



ALMA MATER STUDIORUM  
UNIVERSITÀ DI BOLOGNA

ARCHIVIO ISTITUZIONALE  
DELLA RICERCA

## Alma Mater Studiorum Università di Bologna Archivio istituzionale della ricerca

Direct product primality testing of graphs is GI-hard

This is the final peer-reviewed author's accepted manuscript (postprint) of the following publication:

*Published Version:*

Direct product primality testing of graphs is GI-hard / Calderoni, Luca; Margara, Luciano; Marzolla, Moreno.  
- In: THEORETICAL COMPUTER SCIENCE. - ISSN 0304-3975. - ELETTRONICO. - 860:(2021), pp. 72-83.  
[10.1016/j.tcs.2021.01.029]

*Availability:*

This version is available at: <https://hdl.handle.net/11585/801042> since: 2021-02-18

*Published:*

DOI: <http://doi.org/10.1016/j.tcs.2021.01.029>

*Terms of use:*

Some rights reserved. The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

This item was downloaded from IRIS Università di Bologna (<https://cris.unibo.it/>).  
When citing, please refer to the published version.

(Article begins on next page)

This is the final peer-reviewed accepted manuscript of:

Calderoni, L., L. Margara, and M. Marzolla. "Direct Product Primality Testing of Graphs is GI-Hard." *Theoretical Computer Science*, vol. 860, 2021.

The final published version is available online at: <https://dx.doi.org/10.1016/j.tcs.2021.01.029>

#### Terms of use:

Some rights reserved. The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

This item was downloaded from IRIS Università di Bologna (<https://cris.unibo.it/>)

**When citing, please refer to the published version.**

# Direct Product Primality Testing of Graphs is GI-hard

Luca Calderoni<sup>a,\*</sup>, Luciano Margara<sup>a</sup>, Moreno Marzolla<sup>a</sup>

<sup>a</sup>*Department of Computer Science and Engineering, University of Bologna, Italy*

---

## Abstract

We investigate the computational complexity of the graph primality testing problem with respect to the direct product (also known as Kronecker, cardinal or tensor product). In [1] Imrich proves that both primality testing and a unique prime factorization can be determined in polynomial time for (finite) connected and nonbipartite graphs. The author states as an open problem how results on the direct product of nonbipartite, connected graphs extend to bipartite connected graphs and to disconnected ones. In this paper we partially answer this question by proving that the graph isomorphism problem is polynomial-time many-one reducible to the graph compositeness testing problem (the complement of the graph primality testing problem). As a consequence of this result, we prove that the graph isomorphism problem is polynomial-time Turing reducible to the primality testing problem. Our results show that connectedness plays a crucial role in determining the computational complexity of the graph primality testing problem.

*Keywords:* Kronecker product, graphs factorization, graphs isomorphism, GI complexity

---

## 1. Introduction

Factorization is a fundamental task in mathematics and in many other disciplines including computer science, physics and engineering. The notion of product among mathematical objects not only enables the creation of new objects from smaller ones, but also naturally addresses the more complex task of decomposing an object as the product of simpler components. Factoring a mathematical object is therefore one of the the main methods for understanding its structure.

Integer factorization is by far the most widely known and studied factorization problem; however, many other types of mathematical objects have been extensively studied in order to understand if and how they can be factored. Specifically, graph factorization with respect to several notions of product has been thoroughly investigated both from the theoretical and from the practical point of view.

In this paper we investigate the computational complexity of graph factorization with respect to the direct product (see Definition 2.2) which is one of the most widely studied graph product. Some authors refer to the direct product as the Kronecker, tensor or cardinal product. We will name it direct product and we will denote it by the operator  $\times$  according to the notation used in the recent book by Hammack, Imrich and Klavžar [2].

---

\*Corresponding author

*Email addresses:* `luca.calderoni@unibo.it` (Luca Calderoni), `luciano.margara@unibo.it` (Luciano Margara), `moreno.marzolla@unibo.it` (Moreno Marzolla)

*Preprint submitted to Theoretical Computer Science*

*January 4, 2021*

Direct product is one of the three products (the other two being the Cartesian and Strong products) that satisfies the following fundamental algebraic properties ( $\simeq$  stands for "isomorphic to"):

1. Commutativity:  $G_1 \times G_2 \simeq G_2 \times G_1$
2. Associativity:  $G_1 \times (G_2 \times G_3) \simeq (G_1 \times G_2) \times G_3$
3. Projections from a product to its factors are weak homomorphisms

A fourth product have been considered in the literature, namely the lexicographic product. Lexicographic product does not satisfy properties 1 and 3.

We both consider the primality testing problem and the factorization problem. Informally, primality testing is a decision problem that, given a graph  $G$ , answers the question: "is  $G$  the product of smaller, nontrivial graphs?". Graph factorization aims at decomposing  $G$  into the product of smaller nontrivial graphs (more formal definitions will be given in the next section). Although factorization of general with respect to the direct product is not unique, Imrich [1] proved that if a graph is connected and nonbipartite, then its factorization with respect to the direct product is unique and can be computed in polynomial time. In this paper we address the following question posed by Imrich at the end of his paper.

*How do results on the cardinal product of nonbipartite, connected graphs extend to bipartite connected graphs and to disconnected ones ?*

We prove (Theorem 4.12) that the graph isomorphism problem reduces to the problem of testing the compositeness of possibly unconnected, nonbipartite graphs. Since the reduction we use is a polynomial time many-one reduction, we show (Corollary 4.13) that testing the primality of a graph is  $GI$ -hard. In other words, we prove that testing the primality of a graph in polynomial time would provide a polynomial time algorithm for testing graph isomorphism, which is widely considered to be not feasible, although no formal proof exists. It remains an open question whether testing primality of bipartite, connected graphs can be done in polynomial time

This paper is organized as follows. In section 2 we introduce the notation and definition of terms used in the rest of this work. In section 3 we review the relevant literature related to the graph factorization problem. Section 4 presents the main result of this paper. Finally, conclusions and future research directions are discussed in section 5.

## 2. Notation and Basic Definitions

In this section we give basic notation and definitions that will be used throughout the paper. An undirected graph  $G = (V, E)$  is described as a finite set  $V$  of nodes  $V = \{v_1, \dots, v_n\}$  and a finite set of edges  $E \subseteq \{\{u, v\} \mid u, v \in V\}$ , where an edge  $e \in E$  is a multiset  $e = \{u, v\}$ ,  $u, v \in V$ ; an edge of the form  $\{v, v\}$  denotes a *self-loop*. Given a graph  $G$ ,  $V(G)$  and  $E(G)$  denote the set of nodes and edges of  $G$ , respectively. We denote by  $G_1 \cup G_2$  the disjoint union of graphs  $G_1$  and  $G_2$ , i.e., the graph with node set  $V(G_1) \cup V(G_2)$  and edge set  $E(G_1) \cup E(G_2)$ . Disjoint means that  $V(G_1)$  and  $V(G_2)$  satisfy  $V(G_1) \cap V(G_2) = \emptyset$ .

