Article

Bipartite (P_6,C_6)-Free Graphs: Recognition and Optimization Problems

Ruzayn Quaddoura * and Ahmad Al-Qerem

Computer Science Department, Faculty of Information Technology, Zarqa University, Zarqa 13110, Jordan; ahmad_qerm@zu.edu.jo
* Correspondence: ruzayn@zu.edu.jo

Abstract: The canonical decomposition of a bipartite graph is a new decomposition method that involves three operators: parallel, series, and $K \bigoplus S$. The class of weak-bisplit graphs is the class of totally decomposable graphs with respect to these operators, and the class of bicographs is the class of totally decomposable graphs with respect to parallel and series operators. We prove in this paper that the class of bipartite (P_6, C_6)-free graphs is the class of bipartite graphs that are totally decomposable with respect to parallel and $K \bigoplus S$ operators. We present a linear time recognition algorithm for (P_6, C_6)-free graphs that is symmetrical to the linear recognition algorithms of weak-bisplit graphs and $star_{1,2,3}$-free bipartite graphs. As a result of this algorithm, we present efficient solutions in this class of graphs for two optimization graph problems: the maximum balanced biclique problem and the maximum independent set problem.

Keywords: bipartite graphs; graphs decomposition; complexity; optimization problems

Citation: Quaddoura, R.; Al-Qerem, A. Bipartite (P_6,C_6)-Free Graphs: Recognition and Optimization Problems. *Symmetry* **2024**, *16*, 447. https://doi.org/10.3390/sym16040447

Academic Editor: Manuel Lafond

Received: 25 February 2024
Revised: 24 March 2024
Accepted: 26 March 2024
Published: 7 April 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

All graphs under consideration are undirected and simple. A graph $G = (V, E)$ is called bipartite if the vertex set V can be partitioned into two sets: B, called the black vertices, and W, called the white vertices, such that $E \subseteq B \times W$. So, a bipartite graph will be referred to as $G = (B \cup W, E)$. This property makes bipartite graphs useful in various practical applications, including recommender systems [1], social networks [2], and information retrieval [3]. A bipartite graph $G = (B \cup W, E)$ is called complete or biclique if $E = B \times W$. A complete bipartite graph with $|B| = n$ and $|W| = m$ is referred to as $K_{n,m}$. A $Star_{i,j,k}$ is a tree for which there is only one vertex v of degree three and three other vertices of degree one such that the distance from v to those vertices are, respectively, i, j, and k. For example, $K_{1,3}$ is $Star_{1,1,1}$. We can remark that every connected component in a bipartite $Star_{1,1,1}$-free graph is either a chordless path or a chordless cycle. Fouquet et al. in [4] presented a decomposition method concerning bipartite graphs, called canonical decomposition, which is based on three operators: series, parallel, and $K \bigoplus S$. They proved that the class of graphs that are totally decomposable with respect to these three operators of decomposition is the class of bipartite $\left(Star_{1,2,3}, P_7\right)$-free graphs, which is called the class of weak-bisplit graphs since it is considered as a bipartite analog of the class of split graphs. The class of weak-bisplit graphs is a natural generalization of the class of bipartite $\left(P_6, C_6\right)$-free graphs, which is a natural generalization of the class of bipartite $Star_{1,2,2}$-free graphs, studied by Lozin in [5]. Giakoumakis et. al. [6] defined the class of bicographs as a bipartite analog of the class of cographs. It is proved in [6] that the class of bicographs is exactly the class of bipartite $\left(Star_{1,2,3}, P_7, Sun_4\right)$-free graphs (see Figure 1) and is the class of totally decomposable graphs with respect to parallel and series decompositions. We prove in this work that the class of bipartite (P_6, C_6)-free graphs is the class of totally decomposable graphs with respect to parallel and $K \bigoplus S$ decompositions. As a result of this fact, the class of bipartite (P_6, C_6)-free graphs can be recognized in linear time using the recognition algorithm of weak-bisplit graphs presented in [7] or the recognition algorithm of $Star_{1,2,3}$-free graphs presented in [8]. However, since these two algorithms contain several redundant cases when projected to bipartite (P_6, C_6)-free graphs, we propose a symmetric algorithm of these two algorithms to adapt only the class of bipartite (P_6, C_6)-free graphs. As a result of this adapted algorithm, we present efficient solutions in this class of graphs for two optimization graph problems: the first is the maximum balanced biclique problem, and the second is the maximum independent set problem.

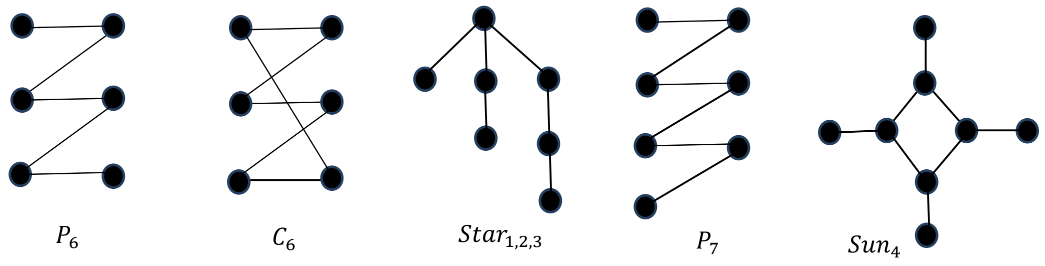

Figure 1. The configuration $P_{6,}C_6$, $Star_{123}$, and Sun_4.

2. Notation and Terminology

For a graph G, we use $V(G)$ to represent its vertex set and $E(G)$ to represent its edge set. The number $|V(G)|$ is represented by n and the number $|E(G)|$ is represented by m. If the two vertex sets B and W in a bipartite graph $G = (B \cup W, E)$ are both nonempty, the graph is referred to as bichromatic; otherwise, it is monochromatic. The bi-complement of bipartite graph $G = (B \cup W, E)$ is defined as $\overline{G}^{bip} = (B \cup W, B \times W - E)$. The set of neighbors of a vertex v in G is represented by $N(v)$, and the number $|N(v)|$ is referred to as the degree of v and is represented by $d(v)$. A vertex v is considered isolated if $d(v) = 0$ and is considered universal if $d(v) = |B|$ when $v \in W$ or $d(v) = |W|$ when $v \in B$. A $2K_2$ is the complement of C_4. A subset S of vertices of $V(G)$ is called an independent set if there is no edge between any two vertices of S. The sub-graph induced by a subset X of vertices of $V(G)$ is represented by $G[X]$. A graph G is considered Z-free, where Z is a set of graphs, if for any subset $X \subseteq V(G)$, $G[X] \notin Z$; that is, there is no sub-graph of G isomorphic to a graph in Z. The decomposition of graphs according to predefined operators is a powerful method for obtaining efficient solutions to a large number of graph problems. The reader can find a survey on graph decomposition methods and their uses in [9]. In this direction, Fouquet et al. in [4] presented a decomposition method concerning bipartite graphs, which is based on three operators: decomposing a bipartite graph into connected components, decomposing the bi-complement of a bipartite graph into connected components, and decomposing a bipartite graph into $K \bigoplus S$ components. Our recognition algorithm of bipartite (P_6, C_6)-free graphs depends mainly on this method of decomposition, so to introduce our algorithm, we need to present an overview of this method.

Definition 1 [4]. *A bipartite graph $G = (B \cup W, E)$ where $n \geq 2$ is a $K \bigoplus S$ graph if the vertex set $V(G)$ contains an isolated vertex or there is a partition of $V(G)$ into two sets: a biclique set K and an independent set S.*

Property 1 [4]. *A bipartite graph $G = (B \cup W, E)$ with $n \geq 2$ is a $K \bigoplus S$ graph if and only if $V(G)$ can be partitioned into two sets, V_1 and V_2, such that for every black vertex $b \in V_1$ and every white vertex $w \in V_2$, $bw \in E$; and for every white vertex $w \in V_1$ and every black vertex $b \in V_2$, $bw \notin E$.*

We denote the ordered pair (V_1, V_2) to represent the partition of the vertex set $V(G)$ of $K \bigoplus S$ graph G and we called it a $K \bigoplus S$ partition of G.

