) $\phi(a) \leq \phi(b)$ and the equality holds only when $b = 2a$ where a is an odd number.

If $v \in V(\Gamma)$ is a CC-generator, it is easy to see that $o(v) = deg(v) + 1 = |\Gamma_v|$. Let $o(v)$ be not a prime power. Then, using Lemma 14, the set of dominating vertices in Γ_v is the set of generators of $\langle v \rangle$ and identity. Thus, the size of the closed-twin-class of v in Γ_v is $\phi(o(v)) + 1$, i.e., $\phi(|\Gamma_v|) + 1$. Also, for all divisors, $1 < k < o(v)$, of $o(v)$, there exists a closed-twin-class of size $\phi(o(k))$ in Γ_v. The proof of the following corollary can be found in the full version of the paper [12].

▶ **Corollary 16.** *Let $v \in V(\Gamma)$ be a CC-generator and $o(v)$ be not a prime power. Then the following holds: (1) The size of the closed-twin-class of v in Γ_v, i.e., the set of dominating vertices in Γ_v, is $\phi(o(v)) + 1$. (2) For each divisor k of $o(v)$, $1 < k < o(v)$, there is a closed-twin-class of size $\phi(k)$ in Γ_v. Moreover, $\phi(k)$ divides $\phi(o(v))$. (3) There are at most two closed-twin-classes of size greater or equal to $\phi(o(v))$.*

The following theorem is a well-known result [11].

▶ **Theorem 17** ([11])**.** *Let G be a finite group. Then, $\Gamma = Pow(G)$ is complete if and only if G is cyclic of prime power order.*

From the above theorem, the following corollary is immediate.

▶ **Corollary 18.** *Let $v \in V(\Gamma)$ be a p-power element for some prime p. Then, $\Gamma[E_v \cup L_v]$ is a complete graph. Moreover, the elements of $E_v \cup L_v$ are closed-twins of v in Γ_v.*

▶ **Lemma 19.** *Let $v \in V(\Gamma)$ be a nontrivial p-power element and not a CC-generator. Suppose for all $u \in U_v$ such that u is a closed-twin of v in Γ_v, $deg_\Gamma(u)$ is at most $deg_\Gamma(v)$. Let y be a closed-twin of v in Γ_v with maximum order and S denote the set $\{x \in V(\Gamma_v) : o(y)|o(x)$ and $o(x) \neq o(y)\}$. Then, (1) The closed-twins of v are exactly the elements in $\langle y \rangle$. (2) $V(\Gamma_v) = \langle y \rangle \sqcup S$, where $\sqcup$ denotes the disjoint union.*

(3) Moreover, if $o(y) = p^j$ where $j \geq 2$, then p divides $|V(\Gamma_v)|$.

Proof. Suppose $o(v) = p^i$. From Corollary 18, we know that the elements in E_v and L_v are closed-twins of v. Observe that these elements have order p^r for some $r \leq i$. Next, we show that all closed-twins of v in U_v have orders of the form p^l for some $l > i$. Suppose not, then let u be a closed-twin of v in U_v. As $u \in U_v$, p^i divides $o(u)$. Then $o(u) = k \cdot p^i$, where $k > 1$ and $gcd(k, p) = 1$. Since u is a closed-twin of v, $|\Gamma_v| - 1 = deg_{\Gamma_v}(u) = deg_{\Gamma_v}(v) = deg_\Gamma(v)$. Now, $\langle u \rangle$ has an element of order k and this element cannot be a neighbor of v. So, $deg_\Gamma(u) > deg_{\Gamma_v}(u) = deg_\Gamma(v)$. Therefore, $deg_\Gamma(u) > deg_\Gamma(v)$. This is a contradiction. Hence, $o(u) = p^l$, for some $l > i$.

Given that y is the closed-twin of v in Γ_v with maximum order, say p^j. Suppose $z \in \langle y \rangle$. If $y \in E_v \cup L_v$, then clearly $\langle y \rangle = \langle v \rangle$ (because y cannot be in L_v). If $y \in U_v$, then noting that $deg_\Gamma(y) \leq deg_\Gamma(v)$ and y is a closed-twin of v in Γ_v, we can say that z is in Γ_v. In both the cases $\langle y \rangle \subseteq V(\Gamma_v)$. We show that every vertex $w \in V(\Gamma_v)$ is adjacent to z. Since $w \in V(\Gamma_v)$ and y is a closed-twin of v, there is an edge between w and y. So, $o(y)|o(w)$ or $o(w)|o(y)$. In the first case, $z \in \langle y \rangle \subseteq \langle w \rangle$. On the other hand, if $o(w)|o(y)$, then $w \in \langle y \rangle$. So either $z \in \langle w \rangle$ or $w \in \langle z \rangle$ as $\langle y \rangle$ is a cyclic group of prime power order. In any case, $\{w, z\}$ is an edge. So, any element z in $\langle y \rangle$ is a closed-twin of v.

Let z be a closed-twin of v. If $z \in \langle v \rangle$ then $z \in \langle y \rangle$. On the other hand, if $z \notin \langle v \rangle$, then $z \in U_v$. Therefore, $o(z)$ is a power of p. As y is a closed-twin of v, there is an edge between y and z. Therefore, $z \in \langle y \rangle$ or $y \in \langle z \rangle$. If $y \in \langle z \rangle$, we must have $\langle y \rangle = \langle z \rangle$ as $o(z) \leq o(y)$ (as both are p-power order closed-twins of v and y is with maximum order). This forces $\langle z \rangle = \langle y \rangle$. Thus, $z \in \langle y \rangle$ in both cases. This completes the proof of part (1).

Now, we prove part (2). Let $x \in V(\Gamma_v) \setminus \langle y \rangle$. In this case, $x \notin E_v \cup L_v$ by Corollary 18. Since y is a closed-twin of v, $\{x, y\}$ is an edge. Therefore, $x \in \langle y \rangle$ or $y \in \langle x \rangle$. However, by assumption, $x \notin \langle y \rangle$. So, $y \in \langle x \rangle$. Therefore, $o(y)|o(x)$. So, $o(x) = p^j \cdot k$ for some $k > 1$.

Therefore, $V(\Gamma_v) = \langle y \rangle \sqcup \{x \in V(\Gamma_v) : p^j | o(x) \text{ and } o(x) \neq p^j\}$, i.e., $V(\Gamma_v) = \langle y \rangle \sqcup S$.

To prove part (3), we define an equivalence relation $\equiv$ on S as follows: $x_1 \equiv x_2$, if and only if $\langle x_1 \rangle = \langle x_2 \rangle$. Note that if $x \in S$, then generators of $\langle x \rangle$ are in S by Corollary 11. Therefore, the equivalence class of any vertex $x \in S$ is of size $\phi(o(x))$. Recall that, $o(x) = o(y) \cdot k = p^j \cdot k$. Now, as $p^j \geq p^2$, so p divides $\phi(o(x))$. Therefore, p divides $|V(\Gamma_v)|$, as claimed. ◀

5 Finding a CCG-set of a group from its power graph and enhanced power graph

For a directed power graph, even if the underlying group is not given, the set of vertices corresponding to a CCG-set $\{g_1, \ldots, g_m\}$ of the underlying group G can be readily found in the graph. The scenario changes when the input graph is a power graph or an enhanced power graph and the underlying group is *not* given as input. Then, it is not possible to recognise these vertices exactly in the input graph as we can not distinguish two closed-twins g_i and g_i' in $Pow(G)$ (or $EPow(G)$). For example, if we take $\mathbb{Z}_{p^m}$ for some prime p and integer m, then $Pow(\mathbb{Z}_{p^m})$ is a clique (Theorem 17). If the vertices of $Pow(\mathbb{Z}_{p^m})$ are named arbitrarily, then it is not possible to distinguish a generator of $\mathbb{Z}_{p^m}$ from any other vertex. Fortunately, the fact that the underlying group is $\mathbb{Z}_{p^m}$ can be concluded just from the graph by Theorem 17.

Therefore, we aim to do the following: Given a power graph (or an enhanced power graph) Γ, mark a set $\{g_1, g_2, \ldots, g_m\}$ of vertices such that (1) each g_i is a CC-generator or g_i is a closed-twin of a CC-generator g_i' in the graph Γ, and (2) $\{h_1, h_2, \ldots, h_m\}$ is a CCG-set where $h_i = g_i$, if g_i is a CC-generator; otherwise, $h_i = g_i'$.

5.1 Finding a CCG-set of a group from its power graph

The next theorem states that given a power graph Γ as input, we can essentially compute a CCG-set corresponding to Γ, even if the underlying group is not given.

