Available online at www.sciencedirect.com
ScienceDirect

Discrete Applied Mathematics 156 (2008) 2855–2866

DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

A covering problem that is easy for trees but **NP**-complete for trivalent graphs

Rolf Bardeli*, Michael Clausen, Andreas Ribbrock

Institut für Informatik III, Universität Bonn, Römerstraße 164, D-53117 Bonn, Germany

Received 24 November 2006; received in revised form 19 October 2007; accepted 27 November 2007
Available online 3 January 2008

Abstract

By definition, a *P2-graph* Γ is an undirected graph in which every vertex is contained in a path of length two. For such a graph, $\mathrm{pc}(\Gamma)$ denotes the minimum number of paths of length two that cover all n vertices of Γ. We prove that $\lceil n/3\rceil \le \mathrm{pc}(\Gamma) \le \lfloor n/2\rfloor$ and show that these upper and lower bounds are tight. Furthermore we show that every connected P2-graph Γ contains a spanning tree T such that $\mathrm{pc}(\Gamma) = \mathrm{pc}(T)$. We present a linear time algorithm that produces optimal 2-path covers for trees. This is contrasted by the result that the decision problem $\mathrm{pc}(\Gamma) \overset{?}{=} n/3$ is **NP**-complete for trivalent graphs. This graph theoretical problem originates from the task of searching a large database of biological molecules such as the Protein Data Bank (PDB) by content.

Keywords: Covering problems; 2-path cover; Edge cover; Optimal tree cover; Tiling problems; Trivalent graphs

1. Introduction

We examine the following graph theoretical problem. Let Γ denote a *P2-graph*, i.e., Γ is a finite undirected graph in which every vertex is contained in a path of length two. For such a graph, $\mathrm{pc}(\Gamma)$ denotes the minimum number of paths of length two that cover all n vertices of Γ. The problem of constructing a minimizer for $\mathrm{pc}(\Gamma)$ arises as a subproblem in a method for searching large molecule databases by content (see Section 2).

Covering problems are plentiful in graph theory. Most prominently, the questions of finding a minimal set of edges that cover all vertices (EDGE COVER) and finding a minimal set of vertices that cover all edges (VERTEX COVER), respectively, are fundamental in graph theory. In contrast to the EDGE COVER problem, which is in **P** [6], most other covering problems are typically found to be **NP**-complete.

Covering with paths of length 2 is related to the EDGE COVER problem: form a dual graph from a given P2-graph by introducing a vertex for each edge and connecting two such vertices by an edge whenever the corresponding edges in the original graph meet at a vertex. Now each path of length 2 corresponds to an edge in the dual. Admittedly, this only shows a relation of the problems up to a certain degree, because a vertex in the original graph may correspond to multiple edges in an optimal edge cover of the dual. On the other hand, the problem under consideration is an

[*] Corresponding author. Fax: +49 228734402.
E-mail address: bardeli@iai.uni-bonn.de (R. Bardeli).

0166-218X/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2007.11.021

instance of the problem of covering graphs with isomorphic subgraphs, which has been shown to be **NP**-complete by Kirkpatrick and Hell [4].

In this paper, we begin with a concise overview of how the graph theoretical problem arises from the task of searching large molecule databases by content. In Section 3 we prove for every P2-graph Γ with n vertices that $\lceil n/3 \rceil \leq \mathrm{pc}(\Gamma) \leq \lfloor n/2 \rfloor$ and show that these upper and lower bounds are tight. If T is a spanning tree of the connected P2-graph Γ then $\mathrm{pc}(T) \geq \mathrm{pc}(\Gamma)$. We prove that such a Γ contains a spanning tree T with $\mathrm{pc}(\Gamma) = \mathrm{pc}(T)$. In Section 4 we present a linear time algorithm that produces optimal 2-path covers for trees. Finally, we show that the decision problem $\mathrm{pc}(\Gamma) \overset{?}{=} n/3$ is **NP**-complete for general P2-graphs as well as for the subclass of trivalent graphs. Trivalent graphs are graphs with no more than three edges incident at any vertex. Completeness results are obtained by reduction from certain tiling problems.

2. A motivating application

We begin by considering the task of searching a large molecule database by content. As an example, the Protein Data Bank [1,8] contains three-dimensional structures of more than 45,000 biological macromolecules such as proteins, nucleic acids, and protein complexes. In a simplified setting, each molecule is represented by a finite attributed graph $M = (V, E, a)$. Here, the elements of the finite vertex set $V \subset \mathbb{R}^3$ describe the centers of the atoms involved in the molecule, the edge set E corresponds to the bonds and $a\colon V \to \mathcal{A}$ denotes a function that specifies the type $a(v) \in \mathcal{A} := \{C, N, O, P, S, \ldots\}$ of the atom located at $v \in V$. In this setting, a molecule is illustrated by a ball and stick model.

As a chemical entity, a molecule will remain unchanged if it is translated or rotated in Euclidean 3-space. More precisely, the special Euclidean group $G = \mathrm{SE}(3)$ of all solid motions acts on such finite attributed graphs via $g(V, E, a) := (gV, gE, ga)$, where $gV := \{gv \mid v \in V\}$, $gE := \{\{gv, gw\} \mid \{v, w\} \in E\}$, and $(ga)(gv) := a(v)$, for all elements $v \in V$ and $g \in G$. Strictly speaking, a *molecule* is a whole G-orbit: $G(V, E, a) := \{g(V, E, a) \mid g \in G\}$. Consequently, each of the *G-equivalent* elements of the G-orbit only *represents* this molecule. Another consequence is a modified notion of containment: let $M = (V, E, a)$ and $M' = (V', E', a')$ be two attributed graphs. By definition, M is *G-contained* in M', for short: $M \subseteq_G M'$, iff there is a solid motion $g \in G$ such that $gV \subseteq V'$, $gE \subseteq E'$, and ga is the restriction of a' to gV. If $g = 1$, then M is said to be *contained* in M', abbreviated $M \subseteq M'$.

Now think of a large molecule database $\mathcal{M}$ consisting of a sequence $\mathcal{M} = (M_1, \ldots, M_N)$ of attributed graphs $M_i = (V_i, E_i, a_i)$ representing N molecules. Suppose a content-based query Q is specified by another attributed graph, $Q = (V, E, a)$, representing parts of a possible molecule. One task is then to find all i such that Q is G-contained in M_i. A more demanding task is to compute all pairs $(g, i) \in G \times [1 : N] := G \times \{1, 2, \ldots, N\}$ such that gQ is contained in M_i. In the sequel, we will focus on this second task, which more formally requires one to compute the set

$$G_{\mathcal{M}}(Q) := \{(g, i) \in G \times [1 : N] \mid gQ \subseteq M_i\}$$

of all *$(G, \mathcal{M})$-matches* with respect to the query Q. More realistic models ask for those (g, i) such that gQ is "near" a Q' contained in M_i. Such tasks go beyond the scope of this motivating section. Interested readers are referred to [7].