Property 2 [4]. *In a bipartite graph $G = (B \cup W, E)$ that does not contain universal or isolated vertices, either G is a $K \bigoplus S$ graph or, for any partition of the black vertices B or white vertices W into two sets X_1 and X_2, there is an induced $2K_2$ in G with vertices in both X_1 and X_2.*

Corollary 1 [4]. *If a graph G is not a $K \bigoplus S$ graph, then for every vertex $v \in V(G)$, v is a vertex of some $2K_2$.*

Theorem 1 provides a method for decomposing a graph of type $K \bigoplus S$.

Theorem 1 [4]. *A bipartite graph G is a $K \bigoplus S$ graph if and only if there exists a unique partition $(V_1, \ldots, V_r)$ of $V(G)$, such that the following conditions hold:*

- *For every $i = 1, \ldots, r, V_i \neq \varnothing$.*
- *For every $i = 1, \ldots, r-1$, the sets $V_1 \cup \ldots \cup V_i$ and $V_{i+1} \cup \ldots \cup V_r$ form a $K \bigoplus S$ partition of G.*
- *For every $i = 1, \ldots, r$, the sub-graph $G[V_i]$ is not a $K \bigoplus S$ graph.*

The partition $(V_1, \ldots, V_r)$ of the vertex set $V(G)$ is called the $K \bigoplus S$ decomposition of the graph G, and each set V_i is referred to as a $K \bigoplus S$ component of G.

According to Theorem 1, the arrangement of components in a $K \bigoplus S$ decomposition is important. Specifically, let $(V_1, \ldots, V_r)$ be the $K \bigoplus S$ decomposition of G; if a black vertex $b \in V_i$, then for every white vertex $w \in V_j$ where $j > i$, $bw \in E(G)$, and for every white vertex $w \in V_k$ where $k < i$, $bw \notin E(G)$. Similarly, if a white vertex $w \in V_i$, then for every black vertex $b \in V_j$ where $j > i$, $bw \notin E(G)$, and for every black vertex $b \in V_k$ where $k < i$, $bw \in E(G)$.

The canonical decomposition of a bipartite graph G is a new decomposition method defined in [4] as follows:

- Decompose G into its $K \bigoplus S$ components; if G is a $K \bigoplus S$ graph, this decomposition is called $K \bigoplus S$ decomposition and is denoted by $K \bigoplus S$.
- Decompose G into the connected components of G; if G is not a connected graph, this decomposition is called parallel decomposition and is denoted by P.
- Decompose G into the connected components of $\overline{G}^{bip}$; if $\overline{G}^{bip}$ is not a connected graph, this decomposition is called series decomposition and is denoted by S.
- If G cannot be decomposed in $K \bigoplus S$, parallel, or series decomposition, then G is called an indecomposable graph or a prime graph.

It has been proven in [4] that no matter the order in which the operator of the decomposition is applied (series decomposition, parallel decomposition, or $K \bigoplus S$ decomposition), the set of indecomposable graphs obtained is unique. This also creates a unique tree (up to isomorphism) associated with this decomposition known as the canonical decomposition tree. The internal nodes of the tree are labeled by the type of decomposition applied, and the leaves correspond to indecomposable graphs. Figure 2 shows an illustration of a bipartite graph and its canonical decomposition tree.

The canonical decomposition tree $T(G)$ of a bipartite graph G resulting in an order from the $K \bigoplus S$ decomposition, parallel decomposition, and series decomposition procedure has several properties outlined below. The terms vertex node, son, parent, and grandparent are used in their conventional sense. If α is an internal node, then $G[\alpha]$ is the sub-graph induced by the set of vertex nodes having α as their least common grandparent.

(1) The tree $T(G)$ consists of three types of internal nodes: parallel nodes denoted by P, series nodes denoted by S, and $K \bigoplus S$-nodes.
(2) Two consecutive internal nodes cannot have the same label.
(3) An internal node δ labeled P or S cannot have a son that is a vertex node v. Otherwise, v would be either an isolated or universal vertex in $G[\delta]$. By the definition of a $K \bigoplus S$ graph, $G[\delta]$ would be a $K \bigoplus S$ graph.
(4) The parent of a vertex node is always labeled $K \bigoplus S$ (a consequence of Property 3).

(5) If G is a bi-chromatic graph, then for any $K \bigoplus S$ node δ, $G[\delta]$ must also be bi-chromatic. Otherwise, if there is a node δ labeled $K \bigoplus S$ and $G[\delta]$ is a monochromatic graph, then the parent of δ, say, γ, would have isolated or universal vertices so $G[\gamma]$ would be a $K \bigoplus S$ graph, a contradiction with Property 2.
(6) The sons of a $K \bigoplus S$ node are ordered according to the $K \bigoplus S$ decomposition.
(7) Let δ be a $K \bigoplus S$ node, and δ_1 and δ_2 are, respectively, the first and last sons of δ. If the parent of δ, say, γ, is a P-node, then δ_1 cannot be a white vertex node and δ_2 cannot be a black vertex node. Otherwise, by Property 1, δ_1 and δ_2 are isolated vertices in $G[\delta]$; since γ is a P-node, δ_1 and δ_2 are also isolated vertices in $G[\gamma]$, so γ must be a $K \bigoplus S$ node, a contradiction with Property 2.
(8) If the parent of a $K \bigoplus S$-node δ is labeled S, and δ_1 and δ_2 are, respectively, the first and last sons of δ, then δ_1 cannot be a black vertex node and δ_2 cannot be a white vertex node (similar to Property 7).

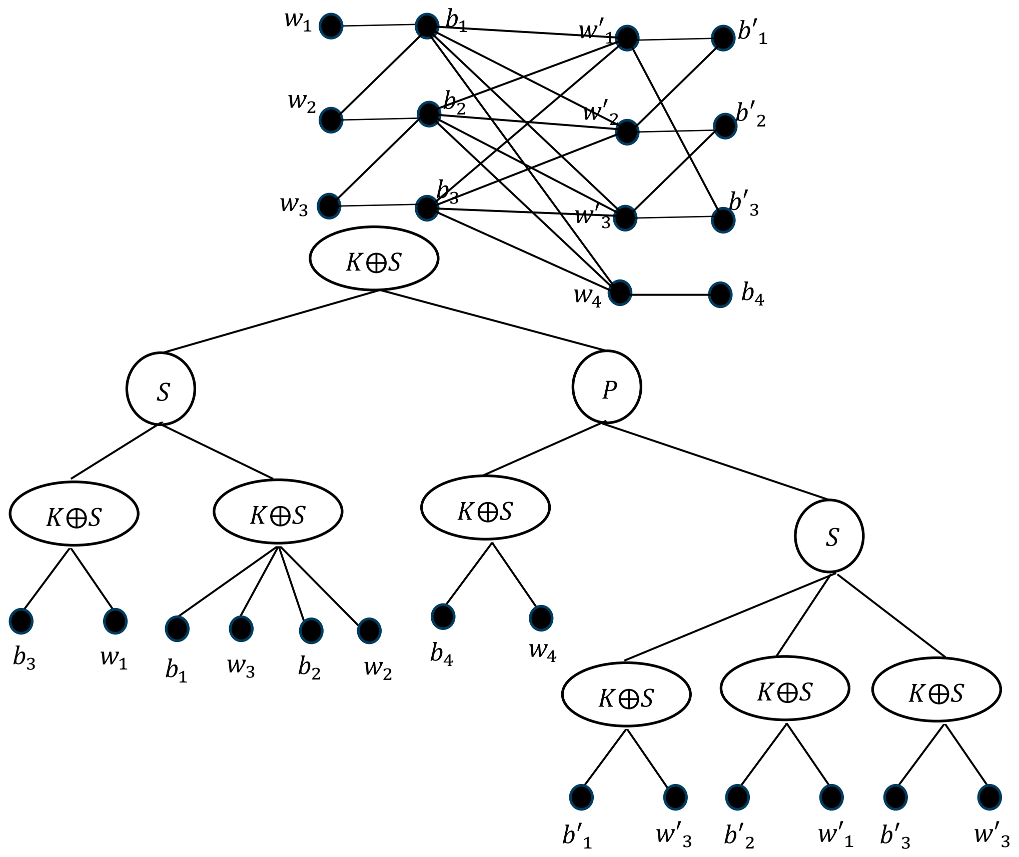

Figure 2. An example of bipartite graph G and the canonical decomposition tree $T(G)$.