▶ **Theorem 20.** *There is a polynomial-time algorithm that, on input a power graph $\Gamma \in \mathcal{P}ow$, outputs a set $\{g_1, g_2, \ldots, g_m\}$ where g_i is a CC-generator or g_i is a closed-twin of a CC-generator g_i' in the graph Γ such that $\{h_1, h_2, \ldots, h_m\}$ is a CCG-set where $h_i = g_i$, if g_i is a CC-generator, otherwise, $h_i = g_i'$.*

Hence, without loss of generality, we call the set $\{g_1, g_2, \ldots, g_m\}$ as CCG-set and g_i's as CC-generators. Before we give the proof of the above theorem, we need the following definition that is required in the algorithm.

▶ **Definition 21.** *Let d be a positive integer. Let $\Gamma \in \mathcal{P}ow$ and v be a vertex in Γ. Let $T_1, \ldots, T_r$ be the partition of $V(\Gamma_v)$ into closed-twin classes of Γ_v. Similarly, let $S_1, \ldots, S_{r'}$ be the closed-twin classes of $Pow(\mathbb{Z}_d)$. We say that* Γ_v closed-twin-partition-wise matches *with $Pow(\mathbb{Z}_d)$ if (1) the closed-twin class containing the dominating vertices*[6] *of both the graphs have the same size, and (2) $r = r'$ and there is some permutation $\pi \in Sym(r)$ such that $|T_i| = |S_{\pi(i)}|$.*

If v is a CC-generator and $o(v) = d$ is not a prime power then, Γ_v twin-partition-wise-matches with $Pow(\mathbb{Z}_d)$, by Corollary 16.

It is not hard to see that testing if Γ_v closed-twin-partition-wise matches with $Pow(\mathbb{Z}_d)$ can be done in polynomial-time. Also, when d is not prime power, the size of the closed-twin class containing v has size $\phi(d)+1$.

Proof of Theorem 20. The process of finding a CCG-set of the underlying group of a given power graph is described in Algorithm 1.

Algorithm 1 Algorithm to mark a CCG-set in a finite power graph.

Input: $\Gamma \in \mathcal{P}ow$

- First, isolate the case when the power graph Γ is a complete graph using Theorem 17. Then, return a singleton set, consisting of any vertex, as the CCG-set.
- If Γ is not a clique, then mark any of the Dominating vertices as the identity. Next, all vertices except the identity are stored in a list L in decreasing order of their degrees.
- During the algorithm, we use the labels: U (undecided), CC (a CC-generator) and NC (not a CC-generator). To start with, mark all the vertices U in the list. Note that identity is not marked with any label.
- The algorithm marks the vertices further in phases. In each phase, pick the first U marked vertex, say v, in the list L and do the following: Let $a = deg(v) + 1$

  [Rule 1a] *If* a is a prime power and $\Gamma_v = \Gamma[N[v]]$ is complete, then mark v as CC and mark all its neighbors NC.

  [Rule 1b] *Else if* a is a prime power and Γ_v is not complete, then mark v as NC.

  [Rule 2a] *Else if* a is not a prime power and if v has a closed-twin w in Γ_v such that w has been marked NC, then mark v as NC.

  [Rule 2b] *Else* (i.e., a is not a prime power and v does not have a NC marked closed-twin in Γ_v)

  If Γ_v closed-twin-partition-wise matches with $Pow(\mathbb{Z}_d)$, where $d = deg(v) + 1$
  Mark v as CC and all its neighbors NC.
  Else
  Mark v as NC.
- Return the set of vertices marked CC.

A vertex picked at any phase is either marked CC or NC, thereby reducing the number of vertices marked U. Therefore, Algorithm 1 terminates in $O(n)$ phases.

[6] *Dominating vertex* in a graph is a vertex that is adjacent to all other vertices in the graph.

We prove the correctness of the algorithm by induction on the number of phases. In each phase, a set of vertices is relabeled using one of the 4 rules. We prove that this labelling is correct. In phase i, we assume that up to phase $(i-1)$, all the labellings were done correctly. In base case, this means that all the vertices are still labelled U.

If Rule 1a is applied: If v is not a CC-generator, then v is contained in at least one covering cycle. If v is contained in two covering cycles, say $\langle g_1 \rangle$ and $\langle g_2 \rangle$, then Γ_v is not complete, as the CC-generators g_1 and g_2 are not adjacent. Now consider the case when v is contained in exactly one covering cycle $\langle x \rangle$. Then $N_{\Gamma_v}(v) \subseteq N_{\Gamma_v}(x)$. So, if $deg_\Gamma(x) > deg_\Gamma(v)$, then x or one of its closed-twins has already been marked as CC in some previous phase, and then v would have been marked as NC. Now if $deg_\Gamma(x) = deg_\Gamma(v)$, then v and x are closed-twins and thus v is also a CC-generator. This is a contradiction.

If Rule 1b is applied: If v is a CC-generator, then Γ_v is a complete graph. Thus, this step works correctly.

If Rule 2a is applied: If v is a CC-generator, then by Lemma 14 its closed-twins in Γ_v are exactly e (identity) and generators of $\langle v \rangle$. So, if any of the closed-twins is marked NC, it must have been because some other closed-twin is already marked CC in some previous phase $t \leq i-1$ of the algorithm. In phase t, the algorithm would have also marked v as NC.

If Rule 2b is applied: If v is a CC-generator, then Γ_v closed-twin-partition-wise matches with $Pow(\mathbb{Z}_d)$. Now if none of v's closed-twins in Γ_v are already marked CC, then v can be marked CC.

On the other hand, suppose that v is not a CC-generator. We first consider the case when v is contained in only one covering cycle, say generated by x. The proof of the next claim is in Section A.1.

▷ Claim 22. $deg_\Gamma(x) > deg_\Gamma(v)$.

By the above claim, the algorithm considers x and other generators of $\langle x \rangle$ before v. Then, by the induction hypothesis, one of these generators would be marked CC, and v would not be labelled U.

Now we consider the case when v is contained in at least two covering cycles, say $\langle g_1 \rangle$ and $\langle g_2 \rangle$. We prove that if v is not a CC-generator, then Γ_v cannot closed-twin-partition-wise match with $Pow(\mathbb{Z}_d)$. This case is divided into two subcases.

In the 1^{st} subcase, we assume that $o(v)$ is not a prime power. Now we count the closed-twins of v in Γ_v present in each of the sets U_v, E_v and L_v.

If $x \in U_v$ is a closed-twin of v in Γ_v, then by Lemma 13, $deg_\Gamma(x) > deg_\Gamma(v)$. So, the algorithm must have considered x before v. At that phase, the algorithm either marked x as NC or CC. If x was marked as NC, v would not satisfy the condition of Rule 2b (i.e., no closed-twin of v in Γ_v is marked NC). Moreover, if x was marked CC, the algorithm would have marked v as NC. So, v has no closed-twin in U_v.

The number of closed-twins of v in Γ_v which are present in E_v is $\phi(o(v))$. By noting that $\Gamma_v[E_v \sqcup L_v] = Pow(\langle v \rangle)$ and using Lemma 14 on $Pow(\langle v \rangle)$, we see that the only closed-twin of v in L_v is the identity. Therefore, the total number of closed-twins of v in Γ_v is $\phi(o(v)) + 1$.

Now CC-generators g_1 and g_2 have distinct [7] closed-twin-classes of size at least $\phi(o(g_1))$ and $\phi(o(g_2))$. But, $\phi(o(g_i)) \geq \phi(o(v))$ for $i = 1, 2$ by Remark 15. This is a contradiction since $Pow(\mathbb{Z}_d)$ can have at most two closed-twin-classes of size greater than or equal to $\phi(o(v))$, by (3) of Corollary 16.

[7] g_1 and g_2 are not adjacent.

In the 2^{nd} subcase, we assume that $o(v)$ is a prime power, say $o(v) = p^i$ for some prime p and some integer $i > 0$. Consider y and S as in Lemma 19. Note that $deg_\Gamma(y) \leq deg_\Gamma(v)$. Since otherwise, the algorithm would have marked y as NC or CC. In both cases, the algorithm would not satisfy the conditions of Rule 2b.

Subsubcase 1: $o(y) = p^j, j \geq 2$. In this case, by (3) of Lemma 19, p divides $|V(\Gamma_v)|$. Therefore, Γ_v must have a closed-twin-class of size $p-1$ for it to closed-twin-partition-wise match with $Pow(\mathbb{Z}_d)$ (because of (2) of Corollary 16). By (1) of Lemma 19, if $x \in \langle y \rangle$, then the number of closed-twins of x in Γ_v is $|\langle y \rangle| = p^j > p-1$. Also, if $x \in S$, then number of closed-twins of x in $\Gamma_v \geq \phi(o(x)) \geq \phi(o(y)) \geq \phi(p^2) > p-1$. Therefore, there is no closed-twin-class of size $p-1$.

