Available online at www.sciencedirect.com
SCIENCE DIRECT
Discrete Applied Mathematics 131 (2003) 199–212
DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

Splitting a graph into disjoint induced paths or cycles

Hoàng-Oanh Le[a], Van Bang Le[a], Haiko Müller[b,*]

[a] Fachbereich Informatik, Universität Rostock, 18051 Rostock, Germany
[b] Institut für Informatik, Friedrich-Schiller-Universität Jena, 07740 Jena, Germany

Received 25 August 2000; received in revised form 05 November 2001; accepted 05 May 2002

Abstract

A graph is an $\mathscr{X}$-graph of $\mathscr{Y}$-graphs (or two-level clustered graph) if its vertices can be partitioned into subsets (called clusters) such that each cluster induces a graph belonging to the given class $\mathscr{Y}$ and the graph of the clusters belongs to another given class $\mathscr{X}$. Two-level clustered graphs are a useful and interesting concept in graph drawing.

We consider the complexity of recognizing two-level clustered graphs. We prove that, for a given integer $k \geqslant 2$, it is NP-complete to decide whether or not a graph is a path of length $k-1$ of paths (cycles). This solves a problem posed by Schreiber, Skodinis and Brandenburg. Similar reductions show that it is NP-complete to decide whether or not a graph is a k-star/k-clique of paths (cycles).

In contrast, we show that k-graphs of path (cycles) can be recognized in polynomial time when the inputs are restricted to graphs of bounded treewidth.

1. Introduction

Two-level clustered graphs have been introduced by Kratochvíl, Goljan and Kučera in a monograph on string graphs [11]. Later on Brandenburg considered this concept in connection with graph drawing [5]. Given two graph classes $\mathscr{X}$ and $\mathscr{Y}$, a graph G is called an *$\mathscr{X}$-graph of $\mathscr{Y}$-graphs* (or *two-level clustered graph*, or *$\mathscr{X}$-graphs of*

* Corresponding author.
E-mail address: hm@minet.uni-jena.de (H. Müller).

0166-218X/03/$ - see front matter
PII: S0166-218X(02)00425-0

Table 1
Known results

$\mathscr{X}$	$\mathscr{Y}$	Complexity	Reference
Planar graphs	$\{K_n{:}n \geqslant 1\}$	NP-complete	[11]
$\{K_k\},\ k \geqslant 3$	$\{K_n{:}n \geqslant 1\}$	NP-complete	[5]
$\{P_n{:}\ n \geqslant 1\}$	$\{K_n{:}n \geqslant 1\}$	$O(n^3)$	[5]
$\{C_n{:}n \geqslant 6\}$	$\{K_n{:}n \geqslant 1\}$	$O(n^2)$	[5]
Trees	$\{P_n{:}n \geqslant 1\}$	NP-complete	[13]
Trees	$\{C_n{:}n \geqslant 3\}$	NP-complete	[13]
Trees	$\{P_n{:}n \leqslant k\}$	$O(n^{2k+3})$	[13]
Trees	$\{C_n{:}n \leqslant k\}$	$O(n^{2k+3})$	[13]
$\{P_k\}, k \geqslant 2$	$\{P_n{:}n \geqslant 1\}$	NP-complete	This paper
$\{P_k\}, k \geqslant 2$	$\{C_n{:}n \geqslant 3\}$	NP-complete	This paper
$\{K_k\}, k \geqslant 2$	$\{P_n{:}n \geqslant 1\}$	NP-complete	This paper
$\{K_k\}, k \geqslant 2$	$\{C_n{:}n \geqslant 3\}$	NP-complete	This paper
$\{K_{1,k}\}, k \geqslant 1$	$\{P_n{:}n \geqslant 1\}$	NP-complete	This paper
$\{K_{1,k}\}, k \geqslant 1$	$\{C_n{:}n \geqslant 3\}$	NP-complete	This paper

$\mathscr{Y}$*-graphs*) if its vertex set can be partitioned into smaller sets (called *clusters*) such that

- every cluster induces a graph isomorphic to a member in $\mathscr{Y}$, and
- the graph G^* obtained from G by shrinking every cluster into a single vertex and replacing multiple edges then replaced by a single one is isomorphic to a graph in $\mathscr{X}$.

For examples, every $m \times n$-grid is a path of paths and every $2n \times 2n$-grid is a path of cycles. Two-level clustered graphs can be drawn according to their structure: On the top level draw the $\mathscr{X}$-graph G^* and on the second level the $\mathscr{Y}$-graphs (the clusters) of G. It is clear that drawing in this way will reflect the nature of the graph and its understanding; see [5,13] for more details.

In [5,13], the recognition problem of two-level clustered graphs has been discussed: Given two graph classes $\mathscr{X}$ and $\mathscr{Y}$. Is the graph G an $\mathscr{X}$-graphs of $\mathscr{Y}$-graphs? Table 1 summarizes known results.

In the time bounds n denotes the number of vertices of the graph considered. For an integer k a k-clique (k-path, k-cycle) is a clique (chordless path, chordless cycle) with k vertices. A k-star is a tree with k vertices and at least $k-1$ leaves. A k-clique (k-path, k-cycle, k-star) is also denoted by K_k (P_k, C_k, $K_{1,k-1}$).

It is interesting to compare the NP-completeness results for k-cliques of paths or cycles with the polynomial time algorithms paths or cycles of cliques. Note that paths or stars of l-paths or l-cycles can be recognized in polynomial time, see [13]. We do not know the complexity of recognizing cliques of l-paths or l-cycles.

The problem of recognizing paths of paths was posed in [13,6]. In Section 2 we prove the main result of this paper: Even the recognition of 2-paths of paths is NP-complete, see Theorem 1. Our other NP-completeness results mentioned in Table 1 follow by simple reductions.

In Section 3 we consider graphs on at most k vertices of paths or cycles.