3. Recognition Algorithm of Bipartite (P_6,C_6)-Free Graphs

The following theorem is the key to our recognition algorithm for bipartite (P_6, C_6)-free graphs.

Theorem 2. *A bipartite graph G is (P_6, C_6)-free if and only if every connected sub-graph of G is a $K \bigoplus S$ graph.*

Proof. Suppose that G is a bipartite (P_6, C_6)-free graph. Let H be a connected sub-graph of G which is not a $K \bigoplus S$ graph. By Corollary 1, H contains a $2K_2$, say, b_1w_1, b_2w_2. Since H is a connected sub-graph of G and G is *a* (P_6, C_6)-free graph, there is a vertex in $V(H)$ that connects b_1w_1 and b_2w_2. Suppose without a loss of generality that b is a black vertex such that bw_1, $bw_2 \in E(H)$. Since H is not a $K \bigoplus S$ graph, the vertex b is not universal, so there is a white vertex w such that $bw \notin E(H)$. Since H is connected, there is

a path in H that connects the vertex w and the path $P_5 = b_1, w_1, b, w_2, b_2$. But now, the set $\{b_1, w_1, b, w_2, b_2, w\}$ forms a P_6 or a C_6, a contradiction.
The inverse is clear since a P_6 or a C_6 is connected and contains a $2K_2$, so it is not a $K \bigoplus S$ graph. □

Theorem 2 states that the class of bipartite (P_6, C_6)-free graphs is the smallest class closed under parallel and $K \bigoplus S$ decomposition. So, the canonical decomposition tree of a bipartite (P_6, C_6)-free graph consists only of P-nodes or $K \bigoplus S$-nodes. Our recognition algorithm builds a decomposition tree with P- or $K \bigoplus S$-labeled internal nodes if the input graph is (P_6, C_6)-free; otherwise, it produces a failure message. This building was influenced by the cograph recognition method proposed by Corneil et al. in [10]. Moreover, this algorithm greatly simplifies two recognition algorithms when projected on bipartite (P_6, C_6)-free graphs. The first is for weak-bisplit graphs presented in [7], and the second for bipartite $Star_{123}$-free graphs presented in [8], where both these two algorithms need to examine more than twenty cases to confirm that the input graph is (P_6, C_6)-free or not, while, as we will see, our algorithm needs to examine only two cases that are presented below in Theorems 3 and 4. The algorithm begins with an empty graph and gradually adds vertices, ensuring that the resulting sub-graphs remain (P_6, C_6)-free. The initial bipartite graph is considered (P_6, C_6)-free if all vertices can be added successfully in this manner. The principal step of the algorithm takes into consideration the decomposition tree T of a (P_6, C_6)-free bipartite graph $G = (B \cup W,\ E)$, a vertex $x \notin B \cup W$, and a set of edges denoted by $E(x) = \{xv : v \in B \cup W \text{ and } v \in N(x)\}$ and produces the decomposition tree T' of the resulting graph $G' = (B \cup W \cup \{x\},\ E \cup E(x))$ if it remains (P_6, C_6)-free or stops otherwise. The algorithm considers the connections of x to other vertices in G using a marking procedure. We can assume without a loss of generality that x is a white vertex and the graph G is a bichromatic graph.

3.1. Marking Procedure

The marking procedure, presented in Algorithm 1 and used in [10], takes into consideration the neighbors of the vertex x in the graph G to mark the nodes of T, the decomposition tree of G.

Algorithm 1 Marking

```
Input: The canonical decomposition tree $T$ of $G$ and the white vertex $x$.
Output: The marking tree $T$.

For every black vertex node $v$ of $T$.
If $v$ is a neighbor of $x$, mark $v$ by $(t)$ if $v$ is not a neighbor of $x$, mark $v$ by $()$.
Traverse $T$ on a bottom-up traversal, let $\alpha$ be an internal node of $T$:
If every son of $\alpha$ which is distinct from a white vertex node is marked by $()$ then mark $\alpha$ by $()$.
If there is a son of $\alpha$ marked by $(t)$ and a son marked by $()$ then mark $\alpha$ by $(p)$.
If every son of $\alpha$ which is distinct from a white vertex node is marked by $(t)$ then mark $\alpha$ by $(t)$.
```

At the end of the marking procedure on tree T, a node can have three possible states: marked by (t), marked by (p), or marked by $()$. If a node δ is marked by (t), it means that x is total for $G[\delta]$, that is, x is connected to all black vertices in $G[\delta]$. If it is marked by (p), it means that x is partial for $G[\delta]$, that is, x is connected to some but not all black vertices in $G[\delta]$. If it is marked by $()$, it means that x is independent of $G[\delta]$, that is, x is not connected to any black vertices in $G[\delta]$. If a node is a vertex node, it can either be marked by $()$ or marked by (t). By Theorem 2, the marking procedure focuses only on P-nodes and $K \bigoplus S$-nodes, ignoring the S-nodes that must be unavailable. For the graph G' to be considered bipartite and (P_6, C_6)-free, it must meet a necessary condition.

Lemma 1. *If two internal nodes in the tree T are marked by (p), and G' is a (P_6, C_6)-free bipartite graph, then one of these two nodes must be a grandparent of the other.*

Proof. Suppose that α and β are two internal nodes marked by (p), then α and β are partial with respect to x. Let δ be the least common grandparent of α and β. We denote α' and β' to be, respectively, the son of δ containing α and the son of δ containing β. Since $G[\alpha]$ and $G[\beta]$ are sub-graphs of $G[\alpha']$ and $G[\beta']$, then α' and β' are partial with respect to x. Assume that δ is labeled P, then $G[\alpha']$ and $G[\beta']$ are connected sub-graphs of $G[\delta]$. Thus, there is an induced path b_1, w_1, b_2 in $G[\alpha']$ (resp. b'_1, w'_1, b'_2 in $G[\beta']$) such that x is adjacent to b_1 and not adjacent to b_2 (resp. to b'_1 and not to b'_2). The set $\{w'_1, b'_1, x, b_1, w_1, b_2\}$ forms a P_6, a contradiction.
Assume that δ is labeled $K \bigoplus S$. By Corollary 1, α' (resp. β') contains a $2K_2$ that is partial with respect to x. Let b_1w_1, b_2w_2 (resp. $b'_1w'_1, b'_2w'_2$) be a $2K_2$ in $G[\alpha']$ (resp. in $G[\beta']$ such that x is adjacent to b_1 and not adjacent to b_2 (resp. to b'_1 and not to b'_2). Now, the set $\{w_2, b_2, w'_2, b_1, x, b'_1, w'_1\}$ forms a P_6, a contradiction. □

By Lemma 1, the nodes in T that are marked by (p) are arranged in a single path that starts from the lowest node marked by (p) and goes up to the root. The lowest node marked by (p) is referred to as α. It is assumed that the conditions of Lemma 1 are met and that α is known. The following notations are introduced:

- Given two internal nodes δ and δ' such that δ is a grandparent of δ', the unique son of δ that contains δ' is denoted as $son(\delta, \delta')$.
- For an internal node δ, which is either α or one of its grandparents, the set of sons of δ that are marked by $()$ is denoted as $sons^{()}(\delta)$, and the set of sons that are marked by (t) is denoted as $sons^{(t)}(\delta)$.
- If δ has a label $K \bigoplus S$, considering the ordering of the sons of δ, the set of sons of δ marked by (t) and located before $son(\delta, \alpha)$ is denoted as $sons_1^{(t)}(\delta)$, and the set of sons marked by (t) and located after $son(\delta, \alpha)$ is denoted as $sons_2^{(t)}(\delta)$. The set of sons of δ marked by $()$ and located before $son(\delta, \alpha)$ is denoted as $sons_1^{()}(\delta)$, and the set of sons marked by $()$ and located after $son(\delta, \alpha)$ is denoted as $sons_2^{()}(\delta)$.