Subsubcase 2: $o(y) = p$. Recall that $\langle g_1 \rangle$ and $\langle g_2 \rangle$ are covering cycles containing v. Since y and v are closed-twins in Γ_v, we can see that $y \in \langle g_1 \rangle \cap \langle g_2 \rangle$. Now by (1) of Lemma 19, the size of the closed-twin-class of v is p. Since $o(y) \big| o(g_1)$ and $o(y) \big| o(g_2)$, the size of the closed-twin-class of both g_1 and g_2 is at least $p-1$. This is not possible by (3) of Corollary 16. ◀

5.2 Finding a CCG-set of a group from its enhanced power graph

▶ **Lemma 23.** *If v is a CC-generator of a group G, then $N_{EPow(G)}[v] \subseteq N_{EPow(G)}[u]$ for all $u \in \langle v \rangle$.*

Algorithm 2 performs the task of finding a CCG-set. The next theorem ensures the correctness of the algorithm.

▶ **Theorem 24.** *Algorithm 2 on input an enhanced power graph*[8] *$\Gamma \in \mathcal{EP}ow$ outputs a set $\{g_1, g_2, \ldots, g_m\}$ where g_i is a CC-generator or g_i is a closed-twin of a CC-generator g_i' in the graph Γ such that $\{h_1, h_2, \ldots, h_m\}$ is a CCG-set where $h_i = g_i$ if g_i is a CC-generator, otherwise, $h_i = g_i'$.*

As before, we call the set $\{g_1, g_2, \ldots, g_m\}$ as CCG-set and g_i's as CC-generators.

Proof. The proof of correctness of Algorithm 2 is by induction on the number of iterations. In any iteration, the first unmarked vertex is marked as CC and its neighbors in the graph are marked as NC. Our goal is to prove that this marking process is correct.

For the base case, $x = v_1$. By Lemma 23, v_1 is either a CC-generator or $v_1 \in \langle g_1 \rangle$, where g_1 is a CC-generator and v_1 is a closed-twin of g_1 in Γ. Since $N[v_1]$ corresponds to $\langle g_1 \rangle$ by Lemma 23, we can safely mark the vertices adjacent to v_1 as NC.

In phase i, we assume that up to iteration $(i-1)$, all the markings were done correctly. Let us pick the first unmarked vertex, say x, in A. It is easy to see that x does not belong to any covering cycle marked till the $(i-1)^{th}$ iteration, i.e., x does not belong to the neighborhood of any CC marked vertex till the $(i-1)^{th}$ iteration. So, again using the same argument given in the base case, it can be seen that the markings done in the i^{th} iteration are correct. ◀

[8] Recall that the underlying group is not given.

Algorithm 2 Algorithm to mark a CCG-set of G in a finite enhanced power graph.

Input: $\Gamma \in \mathcal{EP}ow$

1. Sort the vertices of Γ by their degree in increasing order. Let the sorted array be $A = \{v_1, v_2, \ldots, v_m\}$.
2. Pick the first unmarked element x of A and mark it CC.
   Mark all the elements of $N[x]$ as NC.
3. Pick the next unmarked element in A and repeat Step 2 till all elements of A are marked.

6 Isomorphism of directed power graphs

The isomorphism problems of power graphs, directed power graphs, and enhanced power graphs are equivalent (see [9, 8, 37]). Thus, an algorithm for the isomorphism problem of directed power graphs automatically gives an isomorphism algorithm for power graphs (or enhanced power graphs), provided we can obtain the directed power graph from the power graph (respectively, the enhanced power graph). This is done in Section 7. In the currect section, we focus on the isomorphism problem of directed power graphs. In the last part of this section, we discuss a necessary result that is used in Section 7 for obtaining the directed power graph of an input power graph (or an enhanced power graph).

We perform several reductions on a directed power graph that are isomorphism invariant. The out-degree of a vertex in $DPow(G)$ is the order of the element in the group G, i.e., for a vertex u, $out\text{-}deg(u) = o(u)$. Therefore we can color the vertices by their out-degrees. We call the colored graph $CDPow(G)$. We emphasise that here the colors are numbers, and hence we can perform arithmetic operations on these colors and use the natural ordering of integers inherited by these colors. We recall that by isomorphism we mean color preserving isomorphism when the graphs are colored.

Two vertices u and v are closed-twins in $CDPow(G)$ (in $DPow(G)$ also) if and only if $\langle u \rangle = \langle v \rangle$ in G, i.e., u and v are two generators of the same cyclic subgroup in G. There are $\phi(o(u))$ generators of $\langle u \rangle$ in G. So, for each vertex $u \in CDPow(G)$, there are exactly $\phi(col(u))$ closed-twins in $CDPow(G)$. By the converse of Lagrange's theorem, in each cyclic subgroup of order n, for each divisor k of n, there are exactly $\phi(k)$ generators. So, for each k in the color set of $CDPow(G)$, there are $\phi(k)$ closed-twins in the graph. Observe that u and v are closed-twins in $CDPow(G)$, if and only if $(u, v) \in E(CDPow(G))$ and $col(u) = col(v)$.

Reduction rule 1: Closed-twin Reduction. If there are two closed-twins u and v in $CDPow(G)$, then do a vertex identification of u and v and color the identified vertex with $col(u) = col(v)$. Let $R_1(G)$ denote the reduced graph after applying Reduction rule 1 to $CDPow(G)$.

From the discussion above, the next lemma follows easily.

▶ **Lemma 25.** *$CDPow(G) \cong CDPow(H)$ if and only if $R_1(G) \cong R_1(H)$.*

▶ Remark 26. It is easy to see that we can get back an isomorphic copy of $CDPow(G)$ from $R_1(G)$, by adding $\phi(col(u))$ closed-twins at each vertex u in $R_1(G)$.

Since each vertex has a self-loop, for the purpose of isomorphism we can delete these self-loops. One can check that $R_1(G)$ is a transitively closed directed graph.

Reduction rule 2: Edge-deletion. Let us consider $R_1(G)$. Do the following steps: (1) Delete all self-loops. (2) For all a, b, c, if (a, b) and (b, c) are edges, then mark (a, c) as a transitive edge. Then, delete all edges that are marked as transitive edges. Let $R_2(G)$ denote the resulting graph. Since $R_1(G)$ is the reflexive and transitive closure of $R_2(G)$, we have the following lemma:

▶ **Lemma 27.** *$R_1(G) \cong R_1(H)$ if and only if $R_2(G) \cong R_2(H)$.*

Due to space constraints we omit the proof of the following lemma (see the full version for a proof [12]).

▶ **Lemma 28.** *The reduced graph $R_2(G)$ satisfies the following properties: (1) Vertices with in-degree zero in $R_2(G)$ form a CCG-set of G, (2) If (u, v) is an edge in $R_2(G)$, then $col(u) > col(v)$ and $col(u) = col(v) \cdot p$ for some prime p, (3) $R_2(G)$ is a directed acyclic graph.*

Note that using (1) of Lemma 28, we can easily find a set of vertices, say $\{g_1, g_2, \ldots, g_m\}$, that form a covering cycle generating set (CCG-set) of G.

Reduction rule 3: Removing the direction. Remove the direction of the edges in $R_2(G)$ to obtain an undirected colored graph $R_3(G)$.

Note that the CCG-set of G can still be identified easily in $R_3(G)$: A vertex g is a CC-generator if and only if all its neighbours have smaller orders (or colors).

The following result is an easy consequence of (2) of Lemma 28.

▶ **Lemma 29.** *$R_2(G) \cong R_2(H)$ if and only if $R_3(G) \cong R_3(H)$.*

▶ **Definition 30.** *A path $u_1u_2 \ldots u_l$ in $R_3(G)$ is said to be a* descendant path *if $col(u_i) > col(u_{i+1})$. The vertices in the graph reachable from u using descendant path are called descendant reachable vertices from u. We denote the set of descendant reachable vertices from u in $R_3(G)$ by $Des(u)$.*

Observe that $Des(u)$ in $R_3(G)$ is same as the closed out-neighborhood of u in $R_1(G)$. The colors of the vertices of $Des(u)$ in $R_3(G)$ form the set of all divisors of $col(u)$. Also, no two vertices of $Des(u)$ in R_3 have the same color.