2. NP-completeness proofs

We start with the basic theorem of this section: Recognizing 2-paths of paths is NP-complete.

Theorem 1. *It is* NP*-complete to decide whether or not the vertex set of a connected graph can be partitioned into two subsets each of which induces a path.*

Proof. Clearly, the problem belongs to NP. To prove the NP-completeness, we will reduce the NP-complete problem Not-All-Equal 3Sat (see [9]) to our problem.

Not-All-Equal 3Sat. Let $\mathscr{C}$ be a collection of m clauses over the set U of n Boolean variables such that every clause has exactly three variables. Is there a truth assignment satisfying $\mathscr{C}$ such that each clause in $\mathscr{C}$ has at least one true and at least one false literal?

Let $\mathscr{C}=\{C_1,\dots,C_m\}$ be a collection of m clauses with variable set $U=\{v_1,\dots,v_n\}$ such that every clause C_i of $\mathscr{C}$ contains exactly three literals, $C_i=\{c_{i,1},c_{i,2},c_{i,3}\}$, where each literal $c_{i,j}$ $(1\leqslant i\leqslant m, 1\leqslant j\leqslant 3)$ is either v_k or $\overline{v_k}$ for some suitable k. We will construct a graph $G=G(\mathscr{C})$ such that G is partitionable into two induced paths if and only if $\mathscr{C}$ is satisfiable such that each clause in $\mathscr{C}$ has at least one true and at least one false literal.

For each variable $v_k\in U$ let $G(k)$ be the graph shown in Fig. 1. For each clause $C_i=\{c_{i,1},c_{i,2},c_{i,3}\}$ let $G(C_i)$ be the graph shown in Fig. 2.

Let G be the graph consisting of all graphs $G(k)$, all graphs $G(C_i)$, and additional edges as follows:

- For all i,j and k: If the literal $c_{i,j}$ is $v\in\{v_k,\overline{v_k}\}$, then connect the vertices $c_{i,j}$ and v by an edge.
- Add edges $\{x_k,x_{k+1}\},\{x_k,y_{k+1}\},\{y_k,x_{k+1}\}$, and $\{y_k,y_{k+1}\}$ for all odd $k\geqslant 1$.
  Add edges $\{u_k,u_{k+1}\},\{u_k,w_{k+1}\},\{w_k,u_{k+1}\}$, and $\{w_k,w_{k+1}\}$ for all even $k\geqslant 2$.
- Add edges $\{z_{i,1},z_{i+1,1}\},\{z_{i,1},z_{i+1,2}\},\{z_{i,2},z_{i+1,1}\},\{z_{i,2},z_{i+1,2}\}$ for all even $i\geqslant 2$, and $\{z_{i,3},z_{i+1,3}\},\{z_{i,3},z_{i+1}\},\{z_i,z_{i+1,3}\},\{z_i,z_{i+1}\}$ for all odd $i\geqslant 1$.
- Add edges $\{u_1,z_{1,1}\},\{u_1,z_{1,2}\},\{w_1,z_{1,1}\}$, and $\{w_1,z_{1,2}\}$.

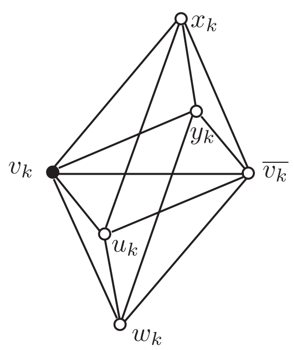

Fig. 1. The graph $G(k)$.

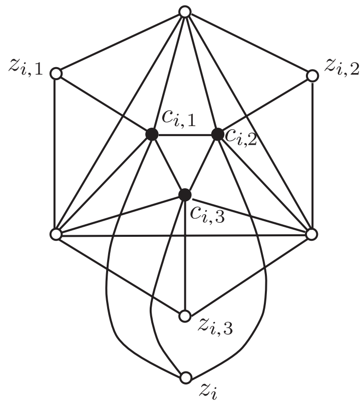

Fig. 2. The graph $G(C_i)$.

Suppose that G has a partition into two induced paths. Before giving a truth assignment for $\mathscr{C}$, we will make two claims. We assume that the vertices of G are colored black and white such that the vertices of each color induce a path.

Claim 1. *For every k, the labeled vertices v_k and $\overline{v_k}$ in $G(k)$ belong to different paths.*

Proof. Otherwise, the 4-cycle induced by the four vertices of $G(k) - \{v_k, \overline{v_k}\}$ would belong to the same path, a contradiction. □

Claim 2. *If $v \in \{v_k, \overline{v_k}\}$ is adjacent to $c_{i,j}$, then v and $c_{i,j}$ belong to different paths.*

Proof. Without loss of generality, assume that v_k and $c_{i,j}$ are adjacent. Suppose to the contrary that v_k and $c_{i,j}$ belong to the same path, say both are black. Then, by Claim 1, $\overline{v_k}$ is white. Since $c_{i,j}$ is black, at most one vertex of the 4-cycle induced by the vertices of $G(k) - \{v_k, \overline{v_k}\}$ is black. Otherwise, the black vertex v_k would have three black neighbors. But then the white vertex $\overline{v_k}$ has three white neighbors in $G(k) - \{v_k, \overline{v_k}\}$, a contradiction. □

We define the truth assignment b for $\mathscr{C}$ as follows: if v_k is black then $b(v_k)$=true else $b(v_k) =$ false. By Claim 1, b is well defined. For each clause C_i, one of the labeled vertices $c_{i,1}, c_{i,2}, c_{i,3}$ is black and one of them is white, because they are pairwise adjacent in G. By Claim 2, one of the corresponding neighbors in $U \cup \{\bar{v}\colon v \in U\}$ of $c_{i,j}$ is white, one is black. Thus, one of the literals in C_i is true and one is false by the assignment b.