In our context, a *barbell* is a connected attributed graph with two vertices. In other words, a barbell is a pair of bonded atoms. Barbells can be viewed as the building blocks of molecules: every molecule is obtained by gluing together suitable barbells. We call $G_{\mathcal{M}}(B) := \{(g, i) \in G \times [1 : N] \mid gB \subseteq M_i\}$ the *$(G, \mathcal{M})$-inverted list* of the barbell B. If $\mathrm{Barb}(Q)$ denotes the set of all barbells contained in Q and if $\mathrm{Barb}(Q)$ *covers* Q, i.e., every element of Q is contained in at least one barbell of Q, it is easy to prove that

$$G_{\mathcal{M}}(Q) = \bigcap_{B \in \mathrm{Barb}(Q)} G_{\mathcal{M}}(B). \tag{1}$$

[In fact, as $\mathrm{Barb}(Q)$ covers Q we have $\bigcup_{B \in \mathrm{Barb}(Q)} B = Q$. Thus, $(g, i) \in G_{\mathcal{M}}(Q)$ iff $gQ \subseteq M_i$ iff $g(\bigcup_{B \in \mathrm{Barb}(Q)} B) \subseteq M_i$ iff $(g, i) \in \bigcap_{B \in \mathrm{Barb}(Q)} G_{\mathcal{M}}(B)$.] Let us look more closely at these inverted lists. A far reaching observation is the fact that inverted lists corresponding to barbells in the same G-orbit are closely related by the *orbit formula*

$$G_{\mathcal{M}}(gB) = G_{\mathcal{M}}(B)g^{-1} := \{(hg^{-1}, i) \mid (h, i) \in G_{\mathcal{M}}(B)\}. \tag{2}$$

Thus for each G-orbit $GB := \{gB \mid g \in G\}$ of barbells one need only compute and store one inverted list. If $G_{\mathcal{M}}(B)$ is stored and if the list $G_{\mathcal{M}}(gB)$ is required in order to compute $G_{\mathcal{M}}(Q)$ along Eq. (1), then one takes the stored list $G_{\mathcal{M}}(B)$ and multiplies it from the right with g^{-1} to obtain $G_{\mathcal{M}}(gB)$, see Eq. (2).

As a consequence, starting with the collection $\mathcal{M}$, it is sufficient to compute in a preprocessing step a maximal set of mutually G-inequivalent barbells contained in the molecules of the database. For each of these barbells, one has to compute and store the corresponding inverted list. But this causes a problem. All the lists contain infinitely many elements as each barbell is stable under all rotations about the axis defined by the barbell. To circumvent this problem, we work with *triplets*, i.e., with two barbells that share one atom. In other words, triplets are paths of length two in an attributed graph. Instead of inverted lists w.r.t. barbells, we now work with inverted lists $G_{\mathcal{M}}(T) := \{(g, i) \in G \times [1 : N] \mid gT \subseteq M_i\}$ corresponding to triplets T. The finiteness of $G_{\mathcal{M}}(T)$ for a non-collinear triplet T results from the fact that the stabilizer subgroup $\{g \in G \mid gT = T\}$ is a cyclic group of order 1, 2, or it is isomorphic to the dihedral group of order 6. Typically, the set $\mathrm{Trip}(Q)$ of all non-collinear triplets in Q covers Q. Similar to Eq. (1), we then obtain

$$G_{\mathcal{M}}(Q) = \bigcap_{T \in \mathrm{Trip}(Q)} G_{\mathcal{M}}(T). \tag{3}$$

Also, the orbit formula $G_{\mathcal{M}}(gT) = G_{\mathcal{M}}(T)g^{-1}$ is valid for triplets T.

In practice, to save space and time, one works implicitly with an approximate set of bonds. To this end, one puts – regardless of chemical reality – a bond between the atom $X \in \mathcal{A}$ located at $v \in \mathbb{R}^3$ and the atom $Y \in \mathcal{A}$ located at $w \in \mathbb{R}^3$ iff the Euclidean distance between v and w is smaller than a positive threshold $\varepsilon_{X,Y}$. Working with such covalent bond radii simplifies both the database and the searching problem. E.g., if $\mathcal{T}$ is a subset of $\mathrm{Trip}(Q)$ such that all vertices of Q are involved in at least one element of $\mathcal{T}$, then

$$G_{\mathcal{M}}(Q) = \bigcap_{T \in \mathcal{T}} G_{\mathcal{M}}(T). \tag{4}$$

Thus, in order to minimize the number of lists to be intersected, we are faced with the problem of finding a smallest family of triplets (= paths of length two) that covers all vertices of the graph. In the present paper we focus on this problem. For readers interested in efficient algorithmic solutions of related problems from multimedia information retrieval using group theory, we refer to [2,3,7].

3. Minimal 2-path covers

Let $\Gamma = (V, E)$ denote an undirected graph with $n := |V|$ vertices. If $\{a, b\}$ and $\{b, c\}$ are different edges in Γ, the corresponding 3-subset $\{a, b, c\}$ of V will be called a *2-path* of Γ. For every family C of 2-paths of Γ we introduce the set $\overline{C} := \bigcup_{\gamma \in C} \gamma$ consisting of all vertices in Γ that are covered by C. If $\overline{C} = V$, we call C a *2-path cover* of Γ. Γ is called a *P2-graph* if it has a 2-path cover. In the sequel, Γ always denotes a P2-graph. For such a Γ we are interested in 2-path covers C of minimum cardinality. All such *optimal* covers have cardinality

$$\mathrm{pc}(\Gamma) := \min\{|C| : C \text{ is a 2-path cover of } \Gamma\}.$$

Example 1. For $n \geq 3$ let $\mathrm{Chain}_n := ([1 : n], \{\{i, i+1\} \mid i \in [1 : n-1]\})$ and $\mathrm{Star}_n := ([1 : n], \{\{1, i\} \mid i \in [2 : n]\})$ denote a *chain* and a *star* with n vertices, respectively. Both types of graphs are P2-graphs. Obviously, if $n = 3m$, then $\{\{3i-2, 3i-1, 3i\} \mid i \in [1 : m]\}$ is a 2-path cover of Chain_n. It is optimal since the involved 2-paths are pairwise vertex-disjoint. Thus $\mathrm{pc}(\mathrm{Chain}_{3m}) = m$. For $m \geq 2$, one easily checks that $\mathrm{pc}(\mathrm{Chain}_{3m}) = \mathrm{pc}(\mathrm{Chain}_{3m-1}) = \mathrm{pc}(\mathrm{Chain}_{3m-2})$. Thus for general $n \geq 3$, we obtain $\mathrm{pc}(\mathrm{Chain}_n) = \lceil n/3 \rceil$. Now let us turn to Star_n, $n \geq 3$. Here, the vertex 1, called the *center* of the star, is involved in every 2-path. Thus, in an optimal 2-path cover of Star_n, the first 2-path covers three new vertices and every subsequent 2-path contributes 2 new vertices, except if n is even. In that case, the last 2-path covers only one new vertex. Hence, for general n, $\mathrm{pc}(\mathrm{Star}_n) = \lfloor n/2 \rfloor$. It is easy to see that all optimal 2-path covers of Star_n are of the type described above. For an illustration see Fig. 1. ◇

The following result shows that the above examples are extreme cases w.r.t. possible pc-values.

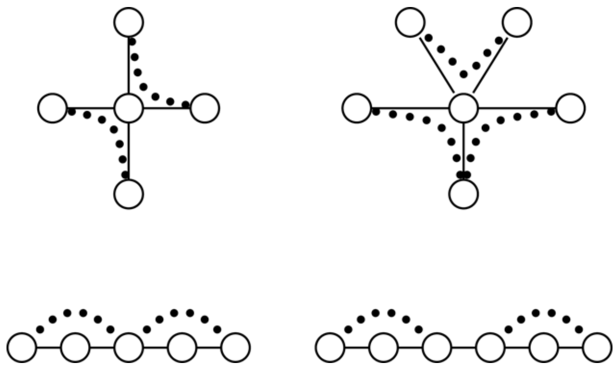


Fig. 1. Optimal 2-path covers of Star_n and Chain_n for $n \in \{5, 6\}$.