So, a node δ labeled P which is a grandparent of α splits its sons into at most three categories: $sons^{(t)}(\delta)$, $sons^{()}(\delta)$, and $son(\delta, \alpha)$, the son that contains α. If δ is labeled $K \bigoplus S$, its sons can be divided into at most five sets: $son(\delta, \alpha)$, $sons_1^{(t)}(\delta)$, $sons_1^{()}(\delta)$, $sons_2^{(t)}(\delta)$, and $sons_2^{()}(\delta)$. Meanwhile, the sons of α are split into two nonempty categories: $sons^{()}(\alpha)$ and $sons^{(t)}(\alpha)$.

Definition 2. *If δ is a grandparent of α in T, then δ is considered an incompatible P-node if it has at least one son marked by (t) (i.e., $sons^{(t)}(\delta) \neq \emptyset$). If δ is a $K \bigoplus S$-node, it is considered incompatible before α if it has at least one son marked by $()$ located before $son(\delta, \alpha)$ (i.e., $sons_1^{()}(\delta) \neq \emptyset$). If δ is a $K \bigoplus S$-node, it is considered incompatible after α if it has at least one son marked by (t) located after $son(\delta, \alpha)$ (i.e., $sons_2^{(t)}(\alpha) \neq \emptyset$).*

3.2. Building the Tree T'

We assume that the necessary condition of Lemma 1 has been verified and that all nodes that are marked by (p) are known and are arranged in a single path that starts from the lowest marked node α and goes up to the root of T. To build T', we examine the label of α and the presence of incompatible marked nodes. When α is a $K \bigoplus S$-node, since it is the lowest node marked by (p), we can divide its group of sons into a maximum of four consecutive subsets, namely, $X_1^{(t)}$, $X_2^{()}$, $X_3^{(t)}$, and X_4, where:

$X_1^{(t)}$ includes the first group of consecutive sons of α that are either a group of white vertex nodes or total with respect to x.

$X_2^{()}$ includes the first group of consecutive sons of α that are not part of $X_1^{(t)}$ and are either a group of white vertex nodes or not related to x.

$X_3^{(t)}$ includes the first group of consecutive sons of α that are not part of $X_2^{()}$ nor of $X_1^{(t)}$ and are either a group of white vertex nodes or total with respect to x.

X_4 represents the remaining sons of α.

Note that as a result of this division of the sons of α, when its label is $K \bigoplus S$, $X_2^{()}$ and $X_3^{(t)}$ cannot be monochromatic graphs together.

Lemma 2. *Assume that α is a $K \bigoplus S$-node. If G' is a bipartite (P_6, C_6)-free graph, then the following conditions hold:*

(1) *x has no neighbor in X_4.*

(2) *If $X_3^{(t)}$ is empty, then $G\left[X_3^{(t)}\right]$ is a monochromatic graph or a complete bipartite graph.*

Proof. Suppose that X_4 is not empty, otherwise, we are done. Let it show that x has no neighbor in X_4. Since X_4 is not empty, then $X_3^{(t)}$ and $X_2^{()}$ are both nonempty. Let $b_4 \in X_4$ such that x is adjacent to b_4. Now, X_4 contains two adjacent vertices b'_4, w'_4 such that x is not adjacent to b'_4; otherwise, $b_4 \in X_3^{(t)}$. Let b_2 and b_3 be two black vertices of $X_2^{()}$ and $X_3^{(t)}$, respectively. By construction, there is a white vertex w such that w is adjacent to b_2 and w is not adjacent to b_3. But now, the set $\{b_4,\ x,\ b_3,\ w_4,\ b_2, w\}$ forms a P_6, a contradiction. □

Now, condition 2 must be satisfied. Suppose that $X_3^{(t)}$ is not empty, then $X_2^{()}$ is also not empty.

Claim 1. *Every element of $X_3^{(t)}$ is a vertex node.*

Proof. Suppose that δ is an element of $X_3^{(t)}$ that is an internal node. Then δ is a P-node; thus, it contains a $2K_2$, say, $b_1w_1,\ b_2w_2$. Let $b \in X_2^{()}$, then the set $\{b, w_1,\ b_1,\ x,\ b_2, w_2\}$ forms a C_6, a contradiction. □

Claim 2. $X_2^{()}$*contains a white vertex.*

Proof. Suppose that $X_2^{()}$ does not contain any white vertex, then $X_3^{(t)}$ contains an element that is an internal node, a contradiction with Claim 1.

Let b_2, w_2 be two vertices of $X_2^{()}$ such that $b_2w_2 \in E(G)$. Suppose that $G\left[X_3^{(t)}\right]$ is neither a monochromatic graph nor a complete bipartite graph. Then $X_3^{(t)}$ contains the vertices $b_3,\ b'_3, w_3$ such that b_3 is adjacent to w_3 and b'_3 is independent of w_3. Consequently, $\{b'_3,\ x,\ b_3, w_3,\ b_2, w_2\}$ forms a P_6, a contradiction. □

Theorem 3. *Assume that there is no incompatible grandparent of α. G' is a bipartite (P_6, C_6)-free graph if and only if either:*

(1) *α is a P-node; or*

(2) *α is a $K \bigoplus S$-node and Lemma 2 is holding.*

Proof. The if part of the Theorem has been proved in Lemma 2. We will describe the building of T' for the only if part. The building of T' when α is a P-node is described in Figure 3a. If $sons^{(t)}(\alpha)$ consists of a unique son, then this son will be a son of the node labeled $K \bigoplus S$. Suppose that α is a $K \bigoplus S$-node. If $X_3^{(t)}$ is empty, then we insert x in T as a new son of α. The building of T' when $G\left[X_3^{(t)}\right]$ is a monochromatic graph or a complete bipartite graph is also described in Figure 3b. In this case, if $G\left[X_3^{(t)}\right]$ is a monochromatic graph, then $W_\alpha = \varnothing$. □