The set of edges of a graph  $G$  can be represented also as an adjacency matrix  $\mathbf{M}$ . If  $G$  has  $n$  nodes,  $\mathbf{M}$  is a  $n \times n$  binary matrix, such that  $\mathbf{M}_{ij} = 1$  if and only if  $\{v_i, v_j\} \in E$ . The adjacency matrix for undirected graphs is symmetric, since every edge  $\{v_i, v_j\}$  can also be written as  $\{v_j, v_i\}$ . As a shorthand notation, we denote with  $\mathbf{Adj}(G)$  the adjacency matrix of graph  $G$ .

We use the symbol  $\Gamma$  to denote the set of finite, undirected graphs where no self-loops are allowed. The symbol  $\Gamma_0$  denotes the set of finite, undirected graphs where self-loops are allowed.

An undirected graph  $G$  is *connected* iff there exists a path connecting any two nodes  $u, v \in V(G)$ .  $G$  is *bipartite* iff the set of nodes  $V(G)$  can be partitioned into two disjoint subsets  $V_1, V_2$  such that each edge  $e \in E(G)$  has one endpoint in  $V_1$  and the other in  $V_2$ . We will denote with  $C, B$  the set of connected and bipartite graphs, respectively.

Four types of graph products have been investigated in the literature: *Cartesian product*, *Direct product*, *Strong product* and *Lexicographic product*. In all cases, the product of two graphs  $G_1, G_2$  is a new graph  $G$  whose set of nodes is the Cartesian product of  $V(G_1)$  and  $V(G_2)$ :

$$V(G) = V(G_1) \times V(G_2) = \{\{u, v\} \mid u \in V(G_1) \wedge v \in V(G_2)\}$$

The edge set  $E(G)$  is defined according to the notion of graph product as follows.

**Definition 2.1** (Cartesian product). *The Cartesian product of two graphs  $G_1, G_2$  is denoted as  $G = G_1 \square G_2$ , where  $V(G) = V(G_1) \times V(G_2)$  and*

$$E(G) = \{\{\{x, y\}, \{x', y'\}\} \mid (x = x' \wedge \{y, y'\} \in E(G_2)) \vee (\{x, x'\} \in E(G_1) \wedge y = y')\}$$

**Definition 2.2** (Direct product). *The direct product of two graphs  $G_1, G_2$  is denoted as  $G = G_1 \times G_2$ , where  $V(G) = V(G_1) \times V(G_2)$  and*

$$E(G) = \{\{\{x, y\}, \{x', y'\}\} \mid \{x, x'\} \in E(G_1) \wedge \{y, y'\} \in E(G_2)\}$$

*The direct product is also known as Kronecker or cardinal product.*

**Definition 2.3** (Strong product). *The strong product of two graphs  $G_1, G_2$  is denoted as  $G = G_1 \boxtimes G_2$ , where  $V(G) = V(G_1) \times V(G_2)$  and*

$$E(G) = E(G_1 \square G_2) \cup E(G_1 \times G_2)$$

**Definition 2.4** (Lexicographic product). *The lexicographic product of two graphs  $G_1, G_2$  is denoted as  $G = G_1 \circ G_2$ , where  $V(G) = V(G_1) \times V(G_2)$  and*

$$E(G) = \{\{\{x, y\}, \{x', y'\}\} \mid \{x, x'\} \in E(G_1) \vee (x = x' \wedge \{y, y'\} \in E(G_2))\}$$

Figure 1 shows the Cartesian, direct, strong and lexicographic product of two graphs  $G_1, G_2$ .

A nontrivial graph  $G \in \Gamma_0$  is a graph with more than one node ( $|V(G)| > 1$ ). We say that a graph  $G$  is *prime* according to a given graph product  $\odot$  if  $G$  is nontrivial and  $G = G_1 \odot G_2$  implies that either  $G_1$  or  $G_2$  are trivial, i.e., one of them has exactly one node.

It has been shown [3, 2] that the adjacency matrix  $\mathbf{Adj}(G_1 \times G_2)$  of the direct product of  $G_1$  and  $G_2$  can be specified in terms of the *Kronecker product* of  $\mathbf{Adj}(G_1)$  and  $\mathbf{Adj}(G_2)$  (see Lemma 4.6). Given a  $n \times m$  matrix  $\mathbf{A}$  and a  $p \times q$  matrix  $\mathbf{B}$ , the Kronecker product  $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$  is a  $np \times mq$  matrix obtained from the scalar multiplication between each element of  $\mathbf{A}$  and the whole matrix  $\mathbf{B}$ :

$$\mathbf{C} = \mathbf{A} \otimes \mathbf{B} = \begin{pmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \dots & a_{1m}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \dots & a_{2m}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}\mathbf{B} & a_{n2}\mathbf{B} & \dots & a_{nm}\mathbf{B} \end{pmatrix} \quad (1)$$

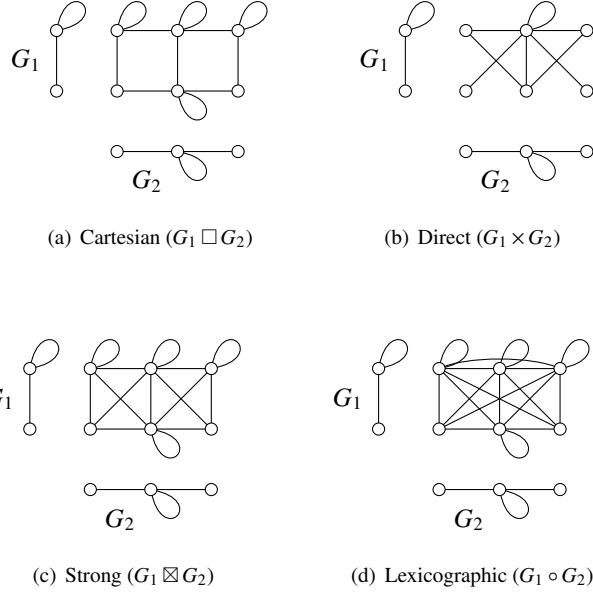


Figure 1: Example of the different types of graph products.

We finally recall the definition of *many-one reducibility* and *Turing reducibility*. Given two sets  $S_1, S_2 \subseteq \mathbb{N}$ , we say that  $S_1$  is many-one reducible to  $S_2$ , if there exists a total computable function  $f : \mathbb{N} \rightarrow \mathbb{N}$  such that [4]

$$n \in S_1 \iff f(n) \in S_2$$

A polynomial time many-one reduction (denoted by  $\leq_M$ ) is a many-one reduction with the additional constraint that  $f$  is computable in polynomial time.