Suppose that $\mathscr{C}$ is satisfied such that every clause has one true and one false literal. First, color the labeled vertices $c_{i,j}$ in each $G(C_i)$ black if the corresponding literal $c_{i,j}$ in C_i is true; otherwise white. By assumption, for each i, at least one of $c_{i,1}, c_{i,2}, c_{i,3}$ is black and at least one of them is white.

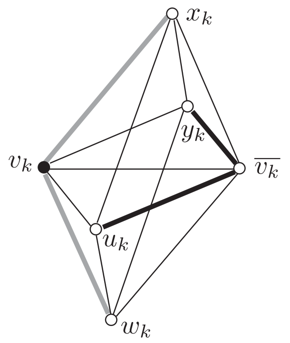

Fig. 3. Two paths partition of $G(k)$ giving colors for $v_k, \overline{v_k}$.

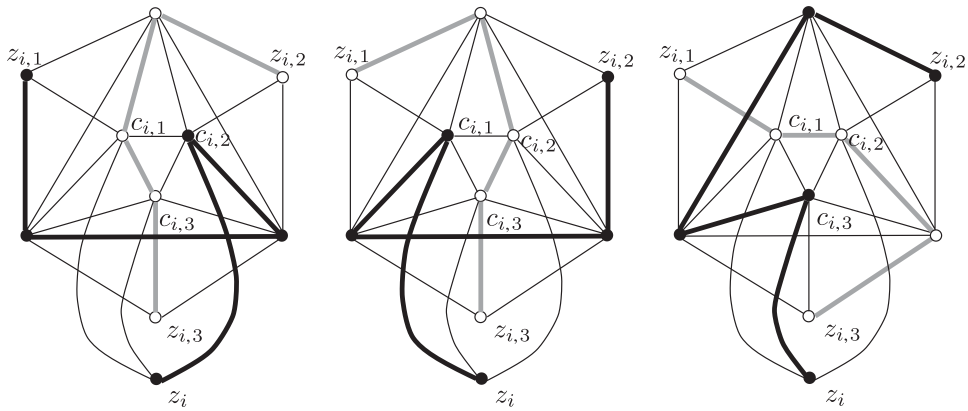

Fig. 4. Two paths partition of $G(C_i)$ giving colors for $c_{i,1}, c_{i,2}, c_{i,3}$.

If $c_{i,j}$ is adjacent to $v \in \{v_k, \overline{v_k}\}$, then color v black (white, respectively), according to whether $c_{i,j}$ is white (black, respectively). Note that for each k, v_k and $\overline{v_k}$ are differently colored.

Next, extend the black–white coloring in each $G(k)$ and each $G(C_i)$ as indicated in Figs. 3 and 4.

More precisely, in each $G(k)$, x_k and w_k receive the same color as v_k, and u_k and y_k receive the same color as $\overline{v_k}$. In particular, there is exactly one edge between $G(k)$ and $G(k+1)$ with two black endvertices and exactly one edge with two white endvertices. Thus, all vertices of the same color in $\bigcup_k G(k)$ induce a path with one endvertex in $\{u_1, w_1\}$.

For each i, the black–white coloring of $\{c_{i,1}, c_{i,2}, c_{i,3}\}$ can be extended for the whole graph $G(C_i)$ such that the vertices of the same color induce a path with one endvertex in $\{z_{i,1}, z_{i,2}\}$ and the other endvertex in $\{z_i, z_{i,3}\}$. In particular, there is exactly one edge between $G(C_i)$ and $G(C_{i+1})$ with two black endvertices and exactly one edge with two white endvertices. Thus, all vertices of the same color in $\bigcup_i G(C_i)$ induce a path with one endvertex in $\{z_{1,1}, z_{1,2}\}$.

Since every vertex in $\{u_1, w_1\}$ is adjacent to every vertex in $\{z_{1,1}, z_{1,2}\}$ and the $c_{i,j}$'s and their neighbors in $\bigcup_k G(k)$ are differently colored, it follows that all vertices of the same color in G induce a path. □

Next, we prove that recognizing 2-paths of cycles is NP-complete.

Theorem 2. *It is* NP-*complete to decide whether or not the vertex set of a connected graph can be partitioned into two subsets each of which induces a cycle.*

Proof. We modify the construction in the proof of Theorem 1 in an obvious way. Given an instance $\mathscr{C}$ of Not-All-Equal 3Sat, let $G(k)$ and $G(C_i)$ be the graphs in Figs. 1 and 2. Recall that n and m denote the number of variables and clauses, respectively. Define the vertices p_n, q_n, s_m, and t_m as follows:

If n is even, let $p_n := u_n$, $q_n := w_n$ and if n is odd, let $q_n := x_n$, $q_n := y_n$.

If m is even, let $s_m := z_{m,1}$, $t_m := z_{m,2}$ and if m is odd, let $s_m := z_{m,3}$, $t_m := z_m$.

Now, let G be the graph constructed in the proof of Theorem 1, and let G' be the graph obtained from G by adding the edges $\{p_n, s_m\}, \{p_n, t_m\}, \{q_n, s_m\}$, and $\{q_n, t_m\}$. The arguments in proving Theorem 1 show that G' can be partitioned into two induced cycles if and only if $\mathscr{C}$ is satisfied such that each clause has at least one true and at least one false literal. □

In the following subsection we complete the proof of Theorem 3 stated at the end of this section. In all these subsections, let G be a connected graph, and let $k \geqslant 2$ be a fixed integer. Let v_0 be an arbitrary vertex of G.

2.1. *Paths of paths (cycles)*

Construct a graph $H = H(G,k)$ from G and $2k-4$ new vertices a_i, b_i $(1 \leqslant i \leqslant k-2)$, and new edges $\{v_0, a_1\}, \{v_0, b_1\}, \{a_i, a_{i+1}\}$ for $1 \leqslant i \leqslant k-3$, $\{a_i, b_i\}$ for $1 \leqslant i \leqslant k-2$, and $\{a_{i-1}, b_i\}$ for $2 \leqslant i \leqslant k-2$, see Fig. 5.