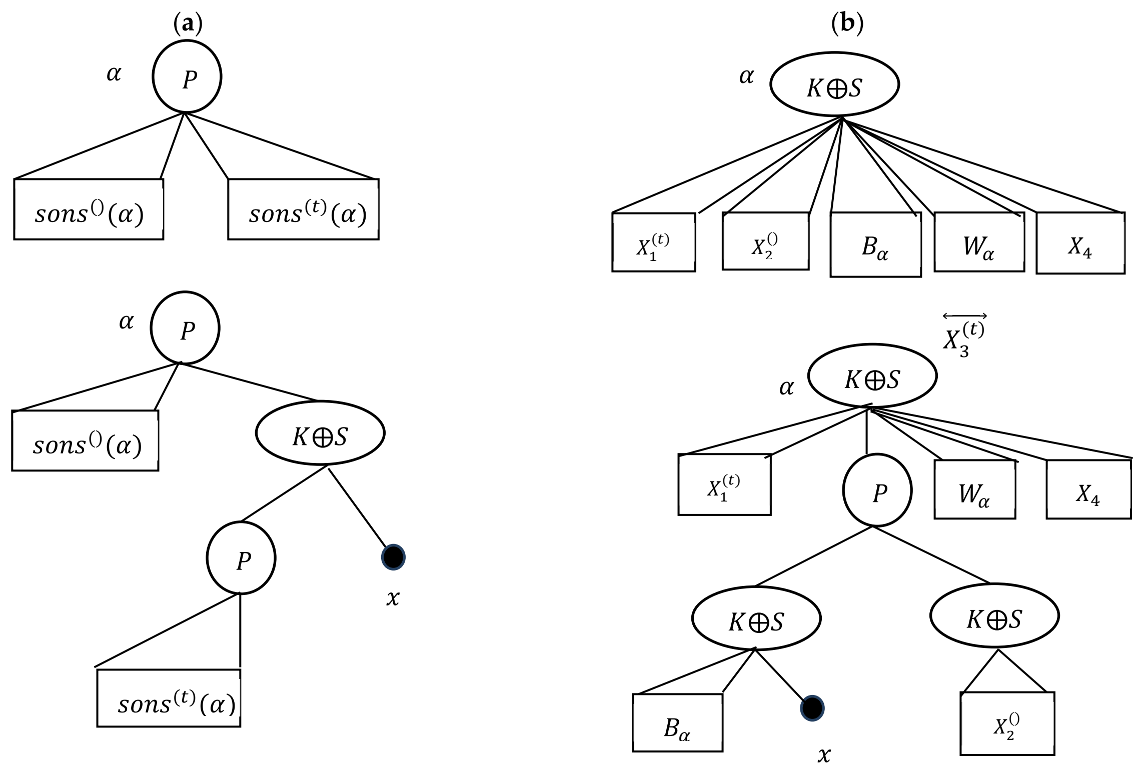

Figure 3. Building of T' when there is no incompatible grandparent of α.

Lemma 3. *Assume that G' is a bipartite $(P_6,\ C_6)$-free graph. If there is an incompatible grandparent β of α, then the following conditions hold:*

(1) *β is a $K \bigoplus S$-incompatible node after α.*

(2) *β is the unique incompatible grandparent of α.*

(3) *The set $sons_2^{(t)}(\beta)$ consists of black vertex nodes located exactly after $son(\beta, \alpha)$.*

Proof. Suppose that β is an incompatible grandparent of α of type P. Then there exists two adjacent vertices, b_3, w_3, in an element of $sons^{(t)}(\beta)$. Since $son(\beta, \alpha)$ induces a connected graph, it contains an induced P_3, say, b_1, w_1, b_2, such that b_1 is adjacent to x and b_2 is independent of x. But now, $\{x,\ b_1, w_1,\ b_2,\ b_3, w_3\}$ forms a P_6, a contradiction.

If β is a $K \bigoplus S$-incompatible node before α, then $sons_1^{()}(\beta)$ contains a black vertex b independent of x. By Corollary 1, $son(\beta, \alpha)$ contains a $2K_2$, say, b_1w_1, b_2w_2, such that x is adjacent to b_1 and x is independent of b_2. The set $\{x,\ b_1, w_1,\ b, w_2,\ b_2\}$ forms a P_6, a contradiction. Consequently, β is a $K \bigoplus S$-incompatible node after α. Let us consider β to be the highest incompatible grandparent of α and let $b \in sons_2^{(t)}(\beta)$. □

Claim 3. There is no grandparent δ of α containing a white vertex total for $son(\delta, \alpha)$.

Proof. Let δ be a grandparent of α and w is a white vertex of the δ total for $son(\delta, \alpha)$. Let b_1w_1, b_2w_2 be an induced $2K_2$ of $son(\delta, \alpha)$ such that x is adjacent to b_1 and independent of b_2. Then the set $\{b,\ x,\ b_1, w,\ b_2, w_2\}$ forms a P_6, a contradiction.

By this claim, if δ is a $K \bigoplus S$-incompatible node after α, then the set $sons_2^{(t)}(\delta)$ consists of black vertex nodes located exactly after $son(\delta, \alpha)$. Moreover, for every grandparent δ of α labeled $K \bigoplus S$ and located between α and β, $son(\delta, \alpha)$ is the last son of δ; otherwise, δ contains a white vertex total for $son(\delta, \alpha)$, a contradiction; thus, δ cannot be incompatible. Therefore, β is the unique incompatible grandparent of α. □

Theorem 4. *Assume that β is the unique $K \bigoplus S$-incompatible grandparent after α and that the set $sons_2^{(t)}(\beta)$ consists of black vertex nodes located exactly after $son(\beta, \alpha)$. G' is a bipartite (P_6, C_6)-free graph if and only if one of the following conditions holds:*

(1) *α is a P-node.*

(2) *α is a $K \bigoplus S$-node such that $X_3^{(t)}$ is an empty set.*

Proof. Suppose that α is a $K \bigoplus S$-node and let $b \in sons_2^{(t)}(\beta)$. Suppose that $X_3^{(t)}$ is nonempty. By Lemma 2, $G\left[X_3^{(t)}\right]$ is a monochromatic graph or a complete bipartite graph. In the two cases, $G\left[X_2^{()}\right]$ cannot be a monochromatic graph, otherwise, $X_3^{(t)}$ would be empty. Thus $X_2^{()}$ contains two adjacent vertices, b_2, w_2. Let b_3 be a black vertex of $X_3^{(t)}$. Since $G[\alpha]$ is a connected graph, then there is a white vertex, say, w_3, in the last son of α, but now $\{b,\ x,\ b_3, w_3,\ b_2, w_2\}$ forms a P_6, a contradiction.
For the only if part, we describe the building of T'. When α is a P-node, the building of T' is illustrated in Figure 4a. If the set $sons^{(t)}(\alpha)$ is a unique son, then this son must be labeled $K \bigoplus S$. In this case, we delete the node δ_1, and the element $sons^{(t)}(\alpha)$ will be a son of δ_2. The building of T' when condition 2 is satisfied is illustrated in Figure 4b. If $X_2^{()}$ is a unique son, then this son is either a node labeled P or a black vertex node. In this case, we delete the node δ_2, and the element $X_2^{()}$ will be a son of δ_1. $\square$

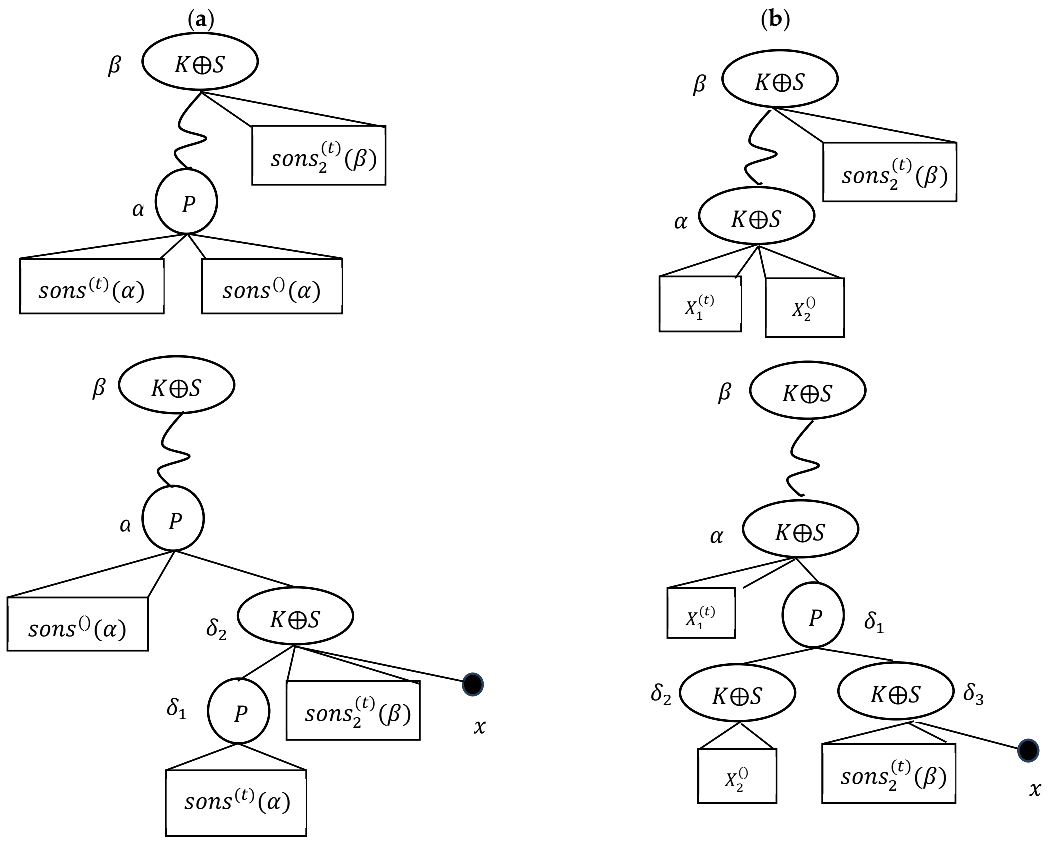

Figure 4. Building of T' when β, the unique $K + S$-incompatible grandparent of α, exists.