Theorem 1. *If $\Gamma = (V, E)$ is a P2-graph with n vertices, then $\lceil \frac{n}{3} \rceil \le \mathrm{pc}(\Gamma) \le \lfloor \frac{n}{2} \rfloor$. Furthermore, both bounds are tight.* $\diamond$

Proof. As $\mathrm{pc}(\Gamma)$ is an integer, it suffices to prove that $n/3 \le \mathrm{pc}(\Gamma) \le n/2$. Now the lower bound follows from the observation that every 2-path in Γ can cover at most three vertices.

We start the proof of the upper bound with a technical remark. If Γ' is a connected subgraph of Γ with only two vertices, then Γ' is not a P2-graph, but it is contained in a connected subgraph Γ^* of Γ with three vertices. Thus it makes sense to define $\mathrm{pc}(\Gamma') := \mathrm{pc}(\Gamma^*) = 1$. Note that the claimed upper bound also holds for Γ': $\mathrm{pc}(\Gamma') = 1 \le 2/2$. We continue with two reductions.

Without loss of generality, we can assume that $\Gamma = (V, E)$ is connected. In fact, if $\Gamma_1 = (V_1, E_1), \ldots, \Gamma_q = (V_q, E_q)$ are the connected components of Γ, $q \ge 2$, then $\mathrm{pc}(\Gamma) = \sum_{i=1}^{q} \mathrm{pc}(\Gamma_i)$. Thus if we can show that for each i, $\mathrm{pc}(\Gamma_i) \le \lfloor |V_i|/2 \rfloor$, then our claim follows.

Without loss of generality, we can assume that the connected P2-graph $\Gamma = (V, E)$ is a tree. In fact, if E' is a subset of E such that $\Gamma' = (V, E')$ is still a P2-graph, then $\mathrm{pc}(\Gamma) \le \mathrm{pc}(\Gamma')$. Thus Γ can be replaced by any of its spanning trees.

Hence, it suffices to prove the claimed upper bound for trees. So, let $\Gamma = (V, E)$ be a tree with $n \ge 3$ vertices. The induction start $n = 3$ is trivial. (By the above remark, this also includes the case $n = 2$ for subtrees.) We prepare for the induction step. For a vertex $v \in V$, let L_v (resp. N_v) denote the set of all leaves (resp. non-leaves) directly linked with v by an edge. We put $\ell_v := |L_v|$ and $n_v := |N_v|$. Note that all $w \in N_v$ have a degree of at least two. Furthermore, the set V' of all non-leaves in Γ is not empty. If V' contains only one element, then Γ is a star and we already know that $\mathrm{pc}(\mathrm{Star}_n) = \lfloor n/2 \rfloor$.

Now we consider the case that Γ contains more than one non-leaf. As Γ restricted to V' is still a tree, T, there is a vertex $v \in V'$ with $\ell_v > 0$ and $n_v = 1$. (In fact, exactly all leaves in T share this property.) Let $N_v = \{w\}$. By deleting the edge $\{v, w\}$, Γ is decomposed into two trees. Each of these two trees contains at least two vertices. Hence, by induction, we are done. This settles the upper bound proof.

The tightness results follow from the pc-values for Star_n and Chain_n. $\square$

How good are spanning subtrees Γ' of Γ at approximating $\mathrm{pc}(\Gamma)$? If $u - v - w$ is a path contributing to a 2-path cover C, then the edges $\{u, v\}$ and $\{v, w\}$ are said to be *involved* in C. Let $E(C)$ denote the set of all edges involved in C and define

$$\varepsilon(\Gamma) := \min\{|E(C)| : C \text{ is an optimal 2-path cover of } \Gamma\}.$$

Lemma 1. *Let C be an optimal 2-path cover of the connected graph $\Gamma = (V, E)$ with a minimal number of involved edges, i.e., $|E(C)| = \varepsilon(\Gamma)$. Then $\Gamma^* = (V, E(C))$ is a cycle-free P2-subgraph of Γ satisfying* $\mathrm{pc}(\Gamma) = \mathrm{pc}(\Gamma^*)$. $\diamond$

Proof. If Γ is isomorphic to a cycle $([1:p], \{\{i, i+1\} \mid i \in [1 : p-1]\} \cup \{\{p, 1\}\})$ of length $p \ge 3$, our claim is obviously true. By way of contradiction suppose that Γ^* contains at least one cycle of length $p \ge 3$, i.e., there exist vertices $v_1, \ldots, v_{p+1} := v_1$ such that $\{v_i, v_{i+1}\} \in E(C)$ for all $i \in [1 : p]$. Among all cycles in Γ^* we take one of minimum length p. Consider the set C' of all 2-paths in C involving at least one edge of the form $\{v_i, v_{i+1}\}$, $i \in [1 : p]$. Then $\overline{C'} = \bigcup_{\gamma \in C'} \gamma$ strictly contains $\{v_1, \ldots, v_p\}$. Otherwise, C' is an optimal 2-path cover of a p-cycle involving all p edges, which is not minimal according to the above remark. Thus let $w \in \overline{C'} \setminus \{v_1, \ldots, v_p\}$. W.l.o.g. let $\{w, v_1, v_2\}$ be in C' and $\{w, v_1\} \in E(C)$. We claim that $\{v_p, v_1\}$ is a redundant edge in $E(C)$. In fact, if $\{v_p, v_1, v_2\}$ is in C', then replace it by $\{v_{p-2}, v_{p-1}, v_p\}$. Let $u \notin \{v_1, \ldots, v_p\}$. If $\{u, v_p\}$ is in $E(C)$ and $\gamma' = \{v_p, v_1, u\}$ is in C', then replace γ' by $\{v_{p-1}, v_p, u\}$. If $\{u, v_1\}$ is in $E(C)$ and $\gamma' = \{v_p, v_1, u\}$ is in C', then replace γ' by $\{u, v_1, v_2\}$. Finally, if $\{v_{p-1}, v_p, v_1\}$ is in C', then replace it by $\{v_{p-2}, v_{p-1}, v_p\}$. As p is minimal, these are all cases to be considered. Thus the edge $\{v_p, v_1\}$ is redundant, contradicting the minimality of $|E(C)|$, thereby proving that Γ^* is cycle-free. □

Theorem 2. *Let Γ be a connected graph with $n \geq 2$ vertices. Then Γ contains a spanning tree Γ_t with* $\mathrm{pc}(\Gamma) = \mathrm{pc}(\Gamma_t)$. ◇

Proof. According to the last lemma, Γ contains a cycle-free P2-subgraph Γ^* with $\mathrm{pc}(\Gamma) = \mathrm{pc}(\Gamma^*)$. As Γ is connected, Γ^* is contained in a spanning tree Γ_t of Γ. Our claim follows from the general observation that for three P2-graphs $\Gamma_i = (V, E_i)$ with $E_1 \subseteq E_2 \subseteq E_3$, we have $\mathrm{pc}(\Gamma_1) \geq \mathrm{pc}(\Gamma_2) \geq \mathrm{pc}(\Gamma_3)$. Hence $\mathrm{pc}(\Gamma) = \mathrm{pc}(\Gamma^*) \geq \mathrm{pc}(\Gamma_t) \geq \mathrm{pc}(\Gamma)$. □

4. Fast construction of optimal 2-path covers for trees

In this section we are going to present a linear time algorithm that computes optimal 2-path covers for trees. We need some preparations. Let $\Gamma = (V, E)$ denote a tree with root r. By definition, the *level* $\lambda(v)$ of a vertex v in Γ is the length of the shortest path connecting r and v. In particular, $\lambda(r) = 0$. Note that all vertices of highest level h are leaves. A vertex v of level $h - 1$ is called *perfect* iff v has the maximum number of children (=leaves) among all vertices of level $h - 1$. As there are vertices of level h, $\ell_v \geq 1$ for every perfect vertex v.