▶ **Theorem 31.** *If G is a finite p-group, then $R_3(G)$ is a colored tree.*

Proof. Let $|G| = p^\alpha$. Any edge in $R_3(G)$ is of the form $\{u, v\}$ where by using (2) of Lemma 28 we can assume without loss of generality that $col(u) = p^t$ and $col(v) = p^{t-1}$, for some $t \in \{1, 2, \ldots, \alpha\}$. Suppose the graph contains a cycle $u_0u_1u_2 \ldots u_nu_0$. By (2) of Lemma 28, we can assume without loss of generality that the colors of the vertices form the following sequence: $p^t p^{t-1} p^{t-2} \ldots p^{t-(i-1)} p^{t-i} p^{t-(i-1)} \ldots p^{t-1} p^t$ for some i. Now, $u_1, u_n \in Des(u_0)$ such that $col(u_1) = col(u_n) = p^{t-1}$. This is a contradiction since no two vertices of $Des(u)$ for any vertex u have the same color. Hence, our assumption is wrong and $R_3(G)$ has no cycle. ◀

Since the isomorphism of trees can be tested in linear time (see, for example, [1]), the isomorphism of the directed power graphs of p-groups can also be tested in polynomial-time.[9] Now we extend our algorithm to check the isomorphism of directed power graphs of finite nilpotent groups. For that, we use the following two results.

[9] It can actually be done in linear time.

▶ **Lemma 32** ([31]). *Let G_1 and G_2 be two finite groups such that $|G_1|$ and $|G_2|$ are co-prime to each other. Then, $DPow(G_1 \times G_2) = DPow(G_1) \boxtimes DPow(G_2)$, where $\boxtimes$ denotes the strong product of two graphs.*

▶ **Lemma 33** ([27]). *There exists a unique prime factor decomposition of a simple connected* [10] *directed graph with respect to strong product and the uniqueness is up to isomorphism and ordering of the factors.*

It is easy to verify that the next lemma follows from the above two lemmas. However, we can prove Lemma 34 without using Lemma 33 and the proof is given in Section A.1.

▶ **Lemma 34.** *Let $G = G_1 \times \cdots \times G_k$ and $H = H_1 \times \cdots \times H_k$ where $|G_i| = |H_i|$ for all $1 \leq i \leq k$. Suppose $gcd(|G_i|, |G_j|) = gcd(|H_i|, |H_j|) = 1$, for all $1 \leq i < j \leq k$. Then, $DPow(G) \cong DPow(H)$ if and only if $DPow(G_i) \cong DPow(H_i)$, for all $1 \leq i \leq k$.*

We are now ready to present one of the main results of the paper. Namely, we show that the isomorphism of the directed power graphs of nilpotent groups can be tested in polynomial-time. Let $\mathcal{DP}ow_{nil} = \{DPow(G) \ : \ G \text{ is a finite nilpotent group}\}$.

Theorem 31 and Lemma 34 suggest that obtaining the directed power graphs corresponding to the factor groups might be useful. One approach would be to decompose an input directed power graph into prime factors with respect to strong product in polynomial-time using the algorithm by Hellmuth et al. [21]. Note that, in a general setting, the prime graphs in the strong product decomposition may not correspond to the directed power graphs of the direct factors of the underlying group. We are also not sure if the Sylow-p subgroups of a nilpotent group generate prime graphs. If not, then just applying the algorithm of Hellmuth et al. is not enough and we need to regroup the prime factors properly to apply Theorem 31. Fortunately, all these complications can be easily avoided as shown in the next theorem.

▶ **Theorem 35.** *There is an efficient polynomial-time algorithm that on inputs $\Gamma_1, \Gamma_2 \in \mathcal{DP}ow_{nil}$ checks if Γ_1 and Γ_2 are isomorphic.*

Proof. We know that a finite nilpotent group is the direct product of its Sylow subgroups. Since the orders of the Sylow subgroups are coprime with each other, by Lemma 34, Γ_1 and Γ_2 are isomorphic if and only if for each prime p dividing $|V(\Gamma_1)|$ (which is same as the order of the underlying group), the directed power graphs of the Sylow p-subgroups of the underlying groups of Γ_1 and Γ_2 are isomorphic. Therefore, if we can find the directed power graphs of the Sylow subgroups associated with each prime divisor, we can test the isomorphism of Γ_1 and Γ_2.

While the underlying groups are not given as input, we can still compute the directed power graph of a Sylow p-subgroup of an input graph by finding the set V_p of all vertices whose order in the underlying group is p^i for some $i \geq 0$. More precisely, the subgraph induced by the set V_p is the directed power graph associated with the Sylow p-subgroup. Note that the order of a vertex (which is also an element in the underlying group) is the out-degree of the vertex in the directed power graph. ◀

We show that all the isomorphism invariant information of $R_3(G)$ is captured by a) the CCG-set of G in $R_3(G)$ along with their colors, and b) elements corresponding to their pairwise common neighborhood along with their colors. For this, we do a further reduction. The results in the rest of this section are required in Section 7.

[10] Here *connected directed graph* means that the underlying undirected graph is connected.

We define a new simple undirected colored graph $HD[n] = (V, E)$ for any natural number n, where $V = \{d : d|n\}$. The name of each vertex is treated as its color, i.e., here $col(v) = v$. The edge set is $E = \{\{u, v\} : v = u \cdot p \text{ or } u = v \cdot p \text{ for some prime } p\}$. One can see that $HD[n]$ is the Hasse diagram of the Poset defined over the set of all divisors of n with respect to the divisibility relation. Moreover, $HD[n]$ is also isomorphic to $R_3(\mathbb{Z}_n)$ (as a consequence of (2) of Lemma 28).

▶ Remark 36. (1) It is easy to see that $R_3(G)[Des(g_i)]$ is isomorphic to $HD[col(g_i)]$ for all $1 \leq i \leq m$. We can see that the isomorphism is unique as in each of these graphs, there is only one vertex with a particular color.
(2) Note that $\{y, y'\} \in E(R_3(G))$ if and only if (a) y, $y' \in Des(g_i)$ for some $1 \leq i \leq m$ and (b) $col(y) = p \cdot col(y')$ or $col(y') = p \cdot col(y)$ for some prime p.

Let $\bar{I}(i, j)$ denote the vertex in $R_3(G)$ that is of maximum color among the common descendant reachable vertices from both g_i and g_j. It is not hard to see that in the group G, $col(\bar{I}(i, j)) = |\langle g_i \rangle \cap \langle g_j \rangle|$. Note that for two distinct pairs (i, j) and (i', j'), $\bar{I}(i, j)$ and $\bar{I}(i', j')$ can be the same vertex in $R_3(G)$. It is not hard to see the proof of the following claim.

▷ Claim 37. In $R_3(G)$, $gcd(col(\bar{I}(i, j)), col(\bar{I}(s, j)))$ divides $col(\bar{I}(i, s))$.

Reduction rule 4. Consider $R_3(G)$. Recall that, in $R_3(G)$ a CCG-set $\{g_1, g_2, ..., g_m\}$ of G can be readily found. We make a new graph $R_4(G)$ as follows:
(1) Introduce the vertices $g_1, g_2, \ldots, g_m$ with their colors. (2) For each pair (i, j), $1 \leq i < j \leq m$, do the following: Find the vertex $\bar{I}(i, j)$ that is of maximum color among the descendant reachable vertices from both g_i and g_j. We add a vertex $I(i, j)$ in $R_4(G)$ and color it with $col(\bar{I}(i, j))$. Add edges $\{g_i, I(i, j)\}$ and $\{g_j, I(i, j)\}$.

Note that $R_4(G)$ is a bipartite graph where one part is a CCG-set and another part contains vertices marked as $I(i, j)$ for all (i, j). In $R_4(G)$, for distinct pairs (i, j) and (i', j'), $I(i, j)$ and $I(i', j')$ are distinct vertices, while in $R_3(G)$, $\bar{I}(i, j)$ and $\bar{I}(i', j')$ may be the same vertex. In other words, $R_4(G)$ may have several copies of vertex $\bar{I}(i, j)$.

We now present an algorithm to get back an isomorphic copy of $R_3(G)$ from $R_4(G)$.