3.3. Recognition Algorithm

The recognition algorithm of bipartite (P_6, C_6)-free graphs is given by Algorithm 2, whereas the procedure of the step Build-tree $(G',\ T,\ head(L))$ is presented in Algorithm 3.

Algorithm 2 Recognition of bipartite (P_6, C_6)-free graph

```
Input: a bipartite graph $G = (B \cup W, E)$.
Output: if $G$ is a $(P_6, C_6)$-free graph then the canonical decomposition tree $T(G)$, otherwise a failure message "$G$ is not $(P_6, C_6)$-free graph".
Initialization step: Let $L$ be the list of all the vertices of $G$ sorted in descending order according to their degrees.
$T =$ new-vertex;
$G' = \emptyset$;
Build-tree $(G', T, head(L))$.
```

Algorithm 3 Procedure Build-tree $(G',\ T,\ head(L))$

```
(1)  Marking $(T, x)$
(2)  Find the set $S = \{\delta : \delta$ is an internal node marked by $(p)\}$
(3)  If $S = \emptyset$ then $T = insert(x, T)$
     (If $x$ is independent of $T$ (resp. total for $T$) then create a new root $\delta$ of $T$ labeled $K \bigoplus S$ such that $x$ is the left (resp. right) son of $\delta$ and the root of $T$ is the right (resp. left) son of $\delta$)
(4)  Else if $|S| > 2$ then Exit with the message "failure".
(5)  Else if $|S| = 1$ then $T = insert(x, T)$ (according to Theorem 3)
(6)  Else if one of the two nodes of $S$ is not a grandparent of the other then Exit with the message "failure".
     Else let $S = \{\alpha, \beta\}$ and $\beta$ is the grandparent of $\alpha$
(7)  If $\beta$ is a $P$-node then Exit with the message "failure".
(8)  Else $T = insert(x, T)$ (according to Theorem 4)
(9)  $G' = G[V(G') \cup \{x\}]$;
(10) If $L = \emptyset$ then Exit else $L = L - \{x\}$; $x = head(L)$; Build-tree$(G',\ T,\ x)$
```

3.4. Complexity

The aim of this section is to demonstrate that recognizing a bipartite (P_6, C_6)-free graph G can be accomplished in $O(n+m)$ time complexity. Since the principal step of our algorithm is the step Build-tree $(G',\ T,\ x)$, we will demonstrate the linearity of our algorithm by showing that this step requires only $O(d_{G'}(x))$ operations, where $d_{G'}(x)$ is the degree of the node x in G'.

It is evident that step 1 runs within $O(d_{G'}(x))$ time, as only a maximum of $O(d_{G'}(x))$ nodes are marked. Furthermore, we can assume that for every node in the tree T, the set of its sons that are marked by (t), by (p), or by $()$ has been calculated. So, finding the set S also requires $O(d_{G'}(x))$ operations. Suppose that $S = |2|$. We can check whether one of the two nodes of S is a grandparent of the other as follows: Choose an element of S and start to mark the parent of this element, then mark the parent of the parent, and so on until the other element of S is marked or until the root of T is marked. For the last case, that is, if the root of T has been marked, we repeat this process for the other element of S. In this manner, we can also determine the node α, the lowest node marked by (p), and the grandparent β. This process can be conducted in $O(d_{G'}(x))$ mark operations.

Next, we must analyze the time complexity of the function *insert* $(x,\ T)$. This requires verifying the necessary conditions in Theorem 3 or Theorem 4. Therefore, we need to compute all the required sets for building the tree T'. If α has the label P, then the computation of the set $sons^{(t)}(\alpha)$ and the set $sons^{()}(\alpha)$ is straightforward. Suppose α has the label $K \oplus S$. We can compute the sets $X_1^{(t)}$, $X_2^{()}$, $X_3^{(t)}$, and X_4 as follows: First, we compute $X_1^{(t)}$ by traversing the set of sons of α from left to right. In this manner, $X_1^{(t)}$ will be the first nodes that are either a set of white vertices or nodes marked by (t); we continue this traversing until a son of α marked by $()$ has been found. The remaining sons of α marked by (t) must belong to $X_3^{(t)}$ since X_4 is independent of x according to Lemma 2. To compute $X_3^{(t)}$, we choose a son of α (let us call it c) from the remaining nodes marked by (t) and we traverse the set of sons of α starting from c in the left and right directions, from the left until a son of

α marked by $()$ has been found and from the right also until a son of α marked by $()$ has been found or until the last son from the right has been found. We continue this traversing until every son is either a white vertex node or a node marked by (t). As soon as the set $X_3^{(t)}$ has been computed, the set $X_2^{()}$ can be computed immediately. The remaining sons of α form the set X_4 which must be independent of x. This computation requires $O(d_{G'}(x))$ time complexity.

Finally, we need to determine if the node β, whose label is $K \oplus S$, is incompatible after α or not and check whether the set $sons_2^{(t)}(\beta)$ is a set of black vertex nodes located exactly after $son(\beta, \alpha)$. These two conditions can be achieved together as follows: Since β is known as a grandparent of α, the son $son(\beta, \alpha)$ is identified. Now, we can traverse the sons of β starting from the first son located exactly after $son(\beta, \alpha)$ and determine whether any one of these sons is a white vertex node or not. In addition, we must traverse the sons of β starting from the first son located exactly before $son(\beta, \alpha)$ to determine if the set $sons_1^{()}(\delta)$ is empty or not. This traverse also requires $O(d_{G'}(x))$ time complexity.

We leave it to the reader to verify that the step Build-tree $(G, T, x))$ that corresponds to insert x in the tree T takes a constant time in all cases.

Since testing whether $G' = G \cup \{x\}$ is a bipartite (P_6, C_6)-free graph or not can be performed within $O(d_{G'}(x))$ time complexity, it is clear that recognition of the bipartite (P_6, C_6)-free graph algorithm runs in $O(n + m)$ time complexity.

4. Optimization Problems

We believe that the canonical decomposition tree for a bipartite (P_6, C_6)-free graph can be used to find efficient solutions for several optimization graph problems because of the simple structure of this tree. In this paper, we limit ourselves to showing that the canonical decomposition tree of a bipartite (P_6, C_6)-free graph can be used to solve, in polynomial time, the maximum balanced biclique problem and, in linear time, the maximum independent set problem. In the concluding section of this paper, we talk about some potential uses for this result and consider it as a subject for further study.

Let $T(G)$ be the canonical decomposition tree for a bipartite (P_6, C_6)-free graph G. To present our solutions to the above two problems, we need to covert $T(G)$ to a binary tree as follows:

Visit the nodes of $T(G)$ in a depth-first search.

Let S be an internal visited node, and $S_1, \ldots, S_k$ are the sons of S. If $k > 2$, then the left son of S is S_1, and the right son becomes a new son, S', that has the same label as S with sons $S_2, \ldots, S_k$.