Turing reducibility is a weaker form of many-one reducibility. Informally,  $S_1$  is Turing reducible to  $S_2$  if there exists an oracle for testing membership in  $S_1$  relying on another oracle for testing membership in  $S_2$  [5]. In other words,  $S_1$  is Turing reducible to  $S_2$  if it is possible to answer the question “is  $n \in S_1$ ” given the existence of an effective procedure for answering the question “is  $m \in S_2$ ” for any  $m \in \mathbb{N}$  [4].

$S_1$  is polynomial-time Turing reducible to  $S_2$  (denoted by  $S_1 \leq_T S_2$ ) if  $S_1$  is Turing reducible to  $S_2$ , with the following two additional constraints:

1. the oracle for testing membership in  $S_1$  makes at most a polynomial number of calls to the oracle for testing membership in  $S_2$  and
2. the overall computational cost of the oracle for testing membership in  $S_1$  (excluding the calls to the oracle for testing membership in  $S_2$ ) is polynomially bounded.

As a final consideration, throughout the paper we intend graphs to be *finite* and *undirected*, unless otherwise specified. Table 1 summarizes the notation used in this paper.

Symbol	Description
$\Gamma$	The set of finite, undirected graphs, without self-loops
$\Gamma_0$	The set of finite, undirected graphs, self-loops allowed
$C, \bar{C}$	The set of connected/unconnected, undirected graphs
$B, \bar{B}$	The set of bipartite/nonbipartite, undirected graphs
$\text{Adj}(G)$	The adjacency matrix of graph $G$
$\mathbf{I}_n$	The $n \times n$ identity matrix
$\mathbf{0}_n$	The $n \times n$ zero matrix
$\times$	The direct graph product operator
$\square$	The Cartesian graph product operator
$\boxtimes$	The strong graph product operator
$\circ$	The lexicographic graph product operator
$\otimes$	The Kronecker matrix product operator
$\simeq$	The graphs isomorphism operator
$\leq_M$	Polynomial many-one reducibility
$\leq_T$	Polynomial Turing reducibility
$\cup$	Disjoint union of graphs

Table 1: Summary of notation.

### 3. Related works

In this section we list the main results on graph factorization that are strictly related to the work presented in this paper. The interested reader may find a comprehensive review of the theory of graph factorization in the recent book by Hammack, Imrich and Klavžar [2].

*Direct product.* Prime factorization of connected, nonbipartite graphs in  $\Gamma_0$  is unique up to isomorphism and the order of the factors, and can be computed in polynomial time [1].

*Cartesian product.* Prime factorization of connected graphs is unique up to isomorphism and the order of the factors [6, 7]. Prime factorization is not unique in the class of possibly disconnected simple graphs. Following Sabidussi’s approach, Feigenbaum et al. [8] derived a polynomial-time algorithm that computes the prime factors of a connected graph. A different polynomial-time algorithm for connected graphs has been independently discovered by Winkler [9].

*Strong product.* Prime factorization of connected graphs is unique up to reordering and isomorphism of factors and it can be computed in polynomial time [1].

*Lexicographic product.* Determining whether a connected graph is prime is at least as difficult as the graph isomorphism problem [10].

An interesting observation relating graph factorization and graph isomorphism problem can be found at the end of page of [2, p. 229]. The authors claim that, if  $X$  is the disjoint union of graphs  $G$  and  $H$ , then  $G \simeq H$  if and only if  $X = D_2 \square G = D_2 \boxtimes G = D_2 \circ G$  where  $D_2$  denotes the graph with two nodes and two self-loops. They conclude that “testing whether a disconnected graph is decomposable with respect to any of these three products is at least as hard as the

Graph type	Product type			
	Direct	Cartesian	Strong	Lexicographic
Connected and nonbipartite	<b>P</b> [1]	P [8, 9]	P [11]	•
Connected	•	<b>P</b> [8, 9]	<b>P</b> [11]	<b>GI-Hard</b> [10]
Unconnected and nonbipartite	<b>GI-Hard</b> (our results)	•	•	•
Nonbipartite	<b>GI-Hard</b> (our results)	•	•	•

Table 2: Complexity of the graph factorization problem for different types of graphs considered in the literature (connected, unconnected, nonbipartite) and different notions of graph product (direct, Cartesian, strong and lexicographic product); **P** stands for polynomially time solvable. Table cells reported in light gray can be easily inferred from another cells in the same column depending on the relation between the corresponding classes of graphs. For instance, a polynomial-time solvable problem for connected graphs is polynomial-time solvable within a restricted class of graphs (e.g., connected and nonbipartite). Dots denote the cases that, to our knowledge, have not yet been explored.

graph isomorphism problem”. They do not give a formal proof of their claim and in particular they do not explain how they get rid of the case in which  $X$  is the disjoint union of two non isomorphic graphs  $G_1$  and  $G_2$  and, at the same time,  $X$  admits as a factor a graph with two nodes that is not isomorphic to  $D_2$ . In that case the decomposability test would lead to erroneously declare  $G_1 \simeq G_2$ . Moreover, if  $X$  admits more than one factorization, also computing a single factorization could not be enough for testing isomorphism.

In Section 4 we show (see Figures 3 and 4) that when the direct product is used, both these cases may occur.

#### 4. Main Results

In this section we prove that testing whether two graphs  $G_1, G_2$  are isomorphic is not harder than testing whether an undirected graph  $G \in \Gamma_0$  is  $\times$ -composite, i.e.,  $G$  admits nontrivial factors with respect to the direct product decomposition. More formally, we show that graph isomorphism problem is polynomially many-one reducible to the problem of testing  $\times$ -compositeness of graphs.

Before starting, we formally define the following three decision problems in terms of their admissible inputs and related outputs.

**Definition 4.1 (GI[S]).** Let  $S$  be any set of graphs. **GI[S]** is defined as follows.

*Input:*  $G_1, G_2 \in S$

*Output:* YES if  $G_1$  is isomorphic to  $G_2$ , NO otherwise.

**Definition 4.2 (Primality[S]).** Let  $S$  be any set of graphs. **Primality[S]** is defined as follows.

*Input:*  $G \in S$

*Output:* YES if  $G$  is prime with respect to the direct product, NO otherwise.

**Definition 4.3 (Compositeness[S]).** Let  $S$  be any set of graphs. **Compositeness[S]** is defined as follows.

*Input:*  $G \in S$

*Output:* YES if  $G$  is composite with respect to the direct product, NO otherwise.