Idea of the algorithm. In $R_4(G)$, we have a set of colored CC-generators. Also, there exist vertices $I(i, j)$ corresponding to each pairwise intersection of maximal cyclic subgroups $\langle g_i \rangle$ and $\langle g_j \rangle$ in G. $I(i, j)$ is the only common neighbor of g_i and g_j in $R_4(G)$. Using this information, we construct $R_3(G)$ in an iterative manner. First, we describe a sketch of the idea behind the process. There are m iterations in the process. In the 1^{st} iteration, we introduce $HD[col(g_1)]$. One can easily verify that $R_3(G)[Des(g_1)]$ is isomorphic to $HD[col(g_1)]$ (by (2) of Remark 36). In the 2^{nd} iteration, we introduce $HD[col(g_2)]$. As we know the color of $I(1, 2)$, we have information about the set of vertices common to both $HD[col(g_1)]$ and $HD[col(g_2)]$. Let u and v be the vertices with color $col(I(1, 2))$ in $HD[col(g_1)]$ and $HD[col(g_2)]$ respectively. We identify (via vertex-identification) the vertices with the same colors in $Des(u)$ (which is in $HD[col(g_1)]$) and $Des(v)$ (which is in $HD[col(g_2)]$)[11]. One can see that the resulting graph is isomorphic to the induced subgraph of $R_3(G)$ on $Des(g_1) \cup Des(g_2)$. Inductively the algorithm introduces $Des(g_1) \cup Des(g_2) \cup \ldots \cup Des(g_{j-1})$ at the end of the $(j-1)^{th}$ iteration. In the j^{th} iteration, we introduce $HD[col(g_j)]$. It is easy to note that the set of

[11] Since $HD[col(g_i)]$ is isomorphic to $R_3(G)[Des(g_i)]$, we can use the concept of Des in the graph $HD[col(g_i)]$ for all i.

vertices in $HD[col(g_j)]$ that are contained in $Des(g_j) \cap Des(g_s)$ for all $s \leq j-1$ has already been introduced. So, we need to identify the vertices introduced by the algorithm earlier with the corresponding subset of vertices in $HD[col(g_j)]$. We get the information of such vertices using the color of $I(s,j)$ for $s \leq j-1$. The details and correctness of the algorithm (Algorithm 3) are given in Section A.2.

7 Reconstruction Algorithms

Cameron asked the following question: "Question 2 [9]: Is there a simple algorithm for constructing the directed power graph or enhanced power graph from the power graph, or the directed power graph from the enhanced power graph?" Bubboloni and Pinzauti [7] gave an algorithm to reconstruct the directed power graph from the power graph. In this section, we show that with the tools we have developed, we can readily design algorithms to reconstruct the directed power graph from both the enhanced power graph and the power graph.

Suppose we are given a power graph (or an enhanced power graph) of some finite group G as input, i.e., $\Gamma = Pow(G)$ (or, $\Gamma = EPow(G)$). However, the group G is not given. As discussed in Section 5, we can find a CCG-set for G from the input graph. Next, we describe how to obtain a graph isomorphic to $R_4(G)$ from the CCG-set. From the vertices corresponding to a CCG-set of G, say $\{g_1, g_2, \ldots, g_m\}$, we get the information about their degree in Γ and the pairwise common neighborhood of g_i and g_j in the respective graph. This immediately gives us $R_4(G)$. From $R_4(G)$, we know how to get back an isomorphic copy of $DPow(G)$ using the results in Section 6. All the steps in the process can be performed in polynomial-time.

For any two vertices u and v we can easily decide when to put an edge between them in the enhanced power graph by looking into the corresponding directed power graph: there is an edge $\{u, v\}$ in the enhanced power graph, if and only if both u and v belong to the closed-out-neighbourhood of some vertex in the directed power graph. In this way, we can construct the enhanced power graph from an input directed power graph. Therefore, we get a complete solution to Cameron's question.

References

1 Alfred V. Aho and John E. Hopcroft. *The design and analysis of computer algorithms*. Pearson Education India, 1974.

2 László Babai. Graph isomorphism in quasipolynomial time. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 684–697, 2016.

3 László Babai, D. Yu Grigoryev, and David M. Mount. Isomorphism of graphs with bounded eigenvalue multiplicity. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, pages 310–324, 1982. doi:10.1145/800070.802206.

4 László Babai and Eugene M Luks. Canonical labeling of graphs. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, pages 171–183, 1983. doi:10.1145/800061.808746.

5 Hans L. Bodlaender. Polynomial algorithms for graph isomorphism and chromatic index on partial k-trees. *Journal of Algorithms*, 11(4):631–643, 1990. doi:10.1016/0196-6774(90)90013-5.

6 Ravi B. Boppana, Johan Hastad, and Stathis Zachos. Does co-np have short interactive proofs? *Information Processing Letters*, 25(2):127–132, 1987. doi:10.1016/0020-0190(87)90232-8.

7 Daniela Bubboloni and Nicolas Pinzauti. Critical classes of power graphs and reconstruction of directed power graphs. *arXiv preprint arXiv:2211.14778*, 2022.

8 Peter J. Cameron. The power graph of a finite group, ii. *Journal of Group Theory*, 13(6):779–783, 2010.

9 Peter J. Cameron. Graphs defined on groups. *International Journal of Group Theory*, 11(2):53–107, 2022.

10 Peter J. Cameron and Shamik Ghosh. The power graph of a finite group. *Discrete Mathematics*, 311(13):1220–1222, 2011. doi:10.1016/J.DISC.2010.02.011.

11 Ivy Chakrabarty, Shamik Ghosh, and M. K. Sen. Undirected power graphs of semigroups. In *Semigroup Forum*, volume 78, pages 410–426. Springer, 2009.

12 Bireswar Das, Jinia Ghosh, and Anant Kumar. The isomorphism problem of power graphs and a question of cameron. *arXiv preprint arXiv:2305.18936*, 2023. doi:10.48550/arXiv.2305.18936.

13 Min Feng, Xuanlong Ma, and Kaishun Wang. The full automorphism group of the power (di) graph of a finite group. *European Journal of Combinatorics*, 52:197–206, 2016. doi:10.1016/J.EJC.2015.10.006.

14 Valentina Grazian and Carmine Monetta. A conjecture related to the nilpotency of groups with isomorphic non-commuting graphs. *Journal of Algebra*, 633:389–402, 2023.

15 Joshua A. Grochow and Youming Qiao. On p-group isomorphism: search-to-decision, counting-to-decision, and nilpotency class reductions via tensors. In *36th Computational Complexity Conference (CCC 2021)*, volume 200, 2021. doi:10.4230/LIPICS.CCC.2021.16.

16 Martin Grohe and Sandra Kiefer. A linear upper bound on the Weisfeiler-Leman dimension of graphs of bounded genus. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.

17 Martin Grohe and Daniel Neuen. Isomorphism, canonization, and definability for graphs of bounded rank width. *Communications of the ACM*, 64(5):98–105, 2021. doi:10.1145/3453943.

18 Martin Grohe, Daniel Neuen, and Pascal Schweitzer. A faster isomorphism test for graphs of small degree. *SIAM Journal on Computing*, pages FOCS18–1, 2020.

19 Martin Grohe, Daniel Neuen, Pascal Schweitzer, and Daniel Wiebking. An improved isomorphism test for bounded-tree-width graphs. *ACM Transactions on Algorithms (TALG)*, 16(3):1–31, 2020. doi:10.1145/3382082.

20 Martin Grohe and Pascal Schweitzer. Isomorphism testing for graphs of bounded rank width. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 1010–1029. IEEE, 2015. doi:10.1109/FOCS.2015.66.

21 Marc Hellmuth and Tilen Marc. On the cartesian skeleton and the factorization of the strong product of digraphs. *Theoretical Computer Science*, 565:16–29, 2015. doi:10.1016/J.TCS.2014.10.045.

22 Wilfried Imrich, Sandi Klavzar, and Douglas F. Rall. *Topics in graph theory: Graphs and their Cartesian product*. CRC Press, 2008.

23 G. James Oxley. *Matroid Theory*. Oxford graduate texts in mathematics. Oxford University Press, 2006.

24 Andrei V. Kelarev and Stephen J. Quinn. A combinatorial property and power graphs of groups. *Contributions to general algebra*, 12(58):3–6, 2000.

25 Ajay Kumar, Lavanya Selvaganesh, Peter J. Cameron, and T. Tamizh Chelvam. Recent developments on the power graph of finite groups–a survey. *AKCE International Journal of Graphs and Combinatorics*, 18(2):65–94, 2021. doi:10.1080/09728600.2021.1953359.

26 Eugene M. Luks. Isomorphism of graphs of bounded valence can be tested in polynomial time. *Journal of computer and system sciences*, 25(1):42–65, 1982. doi:10.1016/0022-0000(82)90009-5.

27 Ralph McKenzie. Cardinal multiplication of structures with a reflexive relation. *Fundamenta Mathematicae*, 70(1):59–101, 1971.

28 Gary Miller. Isomorphism testing for graphs of bounded genus. In *Proceedings of the twelfth annual ACM symposium on Theory of computing*, pages 225–235, 1980. doi:10.1145/800141.804670.

29 Gary L. Miller. On the nlog n isomorphism technique (a preliminary report). In *Proceedings of the tenth annual ACM symposium on theory of computing*, pages 51–58, 1978. doi:10.1145/800133.804331.