4.1. Maximum Balanced Biclique Problem

A sub-graph $F = G[X \cup Y]$ of a bipartite graph G is called a balanced biclique if F is a biclique and $|X| = |Y|$. The balanced biclique problem is computing a balanced biclique in G of maximum size. This problem is important in many different fields of study. It has numerous practical uses in very-large-scale integration (VLSI), such as the design of defect-tolerant devices [11,12], and programmable logic array folding [13]. The balanced biclique problem is NP-complete for a general bipartite graph [14], and there are very few works dedicated to obtaining an exact maximum balanced biclique, aside from the work [15] where two exact algorithms are proposed to find a maximum balanced biclique for small dense and large sparse bipartite graphs, respectively. The majority of known techniques for determining a maximum balanced biclique are heuristic algorithms (see, for example, [16,17]).

We propose in this work an $O(n^3)$ time complexity algorithm to compute a maximum balanced biclique in a bipartite (P_6, C_6)-free graph G using its canonical decomposition binary tree $T(G)$. The idea of our solution is to compute all possible bicliques in G that are maximal with respect to set inclusion, then find among them the one that contains a maximum balanced biclique. A biclique F is maximal with respect to set inclusion if no biclique in G contains F. The structure of $T(G)$ when G is a bipartite (P_6, C_6)-free graph and the definition of $K \bigoplus S$ operation allow us to achieve this computation by a post-order traversal of $T(G)$, associating for each internal node α all possible maximal bicliques in $G[\alpha]$ (with respect to set inclusion) through the two sets of maximal bicliques associated with the left son α_1 and the right son α_2 of α. The set of maximal bicliques associated with α_1 denoted by $L(\alpha_1) = \{F_i^1 = G[X_i^1 \cup Y_i^1] \ : i = 1, \ldots, r\}$ and the set of maximal bi-cliques associated with α_2 denoted by $L(\alpha_2) = \{F_i^2 = G[X_i^2 \cup Y_i^2] \ : i = 1, \ldots, k\}$. We suppose that for every biclique $F_i^j = G[X_i^j \cup Y_i^j]$, X_i^j is a set of black vertices and Y_i^j is a set of white vertices. In addition, we suppose that the members of $L(\alpha_1)$ are arranged from left to right according to their appearance in the sub-tree $T(\alpha_1)$. Likewise, we suppose that the members of $L(\alpha_2)$ are arranged from left to right according to their appearance in the sub-tree $T(\alpha_2)$. This supposition is performed directly according to the arrangement of sons for every $K \bigoplus S$-node in $T(G)$. The reader can simply verify the truth of computation used in Algorithm 4 for the set of maximal bicliques $L(\alpha)$ according to the definition of a $K \bigoplus S$-node and the definition of a P-node. Figure 5 can help us to imagine this computation.

Algorithm 4 Balanced Bi-clique

```
Input: A binary canonical decomposition tree $T(G)$ of a bipartite $(P_6, C_6)-$free graph
$G = (B \cup W,\ E)$.
Output: A maximal balanced biclique $F = G[X \cup Y]$ for $G$
Let $\alpha$ be a node on a post − order traversal of $T(G)$
If $\alpha$ is a black vertex node $b$ (resp. a white vertex node $w$), then
$L(\alpha) = \{G[\{b\} \cup \varnothing]\}$ (resp. $L(\alpha) = \{G[\varnothing \cup \{w\}]\}$
Else let $\alpha_1$ and $\alpha_2$ be the left and right son of $\alpha$, respectively, and let
$L(\alpha_1) = \{F_i^1 = G[X_i^1 \cup Y_i^1] : i = 1, \ldots, r\}$, $L(\alpha_2) = \{F_i^2 = G[X_i^2 \cup Y_i^2] : i = 1, \ldots, k\}$.
If $\alpha$ is a $K \bigoplus S$-node, then
Let            $L_1 = \{G[(X_1^1 \cup \ldots \cup X_r^1) \cup (Y_1^2 \cup \ldots \cup Y_k^2)]\}$
               $L_2 = \{G[X_i^1 \cup (Y_i^1 \cup Y_1^2 \cup \ldots \cup Y_k^2)] : i = 1, \ldots r\}$
               $L_3 = \{G[(X_i^2 \cup X_1^1 \cup \ldots \cup X_r^1) \cup Y_i^2] : i = 1, \ldots k\}$
   If $r \neq 1$ or $k \neq 1$, then $L(\alpha) = L_1 \cup L_2 \cup L_3$ else $L(\alpha) = L_2 \cup L_3$
Else//$\alpha$ is a $P$-node//$L(\alpha) = L(\alpha_1) \cup L(\alpha_2)$
If $\alpha$ is the root of $T(G)$, then let $L(\alpha) = \{F_i = G[X_i \cup Y_i], i = 1, \ldots, s\}$
Let $s_t = \max\{\min(|X_1|, |Y_1|), \ldots, \min(|X_s|, |Y_s|)\}$ return $F_t$
```

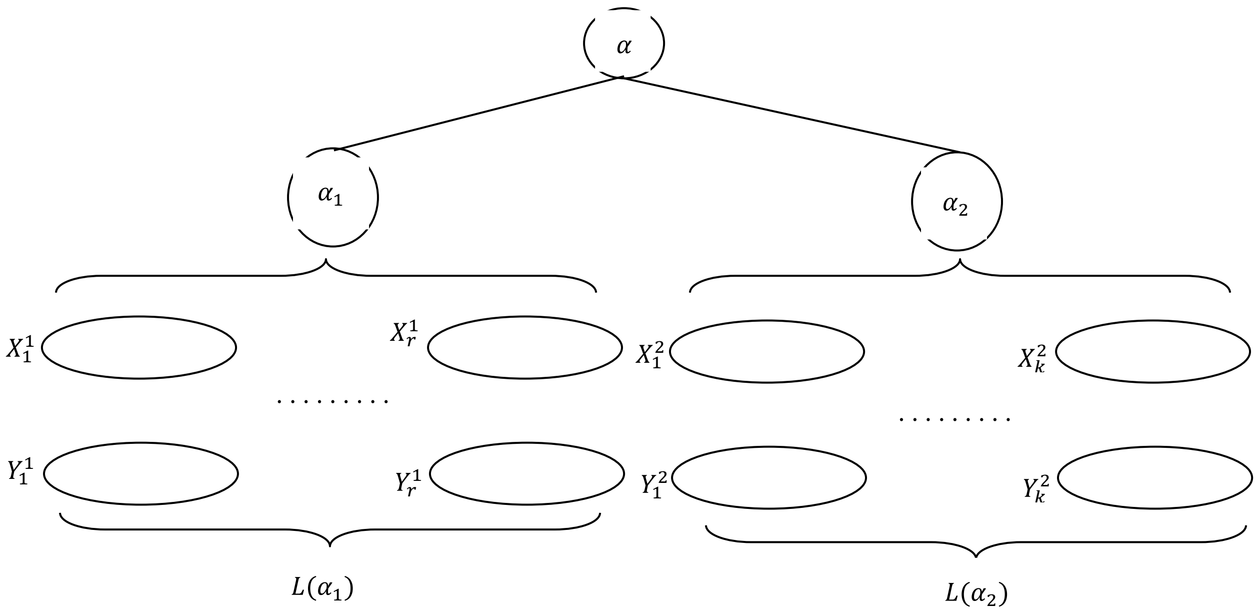

Figure 5. A node α and the sets of all maximal bicliques associated with its sons.

The number of bicliques computed for each internal node is at most $O(n^2)$. Since $T(G)$ contains an $O(n)$ node, the algorithm Balanced Bi-clique has a time complexity of $O(n^3)$.

4.2. Maximum Independent Set Problem

A subset S of the vertex set $V(G)$ in a graph G is called an independent set if any two vertices in S are not adjacent. The maximum independent set problem is the task of computing an independent set in G of maximum size. This problem is NP-complete for general graphs [14], but it can be solved in $O(n^{1.5}\sqrt{m/\log n})$ time complexity for a general bipartite graph [18]. This time complexity can be improved to $O(n)$ for a bipartite (P_6, C_6)-free graph using its canonical binary decomposition tree. The idea of our solution results from the structure of $K \bigoplus S$ graph G as follows: Let (V_1, V_2) be a $K \bigoplus S$ partition of the vertex set $V(G)$. By Property 1, every black vertex of V_1 is connected to every white vertex of V_2, and every white vertex of V_1 is independent of every black vertex of V_2. So, the maximum independent set in G is either the maximum independent set in $G[V_1]$, the maximum independent set in $G[V_2]$, or the independent set formed by the union of white vertices of $G[V_1]$ and black vertices of $G[V_2]$. This remark proves the correctness of Algorithm 5. Note that if G is not connected, then the maximum independent set in G is equal to the union of the maximum independent sets in its connected components.