Lemma 2. *Let $\Gamma = (V, E, r)$ denote a rooted tree with $n \geq 3$ vertices. Then every perfect vertex v is directly connected with at most one non-leaf, i.e., $n_v \leq 1$. Furthermore, if $\ell_v = 1$, then $n_v = 1$.* ◇

Proof. If Γ is a star with center v, then $n_v = 0$. So let $\Gamma \not\simeq \mathrm{Star}_n$. Then $\lambda(v) > 0$, and $\lambda(v) = 1$ implies $r \notin L_v$. Thus if w denotes the father of v, then $w \in N_v$, hence $n_v \geq 1$. The level of every $x \in N_v$ differs by one from $\lambda(v)$. If $\lambda(x) < \lambda(v)$, x must be the father of v, thus $x = w$. The case $\lambda(x) > \lambda(v)$ is impossible, otherwise x would be contained in the highest level, which consists purely of leaves, contradicting $x \in N_v$. Hence $N_v = \{w\}$. Finally, if both $\ell_v = 1$ and $n_v = 0$, then Γ has only two vertices in contradiction to $n \geq 3$. □

Theorem 3. *Algorithm P2T, shown as a flowchart in Fig. 2, computes an optimal 2-path cover for every rooted tree with at least three vertices.* ◇

Fig. 4 illustrates a sample run of Algorithm P2T and the constructed optimal 2-path cover.

Proof. On input $\Gamma = (V, E, r)$, consisting of a rooted tree with at least three vertices, the algorithm typically performs several iterations through the flowchart. A new iteration starts upon the re-entry into the input box. At iteration zero, the algorithm is initialized with $\Gamma_0 := \Gamma$ and the empty set of 2-paths $C'_0 := \emptyset$. In general, at iteration i, the algorithm starts with a rooted tree $\Gamma_i = (V_i, E_i, r)$ and a family C'_i of 2-paths specified in the previous iteration. After selecting a perfect vertex v_i in Γ_i, the algorithm specifies V_{i+1}, E_{i+1} and C'_{i+1}. If $|V_{i+1}| \geq 3$, then iteration $i + 1$ takes place. Note that $|V_{i+1}| \leq |V_i| - 2$, thus the algorithm terminates. We prove the correctness of the algorithm with the help of the following.

Loop invariant. For every i with $|V_i| \geq 3$, C'_i is edge-disjoint to every optimal 2-path cover C_i of Γ_i. Furthermore, $C_i \cup C'_i$ is an optimal 2-path cover of Γ.

We prove this loop invariant by induction on i and use $\mathcal{C}_{\mathrm{opt}}(\Gamma)$ as the shorthand for the set of all optimal 2-path covers of Γ. At iteration zero, C'_0 is empty. Thus our claim holds for $i = 0$. For the induction step $i \to i + 1$, we start

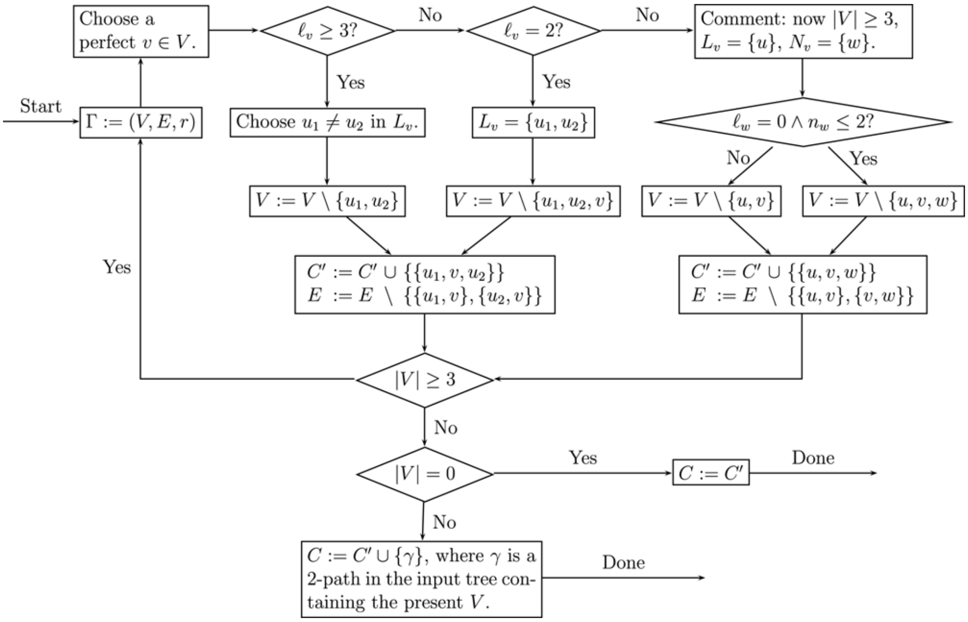

Start
$\Gamma := (V, E, r)$
Choose a perfect $v \in V$.
$\ell_v \geq 3$?
No
$\ell_v = 2$?
No
Comment: now $|V| \geq 3$, $L_v = \{u\}$, $N_v = \{w\}$.
Yes
Choose $u_1 \neq u_2$ in L_v.
Yes
$L_v = \{u_1, u_2\}$
$\ell_w = 0 \wedge n_w \leq 2$?
No
Yes
$V := V \setminus \{u_1, u_2\}$
$V := V \setminus \{u_1, u_2, v\}$
$V := V \setminus \{u, v\}$
$V := V \setminus \{u, v, w\}$
Yes
$C' := C' \cup \{\{u_1, v, u_2\}\}$
$E := E \setminus \{\{u_1, v\}, \{u_2, v\}\}$
$C' := C' \cup \{\{u, v, w\}\}$
$E := E \setminus \{\{u, v\}, \{v, w\}\}$
$|V| \geq 3$
No
$|V| = 0$
Yes
$C := C'$
Done
No
$C := C' \cup \{\gamma\}$, where γ is a 2-path in the input tree containing the present V.
Done

Fig. 2. Algorithm P2T: Optimal P2-covers for trees.

with Γ_i and C'_i. After selection of a perfect vertex $v_i =: v$ in Γ_i, we have to distinguish several cases. By Lemma 2 we know that always $n_v \leq 1$.

Case 1. $\ell_v =: m \geq 3$. If $n_v = 0$, then Γ_i is isomorphic to Star_{m+1}. Let C_i denote an optimal 2-path cover of Γ_i containing the special 2-path $\gamma := \{u_1, v, u_2\}$. By Example 1, such a C_i exists. Now, the algorithm adds γ to C'_i to obtain C'_{i+1}, and subtracts u_1, u_2 from V_i to get V_{i+1}. If V_{i+1} has at least three elements, then $C_{i+1} := C_i \setminus \{\gamma\}$ is an optimal 2-path cover of $\Gamma_{i+1} \simeq \text{Star}_{m-1}$. As $C_i \cup C'_i = C_{i+1} \cup C'_{i+1}$, the right-hand side is an optimal 2-path cover of Γ. Obviously, C_{i+1} can be replaced by any other element in $\mathcal{C}_{\text{opt}}(\Gamma_{i+1})$. This settles the subcase $n_v = 0$.