30 Dave Witte Morris, Joy Morris, and Gabriel Verret. Isomorphisms of cayley graphs on nilpotent groups. *New York J. Math*, 22:453–467, 2016.

31 Himadri Mukherjee. Hamiltonian cycles of power graph of abelian groups. *Afrika Matematika*, 30:1025–1040, 2019.

32 Daniel Neuen. Isomorphism testing for graphs excluding small topological subgraphs. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1411–1434. SIAM, 2022. doi:10.1137/1.9781611977073.59.

33 Joseph J. Rotman. *An introduction to the theory of groups*, volume 148. Springer Science & Business Media, 2012.

34 V. Vikraman Arvind and Peter J. Cameron. Recognizing the commuting graph of a finite group. *arXiv preprint*, 2022. doi:10.48550/arXiv.2206.01059.

35 Douglas B. West. *Introduction to Graph Theory*. Prentice Hall, September 2000.

36 Daniel Wiebking. Graph isomorphism in quasipolynomial time parameterized by treewidth. In *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.

37 Samir Zahirović, Ivica Bošnjak, and Rozália Madarász. A study of enhanced power graphs of finite groups. *Journal of Algebra and Its Applications*, 19(04):2050062, 2020.

A Appendix

A.1 Omitted Proofs

Proof of Lemma 14. Let $o(v) = p_1^{r_1}p_2^{r_2}\dots p_k^{r_k}$, where $k \geq 2$. The case when $u = e$ or u is a generator of $\langle v\rangle$ is easy as $N[u] = V(\Gamma_v)$ for any such element. Otherwise, since v is a CC-generator, $\langle u\rangle \lneq \langle v\rangle$. For u and z to be closed-twins, we must have $u \in \langle z\rangle$ or $z \in \langle u\rangle$. We show that for z to be a closed-twin of u, its order must be the same as that of u. We consider the case when $z \in \langle u\rangle$. The other case can be handled similarly. In this case, we have $o(z)|o(u)$.

Suppose both u and z are p-power elements for some prime $p \in \{p_1, p_2, \dots, p_k\}$. Moreover, without loss of generality, assume that $o(u) = p_1^{s_1}$ and $o(z) = p_1^{s_1'}$ where $s_1 > s_1'$. Note that $r_1 \geq s_1$. In this case, there is an element in $V(\Gamma_v)$ of order $p_1^{s_1'}p_2$ which is adjacent to z but not to u. More precisely, this element is an element in $\langle v\rangle$ of order $p_1^{s_1'}p_2$. So, in this case, u and z are not closed-twins in Γ_v.

Now suppose $o(u)$ is not a prime power. We first take z to be non-identity. Then, let $o(u) = p_1^{s_1}\dots p_k^{s_k}$, where $k \geq 2$. Let $o(z) = p_1^{s_1'}\dots p_k^{s_k'}$, where $s_j \geq s_j'$. Assume without loss of generality that $s_1 > s_1'$. As $o(u)$ is not a prime power order, we can take $s_2 \neq 0$. Now if $s_2' = 0$, consider an element x of order p_2 in Γ_v. Then x is a neighbor of u, but not of z. On the other hand, if $s_2' \neq 0$, we take an element y of order $p_1^{s_1}$. Again y is a neighbor of u but not of z. So, in this case also, u and z are not closed-twins in Γ_v.

Now suppose $o(u)$ is not a prime power, i.e., $o(u) = p_1^{s_1}p_2^{s_2}\dots p_k^{s_k}$ where $k \geq 2$ and z is the identity. We recall that since u is not a generator $\langle v\rangle$, there exists i such that $r_i > s_i$. We take an element x of order $p_i^{r_i}$ in Γ_v. One can check that x is adjacent to z but not to u. So, here also u and z are not closed-twins in Γ_v.

Note that if $o(u) = o(z)$, then they are closed-twins in Γ_v. ◀

▷ Claim 22. $deg_\Gamma(x) > deg_\Gamma(v)$.

Proof. As v is contained in only one covering cycle, we have $N_\Gamma(v) \subseteq N_\Gamma(x)$. This implies $deg_\Gamma(x) \geq deg_\Gamma(v)$. Moreover in Γ_v, the vertices x and v are closed-twins. If $o(x) = p^i$, then $deg(x) + 1 = p^i$. The graph $\Gamma_x = \Gamma[N[x]]$ is complete. So, $deg(v) + 1 = p^i$. Therefore, this case cannot arise. On the other hand, if $o(x)$ is not a prime power, we can apply [12] Lemma 14 and since $v \neq e$ and v is not a CC-generator, we can see that v and x are not closed-twins in Γ_x. Thus,$deg_\Gamma(x) > deg_\Gamma(v)$. ◁

Proof of Lemma 34. It is enough to prove the lemma for $k = 2$. Let $f_1 : V(DPow(G_1)) \to V(DPow(H_1))$ and $f_2 : V(DPow(G_2)) \to V(DPow(H_2))$ be two isomorphisms from $DPow(G_1)$ to $DPow(H_1)$ and from $DPow(G_2)$ to $DPow(H_2)$ respectively. Let us define $f : V(DPow(G)) \to V(DPow(H))$ as $f((u_1, u_2)) = (f_1(u_1), f_2(u_2))$. Since f_1 and f_2 are bijections, so is f. We show that f preserves the edge relations between $DPow(G)$ and $DPow(H)$. Let us consider an edge $((u_1, u_2), (v_1, v_2))$ from $E(DPow(G)) = E(DPow(G_1) \boxtimes DPow(G_2))$ (This equality follows from Lemma 32.). Now from Definition 2 and the facts that f_1 and f_2 are isomorphisms from $DPow(G_1)$ to $DPow(H_1)$ and from $DPow(G_2)$ to $DPow(H_2)$ respectively, we have the following three scenarios:

1. $u_1 = v_1$ and $(u_2, v_2) \in E(DPow(G_2))$. In this case, $f_1(u_1) = f_1(v_1)$ and $(f_2(u_2), f_2(v_2)) \in E(DPow(H_2))$.
2. $u_2 = v_2$ and $(u_1, v_1) \in E(DPow(G_1))$. In this case, $f_2(u_2) = f_2(v_2)$ and $(f_1(u_1), f_1(v_1)) \in E(DPow(H_1)$.
3. $(u_1, v_1) \in E(DPow(G_1))$ and $(u_2, v_2) \in E(DPow(G_2))$. In this case, $(f_1(u_1), f_1(v_1)) \in E(DPow(H_1))$ and $(f_2(u_2), f_2(v_2)) \in E(DPow(H_2))$.

In all the three scenarios, by Definition 2, we have $((f_1(u_1), f_2(u_2)), (f_1(v_1), f_2(v_2))) \in E(Dpow(H_1) \boxtimes DPow(H_2))$. Therefore by Lemma 32, $(f((u_1, u_2)), f((v_1, v_2))) \in E(DPow(H))$.

For the other direction, let $f : V(DPow(G)) \to V(DPow(H))$ be an isomorphism between $DPow(G)$ and $DPow(H)$. Consider the sets $A_i = \{(u, v) \in V(DPow(G)) : out\text{-}deg((u, v))$ divides $|G_i|\}$ and $B_i = \{(u', v') \in V(DPow(H)) : out\text{-}deg((u', v'))$ divides $|H_i|\}$ for $i = 1, 2$. Recall that here the out-degree of a vertex is the order of the element and $o((u, v)) = o(u) \cdot o(v)$. Since $|G_1| \times |G_2| = |G|$ and $gcd(|G_1|, |G_2|) = 1$, it is easy to see that A_i indeed corresponds to $V(DPow(G_i))$ for $i = 1, 2$. Also, the subgraph of $DPow(G_1 \times G_2)$ induced by A_i corresponds to $DPow(G_i)$ for $i = 1, 2$. Similarly, we can see that B_i corresponds to $V(DPow(H_i))$ and the subgraph induced by B_i corresponds to $DPow(H_i)$ for $i = 1, 2$. Now the isomorphism f preserves the out-degrees of the vertices. We denote the restriction of f on A_i by f_i. Then it is easy to see that f_i is a bijection from A_i to B_i. Also, there is only one element, namely the identity element, of out-degree 1 (self-loop) and common in both A_1 and A_2. Also, that element is unique in $DPow(G)$. One can see that $f_i : V(DPow(G_i)) \to V(DPow(H_i))$ is an isomorphism between $DPow(G_i)$ and $DPow(H_i)$, for all $i = 1, 2$. ◀

A.2 Algorithm to construct an isomorphic copy of Reduction graph

Here, we give a detailed description of the algorithm to construct $R_3(G)$ from $R_4(G)$ discussed in Section 6.