Algorithm 5 Maximum Independent Set

```
Input: A binary canonical decomposition tree $T(G)$ of a bipartite $(P_6, C_6)$-free graph
$G = (B \cup W,\ E)$.
Output: A maximum independent set $S$ for $G$

Let $\alpha$ be a node on a post-order traversal of $T(G)$.
If $\alpha$ is a black vertex node $b$ (resp. a white vertex node $w$), then
$S(\alpha) = \{b\},\ B(\alpha) = \{b\},\ W(\alpha) = \varnothing$ (resp. $S(\alpha) = \{w\},\ B(\alpha) = \varnothing,\ W(\alpha) = \{w\}$)
Else let $\alpha_1$ and $\alpha_2$ be, respectively, the left and right son of $\alpha$ and let $S(\alpha_1)$ be a maximum
independent set of $G[\alpha_1]$ and $S(\alpha_2)$ be a maximum independent set of $G[\alpha_2]$; let
$W(\alpha_1),\ B(\alpha_1)$ be, respectively, the white and black vertices of $G[\alpha_1]$; and let
$W(\alpha_2),\ B(\alpha_2)$ be, respectively, the white and black vertices of $G[\alpha_2]$.
If $\alpha$ is a $K \bigoplus S$-node, then
$s = \max\{|S(\alpha_1)|,\ |S(\alpha_2)|,\ |W(\alpha_1) \cup B(\alpha_2)|\}$
$S(\alpha) = S$ where $|S| = s,\ W(\alpha) = W(\alpha_1) \cup W(\alpha_2)$ and $B(\alpha) = B(\alpha_1) \cup B(\alpha_2)$
else//$\alpha$ is a $P$-node
$S(\alpha) = S(\alpha_1) \cup S(\alpha_2),\ W(\alpha) = W(\alpha_1) \cup W(\alpha_2)$ and $B(\alpha) = B(\alpha_1) \cup B(\alpha_2)$
```

Since $T(G)$ contains an $O(n)$ node, the algorithm's maximum independent set has a time complexity of $O(n)$.

5. Conclusions

We have shown in this paper that bipartite (P_6, C_6)-free graphs can be recognized in linear time. Using this result, we solved two optimization graph problems in this class of graphs: the first is the maximum balanced biclique problem, and the second is the maximum independent set problem. An additional potential use of the canonical decomposition tree of a bipartite (P_6, C_6)-free graph is to solve the problem P∣ prec. $p_j = 1$∣ $C_{\max}$: Suppose there are n tasks with a unit execution time, and their order is constrained by a directed acyclic graph. Additionally, there are m machines of the same type. The goal is to discover a schedule that minimizes the makespan, which is the time when the final task in the graph finishes its execution. It is proved in [19] that this problem is NP-complete even if the precedence constraints form a bipartite graph of depth one. We conjecture that this problem can be solved in polynomial time for a bipartite (P_6, C_6)-free graph.

Author Contributions: Conceptualization, R.Q. and A.A.-Q.; methodology, R.Q. and A.A.-Q.; investigation, R.Q. and A.A.-Q.; writing—original draft preparation, R.Q. and A.A.-Q. All authors have read and agreed to the published version of the manuscript.

Funding: This research is funded by the Deanship of Scientific Research at Zarqa University/Jordan.

Data Availability Statement: Data are contained within the article.

Acknowledgments: The authors would like to thank the reviewers for their valuable suggestions and useful comments, which helped improve the presentation of the manuscript. The authors also would like to thank the Deanship of Scientific Research at Zarqa University.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Chakraborty, X.; Kumar, A.; Tomar, G. A survey on bipartite graph based recommender systems. *Inf. Process. Manag.* **2021**, *58*, 102536.
2. Wang, R.; Wu, Y.; Hu, X.; Liu, J. Bipartite graph neural networks for social recommendation. *Inf. Sci.* **2021**, *572*, 396–409. [CrossRef]
3. Biró, P.; Fleiner, T. Recent advances in matching algorithms. *Eur. J. Oper. Res.* **2022**, *294*, 745–769.
4. Fouquet, J.L.; Giakoumakis, V.; Vanherpe, J.M. Bipartite graphs totally decomposable by canonical decomposition. *Internat. J. Found. Comput. Sci.* **1999**, *10*, 513–533. [CrossRef]
5. Lozin, V.V. E-free bipartite graphs. *Discret. Anal. Oper. Res. Ser.* **2000**, *7*, 49–66. (In Russian)
6. Giakoumakis, V.; Vanherpe, J.-M. Bi-complement reducible graphs. *Adv. Appl. Math.* **1997**, *18*, 389–402. [CrossRef]
7. Giakoumakis, V.; Vanherpe, J.M. Linear time recognition and optimization for weak bisplit graphs, bi-cographs and bipartite P_6-free graphs. *Int. J. Found. Comput. Sci.* **2003**, *14*, 107–136. [CrossRef]
8. Quaddoura, R. Linear time recognition of bipartite $Star_{123}$-free graphs. *Int. Arab. J. Inf. Technol.* **2006**, *3*, 193–220.
9. Quaddoura, R.; Mansour, K. Classical graphs decomposition and their totally decomposable graphs. *IJCSNS Int. J. Comput. Sci. Netw. Secur.* **2010**, *10*, 240–250.
10. Corneil, D.G.; Perl, Y.; Stewart, L.K. A linear recognition algorithm for cographs. *SIAM J. Comput.* **1985**, *14*, 926–934. [CrossRef]
11. Al-Yamani, A.; Ramsundar, S.; Pradhan, D.K. A defect tolerance scheme for nanotechnology circuits. *IEEE Trans. Circuits Syst. I* **2007**, *54*, 2402–2409. [CrossRef]
12. Tahoori, M.B. Application-independent defect tolerance of reconfigurable Nano architectures. *ACM J. Emerg. Technol. Comput. Syst. (JETC)* **2006**, *2*, 197–218. [CrossRef]
13. Ravi, S.; Lloyd, E.L. The complexity of near-optimal programmable logic array folding. *SIAM J. Comput.* **1988**, *17*, 696–710. [CrossRef]
14. Garey, M.R.; Johnson, D.S. *Computers and Intractability, A Guide to the Theory of NP-Completeness*; Mathematical Series; Freeman: San Francisco, CA, USA, 1979.
15. Chen, L.; Liu, C.; Zhou, R.; Xu, J.; Li, J. Efficient Exact Algorithms for Maximum Balanced Biclique Search in Bipartite Graphs. In Proceedings of the SIGMOD '21: Proceedings of the 2021 International Conference on Management of Data, Toronto, ON, Canada, 15–18 June 2021; pp. 248–260.
16. Li, M.; Hao, J.-K.; Wu, Q. General swap based multiple neighborhood adaptive search for the maximum balanced biclique problem. *Comput. Oper. Res.* **2020**, *119*, 104922. [CrossRef]
17. Zhou, Y.; Hao, J.K. Tabu search with graph reduction for finding maximum balanced bicliques in bipartite graphs. *Eng. Appl. Artif. Intell.* **2019**, *77*, 86–97. [CrossRef]
18. Alt, H.; Blum, N.; Mehlhorn, K.; Paul, M. Computing a maximum cardinality matching in bipartite graphs in time $n^{1.5}\sqrt{m/\log n}$. *Inform. Process. Lett.* **1991**, *37*, 237–240. [CrossRef]
19. Bampis, E. The Complexity of short schedules for UET bipartite graphs. *RAIRO Oper. Res.* **1999**, *33*, 367–370. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.