**Definition 4.4 (GI-hard problem).** A decision problem  $\mathcal{P}$  is GI-hard if and only if

$$GI[\text{general graphs}] \leq_T \mathcal{P}$$



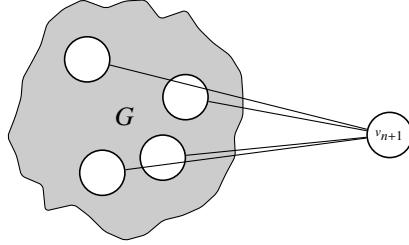


Figure 2: Construction of  $\text{Ext}(G)$

We also prove the following

**Lemma 4.5.** *Graph isomorphism on general graphs is polynomial-time Turing-reducible to graph isomorphism on connected graphs:*

$$\mathbf{GI}[\text{general graphs}] \leq_{\mathbf{T}} \mathbf{GI}[C]$$

*Proof.* We show that, if we have an oracle that can decide whether two connected graphs  $G_1, G_2 \in C$  are isomorphic, we can decide whether two general (i.e., not necessarily connected) graphs are isomorphic by calling the oracle a polynomial number of times, plus a polynomially bounded amount of additional work

Given a general, undirected graph  $G$  with  $n$  nodes, we define a new graph  $\text{Ext}(G)$  as:

$$\begin{aligned} V(\text{Ext}(G)) &:= V(G) \cup v_{n+1} \\ E(\text{Ext}(G)) &:= E(G) \cup \{v_1, v_{n+1}\} \cup \dots \cup \{v_n, v_{n+1}\} \end{aligned}$$

where  $v_1, \dots, v_n$  are the nodes of  $G$  and  $v_{n+1}$  is a new node. Basically,  $\text{Ext}(G)$  is obtained by extending  $G$  with a new node  $v_{n+1}$  that is connected to all other nodes (see Figure 2).  $\text{Ext}(G)$  can be built in polynomial time and is connected by construction.

If two general graphs  $G_1$  and  $G_2$  are isomorphic, then it immediately follows that  $\text{Ext}(G_1)$  and  $\text{Ext}(G_2)$  are isomorphic as well.

Conversely, let us assume that  $\text{Ext}(G_1)$  and  $\text{Ext}(G_2)$  are isomorphic. This implies that  $G_1$  and  $G_2$  have the same number of nodes  $n$ , and both  $\text{Ext}(G_1)$  and  $\text{Ext}(G_2)$  have  $(n + 1)$  nodes. By construction,  $\text{Ext}(G_1)$  and  $\text{Ext}(G_2)$  have exactly one node of degree  $n$ , namely, the new node(s)  $v_{n+1}$ . Therefore, every isomorphism between  $\text{Ext}(G_1)$  and  $\text{Ext}(G_2)$  must define a bijection that maps the new node of  $\text{Ext}(G_1)$  to the new node of  $\text{Ext}(G_2)$ . The bijection can be restricted to  $V(G_1)$  and  $V(G_2)$  of the original graphs, implying that they are isomorphic as well.  $\square$

From Definition 4.4 and Lemma 4.5 we get, by transitivity, that a decision problem  $P$  is  $\mathbf{GI}$ -hard if and only if

$$\mathbf{GI}[C] \leq_{\mathbf{T}} P$$

The following observation highlights the strong relation between the direct product of graphs and the Kronecker product of their adjacency matrices.

**Lemma 4.6** ([3, 2]). *Let  $G_1$  and  $G_2$  be graphs. Then*

$$\mathbf{Adj}(G_1 \times G_2) = \mathbf{P}^\top (\mathbf{Adj}(G_1) \otimes \mathbf{Adj}(G_2)) \mathbf{P},$$

where  $\mathbf{P}$  is a suitable permutation matrix.

*Proof.* This lemma can be proved with the following observation [2]. Eq. 1 shows that the Kronecker product of two matrices  $\mathbf{A}$  and  $\mathbf{B}$  has a block structure so that the rows of  $\mathbf{A} \otimes \mathbf{B}$  can be indexed by an ordered pair  $(i, j)$  that corresponds to the  $j$ -th row of  $\mathbf{B}$  in the  $i$ -th row block. If we let  $V(G_1) = \{g_1, \dots, g_m\}$  and  $V(G_2) = \{h_1, \dots, h_n\}$ , then  $G_1 \otimes G_2$  has adjacency matrix  $\mathbf{Adj}(G_1) \otimes \mathbf{Adj}(G_2)$  with the node ordering  $(g_1, h_1), \dots, (g_1, h_n), (g_2, h_1), \dots, (g_2, h_n), \dots, (g_m, h_1), \dots, (g_m, h_n)$ . The use of a permutation matrix  $\mathbf{P}$  makes the result independent from the node ordering.  $\square$

In the following lemma we prove that two graphs  $G_1$  and  $G_2$  are isomorphic if and only if there exists a permutation matrix  $\mathbf{P}$  that transforms the adjacency matrix of the disjoint union of  $G_1$  and  $G_2$  into the Kronecker product of the identity matrix  $\mathbf{I}_2$  and a suitable binary matrix  $\mathbf{B}$ .

**Lemma 4.7.** *Let  $G_1, G_2 \in \mathcal{C}$  be undirected, connected graphs with  $n$  nodes. Let  $\mathbf{M}_1 = \mathbf{Adj}(G_1)$  and  $\mathbf{M}_2 = \mathbf{Adj}(G_2)$ . Let  $\mathbf{M}$  denote the adjacency matrix of the disjoint union  $G = G_1 \cup G_2$ . Without loss of generality, we may write  $\mathbf{M}$  as*

$$\mathbf{M} = \begin{pmatrix} \mathbf{M}_1 & \mathbf{0}_n \\ \mathbf{0}_n & \mathbf{M}_2 \end{pmatrix} \quad (2)$$

Then,  $G_1$  and  $G_2$  are isomorphic ( $G_1 \simeq G_2$ ) if and only if there exists a  $2n \times 2n$  permutation matrix  $\mathbf{P}$  and a  $n \times n$  binary matrix  $\mathbf{B}$  such that

$$\mathbf{P}^\top \mathbf{M} \mathbf{P} = \mathbf{I}_2 \otimes \mathbf{B} \quad (3)$$

where  $\mathbf{I}_2$  denotes the  $2 \times 2$  identity matrix.