As indicated in the idea behind the algorithm in Section 6 and in Line 12 of Algorithm 3, vertices in the old graph and $HD[col(g_j)]$ are identified. In Claim 40 we show that these vertices can be identified without conflict.

[12] Here x and v are to be treated as the variables v and u in Lemma 14.

Algorithm 3 To construct an isomorphic copy of $R_3(G)$ from $R_4(G)$.

Input: $R_4(G)$

1: $X_1 \leftarrow HD[col(g_1)]$
2: $j \leftarrow 2$
3: **while** $j \leq m$ **do**
4: Introduce $Y_j = HD[col(g_j)]$
5: $s \leftarrow 1$
6: $h_{j,0} \leftarrow \emptyset$ ▷ Mapping for vertex identification
7: **while** $s \leq j-1$ **do**
8: Consider $I(s,j)$.
9: $h_{j,s} \leftarrow h_{j,s-1} \cup$ $\{(u,v) : col(u) = col(v)$ where $u \in Des(g_s) \subseteq V(X_{j-1})$ s.t $col(u) | col(I(s,j))$ and $v \in V(Y_j)\}$
10: $s \leftarrow s+1$
11: **end while**
12: For all $(u,v) \in h_{j,j-1}$ vertex-identify u and v and color the new vertex with $col(u)$.
13: $X_j \leftarrow$ The graph obtained after the above vertex identification of X_{j-1} and Y_j.
14: $j \leftarrow j+1$
15: **end while**
16: Return X_m

▶ **Lemma 38.** *The graph X_m returned by Algorithm 3 is isomorphic to $R_3(G)$.*

Proof. We show by induction on j that the constructed graph up to the j^{th} step is isomorphic to the subgraph of $R_3(G)$ induced on $Des(g_1) \cup Des(g_2) \cup \ldots \cup Des(g_j)$. This shows that after the m^{th} iteration, we can get an isomorphic copy of $R_3(G)$.

▷ Claim 39. $X_j \cong R_3(G)[Des(g_1) \cup Des(g_2) \cup \cdots \cup Des(g_j)]$, $\forall 1 \leq j \leq m$.

Proof of claim. For simplicity, we denote $R_3(G)[Des(g_1) \cup Des(g_2) \cup \cdots \cup Des(g_j)]$ by $R_3(j)$ in the remaining part of the proof. With this, $R_3(1)$ denotes $R_3(G)[Des(g_1)]$.

By Remark 36, $X_1 = HD[col(g_1)]$ is isomorphic to $R_3(1)$ by a unique isomorphism, say f_1. If we take f_0 to be the empty map, then f_1 extends f_0.

We prove by induction on j that X_j is isomorphic to $R_3(j) = R_3[Des(g_1) \cup Des(g_2) \cup \cdots \cup Des(g_j)]$ via a map f_j that extends the isomorphism f_{j-1}.

By induction hypothesis, let us assume that $X_{j-1} \cong R_3(G)[Des(g_1) \cup \cdots \cup Des(g_{j-1})]$ and f_{j-1} is an isomorphism between X_{j-1} and $R_3(j-1)$ derived by extending f_{j-2}. We show that f_j is an extension of f_{j-1} and f_j is an isomorphism between X_j and $R_3(j)$.

However, before we go into the details of the inductive case, we address the following important issue.

In the j^{th} iteration of the outer while loop and just after the execution of Line 4 of Algorithm 3, the current graph is the disjoint union of X_{j-1} and Y_j. Now to get X_j, some vertices of X_{j-1} and Y_j are vertex-identified using the tuples stored in $h_{j,j-1}$ as described in Line 12 of Algorithm 3. Observe that two vertices in Y_j cannot be identified with the same vertex in X_{j-1}, because in $Y_j = HD[col(g_j)]$, no two vertices have the same color. However, there is a possibility that two or more vertices of X_{j-1} are assigned to be identified with the same vertex of Y_j. We show that this case does not arise. To do this, we first define the following sets:

$$Y_{j,1} = \{v \in V(Y_j) \ : \ col(v) \big| col(I(1,j))\}$$
$$Y_{j,s} = Y_{j,s-1} \cup \{v \in V(Y_j) \ : \ col(v) \big| col(I(s,j))\}, \quad s = 2, \ldots, j-1$$
$$X_{j-1,1} = \{u \in V(X_{j-1}) \ : \ u \in Des(g_1) \text{ and } col(u) \big| col(I(1,j))\}$$
$$X_{j-1,s} = X_{j-1,s-1} \cup \{u \in V(X_{j-1}) \ : \ u \in Des(g_s) \text{ and } col(u) \big| col(I(s,j))\}, \quad s = 2, \ldots, j-1$$

Now $h_{j,j-1}$ is updated from $h_{j,0} = \emptyset$ by the following rule: $h_{j,s} = h_{j,s-1} \cup \{(u,v) \mid col(u) = col(v)$ where $u \in X_{j-1,s}$ and $v \in Y_{j,s}\}$ (as described in Line 9 in Algorithm 3)[13]. Since there is a unique vertex of any particular color in Y_j, we can see $h_{j,s}$ as a well-defined function from $X_{j-1,s}$ to $Y_{j,s}$. Now to show that $h_{j,j-1}$ gives a conflict-free vertex identification process, we show that $h_{j,s}$ is a bijection and an extension of $h_{j,s-1}$. Since $h_{j,s-1} \subseteq h_{j,s}$, it is enough to prove the following claim:

▷ Claim 40. The map $h_{j,s} : X_{j-1,s} \to Y_{j,s}$ is a bijection, for all $1 \leq s \leq j-1$.

Proof of claim: First, we show that $h_{j,s}$ is onto for all $s = 1, \ldots, j-1$. For this, take a vertex v from $Y_{j,s}$. Then $col(v) | col(I(i,j))$ for some $i \leq s$. So,[14] there exists a vertex $u \in Des(g_i)$ in $X_{j-1,s}$ such that $col(u) = col(v)$ and $h_{j,s}(u) = v$.

Now we prove that $h_{j,s}$ is one-to-one using induction on s. For the base case, it is easy to see that $h_{j,1} : X_{j-1,1} \to Y_{j,1}$ is a bijection since $X_{j-1,1}$ and $Y_{j,1}$ contains colored vertices corresponding to each divisor of $col(I(1,j))$ and color of each vertex is distinct. By induction hypothesis we assume that $h_{j,s-1} : X_{j-1,s-1} \to Y_{j,s-1}$ is a bijection. Now for the inductive case, we consider $h_{j,s} : X_{j-1,s} \to Y_{j,s}$. We need to prove that $h_{j,s}$ is one-one. Suppose that $u \in X_{j-1,s}$ is paired with $v \in Y_{j,s}$ to be stored at $h_{j,s}$ in the s^{th} iteration of the inner while loop (Line 9 of Algorithm 3). We need to argue that the pairing does not violate the one-to-one condition. We do this in two cases.

Case 1: The vertex v was not encountered in any of the previous iterations, i.e., $v \notin Y_{j,s-1}$. So by definition of $X_{j-1,s-1}$, there is no vertex of color $col(v)$ in $X_{j-1,s-1}$. Since $col(u) = col(v)$, we have $u \in X_{j-1,s} \setminus X_{j-1,s-1}$. So, (u,v) is added to $h_{j,s}$ in the s^{th} iteration only, where v is in $Y_{j,s}$. Therefore, $X_{j-1,s}$ contains exactly one vertex of color $col(u)$. This implies that v cannot be paired with any vertex except u.

Case 2: The vertex v was encountered before the s^{th} iteration, and $i \leq (s-1)$ is the most recent such iteration. This means that there exists u' in the old graph (*i.e.*, $X_{j-1,s-1}$) such that $h_{j,s-1}(u') = v$. Since $h_{j,s-1}$ is a bijection by induction hypothesis, u' is the only preimage of v under $h_{j,s-1}$. We show that $u = u'$.

Observe that there is a vertex $w \in Des(g_s)$ in X_{j-1} such that $col(w) = col(I(s,j))$. By the algorithm, $col(u) | col(I(s,j))$. So, $u \in Des(w)$. Similarly, there is a vertex $w' \in Des(g_i)$ in X_{j-1} such that $col(w') = col(I(i,j))$ and by the algorithm $col(u') | col(I(i,j))$. So $u' \in Des(w')$. Since $col(u) = col(u')$, $col(u') | col(I(i,j))$ and $col(u) | col(I(s,j))$, we conclude that $col(u)$ divides $gcd(col(I(i,j)), col(I(s,j)))$. So, by Claim 37, $col(u) | col(I(i,s))$.