Now let $L_v = \{u_1, \ldots, u_m\}$ and $N_v = \{w\}$. We claim that there is a $C_i \in \mathcal{C}_{\text{opt}}(\Gamma_i)$ such that $\gamma := \{u_1, v, u_2\}$ is the only 2-path in C_i with a non-empty intersection with $\{u_1, u_2\}$. To check this, start with an arbitrary optimal 2-path cover C_i^* of Γ_i. If C_i^* already contains γ, replace every 2-path in C_i^* of the forms $\{u_i, v, u_j\}$ and $\{u_i, v, w\}$, where $i \in [1 : 2]$ and $j \in [3 : m]$, by $\{w, v, u_j\}$ and $\{u_3, v, w\}$, respectively. This yields a new optimal 2-path cover C_i with the stated properties. If γ is not contained in C_i^*, then u_1 and u_2 are covered by different 2-paths in C_i^*, say, $u_1 \in \gamma_1$ and $u_2 \in \gamma_2$. Several cases have to be considered. If $\gamma_1 = \{u_1, v, u_j\}$ and $\gamma_2 = \{u_2, v, u_k\}$, with $j, k \in [3 : m]$, replace γ_1 and γ_2 by $\gamma'_1 = \{u_1, v, u_2\}$ and $\gamma'_2 = \{u_j, v, u_k\}$ to obtain C_i unless $j = k$. If $j = k$, set $\gamma'_1 := \{u_1, v, u_2\}$ and $\gamma'_2 := \{u_j, v, w\}$. If $\gamma_1 = \{u_1, v, u_j\}$ and $\gamma_2 = \{u_2, v, w\}$, with $j \in [3 : m]$, replace γ_1 and γ_2 by $\gamma'_1 = \{u_1, v, u_2\}$ and $\gamma'_2 = \{u_j, v, w\}$. Finally, if $\gamma_1 = \{u_1, v, w\}$ and $\gamma_2 = \{u_2, v, w\}$, replace γ_1 and γ_2 by $\gamma'_1 = \{u_1, v, u_2\}$ and $\gamma'_2 = \{v, w, x\}$, where $x \neq v$ is a neighbor of w. Note that $w \in N_v$ has a degree of at least two. Hence in all cases we can find a C_i with the stated properties. Now we can proceed as in the subcase $n_v = 0$.

Case 2. $L_v = \{u_1, u_2\}$. If $n_v = 0$, then Γ_i is isomorphic to Chain_3. But $\{\{u_1, v, u_2\}\}$ is the only optimal 2-path cover of Γ_i, thus $C_i = \{\{u_1, v, u_2\}\}$. By the induction hypothesis, $C'_i \cup C_i$ is an optimal 2-path cover of Γ. The algorithm puts $C'_{i+1} = C'_i \cup C_i$ and terminates with $C = C'_{i+1}$, which is correct. This settles the case $n_v = 0$. Now let $N_v = \{w\}$ and assume that $V_{i+1} = V_i \setminus \{u_1, u_2, v\}$ contains at least three elements. Hence there is a 2-path $\{w, x, y\} \subseteq V_{i+1}$ in Γ. We claim that there is a $C_i \in \mathcal{C}_{\text{opt}}(\Gamma_i)$ containing $\gamma := \{u_1, u_2, v\}$ such that γ is edge-disjoint to all other 2-paths in C_i. First of all we show that there is a C_i^* containing γ. In fact, if C_i^* is any optimal 2-path cover of Γ_i not containing γ, then C_i^* must contain the 2-paths $\gamma_1 = \{u_1, v, w\}$ and $\gamma_2 = \{u_2, v, w\}$. Now replace these 2-paths by $\gamma'_1 = \{u_1, v, u_2\}$ and $\gamma'_2 = \{w, x, y\}$. Thus, w.l.o.g., C_i^* contains γ. Every 2-path in C_i^* of the forms $\{u_i, v, w\}$ and $\{v, w, z\}$, with $i \in [1 : 2]$, can be replaced by $\{v, w, x\}$, and $\{w, z, x\}$ or $\{w, z, y\}$, respectively. This shows that there

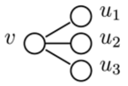
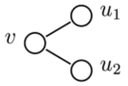
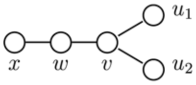
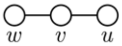
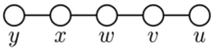
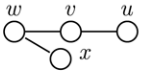
Case	Subcase	Γ_i	$C'_{i+1} \setminus C'_i$	γ
1	$\ell_v = 3,\, n_v = 0$		$\{\{u_1, u_2, v\}\}$	$\{u_1, u_3, v\}$
2	$\ell_v = 2,\, n_v = 0$		$\{\{u_1, u_2, v\}\}$	—
2	$\ell_v = 2,\, n_v = 1$		$\{\{u_1, u_2, v\}\}$	$\{v, w, x\}$
3	$n_w = 1,\, \ell_w = 0$		$\{\{u, v, w\}\}$	—
3	$n_w = 2,\, \ell_w = 0$		$\{\{u, v, w\}\}$	$\{w, x, y\}$
4	$\ell_w = n_w = 1$		$\{\{u, v, w\}\}$	$\{v, w, x\}$

Fig. 3. 2-path γ produced by Algorithm P2T after leaving main loop.

is a $C_i \in \mathcal{C}_{\text{opt}}(\Gamma_i)$ containing $\gamma := \{u_1, u_2, v\}$ such that γ is edge-disjoint to all other 2-paths in C_i. Now we can proceed as in Case 1.

In the remaining cases we can assume that every perfect vertex has exactly one child (=leaf). Furthermore, we can assume $L_v = \{u\}$ and $N_v = \{w\}$.