Clearly, if G is a 2-path of paths, H is a k-path of paths consisting of the two paths in G and the $k-2$ paths $a_1b_1, a_2b_2, \ldots, a_{k-2}b_{k-2}$.

Conversely, assume that H is a k-path of paths. If a_{k-2} and b_{k-2} belong to the same cluster in H, then $\{a_{k-2}, b_{k-2}\}$ is a cluster and corresponds to an endvertex of

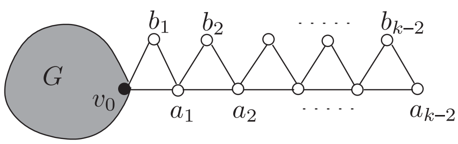

Fig. 5. Path of paths.

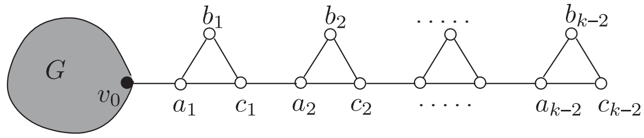

Fig. 6. Path of cycles.

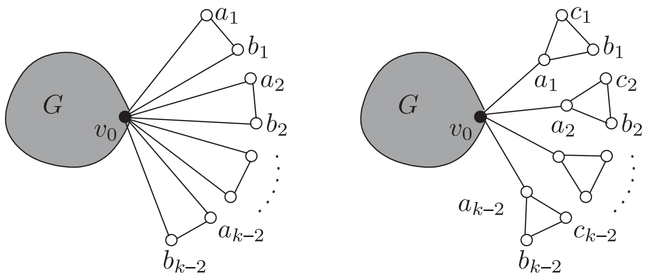

Fig. 7. Star of paths (left) and of cycles (right).

the k-path H^*. Thus, $H - \{a_{k-2}, b_{k-2}\} = H(G, k-1)$ is a $(k-1)$-path of paths, and it follows, by induction on k, that G is a 2-path of paths.

If a_{k-2} and b_{k-2} belong to different clusters in H, then a_{k-3} and a_{k-2}, or a_{k-3} and b_{k-2} must belong to the same cluster; say a_{k-3} and a_{k-2} belong to cluster C. Then $\{b_{k-2}\}$ is a cluster and corresponds to an endvertex of the k-path H^*. Moreover, a_{k-2} is one endvertex of the path in H induced by cluster C, hence $C - \{a_{k-2}\}$ induces a (nonempty) path in H. Thus, $H - \{a_{k-2}, b_{k-2}\} = H(G, k-1)$ is a $(k-1)$-path of paths. Again, by induction on k, G is a 2-path of paths.

We now consider the construction for paths of cycles. Let H be the graph obtained from G by adding $3k-6$ new vertices a_i, b_i, c_i, $1 \leqslant i \leqslant k-2$, and edges $\{a_1, v_0\}, \{a_i, c_i\}, \{a_i, b_i\}, \{b_i, c_i\}$ for $1 \leqslant i \leqslant k-2$, see Fig. 6.

It is easy to see that G is a 2-path of cycles if and only if H is k-path of cycles (consisting of the two cycles in G and the $k-2$ triangles $a_1b_1c_1, \dots, a_{k-2}b_{k-2}c_{k-2}$).

2.2. Stars of paths (cycles)

Construct a graph $H_1 = H_1(G, k)$ from G and $2k-4$ new vertices a_i, b_i, and edges $\{a_i, v_0\}, \{b_i, v_0\}, \{a_i, b_i\}$ for $1 \leqslant i \leqslant k-2$. Construct a graph $H_2 = H_2(G, k)$ from G and $3k-6$ new vertices a_i, b_i, c_i, and edges $\{a_i, v_0\}, \{a_i, b_i\}, \{a_i, c_i\}$ and $\{b_i, c_i\}$ for $1 \leqslant i \leqslant k-2$, see Fig. 7.

Obviously G is a 2-path of cycles if and only if H_2 is a k-star of cycles, consisting of the two cycles in G and the $k-2$ triangles $a_1b_1c_1, \dots, a_{k-2}b_{k-2}c_{k-2}$.

We discuss now the case of k-stars of paths. Assume first that G is a 2-path of paths. Then, H_1 is a k-star of paths, consisting of the two paths in G and the $k-2$ paths $a_1b_1, \dots, a_{k-2}b_{k-2}$. Where, the center vertex of the star H_1^* corresponds to the path in G that contains the vertex v_0.

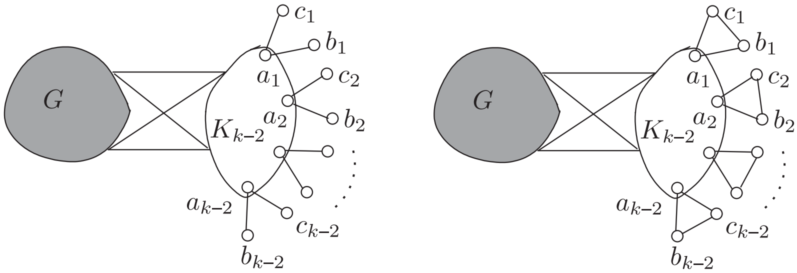

Fig. 8. Clique of paths (left) and of cycles (right).

Conversely, assume that H_1 is a k-star of paths. If a_{k-2} and b_{k-2} belong to the same cluster in H_1, then $\{a_{k-2}, b_{k-2}\}$ is a cluster in H_1 and corresponds to a leaf of the k-star H_1^*. Thus, $H_1 - \{a_{k-2}, b_{k-2}\} = H_1(G, k-1)$ is a $(k-1)$-star of paths. By induction on k it follows that G is a 2-path of paths.