*Proof.* ( $\implies$ ) We first prove that if  $G_1$  and  $G_2$  are isomorphic, then Eq. (3) holds. If  $G_1 \simeq G_2$  then there exists a  $n \times n$  permutation matrix  $\mathbf{Q}$  that transforms the adjacency matrix  $\mathbf{M}_2$  of  $G_2$  in the adjacency matrix  $\mathbf{M}_1$  of  $G_1$ :

$$\mathbf{M}_1 = \mathbf{Q}^\top \mathbf{M}_2 \mathbf{Q} \quad (4)$$

Let us define

$$\mathbf{P} = \begin{pmatrix} \mathbf{I}_n & \mathbf{0}_n \\ \mathbf{0}_n & \mathbf{Q} \end{pmatrix} \quad (5)$$

It follows that

$$\begin{aligned} \mathbf{P}^\top \mathbf{M} \mathbf{P} &= \begin{pmatrix} \mathbf{I}_n & \mathbf{0}_n \\ \mathbf{0}_n & \mathbf{Q}^\top \end{pmatrix} \begin{pmatrix} \mathbf{M}_1 & \mathbf{0}_n \\ \mathbf{0}_n & \mathbf{M}_2 \end{pmatrix} \begin{pmatrix} \mathbf{I}_n & \mathbf{0}_n \\ \mathbf{0}_n & \mathbf{Q} \end{pmatrix} && \text{by (5) and (2)} \\ &= \begin{pmatrix} \mathbf{M}_1 & \mathbf{0}_n \\ \mathbf{0}_n & \mathbf{Q}^\top \mathbf{M}_2 \mathbf{Q} \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{M}_1 & \mathbf{0}_n \\ \mathbf{0}_n & \mathbf{M}_1 \end{pmatrix} && \text{by (4)} \\ &= \mathbf{I}_2 \otimes \mathbf{M}_1 \end{aligned}$$

( $\Leftarrow$ ) We prove that if Eq. (3) holds, then  $G_1, G_2$  are isomorphic.

Observe that the transformation  $\mathbf{P}^\top \mathbf{M} \mathbf{P}$  consists of relabeling the nodes of  $G$  according to the permutation matrix  $\mathbf{P}$ . Let us define this relabeling as the bijection  $\pi : \{1, 2, \dots, 2n\} \rightarrow \{1, 2, \dots, 2n\}$ . Therefore,  $\pi(i) = j$  if and only if the node  $i$  of  $G$  is relabeled as  $j$ . Note also that  $\mathbf{P}^\top \mathbf{M} \mathbf{P}$  is symmetric, since it represents the adjacency matrix of an undirected graph.

Since we are assuming that  $G_1$  is connected, then there always exists a path from node  $i$  to node  $j$ ,  $1 \leq i, j \leq n$ . Thus, should the permutation contain a mapping such that  $\pi(i) \leq n$  and  $\pi(j) > n$ , the relabeled adjacency matrix  $\mathbf{P}^\top \mathbf{M} \mathbf{P}$  would contain at least one 1 in the upper-right quadrant and (by symmetry) in the lower-left one. However, this contradicts the hypothesis (3), since the upper right and lower left quadrants of  $\mathbf{I}_2 \otimes \mathbf{B}$  are the zero matrix  $\mathbf{0}_n$ . The same considerations apply to  $G_2$ .

Thus, from Eq. (3) we observe that  $\pi$  maps the sets  $\{1, 2, \dots, n\}$  and  $\{n+1, n+2, \dots, 2n\}$  into themselves, and therefore  $\mathbf{P}$  must have a block structure:

$$\mathbf{P} = \begin{pmatrix} \mathbf{P}_1 & \mathbf{0}_n \\ \mathbf{0}_n & \mathbf{P}_2 \end{pmatrix} \quad (6)$$

Let  $G_3$  be the undirected graph such that  $\text{Adj}(G_3) = \mathbf{B}$ . Combining (6) and (3) we can conclude that  $G_1 \simeq G_3$  and  $G_2 \simeq G_3$ , because the adjacency matrices  $\mathbf{M}_1, \mathbf{M}_2$  can be transformed into  $\mathbf{B}$  via the permutation matrices  $\mathbf{P}_1, \mathbf{P}_2$ , respectively. By transitivity we conclude  $G_1 \simeq G_2$ .  $\square$

Lemma 4.7 ensures that the adjacency matrix of the disjoint union of two isomorphic graphs may always be written as  $\mathbf{I}_2 \otimes \mathbf{B}$ ; note that  $\mathbf{I}_2$  is the adjacency matrix of  $D_2$ , the graph with two nodes and two self-loops. Unfortunately, simply testing primality of the disjoint union  $X = G_1 \cup G_2$  of two graphs  $G_1$  and  $G_2$  is not enough for deciding whether  $G_1$  and  $G_2$  are isomorphic or not. In fact, as mentioned at the end of Section 3, the graph  $X$  could admit as a factor a graph with two nodes different from  $D_2$ . For example, Figure 4 shows a graph  $X$  such that

- $X$  is the disjoint union of two non isomorphic graphs (connected and having the same number of nodes and edges) and
- $X$  admits as a factor a graph with two nodes that is different from  $D_2$ .

Moreover, the idea of factorizing  $X = G_1 \cup G_2$  to check whether  $D_2$  is a factor (or, equivalently,  $G_1 \simeq G_2$ ) might fail due to the fact that  $X$  could admit two different factorizations  $F_1$  and  $F_2$  where  $F_1$  contains  $D_2$  while  $F_2$  does not. Figure 3 shows an example of a graph  $X$  such that

- $X$  is the disjoint union of two isomorphic graphs (both connected and with a prime number of nodes)
- $X$  admits two distinct factorizations  $F_1$  and  $F_2$  where  $F_1$  contains  $D_2$  while  $F_2$  does not.

In order to prove the main result of this paper, we define a class of graphs  $\Theta$  as follows.

**Definition 4.8** (Class  $\Theta$ ). *A graph  $G = (V, E)$  with  $n$  nodes and  $m$  edges belongs to the class  $\Theta \subset \Gamma_0$  if and only if:*

**P1**  $G$  is undirected, connected and not bipartite ( $G \in C \cap \overline{B}$ );

**P2** The number of nodes  $n$  is prime;

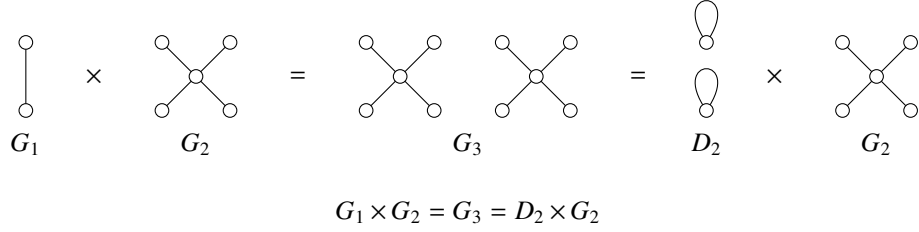


Figure 3:  $G_2$  is a connected graph with a prime number of nodes.  $G_3$  is the disjoint union of two copies of  $G_2$ .  $G_3$  has two different factorizations, namely,  $G_1 \times G_2$  and  $D_2 \times G_2$ .