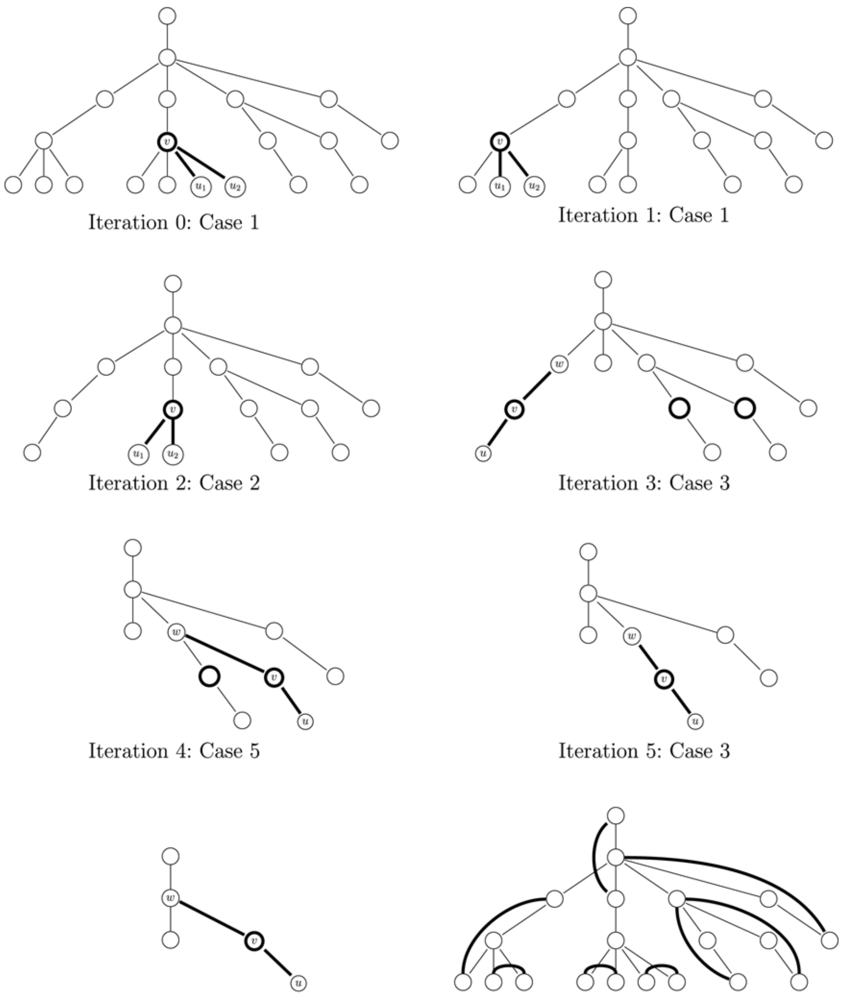
Case 3. $\ell_w = 0$ and $N_w = \{v, x\}$. Suppose that $V_{i+1} = V_i \setminus \{u, v, w\}$ has at least three elements. Let $\{x, y, z\} \subseteq V_{i+1}$ be a 2-path in Γ. In this case, every $C_i^* \in \mathcal{C}_{\text{opt}}(\Gamma_i)$ contains the 2-path $\gamma := \{u, v, w\}$. We claim that there is an optimal 2-path covering C_i of Γ_i in which every other of its 2-paths is edge-disjoint to γ. In fact, if $\{v, w, x\}$ is in C_i^*, then replace it by $\{x, y, z\}$. If $\{w, x, q\}$ is in C_i^*, then replace it by any 2-path in Γ_{i+1} containing x and q. (Note that x and q are neighbors; this follows from $\ell_w = 0$ and $n_w = 2$.) Hence such a C_i exists and we can proceed as in Case 1. If $\ell_w = 0$ and $N_w = \{v\}$, then $\Gamma_i \simeq \text{Chain}_3$ and $|V_{i+1}| = 0$. This subcase will be discussed below.

Case 4. $\ell_w > 0$. Suppose that $V_{i+1} = V_i \setminus \{u, v\}$ has at least three elements. In this case, every $C_i^* \in \mathcal{C}_{\text{opt}}(\Gamma_i)$ contains the 2-path $\gamma := \{u, v, w\}$. We claim that there is an optimal 2-path covering C_i of Γ_i in which every other of its 2-paths is edge-disjoint to $\{u, v\}$ and contains another 2-path involving w. In fact, every 2-path of the form $\{v, w, q\}$ with $q \in L_w \cup N_w$ can be replaced by any 2-path $\gamma' \subseteq V_{i+1}$ in Γ containing w and q. As $\ell_w > 0$, there is a second 2-path involving w. Thus the claimed C_i exists. Again, we can proceed as in Case 1.

Case 5. $\ell_w = 0$, but $N_w = \{v, x_1, \dots, x_s\}$, $s \geq 2$. Note that the levels of all x_i differ by one from $\lambda(w)$. There is at most one x_i that is of smaller level than w. Hence w.l.o.g. x_1 is a perfect vertex. But each perfect vertex has exactly one child. Let y_1 denote the child of x_1. In this case, $V_{i+1} = V_i \setminus \{u, v\}$ has at least three elements and every $C_i^* \in \mathcal{C}_{\text{opt}}(\Gamma_i)$ contains the 2-paths $\gamma := \{u, v, w\}$ and $\gamma' = \{y_1, x_1, w\}$. We claim that there is an optimal 2-path cover C_i of Γ_i in which every other of its 2-paths is edge-disjoint to $\{u, v\}$ and contains another 2-path involving w. In fact, every 2-path of the form $\{v, w, x_j\}$ can be replaced by any 2-path $\gamma^* \subseteq V_{i+1}$ in Γ containing w and x_j. Thus the claimed C_i exists. Again, we can proceed as in Case 1. This completes the proof of the loop invariant.

Now suppose $|V_i| \geq 3$ but $|V_{i+1}| \leq 2$. Then, by the induction hypothesis, we have a family C'_i of 2-paths such that $C'_i \cup C_i \in \mathcal{C}_{\text{opt}}(\Gamma)$ for any optimal 2-path cover C_i of Γ_i. Fig. 3 shows all subcases where this situation can occur. It also shows an optimal 2-path cover of Γ_i computed by Algorithm P2T, when leaving the loop.

Note that the remaining (sub)cases are impossible: Case 4, Subcase $\ell_w \geq 2$ implies $|V_{i+1}| \geq 3$; Case 4, Subcase $\ell_w = 1$ and $n_w \geq 2$ yields $|V_i| \geq 6$ and finally, $n_w \geq 3$ in Case 5 implies $|V_i| \geq 7$. Altogether this proves the correctness of our algorithm. □



Iteration 0: Case 1

Iteration 1: Case 1

Iteration 2: Case 2

Iteration 3: Case 3

Iteration 4: Case 5

Iteration 5: Case 3

Iteration 6: Case 4

An optimal cover

Fig. 4. A sample run of Algorithm P2T and the constructed cover. In each iteration, the perfect nodes and the constructed 2-path are highlighted.

Theorem 4. *With a suitable input format and suitable data structures, Algorithm P2T runs in linear time.* ◇

Sketch of Proof. Let $\Gamma = (V, E, r)$ denote the rooted tree in question. Order the non-leaves of Γ lexicographically, first w.r.t. their level, then within the level from left to right. Suppose $v_1, \ldots, v_m$ are the non-leaves in this ordering. Let $v_{m+1}, \ldots, v_n$ denote the remaining vertices of Γ. For $i \in [1 : m]$ associate to v_i the sequence

$$[v_i] := (i; \lambda_i; \ell_i; L_i; n_i; j_i),$$

where λ_i denotes the level of v_i, ℓ_i is the number of leaves directly linked with v_i, L_i is a repetition-free sequence of all elements of L_{v_i}, $n_i := n_{v_i}$, and v_{j_i} denotes the father of v_i (or is undefined, if v_i is the root). Furthermore define

$$\Lambda_j^1 := \{i \in [1 : m] \mid \lambda_i = j \wedge \ell_i \leq 1\} \quad \text{and} \quad \Lambda_j^2 := \{i \in [1 : m] \mid \lambda_i = j \wedge \ell_i \geq 2\}.$$

For the input corresponding to $\Gamma = (V, E, r)$ Algorithm P2T expects $[v_1], \ldots, [v_m]$ as well as all Λ_j^1 and Λ_j^2. In particular, we assume Λ_j^1 and Λ_j^2 to be presented as lists ordered such that indices of nodes with a higher number of leaves come before those with a smaller number of leaves. Note that the order in which Algorithm P2T performs the cases $\ell_v \geq 3$ and $\ell_v = 2$ does not matter. We define a partial priority on the set of all vertex IDs by

$$\Lambda_{i+1}^2 > \Lambda_{i+1}^1 > \Lambda_i^2 > \Lambda_i^1$$

for all i. I.e., nodes of level $i+1$ are always of higher priority than nodes of level i and, on a given level, nodes with more leaves always have a higher priority than nodes with less leaves. The algorithm always executes a $[v_i]$ of highest priority. From the ordering of Λ_i^1 and Λ_i^2 it is clear that perfect nodes are always of highest priority.