Conversely, if a_{k-2} and b_{k-2} belong to different clusters of H_1, then a_{k-2} and v_0, or b_{k-2} and v_0 must belong to the same cluster; say a_{k-2} and v_0 belong to some cluster C. Then $\{b_{k-2}\}$ is a cluster and corresponds to a leaf of the k-star H_1^*. Moreover, a_{k-2} is an endvertex of the path in H_1 induced by C, hence $C - \{a_{k-2}\}$ induces a path in H_1. Thus, $H_1 - \{a_{k-2}, b_{k-2}\} = H_1(G, k-1)$ is a $(k-1)$-star of paths. Again, by induction on k, G is a 2-path of paths.

2.3. Cliques of paths (cycles)

Construct a graph $H_1 = H_1(G, k)$ from G and $3k-6$ new vertices a_i, b_i, c_i, and edges $\{a_i, b_i\}, \{a_i, c_i\}, \{a_i, v\}$ for $1 \leqslant i \leqslant k-2$ and $v \in G$, as well as $\{a_i, a_j\}$ for $i \neq j$. Thus, every vertex of G is adjacent to every a_i, and the a_i's form a complete graph K_{k-2}. Let $H_2 = H_2(G, k)$ be the graph obtained from H_1 by adding the edges $\{b_i, c_i\}$ for $1 \leqslant i \leqslant k-2$, see Fig. 8.

It is easy to see that G is a 2-path of cycles if and only if H_2 is a k-clique of cycles, consisting of the two cycles in G and the $k-2$ triangles $a_1b_1c_1, \ldots, a_{k-2}b_{k-2}c_{k-2}$.

Assume that G is a 2-path of paths. Then, clearly, H_1 is a k-clique of paths, consisting of the two paths in G and the $k-2$ paths $b_1a_1c_1, \ldots, b_{k-2}a_{k-2}c_{k-2}$.

Conversely, assume that H_1 is a k-clique of paths. If, for some $1 \leqslant i \leqslant k-2$, b_i and c_i belong to different clusters in H_1, then b_i and a_i, or c_i and a_i must belong to the same cluster; say b_i and a_i belong to the same cluster. Then $\{c_i\}$ is a cluster. But this is impossible, because H_1^* is a clique.

Thus, for all i, b_i and c_i belong to the same cluster. Then $\{a_i, b_i, c_i\}$ is a cluster for each i. It follows that G must consist of two clusters. That is, G is a 2-path of paths.

We summarize the results of this section in the following theorem.

Theorem 3. *Let $k \geqslant 2$ be a fixed integer. It is* NP*-complete to decide whether or not a given graph is a $\mathscr{X}$-graph of $\mathscr{Y}$-graphs for $\mathscr{X} \in \{k\text{-paths}, k\text{-stars}, k\text{-cliques}\}$* and *$\mathscr{Y} \in \{paths, cycles\}$.*

3. Related graph partitioning problems

Let a k-graph be a graph on k vertices. By $\pi(G)$ and $\zeta(G)$ we denote the smallest integer k such that G is a k-graph of paths and cycles, respectively. For every k we define the problems

$$k\mathsf{PP} = \{G\colon\ \pi(G) \leqslant k\}, \quad \mathsf{PP} = \{(G,k)\colon\ \pi(G) \leqslant k\},$$

$$k\mathsf{CP} = \{G\colon\ \zeta(G) \leqslant k\}, \quad \mathsf{CP} = \{(G,k)\colon\ \zeta(G) \leqslant k\}.$$

The problems kPP and kCP are NP-complete for $k = 2$ by Theorem 3. Hence, kPP and kCP are NP-complete for all fixed $k \geqslant 2$: For all graphs G, G belongs to 2PP (2CP) if and only if $G \cup (k-2)P_3$ (respectively, $G \cup (k-2)C_3$) belongs to kPP (respectively, kCP).

Theorem 4. PP is NP-complete, even when restricted to bipartite graphs.

Proof. Clearly PP is in NP. We prove the completeness by a reduction of the Hamiltonian Path problem HP. Let $G = (V,E)$ be an instance of HP and let U be a set of cardinality $|U| = 2(1 + |E| - |V|)$ disjoint from $V \cup E$. We define a bipartite graph $B = (E, U \cup V, F)$ by $F = \{\{e,u\}\colon\ e \in E \wedge u \in U\} \cup \{\{e,v\}\colon\ e \in E \wedge v \in e\}$. Obviously, $G \in$ HP if and only if $(B, 2 + |E| - |V|) \in$ PP. □

Note that the related problem "Can G be partitioned into k disjoint forests?" is NP-complete ([9] Problem [GT14]).

Also, the problem "Can G be partitioned into k disjoint trees?" is NP-complete, as pointed out by A. Brandstädt: A graph G can be partitioned into a stable set and a tree if and only if the graph G', obtained from G by adding a new vertex adjacent to all vertices in G, can be partitioned into two trees. Partitioning a graph into a stable set and a tree is NP-complete [7]. Finally, G is a 2-graph of trees if and only if $G \cup (k-2)K_{1,3}$ is a k-graph of trees.

In the last section we will show that the parameters $\zeta(G)$ and $\pi(G)$ can be determined in polynomial time for graphs of bounded treewidth.

3.1. Tree-decomposition

The notion of *treewidth* was introduced by Robertson and Seymour [12] via tree-decompositions.

Definition A. A pair $(\mathscr{X}, T)$ is a *tree-decomposition* of a graph $G = (V,E)$ if $\mathscr{X} = \{X_i\colon\ i \in I\}$ is a set of subsets of V and $T = (I,F)$ is a tree such that

- $\bigcup_{i \in I} X_i = V$,
- $\forall e \in E\ \exists i \in I,\ e \subseteq X_i$,
- $\forall v \in V,\ T[\{i\colon\ v \in X_i\}]$ is connected.