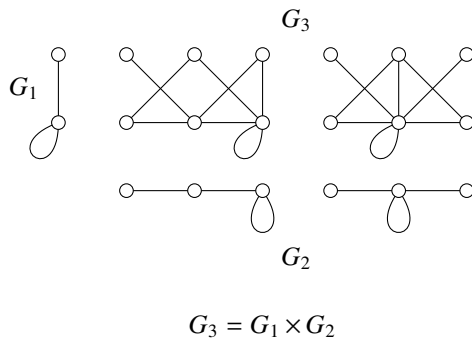


Figure 4:  $G_3$  is the disjoint union of two connected graphs with the same number of nodes and edges.  $G_3$  is the direct product of  $G_1$  and  $G_2$ .  $G_3$  has  $G_1$  as a factor, which differs from  $D_2$  by two nodes.  $G_3$  has not  $D_2$  as a factor.

**P3** The number  $s$  of self-loops is strictly less than the number of edges, i.e.,  $s < m$ ;

**P4**  $(2m - s)$  is not divisible by 2;

**P5**  $(2m - s)$  is not divisible by 3;

In the following theorem we give a polynomial time many-one reduction from **GI**[ $\Theta$ ] to **Compositeness**[ $\Theta$ ]. A consequence of this result is that the existence of a polynomial-time algorithm to determine whether a given graph in  $\Theta$  is composite with respect to the direct product would imply the existence of a polynomial-time algorithm for the graph isomorphism problem between any two graphs in  $\Theta$ .

**Theorem 4.9.**

$$GI[\Theta] \leq_M \text{Compositeness}[\Theta]$$

*Proof.* Let  $\mathcal{A}$  be an algorithm for testing compositeness for graphs in  $\Theta$ . The following algorithm solves the isomorphism problem for graphs in  $\Theta$  relying on a single call to  $\mathcal{A}$ , therefore providing a polynomial time many-one reduction from **GI**[ $\Theta$ ] to **Compositeness**[ $\Theta$ ].

$\Theta$ -GRAPH-ISOMORPHISM ( $G_1, G_2$ )

```

1  if  $|V(G_1)| \neq |V(G_2)|$  or  $|E(G_1)| \neq |E(G_2)|$ 
2      return NO
3   $G = G_1 \cup G_2$  // graphs disjoint union
4  return  $\mathcal{A}(G)$ 

```

Let us prove that  $\Theta$ -GRAPH-ISOMORPHISM is correct. To this end, we consider two cases:

*G is prime.* Let  $G_1, G_2 \in \Theta$  and  $\mathbf{M} = \mathbf{Adj}(G_1 \cup G_2)$ . According to Lemma 4.7 and 4.6, if  $G$  is prime we can conclude that  $G_1$  and  $G_2$  are not isomorphic, since if they were, there should exist a permutation matrix  $\mathbf{P}$  and a suitable adjacency matrix  $\mathbf{B}$  such that  $\mathbf{P}^T \mathbf{M} \mathbf{P} = \mathbf{I}_2 \otimes \mathbf{B}$  and then  $G_1 \cup G_2$  would be composite.

*G is composite.* Let  $G_1, G_2 \in \Theta$  and  $\mathbf{M} = \mathbf{Adj}(G_1 \cup G_2)$ . Let  $n = |V(G_1)| = |V(G_2)|$  be the number of nodes of either  $G_1$  or  $G_2$ . Since  $G_1 \in \Theta$ ,  $n$  is prime (**P2**). Consequently, the number of nodes of  $G = G_1 \cup G_2$  is  $2n$  and its adjacency matrix  $\mathbf{M}$  has size  $2n \times 2n$ . Therefore, the only possible factorization of  $\mathbf{M}$  is  $\mathbf{M} = \mathbf{A} \otimes \mathbf{B}$ , where  $\mathbf{A}$  has size  $2 \times 2$  and  $\mathbf{B}$  has size  $n \times n$ . Additionally, since  $G_1, G_2 \in \Theta$ , their adjacency matrices have exactly  $2m - s$  nonzero elements each, where  $m = |E(G_1)| = |E(G_2)|$  is the number of edges of either  $G_1$  or  $G_2$  and  $s$  is the number of self-loops. Let us consider each possible configuration of the matrix  $\mathbf{A}$ :

$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

Since  $G_1$  and  $G_2$  are undirected,  $G$  is undirected as well, so its adjacency matrix  $\mathbf{M}$  must be symmetric. From 4.7 we deduce that  $\mathbf{A}$  must be symmetric, since  $\mathbf{B}$  is a nonzero matrix. Therefore, we exclude all configurations of matrix  $\mathbf{A}$  that are not symmetric.

$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

Since  $G_1$  and  $G_2$  are connected, the degrees of all their nodes must be strictly greater than zero. Therefore,  $\mathbf{A}$  must contain more than a single nonzero element as, conversely, the resulting matrix  $\mathbf{M}$  would have at least  $n$  unconnected nodes with zero degree. Therefore, we exclude all configurations of  $\mathbf{A}$  that have less than two nonzero elements.

$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

Let us now consider the number  $b$  of nonzero elements in  $\mathbf{B}$ . Should  $\mathbf{A}$  have three nonzero elements, the resulting number of nonzero elements in  $\mathbf{M}$  would be  $3b$ , which is divisible by 3. Moreover, the number of nonzero elements of  $\mathbf{M}$  is equal to the number of nonzero elements of the adjacency matrices of  $G_1$  and  $G_2$ :

$$3b = 2(2m - s)$$

Since  $G_1, G_2 \in \Theta$ , we know that the number of nonzero elements  $(2m - s)$  in their adjacency matrices must not be divisible by 3 (**P5**). Thus,  $2(2m - s)$  must not be divisible by 3 either, contradicting (4). We conclude that  $\mathbf{A}$  can not contain three nonzero elements.

$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

Similarly, should the matrix  $\mathbf{A}$  have four nonzero elements, the number of nonzero elements in  $\mathbf{M}$  would be  $4b$ ; from the same reasoning above, we get:

$$4b = 2(2m - s)$$

However, from **P4** we have that  $(2m - s)$  is not divisible by 2, and therefore  $2(2m - s)$  is not divisible by 4. We conclude that  $\mathbf{A}$  can not have four nonzero elements.

$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

Finally, we point out that since  $G_1$  and  $G_2$  are not bipartite (**P1**),  $G$  is not bipartite as well. Should  $\mathbf{A} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ , matrix  $\mathbf{M} = \mathbf{A} \otimes \mathbf{B}$  would represent a bipartite graph, where the first  $n$  nodes are only connected to the other  $n$  nodes. Therefore,  $\mathbf{A}$  can not be in that form.