Case 1. If $\ell_i \geq 3$ and u_1, u_2 are the first two elements of L_i, then Algorithm P2T adds the 2-path $\{v_i, u_1, u_2\}$ to C', sets $[v_i] := (i; \lambda_i; \ell_i - 2; L_i \setminus (u_1, u_2); n_i; j_i)$, and, if the new ℓ_i equals 1, deletes i from $\Lambda_{\lambda_i}^2$ and inserts i into $\Lambda_{\lambda_i}^1$.

Case 2. If $\ell_i = 2$, $j_i := k$, and $L_i = (u_1, u_2)$, then P2T adds the 2-path $\{v_i, u_1, u_2\}$ to C', deletes i from $\Lambda_{\lambda_i}^2$, and updates $[v_k]$ in the fourth component: $n_k := n_k - 1$. This completes the updating phase unless $\ell_k = 0$ and the new n_k equals 1. In this case, v_k is now a leaf, which accounts for additional update operations. At first, if v_p is the father of v_k, then $n_p := n_p - 1$, $\ell_p := \ell_p + 1$ and k is appended to L_p. Furthermore, k is implicitly deleted from $\Lambda_{\lambda_k}^1 \cup \Lambda_{\lambda_k}^2$ by deleting the list $[v_k]$.

The remaining cases are similar and left to the reader. Altogether it shows that every update operation takes only a constant number of elementary steps, thus the overall complexity is linear in the number of vertices. □

5. Disjoint 2-path covers and tilings

In this section, we examine the complexity of the decision problem, whether a P2-graph Γ with n vertices has a vertex-disjoint 2-path cover, i.e., $n/3 \overset{?}{=} \mathrm{pc}(\Gamma)$. In the sequel we will call this problem Disjoint 2-Path Cover. It is easily seen to be **NP**-complete by the following theorem by Kirkpatrick and Hell [4].

Theorem 5. *For a graph G let* Part*[G] be the decision problem asking whether a given graph H admits a G-partition, i.e., if there are subgraphs $G_1, G_2, \ldots, G_d$ of H such that each G_i is isomorphic to G and the vertex sets of the G_i form a partition of the vertex set of H. Then* Part*[G] is* **NP***-complete iff G has a connected component of at least three vertices.* ◇

Choosing $G = \mathrm{Chain}_3$, we obtain

Corollary 2. Disjoint 2-Path Cover *is* **NP***-complete for general P2-graphs.* ◇

Thus on the one hand, we have a linear time algorithm that constructs optimal 2-path covers for trees. On the other hand, we know that Disjoint 2-Path Cover is **NP**-complete. This poses the question whether there are simple types of graphs for which the Disjoint 2-Path Cover problem remains **NP**-complete. In this respect we will now examine classes of graphs with bounded vertex order and ask for vertex-disjoint 2-path covers of graphs from a given class only.

Theorem 6. *Let G_i denote the class of connected graphs with maximal degree of i. Then the following holds.*

(1) *For the class G_2,* Disjoint 2-Path Cover *is in* **P**.
(2) *For the class G_i, $i \geq 3$,* Disjoint 2-Path Cover *is* **NP***-complete.*

Proof of (1). Note that the class G_2 only consists of circles and chains. By the constructions in Section 3, our claim follows. □

We prepare for the proof of (2). To this end, we will first sketch a proof that Disjoint 2-Path Cover is **NP**-complete for the class G_4. This is done by reduction from the tromino tiling problem. The latter can be shown to be **NP**-complete by a construction due to Moore and Robson [5]. This construction carries over to a certain tiling problem of triangular grids,[1] which eventually leads to the proof that Disjoint 2-Path Cover is **NP**-complete for the class G_3. We will first sketch the proof for the class G_4. Then we will introduce a tiling problem for triangular grids and its reduction to Disjoint 2-Path Cover for the class G_3. Finally we will show that the tiling problem is **NP**-complete using a construction similar to that by Moore and Robson.

[1] In the following, *grids* are allowed to have holes.

Let us have a look at the class G_4. We consider the problem of tiling grids of squares with trominos. A *grid of squares* is a connected set of congruent squares in the plane, parallel to the axes, such that two squares meet at one corner or have a common side or do not meet at all. A *tromino* is a grid of squares composed of exactly three squares. Allowing trominos to be rotated by multiples of 90 degrees, there are exactly two types of trominos. One with all squares in a line, which we call the *straight tromino*, and one forming a right angle, called the *right tromino*. The TROMINO TILING problem poses the question whether a given grid of squares can be partitioned by trominos. This problem reduces to DISJOINT 2-PATH COVER for the class G_4 in the following way. Construct a graph from a grid of $n = 3m$ squares by introducing one vertex for each square. Connect two vertices whenever the corresponding squares are adjacent. Note that paths of length two are in correspondence with trominos. In this way, there is a vertex-disjoint 2-path cover with cardinality $n/3$ iff the grid is tileable by trominos.

In order to show **NP**-completeness for the TROMINO TILING problem, Moore and Robson use a reduction from CUBIC PLANAR MONOTONE 1-IN-3 SATISFIABILITY. This is a restriction of the well-known SAT problem to formulas where each clause contains three variables. Such a formula is not only required to be satisfiable but also exactly one variable in each clause has to be true (1-in-3). Negation of variables is not allowed (monotone) and each variable has to appear exactly three times in a formula (cubic). Finally, the representation of a formula by a graph has to be planar. This representation is obtained by introducing two kinds of vertices: one for each clause and one for each variable. Each variable is connected to each clause in which it appears by an edge. Moore and Robson construct grids from these graphs that are tileable by right trominos iff the formula is satisfiable. The construction works by designing grids for variable vertices, clause vertices, and edges. These grids are subgrids of 6-by-6 grids that can be combined in order to build grids representing formulas of the above type. The construction is used to show that tiling by the right tromino alone is **NP**-complete, but is easily seen to work unchanged if the straight tromino is allowed in addition.

We will now consider a tiling problem for the class of triangular grids. By definition, a *triangular grid* is a set of congruent equilateral triangles in the plane such that each triangle has one side parallel to the horizontal axis, two triangles either do not intersect or they share a side or meet at a corner, and each pair of triangles can be connected by a path inside the grid crossing only sides where two triangles meet. The number of triangles in a grid is called its *order*. We are interested in tiling triangular grids by *tiles* that are triangular grids of order three. TILING TRIANGULAR GRIDS is the decision problem whether a given triangular grid admits a partition by copies of these tiles. For an illustration, see Figs. 5–7. Of course such tiling problems cannot have a solution if the order of the grid is not divisible by three, so we will restrict the problem to grids with order divisible by three. This tiling problem reduces to DISJOINT 2-PATH COVER of G_3 in the following way. For a given triangular grid, construct a graph with one vertex for each triangle in the grid. Connect two vertices by an edge iff the corresponding triangles share a common side. Obviously, the graph is in G_3. Now, there is a tiling of the given triangular grid of order n iff the graph admits a 2-path cover with vertex-disjoint paths, i.e., iff there is a 2-path cover of cardinality $n/3$.

Proof of (2). We will now show that TILING TRIANGULAR GRIDS is **NP**-complete. As in the construction by Moore and Robson, we provide triangular grids representing variables, clause templates, and connections between variables and clause templates. In this way, each Cubic Planar Monotone 1-in-3 SAT formula can be transformed into a triangular grid that is tileable in the above sense iff the formula is satisfiable. Each variable and clause template has three docking points where connections are allowed. Variable templates are constructed in such a way, that either all or none of its docking points are covered by tiles from within the template, thus representing truth values. Similarly, the construction of clause templates forces exactly one of its docking points to be covered by tiles from within the template.