The *width* of a tree-decomposition $(\mathscr{X}, T)$ is $\max\{|X_i|: i \in I\} - 1$, and the *treewidth* $\mathrm{tw}(G)$ of G is the minimum width of a tree-decomposition of G. The decision problem $\{(G,k): \mathrm{tw}(G) \leqslant k\}$ is NP-complete [2]. However, for every integer k, there is a linear time algorithm computing either a tree-decomposition of width at most k of the input graph G or states $\mathrm{tw}(G) > k$ [4].

Many NP-complete problems become polynomial or even linear when the inputs are restricted to graphs of bounded treewidth [1,3], among them all problems that can be stated in monadic second order logic (MSOL) [8].

For every fixed k, the problems kPP and kCP are solvable in linear time when the inputs are restricted to graphs of bounded treewidth, because both problems can be expressed in MSOL. In the following we present dynamic programming algorithms computing the parameters $\pi(G)$ and $\zeta(G)$ of graphs G with $tw(G) \leqslant k$. For simplicity we restrict ourselves to nice tree-decompositions.

Definition. A tree-decomposition $(\mathscr{X}, T)$ is *nice* if T has a root r such that all nodes of T have at most two children and

- if $i \in I$ has no children, then i is called *start node* and $X_i = \emptyset$.
- If $i \in I$ has exactly one child j, then i is either an *introduce node* of T, i.e., $X_i = X_j \cup \{v\}$ for a vertex $v \in V \setminus X_j$, or i is a *forget node* of T, i.e., $X_i = X_j \setminus \{v\}$ for a vertex $v \in X_j$.
- If $i \in I$ has two children j_1 and j_2, then i is a *join node* and $X_i = X_{j_1} = X_{j_2}$.

Moreover, the root r is a forget node of T with $X_r = \emptyset$.

It is known that every graph G with $\mathrm{tw}(G) \leqslant k$ admits a nice tree-decomposition of width at most k such that $|I| = \mathrm{O}(k|V(G)|)$, which can be obtained from an arbitrary tree-decomposition in linear time [10].

3.2. Characteristic

Let $G=(V,E)$ be an arbitrary graph. We call a set $\mathscr{P}$ of pairwise disjoint sets $S \subseteq V$ a *packing* if each set $S \in \mathscr{P}$ induces either a chordless path or a chordless cycle in G. The packing $\mathscr{P}$ is a *partition* if $\bigcup \mathscr{P} = V$.

Let $\mathscr{P}$ be a fixed packing of G. For every vertex $v \in \bigcup \mathscr{P}$ let $\langle\rangle$ denoe the set $S \in \mathscr{P}$ with $v \in S$. For two sets $S, S' \subseteq V$ let $[S,S'] = \{\{s,s'\} \in E: s \in S \wedge s' \in S'\}$.

A system of representatives of $\mathscr{P}$ is a set $R \subseteq \bigcup \mathscr{P}$ such that $\forall S \in \mathscr{P}.\ |S \cap R| = 1$. The *shrink graph* of $\mathscr{P}$ is a graph (R,F) where R is a system of representatives of $\mathscr{P}$ and $F = \{\{s,t\}: [\langle s\rangle, \langle t\rangle] \neq \emptyset\}$. If $H = (R,F)$ is a shrink graph we define $H - v$ and $H + v$ for all vertices $v \in R$ and $v \in V \setminus R$, respectively, by

$$H - v = (R \setminus \{v\}, F \setminus \{\{v,w\}: w \in R\}),$$

$$H + v = (R \cup \{v\}, F \cup \{\{v,w\}: w \in R \wedge [\langle v\rangle, \langle w\rangle] \neq \emptyset\}).$$

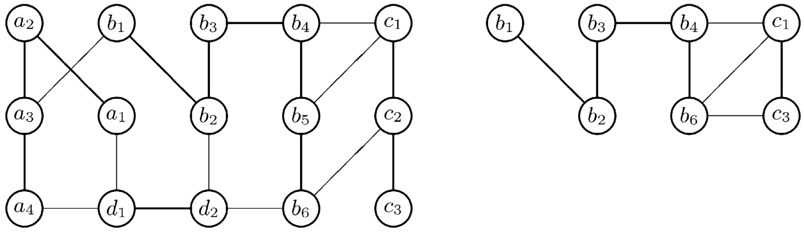

Fig. 9. A partition of G_i into paths and the graph H of a characteristic.

For a set $X \subseteq V$ we define the packings $\mathscr{P} - X$ and $\mathscr{P} + X$ by

$$\mathscr{P} - X = \bigcup_S \{U\colon\ U \text{ induces a connected component of } G[S \setminus X]\},$$

$$\mathscr{P} + X = \{\{x\}\colon\ x \in X\} \cup (\mathscr{P} - X),$$

where the union is taken over all sets $S \in \mathscr{P}$ with $S \cap X \neq \emptyset$.

Now we fix a nice tree-decomposition $(\mathscr{X}, T)$ of G. For two nodes i and j of T we write $j \leqslant i$ if j belongs to the subtree of T rooted at i, i.e., the path from the root r to j in T passes through i. For all $i \in I$ let G_i be the subgraph of G induced by the vertices in $V_i = \bigcup_{j \leqslant i} X_j$.

Our dynamic programming algorithms work bottom–up in the tree-decomposition, i.e., from the start nodes of T to the root r. For each $i \in I$ we consider a set of partitions of G_i.

Let $\mathscr{P}$ be a partition of G_i. Let H be the graph obtained from G_i by the following steps:

- remove all vertices in sets $S \in \mathscr{P}$ such that $G[S]$ is a cycle,
- remove all vertices in sets $S \in \mathscr{P}$ such that the path $G[S]$ has no endpoint in X_i,
- shrink all edges $\{x, y\}$ with $\langle x \rangle = \langle y \rangle$ and $\{x, y\} \cap X_i = \emptyset$.