$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

We conclude that  $\mathbf{A} = \mathbf{I}_2$ . Thus, according to Lemma 4.7,  $G_1$  and  $G_2$  are isomorphic. □

Theorem 4.9 shows that, within the class  $\Theta$ , there exists an intimate relation between graph primality with respect to the direct product and graph isomorphism. In what follows we will extend Theorem 4.9 to the class of graph  $\Gamma_0$  by describing a polynomial-time, isomorphism-preserving transformation that maps any connected graph  $G \in C$  into a graph in  $\Theta$ . Before doing so, we need to prove a small technical lemma.

**Lemma 4.10.** *For each  $n \in \mathbb{Z}$  there exists  $d \in \{0, 1, 2, 3\}$  such that  $(n + d)$  is not divisible by two nor by three; formally,  $(n + d) \bmod 2 \neq 0$  and  $(n + d) \bmod 3 \neq 0$ .*

*Proof.* Let us denote with  $\llbracket n \rrbracket_k$  the equivalence class of all integers that are congruent to  $n$  modulo  $k$  (also called residual class):  $\llbracket n \rrbracket_k = \{\dots, n - 2k, n - k, n, n + k, n + 2k, \dots\}$ . The lemma can then be rephrased as: for each  $n \in \mathbb{Z}$  there exists  $d \in \{0, 1, 2, 3\}$  such that  $(n + d) \notin (\llbracket 0 \rrbracket_2 \cup \llbracket 0 \rrbracket_3)$ . The following table shows all possible values of  $d$  for any combination of residual classes modulo 2 and modulo 3 that  $n$  may belong to.

$n$	$\llbracket 0 \rrbracket_3$	$\llbracket 1 \rrbracket_3$	$\llbracket 2 \rrbracket_3$
$\llbracket 0 \rrbracket_2$	$d = 1$	$d = 1, 3$	$d = 3$
$\llbracket 1 \rrbracket_2$	$d = 2$	$d = 0$	$d = 0, 2$

For example, if  $n \in \llbracket 1 \rrbracket_2 \cap \llbracket 0 \rrbracket_3$ , then  $(n + 2) \in \llbracket 1 \rrbracket_2 \cap \llbracket 2 \rrbracket_3$ , which means that  $(n + 2)$  is not divisible by 2 nor by 3. To build the table, one might observe that:

$$\llbracket n + d \rrbracket_2 = \begin{cases} \llbracket n \rrbracket_2 & \text{if } d \in \{0, 2\} \\ \llbracket n + 1 \rrbracket_2 & \text{if } d \in \{1, 3\} \end{cases} \quad \llbracket n + d \rrbracket_3 = \begin{cases} \llbracket n \rrbracket_3 & \text{if } d \in \{0, 3\} \\ \llbracket n + 1 \rrbracket_3 & \text{if } d = 1 \\ \llbracket n + 2 \rrbracket_3 & \text{if } d = 2 \end{cases}$$

□

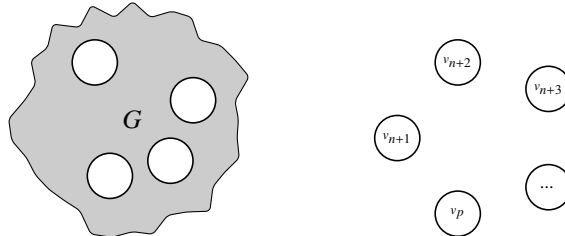
**Theorem 4.11.** *There exists a mapping  $f : C \rightarrow \Theta$  such that for every two connected graphs  $G_1, G_2 \in C$  with the same number of nodes and edges,  $G_1 \simeq G_2$  if and only if  $f(G_1) \simeq f(G_2)$ . Furthermore,  $f(G)$  can be computed in polynomial time with respect to the size of  $G$ .*

*Proof.* Given a connected graph  $G \in C$ , let us define  $G' = f(G)$  as follows. Let  $m = |E(G)|$  and  $n = |V(G)|$ . According to the Bertrand-Chebyshev theorem [12], for any integer  $q > 1$  there exists a prime in  $\{q + 1, \dots, 2q - 1\}$ , and such prime can be found in polynomial time [13]. Therefore, there is a prime  $p$  such that  $2n < p < 4n$ .

The vertex set of  $G'$  is defined as

$$V(G') := V(G) \cup \{v_{n+1}, v_{n+2}, \dots, v_p\}$$

where  $v_{n+1}, v_{n+2}, \dots, v_p$  are new nodes.

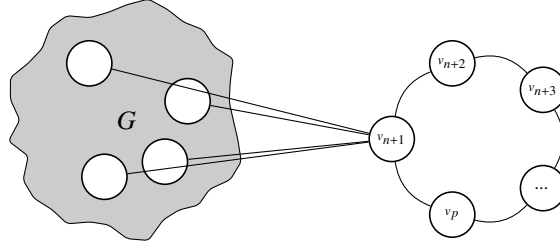


The edge set  $E(G')$  is constructed incrementally from  $E(G)$ , as follows. Let

$$C_f := \{\{x, v_{n+1}\} \mid x \in V(G)\}$$

$$C_c := \{\{v_{n+1}, v_{n+2}\}, \{v_{n+2}, v_{n+3}\}, \dots, \{v_p, v_{n+1}\}\}$$

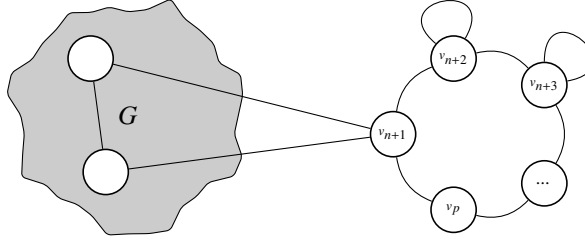
that is,  $C_f$  is a set of new edges that connect each node in  $V(G)$  to the first newly created node  $v_{n+1}$ , and  $C_c$  is a set of new edges that form a cycle within the new nodes. Since we have chosen  $p$  such that  $2n < p < 4n$ , the length of the cycle in  $C_c$  is greater than  $n$ .



We finally add a number  $s$  of self-loops within the new nodes in order to meet conditions **P4** and **P5**. Lemma 4.10 guarantees that  $s$  is at most 3. Thus:

$$C_s := \begin{cases} \emptyset & \text{if } s = 0 \\ \{\{v_{n+2}, v_{n+2}\}\} & \text{if } s = 1 \\ \{\{v_{n+2}, v_{n+2}\}, \{v_{n+3}, v_{n+3}\}\} & \text{if } s = 2 \\ \{\{v_{n+2}, v_{n+2}\}, \{v_{n+3}, v_{n+3}\}, \{v_{n+4}, v_{n+4}\}\} & \text{if } s = 3 \end{cases}$$

The edge set  $E(G')$  is therefore defined as  $E(G') := E(G) \cup C_f \cup C_c \cup C_s$ .