Fig. 5(a) shows a triangular grid T_x representing a variable x. There are two classes of tilings for this grid with *inner tiles*, one incorporating two tiles that leaves the corners uncovered and one with three tiles that covers the whole grid. If the corners are not covered by inner tiles, the variable represents the value *true*, otherwise it represents the value *false*. Having three corners for the representation of the value corresponds to the restriction that each variable has to occur exactly three times in a formula.

Fig. 5(b) shows a grid representing a clause template. It consists of an inner ring and three docking points. Whenever this structure is tiled in a way that all triangles of the inner ring are covered, exactly one of the docking points is covered. This corresponds to the one variable in a clause that is true.

Fig. 6 shows how variables can be connected to clause templates. These connections are constructed in such a way that corners of variables are connected to docking points of clause templates. Each connection enforces that a tiling of

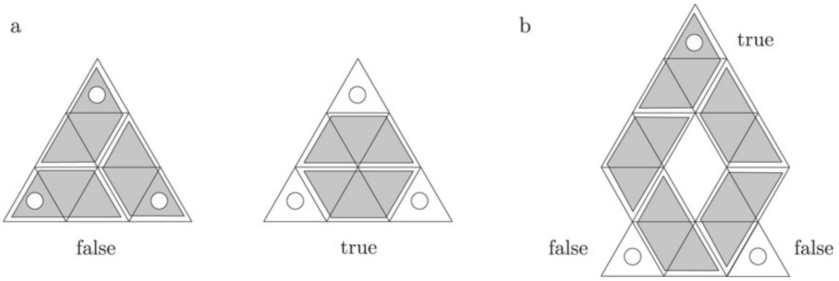


Fig. 5. Triangular grids representing variables and clause templates.

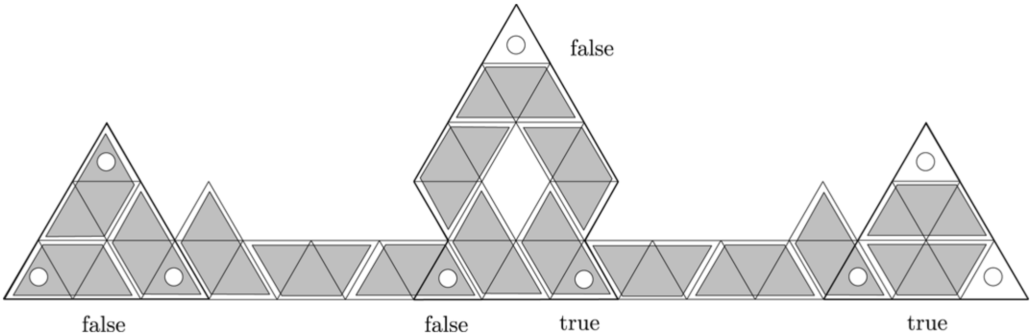


Fig. 6. Connecting variables and clauses.

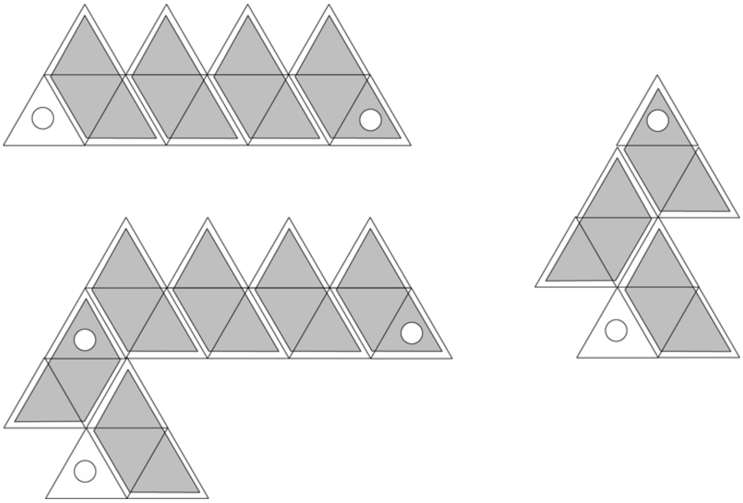


Fig. 7. Arbitrary length horizontal and vertical connections. Combining both, connections can be bent around corners.

the connection either covers a corner of a variable or a docking point of a clause template. In this way, the propagation of truth values from variables to clause templates is possible. Moreover, Fig. 7 shows how arbitrary length horizontal and vertical connections can be constructed and how connections can be bent around corners.

Thus each Cubic Planar Monotone 1-in-3 SAT formula can be transformed into a triangular grid that is tileable iff the formula is satisfiable. Transforming grids into graphs as described above shows that the DISJOINT 2-PATH COVER problem is **NP**-complete even for the restricted subclass of G_3 that corresponds to triangular grids. This completes the proof of Theorem 6. □

6. Conclusions and future work

For the class of P2-graphs we have derived tight lower and upper bounds for the minimum cardinality of 2-path covers. We have presented a linear time algorithm that computes such optimal covers for trees. On the negative side, we have shown that the computation of optimal covers is **NP**-complete even for planar trivalent graphs. Are there classes of graphs, different from trees and circles, for which DISJOINT 2-PATH COVER is in **P**?

Covering graphs with paths of length two is a special case of a whole family of related problems. One can ask for optimal coverings using paths of a given length k. For $k = 1$ the EDGE COVER problem is obtained, which is known to be in **P** by the work of Norman and Rabin [6]. Besides the paper by Kirkpatrick and Hell [4], we are not aware of any specific work for $k > 2$.

A brute force way of applying our results for trees to general graphs would be to construct perfect coverings for all spanning trees of a given graph using Algorithm P2T and then choosing a covering of minimum cardinality. In light of Cayley's Formula stating that a complete graph has n^{n-2} spanning trees and Kirchhoff's Theorem giving the number of spanning trees as $\frac{1}{n}$ times the product of the non-zero eigenvalues of its admittance matrix, this will not result in a desirable algorithm. Nevertheless, there might be good heuristics for choosing a small number of spanning trees resulting in an almost optimal covering.

References

[1] H.M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T.N. Bhat, H. Weissig, I.N. Shindyalov, P.E. Bourne, The protein data bank, Nucleic Acids Research 28 (2000) 235–242.

[2] M. Clausen, F. Kurth, A unified approach to content-based and fault tolerant music recognition, IEEE Transactions on Multimedia 6 (5) (2004) 717–731.

[3] M. Clausen, F. Kurth, Content-based information retrieval by group theoretical methods, in: J. Byrnes (Ed.), Computational Noncommutative Algebra and Applications, NATO Science Series, II. Mathematics, Physics and Chemistry 136 (2004) 29–55.

[4] D.G. Kirkpatrick, P. Hell, On the complexity of general graph factor problems, SIAM Journal on Computing 12 (1983) 601–609.

[5] C. Moore, J.M. Robson, Hard tiling problems with simple tiles, Discrete & Computational Geometry 26 (4) (2001) 573–590.

[6] R.Z. Norman, M.O. Rabin, An algorithm for a minimum cover of a graph, Proceedings of the American Mathematical Society 10 (1959) 315–319.

[7] A. Ribbrock, M. Clausen, Content-based Search in Large Protein Databases (in preparation).

[8] Portal website of the Protein Data Bank (PDB). http://www.rcsb.org/pdb/.