The *characteristic* of $\mathscr{P}$ is a triple (s, H, M) where $s = |\mathscr{P}|$ is the size of the partition (the number of sets in $\mathscr{P}$) and M is the set of edges $\{x, y\}$ of H such that $\langle x \rangle = \langle y \rangle$. More formally, let X be the set of vertices $x \in X_i$ such that $G[\langle x \rangle]$ is a path with one endpoint in X_i, and Y be a system of representatives of $\mathscr{P} - X$. Then $H = (X \cup Y, F)$ is the shrink graph of $\mathscr{P} + X$. Note that $|Y| \leqslant |X|$ and $|H[X] = G[X]$. Furthermore, $H[Y]$ is the shrink graph of $\mathscr{P} - X$.

Example. Let $G_i = (V_i, E_i)$ be the graph given in Fig. 9. We consider the following partition $\mathscr{P} = \{A, B, C, D\}$ of $V_i = A \cup B \cup C \cup D$:

$$A = \{a_1, a_2, a_3, a_4\}, \quad C = \{c_1, c_2, c_3\},$$

$$B = \{b_1, b_2, b_3, b_4, b_5, b_6\}, \quad D = \{d_1, d_2\}$$

and assume $X_i = \{a_2, b_1, b_3, b_4, c_1\}$. This implies $X = \{b_1, b_3, b_4, c_1\}$ and we may choose $Y = \{b_2, b_6, c_3\}$. Hence the triple $(4, H, \{\{b_1, b_2\}, \{b_2, b_3\}, \{b_3, b_4\}, \{b_4, b_6\}, \{c_1, c_3\}\})$ is a characteristic of $\mathscr{P}$.

The connected components of $(X \cup Y, M)$ correspond with a partition of H. Moreover, for all $x, y \in X \cup Y$ we have $[\langle x\rangle, \langle y\rangle] \neq \emptyset$ for this partition of H if and only if this holds for the partition of $\mathscr{P}$ of G_i.

For every $i \in I$ we define a preorder $\leqslant_i$ on the set of characteristics of partitions of G_i by $(s, H, M) \leqslant_i (s', H', M')$ if and only if

- the sizes of the partitions fulfill $s \leqslant s'$,
- H and H' coincide when restricted to X_i, i.e. $V(H) \cap X_i = V(H') \cap X_i$ and $E(H) \cap \binom{X_i}{2} = E(H') \cap \binom{X_i}{2}$,
- M and M' coincide when restricted to X_i, that is $M \cap \binom{X_i}{2} = M' \cap \binom{X_i}{2}$, and
- H is isomorphic to a subgraph of H'. More formally there is an injection ϕ: $Y \to Y'$, where $Y = V(H) \setminus X_i$ and $Y' = V(H') \setminus X_i$, such that $\{x, y\} \in M$ implies $\{x, \phi(y)\} \in M'$ and $\{y, z\} \in E(H)$ implies $\{\phi(y), \phi(z)\} \in E(H')$ for all $x \in X,\ y, z \in Y$.

Note that in case $V(H) = V(H')$ and $M = M'$ we have $(s, H, M) \leqslant_i (s', H', M')$ if and only if $s \leqslant s'$.

3.3. Algorithm

For each $i \in I$ we compute a set of characteristics of G_i using the characteristics stored for the children of i. We do not distinguish between equivalent characteristics and store for $i \in I$ only the $\leqslant_i$-minimal characteristics.

If we compute $\pi(G)$ then each item stored for $i \in I$ characterizes a partition $\mathscr{P}$ such that S induces a path in G_i for all $S \in \mathscr{P}$. If we compute $\zeta(G)$ then $G_i[S]$ is either a cycle or a path with both endpoints in X_i for all S in the partition.

If i is a *start node*, then $V_i = \emptyset$ has only one partition. Hence we store the characteristic $(0, (\emptyset, \emptyset), \emptyset)$.

Next let i be an *introduce node* with child j and $v \in X_i \setminus X_j$. We consider subcases.

- $\{v\}$ *becomes an additional set in the partition*: For each characteristic (s, H, M) stored for j we create a characteristic $(s + 1, H + v, M)$.
- v *prolongs an existing path*: For each characteristic (s, H, M) stored for j such that v prolongs a path in the partition defined by M and each vertex $x \in X_i \cap V(H)$ such that $N(v) \cap \langle x\rangle = \{x\}$ we create a characteristic $(s, H + v, M \cup \{\{v, x\}\})$.
- v *connects two existing paths*: For each characteristic (s, H, M) stored for j such that v connects two paths in the partition defined by M to one path and each pair of vertices $x, z \in X_i \cap V(H)$ such that $N(v) \cap \langle x\rangle = \{x\}$, $N(v) \cap (z) = \{z\}$ and $[\langle x\rangle, \langle z\rangle] = \emptyset$ we create a characteristic $(s - 1, H', M')$, where $H' = H + v$ and $M' = M \cup \{\{x, v\}, \{v, z\}\}$. If the other endpoints of the paths induced by $\langle x\rangle$ and $\langle z\rangle$ do not belong to X_i let $H' = H \setminus (\langle x\rangle, \langle z\rangle)$ and $M' = M \cap E(H')$. Note that this only applies if we compute $\pi(G)$.
  Otherwise let $H' = H + v$ and $M' = M \cup \{\{x, v\}, \{v, z\}\}$.
- v *closes a cycle* (*in case we compute* $\zeta(G)$ *only*): For each characteristic (s, H, M) stored for j such that v closes a path in the partition defined by M to a cycle and

each pair of vertices $x, z \in X_i \cap V(H)$ such that $N(v) \cap \{x, z\}$ we create a characteristic (s, H', M') where $H' = H \setminus (\langle x \rangle \cup \langle z \rangle)$ and $M' = M \cap E(H')$.

Now let i be a *forget node* with child j and $v \in X_j \setminus X_i$. Again we distinguish between subcases.