Observe that, since  $G$  is connected, the edge subset  $C_f$  induces at least one odd cycle (specifically, at least one cycle of length three), and therefore  $G'$  is not bipartite (a graph is bipartite if and only if it has no odd cycles [14]). Therefore we conclude that  $G' \in \Theta$ .

We now prove that  $G_1 \simeq G_2 \iff f(G_1) \simeq f(G_2)$ .

( $\implies$ ) Assume  $G_1 \simeq G_2$ . Then, the isomorphism can be trivially extended to  $f(G_1)$  and  $f(G_2)$  since these graphs are obtained from  $G_1, G_2$  by adding an identical structure.

( $\impliedby$ ) Assume  $f(G_1) \simeq f(G_2)$ . The only possible isomorphisms are those that map one of the cycles  $C_c$  to the corresponding one on the other graph. Since in the transformation we have chosen  $p > 2n$ , the cycle length of  $C_c$  is larger than  $n$ , and therefore is larger than any simple cycle in  $G_1$  (or  $G_2$ ). Consequently, the isomorphism between  $f(G_1)$  and  $f(G_2)$  can be restricted to an isomorphism between  $G_1$  and  $G_2$ .  $\square$



Theorem 4.11 allows us to assert the main result of this paper, that is the relation between primality test and graphs isomorphism.

**Theorem 4.12.**

$$GI[C] \leq_M \text{Compositeness}[\overline{C} \cap \overline{B}]$$

*Proof.* Assume that there exists an algorithm  $\mathcal{A}$  that solves the **Compositeness** $[\overline{C} \cap \overline{B}]$  decision problem. Specifically,  $\mathcal{A}(G)$  returns YES if the unconnected, nonbipartite graph  $G \in \overline{C} \cap \overline{B}$  is composite with respect to the direct product, NO otherwise. Then, the following algorithm solves the **GI** $[C]$  decision problem, providing a polynomial time many-one reduction from **GI** $[C]$  to **Compositeness** $[\overline{C} \cap \overline{B}]$ .

GRAPH-ISOMORPHISM( $G_1, G_2$ )

```

1  if  $|V(G_1)| \neq |V(G_2)|$  or  $|E(G_1)| \neq |E(G_2)|$ 
2      return NO
3   $G_3 = f(G_1)$  // Theorem 4.11
4   $G_4 = f(G_2)$  // Theorem 4.11
5   $G = G_3 \cup G_4$  // disjoint union of graphs
6  return  $\mathcal{A}(G)$ 

```

Indeed, by Theorem 4.11,  $G_1$  is isomorphic to  $G_2$  if and only if  $f(G_1)$  is isomorphic to  $f(G_2)$ . Since both  $f(G_1)$  and  $f(G_2)$  belong to  $\Theta$ , then by Theorem 4.9,  $G = G_3 \cup G_4$  is decomposable if and only if  $G_3$  is isomorphic to  $G_4$ .  $\square$

Note that **Compositeness** $[\overline{C} \cap \overline{B}]$  remains *GI*-hard even if we relax either the *undirected* or the *nonbipartite* constraints, as the resulting class of graphs would be larger than the one which was considered throughout our discussion.

**Corollary 4.13.** *Primality* $[\overline{C} \cap \overline{B}]$  is *GI*-hard or, equivalently,

$$GI[C] \leq_T \text{Primality}[\overline{C} \cap \overline{B}]$$

*Proof.* The proof follows directly from the proof of Theorem 4.12 by negating the result provided by the oracle  $\mathcal{A}$ .  $\square$

## 5. Conclusions

In this paper we proved that primality testing of unconnected, nonbipartite graphs with respect to direct product is at least as hard as deciding graph isomorphism. The same result also applies to the computation of a prime factorization of a graph. This result answer a long standing open question posed in [1] and shows the crucial role played by connectedness in decomposing a graph.

It would be of some interest to investigate the reversed question, i.e., whether deciding graph isomorphism is at least as hard as primality testing. Another interesting research direction is the design and implementation of efficient heuristics for computing a prime factorization or its approximation of large, possibly unconnected and/or weighted graphs knowing that a polynomial time algorithm for computing such a prime factorization is unlikely to exist.

## References

- [1] W. Imrich, Factoring cardinal product graphs in polynomial time, *Discrete Mathematics* 192 (1) (1998) 119–144. doi:10.1016/S0012-365X(98)00069-7.
- [2] R. Hammack, W. Imrich, S. Klavžar, *Handbook of Product Graphs, Second Edition*, Discrete Mathematics and Its Applications, Taylor & Francis, 2011.
- [3] P. M. Weichsel, The kronecker product of graphs, *Proceedings of the American Mathematical Society* 13 (1) (1962) 47–52.
- [4] E. L. Post, Recursively enumerable sets of positive integers and their decision problems, *Bull. Amer. Math. Soc.* 50 (1944) 284–316. doi:10.1090/S0002-9904-1944-08111-1.
- [5] H. Rogers, *Theory of recursive functions and effective computability*, McGraw-Hill, 1967.
- [6] G. Sabidussi, Graph multiplication., *Mathematische Zeitschrift* 72 (1959/60) 446–457. URL <http://eudml.org/doc/183624>
- [7] V. G. Vizing, The cartesian product of graphs, *Vyčisl. Sistemy* No. 9 (1963) 30–43.
- [8] J. Feigenbaum, J. Hershberger, A. A. Schäffer, A polynomial time algorithm for finding the prime factors of cartesian-product graphs, *Discrete Applied Mathematics* 12 (2) (1985) 123–138. doi:10.1016/0166-218X(85)90066-6.
- [9] P. Winkler, Factoring a graph in polynomial time, *Eur. J. Comb.* 8 (2) (1987) 209–212. doi:10.1016/S0195-6698(87)80012-4.
- [10] J. Feigenbaum, A. A. Schäffer, Recognizing composite graphs is equivalent to testing graph isomorphism, *SIAM Journal on Computing* 15 (2) (1986) 619–627. doi:10.1137/0215045.
- [11] J. Feigenbaum, A. A. Schäffer, Finding the prime factors of strong direct product graphs in polynomial time, *Discrete Mathematics* 109 (1) (1992) 77–102. doi:10.1016/0012-365X(92)90280-S.
- [12] M. Aigner, G. Ziegler, *Bertrand’s postulate*, Springer-Verlag Berlin Heidelberg, 2010, Ch. Bertrand’s postulate, pp. 7–12. doi:10.1007/978-3-642-00856-6\_2.
- [13] T. Tao, E. C. III, H. Helfgott, Deterministic methods to find primes, *Mathematics of Computation* 81 (2012) 1233–1246. doi:10.1090/S0025-5718-2011-02542-1.
- [14] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, *Introduction to Algorithms*, 3rd Edition, MIT Press, 2009.