Now we consider the subgraph of X_{j-1} induced by $Des(g_i) \cap Des(g_s)$. If $x \in Des(g_i) \cap Des(g_s)$ is the vertex with color $col(I(i,s))$, then this subgraph is formed by the descendants of x. Since the descendants of x are exactly the vertices in $Des(g_i)$ and $Des(g_s)$ with colors as factors of $col(I(i,s))$, both u and u' are in $Des(x)$. Now, $Des(x)$ has a unique vertex of a particular color. Therefore, as u and u' have the same color, $u = u'$.

[13] Note that when $u \in X_{j-1,j-1}$ is identified with $v \in Y_{j,j-1}$, we color it with $col(u)$ and for simplicity we name the new vertex as u.

[14] Since $X_{j-1} \cong R_3(j-1)$, the concept of descendant reachability can also be defined in X_{j-1}. Therefore, it makes sense to use $Des(g)$ in X_{j-1} for any vertex g.

Hence, we have proved that $h_{j,s}$ is one-one in both the cases. Therefore, we can conclude that $h_{j,s} : X_{j-1,s} \to Y_{j,s}$ is a bijection for all $1 \leq s \leq j-1$. ◁

From the above claim, we can conclude that in the j^{th} iteration of the outer while loop, the identification process done in Line 12 in Algorithm 3 via the mapping $h_{j,j-1}$ is correct. Next, we show that the graph X_j (output in Line 13), derived after the identification process on X_{j-1} and Y_j, is indeed isomorphic to $R_3(j)$.

For $j \geq 2$, we define $f_j : V(X_j) \to V(R_3(j))$ in the following manner:

$$f_j(x) = \begin{cases} f_{j-1}(x) & \text{if } x \in V(X_{j-1}) \\ y & \text{otherwise where } y \in V(R_3(j)) \setminus V(R_3(j-1)) \\ & \text{and } col(y) = col(x). \end{cases} \tag{1}$$

To show that f_j is well defined, it is enough to argue that for each $x \in V(X_j) \setminus V(X_{j-1})$, there exists a unique $y \in V(R_3(j)) \setminus V(R_3(j-1))$ such that $col(y) = col(x)$. Observe that $V(X_j) \setminus V(X_{j-1})$ is the set of vertices of $Y_j = HD[col(g_j)]$ that have not been identified in the j^{th} iteration. So, for any vertex $x \in V(X_j) \setminus V(X_{j-1})$, $col(x)$ divides $col(g_j)$ but $col(x)$ does not divide $col(I(i,j))$ for any $i < j$. This means, for each such x, there exists y in $V(R_3(j)) \setminus V(R_3(j-1))$ with color $col(x)$ and this y is unique since $V(R_3(j)) \setminus V(R_3(j-1))$ contains the vertices of $Des(g_j)$ that are not descendant reachable from any g_i where $i < j$. The uniqueness of colors in $Y_j = HD[col(g_j)]$ also implies that f_j is a bijection.

Now to show that f_j is an isomorphism between X_j and $R_3(j)$, it remains to show that f_j preserves edge relations between X_j and $R_3(j)$.

Here, we want to emphasize that it might happen that two vertices x, x' in X_{j-1} are not adjacent to each other, but after the vertex identification process in the j^{th} iteration, there is an edge between x and x' in X_j. Moreover, through the following claim, we want to show that this incident has a correspondence in $R_3(j)$.

▷ Claim 41. Let x, x' be two vertices in the old graph (i.e., X_{j-1}) that take part in the vertex identification process in the j^{th} iteration, i.e., $x, x' \in X_{j-1,j-1}$. Then, $\{x, x'\} \notin E(X_{j-1})$, but $\{x, x'\} \in E(X_j)$ if and only if $\{f_{j-1}(x), f_{j-1}(x')\} \notin E(R_3(j-1))$, but $\{f_j(x), f_j(x')\} \in E(R_3(j))$.

Proof of claim. As f_{j-1} is an isomorphism between X_{j-1} and $R_3(j-1)$, we have $\{x, x'\} \notin E(X_{j-1})$ if and only if $\{f_j(x), f_j(x')\} \notin E(R_3(j-1))$.

Now, assume that $\{x, x'\} \notin E(X_{j-1})$ but $\{x, x'\} \in E(X_j)$. Since $x, x' \in X_{j-1,j-1}$, the vertices x, x' get identified with some elements z, z' respectively in Y_j such that $\{z, z'\} \in E(Y_j)$. Also, $col(x) = col(z) = col(f_j(x))$ and $col(x') = col(z') = col(f_j(x'))$. Since $Y_j = HD[col(g_j)]$ and $\{z, z'\} \in E(Y_j)$, by definition either $col(z) = col(z') \cdot p$ or $col(z') = col(z) \cdot p$ for some prime p. Therefore, either $col(f_j(x)) = col(f_j(x')) \cdot p$ or $col(f_j(x')) = col(f_j(x)) \cdot p$ for some prime p. Moreover, $f_j(x), f_j(x') \in Des(g_j)$. Hence, by (2) of Remark 36, $\{f_j(x), f_j(x')\} \in E(R_3(j))$.

Conversely, assume that $\{f_j(x), f_j(x')\} \in E(R_3(j))$. Since $x, x' \in X_{j-1,j-1}$, x and x' must have been identified with some vertices z and z' in Y_j respectively such that $col(x) = col(z)$ and $col(x') = col(z')$. Now, because of (2) of Remark 36, $\{f_j(x), f_j(x')\} \in E(R_3(j))$ implies either $col(f_j(x)) = col(f_j(x')) \cdot p$ or $col(f_j(x')) = col(f_j(x)) \cdot p$ for some prime p. Therefore, either $col(z) = col(z') \cdot p$ or $col(z') = col(z) \cdot p$. So, $\{z, z'\} \in E(Y_j)$. Hence, after the vertex identification, $\{x, x'\} \in E(X_3(j))$. ◁

Now to show the preservation of edge relations, we consider the following cases, not necessarily disjoint:

(a) Let $x, x' \in V(X_{j-1})$, i.e., both the vertices are from the graph obtained in the previous iteration of the outer while loop. Then, by definition of f_j in 1, $f_j(x) = f_{j-1}(x)$ and $f_j(x') = f_{j-1}(x')$. Since by induction hypothesis f_{j-1} is an isomorphism between X_{j-1} and $R_3(j-1)$, $\{x, x'\} \in E(X_{j-1}) \iff \{f_{j-1}(x), f_{j-1}(x')\} \in E(R_3(j-1))$. The remaining case is covered by Claim 41.

(b) Let x, x' be two vertices in X_j that appear in the "Y_j-part" of X_j. More precisely, x, x' belong to the disjoint union of $V(X_j) \setminus V(X_{j-1})$ (which is the set of vertices which are newly introduced in the j^{th} iteration of the outer while loop but not identified in the same) and $X_{j-1,j-1}$ (which corresponds to the set of vertices that are the result of vertex identification of $X_{j-1,j-1}$ and $Y_{j,j-1}$ in the j^{th} iteration). Since $Y_j = HD[col(g_j)] \cong R_3(G)[Des(g_j)]$ by Remark 36, $\{x, x'\} \in E(X_j) \iff \{f_j(x), f_j(x')\} \in E(R_3(j))$.

(c) Let x be a vertex from the old graph X_{j-1} which has not been identified in the j^{th} iteration, i.e., $x \in V(X_{j-1}) \setminus X_{j-1,j-1}$. Let x' be a newly added vertex that has not been identified in the j^{th} iteration, i.e., $x' \in V(X_j) \setminus V(X_{j-1})$. It is not hard to see that $\{x, x'\}$ is not an edge of the disjoint union of X_{j-1} and Y_j (before the identification process). Since none of x and x' has taken part in the identification process in this iteration, we have $\{x, x'\} \notin E(X_j)$. Now as f_j is a bijection, we also have the following: $f_j(x) \in V(R_3(j-1)) \setminus Des(g_j)$ and $f_j(x') \in V(R_3(j)) \setminus V(R_3(j-1))$. Since $f_j(x)$ and $f_j(x')$ are not in same $Des(u)$ for any vertex u in $R_3(j)$, $\{f_j(x), f_j(x')\}$ is not an edge in $R_3(j)$. Thus, it is proved that f_j is an isomorphism between X_j and $R_3(j)$. So, we can conclude that $X_m \cong R_3(m)$. It is easy to see that $R_3(m)$ is $R_3(G)$. This concludes the proof of Claim 39. ◁

Hence, the algorithm is correct, and we can return an isomorphic copy of $R_3(G)$ from $R_4(G)$. ◀