- *v is an inner vertex of a path or cycle*: For each characteristic (s, H, M) stored for j such that v is incident with two edges $\{v, u\}, \{v, w\} \in M$ we create the characteristic $(s, H - v, M \setminus \{\{v, u\}, \{v, w\}\})$.
  For each characteristic (s, H, M) stored for j such that $v \notin V(H)$ we keep (s, H, M).
- *v is an endpoint of a path* (*in case we compute $\pi(G)$ only*): For each characteristic (s, H, M) stored for j such that v is incident with at most one edge $\{v, w\} \in M$ we create the characteristic $(s, H - v, M \setminus \{\{v, w\}\})$.

Finally, let i be a *join node* with children j_1 and j_2. For each characteristic (s_1, H_1, M_1) stored for j_1 and each characteristic (s_2, H_2, M_2) stored for j_2, with $H_\imath = (X_\imath \cup Y_\imath, F_\imath)$, $\imath = 1, 2$, we compute the graph $H' = (X' \cup Y_1 \cup Y_2, F_1 \cup F_2)$ if $X_1 = X' = X_2$. If furthermore each connected component of $(V(H'), M_1 \cup M_2)$ is either a cycle (in case we compute $\zeta(G)$) or a path, we compute the characteristic (s', H, M) of this graph and $s = s_1 - s_1' + s_2 - s_2' + s'$ where $s_\imath'$ is the number of connected components of $(V(H_\imath), M_\imath)$, $\imath = 1, 2$. Note that this implies $d_{H_1}(x) + d_{H_2}(x) \leqslant 2$ for all $x \in X'$. This way we computed a set of characteristics (s, H, M) for node i, and we store the minimal ones among them.

Our algorithm stops when the minimal characteristics for the root r are stored. Observe that there is exactly one such minimal characteristic, namely $(\pi(G), (\emptyset, \emptyset), \emptyset)$ and $(\zeta(G), (\emptyset, \emptyset), \emptyset)$, respectively.

We can prove the *correctness* of our algorithms by induction. Therefore, it suffices to show that for each partition $\mathscr{P}$ of G_i we store either the characteristic of $\mathscr{P}$ or another characteristic (s, H, M) representing a better partition. Thereby better means either less cardinality or more possible extensions at the parent node, more precisely less with respect to $\leqslant_i$. We leave out the technical details here.

To bound the *running time* of the algorithms we consider a largest set of pairwise incomparable characteristics. First we observe $|X_i| \leqslant k+1$ for all $i \in I$ because $tw(G) \leqslant k$. This implies $|Y| \leqslant |X| \leqslant |X_i| \leqslant k+1$ for each entry (s, H, M) with $X = V(H) \cap X_i$ and $Y = V(H) \setminus X_i$. For each $X \subseteq X_i$ the graph $H[X] = G[X]$ is fixed. For fixed subsets $X \subseteq X_i$ and $Y \subseteq V_i \setminus X_i$ exist at most $2^{2(k+1)}$ different sets $[X, Y]$ and at most $2^{\binom{k+1}{2}}$ different sets $[Y, Y]$. Similarly, the number of possible sets M are bounded by $2^{\binom{k+1}{2}}$ if X and Y are given. Hence there exist at most $O(2^{(k+4)(k+1)})$ pairwise incomparable characteristics for each node $i \in I$.

The costliest steps of our algorithms are for join-nodes because we have to consider up to $O(2^{2(k+4)(k+1)})$ pairs of characteristics. Since I sizes $O(kn)$ for $|V| = n$ we can bound the overall running time by $O(2^{2(k+4)(k+1)}kn)$.

Theorem 5. *The parameters $\pi(G)$ and $\zeta(G)$ are computable in linear time for graphs of bounded treewidth.*

Acknowledgements

We thank Andreas Brandstädt for his communication [6] to the problem considered in Theorem 1, Konstantinos Skodinis for pointing out to us the Refs. [5,13], and an anonymous referee for lots of useful hints.

References

[1] S. Arnborg, Efficient algorithms for combinatorial problems on graphs with bounded decomposability—a survey, BIT 25 (1985) 2–23.

[2] S. Arnborg, D.G. Corneil, A. Proskurowski, Complexity of finding embeddings in a k-tree, SIAM J. Algebraic Discrete Methods 8 (1987) 277–284.

[3] S. Arnborg, J. Lagergren, D. Seese, Easy problems for tree-decomposable graphs, J. Algorithms 12 (1991) 308–340.

[4] H.L. Bodlaender, A linear time algorithm for finding tree-decomposition s of small treewidth, SIAM J. Comput. 25 (1996) 1305–1317.

[5] F.J. Brandenburg, Graph clustering I: Cycles of cliques, in: G. Di Battista (Ed.), Proceedings of the Fifth International Symposium on Graph Drawing, GD'97, Lecture Notes in Computer Science, Vol. 1353, 1997, pp. 158–168.

[6] F.J. Brandenburg, Open problem: partitioning a graph into two induced paths, personal communication to A. Brandstädt, 1999.

[7] A. Brandstädt, V.B. Le, Th. Szymczak, On the complexity of some problems related to GRAPH3-COLORABILITY, Discrete Appl. Math. 89 (1998) 59–73.

[8] B. Courcelle, The monadic second-order logic of graphs I: recognizable sets of finite graphs, Inform. Comput. 85 (1990) 12–75.

[9] M. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-completeness, W.H. Freeman, San Fransisco, 1979.

[10] T. Kloks, Treewidth—Computations and Approximations, in: Lecture Notes in Computer Science, Vol. 842, Springer, Berlin, 1994.

[11] J. Kratochvíl, M. Goljan, P. Kučera, String Graphs, Academia, Prague, 1986.

[12] N. Robertson, P.D. Seymour, Graph minors. II. Algorithmic aspects of tree-width, J. Algorithms 7 (1986) 309–322.

[13] F. Schreiber, K. Skodinis, NP-completeness of some tree-clustering problems, in: S.H. Whitesides (Ed.), Proceedings of the Sixth International Symposium on Graph Drawing, GD'98, Lecture Notes in Computer Science, Vol. 1547, 1998, pp. 288–301.